DevTechs Documentation

Gabriel

Jun 14, 2020

Ansible

CHAPTER 1

Ansible

CHAPTER 2

Git y GitHub

CHAPTER 3

Git & GitHub Lessons

Basado en: Git and GitHub LiveLessons - Workshop (by Peter Bell)

Temas principales:

- Cómo funciona Git y cómo usarlo efectivamente para guardar el historial de los cambios hechos en un proyecto.
- Cómo usar GitHub para colaborar con otras personas.

Lessons:

1. Lesson 1: Configuring Git, Three Stage thinking
2. Lesson 2: Getting Started with Git
3. Lesson 3: Getting Started with GitHub
4. Lesson 4: Files in Git: renaming, deleting, ignoring
5. Lesson 5: Branching, Merging, Rebasing
6. Lesson 6: Git Internals
7. Lesson 7: Collaborating via GitHub
8. Lesson 8: Reviewing a Project on GitHub
9. Lesson 9: Configuring a Project on GitHub
10. Lesson 10: Tags and Releases
11. Lesson 11: How to undo almost everything using Git

3.1 Lesson 1

Lesson 1: Configuring Git, Three Stage thinking

Table of Contents

3.1.1 Three levels of configuration

Hay 3 niveles de configuración en Git: Local, Global y System:

- La mayoría de las veces solo usaremos configuraciones a nivel global. **Global** configura cosas de todos los repositorios que usamos, logueados con nuestro usuario en nuestra PC.
- **Local** solo funciona específicamente en un solo repo. Es para hacer configuraciones específicas de solo ese repositorio en el que estamos trabajando (`--local`).
- El tercer nivel de configuración menos usado es la opción de configuración `--system`. Es para cualquier otra persona que se loguee en nuestra computadora. Si tenemos múltiples personas logueandose en una misma computadora usando distintas cuentas de usuario, es ahí donde usaríamos una opción de configuración de nivel **System**.

3.1.2 Basic Configuration settings

El comando principal de configuración en Git es `git config`.

Para realizar una configuración a nivel Global usamos: `git config --global`. Por ejemplo:

```
$ git config --global user.name
```

Este comando nos retorna el username si es que tenemos configurado alguno. Si no hay ningún username configurado no retornará nada.

Con el mismo comando podemos ingresar el nombre por el cual queremos ser conocidos:

```
$ git config --global user.name "Nombre Apellido"

$ git config --global user.name
Nombre Apellido
```

Podemos usar el comando `git config` como *getter* para obtener información o como *setter* para configurar.

Podemos hacer lo mismo para el correo del usuario:

```
$ git config --global user.email username@example.com

$ git config --global user.email
username@example.com
```

Podemos usar el comando `git config` para listar las configuraciones globales:

```
$ git config --global --list
user.name=Nombre Apellido
user.email=username@example.com
```

Las configuraciones globales se guardan en un archivo *~/.gitconfig*:

```
$ cat ~/.gitconfig
[user]
        name = Nombre Apellido
        email = username@example.com
```

Podemos editar el archivo:

```
$ vi ~/.gitconfig
```

Podemos listar toda la configuración global:

```
$ git config --global --list
user.name=Nombre Apellido
user.email=newuser1@mail.com
```

La siguiente configuración sirve para colorear el output de ciertos comandos:

```
$ git config --global color.ui true
```

Note: Si estamos usando una versión de Git 1.8.4 o más nueva esta opción estará por defecto.

3.1.3 Configuring line endings

Una de las configuraciones más importantes es 'auto carriage return line feed'.

STACKOVERFLOW - Carriage Return _ Linefeeds _ Form Feeds

`autocrlf` - Auto Carriage Return Line Feed

```
# Linux/MAC
$ git config --global core.autocrlf input

# Windows
$ git config --global core.autocrlf true
```

Cuando alguien está haciendo `commit` a un proyecto, y se ve como si hubiese cambiado todas las líneas en 2 archivos diferentes. Pero cuando vemos los archivos, el contenido parece el mismo. Lo que pudo haber pasado es que estaba usando un editor particular de Windows que agregaba caracteres escondidos: **Carriage Returns** y **Line Feeds**.

Esto es negativo porque se pensará que se han cambiado líneas en las que en realidad el editor no ha cambiado el código. Y tampoco sabremos los cambios que se han hecho realmente.

Por eso, es importante configurar el manejo de Auto Carriage Return Line Feed. La forma en que trabaja es que en la mayoría de sistemas **Linux/MAC** se tiene un Line Feed al final de cada línea y en **Windows** tenemos 2 caracteres escondidos: Line Feed y Carriage Return.

En Linux:

```
# Linux/MAC
$ git config --global core.autocrlf input
```

Cuando guardemos un archivo en git o en el repositorio, se eliminará todos los Carriage Returns y solo dejará Line Feeds. Esto es porque podemos haber descargado un archivo que proviene de Windows con caracteres Carriage Returns escondidos.

En Windows:

```
# Windows
$ git config --global core.autocrlf true
```

Para cualquier archivo que pongamos en un repositorio, se extraerán los Carriage Returns. Pero cuando extraigamos archivos del repo, pondrá los Carriage Return de vuelta, para poderlos editar en editores como Notepad.

3.1.4 Configuring aliases

Los alias en Git permiten crear nuestros propios comandos, especificando varias opciones, de forma que facilite y agilice el trabajo.

- Creemos nuestro primer alias:

```
$ git config --global alias.s "status -s"
```

El nombre del alias es `s`. De forma que en el futuro cuando escribamos `git s`, correrá el comando que está dentro de las comillas `"status -s"`. Con el parámetro `-s`, `git status` correrá en modo silencioso.

- Creemos otro alias:

```
$ git config --global alias.lg "log --oneline --all --graph --decorate"
```

El alias llamdo `lg` correrá el comando `git log` con estas opciones:

- `oneline` - poner el output del historial o el log en una sola línea.
- `--all` - mostrar todos los branches. No solo el historial del branch donde estamos actualmente.
- `--graph` - queremos que se muestra en forma gráfica
- `--decorate` - mostrar información adicional.

—

3.2 Lesson 2

Lesson 2: Getting Started with Git

Table of Contents

Lesson 2: Getting Started with Git

3.2.1 Creating your first Git repository

Inicialmente al tipear el comando `git status` un error nos indica que no estamos en un repositorio:

```
$ git status
fatal: not a git repository (or any of the parent directories): .git
```

Para crear un nuevo repositorio en Git llamado `app1` usamos:

```
$ git init app1
Initialized empty Git repository in /home/user/app1/.git/
```

No hemos necesitado conexión a Internet para que este comando funcione. Hemos creado una **copia local** de un repositorio. Podemos usar branches, tags, ver el historial con esta copia de Git. Luego podemos decidir compartirlo con un grupo, y es ahí donde necesitamos conectarnos a un servidor remoto como **Github**.

Podemos cambiar el directorio a nuestro repositorio creado y ver su estado:

```
$ cd app1/

$ git status
On branch master

No commits yet

nothing to commit (create/copy files and use "git add" to track)
```

- `On branch master` - nos dice que estamos en el branch master (branch por defecto).
- `No commits yet` - nos dice que hemos creado un repositorio pero no hemos guardado, todavía no hemos hecho nuestro "Initial commit".
- `nothing to commit` - nos dice que no hay nada para hacerle commit. No hay archivos que podamos añadir al historial.

3.2.2 Committing in Git

En Git se tiene la idea de pensamiento en 3 etapas (3 stage thinking):

Crear un archivo de prueba sin contenido:

```
$ touch index.html
```

Veamos el estado del repositorio:

```
$ git status
On branch master

No commits yet

Untracked files:
(use "git add <file>..." to include in what will be committed)
```

(continues on next page)

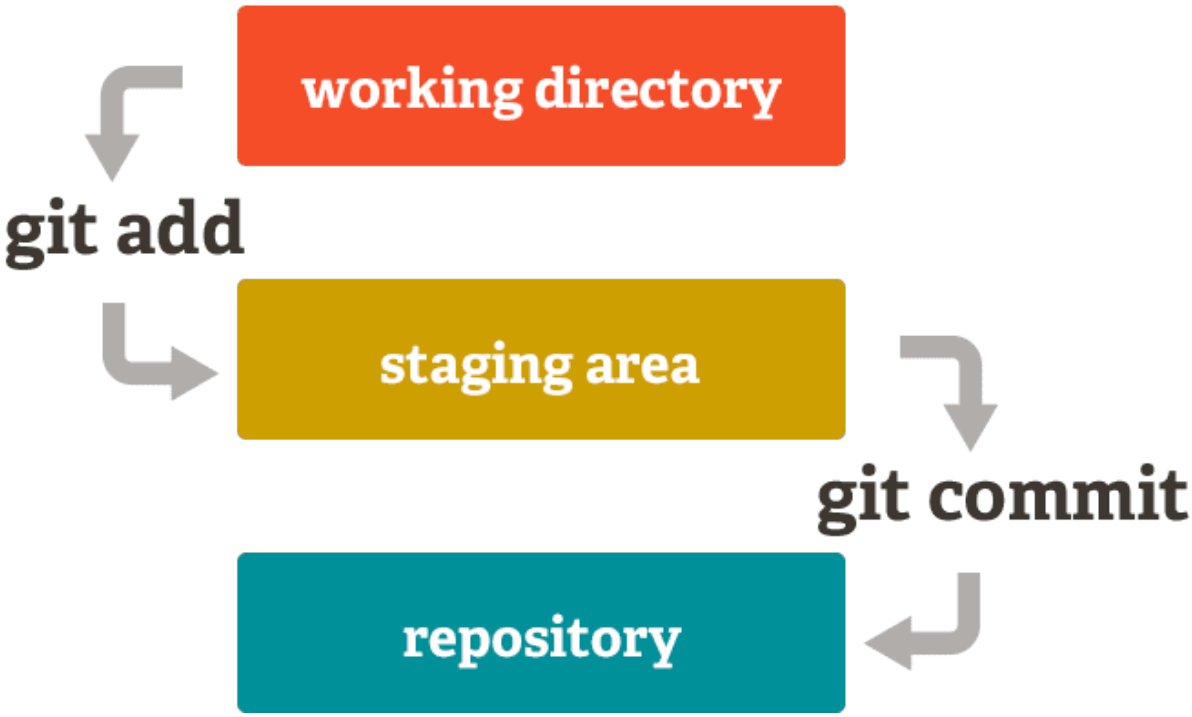


Fig. 1: 3 stage thinking de Git

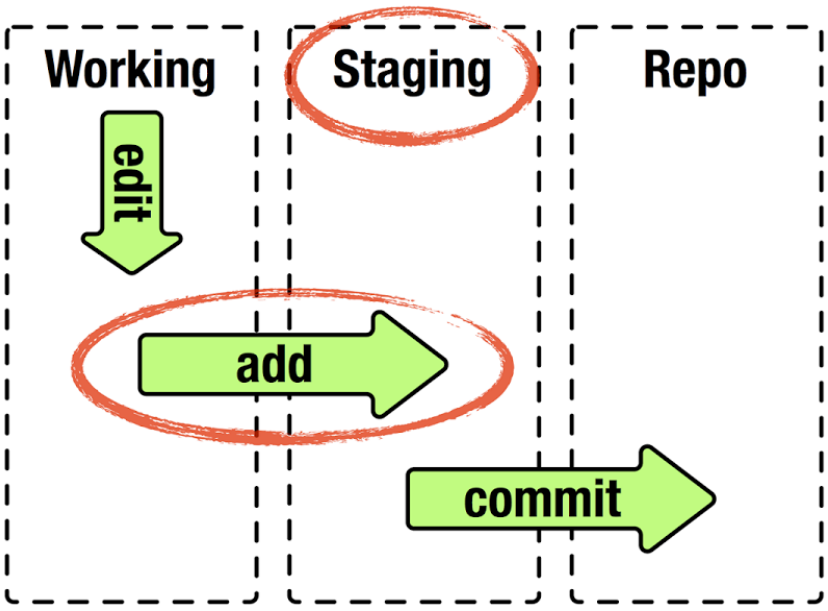


Fig. 2: 3 stage thinking de Git diagram

(continued from previous page)

```
        index.html

nothing added to commit but untracked files present (use "git add" to track)
```

Todavía no hemos hecho un `commit`, es decir, no hemos guardado nada en el historial.

Nos ha salido un mensaje de `Untracked files`. Es decir, existen archivos que no hemos notificado a Git dentro del repo. Hay 2 cosas que le diremos que haga:

1. Le diremos que agregue el archivo a "**staging area**". Y usamos el comando `git add`:

- Una opción es añadir solamente el archivo `index.html`:

```
$ git add index.html
```

- Otra opción es agregar todos los archivos al repositorio usando un punto (`.`):

```
$ git add .
```

Una vez que hayamos puesto cualquiera de los dos comandos anteriores, veremos que Git nos dice que está listo para guardar el archivo en historial:

```
$ git status
On branch master

No commits yet

Changes to be committed:
(use "git rm --cached <file>..." to unstage)

        new file:   index.html
```

Esta sección de "staging area" es como una página web de compras, donde ponemos objetos en el carrito de compras, pero esto no quiere decir que nos lo vayan a enviar. La página esperará hasta que hagamos clic en el botón de comprar. Podemos agregar o quitar productos, esto nos permite hacer cambios antes de la compra. Para el "staging area" en Git es similar.

Creemos otro archivo de prueba:

```
$ touch index.css
```

Veamos el estado del repositorio:

```
$ git status
On branch master

No commits yet

Changes to be committed:
(use "git rm --cached <file>..." to unstage)

        new file:   index.html

Untracked files:
(use "git add <file>..." to include in what will be committed)

        index.css
```

Agreguemos el nuevo archivo que está "Untracked":

```
$ git add .
```

Y vemos que tenemos los dos archivos agregados al repositorio:

```
$ git status
On branch master

No commits yet

Changes to be committed:
(use "git rm --cached <file>..." to unstage)

        new file:   index.css
        new file:   index.html
```

2. El siguiente paso es hacer el `commit`:

Cuando hagamos un `commit` se hará un `commit` de ambos archivos juntos:

```
$ git commit -m "Added home page"

[master (root-commit) 49ab535] Added home page
2 files changed, 0 insertions(+), 0 deletions(-)
create mode 100644 index.css
create mode 100644 index.html
```

Con el parámetro `-m` se agrega un mensaje al `commit`.

```
$ git status
On branch master
nothing to commit, working tree clean
```

El mensaje nos dice que cualquier trabajo que hemos hecho se ha guardado satisfactoriamente en Git, o en el historial. Podremos volver a este punto en el tiempo.

Generalmente este mensaje es lo que queremos ver al correr `git status`, sin importar el branch donde estemos trabajando.

3.2.3 Understanding Git Commit

Al hacer nuestro primer `commit` obtuvimos el siguiente mensaje:

```
$ git commit -m "Added home page"

[master (root-commit) 49ab535] Added home page
2 files changed, 0 insertions(+), 0 deletions(-)
create mode 100644 index.css
create mode 100644 index.html
```

- `master` - nos dice que estamos en el branch `master`.
- `root-commit` - es el primer `commit`. Solo lo veremos una vez por proyecto.
- `49ab535` - son los primeros dígitos del "**commit HASH**", es decir, el identificador único del `commit`.

Note: ¿Por qué no se usan números incrementales para los "commits": `commit-1`, `commit-2`, `commit-3`, ... ?:

La razón es que cuando trabajamos con Git estamos trabajando con un sistema de control distribuido. Si un repositorio estuviese colgado en Github y alguien descargara una copia, ambos estaríamos haciendo `commits` de nuestras propias copias del repositorio sin conectarnos a Internet. Git no tiene idea de quién hizo `commit-3` o `commit-4` porque varios tienen sus copias y hacen `commit` localmente. Por eso se usan **identificadores globales únicos**, un **HASH SHA-1** que identifican singularmente un `commit`.

- `2 files changed` - pues hemos hecho `commit` a 2 archivos.
- `0 insertions(+), 0 deletions(-)` - esto es inusual, pero se debe a que los archivos están en blanco. Si hubiese 3 líneas de texto en un archivo y 2 en el otro archivo, tendríamos `5 insertions` y `0 deletions`.
- `create mode 100644` - Solo podremos ver 3 números como este: 1 para links simbólicos, 1 para ejecutables y 1 para todo lo demás.
- `create mode 100644 index.css` - archivo agregado a historial.

3.2.4 The benefits of the staging area

¿Por qué tenemos que añadir archivos a "staging area" y luego hacer commit?:

Porque Git está diseñado para ayudarnos a hacer un correcto control de versiones. Y una parte de hacer un buen control de versiones es hacer `commits` atómicos con sentido.

Tip: Una de las mejores prácticas en Git y Github es escribir mejores mensajes de `commit` y ver qué archivos poner juntos. Los `commits` nos deben decir la historia del trabajo que hemos realizado.

Hagamos un ejemplo de cómo nos complicaría no tener una `staging area`.

- Creemos varios archivos de prueba:

```
$ touch about.html about.css contact.html contact.css
```

- Listar el contenido del directorio:

```
$ ls
about.css  about.html  contact.css  contact.html  index.css  index.html
```

- Ver el estado del repositorio:

```
$ git status

On branch master
Untracked files:
(use "git add <file>..." to include in what will be committed)

        about.css
        about.html
        contact.css
        contact.html

nothing added to commit but untracked files present (use "git add" to track)
```

Vemos que tenemos a los 4 archivos agregados sin trackear.

- Usar el alias creado en la sección *Configuring aliases* de *Lesson 1* para ver el estado del repositorio en modo silencioso (`git config --global alias.s "status -s"`):

```
$ git s

?? about.css
?? about.html
?? contact.css
?? contact.html
```

`git status` toma mucho espacio, es útil pero la mayoría de las veces sabemos el estado de los archivos. Así que podemos saber el estado de forma resumida con el modo silencioso.

Un color rojo de los signos de interrogación nos indica que los archivos no han sido agregados al `staging area`. Doble signo de interrogación `??` nos indica que son archivos nuevos sin trackear.

- Podemos añadir al repositorio los 4 archivos en dos partes para mantener el orden (las páginas de `about` y `contact`). Primero para la página `about` usamos:

```
$ git add ab*

$ git s
A  about.css
A  about.html
?? contact.css
?? contact.html
```

Habremos agregado 2 nuevo archivos (`about.css`, `about.html`) al repositorio. Al ejecutar `git s`, un color verde en las letras `A` nos indica que han sido `staged`, es decir, agregados al repositorio.

- Luego hacemos un `commit` con un mensaje descriptivo:

```
$ git commit -m "Added about us page"

[master aa2d60c] Added about us page
2 files changed, 0 insertions(+), 0 deletions(-)
create mode 100644 about.css
create mode 100644 about.html
```

- Podemos hacer el mismo procedimiento para la página `contact`:

```
$  git s

?? contact.css
?? contact.html
```
```

```
$ git add .

$ git s
A  contact.css
A  contact.html
```

```
$ git commit -m "Added contact us page"

[master 4388cdf] Added contact us page
2 files changed, 0 insertions(+), 0 deletions(-)
create mode 100644 contact.css
create mode 100644 contact.html
```
```

3.2.5 Git log for viewing history

Podemos ver el historial usando el alias creado en la sección *Configuring aliases* de *Lesson 1* (`git config --global alias.lg "log --oneline --all --graph --decorate"`)

- `--oneline` - imprime cada `commit` en una sola línea
- `--decorate` - nos da información de los HEAD de los branches.
- `--graph` - imprime estrellas al lado del identificador del *commit* que ayudan en el trabajo con branches.

Primero veamos el log con el comando por defecto:

```
$ git log
commit 4388cdfb26762db96df7fe845d09f6dbca7c5257 (HEAD -> master)
Author: Nombre Apellido <newuser1@mail.com>
Date:   Thu Sep 26 23:40:51 2019 -0500

    Added contact us page

commit aa2d60c167b1d066771d871e14616f1ff89fcaef
Author: Nombre Apellido <newuser1@mail.com>
Date:   Thu Sep 26 23:38:36 2019 -0500

    Added about us page

commit 49ab5355d7bc3e8ffed85cd6f9c7b491007ecfaa
Author: Nombre Apellido <newuser1@mail.com>
Date:   Wed Sep 25 23:23:45 2019 -0500

    Added home page
```

Si tuviesemos muchos más `commits` nos tomaría mucho tiempo leer todo. Así que podemos usar el alias:

```
$ git lg

* 4388cdf (HEAD -> master) Added contact us page
* aa2d60c Added about us page
* 49ab535 Added home page
```

3.3 Lesson 3

Lesson 3: Getting Started with GitHub

Table of Contents

3.3.1 Creating a repository on GitHub

Luego de haber hecho nuestros `commits` localmente necesitaremos tener un backup de nuestros archivos.

Entrar a nuestra cuenta de GitHub:

En la esquina superior derecha, clic en el ícono +. Elegir la opción *New repository*:

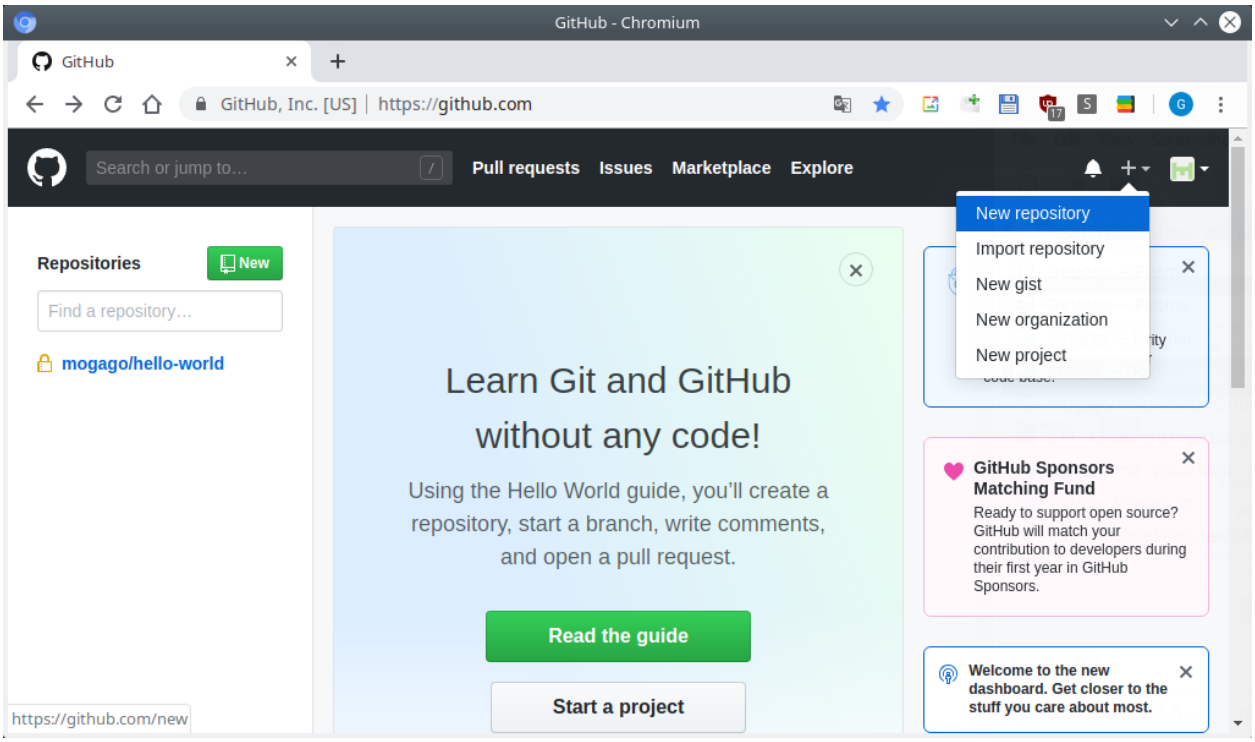


Fig. 3: GitHub - create new repository - Paso 1

Normalmente pondremos el nombre al repositorio igual al que hemos estado trabajando localmente:

Mantendremos este repositorio de forma pública, permitiendo que todos puedan acceder al repositorio.

Si fuéramos administradores no-técnicos y solo quisieramos crear un repositorio en GitHub que los desarrolladores puedan usar, podemos inicializar el repositorio con archivo `README`, de forma que no hagamos ninguna tarea en la línea de comandos. Este no es el caso.

Finalmente, clic en el botón *Create Repository*:

3.3.2 Uploading your repo to Github

Ocasionalmente tendremos que usar SSH. Primero deberemos crear llaves SSH.

En esta guía usaremos HTTPS. Seleccionar este botón:

En la parte inferior nos muestran comandos para subir (`push`) un repositorio existente a GitHub. Copiar y pegar ambas líneas en el terminal:

```
$ git remote add origin https://github.com/mogago/app1.git

$ git push -u origin master
```

(continues on next page)

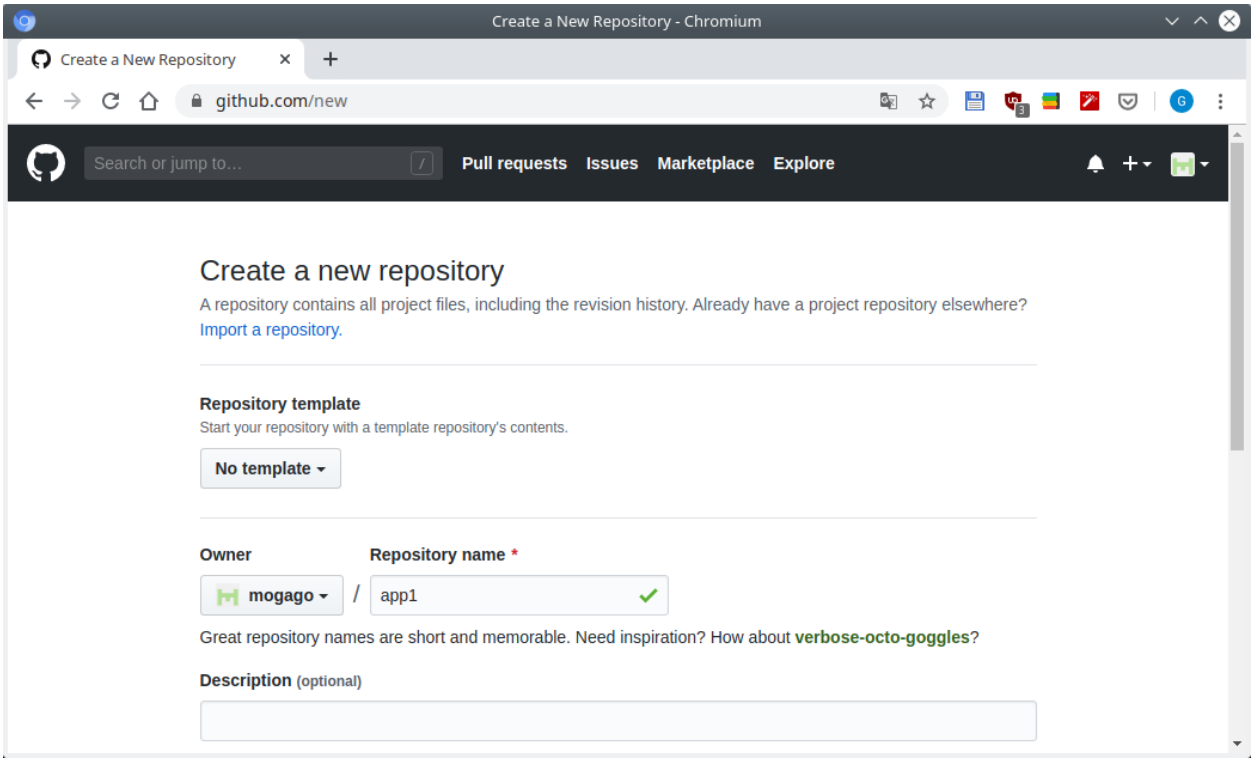


Fig. 4: GitHub - create new repository - Paso 2

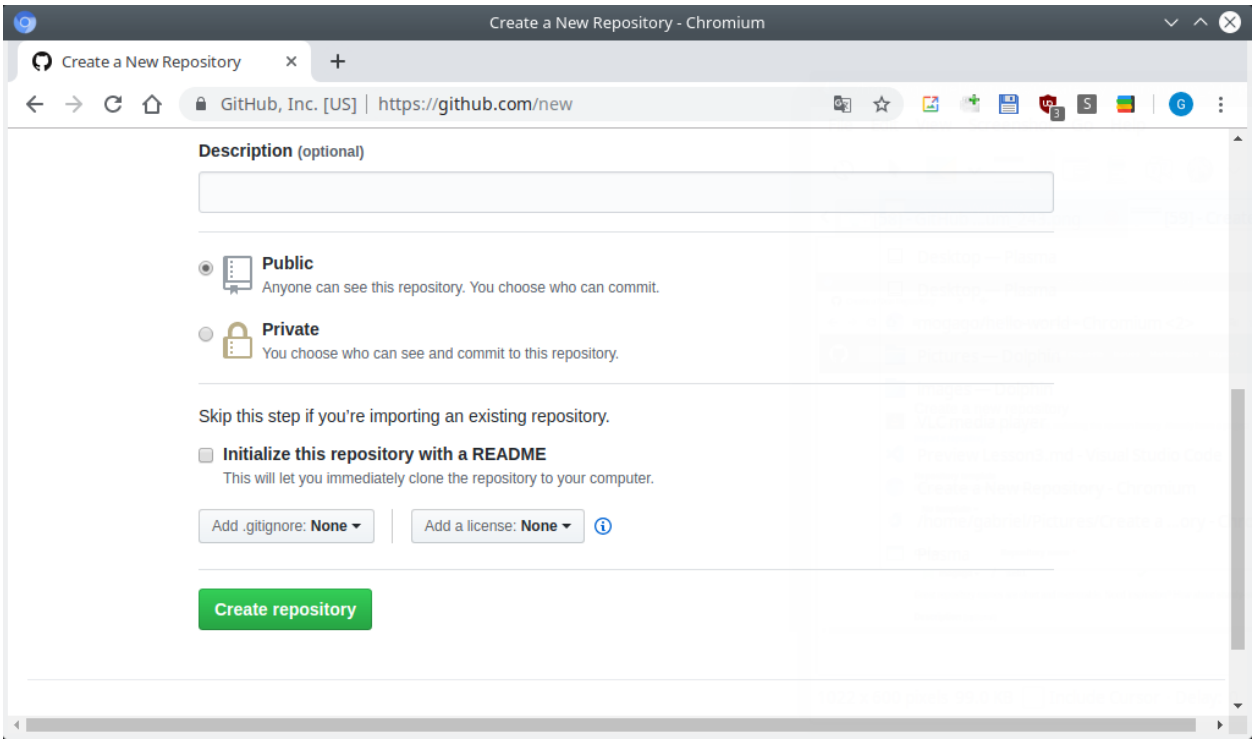


Fig. 5: GitHub - create new repository - Paso 3

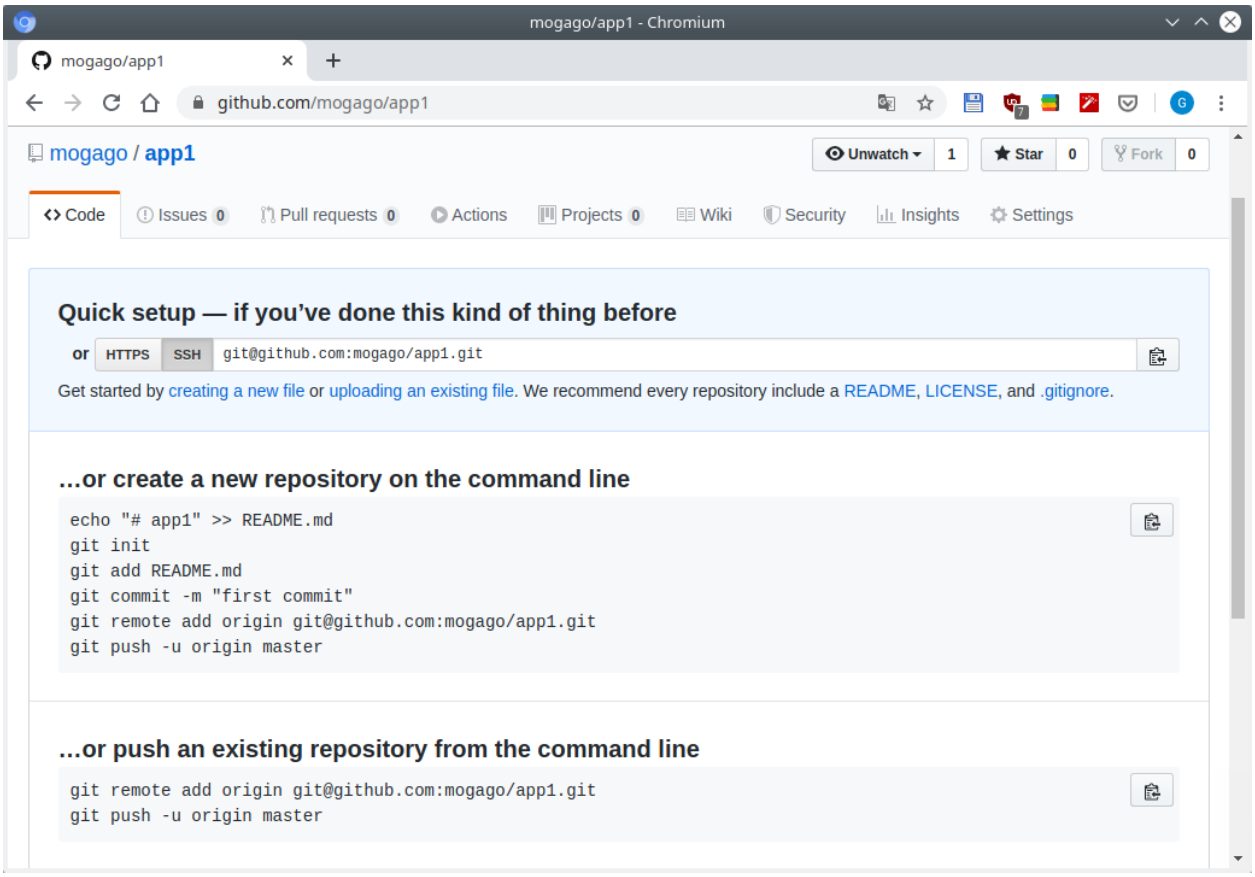


Fig. 6: GitHub - mode SSH

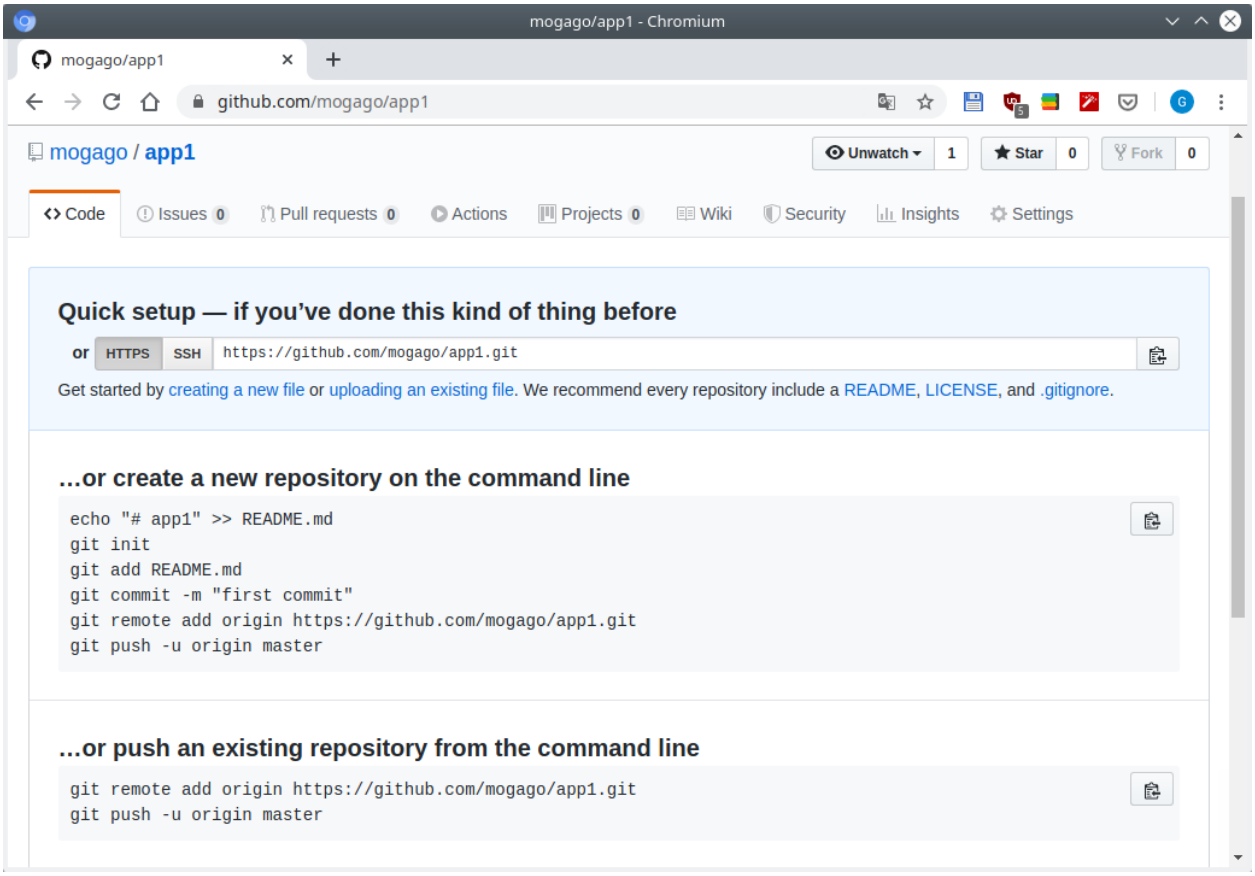


Fig. 7: GitHub - mode HTTPS

(continued from previous page)

```
Username for 'https://github.com': mogago
Password for 'https://mogago@github.com':
Counting objects: 7, done.
Delta compression using up to 4 threads.
Compressing objects: 100% (6/6), done.
Writing objects: 100% (7/7), 651 bytes | 651.00 KiB/s, done.
Total 7 (delta 1), reused 0 (delta 0)
remote: Resolving deltas: 100% (1/1), done.
To https://github.com/mogago/app1.git
* [new branch]      master -> master
Branch 'master' set up to track remote branch 'master' from 'origin'.
```

Si actualizamos nuestra página en GitHub veremos que los archivos se de nuestro repositorio local han sido subidos:

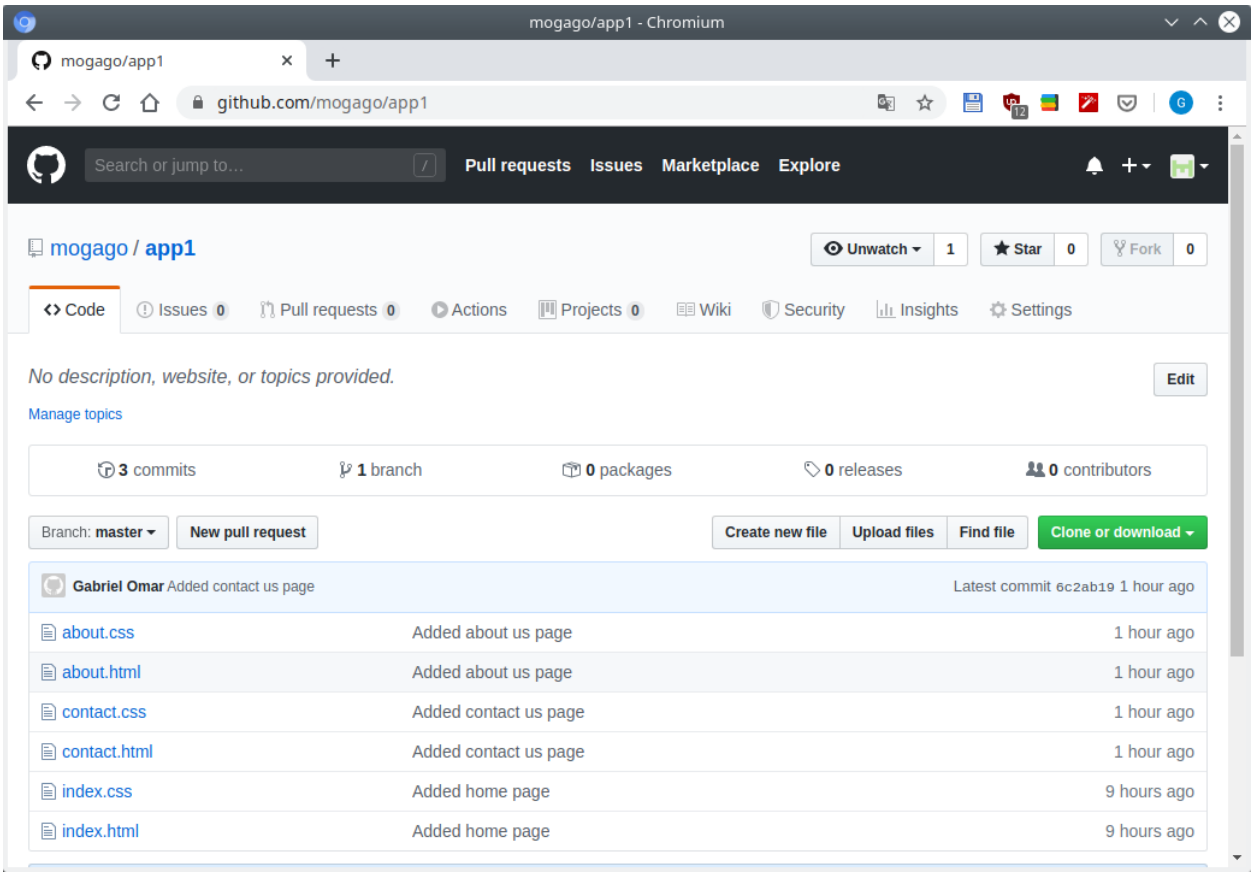


Fig. 8: GitHub - repo pushed

3.3.3 Creating a repository after starting to code

Creemos un nuevo repositorio. Primero iremos un directorio hacia arriba, comprobar que no estamos en un repositorio de Git y crear un directorio:

```
$ cd ..
```

(continues on next page)

(continued from previous page)

```
$ git status
fatal: not a git repository (or any of the parent directories): .git

$ mkdir web2
```

Esta vez haremos algo diferente. Idealmente haríamos `git init web2` y estaría listo. Pero generalmente la persona comenzaría a crear código:

```
$ cd web2/
$ touch index.html index.css
```

En este punto intentaríamos agregar los archivos a Git. Pero no funcionará porque no hemos creado un repositorio:

```
$ git add .
fatal: not a git repository (or any of the parent directories): .git
```

Lo que podemos hacer es ir al directorio que queremos inicializar y usar el comando *git init*:

```
$ git init
Initialized empty Git repository in /home/user/Documents/web2/.git/

$ git status
On branch master

No commits yet

Untracked files:
(use "git add <file>..." to include in what will be committed)

        index.css
        index.html

nothing added to commit but untracked files present (use "git add" to track)
```

Luego podremos agregar los archivos y hacer un `commit`. Por el momento los eliminaremos para tener un repositorio de Git sin archivos:

```
$ rm index.*
$ git s
```

3.4 Lesson 4

Lesson 4: Files in Git: renaming, deleting, ignoring

Table of Contents

- *Global gitexcludes and other Git ignore options*
- *Git ignore precedence*
- *`git commit -a` shortcut*

3.4.1 How to rename a file in Git

Listar los archivos del directorio `app1`:

```
$ ls
about.css  about.html  contact.css  contact.html  index.css  index.html
```

Para **renombrar** un archivo podemos hacerlo de **2 formas**:

1. Con el comando `git mv`:

```
$ git mv index.html home.htm
```

Veamos el estado del repo:

```
$ git s
R  index.html -> home.htm

$ git status
On branch master
Your branch is up to date with 'origin/master'.

Changes to be committed:
(use "git reset HEAD <file>..." to unstage)

        renamed:    index.html -> home.htm
```

Nos aparece una `R` en color verde (verde = "staged") en la versión resumida del comando. En la versión completa vemos que se ha renombrado el archivo. Pero todavía no se ha hecho el `commit`, solo un "stage".

Note: ¿Por no se hace el `commit` automáticamente?:

En un escenario real probablemente tendríamos otros archivos que quisiéramos formasen parte de este `commit`. Sobre todo, archivos que dependen del nombre que estamos cambiando. Por ejemplo, otras páginas que redirigen a `home.htm`.

Ahora podremos hacer el `commit`:

```
$ git commit -m "Renamed home page to follow corporate guidelines"

[master 57e910e] Renamed home page to follow corporate guidelines
1 file changed, 0 insertions(+), 0 deletions(-)
rename index.html => home.htm (100%)
```

2. La otra alternativa es no usar `git`:

En la práctica probablemente no usaremos Git para renombrar un archivo, sino Windows Explorer, IntelliJ, u otro IDE. En este caso lo haremos con el comando `mv` de Unix.

Renombrar `index.css` a `home.css` y ver el estado del repo:

```
$ mv index.css home.css

$ git status
On branch master
Your branch is ahead of 'origin/master' by 1 commit.
(use "git push" to publish your local commits)

Changes not staged for commit:
(use "git add/rm <file>..." to update what will be committed)
(use "git checkout -- <file>..." to discard changes in working directory)

        deleted:    index.css

Untracked files:
(use "git add <file>..." to include in what will be committed)

        home.css

no changes added to commit (use "git add" and/or "git commit -a")


$ git s

D index.css
?? home.css
```

Git asume que se ha eliminado el archivo `index.css` y se ha creado un nuevo archivo `home.css` que todavía no ha sido añadido al repositorio. Para arreglar esto usaremos:

```
$ git add -A
```

En versiones de Git `2.0` o superiores también se puede usar `git add .`

Si vemos el estado del repositorio, Git habrá registrado el cambio de nombre:

```
$ git status
On branch master
Your branch is ahead of 'origin/master' by 1 commit.
(use "git push" to publish your local commits)

Changes to be committed:
(use "git reset HEAD <file>..." to unstage)

        renamed:    index.css -> home.css

$ git s
R  index.css -> home.css
```

La razón por la cual Git se ha dado cuenta de que es un cambio de nombre se debe a que Git corrió el contenido de ambos archivos en un código que los comparó y retornó un "**similarity index**". Retorna un decimal entre `0` y `1`:

- `0` si los 2 archivos no tienen nada en común
- `1` si son idénticos caracter por caracter.
- Si el `similarity index` excede `0.5` Git supondrá que hemos renombrado el archivo.
- Si el `similarity index` es menor a `0.5` Git supondrá que hemos eliminado un archivo y agregado otro distinto.

Warning: Tener en cuenta estas consejos:

- NO hacer cambios sustanciales al contenido del archivo en el mismo `commit` con el que renombramos el archivo.
- NO debemos eliminar un archivo y agregar otro archivo que son distintos pero tienen contenido en común. Git supondrá que estamos renombrando el archivo.

Ahora podremos hacer el `commit`:

```
$ git commit -m "Renamed stylesheet for the home page"

[master 40afaa5] Renamed stylesheet for the home page
1 file changed, 0 insertions(+), 0 deletions(-)
rename index.css => home.css (100%)

$ git s
```

3.4.2 Deleting a file

Igual que para renombrar un archivo, hay 2 formas de eliminar un archivo:

1. Usar `git` para eliminar un archivo
2. Eliminar el archivo y luego decírselo a Git

A continuación el detalle de las 2 formas:

1. Usando Git:

Usar `git rm`:

```
$ git rm contact.html
rm 'contact.html'
```

El comando habrá realizado 2 acciones: el archivo se habrá eliminado

```
$  ls
about.css  about.html  contact.css  home.css  home.htm

$ git s
D  contact.html
```

El estado del repositorio nos indica con una letra `D` en color verde ("staged") que se ha eliminado ("Deleted").

Ahora podemos hacer el `commit`:

```
$ git commit -m "Deleted contact us page"
[master 4702c3a] Deleted contact us page
1 file changed, 0 insertions(+), 0 deletions(-)
delete mode 100644 contact.html
```

2. Eliminando un archivo con cualquier otro programa:

```
$ rm contact.css

$ git s
D contact.css
```

Nos aparece una letra `D` en color rojo, indicando que la eliminación está en estado "unstaged". Haremos el mismo procedimiento que para renombrar un archivo:

Usar `git add -A` o `git add .` para cambiar el estado a "staged":

```
$ git add -A

$ git s
D  contact.css
```

Ahora nos aparece la letra `D` en verde y podemos hacer el `commit`:

```
$ git commit -m "Deleted stylesheet for contact us page"
[master f1ebec7] Deleted stylesheet for contact us page
1 file changed, 0 insertions(+), 0 deletions(-)
delete mode 100644 contact.css
```

3.4.3 Ignoring files using a .gitignore file

Podríamos querer ignorar un archivo por distintas razones. Por ejemplo, cada vez que corremos una aplicación puede generar archivos de log.

Creemos un archivo que procederemos a ignorar:

```
$ touch test.log

$ git s
?? test.log
```

Tenemos un archivo "untracked", "unstaged". Git sabe que está ahí pero no va a ser guardado en historial al hacer un `commit`.

Podríamos dejarlo ahí, ya que, gracias al "staging area" de Git no necesitamos tener una forma especial de decirle a Git que ignore los archivos. Podríamos dejarlo ahí y nunca agregarlo a "staging area".

Sin embargo, mientras el repositorio vaya creciendo en cantidad de archivos se creará una larga lista de archivos no agregados. Y cuando queramos agregar un archivo a "staging area" no podremos encontrarlo con facilidad.

Podemos decirle a Git que ignore estos archivos, mediante la creación de un archivo `ignore`.

```
$ vi .gitignore

*.log
```

Con este archivo le decimos a Git que ignore todos los archivos con formato `.log`.

Veamos el estado del repositorio:

```
$ git s
?? .gitignore
```

Nuestro archivo `.gitignore` funcionó, no nos dice que el archivo `test.log` está en estado "untracked". Es decir, está ignorando su existencia. Sin embargo, nos está diciendo que el archivo `.gitignore` está en estado `untracked` y presuntamente debería ser agregado y luego hacerle un `commit`.

La mayoría de configuraciones en Git son particulares a la computadora donde estamos trabajando. Pero `.gitignore` está diseñado para ser compartido. Esto es porque queremos informar a las personas que descargan nuestro código del repo que también ignoren estos archivos. No queremos que ellos hagan un `commit` de los archivos que nosotros ignoramos.

Tip: La regla de oro para archivos que están en `.gitignore` es que si estamos en duda de agregar o no un archivo, debemos agregarlo. Si es algo que pueda afectar a parte del equipo, agregarlo a `.gitignore`.

Como cualquier otro archivo, lo agregamos y hacemos `commit`:

```
$ git add .
$ git s
A  .gitignore

$ git commit -m "Added log files to .gitignore"
[master 3c8e5e3] Added log files to .gitignore
1 file changed, 1 insertion(+)
create mode 100644 .gitignore
```

3.4.4 Global gitexcludes and other Git ignore options

Hay otras formas de ignorar archivos usando Git. Una forma es usando archivos "Global excludes". Hay algunos riesgos de usar esto pero es importante usarlo en el contexto correcto.

```
$ git config --global core.excludesfile ~/.gitignore
```

La ventaja de usar un archivo `.gitignore` global es que podemos ignorar algunos archivos en cada proyecto que trabajemos. El riesgo es que vamos a ignorar archivos en proyectos que algunos miembros del equipo no lo harán. Esto terminará haciendo que algunos miembros hagan `commits` que no deberían.

Tip: La recomendación es usar un `.gitignore` global en archivos que sabemos que nadie en el equipo nunca va a crear.

Otros 2 puntos importantes sobre `.gitignore`:

- Podemos crear cuantos `.gitignore` queramos. Podemos poner un `.gitignore` diferente en cada directorio de nuestro proyecto. Sin embargo NO debemos hacerlo. Es una mala idea porque el usuario quiere leer todo lo que va a ser ignorado de un único archivo.
- Si encontramos que alguien está ignorando archivos y no sabemos por qué. Buscamos el archivo `.gitignore` en la raíz del proyecto, o buscamos en cada subdirectorio, o en un archivo Global Exclude. Pero está ignorando un archivo que no encontramos. Esto es debido al siguiente archivo:

```
$ cat .git/info/exclude
# git ls-files --others --exclude-from=.git/info/exclude
# Lines that start with '#' are comments.
# For a project mostly in C, the following would be a good set of
# exclude patterns (uncomment them if you want to use them):
# *.[oa]
# *~
```

Este archivo es otro `.gitignore` local. Este archivo no se comparte con las personas cuando hacemos `push` al repositorio o cuando lo clonamos. No se usa en otros proyectos. Por tanto, si queremos ignorar archivos localmente en un proyecto y no dejar saber a nadie que estamos ignorando estos archivos podemos usar el archivo `.git/info/exclude`

3.4.5 Git ignore precedence

Digamos que estamos en un subdirectorio del proyecto que tiene su propio archivo `.gitignore`, el directorio padre tiene otro `.gitignore` y también tenemos el `.gitignore` global.

Lo primero que Git hará será correr el archivo global `.gitignore` para todos nuestros proyectos. Luego correrá el `.gitignore` del directorio raíz del proyecto. Finalmente correra las reglas `.gitignore` del subdirectorio.

La forma en que Git trabaja con archivos `.gitignore` es que "la prioridad importa" ("precedence matters"). Veamos el ejemplo:

```
$ touch special.log
$ git s
```

El archivo `special.log` es ignorado por Git. Pero qué pasa si queremos hacer un `commit` a este archivo. Tenemos que cambiar el documento `.gitignore`:

```
$ vi .gitignore

*.log
!special.log
```

Con el signo de exclamación (`!`) decimos que no ignore al archivo `special.log`. Podemos pensar en el archivo `.gitignore` como un conjunto de reglas que ponemos en donde el orden/prioridad importa. **La última regla manda sobre la primera**.

```
$ git s
M .gitignore
?? special.log
```

El estado nos dice que hemos modificado el archivo `.gitignore`. Pero ahora el archivo `special.log` es visible y puedo añadirlo y hacerle `commit`.

Si el orden de las instrucciones fuera distinto veríamos que se ignoran todos los archivos `.log`:

```
$ vi .gitignore

!special.log
*.log
```

```
$ git s
M .gitignore
```

Regresemos a la anterior configuración:

```
$ vi .gitignore

*.log
!special.log
```

```
$ git s
M .gitignore
?? special.log
```

Ahora estamos listos para agregar los cambios y guardarlos en historial.

3.4.6 `git commit -a` shortcut

Sabemos las ventajas de primero añadir un archivo y luego hacerle `commit`, pero podemos usar un atajo para hacer ambos pasos. El atajo es `git commit -a`, esto es añadir y también hacer `commit`.

> **Warning:** Pero este atajo viene con una advertencia. Solo funciona en archivos modificados (`M`), no en archivos "untracked".

Si tenemos un archivo que estaba en Git y lo hemos guardado, luego hicimos un cambio y queremos añadir y guardar ese cambio, podemos usar `git commit -a` pero no funcionará en archivos "untracked" (`??`).

En el último ejemplo teníamos:

```
$ git s
M .gitignore
?? special.log
```

Nos conviene hacer un `commit` sobre `.gitignore` primero y luego hacer un `commit` separado sobre `special.log`.

```
$ git commit -am "Modified gitignore to allow special log to be committed"
[master 1691860] Modified gitignore to allow special log to be committed
1 file changed, 1 insertion(+)

$ git s
?? special.log
```

Para el archivo `special.log` no hay atajos. Debemos agregarlo y luego hacer `commit`:

```
$ git add special.log

$ git s
A  special.log

$ git commit -m "Adding special log to show how log output should look"
[master c8ee367] Adding special log to show how log output should look
1 file changed, 0 insertions(+), 0 deletions(-)
create mode 100644 special.log
```

3.5 Lesson 5

Lesson 5: Branching, Merging, Rebasing

Table of Contents

3.5.1 Introducing Branching

El propósito de un "**branch**" es permitirnos crear una línea independiente de trabajo. Trabajar con "branches" en Git es de forma local. Si chequear otro branch toma más de medio segundo o más, algo está mal probablemente, ya que, es una operación local.

Veamos "branches" en Git:

```
$ git branch
* master
```

Esto nos muestra que solo tenemos el "branch" por defecto, llamado `master`. En Github podemos establecer otro "branches" como nuestro "branch" por defecto.

Generalmente el flujo de trabajo en Git es crear "**feature branches**" cada vez que creamos una nueva característica. Imaginemos que agregamos una característica de buscado a nuestra página web:

```
$ git branch search

$ git branch
* master
search
```

Hemos creado un nuevo "branch" llamdo `search`. Podemos cambiar de "branch" con el siguiente comando:

```
$ git checkout search
Switched to branch 'search'

$ git branch
master
* search
```

El color verde del nombre del "branch" y el asterisco a su costado nos dice que ahora estamos en el branch `search`.

```
$ touch search.html search.css

$ git add .

$ git commit -m "Added search page"
[search 54f68cb] Added search page
2 files changed, 0 insertions(+), 0 deletions(-)
create mode 100644 search.css
create mode 100644 search.html
```

Note: Hemos creado un `commit` que solo existe en el "branch" `search`.

Si corremos el alias de log creado anteriormente:

```
$ git lg
* 54f68cb (HEAD -> search) Added search page
* c8ee367 (master) Adding special log to show how log output should look
* 1691860 Modified gitignore to allow special log to be committed
* 3c8e5e3 Added log files to .gitignore
* f1ebec7 Deleted stylesheet for contact us page
* 4702c3a Deleted contact us page
* 40afaa5 Renamed stylesheet for the home page
* 57e910e Renamed home page to follow corporate guidelines
* 4388cdf (origin/master) Added contact us page
* aa2d60c Added about us page
* 49ab535 Added home page
```

Tip: Debemos leer el log desde abajo hacia arriba.

La primera línea del log nos dice que estamos en el "branch" `search` (`HEAD -> search`). `4388cdf (origin/master) Added contact us page` es la última vez que hicimos `push` hacia el servidor `origin` a Github.

Las cosas que tenemos en el directorio de trabajo actual (`HEAD`) es la punta del "branch" `search`. Y si listamos los archivos veremos:

```
$ ls
about.css   home.css  search.css   special.log
about.html  home.htm  search.html  test.log
```

Si cambiamos de "branch" a `master`:

```
$ git checkout master
Switched to branch 'master'
Your branch is ahead of 'origin/master' by 7 commits.
(use "git push" to publish your local commits)
```

Si vemos el directorio, habrá dos archivos menos:

```
$ ls
about.css  about.html  home.css  home.htm  special.log  test.log
```

En el log veremos lo siguiente:

```
$ git lg
* 54f68cb (search) Added search page
* c8ee367 (HEAD -> master) Adding special log to show how log output should look
* 1691860 Modified gitignore to allow special log to be committed
* 3c8e5e3 Added log files to .gitignore
* f1ebec7 Deleted stylesheet for contact us page
* 4702c3a Deleted contact us page
* 40afaa5 Renamed stylesheet for the home page
* 57e910e Renamed home page to follow corporate guidelines
* 4388cdf (origin/master) Added contact us page
* aa2d60c Added about us page
* 49ab535 Added home page
```

El log nos dice que estamos en el "branch" `master` como indica el texto `(HEAD, master)`. Además, todo lo que está encima de esta línea no ha ocurrido aún en la historia. Es como si hubiésemos retrocedido en el tiempo o ido u otra línea de historia. Para nosotros todavía no ha ocurrido el trabajo en el "branch" `serarch`.

Gracias a esto, distintos desarrolladores pueden trabajar al mismo tiempo, sobre diferentes características y luego solo unirlas en `master` cuando esté listo para poner el trabajo en producción.

3.5.2 Merging a branch

Imaginemos que hemos acabado con la página `search`. Ahora lo que debemos hacer es traer de vuelta el trabajo al "branch" `master`:

```
$ git checkout master
Already on 'master'
Your branch is ahead of 'origin/master' by 7 commits.
(use "git push" to publish your local commits)

$ git merge search
Updating c8ee367..54f68cb
Fast-forward
search.css  | 0
search.html | 0
2 files changed, 0 insertions(+), 0 deletions(-)
create mode 100644 search.css
create mode 100644 search.html
```

Este comando traerá el trabajo que hemos hecho en el "branch" `search` al "branch" `master`. Y ahora que vemos el directorio de trabajo del "branch" `master` tendrá los archivos creados por el "branch" `search`:

```
$ ls
about.css   home.css  search.css   special.log
about.html  home.htm  search.html  test.log
```

```
$ git lg
* 54f68cb (HEAD -> master, search) Added search page
* c8ee367 Adding special log to show how log output should look
* 1691860 Modified gitignore to allow special log to be committed
* 3c8e5e3 Added log files to .gitignore
* f1ebec7 Deleted stylesheet for contact us page
* 4702c3a Deleted contact us page
* 40afaa5 Renamed stylesheet for the home page
* 57e910e Renamed home page to follow corporate guidelines
* 4388cdf (origin/master) Added contact us page
* aa2d60c Added about us page
* 49ab535 Added home page
```

El log nos muestra que el "branch" `master` ha ido "fast forward". Este es el tipo de "merge" que se ha realizado. Tanto el "branch" `search` como `master` tienen el mismo ID de `commit`. Es decir, ambos tienen integrada la funcionalidad de `search`.

Una vez que hemos unido el "branch" `search` al "branch" `master` ya no necesitamos más al "branch" `search`. Por lo que podemos eliminarlo:

```
$ git branch -d search
Deleted branch search (was 54f68cb).
```

(continues on next page)

(continued from previous page)

```
$ git branch
* master

$ git lg
* 54f68cb (HEAD -> master) Added search page
* c8ee367 Adding special log to show how log output should look
* 1691860 Modified gitignore to allow special log to be committed
* 3c8e5e3 Added log files to .gitignore
* f1ebec7 Deleted stylesheet for contact us page
* 4702c3a Deleted contact us page
* 40afaa5 Renamed stylesheet for the home page
* 57e910e Renamed home page to follow corporate guidelines
* 4388cdf (origin/master) Added contact us page
* aa2d60c Added about us page
* 49ab535 Added home page
```

Eliminar el "branch" `search` no significa eliminar los archivos o `commits` que pasaron en ese "branch" solo eliminamos la etiqueta `search` del output del log.

3.5.3 Creating a fast forward merge

Hay 2 tipos de "**merge**" que Git hará usualmente, aunque existen más: **fast forward** y **recursive**. Veremos estos dos tipos de "merge" y cómo o cuándo usarlos.

- **Fast forward merge**:

La última vez usamos `git branch {new_branch}` para crear un nuevo "branch" y luego `git checkout {new_branch}` para cambiar al "branch" creado. Una opción para hacer los dos comandos al mismo tiempo es usar:

```
$ git checkout -b categories
Switched to a new branch 'categories'

$ git branch
* categories
master
```

Este comando ha creado un nuevo "branch" llamado `categories` y ha hecho "checkout" a ese "branch" automáticamente (es decir, hemos cambiado a ese "branch").

Tip: Buena práctica: crear un nuevo "branch" solo cuando se necesite. No crear varios "branch" que no serán usados próximamente.

Creemos un archivo de prueba y hagamos un `commit`:

```
$ touch categories.html

$ git add .

$ git commit -m "Added categories page"
[categories e92fcca] Added categories page
1 file changed, 0 insertions(+), 0 deletions(-)
create mode 100644 categories.html
```

Digamos que paralelamente otros estaban trabajando en otro "branch":

```
$ git checkout master
Switched to branch 'master'
Your branch is ahead of 'origin/master' by 8 commits.
(use "git push" to publish your local commits)

$ git checkout -b products
Switched to a new branch 'products'
```

Para crear un nuevo "branch" hemos partido del "branch" `master` pues lo estamos tomando como punto inicial.

En ese branch crearemos archivos de prueba:

```
$ touch products.html

$ git add .

$ git commit -m "Added products page"
[products e0b254d] Added products page
1 file changed, 0 insertions(+), 0 deletions(-)
create mode 100644 products.html
```

Veamos el log:

```
$ git lg
* e0b254d (HEAD -> products) Added products page
| * e92fcca (categories) Added categories page
|/
* 54f68cb (master) Added search page
* c8ee367 Adding special log to show how log output should look
* 1691860 Modified gitignore to allow special log to be committed
* 3c8e5e3 Added log files to .gitignore
* f1ebec7 Deleted stylesheet for contact us page
* 4702c3a Deleted contact us page
* 40afaa5 Renamed stylesheet for the home page
* 57e910e Renamed home page to follow corporate guidelines
* 4388cdf (origin/master) Added contact us page
* aa2d60c Added about us page
* 49ab535 Added home page
```

Interpretemos el log:

- Vemos gráficamente que existe una división del ID `54f68cb` en dos partes: `e92fcca` y `e0b254d`.
- A los 3 primeros `commits` se le han hecho `push` a Github. Las líneas de arriban indican `commits` que solo se han hecho localmente en el "branch" `master`.
- Y en las últimas 2 líneas vemos otros 2 `commits`: uno en la "branch" `categories` y el otro en el "branch" `products`.

Cuando decidamos que hemos acabado con el desarrollo de `products` debemos unirlo ("merge") con el "branch" `master`. Podemos hacerlo haciendo un "**fast forward merge**".

Hacemos `checkout` al "branch" `master` y hacemos "merge" del "branch" `products`. Por defecto se hará un **"merge" tipo Fast-forward**.

```
$ git checkout master
Switched to branch 'master'
Your branch is ahead of 'origin/master' by 8 commits.
(use "git push" to publish your local commits)
```

(continues on next page)

(continued from previous page)

```
$ git merge products
Updating 54f68cb..e0b254d
Fast-forward
products.html | 0
1 file changed, 0 insertions(+), 0 deletions(-)
create mode 100644 products.html
```

Como indica el output, se ha realizado un "merge" tipo fast forward. Y el log:

```
$ git lg
* e0b254d (HEAD -> master, products) Added products page
| * e92fcca (categories) Added categories page
|/
* 54f68cb Added search page
* c8ee367 Adding special log to show how log output should look
* 1691860 Modified gitignore to allow special log to be committed
* 3c8e5e3 Added log files to .gitignore
* f1ebec7 Deleted stylesheet for contact us page
* 4702c3a Deleted contact us page
* 40afaa5 Renamed stylesheet for the home page
* 57e910e Renamed home page to follow corporate guidelines
* 4388cdf (origin/master) Added contact us page
* aa2d60c Added about us page
* 49ab535 Added home page
```

En la primera línea vemos que el "branch" `master` ha saltado hacia adelante ("fast forward"), hacia el `commit` de ID `e0b254d`. Es decir, se ha añadido el `commit` hecho en el "branch" `products` como si fuera parte del "branch" `master`.

Eliminamos el "branch" `products`:

```
$ git branch -d products
Deleted branch products (was e0b254d).
```

3.5.4 Introducing recursives merges

- **Recursive merge**:

La razón por la cual pudimos hacer un "merge" `fast forward` fue porque no había diferencia en historia. Se puede pretender que los `commits` se hicieron directamente el "branch" `master`, ya que, no habían cambios en este "branch".

En el log actual:

```
$ git lg
* e0b254d (HEAD -> master) Added products page
| * e92fcca (categories) Added categories page
|/
* 54f68cb Added search page
* c8ee367 Adding special log to show how log output should look
* 1691860 Modified gitignore to allow special log to be committed
* 3c8e5e3 Added log files to .gitignore
* f1ebec7 Deleted stylesheet for contact us page
* 4702c3a Deleted contact us page
* 40afaa5 Renamed stylesheet for the home page
```

(continues on next page)

(continued from previous page)

```
* 57e910e Renamed home page to follow corporate guidelines
* 4388cdf (origin/master) Added contact us page
* aa2d60c Added about us page
* 49ab535 Added home page
```

Vemos que tenemos diferencias en historial de los "branches" `master` y `categories`. El "branch" `master` tiene un `commit` que `categories` no tiene (`e0b254d`) y `categories` tiene un `commit` que `master` no tiene (`e92fcca`).

Para resolver esto, Git tendrá que hacer un "**recursive merge**" , que juntará dos líneas de historia distintas y creará un nuevo `commit` que unirá esos 2 trabajos.

Primero comprobemos que estamos en el "branch" `master`:

```
$ git branch
categories
* master
```

Luego ejecutar el comando `git merge`. Este comando nos retorna a un editor de texto como `nano` donde podemos personalizar el mensaje de `commit`. Salir del editor y ver el resultado del "merge":

```
$ git merge categories

Merge made by the 'recursive' strategy.
categories.html | 0
1 file changed, 0 insertions(+), 0 deletions(-)
create mode 100644 categories.html
```

En el log veremos:

```
$ git lg
*   3662531 (HEAD -> master) Merge branch 'categories'
|\
| * 44409c8 (categories) Added categories page
* | 7c5c668 Added products page
|/
* 564baf7 Added search page
* d71267c Adding special log to show how log output should look
* 01f02ee Modified gitignore to allow special log to be committed
* 1b59604 Added log files to .gitignore
* ced7c37 Deleted stylesheet for contact us page
* ffaef7d Deleted contact us page
* 48b8639 Renamed stylesheet for the home page
* faf6cf4 Renamed home page to follow corporate guidelines
* 6c2ab19 (origin/master) Added contact us page
* 49bfaa2 Added about us page
* eaeaa56 Added home page
```

Como vemos en el `commit` más reciente notamos que se ha hecho un "merge" los cambios hechos en el "branch" `master` con los hechos en el "branch" `categories`.

Tip: La regla de oro es que si tenemos dos "branches" distintos que tienen al menos un `commit` en cada uno, tendremos que hacer un "**recursive merge**".

Note: Los "recursive merge" nos dan más información en el log que los "fast forward merge". Viendo el log podemos

---

ver los `commits` realizados en un "branch" distinto al branch `master` en el caso del "recursive merge".

3.5.5 'No fast forward' recursive merges

Esta idea de agregar la página de `products` en un lado y la página de `categories` en otra puede resultar confuso. Veremos una forma de realizar "fast forward merges" pero sin la complejidad de ver registros en diferentes tiempos.

Primero eliminar el branch `categories` y luego usar el "**no fast forward**" para crear un "recursive mersh".

```
$ git branch -d categories
Deleted branch categories (was 44409c8).

$ git lg
*   3662531 (HEAD -> master) Merge branch 'categories'
|\
| * 44409c8 Added categories page
* | 7c5c668 Added products page
|/
* 564baf7 Added search page
* d71267c Adding special log to show how log output should look
* 01f02ee Modified gitignore to allow special log to be committed
* 1b59604 Added log files to .gitignore
* ced7c37 Deleted stylesheet for contact us page
* ffaef7d Deleted contact us page
* 48b8639 Renamed stylesheet for the home page
* faf6cf4 Renamed home page to follow corporate guidelines
* 6c2ab19 (origin/master) Added contact us page
* 49bfaa2 Added about us page
* eaeaa56 Added home page
```

Ahora creemos una página `my_account` con un nuevo "branch" y realicemos algunos `commits`:

```
$ git checkout -b my_account
Switched to a new branch 'my_account'

$ touch account.html
$ git add .
$ git commit -m "Added my_account page"
[my_account 33d2f6e] Added my_account page
 1 file changed, 0 insertions(+), 0 deletions(-)
 create mode 100644 account.html

$ touch account.css
$ git add .
$ git commit -m "Styled my_account page"
[my_account 0c3519b] Styled my_account page
 1 file changed, 0 insertions(+), 0 deletions(-)
 create mode 100644 account.css
```

Tenemos un historial parecido al anterior:

```
$ git lg
* 0c3519b (HEAD -> my_account) Styled my_account page
* 33d2f6e Added my_account page
*   3662531 (master) Merge branch 'categories'
|\
```

(continues on next page)

(continued from previous page)

```
| * 44409c8 Added categories page
* | 7c5c668 Added products page
|/
* 564baf7 Added search page
* d71267c Adding special log to show how log output should look
* 01f02ee Modified gitignore to allow special log to be committed
* 1b59604 Added log files to .gitignore
* ced7c37 Deleted stylesheet for contact us page
* ffaef7d Deleted contact us page
* 48b8639 Renamed stylesheet for the home page
* faf6cf4 Renamed home page to follow corporate guidelines
* 6c2ab19 (origin/master) Added contact us page
* 49bfaa2 Added about us page
* eaeaa56 Added home page
```

Podemos ver los últimos 2 `commits` que solo pertenecen al branch `my_account`.

Cuando hayamos acabado con la sección `my_account` podríamos hacer un "merge" con el branch `master` y por defecto hará un "fast forward merge", pues no existen `commits` en `master` que no formen parte del historial del branch `my_account`.

Sin embargo, queremos esa información adicional que nos indique que esos dos `commits` fueron realizados juntos como parte de un mismo "branch". Para lograr esto ejecutamos:

```
$ git merge --no-ff my_account

Merge branch 'my_account'
```

La opción `--no-ff` significa "no fast forward" nos permite decirle a Git que preserve la información adicional de los `commits` realizados en el "branch". Git puede pretender que estos `commits` se realizaron el `master` porque no han habido otros cambios, pero le decimos que no lo haga.

Este comando nos retorna a un editor de texto, pues si no es un "fast forward merge", será un "recursive merge", es decir, se necesita un `commit` para juntar el historial.

Veamos el log:

```
$ git lg
*   777cd1a (HEAD -> master) Merge branch 'my_account'
|\
| * 0c3519b (my_account) Styled my_account page
| * 33d2f6e Added my_account page
|/
*   3662531 Merge branch 'categories'
|\
| * 44409c8 Added categories page
* | 7c5c668 Added products page
|/
* 564baf7 Added search page
* d71267c Adding special log to show how log output should look
* 01f02ee Modified gitignore to allow special log to be committed
* 1b59604 Added log files to .gitignore
* ced7c37 Deleted stylesheet for contact us page
* ffaef7d Deleted contact us page
* 48b8639 Renamed stylesheet for the home page
* faf6cf4 Renamed home page to follow corporate guidelines
* 6c2ab19 (origin/master) Added contact us page
```

(continues on next page)

(continued from previous page)

```
* 49bfaa2 Added about us page
* eaeaa56 Added home page
```

Al ver el log notamos que los `commits` con ID `0c3519b` y `33d2f6e` se realizaron juntos bajo un mismo "branch", aún cuando eliminemos el "branch":

```
$ git branch -d my_account
Deleted branch my_account (was 0c3519b).

$ git lg
*   777cd1a (HEAD -> master) Merge branch 'my_account'
|\
| * 0c3519b Styled my_account page
| * 33d2f6e Added my_account page
|/
*   3662531 Merge branch 'categories'
|\
| * 44409c8 Added categories page
* | 7c5c668 Added products page
|/
* 564baf7 Added search page
* d71267c Adding special log to show how log output should look
* 01f02ee Modified gitignore to allow special log to be committed
* 1b59604 Added log files to .gitignore
* ced7c37 Deleted stylesheet for contact us page
* ffaef7d Deleted contact us page
* 48b8639 Renamed stylesheet for the home page
* faf6cf4 Renamed home page to follow corporate guidelines
* 6c2ab19 (origin/master) Added contact us page
* 49bfaa2 Added about us page
* eaeaa56 Added home page
```

3.5.6 Resolving merge conflicts

Creemos un nuevo branch `cart` y realicemos `commit` con él:

```
$ git checkout -b cartSwitched to a new branch 'cart'

$ touch cart.html
$ git add .
$ git commit -m "Added shopping cart"
[cart 230d74e] Added shopping cart
 1 file changed, 0 insertions(+), 0 deletions(-)
 create mode 100644 cart.html
```

Editemos el archivo `home.htm` simulando que creamos un link al shopping cart `cart.html`:

```
$ vi home.htm

link to cart
```

Viendo el estado veremos que ha sido modificado el archivo `home.htm`:

```
$ git s
 M home.htm
```

Podemos usar el shortcut para hacer un `commit` a archivos modificados:

```
$ git commit -am "Added link on home page to shopping cart"

[cart 7b4c481] Added link on home page to shopping cart
 1 file changed, 1 insertion(+)
```

Ahora volviendo al branch `master` decidimos hacer un rediseño de la página `home`:

Tip: No preocuparnos por nombres de "branch" largos, es mejor que este sea descriptivo.

```
$ git checkout master
Switched to branch 'master'
Your branch is ahead of 'origin/master' by 14 commits.
  (use "git push" to publish your local commits)


$ git checkout -b home_page_redisign
Switched to a new branch 'home_page_redisign'

$ git branch
  cart
* home_page_redisign
  master
```

Desde este branch también editaremos el archivo `home.htm`:

```
$ vi home.htm

    Company logo
    Links:
    - Products
    - Categories

    Copyright 2014
```

Luego el estado será de modificado (`M`), por lo que podremos hacer `commit` directamente:

```
$ git s
 M home.htm

$ git commit -am "Implemented home page redesign"
[home_page_redisign d2276e2] Implemented home page redesign
 1 file changed, 7 insertions(+)
```

Veamos el log:

```
$ git lg
* d2276e2 (HEAD -> home_page_redisign) Implemented home page redesign
| * 7b4c481 (cart) Added link on home page to shopping cart
| * 230d74e Added shopping cart
|/
*   777cd1a (master) Merge branch 'my_account'
|\
| * 0c3519b Styled my_account page
| * 33d2f6e Added my_account page
```

(continues on next page)

(continued from previous page)

```
|/
*   3662531 Merge branch 'categories'
|\
| * 44409c8 Added categories page
* | 7c5c668 Added products page
|/
* 564baf7 Added search page
* d71267c Adding special log to show how log output should look
* 01f02ee Modified gitignore to allow special log to be committed
* 1b59604 Added log files to .gitignore
* ced7c37 Deleted stylesheet for contact us page
* ffaef7d Deleted contact us page
* 48b8639 Renamed stylesheet for the home page
* faf6cf4 Renamed home page to follow corporate guidelines
* 6c2ab19 (origin/master) Added contact us page
* 49bfaa2 Added about us page
* eaeaa56 Added home page
```

Tenemos una situación en la que se han realizado dos branches distintos con sus respectivos `commits` por separado. Ahora digamos que el rediseño de la página web ha finalizado y queremos hacer un merge con el branch `master`:

```
$ git checkout master
Switched to branch 'master'
Your branch is ahead of 'origin/master' by 14 commits.
  (use "git push" to publish your local commits)


$ git merge --no-ff home_page_redisign

Merge branch 'home_page_redesign'

Merge made by the 'recursive' strategy.
 home.htm | 7 +++++++
 1 file changed, 7 insertions(+)
```

Se realizó un "fast forward merge" usando una estrategia "recursive merge" mediante `--no-ff`. Ahora podemos eliminar el `branch`:

```
$ git branch -d home_page_redisign
Deleted branch home_page_redisign (was d2276e2).

$ git branch
  cart
* master

$ git lg
*   2bf9637 (HEAD -> master) Merge branch 'home_page_redesign'
|\
| * d2276e2 Implemented home page redesign
|/
| * 7b4c481 (cart) Added link on home page to shopping cart
| * 230d74e Added shopping cart
|/
*   777cd1a Merge branch 'my_account'
|\
| * 0c3519b Styled my_account page
```

(continues on next page)

(continued from previous page)

```
| * 33d2f6e Added my_account page
|/
*   3662531 Merge branch 'categories'
|\
| * 44409c8 Added categories page
* | 7c5c668 Added products page
|/
* 564baf7 Added search page
* d71267c Adding special log to show how log output should look
* 01f02ee Modified gitignore to allow special log to be committed
* 1b59604 Added log files to .gitignore
* ced7c37 Deleted stylesheet for contact us page
* ffaef7d Deleted contact us page
* 48b8639 Renamed stylesheet for the home page
* faf6cf4 Renamed home page to follow corporate guidelines
* 6c2ab19 (origin/master) Added contact us page
* 49bfaa2 Added about us page
* eaeaa56 Added home page
```

Como vemos en el log, la funcionalidad del branch `cart` todavía no se le ha realizado un "merge", no existe una línea que vuelva a la izquierda, desde el `commit 7b4c481` o `230d74e`.

Supondremos que ocurrirá un problema porque en los dos branches hemos realizado cambios totalmente distintos a `home.htm`, por lo que hemos creado deliberadamente una situación de conflicto en el "merge". Veamos que ocurre:

```
$ git branch
  cart
* master

$ git merge cart
Auto-merging home.htm
CONFLICT (content): Merge conflict in home.htm
Automatic merge failed; fix conflicts and then commit the result.
```

Este merge no puede ser tipo "fast forward" porque hay cosas en `master` que no están en `cart` y hay cosas en `cart` que no están en `master`.

El merge nos da un error al `Auto-merging` de `home.htm`. Git hace un buen trabajo de "merging" automático; por ejemplo, si tenemos un cambio al principio del archivo y otro al final del archivo, Git no dará ningún error. Pero si dos personas hicieron cambios al mismo archivo en las mismas líneas, Git no sabrá que habrás hecho.

Tip: Para resolver el problema de merge usar `git status`.

Veamos el estado del repositorio:

```
$ git status

On branch master
Your branch is ahead of 'origin/master' by 16 commits.
(use "git push" to publish your local commits)

You have unmerged paths.
(fix conflicts and run "git commit")
(use "git merge --abort" to abort the merge)
```

(continues on next page)

(continued from previous page)

```
Changes to be committed:

        new file:   cart.html

Unmerged paths:
    (use "git add <file>..." to mark resolution)

        both modified:   home.htm
```

Antes de hacer `commit` al archivo indicado debemos arreglar lo indicado en `Unmerged paths`. Estos podrían ser múltiples archivos. Arreglamos este error de merge en Git y podremos generalizarlo para otros casos donde tengamos varios archivos.

En Git, para resolver un problema de merge solo necesitamos un editor de texto. Por ejemplo con `vi`:

```
<<<<<<< HEAD
Company logo

Links:
- Products
- Categories

Copyright 2014
=======
link to cart
>>>>>>> cart
```

Vemos que se ha insertado texto adicional al archivo `home.htm`. `<<<<<<< HEAD` (**current change**) indica que en el master branch (`HEAD`) tenemos todo el contenido delimitado hasta el `=======`, luego de este delimitador tenemos contenido realizado con otro branch indicado por `>>>>>>> cart` (**incoming change**). Es decir, tenemos 2 contenidos distintos en el mismo archivo.

Lo que debemos hacer es editar el archivo de forma que se vea como debería:

```
Company logo

Links:
- Products
- Categories
- Cart

Copyright 2014
```

Warning: Es importante notar que nosotros podríamos editar el archivo como deseemos, agregando o eliminado líneas. Pero NO debemos modificar cosas no relacionadas al branch al cual le estamos haciendo merge, pues en el `commit` estaríamos mintiendo.

Una vez hemos guardado los cambios del archivo, revisar el estado:

```
$ git status
On branch master
Your branch is ahead of 'origin/master' by 16 commits.
(use "git push" to publish your local commits)

You have unmerged paths.
```

(continues on next page)

(continued from previous page)

```
(fix conflicts and run "git commit")
(use "git merge --abort" to abort the merge)

Changes to be committed:

        new file:   cart.html

Unmerged paths:
    (use "git add <file>..." to mark resolution)

        both modified:   home.htm
```

Vemos que el estado no ha cambiado. Así que ahora debemos decirle a Git que hemos arreglado el conflicto.

```
$ git add .

$ git status
On branch master
Your branch is ahead of 'origin/master' by 16 commits.
(use "git push" to publish your local commits)

All conflicts fixed but you are still merging.
(use "git commit" to conclude merge)

Changes to be committed:

        new file:   cart.html
        modified:   home.htm
```

Para finalizar este merge debemos hacer un `commit`. No usaremos la opción `-m` para ver que nos mande a un editor de texto y ver el mensaje por defecto del `commit`:

```
Merge branch 'cart'

# Conflicts:
#       home.htm
#
# It looks like you may be committing a merge.
# If this is not correct, please remove the file
#       .git/MERGE_HEAD
# and try again.


# Please enter the commit message for your changes. Lines starting
# with '#' will be ignored, and an empty message aborts the commit.
#
# On branch master
# Your branch is ahead of 'origin/master' by 16 commits.
#   (use "git push" to publish your local commits)
#
# All conflicts fixed but you are still merging.
#
# Changes to be committed:
#       new file:   cart.html
#       modified:   home.htm
#
```

Vemos que la primera línea es el mensaje de `commit` tradicional al hacer merge pero también tenemos información de los archivos en los que hubieron conflictos (`home.htm`). Podemos agregar información adicional a este mensaje:

```
Merge branch 'cart'

Conflicts:
        home.htm

Conflict in home.html between link to cart and new page design. Make link match the␣
↪new page redesign and saved.

$ git commit
[master 82ae112] Merge branch 'cart'
```

Guardemos el mensaje de `commit` y vemos el log:

```
$ git lg
*   82ae112 (HEAD -> master) Merge branch 'cart'
|\
| * 7b4c481 (cart) Added link on home page to shopping cart
| * 230d74e Added shopping cart
* |   2bf9637 Merge branch 'home_page_redesign'
|\ \
| |/
|/|
| * d2276e2 Implemented home page redesign
|/
*   777cd1a Merge branch 'my_account'
|\
| * 0c3519b Styled my_account page
| * 33d2f6e Added my_account page
|/
*   3662531 Merge branch 'categories'
|\
| * 44409c8 Added categories page
* | 7c5c668 Added products page
|/
* 564baf7 Added search page
* d71267c Adding special log to show how log output should look
* 01f02ee Modified gitignore to allow special log to be committed
* 1b59604 Added log files to .gitignore
* ced7c37 Deleted stylesheet for contact us page
* ffaef7d Deleted contact us page
* 48b8639 Renamed stylesheet for the home page
* faf6cf4 Renamed home page to follow corporate guidelines
* 6c2ab19 (origin/master) Added contact us page
* 49bfaa2 Added about us page
* eaeaa56 Added home page
```

Comprobamos que tenemos un `commit` con el merge al branch de `cart`. Finalmente eliminamos este branch:

```
$ git branch -d cart
Deleted branch cart (was 7b4c481).

$ git lg
*   82ae112 (HEAD -> master) Merge branch 'cart'
|\
| * 7b4c481 Added link on home page to shopping cart
```

(continues on next page)

(continued from previous page)

```
| * 230d74e Added shopping cart
* |   2bf9637 Merge branch 'home_page_redesign'
|\ \
| |/
|/|
| * d2276e2 Implemented home page redesign
|/
*   777cd1a Merge branch 'my_account'
```

3.5.7 Another merge conflict example

Veamos otro ejemplo de conflicto con merges. Creemos una página de `checkout` con un nuevo branch:

```
$ git checkout -b checkout_page
Switched to a new branch 'checkout_page'

$ touch checkout.html
$ git add .
$ git commit -m "Added checkout page"
[checkout_page a1b7834] Added checkout page
 1 file changed, 0 insertions(+), 0 deletions(-)
 create mode 100644 checkout.html
```

Simulemos la creación de un link a esta página creada desde el home:

```
$ vi home.htm

    Company logo

    Links:
    - Products
    - Categories
    - Cart
    - Checkout

    Copyright 2014
```

Luego hagamos un `commit`:

```
$ git commit -am "Added link to checkout page"
[checkout_page 048ec79] Added link to checkout page
 1 file changed, 1 insertion(+)
```

Volver al branch `master`:

```
$ git checkout master Switched to branch 'master'
Your branch is ahead of 'origin/master' by 19 commits.
  (use "git push" to publish your local commits)
```

Veamos el contenido del archivo `home.htm`, comprobando que no se han registrado los cambios hechos con el branch `checkout`.

```
$ cat home.htm
    Company logo

```

(continues on next page)

(continued from previous page)

```
    Links:
    - Products
    - Categories
    - Cart

    Copyright 2014
```

Añadir un nuevo branch `add_home_page_links` y editemos el archivo:

```
$ git checkout -b add_home_page_links
Switched to a new branch 'add_home_page_links'

$ vi home.htm

    Company logo

    Links:
    - Products
    - Categories
    - Cart
    - About
    - My Account

    Copyright 2014

$ git commit -am "Added new links to the home page"
[add_home_page_links d044e8e] Added new links to the home page
 1 file changed, 2 insertions(+)
```

Hemos creado una situación que presentará un conflicto cuando intentemos hacer un merge, pues hemos editado el mismo archivo con dos branches distintos.

Regresemos al branch `master` y hagamos merge con el branch `add_home_page_links`, posteriormente borremos este branch:

```
$ git checkout master
Switched to branch 'master'
Your branch is ahead of 'origin/master' by 19 commits.
  (use "git push" to publish your local commits)

$ git merge --no-ff add_home_page_links
Merge branch 'add_home_page_links'

Merge made by the 'recursive' strategy.
 home.htm | 2 ++
 1 file changed, 2 insertions(+)

$ git branch -d add_home_page_links
Deleted branch add_home_page_links (was d044e8e).
```

Veamos el log para comprobar que el trabajo hecho con el branch `add_home_page_links` se ha unido con el branch `master` mediante merge:

```
$ git lg
*   c3f3863 (HEAD -> master) Merge branch 'add_home_page_links'
|\
| * d044e8e Added new links to the home page
```

(continues on next page)

(continued from previous page)

```
|/
| * 048ec79 (checkout_page) Added link to checkout page
| * a1b7834 Added checkout page
|/
*   82ae112 Merge branch 'cart'
|\
| * 7b4c481 Added link on home page to shopping cart
| * 230d74e Added shopping cart
* |   2bf9637 Merge branch 'home_page_redesign'
|\ \
| |/
|/|
| * d2276e2 Implemented home page redesign
|/
*   777cd1a Merge branch 'my_account'
```

Lo siguiente será hacer un merge del branch `checkout` con el `master`. Es aquí donde obtendremos un conflicto:

```
$ git merge checkout_page
Auto-merging home.htm
CONFLICT (content): Merge conflict in home.htm
Automatic merge failed; fix conflicts and then commit the result.
```

Veamos `home.htm` para resolver el conflicto:

```
$ vi home.htm

    Company logo

    Links:
    - Products
    - Categories
    - Cart
    <<<<<<< HEAD
    - About
    - My Account
    =======
    - Checkout
    >>>>>>> checkout_page

    Copyright 2014
```

En ambos lados del historial tenemos las líneas del logo y copyright, pero en el `HEAD` (merge con `master` branch) tenemos links `About` y `My Account`. Mientras que en `>>>>>>> checkout_page` tenemos un link a `Checkout`. Git no sabe qué hacer ni el orden en que debería posicionar los elementos si todos deberían estar. Por lo tanto, es nuestra labor editar el contenido según deseemos con nuestro editor:

```
$ vi home.htm

    Company logo

    Links:
    - Products
    - Categories
    - Cart
    - About
```

(continues on next page)

(continued from previous page)

```
    - My Account
    - Checkout

    Copyright 2014
```

Tip: Algunos editores como Visual Studio Code proveen funcionalidad adicionales para realizar acciones sobre estos archivos con conflicto.

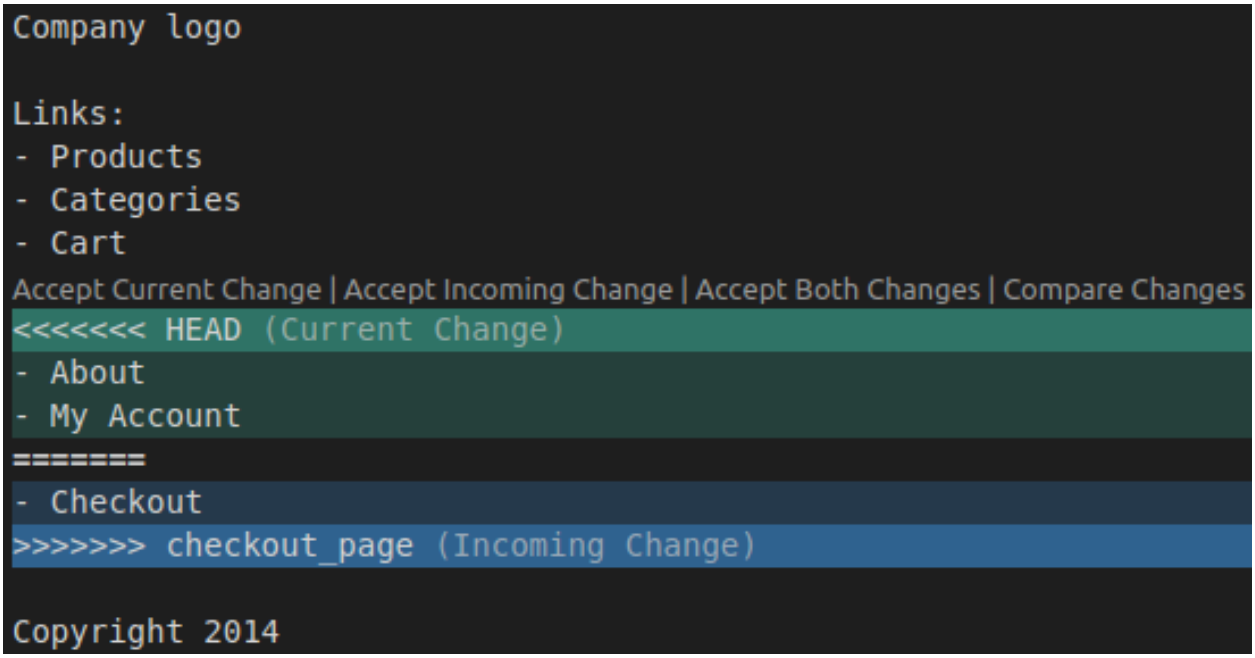


Fig. 9: Visual Studio Code - Git Conflicts

Guardar los cambios del archivo editado y ver el estado del repo con los conflictos todavía presentes:

```
$ git status
On branch master
Your branch is ahead of 'origin/master' by 21 commits.
(use "git push" to publish your local commits)

You have unmerged paths.
(fix conflicts and run "git commit")
(use "git merge --abort" to abort the merge)

Changes to be committed:

        new file:   checkout.html

Unmerged paths:
(use "git add <file>..." to mark resolution)

        both modified:   home.htm
```

Ahora podemos resolver el conflicto en Git añadiendo y haciendo un `commit`:

```
$ git add .
$ git status
On branch master
Your branch is ahead of 'origin/master' by 21 commits.
(use "git push" to publish your local commits)

All conflicts fixed but you are still merging.
(use "git commit" to conclude merge)

Changes to be committed:

        new file:   checkout.html
        modified:   home.htm

$ git commit

    Merge branch 'checkout_page'

    # Conflicts:
    #       home.htm

[master 064b2c7] Merge branch 'checkout_page'
```

Veamos el log para comprobar que hemos realizado un merge del trabajo realizado en `checkout_page` al `master` branch:

```
$ git lg
*   064b2c7 (HEAD -> master) Merge branch 'checkout_page'
|\
| * 048ec79 (checkout_page) Added link to checkout page
| * a1b7834 Added checkout page
* |   c3f3863 Merge branch 'add_home_page_links'
|\ \
| |/
|/|
| * d044e8e Added new links to the home page
|/
*   82ae112 Merge branch 'cart'
|\
| * 7b4c481 Added link on home page to shopping cart
| * 230d74e Added shopping cart
* |   2bf9637 Merge branch 'home_page_redesign'
|\ \
| |/
|/|
| * d2276e2 Implemented home page redesign
|/
*   777cd1a Merge branch 'my_account'
```

Eliminemos el branch `checkout_page`:

```
$ git branch -d checkout_page
Deleted branch checkout_page (was 048ec79).

$ git lg
*   064b2c7 (HEAD -> master) Merge branch 'checkout_page'
|\
```

(continues on next page)

(continued from previous page)

```
| * 048ec79 Added link to checkout page
| * a1b7834 Added checkout page
* |   c3f3863 Merge branch 'add_home_page_links'
|\ \
| |/
|/|
| * d044e8e Added new links to the home page
|/
*   82ae112 Merge branch 'cart'
|\
| * 7b4c481 Added link on home page to shopping cart
| * 230d74e Added shopping cart
* |   2bf9637 Merge branch 'home_page_redesign'
|\ \
| |/
|/|
| * d2276e2 Implemented home page redesign
|/
*   777cd1a Merge branch 'my_account'
```

3.5.8 Git Diff

`git diff` nos permite ver las diferencias. Con el comando `git help` veremos las posibilidades y opciones del comando:

```
$ git help diff
```

Hagamos un par de cambios con el `master` branch:

```
$ vi home.htm

    Company logo

    Links:
    - Products
    - Categories
    - Cart
    - About
    - History
    - My Account

    Copyright 2014
```

Si vemos el estado del repo, comprobaremos que existe un archivo modificado:

```
$ git s
 M home.htm
```

¿Pero cual es la modificación de ese archivo?: es para esto que sirve `git diff`

```
$ git diff

diff --git a/home.htm b/home.htm
index afd16cf..1fdd419 100644
--- a/home.htm
```

(continues on next page)

(continued from previous page)

```
+++ b/home.htm
@@ -5,7 +5,7 @@ Links:
- Categories
- Cart
- About
+- History
- My Account
-- Checkout

Copyright 2014
```

Veremos que hemos agregado una línea para '- History' (+) y hemos eliminado una línea para '- Checkout' (−).

Pasemos este cambio a 'staged area'. Y veamos el resultado del comando `git diff`:

```
$ git add .

$ git diff

$
```

`git diff` no nos muestra nada pues, solo muestra diferencias de cambios 'unstaged' por defecto.

Para ver los cambios 'staged' debemos agregar un parámetro:

```
$ git diff --staged

diff --git a/home.htm b/home.htm
index afd16cf..1fdd419 100644
--- a/home.htm
+++ b/home.htm
@@ -5,7 +5,7 @@ Links:
- Categories
- Cart
- About
+- History
- My Account
-- Checkout

Copyright 2014
```

Con `git diff HEAD` veremos las diferencias con el directorio actual y el último `commit` realizado.

```
$ git diff HEAD

diff --git a/home.htm b/home.htm
index afd16cf..1fdd419 100644
--- a/home.htm
+++ b/home.htm
@@ -5,7 +5,7 @@ Links:
- Categories
- Cart
- About
+- History
- My Account
-- Checkout

Copyright 2014
```

Ahora hagamos otros cambios:

```
Company logo

Links:
- About
- History
- My Account

Copyright 2014
```

Veamos el estado:

```
$ git s
MM home.htm
```

Esto nos dice que podemos tener cambios 'staged' y 'unstaged' del mismo archivo. Si corrieramos `git commit` cómo encontramos qué cambiaría, veamos las tres posibilidades:

```
$ git diff --staged
diff --git a/home.htm b/home.htm
index afd16cf..1fdd419 100644
--- a/home.htm
+++ b/home.htm
@@ -5,7 +5,7 @@ Links:
- Categories
- Cart
- About
+- History
- My Account
-- Checkout

Copyright 2014


$ git diff
diff --git a/home.htm b/home.htm
index 1fdd419..516fd1f 100644
--- a/home.htm
+++ b/home.htm
@@ -1,9 +1,6 @@
Company logo

Links:
-- Products
-- Categories
-- Cart
- About
- History
- My Account

$ git diff HEAD
diff --git a/home.htm b/home.htm
index afd16cf..516fd1f 100644
--- a/home.htm
+++ b/home.htm
@@ -1,11 +1,8 @@
Company logo
```

(continues on next page)

(continued from previous page)

```
Links:
-- Products
-- Categories
-- Cart
- About
+- History
- My Account
-- Checkout

Copyright 2014
```

- `git diff --staged` nos mostrará todas las diferencias que han sido añadidas al 'staged area'.
- `git diff` nos mostrará los cambios que no han sido añadidos al 'staged area'.
- `git diff HEAD` nos mostrará todos los cambios hechos desde el último `commit` al historial. Esto es lo que ocurrirá si añadimos y hacemos `commit` de todos los cambios.

```
$ git add .
$ git commit -m "Changed the home page links"
[master 51975c0] Changed the home page links
 1 file changed, 1 insertion(+), 4 deletions(-)
```

Note: `git diff --staged`, `git diff` y `git diff HEAD` son 3 comandos comúnes usados con `git diff` para ver qué pasará antes de hacer `commit` a los cambios.

3.5.9 Introducing rebasing

Rebasing es una herramienta intermedia que no es imprescindible para un buen uso de Git. Es usada para mejorar la calidad de la experiencia cuando colaboramos con otras personas.

Note: En Git hay dos comandos que usan la palabra `rebase` pero son totalmente diferentes: `git rebase` y `git rebase -i`. Debemos tratarlos como cosas diferentes que resulven problemas distintos. `git rebase -i` o "git rebase interfactive" se enfoca en reescribir y limpiar el historial, cambiar el orden de `commits`, comprimir `commits`. En cambio `git rebase` también sirve para limpiar el historial pero de una manera muy diferente.

Cuando el historial se vuelve muy complejo de leer (viendo el log), debemos considerar usar `rebase`.

Note: ¿Debería usar `rebase` o `merge`?: Usarás merge, la única pregunta será ¿harás un `rebase` primero?

Hacer un `rebase`, equivale a decir cambiar la base o cambiar el punto inicial de ese branch antes de hacerle merge.

3.5.10 Rebasing a branch

Creemos una situación en la que veamos el valor de 'rebasing'. Para esto crearemos 2 branches (`feature1` y `feature2`):

```
$ git checkout -b feature1
Switched to a new branch 'feature1'

$ touch feature1.html
$ git add .
$ git commit -m "Added feature 1"
[feature1 1e1bb9a] Added feature 1
 1 file changed, 0 insertions(+), 0 deletions(-)
 create mode 100644 feature1.html

$ git checkout master
Switched to branch 'master'
Your branch is ahead of 'origin/master' by 25 commits.
(use "git push" to publish your local commits)

$ git checkout -b feature2
Switched to a new branch 'feature2'

$ touch feature2.html
$ git add .
$ git commit -m "Added feature 2"
[feature2 b1c8223] Added feature 2
 1 file changed, 0 insertions(+), 0 deletions(-)
 create mode 100644 feature2.html
```

Acabamos en una situación con dos branches y un `commit` en cada uno:

```
$ git lg
* b1c8223 (HEAD -> feature2) Added feature 2
| * 1e1bb9a (feature1) Added feature 1
|/
* 51975c0 (master) Changed the home page links
*   064b2c7 Merge branch 'checkout_page'
|\
| * 048ec79 Added link to checkout page
| * a1b7834 Added checkout page
* |   c3f3863 Merge branch 'add_home_page_links'
|\ \
| |/
|/|
| * d044e8e Added new links to the home page
|/
*   82ae112 Merge branch 'cart'
|\
| * 7b4c481 Added link on home page to shopping cart
| * 230d74e Added shopping cart
* |   2bf9637 Merge branch 'home_page_redesign'
|\ \
| |/
|/|
| * d2276e2 Implemented home page redesign
|/
*   777cd1a Merge branch 'my_account'
```

Una vez que hemos acabado de desarrollar esos features queremos hacerles merge. Hagamos merge del branch `feature1` que será tipo recursive:

```
$ git checkout master
Switched to branch 'master'
Your branch is ahead of 'origin/master' by 25 commits.
  (use "git push" to publish your local commits)

$ git merge --no-ff feature1
Merge branch 'feature1'

Merge made by the 'recursive' strategy.
 feature1.html | 0
 1 file changed, 0 insertions(+), 0 deletions(-)
 create mode 100644 feature1.html

$ git branch -d feature1
Deleted branch feature1 (was 1e1bb9a).
```

Viendo el log comprobamos que hemos hecho merge de `feature1` en el `master` branch:

```
$  git lg
*   cdbc4aa (HEAD -> master) Merge branch 'feature1'
|\
| * 1e1bb9a Added feature 1
|/
| * b1c8223 (feature2) Added feature 2
|/
* 51975c0 Changed the home page links
*   064b2c7 Merge branch 'checkout_page'
```

Podríamos volver a hacer lo que hicimos antes: asegurarnos que estamos en el branch `master` y hacer merge de `feature2`, esto funcionaría pero no se verá bien conforme el proyecto sea más grande.

Ahora haremos un rebase de `feature2`. Esto significa que cambiaremos la base o el punto inicial de `feature2`. Pretendremos que no comenzamos a trabajar en `feature2` hasta después que se hizo un merge de `feature1` satisfactoriamente. Esto facilitará el orden de lectura de desarrollo de trabajo.

Viendo nuevamente el log notamos que `feature2` tiene una base identificada con el ID `51975c0`:

```
$  git lg
*   cdbc4aa (HEAD -> master) Merge branch 'feature1'
|\
| * 1e1bb9a Added feature 1
|/
| * b1c8223 (feature2) Added feature 2
|/
* 51975c0 Changed the home page links
*   064b2c7 Merge branch 'checkout_page'
```

Deberemos hacer un rebase hacia el `HEAD` de `master`, con punto inicial o base el ID `cdbc4aa`. Para esto haremos checkout a `feature2`, luego rebase a `master` y deshará cualquier trabajo que hayamos hecho en el branch `feature2`, irá al `HEAD` de `master` y volverá a hacer el trabajo como si hubiera escrito el código recién:

```
$ git checkout feature2
Switched to branch 'feature2'

$ git rebase master
First, rewinding head to replay your work on top of it...
Applying: Added feature 2

```

(continues on next page)

(continued from previous page)

```
$ git lg
* 9baf6a2 (HEAD -> feature2) Added feature 2
*   cdbc4aa (master) Merge branch 'feature1'
|\
| * 1e1bb9a Added feature 1
|/
* 51975c0 Changed the home page links
*   064b2c7 Merge branch 'checkout_page'
```

En el log comprobamos que ahora se ve como si no se hubiera trabajo en el branch `feature2` hasta que se hizo el merge de `feature1`. Lo que sigue es hacerle merge hacia el `master`:

```
$ git checkout master
Switched to branch 'master'
Your branch is ahead of 'origin/master' by 27 commits.
  (use "git push" to publish your local commits)

$ git merge --no-ff feature2
Merge made by the 'recursive' strategy.
 feature2.html | 0
 1 file changed, 0 insertions(+), 0 deletions(-)
 create mode 100644 feature2.html

Merge branch 'feature2'

$ git lg
*   08875c4 (HEAD -> master) Merge branch 'feature2'
|\
| * 9baf6a2 (feature2) Added feature 2
|/
*   cdbc4aa Merge branch 'feature1'
|\
| * 1e1bb9a Added feature 1
|/
* 51975c0 Changed the home page links
*   064b2c7 Merge branch 'checkout_page'

$ git branch -d feature2
Deleted branch feature2 (was 9baf6a2).
```

Ahora se ve como si hubiésemos trabajado en `feature1`, le hicimos merge y cuando acabamos trabajamos en `feature2`.

El otro beneficio es que nos permite realizar todas nuestras pruebas de integración en nuestro branches de `feature` antes de hacer hacerle merge en el `master`.

3.5.11 Handling rebase conflicts

Hemos visto conflicto con merge, ¿cómo será un conflicto con rebase?. Creemos un caso:

```
$ git checkout -b feature3
Switched to a new branch 'feature3'

$ touch feature3.html
$ git add .
```

(continues on next page)

(continued from previous page)

```
$ git commit -m "Added feature 3"
[feature3 4d5f561] Added feature 3
 1 file changed, 0 insertions(+), 0 deletions(-)
 create mode 100644 feature3.html

$ vi home.htm

    Company logo

    Links:
    - About
    - History
    - My Account
    - Feature 3

    Copyright 2014
```

Veamos el estado y hagamos un `commit`:

```
$ git s
 M home.htm

$ git commit -am "Added link to feature 3"
[feature3 07866ad] Added link to feature 3
 1 file changed, 1 insertion(+)
```

Para crear el conflicto haremos lo mismo con `feature4`:

```
$ git checkout master Switched to branch 'master'
Your branch is ahead of 'origin/master' by 29 commits.
  (use "git push" to publish your local commits)

$ git checkout -b feature4
Switched to a new branch 'feature4'

$ touch feature4.html

$ git add .

$ git commit -m "Added feature 4"
[feature4 411e7b2] Added feature 4
 1 file changed, 0 insertions(+), 0 deletions(-)
 create mode 100644 feature4.html

$ vi home.htm

    Company logo

    Links:
    - About
    - History
    - My Account
    - Feature 4

    Copyright 2014

$ git commit -am "Added link to feature 4"
```

(continues on next page)

(continued from previous page)

```
[feature4 df7ff8c] Added link to feature 4
 1 file changed, 1 insertion(+)
```

Hemos creado 2 features que harán conflicto, veamos el log:

```
$ git lg
* df7ff8c (HEAD -> feature4) Added link to feature 4
* 411e7b2 Added feature 4
| * 07866ad (feature3) Added link to feature 3
| * 4d5f561 Added feature 3
|/
*   08875c4 (master) Merge branch 'feature2'
|\
| * 9baf6a2 Added feature 2
|/
*   cdbc4aa Merge branch 'feature1'
|\
| * 1e1bb9a Added feature 1
|/
* 51975c0 Changed the home page links
*   064b2c7 Merge branch 'checkout_page'
```

Una vez que hayamos acabado con `feature3` le haremos merge y luego rebase de `feature4` y resolveremos el conflicto.

```
$ git checkout master
Switched to branch 'master'
Your branch is ahead of 'origin/master' by 29 commits.
  (use "git push" to publish your local commits)

$ git merge --no-ff feature3

Merge branch 'feature3'

Merge made by the 'recursive' strategy.
 feature3.html | 0
 home.htm      | 1 +
 2 files changed, 1 insertion(+)
 create mode 100644 feature3.html

$ git branch -d feature3
Deleted branch feature3 (was 07866ad).

$ git lg
*   e59b16c (HEAD -> master) Merge branch 'feature3'
|\
| * 07866ad Added link to feature 3
| * 4d5f561 Added feature 3
|/
| * df7ff8c (feature4) Added link to feature 4
| * 411e7b2 Added feature 4
|/
*   08875c4 Merge branch 'feature2'
|\
| * 9baf6a2 Added feature 2
|/
```

(continues on next page)

(continued from previous page)

```
*   cdbc4aa Merge branch 'feature1'
|\
| * 1e1bb9a Added feature 1
|/
* 51975c0 Changed the home page links
*   064b2c7 Merge branch 'checkout_page'
```

Viendo el log vemos que habrá un conflicto en `df7ff8c`. La única pregunta será si es un conflicto de merge o de rebase. Lo bueno es que realmente no importa de qué tipo es, pues los resolvemos de la misma forma.

Hagamos el rebase de `feature4` para tener un orden en la construcción de los 4 features en el historial.

```
$ git checkout feature4
Switched to branch 'feature4'

$ git rebase master
First, rewinding head to replay your work on top of it...
Applying: Added feature 4
Applying: Added link to feature 4
Using index info to reconstruct a base tree...
M       home.htm
Falling back to patching base and 3-way merge...
Auto-merging home.htm
CONFLICT (content): Merge conflict in home.htm
error: Failed to merge in the changes.
Patch failed at 0002 Added link to feature 4
Use 'git am --show-current-patch' to see the failed patch

Resolve all conflicts manually, mark them as resolved with
"git add/rm <conflicted_files>", then run "git rebase --continue".
You can instead skip this commit: run "git rebase --skip".
To abort and get back to the state before "git rebase", run "git rebase --abort".
```

Esta vez hemos encontrado un conflicto, similar al de merge. El mensaje nos dice que una vez arreglemos el conflicto, corramos `git rebase --continue`. `git status` nos dirá qué hacer:

```
$ git status
rebase in progress; onto e59b16c
You are currently rebasing branch 'feature4' on 'e59b16c'.
(fix conflicts and then run "git rebase --continue")
(use "git rebase --skip" to skip this patch)
(use "git rebase --abort" to check out the original branch)

Unmerged paths:
(use "git reset HEAD <file>..." to unstage)
(use "git add <file>..." to mark resolution)

        both modified:   home.htm

no changes added to commit (use "git add" and/or "git commit -a")
```

El estado nos indica que estamos en medio de un rebase de `e59b16c`, y debemos arreglar el archivo `home.htm`. Veamos el log:

```
$ git lg
* 7eda1d6 (HEAD) Added feature 4
*   e59b16c (master) Merge branch 'feature3'
```

(continues on next page)

(continued from previous page)

```
|\
| * 07866ad Added link to feature 3
| * 4d5f561 Added feature 3
|/
| * df7ff8c (feature4) Added link to feature 4
| * 411e7b2 Added feature 4
|/
*   08875c4 Merge branch 'feature2'
|\
| * 9baf6a2 Added feature 2
|/
*   cdbc4aa Merge branch 'feature1'
|\
| * 1e1bb9a Added feature 1
|/
* 51975c0 Changed the home page links
*   064b2c7 Merge branch 'checkout_page'
```

Veremos el mismo commit 2 veces: `7eda1d6` y `df7ff8c`: Added feature 4. Una vez que resolvamos el conflicto, finalizará el rebase y eliminará los commits sobrantes.

Arreglémoslo igual que un conflicto de merge, excepto que veremos una diferencia en el archivo conflictivo. Ahora no veremos el nombre del branch sino con un commit específico que está dando conflicto:

```
$ cat home.htm

    Company logo

    Links:
    - About
    - History
    - My Account
    <<<<<<< HEAD
    - Feature 3
    =======
    - Feature 4
    >>>>>>> Added link to feature 4

    Copyright 2014

$ vi home.htm

    Company logo

    Links:
    - About
    - History
    - My Account
    - Feature 3
    - Feature 4

    Copyright 2014

$ git add .

$ git status
rebase in progress; onto e59b16c
```

(continues on next page)

(continued from previous page)

```
You are currently rebasing branch 'feature4' on 'e59b16c'.
(all conflicts fixed: run "git rebase --continue")

Changes to be committed:
(use "git reset HEAD <file>..." to unstage)

        modified:   home.htm
```

El estado nos dice que se han arreglado todos los conflictos y solo corramos el siguiente comando:

```
$ git rebase --continue
Applying: Added link to feature 4

$ git lg
* 5f75f25 (HEAD -> feature4) Added link to feature 4
* 7eda1d6 Added feature 4
*   e59b16c (master) Merge branch 'feature3'
|\
| * 07866ad Added link to feature 3
| * 4d5f561 Added feature 3
|/
*   08875c4 Merge branch 'feature2'
|\
| * 9baf6a2 Added feature 2
|/
*   cdbc4aa Merge branch 'feature1'
|\
| * 1e1bb9a Added feature 1
|/
* 51975c0 Changed the home page links
*   064b2c7 Merge branch 'checkout_page'
```

Hemos hecho un rebase de `feature4` sobre el `master` y solo nos quedaría hacer merge:

```
$ git checkout master Switched to branch 'master'
Your branch is ahead of 'origin/master' by 32 commits.
  (use "git push" to publish your local commits)

$ git merge --no-ff feature4
Merge branch 'feature4'

Merge made by the 'recursive' strategy.
 feature4.html | 0
 home.htm      | 1 +
 2 files changed, 1 insertion(+)
 create mode 100644 feature4.html

$ git branch -d feature4
Deleted branch feature4 (was 5f75f25).

$ git lg
*   d0ade3c (HEAD -> master) Merge branch 'feature4'
|\
| * 5f75f25 Added link to feature 4
| * 7eda1d6 Added feature 4
|/
*   e59b16c Merge branch 'feature3'
```

(continues on next page)

(continued from previous page)

```
|\
| * 07866ad Added link to feature 3
| * 4d5f561 Added feature 3
|/
*   08875c4 Merge branch 'feature2'
|\
| * 9baf6a2 Added feature 2
|/
*   cdbc4aa Merge branch 'feature1'
|\
| * 1e1bb9a Added feature 1
|/
* 51975c0 Changed the home page links
*   064b2c7 Merge branch 'checkout_page'
```

Vemos que el historial está limpio y es linear en la construcción de `feature1`, `feature2`, `feature3` y `feature4`.

3.6 Lesson 6

Lesson 6: Git Internals

Table of Contents

3.6.1 Introducing Git under the hood

En esta sección veremos Git y GitHub a bajo nivel (under the hood). Nos servirá para entender varias cosas que hacemos con Git y por qué trabajan de la forma en que lo hacen.

3.6.2 Creating the first commit

Crear el siguiente archivo en un nuevo repositorio:

```
$ git init app2
Initialized empty Git repository in /home/user/Documents/gittests/app2/.git/

$ cd app2
$ vi index.html

    hello world

$ git s
?? index.html
```

según el estado tenemos un archivo sin trackear. Hagamos un `commit` de ese archivo:

```
$ git add .
$ git commit -m "Added home page"
[master (root-commit) 6c4de50] Added home page
 1 file changed, 1 insertion(+)
 create mode 100644 index.html
```

3.6.3 Exploring the object store

En el directorio de trabajo actual abramos el directorio oculto `.git`:

```
$ ls -a
.  ..  .git  index.html

$ ls -a .git/
.   branches        config       HEAD   index  logs     refs
..  COMMIT_EDITMSG  description  hooks  info   objects
```

Veamos los elementos dentro del subdirectorio `.git/objects`

```
$ ls .git/objects/
3b  6c  ff  info  pack

$ tree .git/objects/
.git/objects/
├── 3b
│   └── 18e512dba79e4c8300dd08aeb37f8e728b8dad
├── 6c
│   └── 4de5016bf89390a661676c855d82005b88c448
├── ff
│   └── 46f55ec679bb30e1b2291bf56558ec435b1df3
├── info
└── pack

5 directories, 3 files
```

Se han creado 3 nuevos directorios en en este subdirectorio: `3b`, `6c`, `ff`. Aquí es donde Git guarda todos sus objetos. Si ponemos un archivo en control de versiones con Git estará en este directorio, si tenemos un filename en el directorio estará aquí, si creamos un `commit` será puesto también en el directorio de objetos.

Tenemos 3 archivos, pudiendo concatenar los nombres del directorio y el nombre del archivo para obtener el hash SHA-1. Por ejemplo, en este caso los 3 objetos serían:

- `3b 18e512dba79e4c8300dd08aeb37f8e728b8dad`

- `6c 4de5016bf89390a661676c855d82005b88c448`
- `ff 46f55ec679bb30e1b2291bf56558ec435b1df3`

3.6.4 cat-file to explore object contents

Veamos qué son estos 3 objetos. Usaremos un comando a bajo nivel `git cat-file -p` para ver qué contiene dentro un objeto en el git object store. Es suficiente dar 4 caracteres del objeto para obtener la información:

```
$ git cat-file -p 3b18
hello world

$ git cat-file -p 3b18e512dba79e4c8300dd08aeb37f8e728b8dad
hello world
```

Esto quiere decir que si en otra PC escribiéramos en el archivo `index.html` los caracteres `hello world` también tendríamos un objeto con el mismo identificador `3b18....` La razón de esto es porque Git usa un algoritmo hash SHA-1.

Probemos con el ultimo directorio:

```
$ git cat-file -p ff46
100644 blob 3b18e512dba79e4c8300dd08aeb37f8e728b8dad    index.html
```

Los contenidos del primer respositorio eran sobre el contenido del archivo `index.html` (content store) pero no decían nada sobre el nombre del archivo. Esto es útil porque si tuviese otro archivo con exactamente el mismo contenido pero con otro nombre y otro directorio no tendría que tenerlo duplicado.

En este último directorio obtenemos el nombre del archivo con detalles extra:

- `100644`: no es un archivo ejecutable
- `blob`: binary large object
- `3b18e512dba79e4c8300dd08aeb37f8e728b8dad`: contenidos del archivo descritos por este identificador
- `index.html`: nombre del archivo

Este objeto es llamado **tree object**, que guarda el directorio con toda la información del nombre del archivo y qué objetos guarda el contenido de esos archivos en el directorio.

El otro objeto que tenemos y varía de PC en PC sin importar que sigamos los mismos pasos es el siguiente:

```
$ git cat-file -p 6c4d
tree ff46f55ec679bb30e1b2291bf56558ec435b1df3
author Nombre Apellido <newuser1@mail.com> 1577681651 -0500
committer Nombre Apellido <newuser1@mail.com> 1577681651 -0500

Added home page
```

Este es el mensaje de `commit` que también vemos al momento de crear el `commit` o listar el log:

```
$ git lg
* 6c4de50 (HEAD -> master) Added home page
```

La razón de que sea diferente para cualquier PC es que tiene autores, correo, día, hora y mensaje del `commit`.

3.6.5 The benefits of Git's use of SHA1 hashes

La ventaja de usar SHA1 nos da un nivel de seguridad. Si el hash SHA1 es el mismo todo será lo mismo para el caso del tree hash y el hash de los contenidos del archivo. Es decir, tenemos una consistencia garantizada en términos de qué contenidos es guardada dentro de los repositorios.

3.6.6 Git as a content store (how many new hashes)

Agreguemos otro archivo:

```
$ vi about.html
hello world

$ git add .
$ git commit -m "Added about us page"
[master 7c71072] Added about us page
 1 file changed, 1 insertion(+)
 create mode 100644 about.html
```

¿Cuántos nuevos objetos deberá haber?: podríamos suponer que 3 (contenidos del nuevo archivo, nuevo tree del directorio con 2 archivos en él y el nuevo commit).

Si listamos el directorio veremos que solo hay 2 nuevos:

```
$ tree .git/objects/
.git/objects/
├── 3b
│   └── 18e512dba79e4c8300dd08aeb37f8e728b8dad
├── 40
│   └── 83ab857b34cf3ee53639f6ee188fa36e751c11
├── 6c
│   └── 4de5016bf89390a661676c855d82005b88c448
├── 7c
│   └── 71072f9f35c27abf0a39ba71414ca4e64540f6
├── ff
│   └── 46f55ec679bb30e1b2291bf56558ec435b1df3
├── info
└── pack
```

Veamos que hay en ellos:

```
$ git cat-file -p 4083
100644 blob 3b18e512dba79e4c8300dd08aeb37f8e728b8dad    about.html
100644 blob 3b18e512dba79e4c8300dd08aeb37f8e728b8dad    index.html
```

Este es el nuevo tree que debería tener el mismo identificador en cualquier PC que haya creado los mismo archivos.

```
$ git cat-file -p 7c71tree 4083ab857b34cf3ee53639f6ee188fa36e751c11
parent 6c4de5016bf89390a661676c855d82005b88c448
author Nombre Apellido <newuser1@mail.com> 1577716707 -0500
committer Nombre Apellido <newuser1@mail.com> 1577716707 -0500

Added about us page
```

Este objeto hace referencia al último `commit`. No existe un sexto objeto porque nosotros sabemos qué pasa cuando le hacemos un hash SHA1 al string `hello world`. Git ya calculó el hash SHA1 de ese texto anteriormente y al crear

un nuevo archivo con exactamente el mismo contenido apuntará al mismo objeto. Git no necesita duplicar objetos en el **content store**.

Hay algunas implicancias para esto: es eficiente tener el mismo contenido del archivo a través del tiempo, en diferentes directorios o diferentes branches. Serán guardados como una sola referencia, y será eficiente en términos de espacio. Sin embargo, hay una desventaja, ¿qué pasa si cambio un caracter en el contenido del archivo?: cambia el hash SHA1 sin importar qué tan grande o pequeño es el cambio. Git irá creando copias y más copias de esos archivos con pequeñas diferencias. ¿Por qué no seguir el procedimiento de Subversion donde solo se almacenan los deltas?. Resulta que esta decisión de Git sí es inteligente.

La clave de que funcione este modo de trabajo en Git es por los **pack files**. Pensemos en los archivos zip; si tenemos 100 archivos que son similares y son todos archivos de texto los zipeamos antes de enviarlo a alguna persona, donde el trabajo de compresión puede realizar un buen trabajo tomando archivos similares y generando una archivo más pequeño. Esto es lo que hace Git a bajo nivel, hace una copia entera de cada modificación de un archivo y no representará un problema porque creará un paquete de ellos y los guardará oficialmente en disco.

Note: Una regla que debemos seguir es no guardar archivos binarios grandes en Git que frecuentemente estén cambiando. Desafortunadamente los programas de compresión no harán un buen trabajo comprimiendo archivos binarios con pequeñas modificaciones.

3.6.7 Understanding remotes and their configuration

Hagamos una conexión a un servidor remoto. Tomemos estos archivos y hagámosle push a GitHub:

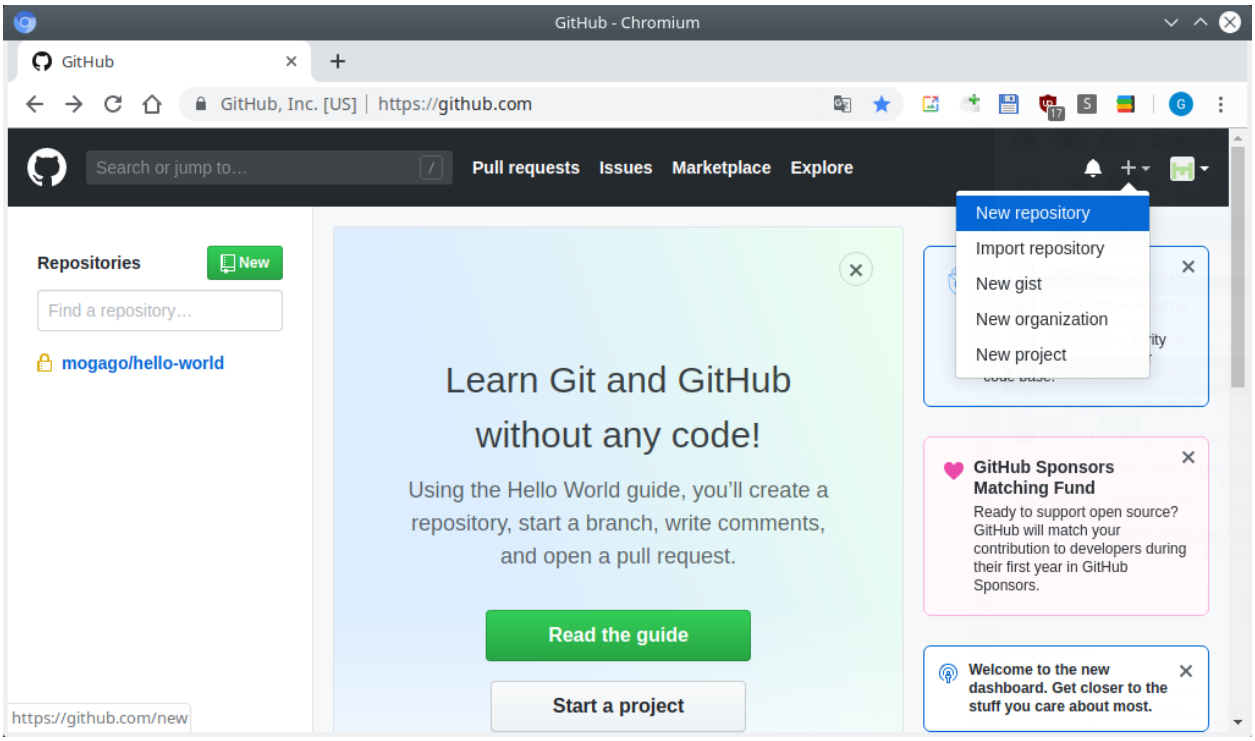


Fig. 10: GitHub - create new repository - Paso 1

Ya hemos creado nuestro repositorio. La última vez copiamos y pegamos los siguientes comandos sin pensar qué hacían, ahora veamos qué está ocurriendo:

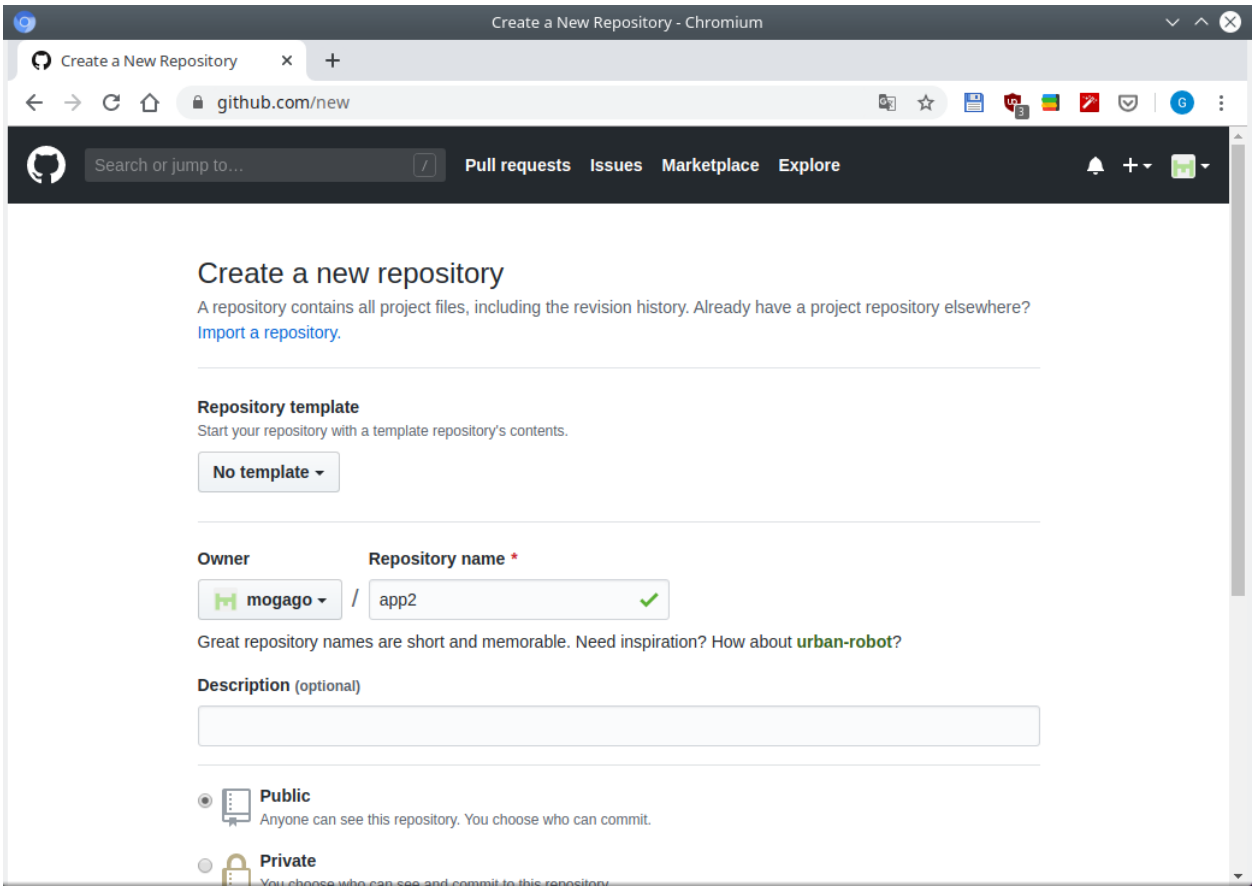


Fig. 11: GitHub - create new repository - Paso 2

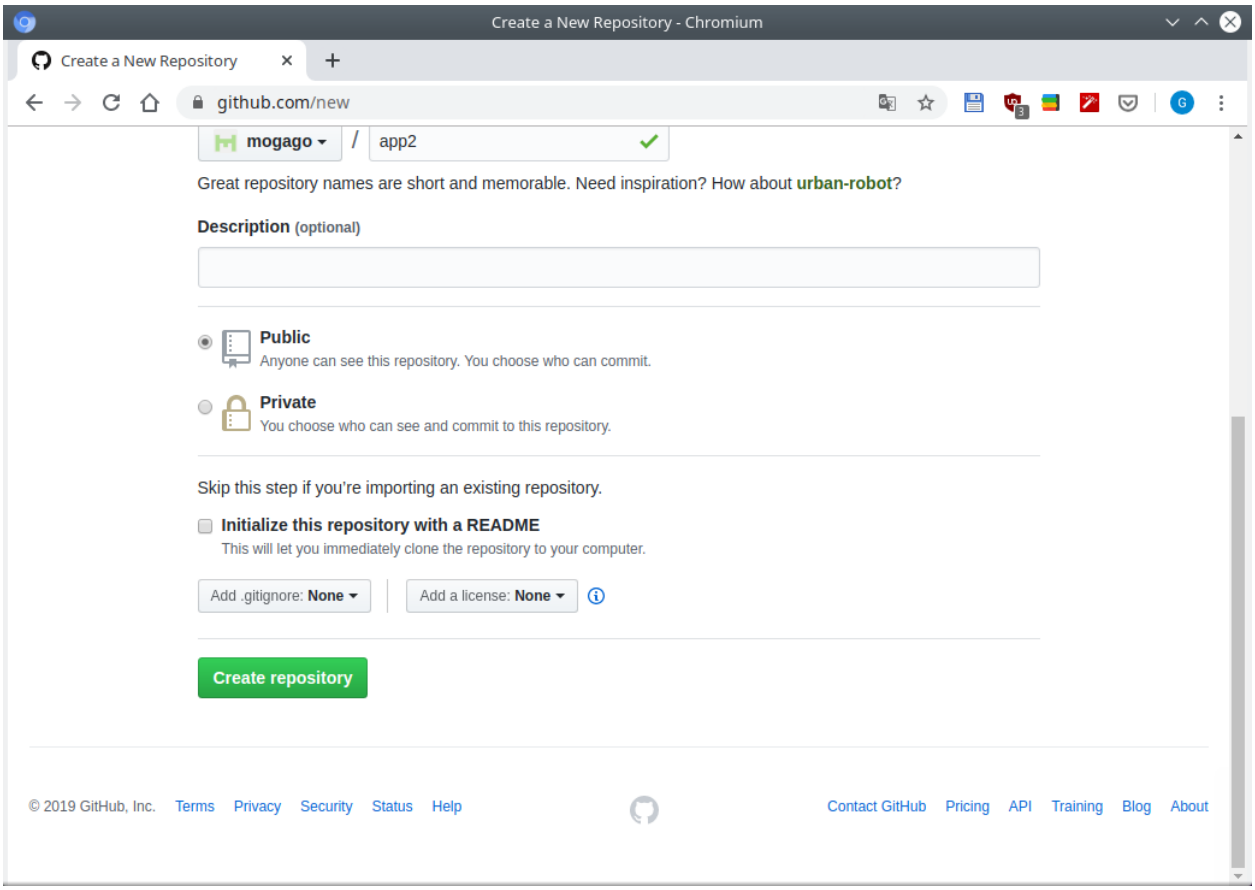


Fig. 12: GitHub - create new repository - Paso 3

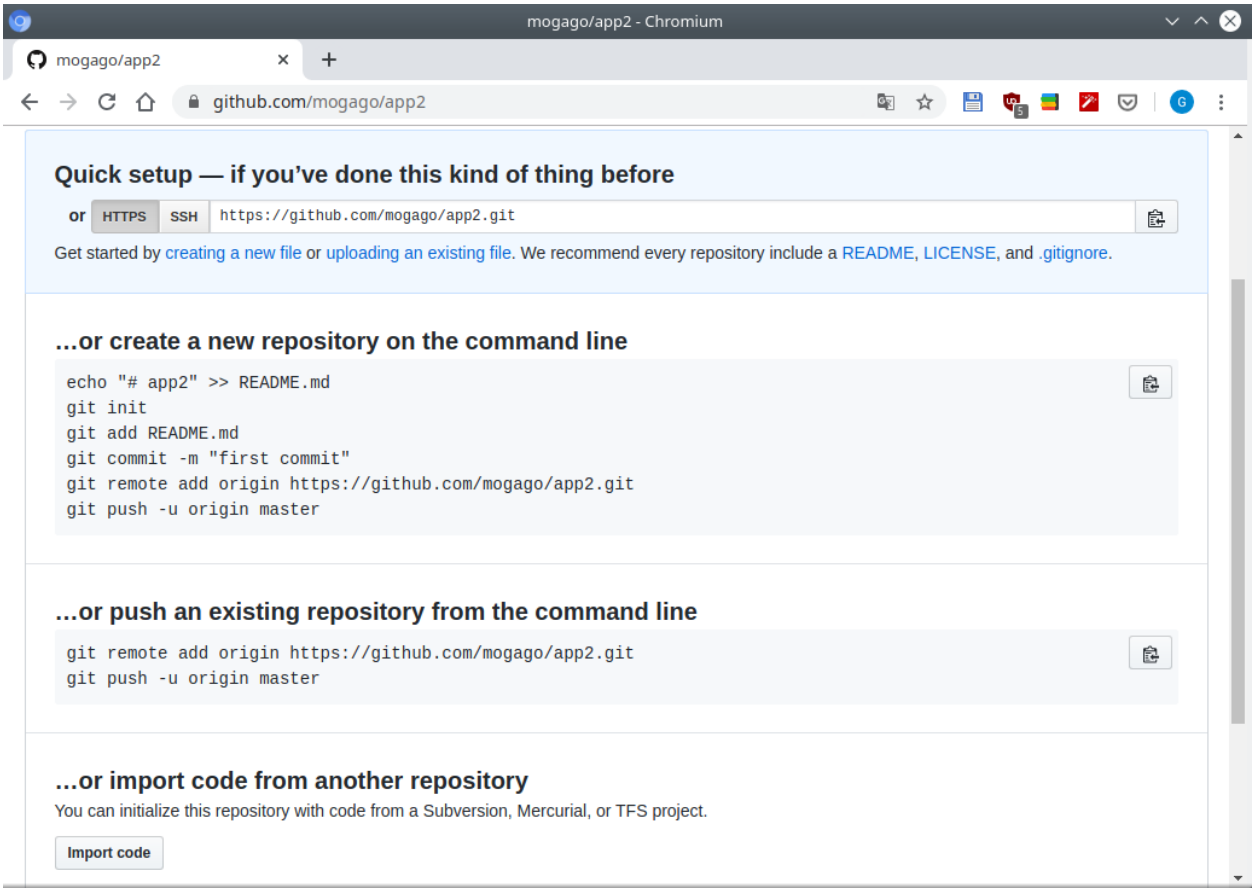


Fig. 13: GitHub - create new repository - Paso 4

En un terminal catalogar los contenidos del archivo `config` del directorio `.git`, **este directorio es nuestro repositorio**. Si eliminamos este directorio, nuestro historial, branches, tags y más se irá con él incluyendo nuestro archivo de configuración local `.git/confi`:

```
$ cat .git/config
[core]
        repositoryformatversion = 0
        filemode = true
        bare = false
        logallrefupdates = true
```

Así como vimos el archivo de configuración global de Git `~/.gitconfig`, tenemos el archivo de configuración local de Git `.git/config`.

Veamos que ocurre cuando agregamos un remote. Copiemos el archivo de GitHub a nuestro terminal:

```
$ git remote add origin https://github.com/mogago/app2.git
```

Este comando indica que queremos agregar un lugar remoto donde podamos interactuar con nuestro repositorio de Git. Los remotes pueden ser engañosos porque se pueden configurar en otro directorio de nuestro disco duro, es decir, o debe ser realmente remoto a través de Internet, pero lo será la mayoría de las veces.

Llamamos a nuestro remote `origin` como alias que apunta a la dirección `https://github.com/mogago/app2.git`. Veremos que nuestro configuración local tiene algo extra:

```
$ cat .git/config
[core]
        repositoryformatversion = 0
        filemode = true
        bare = false
        logallrefupdates = true
[remote "origin"]
        url = https://github.com/mogago/app2.git
        fetch = +refs/heads/*:refs/remotes/origin/*
```

Ahora tendremos un `remote "origin"` con una URL y un **Refspecs**, que indica cómo conectar branches en el servidor remoto a los branches locales. Ahora podemos correr el último comando de GitHub:

```
$ git push -u origin master

Username for 'https://github.com':
Password for 'https://mogago@github.com':
Counting objects: 5, done.
Delta compression using up to 4 threads.
Compressing objects: 100% (3/3), done.
Writing objects: 100% (5/5), 449 bytes | 449.00 KiB/s, done.
Total 5 (delta 0), reused 0 (delta 0)
To https://github.com/mogago/app2.git
* [new branch]      master -> master
Branch 'master' set up to track remote branch 'master' from 'origin'.
```

Si solo hubiésemos corrido `git push origin master` hubiese hecho push de una copia de todos nuestro cambios y la información de nuestro repositorio en el servidor remoto de GitHub. Específicamente tomaría todo el contenido del branch `master` y apuntará al branch `master` del servidor remoto. Pero usando `git push -u` no solo subimos los archivos a GitHub, también estamos configurando un **default upstream** (`-u`) para este branch:

Veamos el contenido de nuestro archivo de configuración local:

```
$ cat .git/config
[core]
        repositoryformatversion = 0
        filemode = true
        bare = false
        logallrefupdates = true
[remote "origin"]
        url = https://github.com/mogago/app2.git
        fetch = +refs/heads/*:refs/remotes/origin/*
[branch "master"]
        remote = origin
        merge = refs/heads/master
```

Ahora tenemos 3 nuevas líneas que nos indican que para este repo, si estamos en el branch `master`, cuando hagamos `git push` o `git pull` por defecto se debe conectar al servidor remote `"origin"` y que obtenga o entregue los contenidos del branch `master`.

Si usáramos `git pull` sin indicar una dirección, por defecto sabrá que debe hacer pull de `origin master`:

```
$ git pull
Already up to date.
```

3.6.8 Configuring your push default

Hagamos `git push`.

Antes de la versión de Git 2.0, el `push` default estaba configurado a 'matching'. Digamos que tenemos 2 branches locales `master` y `feature1`. Si estamos en uno de estos branches, si hiciéramos `git push` haría push no solo de este branch si no también de todos los demás, por ejemplo del branch `master`. Este modo se llama `matching`.

Desde la versión Git 2.0 se cambió el push por defecto a `simple`, es decir, si hacemos push en un branch solo hará push del branch donde nos encontramos.

Note: Para versiones anteriores a Git 2.0 cambiar el push por defecto a `simple`:

```
$ git config --global push.default simple
```

3.6.9 Fetch versus pull

Miremos las diferencias entre `fetch` y `pull`. Comencemos creando un escenario.

En GitHub, actualizar la página web del repositorio. Se recomienda tener un archivo `README`, así que lo crearemos, haciendo clic en el botón *Create new file*:

Crear el archivo `README.md` con el editor de GitHub integrado. Este archivo sirve para compartir información del proyecto en el home:

Dejemos el mensaje de commit por defecto y clic en el botón *Commit new file*:

Veremos el archivo `README` en el home del proyecto:

Veamos los branches:

```
$ git branch
* master
```

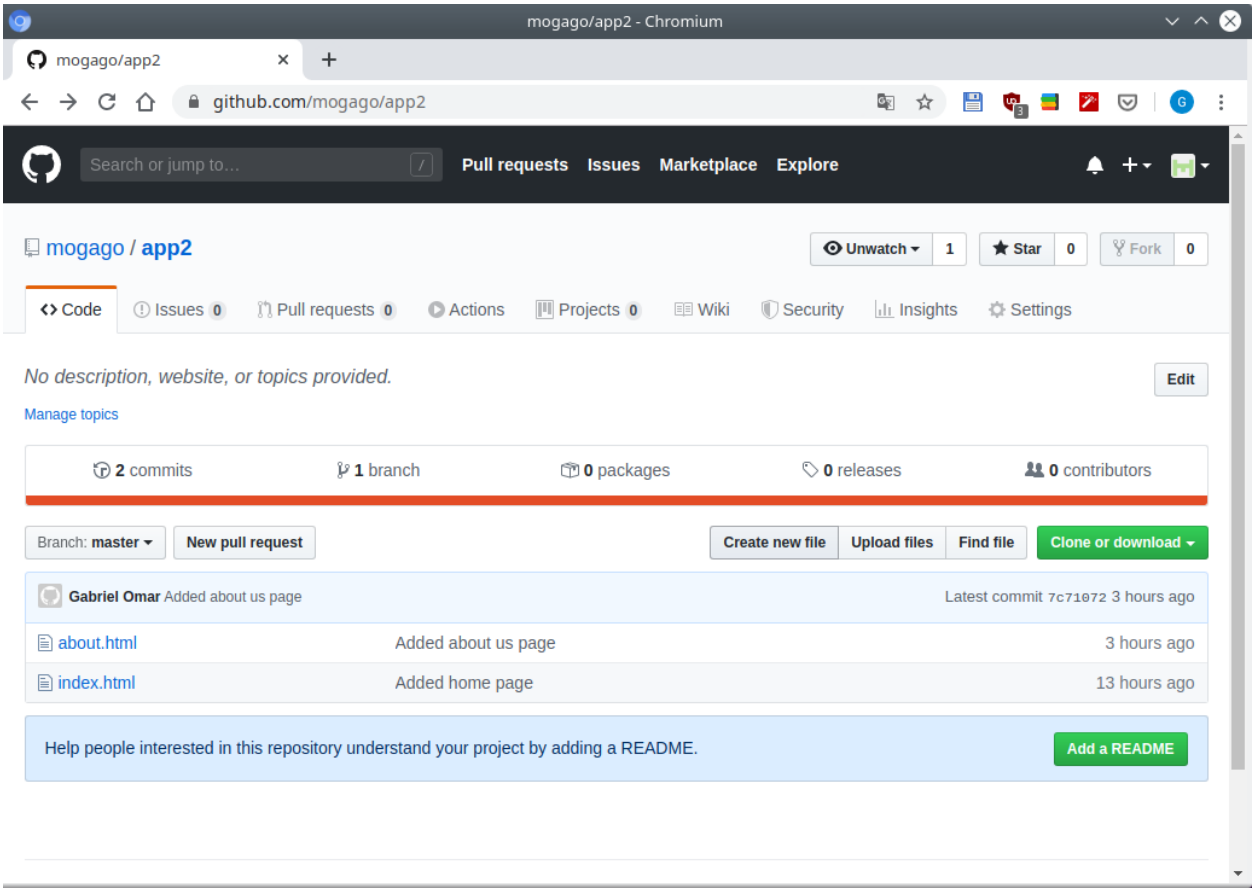


Fig. 14: GitHub - create new file - Paso 1

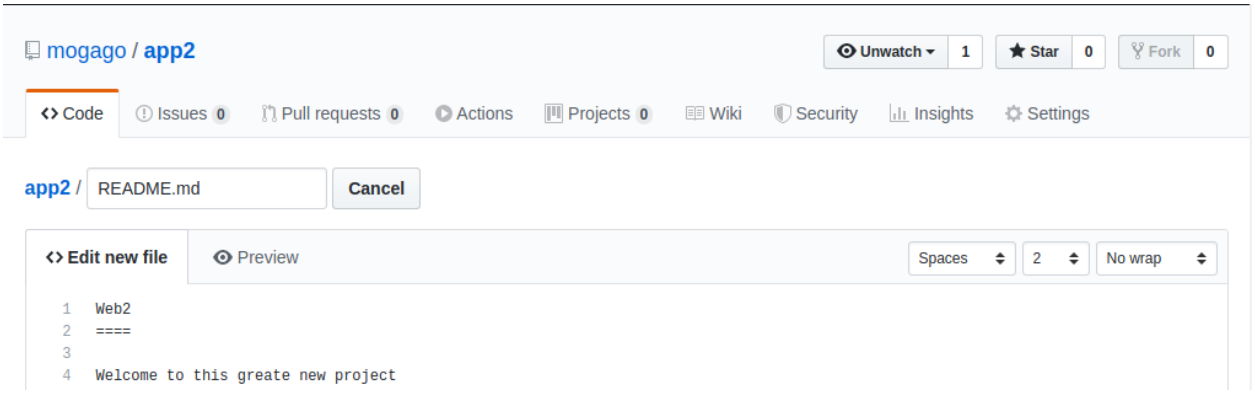


Fig. 15: GitHub - create new file - Paso 2

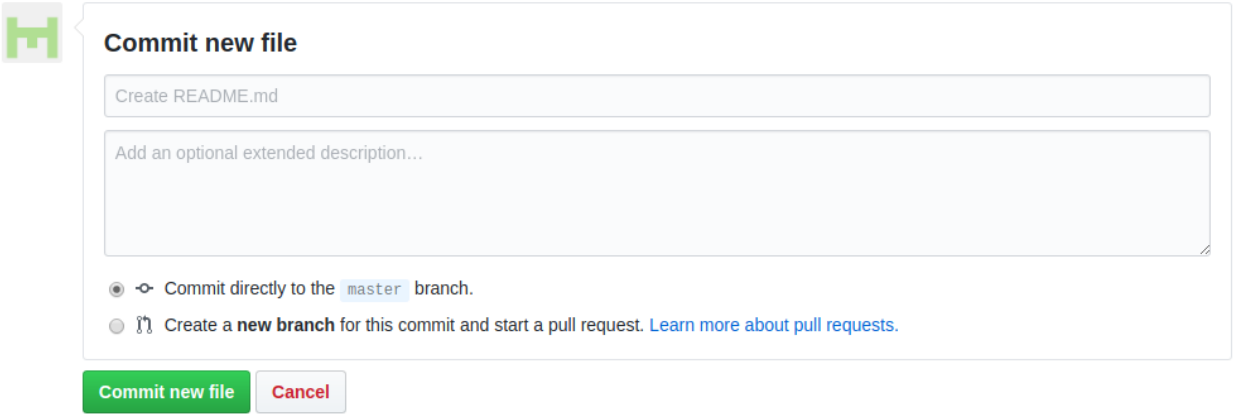


Fig. 16: GitHub - create new file - Paso 3

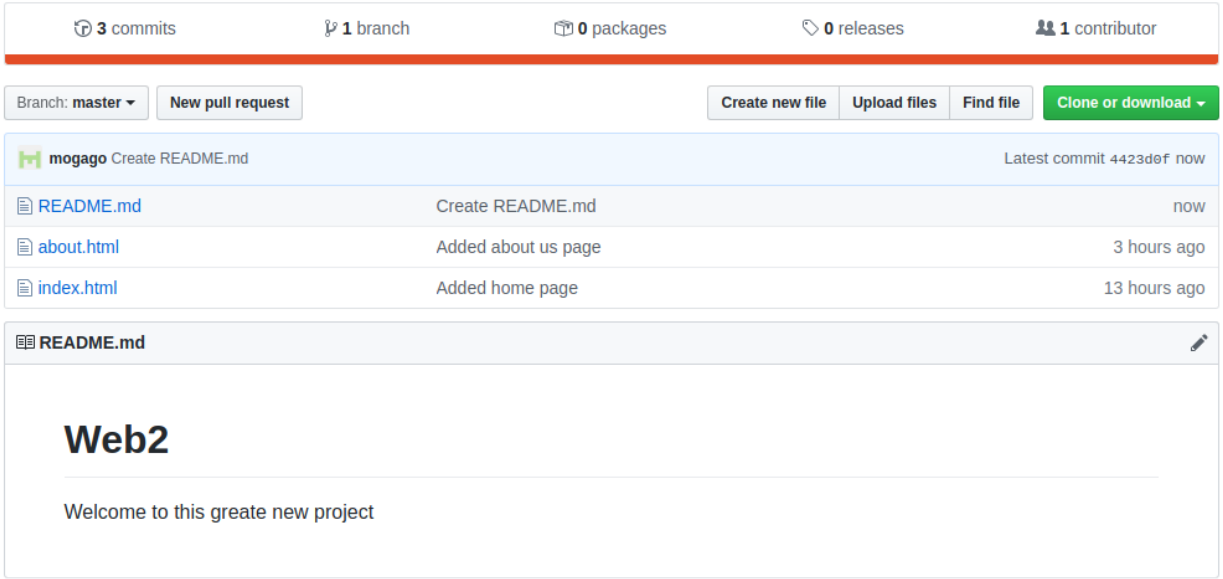


Fig. 17: GitHub - create new file - Paso 4

Pero ahora usemos la opción `-a`:

```
$ git branch -a
* master
  remotes/origin/master
```

Tenemos el branch local `master` pero también un branch denominado "**remote tracking branch**" (`remote/origin/master`).

Si vemos el log nos explicará por qué esta ahí y qué hace:

```
$ git lg
* 7c71072 (HEAD -> master, origin/master) Added about us page
* 6c4de50 Added home page
```

Vemos que los branches `origin/master` y `master` están apuntando al mismo `commit`.

Note: Cabe señalar que a pesar que vemos este branch remoto, Git no realiza una búsqueda por la red en Internet.

Este "remote tracking branch" es una copia local de todas las cosas que estaban en nuestro servidor remoto hasta la última vez que hablamos con él (le hicimos `pull` o clonamos el repositorio).

Algunas veces haremos checkout al branch `origin/master` para ver qué cambios están hanciendo en el servidor remoto y decir si lo pasamos a nuestro trabajo. Hay 2 formas en que podemos hacer esto:

1. Usualmente usamos `git pull` para obtener los cambios de GitHub y hacer merge en nuestro branch local `master`:

```
$ git pull

remote: Enumerating objects: 4, done.
remote: Counting objects: 100% (4/4), done.
remote: Compressing objects: 100% (2/2), done.
remote: Total 3 (delta 0), reused 0 (delta 0), pack-reused 0
Unpacking objects: 100% (3/3), done.
From https://github.com/mogago/app2
    7c71072..4423d0f  master     -> origin/master
Updating 7c71072..4423d0f
Fast-forward
 README.md | 4 ++++
 1 file changed, 4 insertions(+)
 create mode 100644 README.md

$ git lg
* 4423d0f (HEAD -> master, origin/master) Create README.md
* 7c71072 Added about us page
* 6c4de50 Added home page
```

Vemos que el log se habrá actualizado y tenemos un `commit` más y tanto `origin/master` y `master` están apuntando al nuevo `commit`.

2. Hay otra cosa que podemos hacer de forma más granular. Podemos hacer "**fetch**" (extraer) y luego merge.

En GitHub, editemos el archivo `README.md`:

Clic en *Commit new file*:

En GitHub veremos que tenemos 1 commit más, sin embargo en Git, seguiremos teniendo los mismos:

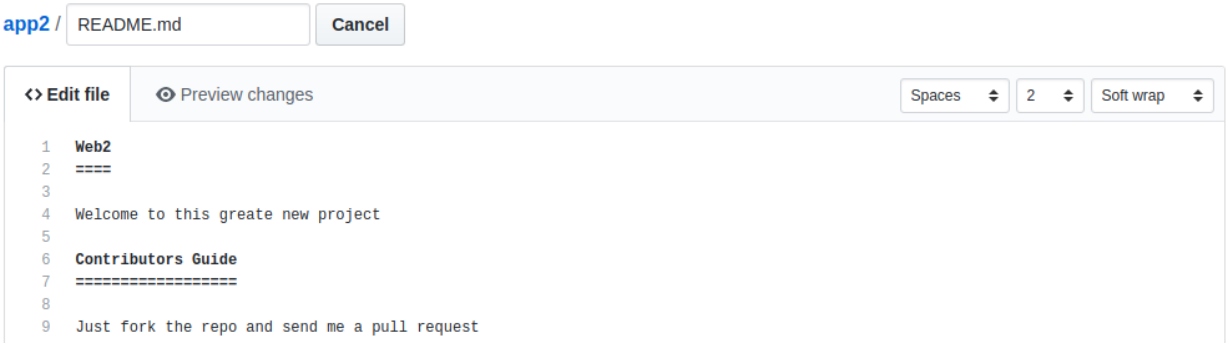


Fig. 18: GitHub - new commit

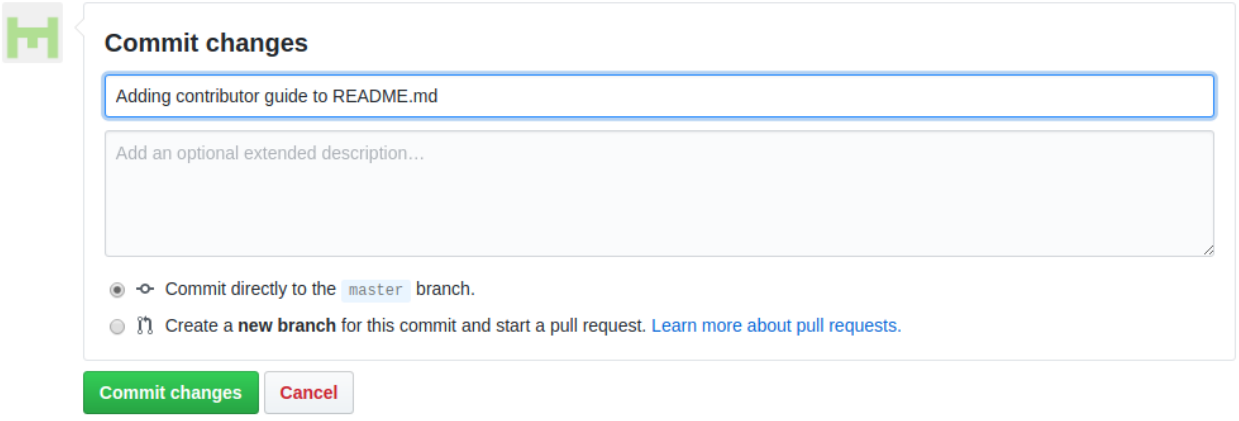


Fig. 19: GitHub - new commit

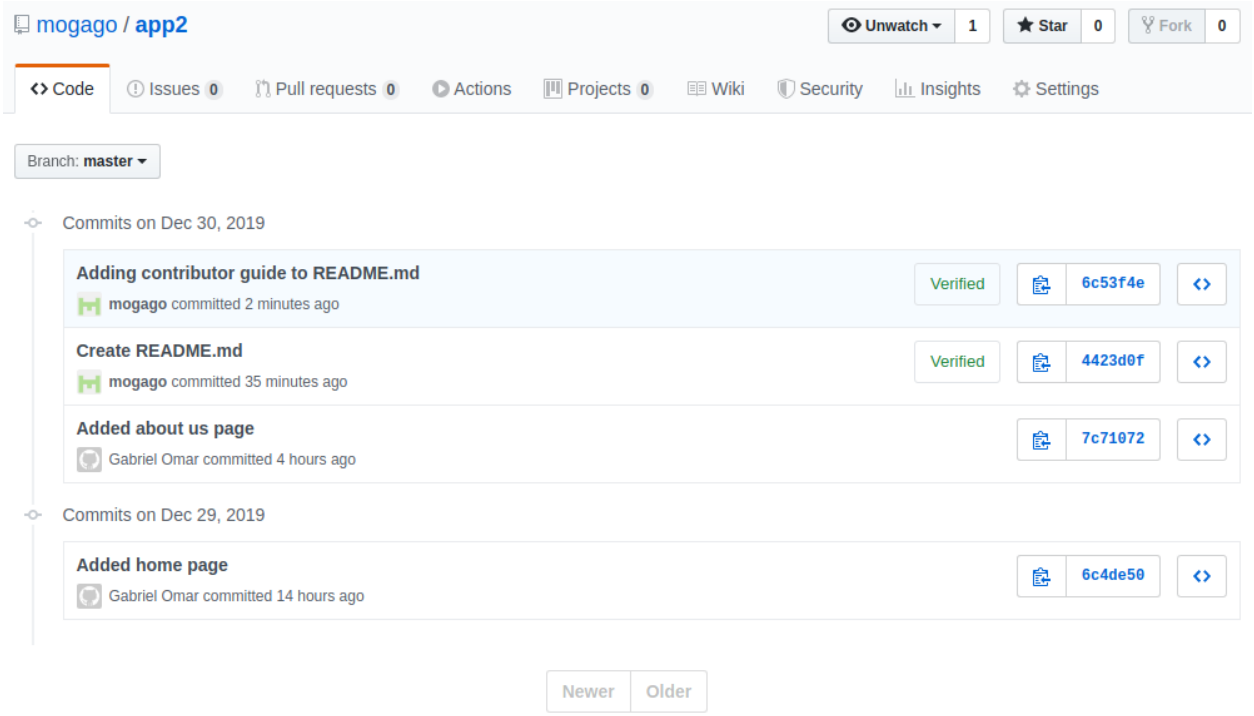


Fig. 20: GitHub - new commit

```
$  git lg
* 4423d0f (HEAD -> master, origin/master) Create README.md
* 7c71072 Added about us page
* 6c4de50 Added home page
```

Ahora seamos más cuidadosos usando `git fetch`. Esta es en realidad la operación de red. Cuando decimos `git pull`, en realidad estamos corriendo 2 comandos: fetch y merge. Fetch para que obtenga la información de Internet y merge los cambios de `origin/master` a nuestro branch `master`.

```
$ git fetch

remote: Enumerating objects: 5, done.
remote: Counting objects: 100% (5/5), done.
remote: Compressing objects: 100% (3/3), done.
remote: Total 3 (delta 0), reused 0 (delta 0), pack-reused 0
Unpacking objects: 100% (3/3), done.
From https://github.com/mogago/app2
    4423d0f..6c53f4e  master     -> origin/master

$ git lg
* 6c53f4e (origin/master) Adding contributor guide to README.md
* 4423d0f (HEAD -> master) Create README.md
* 7c71072 Added about us page
* 6c4de50 Added home page
```

`git fetch` ha bajado los cambios desde Internet pero no ha hecho el merge en el branch `master`.

Si quisiéramos ver qué contiene es guía de contribuidores podríamos hacer checkout a `origin/master` como cualquier otro branch:

```
$ cat README.md
    Web2
    ====

    Welcome to this greate new project


$ git checkout origin/master
Note: checking out 'origin/master'.

You are in 'detached HEAD' state. You can look around, make experimental
changes and commit them, and you can discard any commits you make in this
state without impacting any branches by performing another checkout.

If you want to create a new branch to retain commits you create, you may
do so (now or later) by using -b with the checkout command again. Example:

git checkout -b <new-branch-name>

HEAD is now at 6c53f4e Adding contributor guide to README.md
```

Veamos el contenido del `README.md`

```
$ cat README.md
    Web2
    ====

    Welcome to this greate new project

    Contributors Guide
    ==================

    Just fork the repo and send me a pull request
```

Si estamos contentos con el contenido del archivo nos queda hacer merge (tipo "fast forward") hacia el `master`:

```
$ git checkout master
Previous HEAD position was 6c53f4e Adding contributor guide to README.md
Switched to branch 'master'
Your branch is behind 'origin/master' by 1 commit, and can be fast-forwarded.
  (use "git pull" to update your local branch)

$ git merge origin/master
Updating 4423d0f..6c53f4e
Fast-forward
 README.md | 5 +++++
 1 file changed, 5 insertions(+)
```

Comprobemos que tenemos el archivo actualizado desde el branch `master`:

```
$ cat README.md
    Web2
    ====

    Welcome to this greate new project

    Contributors Guide
```

(continues on next page)

(continued from previous page)

```
    ==================

    Just fork the repo and send me a pull request

$ git lg
* 6c53f4e (HEAD -> master, origin/master) Adding contributor guide to README.md
* 4423d0f Create README.md
* 7c71072 Added about us page
* 6c4de50 Added home page
```

3.6.10 Merge versus rebase on pull

La última cosa que veremos sobre fetching y mergin es: merging versus rebasing cuando hacemos `pull`.

Acabamos de ver el caso que hacemos fetch y luego merge hacia el branch `master`. ¿Qué hubiese pasado si hubiéramos tenido nuestro propio `commit` en el branch `master`?

Volvamos a editar el archivo README.md desde GitHub con el mensaje de `commit` por defecto:

```
Web2
====

Welcome to this greate new project

Contributors Guide
==================

Just fork the repo and send me a pull request

Add more text
```

Veamos los commits en GitHub:

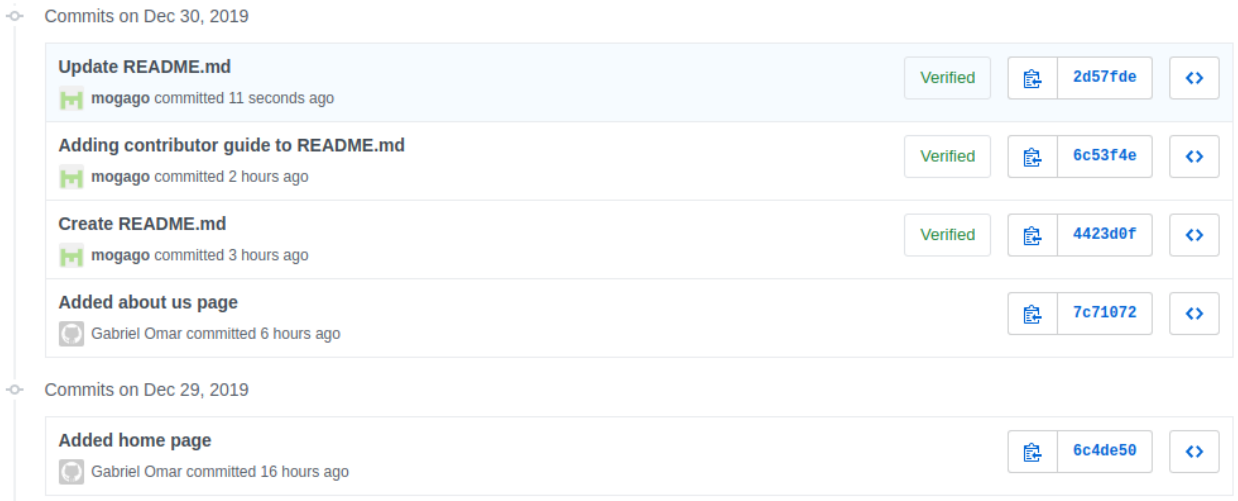


Fig. 21: GitHub - new commit

Ahora localmente tenemos un `commit` menos:

```
$ git lg
* 6c53f4e (HEAD -> master, origin/master) Adding contributor guide to README.md
* 4423d0f Create README.md
* 7c71072 Added about us page
* 6c4de50 Added home page
```

Hagamos fetch y veamos el log:

```
$ git fetch
remote: Enumerating objects: 5, done.
remote: Counting objects: 100% (5/5), done.
remote: Compressing objects: 100% (3/3), done.
remote: Total 3 (delta 1), reused 0 (delta 0), pack-reused 0
Unpacking objects: 100% (3/3), done.
From https://github.com/mogago/app2
    6c53f4e..2d57fde  master     -> origin/master

$ git lg
* 2d57fde (origin/master) Update README.md
* 6c53f4e (HEAD -> master) Adding contributor guide to README.md
* 4423d0f Create README.md
* 7c71072 Added about us page
* 6c4de50 Added home page
```

Imaginemos que hagamos un `commit` localmente:

```
$ vi index.html
    hello world
    welcome to our website!

$ git commit -am "Added longer welcome to the home page"
[master 1e4341a] Added longer welcome to the home page
 1 file changed, 1 insertion(+)

$ git lg
* 1e4341a (HEAD -> master) Added longer welcome to the home page
| * 2d57fde (origin/master) Update README.md
|/
* 6c53f4e Adding contributor guide to README.md
* 4423d0f Create README.md
* 7c71072 Added about us page
* 6c4de50 Added home page
```

Antes hicimos un cambios en el branch `origin/master` desde GitHub, pero ahora hemos hecho un cambio localmente en nuestro branch `master`. Encontrando una divergencia en el historial.

Si corriéramos `git pull` o `git merge origin/master` no haría un fast forward y tendríamos un mensaje de `commit` no aconsejable. Funciona pero está mal:

```
$ git merge origin/master

    Merge remote-tracking branch 'origin/master'

    # Please enter a commit message to explain why this merge is necessary,
    # especially if it merges an updated upstream into a topic branch.
    #
    # Lines starting with '#' will be ignored, and an empty message aborts
```

(continues on next page)

(continued from previous page)

```
    # the commit.

Merge made by the 'recursive' strategy.
README.md | 2 ++
1 file changed, 2 insertions(+)

$ git lg
*   6a4c9da (HEAD -> master) Merge remote-tracking branch 'origin/master'
|\
| * 2d57fde (origin/master) Update README.md
* | 1e4341a Added longer welcome to the home page
|/
* 6c53f4e Adding contributor guide to README.md
* 4423d0f Create README.md
* 7c71072 Added about us page
* 6c4de50 Added home page
```

No es aconsejable porque no tenemos una información semática útil. Así que borremos el último `commit`:

```
$ git reset --hard HEAD~1
HEAD is now at 1e4341a Added longer welcome to the home page

$ git lg
* 1e4341a (HEAD -> master) Added longer welcome to the home page
| * 2d57fde (origin/master) Update README.md
|/
* 6c53f4e Adding contributor guide to README.md
* 4423d0f Create README.md
* 7c71072 Added about us page
* 6c4de50 Added home page
```

Hemos vuelto al punto de referencia anterior. Queremos juntar esas dos líneas de historia pero no queremos hacer merge, es decir, hacer que parezca como si viéramos cosa por cosa pero en realidad estamos haciendo más de una cosa al mismo tiempo. Esto lo haremos usando **rebase**.

En el branch `master` usamos rebase:

```
$ git rebase origin/master
First, rewinding head to replay your work on top of it...
Applying: Added longer welcome to the home page

$ git lg
* 42c4494 (HEAD -> master) Added longer welcome to the home page
* 2d57fde (origin/master) Update README.md
* 6c53f4e Adding contributor guide to README.md
* 4423d0f Create README.md
* 7c71072 Added about us page
* 6c4de50 Added home page
```

Hemos cambiado la base del `master` a `2d57fde`, como si no hubiéramos comenzado a programar localmente hasta que `origin/master` haya acabado su trabajo.

Ahora actualizaremos nuestro trabajo en el servidor remoto con `push`:

```
$ git push
Username for 'https://github.com': mogago
Password for 'https://mogago@github.com':
```

(continues on next page)

(continued from previous page)

```
Counting objects: 3, done.
Delta compression using up to 4 threads.
Compressing objects: 100% (2/2), done.
Writing objects: 100% (3/3), 313 bytes | 313.00 KiB/s, done.
Total 3 (delta 1), reused 0 (delta 0)
remote: Resolving deltas: 100% (1/1), completed with 1 local object.
To https://github.com/mogago/app2.git
    2d57fde..42c4494  master -> master
```

En lugar de usar fetch y merge podemos hacer un fetch y rebase en un solo comando:

```
$ git pull --rebase
```

3.7 Lesson 7

Lesson 7: Collaborating via GitHub

Table of Contents

3.7.1 Cloning a repository

Hasta este momento solo hemos creado nuestros proyectos localmente y le hemos hecho `push` hacia GitHub. Pero es más común que trabajemos en proyectos que alguien más haya iniciado:

Un usuario distinto al nuestro ha creado su repositorio:

```
$ git status
fatal: not a git repository (or any of the parent directories): .git

$ mkdir clone_me
$ cd clone_me/
```

(continues on next page)

(continued from previous page)

```
$ touch index.html
$ git init
Initialized empty Git repository in /home/user/Documents/gittests/clone_me/.git/

$ git add .
$ git commit -m "Added home page"
[master (root-commit) 9568a2a] Added home page
 1 file changed, 0 insertions(+), 0 deletions(-)
 create mode 100644 index.html
```

El usuario, desde GitHub creó un repositorio con el mismo nombre `clone_me` y lo añadió a su repositorio local de Git creado. Presionar el botón +, opción *New Repository*, usar el nombre `clone_me` y clic en el botón *Create repository*:

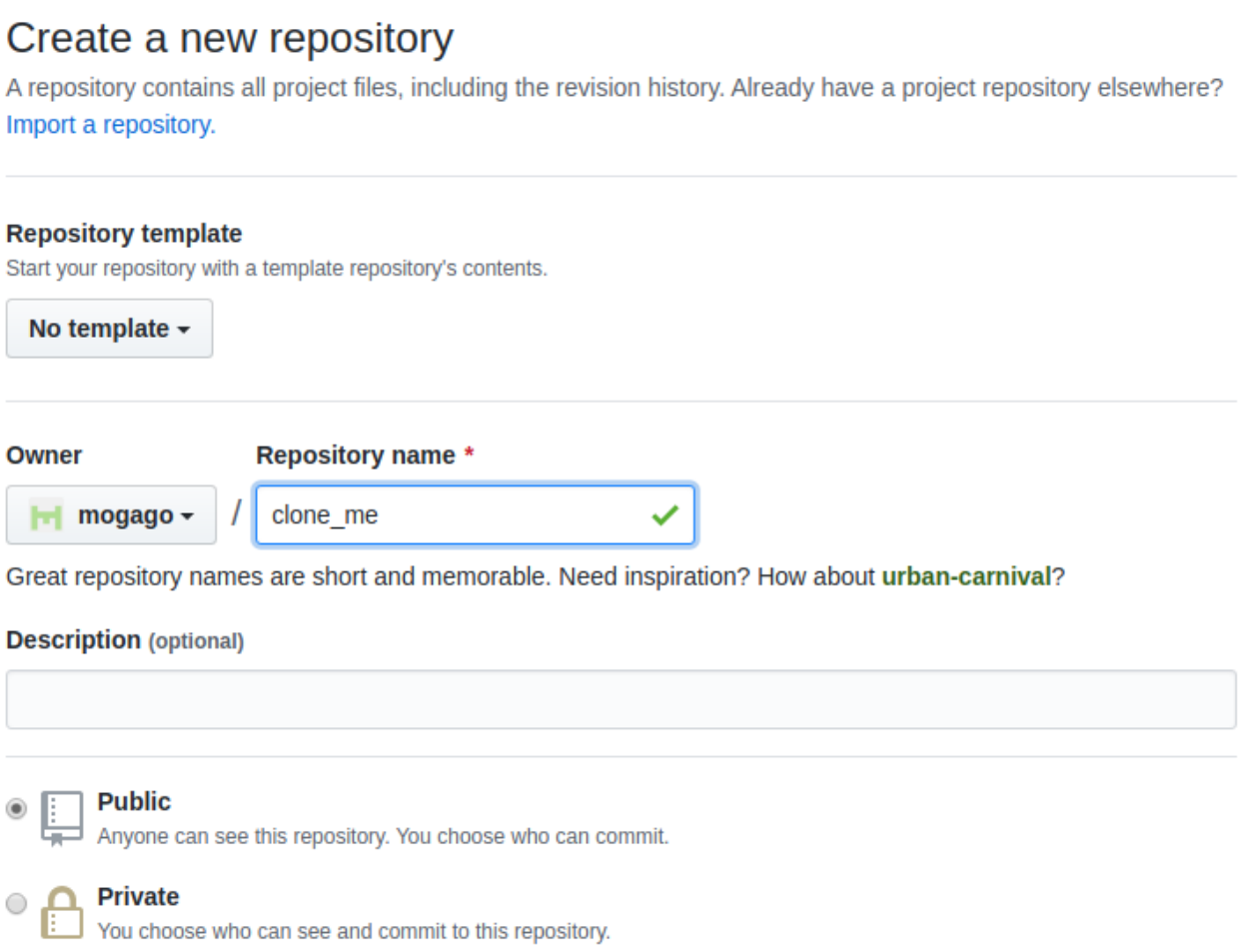


Fig. 22: GitHub - create new repository `clone_me`

En el terminal, el usuario ejecutó:

```
$ git remote add origin https://github.com/mogago/clone_me.git
$ git push -u origin master
```

Desde un lugar que no sea un directorio Git haremos una copia local de este repositorio externo:

```
$ cd ..
$ mkdir student
$ cd student/
$ git status
fatal: not a git repository (or any of the parent directories): .git

$ git clone https://github.com/mogago/clone_me
Cloning into 'clone_me'...
remote: Enumerating objects: 3, done.
remote: Counting objects: 100% (3/3), done.
remote: Total 3 (delta 0), reused 3 (delta 0), pack-reused 0
Unpacking objects: 100% (3/3), done.
```

Ahora tenemos nuestra propia copia del repositorio y podemos hacer cualquier cambio localmente. Veamos el log:

```
$ cd clone_me/
$ git lg
* 9568a2a (HEAD -> master, origin/master, origin/HEAD) Added home page

$ touch about.html
$ git add .
$ git commit -m "Added about as page"
[master f76a9a0] Added about as page
 1 file changed, 0 insertions(+), 0 deletions(-)
 create mode 100644 about.html
```

Pero si ahora tratamos de hacer `git push` obtendremos un error de autenticación porque somos un usuario distinto. No somo invitados como colaboradores a este proyecto.

El primer uso de cloning es tener una copia a la cual podemos hacerle cambios localmente.

3.7.2 Forking a repository

¿Qué pasa si hay un proyecto al que no tenemos acceso de hacer `push` pero queremos contribuir al proyecto?: creamos un **fork**.

```
$ git status
fatal: not a git repository (or any of the parent directories): .git

$ git init fork_me
Initialized empty Git repository in /home/gabriel/Documents/gittests/fork_me/.git/

$ cd fork_me/
$ touch index.html
$ git add .
$ git commit -m "Added home page to forkable repo"
[master (root-commit) 70b87af] Added home page to forkable repo
 1 file changed, 0 insertions(+), 0 deletions(-)
 create mode 100644 index.html
```

En GitHub:

En el terminal:

```
$ git remote add origin https://github.com/mogago/fork_me.git
$ git push -u origin master
```

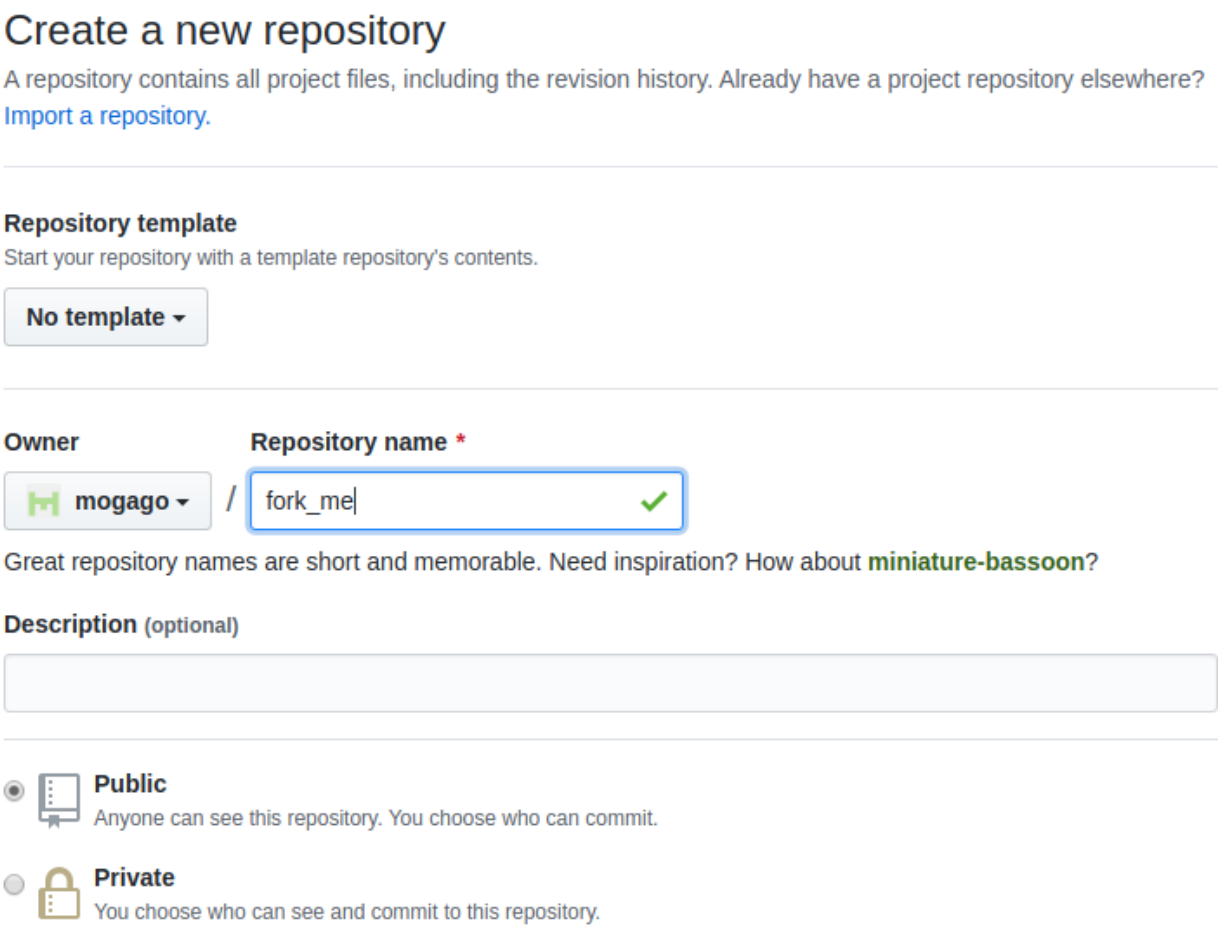


Fig. 23: GitHub - create new repository `fork_me`

Desde una ventan en incógnito sin loguearnos o usando un usuario distinto acceder al URL: https://github.com/mogago/fork_me.git

Note: ¿Cuál es la diferencia entre clone y fork?. Es como la pregunta ¿usar merge o rebase?, siempre harás merge pero la pregunta es si harás rebase primero. Es similiar con cloning y forking.

Siempre clonaremos un repositorio si queremos una copia local en nuestra laptop donde podamos hacer cambios. La única pregunta es si primero haremos fork al repositorio. Veamos algunos casos:

- Si no queremos hacer contribuciones a un proyecto, no requerimos hacer fork para tener una copia del repositorio, solo basta hacer cloning.
- Si es un repositorio al que hemos sido añadidos como colaboradores, no necesitamos hacer fork, solo hacer cloning y hacer `push` de vuelta con nuestros cambios porque alguien nos agregó como colaborador.
- Si queremos contribuir a un proyecto del cual desconocemos quienes lo crearon, deberemos hacer **fork** de ese repositorio. Esto es, tomaremos su copia del repositorio de Github y haremos nuestra propia copia del repositorio también en GitHub. Es decir el mismo proyecto bajo nuestro nombre de usuario (`https://github.com/anotheruser/fork_me.git`). Luego podemos clonarlo, hacerle cambios y hacerle `push` a nuestra propia copia, nuestro fork del repositorio.

Desde otra cuenta de GitHub, entrar a la URL del creador del repositorio: https://github.com/mogago/fork_me.git y clic en la opción *Fork*:

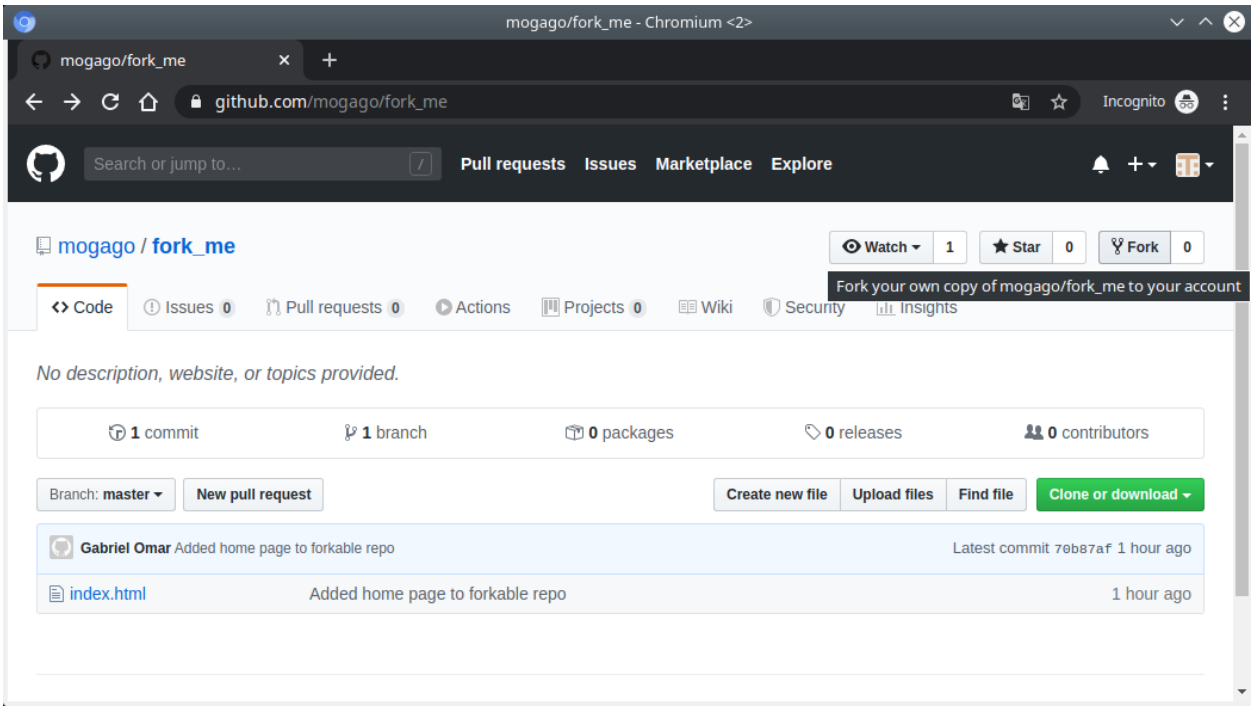


Fig. 24: GitHub - Fork de repositorio externo

Ahora veremos que hemos hecho un fork del repositorio en nuestra dashboard de GitHub:

Hagamos una copia local de nuestro repositorio, nuestro fork:

```
$ cd ..
$ cd student/
```

(continues on next page)

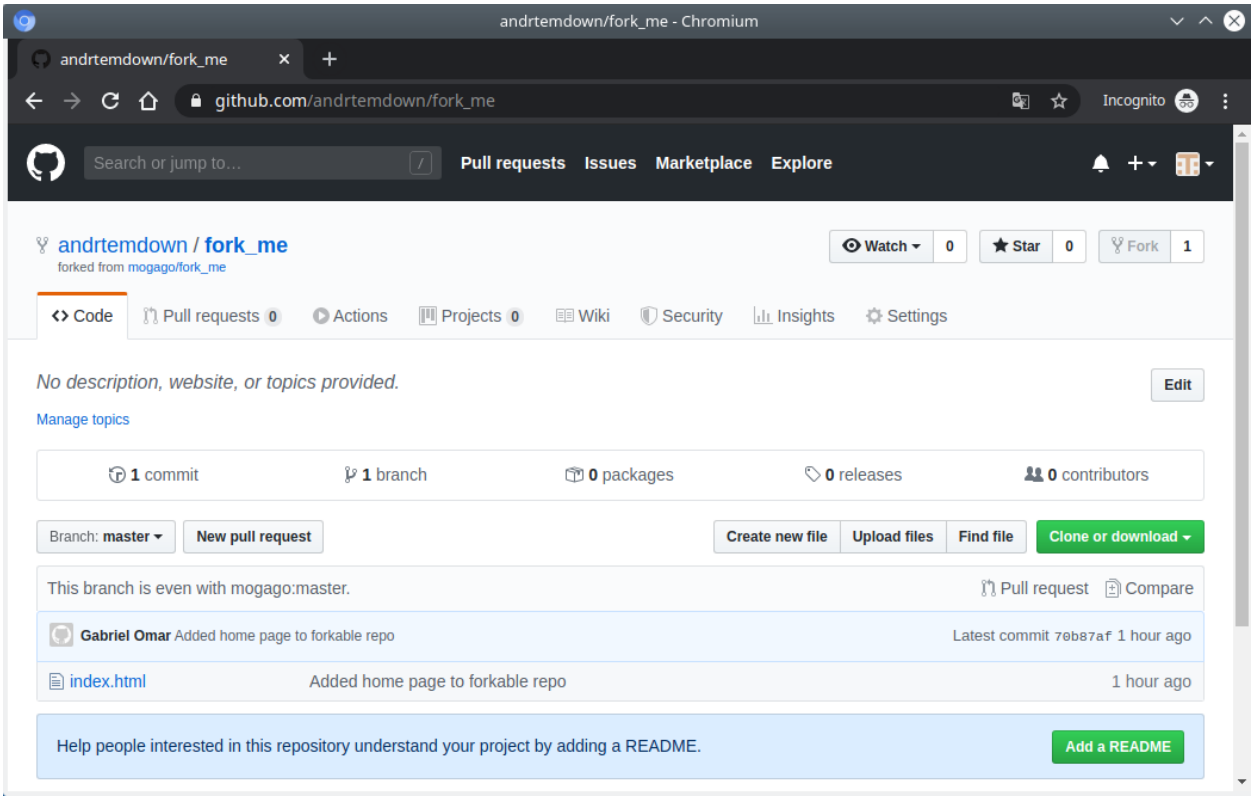


Fig. 25: GitHub - Fork de repositorio externo

(continued from previous page)

```
$ git status
fatal: not a git repository (or any of the parent directories): .git

$ git clone https://github.com/externaluser/fork_me
Cloning into 'fork_me'...
remote: Enumerating objects: 3, done.
remote: Counting objects: 100% (3/3), done.
remote: Total 3 (delta 0), reused 3 (delta 0), pack-reused 0
Unpacking objects: 100% (3/3), done.

$ cd fork_me/
```

Con nuestra copia, creemos un archivo con nuestro nombre de usuario:

```
$ git push
Username for 'https://github.com': externaluser
Password for 'https://externaluser@github.com':
Counting objects: 2, done.
Delta compression using up to 4 threads.
Compressing objects: 100% (2/2), done.
Writing objects: 100% (2/2), 272 bytes | 272.00 KiB/s, done.
Total 2 (delta 0), reused 0 (delta 0)
To https://github.com/externaluser/fork_me
    70b87af..241bd97  master -> master
```

Si vamos al repositorio original veremos que ninguno de nuestros cambios están aquí porque nuestros cambios se han

hecho en el fork:

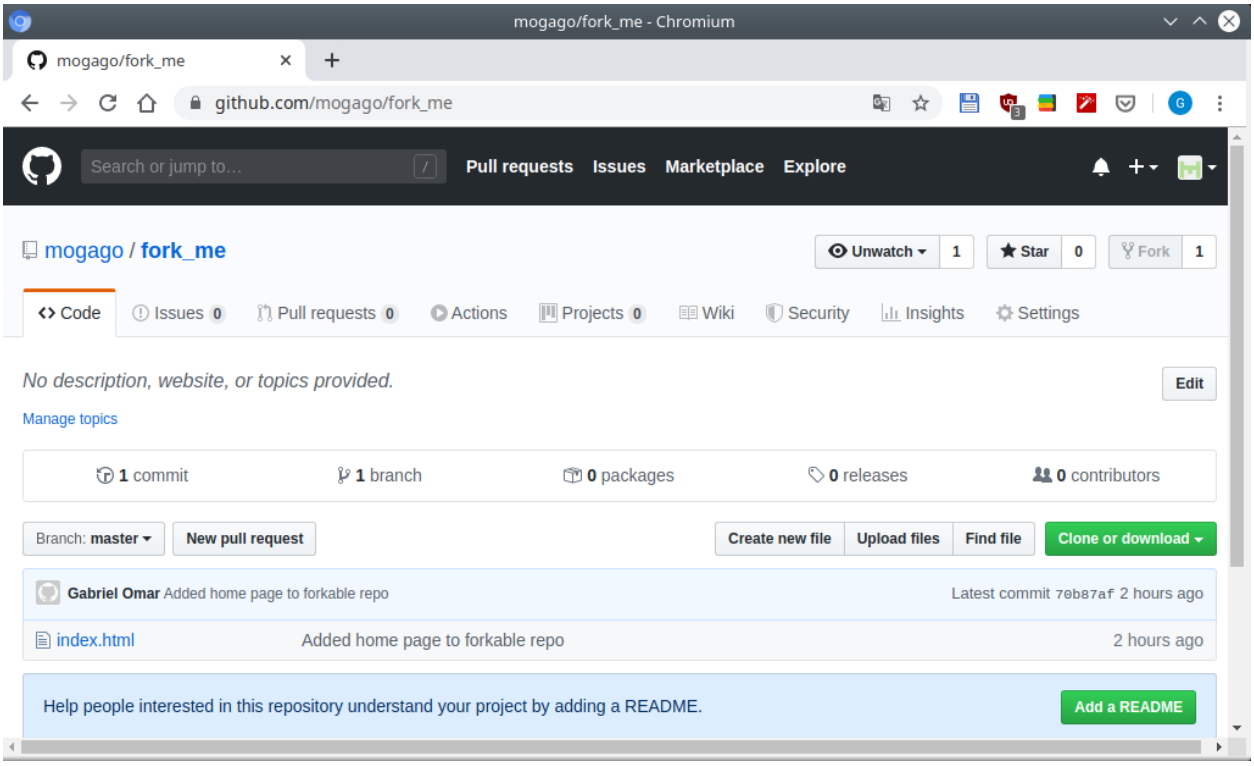


Fig. 26: GitHub - Repositorio original

En los commits de GitHub veremos 2 commits en el fork. Un fork hecho por el creador original y otro hecho por nosotros en nuestra copia local.

3.7.3 Contributing via a pull request from a fork

Una vez que he hecho fork de un repositorio, lo he clonado para tener una copia local, he hecho mis cambios y le he hecho `push` de vuelta a mi fork, ¿cómo haríamos para que nuestros cambios vayan al repositorio original?. Debemos usar un `pull request`.

El propósito de un `pull request` es pedir o preguntar a personas en otros proyectos para que incorporen nuestros cambios.

Con el usuario que ha creado el fork ingresar a nuestro fork desde GitHub y clic en el botón *New pull request*:

Ya que este es un fork de otro proyecto, GitHub supone que lo que queremos hacer es tomar los cambios del `master` branch que hemos hecho en el fork del repositorio y hacer merge con el `master` branch del repositorio original. Clic en botón *Create pull request*:

Ahora deberemos ingresar un mensaje donde fundamentamos por qué deben incluir nuestro código en ese repositorio. Clic en botón *Create pull request*:

Alguien que posea el proyecto podrá hacer merge con nuestros cambios.

3.7.4 Approving a pull request from a fork

Ahora logueados como usuarios propietarios del repositorio podemos ver la lista de `pull request`:

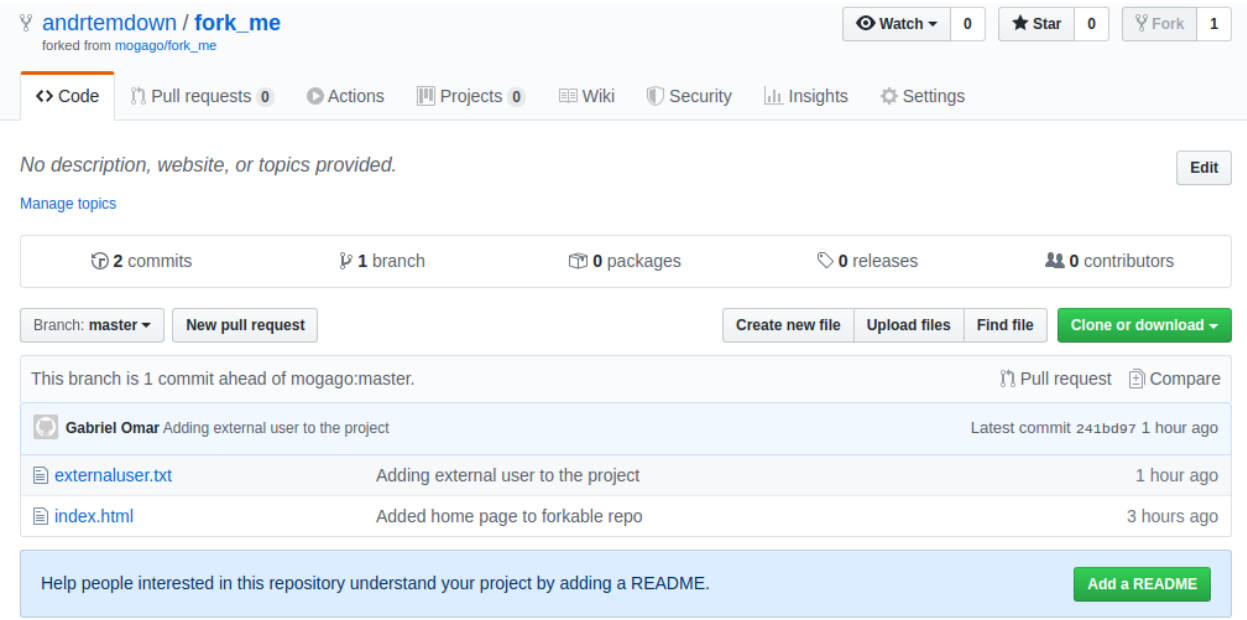


Fig. 27: GitHub - Repositorio original

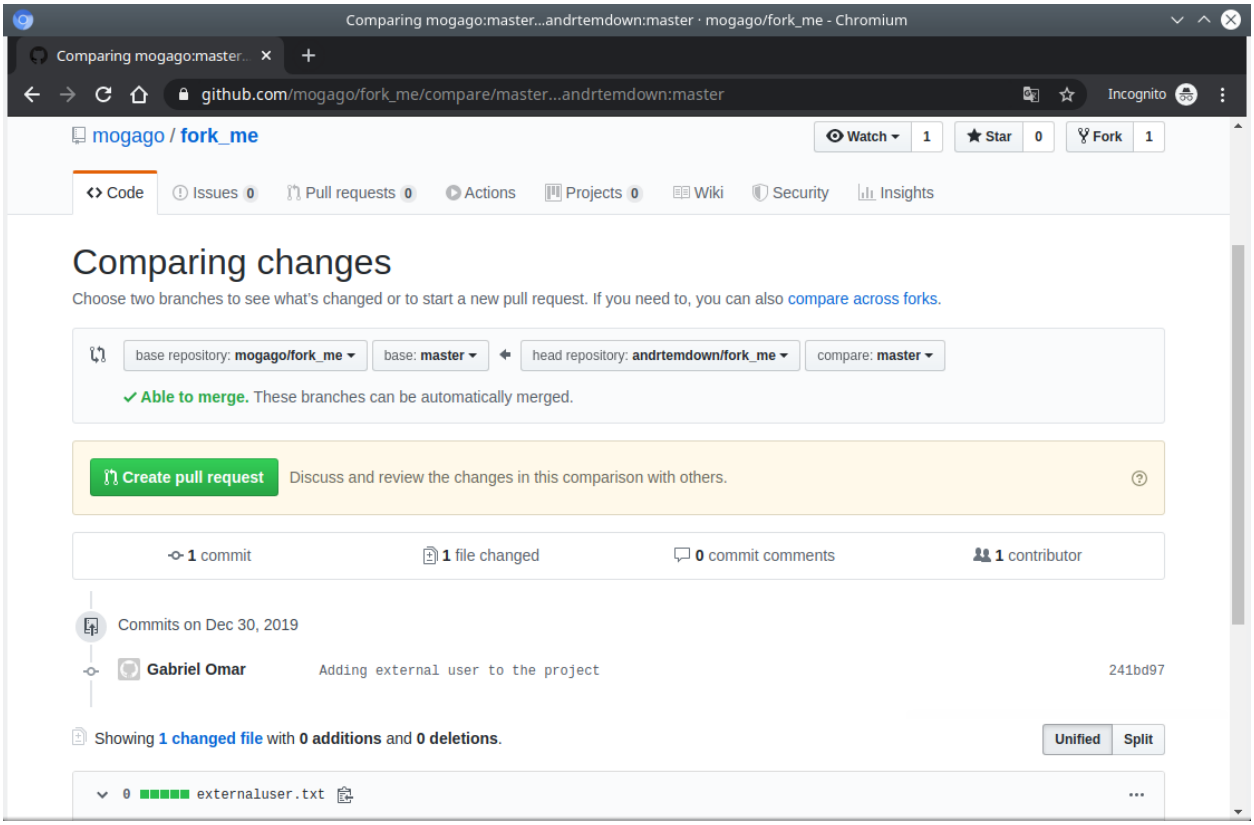


Fig. 28: GitHub - `pull request`

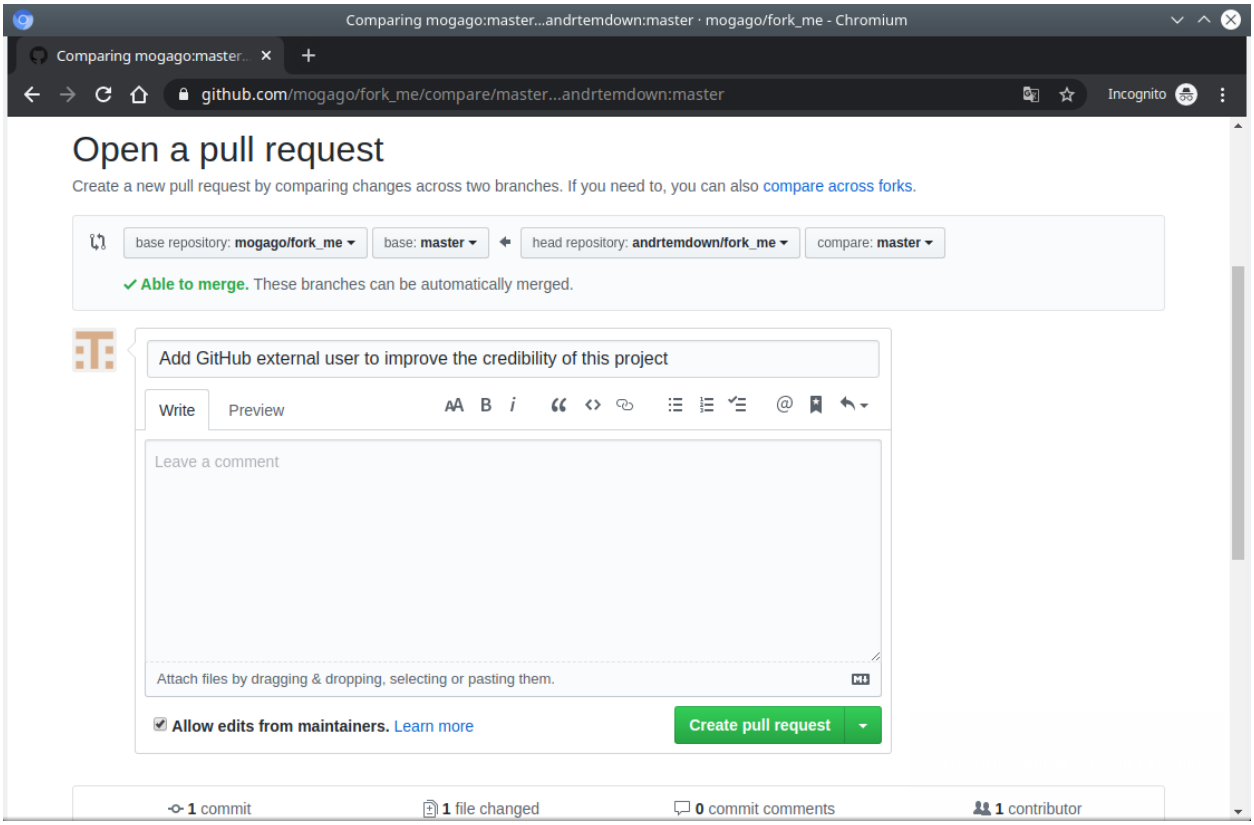


Fig. 29: GitHub - Mensaje `pull request`

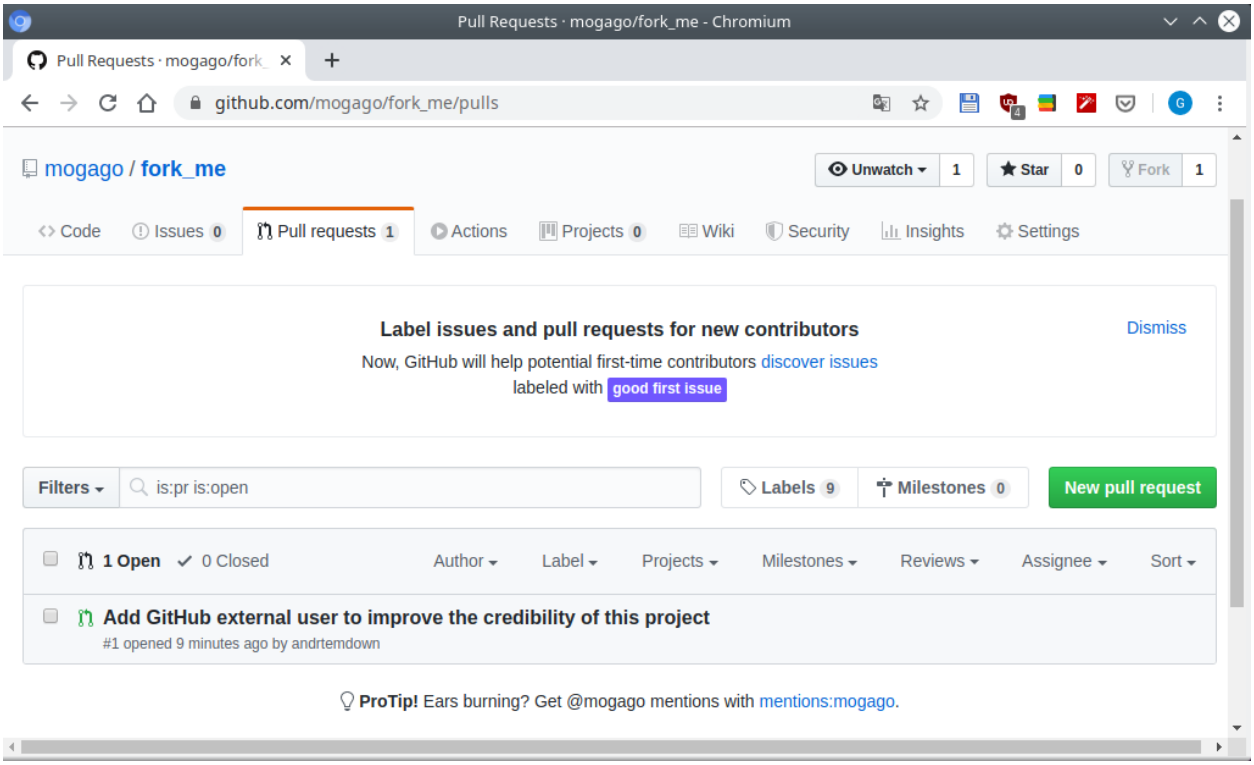


Fig. 30: GitHub - Lista de peticiones de `pull request`

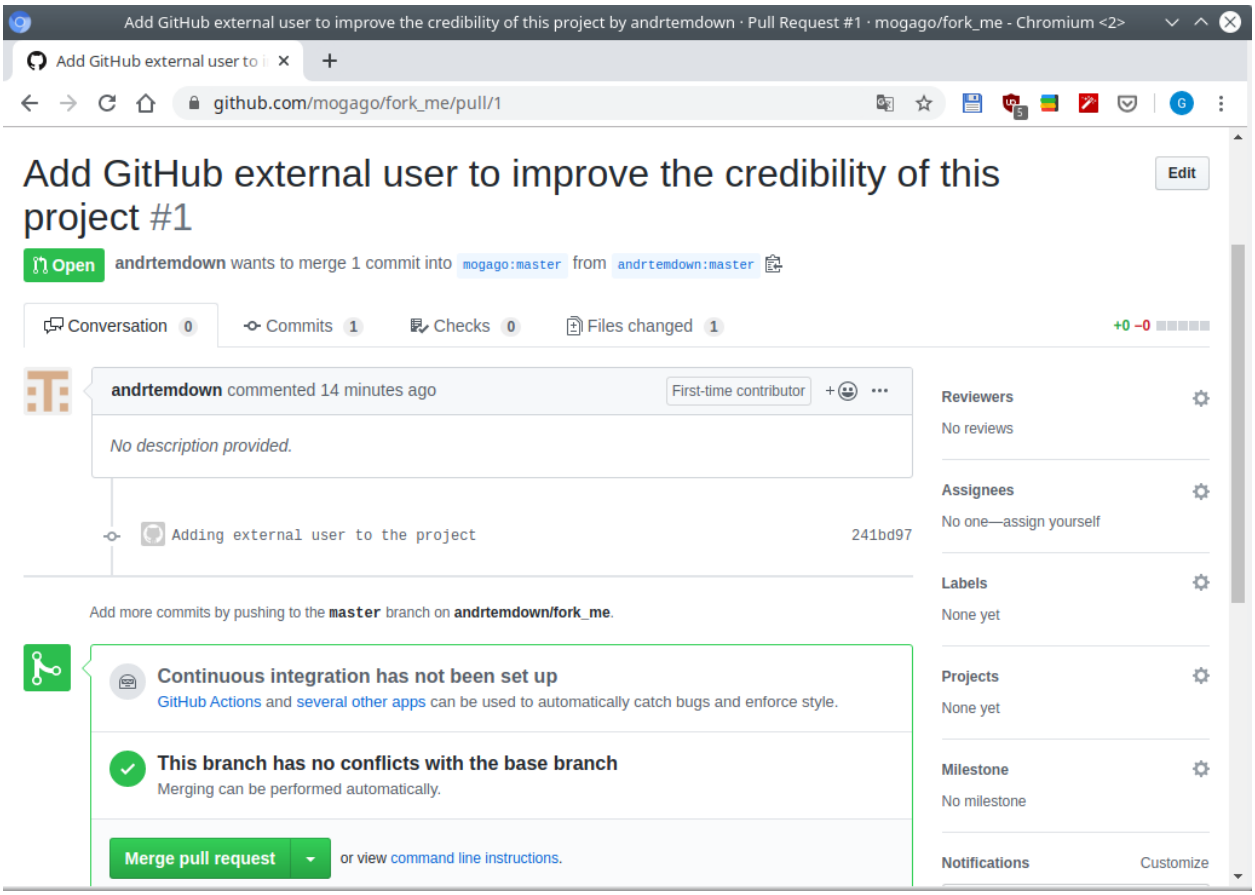


Fig. 31: GitHub - Aceptar petición `pull request`

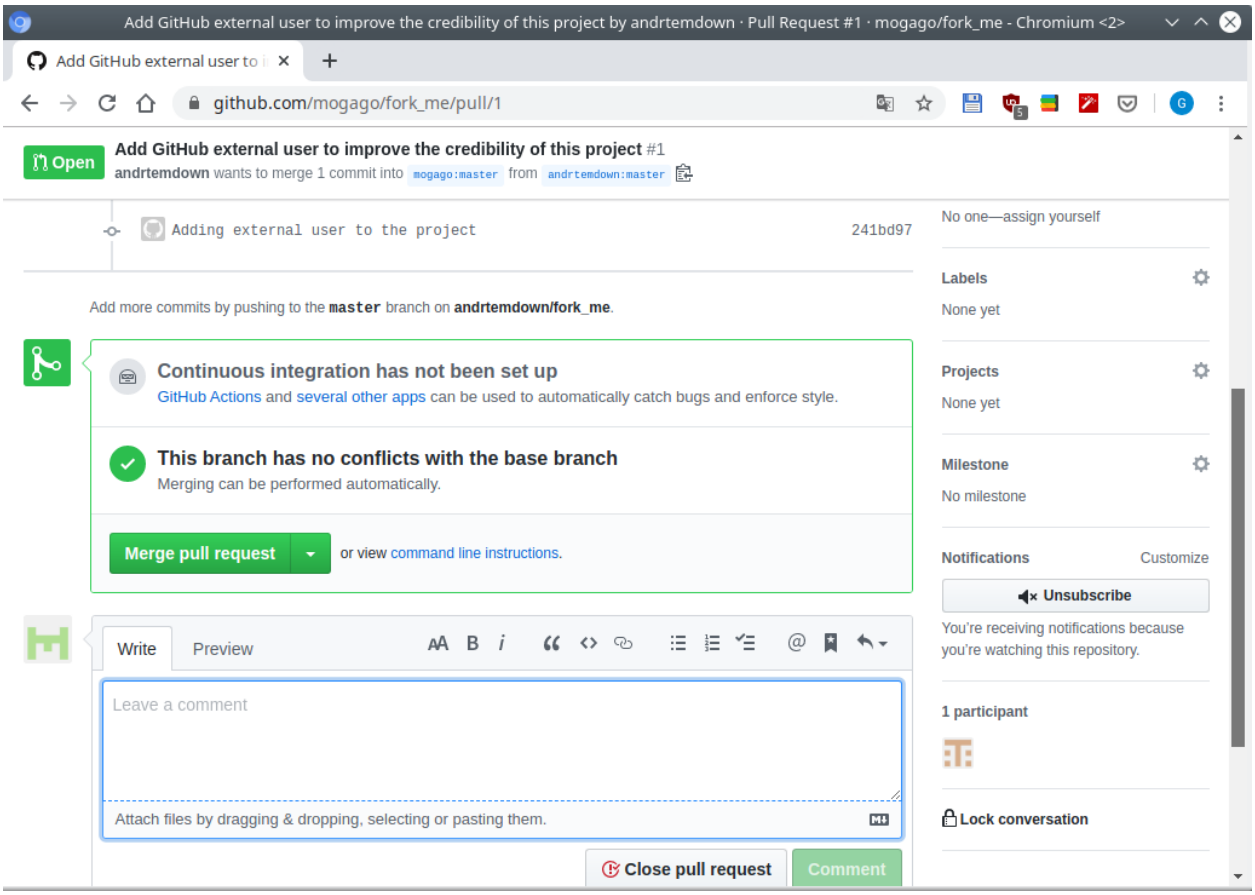


Fig. 32: GitHub - Aceptar petición `pull request`

Una vez que aceptemos los cambios de la petición `pull request` podemos hacer merge de este. Para hacer esto será igual que hacer merge de dos branches en la línea de comandos. Tomará la divergencia del historial si existe y nunca creará un fast forward merge. Clic en la opción *Merge pull request*:

Nos saldrá en la misma ventana un recuadro de texto para confierma el merge, editarlo según deseemos y clic en *Confirm merge*:

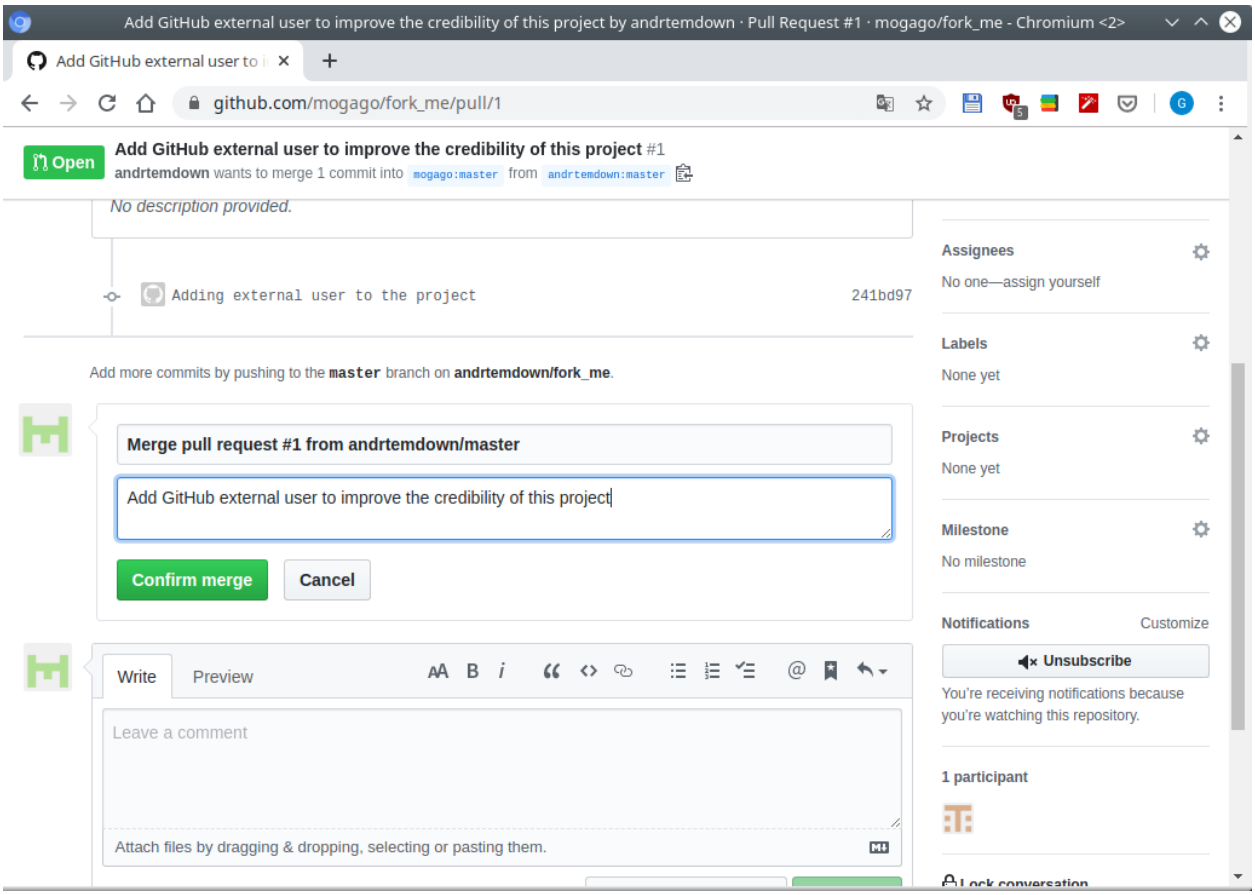


Fig. 33: GitHub - Aceptar petición `pull request`

En los archivos del repositorio externo veremos el archivo con el cual contribuyó el usuario externo:

3.7.5 Use cases for fork based collaboration

Hay 2 casos de uso principales de fork based `pull request`:

1. Entorno de confianza: Si estamos trabajando con un equipo de personas que no conocemos. Aceptamos contribuciones que no conocemos personalmente.
2. Contribuidores ocasiones. En una compañía podría servir para no dar permisos totales a personas que no deseemos darles permisos dentro de nuestro repo.

Donde no funciona muy bien es con un grupo pequeño de personas que trabajan juntos regularmente. No es deseable que cada uno tenga su propio fork.

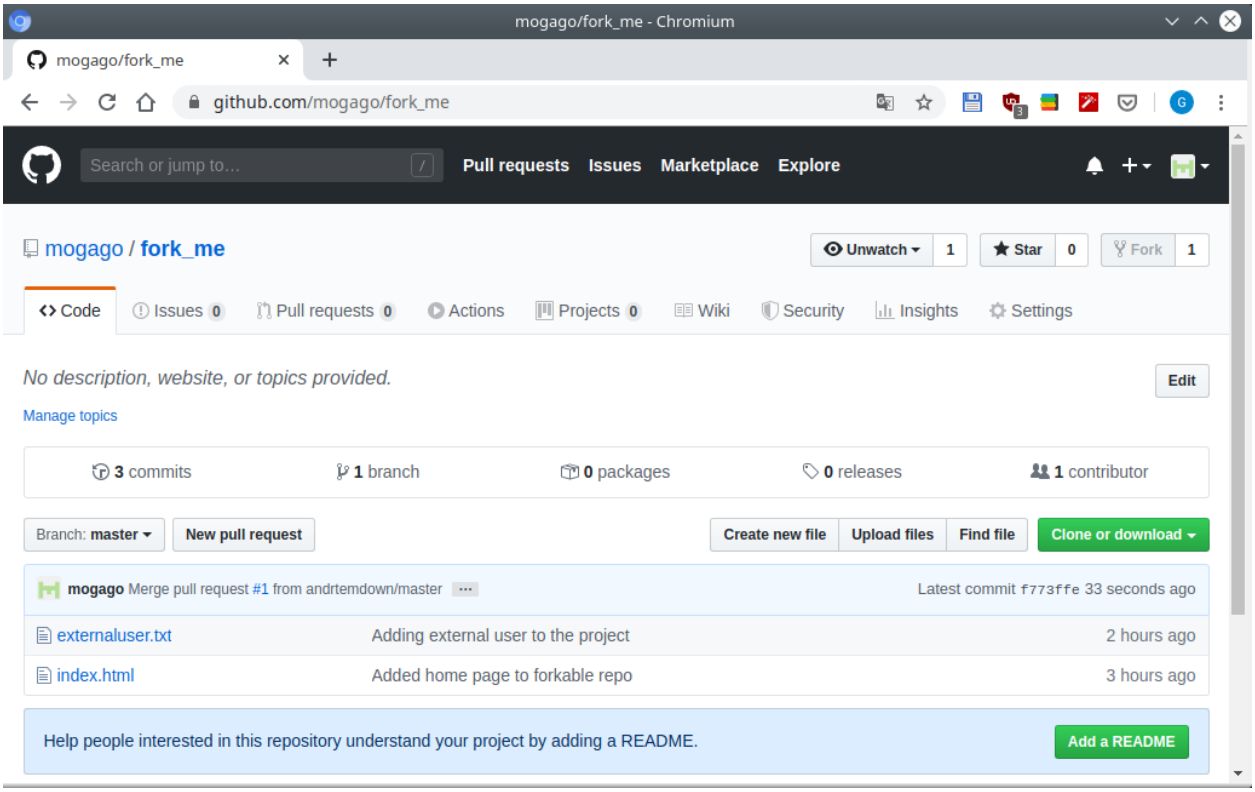


Fig. 34: GitHub - Petición `pull request` aceptada

3.7.6 Single repo collaboration directly on master

1. Nuevo repositorio en Git y GitHub.
2. Desde otro usuario distinto al creador del repositorio clonar el repositorio y agregar un archivo, commit y push. No podremos hacer push porque no somos colaboradores.
3. Agregar como colaborador desde GitHub al usuario que hizo fork del proyecto.
4. Si hacemos un push desde el creador del repo, antes que los colaboradores haga un repo no obtendremos error.
5. Si alguien hace push obtendrá error. Deberá hacer git pull primero. Pero para que no sea desordenado, usar –rebase.
6. Si nadie ha hecho un push, podremos hacer nuestro git push

Creemos otro proyecto. Desde el terminal:

```
$ git init single_repo
Initialized empty Git repository in /home/gabriel/Documents/gittests/single_repo/.git/

$ cd single_repo/
$ touch index.html
$ git add .

$ git commit -m "Added new home page"
[master (root-commit) aa2d72f] Added new home page
1 file changed, 0 insertions(+), 0 deletions(-)
create mode 100644 index.html
```

Desde GitHub:

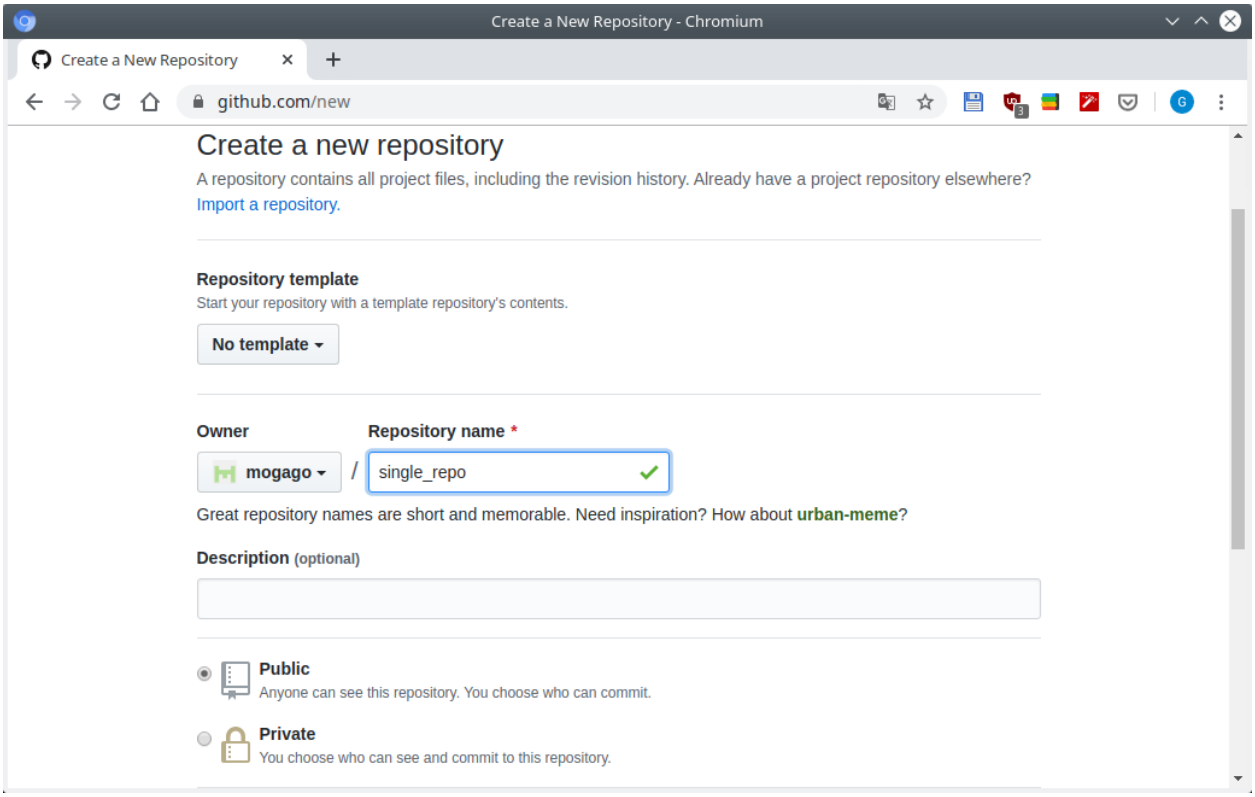


Fig. 35: GitHub - Creación de nuevo repo

En el terminal hacemos `git push` del repositorio local:

```
$ git remote add origin https://github.com/mogago/single_repo.git
$ git push -u origin master
Username for 'https://github.com': mogago
Password for 'https://mogago@github.com':
Counting objects: 3, done.
Writing objects: 100% (3/3), 219 bytes | 219.00 KiB/s, done.
Total 3 (delta 0), reused 0 (delta 0)
To https://github.com/mogago/single_repo.git
* [new branch]      master -> master
Branch 'master' set up to track remote branch 'master' from 'origin'.
```

Hasta acá hemos hecho una configuración y creación básica de un repositorio Git/GitHub.

Desde otra cuenta diferente al creador del repositorio ireamos al URL del repo (https://github.com/mogago/single_repo), lo clonaremos y haremos un `commit`. Sin embargo, no podremos hacer `push` a este repositorio porque no somos colaboradores.

Primero añadiremos como colaborador desde el usuario dueño del repositorio al otro usuario. En el repositorio de GitHub seleccionar la opción *Settings* de la barra superior, luego seleccionar la opción *Collaborators* de la barra lateral izquierda.

Ingresar el nombre de usuario o correo del colaborador que deseemos agregar. Clic en *Add collaborator*

Se habrá enviado una invitación de colaboración al usuario externo:

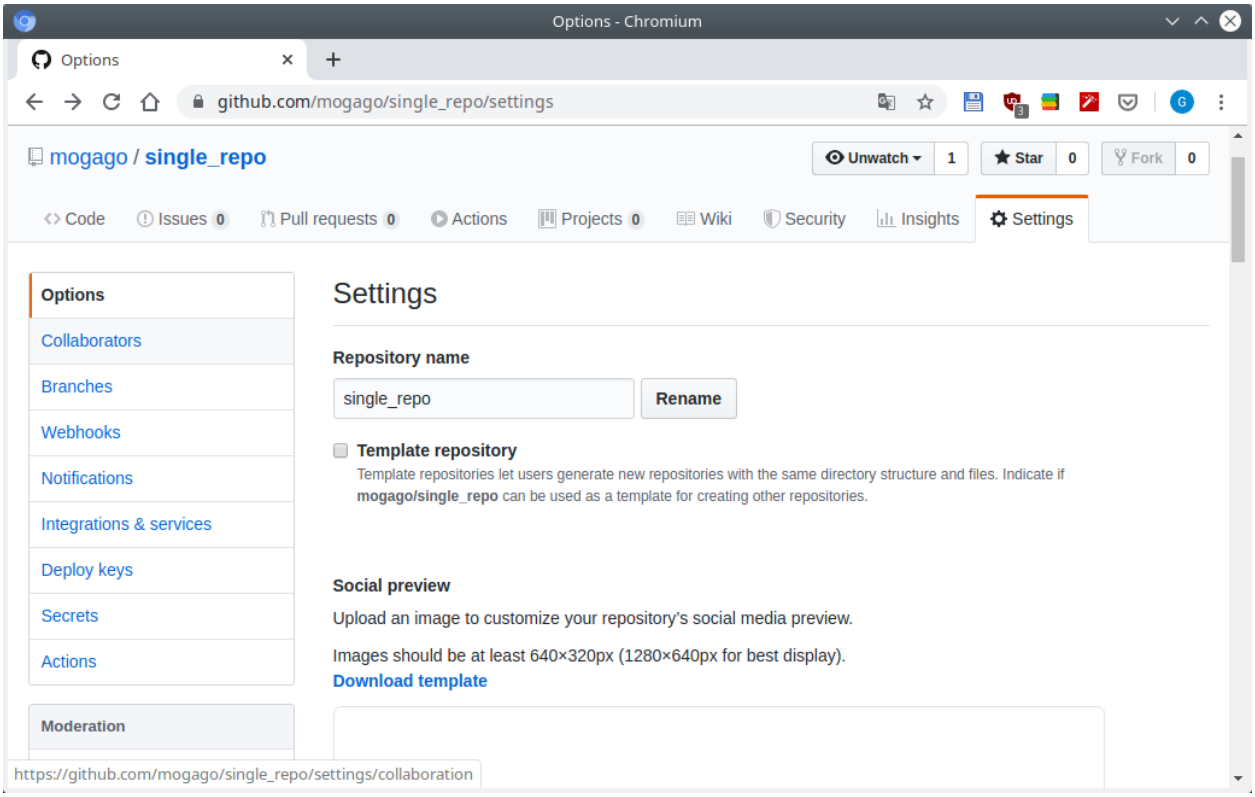


Fig. 36: GitHub - Añadir Colaborador, opción *Settings*, *Collaborators*

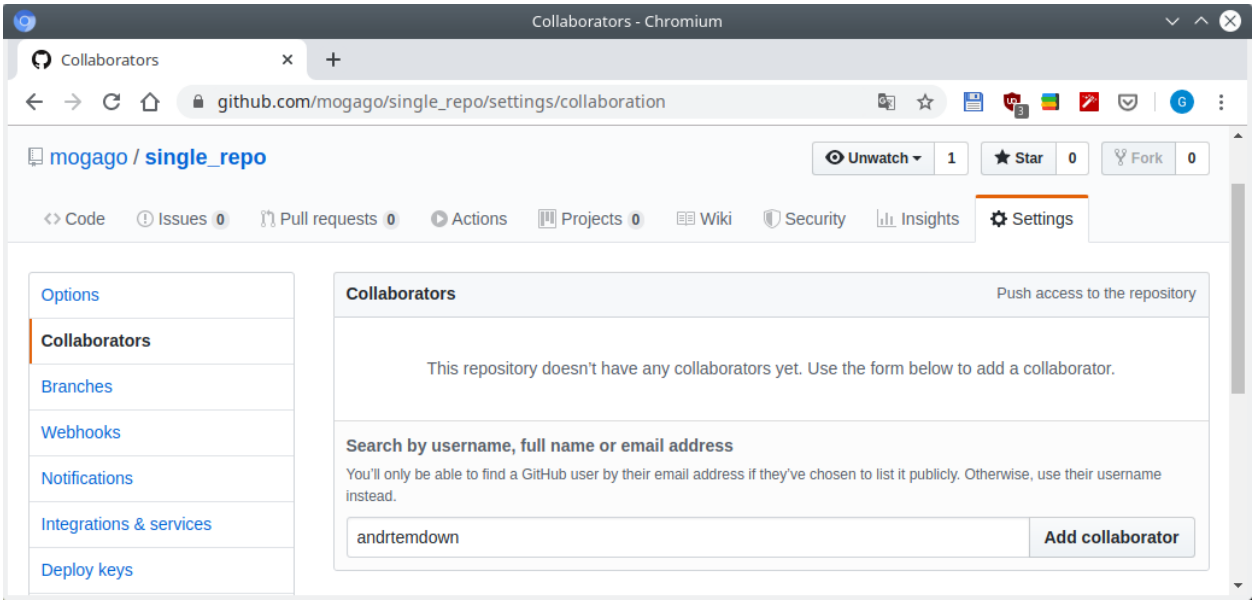


Fig. 37: GitHub - Añadir colaborador, nombre de usuario

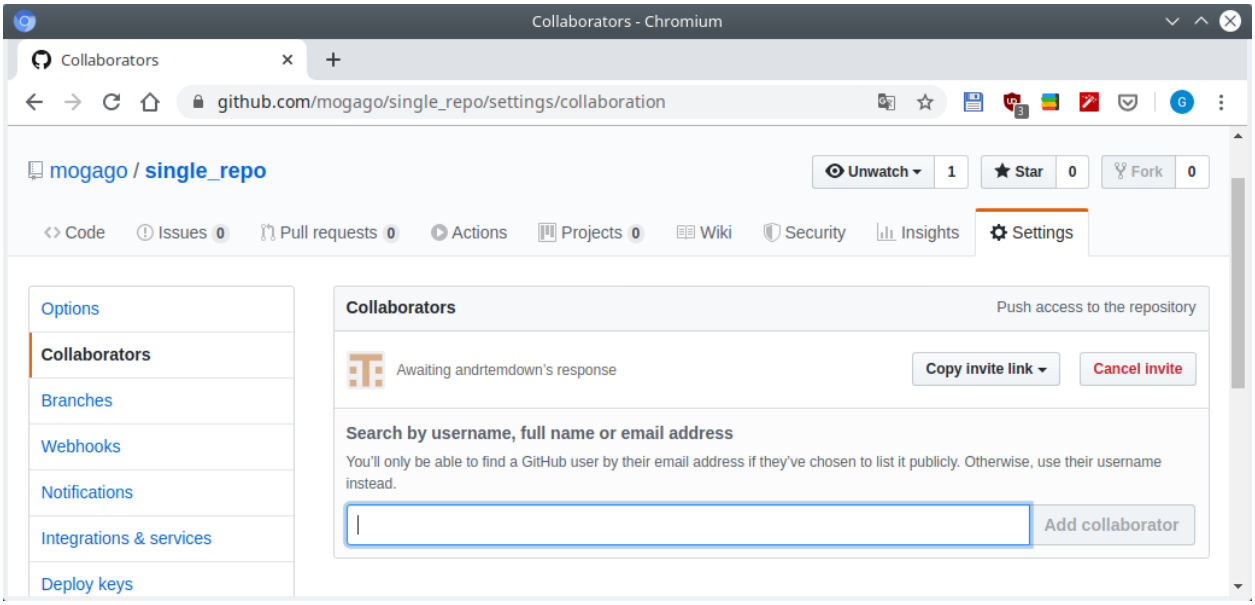


Fig. 38: GitHub - Añadir colaborador, invitación enviada

El usuario externo recibirá un correo con la invitación de colaboración mediante la URL: https://github.com/mogago/single_repo/invitations

Clic en la opción *Accept invitation*. Ahora en el repositorio veremos el siguiente mensaje: `You now have push access to the mogago/single_repo repository.`. Es decir, ahora podremos hacer `push` al repositorio.

Los dos colaboradores están trabajando en el mismo branch `master` y harán `push` a este branch.

Desde el usuario externo añadido como colaborador clonaremos el repositorio, haremos un `commit` y `push`:

```
$ git clone https://github.com/mogago/single_repo.git
$ touch externaluser.txt
$ git add .
$ git commit -m "Added my name"
[master f395a21] Added my name
 1 file changed, 0 insertions(+), 0 deletions(-)
 create mode 100644 externaluser.txt

$ git push
Username for 'https://github.com': andrtemdown
Password for 'https://andrtemdown@github.com':
Counting objects: 2, done.
Delta compression using up to 4 threads.
Compressing objects: 100% (2/2), done.
Writing objects: 100% (2/2), 257 bytes | 257.00 KiB/s, done.
Total 2 (delta 0), reused 0 (delta 0)
To https://github.com/mogago/single_repo.git
    aa2d72f..f395a21  master -> master
```

Desde el usuario creador del repo tendremos problemas al hacer un `push`:

```
$ touch mogago.txt
$ git add .
```

(continues on next page)

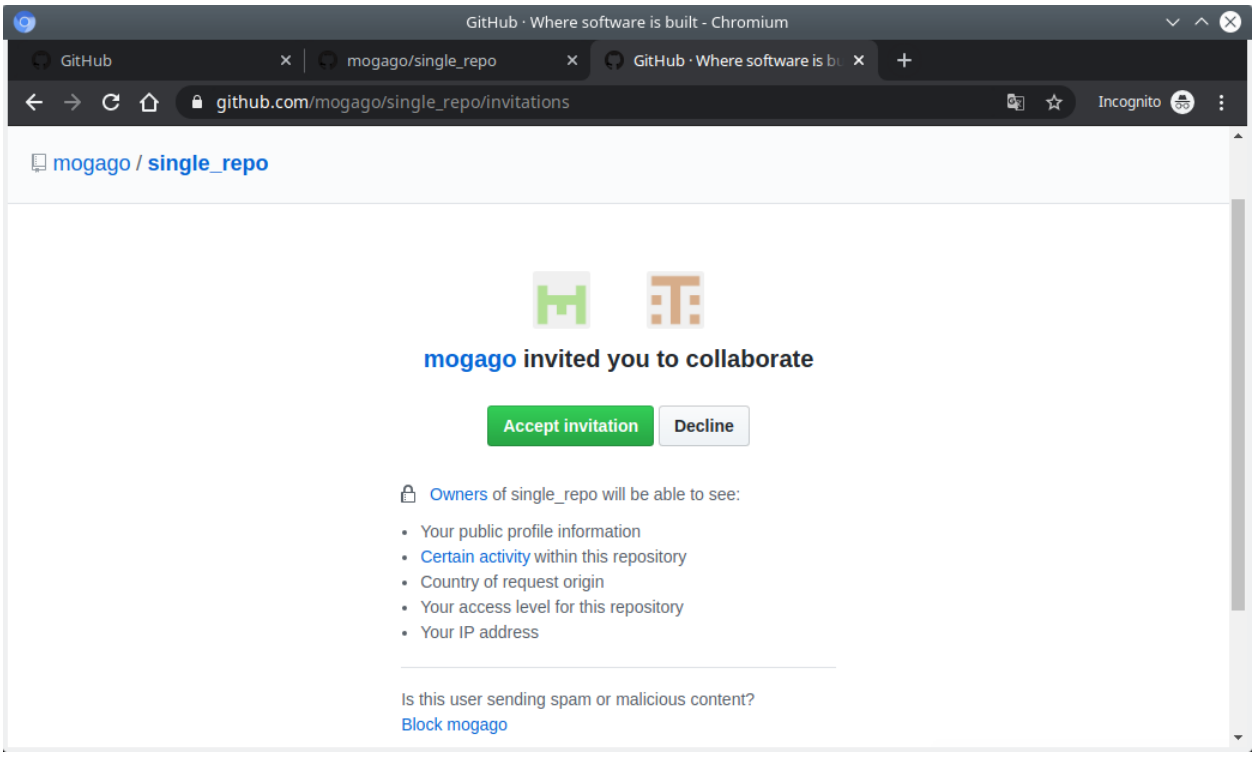


Fig. 39: GitHub - Aceptar invitación, opción *Accept invitation*

(continued from previous page)

```
$ git commit -m "Added my name mogago"
[master b0c8a81] Added my name mogago
 1 file changed, 0 insertions(+), 0 deletions(-)
 create mode 100644 mogago.txt

$ git push
Username for 'https://github.com': mogago
Password for 'https://mogago@github.com':
To https://github.com/mogago/single_repo.git
    ! [rejected]        master -> master (fetch first)
error: failed to push some refs to 'https://github.com/mogago/single_repo.git'
hint: Updates were rejected because the remote contains work that you do
hint: not have locally. This is usually caused by another repository pushing
hint: to the same ref. You may want to first integrate the remote changes
hint: (e.g., 'git pull ...') before pushing again.
hint: See the 'Note about fast-forwards' in 'git push --help' for details.
```

El `push` no funciona porque hay cambios en GitHub que nosotros no tenemos, por lo tanto primero debemos hacer `git pull`. Sin embargo, con este comando será un poco desordenado, así que usaremos `git pull --rebase` desde el usuario creador del repo. Y luego `git push` para agregar nuestros cambios:

```
$ git pull --rebase
remote: Enumerating objects: 3, done.
remote: Counting objects: 100% (3/3), done.
remote: Compressing objects: 100% (2/2), done.
remote: Total 2 (delta 0), reused 2 (delta 0), pack-reused 0
Unpacking objects: 100% (2/2), done.
```

(continues on next page)

(continued from previous page)

```
From https://github.com/mogago/single_repo
aa2d72f..f395a21  master     -> origin/master
First, rewinding head to replay your work on top of it...
Applying: Added my name mogago

$ ls
externaluser.txt  index.html  mogago.txt

$ git push
Username for 'https://github.com': mogago
Password for 'https://mogago@github.com':
Counting objects: 2, done.
Delta compression using up to 4 threads.
Compressing objects: 100% (2/2), done.
Writing objects: 100% (2/2), 276 bytes | 276.00 KiB/s, done.
Total 2 (delta 0), reused 0 (delta 0)
To https://github.com/mogago/single_repo.git
    f395a21..31c9eb4  master -> master
```

Podemos ver que esta no es una buena forma de trabajo porque si fuesen más usuarios, todos estarían haciendo `push` sobre el mismo branch al mismo tiempo sobre el trabajo de otros. El problema es que no podremos hacer `push` sin antes haber hecho `pull` y obstaculiza el trabajo del grupo. Todos trabajando en `master` branch no funciona.

3.7.7 Single repo collaboration using feature branches

Para solucionar el problema anterior, usaremos **feature branches**. Con feature branches, en lugar de que todos colaboremos directamente en el branch `master`, todos construyamos nuestros feature branches por separado.

Usualmente en un proyecto real, cada feature branch será nombrado bajo el nombre del feature que deseemos construir. En este caso, podemos crear un brach con nuestro nombre de usuario.

Desde el usuario añadido como colaborador:

```
$ git checkout -b externaluser
Switched to a new branch 'externaluser'
$ vi externaluser.txt
    Here is some text I added

$ git commit -am "Added some great content to my file"
[externaluser fb2938b] Added some great content to my file
1 file changed, 1 insertion(+)
$ git lg
* fb2938b (HEAD -> externaluser) Added some great content to my file
* f395a21 (origin/master, origin/HEAD, master) Added my name
* aa2d72f Added new home page
```

En el log veremos que nuestro branch está un `commit` adelante del branch `master` y `origin/master`. Lo siguiente que debemos hacer es un `push` de nuestro branch hacia GitHub. Para esto, configuraremos otro **default upstream** con nuestro **feature branch**:

```
$ git push -u origin externaluser
Username for 'https://github.com': andrtemdown
Password for 'https://andrtemdown@github.com':
Counting objects: 8, done.
Delta compression using up to 4 threads.
```

(continues on next page)

(continued from previous page)

```
Compressing objects: 100% (5/5), done.
Writing objects: 100% (8/8), 743 bytes | 743.00 KiB/s, done.
Total 8 (delta 0), reused 0 (delta 0)
remote:
remote: Create a pull request for 'externaluser' on GitHub by visiting:
remote:      https://github.com/mogago/single_repo/pull/new/externaluser
remote:
To https://github.com/mogago/single_repo.git
* [new branch]      externaluser -> externaluser
Branch 'externaluser' set up to track remote branch 'externaluser' from 'origin'.
```

En GitHub veremos que se ha creado un nuevo branch `externaluser`:

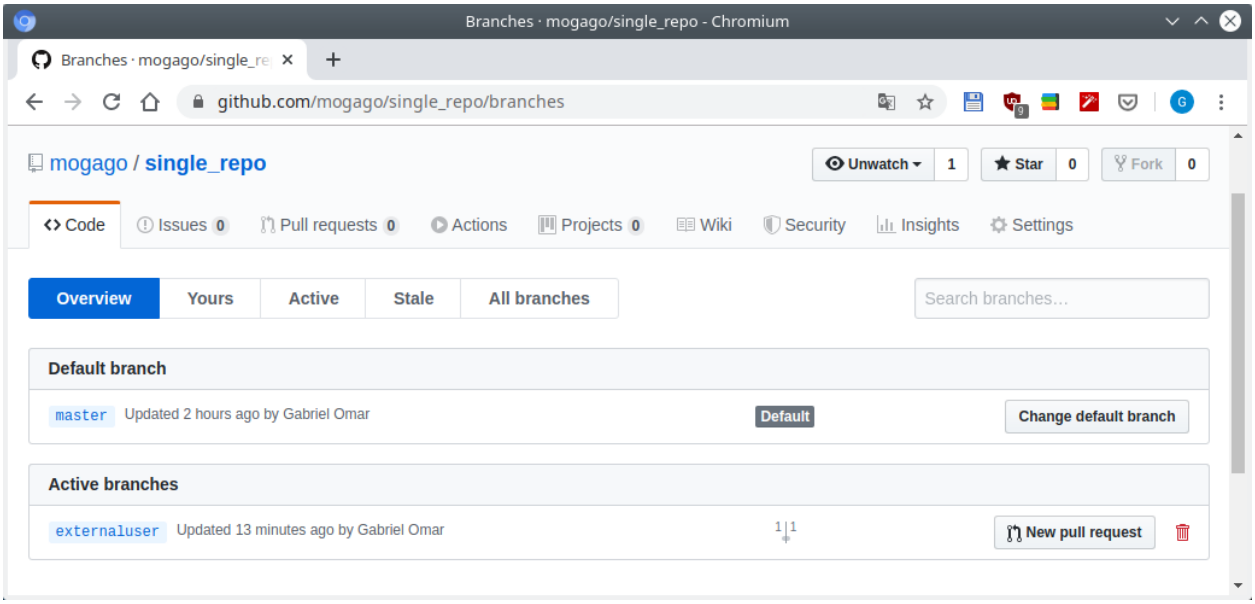


Fig. 40: GitHub - Nuevo branch

Con el creador del repositorio recuperaremos los cambios hechos por el usuario externo:

```
$ git lg
* 31c9eb4 (HEAD -> master, origin/master) Added my name mogago
* f395a21 Added my name
* aa2d72f Added new home page

$ cat externaluser.txt
$ git checkout master
Already on 'master'
Your branch is up to date with 'origin/master'.

$ git pull
remote: Enumerating objects: 5, done.
remote: Counting objects: 100% (5/5), done.
remote: Compressing objects: 100% (2/2), done.
remote: Total 3 (delta 0), reused 3 (delta 0), pack-reused 0
Unpacking objects: 100% (3/3), done.
From https://github.com/mogago/single_repo
* [new branch]      externaluser -> origin/externaluser
```

(continues on next page)

(continued from previous page)

```
Already up to date.
```

Notamos que se tiene un nuevo branch `externaluser` al haber hecho `git pull`. Ahora veremos los cambios en el repositorio:

```
$ git lg
* fb2938b (origin/externaluser) Added some great content to my file
| * 31c9eb4 (HEAD -> master, origin/master) Added my name mogago
|/
* f395a21 Added my name
* aa2d72f Added new home page
```

3.7.8 Contributing to another feature branch

Desde el usuario creador del repositorio podemos ver todos los branches con la opción `branch -a`:

```
$ git branch -a
* master
remotes/origin/externaluser
remotes/origin/master
```

Con el mismo usuario creador del repositorio haremos checkout al branch creado por el usuario externo colaborador:

```
$ git checkout externaluser
Branch 'externaluser' set up to track remote branch 'externaluser' from 'origin'.
Switched to a new branch 'externaluser'

$ git branch
* externaluser
  master

$ cat externaluser.txt
Here is some text I added
```

Veremos que tenemos un branch más agregado y podemos ver el contenido del archivo `externaluser.txt`, hacerle cambios y contribuir.

Note: ¿Qué hemos hecho?: Tenemos un solo repositorio, cada colabordor está trabajando en un feature branch y cada uno puede bajar (`pull`) cualquier feature branch que se haya subido a GitHub (`push`), hacer checkout a esos branches y contribuir si deseamos.

3.7.9 Creating a pull request within a single repo

En algún punto querremos crear un `pull request`. Previamente vimos a `pull request` como una forma de pedir a una persona en otro proyecto agregue nuestro trabajo de nuestro fork del proyecto.

`pull request` también se usa por varios equipos como la forma por defecto de colaborar en un nuevo feature. La forma en que funciona es que creamos un nuevo feature branch y le hacemos `push` hacia GitHub (como hemos visto en el tema anterior). Cuando hemos acabado con esto, creamos un `pull request`. Ahora cualquiera en el equipo, puede comentarlo, interactuar y hacer cambios si lo necesita.

Creemos un `pull request`. Desde el usuario añadido como colaborador en GitHub ir a la sección *Pull requests*, opción *New pull request*, comparar los branches `master` y `externaluser`:

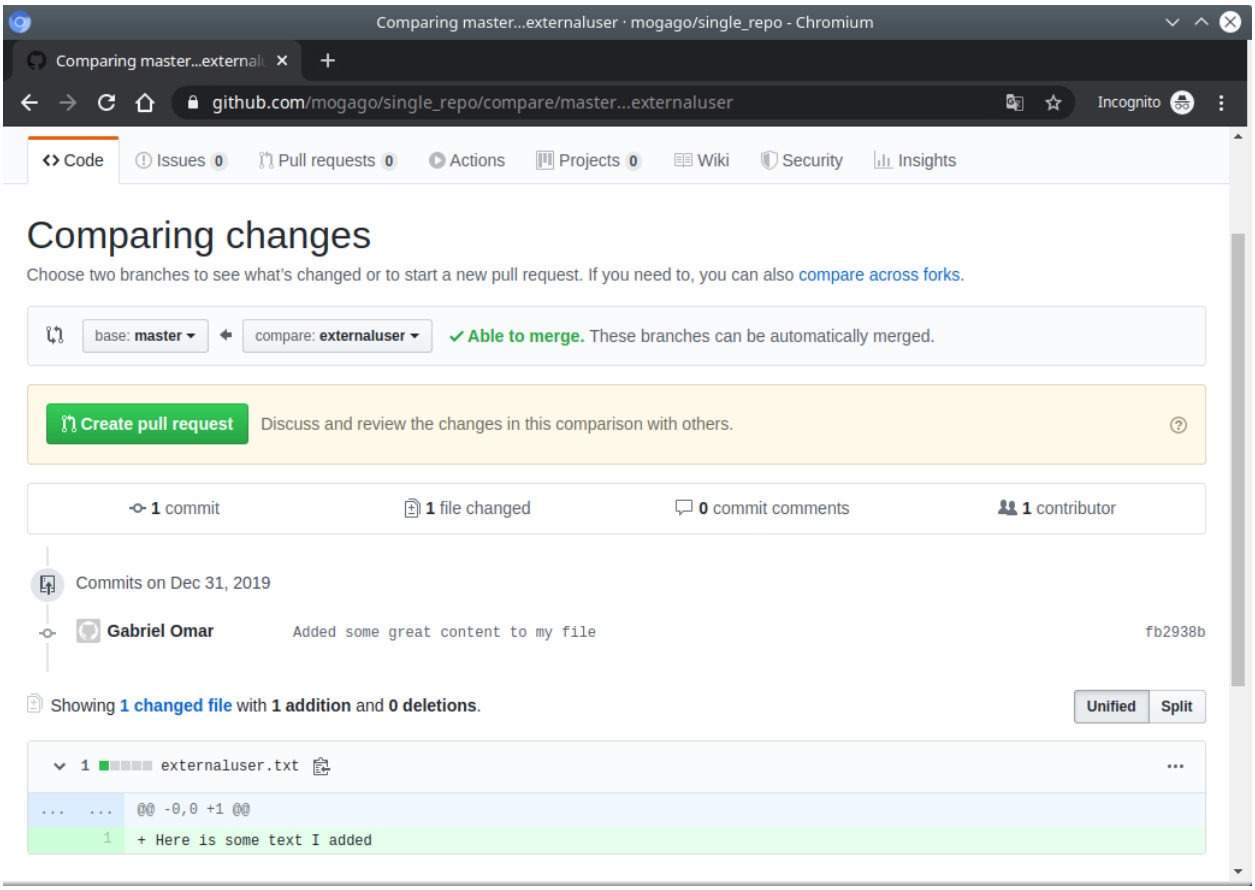


Fig. 41: GitHub - Comparing changes

Seleccionar la opción *pull request*. Dar una descripción y clic en la opción *Create pull request*. El usuario creador del repositorio podrá interactuar con este `pull request`.

3.7.10 Collaborating on a pull request

Todos los colaboradores pueden hacer merge del `pull request`. La idea es que el mismo usuario que ha realizado el `pull request` no haga el merge sin antes haber recibido feedback de su trabajo. Ya que es un repositorio público, cualquiera puede poner comentarios en la siguiente ventana:

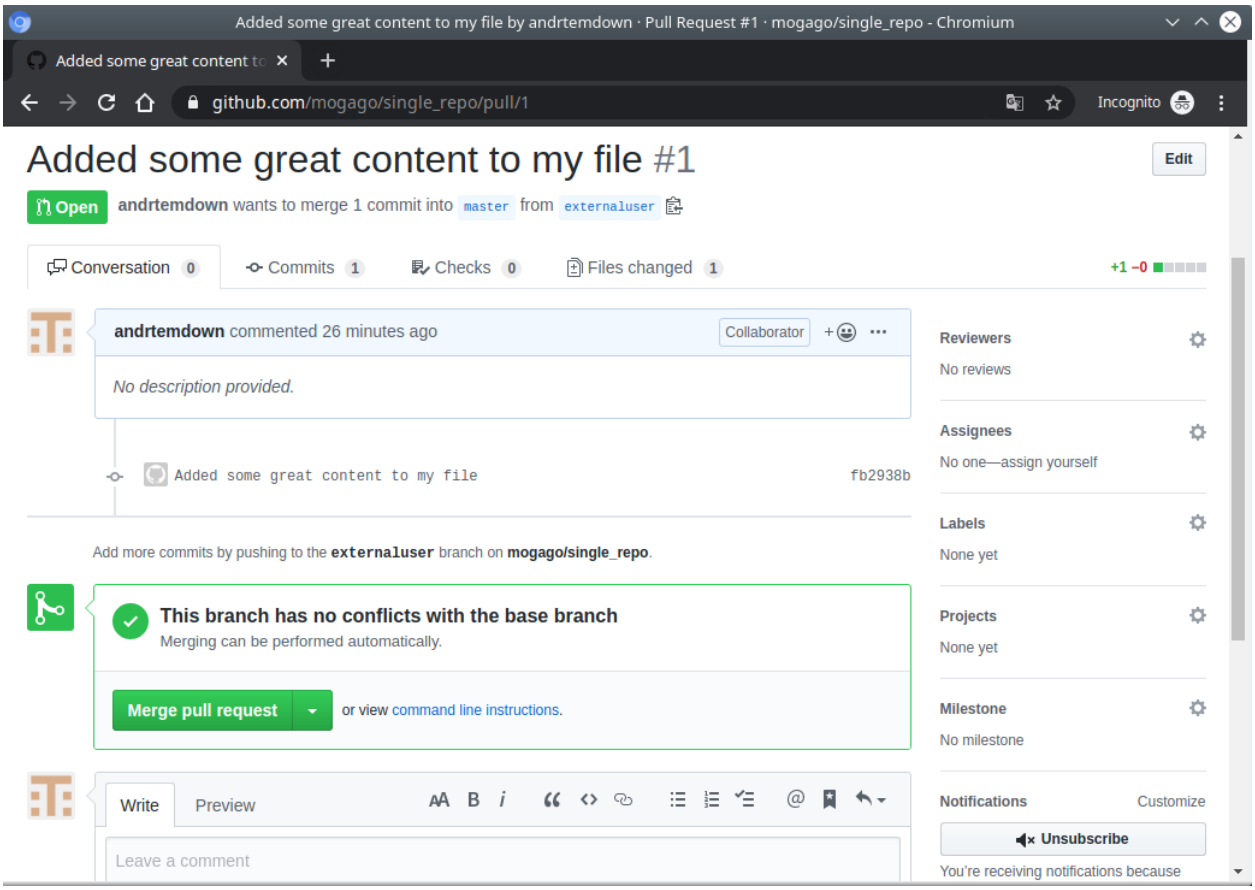


Fig. 42: GitHub - *Merge pull request*

Un `pull request` no apunta a un `commit`, apunta a un branch. Si agregamos más cosas a un branch, añadirá más `commits` al pull request. Luego de que sintamos que el código está listo, algunos de los colaboradores hará merge del `pull request`.

3.7.11 Merging in a pull request

Generalmente queremos asegurarnos que otros colaboradores haya aceptado nuestro trabajo y hacerle merge. ¿Quién hará el merge?: mejor será que quien ha creado el `pull request` haga el merge, habiendo recibido aceptación en su trabajo de otros colaboradores.

3.8 Lesson 8

Lesson 8: Reviewing a Project on GitHub

Table of Contents

3.8.1 Getting an overview of a project on GitHub using the README

Veamos la forma recomendada de ver un repositorio en GitHub. Por ejemplo el repositorio de Ruby on Rails.

Si fuese un proyecto que nunca hemos visto antes, primero deberíamos leer el archivo `README.md`. Aquí encontraremos información general del proyecto.

En la descripción podemos usar medallas (badges) en el archivo README.md. Por ejemplo en código markdown:

```
[![Build Status](https://badge.buildkite.com/
↪ab1152b6a1f6a61d3ea4ec5b3eece8d4c2b830998459c75352.svg?branch=master)](https://
↪buildkite.com/rails/rails)
```

3.8.2 Getting more information about a project

Luego de leer el archivo README deberíamos mirar los `commits` recientes. No veremos el último trabajo que las personas han hecho sino el último trabajo que se le ha hecho merge hacia el branch `master`, que presuntamente ya debe estar listo para producción.

Ver los `commits`:

Si queremos obtener actividades aún más recientes, no solo los features que han sigo añadidos con merge, debemos ver los `pull request`:

Estas son las unidades de trabajo que se están trabajando, siendo arreglos de bugs o nuevos features.

3.8.3 Introducing issues

Otro lugar que vale la pena ver es "Issues". Es usado para dos casos distintos:

1. Track de bugs
2. Administrar feature requests

3.8.4 Viewing project state through pulse and graphs

Otra área recomendable de mirar para obtener un sentido del proyecto es "Pulse" y "Graphs":

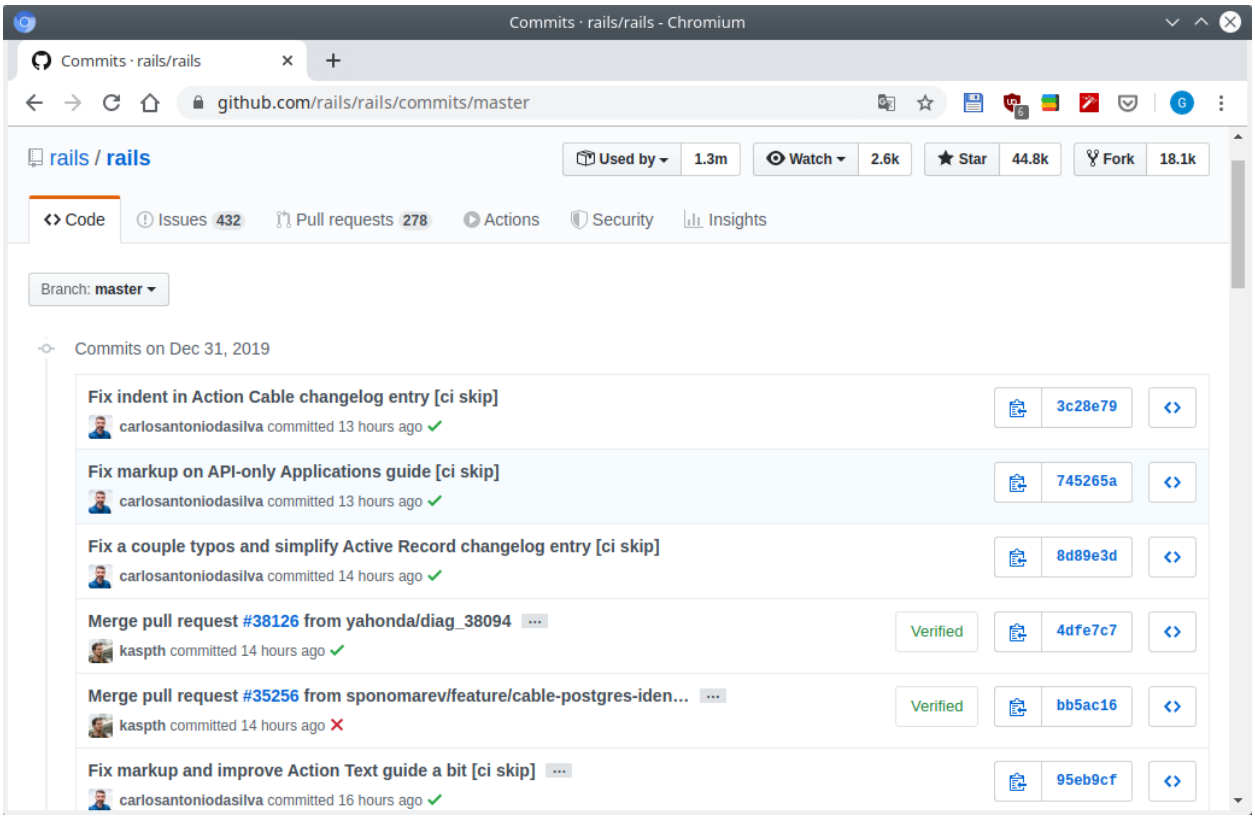


Fig. 43: GitHub - Commits

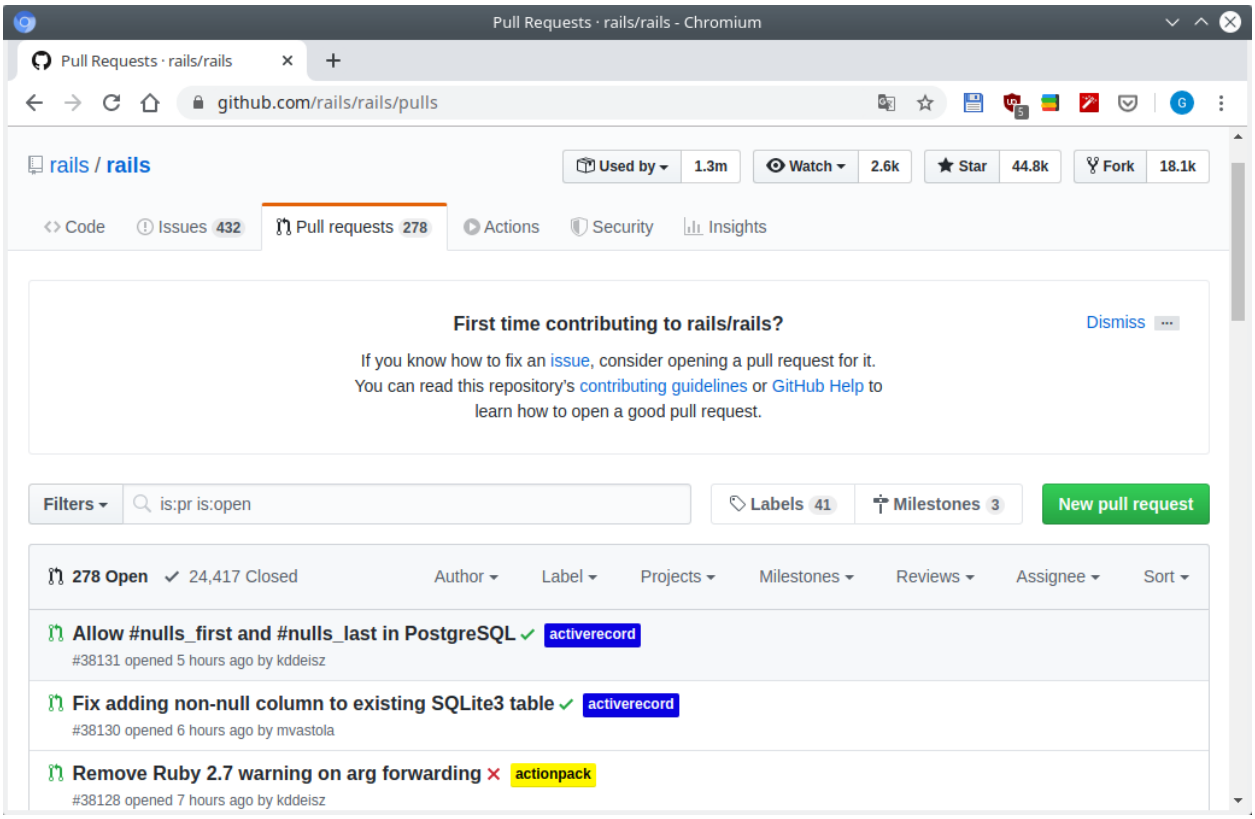


Fig. 44: GitHub - Pull requests

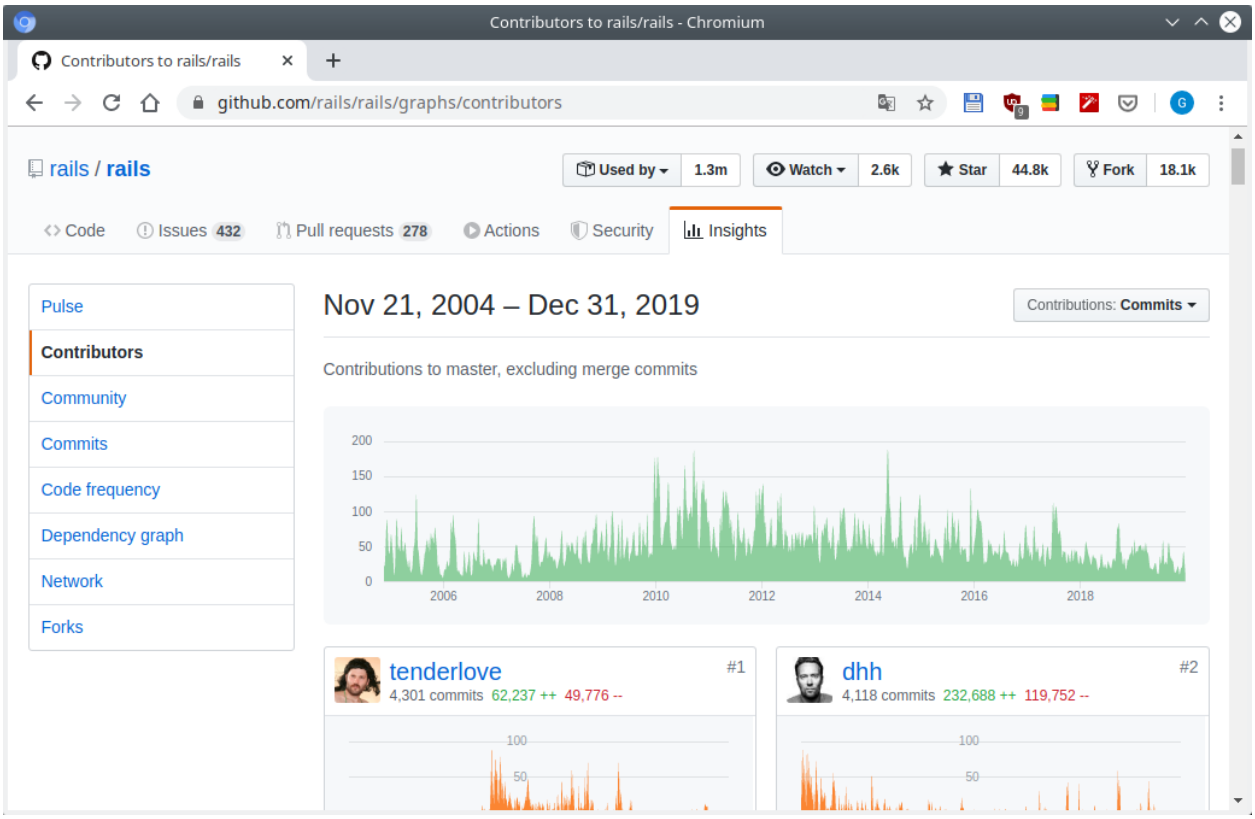


Fig. 45: GitHub - Gráfico de contribuidores

Podemos ver quienes han contribuido al proyecto y quienes lo han hecho más:

Las gráficas de `commits` mostrará la cantidad de `commits` por semana a lo largo del tiempo de vida del proyecto en forma de barras:

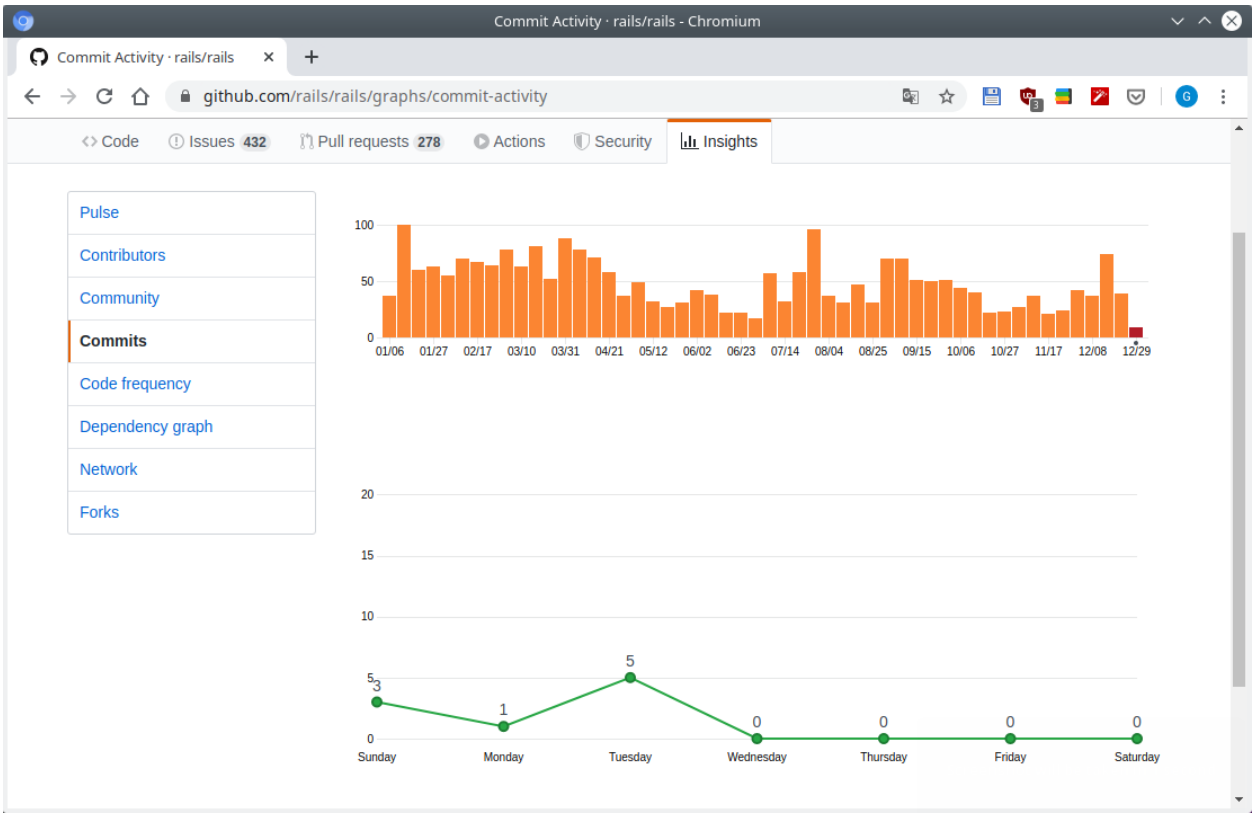


Fig. 46: GitHub - Gráficas de commits

La gráfica de code frequency nos mostrará el número de líneas añadidas y eliminadas del código:

Si Graph nos muestra qué pasa a lo largo de la historia de nuestro proyecto, Pulse nos dice qué ha pasado recientemente en el proyecto (el último día, semana, mes):

Estos números no nos dicen cúantos `pull request` o issues están abiertos, nos dice cuántos han sido cambiados:

3.9 Lesson 9

Lesson 9: Configuring a Project on GitHub

Table of Contents

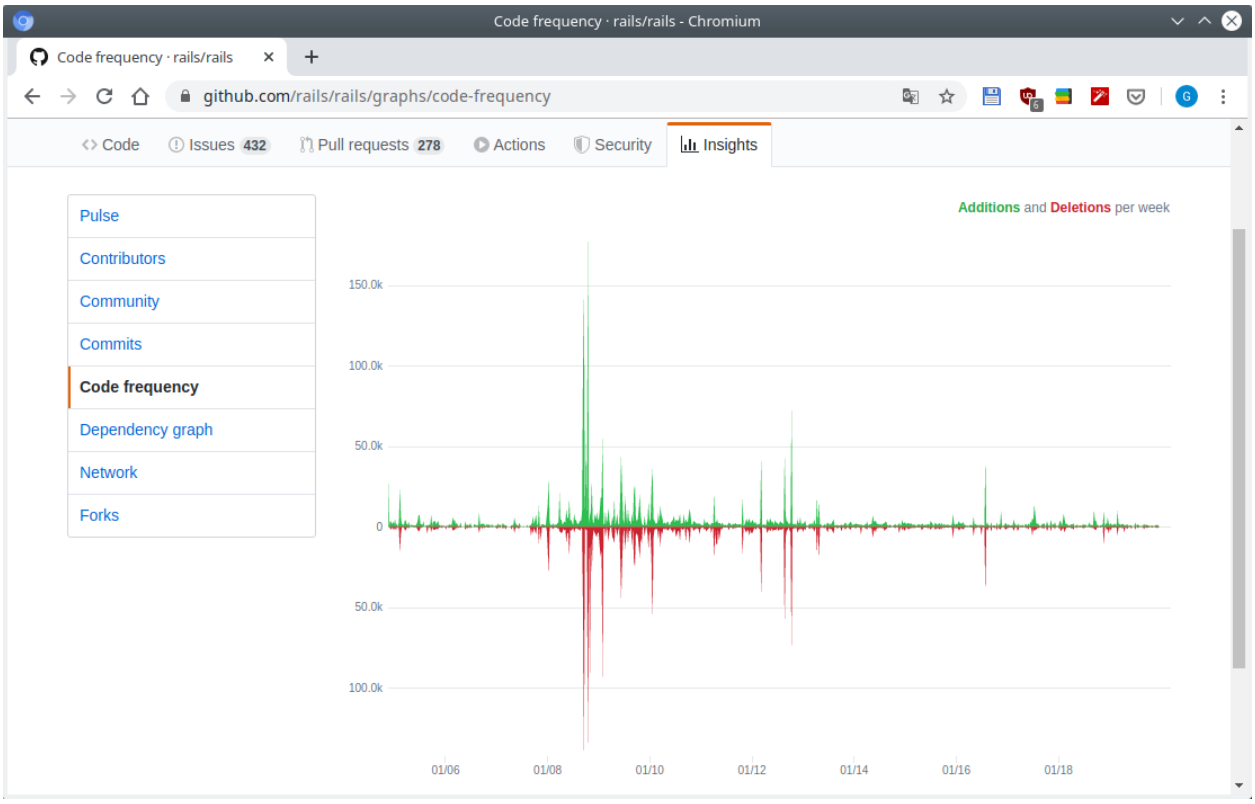


Fig. 47: GitHub - Gráficas de Code frequency

– *Configuring deploy keys*

3.9.1 Basic configuration options and renaming on a repo

Veamos cómo se configura GitHub. En la opción *Settings* del proyecto podemos ver disitintas configuraciones. Por ejemplo podemos renombrar un repositorio:

Hemos renombrado el repositorio de GitHub de `app2` a `newapp2` pero aun así seríamos capaces de hacer `push` desde el terminal:

```
$ cat .git/config
[core]
        repositoryformatversion = 0
        filemode = true
        bare = false
        logallrefupdates = true
[remote "origin"]
        url = https://github.com/mogago/app2.git
        fetch = +refs/heads/*:refs/remotes/origin/*
[branch "master"]
        remote = origin
        merge = refs/heads/master
```

Pensaríamos que al hacer un push a la dirección https://github.com/mogago/app2.git daría error pues ahora el repositorio apunta a https://github.com/mogago/newapp2 , sin embargo veremos que no encontraremos este problema:

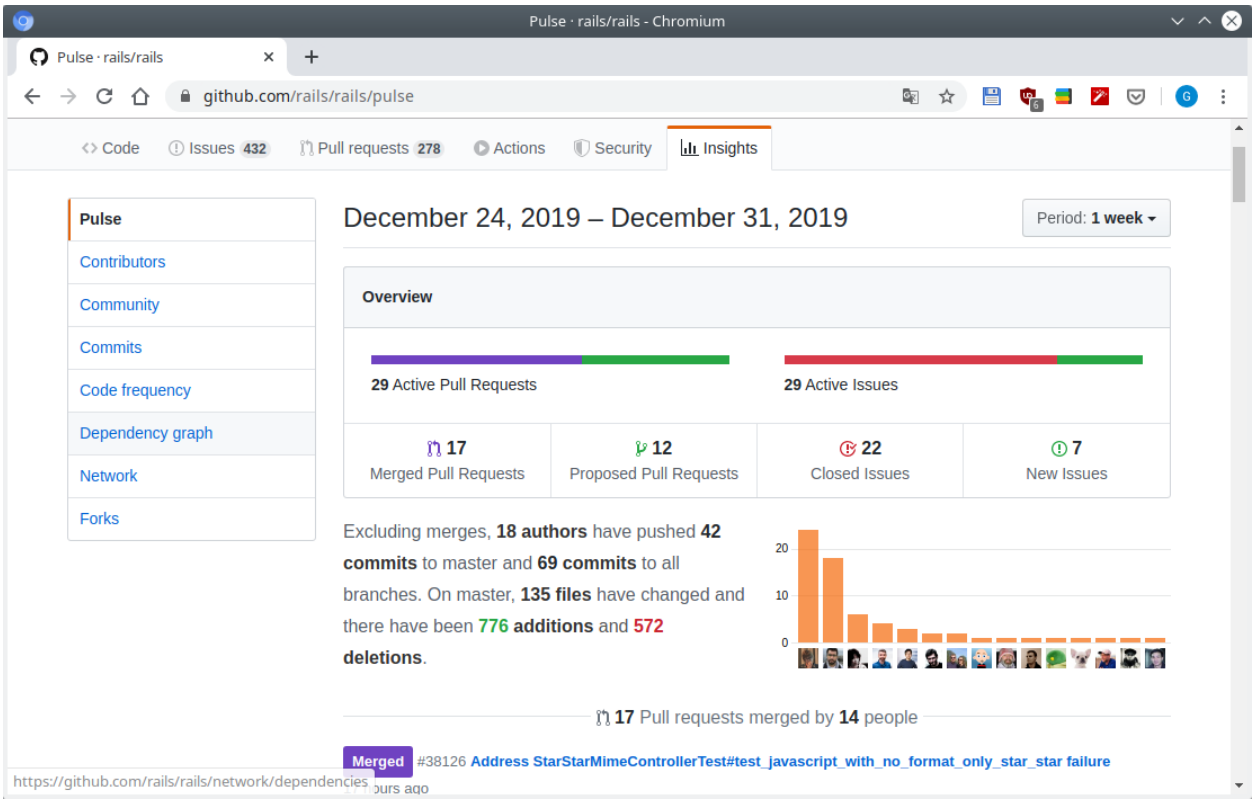


Fig. 48: GitHub - Pulse

```
$ touch test.html
$ git add .
$ git commit -m "Whatever"
[master 1d8cd6b] Whatever
1 file changed, 0 insertions(+), 0 deletions(-)
create mode 100644 test.html

$ git push
Username for 'https://github.com': mogago
Password for 'https://mogago@github.com':
Counting objects: 3, done.
Delta compression using up to 4 threads.
Compressing objects: 100% (2/2), done.
Writing objects: 100% (3/3), 267 bytes | 267.00 KiB/s, done.
Total 3 (delta 1), reused 0 (delta 0)
remote: Resolving deltas: 100% (1/1), completed with 1 local object.
To https://github.com/mogago/app2.git
    42c4494..1d8cd6b  master -> master
```

En GitHub vemos que tenemos el archivo `test.html` creado a pesar del cambio de nombre:

En *Settings* también podemos configurar Wikis, prender/apagar issues.

3.9.2 GitHub Pages

GitHub Pages permite crear una página web de un proyecto que se ve mejor que una simple wiki.

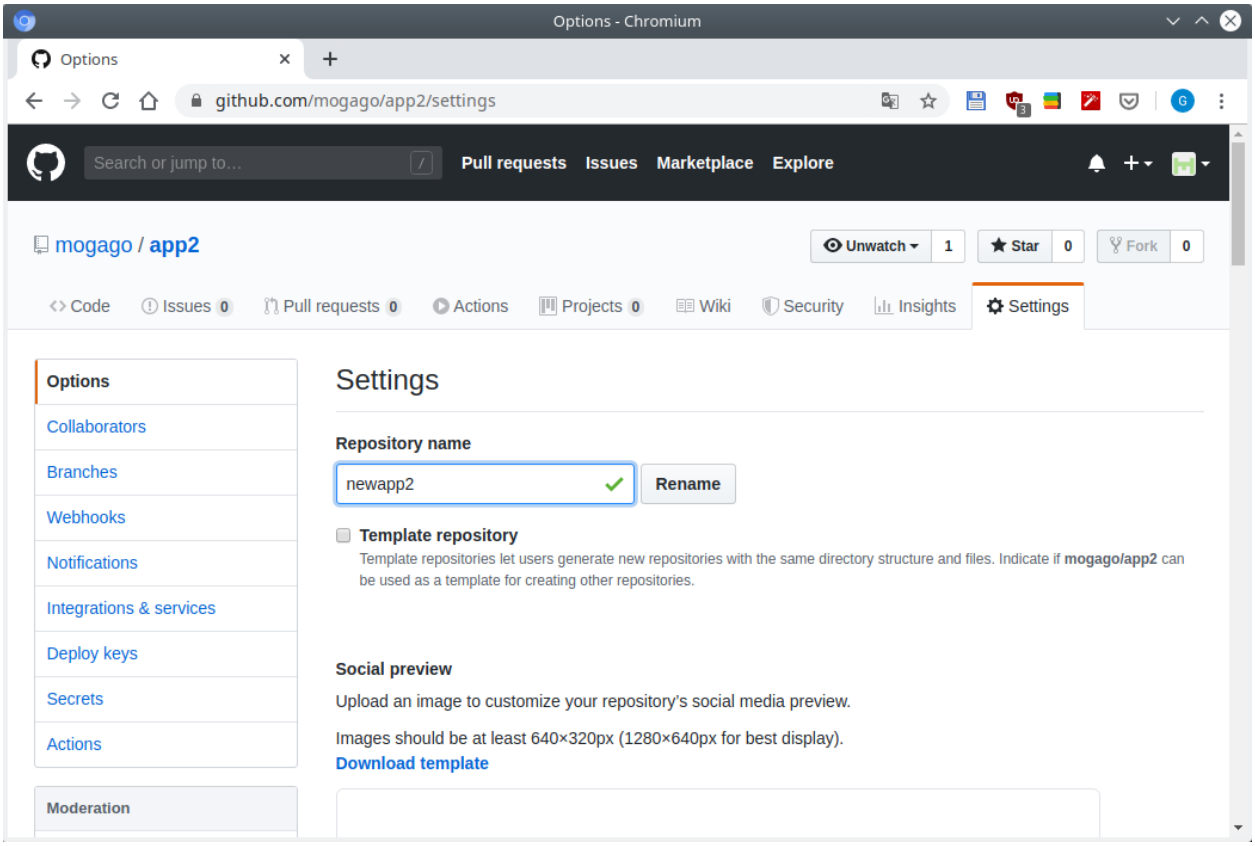


Fig. 49: GitHub - Renaming a repo

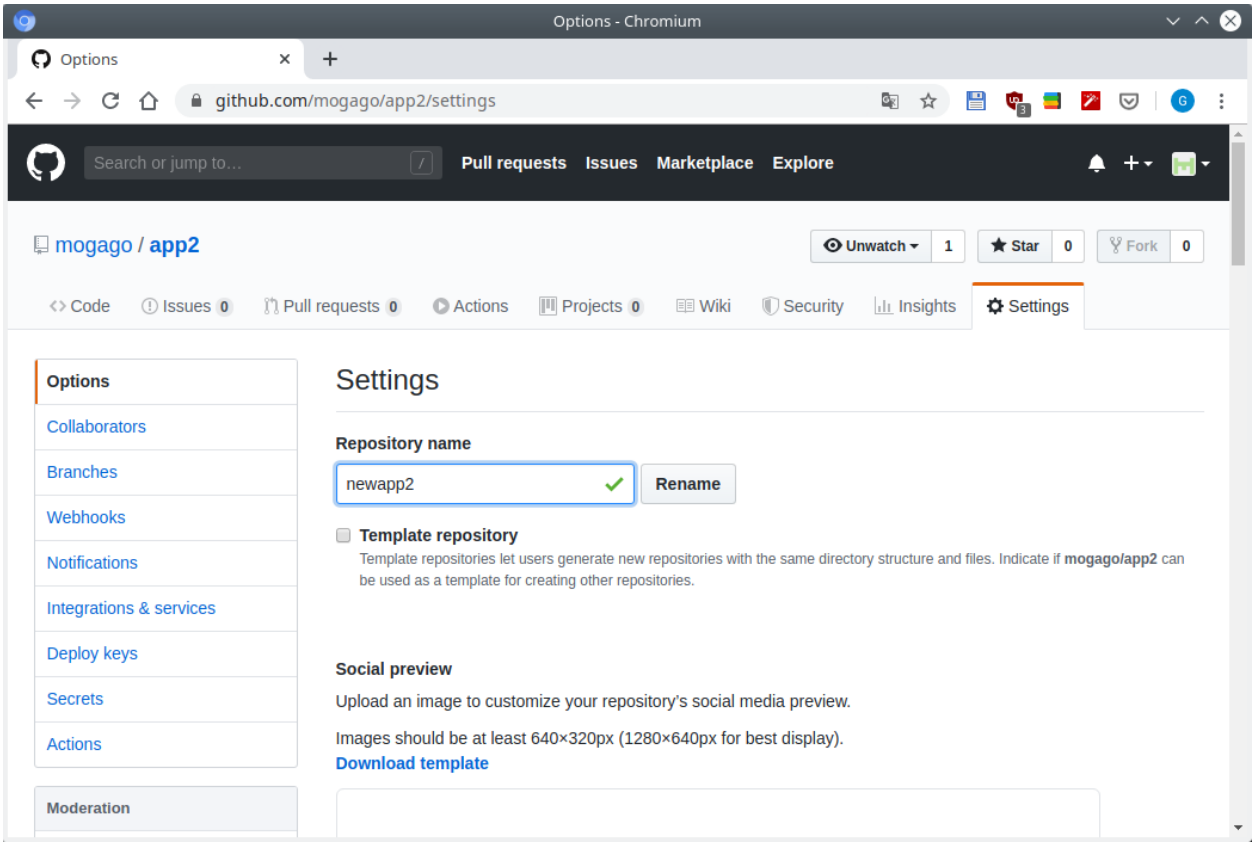


Fig. 50: GitHub - Repo renamed

Dentro de la opción *Settings* donde veremos la sección de **GitHub Pages**:

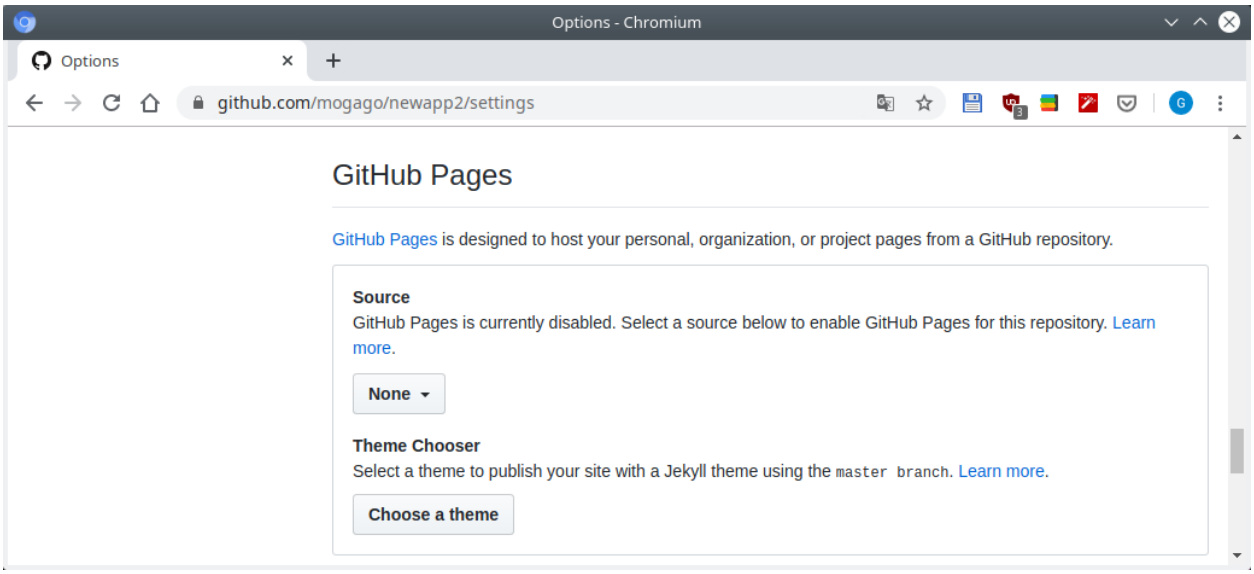


Fig. 51: GitHub - Settings, GitHub Pages

Al seleccionar *master branch* bajo la opción Source, tendremos otras opciones de selección como elegir un dominio personalizado:

Además se habrá creado nuestra página web bajo la siguiente dirección: https://mogago.github.io/newapp2/

En la sección de GitHub Pages podemos seleccionar un tema para la página web presionando el botón *Choose a theme*:

Seleccionar un tema de nuestro agrado y hacer clic en el botón *Select theme*.

3.9.3 Adding collaborators and integrations

Otra sección de *Settings* importante es la de *Collaborators*:

Si hay otra persona que deseamos sea capaz de hacer `push` al repositorio debemos agregarlo a esta lista de colaboradores. (Referencia para agregar colaboradores en: sección *Single repo collaboration directly on master* de *Lesson 7*). Sin embargo ellos no podrán agregar nuevos colaboradores o cambiar configuraciones de administrador.

Otra parte importante de integración con GitHub es trabajar con **web hooks** y **services**. Estas son unas formas de integrar aplicaciones de terceros en nuestro proyecto de GitHub.

Podemos agregar un webhook, especificando una URL que usamos para que nos contacten. Si estamos buscando escribir nuestra propia aplicación y deseamos recibir notificaciones en tiempo real cuando algo sucede en nuestro repositorio de GitHub, esta sección es la que buscamos:

Si queremos integrar algo que ya ha sido construido, lo mejor será usar **Services**:

3.9.4 Configuring deploy keys

La última sección de Settings que veremos es **Deploy keys**. Una de las formas que podemos lidiar con seguridad en GitHub es creando un **deploy key** para un repositorio en específico:

La razón por la que usaríamos esto, por ejemplo, sería cuando tengamos una aplicación que queramos que tenga acceso a este repositorio pero no a otros. Lo que haríamos es configurarlo con un deploy key (una llave SSH) e ingresaríamos

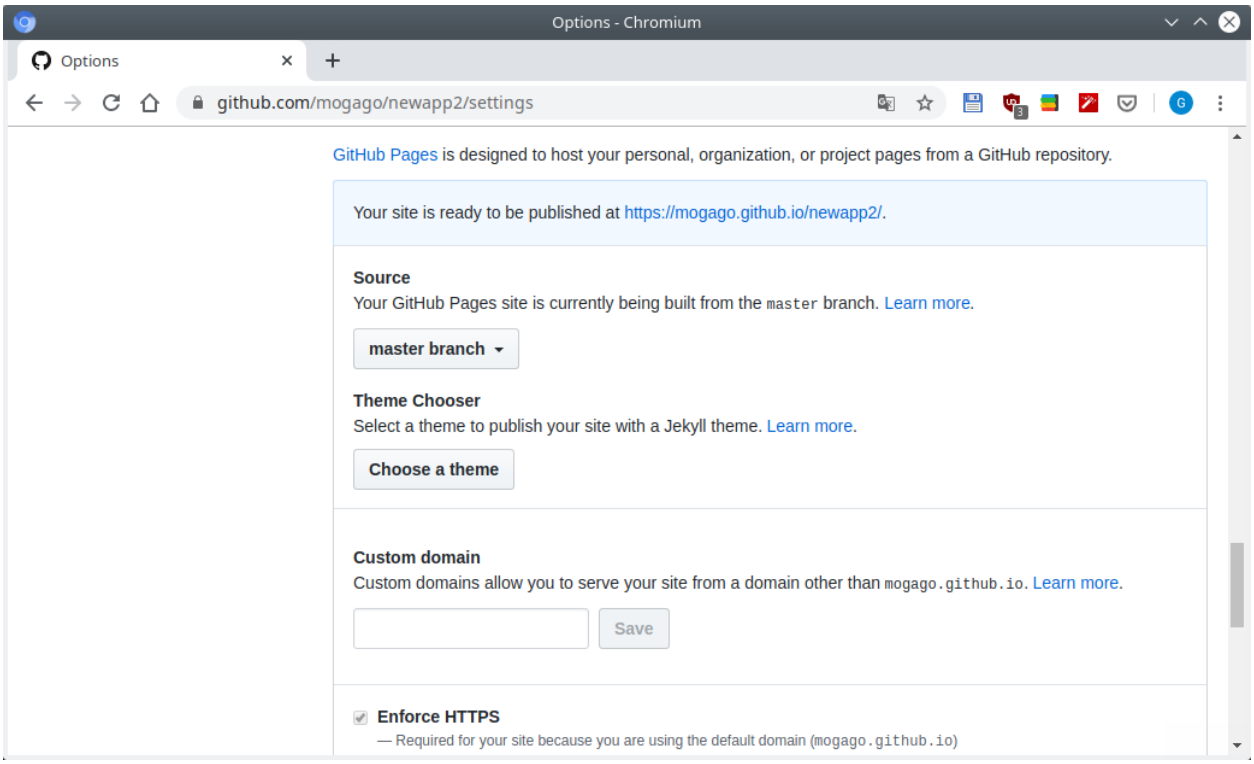


Fig. 52: GitHub - GitHub Pages, branch `master` seleccionado

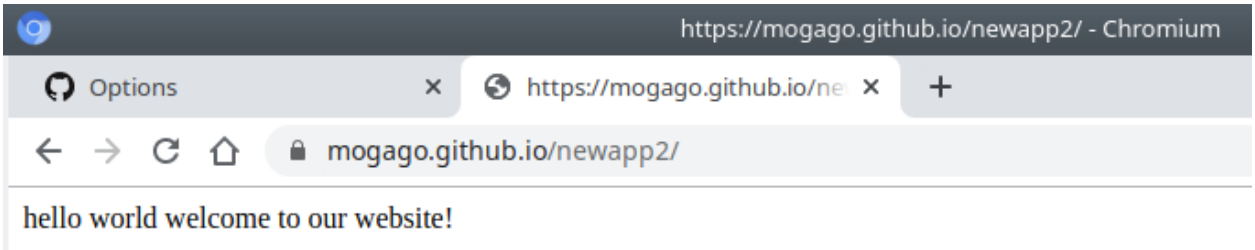


Fig. 53: GitHub - GitHub Pages webpage

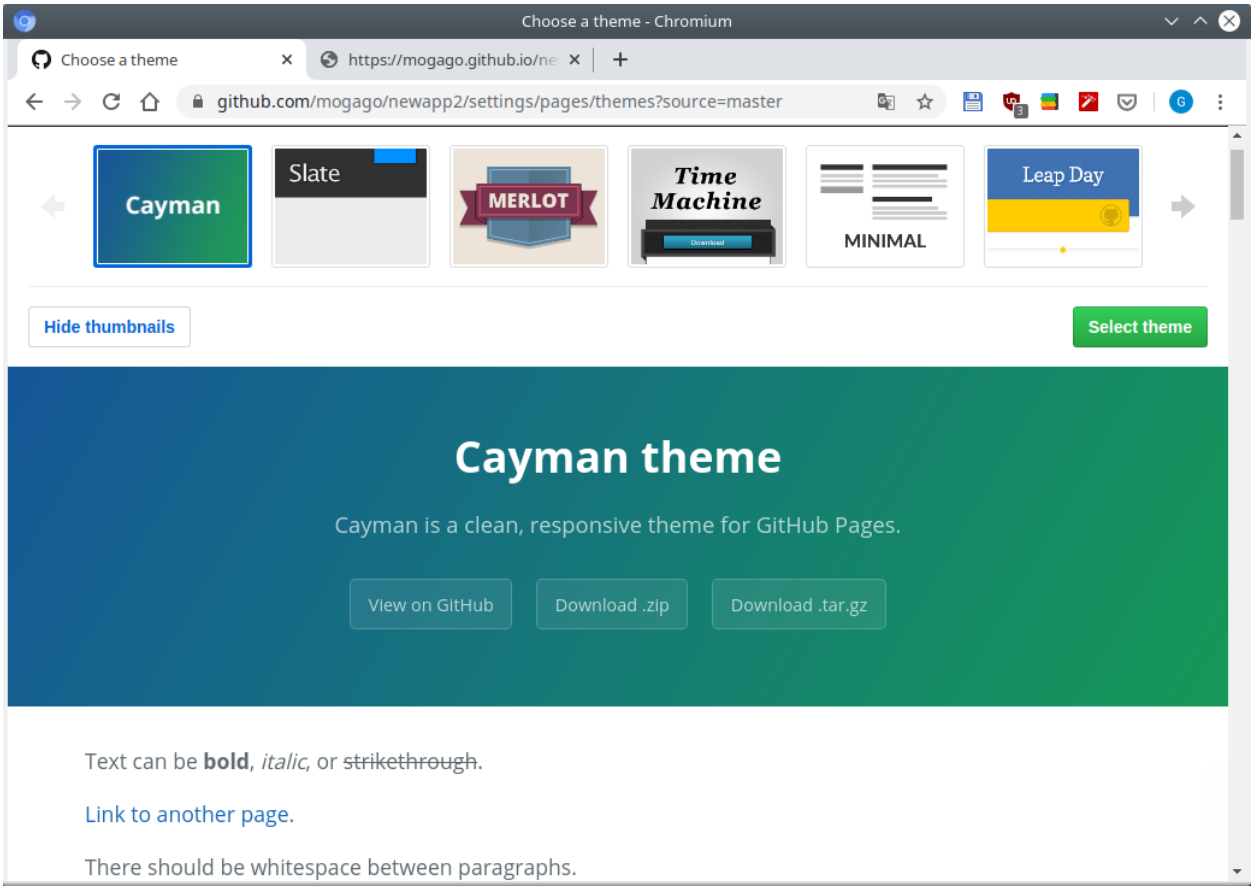


Fig. 54: GitHub - GitHub Pages themes

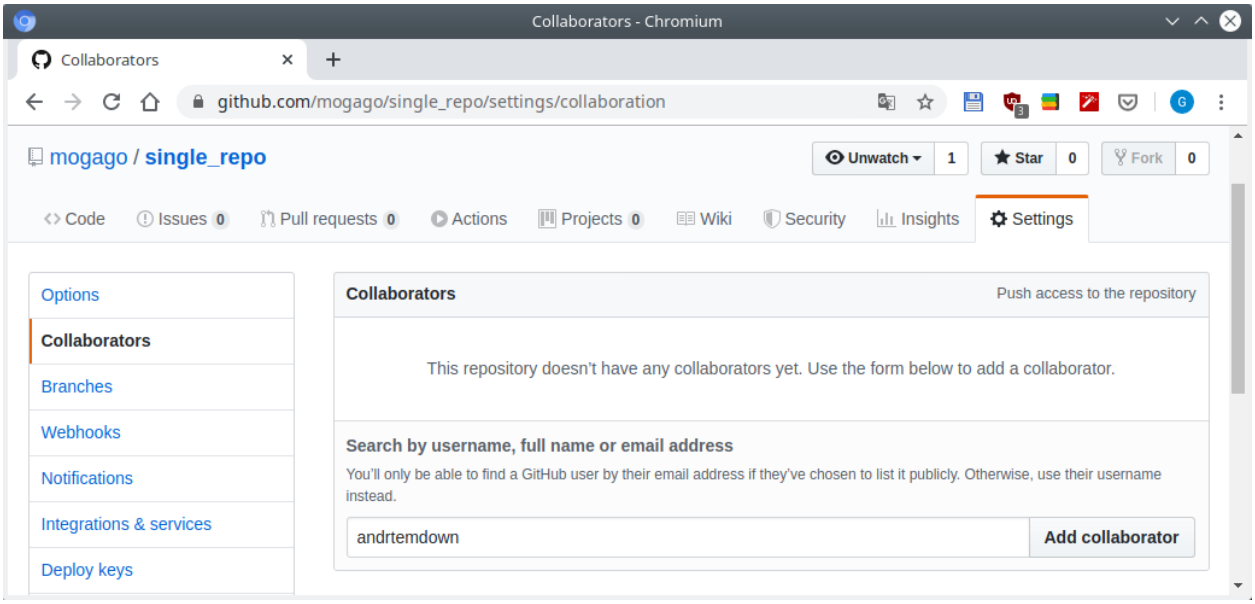


Fig. 55: GitHub - Añadir colaborador

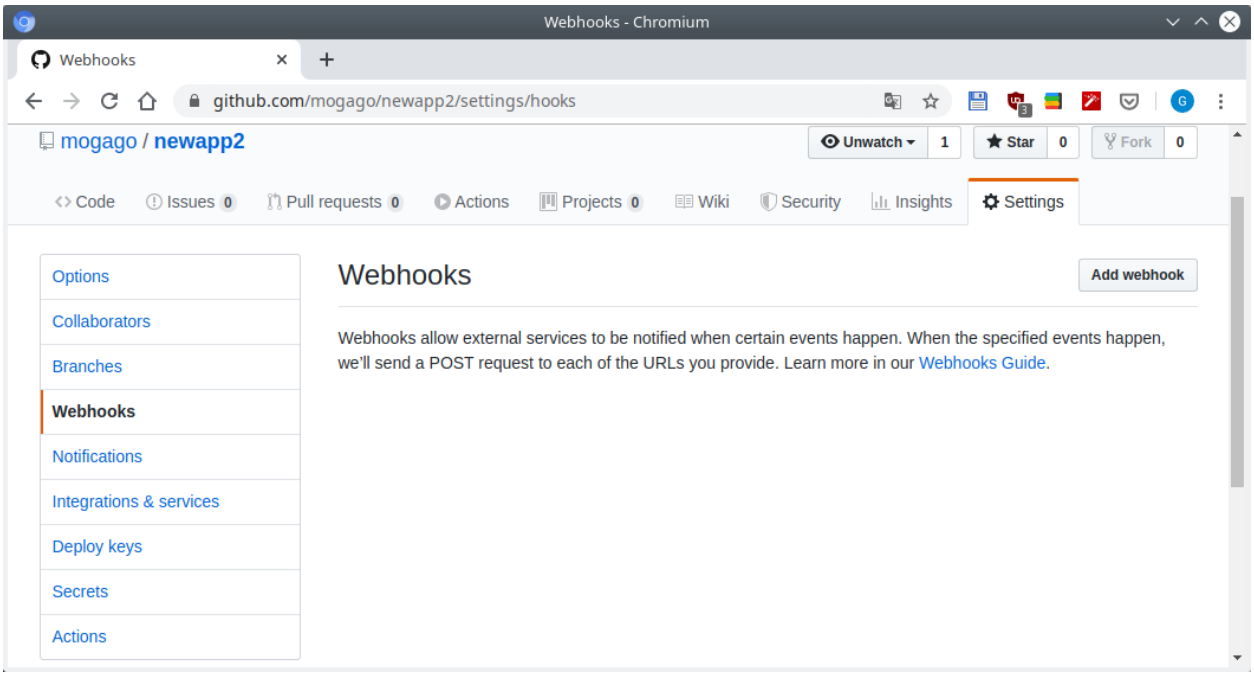


Fig. 56: GitHub webhooks

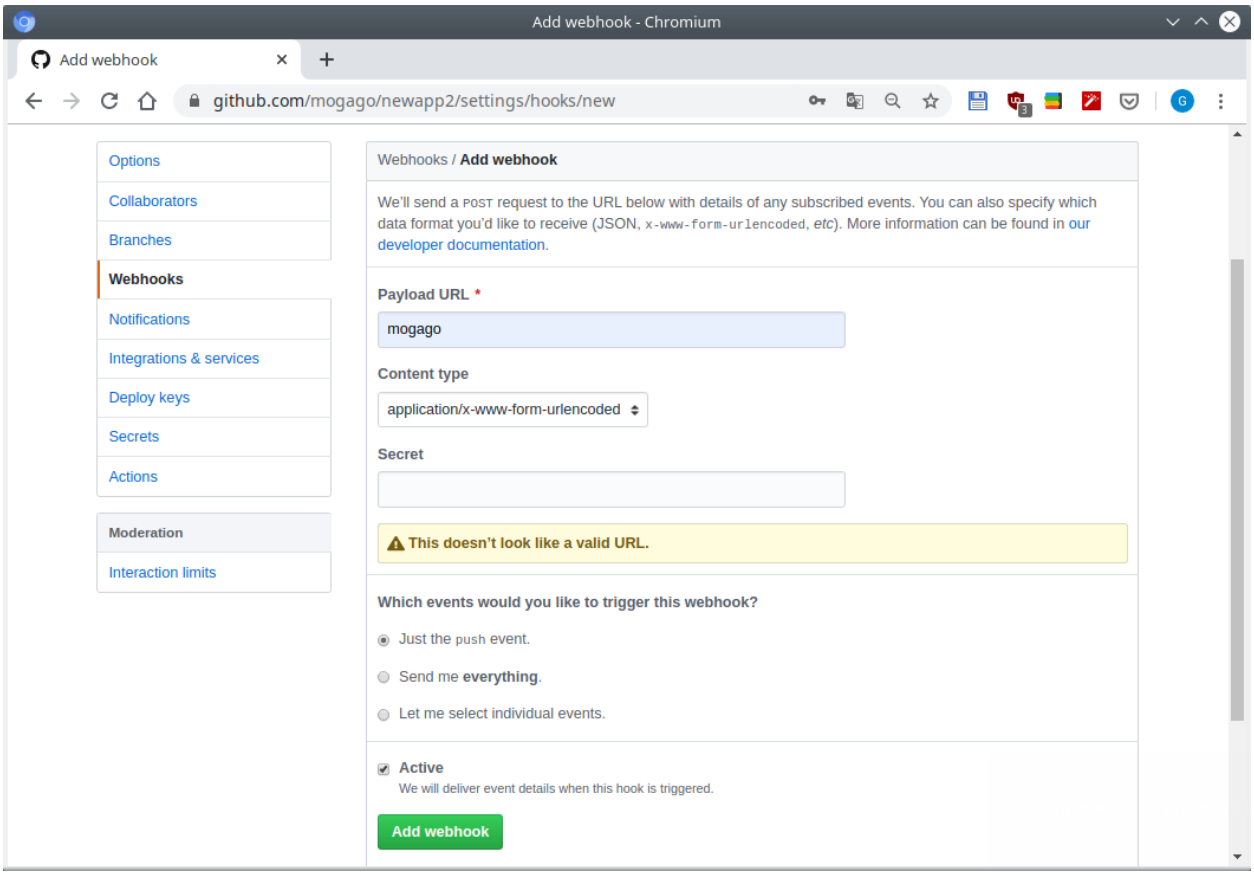


Fig. 57: GitHub webhooks

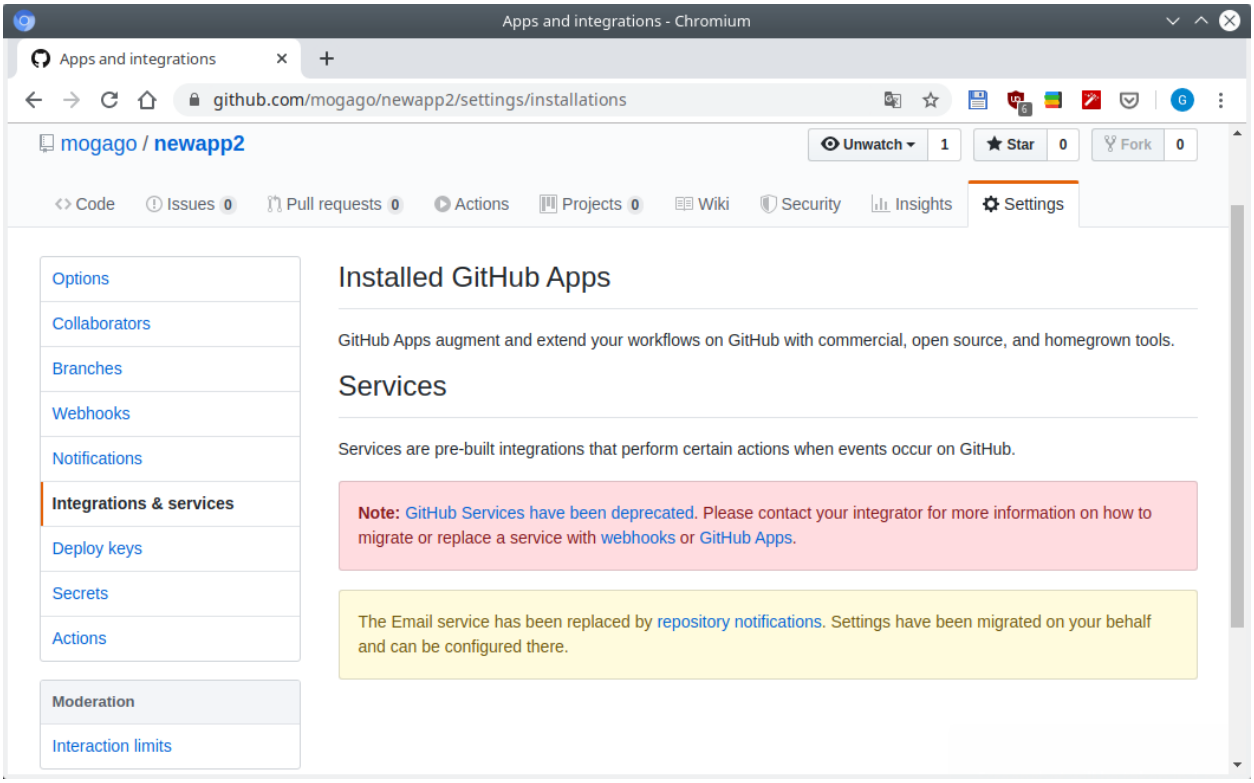


Fig. 58: GitHub service

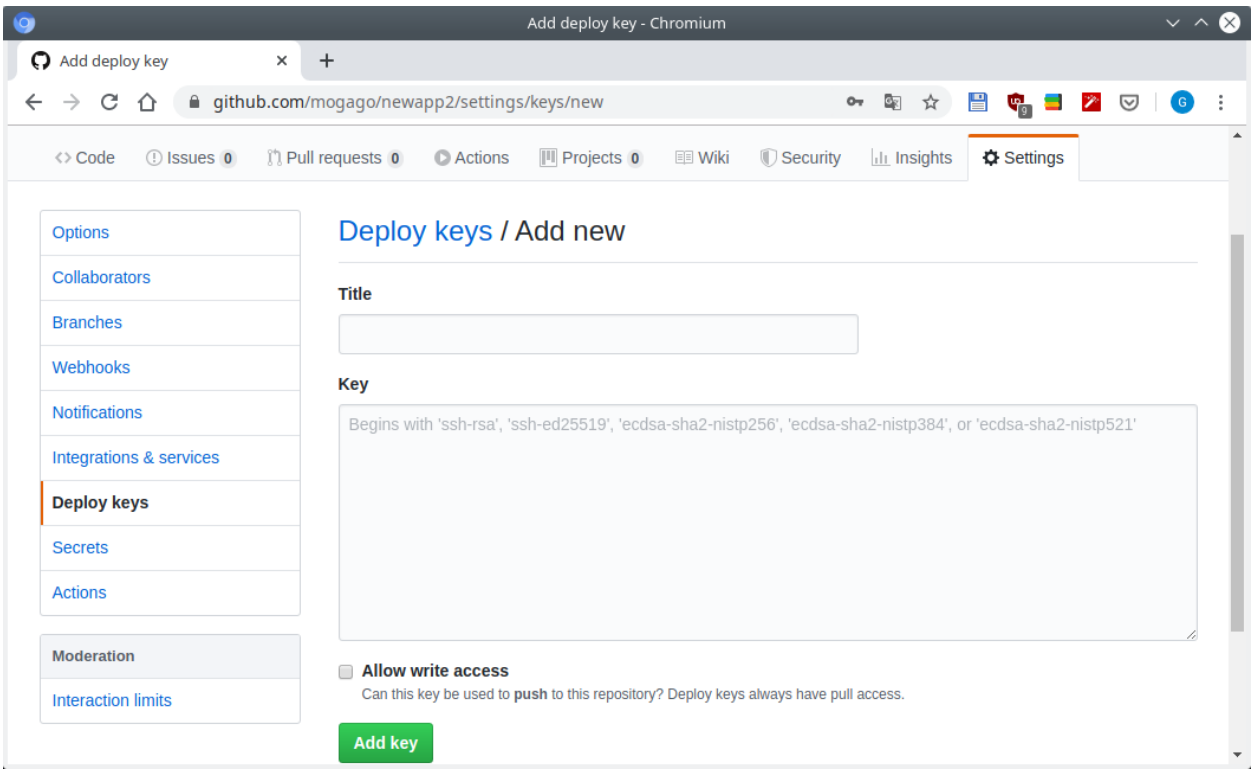


Fig. 59: GitHub deploy keys

esta información para que nuestro script tenga permisos para acceder y trabajar con ese repositorio en específico pero no con otros.

3.10 Lesson 10

Lesson 10: Tags and Releases

Table of Contents

3.10.1 Three types of tags

¿Cómo lanzar features?, ¿cómo hacemos seguimiento de nuestros lanzamientos usando GitHub?. Para hacer esto creemos otro proyecto:

```
$ git init app3
$ touch index.html
$ git add .
$ git commit -m "Added home page"
[master (root-commit) ebaf39d] Added home page
 1 file changed, 0 insertions(+), 0 deletions(-)
 create mode 100644 index.html
```

Generalmente, cuando lanzamos a producción, querremos crer un tag. La única excepción a esto es que si estamos corriendo despliegues continuos donde cada vez un `commit` termina en el branch `master`, termina automáticamente en producción. En este caso no se requiere etiquetarlo porque sabemos que todos acabarán en producción.

Si estamos más lejos de una entrega continua, cada vez que hagamos push de master a producción podemos crear un tag. Hay tres tipos de tags en Git:

1. Lightweight tags: solo nos dirá que existe un tag para un commit en particular.

```
$ git tag v1.0.0
$ git lg
* ebaf39d (HEAD -> master, tag: v1.0.0) Added home page

$ git show v1.0.0
commit ebaf39d20013c69e4d21544f74f5271d8d9ed1ad (tag: v1.0.0)
Author: Nombre Apellido <newuser1@mail.com>
Date:   Wed Jan 1 21:21:22 2020 -0500

    Added home page

diff --git a/index.html b/index.html
```

(continues on next page)

(continued from previous page)

```
new file mode 100644
index 0000000..e69de29
```

2. Annotated tags: (opción `tag -a`), usamos un annotated tag versus un lightweight tag para mostrar quién hizo el tag, a cuándo y qué dijeron cuando hicieron el tag, a parte del último commit realizado.

Generalmente generamos este tag justo antes de pasar el proyecto a producción.

```
$ touch index.css
$ git add .
$ git commit -m "Added stylesheet for homepage"
[master 128d3cd] Added stylesheet for homepage
 1 file changed, 0 insertions(+), 0 deletions(-)
 create mode 100644 index.css

$ git tag -a v1.0.1 -m  "Updated homepage to include stylesheet"
$ git lg
* 128d3cd (HEAD -> master, tag: v1.0.1) Added stylesheet for homepage
* ebaf39d (tag: v1.0.0) Added home page

$ git show v1.0.1
tag v1.0.1
Tagger: Nombre Apellido <newuser1@mail.com>
Date:   Wed Jan 1 21:30:42 2020 -0500

Updated homepage to include stylesheet

commit 128d3cdd49d95e9b264d43ccf770ae28e18e9cd4 (HEAD -> master, tag: v1.0.1)
Author: Nombre Apellido <newuser1@mail.com>
Date:   Wed Jan 1 21:30:02 2020 -0500

    Added stylesheet for homepage

diff --git a/index.css b/index.css
new file mode 100644
index 0000000..e69de29
```

El tag se aplica específicamente a un commit en espeífico, no a un branch.

3. Signed tags: (opción `tag -s`), usado para tags criptográfico, donde se necesita más niveles de seguridad y confirmación de que fue la persona quien lo creó.

3.10.2 Release tags versus release branches

Veamos con los tags encajan en un proceso de release. Cuando lanzamos algo a producción, seguremante lo etiquetaremos. Pero, ¿cómo decido entre release tags y release branches?:

Un buen punto de inicio es cuando solo tenemos release tags, a menos que necesitemos release branches. ¿Cuándo necesitamos un release branch?: si nuestro cliente está pagando para soportar branches de larga duración, es decir, dar soporte a la versión 1, 2, 3, ... del código.

Otro razón para usar release branches si tenemos un proceso manual y necesitamos hacer varios commits antes de lanzar a producción. Pero si no tenemos branches de larga duración, y no tenemos bugs o fallos, podemos usar un release tag.

[...]

3.10.3 Cherry pick for reusing code across long running release branches

¿Cómo vamos a tratar long-runnning release branches?.

[...]

3.10.4 Git stash for reusing code

Hay otra forma de reusar pequeñas unidades de código: `git stash`.

```
$ git checkout master
Already on 'master'

$ vi index.html
$ cat index.html
Add some new content

$ git s
M index.html
```

Podríamos estar haciendo varios cambios y modificaciones pero alguien necesita hacer un arreglo rápido o hacer un mismo cambio a múltiples branches.

Para mover nuestro trabajo hacia un lado haremos lo siguiente:

```
$ git s
$ git status
On branch master
nothing to commit, working tree clean
$ git lg
*   45b2ffb (refs/stash) WIP on master: 128d3cd Added stylesheet for homepage
|\
| * 34d74a4 index on master: 128d3cd Added stylesheet for homepage
|/
* 128d3cd (HEAD -> master, tag: v1.0.1) Added stylesheet for homepage
* ebaf39d (tag: v1.0.0) Added home page
```

El estado del repositorio nos dice que estamos libres, pero en el log veremos un `commit (refs/stash)` que está siguiendo nuestro cambios.

Ahora podemos aplicar este trabajo a todos los release branches:

[...]

3.10.5 Pushing tags up to GitHub and using releases

[...]

3.11 Lesson 11

Lesson 11: How to undo almost everything using Git

Table of Contents

3.11.1 Private versus public history and git revert

Veamos ahora cómo deshacer casi todo usando Git. La primera área que cubriremos es la diferencia entre historial público y privado, y qué comando debemos usar.

El historial público es aquel historial al que le hemos hecho `push` hacia Git en un repositorio en el que al menos una persona tiene acceso, ya sea un colaborador o un proyecto público. Tan pronto hemos creado un proyecto público debemos asumir que alguien más tiene nuestro código fuente.

Como regla de oro, solo hay una forma en la que queremos cambiar el historial público, será a través del comando `git revert`. Lo que hace este comando es tomar un `commit` existente y añade un `commit` que hace exactamente lo contrario: deshacer nuestro trabajo. Veamos un ejemplo:

```
$ git init undoing
Initialized empty Git repository in /home/user/Documents/gittests/undoing/.git/

$ cd undoing/
$ touch index.html
$ git add .
$ git commit -m "Added home page"
[master (root-commit) 29519be] Added home page
 1 file changed, 0 insertions(+), 0 deletions(-)
 create mode 100644 index.html

$ touch badidea.html
$ git add .
$ git commit -m "Bad idea"
[master 4e9acd9] Bad idea
 1 file changed, 0 insertions(+), 0 deletions(-)
 create mode 100644 badidea.html

$ git lg
* 4e9acd9 (HEAD -> master) Bad idea
* 29519be Added home page
```

Creemos el repositorio en GitHub:

En el terminal agregamos un remote:

```
$ git remote add origin https://github.com/mogago/undoing.git
$ git push -u origin master
Username for 'https://github.com': mogago
```

(continues on next page)

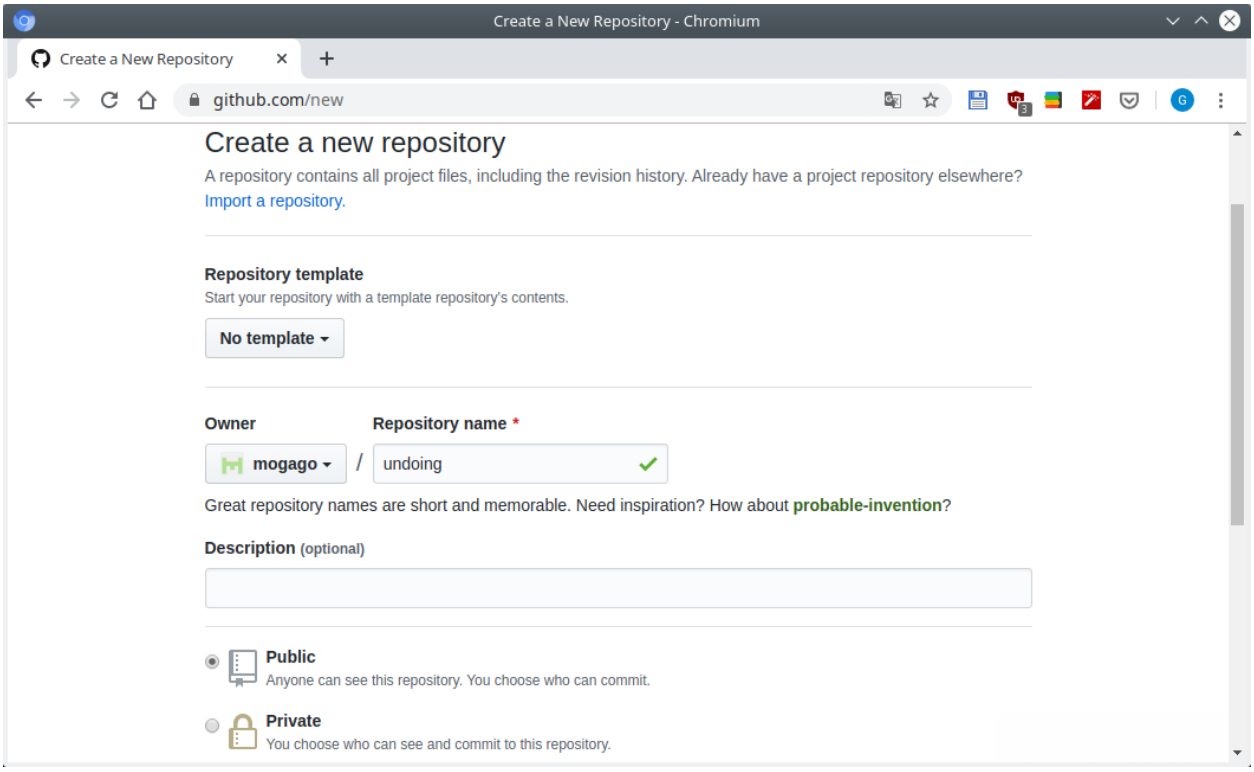


Fig. 60: GitHub - Creating a repo

(continued from previous page)

```
Password for 'https://mogago@github.com':
Counting objects: 5, done.
Delta compression using up to 4 threads.
Compressing objects: 100% (3/3), done.
Writing objects: 100% (5/5), 429 bytes | 429.00 KiB/s, done.
Total 5 (delta 0), reused 0 (delta 0)
To https://github.com/mogago/undoing.git
 * [new branch]      master -> master
Branch 'master' set up to track remote branch 'master' from 'origin'.
```

Ahora tenemos historial público, por ejemplo veremos los `commits`:

Si hemos compartido públicamente algo que no deseamos que lo sea, podemos usar varios comandos para borrar esos commits o cambiarlos. Sin embargo, tan pronto como hemos compartido esa información con el mundo, debemos usar `git revert` con el identificador hash SHA1 del `commit`:

```
$ git revert 4e9acd90b9d191414f2fe4a8f16f768125ed7287

    Revert "Bad idea"
    This reverts commit 4e9acd90b9d191414f2fe4a8f16f768125ed7287.
```

Se abrirá un editor de texto para agregar un mensaje al `commit` de revertir el cambio. Hagamos `push` y veamos el log:

```
$ git push

$ git lg
```

(continues on next page)

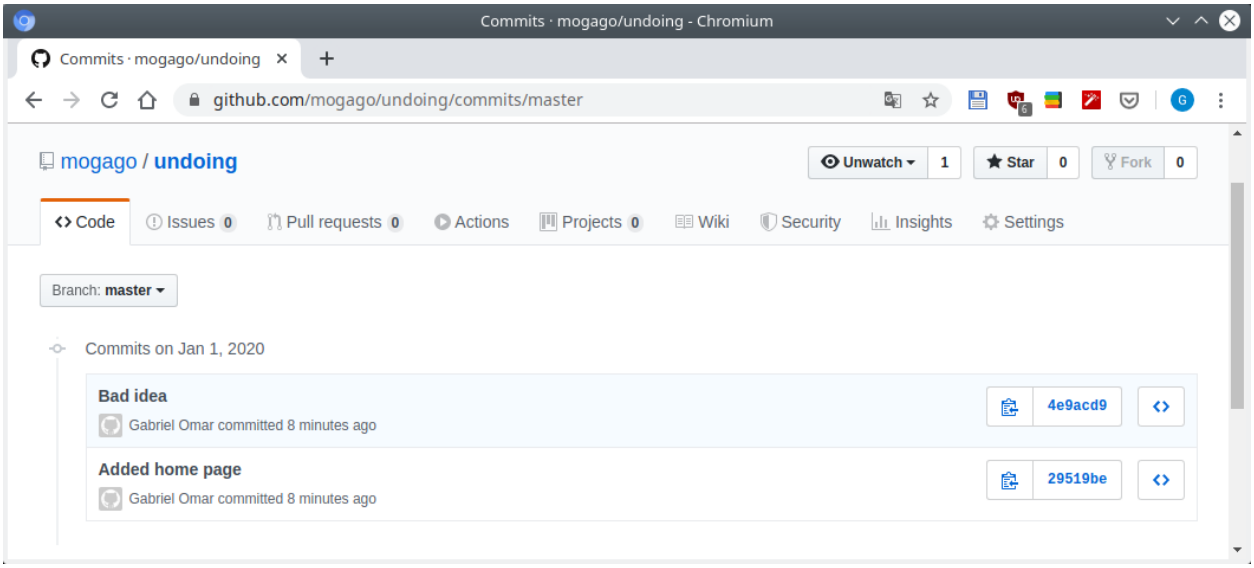


Fig. 61: GitHub - Commits

(continued from previous page)

```
* c76cba8 (HEAD -> master) Revert "Bad idea"
* 4e9acd9 (origin/master) Bad idea
* 29519be Added home page
```

Tenemos dos `commits`, no es bueno pero al menos es seguro. Esto significa que si cualquier otra persona descarga una copia del repositorio no tendrán problemas porque tendrán un `commit` que ocasiona el problema y otro `commit` posterior que lo arregla.

Lo importante es que eliminado el problema de borrar ese código problemático (`badidea.html`):

```
$ ls
index.html
```

3.11.2 Don't push too often

Si estamos trabajando con cosas que no le hemos hecho `push`, hay muchas otras formas que podemos usar para reescribir el historial. Debido a esta distinción entre historial público y privado, cambia drásticamente la forma de trabajo con Git y GitHub.

Experiencia: antes mi manera de trabajo era escribir un poco de código y hacía `add`, `commit`, `push`. Luego escribía más código y hacía `add`, `commit`, `push`. Haciendo `push` a GitHub cada 5-10 minutos. Ahora trabajo de forma diferente porque hay varias herramientas que se pueden usar para limpiar el historial. Ahora escribo código y solo hago `add`, `commit`, desarrollo más código y hago `add`, `commit`, pero no hago `push`. Puedo seguir haciendo un commit cada 5 minutos pero hago mucho menos `push` a GitHub. La ventaja de esto es que antes de hacer `push` a los cambios, puedo ver lo `commits` realizados y limpiar el historial.

En las siguientes secciones se verán algunos comando para limpiar el historial.

3.11.3 Git commit –ammend

Uno de los comandos más fáciles es `git commit --ammend`. Veamos un ejemplo:

```
$ ls
index.html
$ touch about.html
$ git add .
$ git commit -m "Added about us page dasndksajdnjksa"
[master c425487] Added about us page dasndksajdnjksa
 1 file changed, 0 insertions(+), 0 deletions(-)
 create mode 100644 about.html

$ git lg
* c425487 (HEAD -> master) Added about us page dasndksajdnjksa
* c76cba8 (origin/master) Revert "Bad idea"
* 4e9acd9 Bad idea
* 29519be Added home page
```

En este caso hemos cometido un error al escribir el mensaje del `commit`. Esto claramente se ve mal. Siempre y cuando sea el último `commit`, podemos arreglarlo con:

```
$ git commit --amend

    Added about us page

$ git lg
* c9d0d2a (HEAD -> master) Added about us page
* c76cba8 (origin/master) Revert "Bad idea"
* 4e9acd9 Bad idea
* 29519be Added home page
```

Al haber ejecutado el comando `git commit --amend` nos retorna a un editor de texto en el que podemos editar el mensaje del `commit`. Notar que el hash del `commit` cambiará, pues es otra hora y otro mensaje.

Otro uso de este comando es cuando no hacemos `commit` a todo lo que queríamos. Veamos el ejemplo:

```
$ touch contact.html
$ vi index.html
$ cat index.html
link to contact us page

$ git s
M index.html
?? contact.html

$ git commit -am "Added contact us page"
 [master 7e762db] Added contact us page
 1 file changed, 1 insertion(+)

$ git s
?? contact.html

$ git lg
* 7e762db (HEAD -> master) Added contact us page
* c9d0d2a Added about us page
* c76cba8 (origin/master) Revert "Bad idea"
* 4e9acd9 Bad idea
* 29519be Added home page

$ git show 7e762db
```

(continues on next page)

(continued from previous page)

```
commit 7e762dbdca91e202b3bfd3d8461bd68948bf3d58 (HEAD -> master)
Author: Nombre Apellido <newuser1@mail.com>
Date:   Wed Jan 1 23:33:41 2020 -0500

    Added contact us page

diff --git a/index.html b/index.html
index e69de29..79cfe97 100644
--- a/index.html
+++ b/index.html
@@ -0,0 +1 @@
+link to contact us page
```

En este caso, el `commit -am` lo ha tomado solo el archivo modificado, pues el otro no ha sido añadido con `git add`. Nuestro `commit` indica que hemos agregado la página de contacto pero esta no está siendo trackeada por Git, como lo indica el estado del repositorio.

Para solucionar esto ejecutaremos:

```
$ git add .
$ git s
A  contact.html
$ git commit --amend
[master 94ffa02] Added contact us page
 Date: Wed Jan 1 23:33:41 2020 -0500
 2 files changed, 1 insertion(+)
 create mode 100644 contact.html

Added contact us page

# Please enter the commit message for your changes. Lines starting
# with '#' will be ignored, and an empty message aborts the commit.
#
# Date:      Wed Jan 1 23:33:41 2020 -0500
#
# On branch master
# Your branch is ahead of 'origin/master' by 2 commits.
#   (use "git push" to publish your local commits)
#
# Changes to be committed:
#       new file:   contact.html
#       modified:   index.html
```

`git commit --amend` también sirve para agregar un archivo que se nos olvidó poner en el `commit`.

```
$ git lg
* 94ffa02 (HEAD -> master) Added contact us page
* c9d0d2a Added about us page
* c76cba8 (origin/master) Revert "Bad idea"
* 4e9acd9 Bad idea
* 29519be Added home page

$ git show 94ffa02
commit 94ffa02184f7d4e69d0086c82f1738507bf507c6 (HEAD -> master)
Author: Gabriel <newuser1@mail.com>
Date:   Wed Jan 1 23:33:41 2020 -0500
```

(continues on next page)

(continued from previous page)

```
    Added contact us page

diff --git a/contact.html b/contact.html
new file mode 100644
index 0000000..e69de29
diff --git a/index.html b/index.html
index e69de29..79cfe97 100644
--- a/index.html
+++ b/index.html
@@ -0,0 +1 @@
+link to contact us page

$ git s
$
```

3.11.4 Git reset

Si tenemos un `commit` antiguo que debemos arreglar no podemos usar `git commit --amend` porque solo funciona en el último `commit`. Lo que debemos hacer es usar `git reset`. Veamos los varios modos que tiene este comando, cuándo y cómo usarlos.

Supongamos que estamos desarrollando una página e-commerce:

```
$ touch categories.html categories.css products.html products.css
$ git s
?? categories.css
?? categories.html
?? products.css
?? products.html
```

Supongamos que como desarrolladores inexpertos decidimos hacer un `commit` para las páginas `html` y otro para las `css`:

```
$ git add *.html
$ git commit -m "Added new pages to the site"
[master 2079fef] Added new pages to the site
2 files changed, 0 insertions(+), 0 deletions(-)
create mode 100644 categories.html
create mode 100644 products.html

$ git add .
$ git commit -m "Styled new pages"
[master a03eaf5] Styled new pages
2 files changed, 0 insertions(+), 0 deletions(-)
create mode 100644 categories.css
create mode 100644 products.css

$ git lg
* a03eaf5 (HEAD -> master) Styled new pages
* 2079fef Added new pages to the site
* 94ffa02 Added contact us page
* c9d0d2a Added about us page
* c76cba8 (origin/master) Revert "Bad idea"
* 4e9acd9 Bad idea
* 29519be Added home page
```

Esta forma de trabajo no es buena porque no describimos con qué páginas estamos trabajando en los `commits`. El propósito de `git reset` es eliminar `commits`. Hay 3 niveles de `git reset`:

1. `git reset --soft`: deja los archivos del `commit` en el staging area para que se les pueda aplicar un `commit` nuevamente.
2. `git reset --hard`: no solo va a eliminar los `commit`, sino que eliminará todo nuestros archivos involucrados en el `commit`.
3. `git reset --mixed` o `git reset`: opción por defecto. Elimina los `commits`, limpia la staging area pero deja los archivos en el directorio de trabajo, para que puedan ser agregados y hacerles `commit` nuevamente.

Para este caso que deseamos eliminar los dos últimos commits podemos usar el último caso:

```
$ git reset HEAD~2
$ git s
?? categories.css
?? categories.html
?? products.css
?? products.html

$ git add ca*
$ git commit -m "Added categories page"
[master 8943c44] Added categories page
2 files changed, 0 insertions(+), 0 deletions(-)
create mode 100644 categories.css
create mode 100644 categories.html

$ git add .
$ git commit -m "Added products page"
[master 3d2c3e0] Added products page
2 files changed, 0 insertions(+), 0 deletions(-)
create mode 100644 products.css
create mode 100644 products.html

$ git lg
* 3d2c3e0 (HEAD -> master) Added products page
* 8943c44 Added categories page
* 94ffa02 Added contact us page
* c9d0d2a Added about us page
* c76cba8 (origin/master) Revert "Bad idea"
* 4e9acd9 Bad idea
* 29519be Added home page
```

Si ahora deseamos menos granularidad en los `commits` y transformar los últimos dos `commits` en uno solo podríamos usar:

```
$ git commit -m "Added e-commerce page"
[master c1d6629] Added e-commerce page
4 files changed, 0 insertions(+), 0 deletions(-)
create mode 100644 categories.css
create mode 100644 categories.html
create mode 100644 products.css
create mode 100644 products.html
$ git lg
* c1d6629 (HEAD -> master) Added e-commerce page
* 94ffa02 Added contact us page
* c9d0d2a Added about us page
* c76cba8 (origin/master) Revert "Bad idea"
```

(continues on next page)

(continued from previous page)

```
* 4e9acd9 Bad idea
* 29519be Added home page
```

Si ahora deseamos eliminar todo rastro de trabajo y los `commits` usaremos:

```
$ git reset --hard HEAD~1
HEAD is now at 94ffa02 Added contact us page

$ ls
about.html  contact.html  index.html

$ git lg
* 94ffa02 (HEAD -> master) Added contact us page
* c9d0d2a Added about us page
* c76cba8 (origin/master) Revert "Bad idea"
* 4e9acd9 Bad idea
* 29519be Added home pag
```

3.11.5 Introducing the reflog

Veremos cómo usar reflog para traer archivos de vuelta. Reflog no trabaja en cualquier situación, solo mantiene seguimiento de todas las cosas que hacemos en Git. Si estamos en la misma PC, no hemos corrido el recolector de basura en Git y estamos buscando un cambio de los últimos 30 días podremos usar Reflog para recuperar archivos de vuelta. Veamos un ejemplo:

```
$ git reflog
94ffa02 (HEAD -> master) HEAD@{0}: reset: moving to HEAD~1
c1d6629 HEAD@{1}: commit: Added e-commerce page
94ffa02 (HEAD -> master) HEAD@{2}: reset: moving to HEAD~2
3d2c3e0 HEAD@{3}: commit: Added products page
8943c44 HEAD@{4}: commit: Added categories page
94ffa02 (HEAD -> master) HEAD@{5}: reset: moving to HEAD~2
a03eaf5 HEAD@{6}: commit: Styled new pages
2079fef HEAD@{7}: commit: Added new pages to the site
94ffa02 (HEAD -> master) HEAD@{8}: commit (amend): Added contact us page
7e762db HEAD@{9}: commit: Added contact us page
c9d0d2a HEAD@{10}: commit (amend): Added about us page
c42641d HEAD@{11}: commit (amend): Added about us page
c425487 HEAD@{12}: commit: Added about us page dasndksajdnjksa
c76cba8 (origin/master) HEAD@{13}: revert: Revert "Bad idea"
4e9acd9 HEAD@{14}: commit: Bad idea
29519be HEAD@{15}: commit (initial): Added home page
```

Buscaremos el hash del commit que deseemos recuperar, por ejemplo `c1d6629 HEAD@{1}: commit: Added e-commerce page`, luego:

```
$ git reset --hard c1d6629
HEAD is now at c1d6629 Added e-commerce page

$ git lg
* c1d6629 (HEAD -> master) Added e-commerce page
* 94ffa02 Added contact us page
* c9d0d2a Added about us page
* c76cba8 (origin/master) Revert "Bad idea"
```

(continues on next page)

(continued from previous page)

```
* 4e9acd9 Bad idea
* 29519be Added home page

$ ls
about.html      categories.html  index.html    products.html
categories.css  contact.html     products.css
```

Veremos que tenemos nuestro archivos y el `commit` de vuelta. Pero lo hicimos reiniciando `--hard` y esta no siempre es una opción.

Veamos cómo recuperar el historial si hemos hecho algunos `commmits`:

```
$ git reset --hard HEAD~1
HEAD is now at 94ffa02 Added contact us page

$ git lg
* 94ffa02 (HEAD -> master) Added contact us page
* c9d0d2a Added about us page
* c76cba8 (origin/master) Revert "Bad idea"
* 4e9acd9 Bad idea
* 29519be Added home page

$ touch test.html
$ git add .
$ git commit -m "Added a test file"
[master 85c090e] Added a test file
1 file changed, 0 insertions(+), 0 deletions(-)
create mode 100644 test.html

$ git lg
* 85c090e (HEAD -> master) Added a test file
* 94ffa02 Added contact us page
* c9d0d2a Added about us page
* c76cba8 (origin/master) Revert "Bad idea"
* 4e9acd9 Bad idea
* 29519be Added home page
```

Ahora que hemos eliminado la funcionalidad de e-commerce y hemos creado un archivo `test.html`, si deseamos volver a un punto de la historia donde recuperamos la funcionalidad de e-commerce, perderíamos el archivo `test.html`. Esto no es conveniente, afortunadamente hay 2 formas de tener el commit de vuelta usando el reflog:

1. Mirar el reflog, copiar el hash del `commit` donde agregaba las páginas de e-commerce y usar `cherry-pick`:

```
$ git reflog
85c090e (HEAD -> master) HEAD@{0}: commit: Added a test file
94ffa02 HEAD@{1}: reset: moving to HEAD~1
c1d6629 HEAD@{2}: reset: moving to c1d6629
94ffa02 HEAD@{3}: reset: moving to HEAD~1
c1d6629 HEAD@{4}: commit: Added e-commerce page
94ffa02 HEAD@{5}: reset: moving to HEAD~2
3d2c3e0 HEAD@{6}: commit: Added products page
8943c44 HEAD@{7}: commit: Added categories page
94ffa02 HEAD@{8}: reset: moving to HEAD~2
a03eaf5 HEAD@{9}: commit: Styled new pages

$ git cherry-pick c1d6629
[master c5631c6] Added e-commerce page
```

(continues on next page)

(continued from previous page)

```
 Date: Thu Jan 2 00:10:44 2020 -0500
 4 files changed, 0 insertions(+), 0 deletions(-)
 create mode 100644 categories.css
 create mode 100644 categories.html
 create mode 100644 products.css
 create mode 100644 products.html

$ git lg
* c5631c6 (HEAD -> master) Added e-commerce page
* 85c090e Added a test file
* 94ffa02 Added contact us page
* c9d0d2a Added about us page
* c76cba8 (origin/master) Revert "Bad idea"
* 4e9acd9 Bad idea
* 29519be Added home page
```

Lo que hará será recuperar las páginas de e-commerce sobre mi trabajo actual sin borrarlo.

Si quisiera múltiples `commits` podría usar `cherry-pick` en cada uno pero más eficiente será crear un branch:

Borro el el último `commit` y hago checkout al punto en el tiempo en que estoy añadiendo las páginas de categories (`8943c44 HEAD@{7}: commit: Added categories page`) y products (`3d2c3e0 HEAD@{6}: commit: Added products page`):

```
$ git reset --hard HEAD~1
HEAD is now at 85c090e Added a test file
$ git lg
* 85c090e (HEAD -> master) Added a test file
* 94ffa02 Added contact us page
* c9d0d2a Added about us page
* c76cba8 (origin/master) Revert "Bad idea"
* 4e9acd9 Bad idea
* 29519be Added home page

$ git checkout 3d2c3e0
Note: checking out '3d2c3e0'.

You are in 'detached HEAD' state. You can look around, make experimental
changes and commit them, and you can discard any commits you make in this
state without impacting any branches by performing another checkout.

If you want to create a new branch to retain commits you create, you may
do so (now or later) by using -b with the checkout command again. Example:

    git checkout -b <new-branch-name>

HEAD is now at 3d2c3e0 Added products page
```

Creo un branch en ese punto en el tiempo y puedo hacer merge o rebase con el branch `master`:

```
$ git checkout -b tmp
Switched to a new branch 'tmp'
$ git lg
* 85c090e (master) Added a test file
| * 3d2c3e0 (HEAD -> tmp) Added products page
| * 8943c44 Added categories page
|/
```

(continues on next page)

(continued from previous page)

```
* 94ffa02 Added contact us page
* c9d0d2a Added about us page
* c76cba8 (origin/master) Revert "Bad idea"
* 4e9acd9 Bad idea
* 29519be Added home page
$ git checkout tmp
Already on 'tmp'

$ git rebase master
First, rewinding head to replay your work on top of it...
Applying: Added categories page
Applying: Added products page

$ git lg
* 7c7c945 (HEAD -> tmp) Added products page
* d2a6ff4 Added categories page
* 85c090e (master) Added a test file
* 94ffa02 Added contact us page
* c9d0d2a Added about us page
* c76cba8 (origin/master) Revert "Bad idea"
* 4e9acd9 Bad idea
* 29519be Added home page
```

Tengo estos dos `commits` de vuelta sin haber perdido los cambios hechos en medio.

3.11.6 Rebase interactive

Limpiemos el repositorio:

```
$ git checkout master
Switched to branch 'master'
Your branch is ahead of 'origin/master' by 3 commits.
  (use "git push" to publish your local commits)

$ git merge tmp
Updating 85c090e..7c7c945
Fast-forward
 categories.css  | 0
 categories.html | 0
 products.css    | 0
 products.html   | 0
 4 files changed, 0 insertions(+), 0 deletions(-)
 create mode 100644 categories.css
 create mode 100644 categories.html
 create mode 100644 products.css
 create mode 100644 products.html

$ git branch -d tmp
Deleted branch tmp (was 7c7c945).
$ git lg
* 7c7c945 (HEAD -> master) Added products page
* d2a6ff4 Added categories page
* 85c090e Added a test file
* 94ffa02 Added contact us page
* c9d0d2a Added about us page
```

(continues on next page)

(continued from previous page)

```
* c76cba8 (origin/master) Revert "Bad idea"
* 4e9acd9 Bad idea
* 29519be Added home page
```

Imaginemos que hemos estado trabajando en el código por un rato y queremos limpiar el historial antes de hacerle `push`. Lo que podemos hacer es ver los últimos commits y ejecutar:

```
$ git lg
* 7c7c945 (HEAD -> master) Added products page
* d2a6ff4 Added categories page
* 85c090e Added a test file
* 94ffa02 Added contact us page
* c9d0d2a Added about us page
* c76cba8 (origin/master) Revert "Bad idea"
* 4e9acd9 Bad idea
* 29519be Added home page

$ git rebase -i HEAD~5

pick c9d0d2a Added about us page
pick 94ffa02 Added contact us page
pick 85c090e Added a test file
pick d2a6ff4 Added categories page
pick 7c7c945 Added products page

# Rebase c76cba8..7c7c945 onto c76cba8 (5 commands)
#
# Commands:
# p, pick = use commit
# r, reword = use commit, but edit the commit message
# e, edit = use commit, but stop for amending
# s, squash = use commit, but meld into previous commit
# f, fixup = like "squash", but discard this commit's log message
# x, exec = run command (the rest of the line) using shell
# d, drop = remove commit
#
# These lines can be re-ordered; they are executed from top to bottom.
#
# If you remove a line here THAT COMMIT WILL BE LOST.
#
# However, if you remove everything, the rebase will be aborted.
#
# Note that empty commits are commented out
```

Esto me permitirá realizar una serie de acciones a los `commits` seleccionados. Puedo borrarlos, reordenarlos o comprimirlos en menos `commits`:

```
$ git rebase -i HEAD~5

pick c9d0d2a Added about us page
pick 94ffa02 Added contact us page
f 85c090e Added a test file
pick d2a6ff4 Added categories page
pick 7c7c945 Added products page

$ git lg
```

(continues on next page)

(continued from previous page)

```
* 6681986 (HEAD -> master) Added products page
* 7c963d9 Added categories page
* b5ebb9c Added contact us page
* c9d0d2a Added about us page
* c76cba8 (origin/master) Revert "Bad idea"
* 4e9acd9 Bad idea
* 29519be Added home page
```

Vemos que hemos eliminado el `commit` de `test.html`, pues le hemos merge a otro `commit` previo.

También podemos volver a redactar el primer `commit` y comprimir los demás:

```
$ git rebase -i HEAD~4

r c9d0d2a Added about us page
f b5ebb9c Added contact us page
f 7c963d9 Added categories page
f 6681986 Added products page
```

CHAPTER 4

GNU/Linux

- Linux Basics: SO, distros, arquitectura, kernel
- Comandos: administración de usuarios, ssh, scp
- Instalación de distros: Live USB, persistent USB
- Errores comunes

CHAPTER 5

SSH (Secure SHell)

5.1 Configuración de passwordless SSH login

Basado en: How to Setup Passwordless SSH Login

Note:

- [PC1] : SSH client
- [PC2] : SSH server

1. [PC1] Verificar si hay pares de llaves SSH existentes en el cliente:

```
$ ls -al ~/.ssh/id_*.pub
```

Si existe un par de llaves podemos saltarnos el siguiente paso o generar un nuevo par de llaves.

Si el comando no devuelve ningún resultado, significa que no existen llaves SSH en el cliente y podemos seguir con el siguiente paso para la generación del par de llaves.

2. [PC1] Generar un nuevo par de llaves SSH:

El siguiente comando generará un nuevo par de llaves SSH de 4096 bits para el usuario `user` del dominio `domain`:

```
$ ssh-keygen -t rsa -b 4096 -C "user@domain"
```

Por ejemplo:

```
$ ssh-keygen -t rsa -b 4096 -C "user1@localhost.localdomain"

Generating public/private rsa key pair.
Enter file in which to save the key (/home/user1/.ssh/id_rsa):
Created directory '/home/user1/.ssh'.
Enter passphrase (empty for no passphrase):
```

(continues on next page)

(continued from previous page)

```
Enter same passphrase again:
Your identification has been saved in /home/user1/.ssh/id_rsa.
Your public key has been saved in /home/user1/.ssh/id_rsa.pub.
The key fingerprint is:
SHA256:4v7tURAq6CMnhX3YI2yVLlensvMGMjFANEop5LVThAo user1@localhost.localdomain
The key s randomart image is:
+---[RSA 4096]----+
|.==.oo..  .      |
|Eoo*o=. ....     |
|+.o+O.=..o.      |
| . +=o+o.  .     |
|  o +=.oS   .    |
|   +oo+.   .     |
|     o.+  .      |
|     .  o. .     |
|      .o..o      |
+----[SHA256]-----+
```

Se eligió el nombre y ubicación por defecto. Se ha usado SSH sin ningún passphrase, ya que, a pesar de que da una capa extra de seguridad no permite hacer procesos automatizados.

Para asegurarnos que las llaves SSH se han generado podemos listar las llaves pública y privada con:

```
$ ls ~/.ssh/id_*
/home/user1/.ssh/id_rsa  /home/user1/.ssh/id_rsa.pub
```

3. [PC1] Copiar la llave pública al servidor PC2.

Una vez que hemos generado el par de llaves SSH, para iniciar sesión en el servidor sin ninguna contraseña debemos copiar la llave pública al servidor que queremos manejar.

La forma más sencilla de copiar la llave pública al servidor es usando el comando `ssh-copy-id`.

```
$ ssh-copy-id remote_username@server_ip_address
```

Por ejemplo:

```
$ ssh-copy-id user1@192.168.122.158

/bin/ssh-copy-id: INFO: Source of key(s) to be installed: "/home/user1/.ssh/id_rsa.pub
↪"
The authenticity of host '192.168.122.158 (192.168.122.158)' can t be established.
ECDSA key fingerprint is SHA256:OxiFa4EyYuNC6L0+vnMVZ59XYLKek7bAg91jhrbHVoc.
ECDSA key fingerprint is MD5:2d:48:e3:03:18:49:0c:fd:3a:7b:ca:6b:65:b8:66:4e.
Are you sure you want to continue connecting (yes/no)? yes
/bin/ssh-copy-id: INFO: attempting to log in with the new key(s), to filter out any␣
↪that are already installed
/bin/ssh-copy-id: INFO: 1 key(s) remain to be installed -- if you are prompted now it␣
↪is to install the new keys
user1@192.168.122.158 s password:

Number of key(s) added: 1

Now try logging into the machine, with:   "ssh 'user1@192.168.122.158'"
and check to make sure that only the key(s) you wanted were added.
```

Se pidió ingresar la contraseña del usuario remoto al que nos intentamos conectar.

Una vez que el usuario se haya autenticado, la llave pública será añadida al archivo del usuario remoto `authorized_keys` y la conexión se cerrará.

Si por algún motivo la utilidad `ssh-copy-id` no está disponible en nuestra máquina local podemos copiar manualmente la llave pública con:

```
$ cat ~/.ssh/id_rsa.pub | ssh remote_username@server_ip_address "mkdir -p ~/.ssh &&␣
↪chmod 700 ~/.ssh && cat >> ~/.ssh/authorized_keys && chmod 600 ~/.ssh/authorized_
↪keys"
```

4. [PC1] Conectarnos a nuestro servidor usando las llaves SSH.

La conexión SSH debería dejarnos conectar al servidor remoto sin pedirnos ninguna contraseña.

```
$ ssh remote_username@server_ip_address
```

Por ejemplo:

```
$ ssh user1@192.168.122.158
```

CHAPTER 6

User Management

6.1 Creación de un usuario

Table of Contents

6.1.1 Creando el usuario

Forma 1 - usando `adduser`

Note: DEBIAN/UBUNTU

```
root@ubuntudskt1:~'#' adduser user1
```

(continues on next page)

(continued from previous page)

```
Añadiendo el usuario user1 ...
Añadiendo el nuevo grupo user1 (1001) ...
Añadiendo el nuevo usuario user1 1001 con grupo user1 ...
Creando el directorio personal /home/user1 ...
Copiando los ficheros desde /etc/skel ...

Introduzca la nueva contraseña de UNIX:
Vuelva a escribir la nueva contraseña de UNIX:
passwd: contraseña actualizada correctamente

Cambiando la información de usuario para user1
Introduzca el nuevo valor, o presione INTRO para el predeterminado
    Nombre completo []:
    Número de habitación []:
    Teléfono del trabajo []:
    Teléfono de casa []:
    Otro []:
¿Es correcta la información? [S/n] S
```

Note: RHEL/CENTOS

```
[root@localhost ~]'#' adduser -md /home/user1 -s /bin/bash user1
```

```
[root@localhost ~]'#' passwd user1

Changing password for user user1.
New password:
Retype new password:
passwd: all authentication tokens updated successfully.
```

Forma 2 - usando `useradd`

Note: DEBIAN/UBUNTU

```
root@ubuntudskt1:~'#' useradd -md /home/user1 -s /bin/bash user1
```

```
root@ubuntudskt1:~'#' passwd user1

Introduzca la nueva contraseña de UNIX:
Vuelva a escribir la nueva contraseña de UNIX:
passwd: contraseña actualizada correctamente
```

Note: RHEL/CENTOS

```
[root@localhost ~]'#' useradd -md /home/user1 -s /bin/bash user1
```

```
[root@localhost ~]'#' passwd user1

Changing password for user user1.
New password:
Retype new password:
passwd: all authentication tokens updated successfully.
```

6.1.2 Dando permisos de `sudo` al usuario

Note: DEBIAN/UBUNTU

```
root@ubuntudskt1:~'#' usermod -aG sudo user1
```

Note: RHEL/CENTOS

```
[root@localhost ~]'#' usermod -aG wheel user1
```

6.1.3 Habilitando passwordless `sudo` al usuario

Note: DEBIAN/UBUNTU = RHEL/CENTOS

Forma 1 - creando un archivo en el directorio `/etc/sudoers.d`

```
$ echo "user1 ALL = (root) NOPASSWD:ALL" | sudo tee /etc/sudoers.d/user1

[sudo] contraseña para user1:
user1 ALL = (root) NOPASSWD:ALL
```

```
$ sudo chmod 0440 /etc/sudoers.d/user1
```

Forma 2 - agregando una entrada archivo `/etc/sudoers` con `visudo`

```
$ sudo visudo -f /etc/sudoers
[sudo] contraseña para user1:
```

```
[...]
#includedir /etc/sudoers.d
user1 ALL=(ALL) NOPASSWD:ALL
```

6.1.4 Links útiles

- How To Create a Sudo User on Ubuntu

- How To Create a Sudo User on CentOS

CHAPTER 7

Creando una imagen de disco con `dd`

`dd` es una herramienta para convertir y copiar archivos.

En SOs Unix-like, drivers de dispositivos para hardware (como discos duros) y archivos de dispositivos especiales (como `/dev/zero` o `/dev/random`) aparecen en el filesystem como archivos normales. `dd` puede leer y/o escribir desde/en esos archivos, siempre que la función se implemente en su driver respectivo.

Gracias a este funcionamiento, `dd` puede usarse para tareas como respaldar del sector de arranque de un disco duro, obtener una cantidad fija de datos aleatorios. También se puede realizar conversiones de datos según son copiados.

Por defecto, `dd` lee de un archivo input file (`if`) y escribe en un output file (`of`). Un **bloque (block)** es la unidad de medida de bytes que son leídos y escritos o convertidos de una sola vez; para especificar el tamaño del bloque se usa la opción `bs`.

En este caso se usará `dd` para crear una imagen de disco vacía desde el directorio `/dev/zero`:

- Creando un disco preallocated:

```
$ dd if=/dev/zero of=disk1.img bs=1G count=10

0+0 records in
10+0 records out
10737418240 bytes (11 GB, 10 GiB) copied, 27,016 s, 397 MB/s

$ qemu-img info disk1.img
image: disk1.img
file format: raw
virtual size: 10G (10737418240 bytes)
disk size: 10G
```

- Creando un disco thin-provisioned:

Con la opción `seek` logramos que la imagen de disco creada sea thin-provisioned.

```
$ dd if=/dev/zero of=disk2.img bs=1G seek=10 count=0

0+0 records in
```

(continues on next page)

(continued from previous page)

```
0+0 records out
0 bytes copied, 0,000124988 s, 0,0 kB/s

$ qemu-img info disk2.img
image: disk2.img
file format: raw
virtual size: 10G (10737418240 bytes)
disk size: 0
```

7.1 Referencias

- dd (Unix) - Wikipedia

CHAPTER 8

ArchLinux

8.1 Instalando Arch

Table of Contents

El siguiente procedimiento se basa en la guía de instalación oficial de la Wiki de Arch Linux. Los pasos describen **cómo instalar el sistema operativo (SO) Arch Linux desde un sistema live que haya arrancado con la imagen de instalación oficial**.

Los requerimientos necesarios en el sistema donde instalaremos Arch Linux es que debe ser compatible con la arquitectura x86_64, debe contar con un mínimo de 512 MiB de RAM, deben haber 800 MiB libres de espacio en el disco (lo que ocupa aproximadamente una instalación básica de Arch Linux) y conectividad a Internet para obtener paquetes de repositorios externos.

Para obtener una imagen .iso de Arch Linux podemos descargar un torrent desde la página de descargas de Arch Linux.

8.1.1 Creando y configurando el sistema

En este caso se usará una máquina virtual (VM) para la instalación de Arch Linux, simulando que tenemos un disco físico con el SO Arch Linux dentro de él. Además, simularemos que tenemos un disco físico vacío donde instalaremos nuestro SO. El concepto del esquema físico simulado que pensamos es:

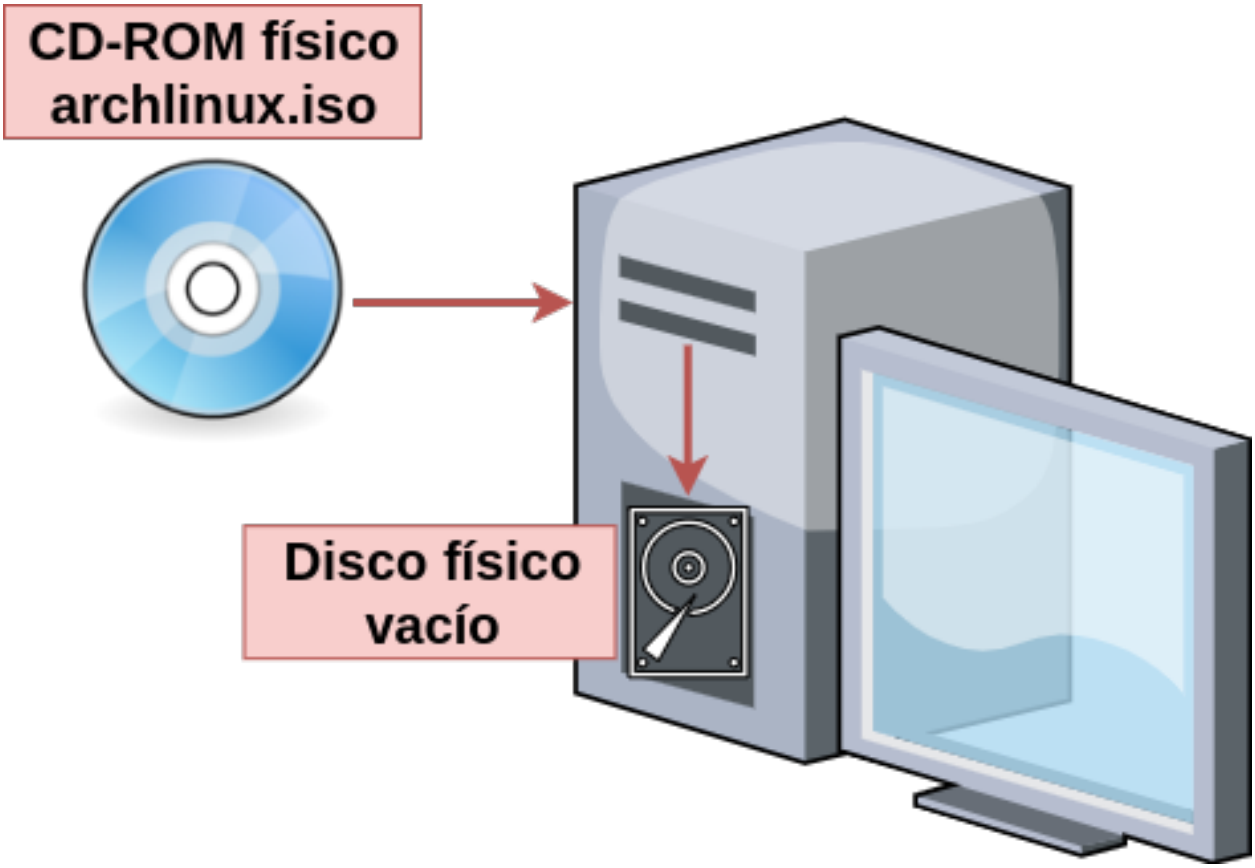


Fig. 1: Esquema de instalación de SO con sistema físico

El esquema real que usaremos para la instalación será con una máquina virtual y la imagen .iso de Arch Linux:

Creando la VM en VirtualBox

El procedimiento de creación de la VM será usando VirtualBox:

1. Crear una nueva VM en VirtualBox presionando el botón *New*:
2. Escribir un nombre en *Name* y una ubicación para la VM:
3. Asignar la cantidad de memoria RAM deseada a la VM:
4. Crear un nuevo disco virtual, seleccionar la opción *Create a virtual hard disk now* :
5. Seleccionar el tipo de disco duro como *VDI (VirtualBox Disk Image)*, formato nativo de VirtualBox:
6. Seleccionar el tipo de asignación que se le dará al disco como *Dynamically allocated*:
7. Asignar un tamaño virtual al disco:
8. Clic en el botón *Create*.

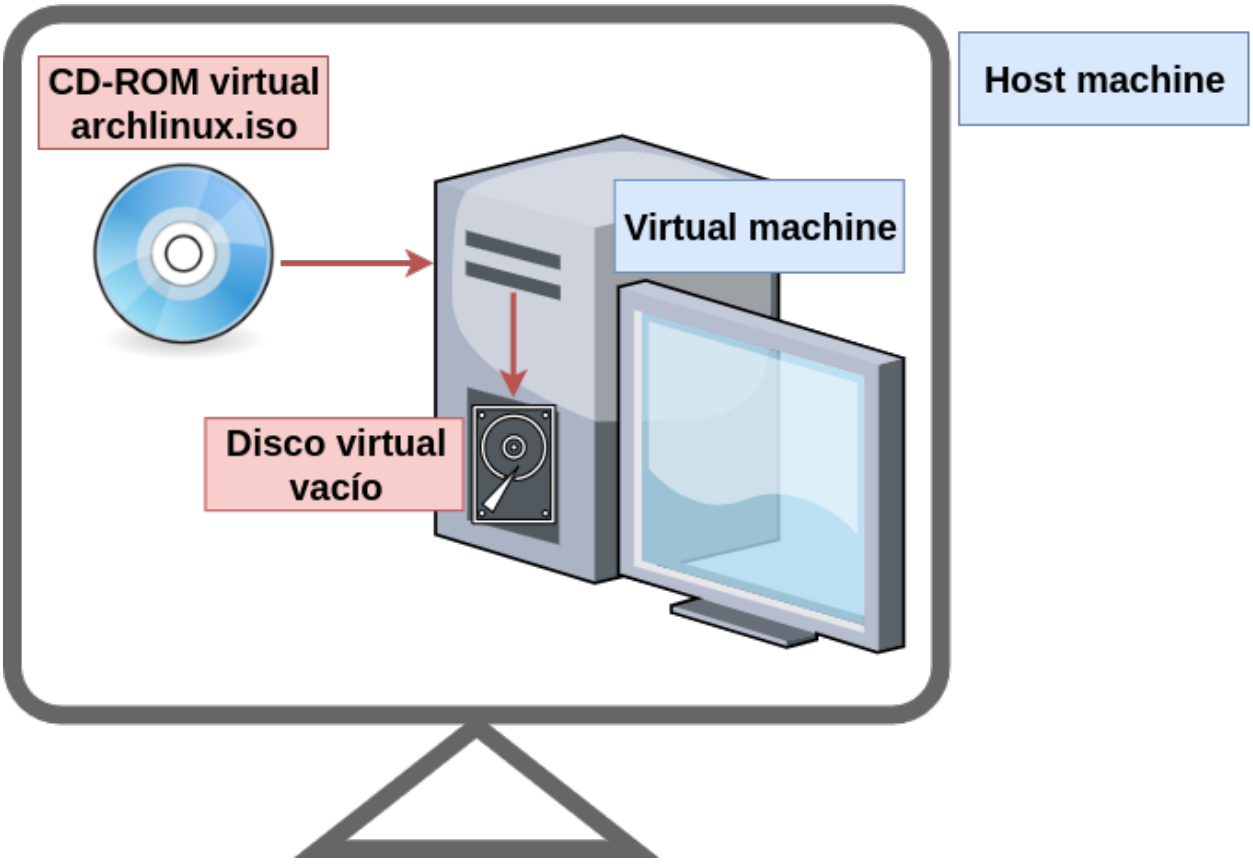


Fig. 2: Esquema de instalación de SO con sistema virtual

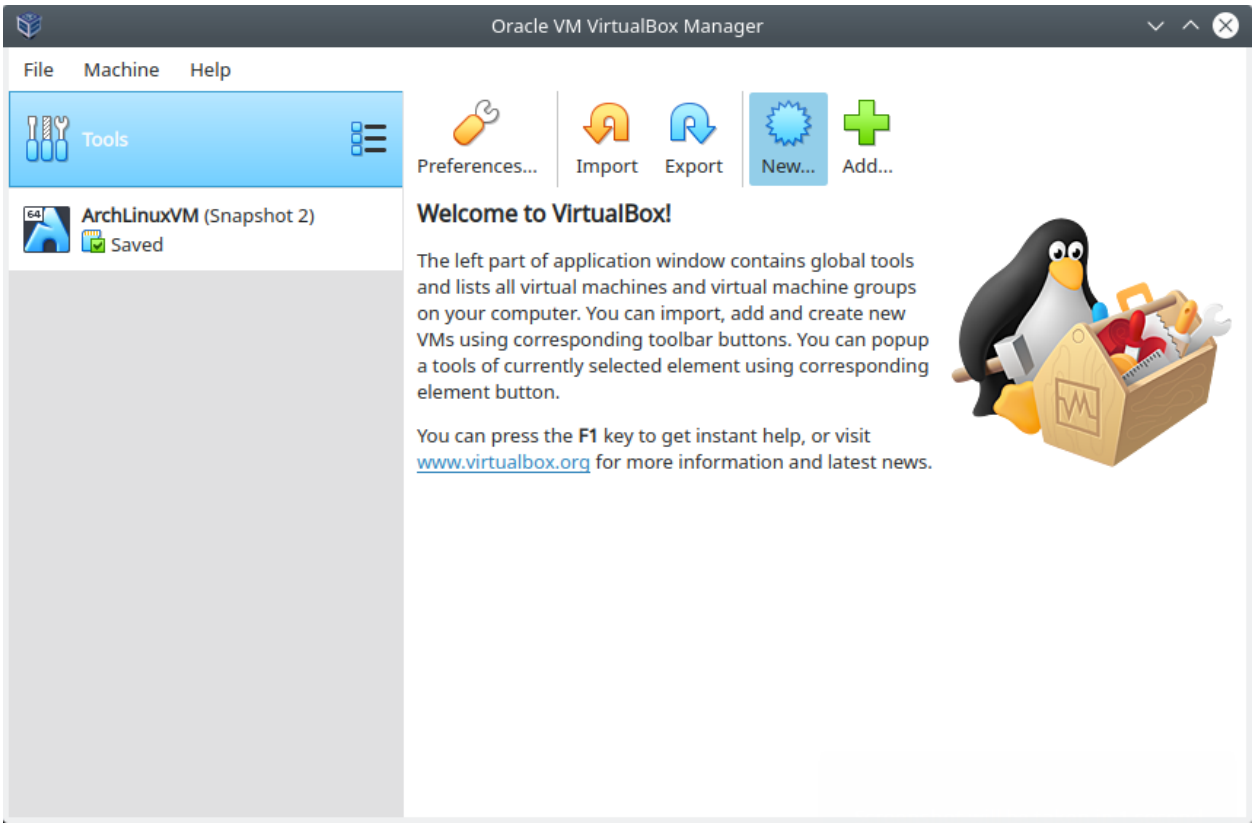


Fig. 3: Creando la VM en VirtualBox - Botón *New*

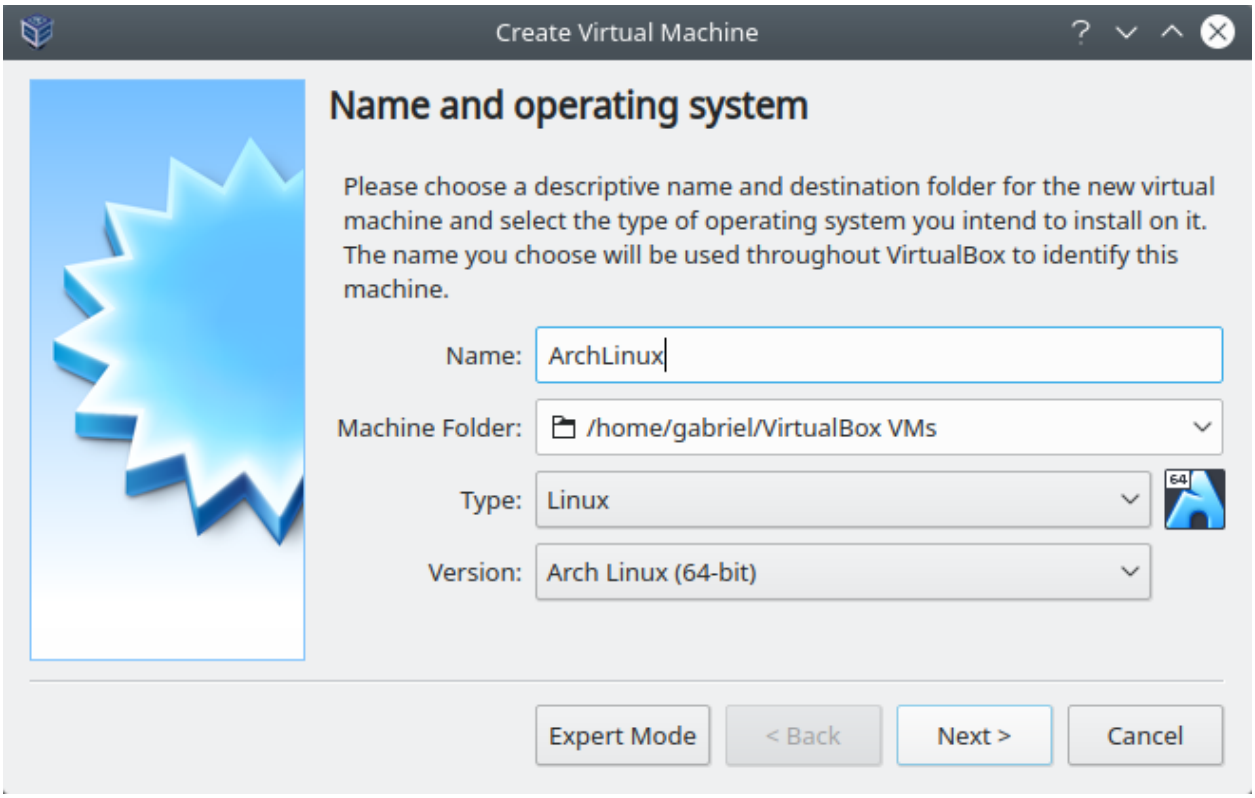


Fig. 4: Creando la VM en VirtualBox - Name and operating system

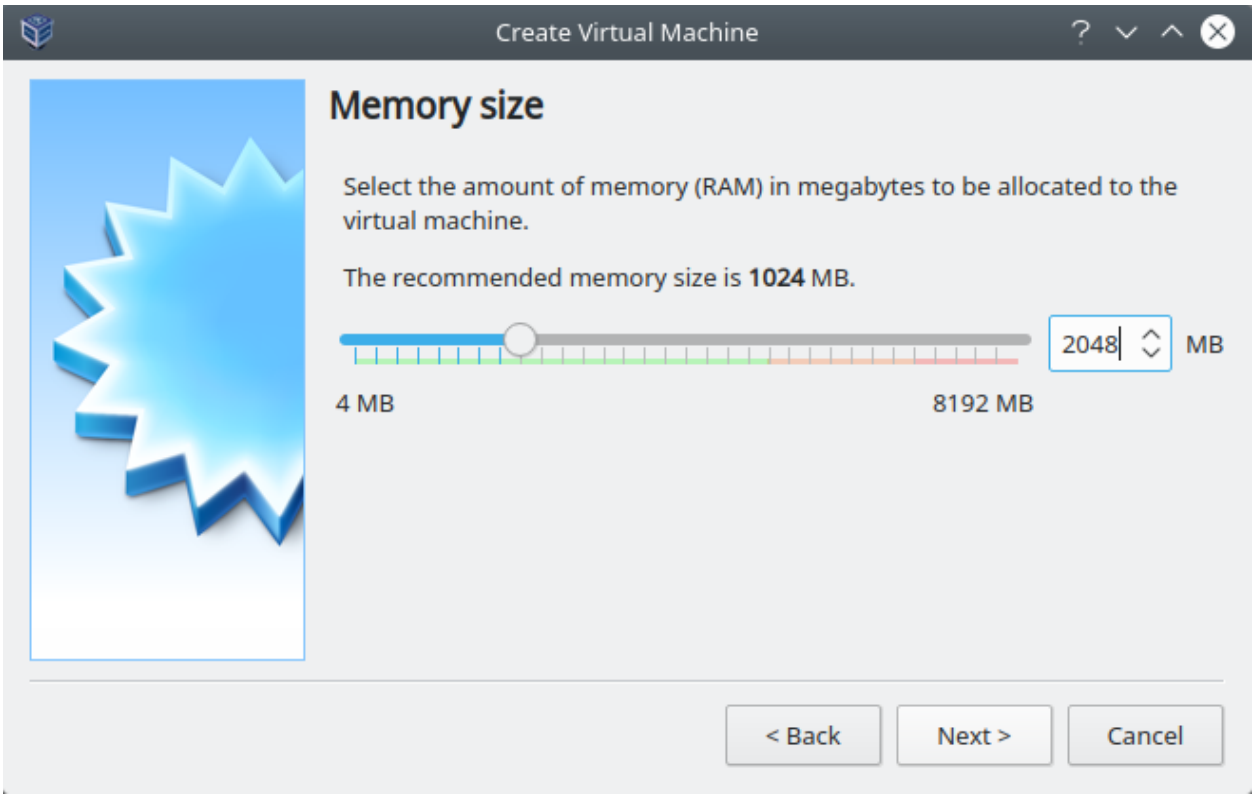


Fig. 5: Creando la VM en VirtualBox - Memory size

Fig. 6: Creando la VM en VirtualBox - Hard disk

8.1.2 Configurando la VM

Luego de haber creado la VM, la veremos listada en la ventana principal de VirtualBox. Ahora deberemos configurarla:

1. Seleccionar la VM recién creada y hacer clic en el botón *Settings*:
2. En la nueva ventana, clic en la sección *System*, pestaña *Processor*. Seleccionar la cantidad de CPUs que deseamos asignar a nuestra VM:
3. Clic en la sección *Display*, pestaña *Screen*. En cantidad de *Video Memory* asignar 128 MB y elegir `VBoxVGA` como *Graphics Controller*:
4. Para tener conexión a Internet desde nuestra VM, clic en la sección *Network*, pestaña *Adapter1*. Seleccionar `NAT` como la red a la cual conectaremos la VM:
5. Clic en la sección *Storage*. Seleccionar el ícono de un disco con un + (*Adds optical drive*):
6. En la ventana emergente seleccionar el botón *Choose disk*:
7. En la nueva ventana seleccionar el botón *Add...* (*Add Disk Image*)
8. Buscar la imagen .iso del SO en el navegador de archivos y clic en *Open*:
9. En la ventana previa ahora aparecerá cargada la imagen .iso. Clic en *Choose*:
10. Verificar que la imagen .iso se haya cargado en la parte de dispositivos de almacenamiento. Clic en *OK*:
11. Finalmente, seleccionar la VM y clic en el botón *Start*:

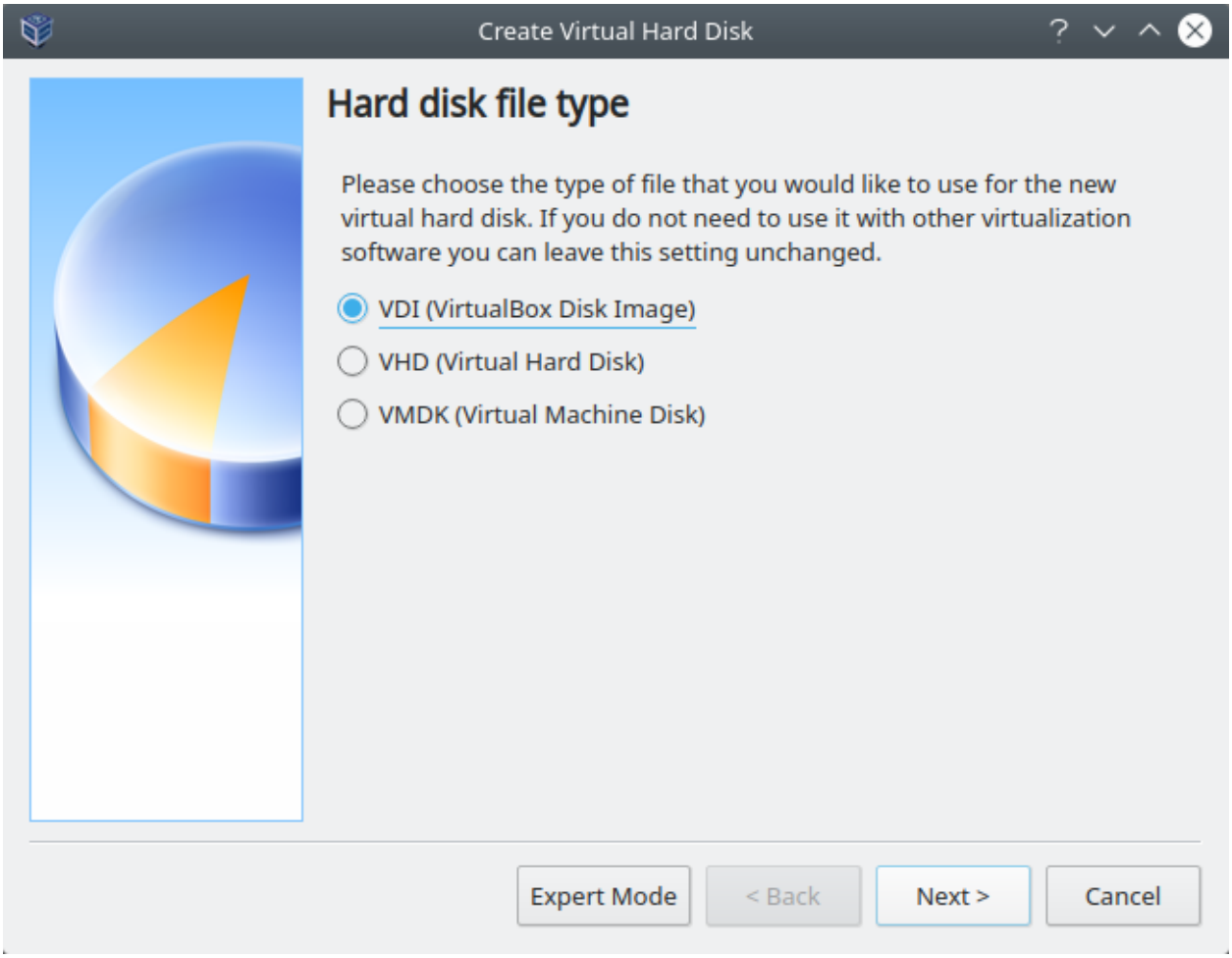


Fig. 7: Creando la VM en VirtualBox - Hard disk file type

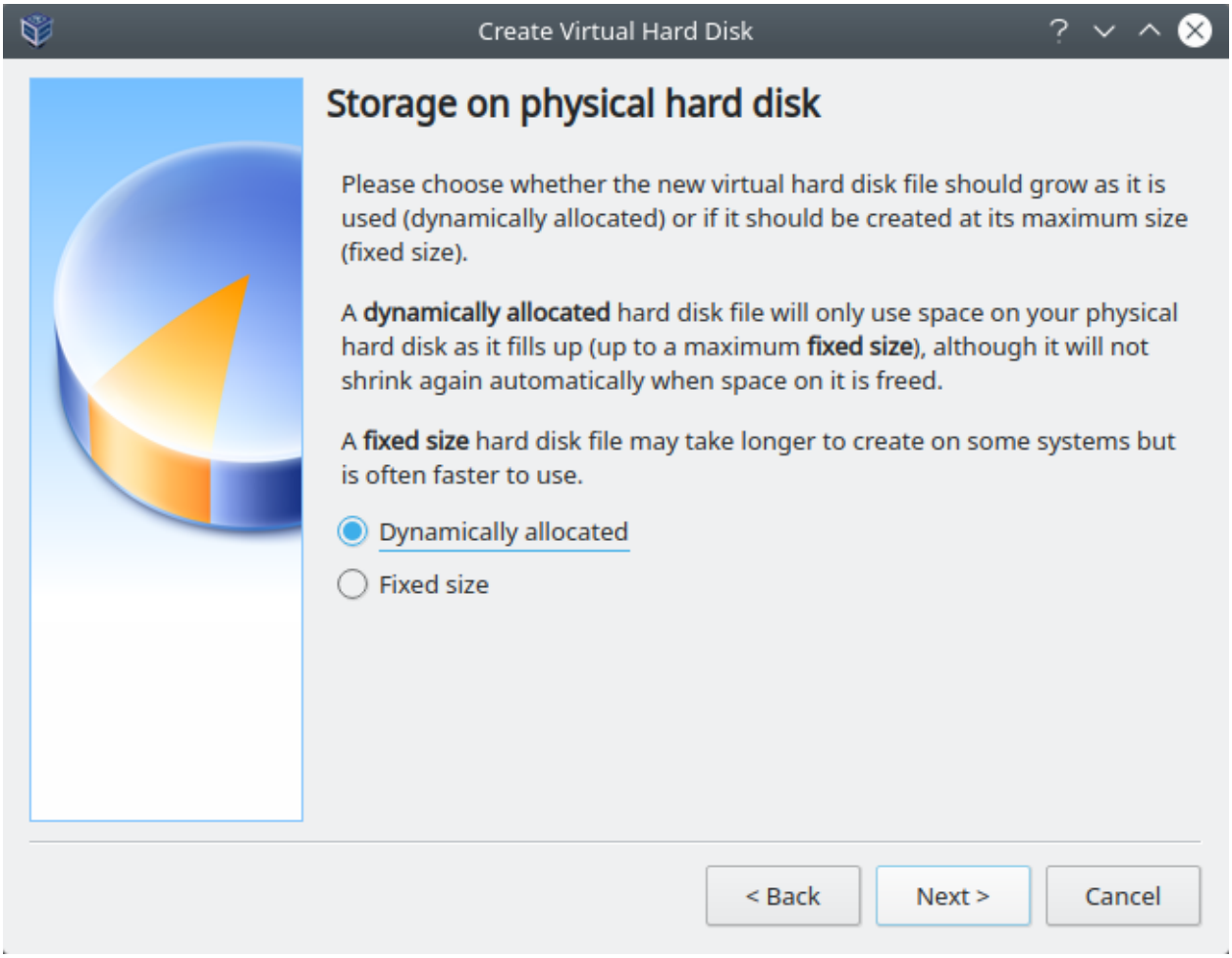


Fig. 8: Creando la VM en VirtualBox - Storage on physical hard disk

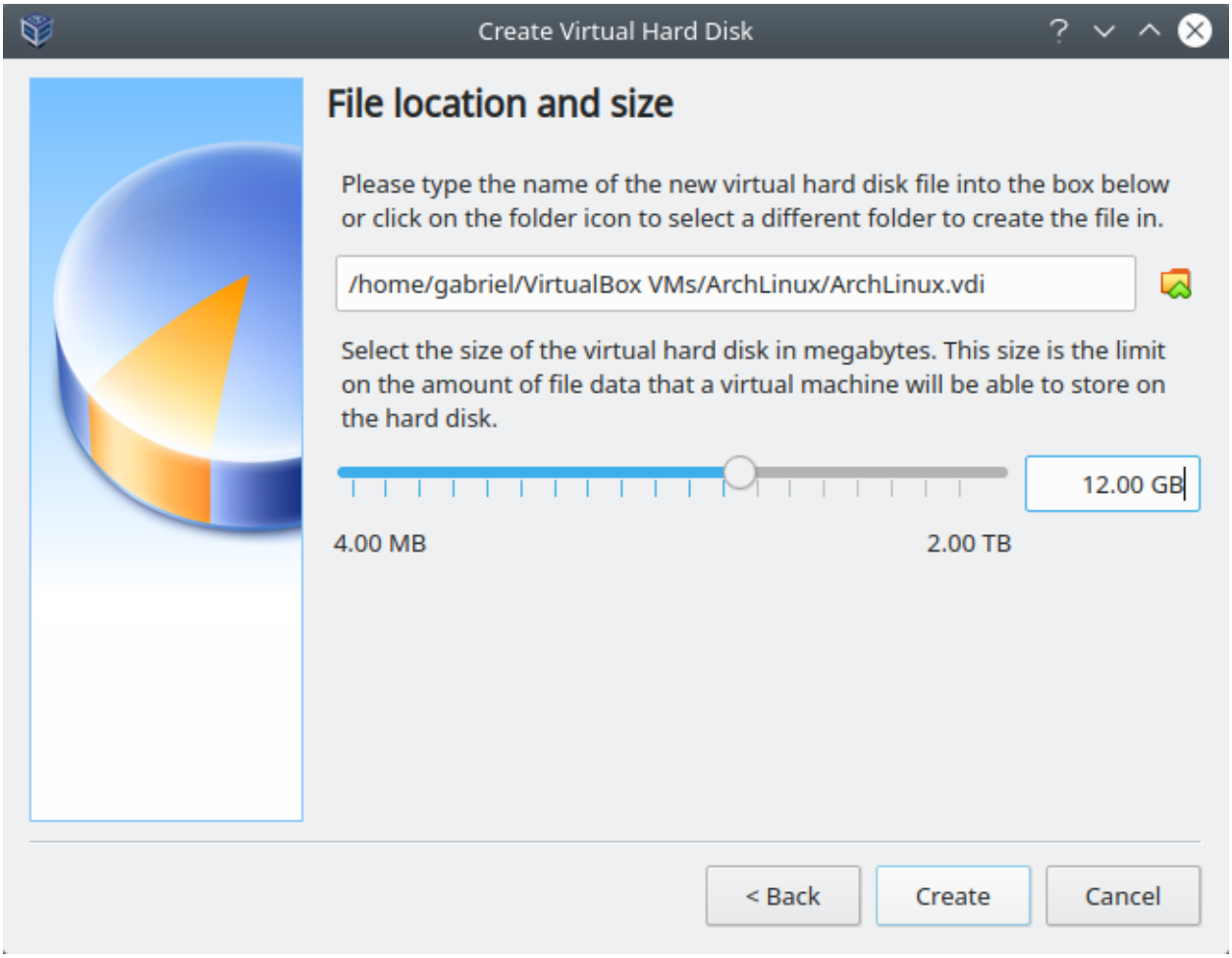


Fig. 9: Creando la VM en VirtualBox - File location and size

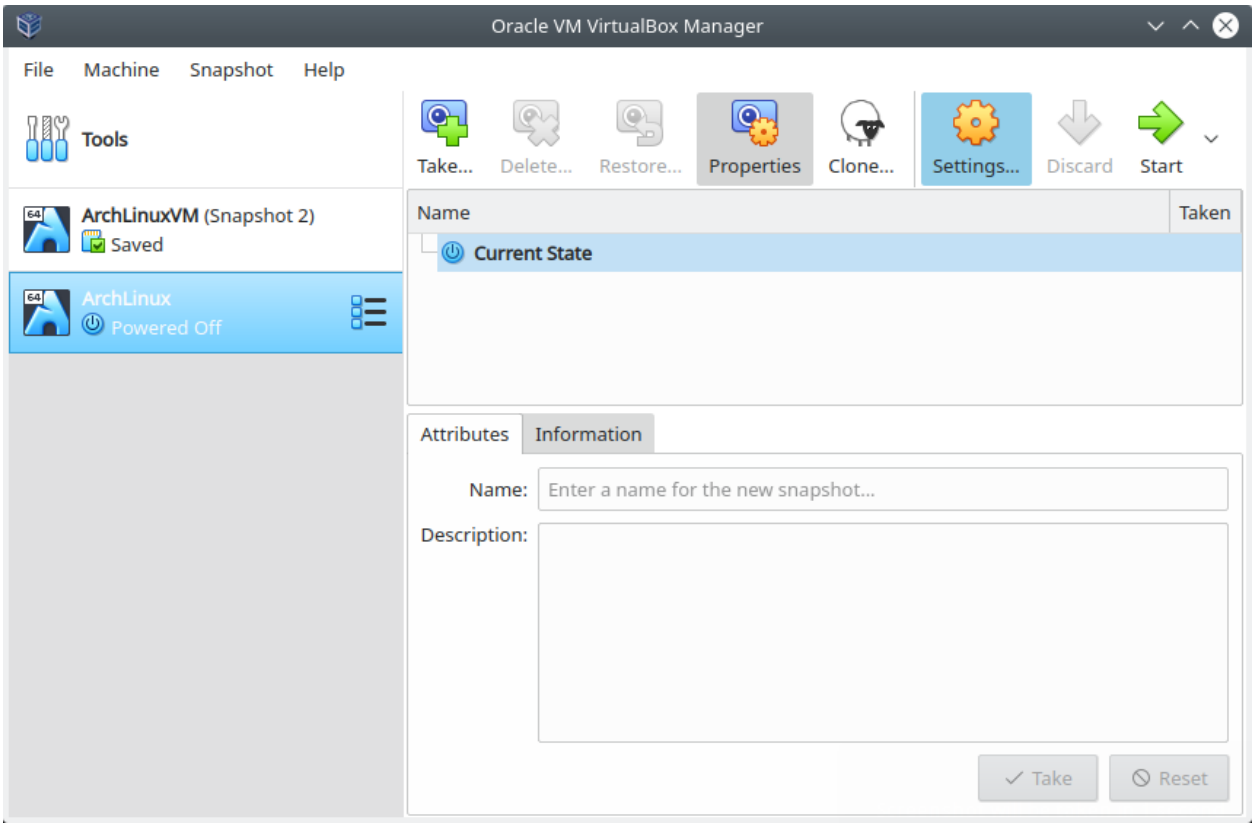


Fig. 10: Configurando la VM en VirtualBox - Botón *Settings*

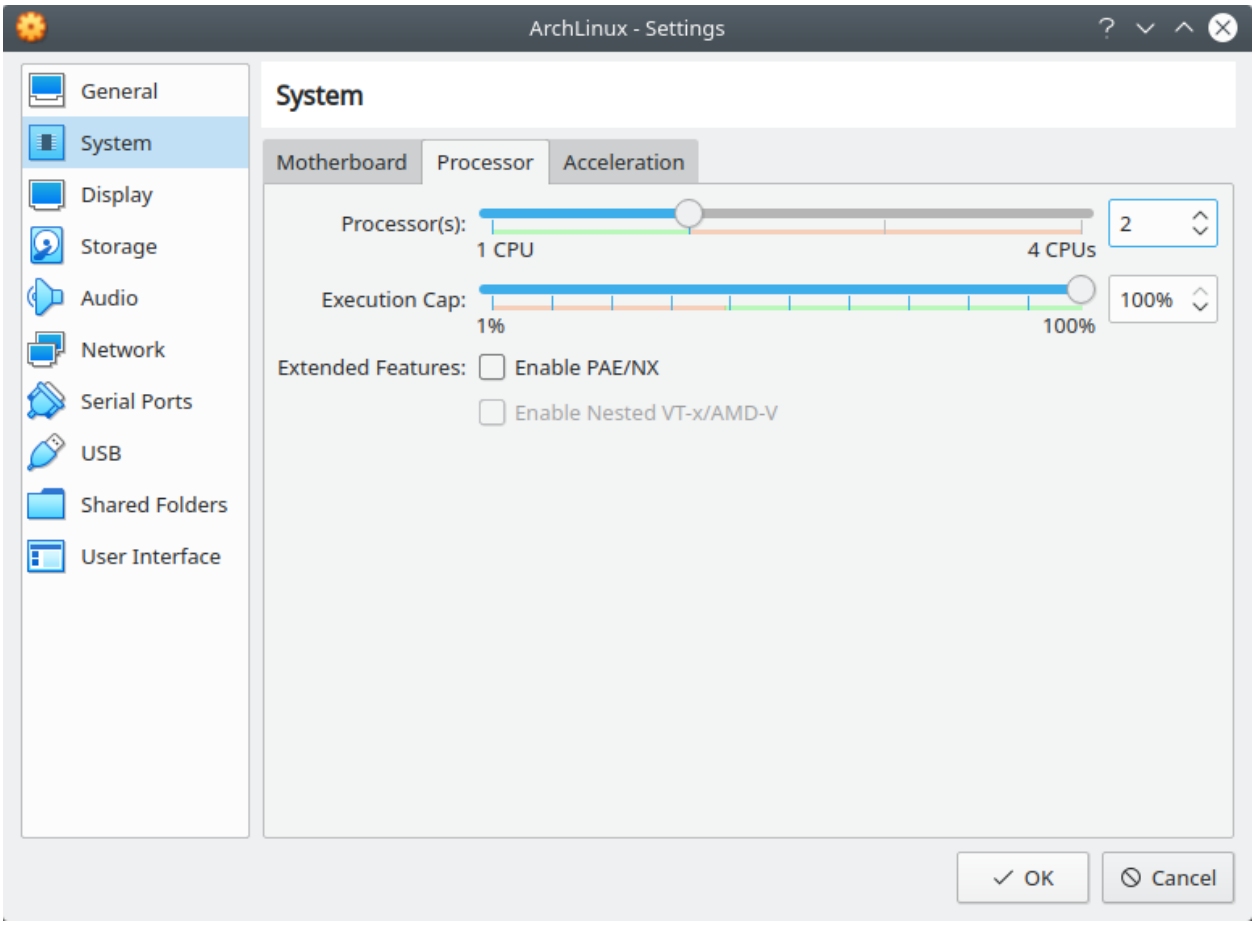


Fig. 11: Configurando la VM en VirtualBox - sección *System*, pestaña *Processor*

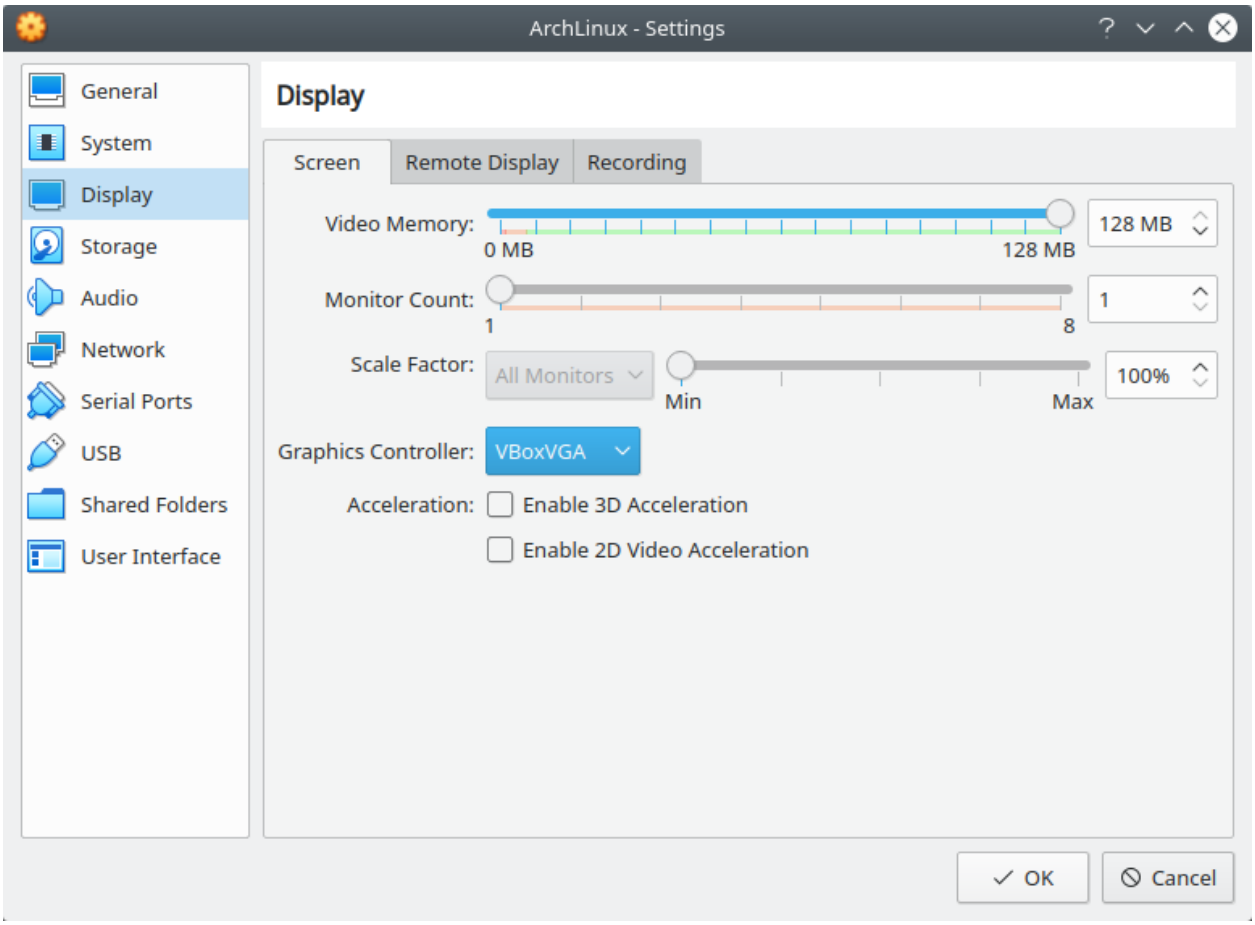


Fig. 12: Configurando la VM en VirtualBox - sección *Display*, pestaña *Screen*

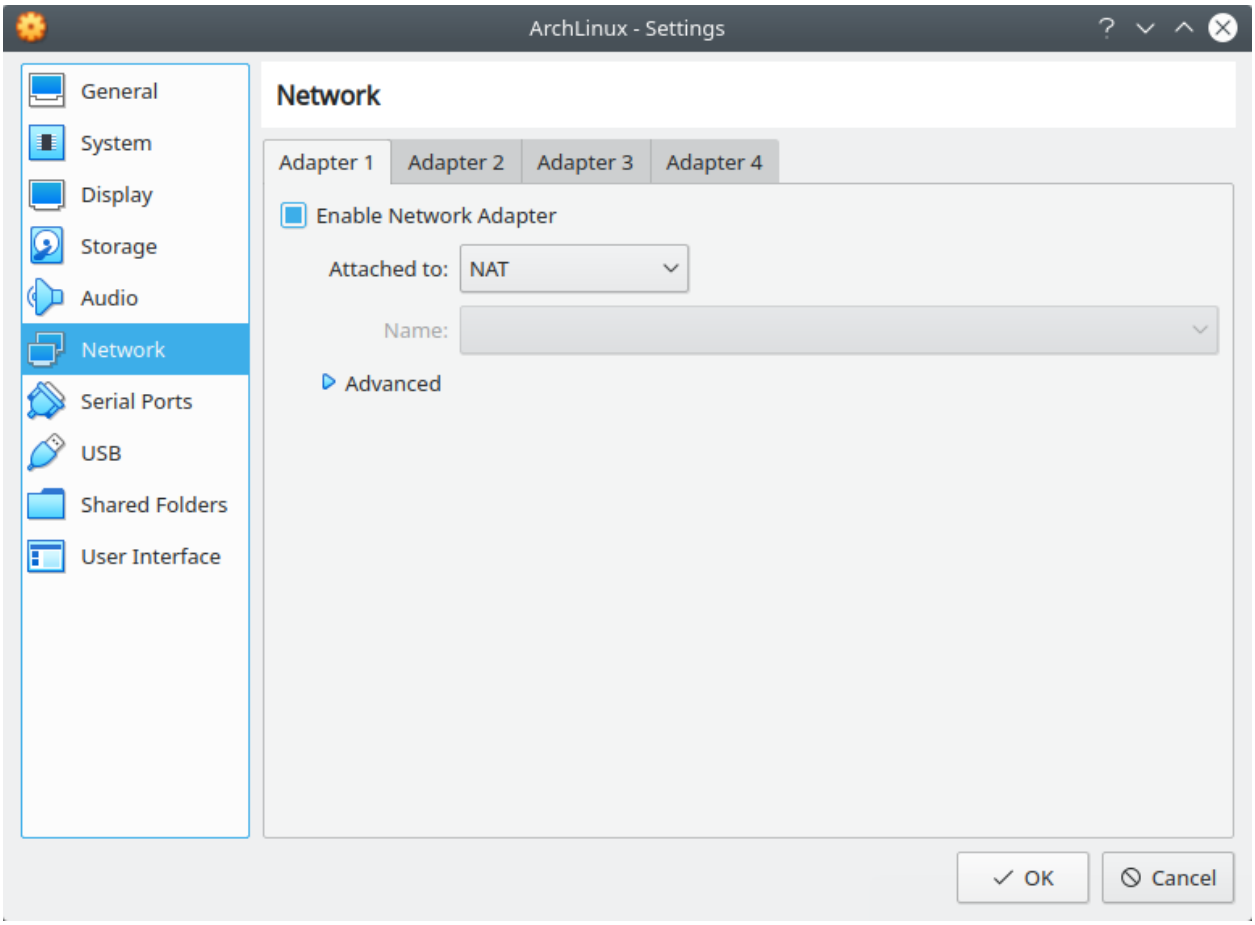


Fig. 13: Configurando la VM en VirtualBox - sección *Network*, pestaña *Adapter1*

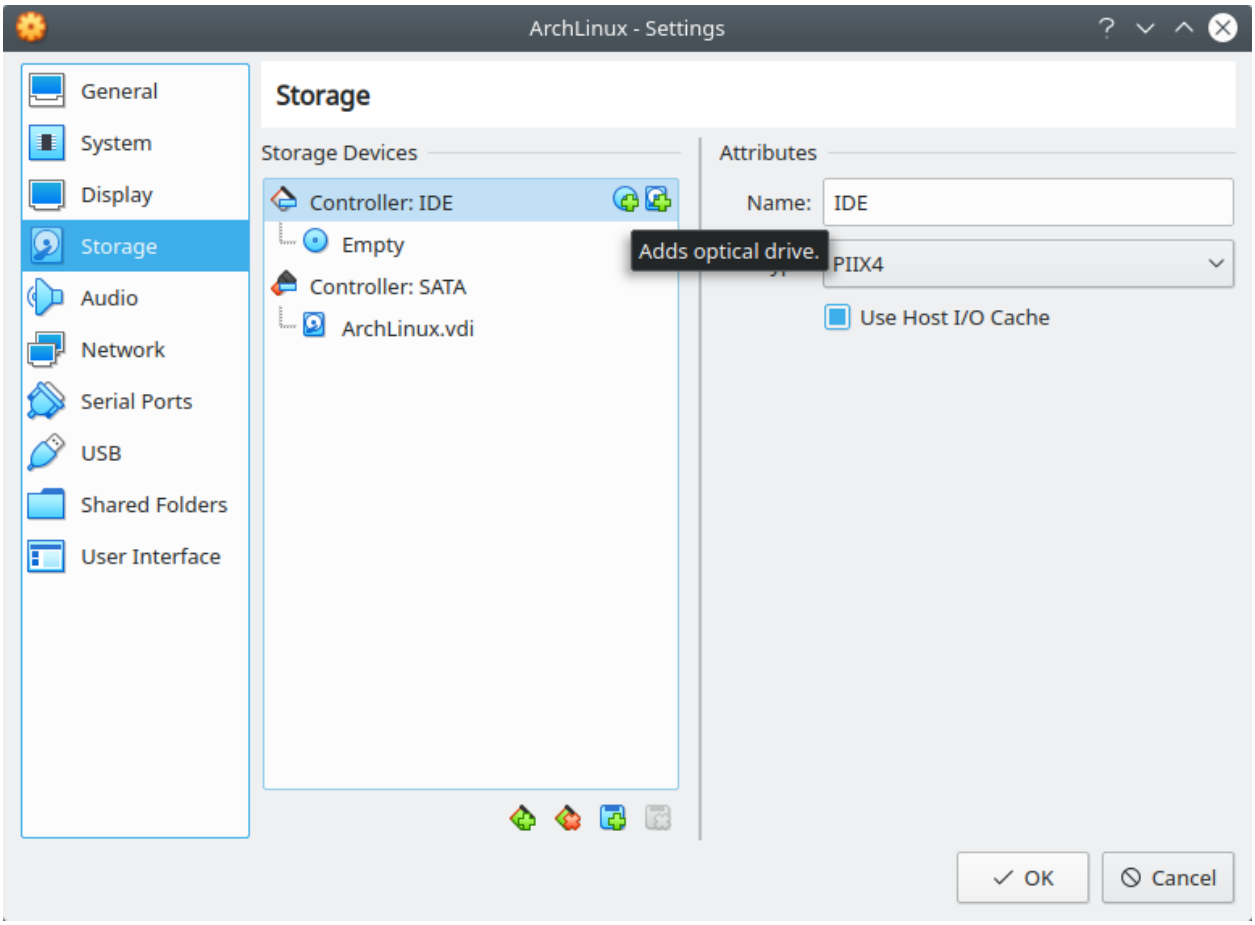


Fig. 14: Configurando la VM en VirtualBox - *Adds optical drive*

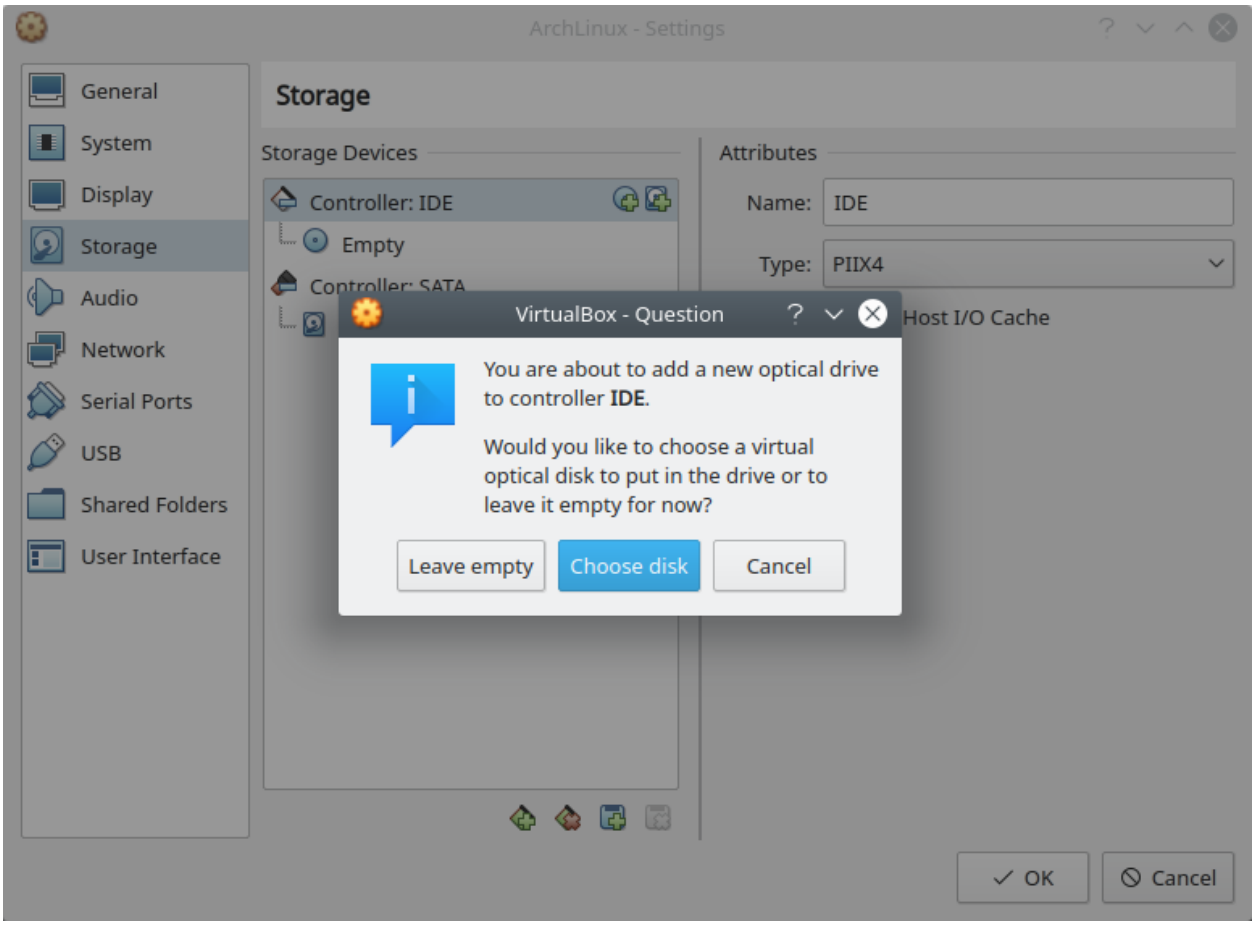


Fig. 15: Configurando la VM en VirtualBox - botón *Choose disk*:

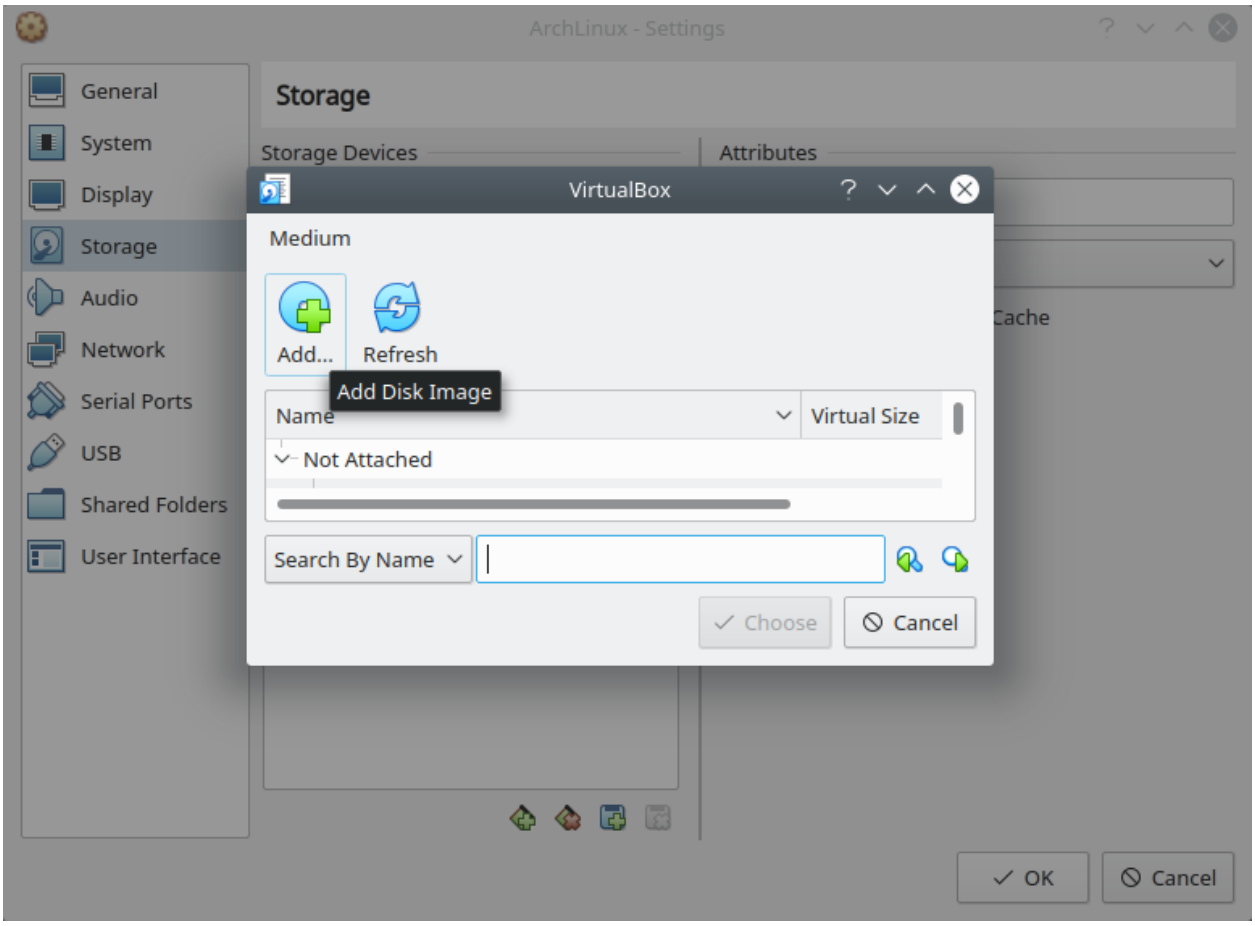


Fig. 16: Configurando la VM en VirtualBox - botón *Add...*:

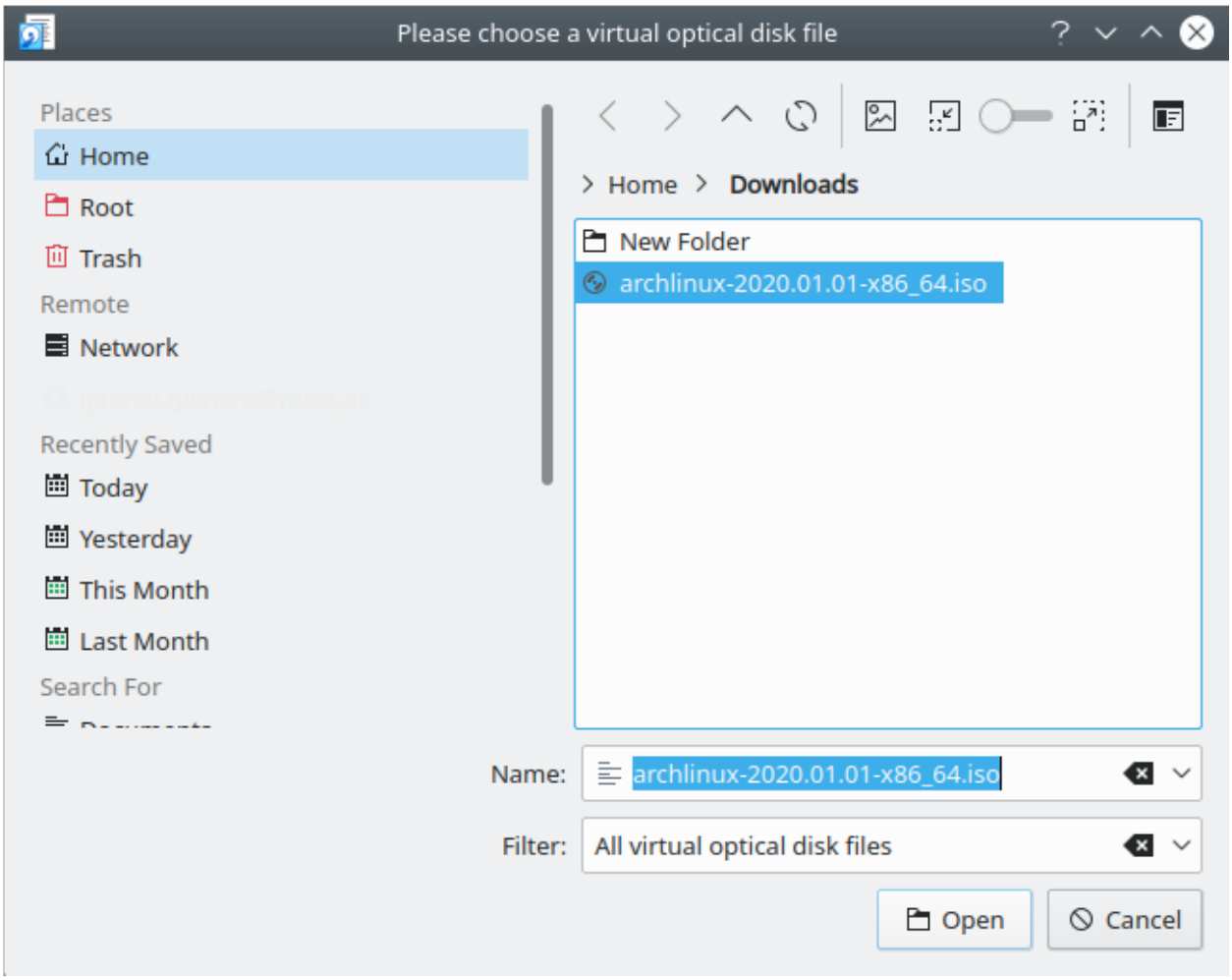


Fig. 17: Configurando la VM en VirtualBox - seleccionar la imagen .iso

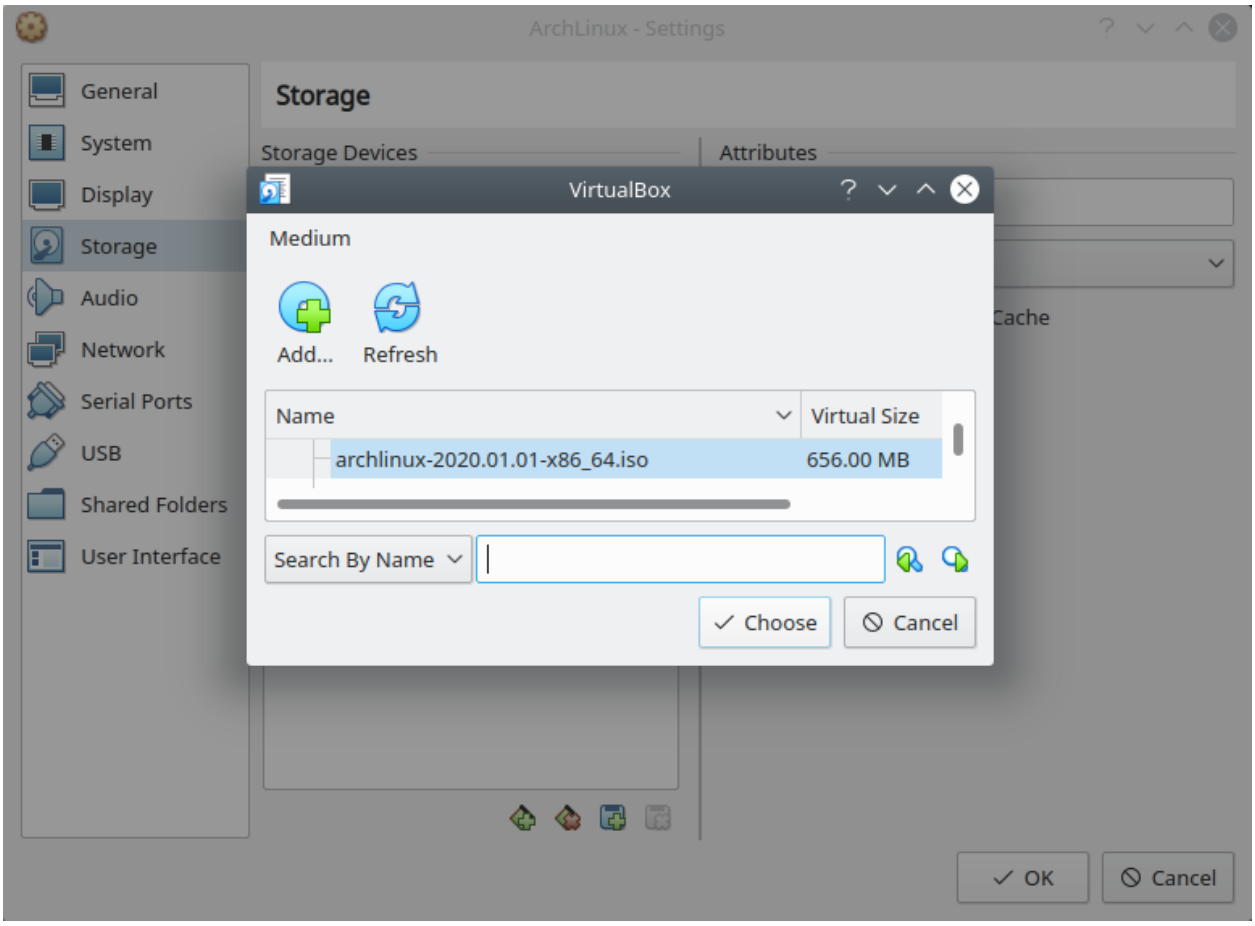


Fig. 18: Configurando la VM en VirtualBox - seleccionar la imagen .iso

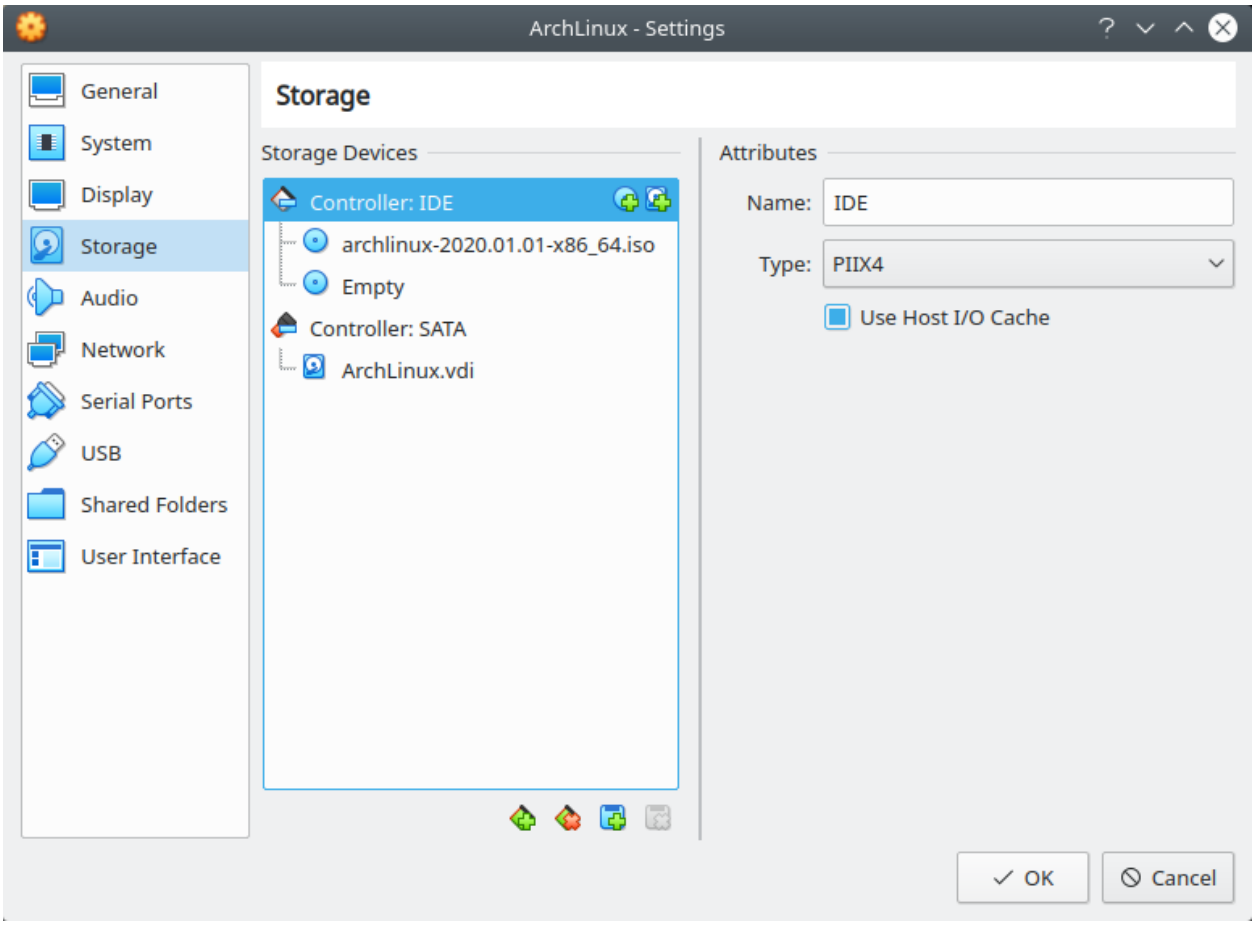


Fig. 19: Configurando la VM en VirtualBox

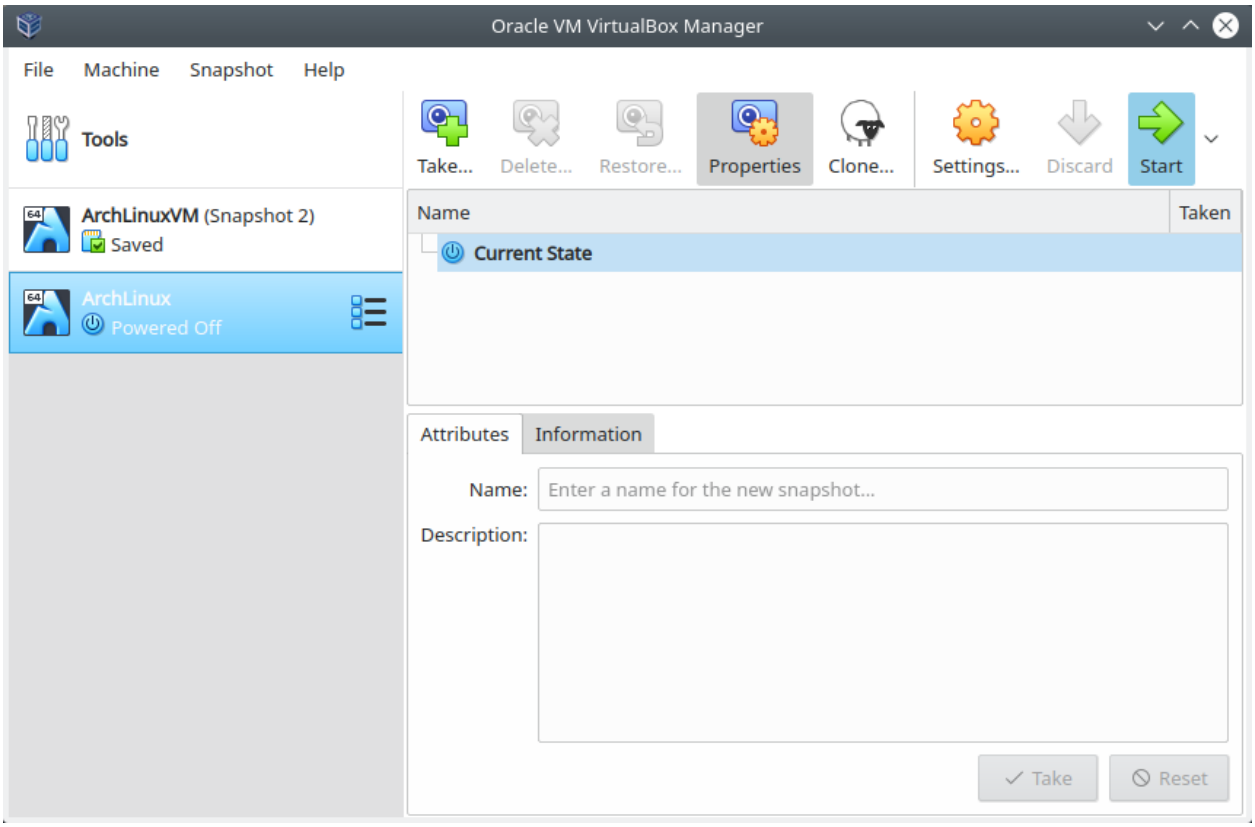


Fig. 20: Configurando la VM en VirtualBox - botón *Start*

8.1.3 Proceso de instalación de Arch Linux

Al iniciar nuestro sistema desde la imagen de Arch Linux veremos la pantalla de arranque de Arch Linux, seleccionar la opción *Boot Arch Linux (x86_64)*:

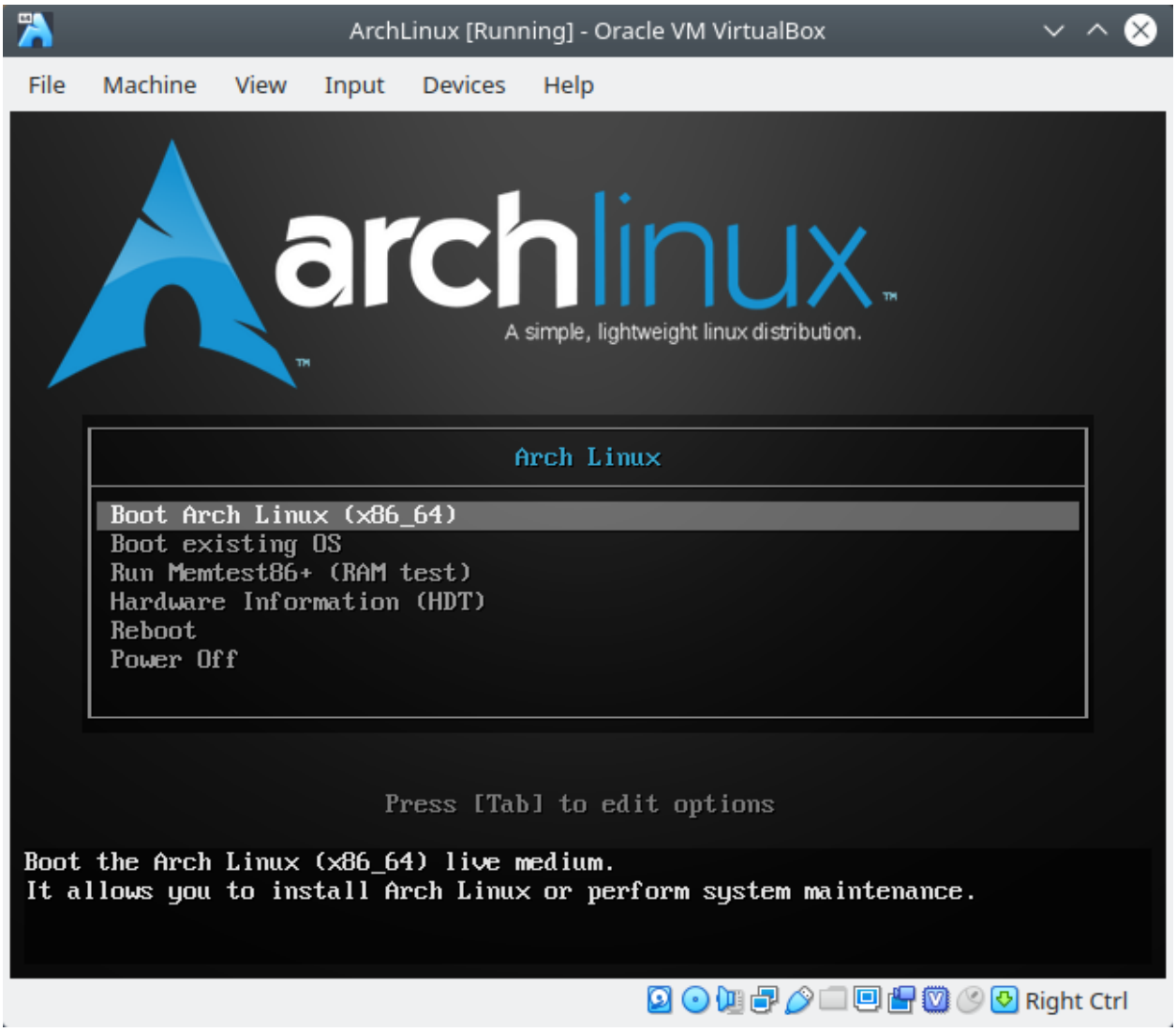


Fig. 21: Pantalla de arranque de la imagen de Arch Linux - opción *Boot Arch Linux (x86_64)*

Cuando haya terminado de cargar el arranque de la imagen iniciaremos sesión automáticamente como usuarios `root` bajo el prompt:

```
root@archiso ~ '#'
```

A partir de esto momento seguiremos los pasos indicados en la guía de instalación oficial de la Wiki de Arch Linux

Pre-instalación

Note: Para realizar los pasos de la guía remotamente desde el terminal del host realizar lo siguiente:

1. En VirtualBox, sin apagar la VM, cambiar el tipo de red a la que está conectada el Adadptador 1 de la VM: de `NAT` a `Bridge Adapter`:

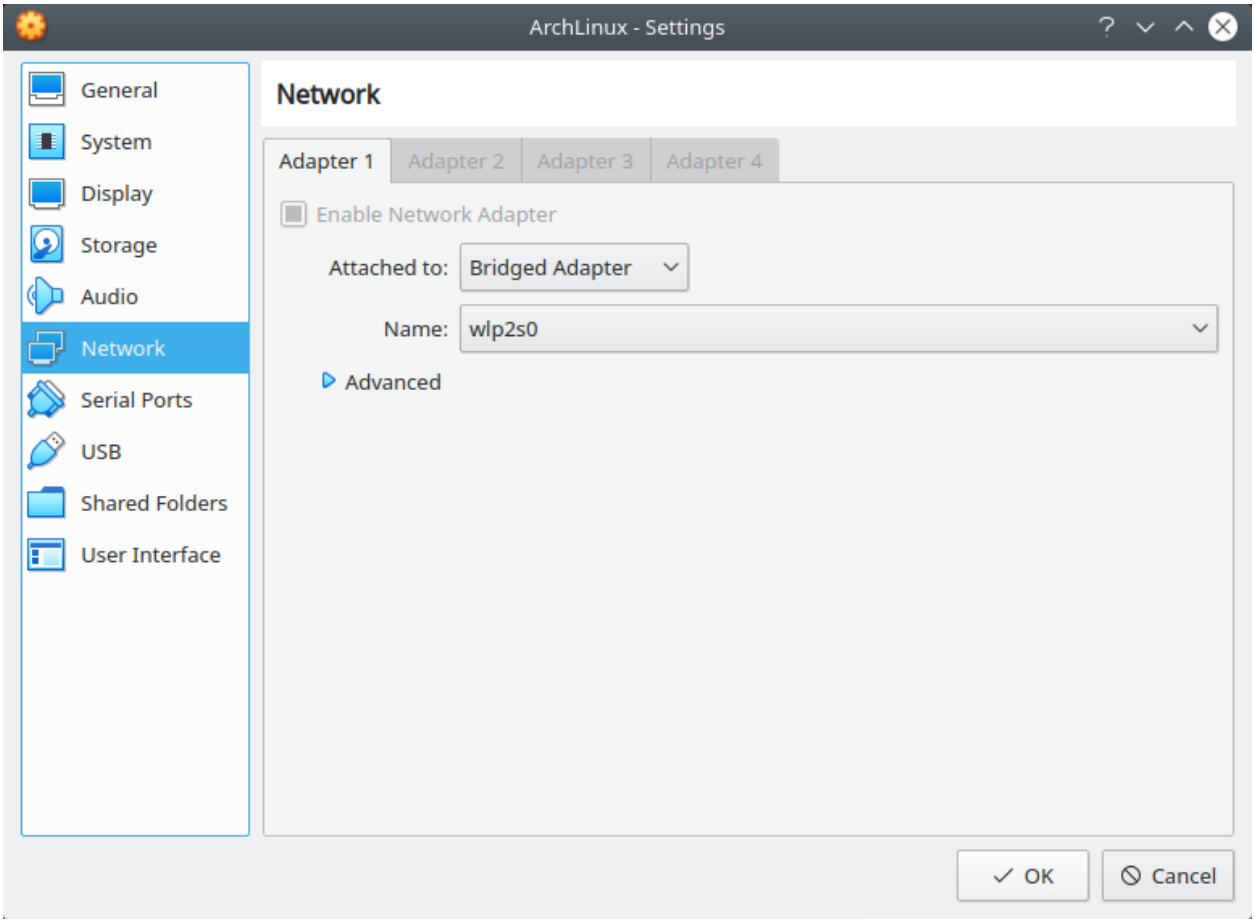


Fig. 22: VM conectada por Bridge Adapter

2. En la VM, pedir configuración de red por DHCP:

```
'#' dhclient -v enp0s3
```

3. Crear un nuevo usuario en la VM:

```
'#' useradd -md /home/user1 -s /bin/bash user1
'#' passwd user1
'#' usermod -aG wheel user1
'#' visudo /etc/sudoers

    root ALL=(ALL) ALL

    ## Uncomment to allow members of group wheel to execute any command
    %wheel ALL=(ALL) ALL
```

4. Instalar sudo y SSH en la VM:

```
'#' pacman -Sy
'#' pacman -S openssh
'#' pacman -S sudo
```

5. Habilitar openSSH server

```
'#' systemctl restart sshd
'#' systemctl status sshd
```

6. Desde un terminal en el host conectarnos por SSH a la VM y cambiar a `root`:

```
$ ssh user1@192.168.1.24
[user1@archiso ~]$ sudo su
root@archiso /home/user1 '#' cd ~
root@archiso ~ '#'
```

---

1. Configurar la distribución del teclado (keyboard layout) a español:

```
'#' ls /usr/share/kbd/keymaps/**/*.map.gz | grep la-
'#' loadkeys la-latin1
```

2. Comprobar conectividad a Internet:

```
'#' ping archlinux.org
```

3. Actualizar el reloj del sistema

```
'#' timedatectl set-ntp true
'#' timedatectl list-timezones | grep Lima

    America/Lima

'#' timedatectl set-timezone America/Lima
'#' timedatectl status

                           Local time: Mon 2020-01-20 15:58:09 -05
                       Universal time: Mon 2020-01-20 20:58:09 UTC
                             RTC time: Mon 2020-01-20 20:58:09
                            Time zone: America/Lima (-05, -0500)
           System clock synchronized: yes
    systemd-timesyncd.service active: yes
                     RTC in local TZ: no
```

4. Listar dispositivos y particiones identificados en el sistema (con `lblk` o `fdisk`):

- Con `lsblk`:

```
'#' lsblk
NAME  MAJ:MIN RM   SIZE RO TYPE MOUNTPOINT
loop0   7:0    0 541.5M  1 loop /run/archiso/sfs/airootfs
sda     8:0    0    12G  0 disk
sr0    11:0    1   656M  0 rom  /run/archiso/bootmnt
sr1    11:1    1  1024M  0 rom
```

- Con `fdisk`:

```
'#' fdisk -l
Disk /dev/sda: 12 GiB, 12884901888 bytes, 25165824 sectors
Disk model: VBOX HARDDISK
Units: sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
```

(continues on next page)

(continued from previous page)

```
I/O size (minimum/optimal): 512 bytes / 512 bytes

Disk /dev/loop0: 541.5 MiB, 567787520 bytes, 1108960 sectors
Units: sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes
```

Vemos que tenemos un único disco de 12 GiB (`sda`), en el que haremos las particiones e instalaremos el sistema.

Note: La herramienta `fdisk` tiene una lista de distintos de particiones para una unidad de almacenamiento tipo disco duro:

```
'#' fdisk /dev/sda

Welcome to fdisk (util-linux 2.31.1).
Changes will remain in memory only, until you decide to write them.
Be careful before using the write command.

Command (m for help): m

Help:

DOS (MBR)
a   toggle a bootable flag
b   edit nested BSD disklabel
c   toggle the dos compatibility flag

Generic
d   delete a partition
F   list free unpartitioned space
l   list known partition types
n   add a new partition
p   print the partition table
t   change a partition type
v   verify the partition table
i   print information about a partition

Misc
m   print this menu
u   change display/entry units
x   extra functionality (experts only)

Script
I   load disk layout from sfdisk script file
O   dump disk layout to sfdisk script file

Save & Exit
w   write table to disk and exit
q   quit without saving changes

Create a new label
g   create a new empty GPT partition table
G   create a new empty SGI (IRIX) partition table
o   create a new empty DOS partition table
s   create a new empty Sun partition table
```

(continues on next page)

(continued from previous page)

```
Command (m for help): l

0  Empty            24  NEC DOS         81  Minix / old Lin bf  Solaris
1  FAT12            27  Hidden NTFS Win 82  Linux swap / So c1  DRDOS/sec (FAT-
2  XENIX root       39  Plan 9          83  Linux           c4  DRDOS/sec (FAT-
3  XENIX usr        3c  PartitionMagic  84  OS/2 hidden or  c6  DRDOS/sec (FAT-
4  FAT16 <32M       40  Venix 80286     85  Linux extended  c7  Syrinx
5  Extended         41  PPC PReP Boot   86  NTFS volume set da  Non-FS data
6  FAT16            42  SFS             87  NTFS volume set db  CP/M / CTOS / .
7  HPFS/NTFS/exFAT  4d  QNX4.x          88  Linux plaintext de  Dell Utility
8  AIX              4e  QNX4.x 2nd part 8e  Linux LVM       df  BootIt
9  AIX bootable     4f  QNX4.x 3rd part 93  Amoeba          e1  DOS access
a  OS/2 Boot Manag  50  OnTrack DM      94  Amoeba BBT      e3  DOS R/O
b  W95 FAT32        51  OnTrack DM6 Aux 9f  BSD/OS          e4  SpeedStor
c  W95 FAT32 (LBA)  52  CP/M            a0  IBM Thinkpad hi ea  Rufus alignment
e  W95 FAT16 (LBA)  53  OnTrack DM6 Aux a5  FreeBSD         eb  BeOS fs
f  W95 Extd (LBA)   54  OnTrackDM6      a6  OpenBSD         ee  GPT
10  OPUS            55  EZ-Drive        a7  NeXTSTEP        ef  EFI (FAT-12/16/
11  Hidden FAT12    56  Golden Bow      a8  Darwin UFS      f0  Linux/PA-RISC b
12  Compaq diagnost 5c  Priam Edisk     a9  NetBSD          f1  SpeedStor
14  Hidden FAT16 <3 61  SpeedStor       ab  Darwin boot     f4  SpeedStor
16  Hidden FAT16    63  GNU HURD or Sys af  HFS / HFS+      f2  DOS secondary
17  Hidden HPFS/NTF 64  Novell Netware  b7  BSDI fs         fb  VMware VMFS
18  AST SmartSleep  65  Novell Netware  b8  BSDI swap       fc  VMware VMKCORE
1b  Hidden W95 FAT3 70  DiskSecure Mult bb  Boot Wizard hid fd  Linux raid auto
1c  Hidden W95 FAT3 75  PC/IX           bc  Acronis FAT32 L fe  LANstep
1e  Hidden W95 FAT1 80  Old Minix       be  Solaris boot    ff  BBT
```

5. Particionar disco(s) (con `fdisk` o `cfdisk`) (Guía - How to use fdisk to Manage Partitions on Linux):

Note: `fdisk` y `cfdisk` hacen lo mismo, pero `cfdisk` es más interactivo

Deseamos el siguiente esquema de particiones:

- Label Type: DOS (para sistemas legacy BIOS) / GPT (para sistemas UEFI)
- **Partición 1:**
  - Size: 10GB/12GB
  - Primary
  - Type: Linux (83)(default)
  - Bootable (`Boot *`)
- **Partición 2:**
  - Size: 1GB/12GB
  - Primary
  - Type: Linux swap (82)

5.1 Usar el comando fdisk sobre la partición del disco seleccionado para la instalación del SO:

```
'#' fdisk /dev/sda
```

(continues on next page)

(continued from previous page)

```
Welcome to fdisk (util-linux 2.34).
Changes will remain in memory only, until you decide to write them.
Be careful before using the write command.

Device does not contain a recognized partition table.
Created a new DOS disklabel with disk identifier 0x82871abf.
```

5.2 Imprimir la tabla de particiones con la opción `p`:

```
Command (m for help): p
Disk /dev/sda: 12 GiB, 12884901888 bytes, 25165824 sectors
Disk model: VBOX HARDDISK
Units: sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes
Disklabel type: dos
Disk identifier: 0x82871abf
```

5.3 Crear una nueva partición con la opción `n`. En este caso creamos una partición primaria de 10GiB de tamaño y tipo Linux (por defecto):

```
Command (m for help): n
Partition type
   p   primary (0 primary, 0 extended, 4 free)
   e   extended (container for logical partitions)
Select (default p): p
Partition number (1-4, default 1): 1
First sector (2048-25165823, default 2048):
Last sector, +/-sectors or +/-size{K,M,G,T,P} (2048-25165823, default 25165823): +10G

Created a new partition 1 of type 'Linux' and of size 10 GiB.
```

5.4 Crear otra nueva partición con la opción `n`. En este caso creamos una partición primaria de 1GiB de tamaño y tipo Linux (por defecto):

```
Command (m for help): n
Partition type
   p   primary (1 primary, 0 extended, 3 free)
   e   extended (container for logical partitions)
Select (default p): p
Partition number (2-4, default 2):
First sector (20973568-25165823, default 20973568):
Last sector, +/-sectors or +/-size{K,M,G,T,P} (20973568-25165823, default 25165823):␣
↪+1G
```

5.5 Cambiaremos el tipo de la última partición con la opción `t`, de Linux (83) a swap (82):

```
Created a new partition 2 of type 'Linux' and of size 1 GiB.

Command (m for help): t
Partition number (1,2, default 2): 2
Hex code (type L to list all codes): L

 0  Empty           24  NEC DOS         81  Minix / old Lin bf  Solaris
 1  FAT12           27  Hidden NTFS Win 82  Linux swap / So c1  DRDOS/sec (FAT-
 2  XENIX root      39  Plan 9          83  Linux           c4  DRDOS/sec (FAT-
```

(continues on next page)

(continued from previous page)

```
 3  XENIX usr       3c  PartitionMagic  84  OS/2 hidden or  c6  DRDOS/sec (FAT-
 4  FAT16 <32M      40  Venix 80286     85  Linux extended  c7  Syrinx
 5  Extended        41  PPC PReP Boot   86  NTFS volume set da  Non-FS data
 6  FAT16           42  SFS             87  NTFS volume set db  CP/M / CTOS / .
 7  HPFS/NTFS/exFAT 4d  QNX4.x          88  Linux plaintext de  Dell Utility
 8  AIX             4e  QNX4.x 2nd part 8e  Linux LVM       df  BootIt
 9  AIX bootable    4f  QNX4.x 3rd part 93  Amoeba          e1  DOS access
 a  OS/2 Boot Manag 50  OnTrack DM      94  Amoeba BBT      e3  DOS R/O
 b  W95 FAT32       51  OnTrack DM6 Aux 9f  BSD/OS          e4  SpeedStor
 c  W95 FAT32 (LBA) 52  CP/M            a0  IBM Thinkpad hi ea  Rufus alignment
 e  W95 FAT16 (LBA) 53  OnTrack DM6 Aux a5  FreeBSD         eb  BeOS fs
 f  W95 Ext'd (LBA) 54  OnTrackDM6      a6  OpenBSD         ee  GPT
10  OPUS            55  EZ-Drive        a7  NeXTSTEP        ef  EFI (FAT-12/16/
11  Hidden FAT12    56  Golden Bow      a8  Darwin UFS      f0  Linux/PA-RISC b
12  Compaq diagnost 5c  Priam Edisk     a9  NetBSD          f1  SpeedStor
14  Hidden FAT16 <3 61  SpeedStor       ab  Darwin boot     f4  SpeedStor
16  Hidden FAT16    63  GNU HURD or Sys af  HFS / HFS+      f2  DOS secondary
17  Hidden HPFS/NTF 64  Novell Netware  b7  BSDI fs         fb  VMware VMFS
18  AST SmartSleep  65  Novell Netware  b8  BSDI swap       fc  VMware VMKCORE
1b  Hidden W95 FAT3 70  DiskSecure Mult bb  Boot Wizard hid fd  Linux raid auto
1c  Hidden W95 FAT3 75  PC/IX           bc  Acronis FAT32 L fe  LANstep
1e  Hidden W95 FAT1 80  Old Minix       be  Solaris boot    ff  BBT
Hex code (type L to list all codes): 82

Changed type of partition 'Linux' to 'Linux swap / Solaris'.
```

5.6 Listar las particiones con la opción `p` y asegurarnos que están conformes con nuestros requerimientos:

```
Command (m for help): p

Disk /dev/sda: 12 GiB, 12884901888 bytes, 25165824 sectors
Disk model: VBOX HARDDISK
Units: sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes
Disklabel type: dos
Disk identifier: 0x82871abf

Device     Boot    Start      End  Sectors Size Id Type
/dev/sda1           2048 20973567 20971520  10G 83 Linux
/dev/sda2       20973568 23070719  2097152   1G 82 Linux swap / Solaris
```

5.7 Escribir los cambios hechos al disco con la opción `w`:

```
Command (m for help): w
The partition table has been altered.
Calling ioctl() to re-read partition table.
Syncing disks.
```

5.8 Listar los discos y particiones para confirmar que hemos creado las particiones correctamente:

```
'#' lsblk
NAME   MAJ:MIN RM   SIZE RO TYPE MOUNTPOINT
loop0    7:0    0 541.5M  1 loop /run/archiso/sfs/airootfs
sda      8:0    0    12G  0 disk
├─sda1   8:1    0    10G  0 part
└─sda2   8:2    0     1G  0 part
```

(continues on next page)

(continued from previous page)

```
sr0     11:0    1   656M  0 rom  /run/archiso/bootmnt
sr1     11:1    1  1024M  0 rom
```

6. Formatear la partición Linux para crear un filesystem tipo `ext4`:

```
'#' mkfs.ext4 /dev/sda1

mke2fs 1.45.4 (23-Sep-2019)
Creating filesystem with 2621440 4k blocks and 655360 inodes
Filesystem UUID: b6a91f40-6a90-4093-88ff-ca45d4eea178
Superblock backups stored on blocks:
        32768, 98304, 163840, 229376, 294912, 819200, 884736, 1605632

Allocating group tables: done
Writing inode tables: done
Creating journal (16384 blocks): done
Writing superblocks and filesystem accounting information: done
```

7. Formatear la partición swap:

```
'#' mkswap /dev/sda2
Setting up swapspace version 1, size = 1024 MiB (1073737728 bytes)
no label, UUID=d4b01b51-459a-4171-a6df-676646f4ff94
```

```
'#' swapon /dev/sda2
```

8. Montar el filesystem de la partición Linux:

```
'#' mount /dev/sda1 /mnt/
```

Instalación

1. Seleccionar los mirrors. En el archivo `/etc/pacman.d/mirrorlist` definimos la lista de servidores mirror de donde se descargarán los paquetes. Eliminar servidores que no deseemos con un editor de texto (`vim` o `nano`):

Note: En el archivo `/etc/pacman.d/mirrorlist` donde se listan los mirrors, cuanto más arriba se posicione un mirror en la lista, más prioridad tendrá al descargar un paquete.

```
'#' vim /etc/pacman.d/mirrorlist
```

```
'#' cat /etc/pacman.d/mirrorlist
##
## Arch Linux repository mirrorlist
## Filtered by mirror score from mirror status page
## Generated on 2020-01-01
##

## United States
Server = http://mirror.dal10.us.leaseweb.net/archlinux/$repo/os/$arch
## United States
Server = http://archlinux.surlyjake.com/archlinux/$repo/os/$arch
```

(continues on next page)

(continued from previous page)

```
## United States
Server = http://ca.us.mirror.archlinux-br.org/$repo/os/$arch
## United States
Server = http://mirror.kaminski.io/archlinux/$repo/os/$arch
```

2. Instalar paquetes esenciales del sistema Arch Linux

```
'#' pacstrap /mnt base linux linux-firmware
```

Configurar el sistema

1. Generar un archivo fstab (filesystem table) para montar correctamente las particiones:

```
'#' genfstab -U /mnt >> sudo /mnt/etc/fstab
```

2. Cambiar root (`chroot`) en el nuevo sistema:

Más información de chroot de Wiki de Arch.

```
root@archiso ~ '#' arch-chroot /mnt
[root@archiso /]'#'
```

3. Configurar el time zone:

```
'#' ln -sf /usr/share/zoneinfo/America/Lima /etc/localetime
'#' hwclock --systohc
```

4. Instalar un editor de texto en el entorno chroot (comandos no encontrados en chroot):

```
'#' pacman -S vim
```

5. Localización.

5.1 Descomentar la línea `es_PE.UTF-8 UTF-8` del archivo `/etc/locale.gen`:

```
'#' vim /etc/locale.gen

'#' cat /etc/locale.gen

    ...
    #en_SG.UTF-8 UTF-8
    en_US.UTF-8 UTF-8
    #en_US ISO-8859-1
    ...
    #es_PA ISO-8859-1
    es_PE.UTF-8 UTF-8
    #es_PE ISO-8859-1
    ...
```

5.2 Generar archivos de locación en base al archivo `/etc/locale.gen`:

```
'#' locale-gen

Generating locales...
en_US.UTF-8... done
```

(continues on next page)

(continued from previous page)

```
es_PE.UTF-8... done
Generation complete.
```

5.3 Crear el archivo `locale.conf` con la variable `LANG` (`LANG=en_US.UTF-8` para idioma inglés y `LANG=es_PE.UTF-8` para español):

```
'#' echo LANG=en_US.UTF-8 > /etc/locale.conf # export LANG=en_US.UTF-8

'#' cat /etc/locale.conf

    LANG=en_US.UTF-8
```

5.4 Para realizar los cambios hechos anteriormente del keyboard layout, editar el archivo `/etc/vconsole.conf`:

```
'#' vim /etc/vconsole.conf

'#' cat /etc/vconsole.conf

    KEYMAP=la-latin1
```

6. Configuración de red

6.1 Crear el archivo `/etc/hostname`:

```
'#' echo arch > /etc/hostname
'#' cat /etc/hostname

    arch
```

6.2 Editar el archivo `/etc/hosts/` y agregar 3 líneas:

```
'#' vim /etc/hosts
'#' cat /etc/hosts

    127.0.0.1       localhost
    ::1                 localhost
    127.0.1.1       arch.localdomain        arch
```

7. Configurar una contraseña root:

```
'#' passwd

    New password:
    Retype new password:
    passwd: password updated successfully
```

8. Bootloader:

8.1 Descargar los paquetes de grub:

```
'#' pacman -S grub
```

8.2 Instalemos grub en el disco:

```
'#' grub-install /dev/sda

Installing for i386-pc platform.
Installation finished. No error reported.
```

8.3 Crearemos un archivo de configuración de grub:

```
'#' grub-mkconfig -o /boot/grub/grub.cfg

Generating grub configuration file ...
Found linux image: /boot/vmlinuz-linux
Found initrd image: /boot/initramfs-linux.img
Found fallback initrd image(s) in /boot: initramfs-linux-fallback.img
done
```

9. Habilitar el servicio de DHCP `dhcpcd`:

```
'#' pacman -S dhcpcd

'#' systemctl enable dhcpcd
Created symlink /etc/systemd/system/multi-user.target.wants/dhcpcd.service → /usr/lib/
↪systemd/system/dhcpcd.service.
```

Cuando reiniciemos nuestro SO recién instalado, debería habilitar DHCP automáticamente.

10. Crear un nuevo usuario con permisos de sudo:

```
'#' useradd -md /home/user1 -s /bin/bash user1
'#' passwd user1
'#' usermod -aG wheel user1

'#' pacman -S sudo
'#' visudo /etc/sudoers

        root ALL=(ALL) ALL

        ## Uncomment to allow members of group wheel to execute any command
        %wheel ALL=(ALL) ALL
```

Agregar al usuario a demás grupos:

```
'#' usermod -aG audio,video,optical,storage
```

11. Salir de chroot y apagar la VM:

```
[root@archiso /]'#' exit
[user1@archiso ~]$ sudo shutdown now
```

8.1.4 Reconfigurando la VM e iniciando el sistema

1. Cuando la VM se haya apagado, debemos remover el medio de instalación. En VirtuaBox, ir a *Settings* de la VM, sección *Storage* y remover el disco .iso con el botón inferior:
2. Iniciar la VM y veremos que inicia el Grub con las opciones de arranque. Seleccionar *Arch Linux*:
3. Veremos la pantalla de inicio del SO de Arch Linux:

8.1.5 Referencias

- guía de instalación oficial de la Wiki de Arch Linux
- página de descargas de Arch Linux

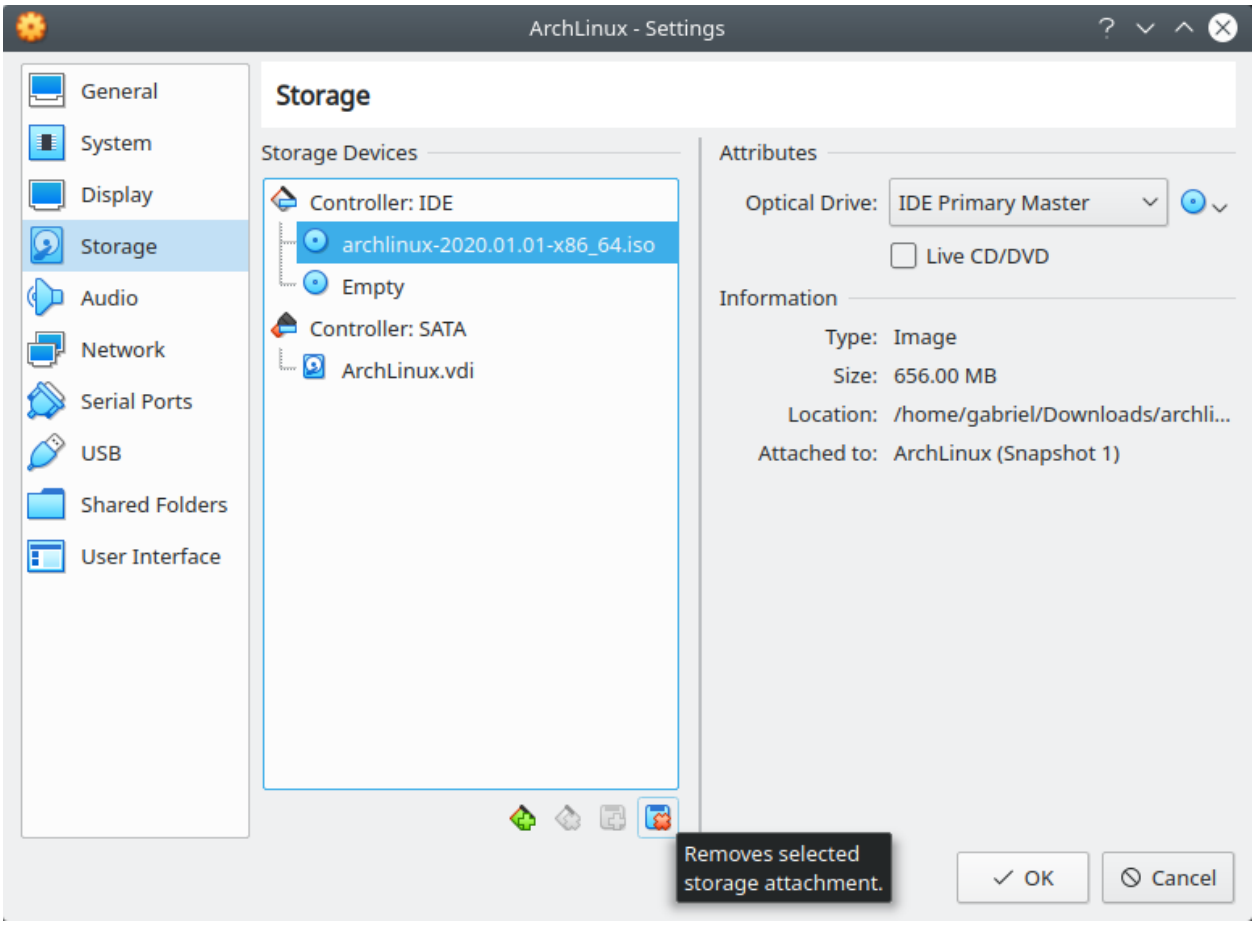


Fig. 23: Arch Linux instalado - Remover imagen .iso

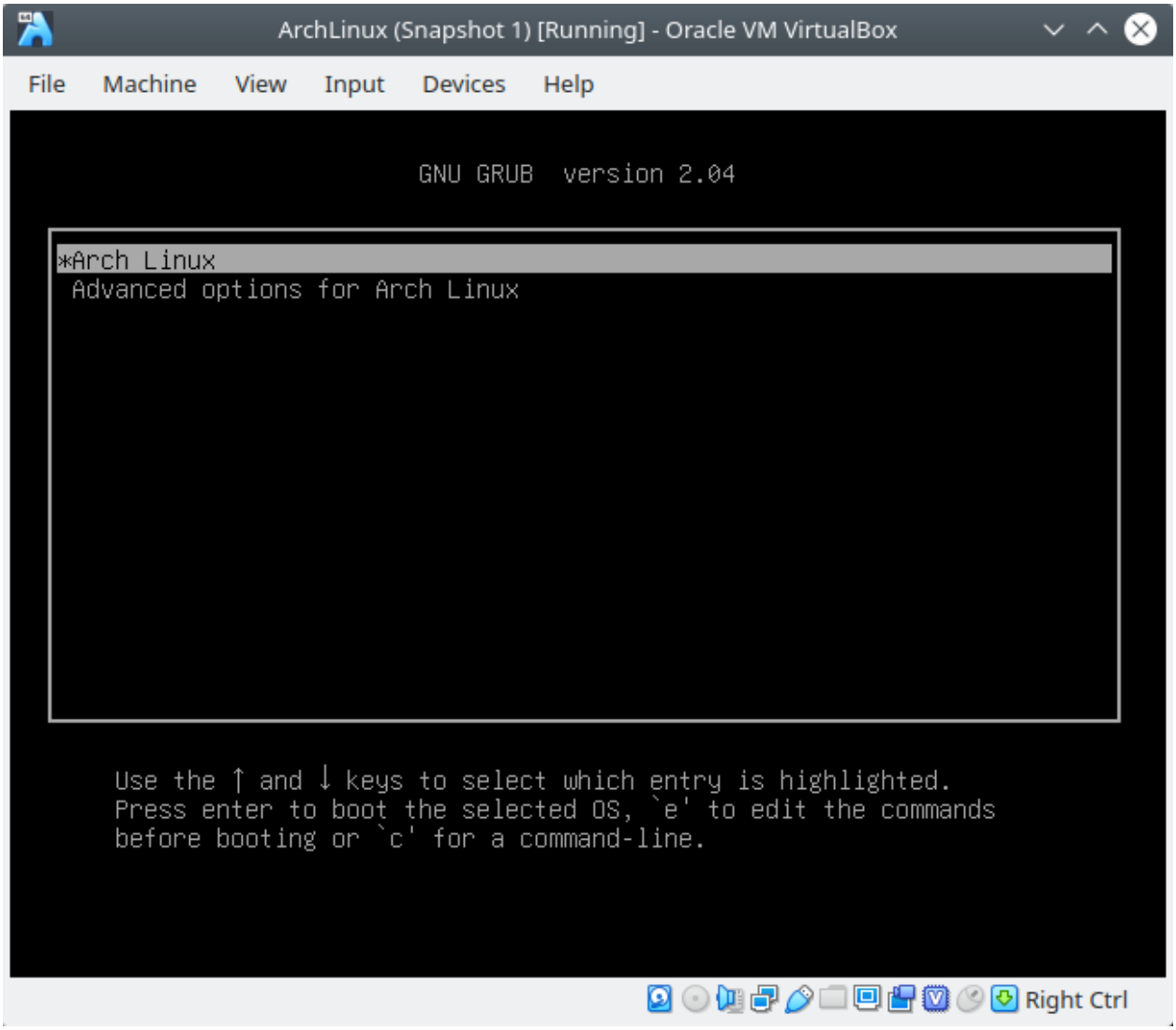


Fig. 24: Arch Linux instalado - Grub

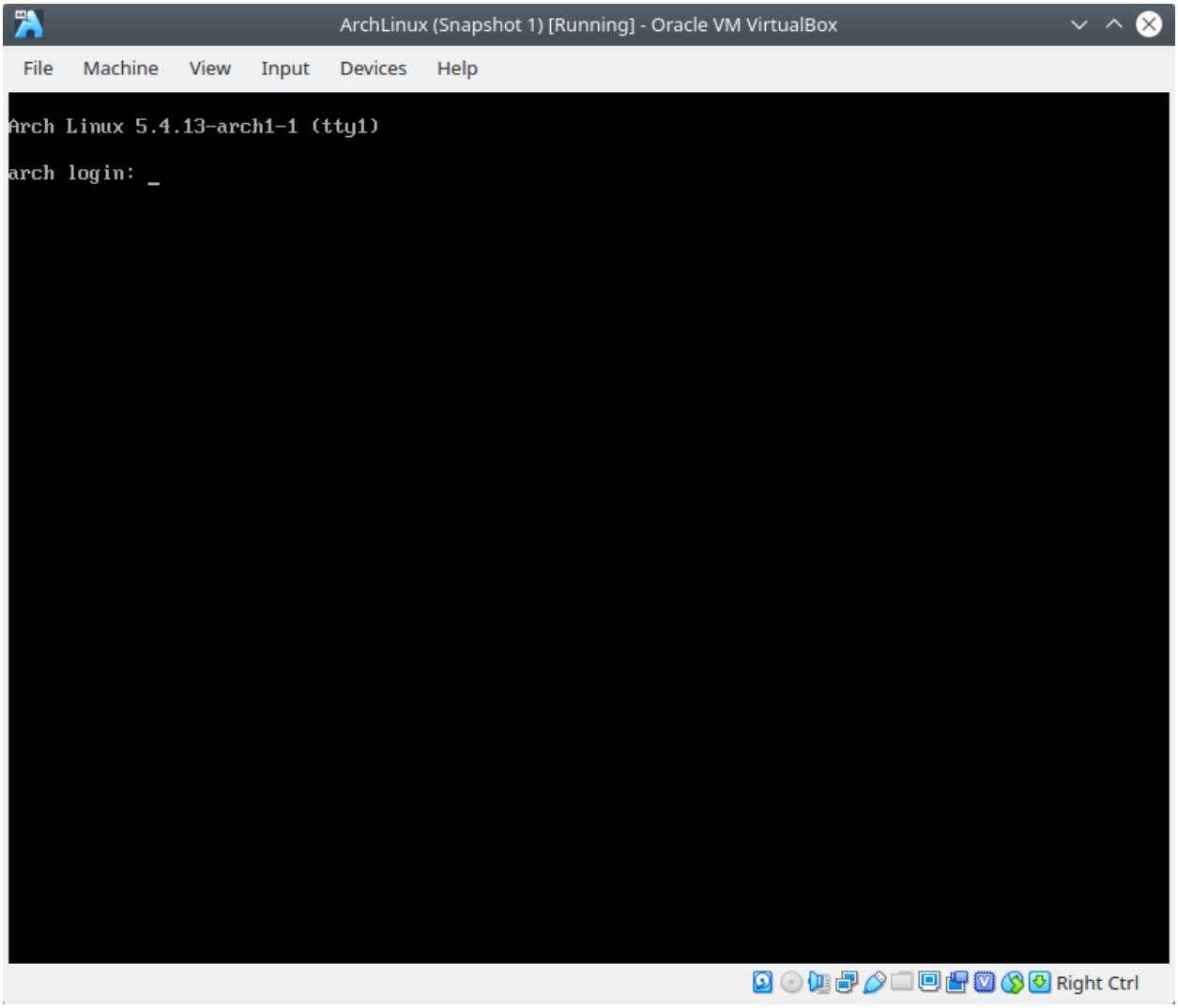


Fig. 25: Arch Linux instalado - Grub

- Guía - How to use fdisk to Manage Partitions on Linux
- información de chroot de Wiki de Arch
- comandos no encontrados en chroot
- Videotutorial - Arch Linux Installation Guide (2019)
- Arch Linux Guía básica de instalación actualizada
- Wiki Arch Linux - Install Arch Linux on a USB key
- Wiki Arch Linux - Install Arch Linux from existing Linux
- Wiki Arch Linux - Network configuration
- Wiki Arch Linux - Users and groups

8.2 Instalando Arch en un dispositivo USB

Table of Contents

El procedimiento para instalar Arch Linux en un dispositivo de almacenamiento USB comparte varios pasos con el procedimiento general de instalar Arch Linux en una PC (ver *Instalando Arch*).

Requisitos:

- Debemos contar con una PC habilitada para cambiar el orden de arranque de los dispositivos de almacenamiento conectados (disco duro, USB, CD-ROM).
- 2 USB vacíos: en uno crearemos el medio de instalación .iso booteable (**live USB**) y el el otro instalaremos el SO en sí (**persistent USB**).
- Una imagen ISO de Arch Linux descargada (página de descargas de Arch Linux)

8.2.1 Creando un medio de instalación

En un USB de poca capacidad (p.ej. 8 GB) escribiremos la imagen ISO de Arch Linux, de forma que sirva como medio de instalación del SO.

Para escrbir un medio de instalación USB booteable sigamos los siguientes pasos:

- En Linux:

1. Conectar el live USB a la PC y desmontarlo:

```
# umount /dev/sdXn
'#' umount /dev/sdb1
```

2. Escribir el medio de instalación en el USB live:

```
# dd bs=4M if=/path/archlinux.iso of=/dev/sdX status=progress && sync
'#' dd bs=4M if=/home/gabriel/Downloads/archlinux-2020.01.01-x86_64.iso of=/dev/sdb␣
↪status=progress && sync
```

Note:

- `if=/path/archlinux.iso`: es la ruta hacia la imagen ISO
- `of=/dev/sdX`: es el dispositivo USB (sdb, sdc, sdd, ...)

- En Windows:

Podemos usar programas como Rufus, Universal USB Installer o YUMI.

Luego de este paso ya deberíamos tener un USB live con el medio de instalación de Arch Linux booteable. Ahora lo arranqueremos en nuestra máquina.

8.2.2 Arrancar el medio de instalación ISO live

Important: En la BIOS o UEFI del sistema debemos cambiar el orden de arranque de los dispositivos y poner con mayor prioridad a los puertos USB. Según el modelo de la computadora, los pasos serán distintos. Seguir la referencia: BIOS-UEFI - Boot from a CD DVD USB Drive or SD Card

---

1. Con la PC apagada, insertar el USB live en la computadora.
2. Si hemos cambiado el orden de los dispositivos en la BIOS o UEFI, arrancará el medio USB.
3. En el menú bootloader elegir la opción *Boot Arch Linux (x86_64)*. Arrancará el medio de instalación de Arch Linux

Cuando haya terminado de cargar el arranque de la imagen iniciaremos sesión automáticamente como usuarios `root` bajo el prompt:

```
root@archiso ~ '#'
```

8.2.3 Proceso de instalación de Arch Linux

Conexión a Internet

Por defecto, el medio de instalación de Arch Linux reconocerá las interfaces de red que tengamos en el enlace, ya sea una interfaz de red cableada o inalámbrica. Sin embargo, el uso de ambas interfaces en el proceso de instalación es distinto.

La conexión recomendada es a través de una red cableada debido a la mayor velocidad y la configuración de red automática.

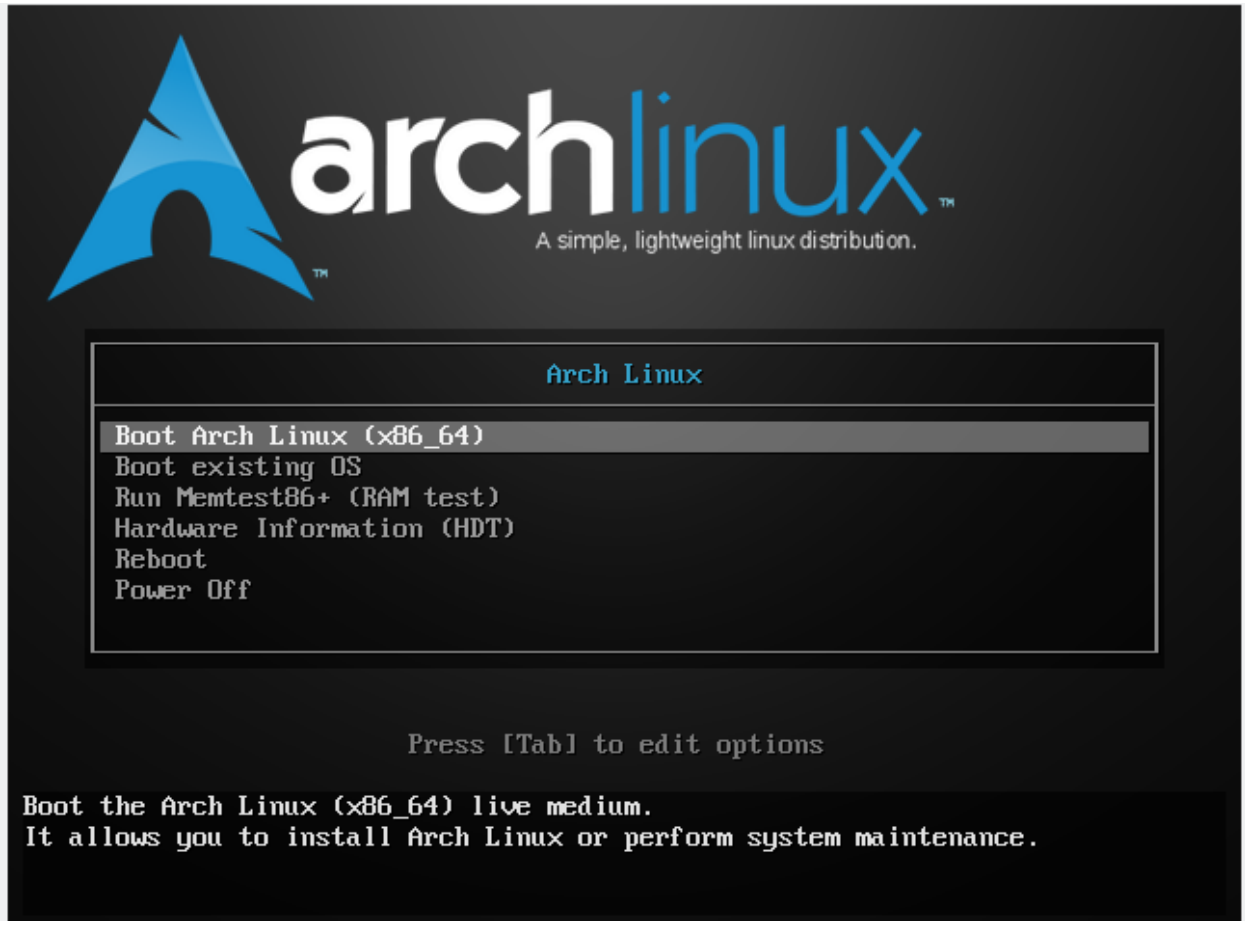


Fig. 26: Live ISO image de Arch Linux

Conexión Cableada

Si tenemos una interfaz física cableada conectada a la red local, donde existe un servidor DHCP configurado, recibiremos la configuración de red automáticamente al arrancar el medio de instalación. Comprobar la conectividad a Internet con:

```
'#' ping archlinux.org
```

En caso que tengamos problemas de conexión a la red cableada debemos hacer troubleshooting con los comandos `ip` y `dhclient`.

Conexión Inalámbrica

1. La mayoría de interfaces inalámbricas son soportadas por los drivers en el medio de instalación ISO. Para comprobar que nuestro sistema es compatible, revisar que los drivers del kernel se hayan cargado para la interfaz inalámbrica:

```
'#' lspci -k | grep -A3 'Network controller'
```

2. Si los drives han cargado, levantar la interfaz inalámbrica:

```
'#' ip link set dev wlan0 up
```

3. Escanear redes disponibles:

```
'#' iw dev wlan0 scan | grep 'SSID:'
```

4. Revisar qué tipo de autenticación es requerida por la red a la que nos deseamos conectar:

```
'#' iw dev wlan0 scan | less
```

5. Según el tipo de seguridad en la red inalámbrica usar el correspondiente comando:

- Red inalámbrica con encriptación WPA/WPA2:

```
'#' wpa_supplicant -i wlan0 -c <(wpa_passphrase 'networkname' 'password')
```

- Red inalámbrica con encriptación WEP:

```
'#' iw dev wlan0 connect 'networkname' key 0:'password'
```

- Red inalámbrica sin encriptación:

```
'#' iw dev wlan0 connect 'networkname'
```

6. Una vez que se ha establecido conexión bifurcar el proceso presionando Ctrl+Z (suspende el proceso) y ejecutando:

```
'#' bg
```

7. Recibir configuración de red por DHCP:

```
'#' dhclient -v wlan0
```

Pre-instalación

1. Configurar la distribución del teclado (keyboard layout) a español:

```
'#' loadkeys la-latin1
```

2. Comprobar conectividad a Internet:

```
'#' ping archlinux.org
```

3. Actualizar el reloj del sistema

```
'#' timedatectl set-ntp true
'#' timedatectl set-timezone America/Lima
'#' timedatectl status

                      Local time: Mon 2020-01-20 15:58:09 -05
                  Universal time: Mon 2020-01-20 20:58:09 UTC
                        RTC time: Mon 2020-01-20 20:58:09
                       Time zone: America/Lima (-05, -0500)
       System clock synchronized: yes
systemd-timesyncd.service active: yes
                  RTC in local TZ: no
```

4. Reconocer el medio USB donde instalaremos el SO:

4.1 Listar dispositivos y particiones identificados en el sistema (con `lblk` o `fdisk`):

```
'#' lsblk

NAME   MAJ:MIN RM   SIZE RO TYPE MOUNTPOINT
loop0    7:0    0 541.5M  1 loop /run/archiso/sfs/airootfs
sda      8:0    0 232.9G  0 disk
├─sda1   8:1    0   549M  0 part
├─sda2   8:2    0 144.5G  0 part
├─sda3   8:3    0  83.8G  0 part
├─sda4   8:4    0     1K  0 part
└─sda5   8:5    0   4.1G  0 part
sdb      8:16   1  58.8G  0 disk
├─sdb1   8:17   1   656M  0 part /run/archiso/bootmnt
└─sdb2   8:18   1    64M  0 part
sr0     11:0    1  1024M  0 rom
```

Note: La herramienta `fdisk` tiene una lista de distintos de particiones para una unidad de almacenamiento tipo USB (no es la misma lista de opciones que para un disco duro):

```
'#' fdisk /dev/sdc

Welcome to fdisk (util-linux 2.34).
Changes will remain in memory only, until you decide to write them.
Be careful before using the write command.

Command (m for help): m

Help:

GPT
```

(continues on next page)

(continued from previous page)

```
M   enter protective/hybrid MBR

Generic
d   delete a partition
F   list free unpartitioned space
l   list known partition types
n   add a new partition
p   print the partition table
t   change a partition type
v   verify the partition table
i   print information about a partition

Misc
m   print this menu
x   extra functionality (experts only)

Script
I   load disk layout from sfdisk script file
O   dump disk layout to sfdisk script file

Save & Exit
w   write table to disk and exit
q   quit without saving changes

Create a new label
g   create a new empty GPT partition table
G   create a new empty SGI (IRIX) partition table
o   create a new empty DOS partition table
s   create a new empty Sun partition table

Command (m for help): l
1 EFI System                     C12A7328-F81F-11D2-BA4B-00A0C93EC93B
1 EFI System                     C12A7328-F81F-11D2-BA4B-00A0C93EC93B
2 MBR partition scheme           024DEE41-33E7-11D3-9D69-0008C781F39F
3 Intel Fast Flash               D3BFE2DE-3DAF-11DF-BA40-E3A556D89593
4 BIOS boot                      21686148-6449-6E6F-744E-656564454649
5 Sony boot partition            F4019732-066E-4E12-8273-346C5641494F
6 Lenovo boot partition          BFBFAFE7-A34F-448A-9A5B-6213EB736C22
7 PowerPC PReP boot              9E1A2D38-C612-4316-AA26-8B49521E5A8B
8 ONIE boot                      7412F7D5-A156-4B13-81DC-867174929325
9 ONIE config                    D4E6E2CD-4469-46F3-B5CB-1BFF57AFC149
10 Microsoft reserved             E3C9E316-0B5C-4DB8-817D-F92DF00215AE
11 Microsoft basic data           EBD0A0A2-B9E5-4433-87C0-68B6B72699C7
12 Microsoft LDM metadata         5808C8AA-7E8F-42E0-85D2-E1E90434CFB3
13 Microsoft LDM data             AF9B60A0-1431-4F62-BC68-3311714A69AD
14 Windows recovery environment   DE94BBA4-06D1-4D40-A16A-BFD50179D6AC
15 IBM General Parallel Fs        37AFFC90-EF7D-4E96-91C3-2D7AE055B174
16 Microsoft Storage Spaces       E75CAF8F-F680-4CEE-AFA3-B001E56EFC2D
17 HP-UX data                     75894C1E-3AEB-11D3-B7C1-7B03A0000000
18 HP-UX service                  E2A1E728-32E3-11D6-A682-7B03A0000000
19 Linux swap                     0657FD6D-A4AB-43C4-84E5-0933C84B4F4F
20 Linux filesystem               0FC63DAF-8483-4772-8E79-3D69D8477DE4
21 Linux server data              3B8F8425-20E0-4F3B-907F-1A25A76F98E8
22 Linux root (x86)               44479540-F297-41B2-9AF7-D131D5F0458A
23 Linux root (ARM)               69DAD710-2CE4-4E3C-B16C-21A1D49ABED3
24 Linux root (x86-64)            4F68BCE3-E8CD-4DB1-96E7-FBCAF984B709
25 Linux root (ARM-64)            B921B045-1DF0-41C3-AF44-4C6F280D3FAE
```

(continues on next page)

(continued from previous page)

```
26 Linux root  (IA-64)              993D8D3D-F80E-4225-855A-9DAF8ED7EA97
27 Linux reserved                   8DA63339-0007-60C0-C436-083AC8230908
28 Linux home                       933AC7E1-2EB4-4F13-B844-0E14E2AEF915
29 Linux RAID                       A19D880F-05FC-4D3B-A006-743F0F84911E
30 Linux extended boot              BC13C2FF-59E6-4262-A352-B275FD6F7172
31 Linux LVM                        E6D6D379-F507-44C2-A23C-238F2A3DF928
32 FreeBSD data                     516E7CB4-6ECF-11D6-8FF8-00022D09712B
33 FreeBSD boot                     83BD6B9D-7F41-11DC-BE0B-001560B84F0F
34 FreeBSD swap                     516E7CB5-6ECF-11D6-8FF8-00022D09712B
35 FreeBSD UFS                      516E7CB6-6ECF-11D6-8FF8-00022D09712B
36 FreeBSD ZFS                      516E7CBA-6ECF-11D6-8FF8-00022D09712B
37 FreeBSD Vinum                    516E7CB8-6ECF-11D6-8FF8-00022D09712B
38 Apple HFS/HFS+                   48465300-0000-11AA-AA11-00306543ECAC
39 Apple UFS                        55465300-0000-11AA-AA11-00306543ECAC
40 Apple RAID                       52414944-0000-11AA-AA11-00306543ECAC
41 Apple RAID offline               52414944-5F4F-11AA-AA11-00306543ECAC
42 Apple boot                       426F6F74-0000-11AA-AA11-00306543ECAC
43 Apple label                      4C616265-6C00-11AA-AA11-00306543ECAC
44 Apple TV recovery                5265636F-7665-11AA-AA11-00306543ECAC
45 Apple Core storage               53746F72-6167-11AA-AA11-00306543ECAC
46 Solaris boot                     6A82CB45-1DD2-11B2-99A6-080020736631
47 Solaris root                     6A85CF4D-1DD2-11B2-99A6-080020736631
48 Solaris /usr & Apple ZFS         6A898CC3-1DD2-11B2-99A6-080020736631
49 Solaris swap                     6A87C46F-1DD2-11B2-99A6-080020736631
50 Solaris backup                   6A8B642B-1DD2-11B2-99A6-080020736631
51 Solaris /var                     6A8EF2E9-1DD2-11B2-99A6-080020736631
52 Solaris /home                    6A90BA39-1DD2-11B2-99A6-080020736631
53 Solaris alternate sector         6A9283A5-1DD2-11B2-99A6-080020736631
54 Solaris reserved 1               6A945A3B-1DD2-11B2-99A6-080020736631
55 Solaris reserved 2               6A9630D1-1DD2-11B2-99A6-080020736631
56 Solaris reserved 3               6A980767-1DD2-11B2-99A6-080020736631
57 Solaris reserved 4               6A96237F-1DD2-11B2-99A6-080020736631
58 Solaris reserved 5               6A8D2AC7-1DD2-11B2-99A6-080020736631
59 NetBSD swap                      49F48D32-B10E-11DC-B99B-0019D1879648
60 NetBSD FFS                       49F48D5A-B10E-11DC-B99B-0019D1879648
61 NetBSD LFS                       49F48D82-B10E-11DC-B99B-0019D1879648
62 NetBSD concatenated              2DB519C4-B10E-11DC-B99B-0019D1879648
63 NetBSD encrypted                 2DB519EC-B10E-11DC-B99B-0019D1879648
64 NetBSD RAID                      49F48DAA-B10E-11DC-B99B-0019D1879648
65 ChromeOS kernel                  FE3A2A5D-4F32-41A7-B725-ACCC3285A309
66 ChromeOS root fs                 3CB8E202-3B7E-47DD-8A3C-7FF2A13CFCEC
67 ChromeOS reserved                2E0A753D-9E48-43B0-8337-B15192CB1B5E
68 MidnightBSD data                 85D5E45A-237C-11E1-B4B3-E89A8F7FC3A7
69 MidnightBSD boot                 85D5E45E-237C-11E1-B4B3-E89A8F7FC3A7
70 MidnightBSD swap                 85D5E45B-237C-11E1-B4B3-E89A8F7FC3A7
71 MidnightBSD UFS                  0394EF8B-237E-11E1-B4B3-E89A8F7FC3A7
72 MidnightBSD ZFS                  85D5E45D-237C-11E1-B4B3-E89A8F7FC3A7
73 MidnightBSD Vinum                85D5E45C-237C-11E1-B4B3-E89A8F7FC3A7
74 Ceph Journal                     45B0969E-9B03-4F30-B4C6-B4B80CEFF106
75 Ceph Encrypted Journal           45B0969E-9B03-4F30-B4C6-5EC00CEFF106
76 Ceph OSD                         4FBD7E29-9D25-41B8-AFD0-062C0CEFF05D
77 Ceph crypt OSD                   4FBD7E29-9D25-41B8-AFD0-5EC00CEFF05D
78 Ceph disk in creation            89C57F98-2FE5-4DC0-89C1-F3AD0CEFF2BE
79 Ceph crypt disk in creation      89C57F98-2FE5-4DC0-89C1-5EC00CEFF2BE
80 VMware VMFS                      AA31E02A-400F-11DB-9590-000C2911D1B8
81 VMware Diagnostic                9D275380-40AD-11DB-BF97-000C2911D1B8
82 VMware Virtual SAN               381CFCCC-7288-11E0-92EE-000C2911D0B2
```

(continues on next page)

(continued from previous page)

```
83 VMware Virsto                    77719A0C-A4A0-11E3-A47E-000C29745A24
84 VMware Reserved                  9198EFFC-31C0-11DB-8F78-000C2911D1B8
85 OpenBSD data                     824CC7A0-36A8-11E3-890A-952519AD3F61
86 QNX6 file system                 CEF5A9AD-73BC-4601-89F3-CDEEEEE321A1
87 Plan 9 partition                 C91818F9-8025-47AF-89D2-F030D7000C2C
88 HiFive Unleashed FSBL            5B193300-FC78-40CD-8002-E86C45580B47
89 HiFive Unleashed BBL             2E54B353-1271-4842-806F-E436D6AF6985
```

4.2 Conectar a la PC el medio de almacenamiento USB donde instalaremos el SO.

4.3 Volver a correr el comando para listar los medios de almacenamiento conectados a la PC y ver cual es el nuevo dispositivo en la lista:

```
'#' lsblk

NAME   MAJ:MIN RM   SIZE RO TYPE MOUNTPOINT
loop0    7:0    0 541.5M  1 loop /run/archiso/sfs/airootfs
sda      8:0    0 232.9G  0 disk
├─sda1   8:1    0   549M  0 part
├─sda2   8:2    0 144.5G  0 part
├─sda3   8:3    0  83.8G  0 part
├─sda4   8:4    0     1K  0 part
└─sda5   8:5    0   4.1G  0 part
sdb      8:16   1  58.8G  0 disk
├─sdb1   8:17   1   656M  0 part /run/archiso/bootmnt
└─sdb2   8:18   1    64M  0 part
sdc      8:32   0 111.8G  0 disk
└─sdc1   8:33   0 111.8G  0 part
sr0     11:0    1  1024M  0 rom
```

Vemos que tenemos un nuevo dispositivo `sdc` en nuestra lista. Esta unidad es el USB que hemos conectado recientemente y donde haremos la instalación del SO.

5. Para realizar la partición del medio de almacenamiento usaremos dos herramientas: `gdisk` y `fdisk`.

Deseamos el siguiente esquema de particiones:

- **Partición 1:**
  - Con `gdisk`
  - Size: 10 MB
  - Type: BIOS boot partition (`ef02`)
- **Partición 2:**
  - Con `gdisk`
  - Size: 500 MB
  - Type: EFI System (`ef00`)
- **Partición 3:**
  - Con `gdisk`
  - Size: 70GB / 110 GB
  - Type: Linux filesystem (`8300`)
- **Partición 4:**

  - Con `fdisk`
  - Size: 40GB / 110 GB
  - Type: HPFS/NTFS/exFAT (`7`)

5.1 Primero eliminar todas las particiones del USB con `gdisk`:

```
'#' gdisk /dev/sdc

GPT fdisk (gdisk) version 1.0.4

Partition table scan:
MBR: MBR only
BSD: not present
APM: not present
GPT: not present

Found invalid GPT and valid MBR; converting MBR to GPT format
in memory. THIS OPERATION IS POTENTIALLY DESTRUCTIVE! Exit by
typing 'q' if you dont want to convert your MBR partitions
to GPT format!

Command (? for help): d
Using 1

Command (? for help): d
No partitions
```

Note:

- Opciones de `gdisk`:

```
Command (? for help): ?
b   back up GPT data to a file
c   change a partition s name
d   delete a partition
i   show detailed information on a partition
l   list known partition types
n   add a new partition
o   create a new empty GUID partition table (GPT)
p   print the partition table
q   quit without saving changes
r   recovery and transformation options (experts only)
s   sort partitions
t   change a partition s type code
v   verify disk
w   write table to disk and exit
x   extra functionality (experts only)
?   print this menu
```

- Lista de tipos de formato en `gdisk`:

```
Command (? for help): L
Type search string, or <Enter> to show all codes:
0700 Microsoft basic data  0c01 Microsoft reserved    2700 Windows RE
3000 ONIE boot             3001 ONIE config           3900 Plan 9
4100 PowerPC PReP boot     4200 Windows LDM data      4201 Windows LDM metadata
```

(continues on next page)

(continued from previous page)

```
4202 Windows Storage Spac  7501 IBM GPFS              7f00 ChromeOS kernel
7f01 ChromeOS root         7f02 ChromeOS reserved     8200 Linux swap
8300 Linux filesystem      8301 Linux reserved        8302 Linux /home
8303 Linux x86 root (/)    8304 Linux x86-64 root (/  8305 Linux ARM64 root (/)
8306 Linux /srv            8307 Linux ARM32 root (/)  8308 Linux dm-crypt
8309 Linux LUKS            8400 Intel Rapid Start     8e00 Linux LVM
a000 Android bootloader    a001 Android bootloader 2  a002 Android boot 1
a003 Android recovery 1    a004 Android misc          a005 Android metadata
a006 Android system 1      a007 Android cache         a008 Android data
a009 Android persistent    a00a Android factory       a00b Android fastboot/ter
a00c Android OEM           a00d Android vendor        a00e Android config
a00f Android factory (alt  a010 Android meta          a011 Android EXT
a012 Android SBL1          a013 Android SBL2          a014 Android SBL3
a015 Android APPSBL        a016 Android QSEE/tz       a017 Android QHEE/hyp
a018 Android RPM           a019 Android WDOG debug/s  a01a Android DDR
a01b Android CDT           a01c Android RAM dump      a01d Android SEC
a01e Android PMIC          a01f Android misc 1        a020 Android misc 2
a021 Android device info   a022 Android APDP          a023 Android MSADP
a024 Android DPO           a025 Android recovery 2    a026 Android persist
a027 Android modem ST1     a028 Android modem ST2     a029 Android FSC
a02a Android FSG 1         a02b Android FSG 2         a02c Android SSD
a02d Android keystore      a02e Android encrypt       a02f Android EKSST
a030 Android RCT           a031 Android spare1        a032 Android spare2
a033 Android spare3        a034 Android spare4        a035 Android raw resource
a036 Android boot 2        a037 Android FOTA          a038 Android system 2
a039 Android cache         a03a Android user data     a03b LG (Android) advance
a03c Android PG1FS         a03d Android PG2FS         a03e Android board info
a03f Android MFG           a040 Android limits        a200 Atari TOS basic data
a500 FreeBSD disklabel     a501 FreeBSD boot          a502 FreeBSD swap
a503 FreeBSD UFS           a504 FreeBSD ZFS           a505 FreeBSD Vinum/RAID
a580 Midnight BSD data     a581 Midnight BSD boot     a582 Midnight BSD swap
a583 Midnight BSD UFS      a584 Midnight BSD ZFS      a585 Midnight BSD Vinum
a600 OpenBSD disklabel     a800 Apple UFS             a901 NetBSD swap
a902 NetBSD FFS            a903 NetBSD LFS            a904 NetBSD concatenated
a905 NetBSD encrypted      a906 NetBSD RAID           ab00 Recovery HD
af00 Apple HFS/HFS+        af01 Apple RAID            af02 Apple RAID offline
af03 Apple label           af04 AppleTV recovery      af05 Apple Core Storage
af06 Apple SoftRAID Statu  af07 Apple SoftRAID Scrat  af08 Apple SoftRAID Volum
af09 Apple SoftRAID Cache  af0a Apple APFS            b300 QNX6 Power-Safe
bc00 Acronis Secure Zone   be00 Solaris boot          bf00 Solaris root
bf01 Solaris /usr & Mac Z  bf02 Solaris swap          bf03 Solaris backup
bf04 Solaris /var          bf05 Solaris /home         bf06 Solaris alternate se
bf07 Solaris Reserved 1    bf08 Solaris Reserved 2    bf09 Solaris Reserved 3
bf0a Solaris Reserved 4    bf0b Solaris Reserved 5    c001 HP-UX data
c002 HP-UX service         e100 ONIE boot             e101 ONIE config
ea00 Freedesktop $BOOT     eb00 Haiku BFS             ed00 Sony system partitio
ed01 Lenovo system partit  ef00 EFI System            ef01 MBR partition scheme
ef02 BIOS boot partition   f800 Ceph OSD             f801 Ceph dm-crypt OSD
f802 Ceph journal          f803 Ceph dm-crypt journa  f804 Ceph disk in creatio
f805 Ceph dm-crypt disk i  f806 Ceph block            f807 Ceph block DB
f808 Ceph block write-ahe  f809 Ceph lockbox for dm-  f80a Ceph multipath OSD
f80b Ceph multipath journ  f80c Ceph multipath block  f80d Ceph multipath block
f80e Ceph multipath block  f80f Ceph multipath block  f810 Ceph dm-crypt block
f811 Ceph dm-crypt block   f812 Ceph dm-crypt block   f813 Ceph dm-crypt LUKS j
f814 Ceph dm-crypt LUKS b  f815 Ceph dm-crypt LUKS b  f816 Ceph dm-crypt LUKS b
f817 Ceph dm-crypt LUKS O  fb00 VMWare VMFS           fb01 VMWare reserved
fc00 VMWare kcore crash p  fd00 Linux RAID
```

5.2 Crear una nueva `GUID partition table (GPT)` con la opción `o`:

```
Command (? for help): o
This option deletes all partitions and creates a new protective MBR.
Proceed? (Y/N): y
```

5.3 Crear las primeras tres particiones con la opción `n` de gdisk:

Referencia: MBR vs GPT y BIOS vs UEFI

- Primero creamos una partición el Master Boot Record (MBR) de 10 MB al inicio de la memoria:

```
Command (? for help): n
Partition number (1-128, default 1):
First sector (34-234441614, default = 2048) or {+-}size{KMGTP}:
Last sector (2048-234441614, default = 234441614) or {+-}size{KMGTP}:+10MB
Current type is 'Linux filesystem'
Hex code or GUID (L to show codes, Enter = 8300): EF02
Changed type of partition to 'BIOS boot partition'
```

- Luego creamos una partición EFI de 500 MB:

```
Command (? for help): n
Partition number (2-128, default 2):
First sector (34-234441614, default = 22528) or {+-}size{KMGTP}:
Last sector (22528-234441614, default = 234441614) or {+-}size{KMGTP}: +500MB
Current type is 'Linux filesystem'
Hex code or GUID (L to show codes, Enter = 8300): EF00
Changed type of partition to 'EFI System'
```

- Finalmente, asignamos espacio para la partición Linux donde instalaremos el SO:

```
Command (? for help): n
Partition number (3-128, default 3):
First sector (34-234441614, default = 1046528) or {+-}size{KMGTP}:
Last sector (1046528-234441614, default = 234441614) or {+-}size{KMGTP}: +70G
Current type is 'Linux filesystem'
Hex code or GUID (L to show codes, Enter = 8300):
Changed type of partition to 'Linux filesystem'
```

5.4 Revisar que la tabla de particiones sea correcta:

```
Command (? for help): p
Disk /dev/sdc: 234441648 sectors, 111.8 GiB
Model: 550
Sector size (logical/physical): 512/4096 bytes
Disk identifier (GUID): 006577F7-F6D7-4A6D-96A3-8CB841B35647
Partition table holds up to 128 entries
Main partition table begins at sector 2 and ends at sector 33
First usable sector is 34, last usable sector is 234441614
Partitions will be aligned on 2048-sector boundaries
Total free space is 86596461 sectors (41.3 GiB)

Number  Start (sector)    End (sector)  Size       Code  Name
   1            2048           22527   10.0 MiB    EF02  BIOS boot partition
   2           22528         1046527   500.0 MiB   EF00  EFI System
   3         1046528       147847167   70.0 GiB    8300  Linux filesystem
```

5.5 Guardar los cambios realizados en el dispositivo de almacenamiento USB:

```
Command (? for help): w

Final checks complete. About to write GPT data. THIS WILL OVERWRITE EXISTING
PARTITIONS!!

Do you want to proceed? (Y/N): y
OK; writing new GUID partition table (GPT) to /dev/sdc.
The operation has completed successfully.
```

5.6 Haremos una última partición con `fdisk` para usarla como almacenamiento común (como si conectáramos un USB a la PC):

```
'#' fdisk /dev/sdc

Welcome to fdisk (util-linux 2.34).
Changes will remain in memory only, until you decide to write them.
Be careful before using the write command.

Command (m for help): p
Disk /dev/sdc: 111.81 GiB, 120034123776 bytes, 234441648 sectors
Disk model: 550
Units: sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 4096 bytes
I/O size (minimum/optimal): 4096 bytes / 4096 bytes
Disklabel type: gpt
Disk identifier: 006577F7-F6D7-4A6D-96A3-8CB841B35647

Device       Start       End   Sectors  Size Type
/dev/sdc1     2048     22527     20480   10M BIOS boot
/dev/sdc2    22528   1046527   1024000  500M EFI System
/dev/sdc3  1046528 147847167 146800640   70G Linux filesystem
```

Para crear una partición que se use como almacenamiento simple, le daremos el tipo **NTFS**:

Error: El type `7` debería pertenecer a `HPFS/NTFS/exFAT`. Comprobar esto usando el parámetro `t` de `fdisk`.

```
Command (m for help): n
Partition number (4-128, default 4):
First sector (147847168-234441614, default 147847168):
Last sector, +/-sectors or +/-size{K,M,G,T,P} (147847168-234441614, default␣
↪234441614): +40G

Created a new partition 4 of type 'Linux filesystem' and of size 40 GiB.

Command (m for help): t
Partition number (1-4, default 4): 4
Partition type (type L to list all types): 7

Changed type of partition 'Linux filesystem' to 'PowerPC PReP boot'.

Command (m for help): p
Disk /dev/sdc: 111.81 GiB, 120034123776 bytes, 234441648 sectors
Disk model: 550
Units: sectors of 1 * 512 = 512 bytes
```

(continues on next page)

(continued from previous page)

```
Sector size (logical/physical): 512 bytes / 4096 bytes
I/O size (minimum/optimal): 4096 bytes / 4096 bytes
Disklabel type: gpt
Disk identifier: 006577F7-F6D7-4A6D-96A3-8CB841B35647

Device         Start       End   Sectors  Size Type
/dev/sdc1       2048     22527     20480   10M BIOS boot
/dev/sdc2      22528   1046527   1024000  500M EFI System
/dev/sdc3    1046528 147847167 146800640   70G Linux filesystem
/dev/sdc4  147847168 231733247  83886080   40G PowerPC PReP boot

Command (m for help): w
The partition table has been altered.
Calling ioctl() to re-read partition table.
Syncing disks.
```

5.7 Listar las particiones de la unidad de almacenamiento para confirmar que hemos creado las particiones correctamente:

```
'#' lsblk /dev/sdc
NAME   MAJ:MIN RM   SIZE RO TYPE MOUNTPOINT
sdc      8:32   0 111.8G  0 disk
├─sdc1   8:33   0    10M  0 part
├─sdc2   8:34   0   500M  0 part
├─sdc3   8:35   0    70G  0 part
└─sdc4   8:36   0    40G  0 part
```

6. Formatear las particiones creadas:

Caution: La partición `sdc1` que contiene a la información de BIOS/MBR no será formateada.

6.1 Formatear la partición EFI `sdc2` con un filesystem `FAT32`:

```
'#' mkfs.fat -F32 /dev/sdc2
mkfs.fat 4.1 (2017-01-24)
```

6.2 Formatear la partición Linux `sdc3` como filesystem `ext4`:

```
'#' mkfs.ext4 /dev/sdc3
mke2fs 1.45.4 (23-Sep-2019)
Creating filesystem with 18350080 4k blocks and 4587520 inodes
Filesystem UUID: c99ccd7a-d73d-42f3-ac3e-cd92156a6c96
Superblock backups stored on blocks:
    32768, 98304, 163840, 229376, 294912, 819200, 884736, 1605632, 2654208,
    4096000, 7962624, 11239424

Allocating group tables: done
Writing inode tables: done
Creating journal (131072 blocks): done
Writing superblocks and filesystem accounting information: done
```

6.3 Formatear la partción `sdc4` como filesystem NTFS:

```
'#' mkfs.ntfs -f /dev/sdc4
```

(continues on next page)

(continued from previous page)

```
Cluster size has been automatically set to 4096 bytes.
Creating NTFS volume structures.
mkntfs completed successfully. Have a nice day.
```

7. Montar el filesystem de las 3 particiones formateadas:

```
'#' mkdir -p /mnt/usb
'#' mount /dev/sdc3 /mnt/usb

'#' mkdir /mnt/usb/boot
'#' mount /dev/sdc2 /mnt/usb/boot

'#' mkdir /mnt/storage
'#' mount /dev/sdc4 /mnt/storage
```

Instalación

1. Seleccionar los mirrors. En el archivo `/etc/pacman.d/mirrorlist` definimos la lista de servidores mirror de donde se descargarán los paquetes. Eliminar servidores que no deseemos con un editor de texto (`vim` o `nano`):

Note: En el archivo `/etc/pacman.d/mirrorlist` donde se listan los mirrors, cuanto más arriba se posicione un mirror en la lista, más prioridad tendrá al descargar un paquete.

```
'#' vim /etc/pacman.d/mirrorlist
```

```
'#' cat /etc/pacman.d/mirrorlist
##
## Arch Linux repository mirrorlist
## Filtered by mirror score from mirror status page
## Generated on 2020-01-01
##

## United States
Server = http://mirror.dal10.us.leaseweb.net/archlinux/$repo/os/$arch
## United States
Server = http://archlinux.surlyjake.com/archlinux/$repo/os/$arch
## United States
Server = http://ca.us.mirror.archlinux-br.org/$repo/os/$arch
## United States
Server = http://mirror.kaminski.io/archlinux/$repo/os/$arch
```

2. Instalar paquetes esenciales del sistema Arch Linux

```
'#' pacstrap /mnt/usb base linux linux-firmware
```

Configurar el sistema

1. Generar un archivo fstab (filesystem table) para montar correctamente las particiones:

El archivo fstab (ver Wikipedia - fstab) es usado en sistemas Linux para montar correctamente las particiones de disco correctamente al momento del arranque. Estas particioens pueden identificarse en el fstab de diferentes maneras, y

algunos métodos aún usan las etiquetas estándar (`/dev/...`), en lugar de UUIDs (ver Wikipedia - Universally unique identifier). Esto será de seguro un punto de falla de una instalación en un USB, pues las etiquetas estándar asignadas para dispositivos removibles no son consistentes luego de cada arranque del sistema.

```
'#' genfstab -U /mnt/usb >> sudo /mnt/usb/etc/fstab
```

Nuestro dispositivo USB debe contener ahora un sistema Linux permanente. Pero aún nos falta configurar algunas cosas antes de que inicie por sí mismo.

2. Cambiar root (`chroot`) en el nuevo sistema:

Más información de chroot de Wiki de Arch.

```
root@archiso ~ '#' arch-chroot /mnt/usb
[root@archiso /]'#'
```

3. Configurar el time zone:

```
'#' ln -sf /usr/share/zoneinfo/America/Lima /etc/localetime
'#' hwclock --systohc
```

4. Instalar un editor de texto en el entorno chroot (comandos no encontrados en chroot):

```
'#' pacman -S vim
```

5. Localización.

5.1 Descomentar la línea `es_PE.UTF-8 UTF-8` del archivo `/etc/locale.gen`:

```
'#' vim /etc/locale.gen

'#' cat /etc/locale.gen

    ...
    #en_SG.UTF-8 UTF-8
    en_US.UTF-8 UTF-8
    #en_US ISO-8859-1
    ...
    #es_PA ISO-8859-1
    es_PE.UTF-8 UTF-8
    #es_PE ISO-8859-1
    ...
```

5.2 Generar archivos de locación en base al archivo `/etc/locale.gen`:

```
'#' locale-gen

Generating locales...
en_US.UTF-8... done
es_PE.UTF-8... done
Generation complete.
```

5.3 Crear el archivo `locale.conf` con la variable `LANG` (`LANG=en_US.UTF-8` para idioma inglés y `LANG=es_PE.UTF-8` para español):

```
'#' echo LANG=en_US.UTF-8 > /etc/locale.conf # export LANG=en_US.UTF-8

'#' cat /etc/locale.conf
```

(continues on next page)

(continued from previous page)

```
    LANG=en_US.UTF-8
```

5.4 Para realizar los cambios hechos anteriormente del keyboard layout, editar el archivo `/etc/vconsole.conf`:

```
'#' vim /etc/vconsole.conf

'#' cat /etc/vconsole.conf

    KEYMAP=la-latin1
```

6. Configuración de red

6.1 Crear el archivo `/etc/hostname`:

```
'#' echo arch > /etc/hostname
'#' cat /etc/hostname

    arch
```

6.2 Editar el archivo `/etc/hosts/` y agregar 3 líneas:

```
'#' vim /etc/hosts
'#' cat /etc/hosts

    127.0.0.1       localhost
    ::1                 localhost
    127.0.1.1       arch.localdomain        arch
```

7. Configurar una contraseña root:

```
'#' passwd

    New password:
    Retype new password:
    passwd: password updated successfully
```

8. Bootloader. Para permitir que el dispositivo USB arranque en ambos modos (BIOS y UEFI), necesitamos instalar 2 bootloader. Por facilidad de instalación, instalaremos GRUB para ambos modos:

8.1 Descargar los paquetes de `grub` y `efibootmgr`:

```
'#' pacman -S grub efibootmgr
```

8.2 Ver los block devices para determinar el dispositivo USB objetivo:

```
'#' lsblk
```

El dispositivo USB es `/dev/sdX` sin ningún número.

8.3 Configurar GRUB para el **modo de arranque MBR/BIOS**:

```
'#' grub-install --target=i386-pc --boot-directory /boot /dev/sdc
```

8.4 Configurar GRUB para el **modo de arranque UEFI**:

```
'#' grub-install --target=x86_64-efi --efi-directory /boot --boot-directory /boot --
↪removable
Installing for x86_64-efi platform.
Installation finished. No error reported.
```

8.5 Generar una configuración GRUB:

```
'#' grub-mkconfig -o /boot/grub/grub.cfg
Generating grub configuration file ...
Found linux image: /boot/vmlinuz-linux
Found initrd image: /boot/initramfs-linux.img
Found fallback initrd image(s) in /boot: initramfs-linux-fallback.img
done
```

9. Habilitar el servicio de DHCP `dhcpcd`:

```
'#' pacman -S dhcpcd

'#' systemctl enable dhcpcd
Created symlink /etc/systemd/system/multi-user.target.wants/dhcpcd.service → /usr/lib/
↪systemd/system/dhcpcd.service.
```

Cuando reiniciemos nuestro SO recién instalado, debería habilitar DHCP automáticamente.

10. Crear un nuevo usuario con permisos de sudo:

```
'#' useradd -md /home/user1 -s /bin/bash user1
'#' passwd user1
'#' usermod -aG wheel user1

'#' pacman -S sudo
'#' visudo /etc/sudoers

        root ALL=(ALL) ALL

        ## Uncomment to allow members of group wheel to execute any command
        %wheel ALL=(ALL) ALL
```

11. Salir de chroot y apagar el sistema:

```
[root@archiso /]'#' exit
root@archiso ~ '#' umount /mnt/usb/boot /mnt/usb
root@archiso ~ '#' poweroff
```

Al volver a iniciar el sistema, si tenemos el arranque de dispositivos USB con prioridad, arrancará nuestro USB con el sistema operativo Linux.

8.3 Instalar un Desktop Environment en Arch

Table of Contents

Con el sistema operativo de Arch Linux instalado podemos agregarle un entorno de escritorio (dekstop environment) para tener una interfaz de usuario gráfica.

Existen varias alternativas de entornos de escritorio, pero los pasos de instalación son casi los mismos en todos.

8.3.1 Pre-requisitos

1. Usuario creado

Como pre-requisito, debemos tener un usuario creado con permisos de sudo.

2. **Display server (X server)**

Enlace entre software y hardware, permite a la PC mostrar las imágenes en pantalla y configurar la tarjeta gráfica de nuestra PC. La solución más popular en Linux es `xorg`:

```
'#' pacman -S xorg
```

Este comando instala 49 miembros en total (112 paquetes).

3. **Graphics driver**

El display server es inútil sin un graphics driver. Podemos instalar el driver específico de nuestro fabricante. Además podemos instalar un driver básico (`xf86-video-vesa`) si no estamos seguros de cual es nuestro fabricante de forma temporal, o si estamos en una VM, o como respaldo por si falla nuestro driver principal:

```
# Vesa
'#' pacman -S xf86-video-vesa
# AMD
'#' pacman -S xf86-video-amdgpu
# Intel
'#' pacman -S xf86-video-intel
#Nvidia
'#' pacman -S xf86-video-nouveau
```

4. **Display manager**

Es esencialmente nuestra pantalla de logueo. Inicia el **xorg display server** automáticamente para que no tenga que ser iniciado por línea de comandos cada vez que iniciamos el sistema. Nos provee con una lista de desktop environments instalados. La mayoría de desktop environments tiene un display manager recomendado a ser usado con él:

- KDE Plasma 5 - `sddm`
- GNOME - `gdm`
- LXDE - `lxdm`
- Universal Display Manager - `lightdm`, `mdm-display-manager`, `slim`, `xorg-xdm`

Note: En teoría podemos cualquier display manager podría funcionar con cualquier desktop environment, sin embargo, es más fácil instalar el que corresponde a nuestro desktop environment.

Para instalar KDE Plasma:

```
'#' pacman -S sddm
```

8.3.2 Instalando el Desktop Environment

1. Para instalar el desktop environment necesitamos instalar el grupo que contiene los paquetes para el desktop environment mismo. Los grupos que contienen los respectivos desktop environments son:

- KDE Plasma 5 - `plasma`
- Cinnamon - `cinnamon`
- GNOME - `gnome`
- LXDE - `lxde`
- MATE - `mate`
- Xfce - `xfce4`

El grupo que contiene los paquetes del desktop environment GNOME es `gnome`:

```
'#' pacman -S plasma
```

Por defecto, solo instala el desktop environment mínimo y no muchas aplicaciones.

2. La mayoría desktop environments tienen otro grupo con un conjunto de aplicaciones que podemos instalar junto con el desktop environment. Obtenemos programas como administradores de archivos, un terminal y media player.

- KDE Plasma 5 - `kde-applications`
- Cinnamon - N/A (`nemo-fileroller`)
- GNOME - `gnome-extra`
- LXDE - N/A
- MATE - `mate-extra`
- Xfce - `xfce4-goodies`

```
'#' pacman -S kde-applications
```

3. Habilitar el boot manager para que inicie automáticamente en el arranque del sistema:

```
'#' systemctl enable sddm
Created symlink /etc/systemd/system/display-manager.service → /usr/lib/systemd/system/
↪gdm.service.
```

4. Reiniciar el sistema

8.3.3 Referencias

- How to Install a Desktop Environment in Arch Linux
- Graphical user interface
- Desktop environment

Errores con GNOME en Arch Linux:

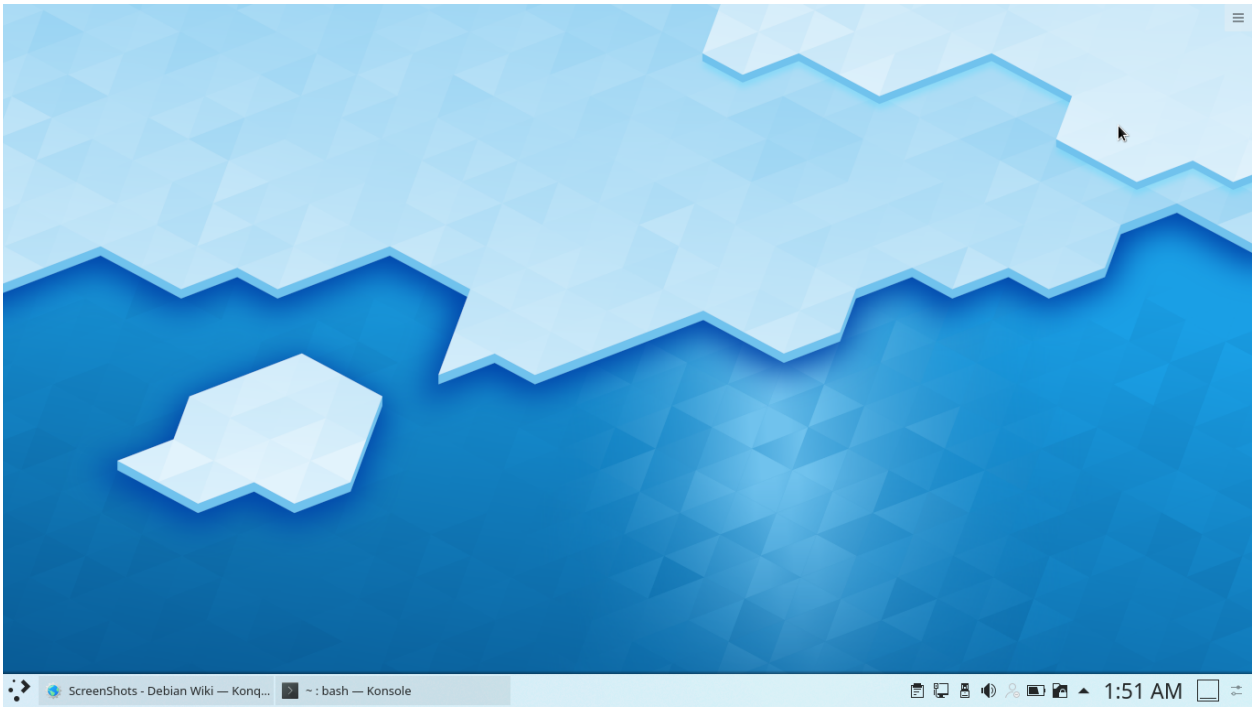


Fig. 27: Arch Linux - KDE Plasma Desktop environment

- Error - gdm black screen possible solution
- Error - gkr-pam unable to locate daemon control file
- Error - GdmDisplay session never registrered, failing

CHAPTER 9

Ubuntu

9.1 Instalando Ubuntu en un USB con mkusb

Table of Contents

En el método de creación de un **Linux live USB**, todo cambios que hagamos al sistema se perderá luego de un reinicio. Por otro lado, el método **Persistent Storage Live USB** guarda cualquier cambio que hagamos al live system, permaneciendo luego de un reinicio del sistema.

Esta documentación muestra cómo crear un persistent live USB con la distribución de Ubuntu (y cualquiera de sus flavors Xubuntu, Kubuntu, Ubuntu Mate, Lubuntu, etc), Linux Mint, Debian o Elementary OS. Esto lo realizaremos usando la herramienta `mkusb`, la cual puede ser instalada desde Ubuntu, Linux Mint o Debian.

`mkusb` puede crear persistent live drives que funcionan con los modos **UEFI** y **BIOS**. La partición permanente que crea mkusb usa `casper-rw`, lo cual le permite tener un tamaño mayor a 4GB.

`mkusb` además tiene otras funcionalidades como crear live USB booteables regulares de Linux, vaciar un dispositivo, y otras más. La interfaz gráfica de `mkusb` usa Zenity.

9.1.1 Instalando `mkusb`

En Ubuntu, Linux Mint, Elementary OS

Instalar `mkusb` en Ubuntu, Linux Mint, Elementary OS y otras distribuciones/flavors basados en Ubuntu mediante su PPA oficial:

```
$ sudo add-apt-repository ppa:/mkusb/ppa
$ sudo apt update
$ sudo apt install --install-recommends mkusb mkusb-nox usb-pack-efi
```

En Debian

A diferencia de la mayoría de repositorios PPA en Debian, podemos usar el mismo PPA usando para Ubuntu. Esto debido a que `mkusb` es un conjunto de scripts que no depende de versiones de paquetes de Ubuntu.

Agregar el PPA de `mksub` manualmente e instalar el programa en Debian:

```
$ echo "deb http://ppa.launchpad.net/mkusb/ppa/ubuntu bionic main" | sudo tee /etc/
↪apt/sources.list.d/mkusb.list

$ sudo apt-key adv --keyserver keyserver.ubuntu.com --recv 54B8C8AC
$ sudo apt update
$ sudo apt install --install-recommends mkusb mkusb-nox usb-pack-efi
```

9.1.2 Instalando el SO con `mkusb`

1. Abrir la aplicación `mkusb`. Nos preguntará si deseamos que `mkusb` corra en la versión `dus` (una interfaz renovada de `mkusb`):

Seleccionar *Yes*.

2. Aparecerá una ventana con un terminal y otra ventana pidiendo la contraseña `sudo`:

Ingresar la contraseña y clic en *OK*. Luego aparecerá una ventana de advertencia.

Clic en *OK*.

3. Luego `mkusb` mostrará otra ventana con una acciones a realizar:

Seleccionar *Install (make a boot device)* y luego clic en *OK*.

4. La siguiente pantalla mostrará otra lista de opciones:

Seleccionar *Persistent live - only Debian and Ubuntu*. Clic en *OK*.

5. Nos aparecerá un navegador de archivos para que seleccionemos el archivo ISO o IMG que queremos instalar en el USB:

Seleccionar la imagen de Ubuntu, Debian o Linux Mint de nuestro sistema. Clic en *OK*.

6. Luego, `mkusb` presentará una lista de dispositivos de almacenamiento. Seleccionar el dispositivo USB al cual le queremos instalar el SO:

Clic en *OK* y vuelve a confirmar que el dispositivo seleccionado es el indicado:

Clic en *Yes*.

> **Danger:** Seleccionar la unidad de almacenamiento correcta, pues toda la información dentro de esta será eliminada.

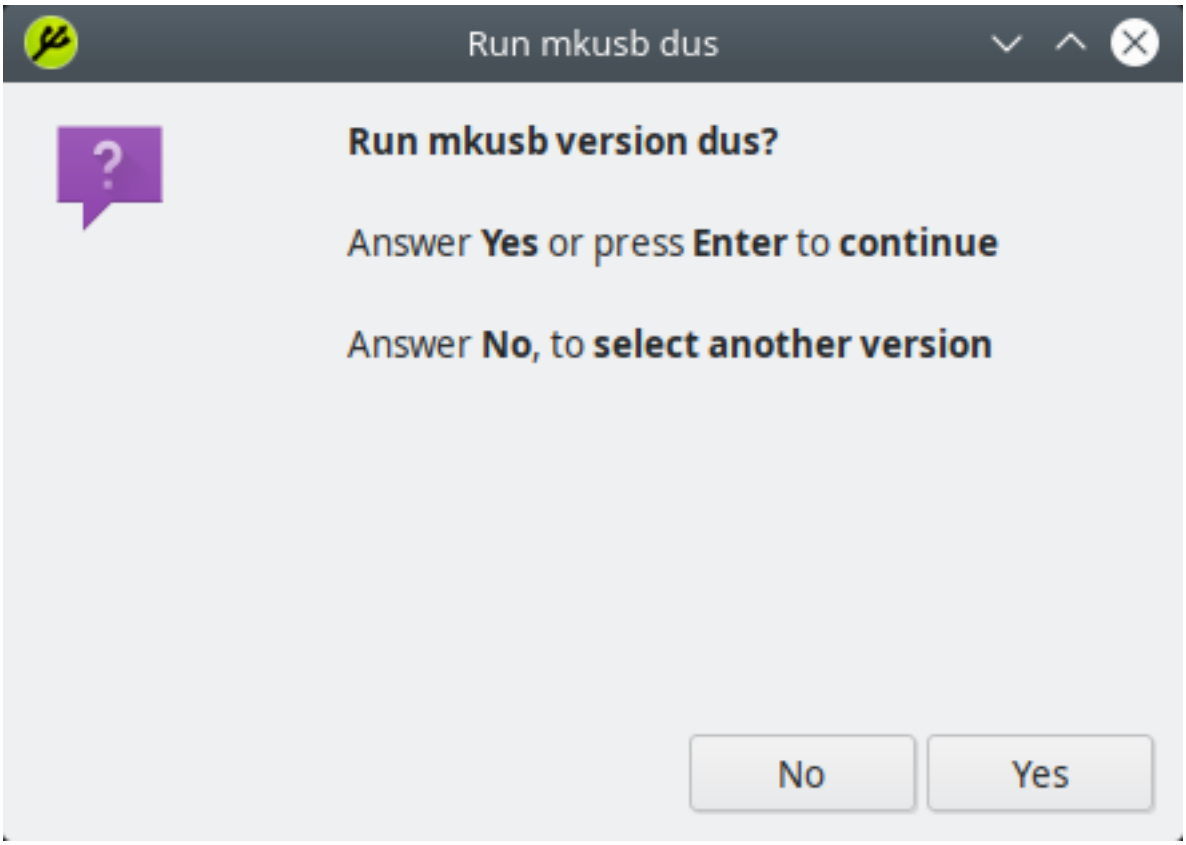


Fig. 1: `mkusb` - Correr la versión `dus`

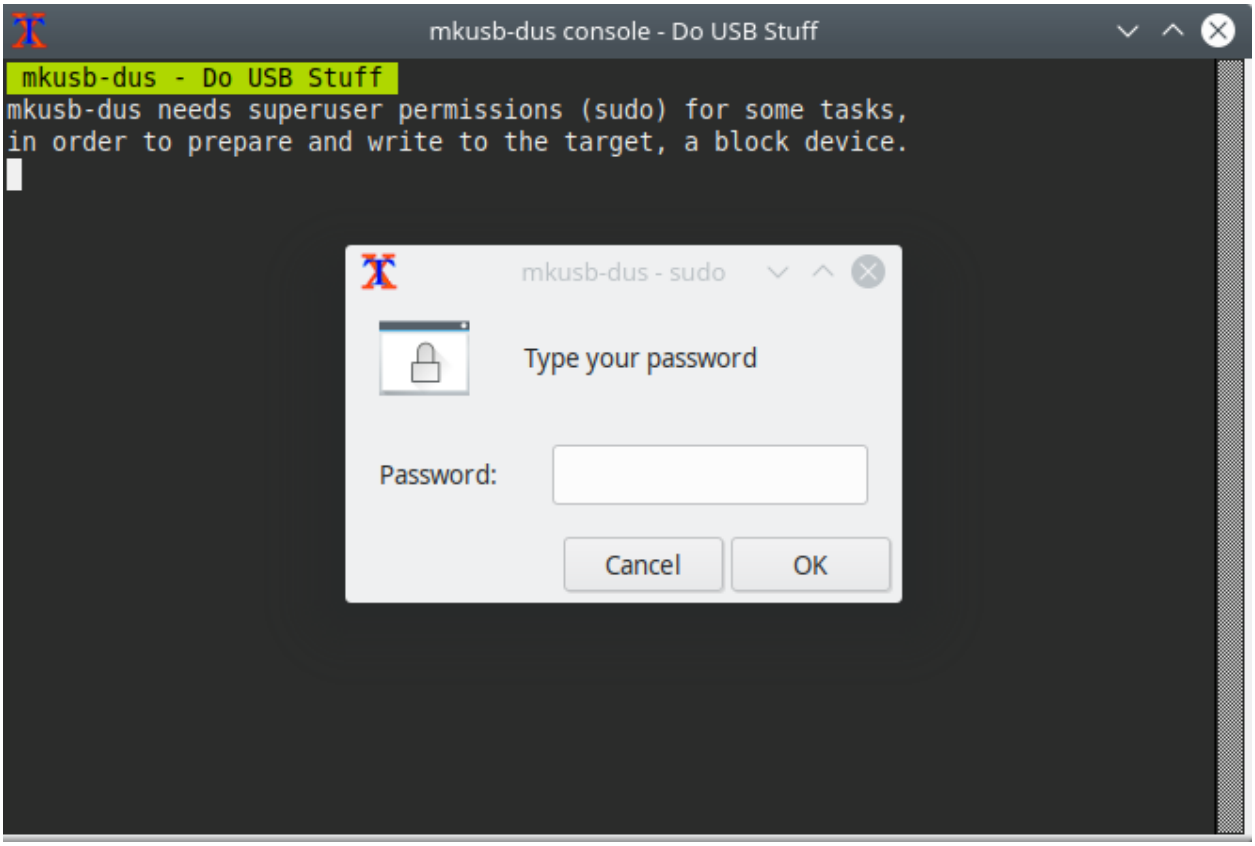


Fig. 2: `mkusb` - Ingresar contraseña root

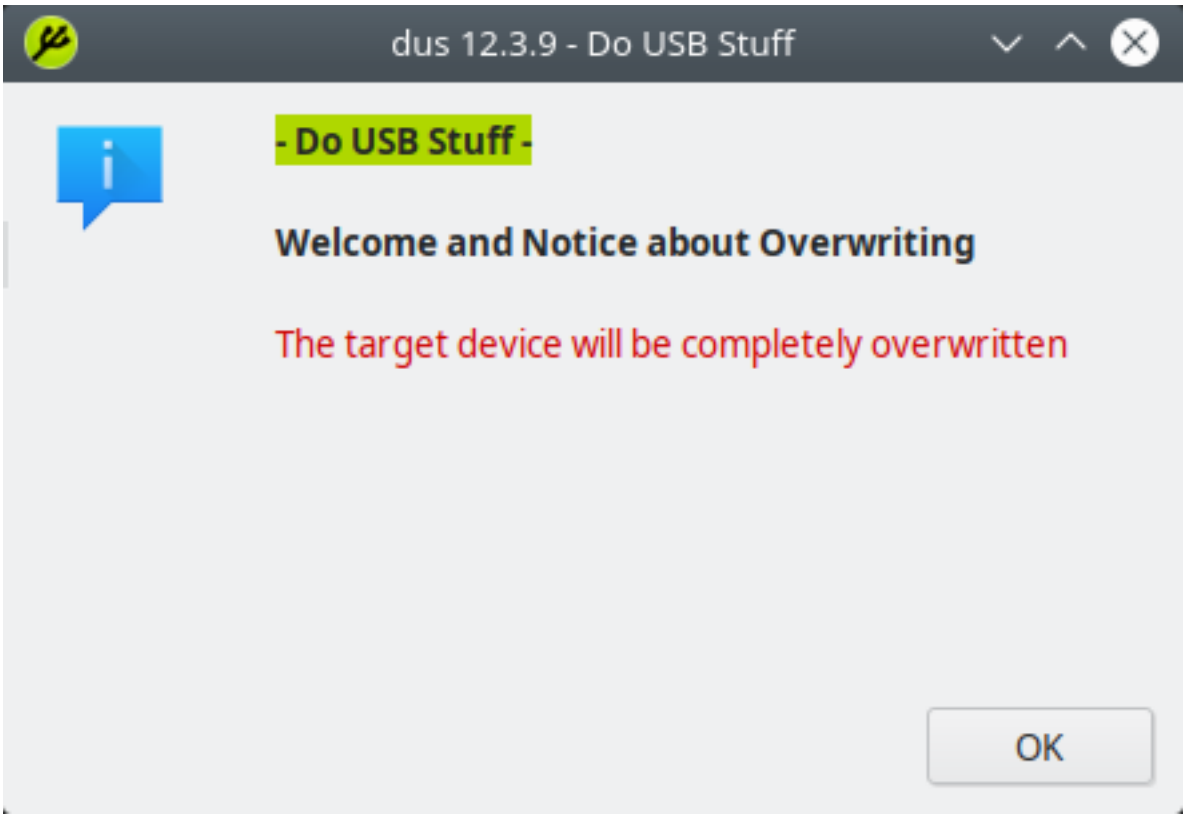


Fig. 3: `mkusb`

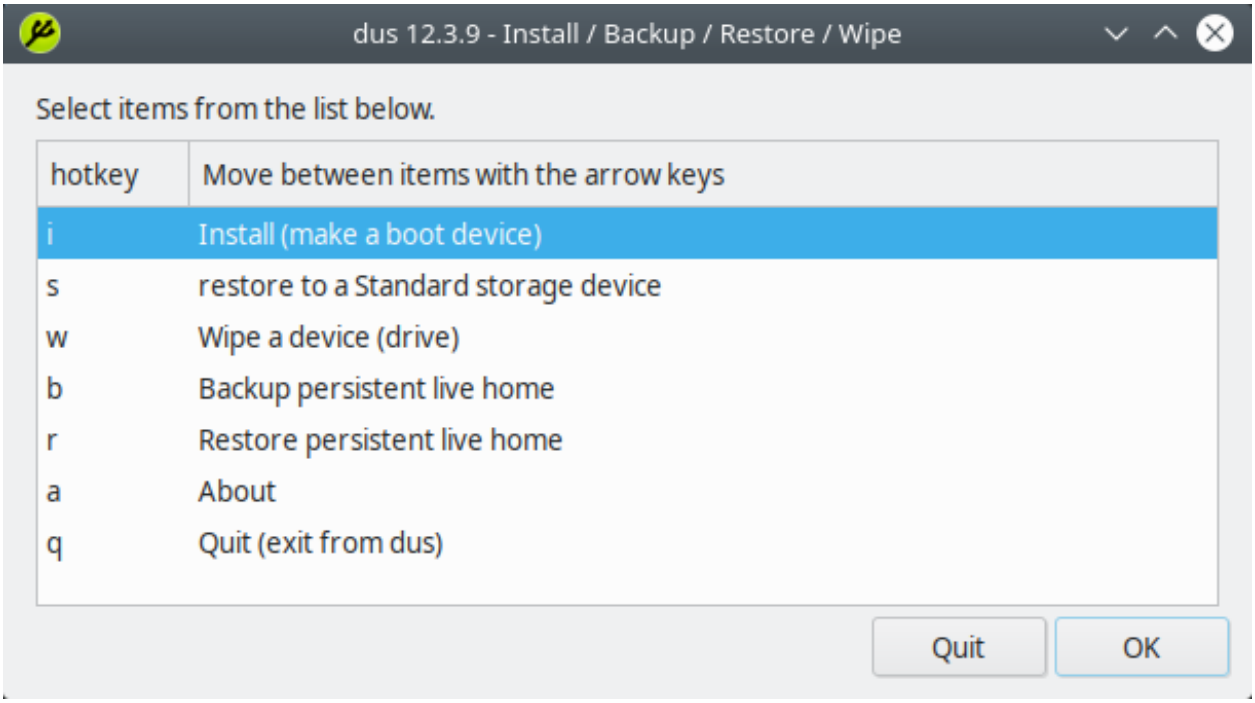


Fig. 4: `mkusb` - lista de acciones

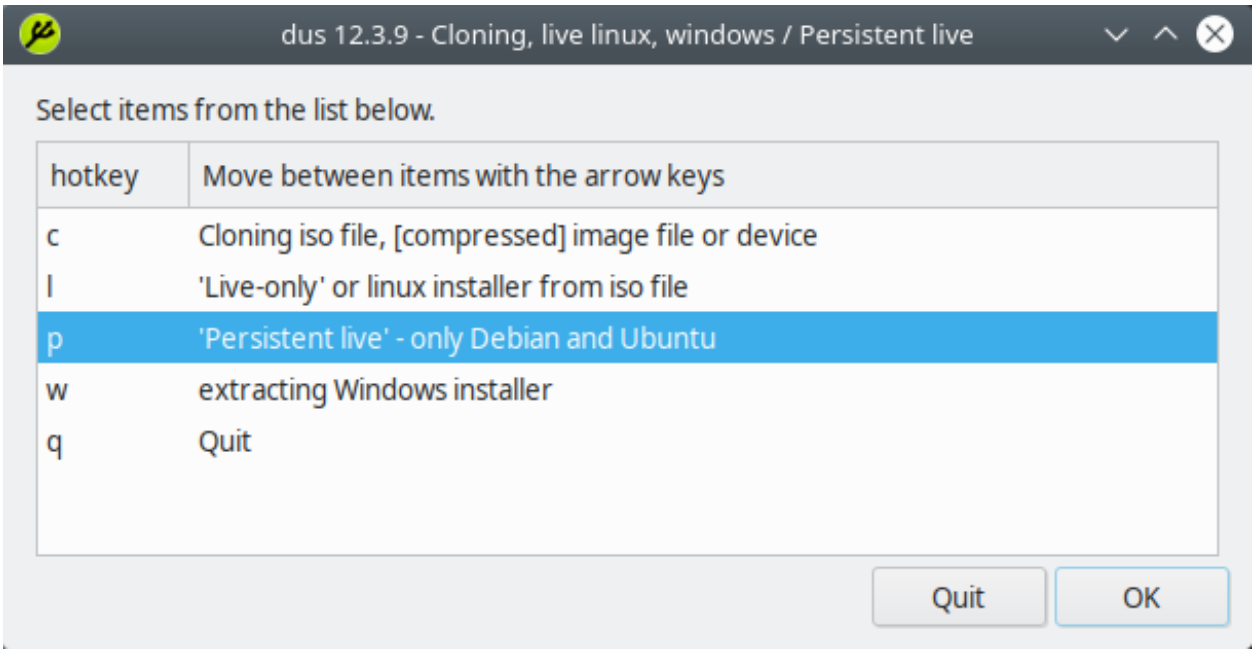


Fig. 5: `mkusb` - lista de acciones 2

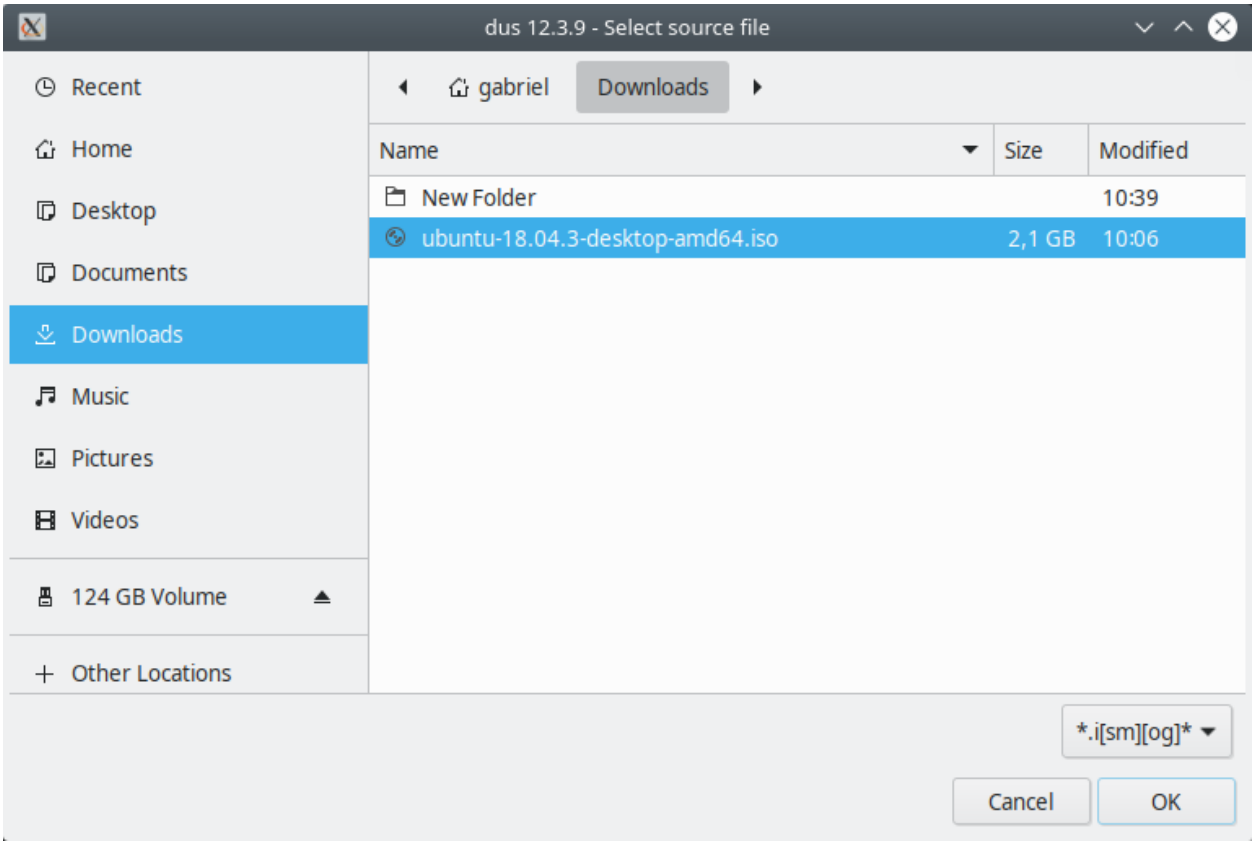


Fig. 6: `mkusb` - seleccionar imagen ISO o IMG

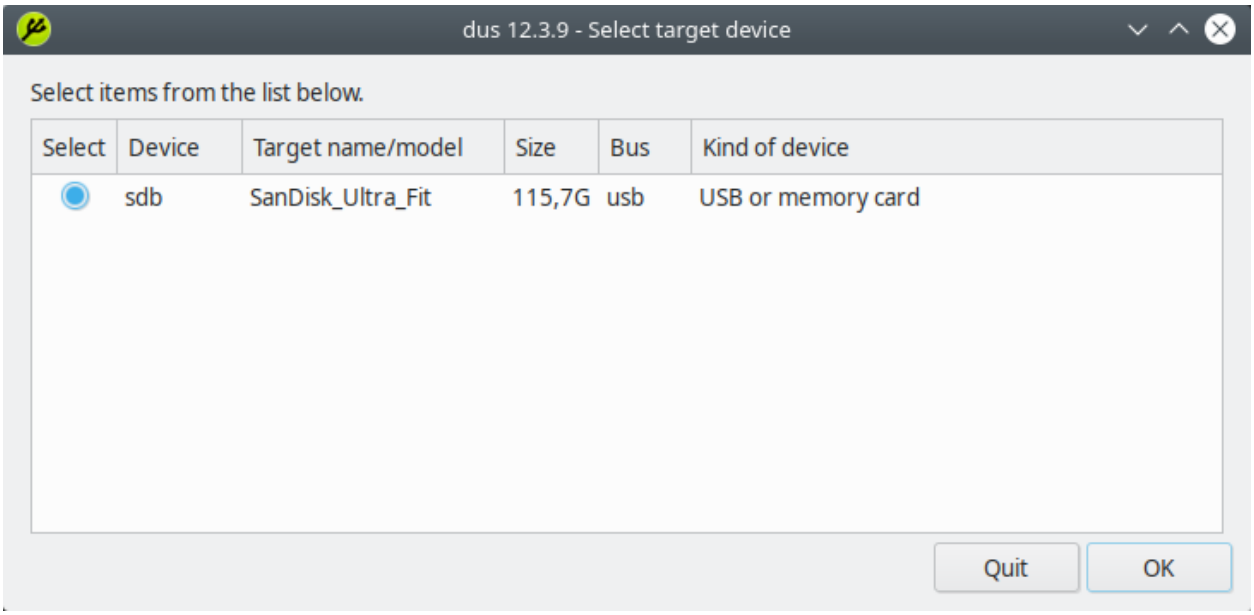


Fig. 7: `mkusb` - seleccionar dispositivo de almacenamiento USB

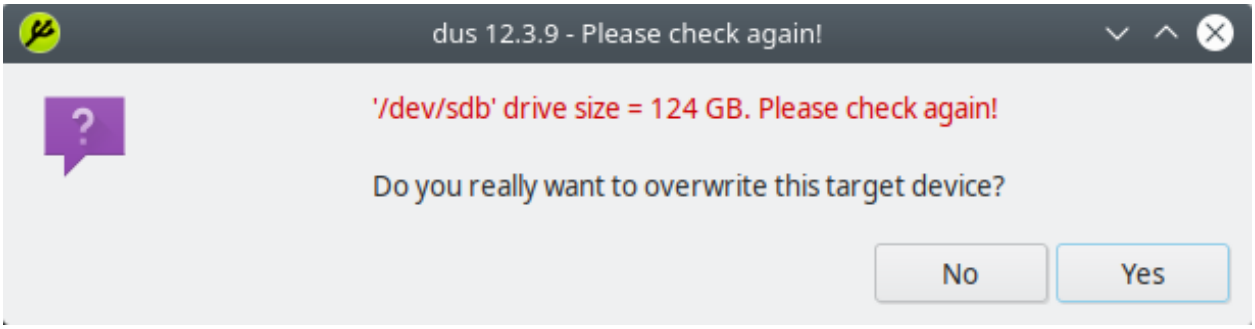


Fig. 8: `mkusb` - confirmar selección de dispositivo

7. Luego se nos presenta una lista de configuración del dispositivo:

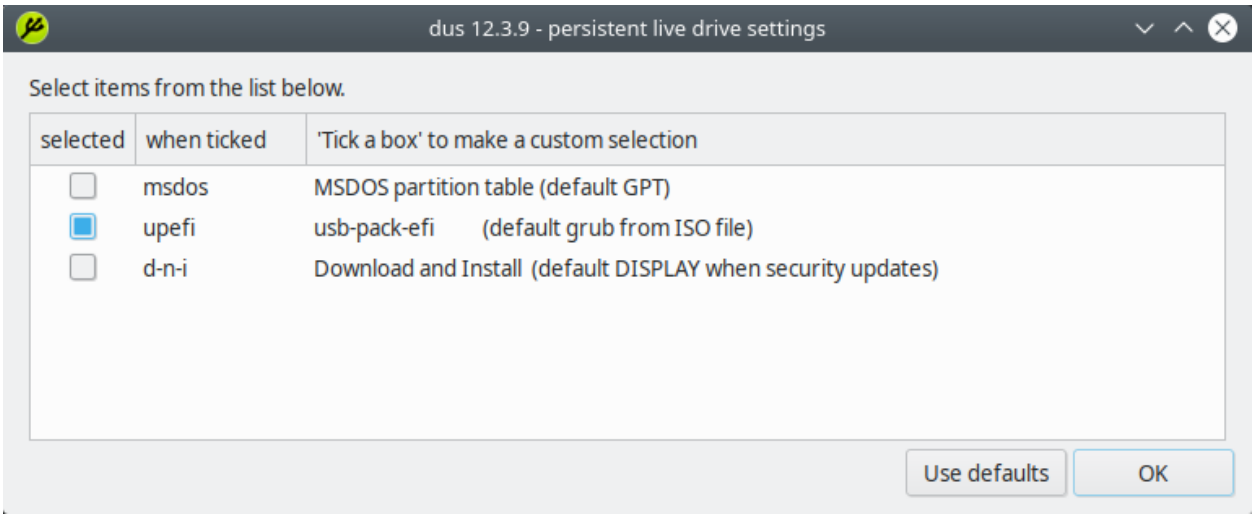


Fig. 9: `mkusb` - opción *upefi*

Seleccionar la opción *upefi* (del paquete `usb-pack-efi` usado por `mkusb`), en el cual Grub puede trabajar en modos BIOS y UEFI.

8. Seleccionar el porcentaje de espacio disponible permanente que será asignado al sistema operativo instalado (partición llamada `casper-rw`):

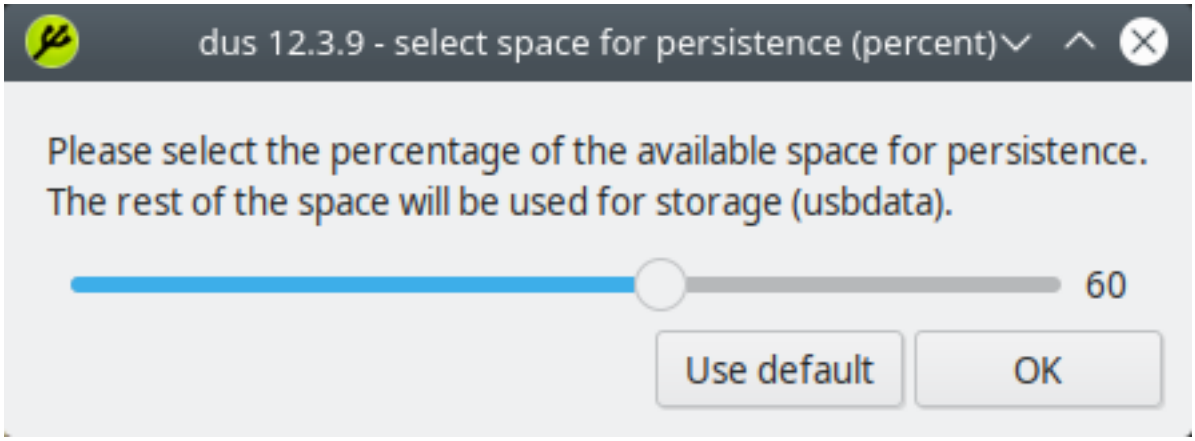


Fig. 10: `mkusb` - porcentaje de almacenamiento permanente

El espacio restante que no sea asignado a la distribución Linux instalada se usará para una partición llamada `usbdata` formateada como NTFS, que puede ser accedida desde cualquier sistema operativo Linux, Windows, macOS. Así, podemos usarlo como un USB de almacenamiento tradicional, guardando datos que sean accesibles desde otros sistemas operativos.

9. En este último paso debemos aceptar la reconfiguración de nuestro USB con los opciones seleccionadas anteriormente:

Si estamos de acuerdo con la configuración seleccionar *Go* para que `mkusb` comience con el proceso de creación del live USB con almacenamiento permanente de Ubuntu, Linux Mint o Debian.

Veremos algunas pantallas de procesos:

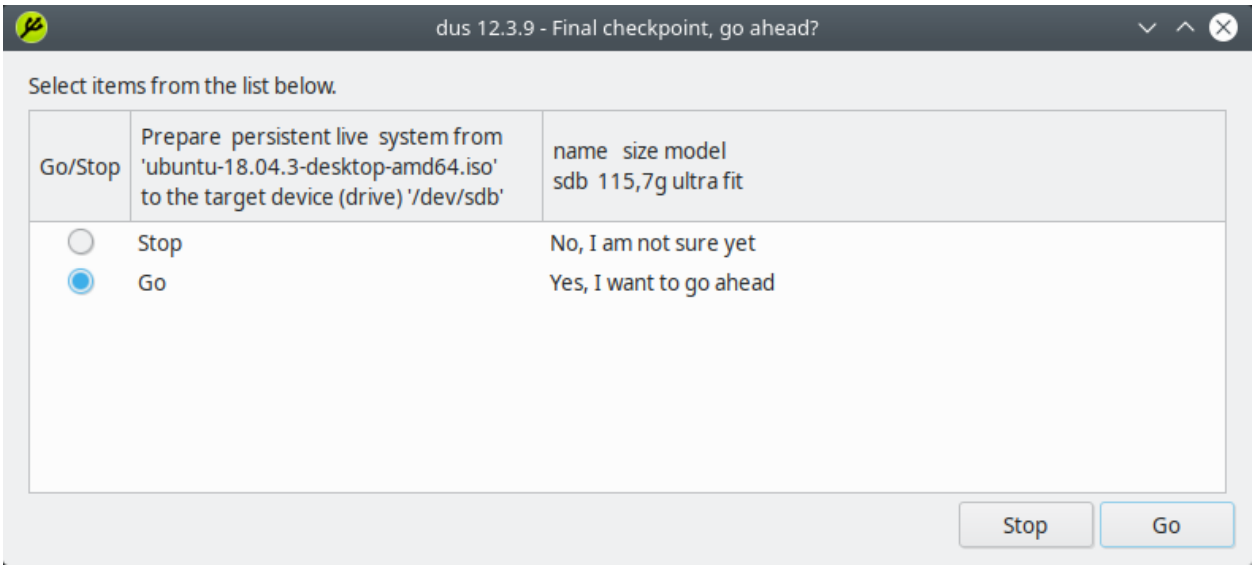


Fig. 11: `mkusb` - aceptar configuración con *Go*

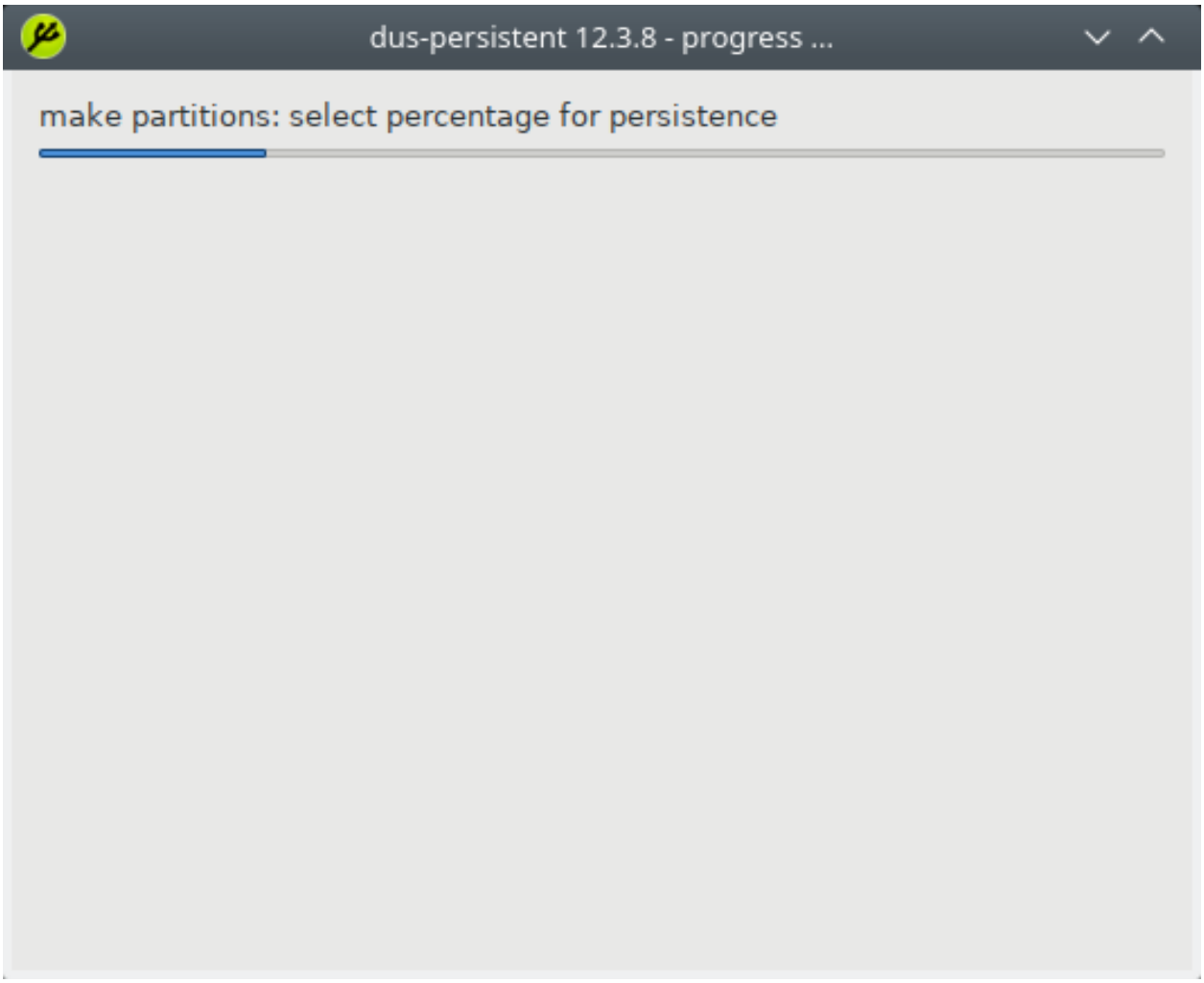


Fig. 12: `mkusb` - proceso

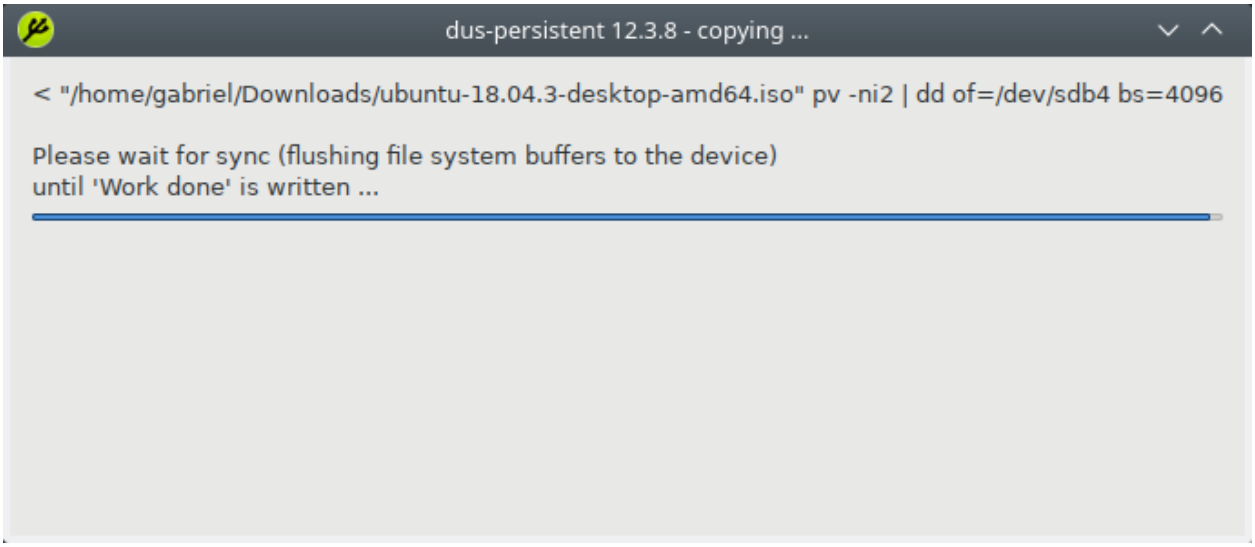


Fig. 13: `mkusb` - proceso

10. Finalmente, si todo ha sido configurado correctamente, obtenedremos una ventana con un mensaje como el siguiente:

Clic en *OK* para acabar con el proceso. Volveremos a un menú de `mkusb`, donde simplemente debemos seleccionar la opción *Quit*:

Y en la ventana con el terminal presionar *Enter* para cerrar la aplicación:

Lo único que resta es conectar nuestro USB a una PC y arrancar el sistema desde este dispositivo.

9.1.3 Referencias

- Create A Persistent Storage Live USB With Ubuntu Linux Mint Or Debian
- Rufus Creating A Persistent Storage Live USB With Ubuntu Or Debian From Windows

9.2 Solucionar error `repository is not valid yet`

Table of Contents

Cuando deseamos actualizar los paquetes podemos obtener el siguiente error: *Release file for [...] is not valid yet (invalid for another [...]). Updates for this repository will not be applied."*

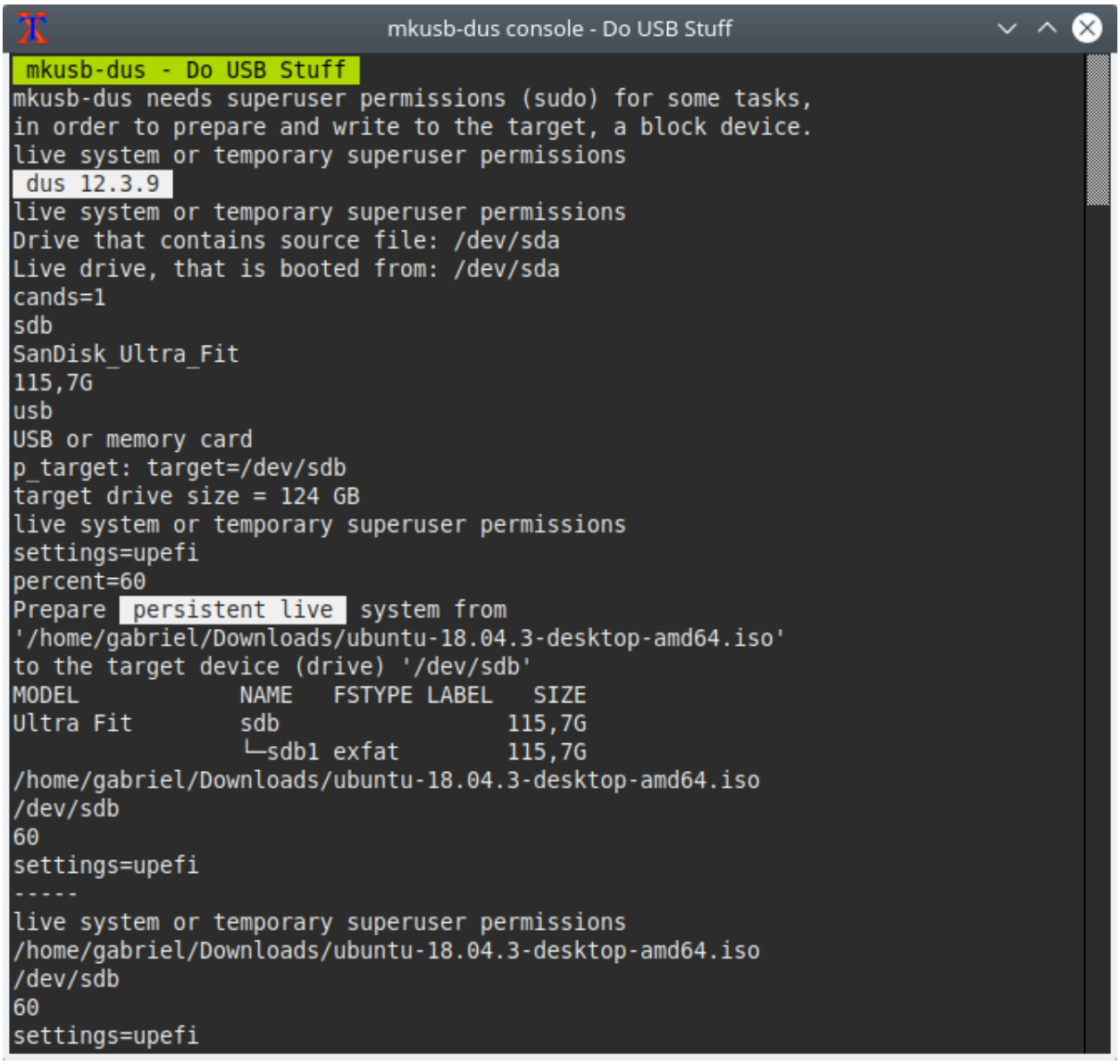


Fig. 14: `mkusb` - proceso

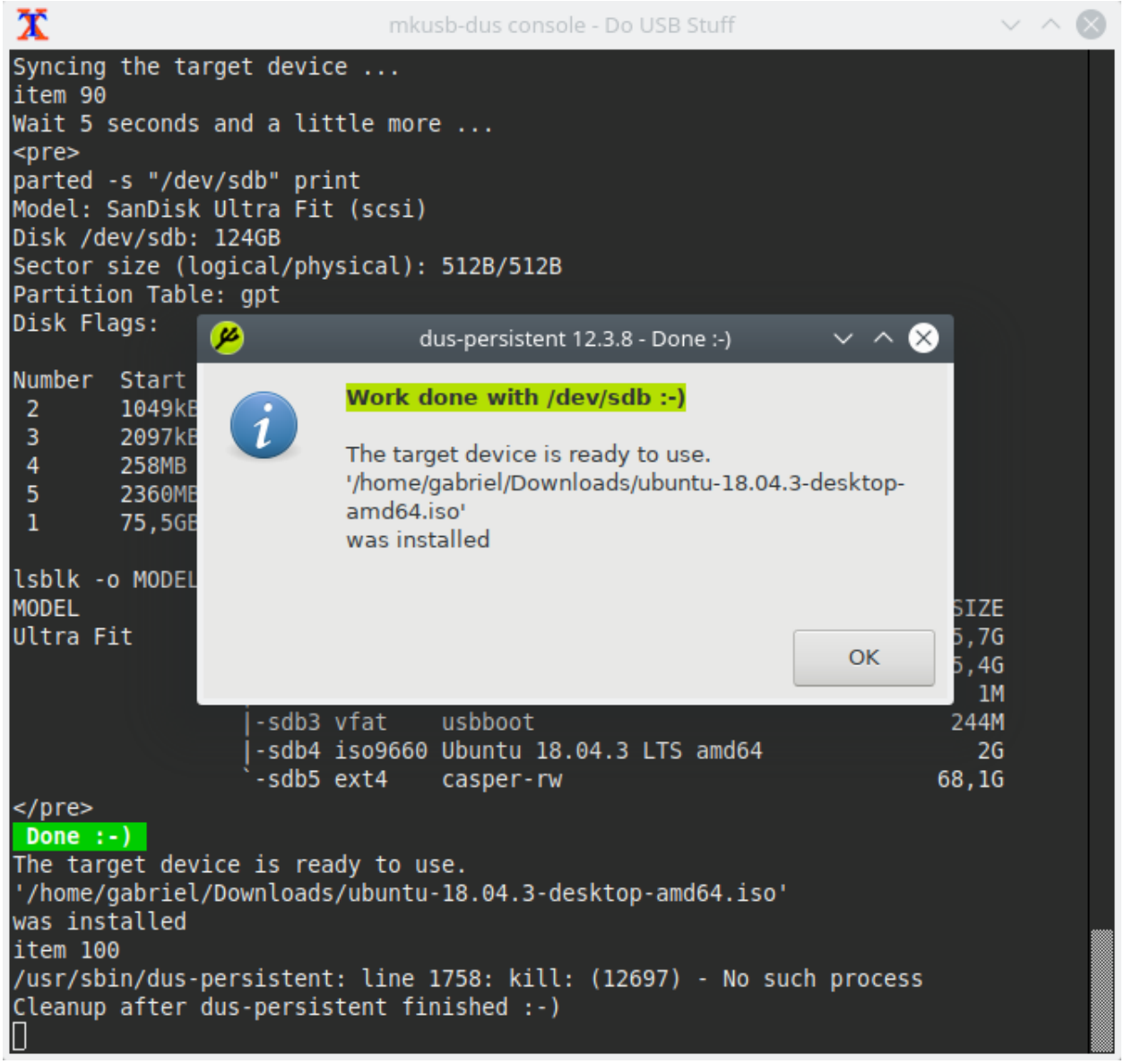


Fig. 15: `mkusb` - pantalla de finalización

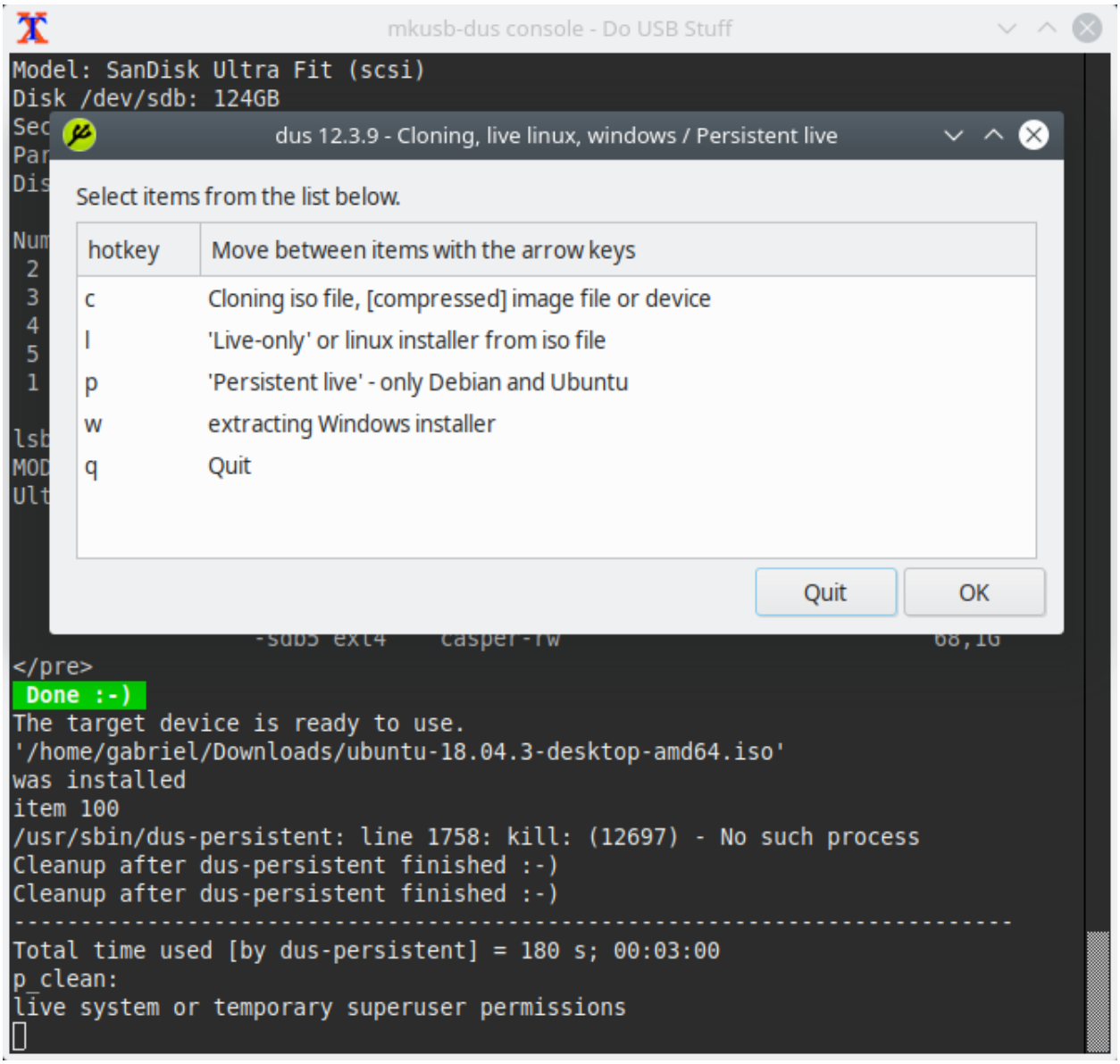


Fig. 16: `mkusb` - opción *Quit*

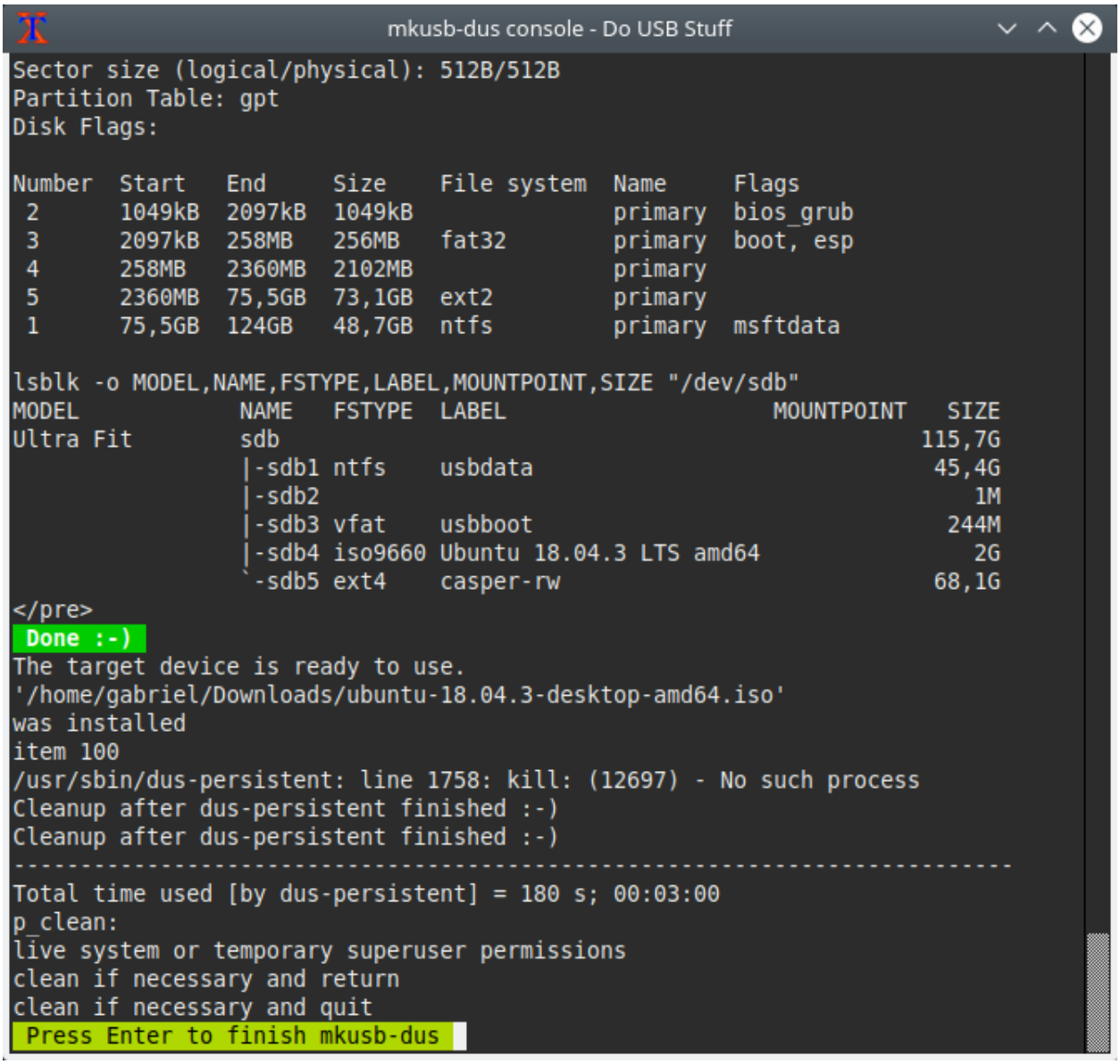


Fig. 17: `mkusb` - presionar *Enter*

9.2.1 Descripción del error

```
$ sudo apt update

Get:1 http://pe.archive.ubuntu.com/ubuntu bionic-updates InRelease [88.7 kB]
E: Release file for http://pe.archive.ubuntu.com/ubuntu/dists/bionic-updates/
↪InRelease is not valid yet (invalid for another 8d 3h 23min 37s). Updates for this␣
↪repository will not be applied.
```

Este error nos indica que el **release file** no es válido aún. La causa de esto es que nuestra hora del sistema está atrasada respecto a la hora del release file. Verifiquemos esto con los siguientes comandos:

- Hora del sistema:

```
$ timedatectl
                      Local time: Sun 2020-03-29 11:30:26 UTC
                  Universal time: Sun 2020-03-29 11:30:26 UTC
                        RTC time: Sun 2020-03-29 10:51:01
                       Time zone: Etc/UTC (UTC, +0000)
       System clock synchronized: no
systemd-timesyncd.service active: yes
                 RTC in local TZ: no
```

- Hora del release file:

```
$ sudo head -14 /var/lib/apt/lists/partial/pe.archive.ubuntu.com_ubuntu_dists_bionic-
↪updates_InRelease

-----BEGIN PGP SIGNED MESSAGE-----
Hash: SHA512

Origin: Ubuntu
Label: Ubuntu
Suite: bionic-updates
Version: 18.04
Codename: bionic
Date: Mon, 06 Apr 2020 14:42:02 UTC
Architectures: amd64 arm64 armhf i386 ppc64el s390x
Components: main restricted universe multiverse
Description: Ubuntu Bionic Updates
MD5Sum:
 e8b27420ceea0481a7c733eca0463406         50695339 Contents-armhf.gz
```

Note: Los release files son encontrados bajo el directorio `/var/lib/apt/lists/partial/`

Como se ve, la hora de nuestro sistema (`Local time:  Sun 2020-03-29 11:30:26 UTC`) está retrasada respecto a la del repository file (`Date:  Mon, 06 Apr 2020 14:42:02 UTC`).

Important: La diferencia de tiempo puede causarse debido a que hemos iniciado nuestro sistema desde un snapshot, haciendo la hora figure sea la hora en que se tomó la snapshot.

9.2.2 Solución del error

Para solucionar el error debemos sincronizar el reloj del sistema con time servers. Con este fin, podemos usar la herramienta `chrony`:

- Instalar `chrony`:

```
$ sudo apt install chrony
```

- Sincronizar el tiempo del sistema:

```
$ sudo chronyd -q

2020-03-29T11:51:27Z chronyd version 3.2 starting (+CMDMON +NTP +REFCLOCK +RTC␣
↪+PRIVDROP +SCFILTER +SECHASH +SIGND +ASYNCDNS +IPV6 -DEBUG)
2020-03-29T11:51:27Z Frequency 1.101 +/- 2.384 ppm read from /var/lib/chrony/chrony.
↪drift
2020-03-29T11:51:35Z System clock wrong by 709923.800611 seconds (step)
2020-04-06T17:03:39Z chronyd exiting
```

Note: Podemos ver cuánto varía el tiempo de nuestro sistema con el del servidor de internet:

```
$ sudo chronyd -Q

2020-04-06T17:04:07Z chronyd version 3.2 starting (+CMDMON +NTP +REFCLOCK +RTC␣
↪+PRIVDROP +SCFILTER +SECHASH +SIGND +ASYNCDNS +IPV6 -DEBUG)
2020-04-06T17:04:07Z Disabled control of system clock
2020-04-06T17:04:07Z Frequency 1.101 +/- 2.384 ppm read from /var/lib/chrony/chrony.
↪drift
2020-04-06T17:04:15Z System clock wrong by -0.005141 seconds (ignored)
2020-04-06T17:04:15Z chronyd exiting
```

9.2.3 Referencias

- How to deal with repository is not valid yet error
- Keep Your Clock Sync with Internet Time Servers in Ubuntu 18.04

CHAPTER 10

Kickstart

CHAPTER 11

Kickstart Methods

11.1 Kickstart Methods - usando Virtualbox

Table of Contents

Descripción de la documentación: Método de instalación de un SO basado en Red Hat Enterprise Linux usando VirtualBox y un archivo ISO como medio de instalación. Este procedimiento sirve como un escenario de pruebas fácil de implementar, con el fin de probar archivos Kickstart que automaticen la instalación del SO.

11.1.1 Creación de VM en VirtualBox

1. En VirtualBox, crear una nueva VM haciendo clic en el botón *New*:

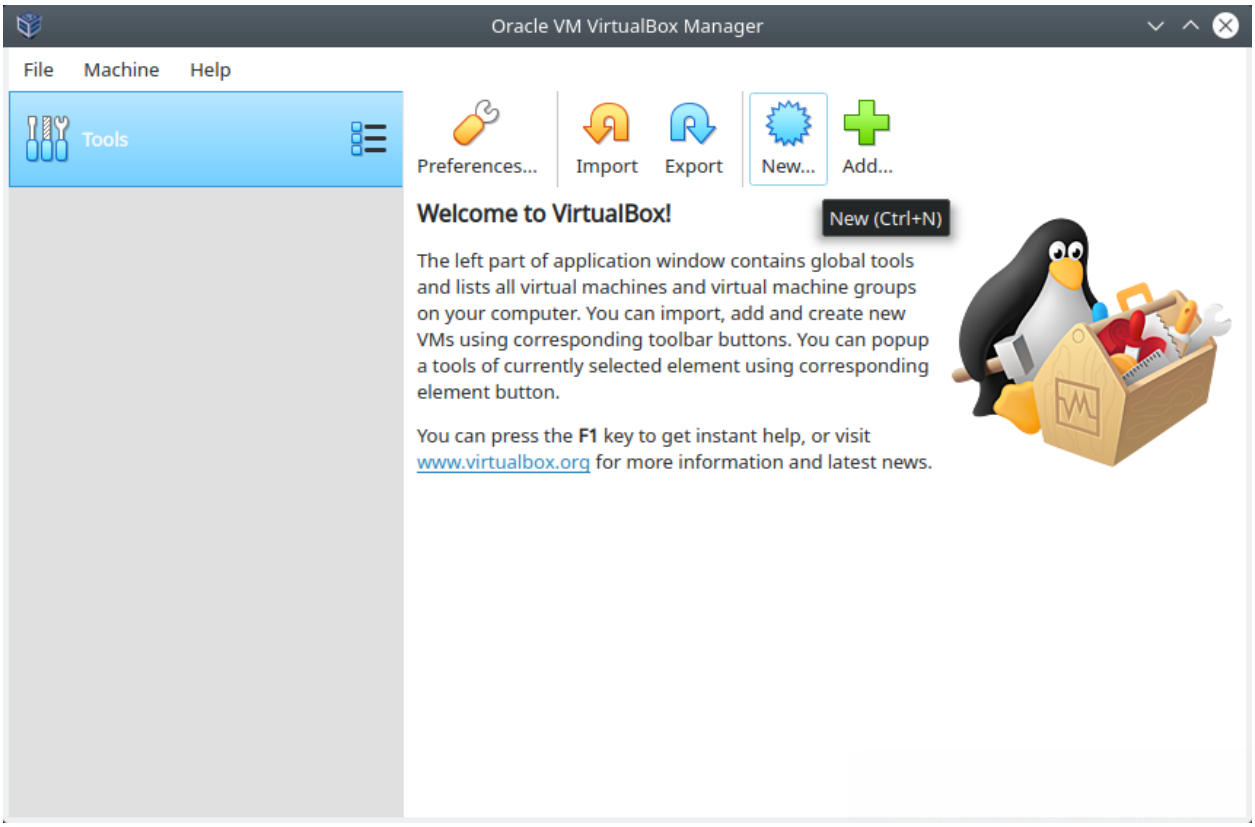


Fig. 1: VirtualBox - Crear una nueva VM

2. Nombrar el nombre de la VM y seleccionar el SO que deseemos instalar; en este caso, es un SO basado em Red Hat:

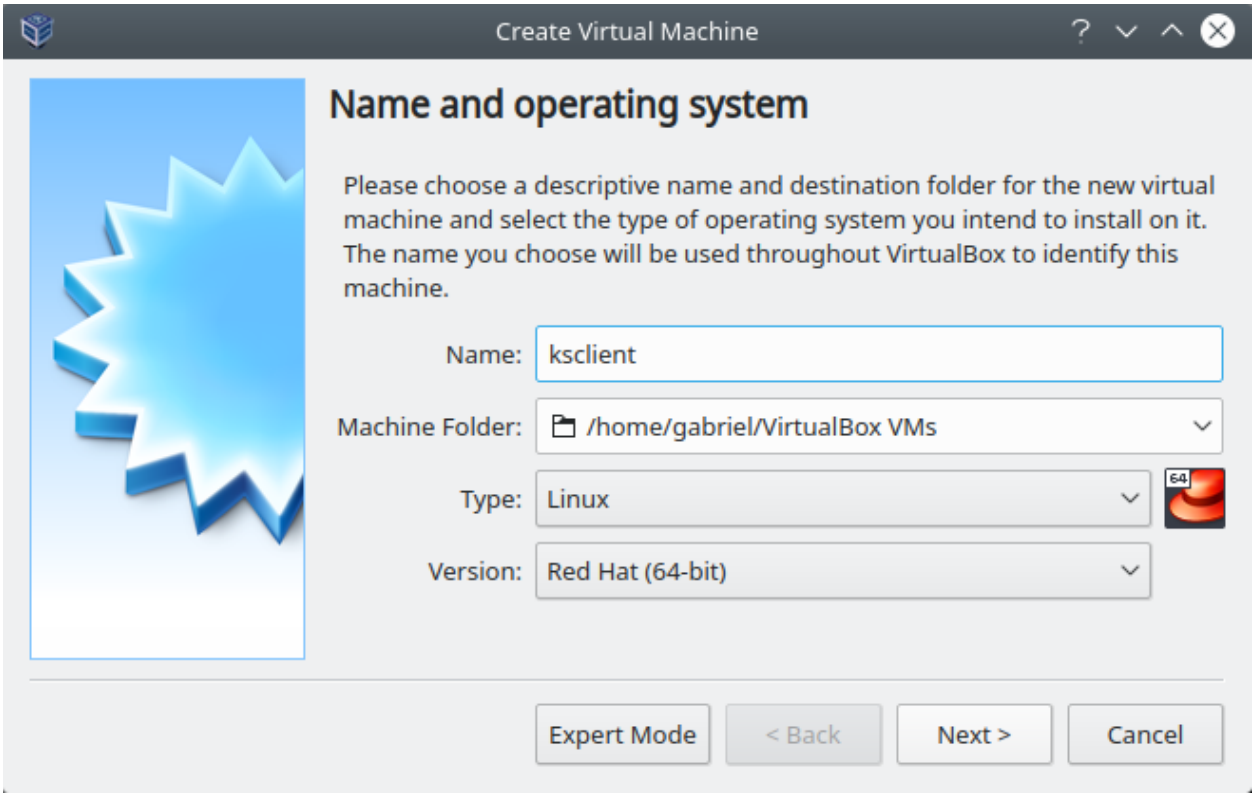


Fig. 2: VirtualBox - Nombrar la VM y seleccionar el SO

3. Elegir la cantidad de memoria RAM que deseemos asignarle a nuestra VM:
4. Seleccionar la opción *Create a virtual hard disk now*:
5. Seleccionar como tipo de disco duro *VDI (VirtualBox Disk Image)*:
6. Para el provisionamiento de espacio en el disco elegir la opción *Dynamically Allocated*:
7. Determinar la ubicación y tamaño del disco:

Finalmente, clic en el botón *Create*.

11.1.2 Configuración de VM en VirtualBox

1. Clic en el botón *Settings* para configurar la VM:
2. Seleccionar la pestaña *System* cambiar el orden de arranque, poniendo primero *Hard Disk* y luego *Optical*:
3. Seleccionar la pestaña *Storage* y clic sobre el ícono de disco con nombre Empty:

El método de instalación del sistema operativo usando en esta guía será a través un archivo ISO de CentOS 7 que contiene el medio de instalación.

Previamente debemos haber descargado el instalador de CentOS 7 en formato `.iso` de alguno de los Mirrors oficiales de imágenes ISO de CentOS 7:

4. Clic el botón con ícono de disco en la parte derecha y seleccionar la opción *Choose Virtual Optical Disk File...*:

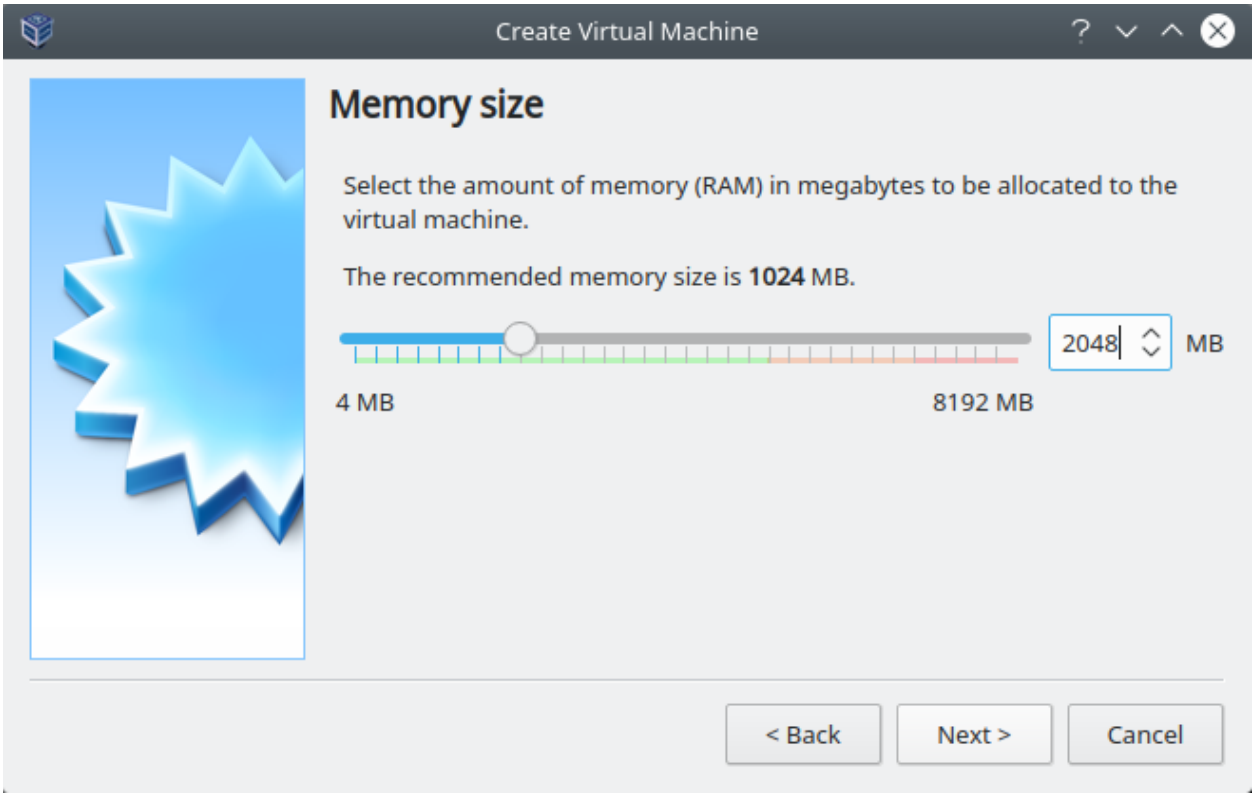


Fig. 3: VirtualBox - Seleccionar la cantidad de RAM

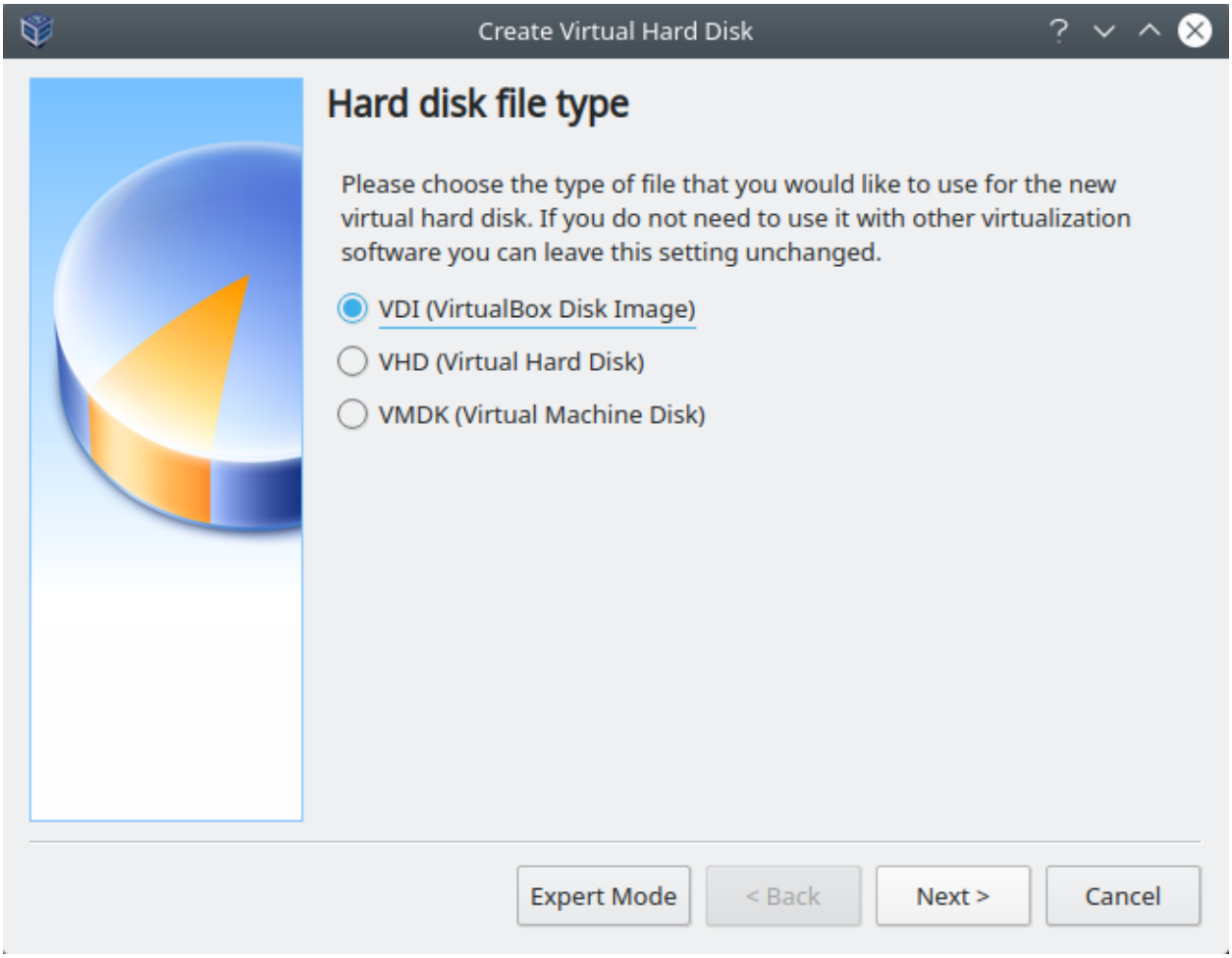


Fig. 4: VirtualBox - Seleccionar el tipo de disco duro que será creado y conectado a la VM

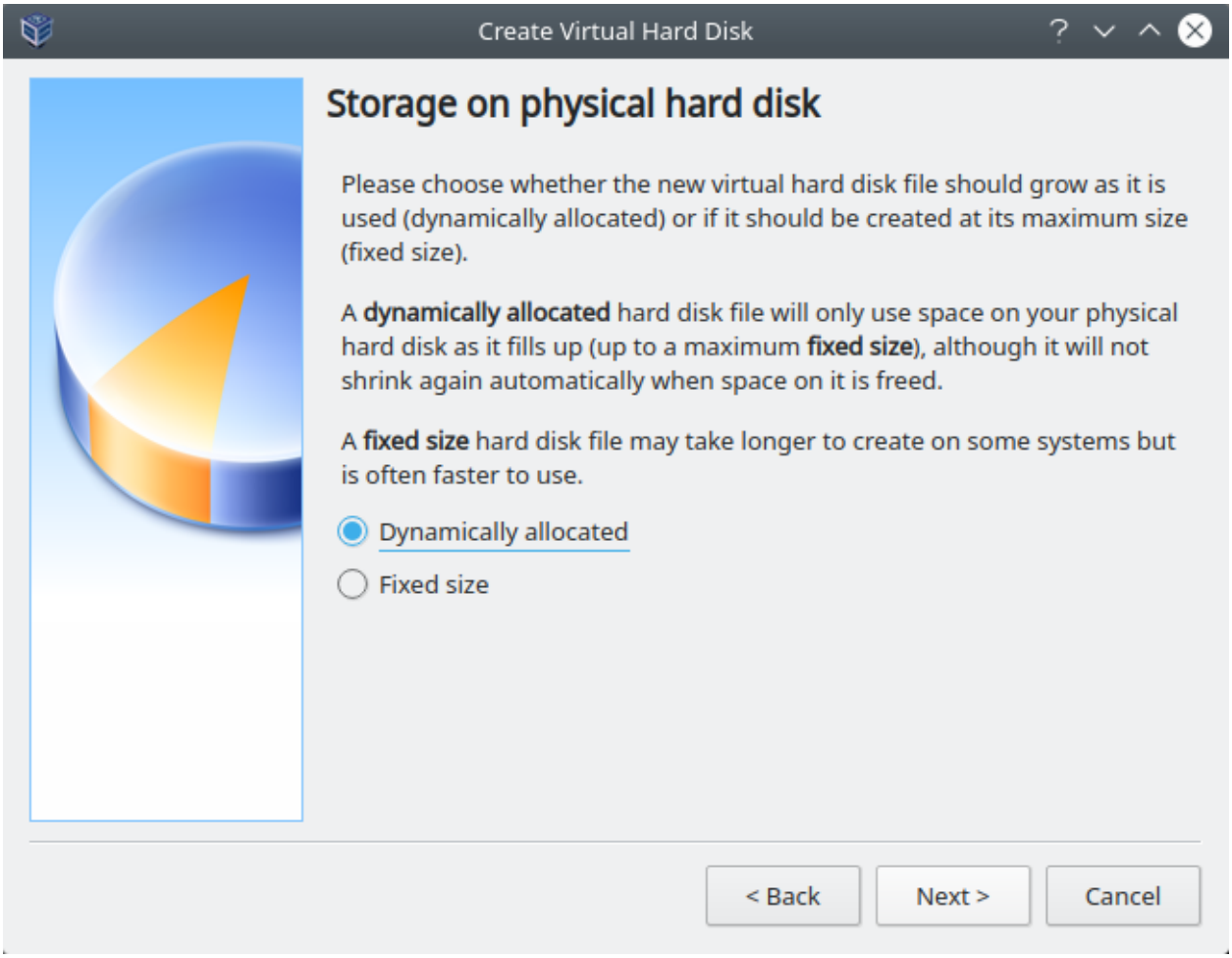


Fig. 5: VirtualBox - Seleccionar *Dynamically Allocated*

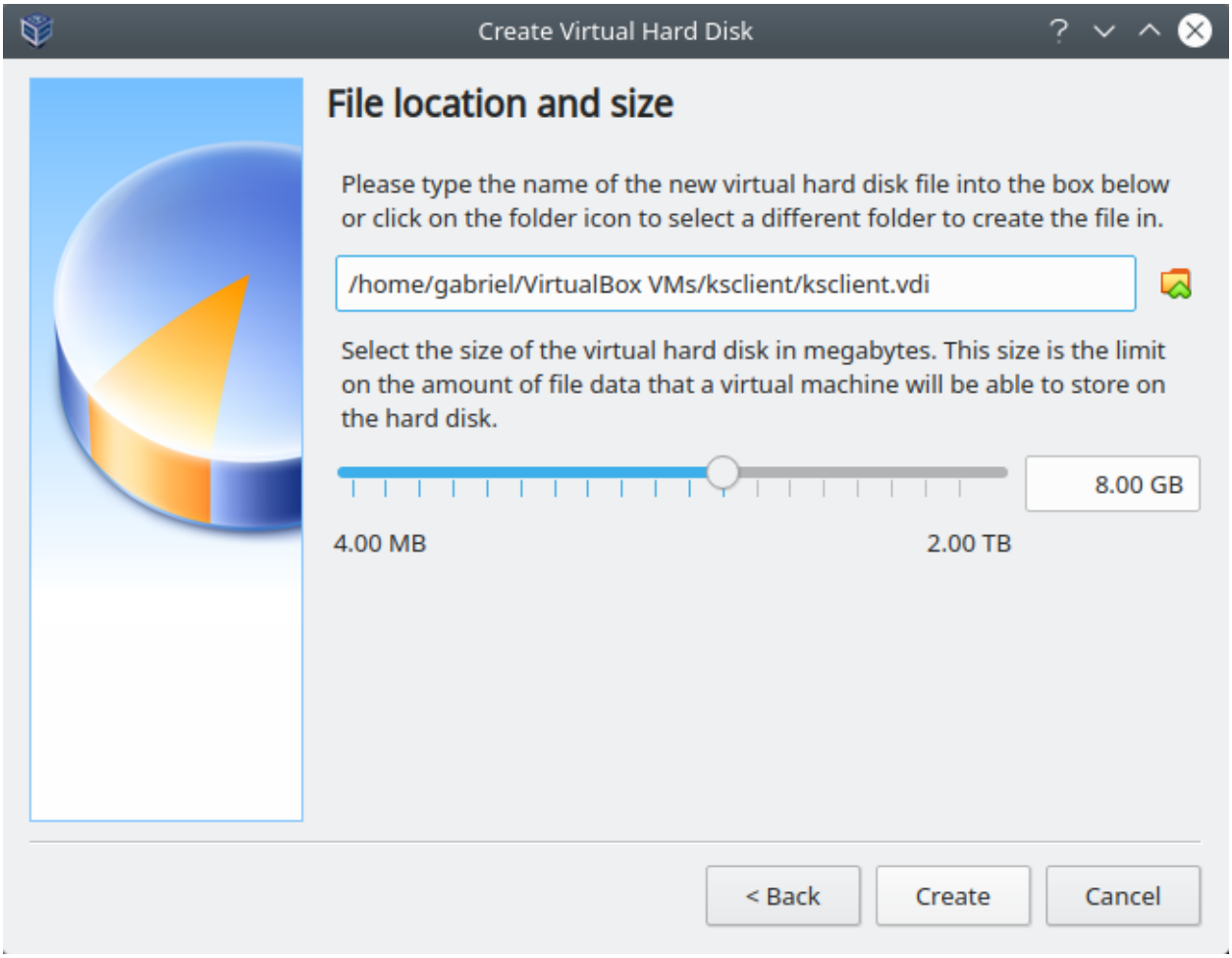


Fig. 6: VirtualBox - Seleccionar la ubicación y tamaño del disco

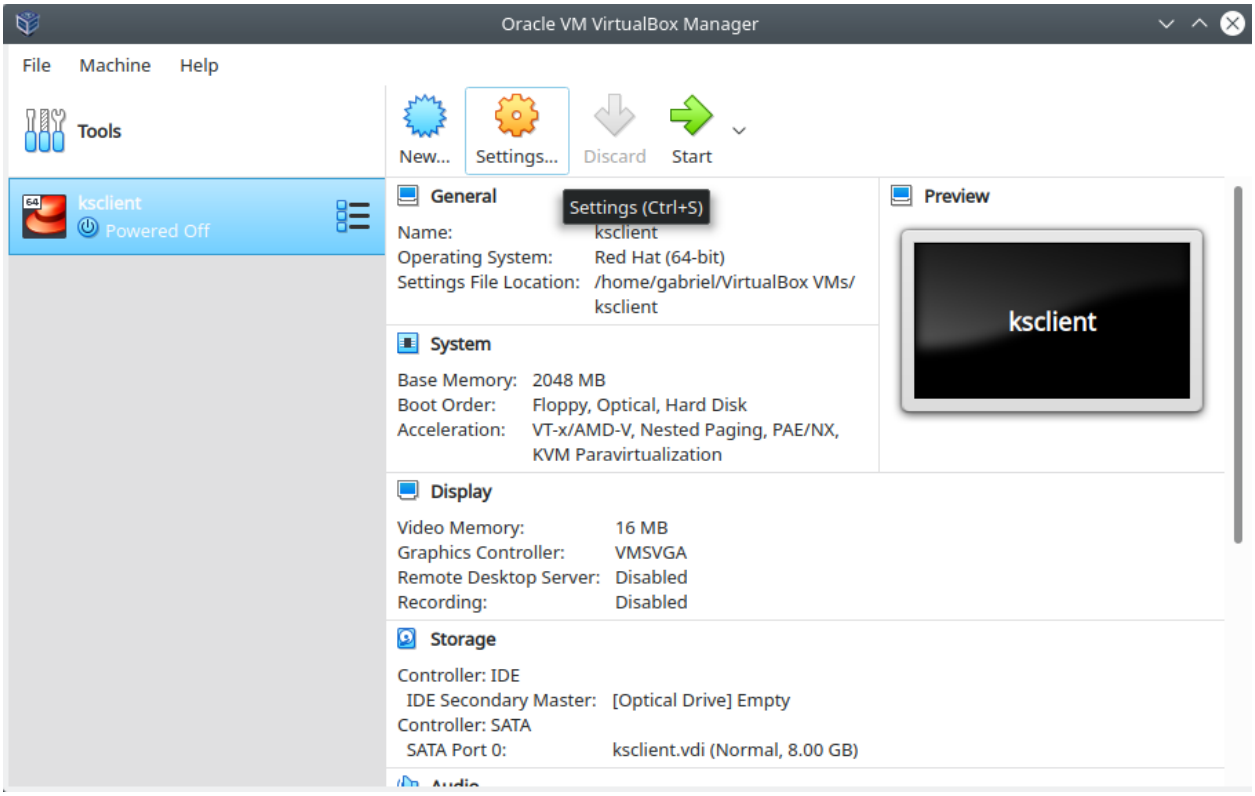


Fig. 7: VirtualBox - Seleccionar la opción *Settings*

5. En el navegador de archivos, elegir el archivo ISO que servirá como medio de instalación del SO:
6. Verificar que el archivo ISO se haya cargado en el controlador IDE. Luego clic en *OK*
7. Iniciar la VM haciendo clic en la opción *Start*:
8. Una vez cargue la página principal del instalador del SO, debemos presionar en el teclado la tecla *Esc* para ingresar a la opciones de *boot*:

Referencia: Editing boot options

9. Verificar que cargue la siguiente pantalla con el prompt de `boot:`:

Important: Pasar el archivo Kickstart por el prompt de `boot:` es una forma equivalente a editar las opciones de arranque del instalador definidas en sus respectivos archivos. Para pasar un archivo Kickstart en el prompt de `boot:` escribiríamos `linux ks=...`. En el archivo de instalación de un sistema **Legacy** usaríamos `ks=...` y en un sistema **UEFI** usaríamos `inst.ks=...`

11.1.3 Formas de pasar el archivo kickstart

Existen distintas formas de pasar el archivo Kickstart al medio de instalación con el fin de automatizar el proceso de instalación.

En el prompt de `boot:` podemos emplear el parámetro `ks=` e indicar a través de qué medio se pasará el archivo kickstart. Entre los métodos soportados para pasar el archivo kickstart están NFS (`nfs`), HTTP (`http`), Diskette (`floppy`), disco montado (`hd`), file system sin montar (`file`) y CD-ROM (`cdrom`).

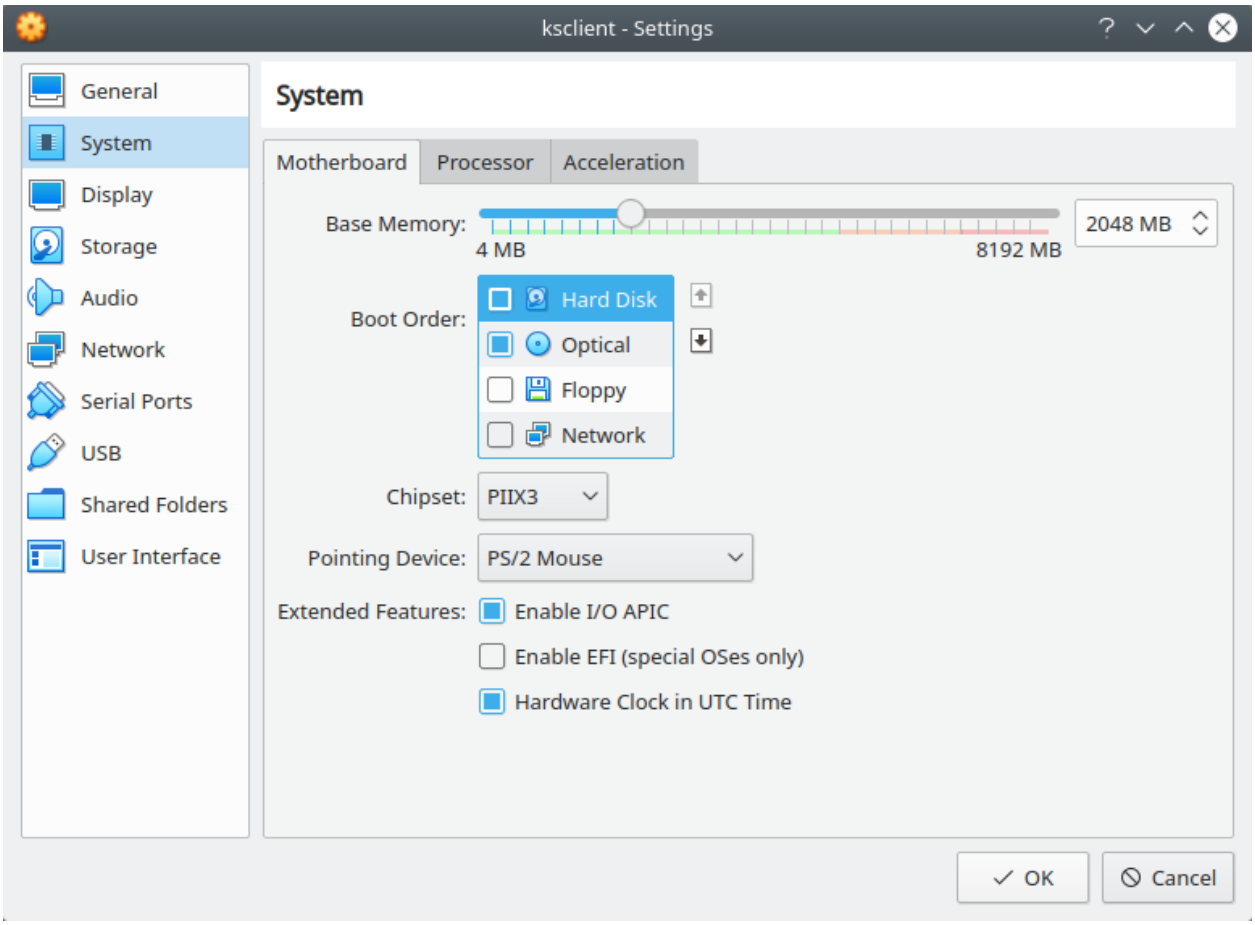


Fig. 8: VirtualBox - Cambiar el orden de arranque del sistema en la pestaña *System*

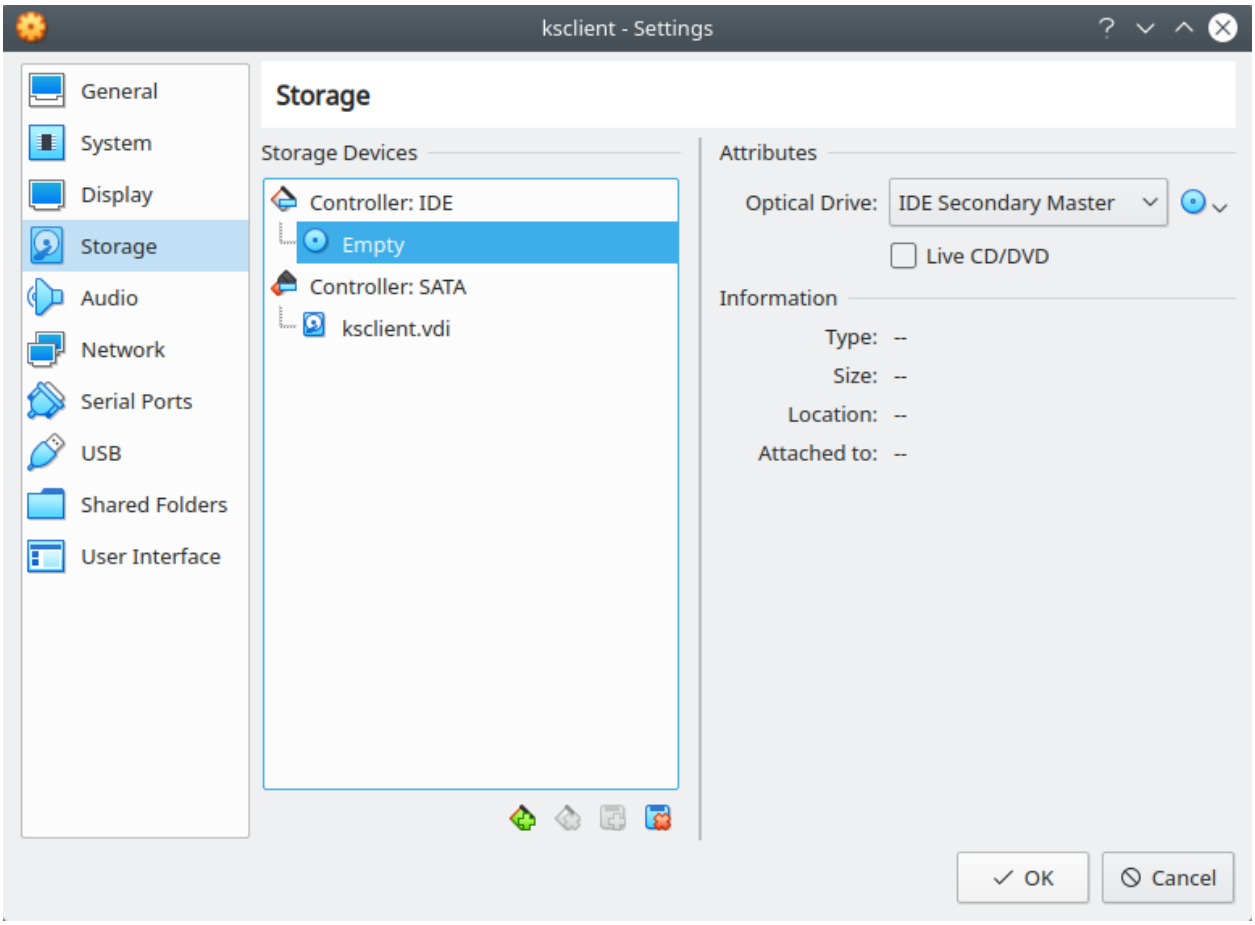


Fig. 9: VirtualBox - Cambiar a la pestaña *Storage*

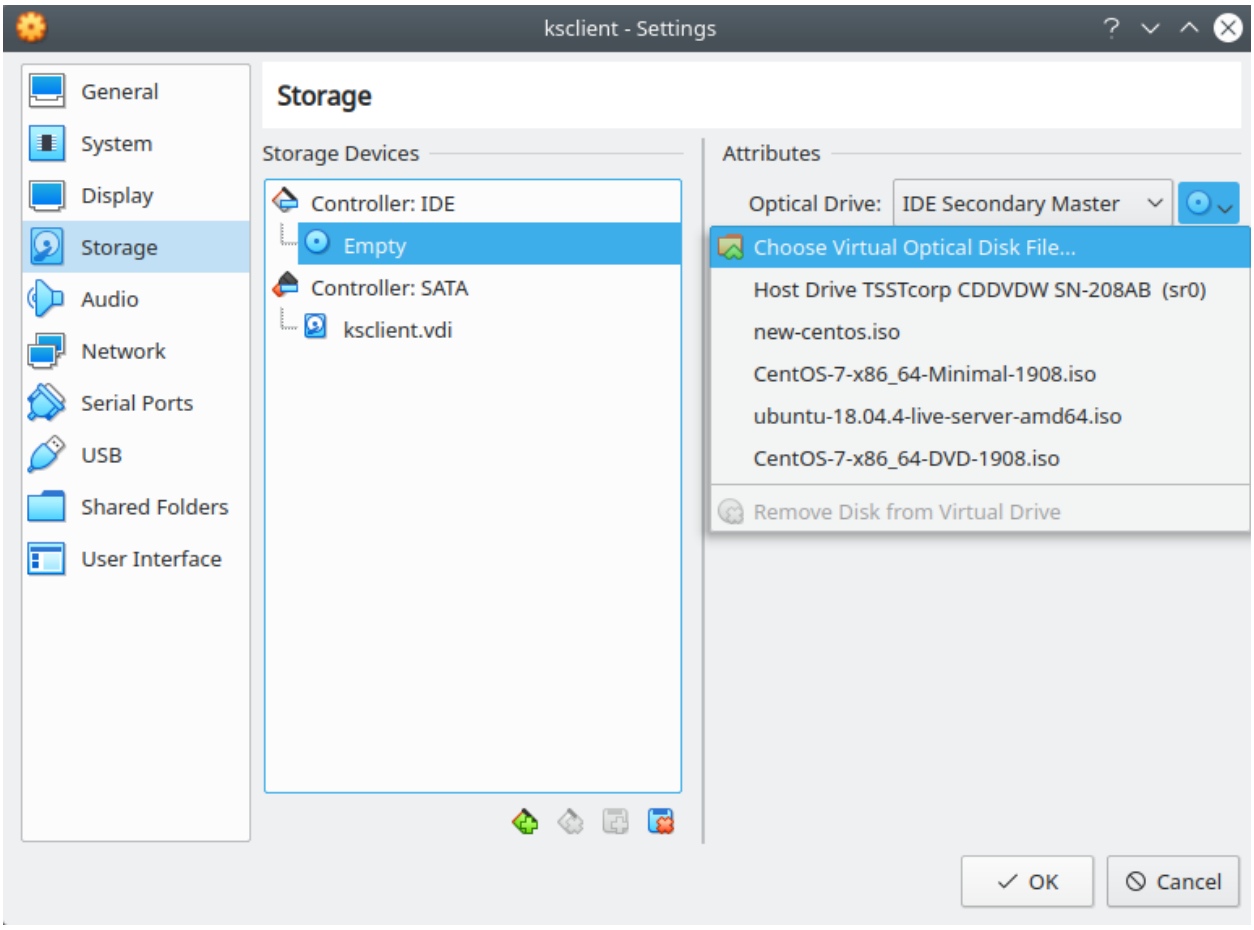


Fig. 10: VirtualBox - Seleccionar el ícono del disco y elegir *Choose Virtual Optical Disk File...*

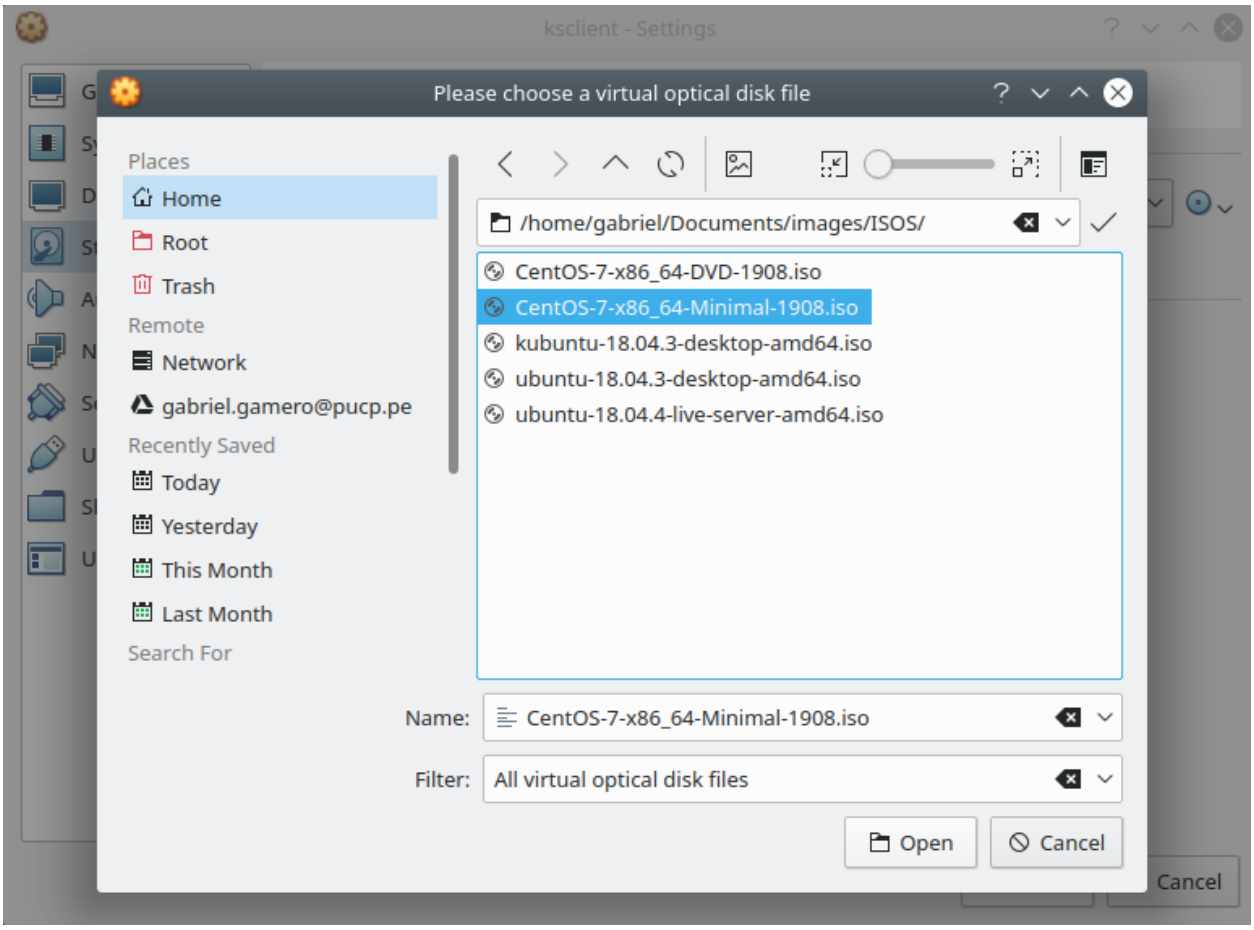


Fig. 11: VirtualBox - Elegir el archivo `.iso` del instalador del SO desde el navegador de archivos

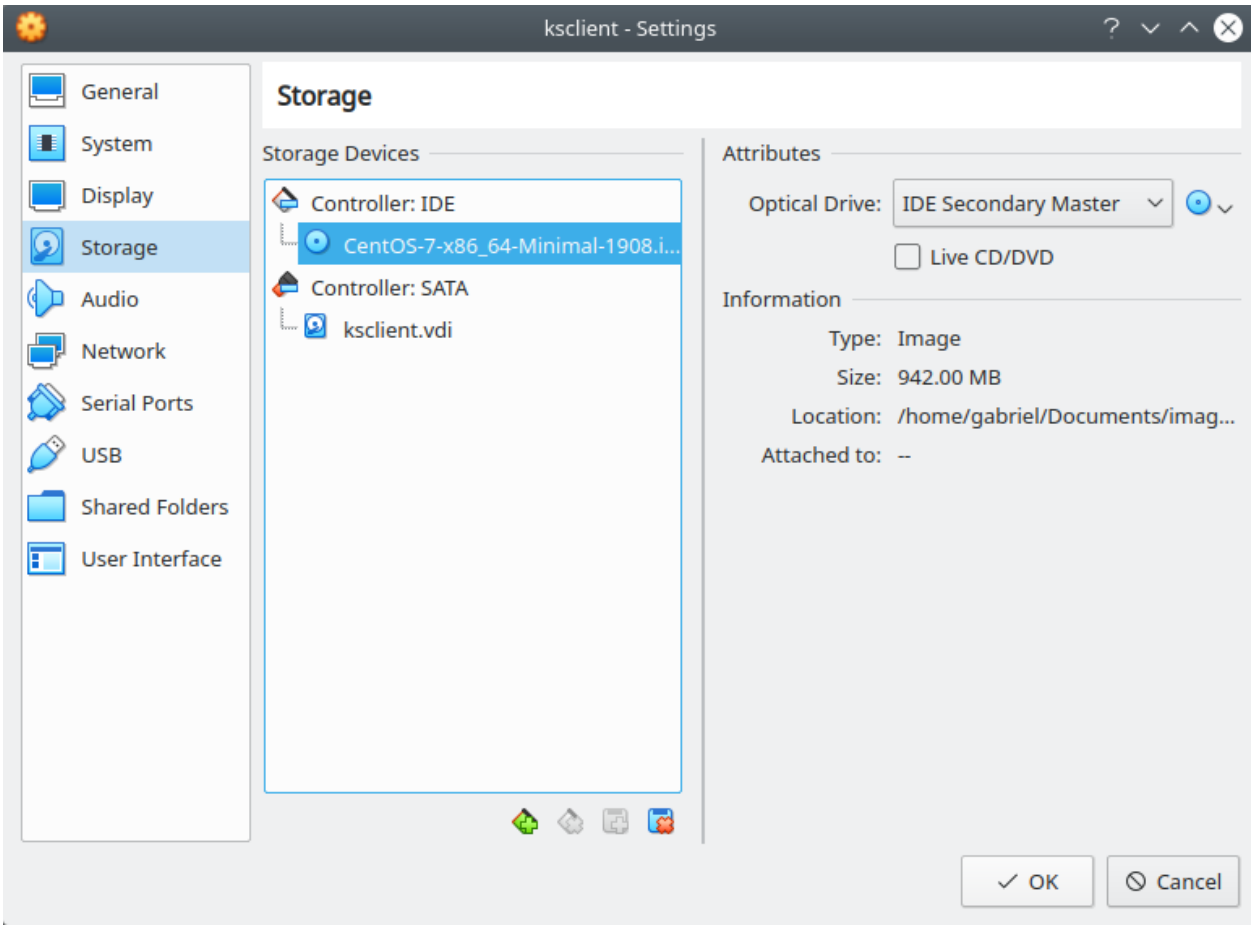


Fig. 12: VirtualBox - Revisar la configuración de Storage y clic en *OK*

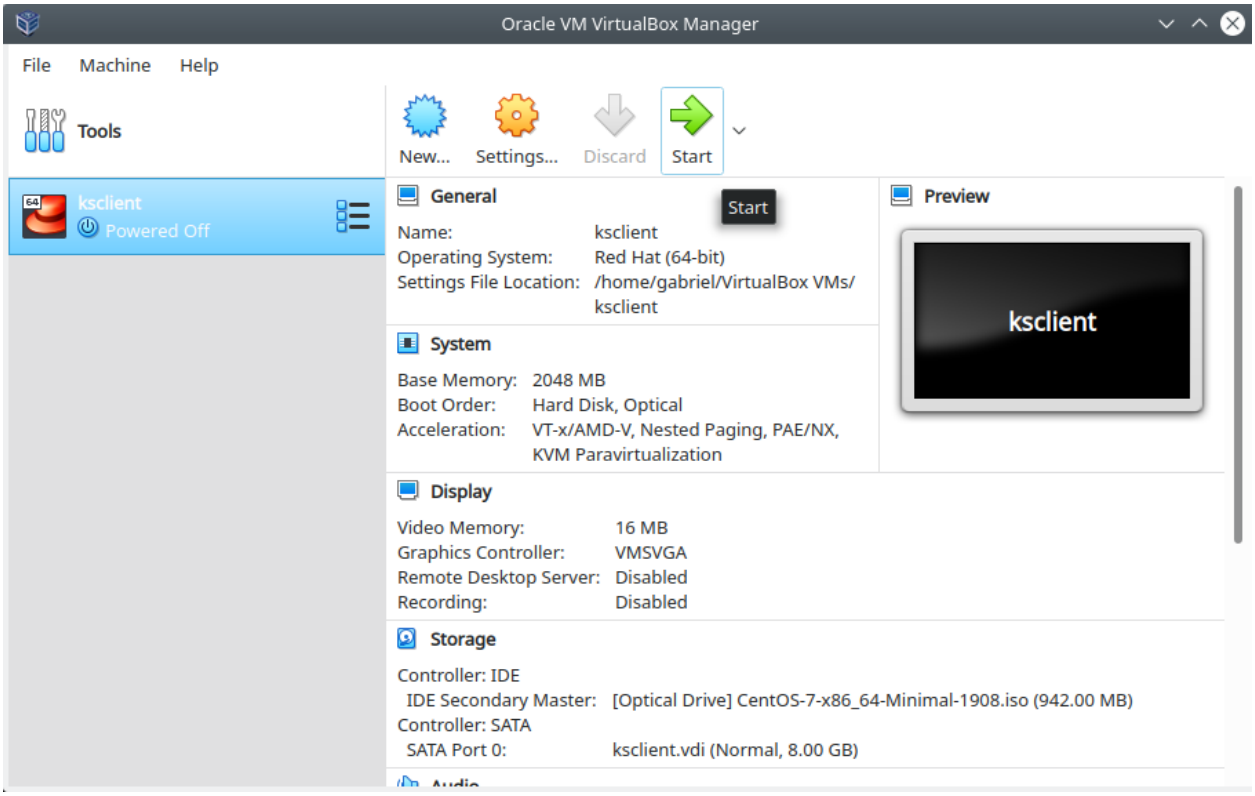


Fig. 13: VirtualBox - Arrancar la VM con el botón *Start*

Referencia 1: Starting a Kickstart installation

Referencia 2: Fuentes de archivo kickstart

Archivo Kickstart pasado por HTTP

Note: Este método es recomendable usarlo cuando el servidor al cual vamos a instalar el sistema operativo tiene un servidor DHCP con una configuración para salir a Internet. Esto, con el fin de obtener el archivo Kickstart de forma remota.

En este método obtenemos el archivo kickstart de un servidor a través de HTTP. Para esto, pasamos la URL de la página que tiene el archivo kickstart como en el siguiente ejemplo:

```
boot: linux ks=http://raw.githubusercontent.com/mogago/kickstart/master/ks1.cfg
```

Important: La página que contiene el archivo kickstart no debe tener ningún tipo de etiquetas HTML ni formatos extra, solo debe contener el archivo kickstart en texto plano. Comprobar esto usando `wget` con la página web en un terminal de comandos.

Note: Opcionalmente, podemos usar `ks=https:` en lugar de `ks=http:`, si la página tiene habilitado el protocolo HTTPS o en caso el servidor web no redirija automáticamente las solicitudes HTTP a HTTPS.

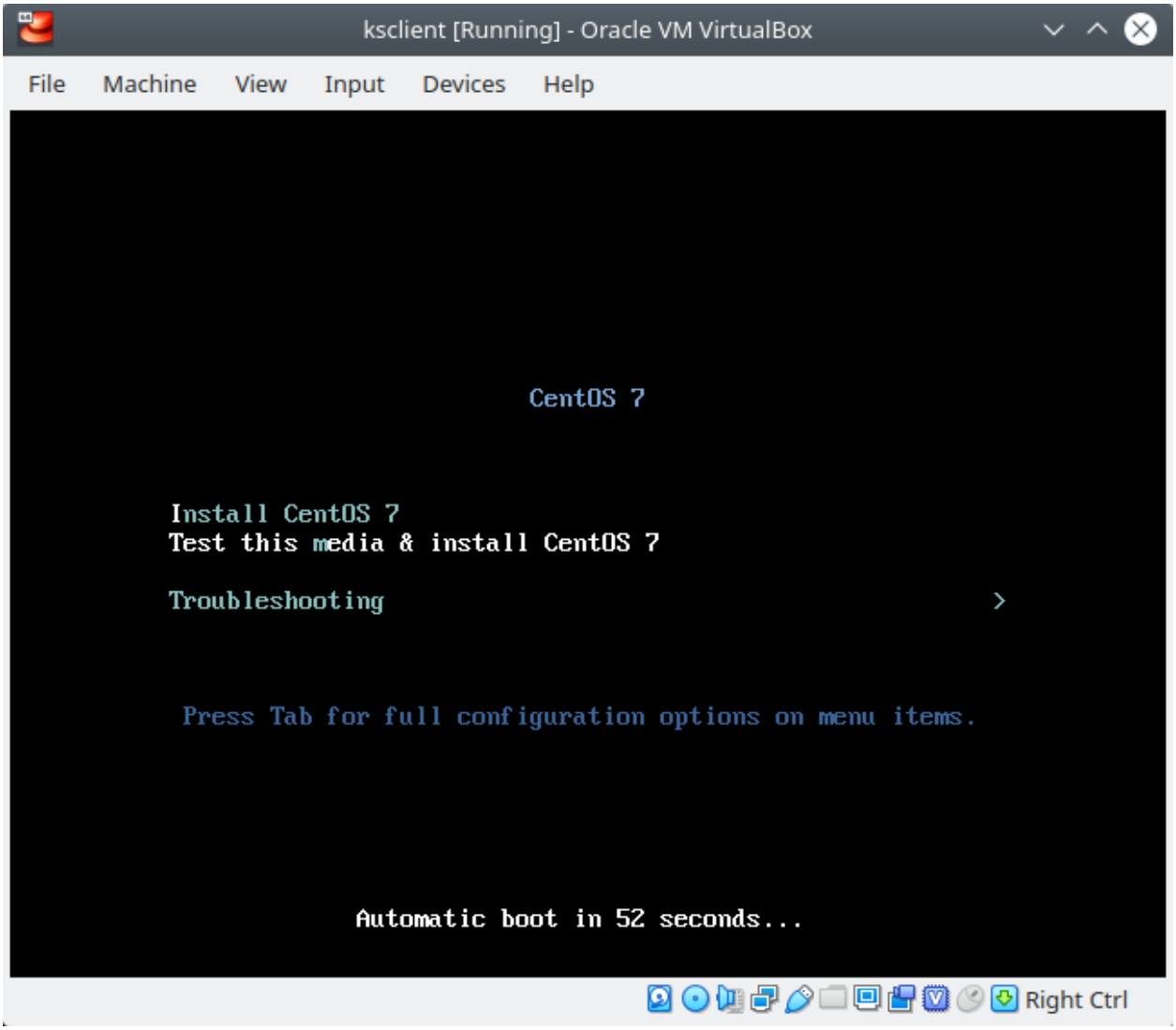


Fig. 14: Centos 7 VM - En la página principal de carga para instalar el SO, presionar la tecla *Esc*

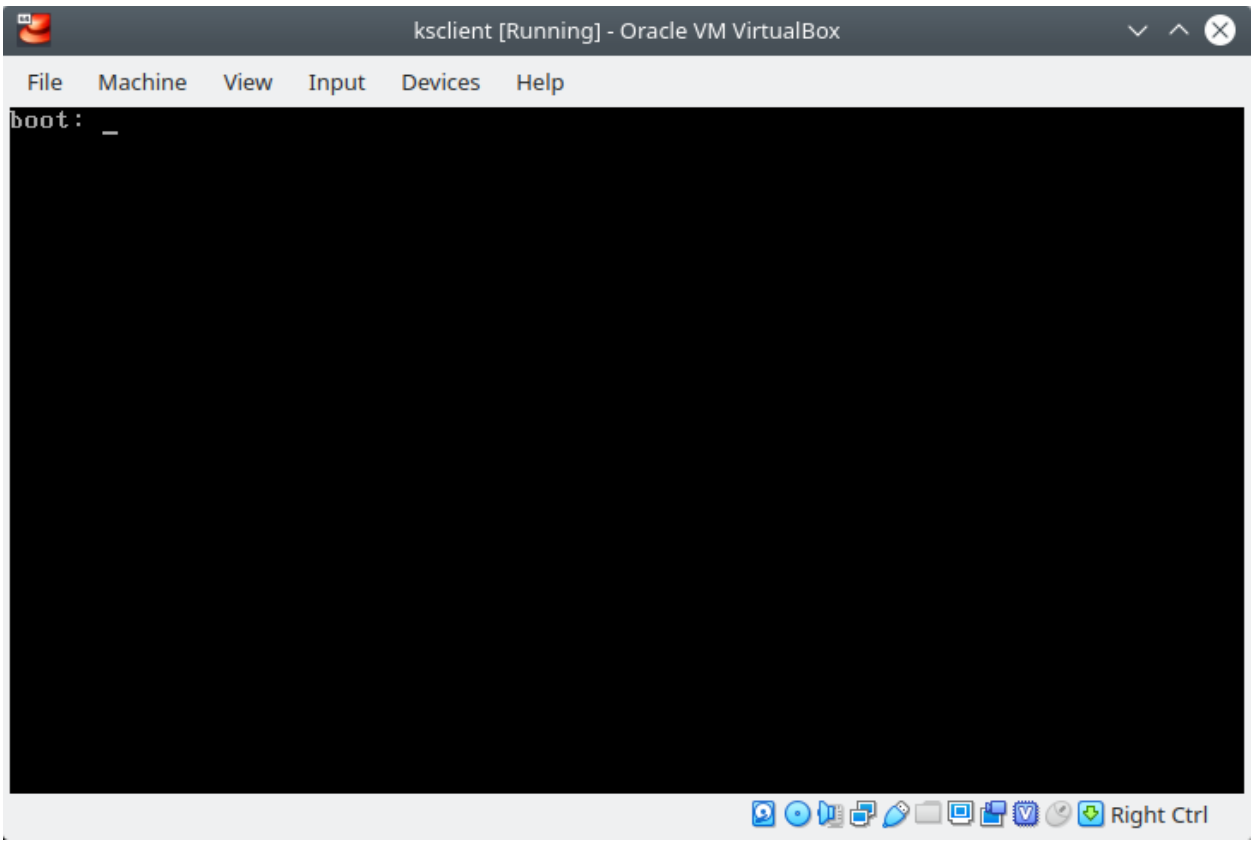


Fig. 15: Centos 7 VM - Cargará una página con el mensaje `boot` :

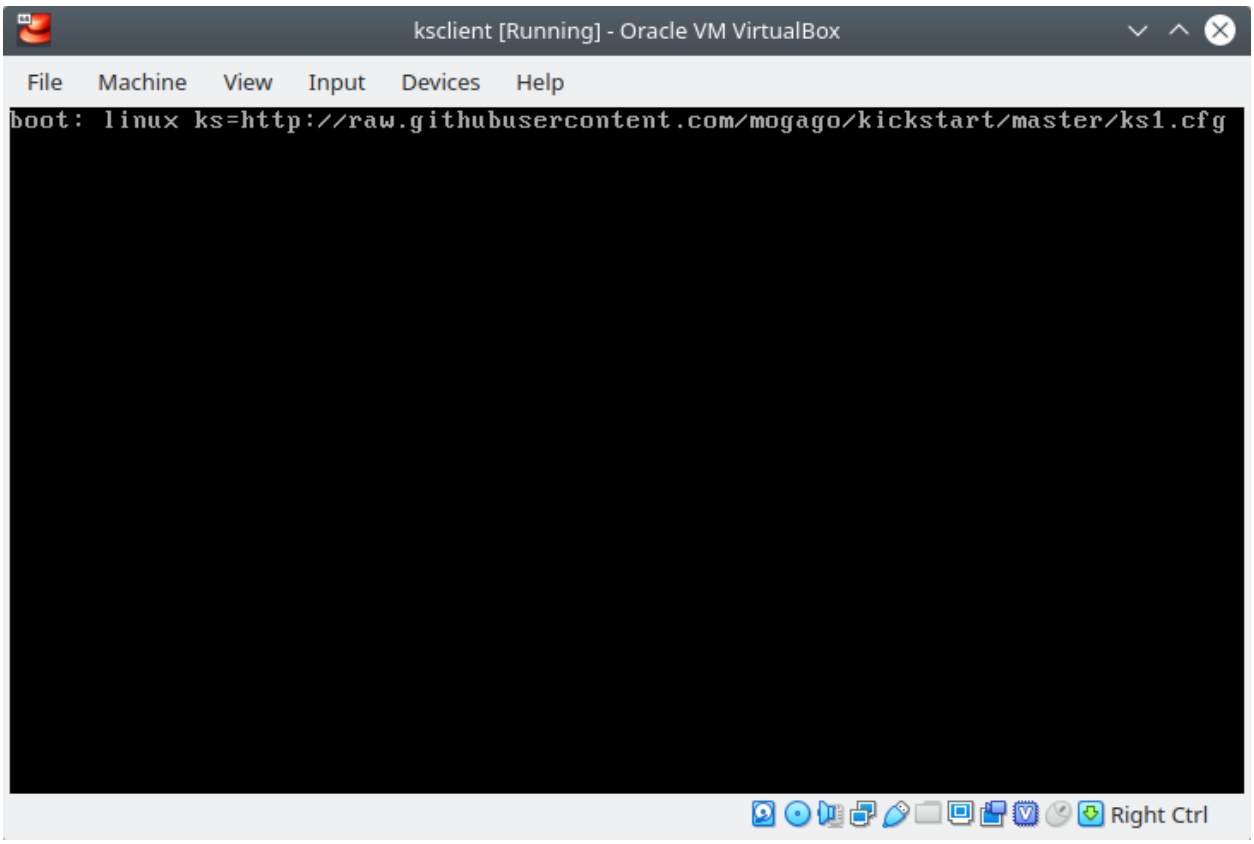
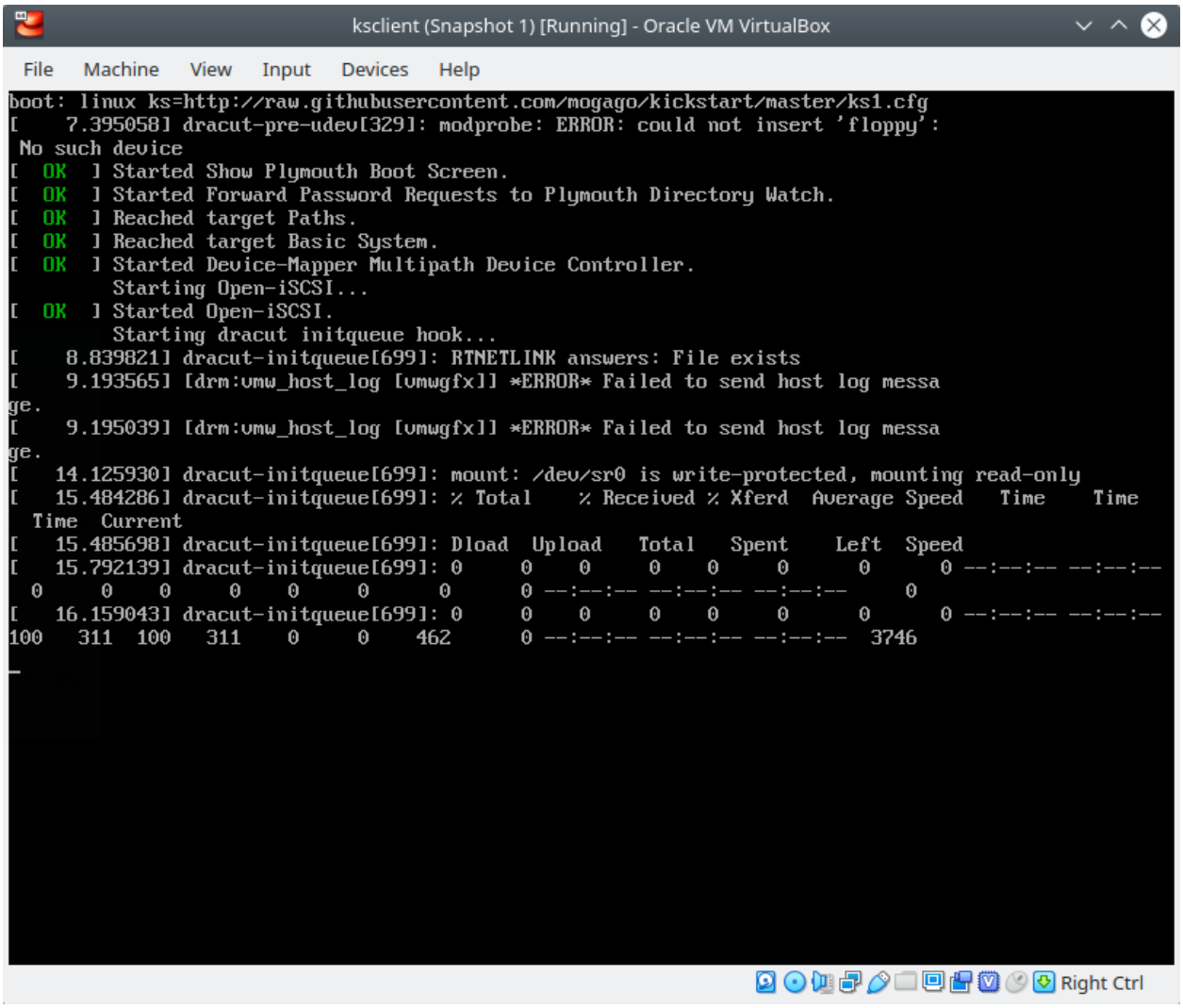


Fig. 16: Centos 7 VM - Ingresar la dirección URL correcta para obtener el archivo kickstart


ksclient (Snapshot 1) [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help
boot: linux ks=http://raw.githubusercontent.com/mogago/kickstart/master/ks1.cfg
[7.395058] dracut-pre-udev[329]: modprobe: ERROR: could not insert 'floppy':
No such device
[OK] Started Show Plymouth Boot Screen.
[OK] Started Forward Password Requests to Plymouth Directory Watch.
[OK] Reached target Paths.
[OK] Reached target Basic System.
[OK] Started Device-Mapper Multipath Device Controller.
Starting Open-iSCSI...
[OK] Started Open-iSCSI.
Starting dracut initqueue hook...
[8.839821] dracut-initqueue[699]: RTNETLINK answers: File exists
[9.193565] [drm:vmw_host_log [vmwgfx]] *ERROR* Failed to send host log messa
ge.
[9.195039] [drm:vmw_host_log [vmwgfx]] *ERROR* Failed to send host log messa
ge.
[14.125930] dracut-initqueue[699]: mount: /dev/sr0 is write-protected, mounting read-only
[15.484286] dracut-initqueue[699]: % Total % Received % Xferd Average Speed Time Time
Time Current
[15.485698] dracut-initqueue[699]: Dload Upload Total Spent Left Speed
[15.792139] dracut-initqueue[699]: 0 0 0 0 0 0 0 0 --:--:-- --:--:--
0 0 0 0 0 0 0 0 --:--:-- --:--:-- --:--:-- 0
[16.159043] dracut-initqueue[699]: 0 0 0 0 0 0 0 0 --:--:-- --:--:--
100 311 100 311 0 0 462 0 --:--:-- --:--:-- --:--:-- 3746
Right Ctrl

En este ejemplo se está usando GitHub como repositorio de los archivos Kickstart:

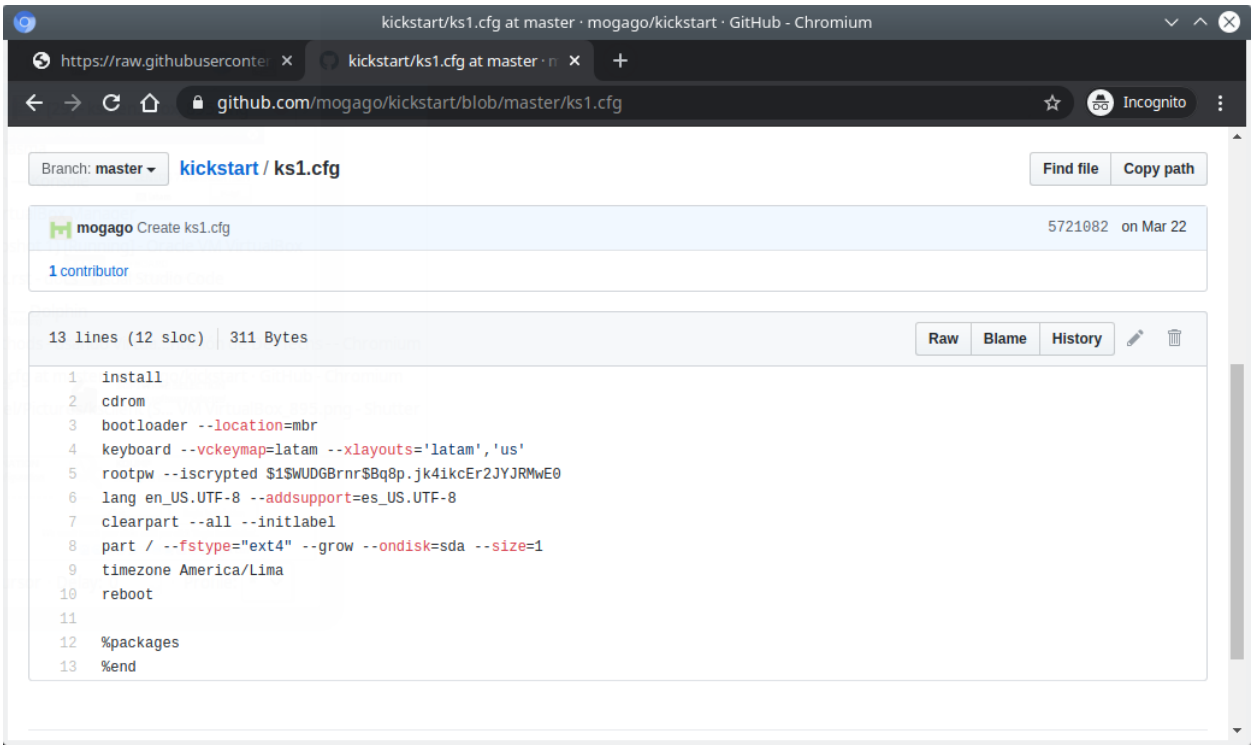


Fig. 17: GitHub - Archivo Kickstart de ejemplo

Pero para obtener el archivo Kickstart en texto plano debemos seleccionar el botón *Raw* en la página anterior, lo cual nos redirigirá a otra URL:

Ya que en esta página solo obtenemos el archivo Kickstart en texto plano, podremos emplear esta dirección URL como parámetro de `ks=http://`.

Important: Este método requiere conexión al servidor web remoto que proporcione el archivo Kickstart. En caso el archivo Kickstart se encuentre en la web, podemos tener una interfaz conectada a una red con un servidor DHCP y obtener una IP y salida a Internet. Por ejemplo, en VirtualBox podemos usar una red NAT o Bridged.

Archivo Kickstart pasado por el mismo ISO de instalación (custom ISO image)

Note: Este método es recomendable usarlo cuando queremos tener un medio de instalación con el archivo de Kickstart pre-cargado. Tenemos la opción de almacenar múltiples archivos Kickstart dentro de archivo ISO.

En este método obtendremos el archivo kickstart del mismo archivo ISO usado para obtener el medio de instalación (kernel, repos, etc.). Para lograr, esto deberemos modificar un imagen ISO oficial, agregarle nuestro archivo kickstart y volver a generar una imagen ISO modficada (también llamada **custom ISO image**). A este método de volver a generar una imagen ISO con ciertas modificaciones personalizadas se le llama **Repackaging de un Linux Install ISO**.

Para esto, primero crearemos nuestra imagen ISO personalizada a través de los siguientes pasos:

1. Crear un directorio para montar nuestro fuente ISO:

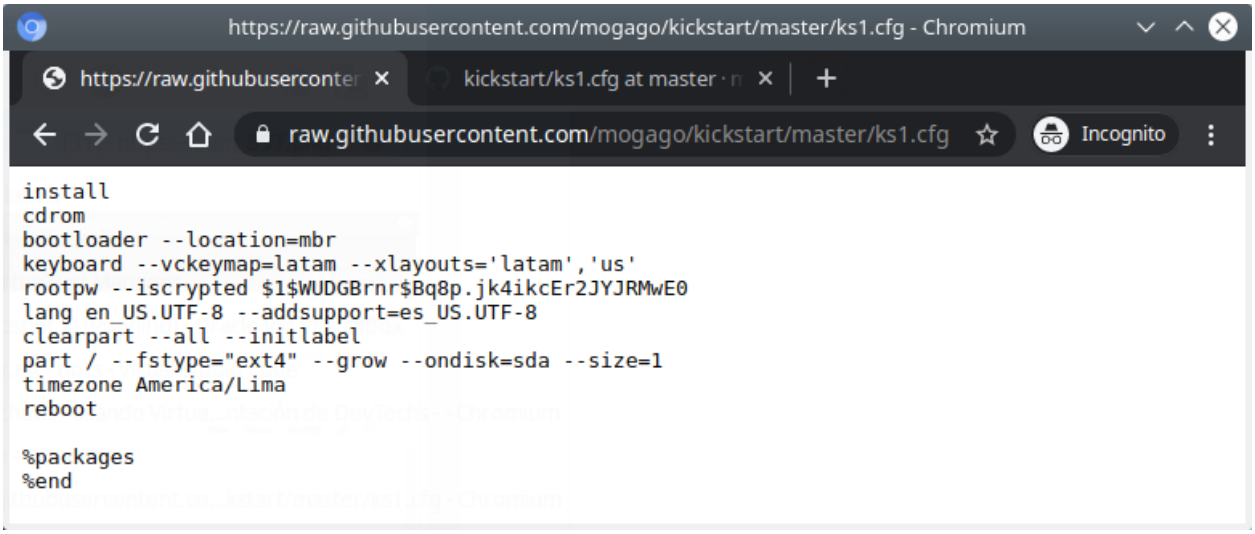


Fig. 18: GitHub - Archivo Kickstart de ejemplo en formato raw

```
sudo -i
mkdir /tmp/bootiso
```

2. Hacer un **loop mount** de nuestra imagen ISO oficial (CentOS/RedHat) que modificaremos:

```
mount -o loop ~/images/ISOS/CentOS-7-x86_64-Minimal-1908.iso /tmp/bootiso
```

3. Crear un directorio de trabajo para nuestro medio personalizado:

```
mkdir /tmp/bootisoks
```

4. Copiar la fuente de medios al directorio de trabajo:

```
cp -r /tmp/bootiso/* /tmp/bootisoks/
```

5. Desmontar la fuente ISO y eliminar el directorio:

```
umount /tmp/bootiso
rmdir /tmp/bootiso
```

6. Cambiar los permisos en el directorio de trabajo:

```
chmod -R u+w /tmp/bootisoks
```

7. Copiar nuestro archivo kickstart personalizado en el directorio de trabajo, bajo el directorio `isolinux/`:

```
cp /path/to/someks.cfg /tmp/bootisoks/isolinux/ks1.cfg

cat /tmp/bootisoks/isolinux/ks1.cfg

# install
# cdrom
# bootloader --location=mbr
# keyboard --vckeymap=latam --xlayouts='latam','us'
# rootpw --iscrypted $1$WUDGBrnr$Bq8p.jk4ikcEr2JYJRMwE0
```

(continues on next page)

(continued from previous page)

```
# lang en_US.UTF-8 --addsupport=es_US.UTF-8
# clearpart --all --initlabel
# part / --fstype="ext4" --grow --ondisk=sda --size=1
# timezone America/Lima
# reboot

# %packages
# %end
```

8. (Opcional) Copiar cualquier RPM adicional a la estructura de directorio y actualizar la metadata:

```
cp /path/to/*.rpm /tmp/bootisoks/Packages/.
cd /tmp/bootisoks/Packages && createrepo -dpo .. .
```

9. (Opcional) Para que las opciones de instalación del menú inicial arranquen usen el archivo Kickstart debemos modificar las opciones de boot en el archivo `isolinux.cfg`:

```
sed -i 's/append\ initrd\=initrd.img/append initrd=initrd.img\ ks\=cdrom:\/ks1.cfg/' /
↪tmp/bootisoks/isolinux/isolinux.cfg
```

10. Crear el nuevo archivo ISO personalizado y conceder permisos al usuario:

```
cd /tmp/bootisoks

mkisofs -o /tmp/boot.iso -b isolinux.bin -c boot.cat -no-emul-boot -boot-load-size 4 -
↪boot-info-table -V "CentOS 7 x86_64" -R -J -v -T isolinux/. .
```

```
chown mogago:mogago /tmp/boot.iso
```

11. (Opcional) Usar `isohybrid` si queremos aplicar `dd` sobre el archivo ISO en un dispositivo USB booteable:

```
isohybrid /tmp/boot.iso
```

12. (Opcional) Añadir un MD5 checksum para permitir realizar pruebas con el medio:

```
implantisomd5 /tmp/boot.iso
```

Ahora debemos cambiar la imagen ISO de CentOS por la que acabamos de crear en la interfaz de VirtualBox:

Al momento de arrancar el medio de instalación, si no hemos usado el paso 9 para editar el archivo `isolinux/isolinux.cfg` y se agregue el archivo Kickstart a las opciones de instalación del menú inicial, deberemos entrar al prompt de `boot:` (tecla *Esc* al cargar el medio de instalación). Aquí, escribiremos:

```
boot: linux ks=cdrom:/ks1.cfg
```

- Referencia 1: How to create a custom ISO image in CentOS
- Referencia 2: mkisofs - Repackaging a Linux Install ISO
- Referencia 3: Mkisofs Wiki

Archivo Kickstart pasado por imagen de disco

Note: Este método es recomendable usarlo cuando tenemos un disco duro o USB con el archivo Kickstart que podamos conectar al servidor donde instalaremos el sistema operativo (o un Virtual Hard Disk en una VM). Además,

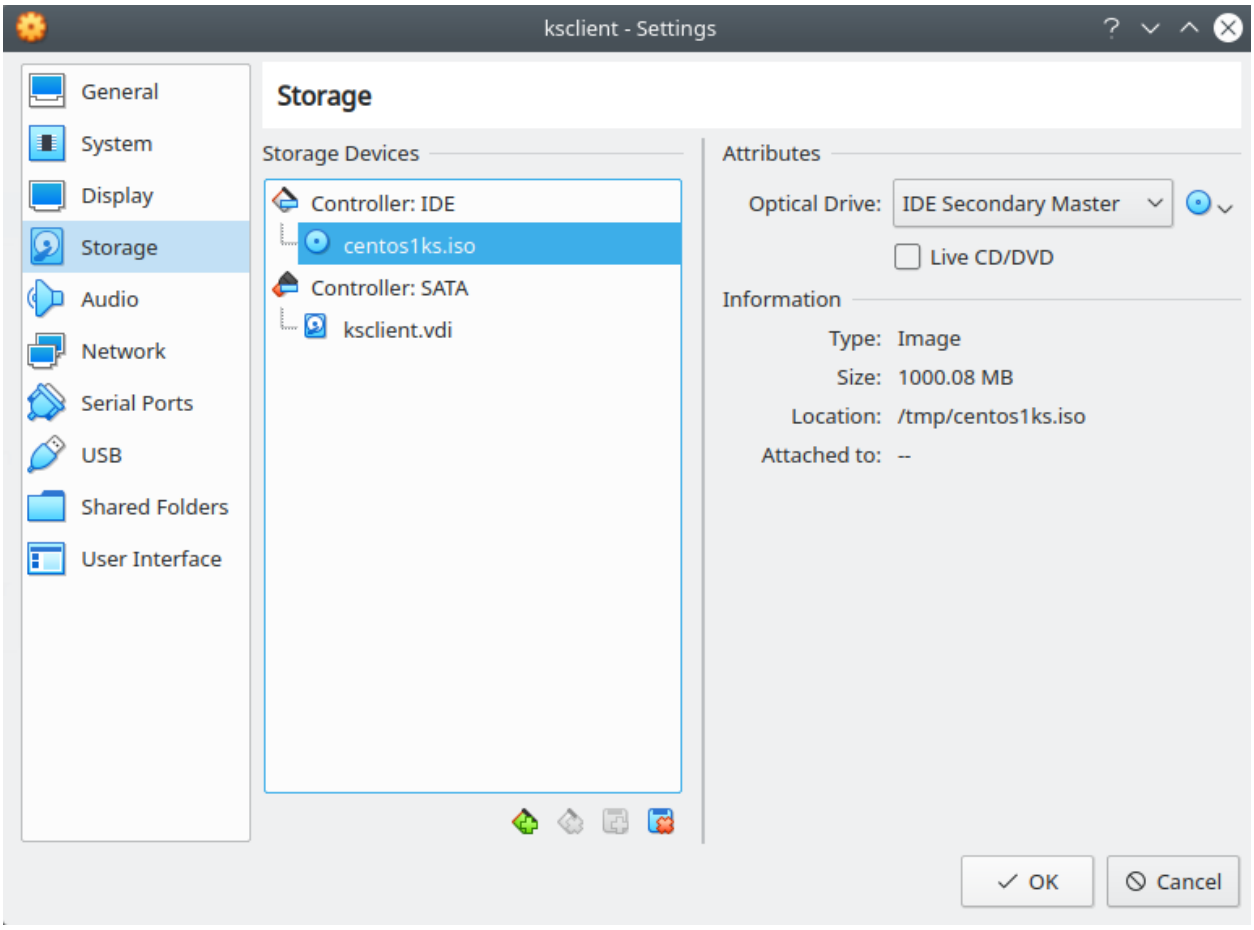


Fig. 19: VirtualBox - seleccionar la imagen ISO personalizada

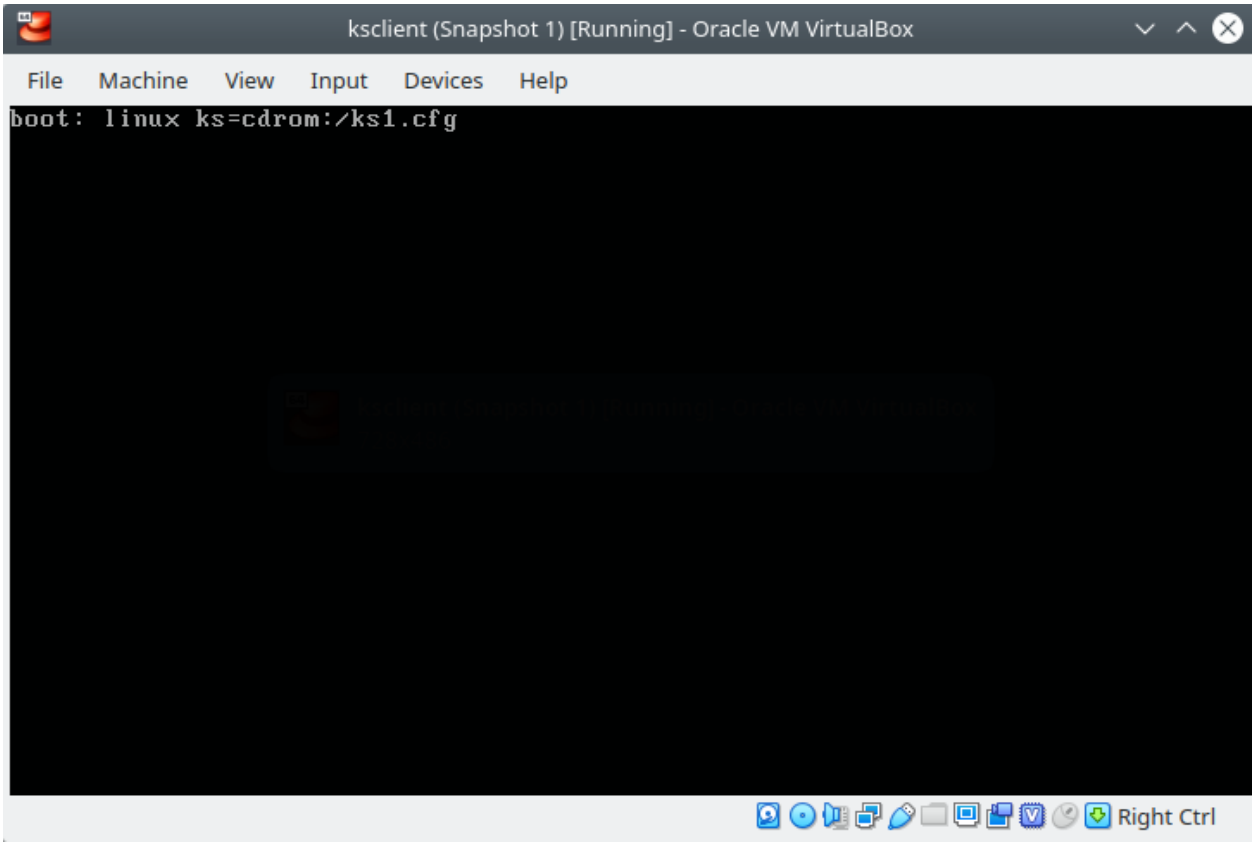
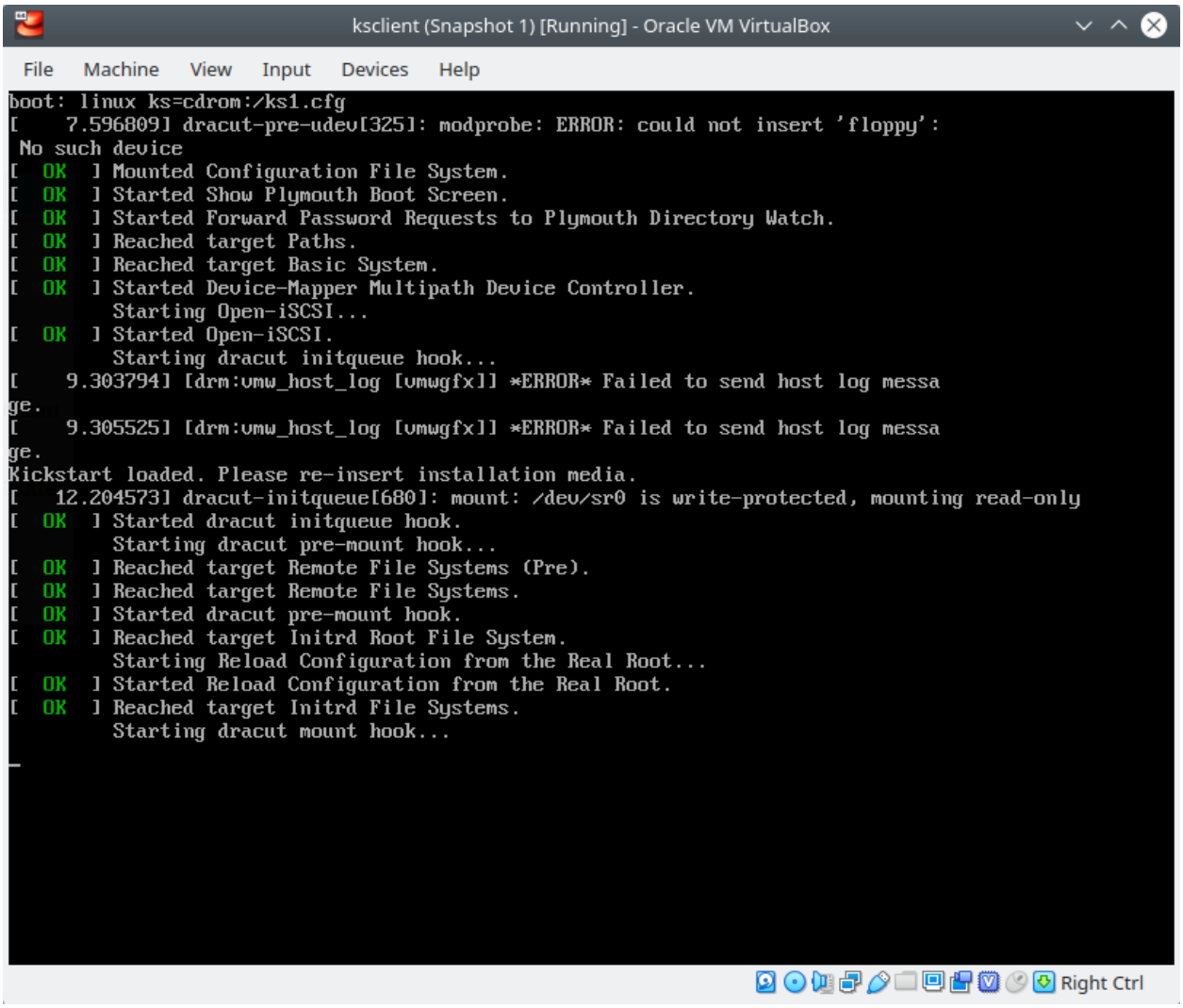


Fig. 20: Centos 7 VM - Ingresar la dirección al archivo kickstart (el directorio actual es `isolinux/`)


ksclient (Snapshot 1) [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help
boot: linux ks=cdrom:/ks1.cfg
[7.596809] dracut-pre-udev[325]: modprobe: ERROR: could not insert 'floppy':
No such device
[OK] Mounted Configuration File System.
[OK] Started Show Plymouth Boot Screen.
[OK] Started Forward Password Requests to Plymouth Directory Watch.
[OK] Reached target Paths.
[OK] Reached target Basic System.
[OK] Started Device-Mapper Multipath Device Controller.
Starting Open-iSCSI...
[OK] Started Open-iSCSI.
Starting dracut initqueue hook...
[9.303794] [drm:vmw_host_log [vmwgfx]] *ERROR* Failed to send host log messa
ge.
[9.305525] [drm:vmw_host_log [vmwgfx]] *ERROR* Failed to send host log messa
ge.
Kickstart loaded. Please re-insert installation media.
[12.204573] dracut-initqueue[680]: mount: /dev/sr0 is write-protected, mounting read-only
[OK] Started dracut initqueue hook.
Starting dracut pre-mount hook...
[OK] Reached target Remote File Systems (Pre).
[OK] Reached target Remote File Systems.
[OK] Started dracut pre-mount hook.
[OK] Reached target Initrd Root File System.
Starting Reload Configuration from the Real Root...
[OK] Started Reload Configuration from the Real Root.
[OK] Reached target Initrd File Systems.
Starting dracut mount hook...
Right Ctrl

lo podemos usar si no deseamos modificar la imagen ISO con el medio de instalación, a diferencia del método por `cdrom`.

En este método obtendremos el archivo kickstart mediante una imagen de disco. Para el caso de VirtualBox usaremos almacenaremos el archivo Kickstart en una images de disco tipo `vdi`. Sin embargo, en servidores físicas podemos usar imágenes con formato raw (`.img`).

Siguiendo la guía *Creando un disco virtual*:, usaremos las herramientas `dd`, `mkfs` y `mount` para crear nuestro **disco virtual** (Virtual Hard Disk o VHD) y agregarle el contenido deseado. Luego lo desmontaremos con `umount` y lo converterimos de `raw` a `vdi` usado `qemu-img`.

1. Crear una imagen que contenga el volumen virtual con `dd`:

```
sudo -i
dd if=/dev/zero of=/media/disk1.img bs=100M count=1

    1+0 records in
    1+0 records out
    104857600 bytes (105 MB, 100 MiB) copied, 0,0962088 s, 1,1 GB/s
```

2. Formatear el archivo de la imagen VHD con `mkfs`, por ejemplo con `ext4`:

```
mkfs -t ext4 /media/disk1.img

    mke2fs 1.44.1 (24-Mar-2018)
    Discarding device blocks: done
    Creating filesystem with 102400 1k blocks and 25688 inodes
    Filesystem UUID: c08e1aa7-8aff-45ed-a913-6aba75bbed9d
    Superblock backups stored on blocks:
            8193, 24577, 40961, 57345, 73729

    Allocating group tables: done
    Writing inode tables: done
    Creating journal (4096 blocks): done
    Writing superblocks and filesystem accounting information: done
```

3. Crear un directorio y montar nuestra imagen de disco formateada:

```
mkdir /mnt/vhd
mount -t auto -o loop /media/disk1.img /mnt/vhd/
```

4. Revisar qué loop tiene el punto de montado

```
lsblk | grep /mnt/vhd

    loop12   7:12   0   100M  0 loop /mnt/vhd
```

5. Asignar un LABEL a nuestra imagen de disco:

```
e2label /dev/loop12 USBKS
```

Para comprobar que tiene el label asignado:

```
lsblk -o name,mountpoint,label,size,uuid,type
```

6. Copiar el archivo Kickstart al directorio donde está montado el disco:

```
cp /path/to/ks1.cfg /mnt/vhd/ks1.cfg
```

7. Desmontar el disco:

```
umount /mnt/vhd
```

9. Para VirtualBox, usando `qemu-img` convertiremos la imagen de raw (`.img`) a `VDI` (formato especial para imágenes de discos de VirtualBox). Además cambiar de dueño y grupo del archivo para que VirtualBox pueda acceder:

```
qemu-img convert -f raw -O vdi /media/disk1.img /home/mogago/Downloads/disk1.vdi

chown mogago:mogago /home/mogago/Downloads/disk1.vdi
```

10. Revisar el UUID de la imagen de disco generada:

```
virt-filesystems -a /home/mogago/Downloads/disk1.vdi --all --long --uuid -h

    Name     Type       VFS  Label MBR Size Parent UUID
    /dev/sda filesystem ext4 USBKS -   100M -      456b92c7-7c4e-424d-8bcc-
↪5f13618d52ac
    /dev/sda device     -    -     -   100M -      -
```

- Referencia 1: Convert images with qemu-img convert
- Referencia 2: Check image UUID
- Referencia 3: Changing label of ext2 ext3 and ext4 type partitions
- Referencia 4: VBoxManage list vms

Una vez que tengamos la imagen de disco preparada podemos conectarlo en la máquina virtual:

Al momento de arrancar el medio de instalación debemos presionar la tecla *Esc* para pasar la dirección del archivo Kickstart como parámetro en el momento de arranque (`boot:`).

Usando el nombre `sdX` de la imagen de disco

Podemos indicar el dispositivo de la imagen de disco que contiene el archivo Kickstart usando su nombre `sdX`. El dispositivo puede llamarse `sda`, `sdb`, `sdc`, etc. dependiendo de la cantidad de dispositivos que tengamos conectados al sistema y el nombre arbitrario que se le haya asignado:

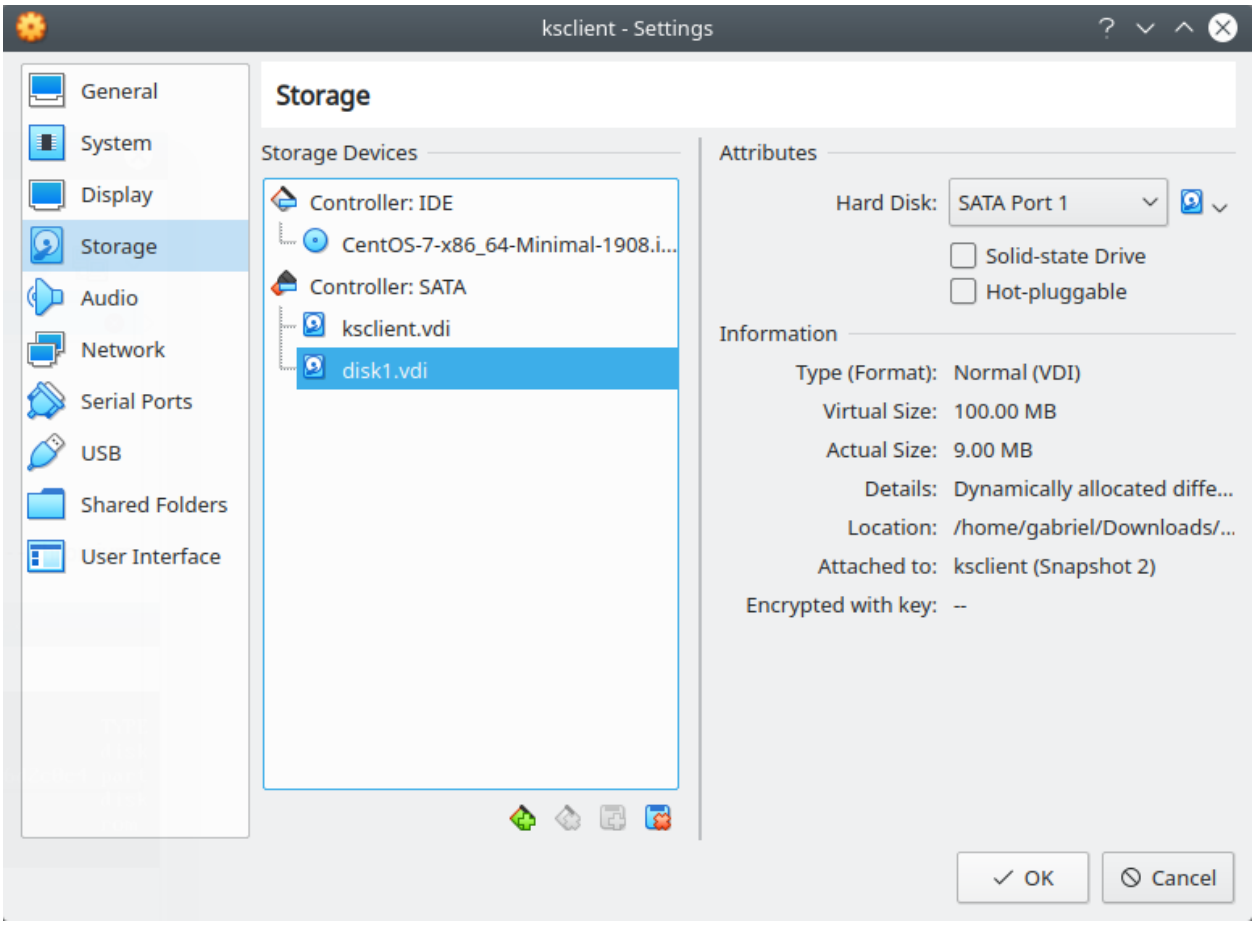
```
boot: linux ks=hd:sdb:/ks1.cfg
```

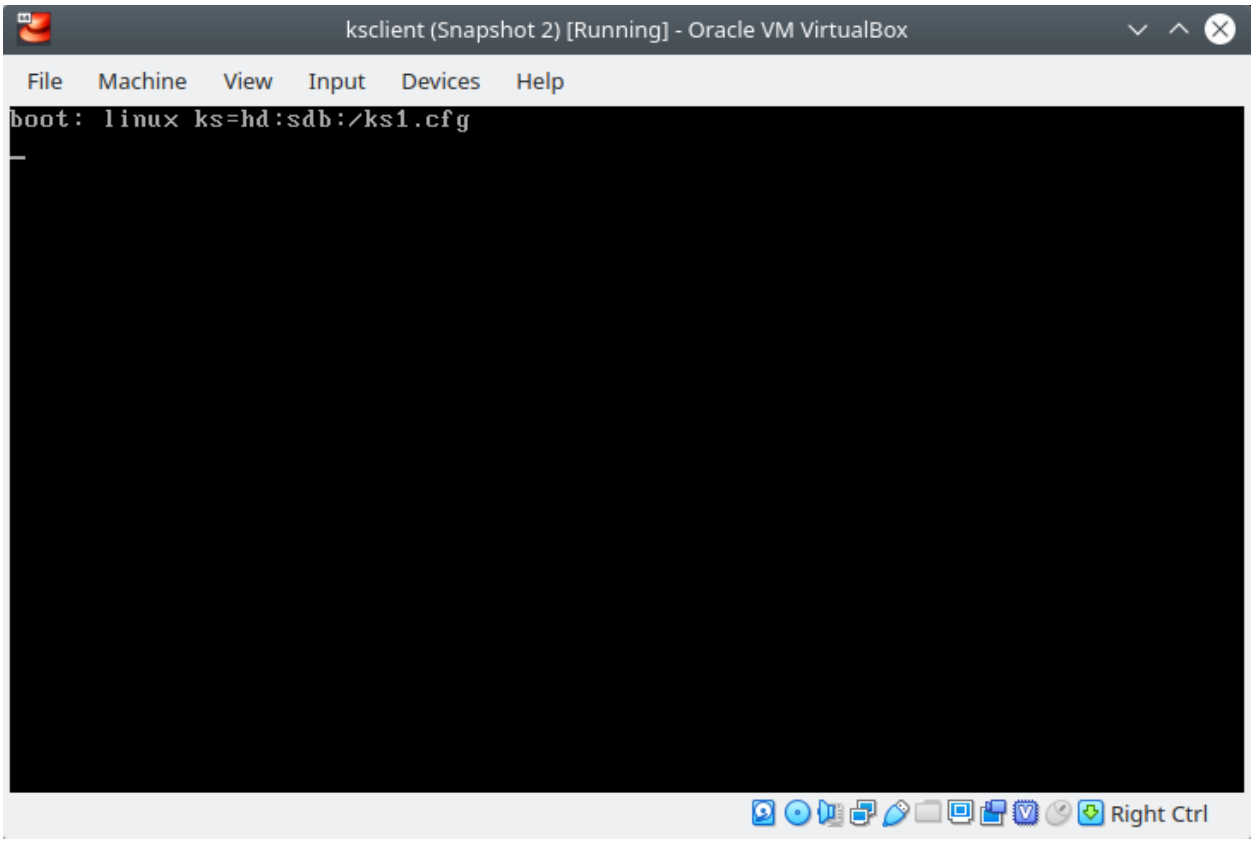
La máquina virtual tiene conectada dos discos (`/dev/sda` y `/dev/sdb`). Si bien hemos obtenido correctamente el archivo Kickstart del diso `sdb`, en otras ocasiones el disco que contenga el archivo Kickstart podría ser el disco `sda`. En el caso que elijamos el disco sin nuestro archivo Kickstart, obtendremos el siguiente log de fallo:

Una vez instalado el sistema operativo podemos comprobar que el tamaño virtual asignado a los discos sigue siendo identificable:

Usando el `LABEL` de la imagen de disco

El método de selección de la imagen de disco por su `sdX` puede resultar en fallo si es que no sabemos exactamente el nombre que el sistema le ha asignado a nuestro disco con el archivo Kickstart. Sin embargo, usar el `LABEL` o `UUID`


ksclient - Settings
General
System
Display
Storage
Audio
Network
Serial Ports
USB
Shared Folders
User Interface
Storage
Storage Devices
Controller: IDE
CentOS-7-x86_64-Minimal-1908.i...
Controller: SATA
ksclient.vdi
disk1.vdi
Attributes
Hard Disk:
SATA Port 1
Solid-state Drive
Hot-pluggable
Information
Type (Format): Normal (VDI)
Virtual Size: 100.00 MB
Actual Size: 9.00 MB
Details: Dynamically allocated diffe...
Location: /home/gabriel/Downloads/...
Attached to: ksclient (Snapshot 2)
Encrypted with key: --
OK
Cancel



del disco puede resultar más específico y evitar errores. Para usar el `LABEL` como parámetro de arranque usamos la siguiente forma:

```
boot: linux ks=hd:LABEL=USBKS:/ks1.cfg
```

Note: En el paso 5 del procedimiento de creación de una imagen de disco, le asignamos una LABEL a nuestro disco. En el paso 5 y el paso 10 comprobamos el nombre del LABEL asignado.

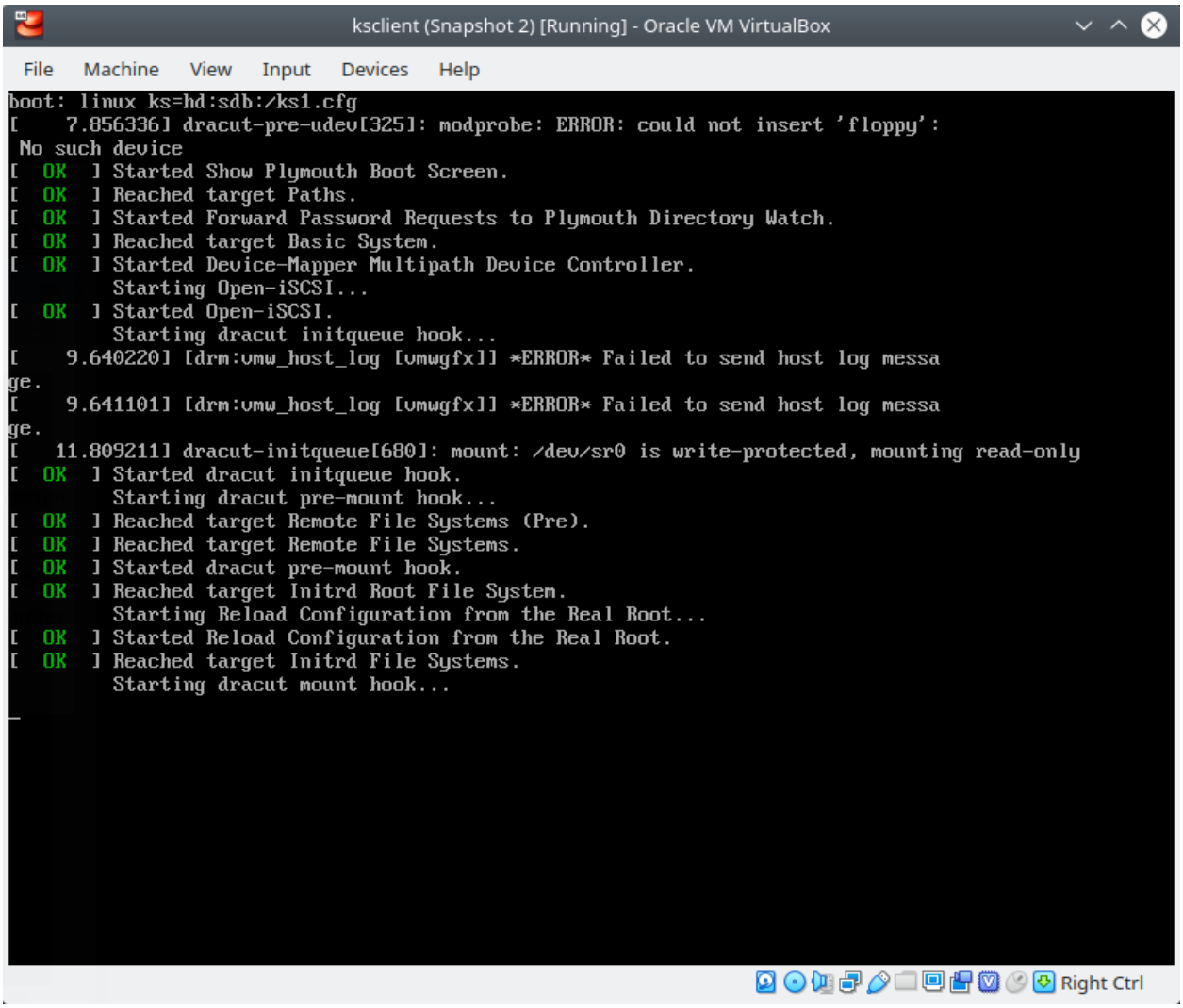
- Referencia: Testcase Kickstart Hd Device Path Ks Cfg

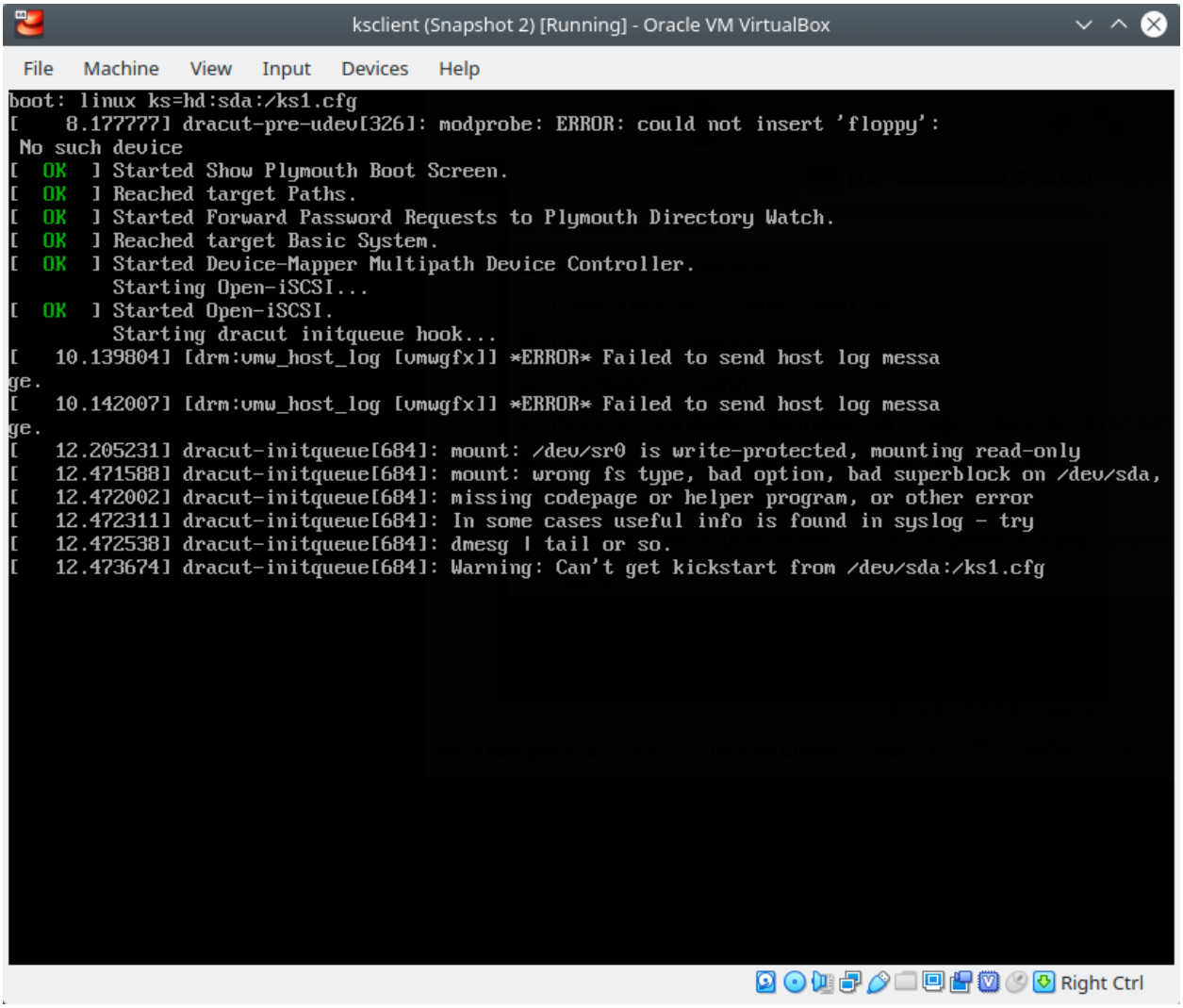
Usando el `UUID` de la imagen de disco

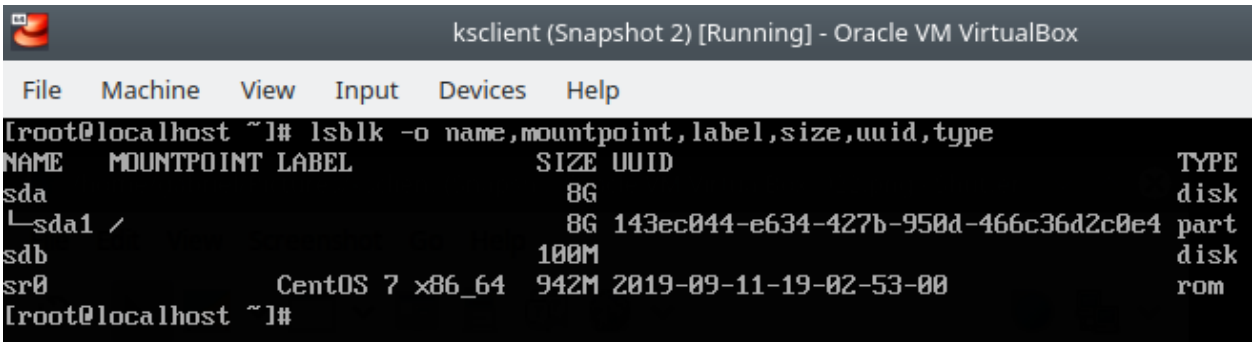
Al igual que la selección de la imagen de disco por su `LABEL`, la elección del disco por su `UUID` es un método más exacto y menos sujeto a errores. Para usar el `UUID` como parámetro de arranque usamos la siguiente forma:

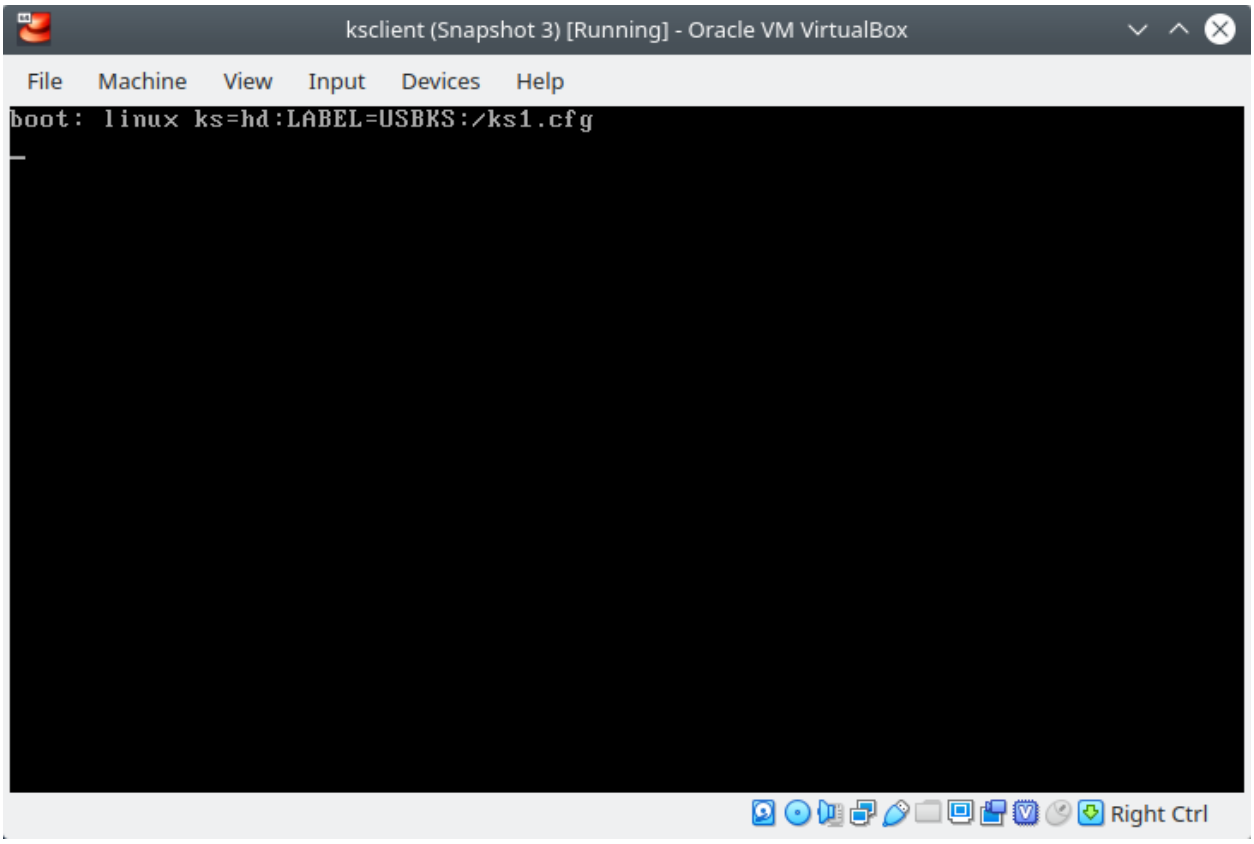
```
boot: linux ks=hd:UUID=456b92c7-7c4e-424d-8bcc-5f13618d52ac:/ks1.cfg
```

Note: En el paso 10 del procedimiento de creación de una imagen de disco comprobamos el UUID.


ksclient (Snapshot 2) [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help
boot: linux ks=hd:sdb:/ks1.cfg
[7.856336] dracut-pre-udev[325]: modprobe: ERROR: could not insert 'floppy':
No such device
[OK] Started Show Plymouth Boot Screen.
[OK] Reached target Paths.
[OK] Started Forward Password Requests to Plymouth Directory Watch.
[OK] Reached target Basic System.
[OK] Started Device-Mapper Multipath Device Controller.
Starting Open-iSCSI...
[OK] Started Open-iSCSI.
Starting dracut initqueue hook...
[9.640220] [drm:vmw_host_log [vmwgfx]] *ERROR* Failed to send host log messa
ge.
[9.641101] [drm:vmw_host_log [vmwgfx]] *ERROR* Failed to send host log messa
ge.
[11.809211] dracut-initqueue[680]: mount: /dev/sr0 is write-protected, mounting read-only
[OK] Started dracut initqueue hook.
Starting dracut pre-mount hook...
[OK] Reached target Remote File Systems (Pre).
[OK] Reached target Remote File Systems.
[OK] Started dracut pre-mount hook.
[OK] Reached target Initrd Root File System.
Starting Reload Configuration from the Real Root...
[OK] Started Reload Configuration from the Real Root.
[OK] Reached target Initrd File Systems.
Starting dracut mount hook...
Right Ctrl

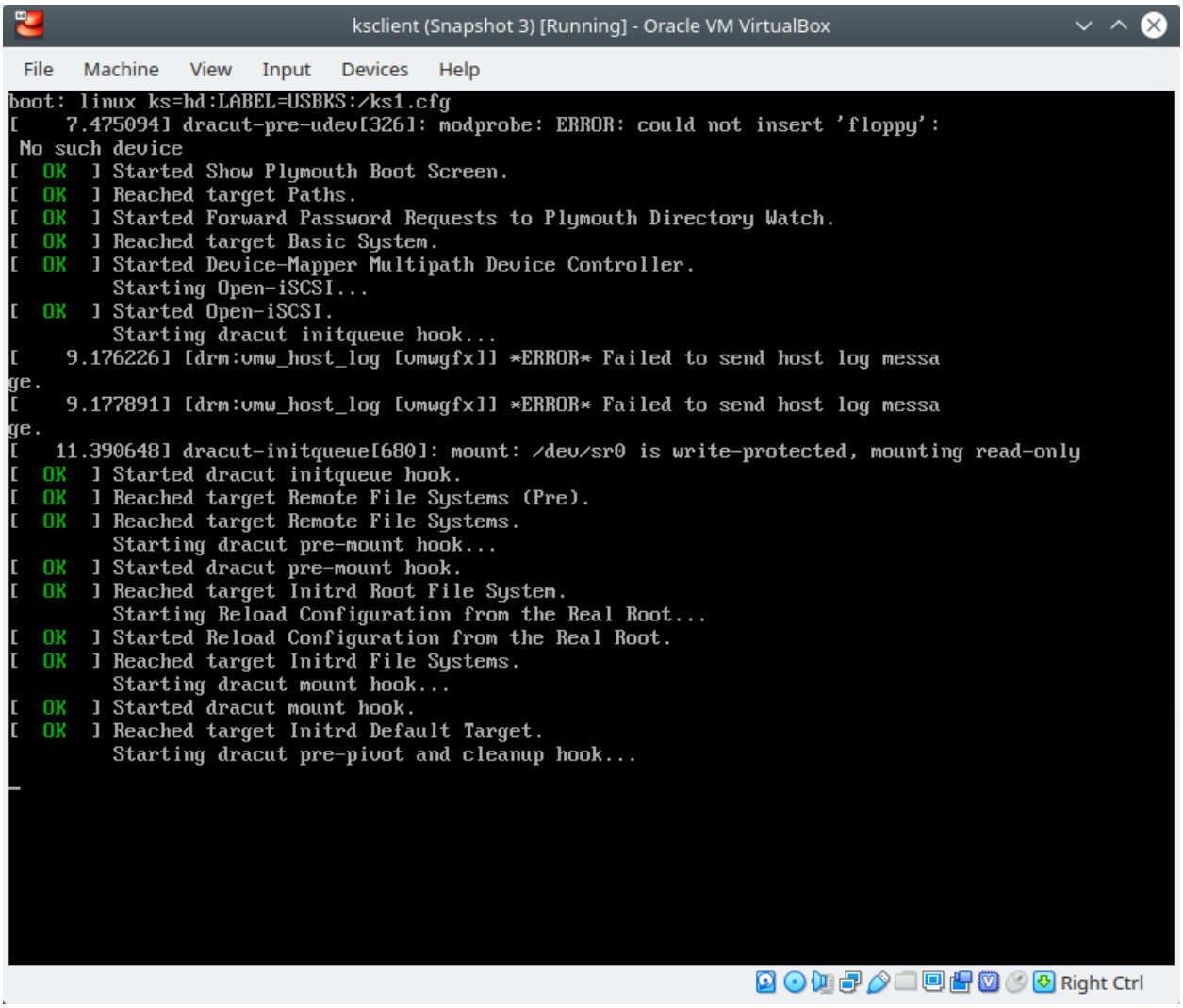

ksclient (Snapshot 2) [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help
boot: linux ks=hd:sda:/ks1.cfg
[8.177777] dracut-pre-udev[326]: modprobe: ERROR: could not insert 'floppy':
No such device
[OK] Started Show Plymouth Boot Screen.
[OK] Reached target Paths.
[OK] Started Forward Password Requests to Plymouth Directory Watch.
[OK] Reached target Basic System.
[OK] Started Device-Mapper Multipath Device Controller.
Starting Open-iSCSI...
[OK] Started Open-iSCSI.
Starting dracut initqueue hook...
[10.139804] [drm:vmw_host_log [vmwgfx]] *ERROR* Failed to send host log messa
ge.
[10.142007] [drm:vmw_host_log [vmwgfx]] *ERROR* Failed to send host log messa
ge.
[12.205231] dracut-initqueue[684]: mount: /dev/sr0 is write-protected, mounting read-only
[12.471588] dracut-initqueue[684]: mount: wrong fs type, bad option, bad superblock on /dev/sda,
[12.472002] dracut-initqueue[684]: missing codepage or helper program, or other error
[12.472311] dracut-initqueue[684]: In some cases useful info is found in syslog - try
[12.472538] dracut-initqueue[684]: dmesg | tail or so.
[12.473674] dracut-initqueue[684]: Warning: Can't get kickstart from /dev/sda:/ks1.cfg
Right Ctrl

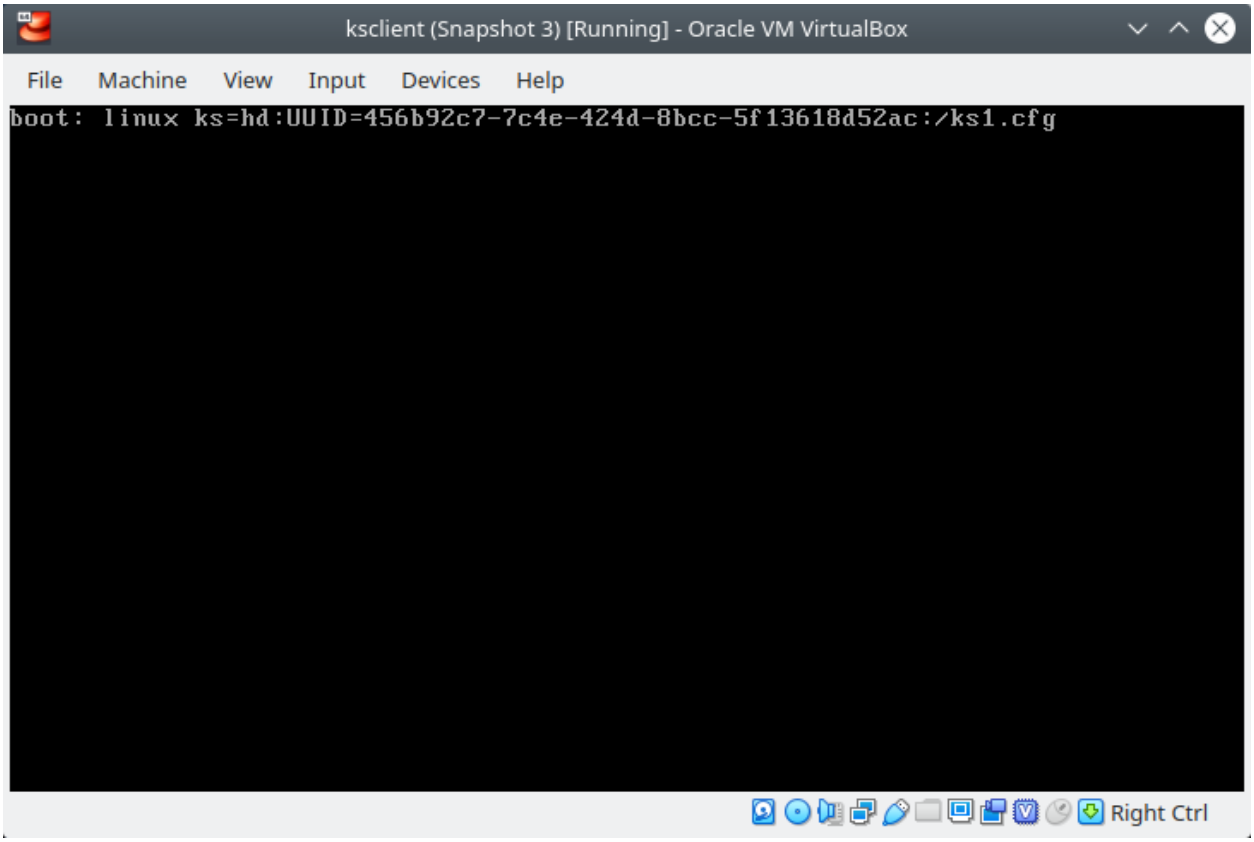

ksclient (Snapshot 2) [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help
[root@localhost ~]# lsblk -o name,mountpoint,label,size,uuid,type
NAME MOUNTPOINT LABEL SIZE UUID TYPE
sda 8G disk
└─sda1 / 8G 143ec044-e634-427b-950d-466c36d2c0e4 part
sdb 100M disk
sr0 CentOS 7 x86_64 942M 2019-09-11-19-02-53-00 rom
[root@localhost ~]#

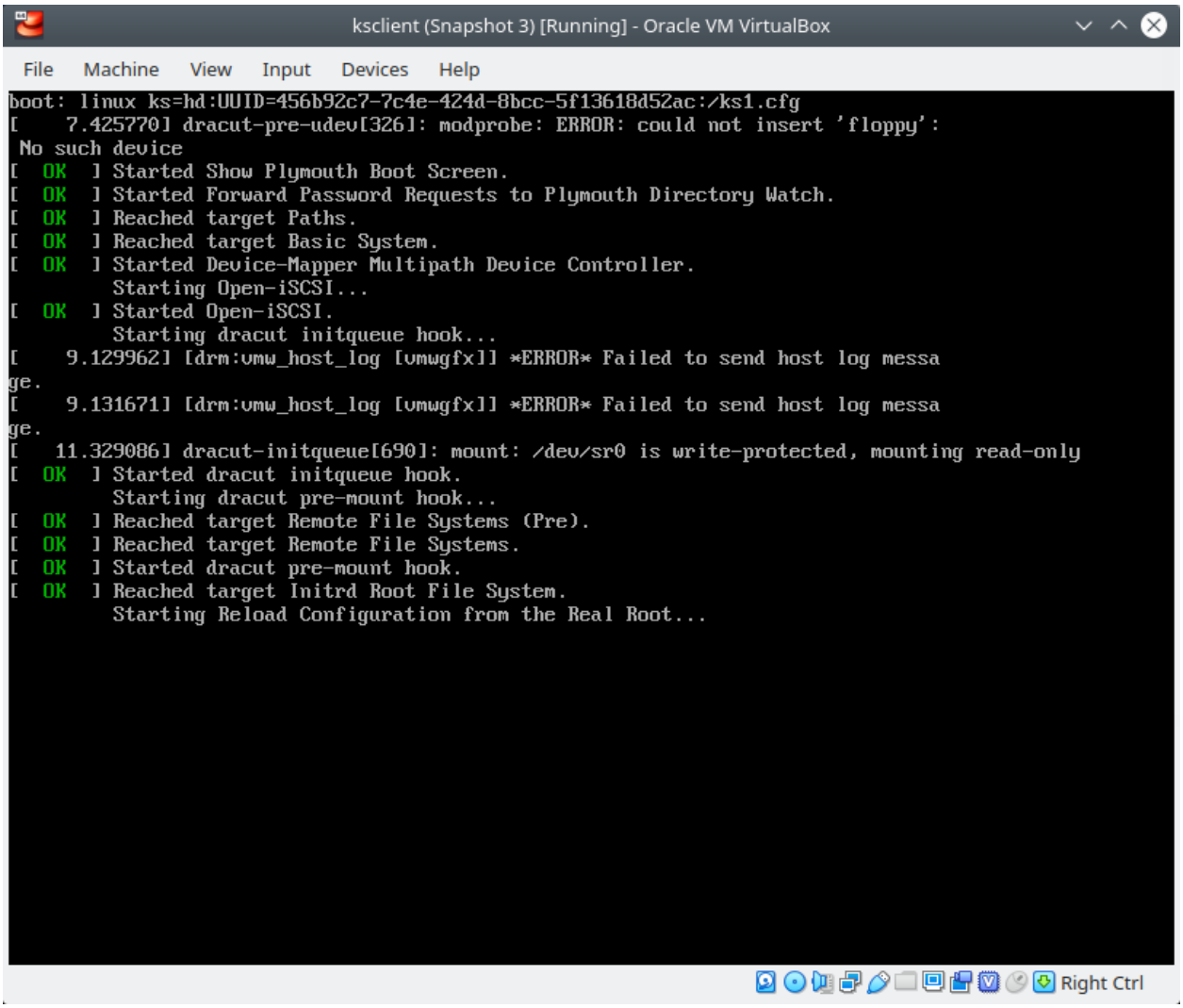


11.1.4 Instalación de CentOS 7

1. Inmediatamente luego de presionar el botón *Enter* en el prompt de `boot:` comenzará el proceso de instalación automatizado con el archivo Kickstart que hemos pasado. Dependiendo de la configuración de Kickstart, una instalación en modo gráfico o en modo texto, obtendremos una de las siguientes dos pantallas:
2. Una vez acabada la instalación desatentidada del SO, la VM se reiniciará automáticamente y cargará desde el disco instalado:
3. Finalmente, iniciará el SO CentOS instalado automáticamente con Kickstart:


ksclient (Snapshot 3) [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help
boot: linux ks=hd:LABEL=USBKS:/ks1.cfg
[7.475094] dracut-pre-udev[326]: modprobe: ERROR: could not insert 'floppy':
No such device
[OK] Started Show Plymouth Boot Screen.
[OK] Reached target Paths.
[OK] Started Forward Password Requests to Plymouth Directory Watch.
[OK] Reached target Basic System.
[OK] Started Device-Mapper Multipath Device Controller.
Starting Open-iSCSI...
[OK] Started Open-iSCSI.
Starting dracut initqueue hook...
[9.176226] [drm:vmw_host_log [vmwgfx]] *ERROR* Failed to send host log messa
ge.
[9.177891] [drm:vmw_host_log [vmwgfx]] *ERROR* Failed to send host log messa
ge.
[11.390648] dracut-initqueue[680]: mount: /dev/sr0 is write-protected, mounting read-only
[OK] Started dracut initqueue hook.
[OK] Reached target Remote File Systems (Pre).
[OK] Reached target Remote File Systems.
Starting dracut pre-mount hook...
[OK] Started dracut pre-mount hook.
[OK] Reached target Initrd Root File System.
Starting Reload Configuration from the Real Root...
[OK] Started Reload Configuration from the Real Root.
[OK] Reached target Initrd File Systems.
Starting dracut mount hook...
[OK] Started dracut mount hook.
[OK] Reached target Initrd Default Target.
Starting dracut pre-pivot and cleanup hook...
Right Ctrl


ksclient (Snapshot 3) [Running] - Oracle VM VirtualBox
File
Machine
View
Input
Devices
Help
boot: linux ks=hd:UUID=456b92c7-7c4e-424d-8bcc-5f13618d52ac:/ks1.cfg
Right Ctrl


ksclient (Snapshot 3) [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help
boot: linux ks=hd:UUID=456b92c7-7c4e-424d-8bcc-5f13618d52ac:/ks1.cfg
[7.425770] dracut-pre-udev[326]: modprobe: ERROR: could not insert 'floppy':
No such device
[OK] Started Show Plymouth Boot Screen.
[OK] Started Forward Password Requests to Plymouth Directory Watch.
[OK] Reached target Paths.
[OK] Reached target Basic System.
[OK] Started Device-Mapper Multipath Device Controller.
Starting Open-iSCSI...
[OK] Started Open-iSCSI.
Starting dracut initqueue hook...
[9.129962] [drm:vmw_host_log [vmwgfx]] *ERROR* Failed to send host log messa
ge.
[9.131671] [drm:vmw_host_log [vmwgfx]] *ERROR* Failed to send host log messa
ge.
[11.329086] dracut-initqueue[690]: mount: /dev/sr0 is write-protected, mounting read-only
[OK] Started dracut initqueue hook.
Starting dracut pre-mount hook...
[OK] Reached target Remote File Systems (Pre).
[OK] Reached target Remote File Systems.
[OK] Started dracut pre-mount hook.
[OK] Reached target Initrd Root File System.
Starting Reload Configuration from the Real Root...
Right Ctrl

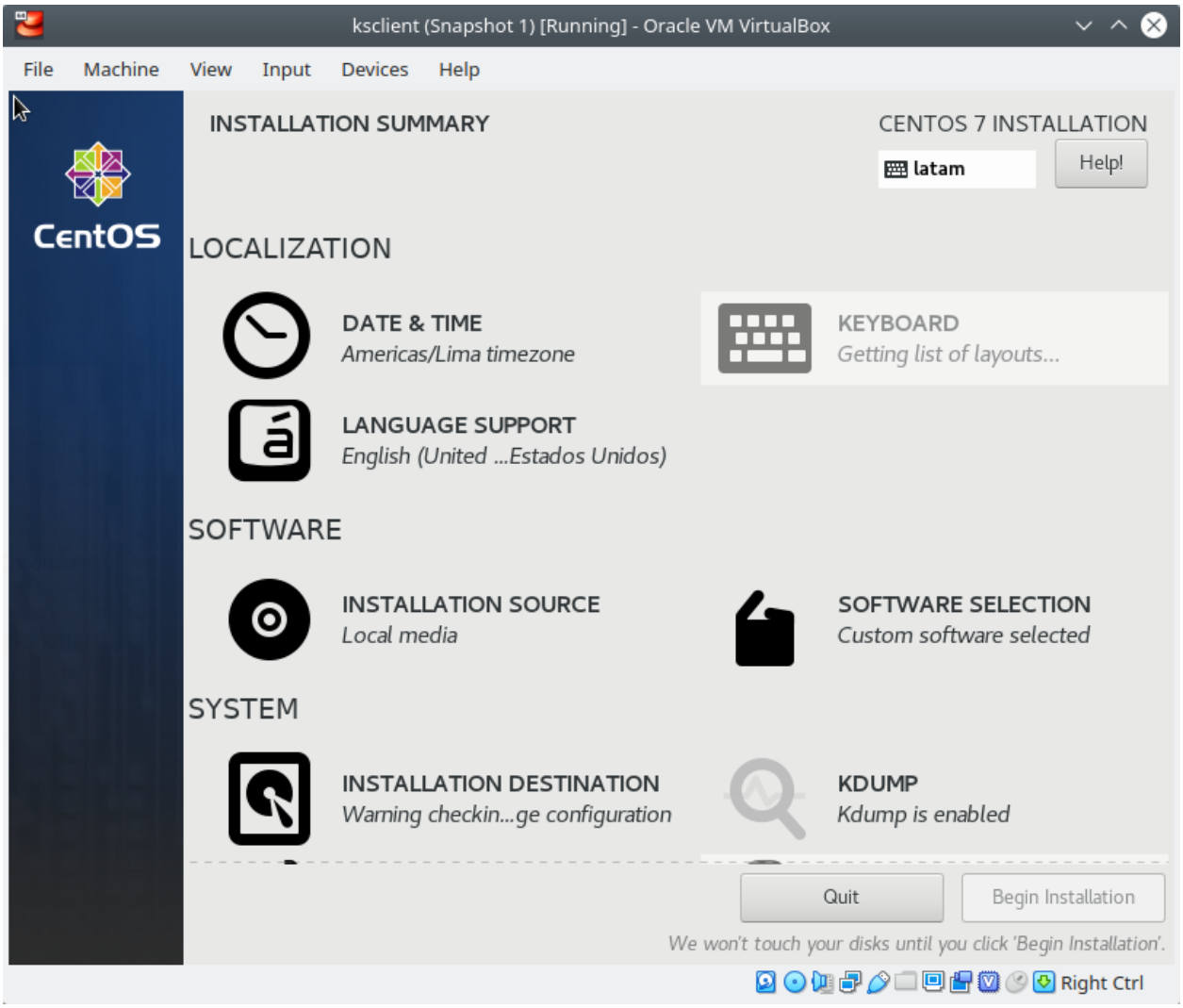


Fig. 21: Centos 7 VM - Iniciará el proceso de instalación automatizado en modo gráfico

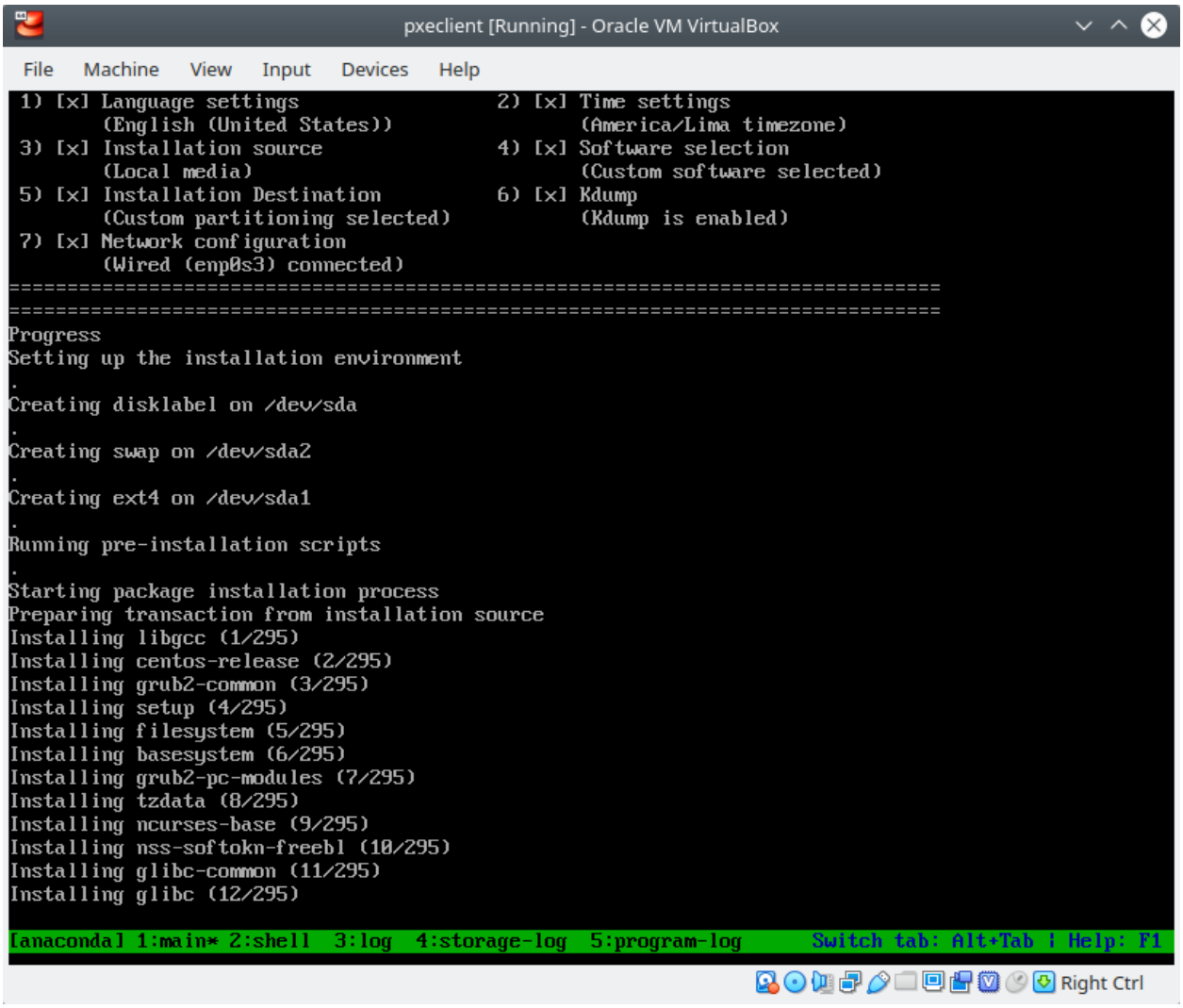


Fig. 22: Centos 7 VM - Iniciará el proceso de instalación automatizado en modo texto

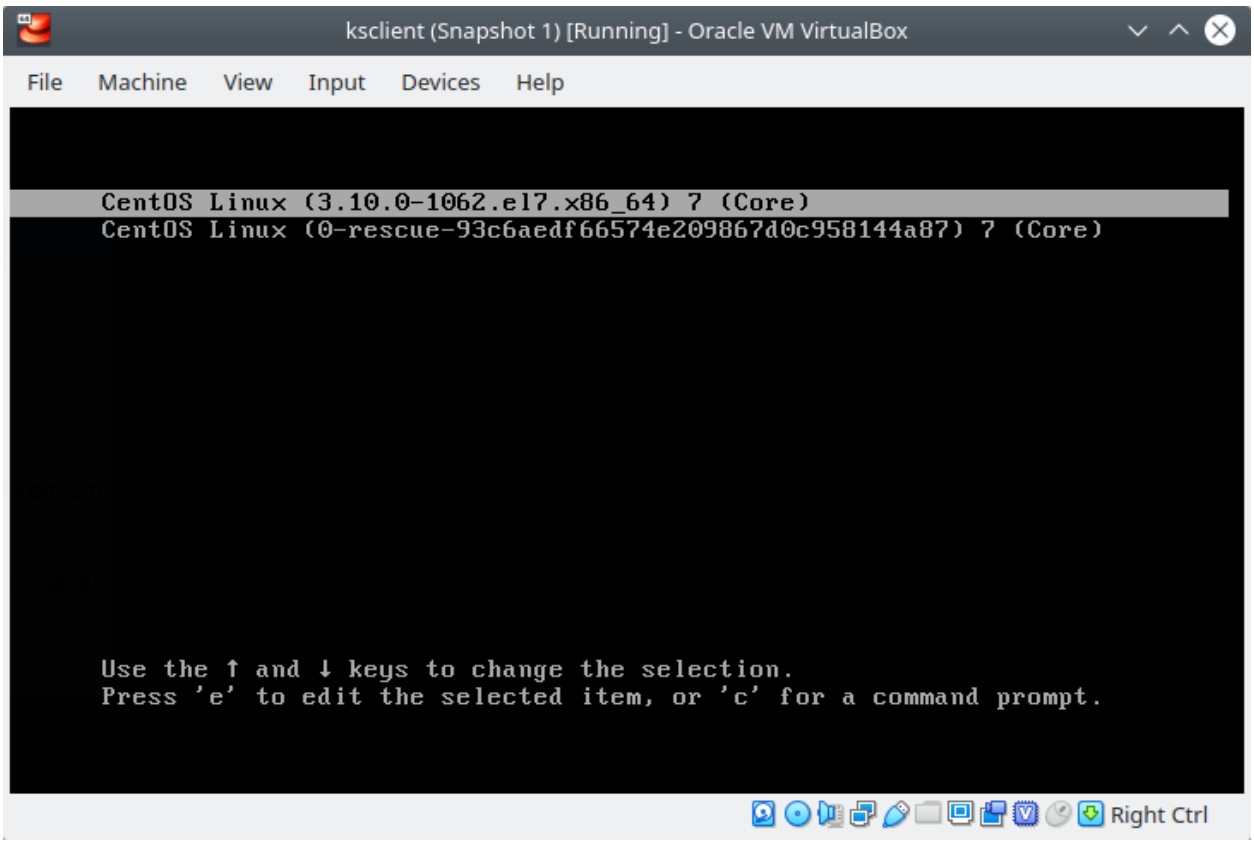


Fig. 23: Centos 7 VM - Iniciará el sistema CentOS 7 instalado

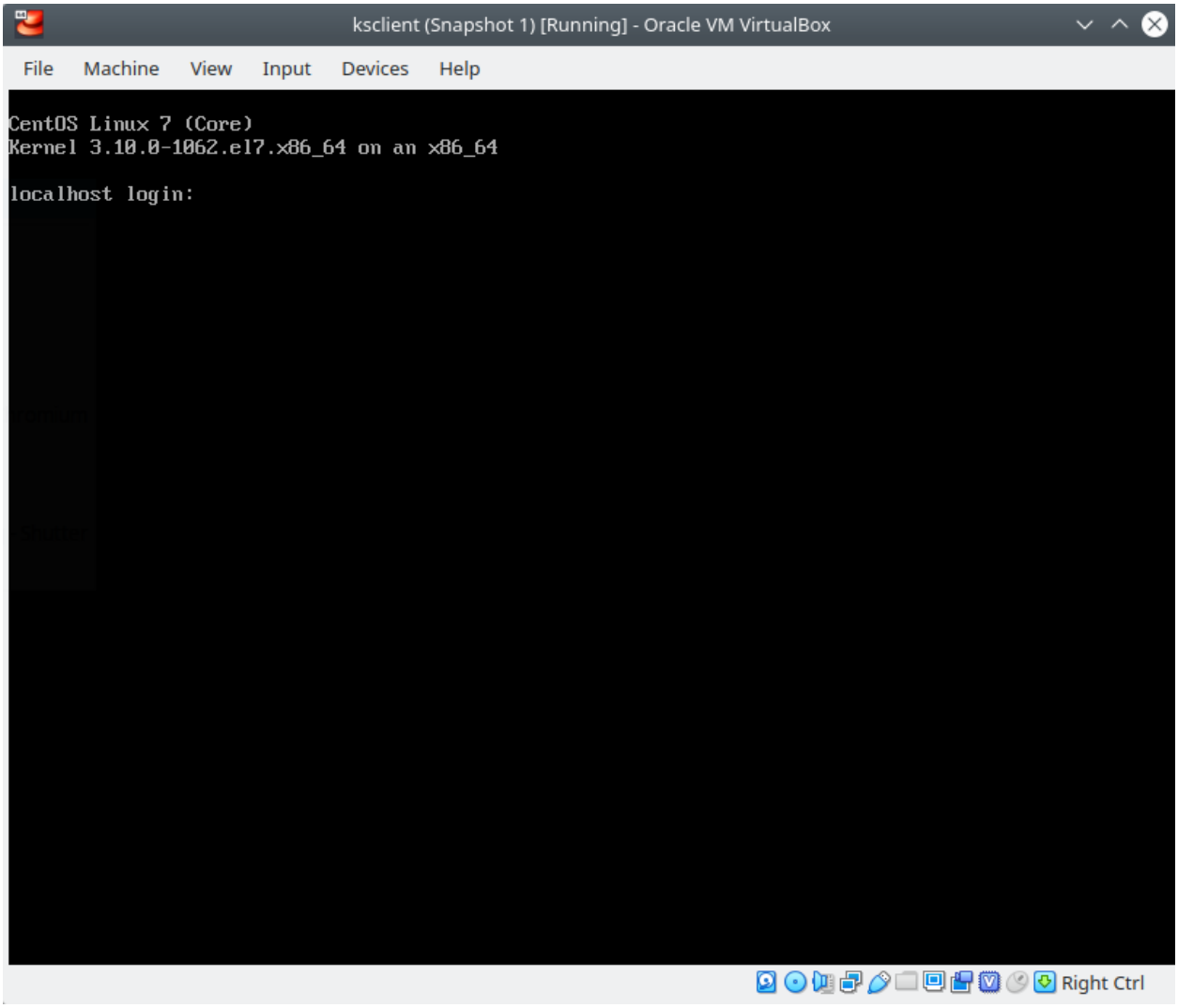


Fig. 24: Centos 7 VM - Sistema operativo CentOS 7 funcionando

CHAPTER 12

Kickstart Templates

12.1 Kickstart Templates 1

Table of Contents

- Referencia 1: Red Hat Docs - KICKSTART SYNTAX REFERENCE

12.1.1 Archivo Kickstart - Configuración mínima

Archivo Kickstart con una configuración mínima para lograr una instalación de CentOS 7 totalmente desatendida:

```
install
cdrom
```

(continues on next page)

(continued from previous page)

```
bootloader --location=mbr
keyboard --vckeymap=latam --xlayouts='latam','us'
rootpw --iscrypted $1$WUDGBrnr$Bq8p.jk4ikcEr2JYJRMwE0
lang en_US.UTF-8 --addsupport=es_US.UTF-8
clearpart --all --initlabel
part / --fstype="ext4" --grow --ondisk=sda --size=1
timezone America/Lima
reboot

%packages
%end
```

- `install` - (optional). Modo de instalación. Debe ir acompañado en otra línea del tipo de instalación: `cdrom`, `harddrive`, `nfs`, `liveimg`, `url` (FTP, HTTP, HTTPS).
- `cdrom` - instalar desde el primer disco óptico en el sistema.
- **`bootloader` - (required). Especifica cómo debe instalarse el boot loader**
  - **`--location=` - especifica donde es escrito el boot record**
    - `mbr` - (default). Depende de si el drive usa el Master Boot Record (MBR) o un esquema GUID Partition Tables (GPT). En un disco con formato GTP, essta opción instala la etapa 1.5 del boot loader en la particion de arranque BIOS. En un disco con formato MBR, la etapa 1.5 es instalada en un espacio vacío entre el MBR y la primera partición.

```
bootloader --location=mbr --append="hdd=ide-scsi ide=nodma"
```

- **`keyboard` - (required). Configura uno o más keyboard layouts para el sistema.**
  - `vckeymap` - especifica un `VConsole` keymap que se debe usar.
  - `--xlayouts` - especifica una lista de X layouts que deben ser usados.

```
keyboard --xlayouts=us,'cz (qwerty)' --switch=grp:alt_shift_toggle
```

- **`rootpw` - (required). Configura la contraseña root con el argumento de la contraseña**
  - `--iscrypted` - si esta opción está presente se asume que el argumento de la contraseña ya está encriptado.

```
rootpw [--iscrypted|--plaintext] [--lock] password
```

- **`lang` - (required). Establece el lenguaje a usar durante la instalación y el lenguaje usado por defecto en el sistema instala**
  - `--addsupport=` - añadir soporte de lenguajes adicionales. Podemos listar varios separados por comas.

```
lang en_US --addsupport=cs_CZ,de_DE,en_UK
```

- **`clearpart` - (optional). Remueve las particiones del sistema, previo a la creación de nuevas particiones. Por defecto ning**
  - `--all` - elimina todas las particiones del sistema. Se eliminarán todos los discos que puedan ser alcanzados por el instalador, incluyendo cualquier almacenamiento conectado por red.
  - `--initlabel` - inicializa un disco discos, creando un disk label por defecto para todos los discos en su arquitectura respectiva (p.ej. msdos para x86) que hayan sido designados para formatear. Ya que

`--initlabel` puede ver todos los discos, es importante asegurarse que solo los drivers que serán formateadas se encuentran conectados.

- **`part` - (o `partiton`). (required). Crea una partición en el sistema. Todas las particiones creadas son formateadas, como**
  - `--fstype=` - configura el tipo de file system para la partición. Valores posibles: `xfs`, `ext2`, `ext3`, `ext4`, `swap`, `vfat`, `efi` y `biosboot`.
  - `--grow` - le indica al volumen lógico que crezca para rellenar el espacio disponible (si existe), o hasta la configuración de tamaño máximo, si se especifica uno. Un tamaño mínimo debe ser especificado usando la opción `--percent=` o `--size=`.
  - `--ondisk=` - (o `--ondrive=`). Crea una partición (especificada por el comando part) en un disco existente. Este comando siempre crea una partición.
  - `--size=` - tamaño del volumen lógico en MiB. No podemos usar esta opción junto con `--percent=`
- `timezone` - (required). Configura la zona horaria.
- `reboot` - (optional). Reinicia luego de que la instalación es completada satisfactoriamente. Esta opción es equivalente a ejecutar el comando `shutdown -r`.
- `%packages` - Usar este comando para iniciar una sección en Kickstart que describa los paquetes de software a ser instalados. Podemos especificar paquetes por environmet (`@^`), group (`@`), o por sus nombres de paquete. El `Core` group siempre es seleccionado por defecto, cuando se usa esta sección `%packages`, por lo que no es obligatorio especificarlo.

Note: Verificar el nombre con el cual son creados los discos, para planear las particiones. En VirtualBox los discos se crean con el formato de nombre `sdx` (`--ondisk=sda`), pero en virt-manager los discos se crean con el formato de nombre `vdx` (`--ondisk=vda`). (x=a,b,c,d,...)

12.1.2 Archivo Kickstart - Creación de usuario

Archivo Kickstart con la funcionalidad para crear un nuevo usuario:

```
install
cdrom
bootloader --location=mbr
keyboard --vckeymap=latam --xlayouts='latam','us'
rootpw --iscrypted $1$WUDGBrnr$Bq8p.jk4ikcEr2JYJRMwE0
lang en_US.UTF-8 --addsupport=es_US.UTF-8
clearpart --all --initlabel
part / --fstype="ext4" --grow --ondisk=sda --size=1
timezone America/Lima
reboot

user --name=user1 --groups=wheel --iscrypted --password=$1$WUDGBrnr$Bq8p.
↪jk4ikcEr2JYJRMwE0

%packages
%end
```

- **`user` - (optional). Crear un nuevo usuario en el sistema.**
  - `--name=` - (required). Provee el nombre del usuario.

- `--groups=` - En adición al grupo por defecto, una lista separada por comas de nombre de grupos a los que el usuario debe pertenecer. Los grupos deben existir antes que la cuenta de usuario sea creada.
- `--homedir=` - El directorio home del usuario. Si no es provisto, por defecto se usará `/home/username`.
- `--password=` - La contraseña del nuevo usuario. Si no es provista, la cuenta será bloqueada por defecto.
- `--iscrypted` - si esta opción está presente se asume que el argumento de la contraseña ya está encriptado.

```
user --name=username [options]
```

Note: Podemos usar una contraseña con HASH MD5, SHA256 o SHA512. El instalador comprenderá el tipo de HASH que se está usando según la longitud de caracteres este. Por ejemplo, estos dos líneas usan un HASH SHA512 y MD5 respectivamente, pero tienen el mismo resultado:

```
user --name=user1 --groups=wheel --iscrypted --password=$6$KnYe93Ga4BpTVh8i
↪$bGNPmIEGhgCUTx9i3wpZvxx6ZS3OP0t7tw4UxCK67dtDftIjaKVMqlE8rwhdwCR9Pq1g.
↪OsyQXQyMv2DUHvjc0
user --name=user1 --groups=wheel --iscrypted --password=$1$WUDGBrnr$Bq8p.
↪jk4ikcEr2JYJRMwE0
```

12.1.3 Archivo Kickstart - Static network configuration

Archivo Kickstart con la funcionalidad para configurar la red:

```
install
cdrom
bootloader --location=mbr
keyboard --vckeymap=latam --xlayouts='latam','us'
rootpw --iscrypted $1$WUDGBrnr$Bq8p.jk4ikcEr2JYJRMwE0
lang en_US.UTF-8 --addsupport=es_US.UTF-8
clearpart --all --initlabel
part / --fstype="ext4" --grow --ondisk=sda --size=1
timezone America/Lima
reboot

network --bootproto=static --device=00:11:22:33:44:55 --ip=10.1.1.5 --netmask=255.255.
↪255.0 gateway=10.1.1.5 --nameserver=8.8.8.8 --hostname=centos7

%packages
%end
```

- **`network` - (optional). Configura la información de red para el sistema objetivo y activa dispositivos de red en el entorno d**

  - `--activate` - activa este dispositivo en el entorno de instalación. Si usamos la opción `--activate` en un dispositivo que ya ha sido activado (por ejemplo, una interfaz que hemos configurado con boot options para que el sistema pueda obtener el archivo Kickstart), el dispositivo es reactivado para usar los detalles especificados en el archivo Kickstart.
  - `--bootproto=` - Toma uno de los siguientes valores: `dhcp` (default), `bootp`, `ibft`, `static`.

- **`--device=` - Especifica el dispositivo a ser configurado con el comando `network`. Podemos especificar un disp**
  - el nombre de dispositivo de la interfaz (p. ej. `em1`)
  - la dirección MAC de la interfaz (p. ej. `01:23:45:67:89:ab`)
- `--ip` - dirección IPv4 del dispositivo.
- `--netmask` - máscara de red para el sistema instalado.
- `--gateway` - default gateway como única dirección IPv4.
- `--nameserver` - DNS name server, como una dirección IP. Para especificar múltiples name servers, listarlos y separar cada dirección IP con comas.
- `--hostname` - El host name del sistema instalado.

12.1.4 Archivo Kickstart - Pre-install script

Caso de uso para un pre-installation script:

Si hemos configurado una dirección IP y un hostname específicos para un cliente (según la MAC del cliente) en el servidor DHCP, podemos volver esta configuración estática. Es decir, los parámetros asignados por el servidor DHCP se mantendrán luego del reinicio del sistema, posterior a la instalación. El fin de este script es que la configuración de red asignada por el servidor DHCP se vuelva estática, aún cuando este servidor no esté activo y el cliente no pueda recibir una IP ni hostname.

Para volver estática la configuración asignada por el servidor DHCP se puede usar un pre-installation script en el archivo Kickstart para la instalación del cliente. Este pre-installation script obtiene IP y netmask del comando `ip` y el hostname del comando `hostname` y los almacena en variables. Luego se establece una entrada para el archivo kickstart que configure estáticamente la configuración de red.

```
install
cdrom
bootloader --location=mbr
keyboard --vckeymap=latam --xlayouts='latam','us'
rootpw --iscrypted $1$WUDGBrnr$Bq8p.jk4ikcEr2JYJRMwE0
lang en_US.UTF-8 --addsupport=es_US.UTF-8
clearpart --all --initlabel
part / --fstype="ext4" --grow --ondisk=sda --size=1
timezone America/Lima
reboot

# Incluir archivo generado en pre installation script
%include /tmp/network.ks

%packages
%end

%pre
#!/bin/bash

inet="$(ip addr show dev enp0s3 scope global | grep 'inet')"
#echo "$inet"

#ipdir="$(echo "$inet" | awk '{ print $2 }')"
ipdir="$(echo "$inet" | cut -d ' ' -f 6)"
#echo "$ipdir"
```

(continues on next page)

(continued from previous page)

```
ip="$(echo "$ipdir" | cut -d / -f1)"
echo "$ip"

netmaskCIDR="$(echo "$ipdir" | cut -d / -f2)"
#echo "$netmaskCIDR"

# ======TRANSFORMAR MASCARA DE CIDR A DECIMAL=======

# Obtener 1's
#for i in $( seq 1 $netmaskCIDR )
for (( i=1; i<=$netmaskCIDR; i++ ))
do
        unos+="1"
done

# Obtener 0's
cidrceros=$(( 32-$netmaskCIDR ))

#for i in $( seq 1 $cidrceros )
for (( i=1; i<=$cidrceros; i++ ))
do
        ceros+="0"
done

# Concatenar 1's con 0's
binario="$unos$ceros"

#echo $unos
#echo $ceros
#echo $binario

# Separar cada 8 digitos
oct1="$(echo "$binario" | cut -c 1-8)"
oct2="$(echo "$binario" | cut -c 9-16)"
oct3="$(echo "$binario" | cut -c 17-24)"
oct4="$(echo "$binario" | cut -c 25-32)"

#echo $oct1
#echo $oct2
#echo $oct3
#echo $oct4

# Transformar cada octeto de binario a decimal
#for i in $( seq 1 8 )
for i in {1..8}
do
#       echo $i
        peso=$(( 128/(2**$(( $i-1 ))) ))
#       echo $peso

        # PRIMER OCTETO
        posicion="$(echo "$oct1" | cut -c $i)"
#       echo $posicion
        valor=$(( $posicion*$peso ))
#       echo $valor
        sum=$(( $sum+$valor ))
```

(continues on next page)

(continued from previous page)

```
        # SEGUNDO OCTETO
        posicion2="$(echo "$oct2" | cut -c $i)"
        valor2=$(( $posicion2*$peso ))
        sum2=$(( $sum2+$valor2 ))

        # TERCER OCTETO
        posicion3="$(echo "$oct3" | cut -c $i)"
        valor3=$(( $posicion3*$peso ))
        sum3=$(( $sum3+$valor3 ))

        # CUARTO OCTETO
        posicion4="$(echo "$oct4" | cut -c $i)"
        valor4=$(( $posicion4*$peso ))
        sum4=$(( $sum4+$valor4 ))
done

#echo $sum
#echo $sum2
#echo $sum3
#echo $sum4

# Formar netmask
netmask="$sum.$sum2.$sum3.$sum4"
echo $netmask

# Variable que almacena el hosntame
hostn="$(hostname)"
echo "$hostn"

echo "network --bootproto=static --device=enp0s3 --ip=$ip --netmask=$netmask --
↪hostname=$hostn" > /tmp/network.ks

%end
```

- `%include` - (optional). Usar el comando `%include /path/to/file` para incluir los contenidos de otro archivo en el archivo Kickstart como si el contenido estuviera en la ubicación del comando `%include` en el archivo Kickstart.

Note: En este ejemplo, la línea `%include /tmp/network.ks` es reemplazada por `network --bootproto=static --device=enp0s3 --ip=$ip --netmask=$netmask --hostname=$hostn`

Note: Usar el parámetro `--bootproto=` con valor `static` configura el parámetro `BOOTPROTO="none"` en el archivo `/etc/sysconfig/network-scripts/ifcfg-enp0s3`, en lugar de `BOOTPROTO="dhcp"` (por defecto).

12.1.5 Archivo Kickstart - Post-install script

Caso de uso para un post-installation script:

Agregar una llave pública para habilitar Passwordless SSH hacia el cliente.

```
install
cdrom
bootloader --location=mbr
keyboard --vckeymap=latam --xlayouts='latam','us'
rootpw --iscrypted $1$WUDGBrnr$Bq8p.jk4ikcEr2JYJRMwE0
lang en_US.UTF-8 --addsupport=es_US.UTF-8
clearpart --all --initlabel
part / --fstype="ext4" --grow --ondisk=sda --size=1
timezone America/Lima
reboot

user --name=user1 --groups=wheel --iscrypted --password=$1$WUDGBrnr$Bq8p.
↪jk4ikcEr2JYJRMwE0

%post
#---- Install our SSH key ----
mkdir -m0700 /home/user1/.ssh/

# SSH public key of PXE boot server (SSH client)
cat <<EOF >/home/user1/.ssh/authorized_keys
ssh-rsa␣
↪AAAAB3NzaC1yc2EAAAADAQABAAACAQDblvZmYVA+ayprv7f25pS3+agpSGzDg0mlZxvxB1g46r7YSGiMDXphujEqE6FJAkecAC
↪os22YrA+aM3blNHcwh+OAjpfcdLnYuAo5YjsAuJ5/
↪1bI6+VOKN6BHQYNVxkkggkeYnrZwQUD0w054cAgpfMp2O7FJGJ5SXDKUR4w7y2gAEjOFe6X/
↪ULpRR0FWGFrDsO894XKnEBh3D2lhrdGZURwKJCtsbWxfO4XOm8P7JHdQK8z1xgKsTaJLotReh8lbwPptlLPGF3QjD0kvab9jqm8
↪xhByt/T6A3pOwMBrVEaClgOwQaz/
↪LiVFfyolyhO9OzOzFNXSw+Z5zZ7hj68lkji0gbjbxHMXNrHPhCMjts9m7dss/
↪LrYIpmNHpbUWwpL8OilS+vofkgESbpJNw+1/fPlwx0UIKiJ4hHQpqzaOcSsRXhmyrBj/
↪T1HWUyxUv34gmTynFTLNsNVMgWfBQTeqPruPQ== user1@localhost.localdomain
EOF

# set permissions
chmod 0600 /home/user1/.ssh/authorized_keys

# fix up selinux context
restorecon -R /home/user1/.ssh/

# change owner from root to user1
chown -R user1:user1 /home/user1/.ssh
%end
```

12.1.6 Archivo Kickstart - Text mode install

Archivo Kickstart con una configuración mínima en modo texto, en lugar del modo gráfico:

```
install
cdrom
bootloader --location=mbr
keyboard --vckeymap=latam --xlayouts='latam','us'
rootpw --iscrypted $1$WUDGBrnr$Bq8p.jk4ikcEr2JYJRMwE0
lang en_US.UTF-8 --addsupport=es_US.UTF-8
clearpart --all --initlabel
part / --fstype="ext4" --grow --ondisk=sda --size=1
timezone America/Lima
reboot
```

(continues on next page)

(continued from previous page)

```
text

%packages
%end
```

- `text` - (optional). Realizar la instalación con Kickstart en modo texto. Por defecto, las instalaciones se realizan en modo gráfico.

12.1.7 Archivo Kickstart - Medio pasado mediante URL

Archivo Kickstart con un medio de instalación pasado mediante una URL. En este caso se usa el protocolo FTP para enviar el medio al destino:

```
install
url --url="ftp://192.168.8.8/pub/centos7"
bootloader --location=mbr
keyboard --vckeymap=latam --xlayouts='latam','us'
rootpw --iscrypted $1$WUDGBrnr$Bq8p.jk4ikcEr2JYJRMwE0
lang en_US.UTF-8 --addsupport=es_US.UTF-8
clearpart --all --initlabel
part / --fstype="ext4" --grow --ondisk=sda --size=1
timezone America/Lima
reboot

%packages
%end
```

- `install` - (optional). Modo de instalación. Debe ir acompañado en otra línea del tipo de instalación: `cdrom`, `harddrive`, `nfs`, `liveimg`, `url` (FTP, HTTP, HTTPS).
- `--url=` - la ubicación de donde instalar. Soporta protocolos `HTTP`, `HTTPS`, `FTP` y `file`.

12.1.8 Archivo Kickstart - esquema de particiones personalizado

Archivo Kickstart con un esquema de particiones más desarrollado. Un mismo disco de 8 GB será particionado en 3 partes:

- partición 1 (`/`): 5 GB
- partición 2: partición para swap del tamaño recomendado por el instalador
- partición 3 (`/home`): ocupar el tamaño restante del disco

```
install
cdrom
bootloader --location=mbr
keyboard --vckeymap=latam --xlayouts='latam','us'
rootpw --iscrypted $1$WUDGBrnr$Bq8p.jk4ikcEr2JYJRMwE0
lang en_US.UTF-8 --addsupport=es_US.UTF-8
clearpart --all --initlabel
part / --fstype="ext4" --ondisk=sda --asprimary --size=5120
part /home --fstype="ext4" --grow --ondisk=sda --size=1
part swap --fstype="swap" --recommended
timezone America/Lima
reboot
```

(continues on next page)

(continued from previous page)

```
%packages
%end
```

El esquema de particiones generado a partir de este archivo Kickstart es el siguiente:

```
$ lsblk

NAME   MAJ:MIN RM   SIZE RO TYPE MOUNTPOINT
sda      8:0    0     8G  0 disk
├─sda1   8:1    0     5G  0 part /
├─sda2   8:2    0   2.2G  0 part /home
└─sda3   8:3    0   820M  0 part [SWAP]
sr0     11:0    1   942M  0 rom
```

12.1.9 Archivo Kickstart - configuración de seguridad

Archivo Kickstart con una configuración personalizada del Firewall y SELinux del sistema:

```
install
cdrom
bootloader --location=mbr
keyboard --vckeymap=latam --xlayouts='latam','us'
rootpw --iscrypted $1$WUDGBrnr$Bq8p.jk4ikcEr2JYJRMwE0
lang en_US.UTF-8 --addsupport=es_US.UTF-8
clearpart --all --initlabel
part / --fstype="ext4" --grow --ondisk=sda --size=1
timezone America/Lima
reboot

firewall --disabled
selinux --enforcing

%packages
%end
```

- **`firewall` - (optional). Especificar la configuración de firewall para el sistema instalado.**
  - `--enabled` - (o `--enable`). Rechaza conexiones entrantes que no responden a solicitudes salientes, como respuestas DNS o solicitudes DHCP. Si se necesita el acceso a servicios que corren en esta máquina, podemos escoger permitir servicios específicos a través del firewall.
  - `--disabled` - (o `--disable`). No configurar ninguna regla de iptables.

```
firewall --enabled|--disabled device [options]
```

Note: Revisar el estado del firewall con `systemctl status firewalld` y ver las reglas de iptables con `sudo iptables -S`.

- **`selinux` - (optional). Configura el estado de SELinux en los sistemas instalados. La política por defecto de SELinux es `e`**
  - `--enforcing` - Habilita SELinux con la política objetivo en estado `enforcing`.

- `--permissive` - Entrega precauciones basados en la política de SELinux, pero en realidad no hace cumplir (enforce) la política.
- `--disabled` - Deshabilita SELinux completamente en el sistema.

```
selinux [--disabled|--enforcing|--permissive]
```

Note: Revisar las políticas de seguridad de SELinux con el comando `getenforce` o leyendo el archivo `/etc/selinux/config`.

12.1.10 Archivo Kickstart - múltiples opciones

Archivo kickstart que agrupa distintos parámetros ya revisados en los anteriores ejemplos:

```
install
url --url="ftp://192.168.8.8/pub/centos7"
bootloader --location=mbr
keyboard --vckeymap=latam --xlayouts='latam','us'
rootpw --iscrypted $1$WUDGBrnr$Bq8p.jk4ikcEr2JYJRMwE0
lang en_US.UTF-8 --addsupport=es_US.UTF-8
clearpart --all --initlabel
part / --fstype="ext4" --ondisk=sda --asprimary --size=5120
part /home --fstype="ext4" --grow --ondisk=sda --size=1
part swap --fstype="swap" --recommended
timezone America/Lima
reboot

user --name=user1 --groups=wheel --iscrypted --password=$1$WUDGBrnr$Bq8p.
↪jk4ikcEr2JYJRMwE0

firewall --disabled
selinux --enforcing

text

# Incluir archivo generado en pre installation script
%include /tmp/network.ks

%packages
%end

%pre
#!/bin/bash

inet="$(ip addr show dev enp0s3 scope global | grep 'inet')"
#echo "$inet"

#ipdir="$(echo "$inet" | awk '{ print $2 }')"
ipdir="$(echo "$inet" | cut -d ' ' -f 6)"
#echo "$ipdir"

ip="$(echo "$ipdir" | cut -d / -f1)"
echo "$ip"

netmaskCIDR="$(echo "$ipdir" | cut -d / -f2)"
```

(continues on next page)

(continued from previous page)

```
#echo "$netmaskCIDR"

# ======TRANSFORMAR MASCARA DE CIDR A DECIMAL=======

# Obtener 1's
#for i in $( seq 1 $netmaskCIDR )
for (( i=1; i<=$netmaskCIDR; i++ ))
do
        unos+="1"
done

# Obtener 0's
cidrceros=$(( 32-$netmaskCIDR ))

#for i in $( seq 1 $cidrceros )
for (( i=1; i<=$cidrceros; i++ ))
do
        ceros+="0"
done

# Concatenar 1's con 0's
binario="$unos$ceros"

#echo $unos
#echo $ceros
#echo $binario

# Separar cada 8 digitos
oct1="$(echo "$binario" | cut -c 1-8)"
oct2="$(echo "$binario" | cut -c 9-16)"
oct3="$(echo "$binario" | cut -c 17-24)"
oct4="$(echo "$binario" | cut -c 25-32)"

#echo $oct1
#echo $oct2
#echo $oct3
#echo $oct4

# Transformar cada octeto de binario a decimal
#for i in $( seq 1 8 )
for i in {1..8}
do
#       echo $i
        peso=$(( 128/(2**$(( $i-1 ))) ))
#       echo $peso

        # PRIMER OCTETO
        posicion="$(echo "$oct1" | cut -c $i)"
#       echo $posicion
        valor=$(( $posicion*$peso ))
#       echo $valor
        sum=$(( $sum+$valor ))

        # SEGUNDO OCTETO
        posicion2="$(echo "$oct2" | cut -c $i)"
        valor2=$(( $posicion2*$peso ))
        sum2=$(( $sum2+$valor2 ))
```

(continues on next page)

(continued from previous page)

```
        # TERCER OCTETO
        posicion3="$(echo "$oct3" | cut -c $i)"
        valor3=$(( $posicion3*$peso ))
        sum3=$(( $sum3+$valor3 ))

        # CUARTO OCTETO
        posicion4="$(echo "$oct4" | cut -c $i)"
        valor4=$(( $posicion4*$peso ))
        sum4=$(( $sum4+$valor4 ))
done

#echo $sum
#echo $sum2
#echo $sum3
#echo $sum4

# Formar netmask
netmask="$sum.$sum2.$sum3.$sum4"
echo $netmask

# Variable que almacena el hosntame
hostn="$(hostname)"
echo "$hostn"

echo "network --bootproto=static --device=enp0s3 --ip=$ip --netmask=$netmask --
↪hostname=$hostn" > /tmp/network.ks

%end

%post
#---- Install our SSH key ----
mkdir -m0700 /home/user1/.ssh/

# SSH public key of PXE boot server (SSH client)
cat <<EOF >/home/user1/.ssh/authorized_keys
ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAABAQDgnCcOFXIm9hmohfEURM7cgE4usT2VbASeoRY4Y/
↪VzxIIgigv8C9+N4Rf4s+uj/
↪65oCys8fHJsC0KCQELl3ORDbBew6wT0YgPKU9D1NjXYXS4DTHy5pkLIVLN05Vs+cSyZphRB4ofuRYIhumJrGV7KwwkRw8cjOkI
↪9Sf94o+TgmjNI7cagQetJD+RbZmR3dW9I0pHYKddr/
↪TIWRkozZWNyaycVsBpBjQfPUYojENCJBaASRAvR21KpqHcxBWczSh8bc6AInAOEO2IpuJiXxZUkDcNX0BmwZvwcE6DsK8QlfYOy
↪vagrant@pxeserver
EOF

# set permissions
chmod 0600 /home/user1/.ssh/authorized_keys

# fix up selinux context
restorecon -R /home/user1/.ssh/

# change owner from root to user1
chown -R user1:user1 /home/user1/.ssh
%end
```

CHAPTER 13

Kickstart Options

13.1 Opciones de Kickstart - selección de paquetes

Table of Contents

- Referencia 1: RHEL 7 Package Groups
- Referencia 2: KICKSTART SYNTAX REFERENCE - Package Selection

13.1.1 Core Package Group

```
%packages
@core
%end
```

`@core` - 'Core' simple package group. Contiene aproximadamente 291 paquetes.

Note: Listar paquetes instalados en sistemas operativos basados en Red Hat:

```
$ rpm -qa
```

Referencia: Package list for Kickstart

Note: The `@Core` package group is defined as a minimal set of packages needed for installing a working system.

Referencia: KICKSTART SYNTAX REFERENCE - Package Selection

13.1.2 Default Package Groups

El grupo de paquetes `Core` es agregado por defecto cuando usamos el parámetro `%packages` en el archivo Kickstart. Por tanto, el siguiente bloque del archivo kickstart:

```
%packages
%end
```

Es equivalente a usar el siguiente bloque en el archivo kickstart:

```
%packages
@core
%end
```

Note: The Core group is always selected - it is not necessary to specify it in the %packages section.

Referencia: KICKSTART SYNTAX REFERENCE - Package Selection

13.1.3 Minimal Install Environment Group

```
%packages
@^minimal
%end
```

`@^minimal` - 'Minimal install' environment group. Contiene el mandatory group `core`, que a su vez se conforma de 291 paquetes.

Listar groups (Mandatory y Optional) que contiene el environment group 'Minimal install':

```
'#' yum group info "Minimal Install"

Loaded plugins: fastestmirror, langpacks
Loading mirror speeds from cached hostfile
...
Environment Group: Minimal Install
 Environment-Id: minimal
 Description: Basic functionality.
 Mandatory Groups:
   +core
 Optional Groups:
   +debugging
```

Note: El prefijo `@^` está reservado para environment groups

Important: If you are not sure what package should be installed, Red Hat recommends you to select the Minimal Install environment. Minimal install provides only the packages which are essential for running Red Hat Enterprise Linux 7.

Referencia: KICKSTART SYNTAX REFERENCE - Package Selection

13.2 Opciones de Kickstart- distribución de teclado

Table of Contents

- Referencia 1: How To Change The System Keyboard Layout
- Referencia 2: Red Hat Docs - CHANGING THE KEYBOARD LAYOUT
- Referencia 3: What is VC keymap

13.2.1 Parámetros kickstart - keyboard

Usando los siguientes parámetros en el archivo kickstart para la distribución del teclado:

```
keyboard --vckeymap=latam --xlayouts='latam','us'
```

Obtendremos el siguiente resultado:

```
$ localectl status

System Locale: LANG=en_US.UTF-8
    VC Keymap: latam
   X11 Layout: latam,us
  X11 Variant: ,
```

En el anterior output vemos la configuración del keyboard layout para la consola virtual y el X11 window system.

Note: Usando los siguientes parámetros en el archivo kickstart para la distribución del teclado:

```
keyboard --vckeymap=latam
```

Obtendremos el siguiente resultado:

```
$ localectl status

System Locale: LANG=en_US.UTF-8
    VC Keymap: latam
   X11 Layout: us
```

13.2.2 Listar keymaps

Para listar todos los keymaps disponibles podemos hacerlo de 2 formas:

1. Con el comando `localectl`:

```
$ localectl list-keymaps | grep latam

latam
latam-deadtilde
latam-dvorak
latam-nodeadkeys
latam-sundeadkeys
```

2. Listando el directorio `/usr/lib/kbd/keymaps/xkb/` (para usar un layout de esta lista debemos quitarle la última parte `.map.gz`):

```
$ ls /usr/lib/kbd/keymaps/xkb/ | grep latam

latam.map.gz
latam-deadtilde.map.gz
latam-dvorak.map.gz
latam-nodeadkeys.map.gz
latam-sundeadkeys.map.gz
```

13.2.3 Cambiar el keyboard layout

Para cambiar el keyboard layout usaremos la herramienta `localectl` tanto para **VC Keymap** como para **X11 window system**:

- Para cambiar el keyboard layout de **VC Keymap**:

```
$ localectl set-keymap us

$ localectl status

System Locale: LANG=en_US.UTF-8
    VC Keymap: us
   X11 Layout: us
    X11 Model: pc105+inet
  X11 Options: terminate:ctrl_alt_bksp
```

- Para cambiar el keyboard layout de **X11 window system**

```
$ localectl set-x11-keymap latam

$ localectl status

System Locale: LANG=en_US.UTF-8
    VC Keymap: latam
   X11 Layout: latam
```

Note: Como vemos, al cambiar el keyboard layout usando `set-x11-keymap` hemos cambiado tanto la distribución de **VC Keymap** como de **X11 window system**. Para cambiar exclusivamente el keyboard layout de X11 window system usamos el parámetro `--no-convert`:

```
$ localectl --no-convert set-x11-keymap us

$ localectl status

System Locale: LANG=en_US.UTF-8
    VC Keymap: latam
   X11 Layout: us
```

13.3 Opciones de Kickstart - contraseña root

Table of Contents

- Referencia 1: Red Hat Docs - KICKSTART SYNTAX REFERENCE
- Referencia 2: Generate Password Hash in Linux

Para establecer una contraseña encriptada para root usamos la siguiente línea en el archivo kickstart:

```
rootpw --iscrypted $1$WUDGBrnr$Bq8p.jk4ikcEr2JYJRMwE0
```

13.3.1 Crear una contraseña encriptada

Podemos usar Python con el fin de generar una contraseña encriptada:

```
$ python -c 'import crypt,getpass;pw=getpass.getpass();print(crypt.crypt(pw) if␣
↪(pw==getpass.getpass("Confirm: ")) else exit())'
```

Este comando genera un hash SHA512 de nuestra contraseña usando un SALT aleatorio.

Si obtenemos algún error con el comando anterior tenemos otras opciones:

- Generar una contraseña hash MD5:

```
$ python -c "import random,string,crypt; randomsalt = ''.join(random.sample(string.
↪ascii_letters,8)); print crypt.crypt('MySecretPassword', '\$1\$%s\$' % randomsalt)"
```

- Generar una contraseña hash SHA-256:

```
$ python -c "import random,string,crypt; randomsalt = ''.join(random.sample(string.
↪ascii_letters,8)); print crypt.crypt('MySecretPassword', '\$5\$%s\$' % randomsalt)"
```

- Generar una contraseña hash SHA-512:

```
$ python -c "import random,string,crypt; randomsalt = ''.join(random.sample(string.
↪ascii_letters,8)); print crypt.crypt('MySecretPassword', '\$6\$%s\$' % randomsalt)"
```

Note: Cambiar `MySecretPassword` por nuestra contraseña que deseamos establecer.

13.4 Opciones de Kickstart - pre-installation script

Table of Contents

- Referencia 1: Red Hat Docs - KICKSTART SYNTAX REFERENCE - Pre-installation Script
- Referencia 2: CentOS 7 Docs - Kickstart Installations - Pre-installation Script
- Referencia 3: Red Hat Docs - PRE-INSTALLATION SCRIPT
- Referencia 4: What commands are available in the pre section of a Kickstart file on CentOS
- Referencia 5: Pre-installation Script Example

El script `%pre` se corre en el sistema inmediatamente luego de que el archivo Kickstart haya analizado la sintáxis (parsed), pero antes que comience la instalación. Esta sección debe ser colocada al final del archivo kickstart, luego de los comandos de kickstart comunes; y debe comenzar con el comando `%pre` y acabar con `%end`. Si nuestro archivo kickstart incluye también una sección `%post`, el orden de las secciones `%pre` y `%post` no es relevante.

El script `%pre` puede usarse para la activación y configuración de dispositivos de networking y storage. También es posible correr scripts, usando intérpretes disponibles en el entorno de instalación. Añadir un script `%pre` puede ser útil si tenemos networking o almacenamiento que necesita una configuración especial previa a la instalación, o tener un script que, por ejemplo, configura parámetros de logging o variables de entorno adicionales.

Podemos acceder a parámetros de red en la sección `%pre`; sin embargo, el *name service* no ha sido configurado hasta este punto. Por lo cual, solo funcionarán direcciones IP, no URLs.

Note: A diferencia del post-installation script, el pre-installation script no corre en el entorno `chroot`.

Las siguientes opciones pueden usarse para cambiar el comportamiento de pre-installation scripts. Para usar una opción adjuntarlo en la línea de `%pre` al inicio del script. Por ejemplo:

```
%pre --interpreter=/usr/bin/python
--- Python script omitted --
%end
```

- `--interpreter=` - Permite especificar un lenguaje de scripting distinto, como Python. Cualquier lenguaje de scripting disponible en el sistema puede ser usado; en la mayoría de los casos, estos son `/usr/bin/sh`, `/usr/bin/bash`, `/usr/bin/python`.
- `erroronfail` - Mostrar un error y detener la instalación si el script falla. El mensaje de error nos dirigirá a donde la causa del fallo se ha registrado.
- `--log=` - Guardar el output del script en el archivo de log especificado. Por ejemplo:

```
%pre --log=/mnt/sysimage/root/ks-pre.log
```

Comandos disponibles para ser usados en pre-installation script:

- Referencia 3: Red Hat Docs - PRE-INSTALLATION SCRIPT
- Referencia 4: What commands are available in the pre section of a Kickstart file on CentOS

13.5 Opciones de Kickstart - post-installation script

Table of Contents

- Referencia 1: Red Hat Docs - KICKSTART SYNTAX REFERENCE - Post-installation Script

Tenemos la posibilidad de añadir comandos para correr en el sistema una vez que la instalación se ha completado, pero antes que el sistema sea reiniciado por primera vez. Esta sección debe ser colocada al final del archivo kickstart, luego de los comandos de kickstart comunes; y debe comenzar con el comando `%post` y acabar con `%end`. Si nuestro archivo kickstart incluye también una sección `%pre`, el orden de las secciones `%pre` y `%post` no es relevante.

Esta sección es útil para funciones, como instalar software adicional o configurar un name server adicional. El post-install script corre en un **chroot environment**, por tanto, realizar tareas como copiar scripts o paquetes RPM del medio de instalación no funcionan por defecto. Podemos cambiar este comportamiento usando la opción `--nochroot`. Además, en el entorno chroot, la mayoría de comandos `systemctl` se negarán a realizar cualquier acción.

Important: Si configuramos la red con información IP estática, incluyendo un name server, podemos acceder a la red y resolver direcciones IP en la sección `%post`. Si hemos configurado la red para DHCP, el archivo `/etc/resolv.conf` no ha sido completado cuando la instalación ejecute la sección `%post`. Podemos acceder la red, pero no podemos resolver direcciones IP. Por tanto, si estamos usando DHCP, debemos especificar direcciones IP en la sección `%post`.

Las siguientes opciones pueden usarse para cambiar el comportamiento de post-installation scripts. Para usar una opción, adjuntarlo en la línea de `%post` al inicio del script. Por ejemplo:

```
%post --interpreter=/usr/bin/python
--- Python script omitted --
%end
```

- `--interpreter=` - Permite especificar un lenguaje de scripting distinto, como Python. Cualquier lenguaje de scripting disponible en el sistema puede ser usado; en la mayoría de los casos, estos son `/usr/bin/sh`, `/usr/bin/bash`, `/usr/bin/python`.
- `--nochroot` - Pemite especificar comandos que deseamos sen ejecutados fuera del entorno chroot.

Por ejemplo, para copiar el archivo `/etc/resolv.conf` al file system que acaba de ser instalado usamos:

```
%post --nochroot
cp /etc/resolv.conf /mnt/sysimage/etc/resolv.conf
%end
```

- `erroronfail` - Mostrar un error y detener la instalación si el script falla. El mensaje de error nos dirigirá a donde la causa del fallo se ha registrado.

- `--log=` - Guardar el output del script en el archivo de log especificado. Por ejemplo:

```
%post --log=/mnt/sysimage/root/ks-post.log
```

con `--nochroot`:

```
%post --nochroot --log=/mnt/sysimage/root/ks-post.log
```

CHAPTER 14

Markdown

CHAPTER 15

OpenStack Guides

15.1 Proceso de instalación manual de OpenStack

Table of Contents

15.1.1 Documentación oficial de OpenStack

- OpenStack Documentation - Train Release
- OpenStack Installation Guide (any release): The following guide provides information about getting started, setting up your environment, and launching your instance.
- OpenStack Train Installation Guides: These documents cover installation procedures for OpenStack services.
- Install OpenStack services

OpenStack Environment (any release)

- **Environment**
  - Security
  - **Host networking**

        * Host networking - Controller node
        * Host networking - Compute node
        * Host networking - Block storage node (Optional)
        * Host networking - Verify connectivity
    - **Network Time Protocol (NTP)**
        * NTP - Controller node
        * NTP - Other nodes
        * NTP - Verify operation
    - **OpenStack packages**
        * OpenStack packages for SUSE
        * OpenStack packages for RHEL and CentOS
        * OpenStack packages for Ubuntu
    - **SQL database**
        * SQL database for SUSE
        * SQL database for RHEL and CentOS
        * SQL database for Ubuntu
    - **Message queue**
        * Message queue for SUSE
        * Message queue for RHEL and CentOS
        * Message queue for Ubuntu
    - **Memcached**
        * Memcached for SUSE
        * Memcached for RHEL and CentOS
        * Memcached for Ubuntu
    - **Etcd**
        * Etcd for SUSE
        * Etcd for RHEL and CentOS
        * Etcd for Ubuntu

OpenStack Services (Train)

Instalar los servicios de OpenStack respetando el siguiente orden:

- Keystone Installation
- Glance Installation
- Placement Installation
- Nova Installation
- Neutron Installation

- Cinder Installation
- Horizon Installation

15.1.2 Archivos de configuración

A continuación se presenta una lista de archivos que deben ser configurados en el proceso de instalación de OpenStack y sus servicios:

- **Environment**
  - Security, Networking, Install Linux Utilities: `/etc/hosts`, `/etc/default/grub`, `/etc/netplan/50-cloud-init.yaml`
  - NTP: `/etc/chrony/chrony.conf`
  - SQL Database: `/etc/mysql/mariadb.conf.d/99-openstack.cnf`
  - Memcached: `/etc/memcached.conf`
  - Etcd: `/etc/default/etcd`, `/etc/etcd/etcd.conf.yml`, `/lib/systemd/system/etcd.service`
- **Keystone**
  - `/etc/keystone/keystone.conf` [controller]
  - `/etc/apache2/apache2.conf` [controller]
  - `~/admin-openrc` [controller]
  - `~/demo-openrc` [controller]
- **Glance**
  - `/etc/glance/glance-api.conf` [controller]
  - `/etc/glance/glance-registry.conf` [controller]
- **Nova**
  - `/etc/nova/nova.conf` [controller]
  - `/etc/nova/nova.conf` [compute]
  - `/etc/nova/nova-compute.conf` [compute]
- **Neutron**
  - `/etc/neutron/neutron.conf` [controller]
  - `/etc/neutron/plugins/ml2/ml2_conf.ini` [controller]
  - `/etc/neutron/plugins/ml2/linuxbridge_agent.ini` [controller]
  - `/etc/neutron/l3_agent.ini` [controller]
  - `/etc/neutron/dhcp_agent.ini` [controller]
  - `/etc/neutron/metadata_agent.ini` [controller]
  - `/etc/nova/nova.conf` [controller]
  - `/etc/neutron/neutron.conf` [compute]
  - `/etc/neutron/plugins/ml2/linuxbridge_agent.ini` [compute]
  - `/etc/nova/nova.conf` [compute]

- **Cinder**
  - `/etc/cinder/cinder.conf` [`block`]
  - `/etc/cinder/cinder.conf` [`controller`]
  - `/etc/nova/nova.conf` [`controller`]
- **Horizon**
  - `/etc/openstack-dashboard/local_settings.py` [`controller`]
  - `/etc/apache2/conf-available/openstack-dashboard.conf` [`controller`]

15.2 OpenStack Train - Manual Install - CentOS7 - VirtualBox

Table of Contents

Resumen: El siguiente procedimiento consiste en la instalación de OpenStack versión Train en máquinas virtuales de VirtualBox con sistema operativo CentOS 7. Para la automatización del despliegue de máquinas virtuales usamos la herramienta Vagrant. El resto del proceso, que consiste en la instalación de herramientas y servicios de OpenStack, se hará manualmente.

Topología: La topología consta de 3 nodos: 1 nodo controller, 1 nodo compute y 1 nodo block.

15.2.1 Scripts y pasos para el despliegue

Los siguientes scripts contienen los pasos y comandos usados a lo largo de esta guía:

- Vagrantfile: `Descargar Vagrantfile`
- Script de controller: `Descargar controller_setup.sh`
- Script de compute1: `Descargar compute1_setup.sh`
- Script de block1: `Descargar block1_setup.sh`
- Archivo de pasos para desplegar el entorno: `Descargar environment-install.sh`
- Archivo de pasos para instalar servicios de OpenStack: `Descargar openstack-train-centos-services-install.sh`

15.2.2 Definición del archivo `Vagrantfile`

```
# -*- mode: ruby -*-
# vi: set ft=ruby :
servers=[
{
    :hostname => "controller",
    :box => "geerlingguy/centos7",
    :ram => 6144,
    :cpu => 2,
    :script => "sh /vagrant/controller_setup.sh"
},
{
```

(continues on next page)

(continued from previous page)

```
    :hostname => "compute1",
    :box => "geerlingguy/centos7",
    :ram => 3072,
    :cpu => 1,
    :script => "sh /vagrant/compute1_setup.sh"
},
{
    :hostname => "block1",
    :box => "geerlingguy/centos7",
    :ram => 3072,
    :cpu => 1,
    :script => "sh /vagrant/block1_setup.sh"
}
]
# All Vagrant configuration is done below. The "2" in Vagrant.configure
# configures the configuration version (we support older styles for
# backwards compatibility). Please don't change it unless you know what
# you're doing.
Vagrant.configure("2") do |config|
servers.each do |machine|
    config.vm.define machine[:hostname] do |node|
    node.vm.box = machine[:box]
    node.vm.hostname = machine[:hostname]
    node.vm.provider "virtualbox" do |vb|
        vb.customize ["modifyvm", :id, "--memory", machine[:ram], "--cpus",␣
↪machine[:cpu]]
        vb.customize ["modifyvm", :id, "--nic2", "hostonly", "--hostonlyadapter2",
↪"VirtualBox Host-Only Ethernet Adapter #2"]
        vb.customize ["modifyvm", :id, "--nic3", "natnetwork", "--nat-network3",
↪"ProviderNetwork1", "--nicpromisc3", "allow-all"]
        vb.customize ['modifyvm', :id, '--nic4', "bridged", '--bridgeadapter4',
↪"Realtek PCIe GBE Family Controller"]
        #vb.customize ["modifyvm", :id, "--nic4", "natnetwork", "--nat-network4",
↪"NatNetwork1"]
    end
    node.vm.provision "shell", inline: machine[:script], privileged: true, run: "once"
    end
end
end
```

15.2.3 Definición de scripts de configuración de nodos

Script de `controller`

```
#! /bin/sh

export LANG=en_US.utf-8
export LC_ALL=en_US.utf-8

# Remove 127.0.1.1 controller, if present
cat << EOF > /etc/hosts
127.0.0.1 localhost
10.1.1.11 controller
10.1.1.31 compute1
```

(continues on next page)

(continued from previous page)

```
10.1.1.32 compute2
10.1.1.41 block1
EOF

# Cambio de enp0sX a ethY (requiere reinicio)

## Ubuntu 18
#sed -i 's/GRUB_CMDLINE_LINUX=""/GRUB_CMDLINE_LINUX="net.ifnames=0 biosdevname=0"/g' /
↪etc/default/grub
#update-grub

## CentOS 7
sed -i 's/GRUB_CMDLINE_LINUX="/GRUB_CMDLINE_LINUX="net.ifnames=0 biosdevname=0 /g' /
↪etc/default/grub
grub2-mkconfig -o /boot/grub2/grub.cfg

# Configuración de interfaz red (Host-only adapter)
cat <<- EOF > /etc/sysconfig/network-scripts/ifcfg-eth1
DEVICE="eth1"
TYPE="Ethernet"
ONBOOT="yes"
BOOTPROTO="none"
IPADDR="10.1.1.11"
NETMASK="255.255.255.0"
EOF

# Reinicio de interfaz de red (Host-only adapter)
ifdown eth1
ifup eth1

# Configuración de interfaz red (Bridged adapter)
cat <<- EOF > /etc/sysconfig/network-scripts/ifcfg-eth3
DEVICE="eth3"
TYPE="Ethernet"
ONBOOT="yes"
BOOTPROTO="none"
IPADDR="192.168.1.111"
NETMASK="255.255.255.0"
EOF

# Reinicio de interfaz de red (Bridged adapter)
ifdown eth3
ifup eth3

# Reiniciar el sistema
reboot
```

Script de `compute1`

```
#! /bin/sh

export LANG=en_US.utf-8
export LC_ALL=en_US.utf-8

# Remove 127.0.1.1 controller, if present
```

(continues on next page)

(continued from previous page)

```
cat << EOF > /etc/hosts
127.0.0.1 localhost
10.1.1.11 controller
10.1.1.31 compute1
10.1.1.32 compute2
10.1.1.41 block1
EOF

# Cambio de enp0sX a ethY (requiere reinicio)

## Ubuntu 18
#sed -i 's/GRUB_CMDLINE_LINUX=""/GRUB_CMDLINE_LINUX="net.ifnames=0 biosdevname=0"/g' /
↪etc/default/grub
#update-grub

## CentOS 7
sed -i 's/GRUB_CMDLINE_LINUX="/GRUB_CMDLINE_LINUX="net.ifnames=0 biosdevname=0 /g' /
↪etc/default/grub
grub2-mkconfig -o /boot/grub2/grub.cfg

# Configuración de interfaz red (Host-only adapter)
cat <<- EOF > /etc/sysconfig/network-scripts/ifcfg-eth1
DEVICE="eth1"
TYPE="Ethernet"
ONBOOT="yes"
BOOTPROTO="none"
IPADDR="10.1.1.31"
NETMASK="255.255.255.0"
EOF

# Reinicio de interfaz de red (Host-only adapter)
ifdown eth1
ifup eth1

# Configuración de interfaz red (Bridged adapter)
cat <<- EOF > /etc/sysconfig/network-scripts/ifcfg-eth3
DEVICE="eth3"
TYPE="Ethernet"
ONBOOT="yes"
BOOTPROTO="none"
IPADDR="192.168.1.131"
NETMASK="255.255.255.0"
EOF

# Reinicio de interfaz de red (Bridged adapter)
ifdown eth3
ifup eth3

# Reiniciar el sistema
reboot
```

Script de `block1`

```
#! /bin/sh
```

(continues on next page)

(continued from previous page)

```
export LANG=en_US.utf-8
export LC_ALL=en_US.utf-8

# Remove 127.0.1.1 controller, if present
cat << EOF > /etc/hosts
127.0.0.1 localhost
10.1.1.11 controller
10.1.1.31 compute1
10.1.1.32 compute2
10.1.1.41 block1
EOF

# Cambio de enp0sX a ethY (requiere reinicio)

## Ubuntu 18
#sed -i 's/GRUB_CMDLINE_LINUX=""/GRUB_CMDLINE_LINUX="net.ifnames=0 biosdevname=0"/g' /
↪etc/default/grub
#update-grub

## CentOS 7
sed -i 's/GRUB_CMDLINE_LINUX="/GRUB_CMDLINE_LINUX="net.ifnames=0 biosdevname=0 /g' /
↪etc/default/grub
grub2-mkconfig -o /boot/grub2/grub.cfg

# Configuración de interfaz red (Host-only adapter)
cat <<- EOF > /etc/sysconfig/network-scripts/ifcfg-eth1
DEVICE="eth1"
TYPE="Ethernet"
ONBOOT="yes"
BOOTPROTO="none"
IPADDR="10.1.1.41"
NETMASK="255.255.255.0"
EOF

# Reinicio de interfaz de red (Host-only adapter)
ifdown eth1
ifup eth1

# Configuración de interfaz red (Bridged adapter)
cat <<- EOF > /etc/sysconfig/network-scripts/ifcfg-eth3
DEVICE="eth3"
TYPE="Ethernet"
ONBOOT="yes"
BOOTPROTO="none"
IPADDR="192.168.1.141"
NETMASK="255.255.255.0"
EOF

# Reinicio de interfaz de red (Bridged adapter)
ifdown eth3
ifup eth3

# Reiniciar el sistema
reboot
```

15.2.4 Instalación y configuración de herramientas de entorno

Verificación previa

Verificar que el servicio de firewall esté desactivado:

```
# Verificar que el servicio de firewall esté desactivado
systemctl status firewalld
sudo firewall-cmd --state
```

- https://docs.openstack.org/install-guide/environment-networking-controller.html
- https://docs.openstack.org/install-guide/environment-networking-compute.html
- https://docs.openstack.org/install-guide/environment-networking-storage-cinder.html

Network Time Protocol (NTP)

- https://docs.openstack.org/install-guide/environment-ntp.html

Nodo(s)

CONTROLLER

- https://docs.openstack.org/install-guide/environment-ntp-controller.html

```
# Instalar el programa chrony:
sudo -i
yum install -y chrony

# Editar el archivo de configuración de chrony:
cat << EOF >> /etc/chrony.conf
allow 10.1.1.0/24
EOF

# Reiniciar el servicio de NTP:
systemctl restart chronyd.service
```

Nodo(s)

COMPUTE, BLOCK

- https://docs.openstack.org/install-guide/environment-ntp-other.html

```
# Instalar el programa chrony:
sudo -i
yum install -y chrony

# Configurar el nodo controller como servidor NTP:
cat << EOF >> /etc/chrony.conf
server controller iburst
EOF

# Comentar las líneas con los otros servidores NTP:
```

(continues on next page)

(continued from previous page)

```
sed -i 's/server 0.centos.pool.ntp.org iburst/#server 0.centos.pool.ntp.org iburst/g'␣
↪/etc/chrony.conf
sed -i 's/server 1.centos.pool.ntp.org iburst/#server 1.centos.pool.ntp.org iburst/g'␣
↪/etc/chrony.conf
sed -i 's/server 2.centos.pool.ntp.org iburst/#server 2.centos.pool.ntp.org iburst/g'␣
↪/etc/chrony.conf
sed -i 's/server 3.centos.pool.ntp.org iburst/#server 3.centos.pool.ntp.org iburst/g'␣
↪/etc/chrony.conf

# Reiniciar el servicio de NTP:
systemctl restart chronyd.service
```

Nodo(s)

ALL

- https://docs.openstack.org/install-guide/environment-ntp-verify.html

```
# Verificar que el nodo controller se sincronice con servidores externos y los demás␣
↪nodos se sincronicen con el controller:
chronyc sources
```

Environment packages

- https://docs.openstack.org/install-guide/environment-packages-rdo.html

Nodo(s)

ALL

```
# Verificar la versión de kernel actual
uname -mrs
# Verificar los kernels instalados en el sistema (el kernel subrayado es el que está␣
↪en uso)
yum list installed kernel

# Habilitar el repositorio de OpenStack Train
yum install -y centos-release-openstack-train

# Actualizar los paquetes
yum update -y

# Verificar si se ha descargado un kernel más nuevo:
yum list installed kernel

# Upgrade de los paquetes:
yum upgrade -y

# Reiniciar el sistema en caso se haya descargado un nuevo kernel:
reboot

# Verificar que estamos usando el nuevo kernel descargado:
```

(continues on next page)

(continued from previous page)

```
sudo -i
uname -mrs

# Instalar el cliente de OpenStack (No existe python3-openstackclient en yum)
yum install -y python-openstackclient

# Instalar el paquete para SELinux
yum install -y openstack-selinux

# Instalar otros paquetes adicionales
yum install -y crudini vim curl
```

SQL Database

- https://docs.openstack.org/install-guide/environment-sql-database-rdo.html

Nodo(s)

CONTROLLER

```
# Instalar la base de datos SQL, en nuestro caso ``mariadb``:
yum install -y mariadb mariadb-server python2-PyMySQL

# Crear y editar el archivo de configuración de MariaDB:
cat << EOF > /etc/my.cnf.d/openstack.cnf
[mysqld]
bind-address = 10.1.1.11

default-storage-engine = innodb
innodb_file_per_table = on
max_connections = 4096
collation-server = utf8_general_ci
character-set-server = utf8
EOF

# Habilitar e iniciar el servicio de base de datos:
systemctl enable mariadb.service
systemctl start mariadb.service

# Asegurar el servicios de base de datos:
mysql_secure_installation

# Enter current password for root (enter for none):
# Set root password? [Y/n]
# New password: openstack
# Re-enter new password: openstack
# Remove anonymous users? [Y/n]
# Disallow root login remotely? [Y/n] n
# Remove test database and access to it? [Y/n]
# Reload privilege tables now? [Y/n]
```

Message Queue - RabitMQ

- https://docs.openstack.org/install-guide/environment-messaging-rdo.html

Nodo(s)

CONTROLLER

```
# Instalar el paquete de RabbitMQ:
yum -y install rabbitmq-server

# Habilitar e iniciar el servicio de cola de mensajería:
systemctl enable rabbitmq-server.service
systemctl start rabbitmq-server.service

# Añadir el usuario a RabbitMQ (rabbitmqctl add_user openstack RABBIT_PASS):
rabbitmqctl add_user openstack openstack

# Configurar los permisos para el usuario openstack:
rabbitmqctl set_permissions openstack ".*" ".*" ".*"
```

Memcached

- https://docs.openstack.org/install-guide/environment-memcached-rdo.html

Nodo(s)

CONTROLLER

```
# Instalar el paquete de Memcached:
yum install -y memcached python-memcached

# Cambiar '-l 127.0.0.1' por '-l 10.1.1.11' en el archivo /etc/sysconfig/memcached
sed -i 's/-l 127.0.0.1/-l 10.1.1.11/g' /etc/sysconfig/memcached

# Habilitar e iniciar el servicio Memcached
systemctl enable memcached.service
systemctl start memcached.service
```

Etcd

- https://docs.openstack.org/install-guide/environment-etcd-rdo.html

Nodo(s)

CONTROLLER

```
# Instalar el paquete de etcd:
yum install -y etcd
```

(continues on next page)

(continued from previous page)

```
# Editar el archivo /etc/etcd/etcd.conf
cat << EOF >> /etc/etcd/etcd.conf
#[Member]
ETCD_DATA_DIR="/var/lib/etcd/default.etcd"
ETCD_LISTEN_PEER_URLS="http://10.1.1.11:2380"
ETCD_LISTEN_CLIENT_URLS="http://10.1.1.11:2379"
ETCD_NAME="controller"
#[Clustering]
ETCD_INITIAL_ADVERTISE_PEER_URLS="http://10.1.1.11:2380"
ETCD_ADVERTISE_CLIENT_URLS="http://10.1.1.11:2379"
ETCD_INITIAL_CLUSTER="controller=http://10.1.1.11:2380"
ETCD_INITIAL_CLUSTER_TOKEN="etcd-cluster-01"
ETCD_INITIAL_CLUSTER_STATE="new"
EOF

# Habilitar e iniciar el servicio etcd
systemctl enable etcd
systemctl start etcd
```

15.2.5 Instalación de servicios de OpenStack

Install and Configure Keystone

- https://docs.openstack.org/keystone/train/install/keystone-install-rdo.html

---

Nodo(s)

CONTROLLER

```
# Conectarnos al servidor de base de datos como usuario root:
mysql -u root --password=openstack

# Crear la base de datos de keystone:
CREATE DATABASE keystone;
# Brindar el acceso apropiado a la base de datos de keystone:
GRANT ALL PRIVILEGES ON keystone.* TO 'keystone'@'localhost' IDENTIFIED BY 'openstack
↪';
GRANT ALL PRIVILEGES ON keystone.* TO 'keystone'@'%' IDENTIFIED BY 'openstack';
EXIT;

# Instalar los paquetes de Keystone (además de crudini para la configuración de␣
↪archivos):
yum install -y openstack-keystone httpd mod_wsgi crudini

# Editar el archivo /etc/keystone/keystone.conf:
crudini --set /etc/keystone/keystone.conf database connection mysql+pymysql://
↪keystone:openstack@controller/keystone
# Comentar cualquier otra opción en la sección [database]

# Configurar el proveedor de token Fernet:
crudini --set /etc/keystone/keystone.conf token provider fernet

# Poblar la base de datos del Identity service:
```

(continues on next page)

(continued from previous page)

```
su -s /bin/sh -c "keystone-manage db_sync" keystone

# Poblar los repositorios de Fernet:
keystone-manage fernet_setup --keystone-user keystone --keystone-group keystone
keystone-manage credential_setup --keystone-user keystone --keystone-group keystone

# Bootstrap del Identity service:
keystone-manage bootstrap --bootstrap-password openstack --bootstrap-admin-url http://
↪controller:5000/v3/ --bootstrap-internal-url http://controller:5000/v3/ --bootstrap-
↪public-url http://controller:5000/v3/ --bootstrap-region-id RegionOne

# Configurr el servidor Apache HTTP:
cat << EOF >> /etc/httpd/conf/httpd.conf
ServerName controller
EOF
# Verificar que no exista otra entrada ServerName en el documento

# Crear un link simbólico al archivo /usr/share/keystone/wsgi-keystone.conf:
ln -s /usr/share/keystone/wsgi-keystone.conf /etc/httpd/conf.d/

# Habilitar e iniciar el servicio http
systemctl enable httpd.service
systemctl start httpd.service
```

Create OpenStack Client Environment Scripts

- https://docs.openstack.org/keystone/train/install/keystone-install-rdo.html#finalize-the-installation
- https://docs.openstack.org/keystone/train/install/keystone-openrc-rdo.html

---

Nodo(s)

CONTROLLER

```
# Crear el archivo admin-openrc en el directorio home del usuario
su -s /bin/sh -c 'cat << EOF > /home/vagrant/admin-openrc
export OS_USERNAME=admin
export OS_PASSWORD=openstack
export OS_PROJECT_NAME=admin
export OS_USER_DOMAIN_NAME=Default
export OS_PROJECT_DOMAIN_NAME=Default
export OS_AUTH_URL=http://controller:5000/v3
export OS_IDENTITY_API_VERSION=3
export OS_IMAGE_API_VERSION=2
EOF' vagrant

# Crear el archivo demo-openrc en el directorio home del usuario
su -s /bin/sh -c 'cat << EOF > /home/vagrant/demo-openrc
export OS_USERNAME=demo
export OS_PASSWORD=openstack
export OS_PROJECT_NAME=demo
export OS_USER_DOMAIN_NAME=Default
export OS_PROJECT_DOMAIN_NAME=Default
export OS_AUTH_URL=http://controller:5000/v3
```

(continues on next page)

(continued from previous page)

```
export OS_IDENTITY_API_VERSION=3
export OS_IMAGE_API_VERSION=2
EOF' vagrant

# Verificar la operación de Keystone:
. /home/vagrant/admin-openrc
openstack token issue
```

Create Projects, Users and Roles

- https://docs.openstack.org/keystone/train/install/keystone-users-rdo.html
- https://docs.openstack.org/keystone/train/install/keystone-verify-rdo.html

Nodo(s)

CONTROLLER

```
# Crear un nuevo dominio (por defecto existe el dominio default del paso keystone-
↪manage bootstrap):
openstack domain create --description "An Example Domain" example

# Crear un proyecto service:
openstack project create --domain default --description "Service Project" service

# Crear un proyecto sin privilegios para tareas no administrativas:
openstack project create --domain default --description "Demo Project" demo

# Crear un usuario sin privilegios para tareas no administrativas:
openstack user create --domain default --password openstack demo

# Crear un rol usuario
openstack role create user

# Asignar el rol user al usuario demo del proyecto demo:
openstack role add --project demo --user demo user

# Listar usuarios:
openstack user list

# Verificar la funcionalidad del usuario demo:
. /home/vagrant/demo-openrc
openstack token issue
```

Install & Configure Glance

- https://docs.openstack.org/glance/train/install/install-rdo.html

Nodo(s)

CONTROLLER

```
# Conectarnos al servidor de base de datos como usuario root:
mysql -u root --password=openstack

# Crear la base de datos de glance:
CREATE DATABASE glance;

# Brindar el acceso apropiado a la base de datos de glance:
GRANT ALL PRIVILEGES ON glance.* TO 'glance'@'localhost' IDENTIFIED BY 'openstack';
GRANT ALL PRIVILEGES ON glance.* TO 'glance'@'%' IDENTIFIED BY 'openstack';
EXIT;

# Crear un usuario glance:
. /home/vagrant/admin-openrc
openstack user create --domain default --password openstack glance

# Añadir el rol admin al usuario glance:
openstack role add --project service --user glance admin

# Crear el servicio glance:
openstack service create --name glance --description "OpenStack Image" image

# Crear los Glance Service Endpoints (public, internal, admin):
openstack endpoint create --region RegionOne image public http://controller:9292
openstack endpoint create --region RegionOne image internal http://controller:9292
openstack endpoint create --region RegionOne image admin http://controller:9292

# Instalar el paquete de glance:
yum update -y
yum install -y openstack-glance

# Configure database access for glance
crudini --set /etc/glance/glance-api.conf database connection mysql+pymysql://
↪glance:openstack@controller/glance

# Configurar el acceso a Identity Service:
crudini --set /etc/glance/glance-api.conf keystone_authtoken www_authenticate_uri␣
↪http://controller:5000
crudini --set /etc/glance/glance-api.conf keystone_authtoken auth_url http://
↪controller:5000
crudini --set /etc/glance/glance-api.conf keystone_authtoken memcached_servers␣
↪controller:11211
crudini --set /etc/glance/glance-api.conf keystone_authtoken auth_type password
crudini --set /etc/glance/glance-api.conf keystone_authtoken project_domain_name␣
↪default
crudini --set /etc/glance/glance-api.conf keystone_authtoken user_domain_name default
crudini --set /etc/glance/glance-api.conf keystone_authtoken project_name service
crudini --set /etc/glance/glance-api.conf keystone_authtoken username glance
crudini --set /etc/glance/glance-api.conf keystone_authtoken password openstack
crudini --set /etc/glance/glance-api.conf paste_deploy flavor keystone

# Configure Glance to store Images on Local Filesystem
crudini --set /etc/glance/glance-api.conf glance_store stores "file,http"
crudini --set /etc/glance/glance-api.conf glance_store default_store file
crudini --set /etc/glance/glance-api.conf glance_store filesystem_store_datadir /var/
↪lib/glance/images/

# Poblar la base de datos de Image Service:
```

(continues on next page)

(continued from previous page)

```
su -s /bin/sh -c "glance-manage db_sync" glance

# Habilitar e iniciar el servicio de glance
systemctl enable openstack-glance-api.service
systemctl start openstack-glance-api.service

# Glance Registry Service and its APIs have been DEPRECATED in the Queens release
```

Download & Create Test Images

- https://docs.openstack.org/glance/train/install/verify.html

---

Nodo(s)

CONTROLLER

```
# Descargar la imagen de CirrOS:
wget http://download.cirros-cloud.net/0.4.0/cirros-0.4.0-x86_64-disk.img

# Crear una nueva imagen:
. /home/vagrant/admin-openrc
openstack image create cirros4.0 --file cirros-0.4.0-x86_64-disk.img --disk-format␣
↪qcow2 --container-format bare --public

# Listar nuestras imágenes:
openstack image list
```

Placement service - Install and configure Placement

- https://docs.openstack.org/placement/train/install/install-rdo.html

---

Nodo(s)

CONTROLLER

```
# Conectarnos al servidor de base de datos como usuario root:
mysql -u root --password=openstack

# Crear la base de datos de placement:
CREATE DATABASE placement;

# Brindar el acceso apropiado a la base de datos de placement:
GRANT ALL PRIVILEGES ON placement.* TO 'placement'@'localhost' IDENTIFIED BY
↪'openstack';
GRANT ALL PRIVILEGES ON placement.* TO 'placement'@'%' IDENTIFIED BY 'openstack';
EXIT;

# Crear un usuario placement y añadir el rol admin al usuario placement:
. /home/vagrant/admin-openrc
openstack user create --domain default --password openstack placement
openstack role add --project service --user placement admin
```

(continues on next page)

(continued from previous page)

```
# Crear la entrada de Placement API en el catálogo de servicios:
openstack service create --name placement --description "Placement API" placement

# Crear los Placement API service endpoints:
openstack endpoint create --region RegionOne placement public http://controller:8778
openstack endpoint create --region RegionOne placement internal http://controller:8778
openstack endpoint create --region RegionOne placement admin http://controller:8778
# NOTE: Depending on your environment, the URL for the endpoint will vary by port␣
↪(possibly 8780 instead of 8778, or no port at all) and hostname. You are␣
↪responsible for determining the correct URL.

# Instalar paquetes:
yum install -y openstack-placement-api python-pip

# Editar el archivo /etc/placement/placement.conf:
crudini --set /etc/placement/placement.conf placement_database connection␣
↪mysql+pymysql://placement:openstack@controller/placement

crudini --set /etc/placement/placement.conf api auth_strategy keystone

crudini --set /etc/placement/placement.conf keystone_authtoken auth_url http://
↪controller:5000/v3
crudini --set /etc/placement/placement.conf keystone_authtoken memcached_servers␣
↪controller:11211
crudini --set /etc/placement/placement.conf keystone_authtoken auth_type password
crudini --set /etc/placement/placement.conf keystone_authtoken project_domain_name␣
↪Default
crudini --set /etc/placement/placement.conf keystone_authtoken user_domain_name␣
↪Default
crudini --set /etc/placement/placement.conf keystone_authtoken project_name service
crudini --set /etc/placement/placement.conf keystone_authtoken username placement
crudini --set /etc/placement/placement.conf keystone_authtoken password openstack
# Comentar cualquier otro parámetro en la sección [keystone_authtoken]

# Poblar la base de datos placement:
su -s /bin/sh -c "placement-manage db sync" placement
# El comando anterior no retorna ningún outut

# Reiniciar el servicio httpd:
systemctl restart httpd memcached

# Ver errores del log de Placement:
tail -f /var/log/placement/placement-api.log | grep -i error &
```

Placement service - Verify Installation

- https://docs.openstack.org/placement/train/install/verify.html

Nodo(s)

CONTROLLER

```
# Realizar revisión de estados para ver que todo esté en orden:
. /home/vagrant/admin-openrc
placement-status upgrade check

# Correr algunos comandos sobre el placement API:
pip install osc-placement
openstack --os-placement-api-version 1.2 resource class list --sort-column name
openstack --os-placement-api-version 1.6 trait list --sort-column name
```

Nova service - Install and configure controller node

- https://docs.openstack.org/nova/train/install/controller-install-rdo.html

Nodo(s)

CONTROLLER

```
# Conectarnos al servidor de base de datos como usuario root:
mysql -u root --password=openstack

# Crear las bases de datos:
CREATE DATABASE nova_api;
CREATE DATABASE nova;
CREATE DATABASE nova_cell0;

# Brindar el acceso apropiado a la base de datos de Nova:
GRANT ALL PRIVILEGES ON nova_api.* TO 'nova'@'localhost' IDENTIFIED BY 'openstack';
GRANT ALL PRIVILEGES ON nova_api.* TO 'nova'@'%' IDENTIFIED BY 'openstack';
GRANT ALL PRIVILEGES ON nova.* TO 'nova'@'localhost' IDENTIFIED BY 'openstack';
GRANT ALL PRIVILEGES ON nova.* TO 'nova'@'%' IDENTIFIED BY 'openstack';
GRANT ALL PRIVILEGES ON nova_cell0.* TO 'nova'@'localhost' IDENTIFIED BY 'openstack';
GRANT ALL PRIVILEGES ON nova_cell0.* TO 'nova'@'%' IDENTIFIED BY 'openstack';
EXIT;

# Crear un usuario nova y añadir el rol admin al usuario:
. /home/vagrant/admin-openrc
openstack user create --domain default --password openstack nova
openstack role add --project service --user nova admin

# Crear el servicio de Nova:
openstack service create --name nova --description "OpenStack Compute" compute

# Crear los Compute API Endpoints (public, internal, admin):
openstack endpoint create --region RegionOne compute public http://controller:8774/v2.
↪1
openstack endpoint create --region RegionOne compute internal http://controller:8774/
↪v2.1
openstack endpoint create --region RegionOne compute admin http://controller:8774/v2.1

# Instalar los paquetes de Nova Controller:
yum install -y openstack-nova-api openstack-nova-conductor openstack-nova-novncproxy␣
↪openstack-nova-scheduler

# Editar el archivo /etc/nova/nova.conf:
```

(continues on next page)

(continued from previous page)

```
crudini --set /etc/nova/nova.conf DEFAULT enabled_apis osapi_compute,metadata

crudini --set /etc/nova/nova.conf api_database connection mysql+pymysql://
↪nova:openstack@controller/nova_api
crudini --set /etc/nova/nova.conf database connection mysql+pymysql://
↪nova:openstack@controller/nova
crudini --set /etc/nova/nova.conf DEFAULT transport_url rabbit://
↪openstack:openstack@controller:5672/
# En Stein: transport_url = rabbit://openstack:RABBIT_PASS@controller

crudini --set /etc/nova/nova.conf api auth_strategy keystone

crudini --set /etc/nova/nova.conf keystone_authtoken www_authenticate_uri http://
↪controller:5000/
# En Stein: www_authenticate_uri no está presente
crudini --set /etc/nova/nova.conf keystone_authtoken auth_url http://controller:5000/
# En Stein: auth_url = http://controller:5000/v3
crudini --set /etc/nova/nova.conf keystone_authtoken memcached_servers␣
↪controller:11211
crudini --set /etc/nova/nova.conf keystone_authtoken auth_type password
crudini --set /etc/nova/nova.conf keystone_authtoken project_domain_name default
crudini --set /etc/nova/nova.conf keystone_authtoken user_domain_name default
crudini --set /etc/nova/nova.conf keystone_authtoken project_name service
crudini --set /etc/nova/nova.conf keystone_authtoken username nova
crudini --set /etc/nova/nova.conf keystone_authtoken password openstack

crudini --set /etc/nova/nova.conf DEFAULT my_ip 10.1.1.11
crudini --set /etc/nova/nova.conf DEFAULT use_neutron true
crudini --set /etc/nova/nova.conf DEFAULT firewall_driver nova.virt.firewall.
↪NoopFirewallDriver

crudini --set /etc/nova/nova.conf vnc enabled true
crudini --set /etc/nova/nova.conf vnc server_listen 10.1.1.11
crudini --set /etc/nova/nova.conf vnc server_proxyclient_address 10.1.1.11

crudini --set /etc/nova/nova.conf glance api_servers http://controller:9292

crudini --set /etc/nova/nova.conf oslo_concurrency lock_path /var/lib/nova/tmp

crudini --set /etc/nova/nova.conf placement region_name RegionOne
crudini --set /etc/nova/nova.conf placement project_domain_name Default
crudini --set /etc/nova/nova.conf placement project_name service
crudini --set /etc/nova/nova.conf placement auth_type password
crudini --set /etc/nova/nova.conf placement user_domain_name Default
crudini --set /etc/nova/nova.conf placement auth_url http://controller:5000/v3
crudini --set /etc/nova/nova.conf placement username placement
crudini --set /etc/nova/nova.conf placement password openstack

# Revisar si este parámetro se encuentra en el archivo de configuración. De ser el␣
↪caso, removerlo:
crudini --del /etc/nova/nova.conf DEFAULT log_dir

# Poblar la base de datos nova-api:
su -s /bin/sh -c "nova-manage api_db sync" nova

# Register cell0 Database:
su -s /bin/sh -c "nova-manage cell_v2 map_cell0" nova
```

(continues on next page)

(continued from previous page)

```
# Create cell1 Cell:
su -s /bin/sh -c "nova-manage cell_v2 create_cell --name=cell1 --verbose" nova

# Populate nova Database:
su -s /bin/sh -c "nova-manage db sync" nova

# Verificar que cell0 y cell1 han sido registradas correctamente:
su -s /bin/sh -c "nova-manage cell_v2 list_cells" nova

# Habilitar e iniciar servicios de Nova:
systemctl enable openstack-nova-api.service openstack-nova-scheduler.service␣
↪openstack-nova-conductor.service openstack-nova-novncproxy.service
systemctl start openstack-nova-api.service openstack-nova-scheduler.service openstack-
↪nova-conductor.service openstack-nova-novncproxy.service
```

Nova service - Install and configure a compute node

- https://docs.openstack.org/nova/train/install/compute-install-rdo.html

Nodo(s)

COMPUTE

```
# Instalar los paquetes de Nova (además de crudini para la configuración de archivos):
yum install -y openstack-nova-compute crudini

# Editar el archivo /etc/nova/nova.conf:
crudini --set /etc/nova/nova.conf DEFAULT enabled_apis osapi_compute,metadata

# Configurar acceso a RabbitMQ:
crudini --set /etc/nova/nova.conf DEFAULT transport_url rabbit://
↪openstack:openstack@controller

# Configurar el acceso a Identity Service:
crudini --set /etc/nova/nova.conf api auth_strategy keystone

crudini --set /etc/nova/nova.conf keystone_authtoken www_authenticate_uri http://
↪controller:5000/
# En Stein: www_authenticate_uri no está presente
crudini --set /etc/nova/nova.conf keystone_authtoken auth_url http://controller:5000/
# En Stein: auth_url = http://controller:5000/v3
crudini --set /etc/nova/nova.conf keystone_authtoken memcached_servers␣
↪controller:11211
crudini --set /etc/nova/nova.conf keystone_authtoken auth_type password
crudini --set /etc/nova/nova.conf keystone_authtoken project_domain_name default
crudini --set /etc/nova/nova.conf keystone_authtoken user_domain_name default
crudini --set /etc/nova/nova.conf keystone_authtoken project_name service
crudini --set /etc/nova/nova.conf keystone_authtoken username nova
crudini --set /etc/nova/nova.conf keystone_authtoken password openstack
# Comentar cualquier otro parámetro en la sección [keystone_authtoken]

# Reemplazar IP por la del compute node que se está configurando:
crudini --set /etc/nova/nova.conf DEFAULT my_ip 10.1.1.31
```

(continues on next page)

(continued from previous page)

```
crudini --set /etc/nova/nova.conf DEFAULT use _neutron true
crudini --set /etc/nova/nova.conf DEFAULT firewall_driver nova.virt.firewall.
↪NoopFirewallDriver

crudini --set /etc/nova/nova.conf vnc enabled true
crudini --set /etc/nova/nova.conf vnc server_listen 0.0.0.0
# Reemplazar IP por la del compute node que se está configurando:
crudini --set /etc/nova/nova.conf vnc server_proxyclient_address 10.1.1.31
crudini --set /etc/nova/nova.conf vnc novncproxy_base_url http://10.1.1.11:6080/vnc_
↪auto.html

crudini --set /etc/nova/nova.conf glance api_servers http://controller:9292

crudini --set /etc/nova/nova.conf oslo_concurrency lock_path /var/lib/nova/tmp

crudini --set /etc/nova/nova.conf placement region_name RegionOne
crudini --set /etc/nova/nova.conf placement project_domain_name Default
crudini --set /etc/nova/nova.conf placement project_name service
crudini --set /etc/nova/nova.conf placement auth_type password
crudini --set /etc/nova/nova.conf placement user_domain_name Default
crudini --set /etc/nova/nova.conf placement auth_url http://controller:5000/v3
crudini --set /etc/nova/nova.conf placement username placement
crudini --set /etc/nova/nova.conf placement password openstack

# Revisar si este parámetro se encuentra en el archivo de configuración. De ser el␣
↪caso, removerlo:
crudini --del /etc/nova/nova.conf DEFAULT log_dir

# Determinar si nuestra computadora soporta aceleración de hardware para máquinas␣
↪virtuales (si lo soporta el resultado será 1 o más):
egrep -c '(vmx|svm)' /proc/cpuinfo

# Para un setup con máquinas virtuales configurar:
crudini --set /etc/nova/nova-compute.conf libvirt virt_type qemu

# Habilitar e iniciar el servicio Compute:
systemctl enable libvirtd.service openstack-nova-compute.service
systemctl start libvirtd.service openstack-nova-compute.service
# If the nova-compute service fails to start, check /var/log/nova/nova-compute.log.␣
↪The error message AMQP server on controller:5672 is unreachable likely indicates␣
↪that the firewall on the controller node is preventing access to port 5672.␣
↪Configure the firewall to open port 5672 on the controller node and restart nova-
↪compute service on the compute node.
```

Discover Compute Nodes and Finalize Nova Installation

- https://docs.openstack.org/nova/train/install/compute-install-rdo.html#add-the-compute-node-to-the-cell-database

Nodo(s)

CONTROLLER

```
# Comprobar que hay compute nodes en la base de datos:
. /home/vagrant/admin-openrc
```

(continues on next page)

(continued from previous page)

```
openstack compute service list --service nova-compute

# Descubrir compute hosts:
su -s /bin/sh -c "nova-manage cell_v2 discover_hosts --verbose" nova

# Listar servicios de Compute:
openstack compute service list

# Ver errores del log de Nova:
tail -f /var/log/nova/* | grep -i error &
```

Solución del error "Skipping removal of allocations for deleted instances: Failed to retrieve allocations for resource provider"

Según las siguientes referencias:

- https://louky0714.tistory.com/entry/Openstack-Train-Error-Nova-Instance-fail-You-dont-have-permission-to-access-resourcepro
- http://www.programmersought.com/article/2585745102/
- https://blog.csdn.net/hutiewei2008/article/details/87971379
- https://ask.openstack.org/en/question/103325/ah01630-client-denied-by-server-configuration-usrbinnova-placement-api/

Seguimos el siguiente procedimiento:

1. Editar el archivo `/etc/httpd/conf.d/00-placement-api.conf`

```
vim /etc/httpd/conf.d/00-placement-api.conf
```

Añadir la siguiente sección (Probar al final del archivo o dentro de la sección `<virtualhost *:8778="">`):

```
<Directory /usr/bin>
    <IfVersion >= 2.4>
        Require all granted
    </IfVersion>
    <IfVersion < 2.4>
        Order allow,deny
        Allow from all
    </IfVersion>
</Directory>
```

2. Reiniciar servicios de Nova del controller node:

```
systemctl restart openstack-nova-api.service openstack-nova-scheduler.service␣
↪openstack-nova-conductor.service openstack-nova-novncproxy.service httpd
```

3. Reiniciar servicios de Nova del compute node:

```
systemctl restart libvirtd.service openstack-nova-compute.service
```

Error: Log con el error:

```
# ERROR:

2020-04-23 04:41:07.504 19347 ERROR nova.compute.resource_tracker [req-1c29a29d-
↪0f43-46c9-8255-d49f3ec1084b - - - - -] Skipping removal of allocations for␣
↪deleted instances: Failed to retrieve allocations for resource provider 5a5ae2e2-
↪063b-4311-9b24-23373417dc5b: <!DOCTYPE HTML PUBLIC "-//IETF//DTD HTML 2.0//EN">
2020-04-23 04:41:07.517 19347 ERROR nova.scheduler.client.report [req-1c29a29d-0f43-
↪46c9-8255-d49f3ec1084b - - - - -] [None] Failed to retrieve resource provider␣
↪tree from placement API for UUID 5a5ae2e2-063b-4311-9b24-23373417dc5b. Got 403: <!
↪DOCTYPE HTML PUBLIC "-//IETF//DTD HTML 2.0//EN">
2020-04-23 04:41:07.518 19347 ERROR nova.compute.manager [req-1c29a29d-0f43-46c9-
↪8255-d49f3ec1084b - - - - -] Error updating resources for node compute1.:␣
↪ResourceProviderRetrievalFailed: Failed to get resource provider with UUID␣
↪5a5ae2e2-063b-4311-9b24-23373417dc5b
2020-04-23 04:41:07.518 19347 ERROR nova.compute.manager Traceback (most recent␣
↪call last):
2020-04-23 04:41:07.518 19347 ERROR nova.compute.manager   File "/usr/lib/python2.7/
↪site-packages/nova/compute/manager.py", line 8673, in _update_available_resource_
↪for_node
2020-04-23 04:41:07.518 19347 ERROR nova.compute.manager     startup=startup)
2020-04-23 04:41:07.518 19347 ERROR nova.compute.manager   File "/usr/lib/python2.7/
↪site-packages/nova/compute/resource_tracker.py", line 887, in update_available_
↪resource
2020-04-23 04:41:07.518 19347 ERROR nova.compute.manager     self._update_available_
↪resource(context, resources, startup=startup)
2020-04-23 04:41:07.518 19347 ERROR nova.compute.manager   File "/usr/lib/python2.7/
↪site-packages/oslo_concurrency/lockutils.py", line 328, in inner
2020-04-23 04:41:07.518 19347 ERROR nova.compute.manager     return f(*args,␣
↪**kwargs)
2020-04-23 04:41:07.518 19347 ERROR nova.compute.manager   File "/usr/lib/python2.7/
↪site-packages/nova/compute/resource_tracker.py", line 972, in _update_available_
↪resource
2020-04-23 04:41:07.518 19347 ERROR nova.compute.manager     self._update(context,␣
↪cn, startup=startup)
2020-04-23 04:41:07.518 19347 ERROR nova.compute.manager   File "/usr/lib/python2.7/
↪site-packages/nova/compute/resource_tracker.py", line 1237, in _update
2020-04-23 04:41:07.518 19347 ERROR nova.compute.manager     self._update_to_
↪placement(context, compute_node, startup)
2020-04-23 04:41:07.518 19347 ERROR nova.compute.manager   File "/usr/lib/python2.7/
↪site-packages/retrying.py", line 68, in wrapped_f
2020-04-23 04:41:07.518 19347 ERROR nova.compute.manager     return Retrying(*dargs,
↪ **dkw).call(f, *args, **kw)
2020-04-23 04:41:07.518 19347 ERROR nova.compute.manager   File "/usr/lib/python2.7/
↪site-packages/retrying.py", line 223, in call
2020-04-23 04:41:07.518 19347 ERROR nova.compute.manager     return attempt.
↪get(self._wrap_exception)
2020-04-23 04:41:07.518 19347 ERROR nova.compute.manager   File "/usr/lib/python2.7/
↪site-packages/retrying.py", line 261, in get
2020-04-23 04:41:07.518 19347 ERROR nova.compute.manager     six.reraise(self.
↪value[0], self.value[1], self.value[2])
2020-04-23 04:41:07.518 19347 ERROR nova.compute.manager   File "/usr/lib/python2.7/
↪site-packages/retrying.py", line 217, in call
2020-04-23 04:41:07.518 19347 ERROR nova.compute.manager     attempt =␣
↪Attempt(fn(*args, **kwargs), attempt_number, False)
2020-04-23 04:41:07.518 19347 ERROR nova.compute.manager   File "/usr/lib/python2.7/
↪site-packages/nova/compute/resource_tracker.py", line 1151, in _update_to_
↪placement
2020-04-23 04:41:07.518 19347 ERROR nova.compute.manager     context, compute_node.
↪uuid, name=compute_node.hypervisor_hostname)
2020-04-23 04:41:07.518 19347 ERROR nova.compute.manager   File "/usr/lib/python2.7/
↪site-packages/nova/scheduler/client/report.py", line 858, in get_provider_tree_
↪and_ensure_root
2020-04-23 04:41:07.518 19347 ERROR nova.compute.manager     parent_provider_
↪uuid=parent_provider_uuid)
```

Install Network Service on Controller Node

- https://docs.openstack.org/neutron/train/install/controller-install-rdo.html
- https://docs.openstack.org/neutron/train/install/controller-install-option2-rdo.html

Nodo(s)

CONTROLLER

```
# Conectarnos al servidor de base de datos como usuario root:
mysql -u root --password=openstack

# Crear la base de datos de Neutron:
CREATE DATABASE neutron;

# Brindar el acceso apropiado a la base de datos de Neutron:
GRANT ALL PRIVILEGES ON neutron.* TO 'neutron'@'localhost' IDENTIFIED BY 'openstack';
GRANT ALL PRIVILEGES ON neutron.* TO 'neutron'@'%' IDENTIFIED BY 'openstack';
EXIT;

# Crear un usuario neutron y añadir el rol admin al usuario:
. /home/vagrant/admin-openrc
openstack user create --domain default --password openstack neutron
openstack role add --project service --user neutron admin

# Crear el servicio de Neutron:
openstack service create --name neutron --description "OpenStack Networking" network

# Crear los Compute API Endpoints (public, internal, admin):
openstack endpoint create --region RegionOne network public http://controller:9696
openstack endpoint create --region RegionOne network internal http://controller:9696
openstack endpoint create --region RegionOne network admin http://controller:9696
```

```
# Networking Option 2: Self-service networks

# Instalar los paquetes de Neutron:
yum install -y openstack-neutron openstack-neutron-ml2 openstack-neutron-linuxbridge␣
↪ebtables

# Configurar la base de datos SQL para Neutron:
crudini --set /etc/neutron/neutron.conf database connection mysql+pymysql://
↪neutron:openstack@controller/neutron
# Configurar el acceso RabbitMQ para Neutron:
crudini --set /etc/neutron/neutron.conf DEFAULT transport_url rabbit://
↪openstack:openstack@controller

# Habilitar el plug-in ML2, servicio de router, y overlaping de direcciones IP:
crudini --set /etc/neutron/neutron.conf DEFAULT core_plugin ml2
crudini --set /etc/neutron/neutron.conf DEFAULT service_plugins router
crudini --set /etc/neutron/neutron.conf DEFAULT allow_overlapping_ips true

# Configurar el acceso a Identity service:
```

(continues on next page)

(continued from previous page)

```
crudini --set /etc/neutron/neutron.conf DEFAULT auth_strategy keystone

crudini --set /etc/neutron/neutron.conf keystone_authtoken www_authenticate_uri http:/
↪/controller:5000
crudini --set /etc/neutron/neutron.conf keystone_authtoken auth_url http://
↪controller:5000
crudini --set /etc/neutron/neutron.conf keystone_authtoken memcached_servers␣
↪controller:11211
crudini --set /etc/neutron/neutron.conf keystone_authtoken auth_type password
crudini --set /etc/neutron/neutron.conf keystone_authtoken project_domain_name default
crudini --set /etc/neutron/neutron.conf keystone_authtoken user_domain_name default
crudini --set /etc/neutron/neutron.conf keystone_authtoken project_name service
crudini --set /etc/neutron/neutron.conf keystone_authtoken username neutron
crudini --set /etc/neutron/neutron.conf keystone_authtoken password openstack
# Comentar cualquier otro parámetro en la sección [keystone_authtoken]

# Configurar Networking para que notifique a Compute de cambios en la topología de␣
↪red:
crudini --set /etc/neutron/neutron.conf DEFAULT notify_nova_on_port_status_changes␣
↪true
crudini --set /etc/neutron/neutron.conf DEFAULT notify_nova_on_port_data_changes true

crudini --set /etc/neutron/neutron.conf nova auth_url http://controller:5000
crudini --set /etc/neutron/neutron.conf nova auth_type password
crudini --set /etc/neutron/neutron.conf nova project_domain_name default
crudini --set /etc/neutron/neutron.conf nova user_domain_name default
crudini --set /etc/neutron/neutron.conf nova region_name RegionOne
crudini --set /etc/neutron/neutron.conf nova project_name service
crudini --set /etc/neutron/neutron.conf nova username nova
crudini --set /etc/neutron/neutron.conf nova password openstack

# Configurar el lock path:
crudini --set /etc/neutron/neutron.conf oslo_concurrency lock_path /var/lib/neutron/
↪tmp

# Configurar el ML2 plug-in:

# Enable flat, VLAN and VXLAN Networks
crudini --set /etc/neutron/plugins/ml2/ml2_conf.ini ml2 type_drivers flat,vlan,vxlan

# Enable VXLAN Self-service Networks
crudini --set /etc/neutron/plugins/ml2/ml2_conf.ini ml2 tenant_network_types vxlan

# Enable Linux Bridge and L2Population mechanisms
crudini --set /etc/neutron/plugins/ml2/ml2_conf.ini ml2 mechanism_drivers linuxbridge,
↪l2population

# Enable Port Security Extenstion Driver
crudini --set /etc/neutron/plugins/ml2/ml2_conf.ini ml2 extension_drivers port_
↪security

# Configure provider Virtual Network as flat Network
crudini --set /etc/neutron/plugins/ml2/ml2_conf.ini ml2_type_flat flat_networks␣
↪provider

# Configure VXLAN Network Identifier Range for Self-service Networks
crudini --set /etc/neutron/plugins/ml2/ml2_conf.ini ml2_type_vxlan vni_ranges 1:1000
```

(continues on next page)

(continued from previous page)

```
# Enable ipset to increase efficiency of Security Group Rules
crudini --set /etc/neutron/plugins/ml2/ml2_conf.ini securitygroup enable_ipset true

# Configuar el Linux bridge agent:

# Configure provider Virtual Network mapping to Physical Interface
crudini --set /etc/neutron/plugins/ml2/linuxbridge_agent.ini linux_bridge physical_
↪interface_mappings provider:eth2

# Enable VXLAN for Self-service Networks, configure IP address of the Management␣
↪Interface handling VXLAN traffic
crudini --set /etc/neutron/plugins/ml2/linuxbridge_agent.ini vxlan enable_vxlan true
crudini --set /etc/neutron/plugins/ml2/linuxbridge_agent.ini vxlan local_ip 10.1.1.11
crudini --set /etc/neutron/plugins/ml2/linuxbridge_agent.ini vxlan l2_population true

# Enable security groups and configure the Linux bridge iptables firewall driver
crudini --set /etc/neutron/plugins/ml2/linuxbridge_agent.ini securitygroup enable_
↪security_group true
crudini --set /etc/neutron/plugins/ml2/linuxbridge_agent.ini securitygroup firewall_
↪driver neutron.agent.linux.iptables_firewall.IptablesFirewallDriver

# Configurar el agent L3:
crudini --set /etc/neutron/l3_agent.ini DEFAULT interface_driver linuxbridge

# Configurar el agente DHCP:
crudini --set /etc/neutron/dhcp_agent.ini DEFAULT interface_driver linuxbridge
crudini --set /etc/neutron/dhcp_agent.ini DEFAULT dhcp_driver neutron.agent.linux.
↪dhcp.Dnsmasq
crudini --set /etc/neutron/dhcp_agent.ini DEFAULT enable_isolated_metadata true
```

```
# Configurar el agente metadata:
crudini --set /etc/neutron/metadata_agent.ini DEFAULT nova_metadata_host controller
crudini --set /etc/neutron/metadata_agent.ini DEFAULT metadata_proxy_shared_secret␣
↪openstack

# Configurar el Compute service para que use el Networking service:
# En Stein: url = http://controller:9696 es un parámetro de configuración extra
crudini --set /etc/nova/nova.conf neutron auth_url http://controller:5000
crudini --set /etc/nova/nova.conf neutron auth_type password
crudini --set /etc/nova/nova.conf neutron project_domain_name default
crudini --set /etc/nova/nova.conf neutron user_domain_name default
crudini --set /etc/nova/nova.conf neutron region_name RegionOne
crudini --set /etc/nova/nova.conf neutron project_name service
crudini --set /etc/nova/nova.conf neutron username neutron
crudini --set /etc/nova/nova.conf neutron password openstack
crudini --set /etc/nova/nova.conf neutron service_metadata_proxy true
crudini --set /etc/nova/nova.conf neutron metadata_proxy_shared_secret openstack

# Finalizar la instalación:

# Si no existe, crear un link simbólico /etc/neutron/plugin.ini apuntando al archivo /
↪etc/neutron/plugins/ml2/ml2_conf.ini
ln -s /etc/neutron/plugins/ml2/ml2_conf.ini /etc/neutron/plugin.ini

# Poblar la base de datos:
```

(continues on next page)

(continued from previous page)

```
su -s /bin/sh -c "neutron-db-manage --config-file /etc/neutron/neutron.conf --config-
↪file /etc/neutron/plugins/ml2/ml2_conf.ini upgrade head" neutron

# Reiniciar el servicio de Compute API:
systemctl restart openstack-nova-api.service

# Habilitat e iniciar los servicios de Networking:
systemctl enable neutron-server.service neutron-linuxbridge-agent.service neutron-
↪dhcp-agent.service neutron-metadata-agent.service
systemctl start neutron-server.service neutron-linuxbridge-agent.service neutron-dhcp-
↪agent.service neutron-metadata-agent.service

# Para la opción de Networking 2, habilitamos e iniciamos también el servicio de L3:
systemctl enable neutron-l3-agent.service
systemctl start neutron-l3-agent.service
```

Install Neutron on Compute Node

- https://docs.openstack.org/neutron/train/install/compute-install-rdo.html
- https://docs.openstack.org/neutron/train/install/compute-install-option2-rdo.html
- https://docs.openstack.org/neutron/train/install/verify-option2.html

---

Nodo(s)

COMPUTE

```
# Instalar los componentes:
yum install -y openstack-neutron-linuxbridge ebtables ipset

# Configuar los componentes comúnes:

# En la sección [database], comentar cualquier opción de conexión porque los compute␣
↪nodes no acceden directamente a la base de datos.

# Configurar acceso a RabbitMQ:
crudini --set /etc/neutron/neutron.conf DEFAULT transport_url rabbit://
↪openstack:openstack@controller

# Configurar el acceso a Identity Service:
crudini --set /etc/neutron/neutron.conf DEFAULT auth_strategy keystone
crudini --set /etc/neutron/neutron.conf keystone_authtoken www_authenticate_uri http:/
↪/controller:5000
crudini --set /etc/neutron/neutron.conf keystone_authtoken auth_url http://
↪controller:5000
crudini --set /etc/neutron/neutron.conf keystone_authtoken memcached_servers␣
↪controller:11211
crudini --set /etc/neutron/neutron.conf keystone_authtoken auth_type password
crudini --set /etc/neutron/neutron.conf keystone_authtoken project_domain_name default
crudini --set /etc/neutron/neutron.conf keystone_authtoken user_domain_name default
crudini --set /etc/neutron/neutron.conf keystone_authtoken project_name service
crudini --set /etc/neutron/neutron.conf keystone_authtoken username neutron
crudini --set /etc/neutron/neutron.conf keystone_authtoken password openstack
```

(continues on next page)

(continued from previous page)

```
# Configurar el lock path:
crudini --set /etc/neutron/neutron.conf oslo_concurrency lock_path /var/lib/neutron/
↪tmp

# Configuar el Linux bridge agent:

# Configure provider Virtual Network mapping to Physical Interface
crudini --set /etc/neutron/plugins/ml2/linuxbridge_agent.ini linux_bridge physical_
↪interface_mappings provider:eth2

# Enable VXLAN for Self-service Networks, configure IP address of the Management␣
↪Interface handling VXLAN traffic
crudini --set /etc/neutron/plugins/ml2/linuxbridge_agent.ini vxlan enable_vxlan true
# Reemplazar IP por la del compute node que se está configurando:
crudini --set /etc/neutron/plugins/ml2/linuxbridge_agent.ini vxlan local_ip 10.1.1.31
crudini --set /etc/neutron/plugins/ml2/linuxbridge_agent.ini vxlan l2_population true

# Enable security groups and configure the Linux bridge iptables firewall driver
crudini --set /etc/neutron/plugins/ml2/linuxbridge_agent.ini securitygroup enable_
↪security_group true
crudini --set /etc/neutron/plugins/ml2/linuxbridge_agent.ini securitygroup firewall_
↪driver neutron.agent.linux.iptables_firewall.IptablesFirewallDriver

# Configurar el Compute service para que use el Networking service:
# En Stein: url = http://controller:9696 es un parámetro de configuración extra
crudini --set /etc/nova/nova.conf neutron auth_url http://controller:5000
crudini --set /etc/nova/nova.conf neutron auth_type password
crudini --set /etc/nova/nova.conf neutron project_domain_name default
crudini --set /etc/nova/nova.conf neutron user_domain_name default
crudini --set /etc/nova/nova.conf neutron region_name RegionOne
crudini --set /etc/nova/nova.conf neutron project_name service
crudini --set /etc/nova/nova.conf neutron username neutron
crudini --set /etc/nova/nova.conf neutron password openstack

# Reiniciar el Compute service:
systemctl restart openstack-nova-compute.service

# Habilitar e iniciar el Linux bridge agent:
systemctl enable neutron-linuxbridge-agent.service
systemctl start neutron-linuxbridge-agent.service
```

Nodo(s)

CONTROLLER

```
# Verificar la operación de Neutron:
. /home/vagrant/admin-openrc
openstack extension list --network
openstack network agent list
```

Block Storage service - Controller node

- https://docs.openstack.org/cinder/train/install/cinder-controller-install-rdo.html

Nodo(s)

CONTROLLER

```
# Conectarnos al servidor de base de datos como usuario root:
mysql -u root --password=openstack

# Crear la base de datos de Cinder:
CREATE DATABASE cinder;
# Brindar el acceso apropiado a la base de datos de Cinder:
GRANT ALL PRIVILEGES ON cinder.* TO 'cinder'@'localhost' IDENTIFIED BY 'openstack';
GRANT ALL PRIVILEGES ON cinder.* TO 'cinder'@'%' IDENTIFIED BY 'openstack';
EXIT;

# Crear un usuario cinder y añadir el rol admin al usuario cinder:
. /home/vagrant/admin-openrc
openstack user create --domain default --password openstack cinder
openstack role add --project service --user cinder admin

# Crear los servicios cinderv2 y cinderv3:
openstack service create --name cinderv2 --description "OpenStack Block Storage"␣
↪volumev2
openstack service create --name cinderv3 --description "OpenStack Block Storage"␣
↪volumev3

# Crear los Cinder Service Endpoints (public, internal, admin):
openstack endpoint create --region RegionOne volumev2 public http://controller:8776/
↪v2/%\(project_id\)s
openstack endpoint create --region RegionOne volumev2 internal http://controller:8776/
↪v2/%\(project_id\)s
openstack endpoint create --region RegionOne volumev2 admin http://controller:8776/v2/
↪%\(project_id\)s
openstack endpoint create --region RegionOne volumev3 public http://controller:8776/
↪v3/%\(project_id\)s
openstack endpoint create --region RegionOne volumev3 internal http://controller:8776/
↪v3/%\(project_id\)s
openstack endpoint create --region RegionOne volumev3 admin http://controller:8776/v3/
↪%\(project_id\)s

# Instalar el paquete de Cinder:
yum install -y openstack-cinder

# Configurar la base de datos SQL para Neutron:
crudini --set /etc/cinder/cinder.conf database connection mysql+pymysql://
↪cinder:openstack@controller/cinder
# Configurar el acceso RabbitMQ para Neutron:
crudini --set /etc/cinder/cinder.conf DEFAULT transport_url rabbit://
↪openstack:openstack@controller

# Configurar el acceso a Identity service:
crudini --set /etc/cinder/cinder.conf DEFAULT auth_strategy keystone
crudini --set /etc/cinder/cinder.conf keystone_authtoken www_authenticate_uri http://
↪controller:5000
crudini --set /etc/cinder/cinder.conf keystone_authtoken auth_url http://
↪controller:5000
crudini --set /etc/cinder/cinder.conf keystone_authtoken memcached_servers␣
↪controller:11211
```

(continues on next page)

(continued from previous page)

```
crudini --set /etc/cinder/cinder.conf keystone_authtoken auth_type password
crudini --set /etc/cinder/cinder.conf keystone_authtoken project_domain_name default
crudini --set /etc/cinder/cinder.conf keystone_authtoken user_domain_name default
crudini --set /etc/cinder/cinder.conf keystone_authtoken project_name service
crudini --set /etc/cinder/cinder.conf keystone_authtoken username cinder
crudini --set /etc/cinder/cinder.conf keystone_authtoken password openstack

# Configurar la dirección de la interfaz de administración del controller node:
crudini --set /etc/cinder/cinder.conf DEFAULT my_ip 10.1.1.11

# Configurar el lock path:
crudini --set /etc/cinder/cinder.conf oslo_concurrency lock_path /var/lib/cinder/tmp

# Poblar la base de datos de block storage:
su -s /bin/sh -c "cinder-manage db sync" cinder

# Configurar el Compute service para que use el Block Storage service:
crudini --set /etc/nova/nova.conf cinder os_region_name RegionOne

# Reiniciar el servicio de Compute API:
systemctl restart openstack-nova-api.service

# Habilitar e iniciar los servicios de Block Storage:
systemctl enable openstack-cinder-api.service openstack-cinder-scheduler.service
systemctl start openstack-cinder-api.service openstack-cinder-scheduler.service
```

Block Storage service - Storage node

- https://docs.openstack.org/cinder/train/install/cinder-storage-install-rdo.html

---

Nodo(s)

BLOCK

Important: Agregar un nuevo disco por VirtualBox a la máquina virtual (sdb)

```
# Instalar los paquetes LVM (lvm2 y device-mapper-persistent-data vienen instalados␣
↪por defecto):
yum install -y lvm2 device-mapper-persistent-data

# Habilitar e iniciar el servicio de metada de LVM:
systemctl enable lvm2-lvmetad.service
systemctl start lvm2-lvmetad.service

# Verificar que exista el disco sdb:
fdisk -l

# Crear el volumen físico LVM /dev/sdb:
pvcreate /dev/sdb

# Crear el LVM volume group:
vgcreate cinder-volumes /dev/sdb
```

(continues on next page)

(continued from previous page)

```
# Editar el archivo de configuración LVM /etc/lvm/lvm.conf, para que incluya la␣
↪siguiente línea:
# Si usamos LVM en el disco de sistema operativo, añadimos el dispositivo al filtro␣
↪(p. ej. /dev/sda)
devices {
...
filter = [ "a/sda/", "a/sdb/", "r/.*/"]

# Instalar los paquetes de Cinder (además de crudini para la configuración de␣
↪archivos):
yum install -y openstack-cinder targetcli python-keystone crudini

# Configurar la base de datos SQL para Neutron:
crudini --set /etc/cinder/cinder.conf database connection mysql+pymysql://
↪cinder:openstack@controller/cinder
# Configurar el acceso RabbitMQ para Neutron:
crudini --set /etc/cinder/cinder.conf DEFAULT transport_url rabbit://
↪openstack:openstack@controller

# Configurar el acceso a Identity Service:
crudini --set /etc/cinder/cinder.conf DEFAULT auth_strategy keystone
crudini --set /etc/cinder/cinder.conf keystone_authtoken www_authenticate_uri http://
↪controller:5000
crudini --set /etc/cinder/cinder.conf keystone_authtoken auth_url http://
↪controller:5000
crudini --set /etc/cinder/cinder.conf keystone_authtoken memcached_servers␣
↪controller:11211
crudini --set /etc/cinder/cinder.conf keystone_authtoken auth_type password
crudini --set /etc/cinder/cinder.conf keystone_authtoken project_domain_name default
crudini --set /etc/cinder/cinder.conf keystone_authtoken user_domain_name default
crudini --set /etc/cinder/cinder.conf keystone_authtoken project_name service
crudini --set /etc/cinder/cinder.conf keystone_authtoken username cinder
crudini --set /etc/cinder/cinder.conf keystone_authtoken password openstack
# Comentar cualquier otro parámetro en la sección [keystone_authtoken]

crudini --set /etc/cinder/cinder.conf DEFAULT my_ip 10.1.1.41

# Configurar LVM Backend:
crudini --set /etc/cinder/cinder.conf lvm volume_driver cinder.volume.drivers.lvm.
↪LVMVolumeDriver
crudini --set /etc/cinder/cinder.conf lvm volume_group cinder-volumes
crudini --set /etc/cinder/cinder.conf lvm target_protocol iscsi
crudini --set /etc/cinder/cinder.conf lvm target_helper lioadm

# Habilitar el LVM Backend:
crudini --set /etc/cinder/cinder.conf DEFAULT enabled_backends lvm

# Configurar la ubicación del Image service:
crudini --set /etc/cinder/cinder.conf DEFAULT glance_api_servers http://
↪controller:9292

# Configurar el lock path:
crudini --set /etc/cinder/cinder.conf oslo_concurrency lock_path /var/lib/cinder/tmp

# Habilitar e inicar servicios de volumen Block storage:
systemctl enable openstack-cinder-volume.service target.service
```

(continues on next page)

(continued from previous page)

```
systemctl start openstack-cinder-volume.service target.service
```

Verify Cinder operation

- https://docs.openstack.org/cinder/train/install/cinder-verify.html

Nodo(s)

CONTROLLER

```
# Verificar la operación de Cinder:
. /home/vagrant/admin-openrc

# Listar los componentes del servicio:
openstack volume service list

# Crear un volumen:
. /home/vagrant/demo-openrc
openstack volume create --size 1 test-volume

# Listar volúmenes:
openstack volume list
```

Nodo(s)

BLOCK

```
lvdisplay
```

Install and Configure Horizon Packages

- https://docs.openstack.org/horizon/train/install/install-rdo.html

Nodo(s)

CONTROLLER

```
# Instalar paquetes:
yum install -y openstack-dashboard
```

Editar el archivo `/etc/openstack-dashboard/local_settings` para que incluya:

```
OPENSTACK_HOST = "controller"

ALLOWED_HOSTS = ['*']

SESSION_ENGINE = 'django.contrib.sessions.backends.cache'

CACHES = {
```

(continues on next page)

(continued from previous page)

```
    'default': {
        'BACKEND': 'django.core.cache.backends.memcached.MemcachedCache',
        'LOCATION': 'controller:11211',
    }
}

OPENSTACK_KEYSTONE_URL = "http://%s:5000/v3" % OPENSTACK_HOST

OPENSTACK_KEYSTONE_MULTIDOMAIN_SUPPORT = True

OPENSTACK_API_VERSIONS = {
    "identity": 3,
    "image": 2,
    "volume": 3,
}

OPENSTACK_KEYSTONE_DEFAULT_DOMAIN = "Default"

OPENSTACK_KEYSTONE_DEFAULT_ROLE = "user"

WEBROOT = '/dashboard/'
```

Error: ERROR en OpenStack Dashboard: En el navegador sale el mensaje: 'Not Found. The requested URL /auth/login/ was not found on this server'

SOLUCIÓN:

- https://louky0714.tistory.com/entry/Openstack-Train-Error-The-requested-URL-authlogin-was-not-found-on-this-server
- https://www.youtube.com/watch?v=_sHkUn4fFe8

Debemos agregar una línea extra al archivo `/etc/openstack-dashboard/local_settings`:

```
WEBROOT = '/dashboard/'
```

- Editar el archivo `/etc/httpd/conf.d/openstack-dashboard.conf` para incluir la siguiente línea, si no se encuentra:

```
WSGIApplicationGroup %{GLOBAL}
```

- Reiniciar la configuración del web server:

```
systemctl restart httpd.service memcached.service
```

- Ingresar desde un navegador al dashboard: http://10.1.1.11/dashboard/

Mostrar logs de servicios para troubleshooting de errores

- Controller node (Placemente, Neutron):

```
tail -f /var/log/placement/placement-api.log | grep -i error &
tail -f /var/log/neutron/* | grep -i error &
```

- Compute node (Nova):

```
tail -f /var/log/nova/* | grep -i error &
```

- Block node (Cinder):

```
tail -f /var/log/cinder/* | grep -i error &
```

Primeras pasos con OpenStack

```
source /home/vagrant/admin-openrc
openstack flavor create --id 1 --ram 512 --disk 1 --public m1.tiny
openstack network create  --share --external --provider-physical-network provider --
↪provider-network-type flat provider
openstack subnet create --network provider --dns-nameserver 8.8.4.4 --subnet-range␣
↪203.0.113.0/24 provider

source /home/vagrant/demo-openrc
openstack network create private
openstack subnet create private --subnet-range 192.168.100.0/24 --dns-nameserver 8.8.
↪8.8 --network private
openstack router create demo-nsrouter
openstack router add subnet demo-nsrouter private
openstack router set demo-nsrouter --external-gateway provider
openstack network list
openstack server create --image cirros4.0 --flavor 1 --nic net-id=6a5f05fb-048d-4b5f-
↪8409-ff115d32ac84 inst1
```

15.3 Configuración inicial de OpenStack

Table of Contents

- *Asignar una Floating IP a una instancia*
- *Crear un key pair*
- *Crear un security group*
- *Crear una instancia (2)*
- *Crear snapshot de una imagen*
- *Crear un volumen*
- *Conectar y montar un volumen a una instancia*
- *Crear backup de un volumen*
- *Crear snapshot de un volumen*
- *Crear un contenedor*
- *Crear un objeto*
- *Acceder a un objeto*
- *Copiar llave pública al usuario* `admin`
- *Crear Host Aggregates*

15.3.1 Usar credenciales de usuario

En el CLI, usar las credenciales del usuario `admin`

```
'#' cd /root/
'#' source keystonerc_admin
```

15.3.2 Crear un nuevo proyecto

```
'#' openstack project create --description "for testing purposes" testproject

+-------------+----------------------------------+
| Field       | Value                            |
+-------------+----------------------------------+
| description | for testing purposes             |
| domain_id   | default                          |
| enabled     | True                             |
| id          | 99d8a6cd24734f2aa3fe70140fbdbd64 |
| is_domain   | False                            |
| name        | testproject                      |
| parent_id   | default                          |
| tags        | []                               |
+-------------+----------------------------------+
```

15.3.3 Creación de un usuario asignado a un proyecto

1. Crear un usuario y añadirlo a un proyecto al momento de ser creado:

```
'#' openstack user create --project testproject --password-prompt testuser1

User Password:
Repeat User Password:
+---------------------+----------------------------------+
| Field               | Value                            |
+---------------------+----------------------------------+
| default_project_id  | 99d8a6cd24734f2aa3fe70140fbdbd64 |
| domain_id           | default                          |
| enabled             | True                             |
| id                  | 6643474fffb548b4bd4fb3d6a09d9ecd |
| name                | testuser1                        |
| options             | {}                               |
| password_expires_at | None                             |
+---------------------+----------------------------------+
```

2. Asignar los roles _member_ y admin al usuario creado:

```
'#' openstack role add --project testproject --user testuser1 _member_

'#' openstack role assignment list --project testproject --user testuser1

+----------------------------------+----------------------------------+-------+-------
↪---------------------------+--------+-----------+
| Role                             | User                             | Group |␣
↪Project                          | Domain | Inherited |
+----------------------------------+----------------------------------+-------+-------
↪---------------------------+--------+-----------+
| 75dbe014bfa54197890b46a034f4661e | 6643474fffb548b4bd4fb3d6a09d9ecd |       |␣
↪99d8a6cd24734f2aa3fe70140fbdbd64 |        | False     |
+----------------------------------+----------------------------------+-------+-------
↪---------------------------+--------+-----------+
```

Important: Si queremos que nuestro usuario tenga permisos de administrador, ejecutar la siguiente línea:

```
'#' openstack role add --project testproject --user testuser1 admin
```

15.3.4 Creación de credenciales CLI para un usuario

```
'#' cp keystonerc_admin keystonerc_testuser1

'#' cat <<- EOF > keystonerc_testuser1
unset OS_SERVICE_TOKEN
    export OS_USERNAME=testuser1
    export OS_PASSWORD=testuser1
    export OS_REGION_NAME=RegionOne
    export OS_AUTH_URL=http://192.168.1.100:5000/v3
    export PS1='[\u@\h \W(testuser1)]\$ '

export OS_PROJECT_NAME=testproject
export OS_USER_DOMAIN_NAME=Default
export OS_PROJECT_DOMAIN_NAME=Default
```

(continues on next page)

(continued from previous page)

```
export OS_IDENTITY_API_VERSION=3
EOF
```

15.3.5 Creación de un flavor

Nuevo flavor llamado `m1.tiniest` con id `10`, RAM de 128 MB y 1GB de almacenamiento:

```
'#' openstack flavor create --id 10 --ram 128 --disk 1 --public m1.tiniest

+----------------------------+------------+
| Field                      | Value      |
+----------------------------+------------+
| OS-FLV-DISABLED:disabled   | False      |
| OS-FLV-EXT-DATA:ephemeral  | 0          |
| disk                       | 1          |
| id                         | 10         |
| name                       | m1.tiniest |
| os-flavor-access:is_public | True       |
| properties                 |            |
| ram                        | 128        |
| rxtx_factor                | 1.0        |
| swap                       |            |
| vcpus                      | 1          |
+----------------------------+------------+

'#' openstack flavor list

+----+------------+-------+------+-----------+-------+-----------+
| ID | Name       |   RAM | Disk | Ephemeral | VCPUs | Is Public |
+----+------------+-------+------+-----------+-------+-----------+
| 1  | m1.tiny    |   512 |    1 |         0 |     1 | True      |
| 10 | m1.tiniest |   128 |    1 |         0 |     1 | True      |
| 2  | m1.small   |  2048 |   20 |         0 |     1 | True      |
| 3  | m1.medium  |  4096 |   40 |         0 |     2 | True      |
| 4  | m1.large   |  8192 |   80 |         0 |     4 | True      |
| 5  | m1.xlarge  | 16384 |  160 |         0 |     8 | True      |
+----+------------+-------+------+-----------+-------+-----------+
```

15.3.6 Crear una imagen

1. Descargar la imagen:

```
'#' mkdir /root/images
'#' curl -o /root/images/cirros-0.4.0-x86_64-disk.img -L http://download.cirros-cloud.
↪net/0.4.0/cirros-0.4.0-x86_64-disk.img
```

Note: Usar la opción `-L` de `curl` para rehacer el pedido en la ubicación indicada.

- Referencia 1: Comando curl no descarga el archivo
- Referencia 2: Descargar con curl usando -L para seguir los redirects

Note: Con `wget`:

```
'#' yum install -y wget
'#' wget -P /root/images http://download.cirros-cloud.net/0.4.0/cirros-0.4.0-x86_64-
↪disk.img
```

2. Crear una imagen pública con los requerimientos mínimos de almacenamiento y memoria RAM para el uso de esa imagen:

```
'#' openstack image create --min-disk 1 --min-ram 128 --public --disk-format qcow2 --
↪file /root/images/cirros-0.4.0-x86_64-disk.img cirros

+------------------+------------------------------------------------------+
| Field            | Value                                                |
+------------------+------------------------------------------------------+
| checksum         | 443b7623e27ecf03dc9e01ee93f67afe                     |
| container_format | bare                                                 |
| created_at       | 2020-02-11T02:53:24Z                                 |
| disk_format      | qcow2                                                |
| file             | /v2/images/56dd3671-de42-40db-9637-7c5bef599d11/file |
| id               | 56dd3671-de42-40db-9637-7c5bef599d11                 |
| min_disk         | 1                                                    |
| min_ram          | 128                                                  |
| name             | cirros                                               |
| owner            | 99d8a6cd24734f2aa3fe70140fbdbd64                     |
| protected        | False                                                |
| schema           | /v2/schemas/image                                    |
| size             | 12716032                                             |
| status           | active                                               |
| tags             |                                                      |
| updated_at       | 2020-02-11T02:53:24Z                                 |
| virtual_size     | None                                                 |
| visibility       | public                                               |
+------------------+------------------------------------------------------+

'#' openstack image list

+--------------------------------------+--------+--------+
| ID                                   | Name   | Status |
+--------------------------------------+--------+--------+
| 2995472e-5c8b-4828-af2c-0104b24db391 | cirros | active |
+--------------------------------------+--------+--------+
```

15.3.7 Ver configuración de red

Comprobar que tenemos los agentes de red necesarios para la gestión de redes con OpenStack:

```
'#' openstack network agent list

+--------------------------------------+--------------------+-------------------------
↪----+-------------------+-------+-------+---------------------------+
| ID                                   | Agent Type         | Host                    ␣
↪    | Availability Zone | Alive | State | Binary                    |
+--------------------------------------+--------------------+-------------------------
↪----+-------------------+-------+-------+---------------------------+
```

(continues on next page)

(continued from previous page)

```
| 1755e383-779d-430e-836c-d5ab6300247e | Open vSwitch agent | controllernode1.
↪localdomain | None              | :-)   | UP    | neutron-openvswitch-agent |
| 72d25877-3746-4ebd-bda7-586bd5ee2ddf | Open vSwitch agent | computenode1.
↪localdomain    | None              | :-)   | UP    | neutron-openvswitch-agent |
| ffe83ae4-da2a-411a-a535-8f1fdbf06e60 | Open vSwitch agent | computenode2.
↪localdomain    | None              | :-)   | UP    | neutron-openvswitch-agent |
| 4aff7724-1a9e-42b7-aad3-142fd5c1d736 | DHCP agent         | controllernode1.
↪localdomain | nova              | :-)   | UP    | neutron-dhcp-agent        |
| 8a45485e-f1d8-46ff-a0e4-440f98f377fc | Metering agent     | controllernode1.
↪localdomain | None              | :-)   | UP    | neutron-metering-agent    |
| f533494b-9071-4159-ae80-12ae96534c77 | L3 agent           | controllernode1.
↪localdomain | nova              | :-)   | UP    | neutron-l3-agent          |
| fc03ca4e-a5e4-469b-bbb8-9253248e40a6 | Metadata agent     | controllernode1.
↪localdomain | None              | :-)   | UP    | neutron-metadata-agent    |
+--------------------------------------+--------------------+-------------------------
↪----+-------------------+-------+-------+---------------------------+
```

Note: Por cada Nodo contamos con un agente Open vSwitch (controller node, compute node 1 y compute node 2)

Podemos verificar el estado de cada servicio usando el binario de la tabla así como del proceso `neutron-server` en sí:

```
'#' systemctl status neutron-server neutron-openvswitch-agent neutron-dhcp-agent␣
↪neutron-metering-agent neutron-l3-agent neutron-metadata-agent
```

- Ver los OVS bridges que tenemos creados y los puertos conectados:

```
'#' ovs-vsctl show

496c7134-8b33-4aa9-b752-ab503fccd5d6
    Manager "ptcp:6640:127.0.0.1"
        is_connected: true
    Bridge br-int
        Controller "tcp:127.0.0.1:6633"
            is_connected: true
        fail_mode: secure
        Port br-int
            Interface br-int
                type: internal
        Port patch-tun
            Interface patch-tun
                type: patch
                options: {peer=patch-int}
        Port int-br-ex
            Interface int-br-ex
                type: patch
                options: {peer=phy-br-ex}
    Bridge br-tun
        Controller "tcp:127.0.0.1:6633"
            is_connected: true
        fail_mode: secure
        Port br-tun
            Interface br-tun
                type: internal
        Port "vxlan-0a0a0a66"
```

(continues on next page)

(continued from previous page)

```
            Interface "vxlan-0a0a0a66"
                type: vxlan
                options: {df_default="true", egress_pkt_mark="0", in_key=flow, local_
↪ip="192.168.1.100", out_key=flow, remote_ip="10.10.10.102"}
        Port patch-int
            Interface patch-int
                type: patch
                options: {peer=patch-tun}
        Port "vxlan-0a0a0a65"
            Interface "vxlan-0a0a0a65"
                type: vxlan
                options: {df_default="true", egress_pkt_mark="0", in_key=flow, local_
↪ip="192.168.1.100", out_key=flow, remote_ip="10.10.10.101"}
    Bridge br-ex
        Controller "tcp:127.0.0.1:6633"
            is_connected: true
        fail_mode: secure
        Port br-ex
            Interface br-ex
                type: internal
        Port phy-br-ex
            Interface phy-br-ex
                type: patch
                options: {peer=int-br-ex}
        Port "enp0s3"
            Interface "enp0s3"
    ovs_version: "2.11.0"
```

Note: Los bridges creados por defecto al instalar OpenStack son `br-int` (integration bridge), `br-tun` (tunnel bridge), y `br-ex` (external bridge).

15.3.8 Crear una red y una subred internas

Link: Create and manage networks - Openstack Docs

1. Crear una red:

```
'#' source keystonerc_testuser1

'#' openstack network create intnet

+---------------------------+--------------------------------------+
| Field                     | Value                                |
+---------------------------+--------------------------------------+
| admin_state_up            | UP                                   |
| availability_zone_hints   |                                      |
| availability_zones        |                                      |
| created_at                | 2020-02-10T23:19:50Z                 |
| description               |                                      |
| dns_domain                | None                                 |
| id                        | 2cf9c274-8592-476b-bde5-41e930e01577 |
| ipv4_address_scope        | None                                 |
| ipv6_address_scope        | None                                 |
```

(continues on next page)

(continued from previous page)

```
| is_default                | False                                |
| is_vlan_transparent       | None                                 |
| mtu                       | 1450                                 |
| name                      | intnet                               |
| port_security_enabled     | True                                 |
| project_id                | 99d8a6cd24734f2aa3fe70140fbdbd64     |
| provider:network_type     | vxlan                                |
| provider:physical_network | None                                 |
| provider:segmentation_id  | 84                                   |
| qos_policy_id             | None                                 |
| revision_number           | 2                                    |
| router:external           | Internal                             |
| segments                  | None                                 |
| shared                    | False                                |
| status                    | ACTIVE                               |
| subnets                   |                                      |
| tags                      |                                      |
| updated_at                | 2020-02-10T23:19:50Z                 |
+---------------------------+--------------------------------------+
```

La red creada usará algunos parámetros por defecto:

- `provider:network_type`: `vxlan` - El tipo de red por defecto es VXLAN
- `provider:physical_network`: `None` - La red virtual no se implementará sobre una red física
- `router:external`: `Internal` - Configurar esta red como interna

2. Crear una subred asociada a la red creada:

```
'#' openstack subnet create subnet1 --subnet-range 10.5.5.0/24 --dns-nameserver 8.8.8.
↪8 --network intnet

+-------------------+--------------------------------------+
| Field             | Value                                |
+-------------------+--------------------------------------+
| allocation_pools  | 10.5.5.2-10.5.5.254                  |
| cidr              | 10.5.5.0/24                          |
| created_at        | 2020-02-10T23:44:17Z                 |
| description       |                                      |
| dns_nameservers   | 8.8.8.8                              |
| enable_dhcp       | True                                 |
| gateway_ip        | 10.5.5.1                             |
| host_routes       |                                      |
| id                | 8168b012-2c3c-4114-8145-963dd6646793 |
| ip_version        | 4                                    |
| ipv6_address_mode | None                                 |
| ipv6_ra_mode      | None                                 |
| name              | subnet1                              |
| network_id        | 2cf9c274-8592-476b-bde5-41e930e01577 |
| prefix_length     | None                                 |
| project_id        | 99d8a6cd24734f2aa3fe70140fbdbd64     |
| revision_number   | 0                                    |
| segment_id        | None                                 |
| service_types     |                                      |
| subnetpool_id     | None                                 |
| tags              |                                      |
| updated_at        | 2020-02-10T23:44:17Z                 |
```

(continues on next page)

(continued from previous page)

```
+------------------+--------------------------------------+
```

- La subred creada tiene el ID de la red `intnet` asociada a ella.
- Cuenta con un pool de direcciones reservadas para asignar: `10.5.5.2-10.5.5.254`
- Se estableció como servidor DNS la IP `8.8.8.8`

Link del comando: subnet - Openstack Docs

Además se han configurado unas opciones en la subred por defecto:

- Se establece como su gateway IP a la primera dirección IP del rango CIDR: `10.5.5.1`
- `dhcp` está habilitado, por tanto se creó un namespace DHCP con una interfaz `tap` a la que se le agregó una IP dentro de la subred (Generalmente la IP `.2`). Además, este namespace tiene su interfaz conectada al bridge `br-int`:

```
'#' ip netns

qdhcp-2cf9c274-8592-476b-bde5-41e930e01577 (id: 0)

'#' ip netns exec qdhcp-2cf9c274-8592-476b-bde5-41e930e01577 ip addr

1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default␣
↪qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
        valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host
        valid_lft forever preferred_lft forever
13: tapa2009c32-11: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1450 qdisc noqueue state␣
↪UNKNOWN group default qlen 1000
    link/ether fa:16:3e:51:a7:89 brd ff:ff:ff:ff:ff:ff
    inet 10.5.5.2/24 brd 10.5.5.255 scope global tapa2009c32-11
        valid_lft forever preferred_lft forever
    inet6 fe80::f816:3eff:fe51:a789/64 scope link
        valid_lft forever preferred_lft forever

'#' ovs-vsctl show

496c7134-8b33-4aa9-b752-ab503fccd5d6
    Manager "ptcp:6640:127.0.0.1"
        is_connected: true
    Bridge br-int
        Controller "tcp:127.0.0.1:6633"
            is_connected: true
        fail_mode: secure
        Port "tapa2009c32-11"
            tag: 3
            Interface "tapa2009c32-11"
                type: internal
        Port br-int
            Interface br-int
                type: internal
        Port patch-tun
            Interface patch-tun
                type: patch
                options: {peer=patch-int}
```

(continues on next page)

(continued from previous page)

```
        Port int-br-ex
            Interface int-br-ex
                type: patch
                options: {peer=phy-br-ex}
    ...
```

Important: Los namespaces proveen aislamiento de tráfico en Neutron. Para cada nuevo servidor DHCP, se crea un nuevo namespace. Así podemos diferenciar el tráfico entre distintos proyectos.

Si se agregaran más subredes con `dhcp` habilitado dentro de la misma red, se agregaría una IP extra a la misma interfaz `tap` del mismo namespace DHCP.

```
'#' ip netns exec qdhcp-2cf9c274-8592-476b-bde5-41e930e01577 ip addr

1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default␣
↪qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
        valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host
        valid_lft forever preferred_lft forever
12: tapfbe9d2c4-80: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1450 qdisc noqueue state␣
↪UNKNOWN group default qlen 1000
    link/ether fa:16:3e:05:dd:2e brd ff:ff:ff:ff:ff:ff
    inet 10.5.7.2/24 brd 10.5.7.255 scope global tapfbe9d2c4-80
        valid_lft forever preferred_lft forever
    inet 10.5.8.2/24 brd 10.5.8.255 scope global tapfbe9d2c4-80
        valid_lft forever preferred_lft forever
    inet6 fe80::f816:3eff:fe05:dd2e/64 scope link
        valid_lft forever preferred_lft forever
```

15.3.9 Crear un router

1. Crear un router sin interfaces conectadas a él:

```
'#' openstack router create R1

+-------------------------+--------------------------------------+
| Field                   | Value                                |
+-------------------------+--------------------------------------+
| admin_state_up          | UP                                   |
| availability_zone_hints |                                      |
| availability_zones      |                                      |
| created_at              | 2020-02-11T00:08:40Z                 |
| description             |                                      |
| distributed             | False                                |
| external_gateway_info   | None                                 |
| flavor_id               | None                                 |
| ha                      | False                                |
| id                      | d4cb763e-8578-484d-be6a-6d7da165e161 |
| name                    | R1                                   |
| project_id              | 99d8a6cd24734f2aa3fe70140fbdbd64     |
| revision_number         | 1                                    |
```

(continues on next page)

(continued from previous page)

```
| routes                  |                                       |
| status                  | ACTIVE                                |
| tags                    |                                       |
| updated_at              | 2020-02-11T00:08:40Z                  |
+-------------------------+---------------------------------------+
```

2. Conectar el router con una subred:

```
'#' openstack router add subnet R1 subnet1
```

- Comprobar cambios de configuración:

```
'#' ovs-vsctl show

496c7134-8b33-4aa9-b752-ab503fccd5d6
    Manager "ptcp:6640:127.0.0.1"
        is_connected: true
    Bridge br-int
        Controller "tcp:127.0.0.1:6633"
            is_connected: true
        fail_mode: secure
        Port "tapa2009c32-11"
            tag: 3
            Interface "tapa2009c32-11"
                type: internal
        Port br-int
            Interface br-int
                type: internal
        Port patch-tun
            Interface patch-tun
                type: patch
                options: {peer=patch-int}
        Port "qr-4c7615a5-dd"
            tag: 3
            Interface "qr-4c7615a5-dd"
                type: internal
        Port int-br-ex
            Interface int-br-ex
                type: patch
                options: {peer=phy-br-ex}

'#' ip netns

qrouter-d4cb763e-8578-484d-be6a-6d7da165e161 (id: 1)
qdhcp-2cf9c274-8592-476b-bde5-41e930e01577 (id: 0)

'#' openstack router show R1

+-------------------------+-------------------------------------------------------------
↪---------------------------------------------------------------------+
| Field                   | Value                                                       ␣
↪                                                                     |
+-------------------------+-------------------------------------------------------------
↪---------------------------------------------------------------------+
| admin_state_up          | UP                                                          ␣
↪                                                                     |
| availability_zone_hints |                                                             ␣
↪                                                                     |
```

(continues on next page)

(continued from previous page)

```
| availability_zones      | nova                                                       ␣
↪                                                                           |
| created_at              | 2020-02-11T00:08:40Z                                       ␣
↪                                                                           |
| description             |                                                            ␣
↪                                                                           |
| distributed             | False                                                      ␣
↪                                                                           |
| external_gateway_info   | None                                                       ␣
↪                                                                           |
| flavor_id               | None                                                       ␣
↪                                                                           |
| ha                      | False                                                      ␣
↪                                                                           |
| id                      | d4cb763e-8578-484d-be6a-6d7da165e161                       ␣
↪                                                                           |
| interfaces_info         | [{"subnet_id": "8168b012-2c3c-4114-8145-963dd6646793",
↪"ip_address": "10.5.5.1", "port_id": "4c7615a5-dd19-43d8-833f-37e3505b8175"}] |
| name                    | R1                                                         ␣
↪                                                                           |
| project_id              | 99d8a6cd24734f2aa3fe70140fbdbd64                           ␣
↪                                                                           |
| revision_number         | 2                                                          ␣
↪                                                                           |
| routes                  |                                                            ␣
↪                                                                           |
| status                  | ACTIVE                                                     ␣
↪                                                                           |
| tags                    |                                                            ␣
↪                                                                           |
| updated_at              | 2020-02-11T00:10:29Z                                       ␣
↪                                                                           |
+-------------------------+------------------------------------------------------------
↪--------------------------------------------------------------------------+

'#' ip netns exec qrouter-d4cb763e-8578-484d-be6a-6d7da165e161 ip addr

1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default␣
↪qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
        valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host
        valid_lft forever preferred_lft forever
14: qr-4c7615a5-dd: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1450 qdisc noqueue state␣
↪UNKNOWN group default qlen 1000
    link/ether fa:16:3e:80:07:21 brd ff:ff:ff:ff:ff:ff
    inet 10.5.5.1/24 brd 10.5.5.255 scope global qr-4c7615a5-dd
        valid_lft forever preferred_lft forever
    inet6 fe80::f816:3eff:fe80:721/64 scope link
        valid_lft forever preferred_lft forever
```

Note: Luego de conectar el router a la subred interna, se ha creado una interfaz en el router con una IP dentro de la subred conectada. Además, el OVS bridge `br-int` tiene un nuevo puerto con esta interfaz conectada.

3. Establecer el gateway para nuestro router:

```
# Con Neutron (deprecated): neutron router-gateway-set R1 external_network
'#' openstack router set R1 --external-gateway external_network
```

- Comprobar cambios de configuración:

```
'#' ovs-vsctl show

496c7134-8b33-4aa9-b752-ab503fccd5d6
    Manager "ptcp:6640:127.0.0.1"
        is_connected: true
    Bridge br-int
        Controller "tcp:127.0.0.1:6633"
            is_connected: true
        fail_mode: secure
        Port "tapa2009c32-11"
            tag: 3
            Interface "tapa2009c32-11"
                type: internal
        Port br-int
            Interface br-int
                type: internal
        Port "qg-e364542a-a5"
            tag: 4
            Interface "qg-e364542a-a5"
                type: internal
        Port patch-tun
            Interface patch-tun
                type: patch
                options: {peer=patch-int}
        Port "qr-4c7615a5-dd"
            tag: 3
            Interface "qr-4c7615a5-dd"
                type: internal
        Port int-br-ex
            Interface int-br-ex
                type: patch
                options: {peer=phy-br-ex}

'#' openstack router show R1

+--------------------------+------------------------------------------------------------
↪---------------------------------------------------------------------------------------
↪--------------------------------------------+
| Field                    | Value                                                     ␣
↪                                                                                      ␣
↪                                            |
+--------------------------+------------------------------------------------------------
↪---------------------------------------------------------------------------------------
↪--------------------------------------------+
| admin_state_up           | UP                                                        ␣
↪                                                                                      ␣
↪                                            |
| availability_zone_hints |                                                            ␣
↪                                                                                      ␣
↪                                            |
| availability_zones       | nova                                                      ␣
↪                                                                                      ␣
↪                                            |
```

(continues on next page)

(continued from previous page)

```
| created_at              | 2020-02-11T00:08:40Z                                        ␣
↪                                                                                          ␣
↪                                           |
| description             |                                                             ␣
↪                                                                                          ␣
↪                                           |
| distributed             | False                                                       ␣
↪                                                                                          ␣
↪                                           |
| external_gateway_info   | {"network_id": "f5dad5c1-bba9-41c5-844f-bd19a6a124aa",
↪"enable_snat": true, "external_fixed_ips": [{"subnet_id": "a6ae14ab-2287-4d0d-b8eb-
↪0f503792f32c", "ip_address": "192.168.1.151"}]} |
| flavor_id               | None                                                        ␣
↪                                                                                          ␣
↪                                           |
| ha                      | False                                                       ␣
↪                                                                                          ␣
↪                                           |
| id                      | d4cb763e-8578-484d-be6a-6d7da165e161                        ␣
↪                                                                                          ␣
↪                                           |
| interfaces_info         | [{"subnet_id": "8168b012-2c3c-4114-8145-963dd6646793",
↪"ip_address": "10.5.5.1", "port_id": "4c7615a5-dd19-43d8-833f-37e3505b8175"}]        ␣
↪                                              |
| name                    | R1                                                          ␣
↪                                                                                          ␣
↪                                           |
| project_id              | 99d8a6cd24734f2aa3fe70140fbdbd64                            ␣
↪                                                                                          ␣
↪                                           |
| revision_number         | 4                                                           ␣
↪                                                                                          ␣
↪                                           |
| routes                  |                                                             ␣
↪                                                                                          ␣
↪                                           |
| status                  | ACTIVE                                                      ␣
↪                                                                                          ␣
↪                                           |
| tags                    |                                                             ␣
↪                                                                                          ␣
↪                                           |
| updated_at              | 2020-02-11T00:53:20Z                                        ␣
↪                                                                                          ␣
↪                                           |
+-------------------------+-----------------------------------------------------------
↪-----------------------------------------------------------------------------------
↪-----------------------------------------+

'#' ip netns exec qrouter-d4cb763e-8578-484d-be6a-6d7da165e161 ip addr

1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default␣
↪qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
    valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host
```

(continues on next page)

(continued from previous page)

```
    valid_lft forever preferred_lft forever
14: qr-4c7615a5-dd: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1450 qdisc noqueue state␣
↪UNKNOWN group default qlen 1000
    link/ether fa:16:3e:80:07:21 brd ff:ff:ff:ff:ff:ff
    inet 10.5.5.1/24 brd 10.5.5.255 scope global qr-4c7615a5-dd
    valid_lft forever preferred_lft forever
    inet6 fe80::f816:3eff:fe80:721/64 scope link
    valid_lft forever preferred_lft forever
15: qg-e364542a-a5: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc noqueue state␣
↪UNKNOWN group default qlen 1000
    link/ether fa:16:3e:be:4c:9a brd ff:ff:ff:ff:ff:ff
    inet 192.168.1.151/24 brd 192.168.1.255 scope global qg-e364542a-a5
    valid_lft forever preferred_lft forever
    inet6 2800:200:e840:2918:f816:3eff:febe:4c9a/64 scope global mngtmpaddr dynamic
    valid_lft 3598sec preferred_lft 3598sec
    inet6 fe80::f816:3eff:febe:4c9a/64 scope link
    valid_lft forever preferred_lft forever
```

Note: Luego de establecer el gateway para nuestro router se crea una nueva interfaz en el router y se le asigna una IP dentro del rango de IPs a la subred que tiene como gateway (red externa en este caso). Viendo la información del router vemos que se trata de un SNAT.

15.3.10 Listar las redes y subredes creadas

- Podemos ver que contamos con una red interna y otra externa:

```
'#' openstack network list

+--------------------------------------+-----------------+--------------------------
↪-----------+
| ID                                   | Name            | Subnets                  ␣
↪           |
+--------------------------------------+-----------------+--------------------------
↪-----------+
| 2cf9c274-8592-476b-bde5-41e930e01577 | intnet          | 8168b012-2c3c-4114-8145-
↪963dd6646793 |
| f5dad5c1-bba9-41c5-844f-bd19a6a124aa | external_network | a6ae14ab-2287-4d0d-b8eb-
↪0f503792f32c |
+--------------------------------------+-----------------+--------------------------
↪-----------+
```

- Para cada red creada se le ha asignado una subred:

```
'#' openstack subnet list

+--------------------------------------+--------------+------------------------------
↪--------+----------------+
| ID                                   | Name         | Network                      ␣
↪        | Subnet         |
+--------------------------------------+--------------+------------------------------
↪--------+----------------+
| 8168b012-2c3c-4114-8145-963dd6646793 | subnet1      | 2cf9c274-8592-476b-bde5-
↪41e930e01577 | 10.5.5.0/24    |
```

(continues on next page)

(continued from previous page)

```
| a6ae14ab-2287-4d0d-b8eb-0f503792f32c | public_subnet | f5dad5c1-bba9-41c5-844f-
↪bd19a6a124aa | 192.168.1.0/24 |
+--------------------------------------+---------------+------------------------------
↪--------+----------------+
```

15.3.11 Editar el security group de un proyecto

1. Obtener el ID del security group de un proyecto:

- Listar proyectos:

```
'#' openstack project list

+----------------------------------+-------------+
| ID                               | Name        |
+----------------------------------+-------------+
| 0c2bc29526f4465c95b8eaefcfae7b7c | admin       |
| 99d8a6cd24734f2aa3fe70140fbdbd64 | testproject |
| ed14e8f780b84664accc6aa2d6673624 | services    |
+----------------------------------+-------------+
```

- Ver el ID del security group según el ID del proyecto:

```
'#' openstack security group list

+--------------------------------------+---------+------------------------+-----------
↪-----------------------+
| ID                                   | Name    | Description            | Project  ␣
↪                       |
+--------------------------------------+---------+------------------------+-----------
↪-----------------------+
| 2bc3934c-debf-4477-917c-9f01e23e366e | default | Default security group |          ␣
↪                       |
| 3a9e73cd-0608-49bf-b295-94b987a920ec | default | Default security group |␣
↪99d8a6cd24734f2aa3fe70140fbdbd64 |
| 7e145dac-5186-4b0c-afad-a6766d3818a7 | default | Default security group |␣
↪0c2bc29526f4465c95b8eaefcfae7b7c |
+--------------------------------------+---------+------------------------+-----------
↪-----------------------+
```

Relacionando el ID del proyecto `testproject` vemos que el ID del security group relacionado es `3a9e73cd-0608-49bf-b295-94b987a920ec`

2. Añadir una regla al security group que permita el tráfico ICMP y SSH hacia las instancias del proyecto:

- Regla para ICMP: permite todo el tráfico entrante ICMP a la instancia desde cualquier IP:

```
'#' openstack security group rule create --remote-ip 0.0.0.0/0 --protocol icmp --
↪ingress 3a9e73cd-0608-49bf-b295-94b987a920ec

+-------------------+--------------------------------------+
| Field             | Value                                |
+-------------------+--------------------------------------+
| created_at        | 2020-02-11T01:46:21Z                 |
| description       |                                      |
| direction         | ingress                              |
```

(continues on next page)

(continued from previous page)

```
| ether_type        | IPv4                                 |
| id                | c0eb2ead-fad7-4400-8958-dcf6d43698c7 |
| name              | None                                 |
| port_range_max    | None                                 |
| port_range_min    | None                                 |
| project_id        | 99d8a6cd24734f2aa3fe70140fbdbd64     |
| protocol          | icmp                                 |
| remote_group_id   | None                                 |
| remote_ip_prefix  | 0.0.0.0/0                            |
| revision_number   | 0                                    |
| security_group_id | 3a9e73cd-0608-49bf-b295-94b987a920ec |
| updated_at        | 2020-02-11T01:46:21Z                 |
+-------------------+--------------------------------------+
```

- Regla para SSH: permite todo el tráfico entrante SSH (puerto 22, TCP) a la instancia desde cualquier IP:

```
'#' openstack security group rule create --remote-ip 0.0.0.0/0 --dst-port 22 --
↪protocol tcp --ingress 3a9e73cd-0608-49bf-b295-94b987a920ec

+-------------------+--------------------------------------+
| Field             | Value                                |
+-------------------+--------------------------------------+
| created_at        | 2020-02-11T01:46:40Z                 |
| description       |                                      |
| direction         | ingress                              |
| ether_type        | IPv4                                 |
| id                | 6aaffa9a-932b-432f-aa4e-bacb646fecb3 |
| name              | None                                 |
| port_range_max    | 22                                   |
| port_range_min    | 22                                   |
| project_id        | 99d8a6cd24734f2aa3fe70140fbdbd64     |
| protocol          | tcp                                  |
| remote_group_id   | None                                 |
| remote_ip_prefix  | 0.0.0.0/0                            |
| revision_number   | 0                                    |
| security_group_id | 3a9e73cd-0608-49bf-b295-94b987a920ec |
| updated_at        | 2020-02-11T01:46:40Z                 |
+-------------------+--------------------------------------+
```

15.3.12 Crear una instancia (1)

1. Crear una instancia seleccionando la imagen, el flavor, la red a la cual deseemos que se conecte nuestra VM y el nombre de la instancia:

```
'#' openstack server create --image cirros --flavor 10 --nic net-id=2cf9c274-8592-
↪476b-bde5-41e930e01577 inst1

+-------------------------------------+------------------------------------------------
↪+
| Field                               | Value                                          ␣
↪|
+-------------------------------------+------------------------------------------------
↪+
| OS-DCF:diskConfig                   | MANUAL                                         ␣
↪|
```

(continues on next page)

(continued from previous page)

```
| OS-EXT-AZ:availability_zone          |                                                                                ␣
↪|
| OS-EXT-SRV-ATTR:host                 | None                                                                           ␣
↪|
| OS-EXT-SRV-ATTR:hypervisor_hostname | None                                                                           ␣
↪|
| OS-EXT-SRV-ATTR:instance_name        |                                                                                ␣
↪|
| OS-EXT-STS:power_state               | NOSTATE                                                                        ␣
↪|
| OS-EXT-STS:task_state                | scheduling                                                                     ␣
↪|
| OS-EXT-STS:vm_state                  | building                                                                       ␣
↪|
| OS-SRV-USG:launched_at               | None                                                                           ␣
↪|
| OS-SRV-USG:terminated_at             | None                                                                           ␣
↪|
| accessIPv4                           |                                                                                ␣
↪|
| accessIPv6                           |                                                                                ␣
↪|
| addresses                            |                                                                                ␣
↪|
| adminPass                            | S43XAdqjgaYV                                                                   ␣
↪|
| config_drive                         |                                                                                ␣
↪|
| created                              | 2020-02-11T02:53:57Z                                                           ␣
↪|
| flavor                               | m1.tiniest (10)                                                                ␣
↪|
| hostId                               |                                                                                ␣
↪|
| id                                   | 1265b4c2-d15f-4279-9148-24454ee294ef                                           ␣
↪|
| image                                | cirros (56dd3671-de42-40db-9637-7c5bef599d11)                                  ␣
↪|
| key_name                             | None                                                                           ␣
↪|
| name                                 | inst1                                                                          ␣
↪|
| progress                             | 0                                                                              ␣
↪|
| project_id                           | 99d8a6cd24734f2aa3fe70140fbdbd64                                               ␣
↪|
| properties                           |                                                                                ␣
↪|
| security_groups                      | name='default'                                                                 ␣
↪|
| status                               | BUILD                                                                          ␣
↪|
| updated                              | 2020-02-11T02:53:57Z                                                           ␣
↪|
| user_id                              | 6643474fffb548b4bd4fb3d6a09d9ecd                                               ␣
↪|
| volumes_attached                     |                                                                                ␣
↪|
```

(continues on next page)

(continued from previous page)

```
+--------------------------------------+-----------------------------------------------
↪+
```

2. Despues de un momento, veremos que el estado de la VM se encuentra en activo y tiene asignada una dirección IP:

```
'#' openstack server show inst1

+--------------------------------------+-----------------------------------------------
↪-----------+
| Field                                | Value                                         ␣
↪           |
+--------------------------------------+-----------------------------------------------
↪-----------+
| OS-DCF:diskConfig                    | MANUAL                                        ␣
↪           |
| OS-EXT-AZ:availability_zone          | nova                                          ␣
↪           |
| OS-EXT-SRV-ATTR:host                 | controllernode1.localdomain                   ␣
↪           |
| OS-EXT-SRV-ATTR:hypervisor_hostname | controllernode1.localdomain                   ␣
↪           |
| OS-EXT-SRV-ATTR:instance_name        | instance-00000002                             ␣
↪           |
| OS-EXT-STS:power_state               | Running                                       ␣
↪           |
| OS-EXT-STS:task_state                | None                                          ␣
↪           |
| OS-EXT-STS:vm_state                  | active                                        ␣
↪           |
| OS-SRV-USG:launched_at               | 2020-02-11T02:54:03.000000                    ␣
↪           |
| OS-SRV-USG:terminated_at             | None                                          ␣
↪           |
| accessIPv4                           |                                               ␣
↪           |
| accessIPv6                           |                                               ␣
↪           |
| addresses                            | intnet=10.5.5.3                               ␣
↪           |
| config_drive                         |                                               ␣
↪           |
| created                              | 2020-02-11T02:53:57Z                          ␣
↪           |
| flavor                               | m1.tiniest (10)                               ␣
↪           |
| hostId                               |␣
↪e122795a13958abb7b13d1f480d04f15b58d09d04ae475133c0005a2 |
| id                                   | 1265b4c2-d15f-4279-9148-24454ee294ef          ␣
↪           |
| image                                | cirros (56dd3671-de42-40db-9637-7c5bef599d11)␣
↪           |
| key_name                             | None                                          ␣
↪           |
| name                                 | inst1                                         ␣
↪           |
```

(continues on next page)

(continued from previous page)

```
| progress                            | 0                                                                                ␣
↪           |
| project_id                          | 99d8a6cd24734f2aa3fe70140fbdbd64                                                 ␣
↪           |
| properties                          |                                                                                  ␣
↪           |
| security_groups                     | name='default'                                                                   ␣
↪           |
| status                              | ACTIVE                                                                           ␣
↪           |
| updated                             | 2020-02-11T02:54:03Z                                                             ␣
↪           |
| user_id                             | 6643474fffb548b4bd4fb3d6a09d9ecd                                                 ␣
↪           |
| volumes_attached                    |                                                                                  ␣
↪           |
+-------------------------------------+----------------------------------------------
↪-----------+
```

3. Podemos conectarnos a la nueva instancia creada por SSH:

```
'#' ip netns exec qrouter-d4cb763e-8578-484d-be6a-6d7da165e161 ssh cirros@10.5.5.3

The authenticity of host '10.5.5.3 (10.5.5.3)' can't be established.
ECDSA key fingerprint is SHA256:NcjHkAHTVvp9GRDizktzGg5mlQJnjyCXA7ohVsVV9yM.
ECDSA key fingerprint is MD5:2c:4f:d8:42:93:d3:fa:fe:16:1c:c8:fa:0c:ad:60:12.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added '10.5.5.3' (ECDSA) to the list of known hosts.
cirros@10.5.5.3's password:

$ whoami
cirros

$ ip addr
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue qlen 1
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
       valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host
       valid_lft forever preferred_lft forever
2: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1450 qdisc pfifo_fast qlen 1000
    link/ether fa:16:3e:07:1d:bb brd ff:ff:ff:ff:ff:ff
    inet 10.5.5.3/24 brd 10.5.5.255 scope global eth0
       valid_lft forever preferred_lft forever
    inet6 fe80::f816:3eff:fe07:1dbb/64 scope link
       valid_lft forever preferred_lft forever
```

15.3.13 Asignar una Floating IP a una instancia

Dentro de nuestra red y subred pública, crearemos una Floating IP para poder asignársela a la instancia:

1. Listar IDs de Redes y Subredes:

```
'#' openstack network list
```

(continues on next page)

(continued from previous page)

```
+--------------------------------------+-----------------+--------------------------
↪-----------+
| ID                                   | Name            | Subnets                  ␣
↪           |
+--------------------------------------+-----------------+--------------------------
↪-----------+
| 2cf9c274-8592-476b-bde5-41e930e01577 | intnet          | 8168b012-2c3c-4114-8145-
↪963dd6646793 |
| f5dad5c1-bba9-41c5-844f-bd19a6a124aa | external_network | a6ae14ab-2287-4d0d-b8eb-
↪0f503792f32c |
+--------------------------------------+-----------------+--------------------------
↪-----------+

'#' openstack subnet list

+--------------------------------------+--------------+------------------------------
↪--------+----------------+
| ID                                   | Name         | Network                      ␣
↪        | Subnet         |
+--------------------------------------+--------------+------------------------------
↪--------+----------------+
| 8168b012-2c3c-4114-8145-963dd6646793 | subnet1      | 2cf9c274-8592-476b-bde5-
↪41e930e01577 | 10.5.5.0/24    |
| a6ae14ab-2287-4d0d-b8eb-0f503792f32c | public_subnet | f5dad5c1-bba9-41c5-844f-
↪bd19a6a124aa | 192.168.1.0/24 |
+--------------------------------------+--------------+------------------------------
↪--------+----------------+
```

2. Reservar una IP flotante dentro de la subred pública seleccionada:

```
'#' openstack floating ip create --subnet a6ae14ab-2287-4d0d-b8eb-0f503792f32c␣
↪f5dad5c1-bba9-41c5-844f-bd19a6a124aa

+---------------------+--------------------------------------+
| Field               | Value                                |
+---------------------+--------------------------------------+
| created_at          | 2020-02-11T03:28:04Z                 |
| description         |                                      |
| fixed_ip_address    | None                                 |
| floating_ip_address | 192.168.1.152                        |
| floating_network_id | f5dad5c1-bba9-41c5-844f-bd19a6a124aa |
| id                  | 3ef31d8b-4918-473e-9696-f3820230d393 |
| name                | 192.168.1.152                        |
| port_id             | None                                 |
| project_id          | 99d8a6cd24734f2aa3fe70140fbdbd64     |
| qos_policy_id       | None                                 |
| revision_number     | 0                                    |
| router_id           | None                                 |
| status              | DOWN                                 |
| subnet_id           | a6ae14ab-2287-4d0d-b8eb-0f503792f32c |
| updated_at          | 2020-02-11T03:28:04Z                 |
+---------------------+--------------------------------------+
```

Note: Se ha reservado la IP flotante `192.168.1.152`, pues la subred `public_subnet` tiene reservado el pool de asignación `192.168.1.150-192.168.1.200`.

```
'#' openstack ip availability list

+--------------------------------------+------------------+-----------+-----------+
| Network ID                           | Network Name     | Total IPs | Used IPs |
+--------------------------------------+------------------+-----------+-----------+
| f5dad5c1-bba9-41c5-844f-bd19a6a124aa | external_network |        51 |        2 |
| 2cf9c274-8592-476b-bde5-41e930e01577 | intnet           |       253 |        3 |
+--------------------------------------+------------------+-----------+-----------+
```

3. Asignar la Floating IP a la instancia:

```
'#' openstack server add floating ip inst1 192.168.1.152
```

4. Podemos probar conectividad a esta instancia desde cualquier máquina de nuestra red física local:

```
$ ping 192.168.1.152
PING 192.168.1.152 (192.168.1.152) 56(84) bytes of data.
64 bytes from 192.168.1.152: icmp_seq=1 ttl=63 time=48.0 ms
^C
--- 192.168.1.152 ping statistics ---
1 packets transmitted, 1 received, 0% packet loss, time 0ms
rtt min/avg/max/mdev = 48.024/48.024/48.024/0.000 ms

$ ssh cirros@192.168.1.152
The authenticity of host '192.168.1.152 (192.168.1.152)' can't be established.
ECDSA key fingerprint is SHA256:NcjHkAHTVvp9GRDizktzGg5mlQJnjyCXA7ohVsVV9yM.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added '192.168.1.152' (ECDSA) to the list of known hosts.
cirros@192.168.1.152's password:

$ whoami
cirros

$ ip addr
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue qlen 1
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
       valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host
       valid_lft forever preferred_lft forever
2: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1450 qdisc pfifo_fast qlen 1000
    link/ether fa:16:3e:07:1d:bb brd ff:ff:ff:ff:ff:ff
    inet 10.5.5.3/24 brd 10.5.5.255 scope global eth0
       valid_lft forever preferred_lft forever
    inet6 fe80::f816:3eff:fe07:1dbb/64 scope link
       valid_lft forever preferred_lft forever
```

15.3.14 Crear un key pair

1. Crear un nuevo key pair:

```
#create a keypair named mykeypair and copy to mykeypair.key file
'#' openstack keypair create mykeypair >> mykeypair.key
```

Hemos creado un key pair llamado `mykeypair` y hemos guardado la llave en el archivo llamado `mykeypair.key`:

15.3.15 Crear un security group

Link: Configure access and security for instances - Openstack Docs

1. Crear un security group con un nombre y descripción específicos:

```
'#' openstack security group create testsecgroup --description "Security Group Test"

+-----------------+-------------------------------------------------------------------------
↪-------------------------------------------------------------------------------------------
↪+
| Field           | Value                                                                   ␣
↪                                                                                          ␣
↪|
+-----------------+-------------------------------------------------------------------------
↪-------------------------------------------------------------------------------------------
↪+
| created_at      | 2020-02-11T04:29:20Z                                                    ␣
↪                                                                                          ␣
↪|
| description     | Security Group Test                                                     ␣
↪                                                                                          ␣
↪|
| id              | 6db48e49-c56d-450c-8692-faa0bad3c081                                    ␣
↪                                                                                          ␣
↪|
| name            | testsecgroup                                                            ␣
↪                                                                                          ␣
↪|
| project_id      | 99d8a6cd24734f2aa3fe70140fbdbd64                                        ␣
↪                                                                                          ␣
↪|
| revision_number | 2                                                                       ␣
↪                                                                                          ␣
↪|
| rules           | created_at='2020-02-11T04:29:20Z', direction='egress', ethertype=
↪'IPv6', id='632c1a1f-a6ce-45d1-bf50-0dfc4018467e', updated_at='2020-02-11T04:29:20Z
↪' |
|                 | created_at='2020-02-11T04:29:20Z', direction='egress', ethertype=
↪'IPv4', id='d663e37a-0a80-44ba-b557-ad7715afd0ff', updated_at='2020-02-11T04:29:20Z
↪' |
| updated_at      | 2020-02-11T04:29:20Z                                                    ␣
↪                                                                                          ␣
↪|
+-----------------+-------------------------------------------------------------------------
↪-------------------------------------------------------------------------------------------
↪+
```

2. Crear reglas para el nuevo security group:

- Permitir conexiones SSH remotas:

```
'#' openstack security group rule create testsecgroup --protocol tcp --dst-port 22:22␣
↪--remote-ip 0.0.0.0/0

+-------------------+--------------------------------------+
| Field             | Value                                |
+-------------------+--------------------------------------+
```

(continues on next page)

(continued from previous page)

```
| created_at        | 2020-02-11T04:36:55Z                 |
| description       |                                      |
| direction         | ingress                              |
| ether_type        | IPv4                                 |
| id                | fb4a6bb5-646d-4f02-a7cc-c01d460fb277 |
| name              | None                                 |
| port_range_max    | 22                                   |
| port_range_min    | 22                                   |
| project_id        | 99d8a6cd24734f2aa3fe70140fbdbd64     |
| protocol          | tcp                                  |
| remote_group_id   | None                                 |
| remote_ip_prefix  | 0.0.0.0/0                            |
| revision_number   | 0                                    |
| security_group_id | 6db48e49-c56d-450c-8692-faa0bad3c081 |
| updated_at        | 2020-02-11T04:36:55Z                 |
+-------------------+--------------------------------------+
```

- Permitir tráfico ICMP entrante:

```
'#' openstack security group rule create testsecgroup --protocol icmp

+-------------------+--------------------------------------+
| Field             | Value                                |
+-------------------+--------------------------------------+
| created_at        | 2020-02-11T04:42:00Z                 |
| description       |                                      |
| direction         | ingress                              |
| ether_type        | IPv4                                 |
| id                | c37a0ee4-7760-4e3f-9f24-d59076e6c1b8 |
| name              | None                                 |
| port_range_max    | None                                 |
| port_range_min    | None                                 |
| project_id        | 99d8a6cd24734f2aa3fe70140fbdbd64     |
| protocol          | icmp                                 |
| remote_group_id   | None                                 |
| remote_ip_prefix  | 0.0.0.0/0                            |
| revision_number   | 0                                    |
| security_group_id | 6db48e49-c56d-450c-8692-faa0bad3c081 |
| updated_at        | 2020-02-11T04:42:00Z                 |
+-------------------+--------------------------------------+
```

3. Ver detalles del security group y de las reglas dentro del grupo:

```
'#' openstack security group list

+--------------------------------------+-------------+------------------------+------
↪----------------------------+
| ID                                   | Name        | Description            |␣
↪Project                            |
+--------------------------------------+-------------+------------------------+------
↪----------------------------+
| 2bc3934c-debf-4477-917c-9f01e23e366e | default     | Default security group |     ␣
↪                            |
| 3a9e73cd-0608-49bf-b295-94b987a920ec | default     | Default security group |␣
↪99d8a6cd24734f2aa3fe70140fbdbd64 |
| 6db48e49-c56d-450c-8692-faa0bad3c081 | testsecgroup | Security Group Test    |␣
↪99d8a6cd24734f2aa3fe70140fbdbd64 |
```

(continues on next page)

(continued from previous page)

```
| 7e145dac-5186-4b0c-afad-a6766d3818a7 | default      | Default security group |␣
↪0c2bc29526f4465c95b8eaefcfae7b7c |
| a2475e5d-a9dd-4ee1-a60f-2c2a15f1f299 | default      | Default security group |␣
↪ed14e8f780b84664accc6aa2d6673624 |
+--------------------------------------+--------------+------------------------+------
↪-----------------------------+

'#' openstack security group rule list testsecgroup

+--------------------------------------+-------------+-----------+------------+-------
↪----------------+
| ID                                   | IP Protocol | IP Range  | Port Range |␣
↪Remote Security Group |
+--------------------------------------+-------------+-----------+------------+-------
↪----------------+
| 632c1a1f-a6ce-45d1-bf50-0dfc4018467e | None        | None      |            | None ␣
↪                |
| c37a0ee4-7760-4e3f-9f24-d59076e6c1b8 | icmp        | 0.0.0.0/0 |            | None ␣
↪                |
| d663e37a-0a80-44ba-b557-ad7715afd0ff | None        | None      |            | None ␣
↪                |
| fb4a6bb5-646d-4f02-a7cc-c01d460fb277 | tcp         | 0.0.0.0/0 | 22:22      | None ␣
↪                |
+--------------------------------------+-------------+-----------+------------+-------
↪----------------+

'#' openstack security group rule show fb4a6bb5-646d-4f02-a7cc-c01d460fb277

+-------------------+--------------------------------------+
| Field             | Value                                |
+-------------------+--------------------------------------+
| created_at        | 2020-02-11T04:36:55Z                 |
| description       |                                      |
| direction         | ingress                              |
| ether_type        | IPv4                                 |
| id                | fb4a6bb5-646d-4f02-a7cc-c01d460fb277 |
| name              | None                                 |
| port_range_max    | 22                                   |
| port_range_min    | 22                                   |
| project_id        | 99d8a6cd24734f2aa3fe70140fbdbd64     |
| protocol          | tcp                                  |
| remote_group_id   | None                                 |
| remote_ip_prefix  | 0.0.0.0/0                            |
| revision_number   | 0                                    |
| security_group_id | 6db48e49-c56d-450c-8692-faa0bad3c081 |
| updated_at        | 2020-02-11T04:36:55Z                 |
+-------------------+--------------------------------------+
```

15.3.16 Crear una instancia (2)

Link: Launch an instance on the provider network - Openstack Docs

1. Ahora crearemos una instancia especificando un security group y un keypair:

```
'#' openstack server create --image cirros --flavor 10 --key-name mykeypair --
↪security-group testsecgroup --nic net-id=2cf9c274-8592-476b-bde5-41e930e01577 inst2

+-------------------------------------+------------------------------------------------
↪+
| Field                               | Value                                          ␣
↪|
+-------------------------------------+------------------------------------------------
↪+
| OS-DCF:diskConfig                   | MANUAL                                         ␣
↪|
| OS-EXT-AZ:availability_zone         |                                                ␣
↪|
| OS-EXT-SRV-ATTR:host                | None                                           ␣
↪|
| OS-EXT-SRV-ATTR:hypervisor_hostname | None                                           ␣
↪|
| OS-EXT-SRV-ATTR:instance_name       |                                                ␣
↪|
| OS-EXT-STS:power_state              | NOSTATE                                        ␣
↪|
| OS-EXT-STS:task_state               | scheduling                                     ␣
↪|
| OS-EXT-STS:vm_state                 | building                                       ␣
↪|
| OS-SRV-USG:launched_at              | None                                           ␣
↪|
| OS-SRV-USG:terminated_at            | None                                           ␣
↪|
| accessIPv4                          |                                                ␣
↪|
| accessIPv6                          |                                                ␣
↪|
| addresses                           |                                                ␣
↪|
| adminPass                           | q2rGS9Hbw7nF                                   ␣
↪|
| config_drive                        |                                                ␣
↪|
| created                             | 2020-02-11T04:58:15Z                           ␣
↪|
| flavor                              | m1.tiniest (10)                                ␣
↪|
| hostId                              |                                                ␣
↪|
| id                                  | e597bba0-7440-49e3-b5ee-f972c2f640ab           ␣
↪|
| image                               | cirros (56dd3671-de42-40db-9637-7c5bef599d11)␣
↪|
| key_name                            | mykeypair                                      ␣
↪|
| name                                | inst2                                          ␣
↪|
| progress                            | 0                                              ␣
↪|
| project_id                          | 99d8a6cd24734f2aa3fe70140fbdbd64               ␣
↪|
```

(continues on next page)

(continued from previous page)

```
| properties                          |                                                                                 ␣
↪|
| security_groups                     | name='6db48e49-c56d-450c-8692-faa0bad3c081'                                     ␣
↪|
| status                              | BUILD                                                                           ␣
↪|
| updated                             | 2020-02-11T04:58:15Z                                                            ␣
↪|
| user_id                             | 6643474fffb548b4bd4fb3d6a09d9ecd                                                ␣
↪|
| volumes_attached                    |                                                                                 ␣
↪|
+-------------------------------------+---------------------------------------------------
↪+
```

2. Ver detalles de la instancia creada una vez que está activa:

```
'#' openstack server show inst2

+-------------------------------------+---------------------------------------------------
↪-----------+
| Field                               | Value                                                                           ␣
↪           |
+-------------------------------------+---------------------------------------------------
↪-----------+
| OS-DCF:diskConfig                   | MANUAL                                                                          ␣
↪           |
| OS-EXT-AZ:availability_zone         | nova                                                                            ␣
↪           |
| OS-EXT-SRV-ATTR:host                | controllernode1.localdomain                                                     ␣
↪           |
| OS-EXT-SRV-ATTR:hypervisor_hostname | controllernode1.localdomain                                                     ␣
↪           |
| OS-EXT-SRV-ATTR:instance_name       | instance-00000003                                                               ␣
↪           |
| OS-EXT-STS:power_state              | Running                                                                         ␣
↪           |
| OS-EXT-STS:task_state               | None                                                                            ␣
↪           |
| OS-EXT-STS:vm_state                 | active                                                                          ␣
↪           |
| OS-SRV-USG:launched_at              | 2020-02-11T04:58:21.000000                                                      ␣
↪           |
| OS-SRV-USG:terminated_at            | None                                                                            ␣
↪           |
| accessIPv4                          |                                                                                 ␣
↪           |
| accessIPv6                          |                                                                                 ␣
↪           |
| addresses                           | intnet=10.5.5.4                                                                 ␣
↪           |
| config_drive                        |                                                                                 ␣
↪           |
| created                             | 2020-02-11T04:58:15Z                                                            ␣
↪           |
| flavor                              | m1.tiniest (10)                                                                 ␣
↪           |
```

(continues on next page)

(continued from previous page)

```
| hostId                               |␣
↪e122795a13958abb7b13d1f480d04f15b58d09d04ae475133c0005a2 |
| id                                   | e597bba0-7440-49e3-b5ee-f972c2f640ab␣
↪           |
| image                                | cirros (56dd3671-de42-40db-9637-7c5bef599d11)␣
↪           |
| key_name                             | mykeypair␣
↪           |
| name                                 | inst2␣
↪           |
| progress                             | 0␣
↪           |
| project_id                           | 99d8a6cd24734f2aa3fe70140fbdbd64␣
↪           |
| properties                           |␣
↪           |
| security_groups                      | name='testsecgroup'␣
↪           |
| status                               | ACTIVE␣
↪           |
| updated                              | 2020-02-11T04:58:21Z␣
↪           |
| user_id                              | 6643474fffb548b4bd4fb3d6a09d9ecd␣
↪           |
| volumes_attached                     |␣
↪           |
+--------------------------------------+------------------------------------------------
↪-----------+
```

3. Asignar un IP flotante a la instancia creada:

```
'#' openstack floating ip create --subnet a6ae14ab-2287-4d0d-b8eb-0f503792f32c␣
↪f5dad5c1-bba9-41c5-844f-bd19a6a124aa

+---------------------+--------------------------------------+
| Field               | Value                                |
+---------------------+--------------------------------------+
| created_at          | 2020-02-11T05:03:35Z                 |
| description         |                                      |
| fixed_ip_address    | None                                 |
| floating_ip_address | 192.168.1.153                        |
| floating_network_id | f5dad5c1-bba9-41c5-844f-bd19a6a124aa |
| id                  | a4fc9dad-5e2a-4f56-ad0d-ae386b7238d1 |
| name                | 192.168.1.153                        |
| port_id             | None                                 |
| project_id          | 99d8a6cd24734f2aa3fe70140fbdbd64     |
| qos_policy_id       | None                                 |
| revision_number     | 0                                    |
| router_id           | None                                 |
| status              | DOWN                                 |
| subnet_id           | a6ae14ab-2287-4d0d-b8eb-0f503792f32c |
| updated_at          | 2020-02-11T05:03:35Z                 |
+---------------------+--------------------------------------+
```

```
'#' openstack server add floating ip inst2 192.168.1.153
```

4. Conectarse a la instancia desde cualquier máquina dentro de la red local empleando la key pair:

- Primero debemos obtener el archivo con el contenido de la llave en la máquina que desea conectarse a la instancia. Por ejemplo, pasando por scp la llave desde el controller node a la máquina que se desea conectar:

```
'#' scp mykeypair.key gabriel@192.168.1.9:/home/gabriel/Downloads
```

- Una vez que tengamos la llave localmente, cambiar los permisos del archivo:

```
$ chmod 600 mykeypair.key
```

- Usar la llave para conectarnos a la instancia de forma **passwordless**:

```
$ ssh -i mykeypair.key cirros@192.168.1.153

$ whoami
cirros

$ ip addr
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue qlen 1
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
    valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host
    valid_lft forever preferred_lft forever
2: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1450 qdisc pfifo_fast qlen 1000
    link/ether fa:16:3e:f5:21:35 brd ff:ff:ff:ff:ff:ff
    inet 10.5.5.4/24 brd 10.5.5.255 scope global eth0
    valid_lft forever preferred_lft forever
    inet6 fe80::f816:3eff:fef5:2135/64 scope link
    valid_lft forever preferred_lft forever

$ df -h
Filesystem                Size      Used Available Use% Mounted on
/dev                     51.2M         0     51.2M   0% /dev
/dev/vda1               978.9M     24.0M    914.1M   3% /
tmpfs                    55.2M         0     55.2M   0% /dev/shm
tmpfs                    55.2M     92.0K     55.1M   0% /run
```

5. También podemos conectarnos a la instancia usando el navegador, ingresando la URL que aparece con el siguiente comando:

```
'#' openstack console url show inst2

+-------+-------------------------------------------------------------------------------
↪-------+
| Field | Value
↪       |
+-------+-------------------------------------------------------------------------------
↪-------+
| type  | novnc
↪       |
| url   | http://192.168.1.100:6080/vnc_auto.html?token=957c0893-e5f1-409f-a0b8-
↪d3a833a09863 |
+-------+-------------------------------------------------------------------------------
↪-------+
```

15.3.17 Crear snapshot de una imagen

Link: Use snapshots to migrate instances - Openstack Docs

Crear un snapshot de una imagen definida:

```
'#' openstack server image create --name snap1 inst2

+------------------+------------------------------------------------------------------------
↪------------------------------------------------------------------------------------------
↪------------------------------------------------------------------------------------------
↪----------------------------------+
| Field            | Value
↪                                                                                          ␣
↪                                                                                          ␣
↪                                   |
+------------------+------------------------------------------------------------------------
↪------------------------------------------------------------------------------------------
↪------------------------------------------------------------------------------------------
↪----------------------------------+
| checksum         | None
↪                                                                                          ␣
↪                                                                                          ␣
↪                                   |
| container_format | None
↪                                                                                          ␣
↪                                                                                          ␣
↪                                   |
| created_at       | 2020-02-11T06:06:03Z
↪                                                                                          ␣
↪                                                                                          ␣
↪                                   |
| disk_format      | None
↪                                                                                          ␣
↪                                                                                          ␣
↪                                   |
| file             | /v2/images/bc9fd276-2b21-44a0-aadc-fd971c2034aa/file
↪                                                                                          ␣
↪                                                                                          ␣
↪                                   |
| id               | bc9fd276-2b21-44a0-aadc-fd971c2034aa
↪                                                                                          ␣
↪                                                                                          ␣
↪                                   |
| min_disk         | 1
↪                                                                                          ␣
↪                                                                                          ␣
↪                                   |
| min_ram          | 128
↪                                                                                          ␣
↪                                                                                          ␣
↪                                   |
| name             | snap1
↪                                                                                          ␣
↪                                                                                          ␣
↪                                   |
| owner            | 99d8a6cd24734f2aa3fe70140fbdbd64
↪                                                                                          ␣
↪                                                                                          ␣
↪                                   |
```

(continues on next page)

(continued from previous page)

```
| properties       | base_image_ref='56dd3671-de42-40db-9637-7c5bef599d11', boot_
↪roles='_member_,admin', image_type='snapshot', instance_uuid='e597bba0-7440-49e3-
↪b5ee-f972c2f640ab', owner_project_name='testproject', owner_user_name='testuser1',␣
↪user_id='6643474fffb548b4bd4fb3d6a09d9ecd' |
| protected        | False                                                          ␣
↪                                                                                  ␣
↪                                                                                  ␣
↪                                  |
| schema           | /v2/schemas/image                                              ␣
↪                                                                                  ␣
↪                                                                                  ␣
↪                                  |
| size             | None                                                           ␣
↪                                                                                  ␣
↪                                                                                  ␣
↪                                  |
| status           | queued                                                         ␣
↪                                                                                  ␣
↪                                                                                  ␣
↪                                  |
| tags             |                                                                ␣
↪                                                                                  ␣
↪                                                                                  ␣
↪                                  |
| updated_at       | 2020-02-11T06:06:03Z                                           ␣
↪                                                                                  ␣
↪                                                                                  ␣
↪                                  |
| virtual_size     | None                                                           ␣
↪                                                                                  ␣
↪                                                                                  ␣
↪                                  |
| visibility       | private                                                        ␣
↪                                                                                  ␣
↪                                                                                  ␣
↪                                  |
+------------------+------------------------------------------------------------------
↪-----------------------------------------------------------------------------------
↪-----------------------------------------------------------------------------------
↪----------------------------------+
```

Podremos correr una nueva instancia desde este snapshot realizado. El snapshot creado es una imagen más del tipo `image_type='snapshot'` como se ve en la sección `properties`:

```
'#' openstack image list
+--------------------------------------+--------+--------+
| ID                                   | Name   | Status |
+--------------------------------------+--------+--------+
| 56dd3671-de42-40db-9637-7c5bef599d11 | cirros | active |
| bc9fd276-2b21-44a0-aadc-fd971c2034aa | snap1  | active |
+--------------------------------------+--------+--------+
```

15.3.18 Crear un volumen

Link: Block Storage - Openstack Docs

Crear un volumen de 1 GB de tamaño llamado `vol1`:

```
'#' openstack volume create --size 1 vol1

+---------------------+--------------------------------------+
| Field               | Value                                |
+---------------------+--------------------------------------+
| attachments         | []                                   |
| availability_zone   | nova                                 |
| bootable            | false                                |
| consistencygroup_id | None                                 |
| created_at          | 2020-02-11T06:39:16.000000           |
| description         | None                                 |
| encrypted           | False                                |
| id                  | 325a4a42-d640-43d8-affe-8d5efacee6a4 |
| migration_status    | None                                 |
| multiattach         | False                                |
| name                | vol1                                 |
| properties          |                                      |
| replication_status  | None                                 |
| size                | 1                                    |
| snapshot_id         | None                                 |
| source_volid        | None                                 |
| status              | creating                             |
| type                | iscsi                                |
| updated_at          | None                                 |
| user_id             | 6643474fffb548b4bd4fb3d6a09d9ecd     |
+---------------------+--------------------------------------+

'#' openstack volume list

+--------------------------------------+------+-----------+------+-------------+
| ID                                   | Name | Status    | Size | Attached to |
+--------------------------------------+------+-----------+------+-------------+
| 325a4a42-d640-43d8-affe-8d5efacee6a4 | vol1 | available |    1 |             |
+--------------------------------------+------+-----------+------+-------------+
```

15.3.19 Conectar y montar un volumen a una instancia

1. Conectar el volumen `vol1` a la instancia `inst2`:

```
'#' openstack server add volume inst2 vol1

'#' openstack volume list
+--------------------------------------+------+--------+------+-----------------------
↪---------+
| ID                                   | Name | Status | Size | Attached to
↪         |
+--------------------------------------+------+--------+------+-----------------------
↪---------+
| 325a4a42-d640-43d8-affe-8d5efacee6a4 | vol1 | in-use |    1 | Attached to inst2 on /
↪dev/vdb  |
+--------------------------------------+------+--------+------+-----------------------
↪---------+
```

Comprobamos que el disco se ha conectado a la instancia `inst2` bajo la ruta `/dev/vdb`.

2. Dentro de la instancia formatearemos el volumen para que sea utilizable:

- Nos conectamos a la instancia y comprobamos que existe la ruta `/dev/vdb` dentro de la instancia:

```
$ ssh -i mykeypair.key cirros@192.168.1.153

$ ls /dev | grep vd
vda
vda1
vda15
vdb
```

- Formatear el volumen:

```
$ sudo mkfs.ext4 /dev/vdb
mke2fs 1.42.12 (29-Aug-2014)
Creating filesystem with 262144 4k blocks and 65536 inodes
Filesystem UUID: b4700792-4051-4657-91ed-9b07ca1153ae
Superblock backups stored on blocks:
        32768, 98304, 163840, 229376

Allocating group tables: done
Writing inode tables: done
Creating journal (8192 blocks): done
Writing superblocks and filesystem accounting information: done
```

3. Montar el volumen formateado bajo un directorio:

```
$ sudo mkdir /mydisk
$ sudo mount /dev/vdb /mydisk/

$ df -h
Filesystem                Size      Used Available Use% Mounted on
/dev                     51.2M         0     51.2M   0% /dev
/dev/vda1               978.9M     24.0M    914.1M   3% /
tmpfs                    55.2M         0     55.2M   0% /dev/shm
tmpfs                    55.2M     92.0K     55.1M   0% /run
/dev/vdb                975.9M      1.3M    907.4M   0% /mydisk
```

Comprobamos que el volumen con filesystem `/dev/vdb` y 1GB de almacenamiento se ha montado bajo el directorio `/mydisk` y ya es utilizable.

15.3.20 Crear backup de un volumen

Para crear un backup de un volumen que está siendo usado actualmente usamos la opción `--force` del comando `openstack volume backup create`:

```
'#' openstack volume backup create --name vol1backup1 --force vol1

+-------+--------------------------------------+
| Field | Value                                |
+-------+--------------------------------------+
| id    | d6638c79-7f32-4c88-bf65-c200f71c5279 |
| name  | vol1backup1                          |
+-------+--------------------------------------+
```

Podemos ver detalles del backup creado:

```
'#' openstack volume backup show vol1backup1

+-----------------------+--------------------------------------+
| Field                 | Value                                |
+-----------------------+--------------------------------------+
| availability_zone     | nova                                 |
| container             | volumebackups                        |
| created_at            | 2020-02-11T07:16:26.000000           |
| data_timestamp        | 2020-02-11T07:16:26.000000           |
| description           | None                                 |
| fail_reason           | None                                 |
| has_dependent_backups | False                                |
| id                    | d6638c79-7f32-4c88-bf65-c200f71c5279 |
| is_incremental        | False                                |
| name                  | vol1backup1                          |
| object_count          | 22                                   |
| size                  | 1                                    |
| snapshot_id           | None                                 |
| status                | available                            |
| updated_at            | 2020-02-11T07:17:00.000000           |
| volume_id             | 325a4a42-d640-43d8-affe-8d5efacee6a4 |
+-----------------------+--------------------------------------+
```

Note: Los bakcups son guardados en archivos, por lo que se trata de objetos. Los backups se almacenan en Swift object storage, por defecto.

15.3.21 Crear snapshot de un volumen

Para crear un snapshot de un volumen conectado a una instancia usamos la opción `--force` del comando. Seleccionamos cuál es el volumen al cual se le creará el snapshot con la opción `--volume`. Nombramos al snapshot `vol1snap1`:

```
# Deprecated: openstack snapshot create --name snap1 --force vol1

'#' openstack volume snapshot create --volume vol1 --force vol1snap1

+-------------+--------------------------------------+
| Field       | Value                                |
+-------------+--------------------------------------+
| created_at  | 2020-02-11T08:53:24.358788           |
| description | None                                 |
| id          | 9cbba900-465d-4ac4-b03f-ebe4dea15213 |
| name        | vol1snap1                            |
| properties  |                                      |
| size        | 1                                    |
| status      | creating                             |
| updated_at  | None                                 |
| volume_id   | 325a4a42-d640-43d8-affe-8d5efacee6a4 |
+-------------+--------------------------------------+
```

Link del comando: volumen snapshot - Openstack Docs

15.3.22 Crear un contenedor

1. Listemos las cuentas, bajo la cual se crearán los objetos:

```
'#' openstack object store account show
+------------+----------------------------------------+
| Field      | Value                                  |
+------------+----------------------------------------+
| Account    | AUTH_99d8a6cd24734f2aa3fe70140fbdbd64 |
| Bytes      | 0                                      |
| Containers | 0                                      |
| Objects    | 0                                      |
+------------+----------------------------------------+
```

Las cuentas de Swift tienen contenedores y los contenedores tienen objetos dentro de sí.

2. Crear un contenedor:

```
'#' openstack container create container1

+---------------------------------------+------------+---------------------------------
↪----+
| account                               | container  | x-trans-id                      ⌙
↪    |
+---------------------------------------+------------+---------------------------------
↪----+
| AUTH_99d8a6cd24734f2aa3fe70140fbdbd64 | container1 | tx3da04526eaac4deea9d15-
↪005e427219 |
+---------------------------------------+------------+---------------------------------
↪----+
```

Como vemos, el contenedor se ha creado bajo la cuenta antes listada.

3. Listar contenedores:

```
'#' openstack container list

+------------+
| Name       |
+------------+
| container1 |
+------------+
```

15.3.23 Crear un objeto

Para crear un objeto en un contenedor debemos especificar el archivo que vamos a subir dentro del contenedor y el nombre de este contenedor.

Por ejemplo, para subir el archivo `keystonerc_admin` al contenedor `container1` creamos un objeto:

```
'#' openstack object create container1 keystonerc_admin

+------------------+------------+----------------------------------+
| object           | container  | etag                             |
+------------------+------------+----------------------------------+
| keystonerc_admin | container1 | 75f5a62d38e6dcb7ba8ef259e8f71727 |
+------------------+------------+----------------------------------+
```

15.3.24 Acceder a un objeto

Podemos acceder a un objeto almacenado con Swift de múltiples formas. Por ejemplo, mediante un navegador o desde la línea de comandos:

1. Correr el comando `swift tempurl`:

```
'#' swift tempurl

...
<key>               The secret temporary URL key set on the Swift cluster.
                    To set a key, run 'swift post -m
                    "Temp-URL-Key:b3968d0207b54ece87cccc06515a89d4"'
```

2. El comando `swift tempurl` nos dice que para configurar una llave URL temporal, primero debemos ejecutar:

```
'#' swift post -m "Temp-URL-Key:b3968d0207b54ece87cccc06515a89d4"
```

Se ha generado la llave en el background.

3. Obtener el ID de la cuenta para formar la URL del objeto:

```
'#' openstack object store account show

+------------+------------------------------------------------+
| Field      | Value                                          |
+------------+------------------------------------------------+
| Account    | AUTH_99d8a6cd24734f2aa3fe70140fbdbd64          |
| Bytes      | 373                                            |
| Containers | 1                                              |
| Objects    | 1                                              |
| properties | Temp-Url-Key='b3968d0207b54ece87cccc06515a89d4' |
+------------+------------------------------------------------+
```

4. Generar la URL temporal del objeto con el comando `tempurl`:

```
'#' swift tempurl get 1000 /v1/AUTH_99d8a6cd24734f2aa3fe70140fbdbd64/container1/
↪keystonerc_admin b3968d0207b54ece87cccc06515a89d4

/v1/AUTH_99d8a6cd24734f2aa3fe70140fbdbd64/container1/keystonerc_admin?temp_url_
↪sig=83258744f90d5dee3fc8cf04235754447c16f7b9&temp_url_expires=1581417017
```

En el comando hemos indicamos lo siguiente:

- deseamos acceder al link por el método `GET`
- el link debe estar activo por `1000` segundos
- la versión soportada por Swift es la `v1`
- el ID de la cuenta es `AUTH_99d8a6cd24734f2aa3fe70140fbdbd64`
- el objeto al que deseamos acceder es `keystonerc_admin`
- la llave del URL temporal es `b3968d0207b54ece87cccc06515a89d4`

5. Combinemos la `tempURL` con la información del hosts de Swift. Para obtener los datos restantes ejecutar:

```
'#' openstack endpoint list | grep swift | grep public

| 4bb287495f014c7b97f455784fc1e448 | RegionOne | swift        | object-store | True
↪ | public    | http://192.168.1.100:8080/v1/AUTH_%(tenant_id)s |

'#' openstack endpoint show 4bb287495f014c7b97f455784fc1e448

+--------------+-----------------------------------------------+
| Field        | Value                                         |
+--------------+-----------------------------------------------+
| enabled      | True                                          |
| id           | 4bb287495f014c7b97f455784fc1e448              |
| interface    | public                                        |
| region       | RegionOne                                     |
| region_id    | RegionOne                                     |
| service_id   | 43c335e18ef2446ca4171e16f82f0926              |
| service_name | swift                                         |
| service_type | object-store                                  |
| url          | http://192.168.1.100:8080/v1/AUTH_%(tenant_id)s |
+--------------+-----------------------------------------------+
```

Ahora sabemos más a detalle el formato de la URL (IP y puerto): `http://192.168.1.100:8080/v1/AUTH_%(tenant_id)s`

6. Formar la URL completa y acceder desde el terminal o el navegador a esta dirección para descargar el objeto:
   - Formato de URL: `http://192.168.1.100:8080/` + `tempurl`
   - URL: http://192.168.1.100:8080/v1/AUTH_99d8a6cd24734f2aa3fe70140fbdbd64/container1/keystonerc_admin?temp_url_sig=83258744f90d5dee3fc8cf04235754447c16f7b9&temp_url_expires=1581417017

15.3.25 Copiar llave pública al usuario `admin`

1. Copiar el contenido de la llave pública del keypair `keypair1`:

```
'#' openstack keypair show --public-key keypair1

ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAABAQCxmvWnlwsIFva/
↪0NUdPGI9FyMAQj9UOS36YILnbVF6Zv+SYBJMUJfuim68yEQbNh1U5OcP0oAP2DpSUzIkEVqJTLeJ6tZiL0jT34sDslCmFZd3Md
↪Oavso9lUr3nx9//
↪Caqc9YJiCghCAeF7M1Yp5RnaUz08jnibqJ8ayRxKlj9PXasLtRZcXPgqVySdokmZYvf8M6wXDq/
↪U8Z0pfHsiWczfCgdfKw7CJA8D+HsnDH0iX92rsD94wbir43WklbOR2lrtb8IYF+vzyqhkogzQiFsbKMVIqNSklZcs0BkE7iT7l
↪rVJz/ Generated-by-Nova
```

```
'#' cat keypair1.pub

ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAABAQCxmvWnlwsIFva/
↪0NUdPGI9FyMAQj9UOS36YILnbVF6Zv+SYBJMUJfuim68yEQbNh1U5OcP0oAP2DpSUzIkEVqJTLeJ6tZiL0jT34sDslCmFZd3Md
↪Oavso9lUr3nx9//
↪Caqc9YJiCghCAeF7M1Yp5RnaUz08jnibqJ8ayRxKlj9PXasLtRZcXPgqVySdokmZYvf8M6wXDq/
↪U8Z0pfHsiWczfCgdfKw7CJA8D+HsnDH0iX92rsD94wbir43WklbOR2lrtb8IYF+vzyqhkogzQiFsbKMVIqNSklZcs0BkE7iT7l
↪rVJz/ Generated-by-Nova
```

2. Cambiar de usuario a `admin` y crear el keypair:

```
'#' source keystonerc_admin

'#' openstack keypair create --public-key keypair1.pub keypair1

+-------------+-------------------------------------------------+
| Field       | Value                                           |
+-------------+-------------------------------------------------+
| fingerprint | 4a:ba:08:bc:b8:43:ec:8c:6c:9a:a9:6f:c0:f8:6f:3f |
| name        | keypair1                                        |
| user_id     | 913e33878304445a987b04f5ca4705a0                |
+-------------+-------------------------------------------------+
```

15.3.26 Crear Host Aggregates

```
'#' source keystonerc_admin

'#' openstack aggregate create --zone controller1_zone controller1_aggregate

'#' openstack aggregate add host controller1_aggregate controllernode1.localdomain

'#' openstack aggregate create --zone compute1_zone compute1_aggregate

'#' openstack aggregate create --zone compute2_zone compute2_aggregate

'#' openstack aggregate add host compute2_aggregate computenode2.localdomain
```

Note: Ahora que hemos creado host aggregates y los hemos asociado a availability zones, podemos crear instancias en availability zones determinados sin necesidad de tener permisos de administrador:

```
'#' source keystonerc_testuser1

'#' openstack server create --image cirros --flavor 10 --key-name project1_keypair1 --
↪security-group project1_secgroup --availability-zone compute1_zone --nic net-
↪id=e6760c1d-535b-4c0a-948d-5433e6d9fbec,v4-fixed-ip=10.10.10.203 instC

'#' openstack server create --image cirros --flavor 10 --key-name project1_keypair1 --
↪security-group project1_secgroup --availability-zone compute2_zone --nic net-
↪id=e6760c1d-535b-4c0a-948d-5433e6d9fbec,v4-fixed-ip=10.10.10.204 instD
```

CHAPTER 16

OpenStack Packstack Deployment

16.1 Despliegue de OpenStack con Packstack y VirtualBox

Table of Contents

Teoría de VirtualBox Bridged Networking:

- Bridged Networking - VirtualBox documentation
- How does bridged networking work in Virtualbox
- Neutron with existing external network

16.1.1 Prerequisitos (Controller node)

1. Parar servicios de `firewalld` y `NetworkManager`:

```
'#' systemctl stop firewalld NetworkManager
```

2. Deshabilitar servicios de `firewalld` y `NetworkManager`:

```
'#' systemctl disable firewalld NetworkManager
```

3. Habilitar e iniciar servicio `network`:

```
'#' systemctl enable network
'#' systemctl start network
```

4. Deshabilitar SE Linux

```
'#' cat /etc/selinux/config

SELINUX=disabled
SELINUXTYPE=targeted
```

5. Configuración de interfaz de red bridged:

- INTERFACE CONFIGURATION FILES - Red Hat Documentation
- BOOTPROTO=”none” o BOOTPROTO=”static”: configuración de IP estática (Difference between setting BOOTPROTO=none and BOOTPROTO=static in an ethernet interface configuration script in RHEL)
- IPADDR: Dirección IP dentro de la red local
- Gateway: Dirección IP del equipo con salida a la red exterior (Internet)

```
'#' cat /etc/sysconfig/network-scripts/ifcfg-enp0s3

TYPE="Ethernet"
PROXY_METHOD="none"
BROWSER_ONLY="no"
BOOTPROTO="none"
DEFROUTE="yes"
IPV4_FAILURE_FATAL="no"
IPV6INIT="no"
IPV6_AUTOCONF="yes"
IPV6_DEFROUTE="yes"
IPV6_FAILURE_FATAL="no"
IPV6_ADDR_GEN_MODE="stable-privacy"
NAME="enp0s3"
UUID="18ff926b-73ef-4f82-bb10-8481724746e0"
DEVICE="enp0s3"
ONBOOT="yes"
IPADDR="192.168.1.100"
PREFIX="24"
GATEWAY="192.168.1.1"
DNS1="8.8.8.8"
```

6. Configuración de interfaz de red aislada:

- IPADDR: Dirección IP dentro de la red aislada

```
'#' cat /etc/sysconfig/network-scripts/ifcfg-enp0s8

DEVICE="enp0s8"
TYPE="Ethernet"
BOOTPROTO="static"
IPADDR="10.10.10.100"
NETMASK=255.255.255.0
```

(continues on next page)

(continued from previous page)

```
IPV4_FAILURE_FATAL="no"
IPV6INIT="no"
DNS1="8.8.8.8"
ONBOOT="yes"
```

7. Configuración del hostname:

```
'#' cat /etc/sysconfig/network

HOSTNAME=controllernode1.localdomain

'#' cat /etc/hostname

controllernode1.localdomain

'#' hostname controllernode1.localdomain

'#' cat /etc/hosts

127.0.0.1   controllernode1 controllernode1.localdomain localhost4 localhost4.
↪localdomain4
::1         controllernode1 controllernode1.localdomain localhost6 localhost6.
↪localdomain6
```

8. Reiniciar el servicio de `network` y reiniciar el sistema:

```
'#' systemctl restart network
'#' reboot
```

9. Comprobar que SE Linux está desactivado:

```
'#' getenforce

Disabled
```

10. Clonar la VM Controller Node desde VirtualBox 2 veces. Una VM será el Compute Node 1 y la otra será el Compute Node 2. Luego, en VirtualBox para cada VM:

- Reconfigurar la memoria RAM y CPUs
- Randomizar las direcciones MAC de los 2 adaptadores

16.1.2 Configuración de Compute Node 1

1. Configuración de interfaz de red bridged:

```
'#' cat /etc/sysconfig/network-scripts/ifcfg-enp0s3

TYPE="Ethernet"
PROXY_METHOD="none"
BROWSER_ONLY="no"
BOOTPROTO="none"
DEFROUTE="yes"
IPV4_FAILURE_FATAL="no"
IPV6INIT="no"
IPV6_AUTOCONF="yes"
```

(continues on next page)

(continued from previous page)

```
IPV6_DEFROUTE="yes"
IPV6_FAILURE_FATAL="no"
IPV6_ADDR_GEN_MODE="stable-privacy"
NAME="enp0s3"
UUID="02b06617-95a0-4dfa-9293-16e756b2eccc"
DEVICE="enp0s3"
ONBOOT="yes"
IPADDR="192.168.1.101"
PREFIX="24"
GATEWAY="192.168.1.1"
DNS1="8.8.8.8"
```

2. Configuración de interfaz de red aislada:

```
'#' cat /etc/sysconfig/network-scripts/ifcfg-enp0s8

DEVICE="enp0s8"
TYPE="Ethernet"
BOOTPROTO="static"
IPADDR="10.10.10.101"
NETMASK=255.255.255.0
IPV4_FAILURE_FATAL="no"
IPV6INIT="no"
DNS1="8.8.8.8"
ONBOOT="yes"
```

3. Configuración del hostname:

```
'#' hostname computenode1.localdomain

'#' cat /etc/sysconfig/network

HOSTNAME=computenode1.localdomain

'#' cat /etc/hostname

computenode1.localdomain

'#' cat /etc/hosts

127.0.0.1   computenode1 computenode1.localdomain localhost4 localhost4.localdomain4
::1         computenode1 computenode1.localdomain localhost6 localhost6.localdomain6
```

4. Reiniciar el servicio de `network` y reiniciar el sistema:

```
'#' systemctl restart network
'#' reboot
```

16.1.3 Configuración de Compute Node 2

1. Configuración de interfaz de red bridged:

```
'#' cat /etc/sysconfig/network-scripts/ifcfg-enp0s3

TYPE="Ethernet"
```

(continues on next page)

(continued from previous page)

```
PROXY_METHOD="none"
BROWSER_ONLY="no"
BOOTPROTO="none"
DEFROUTE="yes"
IPV4_FAILURE_FATAL="no"
IPV6INIT="no"
IPV6_AUTOCONF="yes"
IPV6_DEFROUTE="yes"
IPV6_FAILURE_FATAL="no"
IPV6_ADDR_GEN_MODE="stable-privacy"
NAME="enp0s3"
UUID="12b06617-95a0-4dfa-9293-16e756b2eccd"
DEVICE="enp0s3"
ONBOOT="yes"
IPADDR="192.168.1.102"
PREFIX="24"
GATEWAY="192.168.1.1"
DNS1="8.8.8.8"
```

2. Configuración de interfaz de red aislada:

```
'#' cat /etc/sysconfig/network-scripts/ifcfg-enp0s8

DEVICE="enp0s8"
TYPE="Ethernet"
BOOTPROTO="static"
IPADDR="10.10.10.102"
NETMASK=255.255.255.0
IPV4_FAILURE_FATAL="no"
IPV6INIT="no"
DNS1="8.8.8.8"
ONBOOT="yes"
```

3. Configuración del hostname:

```
'#' hostname computenode2.localdomain

'#' cat /etc/sysconfig/network

HOSTNAME=computenode2.localdomain

'#' cat /etc/hostname

computenode2.localdomain

'#' cat /etc/hosts

127.0.0.1   computenode2 computenode2.localdomain localhost4 localhost4.localdomain4
::1         computenode2 computenode2.localdomain localhost6 localhost6.localdomain6
```

4. Reiniciar el servicio de `network` y reiniciar el sistema:

```
'#' systemctl restart network
'#' reboot
```

16.1.4 Descarga de paquetes (Controller node)

1. Descargar OpenStack:

- Para versiones de OpenStack nuevas (queens,rocky,stein):

```
'#' yum install -y centos-release-openstack-queens
```

- Para versiones de OpenStack antiguas (pike,ocata,newton,...):

```
'#' yum install -y https://repos.fedorapeople.org/repos/openstack/openstack-ocata/rdo-
↪release-ocata-0.noarch.rpm
```

2. Actualizar paquetes:

```
'#' yum update -y
```

3. Descargar Packstack:

```
'#' yum install -y openstack-packstack
```

16.1.5 Instalación de OpenStack con PackStack (Controller node)

```
'#' packstack --allinone --provision-demo=n --os-neutron-ovs-bridge-
↪mappings=extnet:br-ex --os-neutron-ml2-type-drivers=vxlan,flat
```

Cliente de Línea de comandos de Packstack:

- `--allinone`: equivalente a `--install-hosts=<local ipaddr> --novanetwork-pubif=<dev> --novacompute-privif=lo --novanetwork-privif=lo --os-swift-install=y`. Esta opción puede usarse para instalar un nodo all-in-one OpenStack en este host.
- `--provision-demo=n`: especificar *y* para aprovisionar el uso de demostración y pruebas [*y* , *n*]
- `--os-neutron-ovs-bridge-mappings=extnet:br-ex`: Lista separada por comas de mapeos del bridge para el plugin OpenStack Networking Open vSwitch. Cada tupla en la lista debe estar en el formato `<physical_network>:<ovs_bridge>`.
- `--os-neutron-ovs-bridge-interfaces=br-ex:enp3s0`: Lista separada por comas de pares Open vSwitch `<bridge>:<interface>`. La interfaz será añadida al bridge asociado.
- `--os-neutron-ml2-type-drivers=vxlan,flat`: añade los tipos de red flat y vxlan a los tipos soportados por la instalación.

16.1.6 Configuración post-instalación (Controller node)

Interfaces/OVS-bridges creados luego de la instalación de Packstack:

- `ovs-system`: interfaz creada al instalar OVS (no es un OVS bridge)
- `br-ex`: External bridge. Usado para conectarse a la red física externa. Tiene agregado el puerto de la VM con salida a la red externa (enp0s3).
- `br-int`: Integration bridge. Bridge del tráfico entre instancias, el túnel y bridges externos.
- `br-tun`: Tunnel bridge. Para crear túneles VXLAN o GRE entre nodos.

1. Configurar a la interfaz física `enp0s3` como un puerto en el OVS `br-ex`. Haremos a la interfaz física `enp3s0` solo un puerto capa 2 en el bridge y le daremos la IP al bridge mismo:

```
'#' cat /etc/sysconfig/network-scripts/ifcfg-enp0s3

TYPE=OVSPort
NAME=enp0s3
DEVICE=enp0s3
ONBOOT=yes
DEVICETYPE=ovs
OVS_BRIDGE=br-ex
```

2. Configurar el bridge externo `br-ex`:

```
'#' cat /etc/sysconfig/network-scripts/ifcfg-br-ex

DEVICE=br-ex
DEVICETYPE=ovs
TYPE=OVSBridge
BOOTPROTO=static
IPADDR=192.168.1.100
NETMASK=255.255.255.0
GATEWAY=192.168.1.1
IPV4_FAILURE_FATAL=no
IPV6INIT=no
DNS1=8.8.8.8
ONBOOT=yes
```

3. Reiniciar el servicio `network` para hacer efectivo los cambios:

```
'#' systemctl restart network
```

4. Crear la red externa con Neutron:

- Primero usar el archivo Keystone RC con las credenciales admin de OpenStack:

```
'#' pwd
/root

'#' ls
anaconda-ks.cfg  keystonerc_admin  packstack-answers-20190920-224100.   txt

'#' source keystonerc_admin
```

- Crear la red externa `ext-net` con Neutron:

```
# Con Neutron (deprecated): neutron net-create external_network --provider:network_
↪type flat --provider:physical_network extnet --router:external
'#' openstack network create external_network --provider-network-type flat --provider-
↪physical-network extnet --external

+---------------------------+--------------------------------------+
| Field                     | Value                                |
+---------------------------+--------------------------------------+
| admin_state_up            | UP                                   |
| availability_zone_hints   |                                      |
| availability_zones        |                                      |
| created_at                | 2020-02-10T18:48:05Z                 |
```

(continues on next page)

(continued from previous page)

```
| description               |                                      |
| dns_domain                | None                                 |
| id                        | f5dad5c1-bba9-41c5-844f-bd19a6a124aa |
| ipv4_address_scope        | None                                 |
| ipv6_address_scope        | None                                 |
| is_default                | False                                |
| is_vlan_transparent       | None                                 |
| mtu                       | 1500                                 |
| name                      | external_network                     |
| port_security_enabled     | True                                 |
| project_id                | 0c2bc29526f4465c95b8eaefcfae7b7c     |
| provider:network_type     | flat                                 |
| provider:physical_network | extnet                               |
| provider:segmentation_id  | None                                 |
| qos_policy_id             | None                                 |
| revision_number           | 5                                    |
| router:external           | External                             |
| segments                  | None                                 |
| shared                    | False                                |
| status                    | ACTIVE                               |
| subnets                   |                                      |
| tags                      |                                      |
| updated_at                | 2020-02-10T18:48:05Z                 |
+---------------------------+--------------------------------------+
```

5. Crear una subred pública con un rango de asignación fuera del rango DHCP de nuestra red física y configurar el gateway de nuestra red como el gateway de la red externa:

```
# Con Neutron (deprecated): neutron subnet-create --name public_subnet --enable_
↪dhcp=False --allocation-pool start=192.168.1.150,end=192.168.1.200 --gateway=192.
↪168.1.1 external_network 192.168.1.0/24
'#' openstack subnet create --network external_network --allocation-pool start=192.
↪168.1.150,end=192.168.1.200 --gateway=192.168.1.1 --subnet-range 192.168.1.0/24 --
↪no-dhcp public_subnet

+-------------------+--------------------------------------+
| Field             | Value                                |
+-------------------+--------------------------------------+
| allocation_pools  | 192.168.1.150-192.168.1.200          |
| cidr              | 192.168.1.0/24                       |
| created_at        | 2020-02-10T18:48:35Z                 |
| description       |                                      |
| dns_nameservers   |                                      |
| enable_dhcp       | False                                |
| gateway_ip        | 192.168.1.1                          |
| host_routes       |                                      |
| id                | a6ae14ab-2287-4d0d-b8eb-0f503792f32c |
| ip_version        | 4                                    |
| ipv6_address_mode | None                                 |
| ipv6_ra_mode      | None                                 |
| name              | public_subnet                        |
| network_id        | f5dad5c1-bba9-41c5-844f-bd19a6a124aa |
| prefix_length     | None                                 |
| project_id        | 0c2bc29526f4465c95b8eaefcfae7b7c     |
| revision_number   | 0                                    |
| segment_id        | None                                 |
```

(continues on next page)

(continued from previous page)

```
| service_types     |                                      |
| subnetpool_id     | None                                 |
| tags              |                                      |
| updated_at        | 2020-02-10T18:48:35Z                 |
+-------------------+--------------------------------------+
```

Podemos asignar IPs públicas a nuestras intancias desde este rango de asignación.

16.1.7 Instalación de Compute Nodes (Controller Node)

1. Realizar un backup del answer file generado automáticamente por Packstack cuando se realizó la instalación all-in-node de OpenStack:

```
'#' cp packstack-answers-20200124-103757.txt packstack-answers-20200124-103757.txt.
↪backup
```

2. Editar el archivo answer file para que los túnesles sean creados desde la segunda interfaz añadida a las VMs CentOS:

```
# Interface for the Open vSwitch tunnel. Packstack overrides the IP
# address used for tunnels on this hypervisor to the IP found on the
# specified interface (for example, eth1).
CONFIG_NEUTRON_OVS_TUNNEL_IF=enp0s8
```

3. Seguir editando el archivo y escribir la subred de nuestra segunda NIC, de forma que Packstack haga las modificaciones para usar esta red para túneles.

```
# Comma-separated list of subnets (for example,
# 192.168.10.0/24,192.168.11.0/24) used for sending tunneling packets.
# This is used to configure IP filtering to accept tunneling packets
# from these subnets instead of specific IP addresses of peer nodes.
# This is useful when you add existing nodes to EXCLUDE_SERVERS
# because, in this case, packstack cannot modify the IP filtering of
# the existing nodes.
CONFIG_NEUTRON_OVS_TUNNEL_SUBNETS=10.10.10.0/24
```

4. Configurar la direcciones IP de los nodos que tendrán el servicio de Compute. En este caso los 3 nodos (Controller Node, Compute Node 1 y 2):

```
# List the servers on which to install the Compute service.
CONFIG_COMPUTE_HOSTS=192.168.1.100,192.168.1.101,192.168.1.102
```

5. Añadir la IP del Controller Node para que su configuración no sea alterada, es decir, excluir este nodo en el proceso de instalación:

```
# Comma-separated list of servers to be excluded from the
# installation. This is helpful if you are running Packstack a second
# time with the same answer file and do not want Packstack to
# overwrite these server's configurations. Leave empty if you do not
# need to exclude any servers.
EXCLUDE_SERVERS=192.168.1.100
```

6. Editar las reglas de `iptables` para permitir que los Compute Nodes puedan acceder a los servicios de mensaje AMQP y base de datos MariaDB:

```
'#' vi /etc/sysconfig/iptables

...
-A INPUT -s 192.168.1.100/32 -p tcp -m multiport --dports 5671,5672 -m comment --
↪comment "001 amqp incoming amqp_192.168.1.100" -j ACCEPT
-A INPUT -s 192.168.1.101/32 -p tcp -m multiport --dports 5671,5672 -m comment --
↪comment "001 amqp incoming amqp_192.168.1.101" -j ACCEPT
-A INPUT -s 192.168.1.102/32 -p tcp -m multiport --dports 5671,5672 -m comment --
↪comment "001 amqp incoming amqp_192.168.1.102" -j ACCEPT
...
-A INPUT -s 192.168.1.100/32 -p tcp -m multiport --dports 3306 -m comment --comment
↪"001 mariadb incoming mariadb_192.168.1.100" -j ACCEPT
-A INPUT -s 192.168.1.101/32 -p tcp -m multiport --dports 3306 -m comment --comment
↪"001 mariadb incoming mariadb_192.168.1.101" -j ACCEPT
-A INPUT -s 192.168.1.102/32 -p tcp -m multiport --dports 3306 -m comment --comment
↪"001 mariadb incoming mariadb_192.168.1.102" -j ACCEPT
...
```

7. Reiniciar servicio de `iptables`:

```
'#' systemctl restart network
'#' systemctl restart iptables
```

8. Comenzar con la instalación de Packstack en los Compute Nodes usando el answer file editado:

```
'#' packstack --answer-file=packstack-answers-20200124-103757.txt
```

9. Desde el Controller Node, listar los hypervisors:

```
'#' openstack hypervisor list

+----+------------------------------+-----------------+---------------+-------+
| ID | Hypervisor Hostname          | Hypervisor Type | Host IP       | State |
+----+------------------------------+-----------------+---------------+-------+
|  1 | controllernode1.localdomain | QEMU            | 192.168.1.100 | up    |
|  2 | computenode2.localdomain    | QEMU            | 192.168.1.102 | up    |
|  3 | computenode1.localdomain    | QEMU            | 192.168.1.101 | up    |
+----+------------------------------+-----------------+---------------+-------+
```

16.2 Despliegue de OpenStack Singlenode con Packstack, VirtualBox y Vagrant

Table of Contents

16.2.1 Prerequisitos del sistema

Desde nuestro sistema Operativo (Windows, macOS, Linux) debemos tener instaladas los siguientes programas:

- VirtualBox
- Vagrant

Este escenario de OpenStack que será desplegado consta de 1 VM: un node que contiene las funcionalidades de controller, network, compute, etc. integradas. Se requiere al menos 10 GB de RAM y 4 CPUs.

Important: Si existen errores en la creación de las instancias o que las instancias creadas se apagan automáticamete, esto podría deberse a la falta de recursos en el host.

Revisar la memoria RAM disponible con `free -m` y el uso de CPU con `top` o `htop`. En caso los recursos se están usando al máximo, apagar la VM y asignarle más recursos desde el host en VirtualBox.

16.2.2 Configuración de VirtualBox

Host-only Network Adapter

1. Crear un Host-only Network Adapter haciendo clic en el botón *Global Tools*, *Host Network Manager*. Clic en el botón *Create*:
2. Se creará un adaptador con un nombre predefinido al cual podremos editar la dirección IPv4 respectiva. En este caso no será necesario activar la opción de DHCP:

Note: Sobre el adaptador de red creado:

- El adaptador pertenece a la red de management

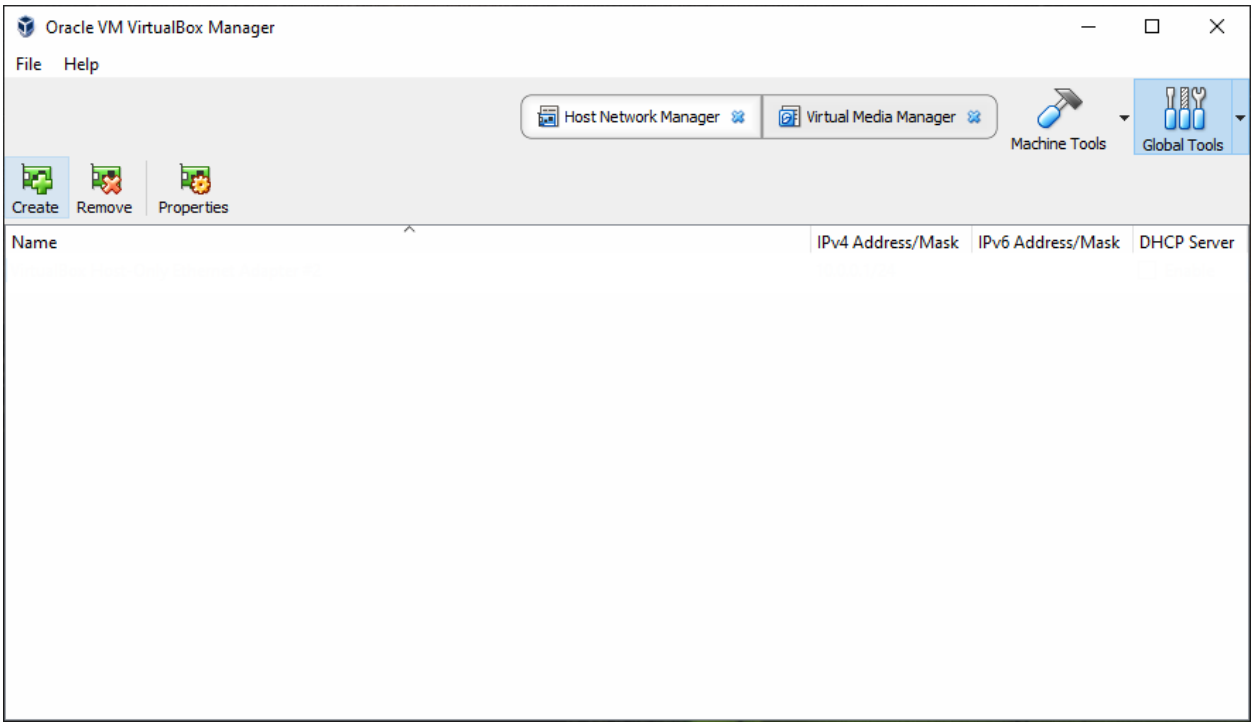


Fig. 1: Windows - VirtualBox v5.2 - Crear un nuevo adaptador

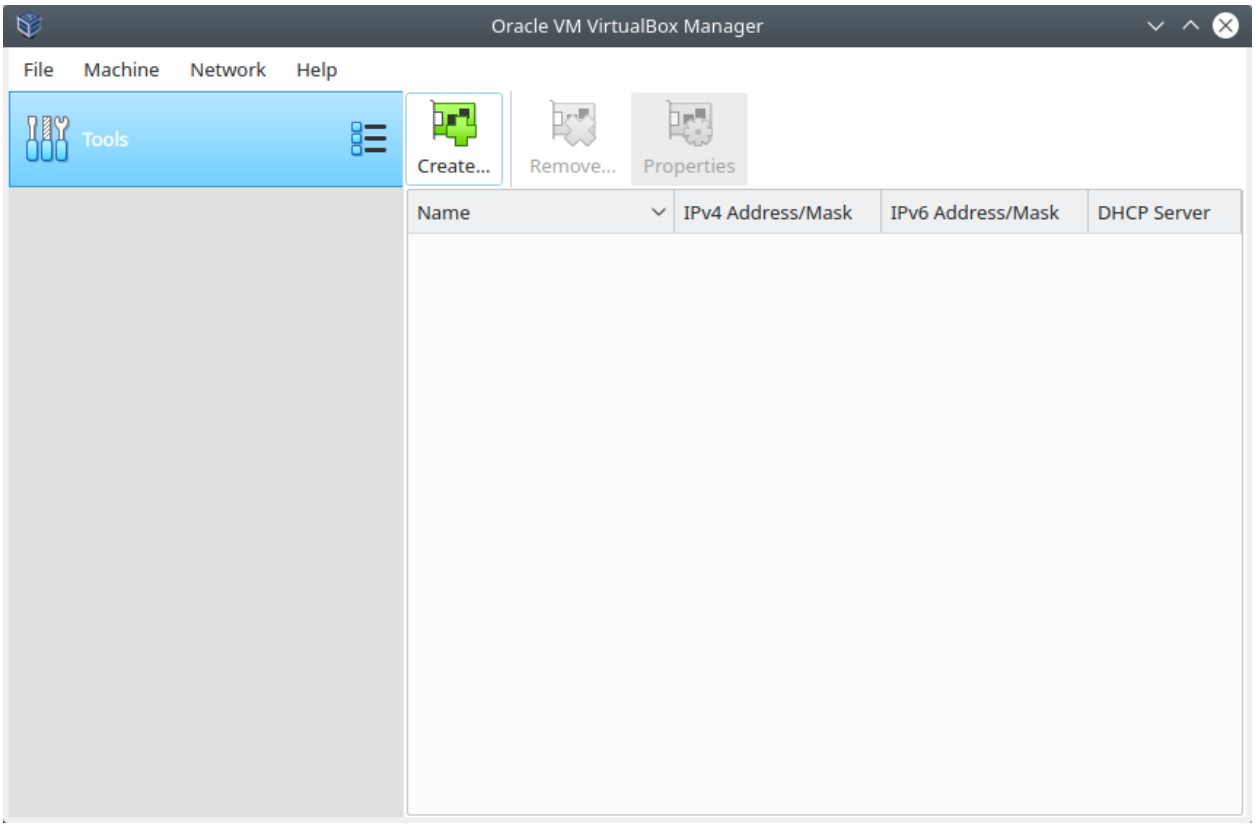


Fig. 2: Linux - VirtualBox v6.0 - Crear un nuevo adaptador

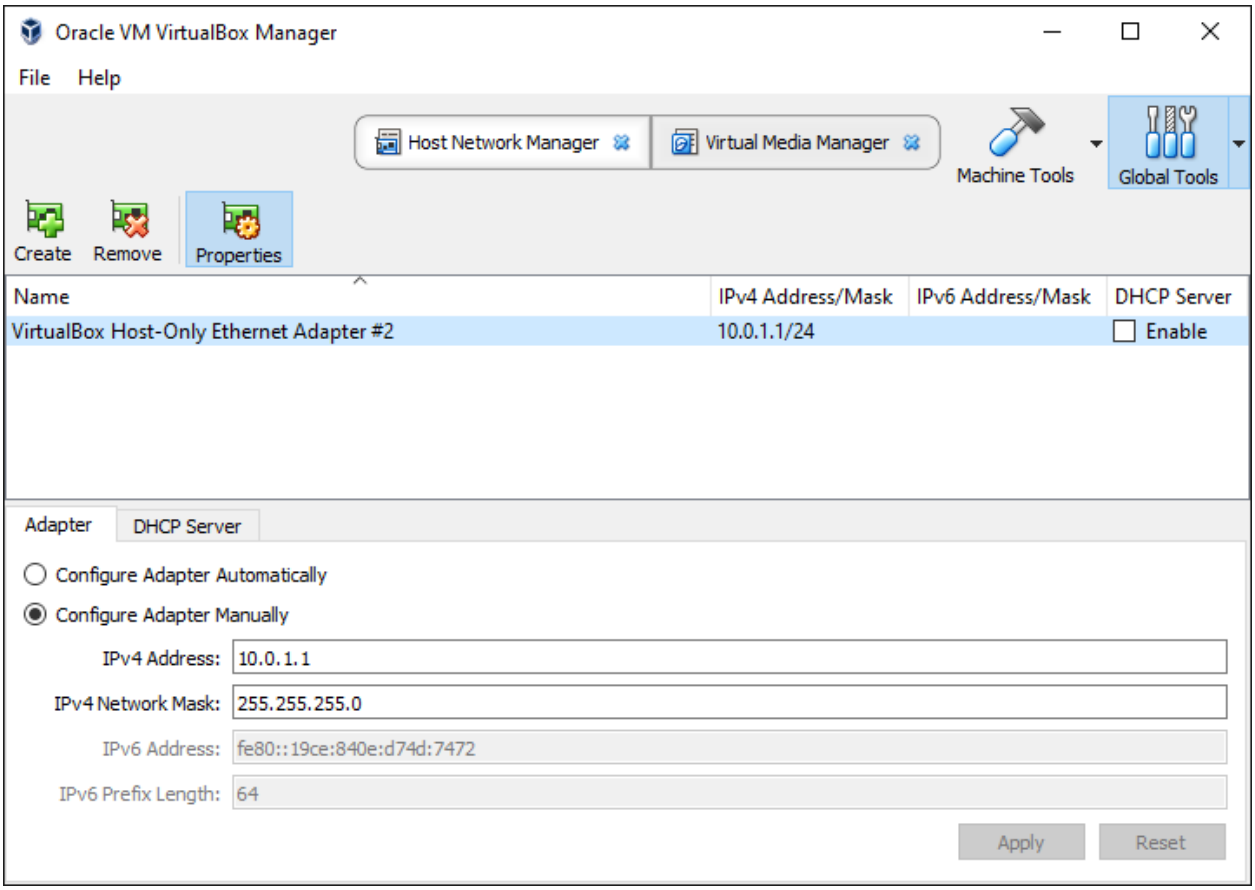


Fig. 3: Windows - VirtualBox v5.2 - Configuración del adaptador

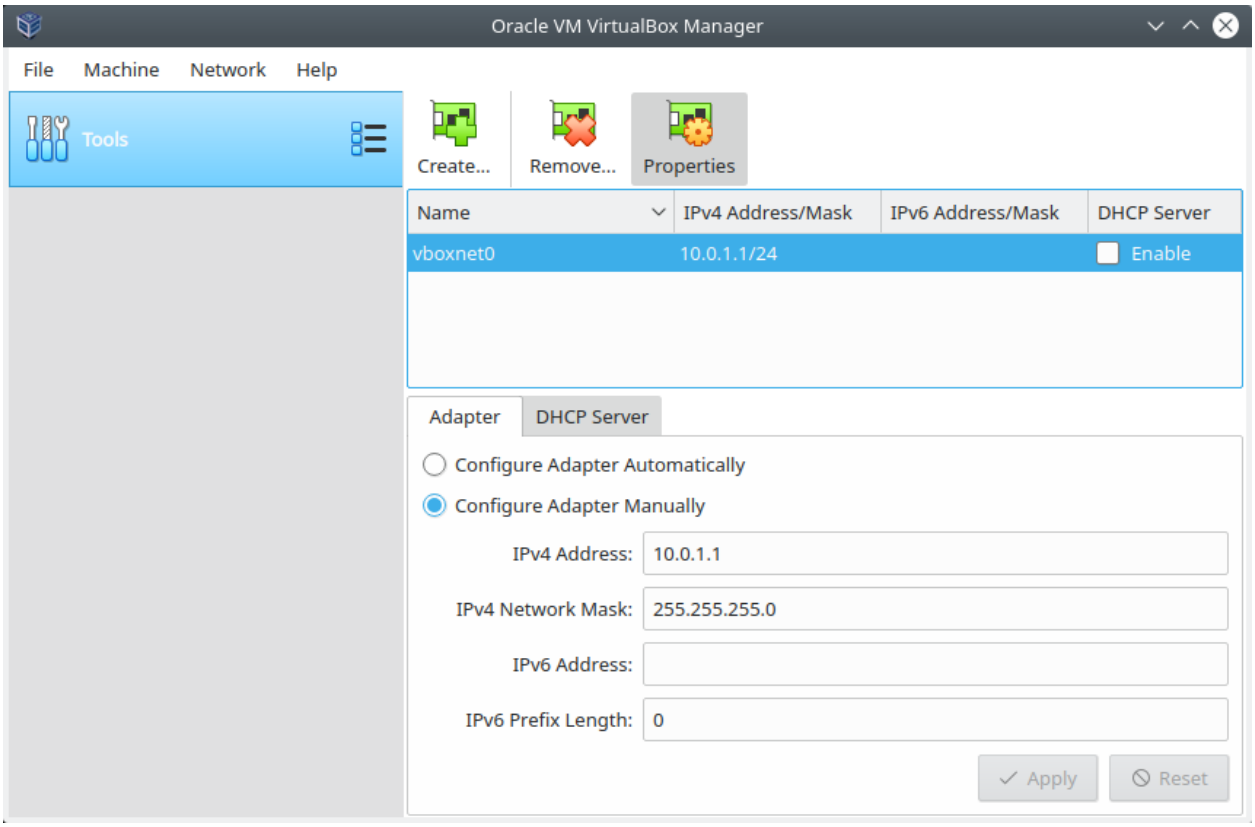


Fig. 4: Linux - VirtualBox v6.0 - Configuración del adaptador

- La dirección IP del adaptador es el Host-side de la red

16.2.3 Definición del archivo `Vagrantfile`

El archivo `Vagrantfile` se define de la siguiente forma:

```
# -*- mode: ruby -*-
# vi: set ft=ruby :
servers=[
  {
    :hostname => "packstack",
    :box => "centos/7",
    :ram => 10240,
    :cpu => 4,
    :script => "sh /vagrant/packstack_setup.sh"
  }
]
# All Vagrant configuration is done below. The "2" in Vagrant.configure
# configures the configuration version (we support older styles for
# backwards compatibility). Please don't change it unless you know what
# you're doing.
Vagrant.configure("2") do |config|
  servers.each do |machine|
    config.vm.define machine[:hostname] do |node|
      node.vm.box = machine[:box]
      node.vm.hostname = machine[:hostname]
      node.vm.provider "virtualbox" do |vb|
        vb.customize ["modifyvm", :id, "--memory", machine[:ram], "--cpus",␣
↪machine[:cpu]]
        vb.customize ["modifyvm", :id, "--nic2", "hostonly", "--hostonlyadapter2",
↪"VirtualBox Host-Only Ethernet Adapter #2"]
      end
      node.vm.provision "shell", inline: machine[:script], privileged: true, run:
↪"once"
    end
  end
end
```

Important:

- Vagrant configura automáticamente la primera interfaz de red de una nueva VM en la red NAT. A través de esta red podemos acceder desde el sistema host al Dashboard o CLI del nodo.
- La primera interfaz de red se usa para tener salida a Internet.
- La segunda interfaz sirve para proveer conectividad del sistema host a la VM (y viceversa).

16.2.4 Definición de archivo de configuración del nodo

En el mismo directorio donde tenemos almacenado el archivo `Vagrantfile` guardaremos el script `.sh` que se correrá cuando Vagrant lance la VM dentro del nodo:

Controller (`packstack_setup.sh`)

```
#! /bin/sh

export LANG=en_US.utf-8
export LC_ALL=en_US.utf-8

sed -i -e 's/enabled=1/enabled=0/g' /etc/yum/pluginconf.d/fastestmirror.conf

cat <<- EOF > /etc/sysconfig/network-scripts/ifcfg-eth1
DEVICE="eth1"
DEFROUTE="no"
BOOTPROTO="static"
IPADDR="10.0.1.20"
NETMASK="255.255.255.0"
DNS1="8.8.8.8"
TYPE="Ethernet"
ONBOOT=yes
EOF

ifdown eth1
ifup eth1

cat <<- EOF > /etc/hosts
127.0.0.1 localhost
10.0.1.20 packstack
EOF

echo 'centos' >/etc/yum/vars/contentdir

systemctl disable firewalld
systemctl stop firewalld
systemctl disable NetworkManager
systemctl stop NetworkManager
systemctl enable network
systemctl start network

yum install -y centos-release-openstack-queens
yum update -y
yum install -y openstack-packstack
yum install -y lvm2

packstack --install-hosts="10.0.1.20" --os-heat-install=y --os-heat-cfn-install=y --
↪os-neutron-lbaas-install=y --keystone-admin-passwd="openstack" --keystone-demo-
↪passwd="openstack"
```

16.2.5 Correr el entorno Vagrant e Instalación de OpenStack con Packstack

En el terminal, cambiar de directorio al lugar donde tenemos almacenado el archivo `Vagrantfile` y el script `.sh` del nodo. Luego, desplegar la máquina virtual con `vagrant up`:

```
$ cd singlenode-packstack-vagrant

$ vagrant up
```

Comenzará la configuración y despliegue de la máquina virtual en VirtualBox conforme se ha especificado en el

archivo `Vagrantfile`, luego correrá automáticamente el script `.sh` que se ha definido para la VM:

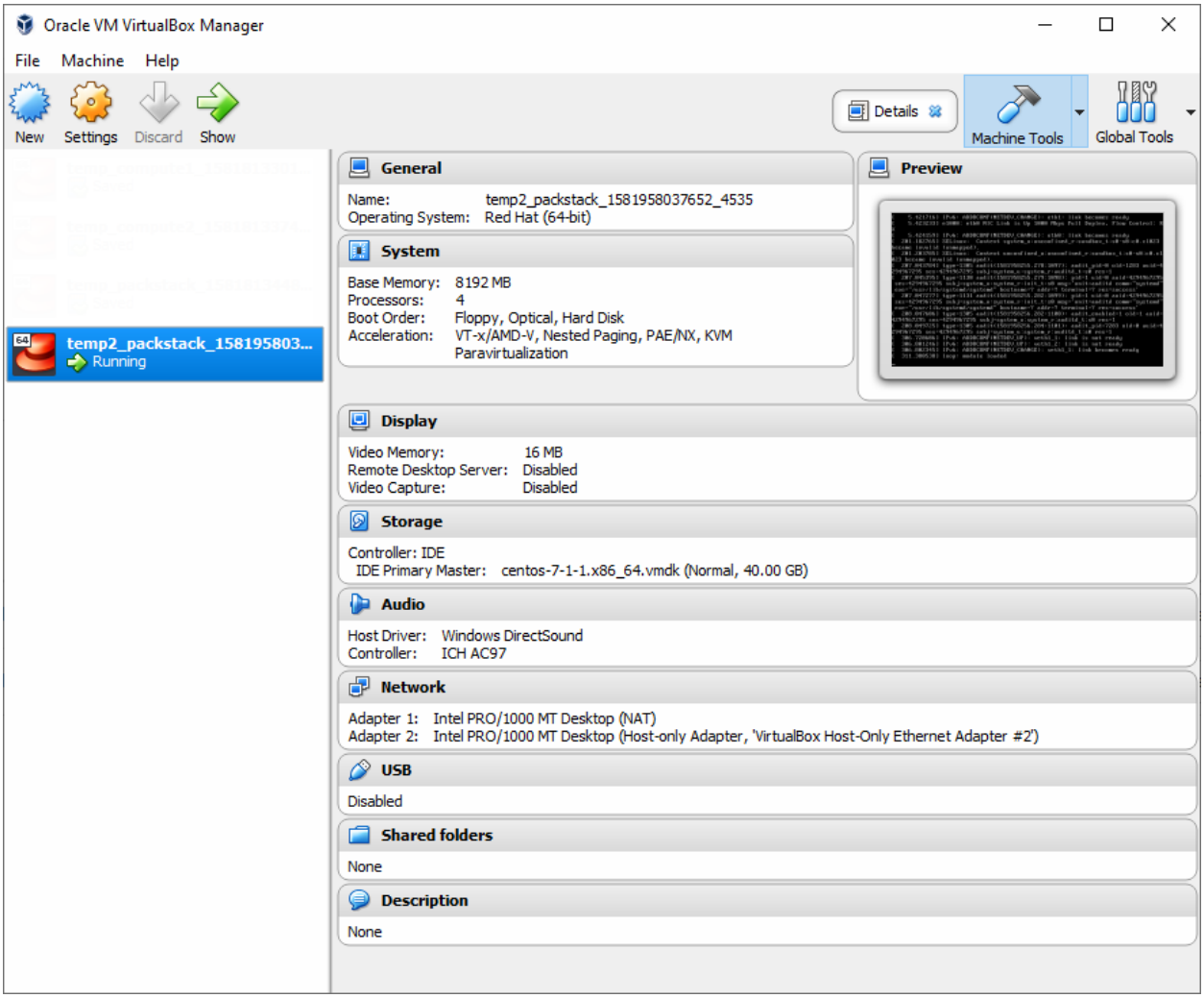


Fig. 5: VirtualBox - VM desplegada por Vagrant

La instalación de Packstack en este nodo iniciará automáticamente. Luego de media hora aproximadamente (32 minutos en total: despliegue de VMs + instalación de OpenStack), la instalación de OpenStack con Packstack habrá finalizado y podremos ingresar al Dashboard o al CLI de OpenStack.

Note: El Dashboard de OpenStack podría demorar un corto tiempo para arrancar la primera vez luego de haber instalado OpenStack.

16.2.6 Pruebas en el entorno OpenStack desplegado

Ubicándonos el mismo directorio donde tenemos el archivo `Vagrantfile` entraremos al terminal de la VM `packstack` con el siguiente comando:

```
$ vagrant ssh packstack
$ sudo su
```

(continues on next page)

(continued from previous page)

```
$ cd /root
```

Ahora realizaremos una prueba de despliegue desde el CLI:

Important: La imagen de CirrOS que se crea por defecto como `demo` al instalar Packstack tiene fallos pues tiene un tamaño reducido de 273 bytes. La causa de esto es que puede ser que se haya descargado sin la opción `-L` del comando `curl`. Por lo tanto, crearemos nuestra propia imagen CirrOS:

La topología que se desea lograr es la siguiente:

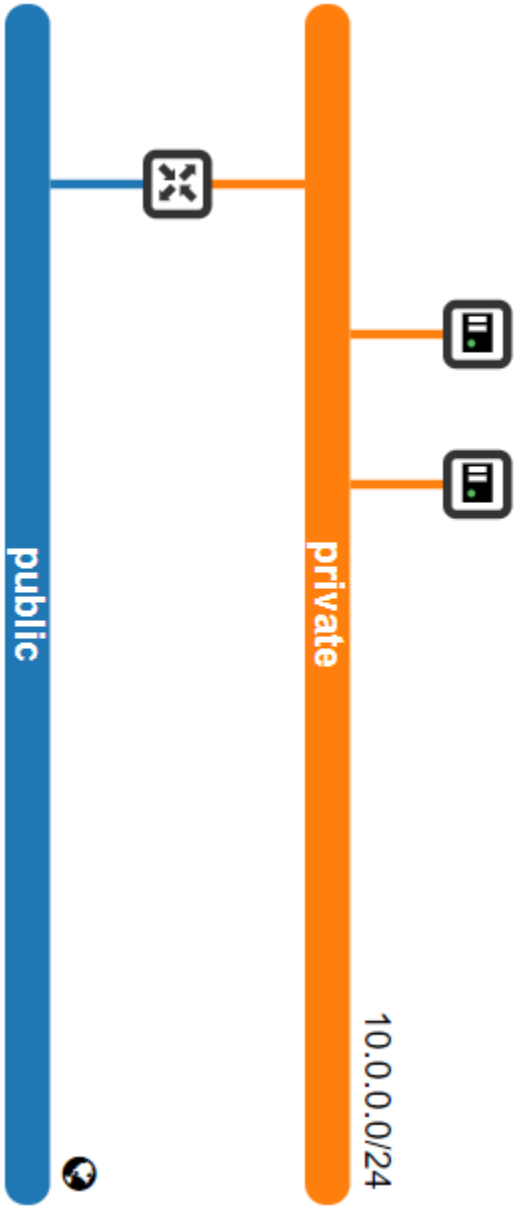


Fig. 6: OpenStack - Topología desplegada

Y los comandos que ejecutaremos en el nodo controller serán los siguientes:

```
'#' source keystonerc_admin

'#' mkdir images
'#' curl -o /root/images/cirros-0.4.0-x86_64-disk.img -L http://download.cirros-cloud.
↪net/0.4.0/cirros-0.4.0-x86_64-disk.img
'#' openstack image create --min-disk 1 --min-ram 128 --public --disk-format qcow2 --
↪file /root/images/cirros-0.4.0-x86_64-disk.img cirros1

'#' source keystonerc_demo

'#' openstack network list

+--------------------------------------+---------+--------------------------------------
↪--+
| ID                                   | Name    | Subnets                              ␣
↪   |
+--------------------------------------+---------+--------------------------------------
↪--+
| 0c2f47d2-d672-4aae-a83e-e5e40faf0197 | public  | 8a0934e6-6826-44b8-8f38-
↪dba27e301021 |
| 6665f7ee-4692-442b-a7dc-e38de59a32a3 | private | 4fccbd2e-ff40-4c75-acb3-
↪5781360af1b5 |
+--------------------------------------+---------+--------------------------------------
↪--+

'#' openstack server create --image cirros1 --flavor 1 --min 2 --max 2 --nic net-
↪id=6665f7ee-4692-442b-a7dc-e38de59a32a3 test
```

En este despliegue de prueba se han creado dos instancias al mismo tiempo y con las mismas características corriendo en el mismo nodo (o equivalentemente, en el mismo hypervisor).

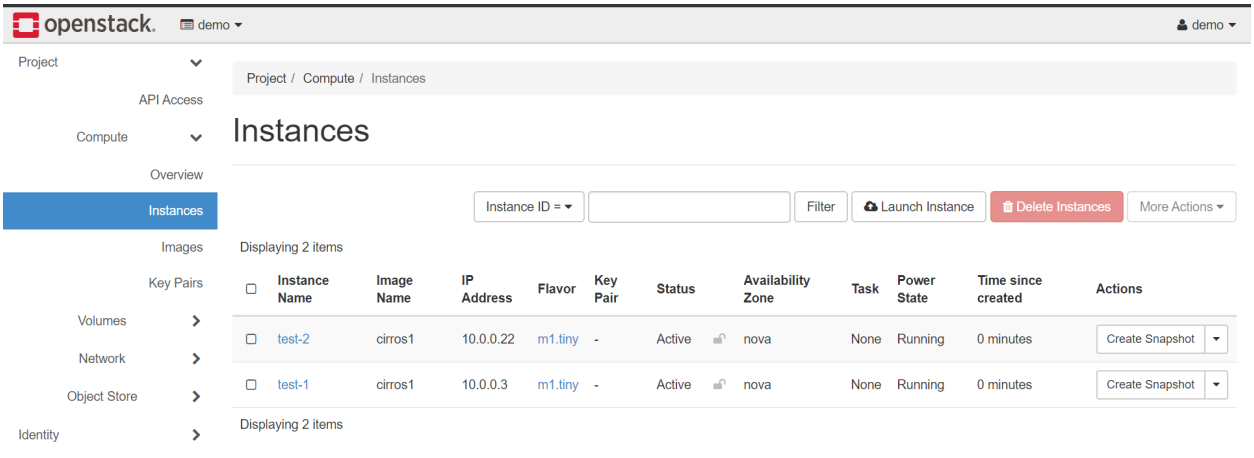


Fig. 7: OpenStack - Instancias desplegadas

Podremos ingresar a la consola de cada instancia desde el dashboard y probar conectividad entre ellas:

16.2.7 Anexo 1: Configuración de red desplegada

Host configuration (Windows user)

Las interfaces de red que tenemos en el sistema host de Windows son las siguientes:

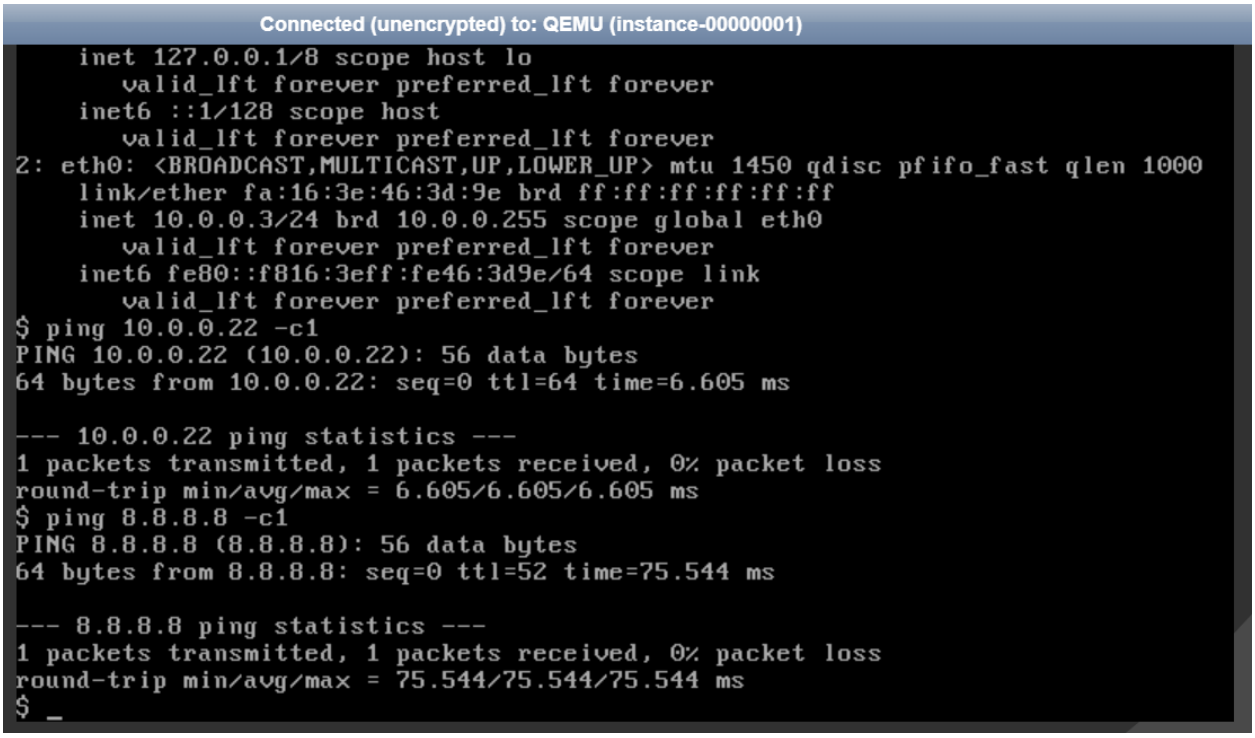


Fig. 8: OpenStack - Consola `test-1`

```
C:\Users\usuario>ipconfig

Configuración IP de Windows


Adaptador de Ethernet VirtualBox Host-Only Network #2:

Sufijo DNS específico para la conexión. . :
Dirección IPv4. . . . . . . . . . . . . . : 10.0.1.1
Máscara de subred . . . . . . . . . . . . : 255.255.255.0
Puerta de enlace predeterminada . . . . . :

Adaptador de Ethernet Ethernet:

Sufijo DNS específico para la conexión. . :
Dirección IPv4. . . . . . . . . . . . . . : 192.168.1.10
Máscara de subred . . . . . . . . . . . . : 255.255.255.0
Puerta de enlace predeterminada . . . . . : 192.168.1.1
```

- La interfaz `Adaptador de Ethernet Ethernet` es nuestra tarjeta de red física con salida a Internet a través de un router físico (`192.168.1.1`). Se le ha asignado la IP `192.168.1.10`.
- La interfaz `Adaptador de Ethernet VirtualBox Host-Only Network #2` ha sido creada con VirtualBox dentro de `Host Network Manager`. Se le ha asignado la IP `10.0.1.1`. Esta interfaz no cuenta con salida al exterior (Internet).

VM configuration (virtualBox config)

La VM creada como único nodo donde se instala OpenStack posee 2 interfaces que pueden verse desde el lado del SO host o del lado del SO guest, obteniendo información útil de su implementación:

Interfaces desde el lado host

La VM con sistema CentOS creada por VirtualBox posee dos interfaces conectadas virtualmente:

1. **Adaptador 1 - Interfaz tipo NAT**:

La primera interfaz de red es usada para que la VM tenga salida a Internet (en la VM se ve como la interfaz `enp0s3` o `eth0`):

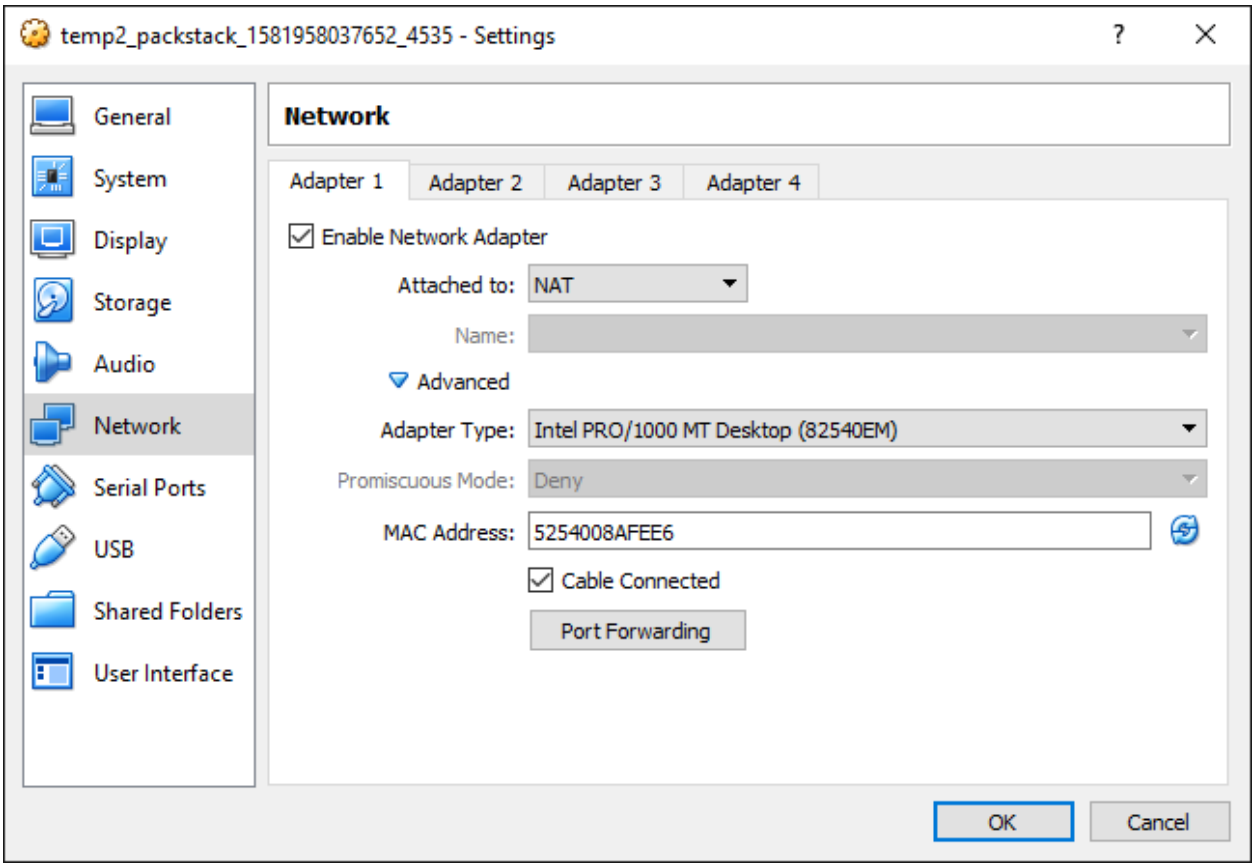


Fig. 9: VirtualBox - Adaptador 1

- La interfaz se encuentra conectada a una red virtual tipo NAT.
- En este caso la interfaz tiene asignada la IP `10.0.2.15`.

2. **Adaptador 2 - Interfaz tipo Host-only Adapter**:

La segunda interfaz de red es usada para que exista comunicación entre el sistema host y la VM (en la VM se ve como la interfaz `enp0s8` o `eth1`):

- La interfaz se encuentra conectada a la red física del host, siendo del tipo Host-only Adapter. Desde la VM podemos llegar a los equipos dentro de la red host (`192.168.1.0/24`).
- En este caso la interfaz tiene asignada la IP `10.0.1.20`.

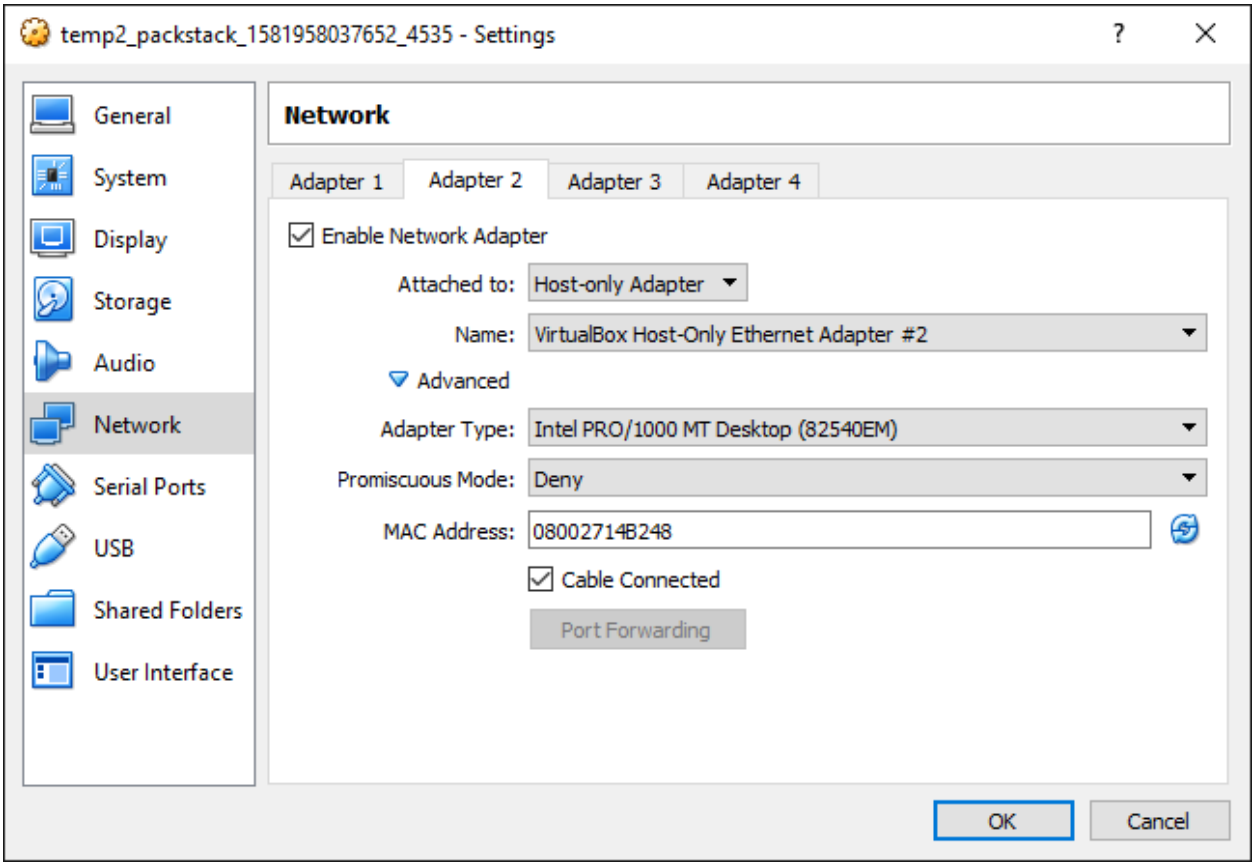


Fig. 10: VirtualBox - Adaptador 2

Interfaces desde el lado guest

Las IPs de cada interfaz han sido obtenidas desde la consola de la VM:

```
'#' ip addr

...

2: enp0s3: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP group␣
↪default qlen 1000
    link/ether 08:00:27:03:ad:05 brd ff:ff:ff:ff:ff:ff
    inet 10.0.2.15/24 brd 10.0.2.255 scope global noprefixroute dynamic enp0s3
       valid_lft 42944sec preferred_lft 42944sec
    inet6 fe80::a00:27ff:fe03:ad05/64 scope link noprefixroute
       valid_lft forever preferred_lft forever
3: enp0s8: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP group␣
↪default qlen 1000
    link/ether 08:00:27:3a:ab:74 brd ff:ff:ff:ff:ff:ff
    inet 10.0.1.20/24 brd 10.0.0.255 scope global noprefixroute enp0s8
       valid_lft forever preferred_lft forever
    inet6 fe80::a00:27ff:fe3a:ab74/64 scope link
       valid_lft forever preferred_lft forever

...
```

1. Adaptador 1 - Interfaz tipo NAT: `enp0s3` o `eth0`
2. Adaptador 2 - Interfaz tipo Host-only Adapter: `enp0s8` o `eth1`

VM configuration (OpenStack config)

- Interfaz dentro de la red pública de OpenStack:

La interfaz `br-ex` es en realidad un OvS con una interfaz interna que posee una IP dentro de la red pública:

```
'#' ip addr

...

5: br-ex: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc noqueue state UNKNOWN␣
↪group default qlen 1000
    link/ether 6a:f2:c8:bc:42:4e brd ff:ff:ff:ff:ff:ff
    inet 172.24.4.1/24 scope global br-ex
       valid_lft forever preferred_lft forever
    inet6 fe80::68f2:c8ff:febc:424e/64 scope link
       valid_lft forever preferred_lft forever

...
```

En este caso, la red pública es `172.24.4.0/24` y `br-ex` tiene la IP `172.24.4.1` asignada.

16.2.8 Anexo 2: Gráficos de red desplegada

Red general total (Host + VM + Tenant Networks)

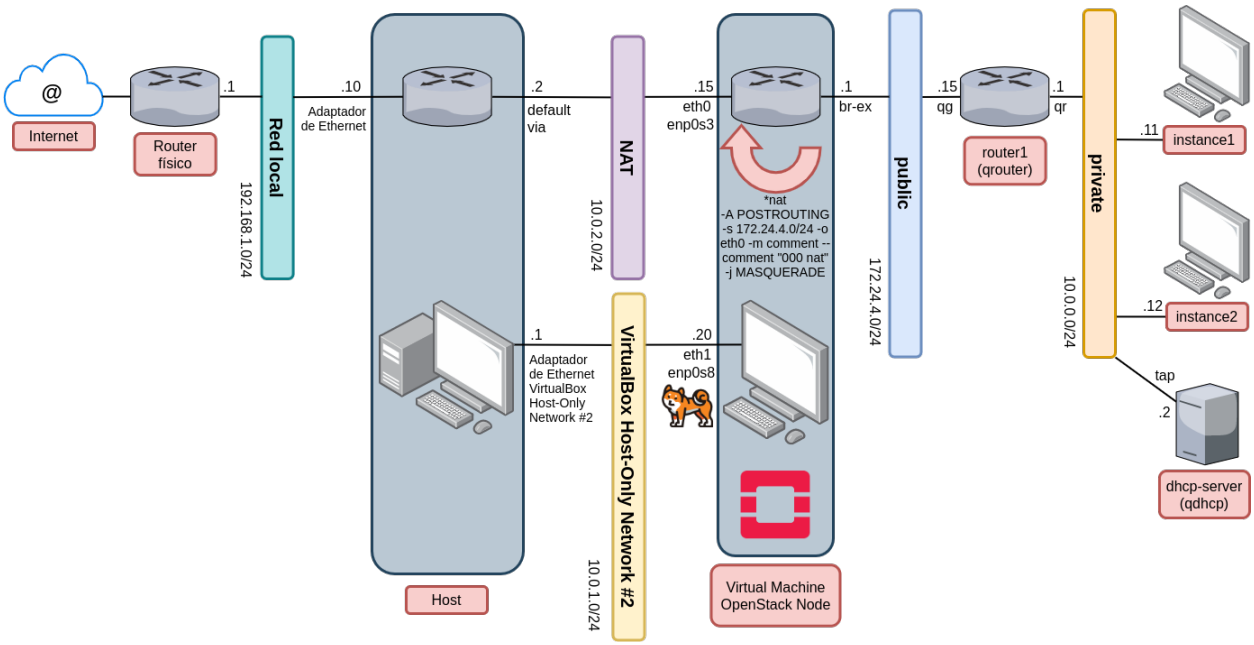


Fig. 11: OpenStack: Host + VM + Tenant Networks

Red e interfaces internas (VM + Tenant Networks)

16.2.9 Anexo 3: Interfaces de red, OvS, bridges

Se presentan los bridges, OvS e interfaces de red de la VM con el despliegue de OpenStack, solo con los elementos de demo instalados y 2 instancias lanzadas manualmente:

```
[root@packstack ~]'#' ip address

1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default␣
↪qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
       valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host
       valid_lft forever preferred_lft forever
2: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP group␣
↪default qlen 1000
    link/ether 52:54:00:8a:fe:e6 brd ff:ff:ff:ff:ff:ff
    inet 10.0.2.15/24 brd 10.0.2.255 scope global noprefixroute dynamic eth0
       valid_lft 80714sec preferred_lft 80714sec
    inet6 fe80::5054:ff:fe8a:fee6/64 scope link
       valid_lft forever preferred_lft forever
3: eth1: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP group␣
↪default qlen 1000
    link/ether 08:00:27:d3:26:2e brd ff:ff:ff:ff:ff:ff
    inet 10.0.1.20/24 brd 10.0.1.255 scope global noprefixroute eth1
       valid_lft forever preferred_lft forever
    inet6 fe80::a00:27ff:fed3:262e/64 scope link
       valid_lft forever preferred_lft forever
4: ovs-system: <BROADCAST,MULTICAST> mtu 1500 qdisc noop state DOWN group default␣
↪qlen 1000
```

(continues on next page)

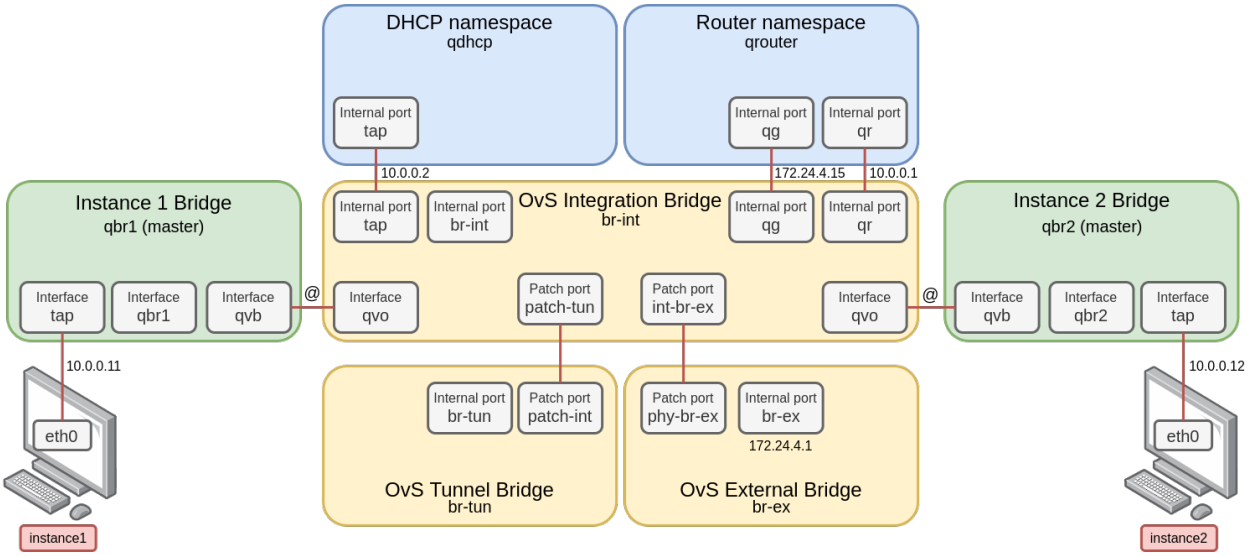


Fig. 12: OpenStack: Internal VM + Tenant Networks

(continued from previous page)

```
    link/ether 92:7f:d2:a2:0b:89 brd ff:ff:ff:ff:ff:ff
5: br-ex: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc noqueue state UNKNOWN␣
↪group default qlen 1000
    link/ether 6a:f2:c8:bc:42:4e brd ff:ff:ff:ff:ff:ff
    inet 172.24.4.1/24 scope global br-ex
       valid_lft forever preferred_lft forever
    inet6 fe80::68f2:c8ff:febc:424e/64 scope link
       valid_lft forever preferred_lft forever
6: br-int: <BROADCAST,MULTICAST> mtu 1450 qdisc noop state DOWN group default qlen␣
↪1000
    link/ether 1a:26:e1:53:59:47 brd ff:ff:ff:ff:ff:ff
7: br-tun: <BROADCAST,MULTICAST> mtu 1500 qdisc noop state DOWN group default qlen␣
↪1000
    link/ether 12:f8:4a:f2:1d:42 brd ff:ff:ff:ff:ff:ff
11: qbrbc1cf336-e1: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1450 qdisc noqueue state UP␣
↪group default qlen 1000
    link/ether fe:16:3e:51:be:1b brd ff:ff:ff:ff:ff:ff
12: qbr0ed97bfb-fa: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1450 qdisc noqueue state UP␣
↪group default qlen 1000
    link/ether ca:06:e8:28:78:a6 brd ff:ff:ff:ff:ff:ff
13: qvobc1cf336-e1@qvbbc1cf336-e1: <BROADCAST,MULTICAST,PROMISC,UP,LOWER_UP> mtu 1450␣
↪qdisc noqueue master ovs-system state UP group default qlen 1000
    link/ether 7a:fb:e2:b2:12:4b brd ff:ff:ff:ff:ff:ff
    inet6 fe80::78fb:e2ff:feb2:124b/64 scope link
       valid_lft forever preferred_lft forever
14: qvbbc1cf336-e1@qvobc1cf336-e1: <BROADCAST,MULTICAST,PROMISC,UP,LOWER_UP> mtu 1450␣
↪qdisc noqueue master qbrbc1cf336-e1 state UP group default qlen 1000
    link/ether fe:74:14:d1:17:28 brd ff:ff:ff:ff:ff:ff
    inet6 fe80::fc74:14ff:fed1:1728/64 scope link
       valid_lft forever preferred_lft forever
15: qvo0ed97bfb-fa@qvb0ed97bfb-fa: <BROADCAST,MULTICAST,PROMISC,UP,LOWER_UP> mtu 1450␣
↪qdisc noqueue master ovs-system state UP group default qlen 1000
    link/ether da:3c:b2:98:fd:df brd ff:ff:ff:ff:ff:ff
    inet6 fe80::d83c:b2ff:fe98:fddf/64 scope link
```

(continues on next page)

(continued from previous page)

```
       valid_lft forever preferred_lft forever
16: qvb0ed97bfb-fa@qvo0ed97bfb-fa: <BROADCAST,MULTICAST,PROMISC,UP,LOWER_UP> mtu 1450␣
↪qdisc noqueue master qbr0ed97bfb-fa state UP group default qlen 1000
    link/ether ca:06:e8:28:78:a6 brd ff:ff:ff:ff:ff:ff
    inet6 fe80::c806:e8ff:fe28:78a6/64 scope link
       valid_lft forever preferred_lft forever
17: tap0ed97bfb-fa: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1450 qdisc pfifo_fast␣
↪master qbr0ed97bfb-fa state UNKNOWN group default qlen 1000
    link/ether fe:16:3e:b3:2d:11 brd ff:ff:ff:ff:ff:ff
    inet6 fe80::fc16:3eff:feb3:2d11/64 scope link
       valid_lft forever preferred_lft forever
18: tapbc1cf336-e1: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1450 qdisc pfifo_fast␣
↪master qbrbc1cf336-e1 state UNKNOWN group default qlen 1000
    link/ether fe:16:3e:51:be:1b brd ff:ff:ff:ff:ff:ff
    inet6 fe80::fc16:3eff:fe51:be1b/64 scope link
       valid_lft forever preferred_lft forever
```

```
[root@packstack ~]'#' ovs-vsctl show

d9b4a0db-3fc2-41b3-a089-e2fa9ddd0a3f
    Manager "ptcp:6640:127.0.0.1"
        is_connected: true
    Bridge br-int
        Controller "tcp:127.0.0.1:6633"
            is_connected: true
        fail_mode: secure
        Port patch-tun
            Interface patch-tun
                type: patch
                options: {peer=patch-int}
        Port "qvo0ed97bfb-fa"
            tag: 1
            Interface "qvo0ed97bfb-fa"
        Port br-int
            Interface br-int
                type: internal
        Port "qvobc1cf336-e1"
            tag: 1
            Interface "qvobc1cf336-e1"
        Port "qg-c38b2759-b4"
            tag: 2
            Interface "qg-c38b2759-b4"
                type: internal
        Port int-br-ex
            Interface int-br-ex
                type: patch
                options: {peer=phy-br-ex}
        Port "tape0a80b5f-e1"
            tag: 1
            Interface "tape0a80b5f-e1"
                type: internal
        Port "qr-480f039f-df"
            tag: 1
            Interface "qr-480f039f-df"
                type: internal
    Bridge br-ex
```

(continues on next page)

(continued from previous page)

```
        Controller "tcp:127.0.0.1:6633"
            is_connected: true
        fail_mode: secure
        Port br-ex
            Interface br-ex
                type: internal
        Port phy-br-ex
            Interface phy-br-ex
                type: patch
                options: {peer=int-br-ex}
    Bridge br-tun
        Controller "tcp:127.0.0.1:6633"
            is_connected: true
        fail_mode: secure
        Port patch-int
            Interface patch-int
                type: patch
                options: {peer=patch-tun}
        Port br-tun
            Interface br-tun
                type: internal
    ovs_version: "2.11.0"
```

```
[root@packstack ~]'#' brctl show

bridge name             bridge id               STP enabled     interfaces
qbr0ed97bfb-fa          8000.ca06e82878a6       no              qvb0ed97bfb-fa
                                                                tap0ed97bfb-fa

qbrbc1cf336-e1          8000.fe163e51be1b       no              qvbbc1cf336-e1
                                                                tapbc1cf336-e1
```

16.2.10 Anexo 4: Archivo answers.txt de packstack generado

```
'#' cat /home/vagrant/packstack-answers-20200217-234314.txt

[general]

# Path to a public key to install on servers. If a usable key has not
# been installed on the remote servers, the user is prompted for a
# password and this key is installed so the password will not be
# required again.
CONFIG_SSH_KEY=/root/.ssh/id_rsa.pub

# Default password to be used everywhere (overridden by passwords set
# for individual services or users).
CONFIG_DEFAULT_PASSWORD=

# The amount of service workers/threads to use for each service.
# Useful to tweak when you have memory constraints. Defaults to the
# amount of cores on the system.
CONFIG_SERVICE_WORKERS=%{::processorcount}

# Specify 'y' to install MariaDB. ['y', 'n']
```

(continues on next page)

(continued from previous page)

```
CONFIG_MARIADB_INSTALL=y

# Specify 'y' to install OpenStack Image Service (glance). ['y', 'n']
CONFIG_GLANCE_INSTALL=y

# Specify 'y' to install OpenStack Block Storage (cinder). ['y', 'n']
CONFIG_CINDER_INSTALL=y

# Specify 'y' to install OpenStack Shared File System (manila). ['y',
# 'n']
CONFIG_MANILA_INSTALL=n

# Specify 'y' to install OpenStack Compute (nova). ['y', 'n']
CONFIG_NOVA_INSTALL=y

# Specify 'y' to install OpenStack Networking (neutron) ['y']
CONFIG_NEUTRON_INSTALL=y

# Specify 'y' to install OpenStack Dashboard (horizon). ['y', 'n']
CONFIG_HORIZON_INSTALL=y

# Specify 'y' to install OpenStack Object Storage (swift). ['y', 'n']
CONFIG_SWIFT_INSTALL=y

# Specify 'y' to install OpenStack Metering (ceilometer). Note this
# will also automatically install gnocchi service and configures it as
# the metrics backend. ['y', 'n']
CONFIG_CEILOMETER_INSTALL=y

# Specify 'y' to install OpenStack Telemetry Alarming (Aodh). Note
# Aodh requires Ceilometer to be installed as well. ['y', 'n']
CONFIG_AODH_INSTALL=y

# Specify 'y' to install OpenStack Events Service (panko). ['y', 'n']
CONFIG_PANKO_INSTALL=n

# Specify 'y' to install OpenStack Data Processing (sahara). In case
# of sahara installation packstack also installs heat.['y', 'n']
CONFIG_SAHARA_INSTALL=n

# Specify 'y' to install OpenStack Orchestration (heat). ['y', 'n']
CONFIG_HEAT_INSTALL=y

# Specify 'y' to install OpenStack Container Infrastructure
# Management Service (magnum). ['y', 'n']
CONFIG_MAGNUM_INSTALL=n

# Specify 'y' to install OpenStack Database (trove) ['y', 'n']
CONFIG_TROVE_INSTALL=n

# Specify 'y' to install OpenStack Bare Metal Provisioning (ironic).
# ['y', 'n']
CONFIG_IRONIC_INSTALL=n

# Specify 'y' to install the OpenStack Client packages (command-line
# tools). An admin "rc" file will also be installed. ['y', 'n']
CONFIG_CLIENT_INSTALL=y
```

(continues on next page)

(continued from previous page)

```
# Comma-separated list of NTP servers. Leave plain if Packstack
# should not install ntpd on instances.
CONFIG_NTP_SERVERS=

# Comma-separated list of servers to be excluded from the
# installation. This is helpful if you are running Packstack a second
# time with the same answer file and do not want Packstack to
# overwrite these server's configurations. Leave empty if you do not
# need to exclude any servers.
EXCLUDE_SERVERS=

# Specify 'y' if you want to run OpenStack services in debug mode;
# otherwise, specify 'n'. ['y', 'n']
CONFIG_DEBUG_MODE=n

# Server on which to install OpenStack services specific to the
# controller role (for example, API servers or dashboard).
CONFIG_CONTROLLER_HOST=10.0.1.20

# List the servers on which to install the Compute service.
CONFIG_COMPUTE_HOSTS=10.0.1.20

# List of servers on which to install the network service such as
# Compute networking (nova network) or OpenStack Networking (neutron).
CONFIG_NETWORK_HOSTS=10.0.1.20

# Specify 'y' if you want to use VMware vCenter as hypervisor and
# storage; otherwise, specify 'n'. ['y', 'n']
CONFIG_VMWARE_BACKEND=n

# Specify 'y' if you want to use unsupported parameters. This should
# be used only if you know what you are doing. Issues caused by using
# unsupported options will not be fixed before the next major release.
# ['y', 'n']
CONFIG_UNSUPPORTED=n

# Specify 'y' if you want to use subnet addresses (in CIDR format)
# instead of interface names in following options:
# CONFIG_NEUTRON_OVS_BRIDGE_IFACES,
# CONFIG_NEUTRON_LB_INTERFACE_MAPPINGS, CONFIG_NEUTRON_OVS_TUNNEL_IF.
# This is useful for cases when interface names are not same on all
# installation hosts.
CONFIG_USE_SUBNETS=n

# IP address of the VMware vCenter server.
CONFIG_VCENTER_HOST=

# User name for VMware vCenter server authentication.
CONFIG_VCENTER_USER=

# Password for VMware vCenter server authentication.
CONFIG_VCENTER_PASSWORD=

# Comma separated list of names of the VMware vCenter clusters. Note:
# if multiple clusters are specified each one is mapped to one
# compute, otherwise all computes are mapped to same cluster.
```

(continues on next page)

(continued from previous page)

```
CONFIG_VCENTER_CLUSTER_NAMES=

# (Unsupported!) Server on which to install OpenStack services
# specific to storage servers such as Image or Block Storage services.
CONFIG_STORAGE_HOST=10.0.2.15

# (Unsupported!) Server on which to install OpenStack services
# specific to OpenStack Data Processing (sahara).
CONFIG_SAHARA_HOST=10.0.2.15

# Comma-separated list of URLs for any additional yum repositories,
# to use for installation.
CONFIG_REPO=

# Specify 'y' to enable the RDO testing repository. ['y', 'n']
CONFIG_ENABLE_RDO_TESTING=n

# To subscribe each server with Red Hat Subscription Manager, include
# this with CONFIG_RH_PW.
CONFIG_RH_USER=

# To subscribe each server to receive updates from a Satellite
# server, provide the URL of the Satellite server. You must also
# provide a user name (CONFIG_SATELLITE_USERNAME) and password
# (CONFIG_SATELLITE_PASSWORD) or an access key (CONFIG_SATELLITE_AKEY)
# for authentication.
CONFIG_SATELLITE_URL=

# Specify a Satellite 6 Server to register to. If not specified,
# Packstack will register the system to the Red Hat server. When this
# option is specified, you also need to set the Satellite 6
# organization (CONFIG_RH_SAT6_ORG) and an activation key
# (CONFIG_RH_SAT6_KEY).
CONFIG_RH_SAT6_SERVER=

# To subscribe each server with Red Hat Subscription Manager, include
# this with CONFIG_RH_USER.
CONFIG_RH_PW=

# Specify 'y' to enable RHEL optional repositories. ['y', 'n']
CONFIG_RH_OPTIONAL=y

# HTTP proxy to use with Red Hat Subscription Manager.
CONFIG_RH_PROXY=

# Specify a Satellite 6 Server organization to use when registering
# the system.
CONFIG_RH_SAT6_ORG=

# Specify a Satellite 6 Server activation key to use when registering
# the system.
CONFIG_RH_SAT6_KEY=

# Port to use for Red Hat Subscription Manager's HTTP proxy.
CONFIG_RH_PROXY_PORT=

# User name to use for Red Hat Subscription Manager's HTTP proxy.
```

(continues on next page)

(continued from previous page)

```
CONFIG_RH_PROXY_USER=

# Password to use for Red Hat Subscription Manager's HTTP proxy.
CONFIG_RH_PROXY_PW=

# User name to authenticate with the RHN Satellite server; if you
# intend to use an access key for Satellite authentication, leave this
# blank.
CONFIG_SATELLITE_USER=

# Password to authenticate with the RHN Satellite server; if you
# intend to use an access key for Satellite authentication, leave this
# blank.
CONFIG_SATELLITE_PW=

# Access key for the Satellite server; if you intend to use a user
# name and password for Satellite authentication, leave this blank.
CONFIG_SATELLITE_AKEY=

# Certificate path or URL of the certificate authority to verify that
# the connection with the Satellite server is secure. If you are not
# using Satellite in your deployment, leave this blank.
CONFIG_SATELLITE_CACERT=

# Profile name that should be used as an identifier for the system in
# RHN Satellite (if required).
CONFIG_SATELLITE_PROFILE=

# Comma-separated list of flags passed to the rhnreg_ks command.
# Valid flags are: novirtinfo, norhnsd, nopackages ['novirtinfo',
# 'norhnsd', 'nopackages']
CONFIG_SATELLITE_FLAGS=

# HTTP proxy to use when connecting to the RHN Satellite server (if
# required).
CONFIG_SATELLITE_PROXY=

# User name to authenticate with the Satellite-server HTTP proxy.
CONFIG_SATELLITE_PROXY_USER=

# User password to authenticate with the Satellite-server HTTP proxy.
CONFIG_SATELLITE_PROXY_PW=

# Specify filepath for CA cert file. If CONFIG_SSL_CACERT_SELFSIGN is
# set to 'n' it has to be preexisting file.
CONFIG_SSL_CACERT_FILE=/etc/pki/tls/certs/selfcert.crt

# Specify filepath for CA cert key file. If
# CONFIG_SSL_CACERT_SELFSIGN is set to 'n' it has to be preexisting
# file.
CONFIG_SSL_CACERT_KEY_FILE=/etc/pki/tls/private/selfkey.key

# Enter the path to use to store generated SSL certificates in.
CONFIG_SSL_CERT_DIR=~/packstackca/

# Specify 'y' if you want Packstack to pregenerate the CA
# Certificate.
```

(continues on next page)

(continued from previous page)

```
CONFIG_SSL_CACERT_SELFSIGN=y

# Enter the ssl certificates subject country.
CONFIG_SSL_CERT_SUBJECT_C=--

# Enter the ssl certificates subject state.
CONFIG_SSL_CERT_SUBJECT_ST=State

# Enter the ssl certificates subject location.
CONFIG_SSL_CERT_SUBJECT_L=City

# Enter the ssl certificates subject organization.
CONFIG_SSL_CERT_SUBJECT_O=openstack

# Enter the ssl certificates subject organizational unit.
CONFIG_SSL_CERT_SUBJECT_OU=packstack

# Enter the ssl certificates subject common name.
CONFIG_SSL_CERT_SUBJECT_CN=packstack

CONFIG_SSL_CERT_SUBJECT_MAIL=admin@packstack

# Service to be used as the AMQP broker. Allowed values are: rabbitmq
# ['rabbitmq']
CONFIG_AMQP_BACKEND=rabbitmq

# IP address of the server on which to install the AMQP service.
CONFIG_AMQP_HOST=10.0.1.20

# Specify 'y' to enable SSL for the AMQP service. ['y', 'n']
CONFIG_AMQP_ENABLE_SSL=n

# Specify 'y' to enable authentication for the AMQP service. ['y',
# 'n']
CONFIG_AMQP_ENABLE_AUTH=n

# Password for the NSS certificate database of the AMQP service.
CONFIG_AMQP_NSS_CERTDB_PW=PW_PLACEHOLDER

# User for AMQP authentication.
CONFIG_AMQP_AUTH_USER=amqp_user

# Password for AMQP authentication.
CONFIG_AMQP_AUTH_PASSWORD=PW_PLACEHOLDER

# IP address of the server on which to install MariaDB. If a MariaDB
# installation was not specified in CONFIG_MARIADB_INSTALL, specify
# the IP address of an existing database server (a MariaDB cluster can
# also be specified).
CONFIG_MARIADB_HOST=10.0.1.20

# User name for the MariaDB administrative user.
CONFIG_MARIADB_USER=root

# Password for the MariaDB administrative user.
CONFIG_MARIADB_PW=438b090ac8954b38
```

(continues on next page)

(continued from previous page)

```
# Password to use for the Identity service (keystone) to access the
# database.
CONFIG_KEYSTONE_DB_PW=2001d80f5ef94c48

# Enter y if cron job for removing soft deleted DB rows should be
# created.
CONFIG_KEYSTONE_DB_PURGE_ENABLE=True

# Default region name to use when creating tenants in the Identity
# service.
CONFIG_KEYSTONE_REGION=RegionOne

# Token to use for the Identity service API.
CONFIG_KEYSTONE_ADMIN_TOKEN=a862d06c55c143e2822ba5fbe3a19e8f

# Email address for the Identity service 'admin' user.  Defaults to
CONFIG_KEYSTONE_ADMIN_EMAIL=root@localhost

# User name for the Identity service 'admin' user.  Defaults to
# 'admin'.
CONFIG_KEYSTONE_ADMIN_USERNAME=admin

# Password to use for the Identity service 'admin' user.
CONFIG_KEYSTONE_ADMIN_PW=openstack

# Password to use for the Identity service 'demo' user.
CONFIG_KEYSTONE_DEMO_PW=openstack

# Identity service API version string. ['v2.0', 'v3']
CONFIG_KEYSTONE_API_VERSION=v3

# Identity service token format (UUID, PKI or FERNET). The
# recommended format for new deployments is FERNET. ['UUID', 'PKI',
# 'FERNET']
CONFIG_KEYSTONE_TOKEN_FORMAT=FERNET

# Type of Identity service backend (sql or ldap). ['sql', 'ldap']
CONFIG_KEYSTONE_IDENTITY_BACKEND=sql

# URL for the Identity service LDAP backend.
CONFIG_KEYSTONE_LDAP_URL=ldap://10.0.2.15

# User DN for the Identity service LDAP backend.  Used to bind to the
# LDAP server if the LDAP server does not allow anonymous
# authentication.
CONFIG_KEYSTONE_LDAP_USER_DN=

# User DN password for the Identity service LDAP backend.
CONFIG_KEYSTONE_LDAP_USER_PASSWORD=

# Base suffix for the Identity service LDAP backend.
CONFIG_KEYSTONE_LDAP_SUFFIX=

# Query scope for the Identity service LDAP backend. Use 'one' for
# onelevel/singleLevel or 'sub' for subtree/wholeSubtree ('base' is
# not actually used by the Identity service and is therefore
# deprecated). ['base', 'one', 'sub']
```

(continues on next page)

(continued from previous page)

```
CONFIG_KEYSTONE_LDAP_QUERY_SCOPE=one

# Query page size for the Identity service LDAP backend.
CONFIG_KEYSTONE_LDAP_PAGE_SIZE=-1

# User subtree for the Identity service LDAP backend.
CONFIG_KEYSTONE_LDAP_USER_SUBTREE=

# User query filter for the Identity service LDAP backend.
CONFIG_KEYSTONE_LDAP_USER_FILTER=

# User object class for the Identity service LDAP backend.
CONFIG_KEYSTONE_LDAP_USER_OBJECTCLASS=

# User ID attribute for the Identity service LDAP backend.
CONFIG_KEYSTONE_LDAP_USER_ID_ATTRIBUTE=

# User name attribute for the Identity service LDAP backend.
CONFIG_KEYSTONE_LDAP_USER_NAME_ATTRIBUTE=

# User email address attribute for the Identity service LDAP backend.
CONFIG_KEYSTONE_LDAP_USER_MAIL_ATTRIBUTE=

# User-enabled attribute for the Identity service LDAP backend.
CONFIG_KEYSTONE_LDAP_USER_ENABLED_ATTRIBUTE=

# Bit mask integer applied to user-enabled attribute for the Identity
# service LDAP backend. Indicate the bit that the enabled value is
# stored in if the LDAP server represents "enabled" as a bit on an
# integer rather than a boolean. A value of "0" indicates the mask is
# not used (default). If this is not set to "0", the typical value is
# "2", typically used when
# "CONFIG_KEYSTONE_LDAP_USER_ENABLED_ATTRIBUTE = userAccountControl".
CONFIG_KEYSTONE_LDAP_USER_ENABLED_MASK=-1

# Value of enabled attribute which indicates user is enabled for the
# Identity service LDAP backend. This should match an appropriate
# integer value if the LDAP server uses non-boolean (bitmask) values
# to indicate whether a user is enabled or disabled. If this is not
# set as 'y', the typical value is "512". This is typically used when
# "CONFIG_KEYSTONE_LDAP_USER_ENABLED_ATTRIBUTE = userAccountControl".
CONFIG_KEYSTONE_LDAP_USER_ENABLED_DEFAULT=TRUE

# Specify 'y' if users are disabled (not enabled) in the Identity
# service LDAP backend (inverts boolean-enalbed values).  Some LDAP
# servers use a boolean lock attribute where "y" means an account is
# disabled. Setting this to 'y' allows these lock attributes to be
# used. This setting will have no effect if
# "CONFIG_KEYSTONE_LDAP_USER_ENABLED_MASK" is in use. ['n', 'y']
CONFIG_KEYSTONE_LDAP_USER_ENABLED_INVERT=n

# Comma-separated list of attributes stripped from LDAP user entry
# upon update.
CONFIG_KEYSTONE_LDAP_USER_ATTRIBUTE_IGNORE=

# Identity service LDAP attribute mapped to default_project_id for
# users.
```

(continues on next page)

(continued from previous page)

```
CONFIG_KEYSTONE_LDAP_USER_DEFAULT_PROJECT_ID_ATTRIBUTE=

# Specify 'y' if you want to be able to create Identity service users
# through the Identity service interface; specify 'n' if you will
# create directly in the LDAP backend. ['n', 'y']
CONFIG_KEYSTONE_LDAP_USER_ALLOW_CREATE=n

# Specify 'y' if you want to be able to update Identity service users
# through the Identity service interface; specify 'n' if you will
# update directly in the LDAP backend. ['n', 'y']
CONFIG_KEYSTONE_LDAP_USER_ALLOW_UPDATE=n

# Specify 'y' if you want to be able to delete Identity service users
# through the Identity service interface; specify 'n' if you will
# delete directly in the LDAP backend. ['n', 'y']
CONFIG_KEYSTONE_LDAP_USER_ALLOW_DELETE=n

# Identity service LDAP attribute mapped to password.
CONFIG_KEYSTONE_LDAP_USER_PASS_ATTRIBUTE=

# DN of the group entry to hold enabled LDAP users when using enabled
# emulation.
CONFIG_KEYSTONE_LDAP_USER_ENABLED_EMULATION_DN=

# List of additional LDAP attributes for mapping additional attribute
# mappings for users. The attribute-mapping format is
# <ldap_attr>:<user_attr>, where ldap_attr is the attribute in the
# LDAP entry and user_attr is the Identity API attribute.
CONFIG_KEYSTONE_LDAP_USER_ADDITIONAL_ATTRIBUTE_MAPPING=

# Group subtree for the Identity service LDAP backend.
CONFIG_KEYSTONE_LDAP_GROUP_SUBTREE=

# Group query filter for the Identity service LDAP backend.
CONFIG_KEYSTONE_LDAP_GROUP_FILTER=

# Group object class for the Identity service LDAP backend.
CONFIG_KEYSTONE_LDAP_GROUP_OBJECTCLASS=

# Group ID attribute for the Identity service LDAP backend.
CONFIG_KEYSTONE_LDAP_GROUP_ID_ATTRIBUTE=

# Group name attribute for the Identity service LDAP backend.
CONFIG_KEYSTONE_LDAP_GROUP_NAME_ATTRIBUTE=

# Group member attribute for the Identity service LDAP backend.
CONFIG_KEYSTONE_LDAP_GROUP_MEMBER_ATTRIBUTE=

# Group description attribute for the Identity service LDAP backend.
CONFIG_KEYSTONE_LDAP_GROUP_DESC_ATTRIBUTE=

# Comma-separated list of attributes stripped from LDAP group entry
# upon update.
CONFIG_KEYSTONE_LDAP_GROUP_ATTRIBUTE_IGNORE=

# Specify 'y' if you want to be able to create Identity service
# groups through the Identity service interface; specify 'n' if you
```

(continues on next page)

(continued from previous page)

```
# will create directly in the LDAP backend. ['n', 'y']
CONFIG_KEYSTONE_LDAP_GROUP_ALLOW_CREATE=n

# Specify 'y' if you want to be able to update Identity service
# groups through the Identity service interface; specify 'n' if you
# will update directly in the LDAP backend. ['n', 'y']
CONFIG_KEYSTONE_LDAP_GROUP_ALLOW_UPDATE=n

# Specify 'y' if you want to be able to delete Identity service
# groups through the Identity service interface; specify 'n' if you
# will delete directly in the LDAP backend. ['n', 'y']
CONFIG_KEYSTONE_LDAP_GROUP_ALLOW_DELETE=n

# List of additional LDAP attributes used for mapping additional
# attribute mappings for groups. The attribute=mapping format is
# <ldap_attr>:<group_attr>, where ldap_attr is the attribute in the
# LDAP entry and group_attr is the Identity API attribute.
CONFIG_KEYSTONE_LDAP_GROUP_ADDITIONAL_ATTRIBUTE_MAPPING=

# Specify 'y' if the Identity service LDAP backend should use TLS.
# ['n', 'y']
CONFIG_KEYSTONE_LDAP_USE_TLS=n

# CA certificate directory for Identity service LDAP backend (if TLS
# is used).
CONFIG_KEYSTONE_LDAP_TLS_CACERTDIR=

# CA certificate file for Identity service LDAP backend (if TLS is
# used).
CONFIG_KEYSTONE_LDAP_TLS_CACERTFILE=

# Certificate-checking strictness level for Identity service LDAP
# backend; valid options are: never, allow, demand. ['never', 'allow',
# 'demand']
CONFIG_KEYSTONE_LDAP_TLS_REQ_CERT=demand

# Password to use for the Image service (glance) to access the
# database.
CONFIG_GLANCE_DB_PW=0d36ed19e654478e

# Password to use for the Image service to authenticate with the
# Identity service.
CONFIG_GLANCE_KS_PW=a7a58564762445b2

# Storage backend for the Image service (controls how the Image
# service stores disk images). Valid options are: file or swift
# (Object Storage). The Object Storage service must be enabled to use
# it as a working backend; otherwise, Packstack falls back to 'file'.
# ['file', 'swift']
CONFIG_GLANCE_BACKEND=file

# Password to use for the Block Storage service (cinder) to access
# the database.
CONFIG_CINDER_DB_PW=d5143e26f1bc4fa4

# Enter y if cron job for removing soft deleted DB rows should be
# created.
```

(continues on next page)

(continued from previous page)

```
CONFIG_CINDER_DB_PURGE_ENABLE=True

# Password to use for the Block Storage service to authenticate with
# the Identity service.
CONFIG_CINDER_KS_PW=24d21c3ac97042ab

# Storage backend to use for the Block Storage service; valid options
# are: lvm, gluster, nfs, vmdk, netapp, solidfire. ['lvm', 'gluster',
# 'nfs', 'vmdk', 'netapp', 'solidfire']
CONFIG_CINDER_BACKEND=lvm

# Specify 'y' to create the Block Storage volumes group. That is,
# Packstack creates a raw disk image in /var/lib/cinder, and mounts it
# using a loopback device. This should only be used for testing on a
# proof-of-concept installation of the Block Storage service (a file-
# backed volume group is not suitable for production usage). ['y',
# 'n']
CONFIG_CINDER_VOLUMES_CREATE=y

# Specify a custom name for the lvm cinder volume group
CONFIG_CINDER_VOLUME_NAME=cinder-volumes

# Size of Block Storage volumes group. Actual volume size will be
# extended with 3% more space for VG metadata. Remember that the size
# of the volume group will restrict the amount of disk space that you
# can expose to Compute instances, and that the specified amount must
# be available on the device used for /var/lib/cinder.
CONFIG_CINDER_VOLUMES_SIZE=20G

# A single or comma-separated list of Red Hat Storage (gluster)
# volume shares to mount. Example: 'ip-address:/vol-name', 'domain
# :/vol-name'
CONFIG_CINDER_GLUSTER_MOUNTS=

# A single or comma-separated list of NFS exports to mount. Example:
# 'ip-address:/export-name'
CONFIG_CINDER_NFS_MOUNTS=

# Administrative user account name used to access the NetApp storage
# system or proxy server.
CONFIG_CINDER_NETAPP_LOGIN=

# Password for the NetApp administrative user account specified in
# the CONFIG_CINDER_NETAPP_LOGIN parameter.
CONFIG_CINDER_NETAPP_PASSWORD=

# Hostname (or IP address) for the NetApp storage system or proxy
# server.
CONFIG_CINDER_NETAPP_HOSTNAME=

# The TCP port to use for communication with the storage system or
# proxy. If not specified, Data ONTAP drivers will use 80 for HTTP and
# 443 for HTTPS; E-Series will use 8080 for HTTP and 8443 for HTTPS.
# Defaults to 80.
CONFIG_CINDER_NETAPP_SERVER_PORT=80

# Storage family type used on the NetApp storage system; valid
```

(continues on next page)

(continued from previous page)

```
# options are ontap_7mode for using Data ONTAP operating in 7-Mode,
# ontap_cluster for using clustered Data ONTAP, or E-Series for NetApp
# E-Series. Defaults to ontap_cluster. ['ontap_7mode',
# 'ontap_cluster', 'eseries']
CONFIG_CINDER_NETAPP_STORAGE_FAMILY=ontap_cluster

# The transport protocol used when communicating with the NetApp
# storage system or proxy server. Valid values are http or https.
# Defaults to 'http'. ['http', 'https']
CONFIG_CINDER_NETAPP_TRANSPORT_TYPE=http

# Storage protocol to be used on the data path with the NetApp
# storage system; valid options are iscsi, fc, nfs. Defaults to nfs.
# ['iscsi', 'fc', 'nfs']
CONFIG_CINDER_NETAPP_STORAGE_PROTOCOL=nfs

# Quantity to be multiplied by the requested volume size to ensure
# enough space is available on the virtual storage server (Vserver) to
# fulfill the volume creation request.  Defaults to 1.0.
CONFIG_CINDER_NETAPP_SIZE_MULTIPLIER=1.0

# Time period (in minutes) that is allowed to elapse after the image
# is last accessed, before it is deleted from the NFS image cache.
# When a cache-cleaning cycle begins, images in the cache that have
# not been accessed in the last M minutes, where M is the value of
# this parameter, are deleted from the cache to create free space on
# the NFS share. Defaults to 720.
CONFIG_CINDER_NETAPP_EXPIRY_THRES_MINUTES=720

# If the percentage of available space for an NFS share has dropped
# below the value specified by this parameter, the NFS image cache is
# cleaned.  Defaults to 20.
CONFIG_CINDER_NETAPP_THRES_AVL_SIZE_PERC_START=20

# When the percentage of available space on an NFS share has reached
# the percentage specified by this parameter, the driver stops
# clearing files from the NFS image cache that have not been accessed
# in the last M minutes, where M is the value of the
# CONFIG_CINDER_NETAPP_EXPIRY_THRES_MINUTES parameter. Defaults to 60.
CONFIG_CINDER_NETAPP_THRES_AVL_SIZE_PERC_STOP=60

# Single or comma-separated list of NetApp NFS shares for Block
# Storage to use.  Format: ip-address:/export-name. Defaults to ''.
CONFIG_CINDER_NETAPP_NFS_SHARES=

# File with the list of available NFS shares.   Defaults to
# '/etc/cinder/shares.conf'.
CONFIG_CINDER_NETAPP_NFS_SHARES_CONFIG=/etc/cinder/shares.conf

# This parameter is only utilized when the storage protocol is
# configured to use iSCSI or FC. This parameter is used to restrict
# provisioning to the specified controller volumes. Specify the value
# of this parameter to be a comma separated list of NetApp controller
# volume names to be used for provisioning. Defaults to ''.
CONFIG_CINDER_NETAPP_VOLUME_LIST=

# The vFiler unit on which provisioning of block storage volumes will
```

(continues on next page)

(continued from previous page)

```
# be done. This parameter is only used by the driver when connecting
# to an instance with a storage family of Data ONTAP operating in
# 7-Mode Only use this parameter when utilizing the MultiStore feature
# on the NetApp storage system. Defaults to ''.
CONFIG_CINDER_NETAPP_VFILER=

# The name of the config.conf stanza for a Data ONTAP (7-mode) HA
# partner.  This option is only used by the driver when connecting to
# an instance with a storage family of Data ONTAP operating in 7-Mode,
# and it is required if the storage protocol selected is FC. Defaults
# to ''.
CONFIG_CINDER_NETAPP_PARTNER_BACKEND_NAME=

# This option specifies the virtual storage server (Vserver) name on
# the storage cluster on which provisioning of block storage volumes
# should occur. Defaults to ''.
CONFIG_CINDER_NETAPP_VSERVER=

# Restricts provisioning to the specified controllers. Value must be
# a comma-separated list of controller hostnames or IP addresses to be
# used for provisioning. This option is only utilized when the storage
# family is configured to use E-Series. Defaults to ''.
CONFIG_CINDER_NETAPP_CONTROLLER_IPS=

# Password for the NetApp E-Series storage array. Defaults to ''.
CONFIG_CINDER_NETAPP_SA_PASSWORD=

# This option is used to define how the controllers in the E-Series
# storage array will work with the particular operating system on the
# hosts that are connected to it. Defaults to 'linux_dm_mp'
CONFIG_CINDER_NETAPP_ESERIES_HOST_TYPE=linux_dm_mp

# Path to the NetApp E-Series proxy application on a proxy server.
# The value is combined with the value of the
# CONFIG_CINDER_NETAPP_TRANSPORT_TYPE, CONFIG_CINDER_NETAPP_HOSTNAME,
# and CONFIG_CINDER_NETAPP_HOSTNAME options to create the URL used by
# the driver to connect to the proxy application. Defaults to
# '/devmgr/v2'.
CONFIG_CINDER_NETAPP_WEBSERVICE_PATH=/devmgr/v2

# Restricts provisioning to the specified storage pools. Only dynamic
# disk pools are currently supported. The value must be a comma-
# separated list of disk pool names to be used for provisioning.
# Defaults to ''.
CONFIG_CINDER_NETAPP_STORAGE_POOLS=

# Cluster admin account name used to access the SolidFire storage
# system.
CONFIG_CINDER_SOLIDFIRE_LOGIN=

# Password for the SolidFire cluster admin user account specified in
# the CONFIG_CINDER_SOLIDFIRE_LOGIN parameter.
CONFIG_CINDER_SOLIDFIRE_PASSWORD=

# Hostname (or IP address) for the SolidFire storage system's MVIP.
CONFIG_CINDER_SOLIDFIRE_HOSTNAME=
```

(continues on next page)

(continued from previous page)

```
# Password to use for OpenStack Bare Metal Provisioning (ironic) to
# access the database.
CONFIG_IRONIC_DB_PW=PW_PLACEHOLDER

# Password to use for OpenStack Bare Metal Provisioning to
# authenticate with the Identity service.
CONFIG_IRONIC_KS_PW=PW_PLACEHOLDER

# Enter y if cron job for removing soft deleted DB rows should be
# created.
CONFIG_NOVA_DB_PURGE_ENABLE=True

# Password to use for the Compute service (nova) to access the
# database.
CONFIG_NOVA_DB_PW=d6fc9a80f919466b

# Password to use for the Compute service to authenticate with the
# Identity service.
CONFIG_NOVA_KS_PW=e4773a61b8684f64

# Whether or not Packstack should manage a default initial set of
# Nova flavors. Defaults to 'y'.
CONFIG_NOVA_MANAGE_FLAVORS=y

# Overcommitment ratio for virtual to physical CPUs. Specify 1.0 to
# disable CPU overcommitment.
CONFIG_NOVA_SCHED_CPU_ALLOC_RATIO=16.0

# Overcommitment ratio for virtual to physical RAM. Specify 1.0 to
# disable RAM overcommitment.
CONFIG_NOVA_SCHED_RAM_ALLOC_RATIO=1.5

# Protocol used for instance migration. Valid options are: ssh and
# tcp. Note that the tcp protocol is not encrypted, so it is insecure.
# ['ssh', 'tcp']
CONFIG_NOVA_COMPUTE_MIGRATE_PROTOCOL=ssh

# PEM encoded certificate to be used for ssl on the https server,
# leave blank if one should be generated, this certificate should not
# require a passphrase. If CONFIG_HORIZON_SSL is set to 'n' this
# parameter is ignored.
CONFIG_VNC_SSL_CERT=

# SSL keyfile corresponding to the certificate if one was entered. If
# CONFIG_HORIZON_SSL is set to 'n' this parameter is ignored.
CONFIG_VNC_SSL_KEY=

# Enter the PCI passthrough array of hash in JSON style for
# controller eg. [{"vendor_id":"1234", "product_id":"5678",
# "name":"default"}, {...}]
CONFIG_NOVA_PCI_ALIAS=

# Enter the PCI passthrough whitelist array of hash in JSON style for
# controller eg. [{"vendor_id":"1234", "product_id":"5678",
# "name':"default"}, {...}]
CONFIG_NOVA_PCI_PASSTHROUGH_WHITELIST=
```

(continues on next page)

(continued from previous page)

```
# The hypervisor driver to use with Nova. Can be either 'qemu' or
# 'kvm'. Defaults to 'qemu' on virtual machines and 'kvm' on bare
# metal hardware. For nested KVM set it explicitly to 'kvm'.
CONFIG_NOVA_LIBVIRT_VIRT_TYPE=%{::default_hypervisor}

# Password to use for OpenStack Networking (neutron) to authenticate
# with the Identity service.
CONFIG_NEUTRON_KS_PW=976c2961895641c4

# The password to use for OpenStack Networking to access the
# database.
CONFIG_NEUTRON_DB_PW=99e053b66e0f4731

# The name of the Open vSwitch bridge (or empty for linuxbridge) for
# the OpenStack Networking L3 agent to use for external  traffic.
# Specify 'provider' if you intend to use a provider network to handle
# external traffic.
CONFIG_NEUTRON_L3_EXT_BRIDGE=br-ex

# Password for the OpenStack Networking metadata agent.
CONFIG_NEUTRON_METADATA_PW=05256ed8a49e4a36

# Specify 'y' to install OpenStack Networking's Load-Balancing-
# as-a-Service (LBaaS). ['y', 'n']
CONFIG_LBAAS_INSTALL=y

# Specify 'y' to install OpenStack Networking's L3 Metering agent
# ['y', 'n']
CONFIG_NEUTRON_METERING_AGENT_INSTALL=y

# Specify 'y' to configure OpenStack Networking's Firewall-
# as-a-Service (FWaaS). ['y', 'n']
CONFIG_NEUTRON_FWAAS=n

# Specify 'y' to configure OpenStack Networking's VPN-as-a-Service
# (VPNaaS). ['y', 'n']
CONFIG_NEUTRON_VPNAAS=n

# Comma-separated list of network-type driver entry points to be
# loaded from the neutron.ml2.type_drivers namespace. ['local',
# 'flat', 'vlan', 'gre', 'vxlan', 'geneve']
CONFIG_NEUTRON_ML2_TYPE_DRIVERS=vxlan,flat

# Comma-separated, ordered list of network types to allocate as
# tenant networks. The 'local' value is only useful for single-box
# testing and provides no connectivity between hosts. ['local',
# 'vlan', 'gre', 'vxlan', 'geneve']
CONFIG_NEUTRON_ML2_TENANT_NETWORK_TYPES=vxlan

# Comma-separated ordered list of networking mechanism driver entry
# points to be loaded from the neutron.ml2.mechanism_drivers
# namespace. ['logger', 'test', 'linuxbridge', 'openvswitch',
# 'hyperv', 'ncs', 'arista', 'cisco_nexus', 'mlnx', 'l2population',
# 'sriovnicswitch', 'ovn']
CONFIG_NEUTRON_ML2_MECHANISM_DRIVERS=openvswitch

# Comma-separated list of physical_network names with which flat
```

(continues on next page)

(continued from previous page)

```
# networks can be created. Use * to allow flat networks with arbitrary
# physical_network names.
CONFIG_NEUTRON_ML2_FLAT_NETWORKS=*

# Comma-separated list of <physical_network>:<vlan_min>:<vlan_max> or
# <physical_network> specifying physical_network names usable for VLAN
# provider and tenant networks, as well as ranges of VLAN tags on each
# available for allocation to tenant networks.
CONFIG_NEUTRON_ML2_VLAN_RANGES=

# Comma-separated list of <tun_min>:<tun_max> tuples enumerating
# ranges of GRE tunnel IDs that are available for tenant-network
# allocation. A tuple must be an array with tun_max +1 - tun_min >
# 1000000.
CONFIG_NEUTRON_ML2_TUNNEL_ID_RANGES=

# Comma-separated list of addresses for VXLAN multicast group. If
# left empty, disables VXLAN from sending allocate broadcast traffic
# (disables multicast VXLAN mode). Should be a Multicast IP (v4 or v6)
# address.
CONFIG_NEUTRON_ML2_VXLAN_GROUP=

# Comma-separated list of <vni_min>:<vni_max> tuples enumerating
# ranges of VXLAN VNI IDs that are available for tenant network
# allocation. Minimum value is 0 and maximum value is 16777215.
CONFIG_NEUTRON_ML2_VNI_RANGES=10:100

# Name of the L2 agent to be used with OpenStack Networking.
# ['linuxbridge', 'openvswitch', 'ovn']
CONFIG_NEUTRON_L2_AGENT=openvswitch

# Comma separated list of supported PCI vendor devices defined by
# vendor_id:product_id according to the PCI ID Repository.
CONFIG_NEUTRON_ML2_SUPPORTED_PCI_VENDOR_DEVS=['15b3:1004', '8086:10ca']

# Comma-separated list of interface mappings for the OpenStack
# Networking ML2 SRIOV agent. Each tuple in the list must be in the
# format <physical_network>:<net_interface>. Example:
# physnet1:eth1,physnet2:eth2,physnet3:eth3.
CONFIG_NEUTRON_ML2_SRIOV_INTERFACE_MAPPINGS=

# Comma-separated list of interface mappings for the OpenStack
# Networking linuxbridge plugin. Each tuple in the list must be in the
# format <physical_network>:<net_interface>. Example:
# physnet1:eth1,physnet2:eth2,physnet3:eth3.
CONFIG_NEUTRON_LB_INTERFACE_MAPPINGS=

# Comma-separated list of bridge mappings for the OpenStack
# Networking Open vSwitch plugin. Each tuple in the list must be in
# the format <physical_network>:<ovs_bridge>. Example: physnet1:br-
# eth1,physnet2:br-eth2,physnet3:br-eth3
CONFIG_NEUTRON_OVS_BRIDGE_MAPPINGS=extnet:br-ex

# Comma-separated list of colon-separated Open vSwitch
# <bridge>:<interface> pairs. The interface will be added to the
# associated bridge. If you desire the bridge to be persistent a value
# must be added to this directive, also
```

(continues on next page)

(continued from previous page)

```
# CONFIG_NEUTRON_OVS_BRIDGE_MAPPINGS must be set in order to create
# the proper port. This can be achieved from the command line by
# issuing the following command: packstack --allinone --os-neutron-
# ovs-bridge-mappings=ext-net:br-ex --os-neutron-ovs-bridge-interfaces
# =br-ex:eth0
CONFIG_NEUTRON_OVS_BRIDGE_IFACES=

# Comma-separated list of Open vSwitch bridges that must be created
# and connected to interfaces in compute nodes when flat or vlan type
# drivers are enabled. These bridges must exist in
# CONFIG_NEUTRON_OVS_BRIDGE_MAPPINGS and
# CONFIG_NEUTRON_OVS_BRIDGE_IFACES. Example: --os-neutron-ovs-bridges-
# compute=br-vlan --os-neutron-ovs-bridge-mappings="extnet:br-
# ex,physnet1:br-vlan" --os-neutron-ovs-bridge-interfaces="br-ex:eth1
# ,br-vlan:eth2"
CONFIG_NEUTRON_OVS_BRIDGES_COMPUTE=

# Name of physical network used for external network when enabling
# CONFIG_PROVISION_DEMO. Name must be one of the included in
# CONFIG_NEUTRON_OVS_BRIDGE_MAPPINGS. Example: --os-neutron-ovs-
# bridge-mappings="extnet:br-ex,physnet1:br-vlan" --os-neutron-ovs-
# bridge-interfaces="br-ex:eth1,br-vlan:eth2" --os-neutron-ovs-
# external-physnet="extnet"
CONFIG_NEUTRON_OVS_EXTERNAL_PHYSNET=extnet

# Interface for the Open vSwitch tunnel. Packstack overrides the IP
# address used for tunnels on this hypervisor to the IP found on the
# specified interface (for example, eth1).
CONFIG_NEUTRON_OVS_TUNNEL_IF=

# Comma-separated list of subnets (for example,
# 192.168.10.0/24,192.168.11.0/24) used for sending tunneling packets.
# This is used to configure IP filtering to accept tunneling packets
# from these subnets instead of specific IP addresses of peer nodes.
# This is useful when you add existing nodes to EXCLUDE_SERVERS
# because, in this case, packstack cannot modify the IP filtering of
# the existing nodes.
CONFIG_NEUTRON_OVS_TUNNEL_SUBNETS=

# VXLAN UDP port.
CONFIG_NEUTRON_OVS_VXLAN_UDP_PORT=4789

# Comma-separated list of bridge mappings for the OpenStack
# Networking Open Virtual Network plugin. Each tuple in the list must
# be in the format <physical_network>:<ovs_bridge>. Example: physnet1
# :br-eth1,physnet2:br-eth2,physnet3:br-eth3
CONFIG_NEUTRON_OVN_BRIDGE_MAPPINGS=extnet:br-ex

# Comma-separated list of colon-separated Open vSwitch
# <bridge>:<interface> pairs. The interface will be added to the
# associated bridge. If you desire the bridge to be persistent a value
# must be added to this directive, also
# CONFIG_NEUTRON_OVN_BRIDGE_MAPPINGS must be set in order to create
# the proper port. This can be achieved from the command line by
# issuing the following command: packstack --allinone --os-neutron-
# ovn-bridge-mappings=ext-net:br-ex --os-neutron-ovn-bridge-interfaces
# =br-ex:eth0
```

(continues on next page)

(continued from previous page)

```
CONFIG_NEUTRON_OVN_BRIDGE_IFACES=

# Comma-separated list of Open vSwitch bridges that must be created
# and connected to interfaces in compute nodes when flat or vlan type
# drivers are enabled. These bridges must exist in
# CONFIG_NEUTRON_OVN_BRIDGE_MAPPINGS and
# CONFIG_NEUTRON_OVN_BRIDGE_IFACES. Example: --os-neutron-ovn-bridges-
# compute=br-vlan --os-neutron-ovn-bridge-mappings="extnet:br-
# ex,physnet1:br-vlan" --os-neutron-ovn-bridge-interfaces="br-ex:eth1
# ,br-vlan:eth2"
CONFIG_NEUTRON_OVN_BRIDGES_COMPUTE=

# Name of physical network used for external network when enabling
# CONFIG_PROVISION_DEMO. Name must be one of the included in
# CONFIG_NEUTRON_OVN_BRIDGE_MAPPINGS. Example: --os-neutron-ovn-
# bridge-mappings="extnet:br-ex,physnet1:br-vlan" --os-neutron-ovn-
# bridge-interfaces="br-ex:eth1,br-vlan:eth2" --os-neutron-ovn-
# external-physnet="extnet"
CONFIG_NEUTRON_OVN_EXTERNAL_PHYSNET=extnet

# Interface for the Open vSwitch tunnel. Packstack overrides the IP
# address used for tunnels on this hypervisor to the IP found on the
# specified interface (for example, eth1).
CONFIG_NEUTRON_OVN_TUNNEL_IF=

# Comma-separated list of subnets (for example,
# 192.168.10.0/24,192.168.11.0/24) used for sending tunneling packets.
# This is used to configure IP filtering to accept tunneling packets
# from these subnets instead of specific IP addresses of peer nodes.
# This is useful when you add existing nodes to EXCLUDE_SERVERS
# because, in this case, packstack cannot modify the IP filtering of
# the existing nodes.
CONFIG_NEUTRON_OVN_TUNNEL_SUBNETS=

# Password to use for the OpenStack File Share service (manila) to
# access the database.
CONFIG_MANILA_DB_PW=PW_PLACEHOLDER

# Password to use for the OpenStack File Share service (manila) to
# authenticate with the Identity service.
CONFIG_MANILA_KS_PW=PW_PLACEHOLDER

# Backend for the OpenStack File Share service (manila); valid
# options are: generic, netapp, glusternative, or glusternfs.
# ['generic', 'netapp', 'glusternative', 'glusternfs']
CONFIG_MANILA_BACKEND=generic

# Denotes whether the driver should handle the responsibility of
# managing share servers. This must be set to false if the driver is
# to operate without managing share servers. Defaults to 'false'
# ['true', 'false']
CONFIG_MANILA_NETAPP_DRV_HANDLES_SHARE_SERVERS=false

# The transport protocol used when communicating with the storage
# system or proxy server. Valid values are 'http' and 'https'.
# Defaults to 'https'. ['https', 'http']
CONFIG_MANILA_NETAPP_TRANSPORT_TYPE=https
```

(continues on next page)

(continued from previous page)

```
# Administrative user account name used to access the NetApp storage
# system.  Defaults to ''.
CONFIG_MANILA_NETAPP_LOGIN=admin

# Password for the NetApp administrative user account specified in
# the CONFIG_MANILA_NETAPP_LOGIN parameter. Defaults to ''.
CONFIG_MANILA_NETAPP_PASSWORD=

# Hostname (or IP address) for the NetApp storage system or proxy
# server. Defaults to ''.
CONFIG_MANILA_NETAPP_SERVER_HOSTNAME=

# The storage family type used on the storage system; valid values
# are ontap_cluster for clustered Data ONTAP. Defaults to
# 'ontap_cluster'. ['ontap_cluster']
CONFIG_MANILA_NETAPP_STORAGE_FAMILY=ontap_cluster

# The TCP port to use for communication with the storage system or
# proxy server. If not specified, Data ONTAP drivers will use 80 for
# HTTP and 443 for HTTPS. Defaults to '443'.
CONFIG_MANILA_NETAPP_SERVER_PORT=443

# Pattern for searching available aggregates for NetApp provisioning.
# Defaults to '(.*)'.
CONFIG_MANILA_NETAPP_AGGREGATE_NAME_SEARCH_PATTERN=(.*)

# Name of aggregate on which to create the NetApp root volume. This
# option only applies when the option
# CONFIG_MANILA_NETAPP_DRV_HANDLES_SHARE_SERVERS is set to True.
CONFIG_MANILA_NETAPP_ROOT_VOLUME_AGGREGATE=

# NetApp root volume name. Defaults to 'root'.
CONFIG_MANILA_NETAPP_ROOT_VOLUME_NAME=root

# This option specifies the storage virtual machine (previously
# called a Vserver) name on the storage cluster on which provisioning
# of shared file systems should occur. This option only applies when
# the option driver_handles_share_servers is set to False. Defaults to
# ''.
CONFIG_MANILA_NETAPP_VSERVER=

# Denotes whether the driver should handle the responsibility of
# managing share servers. This must be set to false if the driver is
# to operate without managing share servers. Defaults to 'true'.
# ['true', 'false']
CONFIG_MANILA_GENERIC_DRV_HANDLES_SHARE_SERVERS=true

# Volume name template for Manila service. Defaults to 'manila-
# share-%s'.
CONFIG_MANILA_GENERIC_VOLUME_NAME_TEMPLATE=manila-share-%s

# Share mount path for Manila service. Defaults to '/shares'.
CONFIG_MANILA_GENERIC_SHARE_MOUNT_PATH=/shares

# Location of disk image for Manila service instance. Defaults to '
CONFIG_MANILA_SERVICE_IMAGE_LOCATION=https://www.dropbox.com/s/vi5oeh10q1qkckh/ubuntu_
↪1204_nfs_cifs.qcow2
```

(continues on next page)

(continued from previous page)

```
# User in Manila service instance.
CONFIG_MANILA_SERVICE_INSTANCE_USER=ubuntu

# Password to service instance user.
CONFIG_MANILA_SERVICE_INSTANCE_PASSWORD=ubuntu

# Type of networking that the backend will use. A more detailed
# description of each option is available in the Manila docs. Defaults
# to 'neutron'. ['neutron', 'nova-network', 'standalone']
CONFIG_MANILA_NETWORK_TYPE=neutron

# Gateway IPv4 address that should be used. Required. Defaults to ''.
CONFIG_MANILA_NETWORK_STANDALONE_GATEWAY=

# Network mask that will be used. Can be either decimal like '24' or
# binary like '255.255.255.0'. Required. Defaults to ''.
CONFIG_MANILA_NETWORK_STANDALONE_NETMASK=

# Set it if network has segmentation (VLAN, VXLAN, etc). It will be
# assigned to share-network and share drivers will be able to use this
# for network interfaces within provisioned share servers. Optional.
# Example: 1001. Defaults to ''.
CONFIG_MANILA_NETWORK_STANDALONE_SEG_ID=

# Can be IP address, range of IP addresses or list of addresses or
# ranges. Contains addresses from IP network that are allowed to be
# used. If empty, then will be assumed that all host addresses from
# network can be used. Optional. Examples: 10.0.0.10 or
# 10.0.0.10-10.0.0.20 or
# 10.0.0.10-10.0.0.20,10.0.0.30-10.0.0.40,10.0.0.50. Defaults to ''.
CONFIG_MANILA_NETWORK_STANDALONE_IP_RANGE=

# IP version of network. Optional. Defaults to '4'. ['4', '6']
CONFIG_MANILA_NETWORK_STANDALONE_IP_VERSION=4

# List of GlusterFS servers that can be used to create shares. Each
# GlusterFS server should be of the form [remoteuser@]<volserver>, and
# they are assumed to belong to distinct Gluster clusters.
CONFIG_MANILA_GLUSTERFS_SERVERS=

# Path of Manila host's private SSH key file.
CONFIG_MANILA_GLUSTERFS_NATIVE_PATH_TO_PRIVATE_KEY=

# Regular expression template used to filter GlusterFS volumes for
# share creation. The regex template can optionally (ie. with support
# of the GlusterFS backend) contain the #{size} parameter which
# matches an integer (sequence of digits) in which case the value
# shall be intepreted as size of the volume in GB. Examples: "manila-
# share-volume-d+$", "manila-share-volume-#{size}G-d+$"; with matching
# volume names, respectively: "manila-share-volume-12", "manila-share-
# volume-3G-13". In latter example, the number that matches "#{size}",
# that is, 3, is an indication that the size of volume is 3G.
CONFIG_MANILA_GLUSTERFS_VOLUME_PATTERN=

# Specifies the GlusterFS volume to be mounted on the Manila host.
# For e.g: [remoteuser@]<volserver>:/<volid>
```

(continues on next page)

(continued from previous page)

```
CONFIG_MANILA_GLUSTERFS_TARGET=

# Base directory containing mount points for Gluster volumes.
CONFIG_MANILA_GLUSTERFS_MOUNT_POINT_BASE=

# Type of NFS server that mediate access to the Gluster volumes
# (Gluster or Ganesha).
CONFIG_MANILA_GLUSTERFS_NFS_SERVER_TYPE=gluster

# Path of Manila host's private SSH key file.
CONFIG_MANILA_GLUSTERFS_PATH_TO_PRIVATE_KEY=

# Remote Ganesha server node's IP address.
CONFIG_MANILA_GLUSTERFS_GANESHA_SERVER_IP=

# Specify 'y' to set up Horizon communication over https. ['y', 'n']
CONFIG_HORIZON_SSL=n

# Secret key to use for Horizon Secret Encryption Key.
CONFIG_HORIZON_SECRET_KEY=a8fe06092db44944a5bc6305911bbcd6

# PEM-encoded certificate to be used for SSL connections on the https
# server. To generate a certificate, leave blank.
CONFIG_HORIZON_SSL_CERT=

# SSL keyfile corresponding to the certificate if one was specified.
# The certificate should not require a passphrase.
CONFIG_HORIZON_SSL_KEY=

CONFIG_HORIZON_SSL_CACERT=

# Password to use for the Object Storage service to authenticate with
# the Identity service.
CONFIG_SWIFT_KS_PW=d09d01ba889b482e

# Comma-separated list of devices to use as storage device for Object
# Storage. Each entry must take the format /path/to/dev (for example,
# specifying /dev/vdb installs /dev/vdb as the Object Storage storage
# device; Packstack does not create the filesystem, you must do this
# first). If left empty, Packstack creates a loopback device for test
# setup.
CONFIG_SWIFT_STORAGES=

# Number of Object Storage storage zones; this number MUST be no
# larger than the number of configured storage devices.
CONFIG_SWIFT_STORAGE_ZONES=1

# Number of Object Storage storage replicas; this number MUST be no
# larger than the number of configured storage zones.
CONFIG_SWIFT_STORAGE_REPLICAS=1

# File system type for storage nodes. ['xfs', 'ext4']
CONFIG_SWIFT_STORAGE_FSTYPE=ext4

# Custom seed number to use for swift_hash_path_suffix in
# /etc/swift/swift.conf. If you do not provide a value, a seed number
# is automatically generated.
```

(continues on next page)

(continued from previous page)

```
CONFIG_SWIFT_HASH=21aa09faa5474ad6

# Size of the Object Storage loopback file storage device.
CONFIG_SWIFT_STORAGE_SIZE=2G

# Password used by Orchestration service user to authenticate against
# the database.
CONFIG_HEAT_DB_PW=f7ca5fea54fe4e60

# Encryption key to use for authentication in the Orchestration
# database (16, 24, or 32 chars).
CONFIG_HEAT_AUTH_ENC_KEY=e3c4823d506c4d6d

# Password to use for the Orchestration service to authenticate with
# the Identity service.
CONFIG_HEAT_KS_PW=678525b8eaca46c4

# Specify 'y' to install the Orchestration CloudFormation API. ['y',
# 'n']
CONFIG_HEAT_CFN_INSTALL=y

# Name of the Identity domain for Orchestration.
CONFIG_HEAT_DOMAIN=heat

# Name of the Identity domain administrative user for Orchestration.
CONFIG_HEAT_DOMAIN_ADMIN=heat_admin

# Password for the Identity domain administrative user for
# Orchestration.
CONFIG_HEAT_DOMAIN_PASSWORD=4e6b56b8b8f14ccd

# Specify 'y' to provision for demo usage and testing. ['y', 'n']
CONFIG_PROVISION_DEMO=y

# Specify 'y' to configure the OpenStack Integration Test Suite
# (tempest) for testing. The test suite requires OpenStack Networking
# to be installed. ['y', 'n']
CONFIG_PROVISION_TEMPEST=n

# CIDR network address for the floating IP subnet.
CONFIG_PROVISION_DEMO_FLOATRANGE=172.24.4.0/24

# Allocation pools in the floating IP subnet.
CONFIG_PROVISION_DEMO_ALLOCATION_POOLS=[]

# The name to be assigned to the demo image in Glance (default
# "cirros").
CONFIG_PROVISION_IMAGE_NAME=cirros

# A URL or local file location for an image to download and provision
# in Glance (defaults to a URL for a recent "cirros" image).
CONFIG_PROVISION_IMAGE_URL=http://download.cirros-cloud.net/0.3.5/cirros-0.3.5-x86_64-
↪disk.img

# Format for the demo image (default "qcow2").
CONFIG_PROVISION_IMAGE_FORMAT=qcow2
```

(continues on next page)

(continued from previous page)

```
# Properties of the demo image (none by default).
CONFIG_PROVISION_IMAGE_PROPERTIES=

# User to use when connecting to instances booted from the demo
# image.
CONFIG_PROVISION_IMAGE_SSH_USER=cirros

# Name of the uec image created in Glance used in tempest tests
# (default "cirros-uec").
CONFIG_PROVISION_UEC_IMAGE_NAME=cirros-uec

# URL of the kernel image copied to Glance image for uec image
# (defaults to a URL for a recent "cirros" uec image).
CONFIG_PROVISION_UEC_IMAGE_KERNEL_URL=http://download.cirros-cloud.net/0.3.5/cirros-0.
↪3.5-x86_64-kernel

# URL of the ramdisk image copied to Glance image for uec image
# (defaults to a URL for a recent "cirros" uec image).
CONFIG_PROVISION_UEC_IMAGE_RAMDISK_URL=http://download.cirros-cloud.net/0.3.5/cirros-
↪0.3.5-x86_64-initramfs

# URL of the disk image copied to Glance image for uec image
# (defaults to a URL for a recent "cirros" uec image).
CONFIG_PROVISION_UEC_IMAGE_DISK_URL=http://download.cirros-cloud.net/0.3.5/cirros-0.3.
↪5-x86_64-disk.img

CONFIG_TEMPEST_HOST=

# Name of the Integration Test Suite provisioning user. If you do not
# provide a user name, Tempest is configured in a standalone mode.
CONFIG_PROVISION_TEMPEST_USER=

# Password to use for the Integration Test Suite provisioning user.
CONFIG_PROVISION_TEMPEST_USER_PW=PW_PLACEHOLDER

# CIDR network address for the floating IP subnet.
CONFIG_PROVISION_TEMPEST_FLOATRANGE=172.24.4.0/24

# Primary flavor name to use in Tempest.
CONFIG_PROVISION_TEMPEST_FLAVOR_NAME=m1.nano

# Primary flavor's disk quota in Gb.
CONFIG_PROVISION_TEMPEST_FLAVOR_DISK=0

# Primary flavor's ram in Mb.
CONFIG_PROVISION_TEMPEST_FLAVOR_RAM=128

# Primary flavor's vcpus number.
CONFIG_PROVISION_TEMPEST_FLAVOR_VCPUS=1

# Alternative flavor name to use in Tempest.
CONFIG_PROVISION_TEMPEST_FLAVOR_ALT_NAME=m1.micro

# Alternative flavor's disk quota in Gb.
CONFIG_PROVISION_TEMPEST_FLAVOR_ALT_DISK=0

# Alternative flavor's ram in Mb.
```

(continues on next page)

(continued from previous page)

```
CONFIG_PROVISION_TEMPEST_FLAVOR_ALT_RAM=128

# Alternative flavor's vcpus number.
CONFIG_PROVISION_TEMPEST_FLAVOR_ALT_VCPUS=1

# Specify 'y' to run Tempest smoke test as last step of installation.
CONFIG_RUN_TEMPEST=n

# Test suites to run, example: "smoke dashboard TelemetryAlarming".
# Optional, defaults to "smoke".
CONFIG_RUN_TEMPEST_TESTS=smoke

# Specify 'y' to configure the Open vSwitch external bridge for an
# all-in-one deployment (the L3 external bridge acts as the gateway
# for virtual machines). ['y', 'n']
CONFIG_PROVISION_OVS_BRIDGE=y

# Password to use for Gnocchi to access the database.
CONFIG_GNOCCHI_DB_PW=46612ab8aebe46c4

# Password to use for Gnocchi to authenticate with the Identity
# service.
CONFIG_GNOCCHI_KS_PW=bdef00d6ab8f4ec3

# Secret key for signing Telemetry service (ceilometer) messages.
CONFIG_CEILOMETER_SECRET=0f2c8fc4b0aa4ca6

# Password to use for Telemetry to authenticate with the Identity
# service.
CONFIG_CEILOMETER_KS_PW=6f6306dc0e504470

# Ceilometer service name. ['httpd', 'ceilometer']
CONFIG_CEILOMETER_SERVICE_NAME=httpd

# Backend driver for Telemetry's group membership coordination.
# ['redis', 'none']
CONFIG_CEILOMETER_COORDINATION_BACKEND=redis

# Whether to enable ceilometer middleware in swift proxy. By default
# this should be false to avoid unnecessary load.
CONFIG_ENABLE_CEILOMETER_MIDDLEWARE=n

# IP address of the server on which to install the Redis server.
CONFIG_REDIS_HOST=10.0.1.20

# Port on which the Redis server listens.
CONFIG_REDIS_PORT=6379

# Password to use for Telemetry Alarming to authenticate with the
# Identity service.
CONFIG_AODH_KS_PW=6452ce6bfc8244a5

# Password to use for Telemetry Alarming (AODH) to access the
# database.
CONFIG_AODH_DB_PW=67194291fed6497c

# Password to use for Panko to access the database.
```

(continues on next page)

(continued from previous page)

```
CONFIG_PANKO_DB_PW=PW_PLACEHOLDER

# Password to use for Panko to authenticate with the Identity
# service.
CONFIG_PANKO_KS_PW=PW_PLACEHOLDER

# Password to use for OpenStack Database-as-a-Service (trove) to
# access the database.
CONFIG_TROVE_DB_PW=PW_PLACEHOLDER

# Password to use for OpenStack Database-as-a-Service to authenticate
# with the Identity service.
CONFIG_TROVE_KS_PW=PW_PLACEHOLDER

# User name to use when OpenStack Database-as-a-Service connects to
# the Compute service.
CONFIG_TROVE_NOVA_USER=trove

# Tenant to use when OpenStack Database-as-a-Service connects to the
# Compute service.
CONFIG_TROVE_NOVA_TENANT=services

# Password to use when OpenStack Database-as-a-Service connects to
# the Compute service.
CONFIG_TROVE_NOVA_PW=PW_PLACEHOLDER

# Password to use for OpenStack Data Processing (sahara) to access
# the database.
CONFIG_SAHARA_DB_PW=PW_PLACEHOLDER

# Password to use for OpenStack Data Processing to authenticate with
# the Identity service.
CONFIG_SAHARA_KS_PW=PW_PLACEHOLDER

# Password to use for the Magnum to access the database.
CONFIG_MAGNUM_DB_PW=PW_PLACEHOLDER

# Password to use for the Magnum to authenticate with the Identity
# service.
CONFIG_MAGNUM_KS_PW=PW_PLACEHOLDER
```

16.3 Despliegue de OpenStack Multinode con Packstack, VirtualBox y Vagrant

Table of Contents

16.3.1 Prerequisitos del sistema

Desde nuestro sistema Operativo (Windows, macOS, Linux) debemos tener instaladas los siguientes programas:

- VirtualBox
- Vagrant

Este escenario de OpenStack que será desplegado consta de 3 VMs: un Controller/Network node y dos Compute nodes. Se requiere al menos 12 GB de RAM y 4 CPUs

---

Important: Si existen errores en la creación de las instancias o que las instancias creadas se apagan automáticamete, esto podría deberse a la falta de recursos en el host.

Revisar la memoria RAM disponible con `free -m` y el uso de CPU con `top` o `htop`. En caso los recursos se están usando al máximo, apagar la VM y asignarle más recursos desde el host en VirtualBox.

16.3.2 Configuración de VirtualBox

Host-only Network Adapter

1. Crear un Host-only Network Adapter haciendo clic en el botón *Global Tools*, *Host Network Manager*. Clic en el botón *Create*:

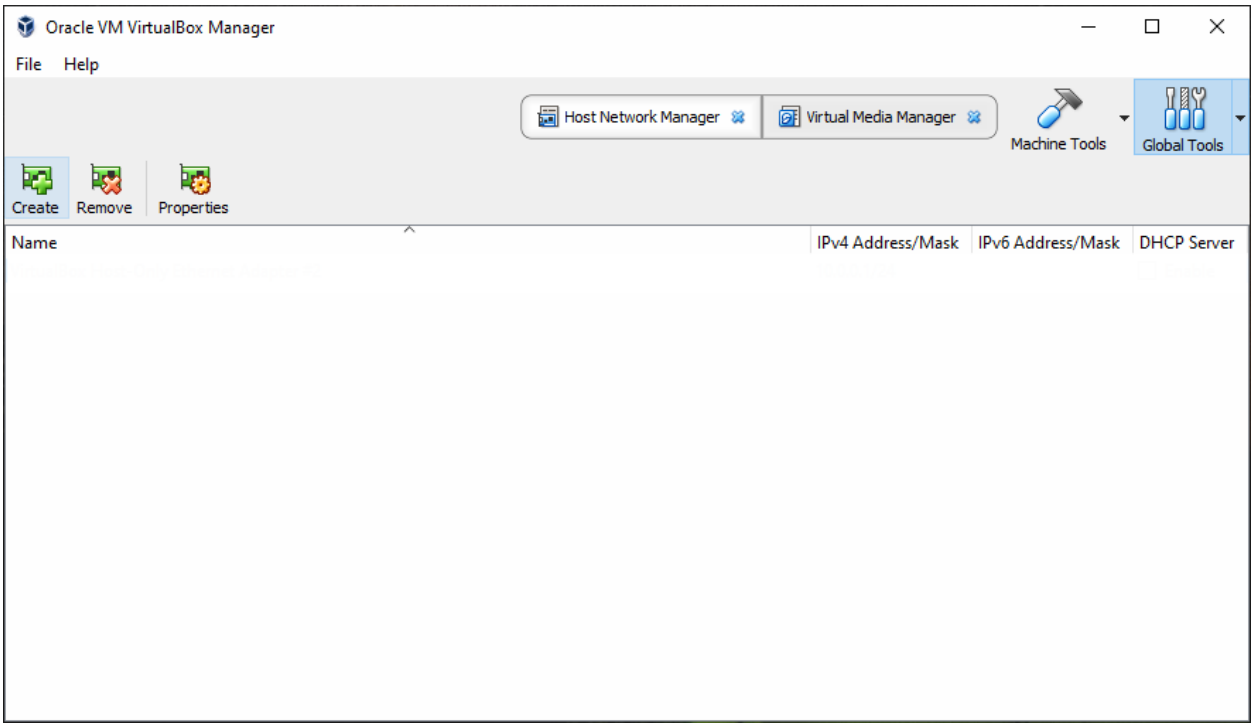


Fig. 13: Windows - VirtualBox v5.2 - Crear un nuevo adaptador

2. Se creará un adaptador con un nombre predefinido al cual podremos editar la dirección IPv4 respectiva. En este caso no será necesario activar la opción de DHCP:

Note: Sobre el adaptador de red creado:

- El adaptador pertenece a la red de management
- La dirección IP del adaptador es el Host-side de la red

16.3.3 Definición del archivo `Vagrantfile`

El archivo `Vagrantfile` se define de la siguiente forma:

```
# -*- mode: ruby -*-
# vi: set ft=ruby :
servers=[
  {
    :hostname => "compute1",
    :box => "geerlingguy/centos7",
    :ram => 2048,
```

(continues on next page)

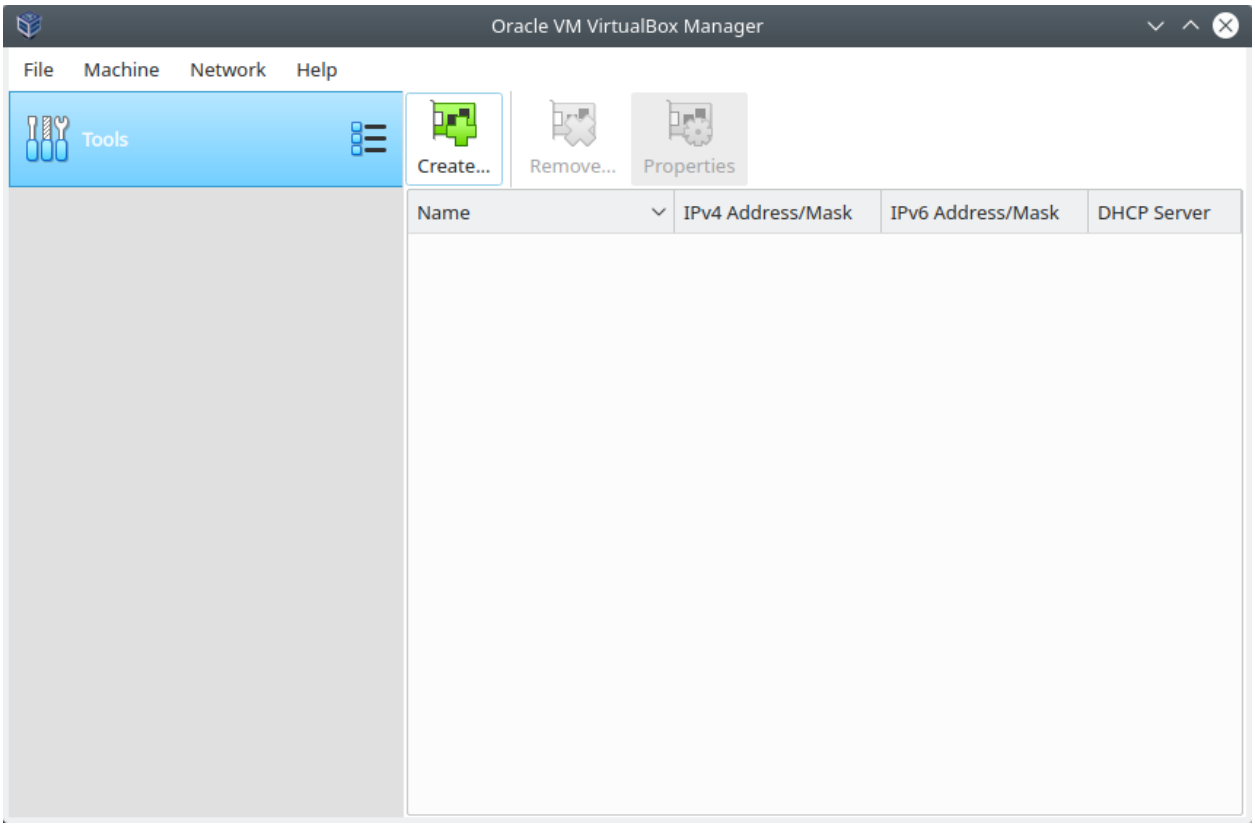


Fig. 14: Linux - VirtualBox v6.0 - Crear un nuevo adaptador

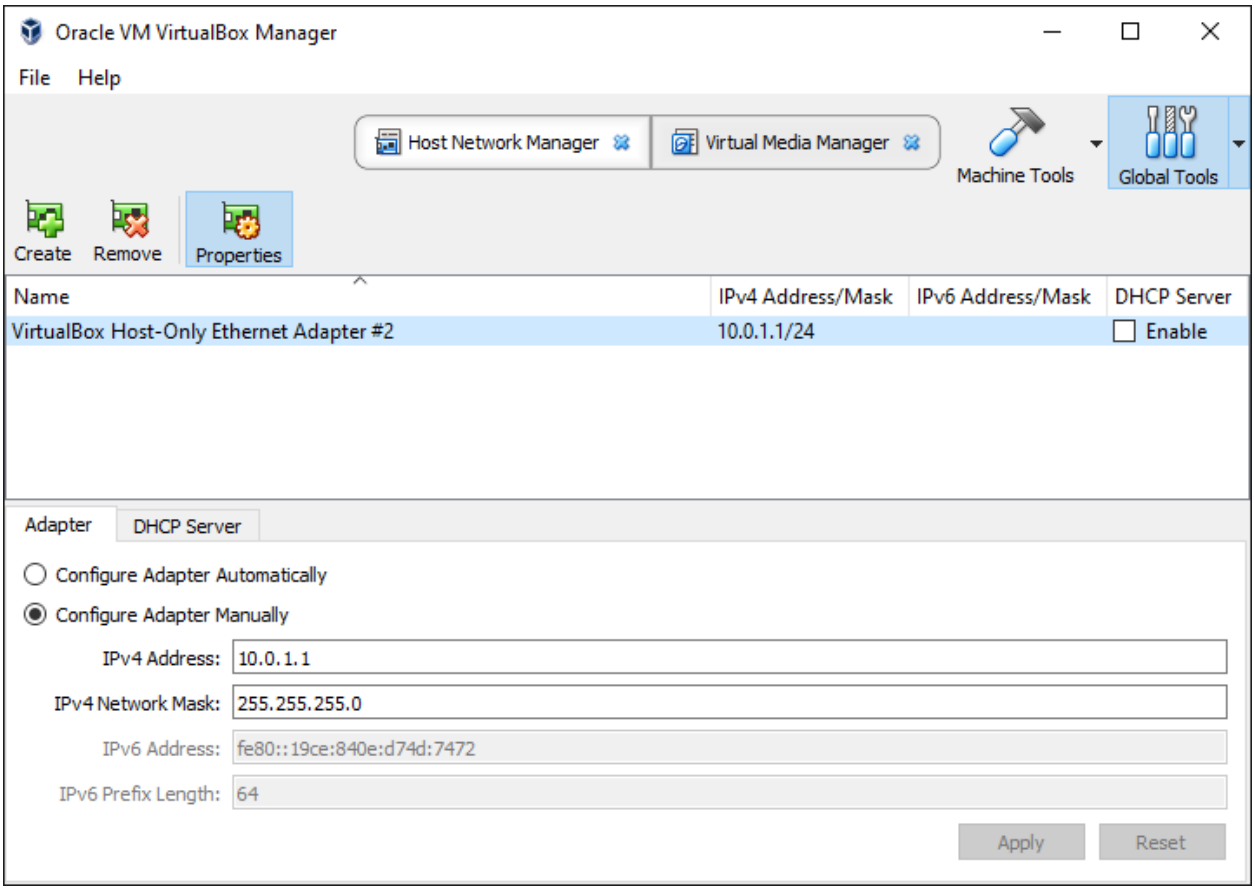


Fig. 15: Windows - VirtualBox v5.2 - Configuración del adaptador

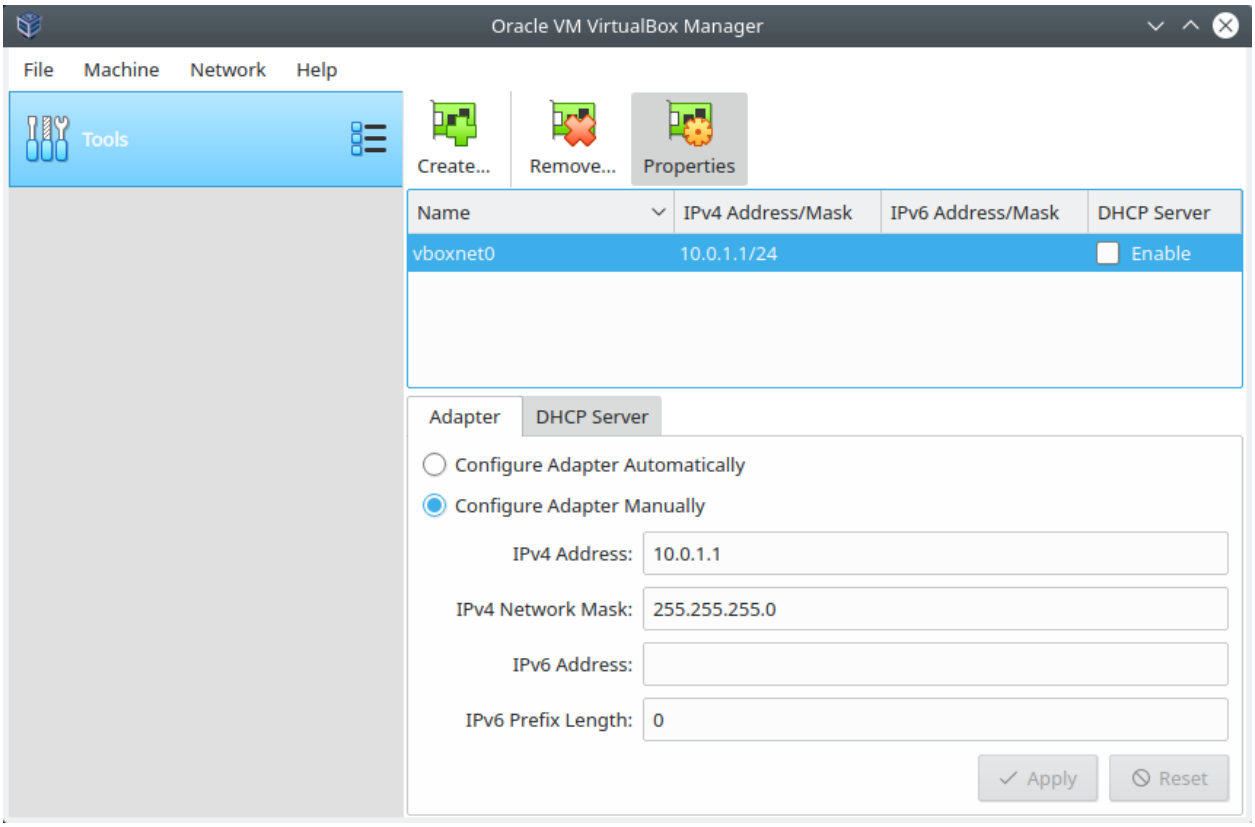


Fig. 16: Linux - VirtualBox v6.0 - Configuración del adaptador

(continued from previous page)

```
    :cpu => 1,
    :script => "sh /vagrant/compute1_setup.sh"
  },
  {
    :hostname => "compute2",
    :box => "geerlingguy/centos7",
    :ram => 2048,
    :cpu => 1,
    :script => "sh /vagrant/compute2_setup.sh"
  },
  {
    :hostname => "packstack",
    :box => "geerlingguy/centos7",
    :ram => 8192,
    :cpu => 2,
    :script => "sh /vagrant/packstack_setup.sh"
  }
]
# All Vagrant configuration is done below. The "2" in Vagrant.configure
# configures the configuration version (we support older styles for
# backwards compatibility). Please don't change it unless you know what
# you're doing.
Vagrant.configure("2") do |config|
  servers.each do |machine|
    config.vm.define machine[:hostname] do |node|
      node.vm.box = machine[:box]
      node.vm.hostname = machine[:hostname]
      node.vm.provider "virtualbox" do |vb|
        vb.customize ["modifyvm", :id, "--memory", machine[:ram], "--cpus",␣
↪machine[:cpu]]
        vb.customize ["modifyvm", :id, "--nic2", "hostonly", "--hostonlyadapter2",
↪"VirtualBox Host-Only Ethernet Adapter #2"]
      end
      node.vm.provision "shell", inline: machine[:script], privileged: true, run:
↪"once"
    end
  end
end
```

Important:

- Vagrant configura automáticamente la primera interfaz de red de una nueva VM en la red NAT. A través de esta red podemos acceder desde el sistema host al Dashboard o CLI del nodo.
- La primera interfaz de red se usa para tener salida a Internet.
- La segunda interfaz sirve para proveer conectividad del sistema host a la VM (y viceversa).

16.3.4 Definición de archivos de configuración de nodos

En el mismo directorio donde tenemos almacenado el archivo `Vagrantfile` guardaremos los scripts `.sh` que se correrán cuando Vagrant lance las VMs en su respectivo nodo:

Controller/Network (`packstack_setup.sh`)

```
#! /bin/sh

export LANG=en_US.utf-8
export LC_ALL=en_US.utf-8

sed -i -e 's/enabled=1/enabled=0/g' /etc/yum/pluginconf.d/fastestmirror.conf

cat <<- EOF > /etc/sysconfig/network-scripts/ifcfg-enp0s8
DEVICE="enp0s8"
DEFROUTE="no"
BOOTPROTO="static"
IPADDR="10.0.1.20"
NETMASK="255.255.255.0"
DNS1="8.8.8.8"
TYPE="Ethernet"
ONBOOT=yes
EOF

ifdown enp0s8
ifup enp0s8

cat <<- EOF > /etc/hosts
127.0.0.1 localhost
10.0.1.20 packstack
10.0.1.21 compute1
10.0.1.22 compute2
EOF

echo 'centos' >/etc/yum/vars/contentdir

systemctl disable firewalld
systemctl stop firewalld
systemctl disable NetworkManager
systemctl stop NetworkManager
systemctl enable network
systemctl start network

yum install -y centos-release-openstack-queens
yum update -y
yum install -y openstack-packstack

cat <<- EOF > /home/vagrant/run-packstack.sh
export LANG=en_US.utf-8
export LC_ALL=en_US.utf-8
echo "Running 'ssh vagrant@compute1'"
echo "The password is 'vagrant'"
ssh vagrant@compute1 echo "OK"
echo "Running 'ssh vagrant@compute2'"
echo "The password is 'vagrant'"
ssh vagrant@compute2 echo "OK"
echo "Running packstack with options"
echo "The 'root' password is 'vagrant'"
packstack --install-hosts="10.0.1.20","10.0.1.21","10.0.1.22" --os-heat-install=y --
↪os-heat-cfn-install=y --os-neutron-lbaas-install=y --keystone-admin-passwd=
↪"openstack" --keystone-demo-passwd="openstack"
```

(continues on next page)

(continued from previous page)

```
EOF

chown vagrant:vagrant /home/vagrant/run-packstack.sh
```

Compute 1 (`compute1_setup.sh`)

```
#! /bin/sh

export LANG=en_US.utf-8
export LC_ALL=en_US.utf-8

sed -i -e 's/enabled=1/enabled=0/g' /etc/yum/pluginconf.d/fastestmirror.conf
echo 'centos' >/etc/yum/vars/contentdir

cat <<- EOF > /etc/sysconfig/network-scripts/ifcfg-enp0s8
DEVICE="enp0s8"
DEFROUTE="no"
BOOTPROTO="static"
IPADDR="10.0.1.21"
NETMASK="255.255.255.0"
DNS1="8.8.8.8"
TYPE="Ethernet"
ONBOOT=yes
EOF

ifdown enp0s8
ifup enp0s8

cat <<- EOF > /etc/hosts
127.0.0.1 localhost
10.0.1.20 packstack
10.0.1.21 compute1
10.0.1.22 compute2
EOF

systemctl disable firewalld
systemctl stop firewalld
systemctl disable NetworkManager
systemctl stop NetworkManager
systemctl enable network
systemctl start network
```

Compute 2 (`compute2_setup.sh`)

```
#! /bin/sh

export LANG=en_US.utf-8
export LC_ALL=en_US.utf-8

sed -i -e 's/enabled=1/enabled=0/g' /etc/yum/pluginconf.d/fastestmirror.conf
echo 'centos' >/etc/yum/vars/contentdir

cat <<- EOF > /etc/sysconfig/network-scripts/ifcfg-enp0s8
```

(continues on next page)

(continued from previous page)

```
DEVICE="enp0s8"
DEFROUTE="no"
BOOTPROTO="static"
IPADDR="10.0.1.22"
NETMASK="255.255.255.0"
DNS1="8.8.8.8"
TYPE="Ethernet"
ONBOOT=yes
EOF

ifdown enp0s8
ifup enp0s8

cat <<- EOF > /etc/hosts
127.0.0.1 localhost
10.0.1.20 packstack
10.0.1.21 compute1
10.0.1.22 compute2
EOF

systemctl disable firewalld
systemctl stop firewalld
systemctl disable NetworkManager
systemctl stop NetworkManager
systemctl enable network
systemctl start network
```

16.3.5 Correr el entorno Vagrant de máquinas virtuales

En el terminal, cambiar de directorio al lugar donde tenemos almacenado el archivo `Vagrantfile` y los scripts `.sh` de los nodos. Luego, desplegar las máquinas virtuales con `vagrant up`:

```
$ cd multinode-packstack-vagrant

$ vagrant up
```

Comenzará la configuración y despliegue de máquinas virtuales en VirtualBox conforme se ha especificado en el archivo `Vagrantfile`, luego se correrán automáticamente los scripts `.sh` que se ha definido para cada VM:

16.3.6 Instalación de OpenStack con Packstack

Ubicándonos en el mismo directorio donde tenemos el archivo `Vagrantfile` entraremos al terminal de la VM `packstack` con el siguiente comando:

```
$ vagrant ssh packstack
```

Dentro del terminal de esta VM correremos el archivo `run-packstack.sh` que se encuentra en el directorio `/home/vagrant/` y fue definido el el script `packstack_setup.sh`:

```
$ . run-packstack.sh
```

Este script corre comandos de prueba `ssh` a las VMs de los compute nodes para compartir las llaves SSH (imprime el mensaje OK en cada VM) y luego corre el comando packstack con opciones para la instalación de OpenStack. Nos pedirá ingresar la contraseña del usuario root de las VMs de los compute nodes.

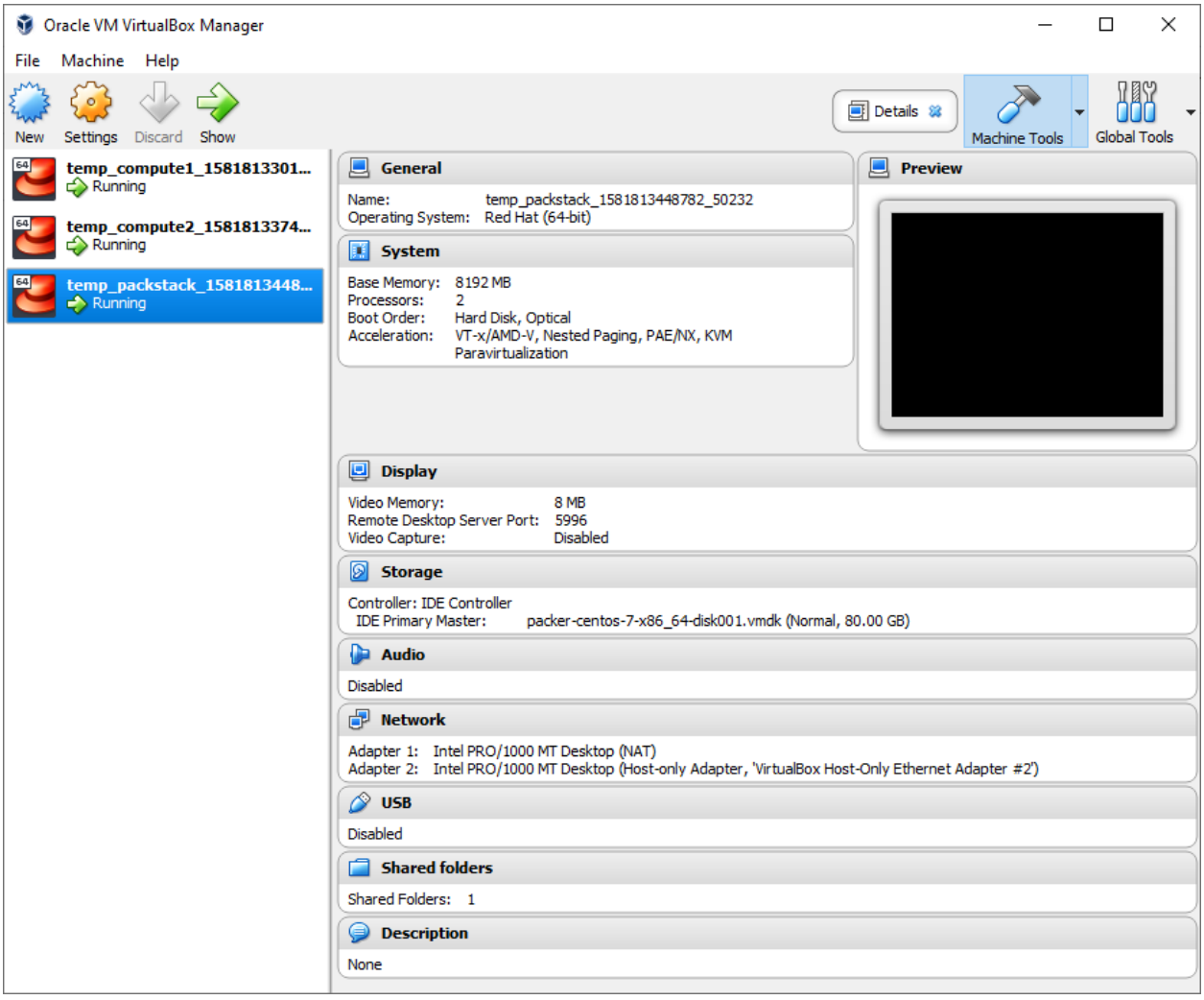


Fig. 17: VirtualBox - VMs desplegadas por Vagrant

16.3.7 Pruebas en el entorno OpenStack desplegado

Luego de media hora aproximadamente (40 minutos en total: 5 min. despliegue de VMs + 35 min. instalación de OpenStack), la instalación de OpenStack con Packstack habrá finalizado y podremos ingresar al Dashboard o al CLI de OpenStack.

Note: El Dashboard de OpenStack podría demorar un corto tiempo para arrancar la primera vez luego de haber instalado OpenStack.

Ahora realizaremos una prueba de despliegue desde el CLI:

Important: La imagen de CirrOS que se crea por defecto como `demo` al instalar Packstack tiene fallos pues tiene un tamaño reducido de 273 bytes. La causa de esto es que puede ser que se haya descargado sin la opción `-L` del comando `curl`. Por lo tanto, crearemos nuestra propia imagen CirrOS:

La topología que se desea lograr es la siguiente:

Y los comandos que ejecutaremos en el nodo Controller/Network serán los siguientes:

```
$ sudo su
'#' cd /root

'#' source keystonerc_admin

'#' mkdir images
'#' curl -o /root/images/cirros-0.4.0-x86_64-disk.img -L http://download.cirros-cloud.
↪net/0.4.0/cirros-0.4.0-x86_64-disk.img
'#' openstack image create --min-disk 1 --min-ram 128 --public --disk-format qcow2 --
↪file /root/images/cirros-0.4.0-x86_64-disk.img cirros1

'#' source keystonerc_demo

'#' openstack network list

+--------------------------------------+---------+--------------------------------------
↪--+
| ID                                   | Name    | Subnets                              ␣
↪  |
+--------------------------------------+---------+--------------------------------------
↪--+
| 836312ac-15d2-462c-a588-ccb6bda9953f | private | 7684b9ae-f2e4-4731-8b22-
↪a4df18925ec0 |
| f1b16d9e-cb71-47e1-9654-d78eb54f1621 | public  | 2faf694b-320e-4e87-9fe6-
↪2146b169c591 |
+--------------------------------------+---------+--------------------------------------
↪--+

'#' openstack server create --image cirros1 --flavor 1 --min 2 --max 2 --nic net-
↪id=836312ac-15d2-462c-a588-ccb6bda9953f test
```

En este despliegue de prueba se han creado dos instancias al mismo tiempo y con las mismas características de forma que usen todos los recursos de un hypervisor al crear una instancia y para crear otra instancia se requiera usar el hypervisor del otro compute node. Logrando tener una instancia en cada compute node (o equivalentemente, cada instancia administrada por un hypervisor distinto).

Podremos ingresar a la consola de cada instancia desde el dashboard y probar conectividad entre ellas:

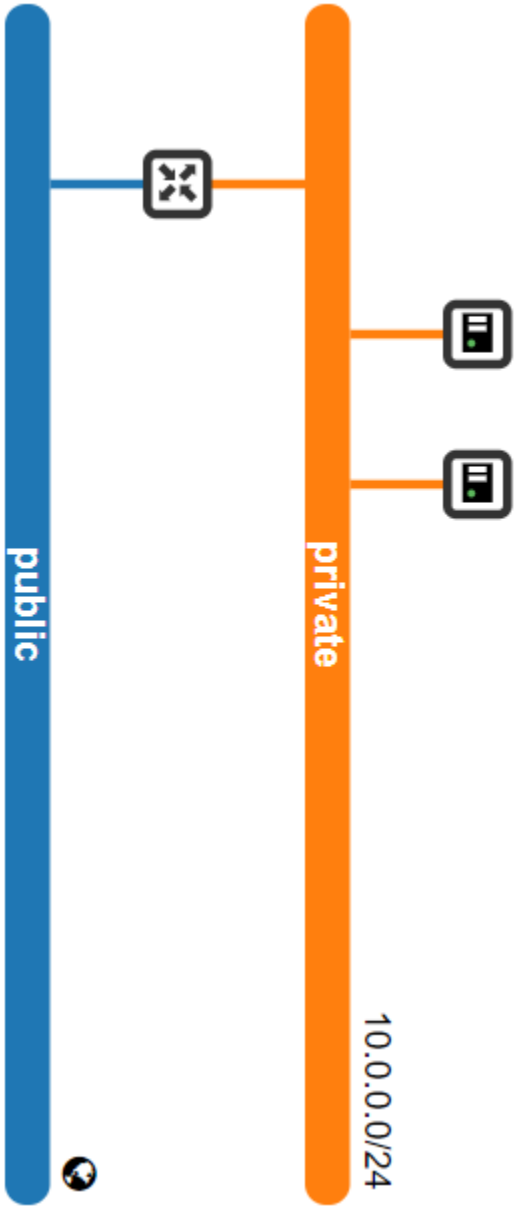


Fig. 18: OpenStack - Topología desplegada

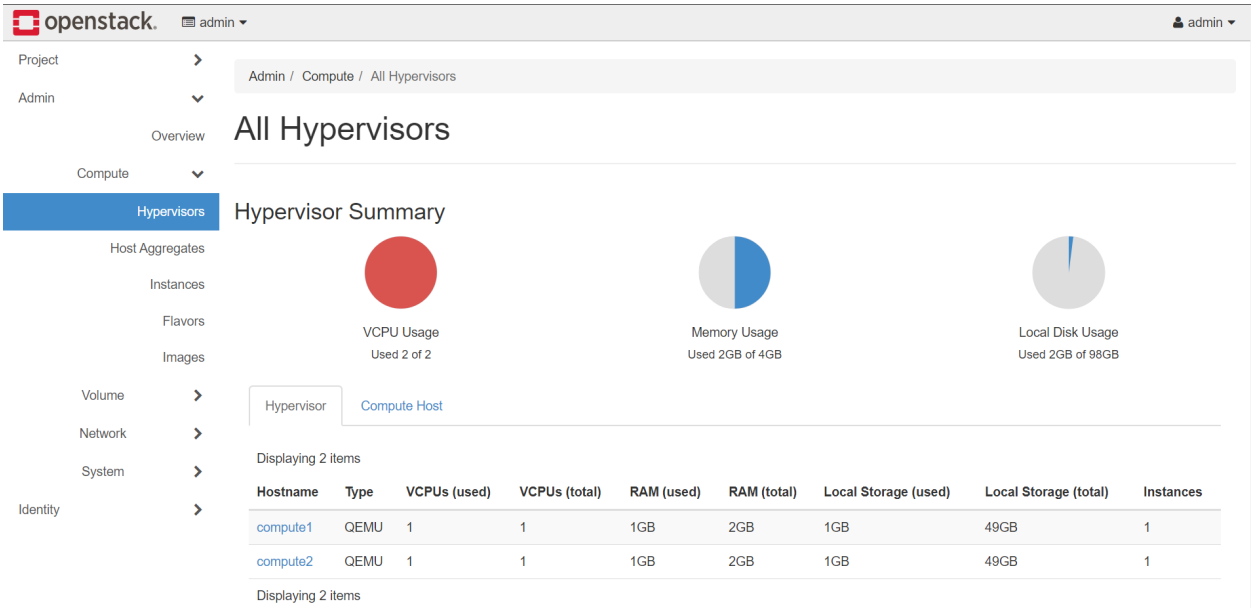


Fig. 19: OpenStack - Uso de Hypervisors (Compute 1 y Compute 2)

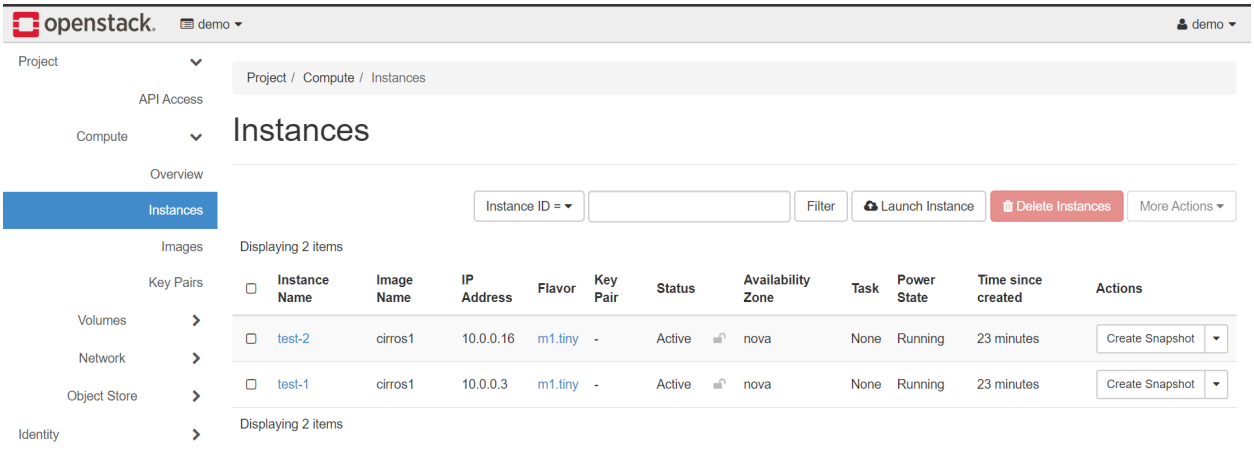


Fig. 20: OpenStack - Instancias desplegadas

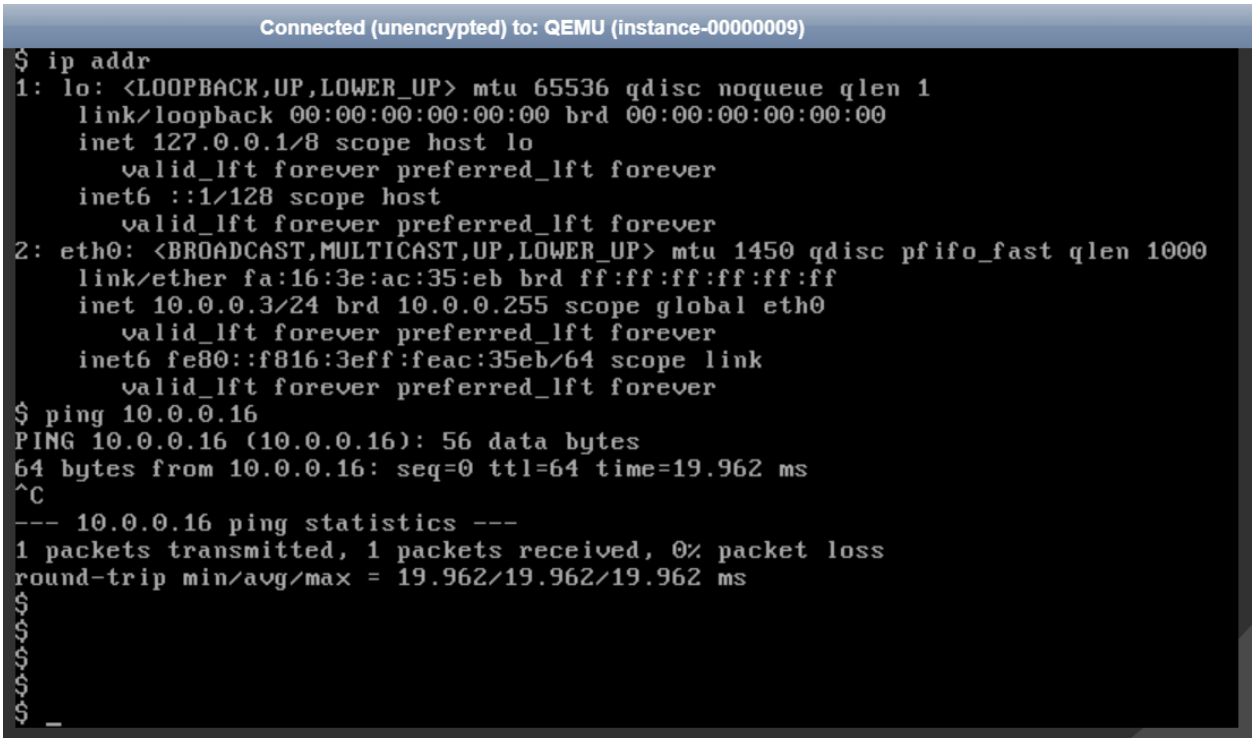


Fig. 21: OpenStack - Consola `test-1`

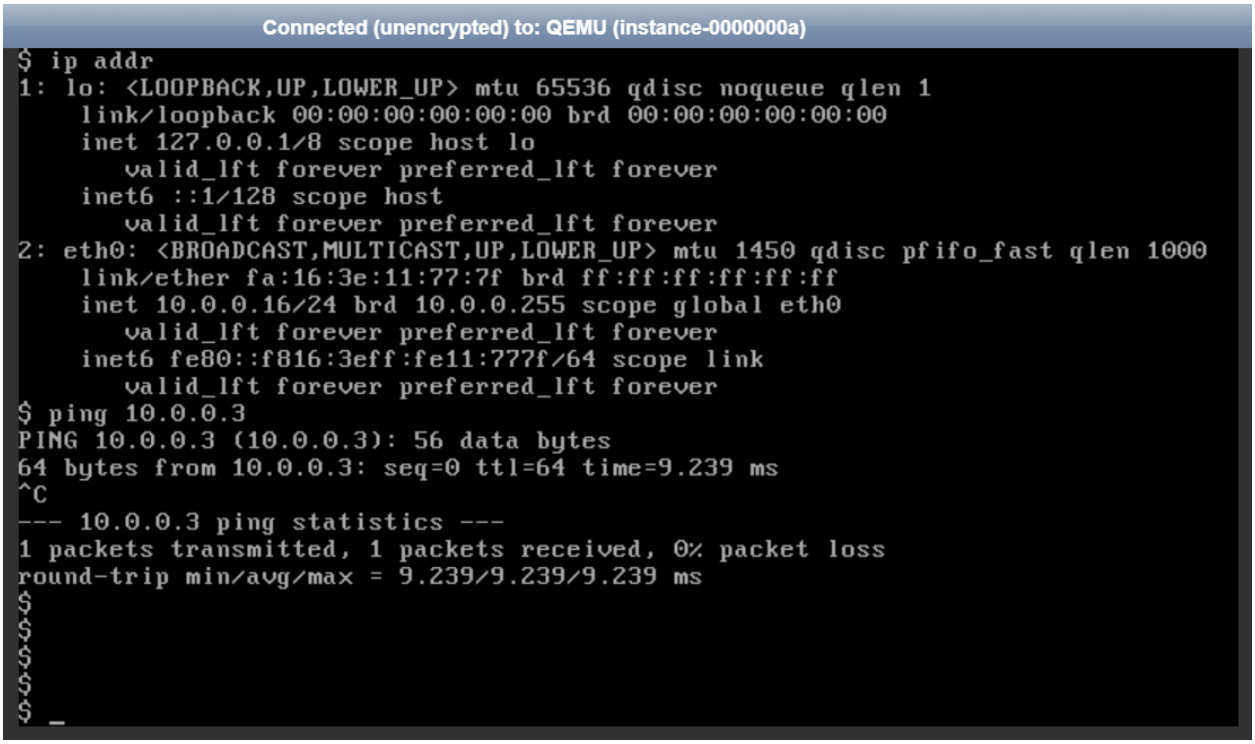


Fig. 22: OpenStack - Consola `test-2`

16.3.8 Anexo 1: Configuración de red desplegada

Host configuration (Windows user)

Las interfaces de red que tenemos en el sistema host de Windows son las siguientes:

```
C:\Users\usuario>ipconfig

Configuración IP de Windows


Adaptador de Ethernet VirtualBox Host-Only Network #2:

Sufijo DNS específico para la conexión. . :
Dirección IPv4. . . . . . . . . . . . . . : 10.0.1.1
Máscara de subred . . . . . . . . . . . . : 255.255.255.0
Puerta de enlace predeterminada . . . . . :

Adaptador de Ethernet Ethernet:

Sufijo DNS específico para la conexión. . :
Dirección IPv4. . . . . . . . . . . . . . : 192.168.1.10
Máscara de subred . . . . . . . . . . . . : 255.255.255.0
Puerta de enlace predeterminada . . . . . : 192.168.1.1
```

- La interfaz `Adaptador de Ethernet Ethernet` es nuestra tarjeta de red física con salida a Internet a través de un router físico (`192.168.1.1`). Se le ha asignado la IP `192.168.1.10`.
- La interfaz `Adaptador de Ethernet VirtualBox Host-Only Network #2` ha sido creada con VirtualBox dentro de `Host Network Manager`. Se le ha asignado la IP `10.0.1.1`. Esta interfaz no cuenta con salida al exterior (Internet).

VM configuration (virtualBox config)

Las 3 VMs creadas poseen cada una 2 interfaces del mismo tipo que pueden verse desde el lado del SO host o del lado del SO guest, obteniendo información útil de su implementación:

Interfaces desde el lado host

Las VMs con sistema CentOS creadas por VirtualBox poseen dos interfaces del mismo tipo conectadas virtualmente:

1. **Adaptador 1 - Interfaz tipo NAT**:

La primera interfaz de red es usada para que la VM tenga salida a Internet (en la VM se ve como la interfaz `enp0s3` o `eth0`):

- La interfaz se encuentra conectada a una red virtual tipo NAT.
- En este caso la interfaz de cada VM tiene asignada la misma IP `10.0.2.15` y comparten la misma configuración de red (misma MAC).

2. **Adaptador 2 - Interfaz tipo Host-only Adapter**:

La segunda interfaz de red de cada VM es usada para que exista comunicación entre el sistema host y la VM (en la VM se ve como la interfaz `enp0s8` o `eth1`):

- La interfaz se encuentra conectada a la red física del host, siendo del tipo Host-only Adapter. Desde cada VM podemos llegar a los equipos dentro de la red host (`192.168.1.0/24`).

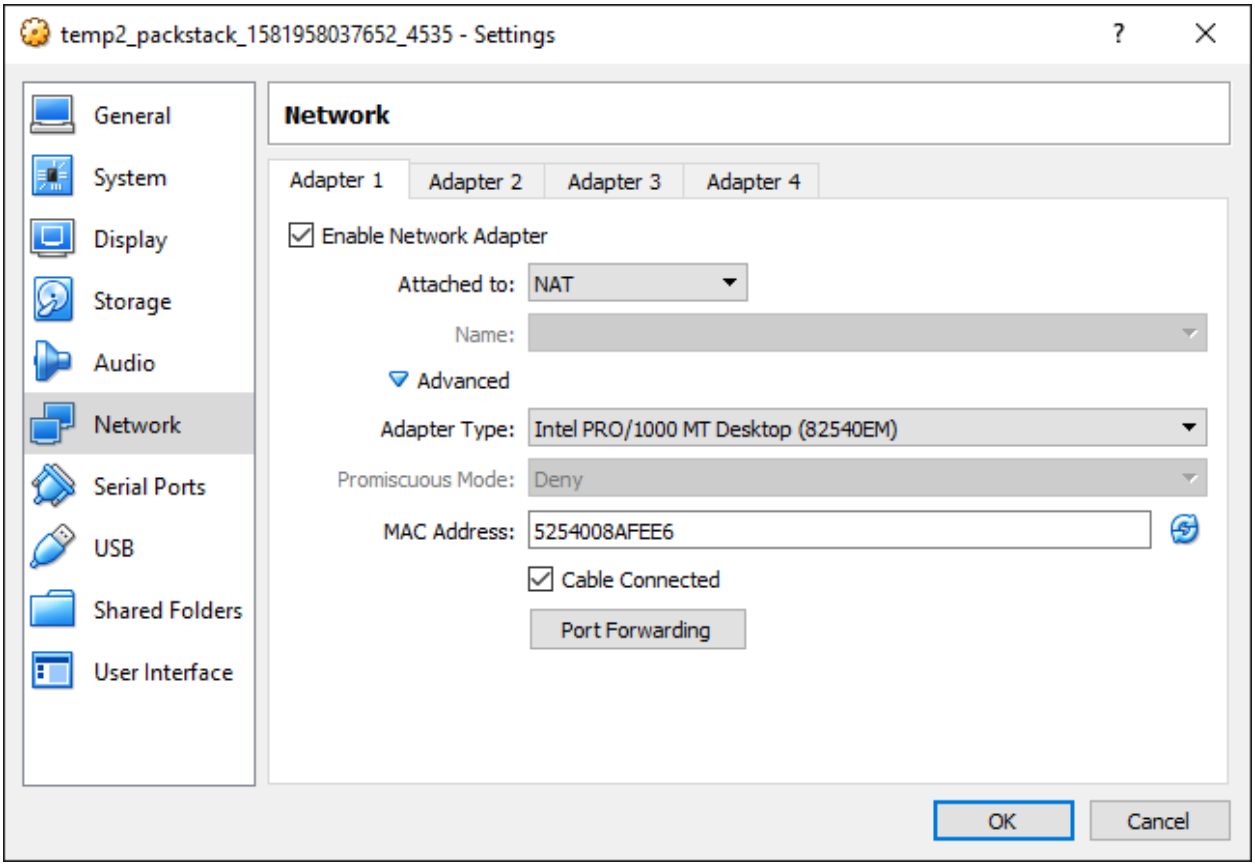


Fig. 23: VirtualBox - Adaptador 1

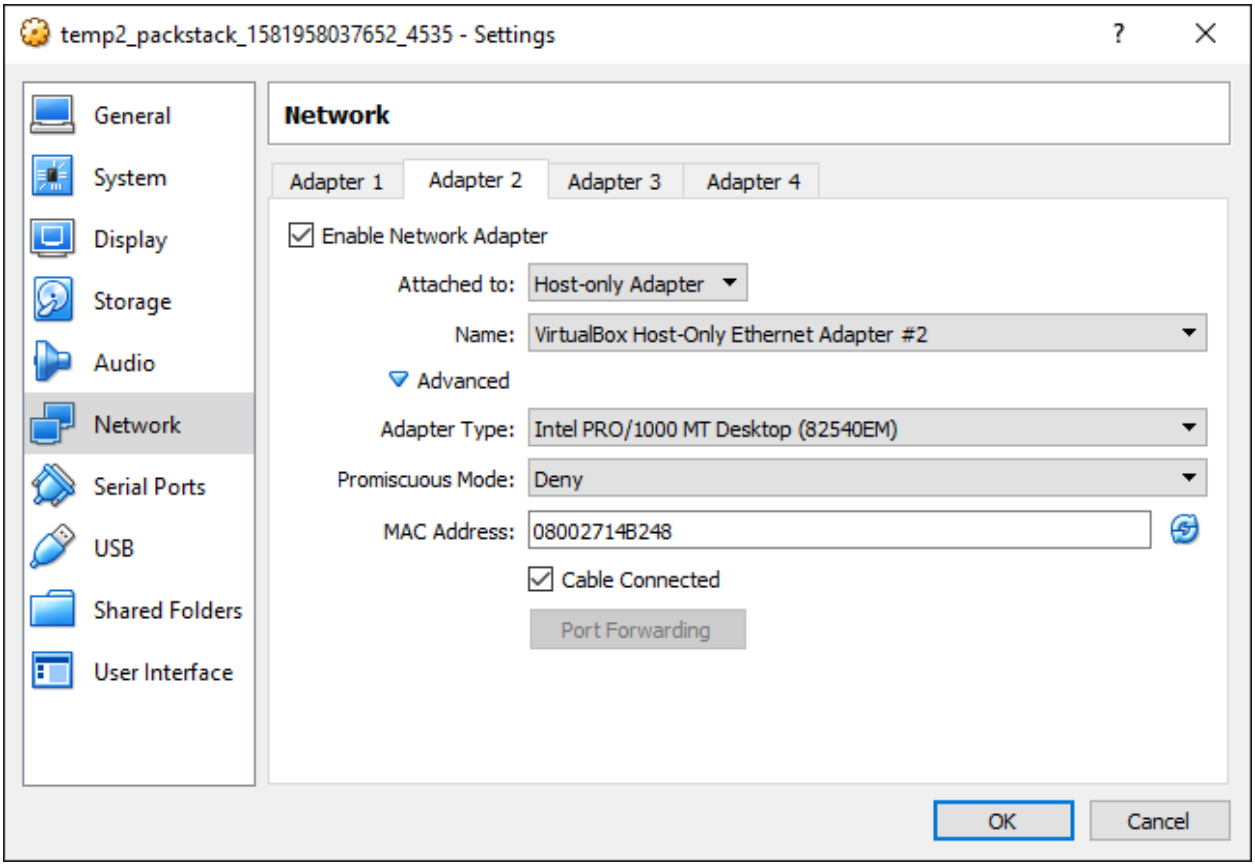


Fig. 24: VirtualBox - Adaptador 2

- En este caso la IP de la interfaz del Controller/Network node es `10.0.1.20`, la IP de la interfaz del compute node 1 es `10.0.1.21`, la IP de la interfaz del compute node 2 es `10.0.1.22`.

Interfaces desde el lado guest

Las IPs de cada interfaz han sido obtenidas desde la consola de las VMs:

1. Adaptador 1 - Interfaz tipo NAT: `enp0s3` o `eth0`
2. Adaptador 2 - Interfaz tipo Host-only Adapter: `enp0s8` o `eth1`

- Controller/Network node:

```
'#' ip addr

...

2: enp0s3: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP group␣
↪default qlen 1000
    link/ether 08:00:27:03:ad:05 brd ff:ff:ff:ff:ff:ff
    inet 10.0.2.15/24 brd 10.0.2.255 scope global noprefixroute dynamic enp0s3
       valid_lft 78655sec preferred_lft 78655sec
    inet6 fe80::a00:27ff:fe03:ad05/64 scope link noprefixroute
       valid_lft forever preferred_lft forever
3: enp0s8: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP group␣
↪default qlen 1000
    link/ether 08:00:27:08:08:b2 brd ff:ff:ff:ff:ff:ff
    inet 10.0.1.20/24 brd 10.0.1.255 scope global noprefixroute enp0s8
       valid_lft forever preferred_lft forever
    inet6 fe80::a00:27ff:fe08:8b2/64 scope link
       valid_lft forever preferred_lft forever

...
```

- Compute node 1:

```
'#' ip addr

...

2: enp0s3: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP group␣
↪default qlen 1000
    link/ether 08:00:27:03:ad:05 brd ff:ff:ff:ff:ff:ff
    inet 10.0.2.15/24 brd 10.0.2.255 scope global noprefixroute dynamic enp0s3
       valid_lft 78414sec preferred_lft 78414sec
    inet6 fe80::a00:27ff:fe03:ad05/64 scope link noprefixroute
       valid_lft forever preferred_lft forever
3: enp0s8: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP group␣
↪default qlen 1000
    link/ether 08:00:27:2b:30:af brd ff:ff:ff:ff:ff:ff
    inet 10.0.1.21/24 brd 10.0.1.255 scope global noprefixroute enp0s8
       valid_lft forever preferred_lft forever
    inet6 fe80::a00:27ff:fe2b:30af/64 scope link
       valid_lft forever preferred_lft forever

...
```

- Compute node 2:

```
'#' ip addr

...

2: enp0s3: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP group␣
↪default qlen 1000
    link/ether 08:00:27:03:ad:05 brd ff:ff:ff:ff:ff:ff
    inet 10.0.2.15/24 brd 10.0.2.255 scope global noprefixroute dynamic enp0s3
       valid_lft 78419sec preferred_lft 78419sec
    inet6 fe80::a00:27ff:fe03:ad05/64 scope link noprefixroute
       valid_lft forever preferred_lft forever
3: enp0s8: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP group␣
↪default qlen 1000
    link/ether 08:00:27:d2:4c:f1 brd ff:ff:ff:ff:ff:ff
    inet 10.0.1.22/24 brd 10.0.1.255 scope global noprefixroute enp0s8
       valid_lft forever preferred_lft forever
    inet6 fe80::a00:27ff:fed2:4cf1/64 scope link
       valid_lft forever preferred_lft forever
```

VM configuration (OpenStack config)

- Interfaz dentro de la red pública de OpenStack:

La interfaz `br-ex` es en realidad un OvS con una interfaz interna que posee una IP dentro de la red pública y solo se encuentra en el Controller/Network node:

```
'#' ip addr

...

5: br-ex: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc noqueue state UNKNOWN␣
↪group default qlen 1000
    link/ether 6a:f2:c8:bc:42:4e brd ff:ff:ff:ff:ff:ff
    inet 172.24.4.1/24 scope global br-ex
       valid_lft forever preferred_lft forever
    inet6 fe80::68f2:c8ff:febc:424e/64 scope link
       valid_lft forever preferred_lft forever

...
```

En este caso, la red pública es `172.24.4.0/24` y `br-ex` tiene la IP `172.24.4.1` asignada.

16.3.9 Anexo 2: Gráficos de red desplegada

Red general total (Host + VMs + Tenant Networks)

Red e interfaces internas (VMs + Tenant Networks)

16.3.10 Anexo 3: Interfaces de red, OvS, bridges

Se presentan los bridges, OvS e interfaces de red de cada VM con el despliegue de OpenStack, solo con los elementos de demo instalados y 2 instancias lanzadas manualmente:

- Controller/Network node:

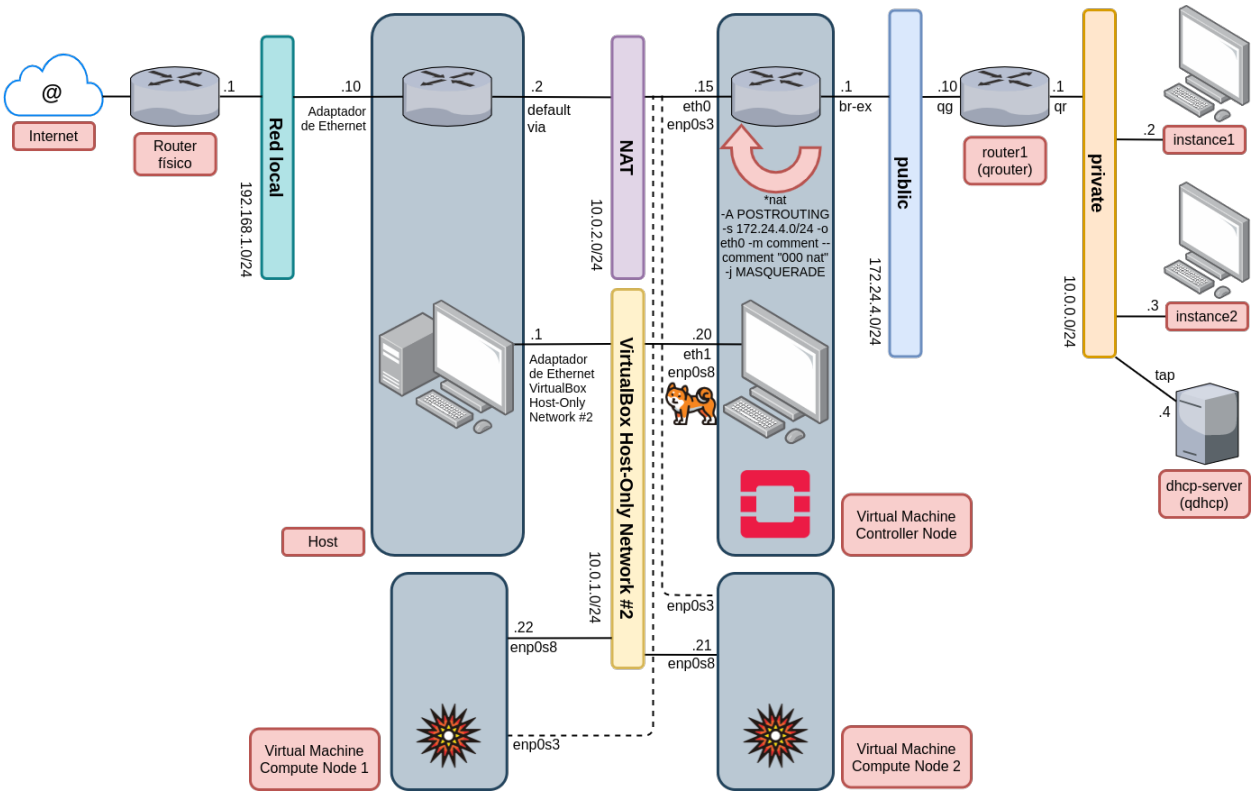


Fig. 25: OpenStack: Host + VMs + Tenant Networks

```
[root@packstack ~]'#' ip address

1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default␣
↪qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
       valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host
       valid_lft forever preferred_lft forever
2: enp0s3: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP group␣
↪default qlen 1000
    link/ether 08:00:27:03:ad:05 brd ff:ff:ff:ff:ff:ff
    inet 10.0.2.15/24 brd 10.0.2.255 scope global noprefixroute dynamic enp0s3
       valid_lft 77961sec preferred_lft 77961sec
    inet6 fe80::a00:27ff:fe03:ad05/64 scope link noprefixroute
       valid_lft forever preferred_lft forever
3: enp0s8: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP group␣
↪default qlen 1000
    link/ether 08:00:27:08:08:b2 brd ff:ff:ff:ff:ff:ff
    inet 10.0.1.20/24 brd 10.0.1.255 scope global noprefixroute enp0s8
       valid_lft forever preferred_lft forever
    inet6 fe80::a00:27ff:fe08:8b2/64 scope link
       valid_lft forever preferred_lft forever
4: ovs-system: <BROADCAST,MULTICAST> mtu 1500 qdisc noop state DOWN group default␣
↪qlen 1000
    link/ether 9e:03:28:91:83:47 brd ff:ff:ff:ff:ff:ff
```

(continues on next page)

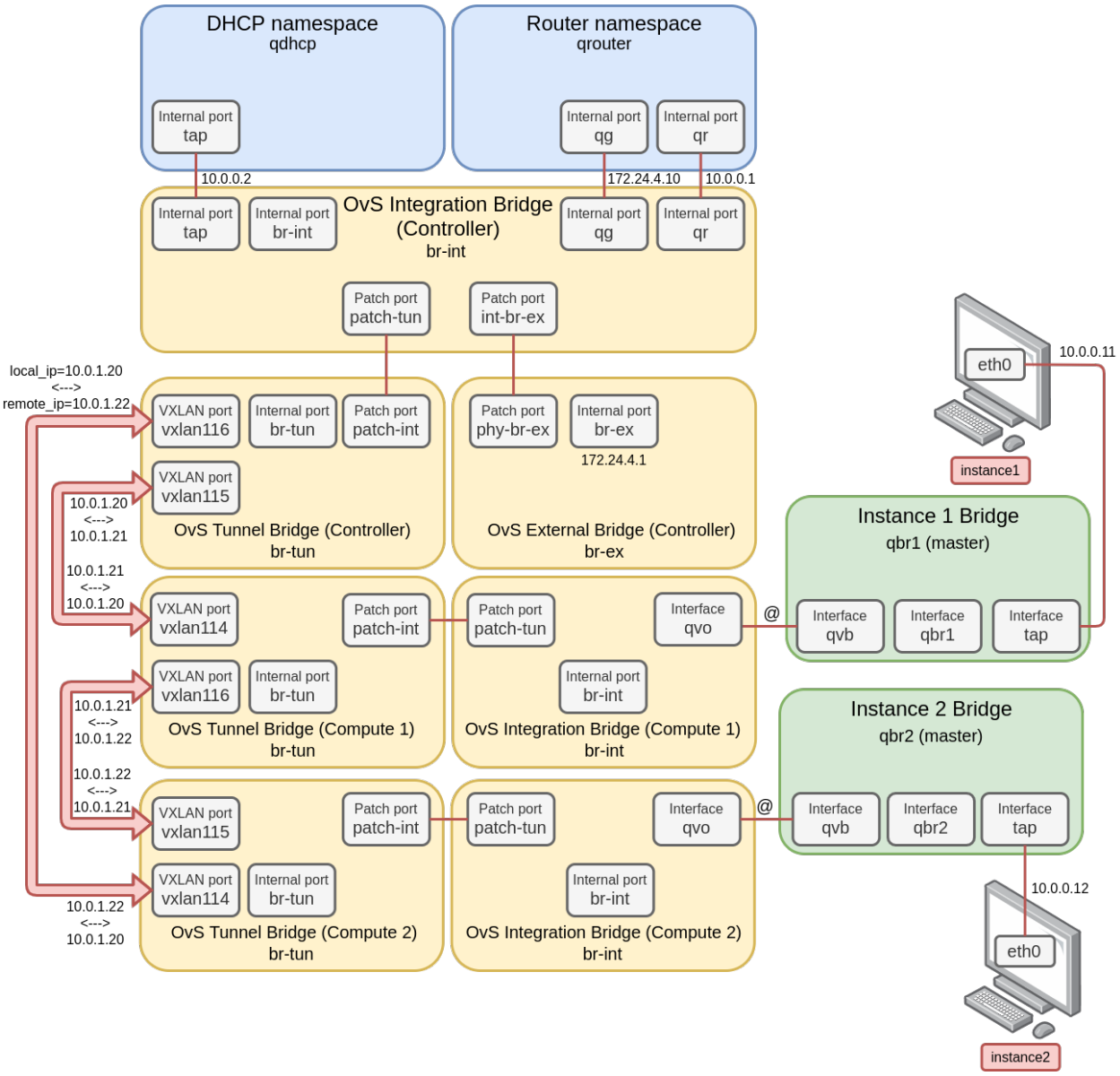


Fig. 26: OpenStack: Internal VMs + Tenant Networks

(continued from previous page)

```
5: br-ex: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc noqueue state UNKNOWN␣
↪group default qlen 1000
    link/ether 32:ff:ae:d9:59:40 brd ff:ff:ff:ff:ff:ff
    inet 172.24.4.1/24 scope global br-ex
       valid_lft forever preferred_lft forever
    inet6 fe80::30ff:aeff:fed9:5940/64 scope link
       valid_lft forever preferred_lft forever
6: br-int: <BROADCAST,MULTICAST> mtu 1500 qdisc noop state DOWN group default qlen␣
↪1000
    link/ether 12:b8:dc:b4:52:4d brd ff:ff:ff:ff:ff:ff
7: br-tun: <BROADCAST,MULTICAST> mtu 1500 qdisc noop state DOWN group default qlen␣
↪1000
    link/ether 6e:9e:09:79:38:41 brd ff:ff:ff:ff:ff:ff
10: vxlan_sys_4789: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 65000 qdisc noqueue master␣
↪ovs-system state UNKNOWN group default qlen 1000
    link/ether 1a:f3:1e:bf:73:fa brd ff:ff:ff:ff:ff:ff
    inet6 fe80::18f3:1eff:febf:73fa/64 scope link
       valid_lft forever preferred_lft forever
```

```
[root@packstack ~]'#' ovs-vsctl show

c82010c9-0b35-4391-97f2-77e0946a1f26
    Manager "ptcp:6640:127.0.0.1"
        is_connected: true
    Bridge br-tun
        Controller "tcp:127.0.0.1:6633"
            is_connected: true
        fail_mode: secure
        Port "vxlan-0a000116"
            Interface "vxlan-0a000116"
                type: vxlan
                options: {df_default="true", egress_pkt_mark="0", in_key=flow, local_
↪ip="10.0.1.20", out_key=flow, remote_ip="10.0.1.22"}
        Port "vxlan-0a000115"
            Interface "vxlan-0a000115"
                type: vxlan
                options: {df_default="true", egress_pkt_mark="0", in_key=flow, local_
↪ip="10.0.1.20", out_key=flow, remote_ip="10.0.1.21"}
        Port br-tun
            Interface br-tun
                type: internal
        Port patch-int
            Interface patch-int
                type: patch
                options: {peer=patch-tun}
    Bridge br-int
        Controller "tcp:127.0.0.1:6633"
            is_connected: true
        fail_mode: secure
        Port int-br-ex
            Interface int-br-ex
                type: patch
                options: {peer=phy-br-ex}
        Port "qg-0dc02115-5b"
            tag: 2
            Interface "qg-0dc02115-5b"
```

(continues on next page)

(continued from previous page)

```
                type: internal
        Port patch-tun
            Interface patch-tun
                type: patch
                options: {peer=patch-int}
        Port "qr-cbc140a5-76"
            tag: 1
            Interface "qr-cbc140a5-76"
                type: internal
        Port br-int
            Interface br-int
                type: internal
        Port "tapb10894b0-97"
            tag: 1
            Interface "tapb10894b0-97"
                type: internal
    Bridge br-ex
        Controller "tcp:127.0.0.1:6633"
            is_connected: true
        fail_mode: secure
        Port phy-br-ex
            Interface phy-br-ex
                type: patch
                options: {peer=int-br-ex}
        Port br-ex
            Interface br-ex
                type: internal
    ovs_version: "2.11.0"
```

- Compute node 1:

```
[root@compute1 ~]'#' ip address

1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default␣
↪qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
       valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host
       valid_lft forever preferred_lft forever
2: enp0s3: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP group␣
↪default qlen 1000
    link/ether 08:00:27:03:ad:05 brd ff:ff:ff:ff:ff:ff
    inet 10.0.2.15/24 brd 10.0.2.255 scope global noprefixroute dynamic enp0s3
       valid_lft 77693sec preferred_lft 77693sec
    inet6 fe80::a00:27ff:fe03:ad05/64 scope link noprefixroute
       valid_lft forever preferred_lft forever
3: enp0s8: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP group␣
↪default qlen 1000
    link/ether 08:00:27:2b:30:af brd ff:ff:ff:ff:ff:ff
    inet 10.0.1.21/24 brd 10.0.1.255 scope global noprefixroute enp0s8
       valid_lft forever preferred_lft forever
    inet6 fe80::a00:27ff:fe2b:30af/64 scope link
       valid_lft forever preferred_lft forever
4: ovs-system: <BROADCAST,MULTICAST> mtu 1500 qdisc noop state DOWN group default␣
↪qlen 1000
    link/ether 12:62:58:1e:15:29 brd ff:ff:ff:ff:ff:ff
```

(continues on next page)

(continued from previous page)

```
5: br-int: <BROADCAST,MULTICAST> mtu 1450 qdisc noop state DOWN group default qlen␣
↪1000
    link/ether fa:31:12:ef:10:47 brd ff:ff:ff:ff:ff:ff
6: br-tun: <BROADCAST,MULTICAST> mtu 1500 qdisc noop state DOWN group default qlen␣
↪1000
    link/ether f2:34:e5:f9:bb:40 brd ff:ff:ff:ff:ff:ff
7: vxlan_sys_4789: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 65000 qdisc noqueue master␣
↪ovs-system state UNKNOWN group default qlen 1000
    link/ether 8e:ea:c0:5b:31:90 brd ff:ff:ff:ff:ff:ff
    inet6 fe80::8cea:c0ff:fe5b:3190/64 scope link
       valid_lft forever preferred_lft forever
8: qbr40be7332-c9: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1450 qdisc noqueue state UP␣
↪group default qlen 1000
    link/ether a6:a3:f6:b8:3e:a1 brd ff:ff:ff:ff:ff:ff
9: qvo40be7332-c9@qvb40be7332-c9: <BROADCAST,MULTICAST,PROMISC,UP,LOWER_UP> mtu 1450␣
↪qdisc noqueue master ovs-system state UP group default qlen 1000
    link/ether 32:16:e4:da:86:52 brd ff:ff:ff:ff:ff:ff
    inet6 fe80::3016:e4ff:feda:8652/64 scope link
       valid_lft forever preferred_lft forever
10: qvb40be7332-c9@qvo40be7332-c9: <BROADCAST,MULTICAST,PROMISC,UP,LOWER_UP> mtu 1450␣
↪qdisc noqueue master qbr40be7332-c9 state UP group default qlen 1000
    link/ether a6:a3:f6:b8:3e:a1 brd ff:ff:ff:ff:ff:ff
    inet6 fe80::a4a3:f6ff:feb8:3ea1/64 scope link
       valid_lft forever preferred_lft forever
11: tap40be7332-c9: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1450 qdisc pfifo_fast␣
↪master qbr40be7332-c9 state UNKNOWN group default qlen 1000
    link/ether fe:16:3e:3e:76:b9 brd ff:ff:ff:ff:ff:ff
    inet6 fe80::fc16:3eff:fe3e:76b9/64 scope link
       valid_lft forever preferred_lft forever
```

```
[root@compute1 ~]'#' ovs-vsctl show

4357a757-09ee-4189-8edf-1c8bc9036850
    Manager "ptcp:6640:127.0.0.1"
        is_connected: true
    Bridge br-int
        Controller "tcp:127.0.0.1:6633"
            is_connected: true
        fail_mode: secure
        Port br-int
            Interface br-int
                type: internal
        Port "qvo40be7332-c9"
            tag: 1
            Interface "qvo40be7332-c9"
        Port patch-tun
            Interface patch-tun
                type: patch
                options: {peer=patch-int}
    Bridge br-tun
        Controller "tcp:127.0.0.1:6633"
            is_connected: true
        fail_mode: secure
        Port patch-int
            Interface patch-int
                type: patch
```

(continues on next page)

(continued from previous page)

```
                options: {peer=patch-tun}
        Port br-tun
            Interface br-tun
                type: internal
        Port "vxlan-0a000114"
            Interface "vxlan-0a000114"
                type: vxlan
                options: {df_default="true", egress_pkt_mark="0", in_key=flow, local_
↪ip="10.0.1.21", out_key=flow, remote_ip="10.0.1.20"}
        Port "vxlan-0a000116"
            Interface "vxlan-0a000116"
                type: vxlan
                options: {df_default="true", egress_pkt_mark="0", in_key=flow, local_
↪ip="10.0.1.21", out_key=flow, remote_ip="10.0.1.22"}
    ovs_version: "2.11.0"
```

```
[root@compute1 ~]'#' brctl show

bridge name         bridge id           STP enabled     interfaces
qbr40be7332-c9      8000.a6a3f6b83ea1   no              qvb40be7332-c9
                                                        tap40be7332-c9
```

- Compute node 2:

```
[root@compute2 ~]'#' ip address

1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default␣
↪qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
       valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host
       valid_lft forever preferred_lft forever
2: enp0s3: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP group␣
↪default qlen 1000
    link/ether 08:00:27:03:ad:05 brd ff:ff:ff:ff:ff:ff
    inet 10.0.2.15/24 brd 10.0.2.255 scope global noprefixroute dynamic enp0s3
       valid_lft 77679sec preferred_lft 77679sec
    inet6 fe80::a00:27ff:fe03:ad05/64 scope link noprefixroute
       valid_lft forever preferred_lft forever
3: enp0s8: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP group␣
↪default qlen 1000
    link/ether 08:00:27:d2:4c:f1 brd ff:ff:ff:ff:ff:ff
    inet 10.0.1.22/24 brd 10.0.1.255 scope global noprefixroute enp0s8
       valid_lft forever preferred_lft forever
    inet6 fe80::a00:27ff:fed2:4cf1/64 scope link
       valid_lft forever preferred_lft forever
4: ovs-system: <BROADCAST,MULTICAST> mtu 1500 qdisc noop state DOWN group default␣
↪qlen 1000
    link/ether 46:5c:23:7e:2b:37 brd ff:ff:ff:ff:ff:ff
5: br-int: <BROADCAST,MULTICAST> mtu 1450 qdisc noop state DOWN group default qlen␣
↪1000
    link/ether fa:2a:a6:6e:31:4f brd ff:ff:ff:ff:ff:ff
6: br-tun: <BROADCAST,MULTICAST> mtu 1500 qdisc noop state DOWN group default qlen␣
↪1000
    link/ether aa:c3:d2:90:32:4b brd ff:ff:ff:ff:ff:ff
```

(continues on next page)

(continued from previous page)

```
7: vxlan_sys_4789: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 65000 qdisc noqueue master␣
↪ovs-system state UNKNOWN group default qlen 1000
    link/ether 6e:9d:de:a6:ec:99 brd ff:ff:ff:ff:ff:ff
    inet6 fe80::6c9d:deff:fea6:ec99/64 scope link
       valid_lft forever preferred_lft forever
8: qbr9a26419b-25: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1450 qdisc noqueue state UP␣
↪group default qlen 1000
    link/ether 46:18:9a:b4:51:0c brd ff:ff:ff:ff:ff:ff
9: qvo9a26419b-25@qvb9a26419b-25: <BROADCAST,MULTICAST,PROMISC,UP,LOWER_UP> mtu 1450␣
↪qdisc noqueue master ovs-system state UP group default qlen 1000
    link/ether c2:5f:0b:e1:1e:09 brd ff:ff:ff:ff:ff:ff
    inet6 fe80::c05f:bff:fee1:1e09/64 scope link
       valid_lft forever preferred_lft forever
10: qvb9a26419b-25@qvo9a26419b-25: <BROADCAST,MULTICAST,PROMISC,UP,LOWER_UP> mtu 1450␣
↪qdisc noqueue master qbr9a26419b-25 state UP group default qlen 1000
    link/ether 46:18:9a:b4:51:0c brd ff:ff:ff:ff:ff:ff
    inet6 fe80::4418:9aff:feb4:510c/64 scope link
       valid_lft forever preferred_lft forever
11: tap9a26419b-25: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1450 qdisc pfifo_fast␣
↪master qbr9a26419b-25 state UNKNOWN group default qlen 1000
    link/ether fe:16:3e:c8:7d:90 brd ff:ff:ff:ff:ff:ff
    inet6 fe80::fc16:3eff:fec8:7d90/64 scope link
       valid_lft forever preferred_lft forever
```

```
[root@compute2 ~]'#' ovs-vsctl show

48edfef1-b892-4471-b143-263034441275
    Manager "ptcp:6640:127.0.0.1"
        is_connected: true
    Bridge br-int
        Controller "tcp:127.0.0.1:6633"
            is_connected: true
        fail_mode: secure
        Port br-int
            Interface br-int
                type: internal
        Port "qvo9a26419b-25"
            tag: 1
            Interface "qvo9a26419b-25"
        Port patch-tun
            Interface patch-tun
                type: patch
                options: {peer=patch-int}
    Bridge br-tun
        Controller "tcp:127.0.0.1:6633"
            is_connected: true
        fail_mode: secure
        Port br-tun
            Interface br-tun
                type: internal
        Port "vxlan-0a000114"
            Interface "vxlan-0a000114"
                type: vxlan
                options: {df_default="true", egress_pkt_mark="0", in_key=flow, local_
↪ip="10.0.1.22", out_key=flow, remote_ip="10.0.1.20"}
        Port "vxlan-0a000115"
```

(continues on next page)

(continued from previous page)

```
            Interface "vxlan-0a000115"
                type: vxlan
                options: {df_default="true", egress_pkt_mark="0", in_key=flow, local_
↪ip="10.0.1.22", out_key=flow, remote_ip="10.0.1.21"}
        Port patch-int
            Interface patch-int
                type: patch
                options: {peer=patch-tun}
    ovs_version: "2.11.0"
```

```
[root@compute2 ~]'#' brctl show

bridge name         bridge id               STP enabled     interfaces
qbr9a26419b-25      8000.46189ab4510c       no              qvb9a26419b-25
                                                            tap9a26419b-25
```

16.3.11 Anexo 4: Pruebas de tráfico

Comunicación entre instancias de Compute nodes distintos

Desde la instancia 1 (`10.0.0.2`) ubicada en el nodo Compute 1 (Hypervisor 1) haremos ping a la instancia 2 (`10.0.0.3`) ubicada en el nodo Compute 2 (Hypervisor 2):

```
$ ping 10.0.0.3
```

- ¿Por qué nodos viaja el tráfico ICMP?

Hacemos uso de `tcpdump` en cada nodo (o VM) en busca de tráfico ICMP y obtenemos los siguientes resultados:

1. Controller/Network Node:

```
[root@packstack ~]'#' tcpdump -i any icmp

tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on any, link-type LINUX_SLL (Linux cooked), capture size 262144 bytes
^C
0 packets captured
0 packets received by filter
0 packets dropped by kernel
```

2. Compute Node 1:

```
[root@compute1 ~]'#' tcpdump -i any icmp

tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on any, link-type LINUX_SLL (Linux cooked), capture size 262144 bytes
23:28:22.640464 IP 10.0.0.2 > 10.0.0.3: ICMP echo request, id 43265, seq 0, length 64
23:28:22.640488 IP 10.0.0.2 > 10.0.0.3: ICMP echo request, id 43265, seq 0, length 64
23:28:22.640489 IP 10.0.0.2 > 10.0.0.3: ICMP echo request, id 43265, seq 0, length 64
23:28:22.640593 IP 10.0.0.2 > 10.0.0.3: ICMP echo request, id 43265, seq 0, length 64
23:28:22.643021 IP 10.0.0.3 > 10.0.0.2: ICMP echo reply, id 43265, seq 0, length 64
23:28:22.643125 IP 10.0.0.3 > 10.0.0.2: ICMP echo reply, id 43265, seq 0, length 64
23:28:22.643127 IP 10.0.0.3 > 10.0.0.2: ICMP echo reply, id 43265, seq 0, length 64
23:28:22.643138 IP 10.0.0.3 > 10.0.0.2: ICMP echo reply, id 43265, seq 0, length 64
23:28:23.641731 IP 10.0.0.2 > 10.0.0.3: ICMP echo request, id 43265, seq 1, length 64
```

(continues on next page)

(continued from previous page)

```
23:28:23.641745 IP 10.0.0.2 > 10.0.0.3: ICMP echo request, id 43265, seq 1, length 64
...
```

3. Compute Node 2:

```
[root@compute2 ~]'#' tcpdump -i any icmp

tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on any, link-type LINUX_SLL (Linux cooked), capture size 262144 bytes
23:28:22.639348 IP 10.0.0.2 > 10.0.0.3: ICMP echo request, id 43265, seq 0, length 64
23:28:22.640678 IP 10.0.0.2 > 10.0.0.3: ICMP echo request, id 43265, seq 0, length 64
23:28:22.640681 IP 10.0.0.2 > 10.0.0.3: ICMP echo request, id 43265, seq 0, length 64
23:28:22.640695 IP 10.0.0.2 > 10.0.0.3: ICMP echo request, id 43265, seq 0, length 64
23:28:22.641265 IP 10.0.0.3 > 10.0.0.2: ICMP echo reply, id 43265, seq 0, length 64
23:28:22.641275 IP 10.0.0.3 > 10.0.0.2: ICMP echo reply, id 43265, seq 0, length 64
23:28:22.641276 IP 10.0.0.3 > 10.0.0.2: ICMP echo reply, id 43265, seq 0, length 64
23:28:22.641419 IP 10.0.0.3 > 10.0.0.2: ICMP echo reply, id 43265, seq 0, length 64
23:28:23.640486 IP 10.0.0.2 > 10.0.0.3: ICMP echo request, id 43265, seq 1, length 64
23:28:23.640509 IP 10.0.0.2 > 10.0.0.3: ICMP echo request, id 43265, seq 1, length 64
...
```

Como se ha comprobado, por el Controller/Network node no cruza tráfico de las instancias. Por los nodos Compute 1 y Compute 2 pasan 4 request y 4 replies cada uno para una misma secuencia (`seq`) de paquete ICMP. Esto nos hace pensar que el paquete ICMP está atravesando 4 interfaces distintas en cada Compute node.

- ¿Qué interfaces atraviesa el tráfico para viajar de instancia a instancia?

Sabemos que el tráfico entre instancias de compute nodes distintos no pasa por el Controller/Network node. Ahora veamos qué interfaces atraviesa el tráfico en Compute 1 y Compute 2:

1. Compute Node 1:

```
[root@compute1 ~]'#' tcpdump -i tap40be7332-c9 icmp
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on tap40be7332-c9, link-type EN10MB (Ethernet), capture size 262144 bytes
23:48:39.283358 IP 10.0.0.2 > 10.0.0.3: ICMP echo request, id 44033, seq 5, length 64
23:48:39.284066 IP 10.0.0.3 > 10.0.0.2: ICMP echo reply, id 44033, seq 5, length 64
...

[root@compute1 ~]'#' tcpdump -i qvb40be7332-c9 icmp
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on qvb40be7332-c9, link-type EN10MB (Ethernet), capture size 262144 bytes
23:48:48.285237 IP 10.0.0.2 > 10.0.0.3: ICMP echo request, id 44033, seq 14, length 64
23:48:48.285952 IP 10.0.0.3 > 10.0.0.2: ICMP echo reply, id 44033, seq 14, length 64
...

[root@compute1 ~]'#' tcpdump -i qbr40be7332-c9 icmp
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on qbr40be7332-c9, link-type EN10MB (Ethernet), capture size 262144 bytes
23:49:02.302033 IP 10.0.0.2 > 10.0.0.3: ICMP echo request, id 44033, seq 28, length 64
23:49:02.302889 IP 10.0.0.3 > 10.0.0.2: ICMP echo reply, id 44033, seq 28, length 64
...

[root@compute1 ~]'#' tcpdump -i qvo40be7332-c9 icmp
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on qvo40be7332-c9, link-type EN10MB (Ethernet), capture size 262144 bytes
23:49:19.305016 IP 10.0.0.2 > 10.0.0.3: ICMP echo request, id 44033, seq 45, length 64
```

(continues on next page)

(continued from previous page)

```
23:49:19.305850 IP 10.0.0.3 > 10.0.0.2: ICMP echo reply, id 44033, seq 45, length 64
...

[root@compute1 ~]# tcpdump -i vxlan_sys_4789 icmp
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on vxlan_sys_4789, link-type EN10MB (Ethernet), capture size 262144 bytes
01:17:41.602490 IP 10.0.0.2 > 10.0.0.3: ICMP echo request, id 44033, seq 50, length 64
01:17:41.603217 IP 10.0.0.3 > 10.0.0.2: ICMP echo reply, id 44033, seq 50, length 64
...

[root@compute1 ~]# tcpdump -i enp0s3 icmp
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on enp0s3, link-type EN10MB (Ethernet), capture size 262144 bytes
^C
0 packets captured
0 packets received by filter
0 packets dropped by kernel

[root@compute1 ~]'#' tcpdump -i enp0s8 icmp
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on enp0s8, link-type EN10MB (Ethernet), capture size 262144 bytes
^C
0 packets captured
0 packets received by filter
0 packets dropped by kernel
```

Vemos que, en realidad, el tráfico ICMP atraviesa 5 interfaces del Compute Node 1: 3 interfaces del Linux Bridge, 1 interfaz del OVS Integration bridge que se conecta con este Linux bridge y 1 la interfaz VXLAN.

2. Compute Node 2:

```
[root@compute2 ~]'#' tcpdump -i qvo9a26419b-25 icmp
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on qvo9a26419b-25, link-type EN10MB (Ethernet), capture size 262144 bytes
23:56:59.857009 IP 10.0.0.2 > 10.0.0.3: ICMP echo request, id 44289, seq 6, length 64
23:56:59.857879 IP 10.0.0.3 > 10.0.0.2: ICMP echo reply, id 44289, seq 6, length 64
...

[root@compute2 ~]'#' tcpdump -i qvb9a26419b-25 icmp
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on qvb9a26419b-25, link-type EN10MB (Ethernet), capture size 262144 bytes
23:57:06.860267 IP 10.0.0.2 > 10.0.0.3: ICMP echo request, id 44289, seq 13, length 64
23:57:06.861080 IP 10.0.0.3 > 10.0.0.2: ICMP echo reply, id 44289, seq 13, length 64
...

[root@compute2 ~]'#' tcpdump -i qbr9a26419b-25 icmp
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on qbr9a26419b-25, link-type EN10MB (Ethernet), capture size 262144 bytes
23:57:12.861566 IP 10.0.0.2 > 10.0.0.3: ICMP echo request, id 44289, seq 19, length 64
23:57:12.862064 IP 10.0.0.3 > 10.0.0.2: ICMP echo reply, id 44289, seq 19, length 64
...

[root@compute2 ~]'#' tcpdump -i tap9a26419b-25 icmp
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on tap9a26419b-25, link-type EN10MB (Ethernet), capture size 262144 bytes
23:57:19.863279 IP 10.0.0.2 > 10.0.0.3: ICMP echo request, id 44289, seq 26, length 64
23:57:19.863987 IP 10.0.0.3 > 10.0.0.2: ICMP echo reply, id 44289, seq 26, length 64
```

(continues on next page)

(continued from previous page)

```
...

[root@compute2 ~]'#' tcpdump -i vxlan_sys_4789 icmp
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on vxlan_sys_4789, link-type EN10MB (Ethernet), capture size 262144 bytes
01:26:21.977054 IP 10.0.0.2 > 10.0.0.3: ICMP echo request, id 44289, seq 30, length 64
01:26:21.977938 IP 10.0.0.3 > 10.0.0.2: ICMP echo reply, id 44289, seq 30, length 64
...

[root@compute2 ~]'#' tcpdump -i enp0s3 icmp
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on enp0s3, link-type EN10MB (Ethernet), capture size 262144 bytes
^C
0 packets captured
0 packets received by filter
0 packets dropped by kernel

[root@compute2 ~]'#' tcpdump -i enp0s8 icmp
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on enp0s8, link-type EN10MB (Ethernet), capture size 262144 bytes
^C
0 packets captured
0 packets received by filter
0 packets dropped by kernel
```

Vemos que, en realidad, el tráfico ICMP atraviesa 5 interfaces del Compute Node 2: 3 interfaces del Linux Bridge, 1 interfaz del OVS Integration bridge que se conecta con este Linux bridge y 1 la interfaz VXLAN.

Important: Hemos comprobado que existe una ruta de comunicación separada para el tráfico entre instancias que pertenecen a distintos Compute Nodes. Esta se llama **Data Path**. El Controller/Network Node no recibe tráfico de esta comunicación entre instancias.

Comunicación de instancia con Internet

Desde la instancia 1 (`10.0.0.2`) ubicada en el nodo Compute 1 (Hypervisor 1) haremos ping a la IP `8.8.8.8`:

- ¿Por qué nodos viaja el tráfico ICMP?

Hacemos uso de `tcpdump` en el nodo Controller/Network y Compute 1 en busca de tráfico ICMP y obtenemos los siguientes resultados:

1. Controller/Network Node:

```
[root@packstack ~]'#' tcpdump -i any icmp

tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on any, link-type LINUX_SLL (Linux cooked), capture size 262144 bytes
00:50:54.291695 IP 10.0.0.2 > dns.google: ICMP echo request, id 45057, seq 1, length␣
↪64
00:50:54.291723 IP 172.24.4.10 > dns.google: ICMP echo request, id 45057, seq 1,␣
↪length 64
00:50:54.291731 IP packstack > dns.google: ICMP echo request, id 45057, seq 1, length␣
↪64
00:50:54.373374 IP dns.google > packstack: ICMP echo reply, id 45057, seq 1, length 64
```

(continues on next page)

(continued from previous page)

```
00:50:54.373401 IP dns.google > 172.24.4.10: ICMP echo reply, id 45057, seq 1, length␣
↪64
00:50:54.373445 IP dns.google > 10.0.0.2: ICMP echo reply, id 45057, seq 1, length 64
00:50:55.293234 IP 10.0.0.2 > dns.google: ICMP echo request, id 45057, seq 2, length␣
↪64
00:50:55.293261 IP 172.24.4.10 > dns.google: ICMP echo request, id 45057, seq 2,␣
↪length 64
...
```

2. Compute Node 1:

```
[root@compute1 ~]'#' tcpdump -i any icmp

tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on any, link-type LINUX_SLL (Linux cooked), capture size 262144 bytes
00:51:04.317637 IP 10.0.0.2 > dns.google: ICMP echo request, id 45057, seq 11, length␣
↪64
00:51:04.317663 IP 10.0.0.2 > dns.google: ICMP echo request, id 45057, seq 11, length␣
↪64
00:51:04.317665 IP 10.0.0.2 > dns.google: ICMP echo request, id 45057, seq 11, length␣
↪64
00:51:04.317670 IP 10.0.0.2 > dns.google: ICMP echo request, id 45057, seq 11, length␣
↪64
00:51:04.399731 IP dns.google > 10.0.0.2: ICMP echo reply, id 45057, seq 11, length 64
00:51:04.399739 IP dns.google > 10.0.0.2: ICMP echo reply, id 45057, seq 11, length 64
00:51:04.399740 IP dns.google > 10.0.0.2: ICMP echo reply, id 45057, seq 11, length 64
00:51:04.399755 IP dns.google > 10.0.0.2: ICMP echo reply, id 45057, seq 11, length 64
00:51:05.324080 IP 10.0.0.2 > dns.google: ICMP echo request, id 45057, seq 12, length␣
↪64
00:51:05.324105 IP 10.0.0.2 > dns.google: ICMP echo request, id 45057, seq 12, length␣
↪64
...
```

Por el nodo Controller/Network pasan 3 request y 3 replies de tráfico ICMP, mientras que por Compute 1 pasan 4 request y 4 replies, para una misma secuencia (`seq`) de paquete ICMP.

- ¿Qué interfaces atraviesa el tráfico para viajar de la instancia a Internet?

Ahora veamos qué interfaces atraviesa el tráfico en Controller/Network node y Compute 1:

1. Controller/Network Node:

```
[root@packstack ~]'#' tcpdump -i enp0s3 icmp

tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on enp0s3, link-type EN10MB (Ethernet), capture size 262144 bytes
01:04:48.753587 IP packstack > dns.google: ICMP echo request, id 45569, seq 6, length␣
↪64
01:04:48.829003 IP dns.google > packstack: ICMP echo reply, id 45569, seq 6, length 64
...

[root@packstack ~]'#' tcpdump -i enp0s8 icmp

tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on enp0s8, link-type EN10MB (Ethernet), capture size 262144 bytes
^C
0 packets captured
0 packets received by filter
```

(continues on next page)

(continued from previous page)

```
0 packets dropped by kernel

[root@packstack ~]'#' tcpdump -i br-ex icmp

tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on br-ex, link-type EN10MB (Ethernet), capture size 262144 bytes
01:05:06.838046 IP 172.24.4.10 > dns.google: ICMP echo request, id 45569, seq 24,␣
↪length 64
01:05:06.911353 IP dns.google > 172.24.4.10: ICMP echo reply, id 45569, seq 24,␣
↪length 64
...

[root@packstack ~]'#' tcpdump -i vxlan_sys_4789 icmp

tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on vxlan_sys_4789, link-type EN10MB (Ethernet), capture size 262144 bytes
01:05:13.858767 IP 10.0.0.2 > dns.google: ICMP echo request, id 45569, seq 31, length␣
↪64
01:05:13.936364 IP dns.google > 10.0.0.2: ICMP echo reply, id 45569, seq 31, length 64
...
```

Comprobamos que el tráfico ICMP atraviesa 3 interfaces del Controller/Network Node: `enp0s3`, `enp0s8` y 1 la interfaz VXLAN.

2. Compute Node 1:

```
[root@compute1 ~]'#' tcpdump -i tap40be7332-c9 icmp
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on tap40be7332-c9, link-type EN10MB (Ethernet), capture size 262144 bytes
01:10:12.675236 IP 10.0.0.2 > dns.google: ICMP echo request, id 46081, seq 15, length␣
↪64
01:10:12.753360 IP dns.google > 10.0.0.2: ICMP echo reply, id 46081, seq 15, length 64

[root@compute1 ~]'#' tcpdump -i qvb40be7332-c9 icmp
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on qvb40be7332-c9, link-type EN10MB (Ethernet), capture size 262144 bytes
01:10:25.676923 IP 10.0.0.2 > dns.google: ICMP echo request, id 46081, seq 28, length␣
↪64
01:10:25.753336 IP dns.google > 10.0.0.2: ICMP echo reply, id 46081, seq 28, length 64

[root@compute1 ~]'#' tcpdump -i qbr40be7332-c9 icmp
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on qbr40be7332-c9, link-type EN10MB (Ethernet), capture size 262144 bytes
01:10:40.680569 IP 10.0.0.2 > dns.google: ICMP echo request, id 46081, seq 43, length␣
↪64
01:10:40.756072 IP dns.google > 10.0.0.2: ICMP echo reply, id 46081, seq 43, length 64

[root@compute1 ~]'#' tcpdump -i qvo40be7332-c9 icmp
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on qvo40be7332-c9, link-type EN10MB (Ethernet), capture size 262144 bytes
01:11:03.686021 IP 10.0.0.2 > dns.google: ICMP echo request, id 46081, seq 66, length␣
↪64
01:11:03.763526 IP dns.google > 10.0.0.2: ICMP echo reply, id 46081, seq 66, length 64

[root@compute1 ~]'#' tcpdump -i vxlan_sys_4789 icmp
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on vxlan_sys_4789, link-type EN10MB (Ethernet), capture size 262144 bytes
```

(continues on next page)

(continued from previous page)

```
01:11:42.705281 IP 10.0.0.2 > dns.google: ICMP echo request, id 46081, seq 105,␣
↪length 64
01:11:42.783655 IP dns.google > 10.0.0.2: ICMP echo reply, id 46081, seq 105, length␣
↪64

[root@compute1 ~]'#' tcpdump -i enp0s3 icmp
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on enp0s3, link-type EN10MB (Ethernet), capture size 262144 bytes
^C
0 packets captured
0 packets received by filter
0 packets dropped by kernel

[root@compute1 ~]'#' tcpdump -i enp0s8 icmp
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on enp0s8, link-type EN10MB (Ethernet), capture size 262144 bytes
^C
0 packets captured
0 packets received by filter
0 packets dropped by kernel
```

Vemos que, en realidad, el tráfico ICMP atraviesa 5 interfaces del Compute Node 1: 3 interfaces del Linux Bridge, 1 interfaz del OVS Integration bridge que se conecta con este Linux bridge y 1 la interfaz VXLAN.

16.3.12 Anexo 5: Archivo answers.txt de packstack generado

```
'#' cat /home/vagrant/packstack-answers-20200218-155331.txt

[general]

# Path to a public key to install on servers. If a usable key has not
# been installed on the remote servers, the user is prompted for a
# password and this key is installed so the password will not be
# required again.
CONFIG_SSH_KEY=/home/vagrant/.ssh/id_rsa.pub

# Default password to be used everywhere (overridden by passwords set
# for individual services or users).
CONFIG_DEFAULT_PASSWORD=

# The amount of service workers/threads to use for each service.
# Useful to tweak when you have memory constraints. Defaults to the
# amount of cores on the system.
CONFIG_SERVICE_WORKERS=%{::processorcount}

# Specify 'y' to install MariaDB. ['y', 'n']
CONFIG_MARIADB_INSTALL=y

# Specify 'y' to install OpenStack Image Service (glance). ['y', 'n']
CONFIG_GLANCE_INSTALL=y

# Specify 'y' to install OpenStack Block Storage (cinder). ['y', 'n']
CONFIG_CINDER_INSTALL=y

# Specify 'y' to install OpenStack Shared File System (manila). ['y',
```

(continues on next page)

(continued from previous page)

```
# 'n']
CONFIG_MANILA_INSTALL=n

# Specify 'y' to install OpenStack Compute (nova). ['y', 'n']
CONFIG_NOVA_INSTALL=y

# Specify 'y' to install OpenStack Networking (neutron) ['y']
CONFIG_NEUTRON_INSTALL=y

# Specify 'y' to install OpenStack Dashboard (horizon). ['y', 'n']
CONFIG_HORIZON_INSTALL=y

# Specify 'y' to install OpenStack Object Storage (swift). ['y', 'n']
CONFIG_SWIFT_INSTALL=y

# Specify 'y' to install OpenStack Metering (ceilometer). Note this
# will also automatically install gnocchi service and configures it as
# the metrics backend. ['y', 'n']
CONFIG_CEILOMETER_INSTALL=y

# Specify 'y' to install OpenStack Telemetry Alarming (Aodh). Note
# Aodh requires Ceilometer to be installed as well. ['y', 'n']
CONFIG_AODH_INSTALL=y

# Specify 'y' to install OpenStack Events Service (panko). ['y', 'n']
CONFIG_PANKO_INSTALL=n

# Specify 'y' to install OpenStack Data Processing (sahara). In case
# of sahara installation packstack also installs heat.['y', 'n']
CONFIG_SAHARA_INSTALL=n

# Specify 'y' to install OpenStack Orchestration (heat). ['y', 'n']
CONFIG_HEAT_INSTALL=y

# Specify 'y' to install OpenStack Container Infrastructure
# Management Service (magnum). ['y', 'n']
CONFIG_MAGNUM_INSTALL=n

# Specify 'y' to install OpenStack Database (trove) ['y', 'n']
CONFIG_TROVE_INSTALL=n

# Specify 'y' to install OpenStack Bare Metal Provisioning (ironic).
# ['y', 'n']
CONFIG_IRONIC_INSTALL=n

# Specify 'y' to install the OpenStack Client packages (command-line
# tools). An admin "rc" file will also be installed. ['y', 'n']
CONFIG_CLIENT_INSTALL=y

# Comma-separated list of NTP servers. Leave plain if Packstack
# should not install ntpd on instances.
CONFIG_NTP_SERVERS=

# Comma-separated list of servers to be excluded from the
# installation. This is helpful if you are running Packstack a second
# time with the same answer file and do not want Packstack to
# overwrite these server's configurations. Leave empty if you do not
```

(continues on next page)

(continued from previous page)

```
# need to exclude any servers.
EXCLUDE_SERVERS=

# Specify 'y' if you want to run OpenStack services in debug mode;
# otherwise, specify 'n'. ['y', 'n']
CONFIG_DEBUG_MODE=n

# Server on which to install OpenStack services specific to the
# controller role (for example, API servers or dashboard).
CONFIG_CONTROLLER_HOST=10.0.1.20

# List the servers on which to install the Compute service.
CONFIG_COMPUTE_HOSTS=10.0.1.21,10.0.1.22

# List of servers on which to install the network service such as
# Compute networking (nova network) or OpenStack Networking (neutron).
CONFIG_NETWORK_HOSTS=10.0.1.20

# Specify 'y' if you want to use VMware vCenter as hypervisor and
# storage; otherwise, specify 'n'. ['y', 'n']
CONFIG_VMWARE_BACKEND=n

# Specify 'y' if you want to use unsupported parameters. This should
# be used only if you know what you are doing. Issues caused by using
# unsupported options will not be fixed before the next major release.
# ['y', 'n']
CONFIG_UNSUPPORTED=n

# Specify 'y' if you want to use subnet addresses (in CIDR format)
# instead of interface names in following options:
# CONFIG_NEUTRON_OVS_BRIDGE_IFACES,
# CONFIG_NEUTRON_LB_INTERFACE_MAPPINGS, CONFIG_NEUTRON_OVS_TUNNEL_IF.
# This is useful for cases when interface names are not same on all
# installation hosts.
CONFIG_USE_SUBNETS=n

# IP address of the VMware vCenter server.
CONFIG_VCENTER_HOST=

# User name for VMware vCenter server authentication.
CONFIG_VCENTER_USER=

# Password for VMware vCenter server authentication.
CONFIG_VCENTER_PASSWORD=

# Comma separated list of names of the VMware vCenter clusters. Note:
# if multiple clusters are specified each one is mapped to one
# compute, otherwise all computes are mapped to same cluster.
CONFIG_VCENTER_CLUSTER_NAMES=

# (Unsupported!) Server on which to install OpenStack services
# specific to storage servers such as Image or Block Storage services.
CONFIG_STORAGE_HOST=10.0.2.15

# (Unsupported!) Server on which to install OpenStack services
# specific to OpenStack Data Processing (sahara).
CONFIG_SAHARA_HOST=10.0.2.15
```

(continues on next page)

(continued from previous page)

```
# Comma-separated list of URLs for any additional yum repositories,
# to use for installation.
CONFIG_REPO=

# Specify 'y' to enable the RDO testing repository. ['y', 'n']
CONFIG_ENABLE_RDO_TESTING=n

# To subscribe each server with Red Hat Subscription Manager, include
# this with CONFIG_RH_PW.
CONFIG_RH_USER=

# To subscribe each server to receive updates from a Satellite
# server, provide the URL of the Satellite server. You must also
# provide a user name (CONFIG_SATELLITE_USERNAME) and password
# (CONFIG_SATELLITE_PASSWORD) or an access key (CONFIG_SATELLITE_AKEY)
# for authentication.
CONFIG_SATELLITE_URL=

# Specify a Satellite 6 Server to register to. If not specified,
# Packstack will register the system to the Red Hat server. When this
# option is specified, you also need to set the Satellite 6
# organization (CONFIG_RH_SAT6_ORG) and an activation key
# (CONFIG_RH_SAT6_KEY).
CONFIG_RH_SAT6_SERVER=

# To subscribe each server with Red Hat Subscription Manager, include
# this with CONFIG_RH_USER.
CONFIG_RH_PW=

# Specify 'y' to enable RHEL optional repositories. ['y', 'n']
CONFIG_RH_OPTIONAL=y

# HTTP proxy to use with Red Hat Subscription Manager.
CONFIG_RH_PROXY=

# Specify a Satellite 6 Server organization to use when registering
# the system.
CONFIG_RH_SAT6_ORG=

# Specify a Satellite 6 Server activation key to use when registering
# the system.
CONFIG_RH_SAT6_KEY=

# Port to use for Red Hat Subscription Manager's HTTP proxy.
CONFIG_RH_PROXY_PORT=

# User name to use for Red Hat Subscription Manager's HTTP proxy.
CONFIG_RH_PROXY_USER=

# Password to use for Red Hat Subscription Manager's HTTP proxy.
CONFIG_RH_PROXY_PW=

# User name to authenticate with the RHN Satellite server; if you
# intend to use an access key for Satellite authentication, leave this
# blank.
CONFIG_SATELLITE_USER=
```

(continues on next page)

(continued from previous page)

```
# Password to authenticate with the RHN Satellite server; if you
# intend to use an access key for Satellite authentication, leave this
# blank.
CONFIG_SATELLITE_PW=

# Access key for the Satellite server; if you intend to use a user
# name and password for Satellite authentication, leave this blank.
CONFIG_SATELLITE_AKEY=

# Certificate path or URL of the certificate authority to verify that
# the connection with the Satellite server is secure. If you are not
# using Satellite in your deployment, leave this blank.
CONFIG_SATELLITE_CACERT=

# Profile name that should be used as an identifier for the system in
# RHN Satellite (if required).
CONFIG_SATELLITE_PROFILE=

# Comma-separated list of flags passed to the rhnreg_ks command.
# Valid flags are: novirtinfo, norhnsd, nopackages ['novirtinfo',
# 'norhnsd', 'nopackages']
CONFIG_SATELLITE_FLAGS=

# HTTP proxy to use when connecting to the RHN Satellite server (if
# required).
CONFIG_SATELLITE_PROXY=

# User name to authenticate with the Satellite-server HTTP proxy.
CONFIG_SATELLITE_PROXY_USER=

# User password to authenticate with the Satellite-server HTTP proxy.
CONFIG_SATELLITE_PROXY_PW=

# Specify filepath for CA cert file. If CONFIG_SSL_CACERT_SELFSIGN is
# set to 'n' it has to be preexisting file.
CONFIG_SSL_CACERT_FILE=/etc/pki/tls/certs/selfcert.crt

# Specify filepath for CA cert key file. If
# CONFIG_SSL_CACERT_SELFSIGN is set to 'n' it has to be preexisting
# file.
CONFIG_SSL_CACERT_KEY_FILE=/etc/pki/tls/private/selfkey.key

# Enter the path to use to store generated SSL certificates in.
CONFIG_SSL_CERT_DIR=~/packstackca/

# Specify 'y' if you want Packstack to pregenerate the CA
# Certificate.
CONFIG_SSL_CACERT_SELFSIGN=y

# Enter the ssl certificates subject country.
CONFIG_SSL_CERT_SUBJECT_C=--

# Enter the ssl certificates subject state.
CONFIG_SSL_CERT_SUBJECT_ST=State

# Enter the ssl certificates subject location.
```

(continues on next page)

(continued from previous page)

```
CONFIG_SSL_CERT_SUBJECT_L=City

# Enter the ssl certificates subject organization.
CONFIG_SSL_CERT_SUBJECT_O=openstack

# Enter the ssl certificates subject organizational unit.
CONFIG_SSL_CERT_SUBJECT_OU=packstack

# Enter the ssl certificates subject common name.
CONFIG_SSL_CERT_SUBJECT_CN=packstack

CONFIG_SSL_CERT_SUBJECT_MAIL=admin@packstack

# Service to be used as the AMQP broker. Allowed values are: rabbitmq
# ['rabbitmq']
CONFIG_AMQP_BACKEND=rabbitmq

# IP address of the server on which to install the AMQP service.
CONFIG_AMQP_HOST=10.0.1.20

# Specify 'y' to enable SSL for the AMQP service. ['y', 'n']
CONFIG_AMQP_ENABLE_SSL=n

# Specify 'y' to enable authentication for the AMQP service. ['y',
# 'n']
CONFIG_AMQP_ENABLE_AUTH=n

# Password for the NSS certificate database of the AMQP service.
CONFIG_AMQP_NSS_CERTDB_PW=PW_PLACEHOLDER

# User for AMQP authentication.
CONFIG_AMQP_AUTH_USER=amqp_user

# Password for AMQP authentication.
CONFIG_AMQP_AUTH_PASSWORD=PW_PLACEHOLDER

# IP address of the server on which to install MariaDB. If a MariaDB
# installation was not specified in CONFIG_MARIADB_INSTALL, specify
# the IP address of an existing database server (a MariaDB cluster can
# also be specified).
CONFIG_MARIADB_HOST=10.0.1.20

# User name for the MariaDB administrative user.
CONFIG_MARIADB_USER=root

# Password for the MariaDB administrative user.
CONFIG_MARIADB_PW=8941bfccbd3645d1

# Password to use for the Identity service (keystone) to access the
# database.
CONFIG_KEYSTONE_DB_PW=1e448e39f1dc4906

# Enter y if cron job for removing soft deleted DB rows should be
# created.
CONFIG_KEYSTONE_DB_PURGE_ENABLE=True

# Default region name to use when creating tenants in the Identity
```

(continues on next page)

(continued from previous page)

```
# service.
CONFIG_KEYSTONE_REGION=RegionOne

# Token to use for the Identity service API.
CONFIG_KEYSTONE_ADMIN_TOKEN=329f0cccb90e44afb9e1d597e60660b2

# Email address for the Identity service 'admin' user.  Defaults to
CONFIG_KEYSTONE_ADMIN_EMAIL=root@localhost

# User name for the Identity service 'admin' user.  Defaults to
# 'admin'.
CONFIG_KEYSTONE_ADMIN_USERNAME=admin

# Password to use for the Identity service 'admin' user.
CONFIG_KEYSTONE_ADMIN_PW=openstack

# Password to use for the Identity service 'demo' user.
CONFIG_KEYSTONE_DEMO_PW=openstack

# Identity service API version string. ['v2.0', 'v3']
CONFIG_KEYSTONE_API_VERSION=v3

# Identity service token format (UUID, PKI or FERNET). The
# recommended format for new deployments is FERNET. ['UUID', 'PKI',
# 'FERNET']
CONFIG_KEYSTONE_TOKEN_FORMAT=FERNET

# Type of Identity service backend (sql or ldap). ['sql', 'ldap']
CONFIG_KEYSTONE_IDENTITY_BACKEND=sql

# URL for the Identity service LDAP backend.
CONFIG_KEYSTONE_LDAP_URL=ldap://10.0.2.15

# User DN for the Identity service LDAP backend.  Used to bind to the
# LDAP server if the LDAP server does not allow anonymous
# authentication.
CONFIG_KEYSTONE_LDAP_USER_DN=

# User DN password for the Identity service LDAP backend.
CONFIG_KEYSTONE_LDAP_USER_PASSWORD=

# Base suffix for the Identity service LDAP backend.
CONFIG_KEYSTONE_LDAP_SUFFIX=

# Query scope for the Identity service LDAP backend. Use 'one' for
# onelevel/singleLevel or 'sub' for subtree/wholeSubtree ('base' is
# not actually used by the Identity service and is therefore
# deprecated). ['base', 'one', 'sub']
CONFIG_KEYSTONE_LDAP_QUERY_SCOPE=one

# Query page size for the Identity service LDAP backend.
CONFIG_KEYSTONE_LDAP_PAGE_SIZE=-1

# User subtree for the Identity service LDAP backend.
CONFIG_KEYSTONE_LDAP_USER_SUBTREE=

# User query filter for the Identity service LDAP backend.
```

(continues on next page)

(continued from previous page)

```
CONFIG_KEYSTONE_LDAP_USER_FILTER=

# User object class for the Identity service LDAP backend.
CONFIG_KEYSTONE_LDAP_USER_OBJECTCLASS=

# User ID attribute for the Identity service LDAP backend.
CONFIG_KEYSTONE_LDAP_USER_ID_ATTRIBUTE=

# User name attribute for the Identity service LDAP backend.
CONFIG_KEYSTONE_LDAP_USER_NAME_ATTRIBUTE=

# User email address attribute for the Identity service LDAP backend.
CONFIG_KEYSTONE_LDAP_USER_MAIL_ATTRIBUTE=

# User-enabled attribute for the Identity service LDAP backend.
CONFIG_KEYSTONE_LDAP_USER_ENABLED_ATTRIBUTE=

# Bit mask integer applied to user-enabled attribute for the Identity
# service LDAP backend. Indicate the bit that the enabled value is
# stored in if the LDAP server represents "enabled" as a bit on an
# integer rather than a boolean. A value of "0" indicates the mask is
# not used (default). If this is not set to "0", the typical value is
# "2", typically used when
# "CONFIG_KEYSTONE_LDAP_USER_ENABLED_ATTRIBUTE = userAccountControl".
CONFIG_KEYSTONE_LDAP_USER_ENABLED_MASK=-1

# Value of enabled attribute which indicates user is enabled for the
# Identity service LDAP backend. This should match an appropriate
# integer value if the LDAP server uses non-boolean (bitmask) values
# to indicate whether a user is enabled or disabled. If this is not
# set as 'y', the typical value is "512". This is typically used when
# "CONFIG_KEYSTONE_LDAP_USER_ENABLED_ATTRIBUTE = userAccountControl".
CONFIG_KEYSTONE_LDAP_USER_ENABLED_DEFAULT=TRUE

# Specify 'y' if users are disabled (not enabled) in the Identity
# service LDAP backend (inverts boolean-enalbed values).  Some LDAP
# servers use a boolean lock attribute where "y" means an account is
# disabled. Setting this to 'y' allows these lock attributes to be
# used. This setting will have no effect if
# "CONFIG_KEYSTONE_LDAP_USER_ENABLED_MASK" is in use. ['n', 'y']
CONFIG_KEYSTONE_LDAP_USER_ENABLED_INVERT=n

# Comma-separated list of attributes stripped from LDAP user entry
# upon update.
CONFIG_KEYSTONE_LDAP_USER_ATTRIBUTE_IGNORE=

# Identity service LDAP attribute mapped to default_project_id for
# users.
CONFIG_KEYSTONE_LDAP_USER_DEFAULT_PROJECT_ID_ATTRIBUTE=

# Specify 'y' if you want to be able to create Identity service users
# through the Identity service interface; specify 'n' if you will
# create directly in the LDAP backend. ['n', 'y']
CONFIG_KEYSTONE_LDAP_USER_ALLOW_CREATE=n

# Specify 'y' if you want to be able to update Identity service users
# through the Identity service interface; specify 'n' if you will
```

(continues on next page)

(continued from previous page)

```
# update directly in the LDAP backend. ['n', 'y']
CONFIG_KEYSTONE_LDAP_USER_ALLOW_UPDATE=n

# Specify 'y' if you want to be able to delete Identity service users
# through the Identity service interface; specify 'n' if you will
# delete directly in the LDAP backend. ['n', 'y']
CONFIG_KEYSTONE_LDAP_USER_ALLOW_DELETE=n

# Identity service LDAP attribute mapped to password.
CONFIG_KEYSTONE_LDAP_USER_PASS_ATTRIBUTE=

# DN of the group entry to hold enabled LDAP users when using enabled
# emulation.
CONFIG_KEYSTONE_LDAP_USER_ENABLED_EMULATION_DN=

# List of additional LDAP attributes for mapping additional attribute
# mappings for users. The attribute-mapping format is
# <ldap_attr>:<user_attr>, where ldap_attr is the attribute in the
# LDAP entry and user_attr is the Identity API attribute.
CONFIG_KEYSTONE_LDAP_USER_ADDITIONAL_ATTRIBUTE_MAPPING=

# Group subtree for the Identity service LDAP backend.
CONFIG_KEYSTONE_LDAP_GROUP_SUBTREE=

# Group query filter for the Identity service LDAP backend.
CONFIG_KEYSTONE_LDAP_GROUP_FILTER=

# Group object class for the Identity service LDAP backend.
CONFIG_KEYSTONE_LDAP_GROUP_OBJECTCLASS=

# Group ID attribute for the Identity service LDAP backend.
CONFIG_KEYSTONE_LDAP_GROUP_ID_ATTRIBUTE=

# Group name attribute for the Identity service LDAP backend.
CONFIG_KEYSTONE_LDAP_GROUP_NAME_ATTRIBUTE=

# Group member attribute for the Identity service LDAP backend.
CONFIG_KEYSTONE_LDAP_GROUP_MEMBER_ATTRIBUTE=

# Group description attribute for the Identity service LDAP backend.
CONFIG_KEYSTONE_LDAP_GROUP_DESC_ATTRIBUTE=

# Comma-separated list of attributes stripped from LDAP group entry
# upon update.
CONFIG_KEYSTONE_LDAP_GROUP_ATTRIBUTE_IGNORE=

# Specify 'y' if you want to be able to create Identity service
# groups through the Identity service interface; specify 'n' if you
# will create directly in the LDAP backend. ['n', 'y']
CONFIG_KEYSTONE_LDAP_GROUP_ALLOW_CREATE=n

# Specify 'y' if you want to be able to update Identity service
# groups through the Identity service interface; specify 'n' if you
# will update directly in the LDAP backend. ['n', 'y']
CONFIG_KEYSTONE_LDAP_GROUP_ALLOW_UPDATE=n

# Specify 'y' if you want to be able to delete Identity service
```

(continues on next page)

(continued from previous page)

```
# groups through the Identity service interface; specify 'n' if you
# will delete directly in the LDAP backend. ['n', 'y']
CONFIG_KEYSTONE_LDAP_GROUP_ALLOW_DELETE=n

# List of additional LDAP attributes used for mapping additional
# attribute mappings for groups. The attribute=mapping format is
# <ldap_attr>:<group_attr>, where ldap_attr is the attribute in the
# LDAP entry and group_attr is the Identity API attribute.
CONFIG_KEYSTONE_LDAP_GROUP_ADDITIONAL_ATTRIBUTE_MAPPING=

# Specify 'y' if the Identity service LDAP backend should use TLS.
# ['n', 'y']
CONFIG_KEYSTONE_LDAP_USE_TLS=n

# CA certificate directory for Identity service LDAP backend (if TLS
# is used).
CONFIG_KEYSTONE_LDAP_TLS_CACERTDIR=

# CA certificate file for Identity service LDAP backend (if TLS is
# used).
CONFIG_KEYSTONE_LDAP_TLS_CACERTFILE=

# Certificate-checking strictness level for Identity service LDAP
# backend; valid options are: never, allow, demand. ['never', 'allow',
# 'demand']
CONFIG_KEYSTONE_LDAP_TLS_REQ_CERT=demand

# Password to use for the Image service (glance) to access the
# database.
CONFIG_GLANCE_DB_PW=55a47171835c4229

# Password to use for the Image service to authenticate with the
# Identity service.
CONFIG_GLANCE_KS_PW=7e2770838bab4988

# Storage backend for the Image service (controls how the Image
# service stores disk images). Valid options are: file or swift
# (Object Storage). The Object Storage service must be enabled to use
# it as a working backend; otherwise, Packstack falls back to 'file'.
# ['file', 'swift']
CONFIG_GLANCE_BACKEND=file

# Password to use for the Block Storage service (cinder) to access
# the database.
CONFIG_CINDER_DB_PW=4b4373ed09dc49f9

# Enter y if cron job for removing soft deleted DB rows should be
# created.
CONFIG_CINDER_DB_PURGE_ENABLE=True

# Password to use for the Block Storage service to authenticate with
# the Identity service.
CONFIG_CINDER_KS_PW=0e7ff9aa05d54997

# Storage backend to use for the Block Storage service; valid options
# are: lvm, gluster, nfs, vmdk, netapp, solidfire. ['lvm', 'gluster',
# 'nfs', 'vmdk', 'netapp', 'solidfire']
```

(continues on next page)

(continued from previous page)

```
CONFIG_CINDER_BACKEND=lvm

# Specify 'y' to create the Block Storage volumes group. That is,
# Packstack creates a raw disk image in /var/lib/cinder, and mounts it
# using a loopback device. This should only be used for testing on a
# proof-of-concept installation of the Block Storage service (a file-
# backed volume group is not suitable for production usage). ['y',
# 'n']
CONFIG_CINDER_VOLUMES_CREATE=y

# Specify a custom name for the lvm cinder volume group
CONFIG_CINDER_VOLUME_NAME=cinder-volumes

# Size of Block Storage volumes group. Actual volume size will be
# extended with 3% more space for VG metadata. Remember that the size
# of the volume group will restrict the amount of disk space that you
# can expose to Compute instances, and that the specified amount must
# be available on the device used for /var/lib/cinder.
CONFIG_CINDER_VOLUMES_SIZE=20G

# A single or comma-separated list of Red Hat Storage (gluster)
# volume shares to mount. Example: 'ip-address:/vol-name', 'domain
# :/vol-name'
CONFIG_CINDER_GLUSTER_MOUNTS=

# A single or comma-separated list of NFS exports to mount. Example:
# 'ip-address:/export-name'
CONFIG_CINDER_NFS_MOUNTS=

# Administrative user account name used to access the NetApp storage
# system or proxy server.
CONFIG_CINDER_NETAPP_LOGIN=

# Password for the NetApp administrative user account specified in
# the CONFIG_CINDER_NETAPP_LOGIN parameter.
CONFIG_CINDER_NETAPP_PASSWORD=

# Hostname (or IP address) for the NetApp storage system or proxy
# server.
CONFIG_CINDER_NETAPP_HOSTNAME=

# The TCP port to use for communication with the storage system or
# proxy. If not specified, Data ONTAP drivers will use 80 for HTTP and
# 443 for HTTPS; E-Series will use 8080 for HTTP and 8443 for HTTPS.
# Defaults to 80.
CONFIG_CINDER_NETAPP_SERVER_PORT=80

# Storage family type used on the NetApp storage system; valid
# options are ontap_7mode for using Data ONTAP operating in 7-Mode,
# ontap_cluster for using clustered Data ONTAP, or E-Series for NetApp
# E-Series. Defaults to ontap_cluster. ['ontap_7mode',
# 'ontap_cluster', 'eseries']
CONFIG_CINDER_NETAPP_STORAGE_FAMILY=ontap_cluster

# The transport protocol used when communicating with the NetApp
# storage system or proxy server. Valid values are http or https.
# Defaults to 'http'. ['http', 'https']
```

(continues on next page)

(continued from previous page)

```
CONFIG_CINDER_NETAPP_TRANSPORT_TYPE=http

# Storage protocol to be used on the data path with the NetApp
# storage system; valid options are iscsi, fc, nfs. Defaults to nfs.
# ['iscsi', 'fc', 'nfs']
CONFIG_CINDER_NETAPP_STORAGE_PROTOCOL=nfs

# Quantity to be multiplied by the requested volume size to ensure
# enough space is available on the virtual storage server (Vserver) to
# fulfill the volume creation request.  Defaults to 1.0.
CONFIG_CINDER_NETAPP_SIZE_MULTIPLIER=1.0

# Time period (in minutes) that is allowed to elapse after the image
# is last accessed, before it is deleted from the NFS image cache.
# When a cache-cleaning cycle begins, images in the cache that have
# not been accessed in the last M minutes, where M is the value of
# this parameter, are deleted from the cache to create free space on
# the NFS share. Defaults to 720.
CONFIG_CINDER_NETAPP_EXPIRY_THRES_MINUTES=720

# If the percentage of available space for an NFS share has dropped
# below the value specified by this parameter, the NFS image cache is
# cleaned.  Defaults to 20.
CONFIG_CINDER_NETAPP_THRES_AVL_SIZE_PERC_START=20

# When the percentage of available space on an NFS share has reached
# the percentage specified by this parameter, the driver stops
# clearing files from the NFS image cache that have not been accessed
# in the last M minutes, where M is the value of the
# CONFIG_CINDER_NETAPP_EXPIRY_THRES_MINUTES parameter. Defaults to 60.
CONFIG_CINDER_NETAPP_THRES_AVL_SIZE_PERC_STOP=60

# Single or comma-separated list of NetApp NFS shares for Block
# Storage to use.  Format: ip-address:/export-name. Defaults to ''.
CONFIG_CINDER_NETAPP_NFS_SHARES=

# File with the list of available NFS shares.   Defaults to
# '/etc/cinder/shares.conf'.
CONFIG_CINDER_NETAPP_NFS_SHARES_CONFIG=/etc/cinder/shares.conf

# This parameter is only utilized when the storage protocol is
# configured to use iSCSI or FC. This parameter is used to restrict
# provisioning to the specified controller volumes. Specify the value
# of this parameter to be a comma separated list of NetApp controller
# volume names to be used for provisioning. Defaults to ''.
CONFIG_CINDER_NETAPP_VOLUME_LIST=

# The vFiler unit on which provisioning of block storage volumes will
# be done. This parameter is only used by the driver when connecting
# to an instance with a storage family of Data ONTAP operating in
# 7-Mode Only use this parameter when utilizing the MultiStore feature
# on the NetApp storage system. Defaults to ''.
CONFIG_CINDER_NETAPP_VFILER=

# The name of the config.conf stanza for a Data ONTAP (7-mode) HA
# partner.  This option is only used by the driver when connecting to
# an instance with a storage family of Data ONTAP operating in 7-Mode,
```

(continues on next page)

(continued from previous page)

```
# and it is required if the storage protocol selected is FC. Defaults
# to ''.
CONFIG_CINDER_NETAPP_PARTNER_BACKEND_NAME=

# This option specifies the virtual storage server (Vserver) name on
# the storage cluster on which provisioning of block storage volumes
# should occur. Defaults to ''.
CONFIG_CINDER_NETAPP_VSERVER=

# Restricts provisioning to the specified controllers. Value must be
# a comma-separated list of controller hostnames or IP addresses to be
# used for provisioning. This option is only utilized when the storage
# family is configured to use E-Series. Defaults to ''.
CONFIG_CINDER_NETAPP_CONTROLLER_IPS=

# Password for the NetApp E-Series storage array. Defaults to ''.
CONFIG_CINDER_NETAPP_SA_PASSWORD=

# This option is used to define how the controllers in the E-Series
# storage array will work with the particular operating system on the
# hosts that are connected to it. Defaults to 'linux_dm_mp'
CONFIG_CINDER_NETAPP_ESERIES_HOST_TYPE=linux_dm_mp

# Path to the NetApp E-Series proxy application on a proxy server.
# The value is combined with the value of the
# CONFIG_CINDER_NETAPP_TRANSPORT_TYPE, CONFIG_CINDER_NETAPP_HOSTNAME,
# and CONFIG_CINDER_NETAPP_HOSTNAME options to create the URL used by
# the driver to connect to the proxy application. Defaults to
# '/devmgr/v2'.
CONFIG_CINDER_NETAPP_WEBSERVICE_PATH=/devmgr/v2

# Restricts provisioning to the specified storage pools. Only dynamic
# disk pools are currently supported. The value must be a comma-
# separated list of disk pool names to be used for provisioning.
# Defaults to ''.
CONFIG_CINDER_NETAPP_STORAGE_POOLS=

# Cluster admin account name used to access the SolidFire storage
# system.
CONFIG_CINDER_SOLIDFIRE_LOGIN=

# Password for the SolidFire cluster admin user account specified in
# the CONFIG_CINDER_SOLIDFIRE_LOGIN parameter.
CONFIG_CINDER_SOLIDFIRE_PASSWORD=

# Hostname (or IP address) for the SolidFire storage system's MVIP.
CONFIG_CINDER_SOLIDFIRE_HOSTNAME=

# Password to use for OpenStack Bare Metal Provisioning (ironic) to
# access the database.
CONFIG_IRONIC_DB_PW=PW_PLACEHOLDER

# Password to use for OpenStack Bare Metal Provisioning to
# authenticate with the Identity service.
CONFIG_IRONIC_KS_PW=PW_PLACEHOLDER

# Enter y if cron job for removing soft deleted DB rows should be
```

(continues on next page)

(continued from previous page)

```
# created.
CONFIG_NOVA_DB_PURGE_ENABLE=True

# Password to use for the Compute service (nova) to access the
# database.
CONFIG_NOVA_DB_PW=6295662e92e94420

# Password to use for the Compute service to authenticate with the
# Identity service.
CONFIG_NOVA_KS_PW=f3b48c3136e544c1

# Whether or not Packstack should manage a default initial set of
# Nova flavors. Defaults to 'y'.
CONFIG_NOVA_MANAGE_FLAVORS=y

# Overcommitment ratio for virtual to physical CPUs. Specify 1.0 to
# disable CPU overcommitment.
CONFIG_NOVA_SCHED_CPU_ALLOC_RATIO=16.0

# Overcommitment ratio for virtual to physical RAM. Specify 1.0 to
# disable RAM overcommitment.
CONFIG_NOVA_SCHED_RAM_ALLOC_RATIO=1.5

# Protocol used for instance migration. Valid options are: ssh and
# tcp. Note that the tcp protocol is not encrypted, so it is insecure.
# ['ssh', 'tcp']
CONFIG_NOVA_COMPUTE_MIGRATE_PROTOCOL=ssh

# PEM encoded certificate to be used for ssl on the https server,
# leave blank if one should be generated, this certificate should not
# require a passphrase. If CONFIG_HORIZON_SSL is set to 'n' this
# parameter is ignored.
CONFIG_VNC_SSL_CERT=

# SSL keyfile corresponding to the certificate if one was entered. If
# CONFIG_HORIZON_SSL is set to 'n' this parameter is ignored.
CONFIG_VNC_SSL_KEY=

# Enter the PCI passthrough array of hash in JSON style for
# controller eg. [{"vendor_id":"1234", "product_id":"5678",
# "name":"default"}, {...}]
CONFIG_NOVA_PCI_ALIAS=

# Enter the PCI passthrough whitelist array of hash in JSON style for
# controller eg. [{"vendor_id":"1234", "product_id":"5678",
# "name':"default"}, {...}]
CONFIG_NOVA_PCI_PASSTHROUGH_WHITELIST=

# The hypervisor driver to use with Nova. Can be either 'qemu' or
# 'kvm'. Defaults to 'qemu' on virtual machines and 'kvm' on bare
# metal hardware. For nested KVM set it explicitly to 'kvm'.
CONFIG_NOVA_LIBVIRT_VIRT_TYPE=%{::default_hypervisor}

# Password to use for OpenStack Networking (neutron) to authenticate
# with the Identity service.
CONFIG_NEUTRON_KS_PW=a06d5f5ee976464f
```

(continues on next page)

(continued from previous page)

```
# The password to use for OpenStack Networking to access the
# database.
CONFIG_NEUTRON_DB_PW=dc36414078674e17

# The name of the Open vSwitch bridge (or empty for linuxbridge) for
# the OpenStack Networking L3 agent to use for external  traffic.
# Specify 'provider' if you intend to use a provider network to handle
# external traffic.
CONFIG_NEUTRON_L3_EXT_BRIDGE=br-ex

# Password for the OpenStack Networking metadata agent.
CONFIG_NEUTRON_METADATA_PW=7174442df1764ba1

# Specify 'y' to install OpenStack Networking's Load-Balancing-
# as-a-Service (LBaaS). ['y', 'n']
CONFIG_LBAAS_INSTALL=y

# Specify 'y' to install OpenStack Networking's L3 Metering agent
# ['y', 'n']
CONFIG_NEUTRON_METERING_AGENT_INSTALL=y

# Specify 'y' to configure OpenStack Networking's Firewall-
# as-a-Service (FWaaS). ['y', 'n']
CONFIG_NEUTRON_FWAAS=n

# Specify 'y' to configure OpenStack Networking's VPN-as-a-Service
# (VPNaaS). ['y', 'n']
CONFIG_NEUTRON_VPNAAS=n

# Comma-separated list of network-type driver entry points to be
# loaded from the neutron.ml2.type_drivers namespace. ['local',
# 'flat', 'vlan', 'gre', 'vxlan', 'geneve']
CONFIG_NEUTRON_ML2_TYPE_DRIVERS=vxlan,flat

# Comma-separated, ordered list of network types to allocate as
# tenant networks. The 'local' value is only useful for single-box
# testing and provides no connectivity between hosts. ['local',
# 'vlan', 'gre', 'vxlan', 'geneve']
CONFIG_NEUTRON_ML2_TENANT_NETWORK_TYPES=vxlan

# Comma-separated ordered list of networking mechanism driver entry
# points to be loaded from the neutron.ml2.mechanism_drivers
# namespace. ['logger', 'test', 'linuxbridge', 'openvswitch',
# 'hyperv', 'ncs', 'arista', 'cisco_nexus', 'mlnx', 'l2population',
# 'sriovnicswitch', 'ovn']
CONFIG_NEUTRON_ML2_MECHANISM_DRIVERS=openvswitch

# Comma-separated list of physical_network names with which flat
# networks can be created. Use * to allow flat networks with arbitrary
# physical_network names.
CONFIG_NEUTRON_ML2_FLAT_NETWORKS=*

# Comma-separated list of <physical_network>:<vlan_min>:<vlan_max> or
# <physical_network> specifying physical_network names usable for VLAN
# provider and tenant networks, as well as ranges of VLAN tags on each
# available for allocation to tenant networks.
CONFIG_NEUTRON_ML2_VLAN_RANGES=
```

(continues on next page)

(continued from previous page)

```
# Comma-separated list of <tun_min>:<tun_max> tuples enumerating
# ranges of GRE tunnel IDs that are available for tenant-network
# allocation. A tuple must be an array with tun_max +1 - tun_min >
# 1000000.
CONFIG_NEUTRON_ML2_TUNNEL_ID_RANGES=

# Comma-separated list of addresses for VXLAN multicast group. If
# left empty, disables VXLAN from sending allocate broadcast traffic
# (disables multicast VXLAN mode). Should be a Multicast IP (v4 or v6)
# address.
CONFIG_NEUTRON_ML2_VXLAN_GROUP=

# Comma-separated list of <vni_min>:<vni_max> tuples enumerating
# ranges of VXLAN VNI IDs that are available for tenant network
# allocation. Minimum value is 0 and maximum value is 16777215.
CONFIG_NEUTRON_ML2_VNI_RANGES=10:100

# Name of the L2 agent to be used with OpenStack Networking.
# ['linuxbridge', 'openvswitch', 'ovn']
CONFIG_NEUTRON_L2_AGENT=openvswitch

# Comma separated list of supported PCI vendor devices defined by
# vendor_id:product_id according to the PCI ID Repository.
CONFIG_NEUTRON_ML2_SUPPORTED_PCI_VENDOR_DEVS=['15b3:1004', '8086:10ca']

# Comma-separated list of interface mappings for the OpenStack
# Networking ML2 SRIOV agent. Each tuple in the list must be in the
# format <physical_network>:<net_interface>. Example:
# physnet1:eth1,physnet2:eth2,physnet3:eth3.
CONFIG_NEUTRON_ML2_SRIOV_INTERFACE_MAPPINGS=

# Comma-separated list of interface mappings for the OpenStack
# Networking linuxbridge plugin. Each tuple in the list must be in the
# format <physical_network>:<net_interface>. Example:
# physnet1:eth1,physnet2:eth2,physnet3:eth3.
CONFIG_NEUTRON_LB_INTERFACE_MAPPINGS=

# Comma-separated list of bridge mappings for the OpenStack
# Networking Open vSwitch plugin. Each tuple in the list must be in
# the format <physical_network>:<ovs_bridge>. Example: physnet1:br-
# eth1,physnet2:br-eth2,physnet3:br-eth3
CONFIG_NEUTRON_OVS_BRIDGE_MAPPINGS=extnet:br-ex

# Comma-separated list of colon-separated Open vSwitch
# <bridge>:<interface> pairs. The interface will be added to the
# associated bridge. If you desire the bridge to be persistent a value
# must be added to this directive, also
# CONFIG_NEUTRON_OVS_BRIDGE_MAPPINGS must be set in order to create
# the proper port. This can be achieved from the command line by
# issuing the following command: packstack --allinone --os-neutron-
# ovs-bridge-mappings=ext-net:br-ex --os-neutron-ovs-bridge-interfaces
# =br-ex:eth0
CONFIG_NEUTRON_OVS_BRIDGE_IFACES=

# Comma-separated list of Open vSwitch bridges that must be created
# and connected to interfaces in compute nodes when flat or vlan type
```

(continues on next page)

(continued from previous page)

```
# drivers are enabled. These bridges must exist in
# CONFIG_NEUTRON_OVS_BRIDGE_MAPPINGS and
# CONFIG_NEUTRON_OVS_BRIDGE_IFACES. Example: --os-neutron-ovs-bridges-
# compute=br-vlan --os-neutron-ovs-bridge-mappings="extnet:br-
# ex,physnet1:br-vlan" --os-neutron-ovs-bridge-interfaces="br-ex:eth1
# ,br-vlan:eth2"
CONFIG_NEUTRON_OVS_BRIDGES_COMPUTE=

# Name of physical network used for external network when enabling
# CONFIG_PROVISION_DEMO. Name must be one of the included in
# CONFIG_NEUTRON_OVS_BRIDGE_MAPPINGS. Example: --os-neutron-ovs-
# bridge-mappings="extnet:br-ex,physnet1:br-vlan" --os-neutron-ovs-
# bridge-interfaces="br-ex:eth1,br-vlan:eth2" --os-neutron-ovs-
# external-physnet="extnet"
CONFIG_NEUTRON_OVS_EXTERNAL_PHYSNET=extnet

# Interface for the Open vSwitch tunnel. Packstack overrides the IP
# address used for tunnels on this hypervisor to the IP found on the
# specified interface (for example, eth1).
CONFIG_NEUTRON_OVS_TUNNEL_IF=

# Comma-separated list of subnets (for example,
# 192.168.10.0/24,192.168.11.0/24) used for sending tunneling packets.
# This is used to configure IP filtering to accept tunneling packets
# from these subnets instead of specific IP addresses of peer nodes.
# This is useful when you add existing nodes to EXCLUDE_SERVERS
# because, in this case, packstack cannot modify the IP filtering of
# the existing nodes.
CONFIG_NEUTRON_OVS_TUNNEL_SUBNETS=

# VXLAN UDP port.
CONFIG_NEUTRON_OVS_VXLAN_UDP_PORT=4789

# Comma-separated list of bridge mappings for the OpenStack
# Networking Open Virtual Network plugin. Each tuple in the list must
# be in the format <physical_network>:<ovs_bridge>. Example: physnet1
# :br-eth1,physnet2:br-eth2,physnet3:br-eth3
CONFIG_NEUTRON_OVN_BRIDGE_MAPPINGS=extnet:br-ex

# Comma-separated list of colon-separated Open vSwitch
# <bridge>:<interface> pairs. The interface will be added to the
# associated bridge. If you desire the bridge to be persistent a value
# must be added to this directive, also
# CONFIG_NEUTRON_OVN_BRIDGE_MAPPINGS must be set in order to create
# the proper port. This can be achieved from the command line by
# issuing the following command: packstack --allinone --os-neutron-
# ovn-bridge-mappings=ext-net:br-ex --os-neutron-ovn-bridge-interfaces
# =br-ex:eth0
CONFIG_NEUTRON_OVN_BRIDGE_IFACES=

# Comma-separated list of Open vSwitch bridges that must be created
# and connected to interfaces in compute nodes when flat or vlan type
# drivers are enabled. These bridges must exist in
# CONFIG_NEUTRON_OVN_BRIDGE_MAPPINGS and
# CONFIG_NEUTRON_OVN_BRIDGE_IFACES. Example: --os-neutron-ovn-bridges-
# compute=br-vlan --os-neutron-ovn-bridge-mappings="extnet:br-
# ex,physnet1:br-vlan" --os-neutron-ovn-bridge-interfaces="br-ex:eth1
```

(continues on next page)

(continued from previous page)

```
# ,br-vlan:eth2"
CONFIG_NEUTRON_OVN_BRIDGES_COMPUTE=

# Name of physical network used for external network when enabling
# CONFIG_PROVISION_DEMO. Name must be one of the included in
# CONFIG_NEUTRON_OVN_BRIDGE_MAPPINGS. Example: --os-neutron-ovn-
# bridge-mappings="extnet:br-ex,physnet1:br-vlan" --os-neutron-ovn-
# bridge-interfaces="br-ex:eth1,br-vlan:eth2" --os-neutron-ovn-
# external-physnet="extnet"
CONFIG_NEUTRON_OVN_EXTERNAL_PHYSNET=extnet

# Interface for the Open vSwitch tunnel. Packstack overrides the IP
# address used for tunnels on this hypervisor to the IP found on the
# specified interface (for example, eth1).
CONFIG_NEUTRON_OVN_TUNNEL_IF=

# Comma-separated list of subnets (for example,
# 192.168.10.0/24,192.168.11.0/24) used for sending tunneling packets.
# This is used to configure IP filtering to accept tunneling packets
# from these subnets instead of specific IP addresses of peer nodes.
# This is useful when you add existing nodes to EXCLUDE_SERVERS
# because, in this case, packstack cannot modify the IP filtering of
# the existing nodes.
CONFIG_NEUTRON_OVN_TUNNEL_SUBNETS=

# Password to use for the OpenStack File Share service (manila) to
# access the database.
CONFIG_MANILA_DB_PW=PW_PLACEHOLDER

# Password to use for the OpenStack File Share service (manila) to
# authenticate with the Identity service.
CONFIG_MANILA_KS_PW=PW_PLACEHOLDER

# Backend for the OpenStack File Share service (manila); valid
# options are: generic, netapp, glusternative, or glusternfs.
# ['generic', 'netapp', 'glusternative', 'glusternfs']
CONFIG_MANILA_BACKEND=generic

# Denotes whether the driver should handle the responsibility of
# managing share servers. This must be set to false if the driver is
# to operate without managing share servers. Defaults to 'false'
# ['true', 'false']
CONFIG_MANILA_NETAPP_DRV_HANDLES_SHARE_SERVERS=false

# The transport protocol used when communicating with the storage
# system or proxy server. Valid values are 'http' and 'https'.
# Defaults to 'https'. ['https', 'http']
CONFIG_MANILA_NETAPP_TRANSPORT_TYPE=https

# Administrative user account name used to access the NetApp storage
# system.  Defaults to ''.
CONFIG_MANILA_NETAPP_LOGIN=admin

# Password for the NetApp administrative user account specified in
# the CONFIG_MANILA_NETAPP_LOGIN parameter. Defaults to ''.
CONFIG_MANILA_NETAPP_PASSWORD=
```

(continues on next page)

(continued from previous page)

```
# Hostname (or IP address) for the NetApp storage system or proxy
# server. Defaults to ''.
CONFIG_MANILA_NETAPP_SERVER_HOSTNAME=

# The storage family type used on the storage system; valid values
# are ontap_cluster for clustered Data ONTAP. Defaults to
# 'ontap_cluster'. ['ontap_cluster']
CONFIG_MANILA_NETAPP_STORAGE_FAMILY=ontap_cluster

# The TCP port to use for communication with the storage system or
# proxy server. If not specified, Data ONTAP drivers will use 80 for
# HTTP and 443 for HTTPS. Defaults to '443'.
CONFIG_MANILA_NETAPP_SERVER_PORT=443

# Pattern for searching available aggregates for NetApp provisioning.
# Defaults to '(.*)'.
CONFIG_MANILA_NETAPP_AGGREGATE_NAME_SEARCH_PATTERN=(.*)

# Name of aggregate on which to create the NetApp root volume. This
# option only applies when the option
# CONFIG_MANILA_NETAPP_DRV_HANDLES_SHARE_SERVERS is set to True.
CONFIG_MANILA_NETAPP_ROOT_VOLUME_AGGREGATE=

# NetApp root volume name. Defaults to 'root'.
CONFIG_MANILA_NETAPP_ROOT_VOLUME_NAME=root

# This option specifies the storage virtual machine (previously
# called a Vserver) name on the storage cluster on which provisioning
# of shared file systems should occur. This option only applies when
# the option driver_handles_share_servers is set to False. Defaults to
# ''.
CONFIG_MANILA_NETAPP_VSERVER=

# Denotes whether the driver should handle the responsibility of
# managing share servers. This must be set to false if the driver is
# to operate without managing share servers. Defaults to 'true'.
# ['true', 'false']
CONFIG_MANILA_GENERIC_DRV_HANDLES_SHARE_SERVERS=true

# Volume name template for Manila service. Defaults to 'manila-
# share-%s'.
CONFIG_MANILA_GENERIC_VOLUME_NAME_TEMPLATE=manila-share-%s

# Share mount path for Manila service. Defaults to '/shares'.
CONFIG_MANILA_GENERIC_SHARE_MOUNT_PATH=/shares

# Location of disk image for Manila service instance. Defaults to '
CONFIG_MANILA_SERVICE_IMAGE_LOCATION=https://www.dropbox.com/s/vi5oeh10q1qkckh/ubuntu_
↪1204_nfs_cifs.qcow2

# User in Manila service instance.
CONFIG_MANILA_SERVICE_INSTANCE_USER=ubuntu

# Password to service instance user.
CONFIG_MANILA_SERVICE_INSTANCE_PASSWORD=ubuntu

# Type of networking that the backend will use. A more detailed
```

(continues on next page)

(continued from previous page)

```
# description of each option is available in the Manila docs. Defaults
# to 'neutron'. ['neutron', 'nova-network', 'standalone']
CONFIG_MANILA_NETWORK_TYPE=neutron

# Gateway IPv4 address that should be used. Required. Defaults to ''.
CONFIG_MANILA_NETWORK_STANDALONE_GATEWAY=

# Network mask that will be used. Can be either decimal like '24' or
# binary like '255.255.255.0'. Required. Defaults to ''.
CONFIG_MANILA_NETWORK_STANDALONE_NETMASK=

# Set it if network has segmentation (VLAN, VXLAN, etc). It will be
# assigned to share-network and share drivers will be able to use this
# for network interfaces within provisioned share servers. Optional.
# Example: 1001. Defaults to ''.
CONFIG_MANILA_NETWORK_STANDALONE_SEG_ID=

# Can be IP address, range of IP addresses or list of addresses or
# ranges. Contains addresses from IP network that are allowed to be
# used. If empty, then will be assumed that all host addresses from
# network can be used. Optional. Examples: 10.0.0.10 or
# 10.0.0.10-10.0.0.20 or
# 10.0.0.10-10.0.0.20,10.0.0.30-10.0.0.40,10.0.0.50. Defaults to ''.
CONFIG_MANILA_NETWORK_STANDALONE_IP_RANGE=

# IP version of network. Optional. Defaults to '4'. ['4', '6']
CONFIG_MANILA_NETWORK_STANDALONE_IP_VERSION=4

# List of GlusterFS servers that can be used to create shares. Each
# GlusterFS server should be of the form [remoteuser@]<volserver>, and
# they are assumed to belong to distinct Gluster clusters.
CONFIG_MANILA_GLUSTERFS_SERVERS=

# Path of Manila host's private SSH key file.
CONFIG_MANILA_GLUSTERFS_NATIVE_PATH_TO_PRIVATE_KEY=

# Regular expression template used to filter GlusterFS volumes for
# share creation. The regex template can optionally (ie. with support
# of the GlusterFS backend) contain the #{size} parameter which
# matches an integer (sequence of digits) in which case the value
# shall be intepreted as size of the volume in GB. Examples: "manila-
# share-volume-d+$", "manila-share-volume-#{size}G-d+$"; with matching
# volume names, respectively: "manila-share-volume-12", "manila-share-
# volume-3G-13". In latter example, the number that matches "#{size}",
# that is, 3, is an indication that the size of volume is 3G.
CONFIG_MANILA_GLUSTERFS_VOLUME_PATTERN=

# Specifies the GlusterFS volume to be mounted on the Manila host.
# For e.g: [remoteuser@]<volserver>:/<volid>
CONFIG_MANILA_GLUSTERFS_TARGET=

# Base directory containing mount points for Gluster volumes.
CONFIG_MANILA_GLUSTERFS_MOUNT_POINT_BASE=

# Type of NFS server that mediate access to the Gluster volumes
# (Gluster or Ganesha).
CONFIG_MANILA_GLUSTERFS_NFS_SERVER_TYPE=gluster
```

(continues on next page)

(continued from previous page)

```
# Path of Manila host's private SSH key file.
CONFIG_MANILA_GLUSTERFS_PATH_TO_PRIVATE_KEY=

# Remote Ganesha server node's IP address.
CONFIG_MANILA_GLUSTERFS_GANESHA_SERVER_IP=

# Specify 'y' to set up Horizon communication over https. ['y', 'n']
CONFIG_HORIZON_SSL=n

# Secret key to use for Horizon Secret Encryption Key.
CONFIG_HORIZON_SECRET_KEY=344bf1addd2d49c0beab4b659bf0e55f

# PEM-encoded certificate to be used for SSL connections on the https
# server. To generate a certificate, leave blank.
CONFIG_HORIZON_SSL_CERT=

# SSL keyfile corresponding to the certificate if one was specified.
# The certificate should not require a passphrase.
CONFIG_HORIZON_SSL_KEY=

CONFIG_HORIZON_SSL_CACERT=

# Password to use for the Object Storage service to authenticate with
# the Identity service.
CONFIG_SWIFT_KS_PW=7e9cb3e48d014ad5

# Comma-separated list of devices to use as storage device for Object
# Storage. Each entry must take the format /path/to/dev (for example,
# specifying /dev/vdb installs /dev/vdb as the Object Storage storage
# device; Packstack does not create the filesystem, you must do this
# first). If left empty, Packstack creates a loopback device for test
# setup.
CONFIG_SWIFT_STORAGES=

# Number of Object Storage storage zones; this number MUST be no
# larger than the number of configured storage devices.
CONFIG_SWIFT_STORAGE_ZONES=1

# Number of Object Storage storage replicas; this number MUST be no
# larger than the number of configured storage zones.
CONFIG_SWIFT_STORAGE_REPLICAS=1

# File system type for storage nodes. ['xfs', 'ext4']
CONFIG_SWIFT_STORAGE_FSTYPE=ext4

# Custom seed number to use for swift_hash_path_suffix in
# /etc/swift/swift.conf. If you do not provide a value, a seed number
# is automatically generated.
CONFIG_SWIFT_HASH=4fa03e02fd9240ec

# Size of the Object Storage loopback file storage device.
CONFIG_SWIFT_STORAGE_SIZE=2G

# Password used by Orchestration service user to authenticate against
# the database.
CONFIG_HEAT_DB_PW=bca90ad488604f3f
```

(continues on next page)

(continued from previous page)

```
# Encryption key to use for authentication in the Orchestration
# database (16, 24, or 32 chars).
CONFIG_HEAT_AUTH_ENC_KEY=43c886595fec4dfa

# Password to use for the Orchestration service to authenticate with
# the Identity service.
CONFIG_HEAT_KS_PW=3d68fd66259c41ee

# Specify 'y' to install the Orchestration CloudFormation API. ['y',
# 'n']
CONFIG_HEAT_CFN_INSTALL=y

# Name of the Identity domain for Orchestration.
CONFIG_HEAT_DOMAIN=heat

# Name of the Identity domain administrative user for Orchestration.
CONFIG_HEAT_DOMAIN_ADMIN=heat_admin

# Password for the Identity domain administrative user for
# Orchestration.
CONFIG_HEAT_DOMAIN_PASSWORD=3435199bed7c4095

# Specify 'y' to provision for demo usage and testing. ['y', 'n']
CONFIG_PROVISION_DEMO=y

# Specify 'y' to configure the OpenStack Integration Test Suite
# (tempest) for testing. The test suite requires OpenStack Networking
# to be installed. ['y', 'n']
CONFIG_PROVISION_TEMPEST=n

# CIDR network address for the floating IP subnet.
CONFIG_PROVISION_DEMO_FLOATRANGE=172.24.4.0/24

# Allocation pools in the floating IP subnet.
CONFIG_PROVISION_DEMO_ALLOCATION_POOLS=[]

# The name to be assigned to the demo image in Glance (default
# "cirros").
CONFIG_PROVISION_IMAGE_NAME=cirros

# A URL or local file location for an image to download and provision
# in Glance (defaults to a URL for a recent "cirros" image).
CONFIG_PROVISION_IMAGE_URL=http://download.cirros-cloud.net/0.3.5/cirros-0.3.5-x86_64-
↪disk.img

# Format for the demo image (default "qcow2").
CONFIG_PROVISION_IMAGE_FORMAT=qcow2

# Properties of the demo image (none by default).
CONFIG_PROVISION_IMAGE_PROPERTIES=

# User to use when connecting to instances booted from the demo
# image.
CONFIG_PROVISION_IMAGE_SSH_USER=cirros

# Name of the uec image created in Glance used in tempest tests
```

(continues on next page)

(continued from previous page)

```
# (default "cirros-uec").
CONFIG_PROVISION_UEC_IMAGE_NAME=cirros-uec

# URL of the kernel image copied to Glance image for uec image
# (defaults to a URL for a recent "cirros" uec image).
CONFIG_PROVISION_UEC_IMAGE_KERNEL_URL=http://download.cirros-cloud.net/0.3.5/cirros-0.
↪3.5-x86_64-kernel

# URL of the ramdisk image copied to Glance image for uec image
# (defaults to a URL for a recent "cirros" uec image).
CONFIG_PROVISION_UEC_IMAGE_RAMDISK_URL=http://download.cirros-cloud.net/0.3.5/cirros-
↪0.3.5-x86_64-initramfs

# URL of the disk image copied to Glance image for uec image
# (defaults to a URL for a recent "cirros" uec image).
CONFIG_PROVISION_UEC_IMAGE_DISK_URL=http://download.cirros-cloud.net/0.3.5/cirros-0.3.
↪5-x86_64-disk.img

CONFIG_TEMPEST_HOST=

# Name of the Integration Test Suite provisioning user. If you do not
# provide a user name, Tempest is configured in a standalone mode.
CONFIG_PROVISION_TEMPEST_USER=

# Password to use for the Integration Test Suite provisioning user.
CONFIG_PROVISION_TEMPEST_USER_PW=PW_PLACEHOLDER

# CIDR network address for the floating IP subnet.
CONFIG_PROVISION_TEMPEST_FLOATRANGE=172.24.4.0/24

# Primary flavor name to use in Tempest.
CONFIG_PROVISION_TEMPEST_FLAVOR_NAME=m1.nano

# Primary flavor's disk quota in Gb.
CONFIG_PROVISION_TEMPEST_FLAVOR_DISK=0

# Primary flavor's ram in Mb.
CONFIG_PROVISION_TEMPEST_FLAVOR_RAM=128

# Primary flavor's vcpus number.
CONFIG_PROVISION_TEMPEST_FLAVOR_VCPUS=1

# Alternative flavor name to use in Tempest.
CONFIG_PROVISION_TEMPEST_FLAVOR_ALT_NAME=m1.micro

# Alternative flavor's disk quota in Gb.
CONFIG_PROVISION_TEMPEST_FLAVOR_ALT_DISK=0

# Alternative flavor's ram in Mb.
CONFIG_PROVISION_TEMPEST_FLAVOR_ALT_RAM=128

# Alternative flavor's vcpus number.
CONFIG_PROVISION_TEMPEST_FLAVOR_ALT_VCPUS=1

# Specify 'y' to run Tempest smoke test as last step of installation.
CONFIG_RUN_TEMPEST=n
```

(continues on next page)

(continued from previous page)

```
# Test suites to run, example: "smoke dashboard TelemetryAlarming".
# Optional, defaults to "smoke".
CONFIG_RUN_TEMPEST_TESTS=smoke

# Specify 'y' to configure the Open vSwitch external bridge for an
# all-in-one deployment (the L3 external bridge acts as the gateway
# for virtual machines). ['y', 'n']
CONFIG_PROVISION_OVS_BRIDGE=y

# Password to use for Gnocchi to access the database.
CONFIG_GNOCCHI_DB_PW=6fb1d9e8bf884e37

# Password to use for Gnocchi to authenticate with the Identity
# service.
CONFIG_GNOCCHI_KS_PW=2818fec28de64c57

# Secret key for signing Telemetry service (ceilometer) messages.
CONFIG_CEILOMETER_SECRET=c26282c0fb404345

# Password to use for Telemetry to authenticate with the Identity
# service.
CONFIG_CEILOMETER_KS_PW=b1388b48ef6147f2

# Ceilometer service name. ['httpd', 'ceilometer']
CONFIG_CEILOMETER_SERVICE_NAME=httpd

# Backend driver for Telemetry's group membership coordination.
# ['redis', 'none']
CONFIG_CEILOMETER_COORDINATION_BACKEND=redis

# Whether to enable ceilometer middleware in swift proxy. By default
# this should be false to avoid unnecessary load.
CONFIG_ENABLE_CEILOMETER_MIDDLEWARE=n

# IP address of the server on which to install the Redis server.
CONFIG_REDIS_HOST=10.0.1.20

# Port on which the Redis server listens.
CONFIG_REDIS_PORT=6379

# Password to use for Telemetry Alarming to authenticate with the
# Identity service.
CONFIG_AODH_KS_PW=2c5c29bc887a4185

# Password to use for Telemetry Alarming (AODH) to access the
# database.
CONFIG_AODH_DB_PW=dda2cdf3fbd64ff6

# Password to use for Panko to access the database.
CONFIG_PANKO_DB_PW=PW_PLACEHOLDER

# Password to use for Panko to authenticate with the Identity
# service.
CONFIG_PANKO_KS_PW=PW_PLACEHOLDER

# Password to use for OpenStack Database-as-a-Service (trove) to
# access the database.
```

(continues on next page)

(continued from previous page)

```
CONFIG_TROVE_DB_PW=PW_PLACEHOLDER

# Password to use for OpenStack Database-as-a-Service to authenticate
# with the Identity service.
CONFIG_TROVE_KS_PW=PW_PLACEHOLDER

# User name to use when OpenStack Database-as-a-Service connects to
# the Compute service.
CONFIG_TROVE_NOVA_USER=trove

# Tenant to use when OpenStack Database-as-a-Service connects to the
# Compute service.
CONFIG_TROVE_NOVA_TENANT=services

# Password to use when OpenStack Database-as-a-Service connects to
# the Compute service.
CONFIG_TROVE_NOVA_PW=PW_PLACEHOLDER

# Password to use for OpenStack Data Processing (sahara) to access
# the database.
CONFIG_SAHARA_DB_PW=PW_PLACEHOLDER

# Password to use for OpenStack Data Processing to authenticate with
# the Identity service.
CONFIG_SAHARA_KS_PW=PW_PLACEHOLDER

# Password to use for the Magnum to access the database.
CONFIG_MAGNUM_DB_PW=PW_PLACEHOLDER

# Password to use for the Magnum to authenticate with the Identity
# service.
CONFIG_MAGNUM_KS_PW=PW_PLACEHOLDER
```

CHAPTER 17

OpenStack-Ansible

17.1 OpenStack-Ansible Deployment

Table of Contents

17.1.1 OpenStack-Ansible - Documentación oficial

- OpenStack-Ansible Documentation (Train release)
- OpenStack-Ansible Deployment Guide (Train release)
- OpenStack-Ansible - Quickstart AIO (Train release)

17.2 OpenStack-Ansible CentOS 7 AIO

Table of Contents

Referencia: OpenStack-Ansible - Quickstart AIO

Despliegues All-in-one (AIO) son una buena forma de realizar un despliegue OpenStack-Ansible para:

- entornos de desarrollo
- una visión de cómo trabajan juntos los servicios de OpenStack
- un despliegue de laboratorio simple

Sin embargo, despliegues AIO no son recomendados entornos de producción, son buenos para entornos de prueba de concepto.

Recursos de servidor mínimos:

- 8 vCPUs
- 50 GB libres de espacio de disco en la partición root
- 8 GB RAM

Recursos de servidor recomendados:

- CPU/motherboard que soporte hardware-assisted virtualization
- 8 CPU cores
- 80 GB libres de espacio de disco en la partición root o 60GB+ en un disco secundario vacío. Usar un disco secundario requiere usar el parámetro `bootstrap_host_data_disk_device`

Es posible realizar AIO builds en una máquina virtual para demostraciones y evaluación, pero nuestras VMs rendirán pobremente, excepto que activemos nested virtualization. Para cargas de trabajo de producción, se recomienda múltiples nodos para roles específicos.

17.2.1 Despliegue con Vagrant

Con el archivo Vagrantfile podremos desplegar una VM con la configuración inicial para desplegar OpenStack-Ansible AIO:

- Vagrantfile: `Descargar Vagrantfile`
- Script de ansibleaio: `Descargar ansibleaio_setup.sh`
- Archivo de texto con detalles sobre el disco de la VM: `Descargar disk_info.txt`

17.2.2 Overview

Hay 4 pasos para lograr correr un entorno AIO:

- Preparar el host
- Boostrap de Ansible y los roles requeridos
- Boostrap de la configuración AIO
- Correr los playbooks

17.2.3 Prepare the host

- Verificar la versión de kernel actual:

```
sudo -i
uname -mrs
```

- Verificar los kernels instalados en el sistema (el kernel subrayado es el que está en uso):

```
yum list installed kernel
```

- Actualizar los paquetes y kernel del sistema:

```
yum upgrade -y
```

- Reiniciar el sistema para cambiar el kernel

```
reboot
```

- Verificar la versión de kernel actualizada

```
sudo -i
uname -mrs
```

- Verificar los kernels instalados en el sistema (el kernel subrayado es el que está en uso):

```
yum list installed kernel
```

- Instalar Git:

Note: La instalación de `git` en CentOS 7, con los repositorios predeterminados, instalará una version antigua de Git, por ejemplo v1.8:

```
yum install -y git
```

Referencia: Instalar Git 2.X en CentOS 7

- Para instalar Git 2.X primero debemos habilitar el repositorio Wandisco GIT, para esto, creamos un nuevo archivo de configuración de repositorio YUM:

```
cat << EOF > /etc/yum.repos.d/wandisco-git.repo
[wandisco-git]
name=Wandisco GIT Repository
baseurl=http://opensource.wandisco.com/centos/7/git/\$basearch/
enabled=1
gpgcheck=1
gpgkey=http://opensource.wandisco.com/RPM-GPG-KEY-WANdisco
EOF
```

- Importamos las llaves GPG del repositorio:

```
rpm --import http://opensource.wandisco.com/RPM-GPG-KEY-WANdisco
```

- Ahora que hemos agregado el repositorio, podemos instalar la última versión de Git:

```
yum install -y git
```

- Revisar la versión de git (2.X)

```
git --version
```

17.2.4 Bootstrap Ansible and the required roles

Referencia: Setting up OpenStack-Ansible All-In-One on a Centos 7 system

- El script de boostrap de OpenStack-Ansible descargará e instalará su propia versión de Ansible y creará un link a `/usr/local/bin`. Por lo cual, `/usr/local/bin` debe estar en nuestra variable `$PATH`. En CentOS 7, la variable `$PATH` no contiene esta dirección, así que la debemos agregar:

```
export PATH=/usr/local/bin:$PATH
```

- Clonar el repositorio OpenStack-Ansible y cambiar al directorio raíz del repo:

```
git clone https://opendev.org/openstack/openstack-ansible /opt/openstack-ansible

cd /opt/openstack-ansible
```

Luego, deberemos cambiar al branch/tag desde el cual se implementará. Desplegar desde el head de un branch puede resultar en un build inestable. Para un build de prueba (no para un build de producción) es usualmente mejor hacer checkout de la última versión tagueada.

- Listar tags

```
git tag -l
```

- Checkout del branch estable y encontrar el último tag:

```
git checkout stable/train
git describe --abbrev=0 --tags
```

- Checkout del último tag:

```
git checkout 20.1.0
```

- Boostrap de Ansible y roles de Ansible para el entorno de desarrollo (Duración: 8:30 - 12:00 min):

```
scripts/bootstrap-ansible.sh
```

- Probar que se pueda ejecutar el comando openstack-ansible:

```
openstack-ansible
```

17.2.5 Bootstrap the AIO configuration

Para que todos los servicios corran, el host debe estar preparado con el particionamiento de disco, paquetes, configuración de red y configuraciones para el OpenStack Deployment correctos.

Por defecto, los bootstrap scripts de AIO despliegan un conjunto base de servicios OpenStack con valores predeterminados razonables con el propósito de un gate check, sistema de despliegue o sistema de pruebas.

El bootstrap script está pre-configurado para pasar la variable de entorno `BOOTSTRAP_OPTS` como una opción adicional para al proceso bootstrap.

- Para el escenario AIO predeterminado, la preparación de configuración AIO es completada ejecutando (Duración: 2:00 min):

```
scripts/bootstrap-aio.sh
```

Note: Entre todas los cambios hechos luego de ejecutar este script, se cambiará el hostname a `aio1`. Además se editará el parámetro `PasswordAuthentication` del archivo `/etc/ssh/sshd_config`, impidiendo conexiones SSH mediante contraseñas.

Para volver a permitir la conexión por SSH al sistema ejecutar:

```
sed -i 's/PasswordAuthentication no/PasswordAuthentication yes/g' /etc/ssh/sshd_config
systemctl restart sshd
```

Note:

- Para añadir OpenStack Services encima de los servicios boostrap-aio predeterminados para el escenario aplicable, copiar los archivos `conf.d` con la extensión `.aio` a `/etc/openstack_deploy` y luego renombrarlos a archivo `.yml`. Por ejemplo, para habiltar el servicio de OpenStack Telemetry, ejecutar:

```
cd /opt/openstack-ansible/

cp etc/openstack_deploy/conf.d/{aodh,gnocchi,ceilometer}.yml.aio /etc/openstack_
↪deploy/conf.d/

for f in $(ls -1 /etc/openstack_deploy/conf.d/*.aio); do mv -v ${f} ${f%.*}; done
```

- También es posible hacer esto (y cambiar otros valores predeterminados) durante la ejecución inicial del boostrap script cambiando la variable de entorno SCENARIO antes de correr el script. La palabra clave 'aio' asegurará que un conjunto de servicios básicos de OpenStack (cinder, glance, horizon, neutron, nova) sean desplegados. Las palabras claves 'lxc' y 'nspawn' pueden usarse para configurar el container back-end, mientras que la palabra 'metal' desplegará todos los servicios sin contenedores. Para implementar cualquier otro servicio, añadir el nombre del archivo `conf.d`, sin la extensión `.yml.aio` en la variable de entorno SCENARIO. Cada palabra clave debe ser limitada por un guión bajo. Por ejemplo, lo siguiente implementará un AIO con barbican, cinder, glance, horizon, neutron y nova. Configurará el almacenamiento de Cinder con Ceph back-end y usará LXC como el container back-end.

```
export SCENARIO='aio_lxc_barbican_ceph'

scripts/bootstrap-aio.sh
```

Note: Si las palabras clave 'metal' y 'aio' son usadas juntas, horizon no será desplegado porque haproxy y horizon entrarán en conflicto en los mismos puertos de escucha.

- Para añadir cualquier sobreescritura global, sobre los valores por defecto del escenario pertinente, editar `/etc/openstack_deploy/user_variables.yml`. Para poder comprender las varios formas que podemos sobreescribir el comportamiento por defecto establecido en los roles, playbooks y variables de group, ir a *OpenStack-Ansible - Overriding default configuration*

17.2.6 Run playbooks

- Finalmente correr los playbooks ejecutando:

```
cd /opt/openstack-ansible/playbooks

openstack-ansible setup-hosts.yml
# Duración: 22:30 min

openstack-ansible setup-infrastructure.yml
# Duración: 13:30 min

openstack-ansible setup-openstack.yml
# Duración: 1:25:00 horas
```

Note: El proceso de instalación tomará un tiempo para que complete pero aquí hay algunos estimados:

- Almacenamiento Bare metal con almacenamiento SSD: ~ 30-50 minutos
- Máquinas virtuale con almacenamiento SSD: ~ 45-60 minutos
- Sistemas con discos duros tradicionales: ~ 90-120 minutos

- Una vez que los playbooks han sido ejecutados completamente, es posible experimentar con varios cambios de configuración en `/etc/openstack_deploy/user_variables.yml` y solo correr playbooks individuales. Por ejemplo, para correr el playbook para el servicio de Keytone, ejecutar:

```
cd /opt/openstack-ansible/playbooks

openstack-ansible os-keystone-install.yml
```

17.3 OpenStack-Ansible CentOS 7 Multinode

Table of Contents

17.3.1 Prepare the deployment host

Referencia: OpenStack-Ansible Prepare the deployment host

Cuando instalamos OpenStack en un entorno de producción, se recomienda usar un host de despliegue separado que contenga Ansible y orqueste la instalación OpenStack-Ansible (OSA) en los target hosts. En un entorno de prueba, se recomienda usar uno de los infrastructure target hosts como el deployment host.

Configuring the operating system

Configurar al menos una interfaz de red para que acceda a Internet o a repositorios locales.

- Verificar la versión de kernel actual:

```
sudo -i
uname -mrs
```

- Verificar los kernels instalados en el sistema (el kernel subrayado es el que está en uso):

```
yum list installed kernel
```

- Actualizar los paquetes y kernel del sistema:

```
yum upgrade -y
```

- Reiniciar el sistema para cambiar el kernel

```
reboot
```

- Verificar la versión de kernel actualizada

```
sudo -i
uname -mrs
```

- Verificar los kernels instalados en el sistema (el kernel subrayado es el que está en uso):

```
yum list installed kernel
```

- Instalar el rpm de OpenStack Train de RDO:

```
yum install -y https://rdoproject.org/repos/openstack-train/rdo-release-train.rpm
```

- Instalar paquetes de software adicionales:

```
yum install -y vim ntp ntpdate openssh-server python-devel sudo '@Development Tools'
```

- Instalar Git:

Note: La instalación de `git` en CentOS 7, con los repositorios predeterminados, instalará una version antigua de Git, por ejemplo v1.8:

```
yum install -y git
```

Referencia: Instalar Git 2.X en CentOS 7

- Para instalar Git 2.X primero debemos habilitar el repositorio Wandisco GIT, para esto, creamos un nuevo archivo de configuración de repositorio YUM:

```
cat << EOF > /etc/yum.repos.d/wandisco-git.repo
[wandisco-git]
name=Wandisco GIT Repository
baseurl=http://opensource.wandisco.com/centos/7/git/\$basearch/
enabled=1
gpgcheck=1
gpgkey=http://opensource.wandisco.com/RPM-GPG-KEY-WANdisco
EOF
```

- Importamos las llaves GPG del repositorio:

```
rpm --import http://opensource.wandisco.com/RPM-GPG-KEY-WANdisco
```

- Ahora que hemos agregado el repositorio, podemos instalar la última versión de Git:

```
yum install -y git
```

- Revisar la versión de git (2.X)

```
git --version
```

- Configurar NTP para que se sincronce con una fuente de tiempo correcta.

```
# Instalar el programa chrony:
yum install -y chrony

# Reiniciar el servicio de NTP:
systemctl restart chronyd.service
```

(continues on next page)

(continued from previous page)

```
# Verificar sincronización de relojes
chronyc sources
```

- Por defecto, `firewalld` está habilitado en la mayoría de sistemas CentOS y su conjunto de reglas predeterminadas previene que algunos componentes de OpenStack se comuniquen apropiadamente. Parar el servicio `firewalld` y enmascararla para prevenirla de iniciar:

```
systemctl stop firewalld
systemctl mask firewalld
```

Configure SSH keys

Ansible usa SSH con autenticación de llave pública para conectar el deployment host y los target hosts. Para reducir la interacción del usuario durante las operaciones de Ansible, no incluir passphrases con key pairs. Sin embargo, si se requiere un passphrase, considerar usar los comandos `ssh-agent` y `ssh-add` para almacenar temporalmente el passphrase antes de realizar operaciones de Ansible.

1. Copiar los contenidos de la llave pública en el deployment host al archivo `/root/.ssh/authorized_keys` en cada target host.

```
ssh-keygen -t rsa -b 4096 -C "root@adh"
```

```
ssh-copy-id root@172.29.236.101
ssh-copy-id root@172.29.236.104
```

2. Probar la autenticación de llave pública desde el deployment host a cada target host usando SSH para conectarnos al target host desde el deployment host. Si podemos conectarnos y obtener el shell sin autenticarnos, está funcionando. SSH provee un shell sin pedirnos una contraseña.

```
ssh root@172.29.236.101 hostname
cc
ssh root@172.29.236.104 hostname
cn
```

Configure the network

Los despliegues de Ansible fallan si el deployment server no puede usar SSH para conectase a los contenedores.

Configurar el **deployment host** (donde es ejecutado Ansible) para que esté **en la misma red capa 2 como la red designada para la administración de contenedores**. Por defecto, esta es la red `br-mgmt`. Esta configuración reduce la probabilidad de falla causada por problemas de conectividad.

Seleccionar una dirección IP del siguiente ejemplo de rango para asignarlo al deployment host:

```
Container management (br-mgmt network): 172.29.236.0/22 (VLAN 10)
```

Install the source and dependencies

Referencia: Setting up OpenStack-Ansible All-In-One on a Centos 7 system

- El script de boostrap de OpenStack-Ansible descargará e instalará su propia versión de Ansible y creará un link a `/usr/local/bin`. Por lo cual, `/usr/local/bin` debe estar en nuestra variable `$PATH`. En CentOS 7, la variable `$PATH` no contiene esta dirección, así que la debemos agregar:

```
export PATH=/usr/local/bin:$PATH
```

- Clonar el repositorio OpenStack-Ansible y cambiar al directorio raíz del repo:

```
git clone https://opendev.org/openstack/openstack-ansible /opt/openstack-ansible

cd /opt/openstack-ansible
```

Luego, deberemos cambiar al branch/tag desde el cual se implementará. Desplegar desde el head de un branch puede resultar en un build inestable. Para un build de prueba (no para un build de producción) es usualmente mejor hacer checkout de la última versión tagueada.

- Listar tags

```
git tag -l
```

- Checkout del branch estable y encontrar el último tag:

```
git checkout stable/train
git describe --abbrev=0 --tags
```

- Checkout del último tag:

```
git checkout 20.1.0
```

- Boostrap de Ansible y roles de Ansible para el entorno de desarrollo (Duración: 8:00 - 12:00 min):

```
scripts/bootstrap-ansible.sh
```

- Probar que se pueda ejecutar el comando openstack-ansible (debemos tener la ruta `/usr/local/bin` en nuestra variable `$PATH`):

```
openstack-ansible --version
```

17.3.2 Prepare the target hosts

Referencia: OpenStack-Ansible Prepare the target hosts

Configuring the operating system

- Verificar la versión de kernel actual:

```
sudo -i
uname -mrs
```

- Verificar los kernels instalados en el sistema (el kernel subrayado es el que está en uso):

```
yum list installed kernel
```

- Actualizar los paquetes y kernel del sistema:

```
yum upgrade -y
```

- Deshabilitar SELinux. Editar `/etc/sysconfig/selinux`, asegurarnos de cambiar `SELINUX=enforcing` a `SELINUX=disabled`:

```
sed -i 's/SELINUX=enforcing/SELINUX=disabled/g' /etc/sysconfig/selinux
sed -i 's/SELINUX=permissive/SELINUX=disabled/g' /etc/sysconfig/selinux
```

- Reiniciar el sistema para cambiar el kernel

```
reboot
```

Note: Debemos tener una versión de kernel `3.10` o superior.

- Verificar la versión de kernel actualizada

```
sudo -i
uname -mrs
```

- Verificar los kernels instalados en el sistema (el kernel subrayado es el que está en uso):

```
yum list installed kernel
```

- Instalar paquetes de software adicionales:

```
yum install -y bridge-utils iputils lsof lvm2 chrony openssh-server sudo tcpdump␣
↪python
```

- Añadir los módulos de kernel apropiados al archivo `/etc/modules-load.d` para habiliar VLAN e interfaces bond:

```
echo 'bonding' >> /etc/modules-load.d/openstack-ansible.conf
echo '8021q' >> /etc/modules-load.d/openstack-ansible.conf
```

- Configurar NTP en `/etc/chrony.conf` para que se sincronce con una fuente de tiempo correcta.

```
# Instalar el programa chrony:
yum install -y chrony

# Reiniciar el servicio de NTP:
systemctl restart chronyd.service

# Verificar sincronización de relojes
chronyc sources
```

- (Opcional) Reducir el nivel de log del kernel cambiando el valor `printk` en nuestro `sysctls`:

```
echo "kernel.printk='4 1 7 4'" >> /etc/sysctl.conf
```

- Reiniciar el host para activar los cambios y usar el nuevo kernel

Configure SSH keys

Ansible usa SSH para conectar el deployment host a los target host.

1. Copiar los contenidos de la llave pública en el deployment host al archivo `/root/.ssh/authorized_keys` en cada target host.
2. Probar la autenticación de llave pública desde el deployment host a cada target host usando SSH para conectarnos al target host desde el deployment host. Si podemos conectarnos y obtener el shell sin autenticarnos, está funcionando. SSH provee un shell sin pedirnos una contraseña.

Note: Estos pasos se han realizado en el deployment host (*Prepare the deployment host*)

Important: Los depliegues OpenStack-Ansible requieren la presencia de un archivo `/root/.ssh/id_rsa.pub` en el deployment host. Los contenidos de este archivo se insertan en un archivo `authorized_keys` para los contenedores, lo cual es un paso necesario para los Ansible playbooks. Podemos sobreescribir este comportamiento configurando la variable `lxc_container_ssh_key` con la llave pública para el contenedor.

Configuring the storage

Logical Volume Manager (LVM) habilita a un único dispositivo ser dividido en múltiples volúmenes lógicos que aparecen como un dispositivo de almacenamiento físico al sistema operativo. El servicio de **Block Storage (Cinder)**, y **LXC containers** que corren opcionalmente la infraestructura de OpenStack, opcionalmente pueden usar LVM para su almacenamiento de datos.

Note: OpenStack-Ansible configura LVM automáticamente en los nodos, y sobreescribe cualquier configuración LVM. Si tenemos una configuración LVM personalizada, editar el archivo de configuración generado como se necesite.

---

1. Para usar el servicio de Block Storage (cinder), crear un LVM volume group llamado `cinder-volumes` en el storage host. Especificar un tamaño de metadata de 2048 cuando creamos el volumen físico. Por ejemplo:

```
pvcreate --metadatasize 2048 physical_volume_device_path
vgcreate cinder-volumes physical_volume_device_path
```

2. Opcionalmente, crear un LVM volume group llamado `lxc` para container file systems si deseamos usar LXC con LVM. Si el `lxc` volume group no existe, los cotenedores serán instalados automáticamente en el file system bajo `/var/lib/lxc` por defecto.

Configuring the network

OpenStack-Ansible usa **bridges** para conectar **interfaces de red físicas y lógicas** en el **host** a **interfaces de red virtuales** dentro de **contenedores**. Los target hosts deben ser configurados con los siguientes network bridges:

Nombre del Bridge	Mejor configurado en	Con IP estática
br-mgmt	En cada nodo	Siempre
br-storage	En cada nodo de storage	Cuando el componente se despliega en metal
br-storage	En cada nodo de compute	Siempre
br-vxlan	En cada nodo de network	Cuando el componente se despliega en metal
br-vxlan	En cada nodo de compute	Siempre
br-vlan	En cada nodo de network	Nunca
br-vlan	En cada nodo de compute	Nunca

Para una referencia detallada sobre cómo implementar el host y container networking, ir a OpenStack-Ansible Architecture - Container networking

Para ejemplos de casos de uso, ir a OpenStack-Ansible User Guide

Host network bridges information

- **LXC internal `lxcbr0`**
  - El bridge `lxcbr0` es requerido para LXC, pero OpenStack-Ansible lo configura automáticamente. Provee conectividad externa (generalmente Internet) a los contenedores con dnsmasq (DHCP/DNS) + NAT.
  - Este bridge no se conecta directamente a ninguna interfaz física o lógica en el host, pues iptables maneja la conectividad. El bridge se conecta a `eth0` en cada contenedor.
  - La red de contenedor a la que el bridge se conecta es configurable en el archivo `openstack_user_config.yml` en el diccionario `provider_networks`.
- **Container management: `br-mgmt`**
  - El `br-mgmt` provee administración y comunicación entre la infraestructura y servicios de OpenStack.
  - El bridge se conecta a una interfaz física o lógica, generalmente una subinterfaz VLAN `bond0`. También se conecta a `eth1` en cada contenedor.
  - La interfaz de red del contenedor a la que se conecta el bridge es configurable en el archivo `openstack_user_config.yml`.
- **Storage: `br-storage`**
  - El bridge `br-storage` provee acceso segregado a dispositivos Block Storage entre servicios servicios de OpenStack y dispositivos Block Storage.
  - El bridge se conecta a una interfaz física o lógica, generalmente una subinterfaz VLAN `bond0`. También se conecta a `eth2` en cada contenedor asociado.
  - La interfaz de red del contenedor a la que se conecta el bridge es configurable en el archivo `openstack_user_config.yml`.
- **OpenStack Networking tunnel: `br-vxlan`**
  - El bridge `br-vxlan` es **requerido si** el entorno es configurado para que permita a proyectos crear redes virtuales usando VXLAN. Provee la interfaz para **redes de túneles virtuales (VXLAN)**.
  - El bridge se conecta a una interfaz física o lógica, generalmente una subinterfaz VLAN `bond1`. También se conecta a `eth10` en cada contenedor asociado.
  - La interfaz de red del contenedor a la que se conecta el bridge es configurable en el archivo `openstack_user_config.yml`.
- **OpenStack Networking provider: `br-vlan`**
  - El bridge `br-vlan` provee infraestructura para **redes VLAN tagged** o **flat (sin VLANT tag)**.
  - El bridge se conecta a una interfaz física o lógica, generalmente una subinterfaz VLAN `bond1`. Se conecta a `eth11` para redes tipo VLAN en cada contenedor asociado. No se le asigna una dirección IP porque maneja solo conectividad capa 2.
  - La interfaz de red del contenedor a la que se conecta el bridge es configurable en el archivo `openstack_user_config.yml`.

17.3.3 Configure the deployment

Referencia: OpenStack-Ansible Configure the deployment

Ansible referencia algunos archivos que contienen directivas de configuración obligatorias y opcionales. Antes de poder correr los Ansible playbooks, modificar estos archivos para definir el entorno del target. Algunas tareas de configuración incluyen:

- Target host networking para definir las interfaces bridge y redes.
- Una lista de target hosts en los cuales instalar el software.
- Relaciones de redes virtuales y físicas para OpenStack Networking (neutron).
- Contraseñas para todos los servicios.

Initial environment configuration

OpenStack-Ansible (OSA) depende de varios archivos que son usados para contruir un inventario para Ansible. Realizar la siguiente configuración en el deployment host:

1. Copiar los contenidos del directorio `/opt/openstack-ansible/etc/openstack_deploy` al directorio `/etc/openstack_deploy`.

```
cp -R /opt/openstack-ansible/etc/openstack_deploy /etc
```

2. Movernos al directorio `/etc/openstack_deploy`.

```
cd /etc/openstack_deploy
```

3. Copiar el archivo `openstack_user_config.yml.example` a `/etc/openstack_deploy/openstack_user_config.yml`.

```
cp openstack_user_config.yml.example /etc/openstack_deploy/openstack_user_config.yml
```

4. Revisar el archivo `openstack_user_config.yml` y realizar cambios al despliegue de nuestro entorno OpenStack.

```
vim openstack_user_config.yml
```

Note: Este archivo está muy comentado con detalles sobre las diversas opciones. Ver OpenStack-Ansible User Guide y OpenStack-Ansible Reference Guide para más detalles.

La configuración en el archivo `openstack_user_config.yml` define qué hosts corren los contenedores y servicios desplegados por OpenStack-Ansible. Por ejemplo, los hosts listados en la sección `shared-infra_hosts` corren contenedores para muchos de los servicios compartidos que nuestro entorno OpenStack requiere. Algunos de estos servicios incluyen bases de datos, Memcached, y RabbitMQ. Muchos otros tipos de hosts contiene otros tipos de contenedores, y todos estos son listados en el archivo `openstack_user_config.yml`.

Algunos servicios, como glance, heat, horizon, nova-infra, no son listados individualmente en el archivo de ejemplo al estar contenidos en los hosts de os-infra. Podemos especificar image-hosts o dashboard-hosts si queremos escalar de una manera específica.

Para ejemplos, ver OpenStack-Ansible User Guide.

Para detalles sobre cómo es generado el inventario, de la configuración de entorno y la precendencia de variable, ver OpenStack-Ansible Reference Guide, en la seccion de inventario.

5. Revisar el archivo `user_variables.yml` para configurar opciones globales y de despliegues de roles específicos. El archivo contiene algunas variables de ejemplos y comentario pero podemos obtener la lista completa de variables en cada documentación específica del rol.

```
vim user_variables.yml
```

Note: Una variable importante es `install_method`, la cual configura el método de instalación para los servicios de OpenStack. Los servicios pueden ser desplegados desde la fuente (por defecto) o de paquetes de distribución. Los despliegues basados en la fuente son más cercanos a instalaciones de OpenStack vanilla y permiten más cambios y personalizaciones. Del otro lado, despliegues basados en distros generalmente proveen una combinación de paquetes que han sido verificados por las mismas distribuciones. Sin embargos, esto significa que las actualizaciones son lanzadas con menos frecuencia y con retrasos potenciales. Aun más, este método puede ofrecernos menos oportunidades para personalizaciones de despliegues. La variable `install_method` es configurada durante el despliegue inicial y no podremos cambiarla, pues OpenStack-Ansible no puede convertirse de un método de instalación a otro. Como tal, es importante juzgar nuestras necesidades con los pros y contras de cada método antes de tomar una decisión. Notar que el método de instalación `distro` fue introducido en el ciclo Rocky, y como resutado, Ubuntu 16.04 no es soportado debido al hecho de que no hay paquetes de Rocky para este.

Installing additional services

Para instalar servicios adicionales, los archivos en `etc/openstack_deploy/conf.d` proveen ejemplos mostrando los host groups correctos para usar. Para añadir otro servicio: añadir el host group, asignarle hosts y luego ejecutar los playbooks.

Advanced service configuration

OpenStack-Ansible tiene muchas opciones que podemos usar para la configuración avanzada de servicios. Cada documentación de rol provee información sobre las opciones disponibles.

Important: Este paso es esencial para adaptar OpenStack-Ansible a nuestras necesidades y es generalmente pasado por alto por nuevos deployers. Hechar un vistazo a cada documentación de rol, guía de usuario y referenciarnos a estas si deseamos un cloud adaptada a nuestra necesidad.

Configuring service credentials

Configurar credenciales para cada servicio en el archivo `/etc/openstack_deploy/user_secrets.yml`. Considerar usar el feature Ansible Vault para incrementar la seguridad encriptando archivos que contengan credenciales.

```
vi user_secrets.yml
```

Ajustar permisos en estos archivos para restringir acceso por usuario no-privilegiados.

La opción `keystone_auth_admin_password` configura la contraseña tenant `admin` tanto para el acceso OpenStack API y Dashboard.

Se recomienda usar el script `pw-token-gen.py` para generar valores aleatorios para las variables en cada archivo que contiene credenciales de servicio:

```
cd /opt/openstack-ansible
./scripts/pw-token-gen.py --file /etc/openstack_deploy/user_secrets.yml
```

Para regenerar las contraseñas existentes, añadir el flag `--regen`.

Warning: Los playbooks no soportan cambiar contraseñas en un entorno existente. Cambiar contraseñas y volver a correr los playbooks causará fallos que puedan malograr nuestro entorno OpenStack.

17.3.4 Run playbooks

Referencia: OpenStack-Ansible Run playbooks

Los procesos de instalación requieren correr 3 playbooks:

- El playbook base de Ansible `setup-hosts.yml` prepara los target hosts para servicios de infraestructura y de OpenStack, construye y reinicia contenedores en target hosts, e instala componentes comúnes en contenedores de target hosts.
- El playbook de infraestructura de Ansible `setup-infrastructure.yml` instala servicios de infraestructura: Memcached, el servicio de repositorio, Galera, RabbitMQ, and rsyslog.
- El playbook de OpenStack `setup-openstack.yml` instala servicios de OpenStack, incluyendo Identity (keystone), Image (glance), Block Storage (cinder), Compute (nova), Networking (neutron), etc.

Checking the integrity of the configuration files

Antes de correr cualquier playbook, revisar la integridad de los archivos de configuración.

1. Asegurar que todos los archivos editados en el directorio `/etc/openstack_deploy` son compatibles con Ansible YAML.
2. Revisar la integridad de nuestros archivos YAML.

Note: Para revisar la sintáxis de nuestros archivo YAML podemos usar el Programa YAML Lint.

3. Movernos al directorio `/opt/openstack-ansible/playbooks` y correr el comando:

```
openstack-ansible setup-infrastructure.yml --syntax-check
```

4. Volver a revisar que toda la identación es correcta. Esto es importante porque la sintáxis de los archivos de configuración puede ser correcta sin ser significativa para OpenStack-Ansible.

Run the playbooks to install OpenStack

1. Movernos al directorio `/opt/openstack-ansible/playbooks`.
2. Correr el playbook de configuración del host:

```
openstack-ansible setup-hosts.yml
```

Confirmar la finalización satisfactoria con cero elementos inalcanzables o fallidos:

```
PLAY RECAP ********************************************************************
...
deployment_host                : ok=18   changed=11   unreachable=0    failed=0
```

3. Correr el playbook de configuración de la infraestructura:

```
openstack-ansible setup-infrastructure.yml
```

Confirmar la finalización satisfactoria con cero elementos inalcanzables o fallidos:

```
PLAY RECAP ********************************************************************
...
deployment_host                : ok=27   changed=0    unreachable=0    failed=0
```

4. Correr los siguientes comandos para verificar el database cluster:

```
ansible galera_container -m shell -a "mysql -h localhost -e 'show status like \"
↪%wsrep_cluster_%\";'"
```

Output de ejemplo:

```
node3_galera_container-3ea2cbd3 | success | rc=0 >>
Variable_name             Value
wsrep_cluster_conf_id     17
wsrep_cluster_size        3
wsrep_cluster_state_uuid  338b06b0-2948-11e4-9d06-bef42f6c52f1
wsrep_cluster_status      Primary

node2_galera_container-49a47d25 | success | rc=0 >>
Variable_name             Value
wsrep_cluster_conf_id     17
wsrep_cluster_size        3
wsrep_cluster_state_uuid  338b06b0-2948-11e4-9d06-bef42f6c52f1
wsrep_cluster_status      Primary

node4_galera_container-76275635 | success | rc=0 >>
Variable_name             Value
wsrep_cluster_conf_id     17
wsrep_cluster_size        3
wsrep_cluster_state_uuid  338b06b0-2948-11e4-9d06-bef42f6c52f1
wsrep_cluster_status      Primary
```

El campo `wsrep_cluster_size` indica el número de nodos en el clúster y el campo `wsrep_cluster_status` indica que es primario.

5. Correr el playbook de configuración de OpenStack:

```
openstack-ansible setup-openstack.yml
```

Confirmar la finalización satisfactoria con cero elementos inalcanzables o fallidos.

Note: Guardar el output de los comandos impresos en el terminal en un archivo, incluyendo mensajes de advertencia y errores (`2>&1`):

```
openstack-ansible setup-hosts.yml 2>&1 | tee /root/setup-hosts.log
```

17.3.5 Verifying OpenStack operation

Referencia: OpenStack-Ansible Verifying OpenStack operation

Para verificar la operación básica de OpenStack API y el Dashboard, realizar las siguientes tareas en un host de infraestructura.

Verify the API

El **utility container** provee un entorno CLI para configuraciones adicionales y pruebas.

1. Determinar el nombre del utility container:

```
lxc-ls | grep utility
```

2. Acceder al utility container:

```
lxc-attach -n infra1_utility_container-161a4084
```

3. Obtener las credenciales de tenant `admin`:

```
. ~/openrc
```

4. Listar los usuarios de openstack:

```
openstack user list --os-cloud=default
```

Verifying the Dashboard (horizon)

1. Dentro de un navegador web, acceder al Dashboard usando la dirección IP del load balancer externo definido por la opción `external_lb_vip_address` en el archivo `/etc/openstack_deploy/openstack_user_config.yml`. El Dashboard usa HTTPS en el puerto 443.
2. Autenticarse usando el nombre de usuario `admin` y la contraseña definida por la opción `keystone_auth_admin_password` en el archivo `/etc/openstack_deploy/user_secrets.yml`.

17.3.6 Resumen de archivos del repositorio `openstack-ansible/`

- **Archivos de configuración:**
  - Configuration for OSA core services: `openstack-ansible/etc/openstack_deploy/openstack_user_config.yml`
  - **Optional servicie configuration: `openstack-ansible/etc/openstack_deploy/conf.d/*`**
    - aodh.yml.aio
    - aodh.yml.example
    - barbican.yml.aio
    - barbican.yml.example
    - blazar.yml.aio
    - ceilometer.yml.aio

* ceilometer.yml.example
* ceph.yml.aio
* cinder.yml.aio
* congress.yml.aio
* designate.yml.aio
* designate.yml.example
* etcd.yml.aio
* etcd.yml.example
* glance.yml.aio
* gnocchi.yml.aio
* haproxy.yml.aio
* haproxy.yml.example
* heat.yml.aio
* horizon.yml.aio
* ironic.yml.aio
* keystone.yml.aio
* magnum.yml.aio
* magnum.yml.example
* manila.yml.aio
* manila.yml.example
* masakari.yml.aio
* masakari.yml.example
* mistral.yml.aio
* mistral.yml.example
* murano.yml.aio
* murano.yml.example
* neutron.yml.aio
* nova.yml.aio
* octavia.yml.aio
* panko.yml.aio
* panko.yml.example
* placement.yml.aio
* placement.yml.example
* qdrouterd.yml.aio
* rally.yml.aio
* sahara.yml.aio

        * swift-remote.yml.sample
        * swift.yml.aio
        * swift.yml.example
        * tacker.yml.aio
        * trove.yml.aio
        * trove.yml.example
        * unbound.yml.aio
        * unbound.yml.example
    - Opciones globales y de despliegues de roles específicos: `openstack-ansible/etc/openstack_deploy/user_variables.yml`
    - Credenciales para cada servicio: `openstack-ansible/etc/openstack_deploy/user_secrets.yml`
- **Archivos de ejecución (playbooks):**
    - **`openstack-ansible/playbooks/setup-hosts.yml`**
        * import_playbook: openstack-hosts-setup.yml
        * import_playbook: security-hardening.yml
        * import_playbook: containers-deploy.yml
    - **`openstack-ansible/playbooks/setup-infrastructure.yml`**
        * import_playbook: unbound-install.yml
        * import_playbook: repo-install.yml
        * import_playbook: haproxy-install.yml
        * import_playbook: utility-install.yml
        * import_playbook: memcached-install.yml
        * import_playbook: galera-install.yml
        * import_playbook: qdrouterd-install.yml
        * import_playbook: rabbitmq-install.yml
        * import_playbook: etcd-install.yml
        * import_playbook: ceph-install.yml
        * import_playbook: rsyslog-install.yml
    - **openstack-ansible/playbooks/setup-openstack.yml**
        * import_playbook: os-keystone-install.yml
        * import_playbook: os-barbican-install.yml
        * import_playbook: os-placement-install.yml
        * import_playbook: os-glance-install.yml
        * import_playbook: os-cinder-install.yml
        * import_playbook: os-nova-install.yml
        * import_playbook: os-neutron-install.yml

    * import_playbook: os-heat-install.yml
    * import_playbook: os-horizon-install.yml
    * import_playbook: os-designate-install.yml
    * import_playbook: os-gnocchi-install.yml
    * import_playbook: os-swift-install.yml
    * import_playbook: os-ceilometer-install.yml
    * import_playbook: os-aodh-install.yml
    * import_playbook: os-panko-install.yml
    * import_playbook: os-ironic-install.yml
    * import_playbook: os-magnum-install.yml
    * import_playbook: os-trove-install.yml
    * import_playbook: os-sahara-install.yml
    * import_playbook: os-octavia-install.yml
    * import_playbook: os-tacker-install.yml
    * import_playbook: os-blazar-install.yml
    * import_playbook: os-masakari-install.yml
    * import_playbook: os-manila-install.yml
    * import_playbook: os-mistral-install.yml
    * import_playbook: os-murano-install.yml
    * import_playbook: ceph-rgw-install.yml
    * import_playbook: os-congress-install.yml
    * import_playbook: os-tempest-install.yml
    * import_playbook: os-rally-install.yml

17.3.7 Referencias

- OpenStack-Ansible Reference Guide
- OpenStack-Ansible User Guide
- OpenStack-Ansible User Guide - Network architectures
- OpenStack-Ansible User Guide - Test environment example
- OpenStack-Ansible Architecture
- OpenStack-Ansible Architecture - Container Networking
- OpenStack-Ansible - Project Update

Flujo de instalación de OpenStack-Ansible:

- OpenStack-Ansible Prepare the deployment host
- OpenStack-Ansible Prepare the target hosts
- OpenStack-Ansible Configure the deployment

- OpenStack-Ansible Run playbooks
- OpenStack-Ansible Verifying OpenStack operation

CHAPTER 18

PXE

CHAPTER 19

PXE Guides

19.1 PXE Install

Table of Contents

Documentación con el proceso de instalación y configuración de herramientas para habilitar un servidor PXE. Este servidor permitará la instalación por red del sistema operativo CentOS 7 en clientes PXE.

19.1.1 Vagrantfile - PXE Server

Archivo `Vagrantfile` para crear una VM con sistema operativo CentOS 7 usando Vagrant, a la cual se le instalará las herramientas para que funcione como servidor PXE. Los detalles de la VM creada en VirtualBox son los siguientes:

- Sistema operativo CentOS 7 con 4 GB de RAM, 2 vCPUs y 20 GB de almacenamiento
- **Tiene 2 interfaces de red:**

  - 1 interfaz conectada a NAT (creada por defecto por Vagrant): 10.0.2.15/24
  - 1 interfaz conectada a una red aislada: 192.168.8.8/24

- Pasamos un script para una configuración inicial: `pxeserver_setup.sh`

```
# -*- mode: ruby -*-
# vi: set ft=ruby :
servers=[
{
    :hostname => "pxeserver",
    :box => "geerlingguy/centos7",
    :ram => 4096,
    :cpu => 2,
    :disk => "20GB",
    :script => "sh /vagrant/pxeserver_setup.sh"
}
]

Vagrant.configure("2") do |config|
servers.each do |machine|
    config.vm.define machine[:hostname] do |node|
    node.vm.box = machine[:box]
    node.vm.hostname = machine[:hostname]
    node.vm.provider "virtualbox" do |vb|
        vb.customize ["modifyvm", :id, "--memory", machine[:ram], "--cpus",␣
↪machine[:cpu]]
        #vb.customize ["modifyvm", :id, "--nic2", "hostonly", "--hostonlyadapter2",
↪"vboxnet0"]
    end
    node.vm.provision "shell", inline: machine[:script], privileged: true, run: "once"
    end
end

config.vm.define "pxeserver" do |pxeserver|
    pxeserver.vm.network "private_network", ip: '192.168.8.8', virtualbox__intnet:␣
↪true, adapter: 2
end
end
```

script `pxeserver_setup.sh`:

```
#! /bin/sh

cat << EOF > /etc/sysconfig/network-scripts/ifcfg-enp0s8
DEVICE="enp0s8"
DEFROUTE="no"
BOOTPROTO="static"
IPADDR="192.168.8.8"
NETMASK="255.255.255.0"
DNS1="8.8.8.8"
TYPE="Ethernet"
ONBOOT=yes
EOF

ifdown enp0s8
ifup enp0s8
```

19.1.2 Configuración de PXE Server

Instalación de herramientas

El servidor PXE contará con servicios que deberán ser instalados en la VM:

- `syslinux`: provee distintos bootloaders
- `dhcp`: ofrece el servicio de DHCP que asignará IPs a todos los clientes conectados en la red interna y booteen usando PXE.
- `tftp-server`: ofrece el servicio de TFTP (Trivial FTP) que proveerá los bootloaders necesarios (de syslinux) a los clientes PXE.
- `vsftpd`: ofrece el servicio de FTP que compartirá el medio de instalación del Sistema Operativo (en nuestro caso CentOS 7) a los clientes PXE. Otras opciones a usarse podrían ser NFS o HTTP.

```
$ sudo yum install -y syslinux dhcp tftp-server vsftpd
```

Instalar otras herramientas de utilidad:

```
$ sudo yum install -y vim wget
```

Configuración del servicio DHCP

El servicio DHCP será configurado con el archivo `/etc/dhcp/dhcpd.conf`:

```
sudo bash -c 'cat << EOF > /etc/dhcp/dhcpd.conf
#DHCP configuration for PXE boot server
ddns-update-style interim;
ignore client-updates;
authoritative;
allow booting;
allow bootp;
allow unknown-clients;
#DHCP pool
subnet 192.168.8.0
netmask 255.255.255.0
{
range 192.168.8.100 192.168.8.199;
option domain-name-servers 192.168.8.2;
option domain-name "midominio.com";
option routers 192.168.8.2;
option broadcast-address 192.168.8.255;
default-lease-time 600;
max-lease-time 7200;
#PXE boot server
next-server 192.168.8.8;
filename "pxelinux.0";
}
# Static lease, fixed IP: hostname-MAC-IP
host vm2-pxe-client
{
option host-name    "vm2.host";
hardware ethernet   00:11:22:33:44:55;
fixed-address       192.168.8.50;
}
EOF'
```

- Se tiene un pool de direcciones IP (192.168.8.100 - 192.168.8.199) que se pueden asignar a los cliente conectados a la red.
- Para una MAC específica se asignará un hostname e IP predeterminados. Esto es útil para controlar la asignación de IPs en los clientes PXE.

Note: Output redirection (e.g., >) is performed by bash, not by cat, while running with your UID. To run with root's UID use `sudo bash -c`

Habilitación de servicios

Habilitaremos e iniciaremos los servicios de DHCP, TFTP y FTP:

```
cat << EOF > enable_pxe_services.sh
#!/bin/bash
#DHCP
systemctl start dhcpd.service && systemctl enable dhcpd.service
#TFTP
systemctl start tftp.service && systemctl enable tftp.service
#FTP
systemctl start vsftpd.service && systemctl enable vsftpd.service
systemctl status dhcpd tftp vsftpd
EOF

chmod +x enable_pxe_services.sh
sudo ./enable_pxe_services.sh
```

Configuración de servicios para habilitar PXE

- Copiar los bootloaders que provee `syslinux` a la carpeta TFTP:

```
$ sudo cp -v /usr/share/syslinux/pxelinux.0 /var/lib/tftpboot/
$ sudo cp -v /usr/share/syslinux/menu.c32 /var/lib/tftpboot/
$ sudo cp -v /usr/share/syslinux/mboot.c32 /var/lib/tftpboot/
$ sudo cp -v /usr/share/syslinux/chain.c32 /var/lib/tftpboot/
```

- Crear los directorios necesarios para FTP y TFTP:

```
$ sudo mkdir /var/lib/tftpboot/pxelinux.cfg
$ sudo mkdir -p /var/lib/tftpboot/networkboot/centos7
$ sudo mkdir /var/ftp/pub/centos7
```

- Descargar el ISO de CentOS 7 'Minimal ISO' usando `wget` con un mirror habilitado:

```
$ wget http://mirror.unimagdalena.edu.co/centos/7.7.1908/isos/x86_64/CentOS-7-x86_64-
↪Minimal-1908.iso
```

- Montar el ISO de CentOS 7:

```
$ sudo mount -o loop CentOS-7-x86_64-Minimal-1908.iso /mnt/
```

- Copiar el contenido del ISO montado de CentOS al directorio que hemos creado:

```
$ sudo cp -rf /mnt/* /var/ftp/pub/centos7
```

- Copiar el archivo de init RAM disk (`initrd`) y el archivo de kernel `vmlinuz` de CentOS al directorio de TFTP `/var/lib/tftpboot/networkboot/centos7/`:

```
$ sudo cp /var/ftp/pub/centos7/images/pxeboot/initrd.img /var/lib/tftpboot/
↪networkboot/centos7/
$ sudo cp /var/ftp/pub/centos7/images/pxeboot/vmlinuz /var/lib/tftpboot/networkboot/
↪centos7/
```

- Crear el un archivo de configuracion del menú de PXE Boot:

```
sudo bash -c 'cat << EOF > /var/lib/tftpboot/pxelinux.cfg/default
default menu.c32
prompt 0
timeout 30
menu title PXE Boot CentOS 7
label Instalar CentOS 7 (Minimal)
kernel /networkboot/centos7/vmlinuz
append initrd=/networkboot/centos7/initrd.img inst.repo=ftp://192.168.8.8/pub/centos7
EOF'
```

En esta entrada del menú de PXE Boot indicamos el **kernel** (`vmlinuz`), el **init RAM disk** (`initrd.img`) y el repositorio de donde se va a descargar los paquetes, en este caso localmente. El kernel y el initrd se pasan a través de **TFTP**, mientras que los archivos de repositorio se pasan mediante **FTP**. Además se añade una etiqueta que idenfique esta entrada para instalar CentOS en el menú.

- Reiniciar los servicios de DHCP, TFTP y FTP

```
cat << EOF > restart_pxe_services.sh
#!/bin/bash
sudo systemctl restart dhcpd
sudo systemctl restart tftp
sudo systemctl restart vsftpd
systemctl status dhcpd tftp vsftpd
EOF

chmod +x restart_pxe_services.sh
./restart_pxe_services.sh
```

Configuración de firewall (OPCIONAL)

Si nuestro sistema tiene activo el servicio de firewall o tiene reglas de firewalls activadas debemos configurar ciertos servicios:

- Agregar reglas al firewall de los servicios:

```
cat << EOF > firewall_pxe_services.sh
#!/bin/bash
#DHCP
sudo firewall-cmd --permanent --add-service={dhcp,proxy-dhcp}
#TFTP
sudo firewall-cmd --permanent --add-service=tftp
#FTP
sudo firewall-cmd --permanent --add-service=ftp
sudo firewall-cmd --reload
```

(continues on next page)

(continued from previous page)

```
sudo firewall-cmd --state
systemctl status firewalld
EOF

chmod +x firewall_pxe_services.sh
./firewall_pxe_services.sh
```

- Agregar una regla de `iptables` para que no existan problemas de conexión del cliente PXE:

```
$ sudo iptables -I INPUT -i enp0s8 -p udp --dport 67:68 --sport 67:68 -j ACCEPT
```

19.1.3 Script de automatización - Configuración de PXE Server

El siguiente script automatiza (agrupa) todos los pasos desarrollados en la sección anterior:

```
sudo yum install -y syslinux dhcp tftp-server vsftpd
sudo yum install -y vim wget

sudo bash -c 'cat << EOF > /etc/dhcp/dhcpd.conf
#DHCP configuration for PXE boot server
ddns-update-style interim;
ignore client-updates;
authoritative;
allow booting;
allow bootp;
allow unknown-clients;
#DHCP pool
subnet 192.168.8.0
netmask 255.255.255.0
{
range 192.168.8.100 192.168.8.199;
option domain-name-servers 192.168.8.2;
option domain-name "midominio.com";
option routers 192.168.8.2;
option broadcast-address 192.168.8.255;
default-lease-time 600;
max-lease-time 7200;
#PXE boot server
next-server 192.168.8.8;
filename "pxelinux.0";
}
# Static lease, fixed IP: hostname-MAC-IP
host vm2-pxe-client
{
option host-name    "vm2.host";
hardware ethernet   00:11:22:33:44:55;
fixed-address       192.168.8.50;
}
EOF'

cat << EOF > enable_pxe_services.sh
#!/bin/bash
#DHCP
systemctl start dhcpd.service && systemctl enable dhcpd.service
#TFTP
```

(continues on next page)

(continued from previous page)

```
systemctl start tftp.service && systemctl enable tftp.service
#FTP
systemctl start vsftpd.service && systemctl enable vsftpd.service
systemctl status dhcpd tftp vsftpd
EOF

chmod +x enable_pxe_services.sh
sudo ./enable_pxe_services.sh

#cat << EOF > firewall_pxe_services.sh
##!/bin/bash
##DHCP
#sudo firewall-cmd --permanent --add-service={dhcp,proxy-dhcp}
##TFTP
#sudo firewall-cmd --permanent --add-service=tftp
##FTP
#sudo firewall-cmd --permanent --add-service=ftp
#sudo firewall-cmd --reload
#sudo firewall-cmd --state
#systemctl status firewalld
#EOF
#
#chmod +x firewall_pxe_services.sh
#./firewall_pxe_services.sh

sudo cp -v /usr/share/syslinux/pxelinux.0 /var/lib/tftpboot/
sudo cp -v /usr/share/syslinux/menu.c32 /var/lib/tftpboot/
sudo cp -v /usr/share/syslinux/mboot.c32 /var/lib/tftpboot/
sudo cp -v /usr/share/syslinux/chain.c32 /var/lib/tftpboot/

sudo mkdir /var/lib/tftpboot/pxelinux.cfg
sudo mkdir -p /var/lib/tftpboot/networkboot/centos7
sudo mkdir /var/ftp/pub/centos7

wget http://mirror.unimagdalena.edu.co/centos/7.7.1908/isos/x86_64/CentOS-7-x86_64-
↪Minimal-1908.iso
sudo mount -o loop CentOS-7-x86_64-Minimal-1908.iso /mnt/

sudo cp -rf /mnt/* /var/ftp/pub/centos7

sudo cp /var/ftp/pub/centos7/images/pxeboot/initrd.img /var/lib/tftpboot/networkboot/
↪centos7/
sudo cp /var/ftp/pub/centos7/images/pxeboot/vmlinuz /var/lib/tftpboot/networkboot/
↪centos7/

sudo bash -c 'cat << EOF > /var/lib/tftpboot/pxelinux.cfg/default
default menu.c32
prompt 0
timeout 30
menu title PXE Boot CentOS 7
label Instalar CentOS 7 (Minimal)
kernel /networkboot/centos7/vmlinuz
append initrd=/networkboot/centos7/initrd.img inst.repo=ftp://192.168.8.8/pub/centos7
EOF'

cat << EOF > restart_pxe_services.sh
#!/bin/bash
```

(continues on next page)

(continued from previous page)

```
sudo systemctl restart dhcpd
sudo systemctl restart tftp
sudo systemctl restart vsftpd
systemctl status dhcpd tftp vsftpd
EOF

chmod +x restart_pxe_services.sh
./restart_pxe_services.sh

# sudo iptables -I INPUT -i enp0s8 -p udp --dport 67:68 --sport 67:68 -j ACCEPT
```

19.1.4 Instalando CentOS 7 en un cliente PXE

Para instalar CentOS 7 en un cliente PXE de VirtualBox configuraremos lo siguiente en una VM creada:

- Un mínimo de 2048 MB de RAM
- Boot order: primero Hard Disk y luego Network

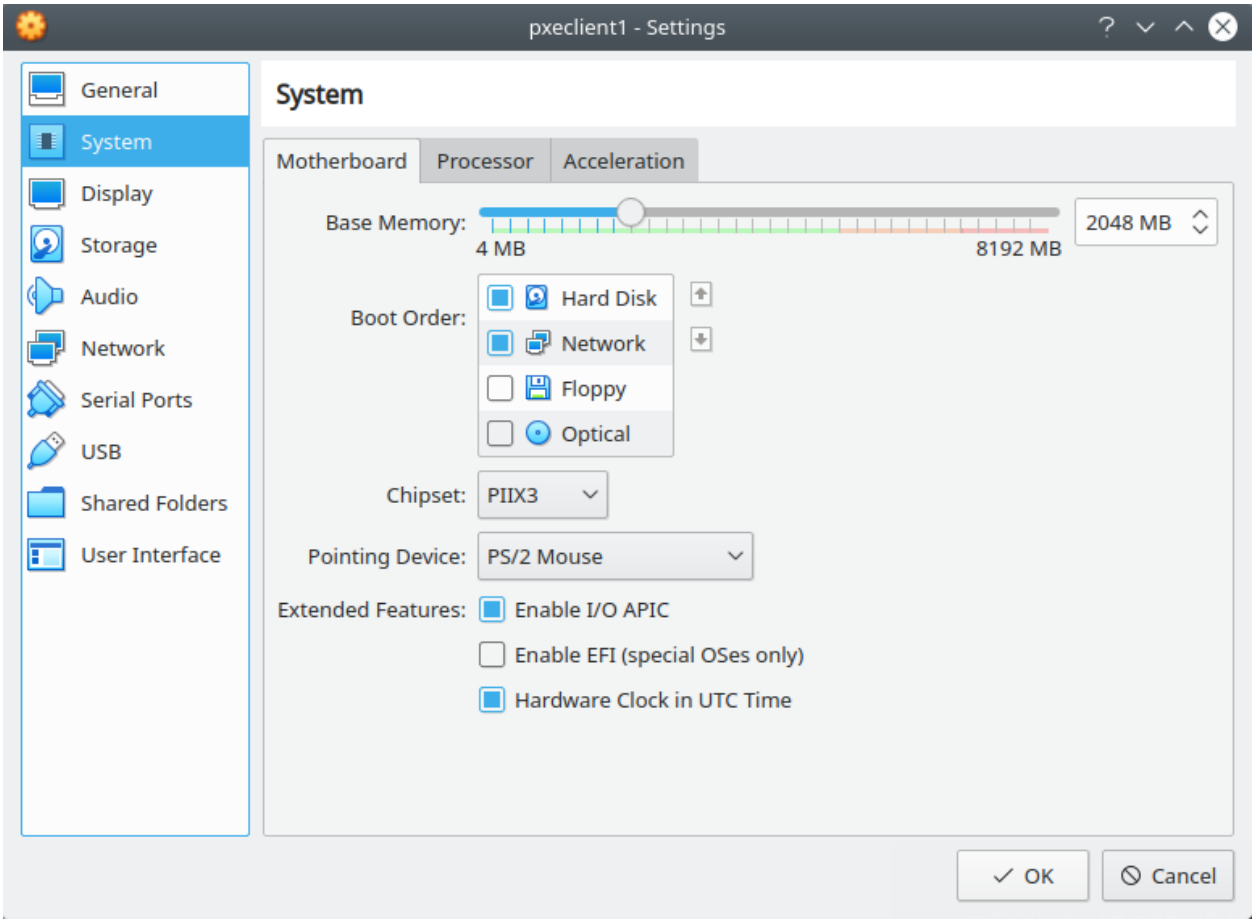


Fig. 1: VirtualBox - PXE Client - System Tab

- Una interfaz de red conectada en la misma red por donde el PXE server ofrece sus servicios de DHCP, FTP, TFTP:

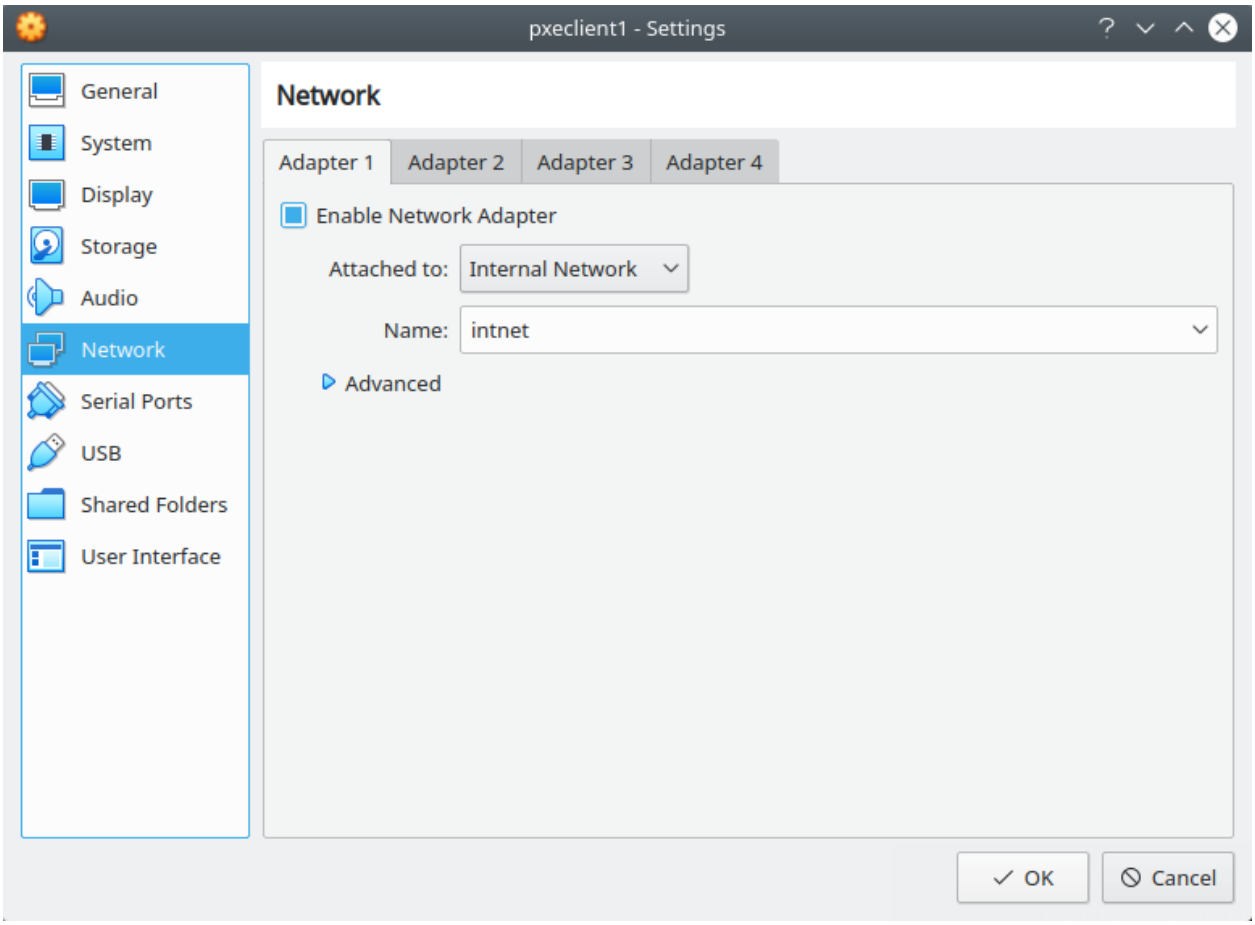


Fig. 2: VirtualBox - PXE Client - Network Tab

El arranque del instalador de CentOS 7 usando PXE se muestra en el siguiente video:

Los siguientes pasos consisten en seguir el proceso del instalador de CentOS 7. Hemos logrado el objetivo de proporcionar un medio de instalación por red con el método PXE. Mediante este, múltiples clientes PXE podrán conectarse al servidor PXE, a través de la red, y obtener los medios para instalar el sistema operativo CentOS 7.

A pesar de que hemos logrado brindar el instalador de CentOS 7 por red, sería útil automatizar la instalación del sistema operativo en sí. Este objetivo lo lograremos a través de archivos Kickstart.

19.2 PXE + Kickstart

Table of Contents

Documentación para habilitar el uso de un archivo Kickstart en un arranque e instalación de sistema operativo CentOS 7 por PXE. La función del archivo Kickstart será automatizar todo el proceso de instalación de CentOS 7 según los parámetros definidos, logrando un sistema operativo personalizado.

Esta documentación requiere haber realizado los pasos de la documentación previa para habilitar un servidor PXE: *PXE Install*

19.2.1 Creación del archivo Kickstart

El archivo Kickstart será pasado al cliente PXE través de FTP, por tanto debemos crear el archivo kickstart en el directorio público de FTP:

```
$ sudo vi /var/ftp/pub/centos7/centos7-ks.cfg
```

```
install
url --url="ftp://192.168.8.8/pub/centos7"
bootloader --location=mbr
keyboard --vckeymap=latam --xlayouts='latam','us'
rootpw --iscrypted $1$WUDGBrnr$Bq8p.jk4ikcEr2JYJRMwE0
lang en_US.UTF-8 --addsupport=es_US.UTF-8
clearpart --all --initlabel
part / --fstype="ext4" --grow --ondisk=sda --size=1
timezone America/Lima
text
reboot

%packages
%end
```

Podemos personalizar nuestro archivo Kickstart según nuestras preferencias. Podemos guiarnos de las siguientes plantillas: *Kickstart Templates 1*

Note: Verificar el nombre con el cual son creados los discos, para planear las particiones. En VirtualBox los discos se crean con el formato de nombre `sdx` (`--ondisk=sda`), pero en virt-manager los discos se crean con el formato de nombre `vdx` (`--ondisk=vda`). (x=a,b,c,d,...)

19.2.2 Edición del menú PXE boot

- Para clientes con sistemas basados en BIOS:

```
$ sudo vi /var/lib/tftpboot/pxelinux.cfg/default
```

Editar la línea de append, agregando la ruta de kickstart:

```
default menu.c32
prompt 0
timeout 30
menu title PXE Boot CentOS 7
label Instalar CentOS 7 (Minimal)
kernel /networkboot/centos7/vmlinuz
append initrd=/networkboot/centos7/initrd.img inst.repo=ftp://192.168.8.8/pub/centos7␣
↪ks=ftp://192.168.8.8/pub/centos7/centos7-ks.cfg
```

- Para clientes con sistemas basados en UEFI:

```
$ sudo vi /var/lib/tftpboot/grub.cfg
```

```
set timeout=60

menuentry 'Instalar CentOS 7 (Minimal)' {
        linuxefi /networkboot/centos7/vmlinuz inst.repo=ftp://192.168.8.8/pub/centos7/
↪ inst.ks=ftp://192.168.8.8/pub/centos7/centos7-ks.cfg
        initrdefi /networkboot/centos7/initrd.img
}
```

19.2.3 Reiniciar los servicios

Finalmente, reiniciamos los servicios de DHCP, TFTP y FTP:

```
$ ./restart_pxe_services.sh
```

19.2.4 Instalación de CentOS 7 usando PXE + Kickstart

El último paso será arrancar una VM como PXE client, tal como se realizó en la documentación previa: *Instalando CentOS 7 en un cliente PXE*. La diferencia es que ahora pasaremos un archivo Kickstart para automatizar la instalación del sistema operativo CentOS 7:

Note: En el archivo Kickstart hemos usado el parámetro `text`, con el fin de que la instalación no se realice en forma gráfica sino mostrando solo texto.

CHAPTER 20

Python

CHAPTER 21

RaspberryPi

CHAPTER 22

NOOBS (New Out Of Box Software)

22.1 Instalando un Sistema Operativo con NOOBS

Links útiles:

- Raspberry Pi Documentation
- Installation: Installing an operating system on your Raspberry Pi
- NOOBS Documentation

Table of Contents

22.1.1 Introducción

La manera más sencilla de instalar un sistema operativo (SO) en el Raspberry Pi es a través de NOOBS.

NOOBS (New Out Of Box Software) es un administrador para la instalación de SOs en Raspberry Pi. Este software incluye los siguientes SOs:

- Raspbian
- LibreELEC
- OSMC
- Recalbox
- Lakka
- RISC OS
- Screenly OSE
- Windows 10 IoT Core
- TLXOS

Existen 2 versiones de NOOBS: la versión offline e instalación por red (NOOBS) y la versión únicamente de instalación por red (NOOBS Lite). La diferencia es que la versión offline tiene el SO Raspbian incluido. Instalar otro SO o instalar Raspbian con NOOBS Lite requiere que contemos con conexión a Internet.

22.1.2 Prerequisitos

- Una PC con Windows o Linux.
- Un Raspberry Pi con monitor, mouse, teclado y conexión a Internet (alámbrica o inalámbrica).
- Una tarjeta micro SD clase 4 o clase 10 vacía de 8GB o más de almacenamiento.

> **Danger:** Se formateará la tarjeta, por lo que se perderán todos los archivos que estén guardados en la memoria.

- Un adaptador o lector de tarjetas micro SD que permita conectar la tarjeta a la computadora.

22.1.3 Objetivos

- Aprender el funcionamiento básico de NOOBS.
- Instalar un sistema operativo en el Raspberry Pi usando NOOBS.

22.1.4 Obtener NOOBS

Se puede obtener NOOBS descargando el archivo correspondiente desde la página de descargas de Raspberry Pi. Podrás elegir la versión de NOOBS que prefieras y descargarla. O puedes dirigirte directamente a las versiones de NOOBS disponibles.

En este tutorial descargaré el archivo ZIP de la versión NOOBS Lite.

Necesitarás una tarjeta micro SD de 8 GB o más de almacenamiento para guardar los archivos de instalación. Esta se deberá formatear como FAT o FAT32. En mi caso usaré una tarjeta de 16GB.

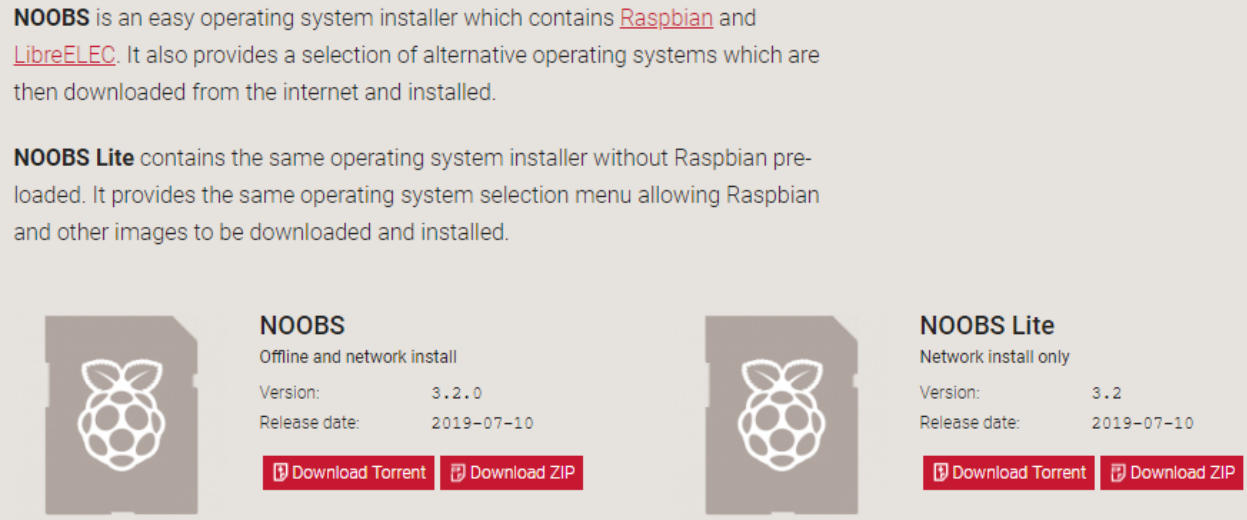


Fig. 1: Versiones de NOOBS

Para tarjetas micro SD de 64 GB o más de almacenamiento se deberá formatear exclusivamente como FAT32. Detalles: formatting an SDXC card for use with NOOBS.

22.1.5 Formateo de tarjeta Micro SD (en Windows)

Note: Procedimiento para sistemas operativos Windows

1. En Windows se puede usar el programa de formateo de la SD Association con el fin de formatear la tarjeta micro SD. Descarga e instala el programa **SD Card Formatter** (Link de descarga de SD Card Formatter).
2. Una vez instalado el programa SD Card Formatter, ingresa la tarjeta micro SD en la lectora de la PC (si tu PC no tiene lectora micro SD puedes usar un adaptador de micro SD a SD o a USB para que la PC lea la tarjeta).
3. Abre el programa SD Card Formatter y selecciona la unidad correspondiente a la tarjeta e ingrésale un nombre que la identifique en la sección *Volume label*. Dale clic en *Format* y espera que el proceso de formateo acabe.
4. Confirma el mensaje de advertencia. Clic en *Sí*.

Danger: TODA TU INFORMACIÓN SERÁ BORRADA DE LA MEMORIA MICRO SD

5. Aparecerá un resumen con las nuevas propiedades de la tarjeta. Clic en *Aceptar*.

22.1.6 Formateo de tarjeta Micro SD (en Linux)

Note: Procedimiento para sistemas operativos Linux

Basado en: NOOBS For Raspberry Pi (Rants and Raves)

1. En Linux, el primer paso será reconocer cuál es tu dispositivo micro SD en la PC. Corre el siguiente comando en un terminal (todavía no insertes tu memoria micro SD):

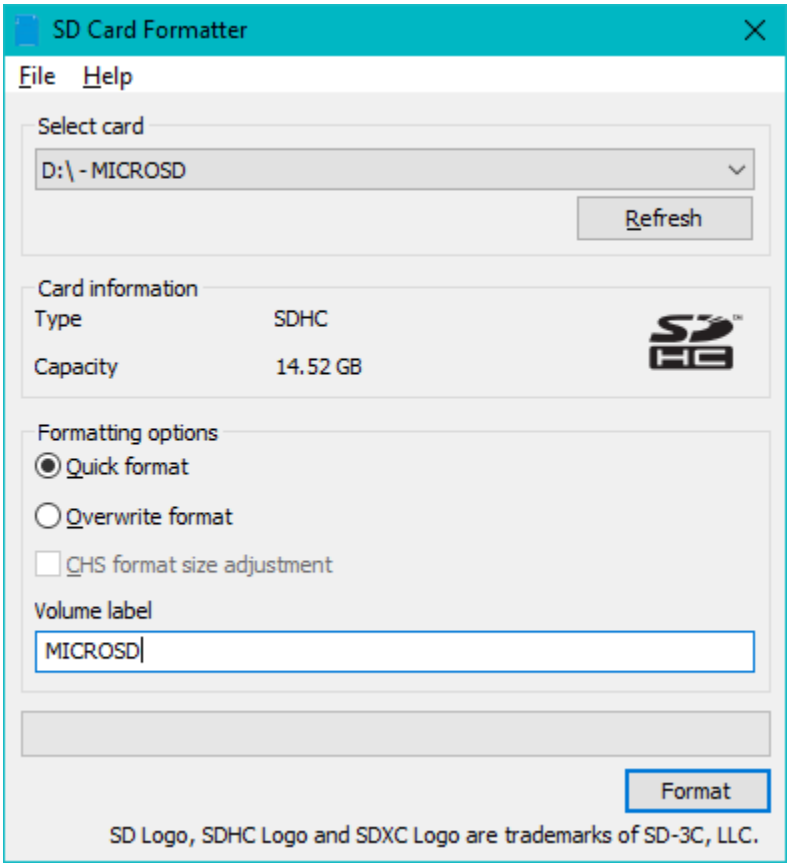


Fig. 2: SD Card Formatter - Formateo de la tarjeta SD - Paso 1

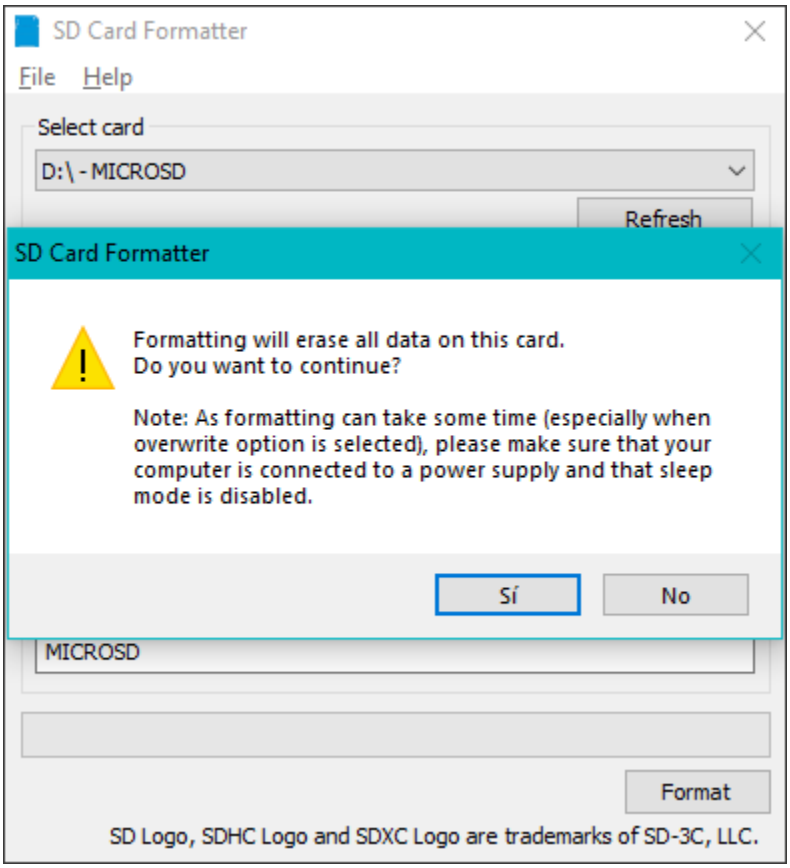


Fig. 3: SD Card Formatter - Formateo de la tarjeta SD - Paso 2

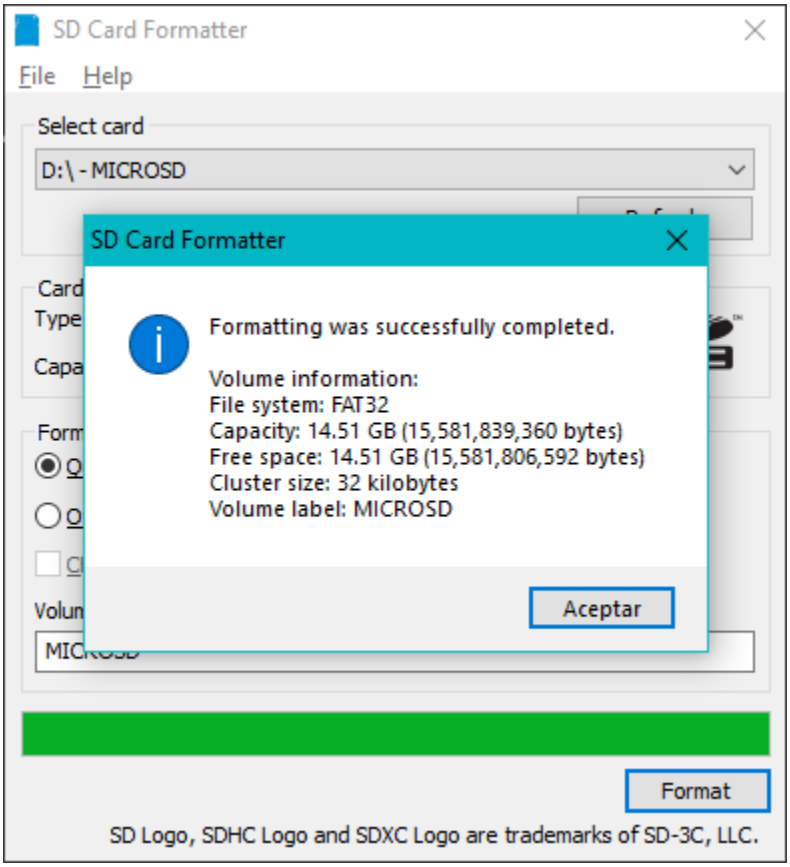


Fig. 4: SD Card Formatter - Formateo de la tarjeta SD - Paso 3

```
$ sudo fdisk -l
```

Se listarán todos los dispositivos montados y no montados en la computadora. Luego insertar la memoria micro SD a la PC y ejecuta el mismo comando de nuevo. Compara los dos listados y ve qué dispositivo nuevo ha sido agregado. En mi caso es el siguiente:

```
Disk /dev/sdb: 14,5 GiB, 15523119104 bytes, 30318592 sectors
Units: sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes
Disklabel type: dos
Disk identifier: 0xe72674a9

Device     Boot Start      End  Sectors  Size Id Type
/dev/sdb1        2048 30316543 30314496 14,5G  c W95 FAT32 (LBA)
```

2. Ahora que conocemos qué dispositivo es nuestra micro SD debemos borrar todas las particiones y archivos existentes dentro de ella. De esta forma, luego podremos crear la única partición que necesitamos. Para esto, usaremos la herramienta fdisk de esta forma:

```
$ sudo fdisk /dev/{nombre_del_dispositivo}
```

Cambia `{nombre_del_dispositivo}` según tu caso. En mi caso, ejecutaré el comando: `sudo fdisk /dev/sdb`

```
$ sudo fdisk /dev/sdb

Welcome to fdisk (util-linux 2.31.1).
Changes will remain in memory only, until you decide to write them.
Be careful before using the write command.


Command (m for help):
```

3. Primero lista las particiones de tu tarjeta micro SD usando la opción *p*:

```
Command (m for help): p
Disk /dev/sdb: 14,5 GiB, 15523119104 bytes, 30318592 sectors
Units: sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes
Disklabel type: dos
Disk identifier: 0xe72674a9

Device     Boot Start      End  Sectors  Size Id Type
/dev/sdb1        2048 30316543 30314496 14,5G  c W95 FAT32 (LBA)

Command (m for help):
```

4. Ahora, asegúrate de borrar cada partición que tenga tu tarjeta micro SD usando la opción *d*. Luego comprueba que ya no exista usando la opción *p* nuevamente:

Danger: TODA TU INFORMACIÓN SERÁ BORRADA DE LA MEMORIA MICRO SD

```
Command (m for help): d
Selected partition 1
Partition 1 has been deleted.

Command (m for help): p
Disk /dev/sdb: 14,5 GiB, 15523119104 bytes, 30318592 sectors
Units: sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes
Disklabel type: dos
Disk identifier: 0xe72674a9

Command (m for help):
```

5. Seguido a esto crearemos una nueva partición usando la opción *n*. Presionar *Enter* a todas las opciones que aparezcan para que tomen los valores por defecto:

```
Command (m for help): n
Partition type
    p   primary (0 primary, 0 extended, 4 free)
    e   extended (container for logical partitions)
Select (default p): p
Partition number (1-4, default 1): 1
First sector (2048-30318591, default 2048):
Last sector, +sectors or +size{K,M,G,T,P} (2048-30318591, default 30318591):

Created a new partition 1 of type 'Linux' and of size 14,5 GiB.
Partition #1 contains a vfat signature.

Do you want to remove the signature? [Y]es/[N]o: y

The signature will be removed by a write command.
```

Obtendremos una nueva partición para nuestra tarjeta micro SD. Vuelve a comprobar con la opción *p*:

```
Command (m for help): p
Disk /dev/sdb: 14,5 GiB, 15523119104 bytes, 30318592 sectors
Units: sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes
Disklabel type: dos
Disk identifier: 0xe72674a9

Device     Boot Start      End  Sectors  Size Id Type
/dev/sdb1        2048 30318591 30316544 14,5G 83 Linux

Filesystem/RAID signature on partition 1 will be wiped.
```

6. Ahora cambiaremos el tipo de partición de Linux a FAT32. Para esto usaremos la opción *t*:

Si quieres listar todos los formatos disponibles, usa la opción *L* primero:

```
Command (m for help): t
Selected partition 1
Hex code (type L to list all codes): L

0  Empty           24  NEC DOS         81  Minix / old Lin bf  Solaris
1  FAT12           27  Hidden NTFS Win 82  Linux swap / So c1  DRDOS/sec (FAT-
```

(continues on next page)

(continued from previous page)

```
2  XENIX root      39  Plan 9          83  Linux           c4  DRDOS/sec (FAT-
3  XENIX usr       3c  PartitionMagic  84  OS/2 hidden or  c6  DRDOS/sec (FAT-
4  FAT16 <32M      40  Venix 80286     85  Linux extended  c7  Syrinx
5  Extended        41  PPC PReP Boot   86  NTFS volume set da  Non-FS data
6  FAT16           42  SFS             87  NTFS volume set db  CP/M / CTOS / .
7  HPFS/NTFS/exFAT 4d  QNX4.x          88  Linux plaintext de  Dell Utility
8  AIX             4e  QNX4.x 2nd part 8e  Linux LVM       df  BootIt
9  AIX bootable    4f  QNX4.x 3rd part 93  Amoeba          e1  DOS access
a  OS/2 Boot Manag 50  OnTrack DM      94  Amoeba BBT      e3  DOS R/O
b  W95 FAT32       51  OnTrack DM6 Aux 9f  BSD/OS          e4  SpeedStor
c  W95 FAT32 (LBA) 52  CP/M            a0  IBM Thinkpad hi ea  Rufus alignment
e  W95 FAT16 (LBA) 53  OnTrack DM6 Aux a5  FreeBSD         eb  BeOS fs
f  W95 Ext'd (LBA) 54  OnTrackDM6      a6  OpenBSD         ee  GPT
10  OPUS            55  EZ-Drive        a7  NeXTSTEP        ef  EFI (FAT-12/16/
11  Hidden FAT12    56  Golden Bow      a8  Darwin UFS      f0  Linux/PA-RISC b
12  Compaq diagnost 5c  Priam Edisk     a9  NetBSD          f1  SpeedStor
14  Hidden FAT16 <3 61  SpeedStor       ab  Darwin boot     f4  SpeedStor
16  Hidden FAT16    63  GNU HURD or Sys af  HFS / HFS+      f2  DOS secondary
17  Hidden HPFS/NTF 64  Novell Netware  b7  BSDI fs         fb  VMware VMFS
18  AST SmartSleep  65  Novell Netware  b8  BSDI swap       fc  VMware VMKCORE
1b  Hidden W95 FAT3 70  DiskSecure Mult bb  Boot Wizard hid fd  Linux raid auto
1c  Hidden W95 FAT3 75  PC/IX           bc  Acronis FAT32 L fe  LANstep
1e  Hidden W95 FAT1 80  Old Minix       be  Solaris boot    ff  BBT
```

Si no deseas listar las opciones de formato, usa directamente la opción *b* para transformar la partición a formato FAT32:

```
Command (m for help): t
Selected partition 1
Hex code (type L to list all codes): b
Changed type of partition 'Linux' to 'W95 FAT32'.
```

De nuevo, comprueba que todo esté correcto con la opción *p*:

```
Command (m for help): p
Disk /dev/sdb: 14,5 GiB, 15523119104 bytes, 30318592 sectors
Units: sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes
Disklabel type: dos
Disk identifier: 0xe72674a9

Device     Boot Start      End  Sectors  Size Id Type
/dev/sdb1        2048 30318591 30316544 14,5G  b W95 FAT32

Filesystem/RAID signature on partition 1 will be wiped.
```

7. Para acabar con la configuración de la partición y escribirla en la tarjeta micro SD usar la opción *w*:

```
Command (m for help): w
The partition table has been altered.
Calling ioctl() to re-read partition table.
Syncing disks.
```

8. Ahora que tenemos la partición creada debemos formatearla, ya que, únicamente fdisk sabe que es del tipo FAT32. Para formatear la partición usar:

Note: No usar el nombre del dispositivo sino de la partición

```
$ sudo mkfs.vfat /dev/{nombre_de_la_partición}

$ sudo mkfs.vfat /dev/sdb1
mkfs.fat 4.1 (2017-01-24)
```

22.1.7 Instalar NOOBS (Windows)

Extraer el archivo `.zip` y copiar todos los archivos a la tarjeta micro SD formateada. ¡! Copia los archivos directamente en el directorio raíz de la tarjeta (por ejemplo `D:/`).

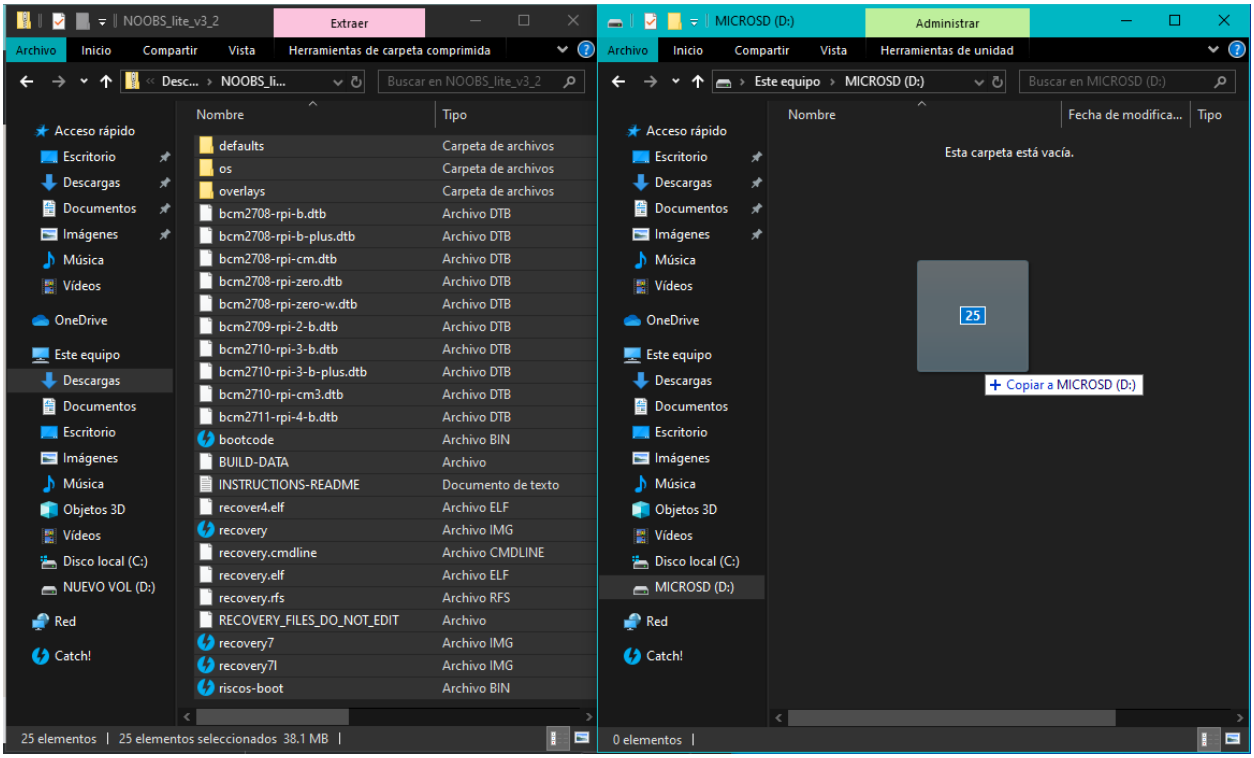


Fig. 5: Copiar contenidos de NOOBS a la tarjeta micro SD

22.1.8 Instalar NOOBS (Linux, con interfaz gráfica)

En primer lugar, saca la tarjeta micro SD de la PC y vuelve a insertarla para montarla en el sistema. Seguramente te aparezca un aviso para abrir el dispositivo con el navegador de archivos; de ser así, hazlo. Descomprime el archivo `.zip` de NOOBS descargado y copia todos los archivos directamente a la tarjeta micro SD.

22.1.9 Instalar NOOBS (Linux, con línea de comandos)

1. En primer lugar, saca la tarjeta micro SD de la PC y vuelve a insertarla para montarla en el sistema. Para comprobar donde está montada la partición usar:

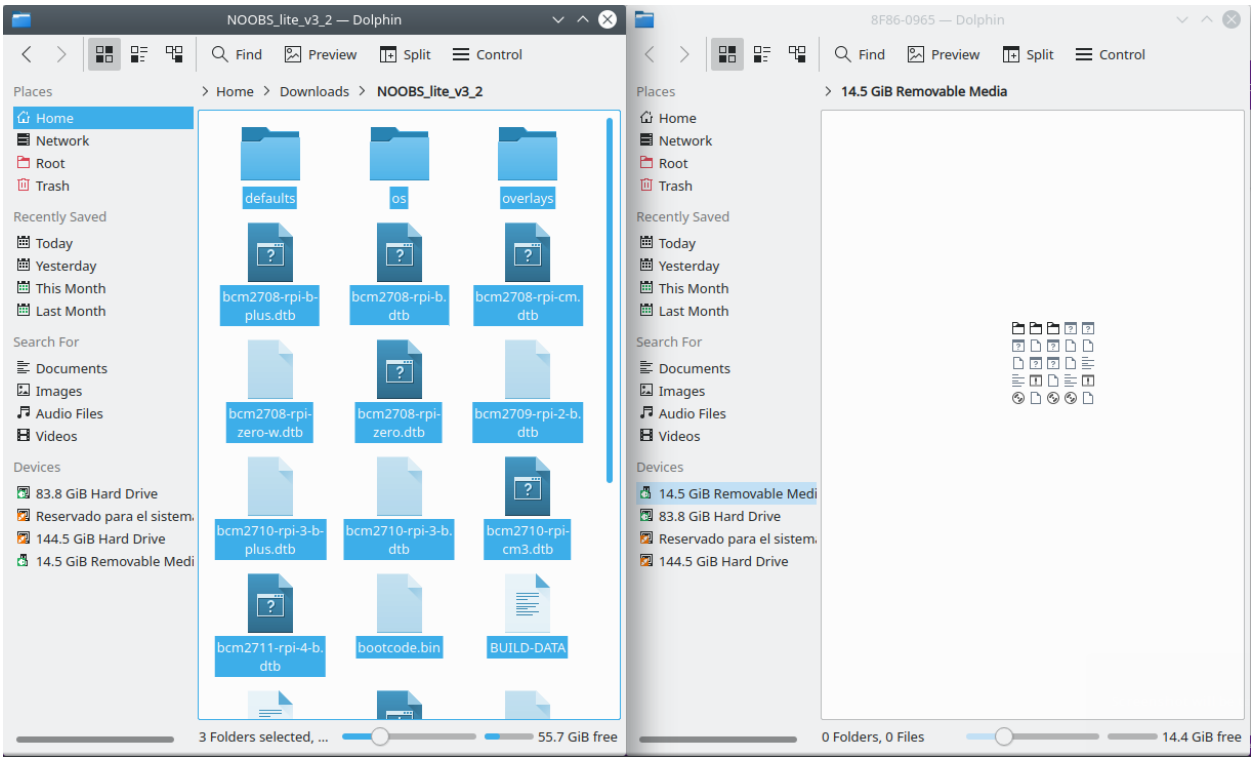


Fig. 6: Copiar contenidos de NOOBS a la tarjeta micro SD

```
$ mount | grep -i /dev/{nombre_de_la_partición}
$ mount | grep -i /dev/sdb1
```

2. Cambia a la ruta donde está montado tu partición y descomprime el archivo .zip de NOOBS descargado:

```
$ cd /media/{ruta_de_la_partición_montada}
$ unzip ~/Downloads/NOOBS_lite_v3_2.zip
```

3. Por último, desmonta la tarjeta micro SD con:

```
$ sudo umount /dev/{nombre_de_la_partición}
```

22.1.10 Instalar un SO usando NOOBS

1. Expulsa la tarjeta micro SD de la PC. Luego inserta la tarjeta micro SD al Raspberry Pi y encenderlo. En el primer arranque, se creará un partición llamada “RECOVERY”. En ella se guardarán las distribuciones y archivos para el proceso de recuperación.
2. Una vez dentro de NOOBS, si no cuentas con una conexión cableada Ethernet a la que puedas conectar tu Raspberry, conéctate usando una red WiFi disponible.

Una vez cuentes con conexión a Internet puede seguir con el tutorial.

3. En la pantalla se verá todos los sistemas operativos que pueden ser instalados en la tarjeta micro SD. También puedes cambiar el idioma y teclado que se muestra. Selecciona el (o los) sistema(s) operativo(s) que desees instalar.

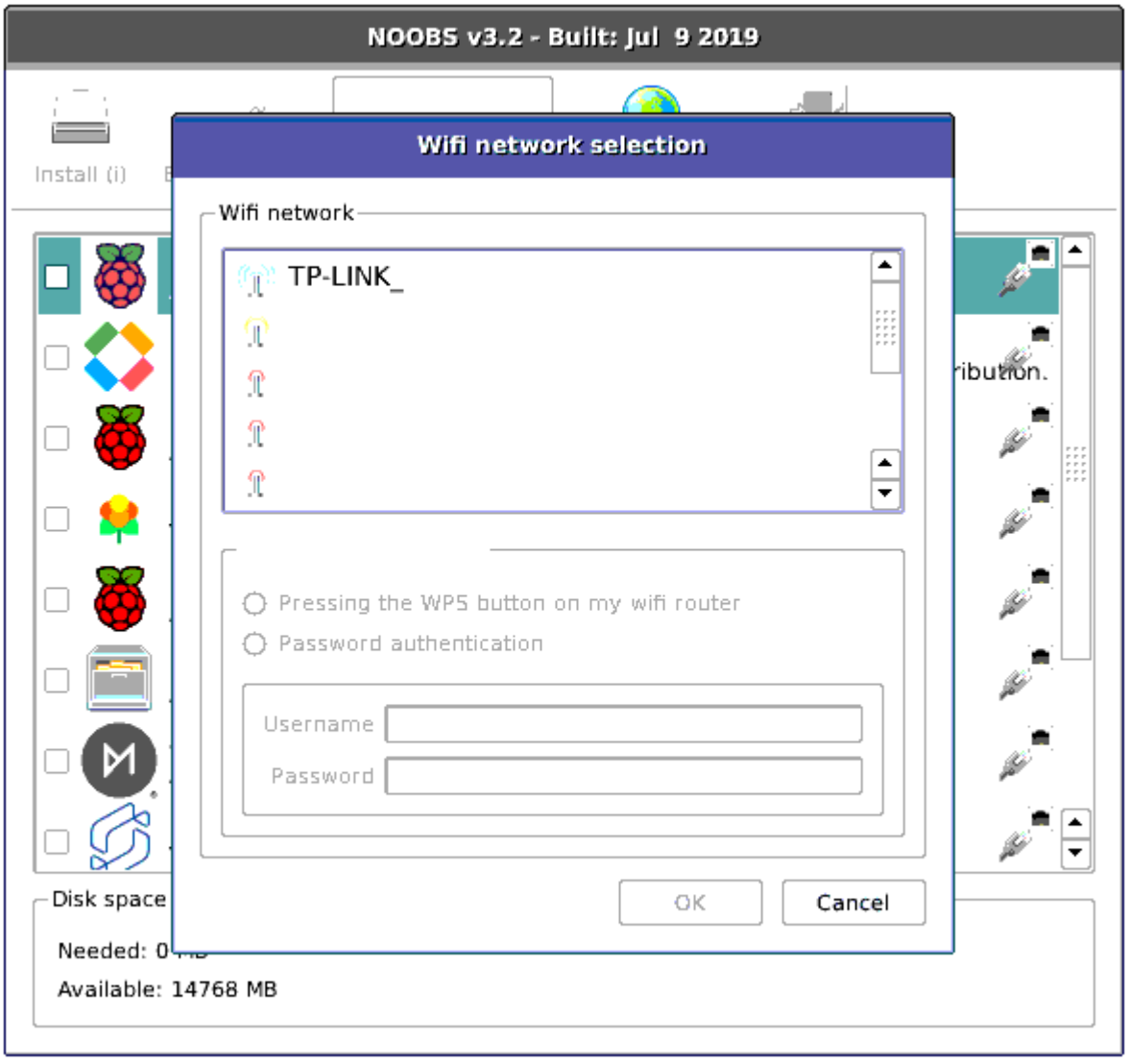


Fig. 7: Proceso de instalación de un SO con NOOBS

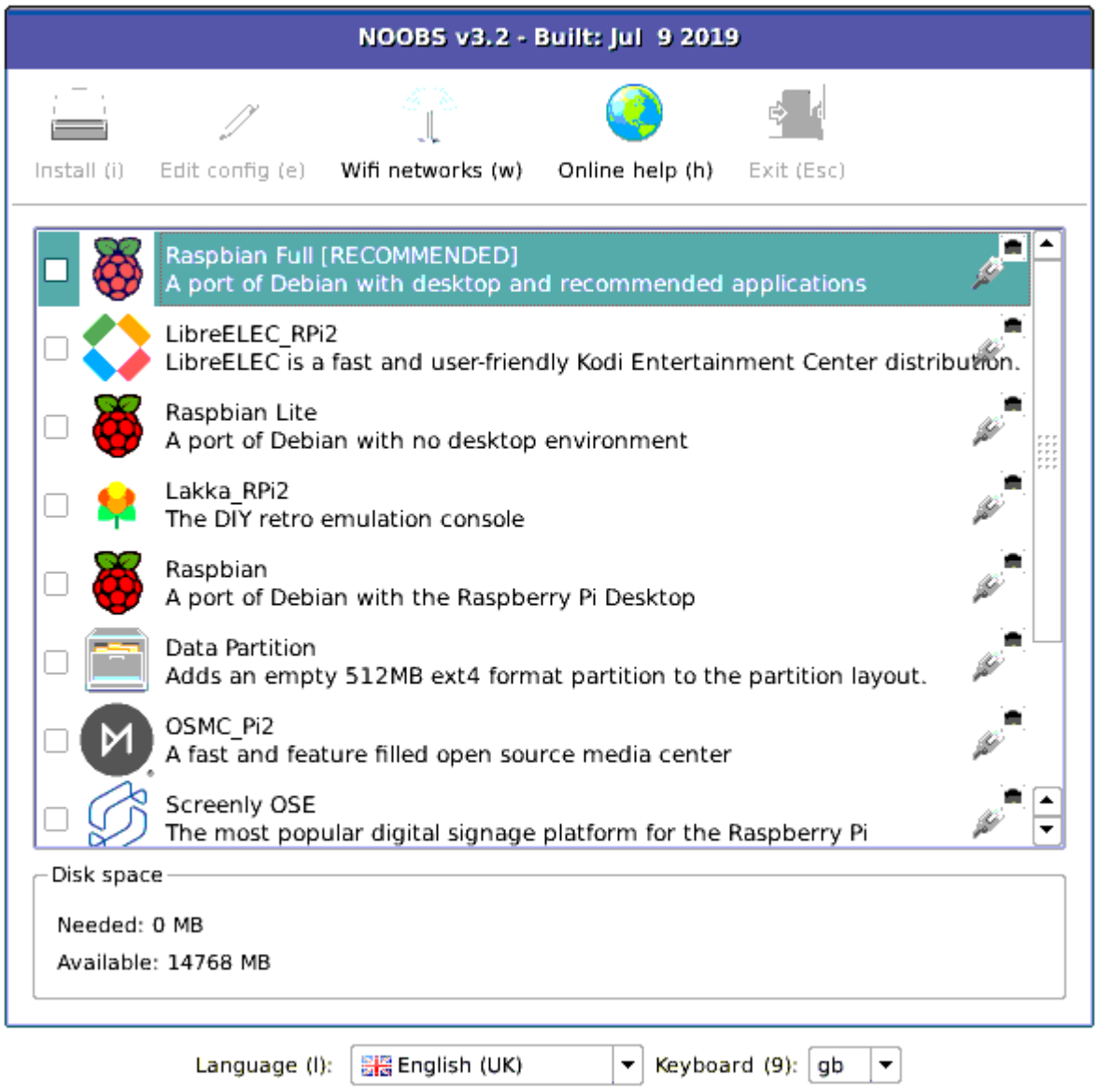


Fig. 8: Proceso de instalación de un SO con NOOBS

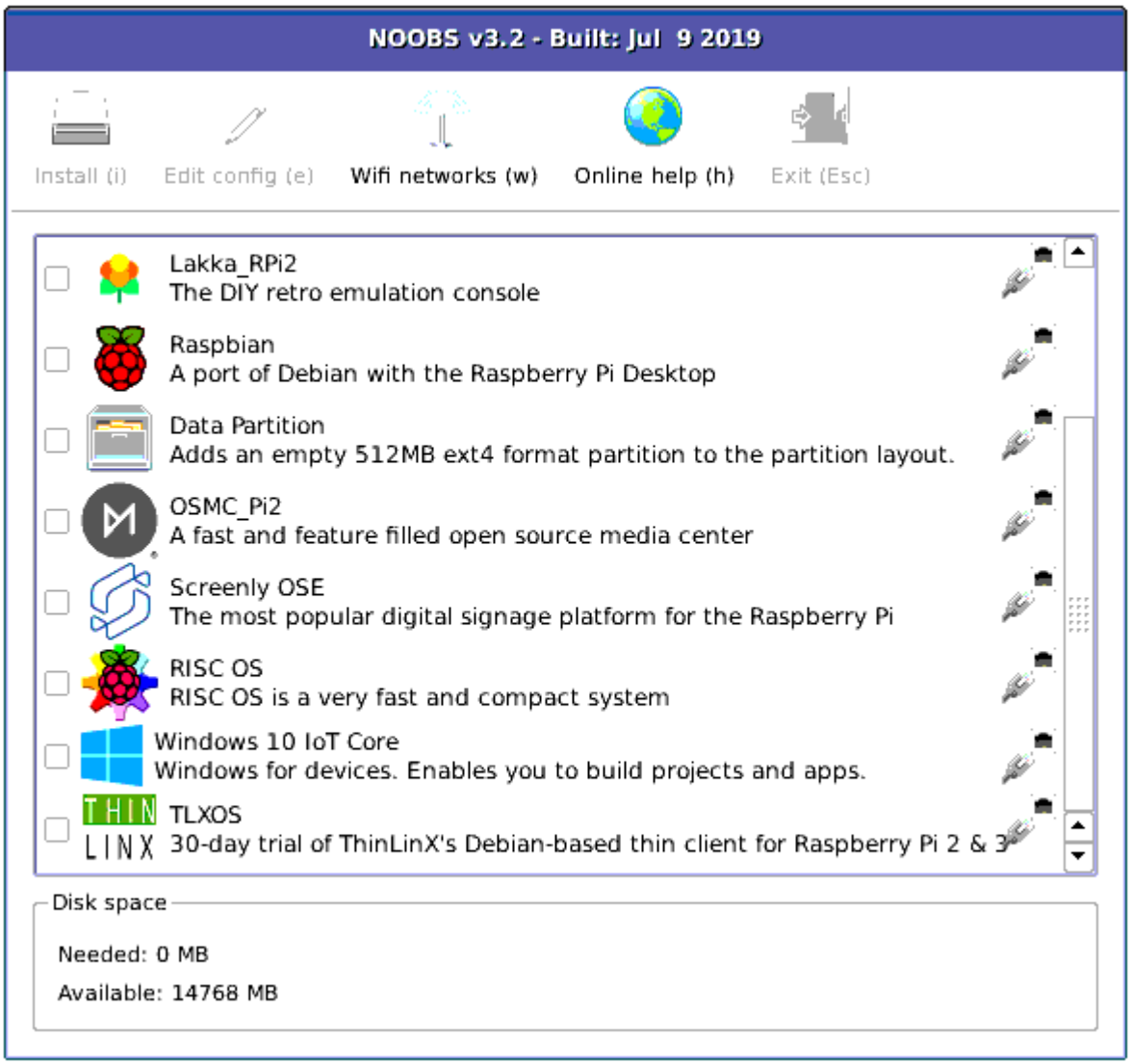


Fig. 9: Proceso de instalación de un SO con NOOBS

4. Una vez selecciones el sistema operativo que desees, que en mi caso es Raspbian Full, dale clic en el botón Install.

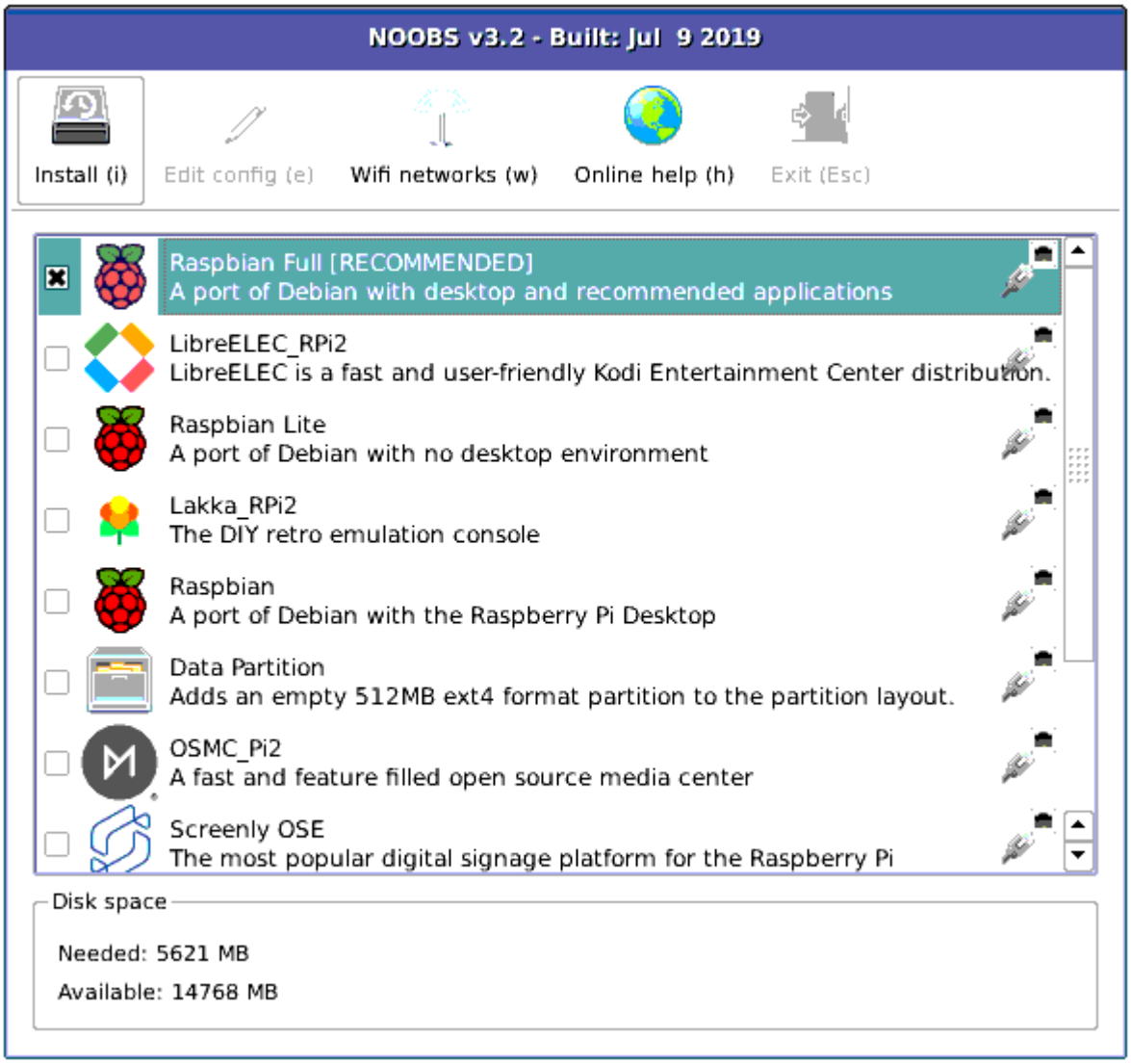


Fig. 10: Proceso de instalación de un SO con NOOBS

5. Acepta el mensaje de confirmación. Seguido a esto, comenzará la descarga del sistema operativo y su instalación.

Warning: Cuando la barra llegue al 100% se demorará unos minutos en acabar la instalación. Esperar pacientemente hasta que veamos el mensaje del siguiente paso.

6. Al acabar la instalación del sistema operativo acepta el mensaje de confirmación.
7. El Raspberry Pi reiniciará automáticamente y ahora arrancará con el sistema operativo que hemos instalado.

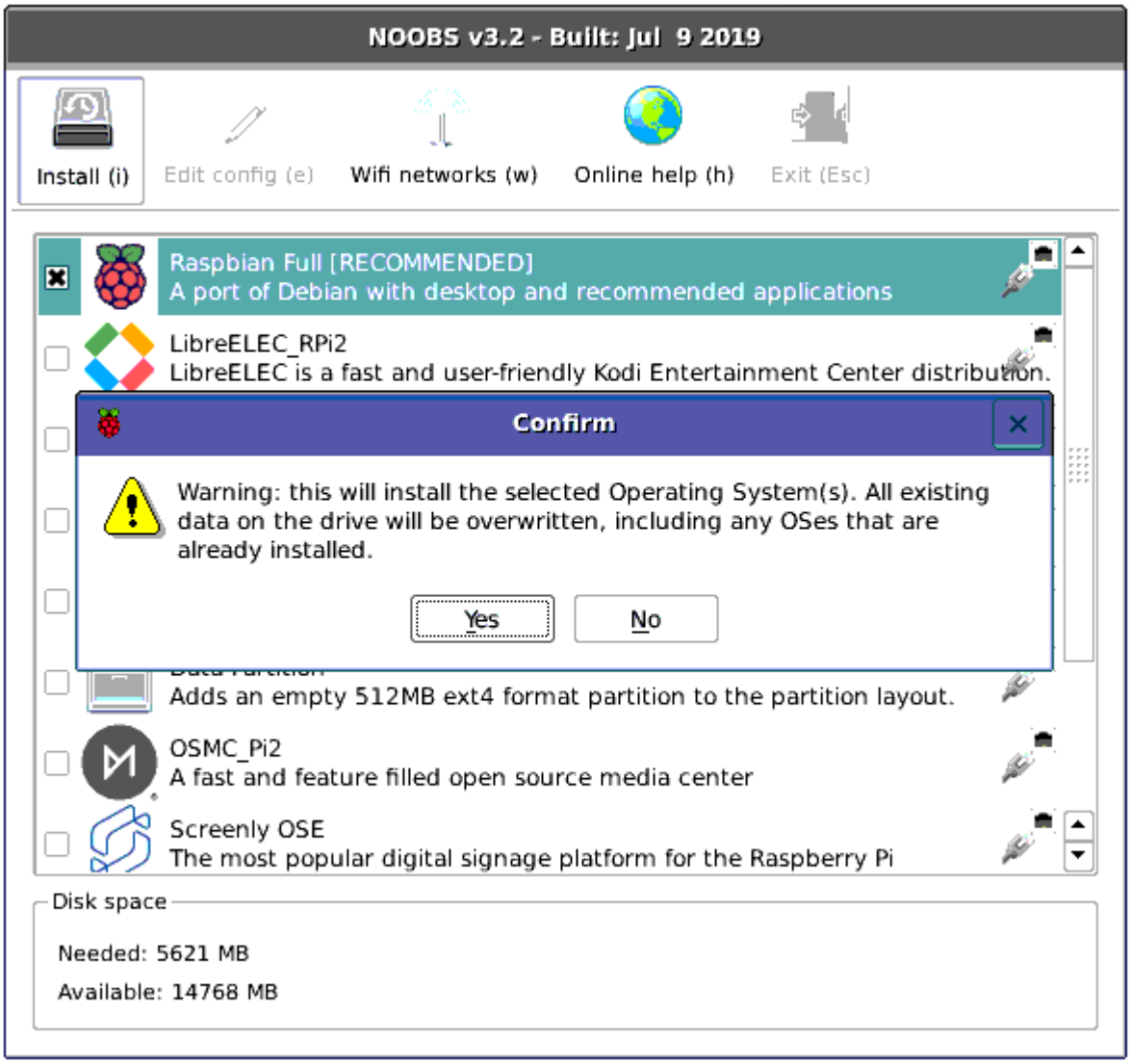


Fig. 11: Proceso de instalación de un SO con NOOBS

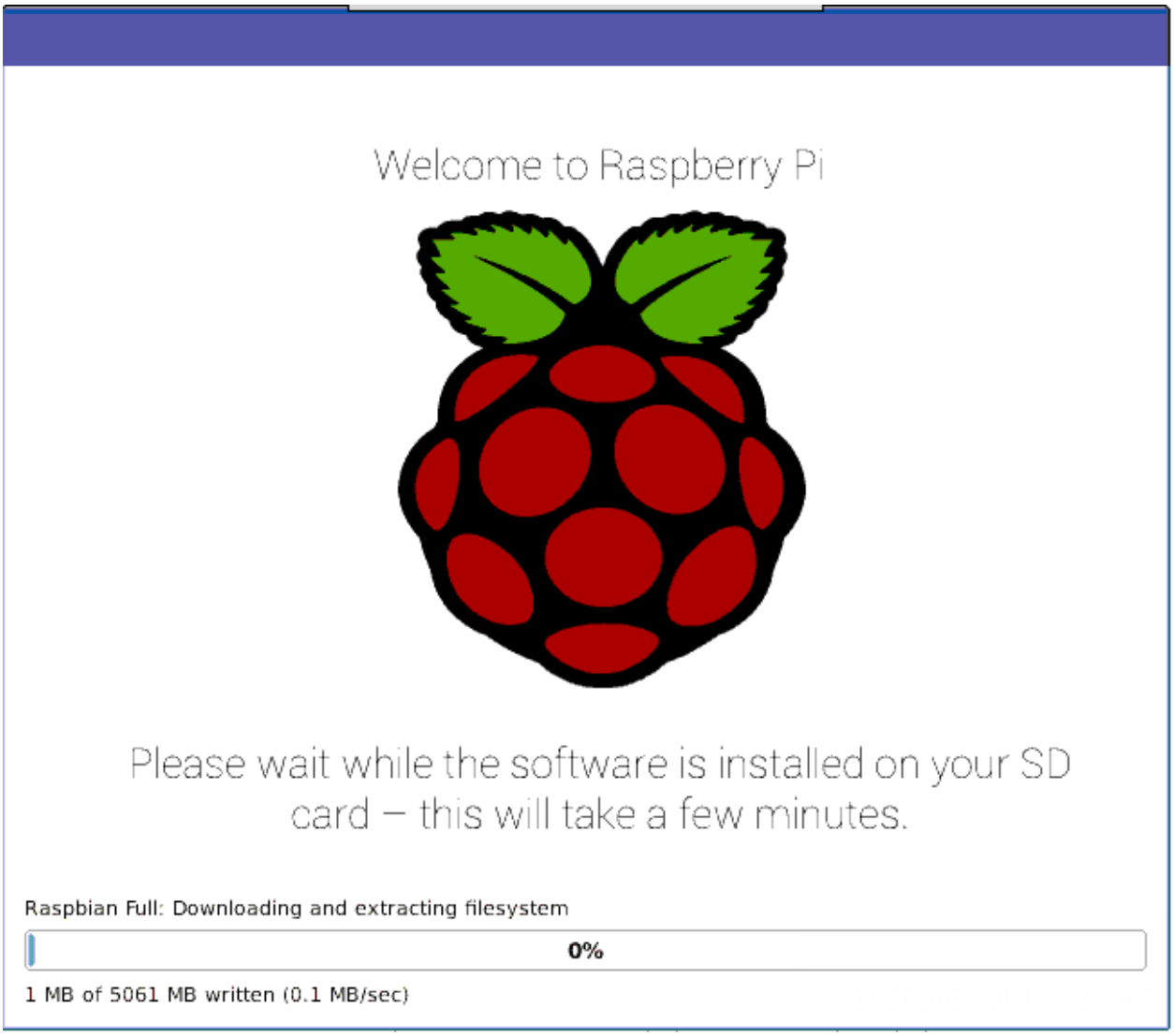


Fig. 12: Proceso de instalación de un SO con NOOBS

Fig. 13: Proceso de instalación de un SO con NOOBS

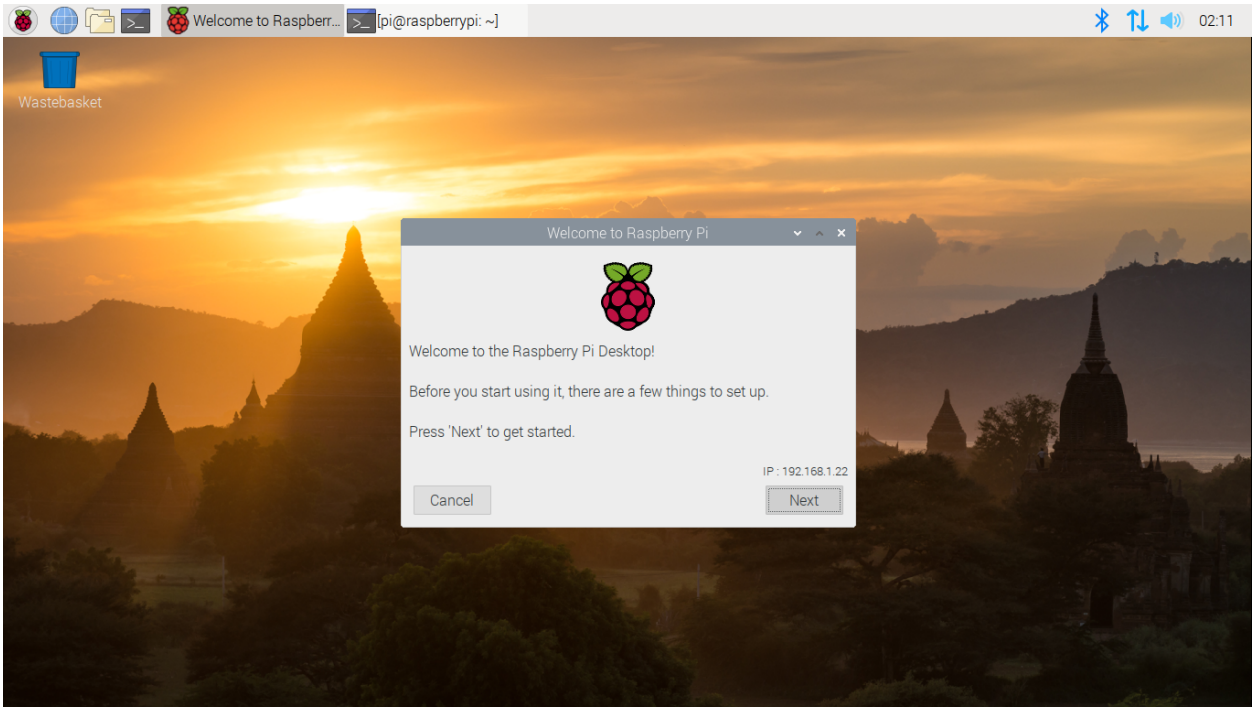


Fig. 14: Proceso de instalación de un SO con NOOBS

22.2 Información extra de NOOBS

Table of Contents

22.2.1 NOOBS's Command Line Interface

- Entrar a la interfaz de línea de comandos de NOOBS con: *Alt+FN+F2*
- Regresar a la interfaz gráfica de NOOBS con: *Alt+FN+F1*
- Credenciales de usuario (NOOBS): user:pass = `root:raspberry`
- Apagar NOOBS: `poweroff -f`

22.2.2 Tomar capturas de pantalla en NOOBS

Cuando se enciende el Raspberry Pi con un tarjeta SD que tiene instalado NOOBS, no se tiene ningún programa que permita tomar capturas de pantalla o usar otras funcionalidades. Para tomar capturas podemos usar VNC desde otra PC en la misma red que el Raspberry Pi y ver su pantalla.

Solución basada en un Foro de Raspberry Pi para activar VNC en NOOBS y una guía de GitHub para usar PINN con VNC.

1. Editar el archivo `recovery.cmdline` de los archivos copiados a la tarjeta micro SD para la instalación de NOOBS. Al final de la primera y única línea ingresar `vncinstall forcetrigger`.
   - `vncinstall` inhabilita la capacidad de ver una interfaz gráfica con la pantalla conectada al Rasberry Pi, solo permite verla a través de un usuario conectado por VNC al Raspberry Pi.
   - `forcetrigger` fuerza a usar VNC para la instalación; eliminarlo una vez esta acabe.
2. Insertar la tarjeta micro SD al Raspberry Pi y encenderlo (procurar que esté conectado mediante cable Ethernet a un servidor DHCP para que pueda recibir una IP). En el monitor solo saldrá:

```
ip: SIOCGIFFLAGS: No such device
Starting system message bus: done
```

3. Entrar a la línea de comandos de NOOBS (`Alt+FN+F2`) y verificar que tiene una ip (`ip address`). De no ser así, asignarle una IP dentro de la red.
4. Instalar VNC Viewer en una PC con Windows o Linux que esté dentro de la misma red que el Raspberry Pi. (Descargar VNC).
5. Conectarnos desde el programa VNC insertando la IP del Raspberry Pi:

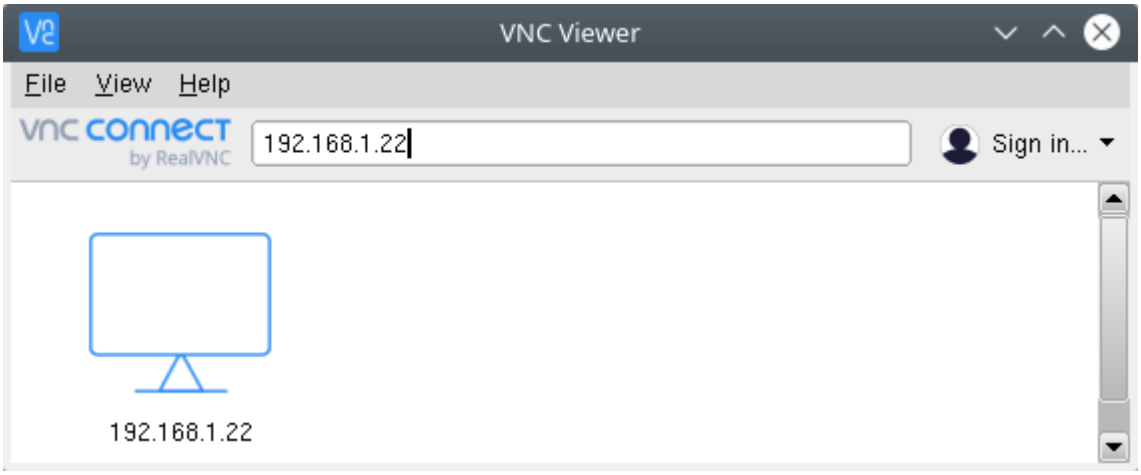


Fig. 15: Conexión remota al Raspberry Pi desde VNC

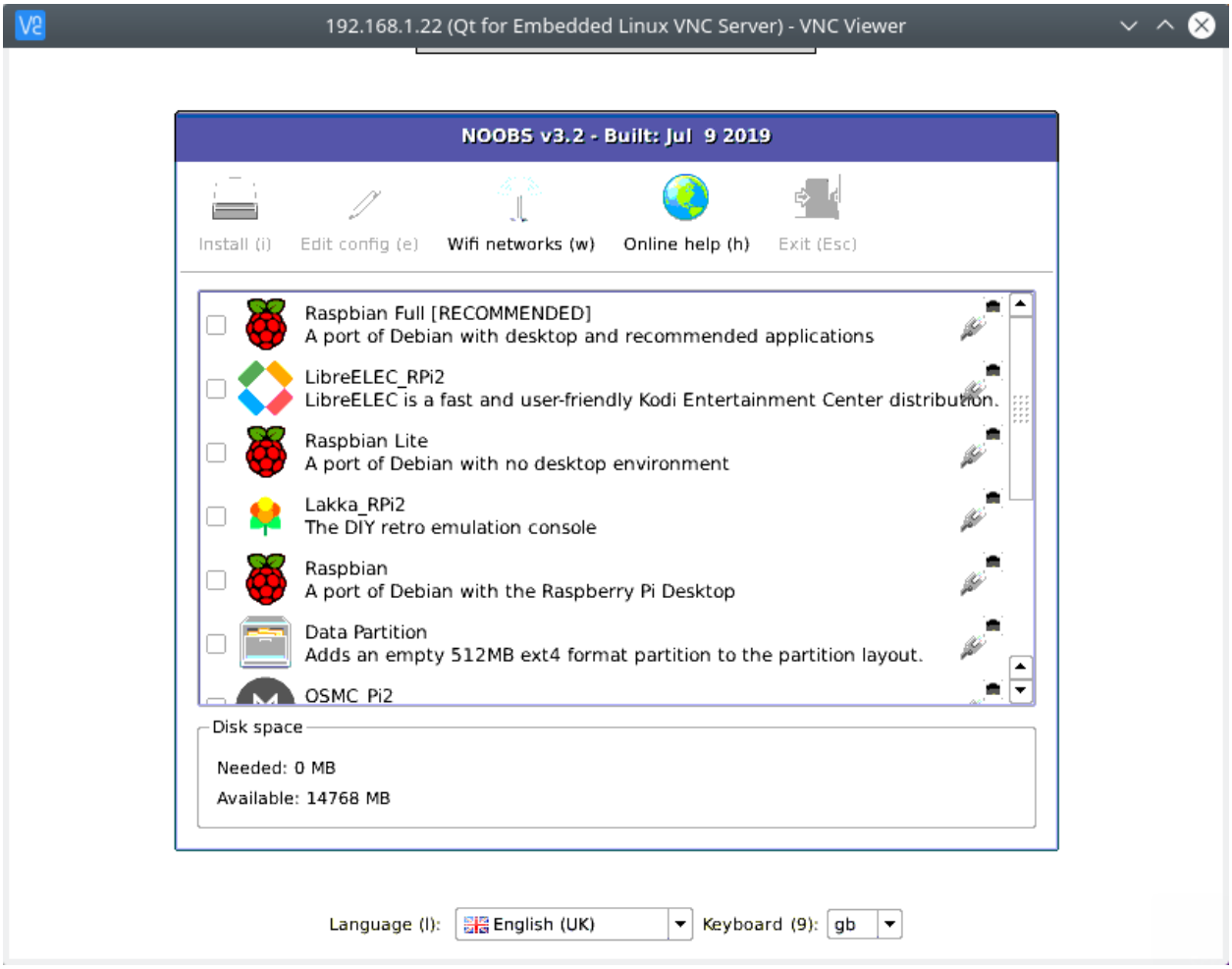


Fig. 16: Conexión remota al Raspberry Pi desde VNC

Luego de instalar el sistema operativo y reiniciar el Raspberry Pi debemos remover las opciones `vncinstall` y `forcetrigger`, ya que de no hacerlo volverá a aparecer la pantalla con el mensaje:

```
ip: SIOCGIFFLAGS: No such device
Starting system message bus: done
```

Insertar la tarjeta micro SD en la PC, abrir la partición RECOVERY y editar el archivo `recovery.cmdline`. Quitar las opciones `vncinstall` y `forcetrigger` de la línea.

Para volver al modo recovery cuando queramos se debe mantener presionada la tecla Shift cuando es Rasberry Pi está prendiendo. Podemos seleccionar y cambiar nuestro sistema operativo pero toda la data del usuario se perderá.

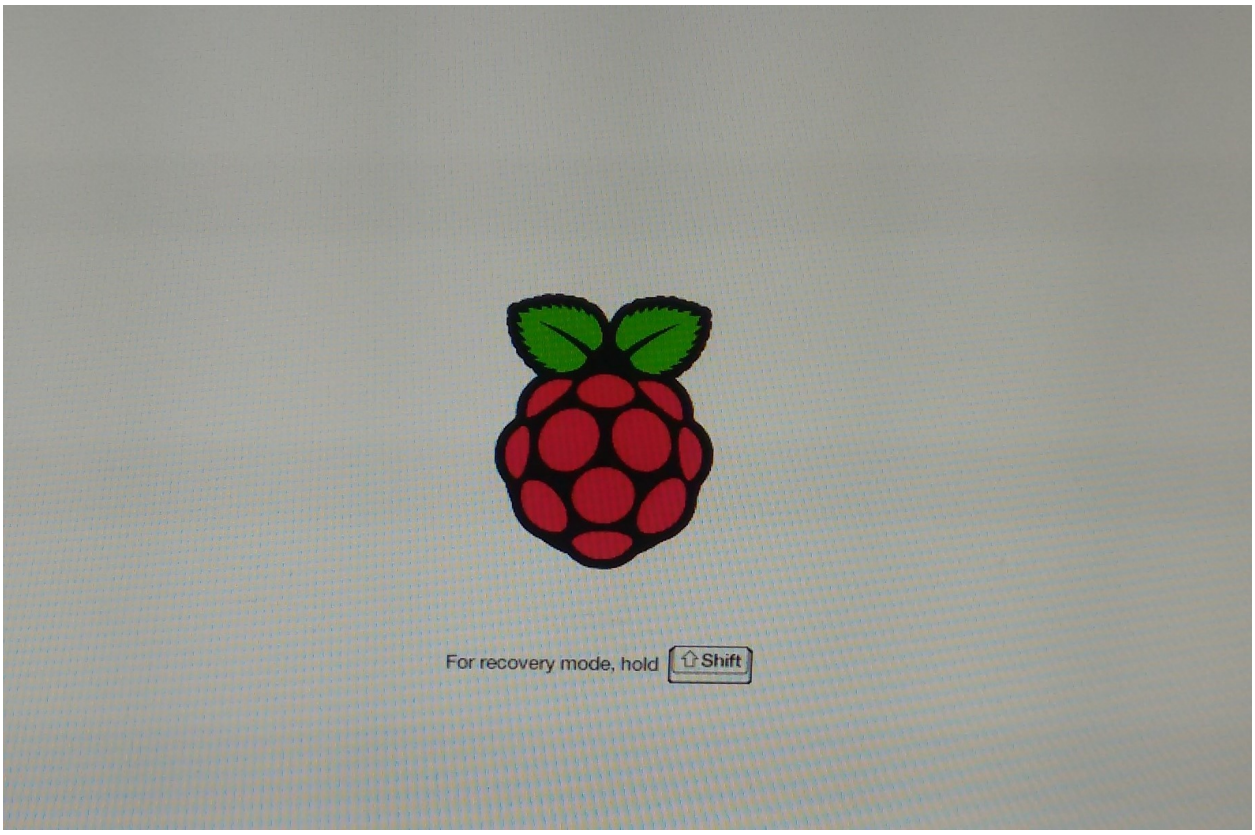


Fig. 17: Ingresar al modo recovery

22.2.3 Configuración inicial de Raspbian (post instalación)

22.2.4 Extra - Tarjeta micro SD está protegida contra escritura

ERROR:

- Cannot format write protected card
- El disco está protegido contra escritura

SOLUCIÓN:

1. Ver que el Lock del adaptador SD - micro SD esté desbloqueado.
2. Probar con varios adapatadores SD.

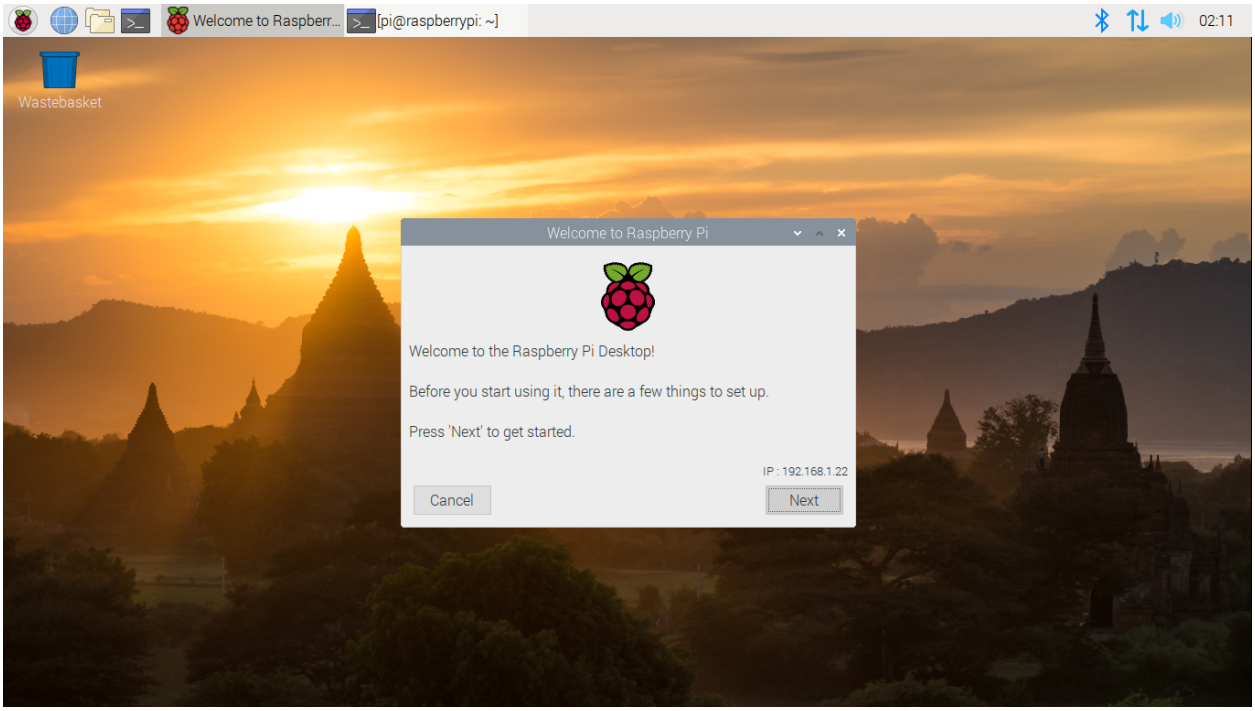


Fig. 18: Proceso de Configuración inicial de Raspbian

Fig. 19: Proceso de Configuración inicial de Raspbian

Fig. 20: Proceso de Configuración inicial de Raspbian

Fig. 21: Proceso de Configuración inicial de Raspbian

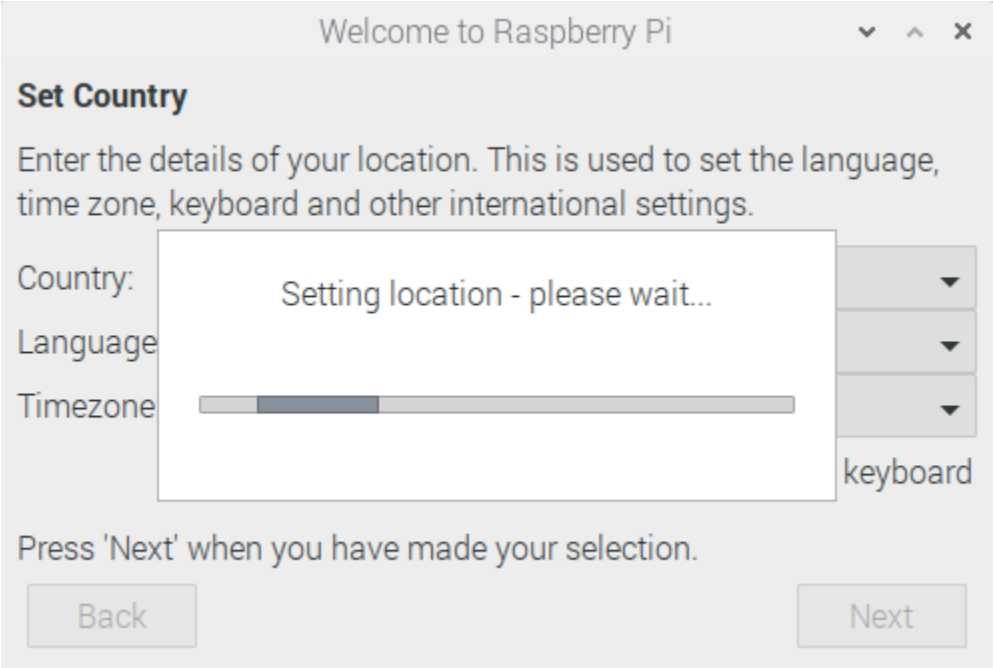


Fig. 22: Proceso de Configuración inicial de Raspbian

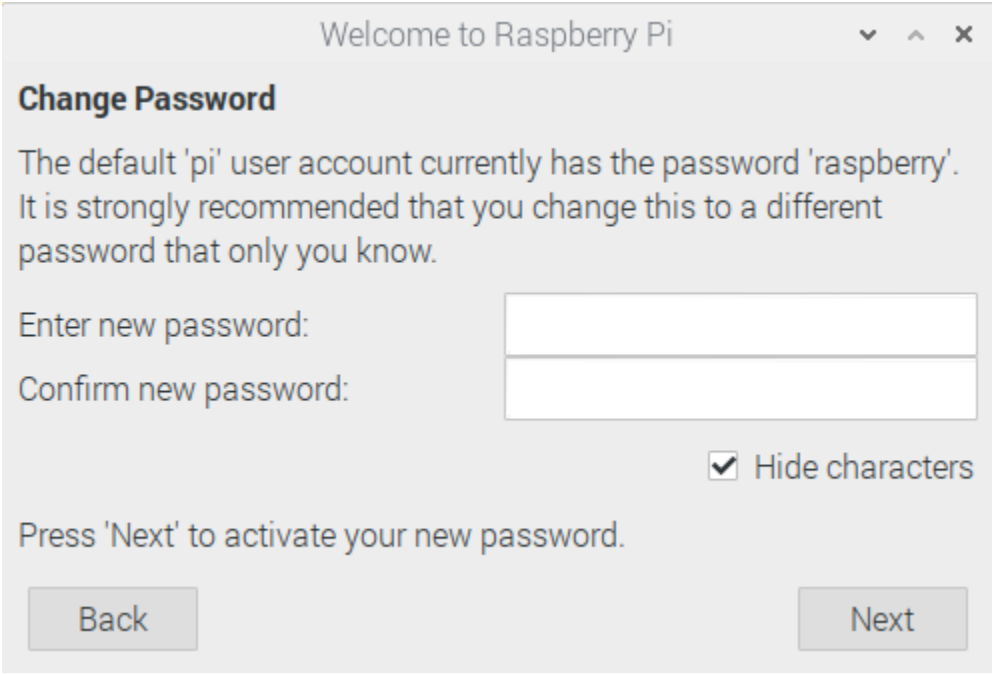


Fig. 23: Proceso de Configuración inicial de Raspbian

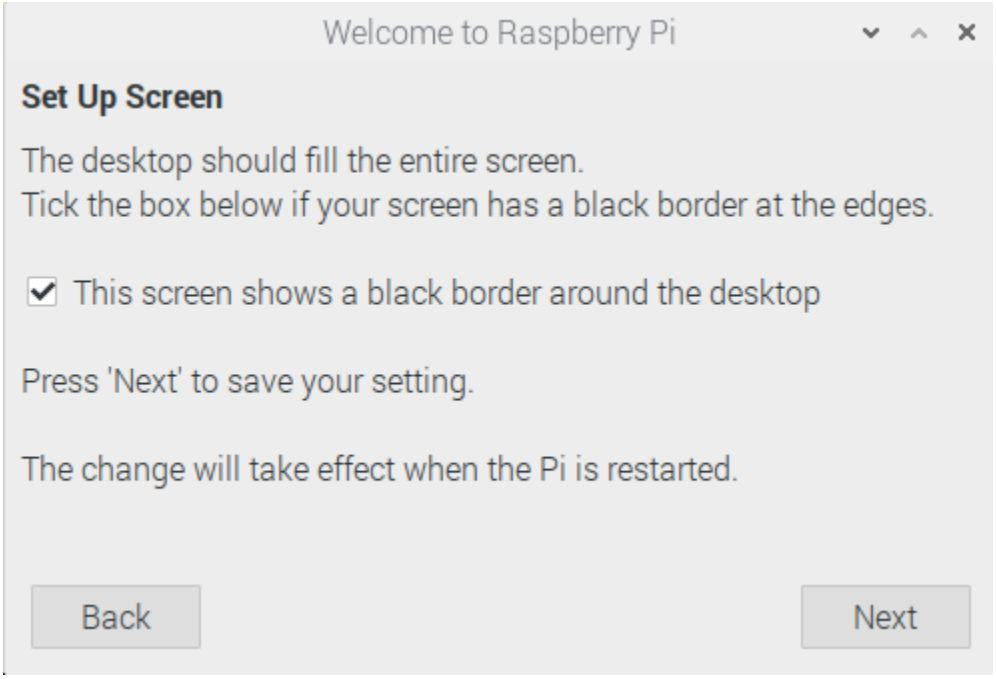


Fig. 24: Proceso de Configuración inicial de Raspbian

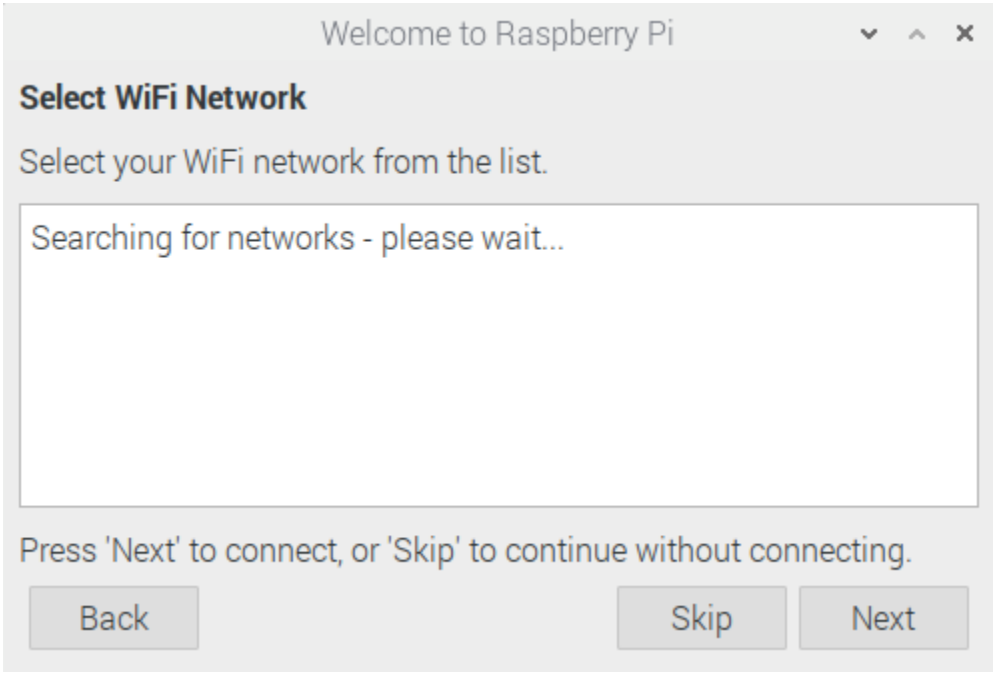


Fig. 25: Proceso de Configuración inicial de Raspbian

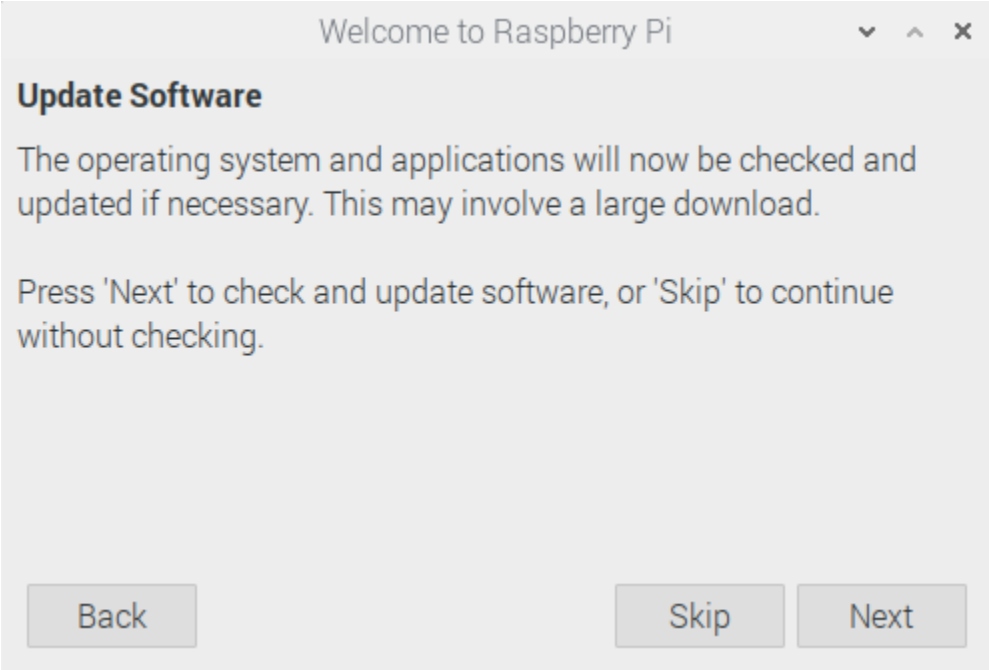


Fig. 26: Proceso de Configuración inicial de Raspbian

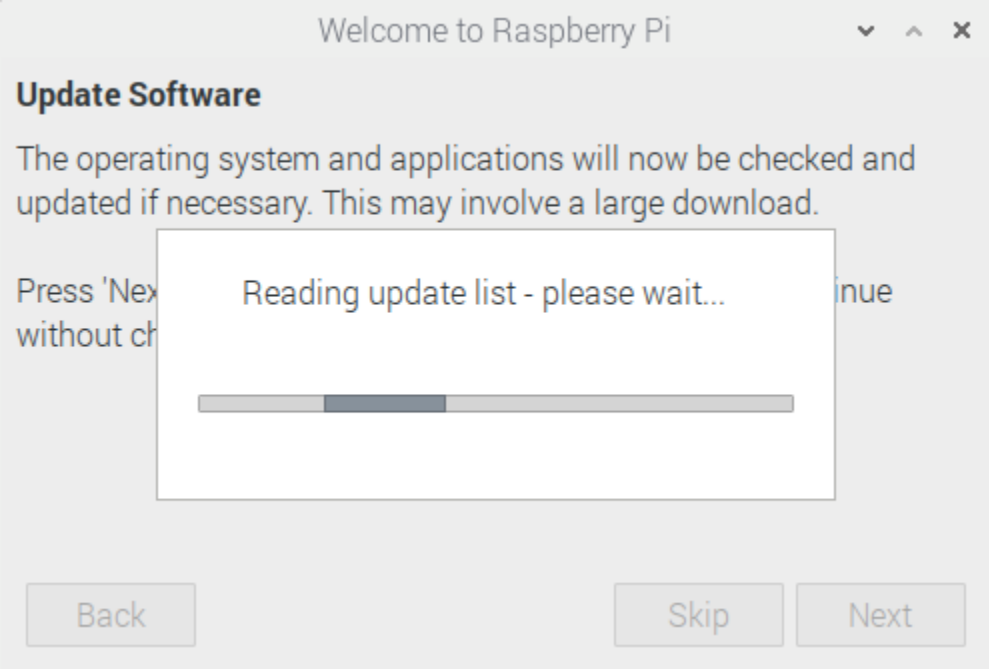


Fig. 27: Proceso de Configuración inicial de Raspbian

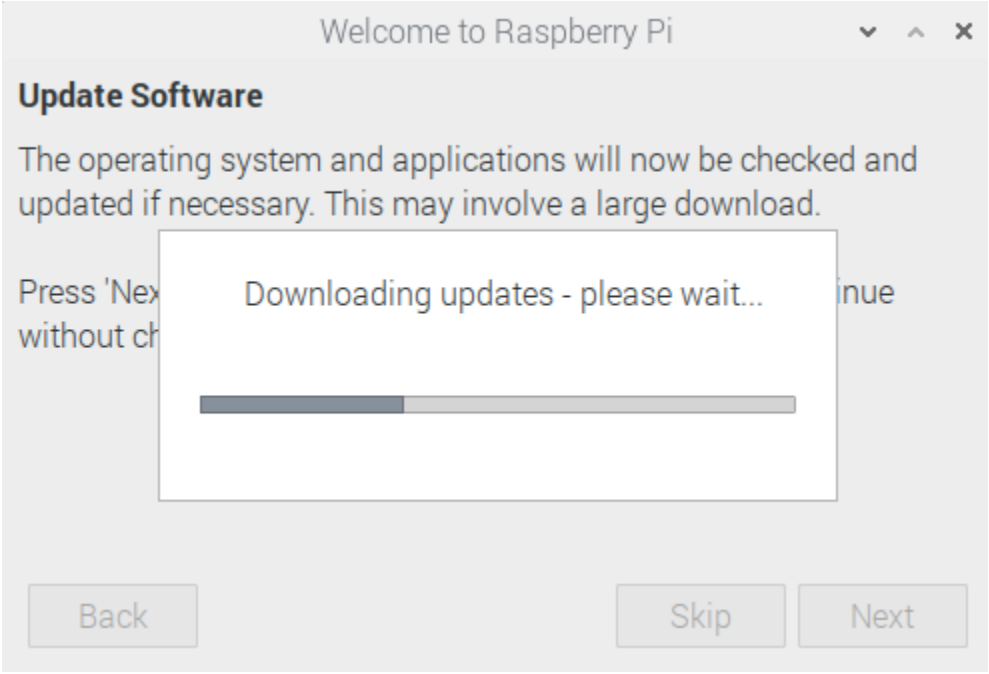


Fig. 28: Proceso de Configuración inicial de Raspbian

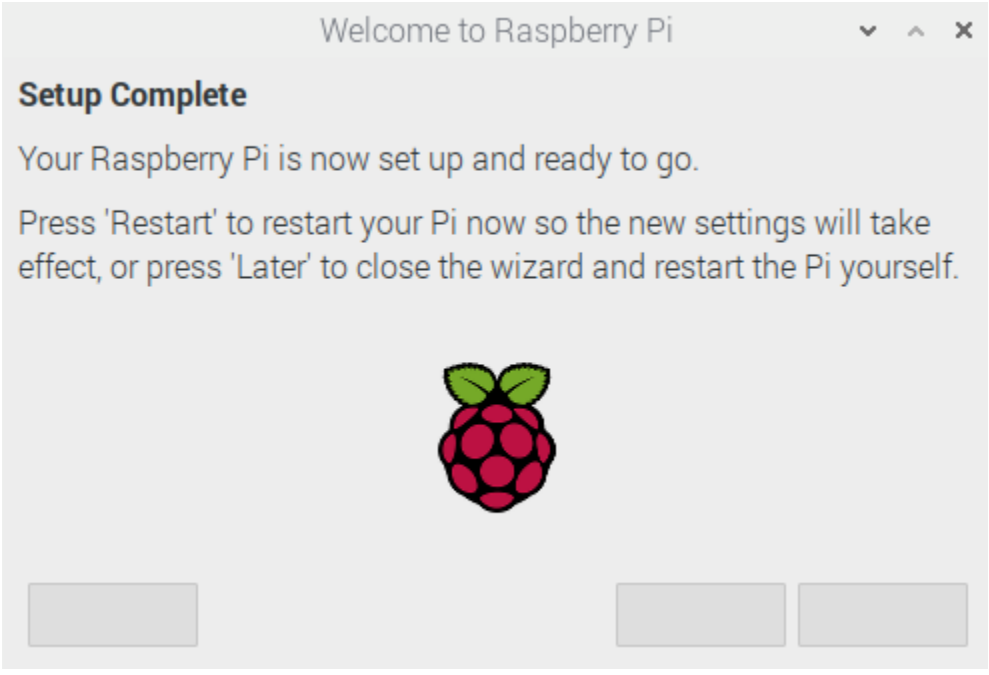


Fig. 29: Proceso de Configuración inicial de Raspbian

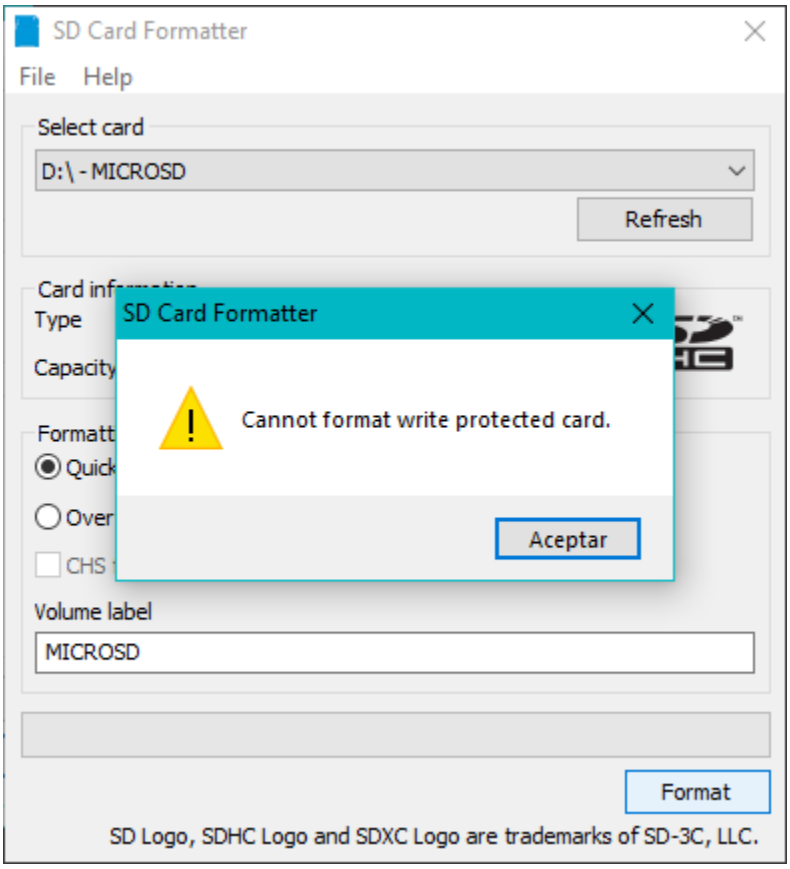


Fig. 30: Cannot format write protected card

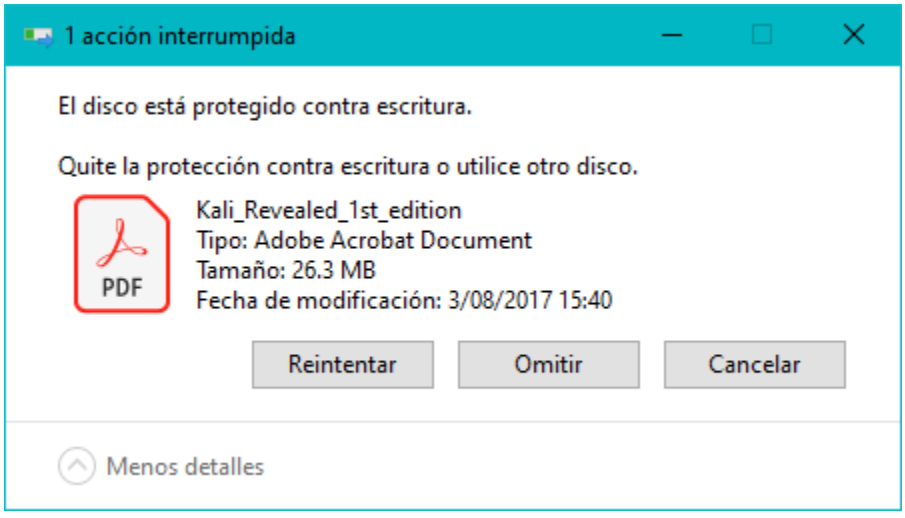


Fig. 31: El disco está protegido contra escritura

3. Seguir el siguiente tutorial: Cómo quitar la protección contra escritura de un USB en Windows

22.2.5 Links importantes

- NOOBS (New Out of Box Software) GitHub page: para una configuración más avanzada y completa de NOOBS y ver su código fuente visitar su página en GitHub.
- PINN GitHub page: An easy enhanced Operating System installer for the Raspberry Pi
- Projects Raspberry - Installing Raspbian with NOOBS
- Getting started with NOOBS
- Rasberry Forum screenshots on NOOBS
- VNC to NOOBS GitHub issues
- How to Take Raspberry Pi Screenshots With VNC
- NOOBS credentials Stackexchange
- Boot Multiple OSs: 3 Ways to Boot Multiple OSes on a Raspberry Pi

CHAPTER 23

Touchscreen

23.1 Instalar pantalla táctil 3.5'

Para la instalación de la pantalla táctil se usa un programa llamado **Elecrow**.

Table of Contents

23.1.1 Obtener paquetes

```
$ cd ~
$ git clone https://github.com/Elecrow-keen/Elecrow-LCD35.git
```

23.1.2 Cambiar a modo pantalla 3.5'

Este modo deja initulizable la pantalla HDMI.

```
$ cd ~/Elecrow-LCD35
$ sudo ./Elecrow-LCD35
```

23.1.3 Cambiar a modo pantalla HDMI

Este modo deja initulizable la pantalla táctil 3.5'.

```
$ cd ~/Elecrow-LCD35
$ sudo ./Elecrow-LCD35 hdmi
```

23.1.4 Scripts para automatizar cambio de modos

Script `mini-screen-mode.sh`

```
#!/bin/bash

cd ~/Programs/Elecrow-LCD35
sudo ./Elecrow-LCD35
```

Script `hdmi-mode.sh`

```
#!/bin/bash

cd ~/Programs/Elecrow-LCD35
sudo ./Elecrow-LCD35 hdmi
```

23.1.5 Links útiles

- Tutorial - 3.5 Inch 480x320 TFT Display with Touch Screen for Raspberry Pi
- Elecrow-LCD35 GitHub repository
- Switching back to HDMI output - GitHub issue

23.2 Calibración de pantalla táctil

Para la calibración de la pantalla se usa un programa llamado **xinput calibrator**.

Table of Contents

23.2.1 Instalación de paquetes

```
$ cd Elecrow-LCD35
$ sudo dpkg -i -B xinput-calibrator_0.7.5-1_armhf.deb
```

23.2.2 Calibración

Luego de instalar este paquete se creará una opción llamada *Calibrate Touchscreen*.

Para entrar ir a *Menú* → *Preferencias* → *Calibrate Touchscreen*.

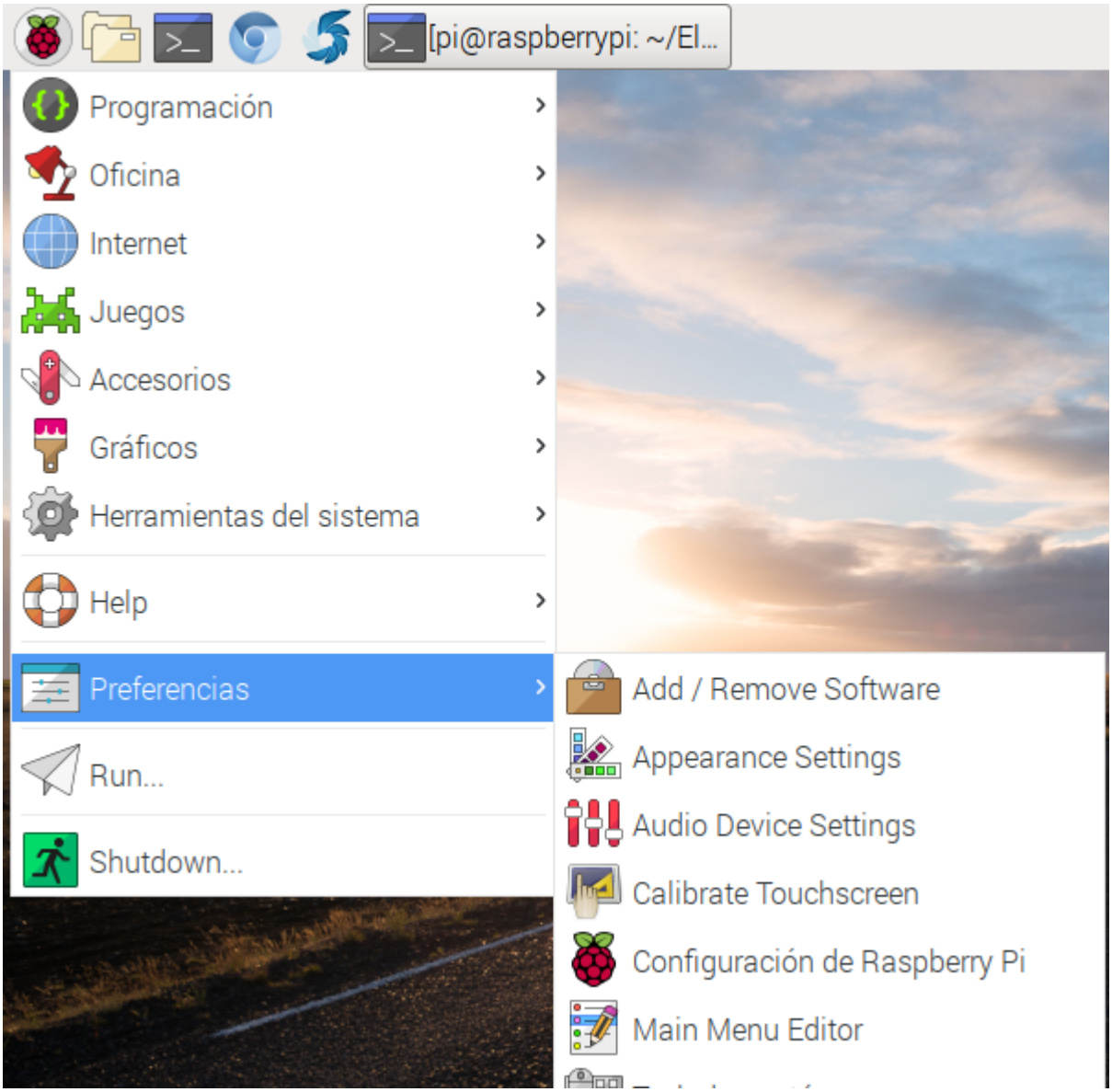


Fig. 1: Menú - Preferencias

Se puede entrar a este menú solo cuando se reconoce un dispositivo que pueda ser configurado. Por lo tanto cambiar de modo a **pantalla 3.5'**.

```
$ cd ~/Elecrow-LCD35
$ sudo ./Elecrow-LCD35
```

Usar *Calibrate Touchscreen* para calibrar la pantalla táctil. Una vez acabada la calibración, se generará un archivo con la siguiente ruta: `/etc/X11/xorg.conf.d/99-calibration.conf`

Opcionalmente podemos copiar esta configuración en un archivo distinto: `cp /etc/X11/xorg.conf.d/99-calibration.conf ~/Documents/files/99-calibration.conf`

```
# --- added by elecrow-pitft-setup sáb jun  1 19:13:04 -05 2019 ---
Section "InputClass"
    Identifier      "calibration"
    MatchProduct    "ADS7846 Touchscreen"
    Option  "Calibration"   "3917 250 242 3863"
    Option  "SwapAxes"      "1"
EndSection
# --- end elecrow-pitft-setup sáb jun  1 19:13:04 -05 2019 ---
```

CHAPTER 24

Raspbian configuration

24.1 Programas instalados

Table of Contents

24.1.1 Verificar la versión de Raspbian

```
$ cat /etc/os-release

PRETTY_NAME="Raspbian GNU/Linux 10 (buster)"
NAME="Raspbian GNU/Linux"
VERSION_ID="10"
VERSION="10 (buster)"
VERSION_CODENAME=buster
ID=raspbian
ID_LIKE=debian
HOME_URL="http://www.raspbian.org/"
```

(continues on next page)

(continued from previous page)

```
SUPPORT_URL="http://www.raspbian.org/RaspbianForums"
BUG_REPORT_URL="http://www.raspbian.org/RaspbianBugs"
```

24.1.2 Verificar arquitectura del sistema

```
$ uname -a
Linux raspberrypi 4.19.75-v7+ #1270 SMP Tue Sep 24 18:45:11 BST 2019 armv7l GNU/Linux
```

El sistema tiene una arquitectura `armv7 32 bits`.

24.1.3 Instalar software-properties-common

```
$ sudo add-apt-repository
sudo: add-apt-repository: command not found

$ sudo apt-get install software-properties-common
```

24.1.4 Instalar Visual Studio Code

Basado en: Installing VS Code on Raspbian buster

1. Instalar GPG key

```
$ sudo wget -qO - https://packagecloud.io/headmelted/codebuilds/gpgkey | sudo apt-key␣
↪add -;
OK
```

2. Añadir source repository

```
$ sudo nano /etc/apt/sources.list
```

... y añadir:

```
deb https://packagecloud.io/headmelted/codebuilds/raspbian/ jessie main
```

Note: Usar la versión `jessie` de Raspbian aún cuando tengamos la versión buster.

3. Instalar VS Code (code-oss)

```
$ sudo apt-get update
$ sudo apt-get install code-oss
```

Ejecutar `sudo apt-get update` es esencial para que funcione

4. Ejecutar VS Code (code-oss)

```
$ code-oss
```

5. Versión de VS Code

```
$ code-oss -version
```

24.1.5 Instalar Teamviewer

Basado en: How to Install Teamviewer on Raspberry Pi 2/3/4 Within a Minute

1. Descargar de la página web oficial de descargas de Teamviewer para Linux, la versión de Teamviewer para Raspbian con **arquitectura armv7 32bit**.
2. Instalar el paquete `.deb`. Se tendrá varias dependencias faltantes:

```
$ sudo dpkg -i ~/Downloads/teamviewer-host_15.1.3937_armhf.deb
```

3. Para descargar las dependencias ejecutar:

```
$ sudo apt-get update
```

4. Instalar las dependencias:

```
$ sudo apt-get -f install
```

24.1.6 Instalar redshift

1. Instalar redshift

```
$ sudo apt-get install redshift
```

2. Ejecutar el siguiente comando:

```
$ sudo raspi-config
```

Ir a *Advanced Options*, luego *GL Driver* y seleccionar *G3 GL (FUll KMS)*.

3. Al hacer el cambio pedirá reiniciar el sistema. Aceptar.
4. Comprobar que redshift funcione ejecutando en el terminal:

```
$ redshift -O 3500
```

5. Regresar a un modo normal: `-x Reset mode (remove adjustment from screen)`

```
$ redshift -x
```

24.1.7 Instalar aplicación para screenshots

Scrot es una aplicación de línea de comandos para tomar capturas de pantallas en un Raspberry Pi.

Generalmente Scrot ya viene instalado con Raspbian, de no ser el caso, instalarlo con:

```
$ sudo apt-get install scrot
```

Tenemos varias opciones de uso para capturar la pantalla:

1. Tomar una captura instantánea a toda la pantalla:

```
$ scrot
```

2. Tomar una captura con delay:

```
# Delay de 3 segundos:
$ scrot -d 3
```

Agregar un contandor al terminal:

```
$ scrot -c -d 3
```

3. Tomar captura solo a la aplicación actual (se agrega un delay para poder cambiar a la ventana deseada):

```
$ scrot -u -d 3
```

4. Tomar una captura a solo un área seleccionada:

```
$ scrot -s
```

5. Ejecutar una aplicación sobre la imagen guardada:

```
# Abrir la imagen con el visor de imágenes por defecto de Raspbian
$ scrot -e gpicview
```

24.2 Cambiar layout del teclado

Para cambiar el layout del teclado, es decir, la distribución de las teclas según el idioma, ir a *Preferences –> Keyboard and Mouse*:

Luego seleccionar la pestaña *Keyboard*, Opción *Keyboard Layout*:

Finalmente seleccionar el layout deseado según tu idioma, *OK*:

24.3 Habilitar acceso remoto

Para habilitar el acceso remoto vía SSH y VNC ir a *Preferences –> Raspberry Pi Configuration*:

En la pestaña *Interfaces*, habilitar SSH y VNC, poniéndolos en estado *Enable*. Seleccionar *OK*:

24.4 VNC (Virtual Network Computing)

Basado en: Documentation Raspberry Pi - Remote Access - VNC

Podemos usar el Raspberry Pi remotamente a través de VNC. Esta aplicación nos permite controlar la interfaz de escritorio de una computadora en la que esté corriendo un **VNC Server** desde otra computadora corriendo un **VNC Viewer**. La idea es que podamos controlar el Raspberry Pi desde una ventana en nuestra computadora o una aplicación en nuestro smartphone como si estuviésemos trabajando frente al Raspberry Pi mismo.

VNC Connect de **RealVNC** viene incluido en la distribución Raspbian. Esta consiste de un VNC Server y un VNC Viewer.

Debemos *habilitando el VNC Server* antes de poder usarlo, dándonos acceso remoto a la interfaz gráfica que corre en el Raspberry Pi. Sin embargo, también podemos usar el VNC Server para obtener acceso gráfico a nuestro Raspberry

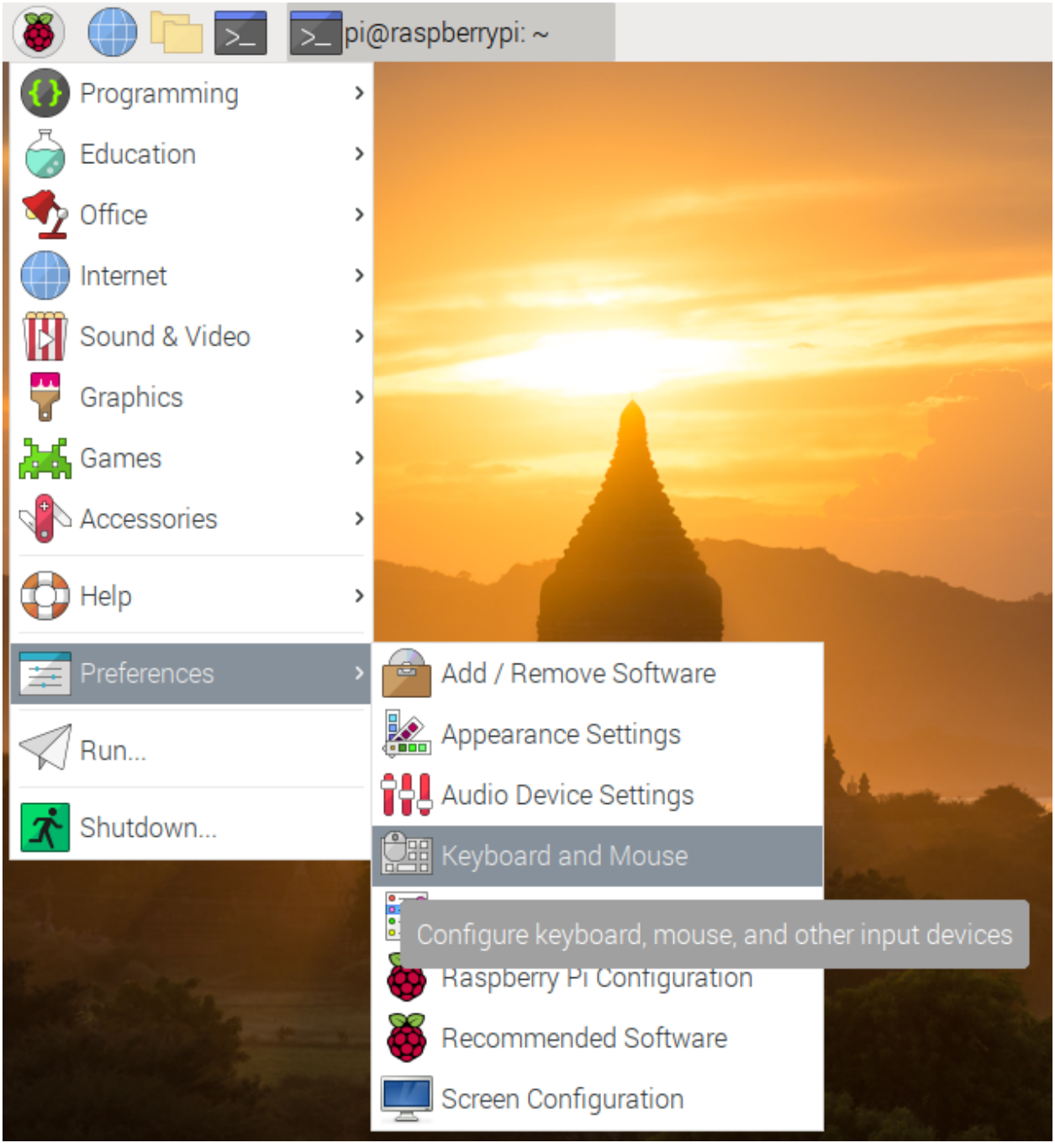


Fig. 1: *Preferences –> Keyboard and Mouse*

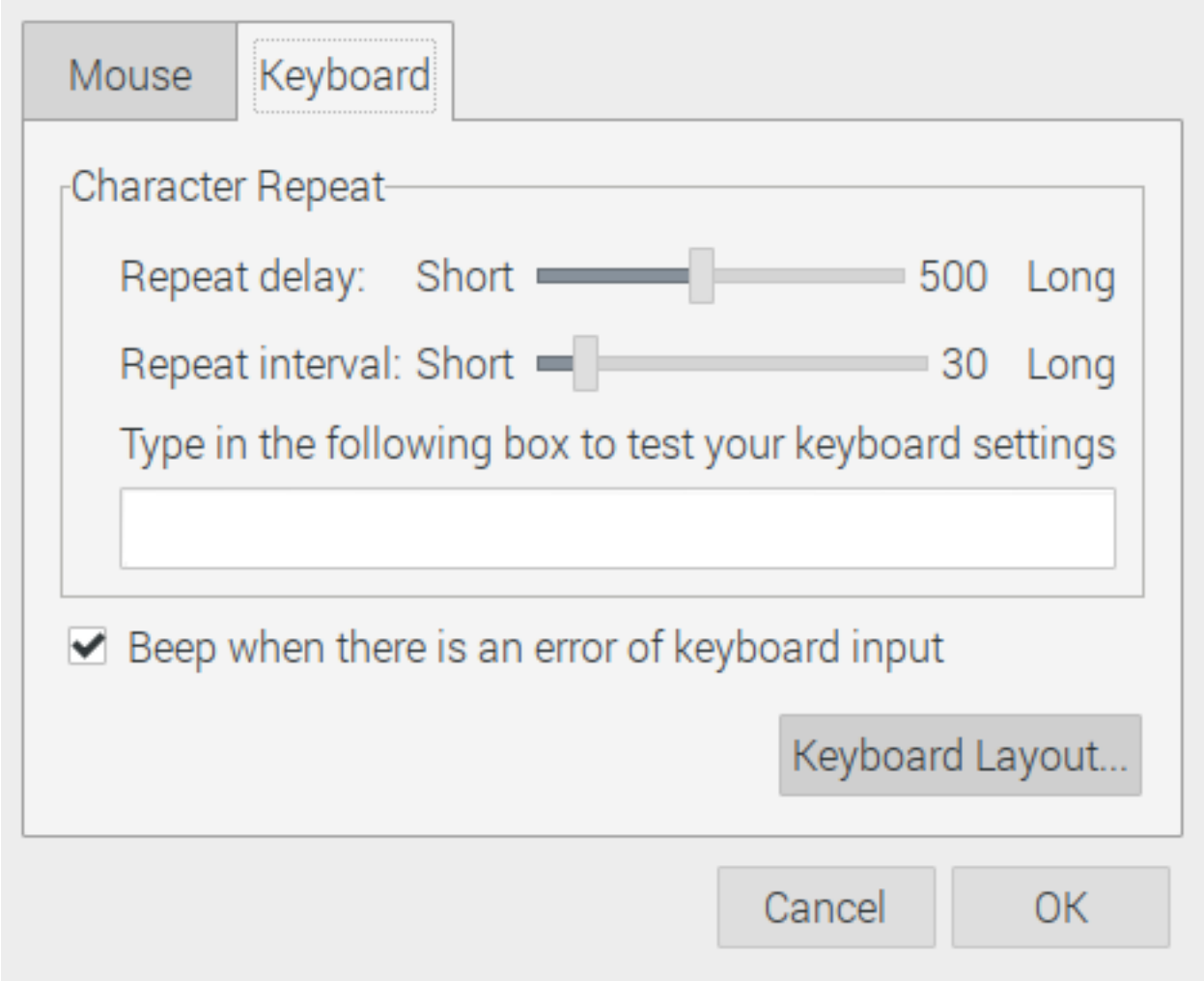


Fig. 2: Pestaña *Keyboard*, Opción *Keyboard Layout*

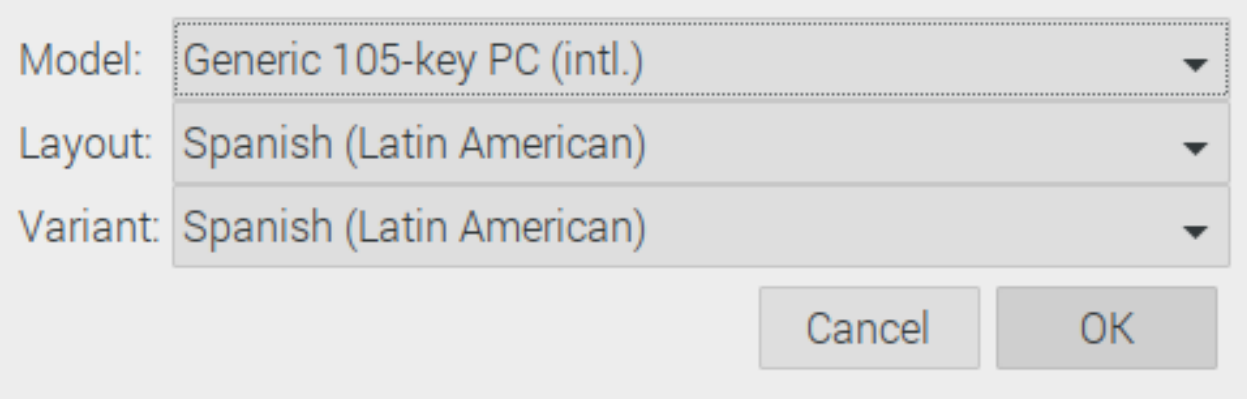


Fig. 3: Seleccionar el layout deseado, *OK*

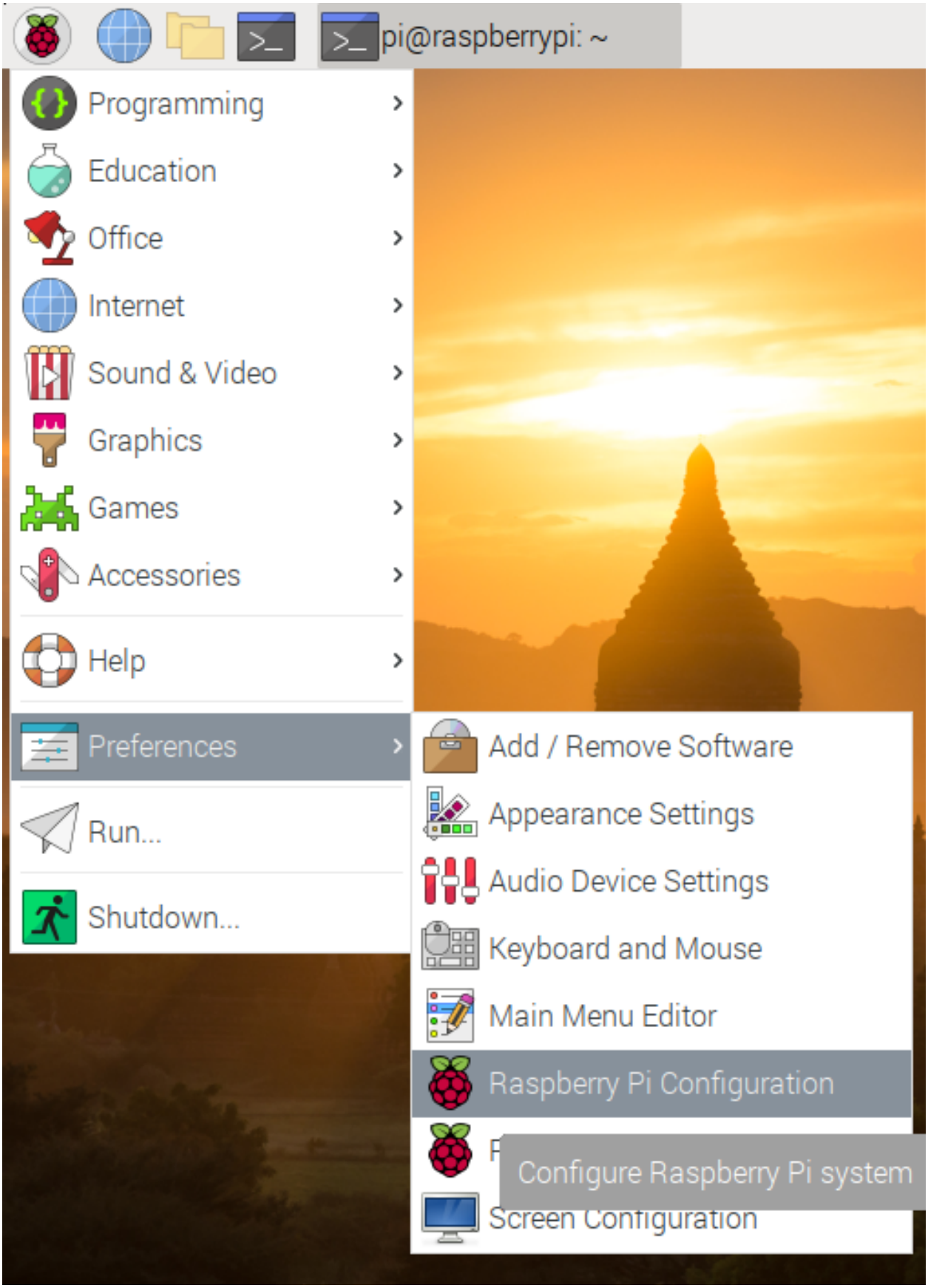


Fig. 4: *Preferences –> Raspberry Pi Configuration*

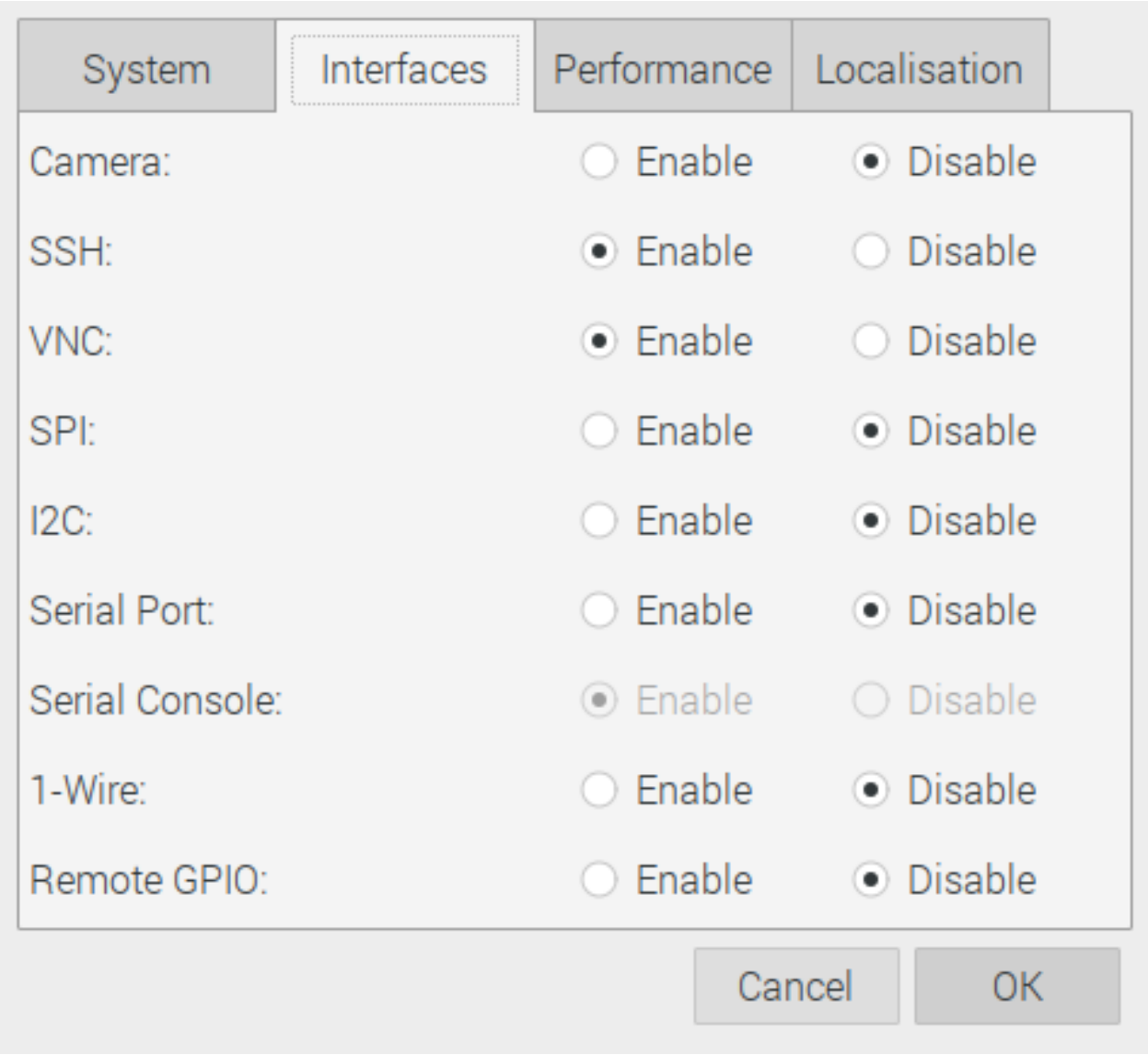


Fig. 5: Pestaña *Interfaces*, habilitar SSH y VNC, seleccionar *OK*

Pi si es **headless** (es decir, no está conectado a un monitor) o no está corriendo un escritorio gráfico. Ver la sección *Creando un escritorio virtual*.

24.4.1 Habilitando el VNC Server

En el Raspberry Pi asegurarnos que tenemos la última versión de VNC Connect:

```
$ sudo apt update
$ sudo apt install realvnc-vnc-server realvnc-vnc-viewer
```

Para habilitar VNC Server podemos hacerlo de forma gráfica o por el terminal:

Habilitando VNC Server de forma gráfica

Seguir documentación: *Habilitar acceso remoto*

Habilitando VNC Server en el terminal

```
$ sudo raspi-config
```

Escoger la opción *Interfacing Options* y seleccionar *VNC > Yes*.

24.4.2 Estableciendo conexión a nuestro Raspberry Pi con VNC Viewer

Existen 2 formas para conectarnos a nuestro Raspberry Pi:

Estableciendo una conexión directa

Las conexiones directas son una forma simple y rápida de conexión a través de la misma red de nuestro Raspberry Pi.

1. Obtener la dirección IP de la interfaz de red de nuestro Raspberry Pi usando `ifconfig` o `ip addr`.
2. Desde el dispositivo desde el cual controlaremos el Raspberry Pi, descargar VNC Viewer. Preferentemente la aplicación de RealVNC.
3. Ingresar la dirección IP de nuestro Raspberry Pi en el VNC Viewer de la computadora:
4. *Autenticándonos al VNC Server*

Estableciendo una conexión cloud

Podemos utilizar el servicio en la nube de RealVNC para conectarnos a nuestro Raspberry Pi a través de Internet. Las conexiones cloud son encriptadas end-to-end y no es necesaria una reconfiguración de firewall o router. Tampoco es necesario saber la dirección IP de nuestro Raspberry Pi.

1. Primero debemos crear una cuenta en RealVNC de forma gratuita.
2. En el VNC Server del Raspberry Pi iniciar sesión yendo a la opción *Licensing*.
3. En la computadora que usaremos para controlar el Raspberry Pi, usar VNC Viewer (de RealVNC)
4. Una vez que hemos iniciado sesión desde el VNC Viewer con la misma cuenta usada para el VNC Server del Raspberry Pi veremos un nuevo equipo disponible para conectarnos en la sección de `Team`:
5. *Autenticándonos al VNC Server*

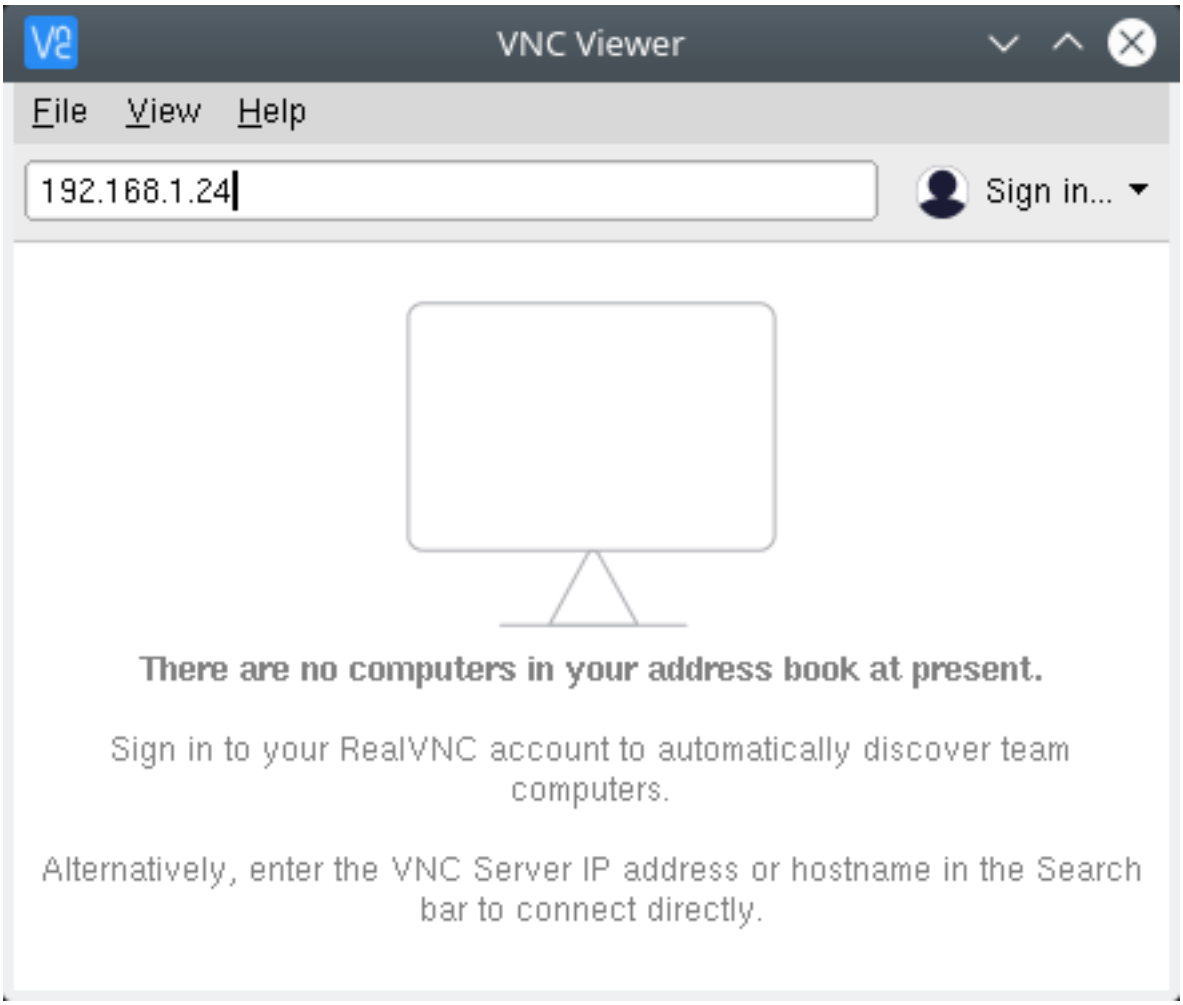


Fig. 6: Conexión directa a Rasberry Pi desde VNC Viewer

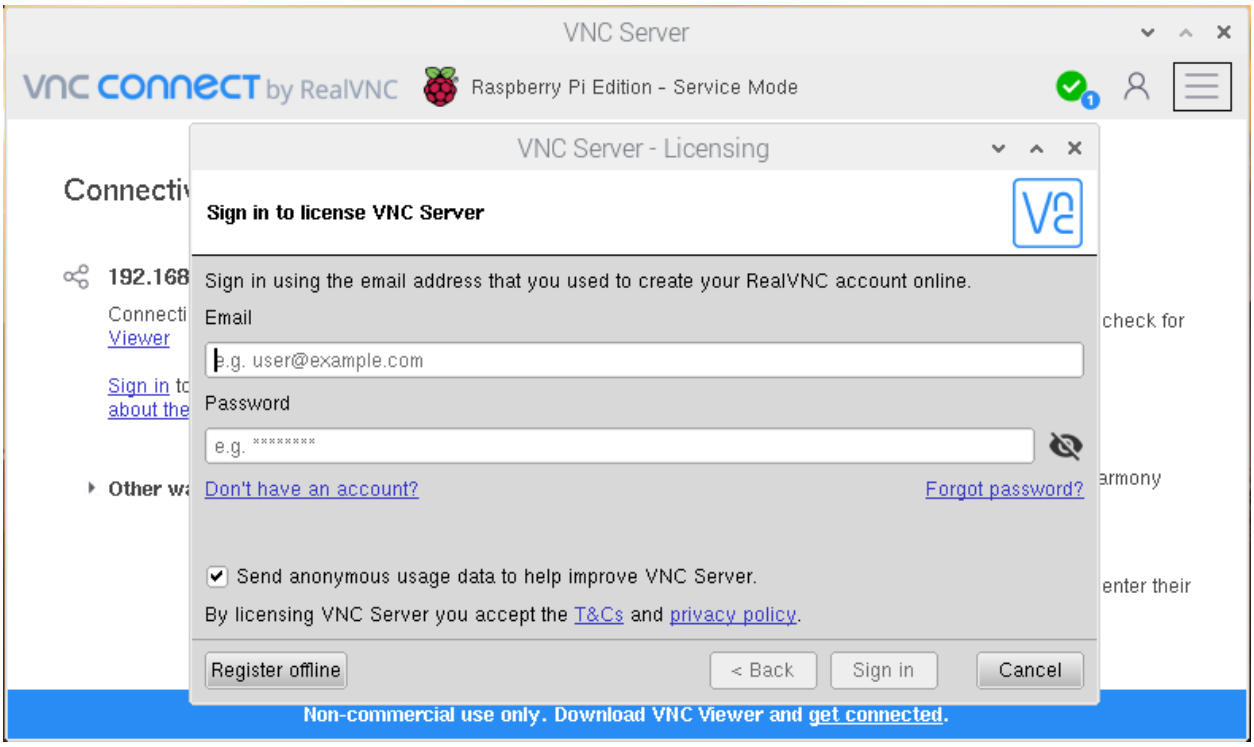


Fig. 7: Iniciar sesión en el Raspberry Pi desde VNC Server

Autenticándonos al VNC Server

Para completar cualquier método conexión, ya sea *Estableciendo una conexión directa* o *Estableciendo una conexión cloud*, debemos autenticarnos en el VNC Server.

Si estamos conectándonos de una aplicación VNC Viewer compatible de RealVNC, ingresar el usuario y password del Raspberry Pi que normalmente usaríamos para usar loguearnos en nuestra cuenta de usuario. Por defecto, las credenciales son `pi:raspberry`.

24.4.3 Creando un escritorio virtual

Si nuestro Raspberry Pi es **headless** (es decir, no está conectado a un monitor) o no está corriendo un escritorio gráfico podemos usar **VNC Server** para crear un escritorio **virtual (virtual desktop)**. Brindándonos acceso remoto gráfico. Este escritorio virtual solo existe en la memoria de nuestro Raspberry Pi:

Para crear y conectarnos a un escritorio virtual:

1. En nuestro Raspberry Pi (usando el terminal o vía SSH) correr:

```
$ vncserver

VNC(R) Server 6.5.0 (r41824) ARMv6 (Aug 16 2019 00:24:44)

[...]

Running applications in /etc/vnc/xstartup

VNC Server catchphrase: "Instant amigo printer. Russian Boston orca."
```

(continues on next page)

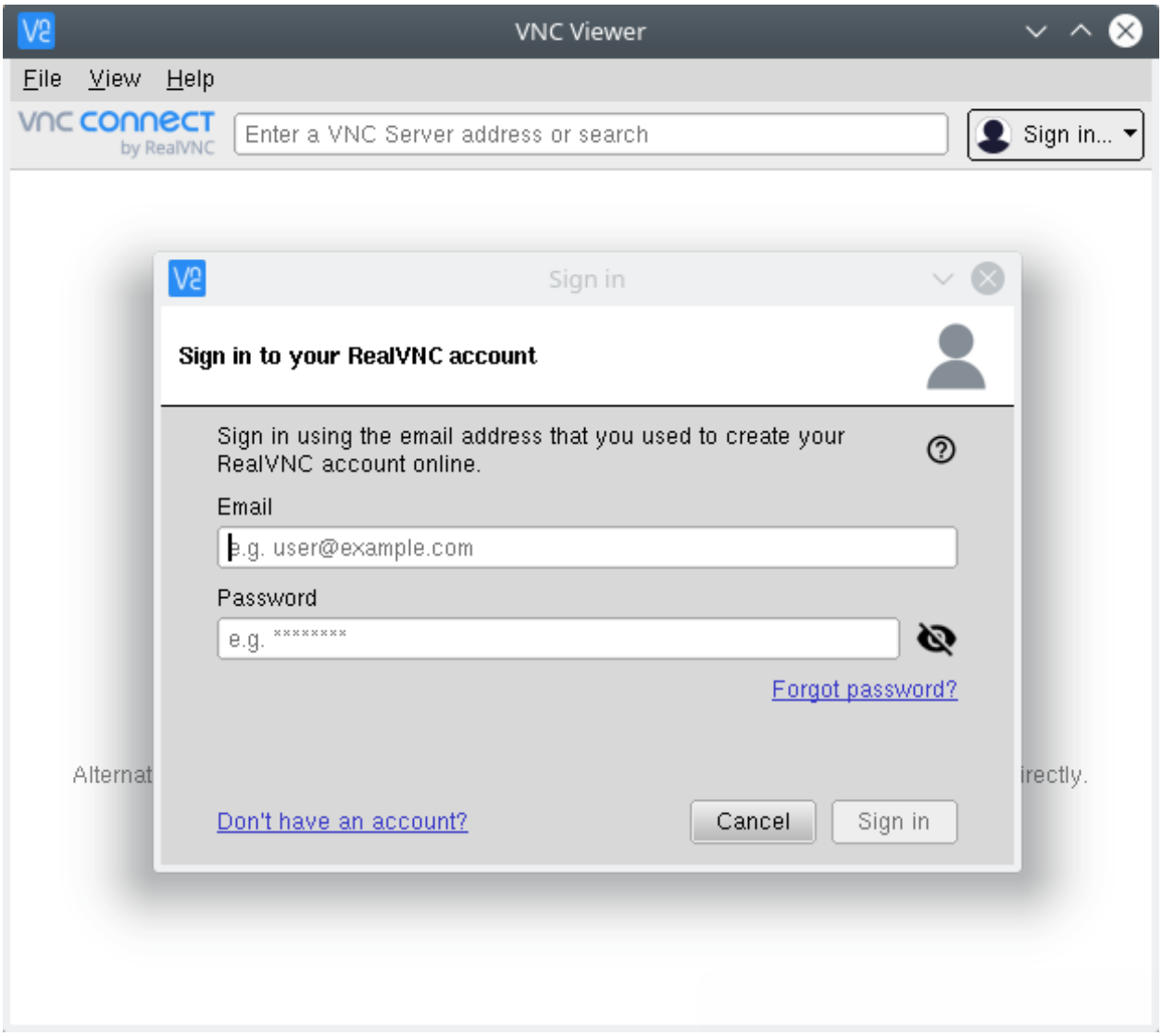


Fig. 8: Iniciar sesión en el Raspberry Pi desde VNC Server

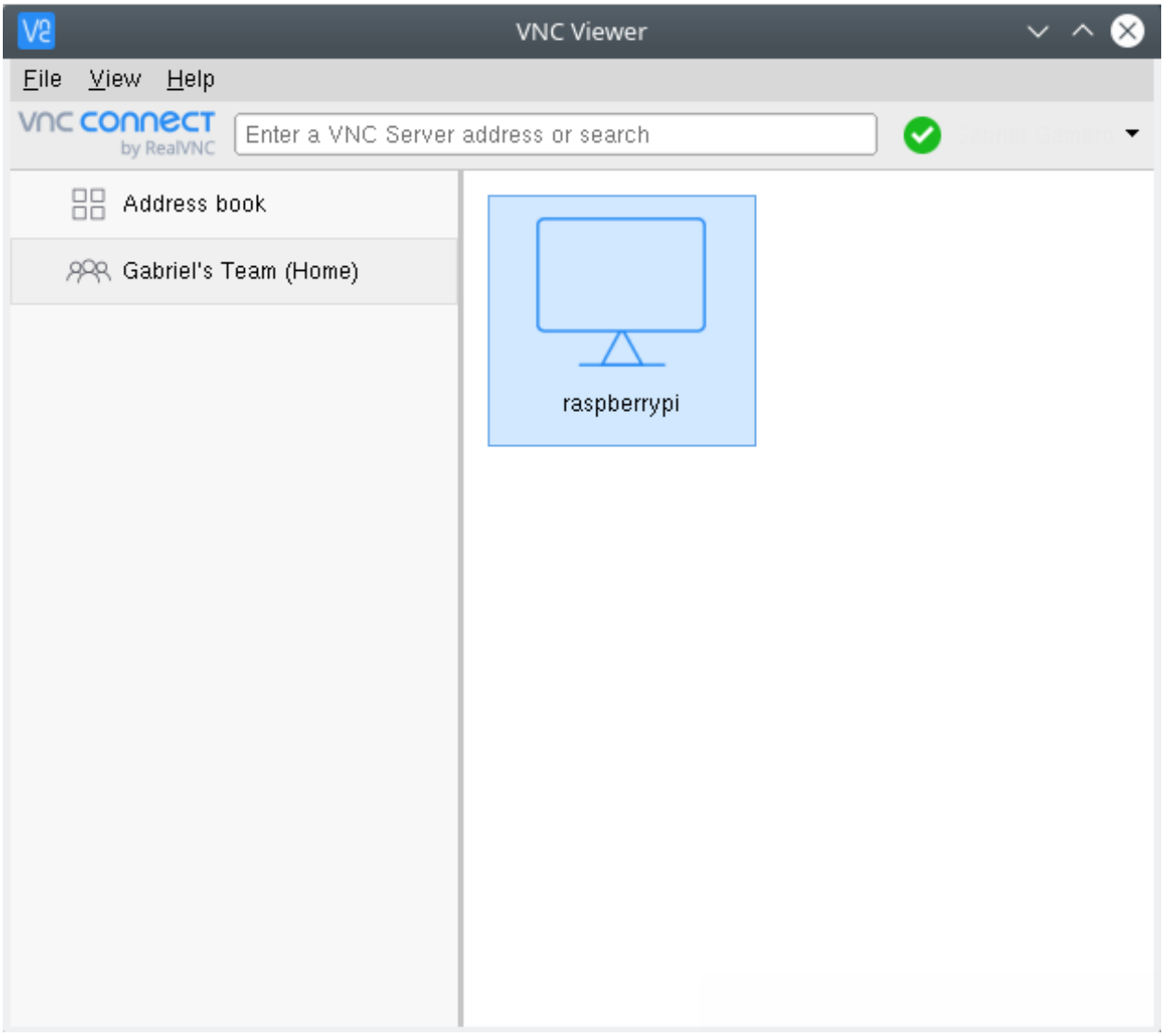


Fig. 9: Nuevo equipo disponible para conectarnos (Raspberry Pi) en la sección `Team`

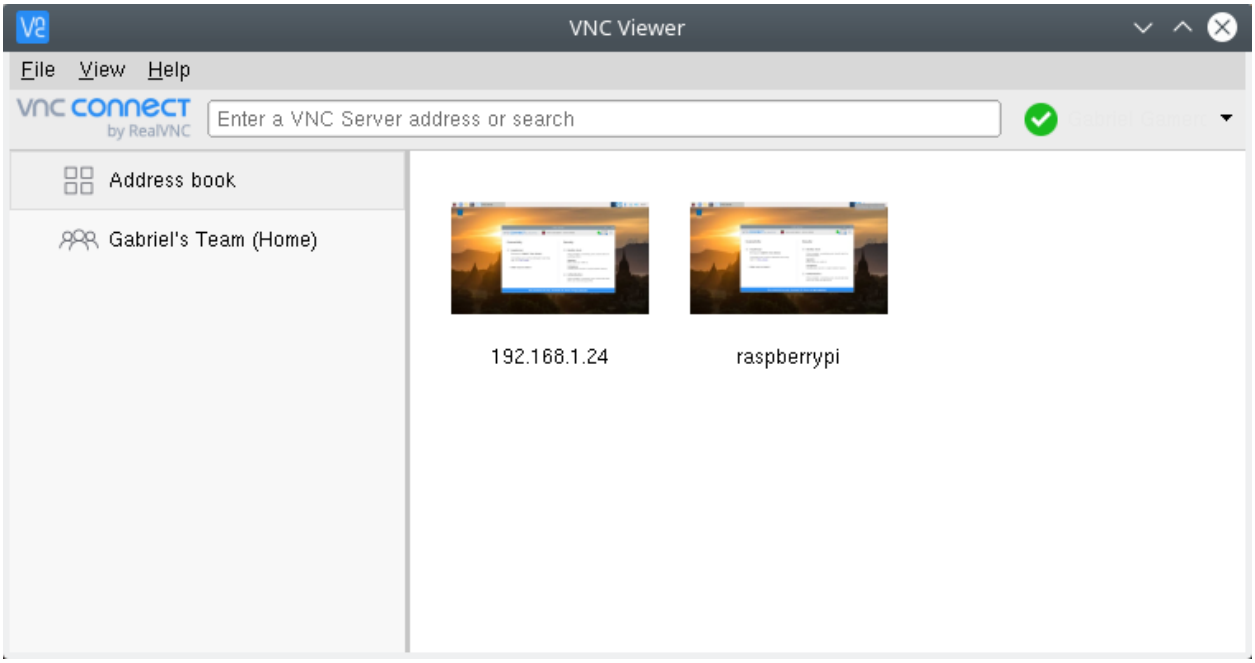


Fig. 10: Mismo equipo Raspberry Pi con 2 tipos de conexión: directa (izquierda) y cloud (derecha)

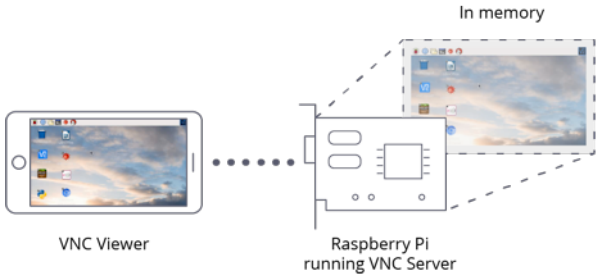


Fig. 11: Creating a virtual desktop - Fuente: https://www.raspberrypi.org/documentation/remote-access/vnc/

(continued from previous page)

```
            signature: 61-7f-1d-5d-fa-37-a6-v7

Log file is /home/pi/.vnc/raspberrypi:1.log
New desktop is raspberrypi:1 (192.168.1.24:1)
```

Important: Tomar nota de la dirección IP y del identificador que VNC Server usa para este escritorio virtual creado (display number). En este caso: `192.168.1.24:1`

2. En el dispositivo que usaremos para controlar el Raspberry Pi ingresar esta información en el VNC Viewer.
3. Para eliminar un escritorio virtual correr el siguiente comando:

```
# vncserver -kill :<display-number>
$ vncserver -kill :1
```

CHAPTER 25

SDN

CHAPTER 26

Virtualization

MASTERING KVM VIRTUALIZATION:

- **CH3:**
  - **Revisión de los requerimientos para nuestro entorno de virtualización (Determining the right system requiremer**
    - Búsqueda de flags: grep –color -Ew 'svm|vmx|lm' /proc/cpuinfo
    - Búsqueda de módulos KVM: lsmod | grep kvm
  - **Instalación de los paquetes de virtualización (Setting up the environment)**
    - Con yum: yum install qemu-kvm libvirt virt-install virt-manager -y
    - Con apt (investigar): apt-get install virt-manager
    - Instalación grupal: yum groupinstall "virtualization" -y
  - Inicializar los servicios: systemctl enable libvirtd && systemctl start libvirtd
  - Ver versión de libvirt: libvirtd –version
  - **Verificar capacidades de virtualización del sistema:**
    - virt-host-validate
    - virsh nodeinfo
    - virsh domcapabilities (virsh domcapabilities | grep -i max), (virsh domcapabilities | grep diskDevice -A 5)
  - Conexión en virt-manager (Hardware configuration examples)

Tutoriales:

1. Revisar los requerimientos para virtualización QEMU/KVM
2. Instalación de paquetes para virtualización QEMU/KVM (yum, apt) e iniciar servicios
3. Crear una conexión en virt-manager

- **CH4:**
  - polkit (Introducing virt-manager)
  - Virtual Network Tab: virt-manager
  - listado de redes con virsh (Virtual Network Tab): virsh net-list –all
  - detalle de red con virsh (Virtual Network Tab): `virsh net-info default` , `virsh net-dumpxml default`
  - Comandos básicos de virsh: `virsh net-destroy` y `virsh net-start`
  - Storage tab de `virt-manager`
  - **Creación de una VM con wizard de virt-manager (Creating a new virtual machine wizard)**
    - Base de datos de SOs: libosinfo, sinfo-query
  - Método de instalación por red en virt-manager (The Network installation (HTTP, FTP, or NFS) method)
  - Método Instalación por PXE en virt-manager, con macvtap y bridge (Network Boot (PXE))
  - Método de Instalación con una imagen de disco existente (Importing an existing disk image)
  - **Usando virt-install para la creación de una VM y configuración del SO guest. (Introducing `virt-install`, Ins**
    - comando `qemu-img` para crear un disco virtual
    - comando `virt-install --prompt` para una instalación interactiva de la VM.
  - **Instalación y uso de `virt-builder` para personalizar un SO con plantillas. Uso junto `virt-install` para a**
    - uso `virt-builder` y combinación con `virt-install`
    - listar VMs con `virsh list --all`
    - iniciar la VM con `virsh start <vname>`
    - opciones de virt-builder y virt-builder –note <guest>
    - caché de virt-builder: `virt-builder --print-cache`
    - eliminar caché: `virt-builder --delete-cache`
    - descargar todas las plantillas a caché: `virt-builder --cache-all-templates`
  - **Creación de una VM usando `oz` (Introducing `oz`): instalación de `oz`, archivo TDL, `oz-install` para construi**
    - Archivo de configuración para creación de VMs en `oz`: `/etc/oz/oz.cfg` (The oz configuration file)
    - Crear una VM usando `oz` (Creating a virtual machine using `oz`)

Tutoriales:

1. Creación de VMs usando `virt-manager` (métodos de medio: ISO, PXE, HTTP)
2. Ver detalles de red con `virsh`
3. Creación de VMs usando `virt-install` (+ `qemu-img` para crear un disco virtual)
4. Creación de VMs usando `virt-builder`
5. Creación de VMs usando `oz`

(*) Falta complementar material de `qemu-img`

- CH 5:

  NETWORKING: - Ejemplo de virtual Networking con Linux bridges e interfaces TAP (Virtual Networking) - Virtual Networking usando libvirt: con clientes `virt-manager` y `virsh`

  - Isolated
  - Routed
  - NATed
  - Bridged

  STORAGE: - Unmanaged storage - Creación de un disco virtual (`dd`): discos preallocated y thin-provisioned - Información de una imagen (raw .img, qcow2, ...) con `qemu-img info` - Conexión de un disco a una VM (Unmanaged storage):

  - usando `virt-manager`: add new virtual hardware
  - usando `virsh`: `virsh attach disk` (discos conectados a una VM: `virsh domblklist`)
  - Managed storage (tipos de pools soportados por libvirt)
  - **Administración y configuración de almacenamiento (pools):**
    - con `virt-manager`: Pestaña Storage de Connection Details.
    - con `virsh`: listar pools (`virsh pool-list`), obtener información de un pool (`virsh pool-info`)
  - **Creando pools de almacenamiento (XML de pools guardadas bajo `/etc/libvirt/storage`):**
    - **Pool con almacenamiento basado en archivos**
      - con `virt-manager`: `dir: Filesystem Directory`
      - con `virsh`: `virsh pool-define-as`, `virsh pool-build`, `virsh pool-start`
      - extra (start y autostart del pool): `virsh pool-start`, `virsh pool-autostart`
    - **Pool con LVM Volume group**
      - con `virt-manager`: `logical: LVM Volume Group`
      - con `virsh`: `virsh pool-define-as`, `virsh pool-build`, `virsh pool-start`
    - **Pool con iSCSI**
      - con `virt-manager`: `iscsi: iSCSI Target`
  - **Creando una biblioteca de imágenes ISO:**
    - con `virsh`: `virsh pool-define-as`, `virsh pool-build`, `virsh pool-start`
    - extra (actualizar contenidos del pool): `virsh pool-refresh`
  - **Eliminando un pool**
    - con `virt-manager`: Pestaña Storage de Connection Details.

        * con `virsh`: `virsh pool-destroy`, `virsh pool-undefine`

    - **Crear volúmenes de almacenamiento:**

        * con `virt-manager`: Pestaña Storage de Connection Details.

        * con `virsh`: `virsh vol-create-as`, `virsh vol-info`

    - **Eliminando un volumen**

        * con `virsh`: `virsh vol-delete`

Tutoriales (almacenamiento):

1. Creación de volúmenes de almacenamiento con virt-manager y virsh
2. **Unmanaged y Managed Storage en libvirt** 1.1 Unmanaged storage: crear un disco no asociado con libvirt y unirlo a una VM (métodos de creación de un volumen: virt-manager, virsh, qemu-img, dd 1.2 Managed storage: creación, eliminación de pools (basado en archivos, LVM volume group): virt-manager, virsh

CHAPTER 27

QEMU/KVM Virtualization

27.1 Requerimientos de virtualización

Table of Contents

Según la cantidad de VMs y la carga de trabajo que deseemos correr podemos determinar los requerimientos físicos del equipo encargado de la virtualización y el aprovisionamiento de recursos, es decir, el equipo que tendrá el **hypervisor QEMU/KVM**. En base a esto decidiremos cuántos CPUs, memoria RAM y almacenamiento se asigna a cada VM.

27.1.1 Requerimientos de CPU

Marcas fabricantes de CPUs han integrado **extensiones de virtualización** dentro de sus procesadores de **arquitectura x86**. A través de estas extensiones, KVM emplea una tecnología llamada **hardware-accelarated virtualization**; que para CPUs de Intel recibe el nombre de **VT-x (Virtual Machine Extension)**, mientras que en AMD es llamado **AMD-V (AMD Virtualization)**. Gracias a esto, es posible ejecutar el código de un Sistema Operativo (SO) guest en el CPU del host directamente. Referencia de virtualización x86.

Desde un sistema Linux podemos descubrir si nuestros procesadores cuentan con esta tecnología buscando un flag especial que lo compruebe: para Intel VT-x es `vmx` y para AMD-V es `svm`. Además si el CPU tiene una **arquitectura de 64 bits** veremos un flag extra de interés: `lm`.

Para buscar los flags ejecutar:

```
$ grep --color -Ew 'svm|vmx|lm' /proc/cpuinfo
flags           : fpu vme de pse tsc msr pae mce cx8 apic sep mtrr pge mca cmov pat
pse36 clflush dts acpi mmx fxsr sse sse2 ss ht tm pbe syscall nx rdtscp lm constant_
↪tsc
arch_perfmon pebs bts rep_good nopl xtopology nonstop_tsc cpuid aperfmperf pni␣
↪pclmulqdq
dtes64 monitor ds_cpl vmx est tm2 ssse3 cx16 xtpr pdcm pcid sse4_1 sse4_2 x2apic␣
↪popcnt
tsc_deadline_timer aes xsave avx f16c rdrand lahf_lm cpuid_fault epb pti ssbd ibrs␣
↪ibpb
stibp tpr_shadow vnmi flexpriority ept vpid fsgsbase smep erms xsaveopt dtherm ida␣
↪arat
pln pts md_clear flush_l1d
```

- `svm`: CPU tiene AMD-V
- `vmx`: CPU tiene VT-x
- `lm`: soporte de 64-bit

En el ejemplo de arriba vemos los flags de `vmx` y `lm`, significa que estamos empleando un procesador de 64-bits marca Intel en el host.

Otro componente necesario para tener un sistema de virtualización funcional son los módulos de kernel responsables de soportar virtualización. Estos módulos pueden encontrarse en un sistema Linux con la herramienta `lsmod` (lsmod o loadble kernel modules):

```
$ lsmod | grep kvm

kvm_intel             217088  0
kvm                   610304  1 kvm_intel
```

En sistemas AMD veremos `kvm_amd` en vez de `kvm_intel`.

En el caso que no aparezca algún módulo, deberemos cargarlo manualmenete con la herramienta de Linux `modprobe`:

```
$ modprobe kvm
$ modprobe kvm_intel  # Intel processors
$ modprobe kvm_amd    # AMD processors
```

Más información en Three main components - KVM.

27.1.2 Instalando herramientas

Instalaremos varios programas útiles para la virtualización:

En UBUNTU/DEBIAN (con `apt`)

```
# QEMU/KVM, Virtual Machine Manager, Libvirt, bridge-utils (brctl)
$ sudo apt-get install -y qemu-kvm virt-manager libvirt-bin bridge-utils
```

Mirando las dependencias de los paquetes instalados veremos que se han instalado otros paquetes importantes como: `virt-viewer`, `virtinst`, `libvirt-daemon-system`, `libvirt-clients`

Comandos útiles con `apt`:

- Paquetes instalados: `apt list --installed`
- Dependencia de paquetes: `apt-cache depends vim`
- Dependencia de paquetes reversa: `apt-cache rdepends vim` (o para ver los que tenemos instalados `apt-cache rdepends --installed vim`)

En CENTOS/RHEL (con `yum`)

```
$ sudo yum install -y qemu-kvm virt-manager libvirt virt-install virt-viewer
```

Comandos útiles con yum:

- Paquetes instalados: `yum list installed`
- Dependencia de paquetes: `yum deplist nfs-utils`

Note: El paquete `bridge-utils` viene pre-instalado en algunas distribuciones como CentOS 7 (GNOME)

27.1.3 Iniciando servicios

Habiendo instalado todos los paquetes de virtualización, habilitaremos el servicio de `libvirtd`, que es el daemon de `libvirt`:

```
$ systemctl enable libvirtd && systemctl start libvirtd
```

`libvirt` mostrará una API a clientes como `virt-manager` y `virsh` para poder comunicarse con `qemu-kvm`.

Note: Para ver la versión de `libvirt` en uso ejecutar: `libvirtd --version`

27.1.4 Validando capacidades del sistema

A continuación se presentan dos herramientas útiles para verificar que nuestro sistema es apto para servir como virtualizador y ver sus capacidades:

Validando con `virt-host-validate`

La herramienta `virt-host-validate` ejecutará una series de pruebas para medir las capacidades del sistema y evaluar si el host está configurado correctamente para servir como un host de virtualización KVM.

```
$ sudo virt-host-validate

QEMU: Checking for hardware virtualization                 : PASS
QEMU: Checking if device /dev/vhost-net exists             : PASS
QEMU: Checking if device /dev/net/tun exists               : PASS
QEMU: Checking for cgroup 'memory' controller support      : PASS
```

(continues on next page)

(continued from previous page)

```
QEMU: Checking for cgroup 'cpu' controller support      : PASS
LXC: Checking for Linux >= 2.6.26                       : PASS
LXC: Checking for namespace user                        : PASS
LXC: Checking for cgroup 'memory' controller support    : PASS
```

Si nuestro CPU no es capaz de realizar virtualización acelerada por hardware veremos un error en la revisión de pasos como se muestra en el siguiente ejemplo:

```
$ virt-host-validate

QEMU: Checking for hardware virtualization              : FAIL (Only emulated CPUs␣
↪are available, performance will be significantly limited)
QEMU: Checking if device /dev/vhost-net exists          : PASS
QEMU: Checking if device /dev/net/tun exists            : PASS
QEMU: Checking for cgroup 'memory' controller support   : PASS
QEMU: Checking for cgroup 'cpu' controller support      : PASS
LXC: Checking for Linux >= 2.6.26                       : PASS
LXC: Checking for namespace user                        : PASS
LXC: Checking for cgroup 'memory' controller support    : PASS
```

El mensaje nos indica que solo estará disponible la emulación de CPU, y el rendimiento se verá afectado por esto. En otras palabras el sistema solo tiene soporte para `qemu`, que es lento comparado con `qemu-kvm`.

También se validan otros parámetros del sistema como:

- `/dev/kvm`: facilita el acceso directo de la VMs al hardware (el usuario actual debe tener permisos de acceso)
- `/dev/vhost-net`: sirve como la interface para configurar la instancia vhost-net.
- `/dev/net/tun`: usado para crear **dispositivos tun/tap** para facilitar la conectividad de red a VMs.

Note: Asegurarnos que `virt-host-validate` pasa todas las pruebas satisfactoriamente para proceder con la creación de VMs.

Validando con `virsh`

Podemos usar `virsh` para obtener información sobre las características del host (CPU, memoria RAM) que obtendríamos normalmente usando comandos por separado con otras herramientas:

```
$ virsh nodeinfo

CPU model:           x86_64
CPU(s):              4
CPU frequency:       1296 MHz
CPU socket(s):       1
Core(s) per socket:  2
Thread(s) per core:  2
NUMA cell(s):        1
Memory size:         8068236 KiB
```

Otro comando útil de `virsh` es `virsh domcapabilities`. Este comando nos imprime un documento XML que describe las capacidades del dominio para el hypervisor al que estamos conectados. Dentro del archivo XML podemos extraer información como el número máximo de `vcpu` que se pueden otorgar a una VM:

```
$ virsh domcapabilities | grep -i max
    <vcpu max='255'/>
```

En este host se pueden definir un máximo de 255 vcpus para una VM. También podemos obtener los tipos de dispositivos que se pueden emplear:

```
$ virsh domcapabilities | grep diskDevice -A 5

    <enum name='diskDevice'>
        <value>disk</value>
        <value>cdrom</value>
        <value>floppy</value>
        <value>lun</value>
    </enum>
```

Note: `virsh` (**o virtualization shell**) es una herramienta de interfaz de línea de comandos para la administración de guest y el hypervisor. Emplea la API de administración `libvirt` y posee muchas funcionalidades que pueden clasificarse en:

- Guest management commands (for example `start`, `stop`)
- Guest monitoring commands (for example `memstat`, `cpustat`)
- Host and hypervisors commands (for example `capabilities`, `nodeinfo`)
- Virtual networking commands (for example `net-list`, `net-define`)
- Storage management commands (for example `pool-list`, `pool-define`)
- Snapshot commands (`create-snapshot-as`)

Mayor referencia en:

- KVM-virsh - Ubuntu Help
- Managing guests using virsh - RedHat Virtualization Guide
- virsh - libvirt guide

27.2 Creando VMs con `virt-manager`

Table of Contents

El programa **virt-manager (Virtual Machine Manager)** es una interfaz de usuario de escritorio para administrar VMs a través de `libvirt`. Se enfoca principalmente en VMs de KVM pero también es compatible con Xen y LXC.

A través de un wizard, `virt-manager` nos permite crear, monitorear y configurar VMs. Se presenta una consola gráfica del guest domain mediante un VNC embebido y un cliente SPICE viewer.

A continuación se tratarán los 4 métodos de creación de una VM guest disponibles a través de `virt-manager` usando su wizard integrado.

27.2.1 Creación de VM con método Local installation media

Este método de instalación requiere que el medio de instalación sea insertado en la bandeja de CD-ROM del sistema físico, se encuentre disponible localmente en formato ISO o disponible a través de la red. Podemos guardar los archivos ISO es `/var/lib/libvirt/images`, que sirve como el pool de almacenamiento por defecto de `virt-manager`.

1. Ejecutemos Virtual Machine Manager desde un terminal con el comando `virt-manager` o buscando la aplicación de escritorio (*Alt + F2* o *Alt + Fn + F2*).

Por defecto nos encontraremos con una ventana en la que ya se encuentra configurada una conexión a un hypervisor local QEMU/KVM.

2. Desde **Virtual Machine Manager** clic en el botón *Create a new virtual machine* en la barra de herramientas o seleccionar *File | New Virtual Machine* para abrir el wizard. Esto nos permitirá crear una nueva VM desde `virt-manager`.

Los pasos que se siguen en el wizard para la creación de una VM son:

1. Escoger el medio de instalación
2. Configurar el medio de instalación
3. Configurar la memoria y CPU
4. Configurar el almacenamiento de la VM
5. Nombrar el SO guest y configurar networking.

3. En el primer paso seleccionar *Local installation media (ISO o CDROM)* como método de instalación del SO guest:
4. Especificar la ubicación de la imagen ISO que emplearemos escribiendo la ruta manualemente o con el boton *Browse*. En el caso de que estemos usando un CDROM o DVD físico seleccionar la opción *Use CDROM or DVD*:

Note: `virt-manager` detecta el SO basado en el medio de instalación automáticamente. `libosinfo-bin` provee una base de datos con más de 300 SOs (Windows, Linux, . . .).

1. Instalar `libosinfo-bin`: `sudo apt-get install libosinfo-bin`
2. Con el comando `osinfo-query os` obtenemos detalled de varios SOs:

Short ID	Name	Version	ID
ubuntu18.04	Ubuntu 18.04 LTS	18.04	http://ubuntu.com/ubuntu/18.04
centos7.0	CentOS 7.0	7.0	http://centos.org/centos/7.0
alpinelinux3.8	Alpine Linux 3.8	3.8	http://alpinelinux.org/alpinelinux/3.8
cirros0.4.0	CirrOS 0.4.0	0.4.0	http://cirros-cloud.net/cirros/0.4.0

---

5. Especificar la memoria y el CPU que deseemos asignar a la VM:

El wizard muestra la cantidad máxima de CPUs y memoria que podemos asignar.

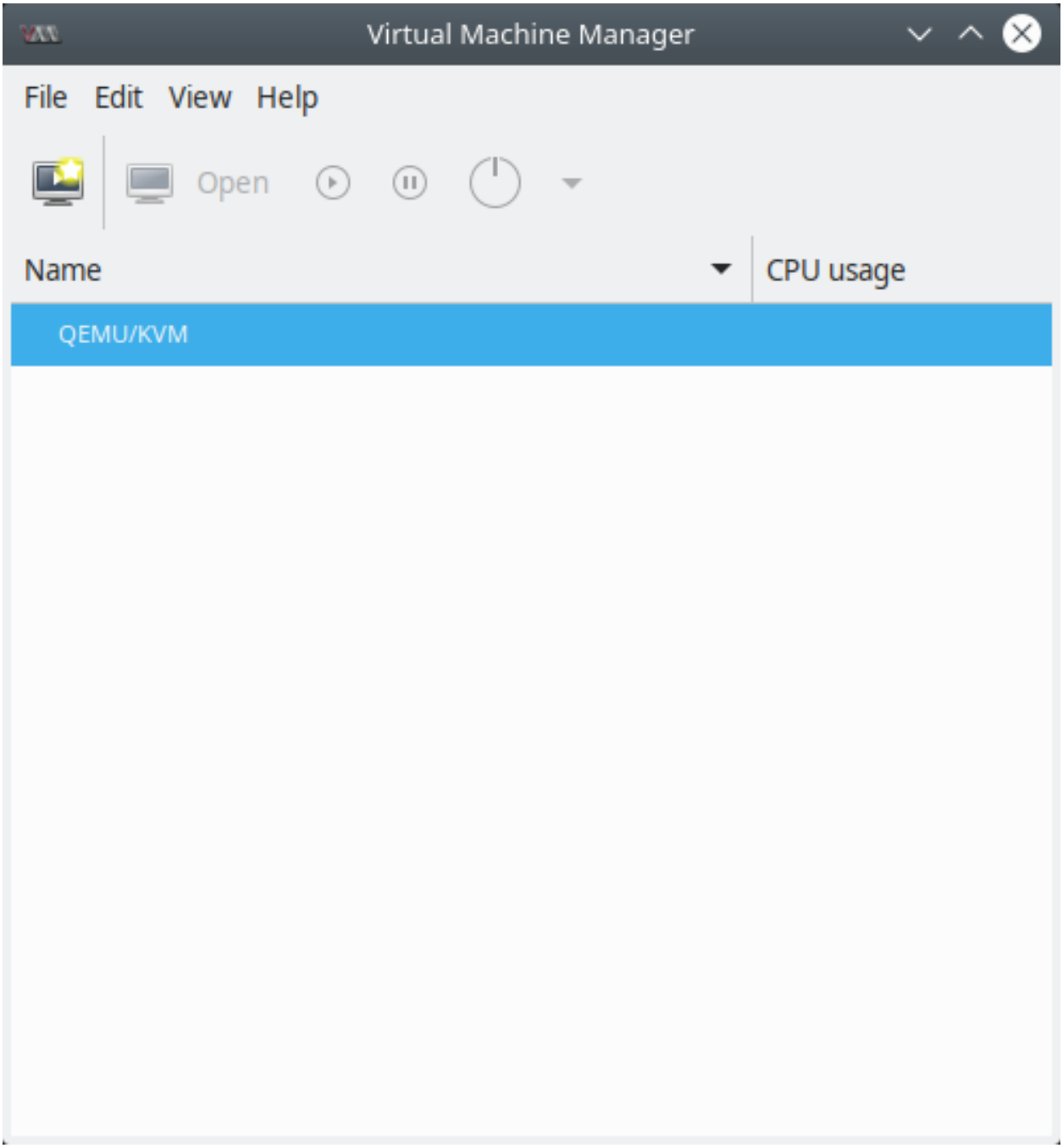


Fig. 1: virt-manager (Virtual Machine Manager)

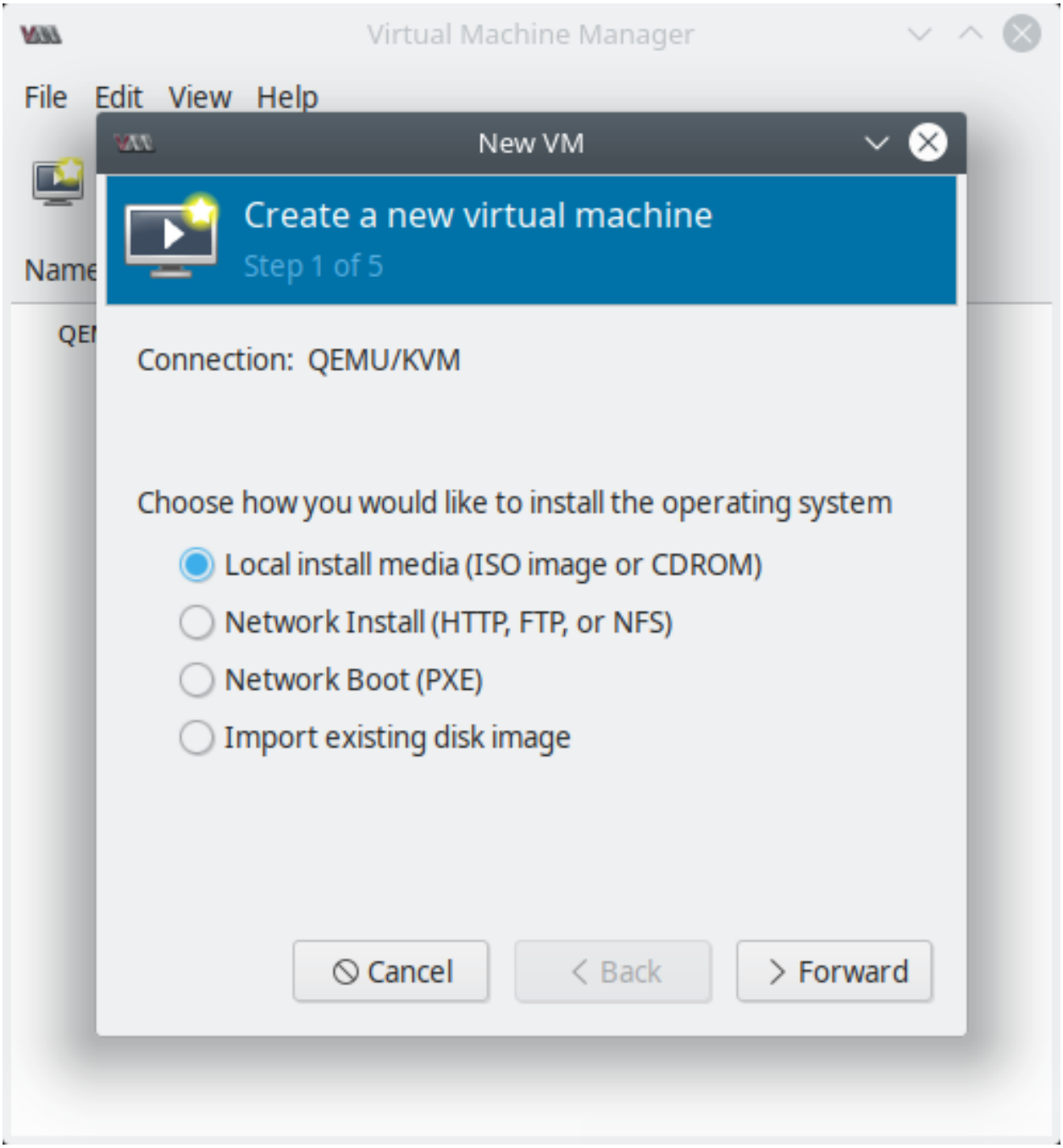


Fig. 2: virt-manager Step 1

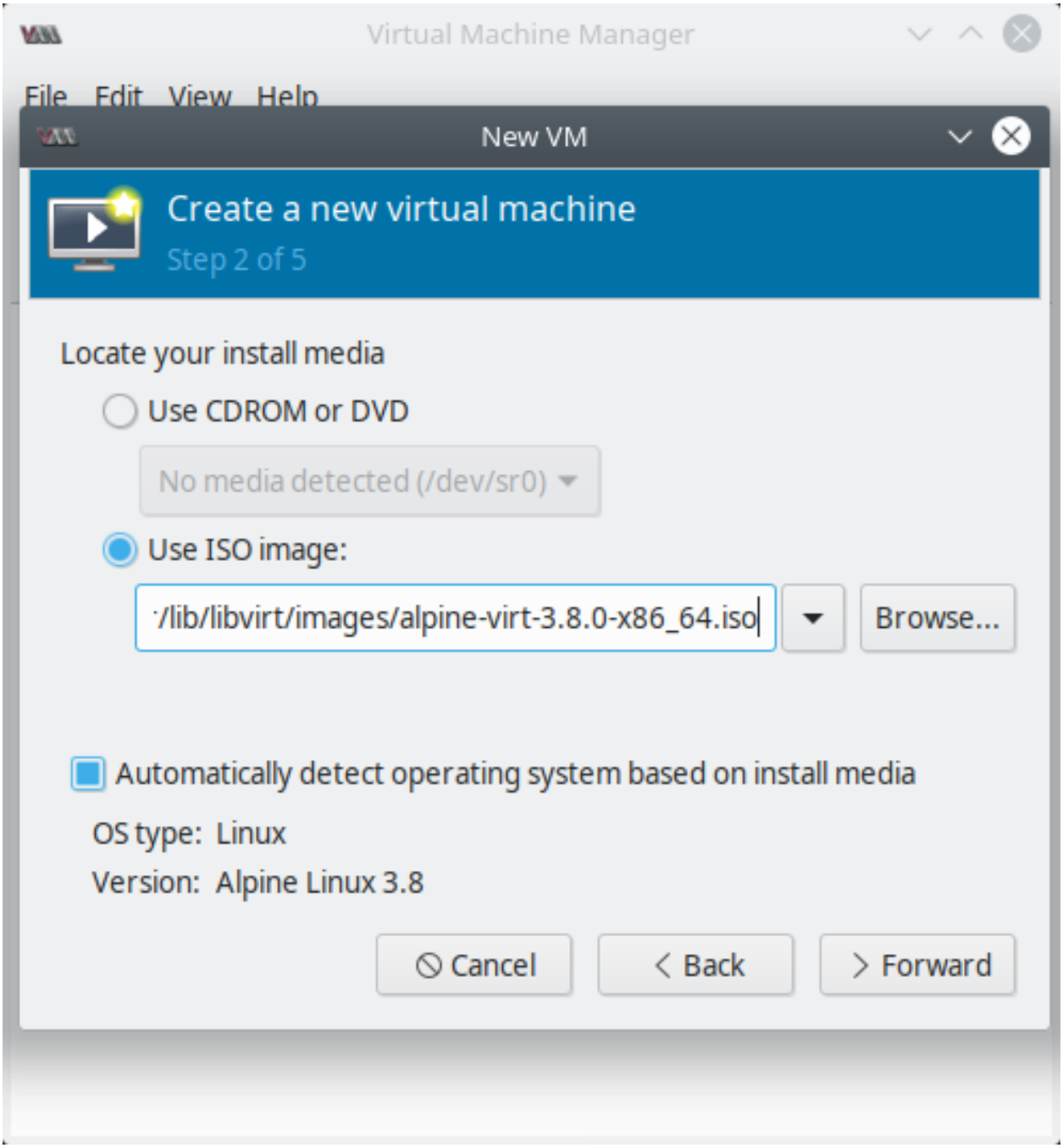


Fig. 3: virt-manager Step 2

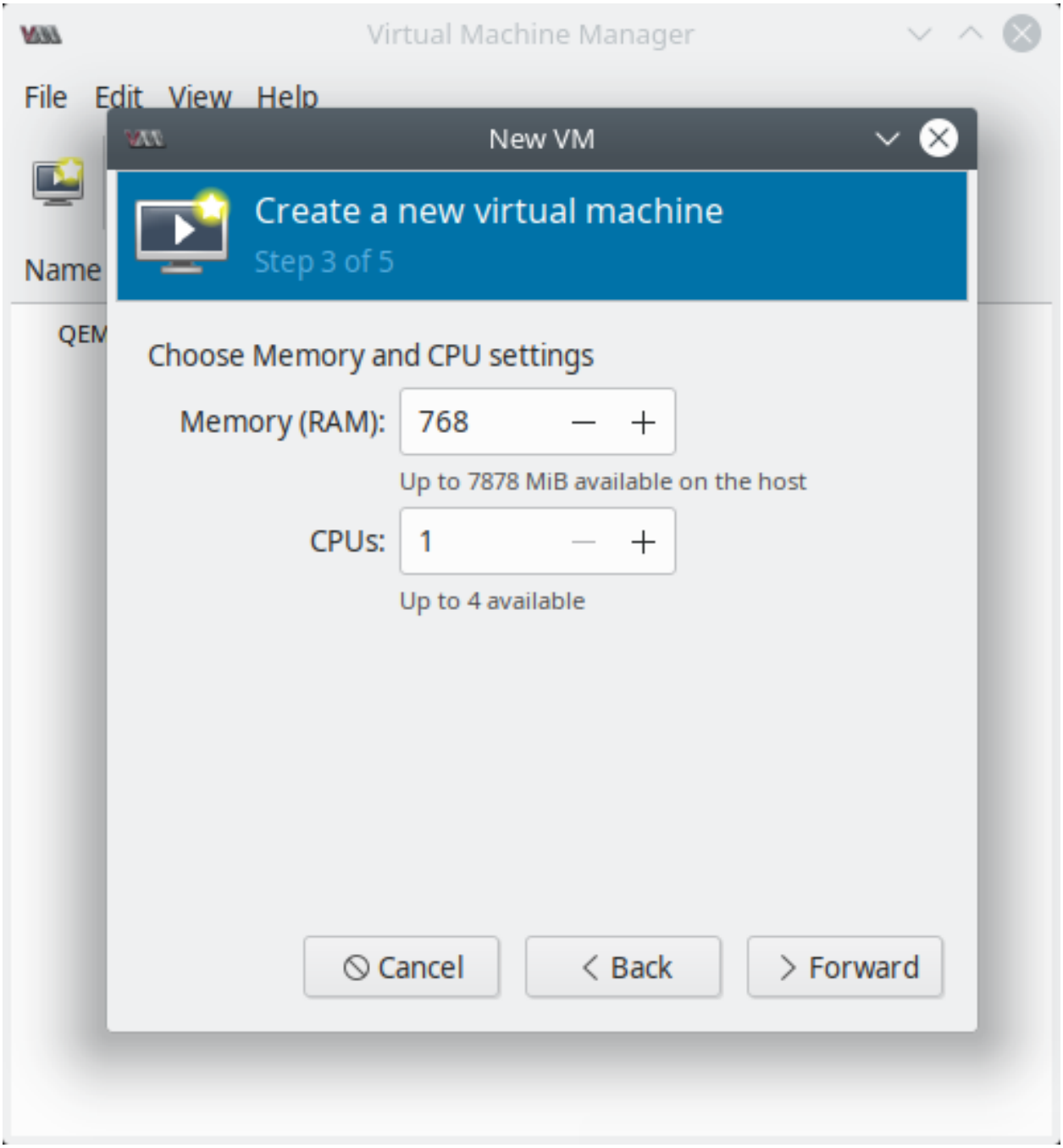


Fig. 4: virt-manager Step 3

6. Configurar el almacenamiento para la VM:

Por defecto se creará un disco virtual en la dirección `/var/lib/libvirt/qemu` dentro del pool de almacenamiento predeterminado. Si deseamos usar otro pool personalizado seleccionar la opción *Select managed or other existing storage* e ingresar la ruta del disco manualmente o seleccionar el botón *Browse* donde tendremos la posibilidad de seleccionar un volumen existente o crear uno nuevo del pool de almacenamiento definido.

Note: Mantener deseleccionada la opción *Allocate entire disk now* resultará en un **disco thin-provisioned** y seleccionarla generará un **disco thick provisioned** (también llamado **disco pre-allocated**).

---

7. El último paso es nombrar el guest y la configuración de networking. Para el nombre solo se permite letras, números, punto (.), guión bajo (_) y guión (-).

Notar que el disco virtual en este caso tiene formato `.qcow2` y se almacena en el directorio `/var/lib/libvirt/images`.

Expandir el menú *Network selection* mostrará la configuración de red virtual. Por defecto, KVM provee NAT-like bridged networking. Las VMs conectadas a esta NAT no aparecen en la red como sus propios dispositivos, pero tendrán acceso a la red a través del SO host.

Important: Si estamos pensando en correr software o un servidor web en nuestra VM y queremos que sea accesible por otros dispositivos en la red, debemos usar otra configuración de networking virtual como **Linux bridge** o **macvtap**.

Si deseamos seguir configurando el hardware de la VM, seleccionar la opción *Customize configuration before install* antes de seleccionar *Finish*. Accederemos a un wizard para agregar, remover o editar configuraciones de hardware de la VM.

8. Finalizado el proceso de creación de la VM, se creará una nueva ventana iniciando el proceso de instalación del SO guest. Además podremos ver la VM en la ventana principal de `virt-manager`:

27.2.2 Creación de VM con método Importing existing disk image

Con este método de instalación podes importar una imagen de disco previamente instalada y configurada con un SO booteable. Este método ayuda a desplegar VMs rápidamente y para transferir una VM a otro host de forma offline. Podemos usar nuestra propias imágenes de disco o importar un disco pre-configurado de alguna distribución de Linux pública:

Los pasos para crear una VM con este método son similares al proceso de *Creación de VM con método Local installation media*. Solo haremos los siguientes cambios en Step 1 y Step 2:

1. Iniciar el wizard para crear una nueva VM desde `virt-manager`, seleccionar *Import existing disk image* como método de instalación del SO.
2. Ingresar la ruta de la imagen, la cual deberá estar dentro de uno de los pools de almacenamiento:

Este método no usa el paso de asignación de tamaño de almacenamiento pues el disco ya ha sido creado previamente y nosotros solo estamos importando este archivo de disco virtual.

27.2.3 Creación de VM con método Network Boot (PXE)

El método Network Boot Preboot eXecution Environment(PXE) emplea un servidor dentro de la misma subred donde instalaremos la VM guest.

La red NAT por defecto creada por `virt-manager` no es compatible con este tipo de instalación, ya que, una VM conectada a NAT no aparece en la red como su propio dispositivo; y por tanto, el servidor PXE no puede verlo. Para

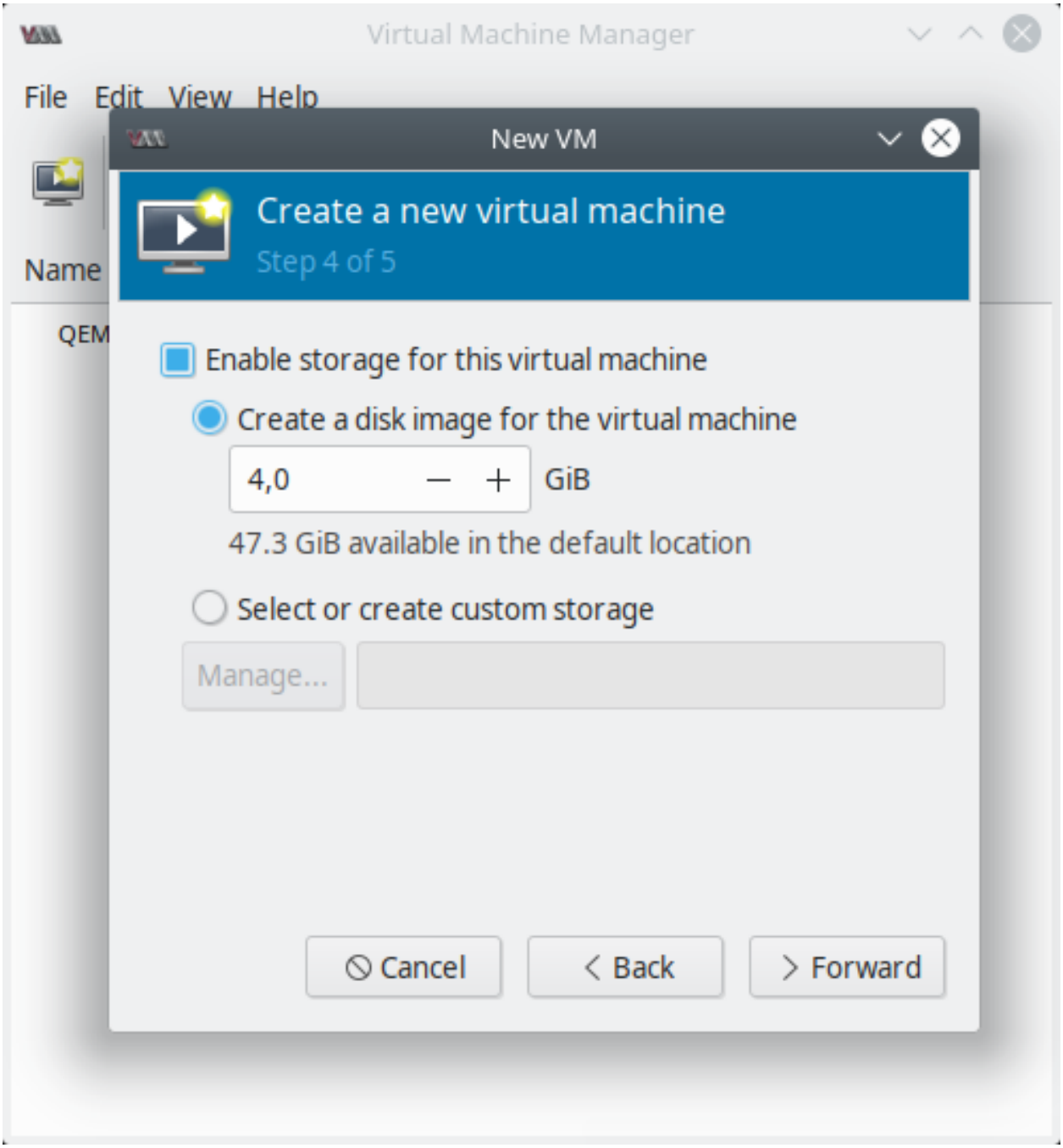


Fig. 5: virt-manager Step 4

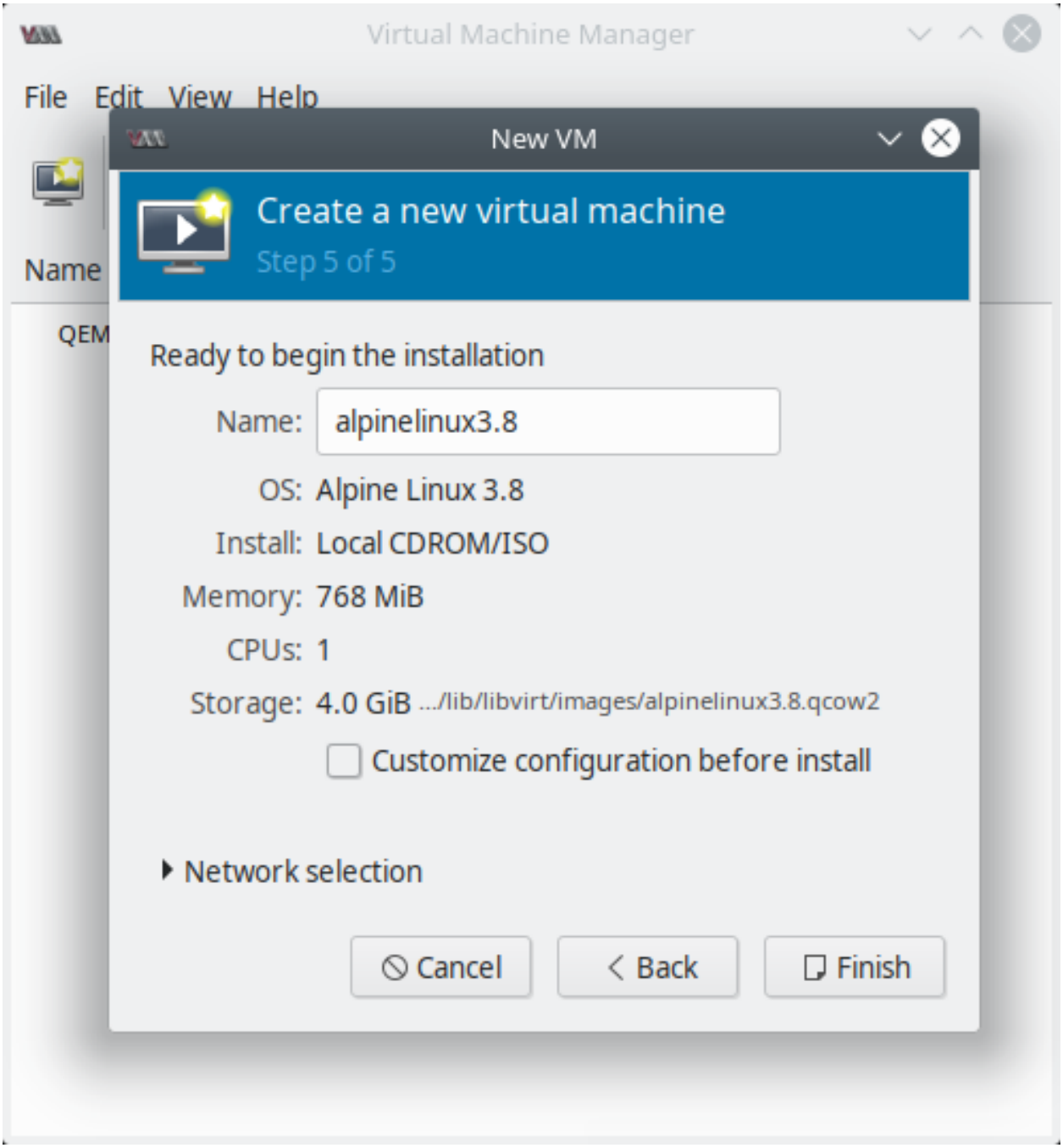


Fig. 6: virt-manager Step 5

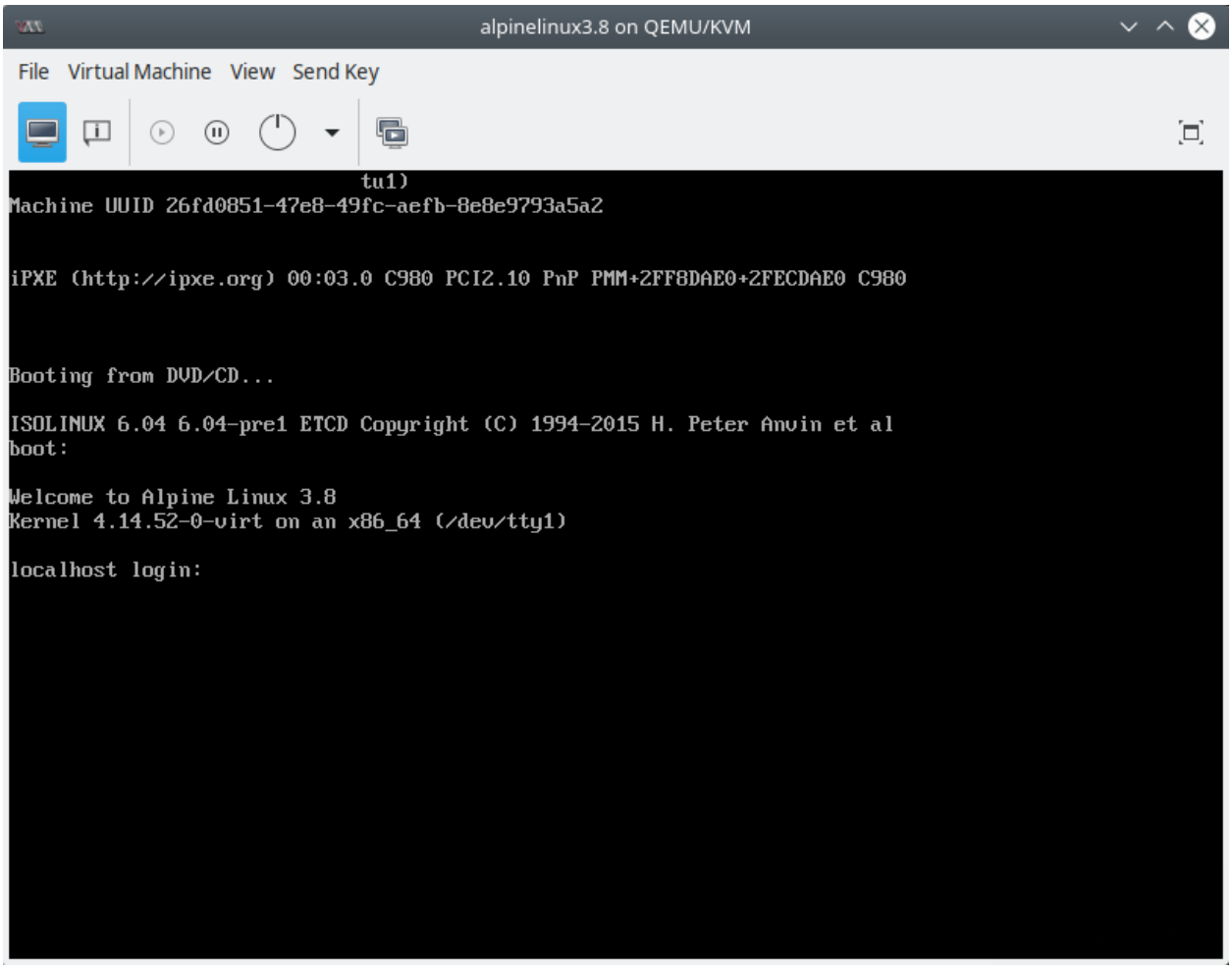


Fig. 7: virt-manager VM running

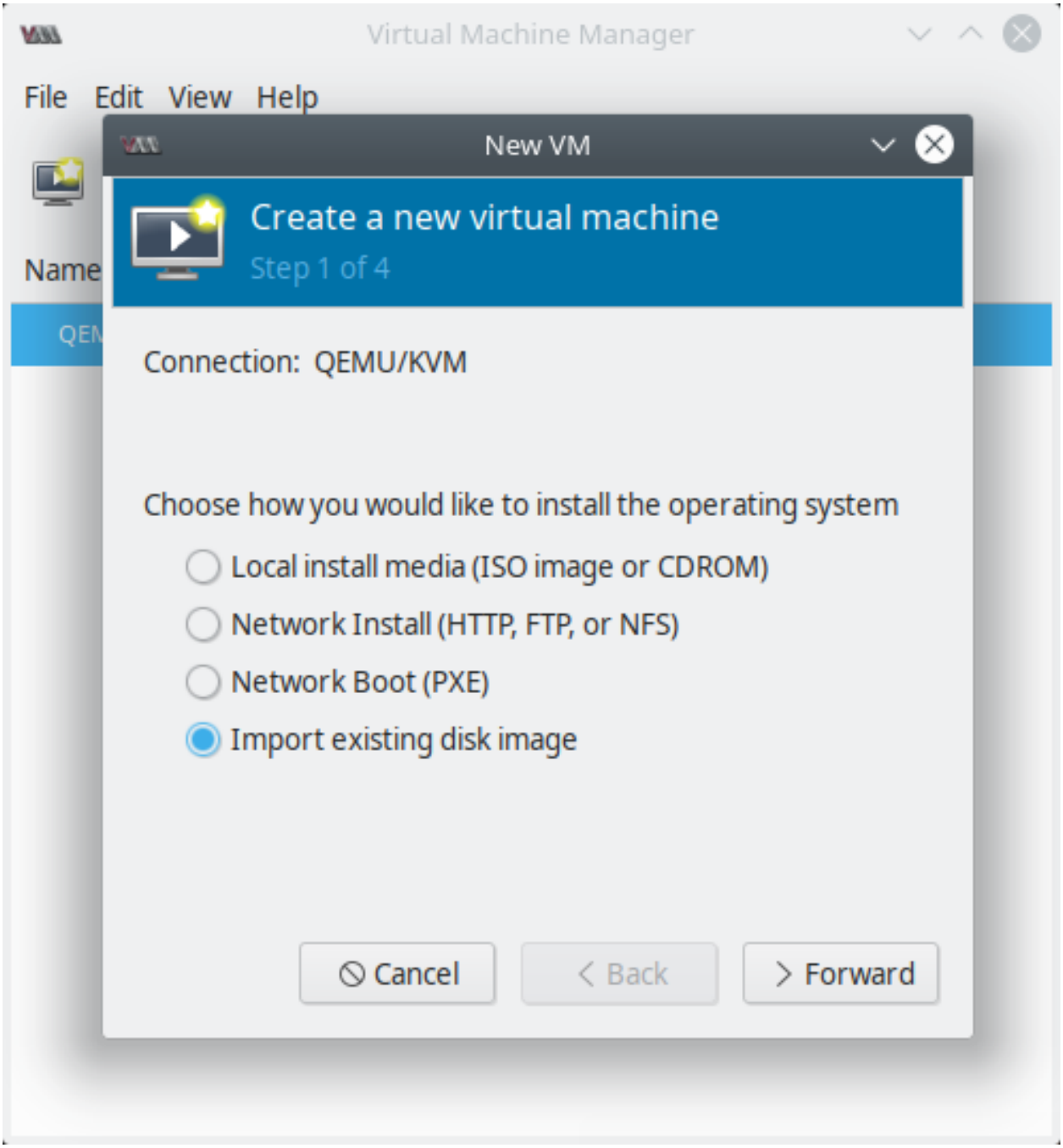


Fig. 8: virt-manager - método Importing existing disk image

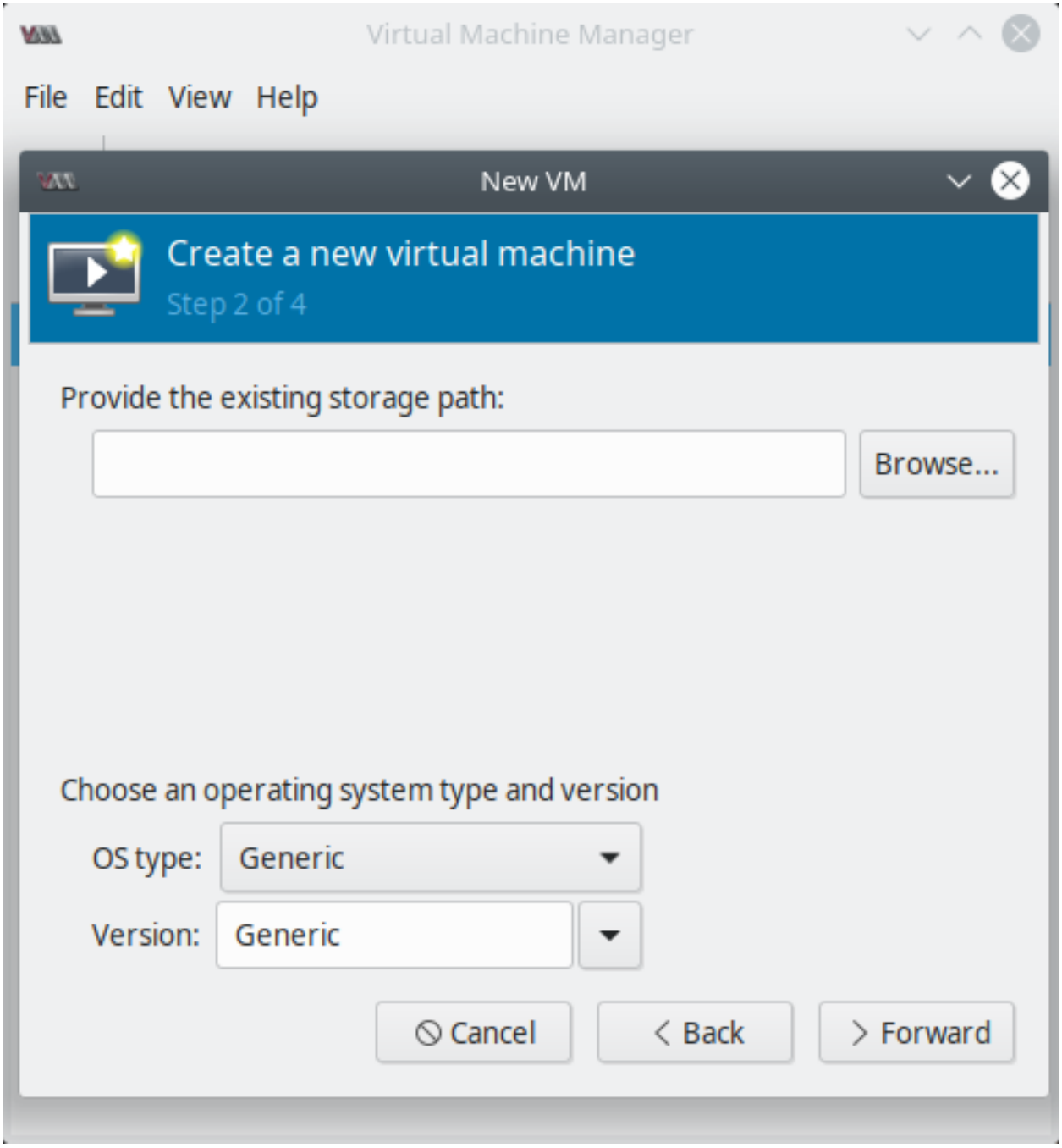


Fig. 9: virt-manager - Step 2

usar una instalación de un guest OS mediante PXE, necesitamos usar o bien un **bridge de red de software** o una **red basada en macvtap** en el sistema host. En este ejemplo usaremos la macvtap.

Los pasos para crear una VM con este método son similares al proceso de *Creación de VM con método Local installation media*. Solo haremos los siguientes cambios en Step 1 y Step 5:

1. Iniciar el wizard para crear una nueva VM desde `virt-manager`, seleccionar *Network Boot (PXE)* como método de instalación del SO.
2. En el paso 5 seleccionar expandir el menú *Network selection*, usar *Host device <interface>:macvtap* de la lista desplegable y seleccionar *Source mode* en *Bridge*. Clic en *Finish*.

27.2.4 Creación de VM con método Network Install (HTTP, FTP o NFS)

El método de instalación por red consiste en pasar una URL con un mirror del SO que deseemos instalar. Esta dirección web contendrá el árbol de instalación del SO. Se repiten los pasos de la sección *Creación de VM con método Local installation media*, solo haremos cambios en Step 1 y Step 5.

1. Seleccionar la opción *Network Install (HTTP, FTP o NFS)* como método de instalación del SO:
2. Opcionalmente podemos pasar **opciones de kernel**, si es que existen. Para pasar un **archivo kisckstart** como opción de kernel podremos especificarlo a través de una URL comenzando con `ks=`:

27.2.5 Referencias

- CREATING GUESTS WITH VIRT-MANAGER - Red Hat Documents
- Virtual Machine Manager Homepage

27.3 Creando VMs con `virt-install`

Table of Contents

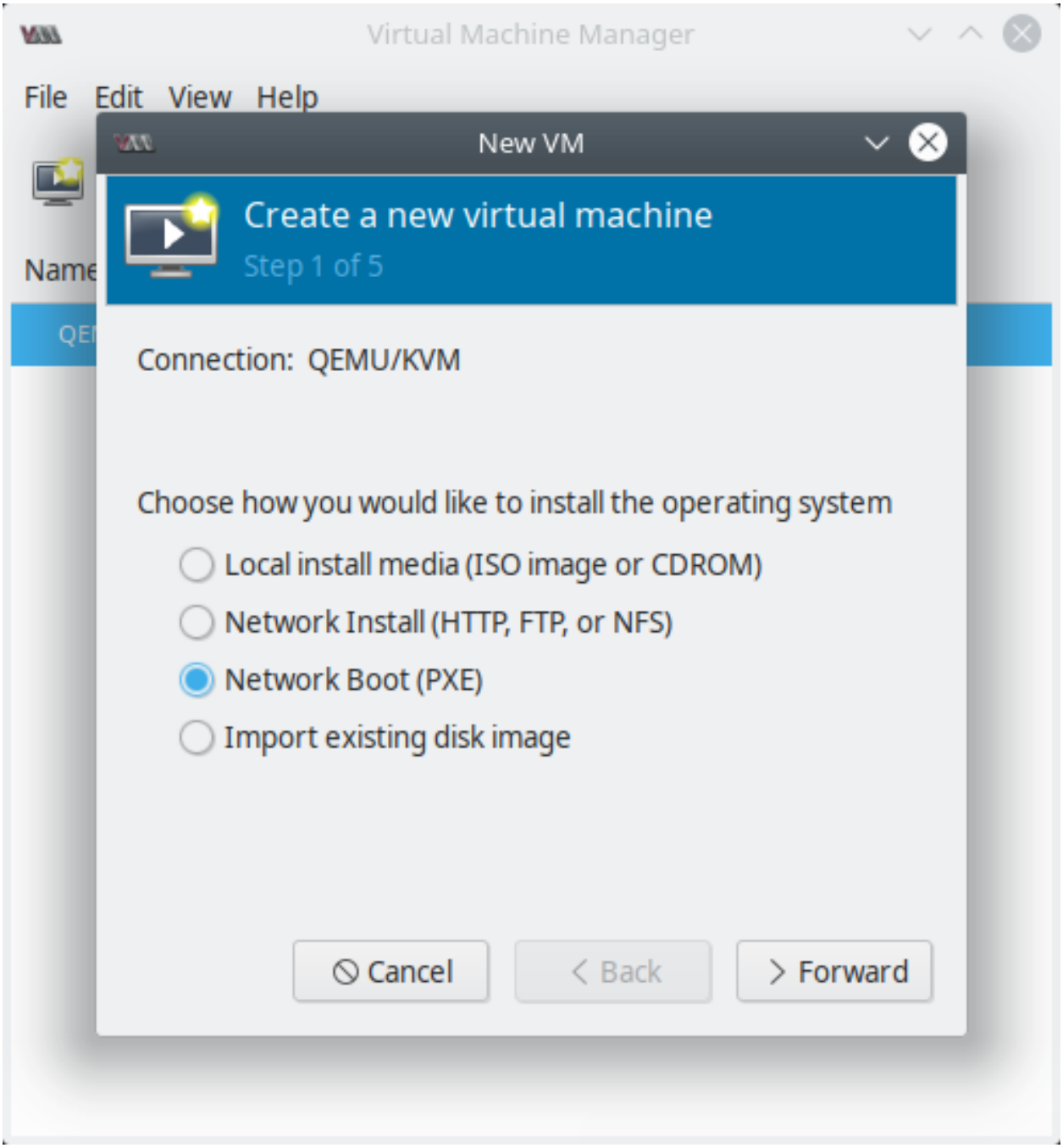


Fig. 10: virt-manager - método Network Boot (PXE)

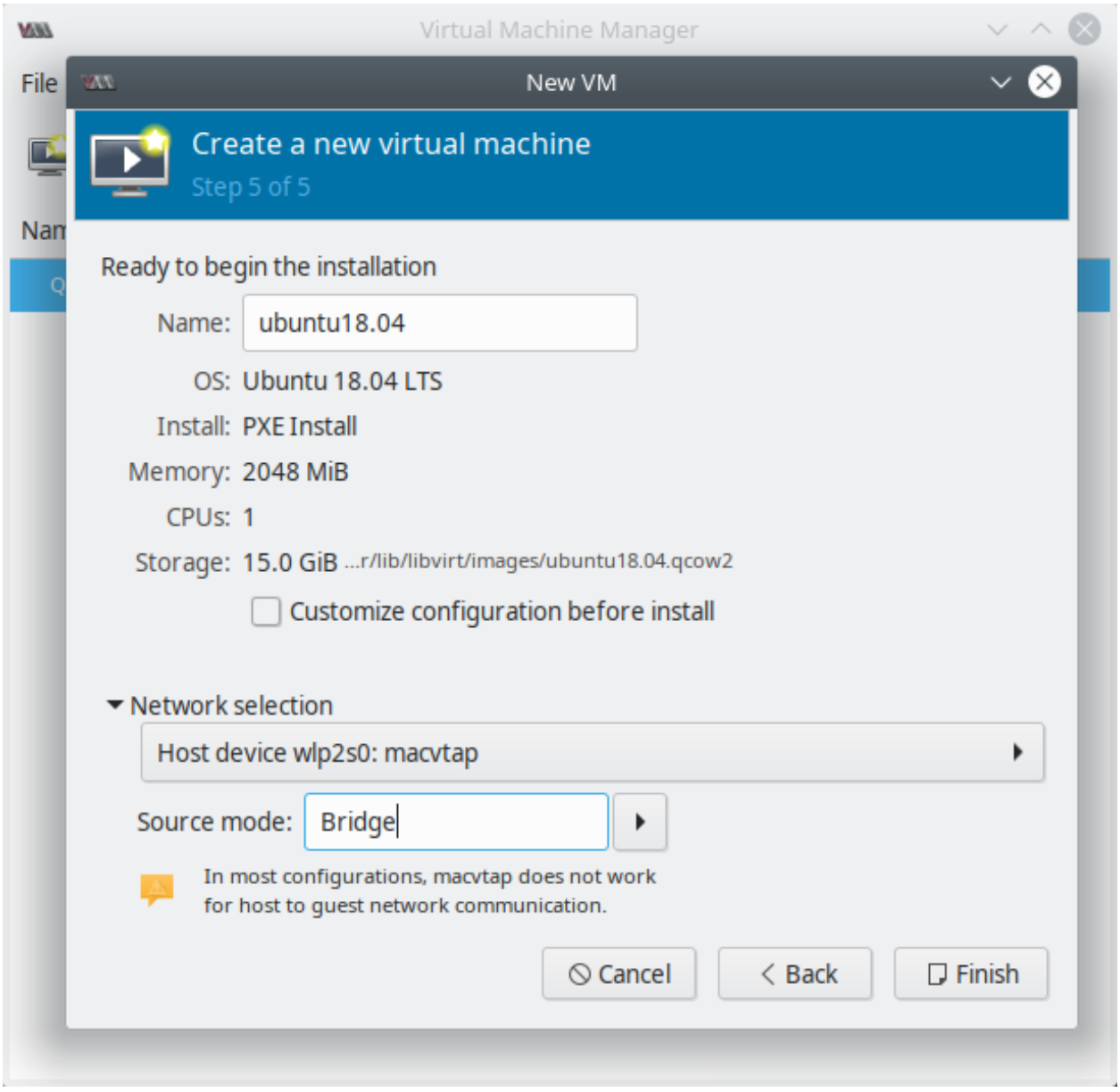


Fig. 11: virt-manager - Step 5

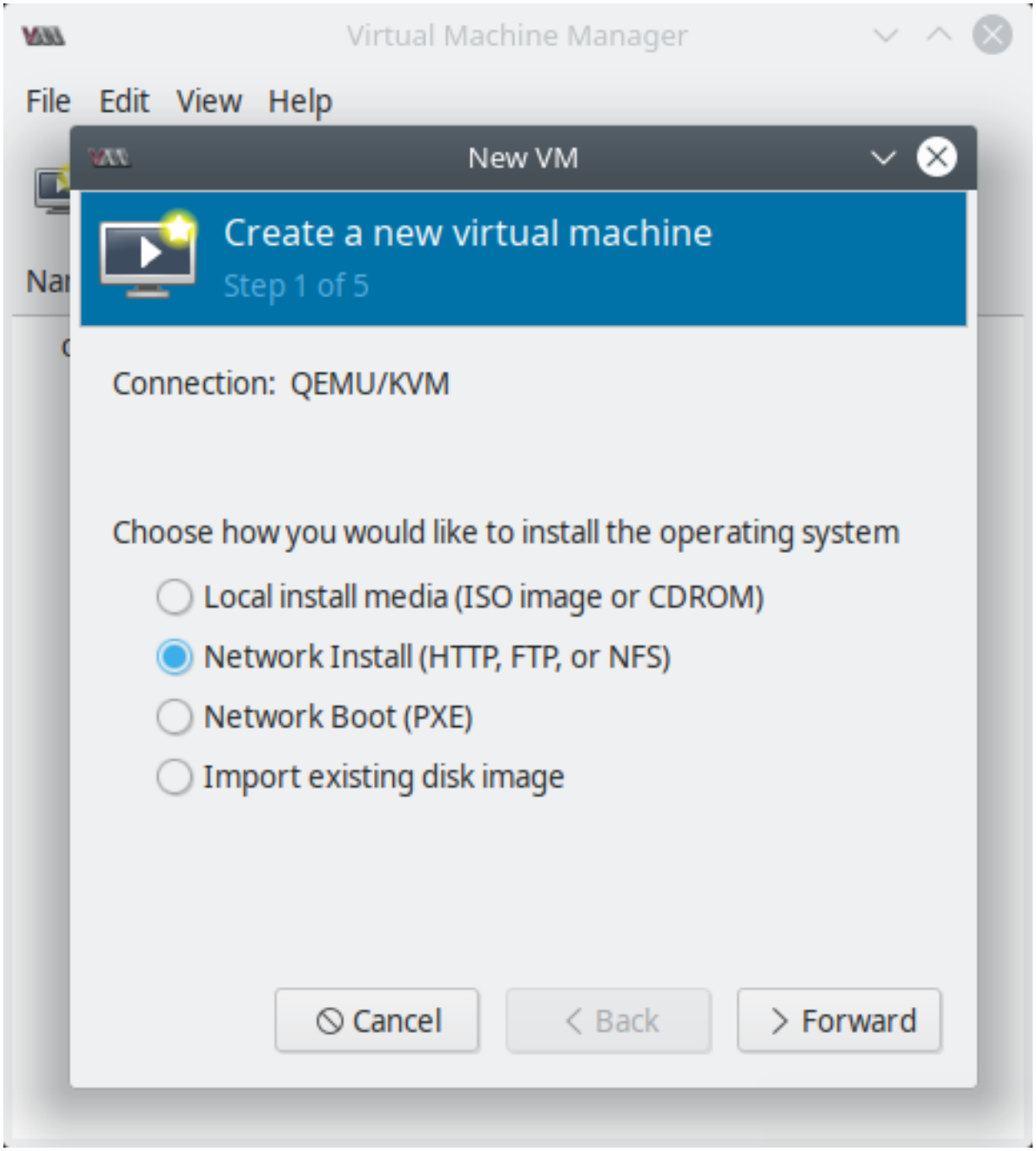


Fig. 12: virt-manager - método Network Install (HTTP, FTP o NFS)

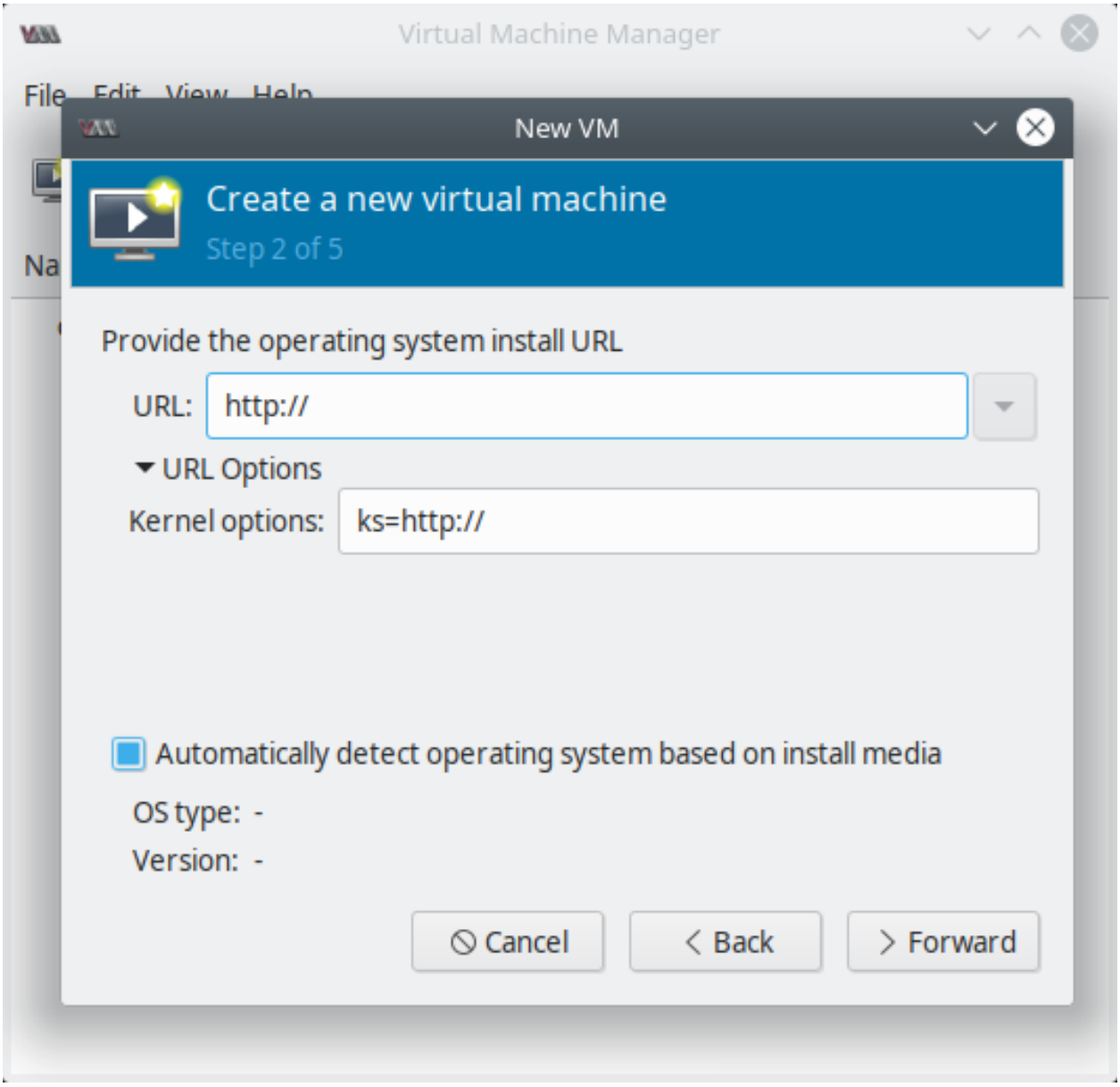


Fig. 13: virt-manager - Step 2

`virt-install` es una herramienta de línea de comandos con la cual podemos crear guest VMs e instalar un SO dentro de ellas. Puede usarse dentro de un script para automatizar la creación de VMs o trabajar con la herramienta interactivamente. Además, podemos iniciar la instalación de SOs en las VMs de manera *unattended* a través de archivos **kickstart**.

`virt-install` abrirá una ventana `virt-viewer` una vez que se haya creado la VM para acceder a la misma.

Note: Se requiere de privilegio root para ejecutar algunos parámetros con `virt-install`.

27.3.1 Opciones principales del comando `virt-install`

Ver la guía de ayuda de `virt-install`:

```
$ virt-install --help
```

Ver una lista completa de opciones para un argumento:

```
# virt-install --option=?
$ virt-install --disk=?

    --disk options:
        address.base
        address.bus
        address.controller
    # ...
```

Opciones generales

- `--name`: nombre de la VM.
- `--memory`: cantidad de RAM para asignar al guest (en MiB).

Almacenamiento guest

Usar una de las siguientes 2 opciones:

- `--disk`: configuración del almacenamiento de la VM. (`--disk none` crea la VM sin espacio de disco).
- `--filesystem`: ruta del filesystem para la VM guest.

Método de instalación

Usar uno de los siguientes métodos:

- `--location`: ubicación del medio de instalación.

- `--cdrom`: archivo o dispositivo usado como cd-rom virtual. Puede ser la ruta a una imagen ISO o a una URL de donde extraer la imagen ISO. Pero no puede ser un dispositivo CD-ROM físico del host.
- `--pxe`: usar el protocolo de booteo PXE para cargar el initial ramdisk y el kernel al iniciar la instlación del guest.
- `--import`: no realiza la instalación del SO y construye el guest a partir de una imagen de disco existente. El inicio usado para el booteo sera el primer dispositivo pasado por la opción `disk` o `filesystem`.
- `--boot`: configuración de arranque de la VM luego de la instalación. Esta opción permite especificar un orden de arranque de dispositivos, arrancar siempre del kernel e initrd con argumentos opcionales de kernel y habilitar un menú de arranque de la BIOS.

27.3.2 Ejemplos de creación de VMs

Usando una imagen ISO

Crear una VM desde una imagen ISO:

```
$ sudo virt-install \
--name ubuntu18-desktop-guest-1 \
--memory 2048 \
--vcpus 2 \
--disk size=10 \
--cdrom ~/Downloads/ubuntu-18.04.2-live-server-amd64.iso \
--os-variant ubuntu18.04
```

El argumento `--cdrom` debe especificar la ruta a la imagen `.iso`.

Importando una imagen de disco

Crear una VM de una imagen de disco importada:

```
$ sudo virt-install \
--name centos7-guest-1 \
--memory 1024 \
--vcpus 1 \
--disk ~/Downloads/centos7-min0.qcow2 \
--import \
--os-variant centos7.0
```

El argumento `--disk` debe especificar la ruta a la imagen de la VM (en este caso es formato `.qcow2`).

Importando una imagen de red

Crear una VM de una imagen de disco importada de internet:

```
$ sudo virt-install \
--name ubuntu18-guest-2 \
--memory 2048 \
--vcpus 2 \
--disk size=8 \
--location http://archive.ubuntu.com/ubuntu/dists/bionic/main/installer-i386/ \
--os-variant ubuntu18.04
```

El argumento `--location` define la ruta del árbol de instalación del SO.

Usando un servidor PXE

Dos parámetros obligatorios que deben ingresarse para hacer una instalación con el protocolo de booteo PXE son `--pxe` y `--network` especificando una red bridged.

Podemos crear una VM usando PXE con el siguiente comando:

```
$ sudo virt-install \
--name ubuntu18-guest-3 \
--memory 2048 \
--vcpus 2 \
--disk size=8 \
--network=bridge:br0 \
--pxe \
--os-variant ubuntu18.04
```

Usando un archivo kickstart

Para SOs como Red Hat Enterprise Linux o CentOS podemos pasar un archivo kickstart para automatizar la instalación del SO al arrancar la VM:

```
$ sudo virt-install \
--name guest1-rhel7 \
--memory 2048 \
--vcpus 2 \
--disk size=8 \
--location http://example.com/path/to/os \
--os-variant rhel7 \
--initrd-inject /path/to/ks.cfg \
--extra-args="ks=file:/ks.cfg console=tty0 console=ttyS0,115200n8"
```

Los últimos dos parámetros `--initrd-inject` y `--extra-args` declaran que la VM usará un archivo kickstart para instalar el SO.

27.3.3 Configuración de red de la VM

Cuando creamos una VM podemos especificar la red para la VM. A continuación se lista una serie de tipos de red que pueden ser usados para los guests:

Red default con NAT

La red default usa un switch de red virtual NAT (network address translation) de `libvirtd`. El paquete `libvirt-daemon-config-network` debe estar instalado antes de crear una VM con la red default.

Para configurar una VM con red NAT usar el siguiente parámetro en el comando `virt-install`:

```
--network default \
```

Note: Cuando no se especifica un valor del parámetro `--network`, por defecto se crea la VM con la red default con NAT.

Red bridged con DHCP

Podemos configurar una red bridged con un servidor DHCP externo. Esta configuración es útil para casos en los que el host tiene una configuración de red estátoca y el guest requiere conectividad entrante y saliente con la LAN. También es útil para migraciones live de una VM.

Para establecer una red bridged con DHCP para la VM guest, debemos tener creado el bridge previamente y agregar el parámetro:

```
# --network <bridge> \
--network br0 \
```

Red bridged con dirección IP estática

Una red bridged también puede usarse para configurar una dirección IP estática en el guest. Para usar una red bridged con dirección IP estática en la VM usar:

```
--network br0 \
--extra-args "ip=192.168.1.2::192.168.1.1:255.255.255.0:test.example.com:eth0:none"
```

Sin red

Para que nuestra VM no tenga una interfaz de red agregar:

```
--network=none \
```

27.3.4 Referencias

- Using qemu-img - RedHat Documents
- Network Address Translation (NAT) with libvirt - RedHat Documents
- Bridged Networking with Virtual Machine Manager - RedHat Documents
- Bridged Networking with libvirt - RedHat Documents

27.4 Creando VMs con `qemu-system-x86_64`

Table of Contents

`qemu-system-<architecture>`, por ejemplo `/usr/local/bin/qemu-system-x86_64`, sirve para correr un sistema de esa arquitectura en la máquina host.

Podemos usar `qemu-system-x86_64` para correr una máquina virtual y con el parámetro `-enable-kvm` podemos habilitar el soporte de KVM full virtualization.

27.4.1 Ejemplos de sintáxis

```
$ sudo qemu-system-x86_64 [-enable-kvm] -name [cirrOS-X] -hda [cirros-0.4.0-x86_64-
↪disk.img] -smp [1] -m [64] -netdev tap,id=[mynet0],ifname=[tap0] -device␣
↪[e1000|virtio-net-pci],netdev=[mynet0],mac=[52:55:00:d1:55:01]
```

Ejemplo:

```
sudo qemu-system-x86_64 -enable-kvm -name vm1 -hda cirros-0.4.0-x86_64-disk.img &
```

27.4.2 Referencias

- Using bare qemu-kvm vs libvirt virt-install virt-manager
- Difference between qemu-kvm qemu-system-x86_64 qemu-x86_64

27.5 Despliegue automatizado de VMs con `virt-builder`

Table of Contents

`virt-builder` es una herramienta para un rápido despliegue de nuevas VMs. Además nos permite personalizar estas VMs a través de plantillas de SOs editables, ahorrándonos el tiempo de hacer una instalación del SOs guest desde cero.

Esta herramienta es provista por el paquete `libguestfs-tools-c` y puede ser instalado corriendo el comando:

- Con `apt`: `sudo apt-get install -y libguestfs-tools`
- Con `yum`: `sudo yum install -y libguestfs-tools-c`

Note: Por defecto, `virt-builder` descarga plantillas de SOs del repositorio `http://libguestfs.org/download/builder/`, siendo necesario conexión a Internet. Sin embargo, también es posible crear un repositorio local para `virt-builder`.

Por ejemplo para crear una VM de Centos 7.0 con 10 GB ejecutaremos:

```
$ cd /var/lib/libvirt/images
$ sudo virt-builder centos-7.0 --format raw --size 10G

[ 1.0] Downloading: http://libguestfs.org/download/builder/centos-7.0.xz
[ 2.0] Planning how to build this image
[ 2.0] Uncompressing
[ 14.0] Resizing (using virt-resize) to expand the disk to 10.0G
[ 149.0] Opening the new disk
[ 179.0] Setting a random seed
[ 180.0] Setting passwords
virt-builder: Setting random password of root to Ldm43dKj12Msalp1x
[ 198.0] Finishing off
    Output file: centos-7.0.img
```

(continues on next page)

(continued from previous page)

```
    Output size: 10.0G
    Output format: raw
    Total usable space: 8.1G
    Free space: 7.3G
```

Primero descargó la plantilla, la descomprimió, redimensionó la imagen del disco para que calce al tamaño dado, sembró datos de la plantilla a la imagen, la edito (configuró una contraseña aleatoria) y finalizó. Se ha creado una contraseña de root aleatoria y usa un espacio de disco mínimo expandible hasta 10 GB. La imagen se almacena en el directorio `/var/lib/libvirt/images`.

Luego, ingresar el siguiente comando para crear la VM con `virt-install`:

```
$ sudo virt-install --name centos --ram 1028 --vcpus=2 --disk path=/var/lib/libvirt/
↪images/centos-7.0.img --import
```

Hay muchas opciones disponibles con `virt-builder`: instalación de software, configuración de hostname, edición de archivos, etc.

`virt-builder` almacena en caché la plantilla descargada en el directorio home del usuario actual con la siguiente ruta: `$HOME/.cache/virt-builder`. Podemos imprimir la información del directorio caché, incluyendo que guest están en caché corriendo:

```
$ virt-builder --print-cache

cache directory: /root/.cache/virt-builder
centos-6                 x86_64      no
centos-7.0               x86_64      cached
centos-7.1               x86_64      no
cirros-0.3.1             x86_64      no
debian-6                 x86_64      no
# ...
```

CentOS 7 se encuentra en caché. a siguiente vez que creemos una VM con este SO usará la plantilla en caché y creará la VM aún más rápido.

- Eliminar caché: `virt-builder --delete-cache`
- Descargar todas las plantillas a caché local: `virt-builder --cache-all-templates`

Note: virt-builder solo soporta guest de Linux, no posee soporte para guest de Windows.

Referencias:

- virt-builder - Ubuntu manpages
- virt-builder - Fedora

27.6 Despliegue automatizado de VMs con oz

Table of Contents

27.6.1 Introducción a oz

oz es un conjunto de programas útiles para crear VMs **JEOS (Just Enough Operating System)** con una mínima entrada del guest. La entrada mínima por parte del usuario será: nombre del guest y detalles de SO como arquitectura, URL para el OS tree o ISO y contraseña root. Todos estos atributos pueden definirse en un archivo XML simple llamado TDL (Template Definition Language).

La plantilla TDL empleada contendrá los siguientes parámetros:

- La ISO o URI en la que la imagen estará basada
- Tamaño de disco
- Los paquetes extra a instalar
- Los comandos a ejecutar luego de que la imagen es creada
- Los archivos a ser introducidos luego de que la imagen es creada

En resumen, el archivo TDL describe al SO que el usuario quiere instalar, de dónde obtener el medio de instalación y cualquier paquete adicional o acción que el usuario desee se realice en el SO. Ver ejemplos de TDL y usos.

Con este grupo de herramientas es posible instalar una variedad de SOs. A bajo nivel usa componentes de infraestructura de virtualización conocidos como qemu/kvm, libvirt y libguestfs. Para la instalación de SOs se emplean un conjunto de archivos predefinidos **kickstart** para sistemas basados en Red Hat, archivos **preseed** para sistemas basados en Debian y archivos **XML** para instalaciones de Windows automatizadas.

Las imágenes del SO compatible con oz debe estar diseñado para una arquitectura de CPU **i386** o **x86_64**. Y puede ser Debian, Fedora, FreeBSD, Mandrake, OpenSUSE, RHEL, CentOS, Ubuntu y Windows.

```
$ oz-install -h

Currently supported architectures are:
    i386, x86_64
Currently supported operating systems are:
    Debian: 5, 6, 7, 8
    Fedora Core: 1, 2, 3, 4, 5, 6
    Fedora: 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25,␣
↪26
    FreeBSD: 10, 11
    Mageia: 2, 3, 4, 4.1, 5
    Mandrake: 8.2, 9.0, 9.1, 9.2, 10.0, 10.1
    Mandriva: 2005, 2006.0, 2007.0, 2008.0
    OpenSUSE: 10.3, 11.0, 11.1, 11.2, 11.3, 11.4, 12.1, 12.2, 12.3, 13.1, 13.2, 42.1,␣
↪42.2
    RHEL 2.1: GOLD, U2, U3, U4, U5, U6
    RHEL 7: Beta, 0, 1, 2
    RHEL/CentOS 3: GOLD, U1, U2, U3, U4, U5, U6, U7, U8, U9
    RHEL/CentOS/Scientific Linux 4: GOLD, U1, U2, U3, U4, U5, U6, U7, U8, U9
    RHEL/OL/CentOS/Scientific Linux{,CERN} 5: GOLD, U1, U2, U3, U4, U5, U6, U7, U8,␣
↪U9, U10, U11
```

(continues on next page)

(continued from previous page)

```
    RHEL/OL/CentOS/Scientific Linux{,CERN} 6: 0, 1, 2, 3, 4, 5, 6, 7, 8
    RHL: 7.0, 7.1, 7.2, 7.3, 8, 9
    Ubuntu: 5.04, 5.10, 6.06[.1,.2], 6.10, 7.04, 7.10, 8.04[.1,.2,.3,.4], 8.10, 9.04,␣
↪9.10, 10.04[.1,.2,.3], 10.10, 11.04, 11.10, 12.04[.1,.2,.3,.4,.5], 12.10, 13.04, 13.
↪10, 14.04[.1,.2,.3,.4,.5], 14.10, 15.04, 15.10, 16.04[.1], 16.10, 17.04
    Windows: 2000, XP, 2003, 7, 2008, 2012, 8, 8.1, 2016, 10
```

Para comenzar a usar `oz` con el fin de crear VMs y desplegarlas de forma automatizada sigamos los siguientes pasos:

27.6.2 Usando `oz`

1. Instalar `oz`:

- Con `apt`: `sudo apt-get install oz`
- Con `yum`: `sudo yum install -y oz`

2. Obtener un archivo ISO (local, HTTP, FTP) con el medio de instalación del SO deseado o una URL con el tree OS para obtener el SO a través de Internet.

3. Crear nuestro archivo TDL (formato `.tdl`) con la descripción personalizada del SO que vamos a instalar:

```
<template>
<name>centos7</name>
<os>
    <name>CentOS-7</name>
    <version>7</version>
    <arch>x86_64</arch>
    <install type='iso'>
        <iso>http://mirror.unimagdalena.edu.co/centos/7.7.1908/isos/x86_64/CentOS-7-
↪x86_64-Minimal-1908.iso</iso>
    </install>
</os>
<description>CentOS 7 x86_64 template</description>
</template>
```

- `/template/name` es un nombre definido por el usuario. Puede ser cualquiera pero debe ser único entre todos los TDLs que deseemos construir.
- `/template/os/name`, `/template/os/version` y `/template/os/arch` es el nombre, versión y arquitectura del SO que deseamos instalar, respectivamente.
- `/template/os/install` le dice a `oz` de dónde obtener el medio de instalación. En este caso usamos un archivo `iso`, entonces necesitamos un elemento ISO en el archivo XML apuntando al medio de instalación ISO.
- `/template/description` es un parámetro opcional que describe esta plantilla

4. Una vez hayamos guardado nuestro archivo `.tdl` ejecutaremos el sigueinte comando:

```
# sudo oz-install /path/to/tdl-file.tdl
$ sudo oz-install -u -d3 ~/Downloads/my-centos7-template.tdl
```

Sintáxis:

- `-u`: Customize the image after installation. This generally installs additional packages onto the disk image after installation.
- **`-d`: loglevel. Turn on debugging output to level loglevel. Log levels:**

  - 0 - errors only (this is the default)
  - 1 - errors and warnings
  - 2 - errors, warnings, and information
  - 3 - all messages
  - 4 - all messages, prepended with the level and classname

Si hemos corrido `oz-install` con el modo de debugueo (`-d`) y con valor `3` veremos todo el log del proceso (*Log de oz-install*):

- Primero descargará el archivo `.iso` que hemos especificado en la URL. La dirección de descarga por defecto será `/var/lib/oz/isos/`.
- Luego comenzará a escanear los archivos que tiene el instalador
- Comenzará el proceso de instalación. Ahora podremos ver el porcentaje de avance.
- Luego se genera imagen de disco bajo la ruta /var/lib/libvirt/image con formato `.dsk`.
- También se habrá creado el archivo XML que puede usarse para iniciar el guest inmediatamente.

5. El archivo XML creado estará en el mismo directorio donde está nuestra plantilla TDL. Usar el archivo XML para definir el guest (`virsh define`) y crear la VM (`virsh start`):

```
# virsh define <XML-file>
$ virsh define centos7Jan_09_2020-18\:19\:42

Domain centos7 defined from centos7Jan_09_2020-18:19:42

# virsh start <domain>
$ virsh start centos7
```

6. Abrir una consola de nuestra VM con `virt-viewer` o `virt-manager`:

```
$ virt-viewer centos7
```

27.6.3 Archivo de configuración de `oz`

El archivo `/etc/oz/oz.cfg` es el archivo de configuración de `oz` para la creación de VMs. El contenido por defecto es el siguiente:

```
[paths]
output_dir = /var/lib/libvirt/images
data_dir = /var/lib/oz
screenshot_dir = /var/lib/oz/screenshots
# sshprivkey = /etc/oz/id_rsa-icicle-gen

[libvirt]
uri = qemu:///system
image_type = raw
# type = kvm
# bridge_name = virbr0
# cpus = 1
# memory = 1024

[cache]
original_media = yes
```

(continues on next page)

(continued from previous page)

```
modified_media = no
jeos = no

[icicle]
safe_generation = no

[timeouts]
install = 1200
inactivity = 300
boot = 300
shutdown = 90
```

Como vemos, aquí podemos establecer varios parámetros de configuración para la creación de VMs. Algunos importantes son:

- `output_dir`: ruta de almancenamiento de las imágenes posterior a su creación. (`/var/lib/libvirt/images` por defecto).
- `bridge_name`: bridge al cual se conectará la VM (`vibr0` por defecto).
- `cpus`: cantidad de CPUs asignados a la VM
- `memoria`: memoria RAM asignada a la VM

El resto de directivas las podemos encontrar en oz-customize.

27.6.4 Referencias

- TDL (Template Definition Language)
- ejemplos de TDL y usos
- Download CentOS Linux ISO images
- VM template creation with oz-install (with preseed)
- Using OZ tool to create virtual guests (with minimal input)
- oz-customize

27.6.5 Log de oz-install

```
$ sudo oz-install -u -d3 ~/Downloads/my-centos7-template.tdl

Libvirt network without a forward element, skipping
libvirt bridge name is virbr0
Libvirt type is kvm
Name: centos7, UUID: 50507349-dbd5-4fc5-8c54-7d6e6ea42de6
MAC: 52:54:00:89:09:e6, distro: CentOS-7
update: 7, arch: x86_64, diskimage: /var/lib/libvirt/images/centos7.dsk
nicmodel: virtio, clockoffset: utc
mousetype: ps2, disk_bus: virtio, disk_dev: vda
icicletmp: /var/lib/oz/icicletmp/centos7, listen_port: 44993
console_listen_port: 36931
Original ISO path: /var/lib/oz/isos/CentOS-77x86_64-iso.iso
Modified ISO cache: /var/lib/oz/isos/CentOS-77x86_64-iso-oz.iso
Output ISO path: /var/lib/libvirt/images/centos7-iso-oz.iso
```

(continues on next page)

(continued from previous page)

```
ISO content path: /var/lib/oz/isocontent/centos7-iso
Checking for guest conflicts with centos7
Generating install media
Fetching the original media
Starting new HTTP connection (1): mirror.unimagdalena.edu.co:80
http://mirror.unimagdalena.edu.co:80 "POST /centos/7.7.1908/isos/x86_64/CentOS-7-x86_
↪64-Minimal-1908.iso HTTP/1.1" 200 987758592
Fetching the original install media from http://mirror.unimagdalena.edu.co/centos/7.7.
↪1908/isos/x86_64/CentOS-7-x86_64-Minimal-1908.iso
Starting new HTTP connection (1): mirror.unimagdalena.edu.co:80
http://mirror.unimagdalena.edu.co:80 "GET /centos/7.7.1908/isos/x86_64/CentOS-7-x86_
↪64-Minimal-1908.iso HTTP/1.1" 200 987758592
10240kB of 964608kB
20480kB of 964608kB
#..............
962560kB of 964608kB
964608kB of 964608kB
Copying ISO contents for modification
Setting up guestfs handle for centos7
Adding ISO image /var/lib/oz/isos/CentOS-77x86_64-iso.iso
Launching guestfs
Mounting ISO
Checking if there is enough space on the filesystem
Extracting ISO contents
Putting the kickstart in place
Modifying isolinux.cfg
Generating new ISO
I: -input-charset not specified, using utf-8 (detected in locale settings)

genisoimage 1.1.11 (Linux)
Scanning /var/lib/oz/isocontent/centos7-iso
Scanning /var/lib/oz/isocontent/centos7-iso/Packages
Excluded: /var/lib/oz/isocontent/centos7-iso/Packages/TRANS.TBL
Excluded: /var/lib/oz/isocontent/centos7-iso/TRANS.TBL
Scanning /var/lib/oz/isocontent/centos7-iso/EFI
Excluded: /var/lib/oz/isocontent/centos7-iso/EFI/TRANS.TBL
Scanning /var/lib/oz/isocontent/centos7-iso/EFI/BOOT
Excluded: /var/lib/oz/isocontent/centos7-iso/EFI/BOOT/TRANS.TBL
Scanning /var/lib/oz/isocontent/centos7-iso/EFI/BOOT/fonts
Excluded: /var/lib/oz/isocontent/centos7-iso/EFI/BOOT/fonts/TRANS.TBL
Scanning /var/lib/oz/isocontent/centos7-iso/isolinux
Excluded: /var/lib/oz/isocontent/centos7-iso/isolinux/TRANS.TBL
Excluded by match: /var/lib/oz/isocontent/centos7-iso/isolinux/boot.cat
Scanning /var/lib/oz/isocontent/centos7-iso/repodata
Excluded: /var/lib/oz/isocontent/centos7-iso/repodata/TRANS.TBL
Scanning /var/lib/oz/isocontent/centos7-iso/LiveOS
Excluded: /var/lib/oz/isocontent/centos7-iso/LiveOS/TRANS.TBL
Scanning /var/lib/oz/isocontent/centos7-iso/images
Excluded: /var/lib/oz/isocontent/centos7-iso/images/TRANS.TBL
Scanning /var/lib/oz/isocontent/centos7-iso/images/pxeboot
Excluded: /var/lib/oz/isocontent/centos7-iso/images/pxeboot/TRANS.TBL
Using RPM_G000.;1 for  /RPM-GPG-KEY-CentOS-Testing-7 (RPM-GPG-KEY-CentOS-7)
Using PYTHO000.RPM;1 for  /var/lib/oz/isocontent/centos7-iso/Packages/python-pyudev-0.
↪15-9.el7.noarch.rpm (python-pycurl-7.19.0-19.el7.x86_64.rpm)
Using LIBGL000.RPM;1 for  /var/lib/oz/isocontent/centos7-iso/Packages/libglvnd-egl-1.
↪0.1-0.8.git5baa1e5.el7.x86_64.rpm (libglvnd-glx-1.0.1-0.8.git5baa1e5.el7.x86_64.rpm)
Using NETWO000.RPM;1 for  /var/lib/oz/isocontent/centos7-iso/Packages/NetworkManager-
↪1.18.0-5.el7.x86_64.rpm (NetworkManager-team-1.18.0-5.el7.x86_64.rpm)
```

(continues on next page)

(continued from previous page)

```
#.............
Using PCIUT000.RPM;1 for  /var/lib/oz/isocontent/centos7-iso/Packages/pciutils-3.5.1-
↪3.el7.x86_64.rpm (pciutils-libs-3.5.1-3.el7.x86_64.rpm)
Using CRYPT000.RPM;1 for  /var/lib/oz/isocontent/centos7-iso/Packages/cryptsetup-libs-
↪2.0.3-5.el7.x86_64.rpm (cryptsetup-2.0.3-5.el7.x86_64.rpm)
Using KERNE000.RPM;1 for  /var/lib/oz/isocontent/centos7-iso/Packages/kernel-tools-3.
↪10.0-1062.el7.x86_64.rpm (kernel-tools-libs-3.10.0-1062.el7.x86_64.rpm)
Using E2FSP000.RPM;1 for  /var/lib/oz/isocontent/centos7-iso/Packages/e2fsprogs-1.42.
↪9-16.el7.x86_64.rpm (e2fsprogs-libs-1.42.9-16.el7.x86_64.rpm)
Writing:   Initial Padblock                        Start Block 0
Done with: Initial Padblock                        Block(s)    16
Writing:   Primary Volume Descriptor               Start Block 16
Done with: Primary Volume Descriptor               Block(s)    1
Writing:   Eltorito Volume Descriptor              Start Block 17
Size of boot image is 4 sectors -> No emulation
Done with: Eltorito Volume Descriptor              Block(s)    1
Writing:   Joliet Volume Descriptor                Start Block 18
Done with: Joliet Volume Descriptor                Block(s)    1
Writing:   End Volume Descriptor                   Start Block 19
Done with: End Volume Descriptor                   Block(s)    1
Writing:   Version block                           Start Block 20
Done with: Version block                           Block(s)    1
Writing:   Path table                              Start Block 21
Done with: Path table                              Block(s)    4
Writing:   Joliet path table                       Start Block 25
Done with: Joliet path table                       Block(s)    4
Writing:   Directory tree                          Start Block 29
Done with: Directory tree                          Block(s)    48
Writing:   Joliet directory tree                   Start Block 77
Done with: Joliet directory tree                   Block(s)    33
Writing:   Directory tree cleanup                  Start Block 110
Done with: Directory tree cleanup                  Block(s)    0
Writing:   Extension record                        Start Block 110
Done with: Extension record                        Block(s)    1
Writing:   The File(s)                             Start Block 111
0.98% done, estimate finish Thu Jan  9 18:11:57 2020
1.95% done, estimate finish Thu Jan  9 18:11:57 2020
#.............
98.63% done, estimate finish Thu Jan  9 18:12:00 2020
99.60% done, estimate finish Thu Jan  9 18:12:00 2020
Total translation table size: 124656
Total rockridge attributes bytes: 55185
Total directory bytes: 92160
Path table size(bytes): 140
Done with: The File(s)                             Block(s)    511779
Writing:   Ending Padblock                         Start Block 511890
Done with: Ending Padblock                         Block(s)    150
Max brk space used ac000
512040 extents written (1000 MB)


Cleaning up old ISO data
Generating 10GB diskimage for centos7
Waiting for volume to be created, 90/90
Running install for centos7
Generate XML for guest centos7 with bootdev cdrom
Generated XML:
```

(continues on next page)

(continued from previous page)

```
<domain type="kvm">
<name>centos7</name>
<memory>1048576</memory>
<currentMemory>1048576</currentMemory>
<uuid>50507349-dbd5-4fc5-8c54-7d6e6ea42de6</uuid>
<clock offset="utc"/>
<vcpu>1</vcpu>
<features>
    <acpi/>
    <apic/>
    <pae/>
</features>
<os>
    <type>hvm</type>
    <boot dev="cdrom"/>
</os>
<on_poweroff>destroy</on_poweroff>
<on_reboot>destroy</on_reboot>
<on_crash>destroy</on_crash>
<devices>
    <graphics type="vnc" port="-1"/>
    <interface type="bridge">
    <source bridge="virbr0"/>
    <mac address="52:54:00:89:09:e6"/>
    <model type="virtio"/>
    </interface>
    <input bus="ps2" type="mouse"/>
    <serial type="pty">
    <target port="0"/>
    </serial>
    <serial type="tcp">
    <source host="127.0.0.1" mode="bind" service="44993"/>
    <protocol type="raw"/>
    <target port="1"/>
    </serial>
    <disk device="disk" type="file">
    <target bus="virtio" dev="vda"/>
    <source file="/var/lib/libvirt/images/centos7.dsk"/>
    <driver type="raw" name="qemu"/>
    </disk>
    <disk device="cdrom" type="file">
    <source file="/var/lib/libvirt/images/centos7-iso-oz.iso"/>
    <target dev="hdc"/>
    </disk>
</devices>
</domain>

Waiting for centos7 to finish installing, 1200/1200
Waiting for centos7 to finish installing, 1190/1200
#..............
Waiting for centos7 to finish installing, 750/1200
Waiting for centos7 to finish installing, 740/1200
Waiting for centos7 to finish shutdown, 90/90
Install of centos7 succeeded
Generate XML for guest centos7 with bootdev hd
Generated XML:
<domain type="kvm">
```

(continues on next page)

(continued from previous page)

```
<name>centos7</name>
<memory>1048576</memory>
<currentMemory>1048576</currentMemory>
<uuid>50507349-dbd5-4fc5-8c54-7d6e6ea42de6</uuid>
<clock offset="utc"/>
<vcpu>1</vcpu>
<features>
    <acpi/>
    <apic/>
    <pae/>
</features>
<os>
    <type>hvm</type>
    <boot dev="hd"/>
</os>
<on_poweroff>destroy</on_poweroff>
<on_reboot>destroy</on_reboot>
<on_crash>destroy</on_crash>
<devices>
    <graphics type="vnc" port="-1"/>
    <interface type="bridge">
    <source bridge="virbr0"/>
    <mac address="52:54:00:89:09:e6"/>
    <model type="virtio"/>
    </interface>
    <input bus="ps2" type="mouse"/>
    <serial type="pty">
    <target port="0"/>
    </serial>
    <serial type="tcp">
    <source host="127.0.0.1" mode="bind" service="44993"/>
    <protocol type="raw"/>
    <target port="1"/>
    </serial>
    <disk device="disk" type="file">
    <target bus="virtio" dev="vda"/>
    <source file="/var/lib/libvirt/images/centos7.dsk"/>
    <driver type="raw" name="qemu"/>
    </disk>
</devices>
</domain>

Cleaning up after install
Customizing image
No additional packages, files, or commands to install, and icicle generation not␣
↪requested, skipping customization
Libvirt XML was written to centos7Jan_09_2020-18:19:42
```

27.7 Habilitando `virsh console`

La funcionalidad `virsh console` permite conectarnos desde la consola del host a la consola de la VM guest.

Al principio, cuando intentemos conectarnos al terminal del guest no tendremos repuesta:

```
$ virsh console ubuntu18
Connected to domain ubuntu18Escape character is ^]
...
```

Para habilitar la consola del guest seguir los siguientes pasos:

1. Desde la VM guest ejecutar:

```
sudo systemctl enable serial-getty@ttyS0.service

sudo systemctl start serial-getty@ttyS0.service
```

Note: Hasta aquí podría bastar para acceder a la consola con `virsh console`.

2. En la VM, reemplazar las siguientes línea del archivo `/etc/default/grub`:

```
GRUB_CMDLINE_LINUX_DEFAULT="quiet"
#GRUB_TERMINAL=console
```

Por las siguientes líneas:

```
GRUB_CMDLINE_LINUX_DEFAULT="console=tty0 console=ttyS0"
GRUB_TERMINAL="serial console"
```

3. En la VM, actualizar el guest con:

```
$ sudo update-grub
```

4. En el host, acceder a la consola de la VM con:

```
$ virsh console ubuntu18
```

O iniciar la VM con la consola adjunta:

```
$ virsh start ubuntu18 --console
```

Referencias:

- Solución - virsh VM console does not show any output

CHAPTER 28

Network Virtualization

28.1 Linux Bridges

Table of Contents

* *Conectando una interfaz veth al bridge*

28.1.1 Iniciando con Linux Bridges

Un **virtual network switch** o **bridge** es equivalente a un switch físico con la diferencia de que un **Linux bridge** posee un número ilimitado de puertos virtuales. Podemos conectar VMs a estos puertos virtuales. Del mismo modo que un switch físico, el bridge aprende direcciones MAC de paquetes recibidos y los guarda en una **MAC table**, la cual usa para tomar decisiones de forwarding de tramas.

Instalando herramientas y cargando módulos

1. Para administrar bridges necesitamos una herramienta llamada `brctl` que se obtiene instalando el paquete `bridge-utils`. Para instalarlo ejecutaremos:

- Con `apt`: `sudo apt-get install -y bridge-utils`
- Con `yum`: `sudo yum install -y bridge-utils`

2. Luego debemos asegurarnos que el módulo de bridge esté cargado:

```
$ lsmod | grep bridge

bridge                155648  0
```

Si el módulo no estuviese cargado usar `modprobe bridge` para cargarlo al kernel.

Creando un Linux Bridge

1. Para crear un bridge llamado `brtest` usamos el comando `brctl`:

```
$ sudo brctl addbr brtest
```

2. Para listar los bridges disponibles con información como el ID del bridge, estado del Spanning Tree Protocol (STP), y las interfaces conectadas al bridge, ejecutamos:

```
$ brctl show

bridge name     bridge id               STP enabled     interfaces
brtest          8000.000000000000       no
```

3. Un Linux bridge también se mostrará como un dispositivo de red. Para ver los detalles de red del bridge creado usaremos el comando `ip` o `ifconfig`:

```
$ ip link show brtest

18: brtest: <BROADCAST,MULTICAST> mtu 1500 qdisc noop state DOWN mode DEFAULT group␣
↪default qlen 1000
    link/ether 66:43:4a:a2:9f:3e brd ff:ff:ff:ff:ff:ff

$ ifconfig brtest

brtest: flags=4098<BROADCAST,MULTICAST>  mtu 1500
        ether 66:43:4a:a2:9f:3e  txqueuelen 1000  (Ethernet)
        RX packets 0  bytes 0 (0.0 B)
```

(continues on next page)

(continued from previous page)

```
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 0  bytes 0 (0.0 B)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
```

El Linux bridge está listo. Ahora creamos y conectaremos un dispositivo TAP a él.

28.1.2 Linux Bridges con Interfaces TAP

Teoría de dispositivos TAP

Podemos conectar a los puertos del bridge dispositivos de red espaciales llamados **TAP devices**. El equivalente de networking físico de un dispositivo TAP sería un **cable de red** encargado de portar los frames de Ethernet entre la VM y el bridge. Los dispositivos TAP forman parte de una tecnología mayor conocida como la implementación TUN/TAP.

Important:

- **TUN**, que representa "tunnel", simula a dispositivo de **capa de red** y trabaja con paquetes de la **capa 3** del modelo de referencia OSI, como los **paquetes IP**. TUN es usado con enrutamiento.
- **TAP** (a saber, un network tap) simula un dispositivo de **capa de enlace** y opera con paquetes de la **capa 2** del modelo de referencia OSI, como los **Ethernet frames (tramas)**. TAP es usado para crear un network bridge.

Cargando módulos TUN/TAP

Igual que se verificó con Linux Bridges, verificar que el módulo TUN/TAP esté cargando en el kernel:

```
$ lsmod | grep tap

Module  Size    Used by
tap     24576   1 vhost_net
```

Si no estuviese cargado alguno de los módulos usar `modprobe tun` y/o `modprobe tap` para cargarlo al kernel.

Creando un dispositivo TAP

Crear un dispositivo TAP llamado `tap-vnic` con el comando `ip`:

```
$ sudo ip tuntap add dev tap-vnic mode tap
```

Veamos la interfaz TAP creada con el comando `ip`:

```
$ ip link show tap-vnic

19: tap-vnic: <BROADCAST,MULTICAST> mtu 1500 qdisc noop state DOWN mode DEFAULT group␣
↪default qlen 1000
link/ether 76:0e:4c:78:0d:be brd ff:ff:ff:ff:ff:ff
```

Conectando un dispositivo TAP al bridge

Hasta el momento tenemos creado un brdige llamado `tap-vnic` y un dispositivo TAP llamado `tap-vnic`. Ahora añadamos el TAP al bridge:

```
$ sudo brctl addif brtest tap-vnic

$ brctl show

bridge name     bridge id               STP enabled     interfaces
brtest          8000.760e4c780dbe       no              tap-vnic
```

Ahora vemos a la interfaz `tap-vnic` añadida al bridge `brtest`. `tap-vnic` puede funcionar como la interfaz entre nuestra VM y el bridge `brtest`, que a su vez permite a la VM comunicarse con otras VMs añadidas a este bridge:

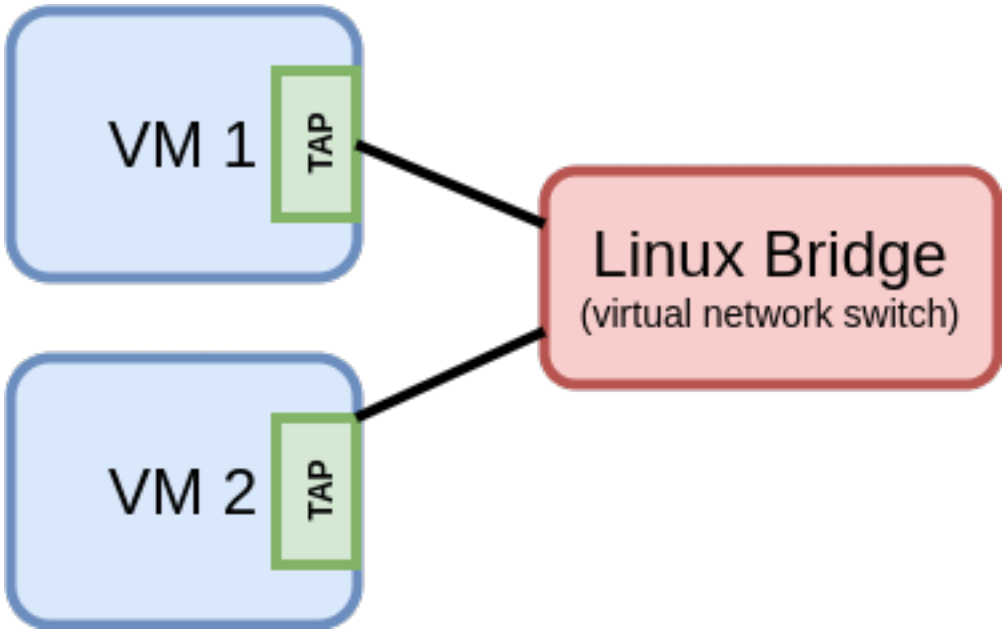


Fig. 1: Conexión entre VMs con Linux bridge e interfaces TAP

Observaremos ciertos cambios luego de haber conectado la interfaz al bridge:

- La dirección MAC del bridge luego de añadir el dispositivo TAP ha cambiado:

```
$ ip link show brtest

18: brtest: <BROADCAST,MULTICAST> mtu 1500 qdisc noop state DOWN mode DEFAULT group␣
↪default qlen 1000
    link/ether 76:0e:4c:78:0d:be brd ff:ff:ff:ff:ff:ff
```

- Es posible ver la tabla MAC del bridge `brtest` con el comando `brctl showmacs brtest`:

```
$ brctl showmacs brtest

port no     mac addr                is local?       ageing timer
1           76:0e:4c:78:0d:be       yes             0.00
1           76:0e:4c:78:0d:be       yes             0.00
```

- Además podemos agregar una dirección IP al bridge:

```
$ sudo ip addr add 192.168.99.1/24 dev brtest

$ ip addr show brtest
18: brtest: <BROADCAST,MULTICAST> mtu 1500 qdisc noop state DOWN group default qlen␣
↪1000
    link/ether 76:0e:4c:78:0d:be brd ff:ff:ff:ff:ff:ff
    inet 192.168.99.1/24 scope global brtest
    valid_lft forever preferred_lft forever
```

Deshacer configuración de Linux Bridge con interfaces TAP

- Primero desconectaremos el dispositivo TAP del bridge:

```
$ sudo brctl delif brtest tap-vnic

$ brctl show brtest

bridge name     bridge id               STP enabled     interfaces
brtest          8000.000000000000       no
```

- Luego eliminaremos el dispositivo TAP usando el comando `ip`:

```
$ sudo ip tuntap del dev tap-vnic mode tap
```

- Finalmente eliminamos el bridge:

```
$ sudo brctl delbr brtest
```

Todos estos pasos que hemos realizado son los mismos que `libvirt` lleva acabo en el backend mientras habilita o deshabilita networking para una VM.

28.1.3 Configuración adicional del Linux Bridge

- Asignar una dirección MAC a la interfaz de red del bridge con el comando `ip link`:

```
$ sudo ip link set brtest address 02:01:02:03:04:05

$ ip link show brtest

16: brtest: <BROADCAST,MULTICAST> mtu 1500 qdisc noop state DOWN mode DEFAULT group␣
↪default qlen 1000
    link/ether 02:01:02:03:04:04 brd ff:ff:ff:ff:ff:ff
```

- Activar STP en el Linux Bridge (sin STP no existe Ping entre PCs conectadas a través de un Linux Bridge con veth pairs):

```
$ sudo brctl stp brtest on
```

Note: Para ver todas las opciones disponibles de `brctl` usar: `brctl --help`

```
$ brctl --help
Usage: brctl [commands]
commands:
```

(continues on next page)

(continued from previous page)

```
        addbr           <bridge>                add bridge
        delbr           <bridge>                delete bridge
        addif           <bridge> <device>       add interface to bridge
        delif           <bridge> <device>       delete interface from bridge
        hairpin         <bridge> <port> {on|off}        turn hairpin on/off
        setageing       <bridge> <time>         set ageing time
        setbridgeprio   <bridge> <prio>         set bridge priority
        setfd           <bridge> <time>         set bridge forward delay
        sethello        <bridge> <time>         set hello time
        setmaxage       <bridge> <time>         set max message age
        setpathcost     <bridge> <port> <cost>  set path cost
        setportprio     <bridge> <port> <prio>  set port priority
        show            [ <bridge> ]            show a list of bridges
        showmacs        <bridge>                show a list of mac addrs
        showstp         <bridge>                show bridge stp info
        stp             <bridge> {on|off}       turn stp on/off
```

28.1.4 Linux Bridges con Network Namespaces e Interfaces veth

Namespaces

Teoría de Network Namespaces

Una instalación de Linux comparte un único conjunto de interfaces de red y entradas de tablas de enrutamiento. Con **network namespaces** podemos tener instancias diferentes y separadas de las interfaces de red y tablas de enrutamiento que operan independientemente una de otra. (Referencia: Introducing Linux Network Namespaces)

Creando y listando network namespaces

- Para crear un network namespace usamos el comando `ip netns`:

```
$ sudo ip netns add mynetns
```

- Para listar los network namespaces creados usamos:

```
$ ip netns show

mynetns
```

Luego de haber creado un network namespace el siguiente paso será asignar interfaces (virtuales o físicas) al namespace para luego configurarlas y lograr conectividad de red.

Interfaces veth

Asignando interfaces veth a network namespaces

Una opción para asignar interfaces a network namespaces son las interfaces virtuales también conocidas como **interfaces virtual Ethernet (veth)** a un network namespace.

Las interfaces veth están construidas en pares, de forma que lo que entre por una interfaz veth salga por su otra interfaz veth par (**peer**). De esta forma podemos usar las interfaces veth para conectar un network namespace con el mundo exterior.

Para crear un veth pair usamos el comando `ip`:

```
$ sudo ip link add vethA type veth peer name vethB
```

Otra opción equivalente es agregando `dev` al comando: `sudo ip link add dev veth1 type veth peer name veth2`

Podemos ver el par de interfaces veth creado con el comando `ip`:

```
$ ip link

8: vethB@vethA: <BROADCAST,MULTICAST,M-DOWN> mtu 1500 qdisc noop state DOWN mode␣
↪DEFAULT group default qlen 1000
    link/ether 96:58:a7:73:a7:b1 brd ff:ff:ff:ff:ff:ff
9: vethA@vethB: <BROADCAST,MULTICAST,M-DOWN> mtu 1500 qdisc noop state DOWN mode␣
↪DEFAULT group default qlen 1000
    link/ether 1e:da:c1:92:7b:99 brd ff:ff:ff:ff:ff:ff
```

Como vemos, ambas interfaces pertenecen al namespace "**default**" o "**global**", junto con las interfaces físicas.

Digamos que ahora queremos conectar el namespace global al namespace creado por nosotros (`mynetns`). Para hacerlo, movemos una de las interfaces veth a nuestro namespace `mynetns` con el comando `ip`:

```
$ sudo ip link set vethA netns mynetns
```

Si fuésemos a listar las interfaces físicas de nuestro sistema, veremos que la interfaz `vethA` ha desaparecido de la lista:

```
$ ip link

8: vethB@if9: <BROADCAST,MULTICAST> mtu 1500 qdisc noop state DOWN mode DEFAULT group␣
↪default qlen 1000
    link/ether 96:58:a7:73:a7:b1 brd ff:ff:ff:ff:ff:ff link-netnsid 0
```

Ahora la interfaz `vethA` se encuentra en el namespace `mynetns`, como lo podemos comprobar con:

```
$ sudo ip netns exec mynetns ip link show

1: lo: <LOOPBACK> mtu 65536 qdisc noop state DOWN mode DEFAULT group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
9: vethA@if8: <BROADCAST,MULTICAST> mtu 1500 qdisc noop state DOWN mode DEFAULT group␣
↪default qlen 1000
    link/ether 1e:da:c1:92:7b:99 brd ff:ff:ff:ff:ff:ff link-netnsid 0
```

Podemos dividir el comando en dos partes:

- `ip netns exec mynetns`: ejecutar un comando en un network namespace diferente (`mynetns`)
- `ip link show`: comando a ejecutar en el network namespace

En general, podemos ejecutar cualquier comando para la configuración de red dentro del namespace (`ip addr`, `ip route`, `ip link`, `ifconfig`, `ping`) con el siguiente formato:

```
$ ip netns exec <netns> <comando a correr en el netns>
```

Conectando una interfaz veth al bridge

Para lograr la comunicación de dos network namespaces (o VMs) podemos conectar estos dispositivos a un bridge. Para esto, conectaremos cada network namespace (o VM) con un veth pair al bridge. Sigamos el siguiente esquema como ejemplo:

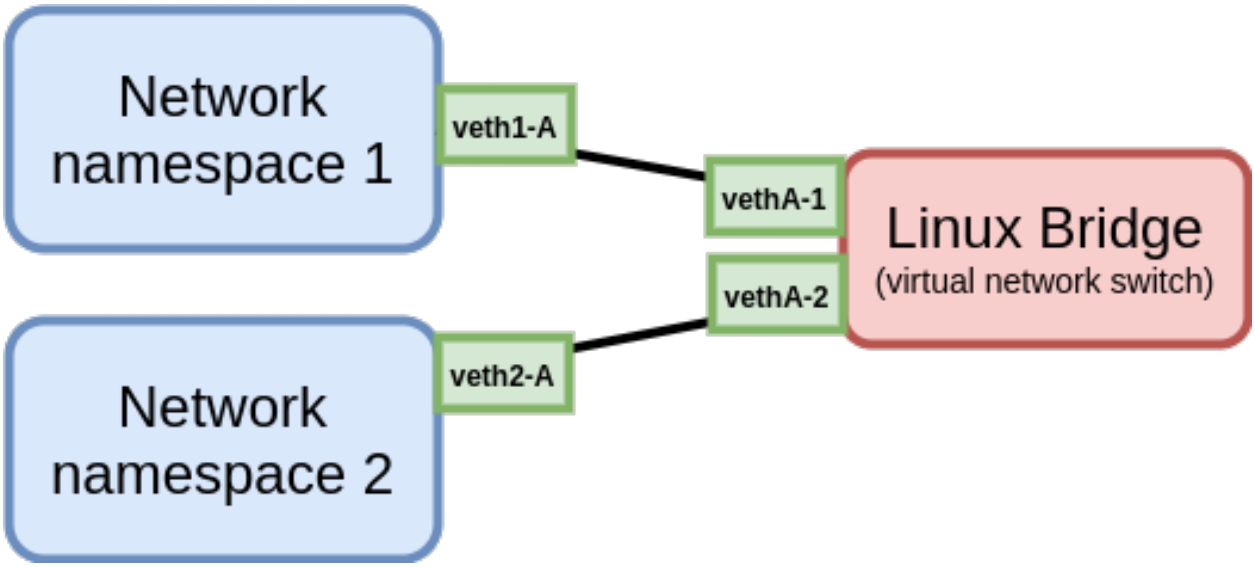


Fig. 2: Conexión entre network namespaces con un Linux bridge e interfaces veth

1. Primero crearemos los namespaces `PC1` y `PC2`:

```
$ sudo ip netns add PC1
$ sudo ip netns add PC2

$ ip netns show
PC2
PC1
```

2. Luego crearemos el bridge `SW-A`:

```
$ sudo brctl addbr SW-A

$ brctl show
bridge name     bridge id               STP enabled     interfaces
SW-A            8000.000000000000       no
```

3. Posteriormente creamos dos veth pair:

```
$ sudo ip link add veth1-A type veth peer name vethA-1
$ sudo ip link add veth2-A type veth peer name vethA-2
```

4. Asignamos una interfaz de cada veth pair a un network namespace:

```
$ sudo ip link set veth1-A netns PC1
$ sudo ip link set veth2-A netns PC2
```

5. Asignamos el otro extremo de cada veth pair al único bridge creado:

```
$ sudo brctl addif SW-A vethA-1
$ sudo brctl addif SW-A vethA-2

$ brctl show
bridge name     bridge id               STP enabled     interfaces
SW-A            8000.168a27b2ea41       no              vethA-1
                                                        vethA-2
```

Como vemos, hemos conectado un extremo de cada veth pair al bridge `SW-A`.

6. Asignar una dirección IP dentro del mismo rango de red a cada interfaz del veth pair conectada a cada namespace:

```
$ sudo ip netns exec PC1 ip addr add 192.168.99.1/24 dev veth1-A
$ sudo ip netns exec PC2 ip addr add 192.168.99.2/24 dev veth2-A
```

7. Prender las interfaces conectadas a los network namespace y al bridge:

```
# Interfaces conectadas a namespaces PC1 y PC2
$ sudo ip netns exec PC1 ip link set dev veth1-A up
$ sudo ip netns exec PC2 ip link set dev veth2-A up

# Interfaz propia del bridge
$ sudo ip link set dev SW-A up

# Interfaces conectadas al bridge
$ sudo ip link set dev vethA-1 up
$ sudo ip link set dev vethA-2 up
```

8. Revisamos la configuración de las interfaces de los network namespace y el bridge:

```
# Configuración de red de network namespace PC1
$ sudo ip netns exec PC1 ip addr show
1: lo: <LOOPBACK> mtu 65536 qdisc noop state DOWN group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
12: veth1-A@if11: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc noqueue state UP␣
↪group default qlen 1000
    link/ether 7e:81:4e:49:c2:a8 brd ff:ff:ff:ff:ff:ff link-netnsid 0
    inet 192.168.99.1/24 scope global veth1-A
        valid_lft forever preferred_lft forever
    inet6 fe80::7c81:4eff:fe49:c2a8/64 scope link
        valid_lft forever preferred_lft forever

# Configuración de red de network namespace PC2
$ sudo ip netns exec PC2 ip addr show
1: lo: <LOOPBACK> mtu 65536 qdisc noop state DOWN group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
14: veth2-A@if13: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc noqueue state UP␣
↪group default qlen 1000
    link/ether 82:84:81:8a:a2:ac brd ff:ff:ff:ff:ff:ff link-netnsid 0
    inet 192.168.99.2/24 scope global veth2-A
        valid_lft forever preferred_lft forever
    inet6 fe80::8084:81ff:fe8a:a2ac/64 scope link
        valid_lft forever preferred_lft forever

# Configuración de red del bridge SW-A
$ ip link
10: SW-A: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc noqueue state UP mode␣
↪DEFAULT group default qlen 1000
```

(continues on next page)

(continued from previous page)

```
    link/ether 16:8a:27:b2:ea:41 brd ff:ff:ff:ff:ff:ff
11: vethA-1@if12: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc noqueue master SW-
↪A state UP mode DEFAULT group default qlen 1000
    link/ether a6:96:81:e7:bb:3b brd ff:ff:ff:ff:ff:ff link-netnsid 1
13: vethA-2@if14: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc noqueue master SW-
↪A state UP mode DEFAULT group default qlen 1000
    link/ether 16:8a:27:b2:ea:41 brd ff:ff:ff:ff:ff:ff link-netnsid 2
```

7. Probar conectividad entre network namespaces `PC1` y `PC2`:

```
$ sudo ip netns exec PC1 ping -c3 192.168.99.2
PING 192.168.99.2 (192.168.99.2) 56(84) bytes of data.
64 bytes from 192.168.99.2: icmp_seq=1 ttl=64 time=0.067 ms
64 bytes from 192.168.99.2: icmp_seq=2 ttl=64 time=0.078 ms
64 bytes from 192.168.99.2: icmp_seq=3 ttl=64 time=0.065 ms

--- 192.168.99.2 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2033ms
rtt min/avg/max/mdev = 0.065/0.070/0.078/0.005 ms

$ sudo ip netns exec PC2 ping -c3 192.168.99.1
PING 192.168.99.1 (192.168.99.1) 56(84) bytes of data.
64 bytes from 192.168.99.1: icmp_seq=1 ttl=64 time=0.057 ms
64 bytes from 192.168.99.1: icmp_seq=2 ttl=64 time=0.077 ms
64 bytes from 192.168.99.1: icmp_seq=3 ttl=64 time=0.108 ms

--- 192.168.99.1 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2029ms
rtt min/avg/max/mdev = 0.057/0.080/0.108/0.023 ms
```

28.2 Redes virtuales con Libvirt

Table of Contents

Podemos crear distintos tipos de redes virtuales con `libvirt` a través de dos herramientas distintas pero con resultados iguales. Una herramienta con interfaz gráfica para conseguir este objetivo es `virt-manager` la otra con interfaz de línea de comandos es `virsh`. Algunas de las redes virtuales que podemos crear con `libvirt` son:

- *Red virtual Isolated*
- *Red virtual Routed*
- *Red virtual NATed*
- Red virtual bridged

El procedimiento general para crear una red virtual en `virt-manager` es entrar al programa, ir a *Edit* | *Connection details* | *Virtual Networks* | Clic en el signo +. Luego solo deberemos seguir el wizard de creación.

Por otro lado, los pasos generales para crear una red virtual en `virsh` serán escribir en un archivo XML las características que definan la red deseada y luego simplemente crearla en base a este archivo.

28.2.1 Red virtual Isolated

Red cerrada para las VMs. Solo las VMs agregadas a esta red pueden comunicarse unas con otras. El host también podrá comunicarse con las VMs a añadidas a la red. El tráfico no pasará fuera del host, ni podrán recibir tráfico de fuera del host. Agregar una nueva interfaz a una VM es proceso simple que puede realizarse con `virt-manager` o `virsh`:

Usando `virt-manager`

Los pasos para crear una red virtual aislada con `virt-manager` son los siguientes:

1. Ingresar un nombre para la red virtual:
2. No realizaremos configuración IPv4:
3. No haremos una configuración para IPv6:
4. Solo mantener seleccionada la opción *Isoated virtual network* y dejar *DNS Domain Name* en blanco. Clic en *Finish* para crear la red virtual aislada:
5. Revisar los detalles de la red virtual creada en la pestaña *Virtual Networks*:

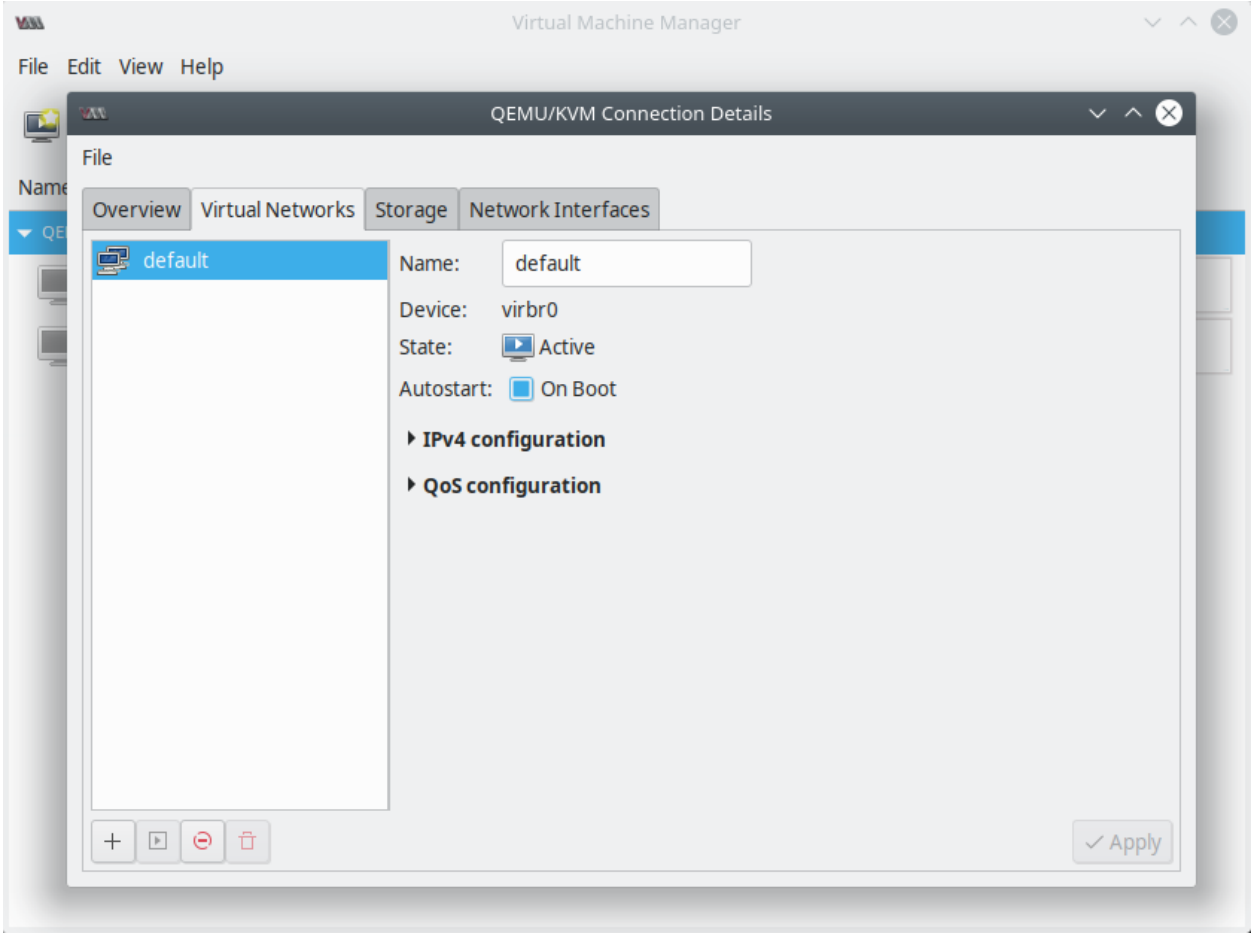


Fig. 3: `virt-manager` - Virtual Networks Tab

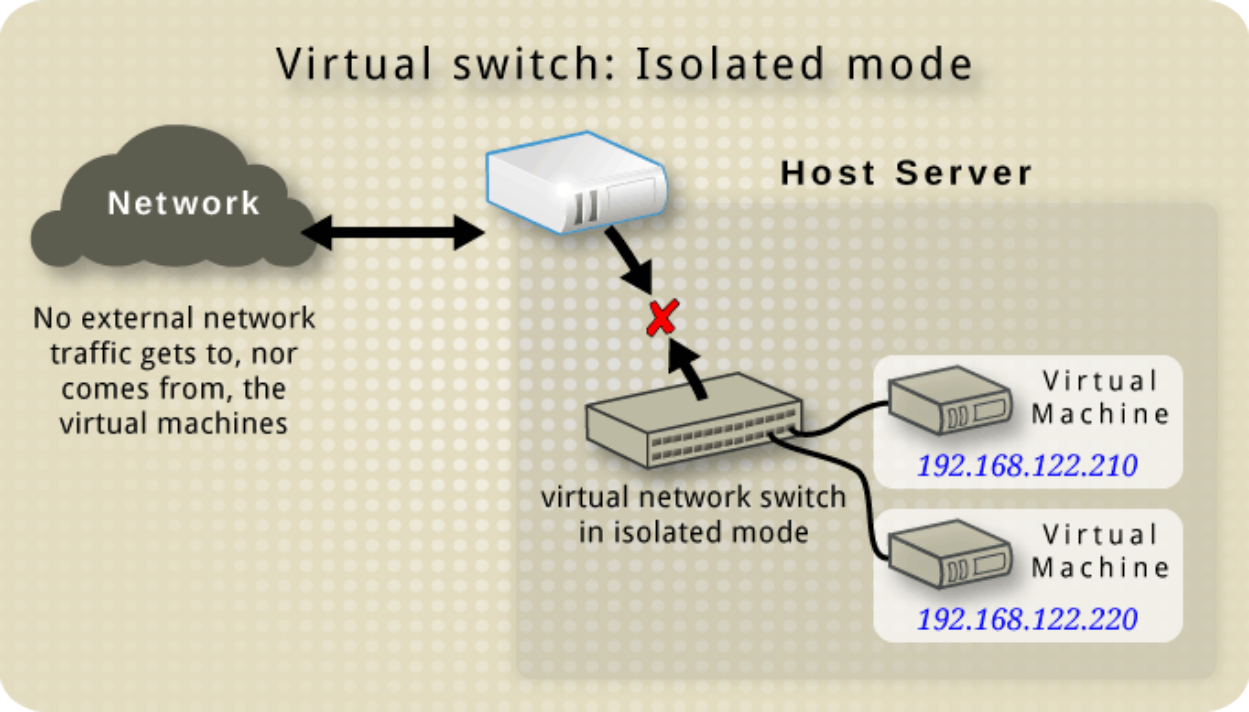


Fig. 4: Isolated mode - Libvirt Wiki

Usando `virsh`

Para crear una red virtual isolated con `virsh` realizaremos los siguientes pasos:

1. Crear un archivo XML con el contenido que defina la red:

```
$ cat isolated.xml
```

```
<network>
        <name>isolated</name>
</network>
```

Según la sintáxis:

- `<network>`: define la red virtual
- `<name>`: define el nombre de la red virtual

2. Guardar el archivo XML. Para crear una red usando el archivo XML, usar la opción `net-define` del comando `virsh` seguido por la ruta del archivo XML:

```
$ virsh net-define isolated.xml

Network isolated defined from isolated.xml
```

3. Una vez que hemos definido una red con `net-define`, el archivo de configuración será guardado en `/etc/libvirt/qemu/networks/` como un archivo XML con el mismo nombre que nuestra red virtual:

```
$ sudo cat /etc/libvirt/qemu/networks/isolated.xml
```

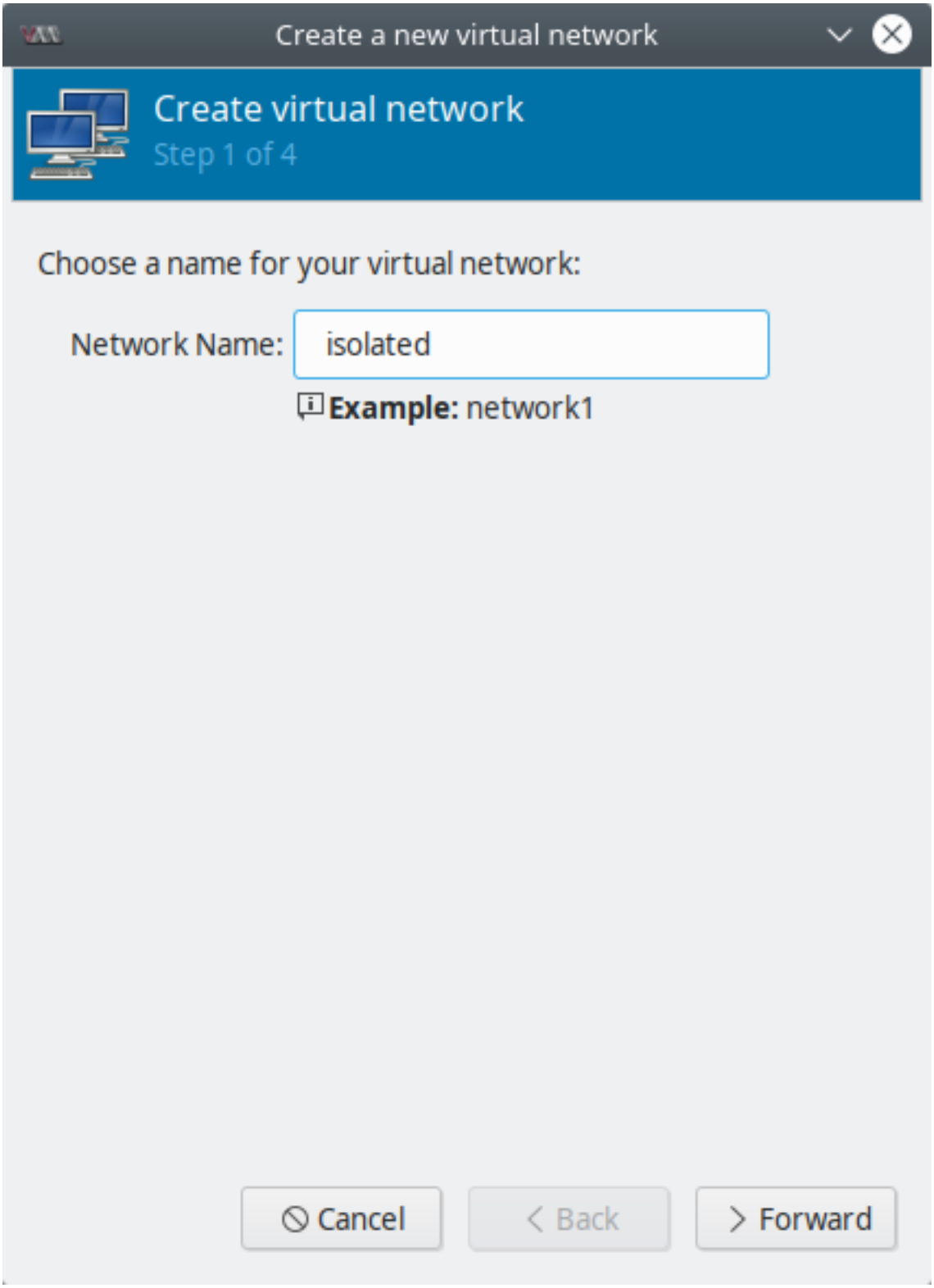


Fig. 5: `virt-manager` - Isolated virtual network - step 1

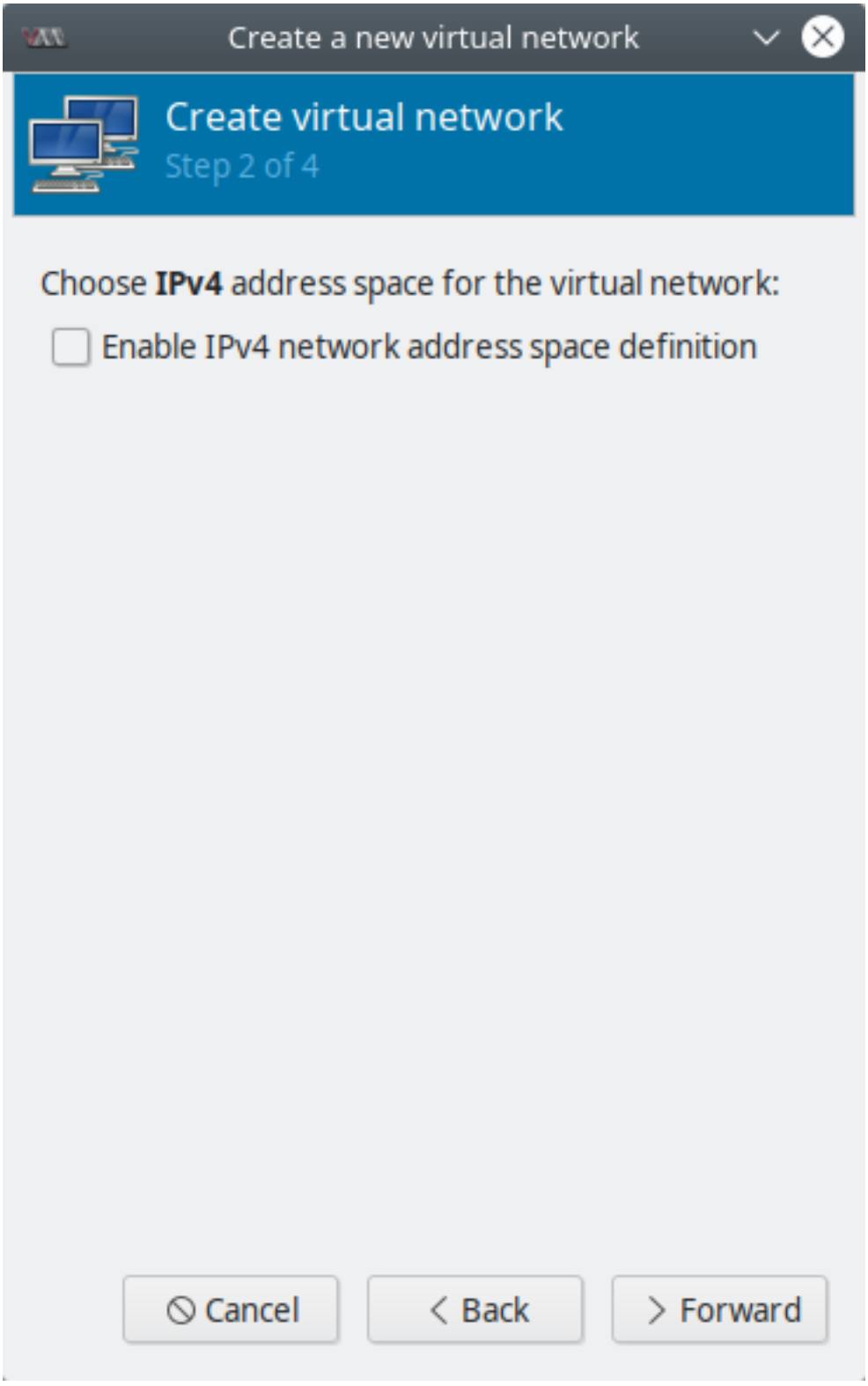


Fig. 6: `virt-manager` - Isolated virtual network - step 2

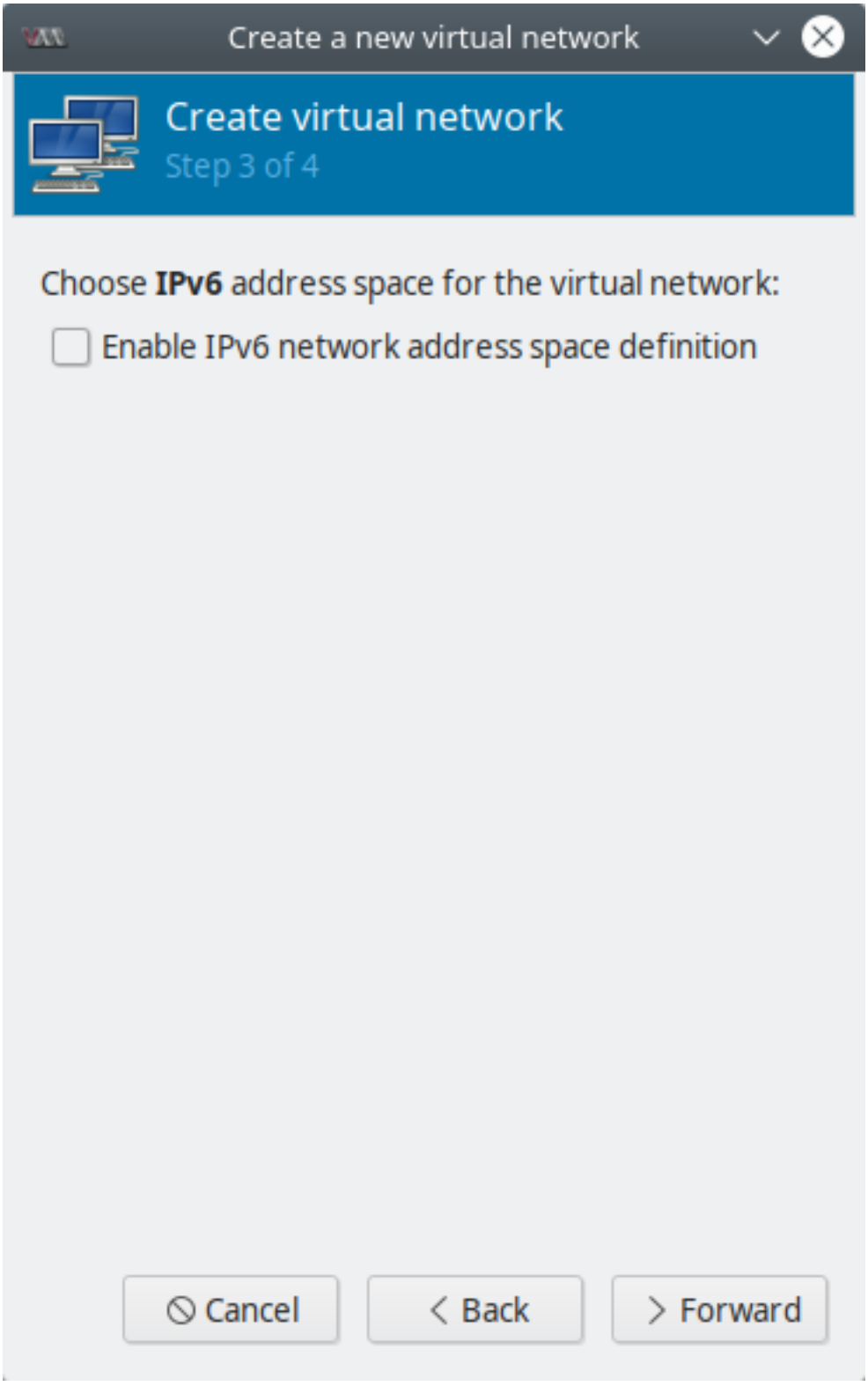


Fig. 7: `virt-manager` - Isolated virtual network - step 3

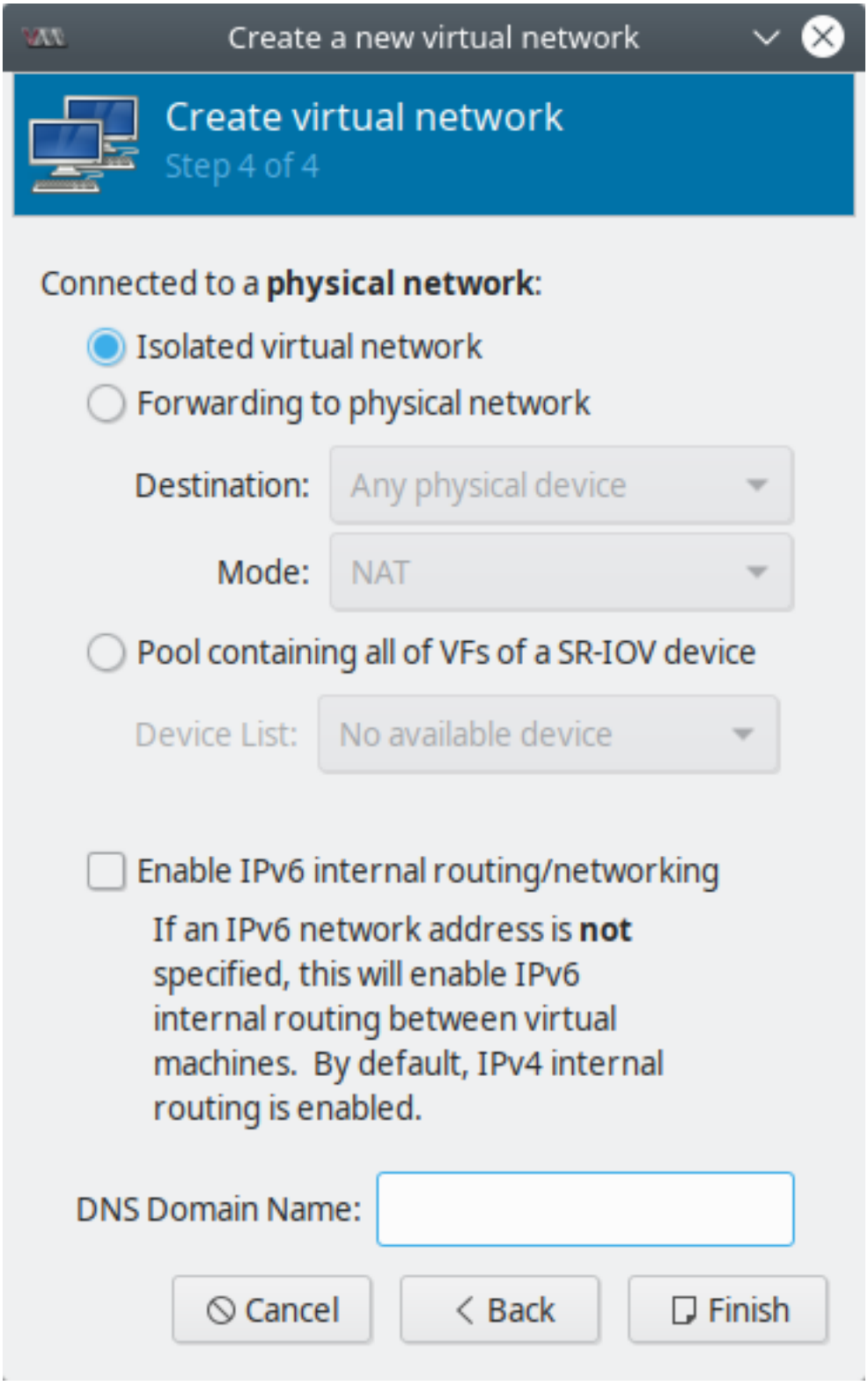


Fig. 8: `virt-manager` - Isolated virtual network - step 4

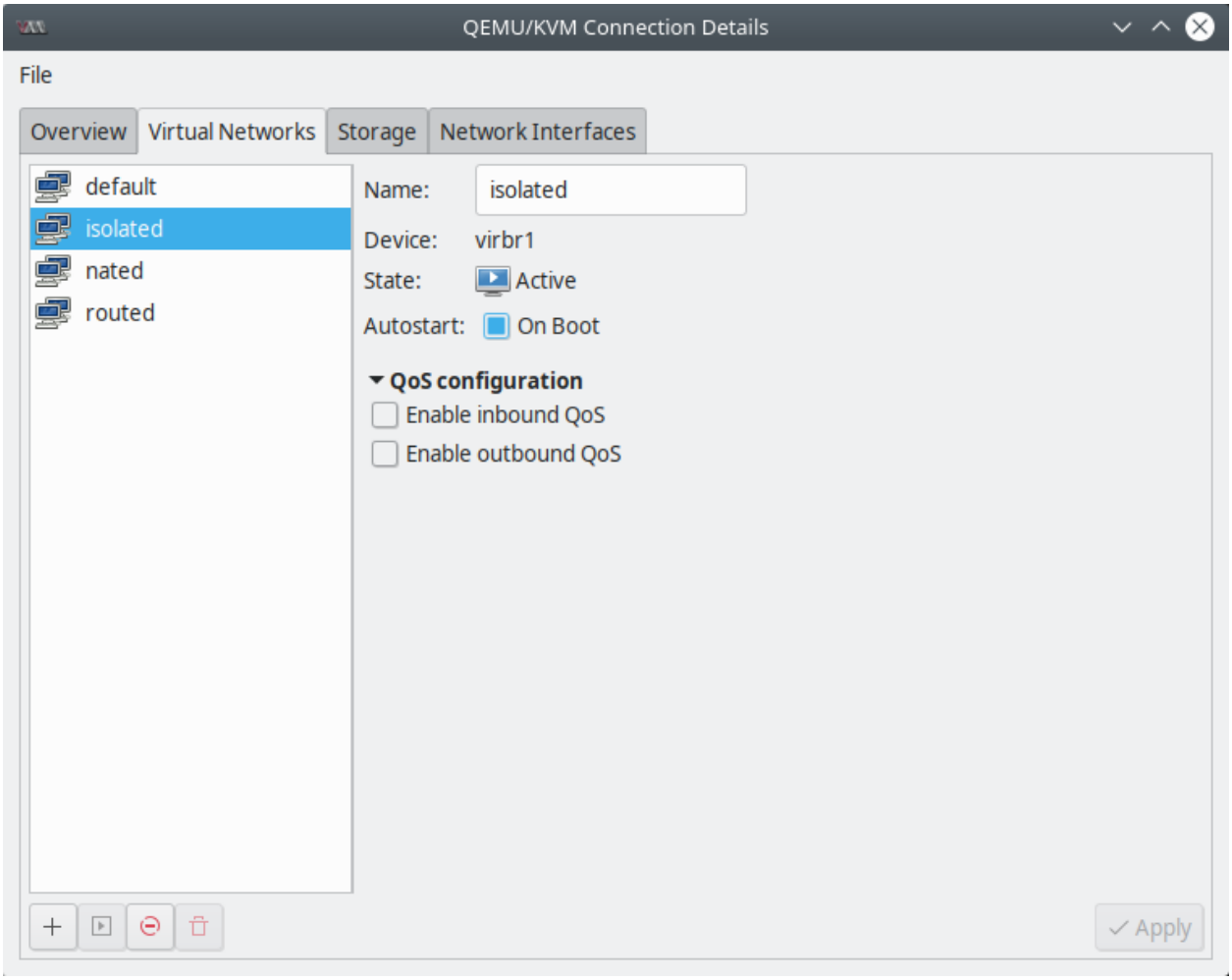


Fig. 9: `virt-manager` - Isolated virtual network creada

```
<!--
WARNING: THIS IS AN AUTO-GENERATED FILE. CHANGES TO IT ARE LIKELY TO BE
OVERWRITTEN AND LOST. Changes to this xml configuration should be made using:
virsh net-edit isolated
or other application using the libvirt API.
-->

<network>
    <name>isolated</name>
    <uuid>13fd5536-cad8-43a3-a066-91243719f95b</uuid>
    <bridge name='virbr1' stp='on' delay='0'/>
    <mac address='52:54:00:c4:d1:d7'/>
</network>
```

`libvirt` añadió el resto de parámetros requeridos, podremos agregarlos nosotros en el archivo XML cuando lo necesitemos al momento de crear el bridge.

Como vemos en el comentario del archivo, cualquier cambio para editar la red virtual deberá ser a través de `virsh net-edit` u otra aplicación que usa la API de `libvirt`.

28.2.2 Red virtual Routed

En **modo routed**, la red virtual es conectada a la red física usando las rutas IP especificadas en el hypervisor. Estas rutas IP son usadas para enrutar el tráfico desde las VMs a la red conectada al hypervisor. La clave será configurar la ruta IP correcta en nuestro dispositivo router o gateway para que el paquete de respuesta alcance de vuelta al hypervisor. Si no existen rutas definidas, el paquete de respuesta nunca alcanzará al host.

En este modo todas las VMs están en una subred enrutada a través del switch virtual. Hasta este punto, los otros hosts en la red física no conocen que esta red existe o cómo llegar a ella. Por lo tanto, es necesario configurar rutas estáticas en la red física. En la siguiente gráfica el host actúa como router, permitiendo que equipos externos puedan comunicarse con la VM:

Usando `virt-manager`

Los pasos para crear una red modo routed usando `virt-manager` serán los siguientes:

1. Ingresar un nombre para la red virtual:
2. En la siguiente pantalla habilitar IPv4 (*Enable IPv4 network address space definition*) y *DHCP* (*Enable DHCPv4*). Deshabilitar *Static Route*.

Definimos `10.10.10.0/24` como nuestra red. `libvirt` asignará automáticamente un gateway para nuestra red. Usualmente será la primera IP en el rango definido (`10.10.10.1` en nuestro caso) y será asignado a la interfaz bridge. Podemos definir el rango DHCP según deseemos:

3. No haremos una configuración para IPv6:
4. En el último paso, seleccionar la opción *Forwarding to physical network* y seleccionar la **interfaz host** donde deseamos que el tráfico sea **reenviado (forward)** desde la red virtual. Y seleccionar el *Mode* como *Routed*
5. Revisar los detalles de la red virtual creada en la pestaña *Virtual Networks*:

Usando `virsh`

Para crear una red virtual routed con `virsh` realizaremos los siguientes pasos:

1. Crear un archivo XML con el contenido que defina la red:

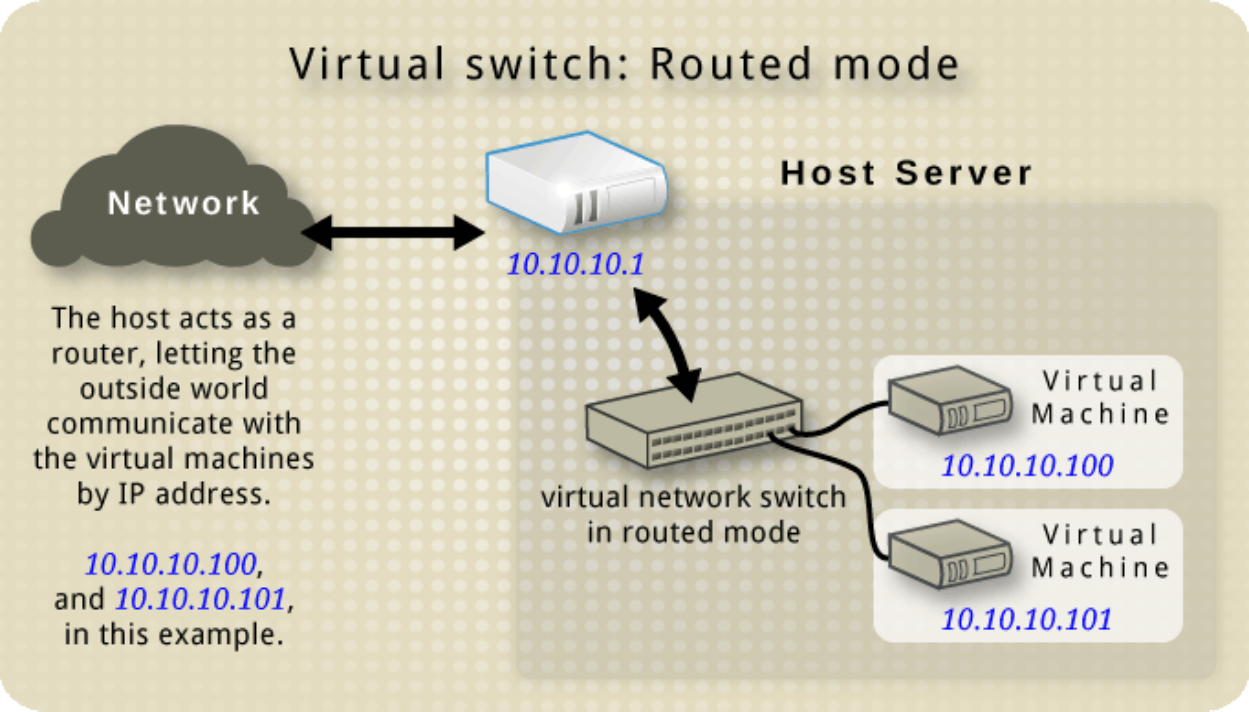


Fig. 10: Routed mode - Libvirt Wiki

```
$ cat routed.xml
```

```
<network>
    <name>routed</name>
    <forward dev='wlp2s0' mode='route'>
        <interface dev='wlp2s0' />
    </forward>
    <ip address='192.168.10.1' netmask='255.255.255.0'>
        <dhcp>
            <range start='192.168.10.128' end='192.168.10.254'/>
        </dhcp>
    </ip>
</network>
```

Según la sintáxis:

- `<network>`: define la red virtual
- `<name>`: define el nombre de la red virtual
- `<forward>`: configura el modo y el dispositivo de forwarding
- `<ip>`: configura la dirección de red
- `<dhcp><range>`: configura el rango DHCP de direcciones IP

2. Guardar el archivo XML. Para crear una red usando el archivo XML, usar la opción `net-define` del comando `virsh` seguido por la ruta del archivo XML:

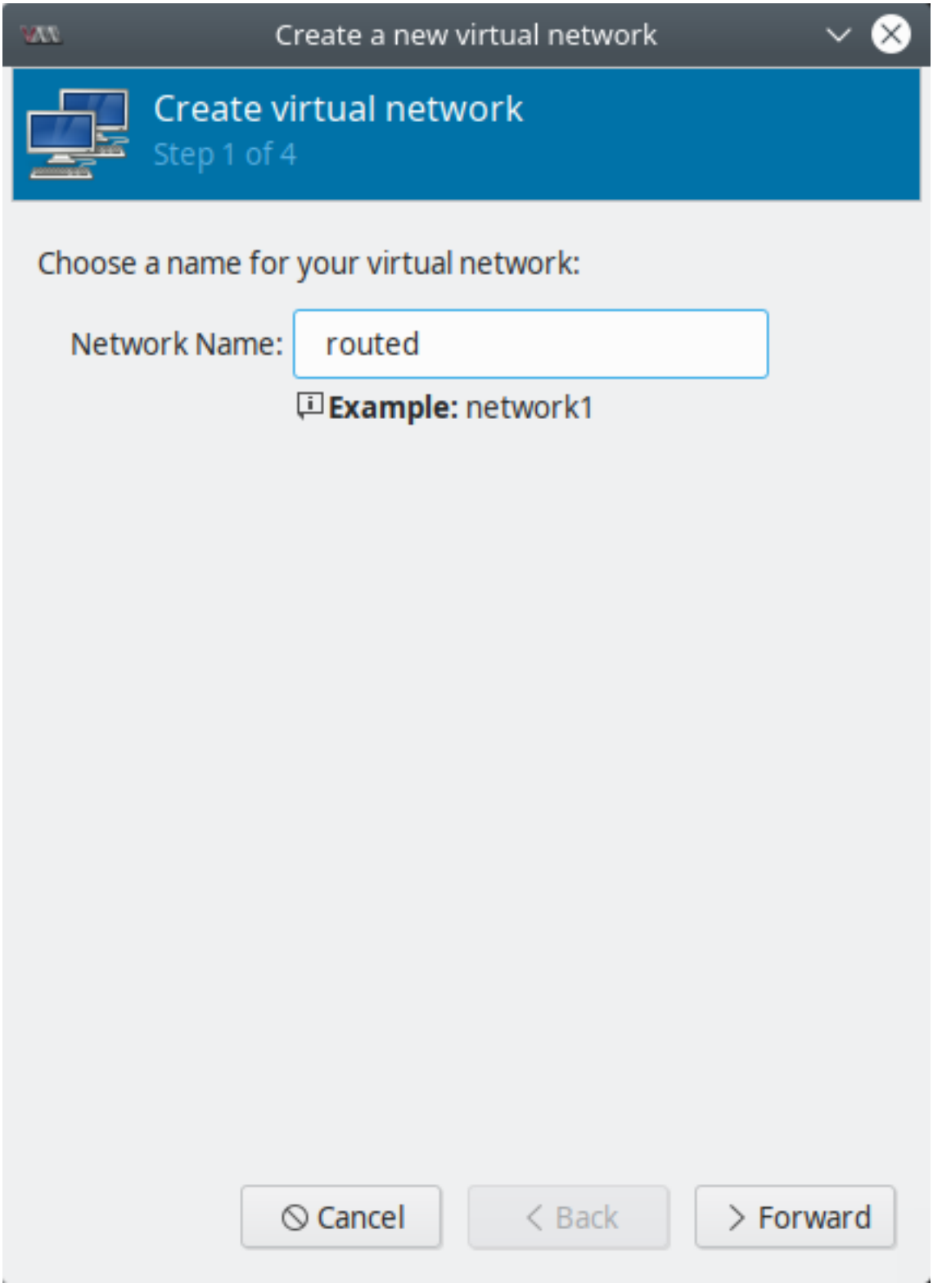


Fig. 11: `virt-manager` - Routed virtual network - step 1

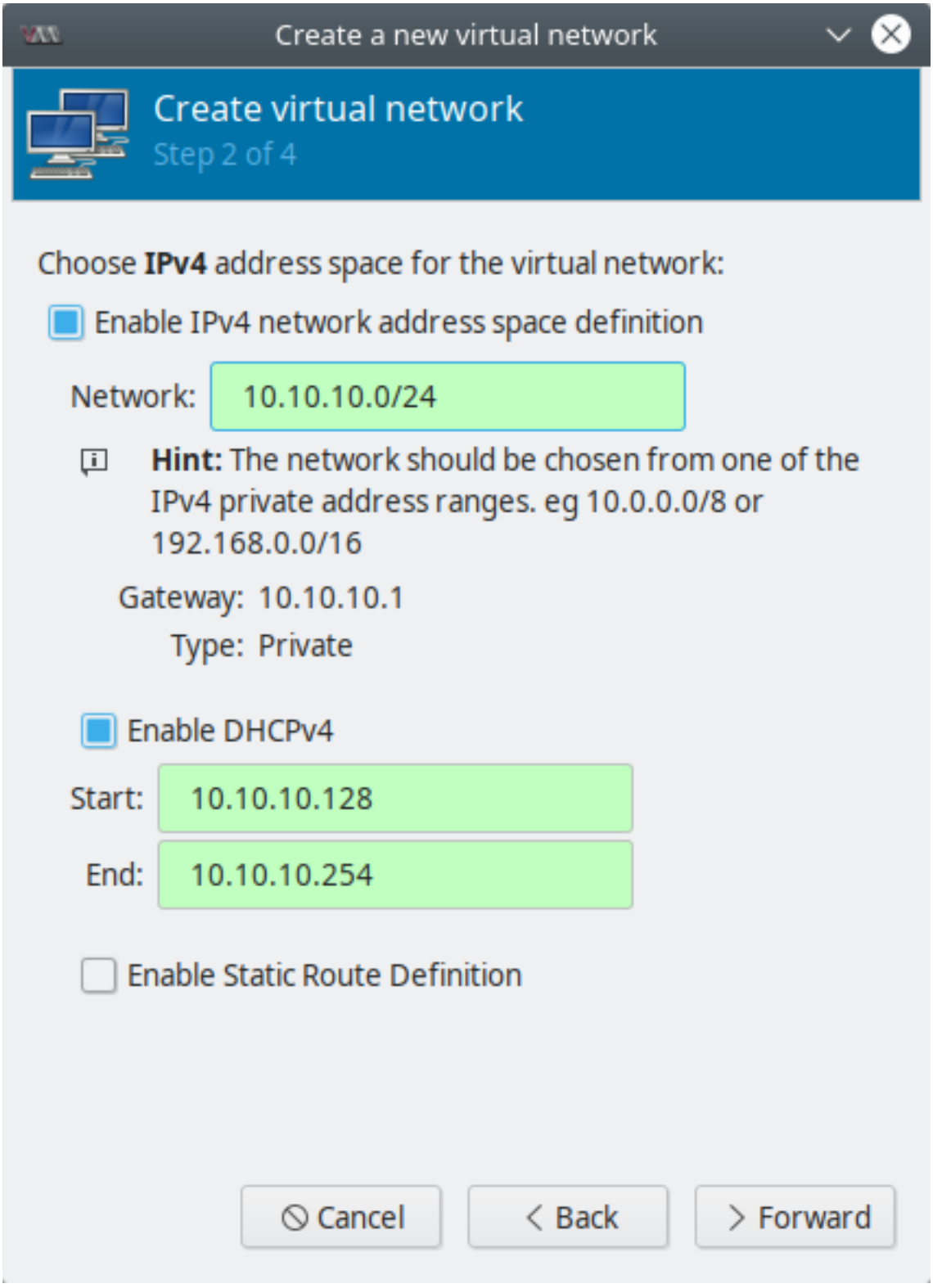


Fig. 12: `virt-manager` - Routed virtual network - step 2

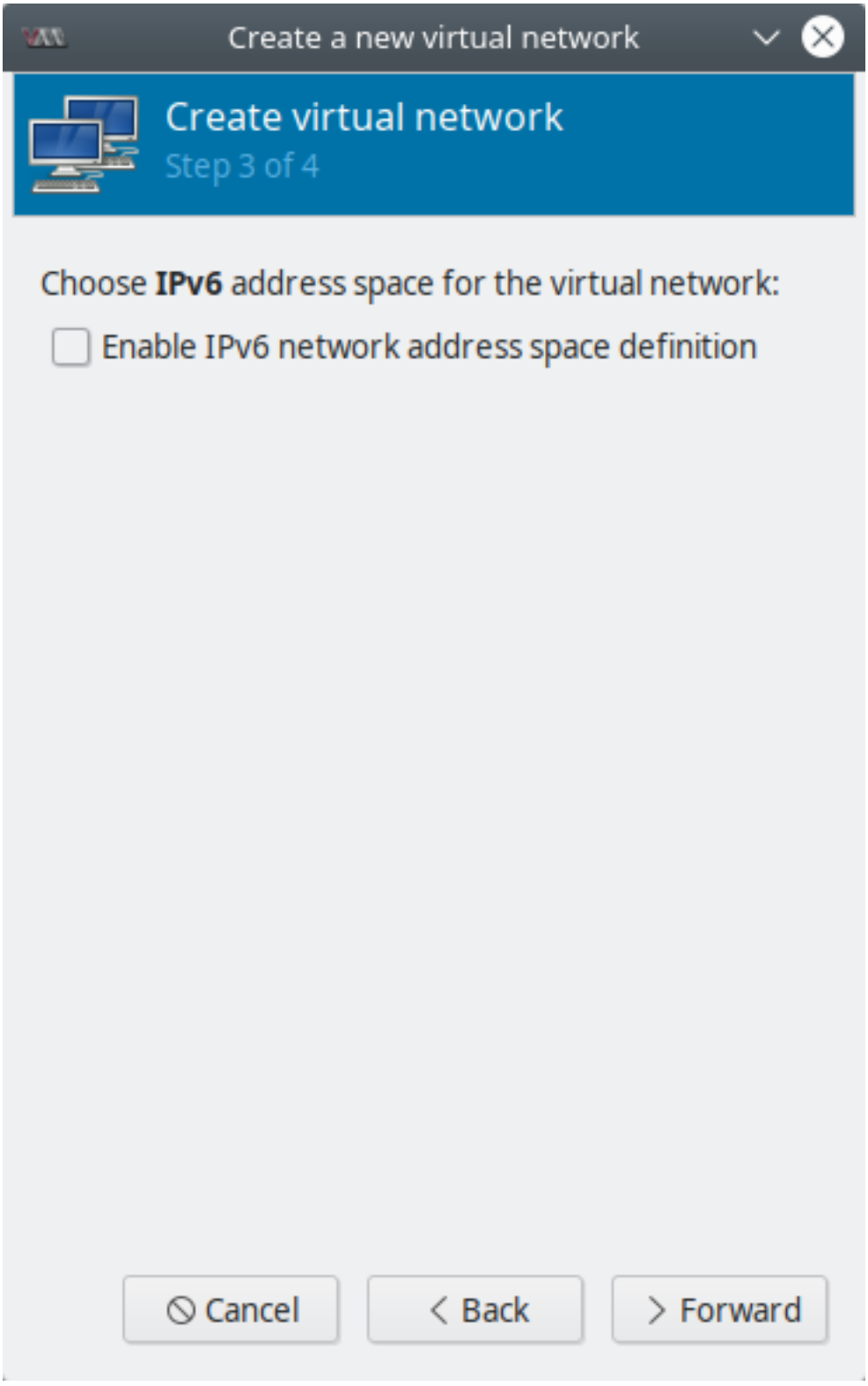


Fig. 13: `virt-manager` - Routed virtual network - step 3

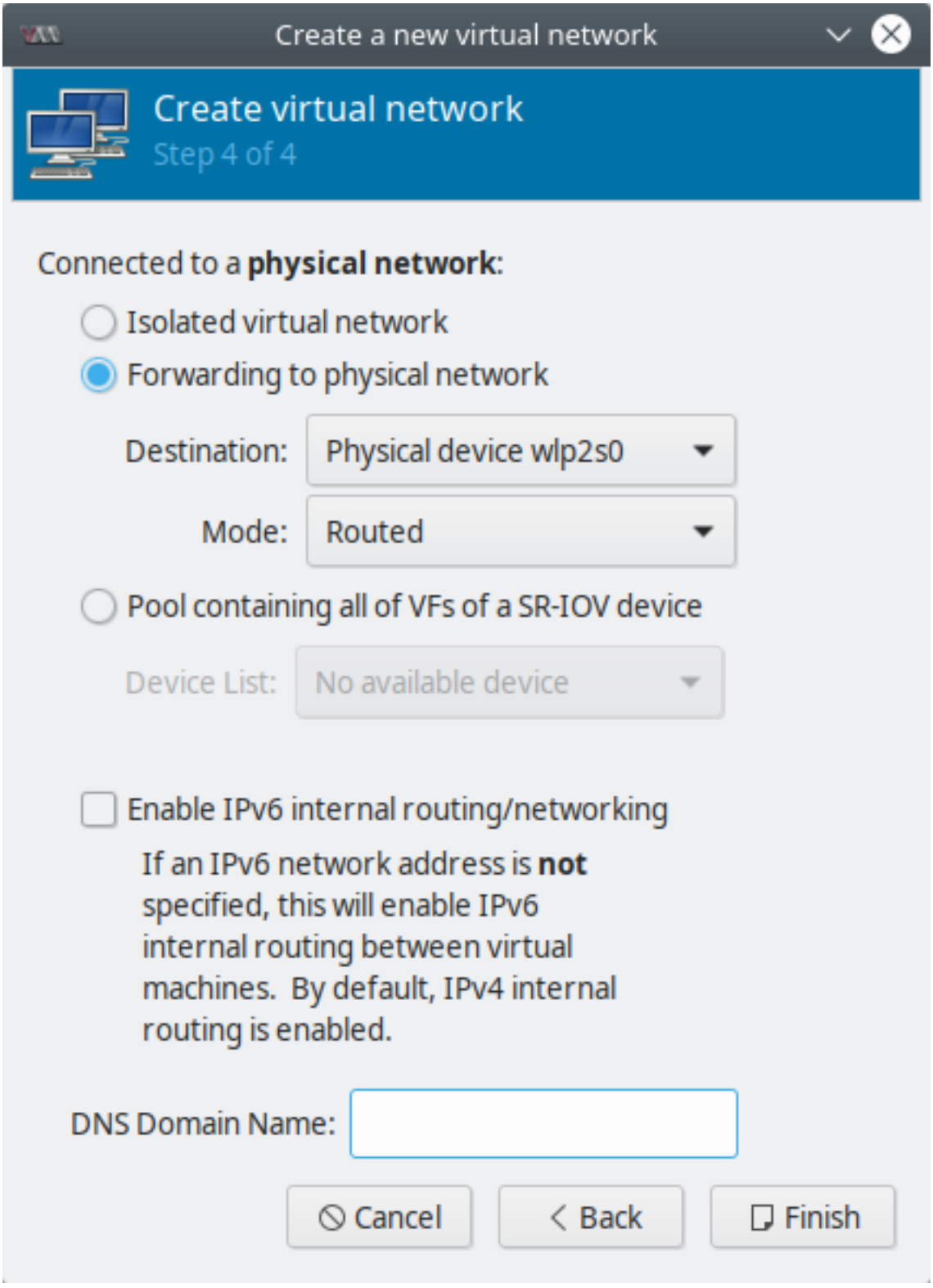


Fig. 14: `virt-manager` - Routed virtual network - Step 4

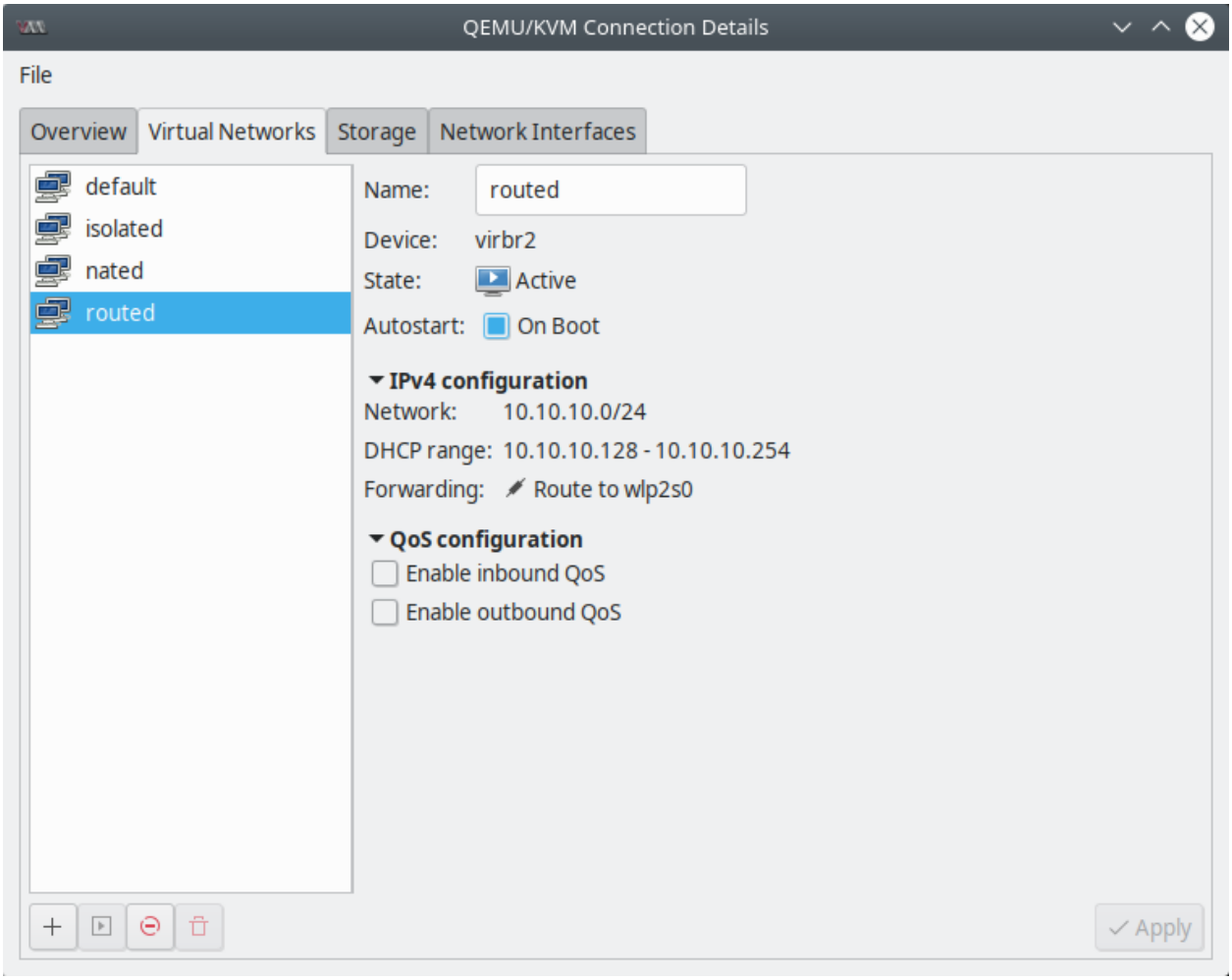


Fig. 15: `virt-manager` - Routed virtual network creada

```
$ virsh net-define routed.xml

Network routed defined from routed.xml
```

3. Una vez que hemos definido una red con `net-define`, el archivo de configuración será guardado en `/etc/libvirt/qemu/networks/` como un archivo XML con el mismo nombre que nuestra red virtual:

```
$ sudo cat /etc/libvirt/qemu/networks/routed.xml
```

```
<!--
WARNING: THIS IS AN AUTO-GENERATED FILE. CHANGES TO IT ARE LIKELY TO BE
OVERWRITTEN AND LOST. Changes to this xml configuration should be made using:
virsh net-edit routed
or other application using the libvirt API.
-->

<network>
<name>routed</name>
<uuid>43d8bf8b-3558-4ed0-85d3-66b8de7783ff</uuid>
<forward dev='wlp2s0' mode='route'>
    <interface dev='wlp2s0'/>
</forward>
<bridge name='virbr2' stp='on' delay='0'/>
<mac address='52:54:00:76:20:86'/>
<ip address='192.168.10.1' netmask='255.255.255.0'>
    <dhcp>
        <range start='192.168.10.128' end='192.168.10.254'/>
    </dhcp>
</ip>
</network>
```

28.2.3 Red virtual NATed

El **modo NATed** es el modo comúnmente usado en virtual networking cuando deseamos configurar un ambiente de pruebas en una máquina. Bajo este modo las VMs pueden comunicarse con el mundo exterior sin usar ninguna configuración adicional. También es posible la comunicación entre el hypervisor y las VMs. La gran desventaja de esta red virtual es que ninguno de los sistemas del exterior del hypervisor pueden llegar a las VMs.

La red virtual NATed es creada con ayuda de **iptables**, específicamente usando la opción **masquerading**. Entonces, si paramos el servicio de `iptables` mientras las VMs están siendo usadas puede ocasionar una error de red en las VMs.

Usando `virt-manager`

Los pasos para crear una red virtual en modo NATed a través de `virt-manager` son similares al modo Routed, a excepción del último paso:

1. Ingresar un nombre para la red virtual:
2. En la siguiente pantalla habilitar IPv4 (*Enable IPv4 network address space definition*) y *DHCP* (*Enable DHCPv4*). Deshabilitar *Static Route*.

Definimos `192.168.120.0` como nuestra red. `libvirt` asignará automáticamente un gateway para nuestra red. Usualmente será la primera IP en el rango definido (`192.168.120.1` en nuestro caso) y será asignado a la interfaz bridge. Podemos definir el rango DHCP según deseemos:

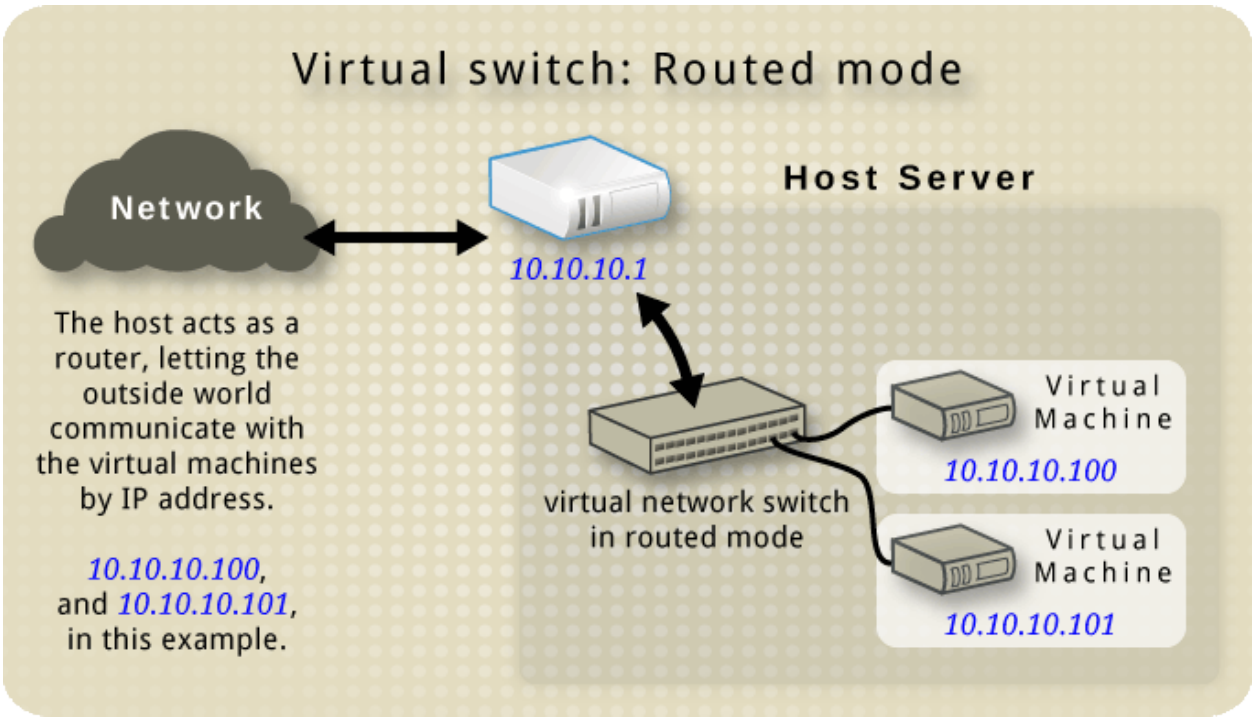


Fig. 16: NATed mode - Libvirt Wiki

3. No haremos una configuración para IPv6:
4. En el último paso, seleccionar la opción *Forwarding to physical network* y seleccionar la **interfaz host** hacia donde deseamos que el tráfico sea **NATeado** desde la red virtual. Y seleccionar el *Mode* como *NAT*
5. Revisar los detalles de la red virtual creada en la pestaña *Virtual Networks*:

28.2.4 Siguientes pasos con `virsh`

Crear una red temporal con `virsh`

`virsh net-create` es similar a `virsh net-define`. La diferencia es que no creará una **red virtual persistent**, en cambio, una vez que sea destruida será removida.

Listar redes con `virsh`

Una vez que la red se ha creado podemos listar todas las redes usando el comando `net-list`:

```
$ virsh net-list --all

Name                 State      Autostart     Persistent
----------------------------------------------------------
default              active     yes           yes
isolated             inactive   no            yes
```

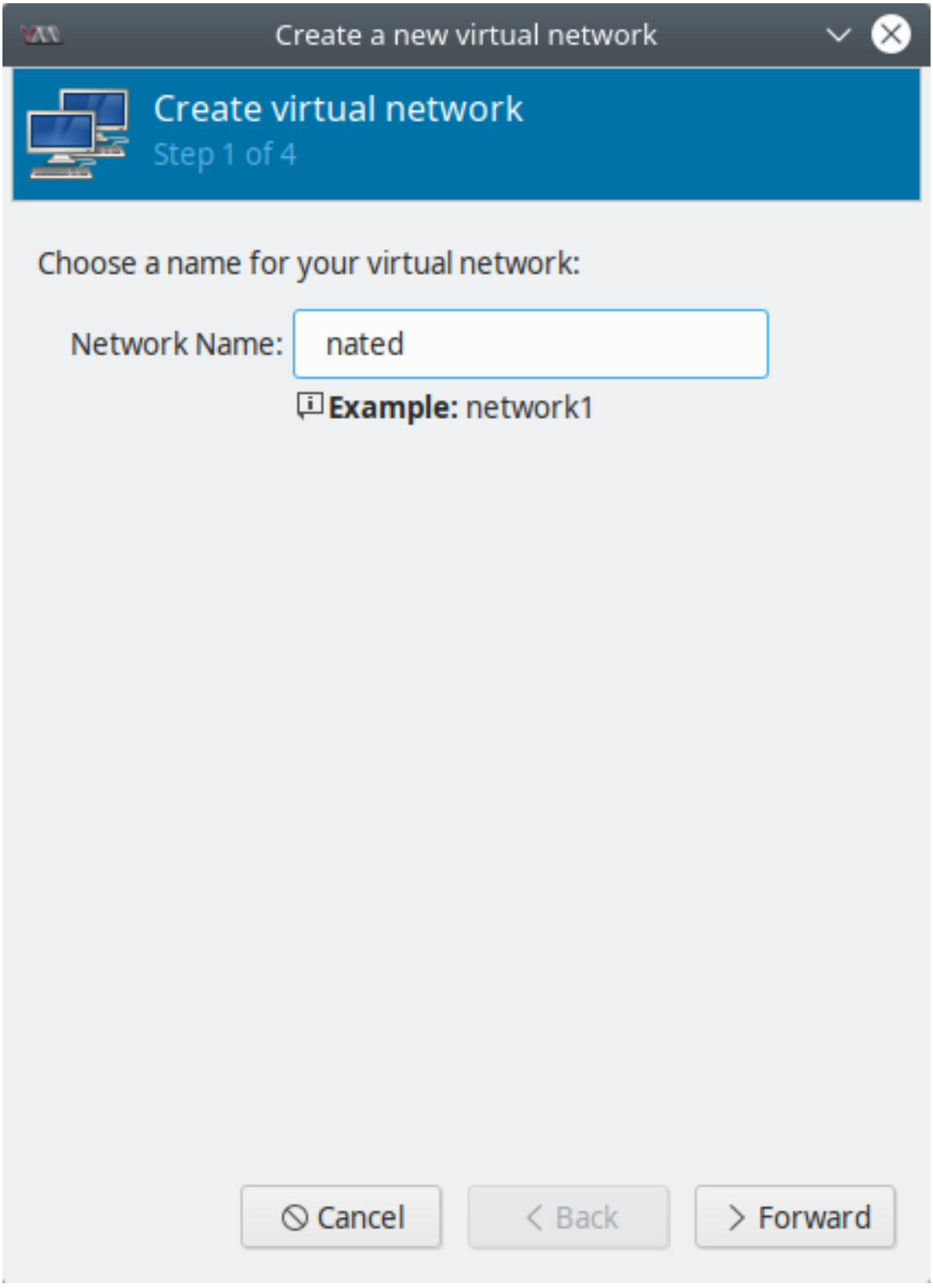


Fig. 17: `virt-manager` - NATed virtual network - step 1

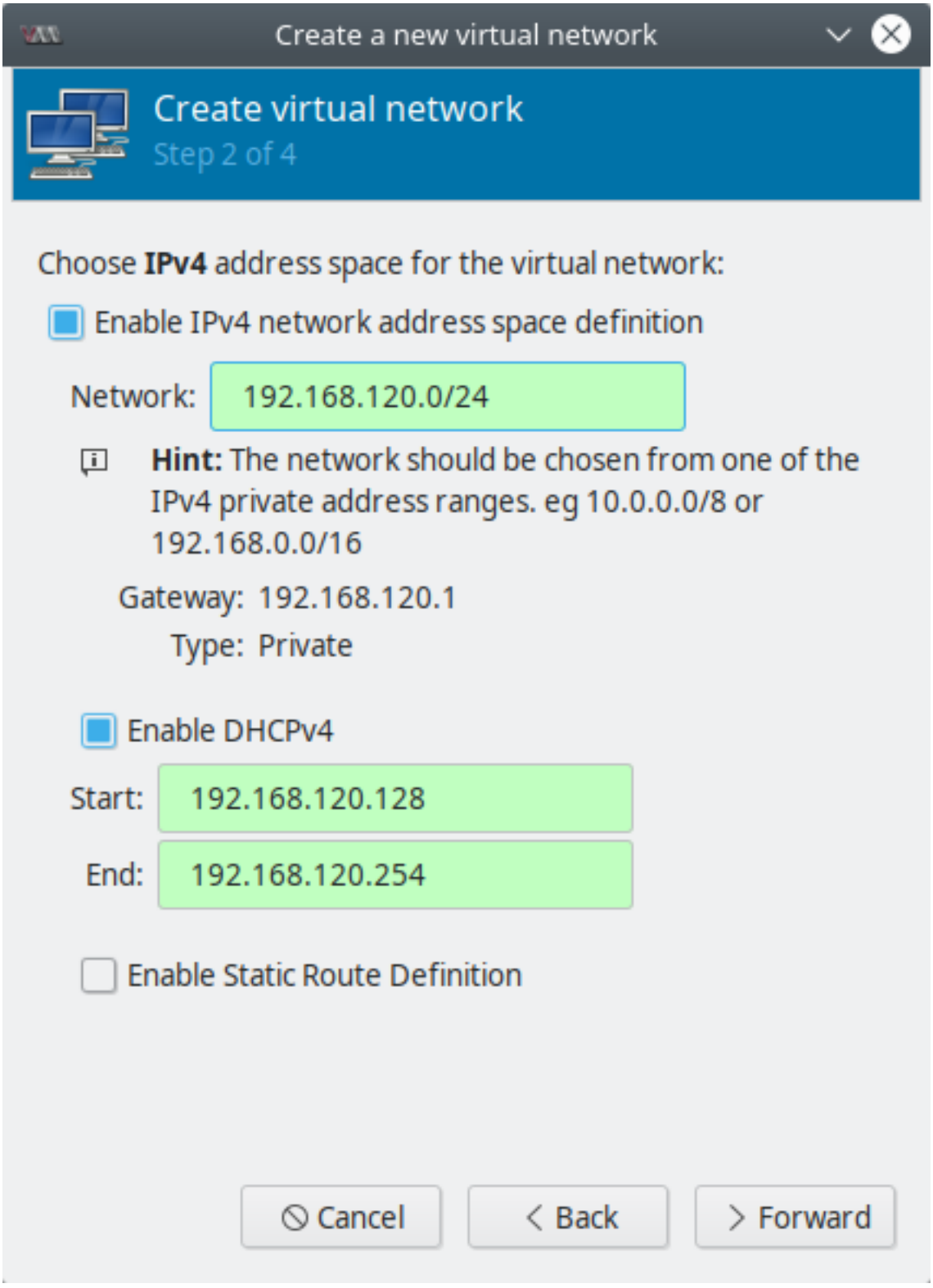


Fig. 18: `virt-manager` - NATed virtual network - step 2

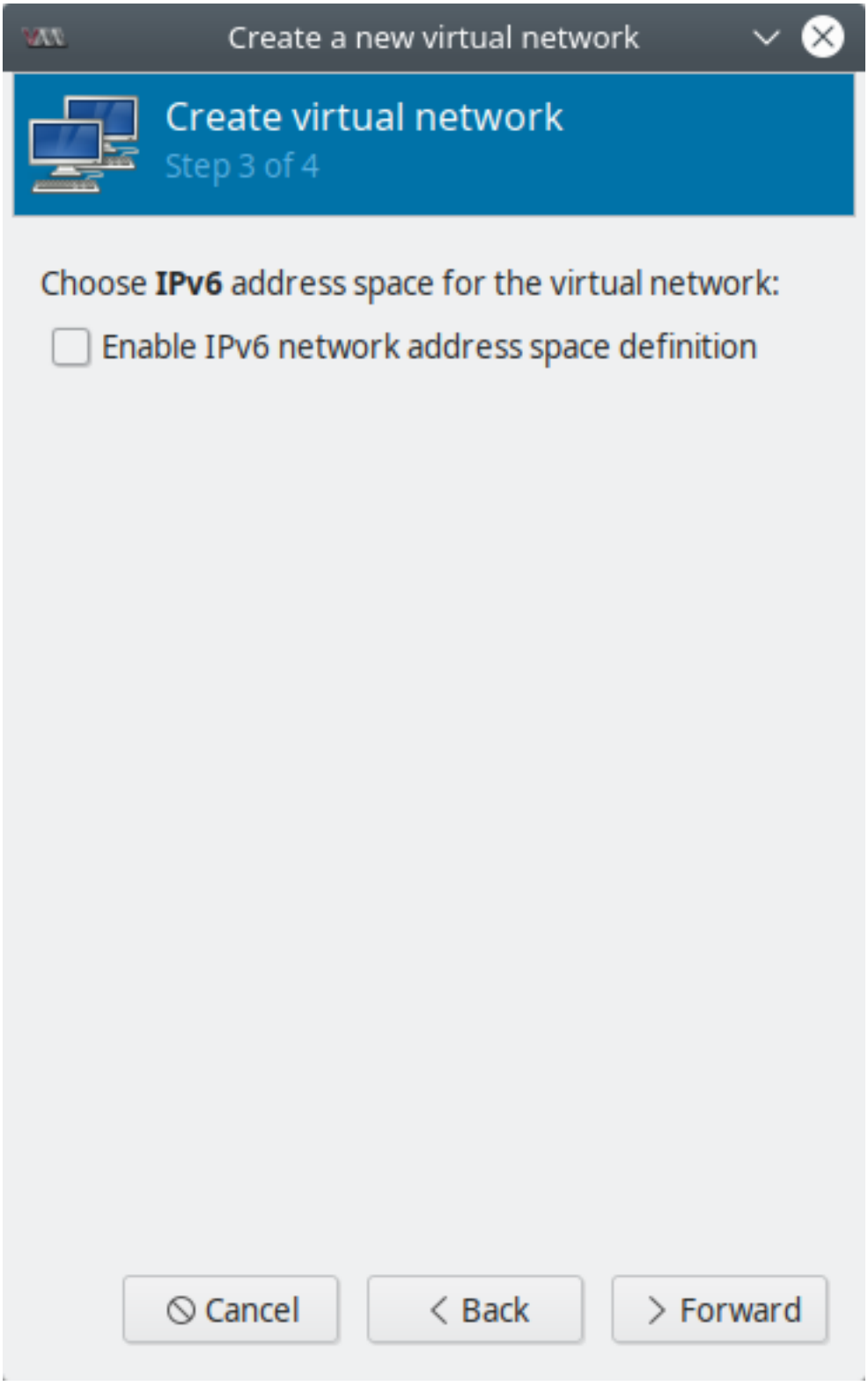


Fig. 19: `virt-manager` - NATed virtual network - step 3

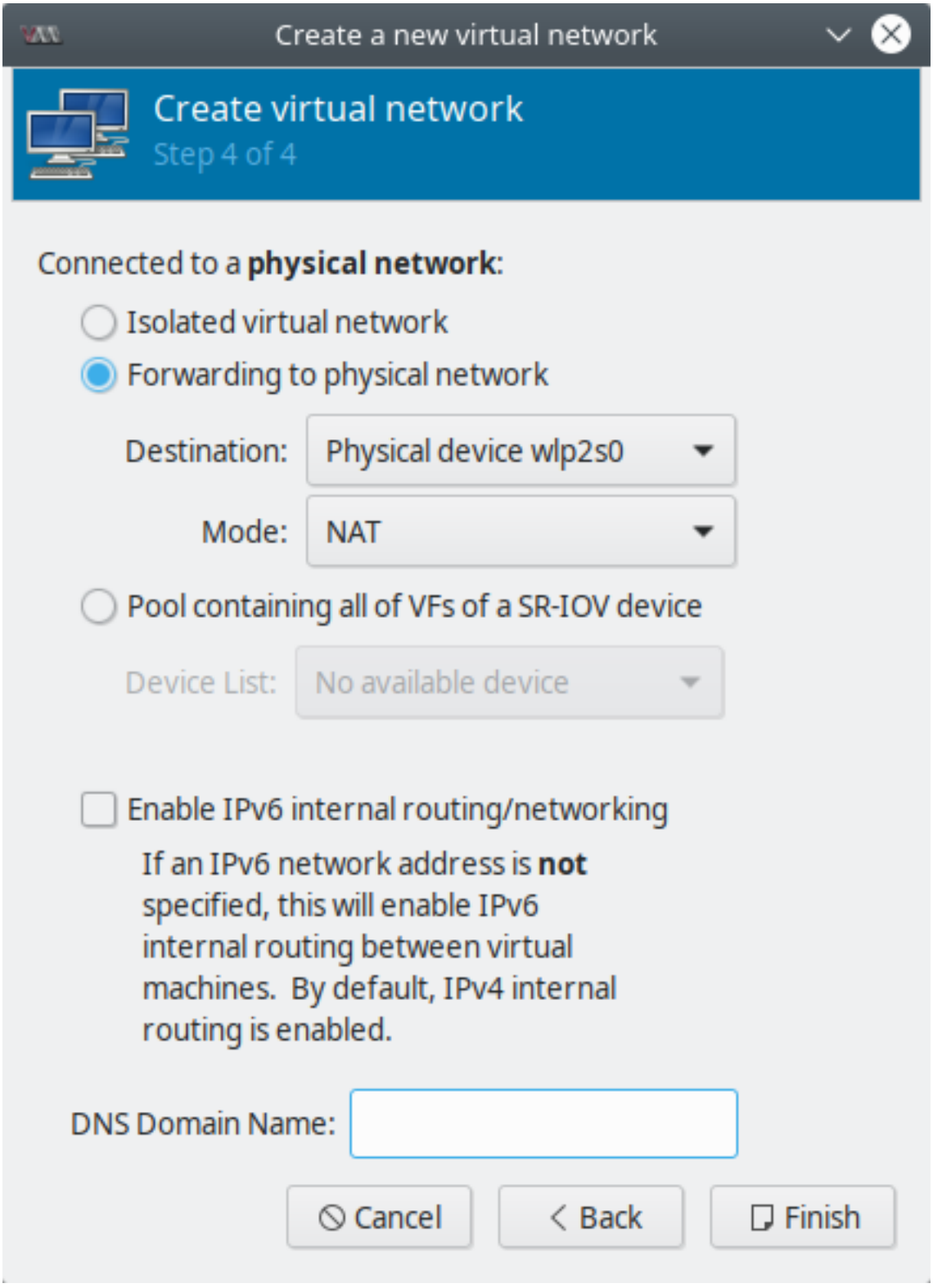


Fig. 20: `virt-manager` - NATed virtual network - step 3

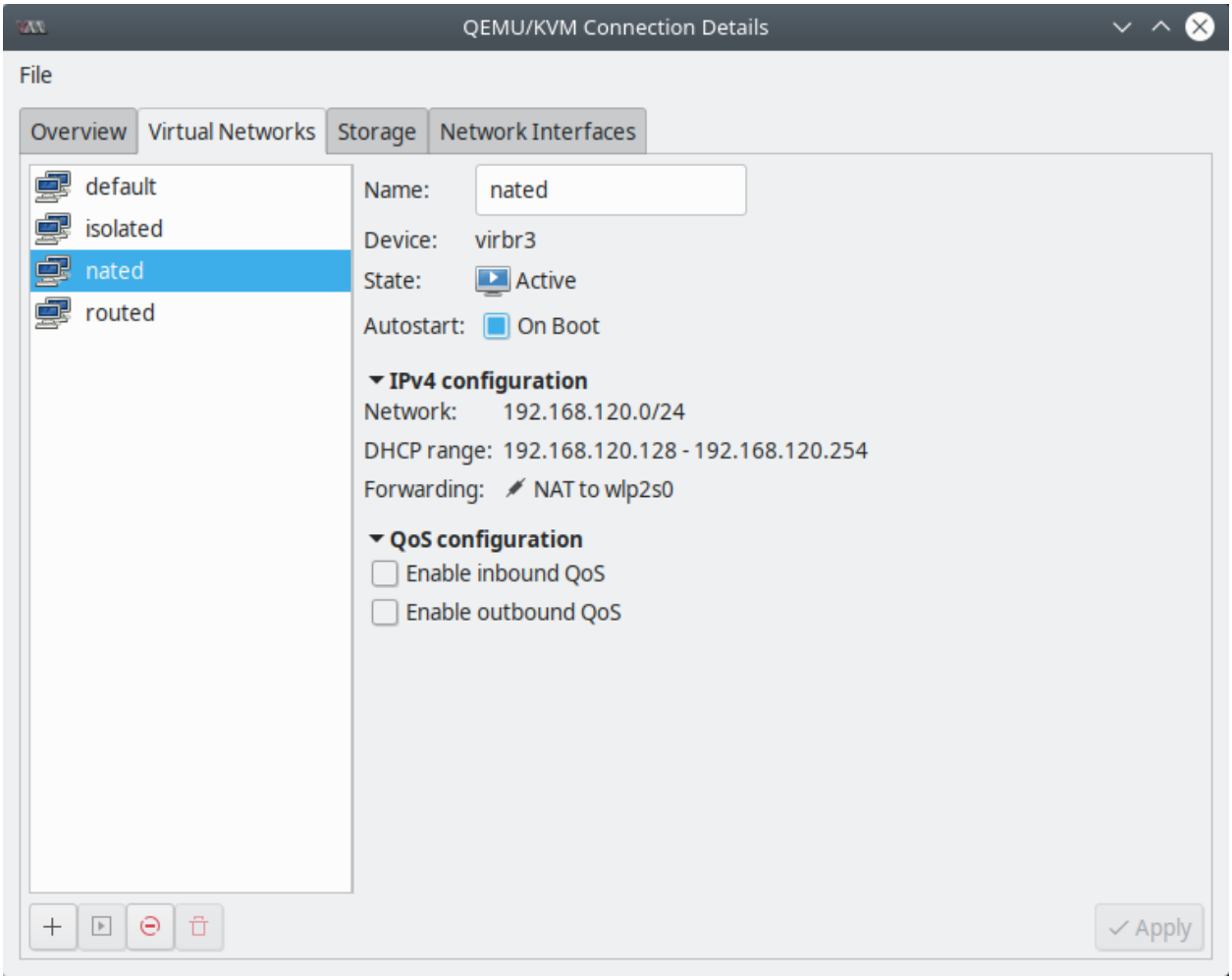


Fig. 21: `virt-manager` - NATed virtual network creada

Iniciar una red con `virsh`

Para iniciar una red virtual usamos `virsh net-start`:

```
$ virsh net-start routed

Network routed started
```

Configurar auto-start de una red con `virsh`

Para activar **auto-start** de una red virtual usamos `virsh net-autostart`:

```
$ virsh net-autostart routed

Network routed marked as autostarted
```

28.2.5 Agregar una interfaz de red virtual a una VM

Para usar una red virtual que hemos creado debemos asociar la interfaz de red virtual de una VM al bridge de la red virtual creada.

Usando `virt-manager`

Podemos agregar una nueva interfaz virtual (o NIC virtual o vNIC) a una VM en `virt-manager` haciendo clic derecho en la VM | *Open* | *Show virtual hardware details* | + *Add Hardware* | *Network* | en *Network source* elegir la red a la que deseamos conectar nuestra VM. La dirección MAC será generada or `libvirt` y el *Device model* como `virtio`. Clic en *Finish*.

Note: Los dispositivos `virtio` están disponibles desde la versión de kernel de Linux 2.6.25.

Usando `virsh`

Para agregar una nueva interfaz virtual (o NIC virtual o vNIC) a una VM usando `virsh` sigamos los siguientes pasos:

1. Obtener el nombre de dominio de la VM (en este caso llamada `centos7.0`) y usar el comando `virsh attach-interface` con ciertos parámetros:

```
$ virsh attach-interface --domain centos7.0 --source isolated --type network --model␣
↪virtio --config --live

Interface attached successfully
```

Hemos conectado un interfaz virtual modelo `virtio` a la VM. La intefaz está usando un red virtual aislada. Hay dos opciones nuevas usadas en este comando:

- `--config`: hace que el cambio sea persistente en la VM (es decir, no se borrará la próxima vez que arranquemos la VM). Quitar esta opción si queremos que el cambio sea temporal hasta el próximo arranque de la VM.
- `--live`: informará a `libvirt` que estamos añadiendo la NIC a una VM que está corriendo. Quitar `--live` si la VM no está corriendo.

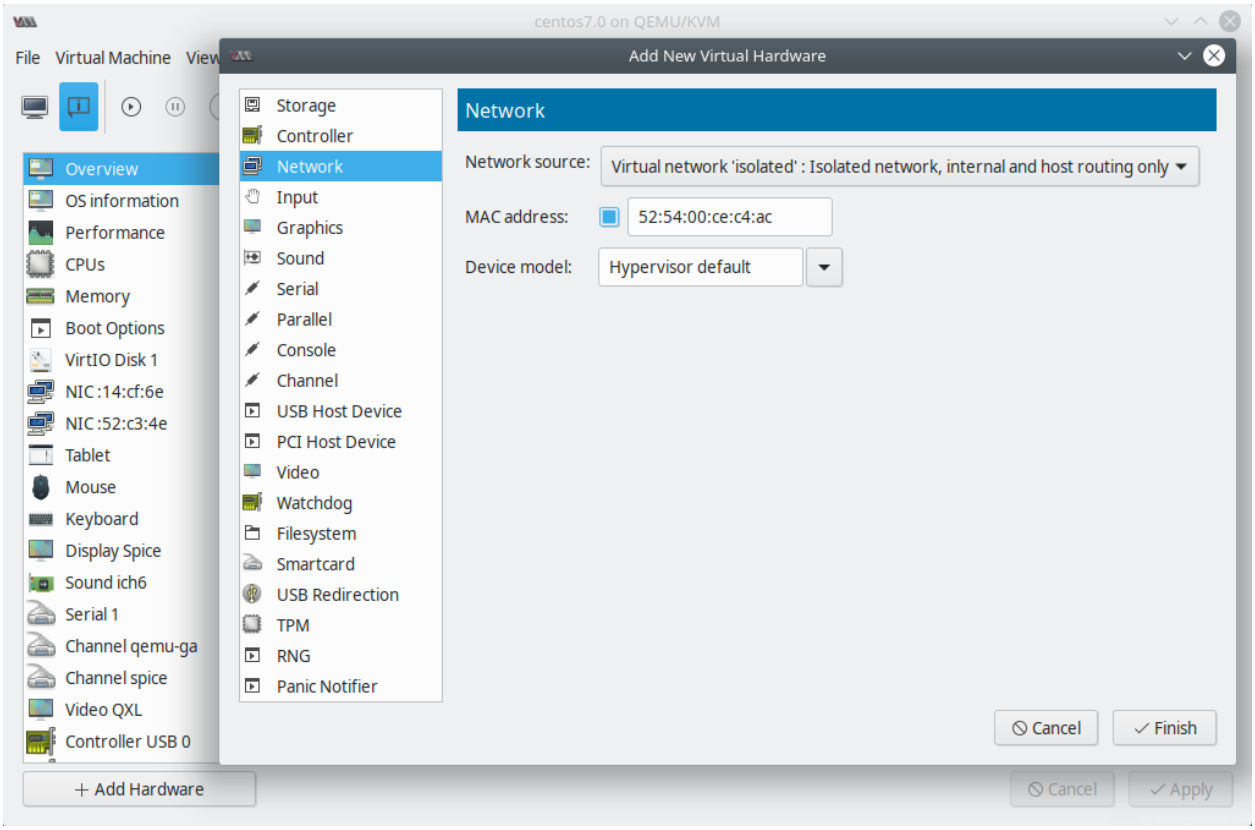


Fig. 22: `virt-manager` - Crear nueva interfaz de red virtual

Note: Una opción extra es `--mac` que puede usarse para añadir una dirección MAC personalizada.

2. Veamos las interfaces añadidas a la VM con `virsh domiflist`:

```
$ virsh domiflist centos7.0
Interface  Type       Source     Model       MAC
-------------------------------------------------------
vnet0      network    default    virtio      52:54:00:a9:f4:12
vnet1      network    isolated   virtio      52:54:00:18:e3:75
```

3. Para eliminar la interfaz añadida a la VM usando `virsh detach-interface`.

```
$ sudo virsh detach-interface --domain centos7.0-2 --type network --mac␣
↪52:54:00:18:e3:75 --config --live
```

Caution: Tener cuidado al usar el parámetro `--live` al momento de quitar una interfaz de red virtual pues podría perder conectividad de red.

28.2.6 Ver propiedades de la red

Para ver las propiedades de una red virtual que hemos creados tenemos distintas opciones , ya sea por `virt-manager` o `virsh`.

Por ejemplo cuando nosotros creamos una red virtual, por debajo, se crea un bridge. Si queremos ver el nombre del bridge creado usaremos cualquiera de estas opciones:

Propiedades de red con `virt-manager`

- `virt-manager`: *Connection details* | *Virtual Networks* | sección *Device* de la red seleccionada.

Propiedades de red con `virsh net-dumpxml`

Cuando generamos una red con `virsh` en base al archivo XML que hemos creado, `libvirt` crea su propio archivo XML en la dirección `/etc/libvirt/qemu/networks/` con el mismo nombre de nuestra red virtual y parámetros adicionales para generar la red.

Usar el comando `virsh net-dumpxml` para obtener detalles de la red y el Linux bridge asociado:

```
$ virsh net-dumpxml isolated
```

```
<network>
    <name>isolated</name>
    <uuid>13fd5536-cad8-43a3-a066-91243719f95b</uuid>
    <bridge name='virbr1' stp='on' delay='0'/>
    <mac address='52:54:00:c4:d1:d7'/>
</network>
```

Encontraremos que `libvirt` ha añadido unos cuantos parámetros adicionales:

- `<uuid>`: un ID único de nuestro bridge

- `<bridge>`: usado para definir los detalles del bridge. Nombre del bridge, estado de STP y Delay. Estos son los mismos comandos que podemos controlar usando el comando `brctl` (`stp` configura el STP y `setfd` configura el DELAY).
- `<mac>`: la dirección MAC del bridge asignada al momento de crearlo

Propiedades de red con `virsh net-info`

```
$ virsh net-info routed

Name:           routed
UUID:           43d8bf8b-3558-4ed0-85d3-66b8de7783ff
Active:         yes
Persistent:     yes
Autostart:      yes
Bridge:         virbr2
```

Interfaces conectadas al bridge

Ahora que conocemos el nombre del bridge veamos las interfaces conectadas a este:

```
$ brctl show virbr1
bridge name     bridge id               STP enabled     interfaces
virbr1          8000.525400c4d1d7       yes             virbr1-nic
                                                        vnet1
                                                        vnet3
```

- La interfaz `vibr1-nic` es creada por `libvirt` cuando inicia el bridge `virbr1`. El propósito de esta interfaz es proveer una dirección MAC consistente y confiable para el bridge `virbr1`. El bridge copia la dirección MAC de la primera interfaz que es añadida al bridge. `virbr1-nic` siempre es la primera interfaz añadida al bridge por `libvirt` y no es eliminada hasta que se borre el bridge.
- `vnet1` y `vnet3` son las interfaces de red virtuales añadidas a sus respectivas VMs.

28.2.7 Editando una red virtual

Editemos una red virtual modo routed y modifiquemos su configuración de enrutamiento para que los paquetes de las VMs sean reenviados a otra interfaz en el host según la reglas de rutas IP especificadas en el host.

Antes de editar la red virtual, primero debemos pararla:

```
$ virsh net-destroy routed

Network routed destroyed
```

Editar la red usando `virsh net-edit`:

```
$ virsh net-edit routed

Network routed XML configuration edited.
```

Este comando hará una copia temporal del archivo de configuración usado por routed en `/tmp` y luego abrirá el editor de texto predeterminado. Editaremos la etiqueta `<forward>`:

- Configuración antigua:

```
<forward dev='wlp2s0' mode='route'>
    <interface dev='wlp2s0'/>
</forward>
```

- Configuración nueva:

```
<forward mode='route' />
```

Verificar la configuración con `net-dumpxml`:

```
$ virsh net-dumpxml routed
```

```
<network>
<name>routed</name>
<uuid>43d8bf8b-3558-4ed0-85d3-66b8de7783ff</uuid>
<forward mode='route'/>
<bridge name='virbr2' stp='on' delay='0'/>
<mac address='52:54:00:76:20:86'/>
<ip address='192.168.10.1' netmask='255.255.255.0'>
    <dhcp>
        <range start='192.168.10.128' end='192.168.10.254'/>
    </dhcp>
</ip>
</network>
```

Luego de verificar la configuración, iniciar la red virtual con `net-start`:

```
$ virsh net-start routed

Network routed started
```

28.2.8 Referencias

- Isolated mode - Libvirt Wiki
- Routed mode - Libvirt Wiki
- NATed mode - Libvirt Wiki

28.3 Bridged Network

Table of Contents

Usando una **Bridged Network** podemos lograr que una VM sea vista por la red como un dispositivo más, sin depender del host donde se encuentra. Es decir, con una **interfaz bridge** la VM podrá pertenecer al mismo segmento de red que el host y tener una IP dentro de ella.

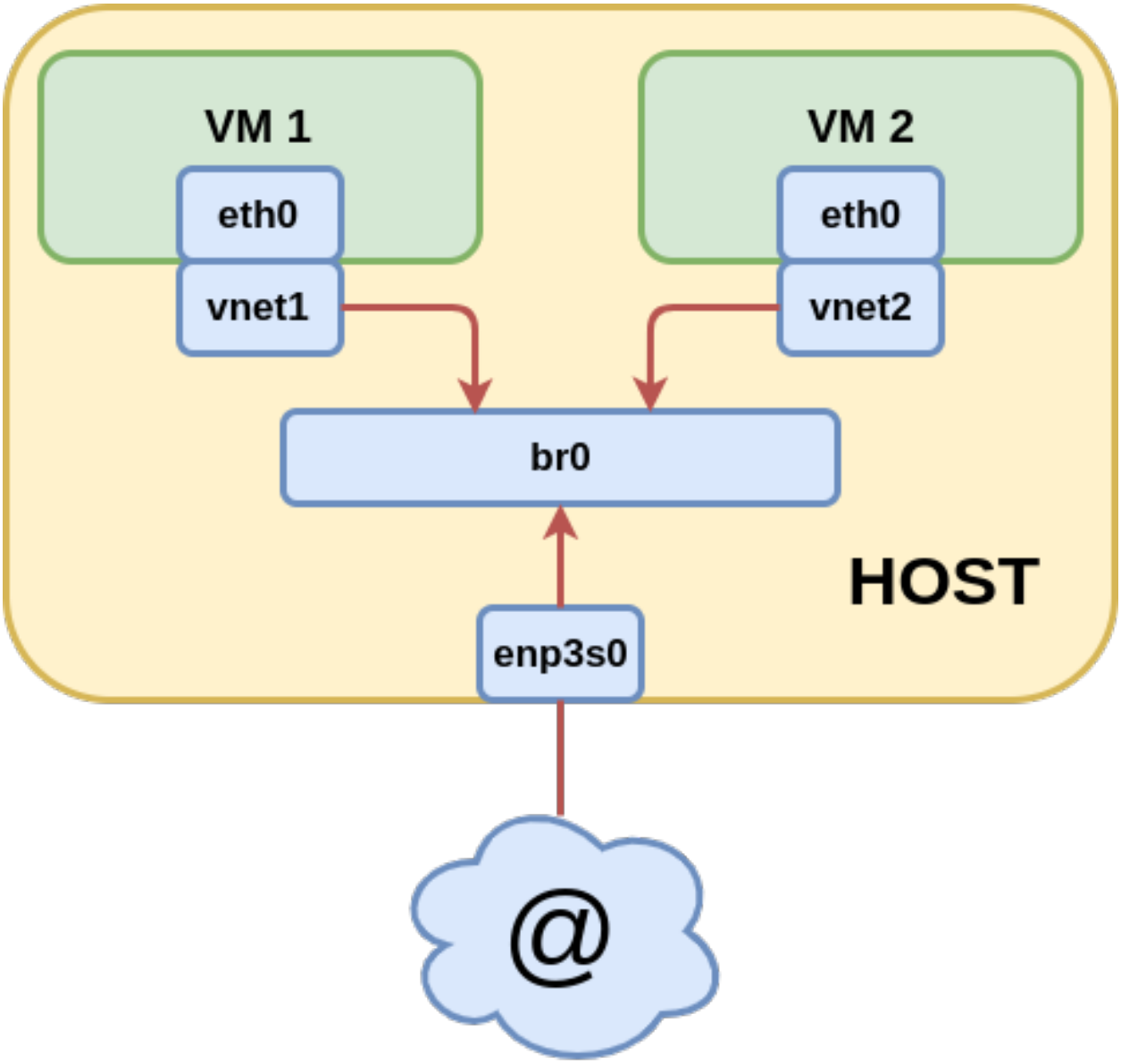


Fig. 23: Esquema de Bridged Network

28.3.1 Host configuration

Método 1 - Usando `iproute2`

En el **host** (máquina física) debemos realizar el siguiente procedimiento:

- Crear el bridge con `ip link`:

```
$ sudo ip link add name br0 type bridge
```

```
$ ip link

31: br0: <NO-CARRIER,BROADCAST,MULTICAST,UP> mtu 1500 qdisc noqueue state DOWN mode␣
↪DEFAULT group default qlen 1000
    link/ether 00:00:00:00:00:00 brd ff:ff:ff:ff:ff:ff
```

- Prender la interfaz del bridge

```
$ sudo ip link set dev br0 up
```

```
$ ip link

31: br0: <NO-CARRIER,BROADCAST,MULTICAST,UP> mtu 1500 qdisc noqueue state DOWN mode␣
↪DEFAULT group default qlen 1000
    link/ether 00:00:00:00:00:00 brd ff:ff:ff:ff:ff:ff
```

- Prender la interfaz física que asociaremos al bridge (sin ninguna configuración/IP):

Important: La interfaz física no deberá tener asignada una IP y no deberá estar siendo usada por otro bridge. **Recomendación**: en una laptop podemos tener la salida a internet por la interfaz inalámbrica y usar la interfaz cableada para asociarla al bridge.

```
$ sudo ip link set dev enp3s0 up
```

- Asociar la interfaz física (`enp3s0`) al bridge creado (`br0`):

```
$ sudo ip link set dev enp3s0 master br0
```

```
$ ip link

2: enp3s0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel master br0 state␣
↪UP mode DEFAULT group default qlen 1000
    link/ether 08:9e:01:54:38:4c brd ff:ff:ff:ff:ff:ff
```

- Para remover la configuración realizada usar:

```
# Desasociar la interfaz al bridge:
$ sudo ip link set dev enp3s0 nomaster
# Eliminar el bridge:
$ sudo ip link delete br0
```

Método 2 - Usando `bridge-utils`

- Crear un nuevo bridge:

```
$ sudo brctl addbr br0
```

- Añadir la interfaz física al bridge:

```
$ sudo brctl addif br0 enp3s0
```

- Verificar el bridge y las intefaces conectadas a él:

```
$ brctl show
bridge name bridge id                   STP enabled interfaces
br2             8000.94c691b241ed   no                  enp3s0
```

- Prender el bridge:

```
$ sudo ip link set dev br0 up
```

```
$ ip link

2: enp3s0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel master br0 state␣
↪UP mode DEFAULT group default qlen 1000
    link/ether 94:c6:91:b2:41:ed brd ff:ff:ff:ff:ff:ff

23: br0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc noqueue state UP group␣
↪default qlen 1000
    link/ether 94:c6:91:b2:41:ed brd ff:ff:ff:ff:ff:ff
    inet6 fe80::96c6:91ff:feb2:41ed/64 scope link
    valid_lft forever preferred_lft forever
```

- Para remover la configuración utilizada:

```
# Apagar la interfaz del bridge:
$ sudo ip link set dev br0 down
# Eliminar el bridge:
$ sudo brctl delbr br0
```

Método 3 - Usando `virt-manager`

- Abrir `virt-manager`, ir a la pestaña *Edit*, opción *Connection Details* :
- En la nueva ventana, clic en el botón +:
- *Interfaz type:* **Bridge**
- Configurar la interfaz de red, especificando las interfaces que se conectarán al bridge:
- Verificar que la interfaz se haya creado:
- En una VM creada o que vayamos a crear se podrá elegir como *Network Source:* **Bridge br0: Host device enp3s0**

28.3.2 Configuración final del host

- Iniciar la VM usando `virt-install` y el tag `--network bridge=br0`:

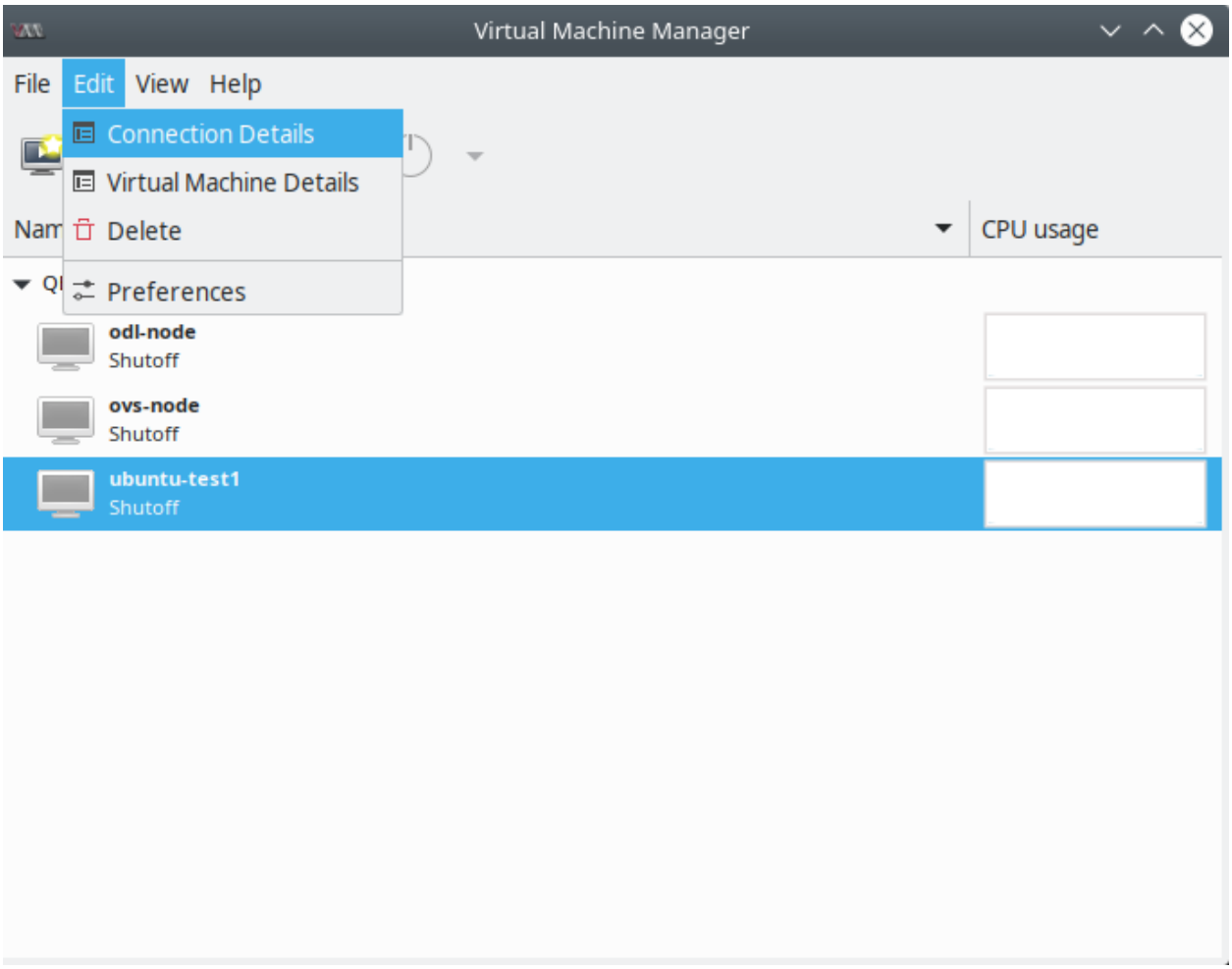


Fig. 24: `virt-manager` - Bridged Network - Step 1

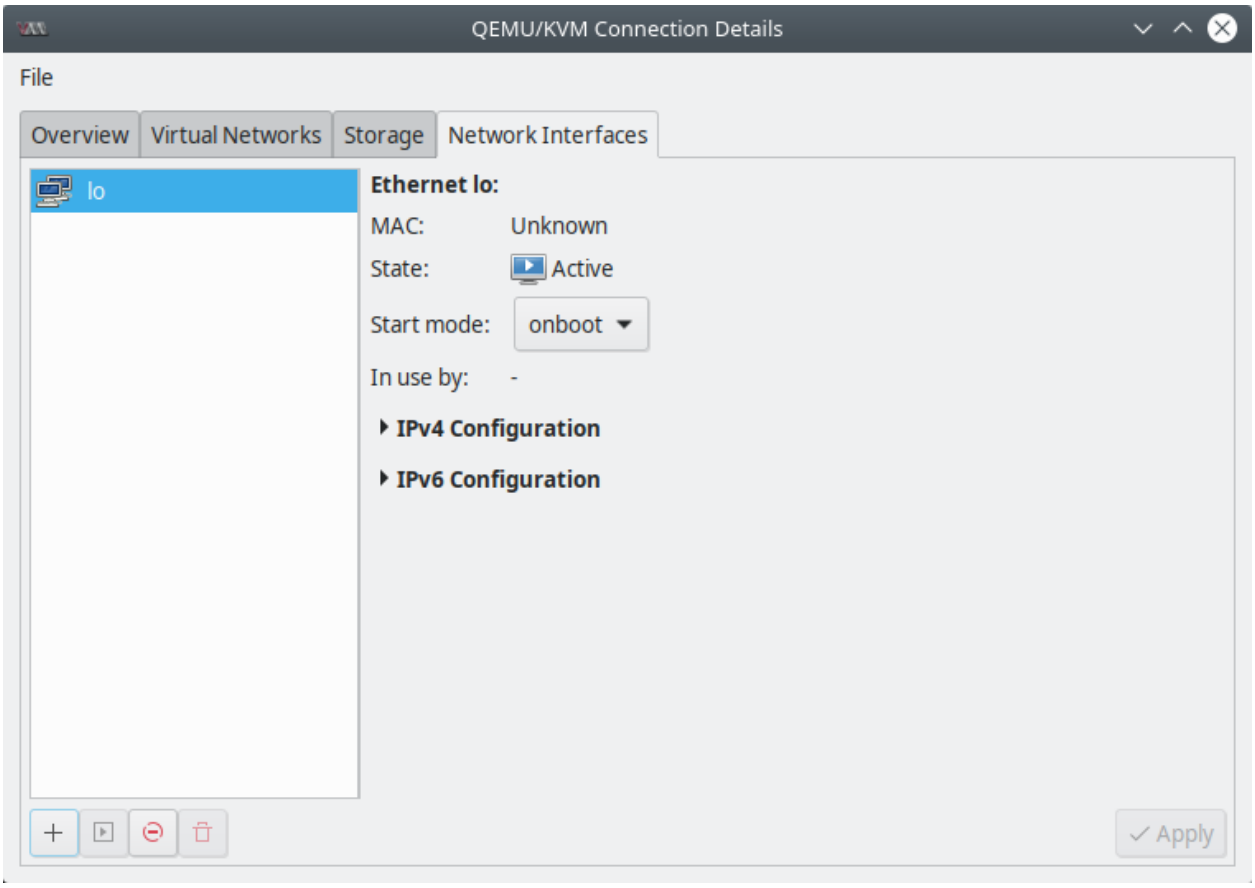


Fig. 25: `virt-manager` - Bridged Network - Step 2

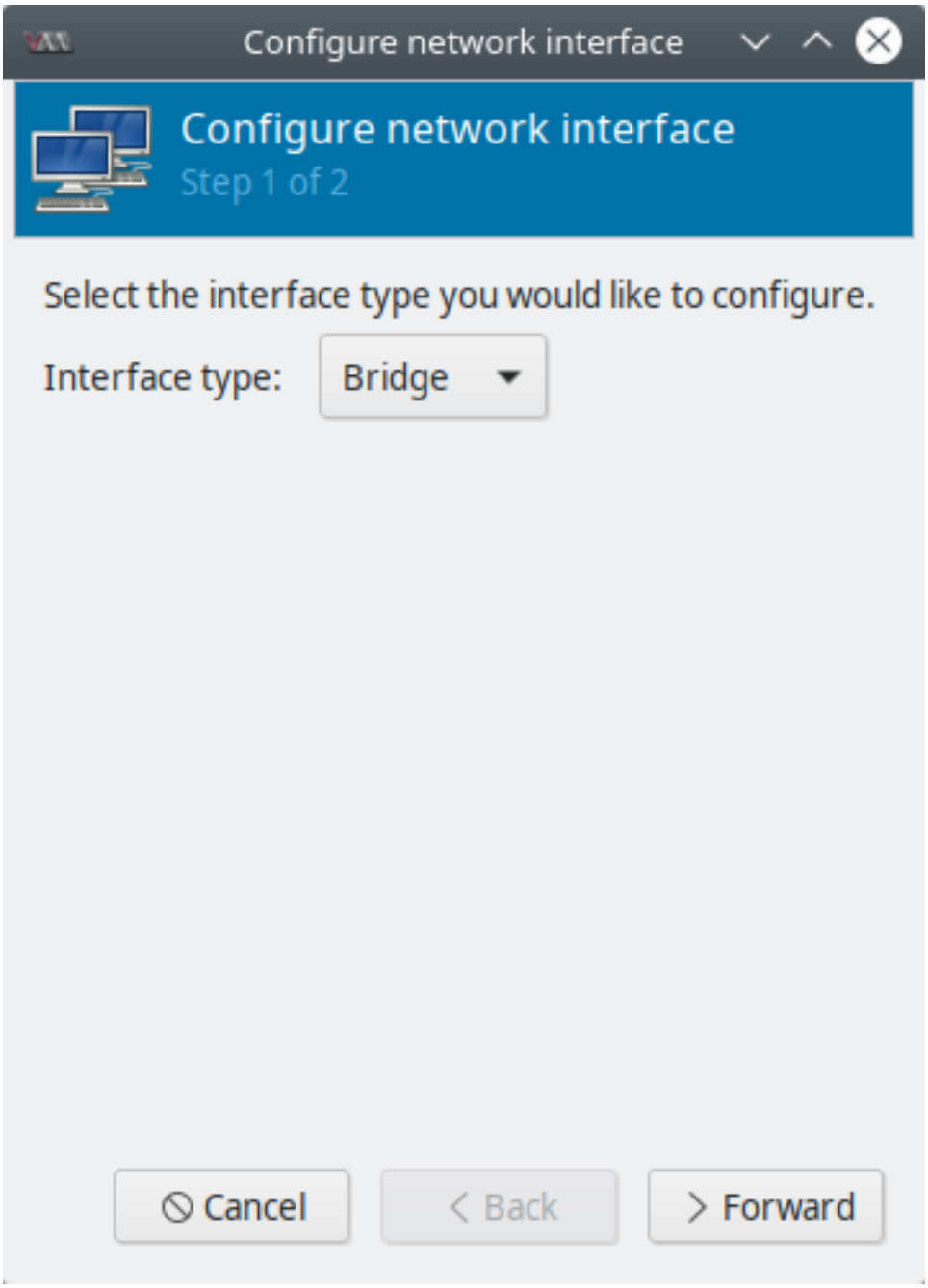


Fig. 26: `virt-manager` - Bridged Network - Step 3

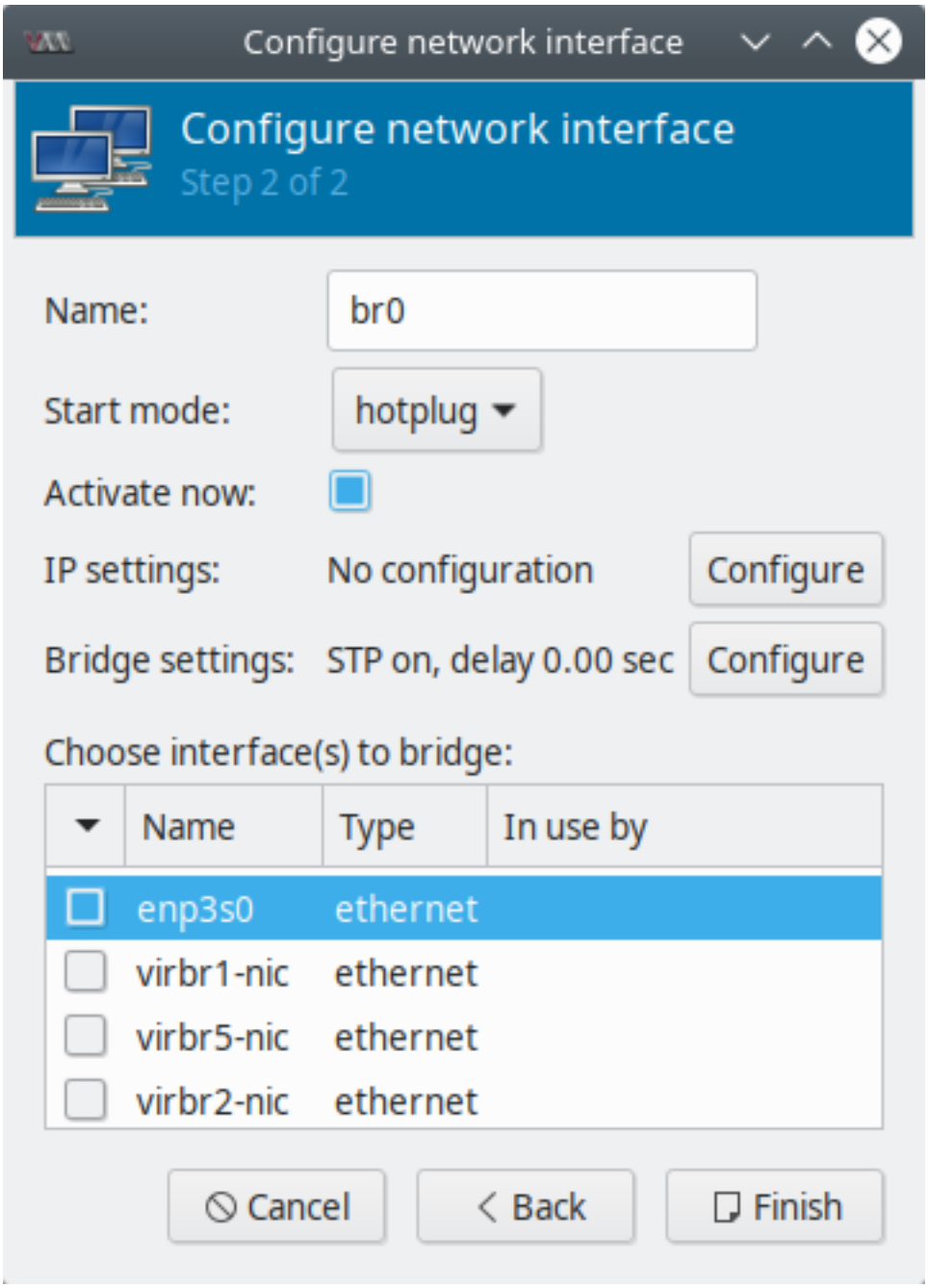


Fig. 27: `virt-manager` - Bridged Network - Step 4

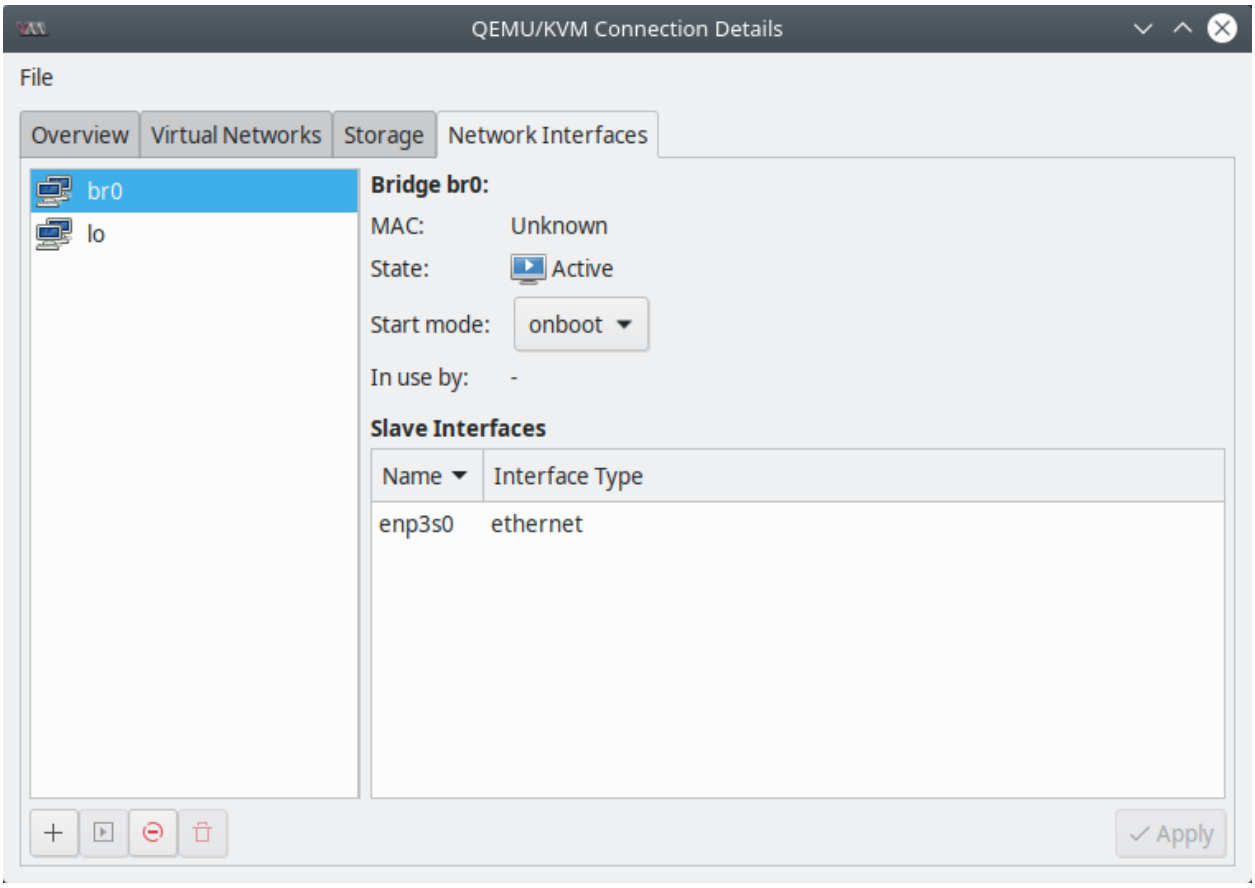


Fig. 28: `virt-manager` - Bridged Network - Step 5

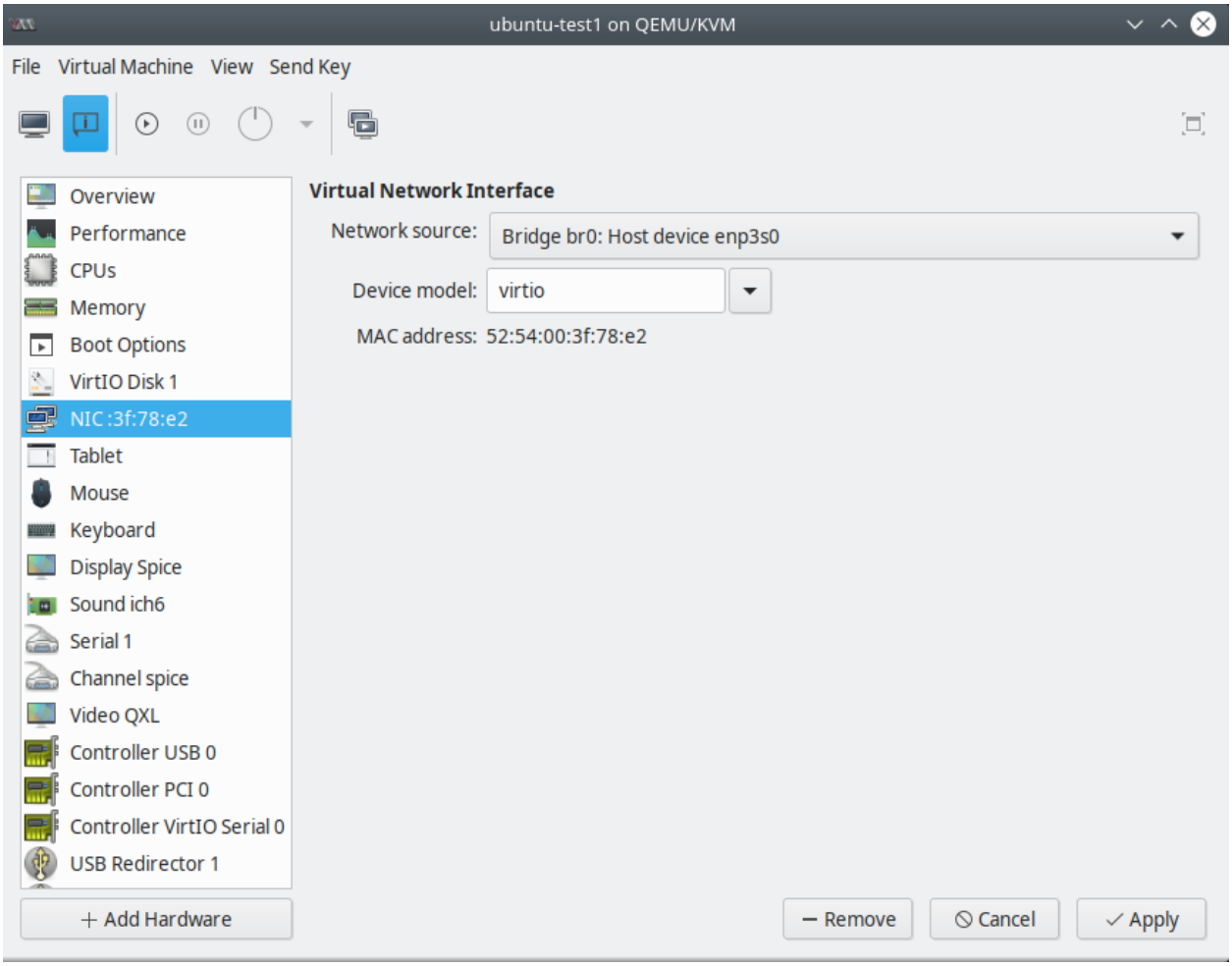


Fig. 29: `virt-manager` - Bridged Network - Step 6

```
sudo virt-install --name=vc3 \
--vcpus=2 \
--memory=4096 \
--network bridge=br0 \
--disk path=/var/lib/libvirt/images/vc3.qcow2 \
--import \
--os-variant=ubuntu18.04 &
```

- Si abrimos la aplicación de `virt-manager` veremos que el nombre de la interfaz que posee la VM es: *Bridge br0: Host device vnet0*
- Luego de asociar la interfaz física al bridge se tendrá problemas de conexión a internet desde el host. Esto se debe a que ahora la interfaz física es un puerto del switch y su configuración IP individual no importa.

No tenemos conexión a Internet pues la interfaz `br0` no tiene IP. Por tanto, hay que remover la configuración IP de la interfaz física `enp3s0` y realizar una configuraración IP en el bridge `br0`:

```
# Remover configuración a interfaz física:
$ sudo dhclient -r enp3s0
# Agregar configuración a interfaz bridge:
$ sudo dhclient -v br0
```

28.3.3 Guest configuration

En el **guest** (VM) la configuración de red será automática según la red a la que se encuentre conectada nuestra interfaz física y el servidor DHCP que exista en la red:

```
$ ip addr

1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default␣
↪qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
    valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host
    valid_lft forever preferred_lft forever
2: ens3: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP group␣
↪default qlen 1000
    link/ether 52:54:00:3f:78:e2 brd ff:ff:ff:ff:ff:ff
    inet 192.168.1.24/24 brd 192.168.1.255 scope global dynamic ens3
    valid_lft 85847sec preferred_lft 85847sec
    inet6 2800:200:e840:125c:5054:ff:fe3f:78e2/64 scope global dynamic mngtmpaddr␣
↪noprefixroute
    valid_lft 3598sec preferred_lft 3598sec
    inet6 fe80::5054:ff:fe3f:78e2/64 scope link
    valid_lft forever preferred_lft forever
```

Podremos crear varias VMs y el servidor DHCP dentro de la red le asignará una IP a cada una de ellas al momento de bootear la VM.

28.3.4 Referencias

- Tutorial Youtube - Setting Up Network Bridge
- Tutorial Network Bridge

- Bridge Interface
- Losing internet access on host with network bridge

CHAPTER 29

Storage Virtualization

29.1 Almacenamiento virtual con Libvirt

Table of Contents

El almacenamiento virtual es abstraído del almacenamiento físico y es asignado a una VM. El almacenamiento es conectado a una VM usando drivers block device emulados o paravirtualizados.

Hay 2 conceptos básicos en la virtualización del almacenamiento: pools y volúmenes. Un **pool de almacenamiento** es una cantidad de almacenamiento reservado para uso de VMs. Los pools de almacenamiento son divididos en volúmenes de almacenamiento. Cada **volumen de almacenamiento** es asignado a una VM guest como un **block device** en un **bus guest**.

Los pools de almacenamiento son administrados usando `libvirt`. La API de `libvirt` puede usarse para consultar la lista de volúmenes dentro de un pool de almacenamiento u obtener información sobre la capcacidad, asignación y almacenamiento disponible en el pool. También podemos consulatar información sobre la capacidad y asignación de un volumen que podrían ser diferentes si se trata de **volúmenes sparse (thin-provisioning)**.

Para pools de almacenamiento soportado, la API de `libvirt` puede emplearse para crear, clonar, redimensionar y eliminar volúmenes. También podemos subir, descargar o limpiar datos de volúmenes de almacenamiento. Una vez que un pool es iniciado, el volumen puede ser asignado a un guest usando el nombre del pool y del volumen en lugar de la ruta del host al volumen en el XML del dominio.

Important: Los pools de almacenamiento pueden ser parados (destroyed), removiendo la abstracción de los datos,

pero manteniendo los datos intactos.

29.1.1 Procedimiento para la creación y asignación de almacenamiento

1. Crear pools de almacenamiento. Referencia: *Pools con clientes de libvirt*
2. Crear volúmenes de almacenamiento. Referencia: *Volúmenes con clientes de libvirt*
    - Creación de volúmenes con: *Volúmenes con clientes de libvirt*
    - Creación de imágenes de disco con: *Imágenes de disco con qemu-img*
    - Creación de imágenes de disco con: *Creando una imagen de disco con dd*
3. Asignar dispositivos de almacenamiento a VMs. Referencia: *Agregar dispositivos de almacenamiento a Guests*

29.1.2 Teoría de almacenamiento virtualizado

El almacenamiento virtualizado suele dividir en dos tipos:

- **Unmanaged storage**: almacenamiento que no es controlado ni monitoreado directamente por `libvirt` y es usado con VMs. Es decir, usar cualquier archivo o bloque disponible en el sistema host como un disco virtual.
- **Managed storage**: almacenamiento controlado y monitoreado por `libvirt` en términos de pools y volúmenes de almacenamiento.

Discos virtuales

Los discos usados para almacenamiento virtual y que son asignados a las VMs son llamados **discos virtuales**. Estos pueden ser de 2 tipos:

- **Preallocated**: un disco virtual preallocated tiene almacenamiento reservado del mismo tamaño que el disco virtual. Al momento de la creación del disco, se asigna todo el espacio requerido y se presenta a la VM sin una capa adicional en medio. Gracias a esto, el rendimiento es mejor que un disco thin-provisioned, pues no requiere asignación de almacenamiento extra durante su uso.
- **Thin-provisioned**: el espacio será asignado al volumen según se necesite. Esto nos permite realizar un overcommitment (sobre-dimensionamiento), bajo la suposición que que todos los discos no serán usados, permitiendo una asignación del recurso de almacenamiento más optimizada. El rendimiento en cargas intensivas de I/O, sin embargo, no es tan bueno como el de un disco pre-allocated.

Información útil:

- Para ver la creación de los 2 tipos de discos virtuales con el comando `dd` ir a: *Creando una imagen de disco con dd*
- Para identificar qué método de asignación usa cierto disco virtual usar `qemu-img info`. Más info: *Imágenes de disco con qemu-img*
- Más información sobre la comparación de discos virtuales preallocated y thin-provisioned: Understanding Virtual Disks - Red Hat Documentation

Comparemos dos disco virtuales:

El parámetro `info` del comando `qemu-img` muestra información sobre una imagen de disco virtual, incluyendo su ruta absoluta, formato de archivo, tamaño virtual y tamaño de disco. Mirando el tamaño de disco y el tamaño virtual, uno puede identificar qué política de asignación de disco está en uso. Veamos la información de los dos discos virtuales que creamos:

```
$ sudo qemu-img info /var/lib/libvirt/images/preallocated-disk.img

image: /var/lib/libvirt/images/preallocated-disk.img
file format: raw
virtual size: 10G (10737418240 bytes)
disk size: 10G

$ sudo qemu-img info /var/lib/libvirt/images/thinprovisioned-disk.img

image: /var/lib/libvirt/images/thinprovisioned-disk.img
file format: raw
virtual size: 10G (10737418240 bytes)
disk size: 0
```

Comparemos el parámetro `disk size` de cada disco. Nos muestra 10G para `preallocated-disk.img`, mientras que para `thinprovisioned-disk.img` es `0`. La diferencia es porque el segundo disco usa un formato thin-provisioned.

- `virtual size`: espacio que el guest observa
- `disk size`: espacio reservado en el host

Si `virtual size` es igual a `disk size`, entonces el disco es preallocated. Si son diferentes el formato del disco es thin-provisioned.

29.1.3 Referencias

- MANAGING STORAGE FOR VIRTUAL MACHINES - Red Hat Documentation
- Understanding Virtual Disks - Red Hat Documentation

29.2 Pools con clientes de `libvirt`

Table of Contents

Un pool de almacenamiento es un archivo, directorio o dispositivo de almacenamiento, administrado por `libvirt` para proveer almacenamiento a VMs. Los pools se dividen en volúmenes de almacenamiento que guardan imáges de VMs o son conectados a VMs como almacenamiento adicional. Varios guest pueden compartir el mismo pool de almacenamiento, permitiendo una mejor asignación de recursos.

Los pools de almacenamiento pueden ser locales o basados en red (compartidos):

- **Pools de almacenamiento locales**: conectados al servidor host directamente. Incluye directorios locales, discos conectados directamente, particiones físicas, y Logical Volume Management (LVM) volume groups en dispositivos locales.
- **Pools de almacenamiento en red (compartidos)**: pools de almacenamiento en red incluyen dispositivos de red compartidos a través de la red usando protocolos estándar. Se requiere este almacenamiento cuando migramos VMs a otro host.

Lista de pools soportados por `libvirt`:

- `-dir`: usa el directorio filesystem para guardar discos virtuales
- `-disk`: usa discos duros físicos para crear discos virtuales
- `-fs`: usa particiones pre-formateadas para guardar discos virtuales
- `-netfs`: usa almacenamiento compartido por red como NFS para discos virtuales
- `-gluster`: permite usar el filesystem gluster para guardar discos virtuales
- `-iscsi`: usa almacenamiento compartido por red ISCSI para guardar disco virtuales
- `-scsi`: usa almacenamiento local SCSI para guardar discos virtuales
- `-lvm`: depende de grupos de volúmenes LVM para guardar discos virtuales
- `-rbd`: permite conectar almacenamiento **Ceph** para discos virtuales

29.2.1 Creando pools

Creando pools con `virt-manager`

A continuación se listan los pasos para la creación de un pool de almacenamiento, de forma general, usando `virt-manager`:

1. Preparar el medio en el que se creará el pool de almacenamiento.

Según el tipo de pool que se use, el procedimiento será distinto. Para ver el procedimiento a seguir de un tipo de pool en específico, buscarlo en *Configuraciones específicas de pools*.

2. Abrir la configuración de almacenamiento

En `virt-manager` ir a *Edit* | *Connection details* | pestaña *Storage*

3. Crear un nuevo pool de almacenamiento

a. Agregar un nuevo pool de almacenamiento (Paso 1):

Hacer clic en el botón + (*Add pool*) bajo la lista de pools. Aparecerá un wizard.

Dar un nombre al pool en el campo *Name*. Seleccionar un tipo de pool de la lista en el campo *Type*.

b. Agregar un nuevo pool de almacenamiento (Paso 2):

Configurar al pool de almacenamiento con los parámetros específicos del tipo de pool. Para más detalles ir a *Configuraciones específicas de pools*.

Note: Para un tipo de pools aparece un check box con la opción *Build Pool*. Si deseamos construir un pool desde el almacenamiento, seleccionar esta opción.

Luego de verificar la configuración clic en el botón *Finish* para crear el pool.

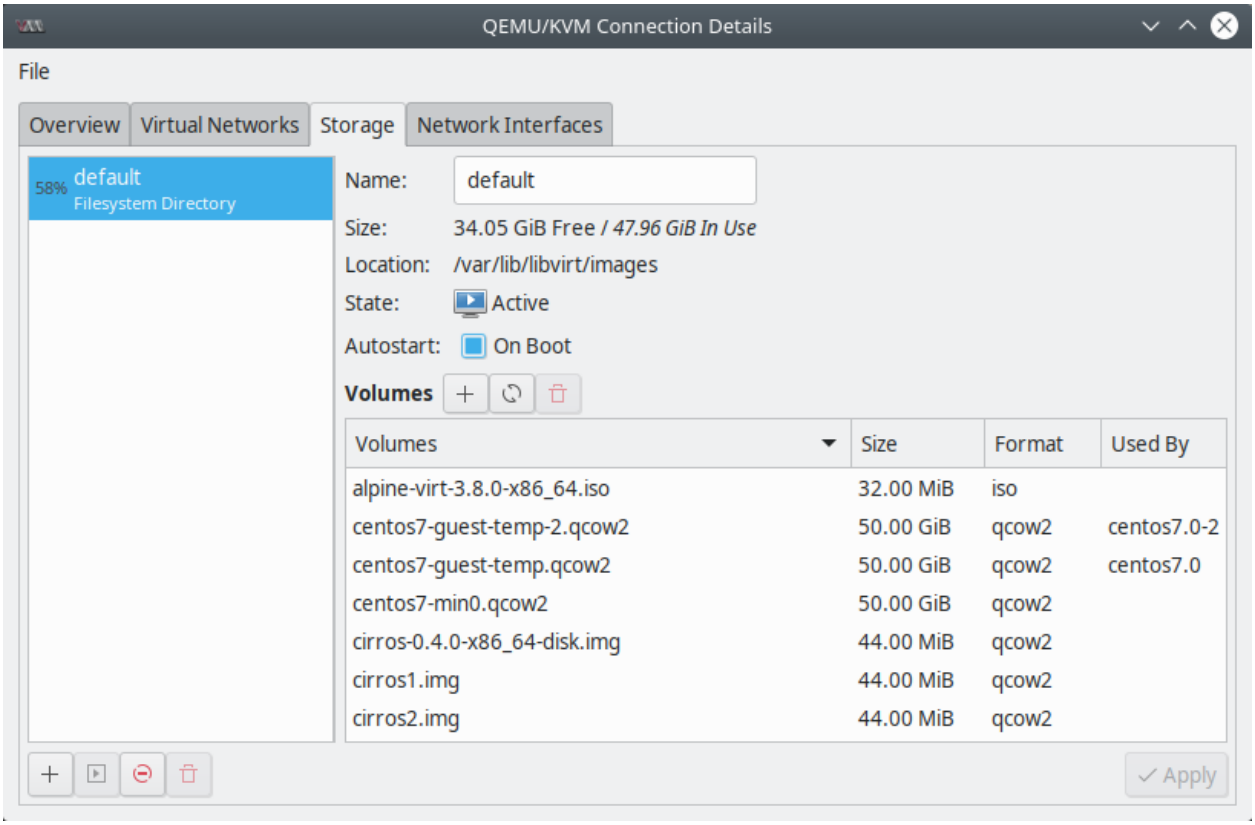


Fig. 1: `virt-manager` - *Connection details* - pestaña *Storage*

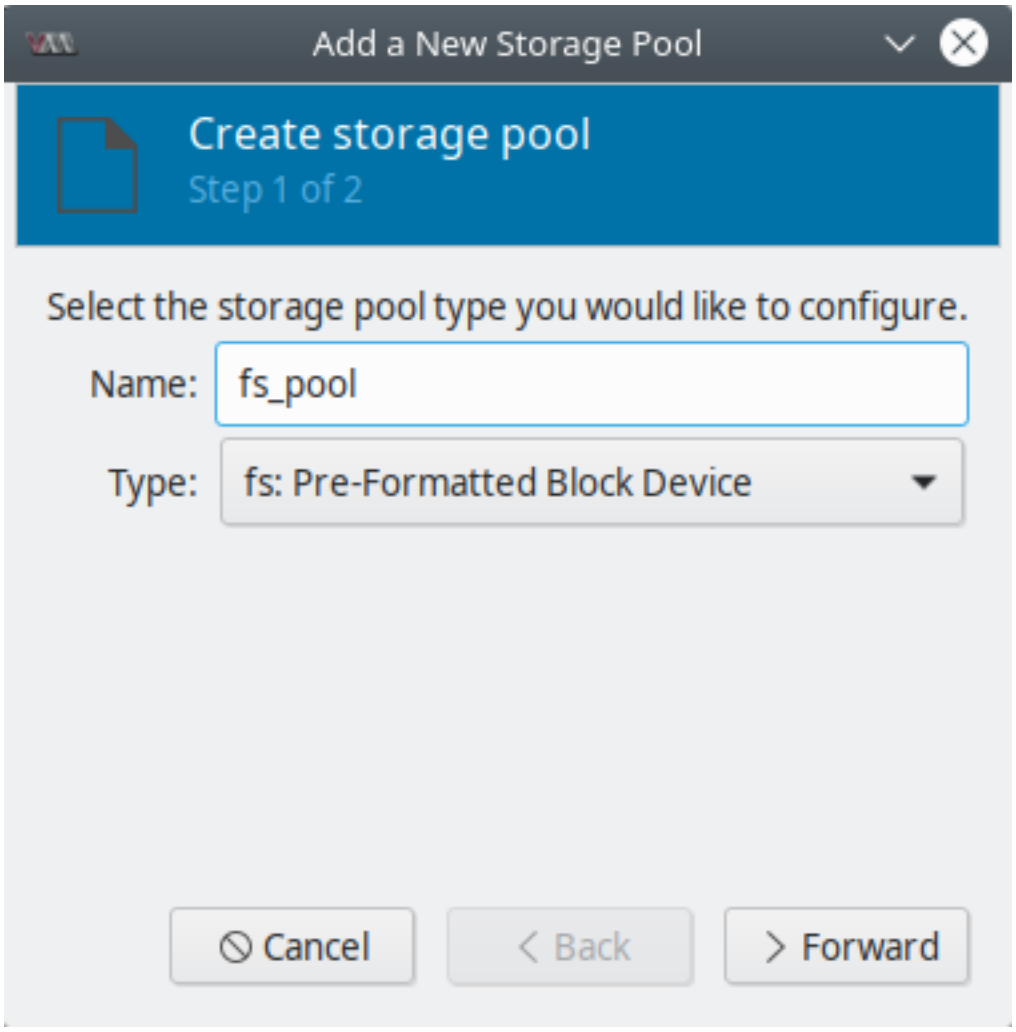


Fig. 2: `virt-manager` - Create storage pool - Step 1

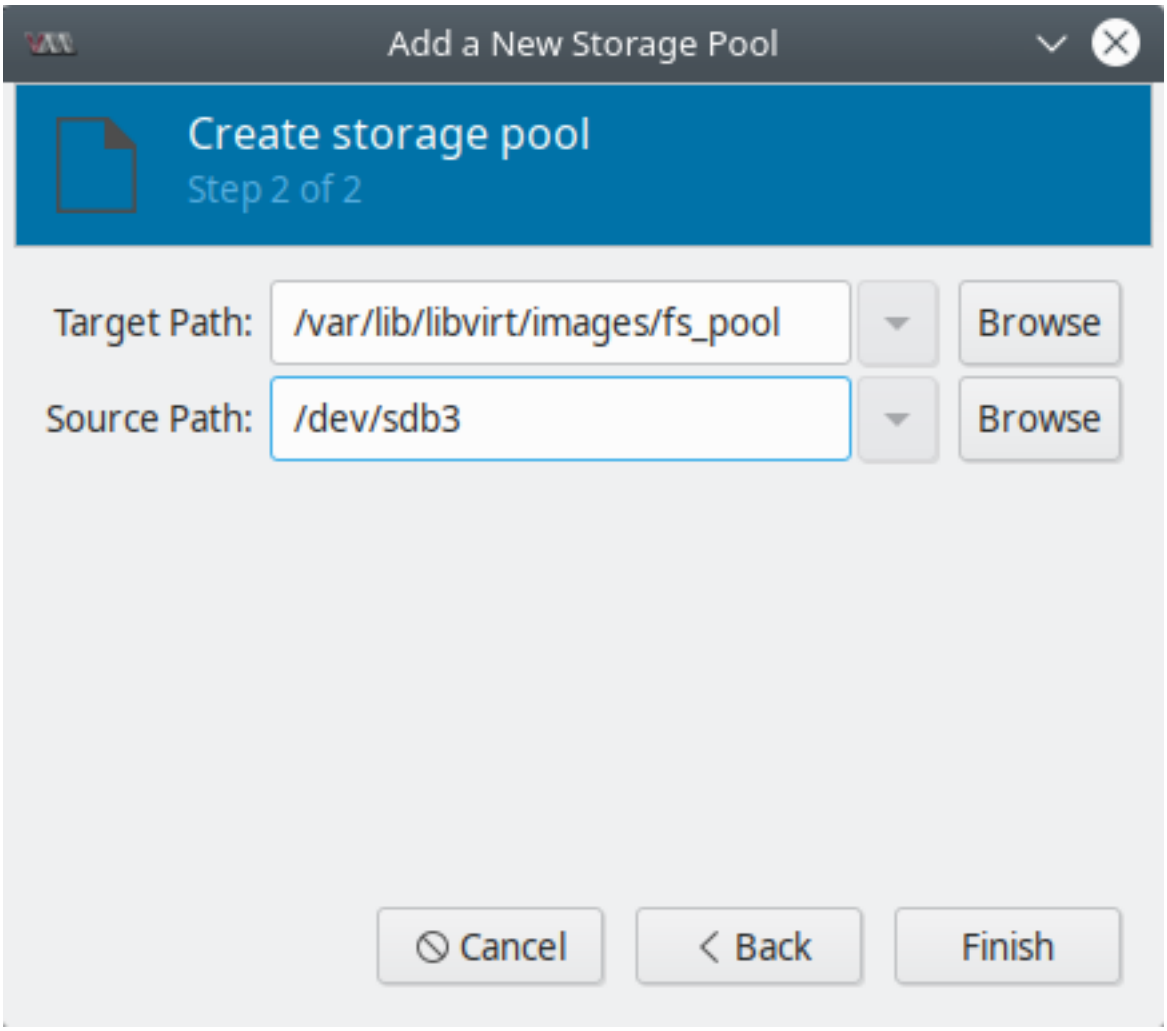


Fig. 3: `virt-manager` - Create storage pool - Step 1

Creando pools con `virsh`

A continuación se listan los pasos para la creación de un pool de almacenamiento, en este caso de tipo **basado en partición**, usando `virsh`:

1. Asegurarse de cumplir los prerequisitos

Los prerequisitos y recomendaciones para los distintos tipos de pool varían. Para ver más a detalle cada tipo de pool ir a *Configuraciones específicas de pools*.

2. Definir el pool

Los pools de almacenamiento pueden ser **persistentes (persistent)** o **transitorio (transient)**. Un pool persistente sigue exstiendo luego de reiniciar el sistema host, mientras que el pool transitorio dura hasta que el sistema se reinicia.

En `virsh`, podemos definir un pool de 2 formas:

- Definir el pool usando un archivo XML

a. Crear un archivo XML temporal (`~/guest_images_fspool.xml`) con la información requerida para el nuevo dispositivo. Según el tipo de pool, los campos dentro del XML cambiarán.

Se muestra un ejemplo de un archivo XML con la definición de un pool tipo basado en partición:

```
<pool type='fs'>
<name>guest_images_fs</name>
<source>
    <device path='/dev/sda3'/>
</source>
<target>
    <path>/guest_images</path>
</target>
</pool>
```

b. Usar el comando `virsh pool-define` o `virsh pool-create` para crear un pool persistente o transitorio, respectivamente:

Persistent (permanente):

```
$ virsh pool-define ~/guest_images_fspool.xml

Pool guest_images_fs defined from ~/guest_images_fspool.xml
```

Transient (temporal):

```
$ virsh pool-create ~/guest_images_fspool.xml

Pool guest_images_fs created from ~/guest_images_fspool.xml
```

c. Eliminar el archivo XML del paso `a`.

- Crear el pool usando `virsh pool-define-as` o `virsh pool-create-as` para crear un pool persistente o transitorio, respectivamente.

Los siguientes comandos crean un pool de almacenamiento (persistent o transient) llamado `guest_images_fs` mapeado a `/dev/sda3` desde el directorio `/guest images`:

Persistent (permanente):

```
$ virsh pool-define-as guest_images_fs fs - - /dev/sda3 - "/guest_images"

Pool guest_images_fs defined
```

Transient (temporal):

```
$ virsh pool-create-as guest_images_fs fs - - /dev/sda3 - "/guest_images"

Pool guest_images_fs created
```

Note: En `virsh`, algunas opciones de nombre son opcionales. Si no serán usados, reemplazarlos por un `-`.

3. Verificar que el pool ha sido creado

Listar los pools con `virsh pool-list`:

```
$ virsh pool-list --all
Name                 State      Autostart
-----------------------------------------
default              active     yes
guest_images_fs      active     no
```

4. Definir el target path del pool

Usar el comando `virsh pool-build` para crear un **pool target path** para pool file system pre-formateado, inicializar el dispositivo origen y definir el formato de los datos:

```
$ virsh pool-build guest_images_fs
Pool guest_images_fs built
```

Note: Hacer build del target path solo es necesario para pools basados en disco (*disk: Physical Disk Device*), file system (*fs: Pre-Formatted Block Device*) y lógicos (*logical: LVM Volume Group*). Si `libvirt` detecta que el formato del dispositivo fuente difiere del tipo de pool, el build falla; a menos que se use la opción `overwrite`.

5. Iniciar el pool

Para poder iniciar un pool y usarlo, ejecutar el comando `virsh pool-start`:

```
$ virsh pool-start guest_images_fs

Pool guest_images_fs started

$ virsh pool-list --all
Name                 State      Autostart
-----------------------------------------
default              active     yes
guest_images_fs      active     no
```

6. Activar autostart

Para que un pool inicie automáticamente cuando el servicio de `libvirtd` inicie usar `virsh pool-autostart`:

```
$ sudo virsh pool-autostart guest_images_fs

Pool guest_images_fs marked as autostarted

# virsh pool-list --all
Name                 State      Autostart
-----------------------------------------
```

(continues on next page)

(continued from previous page)

```
default              active     yes
guest_images_fs      active     yes


$ virsh pool-info guest_images_fs

Name:           guest_images_fs
UUID:           2908eed7-a00b-4629-9ad7-f9bbe00bf2f4
State:          running
Persistent:     no
Autostart:      yes
Capacity:       82,00 GiB
Allocation:     38,62 GiB
Available:      43,38 GiB

$ mount | grep /guest_images
/dev/sda3 on /guest_images type ext4 (rw)
```

Important: Como vemos en el último comando con este tipo de pool (basado en filesystem) se tiene montada una partición del disco (**source**) en nuestro directorio del pool (**target path**).

29.2.2 Configuraciones específicas de pools

Según el tipo de pool de almacenamiento que se planea crear, el procedimiento para preparar el medio donde el pool será creado varía.

Pool basado en directorio

Parámetros:

Tabla con los parámetros requeridos para crear un **pool basado en directorio** con 3 distintos métodos: XML, `pool-define-as`, `virt-manager`

Descripción	XML	`pool-define-as`	`virt-manager`
Tipo de pool	`<pool type='dir'>`	`[type]` dir	*Type: dir: Filesystem Directory*
Nombre de pool	`<name>nombre_pool</name>`	`[name]` *nombre_pool*	*Name*

Target path del pool.
Ruta usada para el pool.
	`<target>`
	`  <path>target_path`
	`↪</path>`
	`</target>`
		`--target` *target_path*	*Target Path*

Ejemplos:

- Pool con archivo XML. Archivo XML para definir un pool basado en el directorio `/guest_images`:

```
<pool type='dir'>
<name>mydirectorypool</name>
```

(continues on next page)

(continued from previous page)

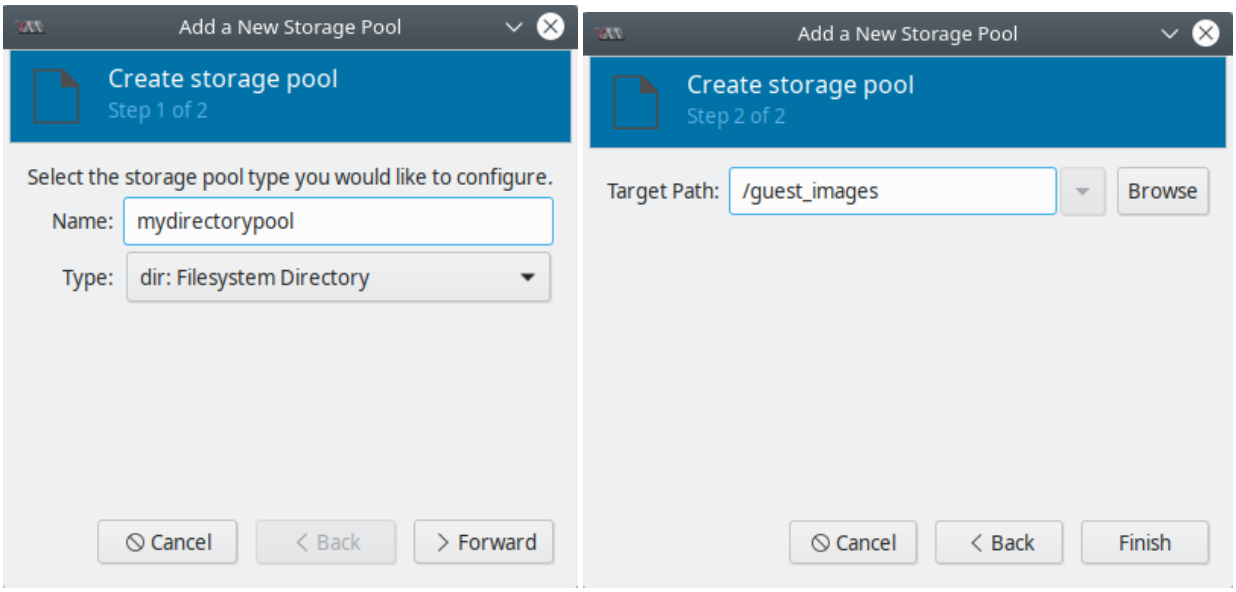
```
<target>
    <path>/guest_images</path>
</target>
</pool>
```

- Pool con `pool-define-as`

  $ sudo virsh pool-define-as dirpool dir –target "/guest_images" Pool FS_directory defined

- Pool con `virt-manager`

Pasos a realizar con `virt-manager` para crear un nuevo pool: ir a *Edit* | *Connection details* | pestaña *Storage* | clic en el botón + (*Add pool*) bajo la lista de pools.



Pool basado en disco

Recomendaciones:

Caution:

- Según la versión de libvirt, dedicar un disco a un pool de almacenamiento puede reformatear y eliminar todos los datos guardados en el disco.
- Los guests no deben tener acceso para escribir en todo el disco o bloque (por ejemplo `/dev/sdb`); en cambio, usar particiones (por ejemplo, `/dev/sdb1`) o volúmenes LVM.

Pre-requisitos:

Note: Estos requisitos solo son válidos si no vamos a usar `virsh pool-build`.

Antes de crear un pool basado en disco en el disco del host, el disco debe ser renombrado con un disk label *GUID Partition Table (GPT)*. Los labels de disco GPT permiten crear hasta 128 particiones en cada dispositivo.

```
$ parted /dev/sdb
GNU Parted 2.1
Using /dev/sdb
Welcome to GNU Parted! Type 'help' to view a list of commands.
(parted) mklabel
New disk label type? gpt
(parted) quit
Information: You may need to update /etc/fstab.
```

Ahora ya estamos listo para seguir con la definición del pool.

Parámetros:

Tabla con los parámetros requeridos para crear un **pool basado en disco** con 3 distintos métodos: XML, `pool-define-as`, `virt-manager`

Descripción	XML	`pool-define-as`	`virt-manager`
Tipo de pool	`<pool type='disk'>`	`[type]` disk	*Type: disk: Physical Disk Device*
Nombre de pool	`<name>nombre_pool</name>`	`[name]` *nombre_pool*	*Name*

Source path del pool. Ruta del dispositivo de almacenamiento.	`<source>`
	`  <device path=/dev/sdb/>`
	`</source>`
		`--source-dev` *path_to_disk*	*Source Path*

Target path del pool. Ruta usada para el pool.	`<target>`
	`  <path>target_path</path>`
	`</target>`
		`--target` *target_path*	*Target Path*

Ejemplos:

- Pool con archivo XML. Archivo XML para definir un pool basado en el disco `/dev/sdb`:

```
<pool type='disk'>
<name>phy_disk</name>
<source>
    <device path='/dev/sdb'/>
    <format type='gpt'/>
</source>
<target>
    <path>/dev</path>
</target>
</pool>
```

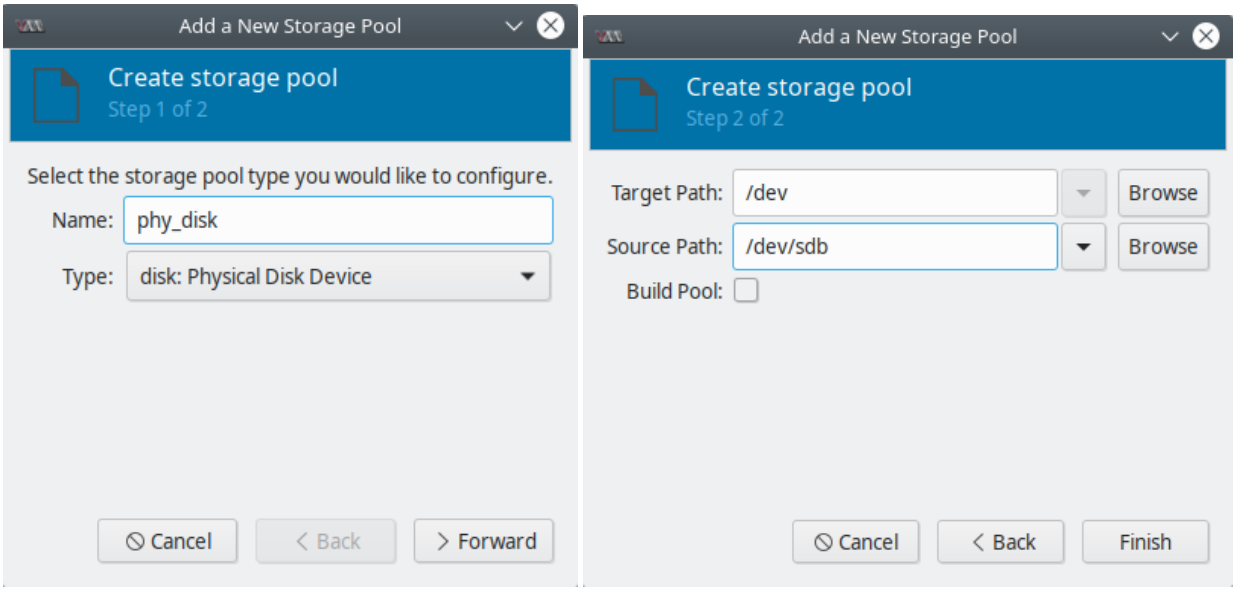
- Pool con `pool-define-as`

```
$ sudo virsh pool-define-as phy_disk disk gpt --source-dev=/dev/sdb --target /dev

Pool phy_disk defined
```

- Pool con `virt-manager`

Pasos a realizar con `virt-manager` para crear un nuevo pool: ir a *Edit* | *Connection details* | pestaña *Storage* | clic en el botón + (*Add pool*) bajo la lista de pools.



Pool basado en filesystem

Recomendaciones:

> **Caution:** No usar este procedimiento para asignar un disco completo como pool (por ejemplo `/dev/sdb`). No se le debe dar acceso de escritura a los guests a discos completos. Usar este método solo para asignar particiones (por ejemplo `/dev/sdb1`) a pools.

Pre-requisitos:

> **Note:** Estos requisitos solo son válidos si no vamos a usar `virsh pool-build`.

Para crear un pool de una partición, formatear el filesystem a **ext4**:

```
$ sudo mkfs.ext4 /dev/sdc1
```

Ahora ya estamos listo para seguir con la definición del pool.

Parámetros:

Tabla con los parámetros requeridos para crear un **pool basado en filesystem** con 3 distintos métodos: XML, `pool-define-as`, `virt-manager`

Descripción	XML	`pool-define-as`	`virt-manager`
Descripción	XML	`pool-define-as`	`virt-manager`
Tipo de pool	`<pool type='fs'>`	`[type]` fs	*Type: fs: Pre-Formated Block Device*
Nombre de pool	`<name>nombre_pool</name>`	`[name]` *nombre_pool*	*Name*
Source path del pool. Ruta de la partición.	`<source> <device path=/dev/sdc1>`	`--source-dev` *path_to_partition*	*Source Path*
Tipo de filesystem (p. ej. ext4)	`<format type='fs_type'/> </source>`	`--source-format` *fs-format*	N/A
Target path del pool. Ruta usada para el pool.	`<target> <path>target_path</path> </target>`	`--target` *target_path*	*Target Path*

Ejemplos:

- Pool con archivo XML. Archivo XML para definir un pool basado en partición `/dev/sdc1`:

```
<pool type='fs'>
<name>guest_images_fs</name>
<source>
    <device path='/dev/sdc1'/>
    <format type='auto'/>
</source>
<target>
    <path>/guest_images</path>
</target>
</pool>
```

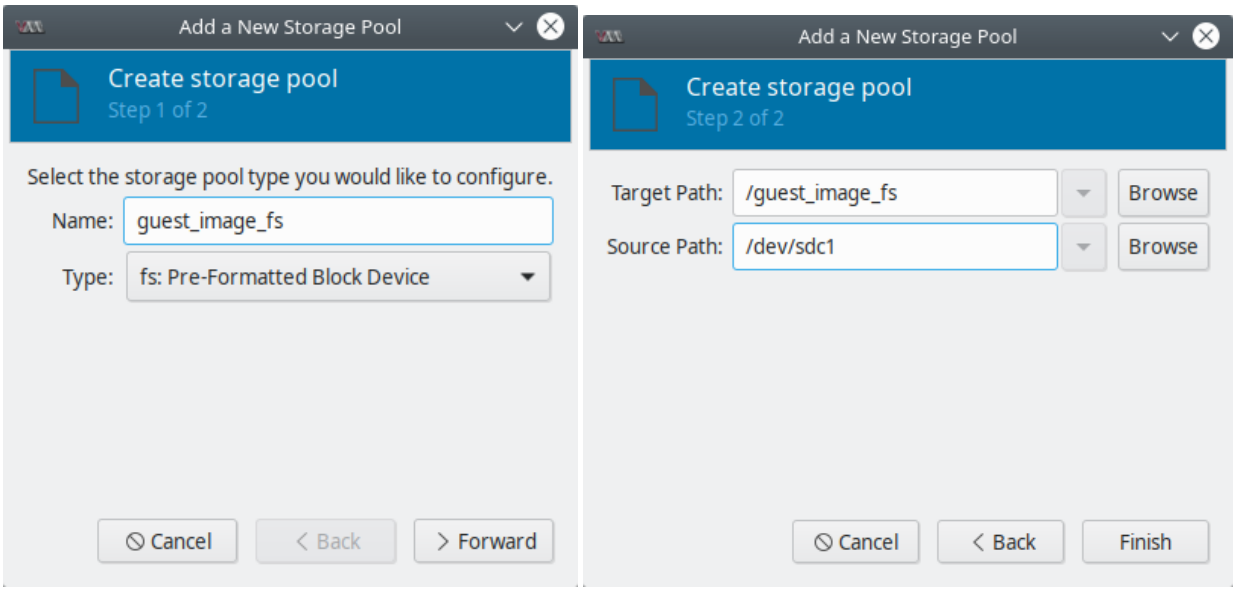
- Pool con `pool-define-as`

```
$ virsh pool-define-as guest_images_fs fs --source-dev /dev/sdc1 --target /guest_
↪images

Pool guest_images_fs defined
```

- Pool con `virt-manager`

Pasos a realizar con `virt-manager` para crear un nuevo pool: ir a *Edit* | *Connection details* | pestaña *Storage* | clic en el botón + (*Add pool*) bajo la lista de pools.



29.2.3 Referencias

- USING STORAGE POOLS - Red Had Documentation

29.3 Volúmenes con clientes de `libvirt`

Table of Contents

A través de los clientes de `libvirt` (`virsh` y `virt-manager`) podemos crear, configurar y eliminar volúmenes de almacenamiento.

Los pools de almacenamiento se dividen en **volúmenes de almacenamiento (storage volumes)**. Los volúmenes de almacenamiento son abstracciones de particiones físicas, volúmenes lógicos LVM, imágenes de disco basada en archivo y demás tipos de almacenamiento soportados por `libvirt`. Los volúmenes de almacenamiento se presentan a las VMs guest como dispositivos de almacenamiento locales sin importar el hardware subyacente.

Luego de la creación de un volumen, ya sea con `virsh` o con `virt-manager`, podemos añadirlos como dispositivos de almacenamiento a las VMs guest usando estas mismas herramientas.

29.3.1 Creando volúmenes

Creando volúmenes con `virt-manager`

1. En `virt-manager` ir a *Edit* | *Connection details* | pestaña *Storge*. El panel izquierdo de la ventana muestra una lista de pools de almacenamiento.
2. Seleccionar el pool de almacenamiento donde queramos crear el volumen de almacenamiento. Se listarán a la derecha todos los volúmenes configurados para ese pool.

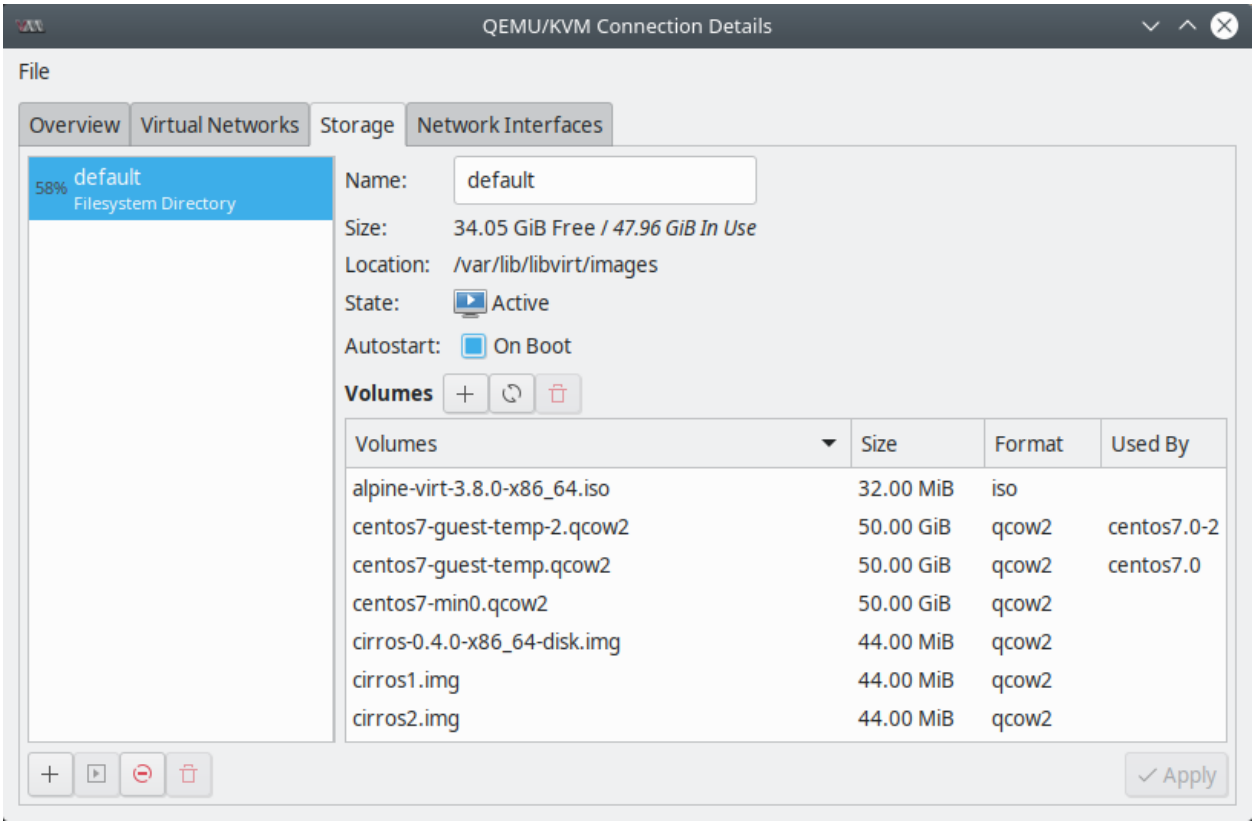


Fig. 4: `virt-manager` - *Connection details* - pestaña *Storage*

3. Añadir un nuevo volumen haciendo clic en el botón + que se encuentra sobre la lista de volúmenes.
4. Ingresar los datos del volumen de almacenamiento:

- Ingresar un nombre para el volumen de almacenamiento.
- Seleccionar un formato para el volumen de almacenamiento de la lista de *Formats*: raw, qcow, qcow2, qed, vmdk, vpc, vdi

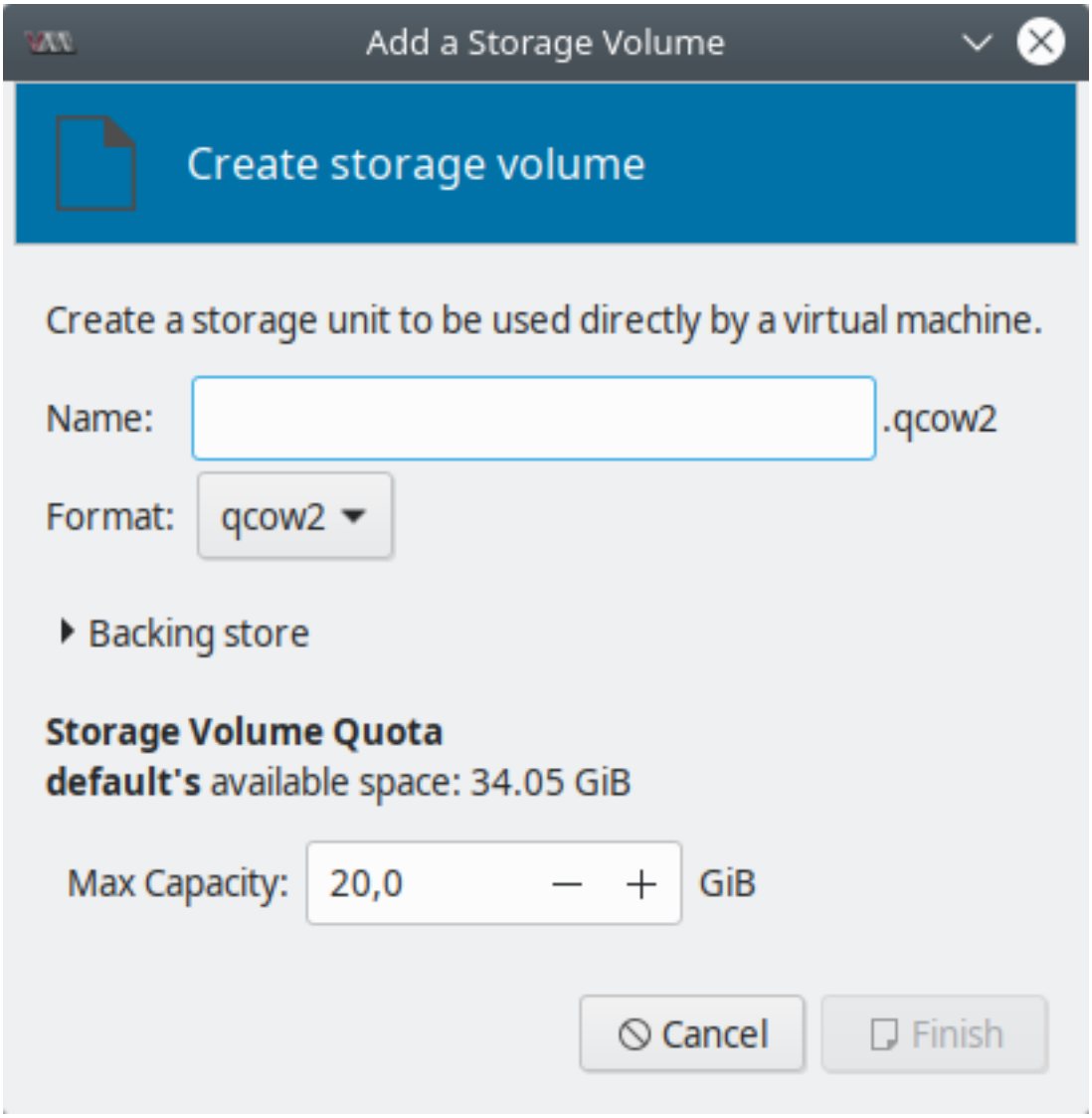


Fig. 5: `virt-manager` - Crear un volumen de almacenamiento

- Ingresar el tamaño máximo para el volumen de almacenamiento en el campo *Max Capacity*.

Clic en *Finish*, luego aparecerá el nuevo volumen la lista de volúmenes del pool seleccionado.

Creando volúmenes con `virsh`

Para crear un volumen de almacenamiento dentro de un pool de almacenamiento usando `virsh` podemos usar cualquiera de los siguientes métodos:

Con un archivo XML

1. Crear un archivo XML temporal que contenga la información de volumen requerida para el nuevo dispositivo, conteniendo campos como:

- `name` - nombre del volumen de almacenamiento
- `allocation` - asignación de almacenamiento total para el volumen
- `capacity` - capacidad lógica del volumen. Si el volumen es disperso (sparse o thin-provisioned), este valor puede difererir del valor de `allocation`.
- `target` - ruta del volumen en el sistema host

Ejemplo de archivo XML con la definición del volumen de almacenamiento.

```
<volume>
    <name>volume1</name>
    <allocation>0</allocation>
    <capacity>10G</capacity>
    <target>
        <path>/var/lib/libvirt/images/sparse.img</path>
    </target>
</volume>
```

Guardarmos el archivo en ~/volume.xml

2. Usar el comando `virsh vol-create` para crear el volumen de almacenamiento basado en el archivo XML:

```
$ sudo virsh vol-create pool1 ~/volume.xml

Vol volume1 created
```

3. Eliminar el archivo XML que creamos en el primer paso.

Con `virsh-vol-create-as`

Para crear un volumen de almacenamiento usamos:

```
$ sudo virsh vol-create-as pool1 volume1 10G --allocation 0
```

Clonando un volumen existente

Con el comando virsh vol-clone podemos clonar un volumen de almacenamiento existente. Se debe especificar el pool de almacenamiento que contiene el volumen ha clonar y el nombre del nuevo volumen:

```
$ sudo virsh vol-clone --pool pool1 volume1 clone1
```

29.3.2 Obtener información de un volumen con `virsh`

Para obtener información extra de un volumen de almacenamiento ubicando en un pool usar:

```
$ virsh vol-info --pool pool1 volume1
Name:           volume1
Type:           file
Capacity:       10,00 GiB
Allocation:     0,00 B
```

29.3.3 Listar volúmenes con `virsh`

Para listar todos los volúmenes de almacenamiento ubicados en un pool usar:

```
$ virsh vol-list default

Name            Path
----------------------------------------
volume1         /home/vms/pool1/volume1
volume2         /home/vms/pool1/volume2
clone1          /home/vms/pool1/clone1
```

29.3.4 Eliminando volúmenes

Eliminando volúmenes con `virt-manager`

1. En `virt-manager` ir a *Edit* | *Connection details* | pestaña *Storge*. El panel izquierdo de la ventana muestra una lista de pools de almacenamiento.
2. Seleccionar el pool de almacenamiento donde se encuentra el volumen de almacenamiento que vamos a eliminar.
3. Seleccionar el volumen de almacenamiento y hacer clic en el botón *Delete volume* (ícono de tacho de basura o una *X*) que se encuentra sobre la lista de volúmenes.

Eliminando volúmenes con `virsh`

Para eliminar un volumen de almacenamiento especificamos el nombre de dicho volumen y el pool de almacenamiento que lo contiene:

```
$ sudo virsh vol-delete volume1 --pool pool1

Vol volume1 deleted
```

29.3.5 Referencias

- USING STORAGE VOLUMES - Red Hat Documents

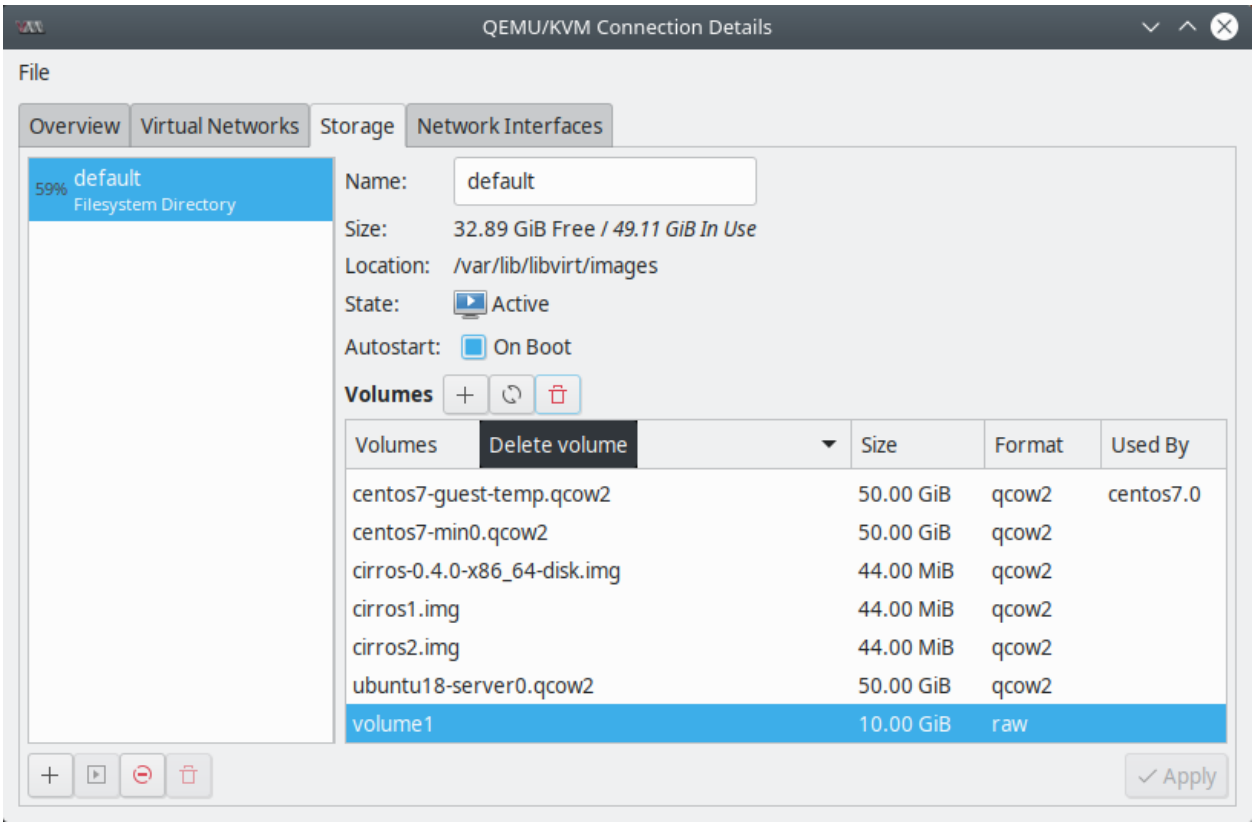


Fig. 6: `virt-manager` - Eliminar un volumen de almacenamiento

29.4 Agregar dispositivos de almacenamiento a Guests

Table of Contents

Podemos añadir dispositivos de almacenamiento a las VMs usando `virsh` o `virt-manager`

29.4.1 Agregando almacenamiento con `virt-manager`

1. Abrir `virt-manager` y seleccionar la máquina virtual a la que deseamos agregarle el volumen de almacenamiento:

Clic derecho en nuestra VM | *Open* | *Show virtual hardware details* | + *Add Hardware*

2. En la ventana de "Add New Virtual Hardware", tenemos dos opciones:

- Seleccionar *Create a disk image for the virtual machine* para crear rápidamente un nuevo disco:

Al hacer clic en *Finish* se habrá el nuevo dispositivo de almacenamiento:

- Seleccionar *Select or create custom storage* para crear o seleccionar un almacenamiento personalizado:

Si hacemos clic en *Manage* podemos crear un nuevo pool/volumen o seleccionar un volumen existente. Clic en *Choose Volume*.

29.4.2 Agregando almacenamiento con `virsh`

`virsh` provee la opción `attach-disk` para conectar un nuevo dispositivo de disco a una VM. Hay muchos parámetros que provee esta opción:

```
attach-disk domain source target [[[--live] [--config] |
[--current]] | [--persistent]] [--targetbus bus] [--driver
driver] [--subdriver subdriver] [--iothread iothread] [--cache
cache] [--io io] [--type type] [--mode mode] [--sourcetype
sourcetype] [--serial serial] [--wwn wwn] [--rawio] [--address
address] [--multifunction] [--print-xml]
```

Lo siguiente es suficiente para realizar una conexión de un disco en caliente a una VM:

```
$ virsh attach-disk centos7.0 /var/lib/libvirt/images/virtual-disk.img vdc --live --
↪config
```

- `centos7.0` es la VM a la que conectaremos el disco.
- `/var/lib/libvirt/images/virtual-disk.img` es la ruta a la imagen de disco.
- `vdb` es el nombre del disco objetivo que será visible en el SO guest.
- `--live` significa que la acción se realizará mientras está corriendo la VM.

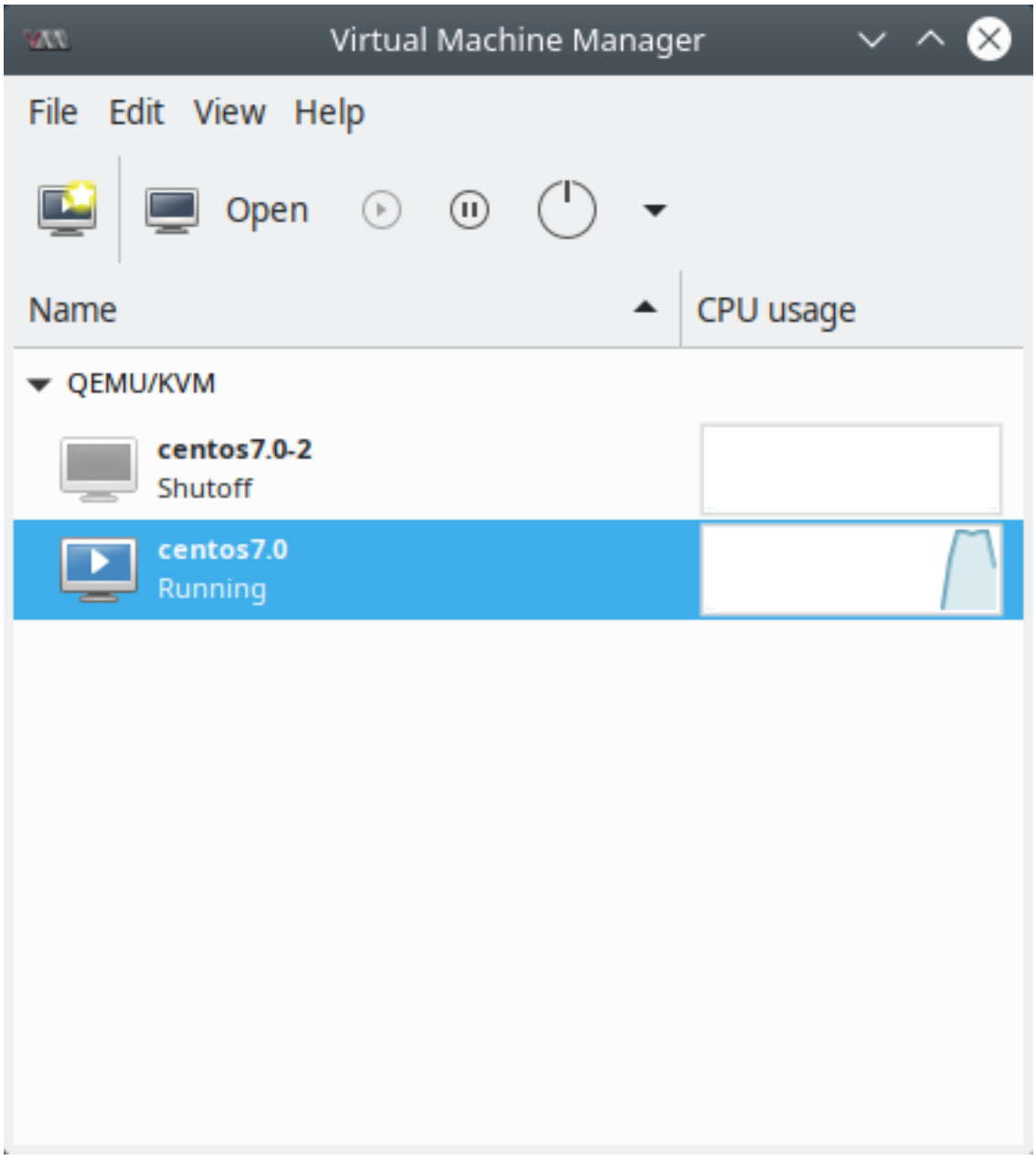


Fig. 7: `virt-manager` - Lista de VMs

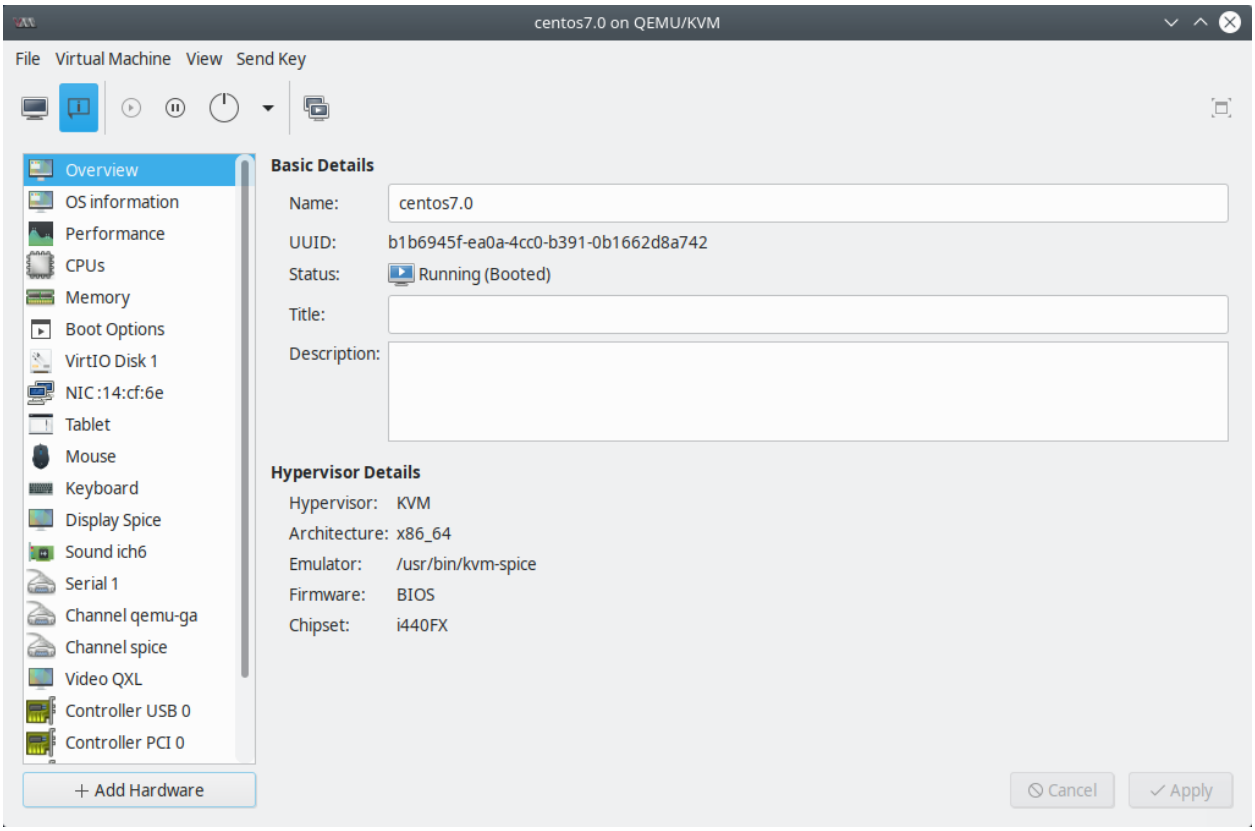


Fig. 8: `virt-manager` - Show virtual hardware details

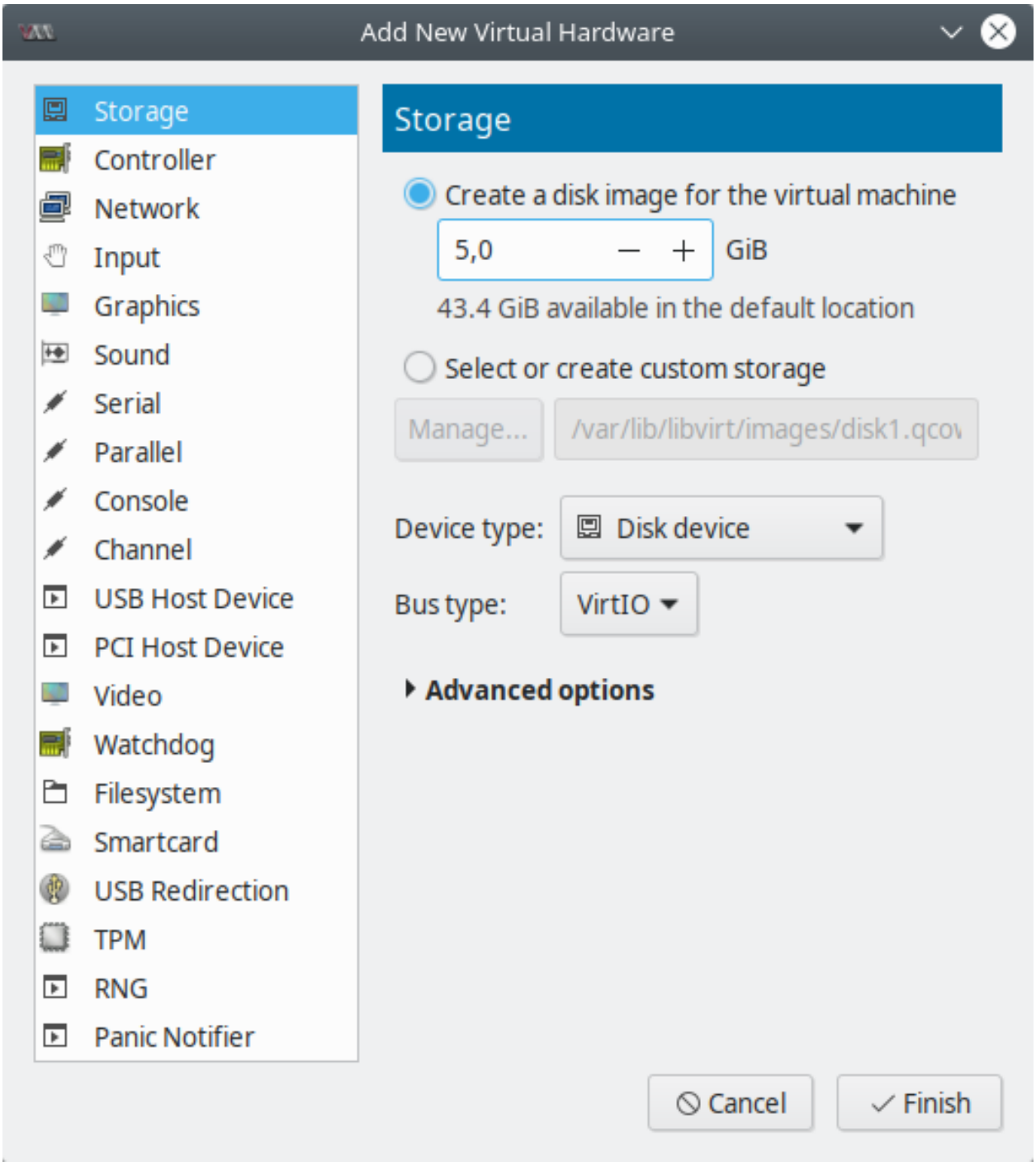


Fig. 9: `virt-manager` - Add new virtual hardware - Crear un nuevo disco

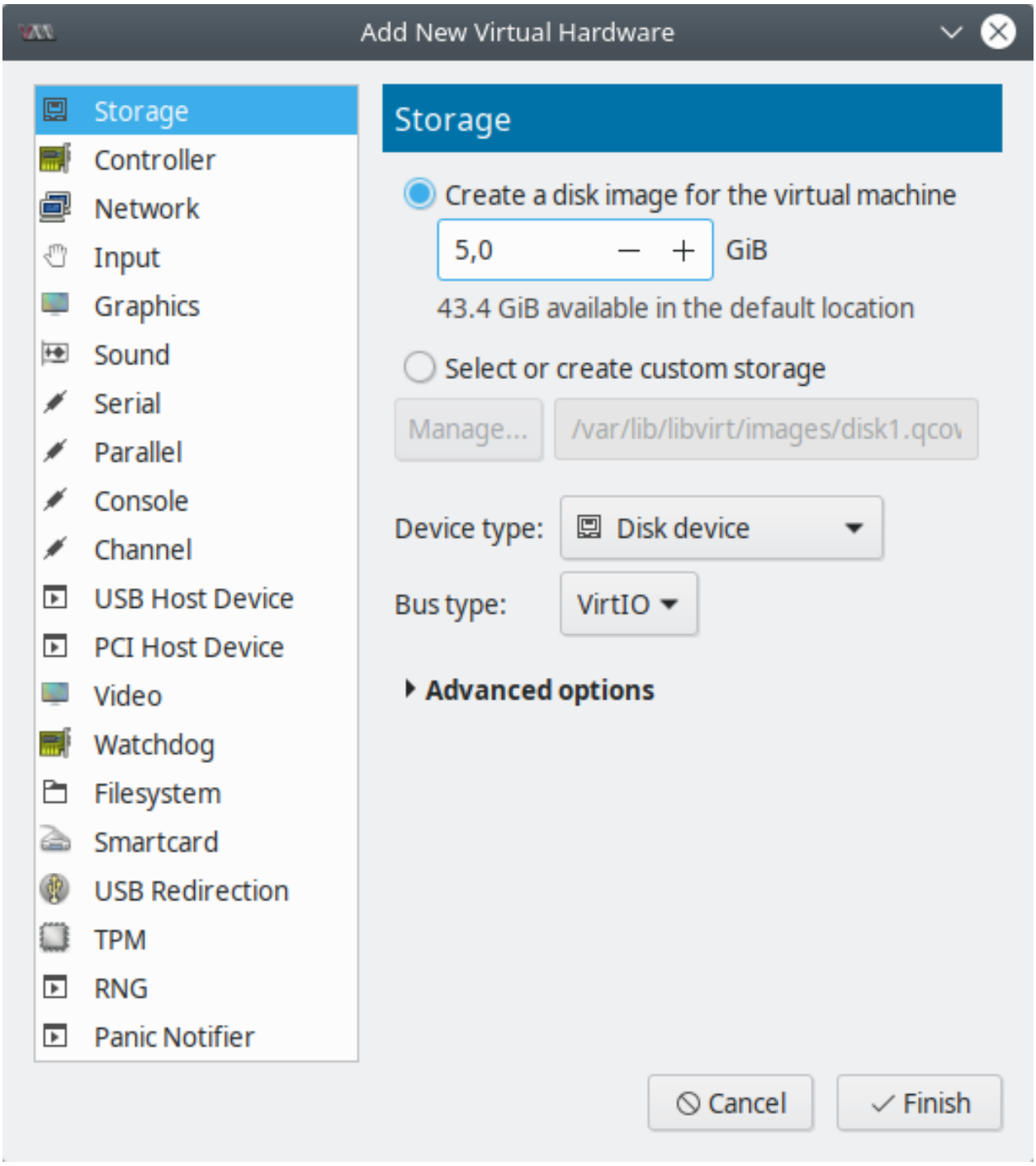


Fig. 10: `virt-manager` - Nuevo Virtual Disk

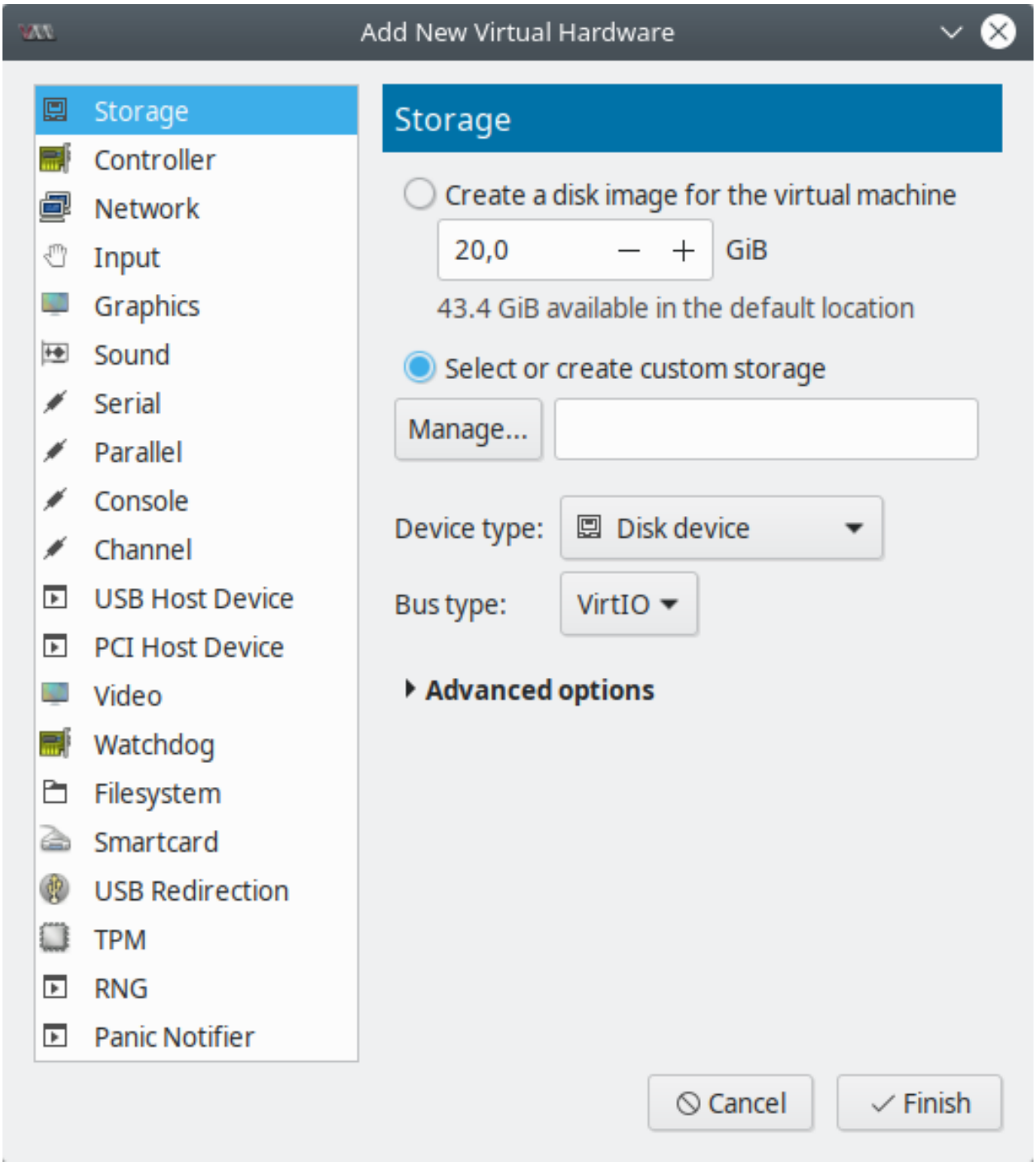


Fig. 11: `virt-manager` - Add new virtual hardware - Seleccionar o crear un almacenamiento personalizado

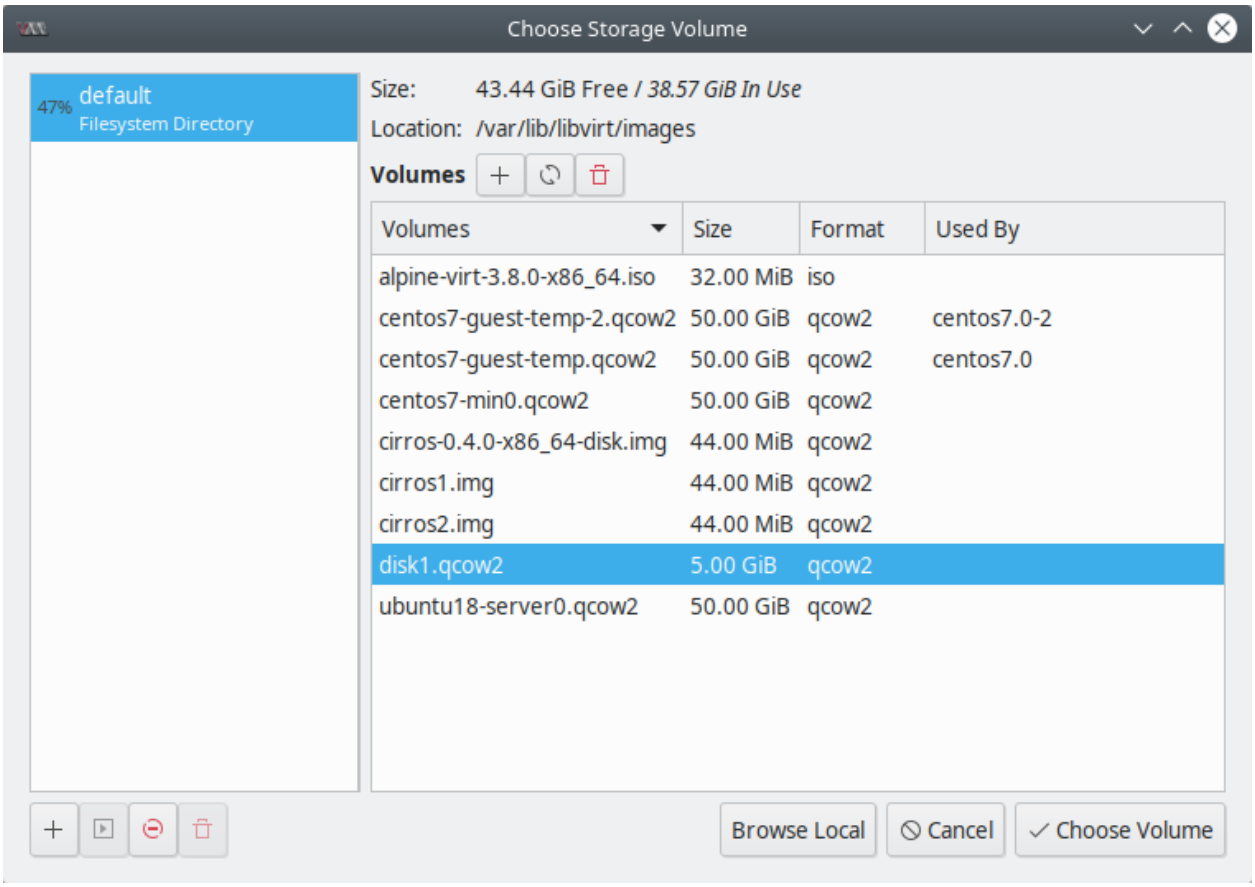


Fig. 12: `virt-manager` - Seleccionar/crear almacenamiento

- `--config` significa que la conexión del disco será permanente luego de un reinicio.

Usar el comando `virsh domblklist` para identificar cuántos vDisks están conectados a una VM:

```
$ virsh domblklist centos7.0 --details

Type       Device     Target     Source
------------------------------------------------
file       disk       vda        /var/lib/libvirt/images/centos7-guest-temp.qcow2
file       disk       vdb        /var/lib/libvirt/images/thinprovisioned-disk.img
```

Significa que los dos vDisks conectados a la VM son imágenes de archivos. Son visibles al SO guest como `vda` y `vdb`. En la última columna se ve la ruta de la imagen de disco en el sistema host.

Note: Otra opción para añadir un dispositivo de almacenamiento con `virsh` es a través de un archivo XML:

```
<disk type='file' device='disk>'>
    <driver name='qemu' type='raw' cache='none'/>
    <source file='/var/lib/libvirt/images/FileName.img'/>
    <target dev='vdb' bus='virtio'/>
</disk>
```

```
$ virsh attach-disk --config Guest1 ~/NewStorage.xml
```

29.4.3 Eliminando almacenamiento con `virsh`

Para eliminar un dispositivo de almacenamiento de una VM (`guest1`) con `virsh` usamos el comando:

```
$ virsh detach-disk guest1 vdb
```

29.4.4 Referencias

- Adding Storage Devices to Guests

29.5 Imágenes de disco con `qemu-img`

Table of Contents

`qemu-img` es una herramienta de QEMU para crear, convertir y administrar imágenes de disco de forma offline. Soporta todos los formatos que usa QEMU.

Warning: No usar `qemu-img` para modificar imágenes si están siendo usadas por una VM u otro proceso, pues dañaría la imagen.

29.5.1 Tipos de imágenes

QEMU soporta diversos tipos de imágenes, siendo **qcow2** el nativo y más flexible. Qcow2 soporta copy on write, encriptación, compresión y snapshots de VMs.

Algunos de los formatos máx comúnes son:

- **raw**: el formato raw es una imagen binaria plana de una images de disco y es muy portable. El filesystems que soportan *sparse files*, las imágenes en este formato solo usan el espacio usado por los datos grabados en ella.
- **cow**: el formato copy-on-write solo es soportado por razones históricas.
- **qcow**: el antiguo formato **QEMU copy-on-write**, también es soportado por razones históricas.
- **qcow2**: formato **QEMU copy-on-write** con distintos features especiales: tomar múltiples snapshots, imágenes pequeñas en el filesystem que no soportan sparse files, encriptación AES opcional y compresión zlib opcional.
- **vmdk**: formato de imagen de VMware
- **vdi**: formato de imagen de VirtuaBox
- **vhdx**: formato de imagen de Hyper-V
- **vpc**: formato de imagen legacy de Hyper-V

Note: Formatos soportados por `qemu-img`:

```
$ qemu-img --help | grep Supported
Supported formats: blkdebug blkreplay blkverify bochs cloop dmg file ftp ftps
host_cdrom host_device http https iscsi iser luks nbd null-aio null-co parallels
qcow qcow2 qed quorum raw rbd replication sheepdog throttle vdi vhdx vmdk vpc vvfat
```

29.5.2 Crear una imagen

Con el comando `qemu-img` de QEMU podemos crear un imagen de disco vacía donde podremos almacenar nuestro SO guest.

- Por defecto, el parámetro `create` de `qemu-img` creará una imagen con formato raw:

```
$ qemu-img create disk1.img 1G
```

Se ha creado una imagen de disco con formato raw de `1G` de tamaño con el nombre `disk1.img`.

- Podemos especificar que use el formato nativo de QEMU (`qcow2`) con el parámetro `-f`:

```
$ qemu-img create -f qcow2 disk2.qcow2 1G
```

Se ha creado una imagen de disco con formato qcow2 de `1G` de tamaño con el nombre `disk2.img`.

29.5.3 Obtener información de una imagen

Para ver los datos de una imagen de disco podemos usar la opción `info` del comando `qemu-img`:

```
$ qemu-img info disk1.img

image: disk1.img
file format: raw
virtual size: 1.0G (1073741824 bytes)
disk size: 0
```

```
$ qemu-img info disk2.img

image: disk2.img
file format: qcow2
virtual size: 1.0G (1073741824 bytes)
disk size: 196K
cluster_size: 65536
Format specific information:
    compat: 1.1
    lazy refcounts: false
    refcount bits: 16
    corrupt: false
```

29.5.4 Convertir formato de imagen

Podemos realizar conversión del formato de imágenes con la opción `convert` del comando `qemu-img` junto con los parámetros `-f`, que indica el formato del archivo de entrada y `-O`, que indica el formato del archivo de salida.

- Convertir una imagen de formato `raw` a `qcow2`:

```
$ qemu-img convert -f raw -O qcow2 disk1.img disk1converted.qcow2

$ qemu-img info disk1converted.qcow2
image: disk1converted.qcow2
file format: qcow2
virtual size: 1.0G (1073741824 bytes)
disk size: 196K
cluster_size: 65536
Format specific information:
    compat: 1.1
    lazy refcounts: false
    refcount bits: 16
    corrupt: false
```

- Convertir una imagen de formato `qcow2` a `raw`:

```
$ qemu-img convert -f qcow2 -O raw disk2.qcow2 disk2converted.img

$ qemu-img info disk2converted.img
image: disk2converted.img
file format: raw
virtual size: 1.0G (1073741824 bytes)
disk size: 0
```

29.5.5 Recortar y comprimir imágenes

Podemos decrementar el tamaño de imágenes con formato qcow2 gracias a la opción `convert` del comando `qemu-img`. Durante la conversión de imagen, los sectores vacios son detectados y suprimidos de la imagen destino.

- Recortar una imagen qcow2 sin compresión (archivo más grande, menor tiempo de procesamiento)

```
$ sudo qemu-img convert -O qcow2 source.qcow2 shrunk.qcow2
```

- Recortar una imagen qcow2 con compresión (`-c`) (archivo más pequeño, mayor tiempo de procesamiento)

```
$ sudo qemu-img convert -O qcow2 -c source.qcow2 shrunk.qcow2
```

29.5.6 Referencias

- qemu-img - Manpage Debian
- qemu-img - Wikibooks
- Convert qcow2 to raw image and raw to qcow2 image
- How to shrink OpenStack qcow2 image using qemu-img

29.6 Creando un disco virtual

Table of Contents

Un disco duro virtual o Virtual Hard Disk (VHD) es un archivo contenedor que actúa similar a un disco duro físico. Un uso típico de un VHD es en VMs, para guardar SOs, sus aplicaciones y datos.

En este tutorial crearemos un VHD que será montado en el host.

1. Crear una imagen que contenga el volumen virtual

Una opción es usar el comando `dd`. Más información en *Pools con clientes de libvirt*.

En este caso crearemos una imagen de disco de 4GB usando bloques de 1GB:

```
$ sudo dd if=/dev/zero of=/media/disk1.img bs=1G count=4

4+0 records in
4+0 records out
4294967296 bytes (4,3 GB, 4,0 GiB) copied, 8,76525 s, 490 MB/s
```

2. Formatear el archivo de la imagen VHD

Utilizar la herramienta `mkfs` para formatear la imagen VHD con el formato `ext4`:

```
$ sudo mkfs -t ext4 /media/disk1.img

mke2fs 1.44.1 (24-Mar-2018)
Discarding device blocks: done
```

(continues on next page)

(continued from previous page)

```
Creating filesystem with 1048576 4k blocks and 262144 inodes
Filesystem UUID: 7ec58ec5-313b-44b6-b3d2-a08d0a08b66e
Superblock backups stored on blocks:
        32768, 98304, 163840, 229376, 294912, 819200, 884736

Allocating group tables: done
Writing inode tables: done
Creating journal (16384 blocks): done
Writing superblocks and filesystem accounting information: done
```

3. Montar el la imagen disco formateada en un directorio

Para poder acceder a un volumen VHD, debemos montarlo en un directorio. Usaremos la herramienta `mount`:

```
$ sudo mkdir /mnt/vhd/
$ sudo mount -t auto -o loop /media/disk1.img /mnt/vhd/
```

- `-t`: tipo
- `-o`: opciones

4. Volver permanente el montado

Para volver permanente el montado del filesystem del VHD luego de reiniciar el sistema, ingresar una nueva entrada en el archivo `/etc/fstab`:

```
/media/disk1.img  /mnt/vhd/  ext4    defaults        0  0
```

5. Comprobar que existe el filesystem del nuevo disco virtual junto con la dirección del directorio donde está montado

```
$ df -hT

Filesystem     Type      Size  Used Avail Use% Mounted on
...
/dev/loop12    ext4      3,9G   16M  3,7G   1% /mnt/vhd
```

Note: Para eliminar el volumen VHD primero desmontar el filesystem VHD y luego eliminar el archivo de imagen

```
$ sudo umount /mnt/vhd/
$ sudo rm /media/disk1.img
```

CHAPTER 30

Web

- Jekyll
- Gatsby
- React
- Electron.js (apps de escritorio)
- Github Pages
- Sphinx + RTD

CHAPTER 31

Windows

CHAPTER 32

Windows Subsystem For Linux

32.1 Enabling Windows Subsystem for Linux

Table of Contents

32.1.1 Enable WSL

Podemos habilitar el **Windows Subsystem for Linux (WSL)** desde la interfaz gráfica o usando comandos en Powershell.

Option 1 - Windows Features dialog

En la barra de navegación de Windows, buscar *Características de Windows* y abrir el programa *Activar o desactivar las características de Windows*. Activar la opción *Subsistema de Windows para Linux* (*Windows Subsystem for Linux*). Presionar *Aceptar*

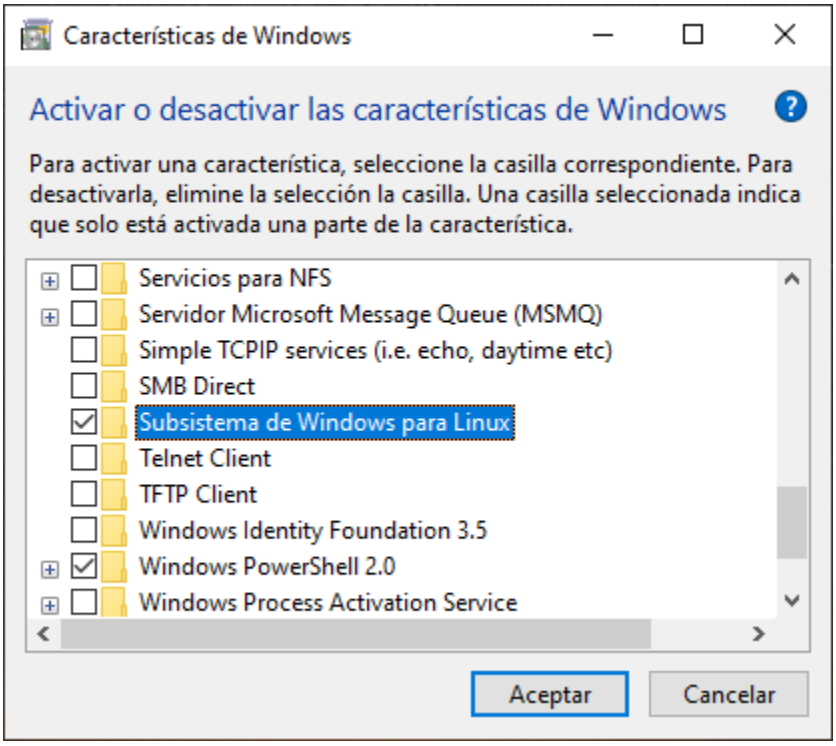


Fig. 1: Habilitando WSL en *Características de Windows*

Luego nos saldrá una opción para reiniciar Windows, aceptarla.

Option 2 - PowerShell

Abrir *PowerShell* con la opción *Ejecutar como Administrador* y ejecutar el siguiente comando:

```
Enable-WindowsOptionalFeature -Online -FeatureName Microsoft-Windows-Subsystem-Linux
```

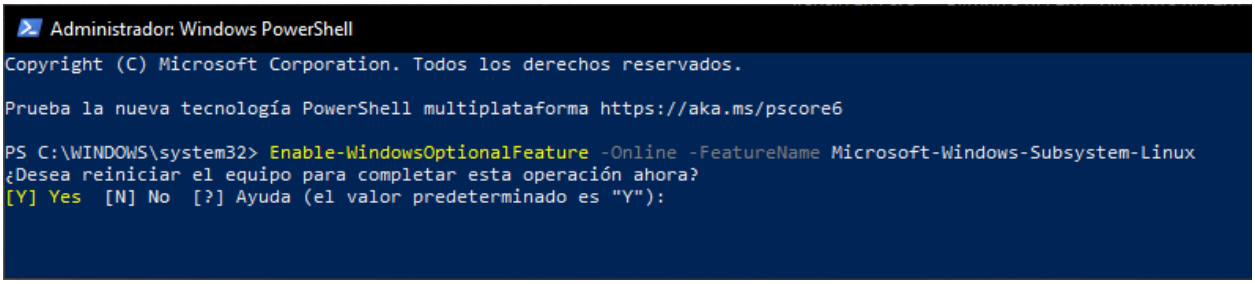


Fig. 2: Habilitando WSL en *Powershell*

Luego de correr el comando debemos reiniciar el sistema, aceptando con *Y*.

32.1.2 Check

Luego de haber reiniciado Windows, podemos comprobar que hemos instalado correctamente WSL corriendo el comando `wsl` en el terminal:

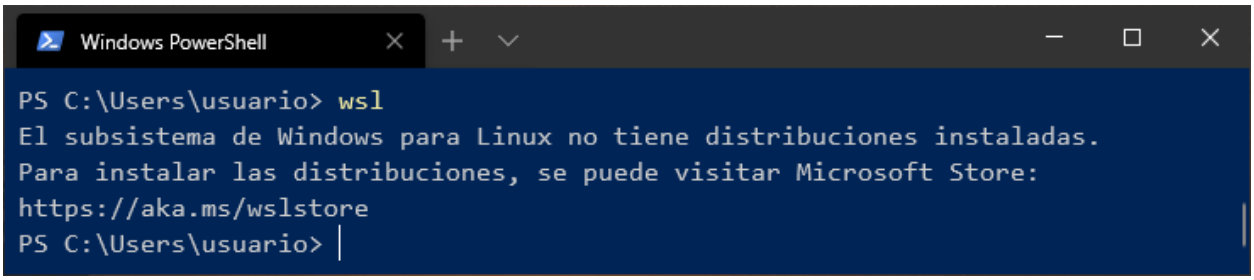


Fig. 3: Check WSL

32.1.3 Install your Linux Distribution of Choice

Option 1 - Download and install from the Microsoft Store

La opción más sencilla es buscar la distribución de nuestra preferencia en el Microsoft Store e instalarla:

Option 2 - Download and install from the Command-Line/Script

Otra opción será usar el programa *Powershell*:

- Descargar la distribución por línea de comandos:

```
cd .\Desktop\
Invoke-WebRequest -Uri https://aka.ms/wsl-ubuntu-1804 -OutFile Ubuntu.appx -
↪UseBasicParsing
```

Note: Ver lista de distros disponibles para descargar en formato `.appx`: Downloading distros

- Instalar la distro:

```
Add-AppxPackage .\Ubuntu.appx
```

32.1.4 Initializing a newly installed distro

Initializing from Ubuntu terminal

- Abrir en el navegador de Windows el programa con el nombre de la aplicación instalada. Por ejemplo, en este caso *Ubuntu*
- Comenzará la configuración inicial y nos pedirá crear un usuario con contraseña:

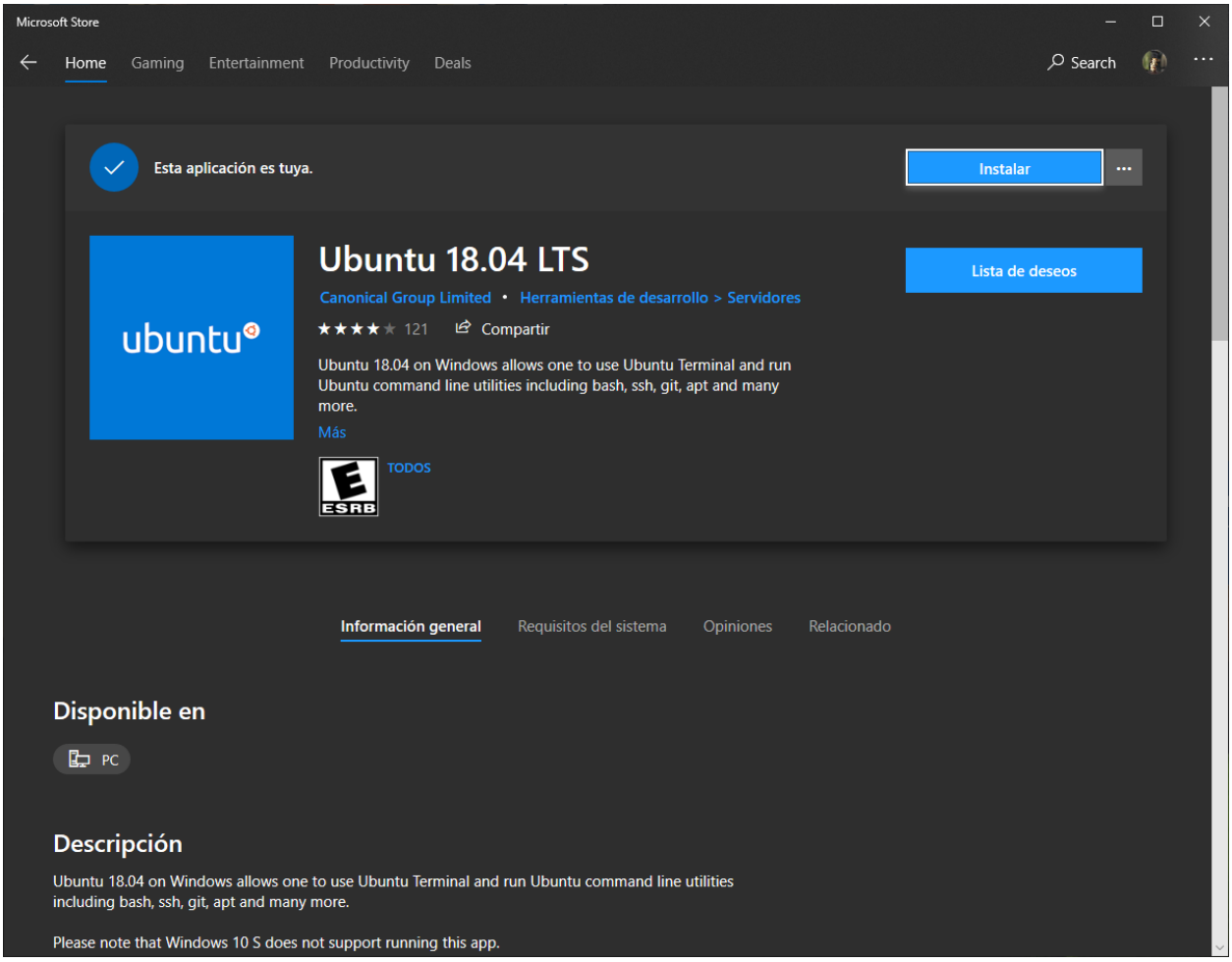


Fig. 4: Instalar Ubuntu 18.04 Distro para WSL

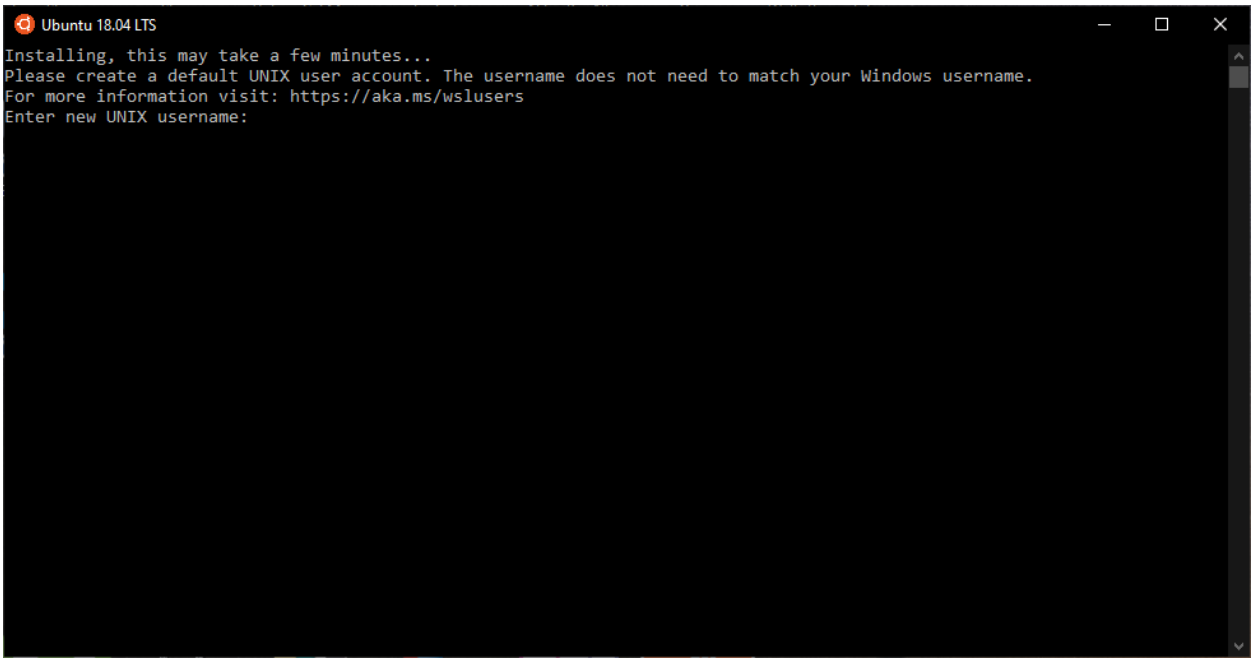


Fig. 5: Iniciar el programa de la distro instalada

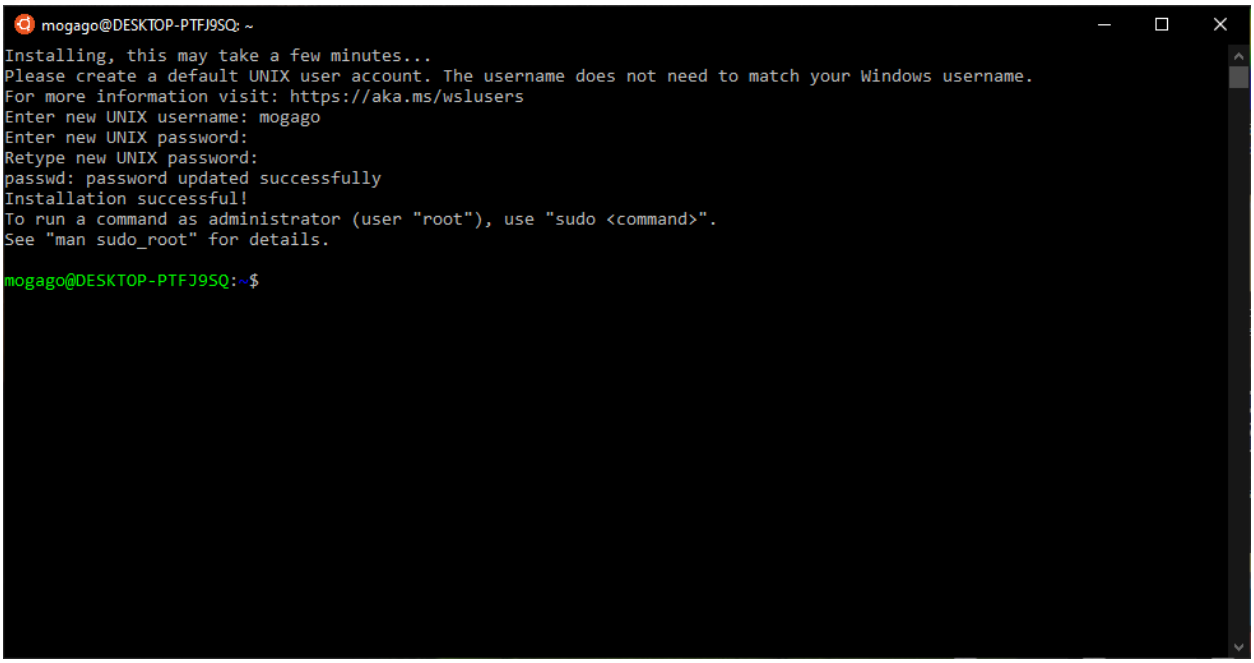


Fig. 6: Iniciar el programa de la distro instalada

Initializing from Powershell

Podemos iniciar el **Windows Subsystem for Linux (WSL)** desde el **Powershell** o la línea de comandos **CMD** de Windows usando el comando `wsl`:

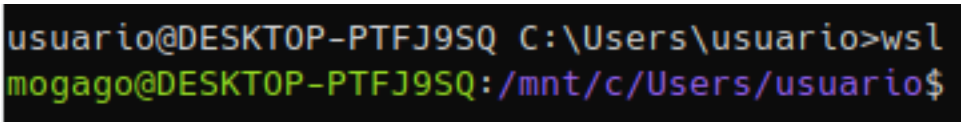


Fig. 7: Iniciar el programa de la distro instalada desde Powershell

32.1.5 Referencias

- Windows Subsystem for Linux Documentation
- Windows Subsystem for Linux Installation Guide for Windows 10
- Enable WSL
- Manually download Windows Subsystem for Linux distro packages
- Downloading distros
- Initializing a newly installed distro
- How to Remove Windows 10s Bash Tools Completely

32.2 Enabling SSH into WSL

Table of Contents

32.2.1 Instalación y configuración de OpenSSH server

- Instalar `openssh-server` en la distro de WSL:

```
$ sudo apt-get install openssh-server

Reading package lists... Done
Building dependency tree
Reading state information... Done
openssh-server is already the newest version (1:7.6p1-4ubuntu0.3).
0 upgraded, 0 newly installed, 0 to remove and 0 not upgraded.
```

Note: Para la distro de WSL Ubuntu 18.04, ya se encontrará instalado el servidor SSH por defecto

- En el WSL bash shell, reinstalar (reconfigurar) OpenSSH con:

```
$ sudo dpkg-reconfigure openssh-server

Creating SSH2 RSA key; this may take some time ...
2048 SHA256:a1Wm2wlyu+xMvJpFADe8oqlSJCUfT96/8qXPYcWGe8U root@DESKTOP-PTFJ9SQ (RSA)
Creating SSH2 ECDSA key; this may take some time ...
256 SHA256:YYqT6dvLFpUA21BduvnqXM7i5RdSurvdkafD2lw7glw root@DESKTOP-PTFJ9SQ (ECDSA)
Creating SSH2 ED25519 key; this may take some time ...
256 SHA256:q084sqn9frKtN9ZcwQw02/mGmYzn7XNQsFoKzdQ4GDo root@DESKTOP-PTFJ9SQ (ED25519)
invoke-rc.d: could not determine current runlevel
```

- Editar el archivo `/etc/ssh/sshd_config` y agregar las 2 líneas siguientes:

```
$ sudo vim /etc/ssh/sshd_config
```

```
PasswordAuthentication yes
Port 2222
```

Note: La razón del cambio de puerto por defecto para SSH (`22`) es debido a que puede estar en uso por Windows.

32.2.2 Crear una regla en el firewall de Windows

- Abrir el puerto `2222` en el firewall de Windows:

1. En el barra de navegación de Windows, abrir el programa *WF.msc* Seleccionar la opción *Reglas de entrada* en la barra izquierda:
2. Seleccionar la opción *Nueva regla...* en la barra derecha:
3. Seleccionar el tipo de regla *Puerto*:
4. Aplicar la regla a *TCP* y usar la opción *Puertos locales específicos:* y escribir *2222*:
5. Elegir la opción *Permitir la conexión*:
6. Aplicar la regla sobre todas las opciones: *Dominio*, *Privado*, *Público*
7. Dar un nombre y descripción a la nueva regla de firewall creada:

32.2.3 Reiniciar el servicio SSH

- En el WSL bash shell, reiniciar el SSH server

```
$ sudo service ssh --full-restart
```

- Comprobar que Windows se encuentra escuchando en el puerto `2222`. Desde el Powershell ejecutar:

```
$ netstat -aon | findstr "2222"
```

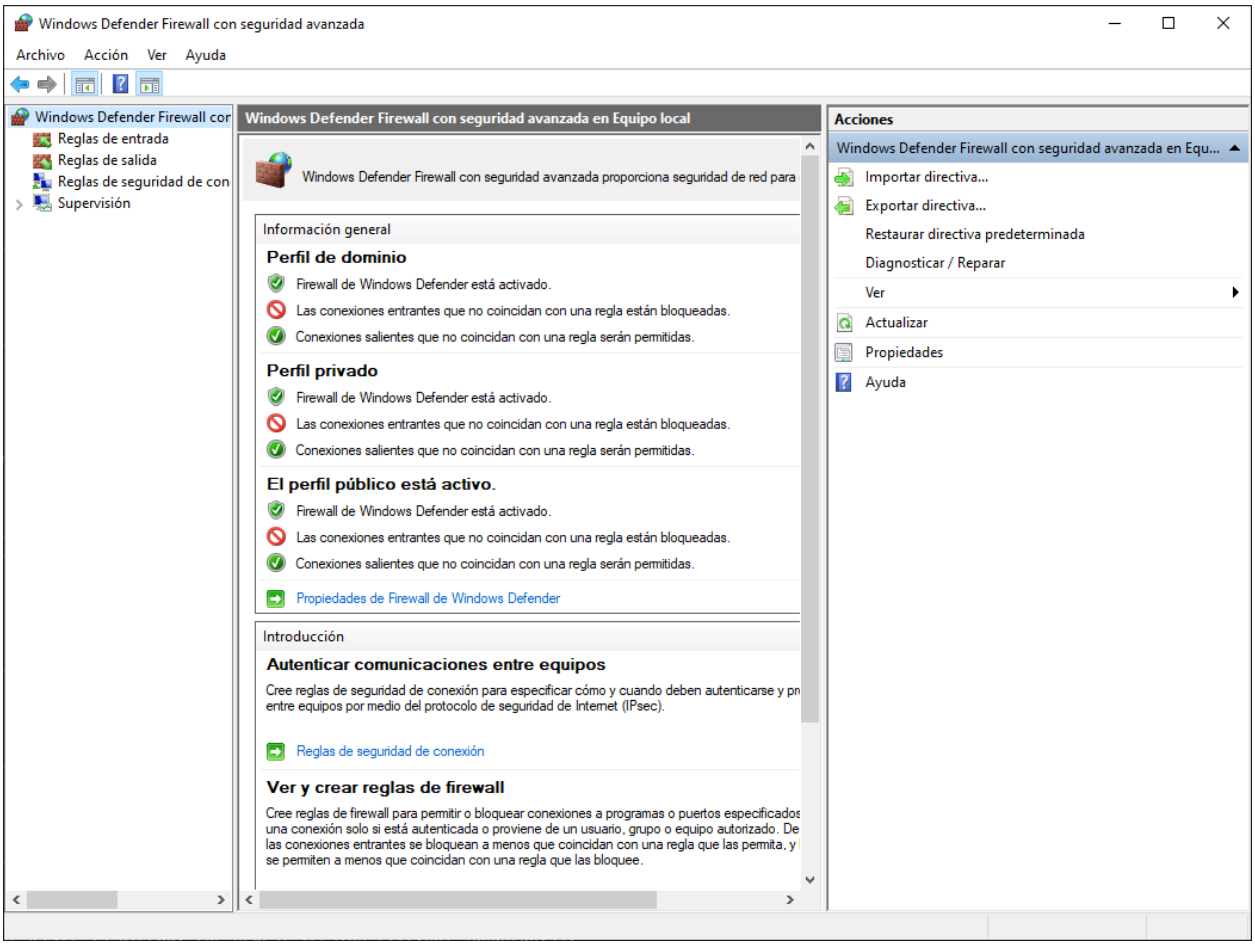


Fig. 8: Abrir el programa *WF.msc* de Windows, *Reglas de entrada*

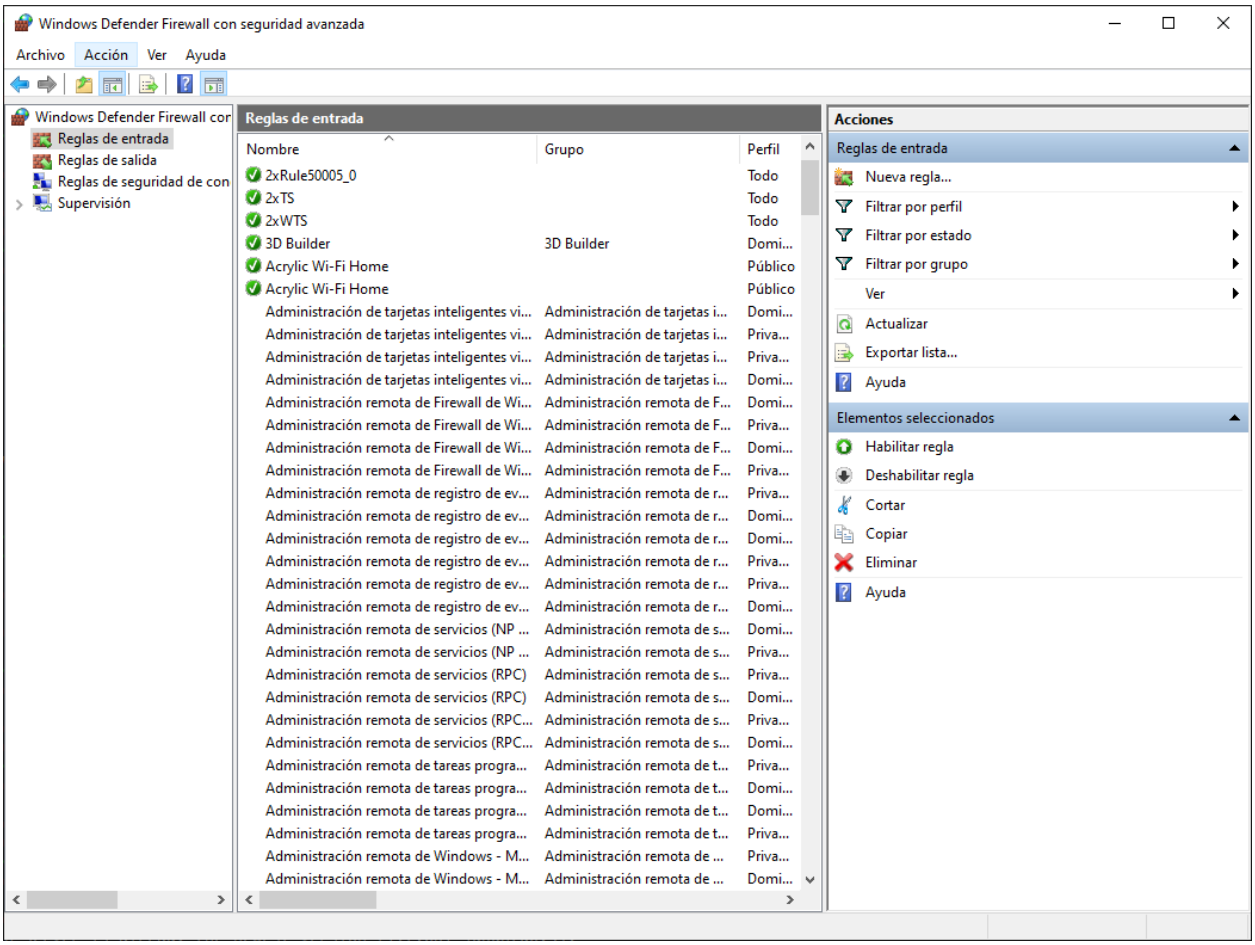


Fig. 9: Seleccionar la opción *Nueva regla...*

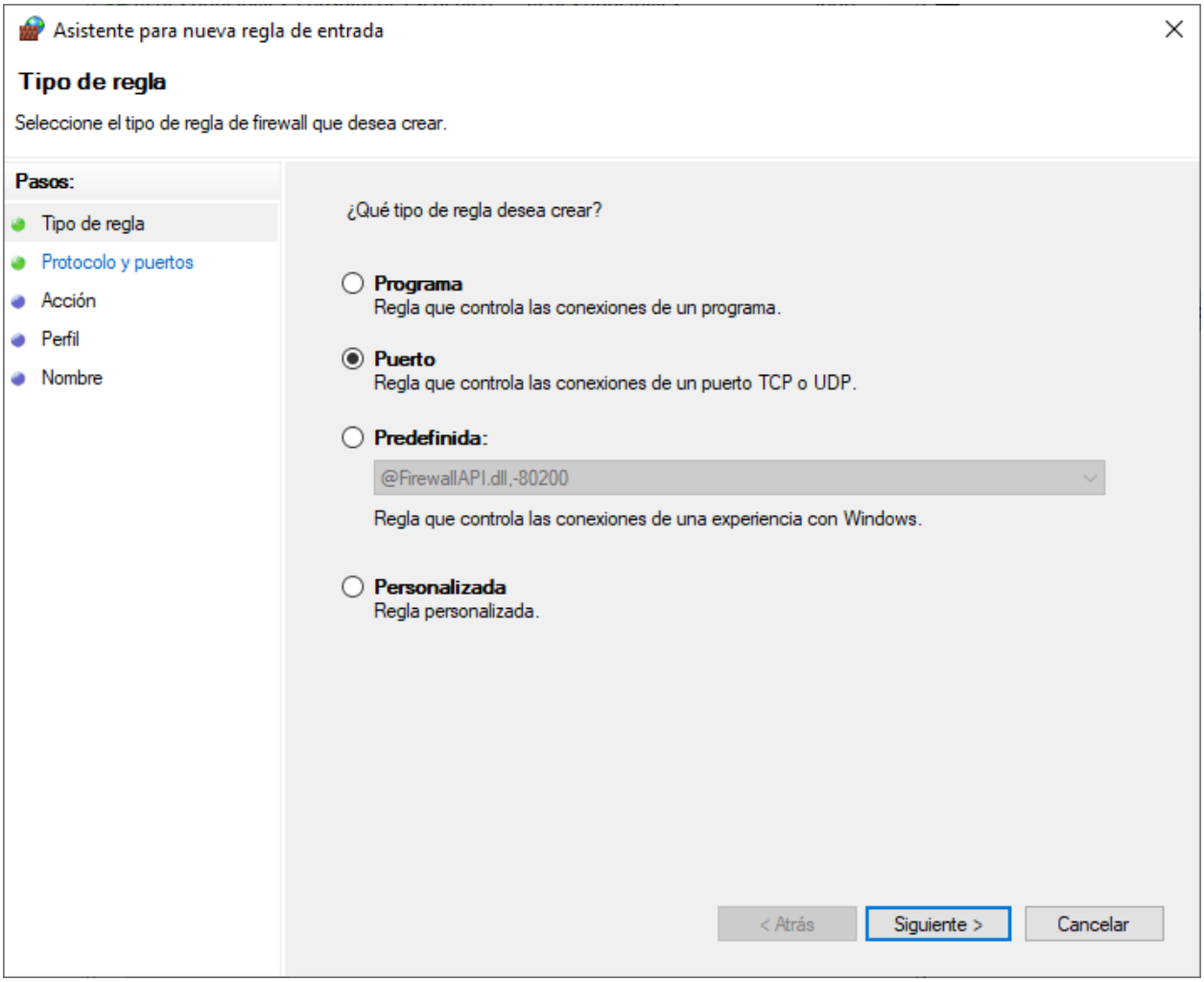


Fig. 10: Tipo de regla: *Puerto*

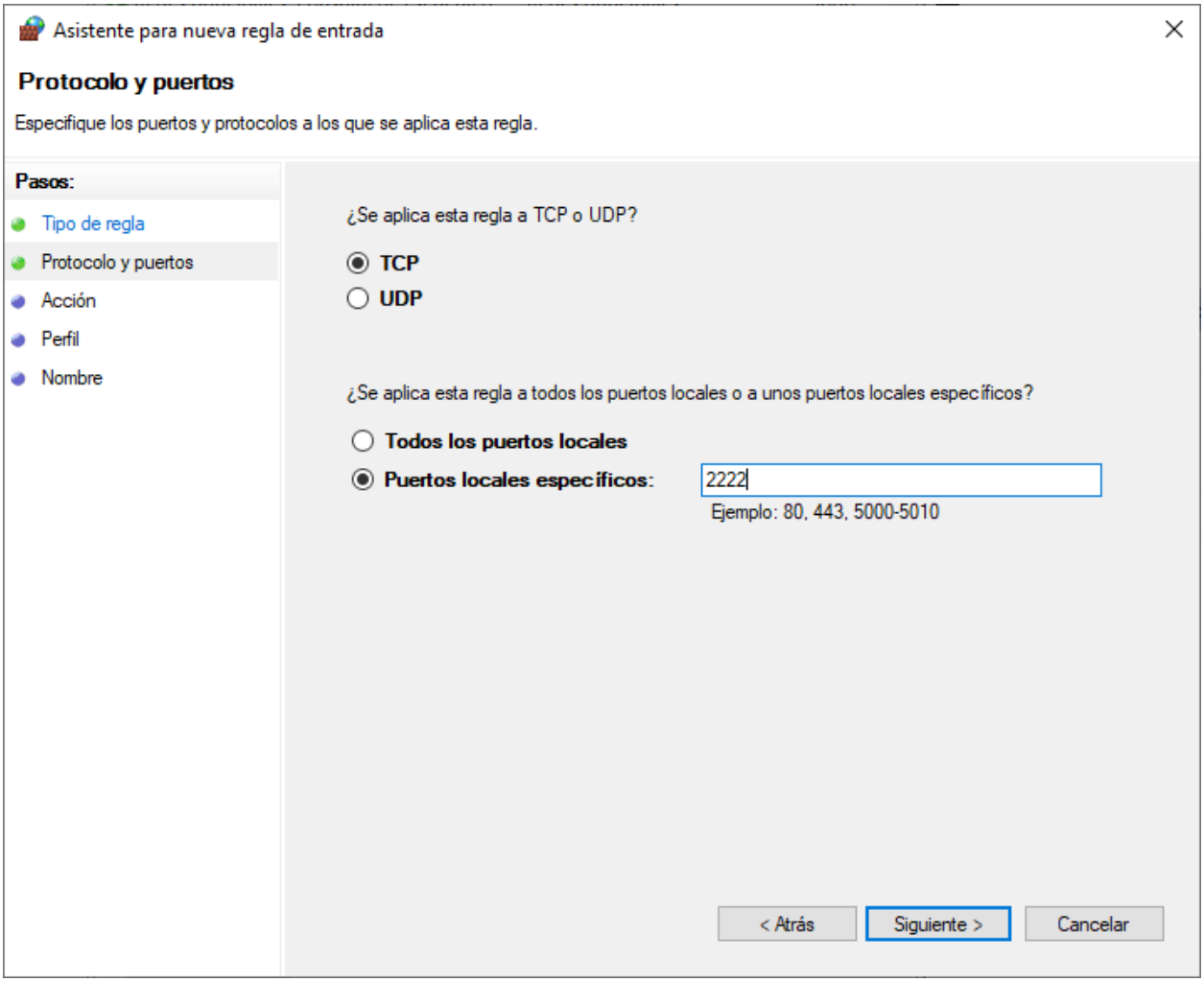


Fig. 11: Aplicar la regla a *TCP*, *Puertos locales específicos: 2222*

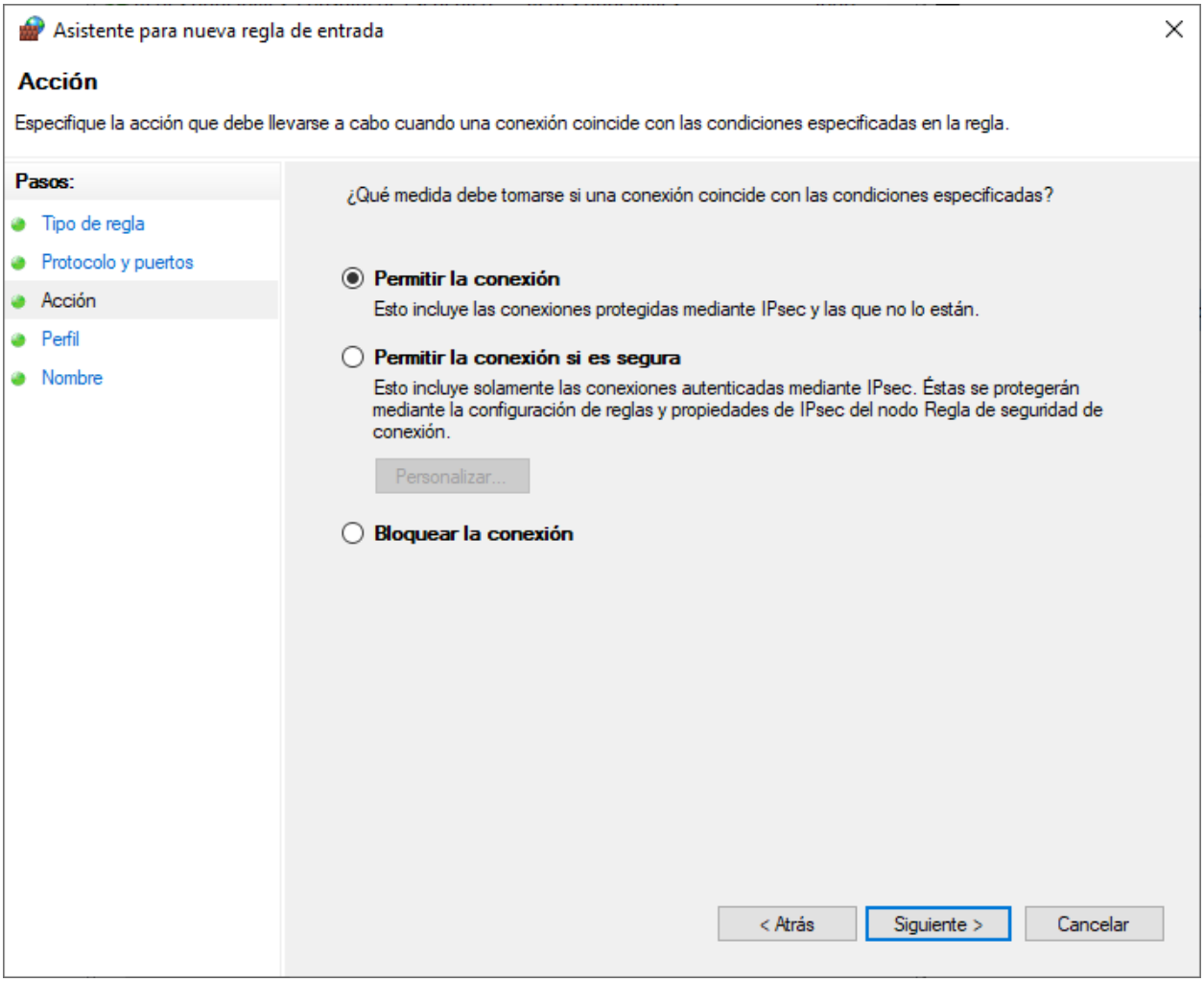


Fig. 12: Seleccionar *Permitir la conexión*

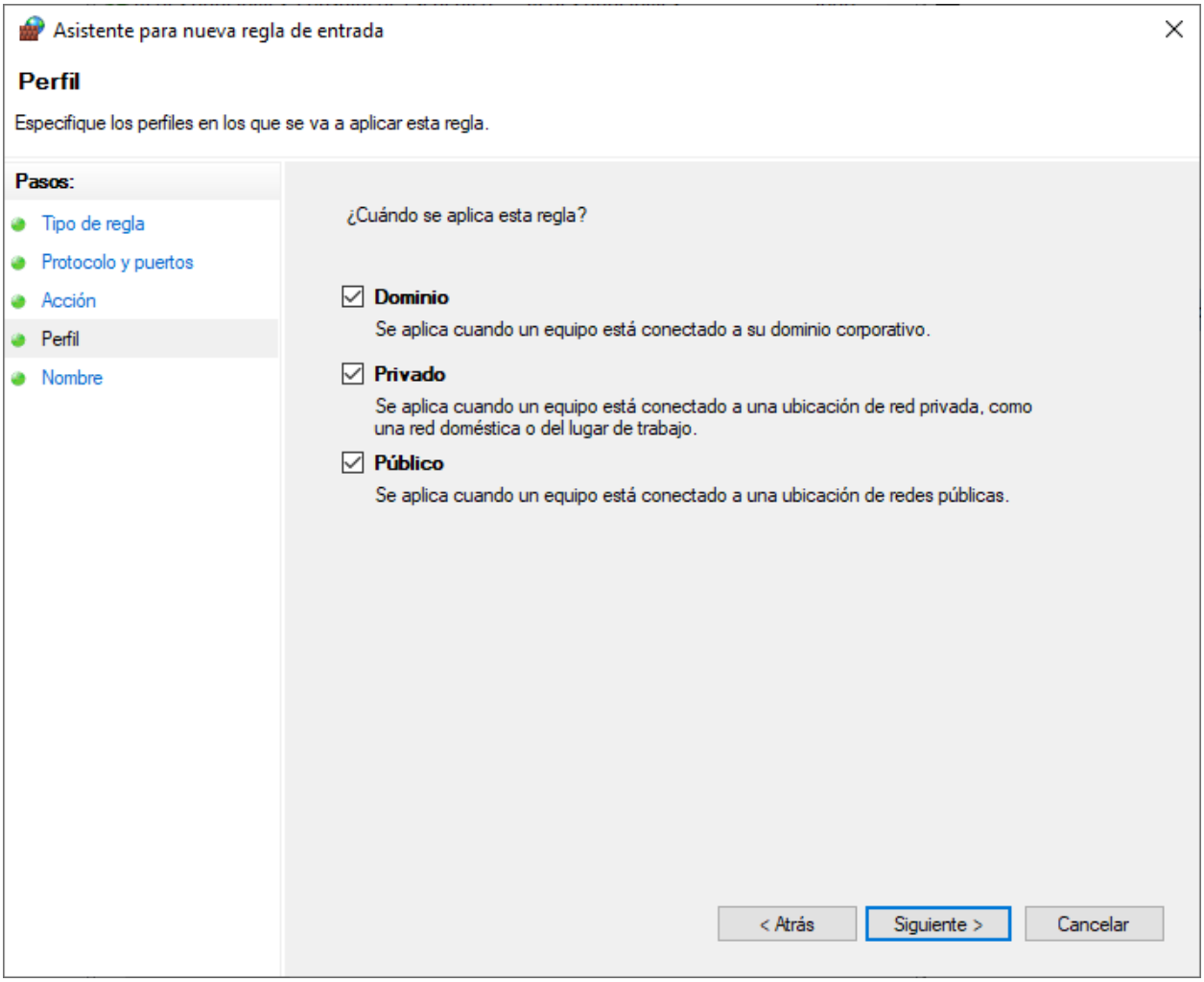


Fig. 13: Seleccionar *Permitir la conexión*

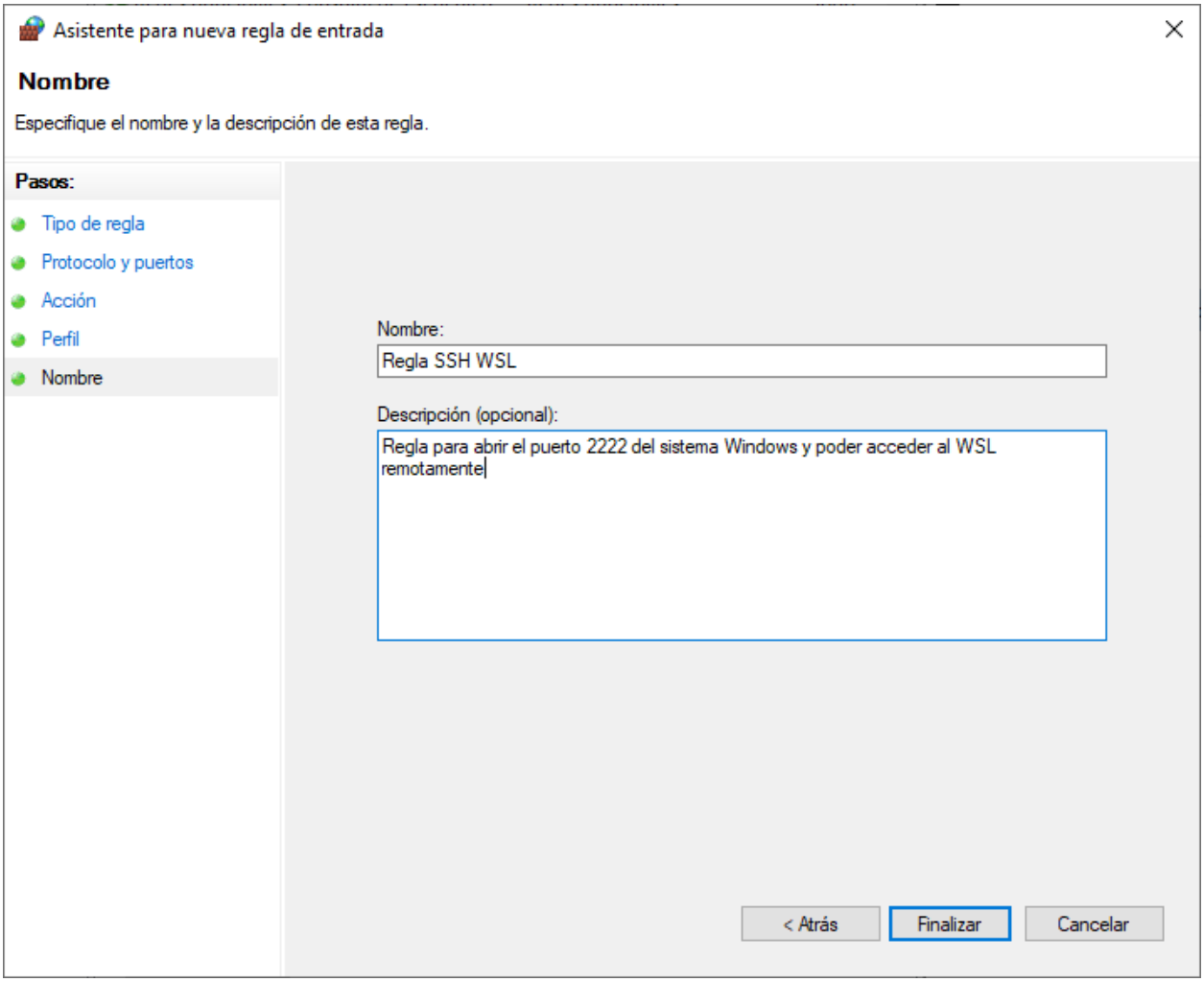


Fig. 14: Nombre y descripción de la regla de firewall

32.2.4 Conexión remota por SSH

- Desde un equipo remoto conectarnos por SSH al WSL de Windows usando un terminal o PuTTY, apuntando a la IP del sistema Windows y el puerto `2222`. Por ejemplo para conectarnos por SSH desde un terminal de Linux usamos:

```
$ ssh mogago@192.168.1.8 -p 2222
```

32.2.5 Referencias

- SSH on Windows Subsystem for Linux
- SSH-ing into a Windows WSL Linux Subsystem
- How to Determine What Ports are Being Used in Windows 10

32.3 Iniciar el servidor SSH automáticamente en WSL

Table of Contents

Documentación para iniciar el servidor SSH de WSL junto con el arranque del sistema de Windows. El comportamiento normal es que WSL no iniciará una tarea mientras no ejecutemos el bash de Linux e iniciemos manualmente el servicio SSH.

32.3.1 Permitir iniciar el servicio SSH sin password

- Editar los permisos de `/etc/sudoers` con `visudo`:

```
$ sudo visudo
```

- Añadir la siguiente línea al final del archivo `/etc/sudoers` para tener permisos de iniciar el servicio de SSH:

```
%sudo ALL=NOPASSWD: /etc/init.d/ssh start
```

Note: Otras opciones de permisos en `/etc/sudoers`:

```
# Start/Stop scripts for ssh service
%sudo ALL=NOPASSWD: /etc/init.d/ssh

# service command
%sudo ALL=NOPASSWD: /usr/sbin/service
```

(continues on next page)

(continued from previous page)

```
# ssh command
%sudo ALL=NOPASSWD: /usr/bin/ssh

# sshd command
%sudo ALL=NOPASSWD: /usr/sbin/sshd
```

32.3.2 Crear una nueva tarea en el Programador de tareas de Windows

- Abrir el programa *Programador de tareas*:

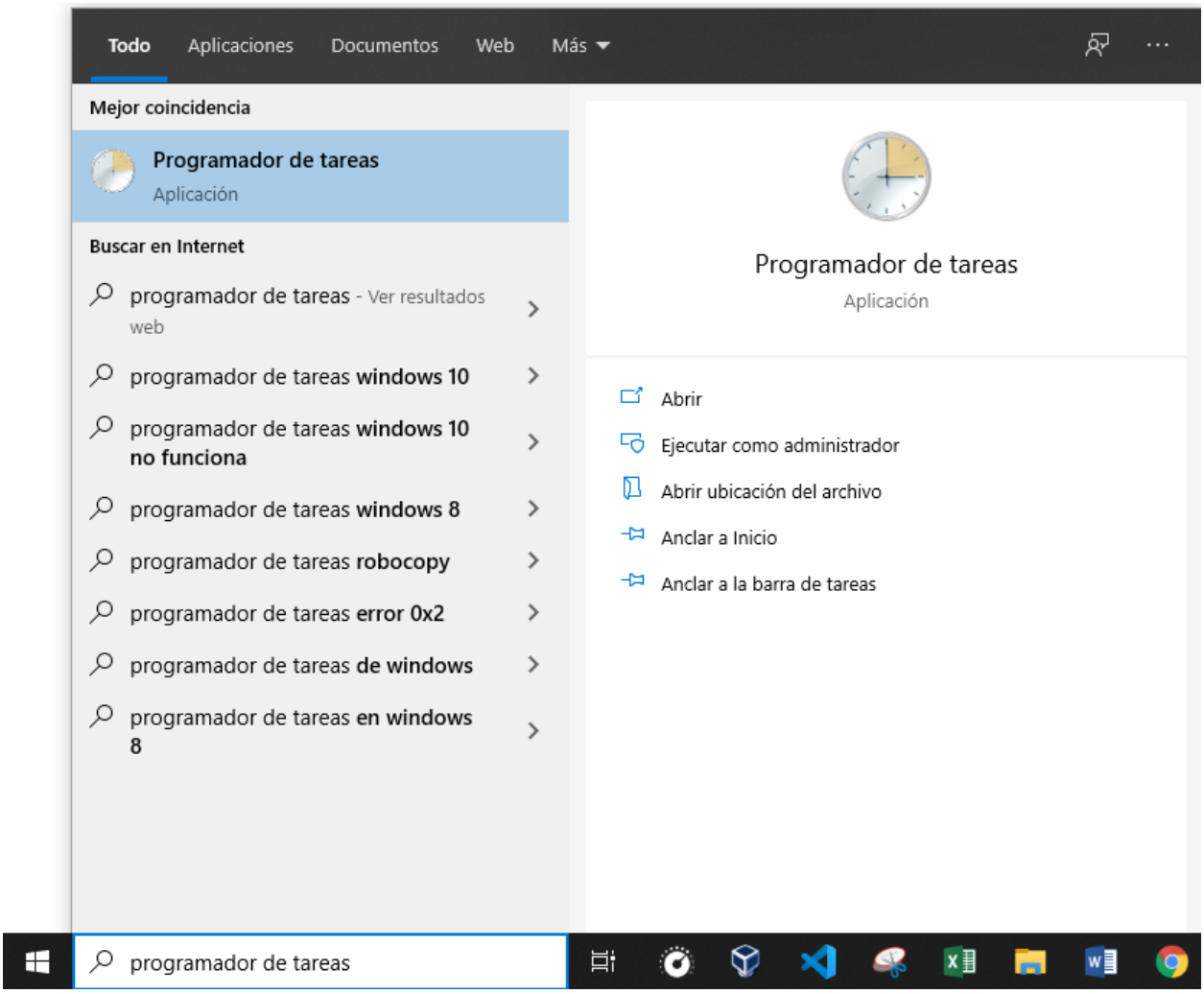


Fig. 15: Búsqueda de Windows: *Programador de tareas*

- En el *Programador de tareas*, seleccionar la opción *Crear tarea básica...*:
- Pasos para la creación de una nueva tarea:

1. Dar un Nombre y Descripción a la nueva tarea:

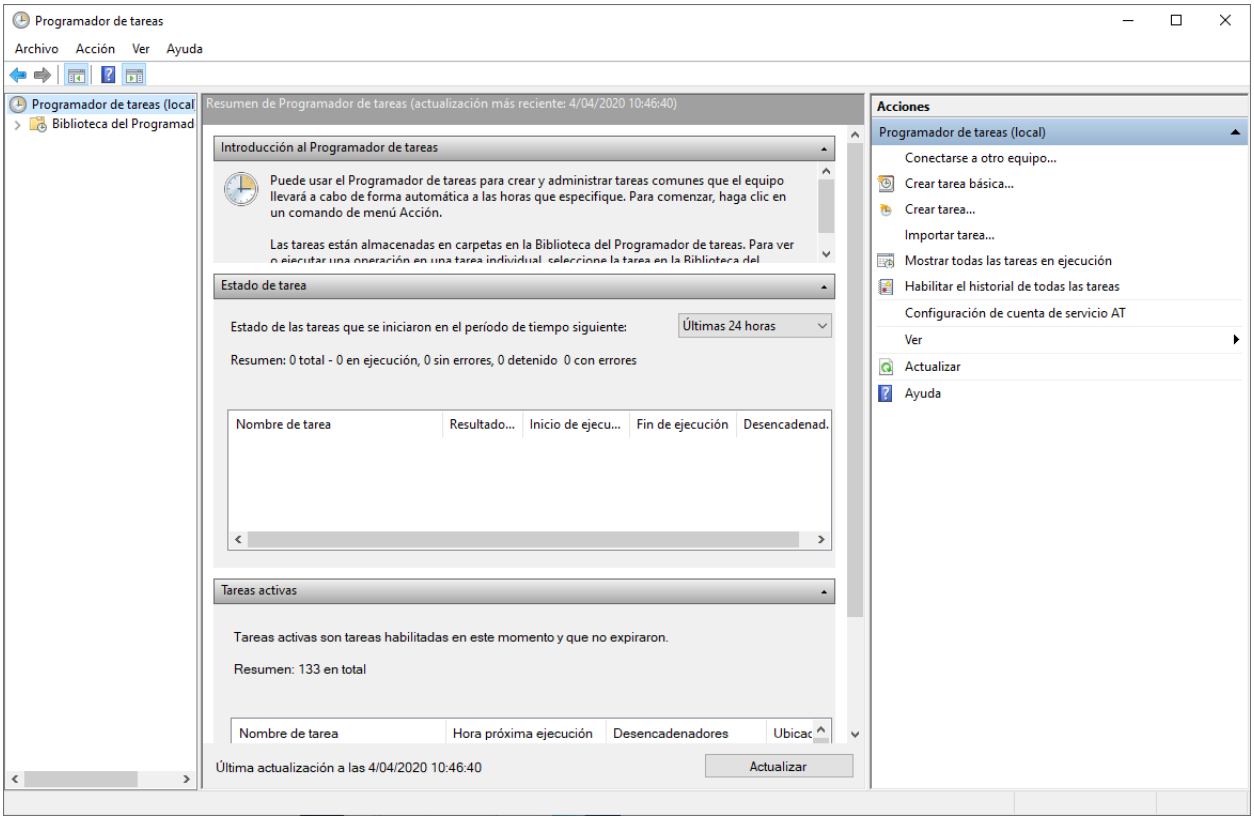


Fig. 16: Búsqueda de Windows: *Programador de tareas*

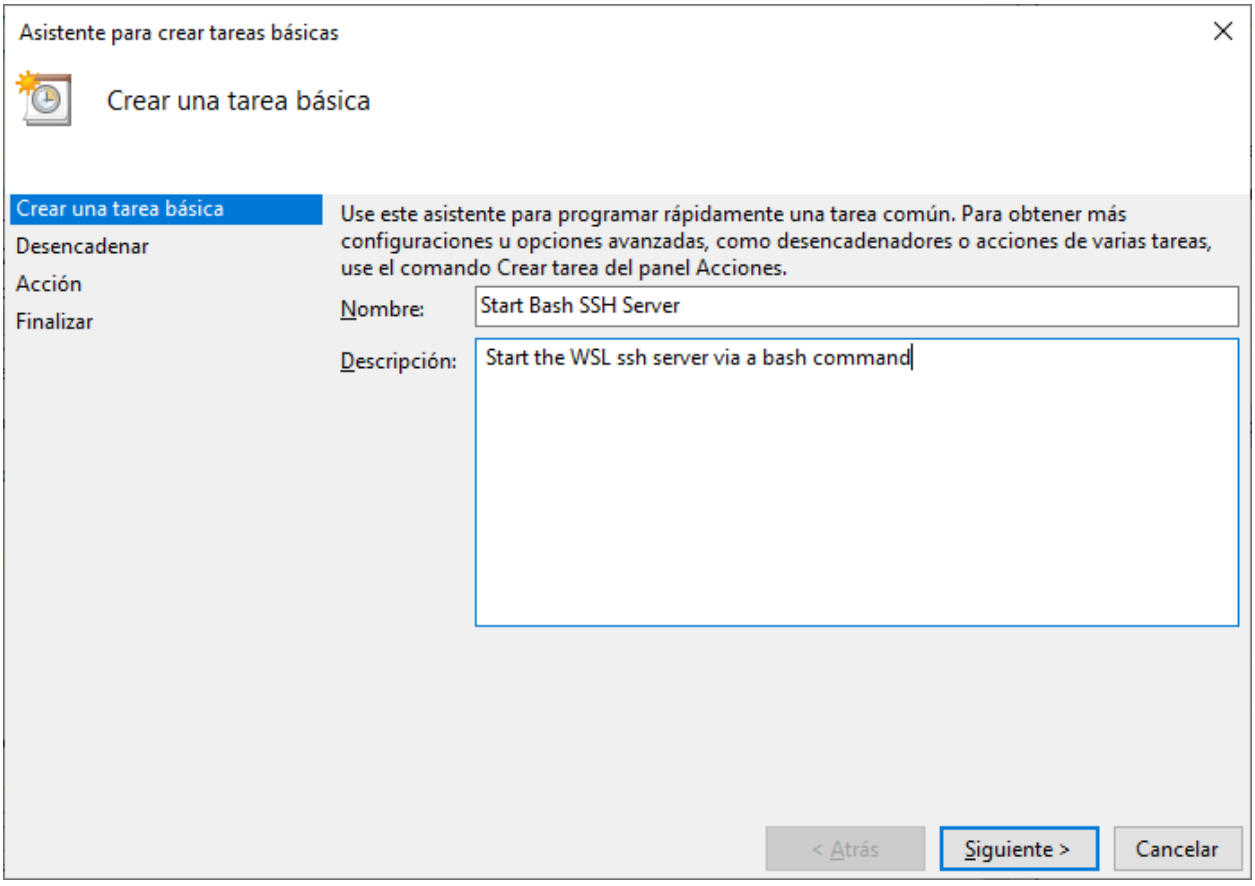


Fig. 17: Dar un Nombre y Descripción a la tarea

2. Elegir como desencadenante de la acción *Al iniciar sesión*:

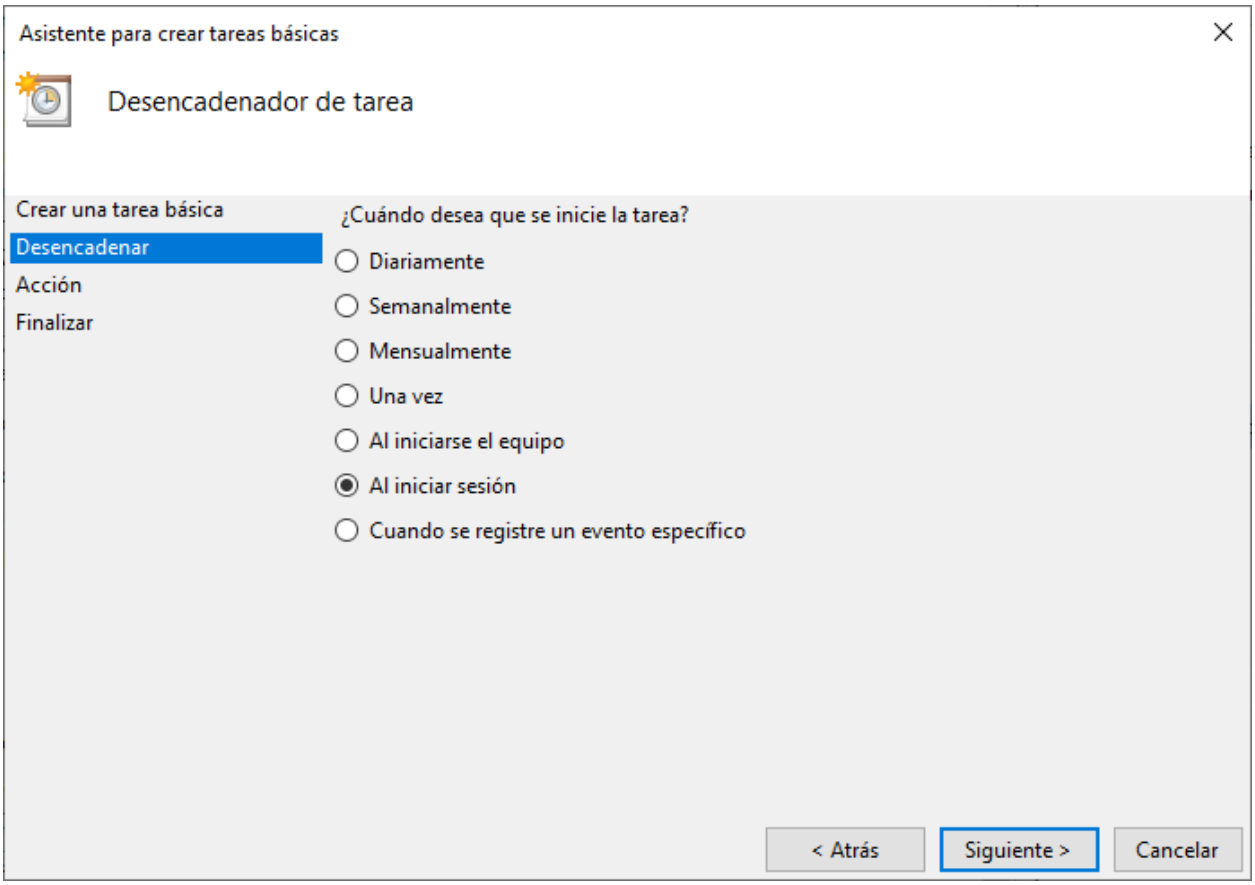


Fig. 18: Desencadenador de tarea: *Al iniciar sesión*

3. Elegir como acción que realizará la tarea *Iniciar un programa*:
4. El programa a ejecutarse será el bash de WSL y le pasamos como argumento el comando a ejecutar:

- Programa o script: `C:\Windows\System32\bash.exe`
- Argumentos: `-c "sudo /etc/init.d/ssh start"`

5. Revisar la configuración de la tarea y clic en *Finalizar*:
6. En la barra lateral izquierda, escogemos la opción *Biblioteca del Programador de tareas* y podremos ver listada nuestra nueva tarea:

32.3.3 Comprobar funcionamiento

Para comprobar que se está ejecutando el servicio de SSH de WSL al iniciar el sistema de Windows realizaremos los siguientes pasos:

1. Reiniciar nuestro sistema de Windows
2. Desde un equipo remoto haremos conexión SSH al usuario, IP y puerto pertenecientes al WSL:

```
ssh mogago@192.168.1.8 -p 2222
mogago@192.168.1.8 s password:
```

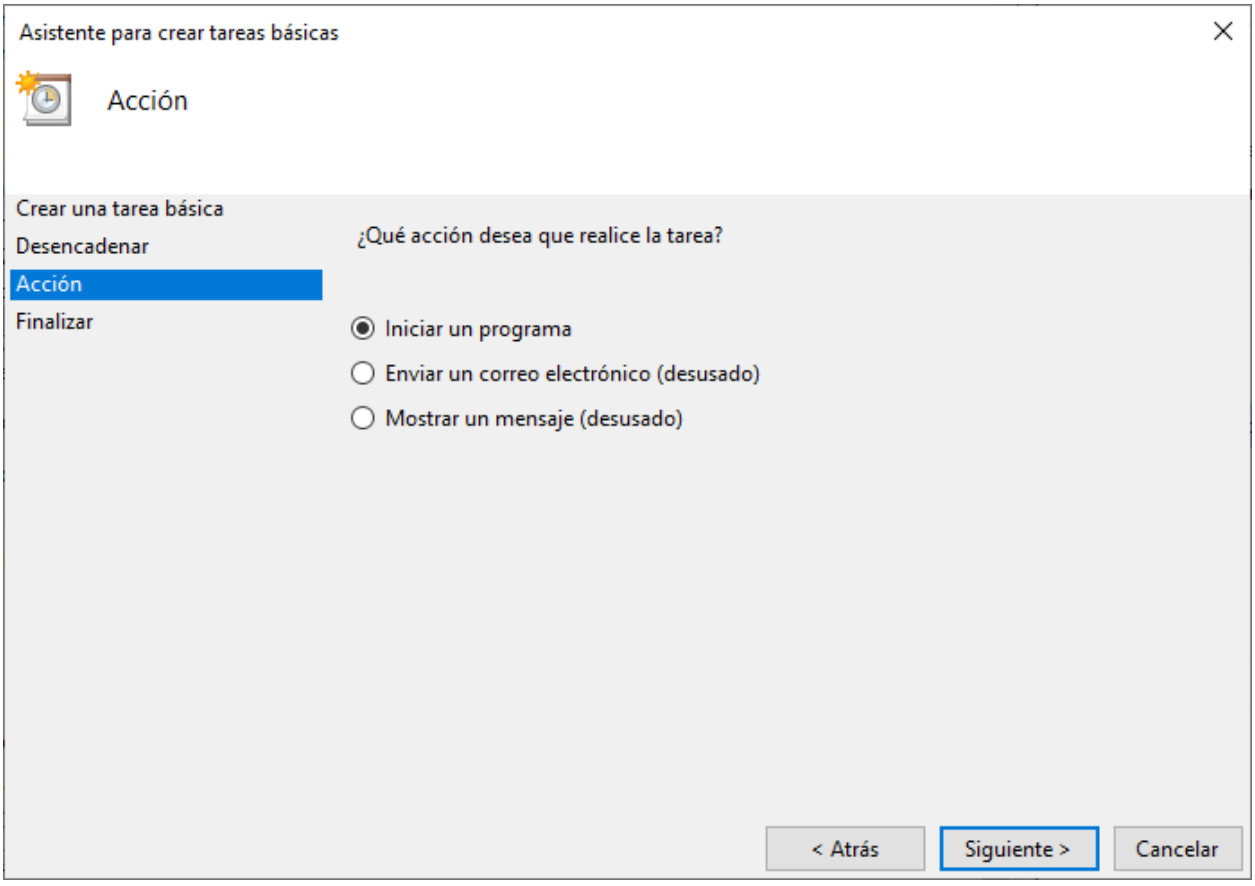


Fig. 19: Acción de la tarea: *Iniciar un programa*

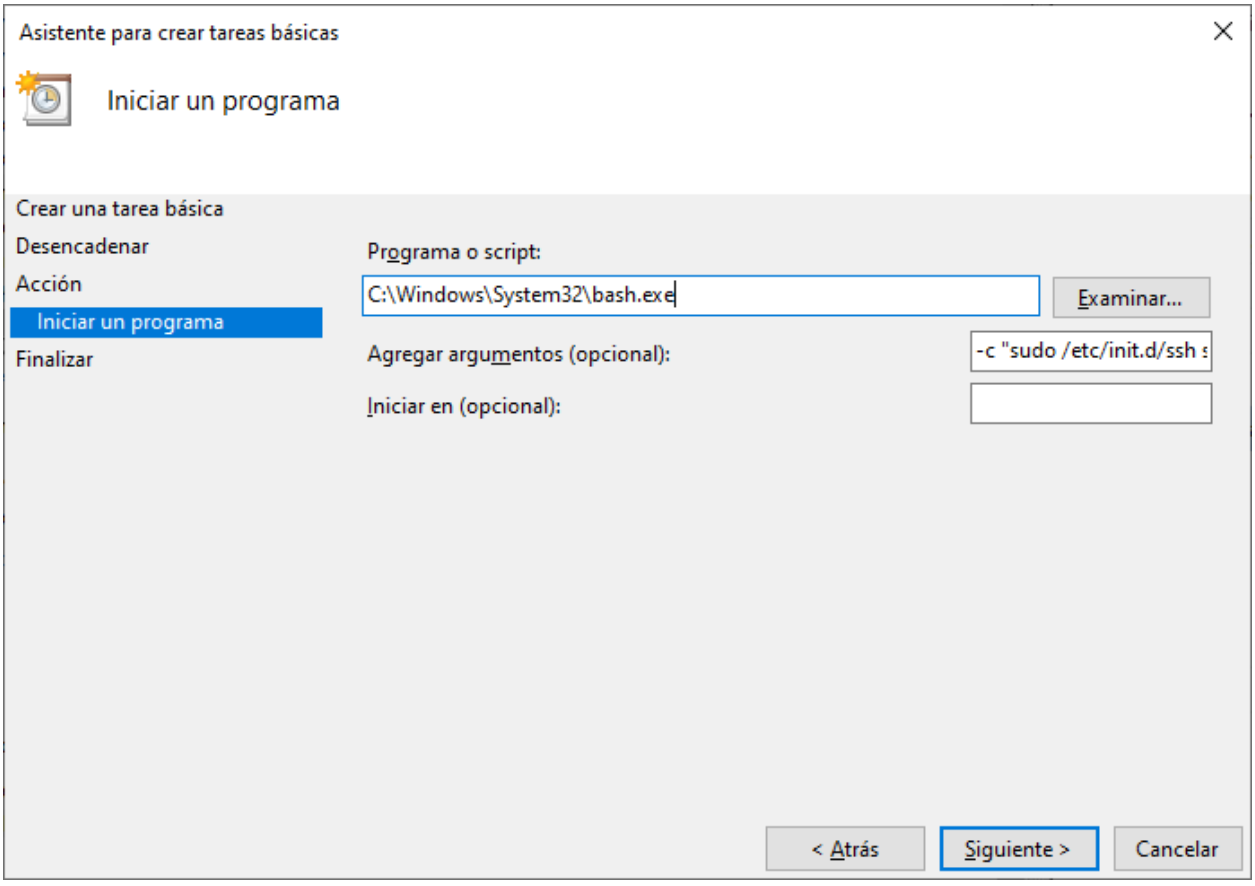


Fig. 20: Programa a ejecutar con sus argumentos

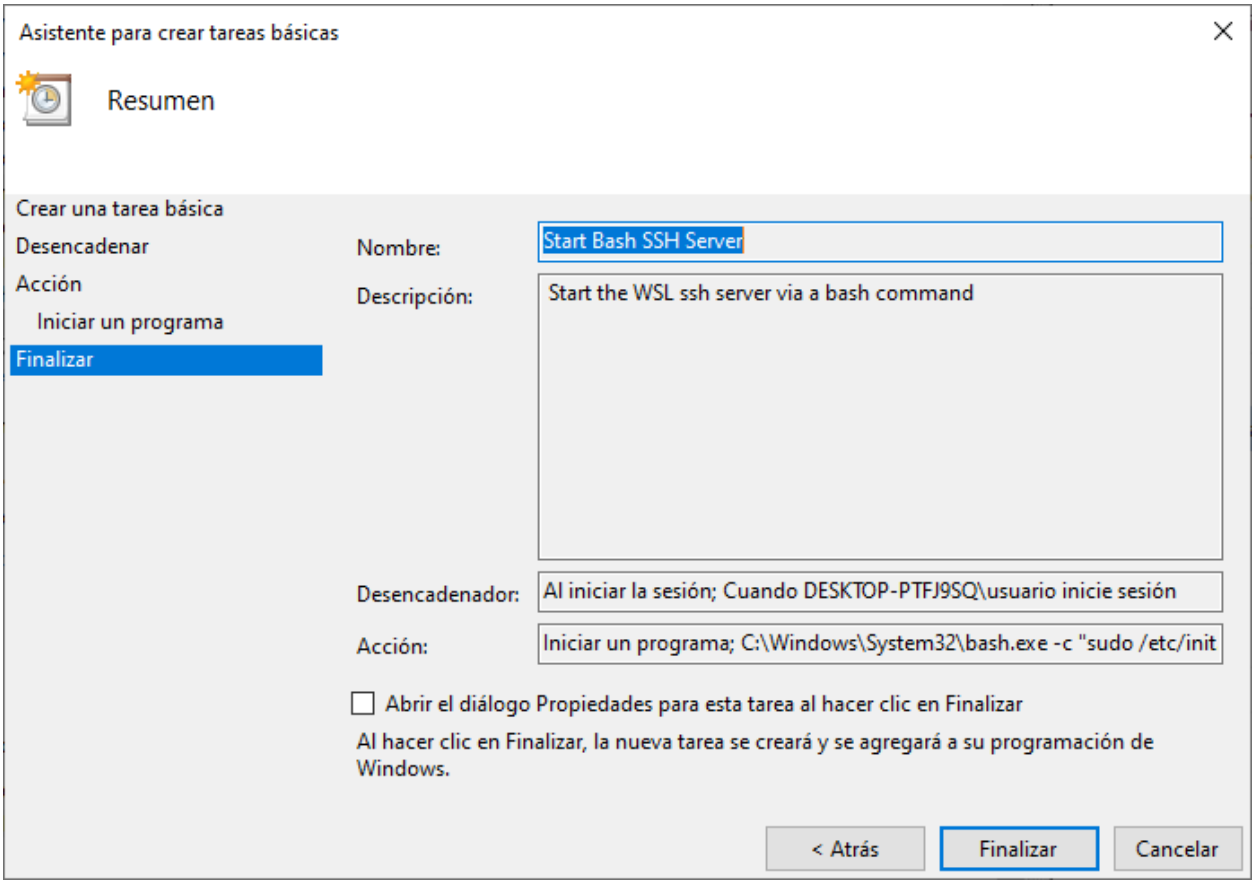


Fig. 21: Revisar el resumen de la tarea

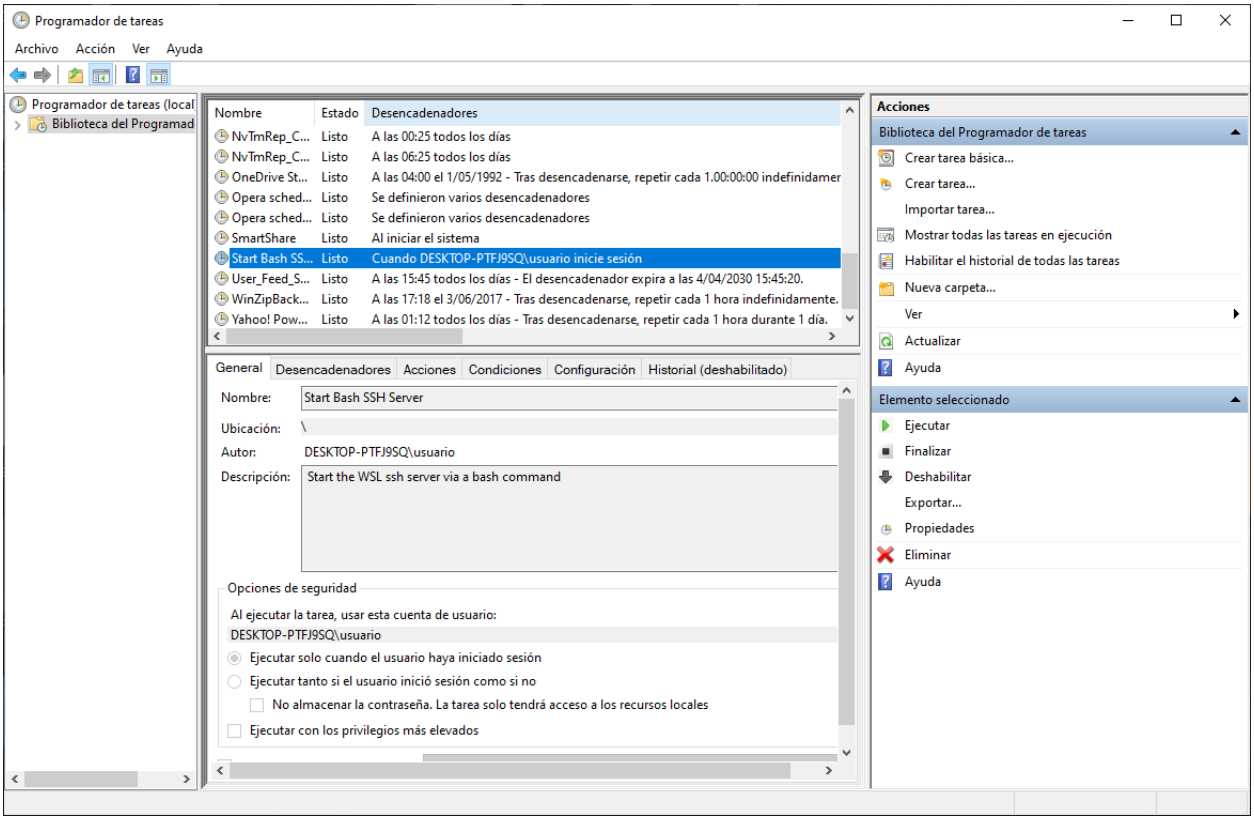


Fig. 22: Nueva tarea agregada a la Biblioteca

Si la conexión por SSH al equipo remoto ha sido exitosa, hemos configurado correctamente la tarea.

32.3.4 Referencias

- SSH on Windows Subsystem for Linux (WSL)

CHAPTER 33

SSH en Windows

33.1 Conectarse por SSH a Windows 10 (OpenSSH server)

Table of Contents

33.1.1 Instalación de OpenSSH Server

Tenemos 2 opciones para instalar OpenSSH Server, desde un entorno gráfico o con la línea de comandos.

Opción 1 - Usando interfaz gráfica

1. Abrir el programa *Aplicaciones y características* de Windows y elegir la opción *Características opcionales*:

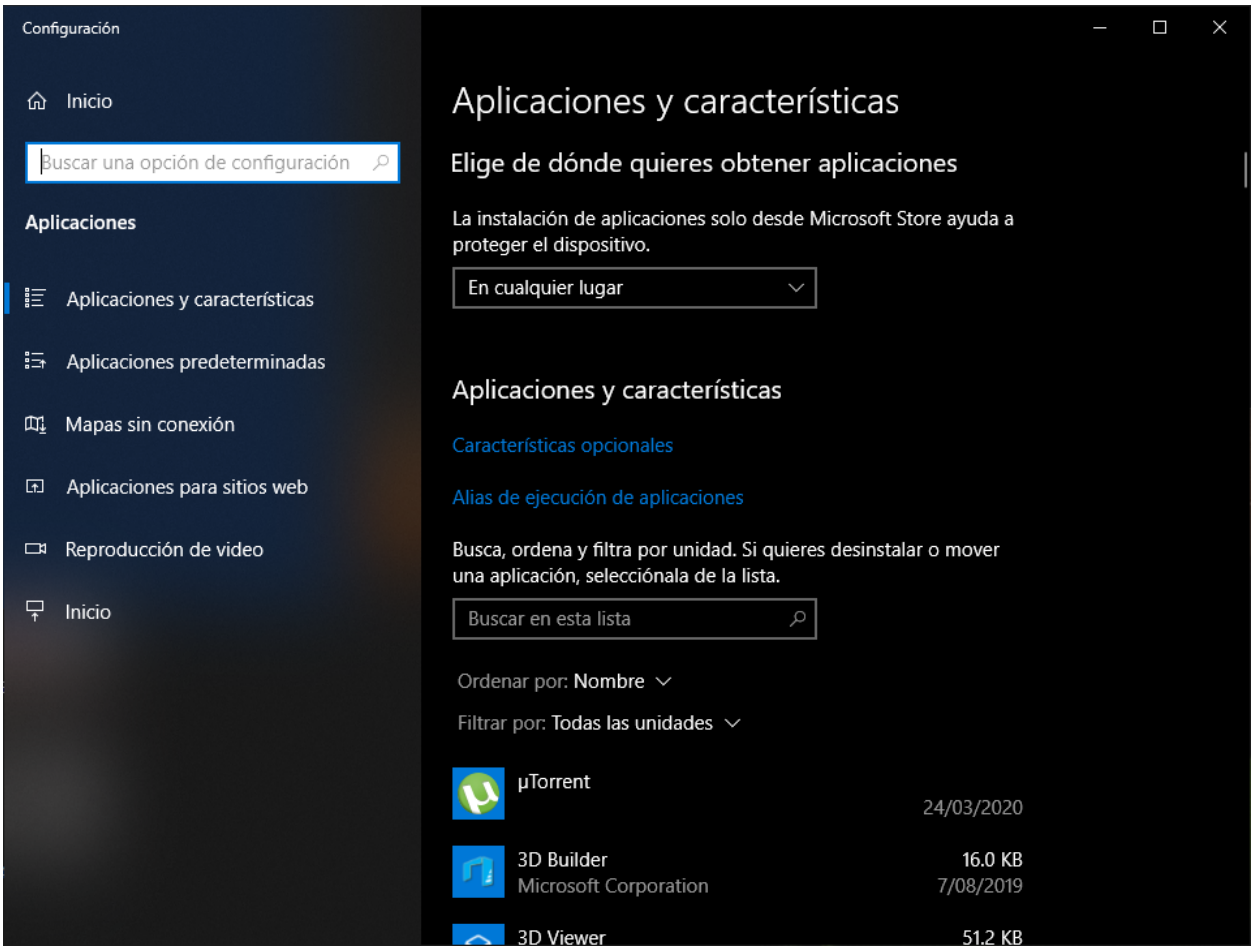


Fig. 1: Opción *Características opcionales* del programa *Aplicaciones y características*

2. Clic en + *Agregar una característica*:
3. Buscar *Servidor de OpenSSH* y clic en *Instalar*:
4. Comprobar que hemos agregado correctamente el *Servidor de OpenSSH*:

Opción 2 - Usando Powershell

Abrir el programa **Powershell** como administrador. Y ejecutar los siguientes comandos:

```
Add-WindowsCapability -Online -Name OpenSSH.Server~~~~0.0.1.0
Set-Service -Name ssh-agent -StartupType 'Automatic'
Set-Service -Name sshd -StartupType 'Automatic'
Get-Service ssh* | Start-Service
```

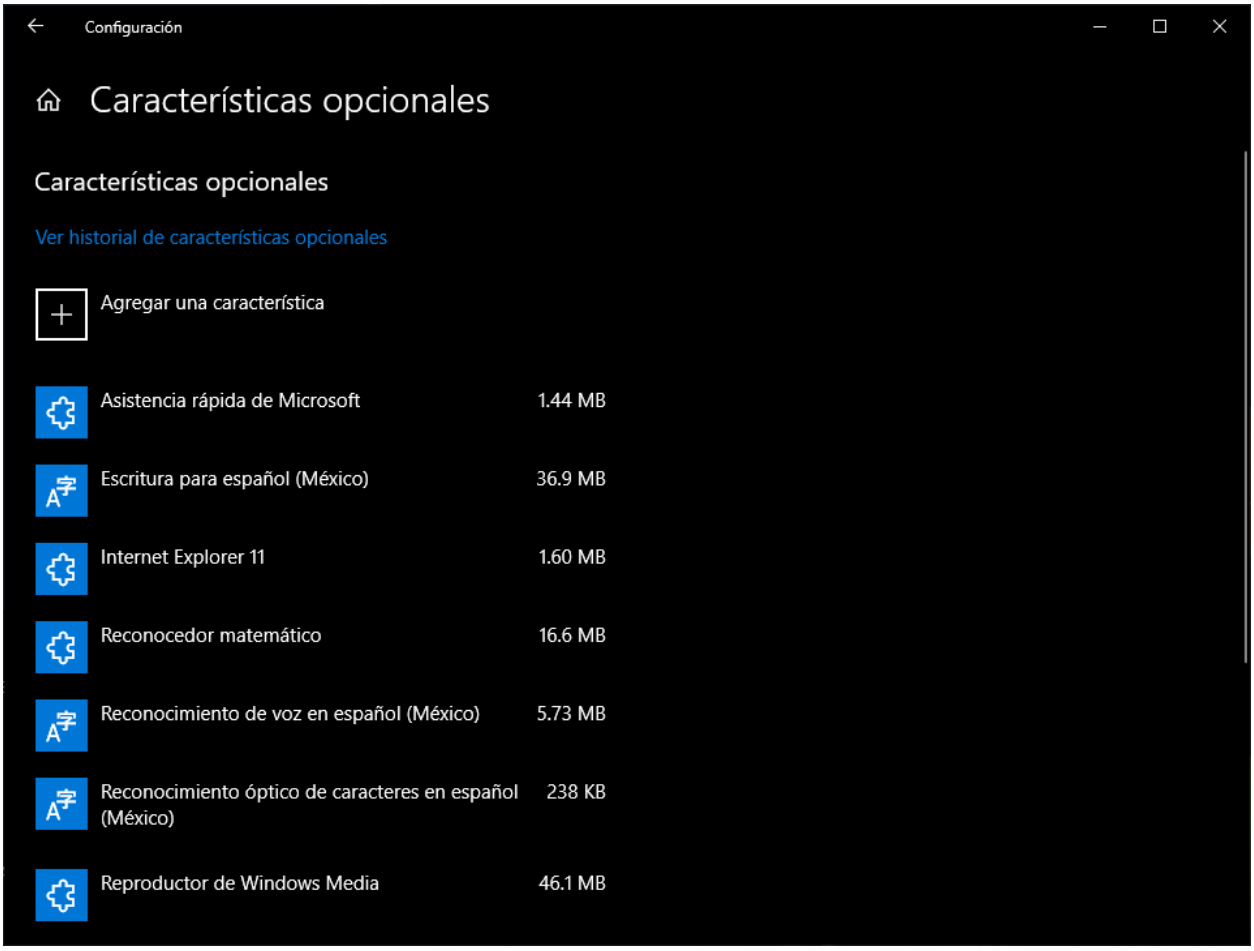


Fig. 2: Opción + *Agregar una característica*

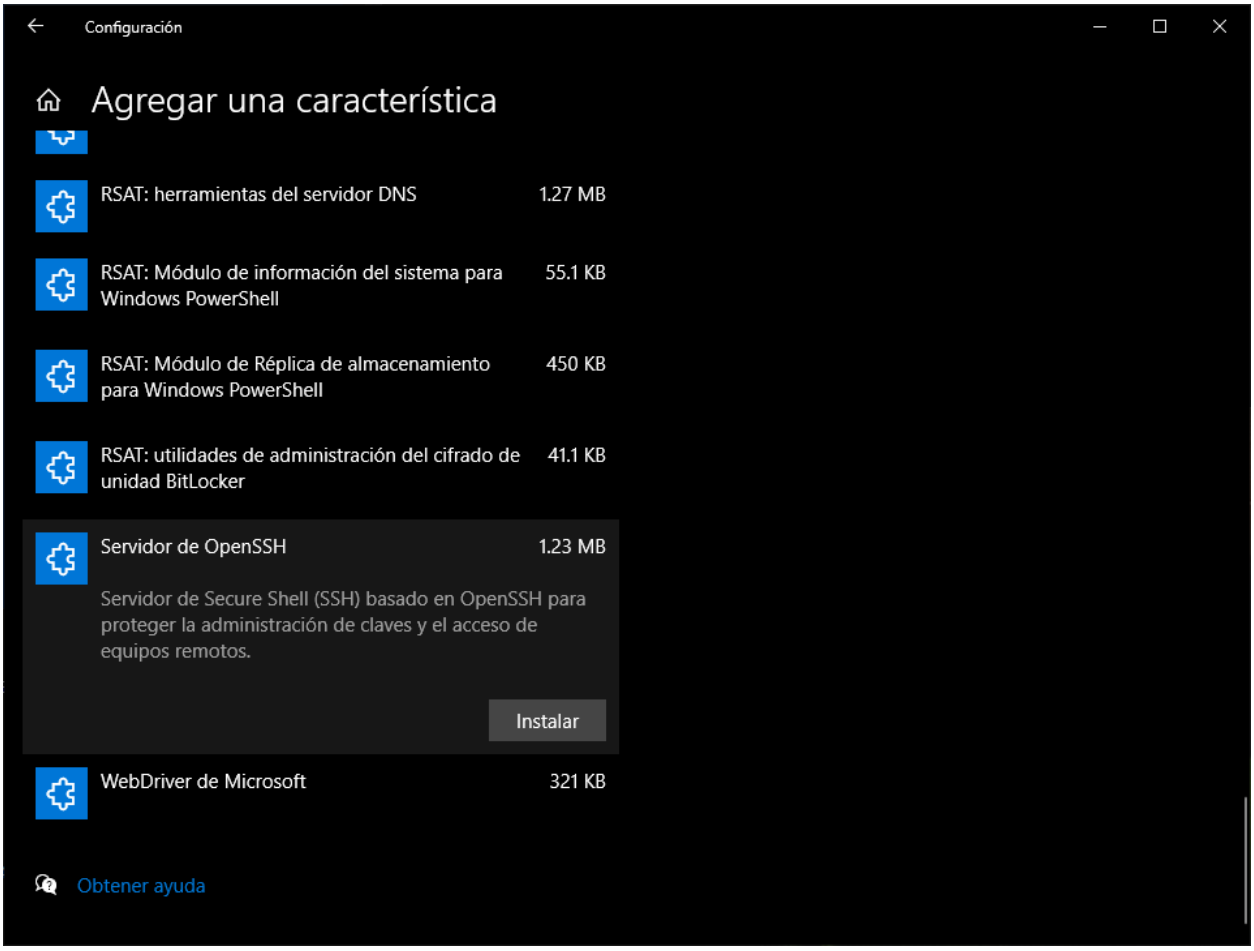


Fig. 3: En la característica *Servidor de OpenSSH*, opción *Instalar*:

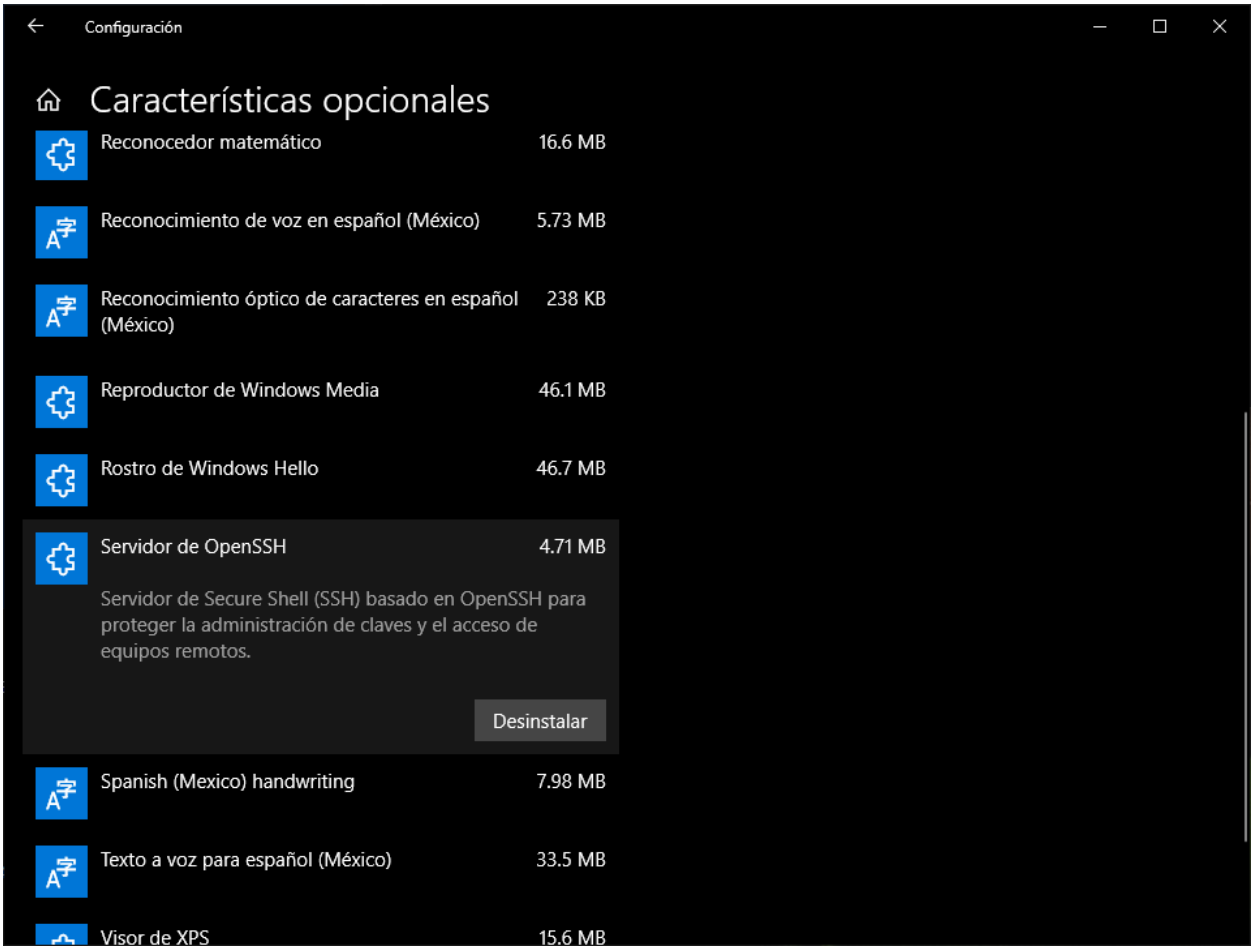


Fig. 4: Comprobar que está instalado el *Servidor de OpenSSH*:

33.1.2 Abrir el archivo `sshd_config` con un editor de texto

- Con un editor de texto editar el archivo `sshd_config`:

Usando Visual Studio Code

- Abrir **Powershell** en modo administrador y ejecutar los siguiente comandos:

```
cd C:\ProgramData\ssh\
code sshd_config
```

Usando Bloc de Notas

- Abrir **Powershell** en modo administrador y ejecutar los siguiente comandos:

```
cd C:\ProgramData\ssh\
notepad sshd_config
```

Usando WSL

- Abrir un terminal de Linux como administrador y ejeutar los siguientes comandos:

```
sudo su
cd /mnt/c/ProgramData/ssh/
```

```
vim sshd_config
```

33.1.3 Configuración del servidor OpenSSH

```
Port 2233
PasswordAuthentication yes
#PermitEmptyPasswords yes
```

Note: En caso, el usuario de Windows no tenga una contraseña establecida; en primer lugar, probar usando como contraseña el nombre de usuario. En caso no funcione, habilitar la línea `PermitEmptyPasswords yes` en el archivo `sshd_config` y reiniciar el servicio de SSH. Con esta última opción se habilita las conexiones SSH para usuario que no tiene contraseña.

33.1.4 Agregar una regla al Firewall de Windows

- Abrir el puerto `2233` en el firewall de Windows:

1. En el barra de navegación de Windows, abrir el programa *WF.msc* Seleccionar la opción *Reglas de entrada* en la barra izquierda:
2. Seleccionar la opción *Nueva regla...* en la barra derecha:
3. Seleccionar el tipo de regla *Puerto*:

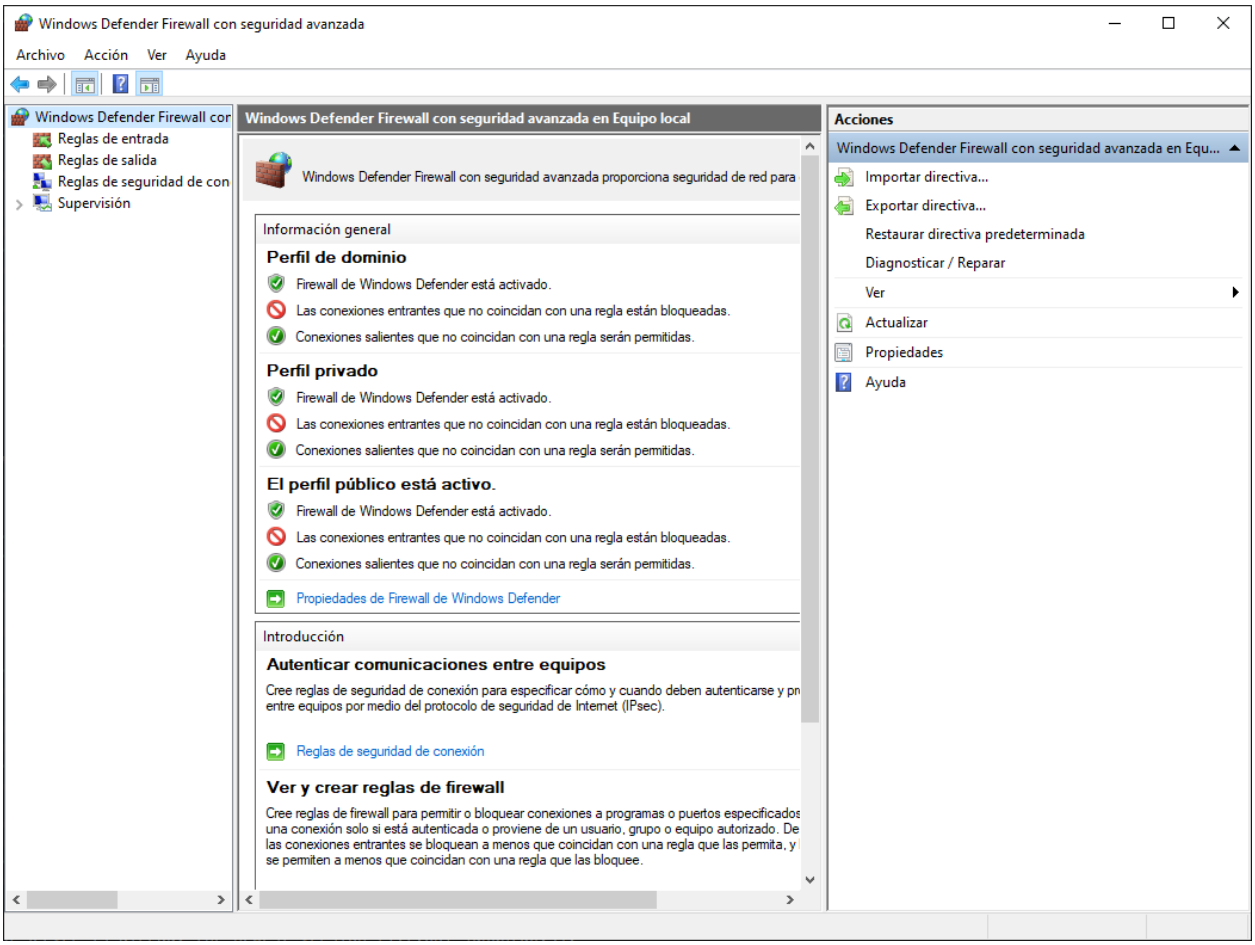


Fig. 5: Abrir el programa *WF.msc* de Windows, *Reglas de entrada*

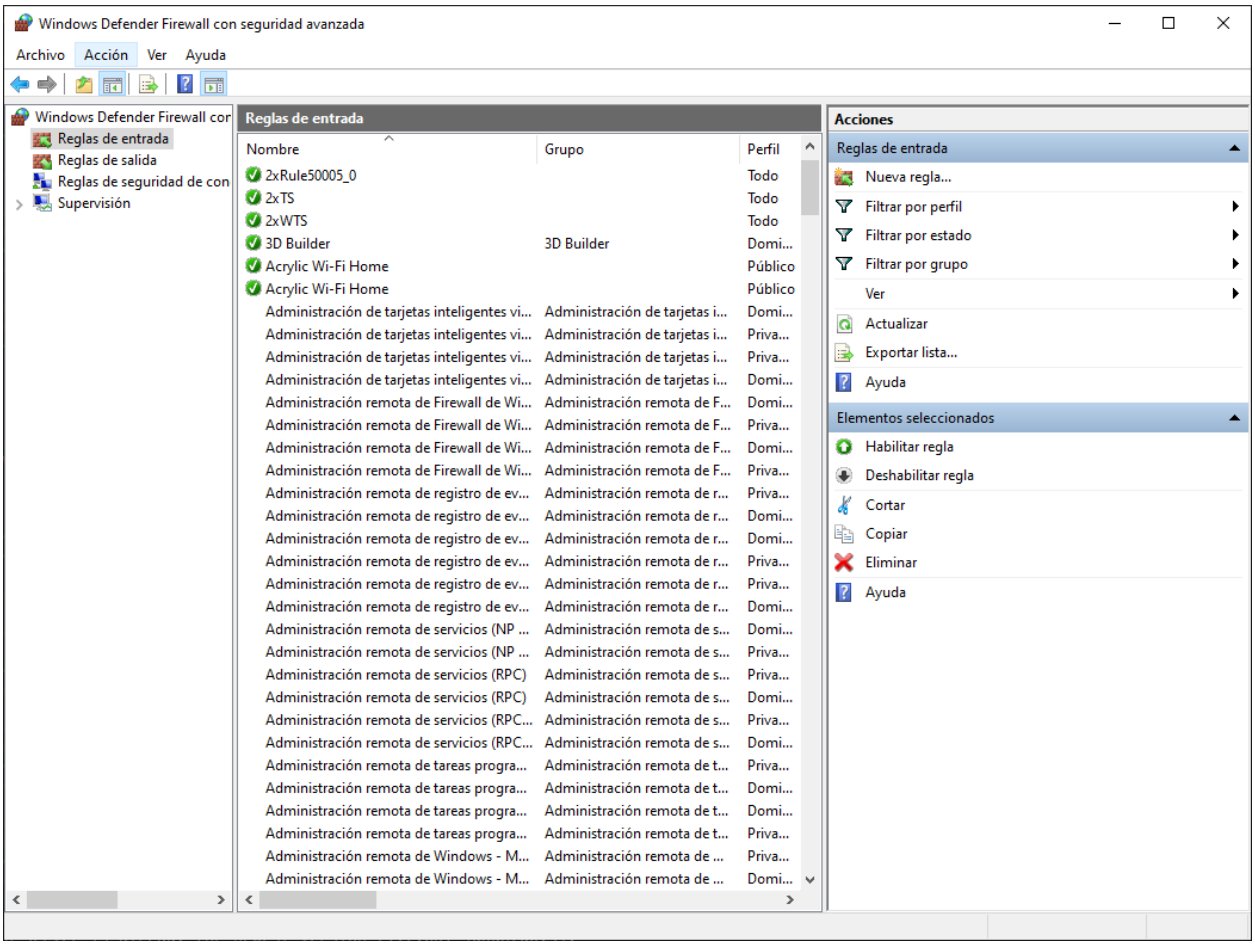


Fig. 6: Seleccionar la opción *Nueva regla...*

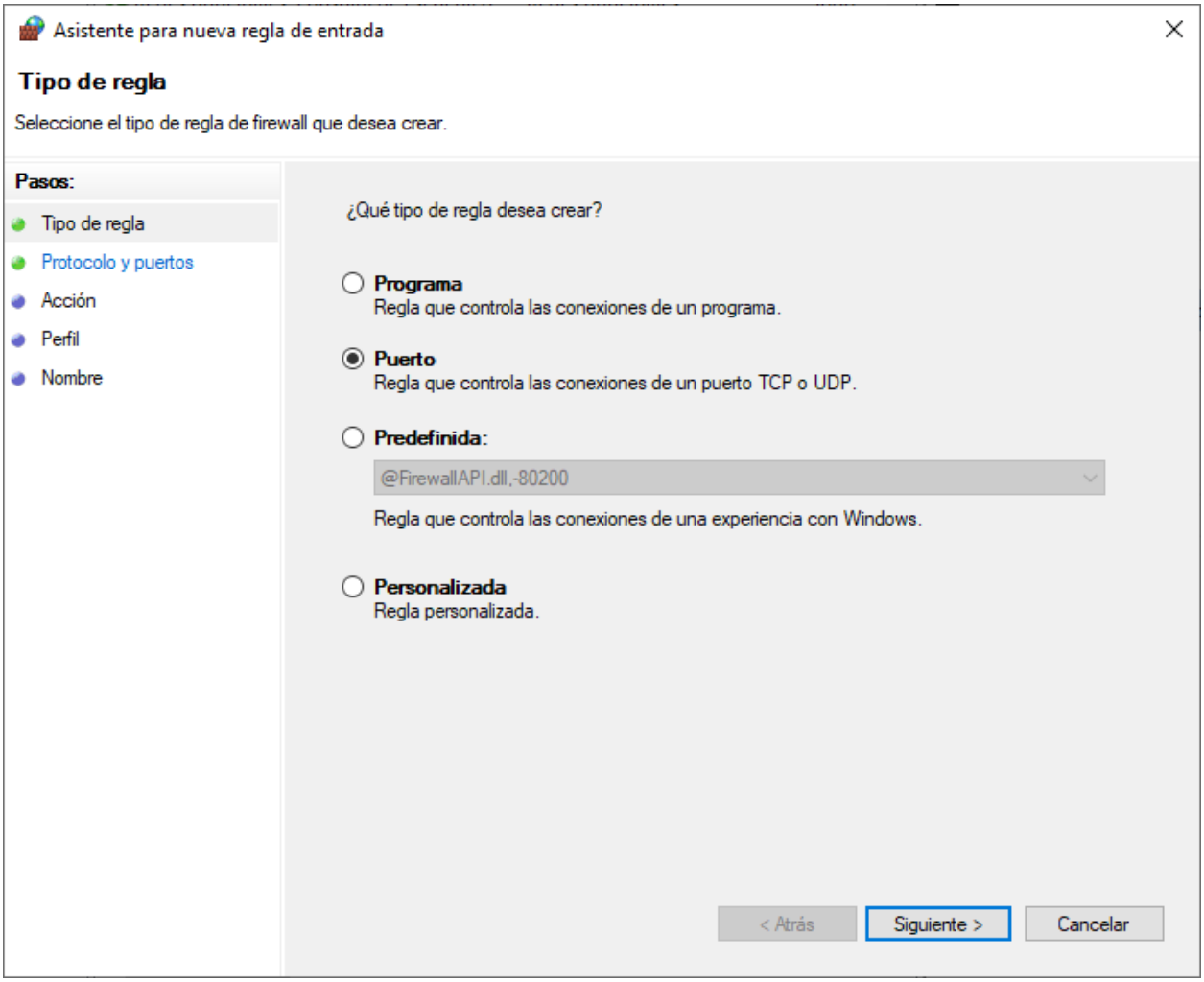


Fig. 7: Tipo de regla: *Puerto*

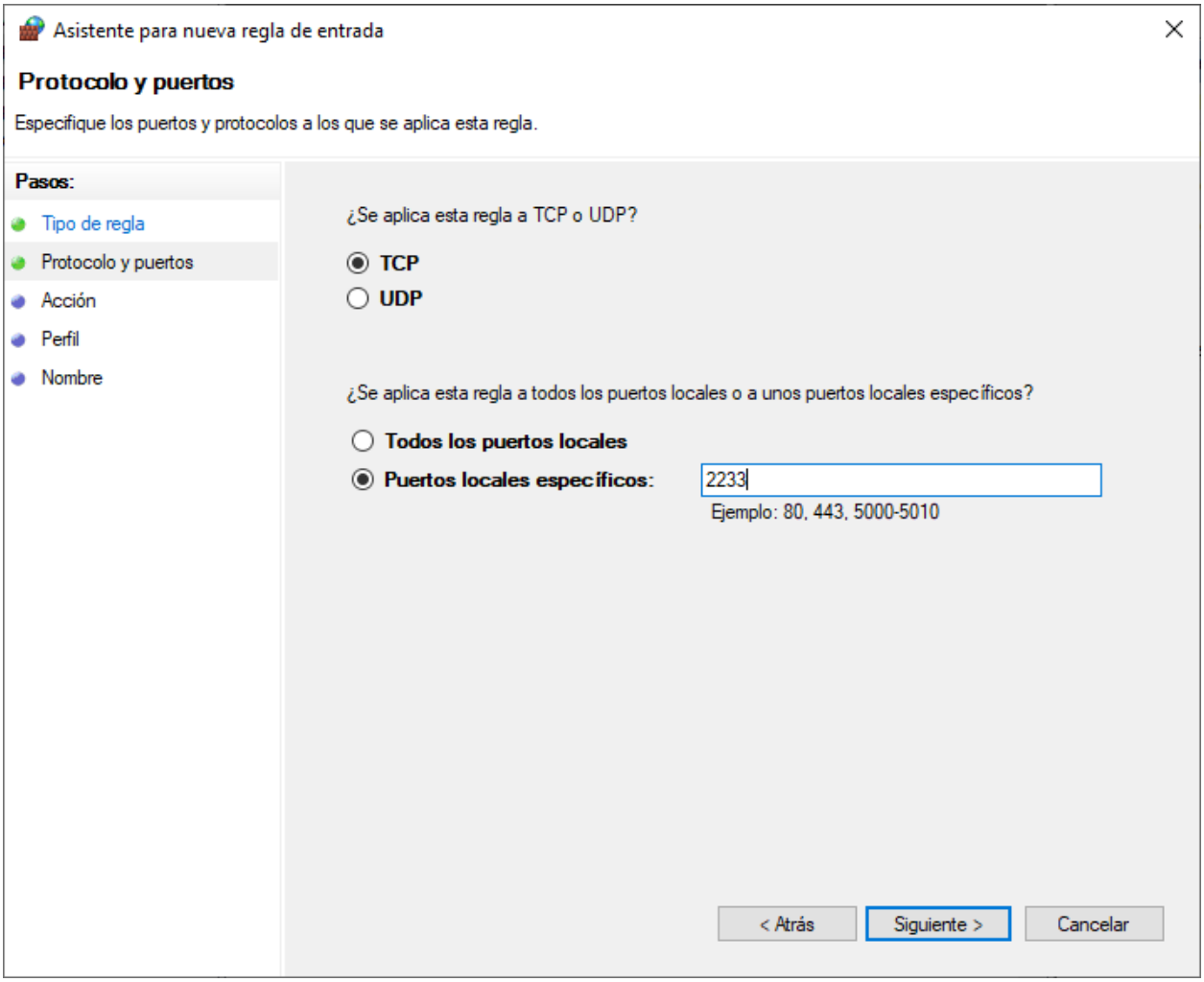


Fig. 8: Aplicar la regla a *TCP*, *Puertos locales específicos: 2233*

4. Aplicar la regla a *TCP* y usar la opción *Puertos locales específicos:* y escribir *2233*:
5. Elegir la opción *Permitir la conexión*:

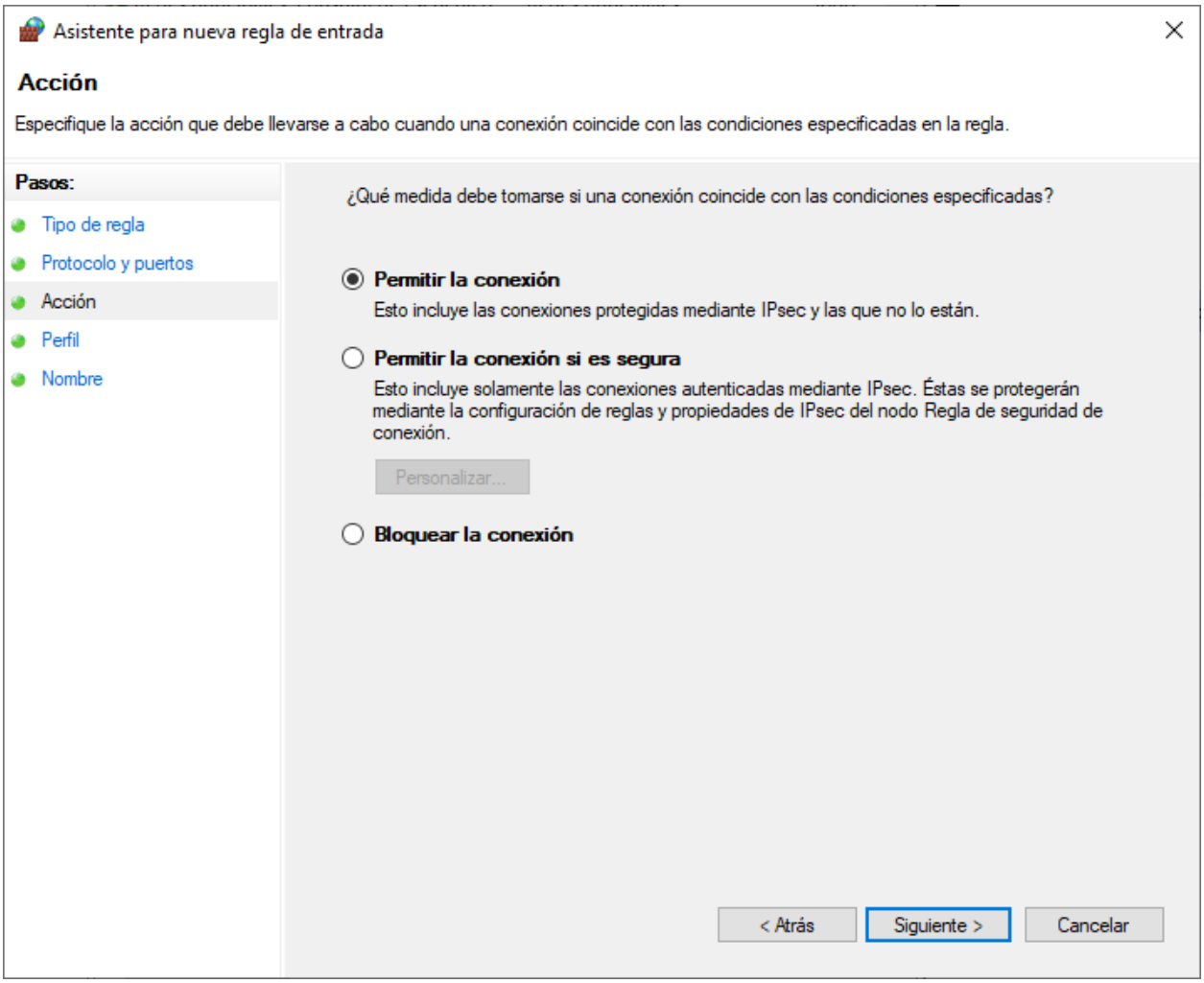


Fig. 9: Seleccionar *Permitir la conexión*

6. Aplicar la regla sobre todas las opciones: *Dominio*, *Privado*, *Público*
7. Dar un nombre y descripción a la nueva regla de firewall creada:

33.1.5 Reiniciar el servicio de SSH

```
Restart-Service sshd
```

- Verificar que se encuentre escuchando en el puerto `2233`:

```
netstat -aon | findstr "2233"
  TCP    0.0.0.0:2233           0.0.0.0:0              LISTENING       9972
  TCP    [::]:2233              [::]:0                 LISTENING       9972
```

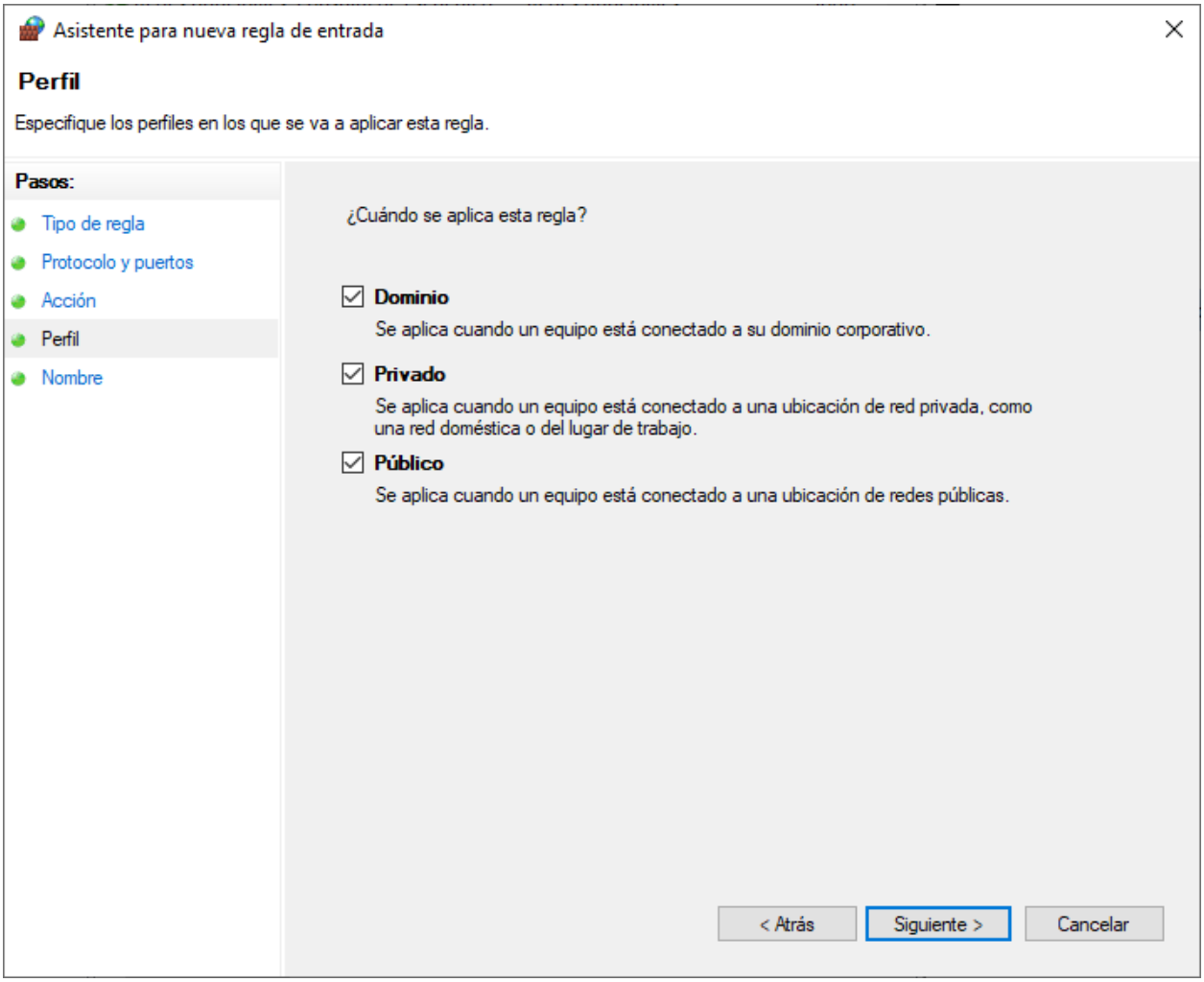


Fig. 10: Seleccionar *Permitir la conexión*

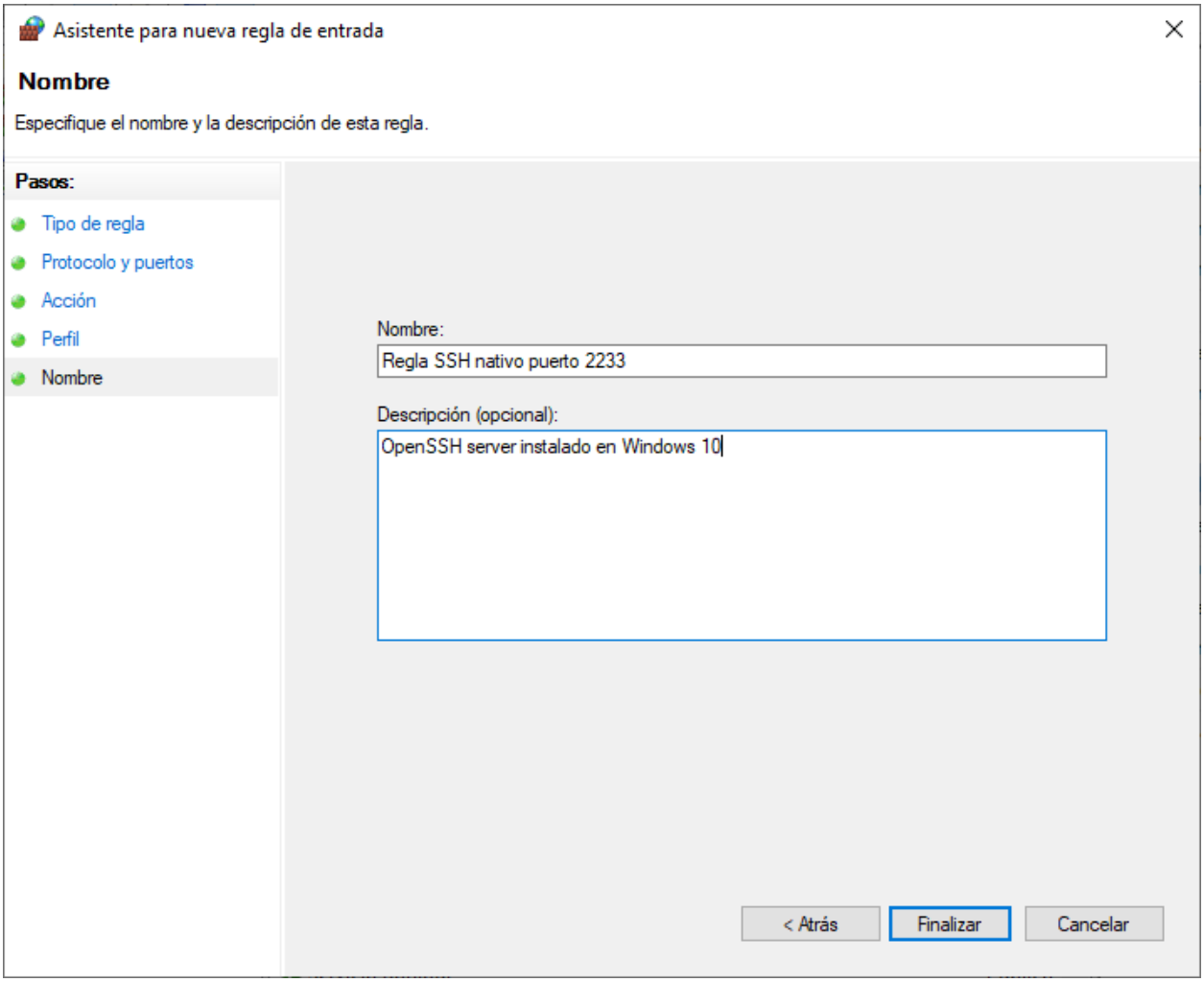


Fig. 11: Nombre y descripción de la regla de firewall

33.1.6 Conexión remota por SSH

- Desde un equipo remoto conectarnos por SSH al sistema Windows usando un terminal o PuTTY, apuntando a la IP del sistema Windows y el puerto `2233`. Por ejemplo para conectarnos por SSH desde un terminal de Linux usamos:

```
$ ssh usuario@192.168.1.8 -p 2233

usuario@192.168.1.8 s password:
Microsoft Windows [Versión 10.0.18362.720]
(c) 2019 Microsoft Corporation. Todos los derechos reservados.

usuario@DESKTOP-PTFJ9SQ C:\Users\usuario>
```

33.1.7 Referencias

- Instalación y configuración de OpenSSH Server en Windows Server 2019
- How To Install OpenSSH On Windows 10
- OpenSSH windows 10 user s password not configured
- PowerShell remoting over SSH
- OpenSSH Key Management