



Curso FORPAS 2023:

**Automatización de tareas en Windows con
Gorilla**

Creación de un entorno de trabajo virtualizado con
Multipass

Formadores:

Luis Vela Morilla

Juan Antonio Fernández Ruíz

Contenido

1 - Montar un servidor de web en un entorno de pruebas virtualizado con Multipass	2
Si en lugar de utilizar el servidor web proporcionado en el curso (luivelmor.net), queremos crear nuestro propio servidor y realizar pruebas en él podemos utilizar varias opciones: VirtualBox, VMWare, o la que vamos a detallar a continuación y que considero la más simple y rápida de todas, Multipass.....	
2 - Qué es Multipass.....	2
2.1 - Instalación de Multipass sobre distintos sistemas operativos.....	2
3 - Comandos de Multipass (multipass --help).....	4
4 - Imágenes existentes en la nube de canonical: multipass find	5
5 - Crear una instancia de una máquina virtual: multipass launch	5
5.1 – Opción 1:	5
5.2 – Opción 2 (definiendo el hardware):	5
6 - Listar las instancias de máquinas virtuales creadas: multipass list y multipass info	6
7 - Abrir la Shell de una máquina virtual existente: multipass shell	6
8 – Ejecutar comandos en la máquina virtual existente: multipass exec	7
9 - Compartir una carpeta entre el host y la instancia de multipass: multipass mount	7
10 – Multipass e Hyper-V	8
11 - Instalamos el servidor web (apache2)	9
11.1 – Desde dentro de la máquina virtual:.....	9
11.2 – Desde fuera de la máquina virtual:	9
12 - IP del adaptador de red de la instancia multipass	9
13 - Configuración de las carpetas del servidor de gorilla	10
14 - Creación de un manifest y catalog inicial.....	11
15 - Configuramos un cliente de gorilla	11
16 - Prueba de ejecución en el cliente	12
17 - Conclusiones	13

1 - Montar un servidor de web en un entorno de pruebas virtualizado con Multipass

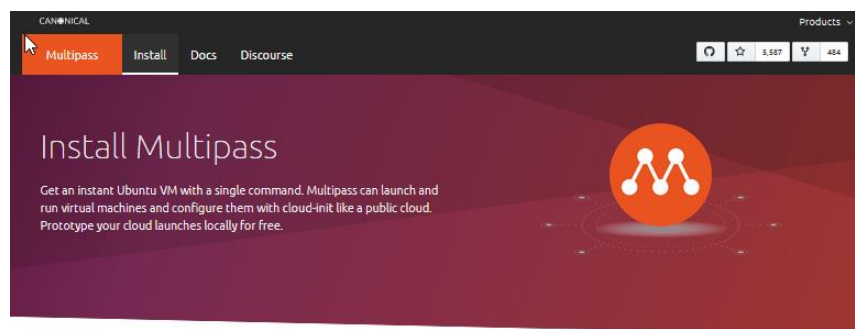
Si en lugar de utilizar el servidor web proporcionado en el curso (luivelmor.net), queremos crear nuestro propio servidor y realizar pruebas en él podemos utilizar varias opciones: VirtualBox, VMWare, o la que vamos a detallar a continuación y que considero la más simple y rápida de todas, Multipass.

2 - Qué es Multipass

Multipass es un proyecto de Canonical (Ubuntu) que nos permite crear y lanzar máquinas virtuales mediante línea de comandos. Puede instalarse en Linux, Ubuntu o MacOS.

2.1 - Instalación de Multipass sobre distintos sistemas operativos

Nos dirigimos a <https://multipass.run> y pulsamos sobre "Install now". A continuación, seleccionamos cuál es nuestro sistema host (en mi caso, Windows):



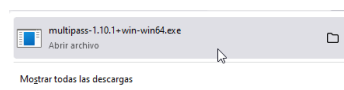
Select OS to get started



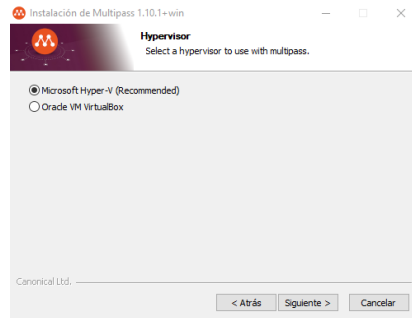
Install Multipass on Windows



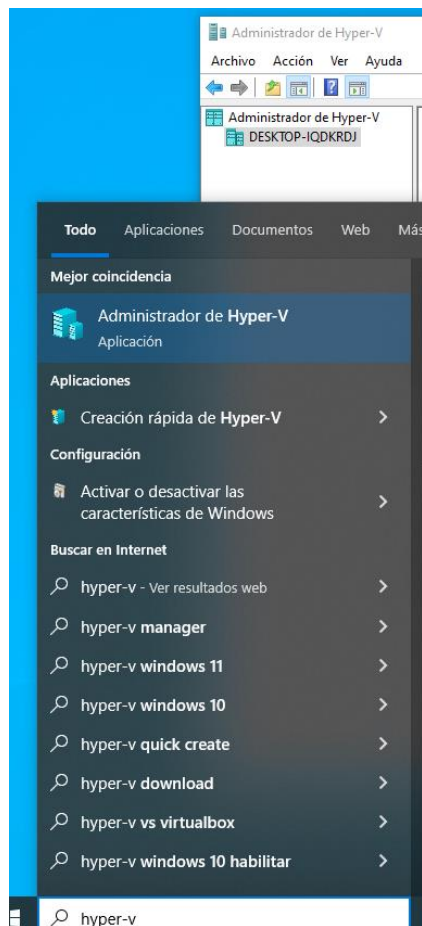
Una vez descargado el fichero .exe, lo ejecutamos:



Ubuntu Multipass es un gestor de máquinas virtuales, pero necesita apoyarse sobre un hipervisor. En este caso, detecta 2 en nuestro sistema, Hyper-V que pertenece al propio sistema operativo Windows y VirtualBox. Elegiremos Hyper-V.



Podemos gestionar todas las máquinas virtuales que vayamos creando desde El Administrador de Hyper-V:



3 - Comandos de Multipass (multipass --help)

Referencia: [Multipass Tutorial - Windows](#) | [Multipass documentation](#)

Multipass crea y trabaja con máquinas virtual al igual que lo haría VirtualBox o VMWare, pero mediante línea de comandos. Es bastante sencillo. En primer lugar, abrimos una instancia de Powershell en Windows:

- 1- Mediante “multipass --help” vemos todas las opciones que nos permite realizar la herramienta

```
Administrador: Windows PowerShell
PS C:\Users\luivelmor> multipass --help
Usage: program files\multipass\bin\multipass.exe [options] <command>
Create, control and connect to Ubuntu instances.

This is a command line utility for multipass, a
service that manages Ubuntu instances.

Options:
  -?, -h, --help    Displays help on commandline options
  -v, --verbose      Increase logging verbosity. Repeat the 'v' in the short
                    option for more detail. Maximum verbosity is obtained with 4
                    (or more) v's, i.e. -vvvv.

Available commands:
  alias             Create an alias
  aliases           List available aliases
  authenticate      Authenticate client
  delete            Delete instances
  exec              Run a command on an instance
  find              Display available images to create instances from
  get               Get a configuration setting
  help              Display help about a command
  info              Display information about instances
  launch            Create and start an Ubuntu instance
  list              List all available instances
  mount             Mount a local directory in the instance
  networks          List available network interfaces
  purge             Purge all deleted instances permanently
  recover           Recover deleted instances
  restart           Restart instances
  set               Set a configuration setting
  shell             Open a shell on a running instance
  start             Start instances
  stop              Stop running instances
  suspend           Suspend running instances
  transfer          Transfer files between the host and instances
  umount            Unmount a directory from an instance
  unalias           Remove aliases
  version           Show version details
PS C:\Users\luivelmor>
```

4 - Imágenes existentes en la nube de canonical: multipass find

El comando “multipass find” nos muestra todas las imágenes que ya EXISTEN en la nube y con las que podemos crear + lanzar una máquina virtual.

Es decir, a diferencia de VirtualBox, por ejemplo, no necesitamos descargar la .iso de Ubuntu, asignársela a una unidad de CDROM y posteriormente instalar el sistema, sino que en multipass ya nos descargamos el sistema Ubuntu completamente operativo. Hay que recordar que multipass es un proyecto de Canonical (Ubuntu) y por tanto todas las imágenes disponibles son las de los sistemas operativos asociados a ellos.

```
PS C:\Users\luivelmor> multipass find
```

Image	Alias	Version	Description
core	core16	20200818	Ubuntu Core 16
core18		20211124	Ubuntu Core 18
snapcraft:core18	18.04	20201111	Snapcraft builder for Core 18
snapcraft:core20	20.04	20210921	Snapcraft builder for Core 20
snapcraft:core22	22.04	20220426	Snapcraft builder for Core 22
18.04	bionic	20221117	Ubuntu 18.04 LTS
20.04	focal	20221115.1	Ubuntu 20.04 LTS
22.04	jammy,lts	20221117	Ubuntu 22.04 LTS
appliance:adguard-home		20200812	Ubuntu AdGuard Home Appliance
appliance:mosquitto		20200812	Ubuntu Mosquitto Appliance
appliance:nextcloud		20200812	Ubuntu Nextcloud Appliance
appliance:openhab		20200812	Ubuntu openHAB Home Appliance
appliance:plexmediaserver		20200812	Ubuntu Plex Media Server Appliance
anbox-cloud-appliance		latest	Anbox Cloud Appliance
charm-dev		latest	A development and testing environment
docker		latest	A Docker environment with Portainer
jellyfin		latest	Jellyfin is a Free Software Media System
aming your media.			
minikube		latest	minikube is local Kubernetes

```
PS C:\Users\luivelmor>
```

5 - Crear una instancia de una máquina virtual: multipass launch

De la lista de arriba, seleccionamos la imagen Ubuntu 22.04 LTS cuyo alias son “jammy” y “lts”. Ahora, creamos y ejecutamos una instancia de dicho sistema operativo en multipass mediante:

5.1 – Opción 1:

```
PS C:\Users\luivelmor> multipass launch lts --name gorillaserver
Retrieving image: 54%
```

5.2 – Opción 2 (definiendo el hardware):

```
PS C:\Windows\system32> multipass launch lts --name gorillaserver --mem 2G --disk 10G --cpus 2
Starting gorillaserver \
```

6 - Listar las instancias de máquinas virtuales creadas: multipass list y multipass info

Con el comando “multipass list” podemos listar todas las máquinas virtuales creadas con multipass y ver su estado actual, versión del SSOO o la IP que tiene y a través de la cual puede comunicarse con el sistema host (equivalente al adaptador sólo-anfitrión de VirtualBox).

```
Administrador: Windows PowerShell
PS C:\Windows\system32> multipass list
Name                State      IPv4          Image
gorillaserver       Running    172.31.68.56  Ubuntu 22.04 LTS
ltsInstance         Deleted    --            Not Available
PS C:\Windows\system32>
```

```
Administrador: Windows PowerShell
PS C:\Windows\system32> multipass info gorillaserver
Name:                gorillaserver
State:               Running
IPv4:                172.31.68.56
Release:             Ubuntu 22.04.1 LTS
Image hash:          3100a27357a0 (Ubuntu 22.04 LTS)
Load:                0.05 0.03 0.01
Disk usage:          1.4G out of 9.5G
Memory usage:        192.4M out of 1.9G
Mounts:              --
PS C:\Windows\system32>
```

7 - Abrir la Shell de una máquina virtual existente: multipass shell

“multipass shell gorillaserver” nos permite abrir una Shell hacia la máquina virtual cuyo nombre es “gorillaserver”, es decir, la creada anteriormente.

```
ubuntu@gorillaserver:~$
PS C:\Windows\system32> multipass list
Name                State      IPv4          Image
gorillaserver       Running    172.31.68.56  Ubuntu 22.04 LTS
ltsInstance         Deleted    --            Not Available
PS C:\Windows\system32> multipass shell gorillaserver
Welcome to Ubuntu 22.04.1 LTS (GNU/Linux 5.15.0-57-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:    https://landscape.canonical.com
 * Support:       https://ubuntu.com/advantage

System information as of Fri Jan 20 10:54:58 CET 2023

System load:  0.01220703125   Processes:            105
Usage of /:   14.8% of 9.51GB Users logged in:             0
Memory usage: 12%            IPv4 address for eth0: 172.31.68.56
Swap usage:   0%

0 updates can be applied immediately.

The list of available updates is more than a week old.
To check for new updates run: sudo apt update

To run a command as administrator (user "root"), use "sudo <command>".
See "man sudo_root" for details.

ubuntu@gorillaserver:~$
```

8 – Ejecutar comandos en la máquina virtual existente: multipass exec

“multipass exec [name] [command]” nos permite ejecutar un comando en la máquina virtual cuyo nombre es “name” y cuyo comando es “command. Para ver su sintaxis podemos escribir “multipass exec --help”

```
Administrador: Windows PowerShell
PS C:\Windows\system32> multipass exec gorillaserver -- sudo apt install apache2
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following additional packages will be installed:
  apache2-bin apache2-data apache2-utils bzip2 libapr1 libaprutil1 libaprutil1-dbd-sqlite3
  libaprutil1-ldap liblua5.3-0 mailcap mime-support ssl-cert
Suggested packages:
  apache2-doc apache2-suexec-pristine | apache2-suexec-custom www-browser bzip2-doc
The following NEW packages will be installed:
  apache2 apache2-bin apache2-data apache2-utils bzip2 libapr1 libaprutil1
  libaprutil1-dbd-sqlite3 libaprutil1-ldap liblua5.3-0 mailcap mime-support ssl-cert
0 upgraded, 13 newly installed, 0 to remove and 0 not upgraded.
Need to get 2136 kB of archives.
After this operation, 8505 kB of additional disk space will be used.
Do you want to continue? [Y/n]
```

9 - Compartir una carpeta entre el host y la instancia de multipass: multipass mount

Ejecutamos la siguiente instrucción para permitir los puntos de montaje entre el host y las instancias de multipass:

```
PS C:\Windows\system32> multipass set local.privileged-mounts=Yes
```

A continuación, entramos en la multipass mediante “multipass shell gorillaserver” y creamos el directorio donde queremos montar el directorio compartido:

```
ubuntu@gorillaserver:~$ sudo mkdir /var/www/html/shared_with_multipass
```

Por último, salimos de la instancia o abrimos una nueva terminal de Powershell y ejecutamos las siguientes instrucciones para compartir el directorio host en directorio creado anteriormente. Para terminar, comprobamos con “multipass exec”:

```
Administrador: Windows PowerShell
PS C:\Users\luivelmor\Desktop> $dir = "C:\Users\luivelmor\Desktop\shared_with_multipass\"
PS C:\Users\luivelmor\Desktop> multipass mount $dir gorillaserver:/var/www/html
PS C:\Users\luivelmor\Desktop> multipass exec gorillaserver -- ls /var/www/html
asdasd.txt
PS C:\Users\luivelmor\Desktop> ls .\shared_with_multipass\

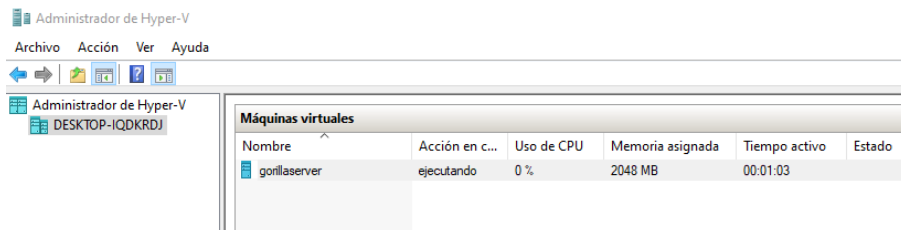
Directorio: C:\Users\luivelmor\Desktop\shared_with_multipass

Mode                LastWriteTime         Length Name
----                -
-a-----         16/01/2023    7:56             0 asdasd.txt

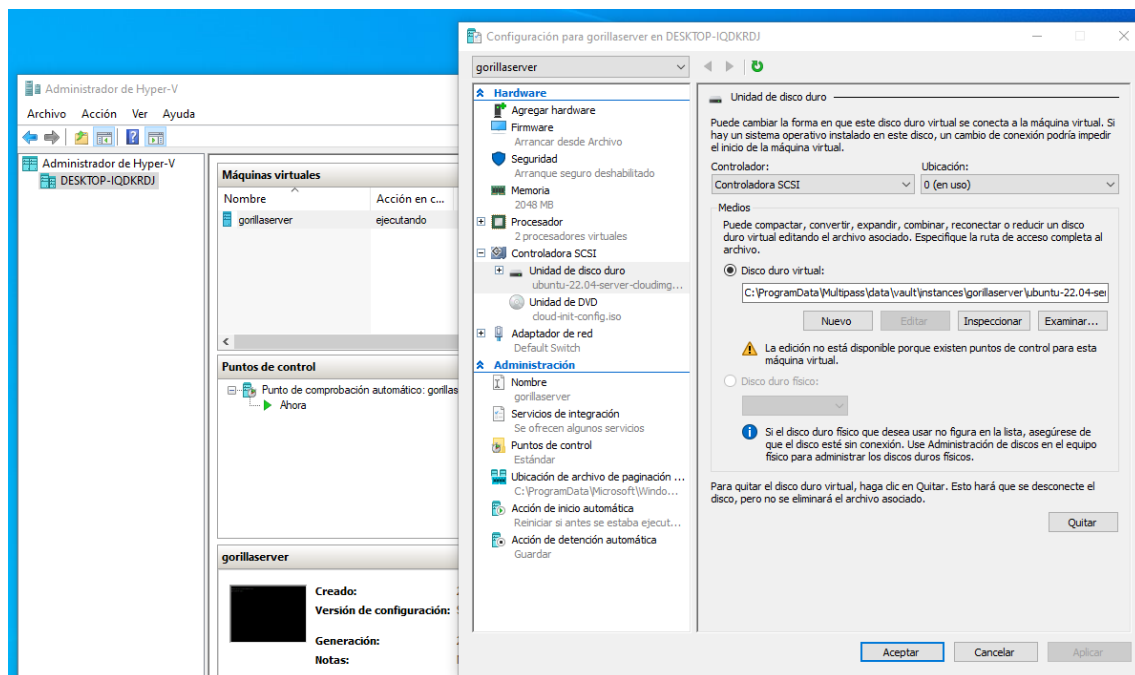
PS C:\Users\luivelmor\Desktop>
```


10 – Multipass e Hyper-V

Tras las operaciones realizadas, si abrimos el administrador de Hyper-v de Windows, podremos ver la máquina virtual creada.



Si hacemos clic derecho sobre la máquina y pulsamos en configuración, podemos ver todas las opciones de configuración de la máquina, al igual que en cualquier otro software como VirtualBox o VMWare. Mostraremos especial atención al apartado “Disco duro virtual”, para conocer la ubicación de las máquinas creadas por Multipass.



11 - Instalamos el servidor web (apache2)

11.1 – Desde dentro de la máquina virtual:

Administrador: Windows PowerShell

```
PS C:\Users\luivelmor\Desktop> multipass shell gorillaserver
```

Una vez dentro de la Shell de la máquina virtual creada, instalamos el servidor de apache mediante la siguiente instrucción:

```
ubuntu@gorillaserver:~$ sudo apt install apache2
```

11.2 – Desde fuera de la máquina virtual:

Administrador: Windows PowerShell

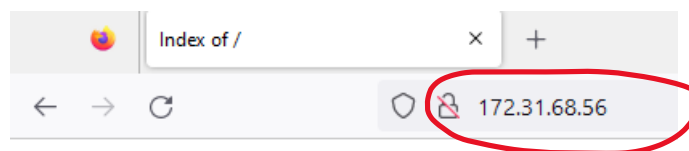
```
PS C:\Users\luivelmor\Desktop> multipass exec gorillaserver -- sudo apt install apache2
```

12 - IP del adaptador de red de la instancia multipass

Todas las máquinas creadas en multipass tienen conectividad entre sí y también con el sistema host (en este caso Windows).


```
PS C:\Users\luivelmor\Desktop> multipass info gorillaserver
Name:          gorillaserver
State:         Running
IPv4:          172.31.68.56
Release:       Ubuntu 22.04.1 LTS
Image hash:    3100a27357a0 (Ubuntu 22.04 LTS)
Load:          0.00 0.00 0.00
Disk usage:    1.4G out of 9.5G
Memory usage:  209.2M out of 1.9G
Mounts:        C:/Users/luivelmor/Desktop/shared_with_multipass => /var/www/html
                UID map: -2:default
                GID map: -2:default
PS C:\Users\luivelmor\Desktop>
```

Mediante dicha IP podemos acceder a un navegador desde el sistema anfitrión. Esto es similar al adaptador “solo-anfitrión” de VirtualBox.



Index of /

<u>Name</u>	<u>Last modified</u>	<u>Size</u>	<u>Description</u>
-------------	----------------------	-------------	--------------------

 asdasd.txt	2023-01-16 07:56	0	
--	------------------	---	--

Apache/2.4.52 (Ubuntu) Server at 172.31.68.56 Port 80

13 - Configuración de las carpetas del servidor de gorilla

Por defecto, el DocumentRoot de apache se ubica en “/var/www/html”

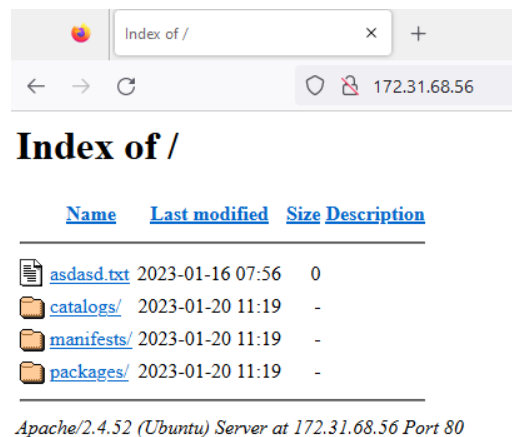
Para gorilla necesitamos la siguiente estructura de carpetas:

- manifests
- catalogs
- packages

El fichero index.html no es necesario. En este caso lo hemos renombrado, pero igualmente podríamos haberlo eliminado.

```
ubuntu@gorillaserver: /var/www/html
ubuntu@gorillaserver:/var/www/html$ cd /var/www/html/
ubuntu@gorillaserver:/var/www/html$ ls
asdasd.txt
ubuntu@gorillaserver:/var/www/html$ sudo mkdir manifests catalogs packages
ubuntu@gorillaserver:/var/www/html$ ls -l
total 0
-rw-rw-rw- 1 ubuntu ubuntu 0 Jan 16 07:56 asdasd.txt
drwxrwxrwx 1 ubuntu ubuntu 0 Jan 20 11:19 catalogs
drwxrwxrwx 1 ubuntu ubuntu 0 Jan 20 11:19 manifests
drwxrwxrwx 1 ubuntu ubuntu 0 Jan 20 11:19 packages
ubuntu@gorillaserver:/var/www/html$
```

Actualizamos el navegador en nuestro sistema anfitrión y ya tenemos la estructura que gorilla necesita.

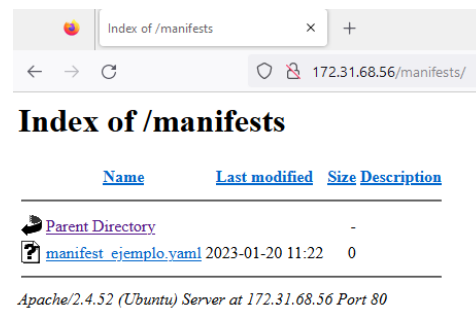
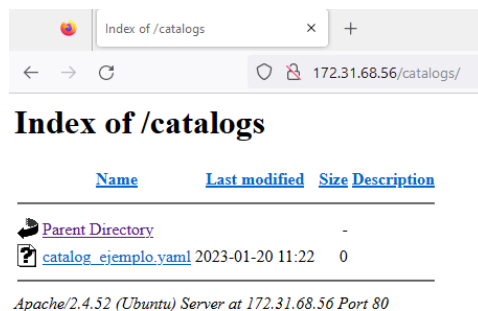


14 - Creación de un manifest y catalog inicial

Crearemos dos ficheros sin contenido (de momento):

- “manifests/manifest_ejemplo.yaml”
- “catalogs/catalog_ejemplo.yaml”

```
ubuntu@gorillaserver: /var/www/html$ sudo touch manifests/manifest_ejemplo.yaml
ubuntu@gorillaserver: /var/www/html$ sudo touch catalogs/catalog_ejemplo.yaml
ubuntu@gorillaserver: /var/www/html$
```

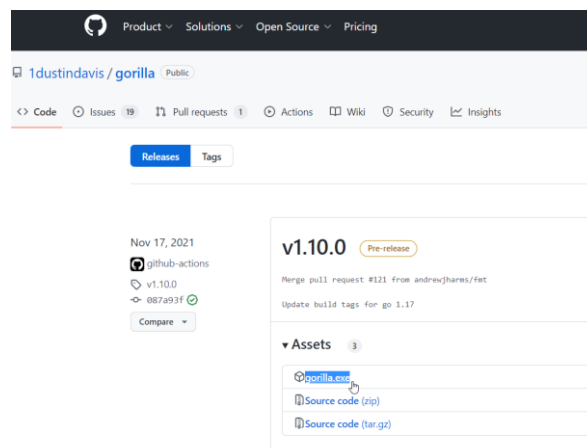


15 - Configuramos un cliente de gorilla

Tenemos que hacer únicamente dos tareas en el cliente:

- 1- Vamos al web del proyecto de gorilla y dentro de ella buscamos las “releases”
 - o Web del proyecto: [GitHub - 1dustindavis/gorilla: Munki-like Application Management for Windows](#)
 - o Releases: [Releases · 1dustindavis/gorilla \(github.com\)](#)

Descargamos el fichero gorilla.exe y lo ubicamos donde queramos. En este caso lo colocaremos dentro de “C:\gorilla\gorilla.exe”



- 2- Creamos un archivo de configuración “config.yaml” en la ubicación “C:/ProgramData/gorilla/config.yaml”, que contendrá la información necesaria para que el fichero “gorilla.exe” sepa dónde está el servidor y qué manifest/catalog le corresponde a este cliente en concreto.

```
config.yaml
url: http://172.31.68.56/
manifest: manifest_ejemplo.yaml
app_data_path: C:/gorilla/cache
catalogs:
  - catalog_ejemplo
```

16 - Prueba de ejecución en el cliente

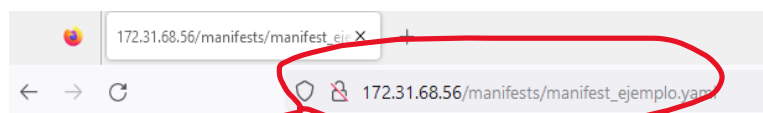
Abrimos una instancia de PowerShell en el sistema anfitrión (importante esto, nos hemos salido de la instancia de multipass del servidor de gorilla, ahora estamos trabajando en el cliente) y ejecutamos “gorilla -d -v”, donde:

-d = debug

-v = verbose

```
Administrador: Windows PowerShell
PS C:\Users\luivelmor\Desktop> gorilla -d -v
Retrieving manifest: manifest_ejemplo
Manifest Url: http://172.31.68.56/manifests/manifest_ejemplo.yaml
Retrieving catalog: [catalog_ejemplo]
Catalog Url: http://172.31.68.56/catalogs/catalog_ejemplo.yaml
Processing manifest...
Processing managed installs...
Processing managed uninstalls...
Processing managed updates...
Saving GorillaReport.json...
Cleaning up the cache...
Cleaning empty directory: packages
Done!
PS C:\Users\luivelmor\Desktop>
```

Como el manifest está vacío, no existe ninguna tarea a realizar.



17 - Conclusiones

Podemos concluir que el equipo cliente que hemos configurado (que se corresponde con el sistema anfitrión) es IGUAL a “manifest_ejemplo”. Por tanto, todo lo que indiquemos en ese manifest serán las tareas que realice este equipo cuando ejecutemos “gorilla.exe”.

Hay una correlación directa entre lo que contiene un manifest y lo que queremos que haga un equipo en función de la configuración del fichero “config.yaml”