

---

# **PRACTICA 3: INSTALACIÓN DE**

# **ZABBIX Y ANSIBLE**

---

Juan Manuel Rodríguez Gómez

Doble Grado en Ingeniería Informática y Matemáticas

Ingeniería de Servidores (Grupo 3)

Curso 2021 – 2022



**UNIVERSIDAD  
DE GRANADA**

## Índice

<b>0. Aclaraciones y Acciones Previas .....</b>	<b>1</b>
<b>1. Ejercicio 1 .....</b>	<b>3</b>
<b>1.0. Enunciado del Ejercicio .....</b>	<b>3</b>
<b>1.1. Primer Paso: Instalación de Zabbix en Ubuntu Server .....</b>	<b>3</b>
<b>1.2. Segundo Paso: Configuración de una BBDD para Utilizarla con Zabbix.....</b>	<b>7</b>
<b>1.3. Tercer Paso: Configuración del Frontend de Zabbix para Monitorización .....</b>	<b>11</b>
<b>1.4. Cuarto Paso: Instalación y Configuración del Zabbix-Agent en CentOS .....</b>	<b>17</b>
<b>1.5. Quinto Paso: Inclusión de los Zabbix-Agent en el Frontend .....</b>	<b>21</b>
<b>1.6. Sexto Paso: Comprobación de la Monitorización en el Frontend .....</b>	<b>27</b>
<b>2. Ejercicio 2 .....</b>	<b>33</b>
<b>2.0. Enunciado del Ejercicio .....</b>	<b>33</b>
<b>2.1. Primer Paso: Creación del Script, del Timer y del Service .....</b>	<b>33</b>
<b>2.2. Segundo Paso: Consultar Log del Servicio de Monitorización del RAID .....</b>	<b>35</b>
<b>2.3. Tercer Paso: Instalación y Configuración de Ansible .....</b>	<b>39</b>
<b>2.4. Cuarto Paso: Comprobación Simultánea de RAID en Ubuntu Server y CentOS mediante Ansible.....</b>	<b>43</b>
<b>2.5. Playbooks de Ansible .....</b>	<b>45</b>
<b>3. Referencias .....</b>	<b>50</b>

## **0. Aclaraciones y Acciones Previas**

- Partiremos de la **instalación de CentOS y Ubuntu Server** que realizamos en la práctica 1. Además, las máquinas virtuales están ya conectadas entre sí. La IP de Ubuntu es 192.168.56.105 y la de CentOS es 192.168.56.110.
- Tenemos que tener instalado y configurado el **servicio sshd** tanto en Ubuntu Server como en CentOS. En mi caso, he partido de lo que realizamos en la práctica 2 (luego, recordemos que tenemos el puerto 22022 para SSH en vez del 22, el cual es el puerto por defecto para SSH). En CentOS ya viene instalado por defecto sshd. Para instalarlo en Ubuntu:
  - *sudo apt install openssh-server* (instalamos el servicio)
  - *sudo systemctl status sshd* (comprobar que se ha iniciado el servicio)
  - *sudo systemctl enable sshd --now* (para iniciar el servicio al levantar el sistema y hacer que se active ya)

Una vez hecho esto podemos editar el servicio con el comando *sudo nano /etc/ssh/sshd\_conf* (podremos cambiar el puerto del servicio, evitar que se conecte por ssh un usuario root, etc.). Si modificamos algo, recordar que hay que lanzar el comando *sudo systemctl restart sshd* para reiniciar el servicio y que se apliquen los cambios realizados.

- Tenemos que tener instalada la **pila LAMP en Ubuntu Server**. Esto ya lo hicimos también en la práctica 2 mediante la herramienta *tasksel*. Sin embargo, esta puede generar futuros problemas de forma que no nos deje arrancar el sistema correctamente (nos lleva a *initramfs*). Por ello, vamos a instalar la pila LAMP en Ubuntu Server de forma manual con los siguientes comandos:
  - *sudo apt upgrade* (por si hace falta)
  - *sudo apt-get install apache2*
  - *sudo systemctl enable apache2 --now*
  - *sudo systemctl status apache2*
  - *sudo apt-get install mariadb-server mariadb-client* (elegimos instalar MariaDB en vez de MySQL ya que este último puede darnos error de versiones a la hora de instalarlo)
  - *sudo systemctl enable mariadb --now*
  - *sudo mysql\_secure\_installation* (script en el que le ponemos contraseña al usuario root de MariaDB, mejorando así la seguridad del servicio)
  - *sudo apt-get install php*
  - *php -a* (Si se ha instalado todo correctamente, se nos abrirá la terminal de PHP. Para salir, escribimos *exit*)

- También tendremos que tener instalado en CentOS el servicio HTTP, lo cual ya hicimos en la práctica 2. Si no lo tuviéramos ejecutamos los siguientes comandos:

- *sudo yum install httpd*
- *sudo systemctl enable httpd --now*
- *sudo systemctl status httpd*
- *sudo firewall-cmd --add-port=80/tcp --permanent* (abrir el puerto permanentemente para así evitar errores debido al cortafuegos)
- *sudo firewall-cmd --add-port=80/tcp* (abrir el puerto en la sesión actual)
- *sudo firewall-cmd --list-ports* (listar los puertos abiertos en la sesión actual)

# **1. Ejercicio 1**

## **1.0. Enunciado del Ejercicio**

*Realice una instalación de Zabbix 5.0 en su servidor con Ubuntu Server20.04 y configure para que se monitorice a él mismo y para que monitorice a la máquina con CentOS. Puede configurar varios parámetros para monitorizar, uso de CPU, memoria, etc. pero debe configurar de manera obligatoria la monitorización de los servicios SSH y HTTP. Documente el proceso de instalación y configuración indicando las referencias que ha utilizado, así como los problemas que ha encontrado. Para ello puede usar cualquier tipo de formato de documento (respetando claridad y corrección) y procure que en las capturas aparezca su nombre de usuario (en el prompt p.ej.). El archivo debe estar subido a SWAD (zona mis trabajos) antes del examen de esta práctica.*

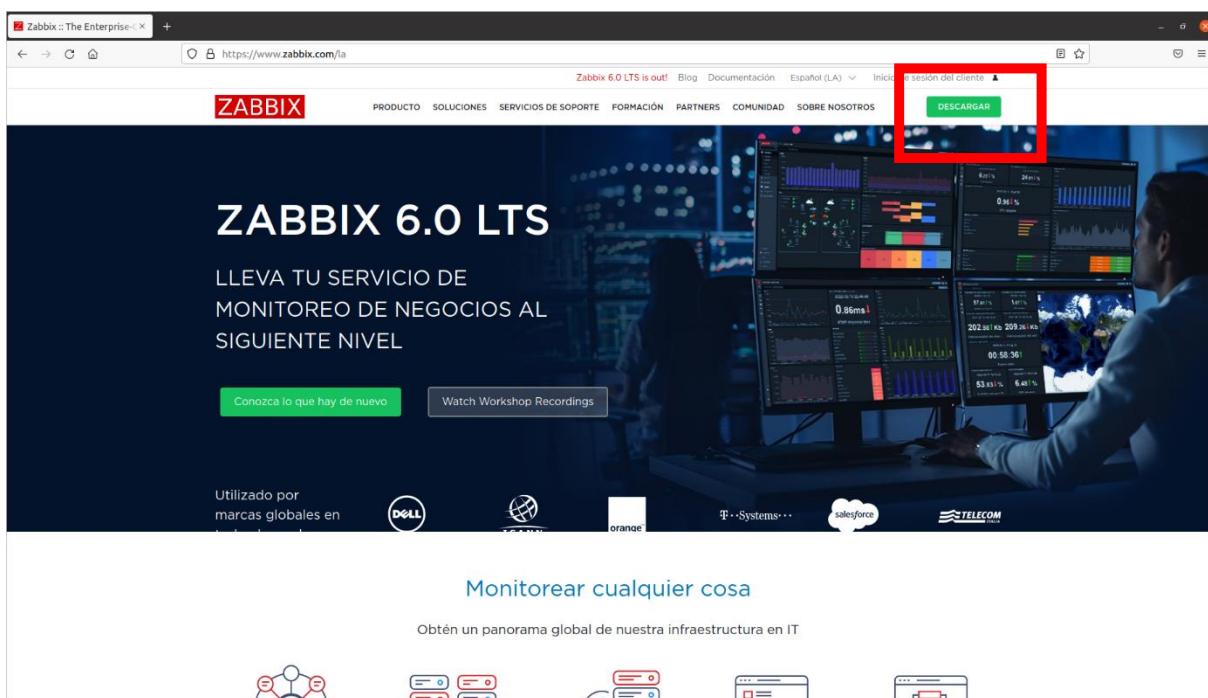
## **1.1. Primer Paso: Instalación de Zabbix en Ubuntu Server**

Zabbix es un programa que nos permite monitorizar distintos elementos (routers, VPNs...), llamarlos y preguntarles información sobre su estado (temperatura, número de instancia de Apache, número de usuarios conectados a un determinado servicio...)

Para instalarlo, comenzamos dirigiéndonos a la página web de Zabbix:

<https://www.zabbix.com/>

Una vez ahí, le damos a Descargar, como se indica en la siguiente imagen.



Una vez pinchemos, nos redirigirá a la siguiente página:

The screenshot shows a web browser window with the URL [https://www.zabbix.com/la/download?zabbix=5.0&os\\_distribution=ubuntu&os\\_version=20.04\\_focal&db=mysql&ws=http](https://www.zabbix.com/la/download?zabbix=5.0&os_distribution=ubuntu&os_version=20.04_focal&db=mysql&ws=http). The page title is "Descargar Zabbix 5.0 LTS". At the top, there's a navigation bar with links like PRODUCTO, SOLUCIONES, SERVICIOS DE SOPORTE, FORMACIÓN, PARTNERS, COMUNIDAD, SOBRE NOSOTROS, and a green "DESCARGAR" button. Below the navigation, there's a large "ZABBIX" logo. Step 1, titled "Elige tu plataforma", is displayed. It shows a grid of options for selecting Zabbix version, distribution, system version, database, and web server. The selected options are highlighted in blue: "5.0 LTS" for Zabbix version, "Ubuntu" for distribution, "20.04 (Focal)" for system version, "MySQL" for database, and "Apache" for web server. A note at the bottom says "Notas de la versión 5.0".

2 Instala y configura el servidor Zabbix para tu plataforma

a. Instalar el repositorio de Zabbix

documentación

```
# wget https://repo.zabbix.com/zabbix/5.0/ubuntu/pool/main/z/zabbix-release/zabbix-release_5.0-1+focal_all.deb  
# dpkg -i zabbix-release_5.0-1+focal_all.deb  
# apt update
```

Como se puede observar, hay diferentes opciones dependiendo de la plataforma en la que queramos instalar Zabbix. Como nosotros vamos a instalarlo en Ubuntu Server, elegimos la configuración que se observa justo en la imagen anterior.

La propia página nos indicará las órdenes que tenemos que ejecutar paso a paso en Ubuntu Server.

The screenshot shows the same web browser window as before, but now it's on step 2. Step 2, titled "Instala y configura el servidor Zabbix para tu plataforma", is displayed. It has three sub-steps: a. Instalar el repositorio de Zabbix, b. Instala el servidor, la interfaz y el agente de Zabbix, and c. Crear base de datos inicial. Each sub-step has a code block with "documentación" link. Sub-step a. shows the command: # wget https://repo.zabbix.com/zabbix/5.0/ubuntu/pool/main/z/zabbix-release/zabbix-release\_5.0-1+focal\_all.deb # dpkg -i zabbix-release\_5.0-1+focal\_all.deb # apt update. Sub-step b. shows the command: # apt install zabbix-server-mysql zabbix-frontend-php zabbix-apache-conf zabbix-agent. Sub-step c. shows the command: # mysql -uroot -p password; create database zabbix character set utf8 collate utf8\_bin; create user zabbix@localhost identified by 'password'; grant all privileges on zabbix.\* to zabbix@localhost; mysql> quit;. A note at the bottom says "En el servidor Zabbix, importe el esquema y los datos iniciales. Se le pedirá que ingrese la contraseña recién creada." and a command: # zcat /usr/share/doc/zabbix-server-mysql\*/create.sql.gz | mysql -uzabbix -p zabbix.

Para facilitar la instalación, como no podemos copiar y pegar desde un navegador de nuestra máquina anfitrión a una máquina virtual, vamos a conectarnos por SSH desde nuestra máquina anfitrión (en la que usamos Ubuntu 20) hacia Ubuntu Server.

Previamente, tenemos que permitir la conexión por SSH mediante la autenticación de contraseña. Para ello, hacemos `sudo nano /etc/ssh/sshd_config` y nos aseguramos de que tengamos `PasswordAuthentication yes` (en caso de tener que cambiarlo, recordar reiniciar el servicio con `sudo systemctl restart sshd` tras salir del archivo).

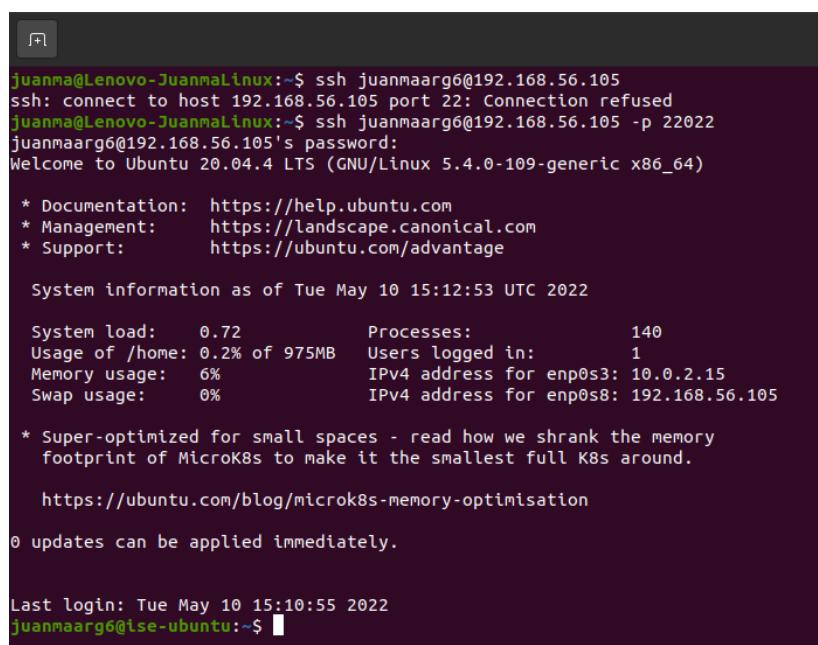
```
#AuthorizedKeysCommand none
#AuthorizedKeysCommandUser nobody

# For this to work you will also need host keys in /etc/ssh/ssh_known_hosts
#HostbasedAuthentication no
# Change to yes if you don't trust ~/.ssh/known_hosts for
# HostbasedAuthentication
#IgnoreUserKnownHosts no
# Don't read the user's ~/.rhosts and ~/.shosts files
#IgnoreRhosts yes

# To disable tunneled clear text passwords, change to no here!
PasswordAuthentication yes ←
#PermitEmptyPasswords no

# Change to yes to enable challenge-response passwords (beware issues with
# some PAM modules and threads)
ChallengeResponseAuthentication no
```

Por tanto, para conectarnos a Ubuntu Server por SSH ejecutamos el comando `ssh juanmaarg6@192.168.56.105 -p 22022` (recordemos que cambiamos el puerto 22, el cual es el puerto por defecto de SSH, por el puerto 22022).



The screenshot shows a terminal window with the following session:

```
juanma@Lenovo-JuanmaLinux:~$ ssh juanmaarg6@192.168.56.105
ssh: connect to host 192.168.56.105 port 22: Connection refused
juanma@Lenovo-JuanmaLinux:~$ ssh juanmaarg6@192.168.56.105 -p 22022
juanmaarg6@192.168.56.105's password:
Welcome to Ubuntu 20.04.4 LTS (GNU/Linux 5.4.0-109-generic x86_64)

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

 System information as of Tue May 10 15:12:53 UTC 2022

 System load:  0.72           Processes:          140
 Usage of /home: 0.2% of 975MB Users logged in:      1
 Memory usage:   6%
 Swap usage:     0%           IPv4 address for enp0s3: 10.0.2.15
                           IPv4 address for enp0s8: 192.168.56.105

 * Super-optimized for small spaces - read how we shrank the memory
   footprint of MicroK8s to make it the smallest full K8s around.

   https://ubuntu.com/blog/microk8s-memory-optimisation

 0 updates can be applied immediately.

 Last login: Tue May 10 15:10:55 2022
 juanmaarg6@ise-ubuntu:~$ █
```

Ahora procedemos a la instalación de Zabbix en Ubuntu Server simplemente copiando las órdenes, las cuales están en la página web, en nuestra terminal conectada por SSH a Ubuntu Server.

Comenzamos con la **descarga e instalación del repositorio de Zabbix** (es necesario cambiarse a usuario root con *sudo su*).

a. Instalar el repositorio de Zabbix

[documentación](#)

```
# wget https://repo.zabbix.com/zabbix/5.0/ubuntu/pool/main/z/zabbix-release/zabbix-release_5.0-1+focal_all.deb  
# dpkg -i zabbix-release_5.0-1+focal_all.deb  
# apt update
```

```
Last login: Tue May 10 15:10:55 2022  
juanmaarg6@ise-ubuntu:~$ sudo su  
[sudo] password for juanmaargo:  
root@ise-ubuntu:/home/juanmaarg6# wget https://repo.zabbix.com/zabbix/5.0/ubuntu/pool/main/z/zabbix-release/zabbix-release_5.0-1+focal_all.deb  
--2022-05-10 15:15:19-- https://repo.zabbix.com/zabbix/5.0/ubuntu/pool/main/z/zabbix-release/zabbix-release_5.0-1+focal_all.deb  
Resolving repo.zabbix.com (repo.zabbix.com)... 178.128.6.101, 2604:a880:2:d0::2062:d001  
Connecting to repo.zabbix.com (repo.zabbix.com)|178.128.6.101|:443... connected.  
HTTP request sent, awaiting response... 200 OK  
Length: 4244 (4.1K) [application/octet-stream]  
Saving to: 'zabbix-release_5.0-1+focal_all.deb'  
  
zabbix-release_5.0- 100%[=====>] 4.14K ---KB/s in 0s  
2022-05-10 15:15:20 (11.9 MB/s) - 'zabbix-release_5.0-1+focal_all.deb' saved [4244/4244]  
  
root@ise-ubuntu:/home/juanmaarg6# dpkg -i zabbix-release_5.0-1+focal_all.deb  
Selecting previously unselected package zabbix-release.  
(Reading database ... 73728 files and directories currently installed.)  
Preparing to unpack zabbix-release_5.0-1+focal_all.deb ...  
Unpacking zabbix-release (1:5.0-1+focal) ...  
Setting up zabbix-release (1:5.0-1+focal) ...  
root@ise-ubuntu:/home/juanmaarg6# apt update  
Hit:1 http://es.archive.ubuntu.com/ubuntu focal InRelease  
Get:2 http://es.archive.ubuntu.com/ubuntu focal-updates InRelease [114 kB]  
Get:3 http://repo.zabbix.com/zabbix/5.0/ubuntu focal InRelease [4958 B]  
Get:4 http://es.archive.ubuntu.com/ubuntu focal-backports InRelease [108 kB]  
Get:5 http://es.archive.ubuntu.com/ubuntu focal-security InRelease [114 kB]  
Get:6 http://repo.zabbix.com/zabbix/5.0/ubuntu focal/main Sources [1203 B]  
Get:7 http://es.archive.ubuntu.com/ubuntu focal-updates/main amd64 Packages [1778 kB]  
Get:8 http://repo.zabbix.com/zabbix/5.0/ubuntu focal/main amd64 Packages [4785 B]  
Get:9 http://es.archive.ubuntu.com/ubuntu focal-updates/main Translation-en [329 kB]  
Get:10 http://es.archive.ubuntu.com/ubuntu focal-updates/main amd64 c-n-f Metadata [15.0 kB]  
Get:11 http://es.archive.ubuntu.com/ubuntu focal-updates/restricted amd64 Packages [971 kB]  
Get:12 http://es.archive.ubuntu.com/ubuntu focal-updates/restricted Translation-en [138 kB]  
Fetched 3579 kB in 3s (1287 kB/s)  
Reading package lists... Done  
Building dependency tree  
Reading state information... Done  
3 packages can be upgraded. Run 'apt list --upgradable' to see them.  
root@ise-ubuntu:/home/juanmaarg6#
```

Ahora continuamos con la **instalación del servicio, la interfaz y el agente de Zabbix**.

b. Instala el servidor, la interfaz y el agente de Zabbix

```
# apt install zabbix-server-mysql zabbix-frontend-php zabbix-apache-conf zabbix-agent
```

```

root@ise-ubuntu:/home/juanmaarg#
Setting up libensors-config (1:3.6.0-0ubuntu1.1) ...
Setting up libjbgb0:amd64 (2.1-3.1build1) ...
Setting up liblwei0:amd64 (0.6.1-2ubuntu0.20.04.1) ...
Setting up libtun0:amd64 (2.0.2-1) ...
Setting up libubccl1:amd64 (2.3.6-0.1build1) ...
Setting up libsensor5:amd64 (1:3.6.0-0ubuntu1.1) ...
Setting up libjpeg-turbo0:amd64 (2.0.3-0ubuntu1.20.04.1) ...
Setting up fonts-dejavu-extra (2.37-1) ...
Setting up php7.4-ldap (7.4.3-4ubuntu2.10) ...

Creating config file /etc/php/7.4/mods-available/ldap.ini with new version
Setting up zabbix-agent (1:5.0.23-1+focal) ...
Created symlink /etc/systemd/system/multi-user.target.wants/zabbix-agent.service → /lib/systemd/system/zabbix-agent.service.
invoke-rc.d: policy-rc.d denied execution of start.
Setting up libopenipm0 (2.0.27-0ubuntu2) ...
Setting up fmpeg0:amd64 (2.8.4-0ubuntu0.20.04.1) ...
Setting up libonig5:amd64 (6.9.4-1) ...
Setting up libjpeg8:amd64 (8c-2ubuntu0) ...
Setting up php-xml (2:7.4+75) ...
Setting up fonts-dejavu (2.37-1) ...
Setting up libsmp35:amd64 (5.8-dfsg-2ubuntu2.3) ...
Setting up zabbix-frontend (1:5.0.23-1+focal) ...
Setting up libfontconfig0 (2.17.4+75) ...
Setting up fontconfig-config (2.13.1-2ubuntu3) ...
Setting up php7.4-mbstring (7.4.3-4ubuntu2.10) ...

Creating config file /etc/php/7.4/mods-available/mbstring.ini with new version
Setting up snmpd (5.8-dfsg-2ubuntu0.3) ...
adduser: warning: user 'snmp' does not belong to the group 'snmp' does not belong to the user you are currently creating.
invoke-rc.d: policy-rc.d denied execution of start.
Created symlink /etc/systemd/system/multi-user.target.wants/snmpd.service → /lib/systemd/system/snmpd.service.
/usr/sbin/policy-rc.d returned 101, not running 'start snmpd.service'
Setting up php-mbstring (2:7.4+75) ...
Setting up php-xml (2:7.4+75) ...
Setting up libcurl4-openssl-dev (7.61.1-0ubuntu11.7-2ubuntu0.20.04.2) ...
Setting up libfontconfig1:amd64 (2.13.1-2ubuntu3) ...
Setting up libgd3:amd64 (2.2.5-5.2ubuntu2.1) ...
Setting up php7.4-gd (7.4.3-4ubuntu2.10) ...

Creating config file /etc/php/7.4/mods-available/gd.ini with new version
Setting up zabbix-frontend-app (1:5.0.23-1+focal) ...
update-alternatives: using /usr/share/fonts/truetype/dejavu/DejaVuSans.ttf to provide /usr/share/zabbix/assets/fonts/graphfont.ttf (zabbix-frontend-font) in auto mode
Setting up php-gd (2:7.4+75) ...
Setting up zabbix-apache-conf (1:5.0.23-1+focal) ...
Enabling conf zabbix.
To activate the new configuration, you need to run:
  sudo a2enconf zabbix
Processing triggers for libc-bin (2.31-0ubuntu9.7) ...
Processing triggers for systemd (245.4-4ubuntu3.10) ...
Processing triggers for man-db (2.9.1-1) ...
Processing triggers for libapache2-mod-php7.4 (7.4.3-4ubuntu2.10) ...
Processing triggers for php7.4-cli (7.4.3-4ubuntu2.10) ...
Setting up mysql (8.0.22-0ubuntu0.20.04.1) ...
root@ise-ubuntu:/home/juanmaarg# 

```

## 1.2. Segundo Paso: Configuración de una BBDD para Utilizarla con Zabbix

### c. Crear base de datos inicial

Make sure you have database server up and running.

[documentación](#)

Ejecuta lo siguiente en el host de base de datos.

```
# mysql -uroot -p
password
mysql> create database zabbix character set utf8 collate utf8_bin;
mysql> create user zabbix@localhost identified by 'password';
mysql> grant all privileges on zabbix.* to zabbix@localhost;
mysql> quit;
```

En el servidor Zabbix, importe el esquema y los datos iniciales. Se le pedirá que ingrese la contraseña recién creada.

```
# zcat /usr/share/doc/zabbix-server-mysql*/create.sql.gz | mysql -uzabbix -p zabbix
```

Cuando finalice la instalación de todos los paquetes de Zabbix, vamos a seguir lo que dice la página web, de forma que vamos a recargar el servicio Apache con *sudo systemctl restart apache2* y asegurarnos de que está activo con *sudo systemctl status apache2*.

```
juanmaarg6@ise-ubuntu:~$ sudo systemctl restart apache2
juanmaarg6@ise-ubuntu:~$ sudo systemctl status apache2
● apache2.service - The Apache HTTP Server
   Loaded: loaded (/lib/systemd/system/apache2.service; enabled; vendor preset: enabled)
     Active: active (running) since Tue 2022-05-10 15:20:37 UTC; 18s ago
       Docs: https://httpd.apache.org/docs/2.4/
    Process: 5491 ExecStart=/usr/sbin/apachectl start (code=exited, status=0/SUCCESS)
   Main PID: 5513 (apache2)
      Tasks: 6 (limit: 5786)
     Memory: 15.9M
        CPU: 0.000 CPU(s) since start
       CGroup: /system.slice/apache2.service
               └─5513 /usr/sbin/apache2 -k start
                  ├─5514 /usr/sbin/apache2 -k start
                  ├─5515 /usr/sbin/apache2 -k start
                  ├─5516 /usr/sbin/apache2 -k start
                  ├─5517 /usr/sbin/apache2 -k start
                  └─5518 /usr/sbin/apache2 -k start

May 10 15:20:37 ise-ubuntu systemd[1]: apache2.service: Succeeded.
May 10 15:20:37 ise-ubuntu systemd[1]: Stopped The Apache HTTP Server.
May 10 15:20:37 ise-ubuntu systemd[1]: Starting The Apache HTTP Server...
May 10 15:20:37 ise-ubuntu apachectl[5505]: AH00558: apache2: Could not reliably determine the server's fully qualified domain name, using ise-ubuntu. (1)
May 10 15:20:37 ise-ubuntu systemd[1]: Started The Apache HTTP Server.
lines 1-21/21 (END)
```

Ya estamos listos para comenzar con la configuración de la base de datos. Comenzamos comprobando que el servicio de MySQL está activo usando el comando *sudo systemctl status mysql*.

```
juanmaarg6@ise-ubuntu:~$ sudo systemctl status mysql
● mariadb.service - MariaDB 10.3.34 database server
   Loaded: loaded (/lib/systemd/system/mariadb.service; enabled; vendor preset: enabled)
     Active: active (running) since Tue 2022-05-10 15:10:52 UTC; 10min ago
       Docs: man:mysqld(8)
              https://mariadb.com/kb/en/library/systemd/
    Main PID: 1026 (mysqld)
      Status: "Taking your SQL requests now..."
      Tasks: 30 (limit: 5786)
     Memory: 94.8M
        CPU: 0.000 CPU(s) since start
       CGroup: /system.slice/mariadb.service
               └─1026 /usr/sbin/mysqld

May 10 15:10:43 ise-ubuntu mysqld[1026]: 2022-05-10 15:10:43 0 [Note] /usr/sbin/mysqld (mysqld 10.3>
May 10 15:10:52 ise-ubuntu systemd[1]: Started MariaDB 10.3.34 database server.
May 10 15:10:52 ise-ubuntu /etc/mysql/debian-start[1247]: Upgrading MySQL tables if necessary.
May 10 15:10:53 ise-ubuntu /etc/mysql/debian-start[1251]: Looking for 'mysql' as: /usr/bin/mysql
May 10 15:10:53 ise-ubuntu /etc/mysql/debian-start[1251]: Looking for 'mysqlcheck' as: /usr/bin/mys>
May 10 15:10:53 ise-ubuntu /etc/mysql/debian-start[1251]: This installation of MariaDB is already u>
May 10 15:10:53 ise-ubuntu /etc/mysql/debian-start[1251]: There is no need to run mysql_upgrade aga>
May 10 15:10:53 ise-ubuntu /etc/mysql/debian-start[1251]: You can use --force if you still want to >
May 10 15:10:53 ise-ubuntu /etc/mysql/debian-start[1286]: Checking for insecure root accounts.
May 10 15:10:53 ise-ubuntu /etc/mysql/debian-start[1294]: Triggering myisam-recover for all MyISAM >
lines 1-22/22 (END)
```

Finalmente, ejecutamos los comandos que aparecen en la página web de Zabbix para **crear la base de datos**.

```
root@ise-ubuntu:/home/juanmaarg6# mysql -uroot -p
Enter password:
Welcome to the MariaDB monitor.  Commands end with ; or \g.
Your MariaDB connection id is 36
Server version: 10.3.34-MariaDB-0ubuntu0.20.04.1 Ubuntu 20.04

Copyright (c) 2000, 2018, Oracle, MariaDB Corporation Ab and others.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

MariaDB [(none)]> create database zabbix character set utf8 collate utf8_bin;
Query OK, 1 row affected (0.001 sec)

MariaDB [(none)]> create user zabbix@localhost identified by 'ise';
Query OK, 0 rows affected (0.215 sec)

MariaDB [(none)]> grant all privileges on zabbix.* to zabbix@localhost;
Query OK, 0 rows affected (0.001 sec)

MariaDB [(none)]> quit;
Bye
root@ise-ubuntu:/home/juanmaarg6#
```

Cabe destacar que instalamos MariaDB en vez de MySQL aunque esto no afecta en nada a los comandos (son los mismos). Por otro lado, si nos fijamos en el comando *create user zabbix@localhost identified by 'ise'* hemos sustituidos '*password*' (así viene en la web) por '*ise*' (contraseña de identificación estándar que hemos escogido en la asignatura).

Una vez creada la base de datos, vamos a **importar el esquema y los datos iniciales** (se nos pedirá la contraseña introducida anteriormente, es decir, *ise*). La sentencia tarda en ejecutarse un rato.

```
juanmaarg6@ise-ubuntu:~$ zcat /usr/share/doc/zabbix-server-mysql*/create.sql.gz | mysql -uzabbix -p zabbix
Enter password:
juanmaarg6@ise-ubuntu:~$
```

Este comando tarda un buen rato en ejecutarse (luego, solo queda tener paciencia y esperar). Ahora solo nos queda **configurar la base de datos creada** para Zabbix.

d. Configure the database for Zabbix server  
Edit file /etc/zabbix/zabbix\_server.conf

```
DBPassword=password
```

Para ello, modificamos el fichero de configuración de Zabbix Server con `sudo nano /etc/zabbix/zabbix_server.conf` y en `DBPassword` ponemos la contraseña que introdujimos anteriormente para la base de datos, es decir, `ise`.

```
### Option: DBPassword
#       Database password.
#       Comment this line if no password is used.
#
# Mandatory: no
# Default:
DBPassword=ise

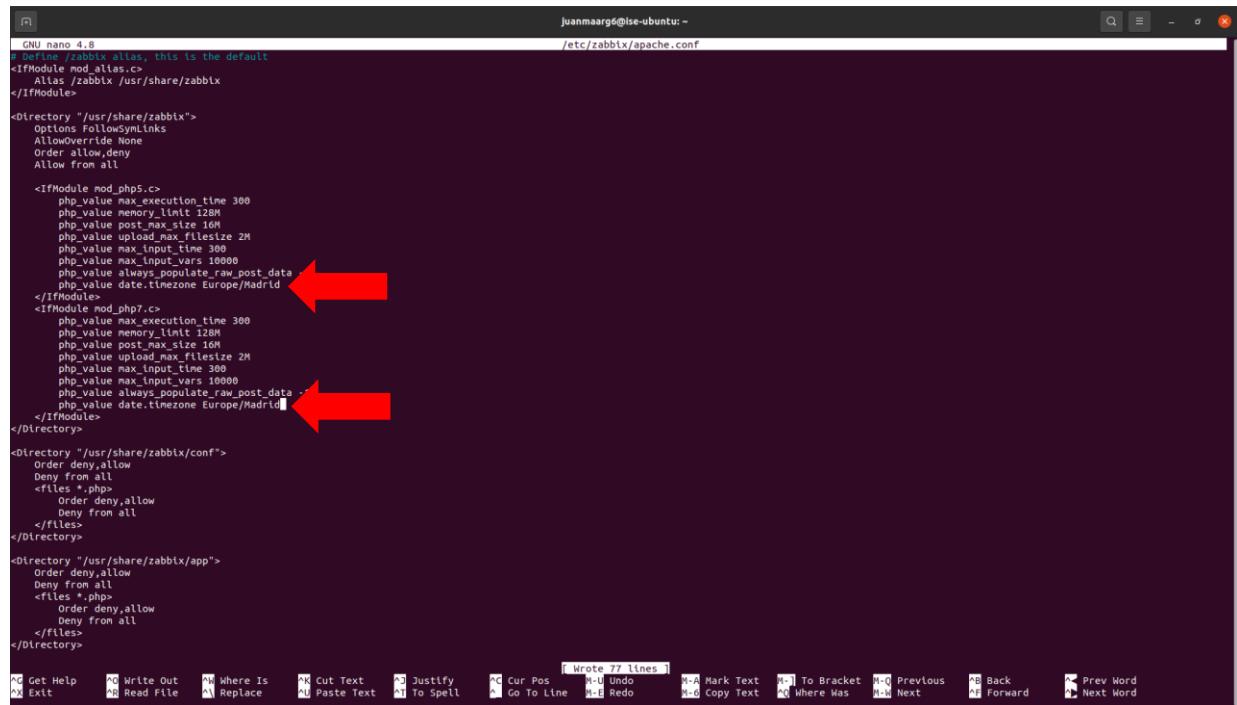
### Option: DBSocket
#       Path to MySQL socket.
#
# Mandatory: no
# Default:
# DBSocket=
```

Ahora, tal y como indica la página web, vamos a modificar la configuración PHP para el frontend de Zabbix, de forma que establecemos la zona horaria en la que nos encontramos (en mi caso, sería *Europe/Madrid*).

#### e. Configure PHP for Zabbix frontend

Edit file `/etc/zabbix/apache.conf`, uncomment and set the right timezone for you.

```
# php_value date.timezone Europe/Riga
```



```
juanma@lse-ubuntu:~ /etc/zabbix/apache.conf
GNU nano 4.8
# Define Zabbix alias, this is the default
<IfModule mod_alias.c>
  Alias /zabbix /usr/share/zabbix
</IfModule>

<Directory "/usr/share/zabbix">
  Options FollowSymlinks
  AllowOverride None
  Order allow,deny
  Allow from all

  <IfModule mod_php5.c>
    php_value max_execution_time 300
    php_value memory_limit 128M
    php_value post_max_size 10M
    php_value upload_max_filesize 2M
    php_value max_input_time 300
    php_value max_input_vars 10000
    php_value always_populate_raw_post_data -
    php_value date.timezone Europe/Madrid
  </IfModule>
  <IfModule mod_php7.c>
    php_value max_execution_time 300
    php_value memory_limit 128M
    php_value post_max_size 10M
    php_value upload_max_filesize 2M
    php_value max_input_time 300
    php_value max_input_vars 10000
    php_value always_populate_raw_post_data -
    php_value date.timezone Europe/Madrid
  </IfModule>
</Directory>

<Directory "/usr/share/zabbix/conf">
  Order deny,allow
  Deny from all
  <files *.php>
    Order deny,allow
    Deny from all
  </files>
</Directory>

<Directory "/usr/share/zabbix/app">
  Order deny,allow
  Deny from all
  <files *.php>
    Order deny,allow
    Deny from all
  </files>
</Directory>
```

Para finalizar, restablecemos los servicios de Zabbix con el comando `sudo systemctl restart zabbix-server zabbix-agent apache2` y hacemos que se inicien al arrancar Ubuntu Server con `systemctl enable zabbix-server zabbix-agent apache2`.

#### f. Start Zabbix server and agent processes

Start Zabbix server and agent processes and make it start at system boot.

```
# systemctl restart zabbix-server zabbix-agent apache2  
# systemctl enable zabbix-server zabbix-agent apache2
```

```
root@ise-ubuntu:/home/juanmaarg6# systemctl restart zabbix-server zabbix-agent apache2
root@ise-ubuntu:/home/juanmaarg6# systemctl enable zabbix-server zabbix-agent apache2
Synchronizing state of zabbix-server.service with SysV service script with /lib/systemd/systemd-sysv-install.
Executing: /lib/systemd/systemd-sysv-install enable zabbix-server
Synchronizing state of zabbix-agent.service with SysV service script with /lib/systemd/systemd-sysv-install.
Executing: /lib/systemd/systemd-sysv-install enable zabbix-agent
Synchronizing state of apache2.service with SysV service script with /lib/systemd/systemd-sysv-install.
Executing: /lib/systemd/systemd-sysv-install enable apache2
Created symlink /etc/systemd/system/multi-user.target.wants/zabbix-server.service → /lib/systemd/system/zabbix-server.service.
root@ise-ubuntu:/home/juanmaarg6#
```

### **1.3. Tercer Paso: Configuración del Frontend de Zabbix para Monitorización**

Vamos a indicar el procedimiento a seguir para configurar Zabbix de forma que monitorice los servicios especificados en el enunciado del ejercicio (SSH y HTTP).

Comenzamos comprobando que todos los servicios de Zabbix (`zabbix-server`, `zabbix-agent` y `apache2`) están activos.

```
root@lse-ubuntu:/home/juanmaarg

● zabbix-server.service - Zabbix Server
   Loaded: loaded (/lib/systemd/system/zabbix-server.service; enabled; vendor preset: enabled)
   Active: active (running) since Tue 2022-05-10 21:55:19 UTC; 1min 14s ago
     Main PID: 7781 (zabbix-server)
       Tasks: 1 (limit: 5786)
      Memory: 30.9M
CGroup: /system.slice/zabbix-server.service
└─7781 /usr/sbin/zabbix_server -c /etc/zabbix/zabbix-server.conf

Jul 10 21:55:19 lse-ubuntu zabbix-server[7781]: configuration syncer [synced configuration in 0.054476 sec, idle 60 sec]
Jul 10 21:55:19 lse-ubuntu zabbix-server[7781]: alarm manager [idle 0 ms, spent 0, failed 0 alerts, idle 5.016588 sec during 5.016797 sec]
Jul 10 21:55:19 lse-ubuntu zabbix-server[7795]: alert #1 started
Jul 10 21:55:19 lse-ubuntu zabbix-server[7799]: alert #2 started
Jul 10 21:55:19 lse-ubuntu zabbix-server[7797]: alert #3 started
Jul 10 21:55:19 lse-ubuntu zabbix-server[7799]: preprocessing manager #1 [queued 0, processed 2 values, idle 5.013927 sec during 5.014293 sec]
Jul 10 21:55:19 lse-ubuntu zabbix-server[7800]: preprocessing worker #1 started
Jul 10 21:55:19 lse-ubuntu zabbix-server[7800]: preprocessing worker #2 started
Jul 10 21:55:19 lse-ubuntu zabbix-server[7801]: preprocessing worker #3 started
Jul 10 21:55:19 lse-ubuntu zabbix-server[7802]: lld manager #1 [processed 0 LLD rules, idle 5.006332sec during 5.006537 sec]
Jul 10 21:55:19 lse-ubuntu zabbix-server[7803]: lld worker #1 started
Jul 10 21:55:19 lse-ubuntu zabbix-server[7804]: lld worker #2 started
Jul 10 21:55:19 lse-ubuntu zabbix-server[7805]: housekeeper [startup idle for 30 minutes]
Jul 10 21:55:19 lse-ubuntu zabbix-server[7806]: hme #1 [updated 0 hosts, suppressed 0 events in 0.000243 sec, idle 59 sec]
Jul 10 21:55:19 lse-ubuntu zabbix-server[7807]: http poller #1 [got 0 values in 0.001081 sec, idle 5 sec]
Jul 10 21:55:19 lse-ubuntu zabbix-server[7808]: history syncer [processed 0 values, 0 triggers in 0.000078 sec, syncing history]
Jul 10 21:55:19 lse-ubuntu zabbix-server[7809]: history syncer [processed 0 values, 0 triggers in 0.000078 sec, syncing history]
Jul 10 21:55:19 lse-ubuntu zabbix-server[7810]: history syncer #2 [processed 0 values, 0 triggers in 0.000078 sec, syncing history]
Jul 10 21:55:19 lse-ubuntu zabbix-server[7811]: history syncer #3 [processed 0 values, 0 triggers in 0.000078 sec, syncing history]
Jul 10 21:55:19 lse-ubuntu zabbix-server[7812]: history syncer [processed 0 values, 0 triggers in 0.000078 sec, syncing history]
Jul 10 21:55:19 lse-ubuntu zabbix-server[7813]: history syncer [processed 0 values, 0 triggers in 0.000078 sec, syncing history]
Jul 10 21:55:19 lse-ubuntu zabbix-server[7814]: history syncer [processed 0 values, 0 triggers in 0.000078 sec, syncing history]
Jul 10 21:55:19 lse-ubuntu zabbix-server[7815]: proxy poller #1 [exchanged data with 0 proxies in 0.000012 sec, idle 5 sec]
Jul 10 21:55:19 lse-ubuntu zabbix-server[7816]: self-monitoring [processed data in 0.000038 sec, idle 1 sec]
Jul 10 21:55:19 lse-ubuntu zabbix-server[7817]: task manager [processed 0 task(s) in 0.000293 sec, idle 5 sec]
Jul 10 21:55:19 lse-ubuntu zabbix-server[7818]: trap receiver [processed data in 0.000008 sec, waiting for connection]
Jul 10 21:55:19 lse-ubuntu zabbix-server[7819]: poller #2 [got 0 values in 0.000025 sec, idle 1 sec]
Jul 10 21:55:19 lse-ubuntu zabbix-server[7820]: poller #3 [got 0 values in 0.000045 sec, idle 1 sec]
Jul 10 21:55:19 lse-ubuntu zabbix-server[7821]: poller #5 [got 0 values in 0.000036 sec, idle 1 sec]
Jul 10 21:55:19 lse-ubuntu zabbix-server[7822]: unreachable proxy [idle 5 sec]
Jul 10 21:55:19 lse-ubuntu zabbix-server[7823]: trap receiver [processed data in 0.000008 sec, waiting for connection]
Jul 10 21:55:19 lse-ubuntu zabbix-server[7824]: trapper #2 [processed data in 0.000008 sec, waiting for connection]
Jul 10 21:55:19 lse-ubuntu zabbix-server[7825]: trapper #3 [processed data in 0.000008 sec, waiting for connection]
Jul 10 21:55:19 lse-ubuntu zabbix-server[7826]: trapper #4 [processed data in 0.000008 sec, waiting for connection]
```

```
root@lse-ubuntu:/home/juanmaarg6
● zabbix-agent.service - Zabbix Agent
   Loaded: loaded (/lib/systemd/system/zabbix-agent.service; enabled; vendor preset: enabled)
   Active: active (running) since Tue 2022-05-10 21:55:19 UTC; 1min 42s ago
     Main PID: 7774 (zabbix_agt)
        Tasks: 6 (limit: 5786)
       Memory: 6.1M
      CGroup: /system.slice/zabbix-agent.service
              └─7774 /usr/sbin/zabbix_agt -c /etc/zabbix/zabbix_agentd.conf

May 10 21:55:17 lse-ubuntu systemd[1]: Starting Zabbix Agent...
May 10 21:55:19 lse-ubuntu systemd[1]: Started Zabbix Agent.
root@lse-ubuntu:/home/juanmaarg6#
```

```
root@lse-ubuntu:/home/juanmaarg6
● apache2.service - The Apache HTTP Server
   Loaded: loaded (/lib/systemd/system/apache2.service; enabled; vendor preset: enabled)
   Active: active (running) since Tue 2022-05-10 21:55:18 UTC; 2min 25s ago
     Docs: https://httpd.apache.org/docs/2.4/
   Main PID: 7776 /usr/sbin/apache2
      Tasks: 6 (limit: 5786)
     Memory: 12.0M
    CGroup: /system.slice/apache2.service
            ├─7776 /usr/sbin/apache2 -k start
            ├─7768 /usr/sbin/apache2 -k start
            ├─7769 /usr/sbin/apache2 -k start
            ├─7770 /usr/sbin/apache2 -k start
            ├─7771 /usr/sbin/apache2 -k start
            └─7772 /usr/sbin/apache2 -k start

May 10 21:55:18 lse-ubuntu systemd[1]: Starting The Apache HTTP Server...
May 10 21:55:18 lse-ubuntu apache2[7766]: AH00558: apache2: Could not reliably determine the server's fully qualified domain name, using 127.0.1.1. Set the 'ServerName' directive globally to suppress this message
May 10 21:55:18 lse-ubuntu systemd[1]: Started The Apache HTTP Server.
(Lines 1-18 (END))
```

Ahora configuraremos el frontend de Zabbix. Para acceder a él, ponemos el siguiente enlace en un navegador tal y como nos indica la página web de Zabbix:

<http://192.168.56.105/zabbix>

### g. Configure Zabbix frontend

Connect to your newly installed Zabbix frontend: [http://server\\_ip\\_or\\_name/zabbix](http://server_ip_or_name/zabbix)  
Follow steps described in Zabbix documentation: [Installing frontend](#)

Nos tiene que aparecer la siguiente página:



Le damos a *Next Step*. Ahora se comprueba que cumplimos todos los requisitos necesarios antes de continuar con la instalación.

The screenshot shows the "Check of pre-requisites" step of the Zabbix setup. On the left, there's a sidebar with the same set of links as the previous page. The main area has a red header bar with "ZABBIX". Below it, the title "Check of pre-requisites" is centered. To the right is a table comparing current PHP settings against required values. The table has columns for "Current value", "Required", and "Status" (which is consistently "OK" in this view). The table rows include:

	Current value	Required	Status
PHP version	7.4.3	7.2.0	OK
PHP option "memory_limit"	128M	128M	OK
PHP option "post_max_size"	16M	16M	OK
PHP option "upload_max_filesize"	2M	2M	OK
PHP option "max_execution_time"	300	300	OK
PHP option "max_input_time"	300	300	OK
PHP option "date.timezone"	Europe/Madrid		OK
PHP databases support	MySQL		OK
PHP bcmath	on		OK
PHP mbstring	on		OK

At the bottom right are "Back" and "Next step" buttons. A note at the very bottom says "Licensed under GPL v2".

Observamos que todo está correcto, luego, le damos a *Next Step*. Ahora toca configurar la conexión con la base de datos creada anteriormente, luego, tendremos que poner el mismo usuario y contraseña que habíamos puesto antes para la base de datos.

The screenshot shows the 'Configure DB connection' step of the Zabbix setup wizard. The 'Database type' is set to MySQL. The 'Database host' is localhost, 'Database port' is 0 (indicating default), 'Database name' is zabbix, 'User' is zabbix, and 'Password' is left blank. A note below states: 'Connection will not be encrypted because it uses a socket file (on Unix) or shared memory (Windows)'. Navigation buttons at the bottom are 'Back' and 'Next step'.

Una vez hecho esto, le damos a *Next Step*. Ahora nos saldrá más información acerca del servidor de Zabbix.

The screenshot shows the 'Zabbix server details' step of the setup wizard. It asks for the host name or IP address and port number of the Zabbix server, and an optional name. The 'Host' is localhost, 'Port' is 10051, and 'Name' is juanmaarg6. Navigation buttons at the bottom are 'Back' and 'Next step'.

Le damos a *Next Step*. Finalmente, nos sale un resumen de todos los parámetros anteriores antes de que se realice la instalación.

The screenshot shows the 'Pre-installation summary' step of the Zabbix installation wizard. On the left, a sidebar lists steps: Welcome, Check of pre-requisites, Configure DB connection, Zabbix server details, Pre-installation summary, and Install. The main area displays configuration parameters:

Database type	MySQL
Database server	localhost
Database port	default
Database name	zabbix
Database user	zabbix
Database password	***
Database TLS encryption	false
Zabbix server	localhost
Zabbix server port	10051
Zabbix server name	juanmaarg6

At the bottom right are 'Back' and 'Next step' buttons. A footer note says 'Licensed under [GPL v2](#)'.

Finalmente, tras darle a *Next Step*, nos saldrá lo siguiente indicándonos que el frontend de Zabbix se ha instalado correctamente.

The screenshot shows the 'Install' step of the Zabbix installation wizard. On the left, a sidebar lists steps: Welcome, Check of pre-requisites, Configure DB connection, Zabbix server details, Pre-installation summary, and Install. The main area displays a success message:

**Congratulations! You have successfully installed Zabbix frontend.**

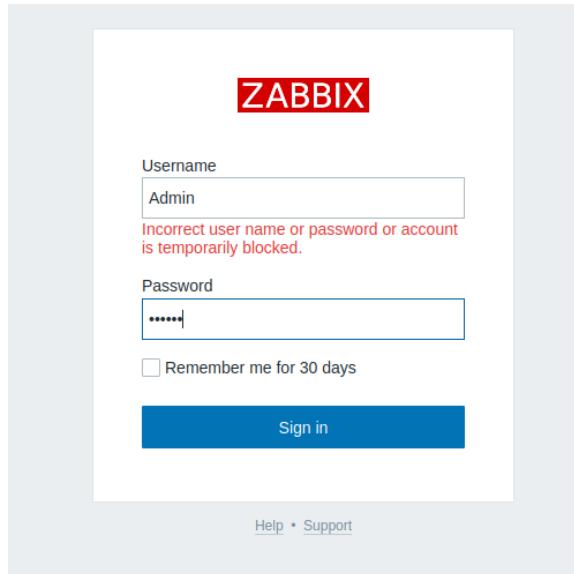
Configuration file "conf/zabbix.conf.php" created.

At the bottom right are 'Back' and 'Finish' buttons. A footer note says 'Licensed under [GPL v2](#)'.

Tras darle a *Finish*, nos aparecerá un inicio de sesión para entrar en el frontend de Zabbix. Las credenciales son las siguientes:

**Username: Admin**

**Password: zabbix**



Tras esto, nos encontraremos en el frontend de Zabbix, la cual se puede ver que está activa.

A screenshot of the Zabbix Global view dashboard. The left sidebar shows navigation links like Monitoring, Problems, Hosts, Overview, Latest data, Screens, Maps, Discovery, Services, Inventory, Reports, Configuration, Administration, Support, Share, Help, User settings, and Sign out. The main area is titled "Global view" and includes sections for "System information" (with a table showing Zabbix server status, host count, templates, items, triggers, and users) and "Problems" (a table showing two recent issues for "Zabbix server"). A large clock icon is in the top right. On the right side, there are sections for "Favourite maps" (empty) and "Favourite graphs" (empty).

## **1.4. Cuarto Paso: Instalación y Configuración del Zabbix-Agent en CentOS**

El ejercicio nos indica que monitoricemos también CentOS aparte de Ubuntu Server. Por tanto, realizaremos también la instalación de Zabbix en este sistema. Pero no instalaremos todos los servicios de Zabbix como hemos hecho en Ubuntu Server. **Solo instalaremos el Agente de Zabbix**, ya que es el único que necesitamos (el servidor y el frontend de Zabbix ya nos lo proporciona Ubuntu Server tras instalárselos).

Al igual que hicimos con Ubuntu Server, para facilitar la instalación, como no podemos copiar y pegar desde un navegador de nuestra máquina anfitrión a una máquina virtual, vamos a conectarnos por SSH desde nuestra máquina anfitrión (en la que usamos Ubuntu 20) hacia CentOS.

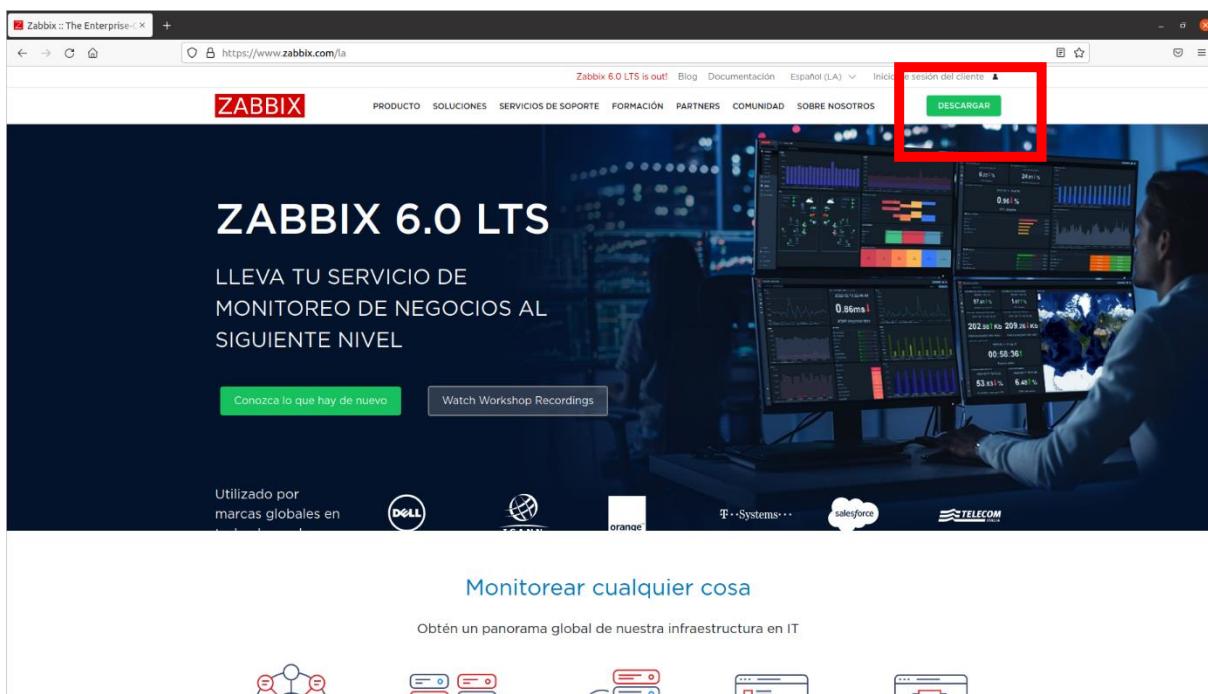
```
juanma@Lenovo-JuanmaLinux:~$ ssh juanmaarg6@192.168.56.110 -p 22022
The authenticity of host '[192.168.56.110]:22022 ([192.168.56.110]:22022)' can't be established.
ECDSA key fingerprint is SHA256:dgM6+HBkzVR8BpDbtF5CK+Gryc95Y4iVJ06bYIs1bLQ.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '[192.168.56.110]:22022' (ECDSA) to the list of known hosts.
juanmaarg6@192.168.56.110's password:
Activate the web console with: systemctl enable --now cockpit.socket

Last login: Tue May 10 18:13:41 2022
[juanmaarg6@localhost ~]$
```

Volvemos a la página web de Zabbix:

<https://www.zabbix.com/>

Una vez ahí, le damos a Descargar, como ya hicimos para Ubuntu Server.



Una vez pinchemos, nos redirigirá a la siguiente página:

1 Elige tu plataforma

VERSIÓN ZABBIX	DISTRIBUCIÓN DE SO	VERSIÓN DEL SISTEMA OPERATIVO	BASE DE DATOS	SERVIDOR WEB
6.0 LTS	Red Hat Enterprise Linux	8 Stream	MySQL	Apache
5.0 LTS	CentOS	7	PostgreSQL	NGINX
4.0 LTS	Rocky Linux	6		
6.2 PRE-RELEASE	Alma Linux			
	Oracle Linux			
	Ubuntu			
	Debian			
	SUSE Linux Enterprise Server			
	Raspberry Pi OS			
	Ubuntu (arm64)			

Notas de la versión 5.0

Ahora vamos a instalar Zabbix ahora en CentOS, luego, elegimos la configuración que se observa justo en la imagen anterior.

Al igual que ocurría con Ubuntu Server, la propia página nos indicará las órdenes que tenemos que ejecutar paso a paso en CentOS.

2 Instala y configura el servidor Zabbix para tu plataforma

a. Instalar el repositorio de Zabbix

```
# rpm -Uvh https://repo.zabbix.com/zabbix/5.0/rhel/8/x86_64/zabbix-release-5.0-1.el8.noarch.rpm  
# dnf clean all
```

b. Instala el servidor, la interfaz y el agente de Zabbix

```
# dnf install zabbix-server-mysql zabbix-web-mysql zabbix-apache-conf zabbix-agent
```

c. Crear base de datos inicial

Make sure you have database server up and running.

Ejecuta lo siguiente en el host de base de datos.

```
# mysql -uroot -p  
password  
mysql> create database zabbix character set utf8 collate utf8_bin;  
mysql> create user zabbix@localhost identified by 'password';  
mysql> grant all privileges on zabbix.* to zabbix@localhost;  
mysql> quit;
```

En el servidor Zabbix, importa el esquema y los datos iniciales. Se le pedirá que ingrese la contraseña recién creada.

```
# zcat /usr/share/doc/zabbix-server-mysql*/create.sql.gz | mysql -uzabbix -p zabbix
```

d. Configurar la base de datos para el servidor Zabbix

Editar archivo /etc/zabbix/zabbix\_server.conf

```
DBPassword=password
```

Procedemos a la instalación de Zabbix en CentOS simplemente copiando las órdenes, las cuales están en la página web, en nuestra terminal de la máquina anfitrión conectada por SSH a CentOS.

Comenzamos con la **descarga e instalación del repositorio de Zabbix** (es necesario cambiarse a usuario root con *sudo su*).

a. Instalar el repositorio de Zabbix

documentación

```
# rpm -Uvh https://repo.zabbix.com/zabbix/5.0/rhel/8/x86_64/zabbix-release-5.0-1.el8.noarch.rpm  
# dnf clean all
```

```
[root@localhost juanmaarg6]# rpm -Uvh https://repo.zabbix.com/zabbix/5.0/rhel/8/x86_64/zabbix-release-5.0-1.el8.noarch.rpm  
Recuperando https://repo.zabbix.com/zabbix/5.0/rhel/8/x86_64/zabbix-release-5.0-1.el8.noarch.rpm  
advertencia:/var/tmp/rpm-tmp.SWkaOX: EncabezadoV4 RSA/SHA512 Signature, ID de clave a14fe591: NOKEY  
Verifying...  
Preparando...  
Actualizando / instalando...  
 1:zabbix-release-5.0-1.el8  #### [100%]  
[root@localhost juanmaarg6]# dnf clean all  
36 archivos eliminados  
[root@localhost juanmaarg6]#
```

Ahora continuamos con la **instalación solo del agente de Zabbix**.

b. Instala el servidor, la interfaz y el agente de Zabbix

```
# dnf install zabbix-server mysql zabbix-web-mysql zabbix-apache-conf zabbix-agent
```

```
[root@localhost juanmaarg6]# dnf install zabbix-agent  
centOS-8 - AppStream  
centOS-8 - Base  
centOS-8 - Extras  
Extra Packages for Enterprise Linux Modular 8 - x86_64  
Extra Packages for Enterprise Linux 8 - x86_64  
Zabbix Official Repository - x86_64  
Zabbix Official Repository non-supported - x86_64  
Dependencias resueltas.  
=====  
Paquete Arquitectura Versión Repositorio Tam.  
=====  
Instalando:  
zabbix-agent x86_64 5.0.23-1.el8 zabbix 492 k  
Resumen de la transacción  
=====  
Instalar 1 Paquete  
Tamaño total de la descarga: 492 k  
Tamaño instalado: 2.1 M  
¿Está de acuerdo [s/N]? s  
Descargando paquetes:  
zabbix-agent-5.0.23-1.el8.x86_64.rpm  
[ 1 ]  
advertisión:/var/cache/dnf/zabbix-b7349ccb4866b08d/packages/zabbix-agent-5.0.23-1.el8.x86_64.rpm: EncabezadoV4 RSA/SHA512 Signature, ID de clave a14fe591: NOKEY  
Importando llave GPG 0xA14FE591:  
ID usuario: "Zabbix LLC <packager@zabbix.com>"  
Huella : A1B4 8F53 52D0 22B0 471D B3D0 082A B56B A14F E591  
Desde : /etc/pki/rpm-gpg/RPM-GPG-KEY-ZABBIX-A14FE591  
¿Está de acuerdo con ello? s  
La llave ha sido importada exitosamente  
Ejecutando verificación de operación  
Verificación de operación exitosa  
Ejecutando prueba de operaciones  
Prueba de operación exitosa.  
Ejecutando operación  
  Preparando :  
    Ejecutando scriptlet: zabbix-agent-5.0.23-1.el8.x86_64  
  Instalando : zabbix-agent-5.0.23-1.el8.x86_64  
  Ejecutando scriptlet: zabbix-agent-5.0.23-1.el8.x86_64  
  Verificando : zabbix-agent-5.0.23-1.el8.x86_64  
Instalado:  
zabbix-agent-5.0.23-1.el8.x86_64  
[listo]  
[root@localhost juanmaarg6]#
```

Comprobamos el estado del agente de Zabbix con *sudo systemctl status zabbix-agent* y vemos que está inactivo. Antes de iniciar el servicio, vamos a configurar algunas cosas del agente modificando el fichero correspondiente: *sudo nano /etc/zabbix/zabbix\_agentd.conf*

En dicho fichero, vamos a especificar la IP del Servidor donde se encuentra el frontend de Zabbix, es decir, tenemos que especificar la IP de Ubuntu Server (recordemos que es la 192.168.56.105).

```

## Option: Server
# List of comma delimited IP addresses, optionally in CIDR notation, or DNS names of Zabbix servers and Zabbix proxies.
# Incoming connections will be accepted only from the hosts listed here.
# If IPv6 support is enabled then '127.0.0.1', '::127.0.0.1', '::ffff:127.0.0.1' are treated equally
# and '::/0' will allow any IPv4 or IPv6 address.
# '0.0.0.0/0' can be used to allow any IPv4 address.
# Example: Server=127.0.0.1,192.168.1.0/24::1,2001:db8::/32,zabbix.example.com

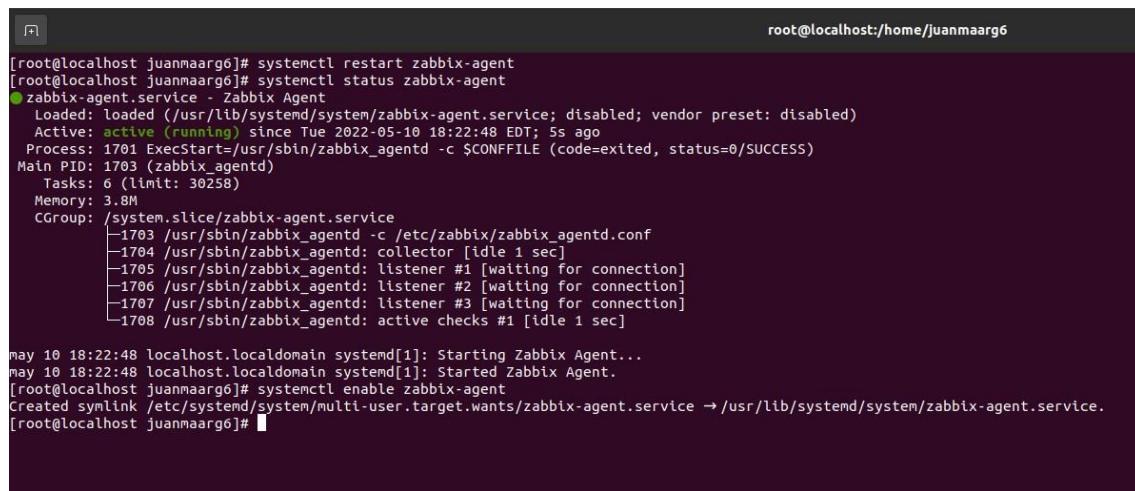
# Mandatory: yes, if StartAgents is not explicitly set to 0
# Default:
# Server=

Server=192.168.56.105

### Option: ListenPort
# Agent will listen on this port for connections from the server.
#

```

Ahora reiniciamos el servicio con `sudo systemctl restart zabbix-agent` y hacemos que se inicie al arrancar el sistema con `sudo systemctl enable zabbix-agent`.



```

root@localhost juanmaarg6]# systemctl restart zabbix-agent
[root@localhost juanmaarg6]# systemctl status zabbix-agent
● zabbix-agent.service - Zabbix Agent
   Loaded: loaded (/usr/lib/systemd/system/zabbix-agent.service; disabled; vendor preset: disabled)
   Active: active (running) since Tue 2022-05-10 18:22:48 EDT; 5s ago
     Process: 1701 ExecStart=/usr/sbin/zabbix_agentd -c $CONFFILE (code=exited, status=0/SUCCESS)
    Main PID: 1703 (zabbix_agentd)
      Tasks: 6 (limit: 30258)
     Memory: 3.8M
        CPU: 0.000 CPU(s) [idle 1 sec]
       CGroup: /system.slice/zabbix-agent.service
               ├─1703 /usr/sbin/zabbix_agentd -c /etc/zabbix/zabbix_agentd.conf
               ├─1704 /usr/sbin/zabbix_agentd: collector [idle 1 sec]
               ├─1705 /usr/sbin/zabbix_agentd: listener #1 [waiting for connection]
               ├─1706 /usr/sbin/zabbix_agentd: listener #2 [waiting for connection]
               ├─1707 /usr/sbin/zabbix_agentd: listener #3 [waiting for connection]
               ├─1708 /usr/sbin/zabbix_agentd: active checks #1 [idle 1 sec]

may 10 18:22:48 localhost.localdomain systemd[1]: Starting Zabbix Agent...
may 10 18:22:48 localhost.localdomain systemd[1]: Started Zabbix Agent.
[root@localhost juanmaarg6]# systemctl enable zabbix-agent
Created symlink /etc/systemd/system/multi-user.target.wants/zabbix-agent.service → /usr/lib/systemd/system/zabbix-agent.service.
[root@localhost juanmaarg6]#

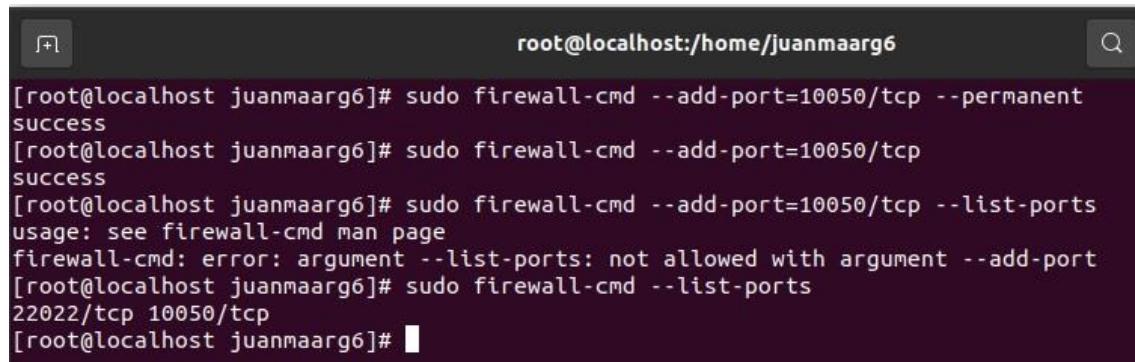
```

Para finalizar, recordemos que hemos puesto un puerto de escucha no convencional para Zabbix (el 10050). Por tanto, tenemos que abrir dicho puerto en el firewall de CentOS. Para ello, ejecutamos los siguientes comandos:

`sudo firewall-cmd --add-port=10050/tcp --permanent` (para abrir el puerto permanentemente)

`sudo firewall-cmd --add-port=10050/tcp` (para abrir el puerto en la sesión actual)

`sudo firewall-cmd --list-ports` (para listar los puertos abiertos en la sesión actual)



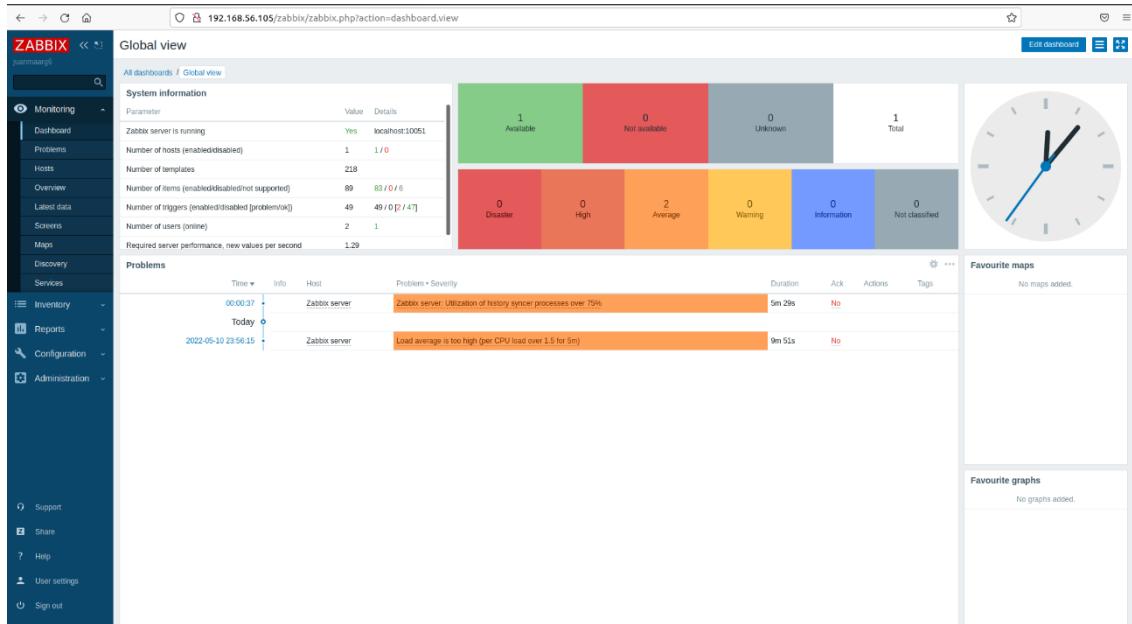
```

root@localhost juanmaarg6]# sudo firewall-cmd --add-port=10050/tcp --permanent
success
[root@localhost juanmaarg6]# sudo firewall-cmd --add-port=10050/tcp
success
[root@localhost juanmaarg6]# sudo firewall-cmd --add-port=10050/tcp --list-ports
usage: see firewall-cmd man page
firewall-cmd: error: argument --list-ports: not allowed with argument --add-port
[root@localhost juanmaarg6]# sudo firewall-cmd --list-ports
22022/tcp 10050/tcp
[root@localhost juanmaarg6]#

```

## 1.5. Quinto Paso: Inclusión de los Zabbix-Agent en el Frontend

Volvemos al frontend de Zabbix en nuestro navegador.



Nos vamos a **Configuración -> Hosts -> Crear Host** y escribimos toda la información relativa a la máquina de CentOS.

The screenshot shows the Zabbix Hosts configuration page. The left sidebar is highlighted with a red box, showing options like Configuration, Host groups, Templates, Hosts, Maintenance, Actions, Event correlation, Discovery, Services, and Administration. The main area shows a table of hosts, with one entry for 'Zabbix server' highlighted. The 'Create host' button in the top right corner is also highlighted with a red box. The URL in the browser is 192.168.56.105/zabbix/hosts.php.

ZABBIX juanmaarg6: Configuration + 192.168.56.105/zabbix/hosts.php?form=create

**Hosts**

Host Hostname: CentOS Agent  
Visible name: CentOS Agent  
Groups: Linux servers  
Interfaces: Type: IP address: 192.168.56.110, Connect to: IP, Port: 10050, Default: IP  
Description: Monitorización de CentOS  
Monitored by proxy: (no proxy)  
Enabled:  Add Cancel



Hacemos lo mismo para Ubuntu Server:

ZABBIX juanmaarg6: Configuration + 192.168.56.105/zabbix/hosts.php?form=create

**Hosts**

Host Hostname: Ubuntu Agent  
Visible name: Ubuntu Agent  
Groups: Linux servers  
Interfaces: Type: IP address: 192.168.56.105, Connect to: IP, Port: 10050, Default: IP  
Description: Monitorización de Ubuntu  
Monitored by proxy: (no proxy)  
Enabled:  Add Cancel



Esto es lo que nos tiene que aparecer ahora en **Configuración -> Hosts**:

The screenshot shows the Zabbix web interface at <http://192.168.56.105/zabbix/hosts.php>. The left sidebar is titled 'ZABBIX' and includes 'Monitoring', 'Inventory', 'Reports', 'Configuration' (selected), 'Host groups', 'Templates', 'Hosts' (selected), 'Maintenance', 'Actions', 'Event correlation', 'Discovery', 'Services', and 'Administration'. The main content area has a green banner at the top stating 'Host added'. Below it, there are search and filter fields for 'Host groups', 'Templates', 'Name', 'DNS', 'IP', and 'Port'. A table lists three hosts:

Name	Applications	Items	Triggers	Graphs	Discovery	Web	Interface	Proxy	Templates	Status	Availability	Agent encryption	Info	Tags
CentOS Agent	Applications	Items	Triggers	Graphs	Discovery	Web	192.168.56.110:10050			Enabled	ZBX SNMP JMX IPMI	NONE		
Ubuntu Agent	Applications	Items	Triggers	Graphs	Discovery	Web	192.168.56.105:10050			Enabled	ZBX SNMP JMX IPMI	NONE		
Zabbix server	Applications 12	Items 89	Triggers 49	Graphs 18	Discovery 3	Web 127.0.0.1:10050			Template App Zabbix Server, Template OS Linux by Zabbix agent, Template Module Linux block devices by Zabbix agent, Template Module Linux CPU by Zabbix agent, Template Module Linux filesystems by Zabbix agent, Template Module Linux generic by Zabbix agent, Template Module Linux memory by Zabbix agent, Template Module Linux network interfaces by Zabbix agent, Template Module Zabbix agent	Enabled	ZBX SNMP JMX IPMI	NONE		

At the bottom, there are buttons for '0 selected', 'Enable', 'Disable', 'Export', 'Mass update', and 'Delete'. A note says 'Displaying 3 of 3 found'.

Ahora vamos a configurar los servicios que queremos monitorizar en los hosts, que son SSH y HTTP. Para ello, pinchamos en *CentOS Agent* (aparece dentro del recuadro rojo de la imagen anterior). Luego nos vamos a la pestaña *Templates* y añadimos *Template App HTTP Service* y *Template App SSH Service*. Finalmente, pulsamos en *Update*.

The screenshot shows the Zabbix web interface at <http://192.168.56.105/zabbix/hosts.php?form=update&hostid=10436>. The left sidebar is identical to the previous screenshot. The main content area shows the 'Hosts' configuration for 'CentOS Agent'. The 'Templates' tab is selected and highlighted with a red box. Below it, there is a section titled 'Link new templates' with a search bar containing 'Template App HTTP Service' and 'Template App SSH Service'. At the bottom of this section is a blue 'Update' button, which is also highlighted with a red box. Other buttons in this section include 'Clone', 'Full clone', 'Delete', and 'Cancel'.

Hacemos lo mismo para Ubuntu Server, de forma que tiene que quedar como sigue:

The screenshot shows the Zabbix interface for managing hosts. On the left, there's a sidebar with navigation links like Monitoring, Inventory, Reports, Configuration (selected), Host groups, Templates, Hosts, Maintenance, Actions, Event correlation, Discovery, Services, and Administration. The main area is titled 'Hosts' and shows a list of three hosts:

Name	Type	Items	Triggers	Graphs	Discovery	Web	Interface	Proxy	Templates	Status	Availability	Agent encryption	Info	Tags		
CentOS Agent	Applications	2	Items	2	Triggers	2	Graphs	Discovery	Web	192.168.56.110:10500	Template App HTTP Service, Template App SSH Service	Enabled	ZBX SNMP JMX IPMI	NONE		
Ubuntu Agent	Applications	2	Items	2	Triggers	2	Graphs	Discovery	Web	192.168.56.105:10500	Template App HTTP Service, Template App SSH Service	Enabled	ZBX SNMP JMX IPMI	NONE		
Zabbix server	Applications	17	Items	119	Triggers	69	Graphs	23	Discovery	3	Web	127.0.0.1:10500	Template App Zabbix Server, Template OS Linux by Zabbix agent, Template Module Linux block devices by Zabbix agent, Template Module Linux CPU by Zabbix agent, Template Module Linux filesystems by Zabbix agent, Template Module Linux generic by Zabbix agent, Template Module Linux memory by Zabbix agent, Template Module Linux network interfaces by Zabbix agent, Template Module Zabbix agent	Enabled	ZBX SNMP JMX IPMI	NONE

At the bottom, there are buttons for '0 selected', 'Enable', 'Disable', 'Export', 'Mass update', and 'Delete'.

Pero recordemos que tenemos asociado el servicio SSH al puerto 22022 en vez de al puerto por defecto, el puerto 22 (el puerto de HTTP es el puerto 80, el puerto por defecto). Por tanto, para que el servicio SSH se monitoree correctamente, tenemos que **crear un nuevo Template de SSH**.

Para ello, nos vamos a *Templates -> Template App SSH Service*. Luego, nos vamos a la pestaña *Items* y le damos a *Create item*:

The screenshot shows the Zabbix interface for managing items under the 'Template App SSH Service' template. The top navigation bar has a red box around the 'Create item' button. The main area is titled 'Items' and shows a list of one item:

Wizard	Name	Triggers	Key	Interval	History	Trends	Type	Applications	Status	Info
	SSH service is running	Triggers 1	net.tcp.service[ssh]	1m	1w	365d	Simple check	SSH service	Enabled	

At the bottom, there are buttons for '0 selected', 'Enable', 'Disable', 'Execute now', 'Clear history', 'Copy', 'Mass update', and 'Delete'.

A continuación, añadimos un nuevo ítem con los siguientes valores:

The screenshot shows the Zabbix interface for creating a new item. The URL is `192.168.56.105/zabbix/items.php?form=create&hostid=10102`. The 'Item' tab is active. The form fields are as follows:

- Name: SSH service is running on Port 22022
- Type: Simple check
- Key: net.tcp.service[ssh,,22022]
- User name: (empty)
- Password: (empty)
- Type of information: Numeric (unsigned)
- Units: (empty)
- Update interval: 1m
- Custom intervals:
  - Type: Flexible
  - Interval: 50s
  - Period: 1-7,00:00-24:00
  - Action: Remove
- History storage period: Do not keep history (Storage period: 90d)
- Trend storage period: Do not keep trends (Storage period: 365d)
- Show value: Service state
- New application: (empty)
- Applications: -None- (SSH service is highlighted)
- Populates host inventory field: -None-
- Description: (empty)

También es necesario especificar un *Trigger* que nos avise cuando no está escuchando en el puerto 22022. Para ello, nos vamos a *Templates* -> *Template App SSH Service*. Luego, nos vamos a la pestaña *Triggers* y le damos a *Create trigger*:

The screenshot shows the Zabbix interface for managing triggers. The URL is `192.168.56.105/zabbix/triggers.php?filter_set=1&filter_hostids[0]=10102`. A red box highlights the search bar and filter section. A red rectangle highlights the 'Create trigger' button in the top right corner. The triggers table shows one entry:

Severity	Name	Operational data	Expression	Status	Tags
Average	SSH service is down on [HOST NAME]	(empty)	[Template App SSH Service].net.tcp.service[ssh,max(#3)]=0	Enabled	(empty)

Below the table, there are buttons for selecting, enabling, disabling, copying, mass updating, and deleting triggers.

A continuación, añadimos un nuevo trigger con los siguientes valores:

The screenshot shows the Zabbix web interface under the 'juanmaarg6' configuration. The left sidebar is visible with various monitoring and configuration options. The main content area is titled 'Triggers' and shows a form for creating a new trigger. The 'Trigger' tab is selected. The configuration fields include:

- Name:** SSH service on Port 22022 is down on {HOST.NAME}
- Operational data:** (empty)
- Severity:** Average (highlighted in orange)
- Expression:** `{Template App SSH Service:net.tcp.service[ssh,,22022].max(#3)=0}`
- OK event generation:** Expression
- PROBLEM event generation mode:** Single
- OK event closes:** All problems
- Description:** (empty)
- Enabled:** checked

At the bottom are buttons for **Update**, **Clone**, **Delete**, and **Cancel**.

Finalmente, los triggers del template *Template App SSH Service* quedan como sigue:

The screenshot shows the Zabbix triggers list page. A green success message at the top says 'Trigger updated'. The search bar shows 'Trigger updated'. The filters section includes:

- Host groups:** type here to search, Select
- Tags:** And/Or, Or, tag, Contains, Equals, value, Remove
- Hosts:** **Template App SSH Service**, Select
- Name:** (empty)
- Severity:** Not classified, Warning, High, Information, Average, Disaster
- Inherited:** all, Yes, No
- State:** all, Normal, Unknown
- Discovered:** all, Yes, No
- Status:** all, Enabled, Disabled
- With dependencies:** all, Yes, No
- Value:** all, Ok, Problem

The results table shows two triggers:

Severity	Name	Operational data	Expression	Status	Tags
Average	SSH service is down on {HOSTNAME}	(empty)	<code>{Template App SSH Service:net.tcp.service[ssh].max(#3)=0}</code>	Enabled	
Average	SSH service on Port 22022 is down on {HOSTNAME}	(empty)	<code>{Template App SSH Service:net.tcp.service[ssh,,22022].max(#3)=0}</code>	Enabled	

At the bottom are buttons for **Apply** and **Reset**. A note at the bottom right says 'Displaying 2 of 2 found'.

Y los ítems del template *Template App SSH Service* quedarían:

Name	Triggers	Key	Interval	History	Trends	Type	Applications	Status	Info
SSH service is running	Triggers 1	net.tcp.service[ssh]	1m	1w	365d	Simple check	SSH service	Enabled	
SSH service is running on Port 22022	Triggers 1	net.tcp.service[ssh,22022]	1m	90d	365d	Simple check	SSH service	Enabled	

## 1.6. Sexto Paso: Comprobación de la Monitorización en el Frontend

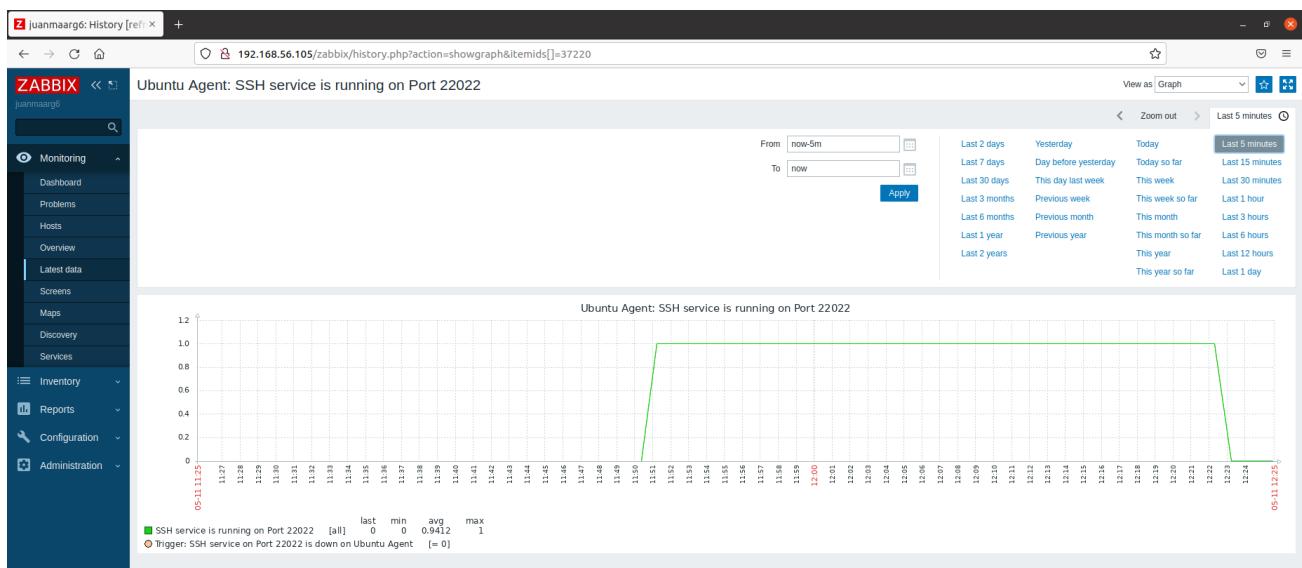
Vamos a comprobar que se monitorizan correctamente los servicios SSH y HTTP tanto en Ubuntu Server como en CentOS. Para ello, nos vamos a **Monitoring -> Latest Data**.

Host	Name	Last check	Last value	Change
CentOS Agent	HTTP service (1 Item)	2022-05-11 12:09:59	Down (0)	Graph
CentOS Agent	SSH service (2 Items)	2022-05-11 12:09:59	Down (0)	Graph
CentOS Agent	SSH service is running	2022-05-11 12:09:59	Down (0)	Graph
CentOS Agent	SSH service is running on Port 22022	2022-05-11 12:15:22	Down (0)	Graph
Ubuntu Agent	HTTP service (1 Item)	2022-05-11 12:09:58	Up (1)	Graph
Ubuntu Agent	SSH service (2 Items)	2022-05-11 12:09:59	Down (0)	Graph
Ubuntu Agent	SSH service is running	2022-05-11 12:09:59	Down (0)	Graph
Ubuntu Agent	SSH service is running on Port 22022	2022-05-11 12:15:20	Up (1)	Graph

Vamos a comenzar comprobando los servicios en **Ubuntu Server**. Si vamos a **Ubuntu Agent -> SSH service is running on Port 22022 -> Graph** veremos la gráfica que monitoriza el funcionamiento del servicio. Si probamos a encender el servicio SSH en Ubuntu Server y luego lo volvemos a encender, esto se mostrará en la gráfica.

```
juanmaarg6@ise-ubuntu:~$ sudo systemctl start sshd
[sudo] password for juanmaarg6:
juanmaarg6@ise-ubuntu:~$ sudo systemctl status sshd
● ssh.service - OpenBSD Secure Shell server
   Loaded: loaded (/lib/systemd/system/ssh.service; enabled; vendor preset: enabled)
   Active: active (running) since Wed 2022-05-11 08:42:00 UTC; 1h 36min ago
     Docs: man:sshd(8)
           man:sshd_config(5)
  Process: 974 ExecStartPre=/usr/sbin/sshd -t (code=exited, status=0/SUCCESS)
 Main PID: 1104 (sshd)
    Tasks: 1 (limit: 5786)
   Memory: 3.8M
      CGroup: /system.slice/ssh.service
              └─1104 sshd: /usr/sbin/sshd -D [listener] 0 of 10-100 startups

May 11 10:09:20 ise-ubuntu sshd[4839]: Connection closed by 192.168.56.105 port 47122 [preauth]
May 11 10:10:20 ise-ubuntu sshd[4874]: Connection closed by 192.168.56.105 port 47238 [preauth]
May 11 10:11:20 ise-ubuntu sshd[4904]: Connection closed by 192.168.56.105 port 47378 [preauth]
May 11 10:12:20 ise-ubuntu sshd[4935]: Connection closed by 192.168.56.105 port 47486 [preauth]
May 11 10:13:20 ise-ubuntu sshd[4968]: Connection closed by 192.168.56.105 port 47594 [preauth]
May 11 10:14:20 ise-ubuntu sshd[5002]: Connection closed by 192.168.56.105 port 47728 [preauth]
May 11 10:15:20 ise-ubuntu sshd[5038]: Connection closed by 192.168.56.105 port 47840 [preauth]
May 11 10:16:20 ise-ubuntu sshd[5074]: Connection closed by 192.168.56.105 port 47950 [preauth]
May 11 10:17:20 ise-ubuntu sshd[5107]: Connection closed by 192.168.56.105 port 48088 [preauth]
May 11 10:18:20 ise-ubuntu sshd[5231]: Connection closed by 192.168.56.105 port 48198 [preauth]
juanmaarg6@ise-ubuntu:~$ sudo systemctl stop sshd
juanmaarg6@ise-ubuntu:~$ sudo systemctl status sshd
```



Si vamos ahora a **Ubuntu Agent -> HTTP service is running -> Graph** y hacemos lo mismo que hemos hecho con SSH:

```
juanmaarg6@ise-ubuntu:~$ sudo systemctl start apache2
juanmaarg6@ise-ubuntu:~$ sudo systemctl status apache2
● apache2.service - The Apache HTTP Server
   Loaded: loaded (/lib/systemd/system/apache2.service; enabled; vendor preset: enabled)
   Active: active (running) since Wed 2022-05-11 08:43:41 UTC; 1h 44min ago
     Docs: https://httpd.apache.org/docs/2.4/
  Process: 1716 ExecStart=/usr/sbin/apachectl start (code=exited, status=0/SUCCESS)
 Main PID: 1738 (apache2)
    Tasks: 12 (limit: 5786)
   Memory: 86.0M
      CPU: 0.000 CPU(s) used
      CGroup: /system.slice/apache2.service
           ├─1738 /usr/sbin/apache2 -k start
           ├─1742 /usr/sbin/apache2 -k start
           ├─1812 /usr/sbin/apache2 -k start
           ├─2079 /usr/sbin/apache2 -k start
           ├─2175 /usr/sbin/apache2 -k start
           ├─2255 /usr/sbin/apache2 -k start
           ├─2937 /usr/sbin/apache2 -k start
           ├─2944 /usr/sbin/apache2 -k start
           ├─5056 /usr/sbin/apache2 -k start
           ├─5745 /usr/sbin/apache2 -k start
           ├─5746 /usr/sbin/apache2 -k start
           └─5747 /usr/sbin/apache2 -k start

May 11 08:43:41 ise-ubuntu systemd[1]: apache2.service: Succeeded.
May 11 08:43:41 ise-ubuntu systemd[1]: Stopped The Apache HTTP Server.
May 11 08:43:41 ise-ubuntu systemd[1]: Starting The Apache HTTP Server...
May 11 08:43:41 ise-ubuntu apachectl[1730]: AH00558: apache2: Could not reliably determine the server's fully qualified name, using ise-ubuntu for Port 80
May 11 08:43:41 ise-ubuntu systemd[1]: Started The Apache HTTP Server.

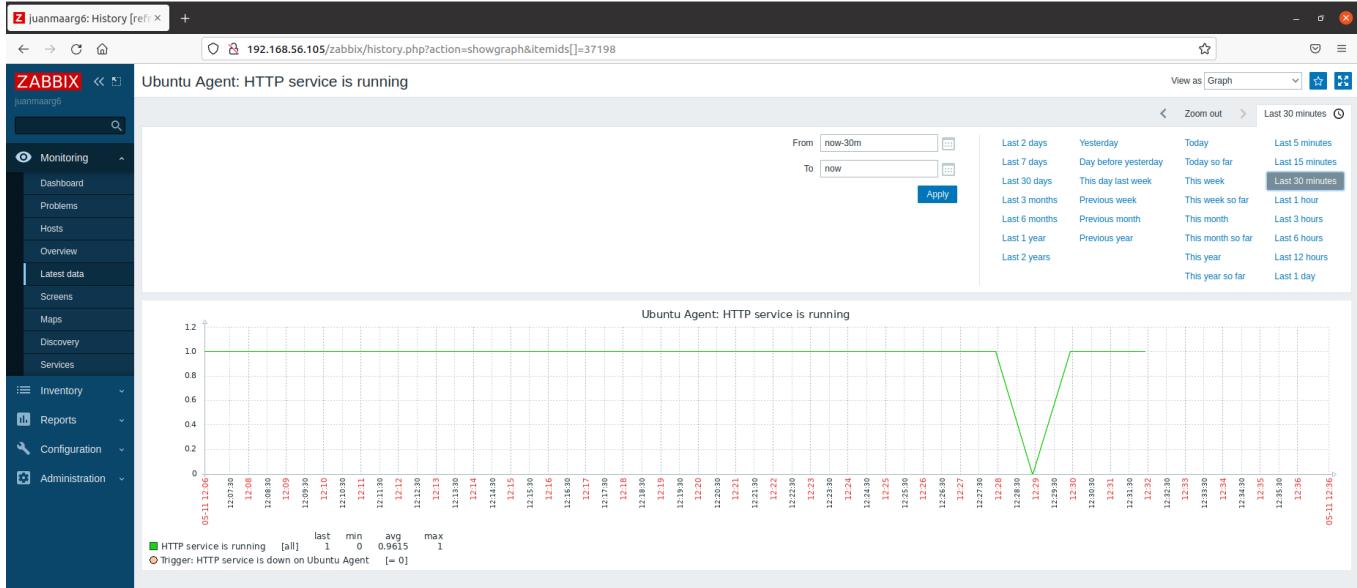
[3]+  Stopped                  sudo systemctl status apache2
juanmaarg6@ise-ubuntu:~$ sudo systemctl stop apache2
juanmaarg6@ise-ubuntu:~$ sudo systemctl status apache2
```

```
Main PID: 1738 (code=exited, status=0/SUCCESS)

May 11 08:43:41 ise-ubuntu systemd[1]: apache2.service: Succeeded.
May 11 08:43:41 ise-ubuntu systemd[1]: Stopped The Apache HTTP Server.
May 11 08:43:41 ise-ubuntu systemd[1]: Starting The Apache HTTP Server...
May 11 08:43:41 ise-ubuntu apachectl[1730]: AH00558: apache2: Could not reliably determine the server's fully qualified name, using ise-ubuntu for Port 80
May 11 08:43:41 ise-ubuntu systemd[1]: Started The Apache HTTP Server.
May 11 10:28:56 ise-ubuntu systemd[1]: Stopping The Apache HTTP Server...
May 11 10:28:56 ise-ubuntu apachectl[5998]: AH00558: apache2: Could not reliably determine the server's fully qualified name, using ise-ubuntu for Port 80
May 11 10:28:56 ise-ubuntu systemd[1]: apache2.service: Succeeded.
May 11 10:28:56 ise-ubuntu systemd[1]: Stopped The Apache HTTP Server.
lines 1-17/17 (END)

[4]+  Stopped                  sudo systemctl status apache2
juanmaarg6@ise-ubuntu:~$ sudo systemctl start apache2
sudo systemctl start apache2
juanmaarg6@ise-ubuntu:~$ sudo systemctl start apache2
juanmaarg6@ise-ubuntu:~$ sudo systemctl status apache2
● apache2.service - The Apache HTTP Server
   Loaded: loaded (/lib/systemd/system/apache2.service; enabled; vendor preset: enabled)
   Active: active (running) since Wed 2022-05-11 10:29:36 UTC; 7s ago
     Docs: https://httpd.apache.org/docs/2.4/
  Process: 6039 ExecStart=/usr/sbin/apachectl start (code=exited, status=0/SUCCESS)
 Main PID: 6061 (apache2)
    Tasks: 6 (limit: 5786)
   Memory: 12.0M
      CPU: 0.000 CPU(s) used
      CGroup: /system.slice/apache2.service
           ├─6061 /usr/sbin/apache2 -k start
           ├─6062 /usr/sbin/apache2 -k start
           ├─6063 /usr/sbin/apache2 -k start
           ├─6064 /usr/sbin/apache2 -k start
           ├─6065 /usr/sbin/apache2 -k start
           └─6066 /usr/sbin/apache2 -k start

May 11 10:29:36 ise-ubuntu systemd[1]: Starting The Apache HTTP Server...
May 11 10:29:36 ise-ubuntu apachectl[6053]: AH00558: apache2: Could not reliably determine the server's fully qualified name, using ise-ubuntu for Port 80
May 11 10:29:36 ise-ubuntu systemd[1]: Started The Apache HTTP Server.
lines 1-19/19 (END)
```



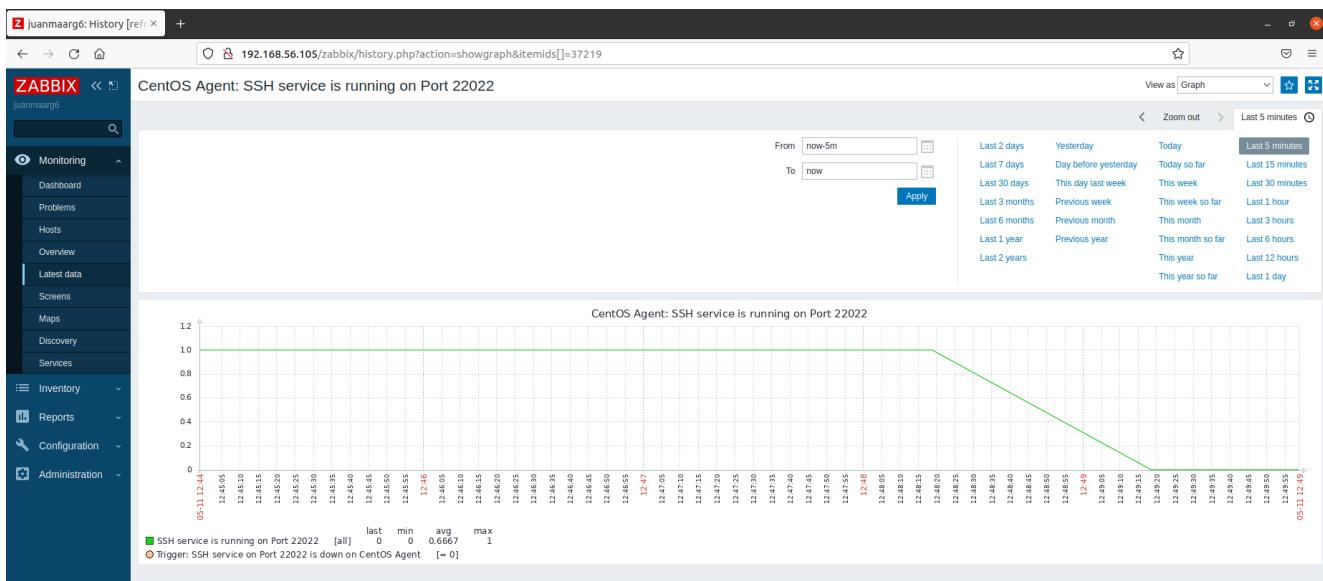
Finalizamos comprobando los servicios en **CentOS**. Si vamos a **CentOS Agent -> SSH service is running on Port 22022 -> Graph** y hacemos lo mismo que hicimos con SSH para Ubuntu Server:

```
[juanmaarg6@localhost ~]$: sudo systemctl start sshd
[sudo] password for juanmaarg6:
[juanmaarg6@localhost ~]$: sudo systemctl status sshd
● sshd.service - OpenSSH server daemon
  Loaded: loaded (/usr/lib/systemd/system/sshd.service; enabled; vendor preset: enabled)
  Active: active (running) since Wed 2022-05-11 06:45:32 EDT; 2min 49s ago
    Docs: man:sshd(8)
           man:sshd_config(5)
   Main PID: 909 (sshd)
      Tasks: 1 (limit: 30258)
     Memory: 2.9M
       CGroup: /system.slice/sshd.service
               └─909 /usr/sbin/sshd -D -oCiphers=aes256-gcm@openssh.com,chacha20-poly1305@openssh.com,a>

may 11 06:45:31 localhost.localdomain systemd[1]: Starting OpenSSH server daemon...
may 11 06:45:32 localhost.localdomain sshd[909]: Server listening on 0.0.0.0 port 22022.
may 11 06:45:32 localhost.localdomain sshd[909]: Server listening on :: port 22022.
may 11 06:45:32 localhost.localdomain systemd[1]: Started OpenSSH server daemon.

[1]+  Detenido                  sudo systemctl status sshd
[juanmaarg6@localhost ~]$: sudo systemctl stop sshd
[juanmaarg6@localhost ~]$: sudo systemctl status sshd
● sshd.service - OpenSSH server daemon
  Loaded: loaded (/usr/lib/systemd/system/sshd.service; enabled; vendor preset: enabled)
  Active: inactive (dead) since Wed 2022-05-11 06:48:34 EDT; 2s ago
    Docs: man:sshd(8)
           man:sshd_config(5)
  Process: 909 ExecStart=/usr/sbin/sshd -D $OPTIONS ${CRYPTO_POLICY} (code=exited, status=0/SUCCESS)
  Main PID: 909 (code=exited, status=0/SUCCESS)

may 11 06:45:31 localhost.localdomain systemd[1]: Starting OpenSSH server daemon...
may 11 06:45:32 localhost.localdomain sshd[909]: Server listening on 0.0.0.0 port 22022.
may 11 06:45:32 localhost.localdomain sshd[909]: Server listening on :: port 22022.
may 11 06:45:32 localhost.localdomain systemd[1]: Started OpenSSH server daemon.
may 11 06:48:34 localhost.localdomain systemd[1]: Stopping OpenSSH server daemon...
may 11 06:48:34 localhost.localdomain systemd[1]: Stopped OpenSSH server daemon.
[juanmaarg6@localhost ~]$
```



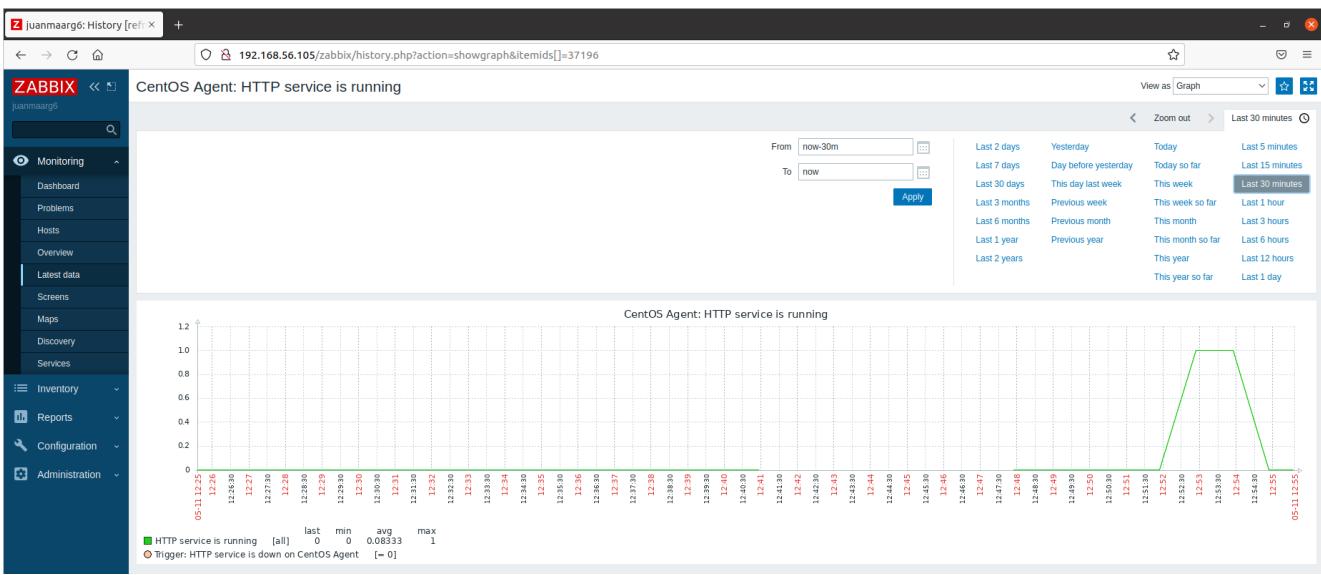
Si vamos ahora a **CentOS Agent -> HTTP service is running -> Graph** y hacemos lo mismo que hemos hecho con SSH:

```
[juanmaarg6@localhost ~]$: sudo systemctl start httpd
[juanmaarg6@localhost ~]$: sudo systemctl status httpd
● httpd.service - The Apache HTTP Server
  Loaded: loaded (/usr/lib/systemd/system/httpd.service; disabled; vendor preset: disabled)
  Drop-In: /usr/lib/systemd/system/httpd.service.d
            └─php-fpm.conf
    Active: active (running) since Wed 2022-05-11 06:53:35 EDT; 2s ago
      Docs: man:httpd.service(8)
   Main PID: 1761 (httpd)
     Status: "Started, listening on: port 80"
       Tasks: 213 (limit: 30258)
      Memory: 24.4M
     CGroup: /system.slice/httpd.service
             ├─1761 /usr/sbin/httpd -DFOREGROUND
             ├─1762 /usr/sbin/httpd -DFOREGROUND
             ├─1763 /usr/sbin/httpd -DFOREGROUND
             ├─1764 /usr/sbin/httpd -DFOREGROUND
             ├─1765 /usr/sbin/httpd -DFOREGROUND

may 11 06:53:35 localhost.localdomain systemd[1]: Starting The Apache HTTP Server...
may 11 06:53:35 localhost.localdomain httpd[1761]: AH00558: httpd: Could not reliably determine the
may 11 06:53:35 localhost.localdomain httpd[1761]: Server configured, listening on: port 80
may 11 06:53:35 localhost.localdomain systemd[1]: Started The Apache HTTP Server.
Lines 1-21/21 (END)
[4]+  Detenido                  sudo systemctl status httpd
```

```
[juanmaarg6@localhost ~]$: sudo systemctl stop httpd
[juanmaarg6@localhost ~]$: sudo systemctl status httpd
● httpd.service - The Apache HTTP Server
  Loaded: loaded (/usr/lib/systemd/system/httpd.service; disabled; vendor preset: disabled)
  Drop-In: /usr/lib/systemd/system/httpd.service.d
            └─php-fpm.conf
    Active: inactive (dead)
      Docs: man:httpd.service(8)

may 11 06:52:49 localhost.localdomain systemd[1]: Started The Apache HTTP Server.
may 11 06:52:50 localhost.localdomain httpd[1519]: Server configured, listening on: port 80
may 11 06:53:18 localhost.localdomain systemd[1]: Stopping The Apache HTTP Server...
may 11 06:53:19 localhost.localdomain systemd[1]: Stopped The Apache HTTP Server.
may 11 06:53:35 localhost.localdomain systemd[1]: Starting The Apache HTTP Server...
may 11 06:53:35 localhost.localdomain httpd[1761]: AH00558: httpd: Could not reliably determine the
may 11 06:53:35 localhost.localdomain httpd[1761]: Server configured, listening on: port 80
may 11 06:53:35 localhost.localdomain systemd[1]: Started The Apache HTTP Server.
may 11 06:53:58 localhost.localdomain systemd[1]: Stopping The Apache HTTP Server...
may 11 06:53:59 localhost.localdomain systemd[1]: Stopped The Apache HTTP Server.
Lines 1-17/17 (END)
```



## **2. Ejercicio 2**

### **2.0. Enunciado del Ejercicio**

*Usted deberá saber cómo instalar y configurar Ansible para poder hacer un ping a las máquinas virtuales de los servidores y ejecutar un comando básico (p.ej. el script de monitorización del RAID1). También debe ser consciente de la posibilidad de escribir acciones más complejas mediante playbooks escritos con YAML. Incluya capturas de pantalla del proceso con una breve descripción en el mismo documento que suba para el ejercicio de Zabbix.*

### **2.1. Primer Paso: Creación del Script, del Timer y del Service**

Nos situamos en **Ubuntu Server**. En la misma carpeta `/home` creamos un script llamado `mon_raid.py` y lo editaremos. Para crearlo y modificarlo directamente usamos `sudo nano mon_raid.py`. Escribimos lo que aparece en la siguiente imagen:

```
GNU nano 4.8                               mon_raid.py                                         Modified
import re

f=open('/proc/mdstat')
for line in f:
    b=re.findall('^\[[\w]*[\_]+[\w]*\]',line)
    if(b!=[]):
        print("ERROR EN RAID")
print("OK SCRIPT")
```

Comprobamos el funcionamiento del script con `python3` (ya que es un script de `python`):

```
juanmaarg6@ise-ubuntu:~$ python3 mon_raid.py
OK SCRIPT
juanmaarg6@ise-ubuntu:~$
```

Vemos que funciona correctamente. Ahora vamos a crear los ficheros *timer* y *service* en la carpeta */etc/systemd/system*. Estando en la carpeta */home*, comenzamos creando el *service* con *sudo nano /etc/systemd/system/mon\_raid.service* y escribimos lo que aparece en la siguiente imagen:

```
GNU nano 4.8                               /etc/systemd/system/mon_raid.service
[Unit]
Description=Monitor RAID status
[Service]
Type=simple
ExecStart=/usr/bin/python3 /home/juanmaarg6/mon-raid.py
```

Para el *timer*, hacemos *sudo nano /etc/systemd/system/mon\_raid.timer* y escribimos en él lo siguiente:

```
GNU nano 4.8                               /etc/systemd/system/mon_raid.timer
[Unit]
Description=Monitor RAID status
[Timer]
OnCalendar=minutely
[Install]
WantedBy=timers.target
```

Ahora ya podemos manejar *mon\_raid* como si fuera un servicio del sistema con *systemctl*. Vamos a activar el servicio y hacer que se inicie al arrancar el sistema.

```
juanmaarg6@ise-ubuntu:~$ sudo systemctl start mon_raid.service mon_raid.timer
juanmaarg6@ise-ubuntu:~$ sudo systemctl enable mon_raid.service mon_raid.timer
juanmaarg6@ise-ubuntu:~$ _
```

Comprobamos que tanto el *service* como el *timer* están activos:

```
juanmaarg6@ise-ubuntu:~$ sudo systemctl status mon_raid.service
● mon_raid.service - Monitor RAID status
  Loaded: loaded (/etc/systemd/system/mon_raid.service; static; vendor preset: enabled)
  Active: active (running) since Wed 2022-05-11 11:10:10 UTC; 8ms ago
TriggeredBy: • mon_raid.timer
    Main PID: 2065 (python3)
      Tasks: 1 (limit: 5786)
     Memory: 1.9M
        CGroup: /system.slice/mon_raid.service
                  └─2065 /usr/bin/python3 /home/juanmaarg6/mon-raid.py

May 11 11:10:10 ise-ubuntu systemd[1]: Started Monitor RAID status.
juanmaarg6@ise-ubuntu:~$ sudo systemctl status mon_raid.timer
● mon_raid.timer - Monitor RAID status
  Loaded: loaded (/etc/systemd/system/mon_raid.timer; enabled; vendor preset: enabled)
  Active: active (waiting) since Wed 2022-05-11 11:00:58 UTC; 9min ago
    Trigger: Wed 2022-05-11 11:11:00 UTC; 40s left
  Triggers: • mon_raid.service

May 11 11:00:58 ise-ubuntu systemd[1]: Started Monitor RAID status.
juanmaarg6@ise-ubuntu:~$ _
```

## 2.2. Segundo Paso: Consultar Log del Servicio de Monitorización del RAID

Una vez creados y configurados correctamente todos los archivos anteriores, podemos acceder al log de eventos que muestra el servicio creado mediante el comando *journalctl*:

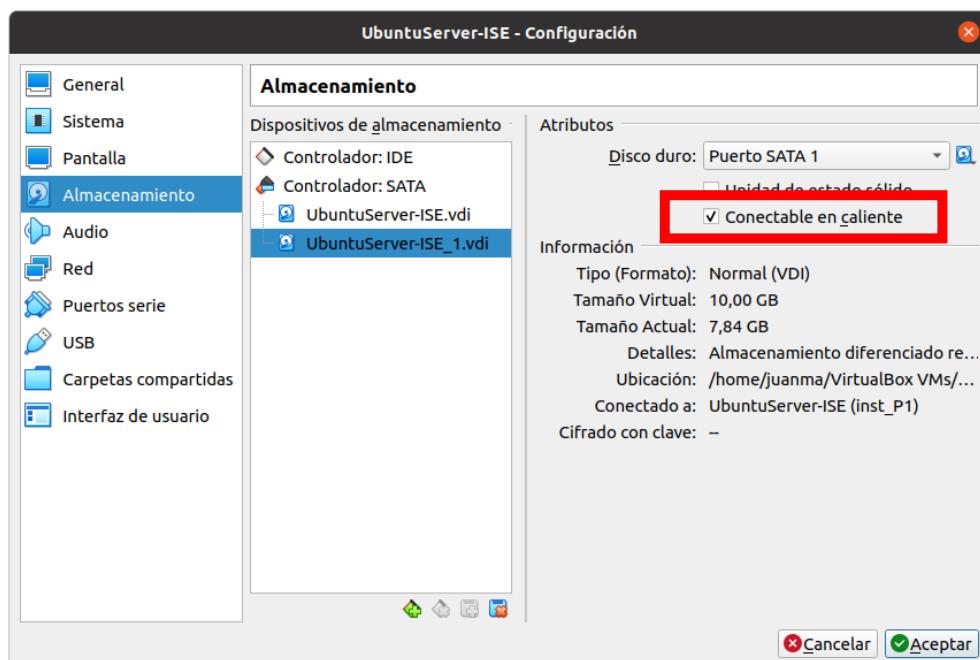
```
sudo journalctl -u mon_raid --since="yesterday"
```

donde “yesterday” es una unidad de tiempo que podemos modificar según queramos para ver los logs desde un momento determinado hasta actualmente (en este caso, veríamos los logs desde ayer hasta actualmente).

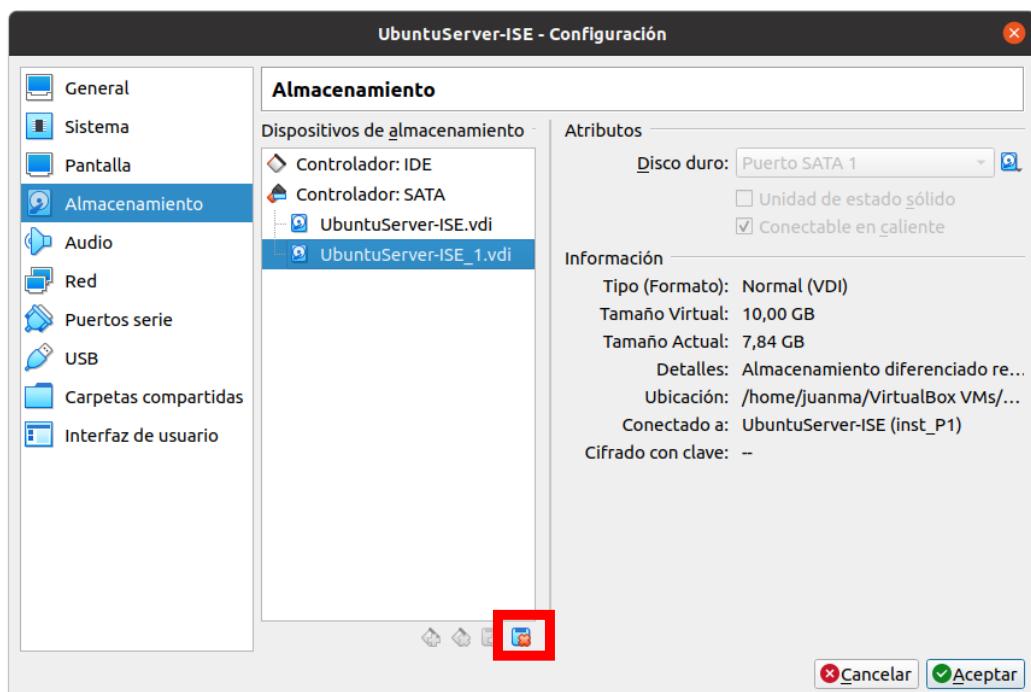
```
-- Logs begin at Fri 2022-04-22 19:39:39 UTC, end at Wed 2022-05-11 11:11:20 UTC. --
May 10 20:15:00 ise-ubuntu systemd[1]: Started Monitor RAID status.
May 10 20:15:03 ise-ubuntu python3[976]: --OK Script--
May 10 20:15:02 ise-ubuntu systemd[1]: mon_raid.service: Succeeded.
May 10 20:16:00 ise-ubuntu systemd[1]: Started Monitor RAID status.
May 10 20:16:00 ise-ubuntu python3[1162]: --OK Script--
May 10 20:16:00 ise-ubuntu systemd[1]: mon_raid.service: Succeeded.
May 10 20:17:19 ise-ubuntu systemd[1]: Started Monitor RAID status.
May 10 20:17:19 ise-ubuntu python3[1657]: --OK Script--
May 10 20:17:19 ise-ubuntu systemd[1]: mon_raid.service: Succeeded.
May 10 20:18:00 ise-ubuntu systemd[1]: Started Monitor RAID status.
May 10 20:18:00 ise-ubuntu python3[1775]: --OK Script--
May 10 20:18:00 ise-ubuntu systemd[1]: mon_raid.service: Succeeded.
May 10 20:19:20 ise-ubuntu python3[2383]: --OK Script--
May 10 20:19:20 ise-ubuntu systemd[1]: mon_raid.service: Succeeded.
May 10 20:20:20 ise-ubuntu systemd[1]: Started Monitor RAID status.
May 10 20:20:20 ise-ubuntu systemd[1]: mon_raid.service: Succeeded.
May 10 20:20:20 ise-ubuntu python3[2641]: --OK Script--
May 10 20:21:04 ise-ubuntu systemd[1]: Started Monitor RAID status.
May 10 20:21:04 ise-ubuntu python3[4534]: --OK Script--
May 10 20:21:04 ise-ubuntu systemd[1]: mon_raid.service: Succeeded.
May 10 20:22:35 ise-ubuntu systemd[1]: Started Monitor RAID status.
May 10 20:22:35 ise-ubuntu python3[5614]: --OK Script--
May 10 20:22:35 ise-ubuntu systemd[1]: mon_raid.service: Succeeded.
May 10 20:23:06 ise-ubuntu systemd[1]: Started Monitor RAID status.
May 10 20:23:06 ise-ubuntu python3[5631]: --OK Script--
May 10 20:23:06 ise-ubuntu systemd[1]: mon_raid.service: Succeeded.
May 10 20:24:03 ise-ubuntu systemd[1]: Started Monitor RAID status.
May 10 20:24:03 ise-ubuntu python3[5653]: --OK Script--
May 10 20:24:03 ise-ubuntu systemd[1]: mon_raid.service: Succeeded.
May 10 20:25:21 ise-ubuntu systemd[1]: Started Monitor RAID status.
May 10 20:25:21 ise-ubuntu systemd[1]: mon_raid.service: Succeeded.
May 10 20:25:21 ise-ubuntu python3[5671]: --OK Script--
May 10 20:26:15 ise-ubuntu systemd[1]: Started Monitor RAID status.
May 10 20:26:15 ise-ubuntu python3[5695]: --OK Script--
lines 1-36
```

Vamos a hacer ahora una prueba más realista para comprobar que nuestro servicio está funcionando correctamente. Lo que vamos a hacer es desactivar uno de los discos duros de máquina de Ubuntu Server, provocando así un error en el RAID, y luego comprobaremos en el log si se ha registrado dicho fallo.

Para ello, comenzamos haciendo en Virtualbox una instantánea de la máquina virtual en la que todo funcione correctamente (para regresar a ella tras hacer esta prueba). Luego, en Virtualbox, nos vamos a la Configuración de la máquina de Ubuntu Server y pinchamos en Almacenamiento. Pinchamos, por ejemplo, en el segundo disco del RAID y le marcamos la opción *Conecitable en caliente*.



Le damos a *Aceptar* e iniciamos la máquina de Ubuntu Server. Una vez iniciada la máquina virtual, con la máquina encendida volvemos a ir en VirtualBox al apartado de Almacenamiento de la máquina de Ubuntu Server. Cogemos el disco anterior y lo desconectamos pinchando en el icono de abajo a la derecha.



Le damos a *Aceptar*. Tras unos segundos, nos tiene que salir el siguiente mensaje de error en Ubuntu Server:

```
[ 222.600325] blk_update_request: I/O error, dev sdb, sector 6954496 op 0x0:(READ) flags 0x4000 phys_seg 166 prio class 0
[ 222.600372] sd 3:0:0:0: rejecting I/O to offline device
[ 222.600390] blk_update_request: I/O error, dev sdb, sector 6956800 op 0x0:(READ) flags 0x0 phys_seg 141 prio class 0
[ 222.600431] sd 3:0:0:0: rejecting I/O to offline device
[ 222.600449] blk_update_request: I/O error, dev sdb, sector 7231960 op 0x0:(READ) flags 0x0 phys_seg 1 prio class 0
[ 222.600482] md/raid1:md1: sdb3: rescheduling sector 6595032
[ 222.600502] sd 3:0:0:0: rejecting I/O to offline device
[ 222.600520] blk_update_request: I/O error, dev sdb, sector 3042736 op 0x1:(WRITE) flags 0x0 phys_seg 1 prio class 0
[ 222.600553] sd 3:0:0:0: rejecting I/O to offline device
[ 222.600570] blk_update_request: I/O error, dev sdb, sector 3061648 op 0x1:(WRITE) flags 0x0 phys_seg 1 prio class 0
[ 222.600603] sd 3:0:0:0: rejecting I/O to offline device
[ 222.600621] blk_update_request: I/O error, dev sdb, sector 3064032 op 0x1:(WRITE) flags 0x0 phys_seg 1 prio class 0
[ 222.600654] sd 3:0:0:0: rejecting I/O to offline device
[ 222.600672] blk_update_request: I/O error, dev sdb, sector 674960 op 0x1:(WRITE) flags 0x0 phys_seg 1 prio class 0
[ 222.600704] sd 3:0:0:0: rejecting I/O to offline device
[ 222.600722] blk_update_request: I/O error, dev sdb, sector 2779144 op 0x1:(WRITE) flags 0x0 phys_seg 1 prio class 0
[ 222.600755] sd 3:0:0:0: rejecting I/O to offline device
[ 222.600773] sd 3:0:0:0: rejecting I/O to offline device
[ 222.600792] sd 3:0:0:0: rejecting I/O to offline device
[ 222.600810] sd 3:0:0:0: rejecting I/O to offline device
[ 222.600827] sd 3:0:0:0: rejecting I/O to offline device
[ 222.600846] sd 3:0:0:0: rejecting I/O to offline device
[ 222.601582] md: super_written gets error=10
[ 222.602205] md/raid1:md1: Disk failure on sdb3, disabling device.
[ 222.602205] md/raid1:md1: Operation continuing on 1 devices.
[ 222.603573] md: super_written gets error=10
[ 231.370760] md/raid1:md0: Disk failure on sdb2, disabling device.
[ 231.370760] md/raid1:md0: Operation continuing on 1 devices.
```

Si ahora hacemos `sudo journalctl -u mon_raid --since="yesterday"`:

```
May 11 11:06:52 ise-ubuntu systemd[1]: Started Monitor RAID status.
May 11 11:06:52 ise-ubuntu python3[1803]: --OK Script--
May 11 11:06:52 ise-ubuntu systemd[1]: mon_raid.service: Succeeded.
May 11 11:07:20 ise-ubuntu systemd[1]: Started Monitor RAID status.
May 11 11:07:20 ise-ubuntu systemd[1]: mon_raid.service: Succeeded.
May 11 11:07:21 ise-ubuntu python3[1816]: --OK Script--
May 11 11:08:16 ise-ubuntu systemd[1]: Started Monitor RAID status.
May 11 11:08:16 ise-ubuntu python3[1850]: --OK Script--
May 11 11:08:16 ise-ubuntu systemd[1]: mon_raid.service: Succeeded.
May 11 11:08:41 ise-ubuntu systemd[1]: Started Monitor RAID status.
May 11 11:08:41 ise-ubuntu python3[1888]: --OK Script--
May 11 11:08:41 ise-ubuntu systemd[1]: mon_raid.service: Succeeded.
May 11 11:09:18 ise-ubuntu systemd[1]: Started Monitor RAID status.
May 11 11:09:32 ise-ubuntu python3[1951]: --OK Script--
May 11 11:09:18 ise-ubuntu systemd[1]: mon_raid.service: Succeeded.
May 11 11:09:49 ise-ubuntu systemd[1]: Started Monitor RAID status.
May 11 11:09:49 ise-ubuntu python3[2058]: --OK Script--
May 11 11:09:49 ise-ubuntu systemd[1]: mon_raid.service: Succeeded.
May 11 11:10:10 ise-ubuntu systemd[1]: Started Monitor RAID status.
May 11 11:10:10 ise-ubuntu python3[2065]: --OK Script--
May 11 11:10:10 ise-ubuntu systemd[1]: mon_raid.service: Succeeded.
May 11 11:11:20 ise-ubuntu systemd[1]: Started Monitor RAID status.
May 11 11:11:20 ise-ubuntu python3[2107]: --OK Script--
May 11 11:11:20 ise-ubuntu systemd[1]: mon_raid.service: Succeeded.
-- Reboot --
May 11 11:16:00 ise-ubuntu systemd[1]: Started Monitor RAID status.
May 11 11:16:01 ise-ubuntu python3[1032]: --OK Script--
May 11 11:16:01 ise-ubuntu systemd[1]: mon_raid.service: Succeeded.
May 11 11:17:07 ise-ubuntu systemd[1]: Started Monitor RAID status.
May 11 11:17:07 ise-ubuntu python3[1564]: --OK Script--
May 11 11:17:07 ise-ubuntu systemd[1]: mon_raid.service: Succeeded.
May 11 11:18:09 ise-ubuntu systemd[1]: Started Monitor RAID status.
May 11 11:18:09 ise-ubuntu python3[1618]: --ERROR en RAID-- 
May 11 11:18:09 ise-ubuntu python3[1618]: --ERROR en RAID--
May 11 11:18:09 ise-ubuntu python3[1618]: --OK Script--
May 11 11:18:09 ise-ubuntu systemd[1]: mon_raid.service: Succeeded.
lines 825-860/860 (END)
```

Vemos que el servicio funciona tal y como se debe esperar. Aunque tengamos un disco duro menos, el sistema sigue funcionando perfectamente. Podemos comprobar con el comando `lsblk` que realmente hemos dejado de tener un disco duro en el sistema:

```
juanmaarg6@ise-ubuntu:~$ lsblk
NAME      MAJ:MIN RM  SIZE RO TYPE  MOUNTPOINT
loop0      7:0    0   55M  1 loop  /snap/core18/1880
loop1      7:1    0 55.5M  1 loop  /snap/core18/2344
loop2      7:2    0 61.9M  1 loop  /snap/core20/1405
loop3      7:3    0 61.9M  1 loop  /snap/core20/1434
loop4      7:4    0 71.8M  1 loop  /snap/lxd/16099
loop5      7:5    0 67.8M  1 loop  /snap/lxd/22753
loop6      7:6    0 44.7M  1 loop  /snap/snappyd/15534
sda        8:0    0   10G  0 disk
└─sda1     8:1    0   1M  0 part
└─sda2     8:2    0 300M  0 part
  └─md0      9:0    0 299M  0 raid1
    └─md0p1   259:1  0 294M  0 part  /boot
  └─sda3     8:3    0 9.7G  0 part
  └─md1      9:1    0 9.7G  0 raid1
    └─md1p1   259:0  0 9.7G  0 part
      └─dm_crypt-0 253:0  0 9.7G  0 crypt
        ├─vg0-lv--home 253:1  0   1G  0 lvm   /home
        ├─vg0-lv--root 253:2  0   8G  0 lvm   /
        └─vg0-lv--swap 253:3  0 688M 0 lvm   [SWAP]
juanmaarg6@ise-ubuntu:~$
```

Vemos que *sdb* no aparece. Una vez realizada esta prueba, restauramos la máquina al estado anterior mediante la instantánea que realizamos anteriormente. Tras hacer esto, ahora sí nos debe de aparecer el disco *sdb* al ejecutar el comando *lsblk*.

```
juanmaarg6@ise-ubuntu:~$ lsblk
NAME      MAJ:MIN RM  SIZE RO TYPE  MOUNTPOINT
loop0      7:0    0 55.5M  1 loop  /snap/core18/2344
loop1      7:1    0 61.9M  1 loop  /snap/core20/1405
loop2      7:2    0 55M   1 loop  /snap/core18/1880
loop3      7:3    0 61.9M  1 loop  /snap/core20/1434
loop4      7:4    0 71.3M  1 loop  /snap/lxd/16099
loop5      7:5    0 44.7M  1 loop  /snap/snapd/15534
loop6      7:6    0 67.8M  1 loop  /snap/lxd/22753
sda        8:0    0 10G   0 disk
└─sda1     8:1    0 1M    0 part
└─sda2     8:2    0 300M  0 part
  └─md0     9:0    0 299M  0 raid1
    └─md0p1  259:1 0 294M  0 part  /boot
  └─sda3     8:3    0 9.7G  0 part
  └─md1     9:1    0 9.7G  0 raid1
    └─md1p1  259:0 0 9.7G  0 part
      └─dm_crypt-0 253:0 0 9.7G  0 crypt
        ├─vg0-lv--home 253:1 0 1G   0 lvm   /home
        ├─vg0-lv--root 253:2 0 8G   0 lvm   /
        └─vg0-lv--swap 253:3 0 688M 0 lvm   [SWAP]
sdb        8:16   0 10G   0 disk
└─sdb1     8:17   0 1M    0 part
└─sdb2     8:18   0 300M  0 part
  └─md0     9:0    0 299M  0 raid1
    └─md0p1  259:1 0 294M  0 part  /boot
  └─sdb3     8:19   0 9.7G  0 part
  └─md1     9:1    0 9.7G  0 raid1
    └─md1p1  259:0 0 9.7G  0 part
      └─dm_crypt-0 253:0 0 9.7G  0 crypt
        ├─vg0-lv--home 253:1 0 1G   0 lvm   /home
        ├─vg0-lv--root 253:2 0 8G   0 lvm   /
        └─vg0-lv--swap 253:3 0 688M 0 lvm   [SWAP]
juanmaarg6@ise-ubuntu:~$
```

## 2.3. Tercer Paso: Instalación y Configuración de Ansible

Ansible es una herramienta que permite extender la monitorización de los servicios de más de un servidor y, a diferencia de Zabbix, no necesita de agente ya que usa SSH. Para instalar Ansible en Ubuntu Server ejecutamos el siguiente comando:

```
sudo apt install ansible
```

```
juanmaarg6@ise-ubuntu:~$ sudo apt install ansible
[sudo] password for juanmaarg6:
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following package was automatically installed and is no longer required:
  libfwupdplugin1
Use 'sudo apt autoremove' to remove it.
The following additional packages will be installed:
  ieee-data python3-argcomplete python3-crypto python3-dnspython python3-jmespath python3-kerberos
  python3-libcloud python3-lockfile python3-netaddr python3-ntlm-auth python3-requests-kerberos
  python3-requests-ntlm python3-selinux python3-winrm python3-xmldict
Suggested packages:
  cowsay sshpass python-lockfile-doc ipython3 python-netaddr-docs
The following NEW packages will be installed:
  ansible ieee-data python3-argcomplete python3-crypto python3-dnspython python3-jmespath
  python3-kerberos python3-libcloud python3-lockfile python3-netaddr python3-ntlm-auth
  python3-requests-kerberos python3-requests-ntlm python3-selinux python3-winrm python3-xmldict
0 upgraded, 16 newly installed, 0 to remove and 16 not upgraded.
Need to get 9644 kB of archives.
After this operation, 90.2 MB of additional disk space will be used.
Do you want to continue? [Y/n]
```

Si todo ha ido correctamente, podemos pasar a comprobar la versión de Ansible que se ha instalado con el comando *ansible --version*:

```
juanmaarg6@ise-ubuntu:~$ ansible --version
ansible 2.9.6
  config file = /etc/ansible/ansible.cfg
  configured module search path = ['/home/juanmaarg6/.ansible/plugins/modules', '/usr/share/ansible/plugins/modules']
    ansible python module location = /usr/lib/python3/dist-packages/ansible
      executable location = /usr/bin/ansible
        python version = 3.8.10 (default, Mar 15 2022, 12:22:08) [GCC 9.4.0]
juanmaarg6@ise-ubuntu:~$
```

Ansible tiene dos ficheros fundamentales que se encuentran en la carpeta */etc/ansible*:

- 1) *ansible.cfg* (Fichero de configuración de Ansible)
- 2) *hosts* (Fichero que almacena las direcciones IP de las máquinas que queremos monitorizar)

Comenzamos accediendo al fichero *hosts* con *sudo nano /etc/ansible/hosts* y añadiremos las IPs de Ubuntu Server, 192.168.56.105, y de CentOS, 192.168.56.110 (ya que estas son las máquinas que queremos monitorizar) en cualquier sitio del fichero.

```

GNU nano 4.8                               /etc/ansible/hosts
# - Blank lines are ignored
# - Groups of hosts are delimited by [header] elements
# - You can enter hostnames or ip addresses
# - A hostname/ip can be a member of multiple groups

# Ex 1: Ungrouped hosts, specify before any group headers.

#green.example.com
#blue.example.com
#192.168.100.1
#192.168.100.10

192.168.56.105 ←
192.168.56.110

# Ex 2: A collection of hosts belonging to the 'webservers' group

#[webservers]
#alpha.example.org
#beta.example.org
#192.168.1.100
#192.168.1.110

# If you have multiple hosts following a pattern you can specify
# them like this:

```

Veamos ahora si Ansible se puede conectar a ambas máquinas. Para ello, vamos a lanzarles un ping a ambas máquinas con Ansible ejecutando el siguiente comando:

*ansible all -m ping -u juanmaarg6*

```

juanmaarg6@ise-ubuntu:~$ ansible all -m ping -u juanmaarg6
192.168.56.105 | UNREACHABLE! => {
    "changed": false,
    "msg": "Failed to connect to the host via ssh: ssh: connect to host 192.168.56.105 port 22: Connection refused",
    "unreachable": true
}
192.168.56.110 | UNREACHABLE! => {
    "changed": false,
    "msg": "Failed to connect to the host via ssh: ssh: connect to host 192.168.56.110 port 22: Connection refused",
    "unreachable": true
}
juanmaarg6@ise-ubuntu:~$ _

```

Como podemos observar, no se está conectando correctamente a ninguna de las dos máquinas. Esto se debe a que Ansible usa SSH, como ya dijimos antes, luego, tiene establecido el puerto 22 como puerto remoto por defecto, pero nosotros tenemos configurado SSH en el puerto 22022. Por tanto, vamos a cambiar el puerto remoto que usa Ansible accediendo al archivo *ansible.cfg* con el comando *sudo nano /etc/ansible/ansible.cfg*.

```

GNU nano 4.8                               /etc/ansible/ansible.cfg
# config file for ansible -- https://ansible.com/
# =====

# nearly all parameters can be overridden in ansible-playbook
# or with command line flags. ansible will read ANSIBLE_CONFIG,
# ansible.cfg in the current working directory, .ansible.cfg in
# the home directory or /etc/ansible/ansible.cfg, whichever it
# finds first

[defaults]

# some basic default values...

#inventory      = /etc/ansible/hosts
#library        = /usr/share/my_modules/
#module_utils   = /usr/share/my_module_utils/
#remote_tmp     = ~/.ansible/tmp
#local_tmp      = ~/.ansible/tmp
#plugin_filters_cfg = /etc/ansible/plugin_filters.yml
#forks          = 5
#poll_interval  = 15
#sudo_user      = root
#ask_sudo_pass  = True
#ask_pass        = True
#transport      = smart
remote_port    = 22022 ←
#module_lang    = C
#module_set_locale = False

```

Una vez hecho esto, si volvemos a ejecutar el comando `ansible all -m ping -u juanmaarg6`, obtendremos lo siguiente:

```

juanmaarg6@ise-ubuntu:~$ ansible all -m ping -u juanmaarg6
192.168.56.105 | UNREACHABLE! => {
  "changed": false,
  "msg": "Failed to connect to the host via ssh: juanmaarg6@192.168.56.105: Permission denied (publickey,password).",
  "unreachable": true
}
192.168.56.110 | SUCCESS => {
  "ansible_facts": {
    "discovered_interpreter_python": "/usr/libexec/platform-python"
  },
  "changed": false,
  "ping": "pong"
}
juanmaarg6@ise-ubuntu:~$ _

```

Ahora sí podemos alcanzar la máquina de CentOS con Ansible pero no alcanzamos la nuestra propia, es decir, Ubuntu Server (recordar que todo esto lo estamos realizando en el propio Ubuntu Server). Esto se debe a que necesitamos copiar la clave pública del SSH de Ubuntu Server en sí mismo. Para ello, como ya tenemos la clave pública generada (si no la tuviéramos ejecutamos el comando `ssh-keygen` y le damos enter a todos los pasos), solo necesitamos copiarla a la misma máquina de Ubuntu Server con el siguiente comando de SSH:

```
ssh-copy-id juanmaarg6@192.168.56.105 -p 22022
```

```

juanmaarg6@ise-ubuntu:~$ ssh-copy-id juanmaarg6@192.168.56.105 -p 22022
/usr/bin/ssh-copy-id: INFO: Source of key(s) to be installed: "/home/juanmaarg6/.ssh/id_rsa.pub"
/usr/bin/ssh-copy-id: INFO: attempting to log in with the new key(s), to filter out any that are already installed
/usr/bin/ssh-copy-id: INFO: 1 key(s) remain to be installed -- if you are prompted now it is to install all the new keys
juanmaarg6@192.168.56.105's password:

Number of key(s) added: 1

Now try logging into the machine, with: "ssh -p '22022' 'juanmaarg6@192.168.56.105'"
and check to make sure that only the key(s) you wanted were added.

juanmaarg6@ise-ubuntu:~$
```

Si ahora volvemos a ejecutar el comando *ansible all -m ping -u juanmaarg6*, veremos que el *ping* ya se realiza correctamente a ambas máquinas:

```

juanmaarg6@ise-ubuntu:~$ ansible all -m ping -u juanmaarg6
192.168.56.105 | SUCCESS => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/bin/python3"
    },
    "changed": false,
    "ping": "pong"
}
192.168.56.110 | SUCCESS => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/libexec/platform-python"
    },
    "changed": false,
    "ping": "pong"
}
juanmaarg6@ise-ubuntu:~$ _
```

## **2.4. Cuarto Paso: Comprobación Simultánea de RAID en Ubuntu Server y CentOS mediante Ansible**

Vamos a comprobar el estado del RAID tanto en Ubuntu Server como en CentOS mediante el script *mon\_raid.py* que creamos anteriormente (ahora nos debería salir que todo está correcto en ambos RAID). Para hacerlo de forma simultánea en ambas máquinas, usaremos Ansible en Ubuntu Server. Ejecutamos el siguiente comando:

```
ansible all -a "python3 /home/juanmaarg6/mon_raid.py" -u juanmaarg6
```

```
juanmaarg6@ise-ubuntu:~$ ansible all -a "python3 /home/juanmaarg6/mon_raid.py" -u juanmaarg6
192.168.56.110 | FAILED | rc=2 >>
python3: can't open file '/home/juanmaarg6/mon_raid.py': [Errno 2] No such file or directorynon-zero
return code
192.168.56.105 | CHANGED | rc=0 >>
OK SCRIPT
juanmaarg6@ise-ubuntu:~$ _
```

Como podemos observar, en Ubuntu Server todo funciona correctamente, pero en CentOS nos da un error. Esto se debe a que no tenemos el script *mon\_raid.py* en la máquina de CentOS (lo creamos solo en Ubuntu Server). Si hacemos un *ls* en el */home* de CentOS no nos saldrá ningún archivo.

```
[juanmaarg6@localhost ~]$ ls
[juanmaarg6@localhost ~]$ _
```

Para solucionar dicho error, vamos a enviarle el script a CentOS ejecutando el siguiente comando en Ubuntu Server:

```
scp -P 22022 /home/juanmaarg6/mon_raid.py juanmaarg6@192.168.56.110:/home/juanmaarg6/mon_raid.py
```

```
juanmaarg6@ise-ubuntu:~$ scp -P 22022 /home/juanmaarg6/mon_raid.py juanmaarg6@192.168.56.110:/home/j
uanmaarg6/mon_raid.py
mon_raid.py                                              100%   144    59.5KB/s   00:00
juanmaarg6@ise-ubuntu:~$ _
```

Si volvemos a hacer *ls* en el */home* de CentOS ya si nos aparecerá el script.

```
[juanmaarg6@localhost ~]$ ls
mon_raid.py
[juanmaarg6@localhost ~]$ _
```

Finalmente, si volvemos a ejecutar el comando *ansible all -a "python3 /home/juanmaarg6/mon\_raid.py" -u juanmaarg6* en Ubuntu Server, veremos que el script se ejecuta correctamente en ambas máquinas y que el RAID funciona perfectamente en ambas.

```
juanmaarg6@ise-ubuntu:~$ ansible all -a "python3 /home/juanmaarg6/mon_raid.py" -u juanmaarg6
192.168.56.105 | CHANGED | rc=0 >>
OK SCRIPT
192.168.56.110 | CHANGED | rc=0 >>
OK SCRIPT
juanmaarg6@ise-ubuntu:~$
```

## 2.5. Playbooks de Ansible

Hasta el momento, has podido utilizar Ansible para realizar operaciones mas o menos sencillas de forma aislada, es decir, no hemos podido realizar varias tareas encadenadas. ¿Qué ocurre si queremos lanzar más de una tarea a varias máquinas? Para eso tenemos la potente herramienta conocida como los **Playbooks de Ansible**.

Los *Playbooks* no son más que archivos de texto que describen de manera declarativa el estado necesario a aplicar en los servidores administrados. El lenguaje con el que se expresan los playbooks es YAML (parecido a un JSON). No se trata por tanto de un lenguaje de programación, sino más bien un lenguaje de declaración de datos.

Los *Playbooks* nos permiten lanzar varias tareas en un solo proceso, así como, realizar tareas más complejas. Además, otra de las grandes ventajas de los *Playbooks*, es que ofrecen la posibilidad de someterlos a **control de versiones**.

A nivel de código, un *Playbook* está formado por una lista de **plays**. Y un play es una lista de tareas que se aplican sobre un determinado conjunto de hosts. Cada tarea no es más que una llamada a un módulo de Ansible. Digamos que cada tarea es como *un comando ad-hoc*.

Vamos a hacer un par de ejemplos de *Playbooks* de Ansible. Vamos a comenzar realizando un **playbook que lance tanto a CentOS como a Ubuntu Server, el comando ping y la ejecución del script mon\_raid.py**. Para ello, comenzamos editando el archivo *hosts* de Ansible, creando un grupo, que llamaremos *servidoresP3*, con las IPs de CentOS y de Ubuntu Server.

```
GNU nano 4.8                               /etc/ansible/hosts                         Modified
# This is the default ansible 'hosts' file.
#
# It should live in /etc/ansible/hosts
#
#   - Comments begin with the '#' character
#   - Blank lines are ignored
#   - Groups of hosts are delimited by [header] elements
#   - You can enter hostnames or ip addresses
#   - A hostname/ip can be a member of multiple groups
#
# Ex 1: Ungrouped hosts, specify before any group headers.

#green.example.com
#blue.example.com
#192.168.100.1
#192.168.100.10

[servidoresP3]
192.168.56.105
192.168.56.110

# Ex 2: A collection of hosts belonging to the 'webservers' group
#
#[webservers]
#alpha.example.org
#beta.example.org
```

Ahora vamos a crear el playbook. Hacemos *sudo nano playbook.yml* y escribimos lo siguiente:

```
GNU nano 4.8                               playbook.yml
---
- name: Hacer ping y ejecutar mon_raid.py
  hosts: servidoresP3
  remote_user: root

  tasks:
    - name: Hacer ping
      ping:

    - name: Ejecutar mon_raid.py
      command: python3 /home/juanmaarg6/mon_raid.py
```

Con *name* estamos indicando un nombre de la tarea que vamos a realizar. En *hosts* indicamos al grupo de hosts que queremos lanzar las tareas recogidas en *tasks*. Con *remote\_user* indicamos el usuario del servidor al que le vamos a lanzar las tareas (en este caso hemos puesto *root*). Estamos indicando el *remote\_user* para todo el playbook pero también se puede indicar por cada tarea, de forma que podemos lanzar diferentes tareas a diferentes usuarios (otra ventaja que nos ofrecen los *Playbooks* de Ansible).

Para ejecutar el playbook, lanzamos el siguiente comando:

```
ansible-playbook playbook.yml
```

```
juanmaarg6@ise-ubuntu:~$ ansible-playbook playbook.yml
PLAY [Hacer ping y ejecutar mon_raid.py] ****
TASK [Gathering Facts] ****
fatal: [192.168.56.105]: UNREACHABLE! => {"changed": false, "msg": "Failed to connect to the host via ssh: root@192.168.56.105: Permission denied (publickey,password).", "unreachable": true}
fatal: [192.168.56.110]: UNREACHABLE! => {"changed": false, "msg": "Failed to connect to the host via ssh: root@192.168.56.110: Permission denied (publickey,gssapi-keyex,gssapi-with-mic,password).", "unreachable": true}

PLAY RECAP ****
192.168.56.105          : ok=0    changed=0    unreachable=1    failed=0    skipped=0    rescued=0    ignored=0
192.168.56.110          : ok=0    changed=0    unreachable=1    failed=0    skipped=0    rescued=0    ignored=0
juanmaarg6@ise-ubuntu:~$
```

Como podemos ver, nos da error en la conexión a los hosts. La solución a este problema se encuentra en el siguiente enlace:

<https://www.enmimaquinafunciona.com/pregunta/101817/error-de-ansible-ssh-aunque-el-ssh-estandar-funciona>

Por tanto, ahora lanzamos el comando para ejecutar el playbook de Ansible, pero añadiéndole la opción `--connection=local`. El resultado es el siguiente:

```
juanmaarg6@ise-ubuntu:~$ ansible-playbook playbook.yml --connection=local
PLAY [Hacer ping y ejecutar mon_raid.py] ****
TASK [Gathering Facts] ****
ok: [192.168.56.110]
ok: [192.168.56.105]

TASK [Hacer ping] ****
ok: [192.168.56.110]
ok: [192.168.56.105]

TASK [Ejecutar mon_raid.py] ****
changed: [192.168.56.110]
changed: [192.168.56.105]

PLAY RECAP ****
192.168.56.105      : ok=3    changed=1    unreachable=0    failed=0    skipped=0    rescued=
0      ignored=0
192.168.56.110      : ok=3    changed=1    unreachable=0    failed=0    skipped=0    rescued=
0      ignored=0
juanmaarg6@ise-ubuntu:~$
```

Como vemos, ahora todo funciona perfectamente. A la hora de los resultados que se nos muestra en pantalla tras lanzar el comando, cabe comentar que nos sale que se han ejecutado tres tareas (Gathering Facts, Hacer ping, Ejecutar mon\_raid.py). La primera de todas las tareas, Gathering Facts, se hace de forma automática y simplemente recoge información de los hosts remotos. Si no queremos que esto se realice, tenemos que añadir la opción `gather_facts: no` en el `play` del playbook.

Ahora vamos a realizar un **playbook que lance un ping a CentOS y ejecute script mon\_raid.py en Ubuntu Server**. Para ello, volvemos a editar el archivo `hosts` de Ansible, creando un primer grupo, que llamaremos `servs1`, con las IP de Ubuntu Server y un segundo grupo, que llamaremos `servs2`, con la IP de CentOS.

```

GNU nano 4.8                               /etc/ansible/hosts
# This is the default ansible 'hosts' file.
#
# It should live in /etc/ansible/hosts
#
#   - Comments begin with the '#' character
#   - Blank lines are ignored
#   - Groups of hosts are delimited by [header] elements
#   - You can enter hostnames or ip addresses
#   - A hostname/ip can be a member of multiple groups

# Ex 1: Ungrouped hosts, specify before any group headers.

#green.example.com
#blue.example.com
#192.168.100.1
#192.168.100.10

[servs1]
192.168.56.105

[servs2]
192.168.56.110

# Ex 2: A collection of hosts belonging to the 'webservers' group

```

Ahora vamos a modificar el playbook anterior. Hacemos *sudo nano playbook.yml* y escribimos lo siguiente:

```

GNU nano 4.8                               playbook.yml
---
- name: Hacer ping_
  hosts: servs1
  remote_user: root

  tasks:
    - name: Hacer ping
      ping:

- name: Ejecutar mon_raid.py
  hosts: servs2
  remote_user: root

  tasks:
    - name: Ejecutar mon_raid.py
      command: python3 /home/juanmaarg6/mon_raid.py

```

Para ejecutar el playbook, lanzamos el siguiente comando:

```
ansible-playbook playbook.yml --connection=local
```

```
juanmaarg6@ise-ubuntu:~$ ansible-playbook playbook.yml --connection=local
PLAY [Hacer ping] ****
TASK [Gathering Facts] ****
ok: [192.168.56.105]
TASK [Hacer ping] ****
ok: [192.168.56.105]
PLAY [Ejecutar mon_raid.py] ****
TASK [Gathering Facts] ****
ok: [192.168.56.110]
TASK [Ejecutar mon_raid.py] ****
changed: [192.168.56.110]
PLAY RECAP ****
192.168.56.105 : ok=2    changed=0    unreachable=0    failed=0    skipped=0    rescued=0    ignored=0
192.168.56.110 : ok=2    changed=1    unreachable=0    failed=0    skipped=0    rescued=0    ignored=0
juanmaarg6@ise-ubuntu:~$ _
```

Como podemos observar, todo ha funcionado perfectamente.

Hemos podido ver un poco acerca del potencial de los Playbooks de Ansible pero existen muchísimas cosas que saber. Aquí hemos comentado lo más básico, pero ya vemos las grandes posibilidades que nos ofrece.

### **3. Referencias**

- <https://www.zabbix.com/>
- <https://www.zabbix.com/documentation/5.0/en/manual>
- <https://es.wikipedia.org/wiki/Zabbix>
- <https://www.digitalocean.com/community/tutorials/how-to-install-and-configure-zabbix-to-securely-monitor-remote-servers-on-ubuntu-20-04-es>
- <https://www.vultr.com/docs/how-to-configure-a-new-ubuntu-server-with-ansible/>
- <https://www.digitalocean.com/community/tutorials/how-to-install-and-configure-ansible-on-ubuntu-20-04-es>
- [https://docs.ansible.com/ansible/latest/user\\_guide/playbooks\\_intro.html](https://docs.ansible.com/ansible/latest/user_guide/playbooks_intro.html)
- <https://www.arsys.es/blog/primer-playbook-ansible>
- <https://atareao.es/tutorial/ansible/playbooks-de-ansible/>