

Memoria Práctica 3

Daniel Alconchel Vázquez

Ejercicio 1: Zabbix

Desde **Ubuntu Server**, y aprovechando la configuración de *apache*, *mysql* y *ssh* de prácticas anteriores; vamos a instalar **Zabbix**. Este programa nos permite monitorizar distintos elementos (routers, VPNs, ...), llamarlos y preguntarles información sobre su estado (telemetría: información, temperatura; procesos más complejo: número de instancias de Apache, cuánta memoria usa MySQL, cuántos usuarios...)

Para instalarlo, vamos a seguir los pasos indicados en la [documentación oficial](#). En nuestro caso, vamos a instalar la *versión 5.0*. Si observamos, vemos que hay bastantes comandos que tenemos que ejecutar como usuario root.

Install and configure Zabbix server for your platform

a. Install Zabbix repository

[Documentation](#)

```
# 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
```

b. Install Zabbix server, frontend, agent

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

c. Create initial database

[Documentation](#)

Make sure you have database server up and running.

Run the following on your database host.

```
# 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;
```

Lo más cómodo, es configurar ssh para que nos podamos conectar como usuario root (ya hemos visto los pasos en prácticas anteriores) y conectarnos desde nuestra máquina *Linux* o *Windows*. De esta forma, podemos copiar directamente los comandos, sin necesidad de teclearlos a mano.

Una vez ejecutados los pasos indicados en la página antes mencionada, tenemos que abrir en nuestro navegador el frontend. Para ello entramos en la dirección `http://server_ip/zabbix`. En nuestro caso, como estamos trabajando desde *Ubuntu Server*, la ip correspondiente será la `192.168.56.105`.

Tras iniciar el frontend, nos saldrá la siguiente pestaña:



Seguimos los pasos indicados hasta completar la instalación.

Si echamos nuevamente un vistazo a la [documentación](#) veremos que para logearnos simplemente tenemos que introducir `Username: Admin` y `Password: zabbix`.

Vamos a comenzar monitorizando CentOS, ya es que es un poco más tedioso.

Si seguimos consultando la [documentación](#), nuestro siguiente paso es crear un nuevo **host** para monitorizar CentOS. Para ello, vamos a comenzar instalando `zabbix-agent` en CentOS, siguiendo la información proporcionada en esta [página](#). En nuestro caso, estamos usando **Zabbix 5.0**, por lo que lo unico que cambia es en el paso 2, en vez de añadir el repositorio indicado (que es el Zabbix 4.4), añadimos el del 5.0. Los pasos son los siguientes:

```
sudo nano /etc/selinux/config
# Change the following line: SELINUX=disable
sudo dnf install https://repo.zabbix.com/zabbix/5.0/rhel/8/x86_64/zabbix-
release-5.0-1.el8.noarch.rpm
sudo dnf install zabbix-agent
```

La configuración también viene explicada en el enlace anterior, pero vamos a detallarla de todas formas:

```
sudo nano /etc/zabbix/zabbix_agentd.conf
```

```
#Server=[zabbix server ip]
#Hostname=[ Hostname of client system ]

Server=192.168.56.105, 127.0.0.1
ServerActive=192.168.56.105, 127.0.0.1
Hostname=CentOS
```

```
# Habilitamos los puertos de zabbix
sudo firewall-cmd --permanent --add-port=10050/tcp
sudo firewall-cmd --reload

# Empezamos el servicio
sudo systemctl enable zabbix-agent
sudo systemctl start zabbix-agent
service zabbix-agent start
```

Volviendo a [documentación](#), seguimos los pasos para crear un host:

Hosts

The screenshot shows the Zabbix web interface for creating a new host. The 'Host' tab is selected in the top navigation bar. The form contains the following fields and options:

- Host name:** A text input field containing 'CentOS'.
- Visible name:** An empty text input field.
- Groups:** A dropdown menu showing 'Linux servers' and 'Zabbix servers' as selected items. A 'Select' button is to the right.
- Interfaces:** A table with columns: Type, IP address, DNS name, Connect to, Port, and Default.

Type	IP address	DNS name	Connect to	Port	Default
Agent	192.168.56.110		IP DNS	10050	<input checked="" type="radio"/> Remove
- Add:** A blue button below the interfaces table.
- Description:** A large text area for adding a description.
- Monitored by proxy:** A dropdown menu with 'no proxy' selected.
- Enabled:** A checkbox that is checked.
- Buttons:** 'Add' and 'Cancel' buttons at the bottom.

El siguiente paso es crear un nuevo ítem. Para ello observemos, como hasta ahora, la [documentación oficial](#). Añadimos los ítems tal y como nos indican:

Items

All hosts / CentOSEnabledZBXSNMPJMXIPMIApplicationsItems 2TriggersGraphsDiscovery rulesWeb scenarios

ItemPreprocessing

* Namessh_status

TypeZabbix agent

* Keynet.tcp.service[ssh,,22022]Select

* Host interface192.168.56.110 : 10050

Type of informationNumeric (unsigned)

Units

* Update interval1m

Custom intervals

Type	Interval	Period	Action
Flexible	Scheduling	50s	1-7,00:00-24:00

AddRemove

* History storage period

Do not keep historyStorage period90d

* Trend storage period

Do not keep trendsStorage period365d

Show value

As is

show value mappings

New application

Applications

-None-

Items

[All hosts](#) / [CentOS](#) [Enabled](#) [ZBX](#) [SNMP](#) [JMX](#) [IPMI](#) [Applications](#) [Items 2](#) [Triggers](#) [Graphs](#) [Discovery rules](#) [Web scenarios](#)

[Item](#) [Preprocessing](#)

* Name

http_status

Type

Zabbix agent

* Key

net.tcp.service[http,,]

Select

* Host interface

192.168.56.110 : 10050

Type of information

Numeric (unsigned)

Units

* Update interval

1m

Custom intervals

Type	Interval	Period	Action
Flexible	Scheduling	50s	1-7,00:00-24:00
			Remove
Add			

* History storage period

Do not keep history

Storage period

90d

* Trend storage period

Do not keep trends

Storage period

365d

Show value

As is

[show value mappings](#)

New application

Applications

-None-

Populates host inventory field

-None-

Description

Para elegir el parámetro **key**, es recomendable leer la documentación de [key agent](#). En la [documentación oficial](#) también encontramos como monitorizar estos agentes recién añadidos.

Si vamos a **Monitoring > Latest Data** y seleccionamos el **ssh_status**, podemos ver la gráfica que monitoriza el servicio. Si probamos a apagar el servicio y lo reiniciamos, veremos lo siguiente:

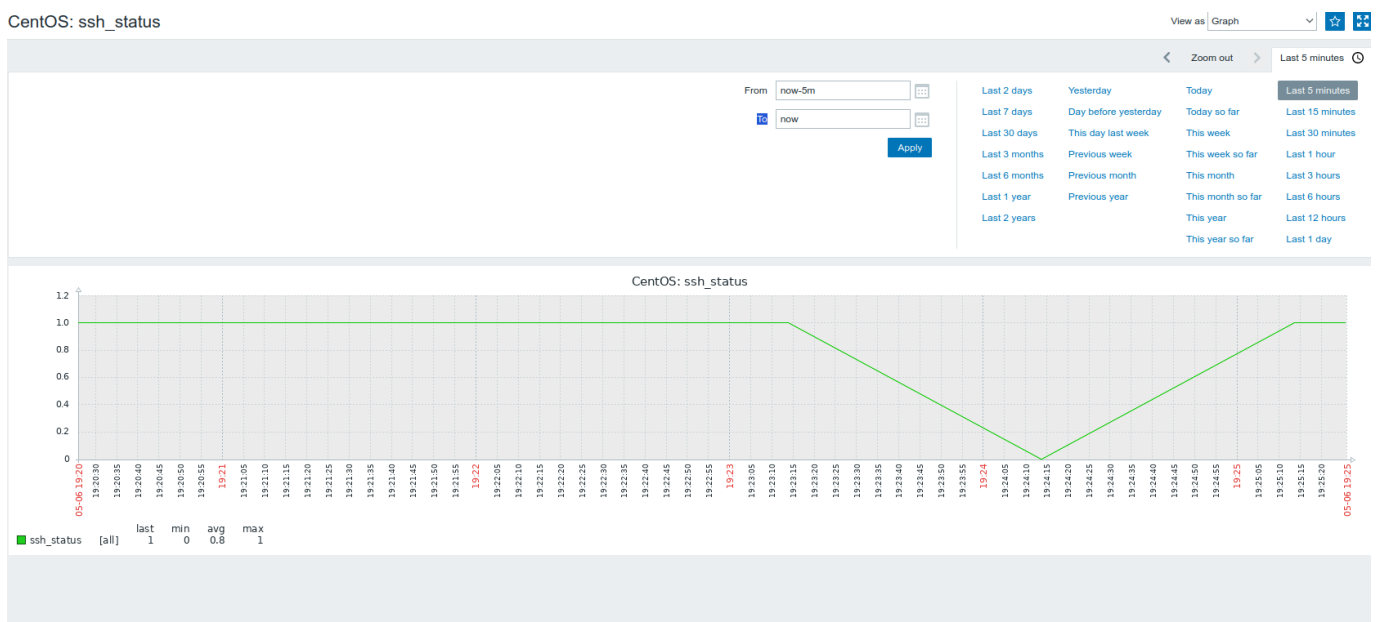
```
CentOSLVM (ssh sin contraseña) [Corriendo] - Oracle VM VirtualBox
Archivo Máquina Ver Entrada Dispositivos Ayuda
# effect here. They will be overridden by command-line options passed on

[danielav@localhost ~]$ sudo systemctl stop sshd
[danielav@localhost ~]$ sudo systemctl status sshd
• sshd.service - OpenSSH server daemon
  Loaded: loaded (/usr/lib/systemd/system/ssh.service; enabled; vendor preset: enabled)
  Active: inactive (dead) since Fri 2022-05-06 13:23:45 EDT; 37s ago
  Docs: man:sshd(8)
        man:sshd_config(5)
  Process: 893 ExecStart=/usr/sbin/sshd -D $OPTIONS $CRYPTO_POLICY (code=exited, status=0/SUCCESS)
  Main PID: 893 (code=exited, status=0/SUCCESS)

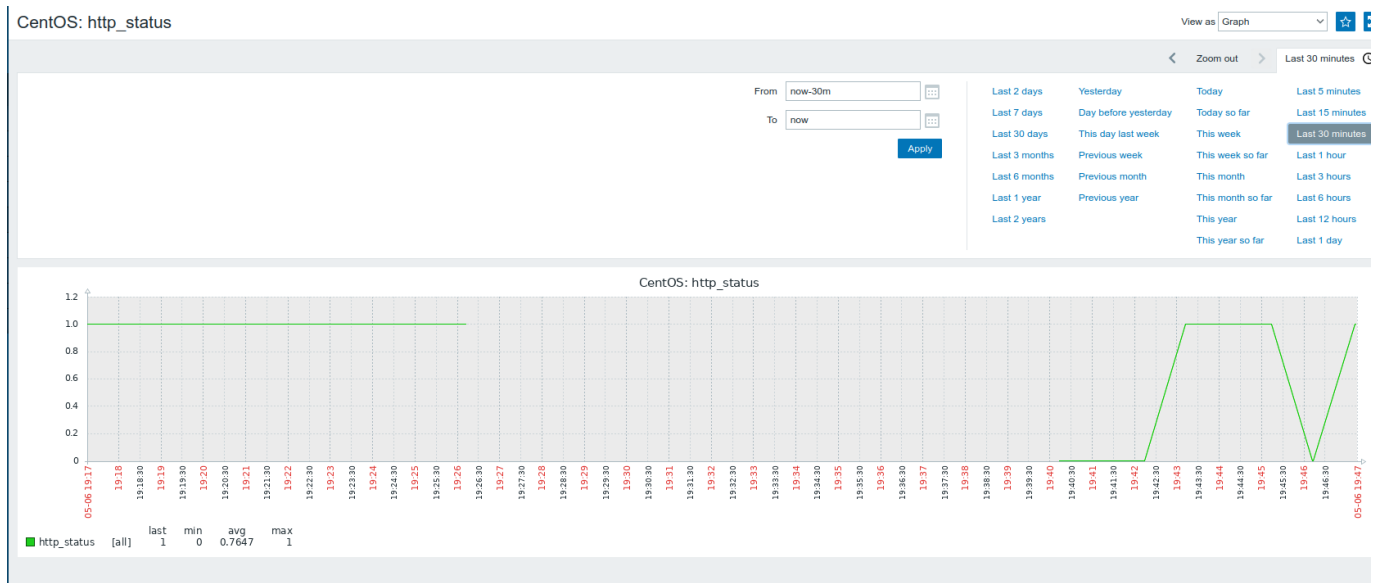
may 06 13:03:56 localhost.localdomain systemd[1]: Starting OpenSSH server daemon...
may 06 13:03:56 localhost.localdomain sshd[893]: Server listening on 0.0.0.0 port 22022.
may 06 13:03:56 localhost.localdomain sshd[893]: Server listening on :: port 22022.
may 06 13:03:56 localhost.localdomain systemd[1]: Started OpenSSH server daemon.
may 06 13:23:45 localhost.localdomain systemd[1]: Stopping OpenSSH server daemon...
may 06 13:23:45 localhost.localdomain systemd[1]: Stopped OpenSSH server daemon.
[danielav@localhost ~]$ sudo systemctl start sshd
[danielav@localhost ~]$ sudo systemctl status sshd
• sshd.service - OpenSSH server daemon
  Loaded: loaded (/usr/lib/systemd/system/ssh.service; enabled; vendor preset: enabled)
  Active: active (running) since Fri 2022-05-06 13:24:42 EDT; 8s ago
  Docs: man:sshd(8)
        man:sshd_config(5)
  Main PID: 1928 (sshd)
  Tasks: 1 (limit: 5019)
  Memory: 1.1M
  CGroup: /system.slice/ssh.service
          └─1928 /usr/sbin/sshd -D -oCiphers=aes256-gcm@openssh.com,chacha20-poly1305@openssh.com,

may 06 13:24:42 localhost.localdomain systemd[1]: Starting OpenSSH server daemon...
may 06 13:24:42 localhost.localdomain sshd[1928]: Server listening on 0.0.0.0 port 22022.
may 06 13:24:42 localhost.localdomain sshd[1928]: Server listening on :: port 22022.
may 06 13:24:42 localhost.localdomain systemd[1]: Started OpenSSH server daemon.
lines 1-15/15 (END)
```

CentOS: ssh_status



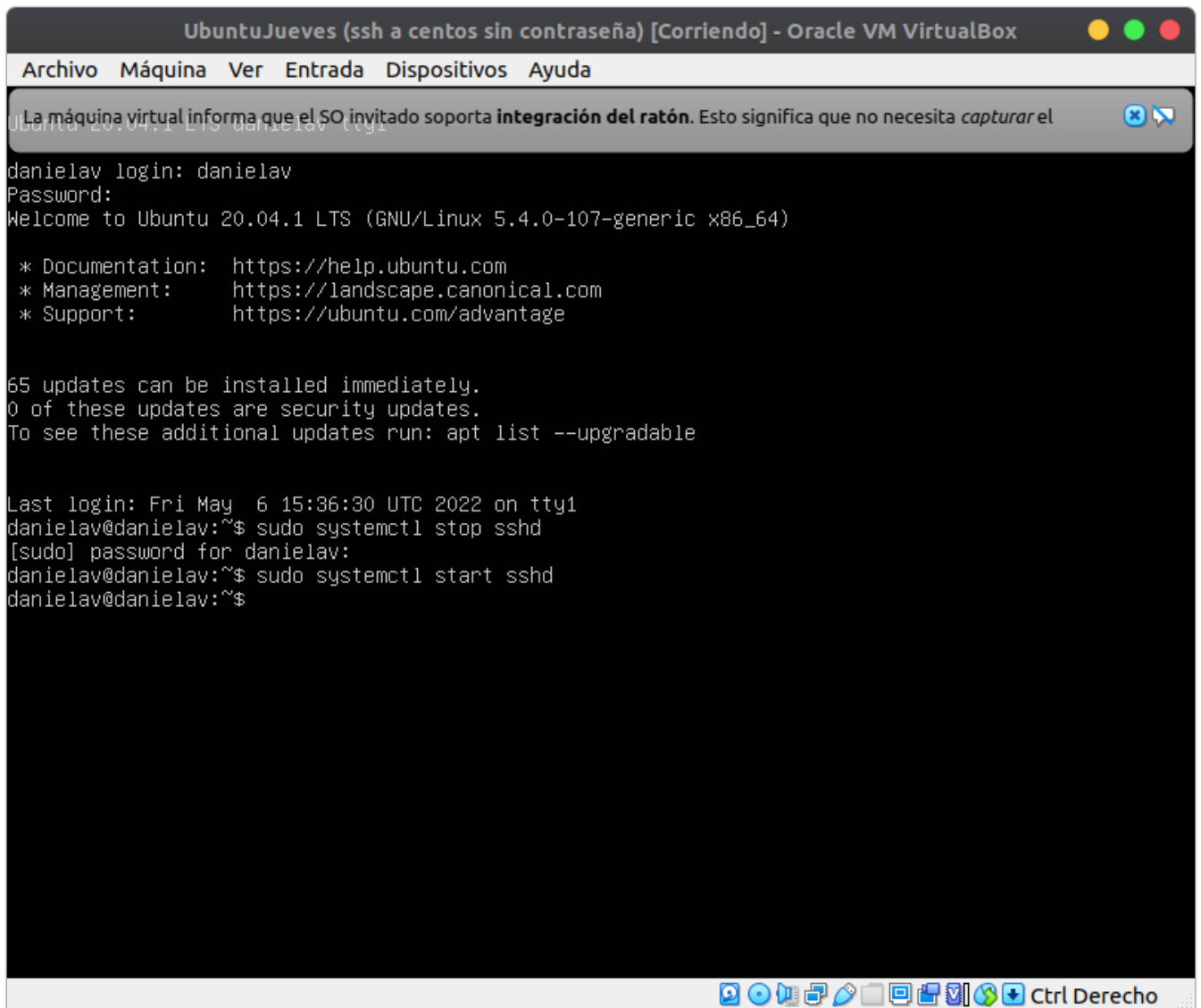
Análogamente con httpd :



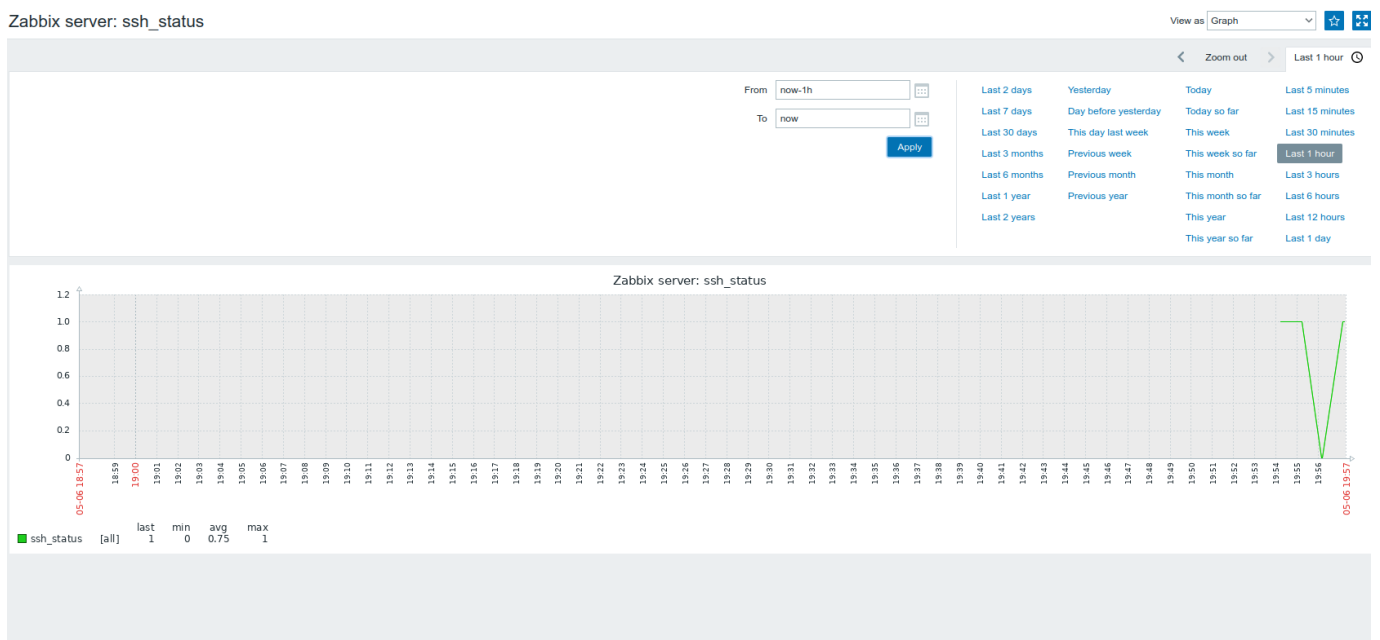
Se puede observar como apagué la máquina de CentOS unos instantes y se corta la monitorización.

Vamos ahora con la monitorización de Ubuntu. Para ello, vamos al apartado de **Configuración > Hosts** nuevamente. Vemos que existe un host llamado **Zabbix Server**. Ahí, añadimos los items tal y como lo hicimos para CentOS (No añadido capturas ya que es análogo).

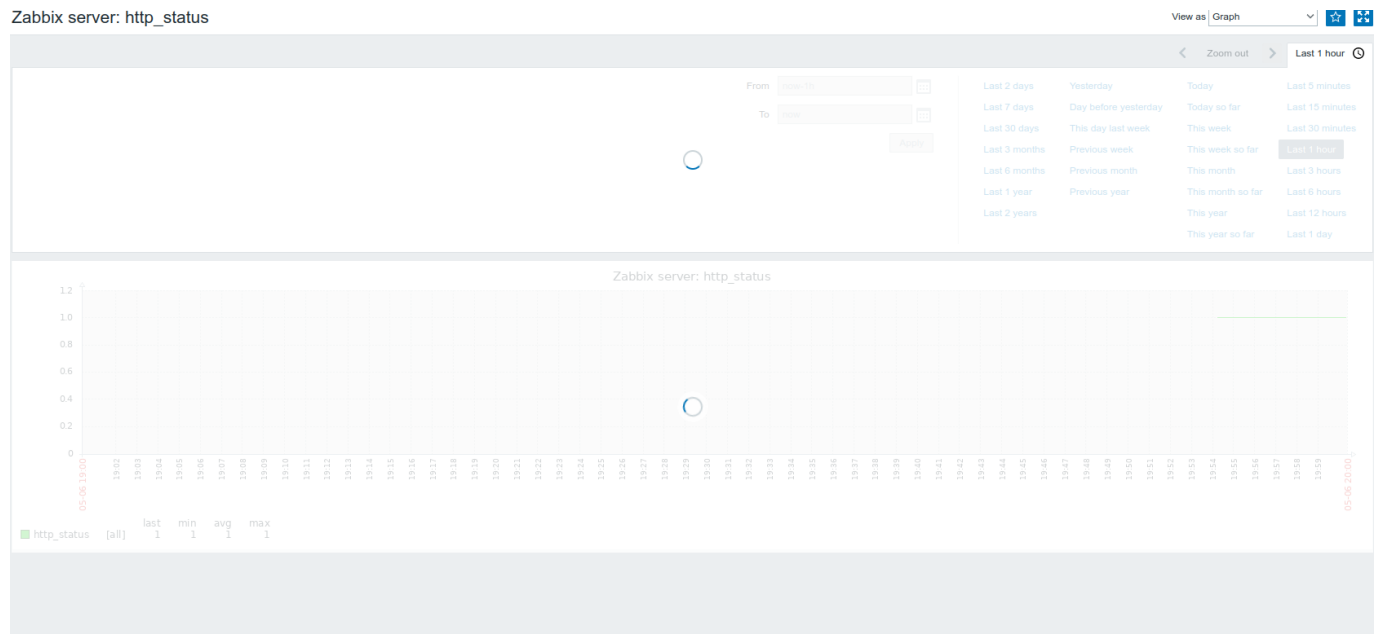
Si vamos a **Monitoring > Latest Data** y seleccionamos el **ssh_status**, pero, esta vez, el de **Zabbix server**, vemos que ocurre exáctamente lo mismo:



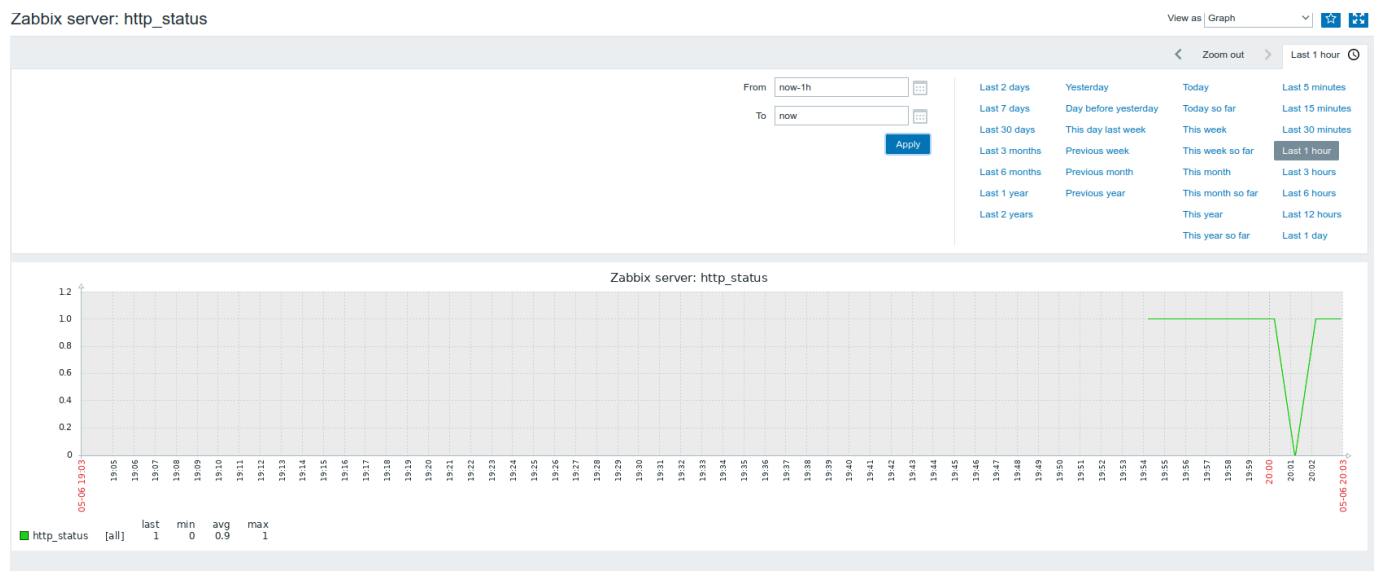
Zabbix server: ssh_status



Pero, si ahora lo hacemos con el servicio `http` , para lo que ejecutamos `sudo systemctl stop apache2` , vemos que, evidentemente, el frontend no recarga, ya que hemos parado el servicio de apache:



Al reiniciar el servicio, ya procesa el pico:



Ejercicio 2: Ansible

Ahora vamos a instalar un programa para poder, por ejemplo, ejecutar la script de la última práctica en múltiples ordenadores.

```
import re
f = open('/proc/mdstat')
for line in f:
    b = re.findall('\[[U]*[_]+[U]*\]', line)
    if(b!=[]):
        print("--ERROR en RAID--")
print("--OK Script--")
```

Esta script la teníamos en Ubuntu.

Para ello vamos a instalar **Ansible** `sudo apt install ansible`. Vamos a su fichero de configuración, `sudo nano /etc/ansible/hosts`. En este fichero, añadimos las IP de nuestros ordenadores, que en este caso son la de Ubuntu y la de CentOS.

Con las dos máquinas encendidas, y comprobando que tenemos conexión ssh de una a otra realizamos:

```
ansible all -m ping -u danielav
# Manda un ping por ssh a todas las máquinas que hayamos especificado
```

Va a dar error, porque como se conecta por ssh, por defecto busca el puerto 22, por lo que vamos a configurar el fichero de configuración e indicarle que busque el puerto 22022.

```
sudo nano /etc/ansible/ansible.cfg
# Buscamos la línea remote_port, la descomentamos e indicamos el puerto 22022
```

Si volvemos a intentarlo va a funcionar en CentOS, pero no en Ubuntu. Esto se debe a que en CentOS se conecta automáticamente por keygen (consecuencia de las prácticas anteriores), pero en Ubuntu requiere la clave, por lo que vamos a especificarle la conexión con su propia clave pública:

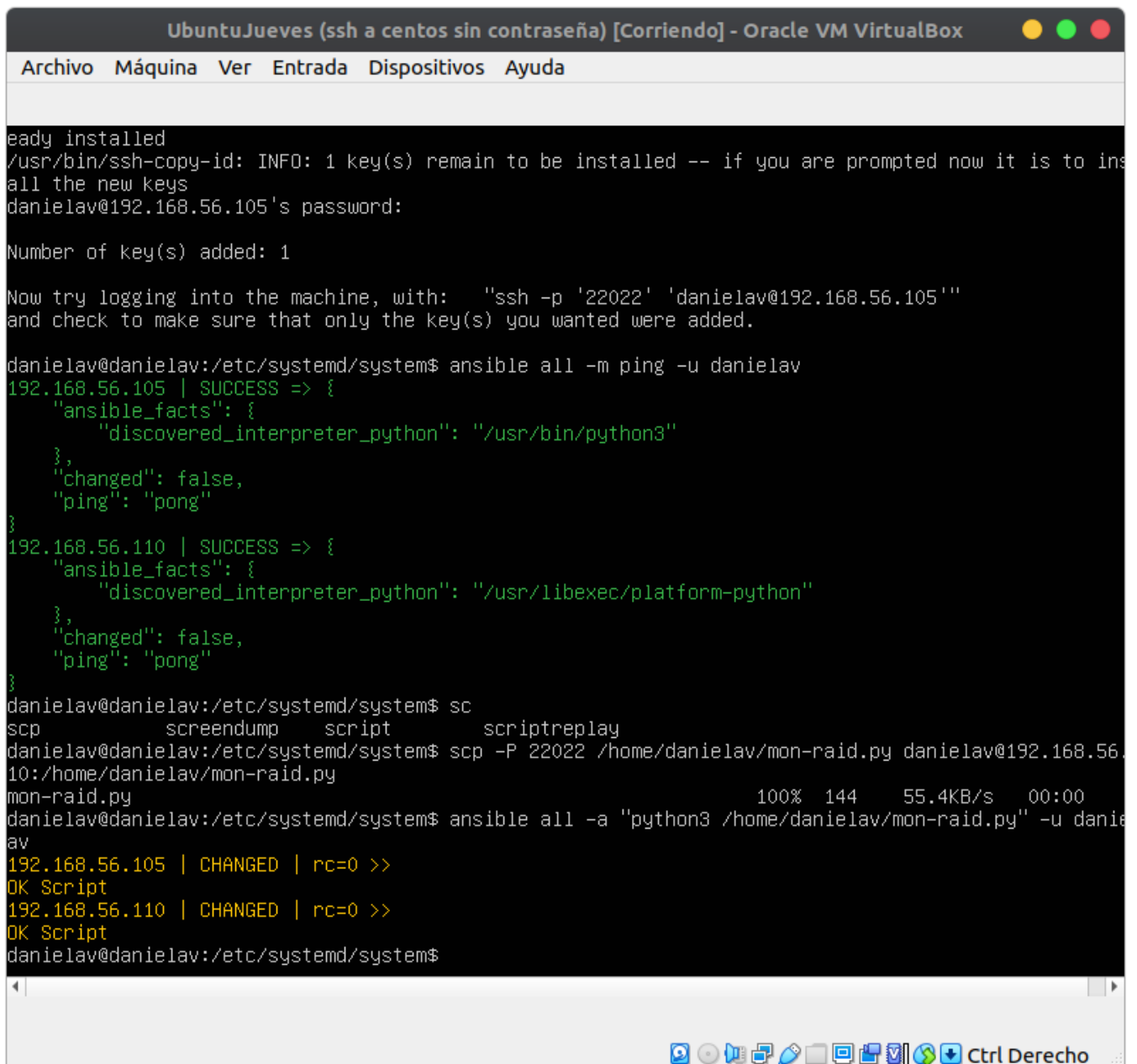
```
ssh-copy-id danielav@192.168.56.105 -p 22022
```

Si no nos dejase, deberíamos ir al fichero de configuración de ssh y activar que puedas conectarte por contraseña:

```
sudo nano /etc/ssh/sshd_config
# Descomentamos PasswordAuthentication y ponemos yes y reiniciamos el sistema
sudo systemctl restart sshd
# Si sale que esta masked puede que se halla jodido y hay que reinstalarlo
sudo apt install openssh-server
```

ansible all -a "python3 /home/danielav/mon-raid.py" -u danielav En una funciona y la otra no, porque en CentOS no esta el fichero, por lo que lo copiamos en CentOS con `scp -P 22022 /home/danielav/mon-raid.py danielav@192.168.56.110:/home/danielav/mon-raid.py`

Si todo ha funcionado correctamente, debe salir esto:



```
UbuntuJueves (ssh a centos sin contraseña) [Corriendo] - Oracle VM VirtualBox
Archivo  Máquina  Ver  Entrada  Dispositivos  Ayuda

easily installed
/usr/bin/ssh-copy-id: INFO: 1 key(s) remain to be installed -- if you are prompted now it is to ins
all the new keys
danielav@192.168.56.105's password:

Number of key(s) added: 1

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

danielav@danielav:/etc/systemd/system$ ansible all -m ping -u danielav
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"
}
danielav@danielav:/etc/systemd/system$ sc
scp      screendump    script      scriptreplay
danielav@danielav:/etc/systemd/system$ scp -P 22022 /home/danielav/mon-raid.py danielav@192.168.56.
10:/home/danielav/mon-raid.py
mon-raid.py                                     100% 144   55.4KB/s   00:00
danielav@danielav:/etc/systemd/system$ ansible all -a "python3 /home/danielav/mon-raid.py" -u daniel
av
192.168.56.105 | CHANGED | rc=0 >>
OK Script
192.168.56.110 | CHANGED | rc=0 >>
OK Script
danielav@danielav:/etc/systemd/system$
```