

Ejercicio 1

Lo primero que vamos a hacer es aprovechar que ya tenemos un LAMP para instalarlo en Ubuntu. Para ellos primero nos descargamos el repositorio con `wget`

`https://repo.zabbix.com/zabbix/5.0/ubuntu/pool/main/z/zabbix-release/zabbix-release_5.0-1+focal_all.deb` . Ahora instalamos los paquetes haciendo `dpkg -i zabbix-release_5.0-1+focal_all.deb` y por ultimo actualizamos con `apt update` .

Ahora necesitamos instalar el frontend, el server y los agentes con el comando `apt install zabbix-server-mysql zabbix-frontend-php zabbix-apache-conf zabbix-agent` .

Vamos a iniciar una consola mysql para root con el comando `mysql -uroot -p` y hacemos lo siguiente:

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

Ahora toca modificar los ficheros `/etc/zabbix/zabbix_server.conf` y `/etc/zabbix/apache.conf`. En el primero de ellos cambiamos la línea `DBPassword=password` y en el segundo `php_value date.timezone Europe/London` .

Vamos ahora a reiniciar los servicios con `systemctl restart zabbix-server zabbix-agent apache2` y `systemctl enable zabbix-server zabbix-agent apache2` .

Ya podemos entrar desde un navegador externo a la dirección <http://192.168.105/zabbix> y estremos en una pantalla como la siguiente:



Welcome

Check of pre-requisites

Configure DB connection

Zabbix server details

Pre-installation summary

Install

Welcome to

Zabbix 5.0

Back

Next step

Recordamos que por defecto el puerto es 10050, el usuario Admin y la contraseña zabbix.

Ahora vamos a proceder a monitorizar Ubuntu:

Para ello nos vamos a la pestaña de Hosts y en el que se llama Zabbix Server creamos dos nuevos ítems como el siguiente que monitorice el servicio ssh:

* Name

Type

* Key

* Host interface

Type of information

Units

* Update interval

Custom intervals

Type	Interval	Period	Action
<input checked="" type="checkbox"/> Flexible <input type="checkbox"/> Scheduling	<input type="text" value="50s"/>	<input type="text" value="1-7,00:00-24:00"/>	Remove
Add			

* History storage period

* Trend storage period

Show value [show value mappings](#)

New application

Applications

-None-

Y otro para el servicio de http:

* Name

Type

* Key

* Host interface

Type of information

Units

* Update interval

Custom intervals

Type	Interval	Period	Action
<input checked="" type="checkbox"/> Flexible <input type="checkbox"/> Scheduling	<input type="text" value="50s"/>	<input type="text" value="1-7,00:00-24:00"/>	<input type="button" value="Remove"/>
<input type="button" value="Add"/>			

* History storage period

* Trend storage period

El resto de opciones no las tocamos.

Con esto ya podemos ir a la pestaña Monitoring -> Latest Data y seleccionamos uno de los dos servicios que queremos monitorizar.

Vamos ahora a la parte de CentOS. Como siempre, esta parte incluye más complicaciones. Para empezar necesitamos instalar el agente:

Para ello modificamos el fichero `/etc/selinux/config`. Cambiamos la línea `SELINUX=disable`. Con esto hacemos los comandos `sudo yum install https://repo.zabbix.com/zabbix/5.0/rhel/8/x86_64/zabbix-release-5.0-1.el8.noarch.rpm` y `sudo yum install zabbix-agent`.

Ahora modificamos el fichero `/etc/zabbix/zabbix_agentd.conf` añadiendo lo siguiente:

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

Con esto ahora vamos a desactivar el firewall como siempre con `sudo firewall-cmd --permanent --add-port=10050/tcp` y lo recargamos con `sudo firewall-cmd --reload`. Ahora iniciamos el servicio con `sudo systemctl enable zabbix-agent`, `sudo systemctl start zabbix-agent` y `service zabbix-agent start`.

Ahora nos vamos a Zabbix y añadimos un nuevo host:

* Host name

Visible name

* Groups Linux servers Zabbix servers Select
type here to search

* Interfaces	Type	IP address	DNS name	Connect to	Port	Default
Agent	<input type="text" value="192.168.56.110"/>	<input type="text"/>	IP DNS	<input type="text" value="10050"/>	<input checked="" type="radio"/> Remove	

[Add](#)

Description

Monitored by proxy (no proxy)

Enabled ☒

Add Cancel

Ahora ya sólo tendríamos que crear los items anteriores pero para este nuevo host.

Ejercicio 2

Empezamos por instalar ansible con `sudo apt install ansible` en Ubuntu.

Vamos a añadir nuestras ips al fichero de hosts haciendo `sudo nano /etc/ansible/hosts` y en las ips que aparecen al principio añadimos la 105 y la 110 de Ubuntu y CentOS respectivamente.

```
ubuntu 20 (Antes de LAMP) [Running]
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
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:
#www[001:006].example.com
#
# Ex 3: A collection of database servers in the 'dbservers' group
```

Get Help	Write Out	Where Is	Cut Text	Justify	Cur Pos	M-U Undo
Exit	Read File	Replace	Paste Text	To Spell	Go To Line	M-E Redo

Vamos a iniciar CentOS y desde Ubuntu hacemos `ansible all -m ping -u alejandorm`.

```
ubuntu 20 (Antes de LAMP) [Running]
alejandrorm@serverdeale:~$ ansible all -m ping -u alejandrorm
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
}
alejandrorm@serverdeale:~$
```

El error que sale se debe a que se está intentando conectar a ssh por el camino predeterminado, el puerto 22, pero nosotros lo cambiamos para que fuera por el puerto 22022. Hacemos `sudo nano /etc/ansible/ansible.cfg` y descomentamos `remote_port` y añadimos 22022.

```
ubuntu 20 (Antes de LAMP) [Running]
GNU nano 4.8 /etc/ansible/ansible.cfg Modified
# 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]

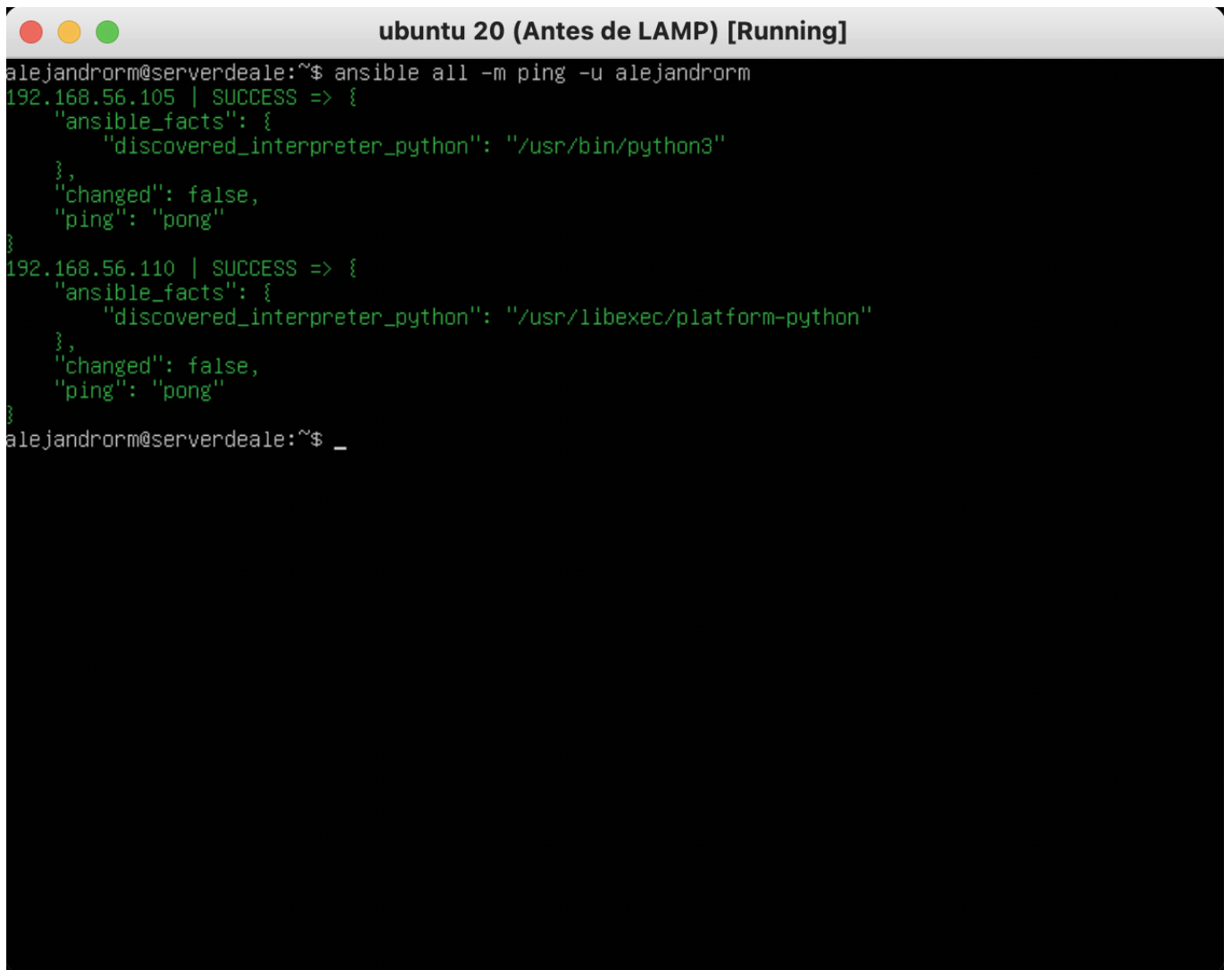
# some basic default values...

#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

# plays will gather facts by default, which contain information about
# the remote system.
#
# smart - gather by default, but don't regather if already gathered

^G Get Help  ^O Write Out  ^W Where Is  ^K Cut Text   ^J Justify    ^C Cur Pos    M-U Undo
^X Exit      ^R Read File  ^_ Replace   ^U Paste Text ^T To Spell   ^_ Go To Line M-E Redo
```

Si utilizamos otra vez `ansible all -m ping -u alejandorm` ya funciona:

A terminal window with a title bar containing three colored circles (red, yellow, green) and the text "ubuntu 20 (Antes de LAMP) [Running]". The terminal shows a user named "alejandrorm" at a host named "serverdeale" running the command "ansible all -m ping -u alejandrorm". The output shows two hosts: "192.168.56.105" and "192.168.56.110", both with a "SUCCESS" status. The output for each host includes "ansible_facts" with "discovered_interpreter_python" and "ping" results. The prompt "alejandrorm@serverdeale:~\$" is followed by an underscore "_".

```
alejandrorm@serverdeale:~$ ansible all -m ping -u alejandrorm
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"
}
alejandrorm@serverdeale:~$ _
```

Ahora vamos a hacer que se ejecute el script que nos informa del estado de nuestro RAID. Empezamos probándolo con `ansible all -a "python3 /home/alejandrorm/mpon-raid.py" -u alejandrorm`.

```
ubuntu 20 (Antes de LAMP) [Running]
alejandrorm@serverdeale:~$ ansible all -a "python3 /home/alejandrorm/mon-raid.py" -u alejandrorm
192.168.56.105 | CHANGED | rc=0 >>
--OK Script--
192.168.56.110 | FAILED | rc=2 >>
python3: can't open file '/home/alejandrorm/mon-raid.py': [Errno 2] No such file or directorynon-zero
return code
alejandrorm@serverdeale:~$ [ 3559.414608] e1000 0000:00:03:0 enp0s3: Reset adapter
```

Obtenemos un fallo porque no hemos añadido el script en CentOS entonces no tiene qué ejecutar. Con `scp -P 22022 /home/alejandrorm/mon-raid.py alejandrorm@192.168.56.110:/home/alejandrorm/mon-raid.py` debería bastar para pasarlo por ssh.

ubuntu 20 (Antes de LAMP) [Running]

```
alejandrorm@serverdeale:~$ scp -P 22022 /home/alejandrorm/mon-raid.py alejandrorm@192.168.56.110:/home/alejandrorm/mon-raid.py
100% 152 174.2KB/s 00:00
alejandrorm@serverdeale:~$ ansible all -a "python3 /home/alejandrorm/mon-raid.py" -u alejandrorm
192.168.56.110 | CHANGED | rc=0 >>
--OK Script--
192.168.56.105 | CHANGED | rc=0 >>
--OK Script--
alejandrorm@serverdeale:~$
```