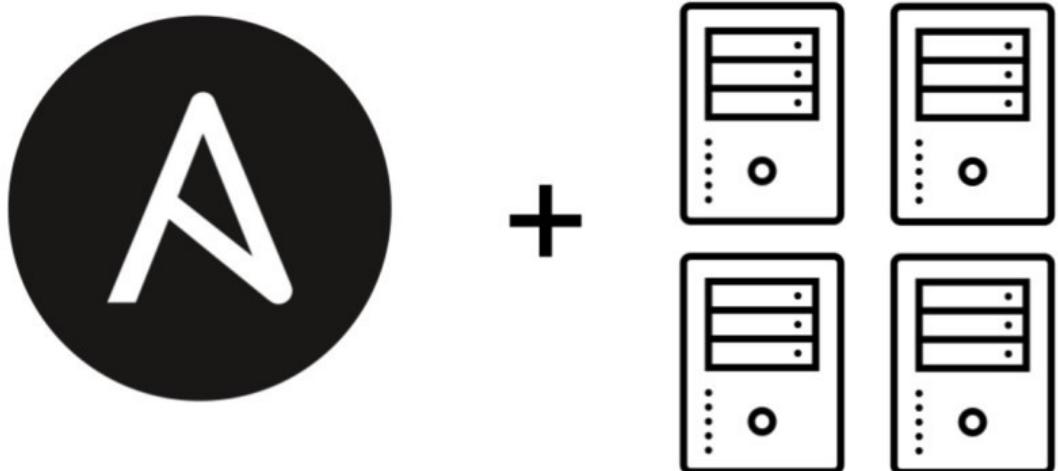


PROYECTO INTEGRADO

SERVIDOR DE SEGURIDAD Y CONTROL DE TRÁFICO



JESÚS ZAMORA JIMÉNEZ
2º ASIR

ÍNDICE

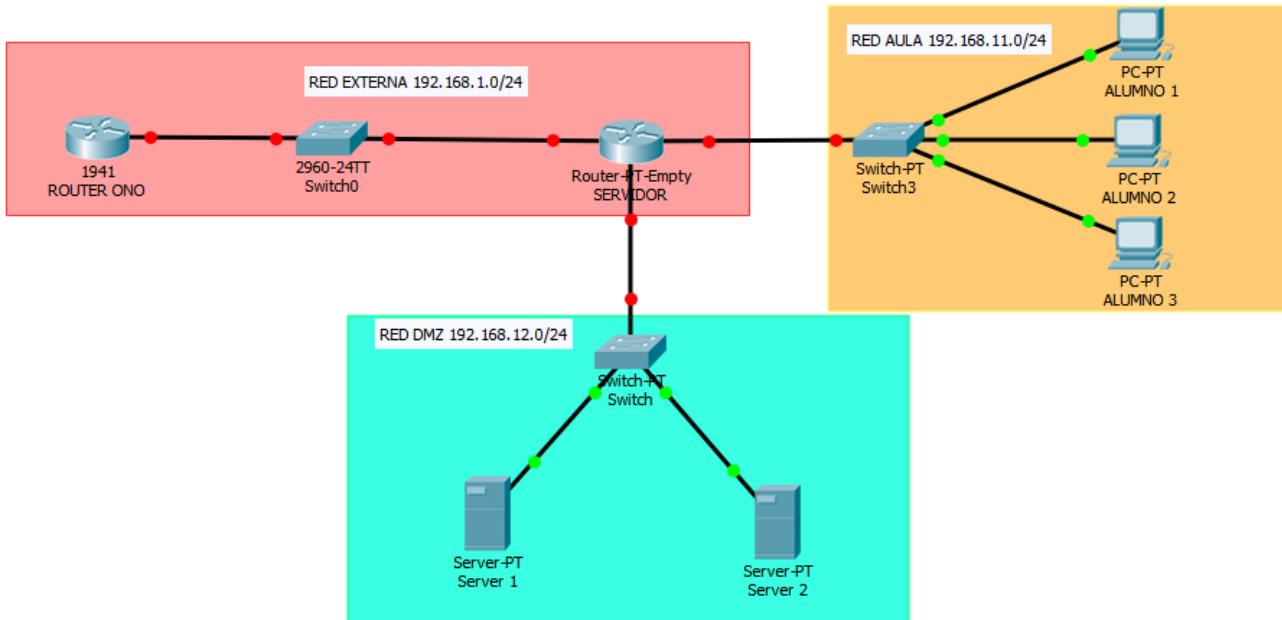
| | | |
|-----|-------------------------------------|-----------|
| 1 | Introducción..... | Página 1 |
| 1.1 | Programación inicial planteada..... | Página 3 |
| 1.2 | Planificación real ejecutada..... | Página 3 |
| 2 | Preparación..... | Página 5 |
| 2.1 | Instalación sistema operativo..... | Página 5 |
| 2.2 | Ansible..... | Página 8 |
| 2.3 | Clave publica y SSH..... | Página 9 |
| 3 | Proxy no transparente..... | Página 13 |
| 3.1 | Listas negras..... | Página 17 |
| 3.2 | Webmin..... | Página 19 |
| 3.3 | Interfaz amigable..... | Página 24 |
| 4 | Dansguardian..... | Página 28 |
| 5 | Reportes de usuarios..... | Página 31 |
| 5.1 | Sarg en Cron..... | Página 33 |
| 6 | Implementación firewall..... | Página 34 |
| 6.1 | Iptables..... | Página 35 |
| 6.2 | Script Iptables en el inicio..... | Página 39 |
| 7 | Servidor SSH..... | Página 40 |
| 8 | Configuración de red..... | Página 43 |
| 9 | Respaldo mediante Ansible..... | Página 45 |
| 10 | Autoformación..... | Página 49 |
| 11 | Bibliografía..... | Página 50 |

1. Introducción

Este proyecto llamado *Servidor de seguridad y control de tráfico* pretende dar diferentes servicios a una red ubicada en un aula de informática mediante un servidor que se situará entre la red interna y la red externa, es decir, entre el router y los diferentes equipos informáticos del aula.

Para ello, se ha empleado un servidor en rack proporcionado por el departamento de informática ubicado en el aula de 2º de ASIR y un ordenador personal para ejecutar la secuencia de paquetes y comandos en Ansible.

Este servidor actuará como enrutador para las diferentes redes. Existirán 3 redes distintas; la red externa, la red del aula y la red del DMZ. En la ilustración se pueden apreciar el esquema de red que será implementado.



El objetivo de este proyecto consiste en implementar seguridad en dicho servidor en rack para que no sea accesible por todos los puertos y aplicaciones, solo por las deseadas, y dar una serie de servicios con DNS, FTP o SSH. Además, el servidor filtrará el tráfico entrante de las diferentes redes permitiendo o no la salida del tráfico en función de la configuración establecida por un proxy no transparente (Squid) y Dansguardian.

Esta configuración permite múltiples opciones en función del origen del tráfico de la red. Si el tráfico proviniera de un cliente de la red del aula se podría ajustar el proxy de manera que cachee, filtre, balancee la carga, registre de eventos o realice autenticación. Con ello se podría conseguir que a un cliente se le permitiera navegar durante un horario establecido o bloquear el tráfico a unas determinadas direcciones IP o direcciones MAC entre otros.

Cada vez que un cliente se conecte al servidor y navegue, el proxy registrará todas las peticiones web provenientes del puerto 80 y 443 (HTTP y HTTPS). Estos eventos serán almacenados en un log y además serán accesibles mediante un interfaz web (Sarg), para obtener un amplio reporte de los sitios webs permitidos y denegados.

El servidor permitirá los servicios FTP, SSH y DNS para la red del aula y la red del DMZ. Esto permite que los clientes conectados al servidor puedan resolver nombres de dominio, realizar transferencias de archivos o iniciar sesión remotamente.

Existirá una herramienta llamada Fail2ban que detectará aquellas direcciones IP cuyo comportamiento resulta inusual, por ejemplo, las que han intentado acceder varias veces con una contraseña incorrecta a los archivos de registro del servidor, y las bloqueará durante un tiempo determinado.

Todo lo anterior será implementado en una distribución de Linux, en este caso Debian versión 8.9. La última versión estable no se ha empleado en este caso ya que algunos nombres de paquetes necesarios y comandos del sistema cambian, y se ha optado por emplear una versión estable anterior a la última.

El proyecto será actualizado en el [repositorio de Github](#) que previamente se ha enviado por correo al tutor del proyecto.

1.1. Programación inicial planteada

Inicialmente, este proyecto se planteó para que fuera instalado en un servidor del aula de 2º, en un servidor en rack usando el sistema operativo Debian 8.9 de forma que implementara un proxy transparente en Linux (Squid + Privoxy) con interfaz amigable para la modificación de reglas (Dansguardians o similar), y que implementase un firewall (Iptables + Fail2ban).

Además, debía implementar un monitor de red (Zabbix) y un servidor SSH para administración remota del servidor.

Tanto el proxy como el firewall tomarían listas negras que se actualizarían de forma periódica.

La instalación del servidor en rack debía realizarse de forma automatizada utilizando la tecnología Ansible, de manera que si el servidor fallase, fuera formateado o cambiado por otro con sistema Debian 8.9, con tan solo ejecutar la maqueta automatizada de Ansible se restaurase fácil y de forma automatizada.

1.2. Planificación real ejecutada

Este proyecto comenzó con la instalación del sistema operativo Debian versión 8.9 en un servidor asignado del aula de 2º, implementando un proxy Squid no transparente versión 3.4.8.

No pudo ser transparente ya que Squid en esa versión necesitaba un certificado generado en el servidor e instalado en los clientes para poder visitar páginas seguras (HTTPS), por lo que tras dos semanas probando e investigando no funcionó. Se optó por que los clientes de la red del aula configuraran de forma manual su navegador web ajustando el proxy por el puerto 3128 o 8080 (Dansguardian)

Se ha elegido Webmin como interfaz amigable, para que así desde un navegador se pudiera ajustar el filtro de Squid. Además se ha implementado otro proxy, Dansguardian, que ofrece una mayor variedad de filtrado.

Al igual que con Squid, Dansguardian no será transparente. Los clientes que quieran navegar por internet necesitarán configurar los navegadores web con los

puertos 3128 o 8080 de la dirección 192.168.1.18.

Para obtener un amplio reporte de lo que se filtra en el servidor, se ha utilizado la herramienta Sarg, ya que ofrece reportes por Ips sobre los usuarios que se conectan al servidor y visitan páginas por los puertos 80 y 443.

También se ha implementado un servidor SSH para administración remota del servidor y una configuración de Iptables para realizar un control del tráfico que pasa por el servidor.

El script de Iptables está configurado de modo que será ejecutado cada vez que se reinicia el equipo.

Mediante Cron, en el servidor se ha programado que Sarg genere informes todos los días por la tarde, y que Squid actualice su lista negra cada semana.

Por último, todo lo anterior está programado en Ansible para que de forma automatizada sea fácil la configuración en caso de fallase el servidor.

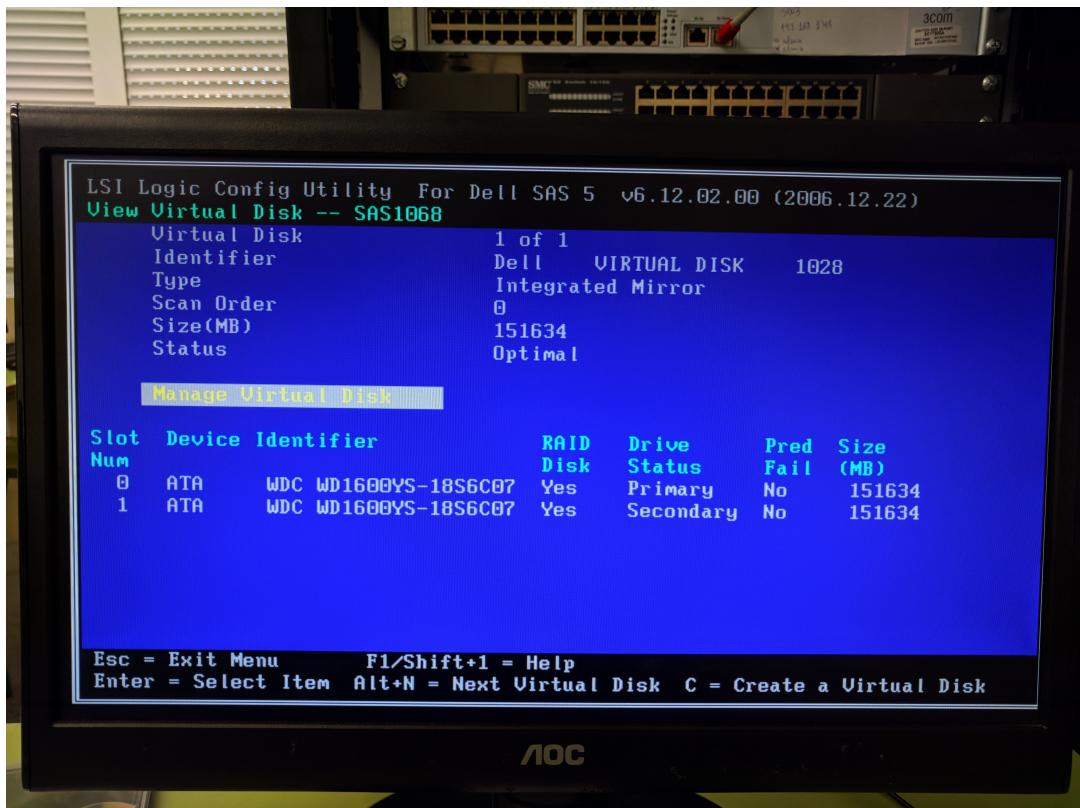
2. Preparación

2.1. Instalación sistema operativo

Para la realización de proyecto se instala Debian versión 8.9 en el servidor rack. El sistema operativo se instala sobre el servidor que tiene 5 puertos Ethernet.

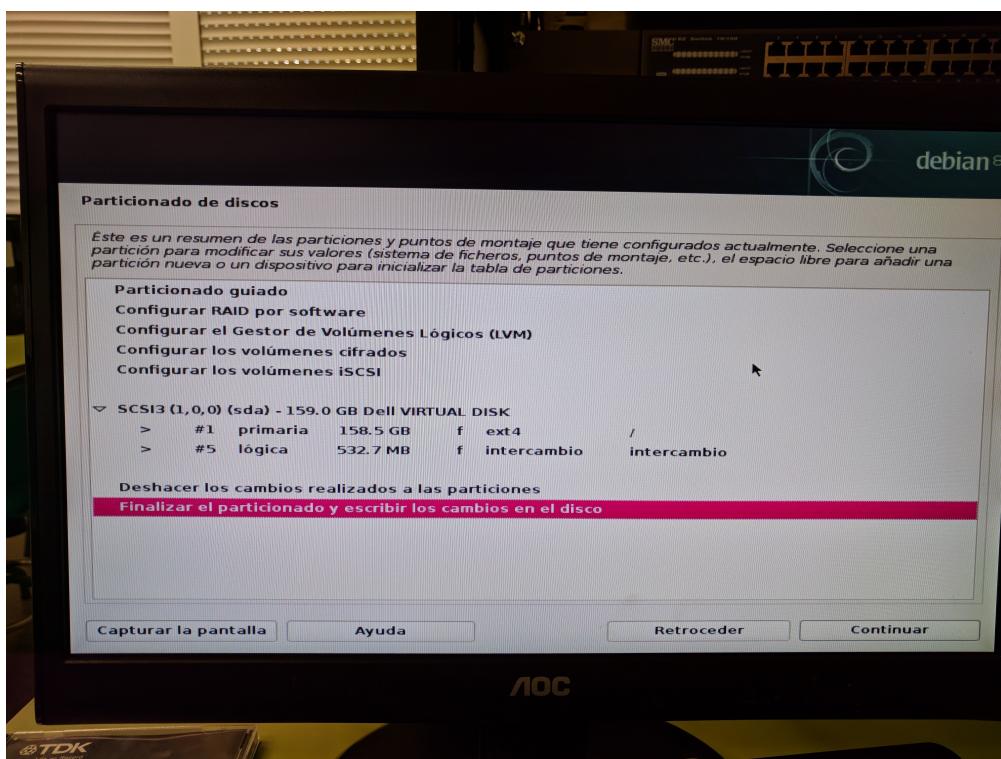
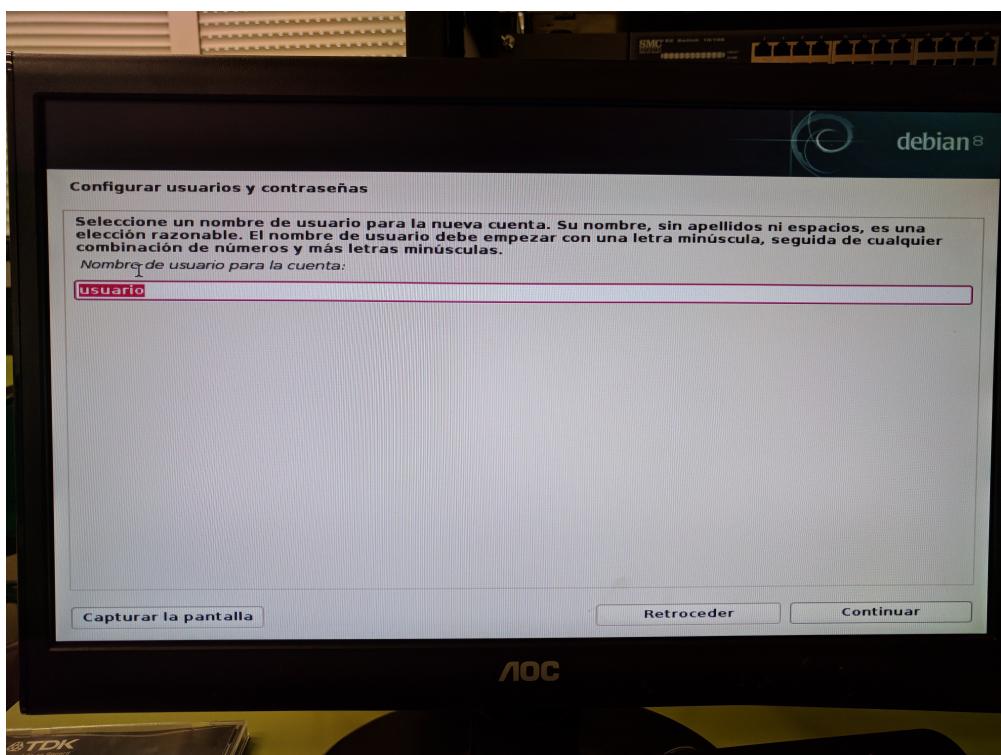


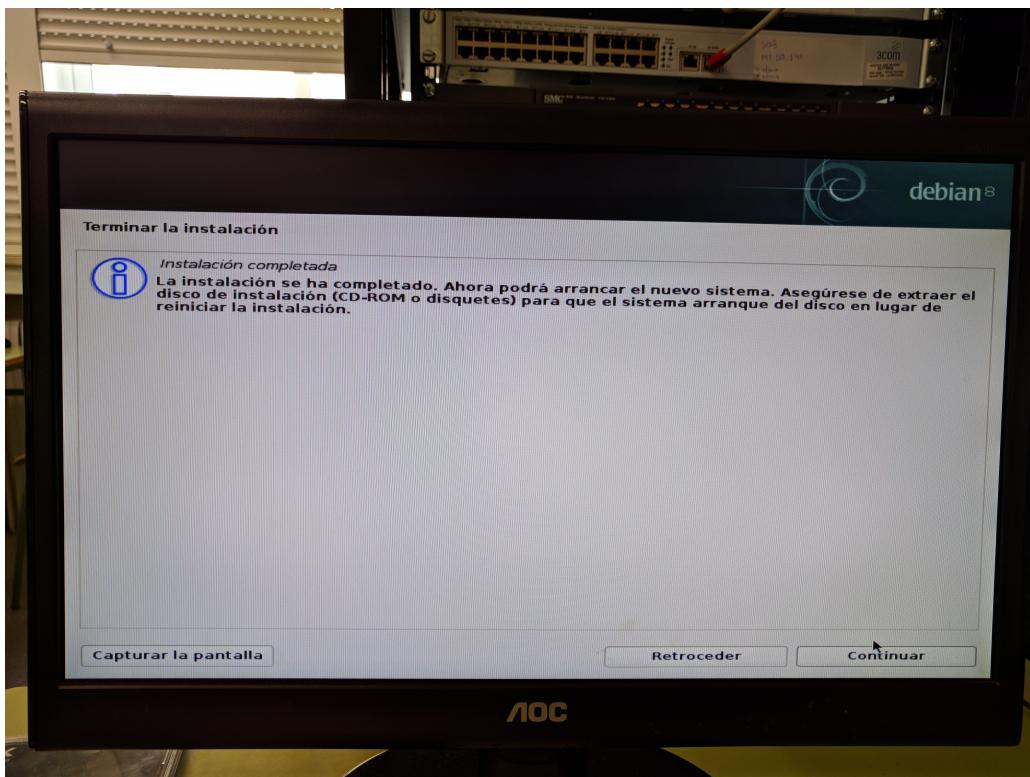
Previamente se configura un raid 0 en dos discos duros mediante la BIOS.



Los datos de los pasos de instalación del sistema operativo son:

- Usuario → usuario
- Contraseña → Tico*65
- Contraseña root → Tico*65
- IP servidor → 192.168.1.18





2.2. Ansible

Para este proyecto se contará con la tecnología Ansible, ya que ofrece una ventaja muy importante; respaldo de toda la instalación y configuración de forma automática y simple.

Ansible es una herramienta *Open-source* desarrollada en *Python* y comercialmente ofrecida por AnsibleWorks que la definen como un motor de orquestación muy simple que automatiza las tareas necesarias.

Permite diferentes formas de configuración, una podría ser mediante un solo fichero llamado *playbook* que contendría todos los parámetros para hacer una determinada tarea sobre un determinado servidor como es el caso, o también un grupo de servidores.

La estructura que sigue ansible es sencilla; se establece en un fichero con extensión *.yml* la definición de todas las tareas que se realizarán sobre un conjunto de hosts. En el fichero *ansible.cfg* se define el nombre del archivo que va a especificar los *hosts*, que a su vez este contiene las direcciones IP de los equipos sobre los cuales recaerán las tareas del archivo con extensión *.yml*.



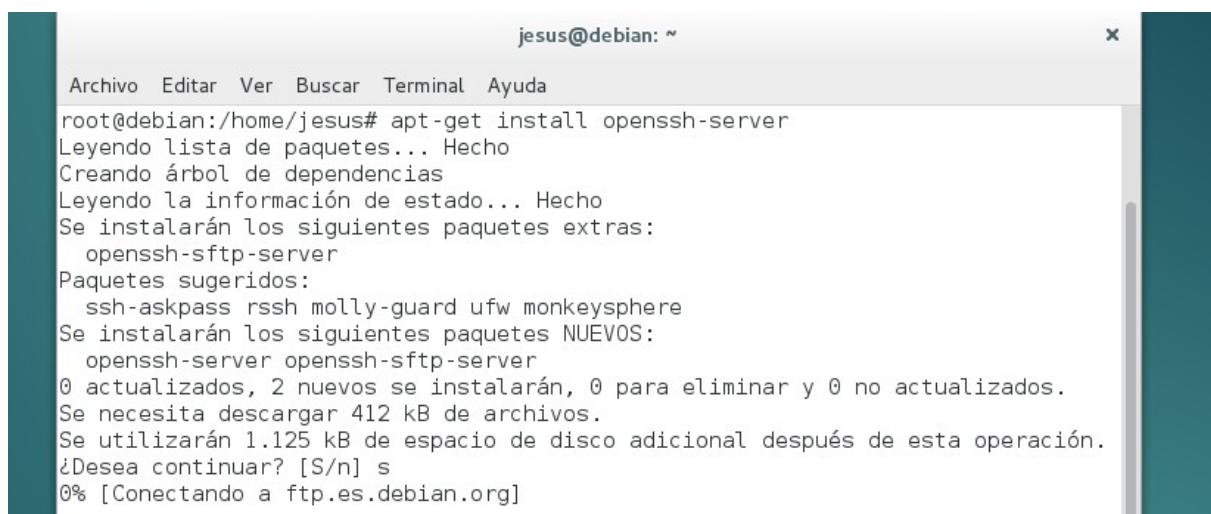
Existirá una carpeta llamada *archive* donde habrá, en este caso, un archivo un paquete de una herramienta que más adelante se utilizará.

En *template* se guardan los archivos que serán plantillas a copiar en el destino para configurar una aplicación, como Squid.

Ansible se ejecuta en un nodo diferente al servidor, el cual se llamará *orquestador*. En *orquestador* se encontrará la maqueta de los comandos y paquetes de Ansible automatizados, llamado *respaldo.yml*, que se ejecutarán en el servidor mediante SSH.

2.3. Clave pública SSH

Una vez finalizado el proceso de instalación del sistema operativo en el servidor, es necesario instalar el paquete SSH, configurar el archivo de configuración de SSH ubicado en /etc/sshd_config editando la línea *PermitRootLogin* para que quede de la siguiente manera y reiniciar el servicio:



```
jesus@debian: ~
Archivo Editar Ver Buscar Terminal Ayuda
root@debian:/home/jesus# apt-get install openssh-server
Leyendo lista de paquetes... Hecho
Creando árbol de dependencias
Leyendo la información de estado... Hecho
Se instalarán los siguientes paquetes extras:
  openssh-sftp-server
Paquetes sugeridos:
  ssh-askpass rssh molly-guard ufw monkeysphere
Se instalarán los siguientes paquetes NUEVOS:
  openssh-server openssh-sftp-server
0 actualizados, 2 nuevos se instalarán, 0 para eliminar y 0 no actualizados.
Se necesita descargar 412 kB de archivos.
Se utilizarán 1.125 kB de espacio de disco adicional después de esta operación.
¿Desea continuar? [S/n] s
0% [Conectando a ftp.es.debian.org]
```

```
# Lifetime and size of ephemeral
Key_regeneration_interval 3600
ServerKeyBits 1024

# Logging
SyslogFacility AUTH
LogLevel INFO

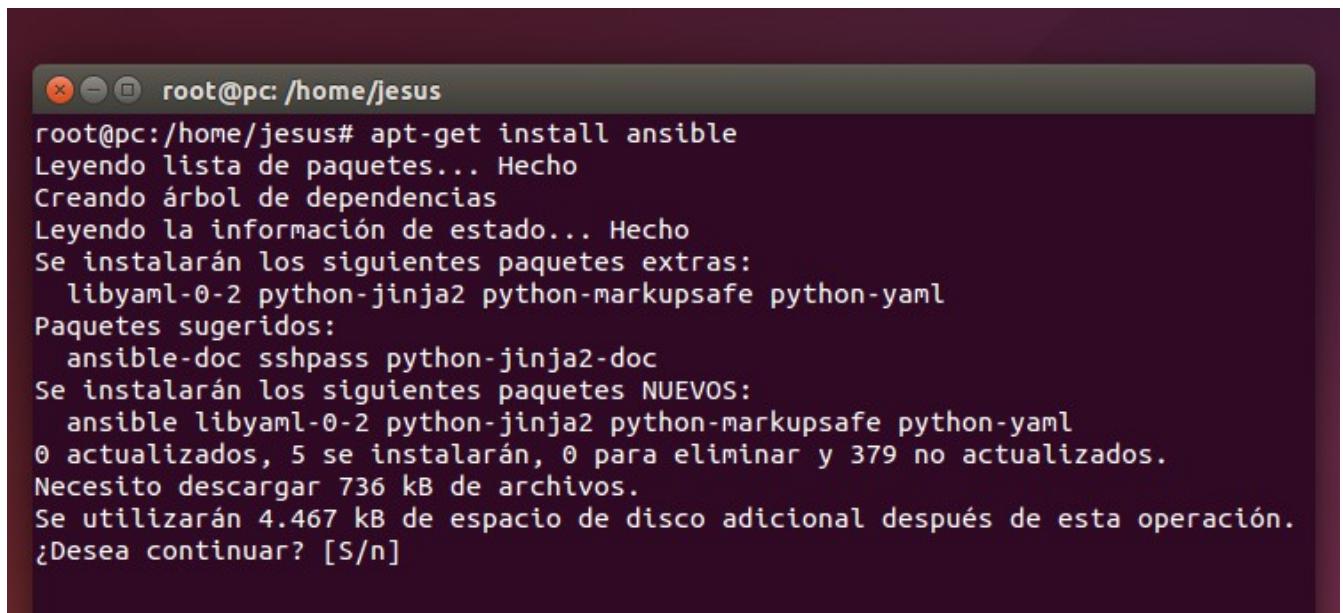
# Authentication:
LoginGraceTime 120
PermitRootLogin yes
StrictModes yes

RSAAuthentication yes
PubkeyAuthentication yes
#AuthorizedKeysFile      %h/.ssh/

# Don't read the user's ~/.rhost
IgnoreRhosts yes
# For this to work you will also
RhostsRSAAuthentication no
# similar for protocol version 2
```

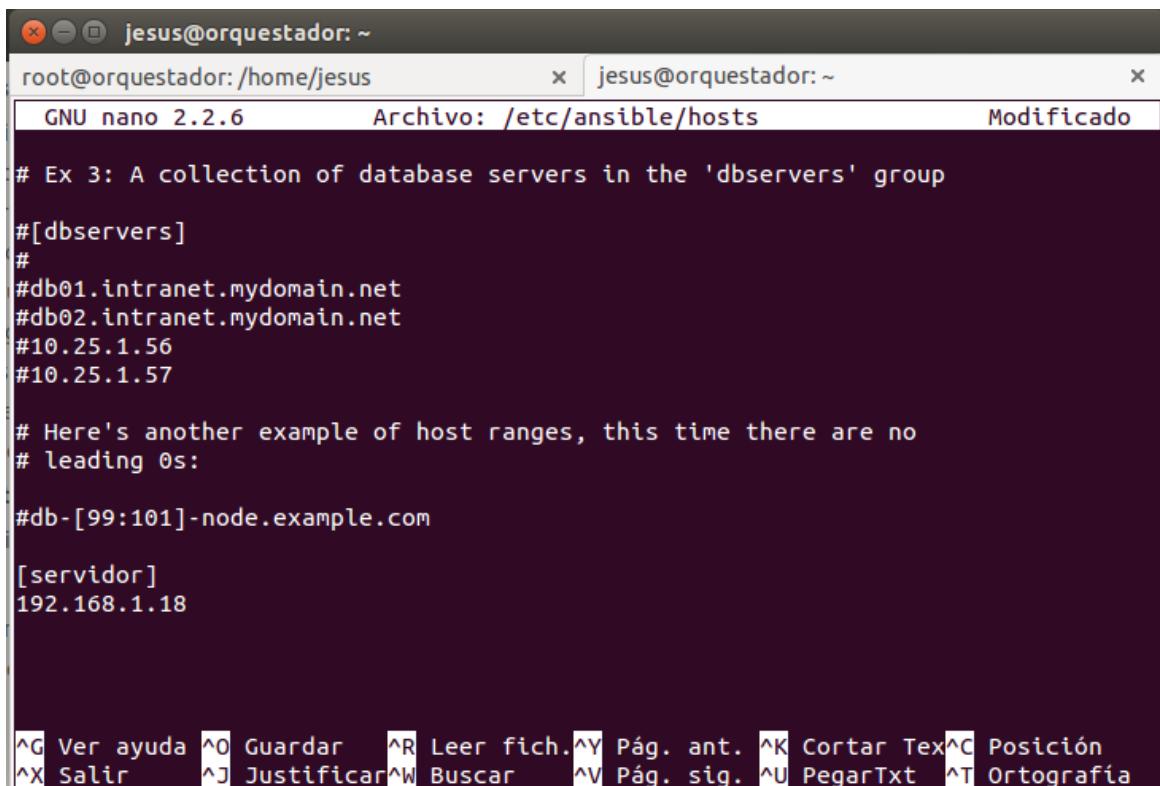
Una vez configurado SSH en el servidor, es necesario autorizar al nodo *orquestador* por SSH enviando desde el nodo *orquestador* la clave pública al servidor.

Lo primero de todo será instalar Ansible desde la terminal del nodo *orquestador*.



```
root@pc:/home/jesus# apt-get install ansible
Leyendo lista de paquetes... Hecho
Creando árbol de dependencias
Leyendo la información de estado... Hecho
Se instalarán los siguientes paquetes extras:
  libyaml-0-2 python-jinja2 python-markupsafe python-yaml
Paquetes sugeridos:
  ansible-doc sshpass python-jinja2-doc
Se instalarán los siguientes paquetes NUEVOS:
  ansible libyaml-0-2 python-jinja2 python-markupsafe python-yaml
0 actualizados, 5 se instalarán, 0 para eliminar y 379 no actualizados.
Necesito descargar 736 kB de archivos.
Se utilizarán 4.467 kB de espacio de disco adicional después de esta operación.
¿Desea continuar? [S/n]
```

Al instalar Ansible, lo primero que se necesita es crear o modificar el fichero *hosts* ubicado en */etc/ansible/hosts* donde se va a indicar el nodo sobre el cual se va a realizar las tareas automatizadas. En este caso se indica la dirección IP de servidor.



```
jesus@orquestador: ~
root@orquestador:/home/jesus          x jesus@orquestador: ~
                                         GNU nano 2.2.6      Archivo: /etc/ansible/hosts      Modificado
                                         # Ex 3: A collection of database servers in the 'dbservers' group
                                         #[dbservers]
                                         #
                                         #db01.intranet.mydomain.net
                                         #db02.intranet.mydomain.net
                                         #10.25.1.56
                                         #10.25.1.57
                                         #
                                         # Here's another example of host ranges, this time there are no
                                         # leading 0s:
                                         #
                                         #db-[99:101]-node.example.com
                                         #
                                         [servidor]
                                         192.168.1.18
```

Es ahora cuando se genera una clave pública desde la terminal con *ssh-keygen*.

```
root@orquestador:/home/jesus
root@orquestador:/home/jesus# ssh-keygen
Generating public/private rsa key pair.
Enter file in which to save the key (/root/.ssh/id_rsa):
/root/.ssh/id_rsa already exists.
Overwrite (y/n)? y
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /root/.ssh/id_rsa.
Your public key has been saved in /root/.ssh/id_rsa.pub.
The key fingerprint is:
56:28:9b:50:3e:33:c0:f3:33:f7:39:31:74:e6:d4:8a root@orquestador
The key's randomart image is:
+--[ RSA 2048]----+
|   . . .
|   o+ ... + .
|   .o* ...* .
|   .+B..E o
|   o+S. +
|   . +
|   .
|   .
+-----+
root@orquestador:/home/jesus#
```

Tras haber creado la clave pública se envía por SSH a servidor. Requerirá introducir la clave el usuario root del servidor. Previamente se ha modificado el fichero de configuración de SSH del servidor para que en este paso permita la autenticación de root.

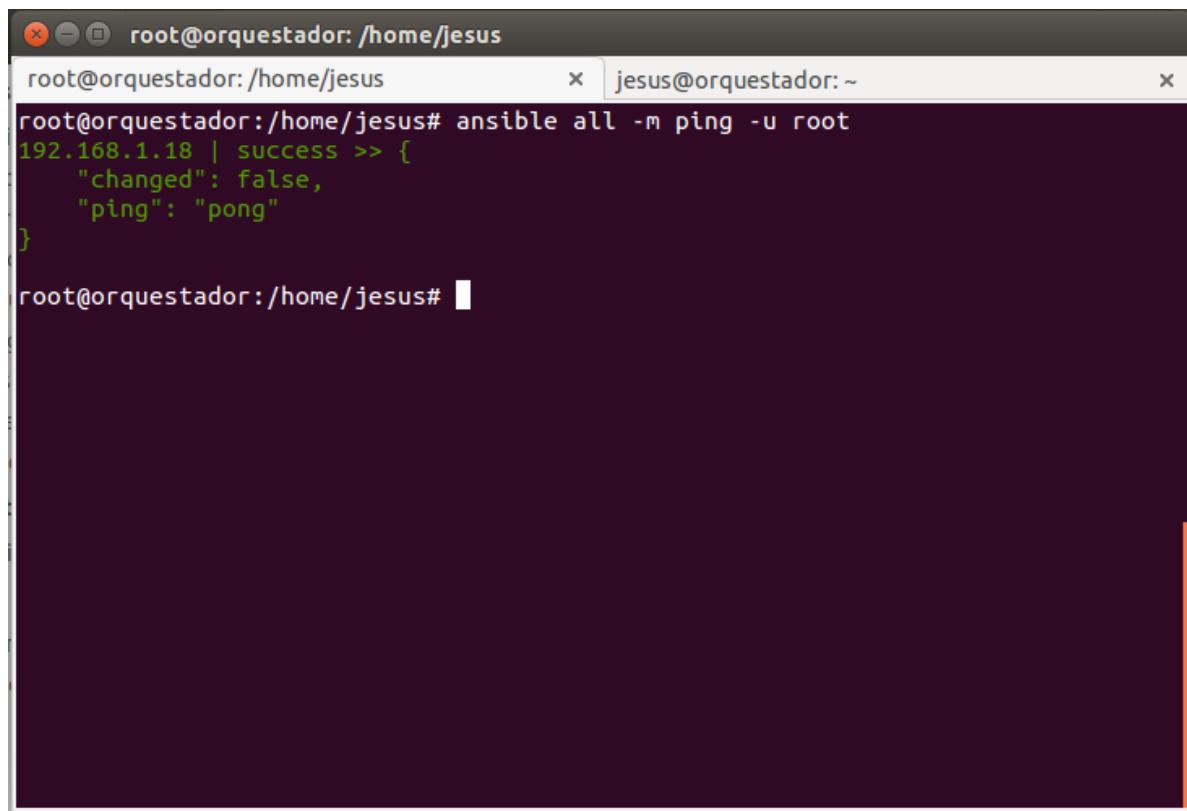
```
root@orquestador:/home/jesus
| .o* ...* .
| .+B..E o
| o+S. +
| . +
| .
|
+-----+
root@orquestador:/home/jesus# ssh-copy-id -i ~/.ssh/id_rsa.pub root@192.168.1.18
The authenticity of host '192.168.1.18 (192.168.1.18)' can't be established.
ECDSA key fingerprint is fa:ea:78:60:f8:2a:0d:7b:98:fe:12:c9:6a:56:f8:ae.
Are you sure you want to continue connecting (yes/no)? yes
/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 the new keys
root@192.168.1.18's password:

Number of key(s) added: 1

Now try logging into the machine, with:  "ssh 'root@192.168.1.18'"
and check to make sure that only the key(s) you wanted were added.

root@orquestador:/home/jesus#
```

Para comprobar que Ansible, desde el nodo orquestador, reconoce al equipo Servidor se realiza una comprobación.



The screenshot shows a terminal window with two tabs. The active tab is titled 'root@orquestador: /home/jesus' and contains the command 'ansible all -m ping -u root'. The output shows a single host at 192.168.1.18 with a success status, indicating it responded to the ping. The second tab, titled 'jesus@orquestador: ~', is visible but empty. The terminal has a dark background with light-colored text.

```
root@orquestador: /home/jesus
root@orquestador: /home/jesus# ansible all -m ping -u root
192.168.1.18 | success >> {
    "changed": false,
    "ping": "pong"
}

root@orquestador: /home/jesus#
```

3. Proxy no transparente

Para este proyecto se contará con un proxy no transparente Squid en la versión 3.2.8, ubicado entre la red interna y el router.

Squid es un programa que permite realizar un proxy con una gran variedad de configuraciones, ya que permite cacheo, filtro, balanceo de carga, registro de eventos o realice autenticación.

El proxy guarda en un log todas las conexiones que hacen los usuarios de modo que se puede realizar un exhaustivo control del tráfico, pudiendo ver, por ejemplo, los usuarios que han intentado acceder a un sitio web al que no estaban autorizados.

La configuración de Squid es sencilla; tiene un fichero de configuración ubicado en /etc/squid3/squid.conf en el que se especificarán los siguientes parámetros:

```
1 #parametros generales
2
3 #nombre del proxy
4 visible_hostname servidor_proxy
5
6 #puerto de escucha del nombre
7 http_port 3128
8
9 #dirección de la cache de squid y el tamaño de la misma
10 cache_dir ufs /var/spool/squid3 2000 16 256
11 cache_mem 32 MB
12 maximum_object_size_in_memory 256 MB
13
14 #dirección de los registros de squid
15 access_log /var/log/squid3/access.log
16 cache_log /var/log/squid3/cache.log
```

- *visible_hostname*: Nombre del proxy que en este caso se llamará *servidor proxy*.
- *http_port*: Número del puerto en el que escuchará Squid.
- *cache_dir ufs /var/spool/squid3 2000 16 256*
cache_mem 32 MB
maximum_object_size_in_memory 256 MB: Dirección de la caché de Squid y el tamaño de la misma.

- *access_log* /var/log/squid3/access.log
cache_log /var/log/squid3/cache.log: Dirección de los registros de squid.

```

18 #autentication
19 auth_param basic program /usr/lib/squid3/basic_ncsa_auth /etc/squid3/claves
20 auth_param basic children 5
21 auth_param basic realm Squid proxy-caching web Server
22 auth_param basic credentialsttl 2 hours
23
24 #listas de control de acceso
25 #acl passwd proxy_auth REQUIRED
26 acl acceso src all
27 #acl nopermitidas url_regex "/etc/squid3/nopermitidas"
28 acl nowebs dstdomain "/etc/squid3/nowebs"
29 #acl extensiones urlpath_regex "/etc/squid3/extensiones"
30 #acl permitidas url_regex "/etc/squid3/permitidas"
31 #acl limit maxconn 20
32
33 #control de acceso
34 #http_access allow permitidas
35 #http_access deny maql
36 #http_access deny extensiones
37 #http_access deny nopermitidas
38 http_access deny nowebs
39 #http_access deny !passwd
40
41 http_access allow acceso

```

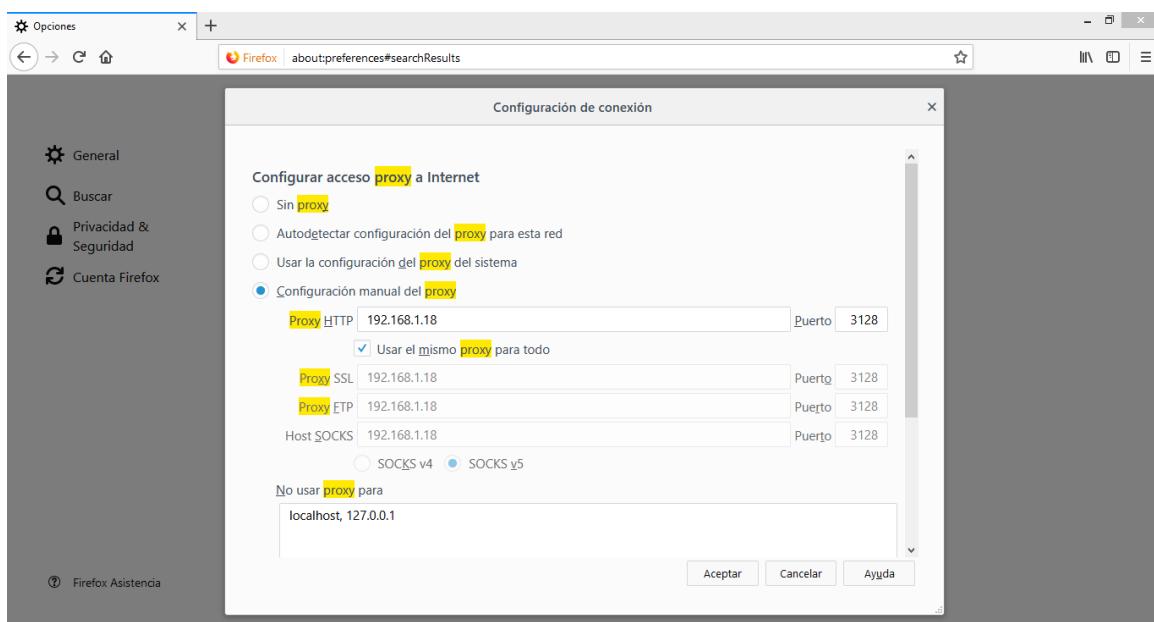
- *auth_param*: Ajustes de autenticación.
- *ACL*: Son listas de control de acceso con las cuales se puede establecer reglas como límites de una conexión, permitir palabras dentro de un fichero o denegar dominios.
- *http_access*: Control de acceso para las acl previamente declaradas.

Para este proyecto se ha creado una *ACL* que especifica en un archivo llamado *nowebs* una serie de dominios que a continuación se denegará con *http_access deny*.

```
GNU nano 2.2.6 Fichero: /etc/squid3/nowebs

www.facebook.com
http://facebook.com
https://facebook.com
www.docecanos.com
http://docecanos.com
https://docecanos.com
www.as.com
https://as.com
http://as.com
www.twitter.com
http://twitter.com
https://twitter.com
```

De modo que los usuarios que quieran navegar necesitarán configurar su navegador estableciendo un proxy con la IP del servidor y puerto 3128.



Tras ello el usuario podrá navegar por Internet exceptuando los dominios que se han denegado mediante la *ACL*.



Para la instalación de Squid y su configuración es necesario programarlo en Ansible pero, primero que todo, es necesario actualizar la lista de los paquetes en el servidor.

Para ello existe el módulo *apt* que con la sintaxis *update_cache=yes* permite realizar esta tarea. Tras ello, el módulo *apt:* con la orden *name=squid3 state=latest*.

De modo que, en el playbook que se ha creado con el nombre *respaldo.yml*, se describe primero el nombre del grupo de hosts y a continuación las tareas que se han descrito anteriormente.

```
1  ---
2  - hosts: respaldo
3    sudo: yes
4    tasks:
5
6      - name: Actualizar paquetes
7        apt: update_cache=yes
8
9
10 ##### S Q U I D #####
11
12
13   - name: Instalar Squid
14     apt: name=squid3 state=latest
15
```

Una vez hecho esto, en el servidor quedaría instalado Squid. Pero para este proyecto, se ha definido una *acl* en el fichero de configuración *squid.conf* llamado *nowebs* que especifica en un fichero llamado *nowebs* una serie de dominios que se prohibirán por lo que, será necesario programar en Ansible el copiado de *squid.conf* y *nowebs* al servidor.

```
16   - name: Configurar squid.conf
17     template: src=template/squid.conf dest=/etc/squid3/ owner=root mode=0640 backup=yes
18
19   - name: Configurar squid.conf
20     template: src=template/nowebs dest=/etc/squid3/ owner=root mode=0640 backup=yes
21
22   - name: Reiniciar el servicio Squid
23     service: name=squid3 state=restarted
```

Por último, se reiniciará Squid con el módulo *service*, como queda detallado en la imagen anterior.

3.1. Listas negras

Squid dispondrá de una lista negra generada por Easylist que se actualizará cada semana. Para ello, con wget, se descargará la lista y se almacenará en /etc/squid3/easylist.txt.

```
root@debian:/home/jesus# wget https://easylist.to/easylist/easylist.txt -P /etc/squid3/
--2018-06-05 19:55:20-- https://easylist.to/easylist/easylist.txt
Resolviendo easylist.to (easylist.to)... 104.27.147.83, 104.27.146.83, 2400:cb00:2048:1::681b:9253, ...
Conectando con easylist.to (easylist.to)[104.27.147.83]:443... conectado.
Petición HTTP enviada, esperando respuesta... 200 OK
Longitud: no especificado [text/plain]
Grabando a: "/etc/squid3/easylist.txt"                                I

/etc/squid3/easylist.txt      [      <=>                               ]  2,53M  2,60MB/s  en 1,0s
2018-06-05 19:55:21 (2,60 MB/s) - “/etc/squid3/easylist.txt” guardado [2654258]
root@debian:/home/jesus# █
```

Además necesario crear una nueva *ACL* en *squid.conf* que deniegue el acceso a las webs y direcciones IP que contenga el fichero creado.

```
#listas de control de acceso
#acl passwd proxy_auth REQUIRED
acl acceso src all
#acl nopermitidas url_regex "/etc/squid3/nopermitidas"
#acl nowebs dstdomain "/etc/squid3/nowebs"
acl easylist dstdomain "/etc/squid3/easylist.txt"
#acl extensiones urlpath_regex "/etc/squid3/extensiones"
#acl permitidas url_regex "/etc/squid3/permitidas"
#acl limit maxconn 20

always_direct allow

#control de acceso
#http_access allow permitidas
#http_access deny maql
#http_access deny extensiones
#http_access deny nopermitidas
http_access deny nowebs
http_access deny easylist
#http_access deny !passwd

http_access allow acceso
```

Para que esta lista negra siempre esté actualizada, es necesario que se programe en *Cron*, de modo que se realizaría de la siguiente manera en Ansible.

```
28      - name: Borrar lista negra antigua
29          shell: echo "00 8    * * 6    root    rm -f /home/jesus/easylist.txt" >> /etc/crontab
30
31      - name: Descargar lista negra
32          shell: echo "00 8    * * 6    root    wget https://easylist.to/easylist/easylist.txt" >> /etc/crontab
33
```

3.2. Webmin

Para este proyecto se incluirá una interfaz amigable web que permitirá configurar y administrar reglas para Squid. La interfaz amigable que se empleará será Webmin, ya que es una herramienta muy potente que además de permitir lo anterior permite:

- Ver la versión del sistema operativo.
- Programar comandos y tareas.
- Gestionar procesos arrancados al inicio.
- Cambiar contraseña de root y otros usuarios.
- Leer logs del sistema.
- Gestionar grupos y usuarios.
- Comprobar procesos y servicios en ejecución.
- Administrar y configurar de servidores POP o servidor web Apache.
- Monitorizar el ancho de banda.
- Planificar copias de seguridad.

Para la instalación de Webmin es necesario instalar otros paquetes. Para ello se crea un array de items en Ansible que, de forma automatizada, se instalarán.

```
45      - name: Lista de paquetes a instalar de Webmin
46          apt: name={{item}} state=installed
47          with_items:
48              - perl
49              - libnet-ssleay-perl
50              - openssl
51              - libauthen-pam-perl
52              - libpam-runtime
53              - libbio-pty-perl
54              - apt-show-versions
55              - python
56
```

Tras esto se ha de descargar el paquete de Webmin. Con el módulo de Ansible `get_url` es posible descargar el paquete, aunque como en varias ocasiones ha fallado la descarga se ha optado por descargarlo manualmente y copiarlo al servidor mediante el módulo `copy`.

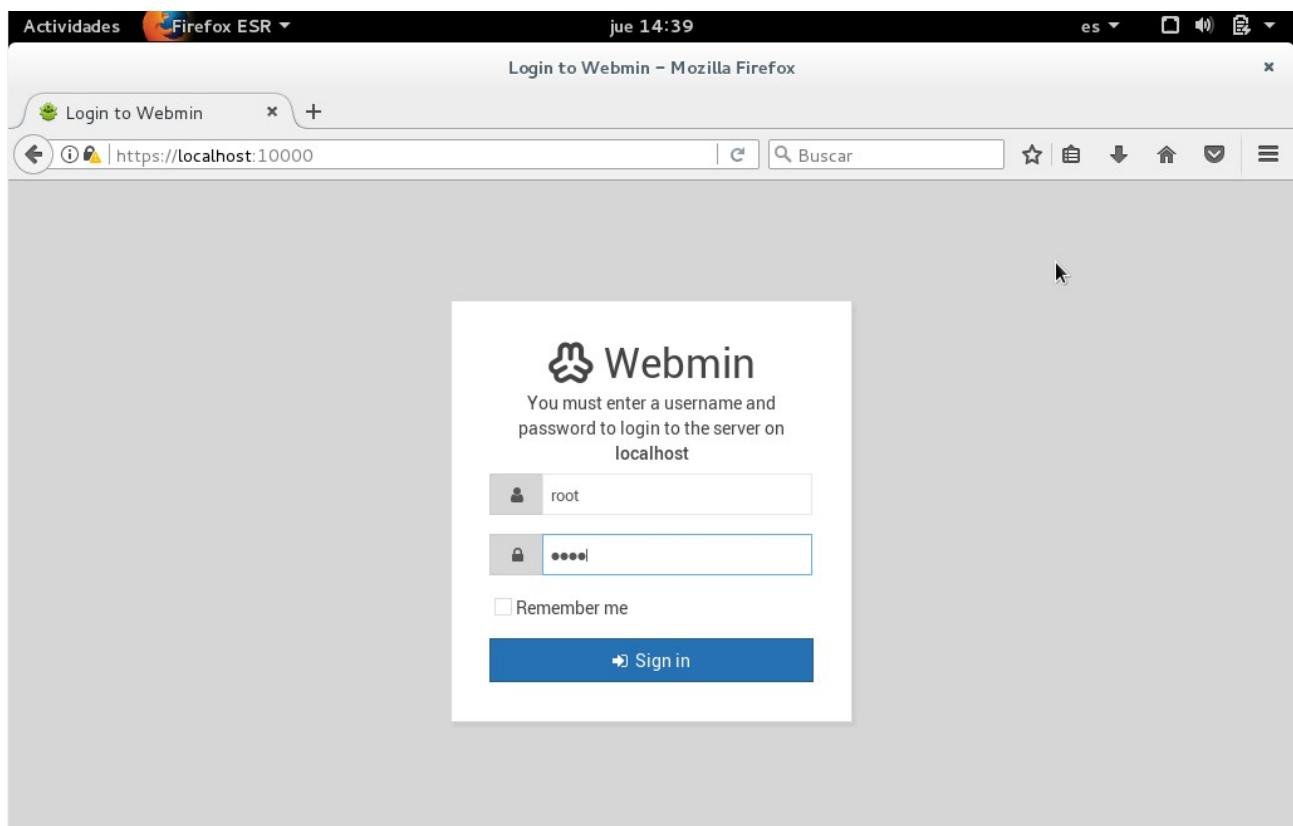
Una vez copiado al destino, el servidor, de instala emitiendo un comando sobre el paquete con el módulo de Ansible `shell`.

```
57      - name: Descargar el paquete Webmin
58        copy: src=archive/webmin_1.881_all.deb dest=/home/ owner=root mode=0640 backup=yes
59
60      - name: Instalando Webmin
61        shell: dpkg --install /home/webmin_1.881_all.deb
```

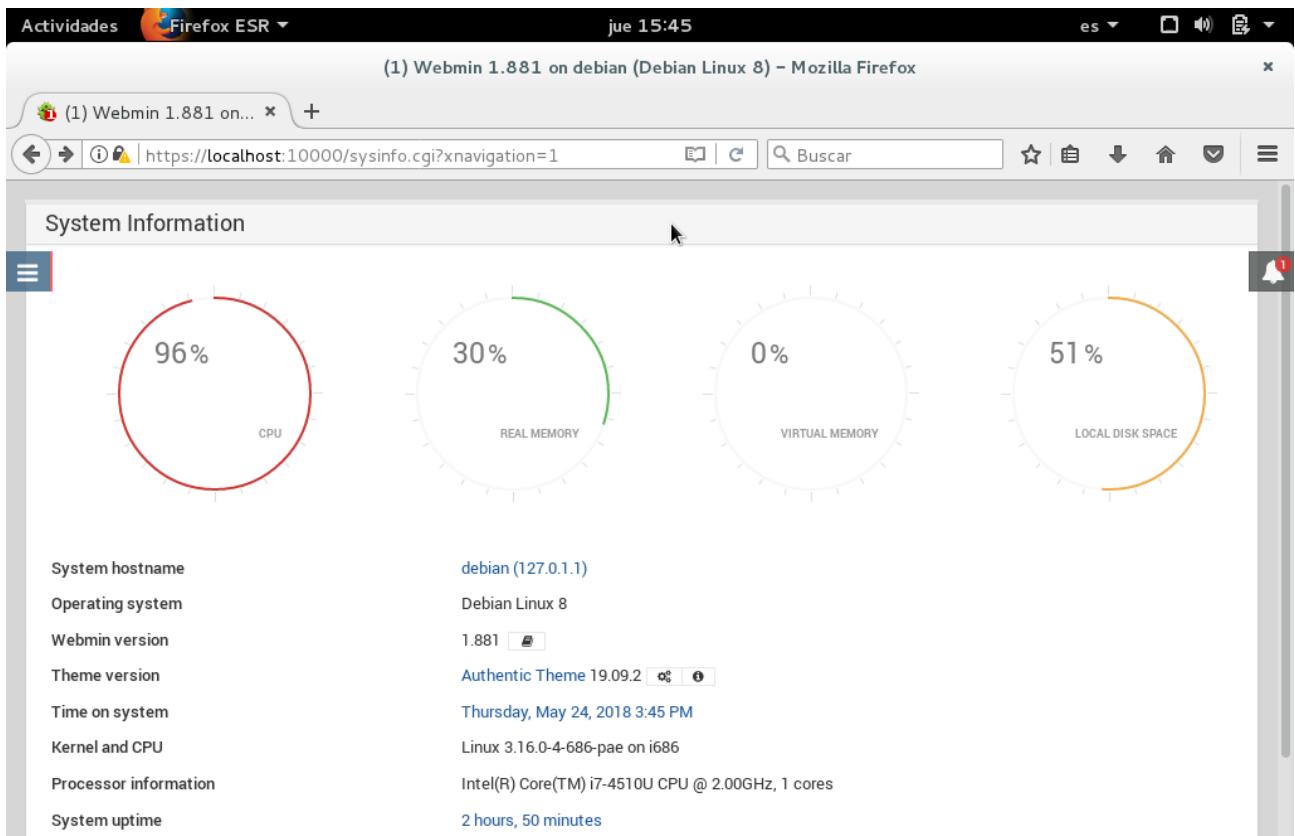
Por último, se reinicia el servicio.

```
63      - name: Reiniciar Webmin
64        service: name=webmin state=restarted
```

Una vez instalado y reiniciado el servicio en el destino, Webmin es visible desde un navegador web accediendo a la dirección <https://localhost:10000>. Las credenciales de acceso es el usuario `root` y su contraseña (Tico*65).



Una vez dentro se podrá ver inicialmente unas gráficas que mostrarán información actual del sistema, como el estado de la memoria RAM, procesador o espacio en disco.



Algunas de las opciones que esta herramienta ofrece pueden encontrarse en el menú lateral, como la opción cambiar contraseña a usuarios del sistema que se encuentra en *System > Change Passwords*.

The screenshot shows the 'Change Passwords' page. The title is 'Change Passwords'. Below it, a message says 'Select a user to change his or her password'. A table lists users in two columns:

| root | daemon | bin | sys |
|-----------------|-------------------|-------------------|-----------------|
| sync | games | man | lp |
| mail | news | uucp | proxy |
| www-data | backup | list | irc |
| gnats | nobody | systemd-timesync | systemd-network |
| systemd-resolve | systemd-bus-proxy | messagebus | avahi |
| Debian-exim | statd | colord | dnsmasq |
| geoclue | pulse | speech-dispatcher | rtkit |
| saned | usbmux | hplip | Debian-gdm |
| jesus | sshd | clamav | dansguardian |
| mysql | | | |

Actividades Firefox ESR jue 15:48 es ▾

(1) Change Password – Webmin 1.881 on debian (Debian Linux 8) – Mozilla Firefox

(1) Change Password... +

https://localhost:10000/passwd/edit_passwd.cgi?user=root&xnav

Buscar

Change Password

Changing Unix user password

Changing password for root (root)

New password:

New password (again):

Force user to change password at next login?

Change password in other modules?

Change

Return to user list

This screenshot shows the 'Change Password' page in Webmin. It's a form for changing the password for the root user. The 'Changing password for' field is set to 'root (root)'. There are two input fields for the 'New password' and 'New password (again)'. Below these are two checkboxes: one for forcing the user to change their password at the next login, and another for changing the password in other modules. A large orange 'Change' button is at the bottom. At the very bottom is a blue 'Return to user list' button.

También leer logs del sistema desde *System > System logs*.

Actividades Firefox ESR jue 15:51 es ▾

(1) System Logs – Webmin 1.881 on debian (Debian Linux 8) – Mozilla Firefox

(1) System Logs – W... +

https://localhost:10000/syslog/?xnavigation=1

Log destination Active? Messages selected

| Log destination | Active? | Messages selected |
|---------------------------------|---------|--------------------------|
| File /var/log/auth.log | Yes | auth,authpriv.* |
| File /var/log/syslog | Yes | *.* ; auth,authpriv.none |
| File /var/log/cron.log | No | cron.* |
| File /var/log/daemon.log | Yes | daemon.* |
| File /var/log/kern.log | Yes | kern.* |
| File /var/log/lpr.log | Yes | lpr.* |
| File /var/log/mail.log | Yes | mail.* |
| File /var/log/user.log | Yes | user.* |
| File /var/log/mail.info | Yes | mail.info |
| File /var/log/mail.warn | Yes | mail.warn |
| File /var/log/mail.err | Yes | mail.err |
| File /var/log/news/news.crit | Yes | news.crit |
| File /var/log/news/news.err | Yes | news.err |
| File /var/log/news/news.notice | Yes | news.notice |
| File /var/log/debug | Yes | news.none ; mail.none |
| File /var/log/messages | Yes | mail,news.none |
| Users :omusrmsg:* | Yes | *.emerg |
| File /dev/tty8 | No | *.=notice ; *.=warn |
| Named pipe /dev/xconsole | Yes | *.=notice ; *.=warn |
| File /var/log/apache2/error.log | Yes | Apache error log |
| File /var/log/fail2ban.log | Yes | Fail2Ban action log |

This screenshot shows the 'System Logs' page in Webmin. It lists various log files and their configuration. Each entry has a 'Log destination' (like /var/log/auth.log), an 'Active?' column (with 'Yes' or 'No'), and a 'Messages selected' column (showing log levels like auth,authpriv.* or cron.*). There are also 'View' buttons for each row. A red alert icon is visible on the right side of the interface.

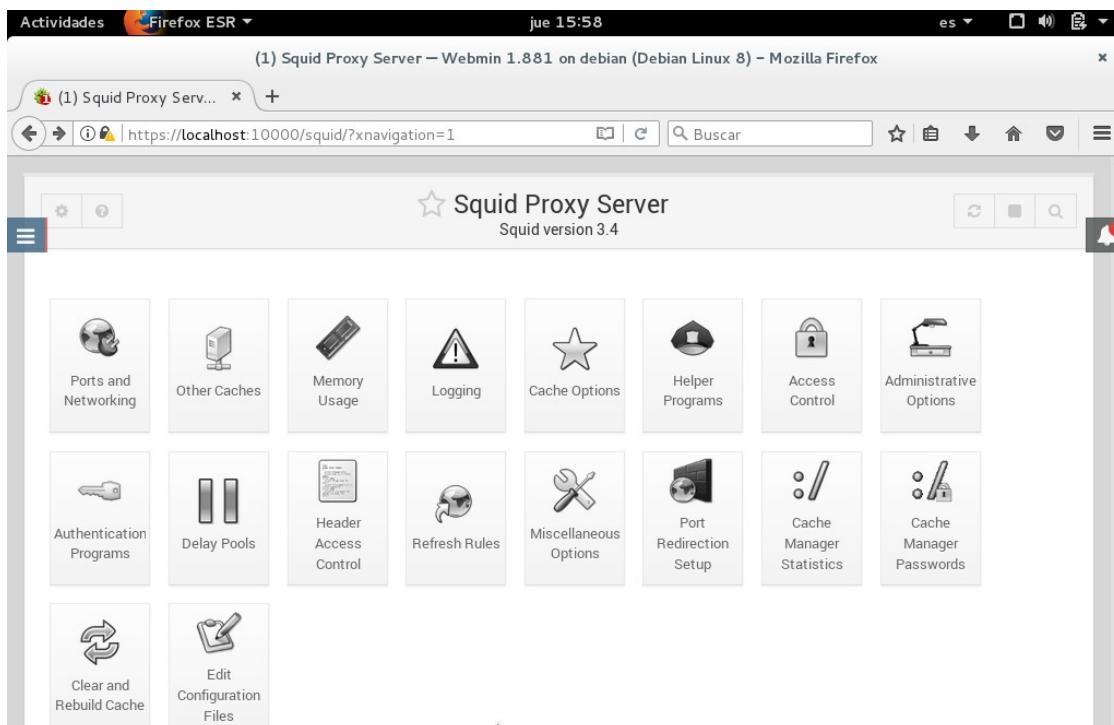
O también monitorizar el tráfico de red que pase por un adaptador concreto.

Este módulo de Webmin se puede encontrar en el menú lateral, en *Networking > Bandwidth Monitoring*.

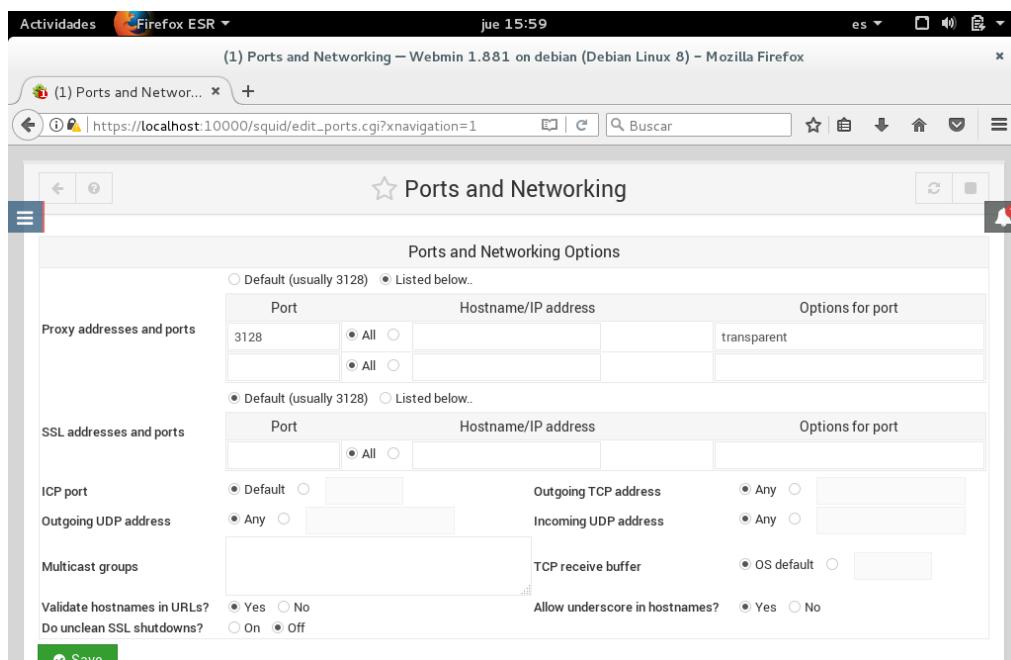
The screenshot shows a Firefox ESR browser window running on a Debian Linux 8 system. The title bar indicates it's 15:53 on a jueves (Thursday). The address bar shows the URL <https://localhost:10000/bandwidth/?xnavigation=1>. The main content area is titled "Bandwidth Monitoring" and sub-titled "Using IPTables firewall and Syslog". It features a "Setup Now" button and a dropdown menu for selecting the "External network interface". The dropdown menu lists "eth0", "eth1", "lo", and "Other..", with "eth0" currently selected. A warning message at the top states: "Before this module can report on network usage on your system, it must be set up to monitor traffic on the selected external network interface. Several firewall rules must be added, and a syslog configuration entry created. Warning - this module will log ALL network traffic sent or received on the selected interface. This will consume a large amount of disk space and CPU time on a fast network connection." A notification icon in the top right corner has a red dot indicating one notification.

3.3. Interfaz amigable

Una vez dentro de Webmin, en el menú lateral derecho dentro de *Servers* se encontrará la opción *Squid Proxy Server* que permitirá acceder a la configuración de Squid.



Algunas de las opciones que se usarían en este caso para configurar Squid es *Ports and Networking*, permite configurar opciones de puertos y red para Squid, como cambiar el puerto o permitir webs SSL determinadas.



El segundo ícono llamado *Other Caches* permite configurar otro servidor para almacenar la caché de Squid.

En el ícono *Memory Usage* se puede configurar el límite del tamaño de memoria que usará Squid.

The screenshot shows the 'Memory Usage' configuration page for Squid. The URL is https://localhost:10000/squid/edit_mem.cgi. The page title is 'Memory Usage'. It contains several configuration options:

- Memory usage limit: Set to 32 MBs.
- Disk high-water mark: Set to Default.
- Maximum cached object size: Set to Default.
- IP cache high-water mark: Set to Default.
- Disk replacement policy: Set to Default.
- FQDN cache size: Set to Default.
- Disk low-water mark: Set to Default.
- IP address cache size: Set to Default.
- IP cache low-water mark: Set to Default.
- Memory replacement policy: Set to Default.

A green 'Save' button is visible at the bottom left. At the bottom center is a link to 'Return to squid index'.

En *Login* se realizan las opciones de registro y archivo de registro de Squid.

En el ícono *Cache Options* se puede definir el tamaño que ocupará el directorio de caché de Squid, siendo por defecto 200 MB en */var/spool/squid3*.

The screenshot shows the 'Cache Options' configuration page for Squid. The URL is https://localhost:10000/squid/edit_cache.cgi?xnavigation=1. The page title is 'Cache Options'. It contains two main sections:

- Caching and Request Options**:
 - Cache directories: Set to Listed. A table shows one entry: /var/spool/squid3 (UFS, 2000 MB, 16 1st level dirs, 256 2nd level dirs).
 - Average object size: Set to Default.
 - Don't cache URLs for ACLs: Set to all.
 - Maximum request body size: Set to Default.
 - Maximum client read-ahead gap: Set to Default.
 - Maximum reply body sizes: Set to Size (kB).
- Objects per bucket**: Set to Default.
- Maximum cache time**: Set to Default.
- Maximum request headers size**: Set to Default.
- Failed request cache time**: Set to Default.

En el ícono *Access Control* se encuentra una de las opciones más importantes de la interfaz, ya que permite añadir nuevas *ACL* en función de lo que se necesite. En la pestaña *Access control lists* existe un desplegable con muchas opciones dependiendo qué se necesite.

En las otras pestañas la interfaz permite restringir ips, añadir restricciones o añadir reglas *ACL* de programas externos

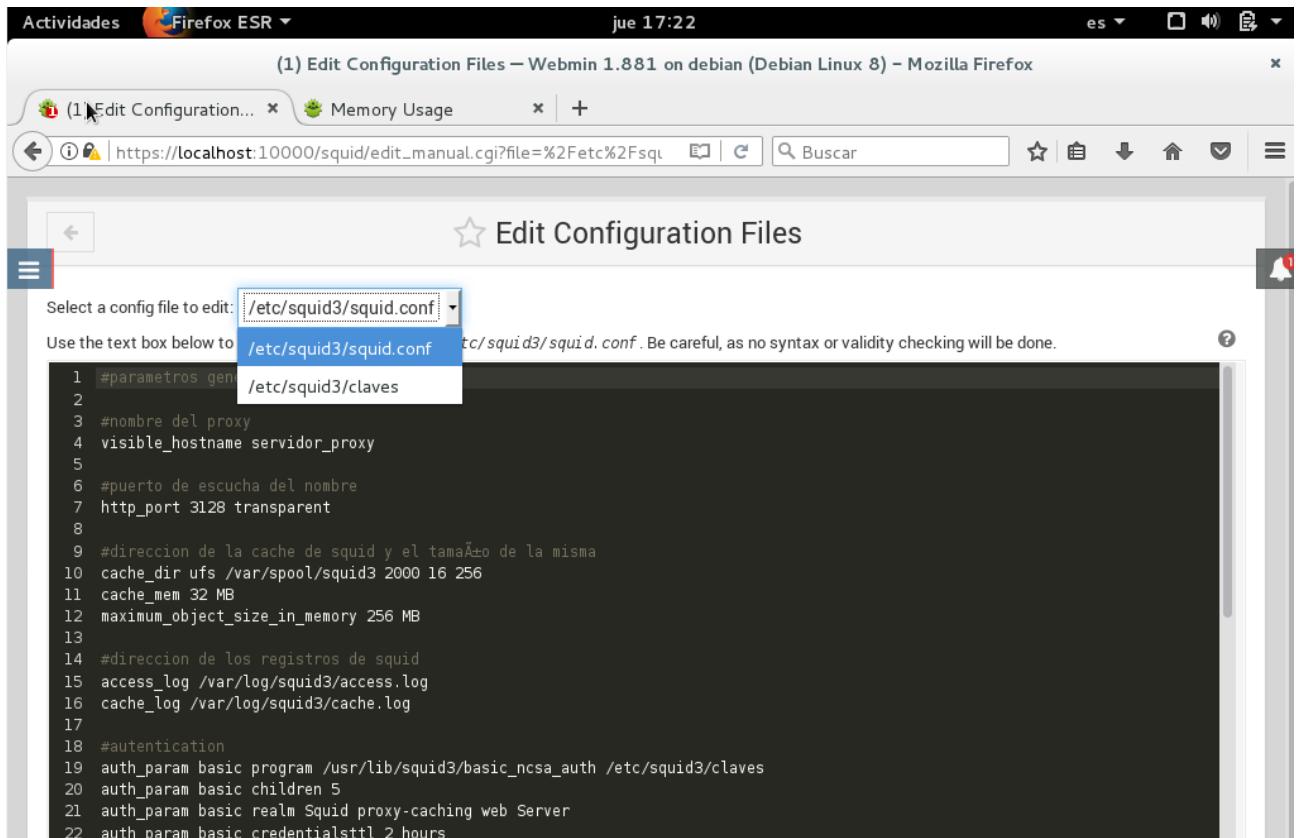
The screenshot shows the 'Access Control' section of the Squid configuration interface. At the top, there are tabs for 'Access control lists', 'Proxy restrictions', 'ICP restrictions', 'External ACL programs', and 'Reply proxy restrictions'. Below these tabs, there is a table with two rows. The first row has columns for 'Name' (containing 'acceso'), 'Type' (containing 'Client Address'), and 'Matching...' (containing 'all'). The second row is a new entry being created, with 'Name' set to '+ Create new ACL' and 'Type' set to 'Browser Regexp'. A dropdown menu is open next to 'Type', showing various options: Client Address, Client Hostname, Client Regexp, Date and Time, Dest AS Number, Ethernet Address, External Auth, External Auth Regexp, External Program, Max User IP, Maximum Connections, Proxy IP Address, Proxy Port, and RFC931 User. A blue button labeled 'Return to squid index' is visible at the bottom left of the interface.

Sobre el ícono llamado *Proxy Authentication* se puede establecer mediante una *ACL* el acceso al servidor proxy limitando a los usuarios enumerados en una lista.

The screenshot shows the 'Proxy Authentication' section of the Squid configuration interface. At the top, there is a message: 'With the right ACLs, access to your proxy server can be limited to the users listed below, taken from the file /etc/squid3/claves. After adding, deleting or changing a user you must use the Apply Changes link for the modification to take effect.' Below this message, it says 'No proxy users are currently defined.' There is a button labeled 'Add a new proxy user'. At the bottom left, there is a blue button labeled 'Return to squid index'. The browser's title bar shows 'Actividades Firefox ESR' and the address bar shows 'https://localhost:10000/squid/edit_nauth.cgi?xnavigation=1'.

Por último destacar una de las opciones más importantes que incluye este módulo de Webmin, *Edit Configuration Files*. Permite editar y actualizar los ficheros de configuración de Squid ubicados en */etc/squid3* y */etc/squid3/claves*.

Permite hacer lo mismo que si se editan mediante un gestor de texto o la terminal.

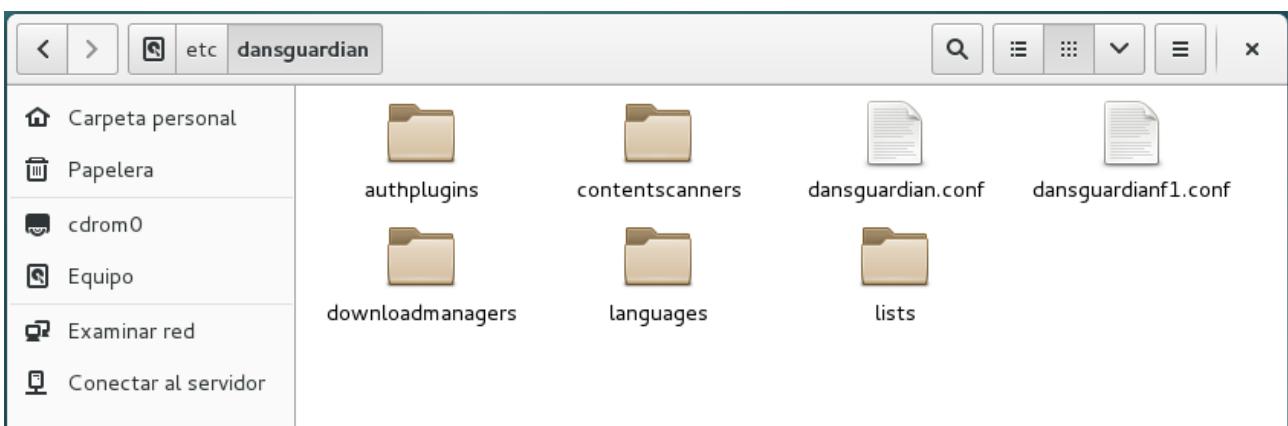


4. Dansguardian

Este proyecto contará con otro proxy llamado Dansguardian, el cual es software de control de contenidos, diseñado para controlar el acceso a sitios web. El usuario podrá elegir entre este proxy o Squid, ya que se encuentran problemas de SSL en el servidor y no es posible instalar un certificado en el servidor.

Incluye un filtro de virus y es usado principalmente en instituciones de educación, gobierno y empresas. Se caracteriza por su alto grado de flexibilidad y adaptación de la implementación.

Dansguardian tiene su carpeta de configuración ubicada en `/etc/dasnguardian`, cuenta con un archivo de configuración llamado `dansguardian.conf` y varias carpetas que contienen diferentes configuraciones.



Respecto al fichero de configuración, es necesario comentar una línea que se encuentra en el principio de documento que empieza con UNCONFIGURED.

```
DansGuardian config file for version 2.10.1.1
# **NOTE** as of version 2.7.5 most of the list files are now in dansguardianf1.conf
#UNCONFIGURED - Please remove this line after configuration
# Web Access Denied Reporting (does not affect logging)
```

Tras esto, Dansguardian permite cambiar el idioma a español, por lo que se editará la línea *language*.

```
# The language file is used no matter what setting however.
#
languagedir = '/etc/dansguardian/languages'

# language to use from languagedir.
language = 'spanish'

# Loaaina Settinias
```

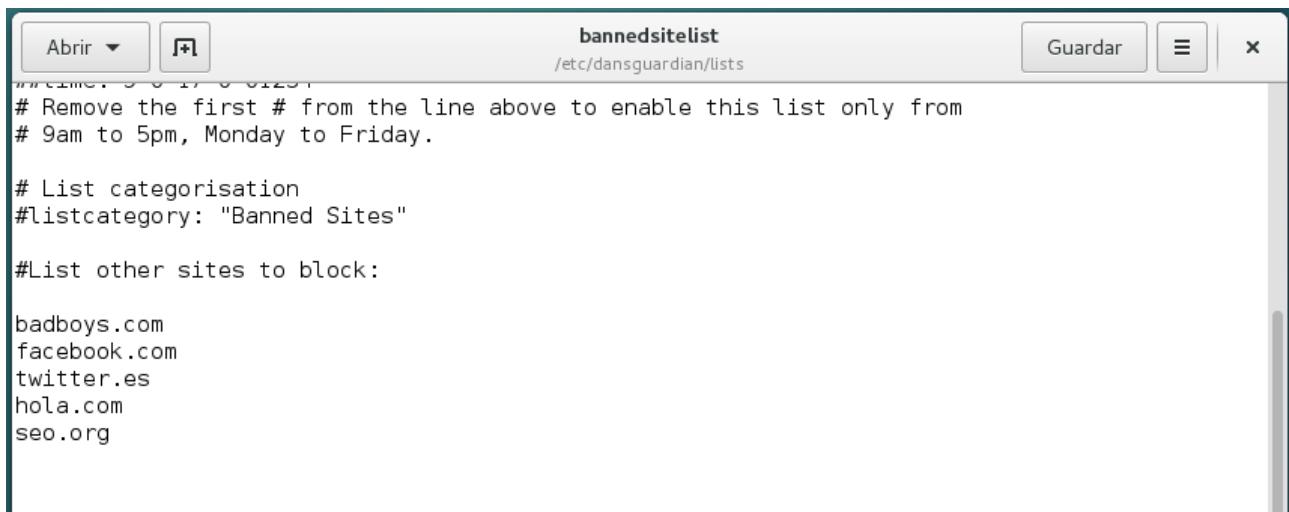
Por último es necesario indicar en el fichero de configuración el puerto donde escuchará Dansguardian, el puerto donde escuchará Squid y la dirección IP donde residirá Squid.

- filterport = 8080 → Puerto de Dansguardian.
- proxyip = 127.0.0.1 → Dirección IP de la maquina donde reside Squid.
- proxyport = 3128 → Puerto donde escucha Squid.

La configuración de Dansguardian para este proyecto se centra en la carpeta *lists*, ya que permite o no a los clientes navegar en función de una dirección web, palabras, frases, excepciones o extensiones de archivos prohibidos. Estas son las opciones que permiten cada uno de los archivos de *lists*:

- bannedextensionlist → Deniega las listas de extensiones de archivos prohibidos.
- bannediplist → Deniega las listas de IP que no van a tener acceso.
- bannedmimetypeslist → Deniega los tipos de listas de contenidos no permitidos.
- bannedphraselist → Deniega las listas de palabras o frases prohibidas.
- bannedregexpurllist → Deniega las listas de expresiones prohibidas en URL.
- bannedsitelist → Bloquea el acceso a un dominio.
- bannedurllist → Impide el acceso a una parte del dominio.
- exceptioniplist → Lista de IP que no serán filtradas.
- exceptionphraselist → Palabras o frases que no se filtrarán.
- exceptionsitelist → Dominios que no se prohibirán.
- exceptionurllist → Parte de un dominio que no se bloqueará.
- weightedphraselist → Lista de palabras con su correspondiente peso e instrucciones de definición.

Este proxy tendrá configurado por defecto en el archivo *bannedsitelist* algunos sitios web.



A screenshot of a text editor window titled "bannedsitelist" located at "/etc/dansguardian/lists". The window has standard OS X-style controls (Abrir, Guardar, etc.). The text in the editor is as follows:

```
# Remove the first # from the line above to enable this list only from
# 9am to 5pm, Monday to Friday.

# List categorisation
#listcategory: "Banned Sites"

#List other sites to block:

badboys.com
facebook.com
twitter.es
hola.com
seo.org
```

Todo lo anterior es implementado en Ansible, copiando el archivo de configuración de *Dansguardian* y el archivo *bannedsitelist* y dejando activo el servicio.

```
26 ##### D A N S G U A R D I A N
27
28
29 - name: Instalar Dansguardian
30   apt: name=dansguardian state=latest
31
32 - name: Configurar Dansguardian
33   template: src=template/dansguardian.conf dest=/etc/dansguardian/ owner=root mode=0640 backup=yes
34
35 - name: Configurar Dansguardian
36   template: src=template/bannedsitelist dest=/etc/dansguardian/lists/ owner=root mode=0640 backup=yes
37
38 - name: Reiniciar el servicio Dansguardian
39   service: name=dansguardian state=restarted
```

5. Reportes de usuarios

A pesar de que Squid constantemente estará escribiendo en su log, resulta algo difícil ver a golpe de vista los usuarios que se han conectado al servidor, los sitios webs que han visitado y también los sitios webs que no han podido visitar debido a una denegación de las reglas del proxy.

Para obtener unos informes detallados se empleará Sarg, una herramienta de código abierto que permite analizar los logs de Squid y genera informes web en formato HTML con información sobre usuarios, direcciones IP, sitios web visitados, uso de ancho de banda total, tiempo transcurrido, descargas e sitios web denegados.

Para la instalación de Sarg solo bastará con instalar el paquete mediante la terminal y modificar su fichero de configuración *sarg.conf* que se ubica en */etc/sarg*.

Será necesario editar la línea *access_log* para indicar donde Sarg debe tomar el log de Squid. Además será necesario indicar donde mostrará la salida de los reportes web, en línea *output_dir* indicando */var/www/html/squid-reports*.



```
# sarg.conf
#
# TAG: access_log
#       Where is the access.log file
#       sarg -l file
#
access_log /var/log/squid/access.log
```



```
# TAG: output_dir
#       The reports will be saved in that directory
#       sarg -o dir
#
output_dir /var/www/html/squid-reports
#output_dir /var/lib/sarg
```

Tras esto es necesario actualizar el servicio, y poner Sarg a funcionar con el comando *Sarg -x*. Es desde este punto cuando Sarg ya comenzará a mostrar en la ruta web que se ha indicado los informes de Squid.

Desde un cliente Ubuntu conectado al adaptador eth1 del servidor, al navegar Sarg comienza a representar vía web los informes de cada usuario mostrando los sitios web visitados y denegados.

| FILE/PERIOD | CREATION DATE | USERS | BYTES | AVERAGE |
|---------------------|-------------------------------|-------|--------|---------|
| 2018Apr24-2018May10 | jue 10 may 2018 15:05:33 CEST | 4 | 49.85M | 12.46M |
| 2018Apr24-2018Apr24 | mar 24 abr 2018 22:18:37 CEST | 2 | 2.31M | 1.15M |

Generated by sarg-2.3.6 Apr-21-2013 on may/10/2018 15:05

| Reportes de acceso de usuario de Squid | | | | | | | |
|--|---------|---------|--------|--------------|--------------|----------|------------------|
| Period: 2018 abr 24 - 2018 may 10 | | | | | | | |
| User: 192.168.11.2 | | | | | | | |
| Sort: bytes, reverse | | | | | | | |
| User report | | | | | | | |
| ACCESSED SITE | CONNECT | BYTES | %BYTES | IN-CACHE-OUT | ELAPSED TIME | MILLISEC | %TIME |
| turismosierradeacena.com | 228 | 17.57M | 78,78% | 0,00% | 100,00% | 00:00:25 | 25,109 42,96% |
| www.juntadeandalucia.es | 52 | 1.92M | 8,61% | 0,00% | 100,00% | 00:00:12 | 12,344 21,12% |
| error.invalid-request | 310 | 1.25M | 5,63% | 100,00% | 0,00% | 00:00:00 | 16 0,03% |
| lalavajera.com | 73 | 806.40K | 3,62% | 0,00% | 100,00% | 00:00:05 | 5,637 9,65% |
| dsms0mj1bbhn4.cloudfront.net | 2 | 158.72K | 0,71% | 0,00% | 100,00% | 00:00:00 | 141 0,24% |
| pagead2.googlesyndication.com | 3 | 123.97K | 0,56% | 0,00% | 100,00% | 00:00:00 | 217 0,37% |
| apps.shareaholic.com | 4 | 88.90K | 0,40% | 3,39% | 96,61% | 00:00:00 | 350 0,60% |
| scontent-cdt1.cdninstagram.com | 3 | 80.92K | 0,36% | 0,00% | 100,00% | 00:00:00 | 523 0,89% |
| beatsaudio.com | 1 | 71.44K | 0,32% | 0,00% | 100,00% | 00:00:00 | 428 0,73% |
| fonthoststatic.com | 3 | 55.37K | 0,25% | 0,00% | 100,00% | 00:00:00 | 135 0,23% |
| partner.googleadservices.com | 2 | 23.81K | 0,11% | 0,00% | 100,00% | 00:00:00 | 197 0,34% |
| www.squid-cache.org | 1 | 13.08K | 0,06% | 100,00% | 0,00% | 00:00:00 | 491 0,84% |
| www.docecanos.com | 2 | 7.44K | 0,03% | 100,00% | 0,00% | 00:00:00 | 169 0,29% DENIED |
| 2.gravatar.com | 5 | 7.14K | 0,03% | 32,81% | 67,19% | 00:00:00 | 251 0,43% |
| www.google.es | 4 | 4.11K | 0,02% | 0,00% | 100,00% | 00:00:00 | 563 0,96% |
| 1.gravatar.com | 3 | 3.99K | 0,02% | 14,06% | 85,94% | 00:00:00 | 129 0,22% |
| %5B%CB%BD%ED%9F%8E%E9%8C%7F%E6%5C%06=%%D7%98%0%F%C6%E3%85%05%A%7C%08%9C=%8C%C2 | 1 | 3.80K | 0,02% | 100,00% | 0,00% | 00:00:00 | 0 0,00% |
| %85%C3 | 1 | 3.80K | 0,02% | 100,00% | 0,00% | 00:00:00 | 0 0,00% |
| %A1%CA%C4%C7ID%EB%0E%5DH%A2%F3%EAC%ED%17%BB%9E0%AE_%DE%C6%D8%9DM%94 | 1 | 3.78K | 0,02% | 100,00% | 0,00% | 00:00:00 | 0 0,00% |
| %C5%D7%88RC%9A%BB%E4%1E%97%FB%EA%95%F%4Aq%0E%3E%AB%07%A8%F3 | 1 | 3.76K | 0,02% | 100,00% | 0,00% | 00:00:00 | 0 0,00% |

Todo lo anterior quedaría programado en Ansible instalando el paquete de Sarg, copiando su archivo de configuración al servidor y poniéndolo en marcha ejecutando Sarg -x desde la terminal con el módulo *shell*.

```

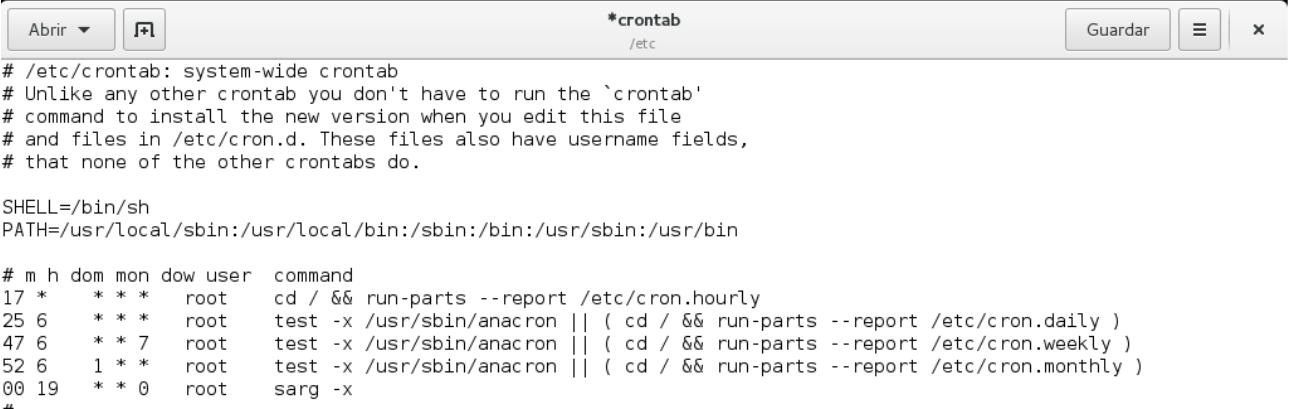
95      - name: Instalar Sarg
96          apt: name=sarg state=latest
97
98      - name: Configurar Sarg
99          template: src=template/sarg.conf dest=/etc/sarg/ owner=root mode=0640 backup=yes
100
101     - name: Iniciando Sarg
102        shell: sarg -x

```

5.1. Sarg en Cron

Para que se tengan informes todos los días es necesario lanzar Sarg para que genere informes, de modo que para que todas las mañanas al empezar las clases se puedan ver los informes del día anterior es necesario programar la tarea.

Para ello se utilizará Cron, para ello hay que editar el fichero *crontab*.



```
# /etc/crontab: system-wide crontab
# Unlike any other crontab you don't have to run the `crontab`
# command to install the new version when you edit this file
# and files in /etc/cron.d. These files also have username fields,
# that none of the other crontabs do.

SHELL=/bin/sh
PATH=/usr/local/sbin:/usr/local/bin:/sbin:/bin:/usr/sbin:/usr/bin

# m h dom mon dow user  command
17 *    * * *    root    cd / && run-parts --report /etc/cron.hourly
25 6    * * *    root    test -x /usr/sbin/anacron || ( cd / && run-parts --report /etc/cron.daily )
47 6    * * 7    root    test -x /usr/sbin/anacron || ( cd / && run-parts --report /etc/cron.weekly )
52 6    1 * * *    root    test -x /usr/sbin/anacron || ( cd / && run-parts --report /etc/cron.monthly )
00 19    * * 0    root    sarg -x
#
```

Tal y como se aprecia en la imagen, todos los días a las 19h de la tarde se lanzará la tarea *Sarg -x* que generará informes accesibles mediante un navegador web visitando el sitio <http://localhost/squid-reports>.

A la instalación y configuración anterior de Sarg en Ansible se debe añadir la siguiente tarea para obtener estos informes.

```
93      - name: Programado de los informes
94          shell: echo "00 19    * * 0    root    sarg -x" >> /etc/crontab
95
96      - name: Reiniciando el programado de los informes
97          service: name=cron state=restarted enabled=yes
98
```

6. Implementación firewall

Sobre este servidor de seguridad se implementará un cortafuegos llamado Iptables, el cual es el cortafuegos utilizado para gestionar las conexiones en Linux.

Las posibilidades de Iptables son prácticamente infinitas y un administrador que quiera sacarle el máximo provecho puede realizar configuraciones extremadamente complejas; Iptables permite crear reglas que analizarán los paquetes de datos que entran, salen o pasan por el servidor y en función de las condiciones que se hayan establecido, se tomará una decisión que normalmente será permitir o denegar que dicho paquete siga su curso.

Para crear las reglas, se analizarán muchos aspectos de los paquetes de datos, pero se filtran los paquetes en función de:

- Tipo de paquetes de datos:
 - **Tipo INPUT** → paquetes que llegan al servidor.
 - **Tipo OUTPUT** → paquetes que salen del servidor.
 - **Tipo FORWARD** → paquetes que pasan por el servidor.
- Interfaz por la que entran (-i = input) o salen (-o = output) los paquetes:
 - eth0, eth1, wlan0, ppp0, ...
- IP origen de los paquetes (-s = source):
 - IP concreta, ej: 10.0.1.3
 - Rango de red, ej: 10.0.1.0/8
- IP destino de los paquetes (-d = destination):
 - IP concreta, ej: 10.0.1.3
 - Rango de red, ej: 10.0.1.0/8

- Protocolo de los paquetes (-p = protocol):
 - Tcp, udp, icmp...
- Protocolo de los paquetes (-p = protocol):
 - Filtrar antes de enrutar: **PREROUTING**.
 - Filtrar después de enrutar: **POSTROUTING**.

6.1. Iptables

Para el filtrado de paquetes en este servidor se creará un script llamado *iptables.sh*, que contendrá las reglas y se ejecutará al inicio de cada encendido del servidor, y que establecerá una política de denegar por defecto. En función de las necesidades que requieran las redes se aceptarán los paquetes de datos.

Primero será necesario activar *IP forwarding* en el servidor Debian, para que el servidor pueda reenviar los paquetes por la interfaz adecuada y así llevarlos hacia su destino.

```

15  #Convertir en enrutador
16
17  echo 1 > /proc/sys/net/ipv4/ip_forward
18

```

Lo siguiente será eliminar otras reglas Iptables que pudiera haber configuradas y aplicar la política de denegar todo por defecto.

```

19
20  #Flush de reglas IPTABLES
21
22  iptables -F
23  iptables -X
24  iptables -Z
25  iptables -t nat -F
26
27
28  #Permitir todo por defecto
29
30  iptables -P INPUT DROP
31  iptables -P OUTPUT DROP
32  iptables -P FORWARD DROP
33

```

Las primeras reglas que se aplicarán serán la de permitir la salida y entrada de paquetes de datos usando los protocolos *tcp* y *udp* en el puerto 53 de la interfaz *eth0*, con el fin de poder resolver nombres de dominio desde el servidor.

```
37 #####DNS
38
39
40 iptables -A OUTPUT -p tcp --dport 53 -o eth0 -j ACCEPT
41 iptables -A INPUT -p tcp --sport 53 -i eth0 -j ACCEPT
42
43 iptables -A OUTPUT -p udp --dport 53 -o eth0 -j ACCEPT
44 iptables -A INPUT -p udp --sport 53 -i eth0 -j ACCEPT
45
```

Para que los clientes también puedan resolver nombres de dominio desde las redes Aula y DMZ será necesario hacer que los paquetes de datos que entren o salgan por el puerto 53 se permitan.

```
48 #DNS para red aula y DMZ
49
50
51 iptables -A FORWARD -p tcp --dport 53 -j ACCEPT
52 iptables -A FORWARD -p tcp --sport 53 -j ACCEPT
53
54 iptables -A FORWARD -p udp --dport 53 -j ACCEPT
55 iptables -A FORWARD -p udp --sport 53 -j ACCEPT
56
57
```

Con lo anterior ya están configuradas las reglas DNS. Respecto a las reglas HTTP/HTTPS, es necesario aceptar las peticiones entrantes y salientes que se dirijan al puerto 80 y 443 para poder navegar desde el servidor a sitios webs. Además será necesario permitir conexiones al puerto 10000 entrantes y salientes, ya que allí se encuentra Webmin.

```
59 ######HTTP & HTTPS
60
61
62
63 iptables -A INPUT -p tcp --sport 80 -i eth0 -j ACCEPT
64 iptables -A OUTPUT -p tcp --dport 80 -o eth0 -j ACCEPT
65
66 iptables -A INPUT -p tcp --sport 443 -i eth0 -j ACCEPT
67 iptables -A OUTPUT -p tcp --dport 443 -o eth0 -j ACCEPT
68
69 iptables -A OUTPUT -p tcp --sport 10000 -o eth0 -j ACCEPT
70 iptables -A INPUT -p tcp --dport 10000 -i eth0 -j ACCEPT
71
```

Con las reglas anteriores el servidor podría navegar por internet a sitios web con seguridad (HTTPS), sitios web sin seguridad (HTTP) y sería accesible desde la red exterior los reportes de usuarios Sarg y Webmin.

Aun queda que los clientes de la red Aula puedan navegar mediante los dos proxys; Dansguardian y Squid.

Para ello son necesarias dos reglas para cada proxy que permitan la salida y entrada de datos a los puertos de entrada y salida 8080 y 3128.

```
73 #Permitir la entrada y salida a squid
74 iptables -A INPUT -p tcp --dport 3128 -j ACCEPT
75 iptables -A OUTPUT -p tcp --sport 3128 -j ACCEPT
76
77 #Permitir la entrada y salida a Dansguardian
78 iptables -A INPUT -p tcp --dport 8080 -j ACCEPT
79 iptables -A OUTPUT -p tcp --sport 8080 -j ACCEPT
80
```

Por último, se necesita que las permitir la salida de los paquetes de datos provenientes de las redes Aula y DMZ

```
86
87 #Permitir que las redes Aula y DMZ tengan salida al exterior por el interfaz eth0
88 iptables -t nat -A POSTROUTING -o eth0 -s 192.168.11.0/24 -j MASQUERADE
89 iptables -t nat -A POSTROUTING -o eth0 -s 192.168.12.0/24 -j MASQUERADE
90
```

Respecto al protocolo ICMP, se permitirá dicho protocolo para poder realizar *ping* entre clientes de diferentes redes. Primeramente se permite que el servidor pueda realizar *ping* por su interfaz. Después, para las redes Aula y DMZs se permite que puedan realizar *ping* entre los clientes dentro de sus redes.

```
109 ######ICMP
110
111
112 iptables -A OUTPUT -p icmp -o eth0 -j ACCEPT
113 iptables -A INPUT -p icmp -i eth0 -j ACCEPT
114
115 #RED CLASE
116
117 iptables -A OUTPUT -p icmp -o eth1 -j ACCEPT
118 iptables -A INPUT -p icmp -i eth1 -j ACCEPT
119
120 #RED DMZ
121
122 iptables -A OUTPUT -p icmp -o eth2 -j ACCEPT
123 iptables -A INPUT -p icmp -i eth2 -j ACCEPT
124
```

También se debe añadir las reglas para el protocolo SSH. La configuración para SSH permitirá que los clientes de la red Aula puedan realizar conexiones SSH entre ellos, al igual que los clientes del DMZ.

```
128 #####SSH
129
130
131 iptables -A INPUT -p tcp --dport 22 -i eth0 -j ACCEPT
132 iptables -A OUTPUT -p tcp --sport 22 -o eth0 -j ACCEPT
133
134
135
136 #RED CLASE
137
138 iptables -A OUTPUT -p tcp --dport 22 -o eth1 -j ACCEPT
139 iptables -A INPUT -p tcp --sport 22 -i eth1 -j ACCEPT
140
141 #RED DMZ
142
143 iptables -A OUTPUT -p tcp --dport 22 -o eth2 -j ACCEPT
144 iptables -A INPUT -p tcp --sport 22 -i eth2 -j ACCEPT
145
```

Por último, el cortafuegos Iptables permitirá que tanto el servidor como los clientes de la red Aula y DMZ realicen conexiones FTP por los puertos 20 y 21.

```
149 #####FTP
150
151
152 iptables -A OUTPUT -p tcp --dport 20:21 -j ACCEPT
153 iptables -A INPUT -p tcp --sport 20:21 -j ACCEPT
154
155
156 #RED CLASE
157
158 iptables -A FORWARD -p tcp --sport 20:21 -j ACCEPT
159 iptables -A FORWARD -p tcp --dport 20:21 -j ACCEPT
160
161 #iptables -A INPUT -p tcp --sport 20:21 -i eth1 -j ACCEPT
162 #iptables -A OUTPUT -p tcp --dport 20:21 -o eth1 -j ACCEPT
163
164
165 #RED DMZ
166
167 iptables -A OUTPUT -p tcp --dport 20:21 -o eth2 -j ACCEPT
168 iptables -A INPUT -p tcp --sport 20:21 -i eth2 -j ACCEPT
```

6.2. Script de Iptables en el inicio

Las reglas del cortafuegos Iptables no se mantienen grabados en el sistema si el servidor es apagado y vuelto a encender. Para que se mantengan, es necesario incluir en el arranque de los servicios del sistema un script con las reglas que anteriormente se han descrito de Iptables.

Para ello es necesario copiar el script a la ruta `/etc/init.d/` y definir permisos de ejecución. Tras ello, incluir ese script llamado `iptables.sh` junto al arranque de servicios siendo el de menor prioridad.

```
106 ##### I P T A B L E S
107
108
109     - name: Copiar Iptables
110         template: src=template/iptables.sh dest=/etc/init.d/ owner=root mode=0640 backup=yes
111
112     - name: Dar permisos para la ejecución al inicio
113         shell: chmod +x /etc/init.d/iptables.sh
114
115     - name: Configurando Iptables al inicio de cada encendido
116         shell: update-rc.d iptables.sh defaults 99
117
```

6.3. Fail2ban

Fail2ban es una herramienta de seguridad que funciona como complemento a un cortafuegos o filtro de paquetes.

Fail2ban detecta aquellas direcciones IP cuyo comportamiento resulta inusual, por ejemplo, las que han intentado acceder varias veces con una contraseña incorrecta a los archivos de registro del servidor. Un cierto número de intentos fallidos, asegurará automáticamente que ese usuario sea "baneado", es decir, que su IP sea bloqueada durante un periodo determinado de tiempo.

El administrador también puede configurar Fail2ban para recibir notificaciones de este tipo de accesos por correo electrónico.

Algunos de sus parámetros del archivo de configuración ubicado en /etc/fail2ban/jail.conf son el tiempo de baneo tras una serie de intentos fallidos o el número de intentos fallidos por dirección IP.

```
# Tiempo en segundos que una dirección IP permanecerá bloqueada.  
bantime = 3600  
  
# A host is banned if it has generated "maxretry" during the last "findtime"  
# seconds.  
findtime = 1000  
  
# Cantidad máxima de intentos fallidos.  
maxretry = 3
```

Por defecto, estará configurado esta dirección de correo electrónico.

```
# Destination email address used solely for the interpolations in  
# jail.{conf,local} configuration files.  
destemail = jsus365@gmail.com  
|  
#  
# Name of the sender for mta actions  
sendername = Fail2Ban  
  
# Email address of the sender  
sender = fail2ban@localhost
```

¶

Además permite el baneo por servicios, como por ejemplo SSH.

```
[ssh]

enabled  = true
port      = ssh
filter    = sshd
logpath   = /var/log/auth.log
maxretry  = 6

[dropbear]
```

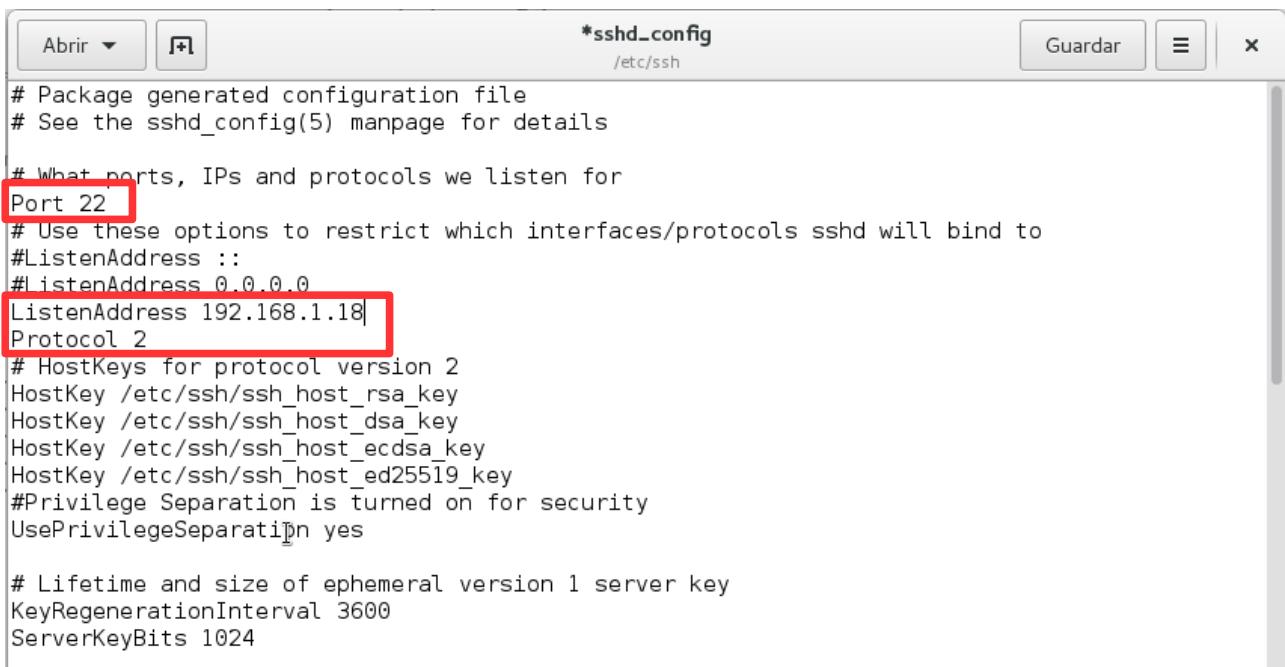
La instalación y configuracion mediante Ansible será la siguiente; instalar el paquete y sustituir el fichero de configuración por uno ya configurado. Por último, reiniciar el servicio.

```
79 ##### FAIL2BAN #####
80
81
82 - name: Instalar Fail2ban
83   apt: pkg=fail2ban state=latest
84
85 - name: Configurar Fail2ban
86   template: src=template/jail.conf dest=/etc/fail2ban/ owner=root mode=0640 backup=yes
87
88 - name: Reinicar servicio Fail2ban
89   service: name=fail2ban state=restarted enabled=yes
```

7. Servidor SSH

Aunque el servidor SSH deberá estar instalado para poder conectar Ansible desde el nodo orquestador y poder desplegar la maqueta de Ansible, será necesario configurar el fichero de configuración de SSH ubicado en `/etc/sshd` durante la instalación automatizada de paquetes.

Para configurar el servidor debe indicarse las direcciones donde el servicio debe responder. En este caso serán conexiones ligadas a la dirección 192.168.1.18, a través del puerto 22 y utilizando la versión 2 del protocolo SSH:



```
# Package generated configuration file
# See the sshd_config(5) manpage for details

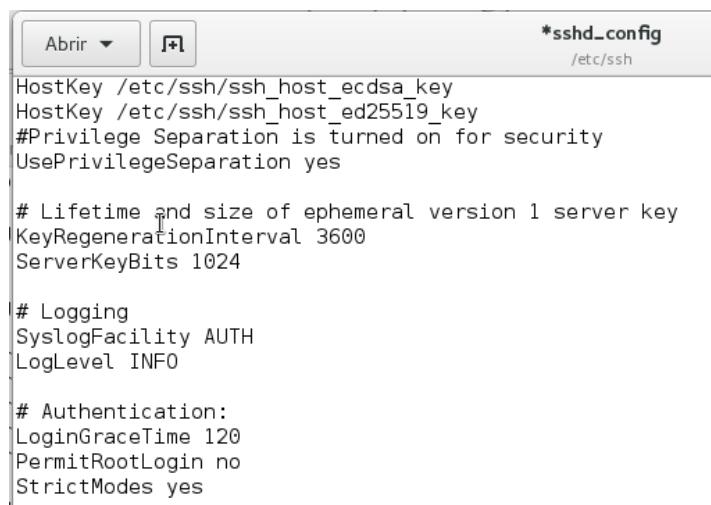
# What ports, IPs and protocols we listen for
Port 22
# Use these options to restrict which interfaces/protocols sshd will bind to
#ListenAddress ::

#ListenAddress 0.0.0.0
ListenAddress 192.168.1.18
Protocol 2

# HostKeys for protocol version 2
HostKey /etc/ssh/ssh_host_rsa_key
HostKey /etc/ssh/ssh_host_dsa_key
HostKey /etc/ssh/ssh_host_ecdsa_key
HostKey /etc/ssh/ssh_host_ed25519_key
#Privilege Separation is turned on for security
UsePrivilegeSeparation yes

# Lifetime and size of ephemeral version 1 server key
KeyRegenerationInterval 3600
ServerKeyBits 1024
```

Por seguridad, se debe desactivar el login como *root*. Para adquirir los privilegios del *root*, se debe hacer un login usuario normal y, después, adquirir los privilegios de *root*. De este modo, se previene que la contraseña del *root* sea objeto de un ataque.



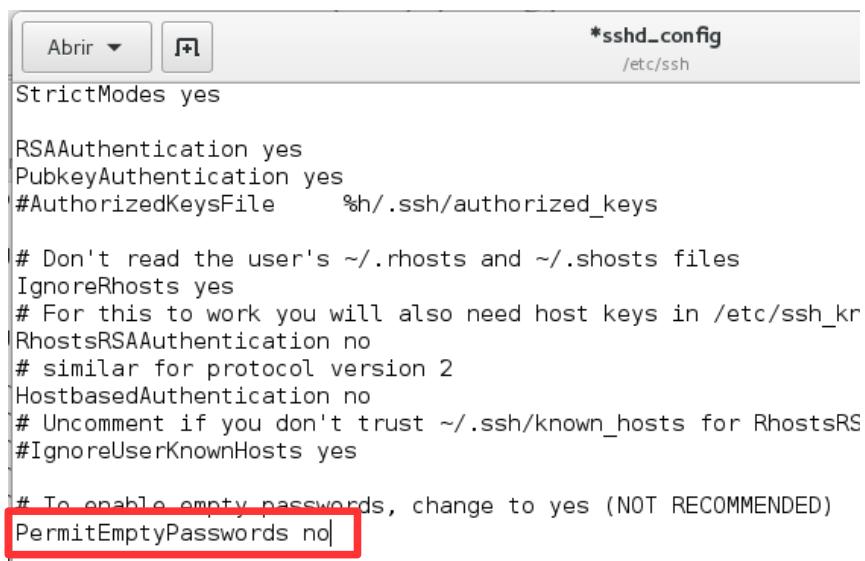
```
HostKey /etc/ssh/ssh_host_ecdsa_key
HostKey /etc/ssh/ssh_host_ed25519_key
#Privilege Separation is turned on for security
UsePrivilegeSeparation yes

# Lifetime and size of ephemeral version 1 server key
KeyRegenerationInterval 3600
ServerKeyBits 1024

# Logging
SyslogFacility AUTH
LogLevel INFO

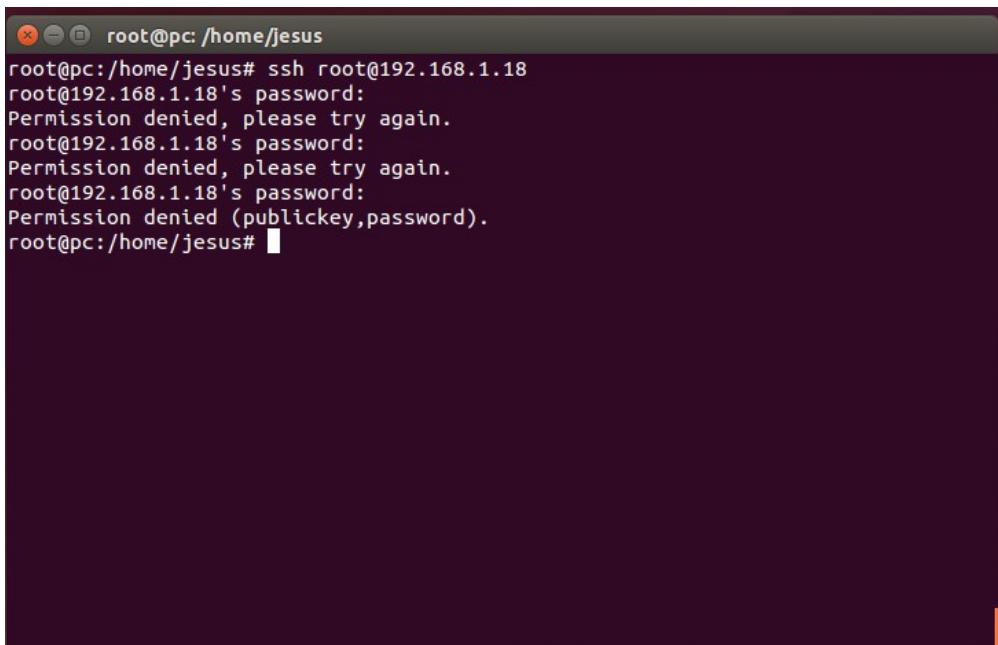
# Authentication:
LoginGraceTime 120
PermitRootLogin no
StrictModes yes
```

También se debe verificar que no sean permitidos los logins con contraseñas vacías.



```
StrictModes yes  
RSAAuthentication yes  
PubkeyAuthentication yes  
#AuthorizedKeysFile      %h/.ssh/authorized_keys  
  
# Don't read the user's ~/.rhosts and ~/.shosts files  
IgnoreRhosts yes  
# For this to work you will also need host keys in /etc/ssh_kr  
RhostsRSAAuthentication no  
# similar for protocol version 2  
HostbasedAuthentication no  
# Uncomment if you don't trust ~/.ssh/known_hosts for RhostsRS  
#IgnoreUserKnownHosts yes  
  
# To enable empty passwords, change to yes (NOT RECOMMENDED)  
PermitEmptyPasswords no
```

Una vez hecho esto es necesario actualizar el servicio. Desde un cliente de la red del DMZ se puede comprobar como conectándose como *root* 3 veces retorna error.



```
root@pc:/home/jesus# ssh root@192.168.1.18  
root@192.168.1.18's password:  
Permission denied, please try again.  
root@192.168.1.18's password:  
Permission denied, please try again.  
root@192.168.1.18's password:  
Permission denied (publickey,password).  
root@pc:/home/jesus#
```

La instalación en Ansible, en este punto, consiste en mandar el fichero de configuración y actualizar el servicio.

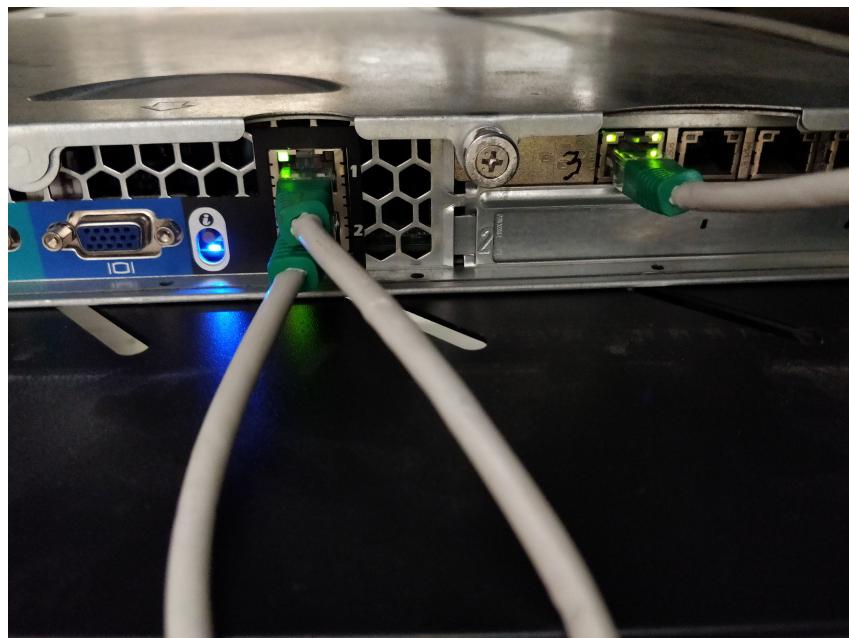
```
122
123 ##### S S H
124
125
126 - name: Copiar configuración servidor SSH
127     template: src=template/sshd_config dest=/etc/ssh/ owner=root mode=0640 backup=yes
128
129 - name: Reinic平 servicio
130     service: name=ssh state=restarted enabled=yes
131
```

8. Configuración de red

EL servidor tendrá 3 interfaces de red necesarios para enrutar los paquetes entre las 3 redes que existirán. Para ello será necesario una configuración de red distinta para separar las 3 redes.

De modo que, tras el servidor, se conectarán las siguientes redes:

- **Boca de red eth0** → Cable de red Ethernet proveniente de un switch conectado al router de Ono, red 192.168.1.0/24.
- **Boca de red eth1** → Cable de red Ethernet proveniente de un switch que conecta los equipos de la red aula, red 192.168.11.0/24.
- **Boca de red eth2** → Cable de red Ethernet proveniente de un switch que conecta el DMZ, red 192.168.12.0/24.



Mediante Ansible es posible configurar los 3 de 5 interfaces de red necesarios de los que dispone el servidor, para ello se reemplazará el fichero de configuración de red ubicado en `/etc/network/interfaces` por uno ya configurado previamente y reiniciando el servicio para que tome los nuevos parámetros.

```
1  # This file describes the network interfaces available on your system
2  # and how to activate them. For more information, see interfaces(5).
3
4  source /etc/network/interfaces/*
5
6  # The loopback network interface
7  auto lo
8  iface lo inet loopback
9
10 auto eth0
11 iface eth0 inet static
12     address 192.168.1.18
13     netmask 255.255.255.0
14     gateway 192.168.1.1
15
16
17 auto eth1
18 iface eth1 inet static
19     address 192.168.11.1
20     netmask 255.255.255.0
21     gateway 192.168.1.1
22
23
24 auto eth2
25 iface eth2 inet static
26     address 192.168.12.1
27     netmask 255.255.255.0
28     gateway 192.168.1.1
```

Para ello, es necesario crear una nueva tarea para el servidor que se llamará *Configurar la red del servidor* y otra para reiniciar el servicio de red.

```
119 ####### R E D #####
120
121
122 - name: Configurar la red del servidor
123   template: src=template/interfaces dest=/etc/network/ owner=root mode=0640 backup=yes
124
125 - name: Reinic平 servicio de red
126   service: name=networking state=restarted enabled=yes
127
```

9. Respaldo mediante Ansible

El respaldo de todo lo anterior redactado y programado en Ansible, como ya se ha detallado, se ejecutará desde de un nodo llamado Orquestador.

El primer paso es clonar [el repositorio donde el proyecto ha sido actualizado](#) y completado. Después entrar en *ansible > respaldo*. Dentro de esa carpeta se encuentra el *playbook* llamado *respaldo*, el cual se necesita ejecutar.

Es importante que si se va a desplegar Ansible en otra dirección IP que no sea la dirección 192.168.1.18 se modifique en el fichero hosts.

Para comenzar se necesita enviar la clave pública de Orquestador al servidor. Primero se crea la clave y se envia por SSH.

```
root@pc:/home/jesus/Proyecto_Jesus_Zamora_Jimenez/ansible/respaldo# ssh-keygen
ssh-keygen  ssh-keyscan
root@pc:/home/jesus/Proyecto_Jesus_Zamora_Jimenez/ansible/respaldo# ssh-keygen
Generating public/private rsa key pair.
Enter file in which to save the key (/root/.ssh/id_rsa):
Created directory '/root/.ssh'.
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /root/.ssh/id_rsa.
Your public key has been saved in /root/.ssh/id_rsa.pub.
The key fingerprint is:
51:0d:89:e2:01:fd:1a:03:d2:b4:b8:17:02:44:57:46 root@pc
The key's randomart image is:
+--[ RSA 2048]--+
|+o +=E .o+      |
| .o.+.+ ... .   |
| o.oo +.        |
|   o ..+ ..     |
|    . . +S       |
|     . .         |
|                 |
|                 |
+-----+
```

```

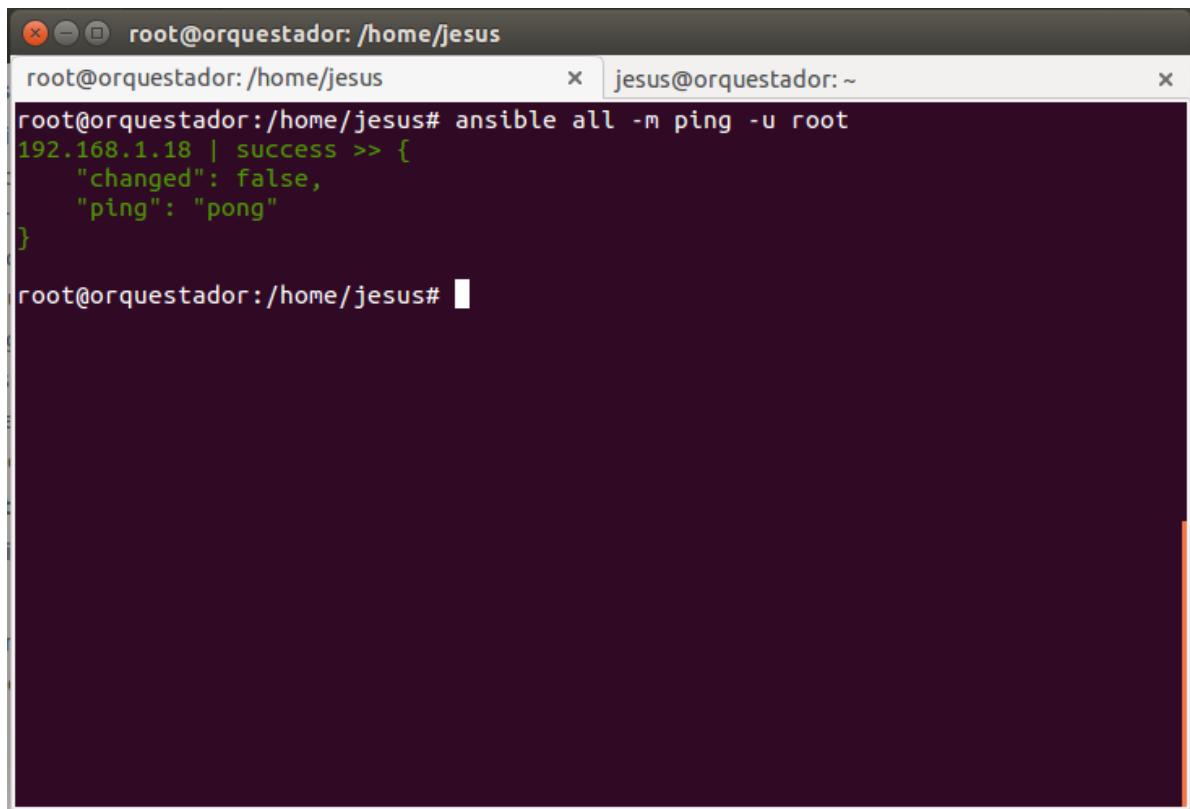
root@pc:/home/jesus/Proyecto_Jesus_Zamora_Jimenez/ansible/respaldo# ssh-copy-id -i ~/.ssh/id_rsa.pub root@192.168.1.18
The authenticity of host '192.168.1.18 (192.168.1.18)' can't be established.
ECDSA key fingerprint is 9f:4d:90:aa:ac:64:eb:c5:26:68:d4:61:26:52:ad:63.
Are you sure you want to continue connecting (yes/no)? yes
/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 prompt
ed now it is to install the new keys
root@192.168.1.18's password:

Number of key(s) added: 1

Now try logging into the machine, with: "ssh 'root@192.168.1.18'"
and check to make sure that only the key(s) you wanted were added.

```

Una vez copiada se verifica que Ansible detecta la IP para poder desplegar la maqueta.



The screenshot shows a terminal window with two tabs. The active tab is titled 'root@orquestador: /home/jesus' and contains the following command and its output:

```

root@orquestador:~# ansible all -m ping -u root
192.168.1.18 | success > {
    "changed": false,
    "ping": "pong"
}

```

Tras esto ya se puede desplegar Ansible sobre el servidor. Para ello se utilizará el comando *ansible-playbook*.

```

root@pc:/home/jesus/Proyecto_Jesus_Zamora_Jimenez/ansible/respaldo# ansible-playbook respaldo.yml

PLAY [respaldo] ****
GATHERING FACTS ****
ok: [192.168.1.18]

TASK: [Actualizar paquetes] ****
ok: [192.168.1.18]

TASK: [Instalar Squid] ****
changed: [192.168.1.18]

TASK: [Configurar squid.conf] ****
changed: [192.168.1.18]

TASK: [Configurar squid.conf] ****
changed: [192.168.1.18]

TASK: [Copiar listas negras para Squid] ****
changed: [192.168.1.18]

TASK: [Reiniciar el servicio Squid] ****
changed: [192.168.1.18]

TASK: [Borrar lista negra antigua] ****
changed: [192.168.1.18]

****

TASK: [Instalar Dansguardian] ****
changed: [192.168.1.18]

TASK: [Configurar Dansguardian] ****
changed: [192.168.1.18]

TASK: [Configurar Dansguardian] ****
changed: [192.168.1.18]

TASK: [Reiniciar el servicio Dansguardian] ****
changed: [192.168.1.18]

TASK: [Lista de paquetes a instalar de Webmin] ****
changed: [192.168.1.18] => (item=perl,libnet-ssleay-perl,openssl,libauthen-pam-perl,libpam-runtime,libio-pty-perl,apt-show-versions,python,apache2)

TASK: [Descargar el paquete Webmin] ****
changed: [192.168.1.18]

TASK: [Instalando Webmin] ****
changed: [192.168.1.18]

TASK: [Reiniciar Webmin] ****
changed: [192.168.1.18]

TASK: [Instalar Fail2ban] ****
changed: [192.168.1.18]

TASK: [Configurar Fail2ban] ****
changed: [192.168.1.18]

TASK: [Reiniciar servicio Fail2ban] ****
changed: [192.168.1.18]

```

```
TASK: [Reiniciar servicio Fail2ban] ****
changed: [192.168.1.18]

TASK: [Instalar Sarg] ****
changed: [192.168.1.18]

TASK: [Configurar Sarg] ****
changed: [192.168.1.18]

TASK: [Programado de los informes] ****
changed: [192.168.1.18]

TASK: [Reiniciando el programado de los informes] ****
changed: [192.168.1.18]

TASK: [Copiar Iptables] ****
changed: [192.168.1.18]

TASK: [Dar permisos para la ejecución al inicio] ****
changed: [192.168.1.18]

TASK: [Configurando Iptables al inicio de cada encendido] ****
changed: [192.168.1.18]

TASK: [Configurar la red del servidor] ****
changed: [192.168.1.18]

TASK: [Reiniciar servicio de red] ****
changed: [192.168.1.18]

TASK: [Copiar configuración servidor SSH] ****
changed: [192.168.1.18]

TASK: [Reiniciar servicio] ****
```

```
PLAY RECAP ****
192.168.1.18 : ok=31    changed=29    unreachable=0    failed=0
```

10. Autoformación.

Para el desarrollo del proyecto he necesitado conocer en profundidad Ansible y sus diferentes módulos, ya que existen varios que permiten una misma función.

Aunque han sido muchas las webs de referencia y vídeos de Youtube que he necesitado para conocer Ansible, he realizado un [curso gratuito emitido por Udemy](#) en el cual he aprendido como funciona Ansible, algunos de sus roles y la puesta en marcha de un playbook.yml.



11. Bibliografía

- https://servidordebian.org/es/squeeze/config/remote_access/ssh_server
- https://servidordebian.org/es/stretch/config/remote_access/ssh_server
- <https://geekytheory.com/programar-tareas-en-linux-usando-crontab>
- <https://www.nerion.es/soporte/tutoriales/tareas-programadas-en-linux/>
- <http://stiven1907.blogspot.com/2013/12/proxy-squid-dansguardian-debian.html>
- <https://www.youtube.com/watch?v=GcYAllkQgFU>
- <https://wiki.squid-cache.org/ConfigExamples/Intercept/SslBumpExplicit>
- <https://codepoets.co.uk/2014/squid-3-4-x-with-ssl-for-debian-wheezy/>
- https://www.systutorials.com/docs/linux/man/8-ssl_crtd/
- http://www.ite.educacion.es/formacion/materiales/85/cd/linux/m6/cortafuegos_iptables.html
- <https://rootear.com/ubuntu-linux/que-son-las-iptables>
- Manual práctico Iptables. <http://es.tldp.org/Manuales-LuCAS/doc-iptables-firewall/doc-iptables-firewall.pdf>
- <https://www.youtube.com/watch?v=sINlwBPeQvE>
- https://www.youtube.com/watch?v=41kc_ETJTNs
- <https://www.youtube.com/watch?v=Wuv0ZPOMLf0>
- <https://www.youtube.com/watch?v=G97sqHIG38w>
- <https://www.youtube.com/watch?v=AxC86Mgf3fs>
- <https://www.youtube.com/watch?v=GcYAllkQgFU&t=697s>
- <https://www.youtube.com/watch?v=CihVgTMFsvc>
- <http://www.squid-cache.org/>
- <https://wiki.squid-cache.org/ConfigExamples/Intercept/LinuxDnat>
- <https://wiki.squid-cache.org/ConfigExamples/Intercept/AtSource>
- https://www.youtube.com/watch?v=tz3IZLB_A8U
- <https://www.udemy.com/ansible-essentials-simplicity-in-automation/>
- <https://www.redeszone.net/2017/01/09/utilizar-cron-crontab-linux-programar-tareas/>
- <https://blog.desdelinux.net/cron-crontab-explicados/>
- <https://geekytheory.com/programar-tareas-en-linux-usando-crontab>
- <https://www.computerhope.com/unix/ucrontab.htm>
- <https://crontab.guru>
- <http://www.webmin.com/deb.html>
- <https://es.wikipedia.org/wiki/Webmin>
- <https://www.tecnoinver.cl/que-es-webmin-y-como-instalarlo-en-ubuntu-14-04-3-Its/>
- <https://github.com/marcinpraczko/ansible-role-squid>
- <https://rooteando.com/pdf/servidor-proxy-squid-sarg-y-dansguardian>

- <https://videlcloud.wordpress.com/2017/07/29/instalacion-y- configuracion-de-squid-con-dansguardian-rhel-7-fedora-26-24-centos-7/>
- <http://blog.deiser.com/primeros-pasos-con-ansible/>

