

Práctica 4 SWAP

El objetivo de esta práctica es, tal como hemos hecho con las páginas que servimos en nuestra granja web, tener nuestra base de datos replicada en varias máquinas virtuales.

En este ejemplo usaremos M1 y M2, pero **no necesariamente deben estar en las mismas máquinas donde tenemos nuestras páginas web**, sino que podemos tener, por ejemplo, un cluster de máquinas en las que se alojan las bases de datos.

1. Creación de la base de datos

En nuestro caso, haremos una base de datos MySQL muy simple, pero que nos será útil para realizar esta práctica. Poseerá una única tabla, sin clave primaria definida.

Los pasos para crear la base de datos son los siguientes:

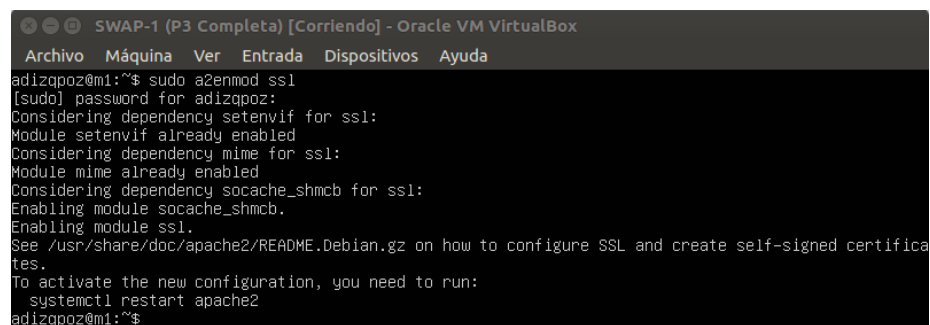
1. Definir el diseño de nuestra base de datos y su paso a tablas correspondiente. Este paso no está en las competencias de esta asignatura, y ya hemos decidido nuestra base de datos.
2. Crear la base de datos.

1.1. Configuración

Para configurar SSL debemos en primer lugar habilitar en el servidor Apache el protocolo SSL, que nos permitirá firmar y cifrar nuestro sitio web. Para ello utilizamos el siguiente comando:

```
sudo a2enmod ssl
```

Y posteriormente reiniciamos Apache para hacer efectivos los cambios.

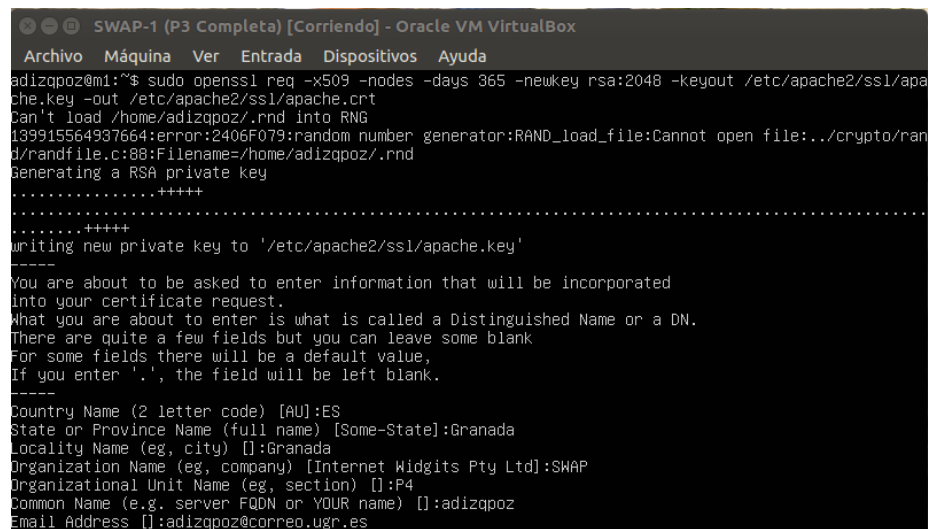


```
SWAP-1 (P3 Completa) [Corriendo] - Oracle VM VirtualBox
Archivo  Máquina  Ver  Entrada  Dispositivos  Ayuda
adizqp0z@m1:~$ sudo a2enmod ssl
[sudo] password for adizqp0z:
Considering dependency setenvif for ssl:
Module setenvif already enabled
Considering dependency mime for ssl:
Module mime already enabled
Considering dependency socache_shmcb for ssl:
Enabling module socache_shmcb.
Enabling module ssl.
See /usr/share/doc/apache2/README.Debian.gz on how to configure SSL and create self-signed certificates.
To activate the new configuration, you need to run:
  systemctl restart apache2
adizqp0z@m1:~$
```

A continuación creamos nuestra clave y certificado para nuestro sitio web. Sin embargo, los navegadores detectarán nuestra firma como insegura, dado que no la ha validado ninguna entidad certificadora. Aún así, por motivos didácticos, crearemos nuestro propio certificado digital con OpenSSL.

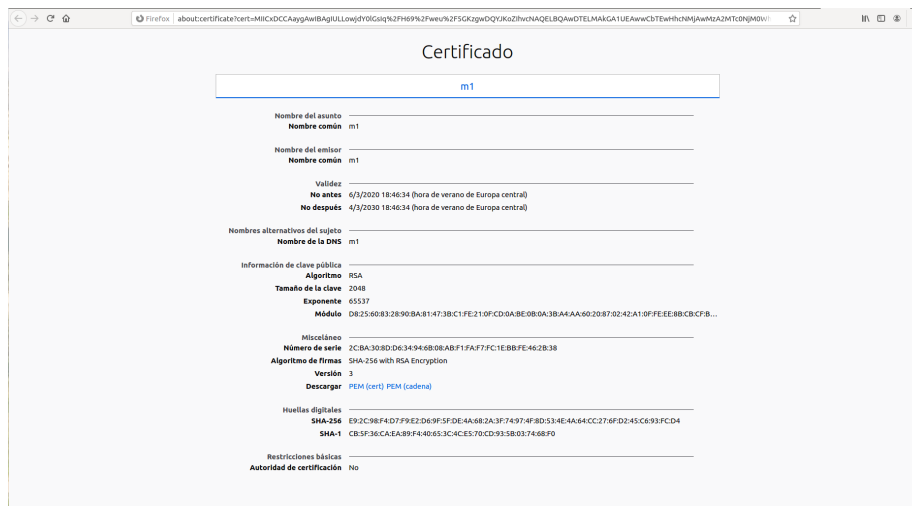
Para ello, en el directorio `/etc/apache2` debemos crear un nuevo subdirectorio llamado `ssl`, en el cual guardaremos la clave y certificado que se generarán. Posteriormente utilizaremos el siguiente comando para generar la clave y certificado:

```
sudo openssl req -x509 -nodes -days 365 -newkey rsa:2048 -keyout /etc/apache2/ssl/apache.key
```



```
SWAP-1 (P3 Completa) [Corriendo] - Oracle VM VirtualBox
Archivo  Máquina  Ver  Entrada  Dispositivos  Ayuda
adizqpoz@m1:~$ sudo openssl req -x509 -nodes -days 365 -newkey rsa:2048 -keyout /etc/apache2/ssl/apache.key -out /etc/apache2/ssl/apache.crt
Can't load /home/adizqpoz/.rnd into RNG
139915564937664:error:2406F079:random number generator:RAND_load_file:Cannot open file:../crypto/rand/randfile.c:88:Filename=/home/adizqpoz/.rnd
Generating a RSA private key
.....+++++
.....+++++
Writing new private key to '/etc/apache2/ssl/apache.key'
-----
You are about to be asked to enter information that will be incorporated
into your certificate request.
What you are about to enter is what is called a Distinguished Name or a DN.
There are quite a few fields but you can leave some blank
For some fields there will be a default value,
If you enter '.', the field will be left blank.
-----
Country Name (2 letter code) [AU]:ES
State or Province Name (full name) [Some-State]:Granada
Locality Name (eg, city) []:Granada
Organization Name (eg, company) [Internet Widgits Pty Ltd]:SWAP
Organizational Unit Name (eg, section) []:P4
Common Name (e.g. server FQDN or YOUR name) []:adizqpoz
Email Address []:adizqpoz@correo.ugr.es
```

Tras rellenar el formulario con los datos pertinentes configuramos Apache para indicarle dónde están el certificado y la clave que se han de utilizar para servir una página HTTPS, reiniciamos Apache y comprobamos que podemos servir una de nuestras páginas mediante el protocolo citado:



Para terminar, copiaremos mediante el comando SCP utilizado en la práctica 2 para copiar las claves de nuestro certificado a M2 y a M3, configuramos Apache en M2 de forma idéntica a M1, y configuramos M3 para que redireccione las peticiones HTTPS de igual forma que hicimos con HTTP en la práctica 3, pero indicando que utilizamos el protocolo SSL, que escuchamos el puerto 443, puerto por defecto en el que se escuchan las peticiones HTTPS, y le indicamos la ruta de los archivos recibidos por SCP.

Por último, comprobamos que nuestra infraestructura puede servir páginas HTTPS de forma satisfactoria:

```

aizpoz@aizpoz-HP-Notebook: ~
aizpoz@aizpoz-HP-Notebook:~$ curl -k https://192.168.56.103/ejemplo.html
<html>
  <head>
    <meta charset="utf-8">
  </head>
  <body>
    <h1>Web de ejemplo de adizqpoz para SWAP</h1>
    <p>Soy la máquina 1</p>
  </body>
</html>
aizpoz@aizpoz-HP-Notebook:~$ curl -k https://192.168.56.103/ejemplo.html
<html>
  <head>
    <meta charset="utf-8">
  </head>
  <body>
    <h1>Web de ejemplo de adizqpoz para SWAP</h1>
    <p>Soy la máquina 2</p>
  </body>
</html>
aizpoz@aizpoz-HP-Notebook:~$

```

2. iptables

iptables es una herramienta de firewall nativa de los sistemas Linux que nos permite controlar qué tipo de tráfico pueden aceptar nuestras máquinas servidoras. Esta herramienta se configura mediante una serie de reglas que se aplican a nuestro cortafuegos.

En primer lugar limpiaremos todas las reglas que tengamos en iptables para asegurarnos de que las reglas que vamos a añadir no interactúan con otras reglas que no controlamos en este momento. Para ello utilizamos las siguientes reglas:

```
iptables -F
iptables -X
iptables -t nat -F
iptables -t nat -X
iptables -t mangle -F
iptables -t mangle -X
iptables -P INPUT ACCEPT
iptables -P OUTPUT ACCEPT
```

Con ello tenemos una configuración por defecto que acepta todas las peticiones.

```
adizqpoz@m1:~$ sudo iptables -L -n -v
Chain INPUT (policy ACCEPT 0 packets, 0 bytes)
 pkts bytes target    prot opt in     out     source                   destination

Chain FORWARD (policy ACCEPT 0 packets, 0 bytes)
 pkts bytes target    prot opt in     out     source                   destination

Chain OUTPUT (policy ACCEPT 0 packets, 0 bytes)
 pkts bytes target    prot opt in     out     source                   destination
adizqpoz@m1:~$
```

Posteriormente denegamos todo tráfico de información para posteriormente definir las excepciones de tráfico que permitiremos que entre y salga de nuestro servidor:

```
iptables -P INPUT DROP
iptables -P OUTPUT DROP
iptables -P FORWARD DROP
```

```
adizqpoz@m1:~$ sudo iptables -P INPUT DROP
adizqpoz@m1:~$ sudo iptables -P OUTPUT DROP
adizqpoz@m1:~$ sudo iptables -P FORWARD DROP
adizqpoz@m1:~$ sudo iptables -L -n -v
Chain INPUT (policy DROP 0 packets, 0 bytes)
 pkts bytes target    prot opt in     out     source                   destination

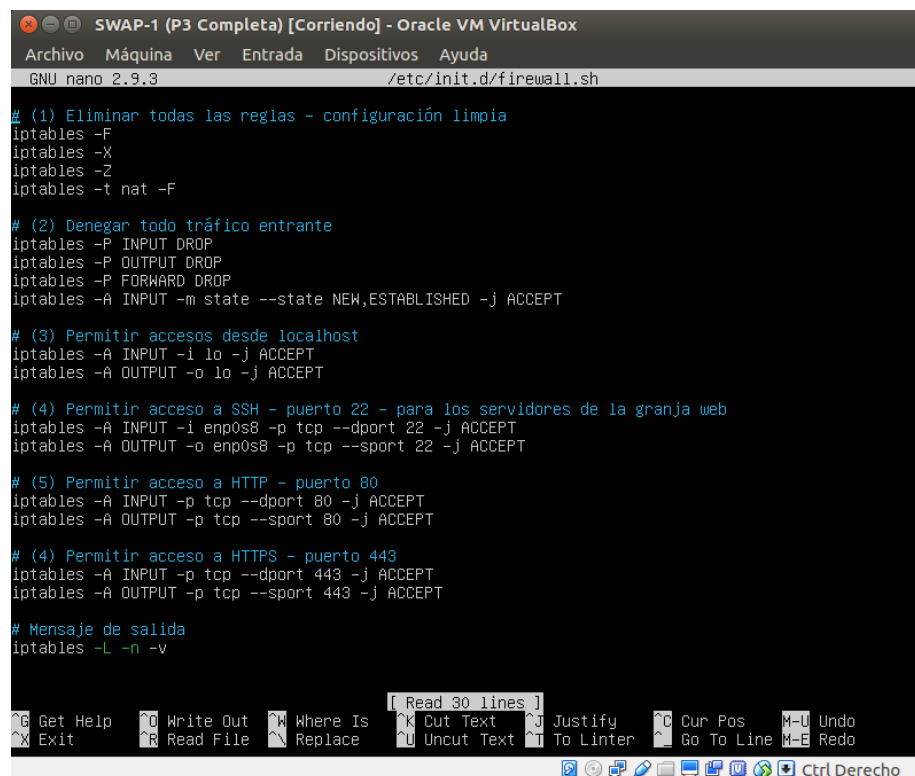
Chain FORWARD (policy DROP 0 packets, 0 bytes)
 pkts bytes target    prot opt in     out     source                   destination

Chain OUTPUT (policy DROP 0 packets, 0 bytes)
 pkts bytes target    prot opt in     out     source                   destination
adizqpoz@m1:~$
```

A continuación, para permitir el tráfico de los servicios que utilizan nuestros servidores debemos definir un conjunto de reglas. En concreto vamos a permitir:

- Accesos desde localhost
- Acceso y salida de SSH desde los equipos de nuestra granja web
- Acceso y salida de peticiones HTTP
- Acceso y salida de peticiones HTTPS

Además, crearemos un script que nos permita tener el firewall configurado desde el momento en el que encendemos nuestros servidores.



```
SWAP-1 (P3 Completa) [Corriendo] - Oracle VM VirtualBox
Archivo  Máquina  Ver  Entrada  Dispositivos  Ayuda
GNU nano 2.9.3 /etc/init.d/firewall.sh

# (1) Eliminar todas las reglas - configuración limpia
iptables -F
iptables -X
iptables -Z
iptables -t nat -F

# (2) Denegar todo tráfico entrante
iptables -P INPUT DROP
iptables -P OUTPUT DROP
iptables -P FORWARD DROP
iptables -A INPUT -m state --state NEW,ESTABLISHED -j ACCEPT

# (3) Permitir accesos desde localhost
iptables -A INPUT -i lo -j ACCEPT
iptables -A OUTPUT -o lo -j ACCEPT

# (4) Permitir acceso a SSH - puerto 22 - para los servidores de la granja web
iptables -A INPUT -i enp0s8 -p tcp --dport 22 -j ACCEPT
iptables -A OUTPUT -o enp0s8 -p tcp --sport 22 -j ACCEPT

# (5) Permitir acceso a HTTP - puerto 80
iptables -A INPUT -p tcp --dport 80 -j ACCEPT
iptables -A OUTPUT -p tcp --sport 80 -j ACCEPT

# (4) Permitir acceso a HTTPS - puerto 443
iptables -A INPUT -p tcp --dport 443 -j ACCEPT
iptables -A OUTPUT -p tcp --sport 443 -j ACCEPT

# Mensaje de salida
iptables -L -n -v
```

El estado en el que queda nuestro firewall es el siguiente:

```

adizqpoz@ml:~$ sudo iptables -L -n -v
Chain INPUT (policy DROP 0 packets, 0 bytes)
 pkts bytes target    prot opt in     out     source            destination
  0      0 ACCEPT    all  --  lo     *       0.0.0.0/0         0.0.0.0/0
  0      0 ACCEPT    tcp  --  enp0s8 *       0.0.0.0/0         0.0.0.0/0          tcp dpt:22
  0      0 ACCEPT    tcp  --  *      *       0.0.0.0/0         0.0.0.0/0          tcp dpt:80
  0      0 ACCEPT    tcp  --  *      *       0.0.0.0/0         0.0.0.0/0          tcp dpt:443

Chain FORWARD (policy DROP 0 packets, 0 bytes)
 pkts bytes target    prot opt in     out     source            destination

Chain OUTPUT (policy DROP 0 packets, 0 bytes)
 pkts bytes target    prot opt in     out     source            destination
  0      0 ACCEPT    all  --  *      lo     0.0.0.0/0         0.0.0.0/0
  0      0 ACCEPT    tcp  --  *      enp0s8 0.0.0.0/0         0.0.0.0/0          tcp spt:22
  0      0 ACCEPT    tcp  --  *      *       0.0.0.0/0         0.0.0.0/0          tcp spt:80
  0      0 ACCEPT    tcp  --  *      *       0.0.0.0/0         0.0.0.0/0          tcp spt:443
adizqpoz@ml:~$

```

Ahora haremos que nuestro script se ejecute cada vez que se reinicie nuestra máquina. Para ello utilizaremos cron, la herramienta que utilizamos en la práctica 2 para automatizar la replicación de contenidos de una máquina maestra a una máquina esclava.

Para ello existe una serie de parámetros que podemos utilizar en las reglas de cron en lugar de los parámetros temporales que conocemos para ejecutar las mismas tras un determinado evento.

Para nuestro caso la regla que utilizamos es la siguiente:

```
@reboot root /home/adizqpoz/fiirewall.sh
```

Y posteriormente reiniciamos el demonio cron, reiniciamos la máquina y comprobamos cómo tras iniciarla las reglas están definidas en nuestro cortafuegos.

Por último, comprobaremos que el firewall nos permite realizar las acciones que deseamos:

```

aizpoz@aizpoz-HP-Notebook:~$ ssh adizqpoz@192.168.56.101
adizqpoz@192.168.56.101's password:
Welcome to Ubuntu 18.04.4 LTS (GNU/Linux 4.15.0-96-generic x86_64)

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

System information as of Sun May  3 16:36:20 UTC 2020

System load:  0.09          Processes:            98
Usage of /:   42.6% of 9.7GB Users logged in:          1
Memory usage: 60%          IP address for enp0s3: 10.0.2.15
Swap usage:   2%           IP address for enp0s8: 192.168.56.101

 * Ubuntu 20.04 LTS is out, raising the bar on performance, security,
   and optimisation for Intel, AMD, Nvidia, ARM64 and Z15 as well as
   AWS, Azure and Google Cloud.

   https://ubuntu.com/blog/ubuntu-20-04-lts-arrives

 * Canonical Livepatch is available for installation.
   - Reduce system reboots and improve kernel security. Activate at:
     https://ubuntu.com/livepatch

Pueden actualizarse 31 paquetes.
8 actualizaciones son de seguridad.

Last login: Sun May  3 15:32:23 2020
adizqpoz@m1:~$ exit
logout
Connection to 192.168.56.101 closed.
aizpoz@aizpoz-HP-Notebook:~$

```

```

aizpoz@aizpoz-HP-Notebook:~$ curl 192.168.56.101/ejemplo.html
<html>
  <head>
    <meta charset="utf-8">
  </head>
  <body>
    <h1>Web de ejemplo de adizqpoz para SWAP</h1>
    <p>Soy la máquina 1</p>
  </body>
</html>
aizpoz@aizpoz-HP-Notebook:~$ curl -k https://192.168.56.101/ejemplo.html
<html>
  <head>
    <meta charset="utf-8">
  </head>
  <body>
    <h1>Web de ejemplo de adizqpoz para SWAP</h1>
    <p>Soy la máquina 1</p>
  </body>
</html>
aizpoz@aizpoz-HP-Notebook:~$

```

En este punto concluimos la práctica, a pesar de que es posible profundizar algo más en el uso del cortafuegos para toda nuestra granja web, de forma que nuestras máquinas finales sólo reciban peticiones web del balanceador, y el balanceador responda a las peticiones web, y redirijan el tráfico a las máquinas servidoras finales.

Autor: Adrián Izquierdo Pozo

Si desea ver el archivo Markdown puede verlo en mi repositorio de Github