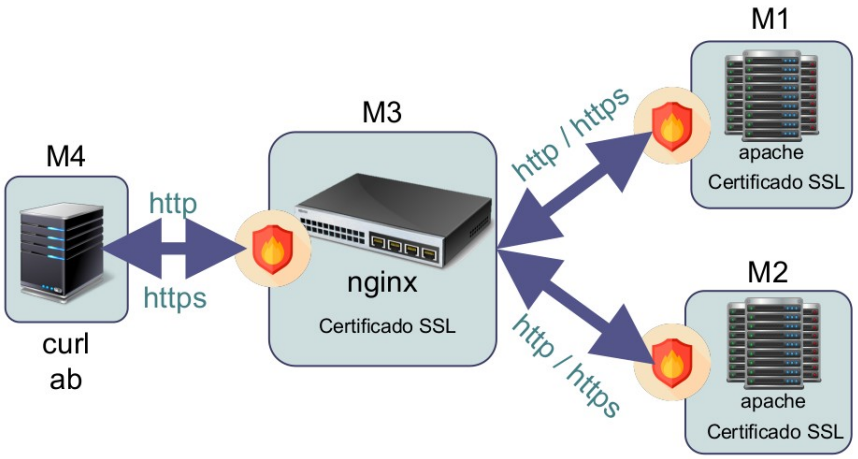
ETSIIT
Escuela Técnica Superior
de Ingenierías Informática
y de Telecomunicación

UNIVERSIDAD
DE GRANADA

SERVIDORES WEB DE ALTAS PRESTACIONES

PRÁCTICA 4: ASEGURAR LA GRANJA WEB



David Armenteros Soto

ÍNDICE

1. **Introducción: Objetivos y cuestiones a resolver**
2. **Apache con certificado autofirmado SSL**
3. **Nginx como balanceador para peticiones HTTPS**
4. **Configuración del cortafuegos con IPTABLES**
   4.1 Configuración del cortafuegos al arranque
5. **Crear, instalar y configurar certificado SSL en Apache y Nginx con CERTBOT**

1. Introducción: Objetivos y cuestiones a resolver

El objetivo principal de la práctica es configurar aspectos relativos a la seguridad de la granja web. Para ello, llevaremos a cabo las siguientes tareas:

- Crear e instalar en las máquinas servidoras y en el balanceador un **certificado SSL** autofirmado para configurar el acceso **HTTPS** al servidor. Comprobaremos que el servidor acepta tanto tráfico HTTP y HTTPS.
- Configurar y documentar las reglas del cortafuegos con **IPTABLES**, a través de un script en cada máquina.
- Hacer que la configuración del cortafuegos se ejecute al **arranque** del sistema en todas las máquinas.
- Crear, instalar y configurar un certificado SSL con **Certbot** .

2. Apache con certificado autofirmado SSL

SSL (Secure Sockets Layer o capa de conexión segura) es un estándar de seguridad global que permite la transferencia de datos cifrados entre un navegador y un servidor web. Para establecer esta conexión segura, necesitamos un **certificado SSL.**

Para generar un certificado SSL autofirmado en la máquina M1 con Ubuntu Server debemos:

- Activar el módulo SSL y crear el directorio de los certificados.

```
daarso98@m1-daarso98:~$ sudo a2enmod ssl
[sudo] password for daarso98:
Sorry, try again.
[sudo] password for daarso98:
sudo: 1 incorrect password attempt
daarso98@m1-daarso98:~$ sudo a2enmod ssl
[sudo] password for daarso98:
Considering dependency setenvif for ssl:
Module setenvif already enabled
Considering dependency mime for ssl:
Module mime already enabled
Considering dependency socache_shmcb for ssl:
Module socache_shmcb already enabled
Module ssl already enabled
daarso98@m1-daarso98:~$ sudo service apache2 restart
daarso98@m1-daarso98:~$ sudo mkdir /etc/apache2/ssl
```

- Generar un certificado SSL autofirmado y configurar el certificado de dominio con los datos.

```
daarso98@m1-daarso98:~$ sudo openssl req -x509 -nodes -days 365 -newkey rsa:2048 -keyout /etc/apache2/ssl/apache_daarso98.key -out /etc/apache2/ssl/apache_daarso98.cr
Can't load /home/daarso98/.rnd into RNG
140398183494080:error:2406F079:random number generator:RAND_load_file:Cannot open file:../crypto/rand/randfile.c:88:Filename=/home/daarso98/.rnd
Generating a RSA private key
.................................+++++
.............+++++
writing new private key to '/etc/apache2/ssl/apache_daarso98.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) []:daarso98
Email Address []:daarso98@correo.ugr.es
```

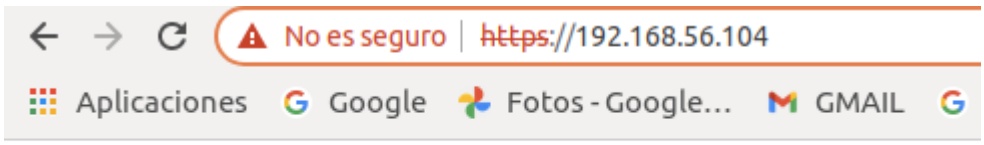
- Configurar apache con la ruta de los certificados. Para ello, nos dirigimos al archivo **/etc/apache2/sites-available/default-ssl.conf** y agregamos la ruta de los certificados.

```
 GNU nano 2.9.3            /etc/apache2/sites-available/default-ssl.conf            Modified

                # For most configuration files from conf-available/, which are
                # enabled or disabled at a global level, it is possible to
                # include a line for only one particular virtual host. For example the
                # following line enables the CGI configuration for this host only
                # after it has been globally disabled with "a2disconf".
                #Include conf-available/serve-cgi-bin.conf

                #   SSL Engine Switch:
                #   Enable/Disable SSL for this virtual host.
                SSLEngine on

                #   A self-signed (snakeoil) certificate can be created by installing
                #   the ssl-cert package. See
                #   /usr/share/doc/apache2/README.Debian.gz for more info.
                #   If both key and certificate are stored in the same file, only the
                #   SSLCertificateFile directive is needed.
                SSLCertificateFile      /etc/apache2/ssl/apache_daarso98.crt
                SSLCertificateKeyFile   /etc/apache2/ssl/apache_daarso98.key
```

- Activamos el sitio **default-ssl**

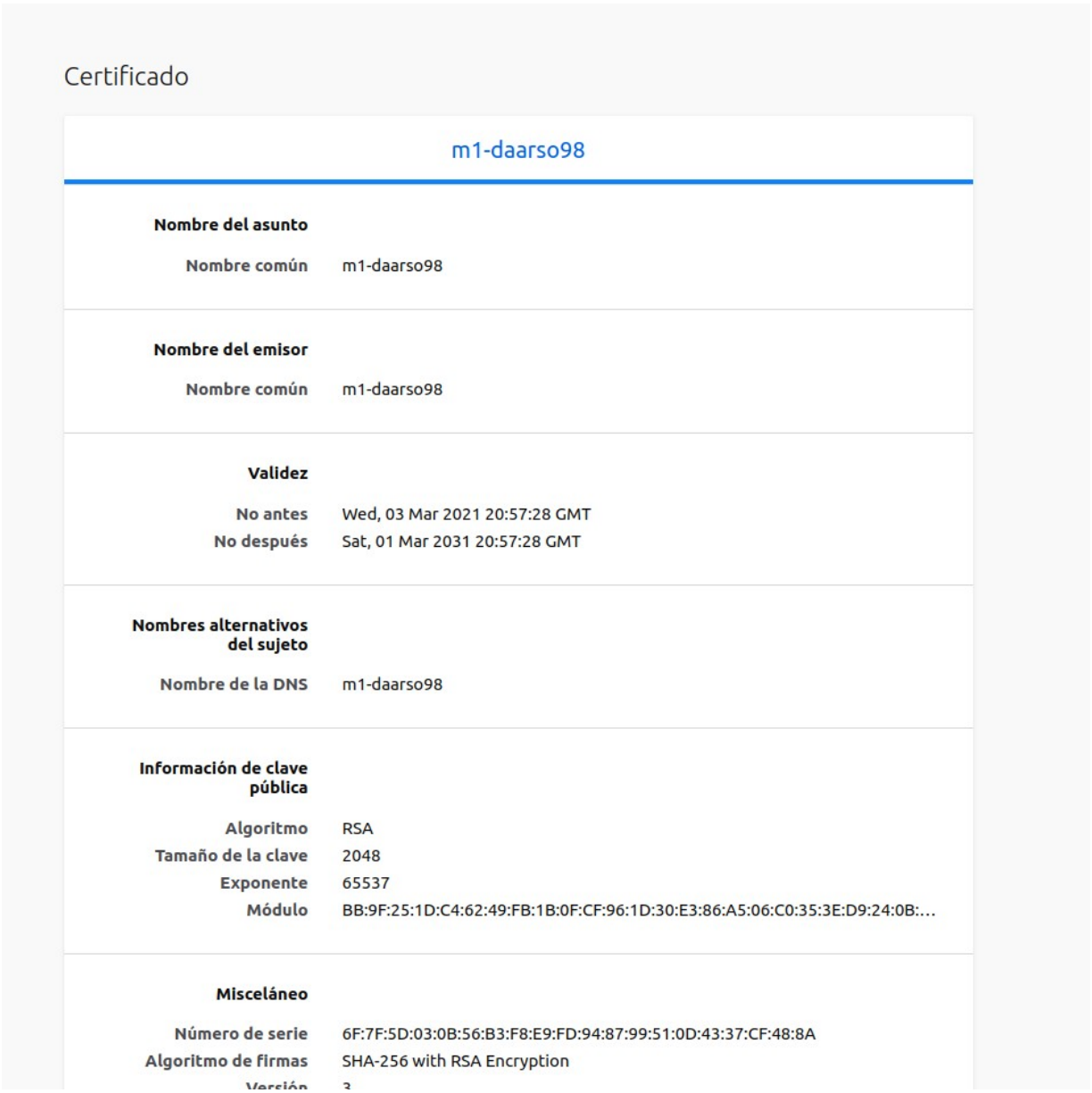
```
daarso98@m1-daarso98:~$ sudo a2ensite default-ssl.conf
Site default-ssl already enabled
daarso98@m1-daarso98:~$ sudo systemctl reload apache2.service
```

Si hemos seguido los pasos correctamente, nuestro servidor debería estar listo para servir peticiones HTTPS como se muestra a continuación:



ESTA ES LA MÃˆQUINA 1 DE DAARSO98 192.168.56.104

```
(base) daarso@daarso-GP62-7RE:~$ curl -k https://192.168.56.104/index.html
<HTML>
  <BODY>
        ESTA ES LA MÁQUINA 1 DE DAARSO98 192.168.56.104
   </BODY>
</HTML>
```



Certificado

m1-daarso98

Nombre del asunto

Nombre común: m1-daarso98

Nombre del emisor

Nombre común: m1-daarso98

Validez

No antes: Wed, 03 Mar 2021 20:57:28 GMT
No después: Sat, 01 Mar 2031 20:57:28 GMT

Nombres alternativos del sujeto

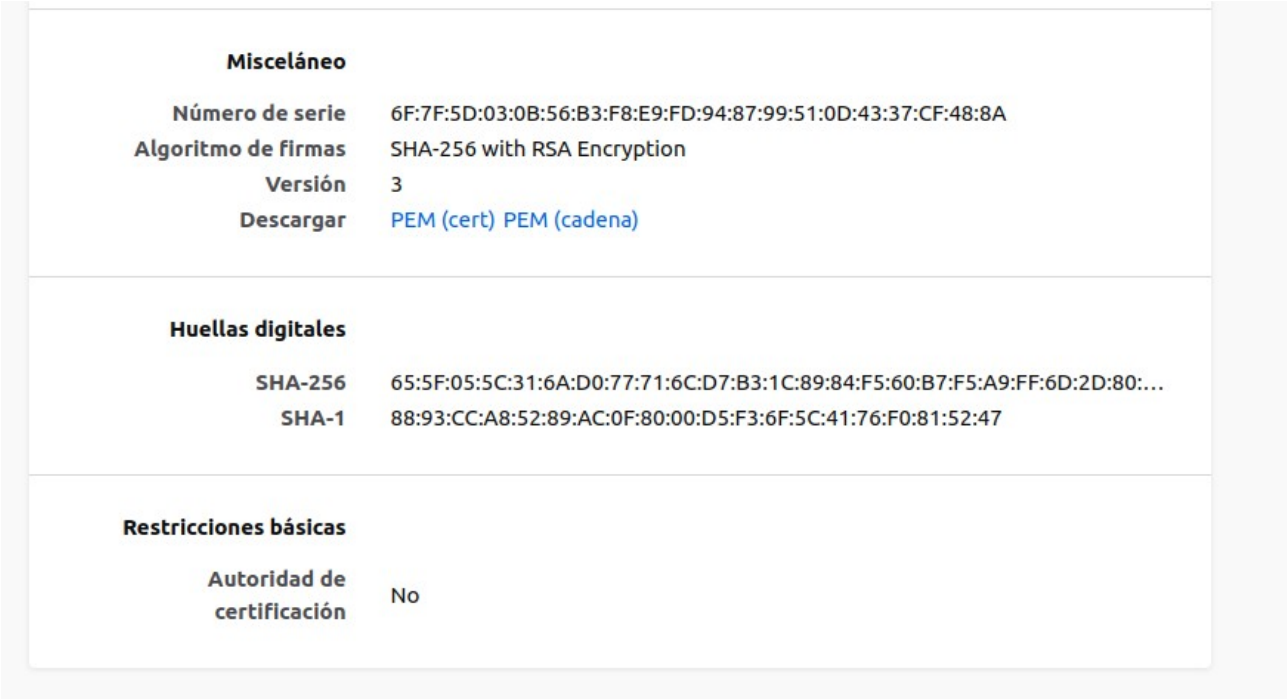
Nombre de la DNS: m1-daarso98

Información de clave pública

Algoritmo: RSA
Tamaño de la clave: 2048
Exponente: 65537
Módulo: BB:9F:25:1D:C4:62:49:FB:1B:0F:CF:96:1D:30:E3:86:A5:06:C0:35:3E:D9:24:0B:...

Misceláneo

Número de serie: 6F:7F:5D:03:0B:56:B3:F8:E9:FD:94:87:99:51:0D:43:37:CF:48:8A
Algoritmo de firmas: SHA-256 with RSA Encryption
Versión: 3



Misceláneo	
Número de serie	6F:7F:5D:03:0B:56:B3:F8:E9:FD:94:87:99:51:0D:43:37:CF:48:8A
Algoritmo de firmas	SHA-256 with RSA Encryption
Versión	3
Descargar	PEM (cert) PEM (cadena)
Huellas digitales	
SHA-256	65:5F:05:5C:31:6A:D0:77:71:6C:D7:B3:1C:89:84:F5:60:B7:F5:A9:FF:6D:2D:80:...
SHA-1	88:93:CC:A8:52:89:AC:0F:80:00:D5:F3:6F:5C:41:76:F0:81:52:47
Restricciones básicas	
Autoridad de certificación	No

Para poder visualizar el certificado, lo podemos hacer desde el navegador en el menú de opciones avanzadas > ver certificado

Al igual que hemos hecho con M1, queremos que la máquina M2 acepte tráfico HTTPS . Para hacer esto, copiaremos la pareja de archivos **.crt** y **.key** a esta máquina. No debemos generar más certificados sino que los archivos generados en el paso anterior los debemos copiar usando **scp**.

```
daarso98@m1-daarso98:/etc/apache2/ssl$ sudo scp -P 2222 -i ~/.ssh/mykey apache_daarso98.crt daarso98@192.168.56.105:/home/daarso98/apache_daarso98.crt
apache_daarso98.crt                           100% 1428   113.2KB/s   00:00
daarso98@m1-daarso98:/etc/apache2/ssl$ sudo scp -P 2222 -i ~/.ssh/mykey apache_daarso98.key daarso98@192.168.56.105:/home/daarso98/apache_daarso98.key
apache_daarso98.key                           100% 1704     3.2MB/s   00:00
```

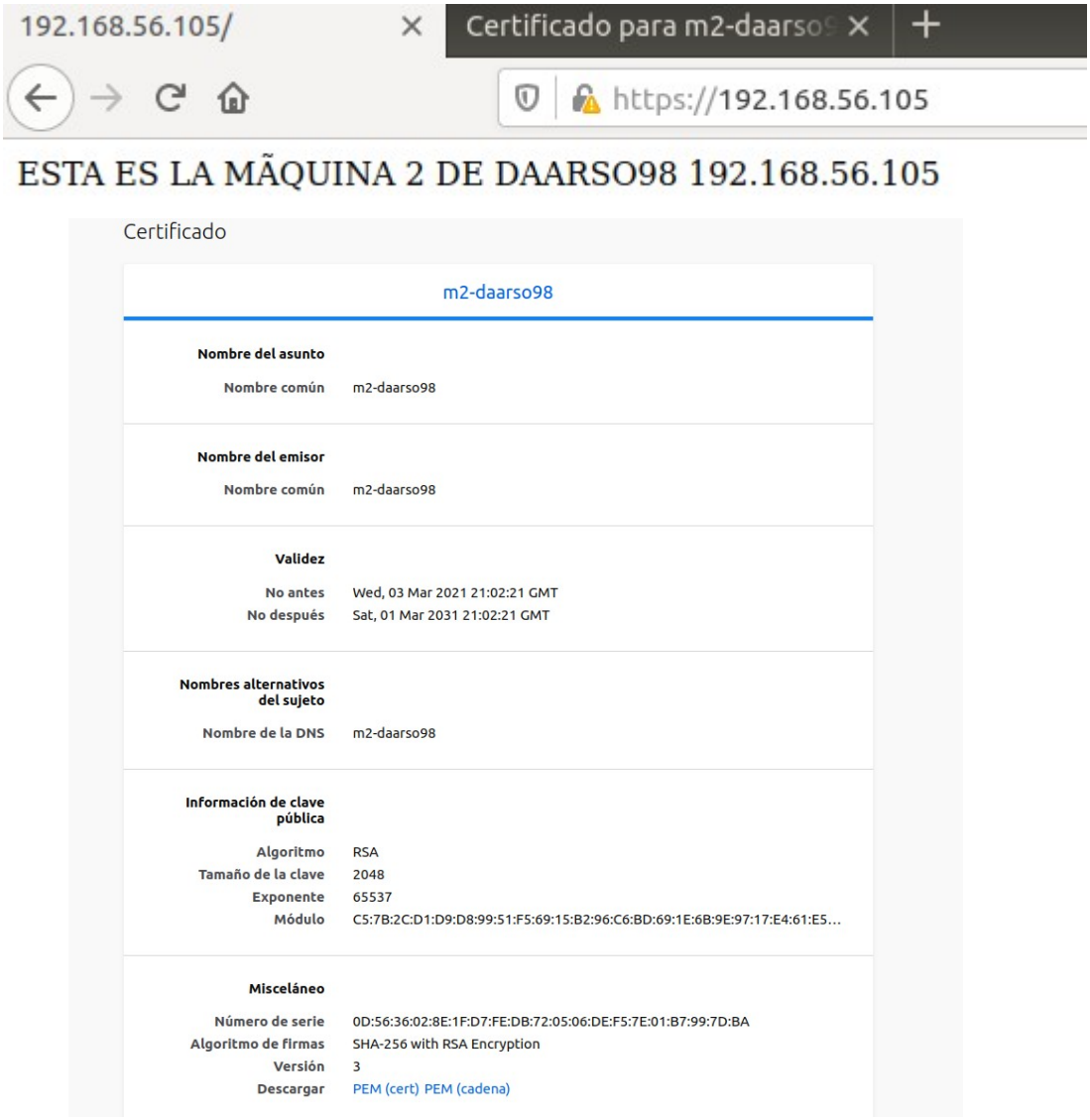
Realizamos los mismos pasos que en la máquina M1, debemos crear el directorio **/etc/apache2/ssl** , configurar **default-ssl.conf**, activar el sitio y reiniciar apache.

```
daarso98@m2-daarso98:~$ sudo mkdir /etc/apache2/ssl
[sudo] password for daarso98:
daarso98@m2-daarso98:~$ ls
Hola.sh    apache_daarso98.crt  archivo.tgz   cookie.txt  imagen.png  logo3w.png
Hola.txt   apache_daarso98.key  archivo2.tgz  directorio  info.txt    prueba
daarso98@m2-daarso98:~$ sudo mv /home/daarso98/apache* /etc/apache2/ssl
daarso98@m2-daarso98:~$ sudo a2enmod ssl & sudo service apache2 restart
[1] 20565
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
elf-signed certificates.
To activate the new configuration, you need to run:
  systemctl restart apache2
[1]+  Done                    sudo a2enmod ssl
```

```
GNU nano 2.9.3                                    /etc/apache2/sites-available/default-ssl.conf

        #Include conf-available/serve-cgi-bin.conf

        #   SSL Engine Switch:
        #   Enable/Disable SSL for this virtual host.
        SSLEngine on

        #   A self-signed (snakeoil) certificate can be created by installing
        #   the ssl-cert package. See
        #   /usr/share/doc/apache2/README.Debian.gz for more info.
        #   If both key and certificate are stored in the same file, only the
        #   SSLCertificateFile directive is needed.
        SSLCertificateFile      /etc/apache2/ssl/apache_daarso98.crt
        SSLCertificateKeyFile   /etc/apache2/ssl/apache_daarso98.key

        SSLCertificateFile      /etc/ssl/certs/ssl-cert-snakeoil.pem
        SSLCertificateKeyFile /etc/ssl/private/ssl-cert-snakeoil.key
```

```
daarso98@m2-daarso98:~$ sudo a2ensite default-ssl.conf
Site default-ssl already enabled
daarso98@m2-daarso98:~$ sudo systemctl restart apache2.service
```

Como podemos comprobar, la máquina M2 esta lista para servir peticiones HTTPS



■ Opciones avanzadas para certificados autofirmados SSL

- Podemos ver información detallada del certificado que acabamos de crear

```
daarso98@m1-daarso98:/etc/apache2/ssl$ openssl x509 -text -noout -in apache_daarso98.crt
Certificate:
    Data:
        Version: 3 (0x2)
        Serial Number:
            04:60:0f:ac:04:93:8a:4f:1e:7e:8b:f9:9d:5e:77:73:d2:fc:a2:eb
        Signature Algorithm: sha256WithRSAEncryption
        Issuer: C = ES, ST = Granada, L = Granada, O = SWAP, OU = P4, CN = daarso98, emailAddress = daarso98@correo.ugr.es
        Validity
            Not Before: Apr 21 11:41:30 2021 GMT
            Not After : Apr 21 11:41:30 2022 GMT
        Subject: C = ES, ST = Granada, L = Granada, O = SWAP, OU = P4, CN = daarso98, emailAddress = daarso98@correo.ugr.es
        Subject Public Key Info:
            Public Key Algorithm: rsaEncryption
                RSA Public-Key: (2048 bit)
                Modulus:
                    00:ad:55:9d:65:eb:9d:ec:a4:a3:d7:a9:08:db:7f:
                    1c:54:f3:2e:8c:80:22:ab:54:e4:1c:72:9b:25:df:
                    b9:85:e2:64:b5:fb:dd:bd:4a:79:d1:5d:5f:68:92:
                    ff:f0:1b:ae:d9:ac:a5:f8:75:46:81:cf:7a:e7:23:
                    3c:9a:86:36:be:52:20:e8:57:71:6a:d6:6d:a3:2f:
                    f0:29:e3:19:f2:42:63:4d:db:be:a5:95:66:3d:55:
                    ab:30:d0:0b:1a:1b:2f:d6:d2:db:4f:83:75:f8:f0:
                    15:85:10:7f:4a:ee:d2:5a:ca:73:54:62:05:ff:c8:
                    0e:b3:85:5f:68:92:ad:fd:c5:a4:9e:f3:af:79:07:
                    ce:e1:cb:cc:dd:93:4b:3a:41:dd:9d:5e:76:88:5d:
                    0e:2d:c8:b7:19:d5:e6:c2:90:0a:ce:51:ce:6b:87:
                    44:18:ca:b6:30:a2:b5:18:93:ee:d5:c8:62:61:ab:
                    17:64:77:7c:8e:c9:55:89:41:b7:8d:f9:4b:f3:fa:
                    f2:71:50:90:c9:dd:a0:0f:ba:66:32:e0:03:ec:36:
                    51:61:87:e0:94:9e:7d:c8:0e:2b:67:30:b5:77:a8:
                    d0:7d:74:af:db:65:2d:1a:13:40:40:f8:91:ec:8e:
                    92:33:fa:0a:9e:04:a5:d8:4a:7a:c1:c0:b6:6e:d3:
                    3f:63
                Exponent: 65537 (0x10001)
        X509v3 extensions:
            X509v3 Subject Key Identifier:
                CC:D3:15:C3:82:EC:1D:3B:DB:77:33:CB:49:07:11:AE:04:4B:D2:F0
            X509v3 Authority Key Identifier:
                keyid:CC:D3:15:C3:82:EC:1D:3B:DB:77:33:CB:49:07:11:AE:04:4B:D2:F0

            X509v3 Basic Constraints: critical
                CA:TRUE
    Signature Algorithm: sha256WithRSAEncryption
         42:64:35:61:ae:aa:86:d5:dd:a7:fb:6d:d4:e8:3b:3d:cd:b1:
         83:99:09:4a:fe:f2:e3:ea:04:ec:c9:65:62:c4:c8:a0:b1:35:
         a2:a8:a8:95:72:54:3e:42:11:5a:a9:6d:f0:7b:f7:86:30:39:
         31:d8:9a:1e:fe:5f:f0:52:d2:b8:32:e2:11:6b:16:b8:63:1a:
         bf:27:e6:f5:8c:09:4c:70:54:78:a3:e5:f7:f5:bd:1d:d0:6c:
         e9:34:31:f3:f1:3d:b5:3c:8d:1d:e3:56:b8:69:f1:57:59:09:
         b7:a3:89:0b:15:51:a4:c5:19:ba:3f:4f:8f:90:9c:10:ea:1e:
         f6:4b:92:ce:e4:65:7b:dd:06:1f:0a:5c:9d:e2:a3:67:50:34:
         3d:5f:45:59:32:2e:03:c7:fb:b9:2e:9d:06:22:a0:ee:c8:0b:
         95:c6:cc:fc:13:67:5e:8d:bc:cb:08:5e:77:0b:2a:7d:0e:77:
```

- Convertir certificado **.crt** → certificado **.pem**

```
daarso98@m1-daarso98:/etc/apache2/ssl$ cat apache_daarso98.key apache_daarso98.crt > newCertificate.pem
```

```
daarso98@m1-daarso98:/etc/apache2/ssl$ openssl x509 -in apache_daarso98 -out newCertificate.pem
```

```
daarso98@m1-daarso98:/etc/apache2/ssl$ openssl x509 -inform der -in apache_daarso98.crt -out newCertificate.pem
```

- Convertir certificado **.pem** → certificado **.der**

```
daarso98@m1-daarso98:/etc/apache2/ssl$ openssl x509 -outform der -in newCertificate.pem -out newCertificate.der
```

- Creamos un nuevo certificado con el comando **req** donde especificamos el nombre del archivo de salida con **out** y nombre de la clave con **keyout**

```
daarso98@m1-daarso98:/etc/apache2/ssl$ openssl req -new -newkey rsa:2048 -nodes -out request.csr -keyout apache_daarso98.key
```

3. Nginx como balanceador para peticiones HTTPS

El objetivo principal de la práctica es que la granja web nos permita usar HTTPS para ello debemos configurar el balanceador para que tambien acepte este tráfico. Esto se consigue siguiendo los pasos anteriormente detallados para las máquinas M1 y M2.

```
daarso98@m1-daarso98:/etc/apache2/ssl$ sudo scp  apache_daarso98.crt daarso98@192.168.56.106:/home/daarso98/apache_daarso98.crt
The authenticity of host '192.168.56.106 (192.168.56.106)' can't be established.
ECDSA key fingerprint is SHA256:beddR/+s/LyfBr4FsxBJXcLRo5B+26HRAE78hEZ06kk.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added '192.168.56.106' (ECDSA) to the list of known hosts.
daarso98@192.168.56.106's password:
apache_daarso98.crt                                   100% 1428   928.8KB/s   00:00
daarso98@m1-daarso98:/etc/apache2/ssl$ sudo scp  apache_daarso98.key daarso98@192.168.56.106:/home/daarso98/apache_daarso98.key
daarso98@192.168.56.106's password:
apache_daarso98.key                                   100% 1704     1.0MB/s   00:00
```

Es muy importante parar el servicio de haproxy de la práctica anterior y reiniciar el servicio de nginx, ya que ambos no pueden estar al mismo tiempo en funcionamiento.

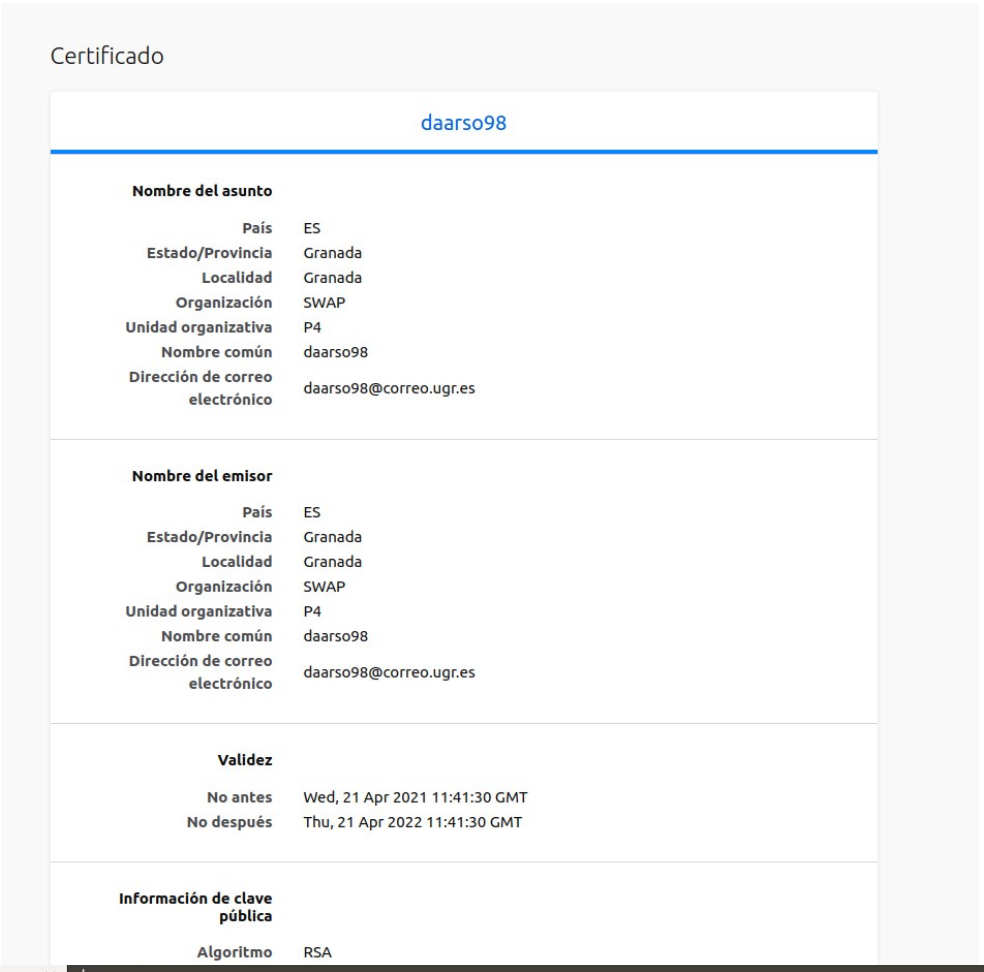
```
daarso98@m3-daarso98:~$ sudo systemctl haproxy stop
Unknown operation haproxy.
daarso98@m3-daarso98:~$ sudo systemctl stop haproxy
daarso98@m3-daarso98:~$ sudo systemctl status haproxy
● haproxy.service - HAProxy Load Balancer
   Loaded: loaded (/lib/systemd/system/haproxy.service; enabled; vendor preset: enabled)
   Active: inactive (dead) since Tue 2021-04-27 15:33:43 UTC; 5s ago
     Docs: man:haproxy(1)
           file:/usr/share/doc/haproxy/configuration.txt.gz
  Process: 9683 ExecStart=/usr/sbin/haproxy -Ws -f $CONFIG -p $PIDFILE $EXTRAOPTS (code=exited, stat
  Process: 9672 ExecStartPre=/usr/sbin/haproxy -f $CONFIG -c -q $EXTRAOPTS (code=exited, status=0/SU
 Main PID: 9683 (code=exited, status=143)

Apr 20 17:13:57 m3-daarso98 haproxy[9683]: [WARNING] 109/171357 (9683) : config : missing timeouts f
Apr 20 17:13:57 m3-daarso98 haproxy[9683]:    | While not properly invalid, you will certainly encou
Apr 20 17:13:57 m3-daarso98 haproxy[9683]:    | with such a configuration. To fix this, please ensur
Apr 20 17:13:57 m3-daarso98 haproxy[9683]:    | timeouts are set to a non-zero value: 'client', 'con
Apr 20 17:13:57 m3-daarso98 systemd[1]: Started HAProxy Load Balancer.
Apr 27 15:33:43 m3-daarso98 systemd[1]: Stopping HAProxy Load Balancer...
Apr 27 15:33:43 m3-daarso98 haproxy[9683]: [WARNING] 109/171357 (9683) : Exiting Master process...
Apr 27 15:33:43 m3-daarso98 haproxy[9683]: [ALERT] 109/171357 (9683) : Current worker 9684 exited wi
Apr 27 15:33:43 m3-daarso98 haproxy[9683]: [WARNING] 109/171357 (9683) : All workers exited. Exiting
Apr 27 15:33:43 m3-daarso98 systemd[1]: Stopped HAProxy Load Balancer.
daarso98@m3-daarso98:~$ sudo systemctl restart nginx
daarso98@m3-daarso98:~$ sudo systemctl status nginx
● nginx.service - A high performance web server and a reverse proxy server
   Loaded: loaded (/lib/systemd/system/nginx.service; enabled; vendor preset: enabled)
   Active: active (running) since Tue 2021-04-27 15:34:04 UTC; 5s ago
     Docs: man:nginx(8)
  Process: 9615 ExecStop=/sbin/start-stop-daemon --quiet --stop --retry QUIT/5 --pidfile /run/nginx.
  Process: 10849 ExecStart=/usr/sbin/nginx -g daemon on; master_process on; (code=exited, status=0/S
```

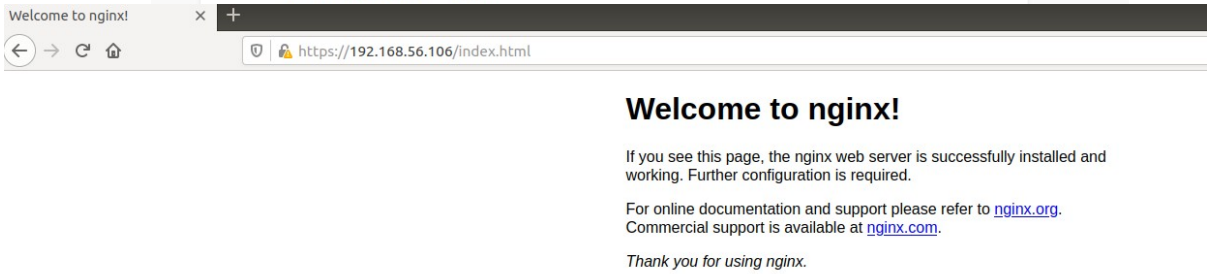
En el balanceador pondremos la ruta a la carpeta donde hayamos copiado el archivo .crt y .key. Posteriormente, debemos añadir un nuevo server en el archivo **/*etc*/nginx/conf.d/default.conf** de la siguiente manera:

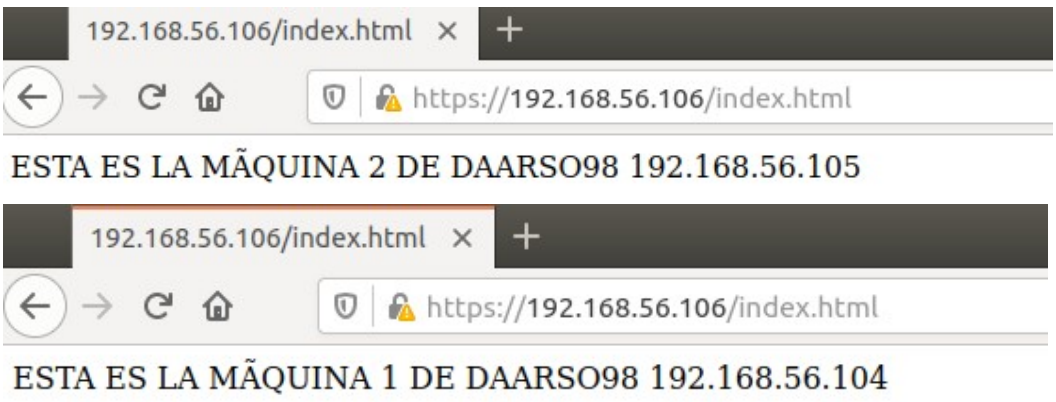
```
server{
        listen 443 ssl;
        ssl on;
        ssl_certificate /home/daarso98/apache_daarso98.crt;
        ssl_certificate_key /home/daarso98/apache_daarso98.key;

        server_name balanceadorSSL_daarso98;
        access_log /var/log/nginx/balanceadorSSL_daarso98.access.log;
        error_log /var/log/nginx/balanceadorSSL_daarso98.error.log;
        root /var/www/;

        location /
        {
        proxy_pass http://balanceo_daarso98;
        proxy_set_header Host $host;
        proxy_set_header X-Real-IP $remote_addr;
        proxy_set_header X-Fowarded-For $proxy_add_x_forwarded_for;
```

```
 GNU nano 2.9.3                        /etc/nginx/conf.d/default.conf

        ssl_certificate_key /home/daarso98/apache_daarso98.key;

        server_name balanceadorSSL_daarso98;
        access_log /var/log/nginx/balanceadorSSL_daarso98.access.log;
        error_log /var/log/nginx/balanceadorSSL_daarso98.error.log;
        root /var/www/;

        location /
        {
        proxy_pass http://balanceo_daarso98;
        proxy_set_header Host $host;
        proxy_set_header X-Real-IP $remote_addr;
        proxy_set_header X-Fowarded-For $proxy_add_x_forwarded_for;
        proxy_http_version 1.1;
        proxy_set_header Connection "";
        }
```

Podemos ver el certificado generado desde el navegador:







- **Opciones avanzadas para Nginx como balanceador para peticiones HTTPS**

- **SSLCertificateChainFile** para ensamblar los certificados en uno. Forman la cadena de certificados del servidor.

```
SSLCertificateFile      /etc/apache2/ssl/apache_daarso98.crt
SSLCertificateKeyFile   /etc/apache2/ssl/apache_daarso98.key
SSLCertificateChainFIle /etc/apache2/ssl/daarso98.crt
```

- **ssl_session_cache_shared** y **ssl_session_timeout** para compartir 20MB de caché durante 10 minutos.

```
ssl_session_cache shared: SSL:20m;
ssl_session_timeout 10m:
```

- **ssl_protocols** para indicar la versiones de protocolos a utilizar .

```
ssl_protocols TLSv1 TLSv1.1 TLSv1.2
```

- **ssl_prefer_server_ciphers,** cuando esta activado el propietario del servidor puede controlar que cifrados están disponibles. Si esta apagado y las versiones de TLS son obsoletas, un adversario puede interferir con el protocolo de enlace y forzar la conexión a utilizar cifrados débiles, lo que permite descifrar la conexión.

```
ssl_prefer_server_ciphers on;
```

4. Configuración del cortafuegos con IPTABLES.

Un cortafuegos es un sistema de seguridad para bloquear accesos no autorizados a un servidor u ordenador mientras sigue permitiendo la comunicación con otros servicios autorizados. Estos criterios se configuran mediante un cojuntos de reglas, usadas para bloquear puertos, direcciones IP, … La herramienta que usaremos será **IPTABLES** con la que podremos definir reglas para el filtrado de paquetes.

A continuación, estableceremos una configuración **básica** en la que denegaremos todo el tráfico entrante a las máquinas M1, M2 y el balanceador excepto el tráfico **HTTP (puerto 80 y 8080)** y **HTTPS (puerto 443)**. Para ello, documentaremos las reglas del cortafuegos a través de un script llamado *iptablesBasico.sh* en cada máquina.

```
 GNU nano 2.9.3                              iptablesBasico.sh

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

#Denegar todo el tráfico entrante
iptables -P INPUT DROP
iptables -P OUTPUT  ACCEPT
iptables -P FORWARD DROP

#Permitir tráfico por el puerto 80 (HTTP)
iptables -A INPUT -p tcp --dport 80 -j ACCEPT
iptables -A OUTPUT -p tcp --sport 80 -j ACCEPT

#Permitir tráfico por el puerto 8080 (HTTP)
iptables -A INPUT -p tcp --dport 8080 -j ACCEPT
iptables -A OUTPUT -p tcp --sport 8080 -j ACCEPT

#Permitir el tráfico por el puerto 443
iptables -A INPUT -p tcp --dport 443 -j ACCEPT
iptables -A OUTPUT -p tcp --sport 443 -j ACCEPT

iptables -L -n -v
```

```
root@m1-daarso98:/home/daarso98# ./iptablesBasico.sh
Chain INPUT (policy DROP 0 packets, 0 bytes)
 pkts bytes target     prot opt in     out     source               destination
    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:808
0
    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 ACCEPT 0 packets, 0 bytes)
 pkts bytes target     prot opt in     out     source               destination
    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:808
0
    0     0 ACCEPT     tcp  --  *      *       0.0.0.0/0            0.0.0.0/0           tcp spt:443
```

Usaremos este mismo script en el balanceador y en la máquina M2 para ello enviaremos mediante **scp** el script y posteriormente los ejecutaremos para establecer dicha configuración en el sistema.

```
root@m1-daarso98:/home/daarso98# ./iptablesRecover.sh
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
root@m1-daarso98:/home/daarso98# exit
exit
daarso98@m1-daarso98:~$ sudo scp -P 2222 -i ~/.ssh/mykey iptablesBasico.sh daarso98@192.168.56.105:/
home/daarso98/iptablesBasico.sh
iptablesBasico.sh                                           100%  654   654.5KB/s   00:00
daarso98@m1-daarso98:~$ sudo scp iptablesBasico.sh daarso98@192.168.56.106:/home/daarso98/iptablesBa
sico.sh
daarso98@192.168.56.106's password:
iptablesBasico.sh                                           100%  654   980.5KB/s   00:00
daarso98@m1-daarso98:~$ sudo scp -P 2222 -i ~/.ssh/mykey iptablesRecover.sh daarso98@192.168.56.105:
/home/daarso98/iptablesReccover.sh
iptablesRecover.sh                                          100%  220   417.3KB/s   00:00
daarso98@m1-daarso98:~$ sudo scp iptablesRecover.sh daarso98@192.168.56.106:/home/daarso98/iptablesR
ecover.sh
daarso98@192.168.56.106's password:
iptablesRecover.sh                                          100%  220   400.1KB/s   00:00
daarso98@m1-daarso98:~$ _
```

```
root@m2-daarso98:/home/daarso98# chmod 777 iptablesBasico.sh
root@m2-daarso98:/home/daarso98# ./iptablesBasico.sh
Chain INPUT (policy DROP 0 packets, 0 bytes)
 pkts bytes target     prot opt in     out     source               destination
    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:808
0
    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 ACCEPT 0 packets, 0 bytes)
 pkts bytes target     prot opt in     out     source               destination
    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:808
0
    0     0 ACCEPT     tcp  --  *      *       0.0.0.0/0            0.0.0.0/0           tcp spt:443
```

```
root@m3-daarso98:/home/daarso98# chmod 777 iptablesBasico.sh
root@m3-daarso98:/home/daarso98# ./iptablesBasico.sh
Chain INPUT (policy DROP 0 packets, 0 bytes)
 pkts bytes target     prot opt in     out     source               destination
    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:808
0
    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 ACCEPT 0 packets, 0 bytes)
 pkts bytes target     prot opt in     out     source               destination
    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:808
0
    0     0 ACCEPT     tcp  --  *      *       0.0.0.0/0            0.0.0.0/0           tcp spt:443
```

Finalmente, realizamos una serie de comprobaciones para ver el correcto funcionamiento del cortafuegos. Como se puede comprobar, no podemos conectarnos mediante **ssh (puerto 2222 en mi caso)** pero si que se permite el tráfico **HTTP (80 y 8080)** y **HTTPS (443)**.

```
(base) daarso@daarso-GP62-7RE:~$ ssh -p 2222 daarso98@192.168.56.104

^C
(base) daarso@daarso-GP62-7RE:~$ ssh -p 2222 daarso98@192.168.56.105

^C
(base) daarso@daarso-GP62-7RE:~$ ssh -p 2222 daarso98@192.168.56.106

^C
```

```
(base) daarso@daarso-GP62-7RE:~$ curl http://192.168.56.104:8080
<HTML>
  <BODY>
        ESTA ES LA MÁQUINA 1 DE DAARSO98 192.168.56.104
   </BODY>
</HTML>
(base) daarso@daarso-GP62-7RE:~$ curl http://192.168.56.105
<HTML>
  <BODY>
    ESTA ES LA MÁQUINA 2 DE DAARSO98 192.168.56.105
  </BODY>
</HTML>
(base) daarso@daarso-GP62-7RE:~$ curl -k https://192.168.56.104
<HTML>
  <BODY>
        ESTA ES LA MÁQUINA 1 DE DAARSO98 192.168.56.104
   </BODY>
</HTML>
(base) daarso@daarso-GP62-7RE:~$ curl -k https://192.168.56.105
<HTML>
  <BODY>
    ESTA ES LA MÁQUINA 2 DE DAARSO98 192.168.56.105
  </BODY>
</HTML>
```

También he creado un script llamado *iptablesRecover.sh* que me permite recuperar la configuración por defecto de la máquina habilitando todos los puertos y todo el tráfico.

```
 GNU nano 2.9.3                          iptablesRecover.sh

# Eliminar todas las reglas
iptables -F
iptables -X
iptables -Z
iptables -t nat -F

#Política por defecto: aceptar todo
iptables -P INPUT ACCEPT
iptables -P OUTPUT ACCEPT
iptables -P FORWARD ACCEPT

iptables -L -n -v
```

■ Configuraciones avanzadas del cortafuegos

A continuación, estableceremos una configuración **avanzada** del cortafuegos en la que permitiremos **SSH(22 y 2222)**, **DNS(53)** y **PING** a las máquinas M1, M2 y al balanceador. Además se permitirá el tráfico consigo misma (**localhost**) y el resto de servicios se denegarán. Para ello, documentaremos las reglas del cortafuegos a través de un script llamado *iptablesAvanzado.sh* en cada máquina.

```
 GNU nano 2.9.3                          iptablesAvanzado.sh

#Eliminar todas las reglas
iptables -F
iptables -X
iptables -Z
iptables -t nat -F

#Política por defecto: denegación de todo el tráfico
iptables -P INPUT DROP
iptables -P OUTPUT DROP
iptables -P FORWARD DROP

#Permitir cualquier acceso desde localhost
iptables -A INPUT -i lo -j ACCEPT
iptables -A OUTPUT -o lo -j ACCEPT

#Abrir el puerto 22 para permitir el acceso por SSH
iptables -A INPUT -p tcp --dport 22 -j ACCEPT
iptables -A OUTPUT -p tcp --sport 22 -j ACCEPT

#Abrir el puerto 2222 para permitir el acceso por SSH
iptables -A INPUT -p tcp --dport 2222 -j ACCEPT
iptables -A OUTPUT -p tcp --sport 2222 -j ACCEPT

#Abrir el puerto 53 para permitir el acceso por DNS
iptables -A INPUT -m state --state NEW -p udp --dport 53 -j ACCEPT
iptables -A OUTPUT -m state --state NEW -p udp --dport 53 -j ACCEPT

#Habilitar PING
iptables -A INPUT -p ICMP -j ACCEPT
iptables -A OUTPUT -p ICMP -j ACCEPT

iptables -L -n -v
```

Utilizamos **scp** para transferir el script entre las máquinas y los ejecutamos

```
Cdaarso98@m1-daarso98:~$ sudo scp -P 2222 -i ~/.ssh/mykey iptablesAvanzado.sh daarso98@192.168.56.1
/home/daarso98/iptablesAvanzado.sh
iptablesAvanzado.sh                                   100%  924   519.0KB/s   00:00
daarso98@m1-daarso98:~$ sudo scp iptablesAvanzado.sh daarso98@192.168.56.106:/home/daarso98/iptables
Avanzado.sh
daarso98@192.168.56.106's password:
iptablesAvanzado.sh                                   100%  924     1.4MB/s   00:00
```

```
daarso98@m1-daarso98:~$ sudo ./iptablesAvanzado.sh
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  --  *      *       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:222
2
    0     0 ACCEPT     udp  --  *      *       0.0.0.0/0            0.0.0.0/0           state NEW u
dp dpt:53
    0     0 ACCEPT     icmp --  *      *       0.0.0.0/0            0.0.0.0/0

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  --  *      *       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:222
2
    0     0 ACCEPT     udp  --  *      *       0.0.0.0/0            0.0.0.0/0           state NEW u
dp dpt:53
    0     0 ACCEPT     icmp --  *      *       0.0.0.0/0            0.0.0.0/0
```

```
daarso98@m2-daarso98:~$ sudo ./iptablesAvanzado.sh
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  --  *      *       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:222
2
    0     0 ACCEPT     udp  --  *      *       0.0.0.0/0            0.0.0.0/0           state NEW u
dp dpt:53
    0     0 ACCEPT     icmp --  *      *       0.0.0.0/0            0.0.0.0/0

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  --  *      *       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:222
2
    0     0 ACCEPT     udp  --  *      *       0.0.0.0/0            0.0.0.0/0           state NEW u
dp dpt:53
    0     0 ACCEPT     icmp --  *      *       0.0.0.0/0            0.0.0.0/0
```

```
root@m3-daarso98:/home/daarso98# sudo ./iptablesAvanzado.sh
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  --  *      *       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:22
2
    0     0 ACCEPT     udp  --  *      *       0.0.0.0/0            0.0.0.0/0           state NEW
dp dpt:53
    0     0 ACCEPT     icmp --  *      *       0.0.0.0/0            0.0.0.0/0

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  --  *      *       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:22
2
    0     0 ACCEPT     udp  --  *      *       0.0.0.0/0            0.0.0.0/0           state NEW
dp dpt:53
    0     0 ACCEPT     icmp --  *      *       0.0.0.0/0            0.0.0.0/0
```

Finalmente, realizamos unas pruebas para comprobar el funcionamiento. Como podemos comprobar **NO** se permite el tráfico **HTTP** y **HTTPS** , pero si **PING, SSH** y **DNS**.

```
(base) daarso@daarso-GP62-7RE:~$ curl http://192.168.56.104
^C
(base) daarso@daarso-GP62-7RE:~$ curl -k https://192.168.56.104
^C
(base) daarso@daarso-GP62-7RE:~$ ssh -p 2222 daarso98@192.168.56.104
Welcome to Ubuntu 18.04.5 LTS (GNU/Linux 4.15.0-136-generic x86_64)

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

  System information as of Tue Apr 27 19:20:45 UTC 2021

  System load:  0.0               Processes:             104
  Usage of /:   53.0% of 8.79GB   Users logged in:       1
  Memory usage: 34%               IP address for enp0s3: 10.0.2.15
  Swap usage:   0%                IP address for enp0s8: 192.168.56.104

 * Pure upstream Kubernetes 1.21, smallest, simplest cluster ops!

     https://microk8s.io/

65 packages can be updated.
0 updates are security updates.

New release '20.04.2 LTS' available.
Run 'do-release-upgrade' to upgrade to it.


*** System restart required ***
Last login: Tue Apr 27 19:13:40 2021 from 192.168.56.1
daarso98@m1-daarso98:~$ exit
logout
Connection to 192.168.56.104 closed.
(base) daarso@daarso-GP62-7RE:~$ ping 192.168.56.104
PING 192.168.56.104 (192.168.56.104) 56(84) bytes of data.
64 bytes from 192.168.56.104: icmp_seq=1 ttl=64 time=0.536 ms
64 bytes from 192.168.56.104: icmp_seq=2 ttl=64 time=0.534 ms
^C
--- 192.168.56.104 ping statistics ---
2 packets transmitted, 2 received, 0% packet loss, time 1020ms
rtt min/avg/max/mdev = 0.534/0.535/0.536/0.001 ms
```

```
(base) daarso@daarso-GP62-7RE:~$ curl http://192.168.56.105
^C
(base) daarso@daarso-GP62-7RE:~$ curl -k https://192.168.56.105
^C
(base) daarso@daarso-GP62-7RE:~$ ping 192.168.56.105
PING 192.168.56.105 (192.168.56.105) 56(84) bytes of data.
64 bytes from 192.168.56.105: icmp_seq=1 ttl=64 time=0.564 ms
64 bytes from 192.168.56.105: icmp_seq=2 ttl=64 time=0.617 ms
^C
--- 192.168.56.105 ping statistics ---
2 packets transmitted, 2 received, 0% packet loss, time 1019ms
rtt min/avg/max/mdev = 0.564/0.590/0.617/0.035 ms
(base) daarso@daarso-GP62-7RE:~$ ssh -p 2222 daarso98@192.168.56.105
Welcome to Ubuntu 18.04.5 LTS (GNU/Linux 4.15.0-136-generic x86_64)

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

  System information as of Tue Apr 27 19:34:54 UTC 2021

  System load:  0.05              Processes:             102
  Usage of /:   45.3% of 8.79GB   Users logged in:       1
  Memory usage: 34%               IP address for enp0s3: 10.0.2.15
  Swap usage:   0%                IP address for enp0s8: 192.168.56.105


86 packages can be updated.
21 updates are security updates.

New release '20.04.2 LTS' available.
Run 'do-release-upgrade' to upgrade to it.


Last login: Tue Apr 27 15:19:35 2021 from 192.168.56.1
daarso98@m2-daarso98:~$
```

```
(base) daarso@daarso-GP62-7RE:~$ ping 192.168.56.106
PING 192.168.56.106 (192.168.56.106) 56(84) bytes of data.
64 bytes from 192.168.56.106: icmp_seq=1 ttl=64 time=0.524 ms
64 bytes from 192.168.56.106: icmp_seq=2 ttl=64 time=0.573 ms
^C
--- 192.168.56.106 ping statistics ---
2 packets transmitted, 2 received, 0% packet loss, time 1017ms
rtt min/avg/max/mdev = 0.524/0.548/0.573/0.033 ms
(base) daarso@daarso-GP62-7RE:~$ ssh daarso98@192.168.56.106
daarso98@192.168.56.106's password:
Welcome to Ubuntu 18.04.5 LTS (GNU/Linux 4.15.0-140-generic x86_64)

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

  System information as of Wed Apr 28 10:09:57 UTC 2021

  System load:  0.0               Processes:             114
  Usage of /:   44.3% of 8.79GB   Users logged in:       1
  Memory usage: 25%               IP address for enp0s3: 10.0.2.15
  Swap usage:   0%                IP address for enp0s8: 192.168.56.106

 * Pure upstream Kubernetes 1.21, smallest, simplest cluster ops!

     https://microk8s.io/

5 packages can be updated.
0 of these updates are security updates.
To see these additional updates run: apt list --upgradable

New release '20.04.2 LTS' available.
Run 'do-release-upgrade' to upgrade to it.


*** System restart required ***
Last login: Tue Apr 27 17:59:42 2021 from 192.168.56.1
daarso98@m3-daarso98:~$
```

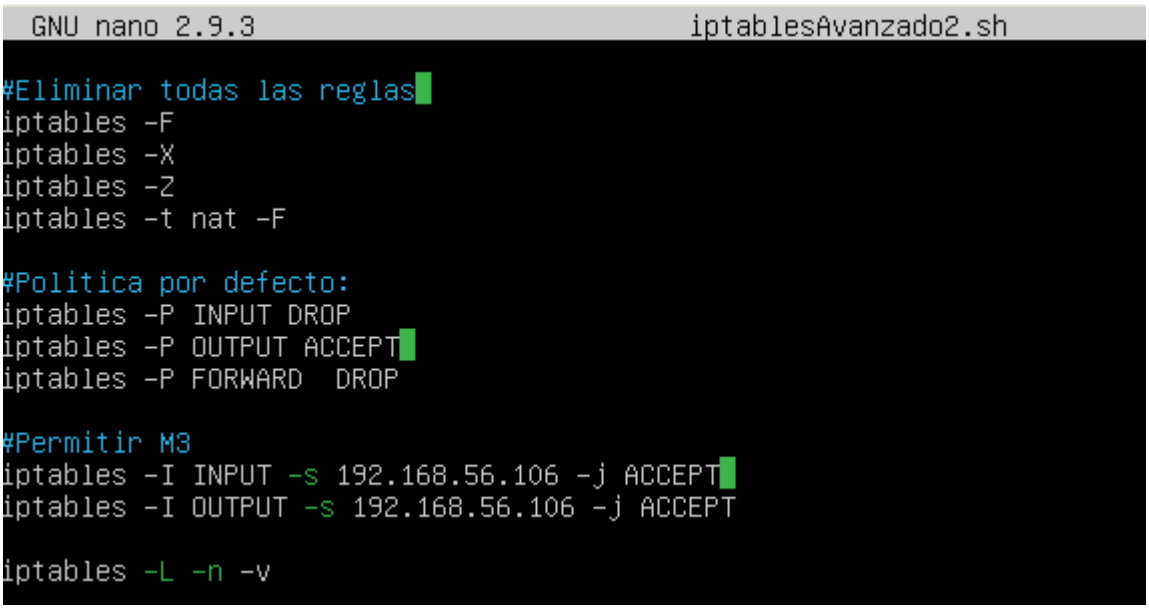
Comprobamos que se permite el tráfico consigo misma (**localhost**)

```
daarso98@m1-daarso98:~$ curl 192.168.56.104:8080
<HTML>
  <BODY>
        ESTA ES LA MÁQUINA 1 DE DAARSO98 192.168.56.104
   </BODY>
</HTML>
```

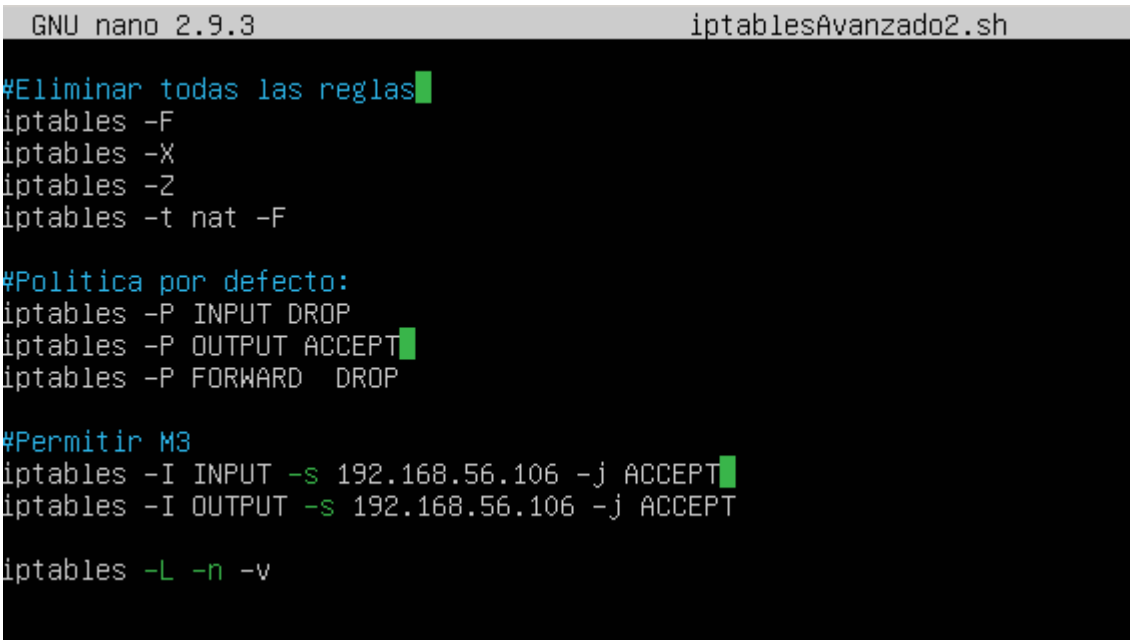
```
daarso98@m2-daarso98:~$ curl 192.168.56.105
<HTML>
  <BODY>
    ESTA ES LA MÁQUINA 2 DE DAARSO98 192.168.56.105
  </BODY>
</HTML>
```

Como segundo ejemplo **avanzado** estableceremos una configuración del cortafuegos en la que **SOLO** se aceptan peticiones **HTTP** y **HTTPS** provenientes del balanceador y M1 y M2 no acepten peticiones. Para ello, documentaremos las reglas del cortafuegos a través de un script llamado *iptablesAvanzado2.sh* en cada máquina.

- **M1**



- **M2**



- **M3 (balanceador)**

```
 GNU nano 2.9.3                              iptablesAvanzado2.sh

#Eliminar todas las reglas
iptables -F
iptables -X
iptables -Z
iptables -t nat -F

#Politica por defecto:
iptables -P INPUT DROP
iptables -P OUTPUT ACCEPT
iptables -P FORWARD  DROP
iptables -A INPUT -m state --state NEW,ESTABLISHED -j ACCEPT

#Permitir  puerto 80 para HTTP
iptables -A INPUT -p tcp --dport 80 -j ACCEPT
iptables -A OUTPUT -p tcp --sport 80 -j ACCEPT

#Permitir puerto 8080 para HTTP
iptables -A INPUT -p tcp --dport 8080 -j ACCEPT
iptables -A OUTPUT -p tcp --sport 8080 -j ACCEPT

#Permitir puerto 443 para HTTPS
iptables -A INPUT -p tcp --dport 443 -j ACCEPT
iptables -A OUTPUT -p tcp --sport 443 -j ACCEPT

iptables -L -n -v
```

A continuación, ejecutamos los scripts en cada una de las máquinas

```
root@m1-daarso98:/home/daarso98# ./iptablesAvanzado2.sh
Chain INPUT (policy DROP 0 packets, 0 bytes)
 pkts bytes target     prot opt in     out     source               destination
    0     0 ACCEPT     all  --  *      *       192.168.56.106       0.0.0.0/0

Chain FORWARD (policy DROP 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
    0     0 ACCEPT     all  --  *      *       192.168.56.106       0.0.0.0/0
```

```
root@m2-daarso98:/home/daarso98# ./iptablesAvanzado2.sh
Chain INPUT (policy DROP 0 packets, 0 bytes)
 pkts bytes target     prot opt in     out     source               destination
    0     0 ACCEPT     all  --  *      *       192.168.56.106       0.0.0.0/0

Chain FORWARD (policy DROP 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
    0     0 ACCEPT     all  --  *      *       192.168.56.106       0.0.0.0/0
```

```
root@m3-daarso98:/home/daarso98# sudo ./iptablesAvanzado2.sh
Chain INPUT (policy DROP 0 packets, 0 bytes)
 pkts bytes target     prot opt in     out     source               destination
    0     0 ACCEPT     all  --  *      *       0.0.0.0/0            0.0.0.0/0           state NEW,E
STABLISHED
    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:808
0
    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 ACCEPT 0 packets, 0 bytes)
 pkts bytes target     prot opt in     out     source               destination
    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:808
0
    0     0 ACCEPT     tcp  --  *      *       0.0.0.0/0            0.0.0.0/0           tcp spt:443
```

Realizamos pruebas para comprobar su funcionamiento. Como se puede apreciar, se deniega el tráfico **HTTP** y **HTTPS** proviniente de M1 y M2 pero no desde M3.

```
(base) daarso@daarso-GP62-7RE:~$ curl http://192.168.56.104

^C
(base) daarso@daarso-GP62-7RE:~$ curl http://192.168.56.105

^C
(base) daarso@daarso-GP62-7RE:~$ curl -k https://192.168.56.104

^C
(base) daarso@daarso-GP62-7RE:~$ curl -k https://192.168.56.105

^C
```

```
root@m3-daarso98:/home/daarso98# curl http://192.168.56.104:8080
<HTML>
  <BODY>
        ESTA ES LA MÁQUINA 1 DE DAARSO98 192.168.56.104
   </BODY>
</HTML>
root@m3-daarso98:/home/daarso98# curl http://192.168.56.105
<HTML>
  <BODY>
    ESTA ES LA MÁQUINA 2 DE DAARSO98 192.168.56.105
  </BODY>
</HTML>
root@m3-daarso98:/home/daarso98# curl -k https://192.168.56.104
<HTML>
  <BODY>
        ESTA ES LA MÁQUINA 1 DE DAARSO98 192.168.56.104
   </BODY>
</HTML>
root@m3-daarso98:/home/daarso98# curl -k https://192.168.56.105
<HTML>
  <BODY>
    ESTA ES LA MÁQUINA 2 DE DAARSO98 192.168.56.105
  </BODY>
</HTML>
```

```
(base) daarso@daarso-GP62-7RE:~$ curl http://192.168.56.106
<HTML>
  <BODY>
        ESTA ES LA MÁQUINA 1 DE DAARSO98 192.168.56.104
   </BODY>
</HTML>
(base) daarso@daarso-GP62-7RE:~$ curl -k https://192.168.56.106
<HTML>
  <BODY>
    ESTA ES LA MÁQUINA 2 DE DAARSO98 192.168.56.105
  </BODY>
</HTML>
```

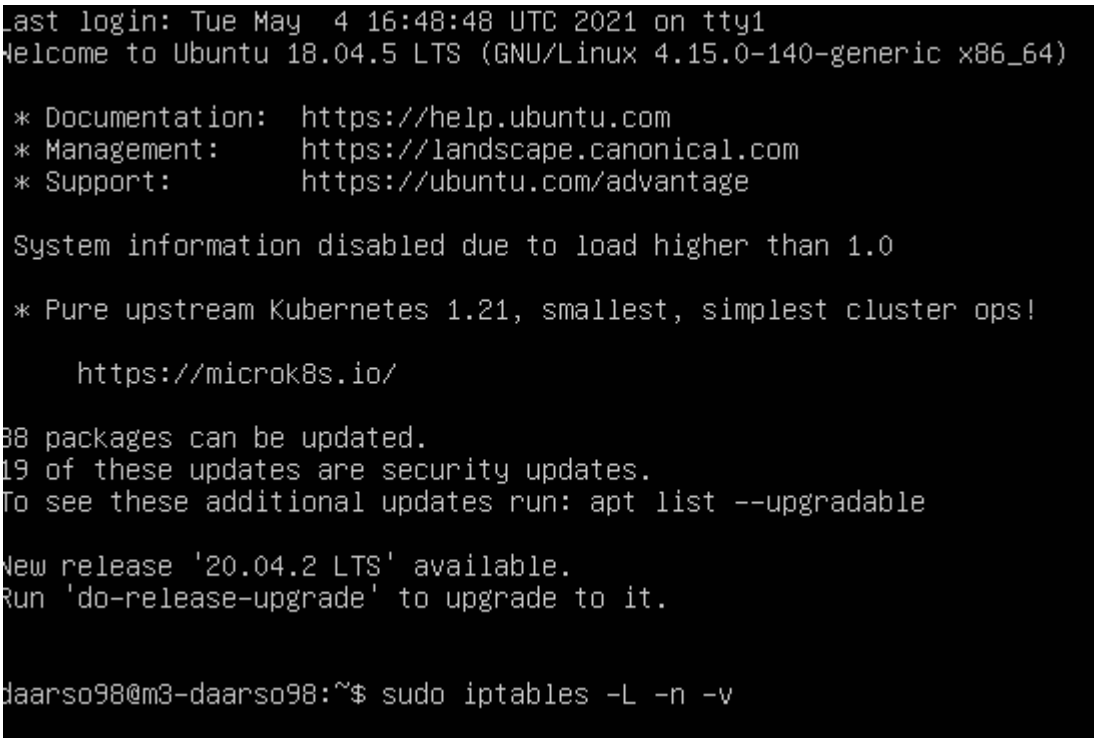
4.1 Configuración del cortafuegos al arranque

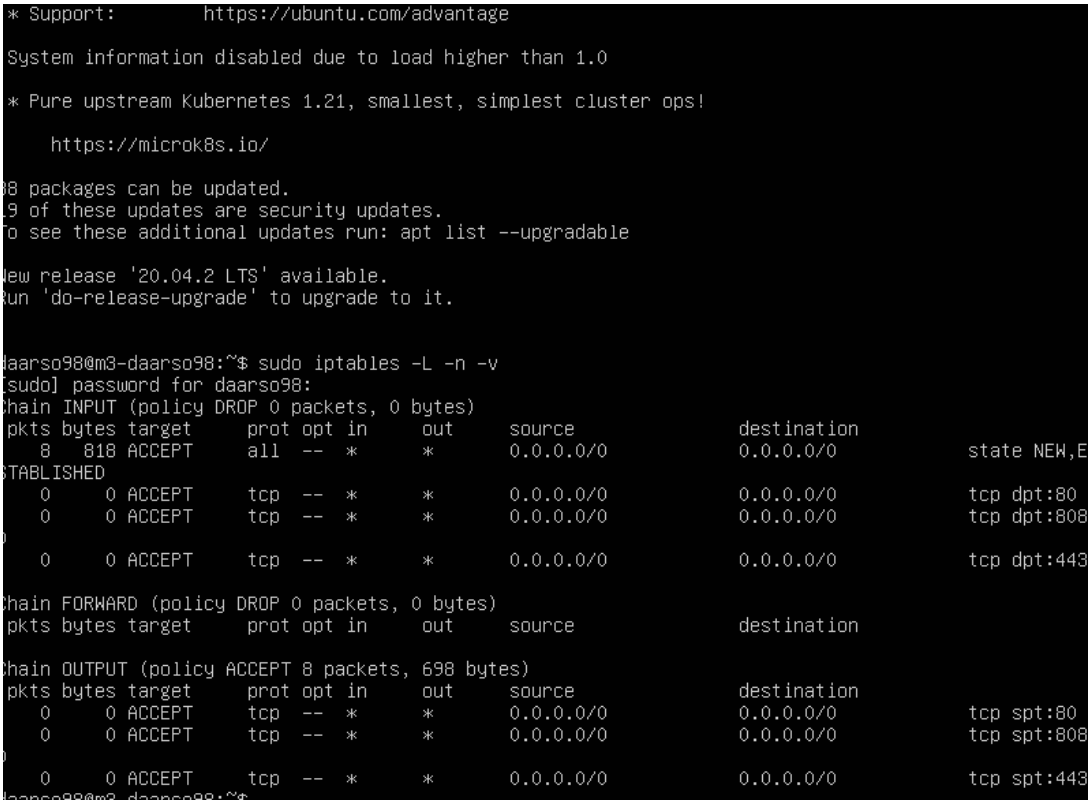
Para hacer las reglas de IPTABLES permanentes , podemos añadir un **crontab** para que se ejecute el script con las reglas de configuración del cortafuegos al arranque del sistema. Para ello, añadimos la siguiente regla en el archivo **/etc/crontab**

```
 GNU nano 2.9.3                          /etc/crontab                          Modified

# /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$
@reboot         root    sh /home/daarso98/iptablesAvanzado.sh _
#
```

```
@reboot         root    sh /home/daarso98/iptablesAvanzado.sh
```

Para comprobar el funcionamiento, solo hay que reiniciar el sitema y verificar que las reglas se han ejecutado.





5. Crear, instalar y configurar certificado SSL en Apache y Nginx con CERTBOT

Certbot es una herramienta de software gratuita y de código abierto para usar automáticamente en sitios web administrados manualmente para habilitar HTTPS. La mayoria de las distribuciones de Linux proporcionan certbot en sus repositorios oficales. A continuación, se encuentran las instrucciones de instalación para plataformas ampliamente utilizadas.

```
root@m1-daarso98:/home/daarso98# sudo apt-get install -y certbot
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following additional packages will be installed:
  libpython-stdlib libpython2.7-minimal libpython2.7-stdlib python python-minimal
  python2.7 python2.7-minimal python3-acme python3-certbot python3-configargparse
  python3-josepy python3-lib2to3 python3-mock python3-parsedatetime python3-pbr
  python3-requests-toolbelt python3-rfc3339 python3-tz python3-zope.component pyth
  python3-zope.hookable
```

Para mostrar una lista de los certificados administrados por certbot en tu servidor, utilice el siguiente comando:

```
root@m1-daarso98:/home/daarso98# certbot certificates
Saving debug log to /var/log/letsencrypt/letsencrypt.log

- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
No certs found.
- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
```

Para usar un servidor web existente, nos tenemos que asegurar de que este ejecutando y escuchando en el puerto 80 antes de ejecutar el siguiente comando. Se le pedirá que ingrese información acerca del nombre de dominio, ruta de la raíz web, …

```
root@m1-daarso98:/home/daarso98# certbot certonly --webroot --webroot-path /var/www/html --agree-tos
 -m daarso98@correo.ugr.es -d www.exampledaarso98.com
Saving debug log to /var/log/letsencrypt/letsencrypt.log
Plugins selected: Authenticator webroot, Installer None
Obtaining a new certificate
Performing the following challenges:
http-01 challenge for www.exampledaarso98.com
Using the webroot path /var/www/html for all unmatched domains.
Waiting for verification...
Cleaning up challenges
Failed authorization procedure. www.exampledaarso98.com (http-01): urn:ietf:params:acme:error:dns ::
 DNS problem: NXDOMAIN looking up A for www.exampledaarso98.com - check that a DNS record exists for
 this domain

IMPORTANT NOTES:
 - The following errors were reported by the server:

   Domain: www.exampledaarso98.com
   Type:   None
   Detail: DNS problem: NXDOMAIN looking up A for
   www.exampledaarso98.com - check that a DNS record exists for this
   domain
```

Como ya se ha mencionado CertBot puede automatizar todo el proceso de configuración HTTPS, incluida la configuración del servidor web. Los complementos están tanto disponibles para **Apache** como para **Nginx.** Para ello, instalamos el complemento certbot específico para su servidor y a continuación ejecute los siguientes comandos dependiendo de del servidor que utilice.

```
root@m1-daarso98:/home/daarso98# apt-get install -y python-certbot-apache
```

```
root@m1-daarso98:/home/daarso98# apt-get install -y python-certbot-nginx
```

```
root@m1-daarso98:/home/daarso98# certbot run --apache
Saving debug log to /var/log/letsencrypt/letsencrypt.log
Plugins selected: Authenticator apache, Installer apache
No names were found in your configuration files. Please enter in your domain
name(s) (comma and/or space separated)  (Enter 'c' to cancel): www.exampledaarso98.com
```

```
root@m1-daarso98:/home/daarso98# certbot run --nginx
Saving debug log to /var/log/letsencrypt/letsencrypt.log
Plugins selected: Authenticator nginx, Installer nginx
No names were found in your configuration files. Please enter in your domain
name(s) (comma and/or space separated)  (Enter 'c' to cancel): www.exampledaarso98.com
```

Suponiendo que el servidor web ya esté configurado para su nombre de dominio, certbot analizará la configuración existente y le pedirá que elijas para qué nombre de dominio HTTPS debe activarse. Si el servidor web no está configurado o si certbot no puede detectar sus nombres de dominio, simplemente ingrese sus nombres de dominio manualmente cuando te lo solicite.

Otra forma de configurar el certificado SSL para **apache** y **nginx** es buscando la directiva **ServerName** y coincidiendo con el dominio para el que solicitamos el certificado.

```
 GNU nano 2.9.3                    000-default.conf                         Modified

<VirtualHost *:80>
        # The ServerName directive sets the request scheme, hostname and port that
        # the server uses to identify itself. This is used when creating
        # redirection URLs. In the context of virtual hosts, the ServerName
        # specifies what hostname must appear in the request's Host: header to
        # match this virtual host. For the default virtual host (this file) this
        # value is not decisive as it is used as a last resort host regardless.
        # However, you must set it for any further virtual host explicitly.
        ServerName www.exampledaarso98.com
```

Se debe de disponer de un bloque VirtualHost para el dominio

```
root@m1-daarso98:/etc/apache2/sites-available# sudo cerboot --apache -d exampledaarso98 -d www.examp
ledaarso98.com
```

De forma análoga, podemos realizar los pasos anteriores para **nginx**

```
root@m1-daarso98:~# sudo nano /etc/nginx/sites-available/exampledaarso98.com
```

```
server_name exampledaarso98.com www.exampledaarso98.com
```

```
root@m1-daarso98:~# sudo cerbot --nginx -d exampledaarso98.com -d www.exampledaarso98.com
```