UNIVERSIDAD DE GRANADA

Servidores Web de Altas Prestaciones

Trabajo #35: Configuración de VPN en máquinas virtuales o Raspberry Pi para asegurar una granja web.

Juan Ocaña Valenzuela

23 de mayo de 2020

Versión: 1.0

Índice

1. Introducción

A lo largo de este documento se explorará qué es una VPN (*Virtual Private Network*), cómo construir, configurar y conectar nuestro propio servidor VPN y qué utilidad puede brindar en un centro de datos de diferentes maneras.

2. Qué es una VPN

Una **Red Privada Virtual** o **VPN** es una tecnología que permite establecer una red local a través de una infraestructura pública a través de un túnel cifrado. De esta forma, se pueden comunicar distintos equipos y servicios de forma distribuida como si estuviesen dentro de la misma red, aun encontrándose repartidos por todo el mundo.

Además, el servidor VPN actúa como intermediario de las conexiones, por lo que, a ojos de internet, nuestra IP pública pasa a ser la del servidor. Las comunicaciones están completamente cifradas, evitando posibles ataques *man in the middle*, y gracias a este tipo de conexiones y una buena configuración de cortafuegos se puede limitar el acceso a una infraestructura a través de internet.

Más adelante veremos distintas posibles aplicaciones de esta tecnología en el ámbito de los centros de datos y las granjas de servidores, pero ahora veremos paso a paso cómo instalar nuestra propia infraestructura VPN —servidor, entidad certificadora y clientes— mediante la herramienta **OpenVPN**.

3. Instalación de infraestructura VPN mediante OpenVPN

3.1. OpenVPN

OpenVPN es una herramienta y protocolo de código abierto utilizada en todo el mundo para establecer este tipo de conexiones. Su extensa documentación y comunidad activa, así como numerosas guías de instalación y uso, son un punto fuerte por el cual es una opción muy recomendable. Además es multiplataforma, por lo que se pueden configurar servidores y clientes en diferentes sistemas operativos.

3.2. Antes de instalar

Inicialmente se iba a realizar la instalación en una Raspberry Pi 3 B+, pero debido a las circunstancias excepcionales es imposible acceder a ella en estos momentos. Sin embargo, el procedimiento es el mismo y no debería haber mayor problema en replicarlo en Raspbian u otros sistemas, salvo ligeras diferencias en el gestor de paquetes y la localización de algunos archivos de configuración.

3.3. Preparación

3.3.1. Servidor VPN

Para el servidor VPN, utilizaremos una máquina virtual con Ubuntu 18.04 LTS, con 10GB de disco y 1024MB de RAM. Para asegurar que dispone de una IP única, debemos configurar el adaptador sólo-anfitrión de VirtualBox.

En el instalador de Ubuntu proporcionamos los siguientes datos:

- **Nombre del servidor:** vpn
- **Nombre de usuario:** patchispatch
- **Contraseña:** Swap1234

Es necesario que el usuario creado tenga privilegios `sudo`, algo que el instalador de Ubuntu hace por nosotros.
Además, le proporcionaremos una IP estática, en este caso `192.168.56.106` mediante la configuración de netplan:

```
network:
    ethernets:
        enp0s3:
            dhcp4: true
        enp0s8:
            addresses:
              - 192.168.56.106/24
    version: 2
```

Una vez modificada la configuración, aplicamos los cambios con `sudo netplan apply`. Ya tenemos nuestra máquina lista para conectarse con el resto de la red local.

3.3.2. Entidad certificadora

Aunque es posible utilizar la misma máquina servidora como entidad certificadora, la documentación oficial de OpenVPN recomienda que se utilice una máquina dedicada de forma exclusiva a importar y firmar certificados.

Para esta máquina certificadora, utilizaremos también Ubuntu 18.04 LTS, con 10GB de disco y 512MB de RAM. De nuevo, configuramos el adaptador sólo-anfitrión de VirtualBox para permitir que nuestra máquina disponga de una IP única para comunicarse con otras máquinas.

En el instalador de Ubuntu proporcionamos los siguientes datos:

- **Nombre del servidor:** ca

- **Nombre de usuario:** patchispatch
- **Contraseña:** Swap1234

De nuevo, necesitamos un usuario con privilegios `sudo`, aunque ya hemos visto que Ubuntu se encarga de ello.
La entidad certificadora tendrá la IP estática `192.168.56.107`:

```
network:
    ethernets:
        enp0s3:
            dhcp4: true
        enp0s8:
            addresses:
              - 192.168.56.107/24
    version: 2
```

3.3.3. Cortafuegos básico

Para establecer seguridad tanto en el servidor VPN como en la entidad certificadora, vamos a establecer un firewall básico. En lugar de utilizar iptables como se ha visto en la asignatura, usaremos la herramienta **ufw**, dada su facilidad de uso. Partiremos de una configuración que permita únicamente el acceso mediante SSH en ambas máquinas.

```
root@vpn:/home/patchispatch# ufw allow OpenSSH
Rules updated
Rules updated (v6)
root@vpn:/home/patchispatch# ufw enable
Firewall is active and enabled on system startup
root@vpn:/home/patchispatch# ufw status
Status: active

To                         Action      From
--                         ------      ----
OpenSSH                    ALLOW       Anywhere
OpenSSH (v6)               ALLOW       Anywhere (v6)
```

3.4. Instalación de servicios

En primer lugar, instalamos en nuestra máquina **vpn** el servidor OpenVPN con `sudo apt install openvpn`.

Para configurar la entidad certificadora, debemos descargar mediante *wget* el paquete `EasyRSA` con la siguiente orden, **tanto en la máquina vpn como en la máquina ca**:

wget -P ~/ https://github.com/OpenVPN/easy-rsa/releases/download/v3.0.7/EasyRSA-3.0.7.tgz

Y extraemos con `tar xvf`.

3.5. Configuración de la entidad certificadora

Para configurar la entidad certificadora, en nuestra máquina **CA** entramos en la carpeta `EasyRSA-3.0.7` que acabamos de descomprimir, y copiamos el archivo `vars.example` como `vars` para editarlo.
Debemos especificar los datos de nuestra entidad certificadora, descomentando los valores del archivo y fijando los que queramos. Los datos de nuestra entidad certificadora serán los siguientes:

```
# Choices are:
#   cn_only  - use just a CN value
#   org      - use the "traditional" Country/Province/City/Org/OU/email/CN format

#set_var EASYRSA_DN     "cn_only"

# Organizational fields (used with 'org' mode and ignored in 'cn_only' mode.)
# These are the default values for fields which will be placed in the
# certificate.  Don't leave any of these fields blank, although interactively
# you may omit any specific field by typing the "." symbol (not valid for
# email.)

set_var EASYRSA_REQ_COUNTRY     "ES"
set_var EASYRSA_REQ_PROVINCE    "Córdoba"
set_var EASYRSA_REQ_CITY        "Villanueva de Córdoba"
set_var EASYRSA_REQ_ORG         "Servidores Web de Altas Prestaciones"
set_var EASYRSA_REQ_EMAIL       "patchispatch@correo.ugr.es"
set_var EASYRSA_REQ_OU          "SWAP"

# Choose a size in bits for your keypairs. The recommended value is 2048.  Using
# 2048-bit keys is considered more than sufficient for many years into the
# future. Larger keysizes will slow down TLS negotiation and make key/DH param
# generation take much longer. Values up to 4096 should be accepted by most
# software. Only used when the crypto alg is rsa (see below.)

"vars" 223L, 8936C escritos                                   100,29-37     41%
```

Para gestionar el servicio de RSA, debemos ejecutar el script `easyrsa` con una serie de opciones. Primero lo ejecutamos con la opción `init-pki`, para crear la infraestructura de claves públicas. Esto lo realizamos en el servidor **CA**, no en el servidor VPN.

```
patchispatch@ca:~/EasyRSA-3.0.7$ ./easyrsa init-pki

Note: using Easy-RSA configuration from: /home/patchispatch/EasyRSA-3.0.7/vars

init-pki complete; you may now create a CA or requests.
Your newly created PKI dir is: /home/patchispatch/EasyRSA-3.0.7/pki
```

Para crear el certificado, ejecutamos de nuevo el script `easyrsa`, esta vez con la opción `build-ca`. Esto generará los archivos `ca.crt` y `ca.key`. Como vamos a generar el certificado sin contraseña esta vez, debemos indicar también la opción `nopass`:

```
patchispatch@ca:~/EasyRSA-3.0.7$ ./easyrsa build-ca nopass

Note: using Easy-RSA configuration from: /home/patchispatch/EasyRSA-3.0.7/vars
Using SSL: openssl OpenSSL 1.1.1  11 Sep 2018
Generating RSA private key, 2048 bit long modulus (2 primes)
.....................................................................+++++
......+++++
e is 65537 (0x010001)
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.
-----
Common Name (eg: your user, host, or server name) [Easy-RSA CA]:patchispatch

CA creation complete and you may now import and sign cert requests.
Your new CA certificate file for publishing is at:
/home/patchispatch/EasyRSA-3.0.7/pki/ca.crt
```

Con estos pasos, la entidad certificadora está lista para firmar certificados.

3.6. Generación y firmado del certificado del servidor

Para generar un certificado para nuestro servidor OpenVPN, debemos abrir la carpeta EasyRSA-3.0.7 en nuestra máquina, e inicializar la infraestructura de claves con `./easyrsa init-pki`, como hicimos con nuestra entidad certificadora.

Una vez realizado, debemos generar una solicitud de certificado con `./easyrsa gen-req`. Identificaremos el servidor VPN como `server`, e incluimos la opción `nopass` para que no nos solicite contraseña para las claves. La instrucción completa quedaría como `./easyrsa gen-req server nopass`:

```
patchispatch@vpn:~/EasyRSA-3.0.7$ ./easyrsa gen-req server nopa
Using SSL: openssl OpenSSL 1.1.1  11 Sep 2018
Generating a RSA private key
................................................+++++
.....................................................+++++
writing new private key to '/home/patchispatch/EasyRSA-3.0.7/pki/easy-rsa-2055.4dk4zS/tmp.0pXO
Yl'
-----
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.
-----
Common Name (eg: your user, host, or server name) [server]:

Keypair and certificate request completed. Your files are:
req: /home/patchispatch/EasyRSA-3.0.7/pki/reqs/server.req
key: /home/patchispatch/EasyRSA-3.0.7/pki/private/server.key
```

Esto nos generará una clave privada y el certificado del servidor, llamado `server.req`. Debemos copiar la clave privada a nuestra carpeta `/etc/openvpn`. Además, debemos enviar el certificado a nuestro servidor CA. Lo haremos mediante `scp`:

```
patchispatch@vpn:~/EasyRSA-3.0.7$ sudo cp ~/EasyRSA-3.0.7/pki/private/server.key /etc/openvpn/
[sudo] password for patchispatch:
patchispatch@vpn:~/EasyRSA-3.0.7$ scp /home/patchispatch/EasyRSA-3.0.7/pki/reqs/server.req pat
chispatch@192.168.56.107:/tmp
patchispatch@192.168.56.107's password:
server.req                                    100%  887     1.6MB/s   00:00
patchispatch@vpn:~/EasyRSA-3.0.7$
```

En la máquina CA debemos firmar el certificado con el polivalente script `easyrsa`. Para importarlo, ejecutamos `./easyreq import-req /tmp/server.req server`, y para firmarlo utilizamos la opción `sign-req`:

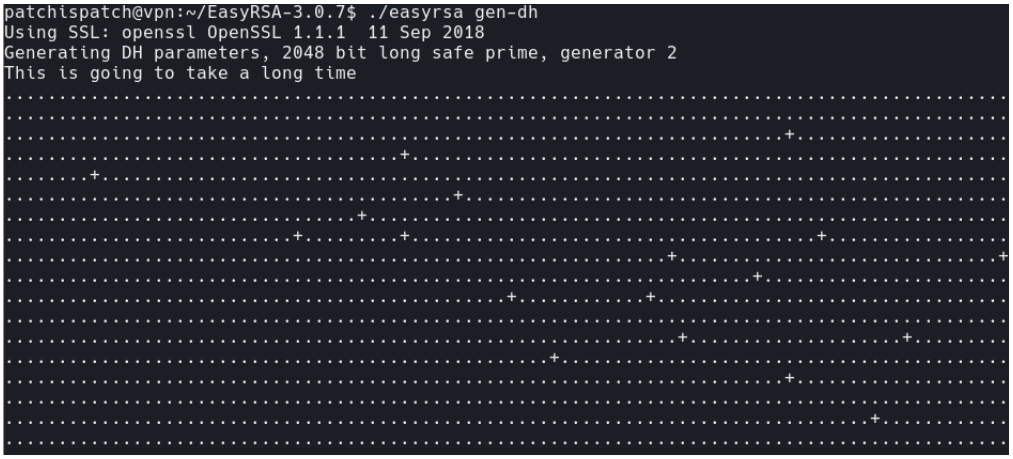
```
patchispatch@ca:~/EasyRSA-3.0.7$ ./easyrsa import-req /tmp/server.req server

Note: using Easy-RSA configuration from: /home/patchispatch/EasyRSA-3.0.7/vars
Using SSL: openssl OpenSSL 1.1.1  11 Sep 2018

The request has been successfully imported with a short name of: server
You may now use this name to perform signing operations on this request.


patchispatch@ca:~/EasyRSA-3.0.7$ ./easyrsa sign-req server server

Note: using Easy-RSA configuration from: /home/patchispatch/EasyRSA-3.0.7/vars
Using SSL: openssl OpenSSL 1.1.1  11 Sep 2018


You are about to sign the following certificate.
Please check over the details shown below for accuracy. Note that this request
has not been cryptographically verified. Please be sure it came from a trusted
source or that you have verified the request checksum with the sender.

Request subject, to be signed as a server certificate for 825 days:

subject=
    commonName                = server


Type the word 'yes' to continue, or any other input to abort.
  Confirm request details: yes
Using configuration from /home/patchispatch/EasyRSA-3.0.7/pki/easy-rsa-14787.V3zuze/tmp.rvhTr8
Check that the request matches the signature
Signature ok
The Subject's Distinguished Name is as follows
commonName            :ASN.1 12:'server'
Certificate is to be certified until Aug 22 10:22:43 2022 GMT (825 days)

Write out database with 1 new entries
Data Base Updated

Certificate created at: /home/patchispatch/EasyRSA-3.0.7/pki/issued/server.crt
```

Ya hemos firmado nuestro certificado, y debemos enviarlo al servidor VPN, de nuevo, utilizando `scp`. Además, transferimos el certificado de la propia entidad:

```
scp pki/issued/server.crt patchispatch@192.168.56.106:/tmp
scp pki/issued/server.crt patchispatch@192.168.56.106:/tmp
```

Desde la máquina VPN, copiamos los dos certificados a `/etc/openvpn`.

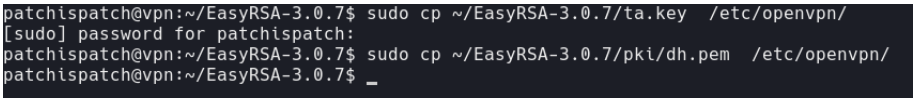
Ahora necesitamos claves de encriptado fuertes, para garantizar la integridad de nuestro servidor VPN. Desde la carpeta EasyRSA-3.0.7 ejecutamos `./easyrsa gen-dh`, lo que nos generará una clave de cifrado Diffie-Hellman.



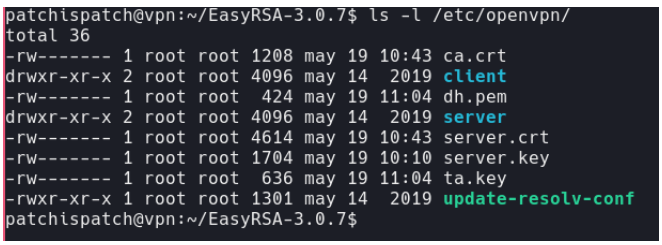
Una vez termine el proceso, que en nuestro caso ha tardado alrededor de un minuto, generaremos una firma HMAC con `openvpn` para fortalecer la seguridad de la verificación aún más:

```
openvpn --genkey --secret ta.key
```

Y cuando termine, copiamos las dos claves a nuestro directorio `/etc/openvpn`



Ahora disponemos de todas las claves y certificados necesarios para que las máquinas cliente accedan al servidor OpenVPN:



Para minimizar el riesgo de que accedan a nuestra entidad certificadora, cerramos la sesión. Si la hubiésemos configurado en nuestro servidor vpn, siempre estaría disponible para que un atacante accediese y pudiese firmar certificados ilícitos.

3.7. Generación del certificado y las claves del cliente

a generación y firmado de claves se va a realizar desde el servidor, ya que para este ejemplo es más rápido y además permite ser automatizada.

Para guardar correctamente los certificados, crearemos la carpeta `~/client-configs/keys` en el servidor VPN, y otorgaremos permisos `700` para asegurarla.

Para generar el certificado del primer cliente, al que llamaremos `client1`, debemos volver al directorio `EasyRSA-3.0.7`, y ejecutar de nuevo el script con la opción `gen-req`. También podemos indicar `nopass` si no queremos contraseña:

```
./easyrsa gen-req client1 nopass
```

```
patchispatch@vpn:~$ cd EasyRSA-3.0.7/
patchispatch@vpn:~/EasyRSA-3.0.7$ ./easyrsa gen-req client1 nopass
Using SSL: openssl OpenSSL 1.1.1  11 Sep 2018
Generating a RSA private key
.......................+++++
.....................+++++
writing new private key to '/home/patchispatch/EasyRSA-3.0.7/pki/easy-rsa-2437.uFHjTO/tmp.BHp8
Wc'
-----
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.
-----
Common Name (eg: your user, host, or server name) [client1]:

Keypair and certificate request completed. Your files are:
req: /home/patchispatch/EasyRSA-3.0.7/pki/reqs/client1.req
key: /home/patchispatch/EasyRSA-3.0.7/pki/private/client1.key
```

Copiamos la clave generada a la carpeta creada anteriormente y transferimos el certificado a nuestra máquina CA:

```
cp pki/private/client1.key ~/client-configs/keys/
scp pki/reqs/client1.req patchispatch@192.168.56.107:/tmp
```

En nuestra máquina CA, importamos el certificado como hemos hecho anteriormente, con `easyrsa import-req`. Una vez hecho esto, procedemos a firmar el certificado, esta vez asegurándonos de que lo hacemos para un cliente:

```
patchispatch@ca:~/EasyRSA-3.0.7$ ./easyrsa sign-req client client1

Note: using Easy-RSA configuration from: /home/patchispatch/EasyRSA-3.0.7/vars
Using SSL: openssl OpenSSL 1.1.1  11 Sep 2018


You are about to sign the following certificate.
Please check over the details shown below for accuracy. Note that this request
has not been cryptographically verified. Please be sure it came from a trusted
source or that you have verified the request checksum with the sender.

Request subject, to be signed as a client certificate for 825 days:

subject=
    commonName                = client1


Type the word 'yes' to continue, or any other input to abort.
  Confirm request details: yes
Using configuration from /home/patchispatch/EasyRSA-3.0.7/pki/easy-rsa-15099.6Mdvg5/tmp.qp8uro
Check that the request matches the signature
Signature ok
The Subject's Distinguished Name is as follows
commonName            :ASN.1 12:'client1'
Certificate is to be certified until Aug 22 13:07:07 2022 GMT (825 days)

Write out database with 1 new entries
Data Base Updated

Certificate created at: /home/patchispatch/EasyRSA-3.0.7/pki/issued/client1.crt
```

Enviamos el certificado firmado al servidor, y una vez allí, lo copiamos a `~/client-configs/keys/`. También debemos copiar los archivos `ta.key` y `ca.crt` que hemos generado anteriormente.

```
patchispatch@vpn:~/EasyRSA-3.0.7$ cp pki/private/client1.key ~/client-configs/keys/
patchispatch@vpn:~/EasyRSA-3.0.7$ scp pki/reqs/client1.req patchispatch@192.168.56.107:/tmp
patchispatch@192.168.56.107's password:
client1.req                                   100%  887     1.3MB/s   00:00
patchispatch@vpn:~/EasyRSA-3.0.7$ cp /tmp/client1.crt ~/client-configs/keys/
patchispatch@vpn:~/EasyRSA-3.0.7$ cp ta.key ~/client-configs/keys/
patchispatch@vpn:~/EasyRSA-3.0.7$ sudo cp /etc/openvpn/ca.crt ~/client-configs/keys/
[sudo] password for patchispatch:
patchispatch@vpn:~/EasyRSA-3.0.7$
```

Utilizaremos los archivos más adelante. Ahora vamos a configurar el servicio OpenVPN.

3.8. Configuración del servicio OpenVPN

3.8.1. Configuración básica

Para comenzar a configurar el servicio, copiamos el archivo de configuración de ejemplo:

```
patchispatch@vpn:~/EasyRSA-3.0.7$ sudo cp /usr/share/doc/openvpn/examples/sample-config-files/
server.conf.gz /etc/openvpn/
[sudo] password for patchispatch:
patchispatch@vpn:~/EasyRSA-3.0.7$ sudo gzip -d /etc/openvpn/server.conf.gz
patchispatch@vpn:~/EasyRSA-3.0.7$ _
```

En el archivo de configuración, buscamos la directiva `tls-auth`. Si la línea en la que se encuentra estuviese comentada, se ha de descomentar. Por lo general, no está comentada por defecto. También debemos descomentar la directiva `cipher` si no lo estuviera.

```
# For extra security beyond that provided
# by SSL/TLS, create an "HMAC firewall"
# to help block DoS attacks and UDP port flooding.
#
# Generate with:
#   openvpn --genkey --secret ta.key
#
# The server and each client must have
# a copy of this key.
# The second parameter should be '0'
# on the server and '1' on the clients.
tls-auth ta.key 0 # This file is secret

# Select a cryptographic cipher.
# This config item must be copied to
# the client config file as well.
# Note that v2.4 client/server will automatically
# negotiate AES-256-GCM in TLS mode.
# See also the ncp-cipher option in the manpage
cipher AES-256-CBC

# Enable compression on the VPN link and push the
# option to the client (v2.4+ only, for earlier
# versions see below)
/tls                                                    244,4         79%
```

Además, debemos añadir debajo de ‘cipher‘ la directiva `auth SHA256`.

```
# to help block DoS attacks and UDP port flooding.
#
# Generate with:
#   openvpn --genkey --secret ta.key
#
# The server and each client must have
# a copy of this key.
# The second parameter should be '0'
# on the server and '1' on the clients.
tls-auth ta.key 0 # This file is secret

# Select a cryptographic cipher.
# This config item must be copied to
# the client config file as well.
# Note that v2.4 client/server will automatically
# negotiate AES-256-GCM in TLS mode.
# See also the ncp-cipher option in the manpage
cipher AES-256-CBC

auth SHA256

# Enable compression on the VPN link and push the
# option to the client (v2.4+ only, for earlier
# versions see below)
;compress lz4-v2
"/etc/openvpn/server.conf" 317L, 10867C escritos                254,11         80%
```

Debemos buscar también la directiva `dh`, y cambiar el nombre de archivo para que corresponda con el que generamos anteriormente. En nuestro caso, debemos cambiar `dh2048.pem` por `dh.pem`.

```
# a unique Common Name for the server
# and each of the client certificates.
#
# Any X509 key management system can be used.
# OpenVPN can also use a PKCS #12 formatted key file
# (see "pkcs12" directive in man page).
ca ca.crt
cert server.crt
key server.key  # This file should be kept secret

# Diffie hellman parameters.
# Generate your own with:
#   openssl dhparam -out dh2048.pem 2048
dh dh.pem

# Network topology
# Should be subnet (addressing via IP)
# unless Windows clients v2.0.9 and lower have to
# be supported (then net30, i.e. a /30 per client)
# Defaults to net30 (not recommended)
;topology subnet

# Configure server mode and supply a VPN subnet
# for OpenVPN to draw client addresses from.
# The server will take 10.8.0.1 for itself,
"/etc/openvpn/server.conf" 317L, 10863C escritos                85,5          24%
```

Y finalmente descomentamos las líneas de usuario y grupo, ya que estamos utilizando un sistema distinto de Windows.

```
# enable it in the client config file.
;comp-lzo

# The maximum number of concurrently connected
# clients we want to allow.
;max-clients 100

# It's a good idea to reduce the OpenVPN
# daemon's privileges after initialization.
#
# You can uncomment this out on
# non-Windows systems.
user nobody
group nogroup

# The persist options will try to avoid
# accessing certain resources on restart
# that may no longer be accessible because
# of the privilege downgrade.
persist-key
persist-tun

# Output a short status file showing
# current connections, truncated
# and rewritten every minute.
"/etc/openvpn/server.conf" 317L, 10861C escritos                277,1         90%
```

3.8.2. Configuración opcional

A continuación se van a realizar cambios opcionales en la configuración, detallados en la guía seguida para la instalación. Para las distintas aplicaciones que proponemos sobre cómo utilizar esta tecnología en una granja web serán de mucha utilidad.

En concreto, vamos a **enviar la configuración de DNS a los clientes para que todo el tráfico se redirija a través de la conexión VPN**.
Pese a que con la configuración anterior es suficiente para establecer un túnel, no todo el tráfico tiene por qué atravesarlo. Para conseguir que así sea, debemos modificar algunas partes de la configuración de nuestro servidor.
Como explica la propia configuración, para que todo el tráfico se redirija al túnel VPN se debe habilitar la siguiente opción, borrando el `;`:

```
# If enabled, this directive will configure
# all clients to redirect their default
# network gateway through the VPN, causing
# all IP traffic such as web browsing and
# and DNS lookups to go through the VPN
# (The OpenVPN server machine may need to NAT
# or bridge the TUN/TAP interface to the internet
# in order for this to work properly).
;push "redirect-gateway def1 bypass-dhcp"
```

También se deben habilitar las dos directivas de debajo para permitir que ciertos tipos de tráfico también pasen por donde queremos:

```
# Certain Windows-specific network settings
# can be pushed to clients, such as DNS
# or WINS server addresses.  CAVEAT:
# http://openvpn.net/faq.html#dhcpcaveats
# The addresses below refer to the public
# DNS servers provided by opendns.com.
;push "dhcp-option DNS 208.67.222.222"
;push "dhcp-option DNS 208.67.220.220"
```

3.9. Configuración de red del servidor

Para que nuestra VPN funcione correctamente, debemos habilitar la redirección de IP. Para ello, debemos modificar el archivo `/etc/sysctl.conf`, y descomentar la siguiente línea:

```
# Uncomment the next line to enable packet forwarding for IPv4
#net.ipv4.ip_forward=1
```

Para configurar el enmascaramiento de red, debemos utilizar un firewall. En la instalación hemos utilizado ufw, pero antes de tocar la configuración, debemos identificar la interfaz pública de red ejecutando el siguiente comando: `ip route | grep default`.

```
patchispatch@vpn:~/EasyRSA-3.0.7$ ip route | grep default
default via 10.0.2.2 dev enp0s3 proto dhcp src 10.0.2.15 metric 100
```

Nuestra interfaz de red es `enp0s3`. Ahora debemos modificar el archivo `before.rules` de la configuración de ufw. Añadimos lo siguiente:

```
# REGLAS DE OPENVPN
# Tabla NAT
*nat
:POSTROUTING ACCEPT [0:0]
# Permitir tráfico desde el cliente OpenVPN a enp0s3
-A POSTROUTING -s 10.8.0.0/8 -o enp0s3 -j MASQUERADE
COMMIT
```

Para decirle a ufw que debe permitir paquetes redirigidos, modificamos el archivo `/etc/default/ufw`, añadiendo la línea siguiente:

DEFAULT_FORWARD_POLICY=“ACCEPT”

Por último, añadimos las excepciones necesarias para que el firewall permita el tráfico de OpenVPN.

OpenVPN utiliza por defecto el puerto 1194 mediante udp. Tanto el puerto como el protocolo pueden modificarse en la configuración, pero nosotros no lo hemos hecho, así que ejecutamos lo siguiente:

```
patchispatch@vpn:~/EasyRSA-3.0.7$ sudo ufw allow 1194/udp
Rule added
Rule added (v6)
```

Si OpenSSH no estuviese ya listado por ufw, sería necesario añadirlo también. Reiniciamos el servicio y todo estaría listo, al menos en esta parte.

3.10. Iniciar y habilitar el servicio OpenVPN

Para iniciar el servicio OpenVPN e indicarle que nuestra configuración se halla en el arhivo `server.conf`, ejecutamos `sudo systemctl start openvpn@server`. Después, comprobamos su estado:

```
patchispatch@vpn:~$ sudo systemctl start openvpn@server
[sudo] password for patchispatch:
patchispatch@vpn:~$ sudo systemctl status openvpn@server
● openvpn@server.service - OpenVPN connection to server
   Loaded: loaded (/lib/systemd/system/openvpn@.service; disabled; vendor preset: enabled)
   Active: active (running) since Tue 2020-05-19 15:53:24 UTC; 37s ago
     Docs: man:openvpn(8)
           https://community.openvpn.net/openvpn/wiki/Openvpn24ManPage
           https://community.openvpn.net/openvpn/wiki/HOWTO
 Main PID: 3573 (openvpn)
   Status: "Initialization Sequence Completed"
    Tasks: 1 (limit: 1108)
   CGroup: /system.slice/system-openvpn.slice/openvpn@server.service
           └─3573 /usr/sbin/openvpn --daemon ovpn-server --status /run/openvpn/server.status 1

may 19 15:53:24 vpn ovpn-server[3573]: Could not determine IPv4/IPv6 protocol. Using AF_INET
may 19 15:53:24 vpn ovpn-server[3573]: Socket Buffers: R=[212992->212992] S=[212992->212992]
may 19 15:53:24 vpn ovpn-server[3573]: UDPv4 link local (bound): [AF_INET][undef]:1194
may 19 15:53:24 vpn ovpn-server[3573]: UDPv4 link remote: [AF_UNSPEC]
may 19 15:53:24 vpn ovpn-server[3573]: GID set to nogroup
may 19 15:53:24 vpn ovpn-server[3573]: UID set to nobody
may 19 15:53:24 vpn ovpn-server[3573]: MULTI: multi_init called, r=256 v=256
may 19 15:53:24 vpn ovpn-server[3573]: IFCONFIG POOL: base=10.8.0.4 size=62, ipv6=0
may 19 15:53:24 vpn ovpn-server[3573]: IFCONFIG POOL LIST
may 19 15:53:24 vpn ovpn-server[3573]: Initialization Sequence Completed
lines 1-22/22 (END)
```

Para que arranque al iniciar, ejecutamos `sudo systemctl enable openvpn@server`.
Podemos comprobar que se ha creado la interfaz de red de VPN `tun0`:ç

```
patchispatch@vpn:~$ ip addr show tun0
4: tun0: <POINTOPOINT,MULTICAST,NOARP,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UNKNOWN group
 default qlen 100
    link/none
    inet 10.8.0.1 peer 10.8.0.2/32 scope global tun0
       valid_lft forever preferred_lft forever
    inet6 fe80::2d9f:23c0:5c6:d5f5/64 scope link stable-privacy
       valid_lft forever preferred_lft forever
```

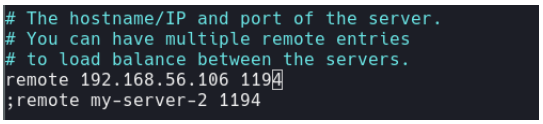
3.11. Creación de la infraestructura de la configuración de clientes

La VPN está lista, pero necesitamos configurar los clientes. Dado que cada cliente necesita su configuración personal, vamos a crear una infraestructura para que generar y almacenar las diferentes configuraciones de usuario.

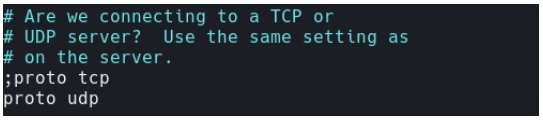
Para empezar, creamos la carpeta `~/client-configs/files`. Después, copiamos dentro una copia del archivo de configuración de ejemplo proporcionado por OpenVPN, que nos servirá como base.

```
cp /usr/share/doc/openvpn/examples/sample-config-files/client.conf
~/client-configs/base.conf
```

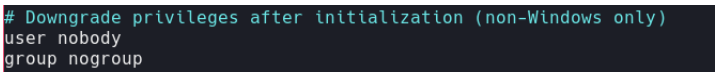
Debemos indicar la IP de nuestro servidor OpenVPN, además del puerto, en la directiva `remote`.



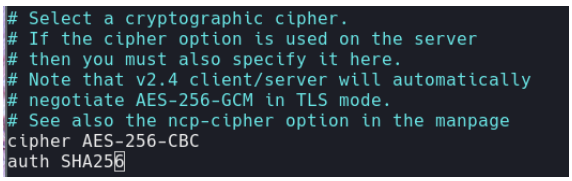
Debemos confirmar también que el protocolo seleccionado es UDP, ya que tiene que coincidir con el utilizado por el servidor.



De nuevo, descomentamos los campos de usuario y grupo:



Debemos cambiar la configuración de cifrado tal y como la tenemos en `/etc/openvpn/server.conf`



Para que la VPN funcione correctamente en el cliente debemos añadir la directiva `key-direction 1`. Además, vamos a añadir líneas comentadas que puede que necesitemos en los clientes de forma específica.

Este primer bloque de instrucciones es para aquellos clientes que **NO** utilizan `systemd-resolve` para la gestión de DNS:

```
; script-security 2
; up /etc/openvpn/update-resolv-conf
; down /etc/openvpn/update-resolv-conf
```

Este es para los que sí lo utilizan:

```
; script-security 2
; up /etc/openvpn/update-systemd-resolved
; down /etc/openvpn/update-systemd-resolved
; down-pre
; dhcp-option DOMAIN-ROUTE .
```

Ahora prepararemos un script que automatice la generación de configuracion junto con las claves necesarias, y lo sitúe todo en la carpeta `client-configs/files`. Abrimos el editor y creamos el script `make_config.sh` en la carpeta `client-configs`:

```
#!/bin/bash

# First argument: Client identifier

KEY_DIR=~/client-configs/keys
OUTPUT_DIR=~/client-configs/files
BASE_CONFIG=~/client-configs/base.conf

cat ${BASE_CONFIG} \
    <(echo -e '<ca>') \
    ${KEY_DIR}/ca.crt \
    <(echo -e '</ca>\n<cert>') \
    ${KEY_DIR}/${1}.crt \
    <(echo -e '</cert>\n<key>') \
    ${KEY_DIR}/${1}.key \
    <(echo -e '</key>\n<tls-auth>') \
    ${KEY_DIR}/ta.key \
    <(echo -e '</tls-auth>') \
    > ${OUTPUT_DIR}/${1}.ovpn
~
~
~
~
~
~
"make_config.sh" [Nuevo] 19L, 465C escritos                    19,29         Todo
```

Este script realizará una copia de la configuración base, recogerá los certificados y claves generadas añadiéndolas a la configuración y las colocará en el directorio correspondiente. No obstante, cada vez que se añada un cliente se deberá generar y firmar el certificado correspondiente, algo que vamos a hacer a continuación.

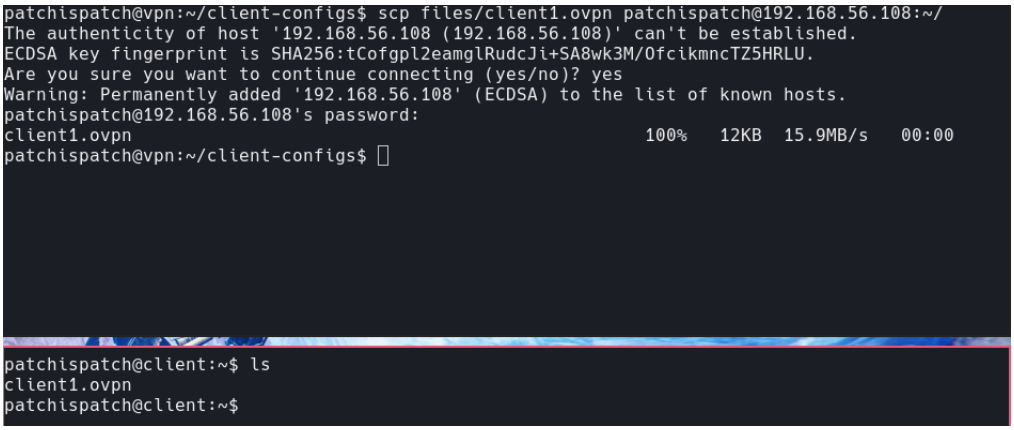
3.12. Generación de configuración de clientes

Como tenemos los certificados generados para un cliente llamado `client1` de un paso anterior, ejecutamos el script. Nos generará un archivo en la carpeta `files`, llamado `client1.ovpn`. Este archivo contiene la configuración y será instalado en el cliente.

```
patchispatch@vpn:~/client-configs$ sudo ./make_config.sh client1
[sudo] password for patchispatch:
patchispatch@vpn:~/client-configs$ ls -l files
total 12
-rw-r--r-- 1 root root 12042 may 19 19:56 client1.ovpn
patchispatch@vpn:~/client-configs$
```

Debemos transferir este archivo al cliente. Se ha creado una nueva máquina virtual con Ubuntu 18.04 para hacer de cliente de pruebas, llamada `client` y con IP estática `192.168.56.108`.

Realizamos la transferencia mediante `scp`, aunque dependiendo del sistema del cliente tendremos que usar diferentes métodos —que cubriremos más adelante—, y una vez finalizada tendremos el archivo listo en en la carpeta home de nuestro cliente.



3.13. Instalación de la configuración del cliente

Dependiendo de las características del cliente, deberemos utilizar diferentes métodos. En este caso realizaremos las pruebas en GNU/Linux y Android.

3.14. GNU/Linux

En GNU/Linux, la forma más sencilla y global de configurar el cliente es utilizar el software de OpenVPN. Instalamos con `sudo apt install openvpn`, como hicimos al principio, ya que estamos utilizando Ubuntu 18.04.

Debemos comprobar si nuestro cliente utiliza `systemd-resolved`. Para ello, comprobamos el archivo `/etc/resolv.conf`:

```
patchispatch@client:~$ cat /etc/resolv.conf
# This file is managed by man:systemd-resolved(8). Do not edit.
#
# This is a dynamic resolv.conf file for connecting local clients to the
# internal DNS stub resolver of systemd-resolved. This file lists all
# configured search domains.
#
# Run "systemd-resolve --status" to see details about the uplink DNS servers
# currently in use.
#
# Third party programs must not access this file directly, but only through the
# symlink at /etc/resolv.conf. To manage man:resolv.conf(5) in a different way,
# replace this symlink by a static file or a different symlink.
#
# See man:systemd-resolved.service(8) for details about the supported modes of
# operation for /etc/resolv.conf.

nameserver 127.0.0.53
options edns0
search home
```

Como la dirección IP que aparece es `127.0.0.53`, sabemos que nuestro cliente está utilizando `systemd-resolved`. Para dar soporte a este tipo de clientes, debemos instalar el paquete `openvpn-systemd-resolved`. De esta forma se forzará al servicio a utilizar la conexión VPN al enviar las resoluciones y consultas DNS.

Ahora debemos abrir el archivo de configuración del cliente y descomentar las líneas que añadimos anteriormente correspondientes a systemd-resolved:

```
# Para clientes que NO utilizan systemd-resolve
;script-security 2
;up /etc/openvpn/update-resolv-conf
;down /etc/openvpn/update-resolv-conf

# Para clientes que SÍ utilizan systemd-resolve
script-security 2
up /etc/openvpn/update-systemd-resolved
down /etc/openvpn/update-systemd-resolved
down-pre
dhcp-option DOMAIN-ROUTE .
```

Una vez hecho esto, podemos conectarnos ejecutando `openvpn` indicando el fichero de configuración.

```
Tue May 19 21:46:36 2020 TUN/TAP TX queue length set to 100
Tue May 19 21:46:36 2020 do_ifconfig, tt->did_ifconfig_ipv6_setup=0
Tue May 19 21:46:36 2020 /sbin/ip link set dev tun0 up mtu 1500
Tue May 19 21:46:36 2020 /sbin/ip addr add dev tun0 local 10.8.0.6 peer 10.8.0.5
Tue May 19 21:46:36 2020 /etc/openvpn/update-systemd-resolved tun0 1500 1552 10.8.0.6 10.8.0.5
 init
<14>May 19 21:46:36 update-systemd-resolved: Link 'tun0' coming up
<14>May 19 21:46:36 update-systemd-resolved: Adding DNS Routed Domain .
<14>May 19 21:46:36 update-systemd-resolved: Adding IPv4 DNS Server 208.67.222.222
<14>May 19 21:46:36 update-systemd-resolved: Adding IPv4 DNS Server 208.67.220.220
<14>May 19 21:46:36 update-systemd-resolved: SetLinkDNS(5 2 2 4 208 67 222 222 2 4 208 67 220
220)
<14>May 19 21:46:36 update-systemd-resolved: SetLinkDomains(5 1 . true)
Tue May 19 21:46:36 2020 /sbin/ip route add 192.168.56.106/32 via 10.0.2.2
RTNETLINK answers: File exists
Tue May 19 21:46:36 2020 ERROR: Linux route add command failed: external program exited with e
rror status: 2
Tue May 19 21:46:36 2020 /sbin/ip route add 0.0.0.0/1 via 10.8.0.5
Tue May 19 21:46:36 2020 /sbin/ip route add 128.0.0.0/1 via 10.8.0.5
Tue May 19 21:46:36 2020 /sbin/ip route add 10.8.0.1/32 via 10.8.0.5
Tue May 19 21:46:36 2020 GID set to nogroup
Tue May 19 21:46:36 2020 UID set to nobody
Tue May 19 21:46:36 2020 WARNING: this configuration may cache passwords in memory -- use the
auth-nocache option to prevent this
Tue May 19 21:46:36 2020 Initialization Sequence Completed
```

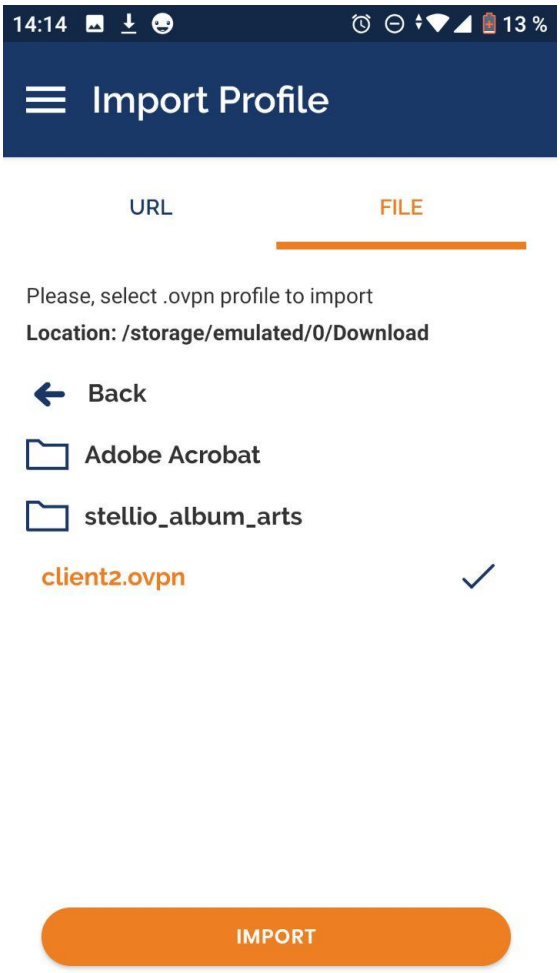
3.14.1. Android

Nota: *para acceder a la VPN desde fuera del host de la máquina virtual y una red distinta, se debe configurar un adaptador puente en la máquina virtual y abrir y configurar los respectivos puertos del router. Para conectarnos desde el móvil en la misma red, basta con configurar un adaptador puente.*

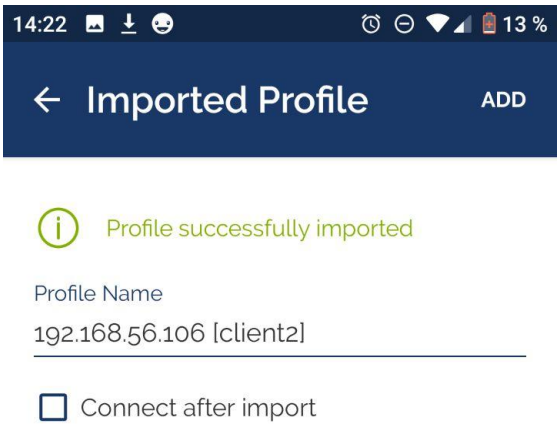
Para empezar, creamos el certificado firmado para un nuevo cliente, `client2`, siguiendo el procedimiento expuesto en el punto **Generación del certificado y las claves del cliente**.

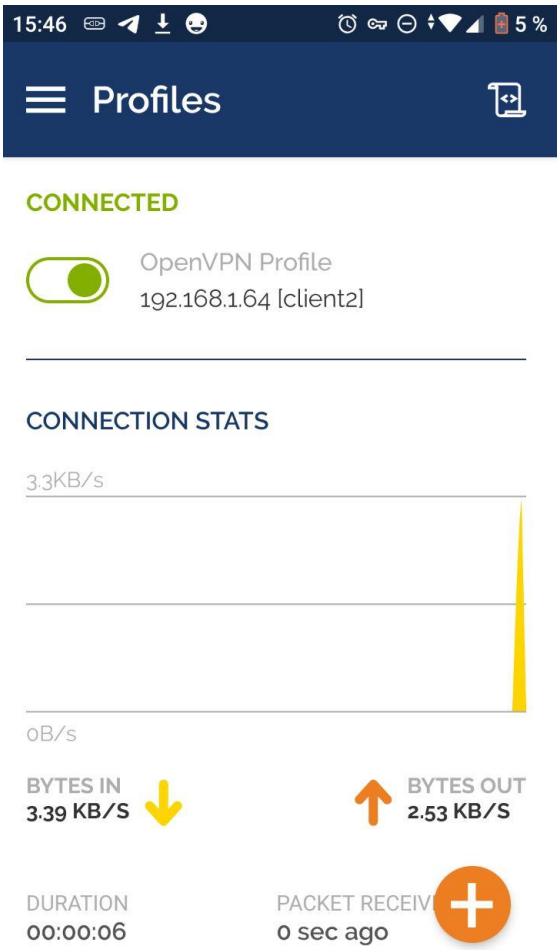
Una vez disponemos del archivo `client2.ovpn`, debemos configurar el cliente desde nuestro móvil. Para ello, transferimos el archivo, ya sea por USB, ssh o cualquier medio.

Descargamos la aplicación **OpenVPN Connect**, la aplicación oficial de OpenVPN. Una vez tenemos el archivo en nuestro dispositivo, la abrimos y seleccionamos nuestro archivo desde la opción *File*:



Una vez importado correctamente el perfil podremos cambiar su nombre, y podremos iniciar la conexión. Si todo va bien, la aplicación nos indicará que estamos conectados a nuestra VPN, y aparecerá una notificación mientrar lo sigamos estando.


14:22
13 %
Imported Profile
ADD
Profile successfully imported
Profile Name
192.168.56.106 [client2]
Connect after import


15:46
5 %
Profiles
CONNECTED
OpenVPN Profile
192.168.1.64 [client2]
CONNECTION STATS
3.3KB/s
0B/s
BYTES IN
3.39 KB/S
BYTES OUT
2.53 KB/S
DURATION
00:00:06
PACKET RECEIV
0 sec ago

4. Aplicaciones a centros de datos y seguridad

El proceso de instalación y configuración del servicio ya está terminado, y nos da una idea de cómo proceder a la hora de utilizarlo de forma relativamente sencilla. La tecnología VPN ofrece una serie de características que pueden ser muy útiles en granjas de servidores. Exploraremos dos usos diferentes para esta tecnología:

4.1. Uso 1: asegurar el acceso remoto a una granja web

Ya sea por tratarse de una red interna de trabajo o por cuestiones de mantenimiento, se puede utilizar una conexión VPN y un firewall para limitar quién puede acceder a las diferentes máquinas de nuestro centro de datos físico.

Permitiendo acceso únicamente a través del túnel VPN para ciertos servicios como por ejemplo SSH, podemos distribuir certificados a los distintos encargados de mantenimiento de nuestros servidores de contenido, permitiendo así una gestión más cómoda sin comprometer la seguridad de nuestra granja.

Podemos limitar el acceso de diferentes maneras:

Limitar el acceso a toda la red

Esta configuración es útil cuando queremos establecer una red de trabajo distribuida y permitir el acceso a empleados y miembros de la institución. La UGR utiliza esta configuración para varios de sus servicios, como el *acortador de URLs* o el servidor Turing de la ETSIIT, a los que se puede acceder únicamente desde la red de la universidad. Si no se dispone de conexión a eduroam, se puede acceder a través de la VPN de la UGR.

Limitar el acceso a los servidores de contenido

Esta configuración no restringe el acceso al balanceador de carga, y permite que se sirvan los contenidos deseados a cualquier usuario, pero para facilitar el mantenimiento si hay algún error o se desea modificar algo de los servidores de contenido de forma remota se puede establecer acceso mediante SSH por VPN.

De esta forma no es necesario mantener un registro de las diferentes IPs de los trabajadores en las reglas del firewall para poder acceder, únicamente es necesario poseer el certificado correspondiente. Hemos visto que generar certificados es sencillo y automatizable, por lo que de cara a conceder acceso supone una opción más cómoda que modificar la configuración del firewall de cada máquina, y a la hora de revocar permisos basta con anular el certificado, algo que también es sencillo.

4.2. Uso 2: construir un centro de datos de forma distribuida

La tecnología VPN permite en esencia recrear una red local de forma distribuida, así que podemos utilizar esto para construir un centro de datos y asegurar las transferencias de información entre servidores.

Al estar las comunicaciones cifradas, los ataques *man in the middle* no suponen un problema, pero hay que tener en cuenta que el tráfico pasará por el servidor VPN, pudiendo saturarse o fallar, y condicionando el estado de la red entera. Para evitar esto se debe proporcionar una infraestructura más compleja, con varios servidores VPN.

El acceso también estaría restringido gracias a los certificados, siempre y cuando configuremos debidamente los servidores y sus respectivos cortafuegos. No todo el tráfico tiene por qué pasar por el túnel VPN, y se puede permitir el acceso desde fuera de la red privada a diferentes protocolos y servicios, como HTTPS en el balanceador de carga.

OpenVPN es bastante flexible en cuanto a la configuración se refiere, por lo que se podría aplicar, por ejemplo, a proteger la duplicación de datos y el acceso a los servidores de contenido, dejando libre el tráfico de datos menos sensibles para no saturar el túnel.

5. Conclusión

La tecnología VPN es mucho más flexible de lo que esperaba encontrar en un principio, y dado que ahora está pasando por una nueva etapa de popularidad debido a los servicios de suscripción y poder cambiar tu identidad en línea, me llamaba la atención explorar cómo podía aplicarse a los contenidos de la asignatura. Que la instalación requiera editar archivos de configuración de red y construir una entidad certificadora SSL me ha parecido muy formativo, y la gran cantidad de opciones de las que dispone el servicio OpenVPN, así como su capacidad multiplataforma y gran comunidad y documentación, han ayudado a comprender mejor los posibles usos de esta herramienta.

6. Bibliografía

https://www.digitalocean.com/community/tutorials/
initial-server-setup-with-ubuntu-18-04
https://www.digitalocean.com/community/tutorials/
how-to-set-up-an-openvpn-server-on-ubuntu-18-04
https://askubuntu.com/questions/920486/
openvpn-config-client-ovpn-fail-rtnetlink-answers-file-exists
https://community.openvpn.net/openvpn/wiki
https://openvpn.net/community-resources/
configuring-client-specific-rules-and-access-policies/
https://openvpn.net/community-resources/ethernet-bridging/
https://openvpn.net/community-resources/installing-openvpn/
https://openvpn.net/community-resources/
setting-up-your-own-certificate-authority-ca/