OpenVPN

Departamento de Sistemas Telemáticos y Computación (GSyC)

gsyc-profes (arroba) gsyc.es

Abril de 2014

Características de OpenVPN

OpenVPN es una aplicación con licencia libre (GPL) para implementar redes privadas virtuales

- Desarrollado inicialmente por James Yonan en 2002. En la actualidad, por el OpenVPN project y OpenVPN Technologies, inc.
- Seguridad robusta, basado en SSL/TLS
- Es sencillo, potente, seguro, aunque no está tan extendido como otras soluciones
- Disponible para MS Windows, Linux, MacOS, xBSD, Solaris, android e iOS
- No es compatible con las VPN de routers dedicados (Cisco, Juniper, etc)

- Multiplexa todos los puertos en uno solo (por omisión, 1194 udp o 1194 tcp), solo necesita un protocolo (udp o tcp)
  - Esto facilita mucho su integración con NAT y cortafuegos
- Arquitectura cliente servidor, soporta IP dinámica en ambos extremos, manteniendo la sesión
- Trabaja en espacio de usuario
- Soporta tarjetas inteligentes PKCS#11 (similares al DNIe)
- Puede ofrecer un nivel de red (con el dispositivo *tun*) o un nivel de enlace (con el dispositivo *tap*)
  - En MS Windows, solamente nivel de enlace

Administración del demonio de OpenVPN

- El demonio de OpenVPN se inicia y se detiene como es habitual en todos los demonios
  - `/etc/init.d/openvpn start`
    Inicia el servicio
  - `/etc/init.d/openvpn stop`
    Detiene el servicio
  - `/etc/init.d/openvpn restart`
    Detiene e inicia el servicio. Relee los ficheros de configuración
- Envía sus logs a `/var/log/syslog`
- Los ficheros de configuración principales son
  - `/etc/openvpn/server.conf`
  - `/etc/openvpn/client.conf`

Autenticación mediante secreto compartido

Cliente y servidor deben autenticarse. Hay dos maneras

1. Secreto compartido, criptografía tradicional
   - Generamos la clave

     ```
     openvpn   --genkey --secret key.txt
     ```

   - Copiamos esta clave en el cliente y en el servidor, en el directorio que estimemos oportuno
   - En `server.conf` y `client.conf`, con la directiva *secret* indicamos el nombre de este fichero, indicando el *path* completo. P.e.
     `secret /dir1/dir2/key.txt`
2. Certificados X.509

Direcciones IP

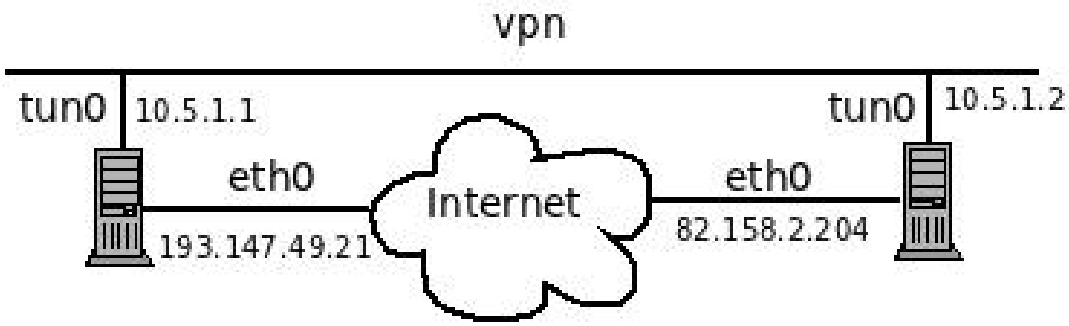


Figura : VPN Mínima

Toda máquina que forme parte de una VPN tendrá al menos dos interfaces y dos direcciones

- Una dirección del exterior de la VPN, preexistente, típicamente pública. En este ejemplo, 193.147.49.21 y 82.158.2.204
- Una dirección del interior de la VPN, típicamente privada. En este ejemplo, 10.5.1.1 y 10.5.1.2

Es necesario que pertenezcan a segmentos distintos (y no confundirlas)

Recordatorio: En IPv4 los rangos reservados para direcciones privadas son

```
10.0.0.0/8
172.16.0.0/12
192.168.0.0/16
```

Para las direcciones del interior de la VPN podemos emplear cualquiera de estos rangos

- Con tal de que evitamos los rangos más comunes: 10.0.0.0/24, 192.168.0.0/24 y 192.168.1.0/24, que podrían estar ya siendo usadas en nuestra LAN

VPN mínima

Para una VPN sencilla, ofreciendo nivel de red y con autenticación por secreto compartido, bastan los siguientes ficheros:

`/etc/openvpn/server.conf`

```
dev tun
ifconfig 10.5.1.1  10.5.1.2
secret /etc/openvpn/key.txt
```

`/etc/openvpn/client.conf`

```
remote 193.147.49.21
dev tun
ifconfig 10.5.1.2  10.5.1.1
secret /etc/openvpn/key.txt
```

VPN ofreciendo nivel de red

Estamos configurando una VPN ofreciendo nivel de red
Por tanto, esta VPN

- Es un segmento de red propio, un único dominio de colisión
  - Las tramas de broadcast de la VPN no salen de la VPN
- Emplea el dispositivo *tun* (no el dispositivo *tap*)
- Trabaja con datagramas IP
- Será necesario contar con las tablas de encaminamiento adecuado

VPN con parámetros adicionales

La manera más habitual de configurar OpenVPN es

1. Hacer una copia de los ficheros `server.conf` y `client.conf` de ejemplo, que contienen parámetros adicionales, comentados
   - La versión estándar está en

     `/usr/share/doc/openvpn/examples/sample-config-files/server.conf.gz`
     `/usr/share/doc/openvpn/examples/sample-config-files/client.conf`

   - En el guión de prácticas de la asignatura encontrarás versiones simplificadas de estos ficheros
2. Editarlos, adaptándolos a nuestras necesidades

Servidor repartiendo direcciones a clientes

- En el ejemplo *VPN mínima*, indicábamos las direcciones del interior de la VPN de forma explícita con la directiva *ifconfig*
- Ahora, haremos que el servidor OpenVPN reparta automáticamente a los clientes las direcciones del interior (de forma muy similar a DHCP)
- Esto suele resultar más cómodo, es lo que hacen los ficheros de ejemplo `server.conf` y `client.conf`

En el fichero `server.conf` editamos esta directiva:

```
# 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,
# the rest will be made available to clients.
# Each client will be able to reach the server
# on 10.8.0.1. Comment this line out if you are
# ethernet bridging. See the man page for more info.

server 10.8.0.0 255.255.255.0
```

- En este ejemplo el servidor tomará la dirección 10.8.0.1 (la primera dirección de la red indicada), y le dará a los clientes otras direcciones, desconocidas a priori, dentro de este rango

Para que el administrador en el servidor pueda saber qué direcciones tienen los clientes, editamos esta directiva en `server.conf`

```
# Maintain a record of client <-> virtual IP address
# associations in this file.  If OpenVPN goes down or
# is restarted, reconnecting clients can be assigned
# the same virtual IP address from the pool that was
# previously assigned.

ifconfig-pool-persist /etc/openvpn/ipp.txt
```

- Dentro `ipp.txt`, el servidor almacena qué dirección IP del interior de la VPN le da a cada cliente
- `ipp.txt` no se escribe inmediatamente
  - Hay que esperar unos 10 minutos
  - O bien ordenarle al demonio que relea y escriba los ficheros de configuración, con `/etc/init.d/openvpn reload`

En `client.conf` no hay que hacer nada especial, simplemente mantener esta directiva

```
# Specify that we are a client and that we
# will be pulling certain config file directives
# from the server.

client
```

E indicar la dirección (del exterior de la VPN) del servidor

```
# The hostname/IP and port of the server.
# You can have multiple remote entries
# to load balance between the servers.

remote 193.147.49.21   1194
```

Autenticación X.509

El mecanismo de autenticación preferido en OpenVPN es X.509

- En OpenVPN no se puede usar secreto compartido para ofrecer VPN de nivel de enlace
- Emplearemos claves (privadas) y certificados (Un certificado es esencialmente una clave pública firmada por una CA, *certification authority*)
- Para que la clave pública de una máquina tenga un certificado firmado por una CA reconocida universalmente tendríamos que pagar la tarifa correspondiente a una empresa como GeoTrust o digicert
- En nuestras prácticas, cada estudiante será una CA (lo que no es tan raro, un administrador como CA de sus propias máquinas)

Pasos para configurar la red con X.509

1. Copiar a un directorio de trabajo los scripts Easy-RSA 2.0
2. Crear la CA
3. Crear, en la máquina que contiene la clave de la CA
   - Clave y certificado del servidor
   - Clave(s) y certificado(s) del cliente(s)
   - Fichero con parámetros Diffie Hellman
4. Llevar al servidor, por canal seguro, parámetros Diffie Hellman, clave y certificado del servidor.
   Llevar a cada cliente, por canal seguro, su clave y su certificado
5. Configurar `server.conf` y `client.conf`, indicando dónde están los parámetros, claves y certificados

Método alternativo para generar claves

1. Crear las claves en servidor y cliente(s)
2. Enviar a la CA un *Certificate Signing Request (CSR)*
3. La CA devuelve a cada máquina su certificado

Esté método tiene la ventaja de que las claves privadas no salen de cada máquina, y por tanto no hace falta un canal seguro
Pero aquí no lo veremos, puesto que es un poco más largo y sí tenemos un canal seguro (ssh)

1-Copiar a directorio de trabajo scripts Easy-RSA

- Copiamos los scripts a cualquier directorio donde podamos trabajar
  En una máquina Debian/Ubuntu, copiaremos todo el contenido de
  `/usr/share/doc/openvpn/examples/easy-rsa/2.0`
- Editamos las últimas líneas del fichero `vars` para identificar al emisor del certificado.

```
export KEY_COUNTRY="US"
export KEY_PROVINCE="CA"
export KEY_CITY="SanFrancisco"
export KEY_ORG="Fort-Funston"
export KEY_EMAIL="me@myhost.mydomain"
export KEY_EMAIL=mail@host.domain
export KEY_CN=changeme
export KEY_NAME=changeme
export KEY_OU=changeme
export PKCS11_MODULE_PATH
export PKCS11_PIN
```

X.509 emplea el concepto *Distinguished Name* definido en la norma X.500

- Código ISO 3166-1 de España:*ES*.
- El nombre de la provincia se escribe completo
- Los campos de más de una palabra deben entrecomillarse

Algunos campos no obvios:

- KEY_ORG
  Organismo que emite el certificado
- KEY_NAME
  Organismo que emite el certificado
- KEY_CN
  Common Name. Nombre de la persona concreta o de la máquina concreta que emite el certificado
- KEY_OU
  Organizational Unit. Departamento, empresa asociada, etc
- PKCS11_MODULE_PATH=
  Directorio donde están los módulos de criptografía PKCS11. Podemos ignorarlo si no usamos *smart cards* (DNIe)
- PKCS11_PIN
  PIN de la tarjeta. Podemos ignorarlo

2-Crear la CA

Siempre dentro del directorio de trabajo, ejecutamos

```
source ./vars
./clean-all
./build-ca
```

Contestamos a todas las preguntas con el valor predeterminado (pulsando intro)

3.1-Crear fichero con parámetros Diffie Hellman

Seguimos en el directorio de trabajo, y ejecutamos

```
./build-dh
```

3.2-Crear claves y certificados del servidor

```
./build-key-server nombre_del_servidor
```

Contestamos a todo con el valor predeterminado (intro), excepto en las dos últimas preguntas:

```
Sign the certificate? [y/n]:
1 out of 1 certificate requests certified, commit? [y/n]
```

Aquí introduciremos 'y' (yes)

3.3 Crear claves y certificados del cliente(s)

Ejecutamos

```
./build-key nombre_del_cliente
```

Contestamos igual que en el paso anterior (todo intro, excepto 'y' en las dos últimas preguntas)

- Si intentamos generar 2 veces un certificado para la misma máquina, nos dará un error: *failed to update database, TXT_DB error number 2*
- Para arreglarlo, borramos la línea correspondente a este certificado en el fichero *index.txt* del subdirectorio *keys*

4-Llevar ficheros a servidor y clientes

Tras ejecutar todos los pasos anteriores, (suponiendo los nombres *myserver*, *client1* y *client2*) ahora el subdirectorio *keys* contendrá:

Fichero	**Contenido**	**Necesario en**	**Secreto**
ca.crt	Certificado raiz de la CA	Servidor y todos clientes	No
ca.key	Clave priv. de la CA	Solo en máquina donde se firma	Sí
dh1024.pem	Parámetros Diffie Hellman	Servidor	No
myserver.crt	Certificado del servidor	Servidor	No
myserver.key	Clave priv. del servidor	Servidor	Sí
client1.crt	Certificado del cliente1	Cliente1	No
client1.key	Clave priv. de cliente1	Cliente1	Sí
client2.crt	Certificado del cliente2	Cliente2	No
client2.key	Clave priv. de cliente2	Cliente2	Sí

Ficheros empleados en autenticación X.509

- Movemos cada fichero a la(s) máquina(s) correspondiente(s)
- El fichero `ca.key` deberíamos guardarlo en lugar seguro, no en una máquina en la red

5.1 Configuración de server.conf

Suponiendo que almacenamos los ficheros en `/etc/openvpn`, editamos el fichero `/etc/openvpn/server.conf` de la máquina *myserver* para que contenga:

```
# 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 /etc/openvpn/ca.crt
cert /etc/openvpn/myserver.crt
key /etc/openvpn/myserver.key  # This file should be kept secret

# Diffie hellman parameters.
# Generated by build-dh script
# Substitute 1024 for 2048 if you are using
# 2048 bit keys.
dh /etc/openvpn/dh1024.pem
```

5.2 Configuración de client.conf

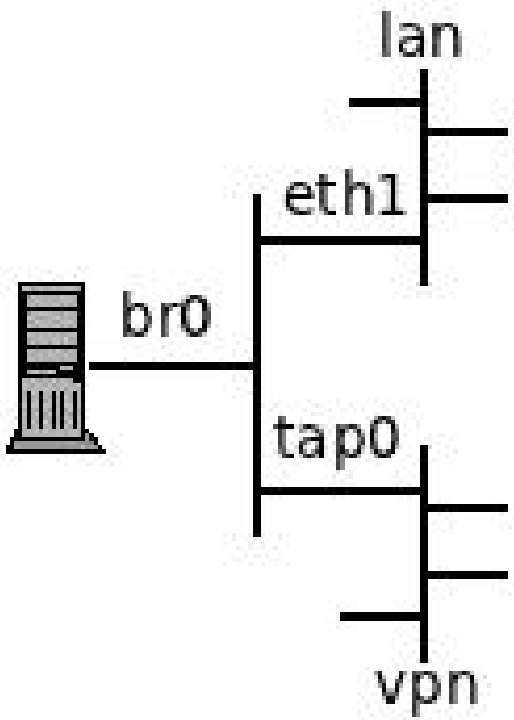
Suponiendo que almacenamos los ficheros en `/etc/openvpn`, editamos el fichero `/etc/openvpn/client.conf` en la máquina *client1* para que contenga:

```
# SSL/TLS parms.
# See the server config file for more
# description.  It's best to use
# a separate .crt/.key file pair
# for each client.  A single ca
# file can be used for all clients.
ca /etc/openvpn/ca.crt
cert /etc/openvpn/client1.crt
key /etc/openvpn/client1.key
```

VPN ofreciendo nivel de enlace

Ahora configuraremos una VPN ofreciendo nivel de enlace
Por tanto, esta VPN

- Es un segmento de red que se puede combinar con una LAN, uniendo el dominio de colisión de ambas redes
  - Las tramas de broadcast se propagan entre la VPN y la LAN
- Emplea el dispositivo *tap0* (no el dispositivo *tun*)
- Trabaja con tramas ethernet
- No será necesario modificar las tablas de encaminamiento
- En la LAN no puede haber MAC repetidas. Si tenemos MV clonadas desde la misma imagen, habrá que cambiar las MAC



- Para unir el segmento de red de la VPN con el segmento de la LAN, usamos un *bridge*
- El *bridge* es un nuevo interfaz de red, en este ejemplo, `br0`
- `eth1` es un interfaz con una dirección del interior de la VPN
- `eth1` y `tap0` pierden su identidad y su dirección IP: ahora el interfaz relevante es `br0`

- Para hacer un *bridge*, usamos una utilidad propia del Sistema Operativo, no es parte de OpenVPN.
  En el caso de Debian/Ubuntu, el paquete `bridge-utils`
  `apt-get install bridge-utils`,
- OpenVPN incluye un script que se encarga de configurar el *bridge*
  Está en

  `/usr/share/doc/openvpn/examples/sample-scripts/bridge-start`

  Lo copiamos a algún directorio de trabajo, y allí lo editamos

Fichero bridge-start

```
br="br0"  # Nombre de interfaz del puente

tap="tap0" # Nombre del interfaz tap

eth="eth0"  # Nombre del interfaz de la LAN
            # (El interfaz interior de la VPN)
eth_ip="192.168.8.4"  # Parámetros del interfaz LAN
eth_netmask="255.255.255.0"
eth_broadcast="192.168.8.255"
```

- En `br`, `tap` y `eth` indicamos los nombres de cada interfaz
- En `eth_ip`, `eth_netmask` y `eth_broadcast` indicamos los parámetros que tenía el interfaz de la LAN antes de incluirlo en el *bridge*
- Estos serán los parámetros que tendrá el puente

Editamos y ejecutamos este script antes de poner en marcha el demonio de OpenVPN