

Práctica VPN

Contents

1	Introducción teórica	2
1.1	PPTP (Point to Point Tunneling Protocol)	2
1.2	L2TP (Layer 2 Tunneling Protocol)	2
1.3	IPSec (Internet Protocol Security)	2
1.4	OpenVPN	3
1.5	Tunnel SSH	3
1.5.1	Redireccionado local	4
1.5.2	Redireccionado remoto	5
1.5.3	Redireccionado dinámico de puertos	5
1.6	Otras alternativas	6
2	Descripción	7
2.1	Situación de partida	7
2.2	Instalación	8
2.3	Preparación inicial	9
3	VPN enrutada	10
3.1	Creación de las claves dinámicas	10
3.2	Servidor VPN en oficina central y cliente VPN en sucursal	12
3.2.1	Prueba desde línea de comandos	13
3.2.2	Configuración de servidor y clientes	13
4	Topología VPN con un servidor y varios clientes	14
5	Troubleshooting	16
6	Líneas Futuras	17

1 Introducción teórica

VPN es un acrónimo de Virtual Private Network (Red privada virtual). Es un término referido a cualquier dispositivo que es capaz de crear un tunel semipermanente a través de una red pública entre dos hosts o redes privadas utilizando cualquier protocolo de aplicación. La encriptación se hace de esa forma a través del enlace y a través de las aplicaciones. Se utiliza para unir de forma segura redes o dispositivos que atraviesan redes inseguras, de forma que tienen una funcionalidad idéntica a si esas redes están directamente conectadas a través de enlaces privados. Otro uso muy común es el de proveer acceso remoto a usuarios remotos de una organización que estén en tránsito o que realicen trabajo a distancia.

Para conocer el funcionamiento de las VPN, es necesario utilizar dos conceptos: el concepto de tunel de red y la encriptación.

Los protocolos que implementan el tunelado (tunneling) de red permiten encapsular el tráfico de una pila de protocolos dentro de un único protocolo para poder replicarlo en otro segmento de la red. La encriptación permite ocultar los datos enviados por una red evitando que el tráfico enviado a través de una red pública (Internet) pueda ser descifrado.

La encriptación es fundamental para proteger los datos que atraviesan la red pública, y se realiza fundamentalmente en el campo de datos de los paquetes IP.

Existen diferentes soluciones abiertas y comerciales para cumplir las funciones de una VPN, entre las que se destacan PPTP, L2TP, IPSec y OpenVPN.

1.1 PPTP (Point to Point Tunneling Protocol)

Es uno de los primeros protocolos de tunneling y ha sido ampliamente empleado debido a su sencillez. Fue creado por Microsoft, Ascend Communications (Alcatel-Lucent) y 3Com entre otros. Este protocolo ha sido ampliamente usado aunque no ha llegado a ser un estándar (RFC). Además, se han encontrado múltiples fallos de seguridad que desaconsejan su uso (<http://poptop.sourceforge.net/dox/protocol-security.phtml>).

Para construir un túnel PPTP se abre un canal de control (habitualmente mediante una conexión TCP al puerto 1723 del servidor) que se emplea para iniciar y gestionar un segundo canal utilizando el protocolo GRE (Generic Routing Encapsulation, RFC 2784, protocolo con ID 47 en la cabecera IP) que se usa para intercambiar paquetes IP entre los dos extremos de un túnel.

En la actualidad se puede desplegar con PopTop (<http://poptop.sourceforge.net/>, servidor PPTP para Linux) y emplear el soporte nativo incluido en los sistemas operativos Windows y Mac OS o el software PPTP Client (<http://pptpclient.sourceforge.net/>) para otros sistemas operativos. También es posible encontrar clientes PPTP incluidos en el firmware de gran cantidad de routers. Es necesario tener en cuenta, que muchos proveedores de Internet impiden el tráfico GRE imposibilitando el uso de este protocolo de tunneling.

1.2 L2TP (Layer 2 Tunneling Protocol)

Fue propuesto en 1999 como estándar (RFC 2661) y está inspirado en PPP, L2F y PPTP. Actualmente se emplea la versión 3 que se publicó en 2005 (RFC 3931). L2TP funciona sobre UDP (puerto 1701) y por defecto no realiza ningún tipo de encriptación (salvo que se use sobre IPSEC). Además, L2TP es capaz de encapsular los paquetes de capa 2 lo cual permite implementar servicios como Wake on LAN. Es posible desplegar un servidor L2TP en Linux mediante *xl2tpd*. Existen clientes de L2TP implementados en el firmware de muchos routers debido a que L2TP ha sido usado por operadoras para vender conectividad de capa 2 en la tecnología ADSL.

1.3 IPSec (Internet Protocol Security)

IPSec es un conjunto de protocolos que permiten crear túneles con los que enviar el tráfico IP. Implementa autenticación, comprobación y encriptación sobre cada paquete IP y negociación de parámetros de criptografía durante la sesión. A excepción de IKE (Internet Key Exchange) y NAT-T (NAT Traversal), todos los protocolos empleados se han desarrollado en la capa de transporte. IPSec puede emplear entre otros, los siguientes protocolos:

- AH (Authentication Header): El protocolo AH (protocol ID 51) permite garantizar la integridad y autenticación de la cabecera de todos los paquetes IP comunicados por el túnel. Para ello, calcula un Hash Message Authentication Code (HMAC) a través de un algoritmo hash que opera teniendo en cuenta una clave secreta, el contenido del paquete IP y las partes inmutables del datagrama.

- ESP (Encapsulation Security Payloads): El protocolo ESP (50) incluye cabecera y campos para dar soporte a la encriptación y a autenticación (opcional ya que se puede no usar encriptación). Es el encargado de transportar los contenidos y provee además, los modos de transporte y túnel.

OpenSwan es una implementación completa de IPSec para Linux que permite el despliegue de servidores. Existen clientes IPSec en Windows y Mac OS X.

1.4 OpenVPN

OpenVPN (<http://openvpn.net/>) es la solución más sencilla y a la vez privada de tunneling. Usa SSL/TLS para implementar túneles con compresión sobre los que se puede encapsular todo el tráfico IP (a nivel de capa 3) y funciona enteramente en capa de aplicación sobre un único puerto (TCP o UDP a elección del administrador de red aunque habitualmente se despliega en el puerto 1194). Muchos usuarios emplean esta solución para evitar cortafuegos de proveedores de Internet y configuran los servidores para el uso del puerto 443 TCP (usado habitualmente para HTTPS).

Para hacer funcionar *OpenVPN* es necesario crear artificialmente una autoridad certificadora y pares de claves (certificados) para servidor y clientes. Este proceso se hace con openssl aunque en las últimas versiones de *openvpn*, se incluye un pequeño toolkit de scripts llamado *easyrsa* que facilita la creación de los certificados.

Un buen artículo sobre configuración de VPN se puede encontrar en esta dirección: <http://linuxconfig.org/vpn-virtual-private-network-and-openvpn>.

Funciona como cliente y servidor aunque no de forma nativa, en todos los sistemas operativos (Windows, MAC OS X y Linux). Existen distintas aplicaciones para realizar los ficheros de configuración o conectar de forma visual incluyendo Tunnelblick para Mac OS X, KVPnc para Linux, o OpenVPN Gui para Windows (<http://openvpn.se/>). Finalmente, existe múltiple documentación por Internet para desplegar un servidor VPN en los routers que pueden ejecutar Linux con las distribuciones DDWRT y OpenWRT.

El funcionamiento de *openvpn* depende en gran medida del tipo de claves utilizadas para la encriptación del tráfico entre el servidor y los clientes. Si el cliente es único, ya sea un sólo host o una red local a través de una pasarela, se pueden utilizar tanto claves estáticas como dinámicas, pero si existen varios clientes o varias redes locales que se conectan a un mismo servidor, es necesaria la creación de claves dinámicas. En esta práctica sólo se utilizarán las claves dinámicas.

1.5 Tunnel SSH

Los túneles tienen como principio colocar una estructura de información dentro de otra. En el caso de tunneling de nivel 3 (el mas usado) consiste colocar la información real de la conexión dentro del paquete de datos de la estructura IP. Normalmente va encriptada y es útil, al igual que las VPN, para conectar diversas oficinas remotas de una organización con sensación de estar en la misma red.

El protocolo/utilidad de conexión remota SSH da la posibilidad de crear túneles de forma sencilla desde línea de comandos, necesiándose para ello un servidor SSH y un cliente SSH.

El proceso de establecimiento de una conexión segura SSH es el siguiente:

1. El cliente y el servidor SSH intercambian información de protocolos soportados (SSHv1 o SSHv2).
2. El servidor SSH inicia un intercambio de clave pública con el cliente para probar su identidad. Cada servidor SSH tiene un par de claves (una pública y una privada) creadas en el proceso de configuración del servidor. Este par identifica al servidor. La primera vez que un cliente se conecta al servidor, el programa SSH pide al usuario aceptar una copia de la clave pública del servidor con un mensaje, al que habrá que responder con un YES o similar.
3. Si se responde YES, SSH copia en el cliente la clave pública del servidor. El cliente usa esa clave pública para autenticarse en el servidor SSH para conexiones sucesivas.
4. Si durante un intento de conexión posterior, el cliente SSH detecta que la clave pública del servidor SSH es diferente a la que tiene almacenada, aparecerá un mensaje de aviso e incluso podrá impedir la conexión. El cliente interpreta que el servidor al que se quiere conectar no es el mismo al que se está conectando. Si la causa de esta diferencia de claves es conocida, será necesario eliminar la clave pública del servidor previamente almacenada.
5. El cliente y servidor SSH negocian parámetros de sesión mediante el intercambio de información de parámetros, incluyendo los métodos de autenticación y compresión de datos.

6. Ambos, cliente y servidor, crean una clave compartida simétrica, (en particular la crea el cliente, que es el que puede enviar información encriptada) y el cliente se la envía al servidor. Una vez ambos conocen la clave compartida, pueden establecer una conexión bidireccional segura simétrica. Esta es una clave de sesión, que expira una vez que expira la sesión.
7. Una vez establecida la conexión segura, el cliente se autentica en el servidor usando esa conexión. El servidor comprueba la identidad y la clave de usuario.
8. El intercambio de datos puede mantenerse a lo largo de toda la sesión, hasta que el usuario termina la sesión o pasa un tiempo sin transmisión de información.

Existen dos posibilidades fundamentales de configuración de túneles de redireccionado con ssh, el redireccionado local y el redireccionado remoto.

1.5.1 Redireccionado local

En la figura 1 se presenta un esquema de ejemplo de tunnel ssh con redireccionado local.

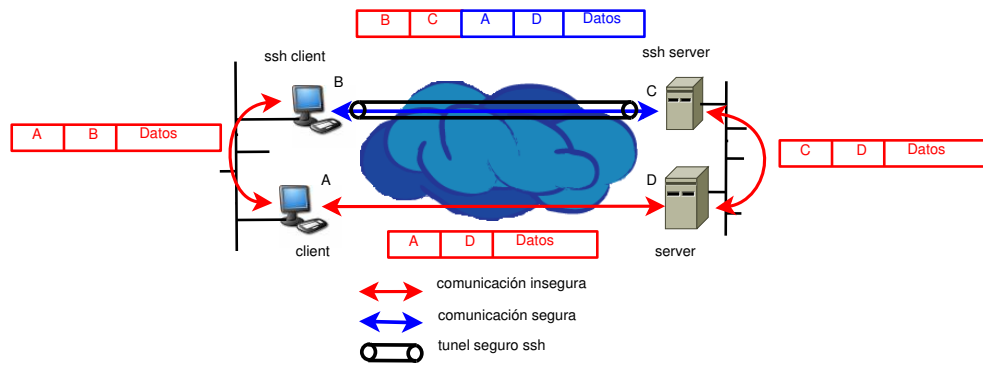


Figure 1: Topología tunnel SSH redireccionado local

El demonio SSH utilizado es el OpenSSH, y el comando de creación del tunnel sería:

```
hostB$ ssh -f -N -L hostB:portB:hostD(visto desde C):portD(visto desde C) hostC
```

donde `-f` implica que la consola ssh permanezca oculta, `-N` implica que no se quieren ejecutar comandos remotos y `-L` es el comando propio de redirección.

Se trata de que un hostA que se tiene que conectar de forma insegura (protocolo inseguro) con un hostD, pueda hacerlo a través de un tunnel creado de forma segura entre hostB y hostC. El host A y el host B pueden ser el mismo. Los paquetes que van entre hostA y hostD quedan encapsulados y encriptados como datos en cabeceras que van entre hostB y hostC. La aplicación ssh abre un tunnel encriptado entre el hostA y el hostD, a través del hostB y el hostC.

El hostA se comunica con el hostD de la siguiente forma:

1. B establece una conexión ssh con C, y de paso abre un tunnel entre A y D.
2. A se comunica con B en el puerto puertoB.
3. B encapsula los paquetes de A a B que le llegan por ese puerto, como paquetes que van de A a D al puerto puertoD, y los envía a C.
4. C recibe los paquetes de B, los desencapsula y los redirige como paquetes que van de C al puertoD de D.
5. D retorna paquetes a C, como en cualquier conexión.
6. C interpreta los paquetes de D como que van destinados a A y los encapsula en paquetes que van de C a B.
7. B desencapsula los paquetes que le llegan y los redirige a A.

Un ejemplo es una conexión web al puerto inseguro 80. En el esquema de la figura, se establece una conexión ssh entre B y C, abriéndose un tunel entre los puertos de B 10080 y el puerto 80 de D. El comando ssh sería:

```
hostB$ ssh -f -N -L 10080:IP_D(vista desde C):80 IP_C
```

El hostA se conecta al puerto “localport” en la IP_B. De esa forma, en el navegador de A se pone como destino a B:10080, y se accedería a la página de forma segura.

1.5.2 Redireccionado remoto

Sirve para utilizar un servidor puente para conexiones entre equipos en redes privadas.

Suponiendo que se dispone de dos equipos (HostA y HostB) en redes privadas diferentes y que no se puede modificar la pasarela de conexión a Internet en ambas redes por parte de los usuarios. Uno de ellos (HostA) es un servidor y el HostB quiere conectarse a él, pero no hay redirección de puertos en la pasarela de HostA. Suponemos que tenemos un servidor `ssh` en Internet (HostC) al cual puede acceder el servidor HostA (con nombre de usuario y contraseña); entonces es posible abrir un puerto en el servidor HostC que se redirecciona al puerto donde escucha el servidor HostA (ver figura 2) de la forma:

```
HostA$ ssh -f -N -R PortA:HostA:PortC HostC
```

En el servidor SSH HostC (que hace de repetidor) tiene que estar habilitado el forwarding remoto en el fichero `/etc/ssh/sshd_config` con la clausula “GatewayPorts yes”

```
.....
GatewayPorts yes
.....
```

Cuando el HostB establece una conexión al puerto PortC del servidor HostC, este último redirecciona al PortA del servidor HostA

Es útil para conexiones cliente-servidor donde ambos están en redes locales diferentes tras gateways NAT.

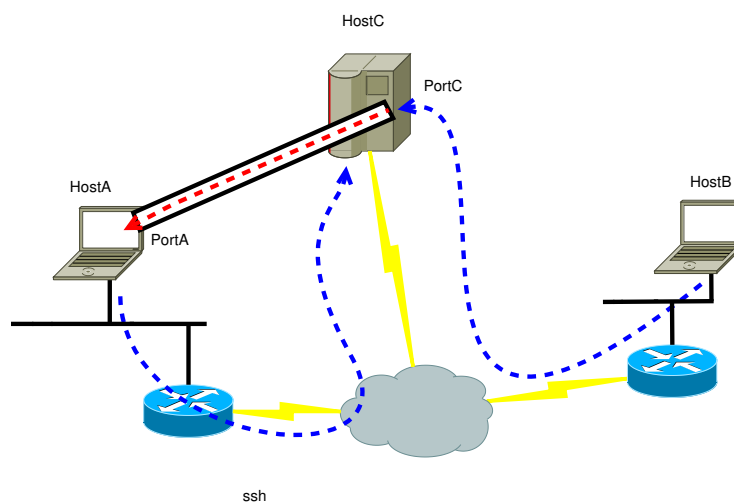


Figure 2: Topología tunel SSH redireccionado remoto

1.5.3 Redireccionado dinámico de puertos

Este redireccionado llamado *SSH Dynamic Port Forwarding* es una forma de abrir un servidor proxy con SOCKS 4/5 en el equipo local y redirigir todo el tráfico que se conecta al puerto abierto localmente hacia el servidor SSH donde quiera que esté.

Es útil para permitir conexiones a Internet (habitualmente tráfico web, aunque puede ser cualquier tipo de tráfico habilitado para utilizar un proxy SOCKS 4/5) a equipos que no disponen de la misma. Si

HostC tiene acceso a Internet, HostA tiene acceso a HostC y un HostB opcional tiene acceso a HostA, la sintaxis sería:

```
HostA$ ssh -f -N -D IP_A:PortA HostC
```

De esta forma, poniendo por ejemplo en un HostB (o HostA) en las preferencias del navegador web como proxy SOCKS al equipo con IP_A y puerto PortA, el navegador se conectaría a Internet a través del HostC.

1.6 Otras alternativas

Al margen de todas las tecnologías anteriores es posible construir una VPN combinando adecuadamente las herramientas SOCAT (opción TUN) y SSH, y realizando la configuración IP apropiada. También se ha contrastado la posibilidad de crear VPNs mediante una combinación adecuada de SSH y pppd (<http://tldp.org/HOWTO/ppp-ssh/index.html>).

Existen alternativas comerciales para la creación de VPNs, de las cuales destaca Hamachi (<https://secure.logmein.com/p>) o similares.

2 Descripción

Se pretende crear diferentes topologías habituales en situaciones reales de conectividad entre dos entidades remotas de forma segura a través de una red insegura como Internet.

- Conexión de un cliente remoto con una red corporativa. Esta topología es habitual para trabajadores remotos que quieren utilizar los servicios y la infraestructura de la red de su corporación, y lo quieren hacer de forma segura y transparente. La situación mas habitual es de un equipo doméstico que se conecta a Internet a través de una pasarela con servicio NAT. La red corporativa dispone de un cortafuegos que separa la red externa de la interna, y en la red interna dispone de los servicios corporativos comunes, como servidor de ficheros, servidor de bases de datos, servidor de aplicaciones corporativas y los equipos terminales y periféricos de los trabajadores.
- Conexión de una red de oficina remota a la oficina central de la organización. Esta topología es análoga a la anterior, con la diferencia de que en lugar de que el cliente VPN esté ubicado en cada equipo, este está instalado y configurado de forma permanente en una pasarela de salida de la oficina remota, que puede hacer a su vez de firewall y router.

Las configuraciones VPN pueden tener a su vez múltiples variantes en cada topología y combinaciones de ellas, entre las que se destacan:

- Configuración de VPN enrutada o VPN puenteada. Una VPN enrutada, mantiene el direccionamiento original entre las redes de cliente y de servidor, de forma que tanto el servidor VPN como el cliente deben enrutar correctamente ambas redes. Una VPN puenteada integra las redes de cliente y servidor en una sola red lógica.
- Configuración del servidor VPN para permitir acceso a múltiples clientes, ya sean estos hosts o redes.
- Configuración de la conexión VPN con claves estáticas o dinámicas.
- Interconexión de clientes a través del servidor VPN.

En esta práctica sólo se verán algunas de ellas, en particular en el modo enrutado. Para conocer e implementar variantes mas específicas, se remite al lector a consultar la excelente documentación de la página principal *OpenVPN* “<https://openvpn.net/community-resources/how-to/>”.

Como infraestructura se dispone de dos equipos PC y de dos pasarelas Mikrotic 2011 llamadas de forma genérica RB (Router Boards).

Las pasarelas RB tienen instalada la utilidad “iproute2” para facilitar la gestión de las interfaces, aunque OpenWRT dispone de un sistema de configuración de red preinstalado llamado LUCI, bastante sencillo y potente. En principio no será necesario utilizarlo.

2.1 Situación de partida

Dadas las características de compartición del laboratorio y por tanto de sus equipos, la configuración inicial de los equipos puede variar, por lo que habrá que uniformarla en la medida de lo posible.

- Situación de partida de los PC's
Los PCs deberán arrancar con un direccionamiento de red acorde a la red plana del laboratorio (192.168.29.0/24) y con salida a Internet a través de F0 (192.168.29.1). Por ejemplo: el PC13 arrancará con la dirección IP 192.168.29.113/24 y tendrá como puerta de enlace (Gateway) a F0.
- Situación de partida de las pasarelas (RBM)
Los RouterBoardsMikrotik (RBM) arrancan con la dirección ip 192.168.29.A siendo A el número de RBM escrito en su carcasa, por ejemplo, el RBM21 arranca con la dirección IP 192.168.29.21. Esta dirección es accesible en la interfaz “br-lan” de la figura 3, es decir, el PC gestor de cada RBM deberá conectarse con el latiguillo con conectores RJ45 a cualquier puerto de la interfaz “lan”.

Para que funcionen como routerboards, será necesario acceder a ellos y ejecutar el comando:

Listing 1: Conversión en router.

```
root@RBM:~# ./isa_vpn.sh
```

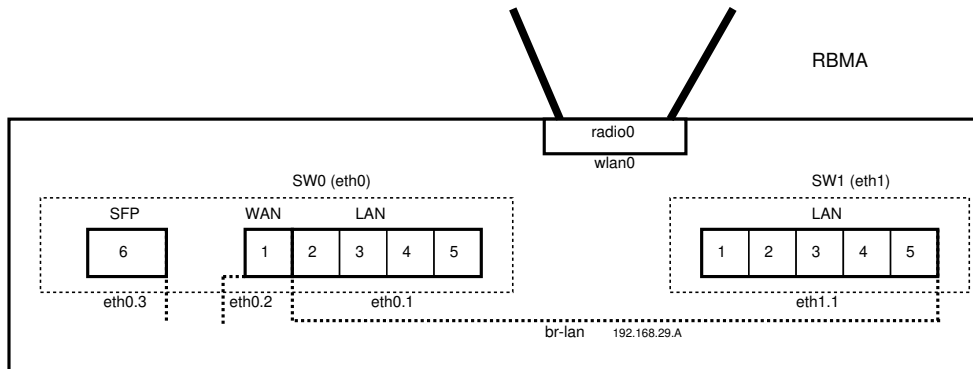


Figure 3: Situación de partida

Table 1: Situación de partida

Interfaz	Dirección IP
eth0.2	172.16.A.100/24
br-lan	192.168.A.1/24
login	passwd
root	provisional

Desde ese momento, la configuración de las interfaces cambia, así como el direccionamiento IP, según se muestra en la figura 4 y en la tabla 1. Los RouterBoardMicrotik pasarán a convertirse en pasarelas y tomarán el nombre de RBA, siendo A el mismo número asociado al RBMA anterior.

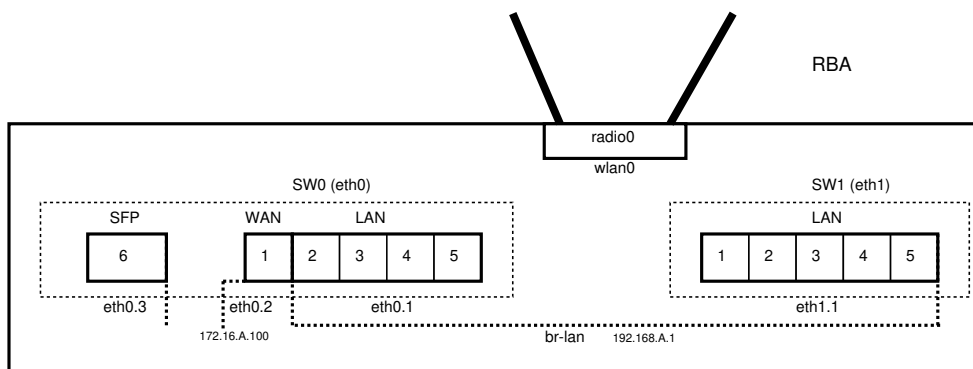


Figure 4: Situación de partida como pasarela.

El sistema operativo OpenWRT incluye además de los demonios de enrutamiento y las utilidades básicas de los sistemas Linux, otras utilidades mas complejas, como lo son el firewall “shorewall” (deshabilitado en esta práctica), el demonio “openvpn”, la weblet para visualizar el funcionamiento del router a través de un navegador y el demonio “sshd” para permitir conexiones remotas seguras y poder prescindir en los equipos con OpenWRT de tarjeta gráfica y monitor. El RBMA dispone de un puerto serie con interfaz RJ45 para conectividad directa.

- Tabla con información de la situación de partida de RBMA (atención a las interfaces y sus direcciones IP)
- Incluye servidor shell seguro (dropbear) configurado para acceder desde cualquier direccion.
- Incluye paquete openvpn.

2.2 Instalación

NOTA: los paquetes necesarios ya están instalados en las diferentes RB.

La práctica VPN se realiza utilizando el paquete “*openvpn-openssl*” disponible para prácticamente todas las distribuciones Linux. Los routers que implementarán el demonio *openvpn* son routerboards Mikrotik

RB2011 con sistema operativo OpenWRT en la versión 19.07. También se implementará el cliente *openvpn* en los equipos cliente linux de la práctica (PC-X).

NOTA: El paquete “openvpn-openssl” ya está instalado en los equipos del laboratorio. Se muestra el proceso de instalación con fines de registro de procedimiento. La instalación en los routers se hace con el procedimiento habitual utilizando el comando “opkg”. Este proceso se hace desde repositorio si se dispone de conexión a Internet en el RBM.

Listing 2: Instalación openvpn en openwrt.

```
root@RBMA:~# opkg update
root@RBMA:~# opkg install openvpn-openssl
```

NOTA: los paquetes necesarios ya están instalados en las diferentes RB.

2.3 Preparación inicial

Se necesitará utilizar los comandos específicos:

```
#ip addr [help]
#ip route [help]
```

Cuyo uso mas común tendrá como ejemplo:

```
# ip addr add <direccion/mask> dev <ethx>
ejemplo# ip addr add 192.168.12.112/24 dev eth1

# ip route add <red destino/mascara> via <gateway> dev <interfaz de salida>
ejemplo# ip route add 192.168.12.0 via 10.10.12.1 dev eth0.1
```

Con los comandos anteriores será necesario crear la topología de la figura XXX. En la figura 5, los segmentos X están puestos en el orden A<B<C<D<...X

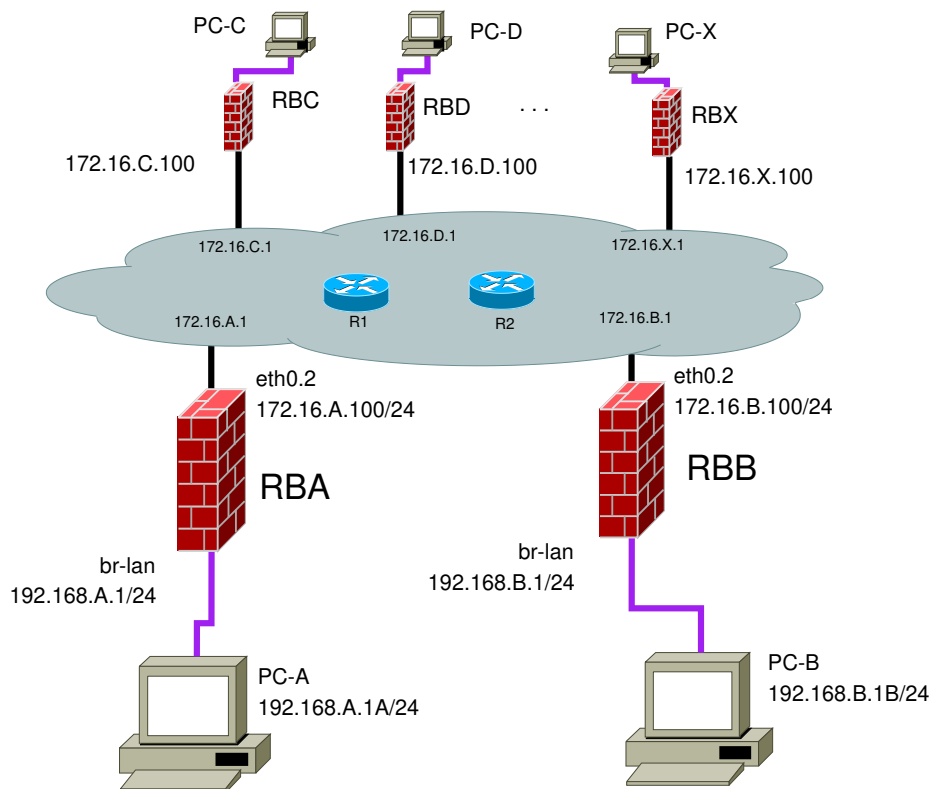
Será necesario configurar los equipos de usuario con la dirección IP correspondiente a cada segmento para tener conectividad tanto con la pasarela de su segmento como eventualmente con otros equipos y otros segmentos.

Los routers R1 y R2 del laboratorio proveen de conectividad entre las diferentes pasarelas RBX dentro de la “nube” de la figura 5.

Los routers *R1* y *R2* de la figura 5 trabajan como un routers estáticos y representan a la red Internet o a cualquier red intermedia entre los clientes y la red central y no tendrá repercusión en el desarrollo de la práctica, considerándose estos ya correctamente configurados con las direcciones IP que serán las puertas por defecto de los RBs y las rutas necesarias para su interconexión. El direccionamiento utilizado para la red intermedia es un direccionamiento privado por cuestiones de compatibilidad, y los routers *R1* y *R2* no deben conocer las rutas a las redes de las LAN privadas. Por ello, se han representado dentro de una nube.

Las redes lógicas IP de los clientes tienen que ser diferentes entre sí para que la pasarela servidora pueda enrutarse hacia ellas correctamente. Será necesario implementar la configuración IP según cada caso estudiado, teniendo en cuenta la configuración inicial presentada en la sección 2.1.

Todos los equipos locales de las LAN tienen como pasarela por defecto a los equipo pasarela cliente o servidor VPN de sus respectivas LAN. Si no fuese así, será necesario indicar a los equipos locales la ruta hacia la red VPN (en el caso de la práctica, la red 10.20.30.0/24, como se verá mas adelante).



3 VPN enrutada

3.1 Creación de las claves dinámicas

OpenVPN utiliza un modelo PFS (Perfect Forward Secrecy) aumentado, que básicamente consiste en:

- Uso del modelo RSA de claves asimétricas para autenticar los extremos de la comunicación, y
- uso del modelo Diffie-Hellman para la generación de claves simétricas en los extremos.
- A mayores, se usará una clave simétrica que se distribuirá entre todos los clientes y el servidor y servirá para encriptar todas las comunicaciones anteriores.

El modelo RSA implica la creación de una autoridad certificadora (CA) confiable, que se encargará de la creación de los pares de claves de cada sede y la firma de las claves públicas. Las claves públicas firmadas se distribuyen entre todas las sedes al igual que la clave pública de la autoridad certificadora (ca.crt). El proceso de autenticación implica el envío de un mensaje encriptado con la clave privada; si ese mensaje puede ser descifrado con la clave pública (firmada por la CA) en el destino, el origen es auténtico.

El modelo Diffie-Hellman (DH) consiste en la generación de una clave única idéntica (y secreta) en los dos extremos, intercambiando sólo información pública previa. Ello resuelve el problema de la distribución de claves y facilita el proceso de encriptación. Lo que se distribuye entre todos los componentes de la comunicación son unos parámetros para la generación de la información pública. El proceso Diffie-Hellman tiene una vulnerabilidad ante un ataque Man in the Middle (MitM); por ello, en las VPN se combina con el model RSA para la autenticación de los extremos.

El proceso de creación de claves dinámicas tipo SSL/TLS implica la creación de una Autoridad de Certificación o CA. *Openvpn* incluye el software y los scripts necesarios para ello, aunque en la versión paquetizada para OpenWRT viene en un paquete independiente llamado “*openvpn-easy-rsa*”. Lo cierto es

que las claves pueden crearse (al igual que en el caso de claves estáticas) en un equipo independiente y aislado, para después llevarse las generadas tanto al servidor como a los clientes. En la topología de esta práctica se propondría crearlas en un equipo ubicado en la LAN de la organización central, por ejemplo en el equipo PCLanS, que en la figura 7 se corresponde con PC-A.

Por tanto, partiendo de que PCLanS será la CA donde esté el “certificado raíz” para poder generar certificados (ca.key, ca.crt).

El proceso de creación de claves SSL/TLS requiere diversos pasos:

1. Crear un certificado-CA con el cual se firmarán y se revocarán certificados de clientes.
2. Crear un certificado y una clave pública distintas para cada uno de los clientes (VPNc1X.crt, VPNc1X.key) y para el servidor VPN (VPNsv.crt, VPNsv.key). Proceso RSA.
3. Crear una clave simétrica (ta.key) llamada TLS-AUTH.
4. Crear unos parámetros para la generación de claves simétricas Diffie-Hellman (dh2018.pem).
5. Distribuir las claves, certificados y parámetros DH a los clientes y servidor VPN

1. Crear un certificado-CA con el cual se firmarán y se revocarán certificados de clientes.

Se hará en el equipo PCLanS. Será necesario tener instalado el paquete “*easy-rsa*” y disponer de un directorio seguro donde estén enlazados los scripts y las utilidades. En este caso se supone en “/home/alumno/PKI/easy-rsa”.

```
PCLanS~ ./easyrsa build-ca
.....
```

Si pide un nombre común CN, se recomienda poner el nombre del equipo donde va, para mantener organizados los certificados y claves; en este caso sería “PCLanS” o cualquier otro nombre siempre que no se vuelva a utilizar ese CN. Esto habrá creado los ficheros “ca.crt” y ca.key“.

Básicamente, se selecciona un tamaño de clave de 1024 bits (en algunos casos extremos se puede recomendar una clave de 2048 bits), con una validez del certificado CA y las claves de 10 años (este dato es importante si las fechas del equipo donde se generan las claves y los equipos cliente y servidor no están sincronizados y tienen diferencias superiores a los 10 años), y los datos de ubicación y contacto (si no se quieren especificar estos datos de ubicación será necesario poner la palabra NA en cada uno de ellos, no se pueden dejar vacíos).

2. Crear un certificado y una clave pública distintas para cada uno de los clientes y para el servidor VPN.

Desde el equipo que hace de CA (PCLanS), se creará primero la clave del que será el servidor VPN (la pasarela VPNs1):

```
PCLanS~/PKI/easy-rsa$ ./easyrsa build-server-full VPNs1 nopass
```

como segundo parámetro del comando anterior se ha puesto VPNs1 para identificar al equipo servidor, pero podía haberse cualquier nombre identificativo que el administrador considere. A este nombre se le conoce como Nombre Común.

Este comando es interactivo, y se deberá responder afirmativamente a las preguntas que plantee.

Tras ello se crearán las claves para cada uno de los clientes:

```
PCLanS~/PKI/easy-rsa$ ./easyrsa build-client-full VPNc1 nopass
PCLanS~/PKI/easy-rsa$ ./easyrsa build-client-full VPNc2 nopass
```

Si se quiere proveer de una contraseña a alguna de las claves, se podrá utilizar el comando “build-key-pass”. Si una clave se provee con contraseña, el cliente que se conecte deberá introducirla cada vez que se conecte al tunel.

Se utilizan los parámetros de Diffie-Hellman:

```
PCLanS ~/PKI/easy-rsa$ ./easyrsa gen-dh
```

El comando anterior genera el fichero “/home/alumno/PKI/easy-rsa/pki/dh.pem” que contiene los parámetros DH.

Finalmente es necesario crear una clave TLS-AUTH. Esta clave deberá estar en el servidor y los clientes.

```
PCLanS ~/PKI/easy-rsa$ openvpn --genkey --secret ta.key
```

3. Distribuir las claves y certificados a los clientes y servidor VPN

La distribución de las claves y certificados se realizará por un canal seguro. Se recomienda que en los equipos pasarela, los certificados y claves se guarden en “/etc/openvpn/keys” para facilitar la organización.

Las claves y certificados generados en PCLanS se reparten según la tabla 2. En esta tabla sólo se indica la existencia de un servidor (VPNsv1) en RBA y un cliente (VPNcl1) en RBB. No obstante, si se añadiesen mas clientes habría que distribuir las claves en cada uno de ellos de forma similar a VPNcl1.

Archivo	Descripción	Destino	Secreto
dh1024.pem	Parámetros Diffie-Hellman	Servidor (VPNsv1)	No
ca.crt	Certificado raíz CA	Servidor y todos los clientes	No
ca.key	Clave raíz CA	Equipo creador de las claves (PCLanS)	Si
ta.key	Clave TLS-AUTH	Servidor y todos los clientes	Si
VPNsv1.crt	Certificado servidor	Servidor (VPNsv1)	No
VPNsv1.key	Clave del servidor	Servidor (VPNsv1)	Si
VPNcl1.crt	Certificado cliente LAN remota	Pasarela LAN remota (VPNcl3)	No
VPNcl1.key	Clave cliente LAN remota	Pasarela LAN remota (VPNcl3)	Si

Table 2: Distribución de claves SSL/TLS.

3.2 Servidor VPN en oficina central y cliente VPN en sucursal

Se implementará inicialmente una topología en la que hay una oficina central (Red A) en la que está el servidor VPN y una delegación (Red B) en la que está el cliente VPN. Tanto el servidor como el cliente VPN están instalados en los gateways (RouterBoards RBX) de ambas sedes. La figura 6 representa esta topología.

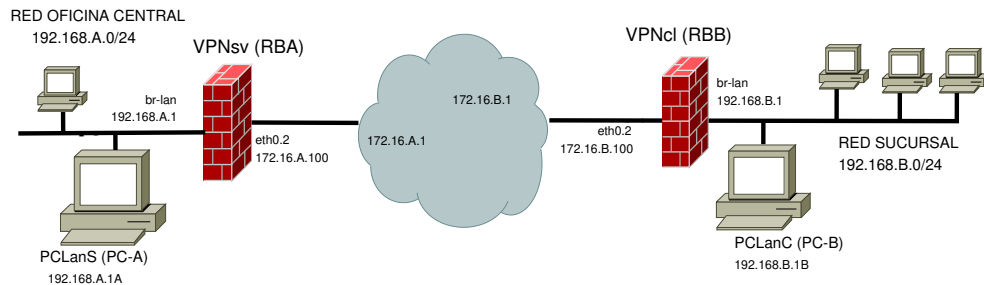


Figure 6: Topología VPN con una LAN como cliente

La topología sobre la que se realiza tiene una oficina central con una pasarela RBA llamada VPNsv que serviría de servidor VPN y una oficina remota con pasarelas RBB llamada VPNcl.

Los ficheros de claves ya distribuidas se encuentran en el directorio del RouterBoard RBX “/etc/openvpn/easy-rsa/keys” y se recomienda no modificar su contenido. En el directorio “/etc/openvpn/easy-rsa/keys” están los ficheros de la tabla 3

Archivo	Descripción
dh1024.pem	Parámetros Diffie-Hellman
ca.crt	Certificado raíz CA
ta.key	Clave TLS-AUTH
RBXs.crt	Certificado servidor. Necesario cuando RBX actúa como servidor.
RBXs.key	Clave del servidor. Necesaria cuando RBX actúa como servidor.
RBXc.crt	Certificado cliente LAN remota. Necesaria cuando RBX actúa como cliente.
RBXc.key	Clave cliente LAN remota. Necesaria cuando RBX actúa como cliente.

Table 3: Claves SSL/TLS en el RBX

3.2.1 Prueba desde línea de comandos

En el servidor (VPN_{s1}) RBA:

```
RBA# openvpn --dev tun1 --ifconfig 10.20.30.1 10.20.30.2 --tls-server --dh /etc/openvpn/easy-rsa/keys/dh2048.pem --ca /etc/openvpn/easy-rsa/keys/ca.crt --cert /etc/openvpn/easy-rsa/keys/RBAs.crt --key /etc/openvpn/easy-rsa/keys/RBAs.key --reneg-sec 60 --verb 5
```

En un cliente (VPN_{c1}) RBB:

```
RBB# openvpn --remote 172.16.A.100 --dev tun1 --ifconfig 10.20.30.2 10.20.30.1 --tls-client --ca /etc/openvpn/easy-rsa/keys/ca.crt --cert /etc/openvpn/easy-rsa/keys/RBBc.crt --key /etc/openvpn/easy-rsa/keys/RBBc.key --reneg-sec 60 --verb 5
```

3.2.2 Configuración de servidor y clientes

- Configuración del servidor RBA (VPN_{s1}) (archivo “/etc/openvpn/RBAs.conf”):

```
dev tun
proto udp
local 172.16.A.100
server 10.20.30.0 255.255.255.0
route 192.168.B.0 255.255.255.0
route 192.168.C.0 255.255.255.0
route 192.168.D.0 255.255.255.0
#push "route-gateway 10.0.0.1"
push "route 192.168.A.0 255.255.255.0"
keepalive 10 60
log-append /var/log/openvpn.log
status /var/log/openvpn-status.log
max-clients 25
client-config-dir ccd
client-to-client

ca /etc/openvpn/easy-rsa/keys/ca.crt
cert /etc/openvpn/easy-rsa/keys/RBAs.crt
key /etc/openvpn/easy-rsa/keys/RBAs.key
dh /etc/openvpn/easy-rsa/keys/dh1024.pem
tls-auth /etc/openvpn/easy-rsa/keys/ta.key 0
ifconfig-pool-persist /etc/openvpn/ipp.txt
#daemon
verb 9
.....
```

Será necesario crear el directorio “/etc/openvpn/ccd” y almacenar en él un fichero por cada LAN cliente con el nombre común de la clave de cliente y con el contenido siguiente “route 192.168.B.0 255.255.255.0” para la LAN cliente B e “route 192.168.C.0 255.255.255.0” para la LAN cliente C. Si hubiera mas LANs cliente habría que crear otro fichero por cada una de ellas.

```
VPNsv# echo "route 192.168.B.0 255.255.255.0" > /etc/openvpn/ccd/RBBc
VPNsv# echo "route 192.168.C.0 255.255.255.0" > /etc/openvpn/ccd/RBBc
```

Esto permite indicar al servidor VPN que hay unas redes detrás de las conexiones certificadas.

- Configuración del cliente RBB (VPN_{c1}) (archivo “/etc/openvpn/RBBc.conf”): (será similar para el resto de las redes cliente)

```

client
pull
remote 172.16.A.100
dev tun
keepalive 10 60

log-append /var/log/openvpn.log
ca /etc/openvpn/easy-rsa/keys/ca.crt
cert /etc/openvpn/easy-rsa/keys/VPNclB.crt
key /etc/openvpn/easy-rsa/keys/VPNclB.key
tls-auth /etc/openvpn/easy-rsa/keys/ta.key 1
ns-cert-type server
#daemon
verb 9
.....

```

Explicación de algunas directivas:

log-append : fichero donde se almacenan las salidas de error y log. Si se da esta directiva en el fichero de configuración, no aparecerá nada por pantalla.

verb : nivel de “verbosity” del log. Cuanto mas alto es este valor, mas información se almacenará. Conviene tener un valor alto cuando hay necesidad de depuración de errores, y un valor bajo cuando la VPN está en operación, para evitar saturación de ficheros y discos.

client-to-client : permite la comunicación entre clientes.

client-config-dir : indica que los archivos de configuración que se aplicaran a los clientes estan en directorio ccd.

pull : cuando en el servidor está la directiva “push”, en los clientes debe estar la directiva “pull”, que implica que trae la información de “push”.

4 Topología VPN con un servidor y varios clientes

Se propone la creación de una topología con varios clientes VPN como en la figura 7 donde el alumnado se ponga de acuerdo para la decisión de quién hace de servidor y quienes hacen de clientes.

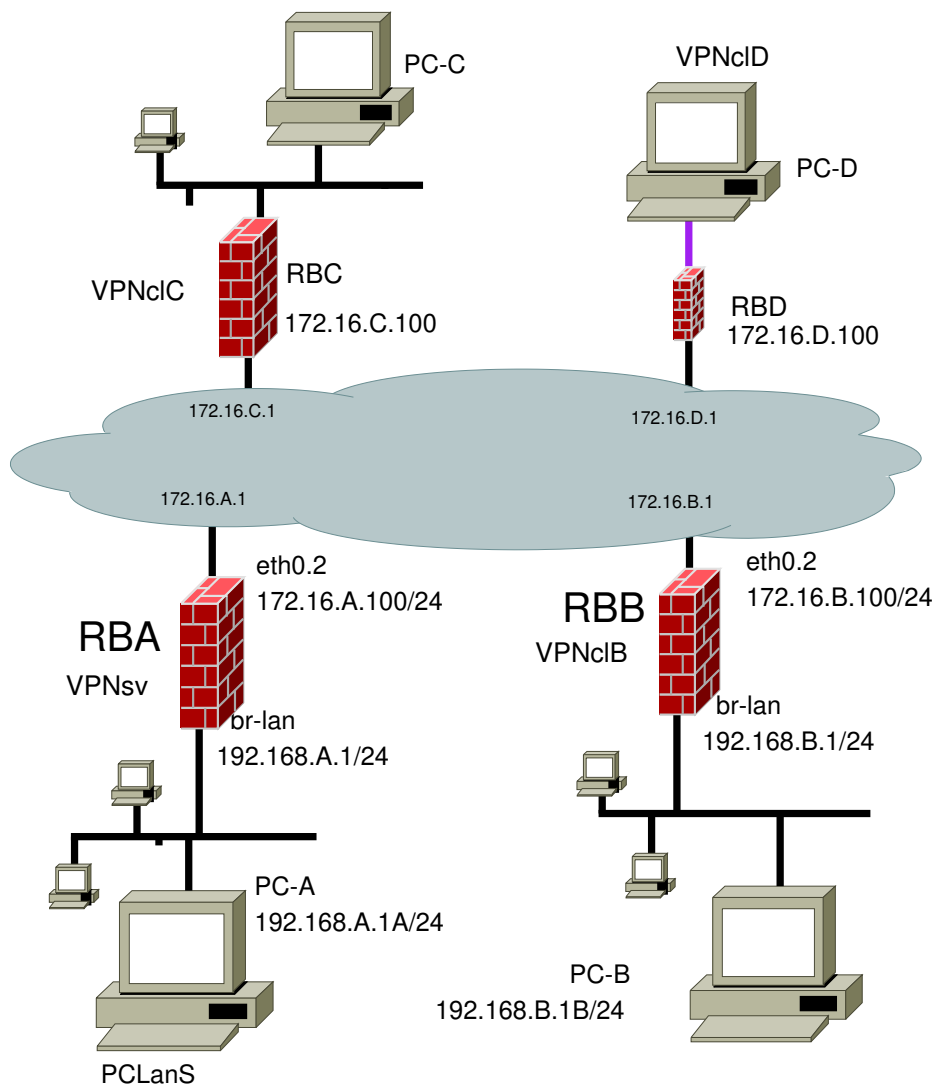


Figure 7: Topología VPN con un servidor y varios clientes

5 Troubleshooting

- La directiva “daemon” en el servidor, hace que no funcione la VPN, aparentemente porque no lee o no aplica el fichero ccd/RBBc.

Solución: no poner la directiva “daemon” en el fichero de configuración y si se quiere utilizar como demonio, utilizar el símbolo &.

- Cuando la diferencia entre la fecha de creación de las claves y la fecha donde se leen es muy alta, openvpn no funciona.

Solución: cambiar la fecha en la pasarela VPN. Si esta es OpenWRT y se quiere poner la fecha del 17 de Agosto de 2024 con hora las 16:00, se hace con el comando:

```
VPN1# date -s 202408171600
```


6 Líneas Futuras

Se propone como continuación de la práctica, la posibilidad de que los clientes VPN accedan a Internet a través de la red de la oficina central. Para ello el alumno deberá tener clara la topología deseada y deberá consultar la documentación de la página principal de OpenVPN referida a esta situación (<http://openvpn.net/index.php/openvpn/documentation/howto.html#redirect>).