

# VPN – VIRTUAL PRIVATE NETWORK

## REDES PRIVADAS VIRTUALES

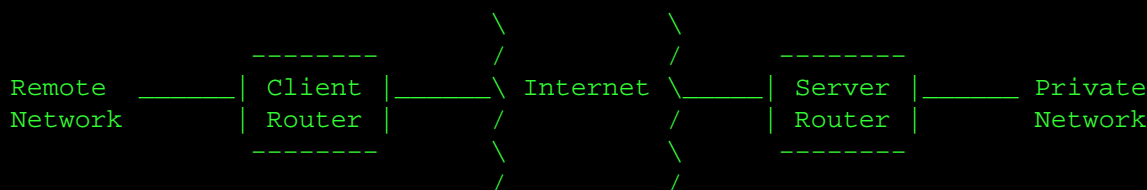
### 1. QUE ES UNA VPN?

El término VPN se refiere a una Red Privada Virtual (Virtual Private Network), la cual utiliza Internet como mecanismo de transporte, manteniendo la seguridad de los datos en la VPN.

La configuración más común de VPN es a través de una red interna principal y nodos remotos usando VPN para lograr el acceso completo a la red central. Los nodos remotos son comunmente oficinas remotas o empleados trabajando en casa. También pueden vincularse dos redes más pequeñas para conformar una red simple, incluso más grande.

### 2. COMO TRABAJA?

Para construir una VPN se crea un túnel seguro entre las dos redes y se realiza enrutamiento IP a través de él.



```

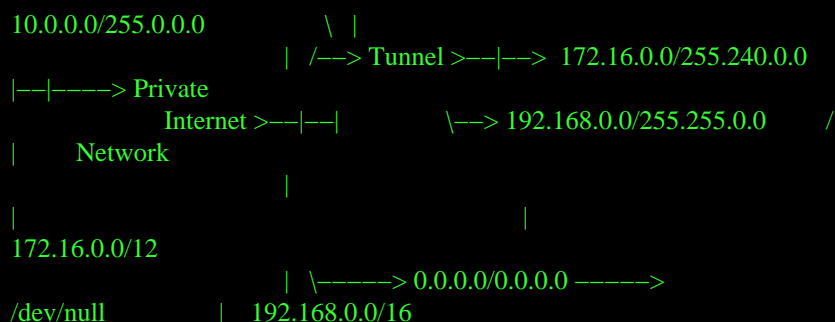
Client
Router
-----
| /-> 10.0.0.0/255.0.0.0
\
  Remote | |-> 172.16.0.0/255.240.0.0 |-> Tunnel >---\ |
  Network >---|---> 192.168.0.0/255.255.0.0 /
|---|----> Internet
      192.168.12.0 |
|
| |
| \-----> 0.0.0.0/0.0.0.0 --> IP Masquerade >--- / |
-----
  
```

Server Router

```

-----
| /->
  
```

vpn



El diagrama anterior muestra como debería configurarse la red. El enrutador cliente es una caja Linux que actúa como gateway o firewall para la red remota. Como se observa, la red remota usa la red local 192.168.12.0. Para un diagrama simple, se libera la información de enrutamiento local en los enrutadores. La idea básica es enrutar el tráfico a todas las redes privadas (10.0.0.0, 172.16.0.0 y 192.168.0.0) a través del túnel. La configuración que se muestra aquí es sólo una opción. Esto hace que aunque la red remota pueda ver la red privada, la red privada no necesariamente puede ver la red remota. Con el fin de que esto ocurra se debe especificar que las rutas sean bidireccionales.

En el diagrama también se puede notar que todo el tráfico emergente del enrutador cliente sale de él, es decir, todo desde una dirección IP. Se podrían enrutar números reales desde el interior de la red, pero eso conlleva a toda clase de problemas de seguridad.

## 2.1 SSH y PPP

El sistema descrito para implementar VPN usa SSH y PPP. Básicamente se emplea SSH para crear un túnel de conexión y después usa PPPD para correr tráfico TCP/IP a través de él (esto significa que enmascara el túnel). Esto le permite a PPPD comunicarse a través del SSH como si fuera una línea serial. En el lado del servidor, PPPD corre como el shell de los usuarios en la sesión SSH, completando el enlace. Después de esto, todo lo que se necesita hacer es el enrutamiento.

### Sistemas VPN alternativos

Otros mecanismos para configurar una VPN:

#### PPTP

PPTP es un protocolo de Microsoft para VPN. Está soportado bajo linux, pero se sabe que tiene serios problemas de seguridad. Su uso está cubierto en Linux VPN Masquerade HOWTO.

#### IP Sec

IP Sec es un conjunto de protocolos diferente de SSH.

## CIPE

CIPE es un sistema de encriptación de red a nivel de kernel que puede ser más apropiado para instalaciones empresariales.

## 2.2 SERVIDOR

### Seguridad

La seguridad es muy importante para una VPN. Se necesita mantener algunas cosas en mente mientras se está configurando el servidor.

### Adaptación de los demonios

Se puede utilizar un servidor web para descargar un par de archivos necesarios para configurar las nuevas máquinas para acceder la VPN. Si realmente se quiere correr diferentes servidores en el gateway, se deben tener en cuenta las restricciones de acceso solo para esas máquinas en la red privada.

No deben permitirse passwords. Aunque suene ilógico, todas las autenticaciones en esta máquina se deben hacer mediante el sistema de autenticación de claves públicas de ssh. Así, sólo con esas claves se puede hacer acceso. Además es mucho más difícil recordar una clave binaria de 530 caracteres de longitud.

### Cómo se hace?

Se debe editar el archivo */etc/passwd*. El segundo campo contiene el password o alternativamente 'x', indicando que el sistema de autenticación se encuentra en el archivo */etc/shadow*. Lo que se hace es cambiar ese campo con '\*'. Esto significa que el sistema de autenticación que hay no es con password y que no se debe permitir ninguno. El archivo */etc/passwd* debería verse así:

```
...
nobody:x:65534:100:nobody:/dev/null:
mwilson:x:1000:100:Matthew Wilson,,,:/home/mwilson:/bin/bash
joe*:504:101:Joe Mode (home),,,:/home/vpn-users:/usr/sbin/pppd
bill*:504:101:Bill Smith (home),,,:/home/vpn-users:/usr/sbin/pppd
frank*:504:101:Frank Jones
(home),,,:/home/vpn-users:/usr/sbin/pppd
...
```

### Acceso de usuarios

El acceso de los usuarios se hace a través del esquema de autenticación de ssh que indica cómo pueden acceder los usuarios al sistema mientras se mantiene un alto nivel de seguridad.

### Configuración de sshd

Se deben configurar las siguientes opciones. La idea es deshabilitar la autenticación de password y la autenticación de rhosts. Las siguientes opciones se deben establecer en el archivo */etc/sshd\_config*:

```
PermitRootLogin yes
IgnoreRhosts yes
StrictModes yes
QuietMode no
CheckMail no
IdleTimeout 3d
X11Forwarding no
PrintMotd no
KeepAlive yes
RhostsAuthentication no
RhostsRSAAuthentication no
RSAAuthentication yes
PasswordAuthentication no
PermitEmptyPasswords no
UseLogin no
```

### Restricción de usuarios

Además de permitir a algunos usuarios acceder al sistema, también hay que asegurarse de que no accedan recursos privados. Esto se puede hacer fácilmente sólo permitiéndoles correr el pppd.

### Utilización de sudo

Existe un pequeño programa llamado *sudo* que permite la administración de un sistema Unix para conceder a ciertos usuarios la capacidad de correr ciertos programas como *root*. Esto es necesario en este caso ya que pppd debe correr como root. Se necesitará el uso de este método si se quiere permitir que los usuarios accesen al shell.

Si se decide no permitir a los usuarios tener acceso al shell, entonces el mejor camino es lograr que el shell sea el pppd. Esto se hace en el archivo */etc/passwd*. El último campo del archivo */etc/passwd* es el shell de usuario. No hay que hacer nada especial para trabajar con el pppd, pues cuando un usuario se conecta, éste se corre como root. Esta es la configuración más simple, así como la más segura. Es ideal para sistemas corporativos y a gran escala.

### Funcionamiento en red

Hay que asegurarse de que los usuarios no solo tengan acceso al sistema, sino también a la red. Esto puede hacerse utilizando las reglas del uso de firewalls del kernel de Linux y las tablas de enrutamiento. Utilizando la ruta y los comandos ipfwadm se puede configurar el kernel para manejar el tráfico en la red en la forma apropiada.

### Kernel

Con el fin de configurar correctamente el kernel se debe asegurar que las siguientes opciones están activadas en adición al funcionamiento básico de la red.

Ejemplo para kernel 2.0:

```
CONFIG_FIREWALL
CONFIG_IP_FORWARD
CONFIG_IP_FIREWALL
CONFIG_IP_ROUTER
CONFIG_IP_MASQUERADE (optional)
CONFIG_IP_MASQUERADE_ICMP (optional)
CONFIG_PPP
```

Ejemplo para kernel 2.2:

```
CONFIG_FIREWALL
CONFIG_IP_ADVANCED_ROUTER
CONFIG_IP_FIREWALL
CONFIG_IP_ROUTER
CONFIG_IP_MASQUERADE (optional)
CONFIG_IP_MASQUERADE_ICMP (optional)
CONFIG_PPP
```

## Reglas de filtro

Primero, se escriben la reglas de filtro del firewall que permita a los usuarios acceder a las redes internas mientras se les restringe el acceso a Internet. Esto es lógico, pues ellos ya tienen acceso a Internet y si se les permite usar el túnel para acceder la red, se estaría derrochando el ancho de banda y el procesador.

Las reglas de filtro dependen de la red interna que se esté utilizando. Se debe permitir que el tráfico entrante desde las VPNs a la red interna ingrese. Esto se puede hacer utilizando la herramienta ipfwadm en kernel 2.0 o ipchains en kernel 2.2.

Para establecer las reglas con ipfwadm se debe utilizar las siguientes opciones:

```
# /sbin/ipfwadm -F -f
# /sbin/ipfwadm -F -p deny
# /sbin/ipfwadm -F -a accept -S 192.168.13.0/24 -D 172.16.0.0/12
```

Para establecer las reglas con ipchains se debe utilizar las siguientes opciones:

```
# /sbin/ipchains -F forward
# /sbin/ipchains -P forward DENY
# /sbin/ipchains -A forward -j ACCEPT -s 192.168.13.0/24 -d
172.16.0.0/12
```

## Enrutamiento

A continuación hay que indicarle al kernel dónde enviar los paquetes.

Se enruta todo el tráfico destinado para las redes privadas en la interface interna, y los demás tráficos en la interface externa. La especificación de comandos de enrutamiento depende de las redes internas que se están utilizando. Se mostrará un ejemplo con rutas básicas para redes locales y también la incertidumbre de estar utilizando los 3 grupos de números internos.

Si asumimos que la 172.16.254.254 es el gateway interno:

```
# /sbin/route add -net 10.0.0.0 netmask 255.0.0.0 gw 172.16.254.254 dev eth1
# /sbin/route add -net 172.16.0.0 netmask 255.240.0.0 gw 172.16.254.254 dev eth1
# /sbin/route add -net 192.168.0.0 netmask 255.255.0.0 gw 172.16.254.254 dev eth1
```

**Nota:** Si se utilizan dos caminos de enrutamiento por decir, una oficina remota, entonces serán necesarias más cosas. Se necesita configurar las rutas en el servidor que apuntan al cliente. El camino más fácil para lograr esto es correr un proceso *cron* cada minuto que devuelva las rutas. Sin embargo, esto no es conveniente si el cliente no está conectado, la ruta solo arrojará un error.

## 2.3 CLIENTE

Cuando se usa acceso a una red remota, esta caja puede servir fácilmente como un servidor Samba (para redes Windows), un servidor DHCP y hasta un servidor web interno. Es importante recordar que esta caja debería ser tan segura como sea posible, ya que corre en toda la red remota.

### Kernel

Se debe tener un ppp disponible en el kernel. Si se va a permitir que múltiples máquinas utilicen el túnel se necesita tener disponible el *firewalling* y el *forwarding*. Si el cliente va a estar en una sola máquina, es suficiente con ppp.

### Vínculos

El vínculo se crea corriendo pppd a través de una pseudo terminal que se crea para redireccionar pty y conectar a ssh. Esto puede hacerse con los siguientes comandos:

```
# /usr/sbin/pty-redir /usr/bin/ssh -t -e none -o 'Batchmode yes' -c
blowfish -i /root/.ssh/identity.vpn -l joe > /tmp/vpn-device
# sleep 10

# /usr/sbin/pppd `cat /tmp/vpn-device`
# sleep 15

# /sbin/route add -net 172.16.0.0 gw vpn-internal.mycompany.com
netmask 255.240.0.0
# /sbin/route add -net 192.168.0.0 gw vpn-internal.mycompany.com
netmask 255.255.0.0
```

Esto corre ssh, redireccionando su entrada y su salida al pppd. Las opciones pasadas al ssh lo configuran para correr sin caracteres de escape (-e), utilizando el algoritmo de encriptación

blowfish (-c), utilizando la identificación de archivo especificada (-i), en modo terminal (-t) y con las opciones 'Batchmode yes' (-o). Los comandos sleep se usan para espaciar las ejecuciones de los comandos para que cada uno pueda completar su ejecución antes de que otro comando comience.

---

## IMPLEMENTACION DE VPN

### 1. Planeación

Antes de comenzar a configurar el sistema, se deben conocer los detalles de la gestión de redes. Se asume que se tienen dos firewalls protectores, uno por cada firewall de Intranet. Ahora se debe tener (por lo menos) dos interfaces de red por cada firewall. Se deben escribir las direcciones IP y de máscara de red. Se necesitará una dirección IP más por cada firewall para la VPN que se desea configurar. Estas direcciones deben estar por fuera de las subredes existentes. Se sugiere utilizar direcciones del rango de direcciones privadas. Estas son:

- 10.0.0.0 - 10.255.255.255
- 172.16.0.0 - 172.31.255.255
- 192.168.0.0 - 192.168.255.255

Un ejemplo de configuración: los dos bastiones se llaman Fellini y Polanski. Ellos tienen una interfaz para internet (-out), otra para Intranet (-in) y otra para el VPN (-VPN). Las direcciones y máscaras serían:

- fellini-out: 193.6.34.12 255.255.255.0
- fellini-in: 193.6.35.12 255.255.255.0
- fellini-vpn: 192.168.0.1 point to point
- polanski-out: 193.6.36.12 255.255.255.0
- polanski-in: 193.6.37.12 255.255.255.0
- polanski-vpn: 192.168.0.2 point to point

### 2. Herramientas requeridas

Se necesita:

- Linux firewall
- kernel
- Una configuración mínima
- ipfwadm
- fwtk
- Herramientas para el VPN
- ssh
- pppd
- sudo
- pty–redir

### **3. Compilación e instalación**

### **4. Configuración de los otros subsistemas**

Se necesita habilitar el tráfico ssh entre los dos host firewall. Esto implica una conexión al puerto 22 sobre el esclavo desde el maestro. Se debe comenzar con sshd sobre el esclavo y verificar si se puede hacer login.

[Los términos “firewall maestro” y “firewall esclavo” se refiere a los participantes activos y pasivos de la configuración de conexión, aunque un VPN no tiene nada que ver con la arquitectura cliente/servidor. El host que comienza la configuración se referencia como maestro y el participante pasivo como el esclavo.]

### **5. Configuración de la cuenta para el VPN**

Se puede crear una cuenta sobre el firewall esclavo utilizando herramientas como vi, mkdir. Chown, chmod, también se puede crear un firewall maestro.

### **6. Generación de una clave ssh para la cuenta maestro**

Se puede utilizar el programa ssh–keygen y colocar un password vacío para la clave privada, si se quiere hacer configuración automática del VPN.

### **7. Configuración automática del login ssh para la cuenta esclavo.**

Se copia nuevamente la clave pública generada en la cuenta esclavo bajo `.ssh/authorized_keys`, y se configuran los permisos de archivo así:

```
drwx----- 2 esclavo esclavo 1024 Abr 7 23:49 ./
drwx----- 4 esclavo esclavo 1024 Abr 24 14:05 ../
-rwx----- 1 esclavo esclavo 328 Abr 7 03:04 claves autorizadas
-rw----- 1 esclavo esclavo 660 Abr 14 15:23 hosts desconocidos
-rw----- 1 esclavo esclavo 512 Abr 21 10:03 random_seed
```

Siendo la primera fila `~slave/.ssh`, y la segunda `~slave`.

## 8. Seguridad ssh sobre los bastiones

Esto significa configurar en `sshd_conf`:

```
PermitRootLogin no
IgnoreRhosts yes
StrictModes yes
QuietMode no
FascistLogging yes
KeepAlive yes
RhostsAuthentication no
RhostsRSAAuthentication no
RSAAuthentication yes
PasswordAuthentication no
PermitEmptyPasswords no
```

La autenticación del password se desactiva, así sólo es posible loggarse con claves autorizadas.

## 9. Ejecución habilitada de ppp y de la ruta para ambas cuentas

Como la cuenta maestro es la raíz en este caso, no hay nada que hacer. Para la cuenta esclavo, las siguientes líneas aparecen en `/etc/sudoers`:

vpn

```
Cmnd_Alias VPN=/usr/sbin/pppd,/usr/local/vpn/route
```

```
slave ALL=NOPASSWD: VPN
```

Se puede ver que se están utilizando algunos scripts para configurar el ppp y las tablas de enrutamiento sobre el host esclavo.

## 10. Scripting

Sobre el host maestro hay un script completo inicial:

```
#!/bin/sh

# skeleton Archivo de ejemplo para construir scripts /etc/init.d/
# Este archivo puede ser utilizado para construir scripts para
/etc/init.d.

#

# Autor Miquel van Smoorenburg <miquels@cistron.nl>.
# Modificado por Debian GNU/Linux
# y por Ian Murdock <imurdock@gnu.ai.mit.edu>.

#

# Version: @(#)skeleton 1.6 11-Nov-1996 miquels@cistron.nl

#

PATH=/usr/local/sbin:/sbin:/bin:/usr/sbin:/usr/bin:/usr/bin/X11/:

PPPAPP=/home/slave/ppp

ROUTEAPP=/home/slave/route

PPPD=/usr/sbin/pppd

NAME=VPN

REDIR=/usr/local/bin/pty-redir

SSH=/usr/bin/ssh

MYPPPIP=192.168.0.1

TARGETIP=192.168.0.2
```

vpn

TARGETNET=193.6.37.0

MYNET=193.6.35.0

SLAVEWALL=polanski-out

SLAVEACC=slave

test -f \$PPPD || exit 0

set -e

case "\$1" in start)

echo setting up vpn

\$REDIR \$SSH -o 'Batchmode yes' -t -l \$SLAVEACC \$SLAVEWALL sudo  
\$PPPAPP >/tmp/device

TTYNAME=`cat /tmp/device`

echo tty is \$TTYNAME

sleep 10s

if [ ! -z \$TTYNAME ] then

\$PPPD \$TTYNAME \${MYPPPPIP}:\${TARGETIP}

else

echo FAILED!

logger "vpn setup failed"

fi

sleep 5s

route add -net \$TARGETNET gw \$TARGETIP

\$SSH -o 'Batchmode yes' -l \$SLAVEACC \$SLAVEWALL sudo \$ROUTEAPP ;;  
stop)

ps -ax | grep "ssh -t -l \$SLAVEACC " | grep -v grep | awk '{print  
\$1}' | xargs kill ;; \*)

# echo "Usage: /etc/init.d/\$NAME {start|stop|reload}"

echo "Usage: /etc/init.d/\$NAME {start|stop}"

exit 1

;;

11/23

```
vpn
```

```
esac
```

```
exit 0
```

Los esclavos utilizan un script para rutear la configuración (*/usr/local/vpn/route*):

```
#!/bin/bash/sbin/route add -net 193.6.35.0 gw 192.168.0.1
```

y su *.ppprc* es:

```
pasivo
```

## 5. Lo que sucede

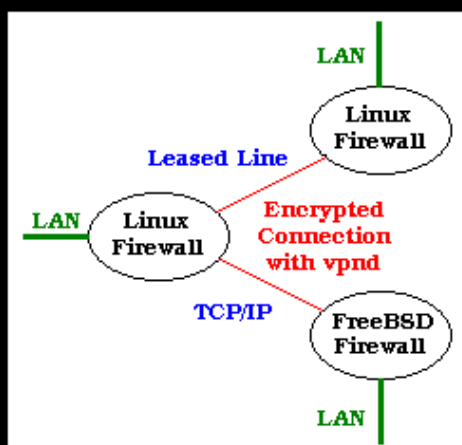
El maestro accede al esclavo, inicia *pppd* y redirecciona todo al *pty* local, así:

- \* Localizar un nuevo *pty*
  - \* Se hace *ssh* en el esclavo
  - \* Se ejecuta el *pppd* en el esclavo
  - \* El maestro ejecuta *pppd* en el *pty* local
  - \* Y se configuran las tablas de enrutamiento en el cliente
- 

# VPND – Virtual Private Network Daemon

## 1. GENERALIDADES

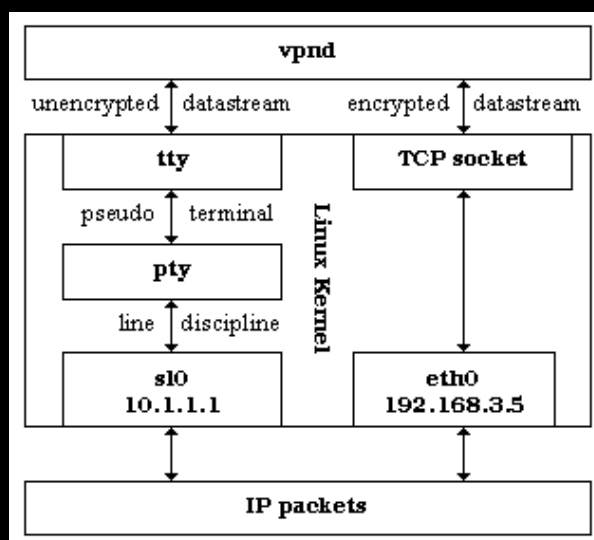
La red privada virtual demonio *vpnd* es un *demonio* que conecta dos redes en el nivel de red a través de TCP/IP o una línea (virtual) arrendada ligada a una interfaz serial. Todos los datos transferidos entre las dos redes se encriptan usando el algoritmo de encriptación Blowfish (no patentado y gratuito).



Vpnd no se propuso como reemplazo del software de seguridad en las comunicaciones existente como ssh o facilidades de *tunneling* del sistema operativo. Sin embargo, se propuso como un medio para asegurar la interconexión transparente de redes a través de canales potencialmente inseguros.

## 2. COMO TRABAJA?

La siguiente figura muestra básicamente como trabaja la vpnd (en lugar del socket TCP, vpnd puede usar un dispositivo serial):



La vpnd adquiere una pseudo terminal (un dispositivo par pty/tty) y enlaza una disciplina de línea SLIP a ella. El efecto de esto es que la vpnd ahora tiene su propia interfaz de red, una interfaz SLIP llamada slx donde x es algún número. Todos los paquetes IP enviados a esta interfaz se leen como un *datastream* por la vpnd y el datastream escrito por la vpnd reaparece como paquetes IP en esta interfaz.

A continuación, vpnd encripta el datastream leído y lo envía a través de una conexión TCP o sobre

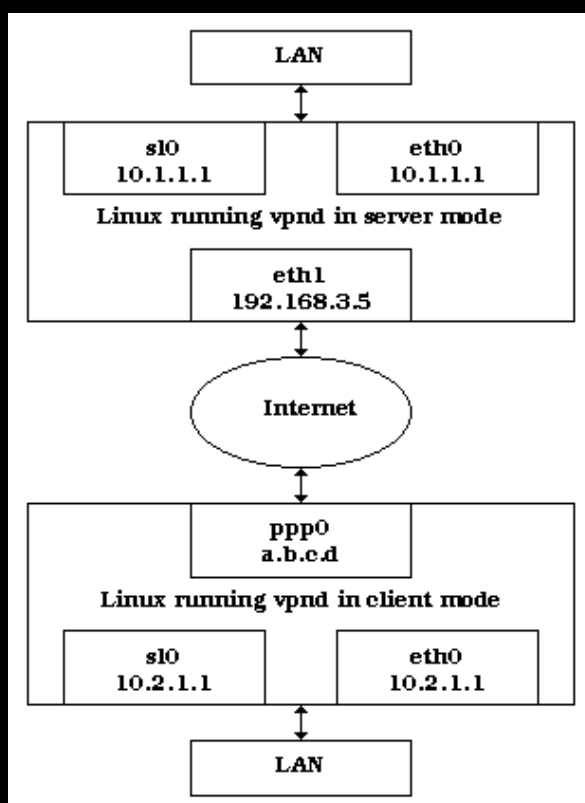
una línea serial a su par vpnd. El datastream recibido por vpnd de su par es descryptado y luego escrito en la pseudo terminal.

Como una vpnd no reconoce el datastream de la pseudo terminal, todos los paquetes escritos por el kernel a la interfaz SLIP son transportados.

Así la red de túneles vpnd trafica entre dos sistemas como un demonio de nivel de usuario.

### 3. CONEXION PAR Y SISTEMAS DIALUP (DE MARCADO)

La vpnd usa una conexión TCP a su par. Por esto en una conexión siempre hay un cliente y un servidor. El servidor debe tener una dirección IP fija; sin embargo, el cliente puede usar una dirección IP asignada dinámicamente:



La figura muestra que el sistema que corre vpnd en modo cliente no necesita una dirección IP fija. Una conexión dialup con asignación IP dinámica es suficiente.

Las interfaces SLIP de los pares vpnd existen si los procesos vpnd están corriendo, mientras que la conexión TCP entre ambos pares puede no existir.

Esto permite por ejemplo desconectar el sistema de Internet en modo cliente mientras existan conexiones telnet ociosas del cliente al servidor a través de interfaces SLIP. Si la conexión a

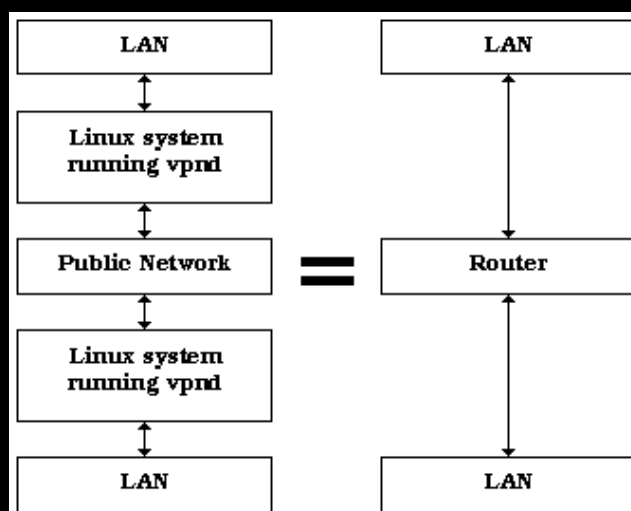
Internet se reestablece entonces en el cliente, el uso de la conexión telnet existente puede continuar.

Cuando se usa un demonio dialup Internet automático como *diald*, tanto la conexión a Internet como la conexión entre los pares vpnd serán automáticamente reestablecidas cuando el uso de la sesión telnet continúe en el cliente.

Cuando ambos pares vpnd usan direcciones IP fijas y tienen por ejemplo una conexión con línea arrendada (*leased line*) a Internet, la conexión TCP entre los pares vpnd puede virtualmente existir por siempre, estableciendo así una red privada virtual completamente funcional a través de Internet.

#### 4. VISTA FUNCIONAL

La vista funcional de una vpnd es muy simple como se observa en la siguiente figura:

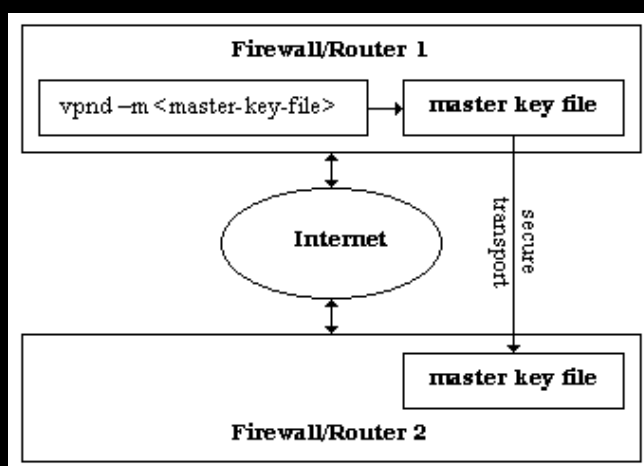


Los dos demonios vpnd actúan como un enrutador entre las dos LANs. La diferencia principal entre una vpnd y un enrutador es que la vpnd encripta todos los datos y los envía a su par vpnd. Además, es una solución de software puro que puede usarse con redes existentes para asegurar el tráfico en la red seleccionada.

#### 5. FORMATOS DE ARCHIVO CLAVE Y USO DE CLAVE

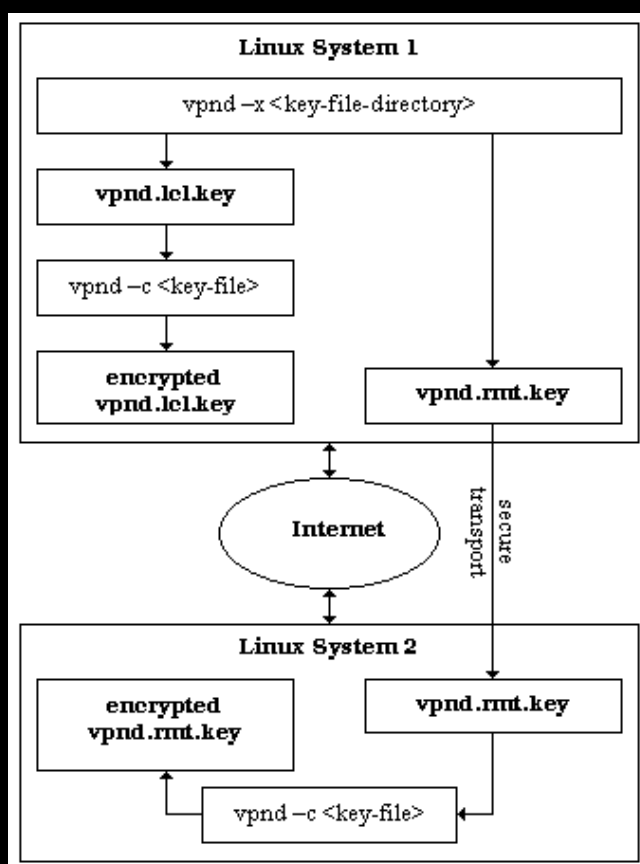
La conexión entre dos pares vpnd depende de una clave maestra compartida. Hay dos formatos de archivo para esta clave: el formato de archivo clave básico (para todas las versiones de vpnd) y el formato de archivo clave extendido (vpnd versión 1.0.5 y posteriores). Aún cuando ambos formatos de archivo contienen la misma clave maestra (el tipo, no el valor) hay algunas diferencias importantes.

El formato de archivo clave básico consiste de un archivo de 72 bytes que contiene una clave maestra de 576 bits no encriptada. Este formato de archivo se diseñó para conexiones entre firewall dedicados y sistemas de enrutadores (mientras no se puedan establecer otras conexiones además de la conexión vpnd a esos sistemas, nadie será capaz de robar el archivo clave remotamente). Un archivo clave tal se crea y utiliza como se muestra en la siguiente figura:



El formato de archivo clave extendido consiste de 2 archivos de 256 bytes. Ambos contienen la misma clave maestra de 576 bits. Las claves están encriptadas con datos almacenados en el otro archivo clave, así que los archivos pueden ser vistos como dos mitades de un banco de notas, donde se puede usar una mitad sin la otra. Cada vez que una conexión se establece entre los pares vpnd, se genera un nuevo dato encriptado aleatorio y se almacena nuevamente en el lado par. La longitud del dato encriptado es de 64 bits, así hay 264 posibilidades diferentes para una fuerza de ataque bruta. Si una clave robada se usara para establecer una conexión con un par vpnd legal, esta sería detectada por el propietario original de la clave robada y no estaría habilitada para conectarse a su par legal si la clave robada se usara dos veces o más frecuentemente. Si la clave robada se usa solo una vez, el hecho de que el dato encriptado anterior esté loggeado, esto también se notará. Además, los archivos clave pueden adicionalmente ser encriptados con un password suministrado por un usuario que debe ser presentado cuando vpnd se inicie (sea en la línea de comando o en la entrada estándar para habilitar el uso de un servidor clave *key server*).

Finalmente los archivos clave contienen un identificador de clave que debe ser presentado por el cliente vpnd cuando se establezca una conexión al servidor vpnd. Cuando se utilizan todos los mecanismos de seguridad, incluyendo el password suministrado por el usuario, una clave robada es virtualmente inutilizable. Así el formato de archivo clave extendido debería utilizarse en la mayoría de sistemas. Los archivos clave se crean y usan como se muestra en la siguiente figura:



## 6. LONGITUD DE CLAVE

Vpnd usa por defecto una clave de 576 bits. Si por requerimientos legales, la longitud de la clave tiene que reducirse, vpnd seguirá generando claves de 576 bits. Sin embargo, solo los primeros  $n$  bits ( $n$  = longitud de clave deseada) de una clave serán utilizadas, los bits no usados se consideran como basura.

## 7. SEGURIDAD

Vpnd usa la clave maestra compartida solo para la instalación de un buffer de limpieza (*whitening*) y el cambio de la clave de la primera sesión cuando se establece la conexión TCP o línea serial. Las claves de sesión constan de datos aleatorios. Como puede verse, la clave maestra sólo se usa para transferir datos aleatorios. Así es imposible recuperar la clave maestra espiando el tráfico de red.

La clave maestra compartida entre dos pares vpnd es una diferencia principal para muchos otros paquetes de software como ssh. Hay, sin embargo, una ventaja mayor: vpnd no requiere ninguno de los algoritmos de clave pública / privada patentados como RSA. El algoritmo Blowfish utilizado no está patentado (el alternativo, DES, tiene mucha más intensidad de cómputo, por lo que no es comúnmente implementado).

La longitud de clave por defecto y máxima de 576 bits es más que suficiente. Se tiene que recordar que esta es la longitud de clave de una cifra simétrica. Para estas cifras una longitud de clave de 128 bits se considera normalmente como muy segura. Así, la longitud de clave máxima debería ser

suficiente para cubrir el incremento del poder de cómputo de los próximos años.

## 8. EJEMPLOS

### Errores de configuración.

Hay errores típicos. Antes que nada asegúrese de que la `vpnd` esté construida con soporte de compresión (el script de configuración advertirá si este no es el caso) en ambos sistemas o use la opción de `nocompress` en ambos archivos de configuración. Luego asegúrese de que SLIP, así como CSLIP estén compilados en el kernel o como un módulo en ambos sistemas. Si no tiene soporte CSLIP debe usar la opción `nocslip` en ambos archivos de configuración. Finalmente, recuerde que `vpnd` utiliza una cifra simétrica, así que tiene que crear la clave o par de claves en el sistema y transferirla por medios seguros al otro sistema.

### Conexión entre sistemas IP estáticos

Este es el caso más simple. Básicamente, se usa si se quiere conectar dos firewalls o si se quiere tener una conexión LAN encriptada entre dos sistemas o dos redes. Normalmente, estos sistemas solo tendrán pocos y confiables usuarios. Para configurar la VPN necesita generar un archivo clave básico con el comando `vpnd -m /etc/vpnd.key` en un sistema y copiar el archivo a través de un medio seguro (por ejemplo `scp`, correo encriptado, entrega personal, etc.) al otro sistema. Como a tales sistemas les puede faltar la entropía suficiente al correr desatendidos, podría utilizar `/dev/urandom` como la fuente aleatoria para `vpnd`. Asumamos que quiere usar el puerto 538 en el cliente y el puerto 599 en el servidor. Finalmente los IPs para las interfaces SLIP usadas por `vpnd` deberían tomarse del rango privado 192.168.x.x. Se tiene entonces la siguiente configuración básica:

```

mode client
client 309.279.5.38 538
server 315.238.9.27 599
local 192.168.2.1
remote 192.168.1.1
keyfile /etc/vpnd.key
randomdev /dev/urandom

mode server
client 309.279.5.38 538
server 315.238.9.27 599
local 192.168.1.1
remote 192.168.2.1
keyfile /etc/vpnd.key
randomdev /dev/urandom

```

La opción de autoruteo en el archivo de configuración previene al `vpnd` para que adicione una entrada que se crea automáticamente por el kernel.

Ahora se puede seleccionar en los sistemas cliente y servidor si se quiere alcanzar el sistema par no encriptado mediante su IP regular o encriptado a través de la IP privada (192.168.x.x). Si se quiere asegurar que la comunicación por defecto se haga encriptada, debería poner una entrada consistente de la IP privada y el hostname del sistema par en `/etc/hosts` en ambos sistemas, respectivamente (asegúrese de que `/etc/hosts` se usan antes de `DNS` o de otros métodos de lookup, esto se puede hacer modificando `/etc/nsswitch.conf`).

Si se quiere configurar el enrutamiento entre las redes cliente y servidor (asumiendo que ambas redes son clase C). Esto es un poco artificioso. Si solo se estableciera una ruta a la red par a través de la IP privada del par, podría ocurrir lo siguiente:

```
paquete a la red par -> SLIP -> vpnd -> conectar al par -> enrutamiento ->
paquete a la red par
```

Como se ve, el enrutamiento simple causaría un loop de enrutamiento, así que nunca se podría comunicar encriptadamente con el par. Sin embargo, hay una solución simple para este problema: una ruta de host tiene precedencia sobre una ruta de red. Así, todo lo que se necesita agregar a la configuración es:

```
peeroute
```

```
route1 315.238.9.0 255.255.255.0
192.168.1.1
```

```
peeroute
```

```
route1 309.279.5.0 255.255.255.0
192.168.2.1
```

Puede pasar que la `vpnd` no sea capaz de encontrar automáticamente una ruta del host al sistema par, entonces debería utilizar el nombre de la interfaz de red (p.ej. `eth0`) a través del cual el sistema par puede alcanzarse como un parámetro de la opción `peeroute`.

Si se tiene una red rápida (p.ej. una LAN) entre los dos sistemas, se podría considerar el uso de las opciones `nocompress` y `nocslip` en ambos archivos de configuración. Esto incrementará el rendimiento de la VPN, pues la encriptación y transferencia de datos no comprimidos es más rápida en tales redes que la compresión o descompresión de datos. El punto de falla, donde la compresión debería ser deshabilitada, depende de la velocidad de la red así como del desempeño de la CPU. Para un primer estimativo se podría utilizar una tasa de transferencia de datos de 300 Kbps.

### Conexión Hogar – Oficina

Un escenario muy común es cuando el sistema casero es un sistema dialup que se le asigna una IP dinámica cada vez que se conecta a Internet.

El sistema casero debe correr con un firewall configurado adecuadamente y un `vpnd` en modo cliente. Si hay otros sistemas conectados mediante una LAN casera al sistema casero, ésta debería usar un rango privado de una red IP, p.ej. en una red clase C usar la red privada `192.168.2.0`.

La red de oficina está conectada estáticamente a Internet. La red VPN es el punto final de la red de oficina (`vpnd` en modo servidor) es el firewall en sí mismo o apoderado transparentemente desde el firewall a un sistema interno. Se debe tener especial cuidado si el firewall de oficina externo IP y el sistema interno IP están en la misma red.

Para hacer las cosas más fáciles, asumamos que los sistemas IP de oficina internos están en diferentes redes que el firewall externo IP. Así, los sistemas internos están en la red clase C `192.168.1.0` y el IP externo del firewall es `307.268.4.9` (esto no es problema si la `vpnd` de oficina corre en el firewall o en un sistema interno detrás de un proxy transparente del firewall, en el último caso reemplaza la IP de la opción `server` del archivo de configuración del servidor con esta IP regular de sistema).

Se debería usar el formato de archivo clave extendido así como la autenticación de mensaje HMAC como precauciones adicionales. Ejecute `vpnd -x /var/adm` en el sistema oficina corriendo `vpnd` en modo servidor, renombre `/var/adm/vpnd.lcl.key` a `/var/adm/vpnd.office.key` y transfiera `/var/adm/vpnd.rmt.key` personalmente a su sistema casero donde lo debe renombrar a `/var/adm/vpnd.home.key`. La configuración base resultante es (el servidor `vpnd` escucha en el puerto 379):

```

mode client                                mode server

client 0.0.0.0                              client 0.0.0.0
server 307.268.4.9                          server 307.268.4.9
local 192.168.2.1                           local 192.168.1.1
remote 192.168.1.1                          remote 192.168.2.1
keyfile /var/adm/vpnd.home.key               keyfile /var/adm/vpnd.office.key
randomdev /dev/urandom                       randomdev /dev/urandom
hmac 2 md5                                   hmac 2 md5
route1 192.168.1.0 255.255.255.0            route1 192.168.2.0 255.255.255.0
192.168.1.1                                 192.168.2.1

```

Ahora hay que recordar algo. Se está usando una conexión dialup. Normalmente algunos demonios no requieren marcado y colgado después de la conexión a Internet como desocupado (excepto para mensajes TCP FIN/ACK) por algún tiempo, dígame 5 minutos. Así la conexión VPN debería desconectarse después de cuatro minutos de tiempo ocioso en esta conexión.

La opción *linkup* en el sistema casero se puede utilizar p.ej. para establecer el tiempo del sistema casero a través de un script. Finalmente, si los sistemas de oficina en la IP de oficina externa están en la misma red, no puede usar una ruta de red para la opción *route1* de la configuración cliente. En lugar de eso debe usar la *route1* a través de las opciones *route9* para rutas de subred y host que excluyan la IP externa del firewall oficina. Si la suma de opciones de enrutamiento es insuficiente para este propósito se requiere agregar y borrar las rutas requeridas mediante scripts. Para correr los scripts se utilizan las opciones *slipup* y *slipdown*.

---

## CIPE – ENCAPSULACIÓN DEL CRIPTOGRAMA IP

Este es un proyecto prolongado para construir enrutadores IP encriptados. Está diseñado para paso de paquetes encriptados entre enrutadores previamente organizados en la forma de paquetes UDP. Esto no es tan flexible como el IPSEC pero es suficiente para el propósito original: conectando subredes firmemente sobre un tránsito inseguro de redes.

La única implementación disponible por ahora es un controlador de Kernel para Linux. Otra implementación, un controlador de nivel de usuario (*user-level*) para Linux y sistemas BSD fue implementado a medias como un test-bed pero fue abandonado. Estas implementaciones están disponibles gratuitamente bajo el GNU GPL o condiciones menos restrictivas.

CIPE es una descripción de un protocolo para una encriptación IP ultraliviana.

La meta principal de este software es proporcionar facilidades para asegurar la interconexión a través de un paquete sobre una red insegura tal como lo es Internet.

## GENERALIDADES

Este protocolo fue diseñado para ser simple y eficiente y para trabajar con facilidades en comunicaciones existentes, especialmente para encapsulamiento de paquetes UDP. La compatibilidad con protocolos existentes tales como el IPSEC RECs no fueron concernientes. Las primeras implementaciones de prueba fueron hechas sobre el nivel de usuario, mientras que últimamente se publicó una que consiste en un modulo kernel y un programa de nivel de usuario.

El modelo CIPE asume un enlace fijo entre dos pares con intercambio de datagramas. La forma más común de implementarlo es enviando mensajes UDP entre ellos (otra forma podría ser encapsulamiento por medio de PPP).

El protocolo CIPE consta de dos partes: encriptación y suma de verificación de los paquetes de datos e intercambio de claves dinámico.

## ENCRIPCIÓN DE PAQUETES

Cada datagrama IP se toma como un todo, incluyendo la cabecera. Se rellena al final con cero a siete octetos aleatorios tal que la longitud total en octetos es congruente a tres módulo ocho. Al paquete relleno se le adiciona un octeto del valor P y el CRC-32 al paquete construido hasta el momento e incluyendo P. Esto hace que la longitud del paquete completo sea un múltiplo de ocho octetos.

El valor P se da así: los bits 6, 5, 4 indican la longitud del relleno entre el final del paquete original y P. Los bits 2 y 1 son un tipo de código e indican la clase de paquete. Los bits sobrantes 7, 3 y 0 están reservados y deben ser cero.

Los tipos de código son:

00 – dato

01 – intercambio de clave

10 – reservado

11 – reservado

## CONSIDERACIONES DE SEGURIDAD

Los paquetes que se transmiten no llevan ninguna información relativa al datagrama IP original aparte de su longitud, rellena a un múltiplo de ocho. Esto debería guardar efectivamente contra la mayoría de los aspectos de análisis de tráfico.



Worldvision Tunnel Vision desarrolló la red privada virtual (VPN) soportada en la worldvision Weaver.

## QUÉ ES?

La Tunnel Vision creó una red privada virtual encriptada o VPN, entre dos sitios en Internet. Esto significa que los sitios con un servidor Linux corriendo Tunnel Vision o con worldvision Weaver actúan como un gateway Internet.



---

## REFERENCIAS

- \* <http://www.crosswinds.net/nuremberg/~anstein/unix/vpnd.html>
- \* <http://www.linuxpowered.com/html/links/networking.html>
- \* <http://www.linuxdoc.org/HOWTO/VPN-HOWTO.html>
- \* <http://www.worldvisions.ca/tunnelv/>
- \* <http://sites.inka.de/sites/bigred/devel/cipe.html>
- \* [RFC 1918](#)

---

### ELABORADO POR:

*Lina María Martínez Restrepo*  
*Catalina Andrea Ramos Rueda*

*Octubre 4 de 2000*

