Anexo

Anexo

Índice del Anexo

Índice de figuras y tablas del anexo

Índice de cuadros del anexo

Introducción

Las pruebas que se realizaron en el Laboratorio de Tecnologías Emergentes de Redes (NETLab) de la Universidad Nacional Autónoma de México (UNAM), están proyectadas para poder tener alternativas para la migración a IPv6 en la RedUNAM. Las VPNs han existido durante algún tiempo y también el IPv6, así como las redes privadas, virtuales o no. Como el IPv6 ha alcanzado un grado de madurez que las hace usables para los objetivos de NETLab, surge la necesidad de realizar pruebas con ambas tecnologías a fin de saber los alcances de las mismas. Se eligió el software OpenVPN por ser robusto, su tipo de licencia y capacidades de crecimiento. En virtud de su estado actual de desarrollo que es muy prometedor y estable.

Mencionaremos un problema que OpenVPN viene acarreando desde sus comienzos en las plataformas Windows y es la ausencia de soporte, tanto del software OpenVPN como del sistema operativo en cuestión, para la interfaz TUN. Esta interfaz es la que se maneja para la mayoría de los sistemas tipo Unix, pero cuyo soporte por parte de Microsoft es deficiente para su sistema operativo, y sin embargo existe la interfaz TAP. La diferencia entre este par de interfaces es que TUN esta diseñado como un dispositivo virtual para la conexión punto a punto, mientras que TAP esta diseñado como un dispositivo virtual para la conexión ethernet. La interfaz tun sólo soporta frames IP y tap sólo soporta frames ethernet. En la página en línea de OpenVPN, http://openvpn.net/index.php/open-source/documentation/install.html?start=1, refiere que la interfaz TAP cuenta con soporte limitado para la interfaz TUN. Pero esto no tuvo resultados satisfactorios cuando se usó con los sistemas Microsoft XP y Vista.

OpenVPN es una solución de conectividad en software, del tipo SSL VPN, para redes virtuales que tiene como cualidad ser escalable via web, es independiente del hardware. Soporta servicios VPN’s seguros y escalables a través de la Internet. Puede trabajar con las soluciones empresariales existentes y habilita la interacción en tiempo real de aplicaciones colaborativas. Es capaz de implementar modos básicos de conexión, en capa 2 o capa 3, con lo que se obtienen túneles capaces de enviar información en protocolos distintos al IP. Provee soporte para conexiones del tipo proxy. Funciona a través del proxy y puede ser configurado para ejecutar como un servicio TCP o UDP y además como servidor (simplemente esperando conexiones entrantes) o como cliente (iniciando conexiones). Solo un puerto en el firewall debe ser abierto para permitir conexiones, dado que desde OpenVPN 2.0 se permiten múltiples conexiones en el mismo puerto TCP o UDP. Las interfaces virtuales (tun0, tun1, etc.) permiten la implementación de reglas de firewall muy específicas. Alta flexibilidad y posibilidades de extensión mediante scripting. OpenVPN ofrece numerosos puntos para ejecutar scripts individuales durante su arranque. Soporte transparente para IPs dinámicas. Se elimina la necesidad de usar direcciones IP estáticas en ambos lados del túnel.

OpenVPN
No es un estándar y no es compatible con IPsec.
Disponible mayoritariamente como software, en casi todos los sistemas operativos existentes, y con algunas implementaciones en hardware.
Tecnología nueva y aun en crecimiento, suficientemente probada.
Sin interfaces gráficas profesionales, aunque ya existen algunos proyectos prometedores
Tecnología sencilla, no necesita modificaciones en la pila del protocolo IP
Usa las interfaces de red establecidas y paquetes estandarizados, no necesita modificar el kernel
Se ejecuta en el espacio del usuario y puede ser configurada para tener permisos elevados
Usa tecnologías de cifrado estandarizadas, no hay diferentes implementaciones o versiones entre varios proveedores.
Diseño fuertemente modular y facilidad de configuración, buena estructuración.
Curva de aprendizaje reducida, fácil de aprender y sencillo para principiantes
Solo Utiliza un puerto del firewall, que puede ser asignado de acuerdo a las necesidades. Por definición de la IANA es el puerto *1194*
Trabaja con servidores de nombres dinámicos como DynDNS o No-IP con reconexiones rápidas y transparentes.
Uso de SSL/TLS como estándar de criptografía.
Control de tráfico (Traffic shaping)
Velocidad (más de 20 Mbps en máquinas de 1Ghz), dependiendo de la conexión.
Alta compatibilidad con firewall y proxies
Ningún problema con NAT (ambos lados puede ser redes NATeadas)
Posibilidades para usuarios remotos (road warriors), dependiendo de las configuraciones y políticas.

Tabla 1 “Características generales de OpenVPN”

La tabla 1 nos muestra a detalle las principales características del OpenVPN. En cuanto a la técnicas conocidas cómo *bridging* y *routing* se pueden usar y de hecho se usan con OpenVPN, dependerá del caso que se esté manejando. El bridging es la técnica para crear una red de área amplia, virtual que se ejecute bajo una subred. Esencialmente es combinar una interfaz Ethernet (física) con una o más interfaces virtuales y ponerlas bajo la cobertura de una solo interfaz compartiendo una solo dirección IP. Las ventajas de usarlo son las siguientes:

4) Se anuncia a través de la VPN – esto permite al software que depende del broadcast como NetBIOS de windows que funciones el compartir archivos y la navegación entre vecinos .
5) No necesita configuraciones de router.
6) Trabaja con cualquier protocol que pueda funcionar sobre ethernet, incluyendo IPv4, IPv6, Netware IPX, AppleTalk, etc.

7) Solución fácil de configurar para los usuarios remotos (road warriors).

Una de sus desventajas sería:

4. Menor eficiencia que en el enrutamiento, y no se escala bien.

Las ventajas del enrutamiento son:

F) Eficiencia y alta escalabilidad.
G) Permite una mayor despersonalización del MTU para mayor eficiencia.

Y algunas desventajas:

- Los clients deben usar servidores WINS (tales como samba) para permitir la navegación a través de la VPN.
- Los Enrutadores deberán ser configurados para dar servicio a cada subred.
- El software que dependa del broadcasts no "podra" ver las máquinas del otro lado de la VPN.
- Trabaja con IPv4 en general, y IPv6 en casos donde las interfaces tun en ambos lados de la conexión lo soporten explicitamente.

Las versiones disponibles son desde la 1.6.0 hasta la versión actual, 2.1.1. Las versiones probadas fueron 2.0.6, 2.0.7, 2.1_rc1 hasta la versión 2.1_rc13, y finalmente la versión 2.1.1. El soporte para IPv6 se empezó a dar en la versión 2.0.x por parte del desarrollador argentino Juan José Ciarlante en el modo P2P, y para el modo servidor se dio en la versión 2.1.x por parte de los desarrolladores alemanes Gert Doering y Bernhard Schmidt.

Objetivo

Informar acerca de las pruebas que se han realizado sobre los escenarios de configuración básica de equipos, configuración de túneles, y funcionamiento.

Recursos

Hardware

- 4 PCs.
- 4 Tarjetas de red Ethernet.
- Cables para conexión RJ45.

Software

- Sistemas operativos tipo Linux [Fedora 11 Leónidas, Debian 5 Lenny].
- Sistemas operativos tipo BSD [OpenBSD 4.2 y 4.6, FreeBSD 7.1 y 8]
- Sistemas operativos tipo Windows [XP SP3 y Vista Home Edition]

Resumen de Pruebas

Las pruebas que se llevaron a cabo fueron:

- Verificación del soporte IPv6
- Configuración manual de túneles en equipos del mismo segmento.
- Configuración manual de túneles en equipos de diferentes segmentos.

Desarrollo y resultados

Verificación del soporte IPv6

Esta prueba consistió en verificar el soporte IPv6 en los sistemas operativos con que se cuenta.

Para verificar el soporte IPv6 en Windows XP se utilizó una ventana de comandos y se teclea el siguiente comando:

Ping ::1

Si el sistema operativo cuenta con soporte IPv6 se mostrara un resultado como el que se muestra:

```
Microsoft Windows [Versión 5.1.]
Copyright (c) 2006 Microsoft Corporation.  Reservados todos los derechos.
C:\> ping ::1
Haciendo ping a ::1 desde ::1 con 32 bytes de datos:
Respuesta desde ::1: tiempo<1m
Respuesta desde ::1: tiempo<1m
Estadísticas de ping para ::1:
```

```
    Paquetes: enviados = 2, recibidos = 2, perdidos = 0    (0% perdidos),
Tiempos aproximados de ida y vuelta en milisegundos:
    Mínimo = 0ms, Máximo = 0ms, Media = 0ms
```

Cuadro 1. "Respuesta positiva del loopback para IPv6 en un sistema MS Windows"

De lo contrario:

```
C:\>ping ::1
La solicitud de ping no pudo encontrar el host ::1. Compruebe el nombre y vuelva a
intentarlo.
```

Cuadro 2. "Respuesta negativa del loopback para IPv6 en un sistema MS Windows"

FreeBSD y OpenBSD

Los sistemas *BSD en sus últimas versiones no tienen soporte IPv6 habilitado por defecto, este se tiene que habilitar en los respectivos archivos de configuración para poder dar de alta en el sistema este soporte.

En el caso de *OpenBSD* se tiene que editar el archivo *sysctl.conf*, donde hay que descomentar las líneas que dicen y asignar los valores pertinentes.

net.inet6.ip6.forwarding=0 # 1=Permit forwarding (routing) of IPv6 Packets
net.inet6.ip6.accept_rtadv=1 # 1=Permit IPv6 autoconf (forwarding must be 0)

A fin de que el sistema en cuestión pueda autoconfigurarse en el segmento en el que se encuentra. Cualquier otra configuración requerirá modificaciones adicionales y el uso del archivo de configuración *rc.conf* posiblemente.

Figura 1. "Archivo de configuración rc.conf de OpenBSD."

Para el caso de *FreeBSD* también se tiene que editar el archivo *rc.conf* ,cuyas variables por defecto se encuentran en */etc/default/rc.conf.* En el rc.conf por defecto encontraremos la sección llamada *### IPv6 options ###.* En esta parte vemos las variables que necesitamos copiar al */etc/rc.conf* para modificarlas y no alterar el archivo por defecto. Las variables son :

ipv6_enable="" # Set to YES to set up for IPv6
ipv6_network_interfaces="" # List of network interfaces (or "auto")

Estas son las variables que hay que modificar para que el sistema se autoconfigure.
Y para verificar que ya se tiene habilitado el IPv6 en una terminal se introduce el comando:

```
$ping6 ::1
PING6(56=40+8+8 bytes) ::1 --> ::1
16 bytes from ::1, icmp_seq=0 hlim=64 time=0.381 ms
16 bytes from ::1, icmp_seq=1 hlim=64 time=0.443 ms
16 bytes from ::1, icmp_seq=2 hlim=64 time=0.495 ms
--- ::1 ping6 statistics ---
3 packets transmitted, 3 packets received, 0.0% packet loss
round-trip min/avg/max/std-dev = 0.381/0.440/0.495/0.047 ms
```

Cuadro 3. "Respuesta positiva del loopback para IPv6 en un sistema FreeBSD"

Linux

En los sistemas Linux el soporte IPv6 esta habilitado por defecto, así que en una terminal se teclea:

```
$ping6 ::1
PING ::1(::1) 56 data bytes
64 bytes from ::1: icmp_seq=1 ttl=64 time=0.057 ms
64 bytes from ::1: icmp_seq=2 ttl=64 time=0.067 ms
64 bytes from ::1: icmp_seq=3 ttl=64 time=0.063 ms
--- ::1 ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 2999ms
rtt min/avg/max/mdev = 0.057/0.064/0.071/0.009 ms
```

Cuadro 4. "Respuesta positiva del loopback para IPv6 en un sistema GNU/Linux"

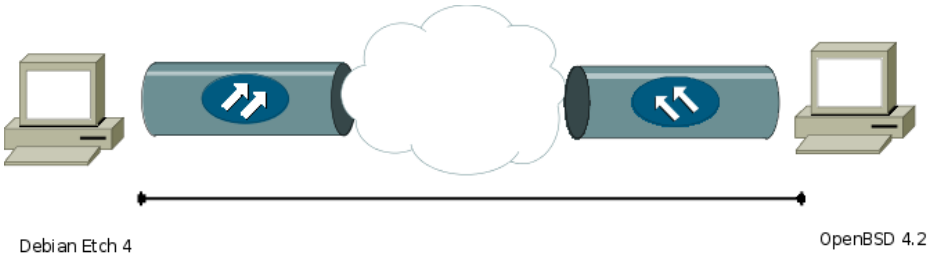

Figura 2. "Túnel manual entre dos equipos."

En esta parte se confirmo que todos los equipos y los sistemas operativos tuvieran conectividad y soporte para IPv6.

Configuración manual de túneles en equipos del mismo segmento, modelo cliente-cliente.

En equipos que cuenten con sistemas operativos GNU/Linux se pueden configurar túneles manuales de la siguiente manera:

GNU/Linux
En este sistema se creó una interfaz para el túnel y se configuró el final del mismo con los comandos:

```
# openvpn --remote 132.248.108.239 1194 --dev tun --ifconfig 10.4.0.2 10.4.0.1 --verb 9
```

Cuadro 5. "sentencia de inicio para openvpn en un sistema GNU/Linux en la línea de comandos"

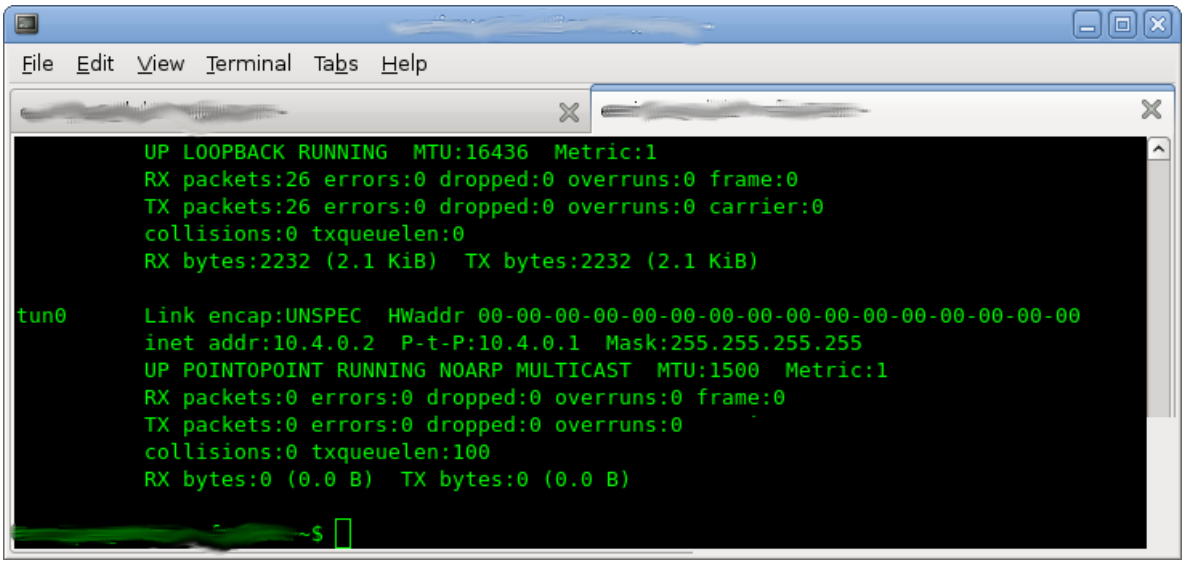

Figura 3. "Interfaz tun0 configurada ."

Para el sistema OpenBSD se creó una interfaz para el túnel y se configuró el final del mismo con los comandos:

```
# openvpn --remote 132.248.108.234 1194 --dev tun0 --ifconfig 10.4.0.1 10.4.0.2 --verb 9
```

Cuadro 6. "sentencia de inicio para OpenVPN en un sistema OpenBSD en la línea de comandos"

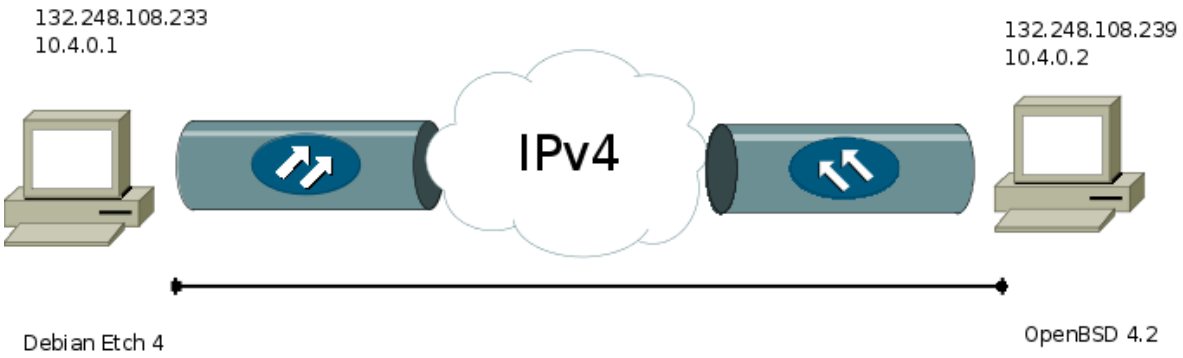

Figura 4." Túnel manual entre dos segmentos."

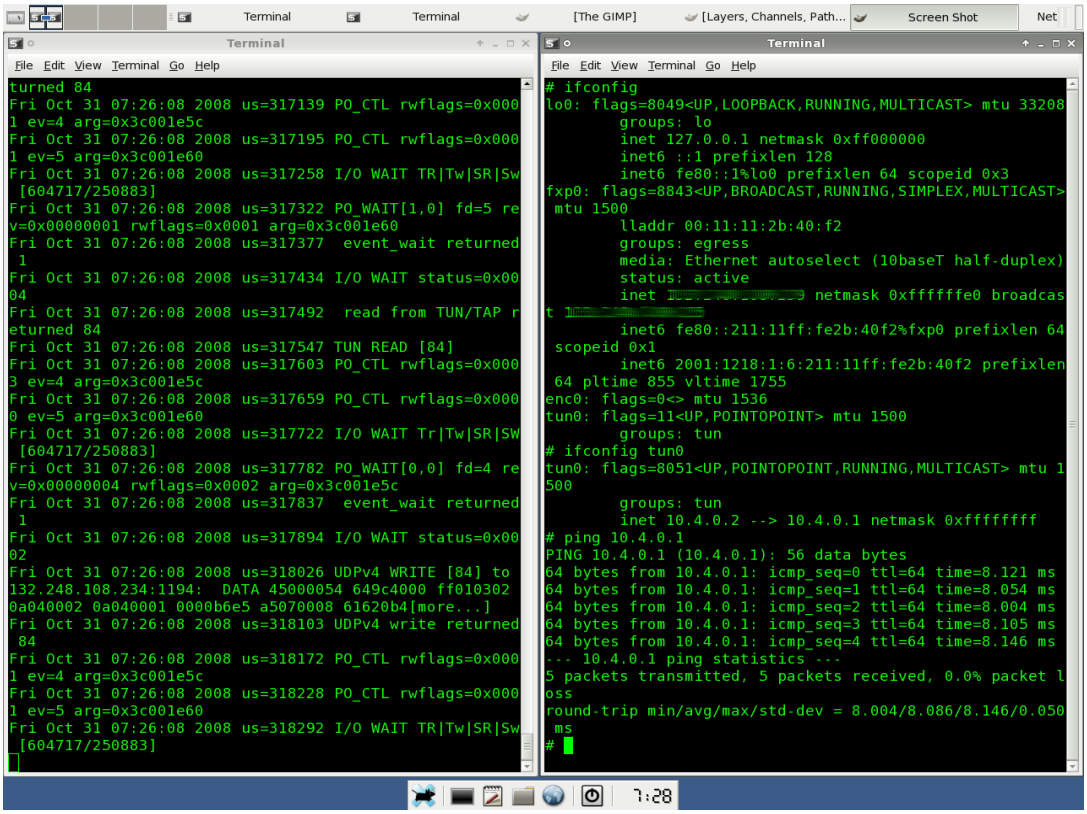

Figura 5. "Túnel manual entre OpenBSD y Debian."

Por último se realizaron pruebas de conectividad entre los hosts:

OpenBSD:

```
$ ping 10.4.0.1
64 bytes from 10.4.0.1: icmp_seq=1 ttl=64 time=0.104 ms
64 bytes from 10.4.0.1: icmp_seq=2 ttl=64 time=0.088 ms
64 bytes from 10.4.0.1: icmp_seq=3 ttl=64 time=0.087 ms
64 bytes from 10.4.0.1: icmp_seq=4 ttl=64 time=0.091 ms
--- 10.4.0.1 ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 2997ms
rtt min/avg/max/mdev = 0.087/0.092/0.104/0.011 ms
```

Cuadro 7. "Respuesta positiva de la dirección IP del primer cliente desde un sistema OpenBSD"

Debian:

```
[netlab@Netlab-3 ~]$ ping 10.4.0.2
64 bytes from 132.248.59.73: icmp_seq=1 ttl=64 time=0.105 ms
64 bytes from 132.248.59.73: icmp_seq=2 ttl=64 time=0.087 ms
64 bytes from 132.248.59.73: icmp_seq=3 ttl=64 time=0.099 ms
64 bytes from 132.248.59.73: icmp_seq=4 ttl=64 time=0.096 ms
--- 10.4.0.2 ping statistics ---
4 packets transmitted, 5 received, 0% packet loss, time 3999ms
rtt min/avg/max/mdev = 0.087/0.098/0.105/0.009 ms
```

Cuadro 8. "Respuesta positiva de la dirección IP de segundo cliente desde un sistema GNU/Linux"

En esta prueba verificamos que existiera conectividad y respuesta de cliente a cliente para la VPN sobre IPv4.

Configuración de un túnel con seguridad de llave estática, modelo cliente-cliente.

Para esta prueba se utilizaron los mismos equipos, con las mismas configuraciones anteriores con el agregado de la generación de la llave estática la cual creamos con el siguiente comando :

openvpn --genkey --secret key

Este comando construyó un archivo llave aleatorio (en formato **ASCII**). Se copió la llave a **OpenBSD** con un medio seguro como el programa/comando scp(1)

Se procedió a ejecutar el siguiente conjunto de instrucciones para recrear el túnel y se configuró de la misma manera además de agregar el uso de la llave pública:

En GNU/Linux:

```
#  openvpn --remote 132.248.108.239 1194 --dev tun --ifconfig 10.4.0.1 10.4.0.2 --verb 9 --secret key
```

Cuadro 9. "sentencia de inicio para OpenVPN en un sistema GNU/Linux en la línea de comandos"

Para OpenBSD

Se creó una interfaz para el túnel y se configuró el final del mismo con los comandos, agregando el uso de la llave estática:

```
# openvpn --remote 132.248.108.234 1194 --dev tun0 --ifconfig 10.4.0.2 10.4.0.1 --verb 9 --secret key
```

Cuadro 10. "sentencia de inicio para OpenVPN en un sistema OpenBSD en la línea de comandos"

Se verificó que el túnel estuviera trabajando

En GNU/Linux

```
[netlab@Netlab-3 ~]$ ping 10.4.0.2
64 bytes from 132.248.59.73: icmp_seq=1 ttl=64 time=0.105 ms
64 bytes from 132.248.59.73: icmp_seq=2 ttl=64 time=0.087 ms
64 bytes from 132.248.59.73: icmp_seq=3 ttl=64 time=0.099 ms
64 bytes from 132.248.59.73: icmp_seq=4 ttl=64 time=0.096 ms
--- 10.4.0.2 ping statistics ---
4 packets transmitted, 5 received, 0% packet loss, time 3999ms
rtt min/avg/max/mdev = 0.087/0.098/0.105/0.009 ms
```

Cuadro 11. "Respuesta positiva de la dirección IP del cliente desde el servidor un sistema GNU/Linux"

Y en OpenBSD

```
$ ping 10.04.0.1
64 bytes from 10.4.0.1: icmp_seq=1 ttl=64 time=0.104 ms
64 bytes from 10.4.0.1: icmp_seq=2 ttl=64 time=0.088 ms
64 bytes from 10.4.0.1: icmp_seq=3 ttl=64 time=0.087 ms
64 bytes from 10.4.0.1: icmp_seq=4 ttl=64 time=0.091 ms
--- 10.4.0.1 ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 2997ms
rtt min/avg/max/mdev = 0.087/0.092/0.104/0.011 ms
```

Cuadro 12. "Respuesta positiva de la dirección IP del servidor desde el cliente en un sistema OpenBSD"

En esta prueba verificamos que existiera conectividad y respuesta de cliente a cliente para la VPN sobre IPv4 con seguridad habilitada parcialmente.

Configuración de un túnel con seguridad TLS, modelo cliente-servidor**.**

Para esta prueba se utilizó el sistema GNU/Linux como servidor TLS y OpenBSD como el cliente. Debe mencionarse que la designación cliente-servidor solo tiene significado en el subsistema TLS, no tiene ninguna influencia en el modelo de comunicación punto a punto que utiliza OpenVPN.

En esta prueba se tuvieron que construir llaves por separado para el cliente y para el servidor, en nuestro caso GNU/Linux y OpenBSD. También se construyó un archivo de parámetros Diffie-Hellman.

Para OpenBSD se ejecutó:

```
# openvpn --remote 132.248.108.234 1194 --dev tun0 --ifconfig 10.4.0.2 10.4.0.1 --tls-client --ca
/usr/local/share/examples/openvpn/sample-keys/tmp-ca.crt –cert
/usr/local/share/examples/openvpn/sample-keys/client.crt --key
/usr/local/share/examples/openvpn/sample-keys/client.key --reneg-sec 60 --verb 9
```

Cuadro 13. "sentencia de inicio del cliente para OpenVPN en un sistema OpenBSD en la línea de comandos"

Para GNU/Linux se procede a ejecutar el siguiente conjunto de instrucciones :

```
# openvpn --remote 132.248.108.239 1194 --dev tun --ifconfig 10.4.0.1 10.4.0.2 --tls-server --dh
/usr/tools/openvpn-2.0.9/sample-keys/dh1024.pem --ca /usr/tools/openvpn-2.0.9/sample-keys/tmp-
ca.crt --cert /usr/tools/openvpn-2.0.9/sample-keys/server.crt --key /usr/tools/openvpn-
2.0.9/sample-keys/server.key --reneg-sec 60 --verb 9
```

Cuadro 14. "sentencia de inicio del servidor para OpenVPN en un sistema GNU/Linux en la línea de comandos"

Se verificó que el túnel estuviera trabajando en GNU/Linux

```
[netlab@Netlab-3 ~]$ ping 10.4.0.2
64 bytes from 132.248.59.73: icmp_seq=1 ttl=64 time=0.105 ms
64 bytes from 132.248.59.73: icmp_seq=2 ttl=64 time=0.087 ms
64 bytes from 132.248.59.73: icmp_seq=3 ttl=64 time=0.099 ms
64 bytes from 132.248.59.73: icmp_seq=4 ttl=64 time=0.096 ms
--- 10.4.0.2 ping statistics ---
4 packets transmitted, 5 received, 0% packet loss, time 3999ms
rtt min/avg/max/mdev = 0.087/0.098/0.105/0.009 ms
```

Cuadro 15. " Respuesta positiva a la IP del cliente desde el servidor en GNU/Linux "

Y en OpenBSD

```
$ ping 10.04.0.1
64 bytes from 10.4.0.1: icmp_seq=1 ttl=64 time=0.104 ms
64 bytes from 10.4.0.1: icmp_seq=2 ttl=64 time=0.088 ms
64 bytes from 10.4.0.1: icmp_seq=3 ttl=64 time=0.087 ms
64 bytes from 10.4.0.1: icmp_seq=4 ttl=64 time=0.091 ms
--- 10.4.0.1 ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 2997ms
rtt min/avg/max/mdev = 0.087/0.092/0.104/0.011 ms
```

Cuadro 16. " Respuesta positiva a la IP del servidor desde el cliente en OpenBSD "

En esta prueba verificamos que existiera conectividad y respuesta de cliente a servidor y viceversa para la VPN sobre IPv4 con seguridad habilitada en el modelo cliente-servidor.

Configuración de un túnel con seguridad TLS, modelo cliente-servidor para el caso GNU/Linux-MS Windows.

Para poder realizar la comunicación entre cliente-servidor de un sistema MS Windows© se utilizó la siguiente configuración. Para el servidor GNU/Linux

```
$openvpn --port 1194 --proto tcp --dev tun --ca /usr/local/share/examples/openvpn/sample-keys/tmp-ca.crt --cert /usr/local/share/examples/openvpn/sample-keys/client.crt
--key /usr/local/share/examples/openvpn/sample-keys/client.key  --dh /usr/tools/openvpn-2.0.9/sample-keys/dh1024.pem --server 10.8.0.0 255.255.255.0 --ifconfig-pool-persist fipp.txt
--keepalive 10 120 –comp-lzo --user nobody --group nobody –persist-key –persist-tun --status openvpn-status.log --verb 9
```

Cuadro 17. "Sentencia de inicio del servidor para OpenVPN en un sistema GNU/Linux en la línea de comandos."

La cual es una variación de la solución propuesta en el foro comunitario de Gentoo (http://en.gentoo-wiki.com/wiki/HOWTO_OpenVPN_Linux_Server_Windows_Client)

Del lado del cliente MS Windows© se utilizó

```
C:\ARCHIV~1\OpenVPN\sample-config>openvpn client.ovpn
```

Cuadro 18. "inicio del cliente OpenVPN y su archivo de configuración desde el promt de MS Windows."

El cual contiene:

```
client
dev tun
proto tcp-client
remote 132.248.108.234 1194
resolv-retry infinite
nobind
persist-key
persist-tun
ca tmp-ca.crt
cert client.crt
key client.key
comp-lzo
verb 9
```

Cuadro 19. "Contenido del archivo de configuración para MS Windows XP "

Se verificó que estuviera trabajando el túnel:

En GNU/Linux

```
[netlab@Netlab-3 ~]$ ping 10.8.0.6
64 bytes from 132.248.108.234: icmp_seq=1 ttl=64 time=0.105 ms
64 bytes from 132.248.108.234: icmp_seq=2 ttl=64 time=0.087 ms
64 bytes from 132.248.108.234: icmp_seq=3 ttl=64 time=0.099 ms
64 bytes from 132.248.108.234: icmp_seq=4 ttl=64 time=0.096 ms
--- 10.8.0.6 ping statistics ---
4 packets transmitted, 5 received, 0% packet loss, time 3999ms
rtt min/avg/max/mdev = 0.087/0.098/0.105/0.009 ms
```

Cuadro 20. "Respuesta positiva de la dirección IP del cliente desde el servidor GNU/Linux "

Y en MS Windows©

```
$ ping 10.4.0.1
64 bytes from 132.248.108.239: icmp_seq=1 ttl=64 time=0.104 ms
64 bytes from 132.248.108.239: icmp_seq=2 ttl=64 time=0.088 ms
64 bytes from 132.248.108.239: icmp_seq=3 ttl=64 time=0.087 ms
64 bytes from 132.248.108.239: icmp_seq=4 ttl=64 time=0.091 ms
--- 10.8.0.1 ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 2997ms
rtt min/avg/max/mdev = 0.087/0.092/0.104/0.011 ms
```

Cuadro 21. "Respuesta positiva de la dirección IP del servidor desde el cliente MS Windows"

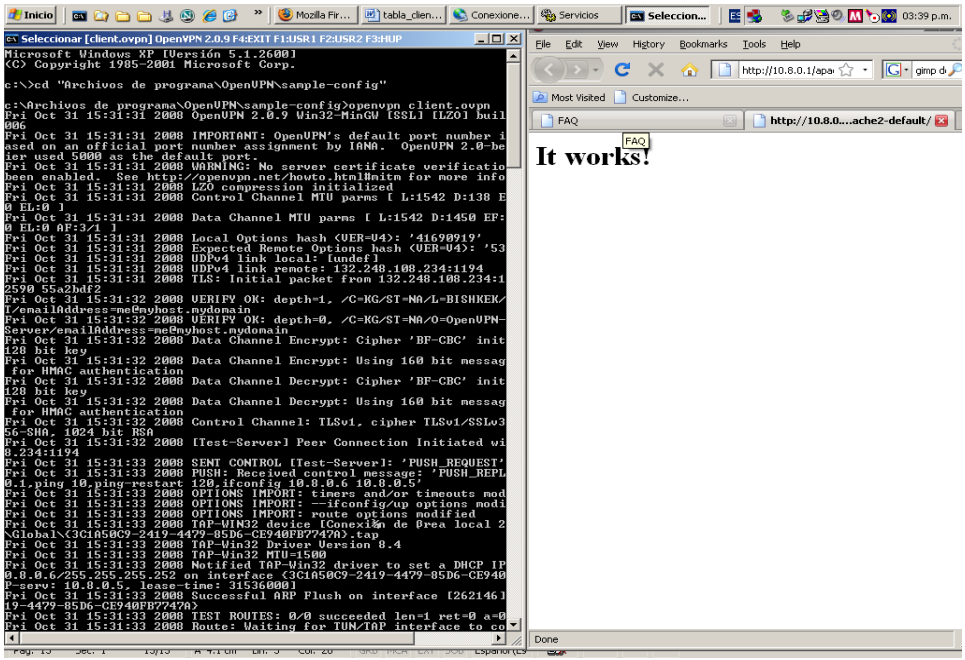

Figura 6. "Túnel entre un cliente MS Windows y un servidor Linux"

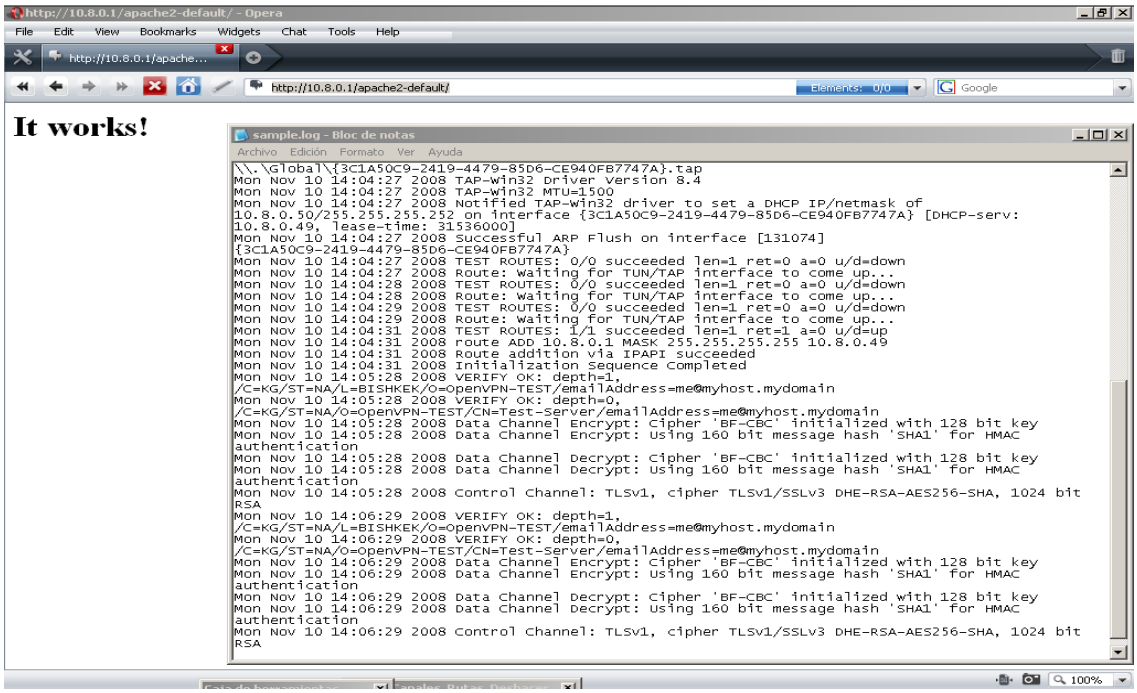

Figura 7. "Túnel entre un cliente MS Windows y servidor Linux bis."

En esta prueba verificamos que existiera conectividad y respuesta de cliente a servidor para la VPN sobre IPv4, con seguridad habilitada en el modelo cliente-servidor.

Configuración de un túnel con seguridad TLS, modelo cliente-servidor para el caso GNU/Linux-Ms Windows/OpenBSD.

Para esta configuración se utilizó la misma configuración que en el caso anterior para el sistema MS Windows© y la misma configuración para el sistema OpenBSD.

Ambos lograron la conexión.

En esta prueba verificamos que existiera conectividad y respuesta de cliente a servidor para la VPN sobre IPv4, con seguridad habilitada en el modelo cliente-servidor. Señalaremos que los sistemas que servían como clientes obtenían la misma dirección ip por causa de la configuración.

Generación de un ejecutable para el sistema MS Windows©

En esta etapa del proceso se tenía planeado generar un ejecutable nativo para todos los sistemas MS Windows© que se adecuara a nuestras necesidades, sin embargo sólo se pudo compilar y generar a partir del código fuente de la versión 2_rc7 del OpenVPN en donde todavía era posible compilarlo con herramientas propietarias de Microsoft como el Visual Studio express C, pero después de esa versión en el desarrollo de OpenVPN se decidió que cambiaría la forma de generar ejecutables para cualquier plataforma, dejando así lo anterior inservible e innecesario.

Configuración de un túnel con seguridad TLS, modelo cliente-servidor para el caso GNU/Linux-OpenBSD a partir del parche que ofrecía Juan José Cirlante.

Para esta prueba se utilizo el parche de Juan José Cirlante, el cual no implementa la capacidad de comunicación en IPv6 para OpenVPN, esta ya viene con el mismo, sino que la expande a más posibilidades.

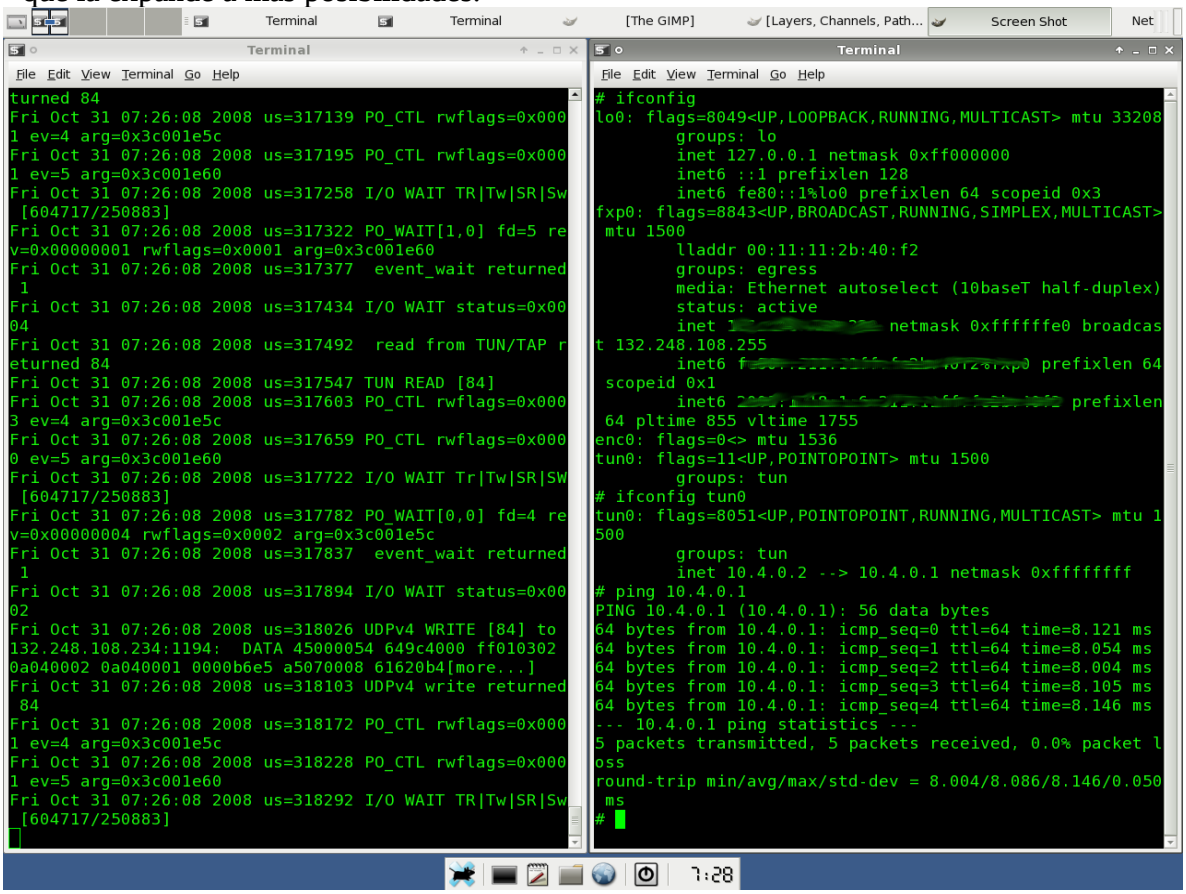

Figura 8. "Túnel con seguridad TLS, modelo cliente-servidor para el caso GNU/Linux-OpenBSD a partir del código parchado bis."

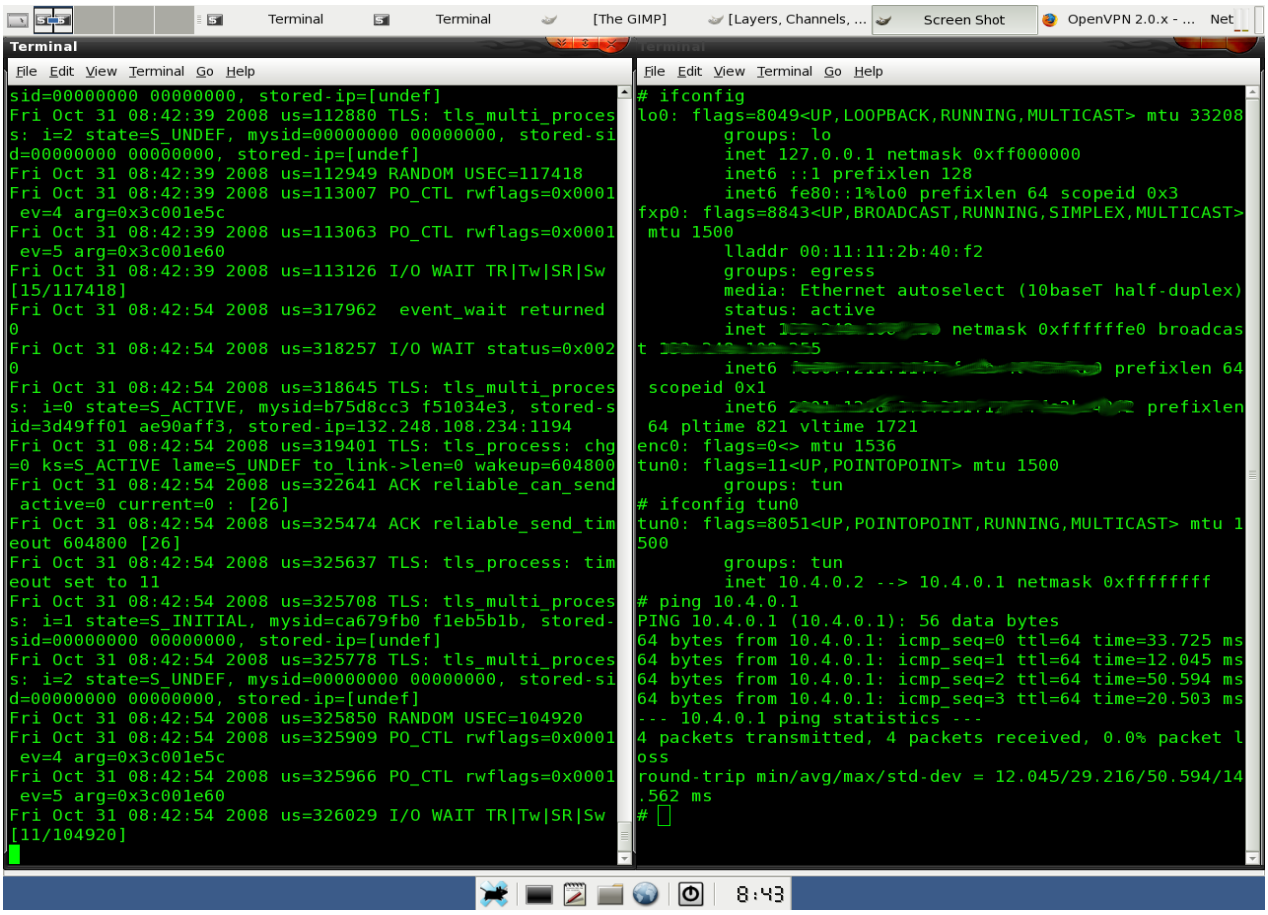

Figura 9. "Túnel con seguridad "

Se procedió a ejecutar el siguiente conjunto de instrucciones para recrear el túnel y se configuró de la misma manera, además de agregar el uso de la llave pública. Para OpenBSD 4.2

Cuadro 22. "Sentencia para ejecutar OpenVPN en un cliente OpenBSD con dirección local IPv6"

```
#  openvpn --remote fe80::2c0:4fff:fead:dcd2 1194 --dev tun0 –proto udp6 --ifconfig 10.4.0.2 10.4.0.1 –tls-client –dh /usr/local/share/examples/openvpn/sample-keys/dh1024.pem –ca /usr/local/share/examples/openvpn/sample-keys/tmp-ca.crt –cert /usr/local/share/examples/openvpn/sample-keys/client.crt –key /usr/local/share/examples/openvpn/sample-keys/client.key –reneg 60 -- verb 9
```

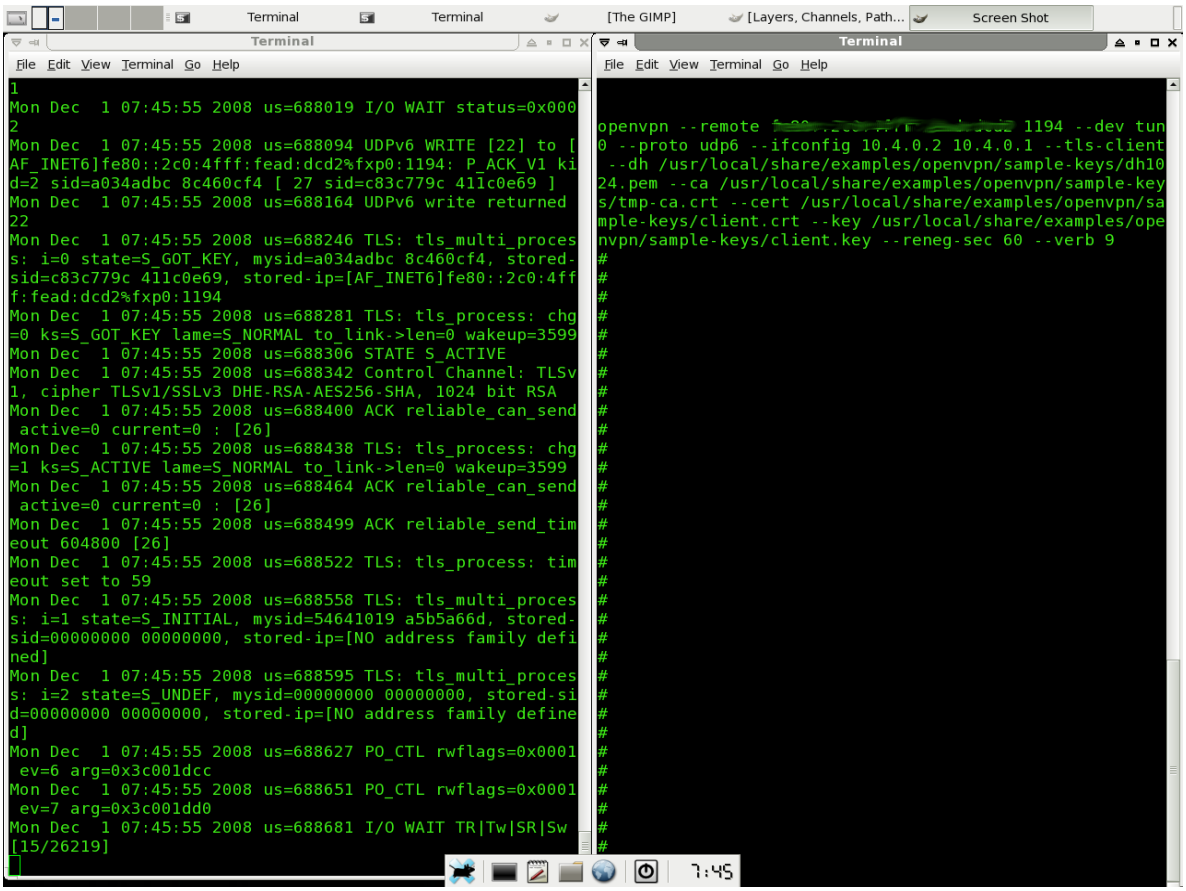

Figura 10. "Túnel con seguridad bis."

Para OpenBSD 4.2

```
#  openvpn --remote 2001:1218:1:6:2c0:4fff:fead:dcd2 1194 --dev tun0 –proto udp6
--ifconfig 10.4.0.2 10.4.0.1 –tls-client –dh /usr/local/share/examples/openvpn/sample-
keys/dh1024.pem –ca /usr/local/share/examples/openvpn/sample-keys/tmp-ca.crt –cert
/usr/local/share/examples/openvpn/sample-keys/client.crt –key
/usr/local/share/examples/openvpn/sample-keys/client.key –reneg 60 -- verb 9
```

Cuadro 23. "Sentencia para ejecutar OpenVPN en un cliente OpenBSD con dirección global IPv6"

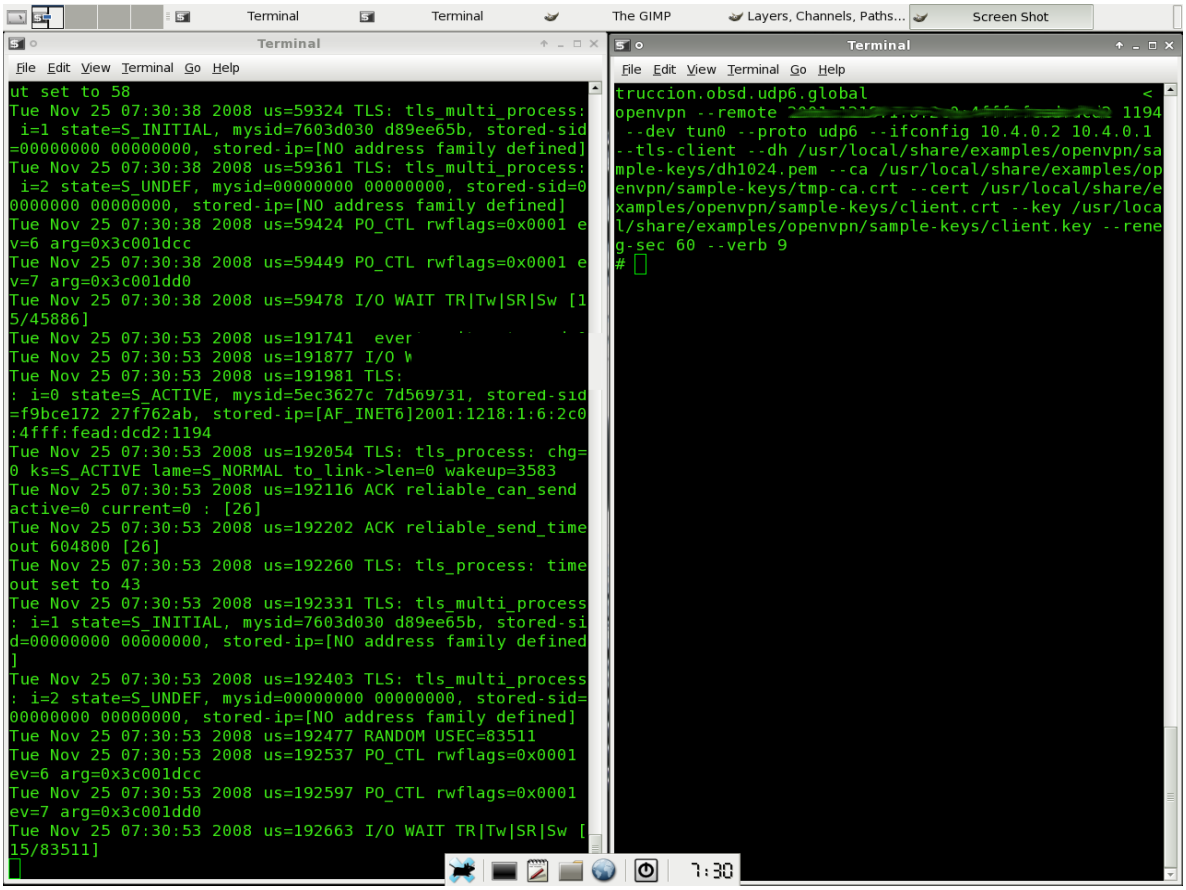

Figura 11. "Túnel con seguridad -tres"

En este caso se logro la conectividad y respuesta positiva desde cliente al servidor y viceversa, con seguridad habilitada en el modelo cliente servidor. En esta prueba no se incluyen clientes MS Windows porque no existe soporte para que estos convivan con clientes Unix.

Configuración de un túnel con seguridad TLS, modelo cliente-servidor para el caso con sistemas GNU/Linux en equipos de diferente segmento.

Para el caso de las pruebas que se realizaron entre equipos en diferentes segmentos utilizamos los mismos scripts del caso **8**. La única variación fue la dirección IP. Pero no se puedo lograr la conexión, entre ambos puntos , la respuesta observada fue el 'congelamiento' de la conexión que se era observada desde el servidor. Se presume que la infraestructura no es la adecuada para realizar estas pruebas.

10.Configuración de un túnel modelo cliente-cliente para el caso con sistemas MS Windows .

Para esta serie de pruebas se utilizaron las versiones XP profesional y Vista Home Edition. En ningún caso se logró la conexión con el otro extremo.

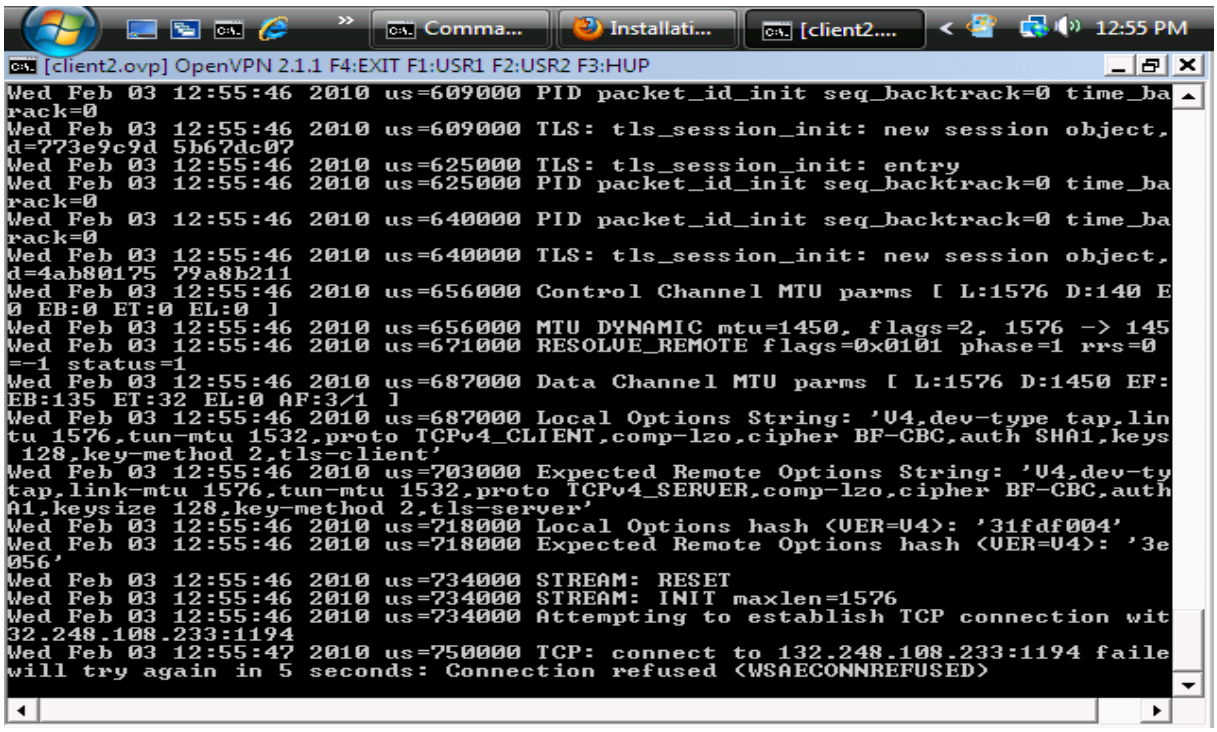

Figura 12. *"Conexión fallida Windows Vista Home Edition"*

11.Configuración de un túnel modelo cliente-cliente con seguridad de llave estática para el caso con sistemas MS Windows .

En estas pruebas los sistemas fallaron al intentar la conexión, como en el caso anterior.

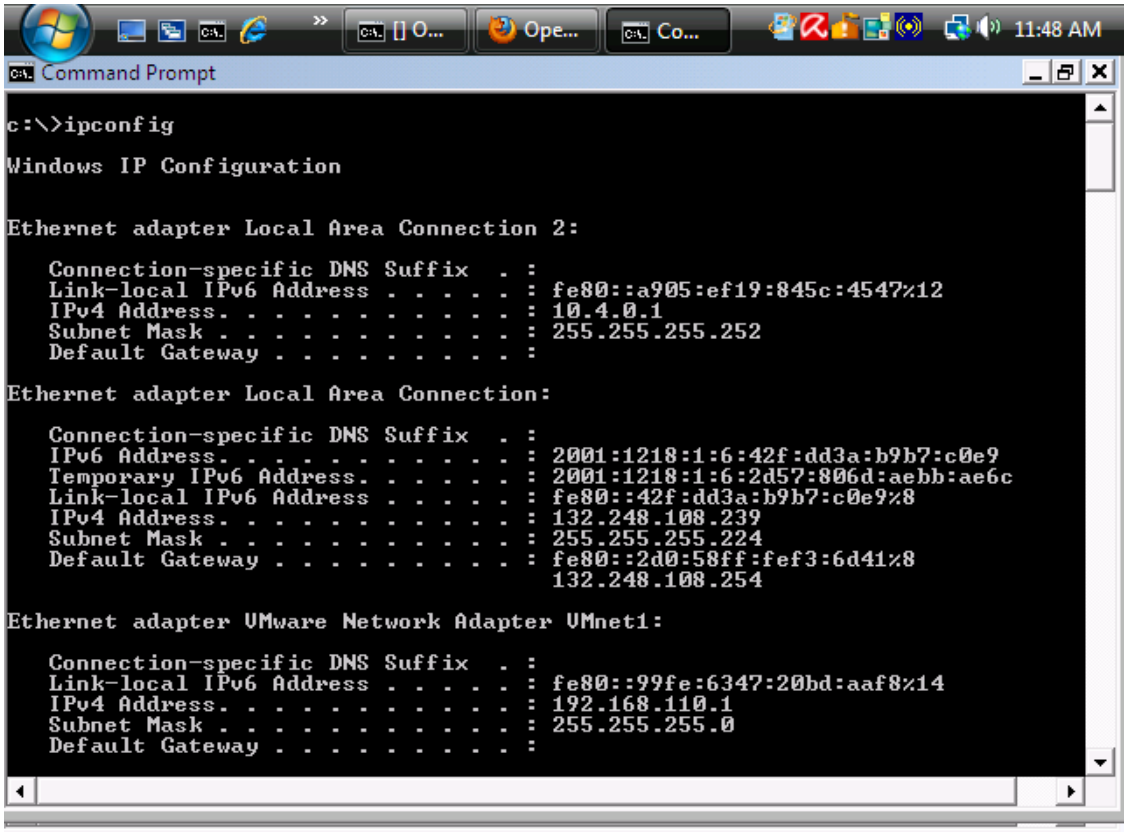

Figura 13. *"Configuración de un túnel IPv4"*

12. Configuración de un túnel modelo cliente-servidor con seguridad completa basada en TLS para el caso con sistemas MS Windows .

No fue posible lograr la conexión para los sistemas en prueba. Se hicieron pruebas de monitoreo de red en aislamiento para dichas pruebas pero sin embargo no se encontró una razón para la falla en estas pruebas con estos sistemas operativos.

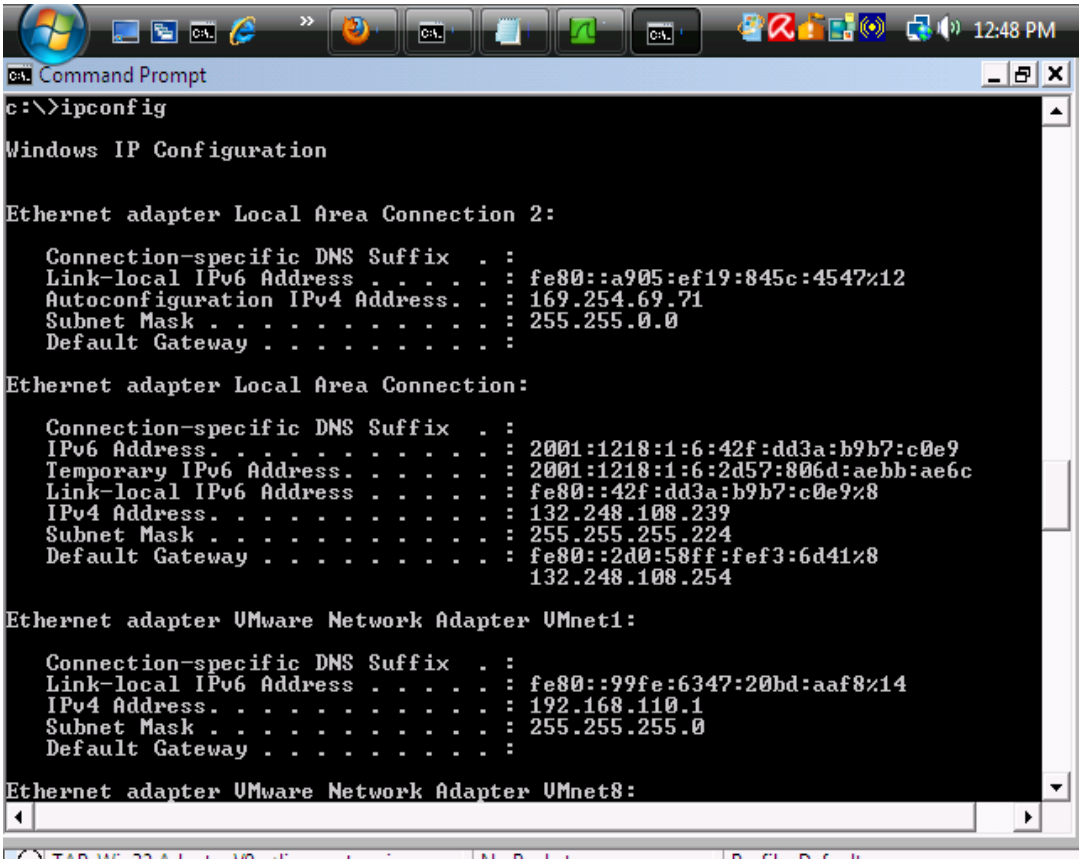

Figura 14. *"Configuración fallida de un túnel IPv4"*

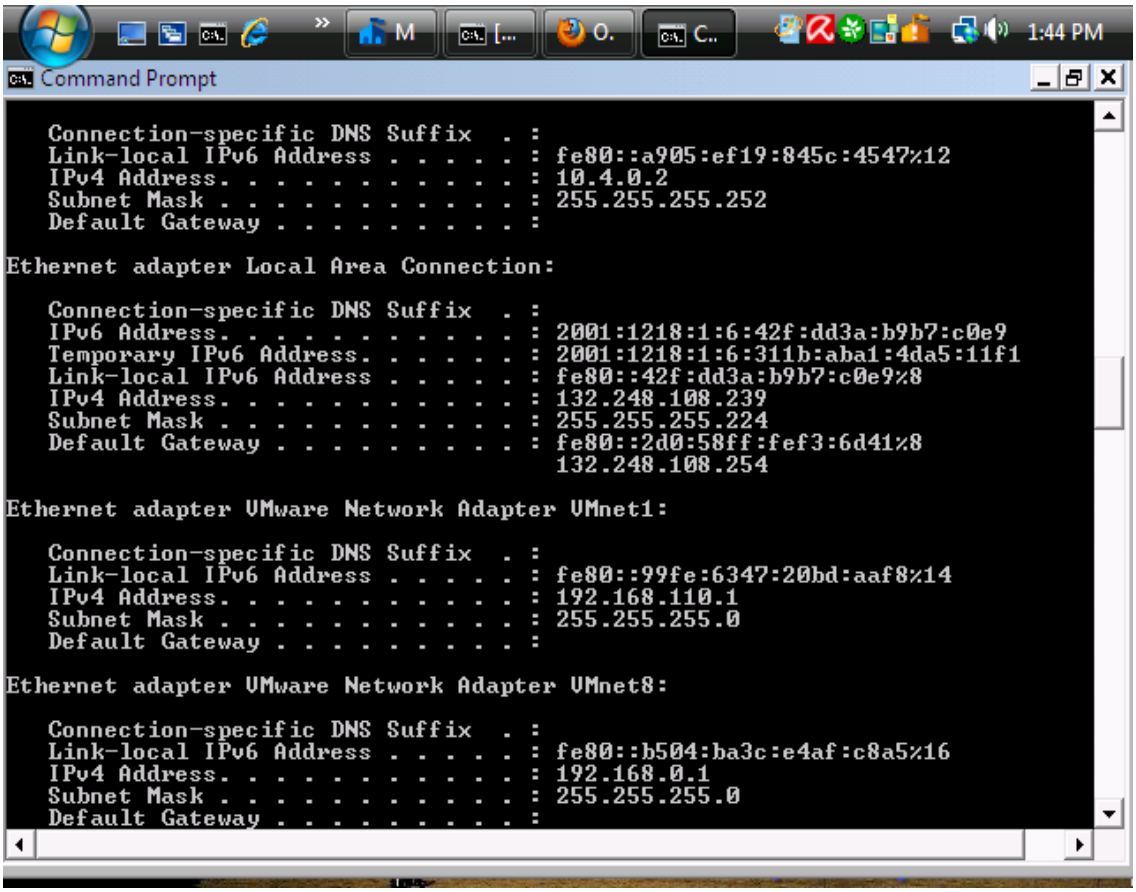

Figura 15. "Interfaces de red configuradas en Windows Vista Home Edition"

13.Configuración de un túnel con seguridad TLS, modelo cliente-servidor a partir del parche que ofrecían Bernhard Schmidt y Gert Döring.

Para este caso, se uso el parche que ofrecen los desarrolladores Bernhard Schmidt y Gert Döring. Este parche ofrece capacidades punto a multipunto con la interfaz tun o tap, punto a punto con interfaz tun y tap. Para instalarlo se procede de la siguiente manera:

1) Se descarga el código fuente de la página oficial del proyecto OpenVPN. Se obtendrá un paquete comprimido, puede ser en formato zip o tar.gz, llamado entonces openvpn N.tar.gz o puede ser openvpn N.zip. La N corresponderá al número de la versión en curso. Al momento de escribir este reporte es la versión openvpn-2.1.1

2) Se descomprime el paquete del código fuente obtenido. Dependerá del formato de compresión escogido para descargarlo. En el caso de un archivo zip y estando en un sistema tipo Unix, utilícese el comando unzip, para un *tarball* o sea un tar.gz, utilícese el comando tar -xfz . En caso de estar en un sistema tipo MS Windows , herramientas como WinZip o WinRar son suficientes para el propósito.

3) Una vez obtenido el directorio generado por el archivo descomprimido, descarge el parche de la página de los desarrolladores. http://www.greenie.net/ipv6/openvpn-2.1-ipv6-20100307-1.patch.gz . Al momento de escribir este reporte el parche es para la versión del punto *1*.

4) Para aplicar el parche del código fuente, descomprima el archivo con el comando gunzip. Después utilice el comando patch, de la manera siguiente.

$ patch < *archivo-de-.parche.patch*

A continuación verá como al código fuente original se le aplican ciertos cambios que como resultado nos darán lo necesario para nuestros propositos.

5) Una vez que ya se tiene el código fuente modificado se procede a compilarlo. Hay que tener en el sistema instaladas las bibliotecas de programación siguientes.

- Propenso
- LZO
- pkcs-helper

Estas son necesarias a fin de que el binario generado sea capaz de proveer el uso de la capa de seguridad SSL, la compresión lzo y el manejo de los certificados de seguridad y autenticaciones pkcs-helper.

Esta es la secuencia de comandos necesaria para generar los binarios nativos a partir del código fuente parchado.

./configure

make

make install

6) Una vez terminada exitosamente la compilación, tenemos instalado en nuestro sistema, los binarios o archivos ejecutables , dado el caso, con lo que podemos realizar nuestros propósitos.

Para levantar el servicio de la VPN con soporte para IPv6, utilizaremos la siguiente

```
openvpn --port 1194 --proto tcp --dev tun --ca /usr/share/doc/openvpn/examples/sample-keys/ca.crt --cert /usr/share/doc/openvpn/examples/sample-keys/server.crt --key /usr/share/doc/openvpn/examples/sample-keys/server.key --dh /usr/share/doc/openvpn/examples/sample-keys/dh1024.pem --server 10.8.0.0 255.255.255.0 --server-ipv6 2001:448:1:1::100/64 --ifconfig-pool-persist ipp.txt --keepalive 10 120 --comp-lzo --max-clients 10 -- persist-key --persist-tun --status /etc/openvpn/logs/openvpn-status.serverIPv6-patch-BS-GD-3.log --log-append /etc/openvpn/logs/openvpn.serverIPv6-pathc-BS-GD-2-2.log --verb 9
```

Cuadro 24. "Sentencia para ejecutar OpenVPN en un servidor con soporte para IPv6"

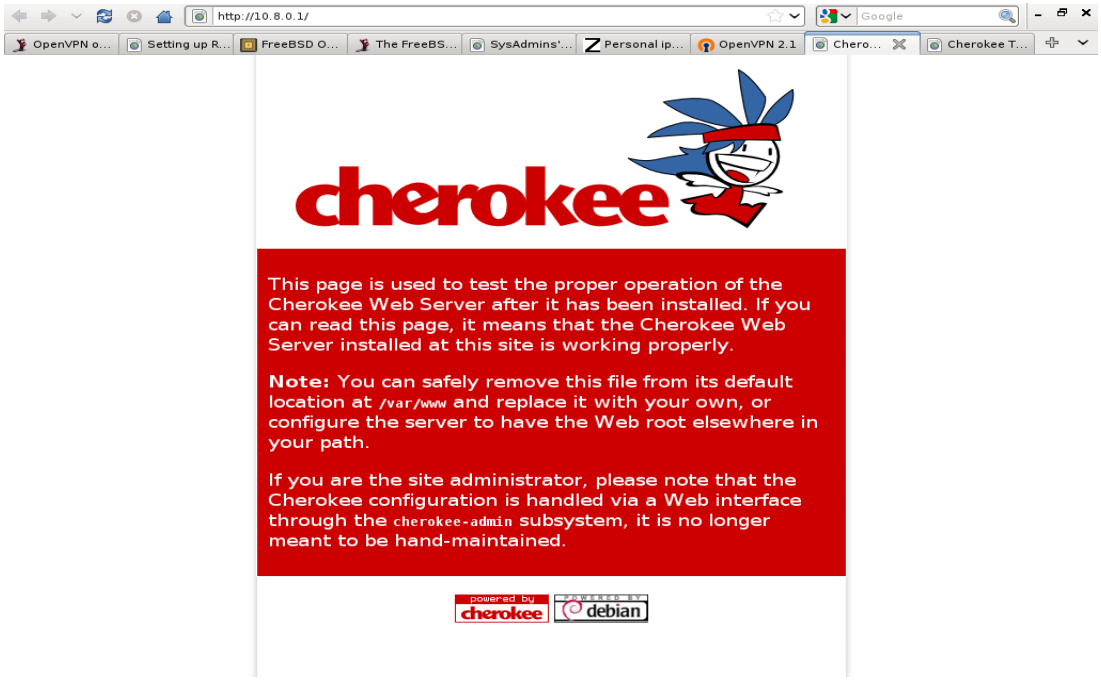

Figura 16 "Servidor Web Cherokee con dirección IPv4"

Con lo que se tiene un servidor VPN con soporte para IPv6. Verificamos que es capaz de transmitir tráfico con el servidor web Cherokee, tanto para IPv4 como para IPv6.

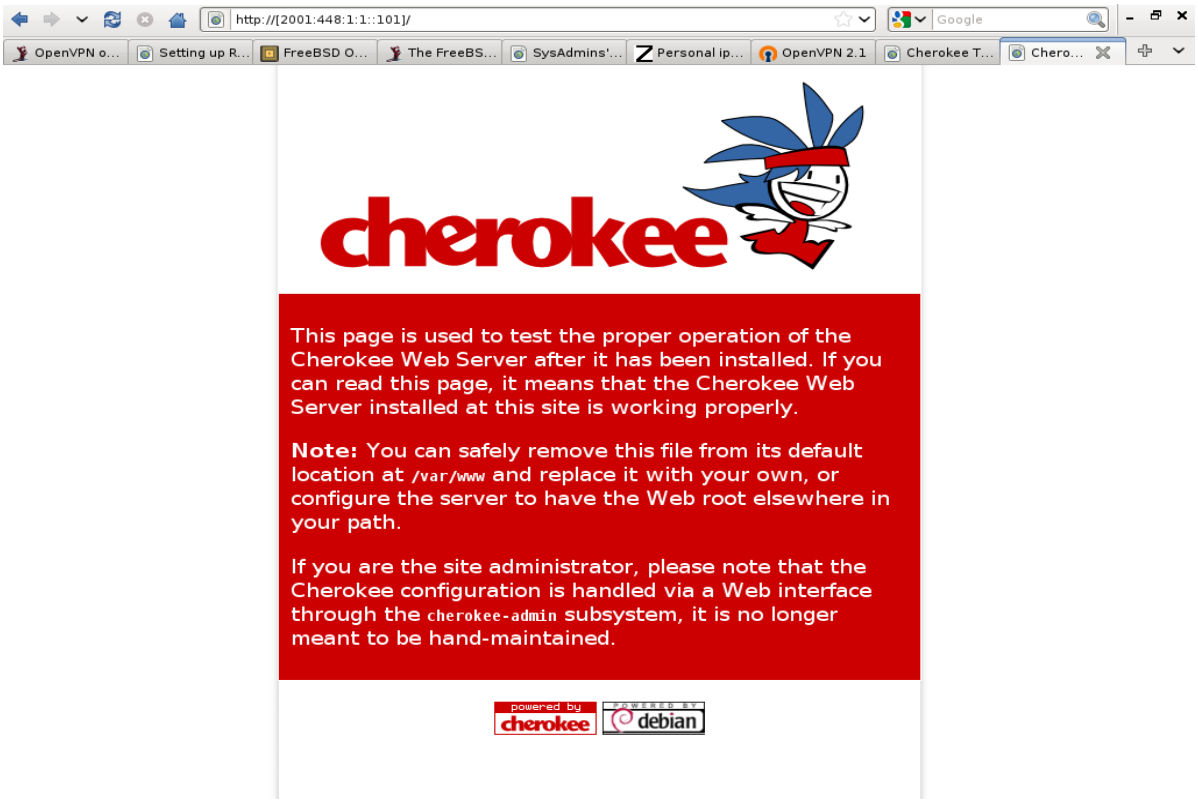

Figura 17 "Servidor Web Cherokee con dirección IPv6"

Se puede ver que este está funcionando en el sistema operativo FreeBSD.

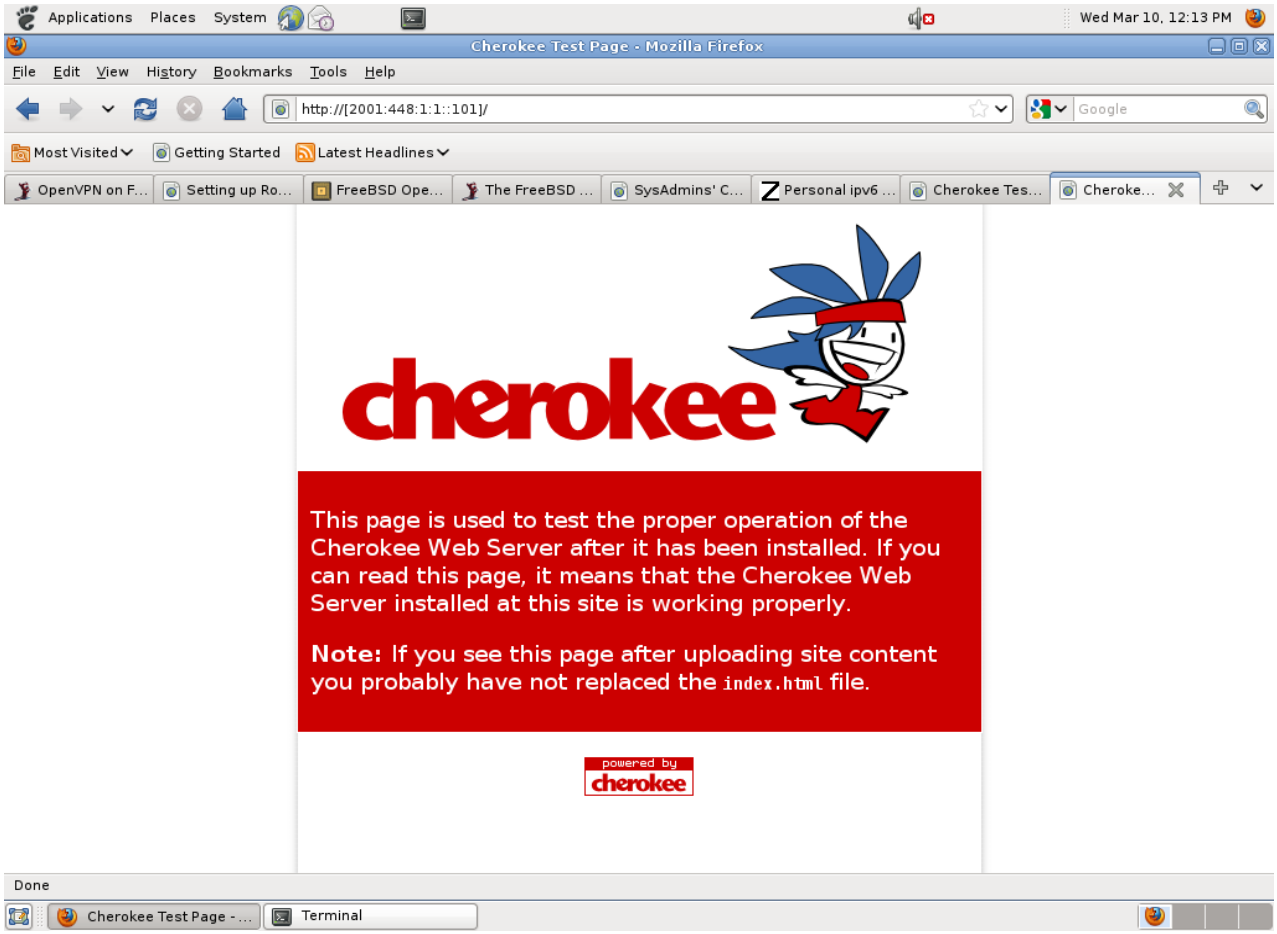

Figura 18 " Servidor Web Cherokee con dirección IPv6 en Fedora"

También funciono para el sistema Fedora Linux, por lo que se puede ver que este funciona en todos los sitemas Unix reales. No fue posible hacerlo funcionar en los sistemas MS Windows. En el caso de los clientes se utilizaron los siguientes archivos de configuración.

```
/usr/local/sbin/openvpn --client --dev tun0 --tun-ipv6 --proto tcp --remote 132.248.108.233 --port 1194 --resolv-retry infinit e --nobind --persist-tun --persist-key --ca /home/netlab/Downloads/openvpn-2.1.1-Berhard-Schmid-Gert-Doering/sample-keys/ca.cr t --cert /home/netlab/Downloads/openvpn-2.1.1-Berhard-Schmid-Gert-Doering/sample-keys/client1Debian.crt --key /home/netlab/Dow nloads/openvpn-2.1.1-Berhard-Schmid-Gert-Doering/sample-keys/client1Debian.key --comp-lzo --status /etc/openvpn/logs/openvpn-s tatus.clientIPv6-patch-BS-GD-deb5-2.log -log-append /etc/openvpn/logs/openvpn.clientIPv6-patch-BS-GD-deb5-2-1.log --verb 9
```

Cuadro 25 "Sentencia de ejecución del cliente Debian al conectarse al servidor Fedora VPN IPv6"

```
/usr/local/sbin/openvpn --client --dev tun0 --tun-ipv6 --proto tcp --remote 132.248.108.239
--port 1194 --resolv-retry infinit e --nobind --persist-tun --persist-key --ca
/home/netlab/Downloads/openvpn-2.1.1-Berhard-Schmid-Gert-Doering/sample-keys/ca.cr t
--cert /home/netlab/Downloads/openvpn-2.1.1-Berhard-Schmid-Gert-Doering/sample-
keys/client2Debian5.crt --key /home/netlab/Do wnloads/openvpn-2.1.1-Berhard-Schmid-
Gert-Doering/sample-keys/client2Debian5.key --comp-lzo --status /etc/openvpn/logs/openvpn
-status.clientIPv6-patch-BS-GD-deb5-FBSD-3.log
--log-append /etc/openvpn/logs/openvpn.clientIPv6-patch-BS-GD-deb5-FBSD-3-1.log
 --verb 9
```

Cuadro 26 "Sentencia de ejecución del cliente Debian al conectarse al servidor FreeBSD VPN IPv6"

14.Tablas de resultados

En resumen, de las pruebas realizadas entre los equipos, las comunicaciones que se realizaron entre ellos pueden verse en las siguientes tablas. La siguiente tabla muestra la conectividad entre los diversos sistemas que se probaron bajo el modelo cliente-cliente (punto a punto). Como se puede apreciar , únicamente el sistema Windows Vista fue incapaz de lograr la conectividad con los demás sistemas. En todas las versiones de Linux y en los BSD's , así como en Windows XP si existe la conexión.

Conectividad IPv4 VPN (modelo cliente-cliente)						
Sistemas Operativos	Windows XP SP3	FreeBSD 6/7.2	OpenBSD 4.2	Fedora 6/11	Debian 4/5	Windows Vista SP1
FreeBSD 6/7.2/8	Sí	----------	Sí	Sí	Sí	NO
OpenBSD 4.2	Sí	Sí	----------	Sí	Sí	NO
Fedora 6 /11	Sí	Sí	Sí	----------	Sí	NO
Debian 4/5	Sí	Sí	Sí	Sí	----------	NO
Windows XP SP3	----------	Sí	Sí	Sí	Sí	NO
Windows Vista SP1	NO	NO	NO	NO	NO	----------

Tabla 2."Resumen de los resultados de la conectividad con IPv4 en la VPN."

En la tabla que se obtuvo para el modelo cliente-servidor, se tiene que existe la conectividad en todas las versiones de Linux y en los BSD's, como servidores. En esta parte no es posible usar a los sistemas Windows como servidores dada la implementación que estos tienen para su interfaz de red. En cuanto la parte de los clientes, nuevamente todos los sistemas Linux y los BSD's, así como Windows XP SP3, son capaces de conectarse con los servidores, para la versión de Windows Vista esto no es posible.

Conectividad IPv4 VPN (modelo cliente-servidor)						
Cliente / Servidor	Windows XP SP3	Windows Vista SP1	FreeBSD 6/7.2/8	OpenBSD 4.X	Fedora 6/11	Debian 4/5
FreeBSD 6/7.2/8	Sí	NO	----------	Sí	Sí	Sí
OpenBSD 4.2	Sí	NO	Sí	----------	Sí	Sí
Fedora 6 /11	Sí	NO	Sí	Sí	---------	Sí
Debian 4/5	Sí	NO	Sí	Sí	Sí	----------

Tabla 3.“Resumen de los resultados de la conectividad con IPv4 en la VPN. Modelo cliente-servidor.”

En la siguiente tabla lo que muestran son los resultados de la conectividad que se lograron, al utilizar scripts de configuración en el lado del servidor a fin de que en los mismos se tuviera una conexión IPv6 . Todas las direcciones se configuraban manualmente tanto del lado del servidor como del cliente. En esta tabla se observa que los sistemas Windows, en sus versiones XP y Vista, además de OpenBSD no fueron capaces de soportar este tipo de configuración, en los primeros por la incapacidad de la plataforma para soportar estas implementaciones y en el ultimo por razones no específicas.

Conectividad IPv6 VPN (modelo cliente-servidor)(Direcciones Globales Manuales) OpenVPN v. X						
Cliente / Servidor	Windows XP SP3	Windows Vista SP1	FreeBSD 7.2	OpenBSD 4.5/4.6	Fedora 11	Debian 5
FreeBSD 7.2/8	Sin soporte	Sin soporte	----------	NO	Sí	Sí
OpenBSD 4.6	Sin soporte	Sin soporte	NO	----------	NO	NO
Fedora 11 Leonidas	Sin soporte	Sin soporte	Sí	Sí	----------	Sí
Debian 5 Lenny	Sin soporte	Sin soporte	Sí	Sí	Sí	----------

Tabla 4.“Resumen de los resultados de la conectividad con IPv6 en la VPN. Modelo cliente-servidor.”

En la siguiente tabla, se utilizó el parche que el desarrollador argentino Juan José Ciarlante ofrece para agregar capacidades IPv6 al software OpenVPN. Este parche ofrece la conectividad punto a punto y funcionó en los sistemas Linux (Debian y Fedora) y FreeBSD, para la parte de los sistemas Windows, esto no fue posible porque no esta bien soportado este desarrollo en Windows. En el sistema OpenBSD al intentar generar el paquete fallaba.

Conectividad IPv6 VPN (modelo cliente-servidor) con parche (Direcciones Automáticas)						
Cliente / Servidor	Windows Vista SP1	Windows XP SP3	FreeBSD 6/7.2/8	OpenBSD 4.6	Fedora 11	Debian 5
FreeBSD 8	No funcionó	No funcionó	----------	No funcionó	Sí	Sí
OpenBSD 4.6	No funcionó	No funcionó	No funcionó	----------	No funcionó	No funcionó
Fedora 11 Leonidas	No funcionó	No funcionó	Sí	No funcionó	----------	Sí
Debian 5 Lenny	No funcionó	No funcionó	Sí	No funcionó	Sí	----------

Tabla 5.“Resumen de los resultados de la conectividad con IPv6 en la VPN. Modelo cliente-servidor punto a punto. Compilado con el parche de Juan José Cirlante.”

Para la última tabla, es donde se pueden ver los resultados del trabajo de los desarrolladores alemanes Bernhard Schmidt y Gert Döring, quienes desarrollaron su propio parche para la conectividad IPv6. Dicho parche ofrece conectividad punto a multipunto con lo cual se alcanzo el objetivo buscado, el modelo cliente-servidor. Desafortunadamente tampoco funciona para los sistemas Windows, ni en OpenBSD. En los Linux (Debian o Fedora) y en FreeBSD no tiene ningún problema.

Conectividad IPv6 VPN con parche de Bernard Schmidt & Gert Döring(Direcciones Automáticas)						
Cliente / Servidor	Windows Vista SP1	Windows XP SP3	FreeBSD 8	OpenBSD 4.6	Fedora 11	Debian 5
FreeBSD 8	Sin soporte	Sin soporte	----------	No funcionó	Sí	Sí
OpenBSD 4.6	Sin soporte	Sin soporte	No funcionó	----------	No funcionó	No funcionó
Fedora 11 Leonidas	Sin soporte	Sin soporte	Sí	No funcionó	----------	sí
Debian 5 Lenny	Sin soporte	Sin soporte	Sí	No funcionó	Sí	----------

Tabla 6.“Resumen de los resultados de la conectividad con IPv6 en la VPN. Modelo cliente-servidor punto a multipunto. Compilado con el parche de Bernhard Shcmidt y Gert Döring.”

Glosario de términos

3DES (Triple **DES**):algoritmo de cifrado.

AES (**A**dvanced **E**ncription **S**tandard):Estandar de Cifrado Avanzado.

AH (**A**uthentication **H**eader):Encabezado de Autenticación ,IPSec.véase capítulo 2.

Blowfish algoritmo de cifrado

CHAP (**C**hallenge **H**andshake **A**uthentication **P**rotocol): Protocolo de Autenticación por Desafío Mutuo, método de autenticación remota o inalámbrica.

DES (**D**ata **E**ncription **S**tandar): algoritmo de cifrado.

DHCP (**D**ynamic **H**ost **C**onfiguration **P**rotocol):Protocolo de Configuración Dinámica de servidores, protocolo de red, que permite a los nodos obtener los parámetros de configuración automáticamente.

ESP (**E**ncapsulating **S**ecurity **P**ayload): Encabezado de Carga de Seguridad de Encapsulamiento,uno de los encabezados de cifrado para IPSec. véase capítulo 2.

GRE (**G**eneric **R**outing **E**ncapsulation): Encapsulamiento Genérico de Enrutamiento, es un protocolo de tuneleo desarrollado por Cisco que puede encapsular una amplia variedad de paquetes de protocolos de Capa de red dentro de túneles IP.

HMAC (**H**ashed **MAC**): algoritmo de autenticación.

IANA (**I**nternet **A**ssigned **N**umber **A**uthority):Agencia de Asignación de Números Internet. Fue el antiguo registro central de protocolos,puertos , números de protocolo y empresa asociada a ellos, opciones y códigos.

ICV (**I**ntegrity **C**heck **V**alue):IPSec, véase capítulo 2.

IDEA (**I**nternational **D**ata **E**ncryption **A**lgorithm):

IETF (**I**nternet **E**ngineering **T**ask **F**orce): Organización internacional que participa en el desarrollo de los estándares (protocolos , algoritmos, etc) de la Internet.

IKE (**I**nternet **K**ey **E**xchange): véase pág. X

IP (**I**nternet **P**rotocol): Protocolo de capa 3. Este protocolo es el de mayor uso e implementación en las redes existentes. Se le denomina IPv4 debido a la nomenclatura de la versión destinada a sustituirlo.

IPLS (**IP**-only **LAN S**ervices): redes VPLS simplificadas, como en aquellas , las interfaces LAN corren en modo promiscuo, y los frames se envían en base a sus direcciones físicas (**MAC**) de destino.

IPSec (**I**nternet **P**rotocol **Sec**urity): Protocolo de seguridad, *véase capítulo 2.*

ISAKMP (**I**nternet **S**ecurity **A**ssociation and **K**ey **M**anagement **P**rotocol): Protocolo de Manejo de Llaves y Asociación Segura de Internet.

ISO (**I**nternational **S**tandars **O**rganization): La Organización Internacional de Estándares, es una organización no gubernamental, encargada de producir normas internacionales con la finalidad de facilitar el intercambio de información y el comercio.

ISP (**I**nternet **S**ervice **P**rovider): Proveedor de Servicio de Internet.

LAN-2-LAN redes virtuales que conectan dos LANs.

L2F (**L**ayer **2 F**orwarding) : Protocolo de envío de datos a través de la Capa 2 el modelo OSI. Creado por CISCO para establecer túneles de tráfico desde los usuarios remotos hasta las oficinas corporativas.

L2TPv2 (**L**ayer **2 T**unneling **P**rotocol **v**ersion **2**): Protocolo de túneleo a través de Capa 2 del modelo OSI. Básicamente el protocolo L2TP cómo se definió en el RFC 2661.

L1VPN redes virtuales que se manejan a nivel de capa 1, generalmente mediante el uso de dispositivos físicos.

L2VPN redes virtuales privadas que se manejan a nivel de capa 2 , del modelo OSI.

L3VPN redes virtuales privadas que se manejan a nivel de capa 3 , del modelo OSI.

MAC (Media **A**cces **C**ontrol**)**:control de acceso al medio, identificador de 8 bits, o 6 bloques hexadecimales , que teóricamente corresponden de forma única a un dispositivo de red ethernet.

MD5 (**M**essage **D**igest version 5):algoritmo de autenticación.

MPLS (**M**ulti **P**rotocol **L**abel **S**witching):Conmutación multiprotocolo mediante Etiquetas, mecanismo de transporte de datos estándar creado por la IETF. Opera entre las capas de enlace de datos (capa 2)y la capa de red (capa 3) del modelo OSI.

NIC (**N**etwork **I**nterface **C**ard): tarjeta de red.

OSI (**O**pen **S**ystem **I**nterconneted): El modelo de Interconexión de Sistemas Abiertos fue propuesto por la ISO, describiendo como deberían conectarse las distintos equipos de computo y redes para poder interactuar entre sí.

PAP (**P**rotocol **A**uthentication **P**assword): Protocolo de autenticación de clave, autentica a un usuario contra un servidor de acceso.

PKI (**P**ublic **K**ey **I**nfrastructure): Infraestructura de Llave/clave Pública, combinación de hardware, software y políticas de seguridad que permiten la ejecución segura de operaciones de cifrado, firma digital , y no repudio de transacciones electrónicas.

PPoE (**P**oint to **P**oint **o**ver **E**thernet): Protocolo Punto a Punto sobre Ehernet, protocolo de red para la encapsulamiento PPP sobre una capa de Ethernet.

PPP (**P**oint to **P**oint **P**rotocol): Protocolo Punto a Punto, protocolo que se maneja a nivel de la Capa de Enlace estandarizado en el RFC 1661.

PPTP (**P**oint to **P**oint **T**unneling **P**rotocol): protocolo desarrollado por el PPTP Forum para implementar redes virtuales privadas.

RFC (**R**equest **F**or **C**omments): Documentos de especificaciones que se exponen públicamente para su discusión.

RIPEMD (**RACE I**ntegrity **P**rimitives **E**valuation **M**essage **D**igest):Algoritmo de Autenticación.

RADIUS (**R**emote **A**uthentication **D**ial **I**n **U**ser **S**ervice): es un protocolo de autenticación y autorización para aplicaciones de acceso a la red o movilidad IP.

SA (**S**ecurity **A**ssociations): Asociaciones de Seguridad de IPSec. véase pág. X

SHA-1(**S**ecure **H**ash **A**lgorithm):algoritmo de autenticación.

SONET (**S**ynchronous **O**ptical **N**etworks): estándar para el transporte de telecomunicaciones en redes de fibra óptica.

SP (**S**ecurity **P**olicies): Políticas de Seguridad de IPSec.

SPI (**S**ecurity **P**arameter **I**ndex): Índice de Parámetros de Seguridad, es el identificador de seguridad utilizado por IPSec.

SSL (**S**ecure **S**ocket **L**ayer): Protocolo de Capa de Conexión Segura, proporciona autenticación y privacidad de la información entre los extremos de una conexión a través de Internet mediante el uso de algoritmos de cifrado.

TCP (**T**ransmission **C**ontrol **P**rotocol): protocolo de transporte orientado a conexión, utilizado en internet para establecer comunicaciones confiables.

TTL (**T**ime **T**o **L**ive): campo de 8 bits dentro del encabezado IPv4.

UDP (**U**ser **D**atagram **P**rotocol):Protocolo de Datagrama a nivel de Usuario, es un protocolo de nivel de transporte basado en el intercambio de datagramas a través de la red sin necesidad de que se haya establecido con anterioridad una conexión.

VPLS (**V**irtual **P**rivate **LAN S**ervices):una manera de proveer comunicacionesss ethernet multipunto a multipunto sobre redes MPLS/IP.

VPN (**V**irtual **P**rivate **N**etwork):Red Privada Virtual, son aquellas redes que utilizan la infraestructura pública para crear una red virtual , que al cifrar el contenido se vuelve 'privada' con respecto al demás contenido.

VPWS (**V**irtual **P**rivate **W**ire **S**ervices):proveen conectividad punto a punto entre los sitios de los clientes, donde la red del proveedor de servicios emula un conjunto de cables entre los sitios de los clientes bajo un túnel MPLS.

VC-ATM (**V**irtual **C**ircuit- **A**synchronous **T**ransfer **M**ode): Circuito Virtual – Modo de Transferencia Asíncrona , sistema de comunicación que hace uso de más de un circuito real durante un periodo de tiempo, donde la conmutación es transparente para el usuario.

WAN (**W**ide **A**rea **N**etwork): uno de los tipos de red de computadoras que se extiende sobre un área geográfica amplia.

WiFi nombre comercial por el que se conoce al protocolo IEEE 802.11b de secuencia directa. Provee comunicaciones inalámbricas.