



# CLICK HERE!

# Sys Admin™

the Journal for UNIX and Linux  
systems administrators

# Sys Admin™

the Journal for UNIX and Linux  
systems administrators

Try our RSS feed [XML](#)

## DEPARTMENTS

- Operating Systems
- Server Management
- Networking
- Security
- Storage
- Databases
- Languages
- Your Career

## SYS ADMIN

- Issues
- Source Code
- Events
- Newsletters
- Whitepapers
- About Us
- Contact Us
- Editors
- Advertise

[Sys Admin Magazine](#) > [Archives](#) > [2000](#) > [0011](#)

[Print-Friendly Version](#)

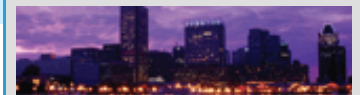
## Introducing FreeS/WAN and IPsec

*Duncan Napier*

FreeS/WAN (Secure Wide Area Network) is a tool that permits the secure transmission of data over untrusted networks, such as the Internet. The central component of FreeS/WAN is the IETF's (Internet Engineering Task Force) IPsec (Internet Protocol SEcurity) specification. Among other things, IPsec is designed to support Virtual Private Networks (VPNs). While FreeS/WAN is Linux-based, it conforms to the IETF's IPsec specification and is known to be interoperable with many other vendor's implementations of IPsec. (Refer to the official Web site, <http://www.freeswan.org>, for the latest news.) The FreeS/WAN project was started by John Gilmore, is maintained by numerous energetic individuals, and is freely available in source code under the GNU General Public License.

IPsec is an extension to the Internet Protocol (IP) that provides authentication and encryption at the OSI Reference Model transport layer. IPsec has been mandated into the IETF's specification of IP version 6 (IPv6). Three protocols are used to handle encryption and authentication: ESP (Encapsulating Security Payload); AH (Authentication Header); and IKE (the Internet Key Exchange). All of these components are included in the FreeS/WAN implementation of IPsec, and are generally invisible to the end user. The ESP and AH handle encryption and authentication, while IKE negotiates the connection parameters, including the initialization, handling, and renewal of encryption keys. The only encryption scheme currently supported by FreeS/WAN is 3DES (the triple DES or Data Encryption Standard, the current standard for IPsec encryption). Authentication is carried out using MD5 digests of a so-called shared secret (a shared key). The shared key could be a mutually agreed upon character string or RSA private keys. FreeS/WAN's KLIPS (kernel IPsec) component, which is compiled into the Linux kernel, implements AH, ESP, and the handling of packets. IKE processes are carried out through FreeS/WAN's standalone **pluto** daemon. The IETF RFCs and Internet drafts on these and related topics may be found at the official FreeS/WAN Web site. The site also contains full on-line manpages and documentation, as well as numerous links to the full IETF specifications, user-developer mailing lists, mailing list archives, and other IPsec-related sites.

FreeS/WAN is an implementation of the IPsec standard through its KLIPS and IKE components. Currently, FreeS/WAN provides support for IPsec over IP version 4 (IPv4), although at the time of writing, FreeS/WAN support for IPv6 was in progress.



## Sys Admin Technical Conference

May 7-8, 2007  
Sheraton Inner Harbor  
Baltimore, MD

Featuring tutorials  
by Randal Schwartz  
and Hal Pomeranz

Perl, Sendmail,  
Linux Security,  
Network Incident  
Response and  
Much More!!!

## REGISTER TODAY!

## Current Issue



[Table of contents](#)

[Buy this issue.](#)

**SUBSCRIBE NOW!**

## Newsletter

Subscribe to the *Sys Admin* update email newsletter...

Email Address

First Name

Last Name

FreeS/WAN is meant to be a fully compliant IPsec implementation that imposes a minimum of overhead and will run even on low-end PCs that use the Linux OS. The overhead imposed and measured performance will depend on the configuration of the hardware platform as well as the network bandwidth that is being served. I recently set up a network of FreeS/WAN firewalling gateways (RedHat 6.1, standard 2.2.12 kernel, FreeS/WAN 1.4 running 3DES encryption/MD5 authentication, using IPchains 1.3.9 for firewalling) interconnected through a 512-kb/s DSL (digital subscriber line) WAN (wide-area network). The hardware platforms were off-the-shelf Dell Dimension 133-MHz Pentium machines with 32 MB of RAM; the only modification was the installation of 2 Intel Ether Express 10/100 network cards on each machine. The uncompressed sustained transfer rate for a single session in this particular case was measured to be about 450-kb/s or about 90% the theoretical maximum. The file transfer showed virtually no effect on either CPU or memory loading. Some performance numbers for FreeS/WAN and other VPN-capable products are given in

<http://www.epm.ornl.gov/~dunigan/vpnperf.html>

FreeS/WAN is written and tested around RedHat's implementation of the Linux operating system, but FreeS/WAN is known, with no or slight modification, to work with SuSE, Slackware, and Debian distributions as well. FreeS/WAN has also been reported to work on Linux systems that run non-Intel CPUs, such as the DEC 64-bit Alpha, Motorola Power PC, Sun's SPARC, MIPS, and StrongARM. FreeS/WAN interoperates to varying degrees with the IPsec implementations of other vendors, including Cisco IOS 12.0 and later, Open BSD, Raptor Firewall, CheckPoint FW-1, F-Secure VPN, Xedia Access Point, PGP 6.5/PGPnet, IRE Safenet/SoftPK, Freegate 1.3, Borderware 6.0, Timestep Permit/Gate 2520, Sun Solaris, Intel/Shiva LANrover, and Windows 2000. The official FreeS/WAN Web site has a regularly updated compatibility list with the latest version of its online documentation.

The following details the steps required to build a FreeS/WAN VPN-firewall gateway from scratch. The steps to configuring a site-to-site VPN with firewalls on both ends are also included. Numerous other variants, such as point-to-site, site-to-multisite, and point-to-point connections are also possible with slight modifications, but will not be covered here.

### Installation

Due to the regulation of cryptographic tools and applications in some countries, FreeS/WAN is generally not included in the compiled Linux kernel of most Linux distributions. SuSe Linux version 6.3 has been reported to have shipped in Europe with FreeS/WAN compiled in the kernel. FreeS/WAN versions 1.1 and later are known to compile with the Linux 2.0.x and 2.2.x kernels. Refer to the FreeS/WAN Web site for information on the Linux 2.3 kernel, or later versions. The following example was compiled on a standard RedHat 6.1 distribution.

FreeS/WAN requires the Linux kernel to be recompiled. The kernel source usually resides in `/usr/src/linux`. If you need a more recent version of the kernel, download the kernel source and install in `/usr/src/linux`.

### Recompiling the Kernel with FreeS/WAN

1. Download the FreeS/WAN source code and digital signatures from:

<http://www.freeswan.org>

or:

**Sys Admin**  
the journal for UNIX and Linux  
systems administrators

**CLICK  
HERE!**

## CD-ROM



*Sys Admin* and *The Perl Journal* CD-ROM version 12.0

Version 12.0 delivers every issue of *Sys Admin* from 1992 through 2006 and every issue of *The Perl Journal* from 1996-2002 in one convenient CD-ROM!

[Order now!](#)

<ftp://ftp.xs4all.nl/pub/crypto/freeswan>

2. Move the **tar.gz** file to: **/usr/local/src/**

3. Go to: **/usr/local/src/**:

```
cd /usr/local/src/
```

4. **Untar** the **tarfile**, set permissions, and go into the FreeS/WAN source directory:

```
cat /usr/local/src/freeswan-x.x.tar.gz | gunzip | tar xvf - \
chown -R root:root freeswan-x.x
```

5. Do a test configure and compile of the kernel (usually found in **/usr/src/linux**) to ensure a functioning kernel. Go into the kernel source root:

```
cd /usr/src/linux
make config
make dep
make bzimage
```

You may wish to load and boot off the kernel as a test, using LILO (refer to section 8).

6. Run the **make** utility that sets the kernel compilation option using a menu interface. (For other options, see FreeS/WAN documentation):

```
cd /usr/local/src/freeswan-x.x
make menugo
```

This particular choice will run the curses-based menu-driven kernel configurator. In nearly all cases, the default options will suffice. Make sure that you choose the option to save the kernel configurations. Go into the Linux source directory and run **bzImage**:

```
cd /usr/src/linux
make bzImage
```

7. Copy the resulting boot image to the **/boot** partition:

```
cp /usr/src/linux/arch/i386/boot/bzImage /boot/vmlinuz-ipsec
```

8. Edit the boot loader config file **/etc/lilo**:

```
boot=/dev/hda
map=/boot/map
install=/boot/boot.b
```

```
prompt
timeout=100
image=/boot/vmlinuz-ipsec
label=linux-ipsec
root=/dev/hda1
read-only
image=/boot/vmlinuz-2.x.x
label=linux
root=/dev/hda1
read-only
```

Rerun LILO and you should see:

```
linux-ipsec *
linux
```

9. Reboot the machine. IPsec starts automatically from init.d scripts, which are installed by default during the FreeS/WAN installation. Once the machine has restarted, FreeS/WAN will need to be configured.

### Configuring TCP/IP for FreeS/WAN

The machine running FreeS/WAN functions as a gateway and will require IP forwarding to be enabled between the interfaces. To do this in Linux, change the parameter in `/etc/sysconfig/network` from:

```
FORWARD_IPV4=false
```

to:

```
FORWARD_IPV4=true
```

### Configuring IPsec with FreeS/WAN

For the following discussion, assume the network setup shown in [Figure 1](#) is a site-to-site VPN. The file `/etc/ipsec.conf` will need to be customized. The other parameters (e.g., in the defaults section) can generally be left as is. Refer to the man page for `ipsec.conf` for full details on the syntax for `ipsec.conf` files. On server 1.2.3.4, the following network entries are made ([Listing 1](#)). On server 5.6.7.8, the converse applies, and left and right are interchanged ([Listing 2](#)).

Replace the `espenckey` and `espauthkey` with keys generated from `/usr/local/bin/ranbits`, the random key generator that is installed with FreeS/WAN.

To reset or restart FreeS/WAN using manual keying, type:

```
/etc/rc.d/init.d/ipsec restart
/usr/local/bin/ipsec manual -up conn_name
```

where **conn\_name** is the interface name specified by the **conn** header in **/etc/ipsec.conf**.

Check that the tunnel is set up by typing:

```
/usr/local/sbin/ipsec eroute
```

to display the extended routes on your system. For the above example on the right subnet, expect to see something like:

```
192.168.1.0/24      -> 192.168.0.0/24      => tun0x10e@5.6.7.8
```

which shows the tunnel between the subnets and the remote (left subnet) gateway. Once the above setup is complete, you should be able to **ping** any node on the subnet at the other end of the VPN. Note that you cannot access remote private subnets from any point outside these subnets, including from your own IPsec gateway.

The above setup is for a manually keyed system, which is less secure than an autoconnection keyed system. It should only be used for testing purposes, since it does not use the key management and exchange systems that FreeS/WAN uses for secure cryptographic exchange.

### FreeS/WAN and Firewalls

If you are running a firewall, you will have to allow access through the firewall. An example of such a setup would comprise a tunnel between two firewalls. FreeS/WAN runs well on Linux firewalls with the standard distribution of **ipchains**. Note that running a FreeS/WAN IPsec tunnel through any device with NAT (Network Address Translation) or IP masquerading will have unpredictable results and is not recommended (refer to documentation, <http://www.freeswan.org>). If the IPsec gateway is not the platform that performs NAT/masquerading, then it is strongly recommended that the IPsec gateway lies in front of the NAT/masquerading platform. A sample firewall script containing filtering rules for FreeS/WAN is available from the FreeS/WAN Web site. In addition to many of the familiar rules for **ipchains**, the following directives are required to enable the passage of data packets across the WAN.

On Server 1.2.3.4, the following firewalling rules should be added:

```
ipchains -A forward -p all -j ACCEPT -s 192.168.0.0/24 \
-d 192.168.1.0/24
ipchains -A forward -p all -j ACCEPT -s 192.168.1.0/24 \
-d 192.168.0.0/24
```

These rules must appear before the masquerading rule. The masquerading rules will look like this:

```
#

# FORWARD RULES
#
ipchains -P forward DENY
#
```

```
ipchains -A forward -p all -j ACCEPT -s 192.168.0.0/24 \
-d 192.168.1.0/24
ipchains -A forward -p all -j ACCEPT -s 192.168.1.0/24 \
-d 192.168.0.0/24
ipchains -A forward -p all -j MASQ -s 192.168.0.0/24 \
-d 0.0.0.0/0
```

On server 5.6.7.8, the converse is done:

```
ipchains -A forward -p all -j ACCEPT -s 192.168.1.0/24 \
-d 192.168.0.0/24
ipchains -A forward -p all -j ACCEPT -s 192.168.0.0/24 \
-d 192.168.1.0/24
```

Again, these rules appear before the masquerading rule, and it should look like this:

```
#
# FORWARD RULES
#
ipchains -P forward DENY
#
ipchains -A forward -p all -j ACCEPT -s 192.168.1.0/24 \
-d 192.168.0.0/24
ipchains -A forward -p all -j ACCEPT -s 192.168.0.0/24
-d 192.168.1.0/24
ipchains -A forward -p all -j MASQ -s 192.168.1.0/24 \
-d 0.0.0.0/0
```

Once the above manually keyed solution works (try **pinging** between subnets as a test), an autokeyed solution that uses the IKE (Integrated Key Exchange mediated via the **pluto** daemon) can be tested.

### Autokeyed Connection

In order to activate the autokeying, a shared secret is placed in the **/etc/ipsec.secrets** file. This file should be readable only to the root user. The shared secret can be any difficult-to-guess string that is identical in the **ipsec.secrets** file on each machine. For the above example, left network, it has the following format:

```
192.168.0.253 192.168.1.253: PSK \
"jx2V2S5grnATGH86gerEpi6Vj1lRTmkuTmS1TRSnuR
1R525WTuTn321m4luVRRjWWU2uR2U3VnkU"
```

where two IP addresses correspond to the ends of the tunnel, separated by a colon, PSK, and a string enclosed in quotes. An alternative is to use RSA private keys (refer to the man page for **ipsec.secrets**). The following shows the relevant portion of the **/etc/ipsec.conf** file for the example network, which is identical to the manual key example above, except with manual keying parameters removed:

```
conn left-test
type=tunnel
left=1.2.3.4
leftnexthop=1.2.3.5
leftsubnet=192.168.0.0/24
right=5.6.7.8
rightnexthop=5.6.7.9
rightsubnet=192.168.1.0/24
keyexchange=ike
keylife=8h
keyingtries=0
```

To automatically restart, replace:

```
auto=add
```

with:

```
auto=start
```

To restart IPsec, run the startup script:

```
/etc/rc.d/init.d/ipsec restart
```

The use of **auto=start** will automatically bring up the IPsec interface. If you use a firewall, you will need to allow access through the following ports: 500 (UDP), 50, and 51.

For IKE (Internet Key Exchange), carried out by the pluto daemon, which handles the negotiation of connection parameters, acceptable algorithms, key sizes, key, and so forth, requires a the UDP protocol connection on port 500. The resulting firewall script would be:

```
ipcha ins -A input -p udp -j ACCEPT -s 0.0.0.0/0 -i eth0 -d $IPSECGW 500
```

where **\$IPSECGW** is the Internet IP address of the a remote gateway requesting a connection. For added security, the source IP should be stated explicitly.

ESP (Encapsulated Security Payload) provides authentication and encryption of the payload contents and requires port 50 to be opened:

```
ipchains -A input -j ACCEPT -i eth0 -p 50 -s $IPSECGW
```

The authentication header (AH) requires access through port 51:

```
ipchains -A input -j ACCEPT -i eth0 -p 51 -s $IPSECGW
```

Troubleshooting

There are several troubleshooting tools available that will give detailed debugging information. Two useful tools are **ipsec barf** and **ipsec look** (refer to manpages **ipsec\_barf** and **ipsec\_look**). The **barf** utility collects, formats, and labels the system configuration and logfile information, which it then dumps to standard output as ASCII text. It is primarily intended as a comprehensive remote diagnostic tool that dumps output to a file that the troubleshooter can then analyze for errors. The output of the **look** utility is a pared-down subset of the **barf** information.

Another useful practice is to turn on debugging by setting the parameters assigned to **klipsdebug** and **plutodebug** in **/etc/ipsec.conf** config setup:

```
klipsdebug=all  
plutodebug=all
```

and restart FreeS/WAN. There is also a mailing list for reporting problems. Refer to the FreeS/WAN documentation.

### Other Configurations

There are numerous themes and variations that you can apply to the above configuration. Point-to-site, site-to-multisite, and point-to-point configurations are some of the other configuration possibilities. Extruded subnet configurations allow the loaning of IP addresses at remote sites over private lines. A common configuration is the so-called "Road Warrior" configuration, which allows remote users with dynamically assigned IP addresses and an IPsec client to tunnel into a private network. FreeS/WAN can work in environments that use dynamically assigned IP addresses. Refer to the FreeS/WAN documentation on these topics, as well as links to discussions of other example configurations.

### Conclusion

With the increasing penetration of broadband technologies into the ISP market, it is now practical to use the Internet to connect geographically dispersed clusters of private networks. It is apparent that VPN products will greatly enhance the safe and secure interconnectivity of networked devices, whether they are used for communication, monitoring, entertainment, or other purposes. Given the impending explosion of network-capable consumer electronics, FreeS/WAN provides an inexpensive option to secure networking for our everyday world.

### Sources

1. The official FreeS/WAN Web site and references cited therein:

<http://www.freeswan.org>

2. Linux FreeS/WAN manpages and documentation.

3. Kurt Seifried's *Linux Administrator's Guide*, specifically the section on IPsec:

<http://www.securityportal.com/lasg/ipsec>

*Duncan Napier is an avid mountain biker. When not out riding trails, he may be found running Napier Systems Research, an IT networking consultancy based in North Vancouver, B.C., Canada. He can be*

contacted at: [napier@computer.org](mailto:napier@computer.org).



**CLICK  
HERE!**

**Sys  
Admin.**  
the Journal for UNIX and Linux  
systems administrators

## MarketPlace

### [We BUY Technology Hardware!](#)

We'll buy your Technology Hardware even if you don't buy ours. Get an offer today!

### [We SELL Technology Hardware!](#)

Let Millenium help with your IT Hardware planning and procurement. Get a Quote Today!

### [Timesheet + time tracking for payroll and projects](#)

Clockware is the first timesheet and time tracking software that is 100% J2EE-compliant. Clockware's Payroll Timesheet integrates with all major Payroll systems. Clockware also supports Time and Attendance, and Project Timesheets in one system.

### [Cross Platform CORBA Tool](#)

Develop distributed systems conforming to open standards like CORBA and Web Services faster with SANKHYA Varadhi - The Digital Bridge.

### [Wanna see your ad here?](#)

Copyright © 2007 CMP Media LLC, [Privacy Policy](#), [Your California Privacy rights](#)

Comments about the Web site: [webmaster@sysadminmag.com](mailto:webmaster@sysadminmag.com)

SDMG Web Sites: [BYTE.com](#), [dotnetjunkies.com](#), [Dr. Dobb's Journal](#), [MSDN Magazine](#), [SD Expo](#), [Sys Admin](#), [sqljunkies.com](#), [UnixReview.com](#),