

Linux as a VPN Client to FireWall-1

Authored By: Ron Naken
Updated By: Elie Bitton
Date: Oct 10, 2000
Purpose: Describe a configuration allowing Linux hosts to establish a VPN tunnel to Check Point VPN-1 using IKE and IPSec.
Version: 4.1.102

If you have not done so already, you will need to install a copy of FreeS/WAN on your Linux host. This product will allow Linux to support IKE and IPSec, and can be easily configured to interoperate with Check Point VPN-1. To obtain a copy of FreeS/WAN, please refer to the following URL:

<http://www.freeswan.org>

For information on installing and compiling S/WAN for your system, please reference the above URL. This document will only address the necessary configuration to allow interoperability between S/WAN and VPN-1.

The following products were tested during the creation of this document:

Check Point VPN-1 **4.1 SP1 & SP2** on Redhat 6.2 (Kernel 2.2.14-5.0)
FreeS/WAN 1.5 (Swan) on Red Hat Linux, 6.2 (Kernel 2.2.14-5.0)

The following issues were discovered during the creation of this document:

S/WAN does not support *Aggressive Mode*.
S/WAN does not support an IKE SA renegotiation time above 480 minutes.
S/WAN does not interoperate when *Perfect Forward Secrecy* is enabled.

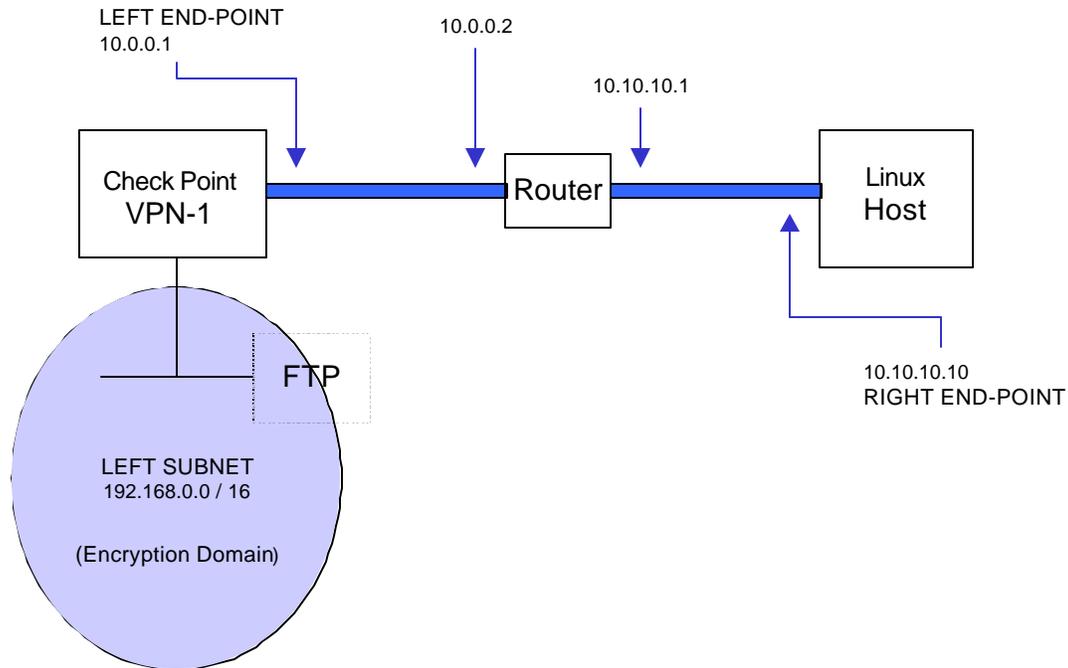
Configuration Checklist

This document assumes you have experience defining virtual private networks with Check Point products. A checklist is provided to ensure the few steps to creating this VPN are completed, and a more detailed discussion of the configuration is covered in later sections. Please note that this document does not cover the Check Point side of this configuration in great detail, since this information can be found in the *Virtual Private Networking User Guide for Firewall-1 / VPN-1*.

- ❑ Create a workstation object in the VPN-1 security policy to represent the Linux host.
 - Use IKE with shared-secrets and enter a shared-secret.
 - Disable *Aggressive Mode*!
 - Support keys exchange for subnets should be on.
- ❑ Create an 'Encrypt' rule to define the VPN.
- ❑ Adjust Policy→Properties: Renegotiate IKE Security Associations to a number less than 480 minutes – this is the maximum S/WAN can support.
- ❑ Configure /etc/ipsec.conf and /etc/ipsec.secrets on the Linux host to define the VPN.

Configuring the VPN: Sample Diagram

The following diagram depicts the configuration used during the creation of this document:



Before You Start:

Before installing FreeS/WAN or VPN-1 make sure that you can communicate to the all the machines in your network and that all routing has been setup.

Configuring the VPN: Linux Host

Once S/WAN is installed, we need to understand some of the basic configuration options available. These are configured in the following two files:

FOR FREES/WAN 1.5:

`/etc/ipsec.conf:`

```
#
# basic configuration
# In my case the interface used is eth0, if yours is different, substitute the proper interface
name
config setup
    interfaces="ipsec0=eth0"
    klipsdebug=none
    plutodebug=none
    manualstart=
    pluto load=
    pluto start=
```

```
# connection configuration
# This tunnel is required if you want the FreeS/WAN host to be able to communicate with
#the Check Point FW/VPN machine. i.e. If you want to be able to ping the CP VPN
#gateway from the FreeS/WAN host, then define the following:
```

```
conn linux-fw1
    type=tunnel
    left=10.0.0.1
    leftnexthop=10.0.0.2
    right=10.10.10.10
    rightnexthop=10.10.10.1
    keyexchange=ike
    auth=esp
    pfs=no
```

```
#This tunnel is necessary in order to allow the FreeS/WAN host to communicate with the
#Encryption Domain behind the CP VPN Gateway.
```

```
conn linux-encdom
    type=tunnel
    left=10.0.0.1
    leftnexthop=10.0.0.2
    leftsubnet=192.168.0.0/16
    right=10.10.10.10
    rightnexthop=10.10.10.1
    keyexchange=ike
    auth=esp
    pfs=no
```

The `/etc/ipsec.conf` file allows us to specify the configuration of our interfaces, as well as the VPN connections we will support. There are two sections we need to define:

1. config setup

This section allows us to attach virtual IPsec interfaces to our physical network interfaces. In the sample above, `ipsec0` is attached to the `eth0` interface. If you require autostart of the VPN during the boot process, please see “`man ipsec.conf`” for information on “`plutoload`” and “`plutostart`”.

2. conn <connection name>

This section must be defined for each VPN connection and holds the configuration of VPN options and routes to be added during the establishment of the VPN. In the sample above, a connection named ‘`linux-fw1`’ is defined and holds the following properties:

```
type=tunnel
    This option specifies to use TUNNEL mode for the VPN, instead of
    TRANSPORT mode. VPN-1 does not support transport mode at this
    time.
left=10.0.0.1
    This is the left end-point or side of the VPN. In the sample above, it is
    used to represent the VPN-1 gateway.
leftsubnet=192.168.0.0/16
```

This defines the topology of the left side of the VPN. Since we are using the left side to represent VPN-1, this should match (in whole or part) the encryption domain defined

right=10.10.10.10

This is the right end-point or side of the VPN. In the sample above, it is used to represent the Linux host.

rightnexthop=10.10.10.1

This represents the IP address of the gateway the right side of our VPN will use to get to the left side.

keyexchange=ike

Use IKE to exchange keys.

auth=esp

Enable ESP headers for packets.

pfs=no

Disable *Perfect Forward Secrecy*.

/etc/ipsec.secrets:

```
# Note that all secrets must now be enclosed in quotes, even if they have  
# no white space inside them.
```

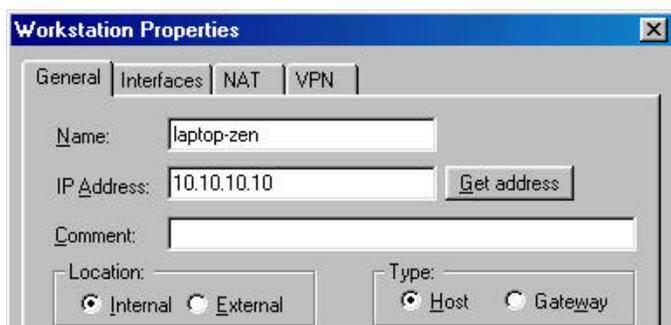
```
10.10.10.10      10.0.0.1      "password"
```

The /etc/ipsec.secrets file allows us to specify shared secrets between the hosts in our VPN. In the sample above, the shared secret for the 10.10.10.10 → 10.0.0.1 VPN is 'password'. This secret must match on both sides of the VPN, so be sure to configure the same shared secret on VPN-1.

Configuring the VPN: VPN-1

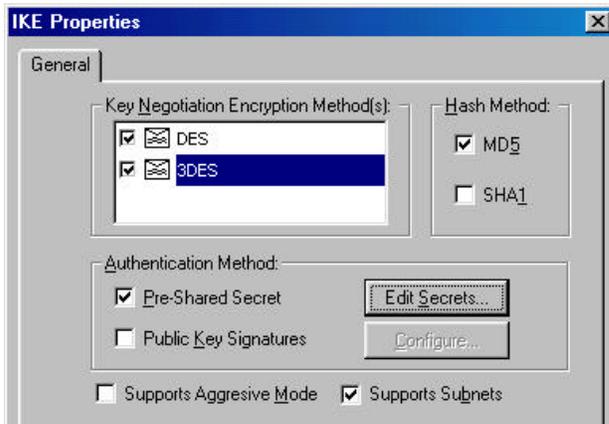
This section will discuss minimal configuration information on VPN-1. Further information can be found in the *Virtual Private Networking User Guide* for Firewall-1 / VPN-1.

Configure an object to represent the Linux host in your security policy. In our sample, it is a workstation object of type 'Host'.



If there are resources behind the Linux host that will be accessed through the VPN, be sure to select 'Gateway' and define an encryption domain for the Linux host. You will also need to define 'RightSubnet' properties in /etc/ipsec.conf.

Use the VPN page of the workstation properties to select IKE, and define an IKE shared-secret to use when communicating with this host – make sure this matches the shared-secret you create in `/etc/ipsec.secrets`:

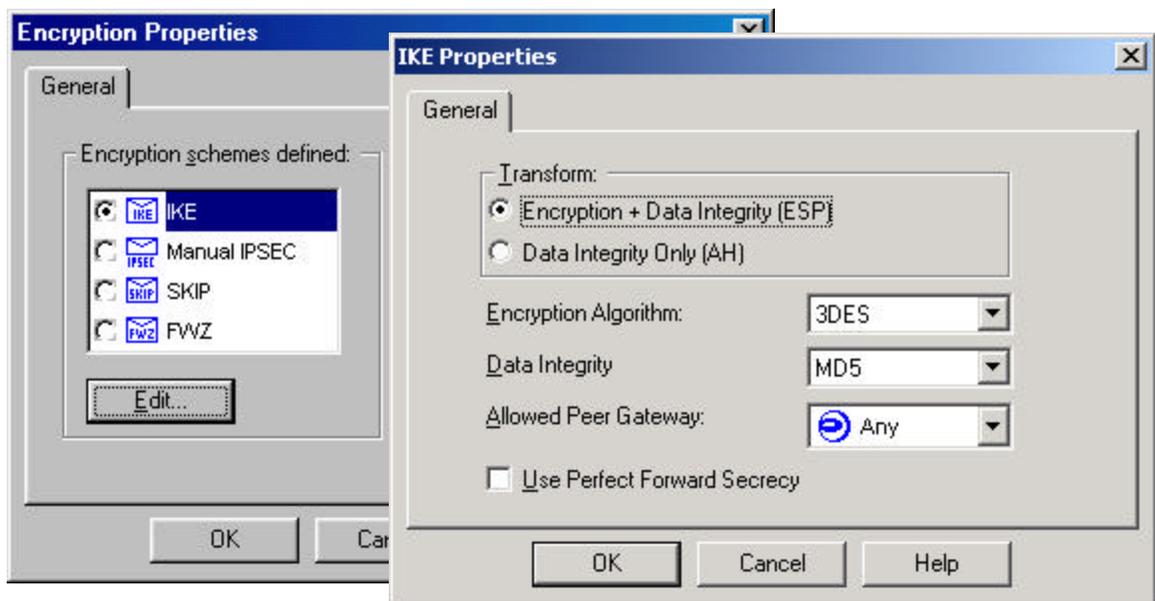


In the sample above, we selected “Pre-Shared Secret” for authentication. Select “Edit Secrets” and enter the secret you will use in `/etc/ipsec.secrets` on the Linux host. Please note that we disabled “Supports Aggressive Mode”.

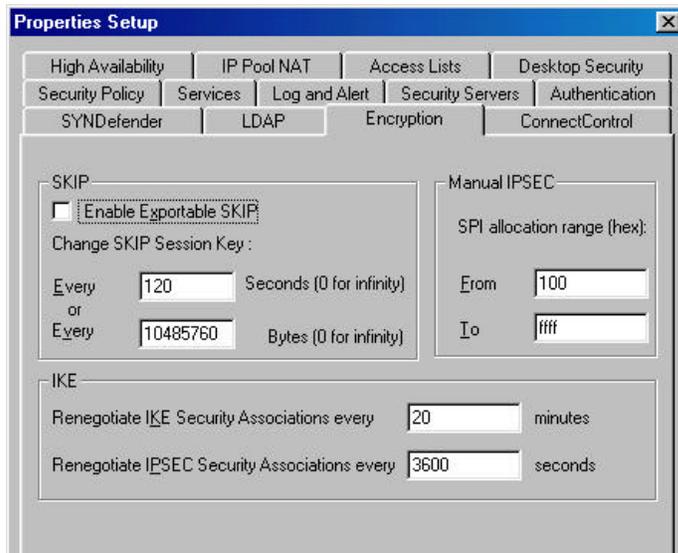
Create an “Encrypt” rule to define the VPN; this should look similar to the following:



It is a good idea to disable *Perfect Forward Secrecy* when interoperating with this version of S/WAN. Do this by right-clicking “Encrypt” in your rule. The option is found on the IKE properties dialog:



Since this version of S/WAN does not support an IKE SA renegotiation time greater than 480 minutes, you must change the default for VPN-1. Do this by selecting Policy->Properties:



Establishing the VPN

After building the proper S/WAN configuration files and configuring VPN-1, it is quite easy to establish a VPN between S/WAN and VPN-1. There are two configurations we will address with steps performed on the Linux host:

- **S/WAN will establish the connection to VPN-1**

If you will originate the VPN from the Linux host to VPN-1, you can use the following two steps to establish the VPN tunnel and enable encrypted communications:

1. Add our connection to the connections table:

Note: If you have made changes to /etc/ipsec.conf or /etc/ipsec.secrets since the last time you added or brought up a connection, it is best to stop FreeS/WAN and then start it again before you proceed.

To do this, use the commands:

```
Ipsec setup --stop  
Ipsec setup --start
```

Use the following format:

```
ipsec auto --config <file> --add <connection>
```

The default config file is /etc/ipsec.conf, so our commands were typed as follows:

```
ipsec auto --add linux-fw1
and ipsec auto --add linux-encdom
```

Below is a dialog of my session.

```
# ipsec setup --stop
ipsec_setup: Stopping FreeS/WAN IPSEC...
# ipsec setup --start
ipsec_setup: Starting FreeS/WAN IPSEC 1.5...
# ipsec auto --add linux-fw1
# ipsec auto --add linux-encdom
```

2. Bring the connection up and establish the VPN:

Use the following format:

```
ipsec auto --config <file> --up <connection>
```

The default config file is /etc/ipsec.conf, so our command was typed as follows:

```
ipsec auto --up linux-fw1
and ipsec auto --up linux-encdom
```

Below is a dialog of my session:

```
# ipsec auto --up linux-fw1
102 "linux-fw1" #1: STATE_MAIN_I1: initiate
104 "linux-fw1" #1: STATE_MAIN_I2: from STATE_MAIN_I1; sent MI2, expecting MR2
106 "linux-fw1" #1: STATE_MAIN_I3: from STATE_MAIN_I2; sent MI3, expecting MR3
004 "linux-fw1" #1: STATE_MAIN_I4: ISAKMP SA established
110 "linux-fw1" #2: STATE_QUICK_I1: initiate
004 "linux-fw1" #2: STATE_QUICK_I2: sent QI2, IPsec SA established

# ipsec auto --up linux-encdom
110 "linux-encdom" #3: STATE_QUICK_I1: initiate
004 "linux-encdom" #3: STATE_QUICK_I2: sent QI2, IPsec SA established
```

The VPN should now be established. For information on how to check or troubleshoot the VPN, please see *Troubleshooting and Debugging* in the following sections.

- **VPN-1 will establish the connection to S/WAN**

If you will originate the VPN from VPN-1 to the Linux S/WAN client, you can use the following commands to allow S/WAN to listen for incoming IKE & IPsec connections:

1. Add our connection to the connections table:

Use the following format:

```
ipsec auto --config <file> --add <connection>
```

The default config file is /etc/ipsec.conf, so our command was typed as follows:

```
ipsec auto --add linux-fw1
and ipsec auto --add linux-encdom
```

2. Enable listen mode to allow incoming IKE and IPsec requests:

```
ipsec whack -listen
```

You can now initiate connections from the Encryption Domain behind the VPN-1 box to the FreeS/WAN box.

Verifying the Connection

Once you have established a VPN with S/WAN, you can confirm the connection with the following command:

```
ipsec auto --status
```

The output should look similar to the following (if you have two tunnels established like we have in this example):

```
# ipsec auto --status
000 "linux-encdom": 10.10.10.10---10.10.10.1...10.0.0.2---10.0.0.1
000 "linux-encdom": ike_life: 3600s; ipsec_life: 28800s; rekey_margin: 540s; rekey_fuzz: 100%; keyingtries: 3
000 "linux-encdom": policy: POLICY_PSK+POLICY_ENCRYPT+POLICY_TUNNEL; interface: eth0; routed
000 "linux-encdom": newest ISAKMP SA: #0; newest IPsec SA: #3; eroute owner: #3
000 "linux-fw1": 10.10.10.10---10.10.10.1...10.0.0.2---10.0.0.1===192.168.0.0/16
000 "linux-fw1": ike_life: 3600s; ipsec_life: 28800s; rekey_margin: 540s; rekey_fuzz: 100%; keyingtries: 3
000 "linux-fw1": policy: POLICY_PSK+POLICY_ENCRYPT+POLICY_TUNNEL; interface: eth0; routed
000 "linux-fw1": newest ISAKMP SA: #1; newest IPsec SA: #2; eroute owner: #2
000
000 #3: "linux-encdom" STATE_QUICK_I2 (sent QI2, IPsec SA established); EVENT_SA_REPLACE in 27775s; newest
IPSEC; eroute owner
000 #3: "linux-encdom" esp0x8f253811@10.0.0.1 esp0xf34ac697@10.10.10.10 tun0x104@10.0.0.1 tun0x103@10.10.10.10
000 #2: "linux-fw1" STATE_QUICK_I2 (sent QI2, IPsec SA established); EVENT_SA_REPLACE in 27971s; newest IPSEC;
eroute owner
000 #2: "linux-fw1" esp0x8f253810@10.0.0.1 esp0xf34ac696@10.10.10.10 tun0x102@10.0.0.1 tun0x101@10.10.10.10
000 #1: "linux-fw1" STATE_MAIN_I4 (ISAKMP SA established); EVENT_SA_REPLACE in 2620s; newest ISAKMP
```

Please note that in the above status output, information on the security association (SA) between the two hosts for connection 'linux-fw1' and 'linux-encdom' are displayed. This tells us that IKE negotiations have completed between VPN-1 and our Linux host.

When you are ready to test encrypting traffic through the tunnel, try using PING to test the connection between your Linux host and a resource in the encryption domain of the firewall. In our example, below, we used PING to test connectivity from our Linux host to an FTP server behind the firewall.

If you can successfully PING from your Linux host to resources in the encryption domain, check the VPN-1 Log Viewer to verify traffic is being encrypted:

The following actions appear in the Log Viewer:

key install	FreeSwan	fw_module		IKE Log: Phase 1 completion. 3DES/MD5/Pre shared secrets Negotiation Id:
key install	FreeSwan	fw_module	ip	scheme: IKE methods: Combined ESP: 3DES + MD5 (phase 2 completion) for host: 10.10.10.10 and for subnet: 192.168.0.0 (mask= 255.25
key install	FreeSwan	fw_module	ip	scheme: IKE methods: Combined ESP: 3DES + MD5 (phase 2 completion) for hosts: 10.10.10.10 and 10.0.0.1
decrypt	FreeSwan	fw_module	icmp	icmp-type 8 icmp-code 0 scheme: IKE methods: Combined ESP: 3DES + MD5
encrypt	fw_module	FreeSwan	icmp	icmp-type 0 icmp-code 0 scheme: IKE methods: Combined ESP: 3DES + MD5
decrypt	FreeSwan	FTPserver	icmp	icmp-type 8 icmp-code 0 scheme: IKE methods: Combined ESP: 3DES + MD5
encrypt	FTPserver	FreeSwan	icmp	icmp-type 0 icmp-code 0 scheme: IKE methods: Combined ESP: 3DES + MD5
decrypt	ftp	FreeSwan	FTPserver	tcp scheme: IKE methods: Combined ESP: 3DES + MD5

Key Install – represents a successful IKE negotiation between the Linux host and VPN-1. The first entry shows completion of IKE Phase 1, while the second and third entries show completion of IKE Phase 2.

Decrypt – represents successful decryption of encrypted traffic coming from the Linux host.

Encrypt – represents successful encryption of traffic going to the Linux host.

If you could not connect to resources in the encryption domain, please refer to the section *Troubleshooting and Debugging*, below.

Troubleshooting and Debugging

There are quite a few debugging options for S/WAN, especially considering a DEBUG tag can be added before compilation to enable extensive debugging information. Here are a few commands I found very useful:

1. ipsec whack --debug-all

This command tells the Pluto daemon to enable debug output for many aspects of the VPN. Once Pluto is instructed to provide debugging information, the following command can be used to view the output, as well as many configuration aspects that affect the VPN:

2. ipsec barf

This command displays debugging information for S/WAN, as well as configuration information from S/WAN's own files, to routes, to other files on the system that might effect S/WAN's ability to establish a VPN. A sample output of 'ipsec barf' is shown below:

```
Dec 8 13:15:50 zen Pluto[498]: "linux-fw1" #12: no acceptable Oakley Transform
Dec 8 13:15:50 zen Pluto[498]: | state transition function for STATE_MAIN_R0 failed: NO_PROPOSAL_CHOSEN
Dec 8 13:15:50 zen Pluto[498]: | next event EVENT_SA_EXPIRE in 0 seconds for #12
```

This example was taken from the tail-end of barf where IKE negotiations can be traced. You can see in the example above, an error was encountered where VPN-1 and S/WAN could not negotiate a valid time for IKE security associations to expire. This error was encountered only when VPN-1 was initiating the VPN connection, since the default IKE SA Expiration time on VPN-1 is 10080 minutes and S/WAN could only handle 480 minutes (or 28800 seconds) maximum.

3. ipsec eroute

This command allows you to configure virtual routes through the VPN tunnel. If you add network routes (i.e. route add) after the VPN tunnel is established, you may also need to setup the route with this command. If the Pluto daemon's debug information has been turned on (see above), virtual route lookup can be traced.

Here is a sample of the "eroute" routing table on the S/WAN host when the VPN is established:

```
# ipsec eroute
10.10.10.10/32      -> 10.0.0.1/32      => tun0x104@10.0.0.1
10.10.10.10/32      -> 192.168.0.0/16   => tun0x102@10.0.0.1
```

Here is an excerpt from the routing table on the S/WAN host when the VPN is established:

Destination	Gateway	Genmask	Flags	Iface
10.0.0.1	10.10.10.1	255.255.255.255	UGH	ipsec0
10.0.0.0	10.10.10.1	255.255.255.0	UG	eth0
10.10.10.10	0.0.0.0	255.255.255.255	UH	eth0
10.10.10.0	0.0.0.0	255.255.255.0	U	eth0
10.10.10.0	0.0.0.0	255.255.255.0	U	ipsec0
192.168.0.0	10.10.10.1	255.255.0.0	UG	ipsec0

The above examples should make it relatively painless to troubleshoot routing issues with the S/WAN host. Keep in mind that the routing examples above are taken from a working S/WAN host in the configuration described in previous sections of this document.

Further information on routing tables and the “ipsec eroute” command are beyond the scope of this document. For more information on troubleshooting S/WAN, consult the following man pages: ipsec, ipsec_auto, klipsdebug, ipsec.conf, ipsec.secrets, ipsec_eroute.

As a last piece of information, you may find the --show and --showonly options of “ipsec auto” very useful.