GSEC Practical Assignment
Version 1.2f

Implementing Site-to-Site IPSec Between a Cisco Router and Linux FreeS/WAN
Neil L. Cleveland
November 23, 2001

Introduction

It has been estimated that 80% of all Internet traffic passes through a Cisco router in reaching its ultimate destination.[1] For small-to-medium sized B2B intranets and extranets that are geographically dispersed, the FreeS/WAN IPSec implementation, running on the Linux operating system, offers a secure, cost-effective, robust, and high performance open source solution to successfully interface with this large existing Cisco population.

This paper begins by providing a brief overview of IPSec. It then discusses the major features, differences, and issues surrounding Cisco's IOS IPSec offering versus the FreeS/WAN offering. Finally, it describes an example implementation and step-by-step procedure that can be used to set up an IPSec site-to-site VPN between a Cisco 2621 IPSec-enabled router and an Intel Architecture-based computer running Linux with the FreeS/WAN 1.9 IPSec implementation.

IPSec Overview

IPSec is an acronym for *Internet Protocol Security*. It is a group of protocols developed by the IETF (Internet Engineering Task Force) to provide security for IP datagrams and any upper-layer protocols that use IP services. It uses modern cryptography to provide both authentication and encryption services. This security takes the form of data origin authentication (verifying the source of each datagram), data integrity authentication (verifying the contents of each datagram), data content confidentiality (ensuring the privacy of payload data), and anti-replay protection (ensuring packets are not intercepted and replayed at a later time).

IPSec uses a variety of existing security technologies to achieve the above goals. These include:

- The Diffee-Hellman key exchange protocol for deriving key material between peers on a public network
- Public Key Cryptography for signing Diffee-Hellman exchanges to guarantee the identity of the two parties and avoid man-in-the-middle attacks
- Bulk encryption algorithms, such as DES and 3DES, for encrypting payload data
- Keyed hash algorithms, such as HMAC-MD5 and HMAC-SHA for providing packet authentication
- Digital Certificates validated by third party certificate authorities to establish a trust relationship between parties

[1] IDG News Service, 3/02/01

IPSec offers three primary protocols…IKE (Internet Key Exchange), ESP (Encapsulating Security Protocol), AH (Authentication Header)…and two modes…Transport and Tunnel. Depending on the services required, these protocols and modes can be used separately or together and are explained in more detail below.

IKE is a key management protocol that provides shared security parameters and authenticated keys…also known as Security Associations (SA's)…between IPSec peers. There are two types of SA's associated with the IPSec framework: an IKE SA and an IPSec SA. An IKE SA defines the way in which two peers communicate; for example, which algorithm to use to encrypt IKE traffic, how to authenticate the remote peer, etc. IKE SA's are created in what is referred to as *Phase One* of the authentication process. The IKE SA is then used to produce the second type of SA, the IPSec SA. The IPSec SA's are used to protect data for the remainder of the session. The IPSec SA's are created in what is referred to as *Phase Two* of the authentication process.

AH provides proof-of-data on received packets, data integrity, and anti-replay protection. It does not provide data confidentiality services.

ESP provides all that AH provides in addition to optional data confidentiality and limited traffic flow confidentiality. Both AH and ESP can be used with either Transport or Tunnel Mode. Transport mode is used to protect upper layer protocols; tunnel mode is used to protect the entire IP datagram. In a typical site-to-site VPN…such as the one implemented in this paper…Tunnel mode is typically used with ESP to protect the entire IP packet while it is traversing between IPSec end points.

The standards governing IPSec and its associated technologies are contained in RFC (Request for Comments) documents available through the IETF (http://www.ietf.org/rfc.html). The primary RFC's are as follows:

RFC 2401	Umbrella document for IPSec
RFC 2406	Encapsulating Security Protocol (ESP)
RFC 2409	Authentication Header Protocol (AH)
RFC 2409	Internet Key Exchange Protocol (IKE)
RFC 2405	DES-CBC Encryption Standard
RFC 2451	3DES-CBC Encryption Standard
RFC 2403	HMAC-MD5 Keyed Hash Standard
RFC 2404	HMAC-SHA Keyed Hash Standard

Cisco IOS IPSec Offering

Cisco provides IPSec as an optional feature to their IOS (Internetworking Operating System). It is compatible with all Cisco routers running IOS 11.3 and later. Cisco's implementation supports all specifications contained in the current IETF RFC documents. Ordering information for the IPSec option can be obtained from their website (http://www.cisco.com).

Linux FreeS/WAN IPSec Offering

Linux FreeS/WAN is an open source implementation of IPSEC and IKE for Linux. According to the FreeS/WAN project official web site (http://www.freeswan.org), their objective is "to help make IPSec widespread by providing source code which is freely available, runs on a range of machines including ubiquitous cheap PCs, and is not subject to US or other nations' export restrictions." [2] The current version of FreeS/WAN is 1.91. It is freely available as stipulated in the GNU General Public license (http://www.fsf.org/copyleft/gpl.html).

FreeS/WAN runs on the Linux operating system and is compiled into the kernel to improve performance. Theoretically, it can support an unlimited number of tunnels and is only limited by its hardware. It supports both gateway-to-gateway applications (subject of this paper) and host-to-gateway applications. The FreeS/WAN implementation complies with all IPSec RFC's with the following exceptions: It does not support the use of X.509 certificates for authentication (although 3rd party patches are available), and it only supports 3DES encryption when using ESP. This latter restriction is self-imposed by the authors of FreeS/WAN because of a belief that lesser encryption standards are not secure[3].

[2] http://www.freeswan.org/intro.html

[3] Since FreeS/WAN does not natively support lesser encryption standards, it may be subject to export restrictions. Before exporting FreeS/WAN, the appropriate legal counsel should be consulted. See http://www.freeswan.org/freeswan_trees/freeswan-1.91/doc/politics.html#exlaw for a detailed discussion of this issue.

IPSec Gateway-to-Gateway Example Solution

This section will present an example solution using IPSec with FreeS/WAN to create a gateway-to-gateway tunnel with an IPSec-enabled Cisco router. It is based on an actual implementation and testing done by the author. The goal of this section is two-fold: first, to provide a real gateway-to-gateway solution that can be used immediately in cases where secure connectivity is required between two geographically dispersed entities…one of which is a Cisco IPSec-enabled router…and, second, to provide a basis…or template…for developing custom solutions using FreeS/WAN with Cisco and other similar IPSec vendor offerings.

Please refer to Figure 1 for the remainder of this discussion.

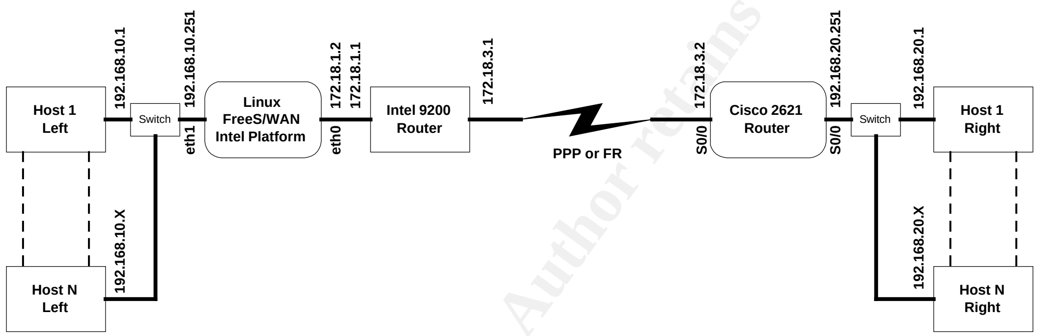

Figure 1 – Example Network for IPSec

Hardware and Software Requirements

The following hardware and software was used for this example solution.

- Cisco 2621 Router
    - Cisco IOS 12.0 with IPSec option
    - WIC-1T Serial Module
    - OPTIONAL: WIC-1DSU-T1 T1 interface
- Intel Architecture-based Server Platform (or equivalent)
    - Dual Pentium® III 700 MHZ Processors
    - 256 Mb Ram, 9.1 GB SCSI HD
    - 100 Mbps Ethernet Adapters (2 ea)
    - Red Hat 7.0 Standard Distribution OS
- Router with WAN interface (T1 or Serial PPP)
- Ethernet switch/hub (2 ea)
- At least two client PCs e/w 100Mb Ethernet adapters for testing

Example Topology/Test Bed

> **Note:** This paper only addresses the requirements and issues for implementing IPSec in a site-to-site (gateway-to-gateway) scenario. Before proceeding, a physical network similar to that shown in Figure 1 should be in place and all router and interfaces appropriately configured and tested.

As shown in Figure 1, the scenario for this example consists of two local area networks. These local networks are connected to a WAN via their specific routers. The network using FreeS/WAN as its gateway is shown on the left side of the diagram and is known as the LEFT network. The network using the Cisco IOS IPSec router as its gateway is on the right side of the diagram and is known as the RIGHT network. The goal of this example is to set up a secure IPSec tunnel between the gateways of the two networks so that clients on either side can communicate securely with each other. Accomplishing this goal will require separately configuring the Cisco router and FreeS/WAN server.

Cisco IOS IPSec Configuration

The following discussion addresses only IPSec issues and assumes that the Cisco IOS router is configured for IP on an Ethernet interface, and Frame Relay or PPP connection on a WAN interface, as depicted in Figure 1. This paper does not cover the initial Cisco configuration. For general Cisco configuration information, consult the Cisco documentation that came with the router or visit the Cisco support site at http://www.cisco.com. It is also assumed that all configuration will be done from the Cisco console port in Configuration Mode. In this example, the hostname for the Cisco router is `Cisco2621`.

Cisco IPSec configuration requires nine (9) tasks which will be covered individually.

Task 1 – Ensure Access Lists are compatible

Explanation: The router must pass IPSec packets through **UDP port 500** using **protocol #50 (ESP)**. Cisco Access Control Lists (ACL's) and any firewalls in place should be modified to allow this traffic to pass in both directions. Consult your firewall/router documentation for how to meet these requirements.

Task 2 - Enable the Internet Key Exchange (IKE) protocol

Description	Command
Enable IKE from Global Configuration Mode	`Cisco2621(config)#`**crypto isakmp enable**

Task 3 - Create IKE Policies

Description	Command
Set Policy Priority Number	`Cisco2621(config)#` **crypto isakmp policy 1**
Set Encryption Algorithm	`Cisco2621(config-isakmp)#` **encryption 3des**
Set Hash Algorithm	`Cisco2621(config-isakmp)#` **hash md5**
Set Authentication Method	`Cisco2621(config-isakmp)#` **authentication pre-share**
Set DH Group Identifier	`Cisco2621(config-isakmp)#` **group 2**
Set SA Lifetime	`Cisco2621(config-isakmp)#` **lifetime 86400**
Enter Config Mode	`Cisco2621(config-isakmp)#` **exit**

Explanation: The above steps create an IKE (ISAKMP) policy with a priority of "1" that will use 3DES encryption, the MD5 hash algorithm, pre-shared authentication, Diffee-Hellman Group 2, and will have a lifetime of 24 hours. The purpose of this policy is to define which security parameters will be used to protect subsequent IKE negotiations and mandates how the peers are authenticated. Later on, a similar policy will be created on FreeS/WAN that the two peers will use to establish an agreed-upon IKE security association (SA).

Task 4 - Create Pre-shared Keys for IKE Authentication

Description	Command
Create Pre-shared Keys	`Cisco2621(config)#` **crypto isakmp key 123456789 address 172.18.1.2**

Explanation: 172.18.1.2 is the address of the FreeS/WAN gateway (remote peer). Also, this example uses a simple pre-shared key. A production environment should use a very long key for better security.

Task 5 - Create Security Association Global Lifetime

Description	Command
Create Global Lifetime	`Cisco2621(config)#` **crypto ipsec security-association lifetime seconds 3600**

Explanation: This sets the IPSec SA lifetime to 1 hour. Once the lifetime expires, a new SA will be negotiated between the two gateways (unless the other gateway has a shorter SA lifetime. If so, it will take precedence).

Task 6 - Create a Crypto Access List

Description	Command
Specify which IP packets will be protected	`Cisco2621(config)#` **access-list 100 permit ip 192.168.10.0 0.0.0.255 192.168.20.0 0.0.0.255**

Explanation: Crypto access lists are used to define which IP traffic will be protected by crypto. The above step creates an access list (100) that will be used to protect all traffic originating from the internal side of the Cisco gateway on subnet 192.168.10.0 that is destined for the 192.168.20.0 subnet on the external side of the Cisco gateway. This access list will be assigned to a crypto map in a later step.

Task 7 - Define a Transform Set

Description	Command
Define a transform set for use during Phase 2	`Cisco2621(config)#` **crypto ipsec transform-set freeswan esp-3des esp-md5-hmac**

Explanation: A transform-set contains a certain set of security protocols and algorithms. During the IPSec security association negotiation, the peer gateways will agree to use a particular transform-set for their data exchange. This step sets up the parameters for the IPSec Security Association. In this example, the transform-set is named 'freeswan'; it will use the ESP algorithm with 3DES encryption for payload confidentiality and HMAC-MD5 for payload integrity. Using these parameters provides maximum security.

Task 8 - Create Crypto Map Entry

Description	Command
Name the crypto map entry to create	`Cisco2621(config)#` **crypto map freeswan 10 ipsec-isakmp**
Match access list to this crypto map	`Cisco2621(config-crypto-m)#` **match address 100**
Specify the remote IPSec peer	`Cisco2621(config-crypto-m)#` **set peer 172.18.1.2**
Specify the transform set to use for this crypto map	`Cisco2621(config-crypto-m)#` **set transform-set freeswan**
Specify Perfect Forward Secrecy	`Cisco2621(config-crypto-m)#` **set pfs group 2**
Enter Config Mode	`Cisco2621(config-crypto-m)#` **exit**

Explanation: In the above steps, the crypto map has been named "freeswan" and assigned a sequence number of "10". It will be used with *ipsec-isakmp* keying since we are running the IKE protocol. It will be matched to access list 100 (created in a previous step). The crypto map entry will apply to the FreeS/WAN peer (172.18.1.2) and will use the transform set named *freeswan*. Finally, all communications using this crypto map entry will be protected using Diffee-Hellman Group 2 for Perfect Forward Secrecy.

Task 9 - Apply Crypto Map Set to Interface

Description	Command
Apply the map set to the Cisco gateway interface	`Cisco2621(config)#` **crypto map freeswan**

Explanation: This step applies the crypto map named "freeswan" to the external gateway of the Cisco router. Applying the Crypto Map Set to an interface instructs the router to evaluate all the interface's traffic against the Crypto Map Set and use the specified policy during connection or security association negotiation.

This completes the Cisco IOS IPSec example configuration. The next section will configure the FreeS/WAN gateway.

FreeS/WAN IPSec Preliminary Preparation

This paper assumes that Red Hat 7.0 has already been successfully installed with the basic services necessary for network connectivity on an Intel-based server meeting the requirements mentioned earlier. In addition to this requirement, the following tarred files should be obtained for use in later configuration:

- The Linux 2.2.18 Kernel
  - Filename: linux-2.2.18.tar.gz
  - Obtain from: ftp://ftp.kernel.org
- FreeS/WAN 1.9
  - Filename: freeswan-1.9.tar.gz
  - Obtain from: http://www.freeswan.org/download.html
- The GMP Library
  - Filename: gmp-3.1.1.tar.gz
  - Obtain from: http://www.swox.com/gmp

Before proceeding to the next section, the tarred packages should be moved to the */usr/src* directory on the system where FreeS/WAN will be installed.

FreeS/WAN Installation and Configuration Overview

Installation and configuration of FreeS/WAN involves the completion of four major tasks:

Task 1 - Build the new kernel with FreeS/WAN
Task 2 - Update the LILO boot manager and reboot the system
Task 3 - Enable Packet Forwarding
Task 4 - Update the *ipsec.secrets* and *ipsec.conf* files

Note 1: All configuration must be done as the *root* user.
Note 2: Commands shown below in Courier font should be entered by the user at the root prompt.

Task 1 - Build the New Linux 2.2.18 FreeS/WAN Kernel

1. Untar GMP Library and Install

```
cd /usr/src
tar xzvf gmp-3.1.1.tar.gz
cd gmp-3.1.1
./configure
make
make install
```

2. Untar FreeS/WAN and Install

```
cd /usr/src
tar xzvf freeswan-1.9.tar.gz
cd freeswan-1.9
make insert
make programs
make install
```

3. Make New Links

```
cd /usr/include
mv scsi scsi.bak
mv asm asm.bak
mv linux linux.bak
ln –s /usr/src/linux/include/scsi scsi
ln –s /usr/src/linux/include/asm-i386 asm
ln –s /usr/src/linux/include/linux linux
```

4. Untar New Kernel and Install

```
cd /usr/src
tar xzvf linux-2.2.18.tar.gz
mv linux linux-2.2.18
ln-s linux-2.2.18 linux
cd linux
make mrproper
make menuconfig
```
(**See guidelines below[4]**)
```
make dep
make clean
make bzImage
make modules
make modules_install
```

5. Move Kernel Library

```
cp /usr/src/linux/arch/i386/boot/bzImage /boot/vmlinuz-
2.2.18
cp /usr/src/linux/vmlinux /boot/vmlinux-2.2.18
cp /usr/src/linux/System.map /boot/System.map-2.2.18
```

[4] 'menuconfig' is an interactive script that allows you to configure your new kernel. The interface consists of menus and submenus. The major change you must make to the kernel is in the **networking -> IPSec Options (FreeS/WAN)** submenu. You should indicate **Yes** to all questions in this section. Other areas that should be enabled are routing, masquerading, loopback device support. You should be familiar with Linux kernel issues before modifying any other settings from their defaults. Generally speaking, from a security standpoint, a minimal installation is recommended.

```
cd /boot
mkinitrd vmlinuz-2.2.18.img 2.2.18 (ram disk optional)
```

Task 2 - Update LILO and Reboot

1. Using an editor (e.g., *vi* or *pico*), add the below entries in */etc/lilo.conf* for the new kernel image. See *man* pages for details on how to update this file. Be sure to save your changes.

   - `vmlinuz-2.2.18`
   - `vmlinuz-2.2.18.img` **(ram disk optional)**

2. Execute LILO to activate changes by entering the following command at the root prompt.

   ```
   lilo
   ```

3. Reboot the system.

   ```
   shutdown –r now
   ```

Note: Be sure to choose the new kernel when prompted during the boot-up process

4. After the reboot is complete, verify the new 2.2.18 kernel is running.

   ```
   uname –r
   ```

Task 3 - Enable Packet Forwarding

1. Edit */etc/sysconfig/network* and verify the following line:

   ```
   net.ipv4.ip_forward=1
   ```

2. From the command line, restart network services so this change is recognized.

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

Task 4 - Configure FreeS/WAN files

1. Update the /etc/*ipsec.secrets* file

   Using an editor, add the pre-shared key to the *ipsec.secrets* file and save your changes.

   ```
   172.18.1.2 172.18.3.2 :PSK "123456789"
   ```

Explanation: This entry will be used during IKE authentication between the two IPSec gateways. This file must be very secure and only be readable and writable by root. Change its permissions by issuing the following command at the root prompt:

```
chmod 600 /etc/ipsec.secrets
```

2. Update the */etc/ipsec.conf* file

This is the main configuration file for FreeS/WAN. Open this file with your editor. Then use the template on the following page to make the required modifications. Be sure to save your changes. The required changes are shown in **bold**.

Explanation: This file is where the main configuration takes place and is analogous to the steps completed in the Cisco IPSec configuration sections. Notice the use of *left*, *leftsubnet*, *leftnexthop*, *right*, *rightsubnet*, and *rightnexthop* to configure the IP addresses, subnets, and next hop addresses that FreeS/WAN will use to route secure datagrams. Other entries in this file can be customized for a particular application. Consult the FreeS/WAN *man* pages for further information.

```
# /etc/ipsec.conf - FreeS/WAN IPsec configuration file

# More elaborate and more varied sample configurations can be found
# in FreeS/WAN's doc/examples file.

# basic configuration
config setup
      # THIS SETTING MUST BE CORRECT or almost nothing will work;
      # %defaultroute is okay for most simple cases.
      interfaces="ipsec0=eth0"
      # Debug-logging controls:  "none" for (almost) none, "all" for
      lots.
      klipsdebug=none
      plutodebug=none
      # Use auto= parameters in conn descriptions to control startup
      actions.
      plutoload=%search
      plutostart=%search
      # Close down old connection when new one using same ID shows up.
      uniqueids=yes


# defaults for subsequent connection descriptions
conn %default
      # How persistent to be in (re)keying negotiations (0 means very).
      keyingtries=1
      # Parameters for manual-keying testing (DON'T USE OPERATIONALLY).
      # Note:  only one test connection at a time can use these
      parameters!
      spi=0x200
      esp=3des-md5-96
      espenckey=0x01234567_89abcdef_02468ace_13579bdf_12345678_9abcdef0
      espauthkey=0x12345678_9abcdef0_2468ace0_13579bdf
      # RSA authentication with keys from DNS.
      authby=rsasig

# sample connection
conn freeswan-cisco
      # Left security gateway, subnet behind it, next hop toward right.
      left=172.18.1.2
      leftsubnet=192.168.10.0/24
      leftnexthop=172.18.1.1
      # Right security gateway, subnet behind it, next hop toward left.
      right=172.18.2.2
      rightsubnet=192.168.20.0/24
      rightnexthop=
      # To authorize this connection, but not actually start it, at
      startup,
      # uncomment this.
      auto=add
      authby=secret
```

Verification of the FreeS/WAN-to-Cisco IPSec Gateway

At this point, all installation and configuration of the two gateways should be complete. This section discusses the verification testing methodology to ensure the gateways are properly authenticating and encrypting traffic between the protected subnets.

- Starting the IPSec Services

  The Cisco router recognizes changes immediately; therefore it is not necessary to initiate a startup procedure. However, FreeS/WAN tunnels must be started manually (This can be changed – See FreeS/WAN documentation). To start the FreeS/WAN service, enter the following from the Linux root command prompt:

```
ipsec auto --up freeswan-cisco
```

  Explanation: This command tells FreeS/WAN to locate a connection type (in */etc/ipsec.conf*) named *freeswan-cisco* and negotiate an IPSec tunnel with the appropriate IPSec gateway server (Cisco).

  Note: To disable this connection, the following command can be issued:

```
ipsec auto --down freeswan-cisco
```

- FreeS/WAN IPSec Verification

  To verify the FreeS/WAN IPSec tunnel, use the following command:

```
ipsec look
```

  You should see output on the console similar to the following:

```
freeswan Thu Jun 14 10:51:29 PDT 2001
192.168.10.0/24    -> 192.168.20.0/24    => tun0x102c@172.18.3.2 esp0x9c2fb807@172.18.2.2
ipsec0->eth0 mtu=16260->1500
esp0x82cd8f10@172.18.1.2 ESP_3DES_HMAC_MD5: dir=in  src=172.18.3.2 iv_bits=64bits
iv=0x072d254cc597307e ooowin=64 alen=128 aklen=128 eklen=192 life(c,s,h)=add(24222,0,0)
esp0x9c2fb807@172.18.2.2 ESP_3DES_HMAC_MD5: dir=out src=172.18.1.2 iv_bits=64bits
iv=0x45617792eded98b7 ooowin=64 alen=128 aklen=128 eklen=192 life(c,s,h)=add(24222,0,0)
tun0x102b@172.18.1.2 IPIP: dir=in  src=172.18.3.2 life(c,s,h)=add(24222,0,0)
tun0x102c@172.18.3.2 IPIP: dir=out src=172.18.1.2 life(c,s,h)=add(24222,0,0)
Destination     Gateway         Genmask         Flags   MSS Window  irtt Iface
0.0.0.0         172.18.1.1      0.0.0.0         UG        0 0          0 eth0
172.18.1.0      0.0.0.0         255.255.255.0   U         0 0          0 eth0
172.18.1.0      0.0.0.0         255.255.255.0   U         0 0          0 ipsec0
172.18.2.0      172.18.1.1      255.255.255.0   UG        0 0          0 eth0
192.168.20.0    172.18.1.1      255.255.255.0   UG        0 0          0 ipsec0
```

This example shows:

- A tunnel between the two subnets named *freeswan*
  192.168.10.0/24 -> 192.168.20.0/24
- The destination gateway being used
  172.18.3.2

- An outgoing connection
- An incoming connection
- Both connections are using ESP with 3DES encryption and MD5 authentication

FreeS/WAN contains many other useful commands for both troubleshooting and customizing your implementation. Consult the FreeS/WAN *man* pages or visit the project web site at http://www.freeswan.org/.

- Cisco IPSec Verification

Cisco IPSec tunnel verification can be monitored from the Cisco console. When the tunnel is first initiated, output on the console should be similar to the following:

```
3d23h: ISAKMP (0:1): Checking ISAKMP transform 1 against priority 3
policy
3d23h: ISAKMP:      life type in seconds
3d23h: ISAKMP:      life duration (basic) of 3600
3d23h: ISAKMP:      encryption 3DES-CBC
3d23h: ISAKMP:      hash MD5
3d23h: ISAKMP:      auth pre-share
3d23h: ISAKMP:      default group 2
3d23h: ISAKMP (0:1): atts are acceptable. Next payload is 3
3d23h: ISAKMP (0:1): SA is doing pre-shared key authentication
3d23h: ISAKMP (1): SA is doing pre-shared key authentication using
id type ID_IPV4_ADDR
3d23h: ISAKMP (1): sending packet to 172.18.1.2 (R) MM_SA_SETUP
3d23h: ISAKMP (0): received packet from 172.18.1.2 (N) NEW SA
3d23h: ISAKMP (0:1): retransmitting phase 1...
3d23h: ISAKMP (1): sending packet to 172.18.1.2 (R) MM_SA_SETUP
3d23h: ISAKMP (0): received packet from 172.18.1.2 (N) NEW SA
3d23h: ISAKMP (0:1): retransmitting phase 1...
3d23h: ISAKMP (1): sending packet to 172.18.1.2 (R) MM_SA_SETUP
3d23h: ISAKMP (0:1): processing KE payload. message ID = 0
3d23h: ISAKMP (0:1): processing NONCE payload. message ID = 0
3d23h: ISAKMP (0:1): SKEYID state generated
3d23h: ISAKMP (1): sending packet to 172.18.1.2 (R) MM_KEY_EXCH
3d23h: ISAKMP (1): received packet from 172.18.1.2 (R) MM_KEY_EXCH
3d23h: ISAKMP (0:1): processing ID payload. message ID = 0
3d23h: ISAKMP (0:1): processing HASH payload. message ID = 0
3d23h: ISAKMP (0:1): SA has been authenticated with 172.18.1.2
3d23h: ISAKMP (1): ID payload
                  next-payload : 8
                  type         : 1
                 protocol     : 17
                port         : 500
                  length       : 8
3d23h: ISAKMP (1): Total payload length: 12
3d23h: ISAKMP (1): sending packet to 172.18.1.2 (R) QM_IDLE
3d23h: ISAKMP (1): received packet from 172.18.1.2 (R) QM_IDLE
3d23h: ISAKMP (0:1): processing SA payload. message ID = 960538520
3d23h: ISAKMP (0:1): Checking IPSec proposal 0
3d23h: ISAKMP: transform 0, ESP_3DES
3d23h: ISAKMP:   attributes in transform:
3d23h: ISAKMP:      group is 2
```

```
3d23h: ISAKMP:      encaps is 1
3d23h: ISAKMP:      SA life type in seconds
3d23h: ISAKMP:      SA life duration (basic) of 28800
3d23h: ISAKMP:      authenticator is HMAC-MD5
3d23h: ISAKMP (0:1): atts are acceptable.
3d23h: ISAKMP (0:1): processing NONCE payload. message ID =
960538520
3d23h: ISAKMP (0:1): processing KE payload. message ID = 960538520
3d23h: ISAKMP (0:1): processing ID payload. message ID = 960538520
3d23h: ISAKMP (1): ID_IPV4_ADDR_SUBNET src
192.168.10.0/255.255.255.0 prot 0 port 0
3d23h: ISAKMP (0:1): processing ID payload. message ID = 960538520
3d23h: ISAKMP (1): ID_IPV4_ADDR_SUBNET dst
192.168.20.0/255.255.255.0 prot 0 port 0
3d23h: ISAKMP (1): sending packet to 172.18.1.2 (R) QM_IDLE
3d23h: ISAKMP (1): received packet from 172.18.1.2 (R) QM_IDLE
3d23h: ISAKMP (0:1): Creating IPSec SAs
3d23h:         inbound SA from 172.18.1.2      to 172.18.3.2
(proxy 192.168.10.0    to 192.168.20.0    )
3d23h:         has spi 231739344 and conn_id 2000 and flags 25
3d23h:         lifetime of 28800 seconds
3d23h:         outbound SA from 172.18.3.2      to 172.18.1.2
(proxy 192.168.20.0    to 192.168.10.0    )
3d23h:         has spi 1066223580 and conn_id 2001 and flags 25
3d23h:         lifetime of 28800 seconds
3d23h: ISAKMP (0:1): deleting node 960538520
3d23h: ISAKMP (0:1): purging node 960538520
..................
```

Explanation: The Cisco router generates this output during the negotiation with the remote gateway (FreeS/WAN). It basically shows the negotiation process that takes place as the two gateways attempt to agree on the security associations to use for the tunnel. Notice that authentication finally takes place and an IPSec SA is agreed to.

- Ping Between Subnets

The *ping* command can be used to verify connectivity between the two remote subnets. Referring to Figure 1:

- Left (FreeS/WAN)

```
Ping 192.168.20.1
```

- Right (Cisco)

```
Ping 192.168.10.1
```

Performance

Based upon the author's testing, it was noted that, even though the tunnel between the two gateways used the 3DES algorithm to encrypt IP datagrams, throughput was still quite good. A one megabyte file was transferred over a T1 link between the gateways in 11.54 seconds giving a throughput of approximately 661 Kbps. Interestingly, when two FreeS/WAN gateways were configured to communicate with each other using IPSec, the same file was transferred in 5.08 seconds with approximate throughput of 1501 Kbps. These results would appear to indicate that the drop in performance in the first scenario was due to the Cisco router. However, this is only speculation, as the same test was not run between two like-model Cisco routers.

Summary

This paper has demonstrated an example implementation using FreeS/WAN with a Cisco IOS IPSec-enabled router to create secure communication between protected subnets such as might be found in a corporate Extranet. Although the implementation was specific, it provides a real working solution that can be used as a template to allow the reader to create his/her own custom IPSec solutions.

Bibliography

CISCO – *White Paper – IPSec*, http://www.cisco.com/warp/public/cc/techno/protocol/ipsecur/ipsec/tech/ipsec_wp.htm

CISCO – *Configuring IPSec Network Security*, http://www.cisco.com/univercd/cc/td/doc/product/software/ios121/121cgcr/secur_c/scprt4/scdipsec.htm

CISCO - *IPSec Network Security Commands*, http://www.cisco.com/univercd/cc/td/doc/product/software/ios121/121cgcr/secur_r/srprt4/srdipsec.htm

CISCO - *Configuring and Troubleshooting Cisco's Network Layer Encryption*, http://www.cisco.com/warp/public/707/16.html

LINUX - IPSEC - *The FreeS/WAN Project - FreeS/WAN Documentation*, http://www.freeswan.org/freeswan_trees/freeswan-1.91/doc/index.html

LINUX – IPSEC – *My FreeS/WAN 1.5 Setup Diary*, http://www.sandelman.ottawa.on.ca/linux-ipsec/html/2000/11/msg00571.html

LINUX – *The Linux Kernel HOWTO* http://www.linuxdoc.org/HOWTO/Kernel-HOWTO.html

"Cisco software flaw threatens TCP Traffic Integrity", IDG News Service, 3/2/01 http://www.itworld.com/Net/1756/Cisco-010302/

Kent, S. et al, RFC 2401 - Security Architecture for the Internet Protocol, IETF Network Working Group, November 1998, http://www.ietf.org/rfc/rfc2401.txt

Doraswamy, N. et al, IPSec – The New Security Standard for the Internet, Intranets, and Virtual Private Networks, Prentice Hall, Upper Saddle River, New Jersey, 1999

Koisur, D., Building and Managing Virtual Private Networks, John Wiley and Sons, 1998