Linux VPN
Technical Analysis and HOWTO

Authors: Hank Janssen, Chris Rawlings, Sam Vaughan

Abstract:

This document intends to give the reader a analysis of VPN within the Linux Operating System. Specifically, it provides a breakdown of VPN components and a description of what is available to Linux Administrators in terms of manageability and functionality. It also provides limited HOWTO's in the area's of VPN and IPsec.

Table of Contents

1 Introduction

1.1 Scope of Document

This document was the result of an Virtual Private Network (VPN) analysis we did in the OSSL at Microsoft.

When the document was created for the analysis we realized that it contained a good overview of what is available for VPN in Linux. In the process of analyzing VPN we had to do installation and testing, so you can also find several HOWTO's as well.

This document you are reading is a hybrid HOWTO and high-level technical analysis of the IPsec VPN capabilities currently available within Linux.

Linux can be configured to be a gateway in a site-to-site IPsec connection scenario and it can also be used as an IPsec VPN server in a remote access scenario, often referred to as a 'Road Warrior' setup.

Unless noted otherwise, the information provided in this document will be based on the software that is available on Red Hat Enterprise Linux 4 WS (RHEL4) and Fedora Core 5 (FC5). The packages installed on each of these operating systems is the same except for their version numbers; the software pre-installed on FC5 is newer and is assumed to have more features and more/less bugs.

1.2 Out of scope

This document is not intended to give the user a definitive overview of the VPN offerings on Linux. It also is not intended to be the end all be all of VPN configurations.

No assumption or in-depth evaluation is done regarding compatibility with Microsoft VPN clients or implementations.

1.3 Conclusion

The document will not make you an IPSec or VPN expert on Linux, and will not provide you with a step by step guide on setting it up. But it will give you a better understanding of the steps involved. And give you an overview of VPN and IPsec on Linux.

2 Linux IPSec Overview

2.1 IPsec – A Description

IPSec is described in many different publications, but a clear overview can be found in wikipedia (http://www.wikipedia.org/). wikipedia describes IPsec as follows; (Edited for the sake of condensing, see wikipedia for the complete entry for IPsec)

"IPsec was intended to provide either ***transport mode****: end-to-end security of packet traffic in which the end-point computers do the security processing, or* ***tunnel mode****: portal-to-portal communications security in which security of packet traffic is provided to several machines (even to whole LANs) by a single node.*

IPsec can be used to create Virtual Private Networks (VPN) in either mode, and this is the dominant use. Note, however, that the security implications are quite different between the two operational modes."

"IPsec protocols operate at the network layer, layer 3 of the OSI model. Other Internet security protocols in widespread use, such as SSL and TLS, operate from the transport layer up (OSI layers 4 - 7). This makes IPsec more flexible, as it can be used for protecting both TCP- and UDP-based protocols, but increases its complexity and processing overhead, as it cannot rely on TCP (OSI layer 4) to manage reliability and fragmentation."

End wikipedia quotation.

A native IPsec stack, known as NETKEY or 26sec, has existed in the Linux kernel since version 2.5.47. There is also another IPsec kernel implementation known as KLIPS, but this document will only cover NETKEY since it has become the standard for 2.6 kernels.

NOTE*: The native IPsec implementation in the 2.6 kernel (NETKEY) has been backported to the 2.4 kernel. It is not contained in the vanilla 2.4 kernel, but distributions such as Debian are mentioned to have support for the backport. Details on the backport are scarce. If a 2.4 kernel was needed for IPsec VPN's, it would be better to install another implementation of IPsec, such as KLIPS, which is included/compatible with FreeS/WAN, Openswan, and strongSwan.*

2.2 Authentication Protocols Supported

The following authentication protocols are supported when using IPsec-Tools:

- Pre-Shared Secret Keys (PSK)
- X.509 Certificates
- GSSAPI Kerberos V (Not clear to what extent it is supported)
- XAuth + Hybrid Authentication (pure XAuth is not supported by raccoon)

2.3 Tunneling Protocols Supported

The following VPN protocols are supported in Linux:

- IPsec Transport Mode – used for host-to-host IPsec connections.

- IPsec Tunnel Mode (VPN) – used for site-to-site IPsec tunnels.
- L2TP/IPsec (VPN) – Using the additional PPP and L2TP daemons available in Linux allows Windows and Mac clients to connect to a Linux VPN server.

With separate l2tp daemon:

- L2TP/IPsec

2.4 Encryption and Authentication Algorithm Supported

The encryption and authentication algorithms supported by IPsec-Tools package are listed below and split according to their support within the following categories:

- Manually Keyed VPN Connections (using 'setkey' utility)
    - Setkey (configured through /etc/setkey.conf)

Automatically Keyed VPN Connections (using the IKE daemon, 'racoon')

- Racoon: IKE Phase 1 (configured through /etc/racoon/racoon.conf)
- Racoon: IKE Phase 2 (configured through /etc/racoon/racoon.conf)

2.4.1 Setkey (For Manually Keyed VPN Connections)

2.4.1.1 Supported Encryption Algorithms

The following list of supported encryption algorithms was taken from the Linux man page for setkey:

Algorithm	*Key Length (bits)*	*Note*
Des-cbc	64	esp: rfc2405, esp-old: rfc1829
3des-cbc	192	rfc2451
null	0 to 2048	rfc2410
blowfish-cbc	40 to 448	rfc2451
cast128-cbc	40 to 128	rfc2451
Des-deriv	64	ipsec-ciph-des-derived-01
3des-deriv	192	no document
rijndael-cbc	128/192/256	rfc3602
twofish-cbc	0 to 256	draft-ietf-ipsec-ciph-aes-cbc-01
Aes-ctr	160/224/288	draft-ietf-ipsec-ciph-aes-ctr-03

2.4.1.2 Supported Authentication Algorithms

The following list of supported authentication algorithms was taken from the Linux man page for setkey:

Algorithm	*Key Length (bits)*	*Note*
hmac-md5	128	rfc2403
hmac-sha1	160	rfc2404
keyed-md5	128	rfc1828 (old)
keyed-sha1	160	no document
null	0 to 2048	(for debugging)
hmac-sha256	256	draft-ietf-ipsec-ciph-sha-256-00
hmac-sha384	384	no document
hmac-sha512	512	no document
hmac-ripemd160	160	rfc2857
Aes-xcbc-mac	128	rfc3566
Tcp-md5	8 to 640	rfc2385

2.4.2 Racoon (For Automatically Keyed VPN Connections)

2.4.2.1 Racoon – What is it

The IKE (ISAKMP/Oakley) key management protocol is used to establish security associations with other hosts, the IKE deamon on Linux is usually Racoon. The SPD (Security Policy Database) in the kernel usually triggers racoon. Racoon is part of the IPsec suite of tools.

IPsec knows two phases;

- Phase 1 - part of the IPsec Key Exchange (IKE) operations performed by the IKE daemon (Usually Racoon). Its goal is to authenticate the peers and set up master keys for performing a secured Phase 2.
- IPsec phase 2 - The goal of phase 2 is to derive the keys used for exchanging IPsec traffic. Phase 2 rekeying can occur regularly while IPsec traffic is exchanged.

2.4.2.2 Racoon: IKE Phase 1

Supported Encryption Algorithms

- des
- 3des
- blowfish
- cast128
- aes

Supported Hash Algorithm

- md5
- sha1

- sha256
- sha384
- sha512

2.4.2.3 Racoon: IKE Phase 2

Supported Encryption Algorithms

- des
- 3des
- des_iv64
- des_iv32
- rc5
- rc4
- idea
- 3idea
- cast128
- blowfish
- null_enc
- twofish
- rijndael
- aes

Supported Authentication Algorithms

- des
- 3des
- des_iv64
- des_iv32
- hmac_md5
- hmac_sha1
- hmac_sha256
- hmac_sha384

3 Initial Configuration and Setup

3.1 Packages needed

A Linux VPN server is not created from a single software package; several packages/components are required. The initial installation and configuration of a Linux VPN server can be quite frustrating.

The following is a list of software packages that are usually included in default installations of RHEL4 and FC5:

- *NETKEY* – This is the native IPsec implementation in 2.6 kernels. It is also known as 26sec. This already compiled into the kernel in RHEL4, FC5, Ubuntu, and most other distributions.
- *IPsec-Tools* – This is the package that contains both an IKE server/daemon (*racoon*) and the command line utility used to manage Security Associations and Security Policies for an IPsec connection (*setkey*). The following command-line interface (CLI) executables are distributed with the IPsec-Tools package:
    - racoon – the IKE daemon
    - racoonctl – the racoon administrative control tool
    - setkey – the tool that manages Security Associations (SA) and Security Policies (SP) for IPsec connections
    - plainrsa-gen – a tool used to generate plain RSA keys. This is an alternative to using Preshared Secret Keys (PSK) and X.509 certififcates for authentication.

    ***NOTE:** The current version of IPsec tools as of this writing was 0.6.6. The version supplied with RHEL4 is 0.3.3 and the version supplied with FC5 is 0.6.5. When using RHEL4 it might be wise to manually download and install the latest version for the features and bug fixes that have changed since version 0.3.3.*
- *OpenSSL* – Needed to generate X.509 certificates. This is should be included in most default installations of RHEL4 and FC5.
    - The version supplied with RHEL4 is 0.9.7a and is dated February 2003.
    - The version supplied with FC5 is 0.9.8a and is dated October 2005.
- *PPPD* – a PPP daemon that is used when configuring an L2TP/IPsec VPN connection for remote clients.
- *Netfilter/IPTables* – a administration tool for IPv4 packet filtering and NAT.
- Sysklogd – System and Kernel logging package

The following is a list of optional software packages that will need to be installed depending on what type of IPsec VPN connection a Linux system is being configured for:

- *L2TPD* – an L2TP daemon that is used when configuring an L2TP/IPsec VPN connection for remote clients on Windows and Mac OS X.
- *Linux-HA* – High Availability package for Linux.

3.1.1 User-Space Tools/Software and additional tools

There are several options available, as far as user-space tools are concerned, to manage and configure the IPsec features provided in the kernel by NETKEY. Each competing package includes an Internet Key Exchange (IKE) server as well as tools for configuring an IPsec connection:

- Ipsec-Tools – This is the default package installed with RHEL4 and FC5. It includes *racoon* (an IKE daemon) as well as *setkey* (used to manually configure Security Associations and Security Policies for an IPsec connection)
- Openswan – A good alternative to IPsec-Tools that is in active development.
- strongSwan – Another alternative to IPsec-Tools, but not quite as common as Openswan.
- FreeS/WAN – one of the first IPsec packages available on Linux which doesn't see much development anymore
- ISAKMPD – a Linux port of the IPsec tools available on OpenBSD

All of these competing packages can be used with NETKEY, but IPsec-Tools is the only one that is pre-installed on RHEL4, FC5, and most of the other distributions such as Ubuntu. Since it is the default IPsec package available it will likely see accelerated use and continued development in the future. As such, this paper will mostly focus on the features and capabilities provided by IPsec-Tools.

In addition to the software mentioned above, other packages for L2TP and PPP may need to be installed when a Linux server is providing VPN access for remote Windows clients.

***NOTE**: The native IPsec implementation in the 2.6 kernel (NETKEY) has been backported to the 2.4 kernel. It is not contained in the vanilla 2.4 kernel, but distributions such as Debian are mentioned to have support for the backport. Details on the backport are scarce. If a 2.4 kernel was needed for IPsec VPN's, it would be better to install another implementation of IPsec, such as KLIPS, which is included/compatible with FreeS/WAN, Openswan, and strongSwan.*

3.2 Installation

3.2.1 Initial Deployment

Initial deployment of a Linux VPN server is not a simple task. First, most management and configuration of a server require the use of the command-line interface (CLI). There are many technologies and acronyms associated with IPsec that an administrator would need to have a good understanding of when setting up the configuration files. Second, it may not even be apparent to an administrator what steps need to be taken to build a Linux VPN server. As mentioned in the chapter *3.1 Packages needed*, there are several software tools to choose from when building a VPN server. Third, the documentation that exists (software documentation, howto's, etc) for building a VPN solution in Linux is somewhat lacking in completeness.

3.2.2 Installing

If you already have the IPSec components installed, upgrading the software components in RHEL4 and FC5 is relatively simple. In RHEL4 software is upgraded and installed using RPM package management. Updates for supported software in RHEL4 can be installed automatically via the up2date tool in Gnome or via the command-line by issuing the command 'up2date –u'. In RHEL4, an RPM for the L2TP daemon will need to be obtained from a third party.

In FC5 and RHEL4, updates for supported software can be performed via the command-line by issuing the command 'yum update'. This will update all currently installed packages to the latest version provided by the *yum* database.

Installation of new packages not present on the server can also be installed through yum.

3.2.3 Wizard

RHEL4 and FC5 come with a Red Hat network configuration tool that provides wizard based IPsec setup, but it has limited functionality for configuring an IPsec connection. At this point in time, it would be best to stick with the command-line for configuring IPsec. For reference, though, the wizard can be accessed through the following system menus in Gnome:

- RHEL4 (System-config-network v 1.3.22.0.EL.4.2)
  - Application → System Settings → Network → Network Tab
- FC5 (System-config-network v 1.3.30)
  - System → Administration → Network → Network Tab

3.3 Minimum Configuration Settings / Default Settings

Although both RHEL4 and FC5 come pre-installed with the software needed to implement an IPsec VPN connection (the NETKEY/IPsec-Tools combination), neither of the Linux distributions ship with much in the way of default configurations. An administrator would need to configure most of the files required for an IPsec connection from scratch. The time required to initially configure an IPsec connection could vary widely depending on an administrator's previous experience with IPsec since knowledge of SA's, SP's, IKE, etc is required. In addition, the OS-installed documentation for configuring an IPsec VPN is somewhat basic (refer to chapter *8 References* of this document for more information).

For the most basic configuration (an IPsec connection that is manually keyed), an administrator would need to edit only one file located at '/etc/setkey.conf'. The file 'setkey.conf' is used to specify Security Policies for an IPsec connection and also Security Associations when the IKE daemon (raccoon) is not used to automatically negotiate them. There is no default configuration that exists for 'setkey.conf', so the file needs to be created manually along with the initial configuration. For a manually keyed IPsec connection in transport mode, the 'setkey.conf' configuration file would look similar to the following:

```
#!/usr/bin/setkey -f

# Flush the Security Association Database (SADB)
flush;
# Flush the Security Policy Database (SPD)
spdflush;

# ESP Security Association (SA) with manual key specified
add 192.168.1.3 192.168.1.2 esp 15700 -E 3des-cbc
        -A hmac-sha1 "12345678901234567890";
add 192.168.1.2 192.168.1.3 esp 24500 -E 3des-cbc
        -A hmac-sha1 "12345678901234567890";

# Security Policy (SP) Settings
spdadd 192.168.1.2 192.168.1.3 any -P out ipsec
        esp/transport//require;
spdadd 192.168.1.3 192.168.1.2 any -P in ipsec
        esp/transport//require;
```

This configuration uses Encapsulated Security Payload (ESP) for both authentication and encryption, but it can also be configured to use AH for authentication and ESP for encryption in an AH+ESP scenario.

Internet Key Exchange (IKE) is a protocol used to set up a shared session secret from which the encryption keys are derived.

More common configurations that would use the Internet Key Exchange (IKE) daemon (racoon) also require that the following files be configured in addition to 'setkey.conf':

- /etc/racoon/racoon.conf – This file is used to configure the default parameters for both Phase 1 and Phase 2 of IKE negotiation. A default 'racoon.conf' file is provided but only contains parameters for IKE Phase 2, so IKE Phase 1 parameters also need to be added to the default configuration.
- /etc/racoon/psk.txt – This file needs to be configured when Pre-Shared Keys (PSK) are used for authentication. The file 'psk.txt' contains several examples of how the pre-shared keys should be entered into the file.
- /etc/racoon/certs – This is the directory used for storing X.509 certificates for authentication.

***NOTE**: When tunneling other protocols through an IPsec VPN connection, there will also be other configuration files that need to be edited. Also, when using a firewall in combination with an IPsec VPN gateway, an administrator will need to open specific ports that are used to initiate the IPsec connection and carry IPsec traffic.*

3.4 Configuration Migration, Importing Tools

If you already have a running implementation and your are just upgrading, the process of migrating existing settings to a new VPN server is simple. Since the configuration of an IPsec connection is contained within text files, migrating settings is just a matter of copying the configuration files/directories that are being used to the new server.

***NOTE**: Linux and Mac OS X include the same IPsec software (IPsec-Tools), so settings could be easily migrated from Linux to another operating system that uses the IPsec-Tools.*

3.5 Configuration Backup / Restore

IPsec-Tools, which encompasses 'setkey' and 'racoon', and the other software packages that may be used to implement a Linux VPN server do not provide a built-in method for backup of their configuration files, but the tools needed to do so are included on all Linux distributions. Backing up the configuration files is just as simple as migrating the configuration settings as described in the previous section. Since the IPsec configuration is contained within text files, an admin could copy/backup the files to an alternate location on the hard drive or any type of external storage desired (network share, USB mass storage device, etc).

4 Management / User Experience

4.1 Management Interface Options

The management interface for NETKEY/IPsec-Tools is strictly command-line based. There are no web-based or GUI interfaces included with IPsec-Tools. There are some GUI tools that are starting to appear on some distribution such as RHEL 4 and FC5 that include the ability to edit some IPsec settings along with common network settings, but the most flexible method of configuration for NETKEY/IPsec-Tools is on the command-line.

Refer to the chapter *8 References* section of this document for more information on where to obtain official documentation, howto's, etc for setting up a Linux VPN server.

4.2 Setkey Sample configuration

Create a file (`/etc/racoon/setkey.conf`) with the following content.

```
# The next 2 lines delete all existing entries from the SPD and SAD
flush;
spdflush;

# Add the policy
# This policy applies to outbound traffic to the Windows XP machine
spdadd 10.197.173.123/32 10.197.173.124/32 any -P out ipsec
esp/transport/10.197.173.123-10.197.173.124/require;

# This policy applies to inbound traffic from the Windows XP machine
spdadd 10.197.173.124/32 10.197.173.123/32 any -P in ipsec
esp/transport/10.197.173.124-10.197.173.123/require;
```

You invoke the file like this;

```
[root@localhost ~]# setkey -f /etc/racoon/setkey.conf
```

For more information, see the man page (`man setkey`)

4.3 Racoon Sample Configuration

```
path pre_shared_key "/etc/racoon/psk.txt";

             remote anonymous
            {
                    exchange_mode main;

                    proposal {
                            encryption_algorithm 3des ;
                            hash_algorithm md5 ;
                            authentication_method pre_shared_key ;
                            dh_group 2 ;
                    }
            }

            sainfo anonymous
            {
                    lifetime time 28800 sec ;
                    encryption_algorithm 3des ;
                    authentication_algorithm hmac_md5 ;
                    compression_algorithm deflate ;
            }
```

Create a file (`/etc/racoon/racoon.conf`) with the above content

For more information, see the man page (`man racoon`)

4.4 Racoonctl

racoonctl is used to control racoon operation, if IPsec-Tools was configured with adminport support. Communication between racoonctl and racoon is done through a UNIX socket. Below is an example of racoonctl being used as a remote access client.

Start the VPN using racoonctl:

```
# racoonctl vc -u username vpn-gateway.example.net
```

Where username is your login, and vpn-gateway.example.net is the DNS or IP address of the VPN gateway. racoonctl will prompt you for the password.

In order to disconnect from the VPN, do this:

```
# racoonctl vd vpn-gateway.example.net
```

NOTE: *Any of the daemons that are used need to be added to the initialization scripts in the /etc directory in Linux, so that their settings can be loaded and services started each time the system is rebooted.*

4.5 Management Interface Target User

Management of IPsec connections on Linux requires previous Linux experience and a working knowledge of IPsec. In order to configure and use IPsec on Linux, it is assumed that an administrator has previous experience with Linux and the ability to navigate to and edit files on the command-line. Also, since there are not many default options or configuration files provided, an administrator would be required to have a good understanding of various IPsec-related technologies. This includes but is not limited to knowledge of the following subjects:

- Authentication Header (AH)
- Encapsulating Security Payload (ESP)
- Ciphers/Encryption/One-way Hash Algorithms (e.g. HMAC-SHA1, HMAC-MD5, AES, 3DES, RSA, etc)
- IPsec Modes (tunnel vs. transport mode)
- IPsec Security Associations (SA)
- IPsec Security Policies (SP)
- Internet Key Exchange (IKE)
- Pre-shared Keys (PSK), X.509 Certificates, etc.
- Firewall Rules (when a firewall is used in conjunction with an IPSec VPN Gateway)

4.6 Configuration file security

There is no management application distributed with IPsec-Tools that provides an ability to control admin roles and access rights. The only way to change access rights to IPsec-Tools configuration files is through normal Linux file system commands. All of the configuration files should be set so that they are only visible to the local admin (root user). This can be accomplished by issuing the following command, for example, on each configuration file that is being used:

```
# chmod 600 /path/to/config.file
```

NOTE: *The configuration files for IPsec-Tools contain keys and other important data, so it is important that only an administrator of the local machine has access to the files.*

5 VPN Server

5.1 VPN – A Description

A Virtual Private Network is a private network that makes use of a public network, frequently the internet. Wikipedia describes VPN as follows; (Edited for the sake of condensing, see wikipedia for the complete entry for VPN)

"Virtual private networks can be a cost effective and secure way for different corporations to provide users access to the corporate network and for remote networks to communicate with each other across the Internet. VPN connections are more cost-effective than dedicated private lines; usually a VPN involves two parts: the protected or "inside" network, which provides physical and administrative security to protect the transmission; and a less trustworthy, "outside" network or segment (usually through the Internet). Generally, a firewall sits between a remote user's workstation or client and the host network or server. As the user's client establishes the communication with the firewall, the client may pass authentication data to an authentication service inside the perimeter. A known trusted person, sometimes only when using trusted devices, can be provided with appropriate security privileges to access resources not available to general users."

5.2 Remote Access Connectivity

Linux can be configured for remote access/road warrior scenarios as either the server or client. The entire process of setting up a Linux VPN server is not an easy or well documented task. As noted earlier, an administrator would be required to know about many different technologies to properly setup a Linux VPN server for remote access. Adding to the difficulty is all the different combinations of how a Linux server could be setup depending on what clients need to connect to it. A Linux VPN server can be setup to allow remote access in the following configurations:

- IPsec Only
  - This remote access setup can be used when a Linux client based on racoon or a 3rd party client (i.e. Cisco VPN Client) is used to connect to a Linux VPN server.
- L2TP/IPSEC
  - This remote access setup is typically needed when the built-in Windows VPN client is used to connect to a Linux VPN server.
  - L2TP and PPP daemons must also be configured in addition to the IPsec-Tools in order for the built-in Windows clients to connect to the server.

Each of these remote access scenarios is described in more detail in each of the following sub-sections. Since there are many software components and configurations possible, it is difficult to specify how many steps would be required for a particular setup.

5.2.1 *IPsec Only - Linux Server Remote Access Setup*

In this type of scenario, IPsec is used to create a VPN connection without having to use any other tunneling protocols such as L2TP. Here are some advantages and disadvantages of this approach over using L2TP/IPsec:

Advantages:

- Easier to setup on the server
- Less packet overhead

Disadvantages:

- 3rd party clients are required in Windows connect to an IPsec-only server. The native Windows client only supports L2TP/IPsec (not IPsec-only).

5.2.2 IPSec only server – example setup

The example below outlines the general steps required to setup a Linux server for remote access using IPsec-only. For this example, IPsec + Extended Authentication Mode (XAuth) + Hybrid authentication is shown as opposed to PSK's or X.509 certificates. XAUTH provides another level of authentication by allowing the IPSEC Gateway to request additional authentication from users before allowing access to VPN. When using hybrid authentication, the Linux VPN server authenticates to the client using a certificate and the client authenticates to the server with a username and password. Here are the steps:

- Configuration of security policies (SP) and security associations (SA) in /etc/setkey.conf is NOT required. Since most remote clients will have dynamically allocated IP addresses, the security policies cannot be defined prior to a VPN connection. With remote access clients, 'racoon' is able to automatically generate a policy during Phase 2 IKE negotiation.

***NOTE:** An administrator may still need to configure /etc/setkey.conf if the Linux VPN server is being utilized for more than just remote access clients.*

- Configure automatic key negotiation and security associations (SA) through the IKE daemon, racoon, in /etc/racoon/racoon.conf. An example hybrid authentication 'racoon.conf' file which authenticates the client against the local system user database would be similar to the following:

```
# Define the path location for the server certificate and key.
# Another common directory for certificate storage is
# /etc/racoon/certs.
path certificate "/etc/openssl/certs";

# Disable 'racoonctl' support
listen {
        adminsock disabled;
}

# IKE Phase 1 Setup
remote anonymous {
        exchange_mode aggressive, main;
        certificate_type x509 "server.crt" "server.key";
        my_identifier asn1dn;
        proposal_check claim;
        generate_policy on;
        nat_traversal on;
        dpd_delay 20;
        ike_frag on;
        proposal {
                encryption_algorithm aes;
                hash_algorithm sha1;
                authentication_method hybrid_rsa_server;
                dh_group 2;
        }
}
```

```
# Define network information for ISAKMP mode configuration
mode_cfg {
        network4 10.99.99.0;
        pool_size 255;
        netmask4 255.255.255.0;
        auth_source system;
        dns4 10.0.12.1;
        wins4 10.0.12.1;
        banner "/etc/racoon/motd";
        pfs_group 2;
}

# IKE Phase 2 Setup
sainfo anonymous {
        pfs_group 2;
        lifetime time 1 hour;
        encryption_algorithm aes;
        authentication_algorithm hmac_sha1;
        compression_algorithm deflate;
}
```

- Install server certificate and key in /etc/openssl/certs, /etc/racoon/certs, or any other directory (just make sure the correct path is reflected in 'racoon.conf')
- Configure Netfilter/IPTables firewall to allow ESP/AH/IPCOMP protocols, IKE communication, and NAT-Traversal.

***NOTE**: Refer to the 'Firewall Capabilities' section of this document for more information on firewall configuration.*

5.2.3 L2TP/IPsec - Linux Server Remote Access Setup

In this type of scenario, L2TP is tunneled through IPsec to create a VPN connection. Here are some advantages and disadvantages of this approach over using IPsec-only.

Advantages:

- Setup and installation is easier on the client side when using Windows
- There is a free, native VPN client in Windows that supports L2TP/IPsec

Disadvantages:

- Higher packet overhead since L2TP and PPP headers are needed in addition to IPsec packet headers.
- Requires Certificates and/or Public Key Infrastructure
- Fewer features (e.g. AES is not supported in the Windows VPN client)

5.2.3.1 Linux L2TP/IPSec server – example setup

The general steps required for setting up a Linux L2TP/IPsec server are listed below. They are similar to the steps of creating an IPsec-only server with the additional requirement that L2TP and PPP daemons be configured.

- Configure security policies (SP) in /etc/setkey.conf. Since the IKE daemon will automatically setup the security associations (SA), only the SP's need to be defined in 'setkey.conf'.
- Configure automatic key negotiation and security associations (SA) through the IKE daemon, racoon, in /etc/racoon/racoon.conf. In an L2TP/IPsec scenario the native Windows client, racoon needs to be configured to use X.509 certificates.

- Install the server certificate and key in /etc/openssl/certs, /etc/racoon/certs, or any other directory (just make sure the correct path is reflected in 'racoon.conf')
- Install and configure a PPP daemon. Most distributions install PPPD by default. Consult the official PPPD documentation at http://www.samba.org/ppp/documentation.html for more information on configuration options.
- Install and configure an L2TP daemon. There are at least eight different L2TP daemons available under Linux (none of which are usually installed by default on most distributions), but L2TPD is probably the one that is most commonly used even though the project is not actively maintained anymore. Once installed, L2TPD is configured through /etc/l2tpd/l2tpd.conf. Documentation on the L2TPD configuration file is sparse. Refer to the following link for an example configuration: http://www.jacco2.dds.nl/networking/freeswan-l2tp.html#L2TPoconfigLinux
- Configure Netfilter/IPTables firewall to allow ESP/AH/IPCOMP protocols, IKE communication, and NAT-Traversal.
    - NOTE: Refer to the 'Firewall Capabilities' section of this document for more information on firewall configuration.

5.2.4 Building Racoon for Remote Access

In order for racoon to provide the necessary functionality for remote access, it needs to be configured and compiled with specific options enabled (which may or may not be enabled depending on the specific Linux distribution being used). Even though RedHat and Fedora come supplied with racoon, it would be beneficial to download and install the latest version since racoon is in active development and updated frequently. When compiling racoon from source, not all of the necessary options for remote access are enabled by default so it is important to ensure that they are enabled before configuring or setting up a VPN connection.

The following is an example of the compile options that need to be enabled when racoon is used as a server in a remote access scenario:

```
# ./configure --enable-natt --enable-frag --enable-hybrid --enable-dpd \

--with-libradius --sysconfdir=/etc/racoon
```

The following is an example of the compile options that need to be enabled when racoon is used as a client in a remote access scenario:

```
# ./configure --enable-natt --enable-frag --enable-hybrid --enable-dpd --enable-
adminport --sysconfdir=/etc/raccoon --localstatedir=/var
```

The following is a list of remote access related options that can be compiled into racoon along with a brief description of each one:

- *--with-libradius=[DIR_PATH]* - Used to enable RADIUS support for remote access authentication.
- *--with-libpam=[DIR_PATH]* – Used to enable PAM support for remote access authentication.
- *--enable-gssapi* – This enables GSS-API authentication.

- *--enable-hybrid* – This enables hybrid authentication (XAuth and mode-config support) for remote access.
- *--enable-frag* – This enables IKE fragmentation payload support.
- *--enable-dpd* – This enables dead peer detection.
- *--enable-natt* – This enables IPsec NAT-T (NAT-Traversal)
- *--enable-adminport* – This enables support for 'racoonctl' which is a command-line utility used to control the racoon server. The utility is typically used when racoon is used as a client in a remote access scenario.

5.3 Site to Site Connectivity

Creating a site-to-site VPN in Linux is less complex than creating a Linux VPN server for remote access. There are only two servers/gateways involved (running IPsec-Tools) and an administrator does not need to worry about compatibility with multiple clients like they would in a remote access scenario.

A site-to-site VPN connection can be created with IPsec-Tools using one of the following configurations:

- *IPsec + Manually Keyed Connection*
- *IPsec + Pre-Shared Secret Keys*
- IPsec + X.509 Certificates

5.3.1 IPsec + Manually Keyed Connection

Using a manually keyed connection in a site-to-site scenario is the simplest of the three options listed above, because there is no need to use the IKE daemon, racoon. The only file that needs to be configured on each gateway as far as IPsec-Tools is concerned is /etc/setkey.conf. A basic configuration file will be as simple as the following:

```
#!/sbin/setkey -f
flush;
spdflush;

# Security Policy (SP) Definitions
spdadd 10.0.1.0/24 10.0.2.0/24 any -P out ipsec
      ah/tunnel/172.16.0.1-172.16.0.2/require ;
spdadd 10.0.2.0/24 10.0.1.0/24 any -P in ipsec
      ah/tunnel/172.16.0.2-172.16.0.1/require ;

# Security Association (SA) Definitions
add 172.16.0.1 172.16.0.2 esp 0x10003
      -m tunnel
      -A hmac-sha1 "this is the test key";
add 172.16.0.2 172.16.0.1 esp 0x10004
      -m tunnel
      -E 3des-cbc
      -A hmac-sha1 "this is the test key";
```

Keep the following points in mind when creating the setkey.conf file on each gateway:

- Adding inbound and outbound security associations (SA's).
    - The following parameters need to be configured for each SA:
        - Security protocol
            - AH + ESP can be used for authentication and encryption

            - Alternatively, ESP can also be used for both authentication and encryption
        - Security Parameter Index (SPI).
            - A unique SPI number needs to be defined for each SA
        - IP Destination Addresses
            - Each SA needs to contain a source and destination IP address (the external address of each gateway).
        - Authentication algorithm
        - Encryption algorithm
            - An encryption algorithm should only be defined if ESP is being used for an SA since AH can only be used for authentication.
    - **NOTE:** The SA's will be exactly the same on each gateway.
- Adding inbound and outbound security policies (SP's)
    - The following parameters need to be configured for each SP:
        - Tunnel/transport mode - In the case of site-to-site VPN's, tunnel mode should be configured.
        - Gateways - each gateway in the site-to-site VPN needs to be identified.
        - Networks - the networks behind each gateway need to be identified.
        - Policy - the policy directive allows an admin to define how the network traffic between should be handled between the two gateways/networks defined in the security policy:
            - ipsec - when the policy is set to 'ipsec' the network traffic defined by an SP is encapsulated by the IPsec protocol
            - none - when the policy is set to 'none' the network traffic defined by an SP is not protected by ipsec
            - discard - when the policy is set to 'discard' the network traffic defined by an SP is dropped/discarded

NOTE: *The SP's will NOT be the same on each gateway; they will be mirrored. In other words, the inbound and outbound SP definitions in setkey.conf will be swapped on each gateway.*

5.3.2 IPsec + Pre-Shared Secret Keys (PSK)

Using PSK's requires the use of the IKE daemon, racoon, so the racoon.conf file needs to be configured in addition to setkey.conf. The following configuration steps should be taken to create a site-to-site VPN using pre-shared secret keys:

- Configure the security policies in setkey.conf. As noted earlier, the security policies will be mirrored on each gateway. Security associations, on the other hand, will not need to be defined because they are automatically created during racoon's IKE negotiation phases.
- Configure the pre-shared keys on each gateway in /etc/racoon/psk.txt. The default file shows examples of how to enter the pre-shared keys.
- Configure racoon to use the pre-shared keys by entering the following directive into racoon.conf:

```
path pre_shared_key "/etc/racoon/psk.txt";
```

5.3.3 IPsec + X.509 Certificates

Using X.509 certificates also requires the use of racoon. The following configuration steps should be taken to create a site-to-site VPN using certificates:

- Configure the security policies in setkey.conf. Again, the security policies will be mirrored on each gateway.
- Generate and install public/private keys on each gateway using OpenSSL
    - Racoon can be configured to look for the keys in any directory but /etc/openssl/certs and /etc/racoon/certs are the most common storage locations for certificates.
    - Copy the public/private keys generated on each gateway to the certificate directory, and then copy/install the public keys on each gateway to the other gateway.
        - Gateway 1 will have the following keys installed in the certificate directory
            - gateway1_public_key
            - gateway1_private_key
            - gateway2_public_key
        - Gateway 2 will have the following keys installed in the certificate directory
            - gateway2_public_key
            - gateway2_private_key
            - gateway1_public_key
- Configure racoon to use the pre-shared keys by entering the following directive into racoon.conf:

```
path certificate "/etc/racoon/certs";
```

5.4 Firewall Capabilities

Netfilter/IPTables is an advanced and powerful packet filtering framework for the Linux 2.6 and 2.4 Kernels. A full teardown of the Linux Fire walling capabilities is beyond the scope of this document so we will be showing just a high level overview of features.

Netfilter/IPTables Features:

- Stateless and Statefull packet filtering (IPv4, IPv6)
- NAT/NAPT (Masquerading, Port Forwarding)
- Extensible infrastructure with multiple layers of API's for 3rd party extensions
- Large number of plugins/modules available
- QoS and policy routing
- Packet manipulation

Some Advanced Features:

- Specifying multiple ports in one rule
- Load balancing
- Restricting the number of connections
- Maintaining a list of recent connections to match against
- Matching against a string in a packet's data payload
- Time-based rules
- Setting transfer quotas
- Packet matching based on TTL values

Configuring Netfilter/IPtables is done via the command line with the iptables command. Though there are many GUI and Web based utilities, not all give you access to all features and should be looked at carefully before selecting one. Freshmeat.net (https://www.freshmeat.net) lists 172 projects based on or related to IPtables ranging from automatic script generators to full blown GUI front ends.

Below is minimal list of rules what are needed for IPSec to work. In this example the remote gateway is 192.168.0.1 and my external IP is 10.0.0.1.

```
# iptables -A INPUT -p udp -s 192.168.0.1 -d 10.0.0.1 --dport 500 --j ACCEPT
# iptables -A INPUT -p udp -s 192.168.0.1 -d 10.0.0.1 --dport 4500 --j ACCEPT
# iptables -A INPUT -p esp -s 192.168.0.1 -d 10.0.0.1 -j ACCEPT
# iptables -A INPUT -p ah -s 192.168.0.1 -d 10.0.0.1 -j ACCEPT

# iptables -A OUTPUT -p udp -s 10.0.0.1 -d 192.168.0.1 --sport 500 --j ACCEPT
# iptables -A OUTPUT -p udp -s 10.0.0.1 -d 192.168.0.1 --sport 4500 --j ACCEPT
# iptables -A OUTPUT -p esp -s 10.0.0.1 -d 192.168.0.1 -j ACCEPT
# iptables -A OUTPUT -p ah -s 10.0.0.1 -d 192.168.0.1 -j ACCEPT
```

IPsec-Tools also has support for NAT-T and is configured in the racoon.conf file.

From the racoon.conf manual page ;

> *"NAT-T allows one or both peers to reside behind a NAT gateway (i.e., doing address- or port-translation). Presence of NAT gateways along the path is discovered during phase 1 handshake and if found, NAT-T is negotiated. When NAT-T is in charge, all ESP and AH packets of a given connection are encapsulated into UDP datagrams (port 4500, by default)."*

See Appendix for sample natt racoon.conf file.

5.5 VLAN Capabilities

A VLAN or Virtual LAN is a way of creating independent logical networks that reside in a physical network. A VLAN can be used to have a network of computers behave as if they are connected to the same wire. Even though they can be physically connected to different segments of a LAN.

Linux has been able to connect to VLAN trunks since the 2.4.14 kernel, and is supported in the 2.6 versions.

Configuring VLAN's under Linux is similar to configuring regular Ethernet interfaces. Using the vconfig command we attach each VLAN to a physical device.

```
# vconfig add eth0 2

# vconfig add eth0 3
```

Once a virtual interface is defined we can use the standard utilities, such as ifconfig and route, and it behaves as any other interface.

5.6 Access Control

Access control to specific servers or resources in the network is not directly provided by NETKEY/IPsec-Tools. However, since IPsec-Tools supports Remote Aithentication Dial In User Service (RADIUS) authentication and accounting, RADIUS could theoretically be used to limit access to specific resources. The specifics of how this could be accomplished are beyond the scope of this document and would require a fair amount of custom scripting with RADIUS.

5.7 Policy Management

The IPsec security policy database (SPD) is maintained and managed with the 'setkey' utility included with IPsec-Tools. The command-line utility, setkey, provides the ability to change/add/delete security policies on-the-fly and also load security policy settings from the setkey.conf file (usually located in /etc/setkey.conf). The security policies defined in setkey.conf are fairly straightforward, but managing a large set of security policies could be time consuming since they need to be edited by hand in setkey.conf. Further, there are not any central management utilities provided by IPsec-Tools, so if multiple servers are being used for a VPN solution the security policies will need to be maintained separately on each individual server.

Here is an example of a security policy defined with setkey:

```
spdadd 192.168.1.10 192.168.1.20 any -P out ipsec
       esp/transport//require;
```

This policy says that outbound traffic going from 192.168.1.10 to 192.168.1.20 is required to be protected by IPsec in transport mode using ESP.

The definition of a security policy should conform to the following usage rules:

```
spdadd src_range dst_range upperspec policy
       [Add SPD entry]
```

- ***src_range*** and ***dst_range*** - These define what network traffic should be secured by IPsec. Each of these can be an IPv4/6 address or address range. TCP/UDP port numbers can also be specified within brackets after the address/range.
- ***upperspec*** - This can be any upper level protocol listed in /etc/protocols.
- ***policy*** - The policy takes the following form:

```
-P direction [priority_specification] [discard|none|ipsec]
  protocol/mode/src-dst/level
```

  - *direction* - This specific the direction of the traffic (outbound, inbound, or forwarded traffic)
  - *priority_specification* - This specifies a priority for the policy within the SPD.
  - *discard|none|ipsec* - This specifies that the network traffic should be discarded, continue unmodified, or be encapsulated by ipsec, respectively.
  - *protocol* - This will be either ah, esp, or ipcomp.
  - *mode* - The mode will be either transport or tunnel.

- *src-dst* - This specifies the server/gateway IP address that will be used to secure traffic in tunnel mode. This addresses are omitted when transport mode is specified in the policy.
- *level* - This specifies the how the kernel handles the creation/negotiation of an SA.

NOTE: Refer to the Documentation section for more information on getting help with security policies.

5.8 Certificate Authority Support

A digital certificate contains information that identifies the user or a device. (name, serial number etc) It also contains a copy of the entity's public key.

IPsec-Tools package do not come with a Certificate Authority or any CA management tools, however it is common to use the openssl command to create and manage a Certificate Authority. X.509 certificates are fully supported.

5.9 Server Logs, Reporting, Accounting Capabilities

RHEL4 and Fedora Core 5 use the sysklogd package for logging of messages received from programs and facilities. Sysklogd provides two system utilities: syslogd for providing system logging and klogd for kernel message trapping. Support for both internet and unix domain sockets enables support for both local and remote logging.

- On FC5 syslogd logs racoon logs to /var/log/messages
- Logs Informational messages, warnings, and errors and can be customized via syslog.conf.
- You can also use the setkey utility for viewing the Security Association Database and the Security Policy Database.

  To view the SAD

  # setkey –D

  To view the SPD

  # setkey –DP

There are many 3rd party packages for viewing, managing and reporting on syslog logs, too many to list here. It is left up to the administrator to manage system logs and generate reports.

5.10 Troubleshooting, Diagnostics, Monitoring

Due to the complexity of the IKE and IPSec protocols troubleshooting can be a challenge. There are several tools available to the Administrator to help monitor and diagnose a problem.

- Packet capture programs
  - Tcpdump - Installed by default on FC5 and RHEL4
  - Ethereal - FC5 and RHEL4 come with RPM packages available to install.

- Verbose Logging
    - (racoon -v -ddd -f /etc/racoon/racoon.conf -l /tmp/racoon.log)
- Viewing SAD/SPD with setkey and racoonctl
- Monitoring may also be facilitated with other open-source utilities, such as configuring and SNMP server on the system.

5.11 NAT Capabilities

There are two types of Network Address Translation that Iptables supports: source NAT and destination NAT.

Source NAT is done just before a packet is finally sent out. This means that on the Linux box itself (routing, packet filtering) will see the packet unchanged. Source NAT is also referred to as Masquerading.

Destination NAT is done just as the packet comes in; this means that anything else on the Linux box itself (routing, packet filtering) will see the packet going to its "real" destination. Destination NAT is sometimes called port forwarding, and can also be used to redirect packets.

Configuring NAT is fairly straight forward. Here is a source NAT example where all traffic from 192.168.0.0/24 that is outgoing on eth1 will get it's source changed to 10.1.1.1.

```
# iptables -t nat -A POSTROUTING -s 192.168.0.0/24 -o eth1 -j SNAT -to 10.1.1.1
```

Source NAT is specified using `-j SNAT', and the `--to-source' option specifies an IP address, a range of IP addresses, and an optional port or range of ports (for UDP and TCP protocols only).

Change source addresses to 10.1.1.1.

```
# iptables -t nat -A POSTROUTING -o eth1 -j SNAT --to 10.1.1.1
```

Change source addresses to 10.1.1.1, 10.1.1.2 or 10.1.1.3

```
# iptables -t nat -A POSTROUTING -o eth1 -j SNAT --to 10.1.1.1-10.1.1.3
```

Change source addresses to 10.1.1.1, ports 1-1023

```
# iptables -t nat -A POSTROUTING -p tcp -o eth1 -j SNAT --to 10.1.1.1:1-1023
```

- Also See Firewall Capabilities.
- Also See Firewall Traversal.

5.12 XAuth, AAA, RADIUS Support

For remote access clients, racoon (the IKE daemon) supports the following:

- ***XAuth*** - Xauth is only supported when used in combination with hybrid authentication. Plain XAuth is not supported by racoon, according to a FAQ supplied with the sources files of the most recent version of racoon.

  - The following authentication sources are supported in racoon when hybrid authentication is used:
    - *System* - Client authentication is performed against the Unix user database.
    - *RADIUS* - Client authentication is performed against a RADIUS server
    - *PAM* - Client authentication is performed using against PAM (Pluggable Authentication Modules)
- ***AAA***
  - *Authentication* - As mentioned above, racoon can be configured for System, RADIUS, or PAM authentication.
  - *Accounting* - Either RADIUS or PAM can be used in racoon for accounting purposes.
  - *Authorization* - Authorization is possible when using RADIUS. See the FreeRADIUS Teardown document for more information.

    NOTE: *In racoon, RADIUS accounting requires RADIUS authentication and PAM accounting requires PAM authentication.*
- *RADIUS* - supported as described above.

5.13 Scalability

In the paper "Scalability Implications of Virtual Private Networks" [2] VPN scalability implications were broken down into three categories *memory consumption, processing power, and management load.* Memory consumption and processing power depend on the number of tunnels established as every tunnel will require session keys, certificates, routing information and other state information to be maintained. Adding more processing power and memory can be done relatively easily as a Linux based VPN server is not tied to any specific form factor. Adding another server and load balancing connections is also a way to scale if the load becomes greater than one server can handle.

Management load is where scalability becomes a big issue. Maintaining existing VPN tunnels, updating configuration files, editing routing information, distribution of security keys/certificates and user management for each node becomes a huge challenge since no central management infrastructure is provided. A setup to maintain 1 server with 10 remote locations does not scale to support multiple load balanced VPN servers with 10,000 remote clients. A custom solution would have to be created or developed.

There are some great papers on the subject of Linux VPN's and Scalability

[1]http://mia.ece.uic.edu/~papers/publications/commMag-2004.pdf

[2]http://ieeexplore.ieee.org/iel5/35/21598/01000229.pdf?tp=&isnumber=21598&arnumbe=1000229&type=ref

6 VPN Client

6.1 3rd Party VPN Client Support

Windows 2000/XP has built in support for L2TP/IPSec connections.

Windows 98/ME will need to download and install a MSL2TP client.

Though pptp is not recommended there is a GUI configuration tool for connection to a pptp server from Linux.

http://pptpclient.sourceforge.net/howto-fedora-core-5.phtml

6.2 End User Setup

Setting up the end user can range from complex to fairly easy depending on how the server is setup. Servers with L2TP/IPSec support make end user setup on Windows 2000/2003/XP clients straight forward to setup and can for the most part follow the VPN setup wizard.

7 Appendix – Racoon.conf Sample natt

```
# $Id: racoon.conf.sample-natt,v 1.3.10.1 2005/04/18 11:10:55 manubsd Exp $
# Contributed by: Michal Ludvig <mludvig@suse.cz>, SUSE Labs

# This file can be used as a template for NAT-Traversal setups.
# Only NAT-T related options are explained here, refer to other
# sample files and manual pages for details about the rest.

path include "/etc/racoon";
path certificate "/etc/racoon/cert";

# Define addresses and ports where racoon will listen for an incoming
# traffic. Don't forget to open these ports on your firewall!
listen
{
        # First define an address where racoon will listen
        # for "normal" IKE traffic. IANA allocated port 500.
        isakmp 172.16.0.1[500];

        # To use NAT-T you must also open port 4500 of
        # the same address so that peers can do 'Port floating'.
        # The same port will also be used for the UDP-Encapsulated
        # ESP traffic.
        isakmp_natt 172.16.0.1[4500];
}


timer
{
        # To keep the NAT-mappings on your NAT gateway, there must be
        # traffic between the peers. Normally the UDP-Encap traffic
        # (i.e. the real data transported over the tunnel) would be
        # enough, but to be safe racoon will send a short
        # "Keep-alive packet" every few seconds to every peer with
        # whom it does NAT-Traversal.
        # The default is 20s. Set it to 0s to disable sending completely.
        natt_keepalive 10 sec;
}

# To trigger the SA negotiation there must be an appropriate
# policy in the kernel SPD. For example for traffic between
# networks 192.168.0.0/24 and 192.168.1.0/24 with gateways
# 172.16.0.1 and 172.16.1.1, where the first gateway is behind
# a NAT which translates its address to 172.16.1.3, you need the
# following rules:
# On 172.16.0.1 (e.g. behind the NAT):
#     spdadd 192.168.0.0/24 192.168.1.0/24 any -P out ipsec \
#            esp/tunnel/172.16.0.1-172.16.1.1/require;
#     spdadd 192.168.1.0/24 192.168.0.0/24 any -P in ipsec \
#            esp/tunnel/172.16.1.1-172.16.0.1/require;
# On the other side (172.16.1.1) either use a "generate_policy on"
# statement in the remote block, or in case that you know
# the translated address, use the following policy:
#     spdadd 192.168.1.0/24 192.168.0.0/24 any -P out ipsec \
#            esp/tunnel/172.16.1.1-172.16.1.3/require;
#     spdadd 192.168.0.0/24 192.168.1.0/24 any -P in ipsec \
#            esp/tunnel/172.16.1.3-172.16.1.1/require;

# Phase 1 configuration (for ISAKMP SA)
remote anonymous
{
       # NAT-T is supported with all exchange_modes.
       exchange_mode main,base,aggressive;

       # With NAT-T you shouldn't use PSK. Let's go on with certs.
       my_identifier asn1dn;
       certificate_type x509 "your-host.cert.pem" "your-host.key.pem";

       # This is the main switch that enables NAT-T.
       # Possible values are:
       #   off - NAT-T support is disabled, i.e. neither offered,
       #         nor accepted. This is the default.
       #    on - normal NAT-T support, i.e. if NAT is detected
       #         along the way, NAT-T is used.
       # force - if NAT-T is supported by both peers, it is used
       #         regardless of whether there is a NAT gateway between them
       #         or not. This is useful for traversing some firewalls.
       nat_traversal on;

       proposal {
              authentication_method rsasig;
              encryption_algorithm 3des;
```

```
                hash_algorithm sha1;
                dh_group 2;
        }

        proposal_check obey;
}

# Phase 2 proposal (for IPsec SA)
sainfo anonymous
{
        pfs_group 2;
        lifetime time 12 hour;
        encryption_algorithm 3des, rijndael;
        authentication_algorithm hmac_sha1;
        compression_algorithm deflate;
}
```

8 References

References to the sources of information used in creating this document.

VLANs on Linux by Paul Frieden	http://www.linuxjournal.com/article/7268
Netfilter FAQ	http://www.netfilter.org/documentation/FAQ/netfilter-faq.htm
IPsec-Tools	http://ipsec-tools.sourceforge.net/
Gentoo wiki	http://gentoo-wiki.com/HOWTO_IPSEC
IPSec Linux HOWTO	http://www.ipsec-howto.org/
VPN Labs	http://www.vpnlabs.com/
Linux HA	http://www.linux-ha.org/
Freeswan HA	http://www.freeswan.ca/docs/HA/HA_VPNS_With_FreeSWAN.html