

Shrew Soft VPN Client Administrators Guide



Version 2.1.6

Copyright 2010
Shrew Soft Inc.

This guide contains the following chapters:

1. [Introduction](#)
2. [Features and Compatibility](#)
3. [System Requirements](#)
4. [Known Issues](#)
5. [IPsec Overview](#)
6. [Using the VPN Client](#)

Introduction

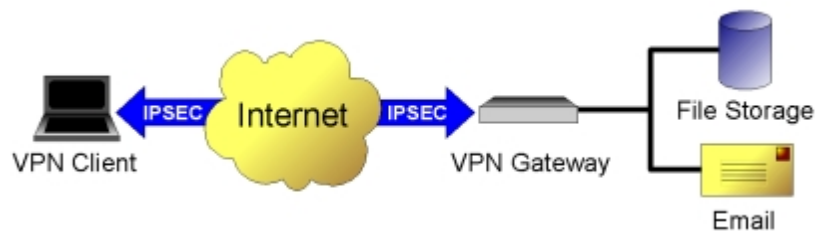
This chapter provides a general overview of the Shrew Soft VPN Client and its terms of use.

List of available topics:

1. [Overview](#)
2. [License](#)
3. [Disclaimer](#)

Overview

This guide describes how to configure and operate the Shrew Soft VPN Client with an open source VPN Gateway. By tunneling traffic between the VPN Client and the VPN Gateway, the host is able to access private network resources in a manner which is secure even when connecting from an insecure public network.



This functionality is provided by implementing the IPsec Protocol standard which is used by a wide variety of both commercial and open source operating systems.

Many commercial VPN Gateways are available on the market. Most of these products are bundled with proprietary VPN Client software that is designed to communicate with a specific gateway device. While most major open source operating systems have had support for basic IPsec functionality for some time, they have lacked a sophisticated IKE daemon and kernel support for protocol extensions that would be required to properly support IPsec Client connectivity. Luckily this is no longer the case. Recent improvements to the [IPsec Tools](#) software and added kernel support for features such as NAT Traversal have enabled open source operating systems such as [Linux](#), [FreeBSD](#) or [NetBSD](#) to be considered as a viable alternative to expensive commercial VPN Gateway solutions. The Shrew Soft VPN Client offers a complimentary Windows IPsec implementation that can be used to communicate with these gateways.

Windows License

The Shrew Soft Client for Windows is free for both commercial and private use. Please read below for complete license details.

Shrew Soft VPN Client For Windows License

Copyright (c) 2007
Shrew Soft Inc.
All rights reserved.

Redistribution in binary form is permitted for both personal and commercial use provided that the following conditions are met:

1) Modification or removal of any portion of this software package prior to redistribution is prohibited. This may include but is not limited to any binary programs, loadable modules, documentation or license agreement files.

2) This software package must not be represented as your own product. If you advertise the availability of this software package or the potential use of this software package in concert with another product or an affiliate's product, you agree to also advertise that the software package is an asset of the legitimate copyright holder, Shrew Soft, Inc .

3) Only a nominal fee may be charged to cover the cost of media and/or delivery fees

for providing a reproduced machine-readable copy of this software package.

4) A third party may not be charged any fee associated with the installation, support or continued operation of this software package regardless of whether or not the software was provided by you or an affiliate.

Waiver; Construction. Failure by Licensor to enforce any provision of this License will not be deemed a waiver of future enforcement of that or any other provision. Any law or regulation which provides that the language of a contract shall be construed against the drafter will not apply to this License.

Severability. If for any reason a court of competent jurisdiction finds any provision of this License, or portion thereof, to be unenforceable, that provision of the License will be enforced to the maximum extent permissible so as to affect the economic benefits and intent of the parties, and the remainder of this License will continue in full force and effect.

Dispute Resolution. Any litigation or other dispute resolution between You and Licensor relating to this License shall take place in the Western District of Texas, and You and Licensor hereby consent to the personal jurisdiction of, and venue in, the state and federal courts within that District with respect to this License. The application of the United Nations Convention on Contracts for the International Sale of Goods is expressly excluded.

Entire Agreement; Governing Law. This License constitutes the entire agreement between the parties with respect to the subject matter hereof. This License shall be governed by the laws of the United States and the State of Texas, except that body of Texas law concerning conflicts of law.

Termination. This License and the rights granted hereunder will terminate automatically if You fail to comply with terms herein and fail to cure such breach within 30 days of becoming aware of the breach. All sublicenses to the Covered Code which are properly granted shall survive any termination of this License. Provisions which, by their nature, must remain in effect beyond the termination of this License shall survive.

Disclaimer of Warranty. THIS SOFTWARE IS PROVIDED "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL LICENSOR BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

This product includes software developed by the OpenSSL Project for use in the OpenSSL Toolkit. (<http://www.openssl.org/>)

Unix License

The Shrew Soft Client for Unix is distributed under a [OSI](#) approved license. Please read below for complete license details.

Shrew Soft VPN Client For Unix License

Copyright (c) 2007
Shrew Soft Inc.
All rights reserved.

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions

are met:

1. Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.
2. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.
3. Redistributions in any form must be accompanied by information on how to obtain complete source code for the software and any accompanying software that uses the software. The source code must either be included in the distribution or be available for no more than the cost of distribution plus a nominal fee, and must be freely redistributable under reasonable conditions. For an executable file, complete source code means the source code for all modules it contains. It does not include source code for modules or files that typically accompany the major components of the operating system on which the executable file runs.

THIS SOFTWARE IS PROVIDED BY SHREW SOFT INC ``AS IS'' AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, OR NON-INFRINGEMENT, ARE DISCLAIMED. IN NO EVENT SHALL SHREW SOFT INC BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

Disclaimer

This software uses strong cryptography provided by the freely available [OpenSSL](#) Toolkit. For this reason, please read the legal notices below. The second notice is a reproduction of the notice posted on the OpenSSL download page.

Shrew Soft Legal Notice

SHREW SOFT INC WILL NOT BE HELD LIABLE FOR THE VIOLATION OF ANY LAW THAT GOVERNS THE IMPORT/EXPORT OF STRONG CRYPTOGRAPHY SOFTWARE. IT IS YOUR RESPONSIBILITY TO DETERMINE WHICH OF THESE LAWS MAY APPLY TO YOU BEFORE OBTAIN, INSTALL OR USE THIS SOFTWARE.

OpenSSL Legal Notice

This software package uses strong cryptography, so even if it is created, maintained and distributed from liberal countries in Europe (where it is legal to do this), it falls under certain export/import and/or use restrictions in some other parts of the world.

PLEASE REMEMBER THAT EXPORT/IMPORT AND/OR USE OF STRONG CRYPTOGRAPHY SOFTWARE, PROVIDING CRYPTOGRAPHY HOOKS OR EVEN JUST COMMUNICATING TECHNICAL DETAILS ABOUT CRYPTOGRAPHY SOFTWARE IS ILLEGAL IN SOME PARTS OF THE WORLD. SO, WHEN YOU IMPORT THIS PACKAGE TO YOUR COUNTRY, RE-DISTRIBUTE IT FROM THERE OR EVEN JUST EMAIL TECHNICAL SUGGESTIONS OR EVEN SOURCE PATCHES TO THE AUTHOR OR OTHER PEOPLE YOU ARE STRONGLY ADVISED TO PAY CLOSE ATTENTION TO ANY EXPORT/IMPORT AND/OR USE LAWS WHICH APPLY TO YOU. THE AUTHORS OF OPENSSL ARE NOT LIABLE FOR ANY VIOLATIONS YOU MAKE HERE. SO BE CAREFUL, IT IS YOUR RESPONSIBILITY.

Copyright and Trademarks

Some portions of this document have been copied and reformatted versions of the racoon man pages and/or documentation. This material is the property of the respective copyright owners. All product and company names herein may be trademarks of their registered owners.

Features and Compatibility

This chapter describes the features currently offered by the Shrew Soft VPN Client as well as some future planned features. It also contains compatibility information to help you determine if the software can be used with a particular VPN Gateway product.

List of available topics:

1. [Features](#)
2. [Compatibility](#)
3. [Future Versions](#)

Features

This Software implements the IPsec Protocol standard and uses ISAKMP version 1.0 to negotiate security parameters with a VPN Gateway. In addition, it includes support for the XAuth protocol extension for user authentication and the Configuration Exchange extension for automatic client configuration. Please read below for a complete list of supported features.

Firewall Traversal Options

- NAT Traversal (RFC & Draft 00-04 versions)
- NAT Keep Alive
- IKE Fragmentation

Authentications Methods

- Hybrid RSA + XAuth
- Mutual RSA + XAuth
- Mutual PSK + XAuth
- Mutual RSA
- Mutual PSK

Identification Types

- ASN1DN
- FQDN
- UFQDN
- Address
- Key Identifier

Exchange Modes

- Main
- Aggressive
- Configuration (push or pull)
- Quick
- Informational

Phase1 Ciphers

- AES
- Blowfish
- 3DES
- CAST
- DES

Phase1 Hash Algorithms

- MD5
- SHA1

Phase2 Transforms

- ESP-AES
- ESP-Blowfish
- ESP-3DES
- ESP-CAST
- ESP-DES

Phase2 HMAC Algorithms

- HMAC-MD5
- HMAC-SHA1

Phase2 Options

- PFS is supported
- Tunnel mode is supported
- Transport mode is not supported
- Compression is not supported

Basic Configuration Exchange Attributes

- Banner
- Address
- Netmask
- WINS Server
- DNS Server

Advanced Configuration Exchange Attributes

- DNS Default Domain
- Login Banner
- Split Network Include List
- Split Network Exclude List
- Split DNS Domain List
- PFS DH Group

Future Versions

Although the VPN Client supports a rapidly growing feature set, it still lacks some options found in the popular commercial solutions. Most of these features will be added in future releases. Please read below for a brief list.

- Pre-Login Connection Support for AD/Domain Logins
- Client Side Stateful Firewall
- Multi Language Support
- Additional Platform Support

Compatibility

For best results, Shrew Soft recommends the use of a VPN Gateway running Linux, FreeBSD or NetBSD and IPsec Tools version 0.7 or later. The VPN Client has also been reported to work with several commercial VPN Gateways. Please consult the Shrew Soft Online support [wiki](#) for more details.

System Requirements

This chapter describes the system requirements for the Shrew Soft VPN Client by platform.

List of available topics:

1. [Windows Client Install](#)
2. [Unix Client Install](#)

Windows Client Install

This software package should install on any reasonable machine running Windows 2000, Windows XP/XP or Windows Vista with network connectivity via an Ethernet or Dial-up adapter. Testing has only been performed on machines running with the latest service packs installed. For this reason, you are encouraged to keep your operating system up to date when using this software.

To install the Shrew Soft VPN Client, you must be logged into the operating system with an account that has administrative privileges. For normal operation, using any account that has local login access should be sufficient.

Unix Client Install

This software package should install on any reasonable machine running Linux, FreeBSD or NetBSD with network connectivity via an Ethernet or Dial-up adapter. Packages are available for the Ubuntu and Debian Linux distributions. A package is also available in the FreeBSD ports collection.

The source code for Unix platforms is available as release archives from the Shrew Soft VPN Client for Unix [downloads](#) page or from the public subversion repository. After obtaining the source code, please see the README.TXT file which contains updated information on how to compile and install the software.

Known Issues

This chapter describes known issues with the Shrew Soft VPN Client by platform.

List of available topics:

1. [Windows Client Issues](#)
2. [IPsec Tools Issues](#)

Windows Client Issues

Several issues have been identified with using the Windows version of the Shrew Soft VPN Client which could cause problems in certain situations. Here is a brief list:

- Will negotiate but not honor lifetime kilobytes for SAs
- May cause problems if installed with other VPN Clients

All problems will be addressed in future releases.

Ipsec Tools Issues

Some issues have been identified with the IPSec Tools racoon daemon which could potentially cause problems in certain situations. Here is a brief list:

- Will generate inappropriate policies (patch committed post 0.7)

Shew Soft has made several contributions to the IPSec Tools project and will continue to submit patches for problems related to Client Operation.

IPSEC Overview

This chapter provides a general overview of IPsec services.

List of available topics:

1. [IP Security](#)
2. [The IKE Protocol](#)
3. [The ESP and AH Protocols](#)
4. [Client Gateways](#)
5. [Problems with IPsec](#)

IP Security

As the name suggests, the IP Security ("IPsec") protocol suite is used to provide encryption and message authentication for IP communications. These protocols are defined by the [IETF](#) and published as [RFC](#) standards or working drafts. The Shrew Soft VPN Client makes every attempt to adhere to the following IPsec and key exchange related documents.

Standards

- [RFC 2367](#) - PF_KEY Key Management API, Version 2
- [RFC 2393](#) - IP Payload Compression Protocol (IPComp)
- [RFC 2394](#) - IP Payload Compression Using DEFLATE
- [RFC 2401](#) - Security Architecture for the Internet Protocol
- [RFC 2402](#) - IP Authentication Header (AH)
- [RFC 2406](#) - IP Encapsulating Security Payload (ESP)
- [RFC 2407](#) - The Internet IP Security Domain of Interpretation for ISAKMP
- [RFC 2408](#) - Internet Security Association and Key Management Protocol (ISAKMP)
- [RFC 2409](#) - The Internet Key Exchange (IKE)
- [RFC 2411](#) - IP Security Document Roadmap
- [RFC 2412](#) - The OAKLEY Key Determination Protocol
- [RFC 3456](#) - DHCPv4 Configuration of IPsec Tunnel Mode
- [RFC 3706](#) - A Traffic-Based Method of Detecting Dead IKE Peers
- [RFC 3947](#) - Negotiation of NAT-Traversal in the IKE
- [RFC 3948](#) - UDP Encapsulation of IPsec ESP Packets

Working Drafts

- [ipsec-isakmp-mode-cfg-05](#) - The ISAKMP Configuration Method
- [ipsec-isakmp-xauth-06](#) - Extended Authentication within ISAKMP/Oakley (XAUTH)
- [ipsec-isakmp-hybrid-auth-05](#) - A Hybrid Authentication Mode for IKE
- [ipsec-nat-t-ike-00](#) - Negotiation of NAT-Traversal in the IKE (v00)
- [ipsec-nat-t-ike-01](#) - Negotiation of NAT-Traversal in the IKE (v01)
- [ipsec-nat-t-ike-02](#) - Negotiation of NAT-Traversal in the IKE (v02)
- [ipsec-nat-t-ike-03](#) - Negotiation of NAT-Traversal in the IKE (v03)
- [ipsec-udp-encaps-00](#) - UDP Encapsulation of IPsec Packets (v00)
- [ipsec-udp-encaps-01](#) - UDP Encapsulation of IPsec Packets (v01)
- [ipsec-udp-encaps-02](#) - UDP Encapsulation of IPsec Packets (v02)
- [ipsec-udp-encaps-03](#) - UDP Encapsulation of IPsec Packets (v03)

Terminology

A host that directly participates in IPsec communications is referred to as an IPsec Peer. Peers use a set of Security Policies to determine which traffic requires protection or authentication. As with any method of secure communication, both parties involved must agree on a common set of parameters that define how data will be secured as well as share common key material. An established set of parameters and key material used to perform cryptographic services is

referred to as a Security Association ("SA").

The IKE Protocol

The Internet Key Exchange protocol ("IKE") offers a means for two Peers to negotiate security parameters and derive suitable keying material. While it may be possible to manually configure the parameters required to participate in an IPsec Peer relationship, most system administrators will elect to use IKE if the option is available.

IKE is a hybrid protocol based on two underlying security protocols, the Internet Security Association and Key Management Protocol ("ISAKMP") and the OAKLEY Key Determination Protocol ("OAKLEY"). According to the IKE RFC, *"ISAKMP provides a framework for authentication and key exchange but does not define them. Oakley describes a series of key exchanges, called 'modes', and details the services provided by each."*

Basic Operation

The basic operation of IKE can be broken down into two phases.

Phase 1 - Negotiates the parameters and key material required to establish an ISAKMP SA. Peer identities and credentials must be verified before Phase 1 can be considered complete. The ISAKMP SA is then used to protect future IKE exchanges.

Phase 2 - Negotiates the parameters and key material required to establish any number of IPsec SA's. The IPsec SA's are then used to protect network traffic that may require security processing.

Exchange Modes

The IKE protocol defines several exchange modes to be used during negotiation. Exchange modes are used to describe a particular packet sequence and the payload requirements for each packet. Some exchanges are similar in purpose but each is unique in their own way.

Identity Protect Mode - The Identity Protect ("Main Mode") Exchange can be used during Phase 1 to negotiate an ISAKMP SA. Transmission of the Peer Identities values is delayed until key material has become available to encrypt the remaining packets in the exchange. This prevents the Identity values from being read by a third party but places some restrictions on the Identity types that can be used with Preshared Key authentication methods.

Aggressive Mode - The Aggressive Exchange can be used during Phase 1 to negotiate an ISAKMP SA. Unlike the Identity Protect Exchange, the Peer Identity values are transmitted before key material is available.

Quick Mode - The Quick Exchange is used during Phase 2 to negotiate an IPsec SA. All packets transmitted during a Quick exchange are encrypted using a previously established ISAKMP SA.

Informational Mode - An Informational Exchange is used to transmit Notification or Security Association Deletion messages between Peers. Whenever possible, packets transmitted during an informational exchange are encrypted using a previously established ISAKMP SA. Unlike other exchange types, Informational exchanges are unidirectional.

The ESP and AH Protocols

A Security Protocol must be used to process traffic between Peers once parameters and key material have become available. Two options have been defined for use with IPsec. The first being the Authentication Header protocol ("AH") and the second being the Encapsulating Security Payload Protocol ("ESP"). While AH can be used to provide message authentication, ESP can be used to provide encryption as well as message authentication.

The only transport protocol currently supported by the Shrew Soft VPN Client is the ESP protocol.

Both Transport Protocols offer two modes of operation. These are referred to as Transport and Tunnel mode. Transport mode is used to protect the data contained within an IP packet payload. Tunnel mode is used to protect an entire IP datagram by encrypting the original header along with the payload data. This encrypted data is then encapsulated in a new IP datagram using header information that is suitable for public network routing. Since Tunnel mode retains the original IP header information, it can be used to process network traffic on behalf of other hosts. This allows an IPsec Peer to function as a security gateway by encrypting and encapsulating all traffic that matches a security policy and then

forwarding the protected traffic to an appropriate Peer gateway. The packets are decapsulated and decrypted and then routed to the final destination based on the original IP header information.

The only mode of operation currently supported by the Shrew Soft VPN Client is Tunnel mode.

Client VPN Gateways

An IPsec VPN Client Gateway is an IPsec capable device that is designed to support client based connectivity. Unfortunately, manually configuring key information is highly undesirable and the IKE Protocol was not originally designed to offer this style of operation.

The relationship between IPsec Peers is defined as one of equal standing. Both Peers provide identities that are verified and credentials that are authenticated. This is referred to as Mutual Authentication. While this behavior may be ideal for Peers that facilitate site to site communications, it is impractical when supporting a large number of mobile devices. Because most aspects of a mobile device configuration can be altered by the operator, it is difficult to ensure that an identity is authentic without introducing a more user-centric authentication mechanism. It is also desirable to have the ability to centrally manage aspects of the remote device operation without user intervention.

For these reasons, several extensions to the protocol have been proposed to extend the functionality of IKE.

Related Protocol Extensions

Configuration Exchange - This extension, also known as Mode Config, was devised to exchange information before negotiating non-ISAKMP SA's (after Phase 1 and before Phase 2). This is accomplished by defining a new exchange type where attributes values may be offered or requested by a Peer. This can be used for purposes such as obtaining an IP address, subnet mask, DNS settings or private network topology information from a gateway.

Extended Authentication - This extension, also known as XAuth, is based on the Configuration Exchange. It was devised to accommodate user-based authentication. Mutual authentication is still required as the additional authentication can only occur after the ISAKMP SA (Phase 1) has been established.

Hybrid Authentication - This extension is based on the Configuration Exchange and Extended Authentication. It was devised to offer user-based authentication without requiring full Mutual Authentication. This is accomplished by simply not authenticating one of the two Peers when attempting to establish the ISAKMP SA (Phase 1). The Peer is later required to pass Extended Authentication to validate the user credentials before allowing IPsec SAs (Phase 2) to be negotiated.

Dead Peer Detection - This extension, also known as DPD, is based on the ISAKMP Informational exchange and provides a method of detecting when a peer is no longer responsive. This is accomplished by submitting and responding to periodic DPD requests. If a Peer fails to respond within a certain time period, all associated SAs are normally considered dead.

All extensions listed above are supported by both the ipsec-tools racoon daemon and the Shrew Soft VPN Client.

Problems with IPSEC

When using a Security Protocol to protect IPsec traffic, packets can often grow to be larger than the Maximum Transmission Unit ("MTU") for a given gateway interface. This is due to the overhead associated with adding new protocol headers and performing packet encapsulation. Some poorly designed routers may simply refuse to fragment or forward certain packet types if they are larger than an arbitrary size. Other routers may drop packet fragments even if they are an acceptable size for the given interface MTU. Finally, it is very common for problems to occur when a router that performs Network Address Translation ("NAT") exists between two IPsec Peers.

To circumvent these issues, several extensions to the IPsec protocol suite have been devised but are not universally supported by all platforms.

Related Protocol Extensions

IKE Fragmentation - In some instances, key exchange packets can be large which will lead to packet loss as described above. By using an extension to the IKE protocol, it is possible for IPsec Peers to exchange large packets even when a trouble router exists between them.

NAT Traversal - Almost all personal firewall appliances employ NAT as a means for multiple devices to share a single Internet connection. By using extensions to the IKE and ESP protocols, it is possible for IPsec Peers to exchange packets even when a NAT device exists between them.

All extensions listed above are supported by the Shrew Soft VPN Client. IKE Fragmentation is a supported feature of the IPsec Tools racoon daemon. NAT Traversal requires kernel support. Please refer to your gateway operating system documentation for more details.

Using the VPN Client

This chapter provides detailed information regarding the installation and configuration of a VPN Gateway. It also describes the steps required to configure the Shrew Soft VPN Client to connect to the VPN Gateway.

List of available topics:

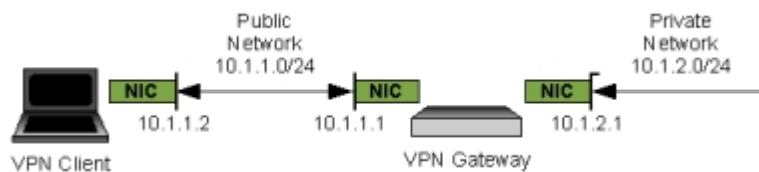
1. [Getting Started](#)
2. [Client Authentication](#)
3. [Client Management](#)
4. [VPN Gateway Installation](#)
5. [VPN Gateway Configuration](#)
6. [VPN Client Configuration](#)

Getting Started

This guide will describe the steps necessary to install and configure both the VPN Gateway and a VPN Client in a small test environment. Before getting started, you will need at least two computers and a network that supports Ethernet connectivity. One computer will act as a VPN Gateway and the other will act as a Windows VPN Client.

The Test Environment

Our test environment will model a single Client connecting to a Gateway that protects a small private network. The same Gateway can be easily redeployed in a production environment by changing the public interface parameters to match your real world configuration. The device acting as the VPN Gateway will need to support two network interface cards. The first will be referred to as the public interface and the second as the private interface. The VPN Client only needs to support a single network interface card that will provide connectivity to the public network.



The IP address scheme chosen for our test environment is arbitrary. It is recommended that you connect the VPN Gateways private interface to your existing network and use valid private network address parameters in place of the parameters used in this guide. This will afford you the opportunity to perform more thorough testing between the Client and any private network resources you may wish to access. This should also allow Internet connectivity for your Gateway which may be necessary when performing some of the installation and configuration steps. At the very least, you need to make sure the Gateways private network interface has an active link status so it will respond to packets sent by the VPN Client.

Gateway Orientation in a Production Environment

In a production environment, the public network that the Client will most likely be connecting from will be the Internet. Depending on the size and complexity of your network, the VPN Gateway functionality can be provided by the same host that provides firewall protection or by a separate dedicated host. We will be using an Integrated Firewall and VPN Gateway for our test environment.

Integrated Firewall and VPN Gateway

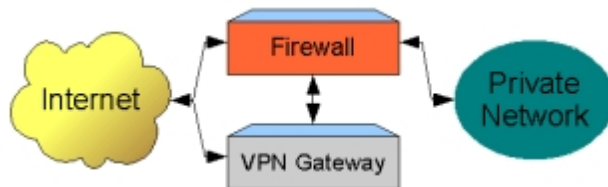
For a small office, it is typical for an existing firewall to also be used as a VPN Gateway. This setup may be practical if the cost associated with maintaining additional hardware is undesirable.



In this scenario, the Integrated Firewall / VPN Gateway device allows the private network to access Internet resources while also allowing VPN Clients to access private network resources.

Dedicated Firewall and VPN Gateway

For a larger office, it is typical for a separate dedicated Firewall and VPN Gateway to be used. This would prevent IPsec packet processing from degrading the Firewall performance and allow for a higher degree of security by enabling a separate host to filter packets between the VPN Gateway and the private network.



In this scenario, the Firewall allows the private network to access Internet resources. The VPN Gateway allows Clients to access certain private network resources by forwarding the traffic through the Firewall. In the event that the VPN Gateway is compromised, the Firewall will still have the ability to protect resources that need to remain private. Inbound access to the private network could be cut off completely while still allowing outbound access to Internet resources.

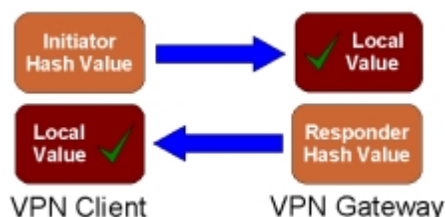
Client Authentication

The IKE Protocol defines several methods that can be used to authenticate a VPN Client with a VPN Gateway. To process Peer authentication, an Identity needs to be verified and credentials need to be validated using a predefined method. The Shrew Soft VPN Client supports the use of standard Preshared Key and RSA Certificate Authentication as well as the Extended and Hybrid Authentication protocol extensions.

The use of an external LDAP account database such as Microsoft Active Directory, Novell eDirectory or OpenLDAP is recommended to support Extended or Hybrid authentication. This allows for centralized account management and group membership checks to be used to limit access to private network resources.

Preshared Key Authentication

To perform Preshared Key Authentication, both Peers must have access to a common shared key value. During IKE authentication, the Peer being validated sends a Hash Value generated using the shared key and ancillary data.



The received Hash Value is then checked against a locally generated Hash Value using the same shared key and ancillary data. If the values match, the peer is considered authentic.

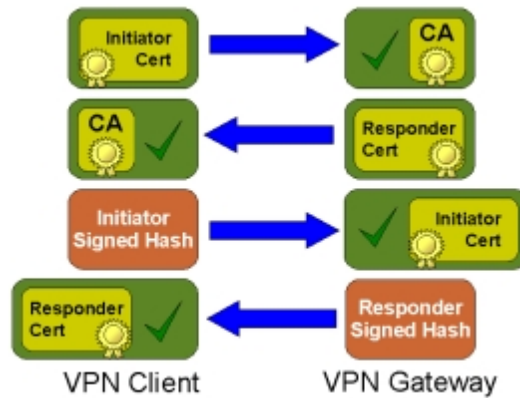
see also:

[Configuring IPsec Tools : Preshared Key Authentication](#)
[VPN Client Configuration : Authentication Method](#)

RSA Authentication

To perform RSA Authentication, the Peer being validated must hold a Digital Certificate signed by a trusted Certificate Authority and the Private Key for the Digital Certificate. The Peer performing the authentication only needs a copy of the trusted Certificate Authorities Digital Certificate. During IKE authentication, the Peer being validated sends a copy of the

Digital Certificate and a Hash Value signed using the Private Key.



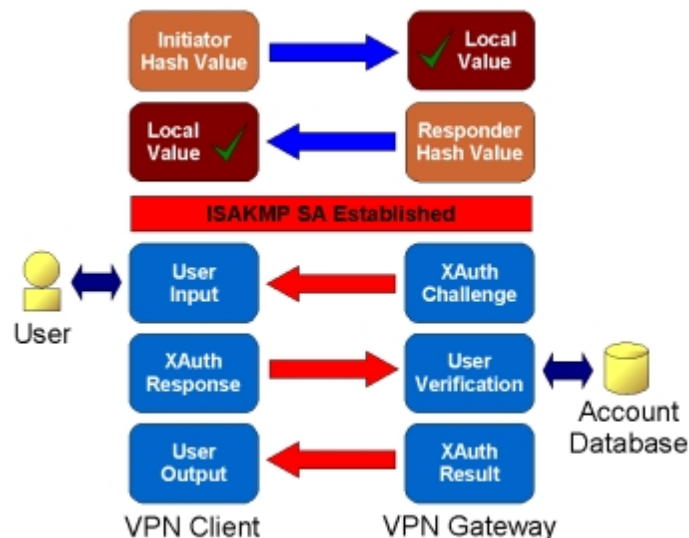
The received Digital Certificate is first verified to have been signed by the Certificate Authority Private Key. The received Hash Value is then verified to have been signed by the Digital Certificate Private Key. If both signatures are valid, the peer is considered authentic.

see also:

[Configuring IPsec Tools : RSA Authentication](#)
[VPN Client Configuration : Authentication Method](#)

Extended Authentication

To perform Extended Authentication, an ISAKMP security association will need to be established using a normal Preshared Key or RSA Authentication. Immediately after, the Client is typically required to provide a username and password. The Gateway will have access to a user account database to validate the credentials.



In this example, an ISAKMP security association is established using Preshared Key authentication. The security association is then used to protect the Extended Authentication which is submitted to an external account database for verification. If both the Preshared Key and Extended Authentication attempts succeed, the peer is considered authentic.

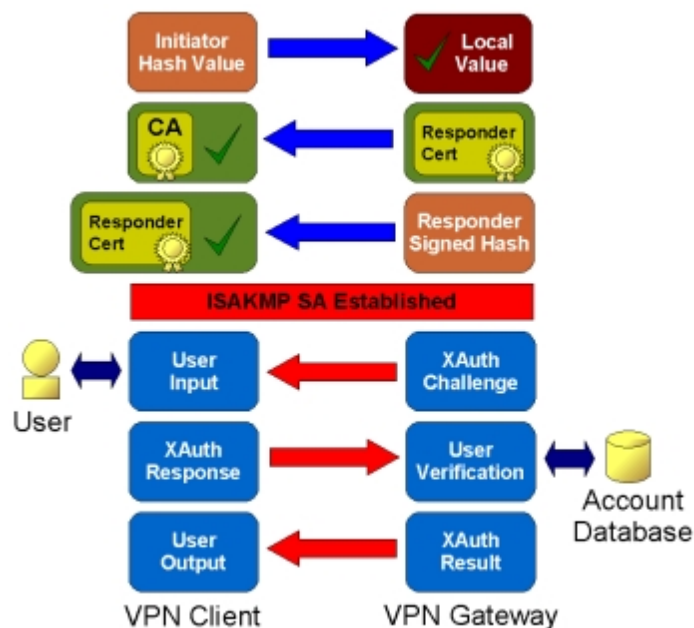
see also:

[Configuring IPsec Tools : Preshared Key Authentication](#)
[Configuring IPsec Tools : User Authentication](#)
[VPN Client Configuration : Authentication Method](#)

Hybrid Authentication

To perform Hybrid Authentication, an ISAKMP security association will need to be established using a modified form of

RSA Authentication that only validates the Gateway. Immediately after, the Client is typically required to provide a username and password. The Gateway will have access to a user account database to validate the credentials.



In this example, an ISAKMP security association is established using RSA authentication. The security association is then used to protect the Extended Authentication which is submitted to an external account database for verification. If both the RSA and Extended Authentication attempts succeed, the peer is considered authentic.

see also:

- [Configuring IPsec Tools : RSA Authentication](#)
- [Configuring IPsec Tools : User Authentication](#)
- [VPN Client Configuration : Authentication Method](#)

Client Management

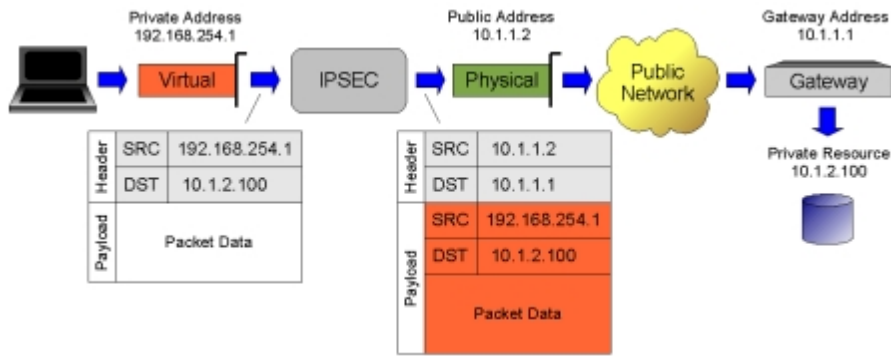
One of the challenges that face a VPN system administrator is keeping the Client configuration up to date when making a network change that may impact remote users. By using the Configuration Exchange, a VPN Gateway can provide dynamic configuration information to the Client during IKE negotiations. The alternative is to maintain a static Client configuration that will require user intervention if changes are required. This section describes some of the available configuration options and how they relate to client operation.

Private Address Configuration

A Gateway can be configured to assign a private address and netmask to a Client. To support this, the Shrew Soft VPN Client offers two modes of operation that determine how private network traffic is sent to the VPN Gateway. The first is referred to as Virtual Adapter Mode and the second as Direct Adapter Mode.

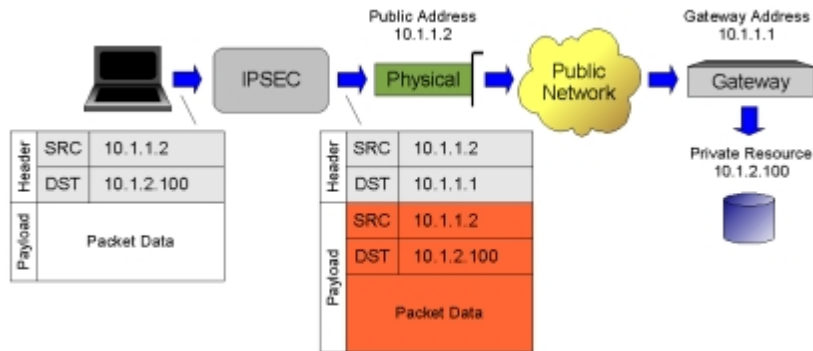
Virtual Adapter Mode

When operating in Virtual Adapter Mode, the VPN Client creates a virtual adapter to use with IPsec communications. Local route management is used to ensure that packets destined to the remote private network will be sourced from the virtual adapter.



Direct Adapter Mode

When operating in Direct Adapter Mode, the VPN Client only uses the physical adapter for IPsec communications. Any private address assigned by the gateway will be ignored. Packets destined to the remote private network will be sourced from the physical adapter.



see also:

[Configuring IPsec Tools : Network Configuration](#)
[VPN Client Configuration : Address Method](#)

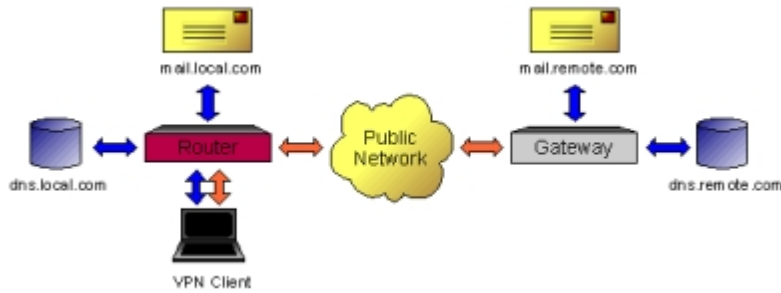
Name Services Configuration

A Gateway can be configured to provide a Client with DNS and WINS configuration information. These parameters include DNS server addresses, a DNS default domain name, DNS split domain lists and WINS server addresses.

DNS Settings

By providing DNS configuration information, a Client will be able to resolve DNS names using a server located in the remote private network. This is useful when the DNS server hosts any number of private zones that are not resolvable using a public DNS server. A more advanced configuration would also provide a list of DNS split domain name suffixes. Using this list, a Client will only forward DNS requests to a private DNS server if they match one of the domain name suffixes. All other requests will be handled using the local DNS settings.

In the example shown below, the VPN Client is connected to a local network that has a DNS server that hosts the private local.com domain. It communicates with a remote network that has a DNS server that hosts the remote.com domain. If no DNS settings are supplied to the Client, it would be able to resolve the mail.local.com host name but not the the mail.remote.com host name. If the remote DNS server setting is supplied to the Client, it would be able to resolve the mail.remote.com host name but not the the mail.local.com host name. If the remote DNS server setting and a remote.com split DNS suffix setting is supplied to the Client, it would be able to resolve both the mail.local.com and mail.remote.com host names.



WINS Settings

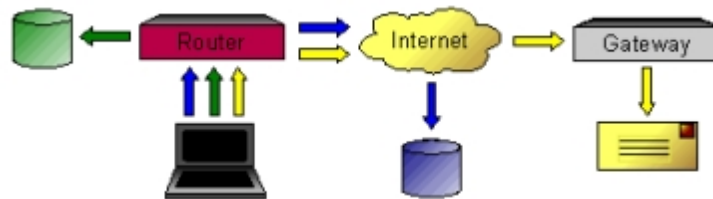
By providing WINS configuration information, a Client will be able to resolve WINS names using a server located in the remote private network. This is useful when attempting to access a remote windows resource using a Uniform Naming Convention (UNC) path name. The WINS server address would typically belong to a Windows Domain Controller or a Samba server.

see also:

[Configuring IPsec Tools : Name Service Configuration](#)
[VPN Client Configuration : Name Resolution Settings](#)

Split Tunnel Configuration

A Gateway can be configured to provide a Client with information that can be used to automatically generate a detailed security policy list. Using a policy list to selectively forward traffic to the VPN Gateway is referred to as Split Tunneling. The use of Split Tunneling can drastically reduce the amount of traffic processed by a VPN Gateway. It may also be required to allow a Client to access resources available from the local private network they connect from.



An example of a VPN Client that selectively tunnels traffic to the VPN Gateway.

If a Client is configured to obtain the policy information automatically from a Gateway but no information is received, all traffic will be forwarded to the Gateway as a last resort. Unless the Gateway and Firewall have been specifically configured to pass Client traffic back out of a public interface, a Client may have problems accessing Internet resources.



An example of a VPN Client that attempts to tunnel all traffic to the VPN Gateway.

see also:

[Configuring IPsec Tools : Split Network Configuration](#)
[VPN Client Configuration : Policy Settings](#)

VPN Gateway Installation

Before the Shrew Soft VPN Client can be used to access your private network from the public network, you must first install and configure a VPN Gateway. Because the Shrew Soft Client has been designed to communicate with an open

source operating system, we will concentrate on installing and configuring a Linux, FreeBSD or NetBSD host.

List of available topics:

1. [Operating System Installation](#)
2. [Kernel Configuration](#)

Operating System Installation

This section provides a basic overview of the steps required to install and configure a Gateway operating system. It should not be considered a substitute for the operating system documentation. When in doubt, please consult the official documentation for more information.

Before beginning your Gateway operating system installation, you will first need to obtain the appropriate installation media for your hardware. There are many Linux distributions and several NetBSD and FreeBSD versions to choose from. In this document, we describe how to use Fedora Core 6, FreeBSD 6.2 or NetBSD 3.1 to build a working VPN Gateway. Please visit the web site of your preferred operating system to download and burn the corresponding ISO images as CDs.

[Fedora Core Web Site](#)

[FreeBSD Web Site](#)

[NetBSD Web Site](#)

Fedora Core 6

If you are unfamiliar with Fedora Core 6, it is recommended that you read the [Installation Guide](#) before beginning the installation. Once you feel comfortable, boot the Gateway computer using the first installation disc. You will be presented with a boot prompt to select a user interface mode. We will be using the text based installation mode so enter the following at the boot prompt and press enter.

```
boot: linux text
```

The Fedora Core text installer should now be loaded. The arrow and tab keys can be used to navigate the dialogs. The space key can be used to toggle options and the enter key can be used to activate the button selections.

Preparing for the Installation

The first dialog presented will ask you if you would like to test the CD media before installation. If the CD burning software you used already verified the CD media, select *Skip* to continue with the installation. Otherwise, select *OK* and follow the prompts to perform the testing. Once you are presented with a Welcome screen, press the Enter key to continue.

Language Selection

To select the language, use the arrow keys to navigate up and down. Once your desired language is highlighted, use the tab key to select *OK* and press Enter to continue. This guide assumes the *English* option will be selected.

Keyboard Selection

To select your keyboard model, use the arrow keys to navigate up and down. Once your desired model is highlighted, use the tab key to select *OK* and press Enter to continue.

Preparing an Installation Partition

The installer program will now examine your hard drive partitions. In the event that there is no partition table, the installer will ask you if you would like to initialize one. If so, use the tab key to select *Yes* and press Enter to continue. If you have an existing partition table, you will select the *Remove all partitions on selected drives and create default layout* option. Use the tab key to select *Ok* and press Enter to continue. The installer will then ask you to confirm the deletion of all partitions, use the tab key to select *Yes* and press Enter to continue. When asked if you to review the Layout, select *No* and press Enter to continue. Depending on your available memory, you may be asked if to enable the swap space immediately. If so, use the tab key to select *Yes* and press Enter to continue.

Configuring Network Interfaces

The installer program will now ask you to enter configuration information for your network interfaces. We will only be concerned with eth0 and eth1 which will be used as the public and private gateway interfaces.

Linux Interface Name	Gateway Interface	IP Address	Prefix
eth0	public	10.1.1.1	24
eth1	private	10.1.2.1	24

For eth0, use the arrow and space keys to highlight and disable both the *Use dynamic IP configuration (DHCP)* and *Enable IPv6 support* options. Make sure the *Activate on boot* option is enabled. Then use the arrow keys to select the IPv4 address field and enter the public address as 10.1.1.1 with a Prefix of 24. Use the tab key to select *Ok* and press Enter to continue.

For eth1, use the arrow and space keys to highlight and disable both the *Use dynamic IP configuration (DHCP)* and *Enable IPv6 support* options. Make sure the *Activate on boot* option is enabled. Then use the arrow keys to select the IPv4 address field and enter the private address as 10.1.2.1 with a Prefix of 24. Use the tab key to select *Ok* and press Enter to continue.

Miscellaneous Network Settings

The installer program will now ask you to enter some optional network settings. If you plan to connect the private interface of your gateway to an existing IP network, it is recommended that you use the appropriate parameters for these options. If you plan to isolate the test environment completely from your existing network, you can leave these options blank.

Hostname Configuration

The installer program will now ask you to enter the hostname configuration. Since DHCP will not be used, select the manual option and enter a hostname value in the field provided. You can use any name you want but for the purposes of this document, we will use the hostname *fedora*. After entering your hostname, use the tab key to select *Ok* and press Enter to continue.

Time Zone Selection

To select your Time Zone, use the arrow keys to navigate up or down. Once your desired zone is highlighted, use the tab key to select *OK* and press Enter to continue.

Root Password

To enter the root password, type a password you won't forget into the provided field. You will need to retype the password again in the next field to ensure no mistakes were made. A strong password with at least eight characters including one upper case and one numeric is recommended. Once you have finished, use the tab key to select *OK* and press Enter to continue.

Package Selection

The installer program will now ask you to enter your package selection. Use the arrow and space keys to disable the *Office and Productivity* option and enable the *Software Development* option. You should also enable the *Customize Software Selection* so you have an opportunity to disable unwanted package groups. Use the tab key to select *OK* and press Enter to continue. When presented with the *Package Group Selection* screen, you can optionally disable the following package groups which are not required for the VPN Gateway to operate.

- Gnome Desktop Environment
- Gnome Software Development
- Graphical Internet
- Printing Support
- Text-based Internet
- X Software Development
- X Windows System

Once you have made your final selections, use the tab key to select *OK* and press Enter to continue.

Beginning the Installation

The installer will now run a dependency check for the package list followed by a screen that informs you that the installation is about to begin. Use the tab key to select *OK* and press Enter to continue. You will then be presented with a

dialog that lists the CDs required for the installation. Pay careful attention because the default action selected will reboot your system. Use the tab key to select *Continue* and press Enter. The system will now create your partitions, format the file systems, copy all required operating system files and install your selected packages. You may be prompted to insert additional CDs as the installation continues. Once complete, the installer will ask to confirm a reboot of your newly installed operating system. Remove any CDs that may be in the drive and press Enter to continue.

Setup Utility

After the system reboots, Fedora will load the setup utility which allows you to configure several operating parameters. This same tool can be run later by typing *setup* from a command line using the root account. For now, you can use the Tab key to select *Exit* and press Enter to continue.

FreeBSD 6.2

If you are unfamiliar with FreeBSD, it is recommended that you read the [Installing FreeBSD](#) section of the FreeBSD Handbook before beginning the installation. Once you feel comfortable, boot the Gateway computer using the first installation disc.

Country Selection

The first dialog presented will ask you for your country selection. To select your country, use the arrow keys to navigate up or down. Once your desired country is highlighted, press Enter to continue.

Custom Installation

You should now be in the root menu of the sysinstall program. The arrow and tab keys can be used to navigate the dialogs. The space key can be used to toggle options and the Enter key can be used to activate the button selections. To begin a custom installation, use the arrow keys to highlight *Custom* and press Enter to continue.

Partition Editor

From the Custom Installation menu, select the *Partition* option and press Enter to continue. This will load the FreeBSD FDISK Partition Editor. Any existing partitions and unused space are listed in the center of the screen and a command reference is listed at the bottom of the screen. The simplest way to proceed is press the A key which will instruct FreeBSD to use the entire disc. Next, press the Q key to quit the editor. The install process will now ask you to select a boot manager option for the target drive. Because the Gateway will only be running FreeBSD, select the *Standard* option and press Enter to continue.

Disklabel Editor

From the Custom Installation menu, select the *Label* option and press Enter to continue. This will load the FreeBSD Disklabel Editor. FreeBSD uses disklabels to subdivide the partition into logical sections. Any existing labels will be listed in the center of the screen and a command reference is listed at the bottom of the screen. The simplest way to proceed is press the A key which will instruct FreeBSD to use the automatic defaults. Afterwards, press the Q key to quit the editor.

Distributions

From the Custom Installation menu, select the *Distributions* option and press Enter to continue. You will then be presented with a screen that allows you to select the distribution sets to be installed. Use the arrow keys to select the 4 *Developer* option and press the space key to continue. Afterwards, you will be asked if you would like to install the FreeBSD ports collection. Use the tab key to select *No* and press Enter to continue. Use the arrow keys to select the X *Exit* option and press the Enter key to continue.

Beginning the Installation

From the Custom Installation menu, select the *Commit* option and press Enter to continue. The sysinstall program will now ask you to select an installation media. Because we are using an install CD as the media, make sure the *CD/DVD* option is highlighted and press the Enter key to continue. You will then be prompted with a message asking for confirmation before proceeding with the installation. Press the Enter key to continue. The system will now create your partitions, format the file systems and copy all required operating system files. Next, you will be asked if you would like to visit the general configuration menu for a chance to set any last options. Use the tab key to select *Yes* and press Enter to continue.

Setting the Root Password

From the Configuration menu, select the *Root Password* Option and press Enter to continue. When prompted, type a password you won't forget and press Enter to continue. You will be prompted to repeat this process again to ensure no mistakes were made. A strong password with at least eight characters including one upper case and one numeric is recommended.

Rebooting the System

After leaving the Network Services interface, you will be placed back in the Custom Installation menu. From here, you need to select the *X Exit* option and press Enter to continue. Now that you are at the Main Menu, use the tab key to select the *Exit Installation* option and press the Enter key to continue. You will be asked to confirm. Use the tab key to select *Yes* and press the Enter key to continue.

Network Configuration

FreeBSD reads its system configuration settings from files located in the `/etc` subdirectory. You will need to add a few lines to the `rc.conf` file which will contain the setting to be applied to your network adapters at boot time. Before these lines can be added, you need to discover your Ethernet device names.

FreeBSD device names are prefixed with the device driver name that supports them. For instance, a device supported by one driver may be named `fxp0` while a device supported by another driver may be name `bge0`. To locate your network device names, use the `ifconfig` command to obtain a list of available network interfaces as shown below:

```
ifconfig
lnc0: flags=108802<BROADCAST,SIMPLEX,MULTICAST,NEEDSGIANT> mtu 1500
    ether 00:0c:29:f3:f3:af
lnc1: flags=108802<BROADCAST,SIMPLEX,MULTICAST,NEEDSGIANT> mtu 1500
    ether 00:0c:29:f3:f3:b9
lo0: flags=8049<UP,LOOPBACK,RUNNING,MULTICAST> mtu 16384
    inet6 fe80::1%lo0 prefixlen 64 scopeid 0x4
    inet6 ::1 prefixlen 128
    inet 127.0.0.1 netmask 0xff000000
```

For our example test environment, the host will have two Ethernet devices named `lnc0` and `lnc1`.

FreeBSD Interface Name	Gateway Interface	IP Address	NetMask
<code>lnc0</code>	<code>public</code>	<code>10.1.1.1</code>	<code>255.255.255.0</code>
<code>lnc1</code>	<code>private</code>	<code>10.1.2.1</code>	<code>255.255.255.0</code>

To configure the devices, open the `/etc/rc.conf` file with a text editor and add an `ifconfig` line for each adapter. The line will specify the device name before the equals sign followed by the network settings contained in quotes. For example, you could add lines similar to the ones shown below:

```
ifconfig_lnc0="10.1.1.1 netmask 255.255.255.0"
ifconfig_lnc1="10.1.2.1 netmask 255.255.255.0"
```

To configure the host name, open the `/etc/rc.conf` file with a text editor and add a `hostname` line that specifies the name contained in quotes. For example, you could add a line similar to the one shown below:

```
hostname="freebsd.shrew.net"
```

The Gateway will need to support packet forwarding to operate correctly. To enable this, open the `/etc/rc.conf` file with a text editor and add the line shown below:

```
gateway_enable="YES"
```

If you plan to connect the private interface of your gateway to an IP network that is larger than the interface subnet, open the `/etc/rc.conf` file with a text editor and add a `defaultrouter` line that specifies the gateway address contained in quotes. For example, you could add a line similar to the one shown below:

```
defaultrouter="10.1.2.254"
```

To configure the DNS settings, open the `/etc/resolv.conf` file with a text editor and add a `domain` line that specifies the default domain name. You can also add any number of `nameserver` lines that specify a name server address. For

example, you could add lines similar to the ones shown below:

domain	shrew.net
nameserver	10.1.2.253

NetBSD 3.1

If you are unfamiliar with NetBSD, it is recommended that you read the [Example Installation](#) section of the NetBSD guide before beginning the installation. Once you feel comfortable, boot the Gateway computer using the installation disc.

Language Selection

The first dialog presented will ask you for your installation language selection. To select your language, use the arrow keys to navigate up or down. Once your desired language is highlighted, press Enter to continue. This guide assumes the *English* option will be selected.

Keyboard Selection

To select your keyboard type, use the arrow keys to navigate up and down. Once your desired type is highlighted, use the tab key to select *OK* and press Enter to continue.

Beginning the Install

You should now be at the system installation tool welcome screen. The arrow and tab keys can be used to navigate the dialogs and the Enter key can be used to activate a selection. The install tool will ask you to select one of several options. Use the arrow keys to select *Install NetBSD to a hard disk* and press the Enter key to continue. Next, you will be prompted to confirm the operation. Use your arrow key to select *Yes* and press Enter to continue.

Selecting a Hard Disk

The installation tool will now ask you to select the hard disk that you would like to install the operating system on. If you have only one hard drive, press Enter to continue. Otherwise, select the first hard drive listed and press Enter to continue.

Selecting Distribution Sets

The installation tool will now ask you to select the distribution sets to be installed. Use the arrow keys to select *Custom installation* and press Enter to continue. When presented with the distribution set selection screen, you can optionally disable the following sets which are not required for the VPN Gateway to operate.

- Games
- X11 sets (Deselect all)

Disk Partition Configuration

The installation tool will now ask you if you would like to manually edit the partition table or use the entire disk. Use the arrow keys to select the *Use the entire disk* option and press Enter to continue. If the installer asks you to install the NetBSD bootcode, use the arrow keys to select *Yes* and press Enter to continue.

Disk Label Configuration

The installation tool will now ask you to create one or more disk labels. NetBSD uses Disk Labels to subdivide the partition into logical sections. Use the arrow keys to select the *Set sizes of NetBSD partitions* option and press Enter to continue. The installation tool will load the disk label editor with the system default values. Use the arrow keys to select *Accept partition sizes* and press Enter to continue. Next, you will be asked to confirm your partition configuration. Select the *Partition sizes ok* and press Enter to continue.

Naming your Hard Disk

The installation tool will now ask you to enter a name for your disk. It should provide a default name enclosed in brackets. Press the Enter key to continue.

Beginning the Installation

The installation tool will now ask you to confirm before writing information to your hard drive. Use the arrow keys to select *Yes* and press *Enter* to continue. The installation tool will now ask you if you would like to use a normal BIOS console (monitor and keyboard) or a serial port as your bootblock. To leave the default *Use Bios console* option, use the arrow keys to select the *Exit* option and press the *Enter* key to continue. You will now be asked to select how you would like the install progress to be presented. Leave the default of *Progress bar* and press *Enter* to continue. The install tool will now ask you to select an installation media. Because we are using an install CD as the media, make sure the *CD-ROM/DVD* option is highlighted and press the *Enter* key to continue. When asked to select the device, leave the default by using the arrow keys to select *Continue* and press the *Enter* key. The system will now create your partitions, format the file systems and copy all required operating system files. The install tool should then inform you that it is finished extracting the distribution sets to your hard drive. Press the *Enter* key to continue.

Time Zone Configuration

The installation tool will now ask you to select a time zone. The time zone lists are organized by country or region. For example, the US central time zone would be located under the US selection. Use the arrow and *Enter* key to highlight and select your preferred time zone. Next, use the arrow keys to select *Exit* and press the *Enter* key to continue.

Password Cipher Configuration

The installation tool will now ask you to select a password cipher. Use the arrow keys to highlight the MD5 option and press *Enter* to continue.

Setting the Root Password

The installation tool will now ask you if you would like to initialize the root password. Leave the default of *Yes* highlighted and press *Enter* to continue. When prompted, type a password you won't forget and press *Enter* to continue. You will be prompted to repeat this process again to ensure no mistakes were made. A strong password with at least eight characters including one upper case and one numeric is recommended.

Default Shell Configuration

The installation tool will now ask you to select a default shell. Unless you prefer another shell, use the arrow keys to highlight the */bin/sh* option and press the *Enter* key to continue.

Rebooting the System

The installation tool will now inform you that the install is complete. Press the *Enter* key to return to the main install menu. Use the arrow keys to highlight the *Reboot the computer* option and press the *Enter* key.

Network Configuration

NetBSD reads its system configuration settings from several files located in the */etc* subdirectory. You will need to add a few lines to the *rc.conf* file which will contain the setting to be applied to your network adapters at boot time. Before these lines can be added, you need to discover your Ethernet device names.

NetBSD device names are prefixed with the device driver name that supports them. For instance, a device supported by one driver may be named *fxp0* while a device supported by another driver may be named *bge0*. To locate your network device names, use the *ifconfig* command to obtain a list of available network interfaces as shown below:

```
ifconfig -a
pcn0: flags=8802<BROADCAST,SIMPLEX,MULTICAST> mtu 1500
    address: 00:0c:29:4d:fd:99
    media: Ethernet autoselect (autoselect)
pcn1: flags=8802<BROADCAST,SIMPLEX,MULTICAST> mtu 1500
    address: 00:0c:29:4d:fd:a3
    media: Ethernet autoselect (autoselect)
lo0: flags=8009<UP,LOOPBACK,MULTICAST> mtu 33192
    inet 127.0.0.1 netmask 0xff000000
```

For our example test environment, the host will have two Ethernet devices named *pcn0* and *pcn1*.

NetBSD Interface Name	Gateway Interface	IP Address	NetMask
pcn0	public	10.1.1.1	255.255.255.0
pcn1	private	10.1.2.1	255.255.255.0

To configure the devices, open the `/etc/rc.conf` file with a text editor and add an `ifconfig` line for each adapter. The line will specify the device name before the equals sign followed by the network settings contained in quotes. For example, you could add lines similar to the ones shown below:

```
ifconfig_pcn0="10.1.1.1 netmask 255.255.255.0"  
ifconfig_pcn1="10.1.2.1 netmask 255.255.255.0"
```

To configure the host name, open the `/etc/rc.conf` file with a text editor and add a `hostname` line that specifies the name contained in quotes. For example, you could add a line similar to the one shown below:

```
hostname="netbsd.shrew.net"
```

If you plan to connect the private interface of your gateway to an IP network that is larger than the interface subnet, open the `/etc/rc.conf` file with a text editor and add a `defaultroute` line that specifies the gateway address contained in quotes. For example, you could add a line similar to the one shown below:

```
defaultroute="10.1.2.254"
```

To configure the DNS settings, open the `/etc/resolv.conf` file with a text editor and add a `domain` line that specifies the default domain name. You can also add any number of `nameserver` lines that specify a name server address. For example, you could add lines similar to the ones shown below:

```
domain      shrew.net  
nameserver  10.1.2.253
```

The Gateway will need to support packet forwarding to operate correctly. To enable this, open the `/etc/sysctl.conf` file and add the following line:

```
net.inet.ip.forwarding=1
```

Kernel Configuration

After the initial Gateway installation, it may be necessary to enable kernel support for IPsec protocols and Firewall support. This will require that a new kernel be compiled and installed. The steps to complete this process are different depending on the operating system you have selected. The sections below will give an overview on how to accomplish this on either a FreeBSD or NetBSD host. The stock Fedora Core 6 kernel already contains all the kernel support we need so compiling a custom kernel is not necessary.

FreeBSD 6.2

This section describes the basic steps required to build a FreeBSD kernel with FAST IPsec and Packet Filter support. For more information of this topic, please read the FreeBSD Handbook chapter titled [Building and installing a Custom Kernel](#). The Handbook also contains another chapter titled [VPN over IPsec](#) that may be useful.

NAT Traversal Support Patch

As of FreeBSD 6.2 stable, NAT Traversal support is not yet available in the stock kernel sources. However, support for this feature is available in the form of a kernel patch that is maintained by a member of the Ipsec Tools development team. Applying this patch is not required but it is highly recommended if you plan to support Clients that connect from behind a NAT enabled Firewall. To obtain the kernel patch and apply it to the FreeBSD kernel sources, follow the prompts listed below using a root login:

```
cd /usr/src/sys  
fetch http://ipsec-tools.sf.net/freebsd6-natt.diff  
patch < freebsd6-natt.diff
```

Creating a Kernel Configuration File

The FreeBSD kernel build system requires a kernel configuration file as input. The file describes what options should be included when compiling a kernel. Instead of creating a configuration file from scratch, the generic kernel configuration file should be copied and edited to add or remove support for specific kernel options. It is important to know what architecture your gateway host supports before beginning this procedure. In most instances, this will be either `i386` or `amd64`. For the

purpose of the example given below, we will assume a host using the i386 architecture. To create a new kernel configuration file named CUSTOM, follow the prompts below using a root login:

```
cd /usr/src/sys/i386/conf
cp GENERIC CUSTOM
```

Note: The i386 directory may need to be different depending on your architecture type.

Now that you have created a configuration file, a few modifications will need to be made. Open the file with a text editor and change the ident line to read CUSTOM instead of GENERIC:

```
machine      i386
cpu          I486_CPU
cpu          I586_CPU
cpu          I686_CPU
ident       CUSTOM
```

To complete the configuration file changes, add the following lines to the end and save the file:

```
# Packet Filter Support
device      pf
device      pflog

# FAST IPsec Support
device      crypto
device      enc
options     FAST_IPsec
options     IPsec_NAT_T
```

Note: The last option line is only valid if the NAT Traversal kernel patch was applied.

Compiling and Installing the Custom Kernel

Now that you have a custom kernel configuration file that includes support for FAST IPsec and Packet Filter, it can be used to compile and install a new kernel. To perform this procedure, follow the prompts below using a root login:

```
cd /usr/src
make buildkernel KERNCONF=CUSTOM
make installkernel KERNCONF=CUSTOM
```

After the new kernel has been installed, reboot the FreeBSD host to begin using the new features.

Compiling and Installing Userland Programs

If your kernel was patched to support NAT Traversal, the FreeBSD userland programs must be recompiled and installed. To perform this procedure, follow the prompts below using a root login:

```
cd /usr/src
make buildworld
make installworld
```

After the new userland programs have been installed, reboot the FreeBSD host to begin using the new features.

NetBSD 3.1

This section describes the basic steps required to build a NetBSD kernel with IPsec and IP Filter support. For more information of this topic, please read the NetBSD Documentation Kernel section titled [How to build a kernel](#). NetBSD also has published a document entitled [How to build a remote user access VPN](#) that may be useful.

Installing the Kernel Sources

Before a kernel can be compiled, you need to ensure that the kernel sources are installed. This can be accomplished by downloading and extracting the archive into the appropriate directory. To download and extract the kernel sources, follow the prompts below using a root login:

```
ftp -a ftp.netbsd.org
ftp> bin
ftp> get pub/NetBSD/NetBSD-3.1/source/sets/syssrc.tgz /tmp/syssrc.tgz
ftp> exit
```

```
cd /
tar xvzpf /tmp/syssrc.tgz
```

Creating a Kernel Configuration File

The NetBSD kernel build system requires a kernel configuration file as input. The file describes what options should be included when compiling a kernel. Instead of creating a configuration file from scratch, the generic kernel configuration file should be copied and edited to enable or disable support for specific kernel options. It is important to know what architecture your gateway host supports before beginning this procedure. In most instances, this will be either i386 or amd64. For the purpose of the example given below, we will assume a host using the i386 architecture. To create a new kernel configuration file named CUSTOM, follow the prompts below using a root login:

```
cd /usr/src/sys/arch/i386/conf
cp GENERIC CUSTOM
```

Note: The i386 directory may need to be different depending on your architecture type.

Now that you have created a configuration file, a few modifications will need to be made. Open the file with a text editor and make sure the following lines are uncommented by removing the leading # character if necessarily:

```
options      GATEWAY      # packet forwarding
options      INET        # IP + ICMP + TCP + UDP
options      IPsec       # IP security
options      IPsec_ESP   # IP security (encryption part; define w/IPsec)
options      IPsec_NAT_T # IPsec NAT traversal (NAT-T)
options      PFIL_HOOKS  # pfil(9) packet filter hooks

pseudo-device ipfilter  # IP filter (firewall) and NAT
```

Compiling and Installing the Custom Kernel

Now that you have a custom kernel configuration file that includes support for IPsec and IP Filter, you need to run the config program which verifies the option syntax and creates a new build directory. To perform this procedure, execute the following command from the directory that contains your new configuration file:

```
config CUSTOM
```

The new kernel configuration is now ready to be compiled and installed. To perform this procedure, follow the prompts below using a root login:

```
cd ../compile/CUSTOM/
make depend
make
make install
```

After the new kernel has been installed, reboot the NetBSD host to begin using the new features.

VPN Gateway Configuration

A VPN Gateway operating system provides the necessary kernel support to protect IP traffic using the IPsec protocols. To support VPN Client connectivity, a set of tools will be needed to allow an administrator to interface with the kernel facilities and provide support for the Internet Key Exchange protocol. We will use the IPsec Tools software to provide this functionality.

List of available topics:

1. [Installing IPsec Tools](#)
2. [Configuring IPsec Tools](#)
3. [Creating a Firewall Rule Set](#)

Installing IPsec Tools

The IPsec-Tools software started as a port of the [KAME](#) IPsec utilities to the Linux platform. The most important component of this software is an advanced Internet Key Exchange daemon that can be used to automatically key IPsec connections. For our test environment, we require version 0.7 or later which will need to be download and compiled manually. To obtain the download url for the latest version of the IPsec Tools source code archive, please visit the [IPsec](#)

Fedora Core 6

Fedora Core 6 will have an older version of IPsec Tools already installed. Before a new version can be installed, you will need to remove the existing version. To lookup the name of the installed package, use the rpm and grep utilities as shown below:

```
rpm -qa | grep ipsec-tools
```

In this example, the ipsec-tools-0.6.5-6 package is currently installed. To remove the package, use the rpm utility as shown below:

```
rpm -e ipsec-tools-0.6.5-6
```

Once you have uploaded the latest stable IPsec Tools source code archive to the Gateway, you can extract the distribution using the tar utility as shown below:

```
tar zxvf <ipsec tools archive>
```

Note: If you downloaded a bzip archive instead of a gzip archive, use jxvf instead of zxvf with the tar command.

Next, change to the newly created directory as shown below:

```
cd <ipsec tools directory>
```

Before compiling the software, the configure script needs to be used to set some compile parameters and enable some advanced options. To view a list of all available options, execute the configure script with the --help switch as shown below:

```
./configure --help
```

The default install prefix for ipsec tools is /usr/local. Because Linux typically installs software in /usr and expects configuration files to exist under /etc, you will need to add some extra options to cope with these differences. The other options shown below are to enable XAuth, Dead Peer Detection, IKE Fragmentation and NAT Traversal support. Execute the configure script as shown below:

```
./configure --prefix=/usr --sysconfdir=/etc/racoon --enable-hybrid --enable-frag --enable-dpd  
--enable-natt
```

To compile and install the software, use the make command as shown below:

```
make  
make install
```

FreeBSD 6.2

Once you have uploaded the latest stable IPsec Tools source code archive to the Gateway, you can extract the distribution using the tar utility as shown below:

```
tar zxvf <ipsec tools archive>
```

Note: If you downloaded a bzip archive instead of a gzip archive, use jxvf instead of zxvf with the tar command.

Next, change to the newly created directory as shown below:

```
cd <ipsec tools directory>
```

Before compiling the software, the configure script needs to be used to set some compile parameters and enable some advanced options. To view a list of all available options, execute the configure script with the --help switch as shown below:

```
./configure --help
```

The default install prefix for ipsec tools is /usr/local. This is the normal location for add-on software in FreeBSD so you won't need any extra options to deal with this. The other options shown below are to enable XAuth, Dead Peer Detection, IKE Fragmentation and NAT Traversal support. Execute the configure script as shown below:

```
./configure --sysconfdir=/usr/local/etc/racoon --enable-hybrid --enable-frag --enable-dpd  
--enable-natt
```

Note: The last option line is only valid if the NAT Traversal kernel patch was applied

To compile and install the software, use the make command as shown below:

```
make  
make install
```

NetBSD 3.1

Once you have uploaded the latest stable IPsec Tools source code archive to the Gateway, you can extract the distribution using the tar utility as shown below:

```
tar zxvf <ipsec tools archive>
```

Note: If you downloaded a bzip archive instead of a gzip archive, use jxvf instead of zxvf with the tar command.

Next, change to the newly created directory as shown below:

```
cd <ipsec tools directory>
```

Before compiling the software, the configure script needs to be used to set some compile parameters and enable some advanced options. To view a list of all available options, execute the configure script with the --help switch as shown below:

```
./configure --help
```

The default install prefix for ipsec tools is /usr/local. NetBSD ships with an older version of ipsec tools installed in the /usr prefix and expects configuration files to exist under /etc. If you choose to overwrite the current version you will need to add some extra options to cope with these differences. You may choose to install the new version of ipsec tools in a different prefix but there may be issues associated with having multiple library versions installed in different paths. The other options shown below are to enable XAuth, Dead Peer Detection, IKE Fragmentation and NAT Traversal support. To overwrite the existing version, execute the configure script as shown below:

```
./configure --prefix=/usr --sysconfdir=/etc/racoon --enable-hybrid --enable-frag --enable-dpd  
--enable-natt
```

To compile and install the software, use the make command as shown below:

```
make  
make install
```

Configuring IPsec Tools

An IPsec Client uses the IKE protocol to establish communications with a VPN Gateway. There are several open source products that can be used with a Gateway that implement the IKE protocol. We will be using the IPsec Tools daemon ("racoon") which provides advanced functionality for VPN Client connectivity. For racoon to be useful, you will need to build a detailed configuration that contains your desired operating parameters.

Configuration Directory

The racoon configuration files are typically stored in a subdirectory named racoon under the etc system configuration

directory. The location of your etc system subdirectory will be operating system dependent. If you use Linux or NetBSD, the path would typically be /etc/racoon. If you are using FreeBSD, the path would typically be /usr/local/etc/racoon as the /etc system configuration directory is reserved for the base operating system and pre-installed software.

When racoon is started, it will attempt to load the main configuration file named racoon.conf from the /etc/racoon subdirectory. You can optionally specify an alternate path to the configuration file by using the -f command line option when starting racoon as shown below:

```
racoon -f <alternate path>/racoon.conf
```

If racoon encounters a problem while loading the configuration file, it will report the error as command line output.

Main Configuration File

For our test environment, we will use the configuration file shown below. This document will describe the example configuration file parameters and how they are used to facilitate VPN Client connectivity. You can use this example as a starting point when writing a configuration file for your VPN Gateway. For a complete reference of the racoon.conf file format, please see the racoon.conf manual page which is distributed with the IPsec Tools package.

```
#
# Path Specifications
#
path pre_shared_key "/etc/racoon/psk.txt";
path certificate "/etc/racoon/cert";

#
# Listen Section
#
listen
{
    isakmp 10.1.1.1 [500];
    isakmp_natt 10.1.1.1 [4500];
}

#
# Timer Section
#
timer
{
    natt_keepalive 15 seconds;
}

#
# Remote Section
#
remote anonymous
{
    exchange_mode aggressive;

    verify_identifier on;
    my_identifier fqdn "vpngw.shrew.net";
    peers_identifier fqdn "client.shrew.net";

    generate_policy unique;
    ike_frag on;
    nat_traversal on;
    dpd_delay 30;

    proposal_check claim;
    lifetime time 24 hours;

    proposal
    {
        encryption_algorithm aes 256;
        hash_algorithm sha1;
        authentication_method xauth_psk_server;
        dh_group 5;
    }
}

#
# Mode Config Section
#
```



```

mode_cfg
{
    network4 192.168.254.1;
    pool_size 253;
    netmask4 255.255.255.0;
    split_network include 10.1.2.0/24;
    split_dns "shrew.net";
    auth_source system;
    auth_groups "vpn-user";
    group_source system;
    conf_source local;
    wins4 10.1.2.253;
    dns4 10.1.2.253;
    default_domain "shrew.net";
    banner "/usr/local/etc/racoon/motd";
}

#
# SA Info Section
#

sainfo anonymous
{
    lifetime time 3600 seconds;
    encryption_algorithm aes 256;
    authentication_algorithm hmac_md5,hmac_sha1;
    compression_algorithm none;
}

```

Example racon.conf configuration file.

Preshared Key Configuration File

If a PSK authentication method is used, you will need to create a preshared key text file and add the path to the listen section of your main configuration file. The preshared key file lists the remote peer identifier followed by a preshared secret value.

```

#
# PSK text file
#

client.shrew.net      mysharedsecret

```

Example psk.txt configuration file.

see also :

[Configuring IPsec Tools : Preshared Key File Path](#)

Path Specifications

Path Specification statements are global parameters used to configure any special file paths or directories that racoon may need to function correctly.

Configuration Example

In our configuration example, we use two path statements as shown below:

```

#
# Path Specifications
#

path pre_shared_key "/etc/racoon/psk.txt";
path certificate "/etc/racoon/cert";

```

Path statements from our example racoon.conf configuration file.

Preshared Key File Path

The preshared key file path statement specifies the location of the file that contains our preshared key values. This statement is required if you plan to use a preshared key mode for Client authentication.

Certificate Directory Path

The certificate directory path statement specifies the directory that will contain any RSA certificates. This statement is required if you plan to use an RSA mode for Client authentication.

Reference

This section specifies various paths used by racoon. When running in privilege separation mode, certificate and script paths are mandatory, and you need to restart racoon if you want to change them.

The following are valid statements:

path include <i>path</i>	Specifies a path to include a file
path pre_shared_key <i>file</i>	Specifies a file containing pre-shared key(s) for various ID(s). See Pre-shared key File.
path certificate <i>path</i>	Racoon will search this directory if a certificate or certificate request is received. If you run with privilege separation, racoon will refuse to use a certificate stored outside of this directory.
path backupsa <i>file</i>	Specifies a file to which SA information negotiated by racoon should be stored. Racoon will install SA(s) from the file when started with the -B flag. The file is growing because racoon simply adds SAs to it. You should maintain the file manually
path script <i>path</i>	Racoon will search this directory for scripts hooks. If you run with privilege separation, racoon will refuse to execute a script stored outside of this directory.
path pidfile <i>file</i>	Specifies a file to store the PID (process id). If path starts with / it is treated as an absolute path; otherwise, relative to VARRUN directory specified at compilation time. Default is racoon.pid.

Listen Section

A Listen Section is used to configure the network interfaces and protocols that racoon will use to communicate.

Configuration Example

In our configuration example, we define the listen section as shown below:

```
#
# Listen Section
#

listen
{
    isakmp 10.1.1.1 [500];
    isakmp_natt 10.1.1.1 [4500];
}
```

Listen section from our example racoon.conf configuration file.

Socket Bindings

An *isakmp* statement informs racoon that it should listen for IKE packets using a socket bound to the specified address and port. An *isakmp_natt* statement informs racoon that it should listen for IKE NAT Traversal packets using a socket bound to the specified address and port. You must define an *isakmp_natt* statement to support clients that use the NAT Traversal feature.

see also:

[VPN Client Configuration : Hostname or IP Address](#)

Reference

If no listen directive is specified, racoon will listen on all available interface addresses.

The following are valid statements:

<code>isakmp address [port];</code>	If this is defined, racoon will only listen to the specified address. The default port is 500, which is specified by IANA. You can provide more than one address definition.
<code>isakmp_natt address [port];</code>	Same as isakmp but also sets the socket options to accept UDP-encapsulated ESP traffic for NAT-Traversal. If you plan to use NAT-T, you should provide at least one address with port 4500, which is specified by IANA. There is no default.
<code>strict_address;</code>	Requires that all addresses for ISAKMP be bound. This statement will be ignored if you do not specify any address.
<code>adminsock path [owner group mode];</code>	Path, owner, and group are the socket path, owner, and group; they must be quoted. Defaults are /var/racoon/racoon.sock, UID 0, and GID 0. mode is the access mode in octal, default is 0600.
<code>adminsock disabled;</code>	This directive tells racoon to not listen on the admin socket.

Timer Section

A Timer Section is used to configure the various event timers used by racoon.

Configuration Example

In our configuration example, we define the timer section as shown below:

```
#
# Timer Section
#

timer
{
    natt_keepalive 15 seconds;
}
```

Timer section from our example racoon.conf configuration file.

NAT Traversal Keep Alive

A NAT Traversal keep alive value can be specified to enable keep alive packets. The value specifies how long racoon should wait between sending the packets. A keep alive packet is used to prevent a firewall or router from expiring a NAT state entry that is being used by a VPN Client. NAT state entries are typically expired when no matching packets have been seen during a configured time interval. If a NAT state expires, the Client could experience communications failure.

see also:

[VPN Client Configuration : Keep Alive Packet Rate](#)

Reference

Specifies various timer values.

The following are valid statements:

<code>counter number;</code>	The maximum number of retries to send. The default is 5.
<code>interval number timeunit;</code>	The interval to resend, in seconds. The default time is 10 seconds.
<code>persend number;</code>	The number of packets per send. The default is 1.
<code>phase1 number timeunit;</code>	The maximum time it should take to complete phase 1. The default time is 15 seconds.
<code>phase2 number timeunit;</code>	the maximum time it should take to complete phase 2. The default time is 10 seconds.

natt_keepalive <i>number timeunit</i> ;	interval between sending NAT-Traversal keep-alive packets. The default time is 20 seconds. Set to 0 to disable keep-alive packets..
---	---

Remote Section

A Remote section is used to configure the parameters required to establish an ISAKMP SA with a given Peer. It also contains some additional parameters that control some features that are useful for VPN Client connectivity.

Configuration Examples

Two configuration examples are shown below. The first can be used for PSK authentication and the second can be used for RSA.

```
#
# PSK XAuth Remote Section
#
remote anonymous
{
    exchange_mode aggressive;

    verify_identifier on;
    my_identifier fqdn "vpngw.shrew.net";
    peers_identifier fqdn "client.shrew.net";

    passive on;
    generate_policy unique;
    ike_frag on;
    nat_traversal on;
    dpd_delay 30;

    proposal_check claim;
    lifetime time 24 hours;

    proposal
    {
        encryption_algorithm aes 256;
        hash_algorithm sha1;
        authentication_method xauth_psk_server;
        dh_group 5;
    }
}
```

PSK XAuth Remote section from our example racoon.conf configuration file.

```
#
# RSA XAuth Remote Section
#
remote anonymous
{
    exchange_mode main;

    verify_identifier on;
    my_identifier asn1dn;
    peers_identifier asn1dn;
    certificate_type x509 "vpngw.crt" "vpngw.key";
    ca_type x509 "ca.crt";

    passive on;
    generate_policy unique;
    ike_frag on;
    nat_traversal on;
    dpd_delay 30;

    proposal_check claim;
    lifetime time 24 hours;

    proposal
    {
        encryption_algorithm aes 256;
        hash_algorithm sha1;
        authentication_method xauth_rsa_server;
        dh_group 5;
    }
}
```

RSA XAuth Remote section from our example racoon.conf configuration file.

Defining a Remote Section

A remote section can be defined using an address value or *anonymous* immediately following the remote keyword as shown below:

```
remote 1.2.3.4
{
...
}
```

Remote section for a specific peer.

```
remote anonymous
{
...
}
```

Remote section that would match any peer.

When IKE communications are initiated with your VPN Gateway, racoon will first attempt to select a remote section that matches the Peer address. If no remote section was found, racoon will select an *anonymous* remote section if available. A remote section that supports Site to Site VPN connectivity would specify the Peer Gateway address. A remote section that supports VPN Client connectivity would use *anonymous* as the Peer Client address will be unknown.

Exchange Modes

The allowed exchange modes can be defined as *main*, *aggressive* or a comma separated list of several modes. Due to the payload information contained in an initial aggressive mode packet, the hash algorithm and DH group parameters cannot be auto-negotiated between a VPN Client and a Gateway. It is recommended that main mode be used whenever possible.

see also:

[VPN Client Configuration : Exchange Type](#)

Identifier Verification

The identifier verification setting can be defined as *on* or *off*. If set to *off*, the Peer identity value will not be compared with the locally defined peer identifier value. It is recommended that identifier verification be enabled whenever possible.

Preshared Key Authentication

A remote section that supports preshared key authentication will always use an aggressive mode exchange with a Fully Qualified Domain Name or a User Fully Qualified Domain Name identification type. The IKE protocol standard only allows an address identity type to be used with the combination of main mode and preshared key authentication. Using an identity value to lookup the preshared key would be impossible since the address value will be unknown until an initial packet is received. This prevents main mode and preshared key authentication from being used with a dynamically addressed Client.

In our example, the exchange mode is defined as aggressive and two Fully Qualified Domain Name identifier values are specified. The *my_identifier* value will be sent to the Peer for validation. The *peers_identifier* value will be used to validate the identity value received from the Peer. Lastly, the proposal is defined using the *xauth_psk_server* authentication method.

see also:

[Client Authentication : Preshared Key Authentication](#)

[VPN Client Configuration : Authentication Settings](#)

RSA Authentication

A remote section that supports RSA authentication can use a main or aggressive mode exchange with an *asn1dn* identification type. Racoon should verify the identity value against the Peer certificate subject value which prevents you from using an alternate identity type.

In our example, the exchange mode is defined as main and both the local and Peer identifier types are specified as *asn1dn*. When the *asn1dn* type is used and no value is specified, racoon will automatically use the subject name from the certificate as an identification value. The *certificate_type* statement is used to specify the Gateway certificate information which includes the certificate format, the certificate file name and the certificate private key file name. The *ca_type*

statement is used to specify the certificate authority that will be used to verify the client certificate which includes the certificate authority format and the certificate authority file name. Lastly, the proposal is defined using the *xauth_rsa_server* authentication method.

see also:

[Client Authentication : RSA Authentication](#)
[VPN Client Configuration : Authentication Settings](#)

Passive

The passive setting can be defined as either *on* or *off*. If set to *on*, racoon will not attempt to initiate negotiations with a remote peer. A remote section that supports VPN Client connectivity should enable passive mode whenever possible.

Policy Generation

The policy generation setting can be defined as *require*, *unique* or *off*. When defined as *require* or *unique*, racoon will generate a policy on behalf of a peer during phase 2 negotiations if an appropriate policy does not already exist. Using *unique* instead of *require* will force racoon to generate a separate security association for each outbound policy used by a client. When defined as *off*, policy generation will not be performed. A remote section that supports VPN Client connectivity should use *unique* policy generation.

IKE Fragmentation

The IKE fragmentation setting can be defined as either *on* or *off*. If set to *on*, racoon will allow the IKE fragmentation to be used which is helpful when communicating with a client behind a troublesome firewall or router. It is recommended that IKE fragmentation be enabled whenever possible.

see also:

[VPN Client Configuration : IKE Fragmentation Mode](#)

NAT Traversal

The NAT Traversal setting can be defined as either *on* or *off*. If set to *on*, racoon will allow the NAT Traversal to be used which is helpful when communicating with a client behind a firewall or router that performs Network Address Translation. It is recommended that NAT Traversal be enabled whenever possible.

see also:

[VPN Client Configuration : NAT Traversal Mode](#)

Dead Peer Detection

The DPD delay value can be defined to enable Dead Peer Detection. The value specifies how many seconds racoon should wait between attempts to validate that a Client is still connected to the Gateway. Dead Peer Detection is useful in preventing unnecessary Gateway resources being dedicated to a Client connection that is no longer active. It is recommended that DPD be enabled whenever possible.

see also:

[VPN Client Configuration : Enable Dead Peer Detection](#)

Proposal Check

The proposal check setting can be defined as *obey*, *strict*, *claim* or *exact*. If *obey* is used, a responder will always use the initiators lifetime value. If *strict* is used, a responder will use the initiators lifetime value if it is shorter than the responders. A responder will reject the proposal if the initiators value is greater than the responders. If *claim* is used, a responder will use the initiators lifetime value if it is shorter than the responders. A responder will use its own value if it is shorter than the initiators. In the second case, the responder will send a responder lifetime notification to the initiator when responding to a phase 2 proposal. If *exact* is used, a responder will reject the proposal if the initiators value is not equal to the responders. It is recommended that proposal check be set to *claim* whenever possible.

Lifetime

The lifetime setting is defined as a value and a time unit of *seconds*, *minutes* or *hours*. The value specifies the ISAKMP

Security Association lifetime and will be used to compare with the Peer proposed lifetime during phase 1 negotiations.

see also:

[VPN Client Configuration : Key Life Time Limit](#)

Proposal Subsection

A remote section must define at least one proposal subsection that contains a list of statements. The statements define the parameters used to negotiate an ISAKMP SAs with Peers during phase 1 negotiations.

The proposal encryption algorithm can be defined as *aes*, *3des*, *blowfish*, *cast128* or *des*.

see also:

[VPN Client Configuration : Cipher Algorithm](#)

[VPN Client Configuration : Cipher Key Length](#)

The proposal hash algorithm can be defined as *sha1* or *md5*.

see also:

[VPN Client Configuration : Hash Algorithm](#)

The proposal authentication method can be defined as *pre_shared_key*, *rsasig*, *hybrid_rsa_server*, *xauth_rsa_server* or *xauth_psk_server*. It is recommended that the authentication method be set to hybrid or an xauth mode to enable user based authentication.

see also:

[VPN Client Configuration : Authentication Method](#)

The proposal dh group setting can be defined as *1*, *2*, *5*, *14* or *15*.

see also:

[VPN Client Configuration : DH Exchange](#)

Reference

```
remote (address | anonymous) [[port]] [inherit parent] { statements }
```

Specifies the IKE phase 1 parameters for each remote node. The default port is 500. If anonymous is specified, the statements apply to all peers which do not match any other remote directive.

Sections with inherit parent statements (where parent is either an address or the anonymous keyword) have all values predefined to those of a given parent. In these sections, it is enough to redefine only the changed parameters.

The following are valid statements:

<code>exchange_mode (main aggressive base);</code>	Defines the exchange mode for phase 1 when racoon is the initiator. It also means the acceptable exchange mode when racoon is responder. More than one mode can be specified by separating them with a comma. All of the modes are acceptable. The first exchange mode is what racoon uses when it is the initiator.
<code>doi ipsec_doi;</code>	Means to use IPsec DOI as specified in RFC 2407. You can omit this statement.
<code>my_identifier [qualifier] idtype ...;</code>	Specifies the identifier sent to the remote host and the type to use in the phase 1 negotiation. <code>address</code> , <code>fqdn</code> , <code>user_fqdn</code> , <code>keyid</code> , and <code>asn1dn</code> can be used as an idtype. The qualifier is currently only used for keyid, and can be either file or tag. The possible values are: <code>my_identifier address [address];</code> The type is the IP address. This is the default type if you do not specify an

	<p>my_identifier user_fqdn string; identifier to use. The type is a USER_FQDN (user fully-qualified domain name).</p> <p>my_identifier fqdn string; The type is a FQDN (fully-qualified domain name)</p> <p>my_identifier keyid [file] file; The type is a KEY_ID, read from the file.</p> <p>my_identifier keyid tag string; The type is a KEY_ID, specified in the quoted string.</p> <p>my_identifier asn1dn [string]; The type is an ASN.1 distinguished name. If string is omitted, racoon will get the DN from the Subject field in the certificate.</p>
peers_identifier idtype ...;	Specifies the peer's identifier to be received. If it is not defined then racoon will not verify the peer's identifier in ID payload transmitted from the peer. If it is defined, the behavior of the verification depends on the flag of verify_identifier. The usage of idtype is the same as my_identifier except that the individual component values of an asn1dn identifier may be specified as * to match any value (e.g. "C=XX, O=MyOrg, OU=*, CN=Mine"). Alternative acceptable peer identifiers may be specified by repeating the peers_identifier statement.
verify_identifier (on off);	If you want to verify the peer's identifier, set this to on. In this case, if the value defined by peers_identifier is not the same as the peer's identifier in the ID payload, the negotiation will fail. The default is off.
certificate_type certspec;	Specifies a certificate specification. certspec is one of the following: <p>x509 certfile privkeyfile; certfile means a file name of a certificate. privkeyfile means a file name of a secret key.</p> <p>plain_rsa privkeyfile; privkeyfile means a file name of a private key generated by plainrsa-gen. Required for RSA authentication.</p>
ca_type cacertspec;	Specifies a root certificate authority specification. cacertspec is one of the following: <p>x509 cacertfile; cacertfile means a file name of the root certificate authority. Default is /etc/openssl/cert.pem</p>
xauth_login [string];	Specifies the login to use in client-side Hybrid authentication. It is available only if racoon has been built with this option. The associated password is looked up in the pre-shared key files, using the login string as the key id.
mode_cfg (on off);	Gather network information through ISAKMP mode configuration. Default is off.
weak_phase1_check (on off);	Tells racoon to act on unencrypted deletion messages for phase 1. This is a small security risk, so the default is off, meaning that racoon will keep on trying to establish a connection even if the user credentials are wrong, for instance.
peers_certfile (dnssec certfile plain_rsa pubkeyfile);	If dnssec is defined, racoon will ignore the CERT payload from the peer, and try to get the peer's certificate from DNS instead. If certfile is defined, racoon will ignore the CERT payload from the peer, and will use this certificate as the peer's certificate. If plain_rsa is defined, racoon will expect pubkeyfile to be the peer's public key that was generated by plainrsa-gen.
script script phase1_up; script script phase1_down;	Shell scripts that get executed when a phase 1 SA goes up or down. Both scripts get either phase1_up or phase1_down as first argument, and the following variables are set in their environment: <p>LOCAL_ADDR The local address of the phase 1 SA.</p> <p>LOCAL_PORT The local port used for IKE for the phase 1 SA.</p> <p>REMOTE_ADDR The remote address of the phase 1 SA.</p> <p>REMOTE_PORT The remote port used for IKE for the phase 1 SA.</p>

	<p>The following variables are only set if mode_cfg was enabled:</p> <table> <tr> <td>INTERNAL_ADDR4</td> <td>An IPv4 internal address obtained by ISAKMP mode config.</td> </tr> <tr> <td>INTERNAL_NETMASK4</td> <td>An IPv4 internal netmask obtained by ISAKMP mode config.</td> </tr> <tr> <td>INTERNAL_CIDR4</td> <td>An IPv4 internal netmask obtained by ISAKMP mode config, in CIDR notation.</td> </tr> <tr> <td>INTERNAL_DNS4</td> <td>The first internal DNS server IPv4 address obtained by ISAKMP mode config.</td> </tr> <tr> <td>INTERNAL_DNS4_LIST</td> <td>A list of internal DNS servers IPv4 address obtained by ISAKMP mode config, separated by spaces.</td> </tr> <tr> <td>INTERNAL_WINS4</td> <td>The first internal WINS server IPv4 address obtained by ISAKMP mode config.</td> </tr> <tr> <td>INTERNAL_WINS4_LIST</td> <td>A list of internal WINS servers IPv4 address obtained by ISAKMP mode config, separated by spaces</td> </tr> <tr> <td>SPLIT_INCLUDE</td> <td>The space separated list of IPv4 addresses and masks (address slash mask) that define the networks to be encrypted (as opposed to the default where all the traffic should be encrypted); obtained by ISAKMP mode config.</td> </tr> <tr> <td>SPLIT_LOCAL</td> <td>The space separated list of IPv4 addresses and masks (address slash mask) that define the networks to be considered local, and thus excluded from the tunnels; obtained by ISAKMP mode config.</td> </tr> <tr> <td>DEFAULT_DOMAIN</td> <td>The DNS default domain name obtained by ISAKMP mode config.</td> </tr> </table>	INTERNAL_ADDR4	An IPv4 internal address obtained by ISAKMP mode config.	INTERNAL_NETMASK4	An IPv4 internal netmask obtained by ISAKMP mode config.	INTERNAL_CIDR4	An IPv4 internal netmask obtained by ISAKMP mode config, in CIDR notation.	INTERNAL_DNS4	The first internal DNS server IPv4 address obtained by ISAKMP mode config.	INTERNAL_DNS4_LIST	A list of internal DNS servers IPv4 address obtained by ISAKMP mode config, separated by spaces.	INTERNAL_WINS4	The first internal WINS server IPv4 address obtained by ISAKMP mode config.	INTERNAL_WINS4_LIST	A list of internal WINS servers IPv4 address obtained by ISAKMP mode config, separated by spaces	SPLIT_INCLUDE	The space separated list of IPv4 addresses and masks (address slash mask) that define the networks to be encrypted (as opposed to the default where all the traffic should be encrypted); obtained by ISAKMP mode config.	SPLIT_LOCAL	The space separated list of IPv4 addresses and masks (address slash mask) that define the networks to be considered local, and thus excluded from the tunnels; obtained by ISAKMP mode config.	DEFAULT_DOMAIN	The DNS default domain name obtained by ISAKMP mode config.
INTERNAL_ADDR4	An IPv4 internal address obtained by ISAKMP mode config.																				
INTERNAL_NETMASK4	An IPv4 internal netmask obtained by ISAKMP mode config.																				
INTERNAL_CIDR4	An IPv4 internal netmask obtained by ISAKMP mode config, in CIDR notation.																				
INTERNAL_DNS4	The first internal DNS server IPv4 address obtained by ISAKMP mode config.																				
INTERNAL_DNS4_LIST	A list of internal DNS servers IPv4 address obtained by ISAKMP mode config, separated by spaces.																				
INTERNAL_WINS4	The first internal WINS server IPv4 address obtained by ISAKMP mode config.																				
INTERNAL_WINS4_LIST	A list of internal WINS servers IPv4 address obtained by ISAKMP mode config, separated by spaces																				
SPLIT_INCLUDE	The space separated list of IPv4 addresses and masks (address slash mask) that define the networks to be encrypted (as opposed to the default where all the traffic should be encrypted); obtained by ISAKMP mode config.																				
SPLIT_LOCAL	The space separated list of IPv4 addresses and masks (address slash mask) that define the networks to be considered local, and thus excluded from the tunnels; obtained by ISAKMP mode config.																				
DEFAULT_DOMAIN	The DNS default domain name obtained by ISAKMP mode config.																				
send_cert (on off);	If you do not want to send a certificate, set this to off. The default is on.																				
send_cr (on off);	If you do not want to send a certificate request, set this to off. The default is on.																				
verify_cert (on off);	<p>By default, the identifier sent by the remote host (as specified in its my_identifier statement) is compared with the credentials in the certificate used to authenticate the remote host as follows:</p> <table> <tr> <td>Type asn1dn:</td> <td>The entire certificate subject name is compared with the identifier, e.g. "C=XX, O=YY, ...".</td> </tr> <tr> <td>Type address, fqdn, or user_fqdn:</td> <td>The certificate's subjectAltName is compared with the identifier.</td> </tr> </table> <p>If the two do not match, the negotiation will fail. If you do not want to verify the identifier using the peer's certificate for some reason, set this to off.</p>	Type asn1dn:	The entire certificate subject name is compared with the identifier, e.g. "C=XX, O=YY, ...".	Type address, fqdn, or user_fqdn:	The certificate's subjectAltName is compared with the identifier.																
Type asn1dn:	The entire certificate subject name is compared with the identifier, e.g. "C=XX, O=YY, ...".																				
Type address, fqdn, or user_fqdn:	The certificate's subjectAltName is compared with the identifier.																				
lifetime time number timeunit;	Define a lifetime of a certain time which will be proposed in the phase 1 negotiations. Any proposal will be accepted, and the attribute(s) will not be proposed to the peer if you do not specify it (them). They can be individually specified in each proposal.																				
ike_frag (on off force);	Enable receiver-side IKE fragmentation if racoon has been built with this feature. If set to on, racoon will advertise itself as being capable of receiving packets split by IKE fragmentation. This extension is there to work around broken firewalls that do not work with fragmented UDP packets. IKE fragmentation is always enabled on the sender-side, and it is used if the peer advertises itself as IKE fragmentation capable. By selecting force, IKE Fragmentation will be used when racoon is acting as the initiator even before the remote peer has advertised itself as IKE fragmentation capable.																				

esp_frag fraglen;	This option is only relevant if you use NAT traversal in tunnel mode. Its purpose is to work around broken DSL routers that reject UDP fragments, by fragmenting the IP packets before ESP encapsulation. The result is ESP over UDP of fragmented packets instead of fragmented ESP over UDP packets (i.e., IP:UDP:ESP:frag(IP) instead of frag(IP:UDP:ESP:IP)). fraglen is the maximum size of the fragments. 552 should work anywhere, but the higher fraglen is, the better the performance.
initial_contact (on off);	Enable this to send an INITIAL-CONTACT message. The default value is on. This message is useful only when the implementation of the responder chooses an old SA when there are multiple SAs with different established time, and the initiator reboots. If racoon did not send the message, the responder would use an old SA even when a new SA was established. The KAME stack has the switch in the system wide value net.key.preferred_oldsa. When the value is zero, the stack always uses a new SA.
passive (on off);	If you do not want to initiate the negotiation, set this to on. The default value is off. It is useful for a server.
proposal_check level;	<p>specifies the action of lifetime length, key length, and PFS of the phase 2 selection on the responder side, and the action of lifetime check in phase 1. The default level is strict. If the level is:</p> <p>obey The responder will obey the initiator anytime.</p> <p>strict If the responder's lifetime length is longer than the initiator's one, or if responder's key length is shorter than the initiator's one, the responder uses the initiator's. Otherwise it rejects the proposal. If PFS is not required by the responder, the responder will obey the proposal. If PFS is required by both sides and if the responder's group is not equal to the initiator's, then the responder will reject the proposal.</p> <p>claim If the responder's length is longer than the initiator's, or if responder's key length is shorter than the initiator's one, the responder will use the initiator's. If the responder's length is shorter than the initiator's, the responder uses its own length AND sends a RESPONDER-LIFETIME notify message to an initiator in the case of lifetime (phase 2 only). For PFS, this directive behaves the same as strict.</p> <p>exact If the initiator's lifetime or key length is not equal to the responder's, the responder will reject the proposal. If PFS is required by both sides and if the responder's group is not equal to the initiator's, then the responder will reject the proposal.</p>
support_proxy (on off);	If this value is set to on, then both values of ID payloads in the phase 2 exchange are always used as the addresses of end-point of IPsec-SAs. The default is off.
generate_policy (on off require unique);	This directive is for the responder. Therefore you should set passive to on in order that racoon only becomes a responder. If the responder does not have any policy in SPD during phase 2 negotiation, and the directive is set to on, then racoon will choose the first proposal in the SA payload from the initiator, and generate policy entries from the proposal. It is useful to negotiate with clients whose IP address is allocated dynamically. Note that an inappropriate policy might be installed into the responder's SPD by the initiator, so other communications might fail if such policies are installed due to a policy mismatch between the initiator and the responder. The on and require values mean the same thing (generate a require policy). unique tells racoon to set up unique policies, with a monotonic increasing reqid number (between 1 and IPsec_MANUAL_REQID_MAX). This directive is ignored in the initiator case. The default value is off.
nat_traversal (on off force);	This directive enables use of the NAT-Traversal IPsec extension (NAT-T). 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). Possible

	<p>values are:</p> <p>on NAT-T is used when a NAT gateway is detected between the peers.</p> <p>off NAT-T is not proposed/accepted. This is the default.</p> <p>force NAT-T is used regardless of whether NAT is detected between the peers or not.</p> <p>Please note that NAT-T support is a compile-time option. Although it is enabled in the source distribution by default, it may not be available in your particular build. In that case you will get a warning when using any NAT-T related config options.</p>
dpd_delay delay;	This option activates the DPD and sets the time (in seconds) allowed between 2 proof of liveness requests. The default value is 0, which disables DPD monitoring, but still negotiates DPD support.
dpd_retry delay;	If dpd_delay is set, this sets the delay (in seconds) to wait for a proof of liveness before considering it as failed and sending another request. The default value is 5.
dpd_maxfail number;	If dpd_delay is set, this sets the maximum number of proof of liveness to request (without reply) before considering the peer dead. The default value is 5.
nonce_size number;	define the byte size of nonce value. Racoon can send any value although RFC2409 specifies that the value MUST be between 8 and 256 bytes. The default size is 16 bytes.
ph1id number;	An optional number to identify the remote proposal and to link it only with sainfos who have the same number. Defaults to 0.
proposal { sub-substatements }	<p>encryption_algorithm algorithm; Specifies the encryption algorithm used for the phase 1 negotiation. This directive must be defined. algorithm is one of following: des, 3des, blowfish, cast128, aes, camellia for Oakley. For other transforms, this statement should not be used.</p> <p>hash_algorithm algorithm; Defines the hash algorithm used for the phase 1 negotiation. This directive must be defined. algorithm is one of following: md5, sha1, sha256, sha384, sha512 for Oakley.</p> <p>authentication_method type; Defines the authentication method used for the phase 1 negotiation. This directive must be defined. type is one of: pre_shared_key, rsa (for plain RSA authentication), gssapi_krb, hybrid_rsa_server, hybrid_rsa_client, xauth_rsa_server, xauth_rsa_client, xauth_psk_server or xauth_psk_client.</p> <p>dh_group group; Defines the group used for the Diffie-Hellman exponentiations. This directive must be defined. group is one of following: modp768, modp1024, modp1536, modp2048, modp3072, modp4096, modp6144, modp8192. Or you can define 1, 2, 5, 14, 15, 16, 17, or 18 as the DH group number. When you want to use aggressive mode, you must define the same DH group in each proposal.</p> <p>lifetime time number timeunit; Defines the lifetime for the phase 1 SA proposal. Refer to the description of the lifetime directive defined in the remote directive.</p> <p>gss_id string; Defines the GSS-API endpoint name, to be included as an attribute in the SA if the gssapi_krb authentication method is used. If this is not defined, the default value of 'host/hostname' is used, where hostname is the value returned by the hostname command.</p>

Mode Config Section

A Mode Config Section is used to configure extended authentication operation and the settings provided to the client during the configuration exchange.

Configuration Example

In our configuration example, we define the Mode Config section as shown below:

```
#
# Mode Config Section
#

mode_cfg
{
    auth_source system;
    auth_groups "vpn-user";
    group_source system;

    conf_source local;
    network4 192.168.254.1;
    pool_size 253;
    netmask4 255.255.255.0;

    dns4 10.1.2.253;
    default_domain "shrew.net";
    split_dns "shrew.net";
    wins4 10.1.2.253;

    split_network include 10.1.2.0/24;

    banner "/usr/local/etc/racoon/motd";
}
```

User Authentication

The user authentication settings can be defined to configure the XAuth operating parameters. The *auth_source* statement specifies the account source that is used to perform authentication. When set to *system*, racoon will use the local unix password database. The *auth_groups* statement prevents an XAuth authentication from succeeding unless a user is a member of one of the specified groups. This is useful when you have a centralized account database and would like to limit remote access to only a subset of the users. The *group_source* specifies the group account source that is used to perform group inclusion checking. When set to *system*, racoon will use the local unix group database.

see also:

[Client Authentication : Extended Authentication](#)

[Client Configuration : Authentication Method](#)

Network Configuration

The network configuration settings can be defined to automatically provide a private address to a Client operating in Virtual Adapter mode. The *conf_source* statement specifies the configuration source that is used for address and netmask assignment. When set to *local*, racoon will manage its own address pool. The other possible options, *radius* and *ldap*, are not covered in this document. The *network4* statement specifies the base address for the pool. The *pool_size* statement specifies the available number of addresses in the pool. The *netmask4* statement specifies the network mask to be assigned to the client virtual interface.

see also:

[Client Management : Private Address Configuration](#)

[Client Configuration : Address Method](#)

Name Service Configuration

The name service configuration can be defined to automatically provide DNS and WINS settings to a Client. The *dns4* statement specifies a list of DNS server addresses to be used by the Client. The *default_domain* statement specifies the default domain to be used by the client when submitting DNS requests. The *split_dns* statement specifies a list of DNS

suffixes to be used by the Client. This is useful if you would like the Client to selectively forward DNS requests to a tunnel defined DNS server. The *wins4* statement specifies a list of WINS server addresses to be used by the Client.

see also:

[Client Management : Name Services Configuration](#)

[Client Configuration : Name Resolution Setting](#)

Split Network Configuration

The split network configuration can be defined to provide a client with Split Tunnel information. The *split_network* statement specifies either the *include* or *local_lan* modifier followed by a list of target networks. If the *include* modifier is used, the client will only tunnel packets destined to the specified target networks. If the *local_lan* modifier is used, the client will only tunnel packets not destined to the specified target networks.

see also:

[Client Management : Split Tunnel Configuration](#)

[Client Configuration : Automatic Policy Configuration](#)

Reference

mode_cfg { statements }

Defines the information to be returned for a remote hosts' ISAKMP modeconfig request. Also defines the authentication source for remote peers authenticating through Xauth.

The following are valid statements:

auth_source (system radius pam ldap);	Specifies the source for authentication of users through Xauth. system means to use the Unix user database. This is the default. radius means to use a RADIUS server. It works only if racoon was built with libradius support, and the configuration is done in radius.conf. pam means to use PAM. It works only if racoon was built with libpam support. ldap means to use LDAP. It works only if racoon was built with libldap support, and the configuration is handled by adding statements to the ldapcfg section.
auth_groups group1, ...;	Specifies the group memberships for Xauth in quoted group name strings. When defined, the authenticating user must be a member of at least one group for Xauth to succeed.
group_source (system ldap);	Specifies the source for group validation of users through Xauth. system means to use the Unix user database. This is the default. ldap means to use LDAP. It works only if racoon was built with libldap support, and the configuration is handled by adding statements to the ldapcfg section.
conf_source (local radius ldap);	Specifies the source for IP addresses and netmask allocated through ISAKMP mode config. local means to use the local IP pool defined by the network4 and pool_size keywords. This is the default. radius means to use a RADIUS server. It works only if racoon was built with libradius support, and the configuration is done in radius.conf. RADIUS configuration requires RADIUS authentication. ldap means to use an LDAP server. It works only if racoon was built with libldap support, and the configuration is done in the ldapcfg section. LDAP configuration requires LDAP authentication.
accounting (none system radius pam);	Enables or disables accounting for Xauth logins and logouts. Default is none, which disables accounting. system enables system accounting through utmp. radius enables RADIUS accounting. It works only if racoon was built with libradius support, and the configuration is done in radius.conf. RADIUS accounting require RADIUS authentication. pam enables PAM accounting. It works only if racoon was built with libpam support. PAM accounting requires PAM authentication.
pool_size size	Specifies the size of the IP address pool, either local or allocated through RADIUS. conf_source selects the local pool or the RADIUS configuration, but in both configurations, you cannot have more than size users connected at the same time. The default is 255.
network4 address; netmask4 address;	The local IP pool base address and network mask from which dynamically allocated IPv4 addresses should be taken. This is used if conf_source is set to local or if the RADIUS server returned 255.255.255.254. Default is 0.0.0.0/0.0.0.0.

auth_source (system radius pam ldap);	Specifies the source for authentication of users through Xauth. system means to use the Unix user database. This is the default. radius means to use a RADIUS server. It works only if racoon was built with libradius support, and the configuration is done in radius.conf. pam means to use PAM. It works only if racoon was built with libpam support. ldap means to use LDAP. It works only if racoon was built with libldap support, and the configuration is handled by adding statements to the ldapcfg section.
dns4 addresses;	A list of IPv4 addresses for DNS servers, separated by commas, or on multiple dns4 lines.
nbns4 addresses;	A list of IPv4 address for WINS servers.
split_network (include local_lan) network/mask, ...;	The network configuration to send, in cidr notation (e.g. 192.168.1.0/24). If include is specified, the tunnel should only be used to encrypt the indicated destinations. Otherwise, if local_lan is used, everything will pass through the tunnel but the indicated destinations.
default_domain domain;	The default DNS domain to send.
split_dns domain, ...;	The split dns configuration to send, in quoted domain name strings. This list can be used to describe a list of domain names for which a peer should query a modecfg assigned dns server. DNS queries for all other domains would be handled locally. (Cisco VPN client only).
banner path;	The path of a file displayed on the client at connection time. Default is /etc/motd.
auth_throttle delay;	On each failed Xauth authentication attempt, refuse new attempts for a set delay of one or more seconds. This is to avoid dictionary attacks on Xauth passwords. Default is one second. Set to zero to disable authentication delay.
pfs_group group;	Sets the PFS group used in the client proposal (Cisco VPN client only). Default is 0.
save_passwd (on off);	Allow the client to save the Xauth password (Cisco VPN client only). Default is off.

SA Info Section

An SA Info Section is used to configure the parameters required to establish an IPsecSA with a given Peer.

Configuration Example

In our configuration example, we define the listen section as shown below:

```
#
# SA Info Section
#
sainfo anonymous
{
    lifetime time 3600 seconds;
    encryption_algorithm aes 256;
    authentication_algorithm hmac_md5,hmac_sha1;
    compression_algorithm deflate;
}
```

Defining an SA Info Section

An SA Info section can be defined using the local and remote ids to be negotiated immediately following the sainfo keyword. An id is specified using an address, subnet value and protocol or *anonymous* as shown below:

```
sainfo subnet 10.1.2.0/24 any anonymous
{
    ...
}
```

Semi Anonymous SA Info section that matches the local 10.1.2.0/24 subnet for any Peer.

```
remote anonymous
{
    ...
}
```

Fully Anonymous SA Info section that would match any local subnet for any Peer.

For a Client to negotiate an IPsec security association, racoon must first locate a SA Info section that matches the

requested local private network id. By specifying Semi-Anonymous SA Info sections, an administrator can limit what networks a Peer can communicate with. If Extended Authentication is used, an ldap or system group database can be used to limit access to certain networks based on the users group membership as shown below:

```
sainfo subnet 10.1.3.0/24 any anonymous group "vpn-admin"
{
  ...
}
```

Semi Anonymous SA Info section that matches the local 10.1.3.0/24 subnet for an XAuth Peer that authenticated using a user account with vpn-admin group membership.

Reference

sainfo (*source_id* | anonymous) (*destination_id* | anonymous) from idtype [*string*] [*group string*] { statements }

defines the parameters of the IKE phase 2 (IPsec-SA establishment). *source_id* and *destination_id* are constructed like:

address *address* [/ *prefix*] [[*port*]] *ul_proto*

or

subnet *address* [/ *prefix*] [[*port*]] *ul_proto*

The idtype modifier allows you to specify the exact content of ID payload to be matched (source is the local end, destination is the remote end). This is not like a filter rule. For example, if you define 3ffe:501:4819::/48 as *source_id*, 3ffe:501:4819:1000:/64 will not match.

In case of longest prefix (selecting single host) address instructs to send ID type of ADDRESS, while subnet instructs to send ID type of SUBNET. Otherwise these instructions are identical.

The anonymous keyword can be used to match any id.

The group keyword allows an xauth group membership check to be performed for this sainfo section. When the mode_cfg auth source is set to system or ldap, the xauth user is verified to be a member of the specified group before allowing a matching sa to be negotiated.

The following are valid statements:

pfs_group group;	define the group of Diffie-Hellman exponentiations. If you do not require PFS then you can omit this directive. Any proposal will be accepted if you do not specify one. group is one of following: modp768, modp1024, modp1536, modp2048, modp3072, modp4096, modp6144, modp8192. Or you can define 1, 2, 5, 14, 15, 16, 17, or 18 as the DH group number.
lifetime time number timeunit;	define how long an IPsec-SA will be used, in timeunits. Any proposal will be accepted, and no attribute(s) will be proposed to the peer if you do not specify it(them). See the proposal_check directive.
remoteid number;	Sainfos will only be used if their remoteid matches the ph1id of the remote section used for phase 1. Defaults to 0, which is also the default for ph1id.
encryption_algorithm algorithms;	des, 3des, des_iv64, des_iv32, rc5, rc4, idea, 3idea, cast128, blowfish, null_enc, twofish, rijndael, aes, camellia (used with ESP)
authentication_algorithm algorithms;	des, 3des, des_iv64, des_iv32, hmac_md5, hmac_sha1, hmac_sha256, hmac_sha384, hmac_sha512, non_auth (used with ESP authentication and AH)
compression_algorithm algorithms;	deflate (used with IPComp)

Racoon does not have a list of security protocols to be negotiated. The list of security protocols are passed by SPD in the kernel. Therefore you have to define all of the potential algorithms in the phase 2 proposals even if there are algorithms which will not be used. These algorithms are defined by using the following three directives, with a single comma as the separator. For algorithms that can take variable-length keys, algorithm names can be followed by a key length, like "blowfish 448". Racoon will compute the actual phase 2 proposals by computing the permutation of the specified algorithms, and then combining them with the security protocol specified by the SPD. For example, if des, 3des,

hmac_md5, and hmac_sha1 are specified as algorithms, we have four combinations for use with ESP, and two for AH. Then, based on the SPD settings, racoon(8) will construct the actual proposals. If the SPD entry asks for ESP only, there will be 4 proposals. If it asks for both AH and ESP, there will be 8 proposals. Note that the kernel may not support the algorithm you have specified.

Generating RSA Credentials

If you are not using one of the PSK authentication modes, RSA credentials will need to be generated for the VPN gateway and possibly the Client as well. The only RSA authentication method that does not require Client credentials to operate is the Hybrid Authentication Method.

To generate RSA credentials, use the openssl tool to create a certificate authority, a private key and a signed certificate. Although the detailed use of the openssl command line tool is beyond the scope of this document, here is an example of how RSA server credentials might be created ...

```
mkdir certs
mkdir -p demoCA/newcerts
touch demoCA/index.txt
echo "00" > demoCA/serial

umask 077
openssl genrsa > certs/ca.key
openssl genrsa > certs/vpngw.key

umask 022
openssl req -days 1825 -x509 -new -key certs/ca.key > certs/ca.crt
openssl req -new -key certs/vpngw.key > certs/vpngw.csr
openssl ca -in certs/vpngw.csr -keyfile certs/ca.key \
  -cert certs/ca.crt -out certs/vpngw.crt
```

After the server credentials have been created, you will need to move the server certificate and private key files to the certificate path specified in your racoon configuration file. The certificate authority public certificate should be given to each user that will be connecting to the gateway.

Creating a Firewall Rule Set

Packet Filtering

When a Client Gateway is Internet facing, it is typical to have firewall software running as well. It is important to remember that you must make allowances for IPsec Client related traffic.

For example, suppose a gateway is configured using our example racoon configuration file. Firewall rules must be added to allow clients to communicate with the gateway.

pf example:

```
pass in proto udp from any to self port 500
pass in proto udp from any to self port 4500
pass in proto esp from any to self
```

ip tables example:

```
iptables -A INPUT -j ACCEPT -p udp --dport 500
iptables -A INPUT -j ACCEPT -p udp --dport 4500
iptables -A INPUT -j ACCEPT -p esp
```

When a client connects, it will establish an IPsec SA to allow traffic from itself to the private network. Suppose that a connected client was assigned a private address of 10.99.99.1 and attempts to communicate with the 10.100.100.0/24 network. If the client can transmit packets but receives no response, one likely cause is that the gateway firewall is blocking the traffic. A rule could be added which allows the client address range to communicate with the private network.

pf example:

```
pass quick from 10.99.99.0/24 to 10.100.100.0/24
```



```
pass quick from 10.100.100.0/24 to 10.99.99.0/24
```

ip tables example:

```
iptables -A FORWARD -j ACCEPT -s 10.99.99.0/24 -d 10.100.100.0/24  
iptables -A FORWARD -j ACCEPT -s 10.100.100.0/24 -d 10.99.99.0/24
```

NOTE : These are just example rules to illustrate the point. An actual rule set should be written with much tighter security in mind.

Packet Fragmentation

Some firewalls require special handling for packet fragments. For instance, using pf or ipf on a BSD Gateway would require special features to be used to handle packet fragments in certain situations.

For pf, it may be necessary to use the 'scrub all fragment reassemble' option to handle VPN related traffic.

For ipf, it may be necessary to use the 'keep frags' modifier when specifying packet filtering rules for VPN related traffic.

VPN Client Configuration

Before the VPN Client can be used to access remote network resources, it must first be configured to communicate with a VPN Gateway.

List of available topics:

1. [VPN Access Manager](#)

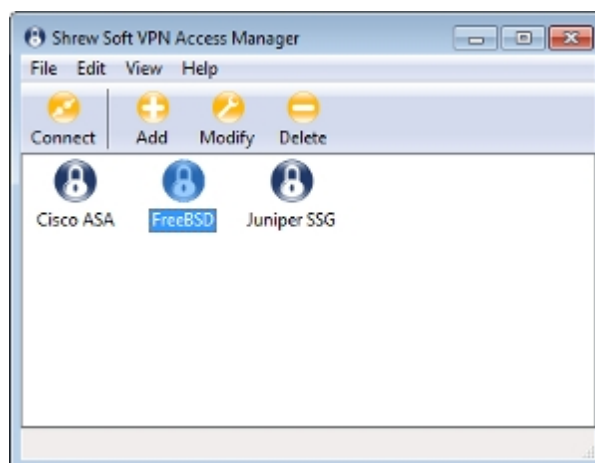
VPN Access Manager

The VPN Access Manager used to manage Site Configurations. It is also used to launch the VPN Connect application for a given site. To open the VPN Access Manager, use the start menu icon installed under the Shrew Soft VPN Client group.

List of available sub topics:

1. [Main Window](#)
2. [User Preferences](#)
3. [Site Configurations](#)

Main Window



The VPN Access Manager with a single Site Configuration Defined.

Using the Access Manager

Connecting to a Site

To initiate a connection, highlight the site you wish to connect to and click the Connect toolbar button or select Connect

from the dropdown File Menu. You may alternately right click on the site you wish to connect to and select Connect from the popup context menu. Double clicking on the site icon will also perform the same action.

Creating a Site Configuration

To create a new site configuration, click the Add toolbar button or select Add from the dropdown Edit Menu. You may alternately right click inside the site list window and select Add from the popup context menu. When the Site Configuration dialog appears, enter all the required settings for your new site. After you have finished, a new icon will be added to the site configuration list.

Modifying a Site Configuration

To modify an existing new site configuration, highlight the site you wish to modify and click the Modify toolbar button or select Modify from the dropdown Edit Menu. You may alternately right click on the site you wish to modify and select Properties from the popup context menu. When the Site Configuration dialog appears, modify all the required settings for the existing site.

Deleting a Site Configuration

To delete an existing site configuration, highlight the site you wish to delete and click the Delete toolbar button or select Delete from the dropdown Edit Menu. You may alternately right click on the site you wish to delete and select Delete from the popup context menu. Confirmation is required before a site is permanently deleted.

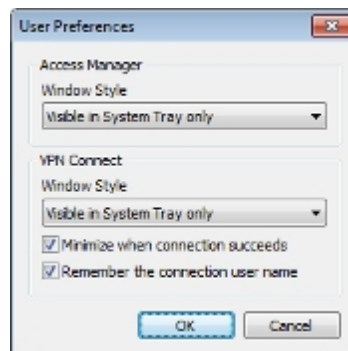
Exporting a Site Configuration

To export an existing site configuration to a file, highlight the site you wish to export and select the Export option from the dropdown File Menu. You will be prompted to select a location and file name for the exported configuration.

Importing a Site Configuration

To import an existing site configuration from a file, select the Import option from the dropdown File Menu. You will be prompted for the path and file name of the configuration to be imported.

User Preferences



Access Manager Preferences

This dialog allows the Access Manager application to be customized based on the user preferences.

Window Style

Visible in Task Bar and System Tray	The application will be visible in both the Task Bar and System Tray. When minimized, the Task Bar or System tray icon can be used to re-display the window.
Visible in System Tray Only	The application will be visible in only the System Tray. When minimized, the System tray icon can be used to re-display the window.
Visible in Task Bar Only	The application will be visible in only the Task Bar. When minimized, the Task Bar icon can be used to re-display the window.

VPN Connect Preferences

This dialog allows the VPN Connection application to be customized based on the user preferences.

Window Style

Visible in Task Bar and System Tray	The application will be visible in both the Task Bar and System Tray. When minimized, the Task Bar or System tray icon can be used to re-display the window.
Visible in System Tray Only	The application will be visible in only the System Tray. When minimized, the System tray icon can be used to re-display the window.
Visible in Task Bar Only	The application will be visible in only the Task Bar. When minimized, the Task Bar icon can be used to re-display the window.

Minimize When Connection Succeeds

When this option is enabled, the client will automatically minimize to the Task Bar or System Tray after a connection succeeds.

Remember the Connection User Name

When this option is enabled, the client will automatically remember the last user name entered for the connection. If the connection window is closed and then re-opened, the user name will automatically be entered.

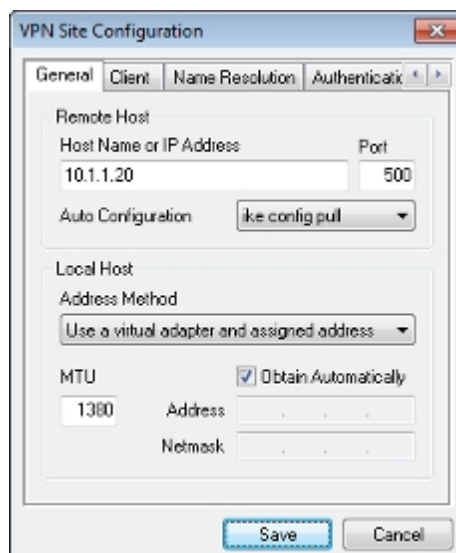
Site Configurations

The Site Configuration dialog is used when Creating or Modifying site configurations within the VPN Access Manager. The Site Configuration dialog contains several Tabs that are used to group settings together. Navigate between these tabs to access all the available settings.

The Tab interface is used to divide the settings into the following categories:

1. [General Settings](#)
2. [Client Settings](#)
3. [Name Resolution Settings](#)
4. [Authentication Settings](#)
5. [Phase 1 Settings](#)
6. [Phase 2 Settings](#)
7. [Policy Settings](#)

General Settings



Remote Host

The Remote Host settings are used to define the basic VPN Gateway operation.

Host Name or IP Address

Enter the host name or IP address for the VPN Client Gateway that will service this site. The use of a host name instead of static IP address is recommended when non-address Peer identifiers are used. This allows the public gateway address to be modified without invalidating Client Site Configurations.

Port

Enter the UDP port that the VPN Client Gateway is using for IKE services. The default value for this setting is UDP port 500.

Configuration Method

Select the method used to automatically configure client settings. The method chosen should match the method supported by your VPN Gateway. The default value for this setting is Pull as it is the only method supported by the IPsec Tools racoon daemon.

Two Configuration Methods are available:

Disabled	This method disables all automatic client configuration.
IKE Configuration Pull	This method allows the client to request settings from a Gateway during the Configuration Exchange. If the Gateway supports the Pull method, it will attempt to return a list of settings that it supports for client use.
IKE Configuration Push	This method allows a Gateway to offer settings to the Client during the Configuration Exchange. If the client is configured to use the Push method, it will return a list of settings that it has accepted for use.
DHCP Over IPsec	This method allows the client to request settings from a Gateway using the DHCP over IPsec configuration method.

PLEASE NOTE: The DHCP over IPsec option is considered experimental.

Local Host

The Local Host settings are used to define the basic VPN Client operation.

Address Method

Select the method used to address the private network traffic.

Two Address Methods are available.

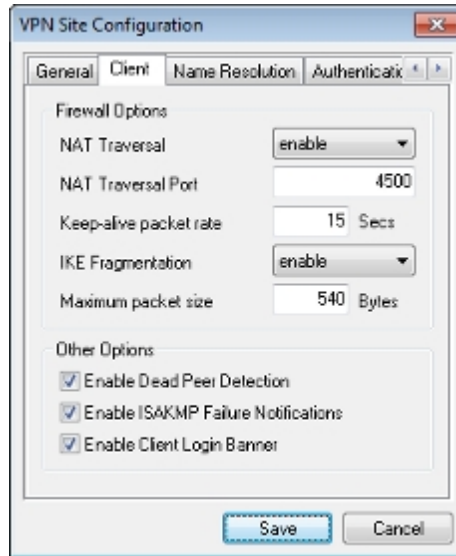
Virtual Adapter Mode	This mode allows the client to use a virtual adapter and a gateway assigned address.
Direct Adapter Mode	This mode allows the client to use an existing adapter and its current address.

Adapter MTU

When the client is set to use Virtual Adapter Mode, the adapter MTU can be specified. Communications problems caused by IP fragmentation issues can be resolved by setting the MTU to a lower value.

Adapter Address and Netmask

When a remote gateway is configured to support the Configuration Exchange, it can be configured to assign a valid client address and netmask automatically. If not, uncheck the Obtain Automatically option and enter a valid Client Address and Client Netmask for this site configuration.



Firewall Options

The Firewall Options settings are used to define what features will be enabled to prevent problems from occurring when a Firewall or NAT router exists between the Client and a Gateway.

NAT Traversal Mode

Set this value to Enable or Force if you want the VPN Client IPsec Daemon to use the IKE and ESP NAT Traversal protocol extensions.

Disable	The NATT protocol extensions will not be used.
Enable	The NATT protocol extensions will only be used if the VPN Gateway indicates support during negotiations and NAT is detected.
Force Draft	The Draft version of the NATT protocol extensions will be used regardless of whether or not the VPN Gateway indicates support during negotiations or NAT is detected.
Force RFC	The RFC version of the NATT protocol extensions will be used regardless of whether or not the VPN Gateway indicates support during negotiations or NAT is detected.

NAT Traversal Port

Enter the UDP port that the VPN Client Gateway is using for NAT-T services. The default value for this setting is UDP port 4500.

Keep-Alive Packet Rate

Enter the rate at which the Client IPsec Daemon should send NAT-T Keep alive packets. Keep-alive packets can help prevent problems from occurring when a Firewall or NAT exists between the VPN Client and the Peer Gateway. The default value for this setting is 30 seconds.

IKE Fragmentation Mode

Enable this option if you would like the VPN Client to use the IKE Fragmentation protocol extension.

Disable	The IKE Fragmentation protocol extension will not be used.
Enable	The IKE Fragmentation protocol extension will only be used if the VPN Gateway indicates support during negotiations.
Force	The IKE Fragmentation protocol extension will be used regardless of whether or not the VPN Gateway indicates support during negotiations.

Maximum Packet Size

When the Fragment Packets option is enabled, this value specifies the largest non-fragmented IKE packet size allowed. If a packet size is larger than this value, IKE fragmentation is performed. The default setting for this value is 540 bytes.

Other Options

The Other Options settings is used to define the miscellaneous features that will be enabled by the VPN Client.

Enable Dead Peer Detection

Enable this option if you would like the VPN Client IPsec Daemon to use the Dead Peer Detection protocol extension. When the option is enabled, the protocol extension will only be used if the VPN Gateway also has support. This will allow the client and Gateway to detect when one side of the tunnel is no longer able to respond. The default value for this setting is Enabled.

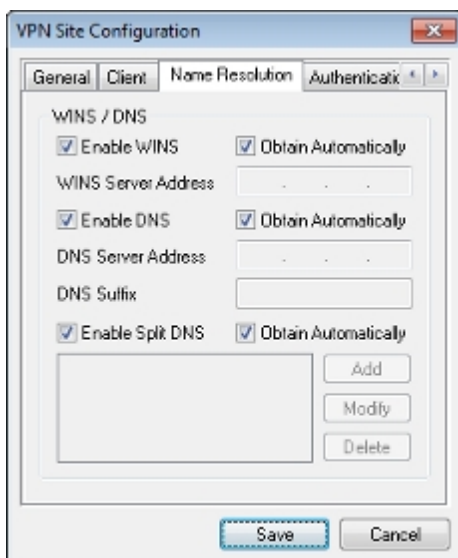
Enable Failure Notifications

Enable this option if you would like the VPN Client IPsec Daemon to forward ISAKMP failure notifications. The default value for this setting is Enabled.

Enable Client Login Banner

Enable this option if you would like the client to display a Login Banner after establishing a connection with the Gateway. The Gateway must support the Transaction Exchange and be configured to forward a login banner to the Client. The default value for this setting is Enabled.

Name Resolution Settings



WINS / DNS

When a remote gateway is configured to support the Configuration Exchange, it should be able to provide valid client DNS and WINS settings automatically. If not, uncheck the corresponding Obtain Automatically option for the DNS or WINS options and enter a valid DNS Server Address, DNS Suffix or WINS Server Address values for this site configuration. If specific DNS or WINS settings are not required for this site configuration, simply uncheck the Enable DNS or Enable WINS option.

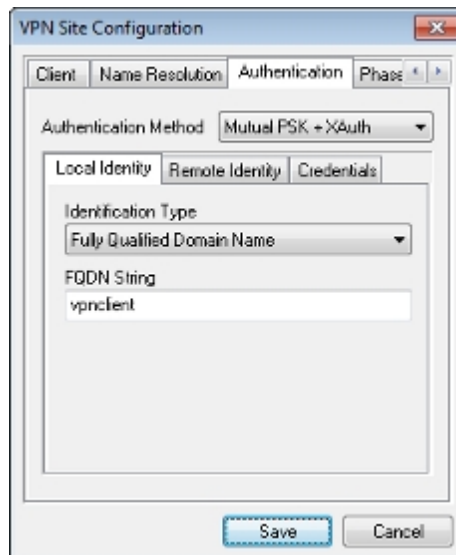
Please be aware that if the Enable DNS or Enable WINS options are used, a security policy may be required to communicate with these servers via the tunnel.

Split DNS

Enable Split DNS support if you would like to selectively send DNS requests to a tunnel specific DNS server. When this option is enabled, a client will inspect all DNS requests and compare them to a list of DNS domain name suffixes. If the request matches one of these domain name suffixes, the request is forwarded to the tunnel specific DNS server. Otherwise, the request is forwarded to a DNS server that is configured for your local network adapter.

When a remote gateway is configured to support the Configuration Exchange, it should be able to provide a valid list of DNS domain suffixes that should be resolved using the tunnel specific DNS server. If not, uncheck the corresponding Obtain Automatically option for Split DNS and enter a list of DNS domain suffixes manually.

Authentication Settings



Authentication Method

The authentication method describes how the Client and Gateway will perform Peer Authentication and Extended Authentication. The default value for this setting is Hybrid RSA + XAuth.

To select an Authentication Method, choose an option from the Authentication method drop down selection window. The behavior of an authentication option can be determined by interpreting the basic keywords that make up the option name.

Keywords and their meaning:

Hybrid	When a Hybrid Authentication mode is selected, it is not necessary to provide credentials for the client. Only the Gateway will be authenticated by the Client during phase 1 negotiations.
Mutual	When a Mutual Authentication mode is selected, it is necessary to provide credentials for both the Client and the Gateway. Both both parties will be authenticated during phase 1 negotiations.
RSA	When an RSA Authentication mode is selected, the provided credentials will be in the form of PEM or PKCS12 certificate files or key files. see also: Client Authentication : RSA Methods Configuring IPsec Tools : RSA Authentication
PSK	When a Pre Shared Key mode is used, the provided credentials will be in the form of a shared secret string. see also: Client Authentication : Preshared Key Methods Configuring IPsec Tools : Preshared Key Authentication

GRP	When a GRP Authentication mode is selected, the provided credentials will be in the form of a PEM or PKCS12 certificate file and a shared secret string. This mode is designed to interoperate with the Cisco proprietary "Mutual Group Authentication" method.
XAuth	When an Extended Authentication mode is selected, a user name and password to be authenticated by the Gateway after phase 1 has been completed.

Local and Remote Identities

To select an Identification Type, choose an option from the Identification Type drop down selection window. Not all options are available for all authentication modes.

Here is a list of the available options:

Any	When the Any option is selected (Remote Identity only), the client will accept any ID type and value. This should be used with caution as it bypasses part of the IKE phase1 identification process.
ASN.1 Distinguished Name	When the ASN.1 Distinguished Name ("ASN.1 DN") option is selected, the value will be automatically read from the PEM or PKCS12 certificate file. The Client will only allow this mode to be selected when an RSA Authentication mode is being used.
Fully Qualified Domain Name	When the Fully Qualified Domain Name ("FQDN") option is selected, you must provide a FQDN String in the form of a DNS domain string. For example, 'shrew.net' would be an acceptable value. The Client will only allow this option to be selected if a PSK Authentication mode is being used.
User Fully Qualified Domain Name	When the User Fully Qualified Domain Name ("UFQDN") option is selected, you must provide a UFQDN String in the form of a USER @ DNS domain string. For example, 'jdoe@shrew.net' would be an acceptable value. The Client will only allow this option to be selected if a PSK Authentication mode is being used.
IP Address	When the IP Address option is selected, the value is determined automatically by default. If you would like to use an address other than the adapter address used to communicate with the Client Gateway, simply uncheck the option and specify the Address String. The client will only allow this option to be selected if a PSK Authentication mode is being used.
Key Identifier	When the Key Identifier option is selected, you must provide an identifier string.

Authentication Credentials

There are four settings that can be used to specify credentials for a Site Configuration.

Server Certificate Authority File

This value is a path to a PEM or PKCS12 file that contains the Certificate Authority certificate and public key that was used to generate the Client Gateways certificate. This value is required when an RSA Authentication mode is selected.

Client Certificate File

This value is a path to a PEM or PKCS12 file that contains the certificate and public key that the client will use during phase 1 authentication. This value is required when a Mutual RSA Authentication mode is selected.

Client Certificate File

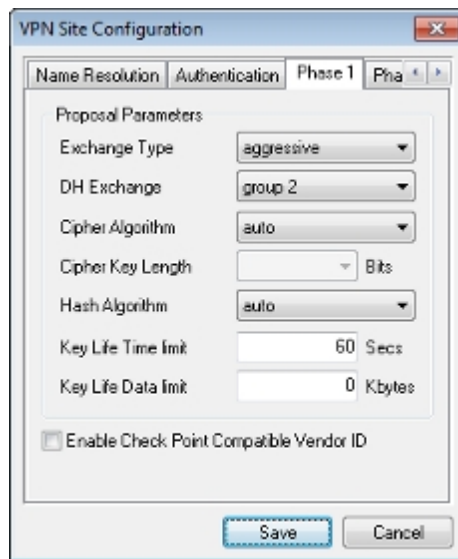
This value is a path to a PEM or PKCS12 file that contains the private key that the client will use during phase 1 authentication. This value is required when a Mutual RSA Authentication mode is selected.

Pre Shared Key

This value is a string that represents the Preshared Key that the client will use during phase 1 authentication. A

Preshared Key value must be 8 characters or more in length. This value is required when a Mutual PSK Authentication mode is selected.

Phase 1 Settings



The screenshot shows the 'VPN Site Configuration' dialog box with the 'Phase 1' tab selected. The 'Proposal Parameters' section contains the following settings:

Parameter	Value
Exchange Type	aggressive
DH Exchange	group 2
Cipher Algorithm	auto
Cipher Key Length	[Variable] Bits
Hash Algorithm	auto
Key Life Time limit	60 Secs
Key Life Data limit	0 Kbytes

There is an unchecked checkbox for 'Enable Check Point Compatible Vendor ID' and 'Save' and 'Cancel' buttons at the bottom.

Proposal Parameters

Exchange Type

Select the Exchange type to be used during phase 1 negotiations. The default value for this setting is aggressive.

DH Exchange

Select the DH exchange group description to be proposed during phase 1 negotiations. The default value for this setting is group 2.

Cipher Algorithm

Select the cryptographic Cipher Algorithm to be proposed during phase 1 negotiations. The default value for this setting is 3des (or Triple DES).

Cipher Key Length

Select the cryptographic Cipher Key Length to be proposed during phase 1 negotiations. Some Cipher Algorithms use a fixed key length. If one of these Ciphers are selected, this option will be grayed out. Other Cipher Algorithms have a variable key length which will need to be defined. The default value for this setting is variable depending on the selected Cipher Algorithm.

Hash Algorithm

Select the Hash Algorithm to be proposed during phase 1 negotiations. The default value for this setting is md5.

Key Life Time Limit

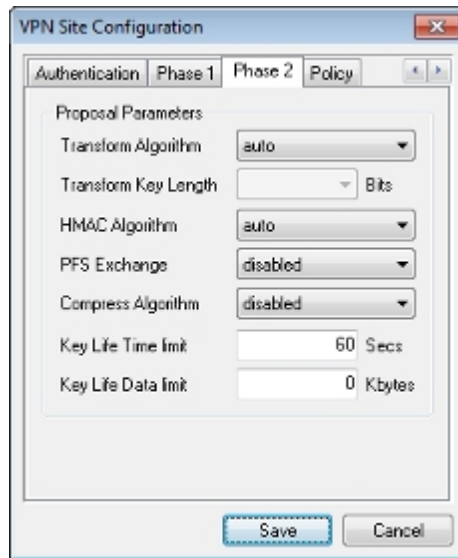
Enter the Key Life Time Limit to be proposed during phase 1 negotiations. This setting will determine the life time of an ISAKMP SA. The default value for this setting is 28800 Seconds.

Key Life Data Limit

Enter the Key Life Data Limit to be proposed during phase 1 negotiations. This setting will determine the number of kilobytes that can be protected by an ISAKMP SA. If a 0 value is specified, no life data limit is negotiated. The default value for this setting is 0.

PLEASE NOTE: This setting is offered for IKE compatibility only. ISAKMP SA data limits are not currently enforced by the Shrew Soft VPN Client.

Phase 2 Settings



Proposal Parameters

Transform Algorithm

Select the cryptographic Transform Algorithm to be proposed during phase 2 negotiations. The default value for this setting is esp-3des (or ESP Triple DES).

Transform Key Length

Select the cryptographic Transform Algorithm Key Length to be proposed during phase 2 negotiations. Some algorithms use a fixed key length. If one of these Transforms are selected, this option will be grayed out. Other algorithms have a variable key length which will need to be defined. The default value for this setting is variable depending on the selected Transform Algorithm.

HMAC Algorithm

Select the HMAC Algorithm to be proposed during phase 2 negotiations. The default value for this setting is md5.

Compression Algorithm

No Compression Algorithms are currently supported. This setting should be grayed out.

PFS Exchange

Select the PFS DH Exchange group description to be proposed during phase 2 negotiations. When a remote gateway is configured to support the Configuration Transaction Exchange, it should be able to assign a valid DH Exchange group for PFS automatically. The default value for this setting is Auto.

Key Life Time Limit

Enter the Key Life Time Limit to be proposed during phase 2 negotiations. This setting will determine the life time of an IPsec SA. The default value for this setting is 3600 Seconds.

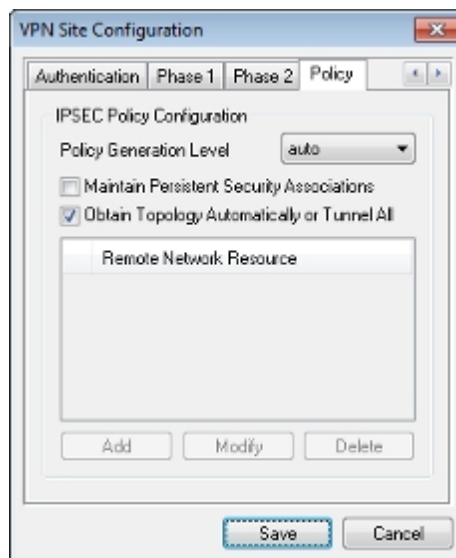
Key Life Data Limit

Enter the Key Life Data Limit to be proposed during phase 2 negotiations. This setting will determine the number of kilobytes that can be protected by an IPsec SA. If a 0 value is specified, no life data limit is negotiated. The default value

for this setting is 0.

PLEASE NOTE: This setting is offered for IKE compatibility only. IPsec SA data limits are not currently enforced by the Shrew Soft VPN Client.

Policy Settings



Policy Generation Level

The Policy Generation Level option modifies the level in which IPsec Policies are generated. The different levels provided map to IPsec SA negotiation behaviors implemented by different vendor implementations.

Auto	The client will attempt to automatically determine the appropriate IPsec Policy level. If the gateway offers a Cisco compatible vendor ID during phase1 negotiation, the client will select the shared level. If the gateway does not offer a Cisco compatible vendor ID, the client will select unique.
Require	The client will not negotiate a unique SA for each policy. Policies are generated using the local public address as the local policy ID and the Remote Network Resource as the remote policy ID. The phase2 proposal will use the policy IDs during negotiation.
Unique	The client will negotiate a unique SA for each policy. Policies are generated using the local public address as the local policy ID and the Remote Network Resource as the remote policy ID. The phase2 proposal will use the policy IDs during negotiation.
Shared	Policies are generated at the require level. However, the phase2 proposal will use the local policy ID as the local ID and Any (0.0.0.0/0) as the remote ID during negotiation.

PLEASE NOTE: The shared level is a custom option that doesn't match standard IPsec negotiation levels. It is intended to mimic the Cisco VPN client behavior. If your gateway offers a Cisco compatible vendor ID but is not an actual Cisco VPN gateway (ipsec-tools, NetGear and other gateways do this), you may need to manually select the require or unique level instead of auto.

Persistent Security Associations

The Maintain Persistent Security Associations option modifies the way in which IPsec SAs are negotiated with the peer. When disabled, the client only negotiates security associations when it needs to process a packet that matches a security policy. When enabled, the client will negotiate an SA for each policy configured immediately after it connects. It then attempts to immediately renegotiate replacement security associations as they expire.

PLEASE NOTE: Maintain Persistent Security Associations option considered experimental.

IPsec Policy Configuration

The Obtain Topology Automatically or Tunnel All options modifies the way security policies are configured for the connection. When disabled, Manual configuration must be performed. When enabled, Automatic configuration is performed.

Automatic Configuration

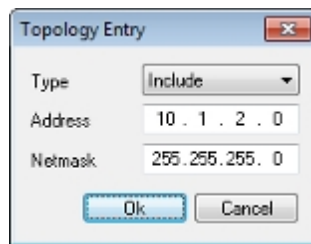
When a remote gateway is configured to support the Configuration Exchange, it provides a list of networks that are accessible via VPN Client Gateway. This network topology information, along with the client address are used to describe the security policies for this site configuration. When Automatic Policy Configuration is enabled but the remote Gateway does not supply topology information, the VPN Client will install a default policy that tunnels all traffic to the Gateway. The default value for this setting is Enabled.

Manual Configuration

The Network Topology List can be manually defined if the VPN Gateway does not provide a list automatically for the client. You can Add, Modify or Delete Network Topology List entries by using the buttons shown at the bottom of the Policy Configuration dialog. These buttons will be grayed out if the Automatic Policy Configuration option is Enabled.

The Topology Entry Dialog

The Topology Entry Dialog is used whenever you are adding or modifying a Network Topology List entry. Simply configure the Entry Type as either Include or Exclude and enter the required information. When you are finished editing the Topology Entry, press the OK button to accept the entry data or the Cancel button to discard it.



Entry Type

The Entry Type describes how the specified network should be accessed when the Client is connected to the VPN Gateway.

There are two options available:

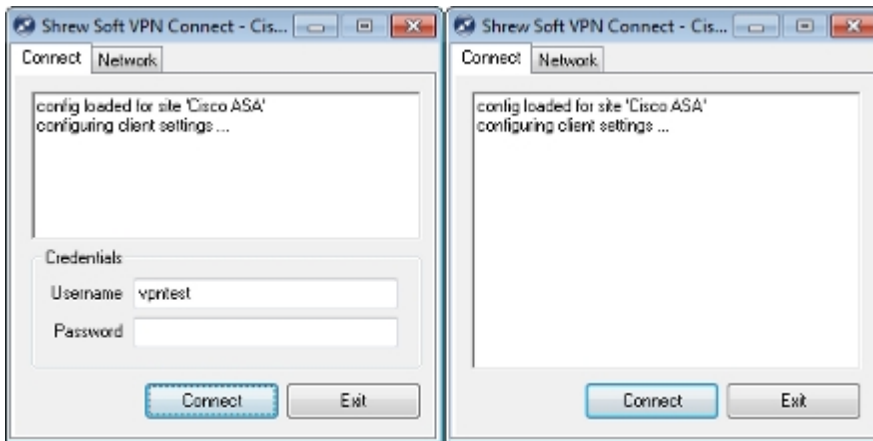
Include	Defines a network that should be accessed via the VPN Gateway.
Exclude	Defines a network that should be accessed via local connectivity.

Address and Netmask

The Address and Netmask are used to define the Network address and size of the network.

VPN Connect

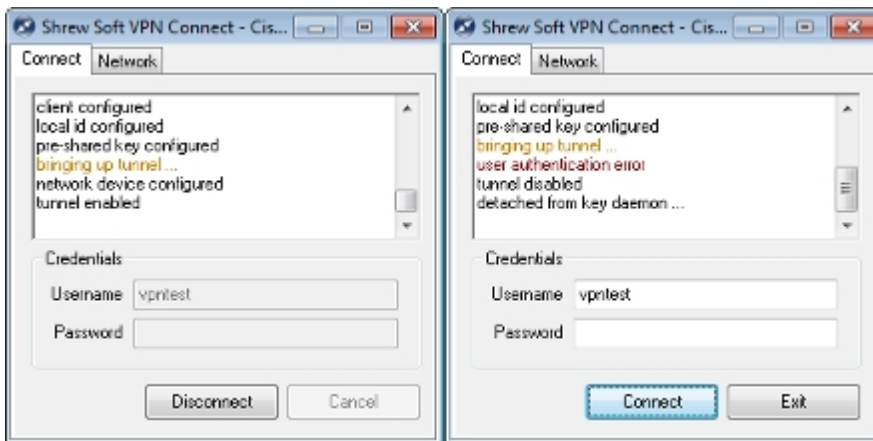
The VPN Connect application is a user interface component that was designed to interact with the IKE Daemon. To open a VPN Connect application instance, please use the VPN Access manager application.



The connect window may look different depending on the style of authentication that has been configured for the site.

Connecting to a site

When the user presses the 'Connect' button, the site configuration information is passed to the IKE Daemon along with a request to initiate communications. Messages regarding the connection state will be displayed in the status output window. If the connection succeeds, the 'Connect' button text will change to 'Disconnect'. Otherwise, an error message will be logged in the status window and the connection will be closed.



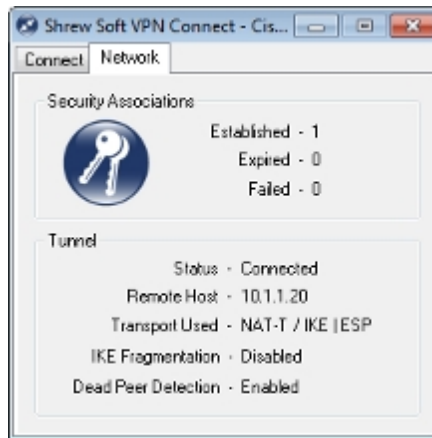
The Connect attempt succeeded for the window on the left but failed for the window on the right.

Disconnecting from a site

When the user presses the disconnect button, a request to terminate communications message will be passed to the IKE Daemon. This may take several seconds to complete as the IKE Daemon will make every attempt to terminate the communications gracefully and cleanup any resources it may have allocated before sending a final status message.

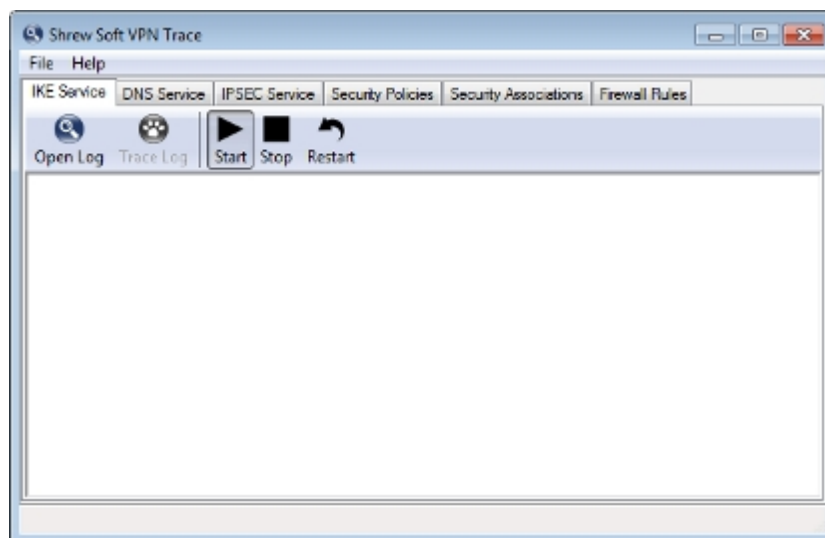
Network Statistics

By selecting the Network Tab, it is possible to view the current network statistics for the connection. This tab also details the current connection status, the network address for the remote peer, the negotiated network transport and the current number of established IPsec Security Associations.



VPN Trace

The VPN Trace application is a user interface component that was designed to view debug output from the client services as well as control the level of output generated. To open the VPN Trace Application, use the start menu icon installed under the Shrew Soft VPN Client group.



Opening and Tracing Log Output

To open a Service log output file, click the Open Log button in the toolbar. This automatically enables the Trace Log option as well. When the Trace Log option is enabled, any new data added to the log file is immediately displayed in the log output window. Disabling the Trace option is useful if you would like to pause and examine information that is already available.

Controlling the Daemon Services

To Start, Stop or Restart a Service, click the appropriate button in the toolbar. If the user currently logged in does not have the necessary Administrative Privileges to perform these actions, the toolbar buttons will be grayed out.

Viewing IPsec Security Policies

A list of active IPsec security policies are listed under the Security Policies Tab. The list is updated automatically when a client connects or disconnects from a VPN Gateway.

Viewing IPsec Security Associations

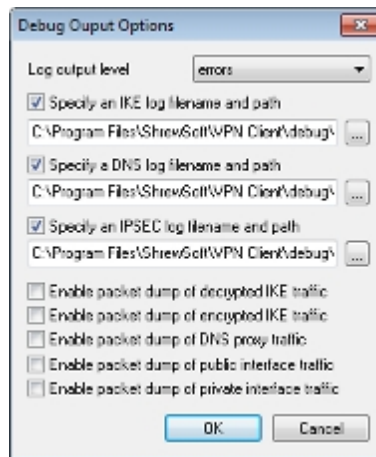
A list of active IPsec security associations are listed under the Security Associations Tab. The list is updated automatically when new associations are created or expired by the IPsec daemon during client operation.

Viewing Firewall Rules

A list of active VPN Client firewall rules are listed in the Firewall Rules Tab. These rules are managed by the different services installed by the VPN Client.

Debug Output Options

To view or modify the Debug Output Options, select Options from the dropdown File Menu. By changing these option values, you can control the level of debug information generated by the Client services.



Log Output Level

The log output level controls the level of debug output that is generated by the Services. After the output level has been modified, Services need to be restarted for the new setting to be used.

The possible values for this setting:

None	No messages are logged to the output file.
Errors	Only error messages are logged to the output file.
Informational	Error and Informational messages are logged to the output file.
Debug	Error, Informational and Debug messages are logged to the output file.
Loud	Error, Informational Debug and Loud Debug messages are logged to the output file.
Decode	Error, Informational, Debug, Loud Debug and Text Decode of binary data is logged to the output file

Enable Packet Dump of Decrypted IKE Traffic

When the Enable Packet Dump of Decrypted IKE Traffic option is enabled, the IKE Daemon will create a binary packet dump of the decrypted IKE conversation that takes place between the Client and the Client Gateway.

Enable Packet Dump of Encrypted IKE Traffic

When the Enable Packet Dump of Encrypted IKE Traffic option is enabled, the IKE Daemon will create a binary packet dump of the encrypted IKE conversation that takes place between the Client and the Client Gateway.

Enable Packet Dump of DNS Proxy Traffic

When the Enable Packet Dump of DNS Proxy Traffic option is enabled, the DNS Proxy Daemon will create a binary packet dump of all DNS packets it inspects.

Enable Packet Dump of Public Interface Traffic

When the Enable Packet Dump of Public Interface Traffic option is enabled, the IKE Daemon will create a binary packet dump of IKE conversation that takes place between the Client and the Client Gateway.

Enable Packet Dump of Private Interface Traffic

When the Enable Packet Dump of Private Interface Traffic option is enabled, the IPsec Daemon will create a binary packet dump of the traffic before outbound IPsec processing and after inbound IPsec processing.

Viewing Debug Output

Client debug output is stored under a directory named debug below the VPN Client installation directory. All log and packet dump files are stored in this location by default. The information stored in this directory is often helpful for a developer to review when attempting to resolve an issue.

Packet dumps are recorded in the PCAP file format and can be viewed using the Wire Shark Traffic Analyzer (formerly Ethereal) which has support for IKE and IPsec packet decode. For more information regarding the Wire Shark Traffic Analyzer, please visit their [homepage](#).