Virtual private networks, Part 1

Presented by developerWorks, your source for great tutorials

ibm.com/developerWorks

Table of Contents

If you're viewing this document online, you can click any of the topics below to link directly to that section.

Section 1. Tutorial tips

Should I take this tutorial?

This tutorial is aimed at technical folks who want to understand the overall workings of a virtual private network or VPN. It is a survey of the VPN field, not an in-depth analysis. However, knowledge of basic networking concepts is recommended.

Part 1 starts with a high-level VPN overview, proceeds to the technologies involved, and delves into the IPSec protocol, while Part 2 will take a closer look at this technology and examine some VPN implementations of note.

About the author

For questions about the content of this tutorial, please contact the author, Larry Loeb, at *larryloeb@prodigy.net* .

Larry Loeb has written for many of the last century's major "dead tree" computer magazines, having been -- among other things -- a consulting editor for *BYTE* magazine and senior editor for the launch of *WebWeek*. He's been online since uucp "bang" addressing (where the world existed relative to !decvax), serving as editor of the *Macintosh Exchange* on BIX, and the *VARBusiness Exchange*. He's also written a book on the Secure Electronic Transaction Internet protocol. His first Mac had 128K of memory. His first 1130 had 4K, as did his first 1401.

Section 2. Introduction

VPNs defined I

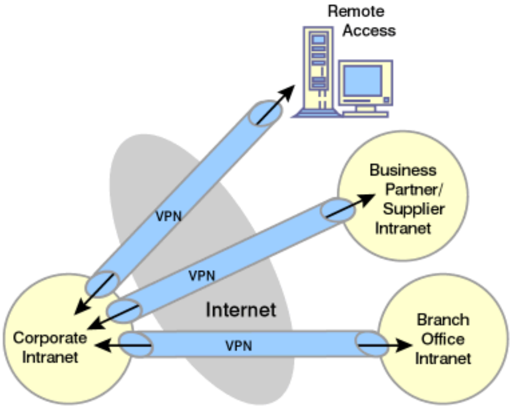
VPNs have become something of a religious icon for the networked user who is concerned about the secure transmission of sensitive data. Where private networks once consisted of secured leased lines that were access-protected by a unique user name and a password, VPNs generally operate over a known non-secure network.

VPNs encrypt data transmitted from an access point before it hits the network, and decrypt all received traffic before it gets to the user. This basic functionality, not the medium the VPN spans, is the true definition of a VPN. The rest is merely details of implementation.

VPNs defined II

Figure 1 shows what the overall topology of a VPN might look like from a host's point of view. The radial data paths of differing business functions emanate from a central host source.

Figure 1



VPNs defined III

Secured networks are usually implemented in hardware that delivers security at a level below the network layer. VPNs act in the network layer and so can utilize any security built into a network. But a VPN should be designed to function without any reliance on the network's assumed security.

There has been some confusion about the difference between a VPN and a DPN. A dedicated private network (DPN) is simply a wide area network (WAN) that has specifiable data paths. If one built a local area network (LAN) with an ATM switch instead of a wiring hub, that would be a DPN. DPNs have data paths that can be specified, and may differ from the physical wiring. This means fewer patch panels, but nothing for the overall system security.

The VPN data path

A typical end-to-end VPN data path could contain:

* Several machines not under control of the corporation (for example, the ISP access box in a dial-in segment and the routers within the Internet).
* A security gateway (firewall or router) that is located at the boundary between an internal segment and an external segment.
* An internal segment (intranet) that contains hosts and routers. Some could be malicious, and some will carry a mix of intracompany and intercompany traffic.
* An external segment (Internet) that carries traffic not only from your company's network but also from other sources.

Once again, a VPN can never trust the network's security. It must make its own.

Section 3. The VPN technology

Underlying technologies

If we use the standard layering model, the network layer is the lowest we can go and still provide end-to-end security. Network-layer security protocols can provide blanket protection for all upper-layer application data carried in the payload of an Internet Protocol (IP) datagram without requiring a user to modify the applications. This transparency of use is quite important for seamlessness across platforms.

Internet-based networking: IPSec One VPN framework is IP Security Architecture (IPSec), an open source framework defined by the IPSec Working Group of the IETF.

IPSec's Working Group has defined protocols in major areas of concern. These are: data origin authentication, data integrity, replay protection, crypto key management, and data confidentiality. The IETF has come up with specific protocols for each of these areas, and the frameworks with which to apply them.

IPSec protocols

The principal IPSec protocols are:

* **IP Authentication Header (AH):** This provides data origin authentication, data integrity, and replay protection. These three functions are collectively known as *authentication*. Data integrity comes from the checksum generated by a message authentication code like MD5, data origin authentication from a shared secret key in the data to be authenticated, and replay protections come from a sequence number in the AH.
* **IP Encapsulating Security Payload (ESP):** This provides data confidentiality, data origin authentication, data integrity, and replay protection. While ESP and AH can both provide authentication, data integrity checking, and replay protection, only ESP can do encryption. When used for authentication, ESP will use the AH algorithms. ESP and AH can be combined or nested.
* **Internet Security Association and Key Management Protocol (ISAKMP):** This protocol provides a method for automatically setting up security associations between sites and managing their cryptographic keys.

More on ISAKMP

To actually process AH or ESP, systems need to have their base assumptions synchronized. A security association (SA) contains all the security assumptions, such as the cryptography to be used, keying information, and party identities. The ISAKMP protocol actually covers how these SAs negotiate all this in an automated way.

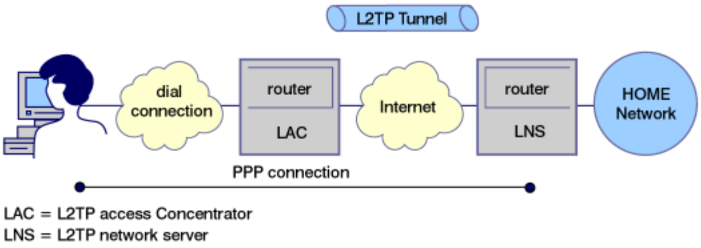
Automating this process allows a VPN to scale without manual intervention. This scalability is one of IPSec's main attractions. ISAKMP deals with the initial key exchanges through the Internet Key Exchange (IKE) key management protocol. (Previously, this was called ISAKMP/Oakley but has been superceded.) IKE is mandatory, because managing keys is one of the most potentially damaging events as far as security is concerned. An initial exchange of keys is a stricture point where an eavesdropper might be able to launch a man-in-the-middle (MITM) attack or just steal information about the keys. ISAKMP authenticates the parties in the exchange before it exchanges any key information that could be of use to an attacker. When IKE does exchange key information, it encrypts the information before it hits the network.

Leased lines and L2TP

Leased line VPNs are usually dial-up in nature and are used by companies that already have the needed multiple points-of-presence available.

Layer 2 Tunnel Protocol (L2TP) was created by the IETF to extend a PPP connection past the connecting ISP's POP, all the way to the corporate gateway. Because PPP can run different transport protocols than IP, the remote host can appear to be on the same subnet as the corporate gateway with PPP tunneling the gateway protocol under the IP protocol. Figure 2 shows one L2TP scenario, with the disparate elements of the actual network connection shown.

Figure 2



L2TP security I

L2TP is not very robust in the security area, because it assumes that the security of the remote (and the wire that connects the remote to the gateway) is as good as the corporate network. That can only be true in very limited cases.

There are other security concerns besides the network, as well.

The tunnel endpoints are authenticated in L2TP, but not the packets themselves. Therefore, L2TP can be easily spoofed (thinking that it's talking to an IP address it's not really in communication with) and is vulnerable to MITM attacks.

Denial of Service (DoS) attacks are possible by terminating the underlying PPP connection (or the IP tunnel) by injecting false control messages into the datastream.

L2TP security II

The limitations of the PPP protocol can come into play in L2TP. PPP packet payloads are encrypted, but keys are not refreshed. This implies that someone listening to the traffic for a long enough time can launch a dictionary-style attack against the key, and thus gain the data.

In practice, an L2TP tunnel is an L2TP frame inside a UDP packet. Because UDP is an IP protocol, we can apply IPSec to increase the security of the dial-up VPN. This comes at the cost of administering IPSec on a phone line; but L2TP is too easily compromised not to consider adding a measure of security through IPSec or another custom IP security framework.

IPSec's interactions I

Applying IPSec to a VPN may be more complicated than it first seems, due to IPSec's interactions with other system components. Administration of a VPN's policy and configuration functions may require the use of **Lightweight Directory Access Protocol (LDAP)** to access a directory server. Application-level encryption such as that offered by **Secure Sockets Layer (SSL)** may prove complementary to that offered by IPSec, as we will discuss soon. However, some other components may prove problematic.

IPSec's interactions II

Network Address Translation (NAT) maps internal IP addresses and external ones. NAT can be implemented in a firewall or router, and is used to hide address information from the external network. But because it changes address information in the IP datagram, the NAT-ed packet will fail the integrity check of IPSec's AH protocol.

One way around this is for all network elements to use public addressing, thus obviating the need for NAT.

Another would be for the VPN to tunnel everything, which would hide non-public addresses from the network. But this will confuse any packet filters used in the network, perhaps at the router or firewall.

IPSec's interactions III

If IPSec is used in a VPN, packet filtering as a method of access control will fail due to the encryption used. But the encrypting AH protocol may serve as access control in the packet filter's stead. The cryptography of AH is robust enough to implement an authentication strategy.

A packet filter can still be a part of the network, but with simplified rules (compared to what they would be otherwise) of filtering. Also, packet filters may be very useful on certain network legs, such as those between the gateway end of the tunnel and the destination host.

Network layer versus application layer I

Some applications build in security at the application level, like SSL in most Web browsers. This can protect data traffic and is usually transparent to the user. But in general, the security at the application level can be variable, and depends on the developer's implementation.

It should not be forgotten that application-level security can ward off an attack in the upper levels of the protocol stack conducted at or near the destination machine (once the datagram has passed the gateway and is being transported to the destination machine, for example). Used alone in external transit, however, the true addresses of the packets can be spoofed. Also, application-level security tends to execute slower than optimized network layer security does.

Network layer versus application layer II

It seems that the use of complementary technologies may be the comprehensive answer to the network/application layer question.

An application-level security scheme like SSL can serve as the upper layer's security, while a protocol like AH could be used in the network layer to prevent spoofing. Each individual case will require an individual decision as to whether to use application-level security.

The developer must be always aware that there are no "magic bullets" and one size won't fit all. While there are general principles of security, the exact mix used should fit the situation at hand.

Section 4. IPSec basics

IPSec structures

In this section, we will examine in detail the IPSec structures that have been previously introduced.

Security Association (SA) Simply put, an SA is a one-way, logical connection between two IPSec systems. It consists of the following elements:

- Security Parameter Index
- IP Destination Address
- Security Protocol

Security Parameter Index (SPI): This is a 32-bit value that identifies different SAs with the same destination address and security protocols. It's carried in the security protocol header, and is usually selected by the destination system.

IP Destination Address (IPDA): This is unicast address. SAs are simplex, thus unidirectional.

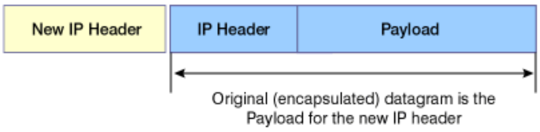
Security Protocol (SP): This is either AH or ESP.

For bidirectional data flow, two SAs (one for each direction) must be defined. Because an SA can only handle one protocol, use of both will require two SAs for each direction, grouped into an SA bundle.

IPSec Tunneling I

Figure 3 shows how tunneling wraps a packet into a new one. The entire packet is treated as a payload, and a new header attached. In IPSec, IP is tunneled through IP. This protects the header of the encapsulated packet. By encrypting a packet, its destination address can be concealed. This can be used, for example, to protect internal network addresses without requiring NAT services to be present.

Figure 3


New IP Header
IP Header
Payload
Original (encapsulated) datagram is the Payload for the new IP header

IPSec Tunneling II

How exactly does this address protection work? Tunneling requires some intermediate processing, and this is usually done at the Internet gateway. The gateway (most likely an IPSec firewall or router) must have some public IP address or it cannot function. The endpoints of the publicly-exposed tunnel are established at these gateways. If IP/IP tunneling is in effect, the gateway is the destination specified in the "outer" address. The gateway obtains the encrypted packet, decodes it for the "inside" address, and then sends it. This, of course, assumes that a secured internal network is in place and that it is not vulnerable to eavesdropping or MITM attacks.

More details about this protocol this can be found in RFC 2003 "IP encapsulation within IP" (see the Resources section).

Mutability in the IP header

Before we consider what the AH protocol can do, first consider what parts of the IP header are going to change en route. These mutable fields will not be protected by AH simply because they can change in normal routing.

The mutable IPv4 fields are Type of Service (TOS), Flags, Fragment Offset, Time to Live (TTL), and the header checksum. If protection is needed for these fields, one can always IP/IP tunnel.

Section 5. IPSec: AH protocol structure

Overview

AH only works on non-fragmented packets. If the offset field is not zero, or the More Fragments bit is set, the packet will be discarded and never reach the upper levels. This prevents an attack that tries to force bogus packets through a firewall by masquerading as fragments, and the discarding of the packet helps prevent a denial of service attack.

As the IPSec RFC 2401 says:

"AH also offers an anti-replay (partial sequence integrity) service at the discretion of the receiver, to help counter denial of service attacks. AH is an appropriate protocol to employ when confidentiality is not required. AH also provides authentication for selected portions of the IP header, which may be necessary in some contexts. For example, if the integrity of an IPv4 option or IPv6 extension header must be protected en route between sender and receiver, AH can provide this service (except for the non-predictable but mutable parts of the IP header)."

Header structure

A diagram of the overall header structure in IPv6 is:

```
+-----------+-----------------+----------+------+-------------+
| IPv6 Header| Hop-by-Hop/Routing| Auth Header| Others| Upper
Protocol|
+-----------+-----------------+----------+------+-------------+
```

It appears after the IPv6 Hop-by-Hop header and before the destination options.

In IPv4, the AH header follows the main IPv4 header. The diagram is:

```
+-------------+-------------+-----------------------------+
| IPv4 Header |   Auth Header| Upper Protocol (e.g. TCP, UDP)|
+-------------+-------------+-----------------------------+
```

The AH itself

The AH itself looks like this:

```
+---------------+---------------+---------------+---------------+
| Next Header   |     Length     |            RESERVED           |
+---------------+---------------+---------------+---------------+
|                    Security Parameters Index                   |
+---------------+---------------+---------------+---------------+
|    Authentication Data (variable number of 32-bit words)      |
+---------------+---------------+---------------+---------------+
|1 2 3 4 5 6 7 8 1 2 3 4 5 6 7 8 1 2 3 4 5 6 7 8 1 2 3 4 5 6 7 8|
```

Fields inside the header I

Let's look at the fields inside the header.

Next header: 8 bits wide. Identifies the next payload after the Authentication Payload.

Payload length: 8 bits wide. The length of the authentication data field in 32-bit words. Minimum value is 0 words, which is only used in the case of a "null" authentication algorithm. This should not happen in IPSec, which must have one specified.

Reserved: 16 bits wide. Reserved for future use. It must be set to all zeros when sent. The value is included in the authentication data calculation, but is otherwise ignored by the recipient.

Fields inside the header II

Security parameters index (SPI): This is a 32-bit pseudo-random value identifying the security association for this datagram. The SPI value 0 is reserved to indicate that "no security association exists."

The set of SPI values in the range 1 through 255 are reserved to the Internet Assigned Numbers Authority (IANA) for future use. A reserved SPI value will not normally be assigned by IANA unless the use of that particular assigned SPI value is openly specified in an RFC.

Fields inside the header III

Authentication Data (AD): The length of this field is variable, but is always an integral number of 32-bit words. Some implementations require padding to other alignments, such as 64-bits, in order to improve performance. All implementations must support such padding, which is specified by the Destination on a per-SPI basis. The value of the padding field is arbitrarily selected by the sender and is included in the AD calculation.

An implementation will normally use the combination of Destination Address and SPI to locate the Security Association that specifies the field's size and use. The field retains the same format for all datagrams of any given SPI and destination address pair. The AD fills the field beginning immediately after the SPI field. If the field is longer than necessary to store the actual AD, then the unused bit positions are filled with unspecified, implementation-dependent values.

Calculating the AD

The AD is calculated with the algorithm selected at the SA initialization. The AD length is an integral multiple of 32 bits.

In theory any MAC algorithm can be used to calculate the AD. The specification requires that HMAC-MD5-96 and HMAC-SHA-1-96 must be supported. In practice, Keyed SHA-1 is also used. Implementations usually support two to four algorithms.

When doing the AD calculation, the mutable fields are considered to be filled with zero. By replacing the field's value with zero rather than omitting these fields, alignment is preserved for the authentication calculation.

The selection of the appropriate SA for an outgoing IP packet is based at least upon the sending userid and the destination address. When host-oriented keying is in use, all sending userids will share the same SA to a given destination.

IPv4 considerations in AD calculations

The IPv4 "Time to live" and "Header checksum" fields are the only fields in the IPv4 base header that are handled specially for the AD calculation. Reassembly of fragmented packets occurs **prior** to processing by the local IP AH implementation. The "more" bit is, of course, cleared upon reassembly.

Hence, no other fields in the IPv4 header will vary in transit from the perspective of the AH implementation. The "Time to live" and "Header checksum" fields of the IPv4 base header have to be set to all zeros for the AD calculation. All other IPv4 base header fields are processed normally with their actual contents. Because IPv4 packets are subject to intermediate fragmentation in transit, it is important that the reassembly of IPv4 packets be performed prior to the AH processing.

If a receiving system does not recognize an IPv4 option that is present in the packet, that option is included in the AD calculation. This means that any IPv4 packet containing an IPv4 option that is unrecognized by the receiver will fail the authentication check and consequently be dropped by the receiver.

IPv6 considerations in AD calculations

The IPv6 "Hop Limit" field is the only field in the IPv6 base header that is handled specially for AD calculation. The value of the Hop Limit field is set to zero for the purpose of calculation. All other fields in the base IPv6 header are included in the AD calculation using the normal procedures for this process.

All IPv6 "Option Type" values contain a bit that is used to determine whether that option data will be included in the AD calculation. This bit is the third-highest-order bit of the IPv6 Option Type field. If this bit is set to zero, then the corresponding option that's included in the AD option is replaced by all zero bits of the same length as the option, for the purpose of the AD calculation.

The IPv6 routing header "Type 0" will rearrange the address fields within the packet during transit from source to destination. Any potential problems with this are avoided because the contents of the packet as it will appear at the receiver are already known to the sender and to all intermediate hops. So, the IPv6 Routing Header "Type 0" is included in the AD calculation using the normal procedure.

Section 6. Wrapup

Next time

In Part 2 of this tutorial, we will examine in more detail how to use the technologies of VPNs, and finish up with some VPN implementations. The discussion will build on what has been covered here in Part 1, so keep your notes handy. You *are* taking copious notes about all this, right?

Resources

Karen Monkhouse's overview of VPN technology, *"Business Taps the Internet with Virtual Private Networks"* , includes several case studies.Visit MIT's *ISAKMP Distribution Page* .RFC 2003, *"IP encapsulation within IP"* , shows you how an IP datagram can be encapsulated within another IP datagram.Check out RFC 2401, *"Security Architecture for the Internet Protocol"* by Stephen Kent and Randall Atkinson. *Winn Schwartau's Network World column* tells you what you need to know about VPNs.RSA Security's Web site features the white paper *"Implementing A Secure Virtual Private Network"* . Get the latest *IBM security news* .Find out how *IBM's Managed Security Services* can help you to identify and solve your real-time security risks by using a proven continuous management process.

Your feedback

Please let us know whether this tutorial was helpful to you and how we could make it better. We'd also like to hear about other tutorial topics you'd like to see covered. Thanks!

For questions about the content of this tutorial, please contact the author, Larry Loeb, at *larryloeb@prodigy.net* .

Colophon

This tutorial was written entirely in XML, using the developerWorks Toot-O-Matic tutorial generator. The Toot-O-Matic tool is a short Java program that uses XSLT stylesheets to convert the XML source into a number of HTML pages, a zip file, JPEG heading graphics, and PDF files. Our ability to generate multiple text and binary formats from a single source file illustrates the power and flexibility of XML.