Using IPSec in Windows 2000 and XP, Part 1

Chris Weber 2001-12-05

Introduction

This article is the first of a three-part series that will describe the technical details of using Internet Protocol Security (IPSec) in a Windows 2000 and XP environment. IPSec is an architecture consisting of several protocol standards aimed at protecting IP traffic. IPSec is commonly used to refer to the secure IP packets of the AH and ESP protocols, because these provide the major security services. This article will offer a brief overview of IPSec, as well as a look at the structure and interface for IPSec in Windows and a look at the two different modes of IPSec authentication methods for IKE in Windows.

The audience for this series of articles is assumed to be system or network administrators, computer security or support professionals, and should have a working technical knowledge of Windows 2000 and TCP/IP. Readers should also be familiar with the concepts of IPSec, and may have used it before in a VPN setting. This series will not cover all of the details of IPSec, since we have the RFCs for that. The goal of this series is to provide readers with a bit of background on IPSec, and an understanding of how IPSec has been implemented in the Windows environment. Since readers may be thinking of ways to take advantage of IPSec, we will definitely cover the implementation subtleties they should be aware of.

An Brief Overview of IPSec

Because security is not inherently built into the protocols of TCP/IP, securing IP traffic has largely been a chore for the higher layers (i.e. application layer). IPSec is designed to secure traffic at the IP layer, transparently to the other layers and applications. As stated in RFC 2401, the security services that IPSec provides can include access control, authentication (through data origin authentication and connectionless integrity), packet anti-replay protection, confidentiality through encryption, and limited traffic flow confidentiality. Because these services operate at the IP layer, any higher-layer protocols such as TCP and UDP can use them.

Although IPSec works for IPv4 and IPv6, the focus of this series is on IPv4. To reiterate, this means that the security services IPSec provides to unicast IP traffic include:

- Authentication of hosts before and during communications
- Confidentiality through encryption of IP traffic
- Integrity of IP traffic by identifying modified or spoofed traffic
- Prevention of replay attacks

That should sufficiently explain why IPSec is used; now, where and when to use it is up to the user.

Not so fast though, before IPSec is implemented in a Microsoft environment, readers should realize that there are certain types of traffic that cannot be protected. For one, other protocols in the network layer, such as IPX/SPX and Appletalk, are off limits to IPSec - it cannot do anything for them. More important, perhaps, are the other types of IP-based traffic that IPSec cannot protect. In accordance with RFC, and in order to provide authenticity and integrity, IPSec is designed to protect unicast IP traffic – traffic between one source IP address and one destination IP address. That means it cannot protect multicast or broadcast traffic. Also, Kerberos, IKE, and RSVP traffic are not protected because they are necessary to the operation of IPSec in a Windows environment. These are known as the exemptions, and we will devote a section to them later in this series of articles. Bear in mind that Microsoft has provided a way to remove these exemptions or make some of these protected by IPSec through registry modification, and is working to make the situation even more configurable by the user in future releases.

The Structure and Interface for IPSec in Windows

The design and integration of IPSec services and support in Windows 2000 was jointly developed by Microsoft and Cisco Systems, Inc. The agreement was made to integrate Cisco's ISAKMP/IKE with the IPSec kernel driver of Microsoft, and also involved developing IPSec policy for use with Active Directory.

IPSec implementation in Windows 2000 and XP has been largely RFC-compliant. With that in mind, it is solid, scalable and robust, as the creators of IPSec had intended. The following sections will briefly cover the structure of the IP Security Protocol; for more detailed understanding, readers are encouraged to review the RFCs that are referred to in the discussion.

Components of IPSec

IPSec in Windows consists of three main components - the **Policy Agent**, the **Internet Key Exchange (IKE) module**, and the **IPSec driver**. These three components, in conjunction with other Windows components such as the TCP/IP driver and cryptoAPI, provide for seamless IPSec functionality in Windows 2000 and XP. Let's look at each.

The **Policy Agent** has two main functions - to acquire and distribute the IPSec policies that the administrator has defined. It first acquires the IPSec policy from the appropriate policy store, which will either be the Active Directory, a set of local configuration policies, or a local cache of policies. The appropriate policy components are then distributed to either the IKE module (where the authentication and security settings go) or IPSec Driver (where the IP filters go). The active part of the Policy Agent runs as a service in Windows, appropriately named "IPSec Policy Agent" in Windows 2000, or "IPSec Services" in Windows XP.

The **Internet Key Exchange (IKE) module** probably has the most complex job of the IPSec architecture. Its function is to negotiate Security Associations (SA) for both the ISAKMP (Phase I) and IPSec (Phase II). It does this based on the authentication and security settings it receives from the Policy Agent. The IKE component is actually started (stopped and restarted too) by the Policy Agent service.

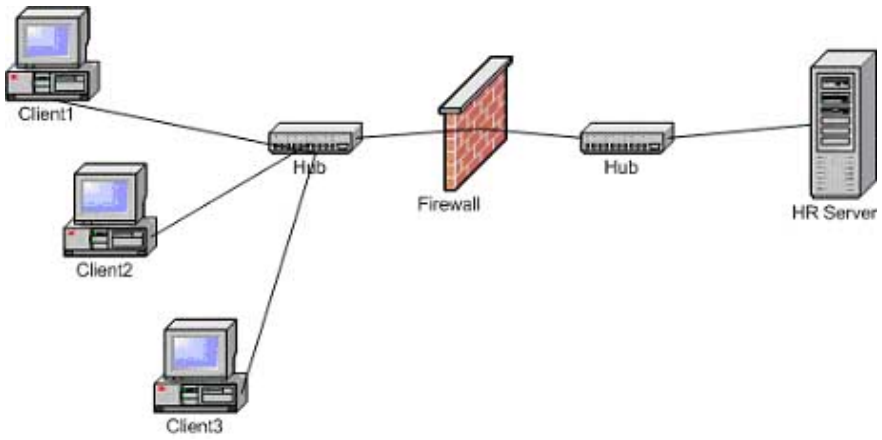
For SA creation between two computers, the Internet Engineering Task Force (IETF) established a standard method of SA creation and key exchange. This is known as IKE (RFC 2409) and is actually a combination of the Internet Security Association and Key Management Protocol (ISAKMP RFC 2408) and the Oakley Key Determination Protocol (RFC 2412).

Finally, the **IPSec driver** is responsible for exercising the filters, and maintaining the stateful status of connections. The IPSec driver uses the defined filters to determine which packets get blocked, permitted, or secured.

Back up just a bit - So what's this talk of policies and rules? Well it's the language of IPSec in Windows. When IPSec is implemented on a Windows computer, the user must first create a **policy**, which is the generic name for the big picture of how IPSec will work for this computer. The policy contains **rules**, and each rule contains the **authentication** method and IP **filters** to exercise. So then, a single policy contains many rules, which contain many IP filters. Only a single policy can be activated on a computer at any one time.

The next few sections describe the components of the IPSec protocols in more detail. Some of this is more focused on the protocol specifics rather than how Windows implements and manages it.

For these sections we will keep in mind a simple network of a few computers, to help illustrate the details. This network consists of three Windows 2000 client desktops and one Windows 2000 SAP server. All systems are part of the Human Resources department. Refer to the following diagram for now. Keep in mind that in a later part of this series we will flesh out all of the details for actually deploying IPSec in such a scenario.



Transports vs. Tunnels

IPSec operates in either one of two modes - transport mode or tunnel mode. Transport mode is meant primarily for protection of upper layer protocols, while tunnel mode protects the IP layer as well. In tunnel mode, used primarily between two gateways or a server and a gateway, the packet has two IP headers, an outer and an inner. The outer header identifies the source and destination endpoints, while the inner contains the original sender and destination addresses, protected by IPSec.

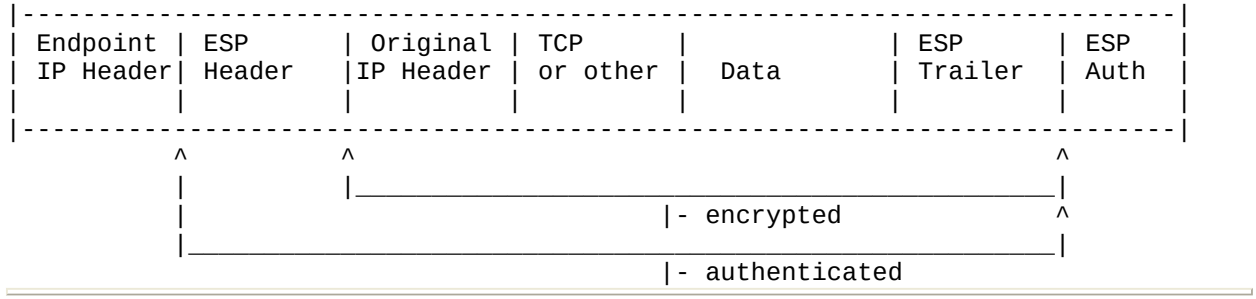
Tunnel Mode

IPSec in Windows was meant mainly for interaction with routers or other IPSec tunnel endpoints. However, as stated, it can be used with L2TP to provide a VPN remote access solution. When this is done, the L2TP headers are encapsulated and protected by IPSec, so if encryption is being used with IPSec, the L2TP headers will be encrypted. The only unencrypted headers will be the outer IP headers (with the destination endpoint IP address) and lower layers.

For details of setting up IPSec tunnels in Windows 2000 (most of which applies to XP also) please take a look at the Microsoft Support Services document How to Configure IPSec Tunneling in Windows 2000.

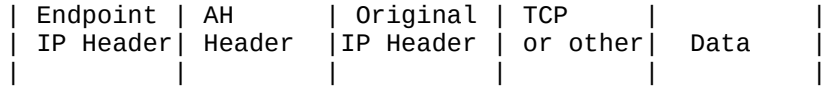
The following diagram illustrates tunnel encapsulation using Encapsulating Security Payload, where the packet is signed starting at the ESP Header and ending at the ESP trailer. The following sections talk more about the ESP protocol. For now, it is important to know only that it is the protocol that does encryption of IPSec packets.

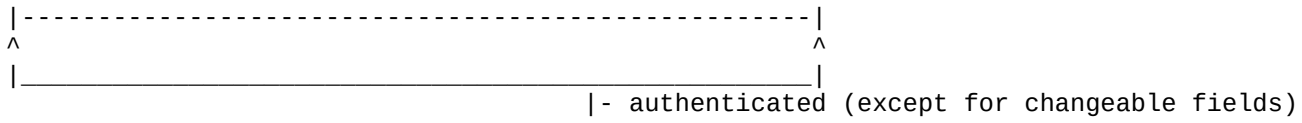
Figure 1: Tunnel Mode ESP packet



In tunnel encapsulation using an Authentication Header, the entire packet is signed, but none of it is encrypted. We talk more of AH later as well, for now just remember it does not do encryption, but rather signs the entire packet for authenticity.

Figure 2: Tunnel Mode AH packet





Transport Mode

Transport mode is the more commonly used of the two modes. In transport mode, the original packet IP headers are not shifted inward and the endpoint IP headers are placed outermost. Instead, the packet has just one set of IP headers, and either the encryption or authentication offered by ESP and/or AH. This is more typical of a situation where users might want to protect two communicating hosts. For example, if the user has an IIS server and a SQL server using IPSec to communicate directly, there is no need for tunnel mode because each computer is the endpoint. Forcing them to use tunnel mode would only be creating unnecessary work.

Figure 3: Transport Mode ESP packet

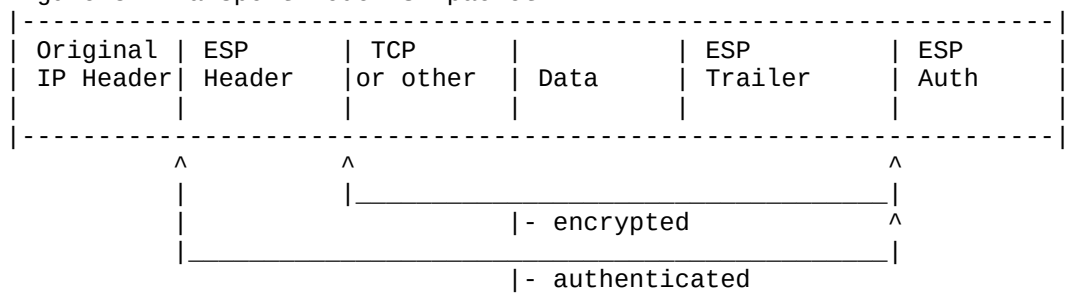
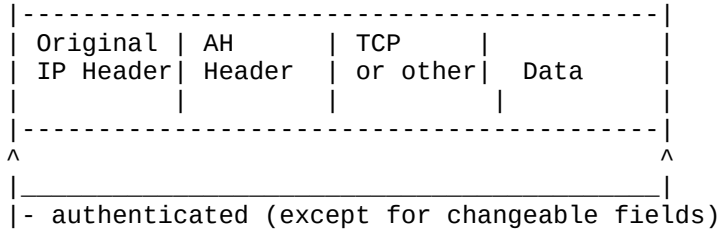


Figure 4:Transport Mode AH packet



Authentication Methods for IKE in Windows

The IPSec architecture is built with authentication of IP packets in mind. IKE is responsible for negotiating Security Associations using a Diffie-Hellman key exchange and Kerberos, certificates, or preshared key authentication. Authentication Methods are set up per rule in the Windows IPSec Security Policies snap-in (policy properties, rule, Authentication Methods tab):

Windows 2000 and XP provide three methods of authentication to use:

- Kerberos
- Certificates
- Preshared Keys

Kerberos is the default authentication method, because it is the inherent system in a Windows 2000 and XP domain. This method is simple to use and secure. The Kerberos system in IPSec uses keys for the machine accounts, rather than the user accounts.

Certificates are one of the strongest forms of authentication. Computers can be configured to specify which Certificate Authority they will use, but the user must select the root CA in the rule configuration. Additionally, the remote computer's root CA must exist in the computer's list of CAs, or authentication will not succeed. All certificate validation is actually performed via the Cryptographic API (CAPI) in Windows, and IKE is merely used for the certificate negotiation parameters.

Preshared Keys are quite simple to set up, but have serious security risks. These risks are discussed more in the "Attacks on IPSec" section in the upcoming third installment in this series, but in general, users should know that preshared keys are stored in clear text in the policy store database (the database that stores IPSec policies.) When setting up preshared keys, the user simply types in the same, shared, string of text on both computers, keeping in mind that this is case sensitive.

Next Time...

This concludes the first part of this three-part examination of IPSec for Windows 2000 and XP. The next installment will look at Security Associations - how connections are managed with Security Associations, Main Mode Authentication and IKE, Quick Mode Authentication Headers and ESP. We will also discuss some of the tools available in 2000 and XP.

Chris Weber has spent many years working various positions in the IT industry, including systems administrator and network architect. Today Chris is a security engineer for Foundstone, where he enjoys pen-testing, network architecture reviews, and application security testing.