CHAPTER 1

VPNs and VPN Technologies

This chapter defines virtual private networks (VPNs) and explores fundamental Internet Protocol Security (IPSec) technologies. This chapter covers the following topics:

- Overview of VPNs and VPN technologies
- Internet Protocol Security (IPSec)
- IPSec crypto components
- IKE overview
- How IPSec works
- IPSec security associations
- CA support overview

Overview of VPNs and VPN Technologies

Cisco products support the latest in VPN technology. A VPN is a service that offers secure, reliable connectivity over a shared public network infrastructure such as the Internet.

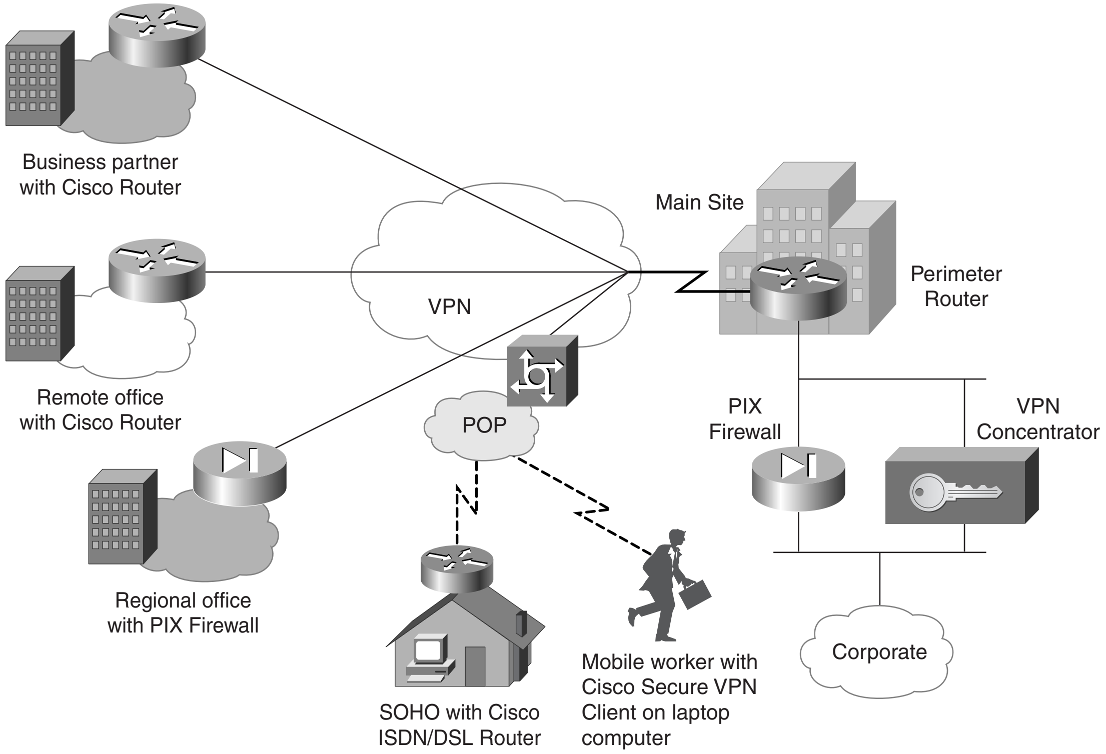
Figure 1-1 shows various VPNs between a main site and branch offices and small office, home office (SOHO) workers.

VPNs maintain the same security and management policies as a private network. They are the most cost effective method of establishing a virtual point-to-point connection between remote users and an enterprise customer's network. There are three main types of VPNs.

- **Access VPNs**—Provide remote access to an enterprise customer's intranet or extranet over a shared infrastructure. Access VPNs use analog, dial, ISDN, digital subscriber line (DSL), mobile IP, and cable technologies to securely connect mobile users, telecommuters, and branch offices.
- **Intranet VPNs**—Link enterprise customer headquarters, remote offices, and branch offices to an internal network over a shared infrastructure using dedicated connections. Intranet VPNs differ from extranet VPNs in that they allow access only to the enterprise customer's employees.

- **Extranet VPNs**—Link outside customers, suppliers, partners, or communities of interest to an enterprise customer's network over a shared infrastructure using dedicated connections. Extranet VPNs differ from intranet VPNs in that they allow access to users outside the enterprise.

Figure 1-1 *Examples of VPNs*


Business partner with Cisco Router
Remote office with Cisco Router
Regional office with PIX Firewall
VPN
POP
SOHO with Cisco ISDN/DSL Router
Mobile worker with Cisco Secure VPN Client on laptop computer
Main Site
Perimeter Router
PIX Firewall
VPN Concentrator
Corporate

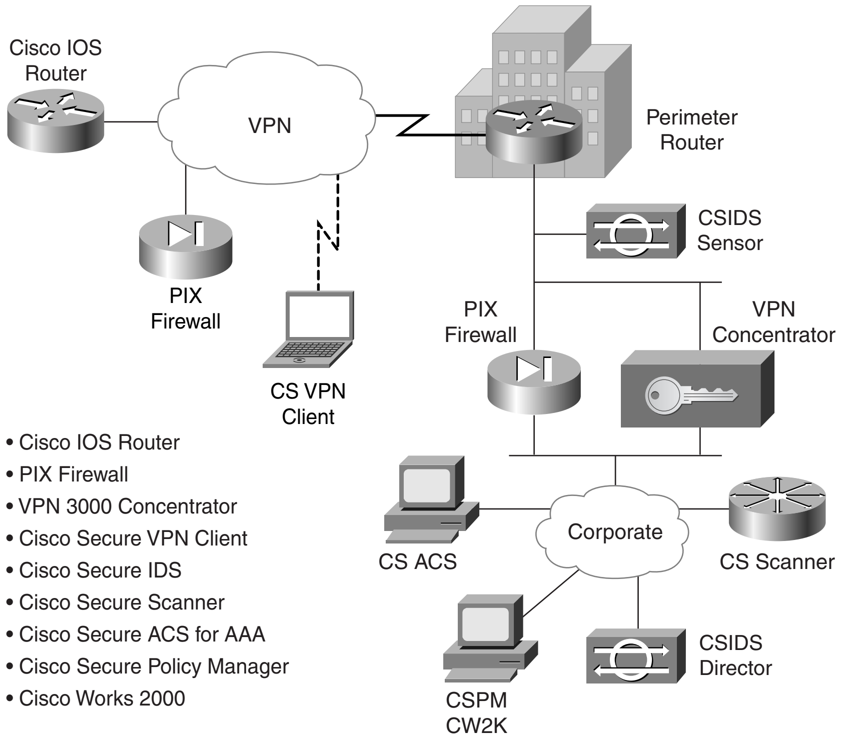
The following main components make up Cisco's VPN offerings:

- **Cisco VPN routers**—Use Cisco IOS software IPSec support to enable a secure VPN. VPN-optimized routers leverage existing Cisco investment, perfect for the hybrid WAN.
- **Cisco Secure PIX Firewall**—Offers a VPN gateway alternative when the security group "owns" the VPN.
- **Cisco VPN Concentrator series**—Offers powerful remote access and site-to-site VPN capability, easy-to-use management interface, and a VPN client.
- **Cisco Secure VPN Client**—Enables secure remote access to Cisco router and PIX Firewalls and runs on the Windows operating system.

- **Cisco Secure Intrusion Detection System (CSIDS) and Cisco Secure Scanner**—Can be used to monitor and audit the security of the VPN.
- **Cisco Secure Policy Manager and Cisco Works 2000**—Provide VPN-wide system management.

These components can all be seen in Figure 1-2.

Figure 1-2 *Cisco Secure VPN Components*



The main Cisco VPN product offerings are discussed in more detail in Chapter 2, "Cisco VPN Family of Products."

Internet Protocol Security (IPSec)

Cisco IOS uses the industry-standard IPSec protocol suite to enable advanced VPN features. The PIX IPSec implementation is based on the Cisco IOS IPSec that runs in Cisco routers.

IPSec acts at the network layer, protecting and authenticating IP packets between a PIX Firewall and other participating IPSec devices (peers), such as other PIX Firewalls, Cisco routers, the Cisco Secure VPN Client, the VPN 3000 Concentrator series, and other IPSec-compliant products.

IPSec enables the following Cisco IOS VPN features:

- **Data confidentiality**—The IPSec sender can encrypt packets before transmitting them across a network.
- **Data integrity**—The IPSec receiver can authenticate packets sent by the IPSec sender to ensure that the data has not been altered during transmission.
- **Data origin authentication**—The IPSec receiver can authenticate the source of the IPSec packets sent. This service is dependent upon the data integrity service.
- **Antireplay**—The IPSec receiver can detect and reject replayed packets.

IPSec Overview

IPSec is a framework of open standards that provides data confidentiality, data integrity, and data authentication between participating peers at the IP layer. IPSec can be used to protect one or more data flows between IPSec peers. IPSec is documented in a series of Internet RFCs, all available at http://www.ietf.org/html.charters/ipsec-charter.html. The overall IPSec implementation is guided by "Security Architecture for the Internet Protocol," RFC 2401. IPSec consists of the following two main protocols:

- Authentication Header (AH)
- Encapsulating Security Payload (ESP)

IPSec also uses other existing encryption standards to make up a protocol suite, which are explained in the next sections.

IPSec has several standards that are supported by Cisco IOS and the PIX Firewall.

- IP Security Protocol
  - — Authentication Header (AH)
  - — Encapsulating Security Payload (ESP)
- Data Encryption Standard (DES)
- Triple DES (3DES)
- Diffie-Hellman (D-H)
- Message Digest 5 (MD5)
- Secure Hash Algorithm-1 (SHA-1)

- Rivest, Shamir, and Adelman (RSA) Signatures
- Internet Key Exchange (IKE)
- Certificate Authorities (CAs)

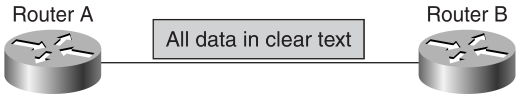
IP Security Protocol—Authentication Header (AH)

Authentication Header (AH) provides authentication and integrity to the datagrams passed between two systems.

It achieves this by applying a keyed one-way hash function to the datagram to create a message digest. If any part of the datagram is changed during transit, it will be detected by the receiver when it performs the same one-way hash function on the datagram and compares the value of the message digest that the sender has supplied. The one-way hash also involves the use of a secret shared between the two systems, which means that authenticity can be guaranteed.

AH can also enforce antireplay protection by requiring that a receiving host sets the replay bit in the header to indicate that the packet has been seen. Without this protection, an attacker might be able to resend the same packet many times: for example, to send a packet that contains "withdraw $100 from account X." Figure 1-3 shows two routers and confirms that the data between them is sent in clear text.

Figure 1-3 *Authentication Header*



- Ensures data integrity
- Provides origin authentication—ensures packets definitely came from peer router
- Uses keyed-hash mechanism
- Does NOT provide confidentiality (no encryption)
- Provides optional replay protection

The AH function is applied to the entire datagram except for any mutable IP header fields that change in transit: for example, Time to Live (TTL) fields that are modified by the routers along the transmission path. AH works as follows:

Step 1 The IP header and data payload is hashed.

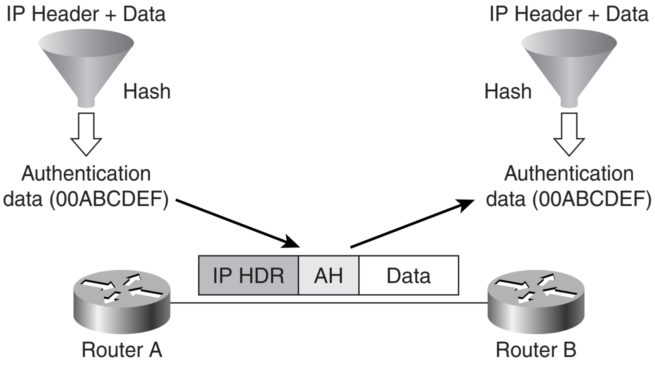
Step 2 The hash is used to build a new AH header, which is appended to the original packet.

Step 3 The new packet is transmitted to the IPSec peer router.

Step 4 The peer router hashes the IP header and data payload, extracts the transmitted hash from the AH header, and compares the two hashes. The hashes must match exactly. Even if one bit is changed in the transmitted packet, the hash output on the received packet will change and the AH header will not match.

This process can be seen in Figure 1-4.

Figure 1-4 *AH Authentication and Integrity*

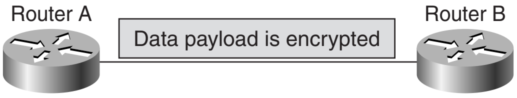


IP Security Protocol—Encapsulating Security Payload (ESP)

Encapsulating Security Payload (ESP) is a security protocol used to provide confidentiality (encryption), data origin authentication, integrity, optional antireplay service, and limited traffic flow confidentiality by defeating traffic flow analysis. Figure 1-5 shows that the data payload is encrypted with ESP.

ESP provides confidentiality by performing encryption at the IP packet layer. It supports a variety of symmetric encryption algorithms. The default algorithm for IPSec is 56-bit DES. This cipher must be implemented to guarantee interoperability among IPSec products. Cisco products also support use of 3DES for strong encryption. Confidentiality can be selected independent of all other services.

Figure 1-5 *Encapsulating Security Payload*



- Data confidentiality (encryption)
- Limited traffic flow confidentiality
- Data integrity
- Optional data origin authentication
- Anti-replay protection
- Does not protect IP header

NOTE Deciding whether to use AH or ESP in a given situation might seem complex, but it can be simplified to a few rules, as follows. When you want to make sure that data from an authenticated source gets transferred with integrity and does not need confidentiality, use the AH protocol. If you need to keep data private (confidentiality), then you must use ESP. ESP will encrypt the upper-layer protocols in transport mode and the entire original IP datagram in tunnel mode so that neither are readable from the wire. However, ESP can now also provide authentication for the packets. This situation is covered later in this chapter in the “ESP Tunnel Versus Transport Mode” section.

DES Algorithm

DES uses a 56-bit key, ensuring high-performance encryption. DES is used to encrypt and decrypt packet data. DES turns clear text into ciphertext with an encryption algorithm. The decryption algorithm on the remote end restores clear text from ciphertext. Shared secret keys enable the encryption and decryption.

Triple DES Algorithm (3DES)

Triple DES (3DES) is also a supported encryption protocol for use in IPSec on Cisco products. The 3DES algorithm is a variant of the 56-bit DES. 3DES operates similarly to DES in that data is broken into 64-bit blocks. 3DES then processes each block three times, each time with an independent 56-bit key. 3DES effectively doubles encryption strength over 56-bit DES.

Diffie-Hellman (D-H)

Diffie-Hellman (D-H) is a public-key cryptography protocol. It allows two parties to establish a shared secret key used by encryption algorithms (DES or MD5, for example) over an insecure communications channel. D-H is used within IKE to establish session keys. 768-bit and 1024-bit D-H groups are supported in the Cisco routers and PIX Firewall. The 1024-bit group is more secure because of the larger key size.

Message Digest 5 (MD5)

Message Digest 5 (MD5) is a hash algorithm used to authenticate packet data. Cisco routers and the PIX Firewall use the MD5 hashed message authentication code (HMAC) variant that provides an additional level of hashing. A hash is a one-way encryption algorithm that takes an input message of arbitrary length and produces a fixed length output message. IKE, AH, and ESP use MD5 for authentication.

Secure Hash Algorithm-1 (SHA-1)

Secure Hash Algorithm-1 (SHA-1) is a hash algorithm used to authenticate packet data. Cisco routers and the PIX Firewall use the SHA-1 HMAC variant, which provides an additional level of hashing. IKE, AH, and ESP use SHA-1 for authentication.

Rivest, Shamir, and Adelman (RSA) Signatures

Rivest, Shamir, and Adelman (RSA) is a public-key cryptographic system used for authentication. IKE on the Cisco router or PIX Firewall uses a D-H exchange to determine secret keys on each IPSec peer used by encryption algorithms. The D-H exchange can be authenticated with RSA signatures or preshared keys.

Internet Key Exchange (IKE)

Internet Key Exchange (IKE) is a hybrid protocol that provides utility services for IPSec: authentication of the IPSec peers, negotiation of IKE and IPSec security associations, and establishment of keys for encryption algorithms used by IPSec.

NOTE IKE is synonymous with Internet Security Association Key Management Protocol (ISAKMP) in Cisco router or PIX Firewall configurations.

Certificate Authorities (CAs)

The Certificate Authority (CA) offered by Cisco routers and the PIX Firewall allows the IPSec-protected network to scale by providing the equivalent of a digital identification card to each device. When two IPSec peers wish to communicate, they exchange digital certificates to prove their identities (thus removing the need to exchange public keys manually with each peer or to specify a shared key manually at each peer). The digital certificates are obtained from a CA. CA support on Cisco products uses RSA signatures to authenticate the CA exchange.

Tunnel and Transport Modes

IPSec can be run in either *tunnel* or *transport* modes. Each of these modes has its own particular uses, and care should be taken to ensure that the correct one is selected for the solution. Figure 1-6 shows that transport mode should be used for end-to-end sessions and tunnel mode should be used for everything else.

Figure 1-6 *Tunnel and Transport Mode IPSec*

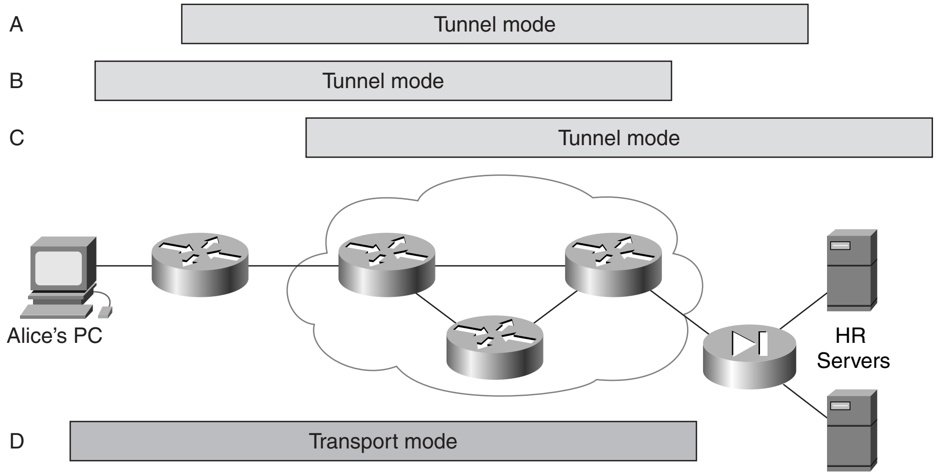


Figure 1-6 illustrates situations where a tunnel or a transport mode is used. Tunnel mode is most commonly used between gateways or from an end station to a gateway. The gateway acts as a proxy for the hosts behind it. Transport mode is used between end stations or between an end station and a gateway, if the gateway is being treated as a host; for example, in an encrypted Telnet session from a workstation to a router, the router is the actual destination.

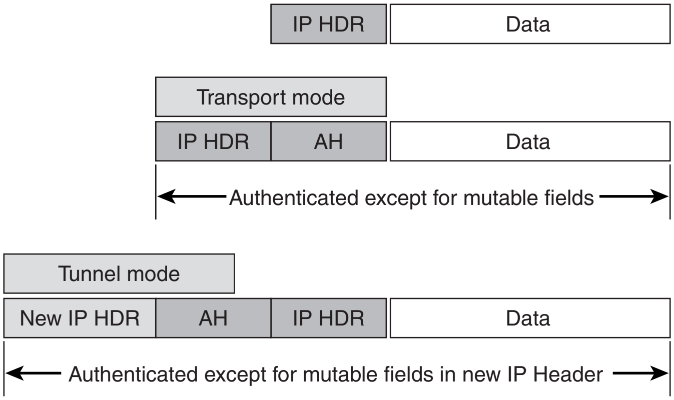
Using Figure 1-6, consider some examples of when to use tunnel or transport mode.

- **Example A**—Tunnel mode is most commonly used to encrypt traffic between secure IPSec gateways, such as between the Cisco router and the PIX Firewall, as shown in Example A in Figure 1-6. The IPSec gateways proxy IPSec for the devices behind them, such as Alice's PC and the HR servers in the figure. In Example A, Alice connects to the HR servers securely through the IPSec tunnel set up between the gateways.
- **Example B**—Tunnel mode is also used to connect an end station running IPSec software, such as the Cisco Secure VPN Client, to an IPSec gateway, as shown in Example B.
- **Example C**—In Example C, tunnel mode is used to set up an IPSec tunnel between the Cisco router and a server running IPSec software. Note that Cisco IOS software and the PIX Firewall set tunnel mode as the default IPSec mode.
- **Example D**—Transport mode is used between end stations supporting IPSec or between an end station and a gateway if the gateway is being treated as a host. In Example D, transport mode is used to set up an encrypted Telnet session from Alice's PC running Cisco Secure VPN Client software to terminate at the PIX Firewall, enabling Alice to remotely configure the PIX Firewall securely.

AH Tunnel Versus Transport Mode

Figure 1-7 shows the differences that the IPSec mode makes to AH. In transport mode, AH services protect the external IP header along with the data payload. AH services protect all the fields in the header that do not change in transport. The AH goes after the IP header and before the ESP header, if present, and other higher-layer protocols.

Figure 1-7 *AH Tunnel Versus Transport Mode*


IP HDR
Data
Transport mode
IP HDR
AH
Data
Authenticated except for mutable fields
Tunnel mode
New IP HDR
AH
IP HDR
Data
Authenticated except for mutable fields in new IP Header

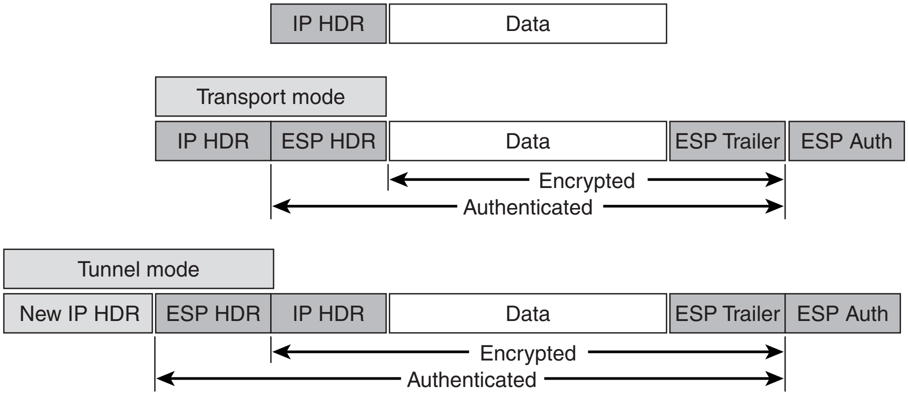
In tunnel mode, the entire original header is authenticated, a new IP header is built, and the new IP header is protected in the same way as the IP header in transport mode.

AH is incompatible with Network Address Translation (NAT) because NAT changes the source IP address, which will break the AH header and cause the packets to be rejected by the IPSec peer.

ESP Tunnel Versus Transport Mode

Figure 1-8 shows the differences that the IPSec mode makes to ESP. In transport mode, the IP payload is encrypted and the original headers are left intact. The ESP header is inserted after the IP header and before the upper-layer protocol header. The upper-layer protocols are encrypted and authenticated along with the ESP header. ESP does not authenticate the IP header itself. Please note that higher-layer information is not available because it is part of the encrypted payload.

Figure 1-8 *ESP Tunnel Versus Transport Mode*



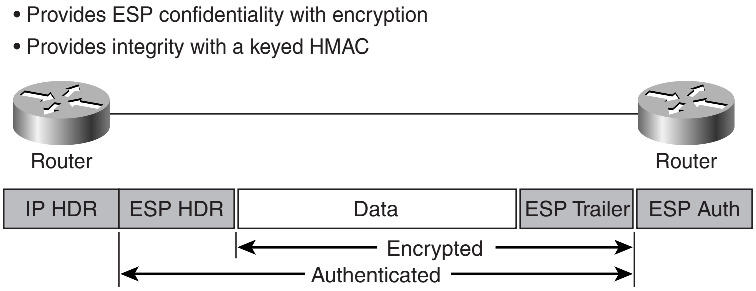
When ESP is used in tunnel mode, the original IP header is well protected because the entire original IP datagram is encrypted. With an ESP authentication mechanism, the original IP datagram and the ESP header are included; however, the new IP header is not included in the authentication.

When both authentication and encryption are selected, encryption is performed first, before authentication. One reason for this order of processing is that it facilitates rapid detection and rejection of replayed or bogus packets by the receiving node. Before decrypting the packet, the receiver can detect the problem and potentially reduce the impact of denial-of-service attacks.

ESP can also provide packet authentication with an optional field for authentication. Cisco IOS software and the PIX Firewall refer to this service as ESP HMAC. Authentication is calculated after the encryption is done. The current IPSec standard specifies SHA-1 and MD5 as the mandatory HMAC algorithms.

The main difference between the authentication provided by ESP and that provided by AH is the extent of the coverage. Specifically, ESP does not protect any IP header fields unless those fields are encapsulated by ESP (tunnel mode). Figure 1-9 illustrates the fields protected by ESP HMAC.

Figure 1-9 *ESP Encryption with a Keyed HMAC*


• Provides ESP confidentiality with encryption
• Provides integrity with a keyed HMAC
Router
Router
IP HDR
ESP HDR
Data
ESP Trailer
ESP Auth
Encrypted
Authenticated

IPSec Transforms

An IPSec *transform* specifies a single IPSec security protocol (either AH or ESP) with its corresponding security algorithms and mode. Some example transforms include the following:

- The AH protocol with the HMAC with MD5 authentication algorithm in tunnel mode is used for authentication.
- The ESP protocol with the 3DES encryption algorithm in transport mode is used for confidentiality of data.

The ESP protocol with the 56-bit DES encryption algorithm and the HMAC with SHA authentication algorithm in tunnel mode is used for authentication and confidentiality.

Transform Sets

A *transform set* is a combination of individual IPSec transforms designed to enact a specific security policy for traffic. During the ISAKMP IPSec security association negotiation that occurs in IKE phase 2 quick mode, the peers agree to use a particular transform set for protecting a particular data flow. Transform sets combine the following IPSec factors:

- **Mechanism for payload authentication**—AH transform
- **Mechanism for payload encryption**—ESP transform
- **IPSec mode (transport versus tunnel)**

Transform sets equal a combination of an AH transform, an ESP transform, and the IPSec mode (either tunnel or transport mode).

IPSec Crypto Components

This section covers in detail the component technologies used in IPSec. This section covers the following:

- DES encryption
- Diffie-Hellman (D-H) key agreement
- HMAC

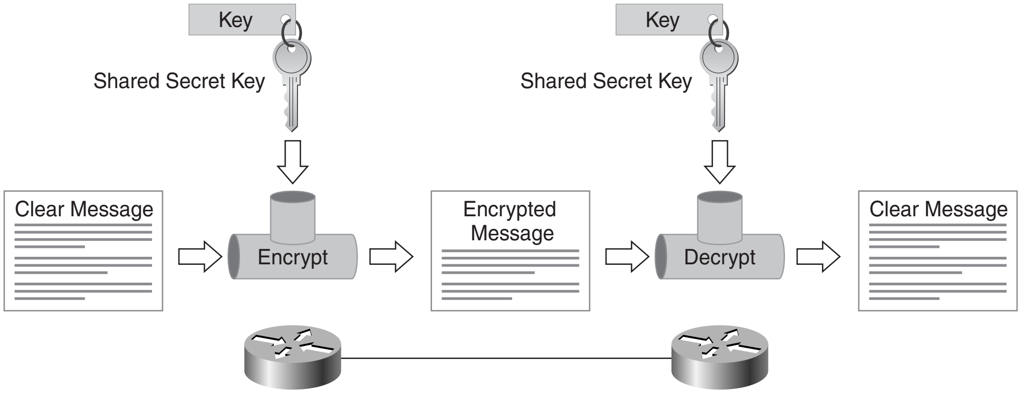
DES Encryption

The components of DES encryption are as follows:

- Encryption and decryption algorithms
- Matching shared secret keys on each peer
- Input clear text data to be encrypted

At the core of DES is the encryption algorithm. A shared secret key is input to the algorithm. Clear text data is fed into the algorithm in fixed-length blocks and is converted to ciphertext. The ciphertext is transmitted to the IPSec peer using ESP. The peer receives the ESP packet, extracts the ciphertext, runs it through the decryption algorithm, and outputs clear text identical to that input on the encrypting peer. The DES encryption algorithm can be seen in action in Figure 1-10. In Figure 1-10, a preshared key is in use.

Figure 1-10 *DES Encryption*



- Encryption turns clear text into ciphertext.
- Decryption restores clear text from ciphertext.
- Keys enable encryption and decryption.

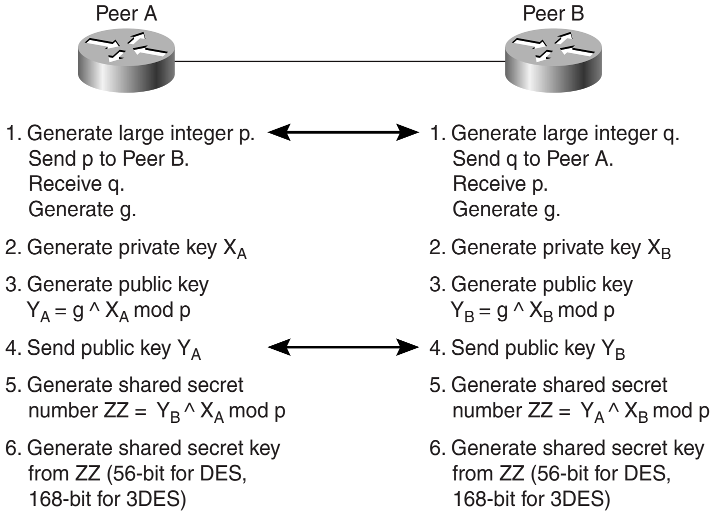
Diffie-Hellman Key Agreement

The Diffie-Hellman (D-H) key agreement is a public key encryption method that provides a way for two IPSec peers to establish a shared secret key that only they know, although they are communicating over an insecure channel.

With D-H, each peer generates a public and private key pair. The private key generated by each peer is kept secret and never shared. The public key is calculated from the private key by each peer and is exchanged over the insecure channel. Each peer combines the other's public key with its own private key and computes the same shared secret number. The shared secret number is then converted into a shared secret key. The shared secret key is never exchanged over the insecure channel.

As you can see in Figure 1-11, Diffie-Hellman key exchange is a complicated process. This adds to the effectiveness of the encryption algorithm.

Figure 1-11 *Diffie-Hellman Key Agreement*


Peer A
Peer B
1. Generate large integer p. Send p to Peer B. Receive q. Generate g.
1. Generate large integer q. Send q to Peer A. Receive p. Generate g.
2. Generate private key X_A
2. Generate private key X_B
3. Generate public key $Y_A = g \wedge X_A \bmod p$
3. Generate public key $Y_B = g \wedge X_B \bmod p$
4. Send public key Y_A
4. Send public key Y_B
5. Generate shared secret number $ZZ = Y_B \wedge X_A \bmod p$
5. Generate shared secret number $ZZ = Y_A \wedge X_B \bmod p$
6. Generate shared secret key from ZZ (56-bit for DES, 168-bit for 3DES)
6. Generate shared secret key from ZZ (56-bit for DES, 168-bit for 3DES)

NOTE Diffie-Hellman is very important because the shared secret key is used to encrypt data using the secret key encryption algorithms specified in the IPSec security associations, such as DES or MD5.

The Diffie-Hellman Process

The Diffie-Hellman process is as follows:

Step 1 The D-H process starts with each peer generating a large prime integer, p and q. Each peer sends the other its prime integer over the insecure channel. For example, Peer A sends p to Peer B. Each peer then uses the p and q values to generate g, a primitive root of p.

Step 2 Each peer generates a private D-H key (Peer A: Xa, Peer B: Xb).

Step 3 Each peer generates a public D-H key. The local private key is combined with the prime number p and the primitive root g in each peer to generate a public key: Ya for Peer A and Yb for Peer B. The formula for Peer A is Ya =g^Xa mod p. The formula for Peer B is Yb =g^Xb mod p. The exponentiation is computationally expensive. The ^ character denotes exponentiation (g^Xa is g to the Xa power); mod denotes modulus.

Step 4 The public keys Ya and Yb are exchanged in public.

Step 5 Each peer generates a shared secret number (ZZ) by combining the public key received from the opposite peer with its own private key. The formula for Peer A is ZZ=(YbXa) mod p. The formula for Peer B is ZZ=(YaXb) mod p. The ZZ values are identical in each peer. Anyone who knows p or g, or the D-H public keys, cannot guess or easily calculate the shared secret value largely because of the difficulty in factoring large prime numbers.

Step 6 Shared secret keys are derived from the shared secret number ZZ for use by DES or HMACs.

NOTE Each IPSec peer has three keys:

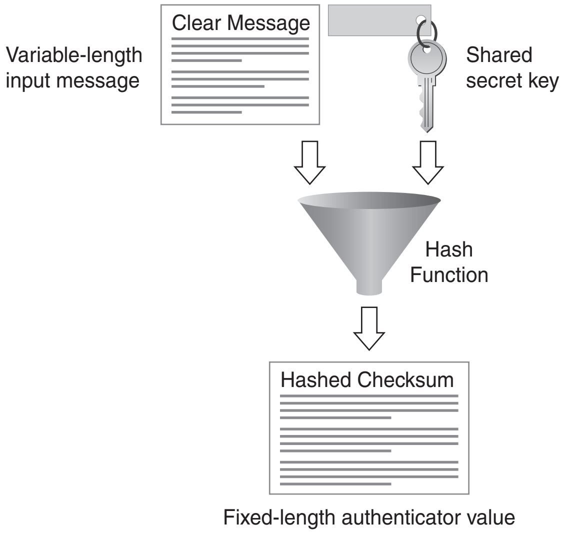
- **A private key that is kept secret and is never shared**—It is used to sign messages.
- **A public key that is shared**—It is used by others to verify a signature.
- **A shared secret key that is used to encrypt data using an encryption algorithm (DES, MD5, and so on)**—The shared secret key is derived from Diffie-Hellman key generation.

HMAC

The fundamental hash algorithms used by IPSec are the cryptographically secure MD5 and SHA-1 hash functions. Hashing algorithms have evolved into HMACs, which combine the proven security of hashing algorithms with additional cryptographic functions. The hash produced is encrypted with the sender's private key, resulting in a keyed checksum as output.

In Figure 1-12, the hash function takes as input the variable-length clear text data that needs to be authenticated and a private key. The private key length is the same as that of the output of the hash. The HMAC algorithm is run, with a resultant fixed-length checksum as output. This checksum value is sent with the message as a signature. The receiving peer runs an HMAC on the same message data that was input at the sender, using the same private key, and the resultant hash is compared with the received hash, which should exactly match.

Figure 1-12 *Hashed Message Authentication Codes (HMAC)*


Clear Message
Variable-length input message
Shared secret key
Hash Function
Hashed Checksum
Fixed-length authenticator value

HMAC-MD5-96

The HMAC-MD5-96 (also known as HMAC-MD5) encryption technique is used by IPSec to ensure that a message has not been altered. HMAC-MD5 uses the MD5 hash developed by Ronald Rivest of the Massachusetts Institute of Technology and RSA Data Security Incorporated and is described in RFC 1321.

HMAC-MD5 uses a 128-bit secret key. It produces a 128-bit authenticator value. This 128-bit value is truncated to the first 96 bits. Upon sending, the truncated value is stored within the authenticator field of AH or ESP-HMAC. Upon receipt, the entire 128-bit value is computed, and the first 96 bits are compared to the value stored in the authenticator field.

MD5 alone has recently been shown to be vulnerable to collision search attacks. This attack and other currently known weaknesses of MD5 do not compromise the use of MD5 within HMAC, as no known attacks against HMAC-MD5 have been proven. HMAC-MD5 is recommended where the superior performance of MD5 over SHA-1 is important.

HMAC-SHA-1-96

The HMAC-SHA-1-96 (also known as HMAC-SHA-1) encryption technique is used by IPSec to ensure that a message has not been altered. HMAC-SHA-1 uses the SHA-1 specified in FIPS-190-1, combined with HMAC (as per RFC 2104), and is described in RFC 2404.

HMAC-SHA-1 uses a 160-bit secret key. It produces a 160-bit authenticator value. This 160-bit value is truncated to the first 96 bits. Upon sending, the truncated value is stored within the authenticator field of AH or ESP-HMAC. Upon receipt, the entire 160-bit value is computed and the first 96 bits are compared to the value stored in the authenticator field.

SHA-1 is considered cryptographically stronger that MD5, yet it takes more CPU cycles to compute. HMAC-SHA-1 is recommended where the slightly superior security of SHA-1 over MD5 is important.

IKE Overview

IKE negotiates the IPSec security associations (SAs). This process requires that the IPSec systems first authenticate themselves to each other and establish ISAKMP, or IKE, shared keys.

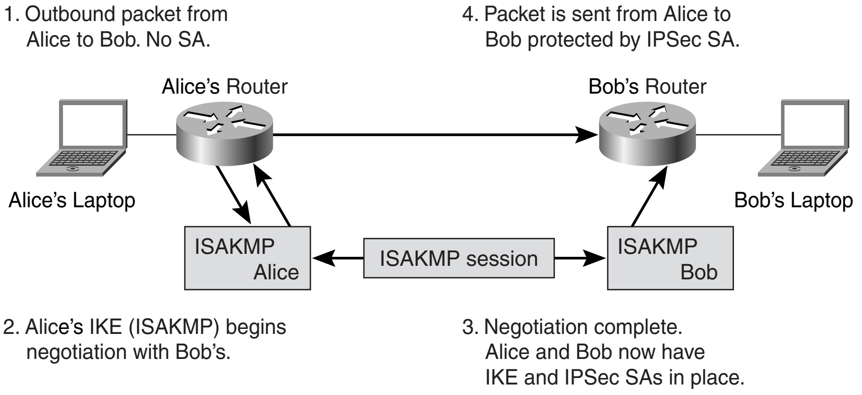
In phase one, IKE creates an authenticated secure channel between the two IKE peers that is called the IKE Security Association. The Diffie-Hellman key agreement is always performed in this phase.

In phase two, IKE negotiates the IPSec security associations and generates the required key material for IPSec. The sender offers one or more transform sets that are used to specify an allowed combination of transforms with their respective settings. The sender also indicates the data flow to which the transform set is to be applied. The sender must offer at least one transform set. The receiver then sends back a single transform set, which indicates the mutually agreed-on transforms and algorithms for this particular IPSec session. A new Diffie-Hellman agreement can be done in phase two, or the keys can be derived from the phase one shared secret.

Figure 1-13 shows the role that IKE takes in the IPSec VPN creation process.

NOTE A security association (SA) is a relationship between two or more entities that describes how the entities will use security services to communicate securely. SAs are covered in detail later in this chapter in the “IPSec Security Associations” section.

Figure 1-13 *The Function of IKE*


1. Outbound packet from Alice to Bob. No SA.
4. Packet is sent from Alice to Bob protected by IPSec SA.
Alice's Router
Bob's Router
Alice's Laptop
Bob's Laptop
ISAKMP Alice
ISAKMP session
ISAKMP Bob
2. Alice's IKE (ISAKMP) begins negotiation with Bob's.
3. Negotiation complete. Alice and Bob now have IKE and IPSec SAs in place.

• IKE sets up a secure channel to negotiate the IPSec security associations.

IKE authenticates the peer and the IKE messages between the peers during IKE phase one. Phase one consists of main mode or aggressive mode. Potential peers in an IPSec session must authenticate themselves to each other before IKE can proceed. Peer authentication occurs during the main mode exchange during IKE phase one. The IKE protocol is very flexible and supports multiple authentication methods as part of the phase one exchange. The two entities must agree on a common authentication protocol through a negotiation process. IKE phase one has three methods to authenticate IPSec peers in Cisco products, which are as follows:

- **Preshared keys**—A key value entered into each peer manually (out of band) used to authenticate the peer
- **RSA signatures**—Use a digital certificate authenticated by an RSA signature
- **RSA encrypted nonces**—Use RSA encryption to encrypt a nonce value (a random number generated by the peer) and other values

A common value used by all authentication methods is the peer identity (ID), which helps identify the peer. Some ID values used are as follows:

- IP address of the peer (four octets), such as 172.30.2.2
- Fully qualified domain name (FQDN), such as student@cisco.com

Preshared Keys

With preshared keys, the same preshared key is configured on each IPSec peer. IKE peers authenticate each other by computing and sending a keyed hash of data that includes the preshared key. If the receiving peer is able to create the same hash independently using its preshared key, it knows that both peers must share the same secret, thus authenticating the other peer. Preshared keys are easier to configure than manually configuring IPSec policy values on each IPSec peer. However, preshared keys do not scale well because each IPSec peer must be configured with the preshared key of every other peer with which it will establish a session.

RSA Signatures

The RSA signatures method uses a digital signature, where each device digitally signs a set of data and sends it to the other party. RSA signatures use a CA to generate a unique identity digital certificate that is assigned to each peer for authentication. The identity digital certificate is similar in function to the preshared key, but provides much stronger security.

RSA is a public-key cryptosystem used by IPSec for authentication in IKE phase 1. RSA was developed in 1977 by Ronald Rivest, Adi Shamir, and Leonard Adelman.

The initiator and the responder to an IKE session using RSA signatures send their own ID value (IDi, IDr), their identity digital certificate, and an RSA signature value consisting of a variety of IKE values, all encrypted by the negotiated IKE encryption method (DES or 3DES).

RSA Encryption

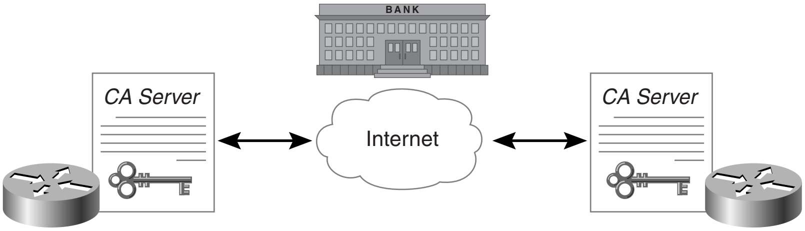
The RSA-encrypted nonces method uses the RSA encryption public key cryptography standard. The method requires that each party generates a pseudorandom number (a nonce) and encrypt it in the other party's RSA public key. Authentication occurs when each party decrypts the other party's nonce with a local private key (and other publicly and privately available information) and then uses the decrypted nonce to compute a keyed hash. This system provides for deniable transactions. That is, either side of the exchange can plausibly deny that it took part in the exchange. Cisco IOS software is the only Cisco product that uses RSA encrypted nonces for IKE authentication. RSA encrypted nonces use the RSA public key algorithm.

CAs and Digital Certificates

The distribution of keys in a public key scheme requires some trust. If the infrastructure is untrusted and control is questionable, such as on the Internet, distribution of keys is troublesome. RSA signatures are used by CAs, which are trusted third-party organizations. Verisign, Entrust, and Netscape are examples of companies that provide digital certificates. To get a digital certificate, a client registers with a CA. After a CA verifies the client's credentials, a certificate is issued. The digital certificate is a package that contains information such as a certificate bearer's identity: his or her name or IP address, the certificate's serial number, the certificate's expiration date, and a copy of the certificate bearer's public key. The standard digital certificate format is defined in the X.509

specification. X.509 version 3 defines the data structure for certificates and is the standard that Cisco supports. Figure 1-14 identifies some key points of CA operation.

Figure 1-14 *CAs and Digital Certificates*



- Certificate Authority (CA) verifies identity.
- CA signs digital certificate containing device's public key.
- VeriSign OnSite, Entrust PKI, Baltimore CA, Microsoft CA.

How IPSec Works

IPSec involves many component technologies and encryption methods. Yet IPSec's operation can be broken down into five main steps. The five steps are summarized as follows:

Step 1 **Interesting traffic initiates the IPSec process**—Traffic is deemed interesting when the IPSec security policy configured in the IPSec peers starts the IKE process.

Step 2 **IKE phase one**—IKE authenticates IPSec peers and negotiates IKE SAs during this phase, setting up a secure channel for negotiating IPSec SAs in phase two.

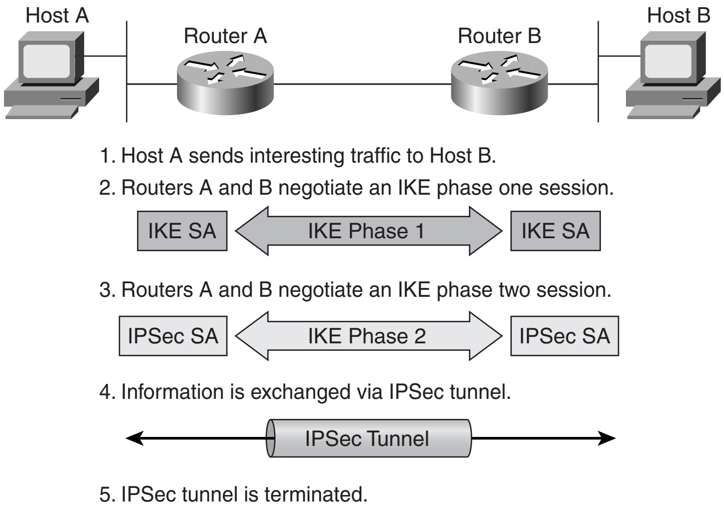
Step 3 **IKE phase two**—IKE negotiates IPSec SA parameters and sets up matching IPSec SAs in the peers.

Step 4 **Data transfer**—Data is transferred between IPSec peers based on the IPSec parameters and keys stored in the SA database.

Step 5 **IPSec tunnel termination**—IPSec SAs terminate through deletion or by timing out.

This five-step process is shown in Figure 1-15.

Figure 1-15 *The Five Steps of IPSec*



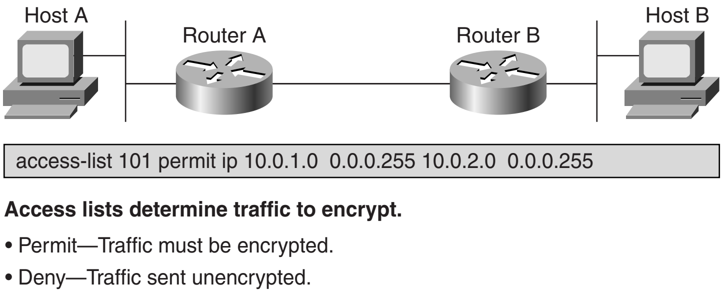
1. Host A sends interesting traffic to Host B.
2. Routers A and B negotiate an IKE phase one session.
3. Routers A and B negotiate an IKE phase two session.
4. Information is exchanged via IPSec tunnel.
5. IPSec tunnel is terminated.

Step 1: Defining Interesting Traffic

Determining what type of traffic is deemed interesting is part of formulating a security policy for use of a VPN. The policy is then implemented in the configuration interface for each particular IPSec peer. For example, in Cisco routers and PIX Firewalls, access lists are used to determine the traffic to encrypt. The access lists are assigned to a crypto policy such that permit statements indicate that the selected traffic must be encrypted, and deny statements can be used to indicate that the selected traffic must be sent unencrypted. With the Cisco Secure VPN Client, you use menu windows to select connections to be secured by IPSec. When interesting traffic is generated or transits the IPSec client, the client initiates the next step in the process, negotiating an IKE phase one exchange.

Step 1 is shown in Figure 1-16.

Figure 1-16 *Defining Interesting Traffic*



```
access-list 101 permit ip 10.0.1.0  0.0.0.255 10.0.2.0  0.0.0.255
```

Access lists determine traffic to encrypt.

- Permit—Traffic must be encrypted.
- Deny—Traffic sent unencrypted.

Step 2: IKE Phase One

The basic purpose of IKE phase one is to authenticate the IPSec peers and to set up a secure channel between the peers to enable IKE exchanges. IKE phase one performs the following functions:

- Authenticates and protects the identities of the IPSec peers
- Negotiates a matching IKE SA policy between peers to protect the IKE exchange
- Performs an authenticated Diffie-Hellman exchange with the end result of having matching shared secret keys
- Sets up a secure tunnel to negotiate IKE phase two parameters

IKE phase one occurs in two modes:

- Main mode
- Aggressive mode

Main Mode

Main mode has three two-way exchanges between the initiator and receiver.

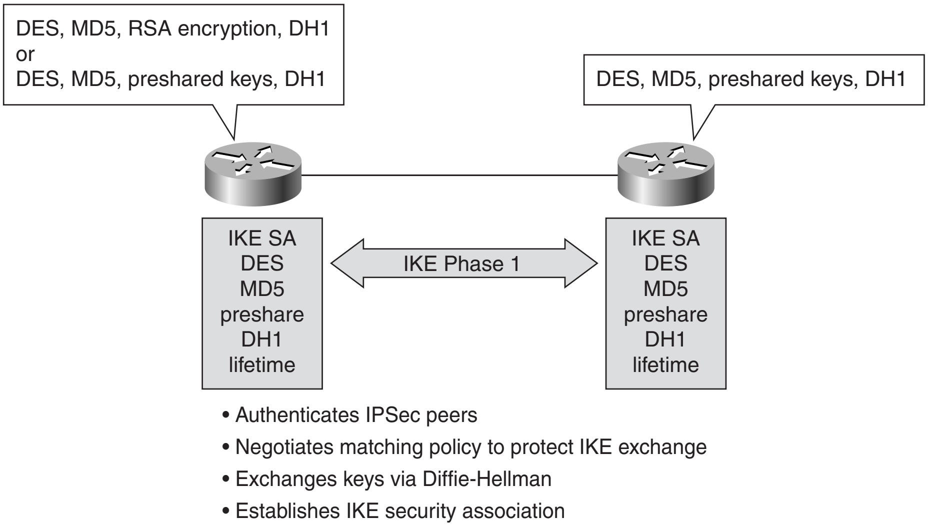
- **First exchange**—The algorithms and hashes used to secure the IKE communications are agreed upon in matching IKE SAs in each peer.
- **Second exchange**—This exchange uses a Diffie-Hellman exchange to generate shared secret keying material used to generate shared secret keys and to pass nonces, which are random numbers sent to the other party, signed, and returned to prove their identity.
- **Third exchange**—This exchange verifies the other side's identity. The identity value is the IPSec peer's IP address in encrypted form. The main outcome of main mode is matching IKE SAs between peers to provide a protected pipe for subsequent protected ISAKMP exchanges between the IKE peers. The IKE SA specifies values for the IKE exchange: the authentication method used, the encryption and hash algorithms, the Diffie-Hellman group used, the lifetime of the IKE SA in seconds or kilobytes, and the shared secret key values for the encryption algorithms. The IKE SA in each peer is bidirectional.

Aggressive Mode

In the aggressive mode, fewer exchanges are done and with fewer packets. In the first exchange, almost everything is squeezed into the proposed IKE SA values, the Diffie-Hellman public key, a nonce that the other party signs, and an identity packet, which can be used to verify the initiator's identity through a third party. The receiver sends everything back that is needed to complete the exchange. The only thing left is for the initiator to confirm the exchange. The weakness of using the aggressive mode is that both sides have exchanged information before there is a secure channel. Therefore, it is possible to sniff the wire and discover who formed the new SA. However, aggressive mode is faster than main mode.

Step 2 is shown in Figure 1-17.

Figure 1-17 *IKE Phase One*


DES, MD5, RSA encryption, DH1
or
DES, MD5, preshared keys, DH1
DES, MD5, preshared keys, DH1
IKE SA DES MD5 preshare DH1 lifetime
IKE Phase 1
IKE SA DES MD5 preshare DH1 lifetime
• Authenticates IPSec peers
• Negotiates matching policy to protect IKE exchange
• Exchanges keys via Diffie-Hellman
• Establishes IKE security association

Step 3: IKE Phase Two

The purpose of IKE phase two is to negotiate IPSec SAs to set up the IPSec tunnel. IKE phase two performs the following functions:

- Negotiates IPSec SA parameters protected by an existing IKE SA
- Establishes IPSec security associations
- Periodically renegotiates IPSec SAs to ensure security
- Optionally performs an additional Diffie-Hellman exchange

IKE phase 2 has one mode, called quick mode. Quick mode occurs after IKE has established the secure tunnel in phase one. It negotiates a shared IPSec policy, derives shared secret keying material used for the IPSec security algorithms, and establishes IPSec SAs. Quick mode exchanges nonces that provide replay protection. The nonces are used to generate new shared secret key material and prevent replay attacks from generating bogus SAs.

Quick mode is also used to renegotiate a new IPSec SA when the IPSec SA lifetime expires. Base quick mode is used to refresh the keying material used to create the shared secret key based on the keying material derived from the Diffie-Hellman exchange in phase one.

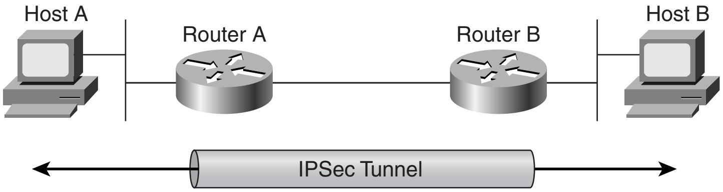
Perfect Forward Secrecy

If perfect forward secrecy (PFS) is specified in the IPSec policy, a new Diffie-Hellman exchange is performed with each quick mode, providing keying material that has greater entropy (key material life) and thereby greater resistance to cryptographic attacks. Each Diffie-Hellman exchange requires large exponentiations, thereby increasing CPU use and exacting a performance cost.

Step 4: IPSec Encrypted Tunnel

After IKE phase two is complete and quick mode has established IPSec SAs, information is exchanged by an IPSec tunnel. Packets are encrypted and decrypted using the encryption specified in the IPSec SA. This IPSec encrypted tunnel can be seen in Figure 1-18.

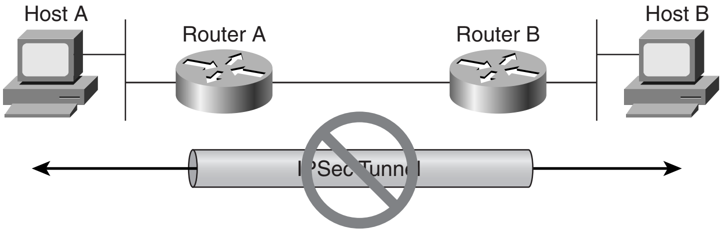
Figure 1-18 *IPSec Encrypted Tunnel*



Step 5: Tunnel Termination

IPSec SAs terminate through deletion or by timing out. An SA can time out when a specified number of seconds have elapsed or when a specified number of bytes have passed through the tunnel. When the SAs terminate, the keys are also discarded. When subsequent IPSec SAs are needed for a flow, IKE performs a new phase two and, if necessary, a new phase one negotiation. A successful negotiation results in new SAs and new keys. New SAs can be established before the existing SAs expire so that a given flow can continue uninterrupted. This can be seen in Figure 1-19.

Figure 1-19 *Tunnel Termination*



IPSec Security Associations (SAs)

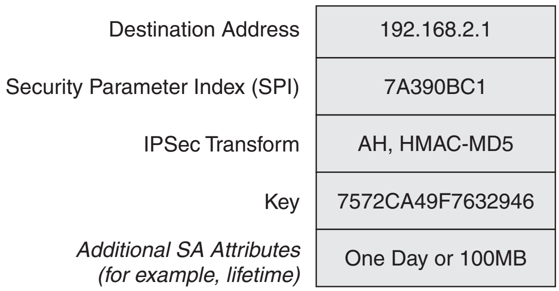
The concept of a security association (SA) is fundamental to IPSec. An SA is a relationship between two or more entities that describes how the entities will use security services to communicate securely. IPSec provides many options for performing network encryption and authentication. Each IPSec connection can provide encryption, integrity, authenticity, or all three. When the security service is determined, the two IPSec peers must determine exactly which algorithms to use (for example, DES or 3DES for encryption, MD5 or SHA for integrity). After deciding on the algorithms, the two devices must share session keys. As you can see, there is quite a bit of information to manage. The security association is the method that IPSec uses to track all the particulars concerning a given IPSec communication session. You will need to configure SA parameters and monitor SAs on Cisco routers and the PIX Firewall.

NOTE The nomenclature gets a little confusing at times, because SAs are used for more than just IPSec. For example, IKE SAs describe the security parameters between two IKE devices.

A separate pair of IPSec SAs are set up for AH and ESP transform. Each IPSec peer agrees to set up SAs consisting of policy parameters to be used during the IPSec session. The SAs are unidirectional for IPSec so that peer 1 will offer peer 2 a policy. If peer 2 accepts this policy, it will send that policy back to peer 1. This establishes two one-way SAs between the peers. Two-way communication consists of two SAs, one for each direction.

Each SA consists of values such as destination address, a security parameter index (SPI), the IPSec transforms used for that session, security keys, and additional attributes such as IPSec lifetime. The SAs in each peer have unique SPI values that will be recorded in the Security Parameter Databases of the devices. The Security Parameter Database is set up in dynamic random-access memory (DRAM) and contains parameter values for each SA. An example of these values is shown in Figure 1-20.

Figure 1-20 *IPSec Security Association*

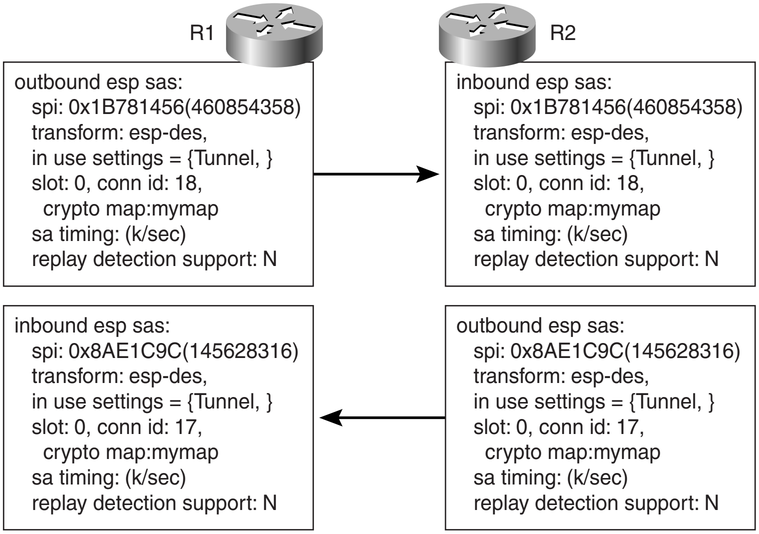


Destination Address	192.168.2.1
Security Parameter Index (SPI)	7A390BC1
IPSec Transform	AH, HMAC-MD5
Key	7572CA49F7632946
Additional SA Attributes (for example, lifetime)	One Day or 100MB

An IPSec transform in Cisco IOS specifies either an AH or an ESP protocol and its corresponding algorithms and mode (transport or tunnel). The Cisco Secure VPN Client uses the concept of security policies to specify the same parameters. Transforms, transform sets, and the corresponding security policies of the Cisco Secure VPN Client are explained in detail in Chapter 12, "Scaling Cisco IPSec-Based VPNs."

Figure 1-21 contains an actual example of SA parameters for two IPSec peers: R1 and R2. Remember that each IPSec SA is unidirectional, and the SA parameters must match on each IPSec peer. The SA parameters are configured by the system administrator and are stored in the SA database. Table 1-1 contains a description of the parameters shown in Figure 1-21.

Figure 1-21 *SA Parameter Example on a Cisco Router*



R1:

```
outbound esp sas:
   spi: 0x1B781456(460854358)
   transform: esp-des,
   in use settings = {Tunnel, }
   slot: 0, conn id: 18,
     crypto map:mymap
   sa timing: (k/sec)
   replay detection support: N
```

R2:

```
inbound esp sas:
   spi: 0x1B781456(460854358)
   transform: esp-des,
   in use settings = {Tunnel, }
   slot: 0, conn id: 18,
     crypto map:mymap
   sa timing: (k/sec)
   replay detection support: N
```

R1:

```
inbound esp sas:
   spi: 0x8AE1C9C(145628316)
   transform: esp-des,
   in use settings = {Tunnel, }
   slot: 0, conn id: 17,
     crypto map:mymap
   sa timing: (k/sec)
   replay detection support: N
```

R2:

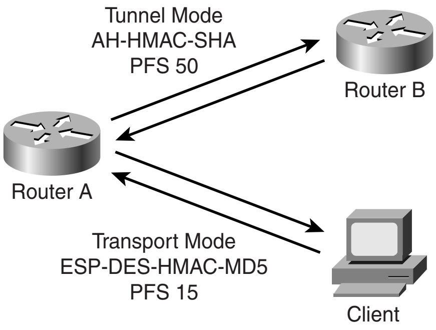
```
outbound esp sas:
   spi: 0x8AE1C9C(145628316)
   transform: esp-des,
   in use settings = {Tunnel, }
   slot: 0, conn id: 17,
     crypto map:mymap
   sa timing: (k/sec)
   replay detection support: N
```

Table 1-1 *SA Parameters*

Parameter	Description
outbound esp sas: spi: 0x1B781456(460854358)	Security parameter index, which matches inbound SPI for that SA
transform: esp-des	IPSec transform
in use settings ={Tunnel, }	IPSec transform mode (tunnel or transport)
slot: 0, conn id: 18, crypto map:mymap	Crypto engine and crypto map information
sa timing: (k/sec)	SA lifetime in KB and seconds
replay detection support: N	Replay detection either on or off

The SAs between IPSec peers enable the configured IPSec policy. When a system sends a packet that requires IPSec protection, it looks up the SA in its database, applies the specified processing, and then inserts the SPI from the SA into the IPSec header. When the IPSec peer receives the packet, it looks up the SA in its database by destination address and SPI, and then processes the packet as required. In summary, the SA is a statement of the negotiated security policy between two devices. Figure 1-22 shows an example of differing policies between peers.

Figure 1-22 *SAs Enable Your Chosen Policy*

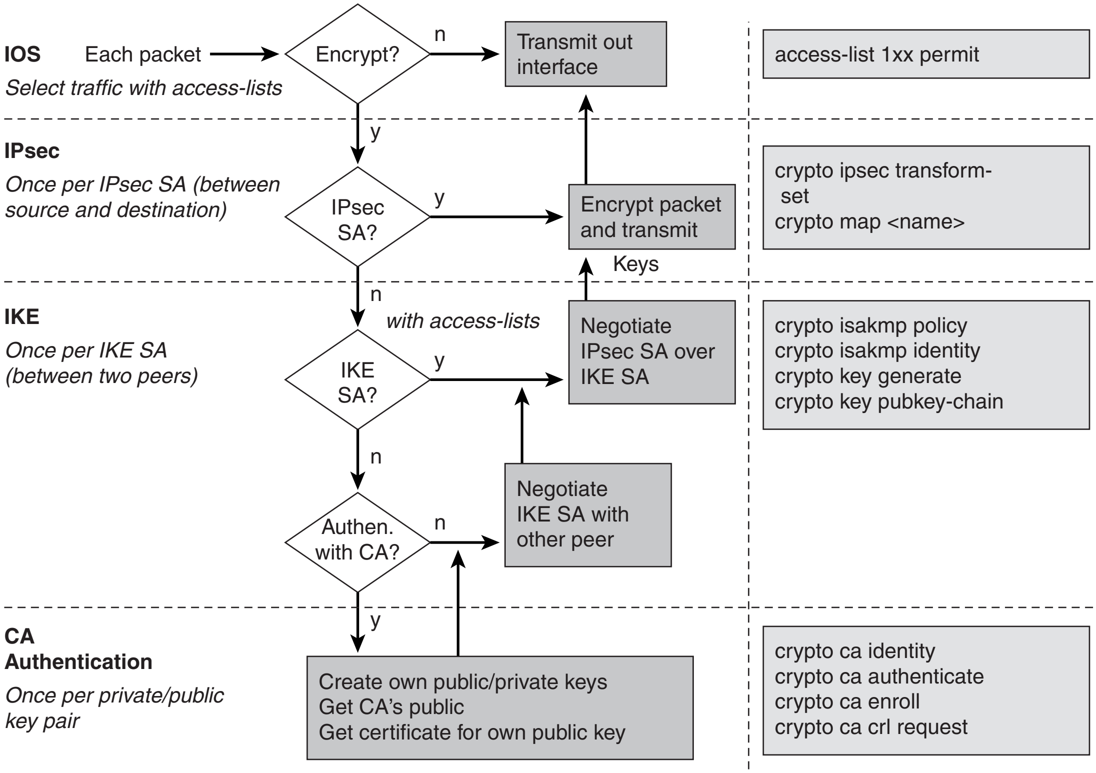


IKE and IPSec Flowchart for Cisco Routers

Cisco IOS software implements and processes IPSec in a predictable and reliable fashion. A summary of how IPSec works in Cisco IOS software is shown in Figure 1-23. The process shown in Figure 1-23 assumes that you have already created your own public and private keys and that at least one access list exists. Figure 1-23 also shows the Cisco IOS commands used to configure each part of the process, although the commands are not shown in the order in which you enter them, which is covered in Chapters 3, “Configuring

Cisco IOS Routers for Preshared Keys Site-to-Site" and 4, "Configuring Cisco IOS Routers for CA Site-to-Site."

Figure 1-23 *IKE and IPSec Flowchart*



NOTE Remember, IKE is synonymous with ISAKMP in Cisco router or PIX Firewall configurations.

The following steps describe the IPSec process.

Step 1 Access lists applied to an interface and crypto map are used by Cisco IOS software to select interesting traffic to be encrypted.

Step 2 Cisco IOS software checks to see if IPSec SAs have been established.

Step 3 If the SA has already been established by manual configuration using the **crypto ipsec transform-set** and **crypto map** commands or has been previously set up by IKE, the packet is encrypted based on the policy specified in the crypto map and is transmitted out of the interface.

Step 4 If the SA has not been established, Cisco IOS software checks to see if an IKE SA has been configured and set up.

Step 5 If the IKE SA has been set up, the IKE SA governs negotiation of the IPSec SA as specified in the IKE policy configured by the **crypto isakmp policy** command, the packet is encrypted by IPSec, and it is transmitted.

Step 6 If the IKE SA has not been set up, Cisco IOS software checks to see if certification authority (CA) has been configured to establish an IKE policy.

Step 7 If CA authentication is configured with the various **crypto ca** commands, the router uses public and private keys previously configured, obtains the CA's public certificate, gets a certificate for its own public key, and then uses the key to negotiate an IKE SA, which in turn is used to establish an IPSec SA to encrypt and transmit the packet.

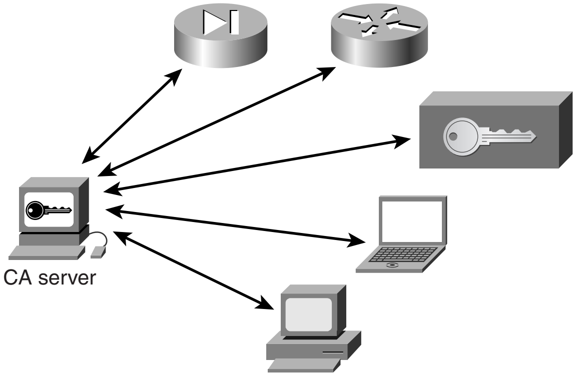
CA Support Overview

With a CA, you do not need to configure keys among all of the encrypting IPSec peers. Instead, you individually enroll each participating peer with the CA and request a certificate. When this has been accomplished, each participating peer can dynamically authenticate all of the other participating routers. To add a new IPSec peer to the network, you only need to configure that new peer to request a certificate from the CA, instead of making multiple key configurations with all the other existing IPSec peers.

This section presents an overview of how CA support works.

CA servers are responsible for managing certificate requests and issuing certificates to participating IPSec network devices. These services provide centralized key management for the participating devices. CAs simplify the administration of IPSec network devices so that IPSec keys do not have to be manually configured on each peer. You can use a CA with a network containing multiple IPSec-compliant devices, such as PIX Firewalls, Cisco routers, the Cisco VPN 3000 Concentrator series, the Cisco Secure VPN Client, and other vendors' IPSec products, as shown in Figure 1-24.

Figure 1-24 *CA Support*



Digital signatures, enabled by public key cryptography, provide a means to digitally authenticate devices and individual users. In public key cryptography, such as the RSA signature system, each user has a key pair containing both a public and a private key. The keys act as complements, and anything encrypted with one of the keys can be decrypted with the other. An RSA signature is formed when data is encrypted with a user's private key. The receiver verifies the signature by decrypting the message with the sender's public key.

The fact that the message could be decrypted using the sender's public key indicates that the holder of the private key—the sender—must have created the message. This process relies on the receiver having a copy of the sender's public key and knowing with a high degree of certainty that it really does belong to the sender, and not to someone pretending to be the sender.

A digital certificate contains information to identify a user or device, such as the name, serial number, company, department, or IP address. It also contains a copy of the entity's public key. A CA signs the certificate. The CA is a third party that is explicitly trusted by the receiver to validate identities and to create digital certificates.

To validate the CA's signature, the receiver must first know the CA's public key. Normally this is handled out-of-band or through an operation done at installation. For instance, most Web browsers are configured with the public keys of several CAs by default. The IKE, a key component of IPSec, can use digital signatures to authenticate peer devices before setting up SAs, while simultaneously providing scalability.

Without digital signatures, you must manually exchange either public keys or secret keys between each pair of devices that use IPSec to protect communications between them. Without certificates, every new device added to the network requires a configuration change on every other device with which it securely communicates. However, by using digital certificates, each device is enrolled with a CA. When two devices wish to communicate, they exchange certificates and digitally sign data to authenticate each other. When a new device is added to the network, one simply enrolls that device with a CA, and none of the other devices need modification. When the new device attempts an IPSec connection, certificates are automatically exchanged and the device can be authenticated. Without CA interoperability, devices could not use CAs when deploying IPSec. CAs provide a manageable, scaleable solution for IPSec networks.

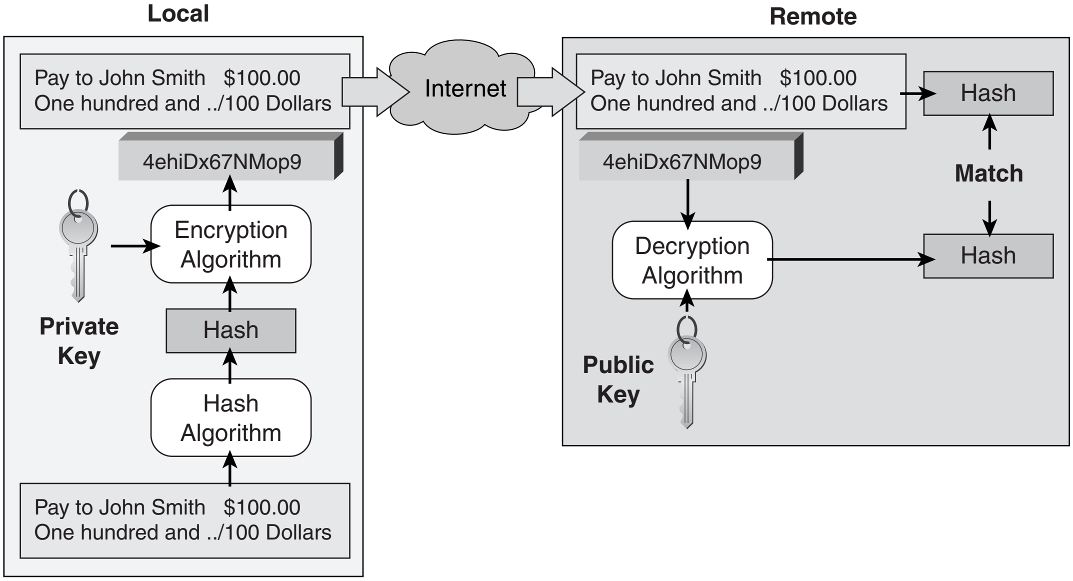
Digital Signatures

The digital signature provides a form of digital credentials that authenticate the identity of the sending party, whoever that may be. In other words, digital signatures are used to link data with the holder of a specific private key and consist of the following:

- At the local end, a private key is used to encrypt the hash.
- At the remote end:
  - — The hash is produced by running the original message through a hash algorithm.
  - — The hash that was appended to the original message is decrypted using the sender's public key.
- If the hashes match, the message is signed by a private key.
- Only a specific private key could have produced the digital signature.

Figure 1-25 shows the digital signature process.

Figure 1-25 *Digital Signatures*


Local
Remote
Pay to John Smith $100.00
One hundred and ../100 Dollars
Internet
Pay to John Smith $100.00
One hundred and ../100 Dollars
Hash
4ehiDx67NMop9
4ehiDx67NMop9
Match
Encryption Algorithm
Decryption Algorithm
Hash
Private Key
Hash
Public Key
Hash Algorithm
Pay to John Smith $100.00
One hundred and ../100 Dollars

A key pair has no intrinsic ties to any person or entity. It could be sourced from Alice, Tom, or Harry Hacker masquerading as Alice or Tom. A solution is necessary to reliably tie a person or entity to a key pair. Digital signatures provide a way to "guarantee" the source of the message. The solution is digital signatures and digital certificates.

- **Digital signatures**—Tie a message to a sender's private key. The hash can only be decrypted by the sender's public key.
- **Digital certificates**—Bind a person or entity to a private key.

Certificate-Based Authentication

Digital certificates are used to authenticate users. They can be used to identify a person, a company, or a server. They are the equivalent of a digital passport or driver's license. The following example and Figure 1-26 illustrate how this works.

Step 1 Users A and B register separately with the CA.

— Digital certificates are issued by a trusted third party, a CA.

— The CA issues separate certificates and digitally signs them with its private key, thereby certifying the authenticity of the user.

Step 2 User A sends the certificate to User B.

Step 3 User B checks the authenticity of the CA signature on the certificate.

— The CA public key is used to verify the CA signature on the certificate.

— If it passes validation, it is "safe" to assume User A is who he says he is, therefore the message is valid.

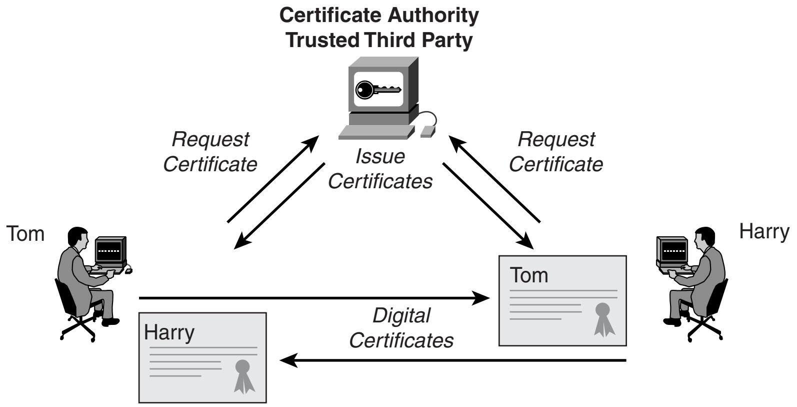
Step 4 User B sends the certificate to User A.

— The CA public key is used to verify the CA signature on the certificate.

— Once verified, all subsequent communications can be accepted.

NOTE Certificates are exchanged during the IPSec negotiations.

Figure 1-26 *Certificate-Based Authentication*



CAs

CAs hold the key to the public key infrastructure (PKI). A CA is a trusted third party whose job is to certify the authenticity of users to ensure that you are who you say you are.

Authenticity is guaranteed by the CA digital signature created with the CA private key. You can verify a digital signature using the CA public key. Only the CA public key can decrypt the digital certificate. The job of a CA is to

- Create certificates
- Administer certificates
- Revoke invalid certificates

The CA can be a corporate network administrator or a recognized third party. Trusted sources supported by the Cisco VPN 3000 Concentrator Series include the following:

- Entrust
- GTE Cybertrust
- Network Associates PGP
- Baltimore
- Microsoft
- Verisign

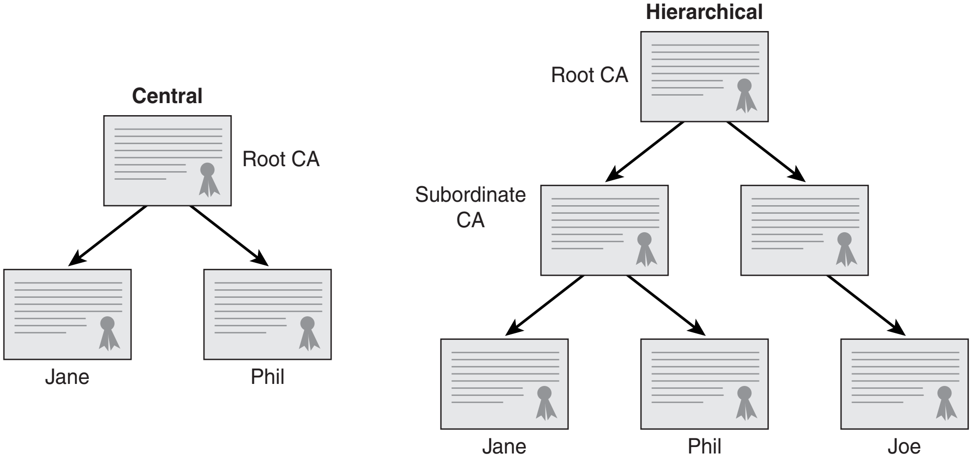
Some CAs also use a Registration Authority (RA) to provide certificate enrollment services. An RA is server software that acts as a proxy for the CA, providing essential CA functions, such as certificate enrollment and distribution.

PKI

PKI is the set of hardware, software, people, policies, and procedures needed to create, manage, store, distribute, and revoke digital certificates. PKI makes it possible to generate and distribute keys within a secure domain and enables CAs to issue keys and associated certificate and certificate revocation lists (CRLs) in a secure manner. There are two PKI models, as shown in the following list and Figure 1-27.

- Central authority
  - — All certificates are signed by a single authority.
  - — All certificates can be checked with that CA's public key.
- Hierarchical authority
  - — The ability to sign a certificate is delegated through a hierarchy. The top of the hierarchy is the *root CA*. It signs certificates for subordinate authorities.
  - — Subordinate CAs sign certificates for lower-level CAs.
  - — To validate a user's certificate, the certificate must be validated up through the chain of authority.

Figure 1-27 *PKI*



Summary

This chapter provided a very detailed overview of VPNs with a concentration on using IPSec as a VPN technology. It started by covering the various VPN components such as the Cisco Secure PIX Firewall, Cisco routers, and the Cisco VPN Concentrator. It then covered the technicalities of IPSec and the components that make up IPSec.

The chapter covered the five-step process of IPSec VPN establishment that includes IKE phase one and IKE phase two.

The chapter finished by looking at IPSec security associations (SAs) and also provided an overview of the certificate authority (CA) process.

Now that you have a foundation of knowledge on IPSec and VPN terminology, the next chapter looks at the individual VPN components and the configuration challenges that each one brings.

Review Questions

1 What are the three types of VPNs?

2 What type of VPNs link outside customers, suppliers, partners, or communities of interest to an enterprise customer's network over a shared infrastructure using dedicated connections?

3 IPSec consists of which two components?

4 You configure an IPSec transform set to use AH. Is the data payload encrypted?

5 You want to establish an extranet VPN over the Internet. Which type of IPSec mode (transport or tunnel) would be the best solution in this scenario?

6 Which is the most secure encryption algorithm: DES, 3DES, or Diffie-Hellman?

7 Step one of configuring IPSec is defining interesting traffic. What Cisco IOS feature do you use to define this?

8 With preshared keys, can each of the keys be different or must each be the same (have the same value as the other's public key)?

9 What is used to relay the shared key to the VPN peer?

10 IKE peers authenticate themselves using one of four methods. What are these four methods?