INTRODUCTION
Key management is an important aspect of IPSec or any encrypted communication that uses keys to provide information confidentiality and integrity. Key management and the protocols utilized are implemented to set up, maintain, and control secure relationships and ultimately the VPN between systems. During key management, there are several layers of system insurance prior to the establishment of an SA, and there are several mechanisms used to accommodate these processes.

KEY HISTORY
Key management is far from obvious definition, and lackadaisical conversation with interchanged acronyms only adds to the perceived misunderstandings. The following is an outline of the different protocols that are used to get keys and data from one system to another.

The Internet Security Association and Key Management Protocol (ISAKMP) (RFC 2408) defines the procedures for authenticating a communicating peer and key generation techniques. All of these are necessary to establish and maintain an SA in an Internet environment. ISAKMP defines payloads for exchanging key and authentication data. As shown Exhibit 1, these formats provide a consistent framework that is independent of the encryption algorithm, authentication mechanism being implemented, and security protocol, such as IPSec.

The Internet Key Exchange (IKE) protocol (RFC 2409) is a hybrid containing three primary, existing protocols that must be understood by the information security officers involved with implementing IPSec. The crux of the matter involves establishing a Public Key Infrastructure (PKI) that provides several areas of secure communication based on trust and digital certificates. This article explains the key protocols and the use of both symmetric and asymmetric forms of encryption to achieve the confidentiality and integrity needed in these systems.
cols that are combined to provide an IPSec-specific key management platform. The three protocols are:

1. ISAKMP
2. Oakley
3. SKEME (Secure Key Exchange Mechanism)

Different portions of each of these protocols work in conjunction to securely provide keying information specifically for the IETF IPSec DOI. The terms IKE and ISAKMP are used interchangeably with various vendors, and many use ISAKMP to describe the keying function. While this is correct, ISAKMP addresses the procedures and not the technical operations as they pertain to IPSec. IKE is the term that best represents the IPSec implementation of key management.

Public Key Infrastructure (PKI) is a suite of protocols that provide several areas of secure communication based on trust and digital certificates. PKI integrates digital certificates, public key cryptography, and certificate authorities into a total, enterprisewide network security architecture that can be utilized by IPSec.

IPSec IKE

As described earlier, IKE is a combination of several existing key management protocols that are combined to provide a specific key management system. IKE is considerably complicated, and several variations are available in the establishment of trust and providing keying material.

Oakley and ISAKMP protocols, which are included in IKE, each define separate methods of establishing an authenticated key exchange between systems. Oakley defines modes of operation to build a secure relationship path, and ISAKMP defines phases to accomplish much the same process in a hierarchical format. The relationship between these
two is represented by IKE with different exchanges as modes, which operate in one of two phases. Implementing multiple phases may add overhead in processing, resulting in performance degradation, but several advantages can be realized. Some of these are:

• first phase creation assisted by second phase
• first phase key material used in second phase
• first phase trust used for second phase

The first phase session can be disbursed among several second phase operations to provide the construction of new ISAKMP security associations (ISA for purposes of clarity in this document) without the renegotiation process between the peers. This allows for the first phase of subsequent ISAs to be preempted via communications in the second phase.

Another benefit is that the first phase process can provide security services for the second phase in the form of encryption keying material. However, if the first phase does not meet the requirements of the second phase, no data can be exchanged or provided from the first to the second phase.

With the first phase providing peer identification, the second phase may provide the creation of the security protocol SAs without the concern for authentication of the peer. If the first phase were not available, each new SA would need to authenticate the peer system. This function of the first phase is an important feature for IPSec communications. Once peers are authenticated by means of certificates or shared secret, all communications of the second phase and internal to the IPSec SAs are authorized for transport. The remaining authentication is for access control. By this point, the trusted communication has been established at a higher level.

PHASES AND MODES
Phase one takes place when the two ISAKMP peers establish a secure, authenticated channel with which to communicate. Each system is verified and authenticated against its peer to allow for future communications. Phase two exists to provide keying information and material to assist in the establishment of SAs for an IPSec communication.

Within phase one, there are two modes of operation defined in IKE: main mode and aggressive mode. Each of these accomplishes a phase one secure exchange, and these two modes only exist in phase one. Within phase two, there are two modes: quick mode and new group mode.

Quick Mode is used to establish SAs on behalf of the underlying security protocol. New Group Mode is designated as a phase two mode only because it must exist in phase two; however, the service provided by
New Group Mode is to benefit phase one operations. As described earlier, one of the advantages of a two-phase approach is that the second phase can be used to provide additional ISAs, which eliminates the reauthorization of the peers.

Phase one is initiated using ISAKMP-defined cookies. The initiator cookie (I-cookie) and responder cookie (R-cookie) are used to establish an ISA, which provides end-to-end authenticated communications. That is, ISAKMP communications are bi-directional and, once established, either peer may initiate a Quick mode to establish SA communications for the security protocol. The order of the cookies is crucial for future second phase operations. A single ISA can be used for many second phase operations, and each second phase operation can be used for several SAs or SA Bundles. Main Mode and Aggressive Mode each use Diffie-Hellman keying material to provide authentication services.

While Main Mode must be implemented, Aggressive Mode is not required. Main Mode provides several messages to authenticate. The first two messages determine a communication policy; the next two messages exchange Diffie-Hellman public data; and the last two messages authenticate the Diffie-Hellman Exchange. Aggressive Mode is an option available to vendors and developers that provides much more information with fewer messages and acknowledgements. The first two messages in Aggressive Mode determine a communication policy and exchange Diffie-Hellman public data. In addition, a second message authenticates the responder; thus completing the negotiation.

Phase two is much simpler in nature in that it provides keying material for the initiation of SAs for the security protocol. This is the point where key management is utilized to maintain the SAs for IPSec communications. The second phase has one mode designed to support IPSec: Quick Mode. Quick Mode verifies and establishes the keying process for the creation of SAs. Not related directly to IPSec SAs is the New Group Mode of operation; New Group provides services for phase one for the creation of additional ISAs.

SYSTEM TRUST ESTABLISHMENT
The first step in establishing communications is verification of the remote system. There are three primary forms of authenticating a remote system:

1. shared secret
2. certificate
3. public/private key

Of these methods, shared secret is currently used widely due to the relatively slow integration of Certificate Authority (CA) systems and the ease of implementation. However, shared secret is not scalable and can become
unmanageable very quickly due to the fact that there can be a separate secret for each communication. Public and private key use is employed in combination with Diffie-Hellman to authenticate and provide keying material. During the system authentication process, hashing algorithms are utilized to protect the authenticating shared secret as it is forwarded over untrusted networks. This process of using hashing to authenticate is nearly identical to the authentication process of an AH security protocol. However, the message — in this case a password — is not sent with the digest. The map previously shared or configured with participating systems will contain the necessary data to be compared to the hash.

An example of this process is a system, called system A, that requires a VPN to a remote system, called system B. By means of a preconfigured map, system A knows to send its hashed shared secret to system B to access a network supported by system B. System B will hash the expected shared secret and compare it to the hash received from system A. If the two hashes match, an authenticated trust relationship is established.

Certificates are a different process of trust establishment. Each device is issued a certificate from a CA. When a remote system requests communication establishment, it will present its certificate. The recipient will query the CA to validate the certificate. The trust is established between the two systems by means of an ultimate trust relationship with the CA and the authenticating system. Seeing that certificates can be made public and are centrally controlled, there is no need to attempt to hash or encrypt the certificate.

KEY SHARING
Once the two systems are confident of each other’s identity, the process of sharing or swapping keys must take place to provide encryption for future communications. The mechanisms that can be utilized to provide keying are related to the type of encryption to be utilized for the ESP. There are two basic forms of keys: symmetrical and asymmetrical.

Symmetrical key encryption occurs when the same key is used for the encryption of information into human unintelligible data (or cipher text) and the decryption of that cipher text into the original information format. If the key used in symmetrical encryption is not carefully shared with the participating individuals, an attacker can obtain the key, decrypt the data, view or alter the information, encrypt the data with the stolen key, and forward it to the final destination. This process is defined as a man-in-the-middle attack and, if properly executed, can affect data confidentiality and integrity, rendering the valid participants in the communication oblivious to the exposure and the possible modification of the information.

Asymmetrical keys consist of a key-pair that is mathematically related and generated by a complicated formula. The concept of asymmetrical
comes from the fact that the encryption is one way with either of the key-pair, and data that is encrypted with one key can only be decrypted with the other key of the pair. Asymmetrical key encryption is incredibly popular and can be used to enhance the process of symmetrical key sharing. Also, with the use of two keys, digital signatures have evolved and the concept of trust has matured to certificates, which contribute to a more secure relationship.

**Symmetrical Keys**
Symmetrical keys are an example of DES encryption, where the same keying information is used to encrypt and decrypt the data. However, to establish communications with a remote system, the key must be made available to the recipient for decryption purposes. In early cases, this may have been a phone call, e-mail, fax, or some form of nonrelated communication medium. However, none of these options are secure or can communicate strong encryption keys that require a sophisticated key that is nearly impossible to convey in a password or phrase.

In 1976, two mathematicians, Bailey W. Diffie at Berkeley and Martin E. Hellman at Stanford, defined the Diffie–Hellman agreement protocol (also known as exponential key agreement) and published it in a paper entitled, “New Directions in Cryptography.” The protocol allows two autonomous systems to exchange a secret key over an untrusted network without any prior secrets. Diffie and Hellman postulated that the generation of a key could be accomplished by fundamental relationships between prime numbers. Some years later, Ron Rivest, Adi Shamir, and Leonard Adelman, who developed the RSA Public and Private key cryptosystem based on large prime numbers, further developed the Diffie–Hellman formula (i.e., the nuts and bolts of the protocol). This allowed communication of a symmetrical key without transmitting the actual key, but rather a mathematical portion or fingerprint.

An example of this process is system A and system B requires keying material for the DES encryption for the ESP to establish a SA. Each system acquires the Diffie–Hellman parameters, a large prime number \( p \) and a base number \( g \), which must be smaller than \( p - 1 \). The generator, \( g \), is a number that represents every number between 1 and \( p \) to the power of \( k \). Therefore, the relationship is \( g^k \equiv n \mod p \).

Both of these numbers must be hardcoded or retrieved from a remote system. Each system then generates a number \( X \), which must be less than \( p - 2 \). The number \( X \) is typically created by a random string of characters entered by a user or a passphrase that can be combined with date and time to create a unique number. The hardcoded numbers will not be exceeded because most, if not all, applications employ a limit on the input.

As shown in Exhibit 2, a new key is generated with these numbers, \( g^X \mod p \). The result \( Y \), or fingerprint, is then shared between the systems.
over the untrusted network. The formula is then exercised again using the shared data from the other system and the Diffie–Hellman parameters. The results will be mathematically equivalent and can be used to generate a symmetrical key. If each system executes this process successfully, they will have matching symmetrical keys without transmitting the key itself. The Diffie–Hellman protocol was finally patented in 1980 (U.S. Patent 4200770) and is such a strong protocol that there are currently 128 other patents that reference Diffie–Hellman.

To complicate matters, Diffie–Hellman is vulnerable to man-in-the-middle attacks because the peers are not authenticated using Diffie-Hellman. The process is built on the trust established prior to keying material creation. To provide added authentication properties within the Diffie-Hellman procedure, the Station-to-Station (STS) protocol was created. Diffie, Oorschot, and Wiener completed STS in 1992 by allowing the two parties to authenticate themselves to each other by the use of digital signatures created by a public and private key relationship.

An example of this process, as shown in Exhibit 3, transpires when each system is provided a public and private key-pair. System A will encrypt the Y value (in this case Ya) with the private key. When system B receives the signature, it can only be decrypted with the system A public key. The only plausible result is that system A encrypted the Ya value authenticating system A. The STS protocol allows for the use of certificates to further authorize the public key of system A to ensure that the man-in-the-middle has not compromised the key-pair integrity.
Many Keys
Asymmetrical keys, such as PGP (Pretty Good Privacy) and RSA, can be used to share the keying information. Asymmetrical keys were specifically designed to have one of the keys in a pair published. A sender of data can obtain the public key of the preferred recipient to encrypt data that can only be decrypted by the holder of the corresponding private key. The application of asymmetrical keys in the sharing of information does not require the protection of the public key in transit over an untrusted network.

KEY ESTABLISHMENT
IPSec standard mandates that key management must support two forms of key establishment: manual and automatic.

The other IPSec protocols (AH and ESP) are not typically affected by the type of key management. However, there may be issues with implementing anti-replay options, and the level of authentication can be related to the key management process supported. Indeed, key
management can also be related to the ultimate security of the communication. If the key is compromised, the communication can be endangered of attack. To thwart the eventuality of such an attack, there are re-keying mechanisms that attempt to ensure that if a key is compromised its validity is limited either by time, amount of data encrypted, or a combination of both.

Manual Keying
Manual key management requires that an administrator provide the keying material and necessary security association information for communications. Manual techniques are practical for small environments with limited numbers of gateways and hosts. Manual key management does not scale to include many sites in a meshed or partially meshed environment. An example is a company with five sites throughout North America. This organization wants to use the Internet for communications, and each office site must be able to communicate directly with any other office site. If each VPN relationship had a unique key, the number of keys can be calculated by the formula \( n(n - 1)/2 \), where \( n \) is the number of sites. In this example, the number of keys is 10. Apply this formula to 25 sites (i.e., five times the number of sites in the previous example) and the number of keys skyrocket to 300, not 50. In reality, the management is more difficult than it may appear by the examples. Each device must be configured, and the keys must be shared with all corresponding systems. The use of manual keying conspires to reduce the flexibility and options of IPSec. Anti-replay, on-demand re-keying, and session-specific key management are not available in manual key creation.

Automatic Keying
Automatic key management responds to the limited manual process and provides for widespread, automated deployment of keys. The goal of the IPSec is to build off existing Internet standards to accommodate a fluid approach to interoperability. As described earlier, the IPSec default automated key management is IKE, a hybrid based in ISAKMP. However, based on the structure of the standard, any automatic key management can be employed. Automated key management, when instituted, may create several keys for a single SA. There are various reasons for this, including:

- encryption algorithm requires more than one key
- authentication algorithm requires more than one key
- encryption and authentication are used for a single SA
- re-keying

The encryption and authentication algorithms’ use of multiple keys, or if both algorithms are used, then multiple keys will need to be generated
for the SA. An example of this would be if Triple-DES is used to encrypt the data. There are several types of applications of Triple-DES (DES-EEE3, DES-EDE3, and DES-EEE2) and each uses more than one key (DES-EEE2 uses two keys, one of which is used twice).

The process of re-keying is to protect future data transmissions in the event a key is compromised. This process requires the rebuilding of an existing SA. The concept of re-keying during data transmission provides a relatively unpredictable communication flow. Being unpredictable is considered a valuable security method against an attacker.

Automatic key management can provide two primary methods of key provisioning:

- multiple string
- single string

Multiple strings are passed to the corresponding system in the SA for each key and for each type. For example, the use of Triple-DES for the ESP will require more than one key to be generated for a single type of algorithm, in this case, the encryption algorithm. The recipient will receive a string of data representing a single key; once the transfer has been acknowledged, the next string representing another key will be transmitted.

In contrast, the single string method sends all the required keys in a single string. As one might imagine, this requires a stringent set of rules for management. Great attention is necessary to ensure that the systems involved properly map the corresponding bits to the same key strings for the SA being established. To ensure that IPSec-compliant systems properly map the bit to keys, the string is read from the left, highest bit order first for the encryption key(s) and the remaining string is used for the authentication. The number of bits used is determined by the encryption algorithm and the number of keys required for the encryption being utilized for that SA.

Copyright © International Network Services 1999

Jim S. Tiller, CISSP, is a network security consultant for Lucent Technologies, Tampa, Florida. He has eight years of experience in network systems consulting, specializing in Internet and communications security.