

Virtual Private Network

Introduction to Virtual Private Network

The world has changed a lot in the last couple of decades. Instead of simply dealing with local or regional concerns, many businesses now have to think about global markets and logistics. Many companies have facilities spread out across the country or around the world, and there is one thing that all of them need : A way to maintain **fast, secure and reliable communications** wherever their offices are.

The leased lines to maintain a wide area network (WAN) provides a company with a way to expand its private network beyond its geographic area. A WAN has obvious advantages over a public network like the Internet when it comes to reliability, performance and security. But maintaining a WAN, particularly when using leased lines, can become quite expensive and often rises in cost as the distance between the offices increases.

As the popularity of the Internet grows, businesses begin to use the Internet as a means of extending their own networks. Intranets with password-protected sites have been designed and introduced within companies for use by employees. Many companies are now creating Virtual Private Network (VPN) to accommodate the needs of remote employees and distant offices.

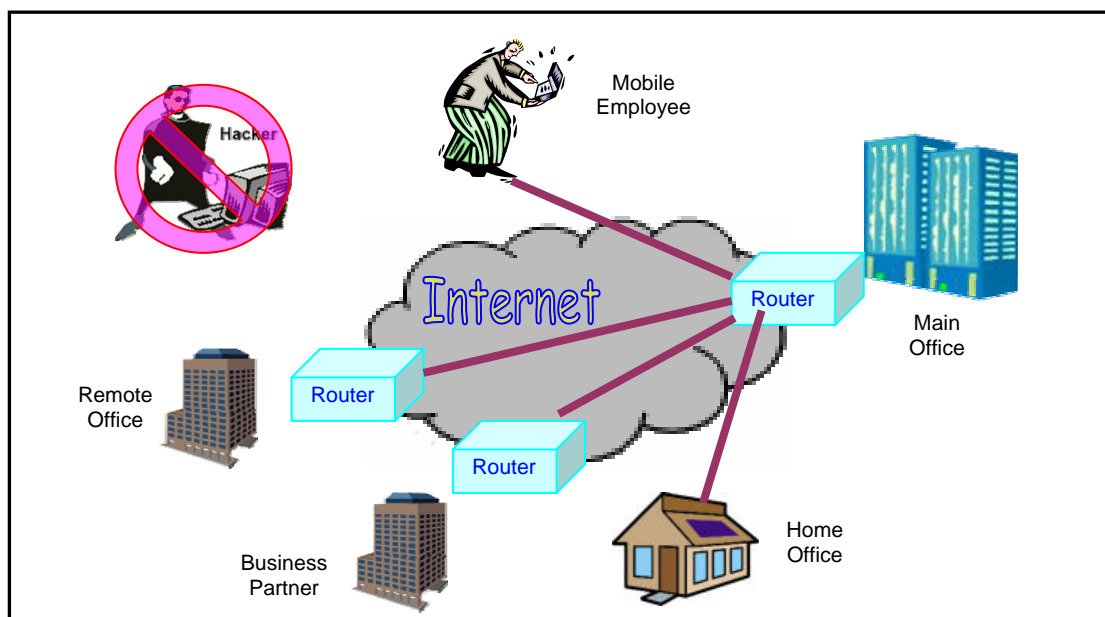


Figure 1. Simple Virtual Private Network

A VPN is a private network that uses a public network (usually the Internet) to connect remote sites or users together. Instead of a real-world dedicated connection such as through a leased line, a VPN uses “virtual” connections routed between the Internet and the company’s private network to reach the remote site or employee.

A Virtual Private Network (VPN) utilizes public networks to conduct private data communications. VPN follows a client and server approach. VPN clients authenticate users, encrypt data and manage sessions with VPN servers utilizing a technique called **tunneling**.

Types of VPNs

There are typically two common types of VPNs: **Virtual Private Dial-Up Network** and **Site-to-Site VPN**. They are illustrated in the following scenarios.

Virtual Private Dial-Up Network supports remote access to an intranet. A user-to-LAN connection is used by a company that has employees who need to connect to the private network from various remote locations. For example, a company with many sales people in the field needs remote-access VPN that permits secure and encrypted connections between a company’s private network and remote users. Employees working at home may use this service to get connected to the company. These remote users need VPN client software to allow them to connect securely.

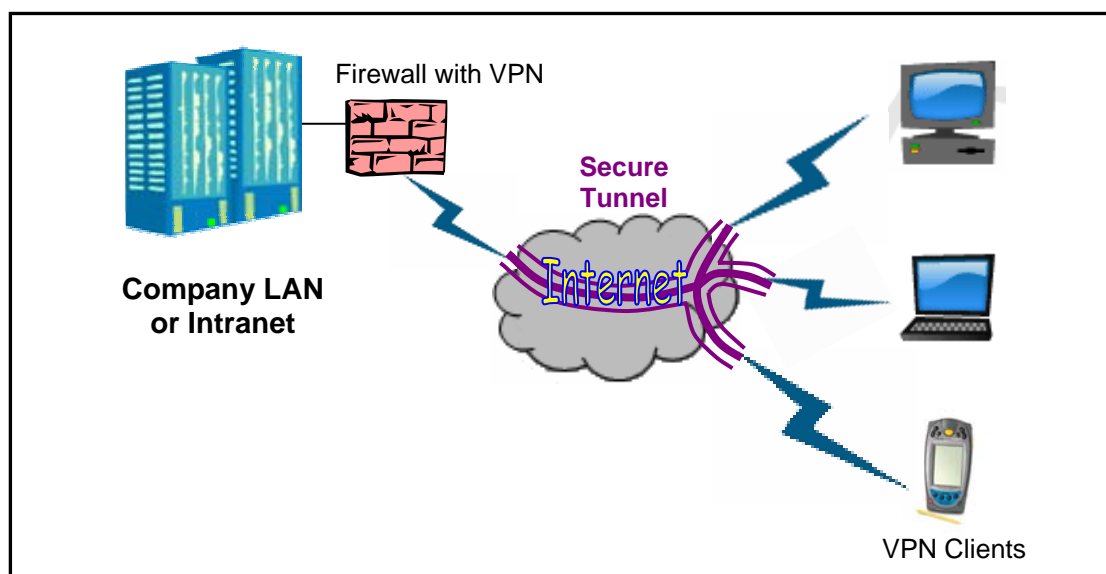


Figure 2. Virtual Private Dial-Up Network

Site-to-Site VPN is implemented with the use of dedicated equipment and large-scale encryption techniques. The company can connect multiple fixed sites over a public network such as the Internet. Site-to-Site VPNs can be one of the following two types :

- **Intranet-Based**

If a company has one or more remote locations that they wish to join in a single private network, they can create an Intranet VPN to connect LAN to LAN.

- **Extranet-Based**

When a company has a close relationship with another company (for example, a partner, supplier or customer), they can build an extranet VPN that connects LAN to LAN, and allow the various companies to work within a shared environment.

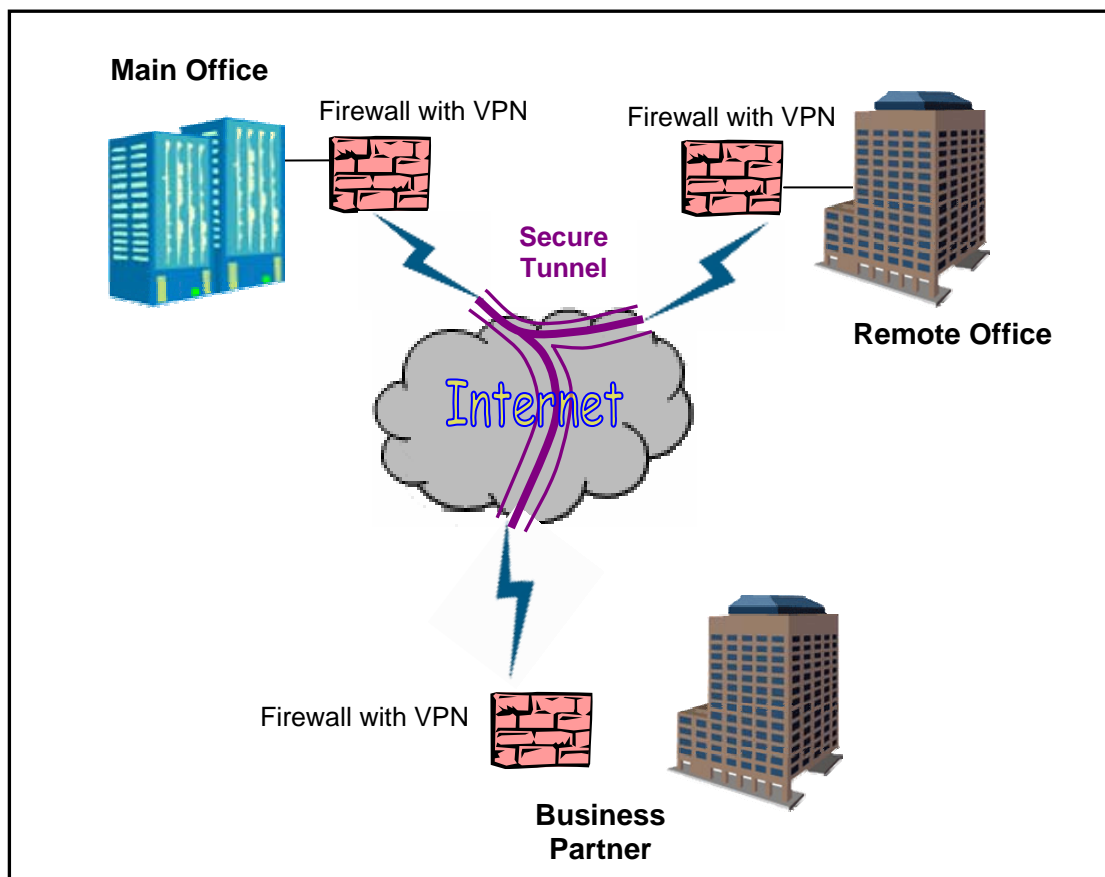


Figure 3. Site-to-Site VPN

Tunnels

A tunnel is a means of forwarding data across a network from one node to another, as if the two nodes were directly connected. This is achieved by encapsulating the data, where an extra header is added to data sent by the transmitting end of the

tunnel. The data is forwarded by intermediate nodes based on this outer header without looking at the contents of the original packet.

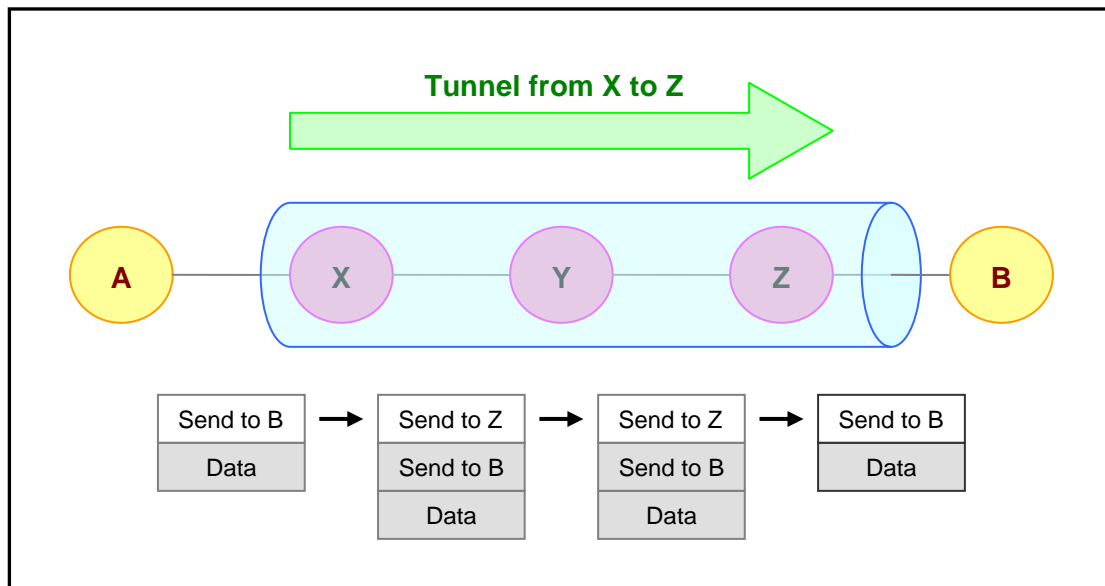


Figure 4. Data Flow in a Tunnel

The above diagram shows data going from A to B being sent through a tunnel between X and Z. The intermediate tunnel node, node Y, does not need to be aware of the final destination, B, but just forwards the data along the tunnel to Z. X is known as the ingress to the tunnel and Z as the egress.

This tunneling of data means that the provider's devices do not need to be aware of the VPNs, but just need to be able to forward tunneled data. This is important as it reduces the network resources consumed by the VPN and the amount of configuration required to set it up.

In addition, by sending data between VPN sites using tunnels, it is possible to maintain separation of data between different VPNs, and to prevent data from a VPN being leaked into the provider network or global Internet. It also means that addresses of devices within the VPN sites are hidden in the data transported over the tunnel, so they do not need to be changed to allow them to communicate over the Internet.

The security of the tunnel is important for maintaining the privacy of VPN data. If the tunnel is not secure, the customer may risk sensitive data being transmitted over VPN.

The main issue with scalability is in the number of tunnels that may be required across the provider backbone, and the amount of network resource that the tunnels consume.

It would be an advantage if the tunneling protocol allows **multiplexing**. This means multiple data streams can be forwarded over the tunnel and then separated at the tunnel endpoint without requiring extra state in intermediate devices. If multiplexing is possible, the provider networks only need to maintain a single mesh of tunnels between the provider end routers, which can be used by all VPNs. However, if multiplexing is not possible, then a separate mesh of tunnels is required for each VPN.

The following diagram shows two data streams being multiplexed through a single tunnel.

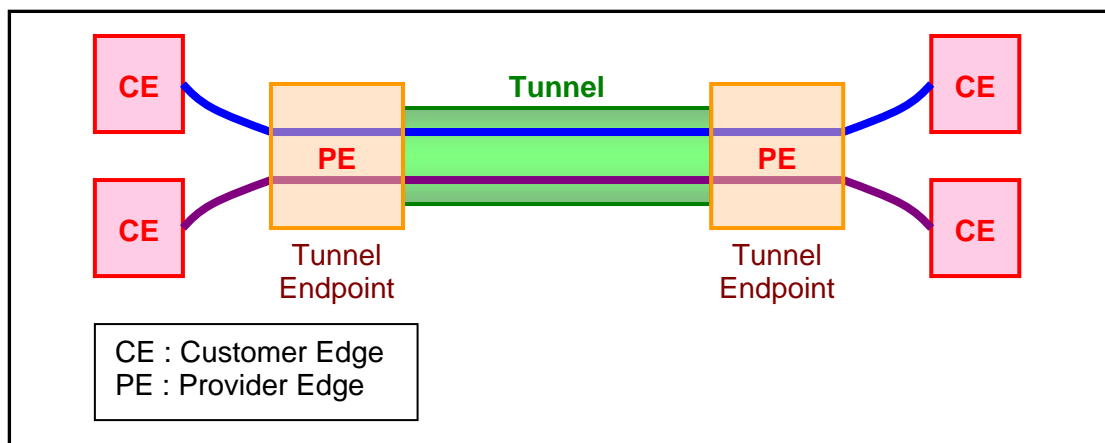


Figure 5. Multiplexing in a Tunnel

The main tunneling protocols used for VPNs are IPSec, PPTP and L2TP.

Encryption and Decryption

The term **encryption** means the changing of the syntax of a message or cleartext, making it unintelligible to the other observer. This altered data is called ciphertext. The term **decryption** is the opposite of encryption. It means changing the ciphertext back to the original cleartext. Encryption and decryption are performed by transforming the cleartext through an algorithm called the cryptographic function into ciphertext. The two inputs into this function are cleartext and a special value called key. The cryptographic function may not be a secret function, but the key must not be

made available to everyone. If someone has this key and the function is known, it is possible to decrypt the ciphertext to reveal the cleartext.

The primary reason of encryption is to keep the data secret. It is the process of encoding a plain text message so that it cannot be understood by intermediate parties. It is used to protect messages so that only the intended recipient who knows the decoding method can read the message. It allows computers to keep secrets by transforming data to a different form using mathematical functions.

There are many encryption methods. **Secret Key Encryption** is one of earliest encryption method. It uses the same key for encoding and decoding the message. It is symmetrical. But it is not secure to transfer key from one party to the other via the electronic medium. Both the sender and recipient must have the same key in order to exchange private messages over an insecure medium. If these two parties want to exchange secret messages over a secure channel, they must decide on a common key. Either party can decide on the key. The main issue of this is that there is no way to send the key to the other party, therefore no way to establish a secure channel.

Data Encryption Standard (DES) is a widely-used method of symmetric-key encryption using a private (secret) key. DES encryption uses a 56-bit key and uses the block cipher method which breaks text into 64-bit blocks and then encrypts them. DES decryption is the inverse of DES encryption and uses the same user key. The DES encryption is performed in three phases.

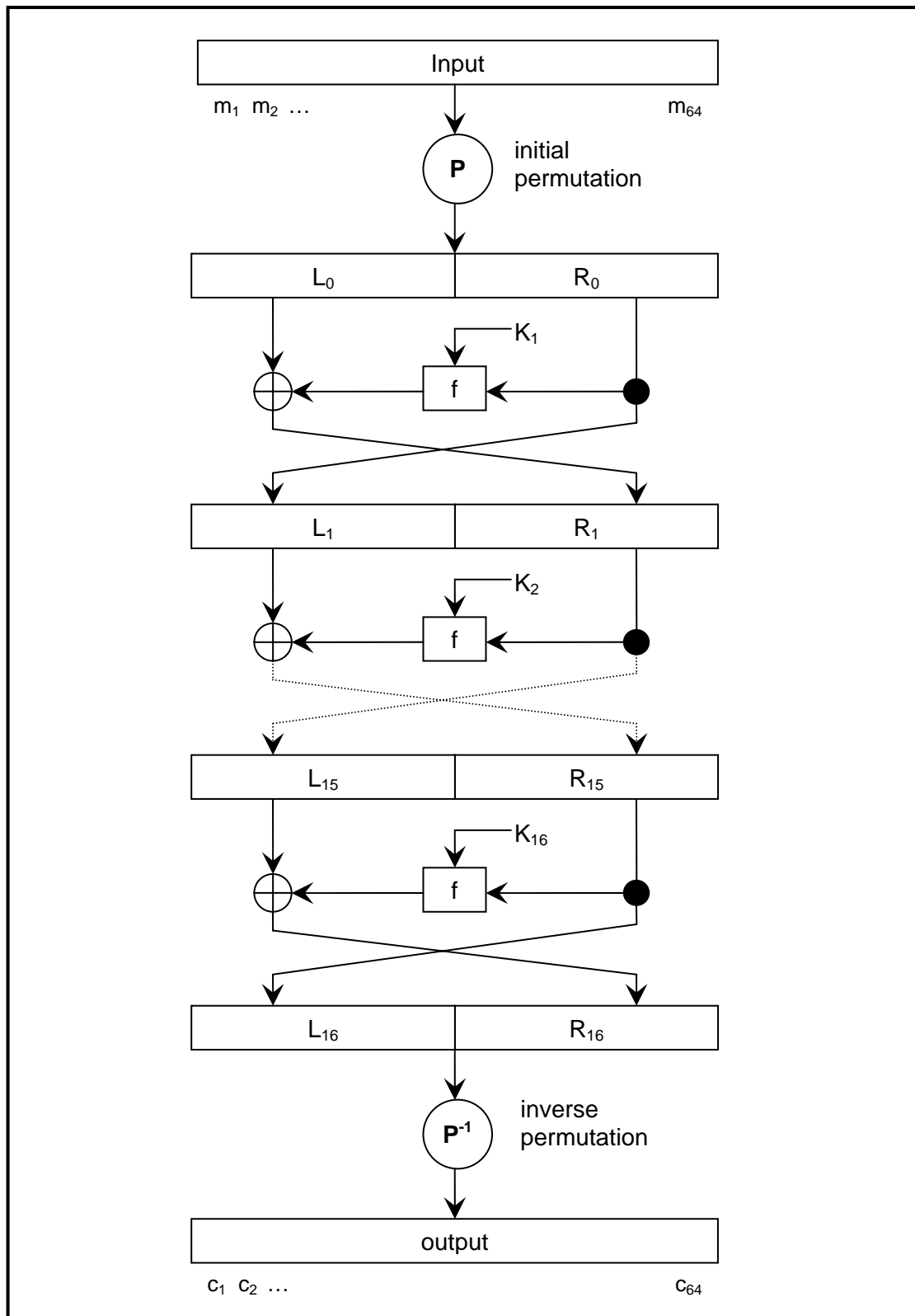


Figure 6. Data Encryption Standard

Phase	Operation
1	The 64-bit plaintext block is bit permuted and stored in two 32-bit registers L (Left) and R (Right).
2	A round operation composed of function f and exclusive-ored (\oplus) is performed 16 times. In the i^{th} round, the 32-bit R and 48-bit round key K_i are inputs to the f function. The output of f function is exclusive-ored with L to form R for the round $i+1$.
3	The two 32-bit outputs of round 16 are concatenated and permuted using the inverse permutation and loaded into the output register.

For round key generation, each of the sixteen rounds uses a 48-bit round key. A round key generation algorithm is used to generate the sixteen round keys, K_1, K_2, \dots, K_{16} from the 56-bit user key.

Triple Data Encryption Standard is a block cipher formed from Data Encryption Standard (DES) cipher. It applies three 56-bit keys, instead of one, for an overall key length of 168 bits. There are several modes of using DES three times. For **DES-EEE mode**, all three keys run in the encryption mode. The first encryption is encrypted with the second key and the resulting cipher text is again encrypted with the third key.

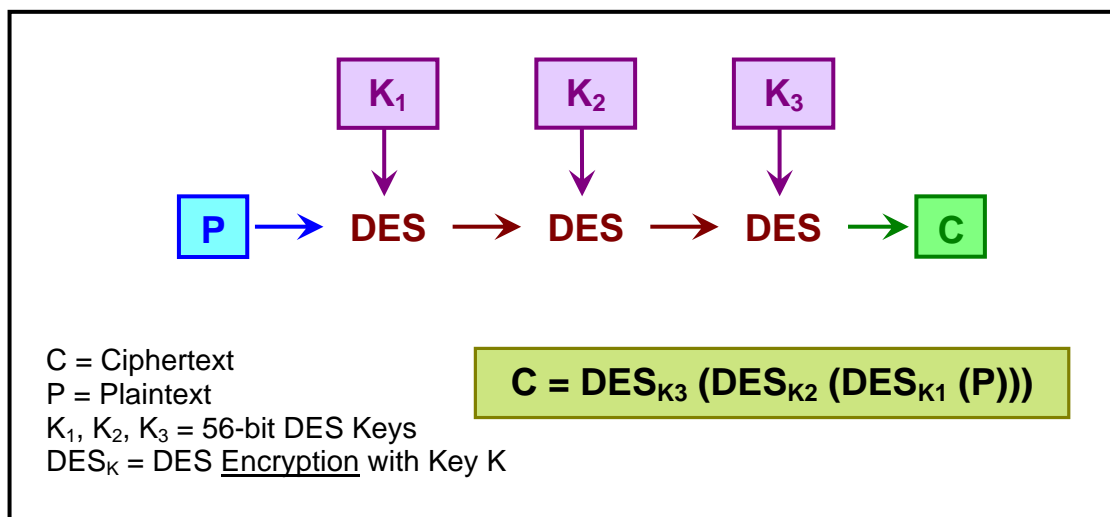


Figure 7. DES-EEE Mode

The other mode is called **DES-EDE mode**. The second stage is run in decryption mode. That means, the first encryption is decrypted with the second key and the result is then encrypted with the third key.

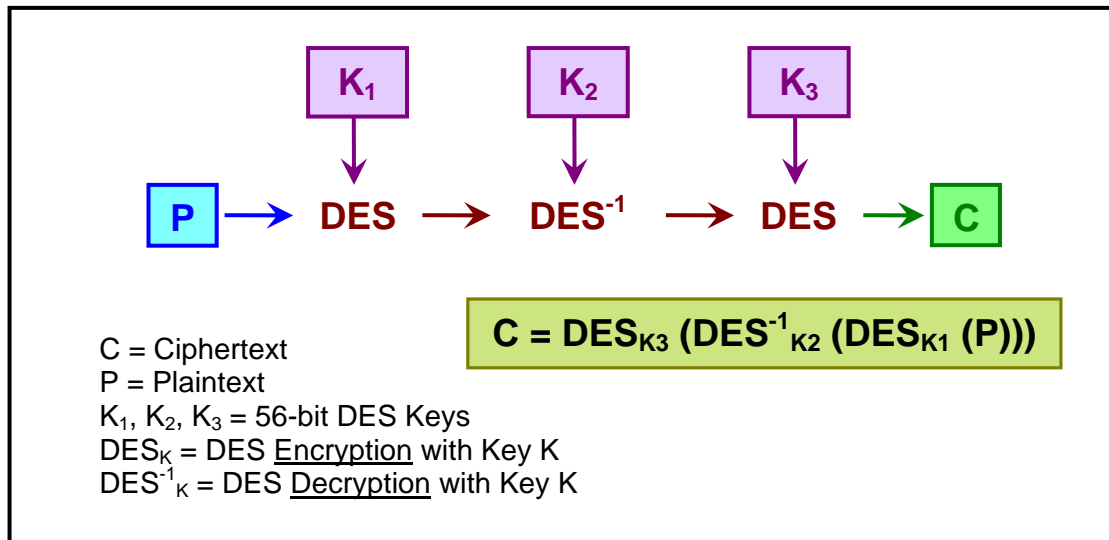


Figure 8. DES-EDE Mode

In some DES-EDE mode, K₁ and K₃ are equal. This means that the effective key size is only 112 bits. This mode is susceptible to some attacks.

When K₁, K₂ and K₃ are equal, Triple DES is reduced to single DES. This is used to provide backward compatibility.

The **Advanced Encryption Standard (AES)** provides a better combination of safety and speed than DES. The AES algorithm converts a block of 128 bits of plaintext to a 128-bit block of encrypted data called cipher text. This algorithm uses one of the three cipher key strengths: a 128-, 192- or 256-bit encryption key. Using 128-bit secret keys, AES offers higher security against brute-force attack than the old 56-bit DES keys. AES can also use larger 192-bit and 256-bit key, if necessary. As a block cipher, AES encrypts data in fixed-size blocks, but each AES cycle encrypts 128-bit, which is twice the size of DES blocks. Each encryption key size causes the algorithm to behave slightly differently. So, the increasing key sizes not only offer a larger number of bits for scrambling the data, but also increase the complexity of the cipher algorithm.

While DES was designed for hardware, AES runs efficiently in a broad range of environment. For examples, programmable gate arrays, smart card, desktop computer software and browsers.

In 2000, the National Institute of Standards and Technology (NIST), a unit of the U.S. Commerce Department, selected the encryption algorithm **Rijndael** as the new AES. Rijndael is fast and compact with a simple mathematical structure. This structure makes it easier to analyze its security during evaluation. This simplicity gives attackers a smaller mathematical realm to study. If a hidden problem lurks somewhere in Rijndael, it can be discovered easily.

The algorithm consists of four stages that make up a round. This is iterated 10 times for a 128-bit length key, 12 times for a 192-bit key and 14 times for a 256-bit key. The first stage is Byte Sub transformation. It is a non-linear byte substitution for each byte of block in a S-box.

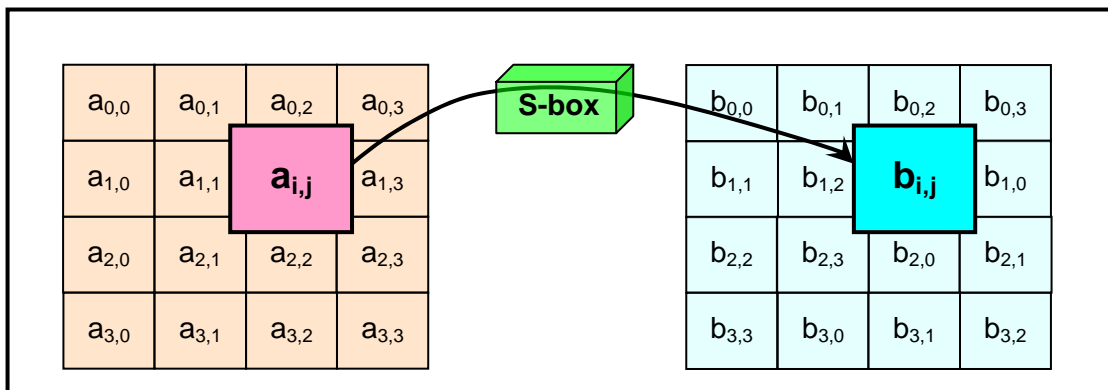


Figure 9. Byte Sub Transformation

The second stage is Shift Row transformation. For 128-bit and 192-bit block, the rows are shifted 1, 2 and 3 places. But for 256-bit block, the rows are shifted 1, 3 and 4 places.

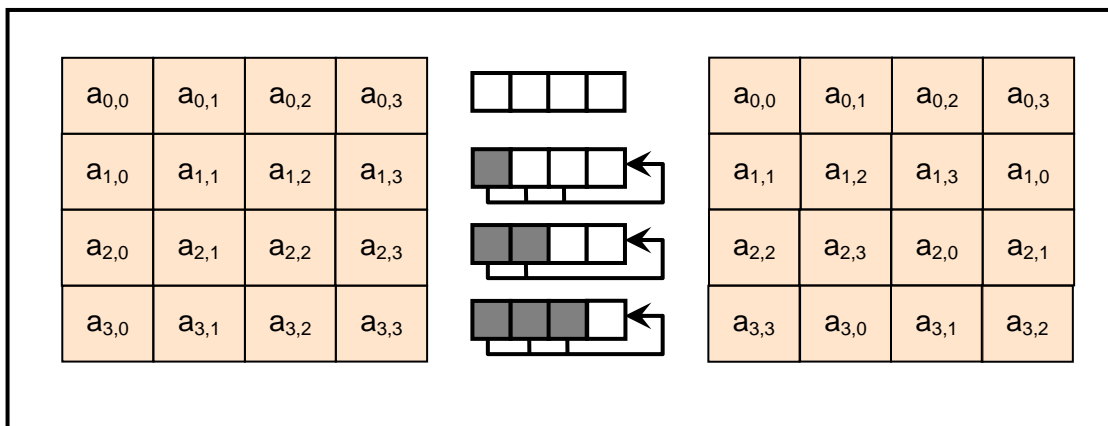


Figure 10. Shift Row Transform (Block that is 128 bits long)

The third stage is Mix Column transformation. Matrix multiplication is performed.

Each column is multiplied by the matrix:

```

2 3 1 1
1 2 3 1
1 1 2 3
3 1 1 2
    
```

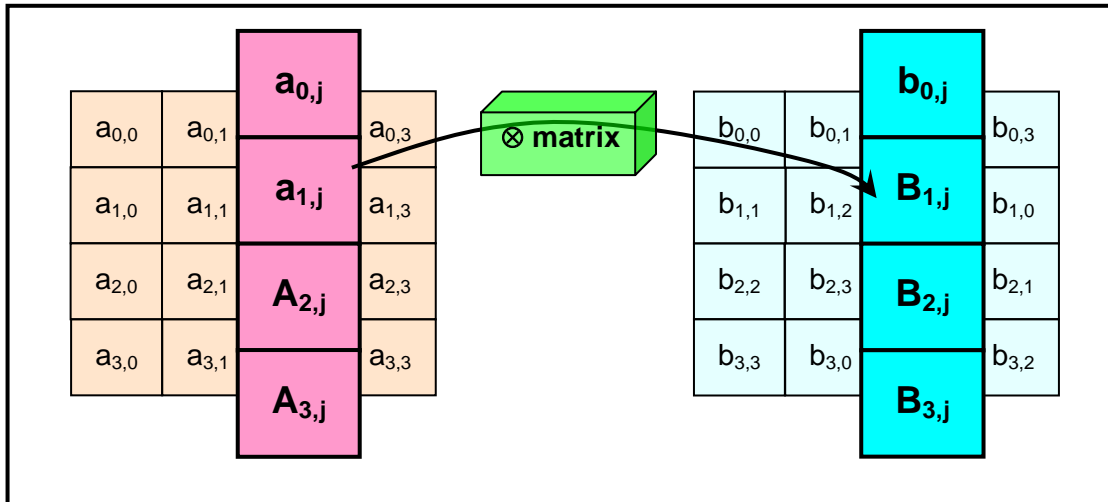


Figure 11. Mix Column Transformation

The fourth stage is the Round Key Addition. The Round Key is applied to the State by XOR. The Round Key is derived from the Cipher Key by means of the key schedule.

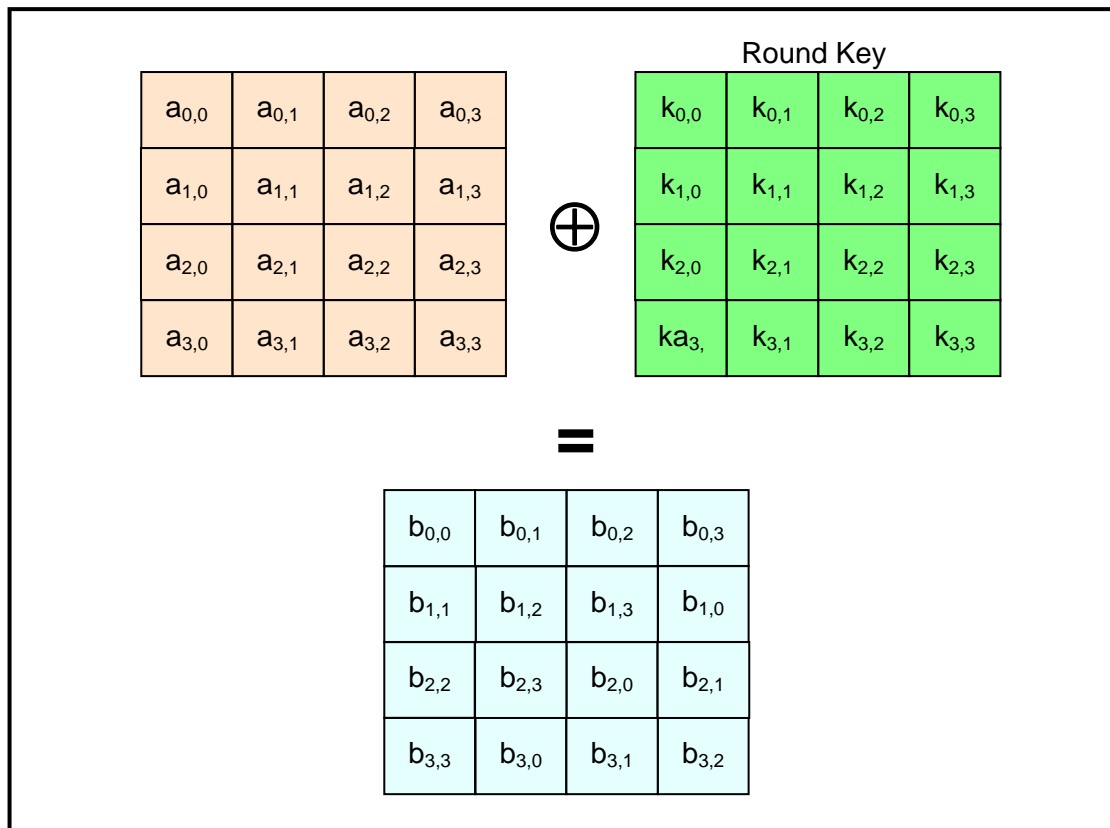


Figure 12. Round Key Addition

Hashes are used to verify that the information is correct and it is based on mathematical algorithms called **One-Way Functions**. These functions are not able to reverse to retrieve the original message.

Public Key Encryption uses a different key to encrypt and decrypt. They are called asymmetric algorithms. In public key encryption system, the encryption key is called the public key as it can be made known to the public. The decryption key is called private key as it must be kept in secret in order to remain secure.

With public key encryption, the receiver can send a public encryption key to the sender. There is no need to keep the public key a secret because it is only used for encoding messages and not decoding them. You can publish your public key to anyone in the world to encrypt message that they are sending to you.

When the receiver gets the message that is encoded by their own public key, they can use their private key to decode the message. No one can decode the private messages with the public key. The encoded message can only be decoded using the secret private key.

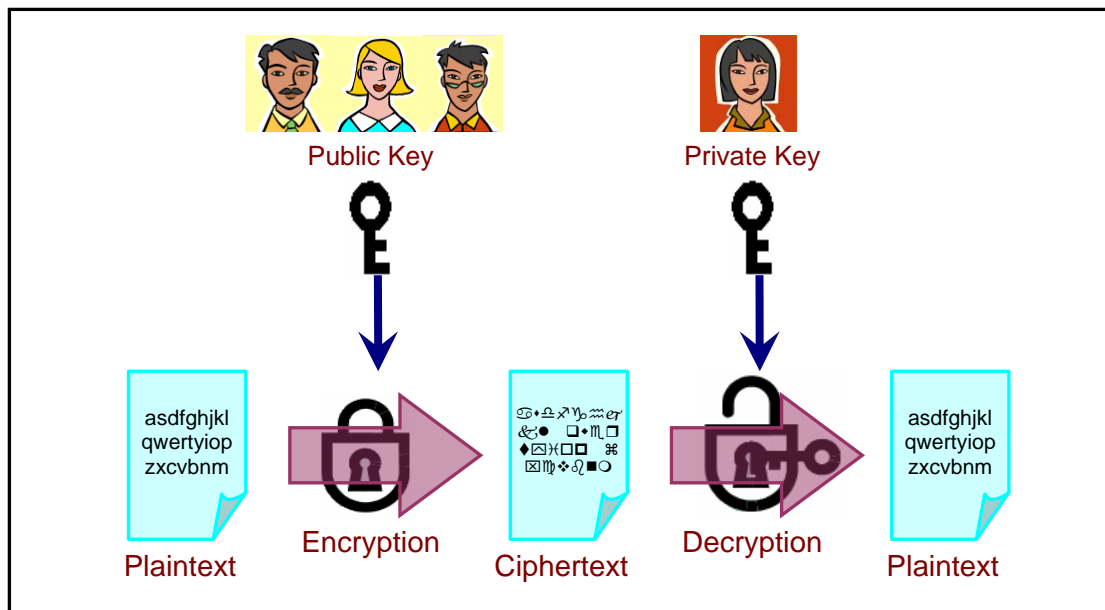


Figure 13. Simple Public Key Encryption

One problem that affects these secure public key ciphers (data that has been encrypted), which are asymmetric, is that they are slow. They are much slower than the symmetric ciphers.

A solution to this is the **Diffie-Hellman (D-H) key agreement**. This 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.

Diffie and Martin Hellman found that some one-way functions could be undone by using a different decryption key that was used for encryption. Their solution takes advantage of a characteristic of prime and prime numbers. It is very hard to find the two factors of a large prime.

Each peer will generate a public and private key pair. The private key generated by each peer is always kept secret and never shared. The public key is calculated from the private key by each peer and it is transmitted and announced over the insecure channel. Each peer combines the other's public key and its own private key to compute a common shared secret number which is then converted to a shared secret key. This shared secret key is never transmitted over the insecure channel.

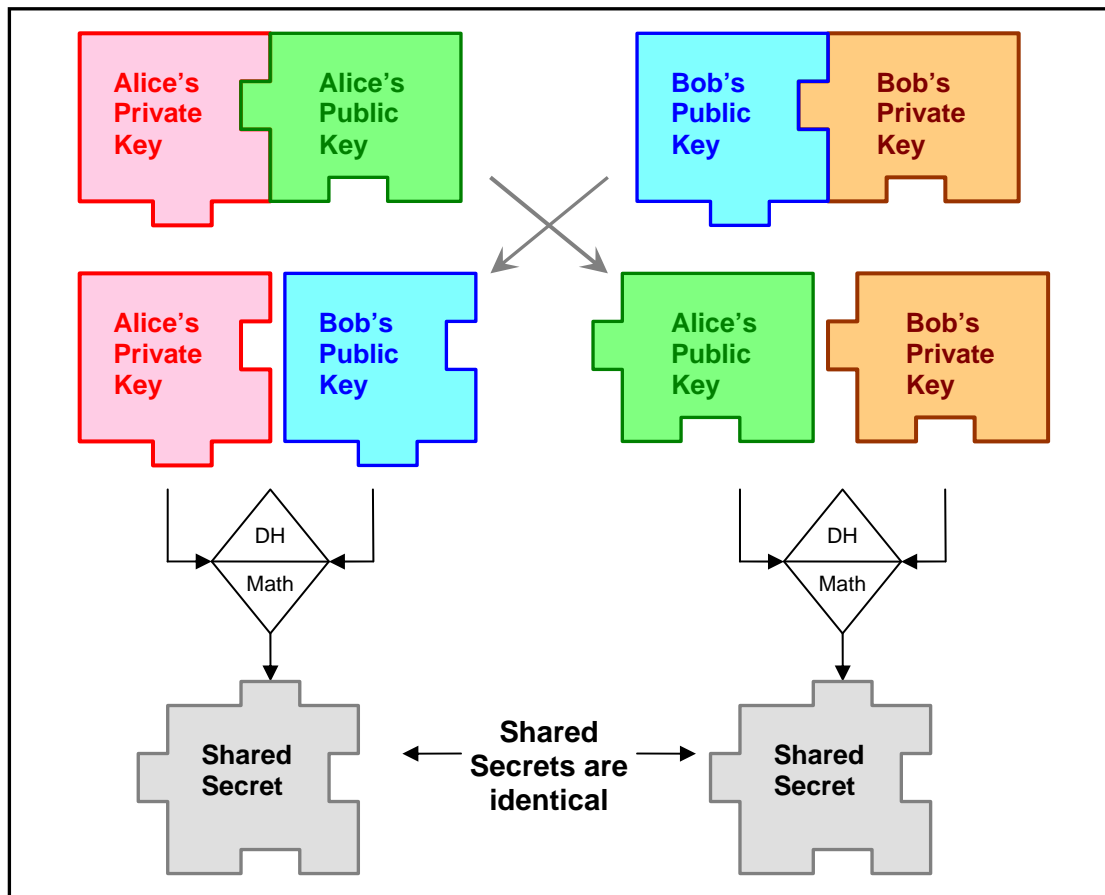


Figure 14. Diffie-Hellman (D-H) Key Agreement

Authentication

Authentication is used to verify the identity of users in order to control access to resources and to prevent unauthorized users from gaining access to the system and record the activities of users in order to hold them accountable for their activities.

Passwords are the oldest and simplest form of authentication. Passwords are secret keys. They are used to prove whether an individual is to be given access to some resource or not. **Password authentication** is implemented by knowing a secret key to prove the user's accessibility to a resource, but it does not prove the user's identity. To prove identity, a password must be unique to a specific person. This is implemented by creating user accounts which are assigned to individuals. The account contains information of who the owner is and provides a unique account name and password. When a user logs on to the system, an input account name and the password that associates with the user account will be required to prove the

user's identity for access to the account. If the password matches the stored password, the user is given access to the system.

Using password-based authentication does not guarantee total security because there is no control on password distribution. Password can be distributed to anyone, therefore, password authentication cannot be used to prove identity, even with unique account name.

Passwords are often easy to guess. Many systems limit password length, which can allow brute-force attack that makes use of every possible combination of values to run sequentially against the password system. The attacks can easily succeed if the password limit is short. This method of attack will not be practical for long password system as it will take ages to crack the password. Attack using brute-force is one of the most common ways hackers gain access to systems.

Replay attack occurs when a secret value, like a hash, is captured and then reused later to gain access to a system using the same secret (hash) value for decoding. Such attack can work against systems that do not uniquely encrypt hashes for each session.

Password hashing is used in order to prevent hackers from capturing the password from your computer's hard disk or while transiting in the network. The passwords should be encrypted using a one-way function or hashing algorithm to keep the password from being revealed. The operating system does not compare your password to the stored password. Instead, it encrypts it using one-way cryptographic function and compares the result to the original result that was stored when the password was created. Since the hashing function is one-way, it cannot be reversed to decrypt your password.

Password hashing, however, can still be cracked using brute-force guessing. The hackers use brute-force to create a list of all possible passwords and by encrypting all of them using the same hashing function and storing the results in a database together with the original text. Then, by capturing your hash, they can look up matching value in their database of hashed values and find the original password.

One-way hashes are great for preventing password lists from being announced publicly, but they cannot be used securely over a network. The hacker may sniff the

network channel between the two computers and intercept the hashed password for replay later to gain access. **Challenge/Response authentication** is able to defeat replay attack by encrypting the hashed password using secret key encryption.

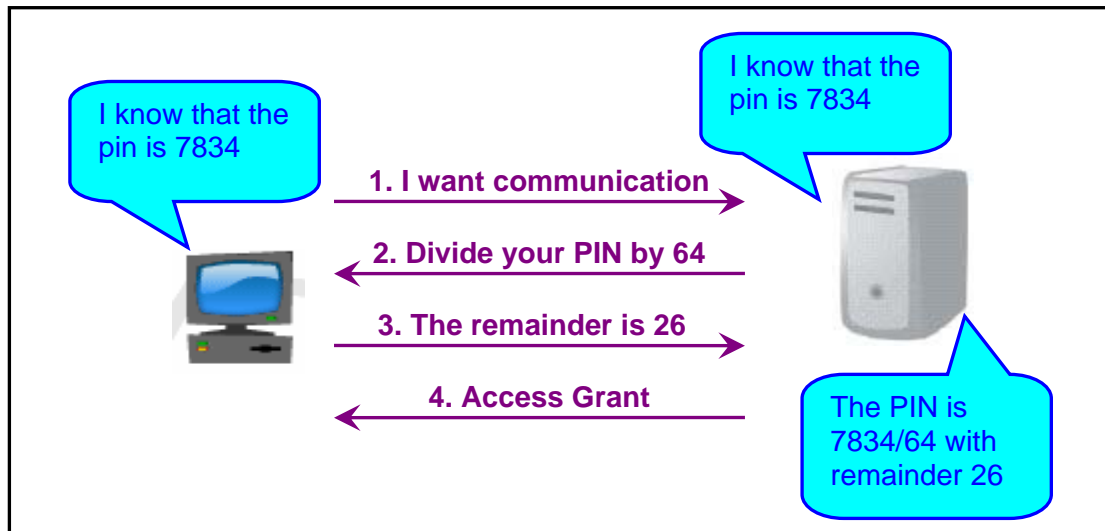


Figure 15. Challenge/Response Authentication

Step	Description
1	The client requests a connection.
2	The server sends a random secret key to the client.
3	The client encrypts the random secret key using a hashed password and transmitted the result to the server.
4	The server decrypts the secret using the stored hashed password and compares it to the original secret key to decide whether to accept the logon.

This system works because the password is never transmitted over the network, even in the hashed form. Only the random number and the encrypted random number are sent. No cryptographic work can be used to decrypt the encrypted random number because all possible results are equal. Challenge and response authentication can also defeat a brute-force indexed hash decryption.

Sessions are used to ensure that, once authentication is done, any further communications between the parties can be trusted. **Session authentication**

occurs at the beginning of the session and it is carried forward throughout the remainder of the packet stream until both parties agree to end the session.

Packets transmitted from one computer to the next are reconstructed into a communication stream by putting them back in their original order using a special number embedded in each packet called a sequence number. The sequence numbers are numbers that indicate which packet in the stream a specific packet represents. Sequence numbers can be simple serial numbers, like 1, 2, 3, 4, etc. but it is too easy to predict. To prevent sequence number prediction, sequence numbers are not sequentially generated. They are generated using a special function called a pseudo-random number generator (PRNG) that can reliably create the same sequence of random numbers called seed number. Seed is the starting point of a specific set of pseudo-random numbers for a specific PRNG.

However, if the same PRNG algorithm generates numbers from the same seed, the exact same series of pseudo-random numbers will be generated every time. Therefore, they are not truly random.

Public key authentication is authentication by means of a **digital signature** which has properties similar to a traditional signature. A digital signature is difficult to forge and is therefore difficult to repudiate, just like the traditional signature. It must also convey message integrity and must be unique. It must prevent additional text from being added to a digitally signed file and prevent a signature from being removed from a signed document.

Instead of encrypting information using someone else's public key, you encrypt it with your own private key. If the information can be decrypted with your public key, then it must have originated with you.

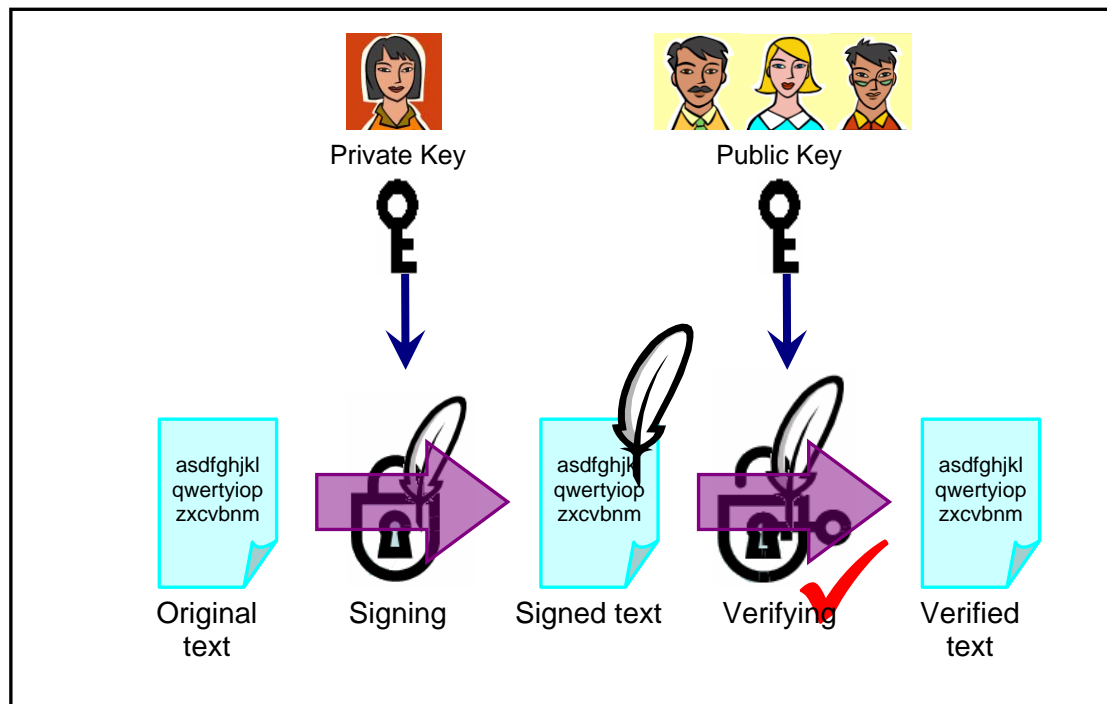


Figure 16. Simple Digital Signature

In summary, digital signature technique has the following properties :

- **Difficult to forge** – only the holder of the private key can generate the signature.
- **Non-repudiable** – a signed document cannot be repudiated later due to extreme difficulty in forging.
- **Unalterable** – once signed, a document cannot be modified.
- **Non-transferable** – the signature cannot be removed and attached to another document

However the simple digital signature technique is slow and it will produce a large volume of data, which is at least double the original data. To improve this, a **one-way hash function** is added into the process. A one-way hash function takes variable-length input and produces a fixed length output. The hash function ensures that, if any part of the information is changed, the output will be different.

A strong hash function is used on the plaintext that the user is signing. This will generate a fixed-length data called the message digest. This digest and the private key are used to create the signature. This signature and the plain text are transmitted together. Upon receiving the message, the recipient will recompute the digest to verify the signature.

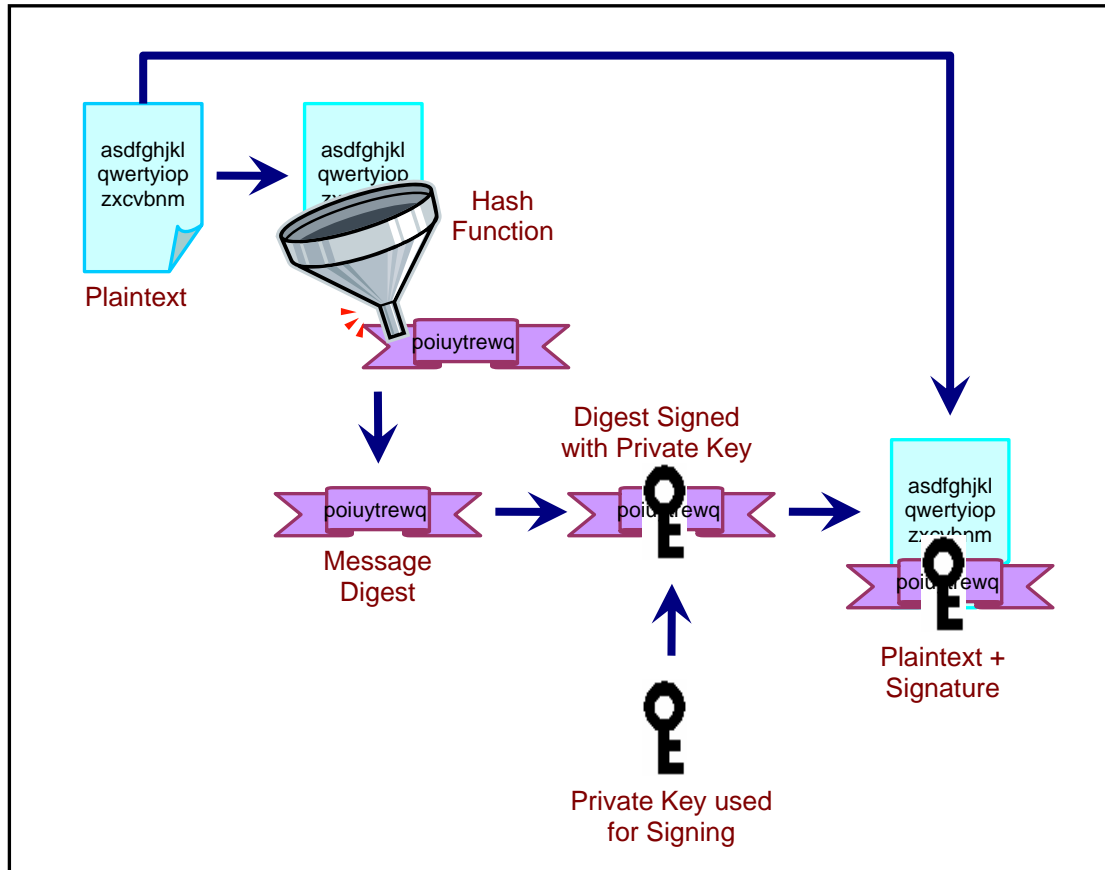


Figure 17. Secure Digital Signature using Hash Functions

In an environment where it is safe to freely exchange keys via public servers, man-in-the-middle attacks are a potential threat. In this attack, the hacker will post a phony key with the name and user ID of the user's intended recipient. In a public key environment, it is important to ensure that the public key being used to encrypt data is in fact the public key of the intended recipient and not a forgery. You can choose to simply encrypt only those keys which are physically handed to you. If you need to exchange information with people whom you have never met, however, you will not be able to tell whether you have the correct key.

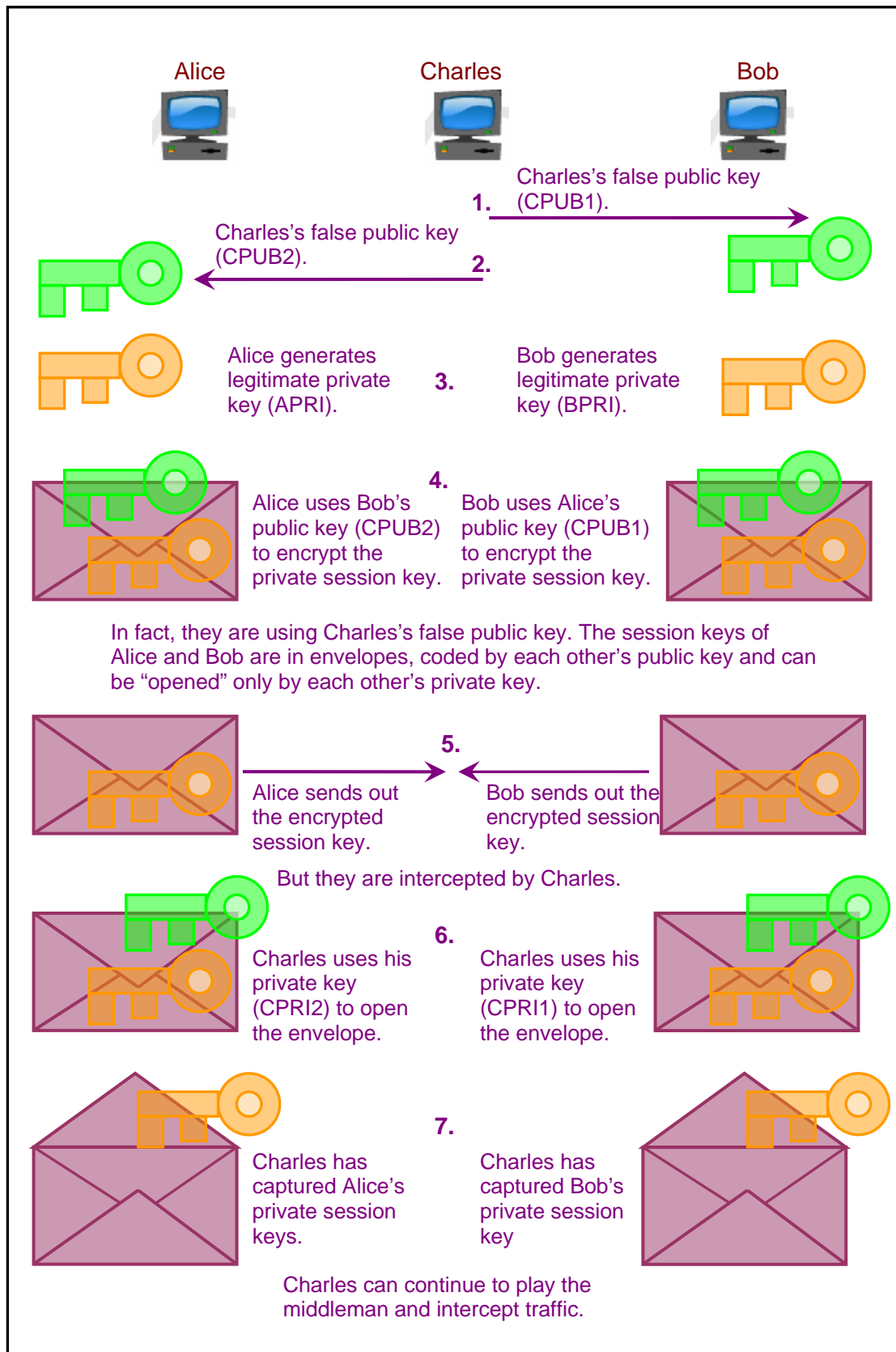


Figure 18. Man-in-the middle attacks

Digital certificates, or **certs**, simplify the task of establishing whether a public key truly belongs to the proper owner. A certificate is a form of credential. It is data that functions like a physical certificate. It has information including a person's public key that will help others to verify that a key is genuine or valid. Digital certificates are used to prevent attempts to substitute one person's key for another.

A digital certificate consists of three things:

- a public key
- certificate information, including identity information about the user such as name, user ID, and so on
- one or more digital signatures

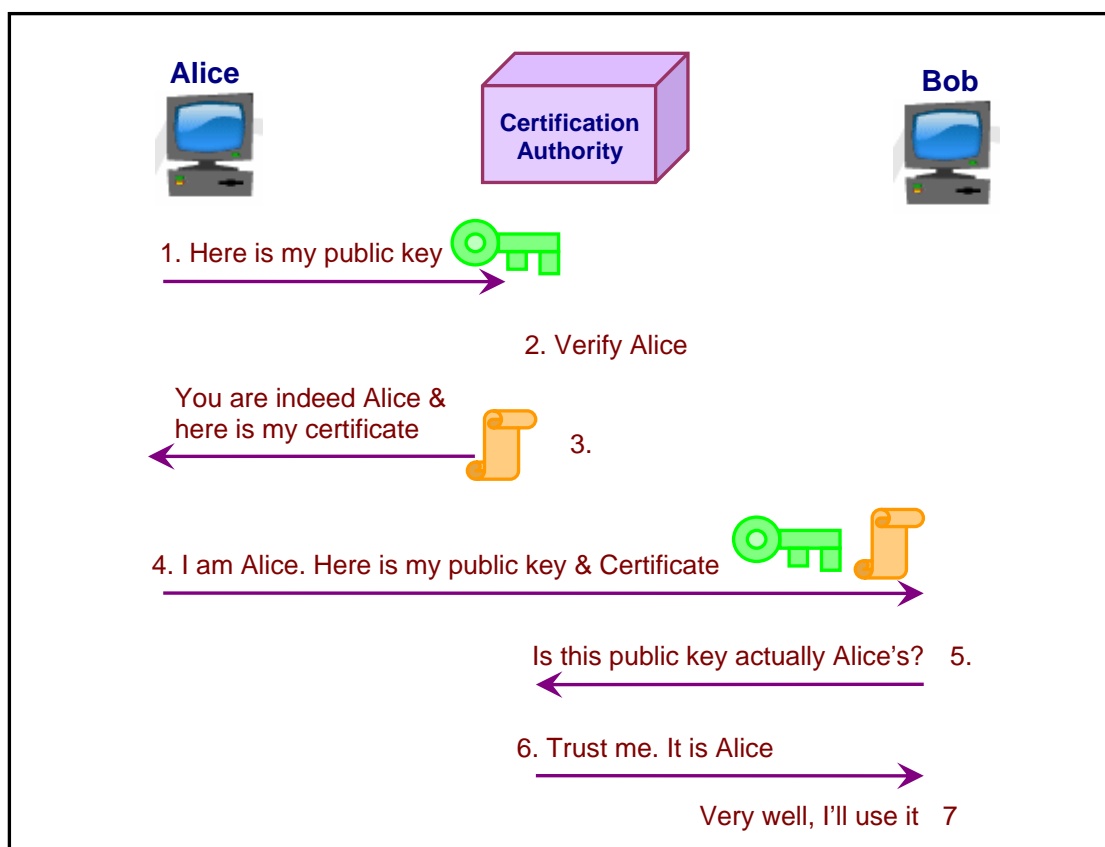


Figure 19. A trusted Certification Authority (CA)

IP Security Protocol (IPSec)

IP packets have no inherent security. Therefore, there is no guarantee that IP datagrams received are

- from the claimed sender
- the original data from the sender
- not inspected by a third party while the packet is being sent from source to destination

IPSec is a method to protect IP datagrams. IPSec protects IP datagrams by defining a method of specifying the traffic to protect, how that traffic is to be protected and to whom the traffic is sent. IPSec can protect packets between hosts, between network security gateways or between hosts and security gateways.

It is a set of protocols developed by IETF (Internet Engineering Task Force) to support secure exchange of packets at the IP layer across physical networks. IPSec has been deployed widely to implement VPNs. IPSec supports two encryption modes: **Transport** and **Tunnel**.

Transport mode encrypts only the data portion (payload) of each packet, but leaves the header untouched. At the receiving side, an IPSec-compliant device will decrypt each packet.

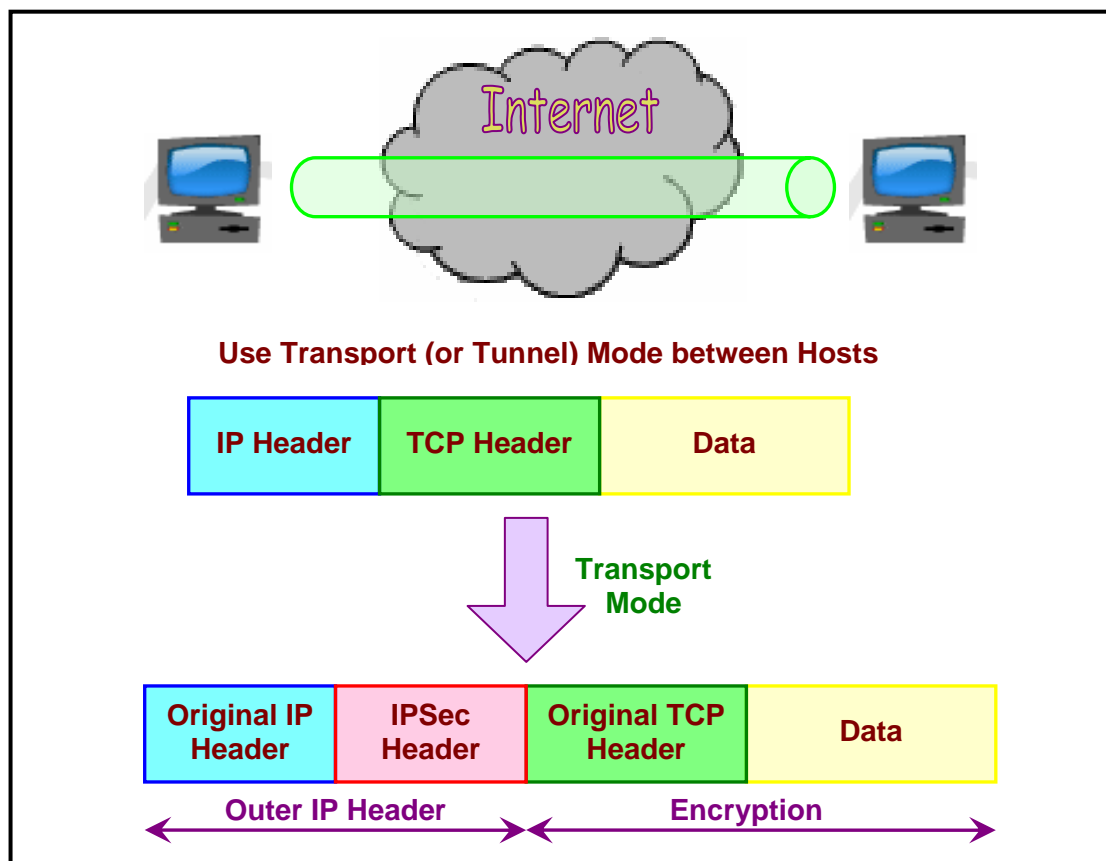


Figure 20. Transport Mode

Tunnel mode encrypts both the header and the payload to provide a more secure exchange of packets. At the receiving side, an IPSec-compliant device will decrypt

each packet. One of the more popular protocol used for building VPNs is the tunnel mode IPsec.

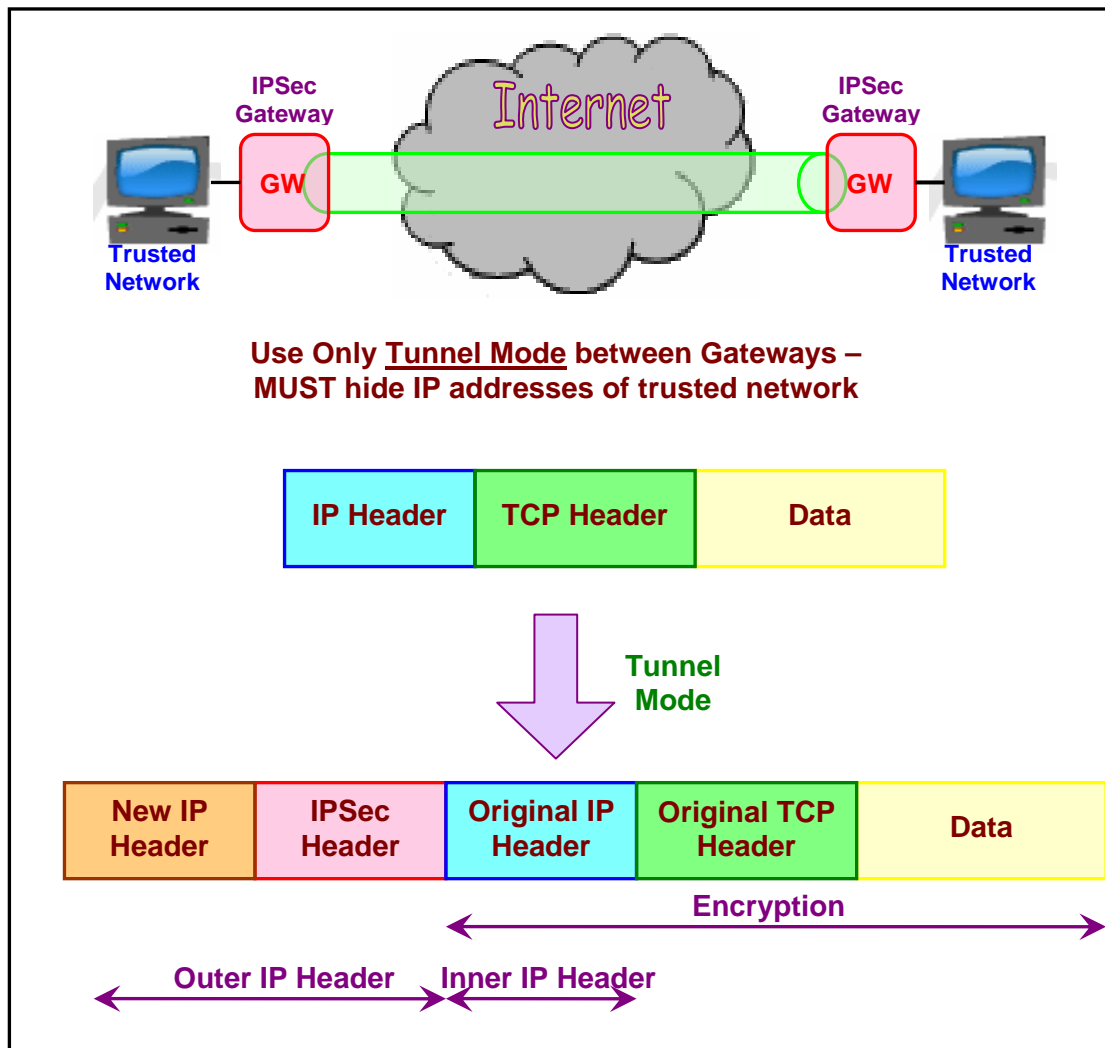


Figure 21. Tunnel Mode

IPsec is developed for security reasons which include connectionless integrity, data origin authentication, anti-replay and encryption. The IETF defines the following functions of IPsec (Reference from R. Atkinson. *Security Architecture for the Internet Protocol*, August 1995. Category : Standards Track.) :

- **Authentication**

The property of knowing that the data received is the same as the data that was sent and that the claimed sender is in fact the actual sender.

- **Integrity**

The property to ensure that data is transmitted from source to destination without undetected alteration.

- **Confidentiality**
The property of communicating such that the intended recipients know what was being sent but unintended parties cannot determine what was sent.

- **Encryption**
A mechanism commonly used to provide confidentiality.

- **Non-repudiation**
The property of a receiver being able to prove that the sender of some data did in fact send the data even though the sender might later desire to deny ever having sent the data.

- **Traffic Analysis**
The analysis of network traffic flow for the purpose of deducting information that is useful to adversary. Examples of such information are frequency of transmission, the identities of the conversing parties, sizes of packets, Flow Identifiers used, etc.

- **SPI**
Acronym for “Security Parameters Index”, it is an unstructured opaque index which is used in conjunction with the Destination Address to identify a particular **Security Association**.

The method of protecting IP datagrams is by using one of the IPSec protocols, **Encapsulating Security Payload (ESP)** or the **Authentication Header (AH)**. AH provides proof to the origin of the received packets, data integrity and anti-replay protection. ESP provides what AH provides plus optional data confidentiality. The ultimate security provided by AH or ESP depends on the cryptographic algorithms applied by them.

The security services that IPSec provides requires shared keys to perform authentication and confidentiality. The key management protocol is **Internet Key Exchange (IKE)**, which is a standard method of dynamically authenticating IPSec peers, negotiating security services and generating shared keys.

Security Association

IPSec is divided into two main protocols, the authentication header (AH) protocol and the encapsulating security payload (ESP) protocol. The fundamental part of the IPSec protocols is the **Security Association (SA)**. An SA contains the state necessary to IPSec processing on an IP packet.

A Security Parameters Index (SPI) and Destination Address uniquely identify a Security Association. A security association must provide the following :

- In an AH packet, the authentication algorithm and algorithm mode being used
- The key or keys used in the above mentioned algorithm
- In an ESP packet, the encryption algorithm, algorithm mode and transform
- The key or keys used of the above algorithm
- Any cryptographic synchronization or initialization vectors being used
- Other pertinent key information such as key lifetime
- The sensitivity level of the packets using a Security Association must be provided under certain circumstances. This sensitivity level is often “unclassified”, “classified”, “secret” or “top-secret” although any identifier may be used.

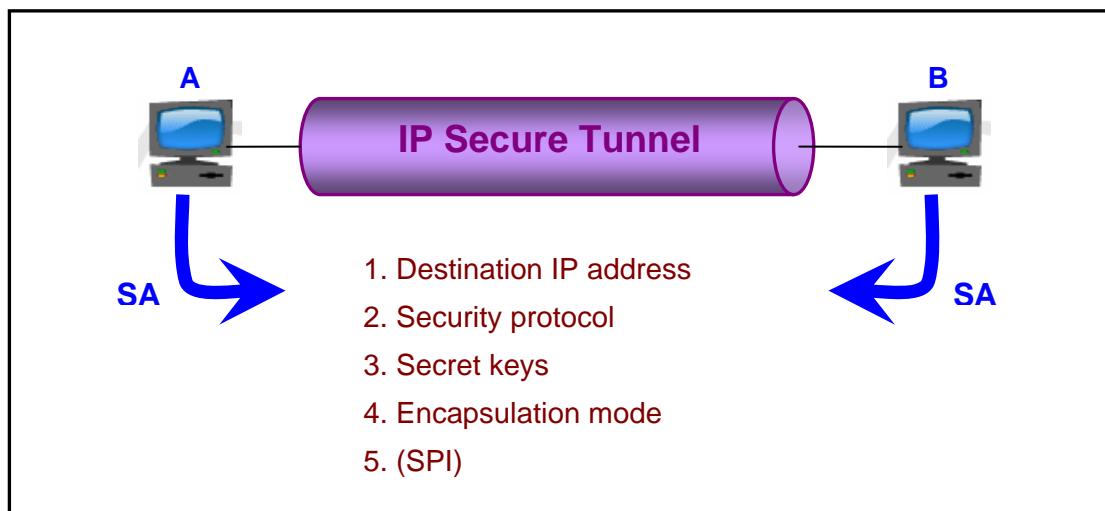


Figure 22. Security Association

Policy

IPSec policy is maintained in the **Security Policy Database (SPD)**. Each entry of the SPD defines the traffic to be protected, how to protect it and with whom the protection is shared. For each packet entering and leaving the IP stack, the SPD must be consulted.

An SPD entry may define one of the three actions :

- **discard** – do not let this packet in or out
- **bypass** – do not apply security services to an outbound packet and do not expect security on an inbound packet
- **protect** – apply security service on the outbound packets and require inbound packets to have security services applied

IP traffic is mapped to IPSec policy by **selectors**. The IPSec selectors are: destination IP address; source IP address; name; upper-layer protocol; source and destination ports; and a data sensitivity level.

SPD entries that define the action “protect” will be pointed to the SA that identify the state used to protect the packet. If an SPD entry is not pointed to any existing SAs in the SA database (SAD), these SAs will have to be created before any traffic may pass. If the rule is applied to inbound traffic and SA does not exist in the SAD, the packets will be dropped. If it is applied to outbound traffic, the SAs can be created dynamically using **Internet Key Exchange (IKE)**.

The IPSec Architecture defines the interaction of the SPD and SAD with the IPSec processing functions such as encapsulate and decapsulate, encrypt and decrypt, integrity protect and integrity verify. It also defines how various IPSec implementations may exist.

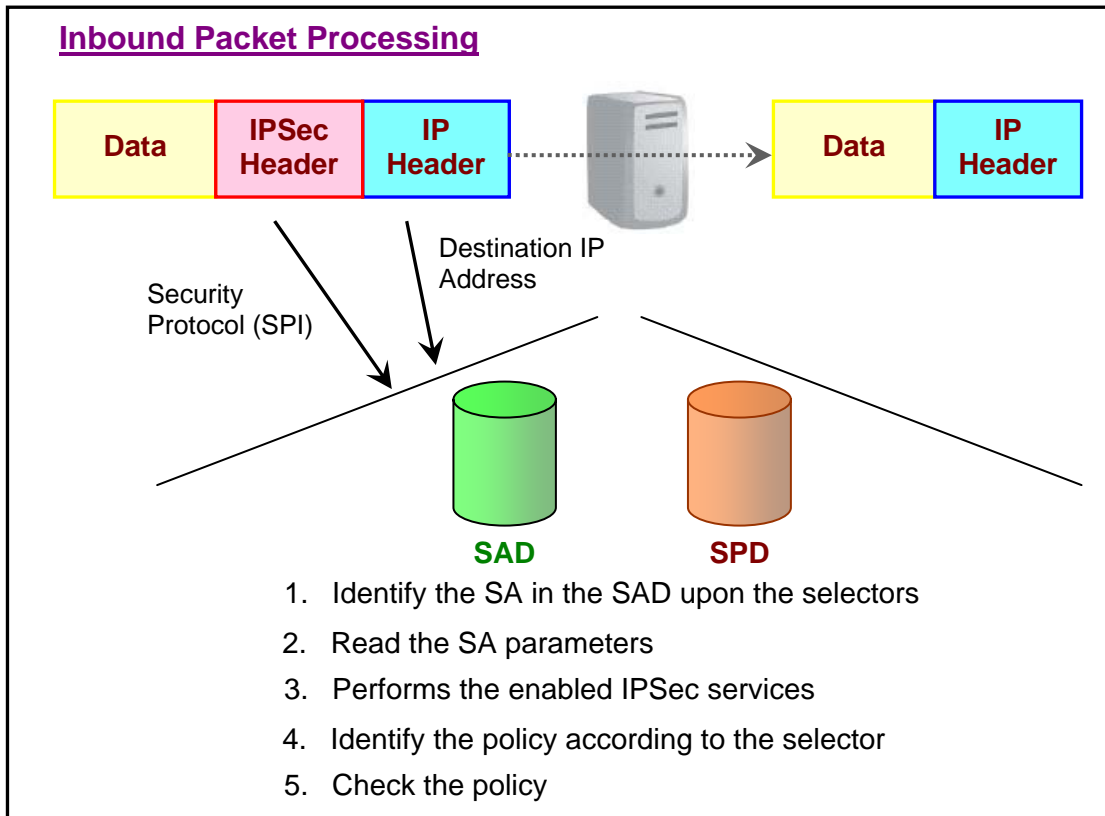


Figure 23. IPSec Policy - Inbound Packet Processing

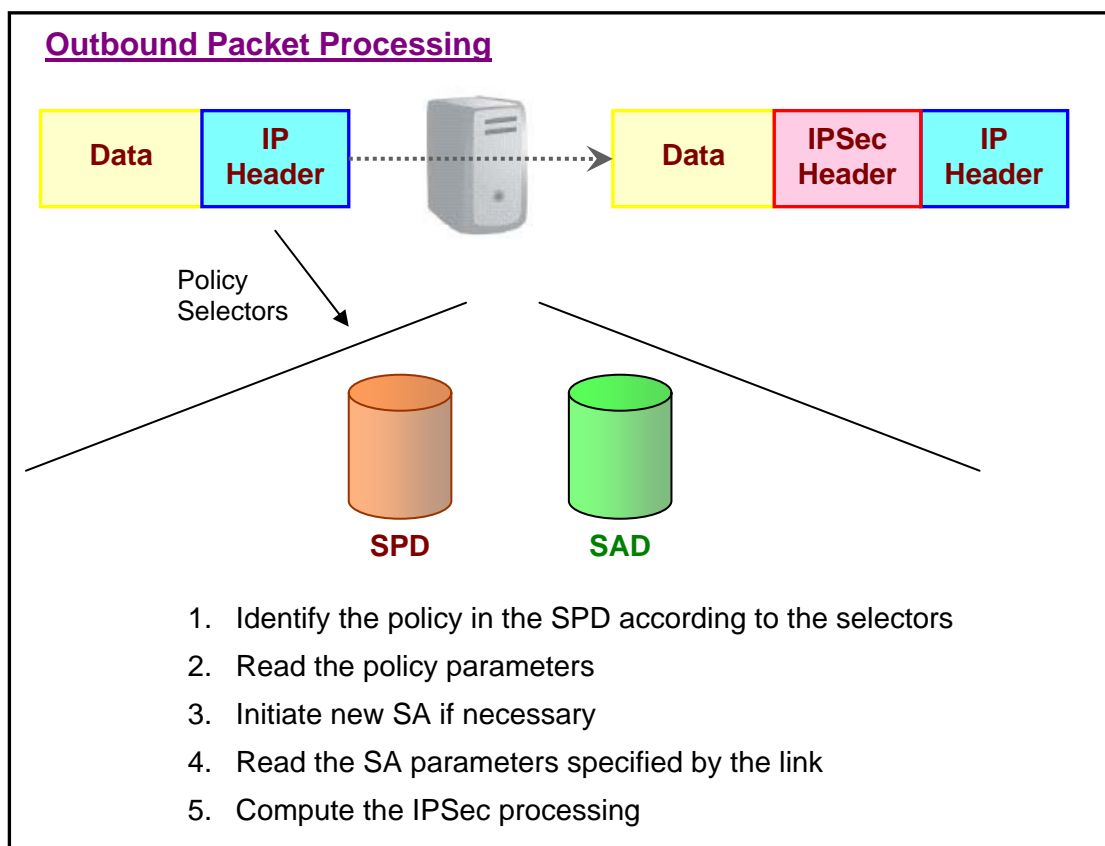


Figure 24. IPSec Policy - Outbound Packet Processing

Authentication Header Protocol

The Authentication Header (AH) is a key protocol in the IPSec architecture. It provides **authentication** and **integrity** to IP datagrams.

It can achieve this by applying a **keyed one-way hash function** to the datagram to create a **message digest**. When the receiver performs the same one-way hash function on the datagram and do a comparison with the value of the message digest the sender supplied, he will detect part of the datagram has been changed during transition if any of the original fields have been modified. In this way, **authenticity** and **integrity** of message can be assured using one secret between two systems by the one-way hash.

AH can also enforce **anti-replay 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, the attacker is able to resend the same packet more than one time.

The AH function is applied to the entire datagram except for any IP header fields that change in transition. AH does not provide encryption and therefore does not provide confidentiality or privacy.

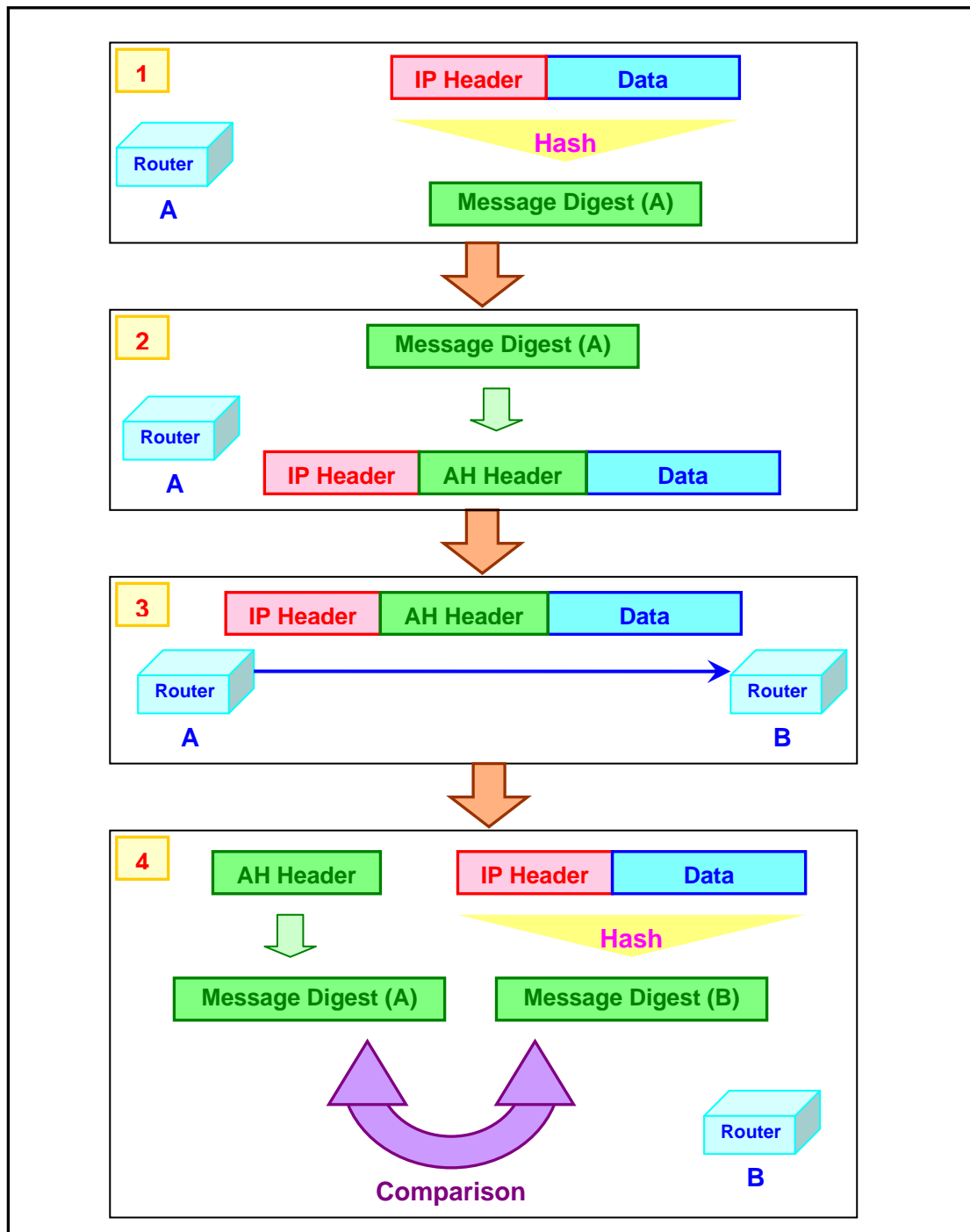


Figure 25. Authentication Header

Step	Action
1	The IP header and data payload are hashed.
2	The hash (message digest) is used to build a new AH header, which is appended to the original packet.

3	The new packet is transmitted to the IPSec router at the destination.
4	The other router at the destination hashes the IP header and the data payload. It then compared with the transmitted hash from the AH header. The two hashes must match exactly.

The Encapsulation Security Payload Protocol

The Encapsulation Security Payload (ESP) is a key protocol in IPSec that provides **integrity** and **confidentiality** (encryption) to IP datagrams. ESP may be used to encrypt an entire IP datagram or just the transport-layer (for example, TCP, UDP, ICMP, IGMP) segment. There are two modes in which ESP can operate: **tunnel-mode** and **transport-mode**.

In **Tunnel-mode** ESP, the original IP datagram is placed in the encrypted portion of the Encapsulating Security Payload and that entire ESP frame is placed within a datagram having unencrypted IP header. The unencrypted portion of the final datagram contains routing information for the IP tunnel. Tunnel mode is most commonly used between gateways or from an end station to a gateway.

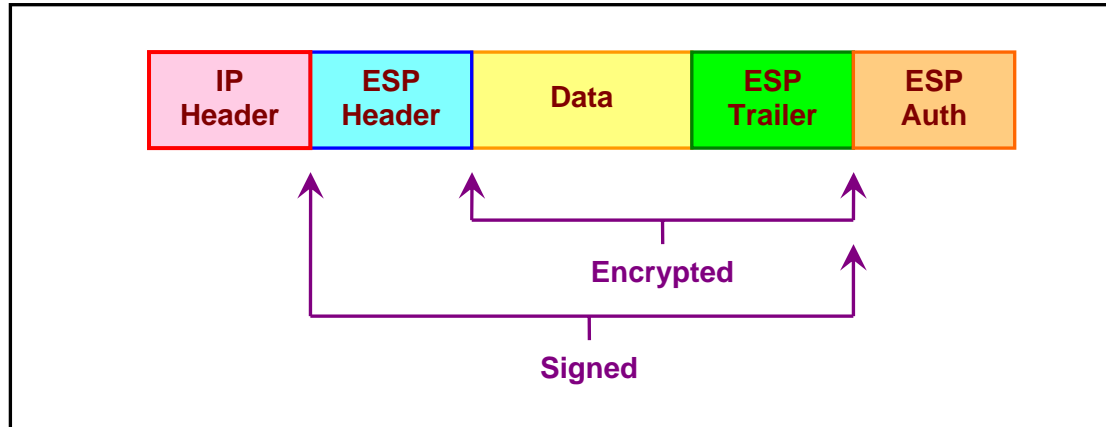


Figure 26. Encapsulation Security Payload Protocol

Transport-mode ESP encrypts the transport-layer protocol and inserts an ESP header immediately before the encrypted protocol header. New IP datagram is not generated and this mode conserves the bandwidth. Transport-mode is used between end stations or between an end station and a gateway if the gateway is treated as a host (for example, Telnet from a computer to a gateway).

Key Management Protocols

All IPSec key management protocols are based on the **Internet Security Association and Key Management Protocol (ISAKMP)**. ISAKMP creates and manages Security Associations. This process requires that the IPSec system first authenticate themselves to each other and establish ISAKMP shared key.

The **Oakley Key Exchange Protocol** is less generic than ISAKMP but offers considerable security with little overhead. It uses Diffie-Hellman key exchange algorithm to facilitate exchange of cryptographic secrets.

Internet Key Exchange (IKE) is a combination of ISAKMP and the Oakley Key Exchange Protocol.

IKE uses two “phases” of negotiation. The **phase one** sets up an authenticated secure association between two ISAKMP servers. This is called **IKE Security Association**. Within phase one are two modes of operations: **Main Mode** and **Aggressive Mode**. This assures the validity and privacy of negotiated Security Associations. Once an association is established between the ISAKMP servers, additional Security Associations are created and distributed. The **Diffie-Hellman key agreement** is always performed in this phase.

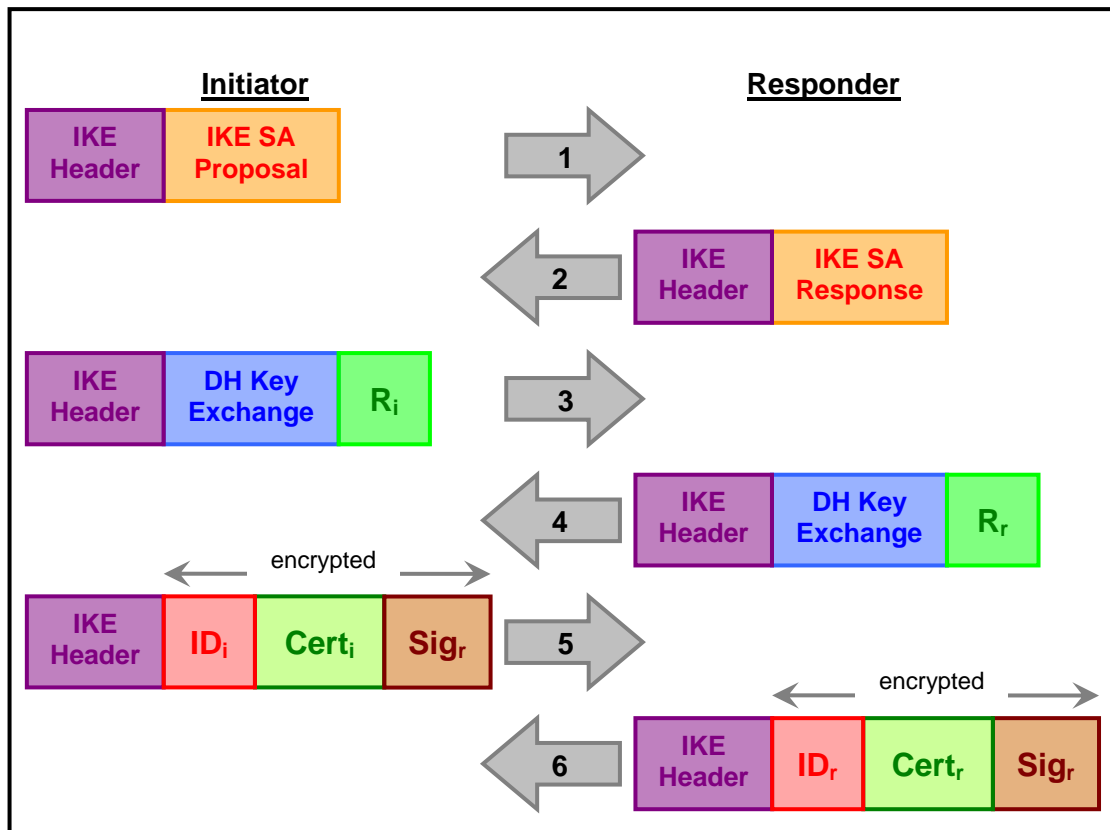


Figure 27. IKE Phase 1 – Main Mode

IKE Main Mode consists of **six messages** that must be exchanged between the initiator and the responder in order to establish an **IKE Security Association**. The IKE protocol uses UDP port 500.

1. The initiator sends an IKE SA Proposal listing all supported authentication methods, Diffie-Hellman groups, a choice of encryption and hash algorithms and the desired SA lifetime.
2. The responder answers with an IKE SA Response indicating the preferred authentication method, Diffie-Hellman group, encryption and hash algorithm and acceptable SA lifetime.

If the two parties were able to successfully negotiate a common set of methods, the protocol is continued by establishing an encrypted communication channel using the Diffie-Hellman Key-Exchange algorithm.

3. The initiator sends his part of the Diffie-Hellman secret plus a random value.
4. The responder does the same by sending his part of the Diffie-Hellman secret plus a random value.

The Diffie-Hellman Key-Exchange can now be completed by both parties forming the common shared secret. This shared secret is used to generate a symmetric session key with which the remaining messages of the IKE protocol are going to be encrypted.

5. The initiator sends his identity optionally followed by a certificate linking the identity to a public key. This is followed by a hash over all message fields signed by a pre-shared secret or a private RSA key.

6. The same as item 5 above, but formed and sent by the responder.

If the identity of both peers is successfully authenticated, then an IKE SA has been established.

In **phase two**, IKE negotiates the **IPSec security associations** and generates the required key material for IPSec. Within phase two are two modes of operations: **Quick Mode** and **New Group Mode**. 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 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.

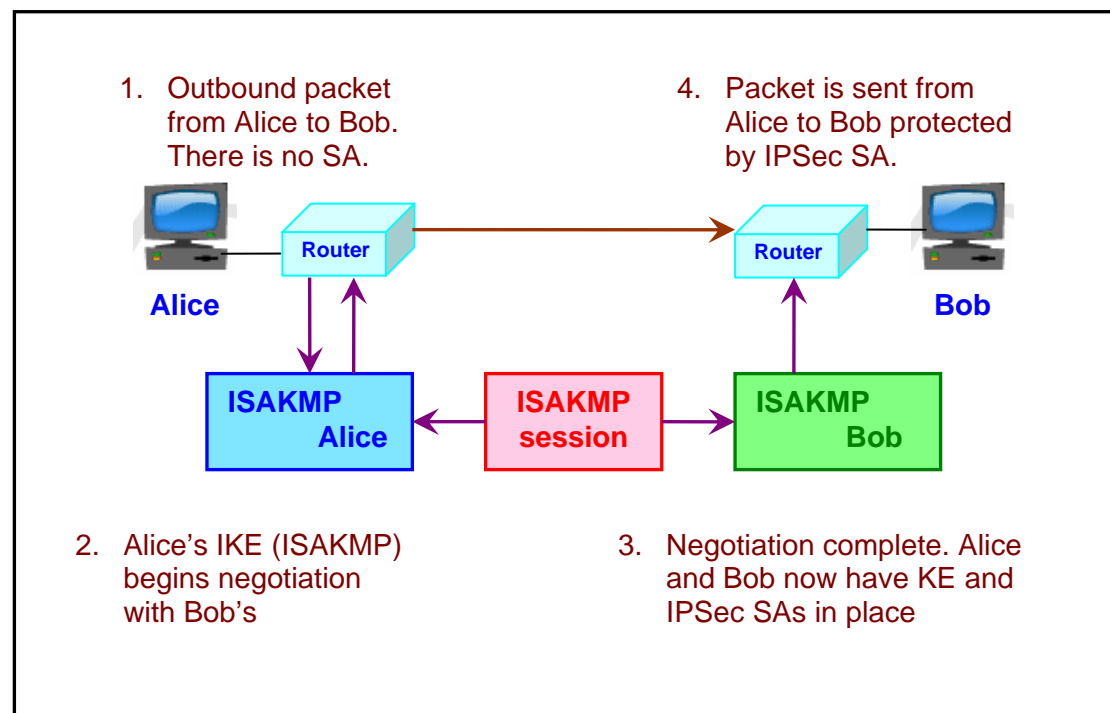


Figure 28. Internet Key Exchange

IPSec

IPSec involves many component technologies and encryption methods. Yet IPSec's operation can be broken down into five main steps.

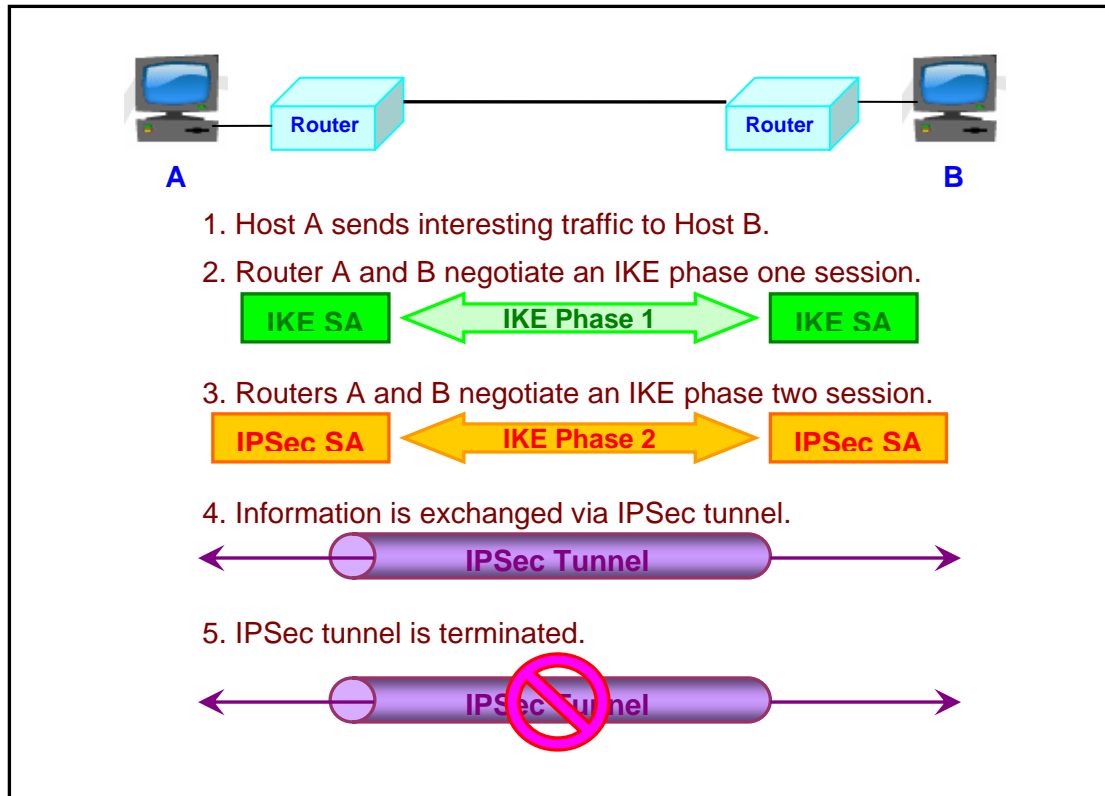


Figure 29. IPSec Operation Steps

The five steps are summarized as follows:

Step	Action	Description
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.
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
3	IKE Phase Two	IKE negotiates IPSec SA parameters and sets up matching IPSec SAs in the peers.

4	Data Transfer	Data is transferred between IPSec peers based on the IPSec parameters and keys stored in the SA database.
5	IPSec Tunnel Termination	IPSec SAs terminate through deletion or by timing out.

Point-to-Point Tunneling Protocol (PPTP)

The Point-to-Point Tunneling Protocol (PPTP) specification was developed by the PPTP forum, collaboration between Microsoft and a group of several leading manufacturers. PPTP can be regarded as an extension of the remote access **Point-to-Point Protocol (PPP)** defined in the document by the Internet Engineering Task Force (IETF). PPTP is a network protocol that encapsulates PPP packets into IP datagrams for transmission over the Internet or other public TCP/IP-based networks. PPTP can also be used in private LAN-to-LAN networking.

PPTP protocol is included with Windows NT Server 4.0 and Windows NT Workstation 4.0 operating systems (and later version). Computers running these operating systems can use the PPTP protocol to securely connect to a private network as a remote access client by using a public data network such as the Internet. PPTP can also be used by the computers connected to a LAN to create a virtual private network across the LAN.

PPTP enables the implementation of secure multi-protocol VPNs through public data networks such as the Internet. It enables remote users to access corporate networks securely across Microsoft workstations and other PPP-enabled systems to dial into the Internet instead of company's private network at low cost. The company does not need to build and maintain a relatively costly Wide Area Network through private lines.

The user can dial and connect to the Internet service provider. The PPTP session will provide a secure connection through the Internet back to the corporate network.

Technically, PPP packets travel end-to-end through the PPTP channel. But from the point of the user, the tunnel is transparent.

Generally, there are three computers involved in every PPTP deployment: PPTP Client; Network Access Server (sometimes referred to as front-end processors – FEP, dial-in servers or point-of-presence – POP server); PPTP Server. You do not need a Network Access Server to create a PPTP tunnel using a PPTP client to connect to a LAN with PPTP Server.

A PPP client sets up a session to an ISP's PPTP-enabled Front End Processor (FEP), typically an Internet router or bridge. When the client requests connection to the RAS Server, the FEP will establish the VPN by starting a PPTP session and tunneling all information from the PPP client through the PPTP channel. The Windows NT server handles all validation and manages data encryption. The PPTP session is transparent to the user and the client only requires PPP to operate. The RAS Server can accept secure PPTP connections from a wide range of operating systems.

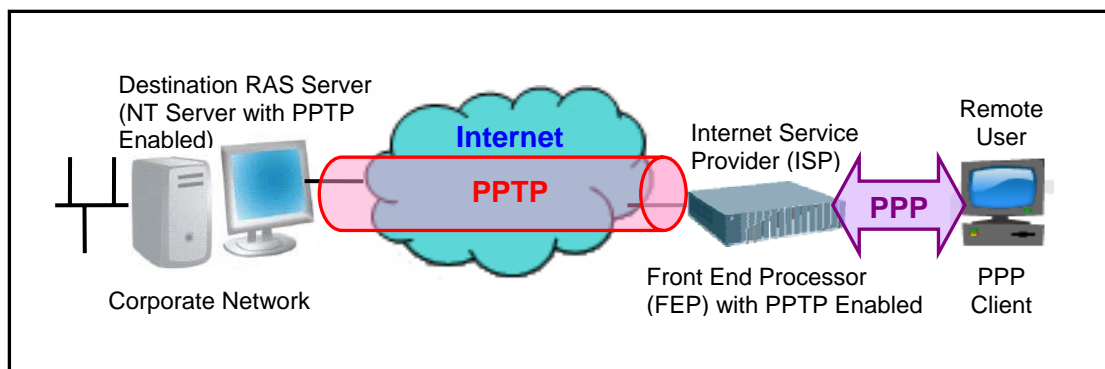


Figure 30. PPTP session example with PPP Client

Another scenario shows that the remote user has PPTP protocol installed. This client which is a Windows NT workstation use Dial-Up Networking and remote access protocol PPP to connect to an ISP. It establishes a PPP session. The same client will then dial the second time, concurrent to the PPP session, and set up the PPTP channel and contact the remote Windows NT Server RAS Server. The PPP packets are then tunneled through the new virtual connection and the client is now a virtual node on the corporate LAN. It will be like a local user which is physically located across the Internet. The Front End Processor does not need to have PPTP enabled. However, only PPTP-enabled clients, such as Windows NT, 2000 and XP workstation, can utilize this configuration. The NT Server will handle all validation and manage data encryption in both directions.

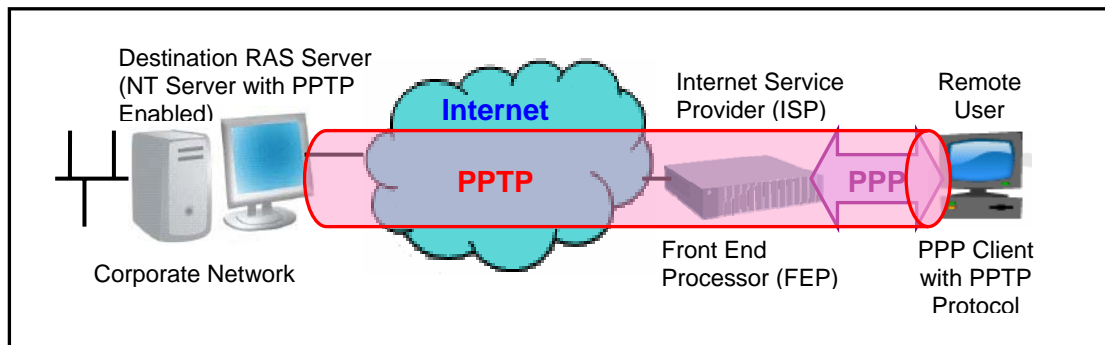


Figure 31. PPTP session example with PPP Client and PPTP Protocol

Authentication of users is done using the existing authentication protocols in Windows NT Remote Access Service – PAP and CHAP. PPTP makes use of the security provided through PPP. MS-CHAP (PPP authentication) is used to validate the user credentials against Windows NT domain and the resulting session key is used to encrypt user data.

There are two basic types of packets in PPTP protocol: data and control packets. The first resides in a data portion of the packet, which can vary in length. The second is found in fixed-length packet header. Data packets contain the normal user data and application commands. Control packets send periodic inquiries on status and manage signals between the PPTP-enabled client or FEP and the destination server, along with embedded management information which contains basic device management and configuration information. Data packets are encapsulated inside an IP envelope, in a method known as tunneling. Control packets, on the other hand, communicate through a TCP connection, one per pairing of an NT server and PPTP client or FEP.

PPTP Tunnel

Tunneling is a technique in which a datagram is contained within the envelope of another higher level protocol (IP). PPTP utilizes only IP, while IPX or NetBEUI is encapsulated in IP packet during data transfer. From the Internet, remote network cannot access the services from a private network. However, through tunneling, remote network can access resources through a PPTP server that is connected to both the Internet and private network. Both the PPTP client and PPTP server use tunneling to securely route packets to a computer on the private network by using routers that only know the address of the private network intermediary server.

When the PPTP server receives the packet from the routing network, it sends it across the private network to the destination computer. The PPTP server does this by processing the PPTP packet to obtain the private network computer name or address information in the encapsulated PPP packet. This encapsulated PPP packet contain multi-protocol data such as TCP/IP, IPX or NetBEUI.

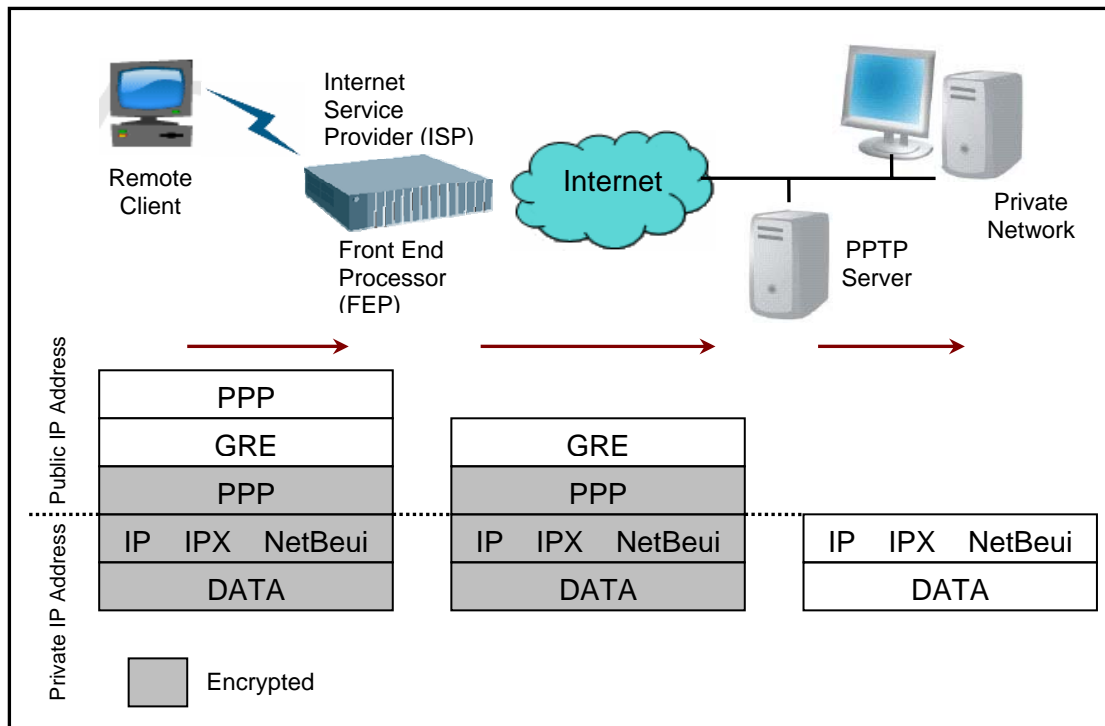


Figure 32. PPTP Operation

PPTP encapsulates the encrypted and compressed PPP packets into IP datagrams for transmission over the Internet. These IP datagrams are routed over the Internet until they reach the PPTP server that is connected to the Internet and the private network. The PPTP server disassembles the IP datagram into a PPP packet and then decrypts the PPP packet using the network protocol of the private network. The network protocols on the private network that are supported by PPTP are IPX, NetBEUI or TCP/IP.

Layer 2 Tunneling Protocol (L2TP)

Layer 2 Tunnel Protocol, L2TP, allows extension of user PPP sessions from an **L2TP Access Concentrator (LAC)** to a remote **L2TP Network Server (LNS)**. The main function of L2TP is encapsulation of PPP frames in UDP packets which in turn are encapsulated in IP datagrams so that a PPP session is extended over a wide area

network. L2TP is based on two earlier protocols, the Layer 2 Forwarding (L2F) and PPTP, proposed by Cisco and Microsoft respectively. L2TP cannot only run over IP but over other protocols such as Frame Relay and ATM.

L2TP Access Concentrator (LAC) is a node that acts as one side of an L2TP tunnel endpoint and is a peer to the L2TP Network Server (LNS). The LAC sits between an LNS and a remote system and forwards packets to and from each. It relays the traffic between the LNS and the user. It is the network access server (NAS) at the ISP with the L2TP client. Packets sent from the LAC to the LNS require tunneling with the L2TP protocol. The connection from the LAC to the remote system is either local or a PPP link.

L2TP Network Server (LNS) is a node that acts as one side of an L2TP tunnel endpoint and is a peer to the L2TP Access Concentrator (LAC). The LNS is the logical termination point of a PPP session that is being tunneled from the remote system by the LAC and handles the server side of the L2TP protocol.

Both LAC and LNS must have access to the Internet. Both are called the L2TP endpoint. The L2TP tunnel exists between a LAC-LNS pair. This tunnel carries encapsulated PPP datagrams and control messages between the LAC and the LNS.

In the **Compulsory Tunneling Mode**, PPP frames from the remote PC are tunneled transparently to the corporate LAN. This means that the remote client has no control of the tunnel, and it will appear as if it is connected directly to the corporate network through a PPP connection. The L2TP software will add an L2TP header to each PPP frame that is to be tunneled. This header is used in the other end of the tunnel, where L2TP packets are demultiplexed.

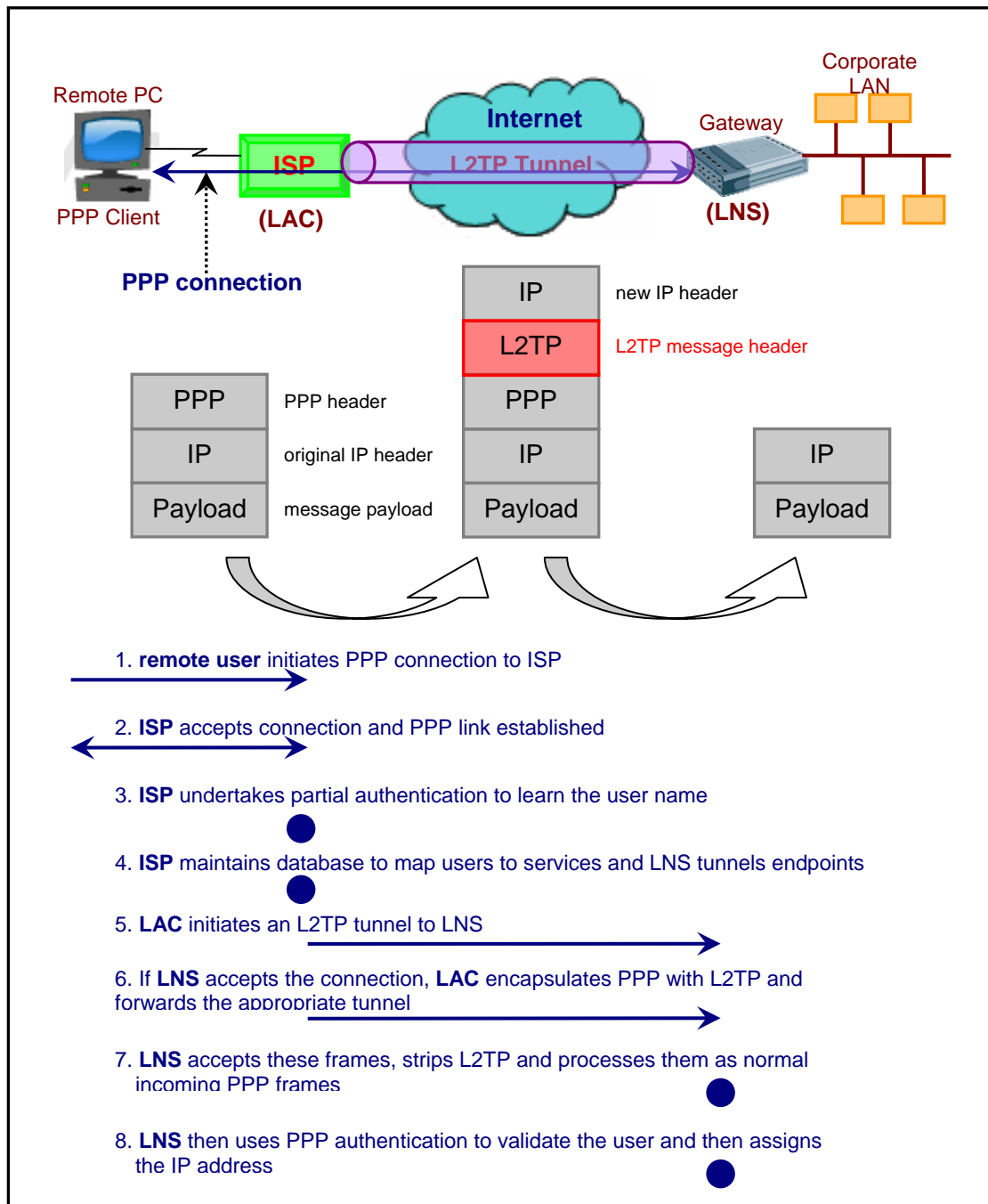


Figure 33. L2TP Compulsory Tunnel Mode

The **Voluntary Tunneling Mode** has the remote client having the built-in LAC functions and it is able to control the tunnel. Since the L2TP protocol operates exactly the same way as when using compulsory tunneling, the LNS will see no difference between the two modes.

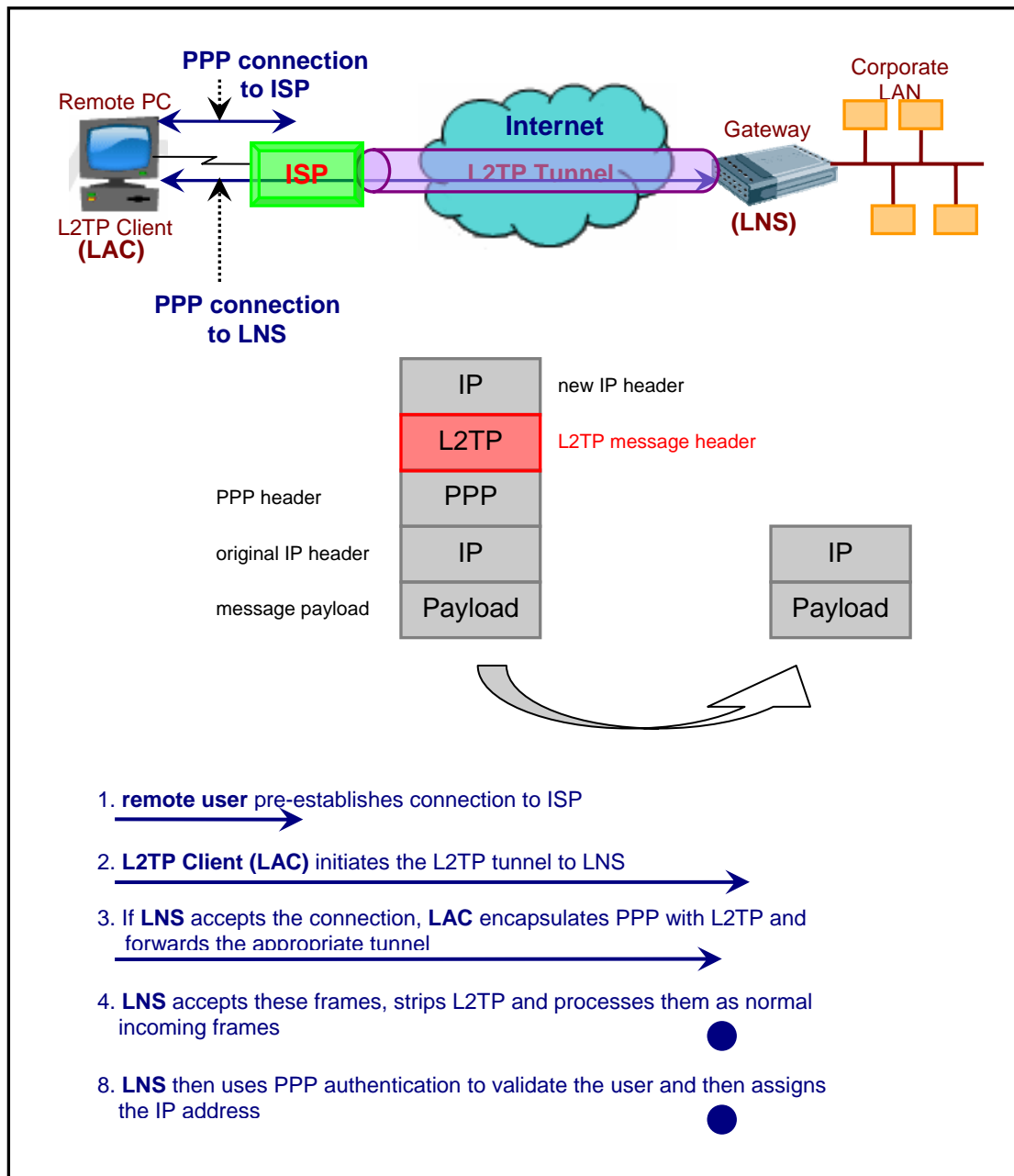


Figure 34. L2TP Voluntary Tunnel Mode

Summary for Virtual Private Network

Virtual Private Networks are convenient, but they can still create security holes to your network. These are some tips that you can practice to help you to avoid trouble. The best way to ensure that you have comprehensive security is to combine security functions on a single machine. **Firewalls** make ideal VPN endpoints because they can route translated packets between private systems. Real firewalls are most likely to be equipped with secure encryption and authentication methods.

No VPN solution is effectively secure if the operating system of the machine is not secure. A **strong filtering function** is implemented on the server with VPN endpoint. Without a **secure base operating system**, the VPN can be easily hacked to gain access to the network from anywhere. You should always use packet filtering to reject connection attempts from every computer except those that you specifically set up to connect to your network remotely. When creating a LAN-to-LAN VPN, you need to cross-filter on the foreign server's IP address. When providing VPN access to remote users whose IP address changes dynamically, you need to filter on the network address of the ISP's dial-up domain.

Using a **single ISP** to connect all the hosts acting as tunnel endpoints will increase both speed and security of your tunnel. This is because the ISPs will keep most of the traffic on their own network and expose lesser to the Internet.

Public key authentication is considered more secure than the simple, shared secret authentication. Select VPN solutions that use **strong public key encryption** to perform authentication and to exchange the secret keys used for bulk stream encryption.

You can get more data through your connection by compressing the data before sending through the VPN tunnel. **Compression** removes redundancy.

Also, make sure that the remote users who connect to the VPN with VPN client software are properly secured. Hacking home computers from the Internet is very easy, and can become an attack vector into your network if the home computer is running a VPN tunnel to it.