Network Security

Damien Magoni

Contents

- Define network security attacks, services and mechanisms
- Understand symmetric/asymmetric encryption/decryption and cryptographic hash functions
- Understand message authentication code, digital signature, key management
- Understand port-based access control, WiFi security and firewalls
- Describe network, transport and application layer security

Network security attacks, services and mechanisms

Part 1

Objectives

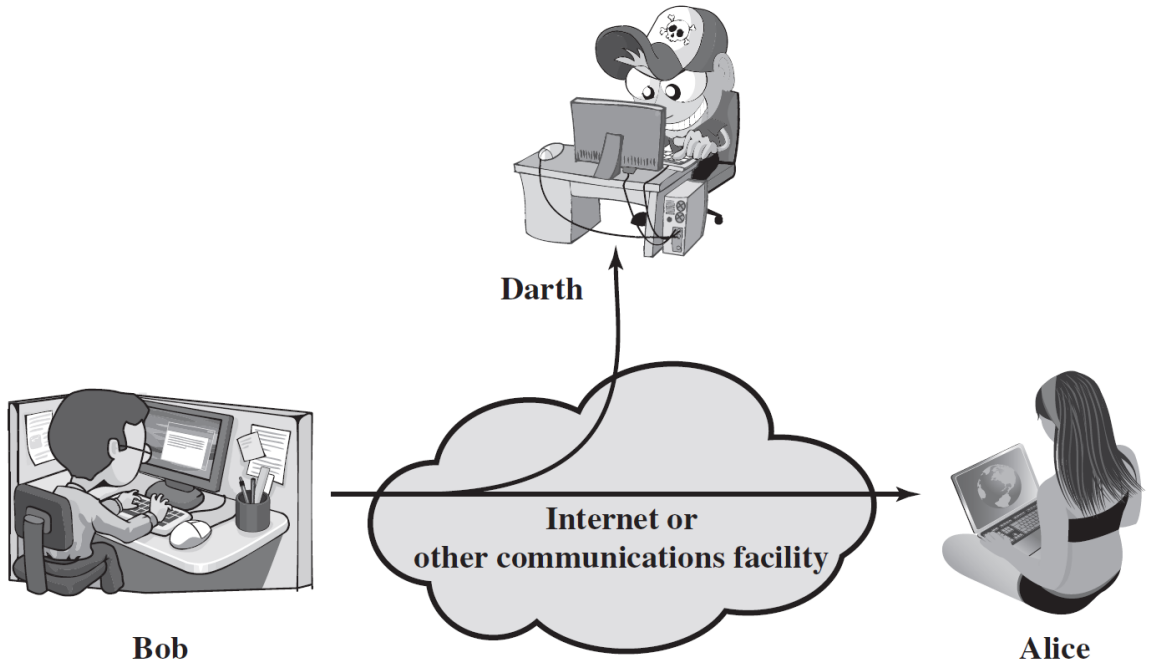
- Define computer security concepts
- Define network security attacks
- Define network security services
- Define network security mechanisms

Computer security concepts

- **CIA triad mnemonic**
- **Confidentiality**: Preserving authorized restrictions on information access and disclosure, including means for protecting personal privacy and proprietary information. A loss of confidentiality is the unauthorized disclosure of information.
- **Integrity**: Guarding against improper information modification or destruction, including ensuring information nonrepudiation and authenticity. A loss of integrity is the unauthorized modification or destruction of information.
- **Availability**: Ensuring timely and reliable access to and use of information. A loss of availability is the disruption of access to or use of information or an information system.

Network security attacks - Passive

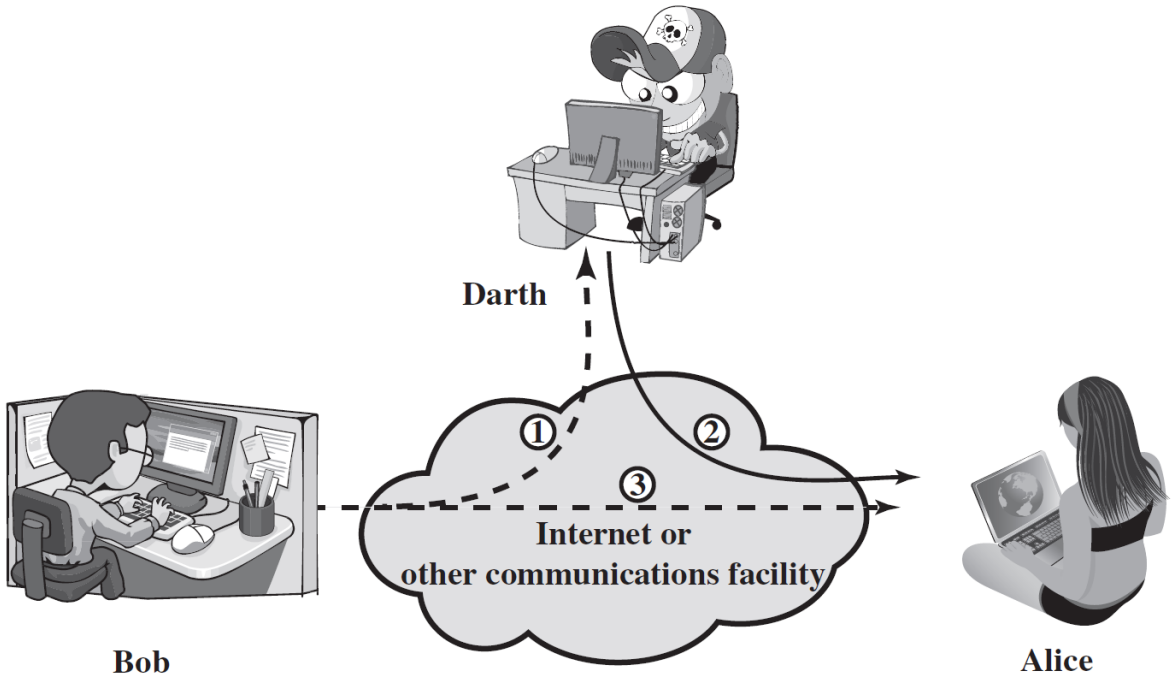
- Passive attacks are eavesdropping on, or monitoring of, transmissions. The goal of the opponent is to obtain information that is being transmitted.
- Two types of passive attacks:
  - The **release of message contents** is, for an opponent, to learn the contents of a transmission.
  - **Traffic analysis** is, for an opponent, to observe the pattern of protected messages, i.e. metadata, in order to determine the location and identity of communicating hosts as well as guessing the nature of the communication that was taking place.
- Passive attacks are very difficult to detect, because they do not involve any alteration of the data.



(a) Passive attacks

Network security attacks - Active

- Active attacks involve modification of the data stream and can be subdivided into four categories:
- A **masquerade** takes place when one entity pretends to be a different entity, in order to obtain extra privileges by impersonating an entity that has those privileges.
- **Replay** involves the passive capture of a data unit and its subsequent retransmission to produce an unauthorized effect.
- **Modification** of messages means that some portion of a legitimate message is altered, or that messages are delayed or reordered, to produce an unauthorized effect.
- The **denial of service** prevents the normal use or management of communications facilities. This attack may have a specific target or may disrupt an entire network, either by disabling its devices or by overloading it with messages so as to degrade its performance.



(b) Active attacks

Network security services

- A security **service** is a processing or communication service that is provided by a system to give a specific kind of protection to system resources.
- Security services implement security **policies** and are implemented by security **mechanisms.**
- The ISO X.800 standard defines 5 service categories and 14 specific services:
  - Authentication
  - Access control
  - Data confidentiality
  - Data integrity
  - Non-répudiation

AUTHENTICATION

The assurance that the communicating entity is the one that it claims to be.

Peer Entity Authentication
Used in association with a logical connection to provide confidence in the identity of the entities connected.

Data-Origin Authentication
In a connectionless transfer, provides assurance that the source of received data is as claimed.

ACCESS CONTROL

The prevention of unauthorized use of a resource (i.e., this service controls who can have access to a resource, under what conditions access can occur, and what those accessing the resource are allowed to do).

DATA CONFIDENTIALITY

The protection of data from unauthorized disclosure.

Connection Confidentiality
The protection of all user data on a connection.

Connectionless Confidentiality
The protection of all user data in a single data block

Selective-Field Confidentiality
The confidentiality of selected fields within the user data on a connection or in a single data block.

Traffic-Flow Confidentiality
The protection of the information that might be derived from observation of traffic flows.

Network security services cont' and Availability Service

- X.800 and RFC 4949 define **availability** to be the property of a system being accessible and usable upon demand by an authorized system entity. A variety of attacks can result in the loss of or reduction in availability.
- X.800 treats availability as a property to be associated with various security services. However, it makes sense to call out specifically an availability service.
- An **availability service** is one that protects a system to ensure its availability. This service addresses the security concerns raised by denial-of-service attacks. It depends on proper management and control of system resources and thus depends on access control service and other security services.

DATA INTEGRITY

The assurance that data received are exactly as sent by an authorized entity (i.e., contain no modification, insertion, deletion, or replay).

Connection Integrity with Recovery
Provides for the integrity of all user data on a connection and detects any modification, insertion, deletion, or replay of any data within an entire data sequence, with recovery attempted.

Connection Integrity without Recovery
As above, but provides only detection without recovery.

Selective-Field Connection Integrity
Provides for the integrity of selected fields within the user data of a data block transferred over a connection and takes the form of determination of whether the selected fields have been modified, inserted, deleted, or replayed.

Connectionless Integrity
Provides for the integrity of a single connectionless data block and may take the form of detection of data modification. Additionally, a limited form of replay detection may be provided.

Selective-Field Connectionless Integrity
Provides for the integrity of selected fields within a single connectionless data block; takes the form of determination of whether the selected fields have been modified.

NONREPUDIATION

Provides protection against denial by one of the entities involved in a communication of having participated in all or part of the communication.

Nonrepudiation, Origin
Proof that the message was sent by the specified party.

Nonrepudiation, Destination
Proof that the message was received by the specified party.

Security mechanisms

- Security mechanisms are defined in X.800.
- The mechanisms are divided in two categories:
  - Those that are implemented in a **specific** protocol layer.
  - Those that are not specific to any particular protocol layer or security service, called **pervasive.**

SPECIFIC SECURITY MECHANISMS

May be incorporated into the appropriate protocol layer in order to provide some of the OSI security services.

Encipherment
The use of mathematical algorithms to transform data into a form that is not readily intelligible. The transformation and subsequent recovery of the data depend on an algorithm and zero or more encryption keys.

Digital Signature
Data appended to, or a cryptographic transformation of, a data unit that allows a recipient of the data unit to prove the source and integrity of the data unit and protect against forgery (e.g., by the recipient).

Access Control
A variety of mechanisms that enforce access rights to resources.

Data Integrity
A variety of mechanisms used to assure the integrity of a data unit or stream of data units.

Pervasive security mechanisms

SPECIFIC SECURITY MECHANISMS

Authentication Exchange
A mechanism intended to ensure the identity of an entity by means of information exchange.

Traffic Padding
The insertion of bits into gaps in a data stream to frustrate traffic analysis attempts.

Routing Control
Enables selection of particular physically secure routes for certain data and allows routing changes, especially when a breach of security is suspected.

Notarization
The use of a trusted third party to assure certain properties of a data exchange.

PERVASIVE SECURITY MECHANISMS

Mechanisms that are not specific to any particular OSI security service or protocol layer.

Trusted Functionality
That which is perceived to be correct with respect to some criteria (e.g., as established by a security policy).

Security Label
The marking bound to a resource (which may be a data unit) that names or designates the security attributes of that resource.

Event Detection
Detection of security-relevant events.

Security Audit Trail
Data collected and potentially used to facilitate a security audit, which is an independent review and examination of system records and activities.

Security Recovery
Deals with requests from mechanisms, such as event handling and management functions, and takes recovery actions.

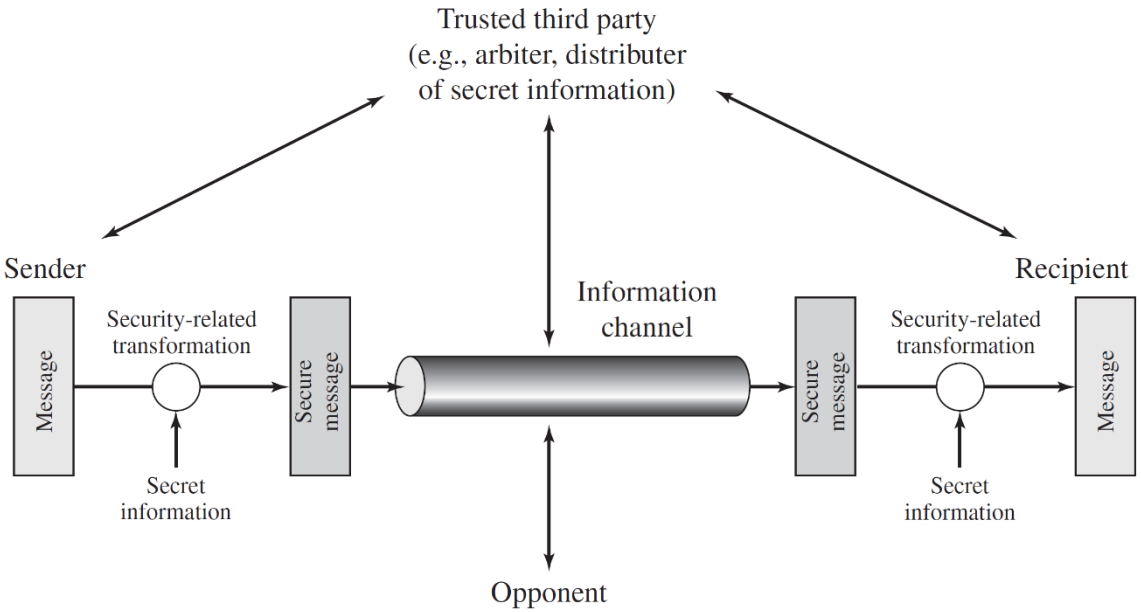
Relationship between services and mechanisms

- To implement a given service, one or more mechanisms must be used as indicated in the table.

SERVICE \ MECHANISM	Encipherment	Digital signature	Access control	Data integrity	Authentication exchange	Traffic padding	Routing control	Notarization
Peer entity authentication	Y	Y			Y			
Data origin authentication	Y	Y						
Access control			Y					
Confidentiality	Y						Y	
Traffic flow confidentiality	Y					Y	Y	
Data integrity	Y	Y		Y				
Nonrepudiation		Y		Y				Y
Availability				Y	Y			

Network security model

- A message is to be transferred from one party to another across an communication channel. The two parties, who are the principals in this transaction, must cooperate for the exchange to take place. A logical information channel is established by the cooperative use of communication protocols by the two principals. Security is necessary to protect the information transmission from an opponent. All the techniques for providing security have two components
  - A security-related transformation on the information to be sent (e.g., encryption of the message, addition of a code based on the contents of the message, which can be used to verify the identity of the sender)
  - Some secret information shared by the two principals and unknown to the opponent (e.g. encryption key)
- A trusted third party may be needed to achieve secure transmission. A third party may be responsible for safely distributing the secret information to the two principals. A third party may be needed to arbitrate disputes between the two principals concerning the authenticity of a message transmission.



Cryptographic algorithms and functions

Part 2

Objectives

- Define terms
- Understand symmetric cyphers with DES and AES as use cases
- Understand asymmetric cyphers with RSA as use case
- Understand cryptographic hash functions with SHA-2 as use case

Terminology

- An original message is known as the **plaintext**, while the coded message is called the **ciphertext**.
- The process of **converting** from plaintext to ciphertext is known as **enciphering** or **encryption.**
- **Restoring** the plaintext from the ciphertext is **deciphering** or **decryption**.
- The many schemes used for encryption constitute the area of study known as **cryptography**. Such a scheme is known as a cryptographic system or a cipher.
- Techniques used for deciphering a message without any knowledge of the enciphering details fall into the area of **cryptanalysis**. Cryptanalysis is what the layperson calls “breaking the code.”
- The areas of cryptography and cryptanalysis together are called **cryptology**.

Characterisation of crypto systems

- **The type of operations used for transforming plaintext to ciphertext.** All encryption algorithms are based on two general principles:
  - **substitution**, in which each element in the plaintext (bit, letter, group of bits or letters) is mapped into another element
  - **transposition**, in which elements in the plaintext are rearranged (**permutation**).
- **The number of keys used.**
  - If both sender and receiver use the same key, the system is referred to as symmetric, single-key, secret-key, or conventional encryption.
  - If the sender and receiver use different keys, the system is referred to as asymmetric, two-key, or public-key encryption.
- **The way in which the plaintext is processed.**
  - A *block cipher* processes the input one block of elements at a time, producing an output block for each input block.
  - A *stream cipher* processes the input elements continuously, producing output one element at a time, as it goes along.

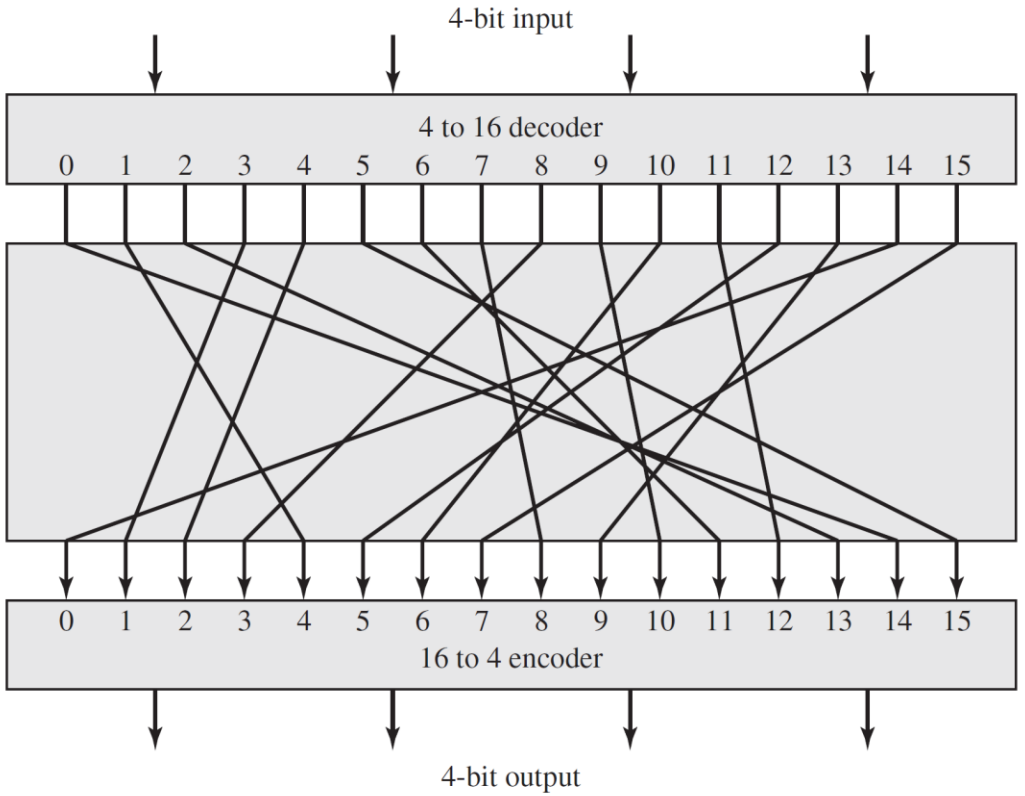
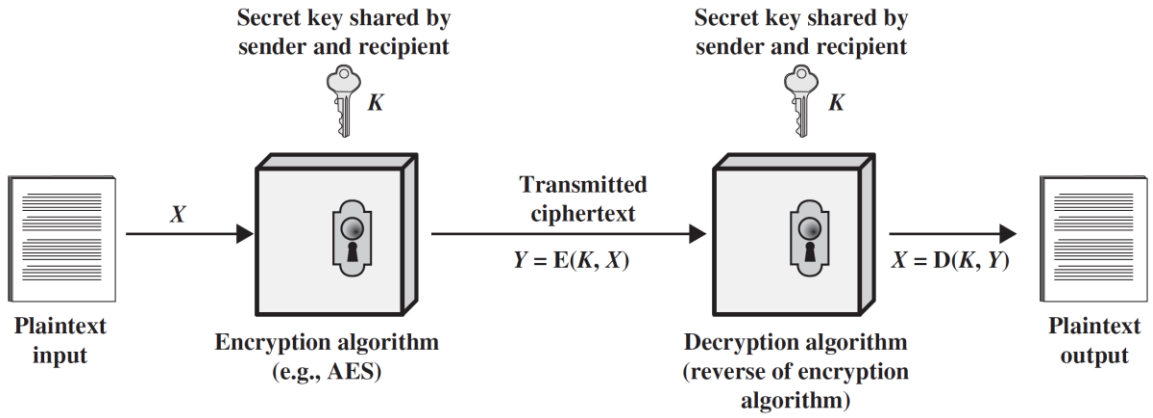


Figure 3.2 General n-bit-n-bit Block Substitution (shown with $n = 4$)

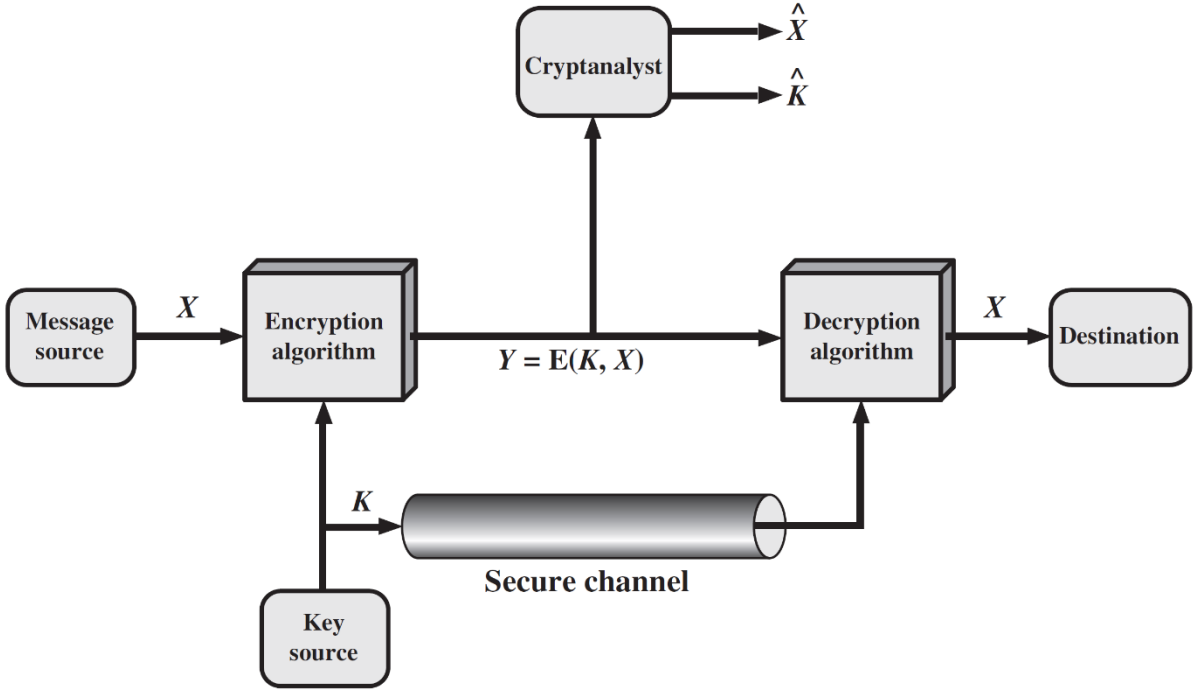
Symmetric cipher/algorithm

- **Encryption algorithm:** performs various substitutions and transpositions (permutations) on the plaintext.
- **Secret key:** is also input to the encryption algorithm. The key is a value independent of the plaintext and of the algorithm. The algorithm will produce a different output depending on the specific key being used at the time. The exact transformations performed by the algorithm depend on the key.
- **Decryption algorithm:** This is essentially the encryption algorithm run in reverse. It takes the ciphertext and the secret key and produces the original plaintext.



Requirements and model for secure use of symmetric encryption

- The opponent should be unable to decrypt ciphertext **or** discover the key even if he or she is in possession of a number of ciphertexts together with the plaintext that produced each ciphertext.
- Sender and receiver must have obtained copies of the secret key in a secure fashion and must keep the key secure. If someone can discover the key and knows the algorithm, all communication using this key is readable.



Feistel block cipher structure

- Feistel defines a product cipher, which is the execution of two or more simple ciphers in sequence (16 in fig) in such a way that the final result or product is cryptographically stronger than any of the component ciphers.
- F is the Feistel function
- The subkey used at each round is derived (i.e. generated) from the secret key

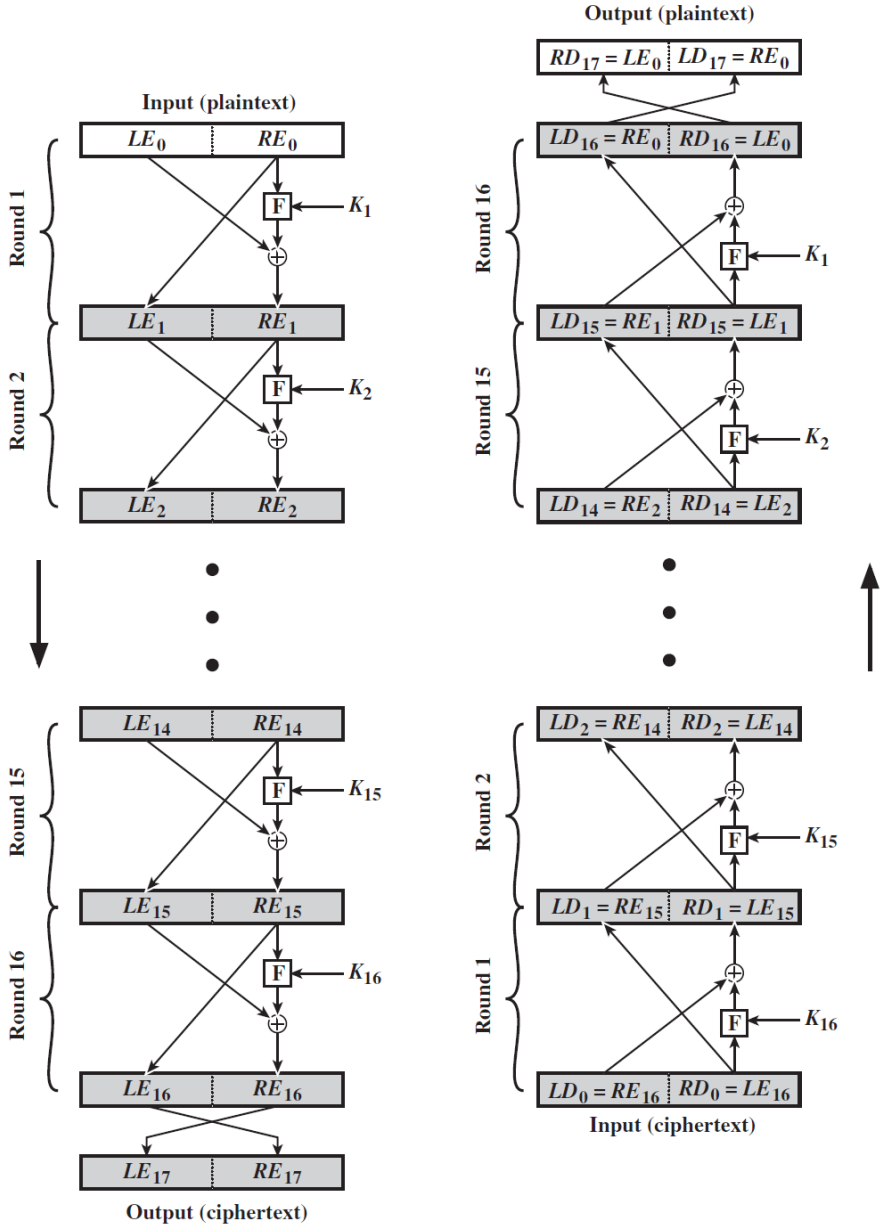
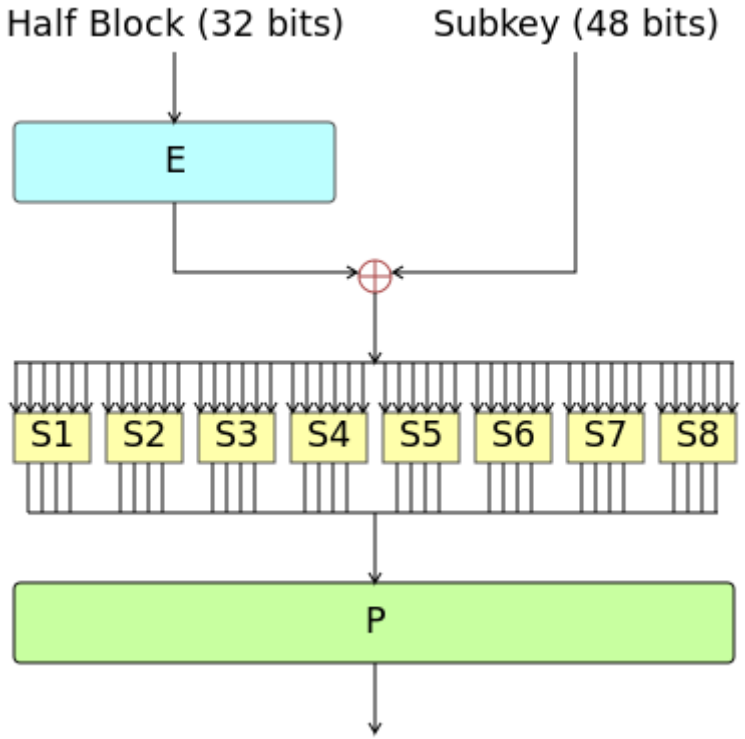


Figure 3.3 Feistel Encryption and Decryption (16 rounds)

Feistel function

- Input is the right-half of the 64-bit block, expanded to 48 bits and XORed with the subkey
- 8 different substitution boxes (S-Boxes) each take 6-bit parts as input and produce 4-bit elements that are concatenated and permuted into a 32-bit block



Data Encryption Standard (DES)

- Based on the Feistel block cipher structure and function with 16 rounds and adding an initial permutation. Standardized by NIST in 1977.
- First, the 64-bit plaintext passes through an initial permutation that rearranges the bits to produce the permuted input.
- This is followed by a phase consisting of 16 rounds of the same function, which involves both permutation and substitution functions.
- The output of the last round consists of 64 bits that are a function of the input plaintext and the key. The left and right halves of the output are swapped to produce the preoutput.
- Finally, the preoutput is passed through a permutation that is the inverse of the initial permutation function, to produce the 64-bit ciphertext

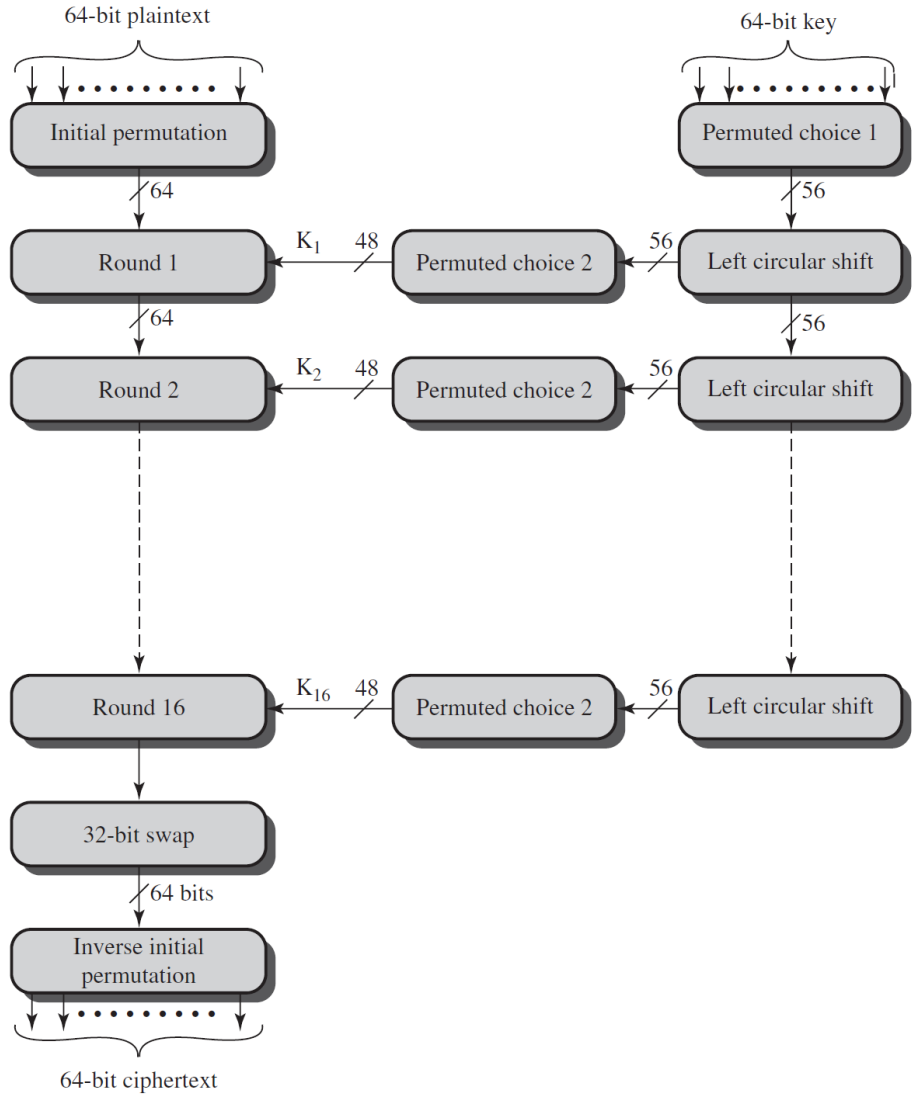


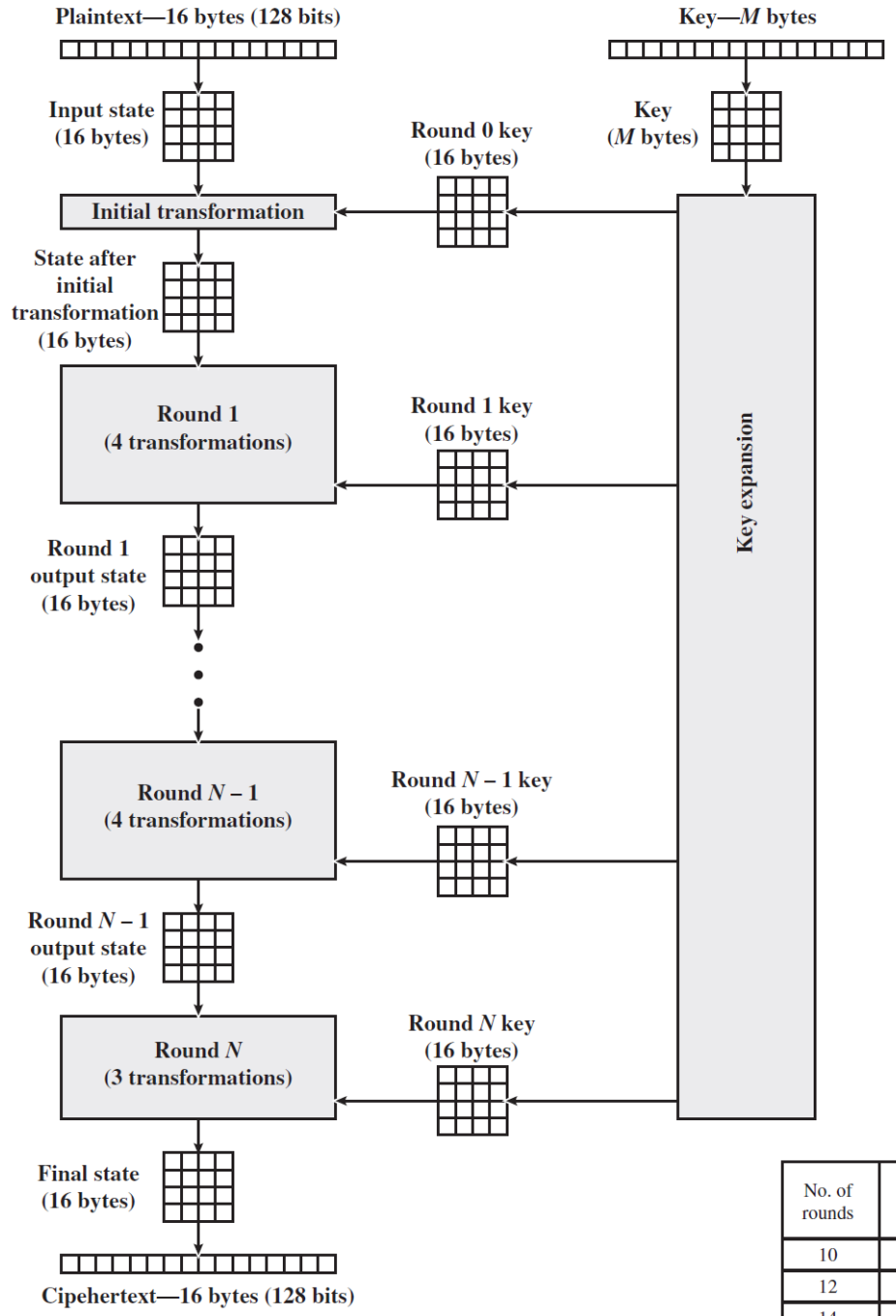
Figure 3.5 General Depiction of DES Encryption Algorithm

Advanced Encryption Standard (AES)

- Best symmetric block cipher to date, standardized in 2001 by NIST to replaces DES
- Operates on 128-bit blocks
- All operations are performed on 8-bit bytes
- Addition, multiplication, and division are performed over the finite Galois field GF(2^8)

AES encryption process

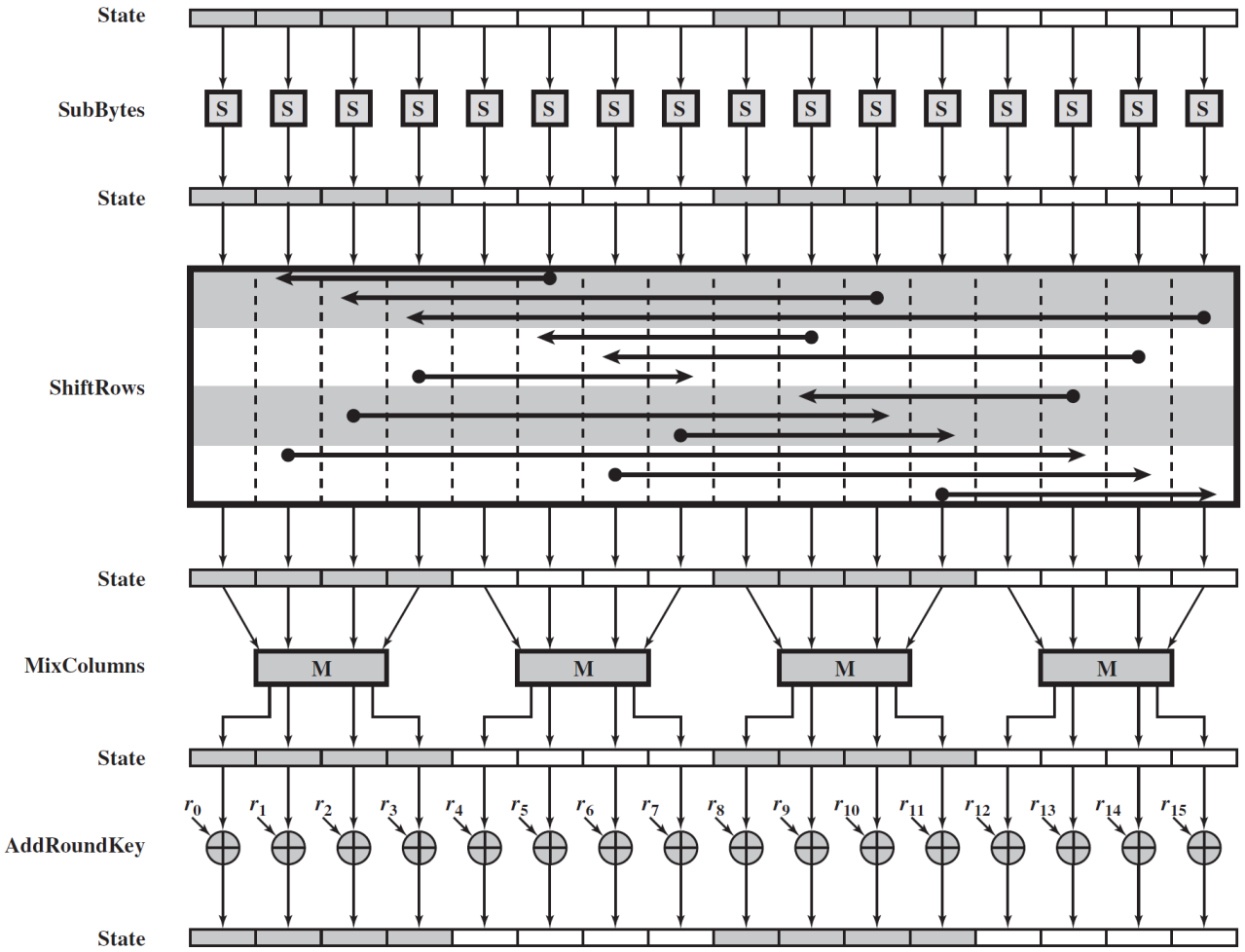
- The cipher takes a plaintext block size of 128 bits (16 bytes).
- The key length can be 128, 192, or 256 bits (16, 24, or 32 bytes). The algorithm is referred to as AES-128, AES-192, or AES-256, depending on the key length
- Each round involves up to 4 transformations. The number of rounds depends on the key length
- The 16 bytes are placed in a 4x4 matrix, column by column



No. of rounds	Key Length (bytes)
10	16
12	24
14	32

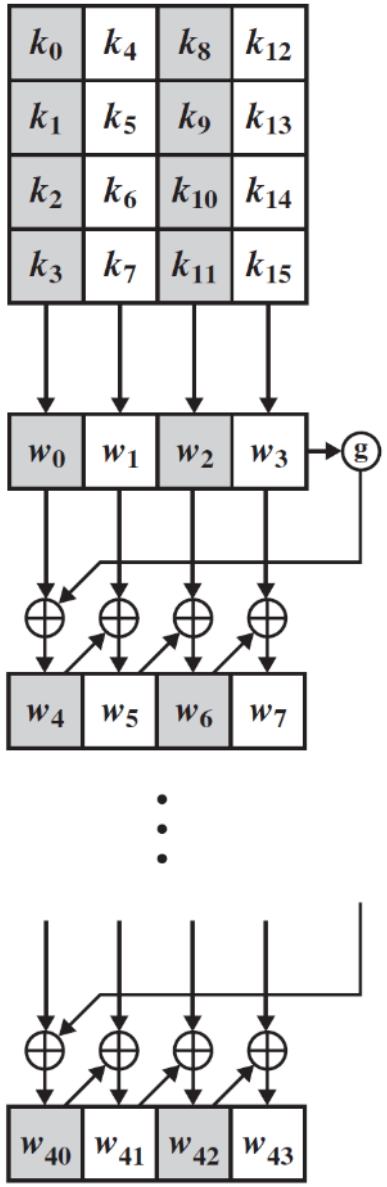
AES round

- Four different stages are used, one of permutation and three of substitution:
  - SubBytes: Uses an S-box to perform a byte-by-byte substitution of the block
  - ShiftRows: A 0 to 3 bytes circular left shift for each row (permutation)
  - MixColumns: A column substitution that involves a matrix multiplication with a vector (input column) over GF(2^8)
  - AddRoundKey: A simple bitwise XOR of the current block with a portion of the expanded key

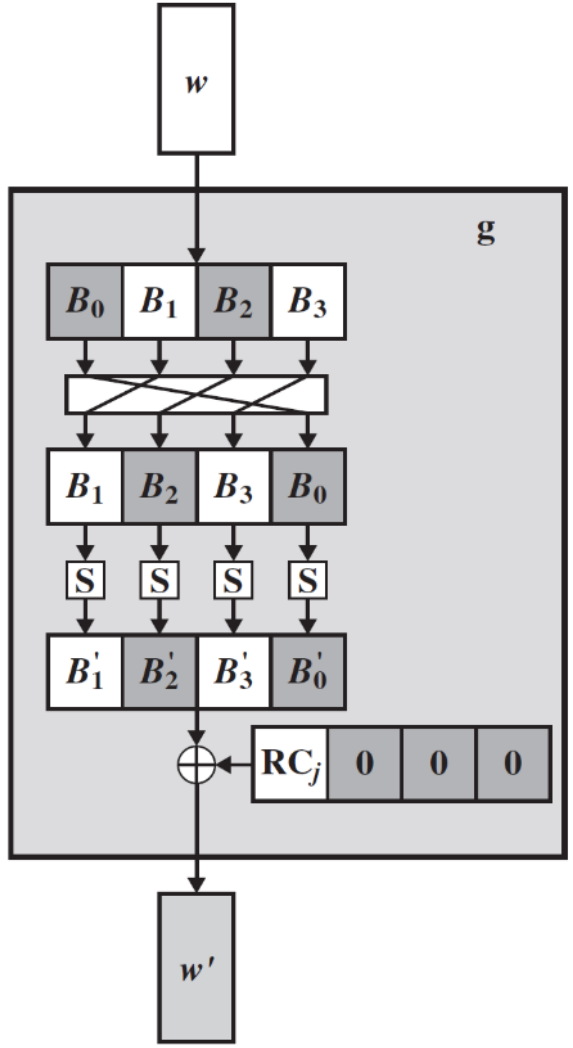


AES key expansion

- The AES key expansion algorithm takes as input a 4-word (16-byte) key and produces a linear array of 44 words (176 bytes) which is sufficient to provide a 4-word round key for the initial AddRoundKey stage and each of the 10 rounds of the cipher.
- S is the same S-Box as the one used in the SubBytes substitution
- The Round Constant (RC) is a word in which the three rightmost bytes are always 0. Thus, the effect of an XOR of a word with RC is to only perform an XOR on the leftmost byte of the word.
- RC is different for each round *j*.



(a) Overall algorithm



(b) Function g

Asymmetric cipher/algorithm

- A public key (asymmetric) cryptographic algorithm uses two related keys: a public key and a private key.
- The two keys are used to perform complementary operations, such as encryption and decryption or signature generation and signature verification.
- They have the property that deriving the private key from the public key is computationally infeasible.
- The public key can be distributed while the private key must remain secret.

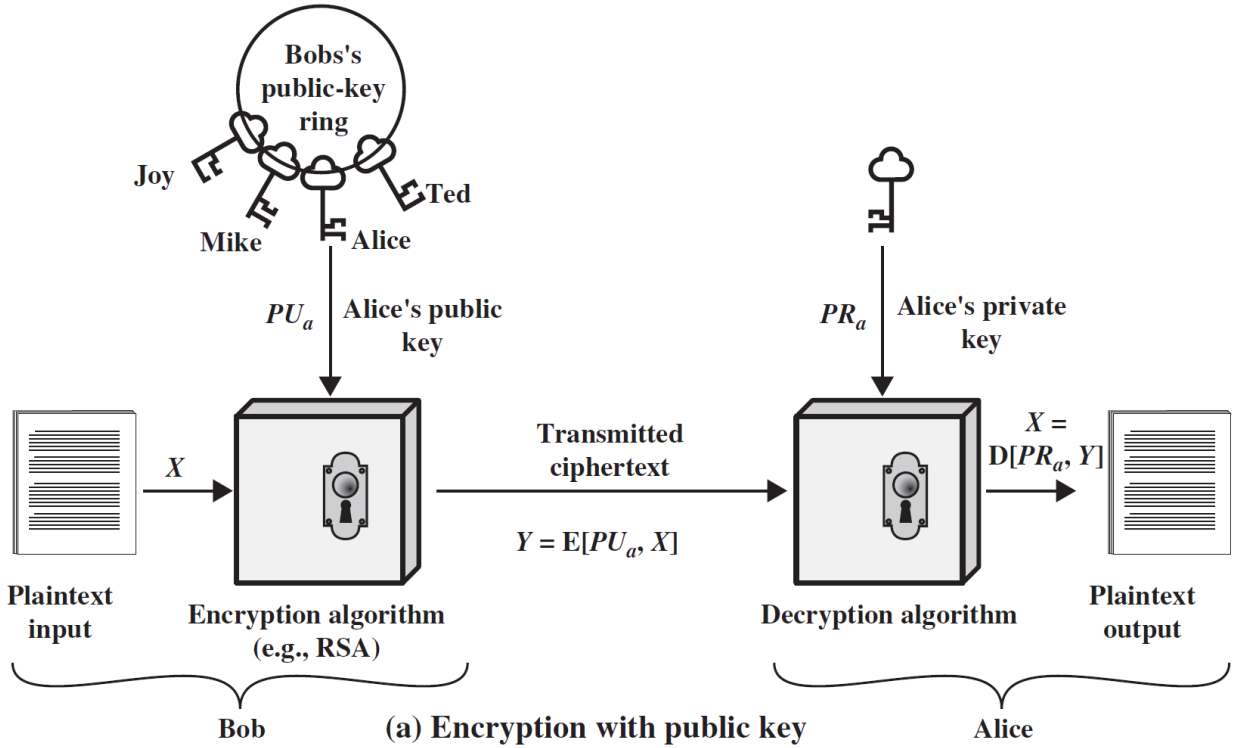
Applications for public key cryptosystems

- Public key systems can perform several functions (encryption, signature, key exchange)
- Among the 4 currently available algorithms only RSA and ECC can be used for any function.
- All algorithms are using computationally hard problems (such as factoring a large number with 2 primes, computing the discrete logarithm or computing a point on an elliptic curve).
- ECC keys are one order of magnitude smaller than RSA keys for the same protection.

Algorithm	Encryption/Decryption	Digital Signature	Key Exchange
RSA	Yes	Yes	Yes
Elliptic Curve	Yes	Yes	Yes
Diffie-Hellman	No	No	Yes
DSS	No	Yes	No

Secrecy by using the receiver's public key

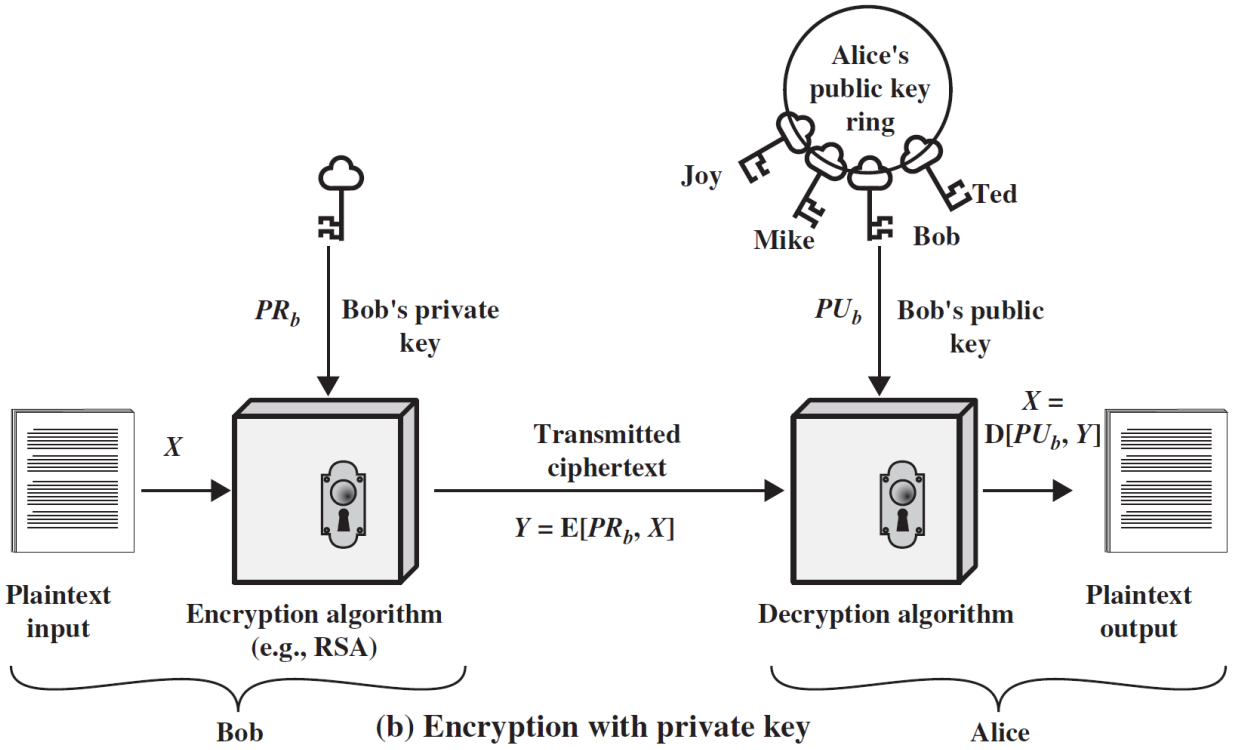
- Everybody can have Alice's public key and use it to encrypt a message for her.
- Only Alice can decrypt the message with her private key (confidentiality/secrecy).



(a) Encryption with public key

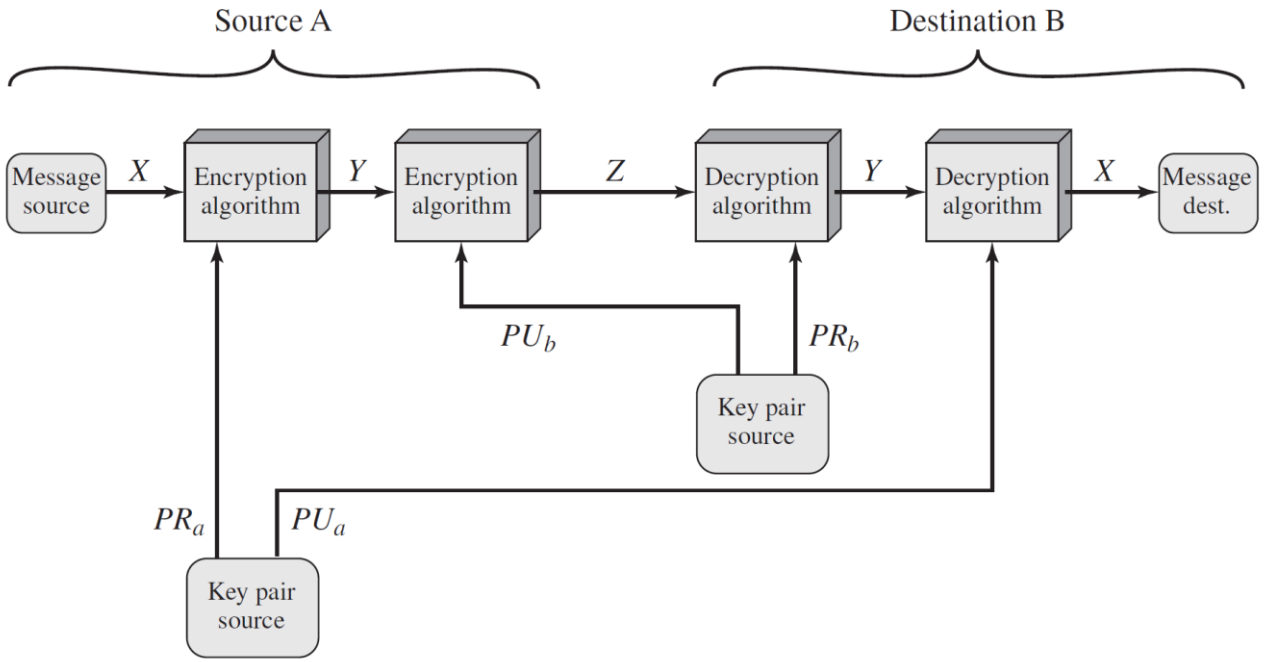
Authentication/signature by using the sender's private key

- Everybody can have Bob's public key and use it to decrypt Bob's message.
- Only Bob could have encrypted the message with its private key (authentication/signature).



Secrecy and authentication with 2 key pairs

- Both methods can be combined to provide secrecy and authentication.
- A authenticates itself by encrypting with its private key PRa and provides secrecy by encrypting with the public key PUb of B.
- B encrypts with PRb and then encrypts with PUa.
- This is not done on data in practice as asymmetric algorithms are computationally costly but secret (symmetric) keys are exchanged that way.



Rivest, Shamir, Adleman (RSA) algorithm

- RSA makes use of an expression with exponentials.
- Plaintext is encrypted in blocks, with each block having a binary value less than some number n. The block size must be less than or equal to $\log_2(n) + 1$, in practice, the block size is i bits, where $2^i < n <= 2^i+1$.
- Encryption and decryption are of the following form, for some plaintext block M and ciphertext block C.
  - $C = M^e \bmod n$
  - $M = C^d \bmod n = (M^e)^d \bmod n = M^{ed} \bmod n$
- Both sender and receiver must know the value of n. The sender knows the value of e, and only the receiver knows the value of d.
- This is a public-key encryption algorithm with:
  - a public key of PU = {e, n}
  - and a private key of PR = {d, n}.

Key Generation by Alice

Select p, q	p and q both prime, $p \neq q$
Calculate $n = p \times q$	
Calcuate $\phi(n) = (p - 1)(q - 1)$	
Select integer e	$\gcd(\phi(n), e) = 1; 1 < e < \phi(n)$
Calculate d	$d \equiv e^{-1} \pmod{\phi(n)}$
Public key	$PU = \{e, n\}$
Private key	$PR = \{d, n\}$

Encryption by Bob with Alice's Public Key

Plaintext:	$M < n$
Ciphertext:	$C = M^e \bmod n$

Decryption by Alice with Alice's Private Key

Ciphertext:	C
Plaintext:	$M = C^d \bmod n$

Cryptographic hash functions

- A **hash** function H accepts a variable-length block of data M as input and produces a fixed-size hash value h = H(M) as output.
- A cryptographic hash function has the following properties:
  - **One-way** property: Given the hash H(M), it is impossible to compute the original input M from H(M).
  - **Non-collision** property: Two different inputs shall not produce the same hash. Any change to any bit or bits in M results, with high probability, in a change to the hash.
  - **Pseudorandomness** property: The results of applying the function to a large set of inputs will produce outputs that are evenly distributed and apparently random.
- A hash code is used as an integrity proof for message authentication and digital signature.
- Existing hash algorithms include MD-4, MD-5, SHA-1, SHA-2, SHA-3.

Secure Hash Algorithm (SHA)

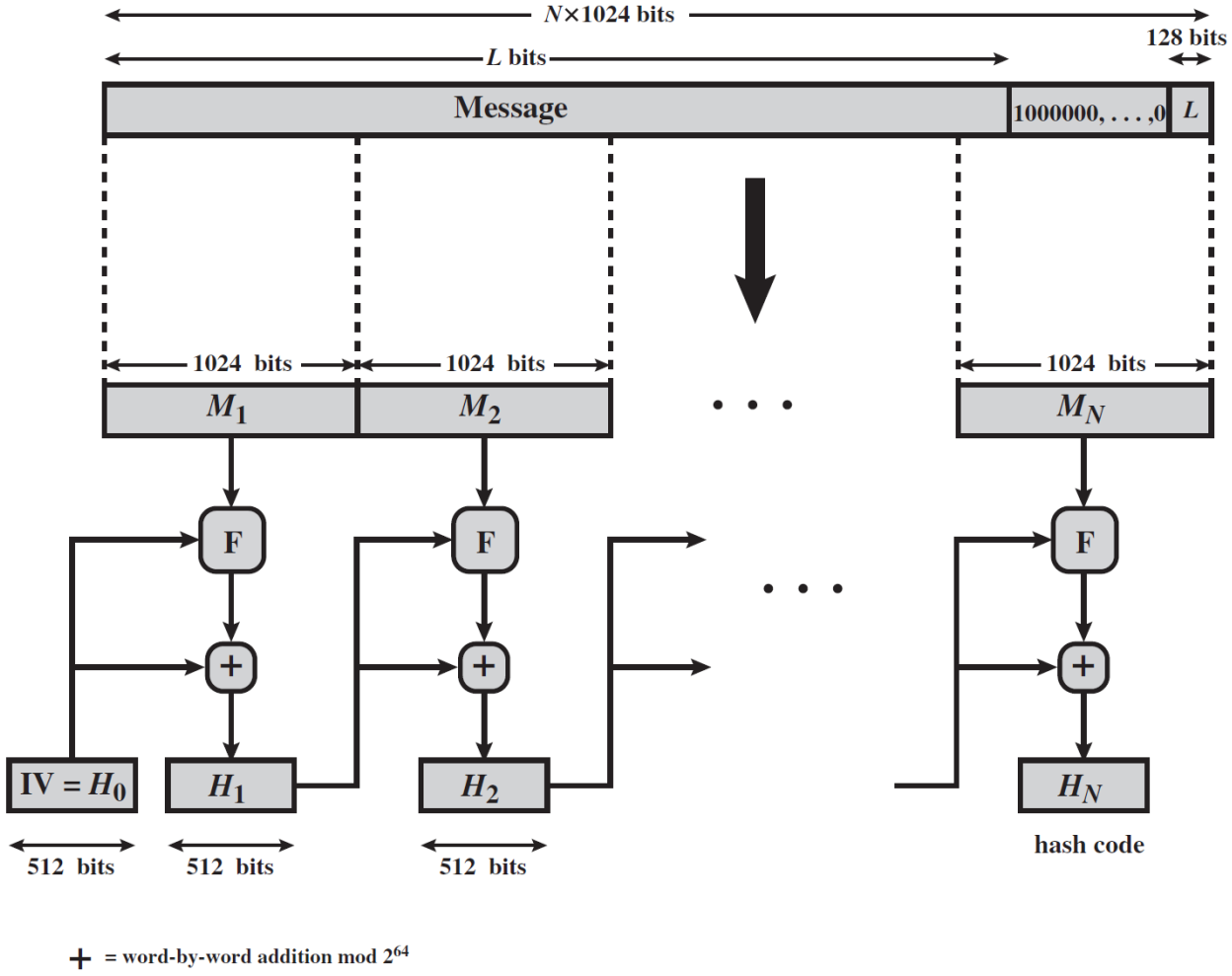
- SHA is a family of hash functions.
- SHA-1 (160-bit hash size) standardized in 1995 is now considered weak.
- SHA-2 (224 to 512-bit hash sizes) standardized in 2002 is now used everywhere but as it shares the same structure as its predecessors it may be broken in the near future.
- SHA-3 based on a different structure (sponge construction) was standardized in 2012 to eventually replace SHA-2.

	SHA-1	SHA-224	SHA-256	SHA-384	SHA-512
Message Digest Size	160	224	256	384	512
Message Size	$< 2^{64}$	$< 2^{64}$	$< 2^{64}$	$< 2^{128}$	$< 2^{128}$
Block Size	512	512	512	1024	1024
Word Size	32	32	32	64	64
Number of Steps	80	64	64	80	80

Note: All sizes are measured in bits.

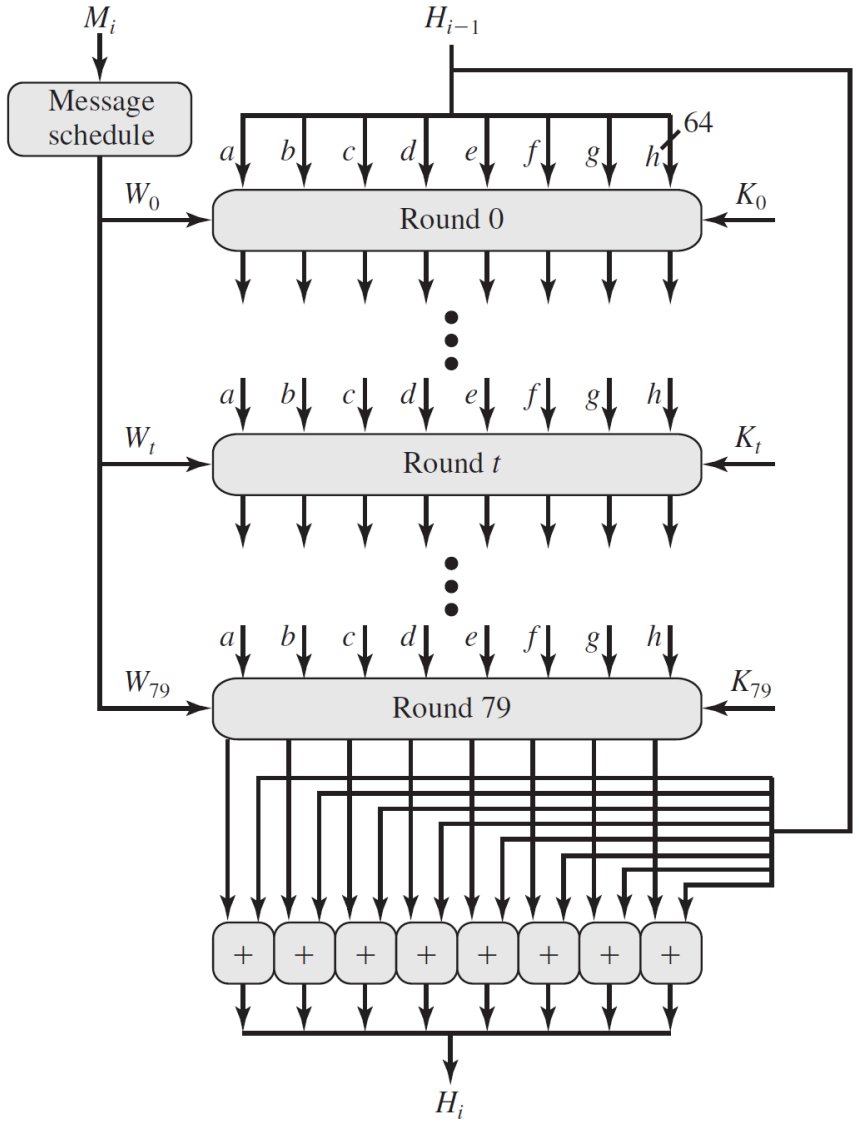
SHA-512 hash code generation

- Step 1 Append padding bits. The message is padded so that its length is congruent to 896 modulo 1024
- Step 2 Append length. A block of 128 bits is appended to the message. This block is treated as an unsigned 128-bit integer (most significant byte first) and contains the length of the original message (before the padding).
- Step 3 Initialize hash buffer. A 512-bit buffer is used to hold intermediate and final results of the hash function. The buffer can be represented as eight 64-bit registers (a, b, c, d, e, f, g, h). These registers are initialized with predefined constants.
- Step 4 Process message in 1024-bit (128-word) blocks. The heart of the algorithm is a module called F that consists of 80 rounds.
- Step 5 Output. After all N 1024-bit blocks have been processed, the output from the Nth stage is the 512-bit message digest.



SHA-512 Processing of a Single 1024-Bit Block

- Each round takes as input the 512-bit buffer value, abcdefgh, and updates the contents of the buffer. At input to the first round, the buffer has the value of the intermediate hash value, H_{i-1}. Each round t makes use of a 64-bit value W_t, derived from the current 1024-bit block being processed (M_i). These values are derived using a message schedule not described here.
- Each round also makes use of an additive constant K_t where 0 … t … 79 indicates one of the 80 rounds. These words represent the first 64 bits of the fractional parts of the cube roots of the first 80 prime numbers.
- The constants provide a “randomized” set of 64-bit patterns, which should eliminate any regularities in the input data.



Authentication and signature

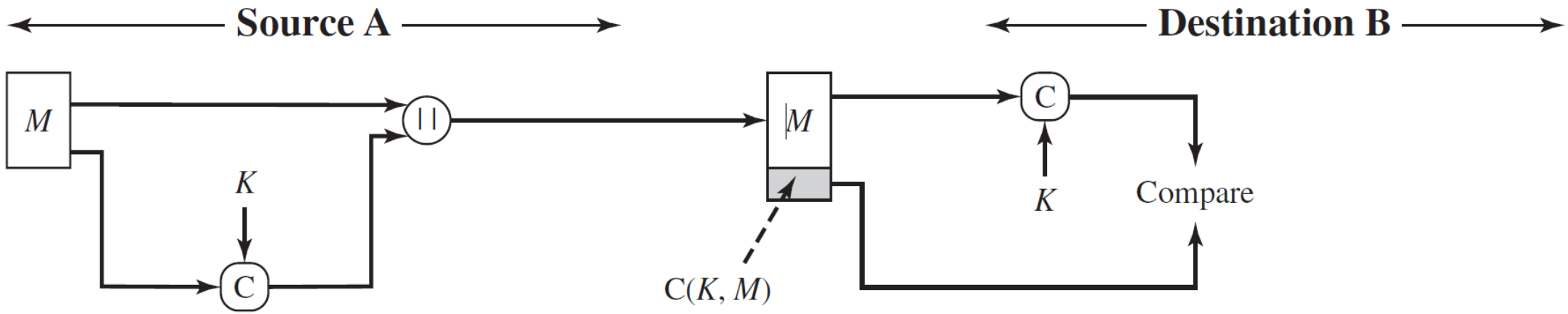
Part 3

Objectives

- Understand message authentication code with HMAC and CMAC as use cases
- Understand digital signature with DSA as a use case
- Understand key distribution and public key certificates

Message Authentication Code (MAC)

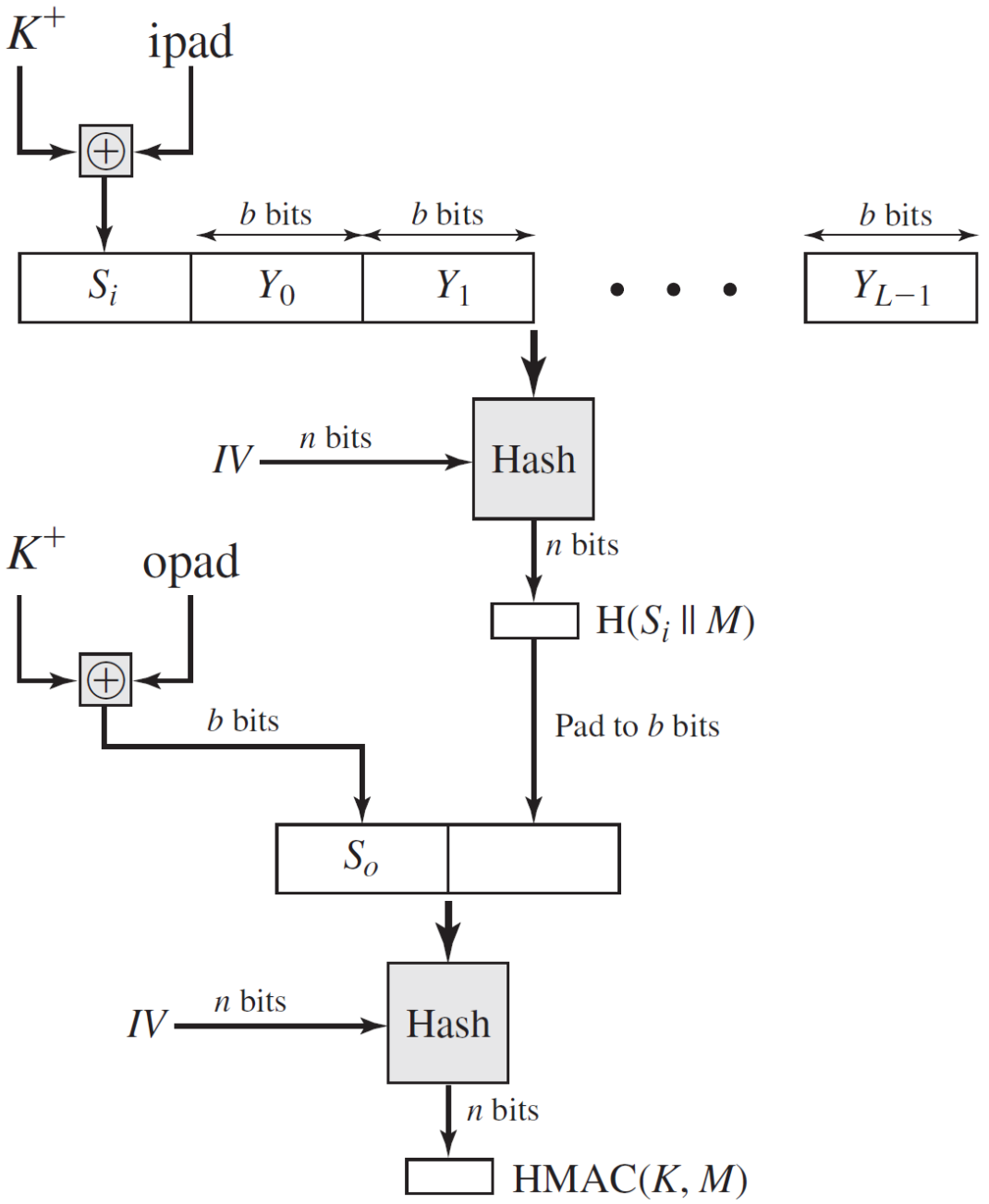
- Asymmetric ciphers can provide message authentication but they are computationally costly.
- An alternative technique involves the use of a secret key to generate a cryptographic **checksum** or **MAC**, that is appended to the message. This technique assumes that the two communicating parties, A and B, **share** a common **secret key** K.
- The message plus the MAC are transmitted to the intended recipient. The recipient performs the same calculation on the received message, using the same secret key, to generate a new MAC. The received MAC is compared to the calculated MAC.
- If only the receiver and the sender know the identity of the secret key, and if the received MAC matches the calculated MAC, then the receiver is assured that the message has not been altered and is from the alleged sender.



Hash-based MAC (HMAC)

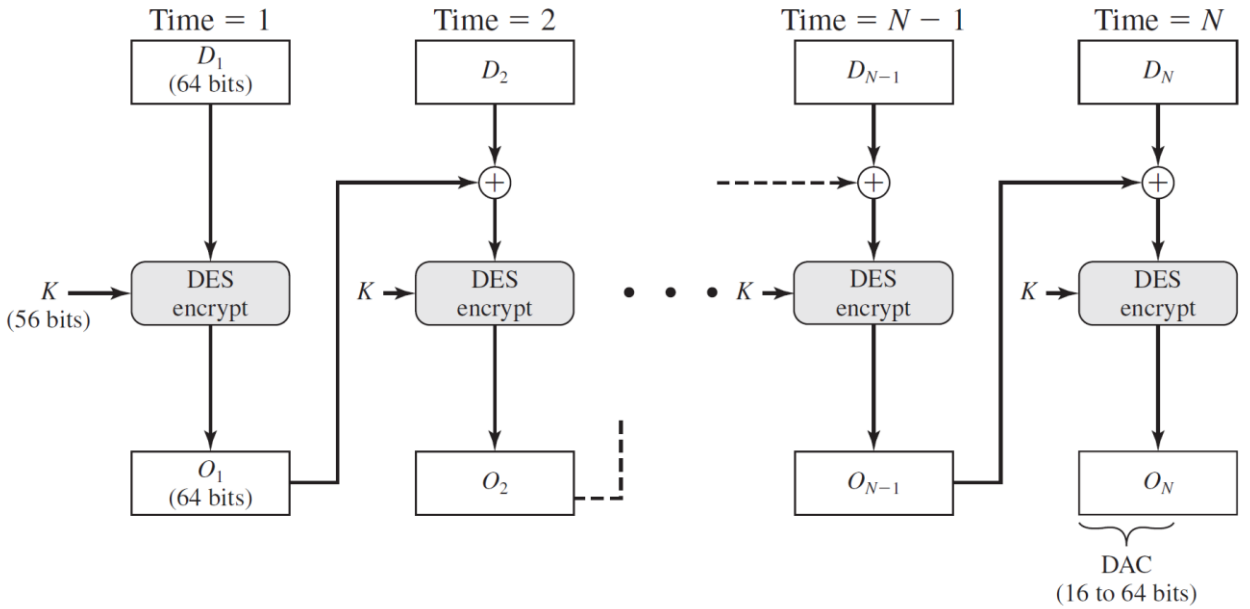
- Defined in RFC2104
- Uses hash function on the message:
  - $HMAC_K$ (M) =

  H[(K' ⊕ opad)||H[(K' ⊕ ipad)||M]]
  - Where K' = key padded to b-bits or hashed to b-bits if |k|>b (with b = block size for the hash)
  - opad, ipad are constants:
    - ipad = 0x36 repeated (b/8) times
    - opad = 0x5C repeated (b/8) times
- Any hash function can be used.
- It is proved that the security of HMAC relates to that of the underlying hash algorithm.



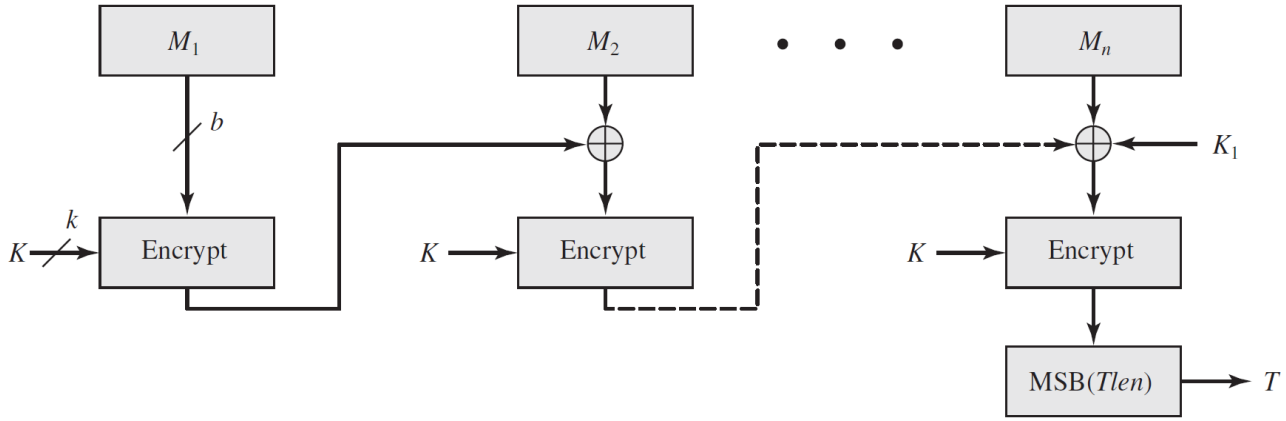
Data Authentication Algorithm (DAA)

- The Data Authentication Algorithm (DAA) is based on DES and is standardized by FIPS PUB 113 and by ANSI X9.17.
- The algorithm uses the cipher block chaining (CBC) mode of operation of DES with an initialization vector of zero.
- The data to be authenticated are grouped into contiguous 64-bit blocks: D_1, D_2, …, D_N. If necessary, the final block is padded on the right with zeroes to form a full 64-bit block.
- DAA has a weakness: given T = CBC-MAC(K, X) of a one-block message X, an adversary immediately knows the CBC-MAC for the two-block message X || (X ⊕ T) since this is once again T.

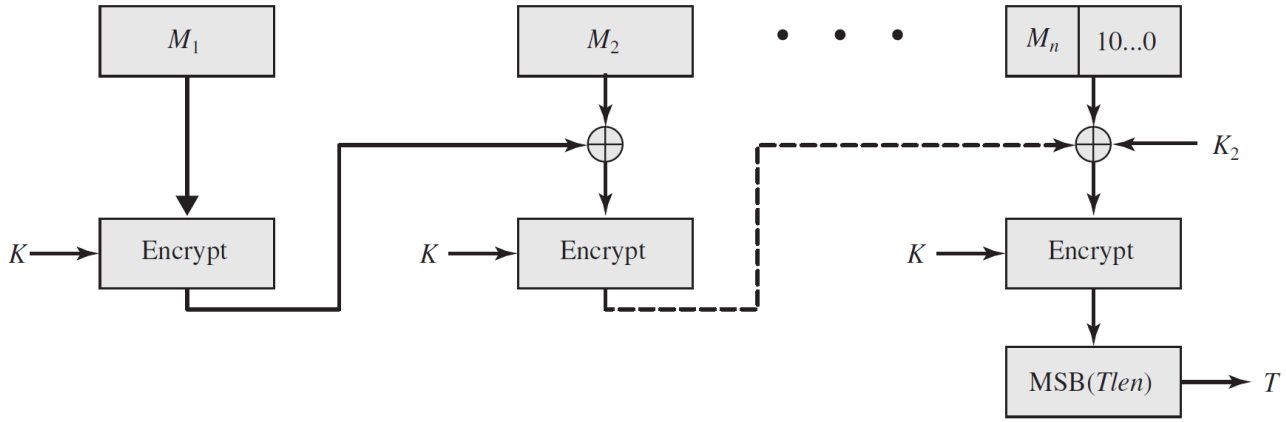


Cipher-Based Message Authentication Code (CMAC)

- The DAA limitation can be overcome using three keys: one key K of length k to be used at each step of the cipher block chaining and two keys of length b, where b is the cipher block length. These keys can be derived from the encryption key, rather than being provided separately.
- This refinement, published by NIST in SP800-38B, is the Cipher-based Message Authentication Code (CMAC) mode of operation for use with AES and triple DES.
- If the message is not an integer multiple of the cipher block length, then the final block is padded to the right (least significant bits) with a 1 and as many 0s as necessary so that the final block is also of length b. The CMAC operation then proceeds as before, except that a different b-bit key K_2 is used instead of K_1.



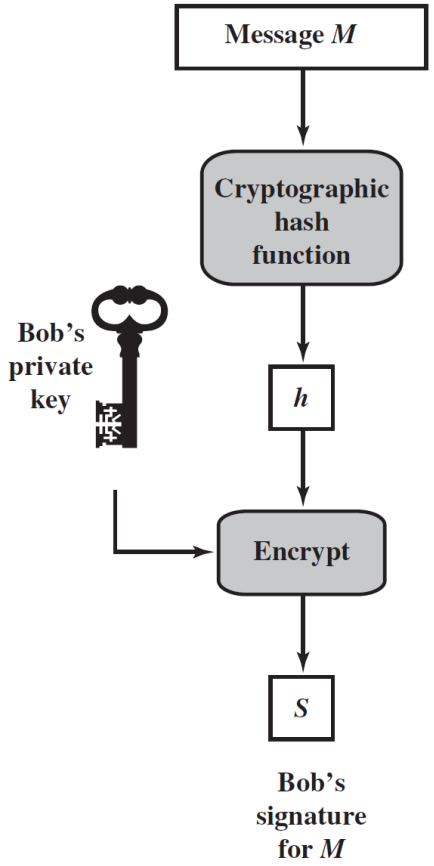
(a) Message length is integer multiple of block size

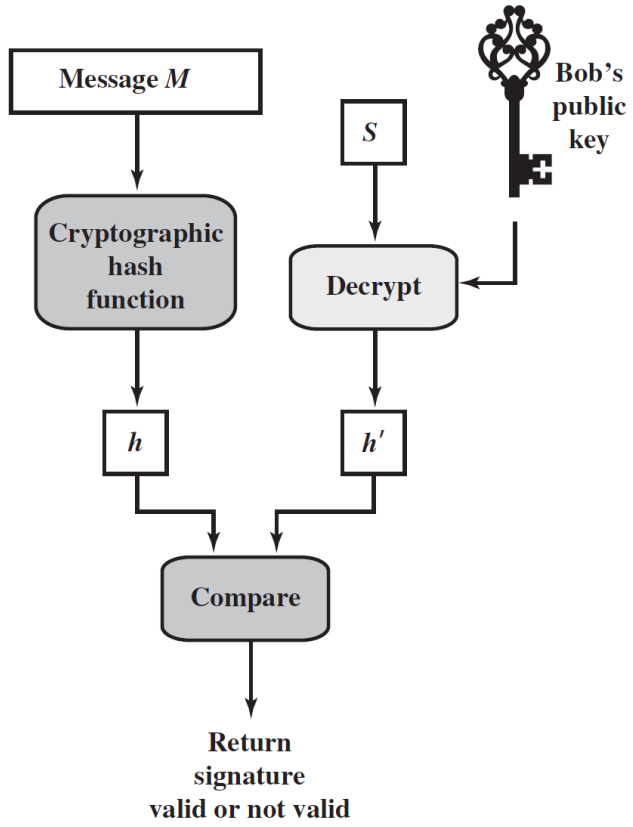


(b) Message length is not integer multiple of block size

Digital signature of messages

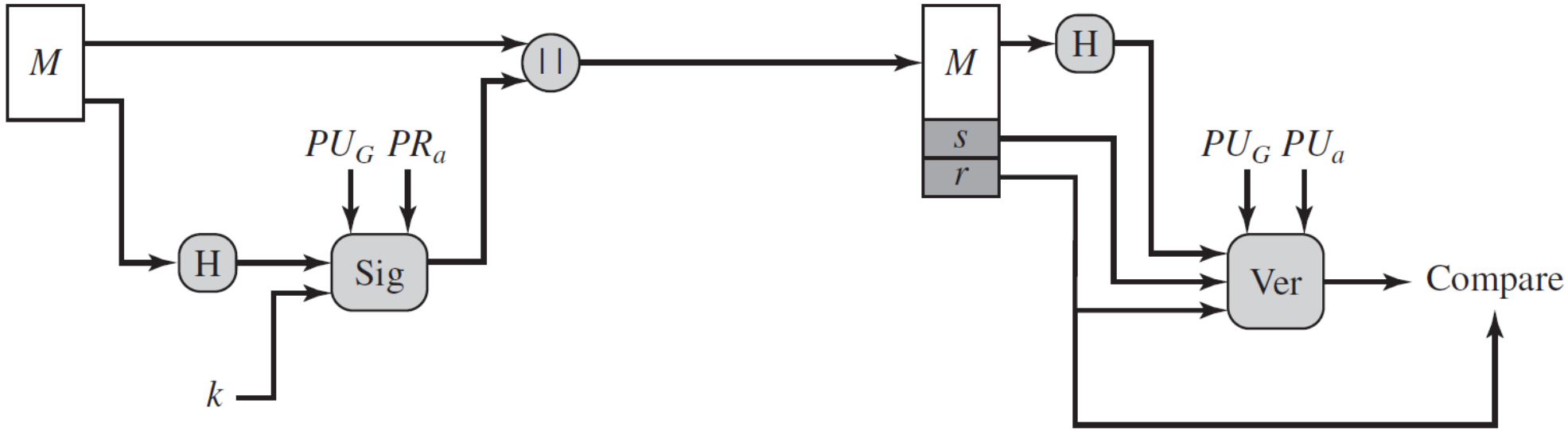
- Use a hash function to compute the hash of the whole message or document.
- Encrypt this hash with your private key and add the result to the document as well as your public key signed by a third party.
- The receiver checks your public key, decrypts the hash and compare it to the hash computed over the document by himself.



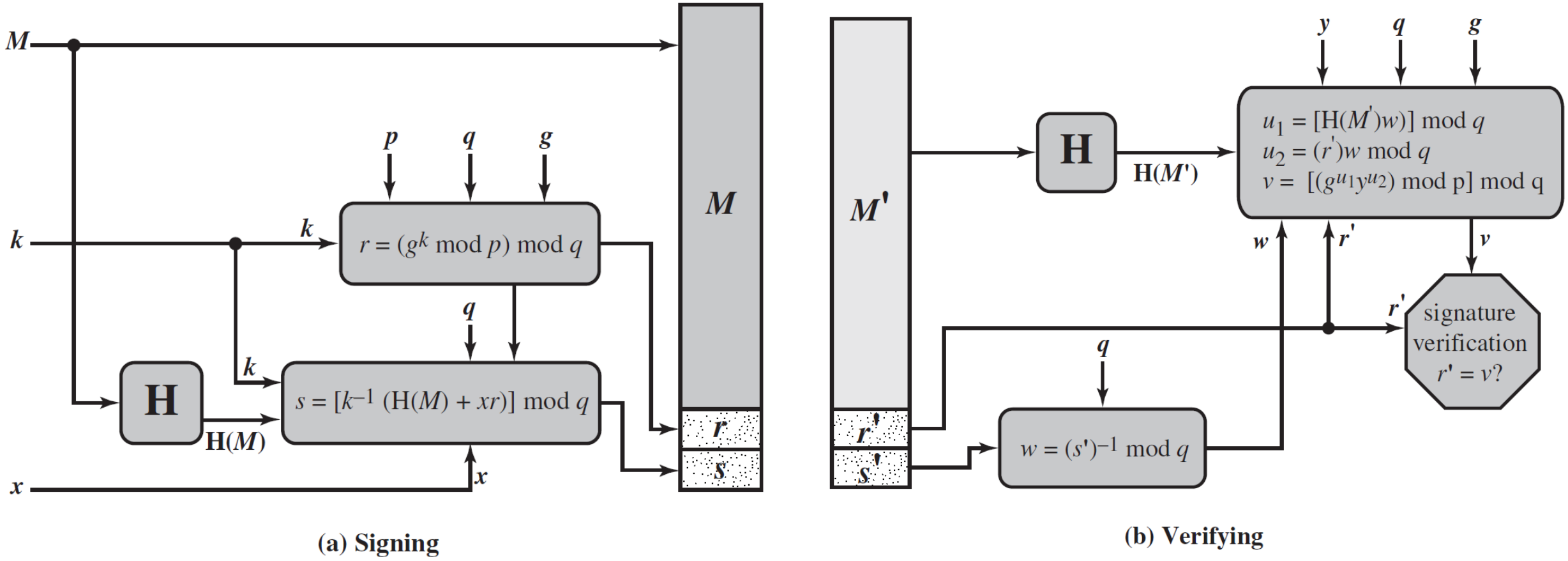


Digital Signature Algorithm (DSA)

- DSA was standardized by NIST as FIPS 186 in 1991, updated and expanded as FIPS 186-2 in 2000. FIPS 186-3 released in 2009 includes newer schemes such as Elliptic Curve DSA (ECDSA) and RSA Probabilistic Signature Scheme (RSA-PSS).
- The SHA hash of the message is provided as input to a signature function along with a random number *k* generated for this particular signature. The signature function also depends on the sender's private key (PR_a) and a set of parameters known to a group of communicating principals and constituting a global public key (PU_G). The result is a signature consisting of two components, labeled s and r.
- On reception, the hash code of the incoming message is generated. This, plus the signature and the public keys PU_G and PU_a are input to a verification function. The output of the verification function is a value that is equal to the signature component r if the signature is valid.



DSA signing and verifying

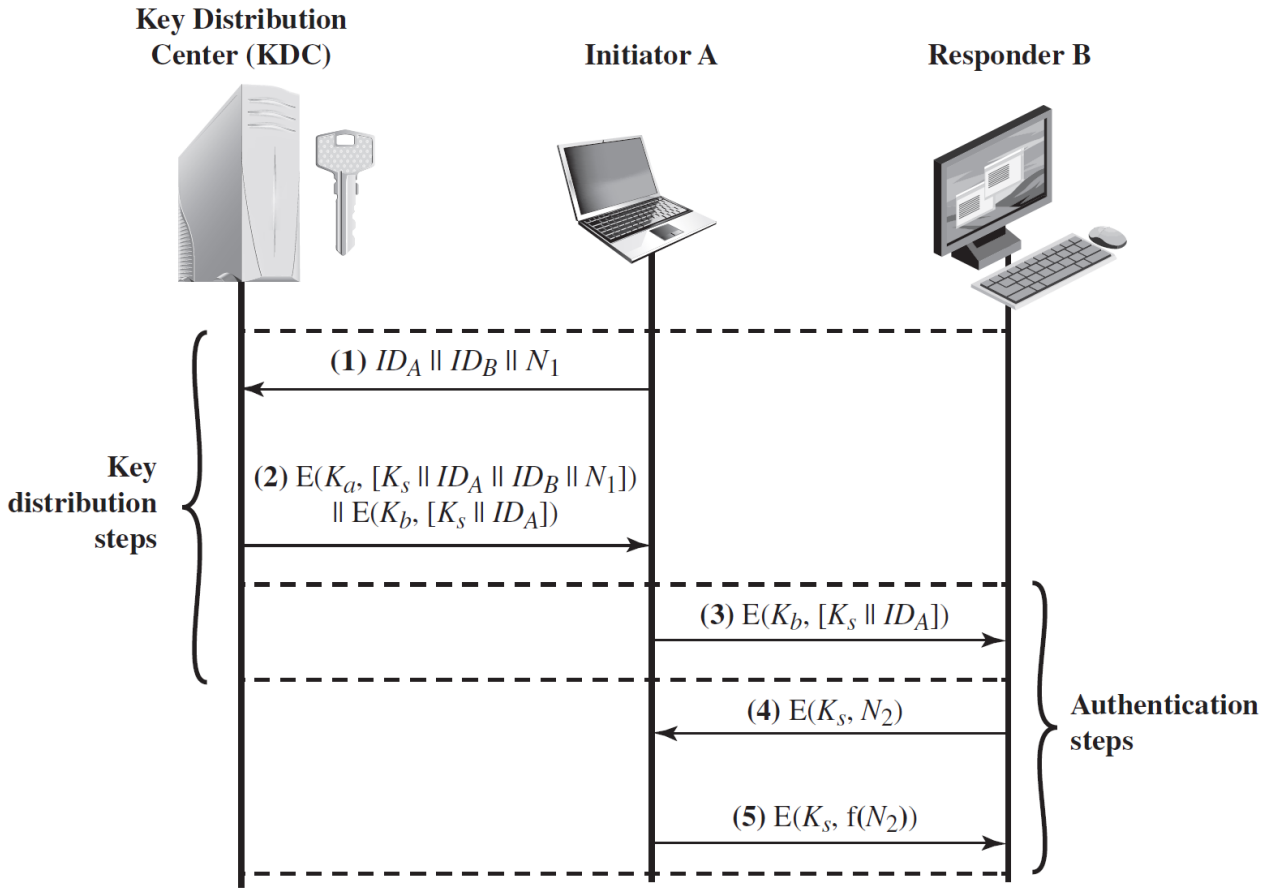


(a) Signing

(b) Verifying

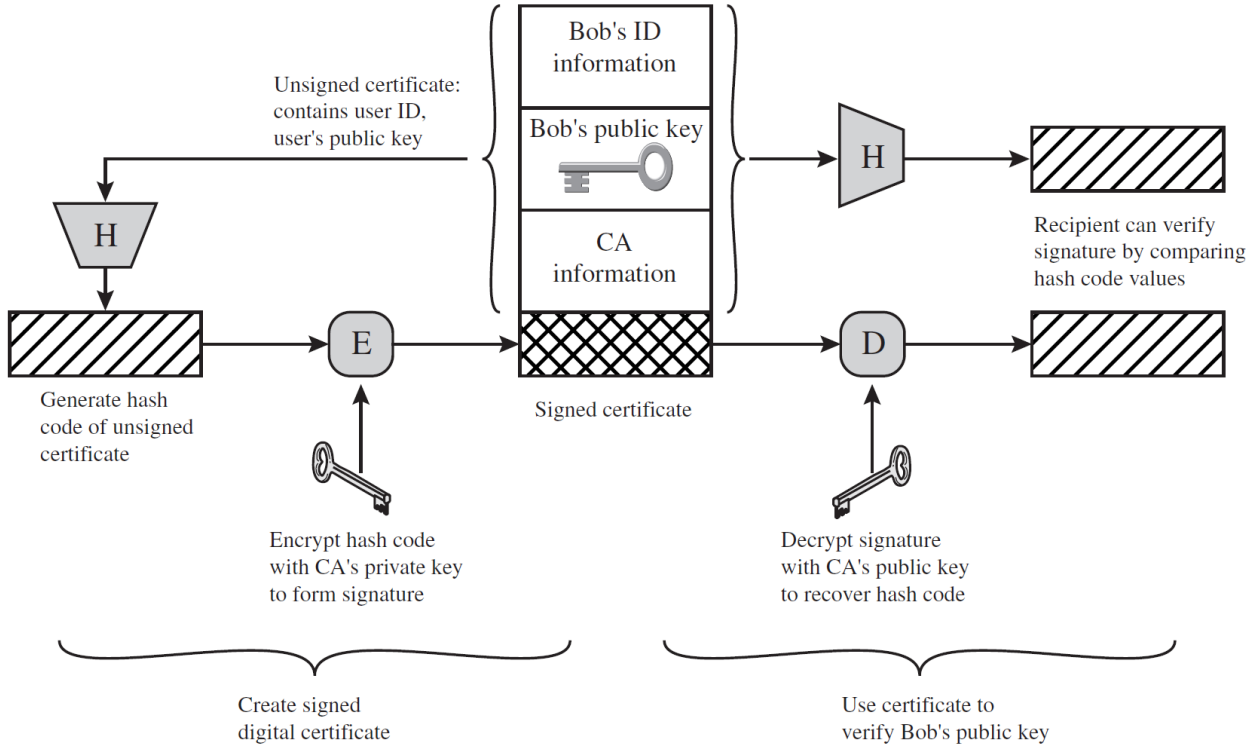
Secret key distribution

- For symmetric encryption to work, the two parties must share the same key, and that key must be protected from access by others.
- Furthermore, frequent key changes are usually desirable to limit the amount of data compromised if an attacker learns the key.
- Therefore, the strength of any cryptographic system rests with the key **distribution** technique, a term that refers to the means of **delivering** a key to two parties who wish to exchange data without allowing others to see the key.
- a **nonce** is an arbitrary number that may only be used once. It is often a random or pseudo-random number issued in an authentication protocol to ensure that old communications cannot be reused in **replay** attacks. They can also be useful as initialization vectors and in cryptographic hash functions.



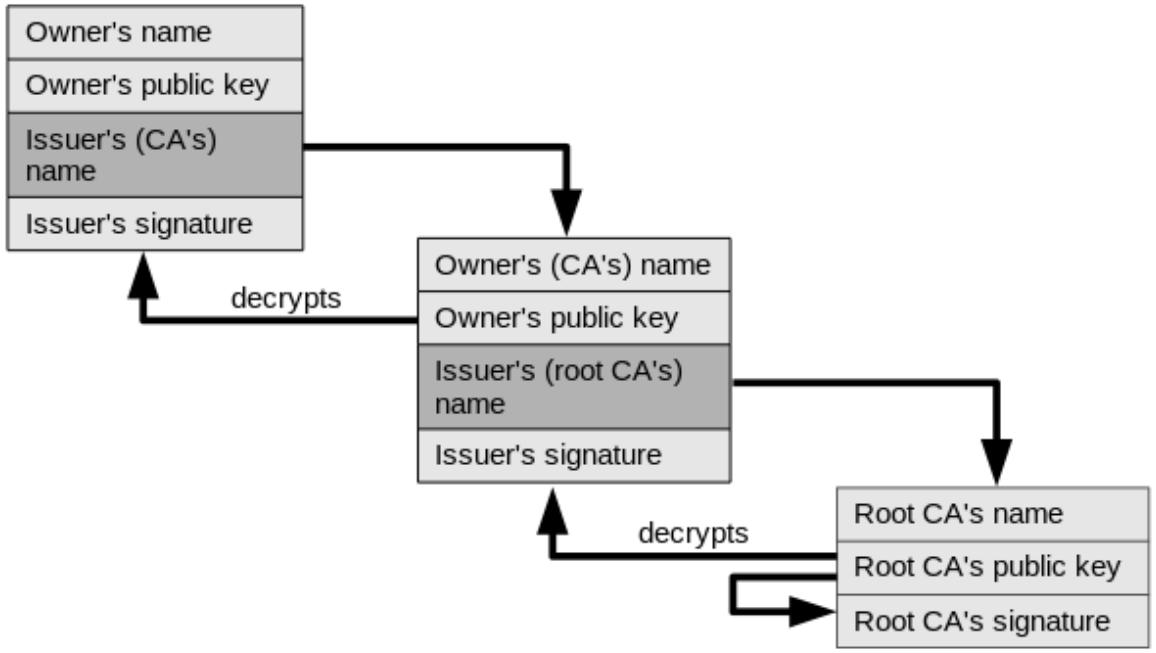
Public key distribution

- Direct key exchange is vulnerable to a man-in-the-middle attack if the keys are not authenticated. The solution is to use public key certificates.
- A **certificate** is a digital document issued and digitally signed by the private key of a trusted 3rd party, named a **Certification Authority** (**CA**), that binds the name of a subscriber to its public key.
- The certificate indicates that the subscriber identified in the certificate has sole control and access to the corresponding private key.
- The ITU-T X.509 standard first released in 1988 defines the structure of certificates.



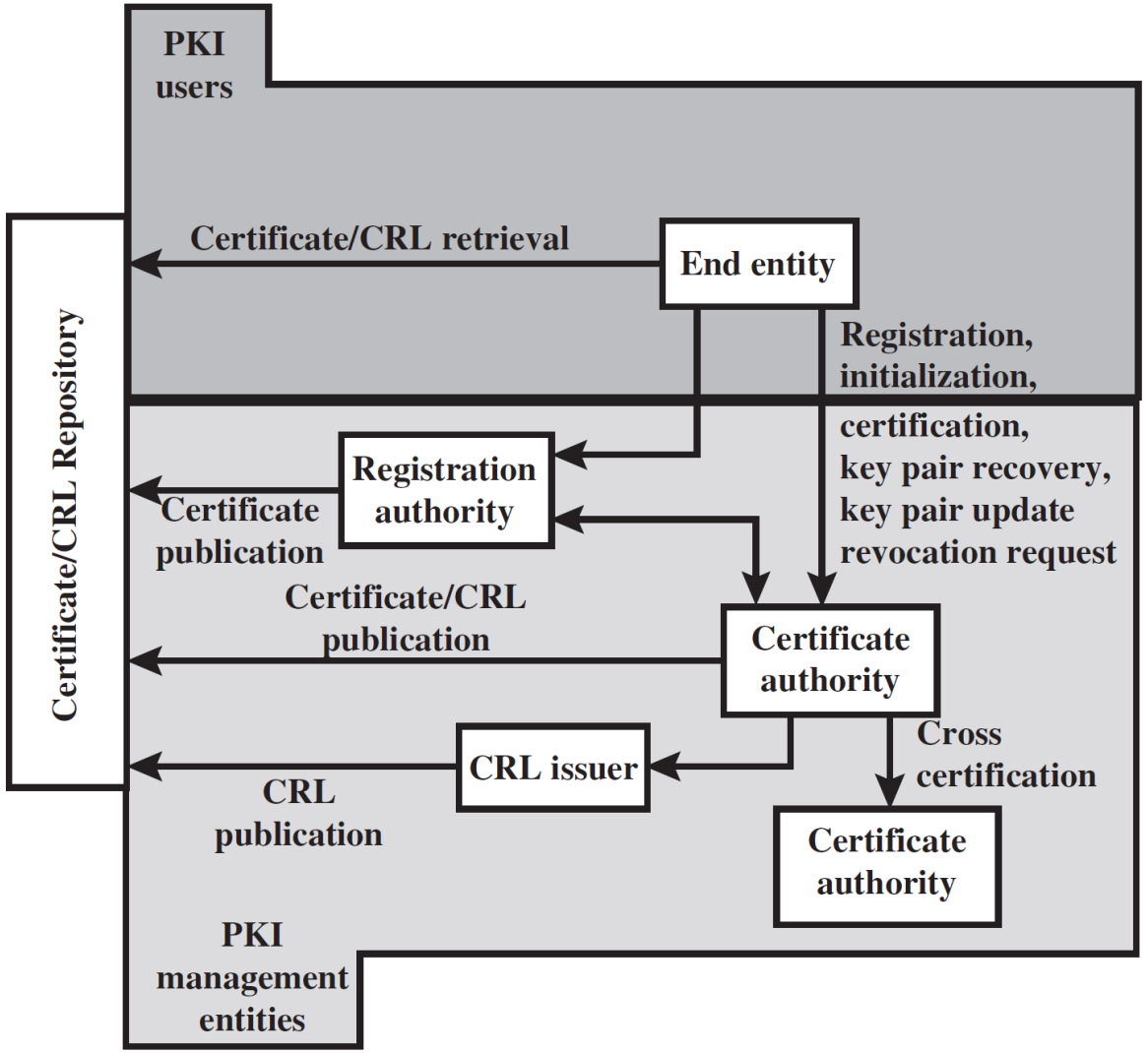
Chain of trust

- To be scalable, CAs are **hierarchically** organized in tree structures where each CA is certified by its parent CA, up to a **root CA**.
- A **root certificate** is an unsigned or a self-signed public key certificate that identifies the root CA.
- Root CAs are preloaded in browsers.
- Certificates are verified using a **chain of trust**. The trust anchor for the certificate is the root CA.



Public Key Infrastructure (PKI)

- A PKI is the set of hardware, software, people, policies, and procedures needed to create, manage, store, distribute, and revoke digital certificates based on asymmetric cryptography.
- The CA is the only mandatory component. Some of its functions can be delegated to Registration Authorities (RA) and Certificate Revocation Lists (CRL) issuers.



Wired and wireless link layer security

Part 4

Objectives

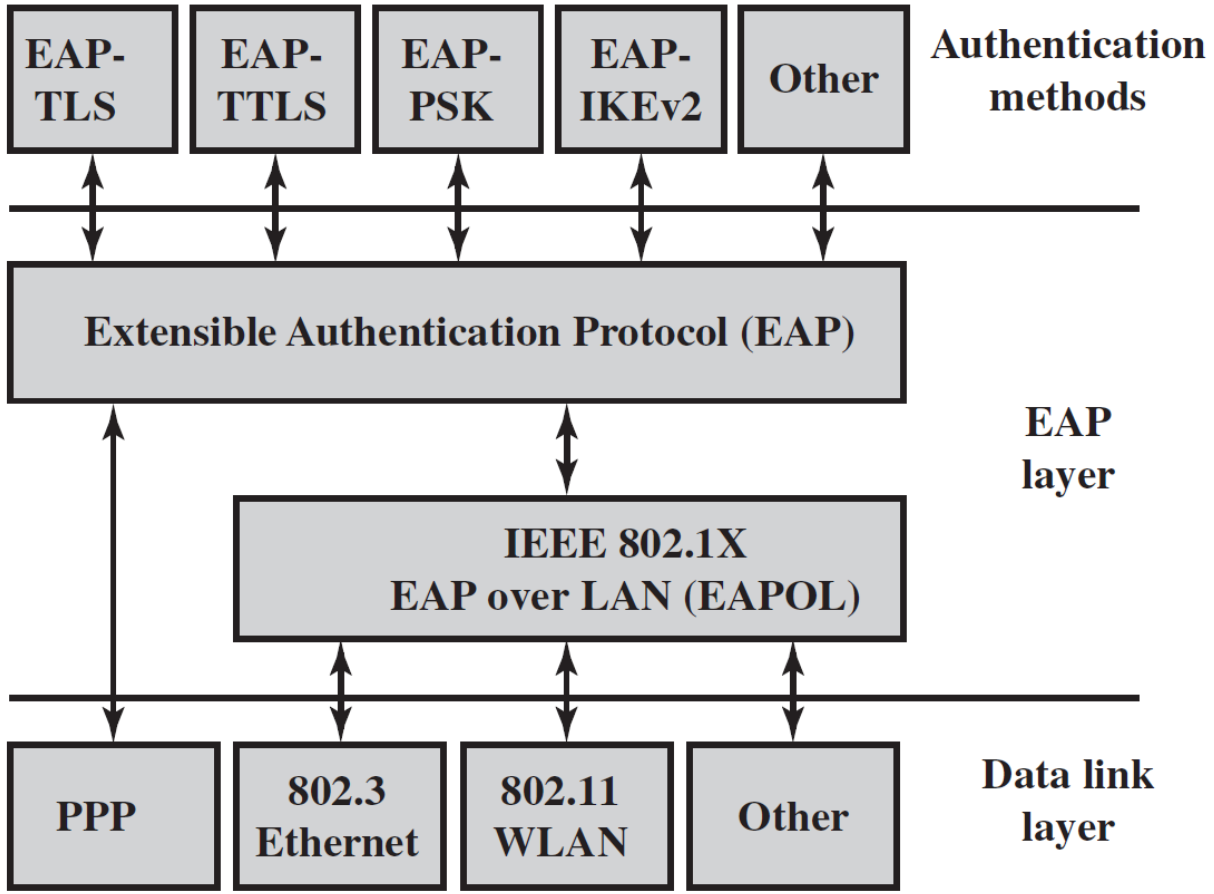
- Define network access control
- Understand Extensible Authentication Protocol (EAP)
- Understand port based access control (IEEE 802.1X)
- Understand WiFi security (IEEE 802.11i)
- Define firewalls

Network access control (NAC)

- Network access control (NAC) is an umbrella term for managing access to a network. NAC authenticates users logging into the network and determines what data they can access and actions they can perform.
- NAC systems deal with three categories of components:
  - **Access requestor** (AR): the AR is the node that is attempting to access the network and may be any device that is managed by the NAC system. ARs are also referred to as supplicants, or simply, clients.
  - **Policy server**: Based on the AR's posture and an enterprise's defined policy, the policy server determines what access should be granted.
  - **Network access server** (NAS): The NAS functions as an access control point for users in remote locations connecting to an enterprise's internal network. Also called a media gateway, a remote access server (RAS), or a policy server, an NAS may include its own authentication services or rely on a separate authentication service from the policy server.

Extensible Authentication Protocol (EAP)

- The Extensible Authentication Protocol (EAP), defined in RFC 3748, acts as a framework for network access and authentication protocols.
- EAP provides a set of protocol messages that can encapsulate various authentication methods to be used between a client and an authentication server.
- EAP can operate over a variety of data link layer protocols, and can accommodate their authentication needs.



Support for multiple authentication methods

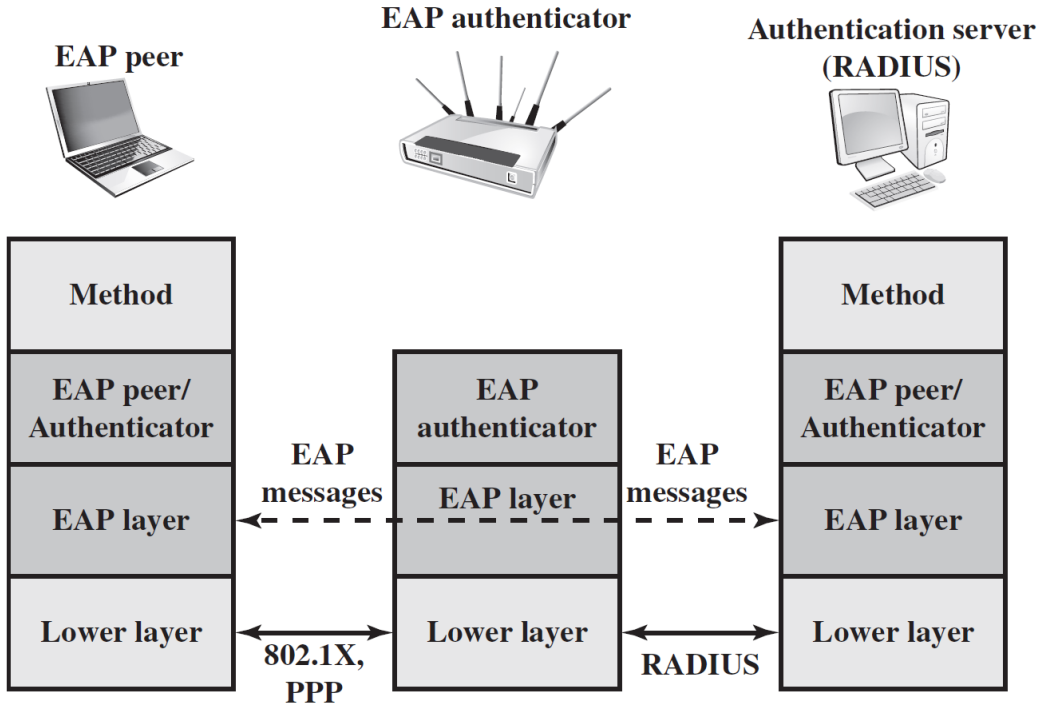
- EAP supports multiple authentication methods hence the term extensible.
- EAP provides a generic transport service for the exchange of authentication information between a client system and an authentication server.
- The basic EAP transport service is extended by using a specific authentication protocol, or method, that is installed in both the EAP client and the authentication server.

Common EAP Methods

- EAP-TLS (EAP Transport Layer Security, RFC 5216) defines how the TLS protocol can be encapsulated in EAP messages. EAP-TLS uses the handshake protocol in TLS, not its encryption method. Client and server authenticate each other using digital certificates. Client generates a pre-master secret key by encrypting a random number with the server's public key and sends it to the server. Both client and server use the pre-master to generate the same secret key.
- EAP-TTLS (EAP Tunneled TLS, RFC 5281) is like EAP-TLS, except only the server has a certificate to authenticate itself to the client first. As in EAP-TLS, a secure connection (the "tunnel") is established with secret keys, but that connection is used to continue the authentication process by authenticating the client and possibly the server again using any EAP method or legacy method such as PAP (Password Authentication Protocol) and CHAP (Challenge-Handshake Authentication Protocol).
- EAP-GPSK (EAP Generalized Pre-Shared Key, RFC 5433) is an EAP method for mutual authentication and session key derivation using a Pre-Shared Key (PSK). EAP-GPSK specifies an EAP method based on pre-shared keys and employs secret key-based cryptographic algorithms. Hence, this method is efficient in terms of message flows and computational costs, but requires the existence of pre-shared keys between each peer and EAP server. The set up of these pairwise secret keys is part of the peer registration. EAP-GPSK does not require any public-key cryptography.
- EAP-IKEv2, defined in RFC 5106, is based on the Internet Key Exchange protocol version 2 (IKEv2). It supports mutual authentication and session key establishment using a variety of methods.

EAP components

- **EAP peer**: Client computer that is attempting to access a network.
- **EAP authenticator**: An access point or NAS that requires EAP authentication prior to granting access to a network.
- **Authentication server**: A server computer that negotiates the use of a specific EAP method with an EAP peer, validates the EAP peer's credentials, and authorizes access to the network. Typically, the authentication server is a Remote Authentication Dial-In User Service (RADIUS) server.



EAP success combination

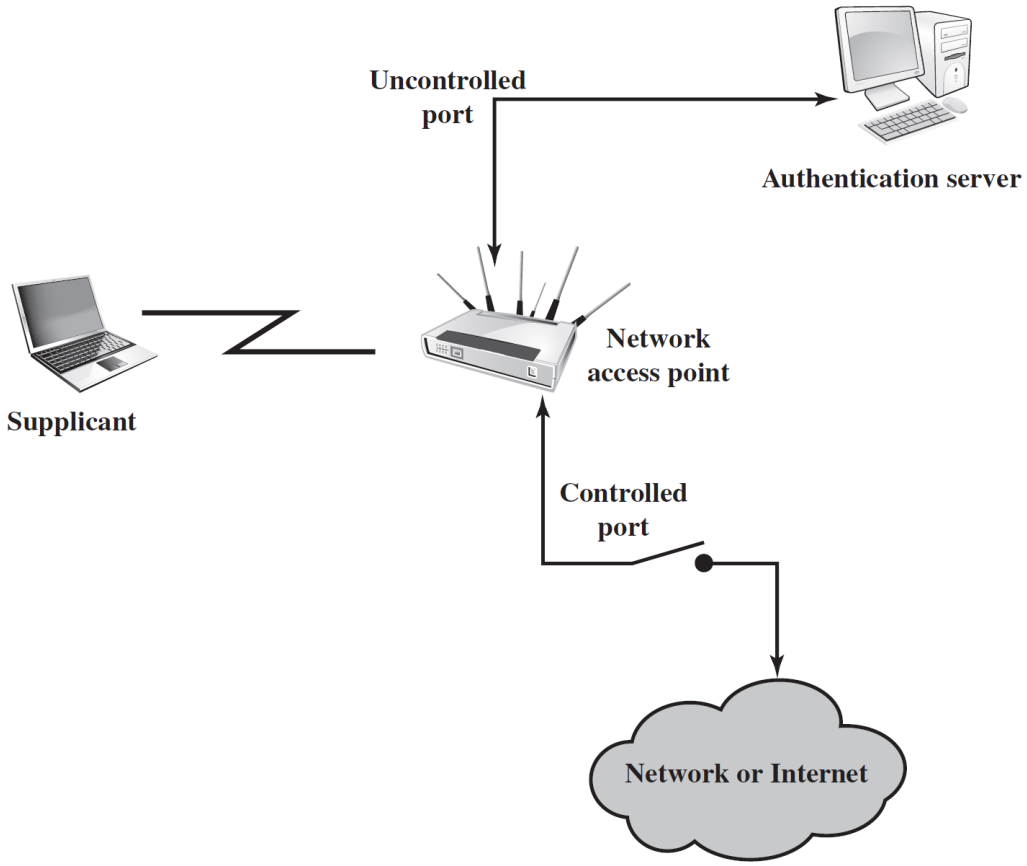
- The authentication information and authentication protocol information are carried in EAP messages whatever the method used for authentication.
- A successful authentication is an exchange of EAP messages, as a result of which the authenticator decides to allow access to the peer, and the peer decides to use this access.
- The peer may successfully authenticate to the authenticator, but authorization/access may be denied by the authenticator due to policy reasons.

EAP authentication exchange

- The authentication server functions as a backend server that can authenticate peers as a service to a number of EAP authenticators. The EAP authenticator then makes the decision of whether to grant access or not. This is referred to as the EAP **passthrough mode**.
- Less commonly, the EAP authenticator also takes over the role of the authentication server.
- The EAP authentication exchange proceeds as follows:
    1. A lower-level protocol, such as PPP or IEEE 802.1X, is used to connect to the EAP **authenticator**. The software entity in the EAP peer that operates at this level is referred to as the **supplicant**.
    2. The authenticator sends a **Request** to the peer to request an identity, and the peer sends a **Response** with the identity information. This is followed by a sequence of Requests by the authenticator and Responses by the peer for the exchange of authentication information. The information exchanged and the number of Request–Response exchanges needed depend on the authentication method.
    3. The conversation continues until either (1) the authenticator determines that it cannot authenticate the peer and transmits an EAP **Failure** or (2) the authenticator determines that successful authentication has occurred and transmits an EAP **Success**.

IEEE 802.1X port based access control

- IEEE 802.1X Port-Based Network Access Control was designed to provide access control functions for LANs
- Until the Authentication Server (AS) authenticates a supplicant (using an authentication protocol), the authenticator only passes control and authentication messages between the supplicant and the AS (i.e., the 802.1X control channel is unblocked).
- Once a supplicant is authenticated and keys are provided, the authenticator can forward data from the supplicant (i.e., the 802.11 data channel is unblocked).
- Ports are logical entities defined within the authenticator and refer to physical network connections. Each logical port is mapped to one of these two types of physical ports:
  - An uncontrolled port allows the exchange of data between the supplicant and the AS, regardless of the authentication state of the supplicant.
  - A controlled port allows the exchange of data between a supplicant and other systems on the network only if the current state of the supplicant authorizes such an exchange.



EAP over LAN (EAPOL)

- The essential element defined in 802.1X is the EAPOL (EAP over LAN) protocol which operates over LANs (such as Ethernet or Wi-Fi), enables a supplicant to communicate with an authenticator and supports the exchange of EAP packets for authentication.
- When the supplicant first connects to the LAN, it sends an EAPOL-Start packet to a special group-multicast address reserved for IEEE 802.1X authenticators, to determine whether an authenticator is present and let it know that the supplicant is ready.
- In wired LANs, the authenticator will already be notified that a new device has connected from some hardware notification. In this case the authenticator may preempt the EAPOL-Start message with its own message.
- In either case, the authenticator sends an EAP-Request Identity message encapsulated in an EAPOL-EAP packet. The EAPOL-EAP is the EAPOL frame type used for transporting EAP packets.

Wireless network threats

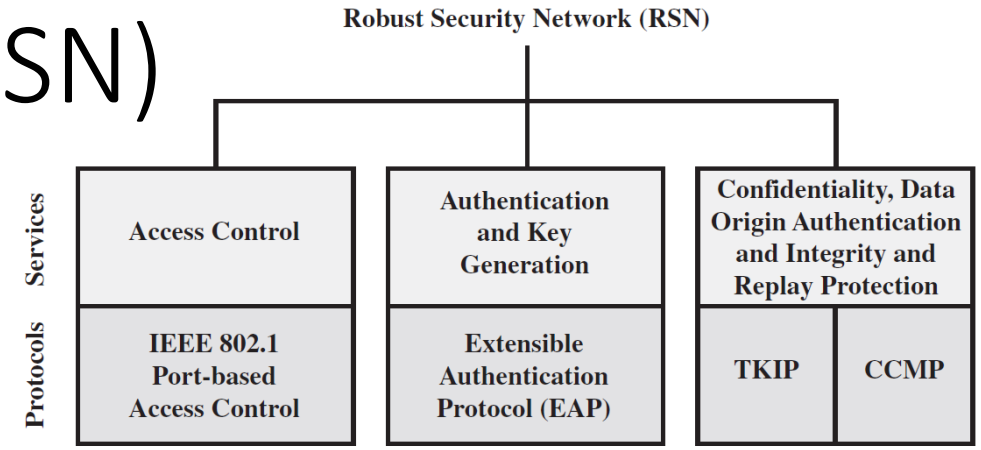
- **Malicious association** occurs when a wireless device is configured to appear to be a legitimate access point, enabling the operator to steal passwords from legitimate users and then penetrate a wired network through a legitimate wireless access point.
- **Ad hoc networks** are peer-to-peer networks between wireless computers with no access point between them. Such networks can pose a security threat due to a lack of a central point of control.
- **Identity theft** (MAC spoofing) occurs when an attacker is able to eavesdrop on network traffic and identify the MAC address of a computer with network privileges.
- A **man-in-the middle** attack involves persuading a user and an access point to believe that they are talking to each other when in fact the communication is going through an intermediate attacking device.
- A **denial of service** (DoS) attack occurs when an attacker continually bombards a wireless access point or some other accessible wireless port with various protocol messages designed to consume system resources. The wireless environment lends itself to this type of attack, because it is so easy for the attacker to direct multiple wireless messages at the target.
- A **network injection** attack targets wireless access points that are exposed to nonfiltered network traffic, such as routing protocol messages or network management messages. An example of such an attack is one in which bogus reconfiguration commands are used to affect routers and switches to degrade network performance.

IEEE 802.11i Wireless LAN security

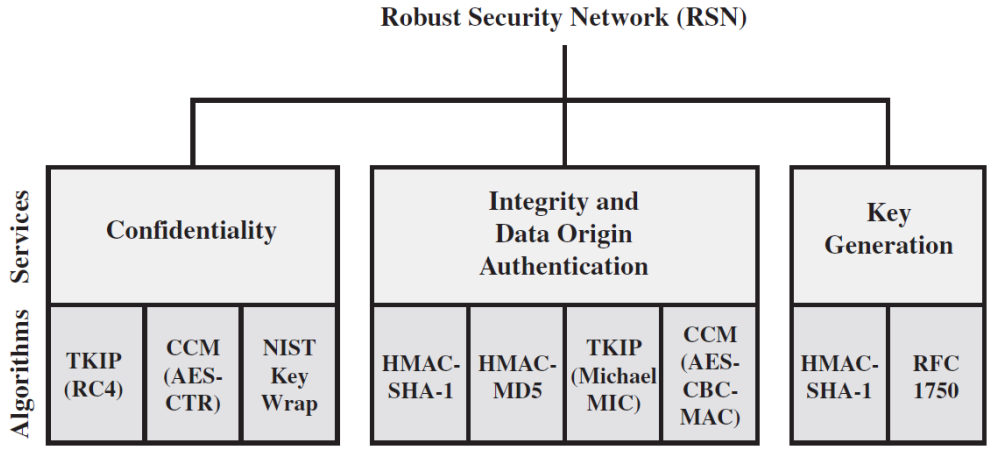
- The 802.11 standard initially defined the Wired Equivalent Privacy (WEP) algorithm which contained major weaknesses.
- The early state of the 802.11i standard was used to define the **Wi-Fi Protected Access** (WPA) certification.
- The final 802.11i standard known as **Robust Security Network** (RSN) is used by the Wi-Fi Alliance to certify as WPA2, vendors in compliance with the full 802.11i specification.
- 802.11i security is concerned only with secure communication between the client station (STA) and its access point (AP).

Robust Security Network (RSN)

- The 802.11i RSN security specification defines the following services:
  - **Authentication**: An exchange between a user and an AS provides mutual authentication and generates temporary keys to be used between the STA and the AP.
  - **Access control**: enforces the use of the authentication function, routes the messages properly, and facilitates key exchange.
  - **Privacy with message integrity**: MAC layer data are encrypted along with a message integrity code that ensures that the data have not been altered.



(a) Services and protocols

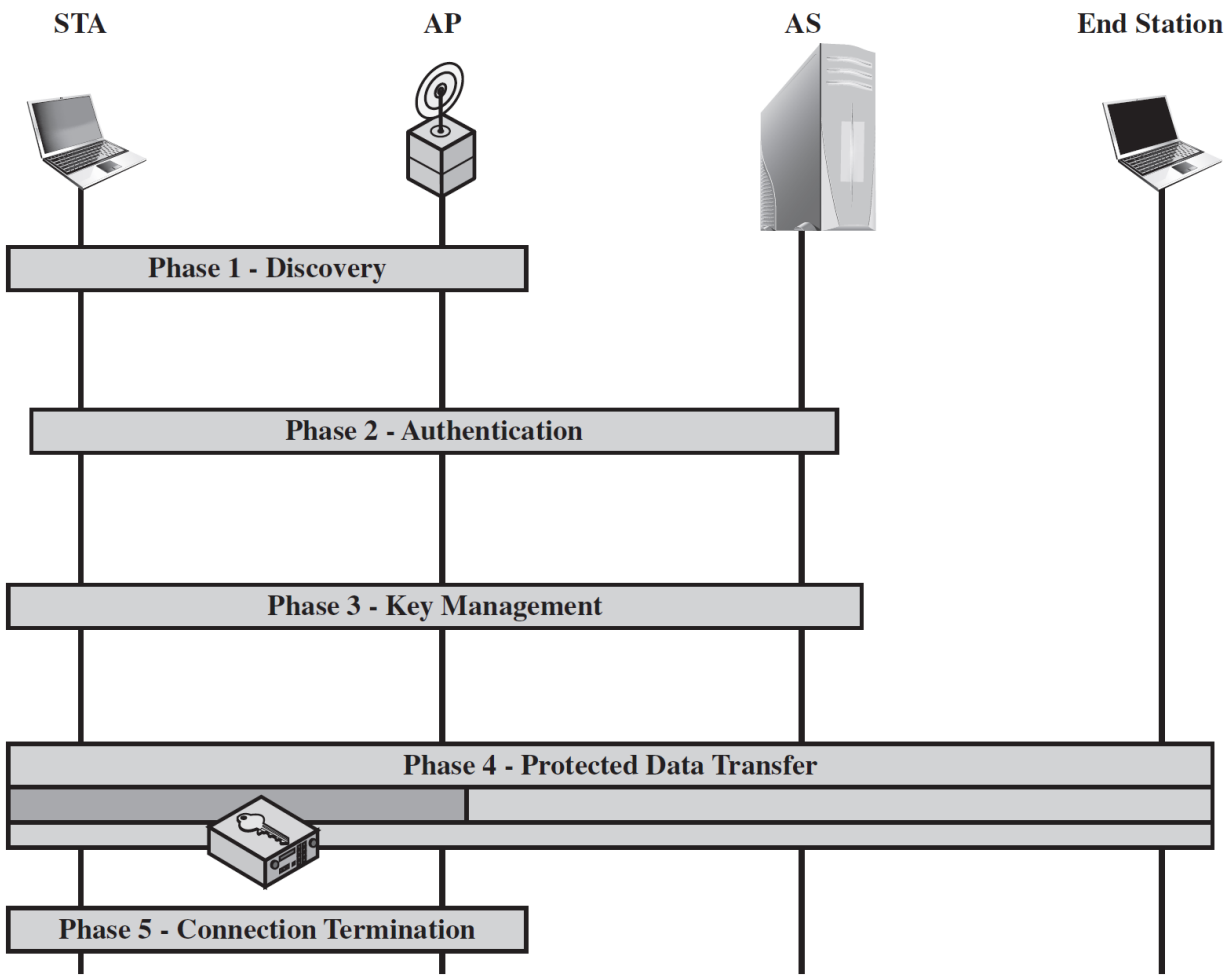


(b) Cryptographic algorithms

CBC-MAC = Cipher Block Chaining Message Authentication Code (MAC)
CCM = Counter Mode with Cipher Block Chaining Message Authentication Code
CCMP = Counter Mode with Cipher Block Chaining MAC Protocol
TKIP = Temporal Key Integrity Protocol

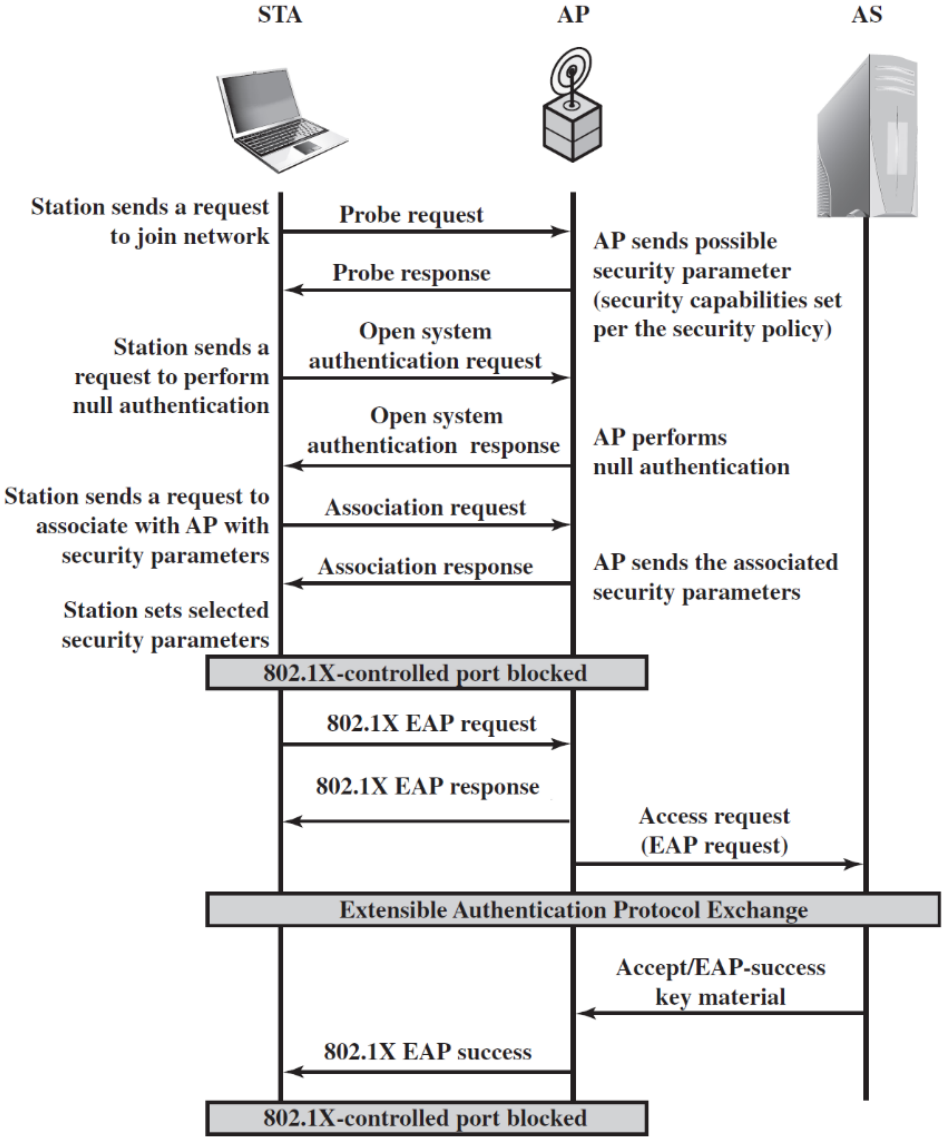
802.11i five phases of operation

1. **Discovery:** An AP advertises its IEEE 802.11i security policy. The STA uses beacons to identify an AP for a WLAN with which it wishes to communicate. The STA then associates with the AP.
2. **Authentication:** the STA and AS prove their identities to each other. The AP blocks non-authentication traffic between the STA and AS until the authentication transaction is successful. The AP does not participate in the authentication transaction other than forwarding traffic between the STA and AS.
3. **Key generation and distribution: t**he AP and the STA perform several operations that cause cryptographic keys to be generated and placed on the AP and the STA. Frames are exchanged between the AP and STA only.
4. **Protected data transfer:** frames are exchanged between the STA and the end station through the AP. As denoted by the shading and the encryption icon, secure data transfer occurs only between the STA and the AP, not end-to-end.
5. **Connection termination:** the AP and STA exchange frames to tear down the secure connection and the connection is restored to the original state.



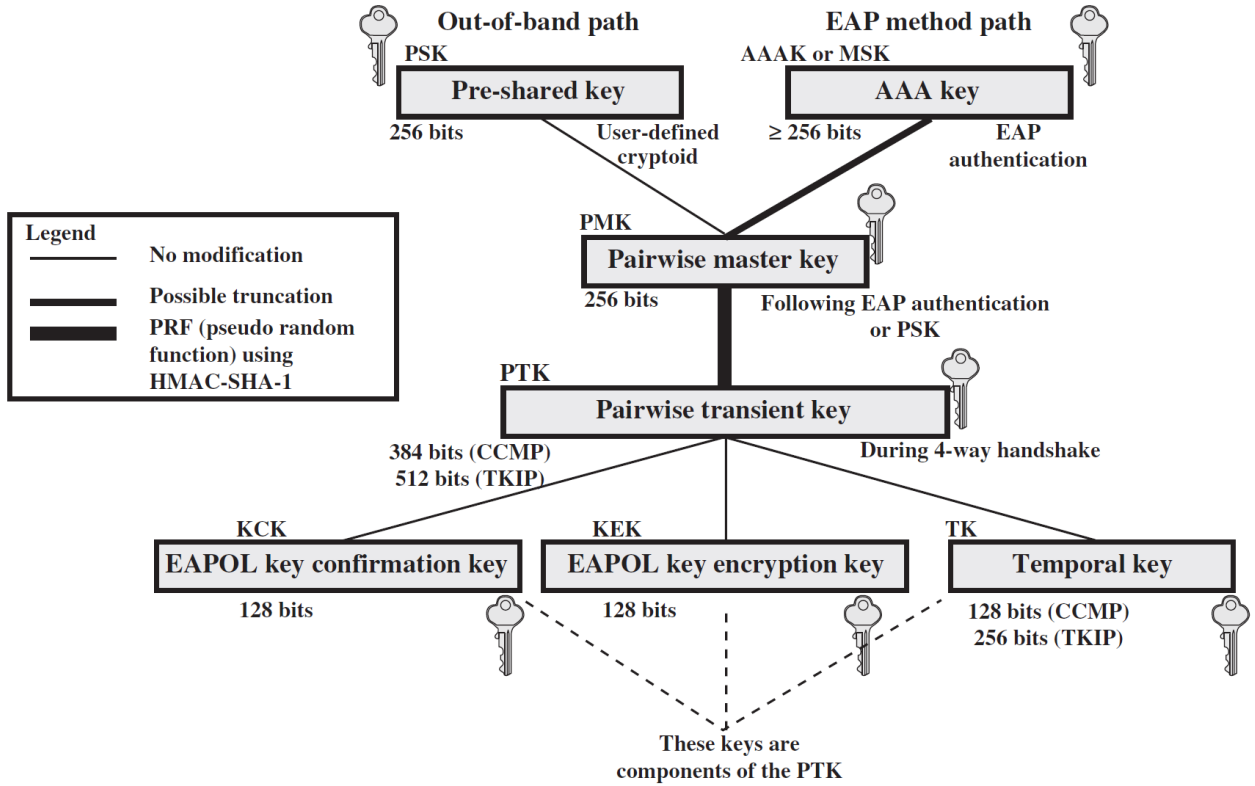
Capability Discovery and Authentication

- **Discovery** (1st phase)
  - **Probe**: a STA discovers available APs and corresponding security capabilities by either passively monitoring the **Beacon** frames or actively probing every channel.
  - **Open** authentication: the two devices (STA and AP) simply exchange identifiers (for backward compatibility).
  - **Association**: the STA sends an Association Request frame to the AP. In this frame, the STA specifies one set of matching capabilities (one authentication and key management suite, one pairwise cipher suite, and one group-key cipher suite) from among those advertised by the AP. If there is no match between them, the AP refuses the Association Request.
- **Authentication** (2nd phase)
  - The STA sends a request to its associated AP for connection to the AS.
  - Typically, the message flow between STA and AP employs the EAP over LAN (EAPOL) protocol, and the message flow between the AP and AS uses the Remote Authentication Dial In User Service (RADIUS) protocol.
  - Once authentication is established, the AS generates a **master session key** (MSK) and sends it to the STA. All the cryptographic keys needed by the STA for secure communication with its AP are generated from this MSK.
  - The AP controlled port remains blocked until the temporal keys are installed in the STA and AP, which occurs during the 4-Way Handshake (see further).



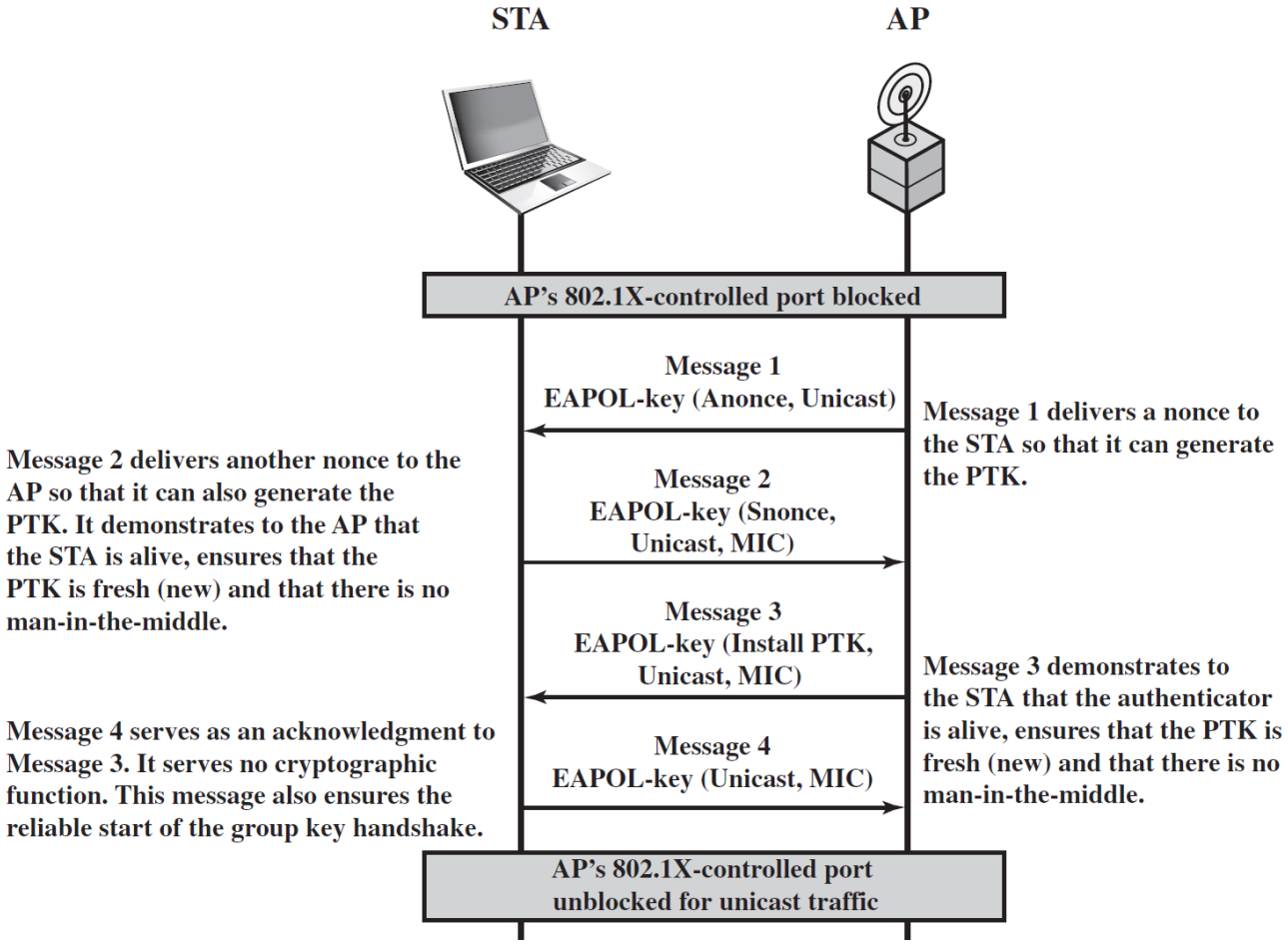
IEEE 802.11i Key Hierarchies

- Pairwise keys are used for communication between a pair of devices, typically between an STA and an AP. These keys form a hierarchy beginning with a **master key** from which other keys are derived dynamically and used for a limited period of time.
- At the top level are two possibilities:
  - A **pre-shared key** (PSK) is a secret key shared by the AP and a STA and manually installed.
  - A **master session key** (MSK), aka AAAK, which is generated during the authentication phase.
- All the other keys derived from this master key are also unique between an AP and an STA. Thus, each STA, at any time, has one set of keys, as depicted in the Figure.
- To derive the PTK, the HMAC-SHA-1 function is applied to the PMK, the MAC addresses of the STA and AP, and nonces generated when needed.
- Using the STA and AP addresses in the generation of the PTK provides protection against session hijacking and impersonation while using nonces provides additional random keying material.
- The PTK is divided into 3 subkeys: the Temporal Key (TK) provides the actual protection for user traffic.



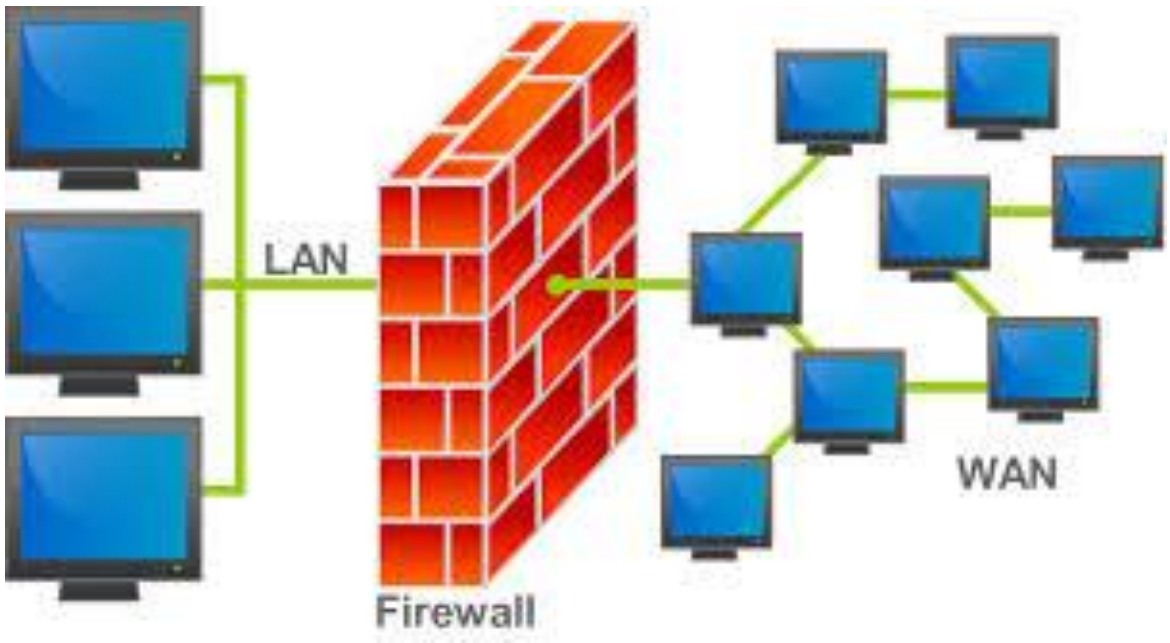
Key Management and Data Transfer

- This exchange is known as the 4-way handshake.
- The STA and AP use this handshake to confirm the existence of the PMK, verify the selection of the cipher suite, and derive a fresh PTK for the following data session.
- Two schemes for protected data transfer
  - TKIP: 'Michael' MIC + RC4
  - CCMP: CB-MAC + AES-CTR mode
- Note:
  - Message Integrity Code = Message Authentication Code (to avoid confusion with MAC = Medium Access Control)



Firewall

- A **firewall** is a network security system that monitors and controls the incoming and outgoing network traffic based on predetermined security rules.
- A firewall typically establishes a barrier between a trusted, secure internal network (LAN) and another outside network (WAN), such as the Internet, that is assumed not to be secure or trusted.
- Firewalls are often categorized as either network firewalls or host-based firewalls.
  - Network firewalls filter traffic between two or more networks; they are either software appliances running on general purpose hardware, or hardware-based firewall computer appliances.
  - Host-based firewalls provide a layer of software on one host that controls network traffic in and out of that single machine.
- Current firewalls can operate on any layer of the OSI model. They can inspect packet headers and data from the link layer up to the application layer.



Network, transport and application layers security

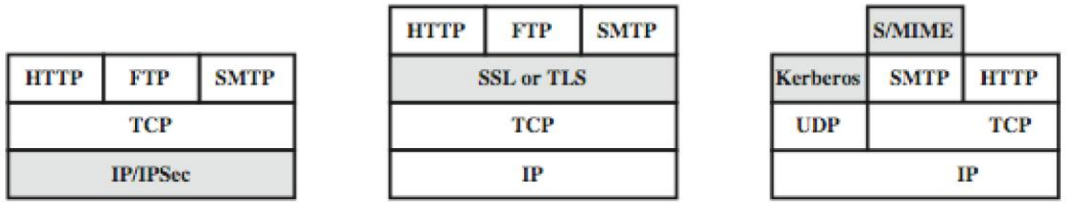
Part 5

Objectives

- Understand network layer security with IPsec as use case
- Understand transport layer security with SSL/TLS and SSH as use cases
- Understand application layer security with PGP and S/MIME as use cases

Security protocols in the TCP/IP stack

- A number of approaches to providing network security have been considered. They are similar in the services they provide and, to some extent, in the mechanisms that they use, but they differ with respect to their scope of applicability and their relative location within the TCP/IP protocol stack.
- The advantage of using IPsec is that it is transparent to end users and applications and provides a general-purpose solution. Furthermore, IPsec includes a filtering capability so that only selected traffic need incur the overhead of IPsec processing.
- Another solution is to implement security above TCP. The foremost example of this approach is the Secure Sockets Layer (SSL) and the follow-on Internet standard known as Transport Layer Security (TLS). SSL (or TLS) can be provided as part of the underlying protocol suite and therefore be transparent to applications or can be integrated in applications.
- Application-specific security services are embedded within the particular application. The advantage of this approach is that the service can be tailored to the specific needs of a given application.

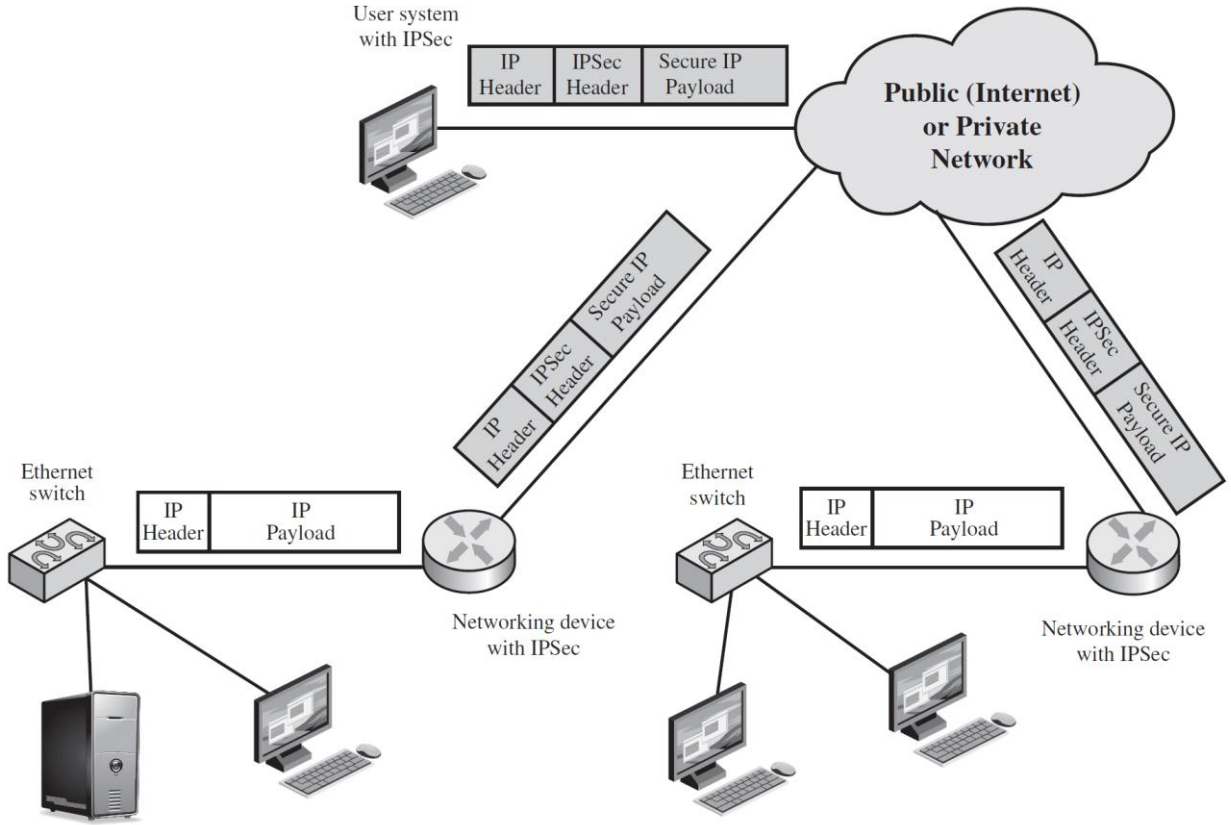


IP Security (IPsec)

- IPsec provides the capability to secure communications across a LAN, across private and public WANs, and across the Internet.
- The principal feature of IPsec is that it can encrypt and/or authenticate *all* traffic at the IP level. Thus, all distributed applications (including remote logon, client/server, e-mail, file transfer, Web access, and so on) can be secured.

IPsec Security Scenario

- An organization maintains LANs at dispersed locations. Nonsecure IP traffic is conducted on each LAN. For traffic offsite, through some sort of private or public WAN, IPsec protocols are used.
- These protocols operate in networking devices, such as a router or firewall, that connect each LAN to the outside world.
- The IPsec networking device will typically encrypt and compress all traffic going into the WAN and decrypt and decompress traffic coming from the WAN. These operations are transparent to workstations and servers on the LAN.
- Secure transmission is also possible with individual users who dial into the WAN. Such remote user workstations must implement the IPsec protocols to provide security.

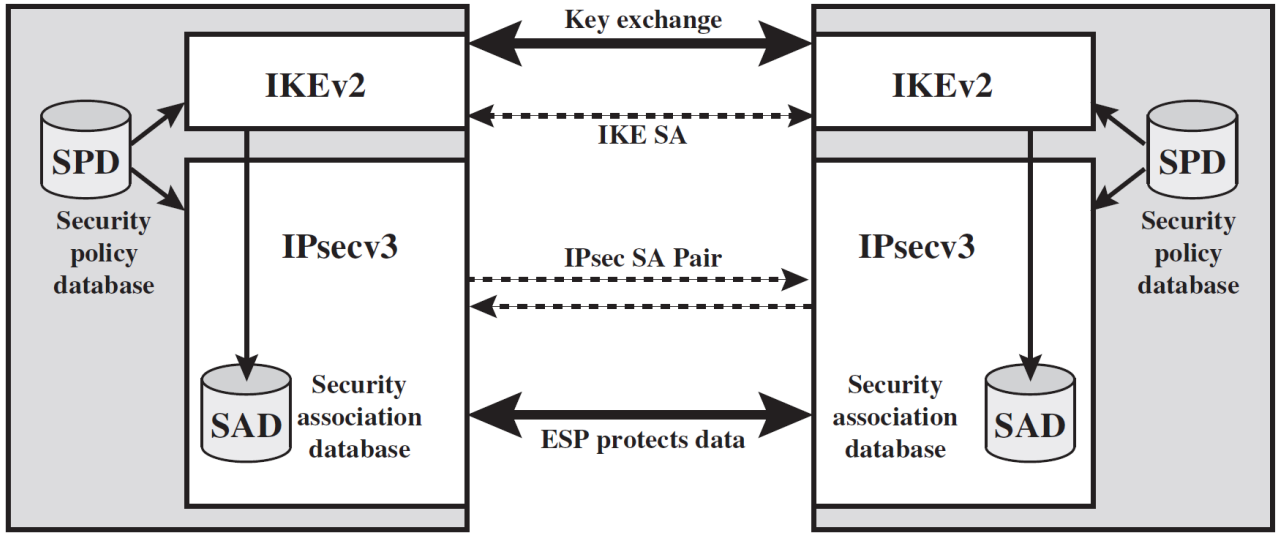


IPsec services

- IPsec provides security services at the IP layer by enabling a system to select required security protocols, determine the algorithm(s) to use for the service(s), and put in place any cryptographic keys required to provide the requested services.
- Two protocols are used for security and one for key management:
  - an authentication protocol called **Authentication Header** (AH)
  - a combined encryption/authentication protocol called **Encapsulating Security Payload** (ESP)
  - a key management protocol for the determination and distribution of secret keys called **Internet Key Exchange** (IKE)
- IPsec suite provides the following services:
  - Access control
  - Connectionless integrity
  - Data origin authentication
  - Rejection of replayed packets (a form of partial sequence integrity)
  - Confidentiality (encryption)
  - Limited traffic flow confidentiality

IP Security Policy

- Fundamental to the operation of IPsec is the concept of a **security policy** applied to each IP packet that transits from a source to a destination.
- IPsec policy is determined primarily by the interaction of two databases, the **security association database (SAD)** and the **security policy database (SPD)**.



Security Associations (SA)

- A **Security Association** (SA) is a one-way logical connection between a sender and a receiver that affords security services to the traffic carried on it.
- If a peer relationship is needed for two-way secure exchange, then two security associations are required.
- A security association is uniquely identified by three parameters:
  - **Security Parameters Index (SPI):** a 32-bit unsigned integer assigned to this SA and having local significance only. The SPI is carried in AH and ESP headers to enable the receiving system to select the SA under which a received packet will be processed.
  - **IP Destination Address:** this is the address of the destination endpoint of the SA, which may be an end-user system or a network system such as a firewall or router.
  - **Security Protocol Identifier:** this field from the outer IP header indicates whether the association is an AH or ESP security association.

Security Association Database (SAD)

- In each IPsec implementation, there is a Security Association Database that defines the parameters associated with each SA.
- A security association is normally defined by the following parameters in an SAD entry:
  - **Security Parameter Index**: A 32-bit value selected by the receiving end of an SA to uniquely identify the SA. In an SAD entry for an outbound SA, the SPI is used to construct the packet's AH or ESP header. In an SAD entry for an inbound SA, the SPI is used to map traffic to the appropriate SA.
  - **Sequence Number Counter**: A 32-bit value used to generate the Sequence Number field in AH or ESP headers.
  - **Sequence Counter Overflow**: A flag indicating whether overflow of the Sequence Number Counter should generate an auditable event and prevent further transmission of packets on this SA.
  - **Anti-Replay Window**: Used to determine whether an inbound AH or ESP packet is a replay.
  - **AH Information**: Authentication algorithm, keys, key lifetimes, and related parameters being used with AH.
  - **ESP Information**: Encryption and authentication algorithm, keys, initialization values, key lifetimes, and related parameters being used with ESP.
  - **Lifetime** of this Security Association: A time interval or byte count after which an SA must be replaced with a new SA (and new SPI) or terminated, plus an indication of which of these actions should occur.
  - **IPsec Protocol Mode**: Tunnel, transport, or wildcard.
  - **Path MTU**: Any observed path maximum transmission unit (maximum size of a packet that can be transmitted without fragmentation) and aging variables.
- The key management mechanism that is used to distribute keys is coupled to the authentication and privacy mechanisms only by way of the Security Parameters Index (SPI). Hence, authentication and privacy have been specified independent of any specific key management mechanism.

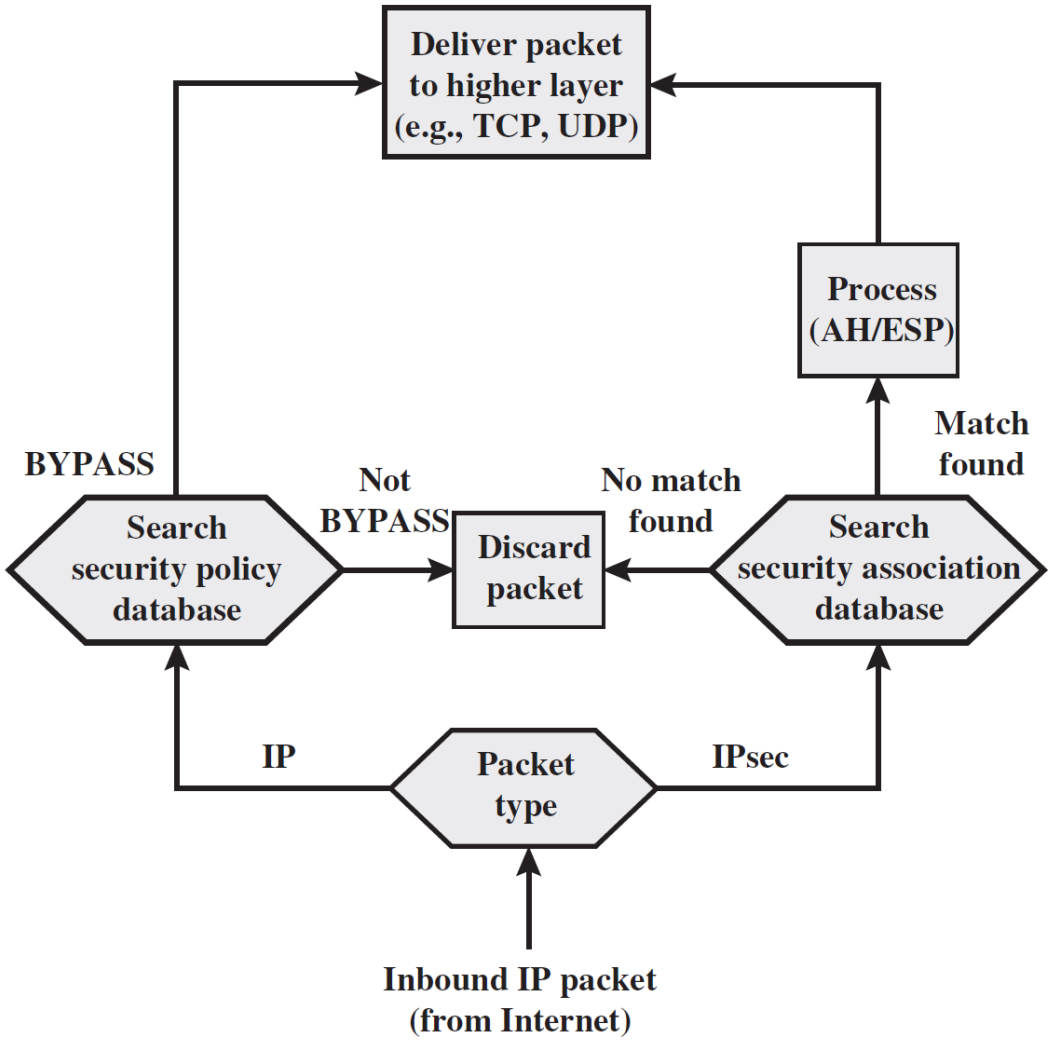
Security Policy Database

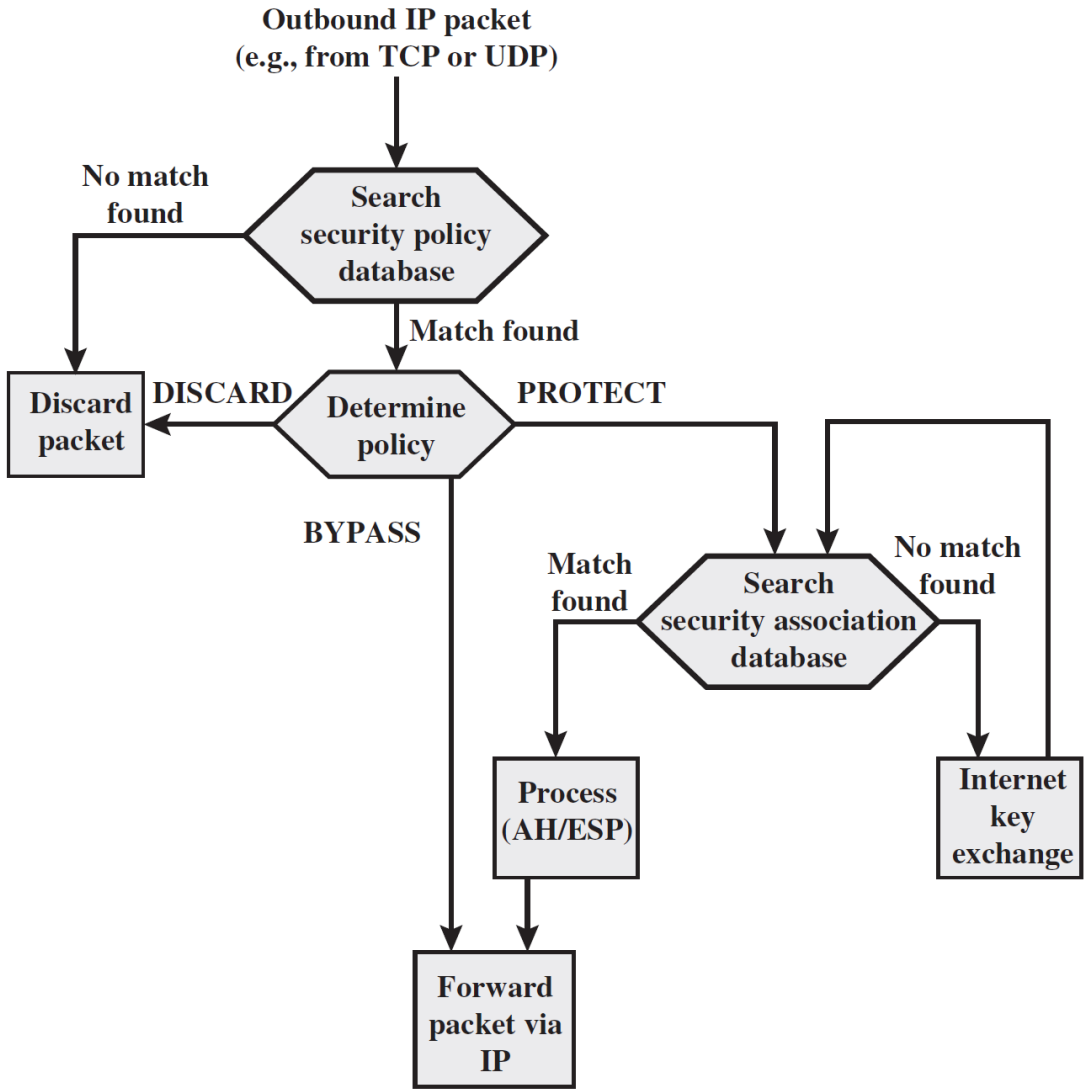
- The means by which IP traffic is related to specific SAs (or no SA in the case of traffic allowed to bypass IPsec) is the Security Policy Database (SPD).
- An SPD contains **entries**, each of which defines a subset of IP traffic and points to an SA for that traffic. In more complex environments, there may be multiple entries that potentially relate to a single SA or multiple SAs associated with a single SPD entry.
- Each SPD entry is defined by a set of IP and upper-layer protocol field values, called **selectors**. In effect, these selectors are used to filter outgoing traffic in order to map it into a particular SA

Protocol	Local IP	Port	Remote IP	Port	Action	Comment
UDP	1.2.3.101	500	*	500	BYPASS	IKE
ICMP	1.2.3.101	*	*	*	BYPASS	Error messages
*	1.2.3.101	*	1.2.3.0/24	*	PROTECT: ESP intransport-mode	Encrypt intranet traffic
TCP	1.2.3.101	*	1.2.4.10	80	PROTECT: ESP intransport-mode	Encrypt to server
TCP	1.2.3.101	*	1.2.4.10	443	BYPASS	TLS: avoid double encryption
*	1.2.3.101	*	1.2.4.0/24	*	DISCARD	Others in DMZ
*	1.2.3.101	*	*	*	BYPASS	Internet

Packet processing

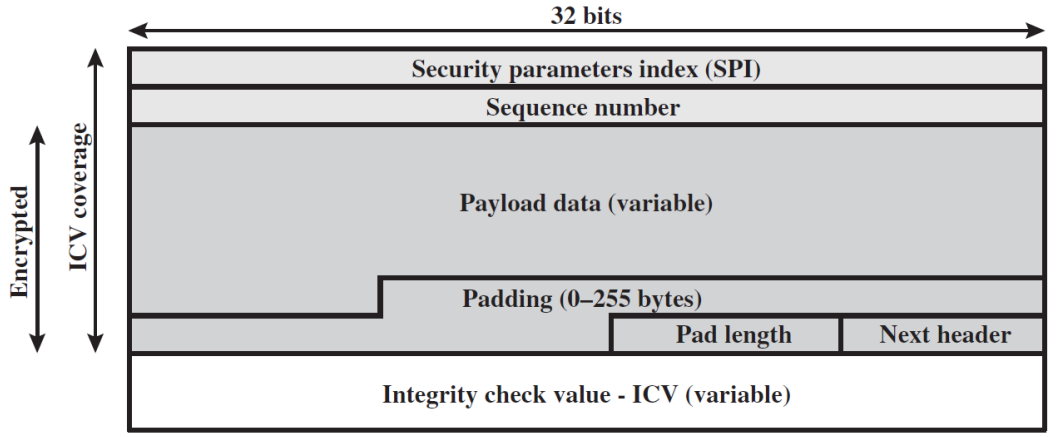
IPsec is executed on a **packet-by-packet** basis. When IPsec is implemented, each outbound IP packet is processed by the IPsec logic before transmission, and each inbound packet is processed by the IPsec logic after reception and before passing the packet contents on to the next higher layer (e.g., TCP or UDP).



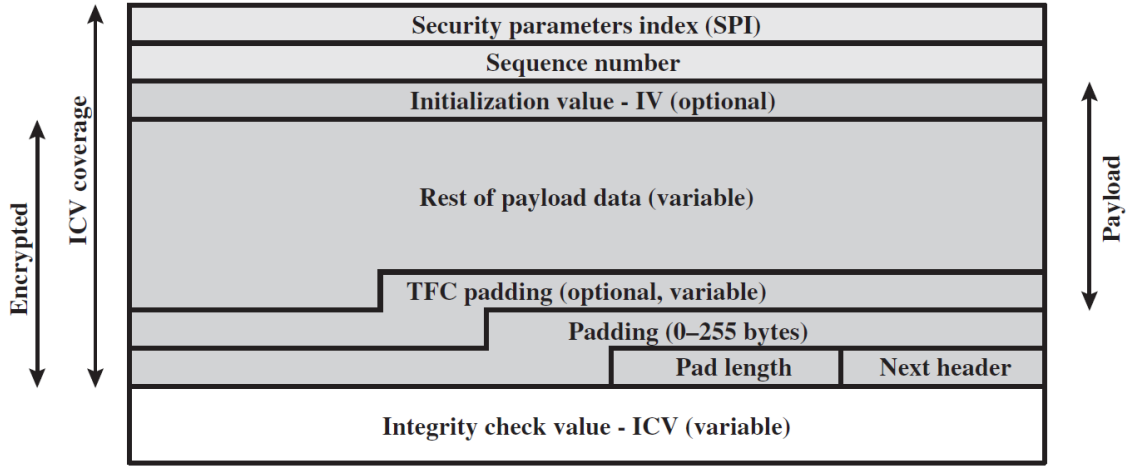


Encapsulating Security Payload (ESP)

- ESP can be used to provide confidentiality, data origin authentication, connectionless integrity, an anti-replay service (a form of partial sequence integrity), and (limited) traffic flow confidentiality.
- The set of services provided depends on options selected at the time of Security Association (SA) establishment and on the location of the implementation in the network topology.
- The **SPI** identifies a given **SA**, the **Sequence Number** provides an **anti-replay** function and the **next header** indicates the type of data being protected (i.e., upper-layer protocol).
- The **ICV field** is present only if the integrity service is selected.
  - The ICV is computed after the encryption is performed to facilitate rapid detection and rejection of replayed or bogus packets by the receiver prior to decrypting the packet. This allows for parallel decryption and integrity checking. Because the ICV is not protected by encryption, a keyed integrity algorithm must be employed to compute the ICV.
- Two additional fields may be present in the payload: an **initialization value** (IV), or *nonce*, is present if this is required by the encryption or authenticated encryption algorithm used for ESP. If tunnel mode is being used, then the IPsec implementation may add traffic flow confidentiality (**TFC**) **padding**.



(a) Top-level format of an ESP Packet



(b) Substructure of payload data

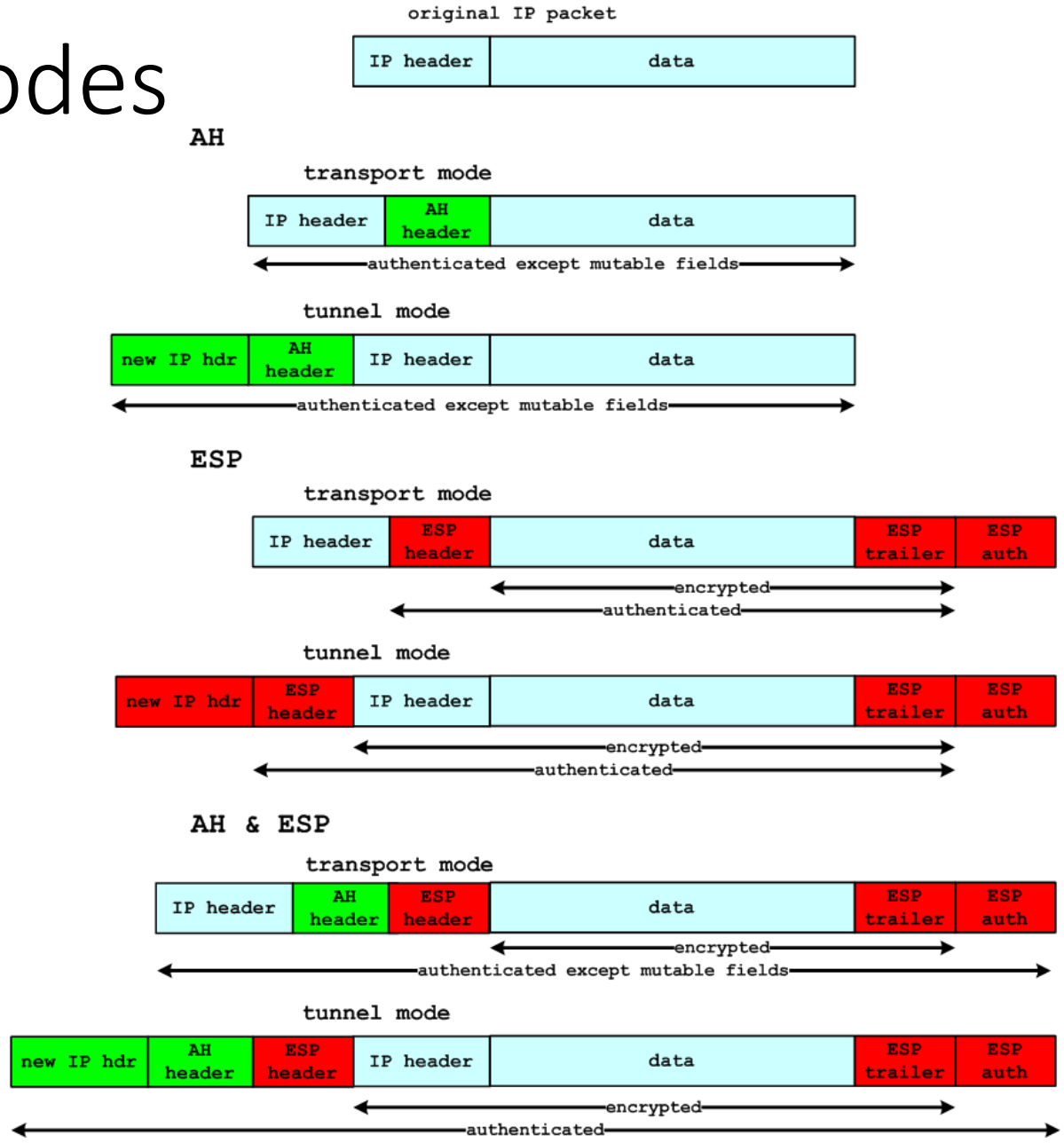
Authentication Header (AH)

- AH guarantees connectionless integrity and data origin authentication of IP packets.
- It can optionally protect against replay attacks by using the sliding window technique and discarding old packets.
- The header fields have the same meaning and usage than in ESP.

Authentication Header format																													
Offsets	$Octet_{16}$	0								1								2								3			
$Octet_{16}$	Bit_{10}	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
0	0	*Next Header*								*Payload Len*								*Reserved*															
4	32	*Security Parameters Index (SPI)*																															
8	64	*Sequence Number*																															
C	96	*Integrity Check Value (ICV)*																															
...	...	...																															

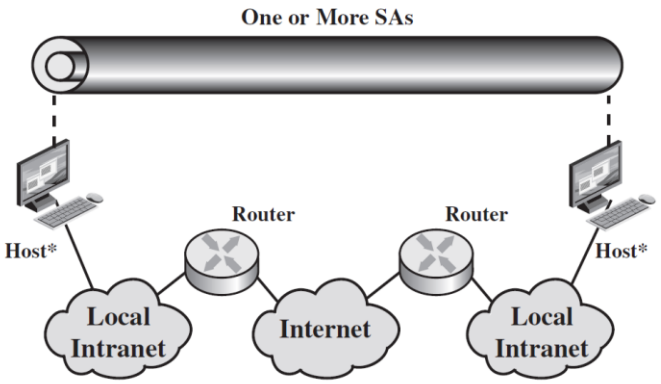
Transport and tunnel modes

- Both AH and ESP support two modes of operation:
  - **Transport** mode: only the payload of the IP packet is encrypted or authenticated.
    - The IP header is neither modified nor encrypted; however, when AH is used, the IP addresses cannot be modified by network address translation (NAT), as this always invalidates the hash value.
  - **Tunnel** mode: the entire IP packet is encrypted and authenticated. It is then encapsulated into a new IP packet with a new IP header. Tunnel mode is used to create virtual private networks for network-to-network communications and host-to-network communications

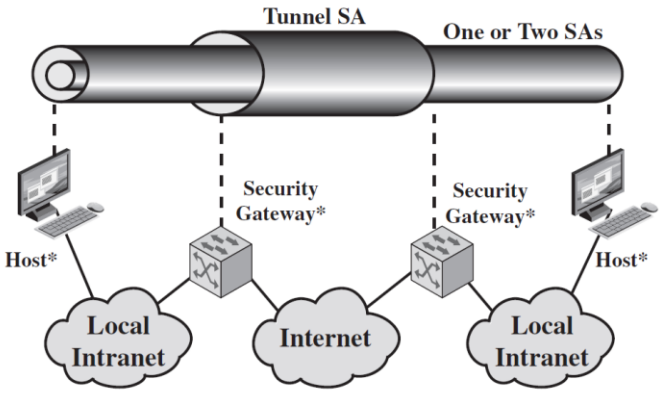


Basic Combinations of Security Associations

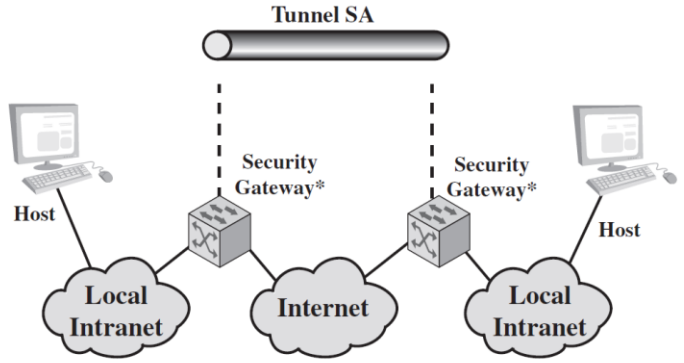
- Case 1. All security is provided between end systems that implement IPsec. For any two end systems to communicate via an SA, they must share the appropriate secret keys.
- Case 2. Security is provided only between gateways (routers, firewalls, etc.) and no hosts implement IPsec. This case illustrates simple virtual private network support.
- Case 3. This builds on case 2 by adding end-to-end security. The gateway-to-gateway tunnel provides authentication and/or confidentiality for all traffic between end systems. When the gateway-to-gateway tunnel is ESP, it also provides a limited form of traffic confidentiality. Individual hosts can implement any additional IPsec services required for given applications or given users by means of end-to-end SAs.
- Case 4. This provides support for a remote host that uses the Internet to reach an organization's firewall and then to gain access to some server or workstation behind the firewall. Only tunnel mode is required between the remote host and the firewall. As in case 1, one or two SAs may be used between the remote host and the local host.



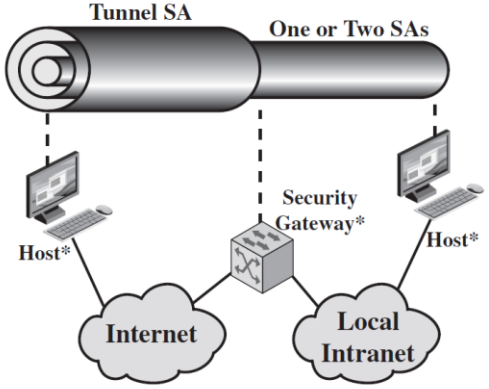
(a) Case 1



(c) Case 3



(b) Case 2



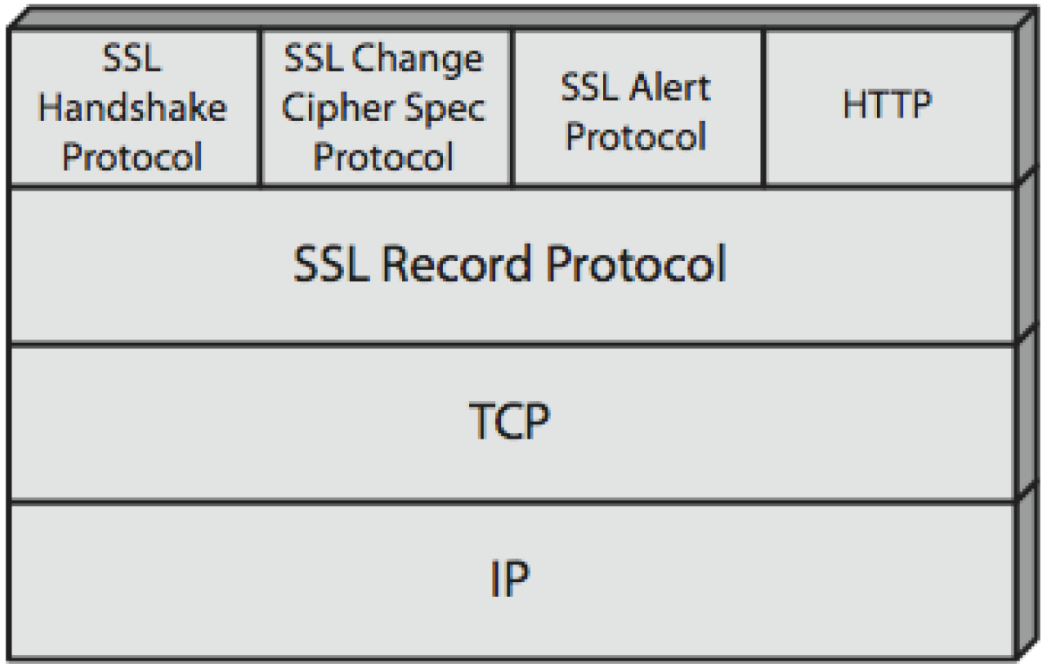
(d) Case 4

Secure Sockets Layer (SSL) / Transport Layer Security (TLS)

- SSL is a general-purpose service implemented as a set of protocols that rely on TCP.
- Two implementation choices:
  - SSL can be provided as part of the underlying protocol suite and therefore be transparent to applications.
  - Alternatively, SSL can be embedded in specific applications. For example, most browsers and Web servers have implemented SSL.
- TLS is an IETF standardization initiative whose goal is to produce an Internet standard version of SSL. TLS is defined as a Proposed Internet Standard in RFC 5246 which is very similar to SSLv3.

SSL/TLS protocol stack

- SSL is not a single protocol but rather two layers of protocols:
  - The SSL **Record Protocol** provides basic security services to various higher layer protocols. In particular, the Hypertext Transfer Protocol (HTTP), which provides the transfer service for Web client/server interaction, can operate on top of SSL.
  - Three higher-layer protocols are defined as part of SSL: the **Handshake Protocol**, The Chan**ge Cipher Spec Protocol**, and the **Alert Protocol**. These SSL-specific protocols are used in the management of SSL exchanges.

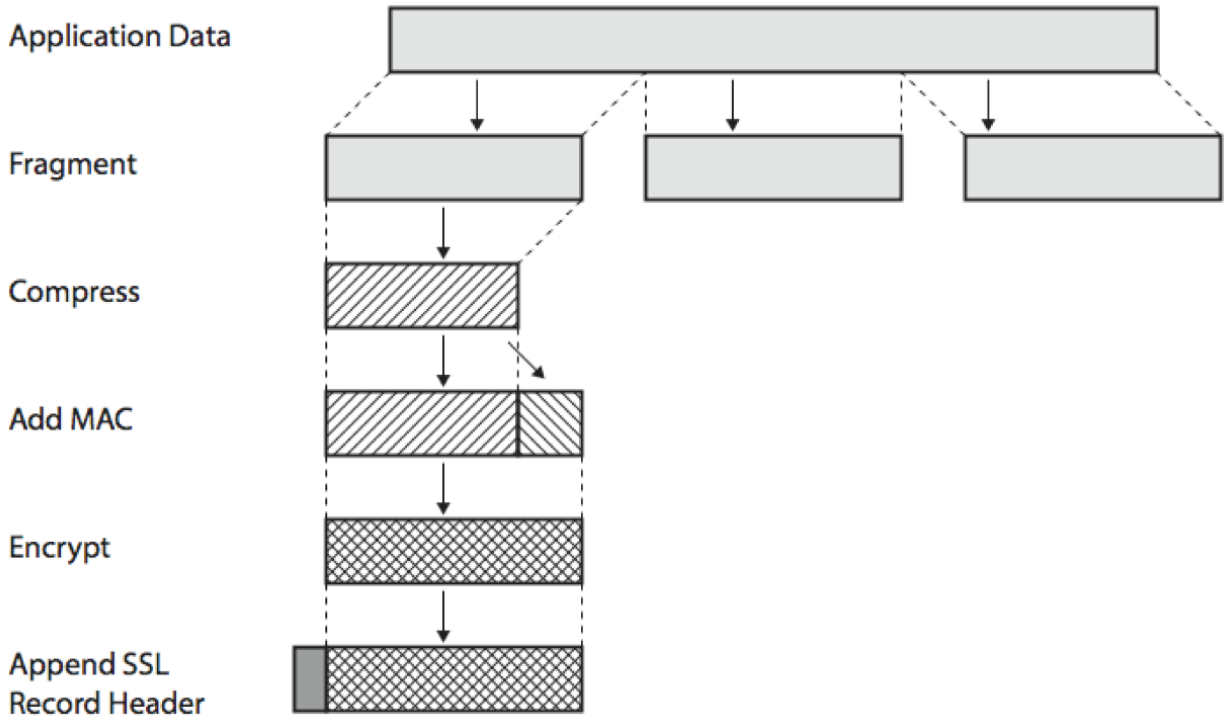


SSL/TLS connections and sessions

- **Connection**: a transport connection that provides a suitable type of service. For SSL, such connections are peer-to-peer relationships. The connections are transient. Every connection is associated with one session.
- **Session**: an SSL session is an association between a client and a server. Sessions are created by the Handshake Protocol. Sessions define a set of cryptographic security parameters which can be shared among multiple connections. Sessions are used to avoid the expensive negotiation of new security parameters for each connection.
- Between any pair of parties (applications such as HTTP client and server), there may be multiple secure connections. In theory, there may also be multiple simultaneous sessions between parties, but this feature is not used in practice.

SSL/TLS record protocol operation

- The Record Protocol proceeds as follows:
  1. The first step is fragmentation. Each upper-layer message is fragmented into blocks of 16384 bytes or less.
  2. Next, compression is optionally applied. It must be lossless and may not increase the content length by more than 1024 bytes.
     - In SSLv3 and the current version of TLS, no compression algorithm is specified, so the default compression algorithm is null.
  3. The next step in processing is to compute a MAC over the compressed data. For this purpose, a shared secret key is used.
     - The MAC is very similar to the HMAC algorithm. The difference is that the two pads are concatenated in SSLv3 and are XORed in HMAC (defined in RFC 2104).
  4. Next, the compressed message plus the MAC are encrypted using symmetric encryption. Encryption may not increase the content length by more than 1024 bytes. Block or stream ciphers are possible.
  5. The final step of processing is to prepare the header.
- Received data are decrypted, verified, decompressed, and reassembled before being delivered to applications.



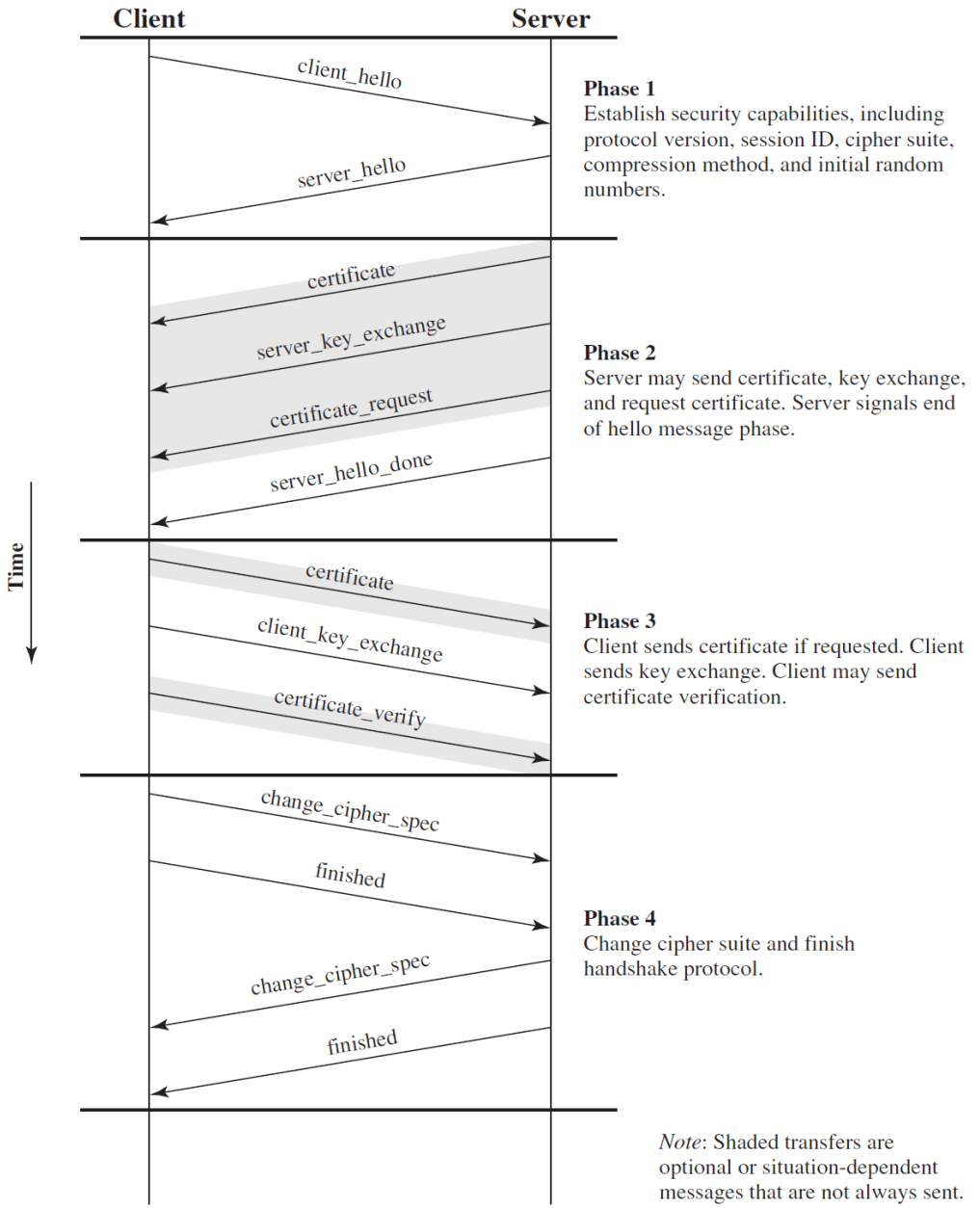
SSL/TLS header

- **Content Type** (8 bits): The higher-layer protocol used to process the enclosed fragment.
- Major **Version** (8 bits): Indicates major version of SSL in use. For SSLv3, the value is 3.
- Minor **Version** (8 bits): Indicates minor version in use. For SSLv3, the value is 0.
- **Compressed Length** (16 bits): The length in bytes of the plaintext fragment (or compressed fragment if compression is used). The maximum value is 2^14 + 2048.

+	Byte +0	Byte +1	Byte +2	Byte +3
Byte 0	Content type			
Bytes 1..4	Version		Length	
	(Major)	*(Minor)*	*(bits 15..8)*	*(bits 7..0)*
Bytes 5..(*m*-1)	Protocol message(s)			
Bytes *m*..(p-1)	MAC (optional)			
Bytes *p*..(q-1)	Padding (block ciphers only)			

SSL/TLS handshake protocol

- This protocol allows the server and client to authenticate each other and to negotiate an encryption and MAC algorithm and cryptographic keys to be used to protect data sent in an SSL record.
- The Handshake Protocol consists of a series of messages exchanged by the client and the server and is used before any application data is transmitted.



SSL/TLS supported key exchange methods

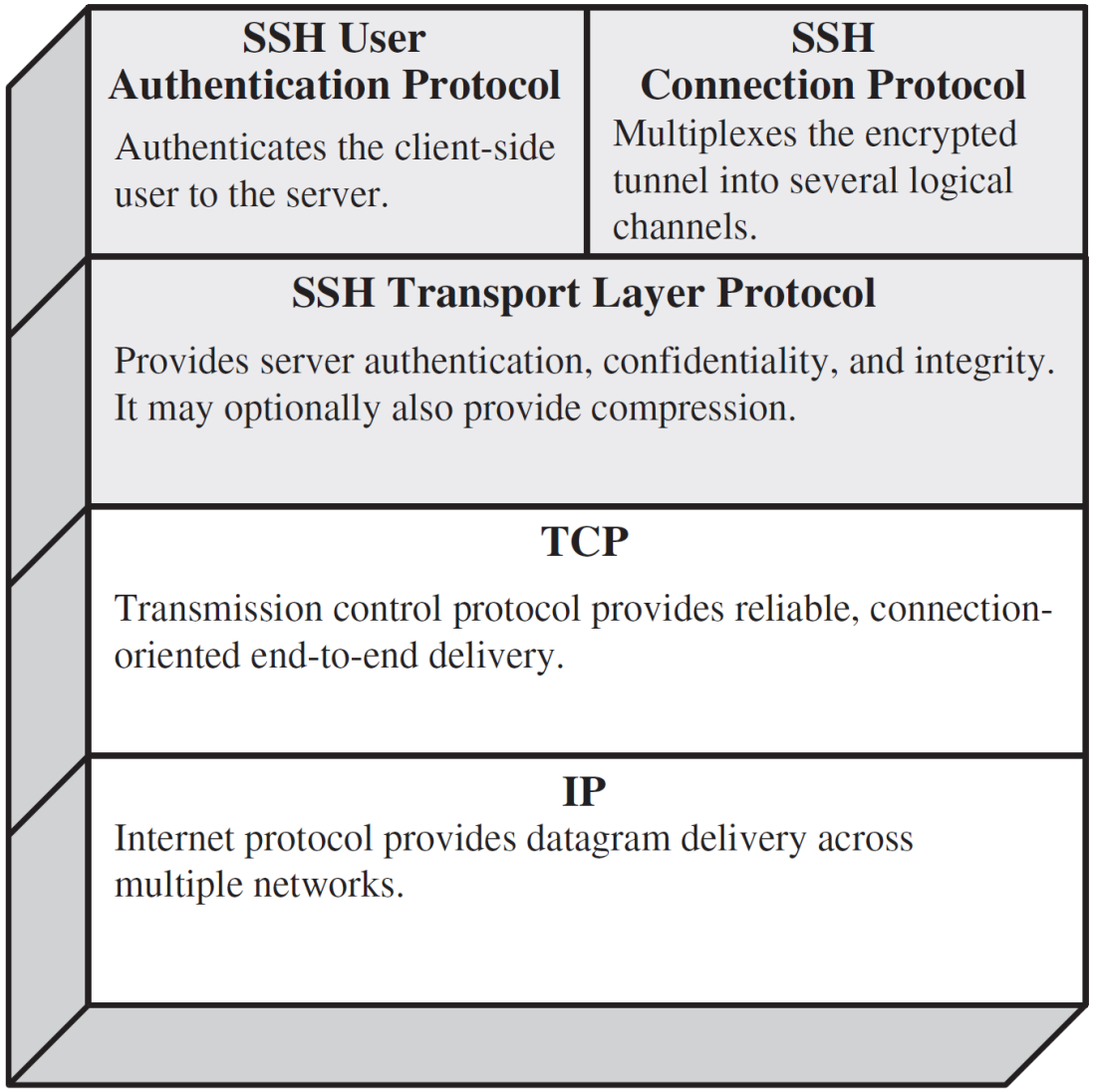
- **RSA**: The secret key is encrypted with the receiver's RSA public key. A public-key certificate for the receiver's key must be made available.
- **Fixed Diffie-Hellman**: This is a Diffie-Hellman key exchange in which the server's certificate contains the Diffie-Hellman public parameters signed by the certificate authority (CA). That is, the public-key certificate contains the Diffie-Hellman public-key parameters. The client provides its Diffie-Hellman public-key parameters either in a certificate, if client authentication is required, or in a key exchange message. This method results in a fixed secret key between two peers based on the Diffie-Hellman calculation using the fixed public keys.
- **Ephemeral Diffie-Hellman**: This technique is used to create ephemeral (one-time temporary) secret keys. In this case, the Diffie-Hellman public keys are exchanged, signed using the sender's private RSA or DSS key. The receiver can use the corresponding public key to verify the signature. Certificates are used to authenticate the public keys. This is the most secure of the three Diffie-Hellman options, because it results in a temporary, authenticated key.
- **Anonymous Diffie-Hellman**: The base Diffie-Hellman algorithm is used with no authentication. That is, each side sends its public Diffie-Hellman parameters to the other with no authentication. This approach is vulnerable to man-in-the-middle attacks.

Secure Shell (SSH)

- Secure Shell (SSH) is a protocol for secure network communications designed to be relatively simple and inexpensive to implement.
- SSH also provides a more general client/server capability and can be used for such network functions as file transfer and e-mail.
- SSH2 is documented as a proposed standard in IETF RFCs 4250 through 4256.
- SSH client and server applications are widely available for most operating systems.
- It has become the method of choice for remote login and X tunneling and is rapidly becoming one of the most pervasive applications for encryption technology outside of embedded systems.

SSH protocol stack

- SSH is organized as three protocols that typically run on top of TCP:
  - **Transport Layer Protocol:** Provides server authentication, data confidentiality, and data integrity with forward secrecy (i.e., if a key is compromised during one session, the knowledge does not affect the security of earlier sessions).
  - **User Authentication Protocol:** Authenticates the user to the server.
  - **Connection Protocol:** Multiplexes multiple logical communications channels over a single, underlying SSH connection.

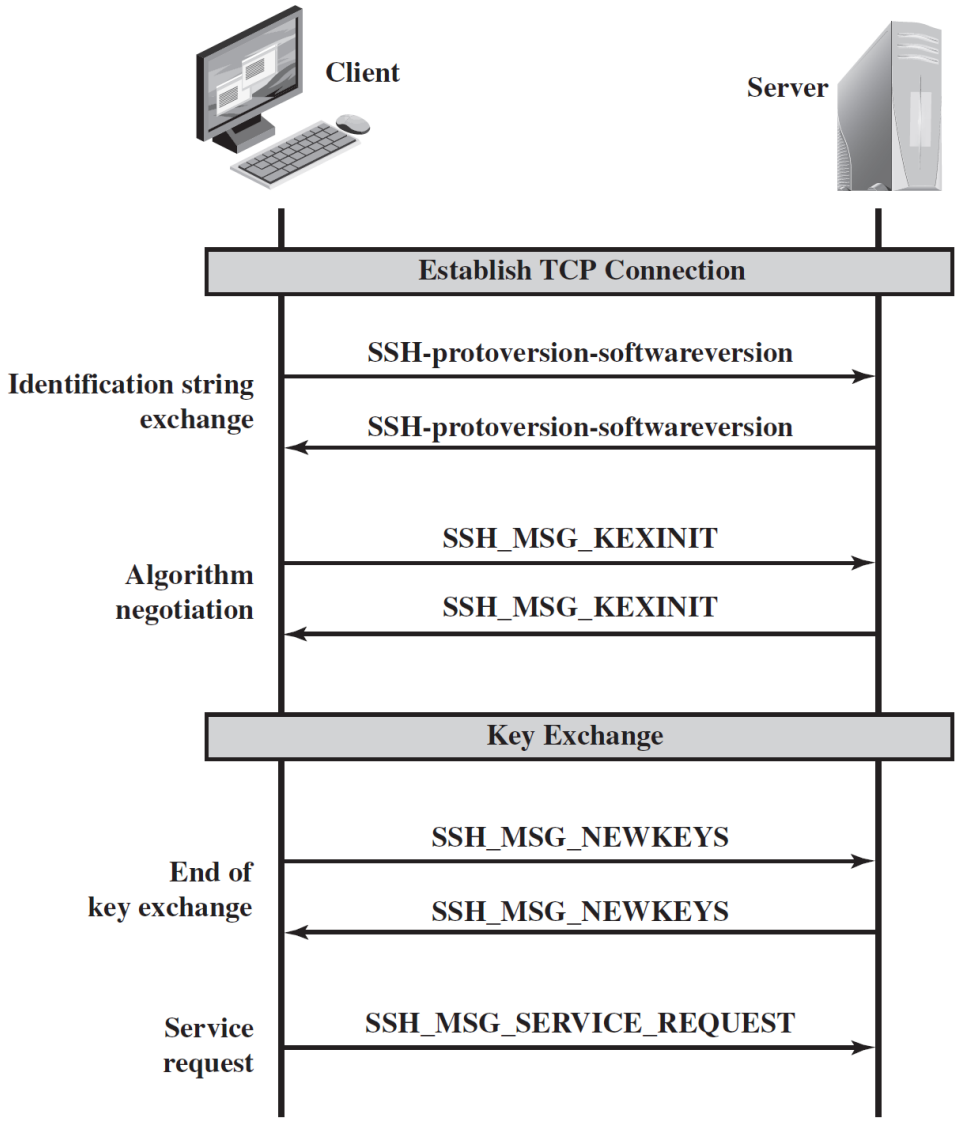


SSH server host keys

- Server authentication occurs at the transport layer, based on the server possessing a public/private key pair.
- A server may have multiple host keys using multiple different asymmetric encryption algorithms. Multiple hosts may share the same host key.
- In any case, the server host key is used during key exchange to authenticate the identity of the host. For this to be possible, the client must have a priori knowledge of the server's public host key. RFC 4251 dictates two alternative trust models that can be used:
  - The client has a local database that associates each host name (as typed by the user) with the corresponding public host key. This method requires no centrally administered infrastructure and no third-party coordination. The downside is that the database of name-to-key associations may become burdensome to maintain.
  - The host name-to-key association is certified by a trusted certification authority (CA). The client only knows the CA root key and can verify the validity of all host keys certified by accepted CAs. This alternative eases the maintenance problem, since ideally, only a single CA key needs to be securely stored on the client. On the other hand, each host key must be appropriately certified by a central authority before authorization is possible.

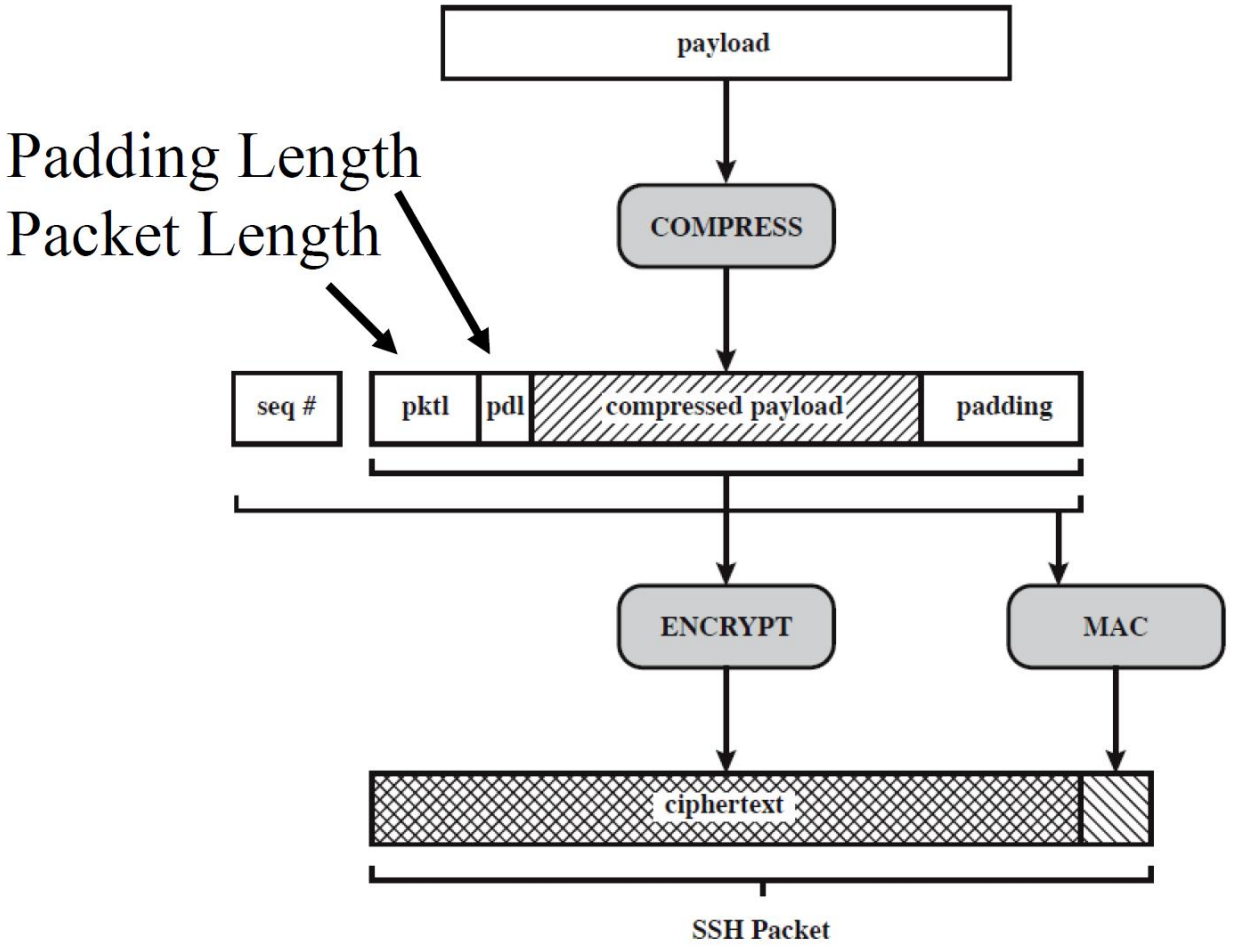
SSH transport layer protocol exchange

- First, the client establishes a TCP connection to the server. This is done via the TCP protocol and is not part of the Transport Layer Protocol.
- Once the connection is established, the client and server exchange data, referred to as packets, in the data field of a TCP segment.



SSH packet format and operation

- **Payload:** Useful contents of the packet. Prior to algorithm negotiation, this field is uncompressed. If compression is negotiated, then in subsequent packets, this field is compressed.
- **Padding length:** Length of the random padding field.
- **Random padding:** Once an encryption algorithm has been negotiated, this field is added. It contains random bytes of padding so that that total length of the packet (excluding the MAC field) is a multiple of the cipher block size, or 8 bytes for a stream cipher.
- **Packet length:** Length of the packet in bytes, not including the packet length and MAC fields.
- **Message authentication code (MAC):** If message authentication has been negotiated, this field contains the MAC value. The MAC value is computed over the entire packet plus a sequence number, excluding the MAC field. The sequence number is an implicit 32-bit packet sequence that is initialized to zero for the first packet and incremented for every packet. The sequence number is not included in the packet sent over the TCP connection.



SSH user authentication protocol

- The server may require one or more of the following user authentication methods:
  - **Public-key:** The details of this method depend on the public-key algorithm chosen. In essence, the client sends a message to the server that contains the client's public key, with the message signed by the client's private key. When the server receives this message, it checks whether the supplied key is acceptable for authentication and, if so, it checks whether the signature is correct.
  - **Password:** The client sends a message containing a plaintext password, which is protected by encryption by the Transport Layer Protocol.
  - **Host-based:** Authentication is performed on the client's host rather than the client itself. Thus, a host that supports multiple clients would provide authentication for all its clients. This method works by having the client send a signature created with the private key of the client host. Thus, rather than directly verifying the user's identity, the SSH server verifies the identity of the client host—and then believes the host when it says the user has already authenticated on the client side.

SSH connection protocol

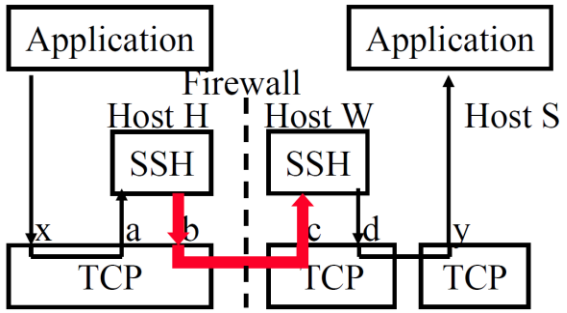
- The SSH Connection Protocol runs on top of the SSH Transport Layer Protocol and assumes that a secure authentication connection is in use. That secure authentication connection, referred to as a **tunnel,** is used by the Connection Protocol to **multiplex** a number of logical channels.
- All types of communication using SSH, such as a terminal session, are supported using separate **channels**. Either side may open a channel. For each channel, each side associates a unique **channel number**, which need not be the same on both ends.
- Channels are flow controlled using a **window** mechanism. No data may be sent to a channel until a message is received to indicate that window space is available.

SSH channel types

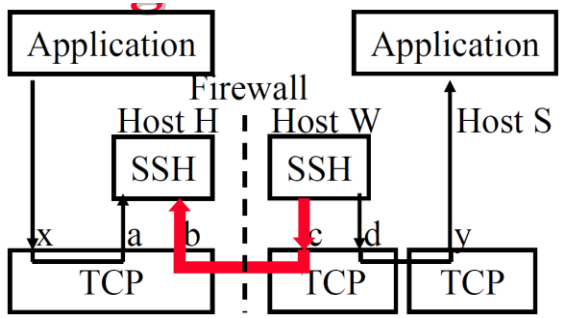
- **session:** The remote execution of a program. The program may be a shell, an application such as file transfer or e-mail, a system command, or some built-in subsystem. Once a session channel is opened, subsequent requests are used to start the remote program.
- **x11:** This refers to the X Window System, a computer software system and network protocol that provides a graphical user interface (GUI) for networked computers. X allows applications to run on a network server but to be displayed on a desktop client machine.
- **forwarded-tcpip:** This is **remote** port forwarding, as explained in the next slide.
- **direct-tcpip:** This is **local** port forwarding, as explained in the next slide.

SSH tunneling/port forwarding

- The –L option specifies **local** forwarding, in which the TCP client is on the local machine with the SSH client. The option is followed by a local port to listen on (**a**), the remote machine name or IP address (**S**), and the remote, target port number (**y**).
- The command logs the user into W and also forwards TCP port **a** on **H** to port **y** on **S**. The client application on **H** must connect to port **a**. The forwarding remains in effect until the user logs out of the session.
- A remotely forwarded port is just like a local one, but the directions are reversed. This time the TCP client is remote, its TCP server is local, and a forwarded connection is initiated by typing the command from the remote machine **W**.
- The –R option specifies **remote** forwarding. It is followed by the remote port to be forwarded (**a**), the machine name or IP address (**S**) and port number (**y**). SSH can now forward connections from (**S,y**) to (**H,a**). Once this command has run, a secure tunnel has been constructed from the port **a** on the machine **H**, to port **y** on the machine **S**.
- If the TCP client application is running locally on the SSH client machine, local forwarding should be used. Otherwise, the client application is on the remote SSH server machine, and remote forwarding should be used.



(a) Local Forwarding
ssh –La:S:y W



(a) Remote Forwarding
ssh –Ra:S:y H

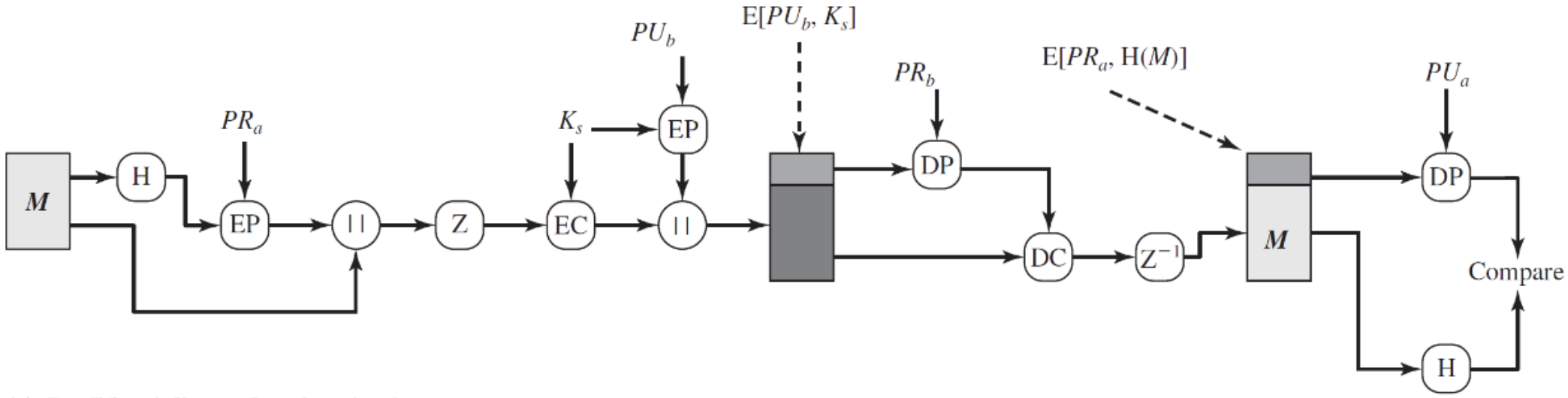
Pretty Good Privacy (PGP)

- PGP provides a confidentiality and authentication service that can be used for electronic mail and file storage applications.
- The actual operation of PGP, as opposed to the management of keys, consists of four services: authentication, confidentiality, compression, and e-mail compatibility (see table).
- OpenPGP is a proposed IETF standard defined in RFC 3156.

Function	Algorithms Used	Description
Digital signature	DSS/SHA or RSA/SHA	A hash code of a message is created using SHA-1. This message digest is encrypted using DSS or RSA with the sender's private key and included with the message.
Message encryption	CAST or IDEA or Three-key Triple DES with Diffie-Hellman or RSA	A message is encrypted using CAST-128 or IDEA or 3DES with a one-time session key generated by the sender. The session key is encrypted using Diffie-Hellman or RSA with the recipient's public key and included with the message.
Compression	ZIP	A message may be compressed for storage or transmission using ZIP.
E-mail compatibility	Radix-64 conversion	To provide transparency for e-mail applications, an encrypted message may be converted to an ASCII string using radix-64 conversion.

PGP Confidentiality and Authentication

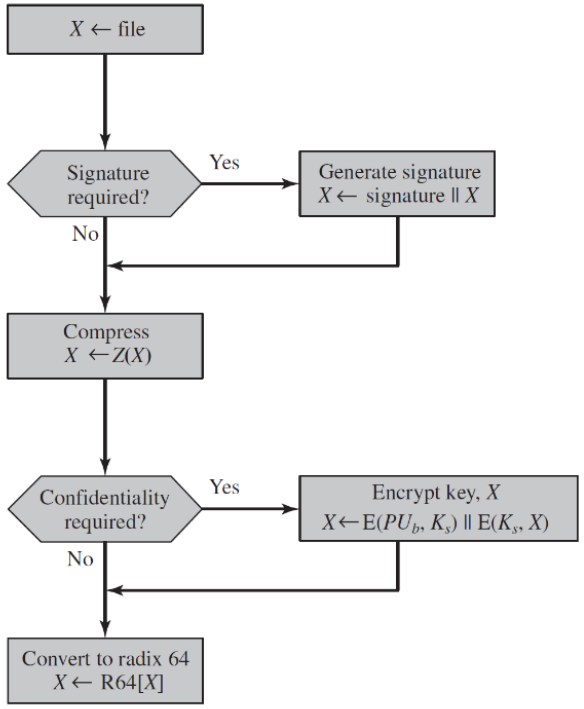
- First, a signature is generated for the plaintext message and prepended to the message. Then the plaintext message plus signature is encrypted using a symmetric encryption (CAST128, IDEA, 3DES), and the session key is encrypted using an asymmetric encryption (RSA or ElGamal).
- This sequence is preferable to the opposite: encrypting the message and then generating a signature for the encrypted message. For purposes of third-party verification, if the signature is performed first, a third party need not be concerned with the symmetric key when verifying the signature.
- When both services are used, the sender first signs the message with its own private key, then encrypts the message with a session key, and finally encrypts the session key with the recipient's public key.



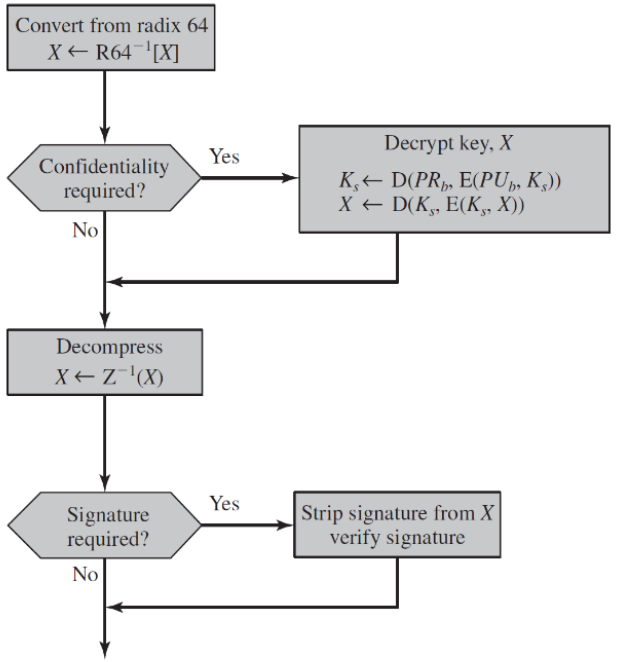
(c) Confidentiality and authentication

Transmission and Reception of PGP Messages

- On transmission (if it is required), a signature is generated using a hash code of the uncompressed plaintext. Then the plaintext (plus signature if present) is compressed. Next, if confidentiality is required, the block (compressed plaintext or compressed signature plus plaintext) is encrypted and prepended with the public key-encrypted symmetric encryption key. Finally, the entire block is converted to radix-64 format.
- On reception, the incoming block is first converted back from radix-64 format to binary. Then, if the message is encrypted, the recipient recovers the session key and decrypts the message. The resulting block is then decompressed. If the message is signed, the recipient recovers the transmitted hash code and compares it to its own calculation of the hash code.



(a) Generic transmission diagram (from A)



(b) Generic reception diagram (to B)

Secure/Multipurpose Internet Mail Extension (S/MIME)

- S/MIME is a security enhancement to the MIME Internet e-mail format standard based on technology from RSA Data Security.
- In terms of general functionality, S/MIME is very similar to PGP. Both offer the ability to sign and/or encrypt messages.
- Although both PGP and S/MIME are on an IETF standards track, S/MIME is considered as the industry standard for commercial and organizational use, while PGP is used for personal e-mail security.
- S/MIME version 3.2 is defined in RFC 5750 and 5751.

S/MIME Functionality

- S/MIME provides the following functions:
  - **Enveloped data:** This consists of encrypted content of any type and encrypted content encryption keys for one or more recipients.
  - **Signed data:** A digital signature is formed by taking the message digest of the content to be signed and then encrypting that with the private key of the signer. The content plus signature are then encoded using base64 encoding. A signed data message can only be viewed by a recipient with S/MIME capability.
  - **Clear-signed data:** As with signed data, a digital signature of the content is formed. However, in this case, only the digital signature is encoded using base64. As a result, recipients without S/MIME capability can view the message content, although they cannot verify the signature.
  - **Signed and enveloped data**: Signed-only and encrypted-only entities may be nested, so that encrypted data may be signed and signed data or clear-signed data may be encrypted.

Cryptographic Algorithms Used in S/MIME

- These are the same algorithms used in PGP and provide a high level of security.
- A sending agent has two decisions to make. First, the sending agent must determine if the receiving agent is capable of decrypting using a given encryption algorithm. Second, if the receiving agent is only capable of accepting weakly encrypted content, the sending agent must decide if it is acceptable to send using weak encryption.
- To support this decision process, a sending agent may announce its decrypting capabilities in order of preference for any message that it sends out. A receiving agent may store that information for future use.
- If a message is to be sent to multiple recipients and a common encryption algorithm cannot be selected for all, then the sending agent will need to send two messages. However, in that case, it is important to note that the security of the message is made vulnerable by the transmission of one copy with lower security.

Function	Requirement
Create a message digest to be used in forming a digital signature.	MUST support SHA-1. Receiver SHOULD support MD5 for backward compatibility.
Encrypt message digest to form a digital signature.	Sending and receiving agents MUST support DSS. Sending agents SHOULD support RSA encryption. Receiving agents SHOULD support verification of RSA signatures with key sizes 512 bits to 1024 bits.
Encrypt session key for transmission with a message.	Sending and receiving agents SHOULD support Diffie-Hellman. Sending and receiving agents MUST support RSA encryption with key sizes 512 bits to 1024 bits.
Encrypt message for transmission with a one-time session key.	Sending and receiving agents MUST support encryption with tripleDES. Sending agents SHOULD support encryption with AES. Sending agents SHOULD support encryption with RC2/40.
Create a message authentication code.	Receiving agents MUST support HMAC with SHA-1. Sending agents SHOULD support HMAC with SHA-1.

S/MIME Content Types

- S/MIME makes use of a number of new MIME content types, which are shown in Table. All of the new application types use the designation **pkcs7** which indicates a public-key cryptographic standard used to sign and/or encrypt messages under a PKI.
- S/MIME secures a MIME entity with a signature, encryption, or both. A MIME entity may be an entire message (except for the RFC 5322 headers), or if the MIME content type is multipart, then a MIME entity is one or more of the subparts of the message.
- The MIME entity is prepared according to the normal rules for MIME message preparation. Then the MIME entity plus some security-related data, such as algorithm identifiers and certificates, are processed by S/MIME to produce what is known as a PKCS object. A PKCS object is then treated as message content and wrapped in MIME (provided with appropriate MIME headers).

Type	Subtype	smime Parameter	Description
Multipart	Signed		A clear-signed message in two parts: one is the message and the other is the signature.
Application	pkcs7-mime	signedData	A signed S/MIME entity.
	pkcs7-mime	envelopedData	An encrypted S/MIME entity.
	pkcs7-mime	degenerate signedData	An entity containing only public-key certificates.
	pkcs7-mime	CompressedData	A compressed S/MIME entity.
	pkcs7-signature	signedData	The content type of the signature subpart of a multipart/signed message.

S/MIME Certificate Processing

- S/MIME uses public-key certificates that conform to X.509v3.
- The key-management scheme used by S/MIME is in some ways a hybrid between a strict X.509 certification hierarchy and PGP's web of trust. As with the PGP model, S/MIME managers and/or users must configure each client with a list of trusted keys and with certificate revocation lists. That is, the responsibility is local for maintaining the certificates needed to verify incoming signatures and to encrypt outgoing messages. On the other hand, the certificates are signed by certification authorities.
- An S/MIME user has several key-management functions to perform.
  - **Key generation:** The user of some related administrative utility (e.g., one associated with LAN management) MUST be capable of generating separate Diffie-Hellman and DSS key pairs and SHOULD be capable of generating RSA key pairs. Each key pair MUST be generated from a good source of nondeterministic random input and be protected in a secure fashion. A user agent SHOULD generate RSA key pairs with a length in the range of 768 to 1024 bits and MUST NOT generate a length of less than 512 bits.
  - **Registration:** A user's public key must be registered with a certification authority in order to receive an X.509 public-key certificate.
  - **Certificate storage and retrieval:** A user requires access to a local list of certificates in order to verify incoming signatures and to encrypt outgoing messages. Such a list could be maintained by the user or by some local administrative entity on behalf of a number of users.

References

- Cryptography and Network Security, Principles and Practice, Sixth Edition, William Stallings, Pearson, ISBN 978-0-13-335469-0 (2014)
- SSL and TLS, Theory and Practice, Rolf Oppliger, Artech House, ISBN 978-1-59693-447-4 (2009)
- SSH, the Secure Shell: The Definitive Guide, Second Edition, Daniel J. Barrett, Richard E. Silverman, and Robert G. Byrnes, O'Reilly, ISBN 978-0-596-00895-6 (2005)