Chapter 3

Public-Key Cryptography

Introduction

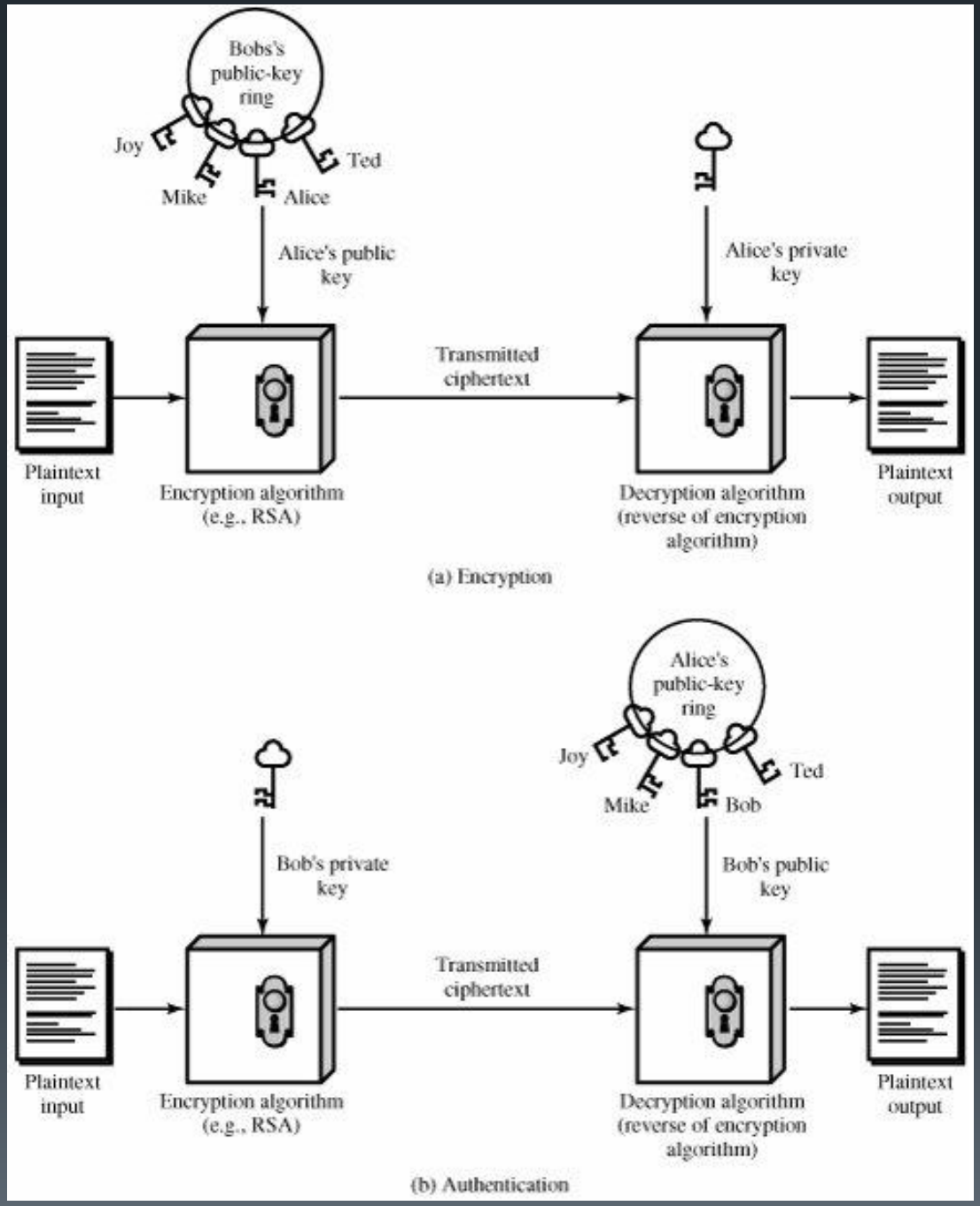
- The development of public-key cryptography is the greatest and perhaps the only true revolution in the entire history of cryptography.
- Public key algorithms are based on mathematical functions rather than on substitution and permutation
- Several common misconceptions :
  - Public-key encryption is more secure from cryptanalysis than is symmetric encryption
  - Public-key encryption is a general-purpose technique that has made symmetric encryption obsolete
  - key distribution is trivial when using public-key encryption, compared to symmetric encryption

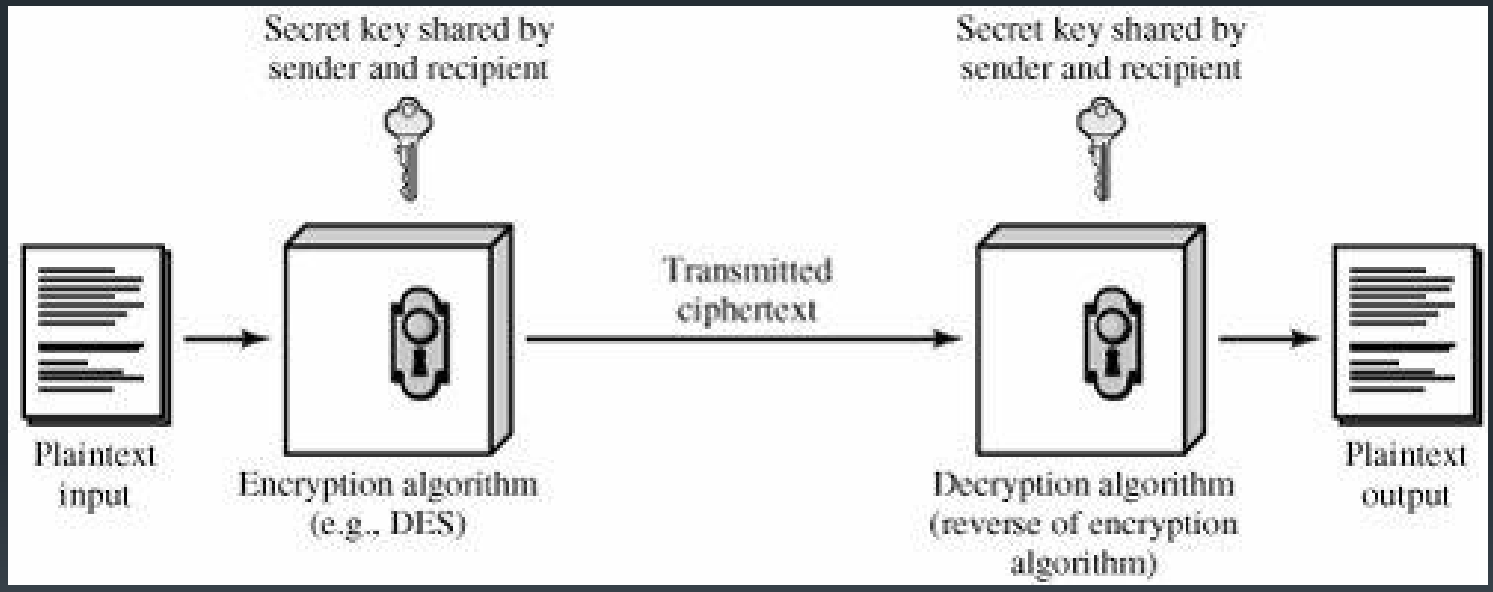
Introduction

- The concept of public-key cryptography evolved from an attempt to attack two of the most difficult problems associated with symmetric encryption
- Key distribution
- Digital signature
- Diffie and Hellman achieved an astounding breakthrough in 1976 by coming up with a method that addressed both problems

Public-Key Cryptosystems

- It is computationally infeasible to determine the decryption key given only knowledge of the cryptographic algorithm and the encryption key.
- Either of the two related keys can be used for encryption, with the other used for decryption


Bobs's public-key ring
Joy
Ted
Mike
Alice
Alice's public key
Alice's private key
Transmitted ciphertext
Plaintext input
Encryption algorithm (e.g., RSA)
Decryption algorithm (reverse of encryption algorithm)
Plaintext output
(a) Encryption
Alice's public-key ring
Joy
Ted
Mike
Bob
Bob's private key
Bob's public key
Transmitted ciphertext
Plaintext input
Encryption algorithm (e.g., RSA)
Decryption algorithm (reverse of encryption algorithm)
Plaintext output
(b) Authentication


Secret key shared by sender and recipient
Secret key shared by sender and recipient
Transmitted ciphertext
Plaintext input
Encryption algorithm (e.g., DES)
Decryption algorithm (reverse of encryption algorithm)
Plaintext output

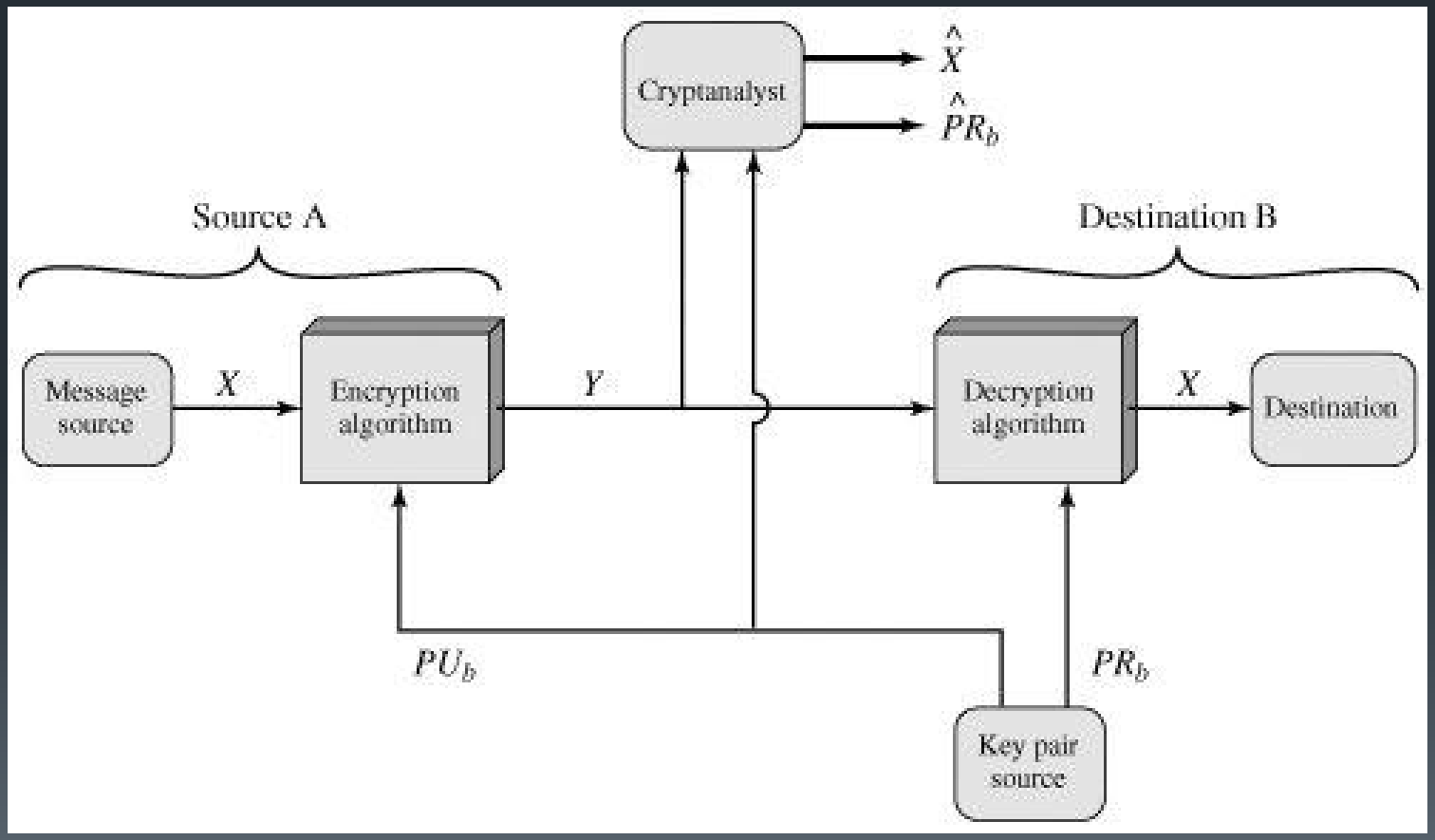
Steps to perform Encryption

- Each user generates a pair of keys to be used for the encryption and decryption of messages.
- Each user places one of the two keys in a public register or other accessible file. This is the public key. The companion key is kept private. As Figure shows, each user maintains a collection of public keys obtained from others.
- If Bob wishes to send a confidential message to Alice, Bob encrypts the message using Alice's public key.
- When Alice receives the message, she decrypts it using her private key. No other recipient can decrypt the message because only Alice knows Alice's private key.

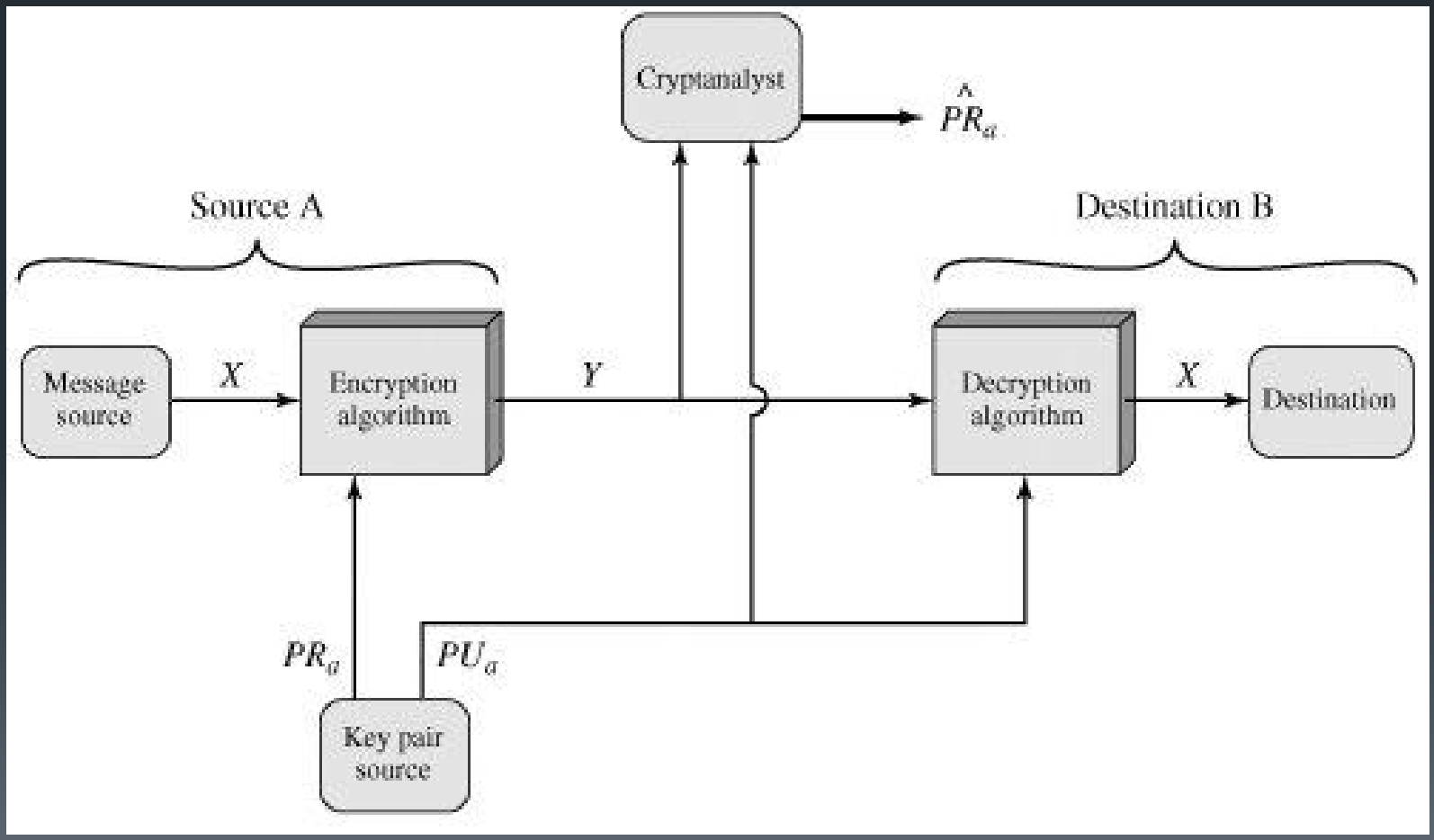
Comparison

Conventional Encryption	Public-Key Encryption
The same algorithm with the same key is used for encryption and decryption.	One algorithm is used for encryption and decryption with a pair of keys, one for encryption and one for decryption
The sender and receiver must share the algorithm and the key.	The sender and receiver must each have one of the matched pair of keys (not the same one).
The key must be kept secret.	One of the two keys must be kept secret.
Knowledge of the algorithm plus samples of ciphertext must be insufficient to determine the key	Knowledge of the algorithm plus one of the keys plus samples of ciphertext must be insufficient to determine the other key

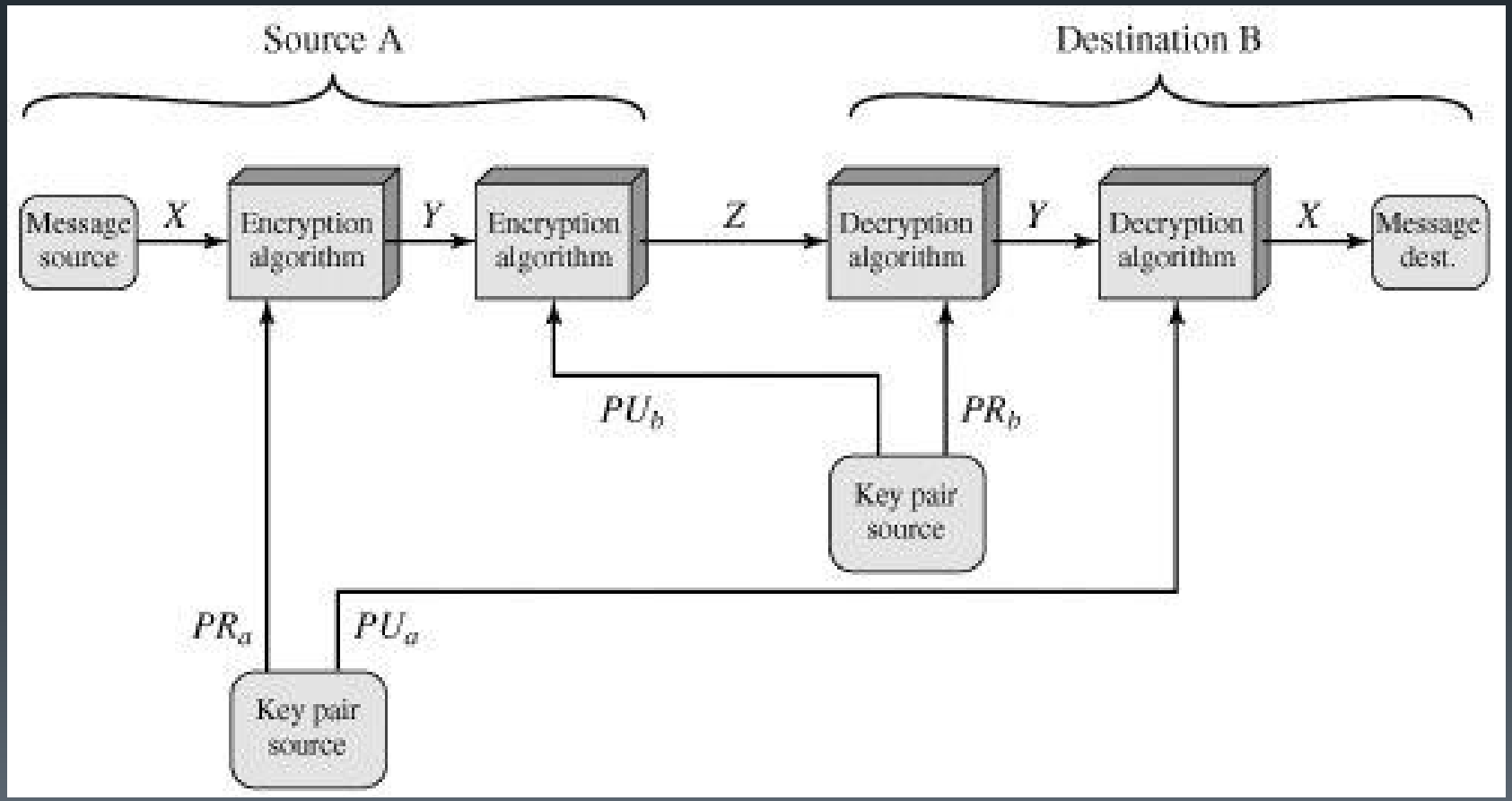
Public-Key Cryptosystem: Secrecy



Public-Key Cryptosystem: Authentication



Public-Key Cryptosystem: Authentication and Secrecy



Requirements for Public-Key Cryptography

- It is computationally easy for a party B to generate a pair (public key PU_b, private key PR_b).
- It is computationally easy for a sender A, knowing the public key and the message to be encrypted, M, to generate the corresponding cipher text: $C = \mathrm{E}(PU_b, M)$
- It is computationally easy for the receiver B to decrypt the resulting cipher text using the private key to recover the original message: $M = \mathrm{D}(PR_b, C) = \mathrm{D}[PR_b, \mathrm{E}(PU_b, M)]$
- It is computationally infeasible for an adversary, knowing the public key, PU_b, to determine the private key, PR_b.
- It is computationally infeasible for an adversary, knowing the public key, PU_b, and a cipher text, C, to recover the original message, M.
- $M = \mathrm{D}[PU_b, \mathrm{E}(PR_b, M)] = \mathrm{D}[PR_b, \mathrm{E}(PU_b, M)]$

The RSA Algorithm

- The pioneering paper by Diffie and Hellman [DIFF76b] introduced a new approach to cryptography and, in effect, challenged cryptologists to come up with a cryptographic algorithm that met the requirements for public-key systems.
- One of the first of the responses to the challenge was developed in 1977 by Ron Rivest, Adi Shamir, and Len Adleman at MIT and first published in 1978.

RSA Key generation:

1. Select 2 large prime numbers of about the same size, p and q
   Typically each p, q has between 512 and 2048 bits
2. Compute n = pq, and ⎯(n) = (q-1)(p-1)
3. Select e, 1<e< ⎯(n), s.t. gcd(e, ⎯(n)) = 1
4. Compute d, 1< d< ⎯(n) s.t. ed ⌒ 1 mod ⎯(n)
   Knowing ⎯(n), d easy to compute.

Public key: (e, n)
Private key: d

RSA Description

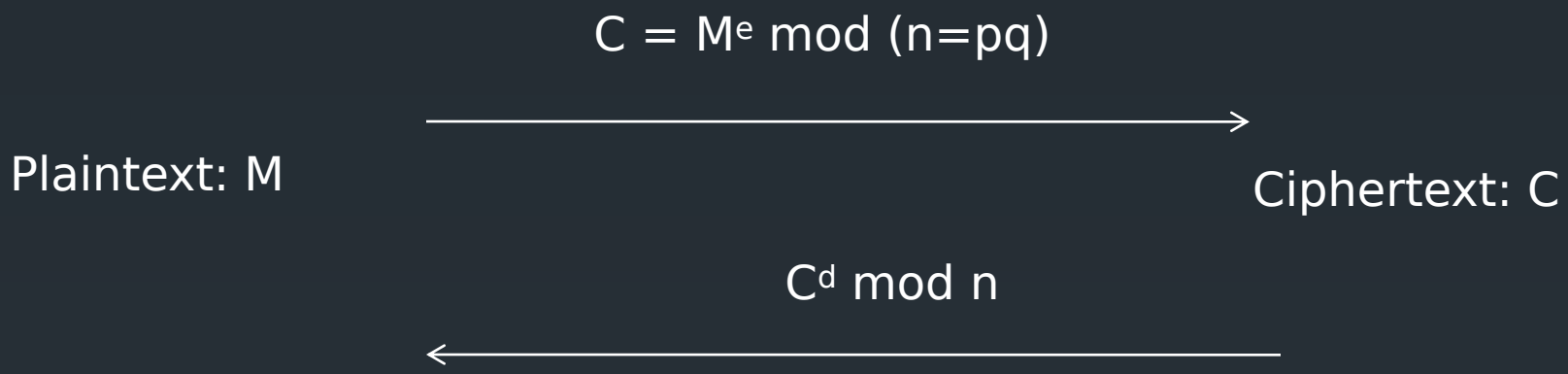
- Encryption :

Given a message M, $0 < M < n$
use public key (e, n)
compute , $C = M^e \bmod n$

- Decryption :

Given a ciphertext C, use private key (d)
Compute $C^d \bmod n = (M^e \bmod n)^d \bmod n = M^{ed} \bmod n = M$

RSA Description



From n, difficult to figure out p,q
From (n,e), difficult to figure d.
From (n,e) and C, difficult to figure out M such that $C = M^e$

RSA Example

- p = 11, q = 7, n = 77, $\phi(n) = 60$
- d = 13, e = 37 (ed = 481; ed mod 60 = 1)
- Let M = 15. Then $C \equiv M^e \bmod n$
  - $C \equiv 15^{37} \pmod{77} = 71$
- $M \equiv C^d \bmod n$
  - $M \equiv 71^{13} \pmod{77} = 15$

RSA Example 2

- Parameters:
  - p = 3, q = 5, q= pq = 15
  - ⌟(n) = ?
- Let e = 3, what is d?
- Given M=2, what is C?
- How to decrypt?

Multiplicative Inverses

Example 3. Find the multiplicative inverse of 8 mod 11, using the Euclidean Algorithm.

Solution. We'll organize our work carefully. We'll do the Euclidean Algorithm in the left column. It will verify that $gcd(8, 11) = 1$. Then we'll solve for the remainders in the right column, before backsolving:

$$\begin{array}{rcl|rcl} \mathbf{11} & = & \mathbf{8}(1) + \mathbf{3} & 3 & = & 11 - 8(1) \\ \mathbf{8} & = & \mathbf{3}(2) + \mathbf{2} & 2 & = & 8 - 3(2) \\ \mathbf{3} & = & \mathbf{2}(1) + \mathbf{1} & 1 & = & 3 - 2(1) \\ \mathbf{2} & = & \mathbf{1}(2) & & & \end{array}$$

Now reverse the process using the equations on the right.

$$1 = 3 - 2(1)$$
$$1 = 3 - (8 - 3(2))(1) = 3 - (8 - (3(2)) = 3(3) - 8$$
$$1 = (11 - 8(1))(3) - 8 = 11(3) - 8(4) = 11(3) + 8(-4)$$

Therefore $1 \equiv 8(-4) \mod 11$, or if we prefer a residue value for the multiplicative inverse,

$$1 \equiv 8(7) \mod 11.$$

So 7 is multiplicative inverse of 8 mod 11. In RSA, e = 8 and d=7.

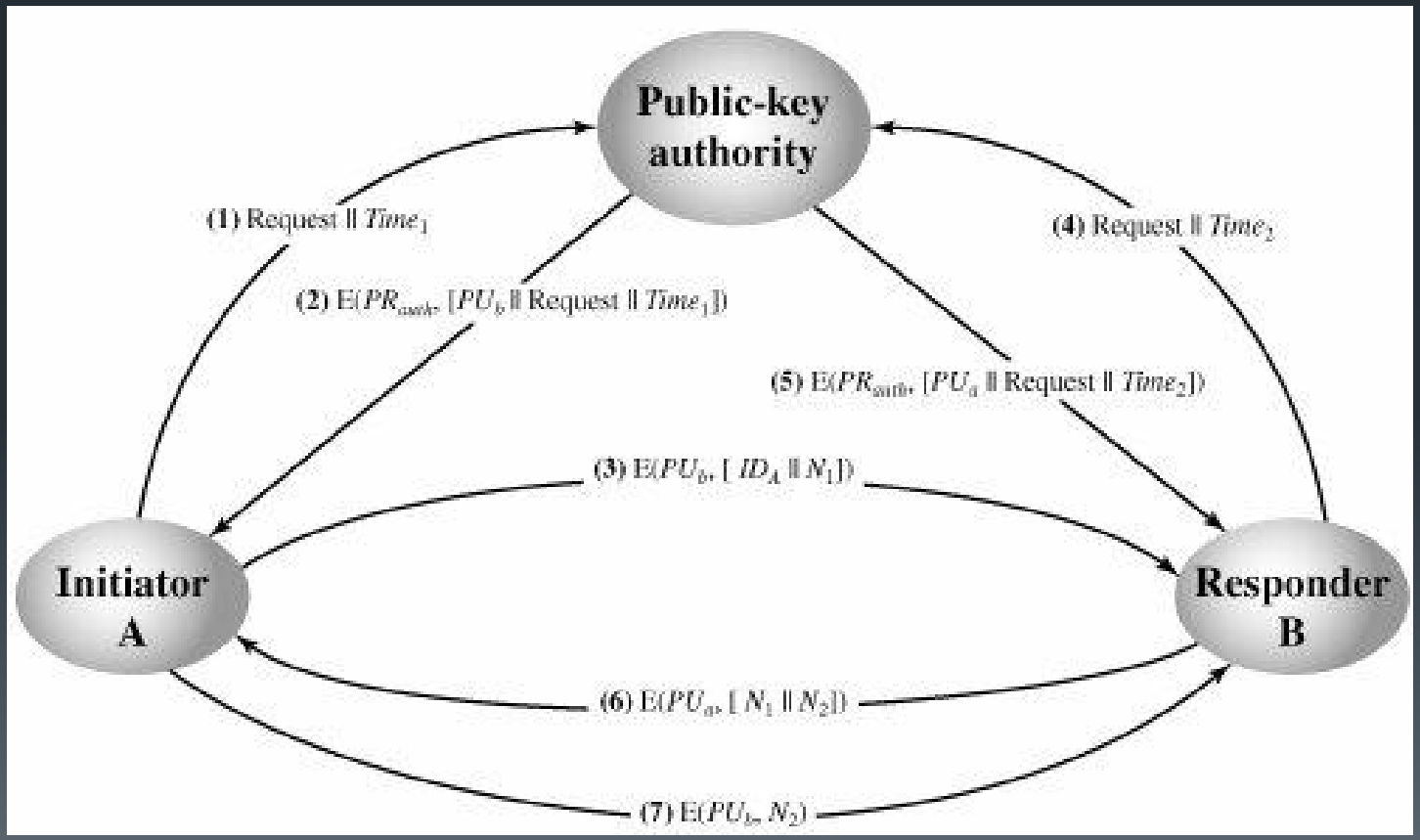
RSA Security

- Security depends on the difficulty of factoring n
  - Factor n => _(n) => compute d from (e, _(n))
- The length of n=pq reflects the strength
  - 700-bit n factored in 2007
  - 768 bit factored in 2009
- 1024 bit for minimal level of security today
  - likely to be breakable in near future
- Minimal 2048 bits recommended for current usage

Distribution of Public Keys

- Several techniques have been proposed for the distribution of public keys. Virtually all these proposals can be grouped into the following general schemes:
  - Public announcement
  - Publicly available directory
  - Public-key authority
  - Public-key certificates

Public-Key Distribution Scenario



Public-Key Distribution Scenario

1. A sends a time stamped message to the public-key authority containing a request for the current public key of B.
2. The authority responds with a message that is encrypted using the authority's private key, Pr_{auth}
3. A stores B's public key and also uses it to encrypt a message to B containing an identifier of A (*IDA) and a nonce (N1), which is used to identify this transaction uniquely.*
4. B retrieves A's public key from the authority in the same manner as A retrieved B's public key.
5. public keys have been securely delivered to A and B, and they may begin their protected exchange
6. B sends a message to A encrypted with *PUa and containing A's nonce (N1) as well as a new nonce* generated by B (*N2) Because only B could have decrypted message (3), the presence of N1 in* message (6) assures A that the correspondent is B.
7. A returns *N2, encrypted using B's public key, to assure B that its correspondent is A.*

Distribution of Secret Keys Using Public-Key Cryptography

- Public-key encryption provides for the distribution of secret keys to be used for conventional encryption.
- Simple Secret Key Distribution:
  - A generates a public/private key pair {PU_a, PR_a} *and transmits a message to B consisting of* Pu_a and an identifier of *A,* ID_A.
  - B generates a secret key, *Ks, and transmits it to A, encrypted with A's public key*
  - A computes D(PR_a, *E(*PU_a, K_s)) *to recover the secret key. Because only A can decrypt the* message, only A and B will know the identity of K_s.
  - A discards *PUa and PRa and B discards PUa*

Distribution of Secret Keys Using Public-Key Cryptography

- The protocol is insecure against an adversary who can intercept messages and then either relay the intercepted message or substitute another message .
- Such an attack is known as a **man-in-the-middle attack**
  - A generates a public/private key pair {*PUa, PRa} and transmits a message intended for B* consisting of *PUa and an identifier of A, IDA.*
  - E intercepts the message, creates its own public/private key pair {*PUe, PRe} and transmits PUe|| IDA to B.*
  - B generates a secret key, *Ks, and transmits E(PUe, Ks).*
  - E intercepts the message, and learns *Ks by computing D(PRe, E(PUe, Ks)).*
  - E transmits E(*PUa, Ks) to A.*

The Diffie-Hellman Algorithm

- The question of key exchange was one of the first problems addressed by a cryptographic protocol. This was prior to the invention of public key cryptography.
- The Diffie-Hellman key agreement protocol (1976) was the first practical method for establishing a shared secret over an unsecured communication channel.
- The point is to agree on a key that two parties can use for a symmetric encryption, in such a way that an eavesdropper cannot obtain the key.

The Diffie-Hellman Algorithm

- How can two parties agree on a secret value when all of their messages might be overheard by an eavesdropper?
- The Diffie-Hellman algorithm accomplishes this, and is still widely used.
- With sufficiently large inputs, Diffie-Hellman is very secure.
  - Allows two users to exchange a secret key
  - Requires no prior secrets
  - Real-time over an untrusted network

The Diffie-Hellman Algorithm

- Steps in the algorithm:

1. Alice and Bob agree on a prime number p and g.
2. Alice chooses a secret number **a**, and sends Bob $(g^a \bmod p)$.
3. Bob chooses a secret number b, and sends Alice $(g^b \bmod p)$.
4. Alice computes $((g^b \bmod p)^a \bmod p)$.
5. Bob computes $((g^a \bmod p)^b \bmod p)$.
6. Both Alice and Bob can use this number as their key. Notice that
7. p and g need not be protected.

Diffie-Hellman Example

1. Alice and Bob agree on $p = 23$ and $g = 5$.
2. Alice chooses $a = 6$ and sends $5^6 \bmod 23 = 8$.
3. Bob chooses $b = 15$ and sends $5^{15} \bmod 23 = 19$.
4. Alice computes $19^6 \bmod 23 = 2$.
5. Bob computes $8^{15} \bmod 23 = 2$.

- Then 2 is the shared secret.
- Clearly, much larger values of a, b, and p are required. An eavesdropper cannot discover this value even if she knows p and g and can obtain each of the messages.

Authentication

- Message authentication assures that data received are exactly as sent by and that the claimed identity of the sender is valid.
- Authentication is the first step in any cryptography solution, there is no use of encryption without authentication
- There are many ways to authenticate a user like user id- password, biometric, smart card, certificate based etc.

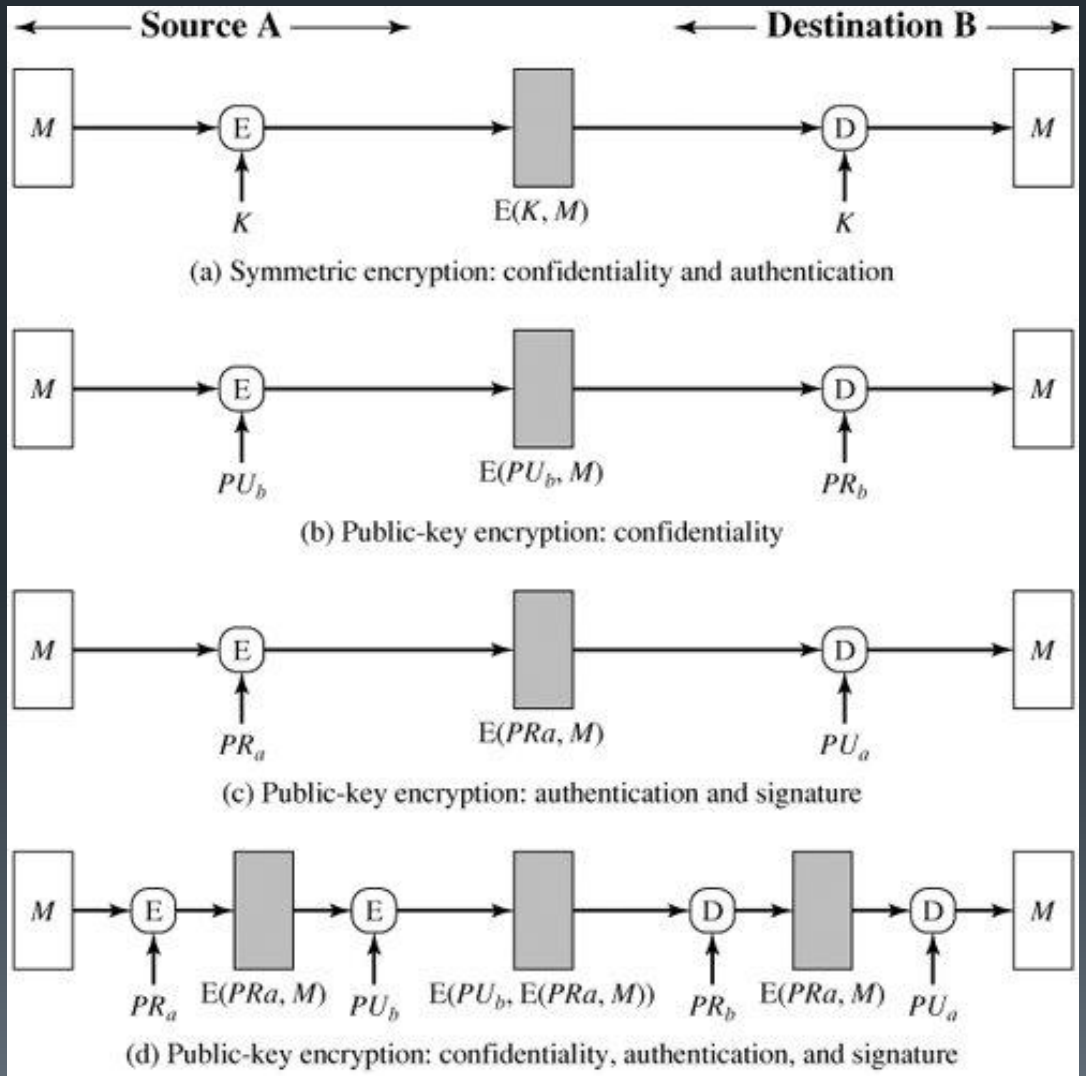
Authentication Requirements

- In the context of communications across a network, the following attacks can be identified:
  - Disclosure
  - Traffic analysis
  - Masquerade
  - Content modification
  - Sequence modification
  - Timing modification
  - Source repudiation
  - Destination repudiation

Authentication Functions

- The functions are grouped into three classes
  - Message encryption: The ciphertext of the entire message serves as its authenticator
  - Message authentication code (MAC): A function of the message and a secret key that produces a fixed-length value that serves as the authenticator
  - Hash function: A function that maps a message of any length into a fixed-length hash value, which serves as the authenticator

Message Encryption



Message Encryption

- Suppose that we are transmitting English-language messages using a Caesar cipher with a shift of one (K = 1). A sends the following legitimate ciphertext:

  nbsftfbupbutboeepftfbupbutboemjuumfmbnctfbujwz

- B decrypts to produce the following plaintext:

  mareseatoatsanddoeseatoatsandlittlelambseativy

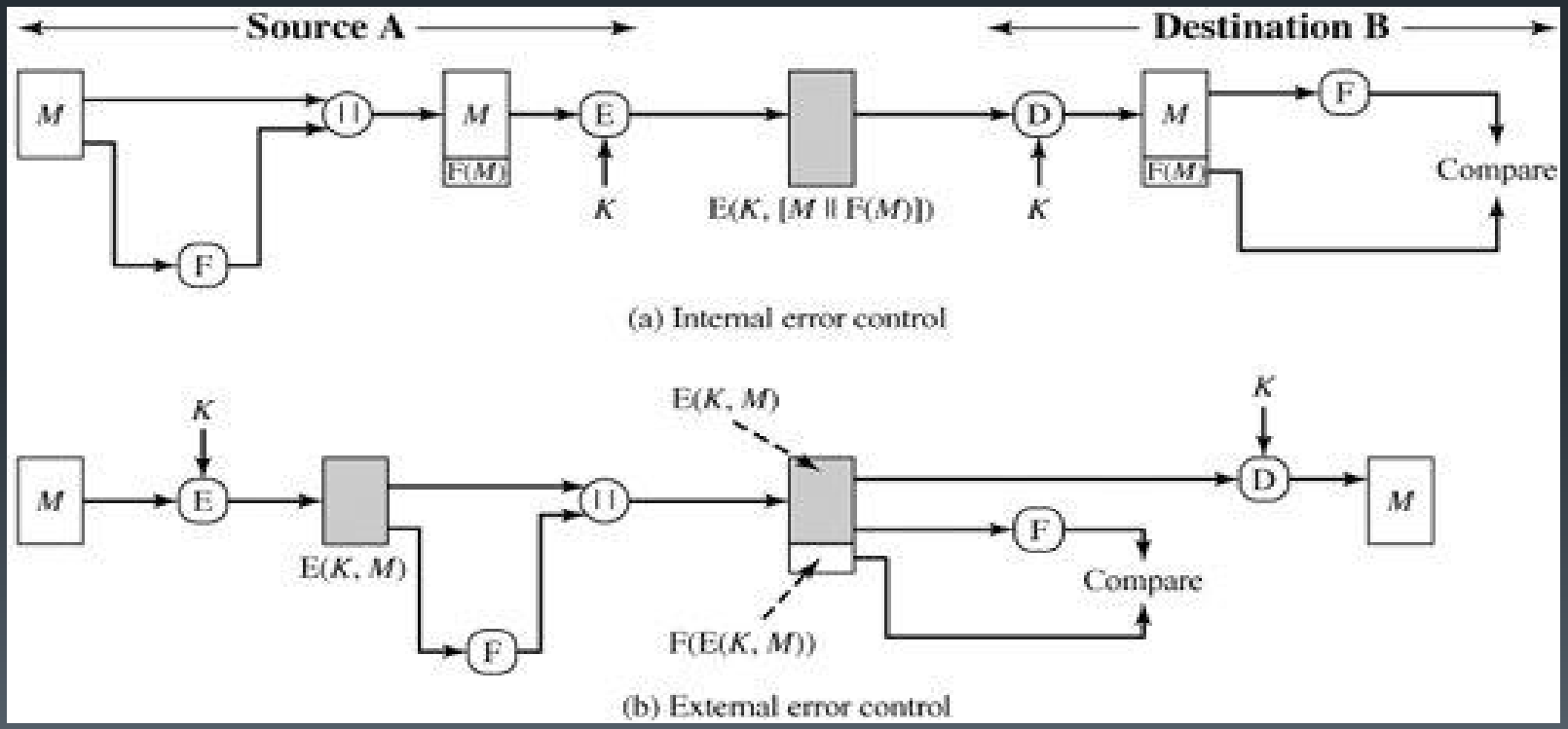
- If an opponent generates the following random sequence of letters:

  zuvrsoevgqxlzwigamdvnmhpmccxiuureosfbcebtqxsxq

- this decrypts to:

  ytuqrndufpwkyvhfzlcumlgolbbwhttqdnreabdaspwrwp

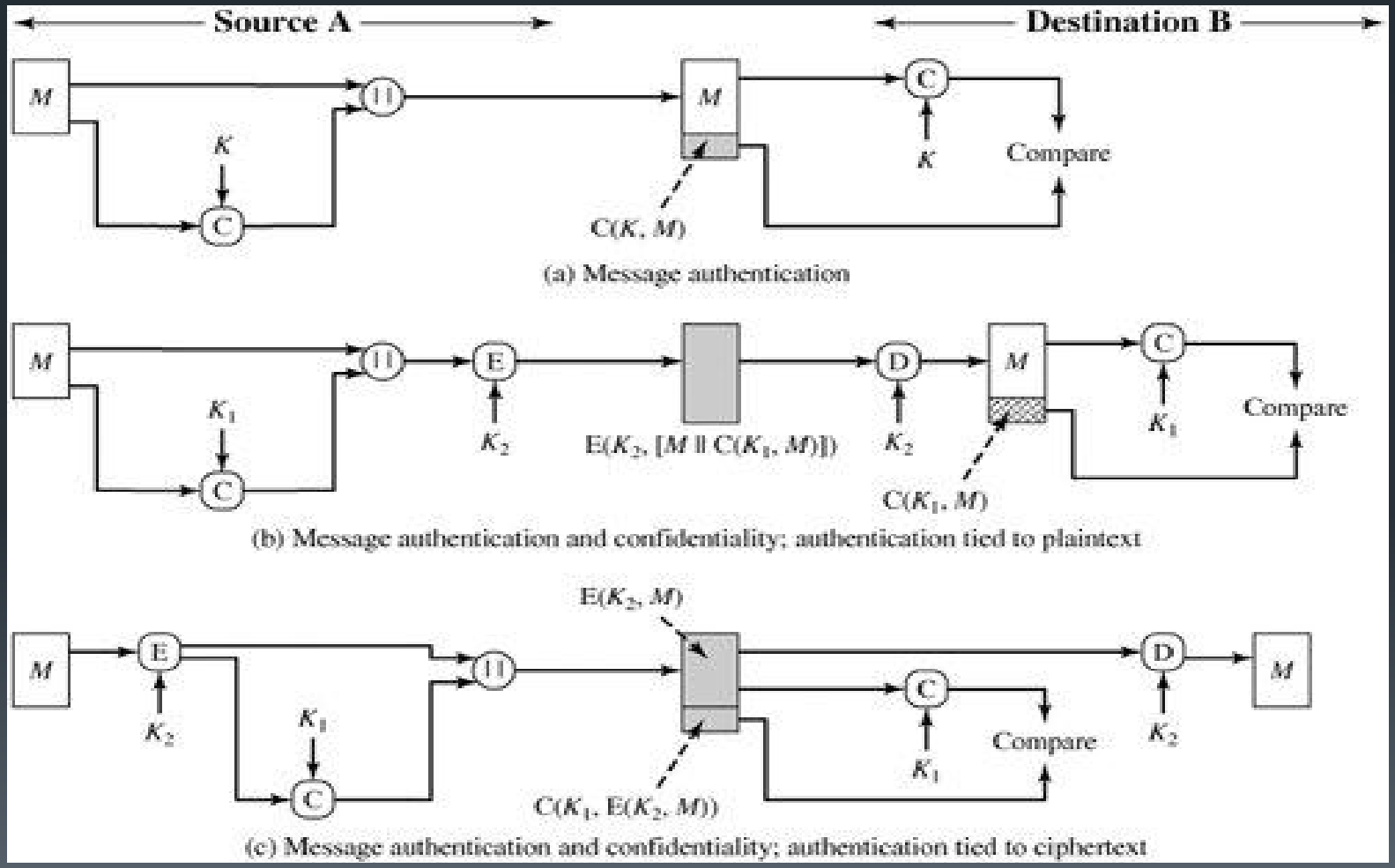
Frame Check Sequence



Message Authentication Code

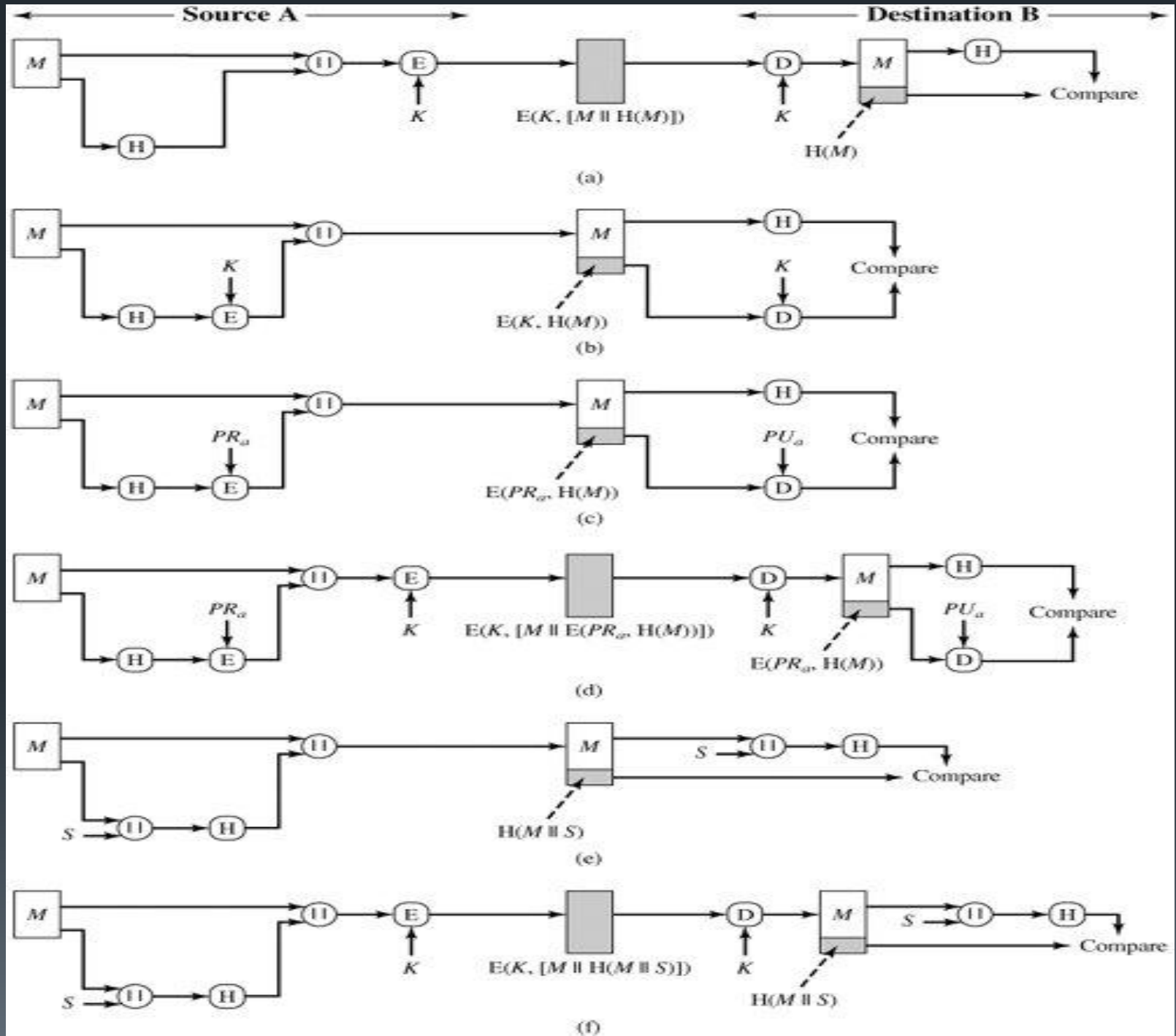
- An alternative authentication technique involves the use of a secret key to generate a small fixed-size block of data, known as a cryptographic checksum or MAC that is appended to the message
- MAC = C(K, M)
- The message plus MAC are transmitted to the intended recipient.
- The recipient performs the same calculation on the received message, using the same secret key, to generate a new MAC.
- If the received and calculated MAC is same the message is authenticated.

Basic Uses of Message Authentication Code

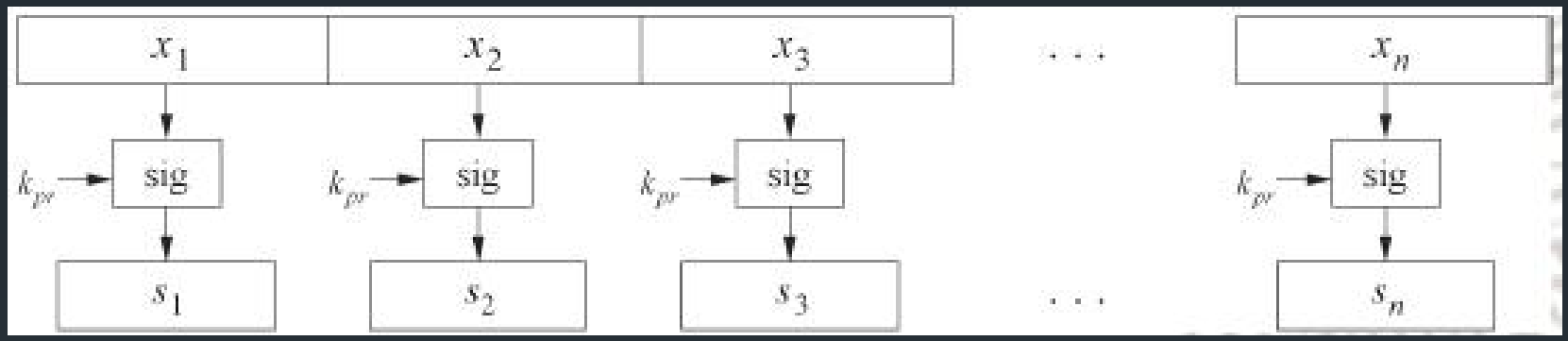


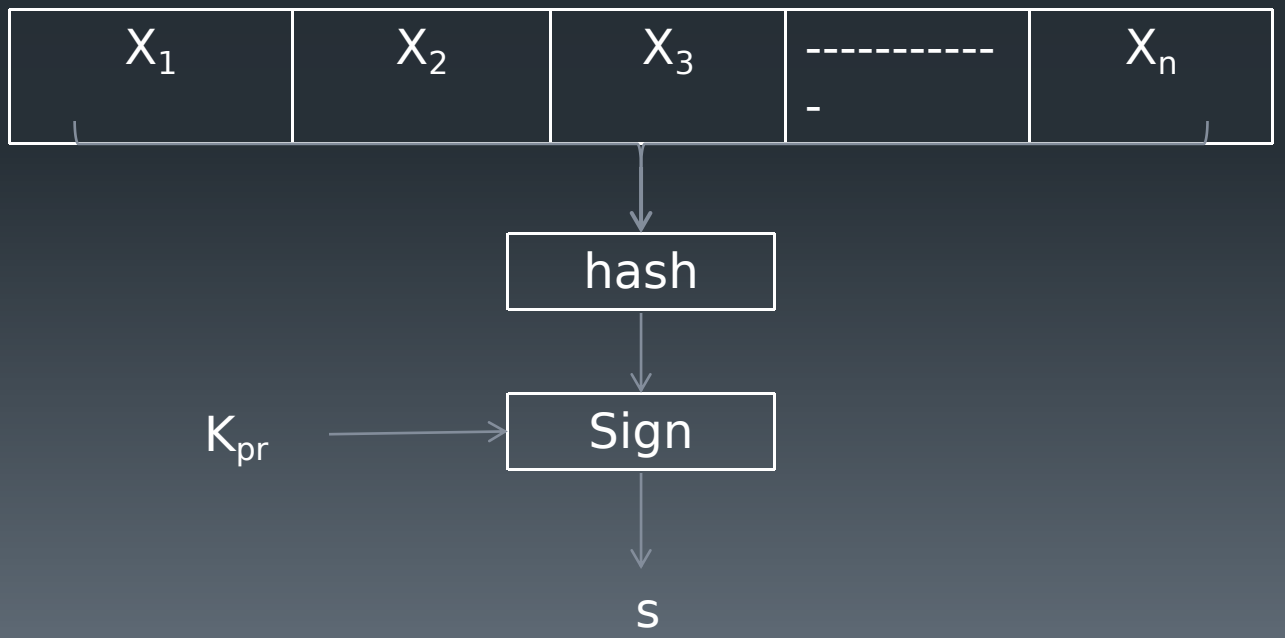
Hash Function

- As with the message authentication code, a hash function accepts a variable-size message M as input and produces a fixed size output, referred to as a hash code H(M).
- Unlike a MAC, a hash code does not use a key but is a function only of the input message.
- The hash code is also referred to as a message digest or hash value.
- The hash code is a function of all the bits of the message and provides an error-detection capability: A change to any bit or bits in the message results in a change to the hash code.


Source A
Destination B
M
H
E
D
K
Compare
E(K, [M || H(M)])
H(M)
(a)
E(K, H(M))
(b)
PR_a
PU_a
E(PR_a, H(M))
(c)
E(K, [M || E(PR_a, H(M))])
(d)
S
H(M || S)
(e)
E(K, [M || H(M || S)])
(f)

Hash Function: Motivation

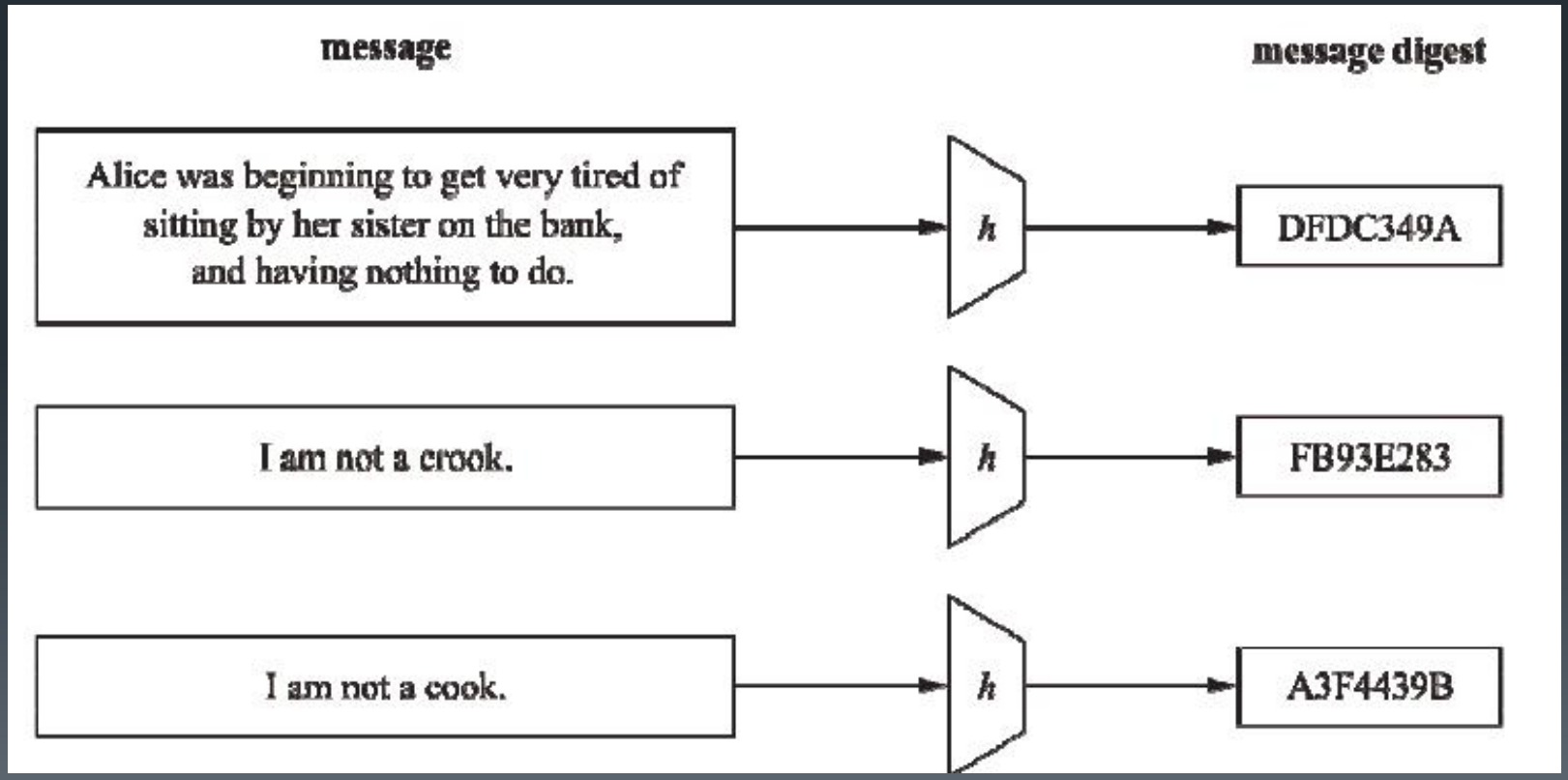




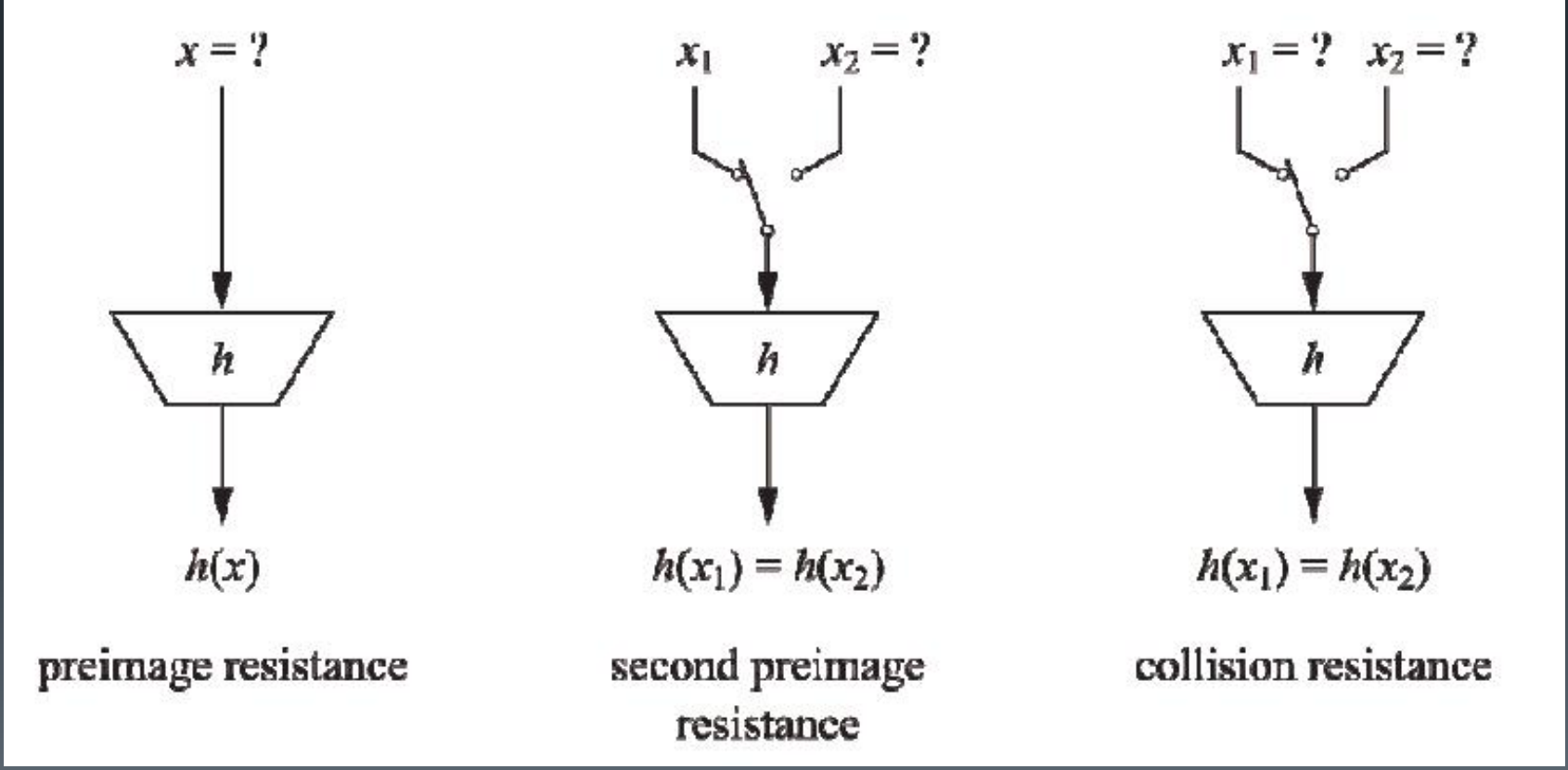
Requirements

- Arbitrary input length
- Fixed, short output length
- Should be efficient and fast
- Preimage resistance or one way function
- Second Preimage resistance
- Collision resistance

Hash Function



Security Properties of Hash Function



Simple Hash Functions

- All hash functions operate using the following general principles.
- The input (message, file, etc.) is viewed as a sequence of *n-bit blocks and processed one block at a time in an iterative fashion* to produce an *n-bit hash function.*
- *One of the simplest hash functions is the bit-by-bit exclusive-OR (XOR) of every block, expressed as*

$$C_i = b_{i1} \oplus b_{i1} \oplus \ldots \oplus b_{im}$$

C_i = ith bit of the hash code, $1 \leq i \leq n$

m = number of n-bit blocks in the input

b_{ij} = ith bit in jth block

Birthday Attacks

- If an encrypted hash code C is transmitted with the corresponding unencrypted message *M* , then an opponent would need to find an *M' such that H(M') = H(M) to substitute another* message and fool the receiver.
- On average, the opponent would have to try about 2^{64} messages to find one that matches the hash code of the intercepted message
- However, a different sort of attack is possible, based on the birthday paradox

Birthday Attacks

1. The source, A, is prepared to "sign" a message by appending the appropriate *m-bit hash code* and encrypting that hash code with A's private key
2. The opponent generates $2^{m/2}$ *variations on the message, all of which convey essentially the* same meaning.
3. The two sets of messages are compared to find a pair of messages that produces the same hash code. The probability of success, by the birthday paradox, is greater than 0.5. If no match is found, additional valid and fraudulent messages are generated until a match is made.
4. The opponent offers the valid variation to A for signature. This signature can then be attached to the fraudulent variation for transmission to the intended recipient. Because the two variations have the same hash code, they will produce the same signature

Dear Anthony,

{This letter is / I am writing} to introduce {you to / to you} {Mr. / --} Alfred {P. / --}

Barton, the {new / newly appointed} {chief / senior} jewellery buyer for {our / the}

Northern {European / Europe} {area / division}. He {will take / has taken} over {the / --}

responsibility for {all / the whole of} our interests in {watches and jewellery / jewellery and watches}

in the {area / region}. Please {afford / give} him {every / all the} help he {may need / needs}

to {seek out / find} the most {modern / up to date} lines for the {top / high} end of the

market. He is {empowered / authorized} to receive on our behalf {samples / specimens} of the

{latest / newest} {watch and jewellery / jewellery and watch} products, {up / subject} to a {limit / maximum}

of ten thousand dollars. He will {carry / hold} a signed copy of this {letter / document}

as proof of identity. An order with his signature, which is {appended / attached}

{authorizes / allows} you to charge the cost to this company at the {above / head office}

address. We {fully / --} expect that our {level / volume} of orders will increase in

the {following / next} year and {trust / hope} that the new appointment will {be / prove}

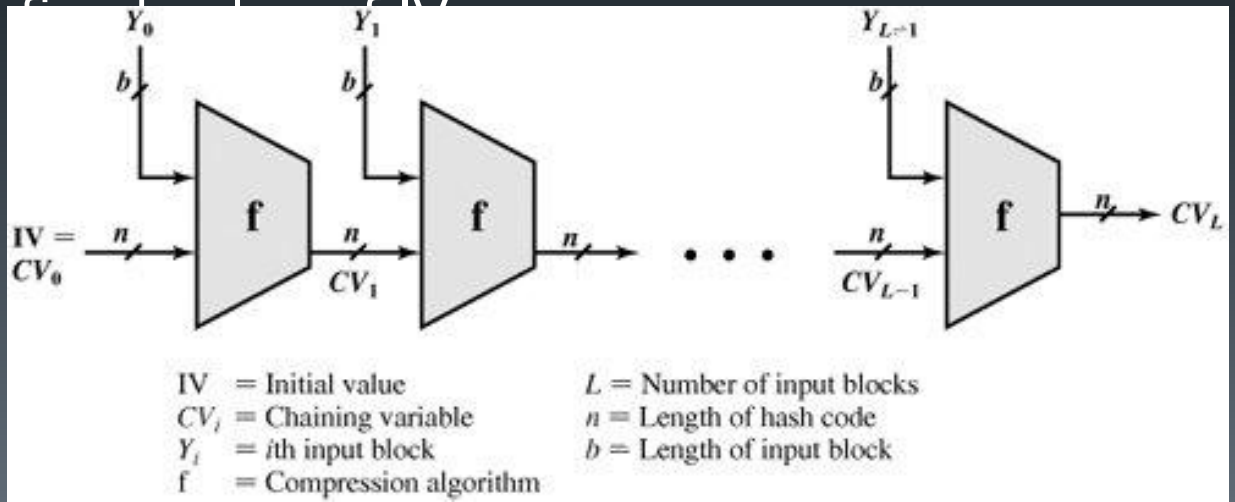
{advantageous / an advantage} to both our companies.

Security of Hash Functions

- Brute-Force Attacks:
  - One way : 2^n
  - Weak collision resistance : 2^n
  - Strong collision resistance : $2^{n/2}$

Cryptanalysis

- Cryptanalysis of hash functions focuses on the internal structure of f and is based on attempts to find efficient techniques for producing collisions for a single execution of f
- Once that is done, the attack must take into account the fi[illegible]



Message Authentication Code

- An alternative authentication technique involves the use of a secret key to generate a small fixed-size block of data, known as a cryptographic checksum or MAC that is appended to the message
- MAC = C(K, M)
- The message plus 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.
- If the received and calculated MAC is same the message is authenticated.

Message Authentication Code

- Arbitrary input length
- Fixed output length
- Message Authentication
- Integrity
- Non repudiation is not provided
- Also called as cryptographic checksum

HMAC

- Proposed in 1996
- Widely used in SSL/TLS
- Use 2 nested secret prefix MACs

Mutual Authentication

- Mutual authentication protocols enable communicating parties to satisfy themselves mutually about each other's identity and to exchange session keys
- Central to the problem of authenticated key exchange are two issues: confidentiality and timeliness
- To prevent masquerade and to prevent compromise of session keys, essential identification and session key information must be communicated in encrypted form
- Timeliness, is important because of the threat of message replays

Replay Attacks

- Simple replay
- Repetition that can be logged
- Repetition that cannot be detected
- Backward replay without modification
- One approach to coping with replay attacks is to attach a sequence number to each message used in an authentication exchange
- Timestamps
- Challenge/Response

One-Way Authentication

- One application for which encryption is growing in popularity is electronic mail (e-mail).
- The nature of electronic mail, and its chief benefit, is that it is not necessary for the sender and receiver to be online
- at the same time, Instead, the e-mail message is forwarded to the receiver's electronic mailbox, where it is buffered until the receiver is available to read it
- The "envelope" or header of the e-mail message must be in the clear, so that the message can be handled by the store-and-forward e-mail protocol
- Accordingly, the e-mail message should be encrypted such that the mail-handling system is not in possession of thedecryption key

Digital Signature

- Message authentication protects two parties who exchange messages from any third party.
- However, it does not protect the two parties against each other, several forms of dispute between the two are possible
- Mary may forge a different message and claim that it came from John. Mary would simply have to create a message and append an authentication code using the key that John and Mary share
- John can deny sending the message. Because it is possible for Mary to forge a message, there is no way to prove that John did in fact send the message

Digital Signature

- It must verify the author and the date and time of the signature
- It must to authenticate the contents at the time of the signature.
- It must be verifiable by third parties, to resolve disputes.

Digital Signature

- The signature must be a bit pattern that depends on the message being signed.
- The signature must use some information unique to the sender, to prevent both forgery and denial.
- It must be relatively easy to produce the digital signature
- It must be relatively easy to recognize and verify the digital signature.
- It must be computationally infeasible to forge a digital signature, either by constructing a new message for an existing digital signature or by constructing a fraudulent digital signature for a given message.
- It must be practical to retain a copy of the digital signature in storage.

Direct Digital Signature

- The direct digital signature involves only the communicating parties (source, destination).
- It is assumed that the destination knows the public key of the source.
- A digital signature may be formed by encrypting the entire message with the sender's private key or by encrypting a hash code of the message with the sender's private key
- If a sender later wishes to deny sending a particular message, the sender can claim that the private key was lost or stolen and that someone else forged his or her signature

Arbitrated Digital Signature

- The problems associated with direct digital signatures can be addressed by using an arbiter
- Every signed message from a sender X to a receiver Y goes first to an arbiter A, who subjects the message and its signature to a number of tests to check its origin and content.
- The message is then dated and sent to Y with an indication that it has been verified to the satisfaction of the arbiter.
- The presence of A solves the problem faced by direct signature schemes: that X might disown the message.

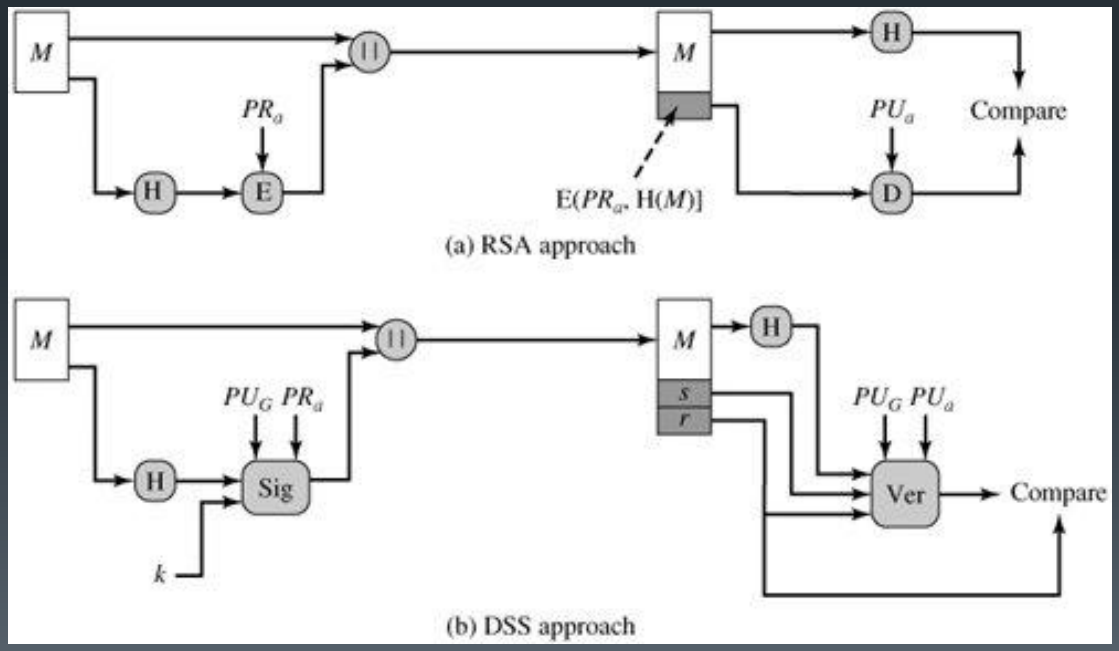
Arbitrated Digital Signature

(1) X $\longrightarrow$ A: $M \| \| E(K_{xa}, [ID_X \| \| H(M)])$
(2) A $\longrightarrow$ Y: $E(K_{ay}, [ID_X \| \| M \| \| E(K_{xa}, [ID_X \| \| H(M)]) \| \| T])$
(a) Conventional Encryption, Arbiter Sees Message
(1) X $\longrightarrow$ A: $ID_X \| \| E(K_{xy}, M) \| \| E(K_{xa}, [ID_X \| \| H(E(K_{xy}, M))])$
(2) A $\longrightarrow$ Y: $E(K_{ay}, [ID_X \| \| E(K_{xy}, M)]) \| \| E(K_{xa}, [ID_X \| \| H(E(K_{xy}, M)) \| \| T])$
(b) Conventional Encryption, Arbiter Does Not See Message
(1) X $\longrightarrow$ A: $ID_X \| \| E(PR_x, [ID_X \| \| E(PU_y, E(PR_x, M))])$
(2) A $\longrightarrow$ Y: $E(PR_a, [ID_X \| \| E(PU_y, E(PR_x, M)) \| \| T])$
(c) Public-Key Encryption, Arbiter Does Not See Message

Notation:	
X	= sender
Y	= recipient
A	= Arbiter
M	= message
T	= timestamp

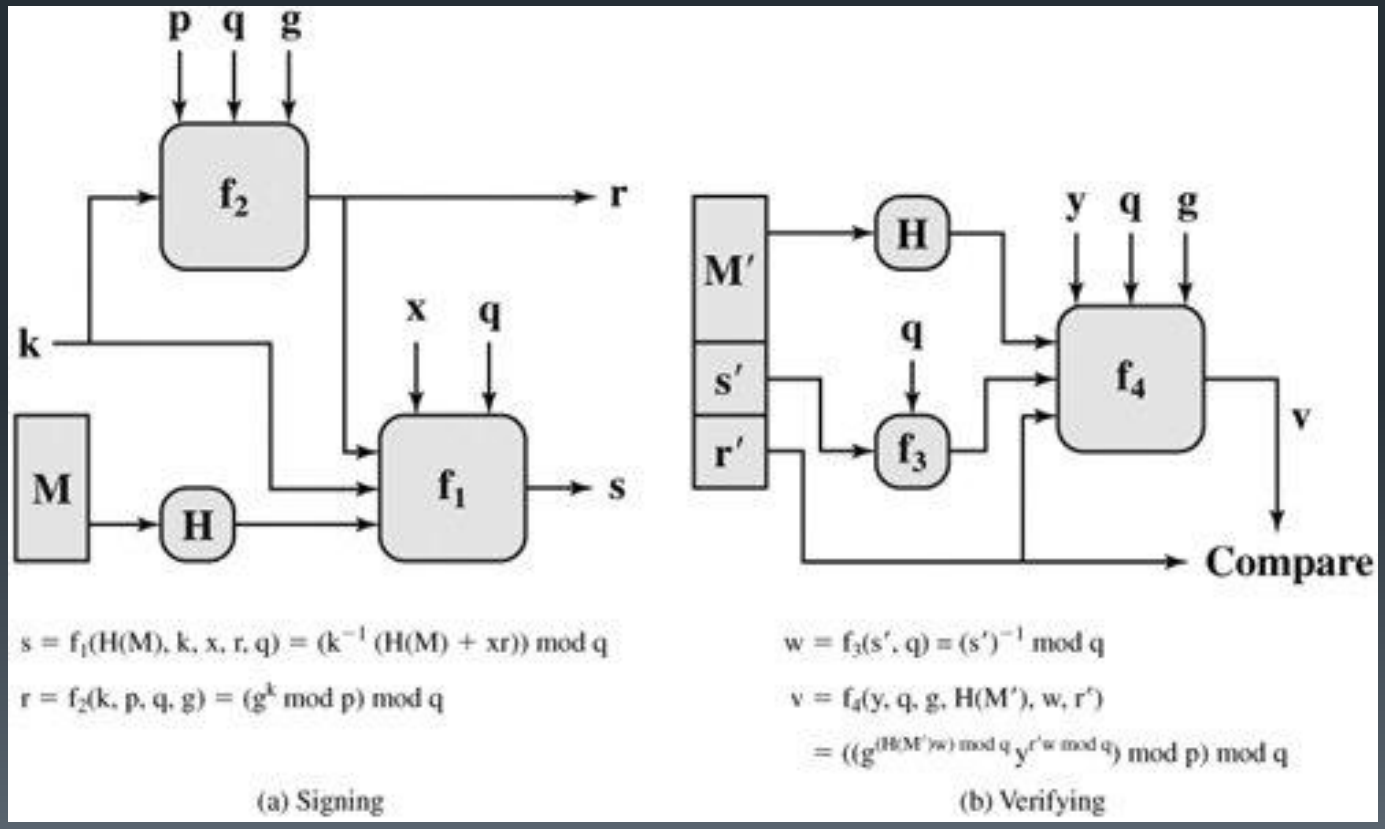
Digital Signature Standard (DSS)

- The DSS uses an algorithm that is designed to provide only the digital signature function, not for encryption or key exchange



Global Public-Key Components	
p	prime number where $2^{L-1} < p < 2^L$ for $512 \leq L \leq 1024$ and L a multiple of 64; i.e., bit length of between 512 and 1024 bits in increments of 64 bits
q	prime divisor of $(p-1)$, where $2^{159} < q < 2^{160}$; i.e., bit length of 160 bits
g	$= h^{(p-1)/q} \bmod p$, where h is any integer with $1 < h < (p-1)$ such that $h^{(p-1)/q} \bmod p > 1$
User's Private Key	
x	random or pseudorandom integer with $0 < x < q$
User's Public Key	
y	$= g^x \bmod p$
User's Per-Message Secret Number	
k	= random or pseudorandom integer with $0 < k < q$
Signing	
r	$= (g^k \bmod p) \bmod q$
s	$= [k^{-1}(H(M) + xr)] \bmod q$
Signature = (r, s)	
Verifying	
w	$= (s')^{-1} \bmod q$
u1	$= [H(M')w] \bmod q$
u2	$=(r')w \bmod q$
v	$= [(g^{u1} y^{u2}) \bmod p] \bmod q$
TEST: $v = r'$	
M	= message to be signed
$H(M)$	= hash of M using SHA-1
M', r', s'	= received versions of M, r, s

Digital Signature Standard (DSS)

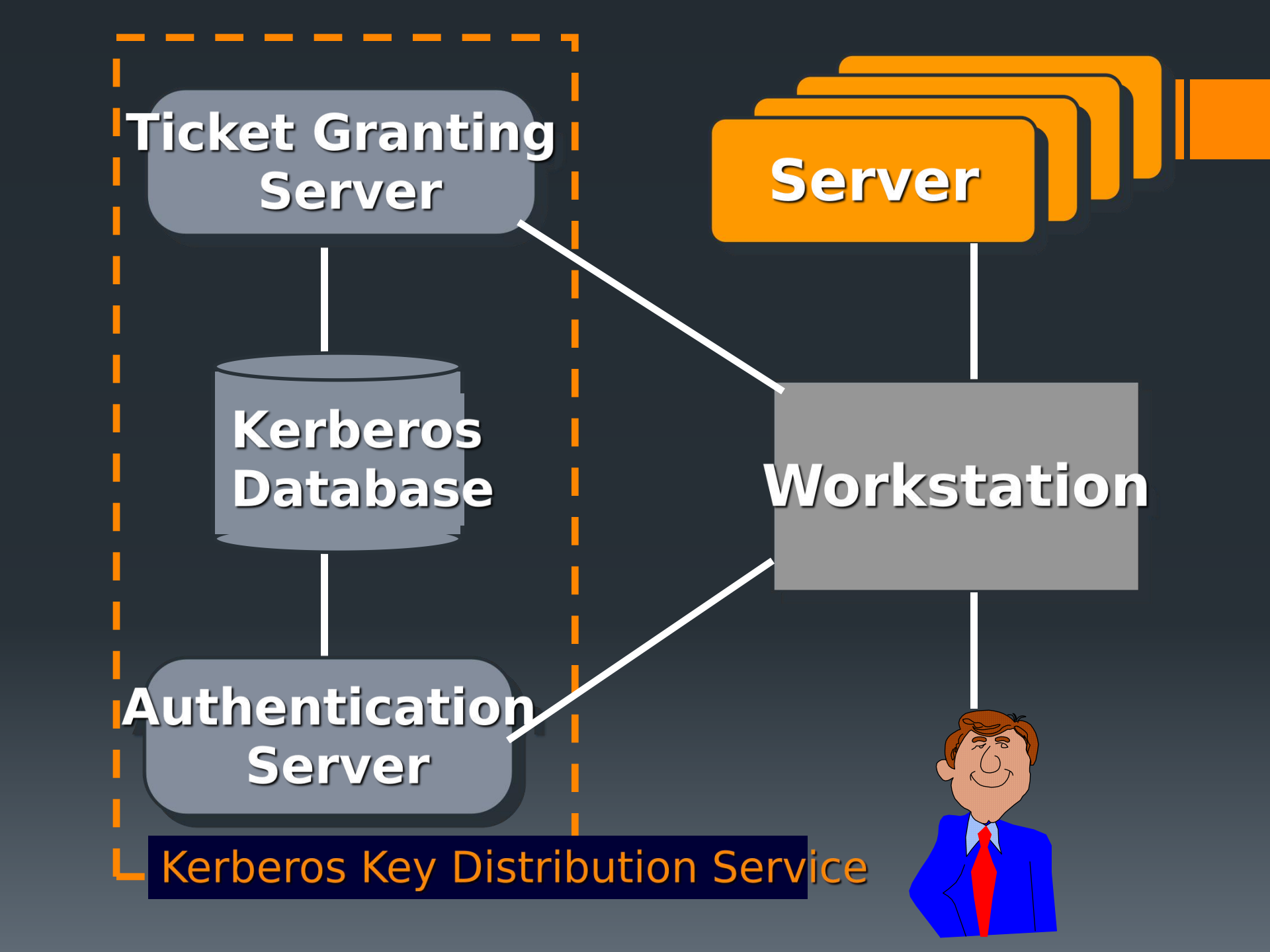


Kerberos

- Network authentication protocol
- Developed at MIT in the mid 1980s
- Available as open source or in supported commercial software
- When using authentication based on cryptography, an attacker listening to the network gains no information that would enable it to falsely claim another's identity. Kerberos is the most commonly used example of this type of authentication technology.

Motivation

- Without knowledge of identity of person requesting an operation difficult to decide if it should be allowed
- Traditional authentication methods are not suitable for use in computer network where attackers can monitor network traffic and intercept passwords.
- Use of strong authentication methods are crucial
- Easy for administrator to manage password by storing them centrally
- No clear text password are transmitted
- Different service with same password, single sign on


Ticket Granting Server
Server
Kerberos Database
Workstation
Authentication Server
Kerberos Key Distribution Service

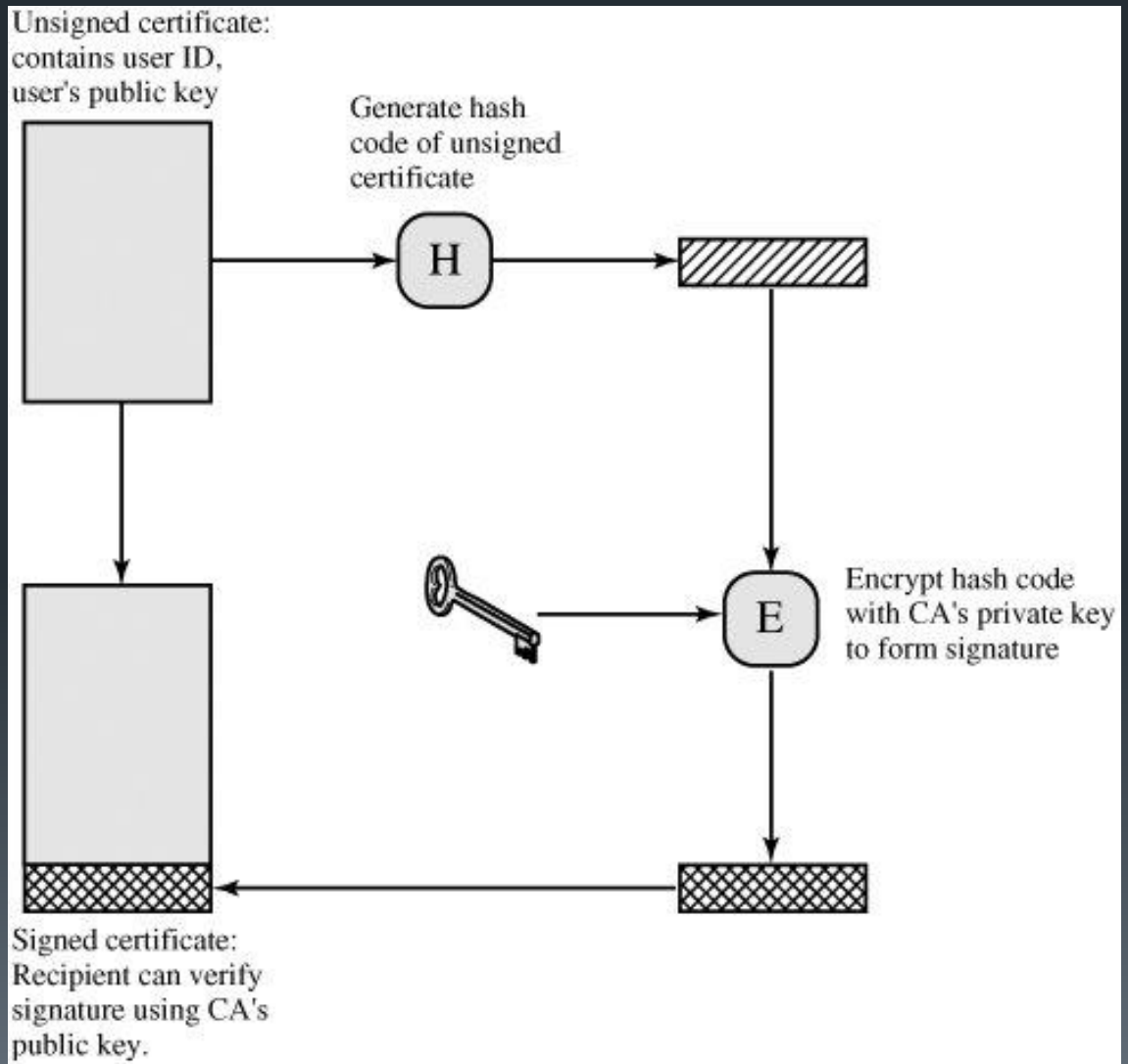
X.509 Authentication Service

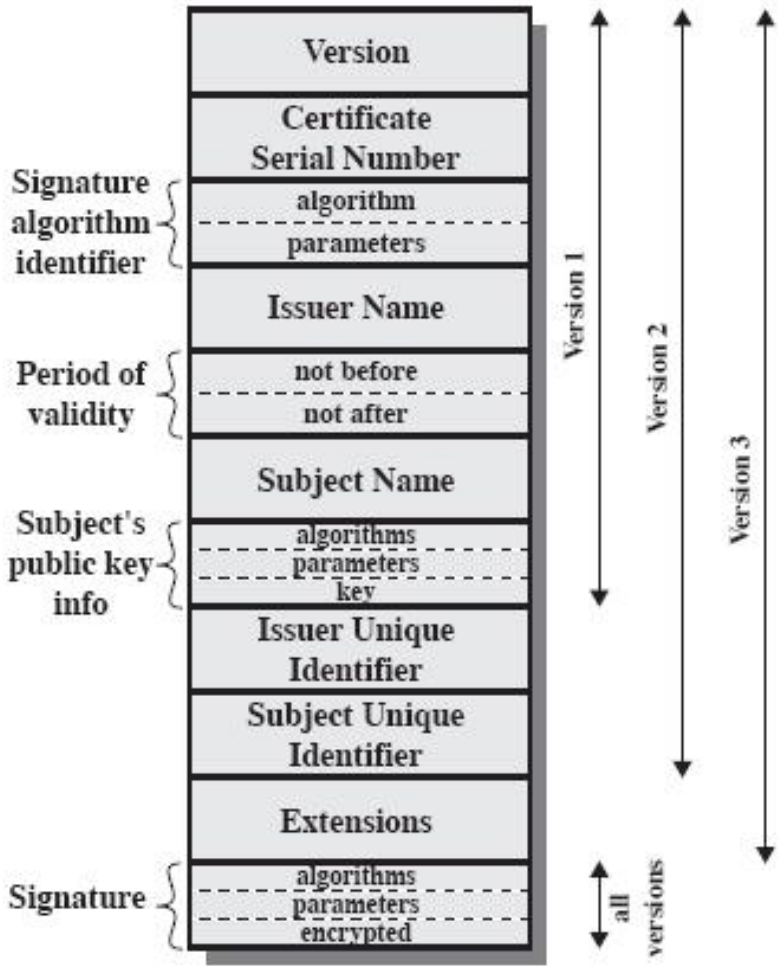
- ITU-T recommendation X.509 is part of the X.500 series of recommendations that define a directory Service
- The heart of X.509 scheme is the public-key certificate associated with each user.
- These user certificates are assumed to be created by some trusted certificate authority (CA) and placed in the directory by the CA or by the user.
- The directory itself is not responsible for the creation of public keys or for the certification function; it merely provides an easily accessible location for users to obtain certificates.

Certification Authorities

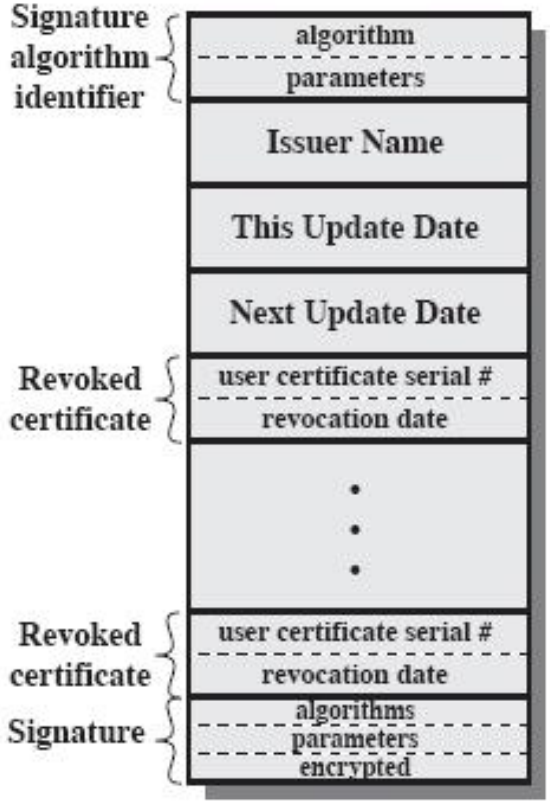
- CA is a general designation for any entity that controls the authentication services and the management of Certificates, also called the issuer.
- The CA can be public (a bank that issues certificates to allow its clients to access their bank account), commercial (a service provider that sells certificates to other parties, such as Verisign) or private (a company that issues certificates to allow its employees to perform job duties).

X.509


Unsigned certificate:
contains user ID,
user's public key
Generate hash
code of unsigned
certificate
H
E
Encrypt hash code
with CA's private key
to form signature
Signed certificate:
Recipient can verify
signature using CA's
public key.



(a) X.509 Certificate



(b) Certificate Revocation List

X.509

- The unique identifier fields were added in version 2 to handle the possible reuse of subject and/or issuer names over time. These fields are rarely used.
- The standard uses the following notation to define a certificate:
  - CA<<A>> = CA{V,SN,AI,CA,T_A,A,Ap} ,Where

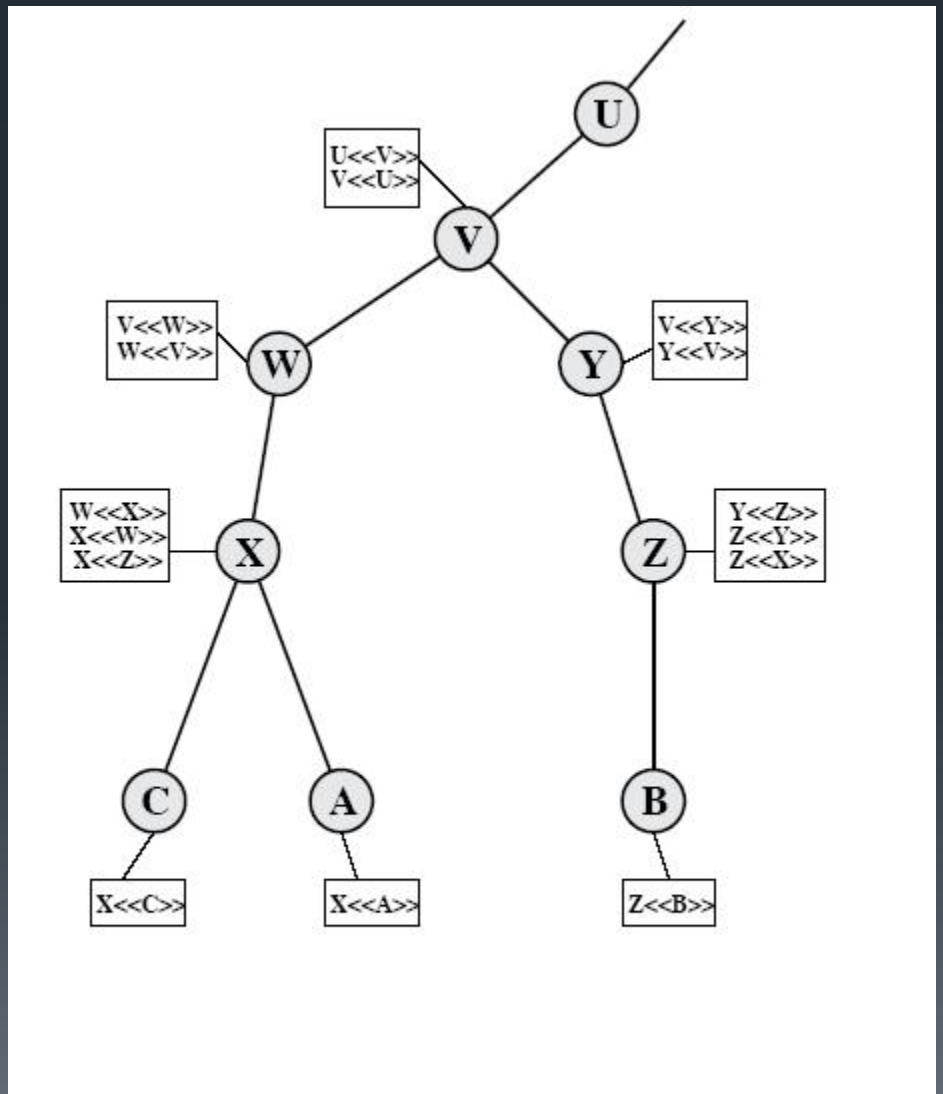
    Y<<X>> = certificate of user X issued by certification authority Y

    Y{I} = the signing of I by Y. It consists of I with an encrypted hash code appended
- The CA signs the certificate with its private key. If the corresponding public key is known to a user, then that user can verify that a certificate signed by the CA is valid.

X.509

- Now suppose that A has obtained a certificate from certification authority X1 and B has obtained a certificate from CA X2.
- A has used a chain of certificates to obtain B’s public key. In the notation of X.509, this chain is expressed as
  - X1<<X2>>X2<<B>>
- In the same fashion, B can obtain A’s public key with the chain:
  - X2<<X1>>X1<<A>>

X.509 Hierarchy



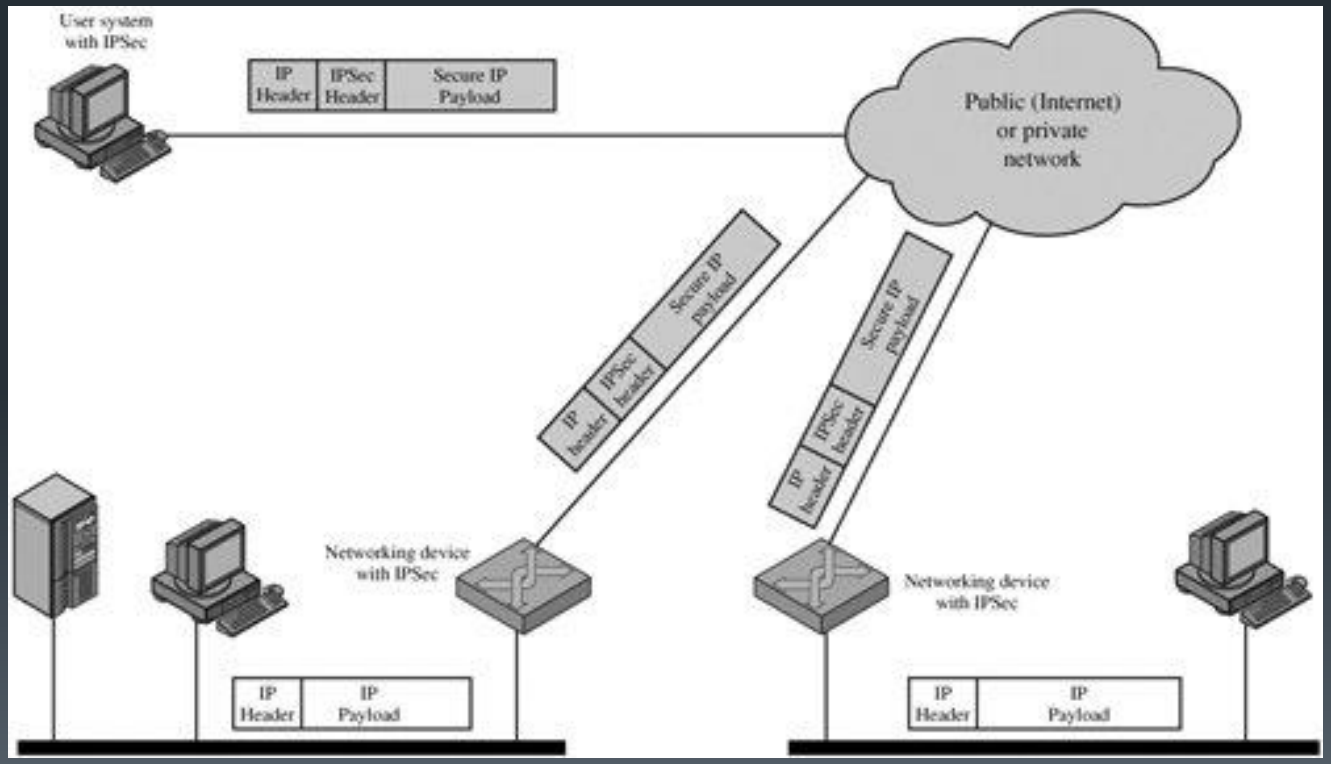
IPSec

- IPSec provides the capability to secure communications across a LAN, across private and public WANs, and across the Internet
- IPSec is combination of various protocols and algorithms to build a secure tunnel between source and destination
- Secure branch office connectivity over the Internet
- Secure remote access over the Internet
- Enhancing electronic commerce security
- It can encrypt and/or authenticate *all traffic at the IP level*

Why IPSec?

- Internet Protocol (IP) is not secure, it was designed in early stages where security was not issue.
- Possible security issues:
  - Source spoofing
  - Replay packets
  - Data integrity or confidentiality
- Ipsec is designed to provide interoperable, high quality, cryptographically based security for IPv4 and IPv6

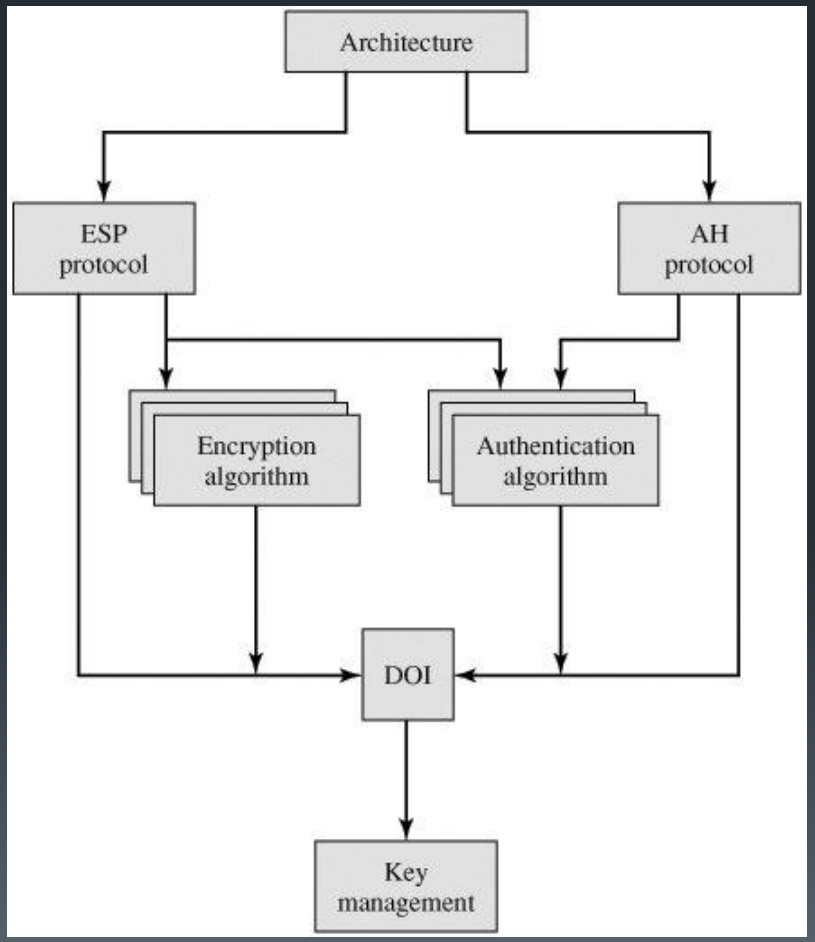
An IP Security Scenario



IPSec Services

	AH	ESP (encryption only)	ESP (encryption plus authentication)
Access control	✔	✔	✔
Connectionless integrity	✔		✔
Data origin authentication	✔		✔
Rejection of replayed packets	✔	✔	✔
Confidentiality		✔	✔
Limited traffic flow confidentiality		✔	✔

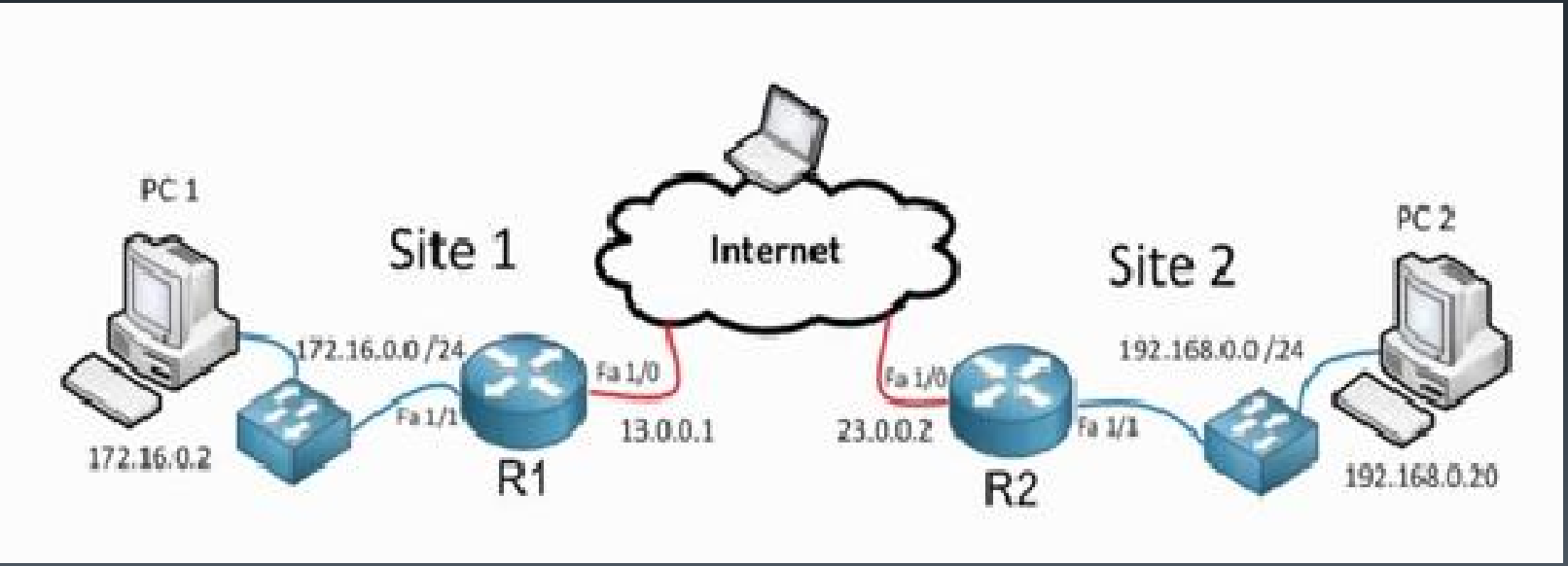
IPSec Architecture

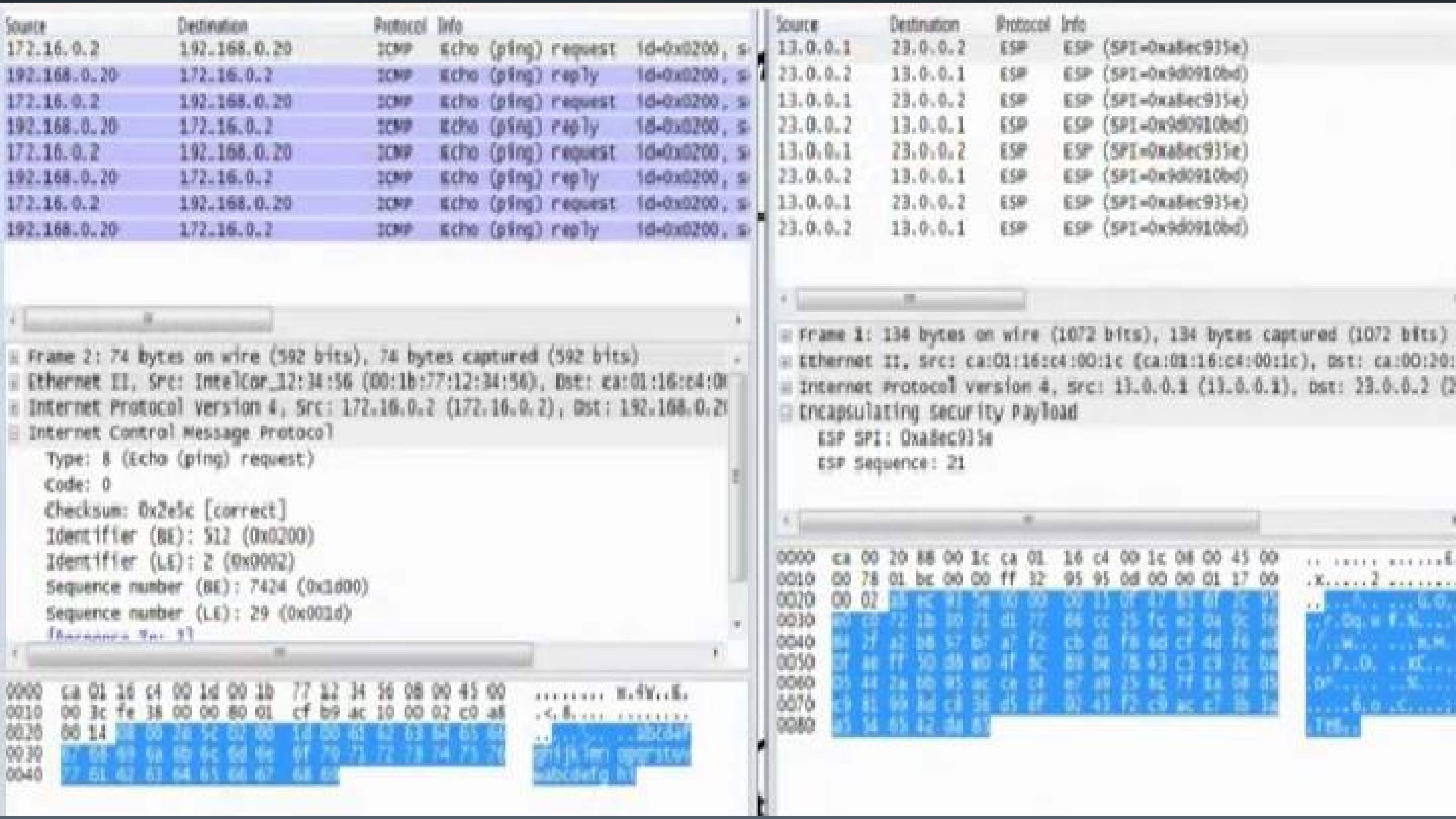


IPSec Modes

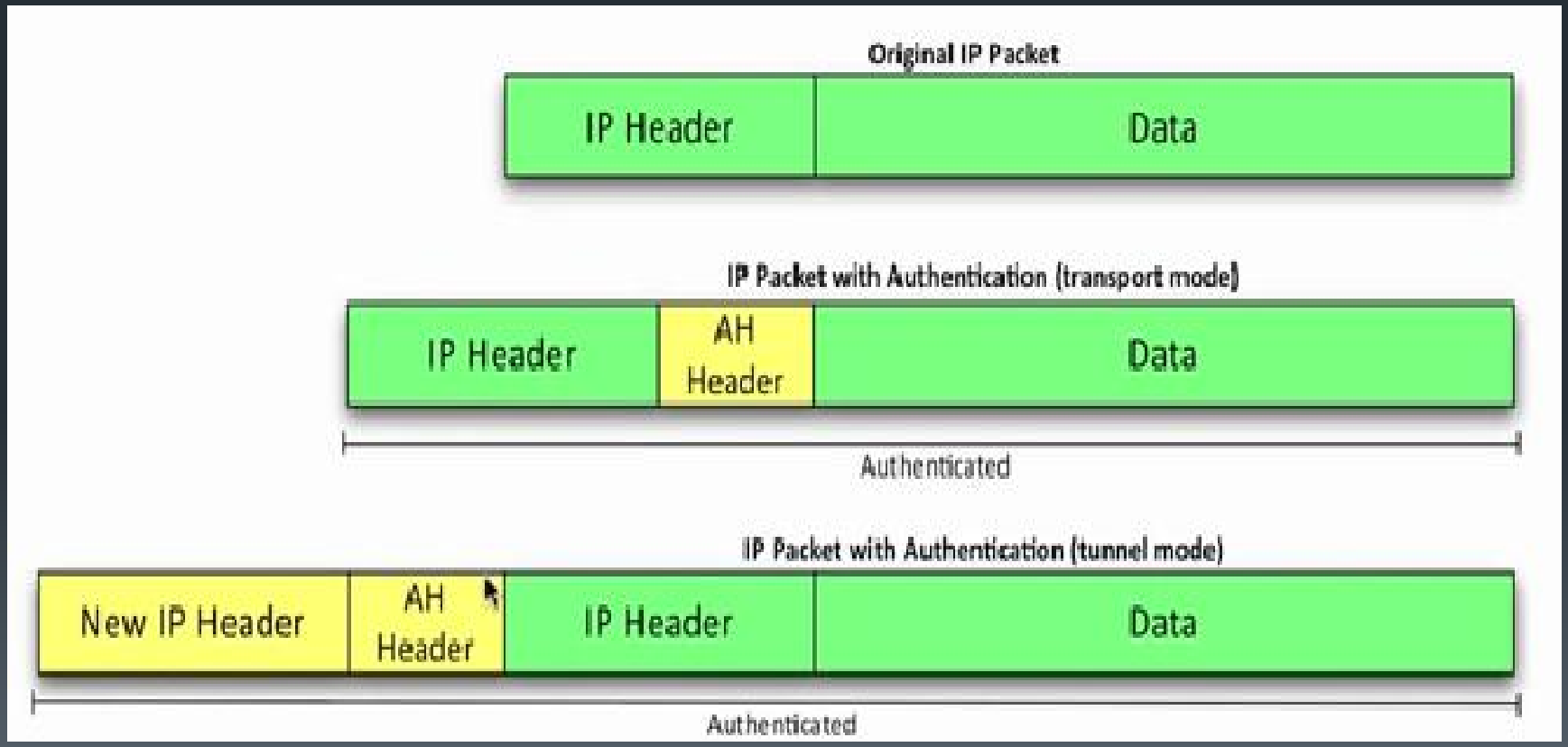
- Tunnel Mode
  - Entire IP packet is encrypted and becomes the data component of a new IP packet
  - Frequently used in IPSec site to site VPN
- Transport Mode
  - IPSec header is inserted into the IP packet
  - No new packet is created
  - Works well in network where increasing a packet's size could cause issue
  - Frequently used for remote access VPNs

Tunneling

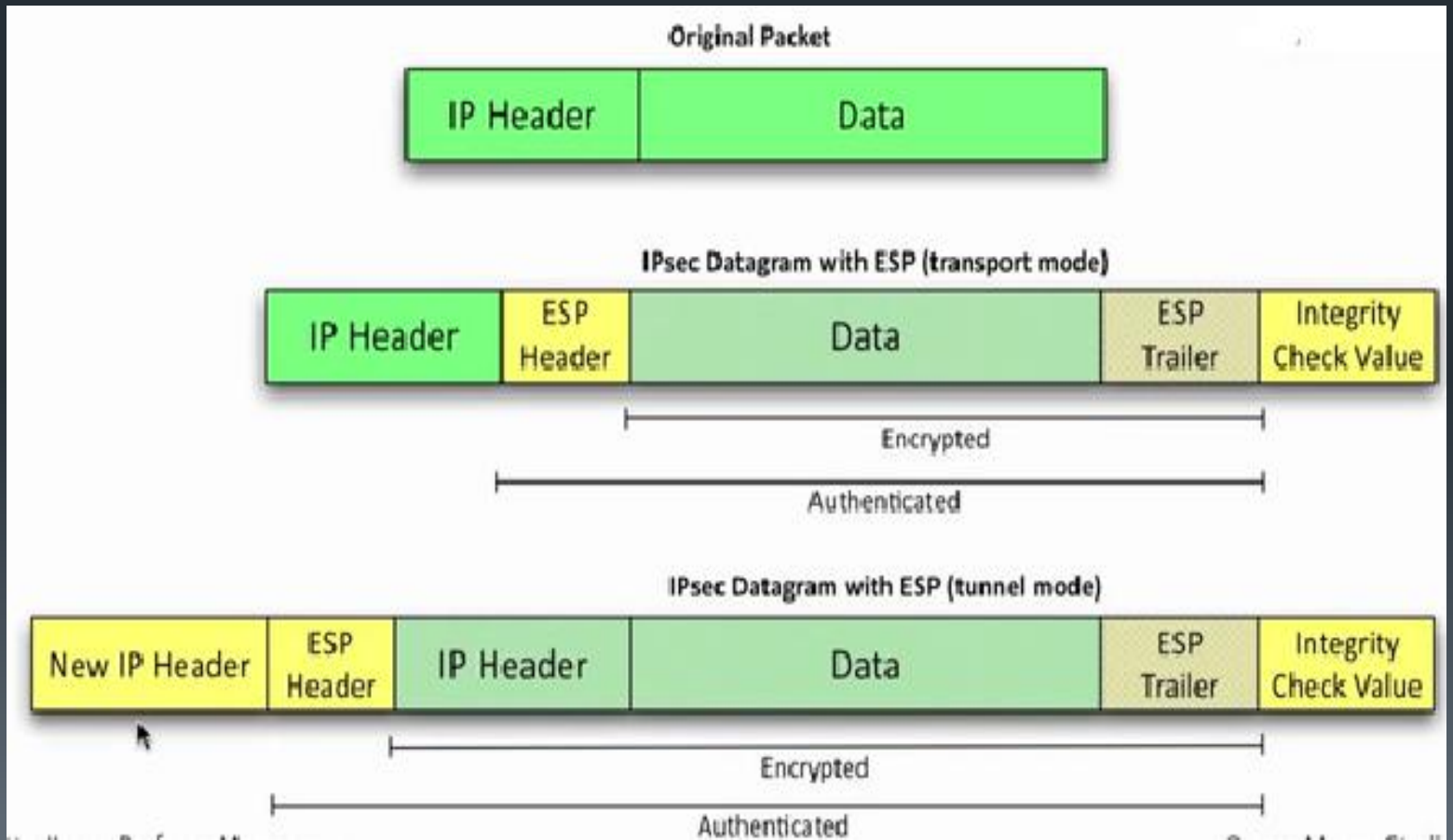

PC 1
Site 1
Internet
Site 2
PC 2
172.16.0.0 /24
Fa 1/0
Fa 1/0
192.168.0.0 /24
Fa 1/1
Fa 1/1
172.16.0.2
13.0.0.1
23.0.0.2
R1
R2
192.168.0.20


Source Destination Protocol Info
172.16.0.2 192.168.0.20 ICMP Echo (ping) request id=0x0200,
192.168.0.20 172.16.0.2 ICMP Echo (ping) reply id=0x0200,
172.16.0.2 192.168.0.20 ICMP Echo (ping) request id=0x0200,
192.168.0.20 172.16.0.2 ICMP Echo (ping) reply id=0x0200,
172.16.0.2 192.168.0.20 ICMP Echo (ping) request id=0x0200,
192.168.0.20 172.16.0.2 ICMP Echo (ping) reply id=0x0200,
172.16.0.2 192.168.0.20 ICMP Echo (ping) request id=0x0200,
192.168.0.20 172.16.0.2 ICMP Echo (ping) reply id=0x0200,
Frame 2: 74 bytes on wire (592 bits), 74 bytes captured (592 bits)
Ethernet II, Src: IntelCor_12:34:56 (00:1b:77:12:34:56), Dst: ca:01:16:c4:0
Internet Protocol Version 4, Src: 172.16.0.2 (172.16.0.2), Dst: 192.168.0.2
Internet Control Message Protocol
Type: 8 (Echo (ping) request)
Code: 0
Checksum: 0x2e5c [correct]
Identifier (BE): 512 (0x0200)
Identifier (LE): 2 (0x0002)
Sequence number (BE): 7424 (0x1d00)
Sequence number (LE): 29 (0x001d)
Source Destination Protocol Info
13.0.0.1 23.0.0.2 ESP ESP (SPI=0xa8ec935e)
23.0.0.2 13.0.0.1 ESP ESP (SPI=0x9d0910bd)
13.0.0.1 23.0.0.2 ESP ESP (SPI=0xa8ec935e)
23.0.0.2 13.0.0.1 ESP ESP (SPI=0x9d0910bd)
13.0.0.1 23.0.0.2 ESP ESP (SPI=0xa8ec935e)
23.0.0.2 13.0.0.1 ESP ESP (SPI=0x9d0910bd)
13.0.0.1 23.0.0.2 ESP ESP (SPI=0xa8ec935e)
23.0.0.2 13.0.0.1 ESP ESP (SPI=0x9d0910bd)
Frame 1: 134 bytes on wire (1072 bits), 134 bytes captured (1072 bits)
Ethernet II, Src: ca:01:16:c4:00:1c (ca:01:16:c4:00:1c), Dst: ca:00:20:
Internet Protocol Version 4, Src: 13.0.0.1 (13.0.0.1), Dst: 23.0.0.2 (2
Encapsulating Security Payload
ESP SPI: 0xa8ec935e
ESP Sequence: 21

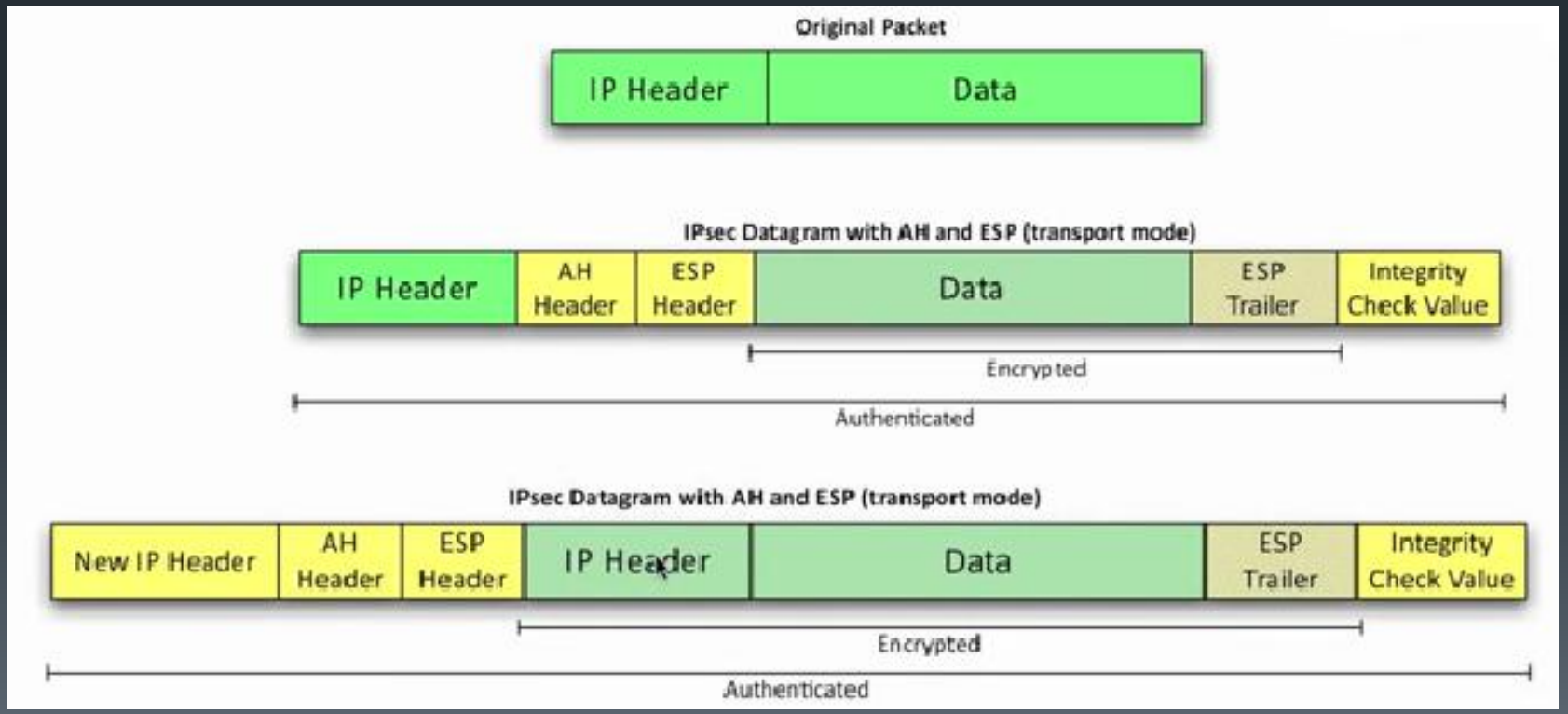
Authentication header



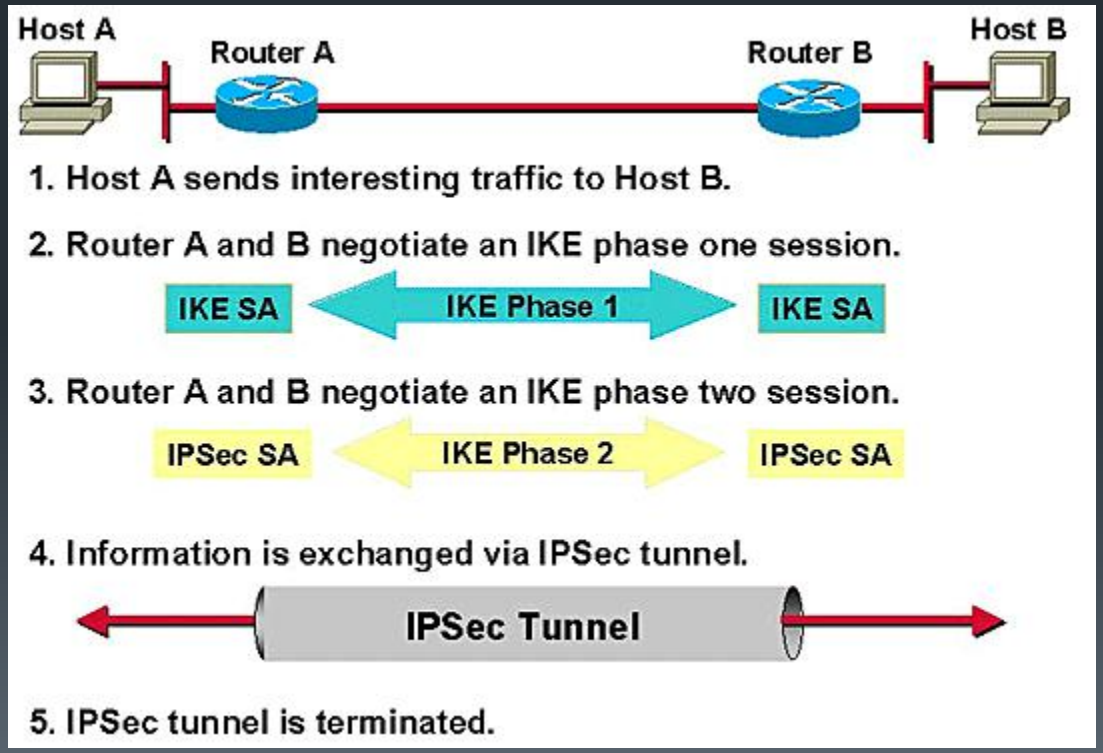
Encapsulating Security Payload



IPSEC with AH & ESP



IPSec Phases



Web Security

- Virtually all businesses, most government agencies, and many individuals now have Web sites, Internet security is very critical and important.
- The most widely used protocols to provide Internet security are:
  - SSL (Secure Socket Layer)
  - TLS (Transport Layer Security)
  - HTTPS (Secure Hypertext Transfer Protocol)
  - S/MIME (Secure Multimedia Internet Mail Exchange)
  - SET (Secure Electronic Transaction)

Internet

- Web Pages
  - Static Web Pages
  - Dynamic Web Pages
- Web Browser
- Web Server

Secure Socket Layer (SSL)

- Netscape Corporation developed SSL in 1994, since then it has become the world's most popular Web security protocol
- SSL is an Internet protocol for secure exchange of information between a web browser and a web server.
- Provides Authentication and Confidentiality
- All the major web browsers support SSL
- SSL is located between application layer and transport layer of TCP/IP model.

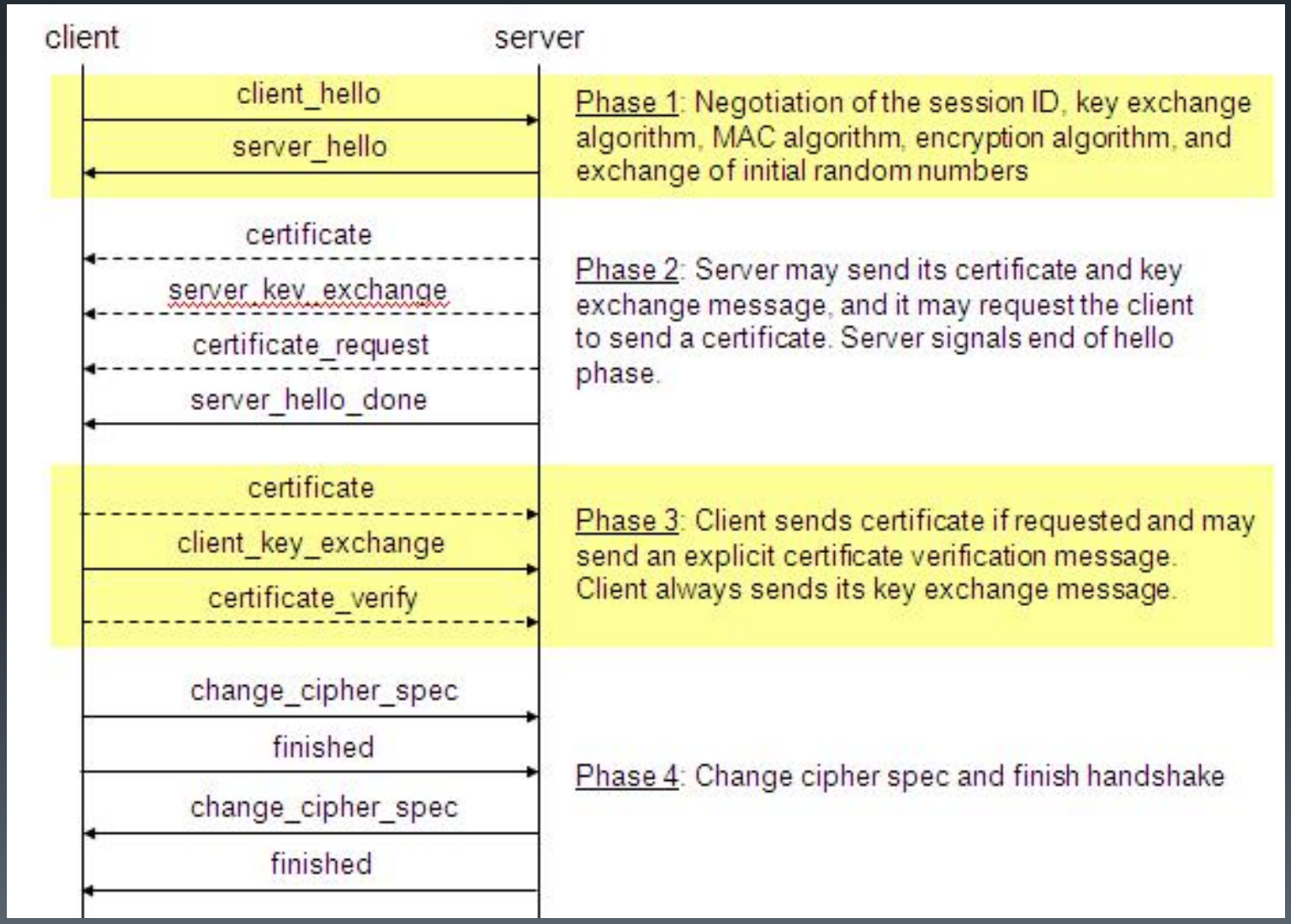
How SSL Works?

- SLL has three sub protocols:
  - Handshake Protocol
  - Record Protocol
  - Alert Protocol

Handshake Protocol

- The handshake protocol is made up of four phases,
  - Establish security capabilities
  - Server authentication and key exchange
  - Client authentication and key exchange
  - Finish

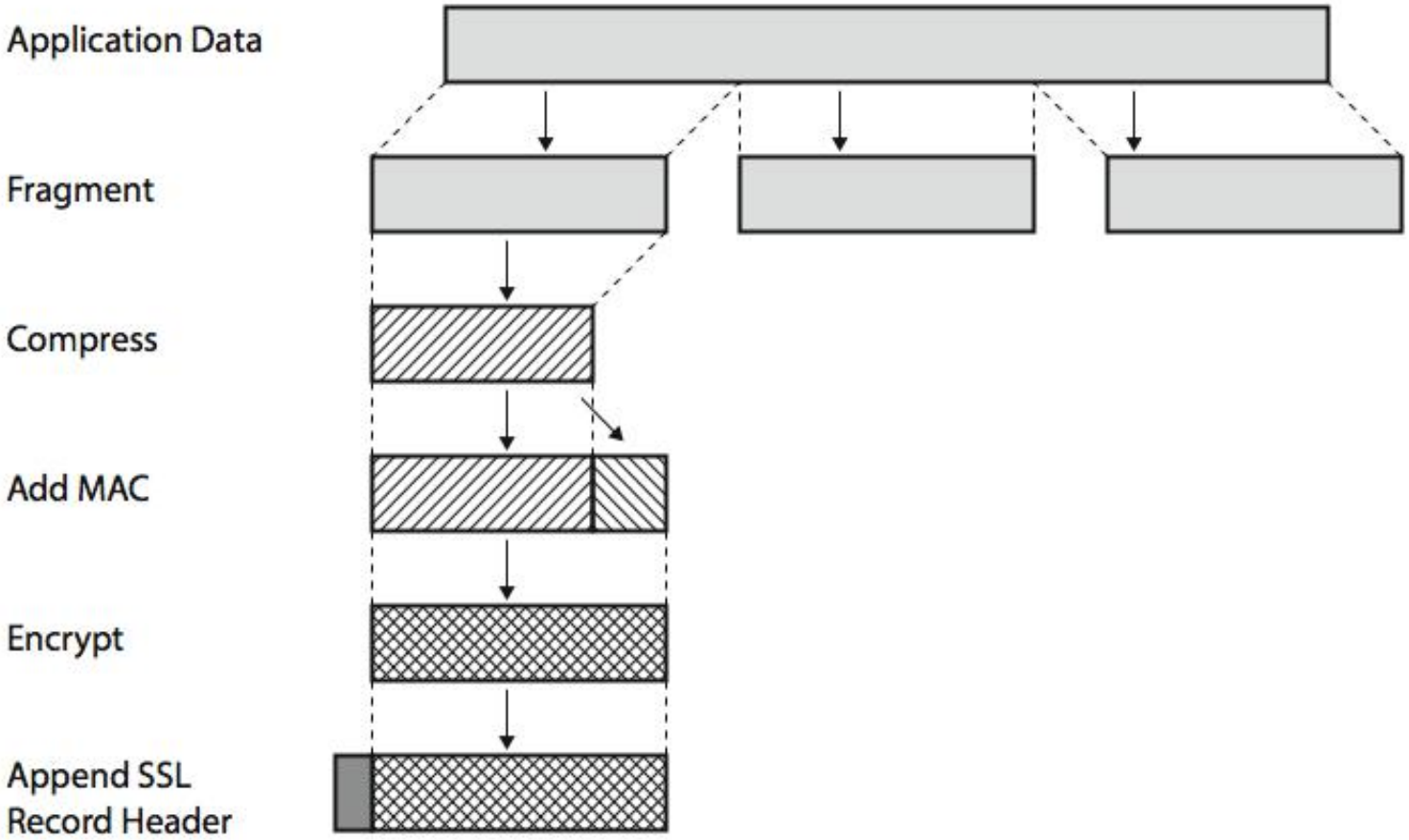
Handshake Protocol



The Record protocol

- After the client and server have authenticated each other and decided what algorithms to use for secure communication, record protocol comes into picture.
- SSL Record Protocol provides two services.
- Message integrity
  - using a MAC with a shared secret key
  - similar to HMAC but with different padding
  - hash functions: MD5, SHA-1
- Message confidentiality
  - using symmetric encryption with a shared secret key
  - Encryption algorithms: AES, IDEA, RC2-40, DES-40, DES, 3DES, RC4-40, RC4-128

The Record protocol



The Alert Protocol

- When either client or server detects an error the detecting party sends an alert message to the other party.
- If the error is fatal both parties immediately closes the SSL connection.
- Each alert message consists of two bytes. First byte signifies the type of error , if it is warning this byte contains 1, if it is fatal error it contains 2.
- The second byte specifies actual error

Fatal Errors

- Unexpected message: An inappropriate message was received.
- Bad record MAC : An incorrect MAC was received.
- Decompression failure: The decompression function received improper input
- Handshake failure: Sender was unable to negotiate an acceptable set of security parameters given the options available.
- Illegal parameter: A field in a handshake message was out of range or inconsistent with other fields.

Secure Electronics Transaction (SET)

- Secure Electronic Transaction (SET) is a system for ensuring the security of financial transactions on the Internet.
- It was supported initially by MasterCard, Visa, Microsoft, Netscape, and others.
- It provides secure communication channel among all parties involved in e-commerce transaction
- Authentication by use of digital certificate
- Ensures confidentiality

SET Participants

- Cardholder
- Merchant
- Issuer
- Acquirer
- Certification Authority

SET Process

1. The customer opens an account
2. The customer receives a certificate/card
3. The merchant receives a certificate
4. The customer places an order
5. The merchant is verified
6. The order and payment details are sent
7. The merchant requests payment authorization
8. The acquirer authorizes payment
9. The merchant confirms the order
10. The merchant provides good/service
11. The merchant requests payment

Secure Electronics Transaction

- There are two main concerns in online payment
- The credit card details travel in clear text:
  - This can be taken care by SSL
- The credit card number is available to the merchant:
  - SET encrypts the payment information using one time session key (generated by cardholder’s computer) , then wraps this key in digital envelope encrypted by acquirer's public key.