SECURITY

Introduction

- Security is one of the most important principles , since security need to be pervasive through the system.
- Security Policy
  - Security policy is just a statement about what is allowed and not allowed to do in a system.
- Security Mechanism
  - Security mechanism is a procedure how to implement the security policy. It is said to be a tool, methodology or procedures for security enforcement.

Historical context: the evolution of security needs

	1965-75	*1975-89*	*1990-99*	*Current*
Platforms	Multi-user timesharing computers	Distributed systems based on local networks	The Internet, wide-area services	The Internet + mobile devices
Shared resources	Memory, files	Local services (e.g. NFS), local network	Email, web sites, Internet commerce	Distributed objects, mobile code
Security requirements	User identification and authentication	Protection of services	Strong security for commercial transactions	Access control for individual objects, secure mobile code
Security management environment	Single authority, single authorization database (e.g. /etc/passwd)	Single authority, delegation, replicated authorization databases (e.g. NIS)	Many authorities, no network-wide authorities	Per-activity authorities, groups with shared responsibilities

Threats and Attacks

- Threat : is a possible danger that might exploit a vulnerability to breach security and thus cause possible harm.

- **Classes of threats**
  - *Leakage: Refers to the acquisition of information by unauthorized recipients.*
  - *Tampering: Refers to the unauthorized alteration of information.*
  - *Vandalism: Refers to interference with the proper operation of a system without gain* to the perpetrator.

- **Attack** : is any attempt to destroy, expose, alter, disable, steal or gain unauthorized access to or make unauthorized use of an asset.
- **Methods of attacks**
  - ***Eavesdropping****: Obtaining copies of messages without authority.*
  - ***Masquerading:*** *Sending or receiving messages using the identity of another* principal without their authority.
  - ***Message tampering:*** *Intercepting messages and altering their contents before* passing them on to the intended recipient. The *man-in-the-middle attack is a form of* message tampering
  - ***Replaying:*** *Storing intercepted messages and sending them at a later date.*
  - ***Denial of service:*** *Flooding a channel or other resource with messages in order to* deny access for others.

Information leakage

- Information Leakage is an application weakness where an application reveals sensitive data, such as technical details of the web application, environment, or user-specific data.
- The approach taken is to assign security levels to information and channels and to analyze the flow of information into channels with the aim of ensuring that high-level information cannot flow into lower-level channels.

Securing electronic transactions

- *Email*
- *Purchase of goods and services*
- *Banking transactions*
- *Micro-transactions*
- Transactions such as these can be safely performed only when they are protected by appropriate security policies and mechanisms.

- Security policies for Internet **vendors and buyers** lead to the following requirements for securing web purchases:
  - Authenticate the vendor to the buyer, so that the buyer can be confident that they are in contact with a server operated by the vendor with whom they intended to deal.(authentication)
  - Keep the buyer's credit card number and other payment details from falling into the hands of any third party and ensure that they are transmitted unaltered from the buyer to the vendor.(leakage)
  - If the goods are in a form suitable for downloading, ensure that their content is delivered to the buyer without alteration and without disclosure to third parties.(tampering)
  - Authenticate the identity of the account holder to the bank before giving them access to their account.(Access Control)

Designing secure systems

- Security is about avoiding disasters and minimizing mishaps.
- To demonstrate the validity of the security mechanisms employed in a system, the system's designers must first construct a list of threats – methods by which the security policies might be violated – and show that each of them is prevented by the mechanisms employed.

Overview of Security Techniques

- This introduces the reader to some of the more important techniques and mechanisms for securing distributed systems and applications.

Worst case Assumptions & Design Guidelines

Interfaces are exposed:

Networks are insecure:

Limit the lifetime and scope of each secret:

Algorithms and program code are available to attackers:

Attackers may have access to large resources:

Minimize the trusted base:

Security Notations

Alice	First participant
Bob	Second participant
Carol	Participant in three- and four-party protocols
Dave	Participant in four-party protocols
Eve	Eavesdropper
Mallory	Malicious attacker
Sara	A server

K_A	Alice's secret key
K_B	Bob's secret key
K_{AB}	Secret key shared between Alice and Bob
K_{Apriv}	Alice's private key (known only to Alice)
K_{Apub}	Alice's public key (published by Alice for all to read)
$\{M\}_K$	Message M encrypted with key K
$[M]_K$	Message M signed with key K

Cryptography

- Encryption is the process of encoding a message in such a way as to hide its contents.
- Secure algorithms for encrypting and decrypting messages are all based on the use of secrets called *keys.*
- Two main classes of encryption algorithm
  - *shared secret keys*
  - *public/private key pairs*

Uses of cryptography

- **Secrecy and integrity**
  - Cryptography is used to maintain the secrecy and integrity of information whenever it is exposed to potential attacks.
  - It maintains the secrecy of the encrypted message as long as the decryption key is not *compromised.*
  - *It* also maintains the integrity of the encrypted information, provided that some redundant information such as a checksum is included and checked.

- **Scenario 1**. Secret communication with a shared secret key: Alice wishes to send some information secretly to Bob. Alice and Bob share a secret key *KAB.*

  1. Alice uses *KAB and an agreed encryption function E(KAB, M) to encrypt and send* any number of messages {*Mi}KAB to Bob.*

  2. Bob decrypts the encrypted messages using the corresponding decryption function *D(KAB, M).*

- **Issues:**
  - ***Key distribution****: How can Alice send a shared key KAB to Bob securely?*
  - ***Freshness of communication****: How does Bob know that any {Mi} isn't a copy of an earlier encrypted message from Alice that was captured by Mallory and replayed later?*

Authentication

- Cryptography is used in support of mechanisms for authenticating communication between pairs of principals.
- key is known only to two parties.

Scenario 2:

Authenticated communication with a server

- Bob is a file server;
- Sara is an authentication service.
- Sara shares secret key K_A with Alice and secret key K_B with Bob.
  1. Alice sends an (unencrypted) message to Sara stating her identity and requesting a *ticket for access to Bob.*
  2. Sara sends a response to Alice. $\{\{Ticket\}_{K_B}, K_{AB}\}_{K_A}$. It is encrypted in K_A and consists of a ticket (to be sent to Bob with each request for file access) encrypted in K_B and a new secret key K_{AB}.
  3. Alice uses K_A to decrypt the response.
  4. Alice sends Bob a request R to access a file: $\{Ticket\}_{K_B}$, Alice, R.
  5. The ticket is actually $\{K_{AB}, Alice\}_{K_B}$. Bob uses K_B to decrypt it, checks that Alice's name matches and then uses K_{AB} to encrypt responses to Alice.

Scenario 3.
Authenticated communication with public keys

- Bob has a public/private key pair <KBpub,KBpriv>

1. Alice obtains a certificate that was signed by a trusted authority stating Bob's public key Kbpub.
2. Alice creates a new shared key KAB, encrypts it using Kbpub using a public-key algorithm and sends the result to Bob.
3. Bob uses the corresponding private key KBpriv to decrypt it.

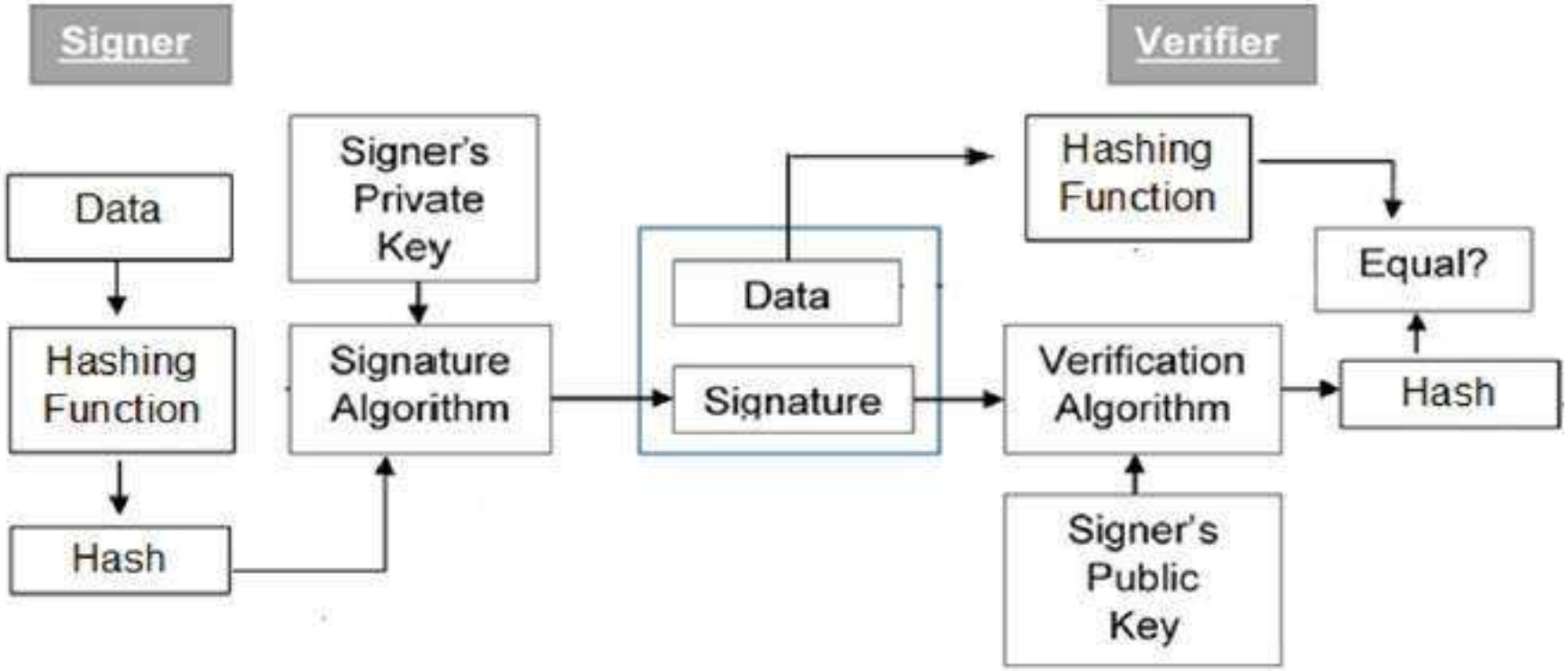
   (If they want to be sure that the message hasn't been tampered with, Alice can add an agreed value to it and Bob can check it.)

- Mallory might intercept Alice’s initial request to a key distribution service for Bob’s public-key certificate and send a response containing his own public key. He can then intercept all the subsequent messages.

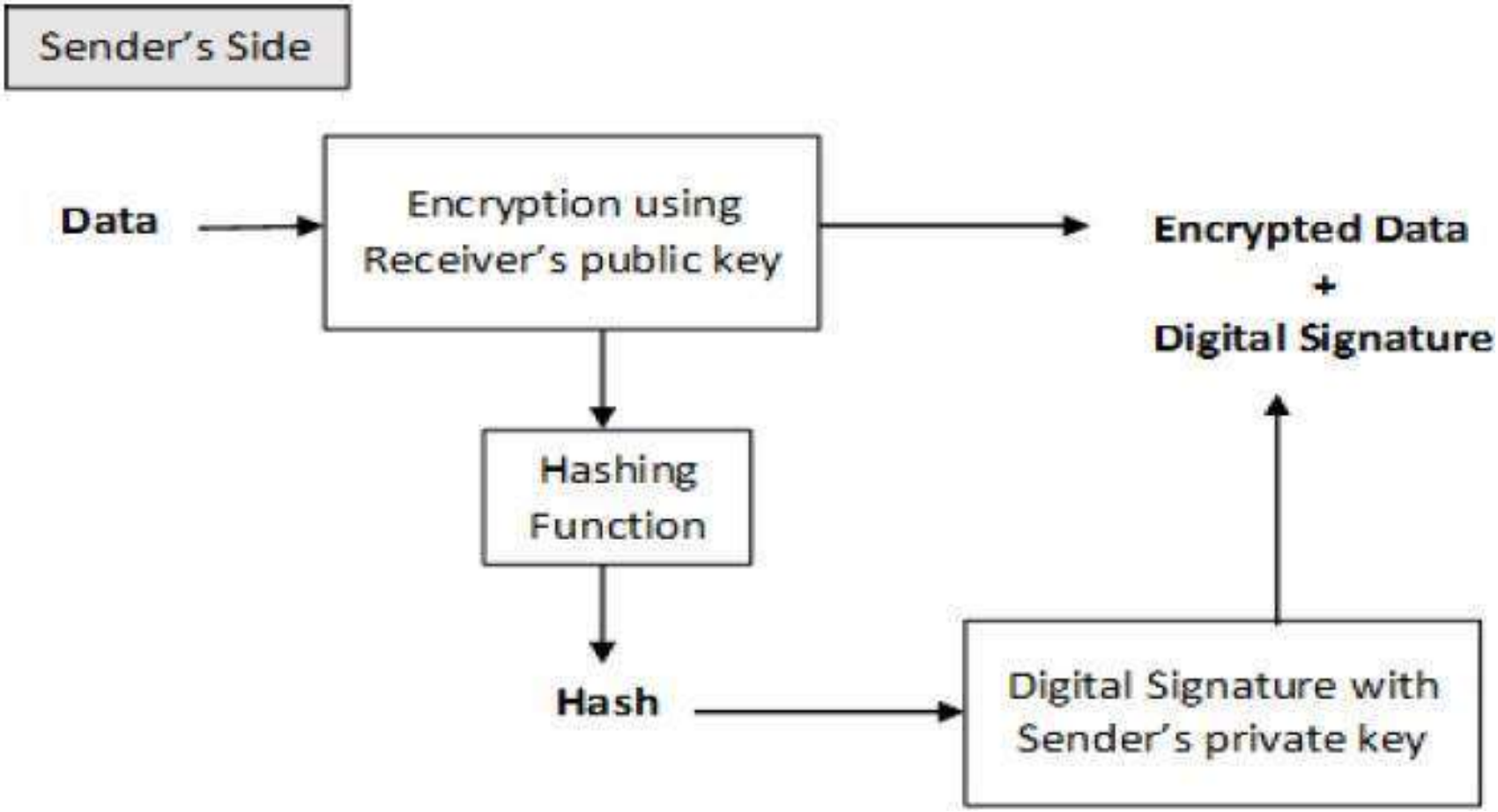
Digital signatures

- Cryptography is used to implement a mechanism known as a *digital signature.*
- *This is similar to conventional signature, verifying to a third* party that a message or a document is an unaltered copy of one produced by the signer.
- This can be achieved by encrypting the message called a *digest – using a key* that is known only to the signer.
- A digest is a fixed-length value computed by applying a *secure digest function.*
- The resulting encrypted digest acts as a signature that accompanies the message.
- The originator generates a signature with their private key, and the signature can be decrypted by any recipient using the corresponding public key.

Model of Digital Signature



Encryption


Sender's Side
Data
Encryption using
Receiver's public key
Encrypted Data
+
Digital Signature
Hashing
Function
Hash
Digital Signature with
Sender's private key

Scenario 4

Digital signatures with a secure digest function

- Alice wants to publish a document M in such a way that anyone can verify that it is from her.

1. Alice computes a fixed-length digest of the document Digest(M).
2. Alice encrypts the digest in her private key, appends it to M and makes the resulting signed document

   (M, {Digest(M)}KApriv) available to the intended users.
3. Bob obtains the signed document, extracts M and computes Digest(M).
4. Bob uses Alice's public key to decrypt {Digest(M)}Kapriv and compares it with his computed digest. If they match, Alice's signature is verified.

Certificates

- A digital certificate is a document containing a statement (usually short) signed by a principal.

Scenario 5. The use of certificates:

  - Bob is a bank.
  - Bob needs to authenticate his customers before he gives them access to their accounts.

- Alice wants the certificate from her bank stating her bank account number.
- **Alice's bank account certificate**

  1. *Certificate type: Account number*
  2. *Name: Alice*
  3. *Account: 6262626*
  4. *Certifying authority: Bob's Bank*
  5. *Signature: {Digest(field 2 + field 3)}KBpriv*).
- Alice could use this certificate when shopping to certify that she has an account with Bob's Bank.
- The certificate is signed using Bob's private key, *KBpriv.*
- *A vendor, Carol, can accept such a certificate for charging items to* Alice's account provided that she can validate the signature in field 5.

- Carol needs to have Bob's public key.
- Assume that Fred represents the Bankers Federation, one of whose roles is to certify the public keys of banks. Fred could issue a *public-key certificate for Bob.*
- **Public-key certificate for Bob's Bank**

  1. *Certificate type: Public key*
  2. *Name: Bob's Bank*
  3. *Public key: KBpub*
  4. *Certifying authority: Fred – The Bankers Federation*
  5. *Signature: {Digest(field 2 + field 3)}KFpriv*).

Access control

- Historically, the protection of resources in distributed systems has been largely service-specific. Servers receive request messages of the form <*op, principal, resource*>.
- In object-oriented distributed systems there may be many types of object to which access control must be applied, and the decisions are often application-specific.

Protection domains

- A protection domain is an execution environment shared by a collection of processes: it contains a set of

  <resource, rights> pairs.

- *Lists the resources* that can be accessed by all processes executing within the domain and specifying the operations permitted on each resource.

Cryptographic algorithms

- The encryption transformation is defined with two parts, a *function E and a key K. The resulting encrypted message is written {M}K .*

  E(K, M) = {M} K

- Decryption is carried out using an inverse function *D, which also takes a key as a parameter. For secret-key encryption, the key used for* decryption is the same as that used for encryption:

  D(K, E(K, M)) = M

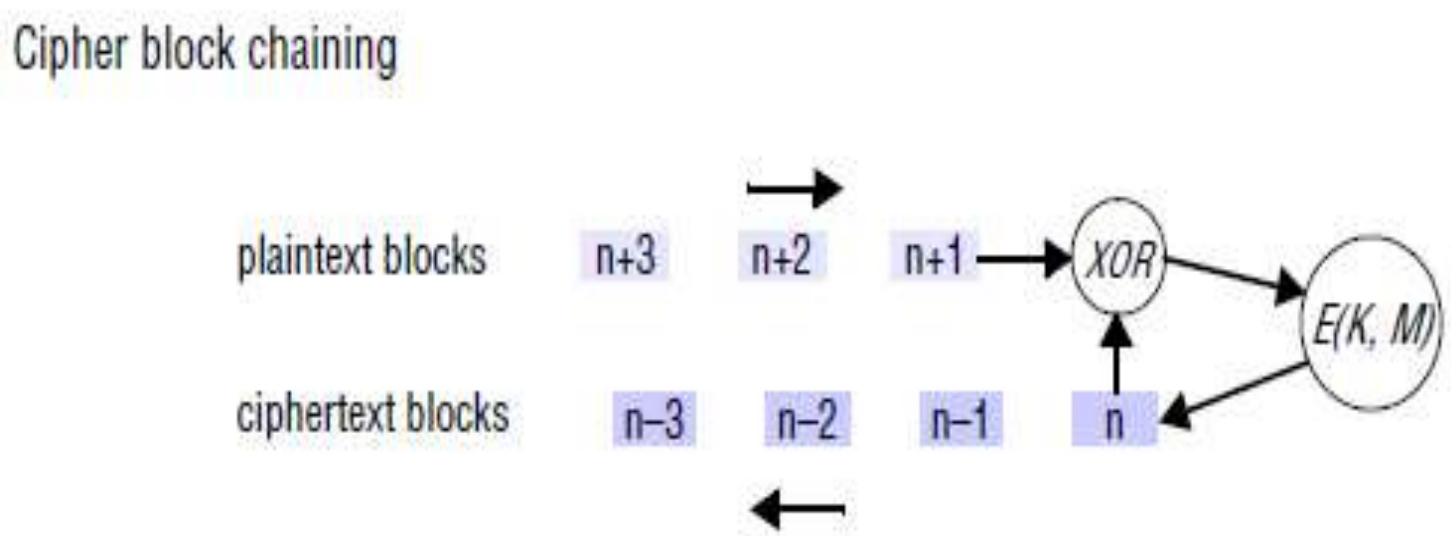
- Symmetric (secret key)
  - E(K, M) = {M}K
  - D(K, E(K, M)) = M
  - Same key for E and D
  - M must be hard (infeasible) to compute if K is not known.
  - Usual form of attack is brute-force: try all possible key values for a known pair M,{M}K. Resisted by making K sufficiently large ~ 128 bits.

- Asymmetric (public key)
  - Separate encryption and decryption keys: K_e, K_d
  - $D(K_d. E(K_e, M)) = M$
  - depends on the use of a *trap-door function to make the keys.*
  - A trap-door function is a one-way function with a secret exit – it is easy to compute in one direction but infeasible to compute the inverse unless a secret is known.

Block Ciphers

- Most encryption algorithms operate on fixed-size blocks of data;
- 64 bits is a popular size for the blocks.
- A message is subdivided into blocks, the last block is padded to the standard length if necessary and each block is encrypted independently.
- The first block is available for transmission as soon as it has been encrypted.

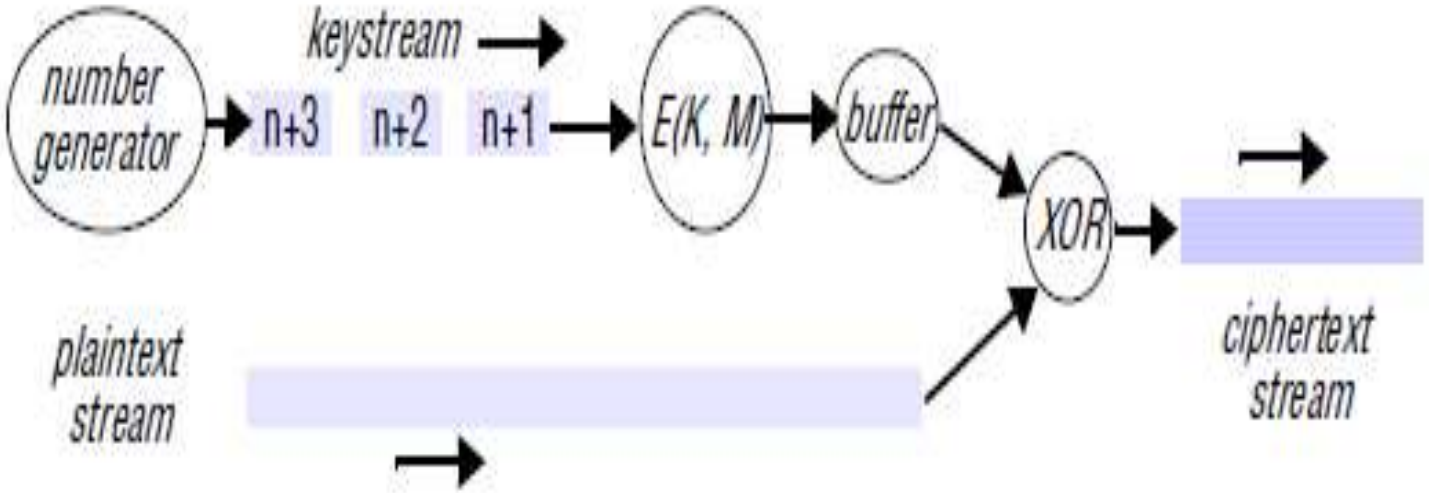
Cipher block chaining



Stream ciphers

- Stream ciphers are encryption algorithms that can perform encryption incrementally, converting plain text to cipher text one bit at a time.

Stream cipher



Symmetric key encryption algorithm

- **Symmetric-key algorithms** are a class of algorithms for cryptography that use trivially related, often identical, cryptographic keys for both decryption and encryption.
- **Types of symmetric-key algorithms**
  1. stream ciphers
  2. block ciphers

RC4 : An Example of Symmetric Algorithm

- RC4 designed in 1987 by RSA (Ron Rivest, Adi Shamir, and Leonard Adleman) .
- symmetric key encryption algorithm .
- Stream Cipher .

Symmetric encryption algorithms

- **TEA (Tiny Encryption Algorithm)**
- The TEA algorithm uses rounds of integer addition, XOR (the ^ *operator) and* bitwise logical shifts (<< *and* >>*) to achieve diffusion and confusion of the bit patterns* in the plaintext.
- The plaintext is a 64-bit block represented as two 32-bit integers in the vector *text[]. The key is 128 bits long, represented as four 32-bit integers.*
- The use of XOR and shifted portions of the text provides confusion, and the shifting and swapping of the two portions of the text provides diffusion.

DES

- The encryption function maps a 64-bit plaintext input into a 64-bit encrypted output using a 56-bit key.
- The algorithm has 16 key-dependent stages known as *rounds, in which the data to be encrypted is bit-rotated* by a number of bits determined by the key.
- *Triple-DES (or 3DES) is frequently used.*
- This involves applying DES three times with two keys, *K1 and K2:*
- *E3DES(K1, K2, M)= EDES (K1, EDES (K2, EDES(k1,M)))*

IDEA

- It uses a 128-bit key to encrypt 64-bit blocks.
- Its algorithm is based on the algebra of groups and has eight rounds of XOR, addition modulo 2^16 and multiplication.
- For both DES and IDEA, the same function is used for encryption and decryption: a useful property for algorithms that are to be implemented in hardware.

Public-key (asymmetric) algorithms

- The keys *Ke* and *Kd are a pair of very large numbers, and the encryption function performs an* operation, such as exponentiation on *M, using one of them. Decryption is a similar* function using the other key.
- RSA

RSA ALGORITHM (Rivest, Shamir and adleman)- example: GENERATING PUBLIC KEY

1. Choose two large prime numbers, P and Q (each greater than 10^{100}), and form
   $N = P \times Q$
   $Z = (P-1) \times (Q-1)$
2. For d choose any number that is relatively prime with Z (that is, such that d has no common factors with Z).

We illustrate the computations involved using small integer values for P and Q:

$P = 13, Q = 17 \rightarrow N = 221, Z = 192$

$d = 5$

3. To find e solve the equation:
   $e \times d = 1 \ mod \ Z$

That is, $e \times d$ is the smallest element divisible by d in the series $Z+1, 2Z+1, 3Z+1, \ldots$.

$e \times d = 1 \ mod \ 192 \ = 1, 193, 385, \ldots$

385 is divisible by d

$e = 385/5 = 77$

RSA

The function for encrypting a single block of plaintext M is:

$$E'(e,N,M) = M^e \bmod N$$

for a message M, the ciphertext is $M^{77} \bmod 221$

The function for decrypting a block of encrypted text c to produce the original plaintext block is:

$$D'(d,N,c) = c^d \bmod N$$

Encryption & Decryption

- Suppose if M=2 Encrypt & Decrypt using RSA
- $E=M^e$ ModN
- $E=2^{77}$Mod 221
- =(151115727451828646838272)Mod221=32
- $D=E^d$ModN
- $D=(32)^5$Mod221=(33554432)Mod221=2(M)

Encryption & Decryption

- If M=3 Encrypt & Decrypt using RSA
- $E=M^e$ ModN
- $E=3^{77}$Mod 221
- =(5.474401089420219382077155933569 8e+36)Mod221=165
- $D=E^d$ModN
- $D=(165)^5$Mod221=(122298103125)Mod221=3