System and Network Security

Dr. Ashok Kumar Das

Center for Security, Theory and Algorithmic Research
International Institute of Information Technology, Hyderabad

E-mail: *ashok.das@iiit.ac.in*
URL: http://www.iiit.ac.in/people/faculty/ashokkdas
Personal Home Page: http://sites.google.com/site/iitkgpakdas/

Protocol Hierarchies

- To reduce design complexity, most networks are organized as a stack of “layers” or “levels”, each one is built upon the one below it.
- The number of layers, the name of the layer, the contents of each layer, and the function of each layer differ from network to network.
- In a sense, each layer is a kind of virtual machine, offering certain services to the layer above it.

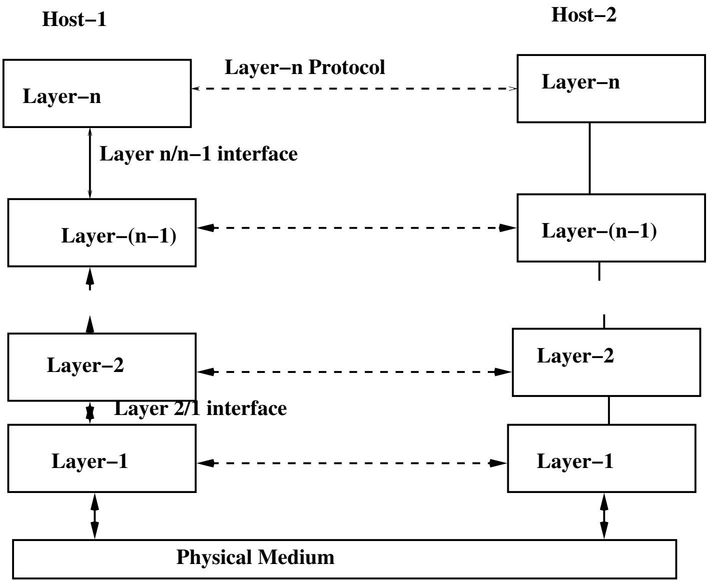
Why layering is needed?

- To provide well-defined interfaces between adjacent layers.
  - A change in one layer does not affect the other layer.
  - Interface must remain the same. [Interface defines which primitive operations and services the lower layer makes available to the upper layer.]
- Allows a structured development of network software.

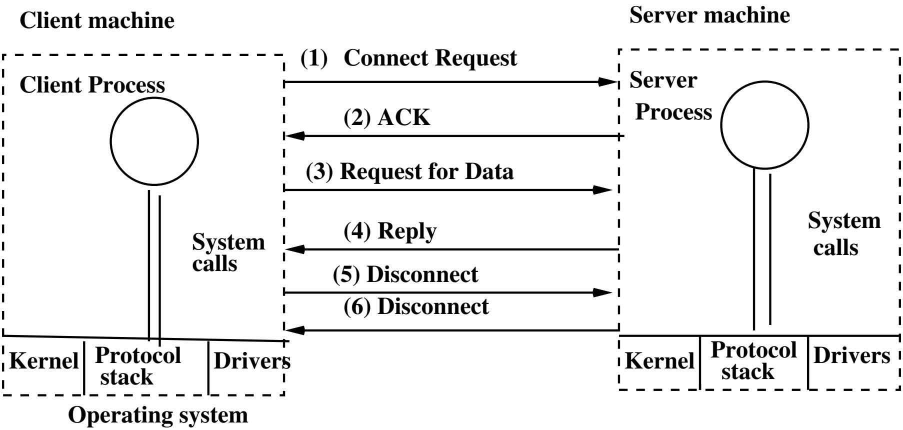
Protocol Hierarchies

- A set of layers and protocols is called a “network architecture”.
- A list of protocols used by a certain system, one protocol per layer, is called a “protocol stack”.

Layered Network Architecture



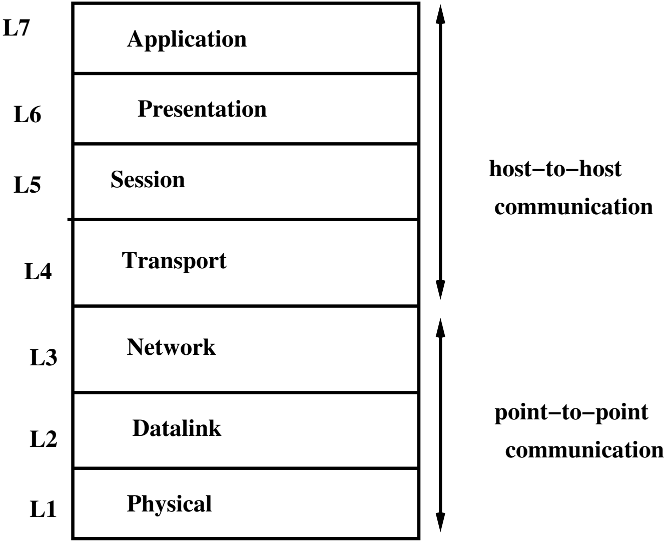
A simple client-server interaction on a connection-oriented network



The OSI Reference Model

- In 1978, International Standards Organization (OSI) proposed a 7-layer reference model for network services and protocols, known as the OSI model.
- The main objective of the OSI model as
  (1) Systematic approach to design.
  (2) Changes in one layer should not require changes in other layers.

The OSI Reference Model



Layer functions

Physical Layer:

- Transmits raw bit stream over a physical medium.
- The design issues have to do making sure that when one side sends a 1 bit, it is received by the other side as a 1 bit, not a 0 bit.
- The design issues largely deal with mechanical, electrical, and timing interfaces, and the physical transmission medium, which lies below the physical layer.
- Network components: Repeater, Multiplexer, Hubs, Amplifier.

Layer functions

Datalink Layer:

- Reliable transfer of frames (data) over a point-to-point link.
- Responsible for flow control, error control (error detection/correction), congestion control.
- Network components: Bridge, Switch, NIC, Advanced Cable Tester.

Layer functions

Network Layer:

- Establishing, maintaining and terminating connections.
- Routes packets (messages) through point-to-point link.
- Network components: Router, Frame Relay Device, ATM Switch.

Layer functions

Transport Layer:

- End-to-end reliable data transfer, with error recovery and flow control.
- Network components: Gateway.

Layer functions

Session Layer:

- Allows users on different machines (hosts) to establish sessions between them.
- Session offer various services, including
  - Dialog Control: Keeping track of whose turn it is to transmit.
  - Token Management: Preventing two parties from attempting the same critical operation at the same time.
  - Synchronization: Checkpointing long transmissions to allow them to continue from where they were after a crash.
- Network components: Gateway.

Layer functions

Presentation Layer:

- Translates data from application to network format, and vice-versa.
- All different formats from all sources are made into a common uniform format that the rest of the OSI model can understand.
- Network components: Gateway.

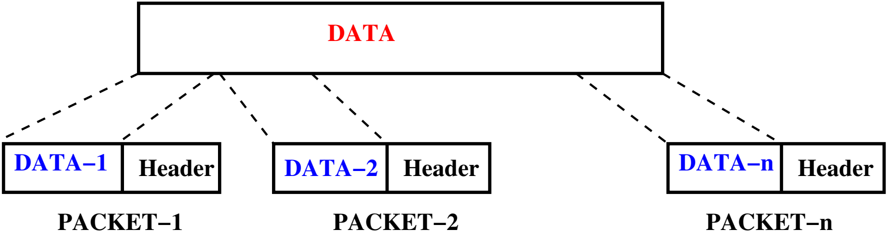
Layer functions

Application Layer:

- Interface point for user applications.
- Network components: Gateway.

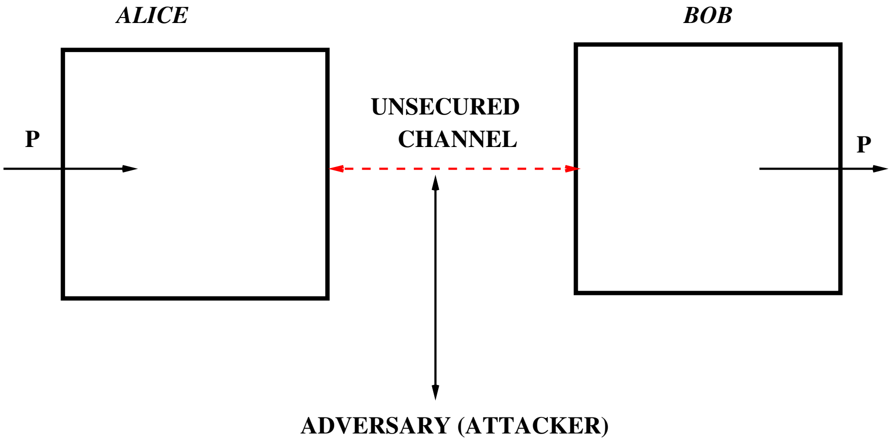
Layer functions

Data handled in a particular layer:



- This is a classical Alice (user A) and Bob (user B) problem: “Alice wants to send Bob a secure message”.
- Question: What does she do ?
- Answer: She encrypts the message.

The classical Alice (user A) and Bob (user B) problem:



Chapter: Encrypting Communications Chennels

- In theory, the encryption can take place at any layer in the OSI communication model.
- In practice, it takes place either at the lowest layers (one and two) or at higher layers.
- If it takes place at the lowest layers, it is called “link-by-link encryption” (LLE).
- In LLE, everything going through a particular data link is encrypted.

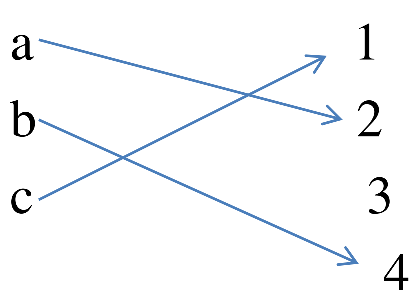
- If the encryption takes place at the higher layers, it is called "end-to-end encryption" (EEE).
- In EEE, the data are encrypted selectively and stay encrypted until they are decrypted by the intended final recipient.
- Each approach has its own benifits and drawbacks.

Thank You!!!

Background on functions

- A function is defined by two sets X and Y and a rule f which assigns to each element of X to an element of Y. It is denoted by $f : X \rightarrow Y$
- X : ***domain*** of the function f
- Y: ***co-domain (range)*** of the function f
- The ***image*** y in Y of an element x in X is denoted by y = f(x)
- For a function f from set X to set Y, if y in Y then a ***pre-image*** of y is an element x in X for which f(x) = y
- The set of all elements in Y which have at least one pre-image is called the image of f, denoted by ***Im(f)***
- ***Example:***

Consider X = {a, b, c} and Y = {1, 2, 3, 4}, f: X -> Y is defined as f(a) = 2, f(b) = 4, f(c) = 1. Here Im(f) = { 1, 2, 4}



Background on functions

Example:

Take X = { 1, 2, 3, …, 10} and f: X -> X is defined as f(x) = x*x mod 11. We then have the following table:

x	1	2	3	4	5	6	7	8	9	10
f(x)	1	4	9	5	3	3	3	5	9	1

The image of f is Im (f) = { 1, 3, 4, 5, 9 }

One-to-one (1-1) Function:

A **function (transformation or mapping) f: X -> Y** is 1-1 (injective) if each element in Y is the image of at most one element in X.

In other words,

$$x_{1,} x_2 \in X (x_1 \neq x_2)$$

$$\Rightarrow f(x_1) \neq f(x_2)$$

i.e., $f(x1) = f(x2)$

$$\Rightarrow x1 = x2$$

- **Onto (Surjective) function**

A function f: X -> Y is onto if each element in Y is the image of at least one element in X. Equivalently, f if onto, if ***Im(f) = Y***

- **Bijective function (Bijection)**

If a function f: X-> Y is 1-1 and onto, then it is called a bijection.

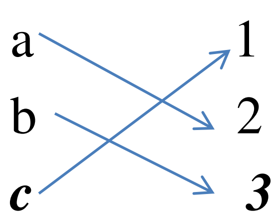
Theorem: If f: X->Y is 1-1, then f: X->Im(f) is a bijection.

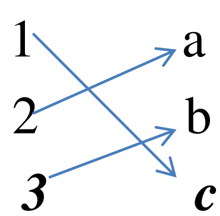
Theorem: If f: X->Y is 1-1, and X and Y are finite sets of same sizes (cardinalities) then f is a bijection.

- Inverse Function: If a function f: X->Y is bijective, then its inverse exists.

If $g = f^{-1}$, then g: Y->X.

Example:





f ***g***

- **One-Way Function**

A function $f: X \to Y$ is called a one-way function if $f(x)$ is ***"easy"*** to compute for all x in X but for "essentially all" elements y in Im(f) it is ***"computationally infeasible"*** to find any x in X such that $f(x) = y$.

In other words, a one-way function is easily computed, but the calculation of its inverse is infeasible.

- **Trap-door one-way function**

A trap-door one-way function is a function $f: X \to Y$ with the additional property that given some extra information (called the trap-door information) it becomes feasible to find for any given y in Im(f), an x in X such that $f(x) = y$.

In other words, we can say that a trap-door function is a function that is easily computed; the calculation of its inverse is infeasible unless privileged information is known

- **Permutation**

Let S be a finite set of elements. A permutation p on S is a bijection from S to itself (S). In other words, p: S->S is a bjection.

If $S = \{a_1, a_2, \ldots, a_n\}$ then p is represented as

$$p = \begin{pmatrix} a_1 & a_2 & \cdots & a_n \\ p(a_1) & p(a_2) & \cdots & p(a_n) \end{pmatrix}$$

- **Inverse Permutation**

$$p^{-1} = \begin{pmatrix} p(a_1) & p(a_2) & \cdots & p(a_n) \\ a_1 & a_2 & \cdots & a_n \end{pmatrix}$$

Example: S = {1, 2, 3, 4}

$$p = \begin{pmatrix} 1 & 2 & 3 & 4 \\ 3 & 4 & 2 & 1 \end{pmatrix}$$

$$p^{-1} = \begin{pmatrix} 3 & 4 & 2 & 1 \\ 1 & 2 & 3 & 4 \end{pmatrix}$$

$$= \begin{pmatrix} 1 & 2 & 3 & 4 \\ 4 & 3 & 1 & 2 \end{pmatrix}$$

Basic terminology and concepts

- **Encryption domains and co-domains**

- A denotes a finite set called the alphabet of definition.
  For example, A = {0, 1}

- M denotes a set called the ***Message Space***. M consists of strings of symbols from an alphabet set of definition, A. An element of M is called a ***plaintext message or simply a plaintext.***

- C denotes a set called the ***ciphertext space***. C consists of strings of symbols from an alphabet of definition, which may differ from the alphabet of definition. An element of C is called a ***ciphertext***.

Encryption and Decryption Transformations

- K denotes a set called the key space. An element of K is called a key.
- Each element e in K uniquely determines a bijection function from M to C, denoted and defined by $E_e : M \rightarrow C$

  $$m \rightarrow c$$

  is called an encryption function or encryption transformation.

- For each d in K, a decryption function or decryption transformation is defined as

  $$D_d : C \rightarrow M$$
  $$c \rightarrow m$$

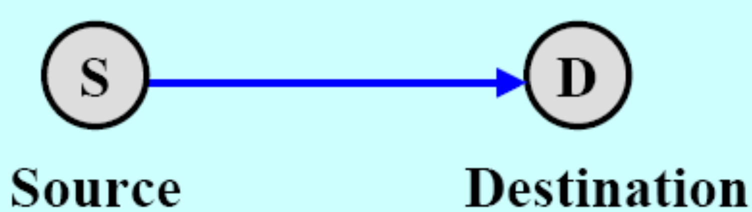
- An encryption scheme consists of a set $\{E_e : e \in K\}$ of encryption transformations and a corresponding set $\{D_d : d \in K\}$ of decryption transformations with the property that for each e in K there is a unique d in K such that

Requirements for constructing an encryption scheme

- A message space , M
- A ciphertext space, C
- A key space, K
- A set of encryption transformations $\{E_e : e \in K\}$
- A corresponding set of decryption transformations $\{D_d : d \in K\}$

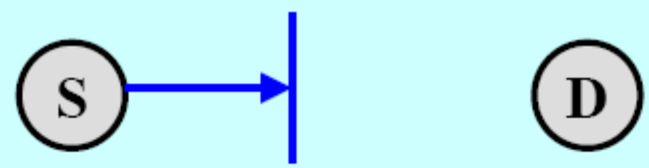
Security Attacks

- Any action that compromises the security of information.
- Four types of attack:
  1. Interruption
  2. Interception
  3. Modification
  4. Fabrication
- Basic model:

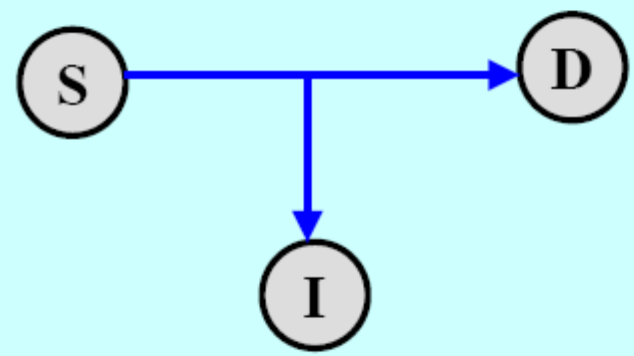


Security Attacks (Continued…)

- **Interruption**:
  - Attack on availability

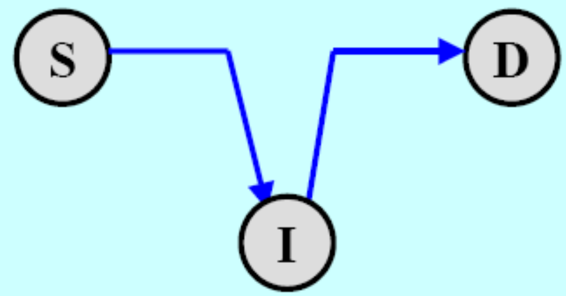


- **Interception**:
  - Attack on confidentiality

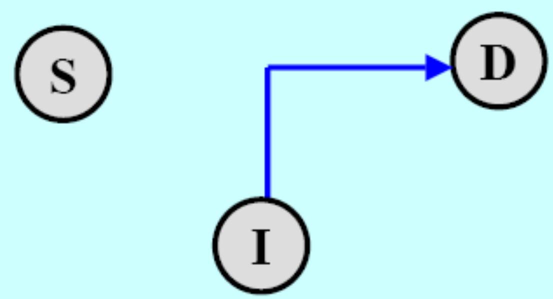


Security Attacks (Continued...)

- **Modification**:
  - Attack on integrity



- **Fabrication**:
  - Attack on authenticity



Passive and Active Attacks

- **Passive attacks**
  - Obtain information that is being transmitted (eavesdropping).
  - Two types:
    - <u>Release of message contents</u>:- It may be desirable to prevent the opponent from learning the contents of the transmission.
    - <u>Traffic analysis</u>:- The opponent can determine the location and identity of communicating hosts, and observe the frequency and length of messages being exchanged.
  - Very difficult to detect.

Passive and Active Attacks (Continued…)

- **Active attacks**
  - Involve some modification of the data stream or the creation of a false stream.
  - Four categories:
    - <u>Masquerade</u>:- One entity pretends to be a different entity.
    - <u>Replay</u>:- Passive capture of a data unit and its subsequent retransmission to produce an unauthorized effect.
    - <u>Modification</u>:- Some portion of a legitimate message is altered.
    - <u>Denial of service</u>:- Prevents the normal use of communication facilities.

Types of Attacks on Encrypted Messages

Type of Attack	Known to Cryptanalyst
Ciphertext only (most difficult)	•Encryption algorithm •Ciphertext to be decoded
Known Plaintext (easier than above)	•Encryption algorithm •Ciphertext to be decoded •One or more plaintext-ciphertext pairs formed with the secret key
Chosen plaintext (easier than above two)	•Encryption algorithm •Ciphertext to be decoded •Plaintext message chosen by cryptanalyst, together with its corresponding ciphertext generated with the secret key (*)
Chosen ciphertext (easier than abobe all)	•Encryption algorithm •Ciphertext to be decoded •Purported ciphertext chosen by cryptanalyst, together with its corresponding decrypted plaintext generated with the secret key (**)
Chosen text (easier than above all)	•Encryption algorithm •Ciphertext to be decoded • * •**

System and Network Security

Dr. Ashok Kumar Das

IEEE Senior Member
Associate Proofessor
Center for Security, Theory and Algorithmic Research
International Institute of Information Technology, Hyderabad

E-mail: *ashok.das@iiit.ac.in*
URL: http://www.iiit.ac.in/people/faculty/ashokkdas
https://sites.google.com/site/iitkgpakdas/

Symmetric-Key Encryption

Principle of Shannon (1945)

- **Diffusion:** The mechanism of diffusion seeks to make the statistical relationship between the plaintext and ciphertext as complex as possible in order to thwart attempts to deduce the key.
  - Diffusion can be achieved by repeatedly performing some permutation on the data followed by applying a function to that permutation.
- **Confusion:** It seeks to make the statistical relationship between the ciphertext and the value of encrypted key as complex as possible in order to thwart attempts to deduce the key.
  - Confusion can be achieved by the use of a complex substitution algorithm.

The Fiestel Cipher

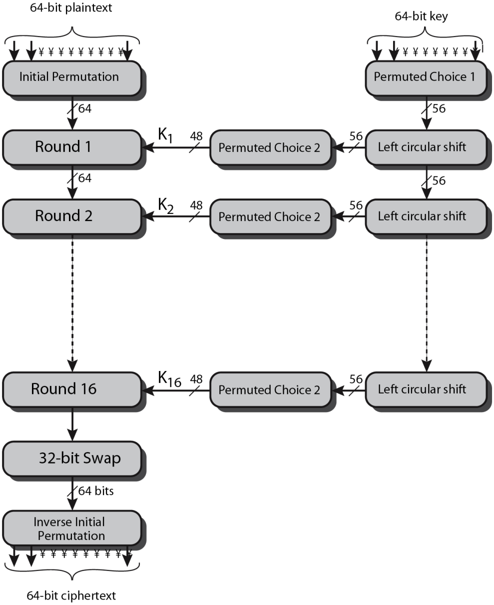
- All modern day block ciphers are based on Fiestel cipher structure.
- Fiestel structure is based on the principle of Shannon (1945): Diffusion and Confusion
- Fiestel structure is useful to construct a SPN (Substitution-Permutation Network) cipher

Symmetric-Key Encryption

Data Encryption Standard (DES)

- The most widely used encryption is based on the Data Encryption Standard (DES) adopted in 1977 by the National Institute of Standards and Technology (NIST), USA.
- For DES, data are encrypted in 64-bit blocks using a 56-bit key.
- The encryption algorithm transforms 64-bit input in a series of steps into a 64-bit output.
- The same steps, with the same key, are used to reverse the encryption (decryption).
- Mathematically, $DES : \{0,1\}^{64} \times \{0,1\}^{56} \longrightarrow \{0,1\}^{64}$ such that the ciphertext be $C = DES_K(P)$, where $K \in \{0,1\}^{56}$ is the 56-bit key, $P \in \{0,1\}^{64}$ is the plaintext message (block) and $C \in \{0,1\}^{64}$ is the ciphertext block.

Overview of Data Encryption Standard (DES)

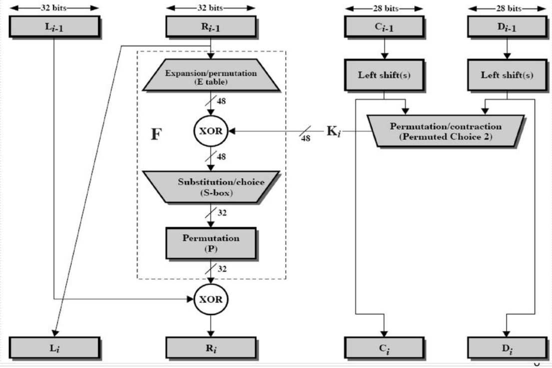


Data Encryption Standard (DES)

- K: given 56 bit key
- K is converted to 64 bit key packed with 8 bit parity:
  parity 8 bits at positions 8, 16, 24, 32, 40, 48, 56, and 64.
- $K_1, K_2, \cdots, K_{16}$: 16 round keys
- **Schedule of left circular shifts:**

  if (**round number = 1, 2, 9, 16**), then bits_rotated = 1
  else
  bits_rotated = 2

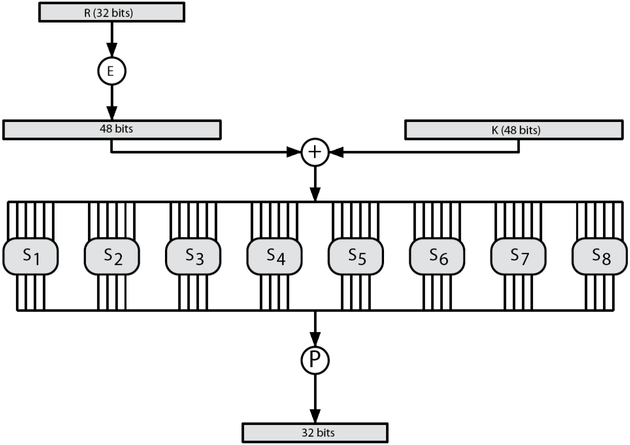
Single Round of DES



$L_i = R_{i-1}$; $R_i = L_{i-1} \oplus F(R_{i-1}, K_i)$, $\forall i = 1, 2, \cdots 16$

E: Expansion/permutation; S-Box (S_i): Substitution/choice; P: permutation; L_i : left half (32 bits) of message; R_i : right half (32 bits) of message; C_i : left half (28 bits) of key; D_i : left half (28 bits) of key.

Calculation of function $F(R_i, K_i)$ in DES



$F(R_i, K_i) = P(S(E(R_i) \oplus K_i))$

E: Expansion/permutation; S: S-Box; L_i : left half (32 bits) of message; R_i : right half (32 bits) of message; K_i: i^{th} round key.

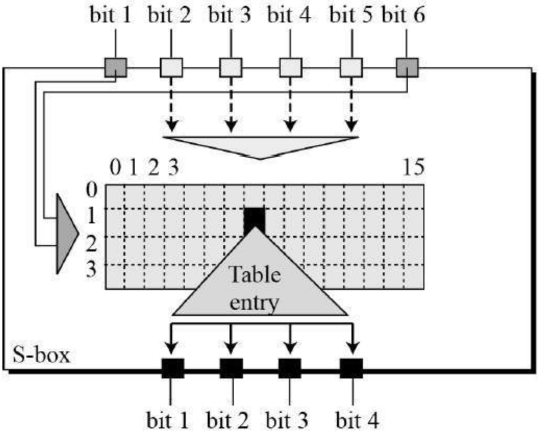
Initial Permutation (IP) and IP^{-1}

IP								IP^{-1}							
58	50	42	34	26	18	10	2	40	8	48	16	56	24	64	32
60	52	44	36	28	20	12	4	39	7	47	15	55	23	63	31
62	54	46	38	30	22	14	6	38	6	46	14	54	22	62	30
64	56	48	40	32	24	16	8	37	5	45	13	53	21	61	29
57	49	41	33	25	17	9	1	36	4	44	12	52	20	60	28
59	51	43	35	27	19	11	3	35	3	43	11	51	19	59	27
61	53	45	37	29	21	13	5	34	2	42	10	50	18	58	26
63	55	47	39	31	23	15	7	33	1	41	9	49	17	57	25

E: Expansion/permutation

32	1	2	3	4	5
4	5	6	7	8	9
8	9	10	11	12	13
12	13	14	15	16	17
16	17	18	19	20	21
20	21	22	23	24	28
24	25	26	27	28	29
28	29	30	31	32	1

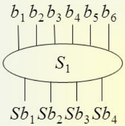
S-Box Rule



bit 1
bit 2
bit 3
bit 4
bit 5
bit 6
0 1 2 3
15
0
1
2
3
Table entry
S-box
bit 1
bit 2
bit 3
bit 4

S-Box (S_1) Example

S-box (substitution box)



Look-up a value from the table using
$b_1 b_6$: row
$b_2 b_3 b_4 b_5$: column

S_1-box table

$b_1 b_6$: row								Sb_1								
0	14	4	13	1	2	15	11	8	3	10	6	12	5	9	0	7
1	0	15	7	4	14	2	13	1	10	6	12	11	9	5	3	8
2	4	1	14	8	13	6	2	11	15	12	9	7	3	10	5	0
3	15	12	8	2	4	9	1	7	5	11	3	14	10	0	6	13
	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15

$b_2 b_3 b_4 b_5$: column

8

Example: Input (6 bits) = 1 1 1 0 0 1; row-index $= b_1 b_6 =$ **(1 1)**$_2 = 3$;
col-index $= b_2 b_3 b_4 b_5 =$ **(1 1 0 0)**$_2 = 12$;
output = S_1**[row-index][col-index]** $= 10 =$ **(1 0 1 0)**$_2$

DES S-Box Tables

Substitution Boxes S-Boxes

Box	Row	Column 0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
S_1																	
	0	14	4	13	1	2	15	11	8	3	10	6	12	5	9	0	7
	1	0	15	7	4	14	2	13	1	10	6	12	11	9	5	3	8
	2	4	1	14	8	13	6	2	11	15	12	9	7	3	10	5	0
	3	15	12	8	2	4	9	1	7	5	11	3	14	10	0	6	13
S_2																	
	0	15	1	8	14	6	11	3	4	9	7	2	13	12	0	5	10
	1	3	13	4	7	15	2	8	14	12	0	1	10	6	9	11	5
	2	0	14	7	11	10	4	13	1	5	8	12	6	9	3	2	15
	3	13	8	10	1	3	15	4	2	11	6	7	12	0	5	14	9
S_3																	
	0	10	0	9	14	6	3	15	5	1	13	12	7	11	4	2	8
	1	13	7	0	9	3	4	6	10	2	8	5	14	12	11	15	1
	2	13	6	4	9	8	15	3	0	11	1	2	12	5	10	14	7
	3	1	10	13	0	6	9	8	7	4	15	14	3	11	5	2	12
S_4																	
	0	7	13	14	3	0	6	9	10	1	2	8	5	11	12	4	15
	1	13	8	11	5	6	15	0	3	4	7	2	12	1	10	14	9
	2	10	6	9	0	12	11	7	13	15	1	3	14	5	2	8	4
	3	3	15	0	6	10	[illegible]	13	8	9	4	5	11	12	7	2	14
S_5																	
	0	2	12	4	1	7	10	11	6	8	5	3	15	13	0	14	9
	1	14	11	2	12	4	7	13	1	5	0	15	10	3	9	8	6
	2	4	2	1	11	10	13	7	8	15	9	12	5	6	3	0	14
	3	11	8	12	7	1	14	2	13	6	15	0	9	10	4	5	3
S_6																	
	0	12	1	10	15	9	2	6	8	0	13	3	4	14	7	5	11
	1	10	15	4	2	7	12	9	5	6	1	13	14	0	11	3	8
	2	9	14	15	5	2	8	12	3	7	0	4	10	1	13	11	6
	3	4	3	2	12	9	5	15	10	11	14	1	7	6	0	8	13
S_7																	
	0	4	11	2	14	15	0	8	13	3	12	9	7	5	10	6	1
	1	13	0	11	7	4	9	1	10	14	3	5	12	2	15	8	6
	2	1	4	11	13	12	3	7	14	10	15	6	8	0	5	9	2
	3	6	11	13	8	1	4	10	7	9	5	0	15	14	2	3	12
S_8																	
	0	13	2	8	4	6	15	11	1	10	9	3	14	5	0	12	7
	1	1	15	13	8	10	3	7	4	12	5	6	11	0	14	9	2
	2	7	11	4	1	9	12	14	2	0	6	10	13	15	3	5	8
	3	2	1	14	7	4	10	8	13	15	12	9	0	3	5	6	11

Data Encryption Standard (DES)

Theorem

Let $DES_{K_1K_2\cdots K_{16}}$ denote the DES encryption function, where $K_1, K_2, \ldots, K_{16}$ be the 16 round keys of a given 56-bit input key K. Then, for all plaintext messages $x \in \{0,1\}^{64}$, $DES_{K_{16}K_{15}\cdots K_1}(DES_{K_1K_2\cdots K_{16}}(x)) = x$, that is, $DES_{K_{16}K_{15}\cdots K_1}$ becomes the DES decryption function.

Symmetric-Key Encryption

Data Encryption Standard (DES)

Theorem

Let DES : $\{0,1\}^{64} \times \{0,1\}^{56} \rightarrow \{0,1\}^{64}$ *be the DES function. Assume that* $\bar{x}$ *represents the bitwise complement of a bit string* x. *Then,* $DES(\bar{k}, \bar{x}) = \overline{DES(k, x)}$, *for every plaintext* $x \in \{0,1\}^{64}$ *and key* $k \in \{0,1\}^{56}$.

Theorem

Using this complementing property of DES, the brute-force attack to break the DES algorithm reduces the complexity from 2^{56} *to* 2^{55}.

Diffusion and Confusion Properties of DES

- In a binary block cipher, such as the DES, **diffusion** is accomplished by using permutations on data, and then applying a function to the permutation to produce ciphertext.
- In DES, **confusion** is accomplished by making the use of substitution operations (S-Boxes).

Avalanche Effect on DES

- A small change in the plaintext (or key) should create a significant change in the ciphertext.
- DES has been proved to be strong with regard to this property.
- **An Example:**
  - Set 1: **key: 2333 4519 ABCD 9513** (64-bits after 8-bit parity padding, 16 digits in hexadecimal)
    plaintext: 0000 0000 0000 0000
    ciphertext: C871 779E 2860 D09E
  - Set 2: **same key: 2333 4519 ABCD 9513**
    plaintext: 0000 0000 0000 0001 (single bit change)
    ciphertext: 10F6 2D55 327E 840A

Limitations of DES

Weak Keys

- In DES encryption/decryption, the initial key is of 56 bits. So, the total number of keys in the key space is 2^{56}.
- Four out of these 2^{56} possible keys (0000000 0000000; 0000000 FFFFFFF; FFFFFFF 0000000; FFFFFFF FFFFFFF) are called weak keys.
- A **weak key** is that one, after parity drop operation, consists of either of all 0s, all 1s, or half 0s and half 1s.
- In addition, there are 12 semi-weak keys and 48 possible weak keys, a total of such keys is (4 + 12 + 48) = 64.
- Probability of randomly selecting a weak, a semi-weak, or a possible weak key turns out to be $\frac{64}{2^{56}} = \frac{2^6}{2^{56}} = 2^{-50} \approx 8.8 \times 10^{-16}$, almost impossible.

Symmetric-Key Encryption

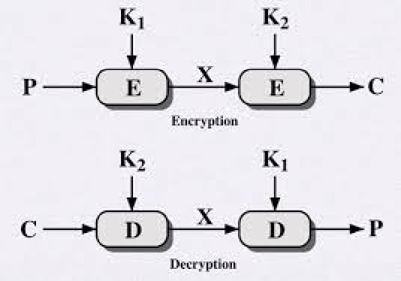
Data Encryption Standard (DES)

- DES finally and definitely proved insecure in July 1998, when the Electronics Frontier Foundation (EFF) announced that it had broken a DES encryption using a special-purpose “DES cracker” machine that was built for less than 250,000 USD.
- The attack took less than three days.

Various modes of operation

Double DES (2DES)

- It uses two 56-bit keys K_1 and K_2, and 64-bit plaintext block.
- It produces 64-bit ciphertext block.
- Known-plaintext attack (meet-in-the-middle attack) is possible against 2DES to derive two keys K_1 and K_2, which has a key size of 112 bits and with an effort on the order of 2^{56}.



Meet-in-the-middle attack in 2DES

- It is based on the observation that, if we have $C = E_{K_2}[E_{K_1}(P)]$, then $X = E_{K_1}(P) = D_{K_2}(C)$.
- Given a known pair, (P, C), the attack proceeds as follows.
  - First, encrypt P for all 2^{56} possible values of K_1 (in offline mode).
  - Store these results in a table and then sort the table by the values of X (in offline mode).
  - Next, decrypt C using all 2^{56} possible values of K_2 (in online mode).
  - As each decryption is produced, check the result against the table for a match.
  - If a match occurs, then test the two resulting keys against a new known plaintextciphertext pair.
  - If the two keys produce the correct ciphertext, accept them as the correct keys.

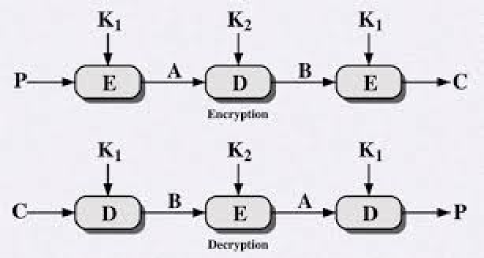
Meet-in-the-middle attack in 2DES

- For any given plaintext P , there are 2^{64} possible ciphertext values that could be produced by double DES.
- Double DES uses, in effect, a 112-bit key, so that there are 2^{112} possible keys. Therefore, on average, for a given plaintext P, the number of different 112-bit keys that will produce a given ciphertext C is $\frac{2^{112}}{2^{64}} = 2^{48}$.
- Thus, the foregoing procedure will produce about 2^{48} false alarms on the first (P, C) pair.
- A similar argument indicates that with an additional 64 bits of known plaintext and ciphertext, the false alarm rate is reduced to $\frac{2^{48}}{2^{64}} = 2^{-16}$.
- If the meet-in-the-middle attack is performed on two blocks of known plaintextciphertext, the probability that the correct keys are determined is $1 - 2^{-16}$.
- The result is that a known plaintext attack will succeed against double DES, which has a key size of 112 bits, with an effort on the order of 2^{56}, which is not much more than the 2^{55} required for single DES.

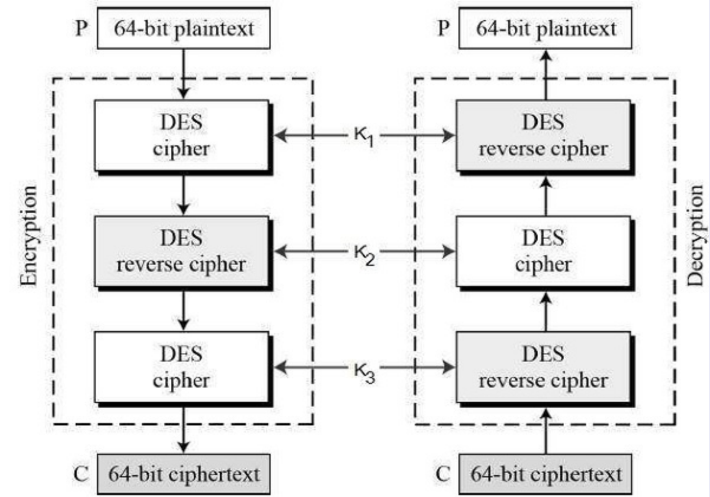
Symmetric-Key Encryption

Triple DES with Two Keys (3DES with Two Keys)

- It uses two 56-bit keys K_1 and K_2, and 64-bit plaintext block.
- It produces 64-bit ciphertext block.
- It is also vulnerable to known-plaintext attack (meet-in-the-middle attack) to derive two keys K_1 and K_2.
- The expected running time of this attack is on the order of $2^{120-\log_2 n}$, where n is the number of plaintext-ciphertext pairs.



Triple DES with Three Keys (3DES with Three Keys)



Symmetric-Key Encryption

Alternatives to Data Encryption Standard (DES)

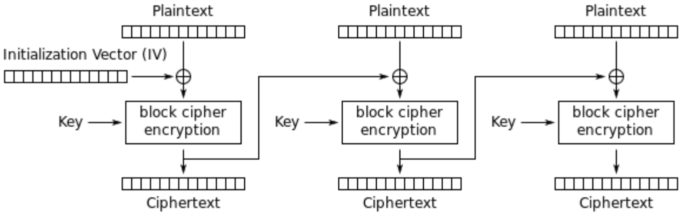
- **Triple DES with Three Keys (3DES with Three Keys)**
  - It uses three 56-bit keys K_1, K_2 and K_3, and 64-bit plaintext block.
  - It produces 64-bit ciphertext block.
  - No practical attack is found on this cipher so far. It is secure.
  - Application: It is used in all Internet-based applications such as PGP (Pretty Good Privacy) and S/MIME (Secure/Multipurpose Internet Main Extension) protocols.
- **AES (Advanced Encryption Standard)**
  - AES takes 128-bit key and 128-bit plaintext blacks as input.
  - AES produces 128-bit cipertext blocks.
  - AES is very efficient.
  - AES is secure against all possible attacks.

Symmetric-Key Encryption

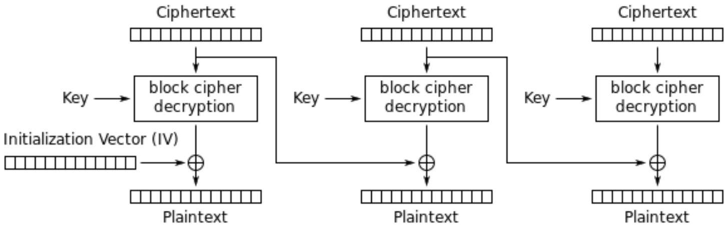
Various modes of operation of Data Encryption Standard (DES)

- Electronic Codebook Mode (ECB)
- Cipher Block Chaining Mode (CBC)
- Cipher Feedback Mode (CFB)
- Output Feedback Mode (OFB)
- Counter Mode (CTR)

Various modes of operation



Cipher Block Chaining (CBC) mode encryption



Cipher Block Chaining (CBC) mode decryption

Thank you

Public Key Cryptography

Private-Key Cryptography

- traditional **private/secret/single key** cryptography uses **one** key
- shared by both sender and receiver
- if this key is disclosed communications are compromised
- also is **symmetric**, parties are equal
- hence does not protect sender from receiver forging a message & claiming is sent by sender

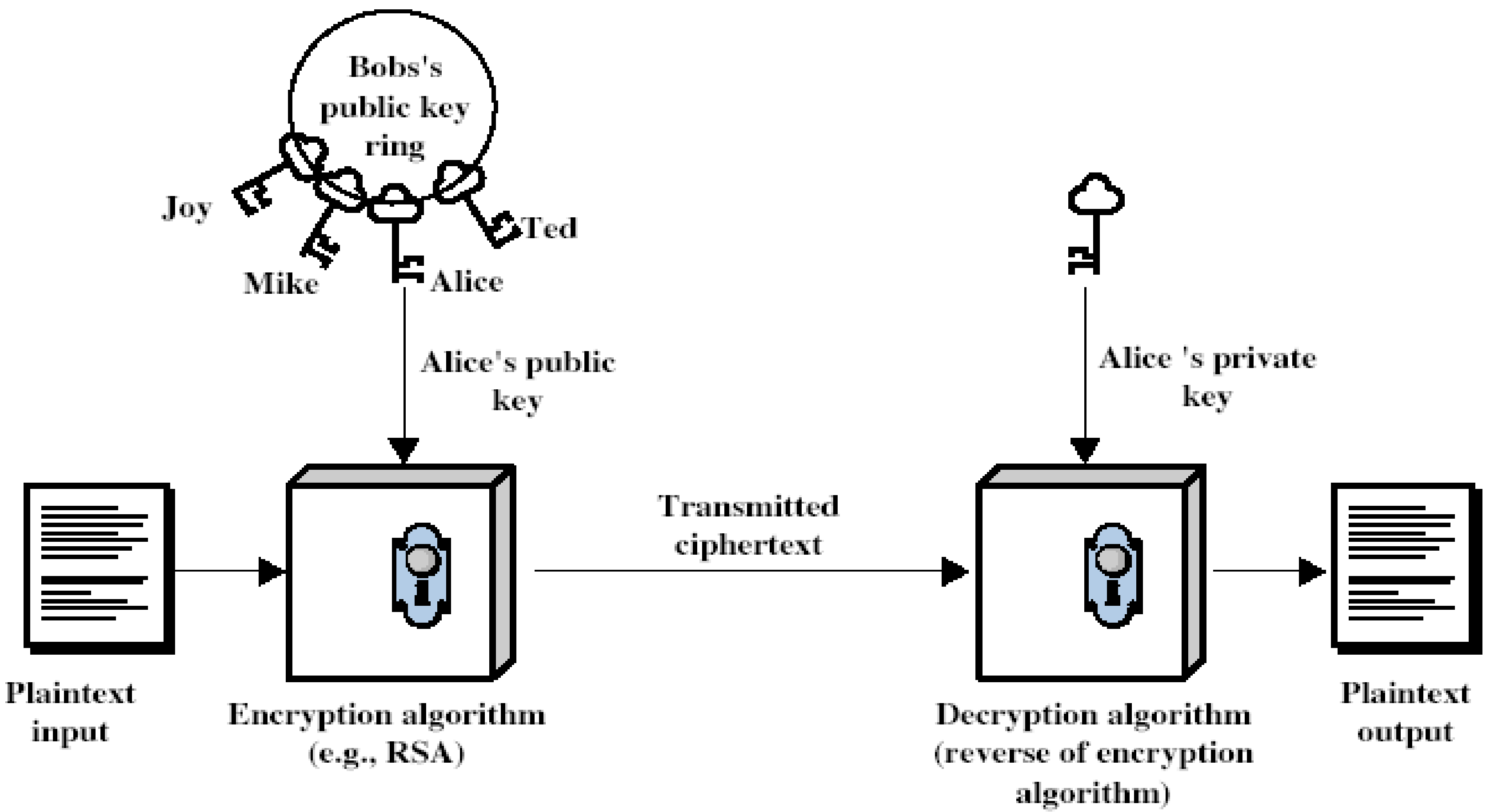
Public-Key Cryptography

- probably most significant advance in the 3000 year history of cryptography
- uses **two** keys – a public & a private key
- **asymmetric** since parties are **not** equal
- uses clever application of number theoretic concepts to function
- complements **rather than** replaces private key crypto

Public-Key Cryptography

- **public-key/two-key/asymmetric** cryptography involves the use of **two** keys:
  - a **public-key**, which may be known by anybody, and can be used to **encrypt messages**, and **verify signatures**
  - a **private-key**, known only to the recipient, used to **decrypt messages**, and **sign** (create) **signatures**
- is **asymmetric** because
  - those who encrypt messages or verify signatures **cannot** decrypt messages or create signatures

Public-Key Cryptography



Why Public Key Cryptography?

- developed to address two key issues:
  - **key distribution** – how to have secure communications in general without having to trust a KDC with your key
  - **digital signatures** – how to verify a message comes intact from the claimed sender
- public invention due to Whitfield Diffie & Martin Hellman at Stanford Uni in 1976
  - known earlier in classified community

Public-Key Characteristics

- Public-Key algorithms rely on two keys with the characteristics that it is:
  - computationally infeasible to find decryption key knowing only algorithm & encryption key
  - computationally easy to en/decrypt messages when the relevant (en/decrypt) key is known
  - either of the two related keys can be used for encryption, with the other used for decryption (in some schemes)

Public-Key Cryptosystems

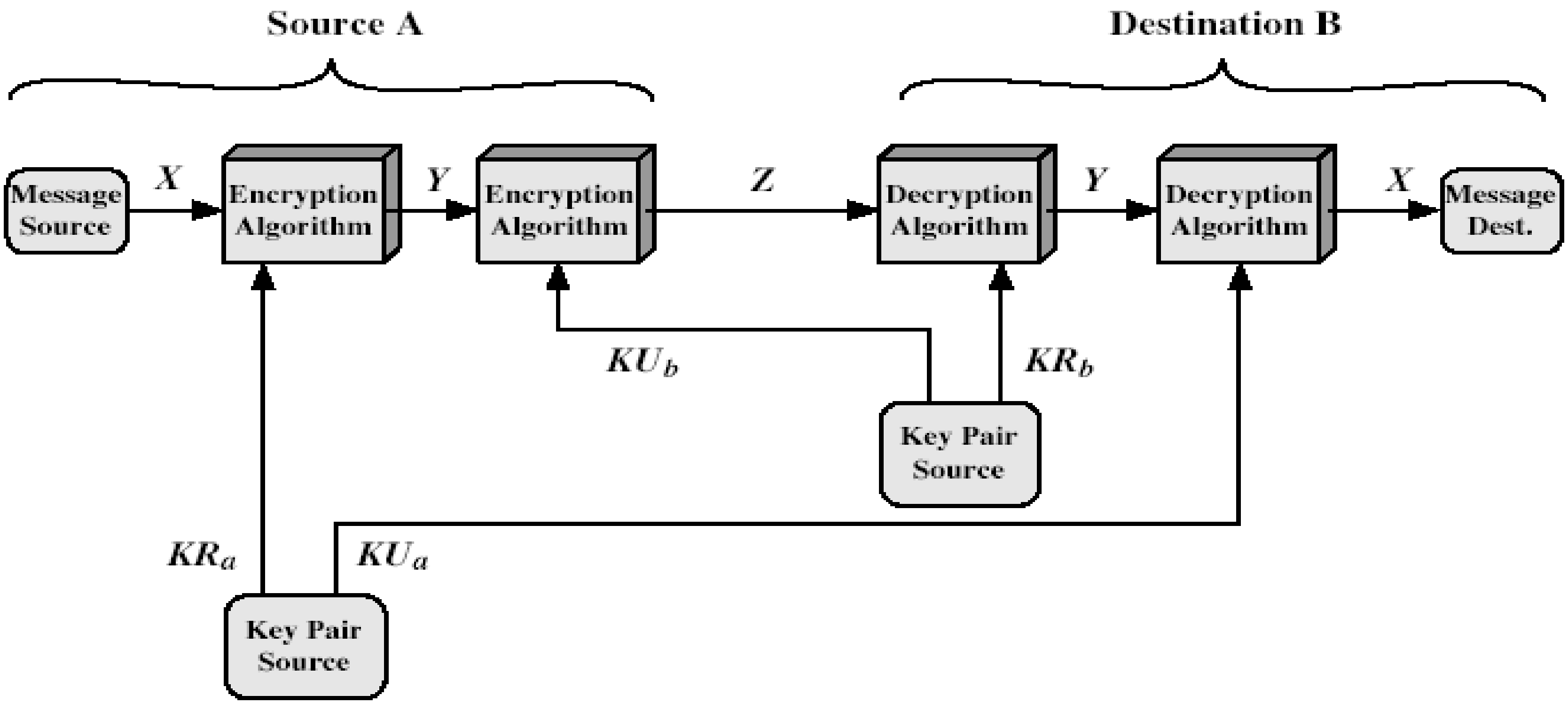


Figure 9.4 Public-Key Cryptosystem: Secrecy and Authentication

Public-Key Applications

- can classify uses into 3 categories:
  - **encryption/decryption** (provide secrecy)
  - **digital signatures** (provide authentication)
  - **key exchange** (of session keys)
- some algorithms are suitable for all uses, others are specific to one

Security of Public Key Schemes

- like private key schemes brute force **exhaustive search** attack is always theoretically possible
- but keys used are too large (>512bits)
- security relies on a **large enough** difference in difficulty between **easy** (en/decrypt) and **hard** (cryptanalyse) problems
- more generally the **hard** problem is known, its just made too hard to do in practise
- requires the use of **very large numbers**
- hence is **slow** compared to private key schemes

Symmetric-key vs. Public-key

Symmetric-key Encryption	Public-key Encryption
• Needed to work:	Needed to work:
1. The same algorithm with the same key is used for encryption and decryption. 2. The sender and receiver must share the secret key and the algorithm.	1. One algorithm is used for encryption and decryption with a pair of keys, one for encryption and one for decryption. 2. The sender and receiver must each have one of the matched keys (not the same one).

Symmetric-key vs. Public-key

Symmetric-key Encryption	Public-key Encryption
Needed for security:	Needed for security:
1. The key must be kept secret. 2. It must be impossible or at least impractical to decipher a message if no other information is available. 3. Knowledge of the algorithm plus the samples of ciphertext must be insufficient to determine the key.	1. One of the two keys must be kept secret. 2. It must be impossible or at least impractical to decipher a message if no other information is available. 3. Knowledge of the algorithm plus one of the keys plus the samples of ciphertext must be insufficient to determine the key.

Principles of Information Security

Dr. Ashok Kumar Das

Center for Security, Theory and Algorithmic Research
International Institute of Information Technology, Hyderabad

E-mail: *ashok.das@iiit.ac.in*
URL: http://www.iiit.ac.in/people/faculty/ashokkdas
Personal Home Page: http://sites.google.com/site/iitkgpakdas/

Diffie-Hellman Key Exchange Protocol

Overview

- Diffie-Hellman key agreement (also called exponential key exchange or Diffie-Hellman key exchange) provided the first practical solution to the secret key distribution problem.
- It is based on public-key cryptography.
- This protocol enables two parties, say A and B, which have never communicated before, to establish a mutual secret key by exchanging messages over a public channel.
- This scheme only resists passive attacks.
- However, this protocol is vulnerable to active attacks.

Diffie-Hellman Key Exchange Protocol (continued...)

Global Public Elements

• q : a sufficiently large prime, such that it is intractible to compute the discrete logarithms in Z_q^*.

• α : $\alpha < q$ and α a primitive root of q.

Diffie-Hellman Key Exchange Protocol (continued...)

User A Key Generation

- Select private X_A such that $X_A < q$
- Calculate public Y_A such that $Y_A = \alpha^{X_A} \mod q$

$A \rightarrow B : \{Y_A, q, \alpha\}$
Here $A \rightarrow B : M$ denotes party A sends a message M to party B.

Diffie-Hellman Key Exchange Protocol (continued...)

User B Key Generation

- Select private X_B such that $X_B < q$
- Calculate public Y_B such that $Y_B = \alpha^{X_B} \mod q$

$B \rightarrow A : \{Y_B\}$

Diffie-Hellman Key Exchange Protocol (continued...)

Generation of secret key by User A

- $\boxed{K_{A,B} = (Y_B)^{X_A} \mod q}$

Diffie-Hellman Key Exchange Protocol (continued...)

Generation of secret key by User B

- $\boxed{K_{B,A} = (Y_A)^{X_B} \mod q}$

Diffie-Hellman Key Exchange Protocol (continued...)

Summary

User A	User B
1. Select private X_A	
2. Calculate public Y_A	
3. $\xrightarrow{Y_A = \alpha^{X_A} \mod q}$	
	1. Select private X_B
	2. Calculate public Y_B
	3. $\xleftarrow{Y_B = \alpha^{X_B} \mod q}$
4. $K_{A,B} = (Y_B)^{X_A} \mod q$	
	4. $K_{B,A} = (Y_A)^{X_B} \mod q$

Correctness Proof

$$
\begin{aligned}
K_{A,B} &= (Y_B)^{X_A} \mod q \text{ [User A]} \\
&= (\alpha^{X_B} \mod q)^{X_A} \mod q \\
&= (\alpha)^{X_B \cdot X_A} \mod q \\
&= (\alpha^{X_A})^{X_B} \mod q \\
&= (\alpha^{X_A} \mod q)^{X_B} \mod q \\
&= (Y_A)^{X_B} \mod q \\
&= K_{B,A} \text{ [User B]}
\end{aligned}
$$

Active Attack on Diffie-Hellman Key Exchange: Man-in-the-middle attack

Table: Man-in-the-middle attack

Alice (User A)	Eve (attacker) C	Bob (User B)
1. private: $X_A < q$ public: $Y_A = \alpha^{X_A} \bmod q$ $\xrightarrow{\langle Y_A \rangle}$		
	2. private: $X_C < q$ public: $Y_C = \alpha^{X_C} \bmod q$ $\xleftarrow{\langle Y_C \rangle}$ $\xrightarrow{\langle Y_C \rangle}$	
		3. private: $X_B < q$ public: $Y_B = \alpha^{X_B} \bmod q$ $\xleftarrow{\langle Y_B \rangle}$

Active Attack on Diffie-Hellman Key Exchange: Man-in-the-middle attack (Continued...)

Table: Man-in-the-middle attack (continued...)

Alice (User A)	Eve (attacker) C	Bob (User B)
4. Computes $K_1 = Y_C^{X_A} \bmod q$		
	5. Computes $K_1 = Y_A^{X_C} \bmod q$ $K_2 = Y_B^{X_C} \bmod q$	
		6. Computes $K_2 = Y_C^{X_B} \bmod q$

Alice-Eve key, $K_1 = Y_C^{X_A} \bmod q = Y_A^{X_C} \bmod q = \alpha^{X_A X_C} \bmod q$.
Eve-Bob key, $K_2 = Y_C^{X_B} \bmod q = Y_B^{X_C} \bmod q = \alpha^{X_C X_B} \bmod q$.

Active Attack on Diffie-Hellman Key Exchange: Man-in-the-middle attack (Continued...)

- Alice (User A) chooses $X_A (< q)$, calculates $Y_A = \alpha^{X_A} \bmod q$, and sends Y_A to Bob (User B).
- Eve, the intruder, intercepts Y_A. She chooses $X_C (< q)$, calculates $Y_C = \alpha^{X_C} \bmod q$, and sends Y_C to both Bob and Alice.
- Bob (User B) chooses $X_B (< q)$, calculates $Y_B = \alpha^{X_B} \bmod q$, and sends Y_B to Alice . Y_B is intercepted by Eve and never reaches Alice.
- Alice and Eve calculate $K_1 = Y_C^{X_A} \bmod q = Y_A^{X_C} \bmod q$ $= \alpha^{X_A X_C} \bmod q$, which becomes a shared key between Alice and Eve. Alice, however, thinks that it is a key shared between Bob and Alice.
- Eve and Bob calculate $K_2 = Y_C^{X_B} \bmod q = Y_B^{X_C} \bmod q$ $= \alpha^{X_B X_C} \bmod q$, which becomes a shared key between Bob and Eve. Bob, however, thinks that it is a key shared between Alice and Bob.

Consequences: Active Attack on Diffie-Hellman Key Exchange: Man-in-the-middle attack (Continued...)

- Two keys, instead of one, are created during this attack: one (K_1) between Alice and Eve and other (K_2) between Bob and Eve.
- Suppose Alice wants to send data to Bob.
- Alice encrypts data using the key K_1 and sends to Bob.
- Eve can deciphered the message using the key K_1 and read all the messages.
- Eve can send the message to Bob encrypted using the key K_2
  or
  even change the message
  or
  send a totally new message.
- Bob is fooled into believing that the message has come from Alice.
- Similar situation, when Bob sends messages to Alice.

Defense: Active Attack on Diffie-Hellman Key Exchange: Man-in-the-middle attack (Continued...)

- The station-to-station key agreement method based on the Diffie-Hellman uses authentication to thwart this serious attack.
- This station-to-station key agreement method uses certificates.
- Self study for station-to-station key agreement method.
- Reference: Behrouz A. Forouzan, Cryptography and Network Security, Special Indian Edition.

System and Network Security

Dr. Ashok Kumar Das

IEEE Senior Member
Associate Professor
Center for Security, Theory and Algorithmic Research
International Institute of Information Technology, Hyderabad

E-mail: *iitkgp.akdas@gmail.com, ashok.das@iiit.ac.in*
Homepage: http://www.iiit.ac.in/people/faculty/ashokkdas
Personal Homepage: https://sites.google.com/site/iitkgpakdas/

Message Authentication and Hash Functions

Authentication Functions

Message Authentication

- "Message authentication" is a process to verify the received message came from alleged source and has not been altered.
- Three procedures to produce an authenticator:
  - Message Encryption (Symmetric/public key)
  - Message Authentication Code (MAC)
  - Hash Function

Message Encryption

Symmetric Encryption

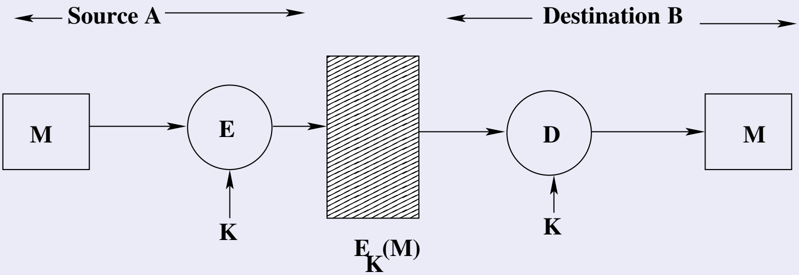


Figure: Symmetric encryption: confidentiality and authentication.

K: symmetric key shared between source A and destination B

Message Encryption

Public-key Encryption

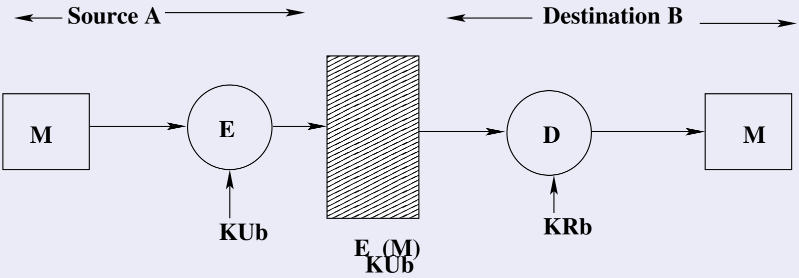


Figure: Public-key encryption: confidentiality.

KU_b: public key of destination B and KR_b: private key of B

Message Encryption

Public-key Encryption

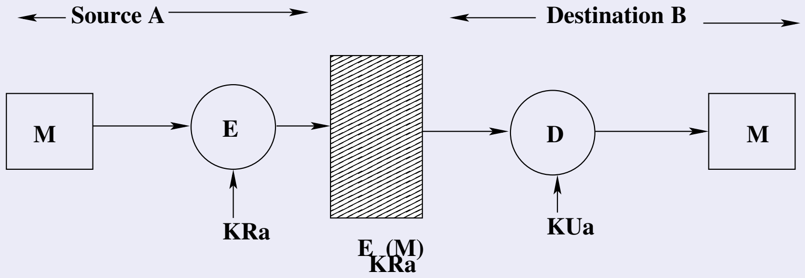


Figure: Public-key encryption: authentication and signature.

KU_a: **public key of source** A **and** KR_a: **private key of** A
$E_{KR_a}(M)$: **signature on message** M **created by the source** A

Message Encryption

Public-key Encryption

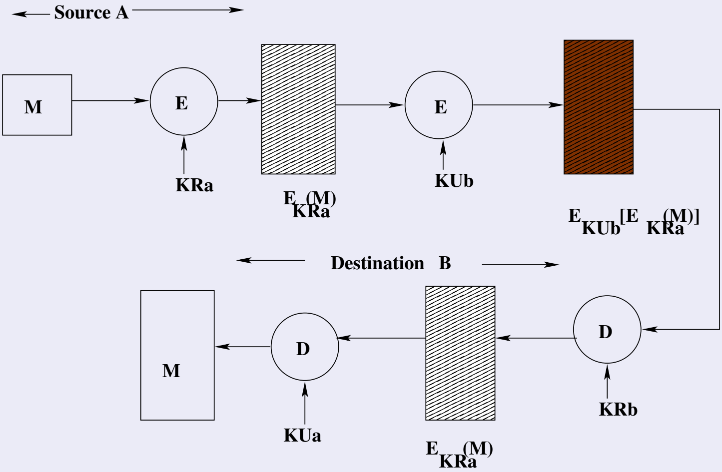


Figure: Public-key encryption: confidentiality, authentication and signature.

Authentication Functions

Message Authentication Code (MAC)

- A public function of the message and a secret key that produces a fixed-length value that serves as an authenticator.
- When user A has to send a message to user B, it calculates MAC as a function of the message and key K as $MAC = C_K(M)$, where M is the input message, C the MAC function, K the shared secret key and MAC the message authentication code.
  Mathematically, $C_K : \{0,1\}^* \longrightarrow \{0,1\}^n$.
- The receiver B is assured that the message has not altered. If an attacker alters the message but does not alter the MAC, then the receiver B's calculation of the MAC will differ from the received MAC.
- Since the attacker does not know the secret key K, the attacker does not have the ability to alter the MAC.

Authentication Functions

Message Authentication Code (MAC)

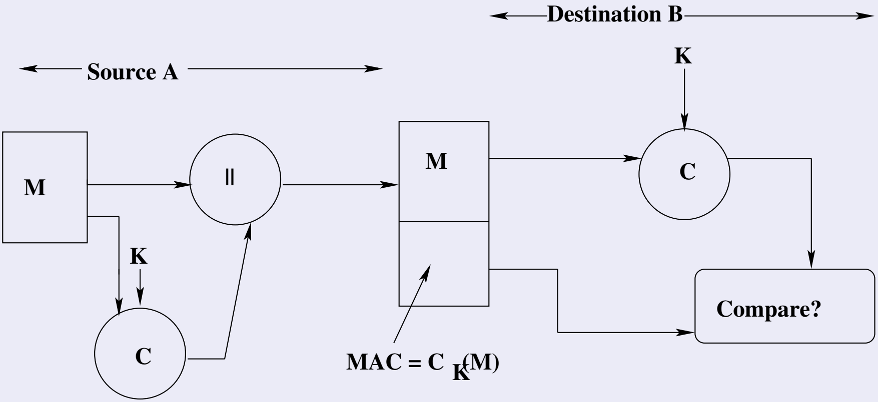


Figure: MAC: message authentication.

Basic Uses of Message Authentication Code

Message authentication

- $A \longrightarrow B : M||C_K(M)$
- Provides authentication
  - Only users A and B share the key K

Basic Uses of Message Authentication Code

Message authentication and confidentiality: authentication tied with plaintext

- $A \longrightarrow B : E_{K_2}[M||C_{K_1}(M)]$
- Provides authentication
  - Only users A and B share the key K_1
- Provides confidentiality
  - Only users A and B share the key K_2

Basic Uses of Message Authentication Code

Message authentication and confidentiality: authentication tied with ciphertext

- $A \longrightarrow B : E_{K_2}[M]||C_{K_1}(E_{K_2}[M])$
- Provides authentication
  - Only users A and B share the key K_1
- Provides confidentiality
  - Only users A and B share the key K_2

Authentication Functions

Hash function

- A cryptographic hash function is an algorithm which accepts a variable length block of data as input and produces a fixed-size bit string, known as cryptographic hash value.
- Hash function can be applied to a large set of inputs which will produce outputs that are evenly distributed, and apparently random.
- Hash function provides data integrity.
- A change to any bit or bits in input data results, with high probability, in a change to the hash value.
- Mathematically, a one-way hash function $h : \{0,1\}^* \rightarrow \{0,1\}^l$ takes an arbitrary-length input $x \in \{0,1\}^*$, and produces a fixed-length (say, l-bits) output $h(x) \in \{0,1\}^l$, called the message digest or hash value.

Authentication Functions

Hash function

The hash function may be the fingerprint of a file, a message, or other data blocks, and has the following attributes.

- h can be applied to a data block of all sizes.
- For any given input x, the message digest $h(x)$ is easy to operate, enabling easy implementation in software and hardware.
- The output length of the message digest $h(x)$ is fixed.
- Deriving the input x from the given hash value $y = h(x)$ and the given hash function $h(\cdot)$ is computationally infeasible. This property is called the *one-way or pre-image resistance* property.
- For any given input x, finding any other input $y \neq x$ so that $h(y) = h(x)$ is computationally infeasible [*weak-collision resistant or second pre-image resistance* property].
- Finding a pair of inputs (x, y), with $x \neq y$, so that $h(x) = h(y)$ is computationally infeasible [*strong-collision resistant or collision resistance* property].

Authentication Functions

Formal definition of a one-way hash function $h(\cdot)$

Definition (Collision-resistant one-way hash function)

A one-way collision-resistant hash function $h : \{0, 1\}^* \rightarrow \{0, 1\}^l$ is a deterministic algorithm that takes an input as an arbitrary length binary string $x \in \{0, 1\}^*$ and outputs a binary string $h(x) \in \{0, 1\}^l$ of fixed-length l. The formalization of an adversary $\mathcal{A}$'s advantage in finding collision is as follows.

$$Adv_{\mathcal{A}}^{HASH}(t) = Pr[(x, x') \leftarrow_R \mathcal{A} : x \neq x' \text{ and } h(x) = h(x')],$$

where $Pr[X]$ denotes the probability of an event X, and $(x, x') \leftarrow_R \mathcal{A}$ denotes the pair (x, x') is selected randomly by $\mathcal{A}$. In this case, the adversary $\mathcal{A}$ is allowed to be probabilistic and the probability in the advantage is computed over the random choices made by the adversary $\mathcal{A}$ with the execution time t. By an (ϵ, t)-adversary $\mathcal{A}$ attacking the collision resistance of $h(\cdot)$, we mean that the runtime of $\mathcal{A}$ is at most t and that $Adv_{(A)}^{HASH}(t) \leq \epsilon$.

Authentication Functions

References

- Palash Sarkar. A Simple and Generic Construction of Authenticated Encryption with Associated Data. ACM Transactions on Information and System Security, Vo. 13, No. 4, pp. 33, 2010.
- Douglas R. Stinson. Some Observations on the Theory of Cryptographic Hash Functions. Designs, Codes and Cryptography (Springer), Vol. 38, No. 2, pp.259-277, 2006.

Applications of a one-way hash function

- There are many applications of the hash functions, for examples, in the field of cryptology and information security, notably in digital signatures, message authentication codes (MACs), and other forms of authentication.
- Thus, a hash function becomes the basis of many cryptographic protocols.

Basic Uses of Hash Function $h(\cdot)$

Encrypt message plus hash code

- $A \longrightarrow B : E_K[M||h(M)]$
- Provides confidentiality
  - Only users A and B share the key K
- Provides authentication
  - $h(M)$ is cryptographically protected

Basic Uses of Hash Function $h(\cdot)$

Encrypt hash code- shared secret key

- $A \longrightarrow B : M || E_K[h(M)]$
- Provides authentication
  - $h(M)$ is cryptographically protected

Basic Uses of Hash Function $h(\cdot)$

Encrypt hash code- sender's private key

- $A \longrightarrow B : M || E_{KR_a}[h(M)]$
- Provides authentication and digital signature
  - $h(M)$ is cryptographically protected
  - Only A could create the signature $E_{KR_a}[h(M)]$

System and Network Security

Dr. Ashok Kumar Das

Center for Security, Theory and Algorithmic Research
International Institute of Information Technology, Hyderabad

E-mail: *ashok.das@iiit.ac.in*
URL: http://www.iiit.ac.in/people/faculty/ashokkdas
Personal Home Page: http://sites.google.com/site/iitkgpakdas/

Topics covered so far...

- Basic Preliminaries of Computer Networks
- Intro to Cryptography: basic attacks
- Symmetric key cryptosystem: Data Encryption Standard (DES) and its variants
- Public key cryptosystem: RSA
- Diffie-Hellman key exchange protocol
- **Lab Assignment 1.** Design of an end to end messaging system like WhatsApp

Message Authentication and Hash Functions

Authentication Functions

Message Authentication

- "Message authentication" is a process to verify the received message came from alleged source and has not been altered.
- Three procedures to produce an authenticator:
  - Message Encryption (Symmetric/public key)
  - Message Authentication Code (MAC)
  - Hash Function

Message Encryption

Symmetric Encryption

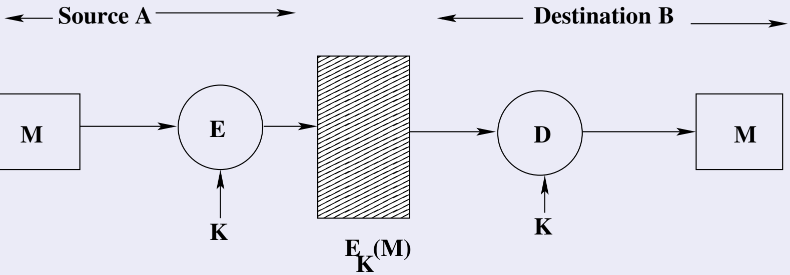


Figure: Symmetric encryption: confidentiality and authentication.

K*: symmetric key shared between source *A* and destination *B

Message Encryption

Public-key Encryption

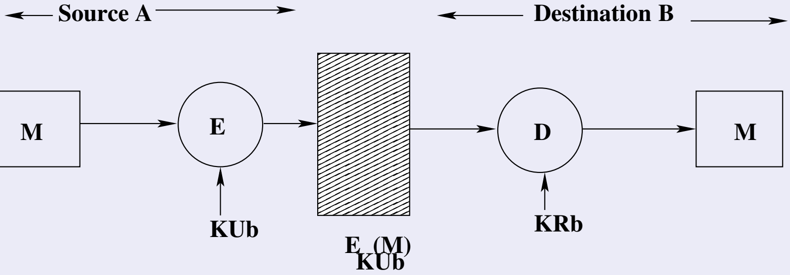


Figure: Public-key encryption: confidentiality.

KU_b: **public key of destination** B **and** KR_b: **private key of** B

Public-key Encryption

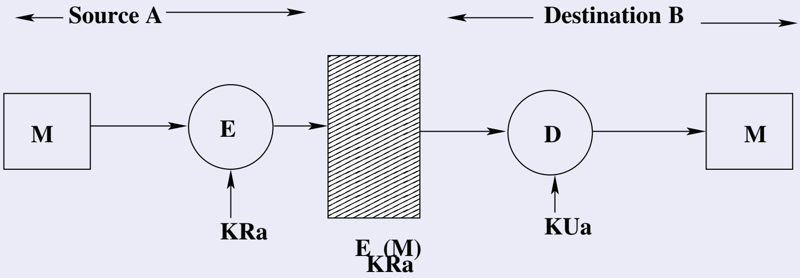


Figure: Public-key encryption: authentication and signature.

KU_a: **public key of source** A **and** KR_a: **private key of** A

$E_{KR_a}(M)$: **signature on message** M **created by the source** A

Message Encryption

Public-key Encryption

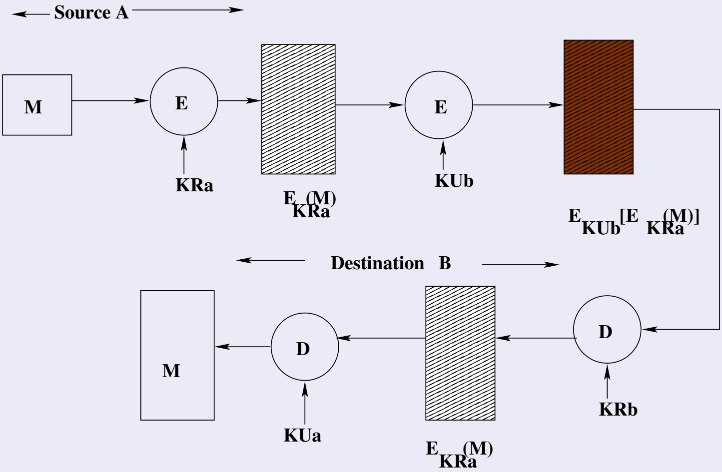


Figure: Public-key encryption: confidentiality, authentication and signature.

Authentication Functions

Message Authentication Code (MAC)

- A public function of the message and a secret key that produces a fixed-length value that serves as an authenticator.
- When user A has to send a message to user B, it calculates MAC as a function of the message and key K as $MAC = C_K(M)$, where M is the input message, C the MAC function, K the shared secret key and MAC the message authentication code.
  Mathematically, $C_K : \{0,1\}^* \longrightarrow \{0,1\}^n$.
- The receiver B is assured that the message has not altered. If an attacker alters the message but does not alter the MAC, then the receiver B's calculation of the MAC will differ from the received MAC.
- Since the attacker does not know the secret key K, the attacker does not have the ability to alter the MAC.

Authentication Functions

Message Authentication Code (MAC)

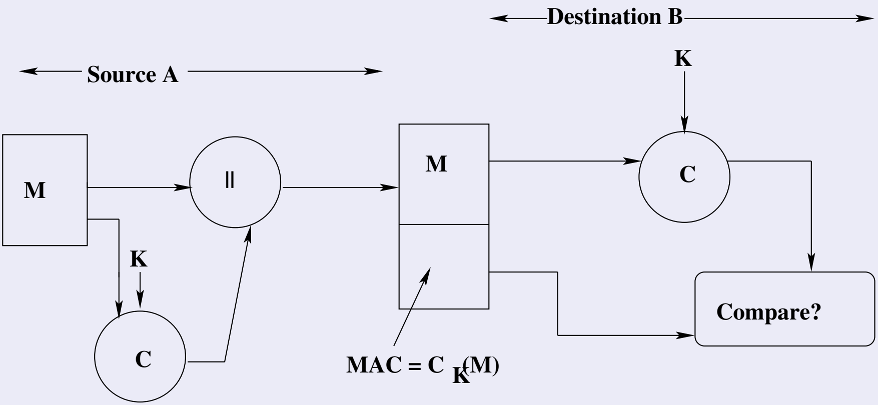


Figure: MAC: message authentication.

Basic Uses of Message Authentication Code

Message authentication

- $A \longrightarrow B : M||C_K(M)$
- Provides authentication
  - Only users A and B share the key K

Basic Uses of Message Authentication Code

Message authentication and confidentiality: authentication tied with plaintext

- $A \longrightarrow B : E_{K_2}[M||C_{K_1}(M)]$
- Provides authentication
  - Only users A and B share the key K_1
- Provides confidentiality
  - Only users A and B share the key K_2

Basic Uses of Message Authentication Code

Message authentication and confidentiality: authentication tied with ciphertext

- $A \longrightarrow B : E_{K_2}[M]||C_{K_1}(E_{K_2}[M])$
- Provides authentication
  - Only users A and B share the key K_1
- Provides confidentiality
  - Only users A and B share the key K_2

Authentication Functions

Hash function

- A cryptographic hash function is an algorithm which accepts a variable length block of data as input and produces a fixed-size bit string, known as cryptographic hash value.
- Hash function can be applied to a large set of inputs which will produce outputs that are evenly distributed, and apparently random.
- Hash function provides data integrity.
- A change to any bit or bits in input data results, with high probability, in a change to the hash value.
- Mathematically, a one-way hash function $h : \{0, 1\}^* \rightarrow \{0, 1\}^l$ takes an arbitrary-length input $x \in \{0, 1\}^*$, and produces a fixed-length (say, l-bits) output $h(x) \in \{0, 1\}^l$, called the message digest or hash value.

Authentication Functions

Hash function

The hash function may be the fingerprint of a file, a message, or other data blocks, and has the following attributes.

- h can be applied to a data block of all sizes.
- For any given input x, the message digest $h(x)$ is easy to operate, enabling easy implementation in software and hardware.
- The output length of the message digest $h(x)$ is fixed.
- Deriving the input x from the given hash value $y = h(x)$ and the given hash function $h(\cdot)$ is computationally infeasible. This property is called the *one-way or pre-image resistance* property.
- For any given input x, finding any other input $y \neq x$ so that $h(y) = h(x)$ is computationally infeasible [*weak-collision resistant or second pre-image resistance* property].
- Finding a pair of inputs (x, y), with $x \neq y$, so that $h(x) = h(y)$ is computationally infeasible [*strong-collision resistant or collision resistance* property].

Authentication Functions

Formal definition of a one-way hash function $h(\cdot)$

Definition (Collision-resistant one-way hash function)

A one-way collision-resistant hash function $h : \{0,1\}^* \rightarrow \{0,1\}^l$ is a deterministic algorithm that takes an input as an arbitrary length binary string $x \in \{0,1\}^*$ and outputs a binary string $h(x) \in \{0,1\}^l$ of fixed-length l. The formalization of an adversary $\mathcal{A}$'s advantage in finding collision is as follows.

$$Adv_{\mathcal{A}}^{HASH}(t) = Pr[(x, x') \leftarrow_R \mathcal{A} : x \neq x' \text{ and } h(x) = h(x')],$$

where $Pr[X]$ denotes the probability of an event X, and $(x, x') \leftarrow_R \mathcal{A}$ denotes the pair (x, x') is selected randomly by $\mathcal{A}$. In this case, the adversary $\mathcal{A}$ is allowed to be probabilistic and the probability in the advantage is computed over the random choices made by the adversary $\mathcal{A}$ with the execution time t. By an (ϵ, t)-adversary $\mathcal{A}$ attacking the collision resistance of $h(\cdot)$, we mean that the runtime of $\mathcal{A}$ is at most t and that $Adv_{(A)}^{HASH}(t) \leq \epsilon$.

Authentication Functions

References

- Palash Sarkar. A Simple and Generic Construction of Authenticated Encryption with Associated Data. ACM Transactions on Information and System Security, Vo. 13, No. 4, pp. 33, 2010.
- Douglas R. Stinson. Some Observations on the Theory of Cryptographic Hash Functions. Designs, Codes and Cryptography (Springer), Vol. 38, No. 2, pp.259-277, 2006.

Authentication Functions

Applications of a one-way hash function

- There are many applications of the hash functions, for examples, in the field of cryptology and information security, notably in digital signatures, message authentication codes (MACs), and other forms of authentication.
- Thus, a hash function becomes the basis of many cryptographic protocols.

Basic Uses of Hash Function $h(\cdot)$

Encrypt message plus hash code

- $A \longrightarrow B : E_K[M||h(M)]$
- Provides confidentiality
  - Only users A and B share the key K
- Provides authentication
  - $h(M)$ is cryptographically protected

Basic Uses of Hash Function $h(\cdot)$

Encrypt hash code- shared secret key

- $A \longrightarrow B : M || E_K[h(M)]$
- Provides authentication
  - $h(M)$ is cryptographically protected

Basic Uses of Hash Function $h(\cdot)$

Encrypt hash code- sender's private key

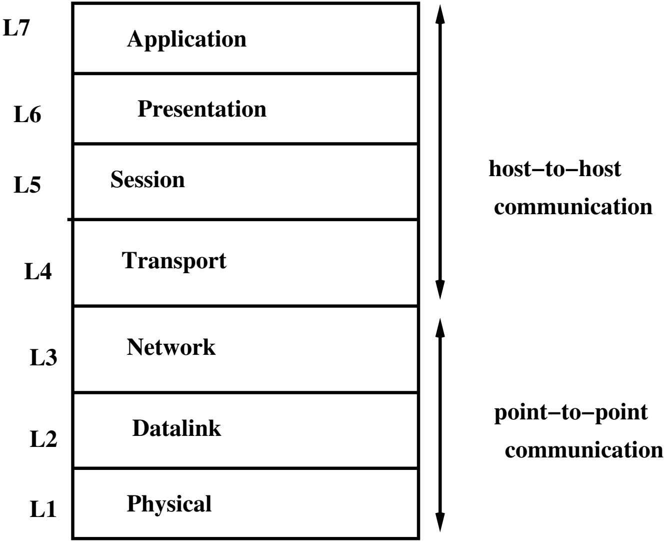
- $A \longrightarrow B : M||E_{KR_a}[h(M)]$
- Provides authentication and digital signature
  - $h(M)$ is cryptographically protected
  - Only A could create the signature $E_{KR_a}[h(M)]$

Encrypting Communications Channels

Encrypting Communications Channels

- This is the classical Alice and Bob problem:
  Alice wants to send Bob a secure message.
- What does she do?
- She encrypts the message.
- In theory, this encryption can take place at any layer in the OSI (Open Systems Interconnect) communication model.

The OSI Reference Model



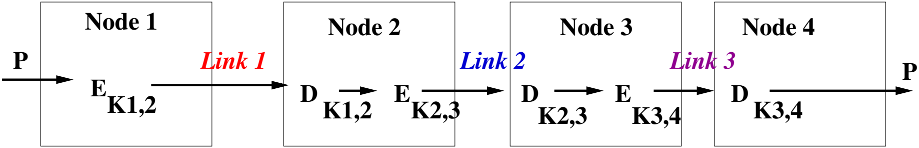
Encrypting Communications Channels

- In practice, it takes place either at the lowest layers (one and two) or at the higher layers.
- If it takes place at the lowest layers, it is called ***link-by-link encryption (LLE)***; everything going through a particular data link is encrypted.
- If it takes place at higher layers, it is called ***end-to-end encryption (EEE)***; the data are encrypted selectively and stay encrypted until they are decrypted by the intended final recipient.

Link-by-link encryption

- The easiest place to add encryption is at the physical layer.
- The interfaces to the physical layer are generally standardized, and it is easy to connect hardware encryption devices at this point.
- These devices encrypt all data passing through them, including data, routing information, and protocol information.
- They can be used on any type of digital communication link.
- On the other hand, any intelligent switching or storing nodes between the sender and the receiver need to decrypt the data stream before processing it.

Link-by-link encryption



P: plaintext message;
$K_{1,2}$: key shared between nodes 1 and 2;
$K_{2,3}$: key shared between nodes 2 and 3;
$K_{3,4}$: key shared between nodes 3 and 4;
$E_K(\cdot)$: encryption using the key K;
$D_K(\cdot)$: decryption using the key K.

Link-by-link encryption

Advantages

- Easier operation, since it can be made transparent to the user. That is, everything is encrypted before being sent over the link.
- Only one set of keys per link is required.
- Provides traffic-flow security, since any routing information is encrypted.

Link-by-link encryption

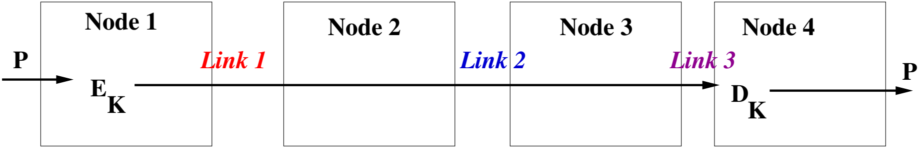
Disadvantages

- Data is exposed in the intermediate nodes.
- The biggest problem with encryption at the physical layer is that each physical link in the network needs to be encrypted: Leaving any link unencrypted reveals the security of the entire network.
  If the network is large, the cost may quickly become prohibitive for this kind of encryption.
- Additionally, every node in the network must be protected, since it processes unencrypted data.
  If all the network's users trust one another, and all nodes are in secure locations, this may be tolerable.

End-to-end encryption

- This approach is to put encryption equipment between the network layer and the transport layer.
- The encryption device must understand the data according to the protocols up to layer three and encrypt only the transport data units, which are then recombined with the un-encrypted routing information and sent to lower layers for transmission.
- This approach avoids the encryption/decryption problem at the physical layer.
- By providing EEE, the data remains encrypted until it reaches its final destination.

End-to-end encryption



P: plaintext message;
K: key shared between nodes 1 and 4;
$E_K(\cdot)$: encryption using the key K;
$D_K(\cdot)$: decryption using the key K.

End-to-end encryption

Advantages

- Higher secrecy level.

End-to-end encryption

Disadvantages

- The primary problem with EEE is that the routing information for the data is not encrypted; a good cryptanalyst can learn much from who is talking to whom, at what times and for how long, without ever knowing the contents of those conversations.
- Key management is also more difficult since individual users must make sure they have common keys.
- Traffic analysis is possible, since routing information is not encrypted.

Combining the Two: Link-by-link encryption and End-to-end encryption

- Combining the two, while most expensive, is the most effective way of securing a network.
- Encryption of each physical link makes any analysis of the routing information impossible, while end-to-end encryption reduces the threat of unencrypted data at the various nodes in the network.
- Key management for the two schemes can be completely separate:
  The network managers can take care of encryption at the physical level, while the individual users have responsibility for end-to-end encryption.

Comparing link-by-link encryption and end-to-end encryption

Link-by-link encryption	End-to-end encryption
Security within hosts	
1. Message exposed in sending host.	1. Message encrypted in sending host.
2. Message exposed in intermediate nodes.	2. Message remains encrypted in intermediate nodes.

Comparing link-by-link encryption and end-to-end encryption

Link-by-link encryption	End-to-end encryption
Role of user	
1. Applied by sending host.	1. Applied by sending process.
2. Invisible to user.	2. User applies encryption.
3. Host maintains encryption.	3. User must find algorithm.
4. One facility for all users.	4. User selects encryption.
5. Can be done in hardware.	5. More easily done in software.
6. All or no messages encrypted.	6. User chooses to encrypt or not, for each message.

Comparing link-by-link encryption and end-to-end encryption

Link-by-link encryption	End-to-end encryption
Implementation concerns	
1. Requires one key per host pair.	1. Requires one key per user pair.
2. Requires encryption hardware or software at each host.	2. Requires encryption hardware or software at each node.
3. Provides node authentication.	3. Provides user authentication.

System and Network Security

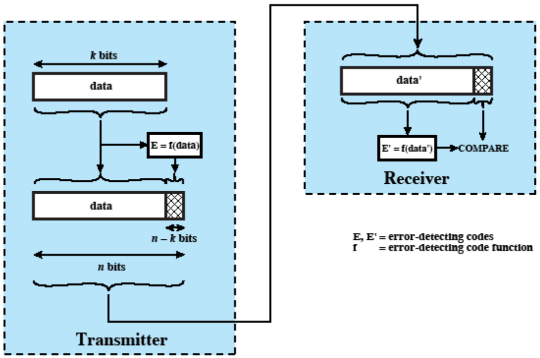
Dr. Ashok Kumar Das

IEEE Senior Member
Associate Professor
Center for Security, Theory and Algorithmic Research
International Institute of Information Technology, Hyderabad

E-mail: *iitkgp.akdas@gmail.com, ashok.das@iiit.ac.in*
Homepage: http://www.iiit.ac.in/people/faculty/ashokkdas
Personal Homepage: https://sites.google.com/site/iitkgpakdas/

Security at the Datalink Layer

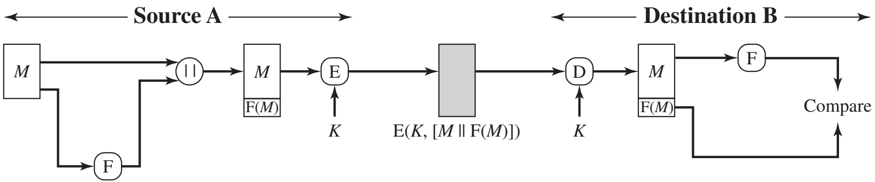
Error Detection Process



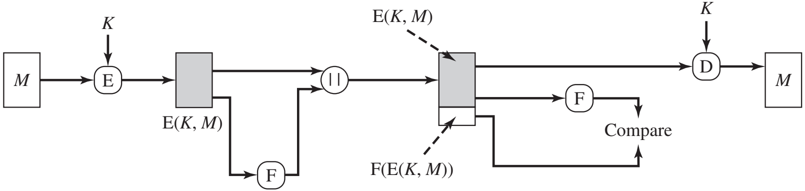
Internal and External Error Control

- It may be difficult to determine *automatically* if incoming ciphertext decrypts to intelligible plaintext.
- For example, if the plaintext is a binary object file or digitized X-rays, determination of properly formed and therefore authentic plaintext may be difficult.
- Thus, an opponent could achieve a certain level of disruption simply by issuing messages with random content purporting to come from a legitimate user.
- One solution to this problem is to force the plaintext to have some structure that is easily recognized but that cannot be replicated without recourse to the encryption function.
- We could, for example, append an error-detecting code, also known as a frame check sequence (FCS) or checksum, to each message before encryption.

Internal and External Error Control



(a) Internal error control



(b) External error control

M: plaintext message; F: function that produces an FCS (frame check sequence); $||$: concatenation operation; $E(K, M)$: encryption of M using key K; $D(K, M)$: decryption of M using key K; K: shared key between source A and destination B.

Internal and External Error Control

- Note that the order in which the FCS and encryption functions are performed is critical.
- With internal error control, authentication is provided because an opponent would have difficulty generating ciphertext that, when decrypted, would have valid error control bits.
- If instead the FCS is the outer code, an opponent can construct messages with valid error-control codes.
- Although the opponent cannot know what the decrypted plaintext will be, he or she can still hope to create confusion and disrupt operations.

Password Management

Password Management

Password Protection

- The front line of defense against intruders is the password system.
- Virtually all multiuser systems require that a user provide not only a name or identifier (ID) but also a password.
- The password serves to authenticate the ID of the individual logging on to the system. In turn, the ID provides security in the following ways:
    - The ID determines whether the user is authorized to gain access to a system. In some systems, only those who already have an ID filed on the system are allowed to gain access.
    - The ID determines the privileges accorded to the user. A few users may have supervisory or "superuser" status that enables them to read files and perform functions that are especially protected by the operating system.
    - The ID is used in what is referred to as discretionary access control. For example, by listing the IDs of the other users, a user may grant permission to them to read files owned by that user.

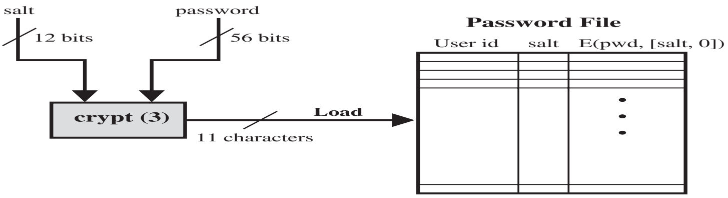
Password Management

INTERNATIONAL INSTITUTE OF INFORMATION TECHNOLOGY HYDERABAD

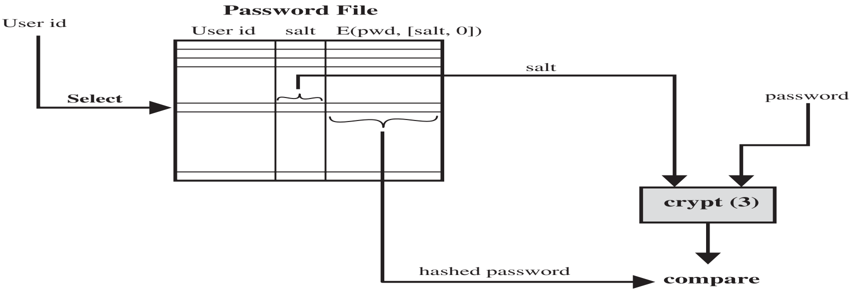
The Vulnerability of Passwords

- To understand the nature of the threat to password-based systems, let us consider a scheme that is widely used on UNIX, in which passwords are never stored in the clear.
- Each user selects a password of up to eight printable characters in length. This is converted into a 56-bit value (using 7-bit ASCII) that serves as the key input to an encryption routine.
- The encryption routine, known as crypt(3), is based on DES. The DES algorithm is modified using a 12-bit "salt" value. Typically, this value is related to the time at which the password is assigned to the user. The modified DES algorithm is exercised with a data input consisting of a 64-bit block of zeros. The output of the algorithm then serves as input for a second encryption. This process is repeated for a total of 25 encryptions. The resulting 64-bit output is then translated into an 11-character sequence.

Password Management in UNIX



(a) Loading a new password



(b) Verifying a password

Password Management

The Vulnerability of Passwords

The salt serves three purposes:

- It prevents duplicate passwords from being visible in the password file. Even if two users choose the same password, those passwords will be assigned at different times. Hence, the "extended" passwords of the two users will differ.
- It effectively increases the length of the password without requiring the user to remember two additional characters.
- It prevents the use of a hardware implementation of DES, which would ease the difficulty of a brute-force guessing attack.

Password Management

The Vulnerability of Passwords

There are two threats to the UNIX password scheme:

- First, a user can gain access on a machine using a guest account or by some other means and then run a password guessing program, called a password cracker, on that machine. The attacker should be able to check hundreds and perhaps thousands of possible passwords with little resource consumption.
- In addition, if an opponent is able to obtain a copy of the password file, then a cracker program can be run on another machine at leisure. This enables the opponent to run through many thousands of possible passwords in a reasonable period.

Password Management

The Vulnerability of Passwords

[Alvare, A. “How Crackers Crack Passwords or What Passwords to Avoid.” Proceedings, UNIX Security Workshop II, August 1990.] reports the following techniques for learning passwords:

- Try default passwords used with standard accounts that are shipped with the system. Many administrators do not bother to change these defaults.
- Exhaustively try all short passwords (those of one to three characters).
- Try words in the system’s online dictionary or a list of likely passwords. Examples of the latter are readily available on hacker bulletin boards.
- Collect information about users, such as their full names, the names of their spouse and children, boyfriends and girlfriends, pictures in their office, and books in their office that are related to hobbies.
- Try users’ phone numbers, Social Security numbers, and room numbers.
- Try all legitimate license plate numbers for this state.
- Use a Trojan horse to bypass restrictions on access.

Password Management

Password Selection Strategies

Four basic techniques are in use:

- User education
- Computer-generated passwords
- Reactive password checking
- Proactive password checking

Password Management

Password Selection Strategies

User education:

- Users can be told the importance of using hard-to-guess passwords and can be provided with guidelines for selecting strong passwords.
- This user education strategy is unlikely to succeed at most installations, particularly where there is a large user population or a lot of turnover.
- Many users will simply ignore the guidelines.

Password Management

Password Selection Strategies

Computer-generated passwords:

- If the passwords are quite random in nature, users will not be able to remember them.
- Even if the password is pronounceable, the user may have difficulty remembering it and so be tempted to write it down.
- In general, computer-generated password schemes have a history of poor acceptance by users.

Password Management

Password Selection Strategies

Reactive password checking:

- This strategy is one in which the system periodically runs its own password cracker to find guessable passwords.
- The system cancels any passwords that are guessed and notifies the user.
- This tactic has a number of drawbacks: It is resource intensive if the job is done right.

Password Selection Strategies

Proactive password checking:

- This is the most promising approach to improve password security.
- A user is allowed to select his or her own password. However, at the time of selection, the system checks to see if the password is allowable and, if not, rejects it.
- Such checkers are based on the philosophy that, with sufficient guidance from the system, users can select memorable passwords from a fairly large password space that are not likely to be guessed in a dictionary attack.
- For example, the following rules could be enforced:
  - All passwords must be at least eight characters long.
  - In the first eight characters, the passwords must include at least one each of uppercase, lowercase, numeric digits, and punctuation marks.

Zipf's Law in Passwords

- D. Wang, H. Cheng, P. Wang, X. Huang, and G. Jian, "Zipf's Law in Passwords," **IEEE Transactions on Information Forensics and Security**, vol. 12, no. 11, pp. 2776–2791, Nov 2017.

Biometrics and Fuzzy Extractor

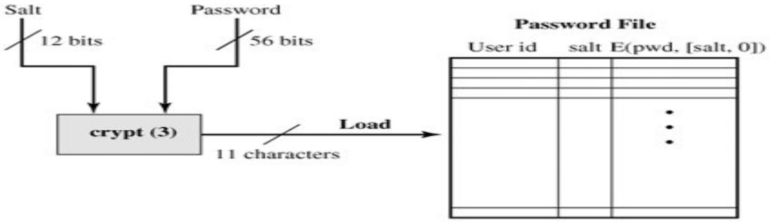
- Let $\mathcal{M} = \{0,1\}^v$ denote a finite v-dimensional metric space of biometric data points, $d : \mathcal{M} \times \mathcal{M} \rightarrow \mathbb{Z}^+$ a distance function, which can be used to calculate the distance between two points based on the metric chosen, l the number of bits of the output string, and t the error tolerance, where $\mathbb{Z}^+$ represents the set of all positive integers.
- The fuzzy extractor is a tuple $(\mathcal{M}, l, t)$, which is composed of the following two algorithms, called *Gen* and *Rep*:
  - **Gen:** It is a probabilistic algorithm, which takes a biometric information $B_i \in \mathcal{M}$ as input, and then outputs a secret key data $\sigma_i \in \{0,1\}^l$ and a public reproduction parameter τ_i, where $Gen(B_i) = \{\sigma_i, \tau_i\}$.
  - **Rep:** This is a deterministic algorithm, which takes a noisy biometric information $B_i' \in \mathcal{M}$ and a public parameter τ_i and t related to B_i, and then it reproduces (recovers) the biometric key data σ_i. In other words, we have $Rep(B_i', \tau_i) = \sigma_i$ provided that the condition $d(B_i, B_i') \leq t$ is met.

Password Management

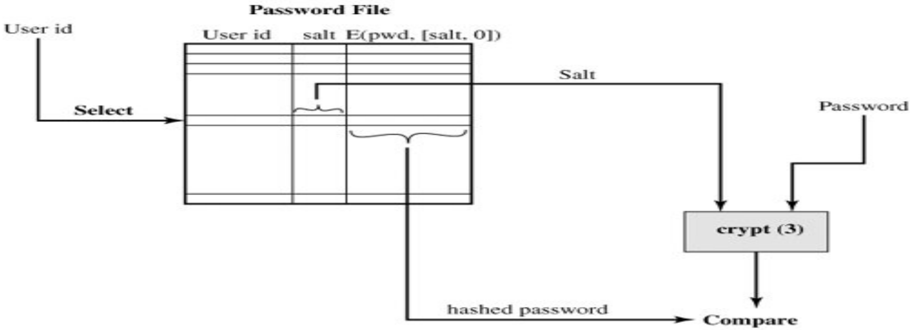
Biometrics and Fuzzy Extractor

- Vanga Odelu, **Ashok Kumar Das**, and Adrijit Goswami. ”A Secure Biometrics-Based Multi-Server Authentication Protocol using Smart Cards,” in ***IEEE Transactions on Information Forensics and Security***, Vol. 10, No. 9, pp. 1953 - 1966, 2015. (2015 SCI Impact Factor: 2.441) [This article is one of the top 50 most frequently downloaded documents for Popular Articles (June 2015 - June 2016)]

Password Management in UNIX



(a) Loading a new password



(b) Verifying a password

System and Network Security

Dr. Ashok Kumar Das

IEEE Senior Member
Associate Professor
Center for Security, Theory and Algorithmic Research
International Institute of Information Technology, Hyderabad

E-mail: *ashok.das@iiit.ac.in*
Homepage: http://www.iiit.ac.in/people/faculty/ashokkdas
Personal Homepage: https://sites.google.com/site/iitkgpakdas/

Welcome
to
System and Network Security

Course Contents

- **Network Security**
  - Overview of Digital Signature Standards
  - **Encrypting communication channels:**
    Link-by-Link Encryption (LLE) and End-to-End Encryption (EEE).
  - **Security at the Application Layer**:
    Authentication Applications: Kerberos, X.509 authentication service.
    PGP, S/MIME, Password Management, Secure Electronics Transaction.
  - **Security at the Transport Layer/Web Security**: Web security considerations, Secure sockets layer and Transport Layer security.
  - **Security at the Network Layer**: IPSec
  - **Security at the Datalink Layer**: Internal and External Error Controls.

Course Contents (Continued...)

- **System Security**
  - Intruders: Intruders, Intrusion detection, Intrusion prevention.
  - Malicious Software: Virus and related threats, Virus countermeasures.
  - Firewalls: Firewall design principles, Trusted systems.
  - Software Vulnerabilities: Phishing, Buffer overflow (BOF), Heap overflow, Format string attacks, Cross-site scripting (XSS), SQL Injection.
  - Malware Threats and Security Solutions

Course Contents (Continued...)

- **Advanced Topics in Network/System Security**
  - **IoT Security:** IoT architecture, various IoT applications, security requirements, security attacks, threat model for the IoT ecosystem, taxonomy of security protocols
  - **Blockchain Technology:** Various applications of blockchain of Things (BCoT), centralized versus decentralized models, types of blockchain, brief overview of various consensus algorithms, block formation and addition in a blockchain, applications of blockchains

Preferred Textbooks and References

- William Stallings, “Cryptography and Network Security: Principles and Practices,” Pearson Education, 6th Edition, 2014.
- Bernard Menezes, “Network Security and Cryptography,” Cengage Learning, 2010.
- Behrouz A. Forouzan, “Cryptography and Network Security,” Special Indian Edition, 2010.
- Research papers [IEEE, ACM, Elsevier, Springer]

Prerequisites

- Design and Analysis of Algorithms (desirable)
- Basics of Operating Systems
- Programming (C, C++, Java, Python, etc.)
- Principle of Information Security (NOT mandatory)

Grading and Examinations Policy

Grading Method:: Relative

- Quizes (2): 30%
- Open Test (1): 20%
- Lab Assignments (includes coding): 50%
- All examinations are ONLINE, and open books and notes

Lab Assignments (Marks: 50)

- FIVE lab assignments for implementing network and system security aspects.
- Programming Languages to be used: C, C++, Java, Python, and assembly language programming,
  socket programming, AVISPA (Automated Validation of Internet Security Protocols and Applications), ProVerif or Scyther...

Thank You!!!

System and Network Security

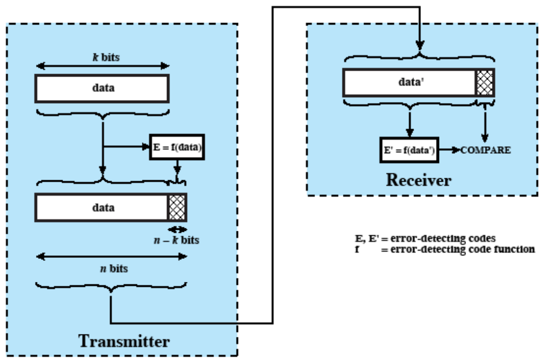
Dr. Ashok Kumar Das

IEEE Senior Member
Associate Professor
Center for Security, Theory and Algorithmic Research
International Institute of Information Technology, Hyderabad

E-mail: *iitkgp.akdas@gmail.com, ashok.das@iiit.ac.in*
Homepage: http://www.iiit.ac.in/people/faculty/ashokkdas
Personal Homepage: https://sites.google.com/site/iitkgpakdas/

Security at the Datalink Layer

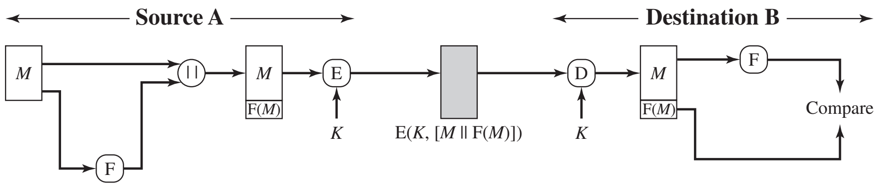
Error Detection Process



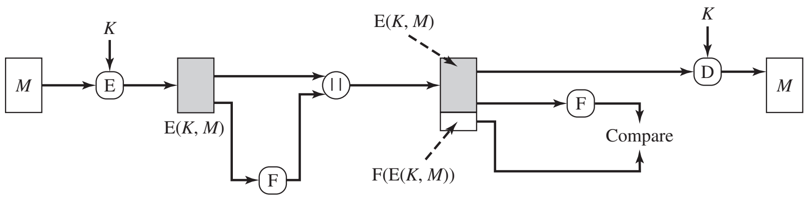
Internal and External Error Control

- It may be difficult to determine *automatically* if incoming ciphertext decrypts to intelligible plaintext.
- For example, if the plaintext is a binary object file or digitized X-rays, determination of properly formed and therefore authentic plaintext may be difficult.
- Thus, an opponent could achieve a certain level of disruption simply by issuing messages with random content purporting to come from a legitimate user.
- One solution to this problem is to force the plaintext to have some structure that is easily recognized but that cannot be replicated without recourse to the encryption function.
- We could, for example, append an error-detecting code, also known as a frame check sequence (FCS) or checksum, to each message before encryption.

Internal and External Error Control



(a) Internal error control



(b) External error control

M: plaintext message; F: function that produces an FCS (frame check sequence); $||$: concatenation operation; $E(K, M)$: encryption of M using key K; $D(K, M)$: decryption of M using key K; K: shared key between source A and destination B.

Internal and External Error Control

- Note that the order in which the FCS and encryption functions are performed is critical.
- With internal error control, authentication is provided because an opponent would have difficulty generating ciphertext that, when decrypted, would have valid error control bits.
- If instead the FCS is the outer code, an opponent can construct messages with valid error-control codes.
- Although the opponent cannot know what the decrypted plaintext will be, he or she can still hope to create confusion and disrupt operations.

Introduction to Cryptography

Dr. Ashok Kumar Das

Center for Security, Theory and Algorithmic Research
International Institute of Information Technology, Hyderabad

E-mail: *ashok.das@iiit.ac.in*
URL: http://www.iiit.ac.in/people/faculty/ashokkdas
https://sites.google.com/site/iitkgpakdas/

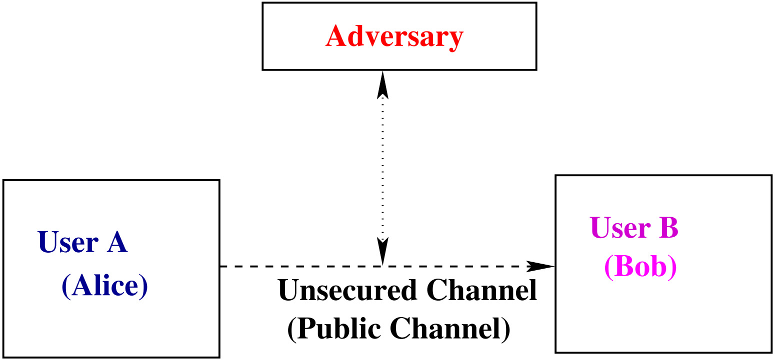
Overview of Cryptography

What is Cryptography?

- Cryptography is the study of **mathematical techniques** related to aspects of information security such as confidentiality, data integrity, entity authentication, message authentication (data origin authentication) and non-repudiation.
- Cryptography is not the only means of providing information security, but rather one set of techniques.
- Now-a-days, cryptography has moved from an art to a science. Thus, cryptography is the science of keeping secrets secret.

Introduction to Cryptography

Consider the following simple two-party communication model:



Introduction to Cryptography

- An **"adversary"** is an entity in a two-party communication which is neither the sender nor the receiver, and which tries to defeat the information security service being provided between the sender and the receiver.
- A **"channel"** is a means of conveying information from one entity to another entity.
- An **"unsecured channel"** is one from which parties other than the sender and the receiver can reorder, delete, insert, or read the data being transmitted.
- A **"secured channel"** is one from which an adversary does not have the ability to reorder, delete, insert, or read the data being transmitted.

Types of adversary

- A **“passive adversary”** is an adversary who is only capable of reading information from an unsecured channel.
- An **“active adversary”** is an adversary who is capable to transmit, alter, or delete information on an unsecured channel.

Cryptographic goals (objectives)

- **Confidentiality:** Privacy (confidentiality) is a service of keeping information secret from all but those who are authorized to see it.
- **Data integrity:** ensuring information has not been altered by unauthorized or unknown means.
- **Entity authentication or identification:** Corroboration of the identity of an entity (i.e., a person, a computer terminal, a credit card, etc.).
- **Message or data origin authentication:** Corroborating the source of information.
- **Non-repudiation:** Preventing the denial of the previous session (preventing the malicious nodes to hide their activities).

Cryptographic goals (objectives)

- **Authorization:** Conveyance to another entity such as a person or group of users. It ensures that the nodes (users) those who are authorized can be involved in providing information to network services.
- **Signature:** a means to bind information to an entity.
- **Access control:** restricting access to resources to privileged entity.
- **Certification:** endorsement of information by a trusted entity.

Cryptographic goals (objectives)

We need also to consider the forward and backward secrecy when new nodes join in the network and existing nodes depart from the network.

- ***Forward secrecy:*** When a node (user) leaves the network, it must not read any future messages after its departure.
- ***Backward secrecy:*** When a new node (user) joins in the network, it must not read any previously transmitted message.

Introduction to Cryptography

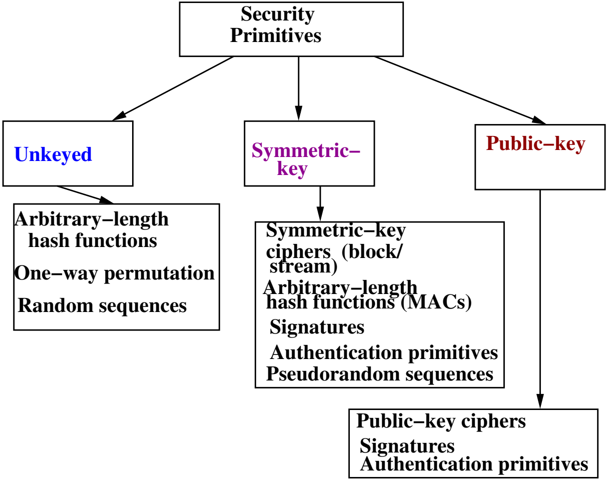


Figure: A taxonomy of cryptographic primitives

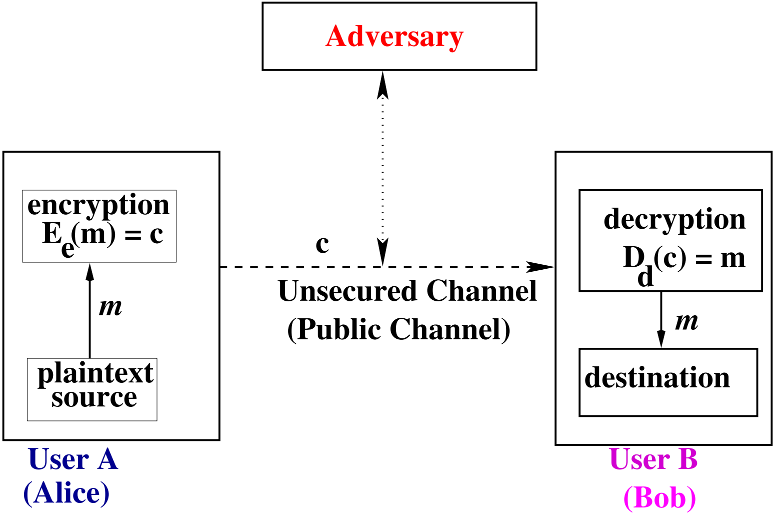
Evaluation criteria for the primitives

- ***Level of security:*** This is usually difficult to quantify.
- ***Functionality:*** Primitives will need to be combined to meet various information security objectives.
- ***Methods of operation:*** One primitive could provide very different functionality depending on its mode of operation or usage.
- ***Performance:*** This refers to the efficiency of a primitive in a particular mode of operation (For example, an encryption algorithm may be rated by the number of bits per second which it can encrypt).
- ***Easy of implementation:*** This might include the complexity of implementing the primitives in either a software or hardware environment.

Note that the relative importance of various criteria is very much dependent on the application and resources availability.

Introduction to Cryptography

Consider the following simple two-party communication model with encryption:



- **Encryption scheme 1:** Have only the encryption and decryption functions and these are kept secret to the sender and receiver only. No key is used in this method.
- **Encryption scheme 2:** Key is being used. However, the encryption and decryption functions are made public.

Quiz: Why keys are necessary? Why not just choose one encryption function and its corresponding decryption function?

- **Security of the scheme**
  - Depends entirely on the secrecy of the key
  - Does not depend on the secrecy of the algorithm (Needs to be public for criticism!)
- Hence, we make the **assumptions** as follows:
  - Algorithms for encryption/decryption are known to the public
  - Keys used are kept secret

Cryptology = Cryptography + Cryptanalysis

Definition

An encryption scheme (cipher or cryptosystem) is said to be ***breakable*** if a third party, without prior knowledge of the key pair (e, d) where e is the encryption key and d is the corresponding decryption key, can systematically recover plaintext from corresponding ciphertext within some appropriate time frame.

Goal: We want this problem for an adversary (attacker) to be NP-hard (computationally infeasible).

Definition (Brute-force attack)

An encryption scheme can be broken by trying all possible keys to see which one the communicating parties are using (assuming that the class of encryption functions is public knowledge).
This is called an exhaustive search of the key space.

INTERNATIONAL INSTITUTE OF INFORMATION TECHNOLOGY HYDERABAD

What is meant by "Security lies in the keys" (using brute-force attack)

Key size (bits)	**Number of alternative keys**	**Time required at 10^6 decryptions per microsecond**
32	$2^{32} = 4.3 \times 10^9$	2.15 milliseconds
56	$2^{56} = 7.2 \times 10^{16}$	10 hours
128	$2^{128} = 3.4 \times 10^{38}$	5.4×10^{18} years
168	$2^{168} = 3.7 \times 10^{50}$	5.9×10^{30} years

INTERNATIONAL INSTITUTE OF INFORMATION TECHNOLOGY
HYDERABAD

Definition (Unconditionally secure scheme)

An encryption scheme is “unconditionally secure” if the ciphertext generated by the scheme does not contain enough information to determine uniquely the corresponding plaintext, no matter how many ciphertexts are available. That is, no matter how much time an opponent has, it is impossible for him/her to decrypt the ciphertext, simply because the required information is not there.

Definition (Computationally secure scheme)

An encryption scheme is said to be “computationally secure” if the following two criteria are met:

- The cost of breaking the cipher exceeds the value of the encrypted information.
- The time required to break the cipher exceeds the useful lifetime of the information.

Principles of Information Security

Dr. Ashok Kumar Das

Center for Security, Theory and Algorithmic Research
International Institute of Information Technology, Hyderabad

E-mail: *ashok.das@iiit.ac.in*
URL: http://www.iiit.ac.in/people/faculty/ashokkdas
Personal Home Page: http://sites.google.com/site/iitkgpakdas/

Symmetric-Key Encryption

Hill Cipher

- Another interesting "multi-letter cipher" is the Hill cipher, developed by the Mathematician Lester Hill in 1929.
  Encryption algorithm
  - The encryption algorithm takes m successive plaintext letters $(p_1, p_2, \ldots, p_m)$ and substitutes for them m ciphertext letters $(c_1, c_2, \ldots, c_m)$.
  - The substitution is determined by m linear equations in which each character is assigned a numerical value ($a = 0, b = 1, \ldots, z = 25$).
  - The system can be described as follows:

$$
\begin{aligned}
c_1 &= k_{1,1}p_1 + k_{1,2}p_2 + \ldots + k_{1,m}p_m \pmod{26} \\
c_2 &= k_{2,1}p_1 + k_{2,2}p_2 + \ldots + k_{2,m}p_m \pmod{26} \\
\ldots & \quad \ldots \\
c_m &= k_{m,1}p_1 + k_{m,2}p_2 + \ldots + k_{m,m}p_m \pmod{26}
\end{aligned}
$$

  - This can thus expressed in matrix form: $C = KP \pmod{26}$.

Hill Cipher

- $P = \begin{pmatrix} p_1 \\ p_2 \\ \ldots \\ p_m \end{pmatrix}$ is the plaintext,
- $C = \begin{pmatrix} c_1 \\ c_2 \\ \ldots \\ c_m \end{pmatrix}$ is the ciphertext,
- $K = \begin{pmatrix} k_{1,1} & k_{1,2} & \ldots & k_{1,m} \\ k_{2,1} & k_{2,2} & \ldots & k_{2,m} \\ \ldots & \ldots & \ldots & \ldots \\ k_{m,1} & k_{m,2} & \ldots & k_{m,m} \end{pmatrix}$ is the encryption key (e).

Hill Cipher

Decryption algorithm

- We have $C = KP \pmod{26}$.
- Then $P = K^{-1}C \pmod{26}$ is the original plaintext.
- Note that K^{-1} is the decrypytion key (d).

In general, the Hill system can be expressed as follows:

$$C = E_K(P) = KP \pmod{26}$$
$$P = D_K(C) = K^{-1}C \pmod{26}.$$

Hill Cipher

Problem: Consider the plaintext, P = “paymoremoney” and the encryption key, $K = \begin{pmatrix} 17 & 17 & 5 \\ 21 & 18 & 21 \\ 2 & 2 & 19 \end{pmatrix}$. Encrypt the plaintext P using the key K. Also show the procedure for decrypting the computed ciphertext C to recover the original plaintext P.

Solution:

- We have $m = 3$.
- The encoding scheme is as follows:

a	b	c	...	v	w	x	y	z
0	1	2	...	21	22	23	24	25

Hill Cipher

Solution (Continued...):

- Then $P_1 = \begin{pmatrix} p_1 \\ p_2 \\ p_3 \end{pmatrix} = \begin{pmatrix} 15 \\ 0 \\ 24 \end{pmatrix}$.
- Thus, the ciphertext is $C_1 = \begin{pmatrix} c_1 \\ c_2 \\ c_3 \end{pmatrix} = K \begin{pmatrix} p_1 \\ p_2 \\ p_3 \end{pmatrix}$

  $= \begin{pmatrix} 375 \\ 819 \\ 486 \end{pmatrix} \pmod{26} = \begin{pmatrix} 11 \\ 13 \\ 18 \end{pmatrix}$.
- Using the decoding technique, we have the ciphertext corresponding to the plaintext ‘pay’ is LNS
- Continuing in this fashion, the ciphertext for the entire plaintext is $C = C_1 C_2 C_3 C_4 =$ LNSHDLEWMTRW

Hill Cipher

Solution (Continued...):

- We have $C = KP \pmod{26}$. Then, $P = K^{-1}C \pmod{26}$.
- Here $K = \begin{pmatrix} 17 & 17 & 5 \\ 21 & 18 & 21 \\ 2 & 2 & 19 \end{pmatrix}$.
- $|K| =$ determinant of K
  $= 17(18.19 - 2.21) - 17(21.19 - 2.21) + 5(21.2 - 2.18)$
  $= -939 \pmod{26} = 23 \pmod{26}$.
- The matrix formed by the co-factors of K is given by
  $A = \begin{pmatrix} 300 & -357 & 6 \\ -313 & 313 & 0 \\ 267 & -252 & -51 \end{pmatrix}$.

Hill Cipher

Solution (Continued...):

- Adjoint K is given by $Adj.K = A^T$ (transposition of A)
$$= \begin{pmatrix} 300 & -313 & 267 \\ -357 & 313 & -252 \\ 6 & 0 & -51 \end{pmatrix} \pmod{26}$$
$$= \begin{pmatrix} 14 & 25 & 7 \\ 7 & 1 & 8 \\ 6 & 0 & 1 \end{pmatrix}.$$
- $K^{-1} = \frac{Adj.K}{|K|}$
$$= \frac{1}{23} \begin{pmatrix} 14 & 25 & 7 \\ 7 & 1 & 8 \\ 6 & 0 & 1 \end{pmatrix} = (23^{-1} \pmod{26}) \begin{pmatrix} 14 & 25 & 7 \\ 7 & 1 & 8 \\ 6 & 0 & 1 \end{pmatrix}$$
$\pmod{26}$.

Hill Cipher

Solution (Continued...):

- Apply the extended Euclid's gcd algorithm to find $23^{-1} \pmod{26}$, which becomes -9.
- Then $K^{-1} = \begin{pmatrix} 4 & 9 & 15 \\ 15 & 17 & 6 \\ 24 & 0 & 17 \end{pmatrix}$.
- Now, $P_1 = K^{-1}C_1$, that is
  $$\begin{pmatrix} p_1 \\ p_2 \\ p_3 \end{pmatrix} = K^{-1} \begin{pmatrix} c_1 \\ c_2 \\ c_3 \end{pmatrix} = \begin{pmatrix} 431 \\ 494 \\ 570 \end{pmatrix} \pmod{26}$$
  $$= \begin{pmatrix} 15 \\ 0 \\ 24 \end{pmatrix}$$
  $=$ 'PAY'.

Strength of Hill Cipher

- It is strong against a ciphertext-only attack.
- For m-letter Hill cipher it hides the letter frequency upto $(m-1)$.
- It is broken against the plaintext attack.
- The attack is as follows:
  - For an $m \times m$ Hill cipher, suppose we have m plaintext-ciphertext pairs, each of length m. We label the pairs as

$$\begin{aligned} P_j &= (p_{1j}, p_{2j}, \ldots, p_{mj}) \\ C_j &= (c_{1j}, c_{2j}, \ldots, c_{mj}) \end{aligned}$$

    such that $C_j = KP_j$ for $1 \leq j \leq m$, and for some unknown key matrix K.
  - Define two $m \times m$ matrices $X = (p_{ij})_{m \times m}$ and $Y = (c_{ij})_{m \times m}$.

Strength of Hill Cipher

- Form the matrix equation:

$$Y = KX \pmod{26}.$$

- **Case I:** If X^{-1} exists, then we can determine $K = YX^{-1} \pmod{26}$.
- **Case II:** If X^{-1} does not exist, then a new version of X can be formed with additional plaintext-ciphertext pairs until an invertible X is obtained.

Hill Cipher

Problem (Cryptanalysis of Hill Cipher): Let the plaintext be "friday" be encrypted using an 2×2 Hill cipher to yield the ciphertext PQCFKU. Determine the unknown encryption key K.

Solution:

- We have $m = 2$.
- plaintext, P : FR ID AY, and ciphertext, C : PQ CF KU
- Then three plaintext-ciphertext pairs (using above encoding technique) are

$$P_1 = \begin{pmatrix} F \\ R \end{pmatrix} = \begin{pmatrix} 5 \\ 17 \end{pmatrix},$$

$$C_1 = \begin{pmatrix} P \\ Q \end{pmatrix} = \begin{pmatrix} 15 \\ 16 \end{pmatrix},$$

Hill Cipher

Solution (Continued...):

- $P_2 = \begin{pmatrix} I \\ D \end{pmatrix} = \begin{pmatrix} 8 \\ 3 \end{pmatrix}$,
  $C_2 = \begin{pmatrix} C \\ F \end{pmatrix} = \begin{pmatrix} 2 \\ 5 \end{pmatrix}$,
  $P_3 = \begin{pmatrix} A \\ Y \end{pmatrix} = \begin{pmatrix} 0 \\ 24 \end{pmatrix}$,
  $C_3 = \begin{pmatrix} K \\ U \end{pmatrix} = \begin{pmatrix} 10 \\ 20 \end{pmatrix}$,
  such that $C_j = KP_j$ where the key K needs to be determined.

Hill Cipher

Solution (Continued...):

- To determine the unknown key matrix $K_{2\times 2}$ we use the first two plaintext-ciphertext pairs:
$$\begin{pmatrix} 15 & 2 \\ 16 & 5 \end{pmatrix} = K \begin{pmatrix} 5 & 8 \\ 17 & 3 \end{pmatrix} \pmod{26}$$
- Let $Y = KX \pmod{26}$, where $X = \begin{pmatrix} 5 & 8 \\ 17 & 3 \end{pmatrix}$, and
$Y = \begin{pmatrix} 15 & 2 \\ 16 & 5 \end{pmatrix}$.
- We have then $K = YX^{-1} \pmod{26}$.
- Determine $X^{-1} = \begin{pmatrix} 9 & 2 \\ 1 & 15 \end{pmatrix} \pmod{26}$.
- Then $K = YX^{-1} = \begin{pmatrix} 137 & 60 \\ 149 & 107 \end{pmatrix} \pmod{26} = \begin{pmatrix} 7 & 8 \\ 19 & 3 \end{pmatrix}$.

Hill Cipher

Solution (Continued...):

- This result is verified by testing the remaining plaintext-ciphertext pair (P_3, C_3) to be confident that the derived K is correct with a very high probablity.
- Note that
$$KP_3 = \begin{pmatrix} 7 & 8 \\ 19 & 3 \end{pmatrix} \begin{pmatrix} 0 \\ 24 \end{pmatrix} \pmod{26}$$
$$= \begin{pmatrix} 192 \\ 72 \end{pmatrix} \pmod{26}$$
$$= \begin{pmatrix} 10 \\ 20 \end{pmatrix} \pmod{26} = C_3.$$
- Conclusion: The derived encryption key is $K = \begin{pmatrix} 7 & 8 \\ 19 & 3 \end{pmatrix}$.

Principles of Information Security

Dr. Ashok Kumar Das

Center for Security, Theory and Algorithmic Research
International Institute of Information Technology, Hyderabad

E-mail: *ashok.das@iiit.ac.in*
URL: http://www.iiit.ac.in/people/faculty/ashokkdas
Personal Home Page: http://sites.google.com/site/iitkgpakdas/

Symmetric-Key Encryption

Symmetric-Key Encryption

Model of conventional encryption

- Consider an encryption scheme consisting of
  - the set of encryption transformations $\{E_e : e \in K\}$
  - the set of corresponding decryption transformations $\{D_d : d \in K\}$, where K is the key space.
- The encryption scheme is said to be S-key or symmetric-key, if for each associated encryption/decryption key pair (e, d), it is computationally "easy" to determine d from e and to determine e from d.
- In most practical symmetric-key encryption schemes, $e = d$.
- Other terms used are single-key, one-key, private-key and conventional encryption.

Symmetric-Key Encryption

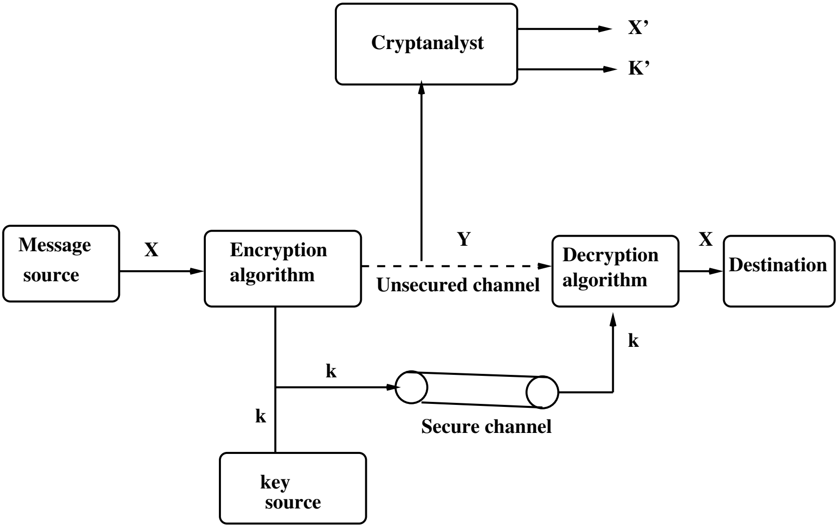


Figure: Model of conventional encryption

Symmetric-Key Encryption

Model of conventional encryption

- With the message $X = [X_1, X_2, \ldots, X_n]$ and the encryption key k as input, the encryption algorithm forms the ciphertext $Y = [Y_1, Y_2, \ldots, Y_n]$.
- $Y = E_k[X]$
- $Y_i = E_k[X_i]$, for $i = 1, 2, \ldots, n$.
- $X = D_k[Y]$
- $X_i = D_k[Y_i]$, for $i = 1, 2, \ldots, n$.

Symmetric-Key Encryption

Classical Techniques

- There are two classical techniques in conventional or symmetric-key encryption scheme:
    - Substitution Techniques: Involve the substitution of a ciphertext symbol for a plaintext symbol.
    - Transposition Techniques: A very different kind of mapping is achieved by performing some sort of permutation on the plaintext letters.

Block Cipher

- A block cipher is an encryption scheme which breaks up the plaintext messages to be transmitted into strings (called ‘blocks’) of a fixed length t over an alphabet A, and encrypts one block at a time.
- Two important class of block ciphers are
  - Substitution Techniques
  - Transposition Techniques

Caesar Cipher

- It is the earliest known use of a substitution cipher, and the simplest, was by Julius Caesar.
- Each letter of the alphabet is replaced with the letter standing the three places further down the alphabet.
- For example,
  plaintext: meet me after the new year party
  ciphertext: PHHW PH DIWHU WKH QHZ BHDU SDUWB
- Each letter is wrapped around, so that the letter following *Z* is A. Define the transformation by listing all possibilities as follows.

plaintext:	a	b	c	...	v	w	x	y	z
ciphertext:	D	E	F	...	Y	Z	A	B	C

Symmetric-Key Encryption

Caesar Cipher

- Encoding technique: Let us assign a numerical equivalent to each letter:

a	b	c	...	v	w	x	y	z
0	1	2	...	21	22	23	24	25

- Mathematical model:
  - Encryption: For each plaintext letter p, substitute the ciphertext letter c: $c = E_k(p) = (p + 3) \pmod{26}$, where $k = 3$.
  - Decryption: For each ciphertext letter c, substitute the plaintext letter p: $p = D_k(c) = (c - 3) \pmod{26}$, where $k = 3$.

Symmetric-Key Encryption

The Generalized Caesar Cipher

- A shift may be of any amount, so that the general Caesar algorithm is as follows.
- Mathematical model
  - Encryption: For each plaintext letter p, substitute the ciphertext letter c: $c = E_k(p) = (p + k) \pmod{26}$, where $0 \leq k \leq 25$.
  - Decryption: For each ciphertext letter c, substitute the plaintext letter p: $p = D_k(c) = (c - k) \pmod{26}$, where $0 \leq k \leq 25$.

Security issues of the Caesar cipher

- If it is known that a given ciphertext is a Caesar cipher, then a brute-force cryptanalysis is easily performed.
- The key space K in this case contains 25 keys, that is $|K| = 25$.
- Attacker simply tries all the 25 possible keys.
- In this case, the attacker could be able to recover the plaintext as well as the encryption key k from the ciphertext easily (It is an example of Ciphertext-only attack (COA)).

Symmetric-Key Encryption

Characteristics of the Caesar cipher

- The encryption an decryption algorithms are known.
- There are only 25 keys to try.
- The language of the plaintext is known and easily recognizable.

Symmetric-Key Encryption

Problem [Caesar cipher]: Consider a set of alphabet of definition, which consists of the following characters. Also, we use the encoding techinique given below. Assume that the uppper case and lower case letters have the same digital alphabet. 00 indicates a space between words.

A = 01	K = 11	U = 21	1 = 31
B = 02	L = 12	V = 22	2 = 32
C = 03	M = 13	W = 23	3 = 33
D = 04	N = 14	X = 24	4 = 34
E = 05	O = 15	Y = 25	5 = 35
F = 06	P = 16	Z = 26	6 = 36
G = 07	Q = 17	, = 27	7 = 37
H = 08	R = 18	. = 28	8 = 38
I = 09	S = 19	? = 29	9 = 39
J = 10	T = 20	0 = 30	! = 40

Symmetric-Key Encryption

Problem [Caesar cipher] (Continued...)

- **(a)** Encrypt the plaintext
  The brown fox is quick!
  using a key $k = 29$.
- **(b)** Encrypt the plaintext
  Meet me after the toga party at 10 P.M. night at IIIT main gate.
  using a key $k = 13$.

Monoalphabetic Cipher

- With only 25 possible keys, the Caesar cipher is far from secure.
- A dramatic increase in the key space can be achieved by allowing an arbitrary substitution.
- Recall the assignment for the Caesar cipher:

plaintext:	a	b	c	...	v	w	x	y	z
ciphertext:	D	E	F	...	Y	Z	A	B	C

- Instead, the cipher line is replaced by any permutation of the 26 alphabetic characters.

Monoalphabetic Cipher

- Then there are 26! or greater than 4×10^{26} possible keys in the key space K.
- Hence, the brute-force attack is not possible.
- Such an approach is referred as a “monoalphabetic substitution cipher”, because a single cipher alphabet (mapping from plain alphabet to cipher alphabet) is used per message.
- Monoalphabetic ciphers are easy to break because they reflect the frequency data of the original alphabet.