COMP90043 Cryptography Notes

Essay Notes

二十世纪二十年代,一次性密码本被发明并被证实是不可破解的,此发现奠定了信息论及其子系统的理论基础,因为一次性密码键盘需要极长的密钥,在大多数建立中所需成本过高.相比之下大多数密码系统的安全性在于破解者在密钥未知的情况下发现明文的计算难度,这个问题属于计算复杂性和算法分析的范畴.

在信息流通中,如果一个信道的安全性不能满足用户需求则信道被视为公共信道.任何信道都可能受到窃听和注入的威胁.

加密学的核心问题在于隐私验证和信息验证, 身份验证可以被拆解为信息验证(上下文定义问题)和用户验证(系统的唯一任务是验证个人的真实身份). 尽管在用户身份验证中表面上没有信息,但这两个问题同源同根. 用户认证中隐含着”我是X用户”的信息,而信息认证只是验证发送信息方的身份.

无条件安全系统**(Unconditionally Secure System)**

定义为无论允许多少计算量,都能抵御任何密码分析攻击的系统,它属于信息论中的shannon理论,关注于在无限计算的情况下获得的最佳性能.

无条件安全系统的安全性源于密码图存在多个有意义的解.一个计算安全的密码图包含足够的信息来确定一个唯一的明文和密钥,其安全性仅在于计算它们的成本.

唯一常用的无条件安全系统是一次性密码本,其中明文与随机选择相同长度的的

密码相结合. 虽然这种系统是安全的,但由于需要大量密钥,在大多数情况下并不实用.

计算机安全系统

需要普遍适用,考虑只需要几百比特的密钥,并可在少量数字硬件或一个只需要几百行代码的软件基础上实现.

计算不可行**(Computationally infeasible)**

以内存使用量或运行时间作为成本衡量,成本虽有限,但不可能达到的巨大量级需求称为计算不可行.

分类

密码系统可以被分为流密码(stream ciphers)和块密码(block ciphers).

流密码**(stream ciphers)**

将明文处理成小块(chunks),通常产生一个伪随即比特序列,然后将该序列加到明文中进行二次调制.

块密码**(block ciphers)**

以纯组合的方式作用域大文本块,输入块中的微小变化也会使结果天差地别.这种特性使得其在许多身份验证应用中存在价值.

在认证系统中,加密技术需要防止攻击者在信道中注入看似真实的新信息,以及防止攻击者通过合并或复制过去的旧信息来制造表面可信的新信息.然而一般以隐私保护作为核心的加密系统并不能防止后一种攻击.

为保证电文真实性,日期和时间功能被添加,

加密技术的强度评估

对系统威胁分类,为保护隐私或验证身份而使用的密码系统可能收到的威胁包括:

1. Ciphertext only attack
   a. 密码分析攻击,攻击者只拥有密文.
   b. 最常用,只需要利用语言的系统性知识(例如英文中字母e出现的频率是13%),以及某些常用单词(例如新的开头”亲爱的先生”).这是系统所能受到的最低阶的威胁,任何能被这种方式攻破的系统都被认为是完全不安全的.
2. Known plaintext attack
   a. 已知明文攻击,攻击者拥有大量相应的明文和密文.
   b. 能够抵御此类攻击的系统使用户无需保密过去的信息,也无需在解密前对这些信息进行密码去除.因为这对系统来说使不合理的负担,例如军事情报和外交信函,类似情况导致许多本应安全的系统被破解.
   c. 虽然这类攻击不常见,但发生频率并不低,如果系统无法抵御此类攻击,则系统不能被认可为安全的系统.
3. Chosen plaintext attack
   a. 攻击者可以提交无限数量的自定义明文信息并检查生成的密文.
   b. 理念在现实中很难安全复现,但存在类似攻击.例如可以向竞争对手泄露自己的提案使其对提案进行加密以传输到对方总部.
   c. 攻击核心在于对手是否可以在己方系统中植入信息.
   d. 常被称为 IFF attack.

根据以上定义,密码分析可以被认知为一个系统识别问题(System identification problem).已知明文攻击和选择明文攻击分别对应于被动和主动系统识别问题. 密码学的目标是建立一个难以识别的系统.

无论是公钥密码系统还是单项认证系统,都不可能是无条件安全的,因为在有限集合中公共信息总是在有限集中决定秘密信息的唯一因素,因此在无线计算下这个问题可以通过简单的直接搜索来破解.

Class Notes

Class 1

Information Security

Mainly about how to prevent attacks, or failing that, to detect attacks on information-based systems.

Cryptography

- Secret Writing
- Refers to the techniques required for protecting data between authorized parties on information communication technologies in the presence of potentially malicious elements.
- Refers to a range of techniques such as Encryption, Signature, Hash functions, assuring Privacy, Integrity, and Authentication of data in the digital world.

Information Security

A broad topic of exchange and processing of information on modern computers and networks.

- **Confidentiality**
  - The information is made available to only those people who are authorized to access it.
  - Preserving authorized restrictions on information access and disclosure, including means for protecting personal privacy and proprietary information.
- **Integrity**
  - The information(message) presented to the other person has not been tampered with.
  - Guarding against improper information modification or destruction, including ensuring information nonrepudiation and authenticity.
- **Availability**
  - The information security information infrastructure is available to the users.
  - Ensuring timely and reliable access to and use of information.

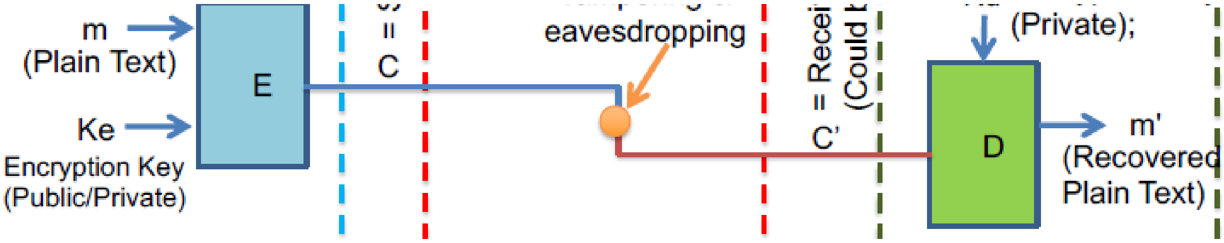
  - Users are freely to use it
- Authentication
  - The property of being genuine and being able to be verified and trusted.
- Accountability
  - A security goal that requires all actions of an entity to be traced uniquely to that entity.

Cyber Security

Refers to management of attacks and risks by adversarial and malicious elements on computers and networks that support modern businesses and economy involving business, government, and community.

Notations & Process

E - Encryption Function - Public
m - Plain text - Public
Ke - Encryption Key(Public/Private)
C - Sent ciphertext
C’ - Received Cipher Text(Could be in error)
D - Decryption Function
Kd - Decryption Key(Private Key)
m’ - Recovered Plain Text
c - ciphertext
c’ - received ciphertext



E - C - C’ - D

Difference

- Ke = Kd
  - Symmetric key also sometimes referred to as private key, but we shall always call symmetric key - Known since antiquity.

- Ke != Kd
    - Asymmetric or Public Key Cryptography
    - Fairly recent since 1974 after the celebrated paper by Diffie Hellman.

Models for Information Security

Traditional Communication Model

- Insecure channel, can be listened to and modified if needed.

Modern Network Model

- Network itself is an adversary. More than two participants. A valid participant also can be an adversary to others. Many models exist.

Diffie-Hellman Idea

One way function

- Given x in domain it is easy to compute f(x)
- Given y in range, it is difficult find x in domain such that f(x)=y

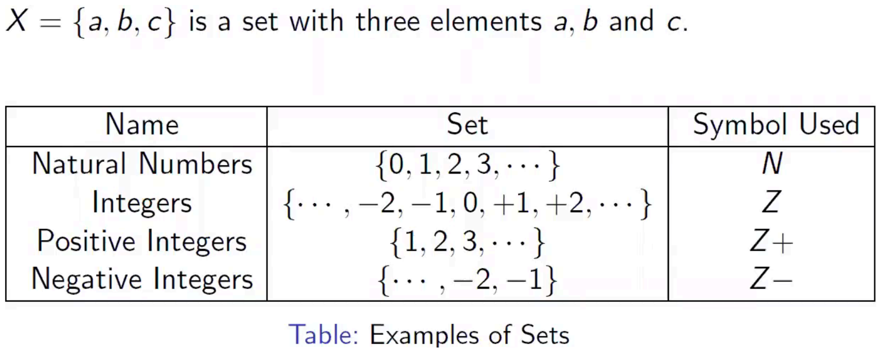
Discrete Logarithm Problem(DLP)

Class 2 Video

Fundamentals

Sets

A set is a collection of objects. The objects are referred to as elements of the set.



$X = \{a, b, c\}$ is a set with three elements a, b and c.

Name	Set	Symbol Used
Natural Numbers	$\{0, 1, 2, 3, \cdots\}$	N
Integers	$\{\cdots, -2, -1, 0, +1, +2, \cdots\}$	Z
Positive Integers	$\{1, 2, 3, \cdots\}$	$Z+$
Negative Integers	$\{\cdots, -2, -1\}$	$Z-$

Table: Examples of Sets

Functions

Defined by <X,Y,f>

- X: a set called domain
- Y: a set called range or codomain
- F: a rule which assigns to each element in X precisely one element in Y. Denoted by f: X → Y.

Alphabet, A - A finite set
Message Space, M - Consists of strings of symbols from an alphabet.
Cipher Text Space, C - Consists of strings of symbols from an alphabet which may differ from the alphabet of M.
Key space, K - A set of key space and an element of K is key.
Encryption function, E_e

$$C = E_e(M)$$

Decryption function, D_d

$$M = D_d(C)$$

Diversability

a|b means a is divisible by a positive integer b.

- a|a
- a|b and b|c implies a|c
- a|b and b|a implies a=b || a=-b
- a|b and a|c implies a|(bx+cy) for all integers x and y.
- a|b implies ca|cb for any c.

Division and Remainders

Remainder modulo

1. a>b
2. c is the largest integer smaller than a and is multiple of b. B|c
3. c = qb < a
4. Then a = c + r = qb + r
5. Q is the quotient and r is called the remainder modulo b.

Prime Numbers

2,3,5,7,11,13,17,19,23,29,31,37,41,47,43,53,59,61,67,71,73,79,83,89,97

GCD computation

gcd(a,b) = gcd(b,r)
gcd(a,b) = gcd(b,(a mod b))
For n in N, GCD(n,n+1) = 1

For n in N, GCD(n^2,n+1) = 1
A function triplet is defined as follow: < {0, 1}E, {0, 1}D, f >, where f:
{0, 1}E− > {0, 1}D What are the sizes for the domain and codomain of function f?
Domain size: 2E; Codomain size: 2D.

Class 2

OSI Security Architecture

Three main aspects.

Security Attacks

1. Attack is any action that compromises the security of information owned by an organization.
2. Threat is a possible potential for violation of security. Attacks might happen.

often threat & attack used to mean same thing (**threat is attack in waiting**)

Generic types of attacks

1. Passive
2. Active

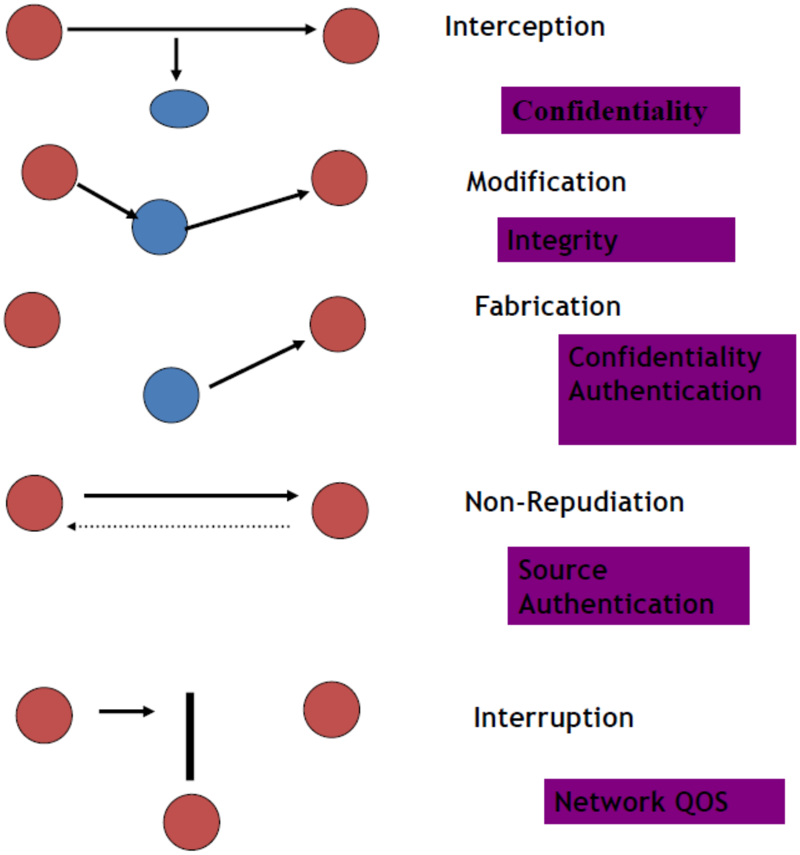
Security Mechanisms

Security Services

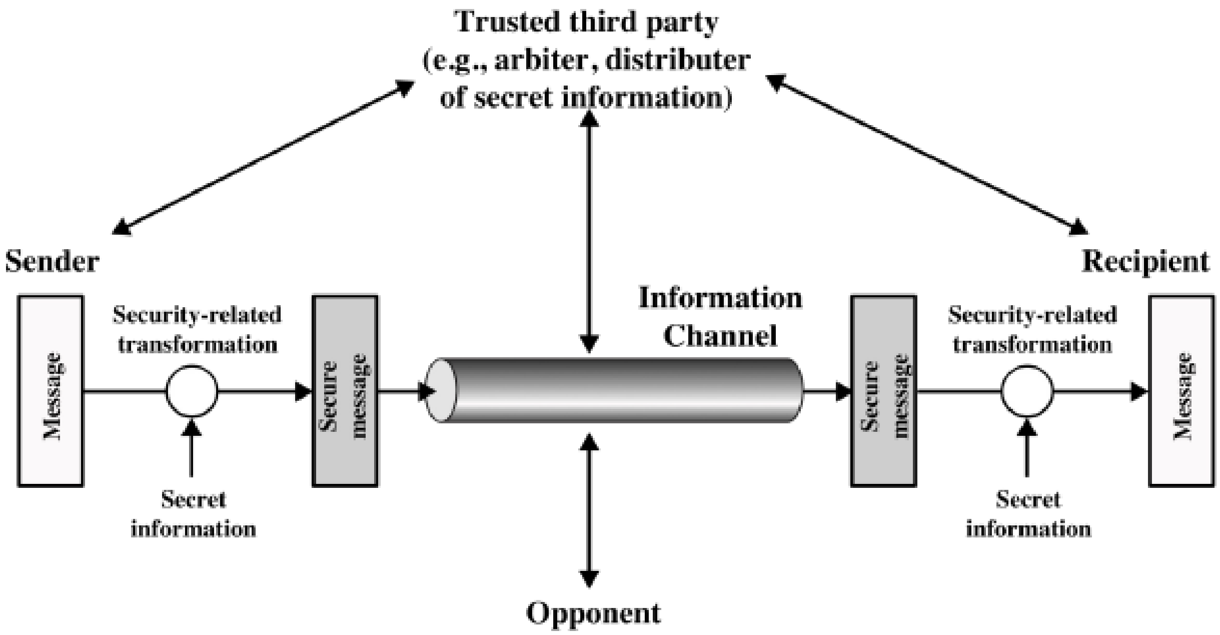
concentrate on Implementation and Mechanism aspects of Information Security. They are defined to address or withstand threats.

- Authentication
- Confidentiality
- Integrity
- Nonrepudiation
- Availability

Security services are defined to address or withstand threats



Model for Network Security



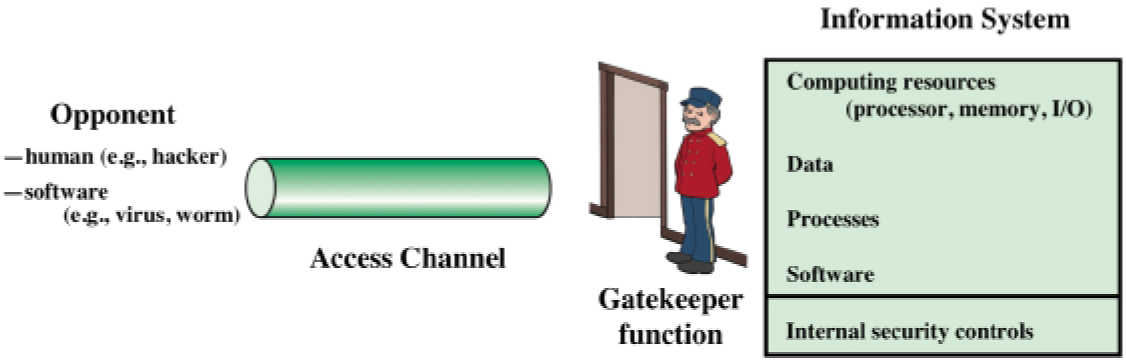
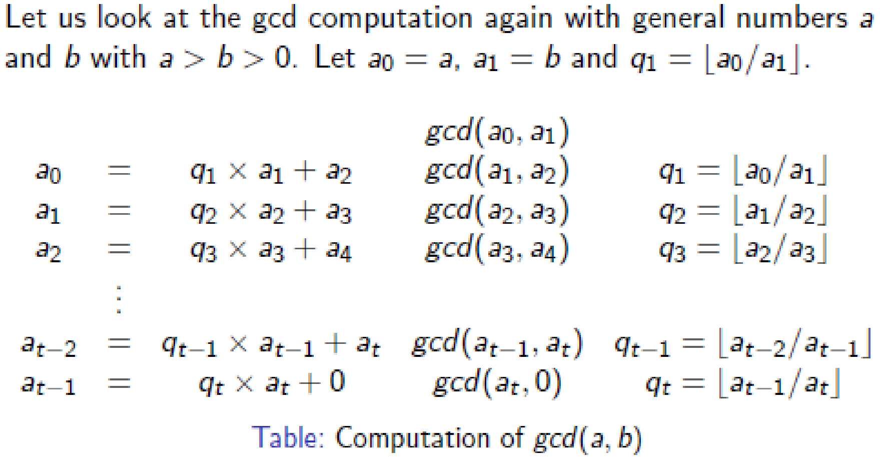


Figure 1.3 Network Access Security Model

Class 3

Extended GCD Algorithm



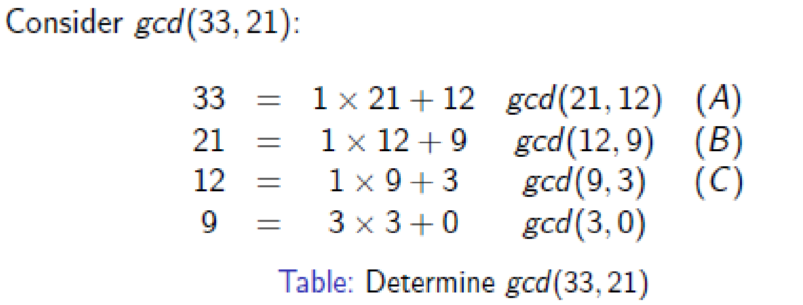
Let us look at the gcd computation again with general numbers a and b with $a > b > 0$. Let $a_0 = a$, $a_1 = b$ and $q_1 = \lfloor a_0/a_1 \rfloor$.

$$
\begin{array}{rclcc}
 & & & gcd(a_0, a_1) & \\
a_0 & = & q_1 \times a_1 + a_2 & gcd(a_1, a_2) & q_1 = \lfloor a_0/a_1 \rfloor \\
a_1 & = & q_2 \times a_2 + a_3 & gcd(a_2, a_3) & q_2 = \lfloor a_1/a_2 \rfloor \\
a_2 & = & q_3 \times a_3 + a_4 & gcd(a_3, a_4) & q_3 = \lfloor a_2/a_3 \rfloor \\
 & \vdots & & & \\
a_{t-2} & = & q_{t-1} \times a_{t-1} + a_t & gcd(a_{t-1}, a_t) & q_{t-1} = \lfloor a_{t-2}/a_{t-1} \rfloor \\
a_{t-1} & = & q_t \times a_t + 0 & gcd(a_t, 0) & q_t = \lfloor a_{t-1}/a_t \rfloor
\end{array}
$$

Table: Computation of $gcd(a, b)$

By using the fact on gcd before, we have
gcd(a; b) = gcd(a0; a1) = gcd(a1; a2) = = gcd(at□1; at) = gcd(at ; 0)
Solving for at in the above equations starting from last-but-one to the first, we can express at as a linear combination of a0 and a1.
gcd(a,b) = at = xa + yb:
The following example illustrates the above point. A theorem proving version of the algorithm is given at the end of this set of slides.

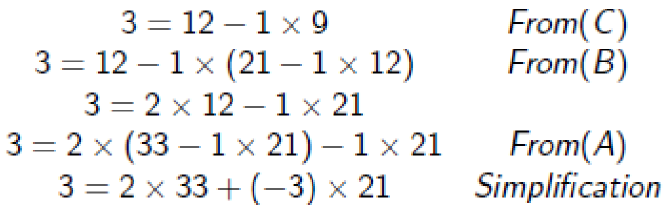
Example



Consider $gcd(33,21)$:

$$\begin{array}{rcccc}
33 & = & 1 \times 21 + 12 & gcd(21,12) & (A) \\
21 & = & 1 \times 12 + 9 & gcd(12,9) & (B) \\
12 & = & 1 \times 9 + 3 & gcd(9,3) & (C) \\
9 & = & 3 \times 3 + 0 & gcd(3,0) &
\end{array}$$

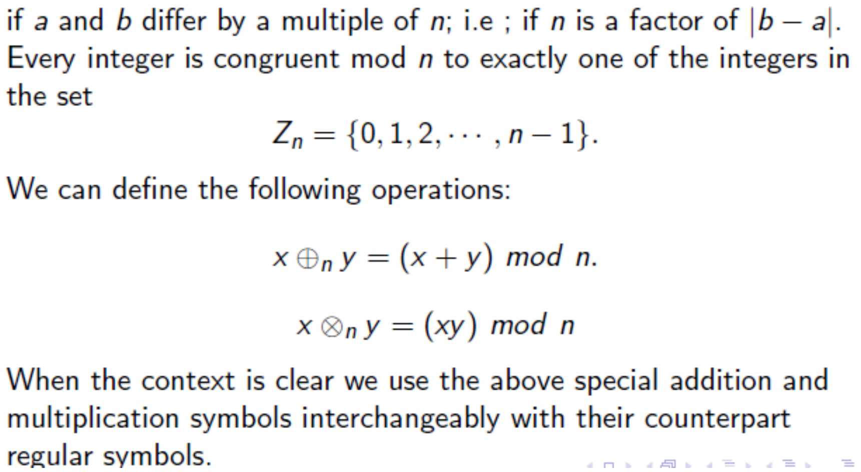
Table: Determine $gcd(33,21)$



$$\begin{array}{cc}
3 = 12 - 1 \times 9 & From(C) \\
3 = 12 - 1 \times (21 - 1 \times 12) & From(B) \\
3 = 2 \times 12 - 1 \times 21 & \\
3 = 2 \times (33 - 1 \times 21) - 1 \times 21 & From(A) \\
3 = 2 \times 33 + (-3) \times 21 & Simplification
\end{array}$$

Modular Arithmetic

Let a and b be integers and let n be a positive integer. We say "a" is congruent to "b", modulo n and write $a \equiv b \ (mod \quad n)$



if a and b differ by a multiple of n; i.e ; if n is a factor of $|b-a|$. Every integer is congruent mod n to exactly one of the integers in the set

$$Z_n = \{0, 1, 2, \cdots, n-1\}.$$

We can define the following operations:

$$x \oplus_n y = (x + y) \ mod \ n.$$

$$x \otimes_n y = (xy) \ mod \ n$$

When the context is clear we use the above special addition and multiplication symbols interchangeably with their counterpart regular symbols.

Modular Multiplicative Inverse

Definition

Let $x \in Z_n$, if there is an integer y such that

$$x \otimes_n y = 1,$$

then we say y is the multiplicative inverse of x. It is denoted by $y = x^{-1}$ usually.

Example: let $n = 5$, 2 is inverse of 3 in Z_5. Or in other words 2 is inverse of 3 modulo 5.

Consider $gcd(13, 25)$:

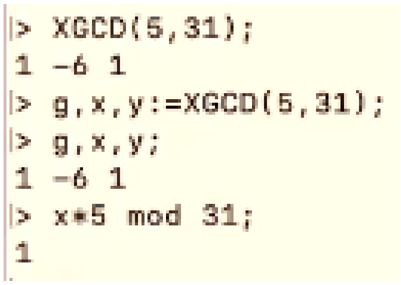
$$\begin{array}{rcccc} 25 & = & 1 \times 13 + 12 & gcd(13, 12) & (A) \\ 13 & = & 1 \times 12 + 1 & gcd(12, 1) & (B) \\ 12 & = & 12 \times 1 + 0 & gcd(1, 0) & \end{array}$$

Table: Determine $gcd(13, 25)$

$$\begin{array}{cc} 1 = 13 - 1 \times 12 & From(B) \\ 1 = 13 - 1 \times (25 - 1 \times 13) & From(A) \\ 1 = 2 \times 13 - 1 \times 25 & \\ 1 = 2 \times 13 + (-1) \times 25 & Simplification \end{array}$$

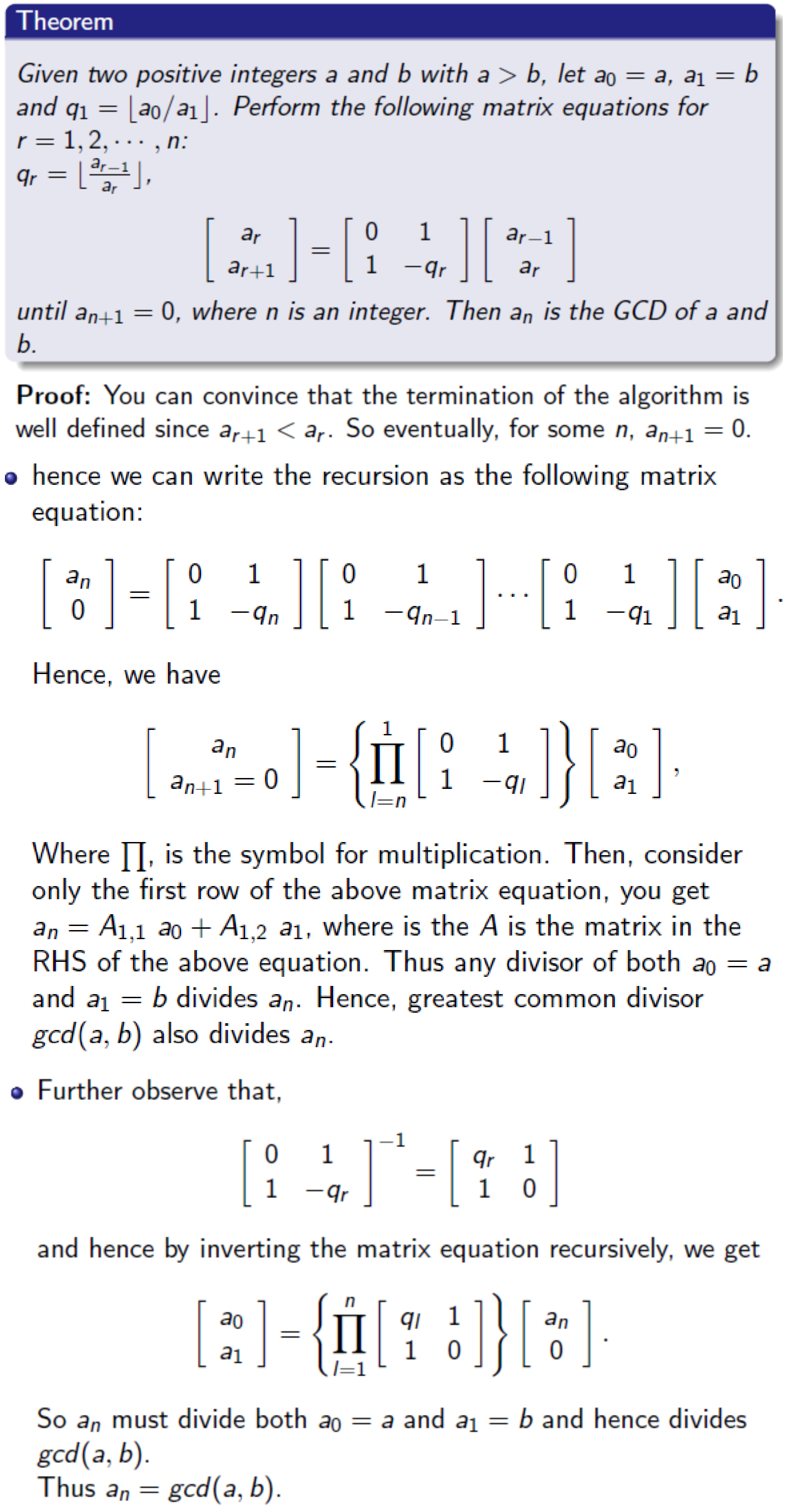
It is easy to see now, 2 is inverse of 13 mod 25.

Magma Example



```
> XGCD(5,31);
1 -6 1
> g,x,y:=XGCD(5,31);
> g,x,y;
1 -6 1
> x*5 mod 31;
1
```

Extended GCD Algorithm: Theorem Proving



Theorem

Given two positive integers a and b with $a > b$, let $a_0 = a$, $a_1 = b$ and $q_1 = \lfloor a_0/a_1 \rfloor$. Perform the following matrix equations for $r = 1, 2, \cdots, n$:
$q_r = \lfloor \frac{a_{r-1}}{a_r} \rfloor$,

$$\begin{bmatrix} a_r \\ a_{r+1} \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ 1 & -q_r \end{bmatrix} \begin{bmatrix} a_{r-1} \\ a_r \end{bmatrix}$$

until $a_{n+1} = 0$, where n is an integer. Then a_n is the GCD of a and b.

Proof: You can convince that the termination of the algorithm is well defined since $a_{r+1} < a_r$. So eventually, for some n, $a_{n+1} = 0$.

- hence we can write the recursion as the following matrix equation:

$$\begin{bmatrix} a_n \\ 0 \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ 1 & -q_n \end{bmatrix} \begin{bmatrix} 0 & 1 \\ 1 & -q_{n-1} \end{bmatrix} \cdots \begin{bmatrix} 0 & 1 \\ 1 & -q_1 \end{bmatrix} \begin{bmatrix} a_0 \\ a_1 \end{bmatrix}.$$

Hence, we have

$$\begin{bmatrix} a_n \\ a_{n+1} = 0 \end{bmatrix} = \left\{ \prod_{l=n}^{1} \begin{bmatrix} 0 & 1 \\ 1 & -q_l \end{bmatrix} \right\} \begin{bmatrix} a_0 \\ a_1 \end{bmatrix},$$

Where $\prod$, is the symbol for multiplication. Then, consider only the first row of the above matrix equation, you get $a_n = A_{1,1}\ a_0 + A_{1,2}\ a_1$, where is the A is the matrix in the RHS of the above equation. Thus any divisor of both $a_0 = a$ and $a_1 = b$ divides a_n. Hence, greatest common divisor $gcd(a, b)$ also divides a_n.

- Further observe that,

$$\begin{bmatrix} 0 & 1 \\ 1 & -q_r \end{bmatrix}^{-1} = \begin{bmatrix} q_r & 1 \\ 1 & 0 \end{bmatrix}$$

and hence by inverting the matrix equation recursively, we get

$$\begin{bmatrix} a_0 \\ a_1 \end{bmatrix} = \left\{ \prod_{l=1}^{n} \begin{bmatrix} q_l & 1 \\ 1 & 0 \end{bmatrix} \right\} \begin{bmatrix} a_n \\ 0 \end{bmatrix}.$$

So a_n must divide both $a_0 = a$ and $a_1 = b$ and hence divides $gcd(a, b)$.
Thus $a_n = gcd(a, b)$.

Symmetric Key Cryptography

Symmetric Cipher Models

Conventional encryption since antiquity-known as single key or private key or **symmetric key systems**.

- A same key is used for both encryption and decryption.
- One of the main assumptions is that both sender and receiver should have access to the symmetric key used in the encryption.
- Widely used in practice.
- Examples: DES, AES etc.

Plaintext

- Source message, 明文

Ciphertext

- Encrypted message, 密文

Cipher

- Also called Encryption Algorithm, 密码
- Procedure for transforming plaintext to ciphertext

Key

- Info or secret used in cipher known only to sender/receiver, 密钥

Encipher

- Converting plaintext to ciphertext, 加密

Decipher

- Recovering plaintext from ciphertext, 解密

Cryptography

- Study of encryption principles/method

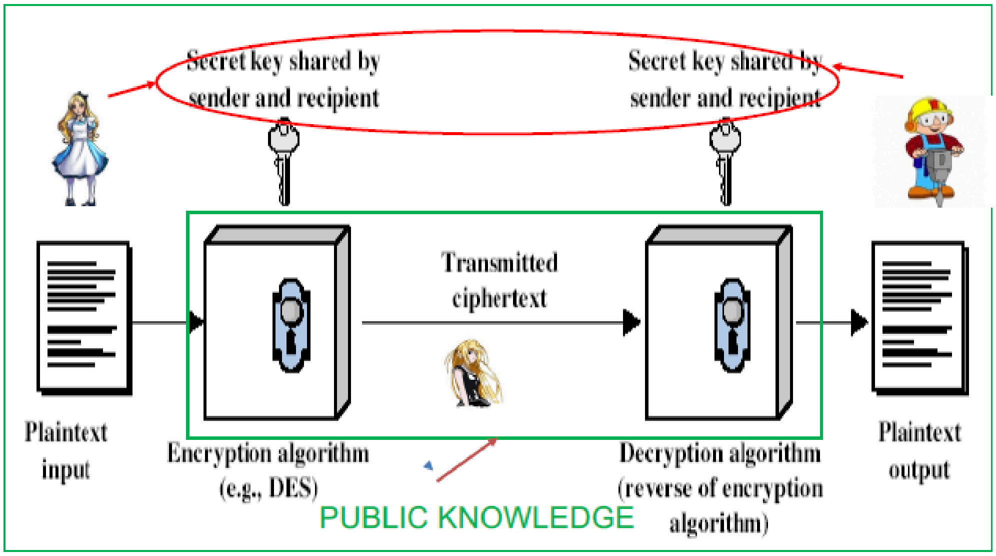
Cryptanalysis

- Also known as codebreaking
- Study of principles/methods of deciphering Ciphertext **without** knowing key

Cryptology

- Field of both cryptography and cryptanalysis

Symmetric Key Cipher Model


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

$$E_{k_1}(m_1) = c_1$$

$$D_{k_1}(c_1) = m_1$$

$$\boxed{\begin{array}{r} D_{k_2}(E_{k_1}(m)) = m \\ \text{iff } k_1 \equiv k_2 \end{array}}$$

Requirements

- Kerckhoffs's principle argues that **security through obscurity is not recommended**.
- All algorithm details are made public and security should be obtained by using **only secrecy of key** used in the encryption
- Stallings recommends two essential requirements:
  - A strong encryption algorithm
  - A secret key known only to participants.
- In symmetric key systems security it is a mandatory requirement that keys are to be kept secret between sender and receiver.

User Perspective

Crypto systems have 3 independent Dimensions according to Stallings.

1. Algorithm for Ciphers
   a. Transformation details of plaintext to cipher text. Based on mathematics, heuristics and pragmatic ideas.
2. Number of Possible Keys used
   a. Higher the key size better protection.
3. Types of Plaintext/Ciphertext processing
   a. Stream ciphers: plaintexts are streamed to cipher producing stream of ciphertexts **element by element**.
   b. Block Ciphers: plaintext is divided into blocks of data, cipher process **one block at a time**

Opponent Perspective

Main task for him to be able to decrypt ciphertexts without access to keys.

- Usually the objective is to obtain keys by observing plaintext/ciphertext pairs called Cryptanalysis.
  - In principle, the opponent can get all information about the cryptosystem except the key involved.
- Keys should **be large enough** such that Brute-force search is impossible.
- There are types of Cryptanalytic attacks based on the capability of the opponent's model.
  - Ciphertext only
  - Known plaintext
  - Chosen plaintext
  - Chosen ciphertext
  - Chosen text

Security

Adversaries are the entities on a communication network.

- They can deploy various services to watch, collect, record and process information that flows at the points that they desire.
- They can use centralized or distributed architecture.

Two important definitions

- Unconditional Security (Shannon)
    - The security of the cipher is independent of the computing resource available to the adversaries.
- Computational Security (Turing)
    - Adversaries are provided with constrained computing resources and the security of the cipher determined by the size of the computations required to break the cipher.

Brute-force Attack

Try all possible messages, but that is generally futile as the space is large.

- Even breaking one ciphertext, one may need to repeat the same steps for every ciphertext. **Not a feasible approach**.

The attack is directly proportional to the size of the key space.

- Generally the size of the key space will tell you the complexity of the Bruteforce key attack.
    - You need at least 128 bit keys to protect against this attack in practice based on the assumption that adversaries are equipped with classical computing resources.
- Need to increase the key size to protect against Quantum computing attacks.

Classical Ciphers

Substitution Ciphers

- Here plaintext symbols are substituted or replaced with other symbols using an unknown key.
- The substitutions can be performed as a sequence of symbols or symbol by symbol.
- Eg
    - RANDOM NODE- DITNAP TANF

Transposition Ciphers

- Here plaintexts are organized as a sequence of plaintext blocks and symbol positions in each block are permuted or transposed using a key. The same permutation is used for every block.
- A permutation of plaintext symbols are employed here as opposed to substitution.

- As a result letters are only rearranged for every d position.
- Divide plain text as a sequence of plaintext blocks of certain size, say, d.
- Then apply a permutation to every d positions of the plaintext. The permutation is the key.
- Eg
    - RANDOM LETTER -> MORADN RELETT

Caesar Cipher

Historically attributed to Julius Caesar.
Assumes letter ordering in a language.
For example, in English

- The alphabet order is: ABCDEFGHIJKLMNOPQRSTUVWXYZ
- Consider plaintext in sequence, each letter is replaced with the letter that
- stands in a certain secret(key) places further in the alphabet.

In math

- P:= Plain Text Space = Z26 Space
- C:=Cipher Text Space = Z26
- p: = plaintext c:=ciphertext
- Encryption: E(k,p) = c = p + k mod 26
- Decryption: D(k,p) = c – k mod 26

Cryptanalysis

- There are only 26 possible keys, In fact, only 25 non-trivial keys.
- Could mount a simple Brute-force attack.
- Try applying shifts on the alphabets in the ciphertext from 1 to 25, when recognizing some meaningful plaintext stop.

Affine Cipher

- P:= Plain Text Space = Z26 Space
- C:=Cipher Text Space = Z26
- Key space = (Z26, Z26),
- Key k = (a,b, a, b belongs to Z26
- p: = plaintext c:=ciphertext
- Encryption: E(k,p) = c = ap + b mod 26
- Decryption: D(k,p) = Inverse(a)(c – b) mod 26

Cryptanalysis

- There are only 26 possible keys.
-

Monalphabetic Cipher

A more general key using a general permutation on 26 alphabets.

A key is a permutation on the alphabet Z26 (plaintext letter maps to a different random ciphertext letter)

- An example: key could be a permutation:
    - ABCDEFGHIJKLMNOPQRSTUVWXYZ
    - DKVQFIBJWPESCXHTMYAUOLRGZN

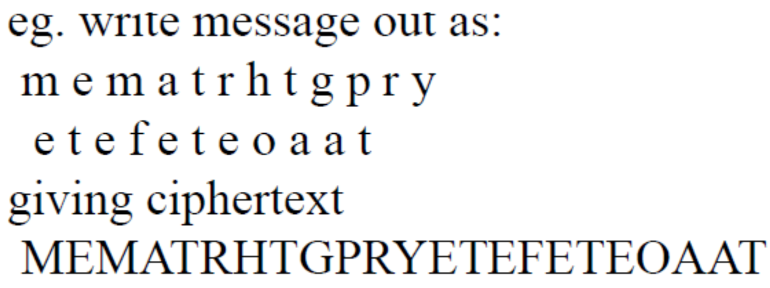
Cryptanalysis

- There are 26 different keys, brute-force impossible.
- Use language statistics.
    - In English some letters appear more frequently than others, for example "e" appears more frequently followed by "t" etc, A monoalphabetic cipher always maps a distinct alphabet to another symbol. **The mapping preserves the language statistics**. With this one can start guessing which is "e" which is "t" etc.

Rail Fence Cipher

A simpler technique where message is written out diagonally over a depth of certain rows (say d). Then ciphertext is read row by row.

- Such ciphers are easy to break if **depth is small**.
- Example



eg. write message out as:

```
m e m a t r h t g p r y
 e t e f e t e o a a t
```

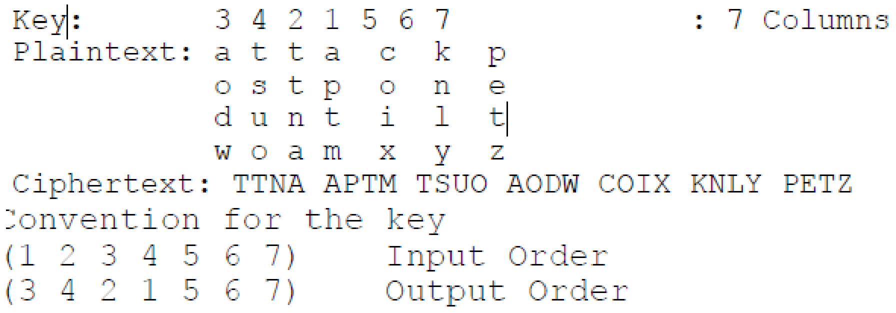
giving ciphertext

- MEMATRHTGPRYETEFETEOAAT

Row Transportation Cipher

A more complex Transposition Cipher is by employing a permutation on blocks of columns, when messages are written row by row.

- Example:
  - Write letters of message out in rows over a specified number of columns
  - Then reorder the columns according to some key before reading off the rows



```
Key:        3 4 2 1 5 6 7            : 7 Columns
Plaintext:  a t t a  c  k  p
            o s t p  o  n  e
            d u n t  i  l  t
            w o a m  x  y  z
Ciphertext: TTNA APTM TSUO AODW COIX KNLY PETZ
Convention for the key
(1 2 3 4 5 6 7)     Input Order
(3 4 2 1 5 6 7)     Output Order
```

  -

Class 4

Complex Ciphers

Polyalphabetic Cipher

Use a set of monoalphabetic ciphers at different times when processing plaintext sequences to make the encryption process more complex so that it is difficult to break.

- A key could be used to specify which monoalphabetic cipher to use in a given time context.

Vigenere Cipher

This is a simple polyalphabetic substitution cipher.

- Here a set of Caesar ciphers is employed.
  - ith plaintext symbol is handled by Caesar cipher with key: k(i mod d)
- The idea is very simple, a key is a multiple letter word: K = k1 k2 ... kd
  - P = p1 p2 ... pd pd+1 pd+2 ... p2d….
  - C = c1 c2 ... cd cd+1 cd+2 ... c2d….
  - Encryption: E(K,P) = C, where ci = pi + ki mod 26
  - Decrypton: D(K,C) = P, where pi = ci - ki mod 26
- If d is large it offers better security.

Example

- key: deceptivedeceptivedeceptive

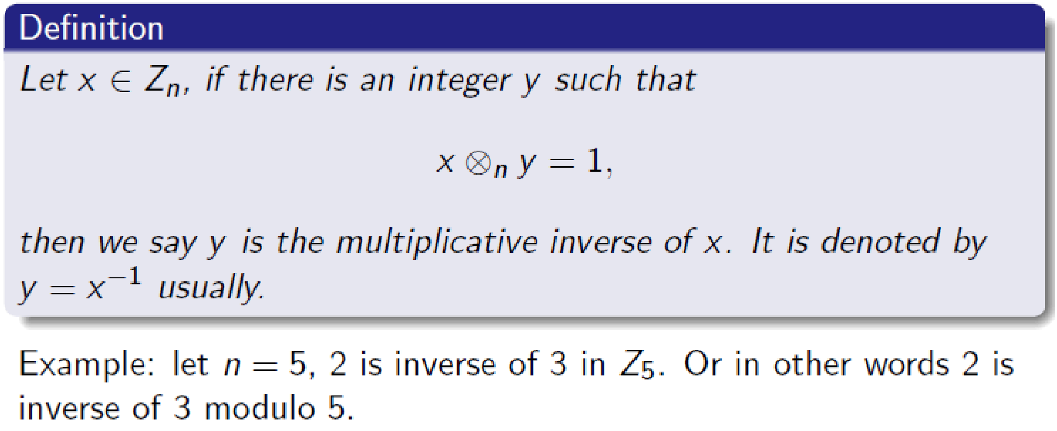
- plaintext: wearediscoveredsaveyourself
- ciphertext: ZICVTWQNGRZGVTWAVZHCQYGLMGJ

Product Cipher

Substitution and Transposition Ciphers are not secure as they are vulnerable to cryptanalysis based on plaintext language characteristics.

- More general product cipher by applying several substitution and transposition ciphers in succession.

Inverse Modulo n



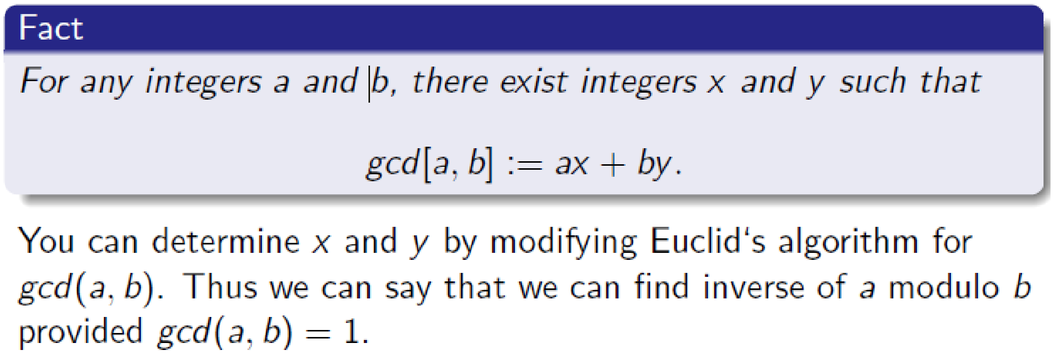
Definition

Let $x \in Z_n$, if there is an integer y such that

$$x \otimes_n y = 1,$$

then we say y is the multiplicative inverse of x. It is denoted by $y = x^{-1}$ usually.

Example: let $n = 5$, 2 is inverse of 3 in Z_5. Or in other words 2 is inverse of 3 modulo 5.



Fact

For any integers a and b, there exist integers x and y such that

$$gcd[a, b] := ax + by.$$

You can determine x and y by modifying Euclid's algorithm for $gcd(a, b)$. Thus we can say that we can find inverse of a modulo b provided $gcd(a, b) = 1$.

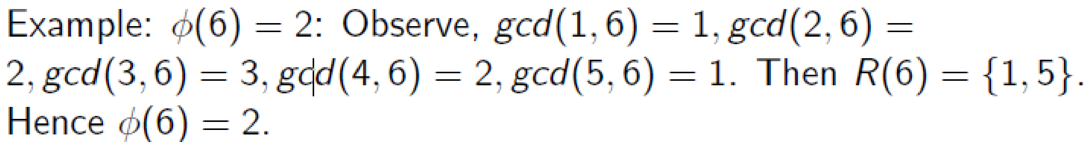
Euler's Phi Function

Two numbers a and b are relatively prime if gcd(a; b) is 1.

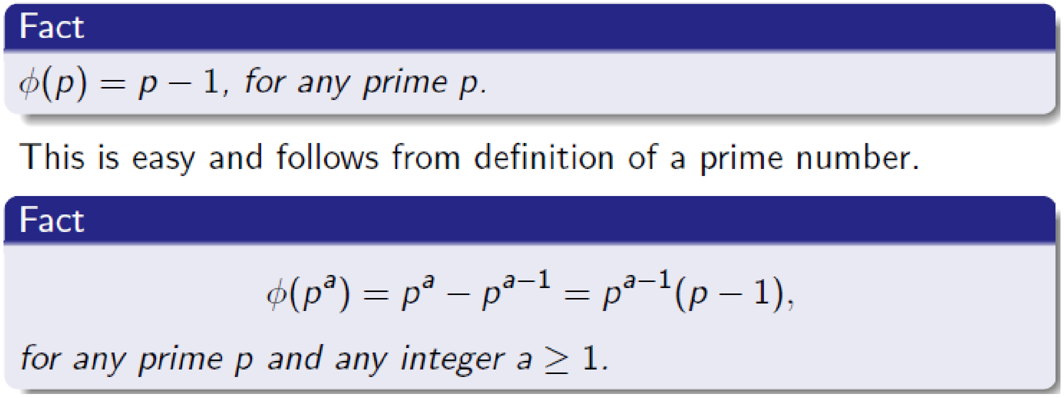
Euler phi function(or Euler totient function): For n 1, let ø(n) denote the number of integers less than n but are relatively prime to n.

Reduced set of residues mod n: For n 1, the reduced set of residues, R(n) is defined as a set of residues modulo n which are relatively prime to n.

Example



Example: $\phi(6) = 2$: Observe, $gcd(1,6) = 1, gcd(2,6) = 2, gcd(3,6) = 3, gcd(4,6) = 2, gcd(5,6) = 1$. Then $R(6) = \{1,5\}$. Hence $\phi(6) = 2$.



Fact

$\phi(p) = p - 1$, for any prime p.

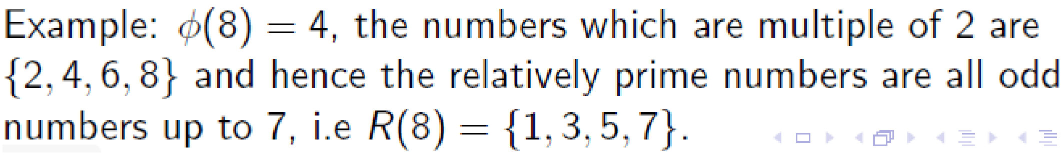
This is easy and follows from definition of a prime number.

Fact

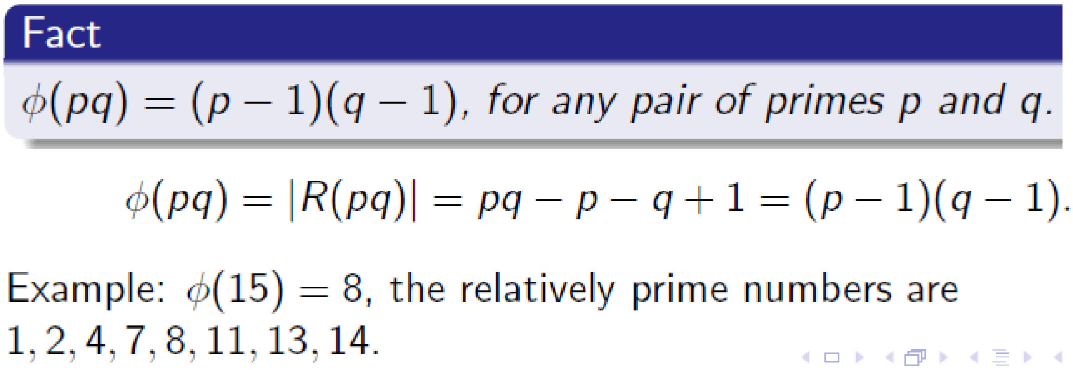
$$\phi(p^a) = p^a - p^{a-1} = p^{a-1}(p-1),$$

for any prime p and any integer $a \geq 1$.

Example



Example: $\phi(8) = 4$, the numbers which are multiple of 2 are $\{2,4,6,8\}$ and hence the relatively prime numbers are all odd numbers up to 7, i.e $R(8) = \{1,3,5,7\}$.

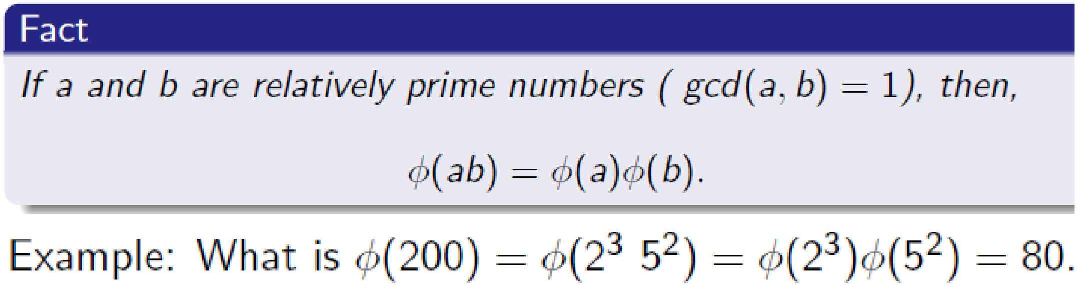


Fact

$\phi(pq) = (p-1)(q-1)$, for any pair of primes p and q.

$$\phi(pq) = |R(pq)| = pq - p - q + 1 = (p-1)(q-1).$$

Example: $\phi(15) = 8$, the relatively prime numbers are $1,2,4,7,8,11,13,14$.



Fact

If a and b are relatively prime numbers ($gcd(a,b) = 1$), then,

$$\phi(ab) = \phi(a)\phi(b).$$

Example: What is $\phi(200) = \phi(2^3\ 5^2) = \phi(2^3)\phi(5^2) = 80$.

[2^(3-1)*(2-1)]*[5^(2-1)*(5-1)] = 4*[5*4] = 80

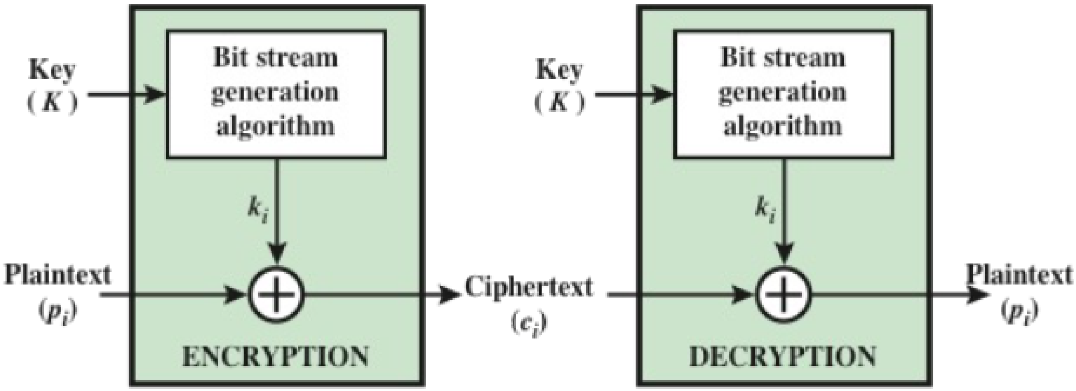
Class 5

Modern Symmetric Key Cipher

Stream Cipher

Takes a key of fixed size and generates a key stream in a pseudo random fashion with a large period; this key stream is then combined with the plain text message stream on a bit by bit basis to form a cipher text stream.

- RC4, A5, BlueTooth cipher etc.
- Extensively employed in modern communication networks.
    - Top choice in LightWeight Cryptographic applications.
        - eSTREAM
            - ECRYPT Stream Cipher Project: An European stream cipher project in the last decade gave impetus to the development of the subject.
- Unit of stream operation can be "**bit by bit**" or "byte by byte" or "symbol by symbol", it encrypts one unit of plain text stream at a time. Useful for processing stream-based data such as voice, connection oriented traffic etc.



Apply One-Time Pad

An idea is to generate a long stream based on a short key and use it as a keystream in the One-Time pad scheme. The resulting cipher is "stream Cipher".

- In general it takes a key and a random nonce (Initial Vector(IV))as input and outputs a keystream of arbitrary size. The keystream is then xored with the plaintext to obtain a ciphertext.

Vernam Cipher

a simple polyalphabetic substitution cipher, also called One-Time-Pad.

- Encryption: E(K,P) = C, where ci = pi + ki mod 26

- Decrypton: D(K,C) = P, where pi = ci - ki mod 26
- Extend the size of the key to be equal to the message (d = n). The resulting cipher is Vernam.
- The scheme can be defined over any alphabet (mod m).

Probability Basics

Conditional Probability

P(A | B) = P(A ∩ B) / P(B)

Perfect Secrecy

An encryption scheme has the property of unconditional security if the cipher text generated by the algorithm does not reveal sufficient information to break the scheme, even with access to an unlimited amount of computational power.

- Given a cipher without a key, the scheme can not be broken.

Let x: input, y: output

Perfect Security implies: PX|Y(x|y) = PX(x)

One-Time Pad Definition

Defined over binary messages, it offers perfect secrecy, but not practical because It demands a key as long as the message.

- An extension of Vernam Cipher for binary messages.
- Here the key is as long as the message.
- For each message you need a distinct random key.
- Encryption and Decryption operation are exactly same, XOR with the key

Let ⊕ denote exclusive or symbol. Let [0,1] be binary alphabet.

- 0 ⊕ 0 = 1 ⊕ 1 = 0;
- 0 ⊕ 1 = 1 ⊕ 0 = 1.

If A, B, C are vectors, is point-wise vector XOR then

- A ⊕ B = C then
    - B = A ⊕ C; A ⊕ A = 0; B ⊕ B =0, C+C= 0

Suppose Alice wishes to send a message M = 0110111 to Bob and they have previously established a shared secret key:K =1011011.

Encryption:

- C = M ⊕ K = 1101100

Decryption is trivial: the message could be obtained by the same process.

- $M = C \oplus K = 0110111$.

Perfect Secrecy Property Prove

Assume that the message space is binary (0 or 1) and key space is also binary. Assume that A chooses a message 0 quarter of the time, i.e Probability that the message is 0 is equal to 1/4, $P(M=0) = 1/4$.

- Perfect secrecy means knowing this fact, any adversary (E) should not get more information by observing the cipher message ($C = M \oplus K$).
    - i.e. The conditional probability, $P(M = 0 \mid C =1)$ should not be different from a priori probability $P(M=0)$.

Implication

Requirement: Encryption transformation should distribute messages to cipher space fairly uniformly irrespective of known a priori statistics of the messages.

- One-time pad analysis tells that if we choose a random secret key pad at least the size of the message, we can achieve the perfect secrecy.
    - Basically, the random key, as long as the message, hides the message completely leading to the perfect "confusion" to the adversary.
        - By perfectly "diffusing" the statistical structure of the plain text to the entire ciphertext.

Two-time pad

$C1=M1 \oplus K$; $C2= M2 \oplus K$; then

- $C1 \oplus C2=M1 \oplus M2 \oplus K \oplus K= M1 \oplus M2$.

Even though $M1 \oplus M2$ may not direct meaning, it still leaks information about both M1 and M2.

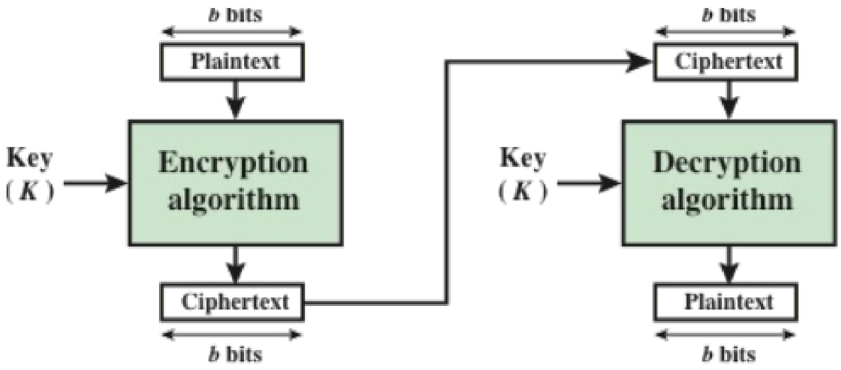
- if one of the messages M1 or M2 is available to the adversary, then he/she can get the other.
- The idea is used in attacking Vegenere cipher (same key-pad is added many times)

To prevent this attack, users need a new key for every message.

Block Cipher

A block cipher takes a fixed length plain text message block (for example, 64 or 128 bits) and a key, and produces a ciphertext block of the same length as the original message.

- DES (56), Triple-DES (168), IDEA (128), Blowfish() and AES (128)
- Unit of operation is always **a block of information**, generally n-bit blocks. Useful in many situations of data traffic.
- Same key is used for many different message blocks.
    - Examples include hash functions, pseudorandom generators, message authentication codes etc.



Confusion and diffusion principles

Diffusion

- dissipates statistical structure of plaintext over bulk of ciphertext.
- created by permutations.

Confusion

- makes the relationship between ciphertext and key as complex as possible.
- created by substitution.

Product Cipher

Combines two or more transformations so that the resulting cipher is more secure than the individual components by making use of confusion and diffusion principles.

substitution-permutation cipher

A product cipher made up of a number of stages each involving substitution and permutation. The operations of substitution and permutation are responsible for effecting the confusion and diffusion respectively.

Iterated block cipher

A block cipher involving sequential repetition of an iterated function called a round function.

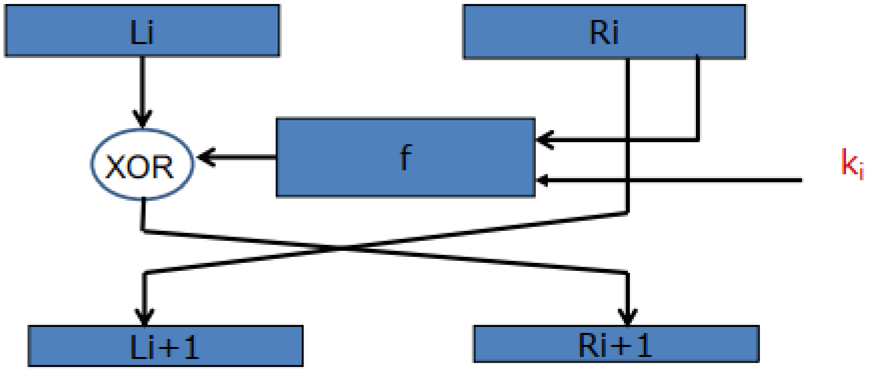
- Parameters
    - r: number of rounds;
    - n: block length;
    - k: bit-size of key, K from which r subkeys (round keys) ki's are derived.

Feistel Block Cipher

an example of an iterated block cipher. Repeat a given operation several times in rounds.

Each round will have the following distinct operations, round operations are repeated certain number of rounds:

1. Substitution
    - Each plaintext bit or group of bits in a block is replaced with a corresponding ciphertext bit or group of bits.
2. Permutation
    - A certain permutation is effected to each transformed ciphertext bit.

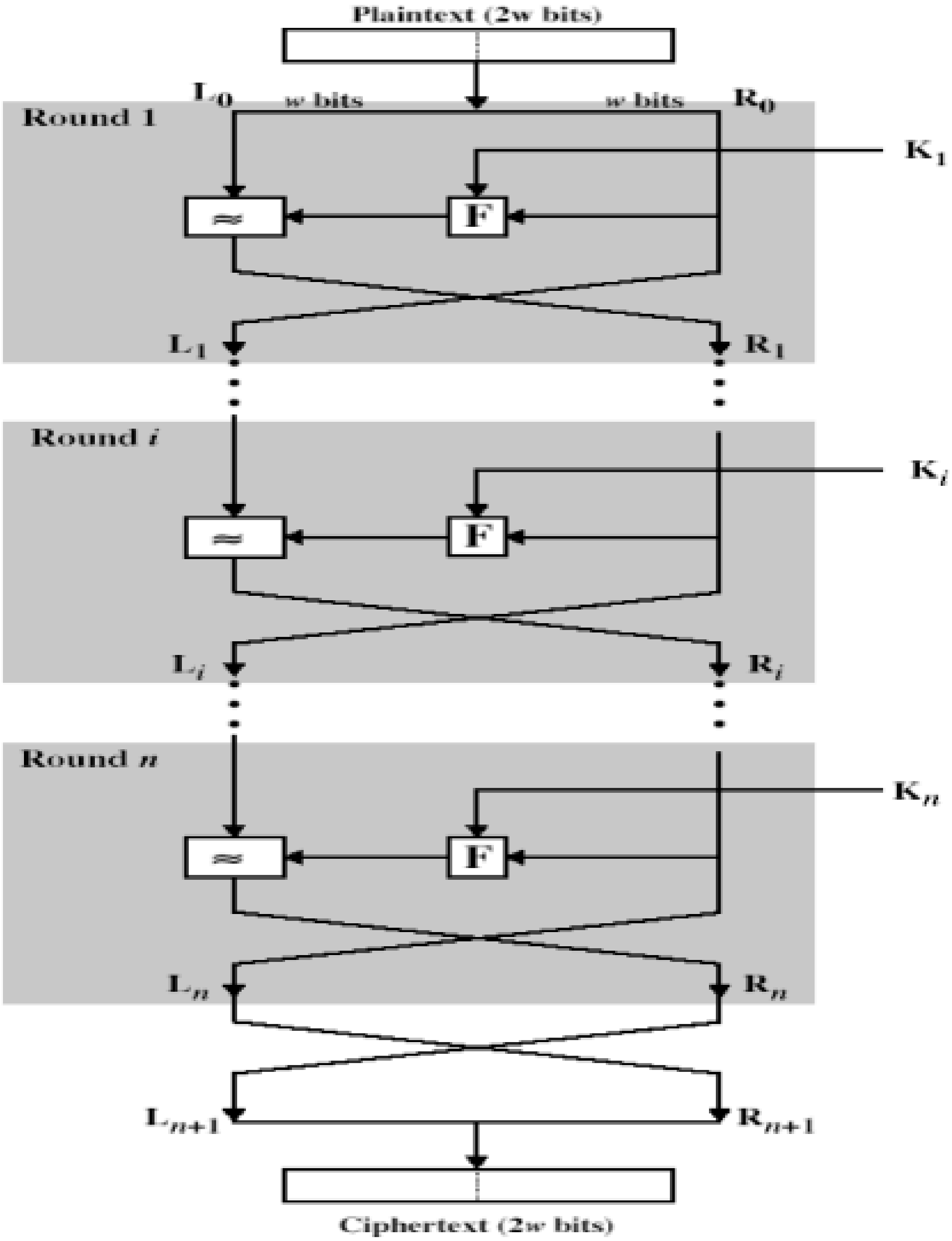


Data Encryption Standard(DES) For Feistel Cipher

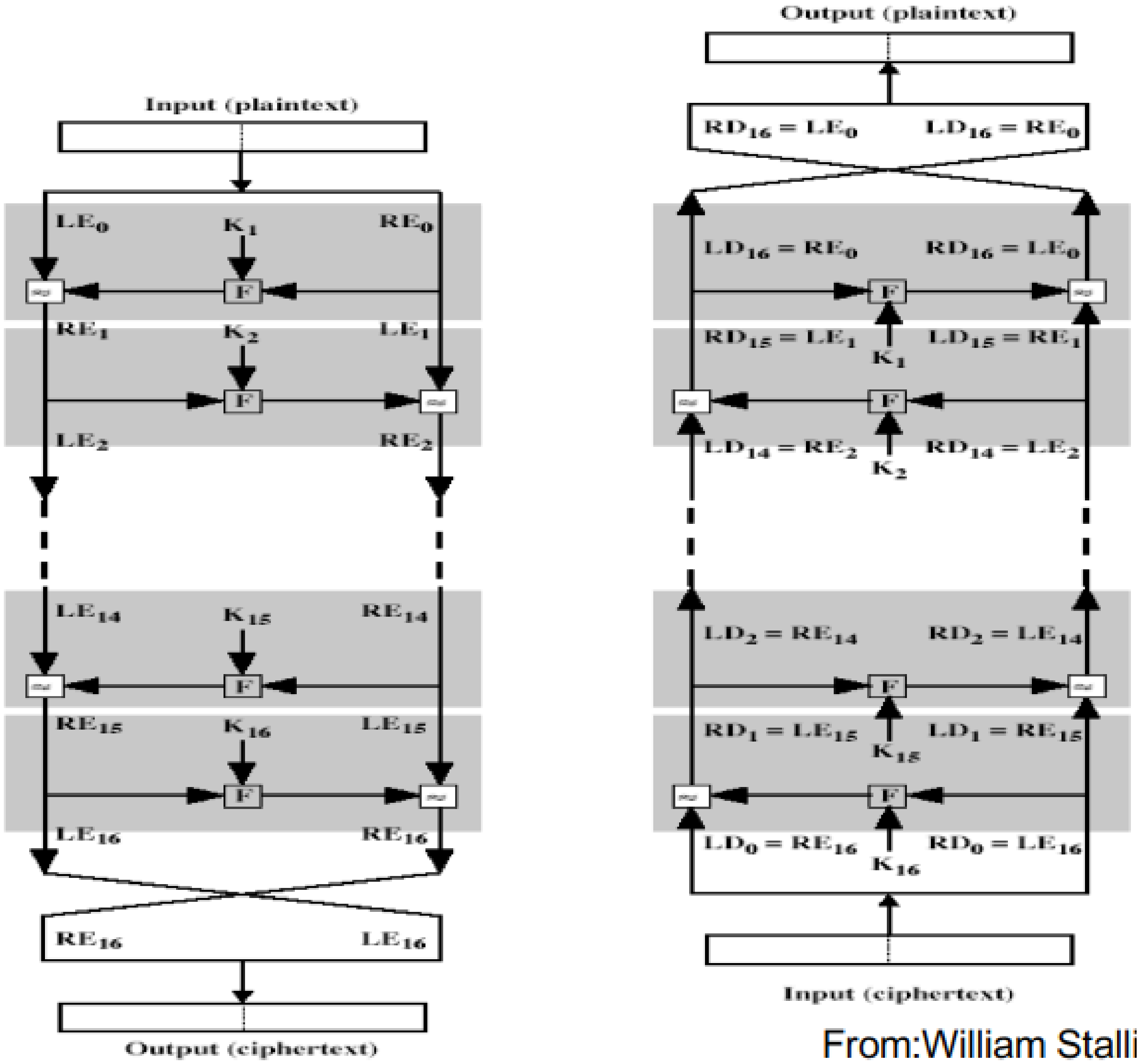
1. Block size: n = 64
2. keysize = k = 56 bits
3. Number of rounds = 16.
4. Strengthening DES:
    a. DESX: Apart from 56 bit key K, choose two new 64 bit keys K_I and K_O, then do encryption

  i. C = K_O ⊕ DES(K, M ⊕ K_I)

b. This method increases the effective key length to 199-t.
  i. t is a quantity related to adversaries' cryptanalytic assumptions that adversaries are able to collect 2t matching input-output pairs.



Encryption & Decryption



From:William Stalli

- Each bit of cipher text depends on all bits of the key and all bits of the plain text.
- No statistical relationship between plain and cipher visible.
- Altering a key bit or a plain text bit should alter each cipher bit with probability close to half.
  - Small changes on key would effect a large change in ciphertext.
- Altering a cipher bit should result in unpredictable change in plain text block.
- Chosen Plaintext attack is possible.
- DES is not recommended as it has small key space and has known theoretical attacks.

  - Financial Systems still use a modification of DES such as Triple-DES, the significant drawbacks include slow and small block size.

Advanced Encryption Standard(AES)

Not a Feistel cipher, but still iterative, key space >= 2^128

Main design requirements:

1. Withstand all known attacks
2. Flexible implementation, to be able to run on varieties of platforms and CPUs.
3. Simple design features

Block Cipher Modes

NIST defined five basic modes of usage of block cipher. Used to enhancing the effect of a cryptographic algorithm or adapting the algorithm for an application.

- The five modes are intended to cover a wide variety of applications of encryption for which a block cipher could be used
- These modes are intended for use with any symmetric block cipher, including triple DES and AES.

ECB

Electronic Codebook Mode

- Uses both Encrypt and Decrypt functions
- Each block of plaintext bits is encoded independently using the same key.

typical application

- Secure transmission of single values(eg. Encryption key)

Superior criteria and properties

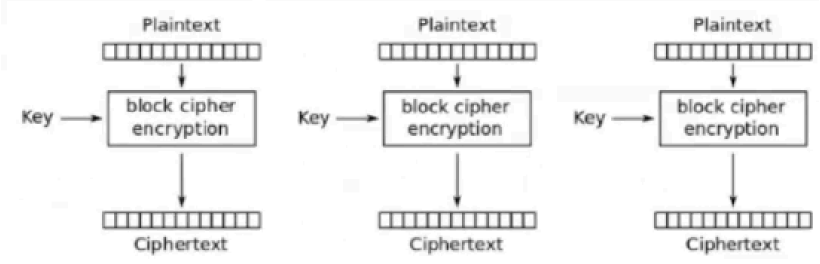
1. Overhead
2. Error recovery
3. Error propagation
4. Diffusion
5. Security

加密: C1 = E(P1,k)

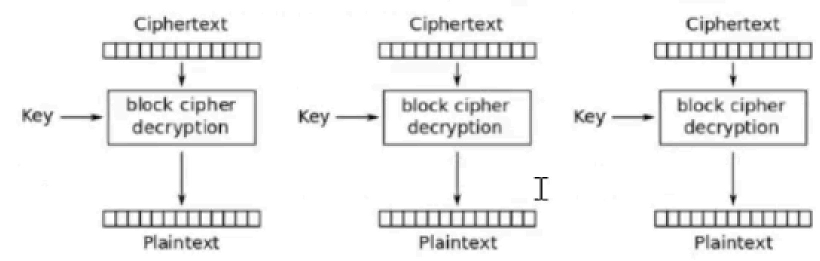
解密: P1 = D(C1,k)

密文的一个比特错误导致一个明文分组错误,但不会影响其他明文分组

加解密可以并行
相同明文被加密为相同密文



Electronic Codebook (ECB) mode encryption



Electronic Codebook (ECB) mode decryption

CBC

Cipher Block Chaining Mode

- Uses both Encrypt and Decrypt functions
- The input to the encryption algorithm is the XOR of the next block of plaintext and the preceding block of ciphertext.

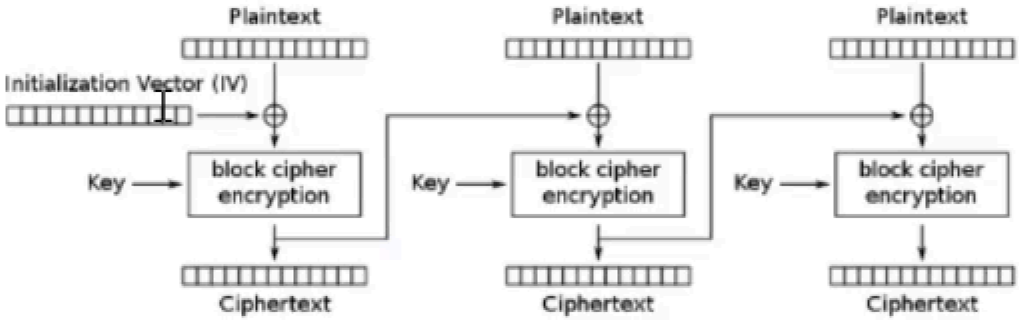
typical application

- General-purpose block-oriented transmission
- Authentication

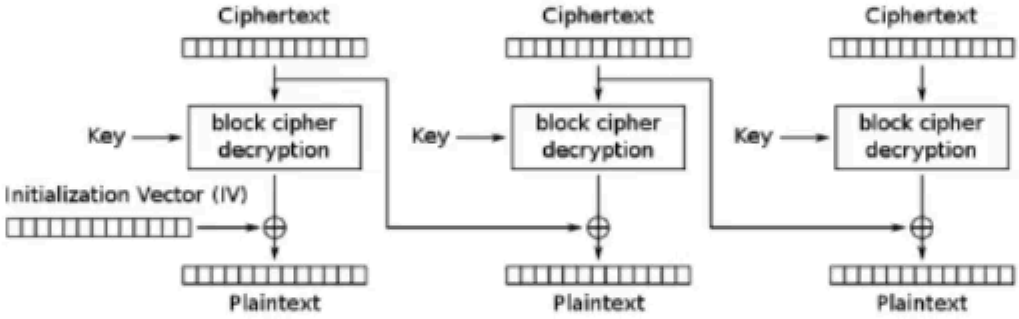
借用上一个分组的加密结果获取不同明文
C[i] = E((P[i] xor C[i-1]),k)
P[i] = D(C[i], k) xor C[i-1]
IV:

1. 充当C[0]
2. 避免第一个分组每次都一样

密文的一个比特错误导致下一个明文分组的一个比特错误,后续分组不受影响
可以并行加密,不能并行解密



Cipher Block Chaining (CBC) mode encryption



Cipher Block Chaining (CBC) mode decryption

CFB

Cipher Feedback Mode

- Uses only Encrypt functions
- Input is processed s bits at a time. Preceding ciphertext is used as input to the encryption algorithm to produce pseudo random output, which is XORed with plaintext to produce the next unit of ciphertext.
- Possible to convert a block cipher into a stream cipher

typical application

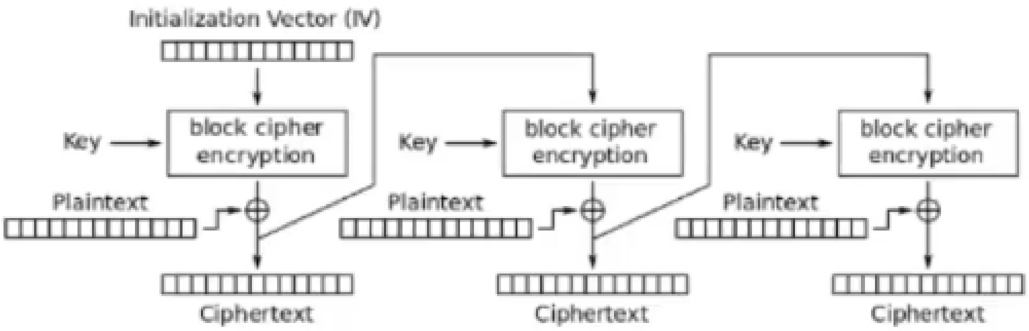
- General-purpose stream-oriented transmission
- Authentication

C[i] = P[i] xor E(C[i-1]),k)

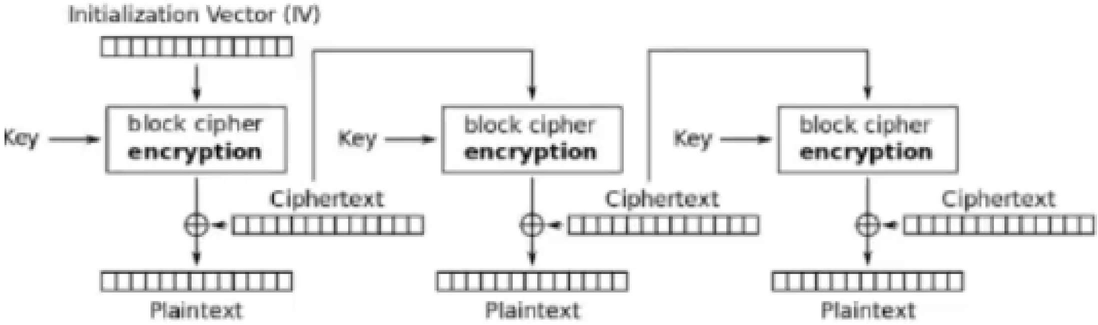
P[i] = C[i] xor E(C[i-1],k)

密文的一个比特错误对应明文分组的一个比特错误,下一个明文分组错误,后续分组不受影响

不能并行加密,可以并行解密.



Cipher Feedback (CFB) mode encryption



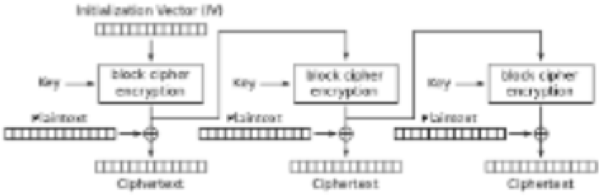
OFB

Output Feedback Mode

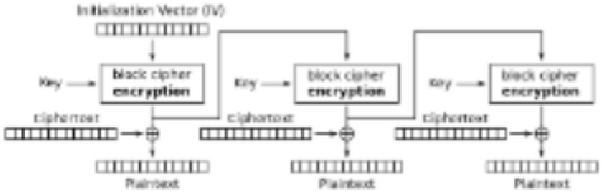
- Uses only Encrypt functions
- Used like a stream cipher
- Similar to CFB, except the input to the encryption algorithm is the preceding encryption output, and full blocks are used.
- Possible to convert a block cipher into a stream cipher

typical application

- Stream-oriented transmission over noisy channel(eg. Satellite communication)



Output Feedback (OFB) mode encryption



Output Feedback (OFB) mode decryption

C[i] = P[i] xor E^i(IV,k) # E^i stands for do encryption i times
P[i] = C[i] xor E^i(IV,k)
密文的一个比特错误对应明文分组的一个比特错误, 不影响其他分组
加密解密均不可并行,但密钥流产生和明文无关,加密方可以预先计算
IV不能重复,重复导致产生完全相同的密钥流,有密码复用的问题

CTR

Counter Mode

- Uses only Encrypt functions
- Used like a stream cipher
- Each block of plaintext is XORed with an encrypted counter. The counter is incremented for each subsequent block.
- Possible to convert a block cipher into a stream cipher

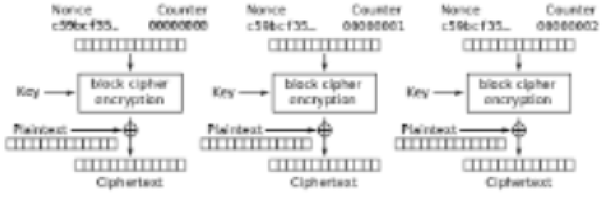
typical application

- General-purpose block-oriented transmission
- Userful for high-speed requirements

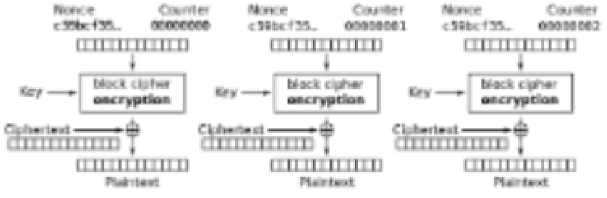
Advantages

1. Hardware efficiency
2. Software efficiency
3. Preprocessing
4. Random access
5. Provable security
6. Simplicity

C[i] = P[i] xor E(Counter[i],k)
P[i] = C[i] xor E[Counter[i],k]
密文的一个比特错误对应明文分组的一个比特错误
加解密均可以并行
IV不能重复,否则密钥流重复,密码复用问题.



Counter (CTR) mode encryption



Counter (CTR) mode decryption

Meet-In-The-Middle Attack

The use of double DES results in a mapping that is not equivalent to a single DES encryption, as a result

- The meet-in-the-middle attack algorithm will attack this scheme and does not depend on any particular property of DES but will work against any block encryption cipher.

Triple-DES with Two-Keys

To counter the meet-in-the-middle attack, use three stages of encryption with three different keys.

- This raises the cost of the meet-in-the-middle attack to 2112, which is beyond what is practical.
- Requiring a key length of 56 x 3 = 168 bits, which may be somewhat unwieldy.
- As an alternative Tuchman proposed a triple encryption method uses only two keys.

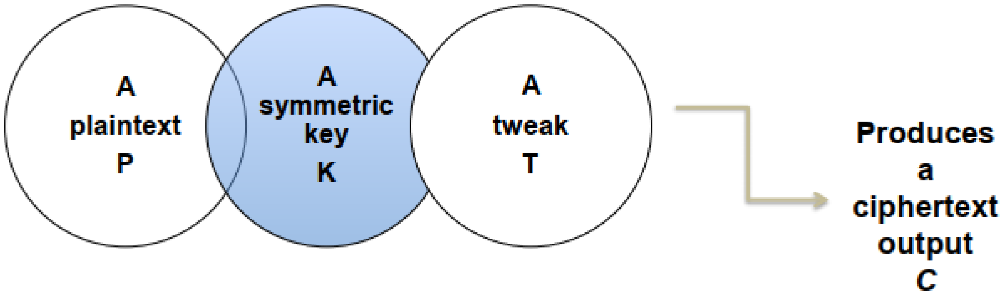
XTS-AES mode

based on the concept of a tweakable block cipher.

- Since the purpose is to provide variability, Tweak do not need to keep secret.
- a method of encryption for data stored in sector-based devices where the threat model includes possible access to stored data by the adversary

General structure

- Has three inputs:



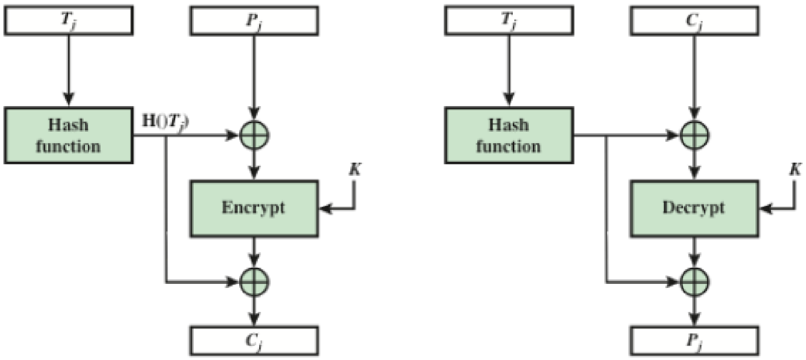
A plaintext P

A symmetric key K

A tweak T

Produces a ciphertext output C

Tweakable Block Cipher



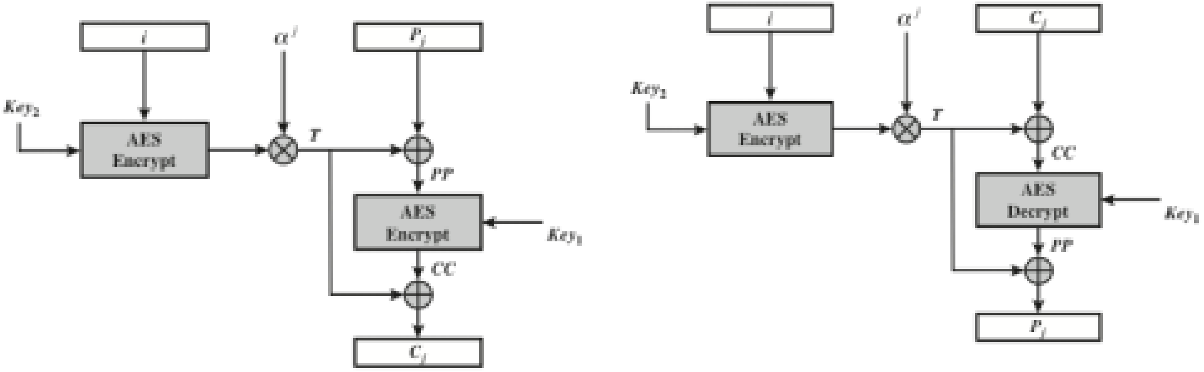
(a) Encryption (a) Decryption

Storage Encryption Requirements

also referred to as “data at rest”, differ from requirements for transmitted data. The P1619 standard characteristics:

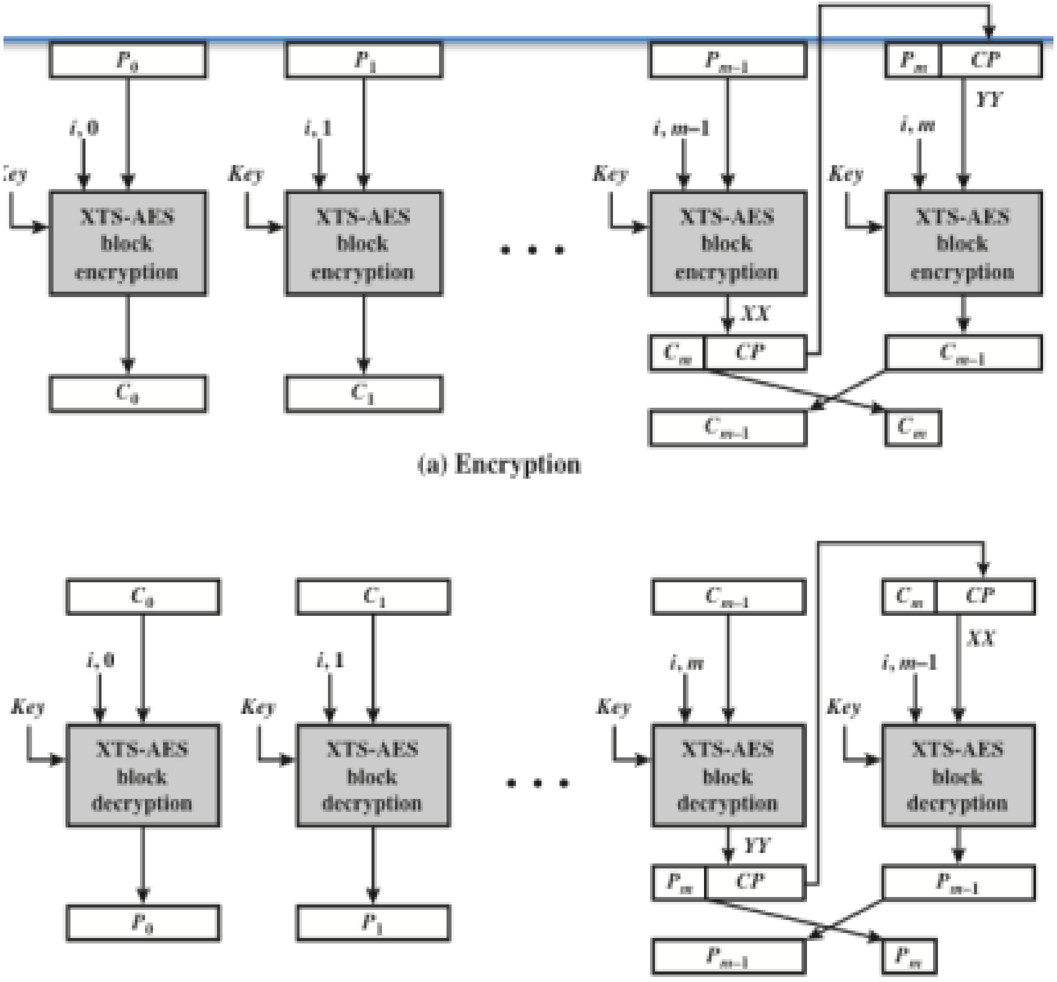
1. The ciphertext is freely available for an attacker
2. The data layout is not changed on the storage medium and in transit
3. Data are accessed in fixed sized blocks, independently from each other
4. Encryption is performed in 16-byte blocks, independently from each other
5. There are no other metadata used, except the location of the data blocks within the whole data set
6. The same plaintext is encrypted to different ciphertexts at different locations, but always to the same ciphertext when written to the same location again.
7. A standard conformant device can be constructed for decryption of data encrypted by another standard conformant device

XTS-AES Operation on Single Block



(a) Encryption (b) Decryption

XTS-AES Mode



(a) Encryption

(b) Decryption

Format-Preserving Encryption(FPE)

Any encryption technique that takes a plaintext in a given format and produces a ciphertext in the same format.

The NIST, and the other FPE algorithms that have been proposed, are used with plaintext consisting of a string of elements, called characters.

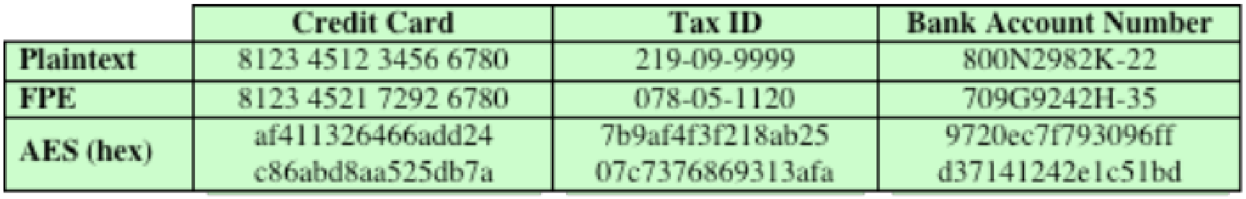
- A finite set of two or more symbols is called an alphabet
- The elements of an alphabet are called characters
- A character string is a finite sequence of characters from an alphabet
- Individual characters may repeat in the string
- The number of different characters in an alphabet is called the base (also referred to as the radix) of the alphabet

Example

credit cards consist of 16 decimal digits.

- An FPE that can accept this type of input would produce a ciphertext output of 16 decimal digits.
    - Note that the ciphertext need not be, and in fact in unlikely to be, a valid credit card number.
- It will have the same format and can be stored in the same way as credit card number plaintext.

Comparison of FPE and AES



	Credit Card	Tax ID	Bank Account Number
Plaintext	8123 4512 3456 6780	219-09-9999	800N2982K-22
FPE	8123 4521 7292 6780	078-05-1120	709G9242H-35
AES (hex)	af411326466add24 c86abd8aa525db7a	7b9af4f3f218ab25 07c7376869313afa	9720ec7f793096ff d37141242e1c51bd

FPE Motivation

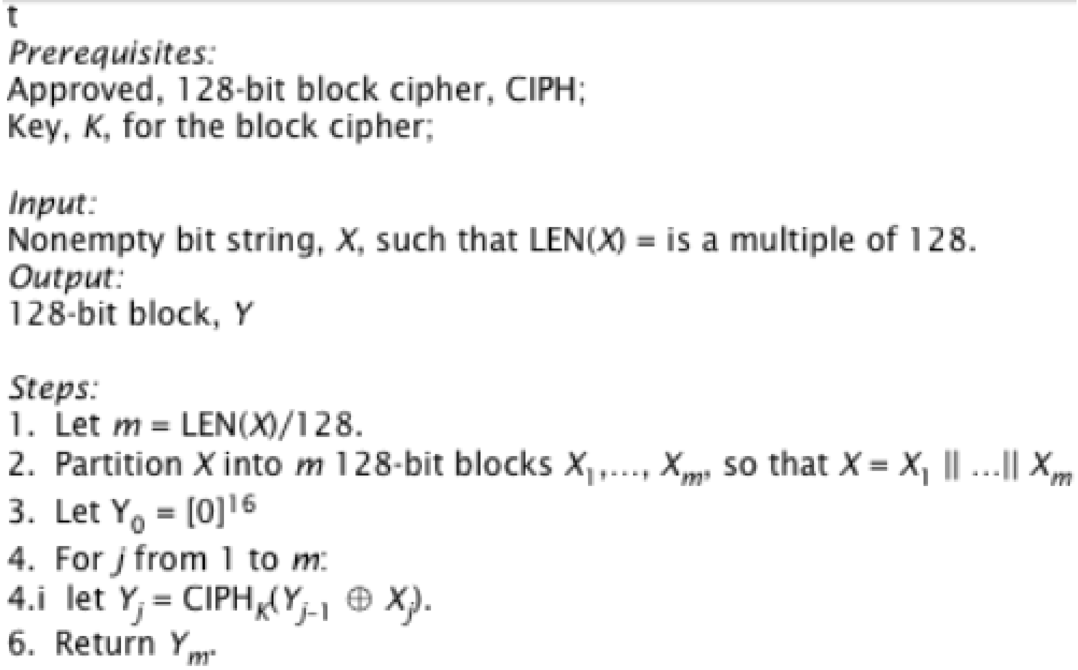
1. FPE facilitates the retrofitting of encryption technology to legacy applications, where a conventional encryption mode might not be feasible because it would disrupt data fields/pathways.
2. FPE has emerged as a useful cryptographic tool.
    a. whose applications include financial-information security, data sanitization, and transparent encryption of fields in legacy databases
3. principal benefit of FPE
    a. enables protection of particular data elements, while still enabling workflows that were in place before FPE was in use
    b. No database schema changes and minimal application changes are required
    c. Only applications that need to see the plaintext of a data element need to be modified and generally these modifications will be minimal
4. Examples of legacy applications where FPE is desirable are:
    a. COBOL data-processing applications
    b. Database applications
    c. FPE-encrypted characters can be significantly compressed for efficient transmission

FPE Requirement in Design

1. The ciphertext is of the same length and format as the plaintext
2. It should be adaptable to work with a variety of character and number types
3. It should work with variable plaintext length
4. Security strength should be comparable to that achieved with AES
5. Security should be strong even for very small plaintext lengths

FPE Algorithms

Algorithm PRF(X)

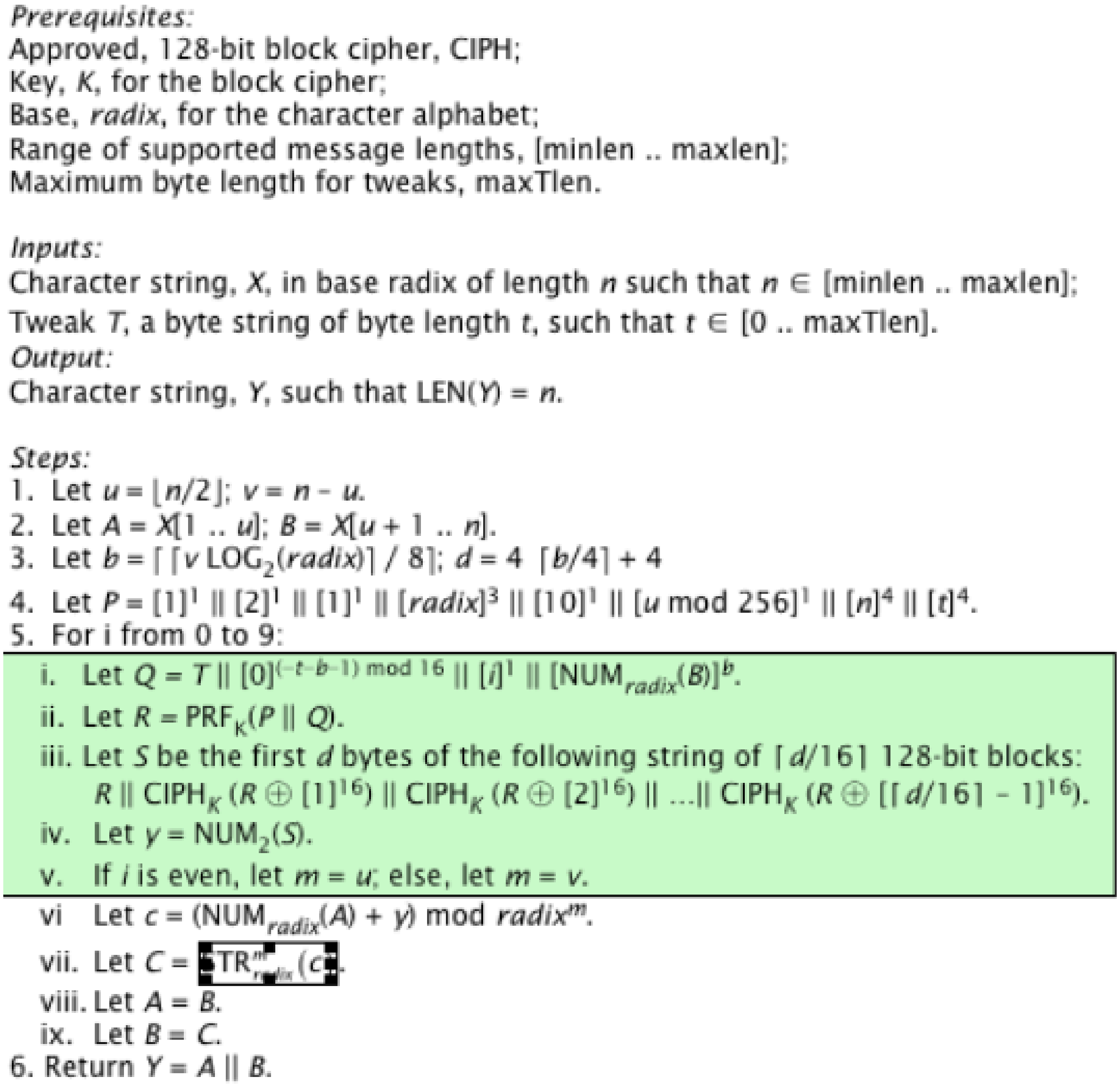


```
t
Prerequisites:
Approved, 128-bit block cipher, CIPH;
Key, $K$, for the block cipher;

Input:
Nonempty bit string, $X$, such that LEN($X$) = is a multiple of 128.
Output:
128-bit block, $Y$

Steps:
1. Let $m$ = LEN($X$)/128.
2. Partition $X$ into $m$ 128-bit blocks $X_1,\ldots, X_m$, so that $X = X_1 \| \ldots \| X_m$
3. Let $Y_0 = [0]^{16}$
4. For $j$ from 1 to $m$:
4.i  let $Y_j = \text{CIPH}_K(Y_{j-1} \oplus X_j)$.
6. Return $Y_m$.
```

Algorithm FF1(FFX|Radix)



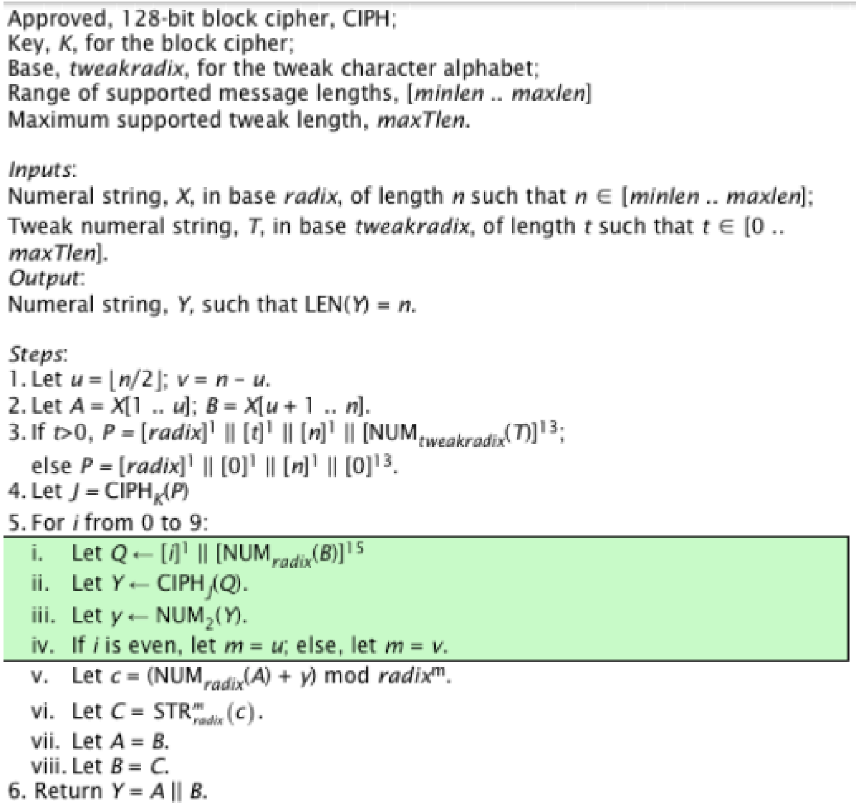
Prerequisites:
Approved, 128-bit block cipher, CIPH;
Key, *K*, for the block cipher;
Base, *radix*, for the character alphabet;
Range of supported message lengths, [minlen .. maxlen];
Maximum byte length for tweaks, maxTlen.

Inputs:
Character string, *X*, in base radix of length *n* such that $n \in$ [minlen .. maxlen];
Tweak *T*, a byte string of byte length *t*, such that $t \in$ [0 .. maxTlen].
Output:
Character string, *Y*, such that LEN(*Y*) = *n*.

Steps:

```
1. Let $u = \lfloor n/2 \rfloor$; $v = n - u$.
2. Let $A = X[1 .. u]$; $B = X[u + 1 .. n]$.
3. Let $b = \lceil \lceil v \, \mathrm{LOG}_2(radix) \rceil / 8 \rceil$; $d = 4 \; \lceil b/4 \rceil + 4$
4. Let $P = [1]^1 \,||\, [2]^1 \,||\, [1]^1 \,||\, [radix]^3 \,||\, [10]^1 \,||\, [u \bmod 256]^1 \,||\, [n]^4 \,||\, [t]^4$.
5. For i from 0 to 9:
    i.    Let $Q = T \,||\, [0]^{(-t-b-1) \bmod 16} \,||\, [i]^1 \,||\, [\mathrm{NUM}_{radix}(B)]^b$.
    ii.   Let $R = \mathrm{PRF}_K(P \,||\, Q)$.
    iii.  Let $S$ be the first $d$ bytes of the following string of $\lceil d/16 \rceil$ 128-bit blocks:
              $R \,||\, \mathrm{CIPH}_K(R \oplus [1]^{16}) \,||\, \mathrm{CIPH}_K(R \oplus [2]^{16}) \,||\, ... \,||\, \mathrm{CIPH}_K(R \oplus [\lceil d/16 \rceil - 1]^{16})$.
    iv.   Let $y = \mathrm{NUM}_2(S)$.
    v.    If $i$ is even, let $m = u$; else, let $m = v$.
    vi    Let $c = (\mathrm{NUM}_{radix}(A) + y) \bmod radix^m$.
    vii.  Let $C = \mathrm{STR}^m_{radix}(c)$.
    viii. Let $A = B$.
    ix.   Let $B = C$.
6. Return $Y = A \,||\, B$.
```

Algorithm FF2(VAES3)



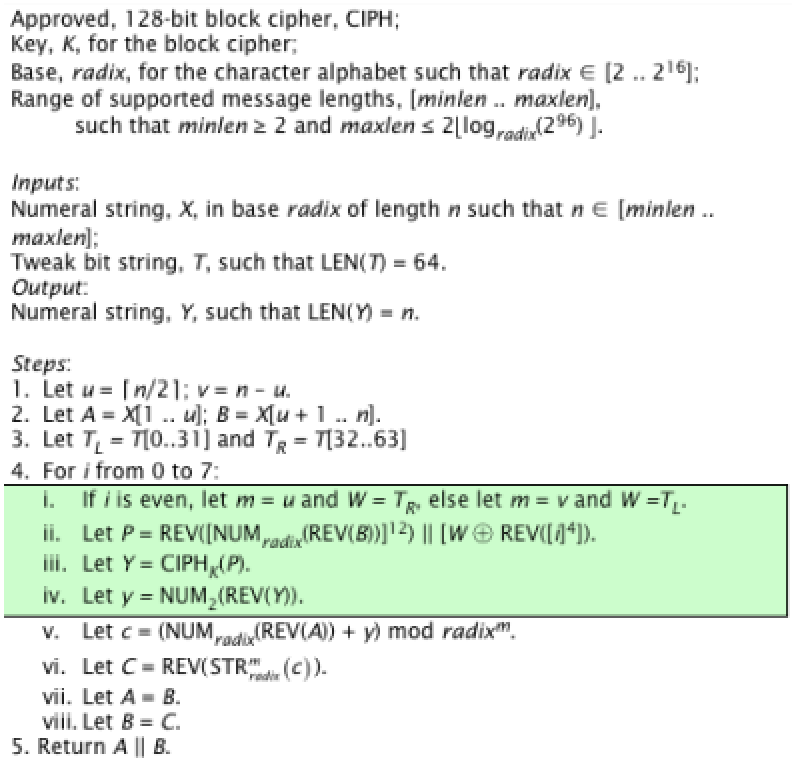
Approved, 128-bit block cipher, CIPH;
Key, *K*, for the block cipher;
Base, *tweakradix*, for the tweak character alphabet;
Range of supported message lengths, [*minlen* .. *maxlen*]
Maximum supported tweak length, *maxTlen*.

Inputs:
Numeral string, *X*, in base *radix*, of length *n* such that $n \in$ [*minlen* .. *maxlen*];
Tweak numeral string, *T*, in base *tweakradix*, of length *t* such that $t \in$ [0 .. *maxTlen*].
Output:
Numeral string, *Y*, such that LEN(*Y*) = *n*.

Steps:

```
1. Let $u = \lfloor n/2 \rfloor$; $v = n - u$.
2. Let $A = X[1 .. u]$; $B = X[u + 1 .. n]$.
3. If $t>0$, $P = [radix]^1 \| [t]^1 \| [n]^1 \| [\mathrm{NUM}_{tweakradix}(T)]^{13}$;
   else $P = [radix]^1 \| [0]^1 \| [n]^1 \| [0]^{13}$.
4. Let $J = \mathrm{CIPH}_K(P)$
5. For $i$ from 0 to 9:
   i.    Let $Q \leftarrow [i]^1 \| [\mathrm{NUM}_{radix}(B)]^{15}$
   ii.   Let $Y \leftarrow \mathrm{CIPH}_J(Q)$.
   iii.  Let $y \leftarrow \mathrm{NUM}_2(Y)$.
   iv.   If $i$ is even, let $m = u$; else, let $m = v$.
   v.    Let $c = (\mathrm{NUM}_{radix}(A) + y) \bmod radix^m$.
   vi.   Let $C = \mathrm{STR}^m_{radix}(c)$.
   vii.  Let $A = B$.
   viii. Let $B = C$.
6. Return $Y = A \| B$.
```

Algorithm FF3(BPS-BC)



Approved, 128-bit block cipher, CIPH;
Key, *K*, for the block cipher;
Base, *radix*, for the character alphabet such that $radix \in [2 .. 2^{16}]$;
Range of supported message lengths, [*minlen* .. *maxlen*],
such that $minlen \geq 2$ and $maxlen \leq 2\lfloor \log_{radix}(2^{96}) \rfloor$.

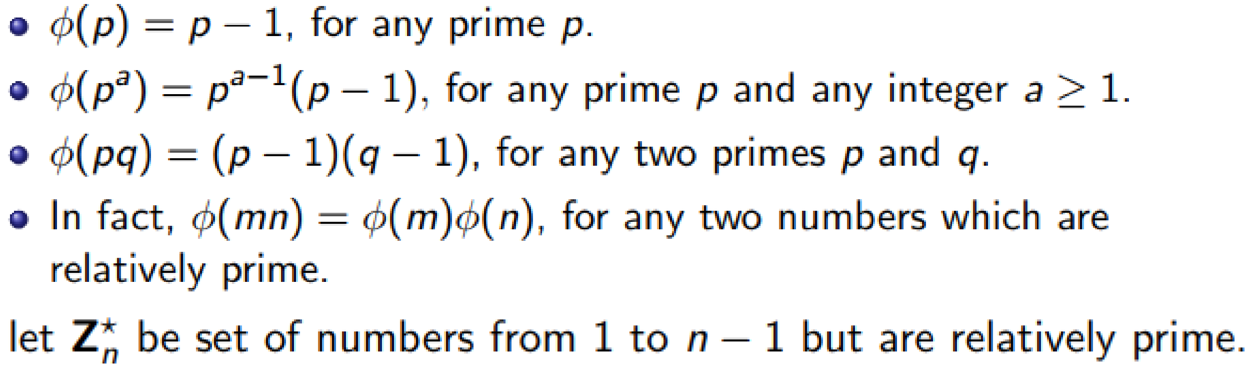
Inputs:
Numeral string, *X*, in base *radix* of length *n* such that $n \in$ [*minlen* .. *maxlen*];
Tweak bit string, *T*, such that LEN(*T*) = 64.
Output:
Numeral string, *Y*, such that LEN(*Y*) = *n*.

Steps:

```
1. Let $u = \lceil n/2 \rceil$; $v = n - u$.
2. Let $A = X[1 .. u]$; $B = X[u + 1 .. n]$.
3. Let $T_L = T[0..31]$ and $T_R = T[32..63]$
4. For $i$ from 0 to 7:
   i.    If $i$ is even, let $m = u$ and $W = T_R$, else let $m = v$ and $W = T_L$.
   ii.   Let $P = \mathrm{REV}([\mathrm{NUM}_{radix}(\mathrm{REV}(B))]^{12}) \| [W \oplus \mathrm{REV}([i]^4])$.
   iii.  Let $Y = \mathrm{CIPH}_K(P)$.
   iv.   Let $y = \mathrm{NUM}_2(\mathrm{REV}(Y))$.
   v.    Let $c = (\mathrm{NUM}_{radix}(\mathrm{REV}(A)) + y) \bmod radix^m$.
   vi.   Let $C = \mathrm{REV}(\mathrm{STR}^m_{radix}(c))$.
   vii.  Let $A = B$.
   viii. Let $B = C$.
5. Return $A \| B$.
```

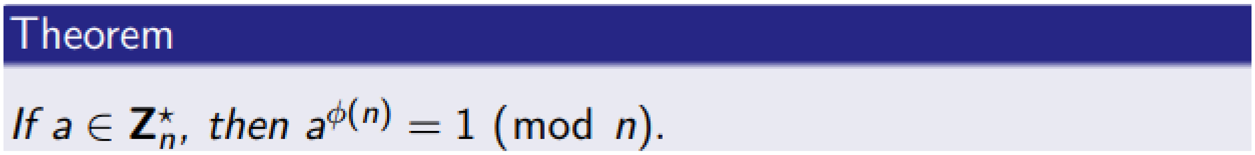
Class 6

Properties Euler's Phi (ϕ)Function



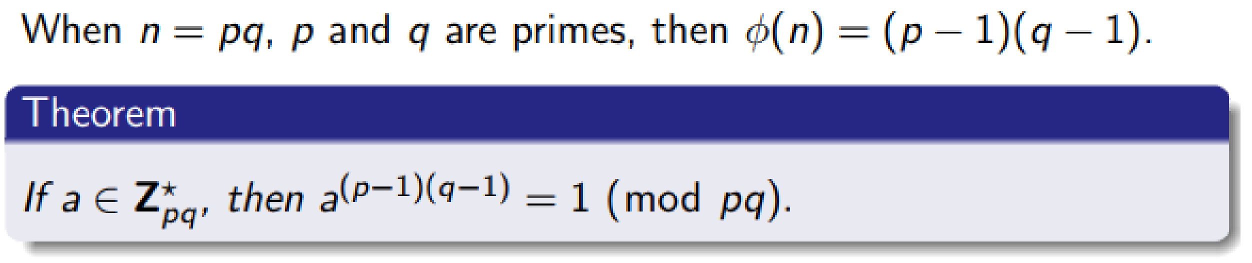
- $\phi(p) = p - 1$, for any prime p.
- $\phi(p^a) = p^{a-1}(p-1)$, for any prime p and any integer $a \geq 1$.
- $\phi(pq) = (p-1)(q-1)$, for any two primes p and q.
- In fact, $\phi(mn) = \phi(m)\phi(n)$, for any two numbers which are relatively prime.

let $\mathbf{Z}_n^\star$ be set of numbers from 1 to $n-1$ but are relatively prime.



Theorem

If $a \in \mathbf{Z}_n^\star$, then $a^{\phi(n)} = 1 \pmod{n}$.

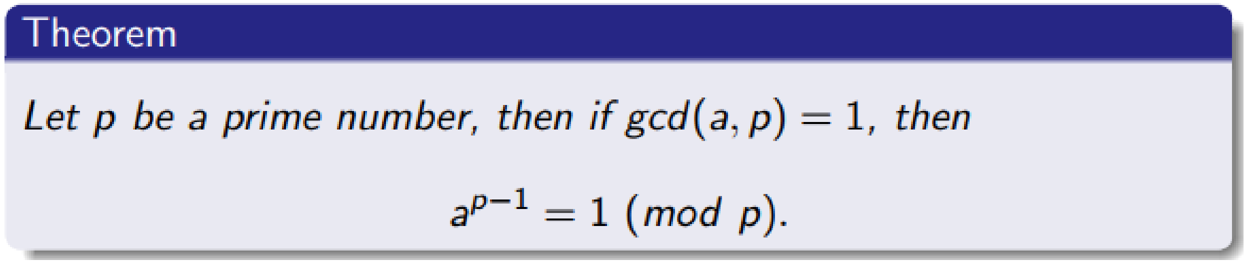


When $n = pq$, p and q are primes, then $\phi(n) = (p-1)(q-1)$.

Theorem

If $a \in \mathbf{Z}_{pq}^\star$, then $a^{(p-1)(q-1)} = 1 \pmod{pq}$.

Fermat's Theorem

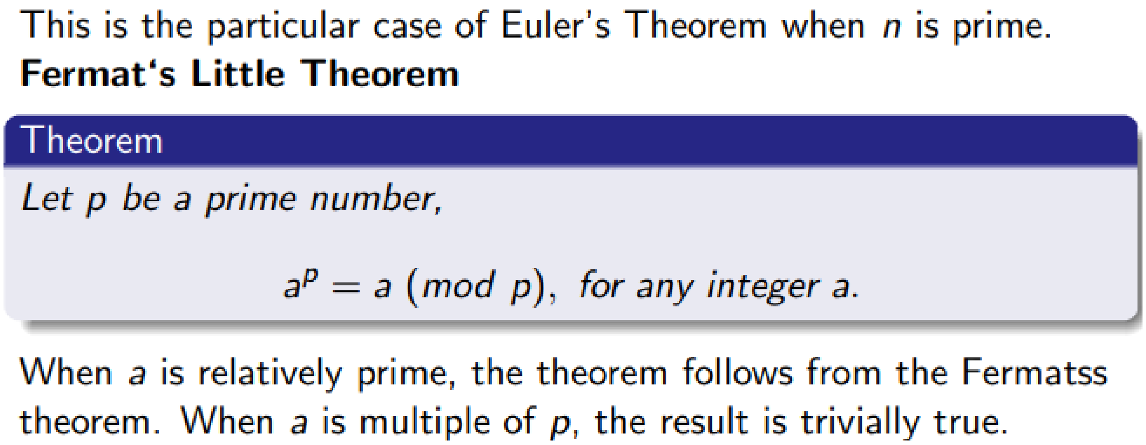


Theorem

Let p be a prime number, then if $gcd(a,p) = 1$, then

$$a^{p-1} = 1 \ (mod \ p).$$

Fermat's Little Theorem



This is the particular case of Euler's Theorem when n is prime.

Fermat‘s Little Theorem

Theorem

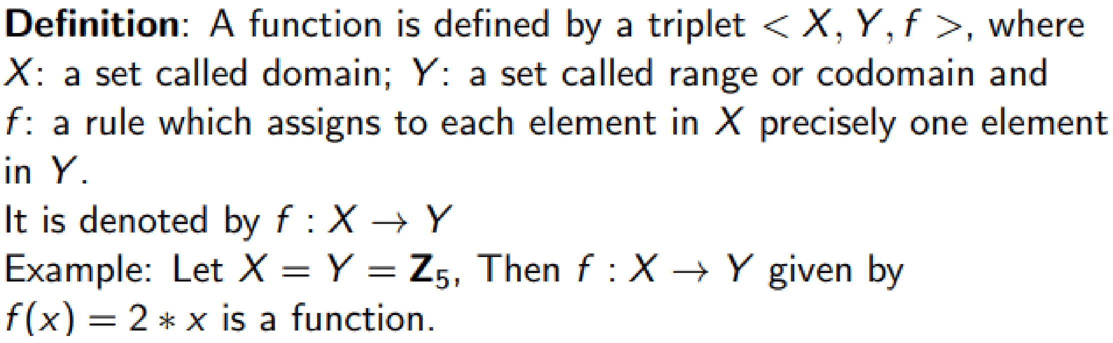
Let p be a prime number,

$$a^p = a \ (mod \ p), \ \textit{for any integer } a.$$

When a is relatively prime, the theorem follows from the Fermatss theorem. When a is multiple of p, the result is trivially true.

Functions & Chinese Remainder Theorem

Function Definition



Definition: A function is defined by a triplet $< X, Y, f >$, where X: a set called domain; Y: a set called range or codomain and f: a rule which assigns to each element in X precisely one element in Y.

It is denoted by $f : X \to Y$

Example: Let $X = Y = \mathbf{Z}_5$, Then $f : X \to Y$ given by $f(x) = 2 * x$ is a function.

Image

If x ∈ X, the image of x in Y is an element y ∈ Y such that y = f (x).

Pre-image

If y ∈ Y , then a Pre-image of y in X is an element x ∈ X such that f (x) = y.

Image of a Function

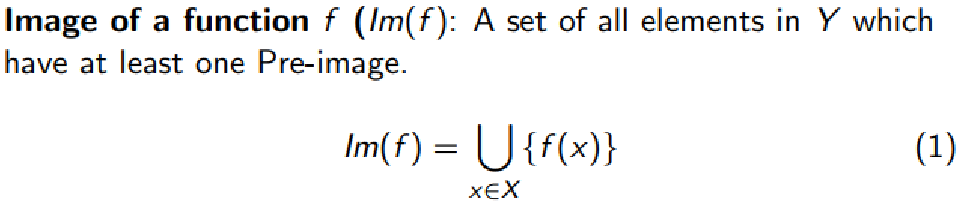


Image of a function f ($Im(f)$: A set of all elements in Y which have at least one Pre-image.

$$Im(f) = \bigcup_{x \in X} \{f(x)\} \qquad (1)$$

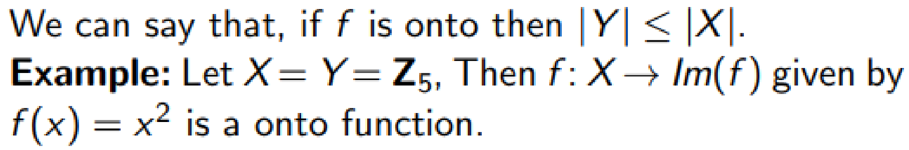
One-to-One(injective) Function

A function is one-to-one (injective) if each element in the codomain Y is the image of **at most** one element in the domain X.

- In other words, each element in x in X is related to different y in X , never two different elements in X map to the same element in Y.
- |X| <= |Y|

Onto(surjective) Function

A function is Onto (surjective) if each element in the codomain Y is the image of **at least** one element in the domain X .

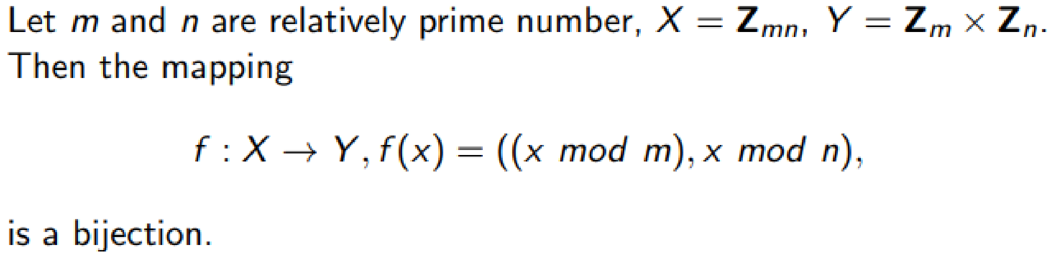


We can say that, if f is onto then $|Y| \leq |X|$.
Example: Let $X = Y = \mathbf{Z}_5$, Then $f \colon X \to Im(f)$ given by
$f(x) = x^2$ is a onto function.

Bijection Function

A function which is both one-to-one and onto.

In this case, we have |X| <= |Y| and |X| >= |Y|. This implies |X| = |Y|.

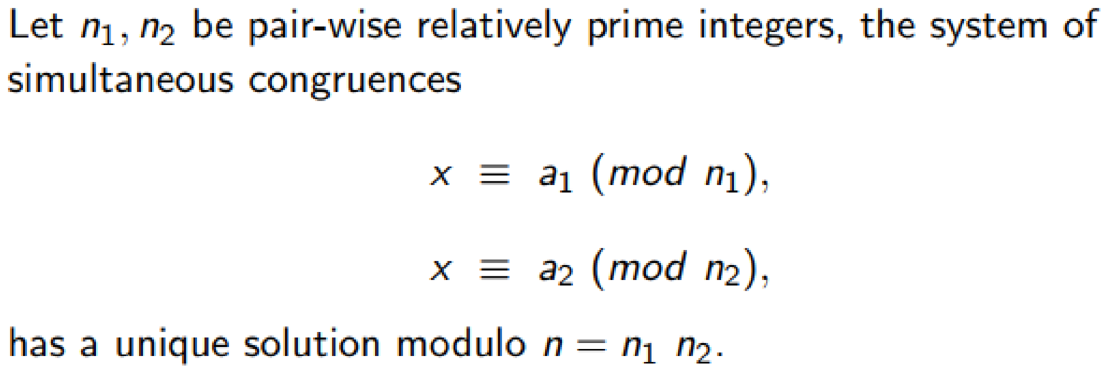


Let m and n are relatively prime number, $X = \mathbf{Z}_{mn}$, $Y = \mathbf{Z}_m \times \mathbf{Z}_n$.
Then the mapping

$$f : X \to Y, f(x) = ((x \ mod \ m), x \ mod \ n),$$

is a bijection.

Chinese Remainder Theorem(CRT)



Let n_1, n_2 be pair-wise relatively prime integers, the system of simultaneous congruences

$$x \equiv a_1 \ (mod \ n_1),$$

$$x \equiv a_2 \ (mod \ n_2),$$

has a unique solution modulo $n = n_1 \ n_2$.

Class 7

Symmetric Key System

Symmetric key is fast and provides built -in Authentication by virtue of users sharing the key.

- provide confidentiality, but never protect against each other due to sharing the same key.
  - Non-repudiation is impossible as sender and receiver are equal. One party can forge the other party’s data.
  - Risk on the sender from a receiver forging a message and then claiming that it is sent by the sender.
- In networked situations, the requirement for the key storage grows quadratic in n, the numbers of users. The number of common keys is n(n-1)/2.
- One key is used for both encryption and decryption, the key needs to be shared by both sender and receiver. If the key is disclosed, the scheme is compromised.

Asymmetric Cryptography

Two keys; a public and a private key. Mechanisms differ from the way you lock and unlock.

- Communication parties are not equal as a precondition
  - Non-Repudiation is possible, leading to natural accountability to the transactions.
- In a networked situation, the requirement for the key storage grows linearly in n, the number of users.

Properties

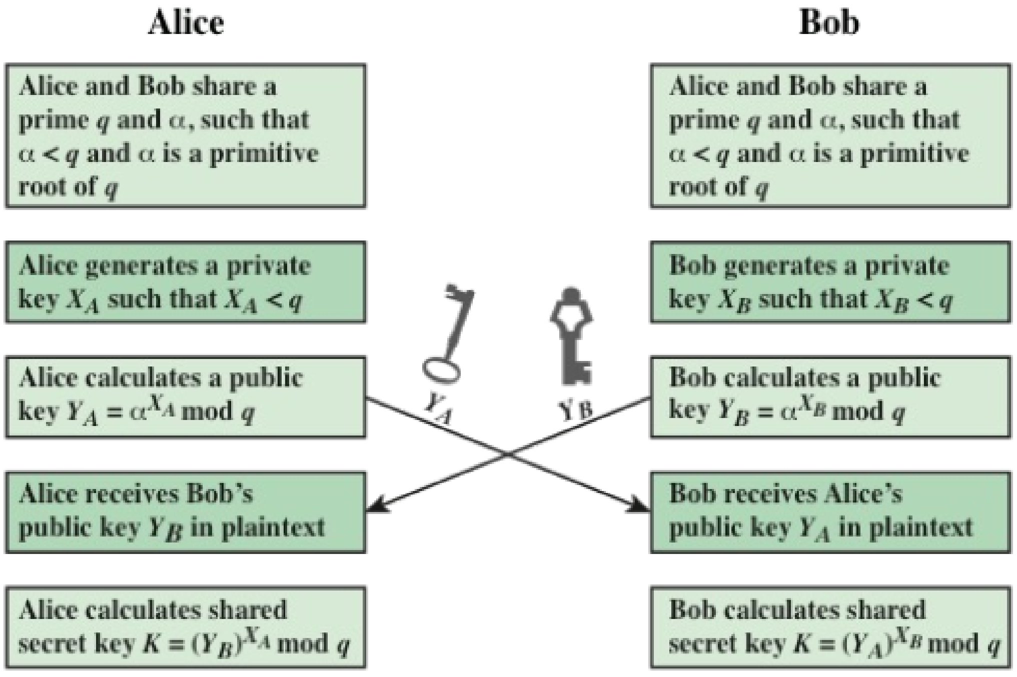
1. Encryption & Decryption
   a. Y = E(PUb, X)
   b. X = D(PRb, Y)
   c. involves the use of two keys:
      i. A public-key, which may be known by anybody and can be used to encrypt messages.Public key of B : Pub
      ii. A private-key, known only to the recipient, used to decrypt messages. Private key of B : PRb

2. Signatures
    1. involves the use of two keys:
        1. A private-key, known only to the signer is used to sign messages.
        2. A public-key, may be known by anybody and can be used to verify messages.
3. It is asymmetric because those who encrypt messages or verify signatures cannot decrypt messages or create signatures.

Diffie-Hellman Public Key Protocol

Public key promises to simplify the key management-any two users who have not met before still be able to obtain a common secret using only Public information.

- However, public key systems also bring in new key management issues that will require a trusted system to distribute public keys.
- The protocol is vulnerable to Man in the Middle Attack
    - With the nature of the Internet, someone could masquerade as Alice or Bob and try to fool Bob or Alice.
        - Because the public information they exchange is not authenticated.
    - To overcome the MITM attack, we need Digital Signatures and related concepts to tackle this attack.



Alice and Bob share a prime q and α, such that $\alpha < q$ and α is a primitive root of q

Alice and Bob share a prime q and α, such that $\alpha < q$ and α is a primitive root of q

Alice generates a private key X_A such that $X_A < q$

Bob generates a private key X_B such that $X_B < q$

Alice calculates a public key $Y_A = \alpha^{X_A} \bmod q$

Bob calculates a public key $Y_B = \alpha^{X_B} \bmod q$

Alice receives Bob's public key Y_B in plaintext

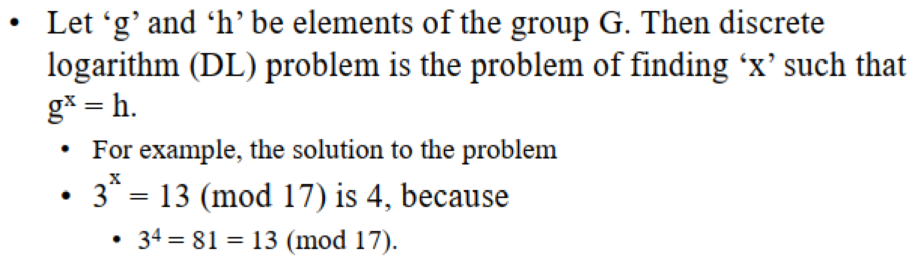
Bob receives Alice's public key Y_A in plaintext

Alice calculates shared secret key $K = (Y_B)^{X_A} \bmod q$

Bob calculates shared secret key $K = (Y_A)^{X_B} \bmod q$

Discrete Logarithm

The discrete log problem is believed to be difficult. Therefore it has become the basis of several public key schemes.



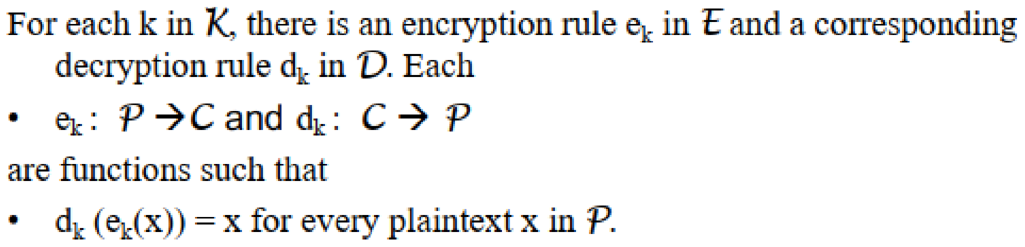
- Let ‘g’ and ‘h’ be elements of the group G. Then discrete logarithm (DL) problem is the problem of finding ‘x’ such that $g^x = h$.
  - For example, the solution to the problem
  - $3^x = 13 \pmod{17}$ is 4, because
    - $3^4 = 81 = 13 \pmod{17}$.

RSA CryptoSystem

The first Public key encryption algorithm, RSA relies on a group of numbers modulo n, which is a product of two large primes.

Basic Facts

1. Definition: A cryptosystem is a five-tuple (P,C,K,E,D).
   a. P: a set of possible plaintexts;
   b. C: a set of possible ciphertexts;
   c. K: the space of keys, a finite set of possible keys;



For each k in $\mathcal{K}$, there is an encryption rule e_k in $\mathcal{E}$ and a corresponding decryption rule d_k in $\mathcal{D}$. Each

- $e_k : \mathcal{P} \rightarrow \mathcal{C}$ and $d_k : \mathcal{C} \rightarrow \mathcal{P}$

are functions such that

- $d_k(e_k(x)) = x$ for every plaintext x in $\mathcal{P}$.

Requirements

Any public key encryption should have features below.

1. Encryption function should be publicly available.
   a. eg. from a directory.
2. Anyone should be able to encrypt.
   a. need a one way function f.
3. Only the designated user should be able to decrypt
   a. The user needs to invert f somehow
      i. f is one-way
4. create a trapdoor function which should enable to get the encrypted message.

RSA Basic

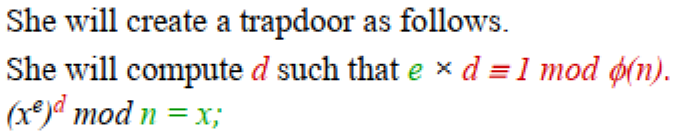
Alice claims that she knows the factorization of n = pq ; p, q Large Primes.

- Currently it is impossible for anyone to get p, q from n: Factorization is a hard problem.

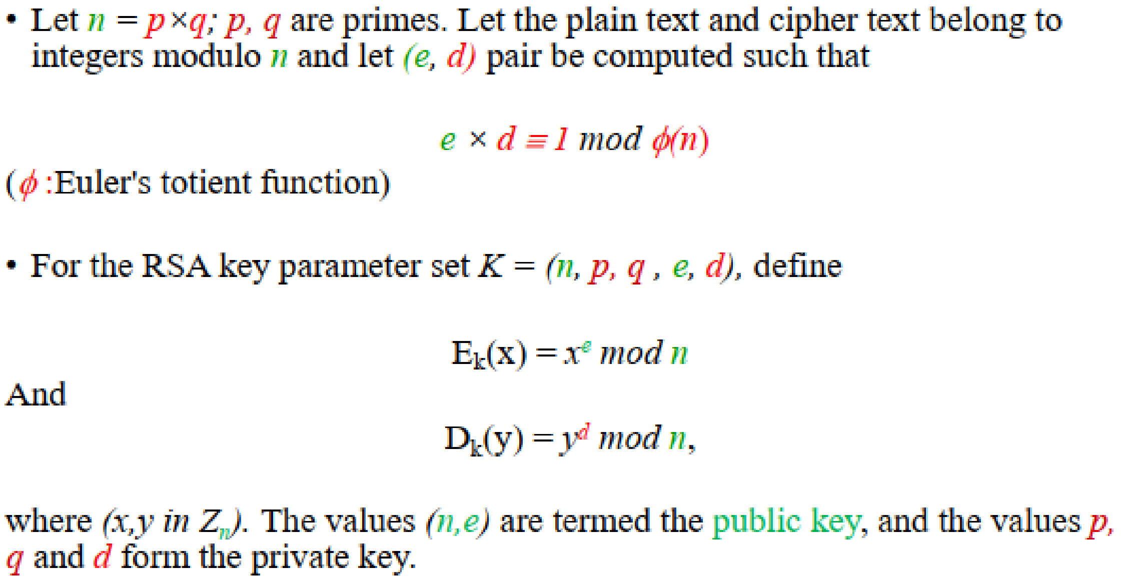
Let us work with some random x mod n.

- Assume that gcd(x,n) = 1. And consider the group generated by x mod n.
  - We can show that xf(n)= 1 mod n.

Alice needs to create a public encryption function that anyone can encrypt, but only she can decrypt.

1. Alice will choose e, a random number between 1 and f(n) and make it public.
2. So, Bob can take (e,n) and compute:
   a.
   x^e *mod n*, as his encryption.
3. No one else can work backwards from x^e to x because it is another hard problem finding e^(th) root mod n (also known as RSA problem).
4. Alice recover x by
   
   She will create a trapdoor as follows.
   She will compute d such that $e \times d \equiv 1 \bmod \phi(n)$.
   a. $(x^e)^d \bmod n = x;$

RSA PKC(1978)



- Let $n = p \times q$; p, q are primes. Let the plain text and cipher text belong to integers modulo n and let (e, d) pair be computed such that

$$e \times d \equiv 1 \bmod \phi(n)$$

(ϕ :Euler's totient function)

- For the RSA key parameter set $K = (n, p, q, e, d)$, define

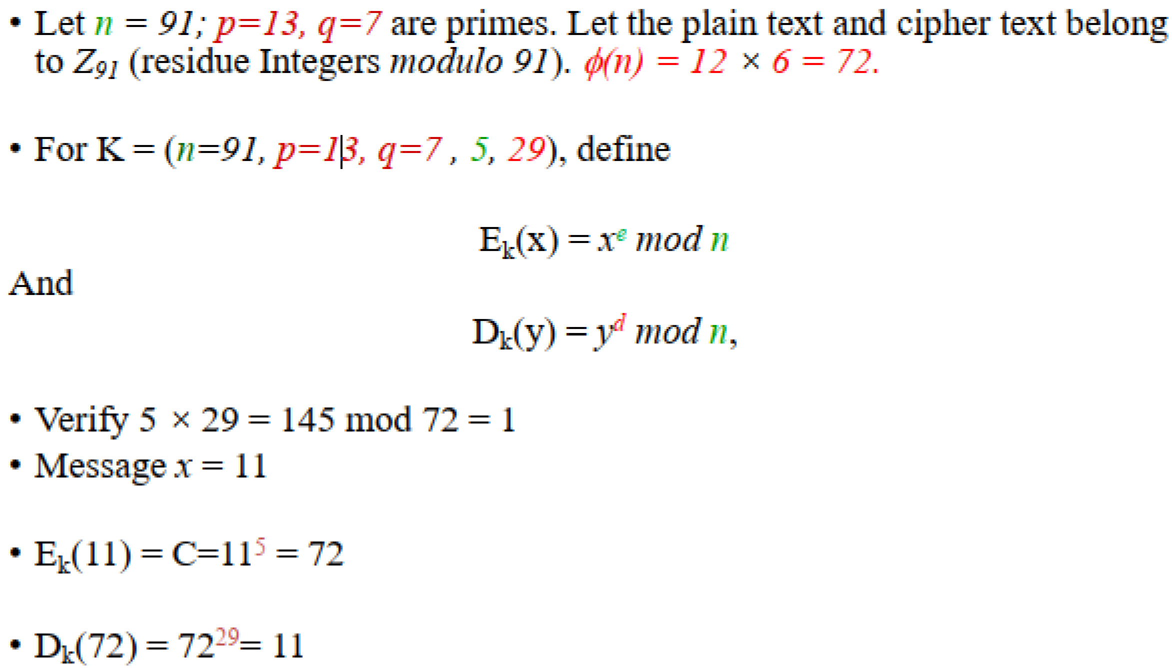
$$E_k(x) = x^e \bmod n$$

And

$$D_k(y) = y^d \bmod n,$$

where $(x,y \text{ in } Z_n)$. The values (n,e) are termed the public key, and the values p, q and d form the private key.

Example



- Let $n = 91$; $p=13$, $q=7$ are primes. Let the plain text and cipher text belong to Z_{91} (residue Integers *modulo 91*). $\phi(n) = 12 \times 6 = 72$.

- For K = ($n=91$, $p=13$, $q=7$, 5, 29), define

$$E_k(x) = x^e \bmod n$$

And

$$D_k(y) = y^d \bmod n,$$

- Verify $5 \times 29 = 145 \bmod 72 = 1$
- Message $x = 11$

- $E_k(11) = C = 11^5 = 72$

- $D_k(72) = 72^{29} = 11$

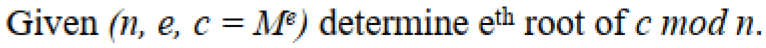
Security of RSA

The security depends on the hardness of the factorization problem.

- If someone can obtain factors p or q, then they can find out ф(n) and can determine the decryption exponent itself.

RSA problem

Given a positive integer n that is a product of two distinct odd primes p and q,(n=pq) and a positive integer e such that gcd(e,(p-1)(q-1)) = 1,and an integer c, find an integer m such that m^e = c (mod n).



Given $(n, e, c = M^e)$ determine e^th^ root of $c \bmod n$.

- Hard to brute force because the complexity is sub exponential on the key size.
    - Quantum computing can help to factor n efficiently, however it may take some years before they are developed.

Security Notions

The security of a cryptosystem is defined with respect to the attacks it can withstand, in default assume attacker will not be given private or secret information of the cryptographic key whose public cryptosystem he is attacking.

Active Attacks:

Chosen-plaintext attack(CPA)

- Encryption box is available to the attacker before the attack.
- The attacker can obtain cipher texts corresponding to any chosen plain texts.
- The goal is to weaken the cryptosystem with the obtained plaintext-ciphertext pairs
- In Public Key Cryptography, attacker can create as many public keys as he can to study its security. Interesting attacks are in fact with breaking decryption, i.e CCA attacks.

Chosen-ciphertext attack(CCA)

- Decryption box is available to the attacker before the attack.
- The basic RSA algorithm is vulnerable to a CCA.
    - In this scenario, the adversary gets decryption of a number of ciphertexts of his choice.
    - Adversary will then be given a challenge cipher text for which he has to produce the decryption (without having access to the private key).
    - This is because of the multiplicative Property of the RSA Algorithm
        - $(M_1 \times M_2)^e = M_1^e \times M_2^e = (M_1 \times M_2)^e$

          $(C_1 \times C_2)^d = C_1^d \times C_2^d = (C_1 \times C_2)^d$

Adaptive Chosen-ciphertext attack(CCA2)

- Decryption box is available to the attacker except for the challenged ciphertext.
- Attackers can obtain plaintexts corresponding to any chosen ciphertexts. This means the attacker gets decryption assistance for any chosen ciphertext.
- The goal for the attacker is to obtain any part of the plaintext after the decryption assistance is terminated.

Timing Attacks

Slightly different to the previous attacks. The attacker will observe the behavior of the Cryptographic algorithms to different inputs and use the experience to break the secret directly.

- The attack is applicable to a wide range of cryptographic algorithms.

- It can be devastating especially because the adversary only needs ciphertexts.
- If you observe variability in any aspects of the crypto algorithm, you may be able to convert into an attack.
    - The generalizations of this attack include power analysis attack and fault based attack.
    - The later, certain faults are introduced deliberately and the attacker studies the algorithm.

Mitigation

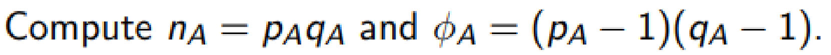
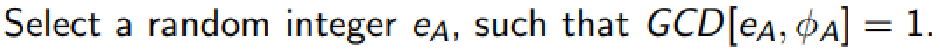
1. Constant time
    a. make sure that your algorithm takes a constant time for all inputs.
        i. This approach requires you to estimate the longest delay in advance and use appropriate idle time when results take less than the worst case time.
        ii. This method may still leak power profiles. In general performance decreases in efficiency.
2. Random delay
    a. Add a random delay to algorithm execution to ensure that the relationship between key and the execution time is uncorrelated.
3. Blinding
    a. Use the blinding technique introduced earlier. With this, the algorithm takes a random amount of time and assures that the relationship between key and the execution time is uncorrelated.

Class 8

Proof of RSA Encryption

Before starting any transactions, Alice(A) and Bob (B) will set up the following key initializations.

Alice will do: (Bob do the same thing)

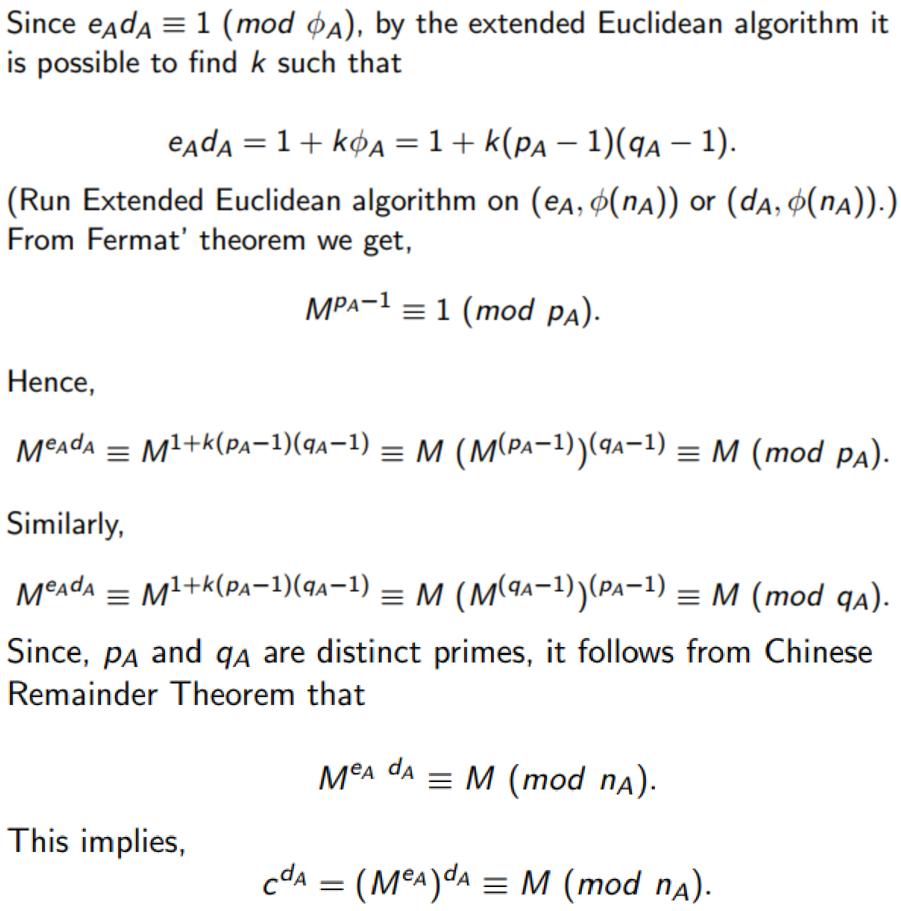
1. Generate two large and distinct primes pA and qA of almost equal size.
2. 
Compute $n_A = p_A q_A$ and $\phi_A = (p_A - 1)(q_A - 1)$.
3. 
Select a random integer e_A, such that $GCD[e_A, \phi_A] = 1$.

4. Compute the integer dA
    a. $e_A d_A \equiv 1 \ (mod \ \phi_A)$.
5. Key generated finished.
    a. Alice's Public key is (nA; eA).
    b. Alice's Private key is dA.

Start message transmission

1. Get A's Public Key (nA; eA).
2. Choose a message M as an integer in the interval [0, nA - 1].
3. Compute $c = M^{e_A} \ (mod \ n_A)$.
4. Send the ciphertext c to A
5. Decryption at A
    a. To recover m compute $M = c^{d_A} \ mod \ n_A$ using the secret d_A.

Proof of RSA Decryption



Since $e_A d_A \equiv 1 \ (mod \ \phi_A)$, by the extended Euclidean algorithm it is possible to find k such that

$$e_A d_A = 1 + k\phi_A = 1 + k(p_A - 1)(q_A - 1).$$

(Run Extended Euclidean algorithm on $(e_A, \phi(n_A))$ or $(d_A, \phi(n_A))$.)
From Fermat' theorem we get,

$$M^{p_A - 1} \equiv 1 \ (mod \ p_A).$$

Hence,

$$M^{e_A d_A} \equiv M^{1+k(p_A-1)(q_A-1)} \equiv M \ (M^{(p_A-1)})^{(q_A-1)} \equiv M \ (mod \ p_A).$$

Similarly,

$$M^{e_A d_A} \equiv M^{1+k(p_A-1)(q_A-1)} \equiv M \ (M^{(q_A-1)})^{(p_A-1)} \equiv M \ (mod \ q_A).$$

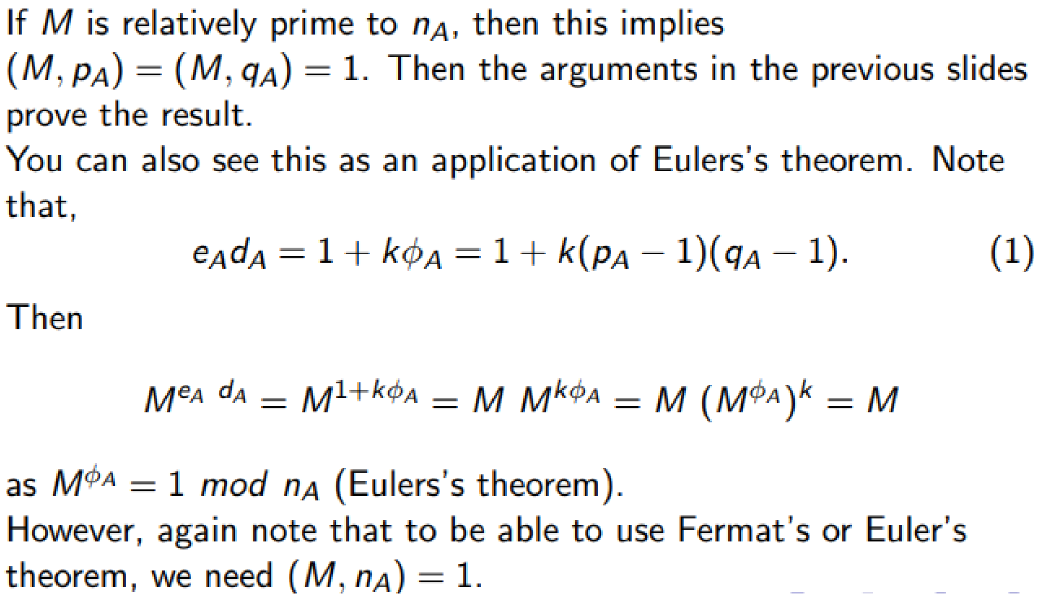
Since, p_A and q_A are distinct primes, it follows from Chinese Remainder Theorem that

$$M^{e_A \ d_A} \equiv M \ (mod \ n_A).$$

This implies,

$$c^{d_A} = (M^{e_A})^{d_A} \equiv M \ (mod \ n_A).$$

If M is relatively prime to n



If M is relatively prime to n_A, then this implies $(M, p_A) = (M, q_A) = 1$. Then the arguments in the previous slides prove the result.

You can also see this as an application of Eulers's theorem. Note that,

$$e_A d_A = 1 + k\phi_A = 1 + k(p_A - 1)(q_A - 1). \qquad (1)$$

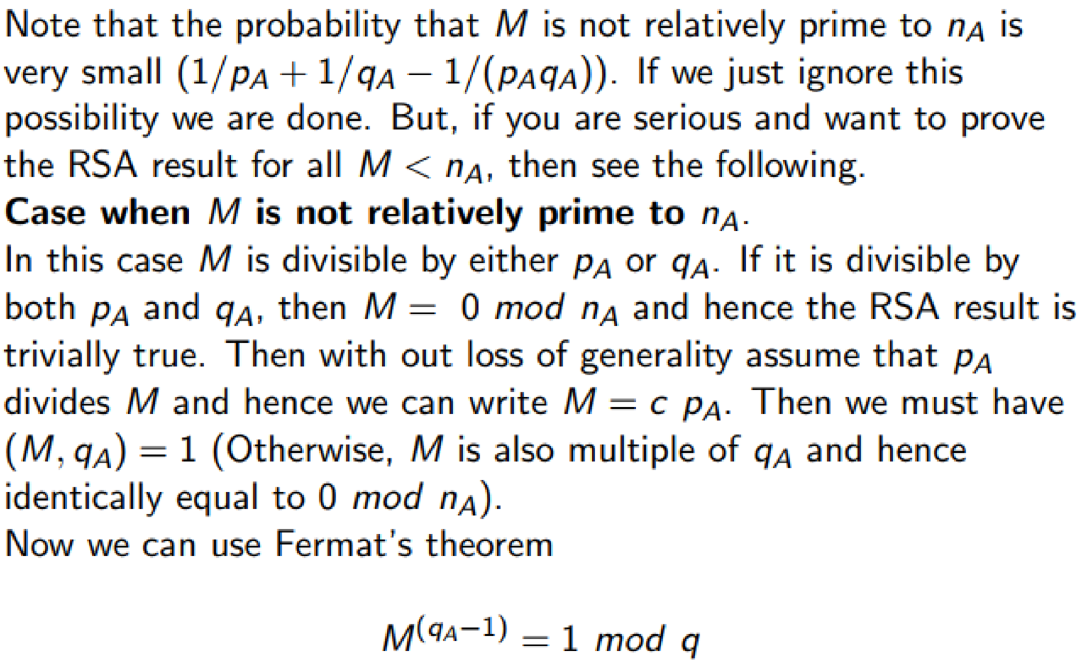
Then

$$M^{e_A\ d_A} = M^{1+k\phi_A} = M\ M^{k\phi_A} = M\ (M^{\phi_A})^k = M$$

as $M^{\phi_A} = 1\ mod\ n_A$ (Eulers's theorem).

However, again note that to be able to use Fermat's or Euler's theorem, we need $(M, n_A) = 1$.

If M is not relatively prime to n



Note that the probability that M is not relatively prime to n_A is very small $(1/p_A + 1/q_A - 1/(p_A q_A))$. If we just ignore this possibility we are done. But, if you are serious and want to prove the RSA result for all $M < n_A$, then see the following.

Case when M is not relatively prime to n_A.

In this case M is divisible by either p_A or q_A. If it is divisible by both p_A and q_A, then $M = \ 0\ mod\ n_A$ and hence the RSA result is trivially true. Then with out loss of generality assume that p_A divides M and hence we can write $M = c\ p_A$. Then we must have $(M, q_A) = 1$ (Otherwise, M is also multiple of q_A and hence identically equal to $0\ mod\ n_A$).

Now we can use Fermat's theorem

$$M^{(q_A - 1)} = 1\ mod\ q$$

Then taking $(k(p_A - 1))^{th}$ power on either side of the above equation, we get,

$$M^{k(p_A-1)(q_A-1)} = 1 \; mod \; q_A,$$

where k is as in (1). This implies

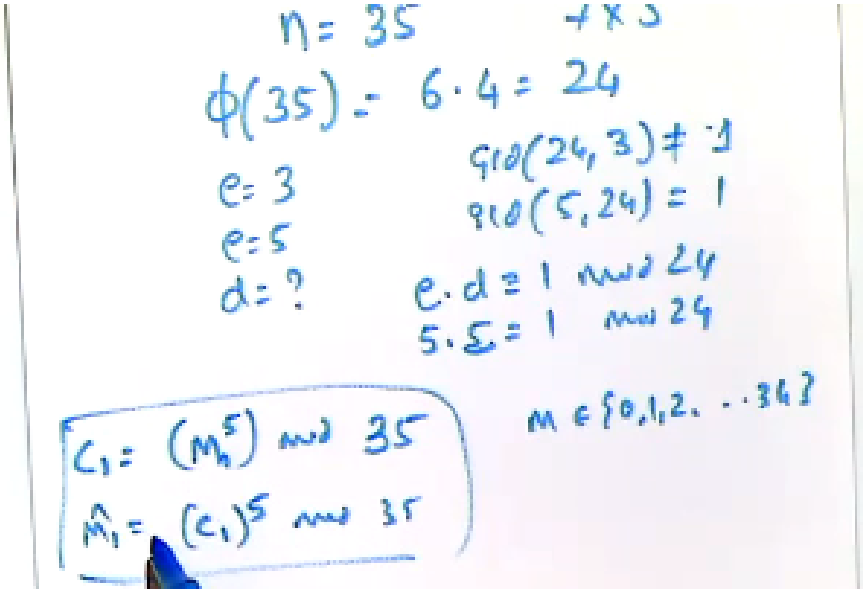
$$M^{k(p_A-1)(q_A-1)} = 1 + k' \; q_A,$$

for some k'. Multiplying each side by $M = cp_A$, we get

$$M^{k(p_A-1)(q_A-1)+1} = (1+k' \; q_A)M = M + k' \; (c \; p_A) \; q_A = M + k'' \; n_A.$$

Taking $mod \; n_A$ on both sides gives the result.

Example



$n = 35$ $\quad 7 \times 5$

$\phi(35) = 6 \cdot 4 = 24$

$e = 3$ $\quad gcd(24,3) \neq 1$

$e = 5$ $\quad gcd(5,24) = 1$

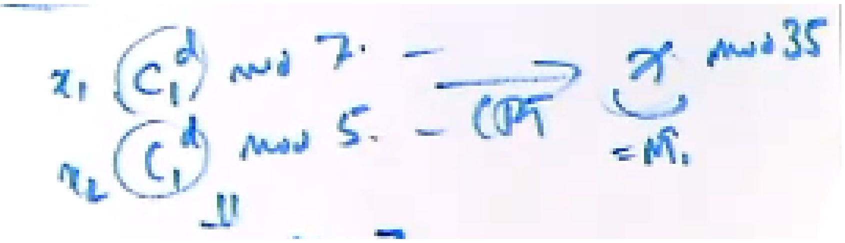
$d = ?$ $\quad e \cdot d \equiv 1 \; mod \; 24$

$5 \cdot 5 \equiv 1 \; mod \; 24$

$M \in \{0,1,2,\ldots,34\}$

$C_1 = (M_1^5) \; mod \; 35$

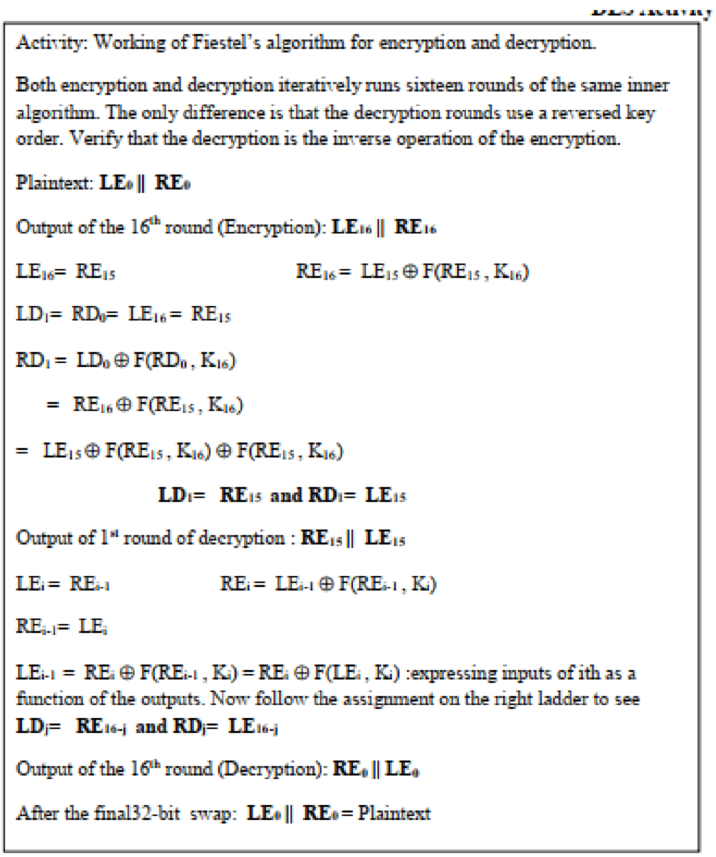
$\hat{M}_1 = (C_1)^5 \; mod \; 35$



$x_1 \; (C_1^d) \; mod \; 7. \; \longrightarrow \; x \; mod \; 35$

$x_2 \; (C_1^d) \; mod \; 5. \; - \; CRT \quad = M_1$

DES Activity Solution



Activity: Working of Fiestel's algorithm for encryption and decryption.

Both encryption and decryption iteratively runs sixteen rounds of the same inner algorithm. The only difference is that the decryption rounds use a reversed key order. Verify that the decryption is the inverse operation of the encryption.

Plaintext: $\mathbf{LE_0 \| RE_0}$

Output of the 16th round (Encryption): $\mathbf{LE_{16} \| RE_{16}}$

$LE_{16} = RE_{15}$ $\quad\quad$ $RE_{16} = LE_{15} \oplus F(RE_{15}, K_{16})$

$LD_1 = RD_0 = LE_{16} = RE_{15}$

$RD_1 = LD_0 \oplus F(RD_0, K_{16})$

$= RE_{16} \oplus F(RE_{15}, K_{16})$

$= LE_{15} \oplus F(RE_{15}, K_{16}) \oplus F(RE_{15}, K_{16})$

$\mathbf{LD_1 = RE_{15}}$ **and** $\mathbf{RD_1 = LE_{15}}$

Output of 1st round of decryption : $\mathbf{RE_{15} \| LE_{15}}$

$LE_i = RE_{i-1}$ $\quad\quad$ $RE_i = LE_{i-1} \oplus F(RE_{i-1}, K_i)$

$RE_{i-1} = LE_i$

$LE_{i-1} = RE_i \oplus F(RE_{i-1}, K_i) = RE_i \oplus F(LE_i, K_i)$:expressing inputs of ith as a function of the outputs. Now follow the assignment on the right ladder to see $\mathbf{LD_j = RE_{16-j}}$ **and** $\mathbf{RD_j = LE_{16-j}}$

Output of the 16th round (Decryption): $\mathbf{RE_0 \| LE_0}$

After the final32-bit swap: $\mathbf{LE_0 \| RE_0}$ = Plaintext

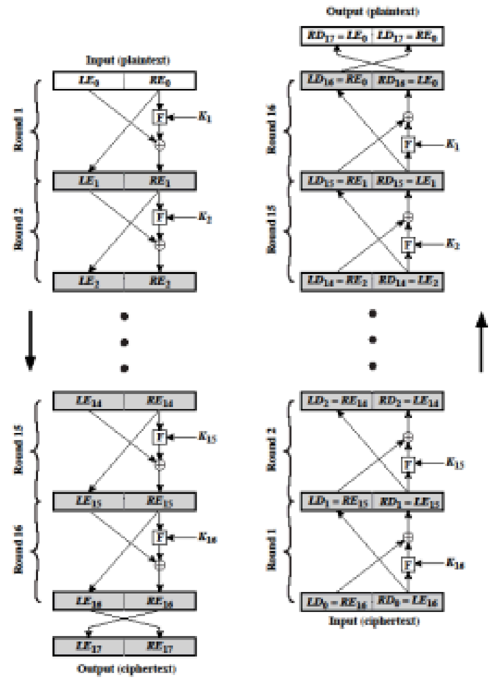


Figure 3.3 Feistel Encryption and Decryption (16 rounds)

Class 9

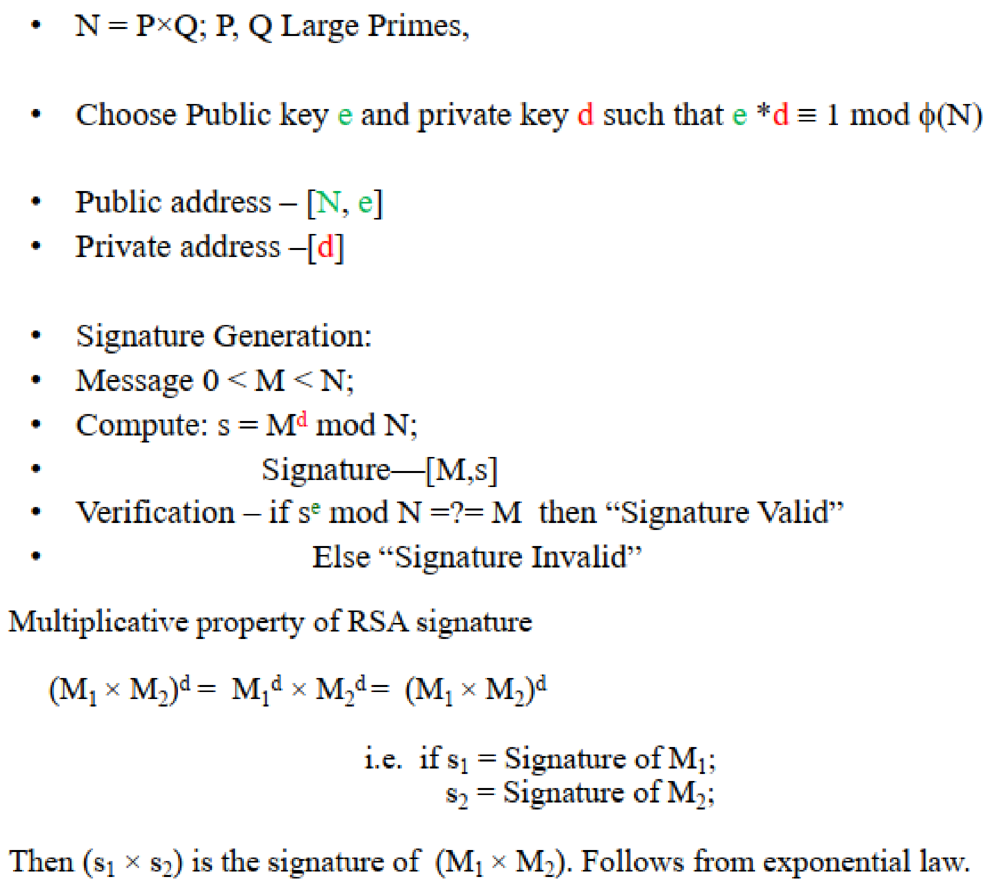
RSA Digital Signature

Digital signature is an electronic tag as a sequence of bits created by a sender on a message sequence.

- Main goal for a signature is to bind the message to the sender with assurance that the signature is not forged by anyone other than the sender.
- It enables the verification on messager sender authentication and assurance the contents of the transmitted bits are not modified before the verification.
- If the private key is stolen, there is no way to verify the digital signature.
- Different from RSA encryption, it is a complement of RSA public encryption.
    - RSA signature is a public key digital signature scheme.
    - The owner for the private key signs messages; Anyone with the public key can verify the signatures.

        - In RSA encryption, anyone with the public key can encrypt messages and the owner of the private key can decrypt the messages.
    - Sig = RF(PR,Message); Verification is a RF(PU, Message, Sig), which outputs 1 if it is a True Signature and 0 if invalid.
        - In RSA encryption, Ciphertext= RF(PU, Message), Message can be decrypted by RF(PR,Ciphertext) which also gives additional output of 1 if correctly decrypted or 0 if erroneous.
- Currently RSA modulus should be as long as 2000 to 4000 bits.

First Version of RSA Signature



- $N = P \times Q$; P, Q Large Primes,
- Choose Public key e and private key d such that $e * d \equiv 1 \bmod \phi(N)$
- Public address – [N, e]
- Private address –[d]
- Signature Generation:
- Message $0 < M < N$;
- Compute: $s = M^d \bmod N$;
- Signature—[M,s]
- Verification – if $s^e \bmod N$ =?= M then "Signature Valid"
- Else "Signature Invalid"

Multiplicative property of RSA signature

$$(M_1 \times M_2)^d = M_1^d \times M_2^d = (M_1 \times M_2)^d$$

i.e. if s_1 = Signature of M_1;
s_2 = Signature of M_2;

Then $(s_1 \times s_2)$ is the signature of $(M_1 \times M_2)$. Follows from exponential law.

Issues:

1. This property leads to a possibility of forgery of signature.
2. if Message is very long, may need to split a long message into several messages of size less than N and sign the parts one by one.
3. Blinding problem.

Blinding

- Alice's Public address – [N, e]
- Alice's Private address –[d]
- You want to get Alice sign a message M, which Alice may not be ready to do.
- Choose a random x – in the range [0..N-1]
- Form a blinded message -- $M_b = x^e M \bmod N$
- Alice may agree to sign this blinded message M_b (assume),
- Alice then signs the message M_b as $s_b = M_b^d \bmod N$
- This blinded signature can be used to compute the signature for M using the multiplicative property:
- Now you can compute signature for M as
- $s = s_b/x \bmod N$
- This is true because, let us apply the verification rule, Note
- $s^e = s_b^e /x^e = (M_b)^{d e} /x^e = (M_b) /x^e = x^e M / x^e = M$
- Hence s is the signature of M. So we have produced a forgery!

Second Version of RSA Signature

- $N = P \times Q$; P, Q Large Primes,
- Choose Public key e and private key d such that $e * d \equiv 1 \bmod \phi(N)$
- Public address – [N, e]
- Private address –[d]

- Signature Generation:
- Message $0 < M < N$;
- Find $M_1 = R(M)$; where R is a redundancy function and $1 < R(M) < N$.

- Compute: $s = M_1^d \bmod N$;
- Signature—[M,s]
- Verification – $M_1 = R(M)$; if $s^e \bmod N = ?= M_1$ then "Signature Valid"
- Else "Signature Invalid"

Issue:

1. Messages are generally long
2. RSA signature scheme needs a redundancy function to avoid existential forgery attacks.
3. Repeated messages carry the same signature.

Third Version of RSA Signature

- $N = P \times Q$; P, Q Large Primes,
- Choose Public key e and private key d such that $e * d \equiv 1 \bmod \phi(N)$
- Public address – [N, e]
- Private address –[d]

- Signature Generation:
- Let M be a Message be of arbitrary length: M =[.........]; Find H(M), where H is a Hash function
- Format M_1 = [H(M), Identity Information, Random Number], such that $1 < M_1 < N$.

- Compute: $s = M_1^d \bmod N$;
- Signature—[M,s]
- Verification – Extract M_1 by computing $s^e \bmod N$;
- If Any formatting violations – Reject the Signature.
- Further Verify H(M) = H(M) on M_1

RSA Attacks

1. Brute force Attack: (infeasible given size of numbers)
2. Attack by making use of loopholes in Key distribution.
3. Mathematical attacks (Factoring and RSA problem)
4. Elementary attacks
5. Advanced Factorization methods
6. Network attacks

Mathematical Attacks

- The RSA function is one way. This is an assumption and we do not have a proof, but it is considered as one-way by researchers.
- The problem
- Given n,e, $c=M^e \pmod{n}$,
  - Can we determine M?
  - Do we have an algorithm to find the e^th root of

    c mod n?
  - Can we find d such that $de = 1 \bmod \Phi(n)$?
- Can we factor n?

- In general the factorization is hard.

- Brute force algorithm is exponential in b, where b is number of bits in the representation of the number n to be factored.

- Complexity of the best known algorithm for factorization:

$$\exp((c+O(1)b^{1/3} \mathrm{Log}^{2/3}(b)),$$

  for some integer $c < 2$

- It is not worth thinking of factoring.
- Maybe quantum computers come to our rescue; but not in immediate future!

Elementary Attack

- Facts:
- Knowing e and d such that

$$ed = 1 \bmod \Phi(n)$$

Is equivalent to factoring.

- Knowing n, $\Phi(n)$ is also equivalent to factoring.

Common modulus

- Every user chooses same modulus n=pq set up by a trusted central authority. But each user chooses their own private and public key pairs
- User i ------(e_i,d_i)
- So using the facts in previous slide, any user can factor common modulus n and can find the private information of other user by using only the public information.
-
- Hence it is extremely important that every entity chooses its own RSA modulus n.
- Again the modulus need to be create using random primes.

Broadcast Problem

A group of entities may all have a same encryption exponent but should have different modulus. Further, to improve the public encryption, let the public key be small, say e=3.

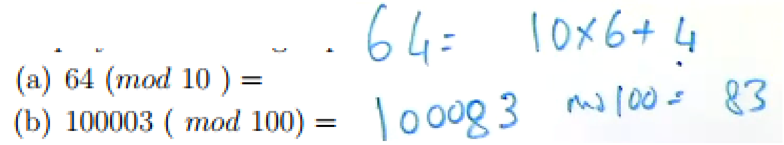
- A wishes to send a common message m to three entities with modulus n n_1,n_2 and n_3.

- The cipher text for three entities are given by
- $c_1 = m^3$ (mod n_1)
- $c_2 = m^3$ (mod n_2)
- $c_3 = m^3$ (mod n_3).

- Then, to recover the message m solve,
- $x = c_1$ (mod n_1),
- $x = c_2$ (mod n_2),
- $x = c_3$ (mod n_3),
- You can use CRT and Then obtain an unique
- $x=m^3$ modulo n_1 n_2 n_3
- m can then be obtained by taking the cube root of x. Finding a cube root in integers is not a hard problem.

Class 10

Exercise Question

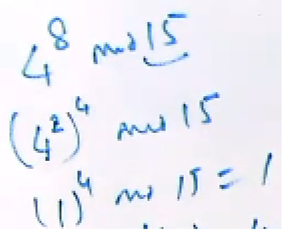


(a) $64 \ (mod\ 10) =$ $64 = 10\times6+4$

(b) $100003 \ (\ mod\ 100) =$ 100083 $mod\ 100 = 83$

(c) $2^{145}\ 3^{777}\ 9^{777}\ (mod\ 4) =$ 0 because 2^145 mod 4 = 0

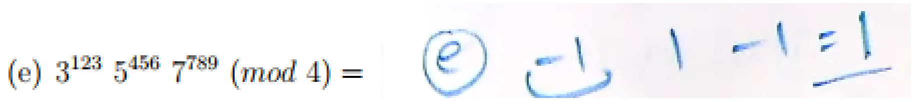
(d) $4^8\ (mod\ 15) =$;



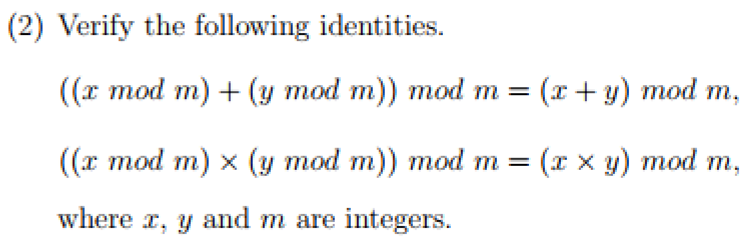
4^8 mod 15

$(4^2)^4$ mod 15

$(1)^4$ mod 15 = 1



(e) $3^{123}\,5^{456}\,7^{789}\ (mod\ 4) =$ ⓔ $-1,\ 1\ -1 = 1$



(2) Verify the following identities.

$((x\ mod\ m) + (y\ mod\ m))\ mod\ m = (x + y)\ mod\ m,$

$((x\ mod\ m) \times (y\ mod\ m))\ mod\ m = (x \times y)\ mod\ m,$

where x, y and m are integers.

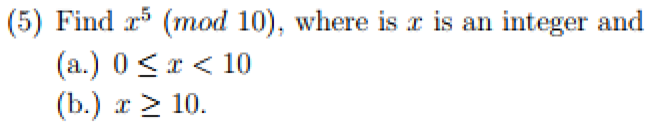
It is possible to apply basic principles of divisibility properties and Euclid's algorithm to prove the identities with rigor. In this subject, it is sufficient you realize the identities with some examples.

(3) Write an efficient algorithm for computing exponentiation in a finite structure (a group, modulo p, finite field etc).

```
Exponentiation:=function(a, exp, n);
p:=1; j:=exp; base:=a;
while (j > 0)
        if even (j)
                base = base^2; j := j div 2;
        Else
                p :=p*base; j:=j-1;
end while;
return p;
end function;
```

(4)The question above is from your background on complexity theory. The objective is to show that exponentiation can be efficiently implemented. Note that the converse operation of finding discrete logarithm is not easily implementable. Nevertheless, people come up with heuristics to solve this problem.

Can you research what is the latest progress on the complexity of the discrete logarithm problem over numbers modulo a prime?



(5) Find x^5 (mod 10), where is x is an integer and

(a.) $0 \leq x < 10$

(b.) $x \geq 10$.

For x > 10, first take x mod 10, and then use the results in (a.) to find the answer.

x	x^5 mod 10
0	0
1	1
2	2
3	3
4	4
5	5
6	6
7	7
8	8
9	9

(6) Express the following numbers as a product of primes and prime powers. 32; 63; 64; 79; 81; 124; 141; 234; 512

32 = 2^5

63 = 7 * 3^2

64 = 2^6

79 = 79

81 = 3^4

124 = 2^2 * 31

141 = 3 * 47

234 = 2 * 3^2 * 13

512 = 2^9

(7) Using the results of the above question, find gcd of the following sequences of numbers.

(a) 32, 63

32 = 2^5; 63 = 3^2*7; 63 = 3^2 * 7; The GCD(2^5, 7 * 3^2) = 1

(b) 141, 81

141=47*3; 81 = 3^4; The GCD(47*3, 3^4) = 3

(c) 81, 124

81 = 3^4, 124 = 2^2 * 31; The GCD(3^4, 2^2 * 31) = 1

(d) 79, 141

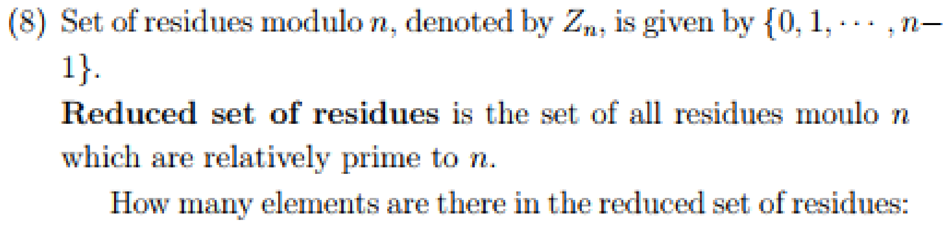
141 = 3 * 47; The GCD(79, 3 * 47) = 1

(e) 512,81

81 = 3^4; 512 = 2^9; The GCD(3^4, 2^9) = 1

(f) 124, 512

124 = 2^2 * 31; 512 = 2^9; The GCD(2^2 * 31, 2^9) = 4



(8) Set of residues modulo n, denoted by Z_n, is given by $\{0, 1, \cdots, n-1\}$.

Reduced set of residues is the set of all residues moulo n which are relatively prime to n.

How many elements are there in the reduced set of residues:

(a) modulo 11;

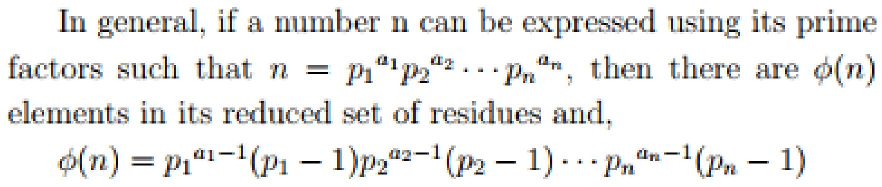
10; they are 1,2,3,4,5,6,7,8,9,10

(b) modulo 35;

(c) modulo 26;

(d) modulo 29;

(e) modulo 77.



In general, if a number n can be expressed using its prime factors such that $n = {p_1}^{a_1}{p_2}^{a_2}\cdots{p_n}^{a_n}$, then there are $\phi(n)$ elements in its reduced set of residues and,

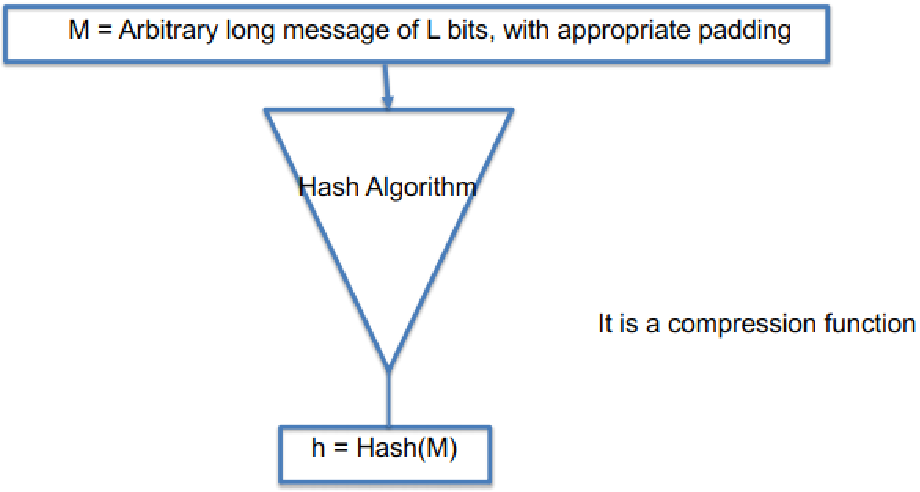
$$\phi(n) = {p_1}^{a_1-1}(p_1-1){p_2}^{a_2-1}(p_2-1)\cdots{p_n}^{a_n-1}(p_n-1)$$

Class 11

Hash Function

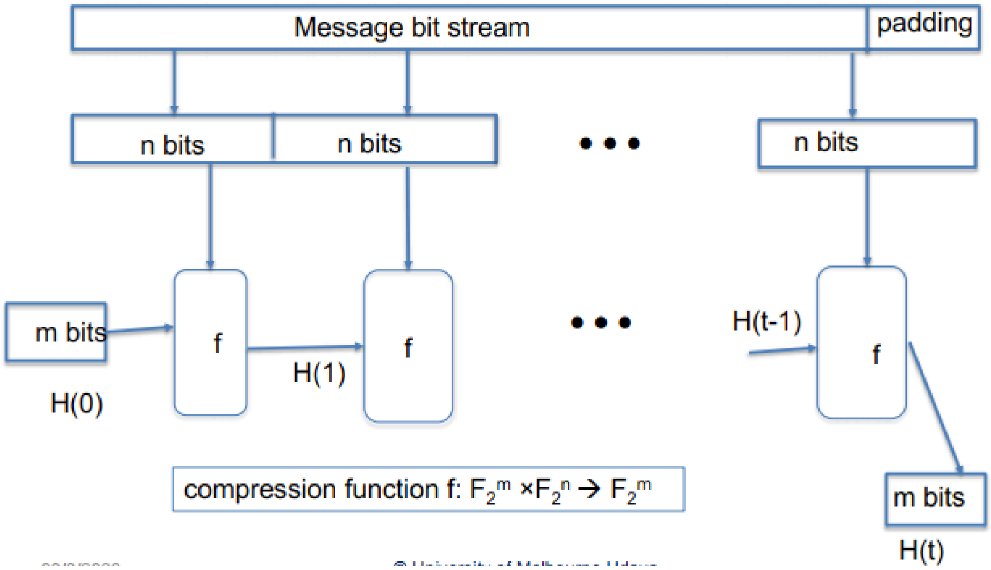
You would have used a hash function while building a Hash table data structure.

- The purpose is to distribute keys evenly in the Hash table.
- The definition of hash function is **public**.
- The function normally used is ((key) mod a p), a prime number.
- It is referred as unkeyed primitive as **does not require any key**.
    - Hash is also referred to as message digest.
- In general, the function takes a variable-length data block as input and produces a fixed length tag or digest satisfying certain properties.
    - The main objective is to obtain data integrity.
- The size of the CRC(Cyclic Redundancy Check) hash is small 32 bits.


M = Arbitrary long message of L bits, with appropriate padding
Hash Algorithm
h = Hash(M)

It is a compression function

- Hash Construction methods and Attacks on the constructions goes hand in hand.
    - Having a secure Hash function is still an open question for researchers.
- Merkle–Damgård construction
    - based on the Block cipher constructions mode CBC.
    - Heart of the construction is a compression function f: $F_2^m \times F_2^n \rightarrow F_2^m$
    - Messages are arranged as n bit blocks with possibly the final block padded to make the message length multiple of n.
    - The size of the hash is m.



Integrity

Provided by Modification Detection Codes (MDC).

- A small change in the message introduces unpredictable changes in the hash value, h = Hash (M).
  - If a message is changed while in transit, then running Hash function at the received message tells how the value is deviated from the hash value computed at the source, thus assuring integrity with high probability.

Authentication

To provide authentication with a Hash function, need to involve other encryption techniques.

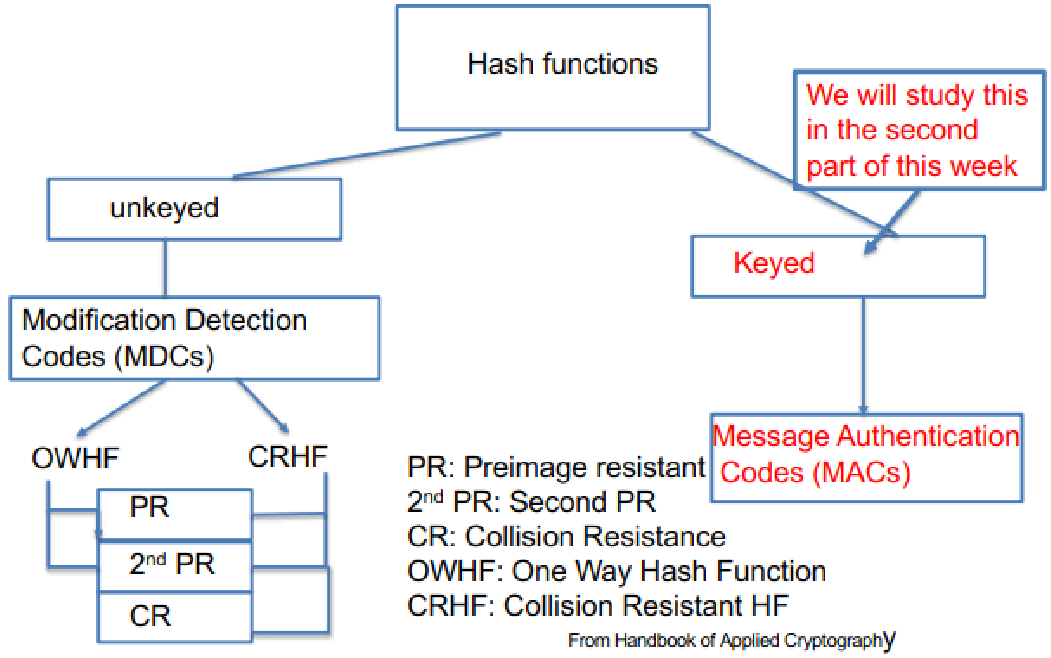
- For example, Alice and Bob need to protect the hash value using encryption techniques.

Digital Signatures

Hash functions are essential to Digital Signatures.

- A hash value is encrypted using the private key of the sender, so the receiver with access to the public key can decrypt and verify the hash value, thereby providing message authentication.
- If need confidentiality, then the message and hash value need to be encrypted as well.

Hash Function Classification


Hash functions
We will study this in the second part of this week
unkeyed
Keyed
Modification Detection Codes (MDCs)
Message Authentication Codes (MACs)
OWHF
CRHF
PR
2nd PR
CR

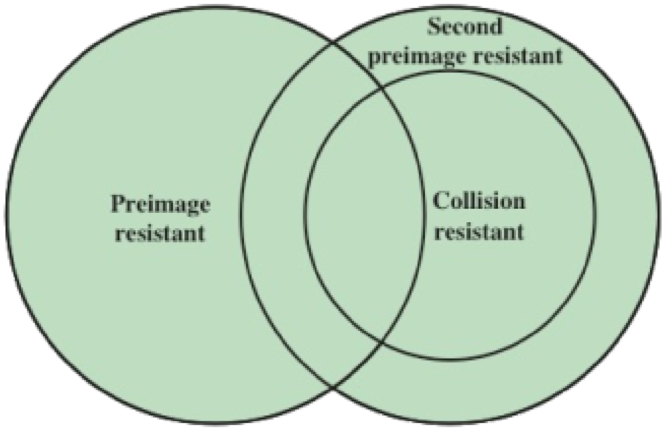
PR: Preimage resistant
2nd PR: Second PR
CR: Collision Resistance
OWHF: One Way Hash Function
CRHF: Collision Resistant HF

From Handbook of Applied Cryptography

Definitions

1. "Easy" means that an efficiently computable
   a. for example computing with polynomial time complexity.
2. "Hard" means that computationally infeasible
   a. for example having only exponential time complexity.
3. PR (Preimage Resistance)
   a. Given an element in the hash space, it is computationally infeasible to work out an input in the message space that results in the hash value as output.
      i. given y in F_2^n, it is not feasible to workout a preimage x' in F_2^* such that $H(x') = y$.
4. 2nd PR (Second Preimage Resistance)
   a. Given a message and hash pair (x, H(x)), it is computationally infeasible to work out another input in the message space that results in the hash value as output.
   b. given x in F_2^* it is not feasible workout another preimage x' in F_2^* such that $H(x') = H(x)$.

5. CR (Collision Resistance)
    a. It is computationally infeasible to find two different input messages in the message space that results in the same hash value as output.
    b. it is not feasible to work out two messages x and x' such that H(x') = H(x).
    c. CR implies Second PreImage Resistance(Second PIR).



Hash Function Requirements

1. Variable input size
    a. H can be applied to a block of data of any size.
2. Fixed output size
    a. H produces a fixed-length output.
3. Efficiency
    a. H(x) is relatively easy to compute for any given x, making both hardware and software implementations practical.
4. Preimage resistant(one-way property)
    a. For any given hash value h, it is computationally infeasible to find y such H(y) = h.
5. Second preimage resistant(weak collision resistant)
    a. For any given block x, it is computationally infeasible to find y != x with H(y) = H(x).
6. Collision resistant(strong collision resistant)
    a. It is computationally infeasible to find any pair(x,y) that H(x) = H(y).
7. Pseudorandomness
    a. Output of H meets standard tests for pseudorandomness.

Attacks on Hash Function

1. The function should resist brute-force attacks and regular cryptanalysis.
2. Attack against PR: Given a random hash value, determine y such that H(y) equals to the hash value.
3. Attack against CR: The task is to determine any two messages whose hashes are same, i.e determine x, y such that H(x) = H(y).
4. Birthday Attack
    a. Can be used to find a collision in a hash function.
    b. The probability of two distinct inputs having the same output (a collision) becomes appreciable well before you've tried all possible inputs. This is especially true for hash functions that produce a fixed size of output bits.
    c. A variant of it is Meet In the Middle Attack.

Block Ciphers as Hash Functions

Use Block ciphers in CBC mode to create hash functions, similar to CBC but without a key.

1. Initialize H_0=0 and zero-pad of final block
2. Compute $H_i = E_{M_i} [H_{i-1}]$
3. use final block as the hash value

If the DES function is used the resulting hash is too small (64-bit) and hence vulnerable to a birthday attack.

Class 12

Message Authentication

Used to address message authentication. It is a dedicated primitive based on symmetric key cryptography.

Issue:

1. Message integrity
2. Validation of originator’s identity
3. Non-repudiation of the message origin

Three way:

1. Message Encryption
2. Hash functions
3. Message Authentication Code (MAC)

Need to separate message authentication function and the protocol that helps us to integrate the message authentication in the application.

- At a basic level, Create a message authentication code using a secret key.
- At a higher level, the keys are carefully managed to obtain higher level guarantees on the exchanged message including source authentication

Security Requirement

1. disclosure
2. traffic analysis
3. masquerade
4. content modification
5. sequence modification
6. timing modification
7. source repudiation
8. destination repudiation

Message Encryption

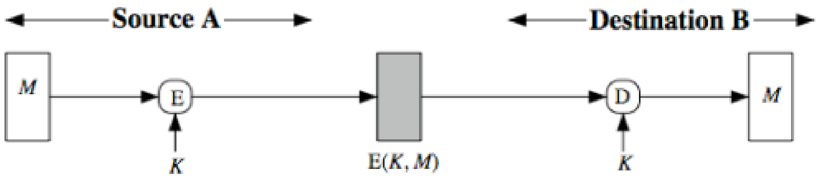
The authentication provided by message encryption are different to symmetric and public key methods.

- With public key encryption, anyone could encrypt based on public key of the receiver and if you want source authentication, the sender needs to use a signature.
    - But a symmetric key assumes that sender and receiver share a secret and encryption naturally provides authentication.

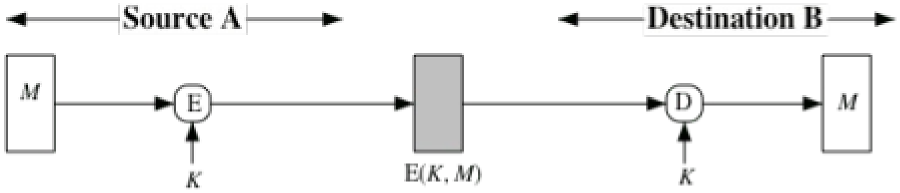
Symmetric Key Encryption

The authentication is obtained by

1. Since they share the key, the receiver is sure that the message was created by the sender.
2. By relying on the format and structure of the messages, they can detect any modification.



Symmetric KE: Confidentiality and authentication

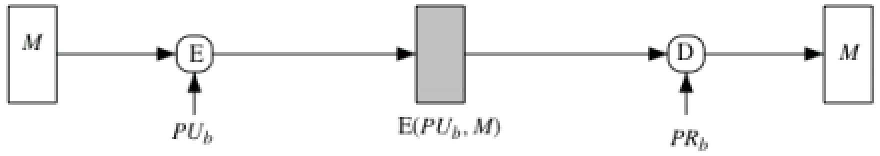


Public Key Encryption

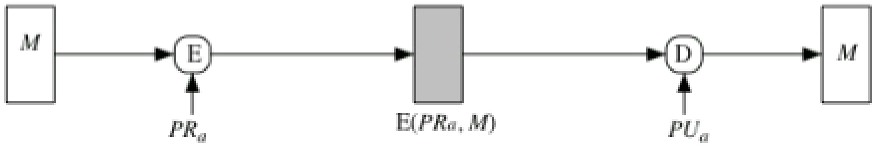
Public keys by nature, anyone can use.

- Does not provide any guarantee for the sender.
- To provide authentication, a sender needs to sign as well (use private key) which can be verified by others using the public key.
- To check if the message is corrupted
    - need some general formatting rules.

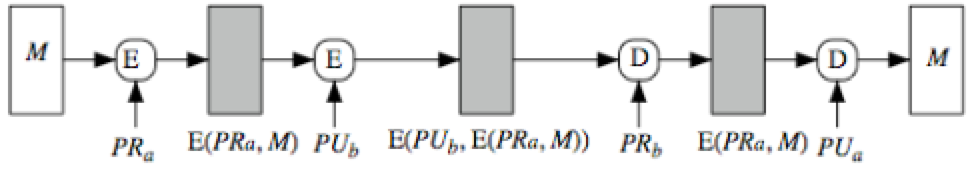
Public KE: confidentiality



Public KE: authentication and signature



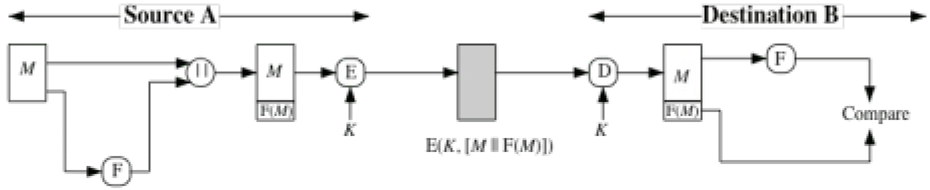
Public KE: confidentiality, authentication & signature



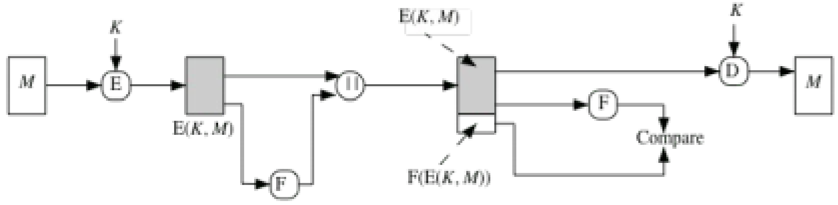
Message Authentication Code(MAC)

A dedicated symmetric key primitive aimed at providing authentication.

- The output of an algorithm can act as a signature.
- Only the receiver with the key can verify the code by running the same algorithm, thus assuring the integrity of the message from the sender.
- With encryption it can be easily integrated to provide secrecy also.
- Useful when in some applications you only need authentication.
- There are many situations where the property of authentication requires longer than confidentiality: authenticated sessions where only at times you may exchange secret information.
- Two ways of using the message authentication code:
    a. Internal Error Control



    b. External Error Control



- In real usage:
    a. A pair TCP hosts shares a secret key and all exchanges between the hosts use the same key
    b. Leads to simple encryptions between hosts-all IP packets between them can be encrypted except the header.

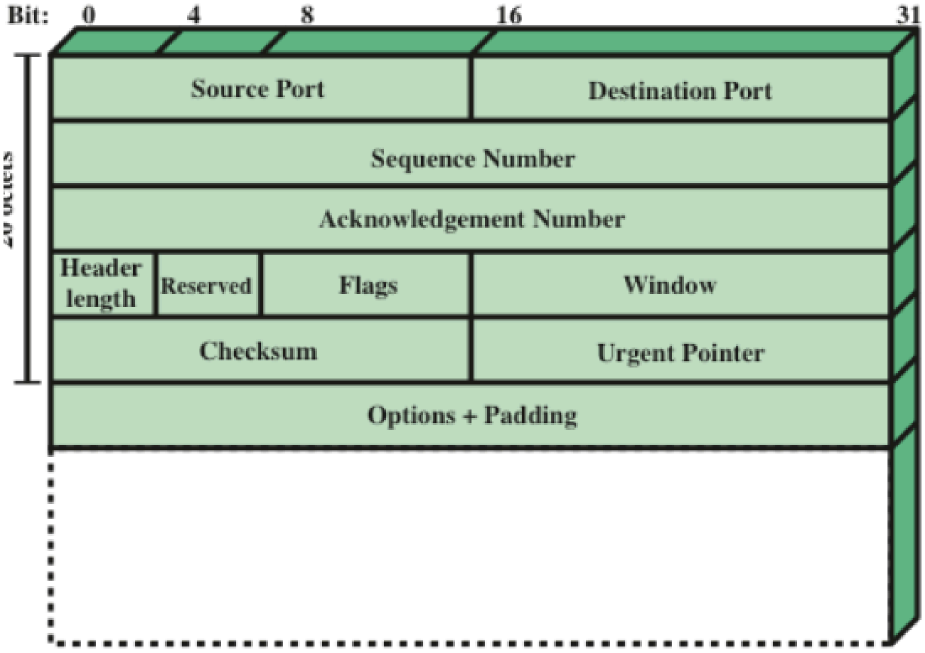


Figure 12.3 TCP Segment

MAC Properties

Different from signatures, many properties are similar to hash.

- **mac:= MAC(Key, message).**
  - Could be treat as a cryptographic checksum/digest
    - Takes an arbitrary length message as input , outputs a fixed length authenticator using a key.
- Like hash functions, it is a many-to-one function with Preimage resistance (PR).
  - For every key, it satisfies hash function properties.
- Sometimes, MAC is referred to as a family of Hash functions.

Attacks on MAC

1. Brute-force attack: The objective is to find a collision.
   a. Similar to Hash functions, MAC has to have a certain length to defeat brute-force attacks.
2. Two approaches for cryptanalysis:
   a. Attacker may first determine the key, then he can produce MAC value for any message.

   b. Sometimes, he may just try to determine a valid tag for a given message.

3. Normally create new MAC functions using existing Hash functions

HMAC

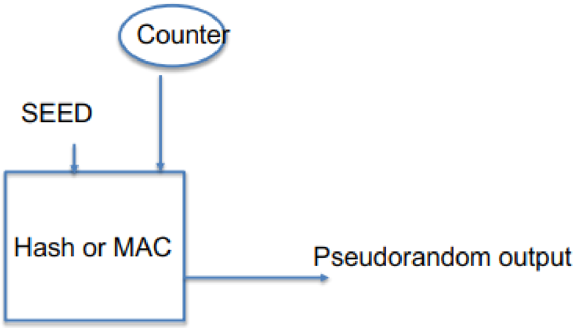
MACs based on hash functions are popular in the real world.

- Think of it as a keyed hash function.
    - One hash function has a MAC, a MAC has many hash functions.
- A simple proposal
    - KeyedHash = Hash(Key ||Message)

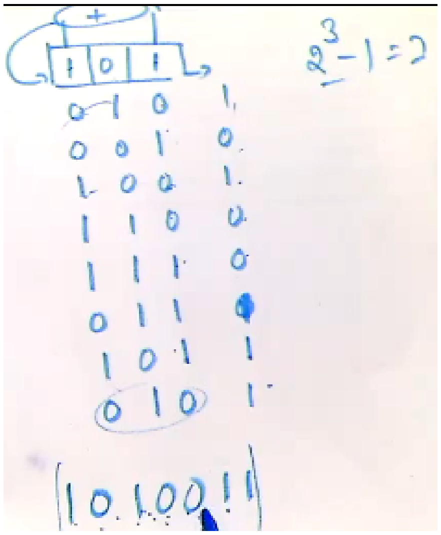
Pseudorandom Generation

As opposed to random numbers, pseudo random number generator takes a seed value as input and generates a sequence of digits.

- Like hash, for the same seed value it generates the same sequence.



Linear Feedback Shift Register(LFSR)



10100011 - 1　　　　11000101 - 1

Class 13

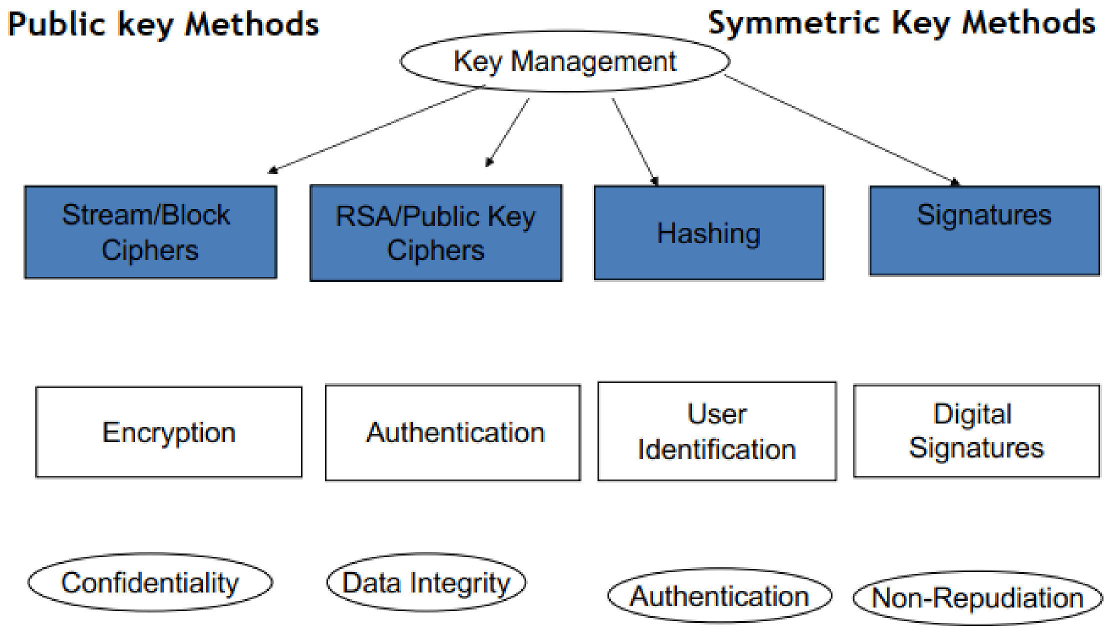
Midterm

Class 14

Key Management & Distribution

Most cryptographic algorithms except Hash functions involve keys.

- Key distribution is all about generation, distribution, storage, archival, …. of keys (symmetric and public).
- Without proper and secure key management, cryptographic algorithms are useless.



Keys & Attacks

Keys are to be formed using purely random sources.

- but in practice, they are usually pseudo random based on some secret seeds or random seeds obtained from physical means.
- Long life keys should be generated from a truly random source
    - examples: Thermal noise, time between keystrokes etc.
- Basic idea of Pseudo-random Bit Generators
    - Extend truly generated random number of length k to length t, where t > k.

- Each key's life cycle functions provide opportunities for attackers.
  - Generation or Creation
  - Distribution
  - Storage and Maintenance
  - Revocation/ Disposal

Symmetric Key Distribution

The security of Symmetric key systems is based on the requirements:

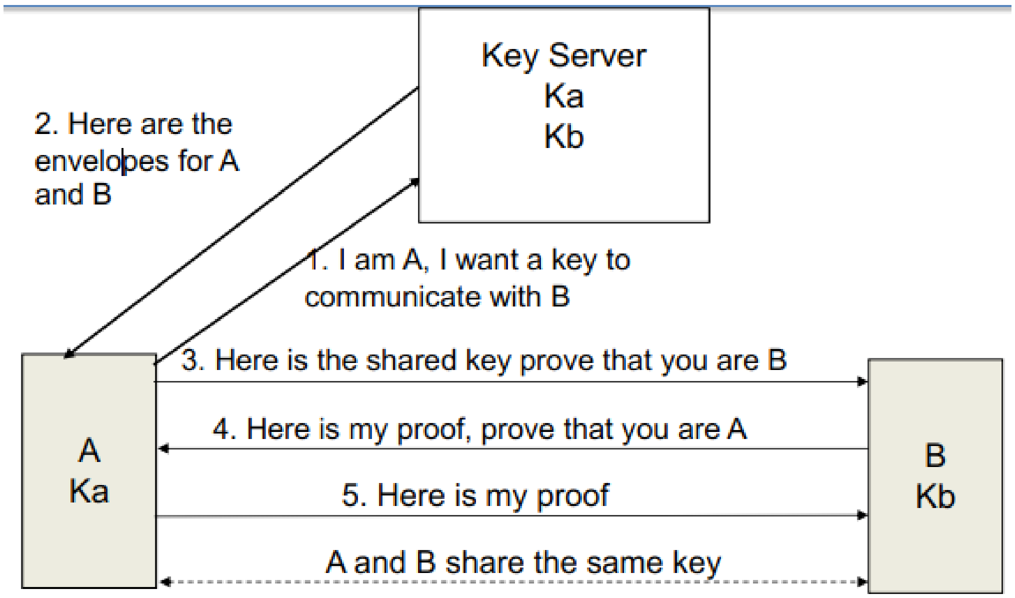
1. The encryption algorithm does not have any weakness.
2. The secret key is private to the sender and receiver.

Methods of key distribution

- The users can meet in advance is impracticable as the users are heterogeneous in communication networks.
  - Need some sort of key distribution framework
    - the keys need to be private between users and their access must be denied to others in the network.
- The keys need to be frequently changed to minimize the risk of attack.
  - In practice, failures in secure systems are the result of vulnerabilities in key management rather than the weaknesses of encryption functions.
- Assume N users.
  - Direct sharing between nodes : Complexity of initial key installation : O(N^2)
  - Public Key Algorithmic Method (Diffie-Hellman Protocols or using RSA) : O(N)
  - Using a Key server (O(N))
- Four alternatives:
  - A can select key and physically deliver to B
  - A third party can select & deliver key to A & B
  - If A & B have communicated previously can use previous key to encrypt a new key
  - If A & B have secure communications with a third party C, C can relay key between A & B

Key Distribution using a Key Server

1. I am A, I want a key to communicate with B
2. Here are the envelopes for A and B
3. Here is the shared key prove that you are B
4. Here is my proof, prove that you are A
5. Here is my proof
6. A and B share the same key

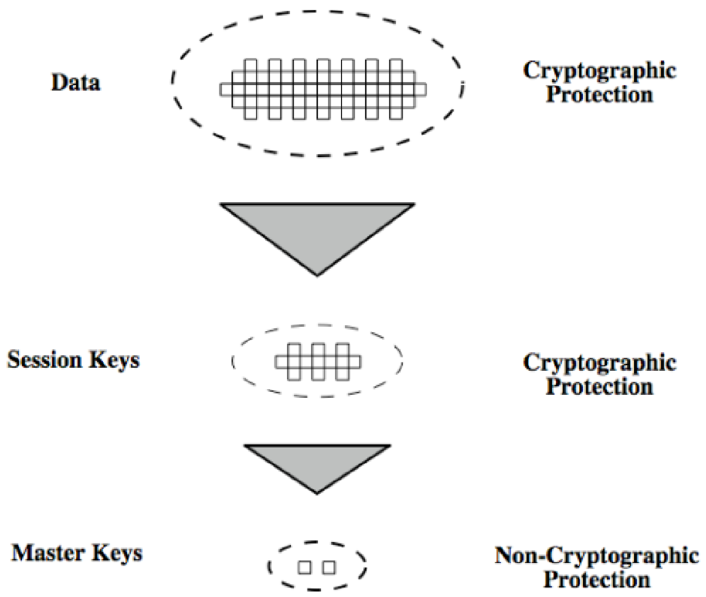

Key Server
Ka
Kb
2. Here are the envelopes for A and B
1. I am A, I want a key to communicate with B
3. Here is the shared key prove that you are B
4. Here is my proof, prove that you are A
5. Here is my proof
A and B share the same key
A
Ka
B
Kb

Server Based Protocol

1. Needham-Schroeder Protocol
   - Problems with freshness of session keys
2. Otway-Rees Protocol
   - Use active authentication to avoid problems with using old keys

Key Hierarchy

- Session key
  - temporary key used for encryption of data between users
  - for one logical session then discarded
- Master key
  - used to encrypt session keys shared by user & key distribution centre



Key distribution method can be extended to multiple KDCs, a local KDC and a global KDC.

- You can have a hierarchy of KDCs.
- Users within a same local domain are supported by the local server,
- Users in two different domains will need to involve global KDC to exchange keys.

Hierarchy of keys minimizes complexity of key distribution. Also localizes the risk of fault or compromise within a local domain.

- A fully decentralized key control may require every node to establish a master key with every other node, thus needing n(n-1)/2 keys for an end point system.
- Control based on nature of the keys may help to identify different types of keys:
    - Data Encrypting keys (DEK) for general communication across networks
    - PIN-encrypting Key for PINS and Electronic transfer applications
    - File Encrypting Key, for accessing files in public locations.

Ppt. 20/31