PESU Center for
Information Security,
Forensics and
Cyber Resilience

PES
UNIVERSITY

Welcome to

PES University

Ring Road Campus, Bengaluru

PESU Center for
Information Security,
Forensics and
Cyber Resilience

PES
UNIVERSITY

APPLIED CRYPTOGRAPHY

Lecture 25

Public (Asymmetric) Key Cryptography

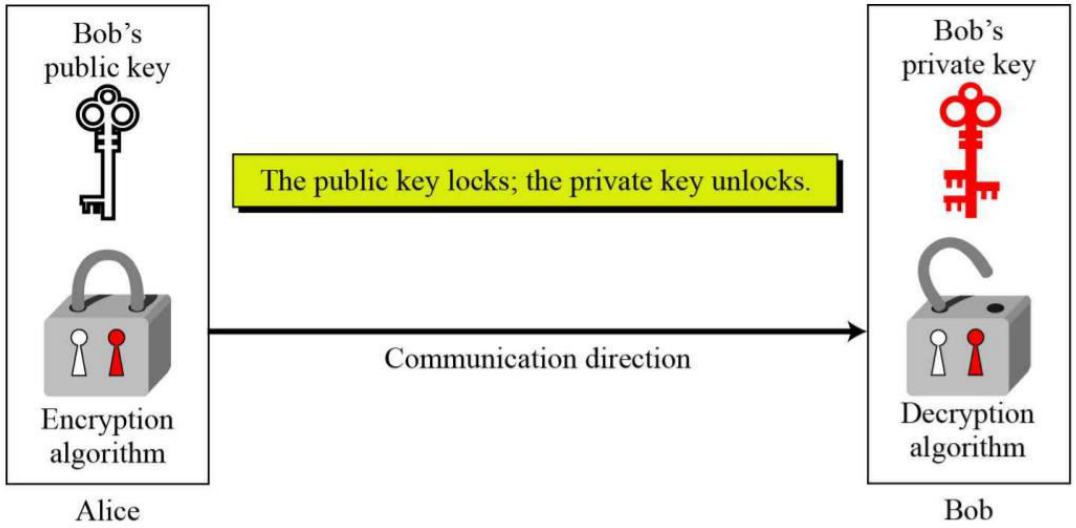
Pairs of Keys

Overview

- How do two entities establish shared secret key over network?
  - Trusted KDC (Key Distribution Center)
- Private and Public-key cryptography will exist in parallel and continue to serve the community.
- We actually believe that they are complements of each other.
- The advantages of one can compensate for the disadvantages of the other.

Establishing a shared secret

Goal: Alice and Bob want shared secret, unknown to eavesdropper. Security against eavesdropping (No tampering).



Pair of Keys

- Generated by the user (receiver) himself.
- A **public-key**, which may be known by anybody, and can be used to encrypt messages, and verify signatures.
- **A private-key**, known only to the recipient, used to decrypt messages, and sign (create) signatures.
- Called **Asymmetric** because those who encrypt messages or verify signatures **cannot** decrypt messages or create signatures.

Why Public-Key Cryptography?

Developed to address two key issues:

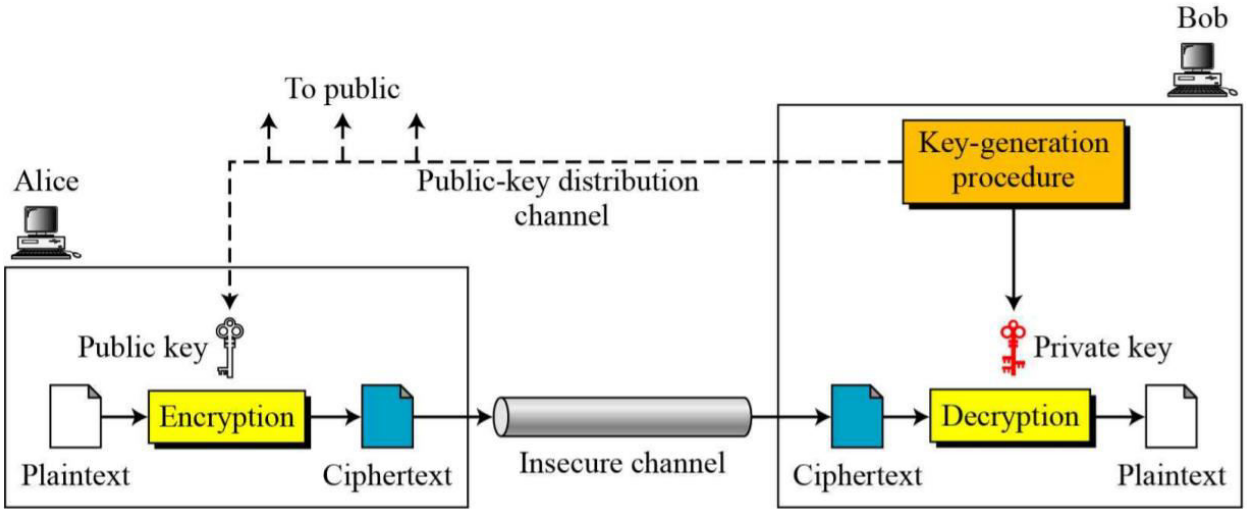
- ☞ **Key distribution** - how to have secure communications in general without having to trust a KDC with your key.
- ☞ **Digital signatures (authentication)** - how to verify a message comes intact from the claimed sender.

- Invented by (Stanford Univ., 1976):
  - Whitefield Diffie
  - Martin Hellman
  - Ralph Merkle

Public-Key Characteristics

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

General Idea



- Plaintext
- Encryption alg.
- Public & Private key
- Ciphertext
- Decryption alG.

- The burden of providing security is mostly on the shoulders of the receiver (Bob, in this case).
- Bob needs to create 2 keys: one private & one public. Bob is responsible for distributing the public key to the community.

Continued..

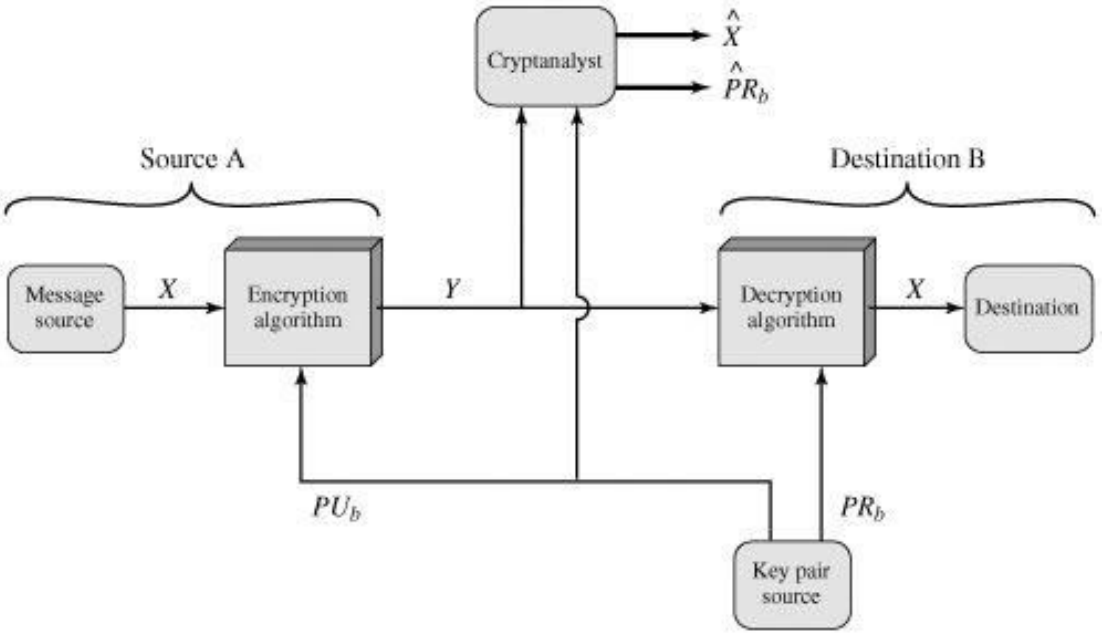
Plaintext/Ciphertext

- Unlike in symmetric-key cryptography, plaintext and ciphertext are treated as integers in asymmetric-key cryptography.

Encryption/Decryption

$$C = f(K_{public}, P), P = g(K_{private}, C)$$

Public-Key Cryptosystem: Secrecy (Confidentiality)

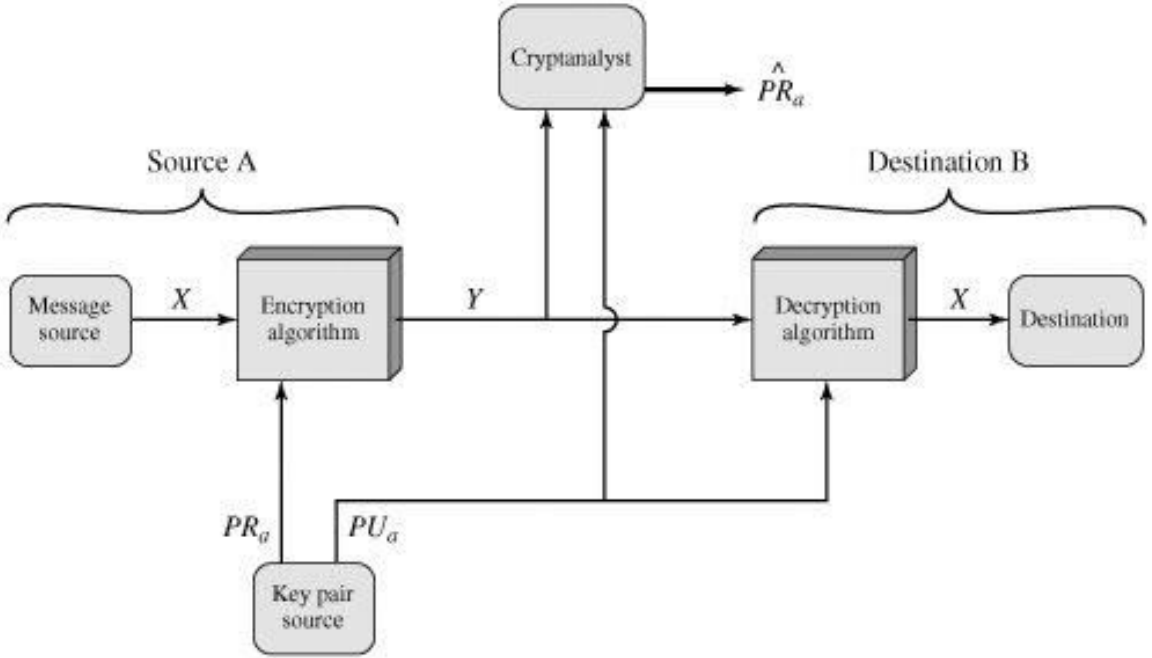


$Y = E(PU_b, X)$

$X = D(PR_b, Y)$

An adversary, observing Y and having access to PU_b but not having access to PR_b or X, must attempt to recover X and/or PR_b.

Public-Key Cryptosystem: Authentication

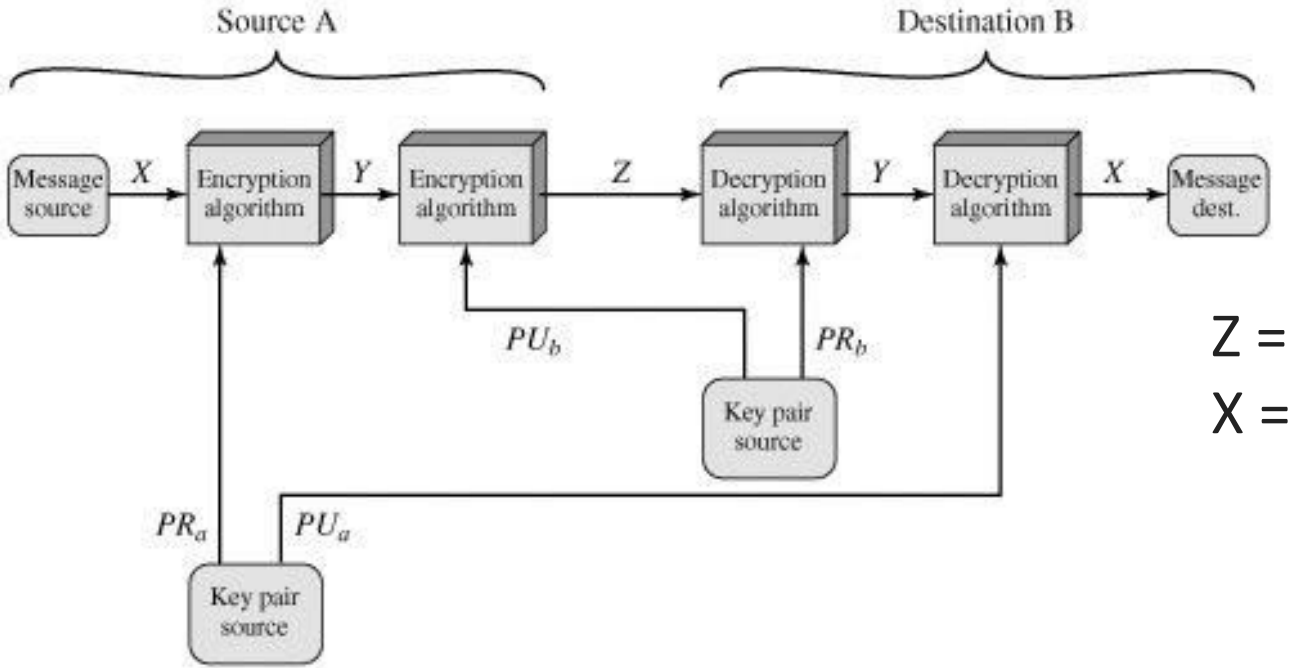


Y = E(PRa, X)

Y = E(PUa, Y)

It is impossible to alter the message without access to A's private key, so the message is authenticated both in terms of source and in terms of data integrity.

Public-Key Cryptosystem: Authentication & Secrecy



Z = E(PUb, E(PRa, X))

X = D(PUa, E(PRb, Z))

Drawback: This public-key algorithm is complex, must be exercised four times rather than two in each communication.

Requirements for Public-Key Cryptography

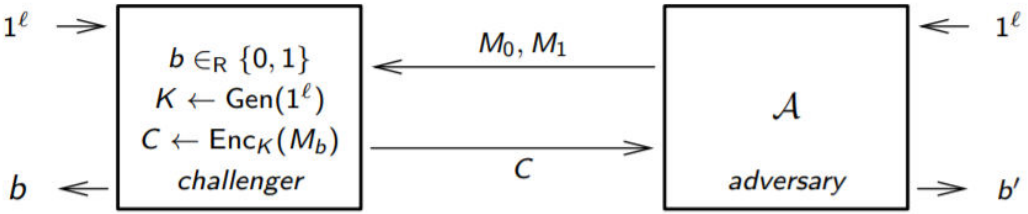
1) It is computationally easy for a party B to generate a pair (public key PU_b, private key PR_b).
2) It is computationally easy for a sender A, knowing the public key and the message to be encrypted, M, to generate the corresponding ciphertext: $C = E(PU_b, M)$.
3) It is computationally easy for the receiver B to decrypt the resulting ciphertext using the private key to recover the original message: $M = D(PR_b, C) = D[PR_b, E(PU_b, M)]$.
4) It is computationally infeasible for an adversary, knowing the public key, PU_b, to determine the private key, PR_b.
5) It is computationally infeasible for an adversary, knowing the public key, PU_b, and a ciphertext, C, to recover the original message, M.

Public-key Encryption Scheme

- Definition: A public-key encryption scheme is a tuple of probabilistic polynomial-time (PPT) algorithms (Gen, Enc, Dec) such that:
  1) The key generation algorithm Gen takes as input the security parameter 1^n and outputs a pair of keys (pk,sk) with $|pk| = n = |sk|$. We refer to these as the public key and the private key, respectively.
  2) The encryption algorithm Enc takes as input a public key pk and a message m from some underlying plaintext space. It outputs a ciphertext c;
     we write $c \leftarrow Enc_{pk}(m)$.
  3) The decryption algorithm Dec: takes as input a private key sk and a ciphertext c, and outputs a message m or a special symbol $\perp$ denoting failure.
- Standard Correctness Requirement, except with negligible probability, for all m and pk, sk output by Gen, $\mathbf{Dec_{sk}(Enc_{pk}(m)) = m}$

The eavesdropping indistinguishability experiment

- Given a public-key encryption scheme ∏ = (Gen, Enc, Dec) and an adversary *A*, consider the following experiment:
- The eavesdropping indistinguishability experiment $PubK^{eav}_{A,\Pi}(n)$:
  1) $Gen(1^n)$ is run to obtain keys (*pk*,*sk*).
  2) Adversary *A* is given *pk*, and outputs a pair of messages m_0, m_1 of the same length.
  3) A random bit *b* <- {0, 1} is chosen, and then a ciphertext *c* $Enc_{pk}(m_b)$ is computed and given to *A*. We call *c* the challenge ciphertext.
  4) *A* outputs a bit *b'*
  5) The output of the otherwise.



CPA-security

- A public-key encryption scheme $PubK^{eav}_{A,\prod}(n)$ has indistinguishable encryptions in the presence of an eavesdropper if for all PPT adversaries *A* there exists a negligible function negl such that:

$$Pr[PubK^{eav}_{A,\prod}(n) = 1] \leq 1/2 + negl(n).$$

- In other words: as we increase the security parameter n, we quickly reach the point where no eavesdropper can do significantly better just randomly guessing b.

Private vs Public key Cryptography

Private Key Encryption	Public Key Encryption
Uses one key for encryption and decryption.	Uses two keys(public and private keys) for encryption and decryption.
Key must be kept secret.	One key must be kept secret and other can be freely exposed.
Low power consumption.	Higher power consumption.
Speed in performance.	Slow in performance.
Inexpensive to generate.	Expensive to generate.

Continued ..

Private Key Encryption	Public Key Encryption
Randomly generated k-bits strings.	Have special structures Ex: Large prime numbers
Best used for secrecy and integrity of data.	Best used for key exchange and authentication.
Key distribution is problematic.	Key distribution is safer.
Example: • AES • DES, Triple DES	Example: • Elliptic Curve Cryptography • RSA algorithm • Diffie-Hellman key exchange • Digital Signature algorithm

Public-Key Applications

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

Applications: Public-Key Cryptosystems

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

Security of Public-Key Cryptography

- Like private key schemes brute force **exhaustive search** attack is always theoretically possible.
- But keys used are too large (>512bits)
- Security relies on a **large enough** difference in difficult between **easy** problem (en/decryption) and **hard** problems (cryptanalysis).
- More generally the **hard** problem is known, but is made hard enough to be impractical to break.
- Requires the use of **very large numbers** (>512bits)**.**
- Hence is **slow** compared to private key schemes.

Thank You!

Next Class

- Mandatory reading for the next class
  - https://math.berkeley.edu/~kpmann/encryption.pdf

Dr. Sivaraman E

Computer Science and Engineering
PES University, Bengaluru

PESU ISFCR
PESU Center for Information Security, Forensics and Cyber Resilience

PESU IoT
PESU Center for Internet of Things

PESU Center for Information Security, Forensics and Cyber Resilience

PES UNIVERSITY

Welcome to

PES University

Ring Road Campus, Bengaluru

PESU Center for
Information Security,
Forensics and
Cyber Resilience

PES
UNIVERSITY

APPLIED CRYPTOGRAPHY

Lecture 26

Modes of Operation

More Security through Input

Outline

- Introduction
- Types of Modes of Operation
- Stream Cipher Modes of Operation
- Block Cipher Modes of Operation
- Comparison

Overview

- Block length is fixed (n-bit)
- A block cipher such as DES or AES only encrypts one block of a fixed size (64-bit for DES and 128-bit for AES).
- How to encrypt (the plaintext) larger than 64-bit or 128-bit?
- It is solved by using a mode of operation - to repeatedly apply DES or AES (both encryption and decryption) on plaintexts (and ciphertexts) with size larger than one block.

Introduction

- A mode of operation is a way of encrypting arbitrary-length messages using a block cipher. (i.e., Pseudo Random Permutation-PRP)
- Configuration methods that allow ciphers to work with large data streams.

Types of Modes of Operations

Although the 5 modes of operations enable the use of block ciphers for encipherment of messages or files in large units and small units, sometimes pure stream are needed for enciphering small units of data such as characters or bits.

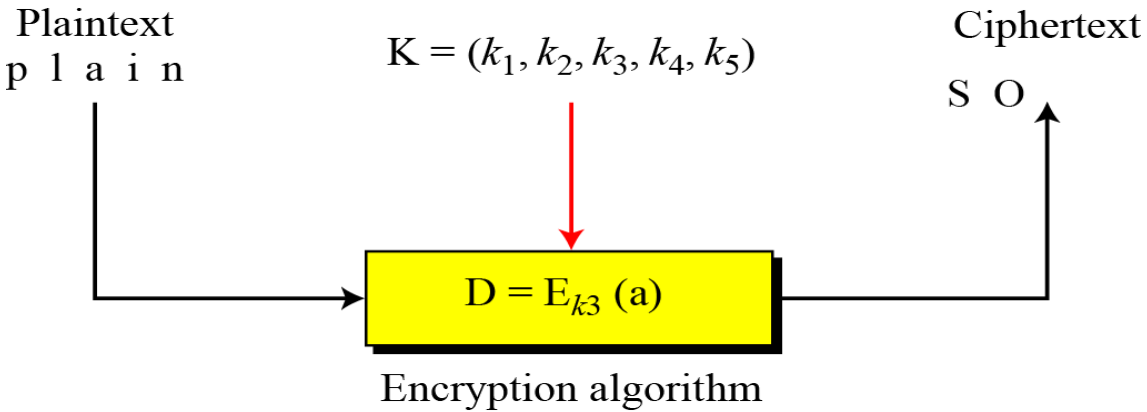
- Stream Cipher
  - Synchronized
  - Unsynchronized
- Block Cipher
  - Electronic Codebook Mode (ECB)
  - Cipher-Block Chaining (CBC)
  - Cipher Feedback Mode (CFB)
  - Output Feedback Mode(OFB)
  - Counter Mode (CTR)

Stream Cipher

$P = P_1P_2P_3, \ldots \qquad C = C_1C_2C_3, \ldots \qquad K = (k_1, k_2, k_3, \ldots)$

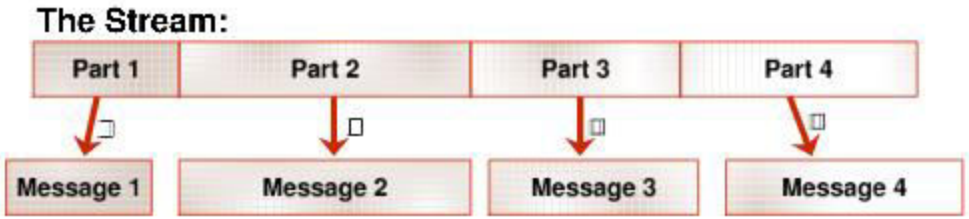
$C_1 = E_{k1}(P_1) \quad C_2 = E_{k2}(P_2) \quad C_3 = E_{k3}(P_3) \ldots$

Call the plaintext stream P, the ciphertext stream C, and the key stream K.



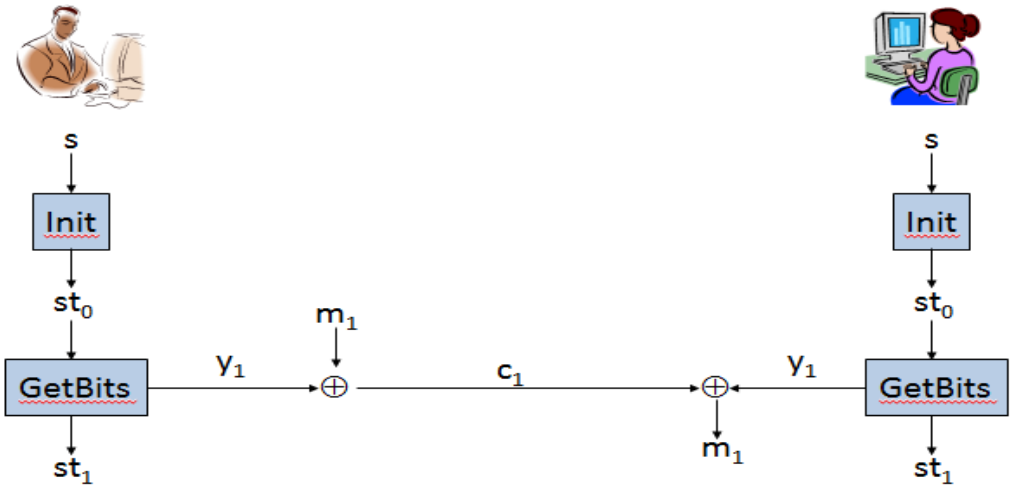
Synchronized mode

- Sender and receiver maintain state (i.e., they are stateful), and must be *synchronized.*
  - Makes sense in the context of a limited-time communication session where messages are received in order, without being lost.



Continued..

- This mode is “synchronized“ because both parties need to know which parts of the stream have already been used in order to prevent reuse, which is not secure.

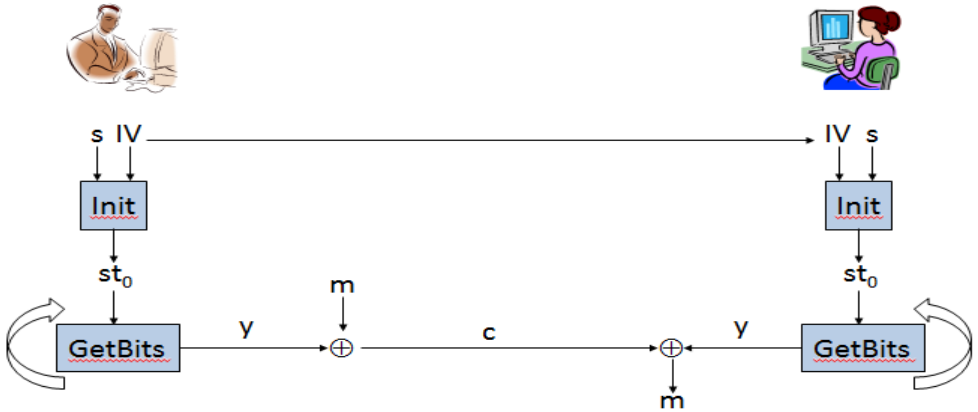


Unsynchronized mode

- Encryptions are carried out independently of one another and the parties do not need to maintain state.
- Choose random IV to encrypt next message.

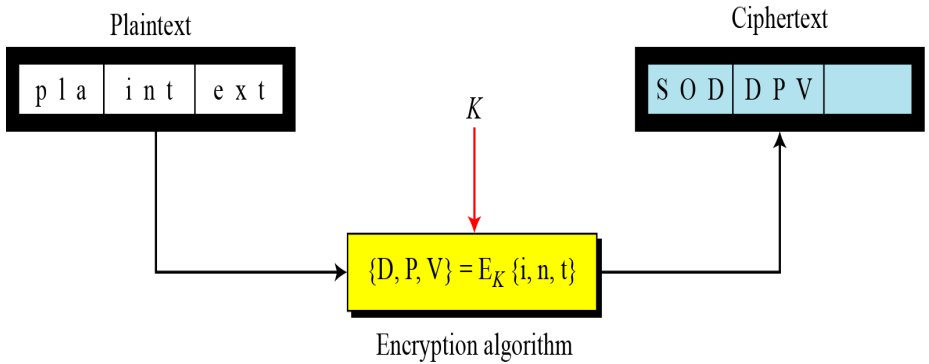
Continued..

- In order to achieve security, pseudorandom generator must be significantly strengthened.
- Pseudorandom generator is taking two inputs: a seed sand an initial vector IV of length n.



Block Cipher

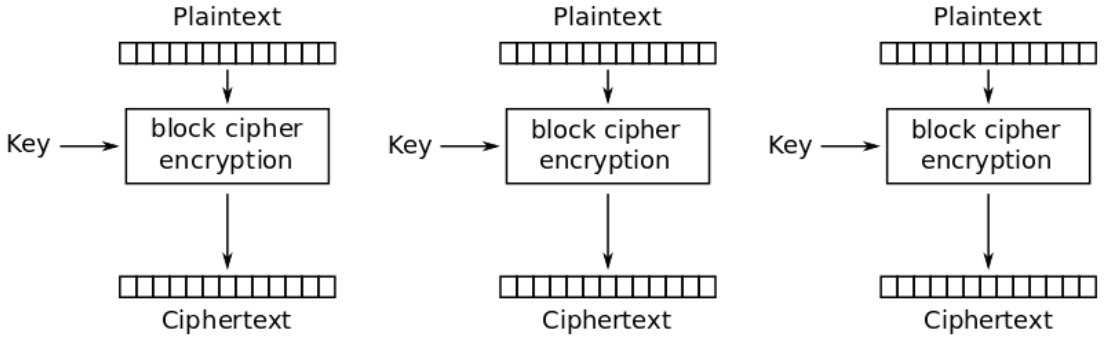
- In a block cipher, a group of plaintext symbols of size, m (m > 1) are encrypted together creating a group of ciphertext of the same size.
- A single key is used to encrypt the whole block even if the key is made of multiple values.



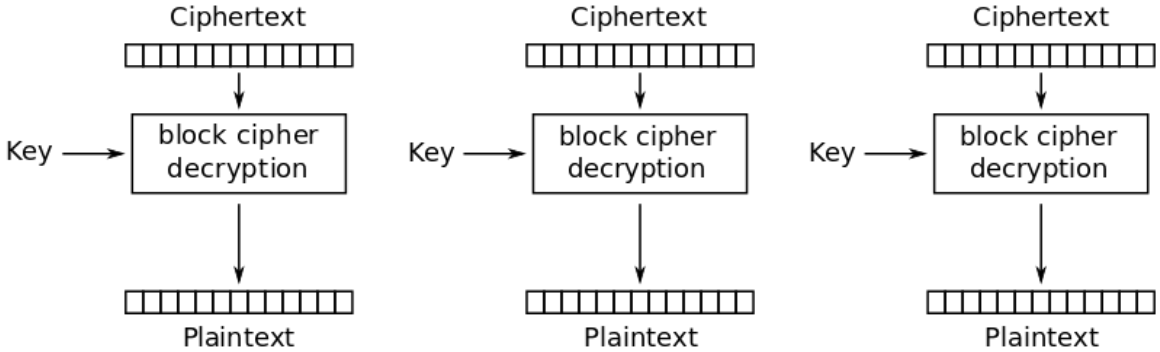
Electronic Codebook (ECB) mode

- It is the simplest mode of encryption.
- Message is broken into independent blocks (book) that are encrypted.
- Each plaintext block is encrypted independently.
- Similarly, each ciphertext block is decrypted separately.
- Thus, it is possible to encrypt and decrypt by using many threads simultaneously.
- However, in this mode the created ciphertext is not blurred.

Electronic Codebook (ECB) mode



Encryption



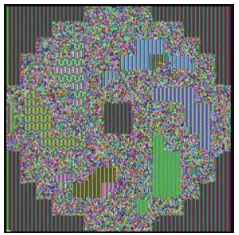
Decryption

Advantages and Limitations of ECB

- No block sync. required between sender and receiver.
- No error propagation, can be parallelized.

- Identical plaintexts results in identical ciphertexts.
- Does not hide data patterns, unsuitable for long messages.

->

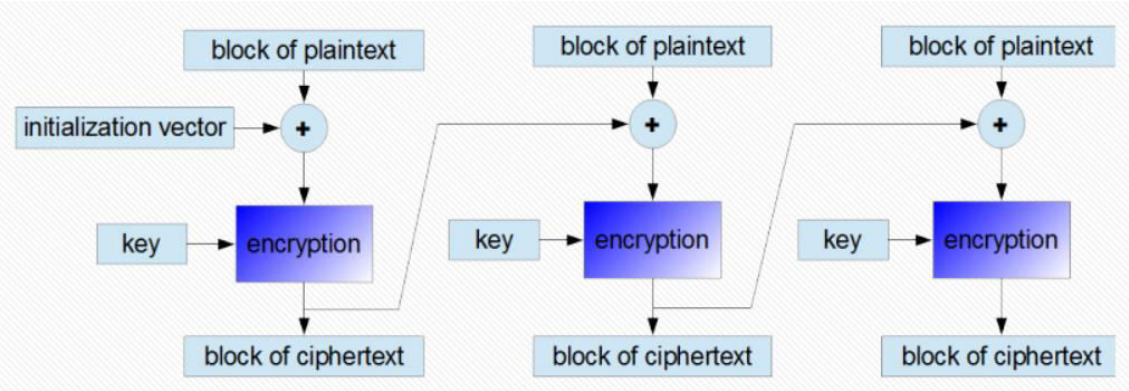


- Weakness due to encrypted message blocks being independent.
- Susceptible to replay attacks.

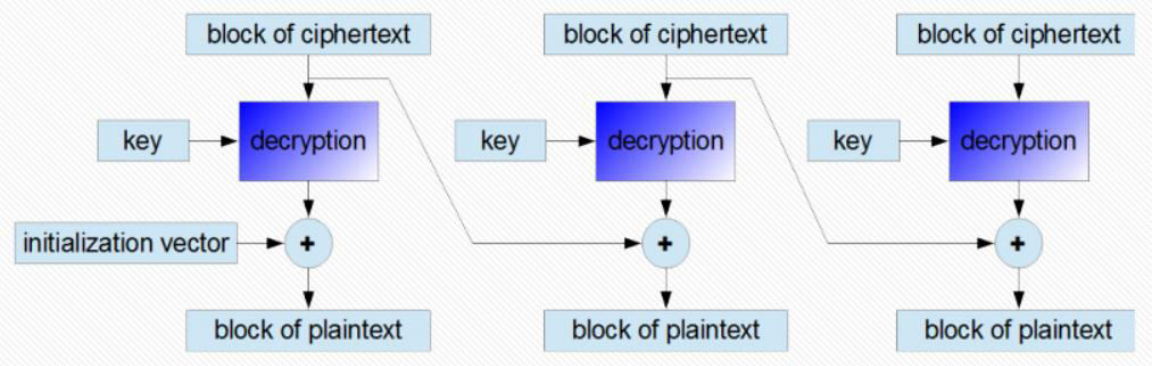
Cipher Block Chaining (CBC) mode

- To overcome block independence in ECB.
- Message is broken into blocks, linked/chained together in encryption operation.
- Add (XOR) current plaintext block to the ciphertext block that was previously produced.
- First plaintext block is added (XOR) to **Initialization Vector (IV),** same size as the plaintext block.
- Performed by using one thread.
- Uses: Bulk data encryption, authentication

Cipher Block Chaining (CBC) mode



Encryption



Decryption

Advantages and Limitations of CBC

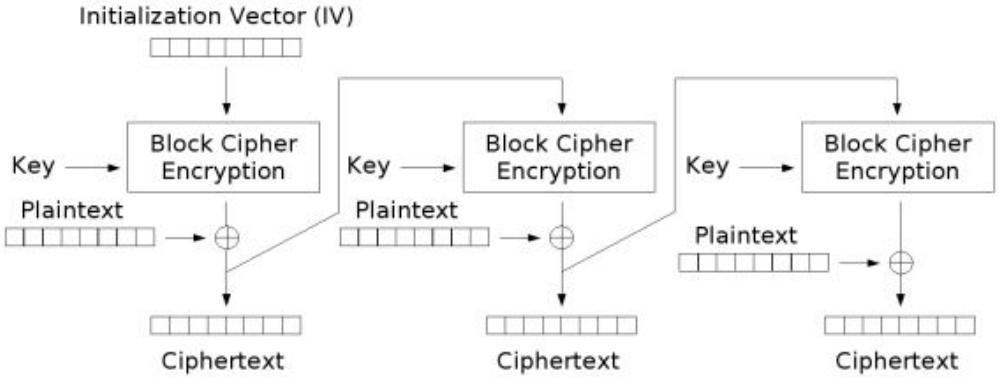
- No identical messages.

- Each ciphertext block depends on all previous message blocks.
- Thus, any change to a block (message) affects all following ciphertext blocks - Avalanche effect.
- Need Initialization Vector (IV) which must be known to sender & receiver in addition to the key.

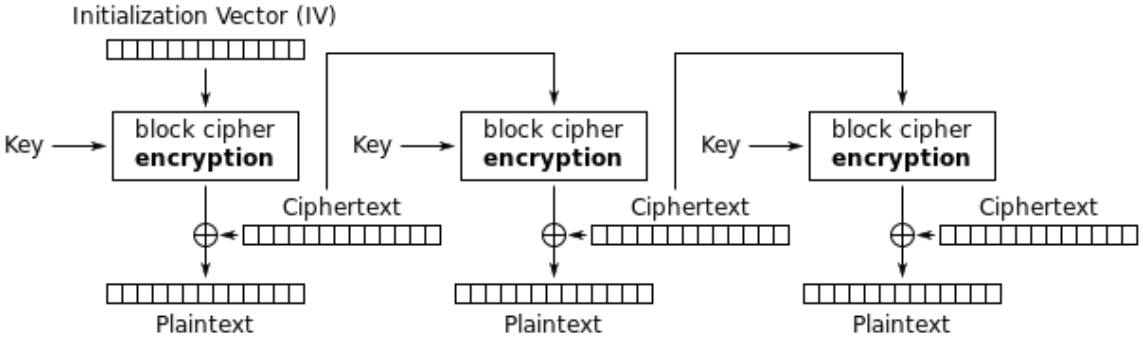
Cipher Feedback (CFB) mode

- Stream Mode – Simulates a stream (bit by bit)
- Operates on segments (a stream of bits) instead of blocks.
- Segment length (called s) – multiples of 8 bits.
- The result is feedback for next stage.
- IV is used.
- Standard allows any number of bits (1,8, 64 or 128 etc) to be feedback, denoted CFB-1, CFB-8, CFB-64, CFB-128, etc.
- Most efficient to use all bits in block (CFB-64)

Cipher Feedback (CFB) mode

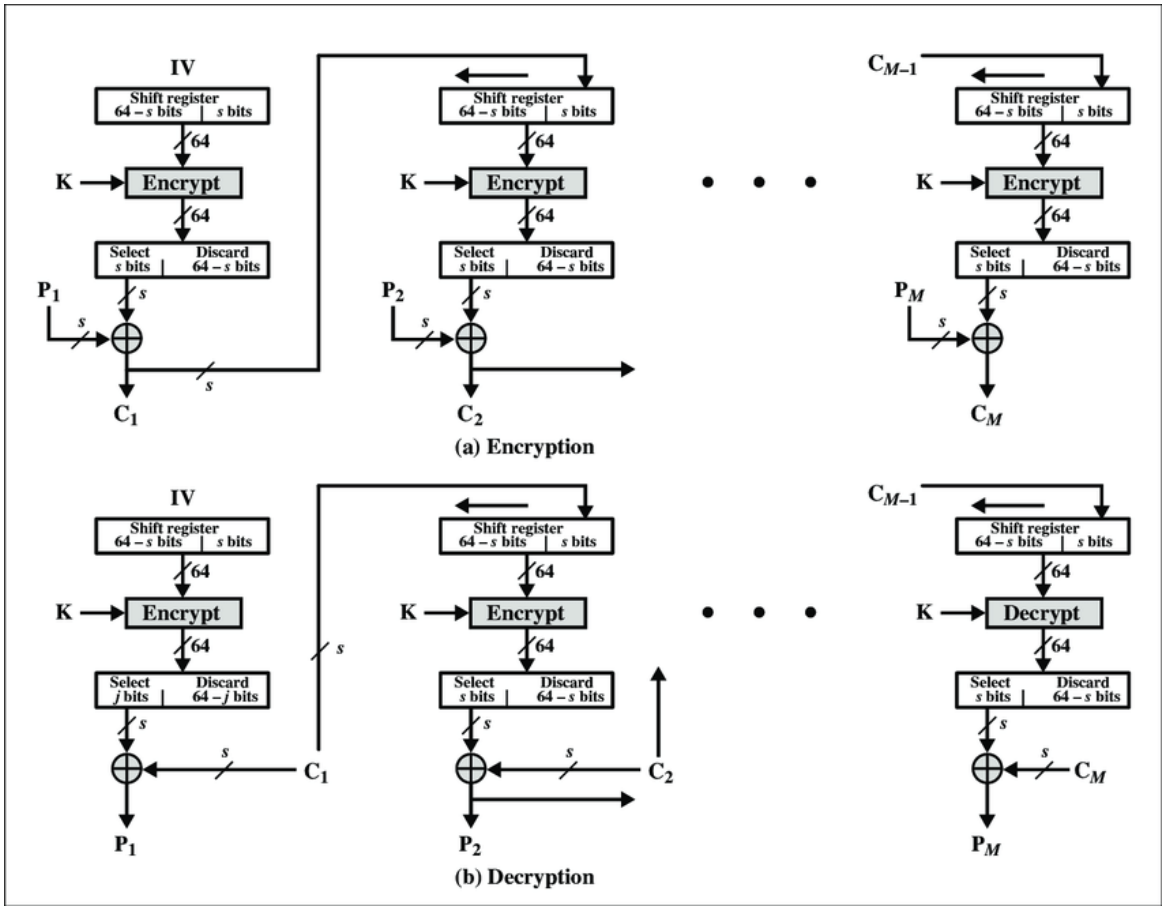


Encryption



Decryption

Cipher Feedback (CFB) mode



Advantages and Limitations of CFB

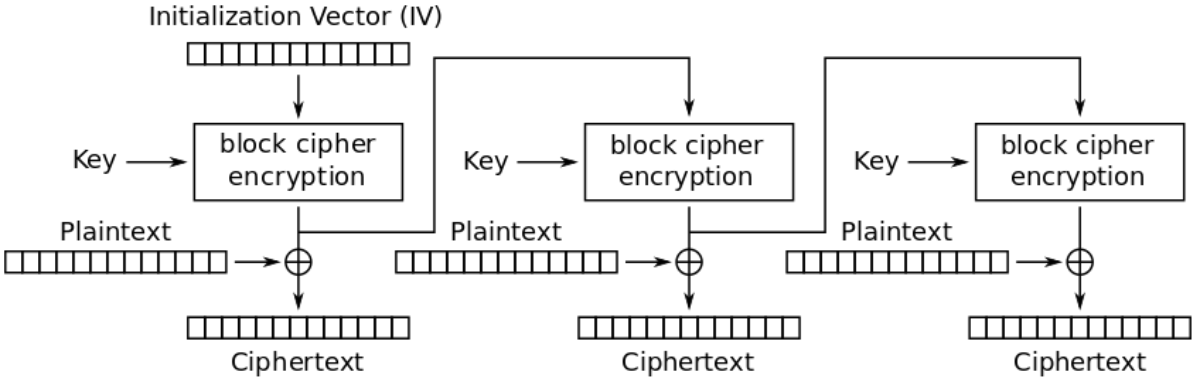
- Most common stream mode and appropriate when data arrives in bits/bytes.

- Errors propagate for several blocks after an error.
- Encryption in CFB mode can be performed only by using one thread.
- The block cipher is used in **encryption** mode at **both** ends (XOR).

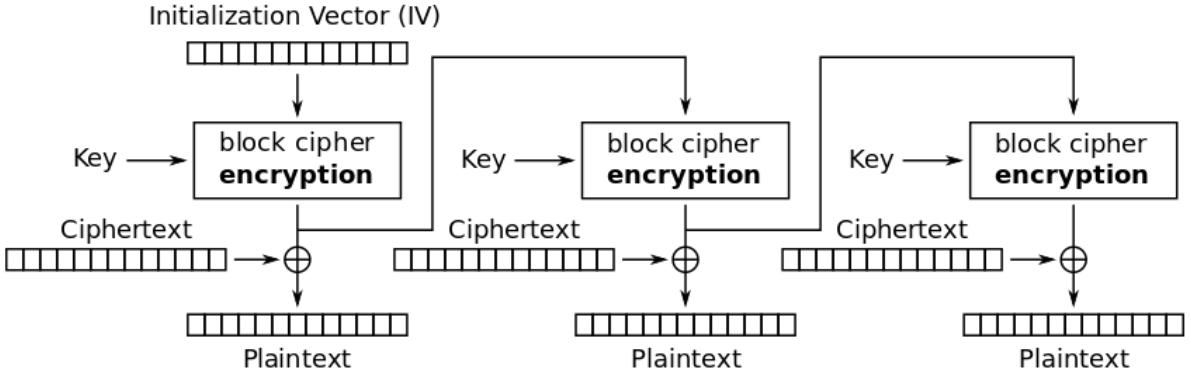
Output FeedBack (OFB) mode

- Used to build a synchronous **stream cipher** from a block cipher.
- The key stream is not generated bitwise but instead in a blockwise fashion.
- The output of the cipher gives us key stream bits S_i with which we can encrypt plaintext bits using the XOR operation.

Output FeedBack (OFB) mode



Encryption



Decryption

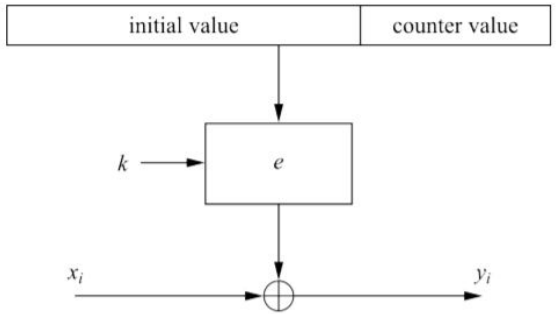
Advantages and Limitations of OFB mode

- No chaining dependencies.

- Needs an IV which is unique for each use. If ever reused, attacker can recover outputs.
- Errors do not propagate.
- More vulnerable to message stream modification, change arbitrary bits by changing ciphertext.

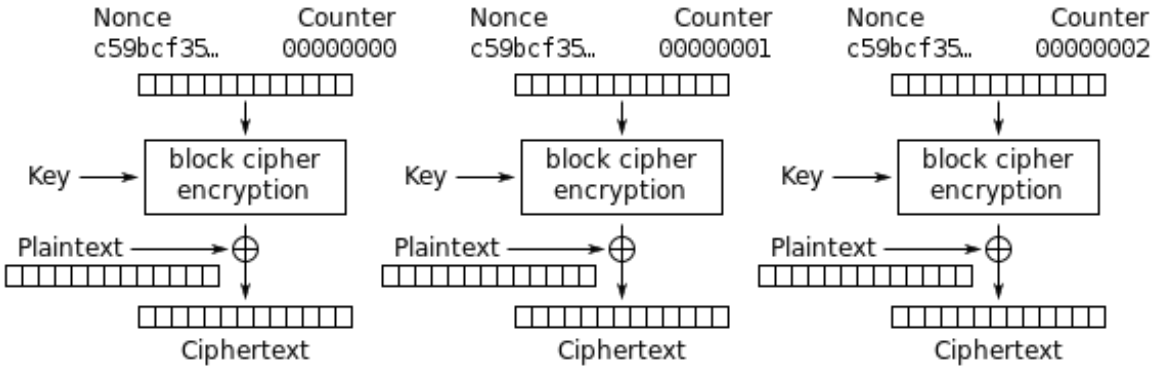
Counter Mode (CTR)

- It uses a block cipher as a stream cipher (like OFB & CFB).
- The key stream is computed in a blockwise fashion.
- The input to the block cipher is a counter which assumes a different value every time the block cipher computes a new key stream block.



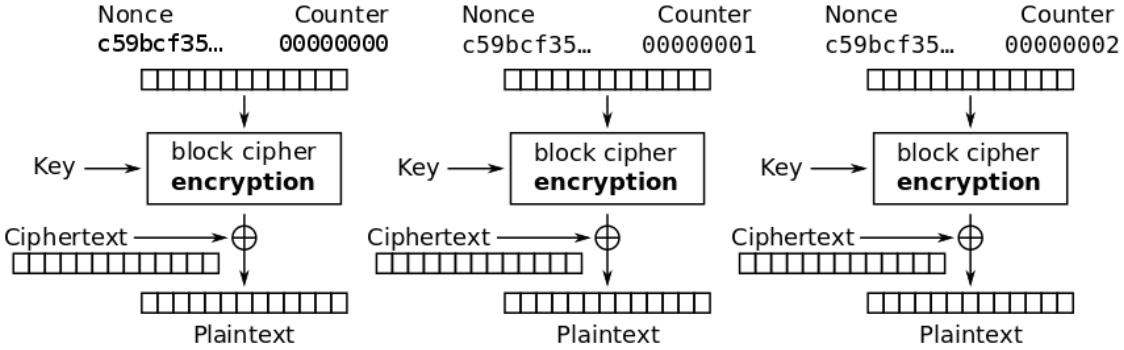
- Unlike CFB and OFB modes, the CTR mode can be parallelized.

Counter Mode (CTR)



Encryption

Nonce – one-time unpredictable value



Decryption

Advantages and Limitations of CTR mode

- Efficiency - Can do parallel en/decryptions.
- Can pre-process in advance of need.
- Good for bursty high speed links.
- Random access to encrypted data blocks.
- Provable security (good as other modes).

Comparison of Different Modes

Mode:	Attr1	Attr2	Attr3	Xor
ECB	No I.V. (nonce)	Short messg		no
CBC	i.v.	Chaining function	CT1 becomes i.v. for next block	yes
CFB	i.v.	Shift Reg+chaining	8 bit	yes
OFB	i.v.	encrypted i.v. as next i.v.		yes
CTR	Counter is i.v.	Counter + 1 is next i.v.		

Thank You!

Next Class

- ☞ Mandatory reading for the next class
  - ☞ https://tools.ietf.org/html/rfc3447

Dr. Sivaraman E

Computer Science and Engineering
PES University, Bengaluru

PESU ISFCR
PESU Center for Information Security, Forensics and Cyber Resilience

PESU IoT
PESU Center for Internet of Things

PESU Center for
Information Security,
Forensics and
Cyber Resilience

PES
UNIVERSITY

Welcome to

PES University

Ring Road Campus, Bengaluru

PESU Center for
Information Security,
Forensics and
Cyber Resilience

PES
UNIVERSITY

APPLIED CRYPTOGRAPHY

Lecture 27

Prime Numbers, Theorems, Primitive Roots

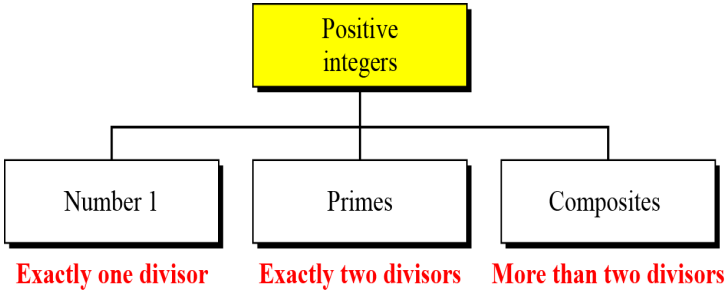
Divisible by 1 and itself

Overview

- To introduce prime numbers and their applications in cryptography.
- Asymmetric-key cryptography uses primes extensively.
- Finding an algorithm to correctly and efficiently test a very large integer and output a prime/a composite has always been a challenge in number theory, and consequently in cryptography.

Prime vs Composite Numbers

- Prime number: has exactly 2 divisors.
- Consider an integer N > 1, if ‘N’ is prime, then the divisors are 1 and N (Eg: 2,3,5). Otherwise, composite (Eg: 9, 33).
- It can’t be divided by other number without leaving a reminder.
- All numbers have prime factors.



Numbers	10	11	100	37	308	14688
Prime Factorization	$2^1 \times 5^1$	$1^1 \times 11^1$	$2^2 \times 5^2$	$1^1 \times 37^1$	$2^2 \times 7^1 \times 11^1$	$2^5 \times 3^3 \times 17^1$
Prime Numbers	2, 5	1, 11	2, 5	1, 37	2, 7, 11	2, 3, 17

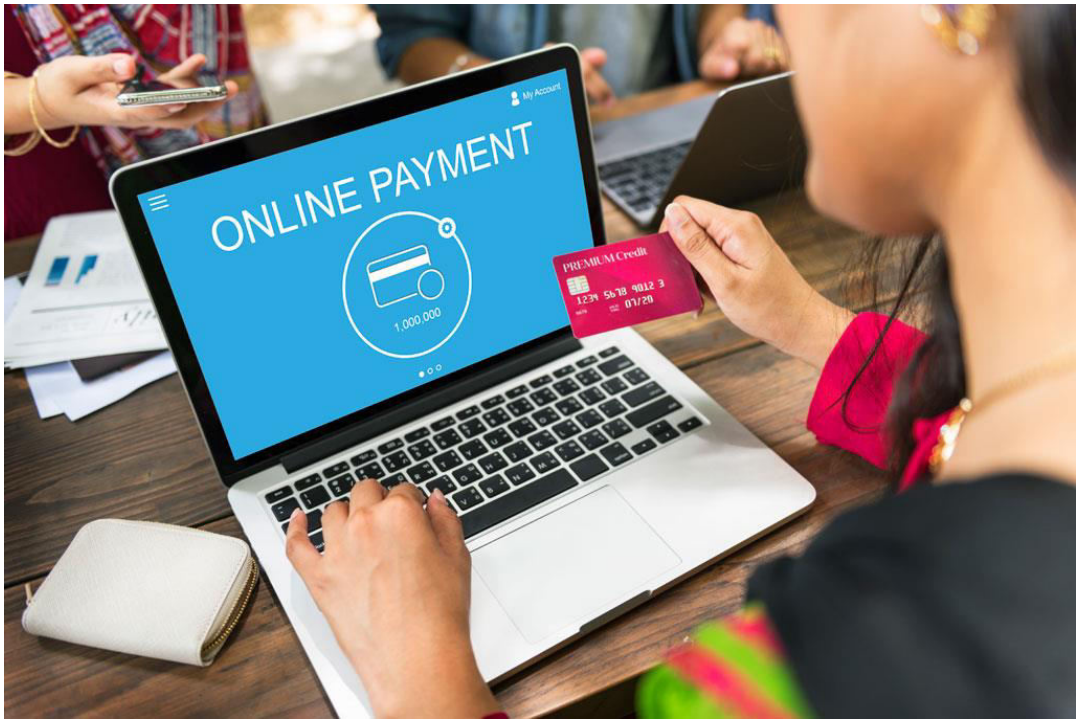
Facts about Prime Numbers

- Only even prime: 2
- Smallest prime: 2 (What about 1?)
- Except for 2 & 5, all prime numbers end in the digit 1, 3, 7 or 9.
- Hunt for Largest known prime number (called Massive Mersenne Primes) - $2^{82,589,933}-1$, having 24,862,048 digits (Dec 2018).
  - Mersenne primes take the form of 2 multiplied by itself a certain number of times, minus 1.

Why Prime Numbers in Cryptography?

- Many encryption algorithms are based on prime numbers.
- Very fast to multiply two large prime numbers.
- Extremely computer-intensive to do the reverse.
- Factoring very large prime is very hard, i.e., long time.
  - Eg: P1 = 709 & P2 = 733, multiple them, M = P1 * P2
  - M = 709 * 733 = 519697 (this is **easy** to compute)
  - A large number (**519697**) and the prime factorization (**709** * **733**)
  - Imagine, the prime factorization is hidden.
    519697 = ? * ? (this is **hard** to compute)
    To find the prime factorization, where would you begin?
- Check if a randomly generated number (**such as 9919485309475497**) is prime or composite.

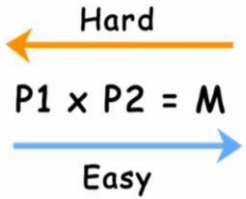
Why Prime Numbers in Cryptography?



How does online payment works?

Solution:

- M is public (shared by merchant).
- P1 & P2 are private (with the merchant).



Prime Numbers make the Internet Possible.

Relatively Prime

- Two integers are relatively prime (or co-prime), if they have only 1 as their common factor.
- For example:
  - 9 & 10 are relatively prime, but 12 & 14 are not.

Euler's phi-function

- Euler's phi-function, f(n), which is sometimes called the Euler's Totient function plays a very important role in cryptography.

1. $\phi(1) = 0$.
2. $\phi(p) = p - 1$ if p is a prime.
3. $\phi(m \times n) = \phi(m) \times \phi(n)$ if m and n are relatively prime.
4. $\phi(p^e) = p^e - p^{e-1}$ if p is a prime.

Examples

- Calculate:
  - $\phi(13)$
  - $\phi(10)$
  - $\phi(240)$

  - $\phi(49)$
  - Can we say that $f(49) = f(7) \times f(7) = 6 \times 6 = 36$?
  - $f(49) = (7)^2 = 49 - 7 = 42$

Examples

What is the value of $\phi(13)$?

What is the value of $\phi(10)$? $= \phi(5) \times \phi(2)$

$= 4 \times 1 = 4$

Examples

What is the value of f(240)?

$= 2^4 \times 3^1 \times 5^1$

Can we say that f(49) = f(7) × f(7) = 6 × 6 = 36?

$= p^e - p^{e-1}$

$= 7^2 - 7 = 42$

Solution is 1) 64

2) 42

Continued..

- We can combine the above four rules to find the value of f(n). For example, if n can be factored as $\mathbf{n = p_1^{e1} \times p_2^{e2} \times \ldots \times p_k^{ek}}$ then we combine the 3rd and 4th rule to find,

$$\phi(n) = (p_1^{e_1} - p_1^{e_1-1}) \times (p_2^{e_2} - p_2^{e_2-1}) \times \cdots \times (p_k^{e_k} - p_k^{e_k-1})$$

- The difficulty of finding f(n) depends on the difficulty of finding the factorization of n.

Question?

- Compute the result of:
  - $213^{522} \bmod 523 = ?$

Fermat's Little Theorem

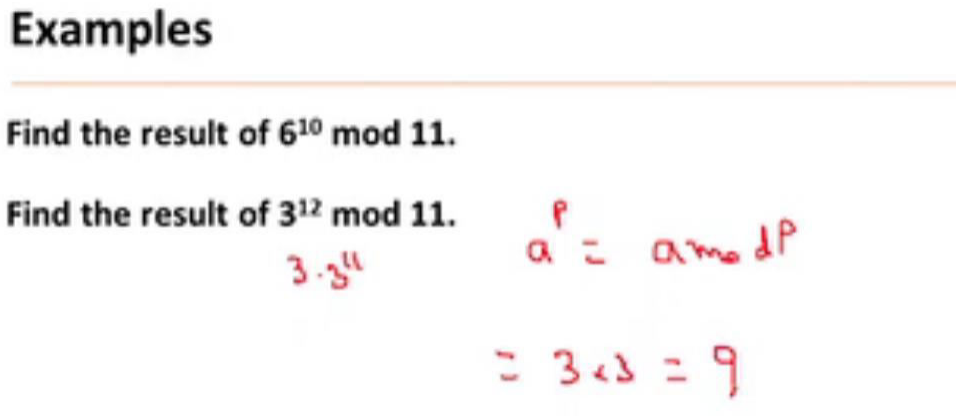
Two versions of the theorem.

- If p is a prime and a is a positive integer not divisible p, then $a^{p-1} \equiv 1 \bmod p = 1$

- If p is a prime and a is an integer, then $a^{p} \equiv a \bmod p = a$

Examples

- Find the result of:
  - 6^{10} mod 11
  - 3^{12} mod 11
  - 5^{15} mod 13



Euler's Theorem

First Version $a^{f(n)} \equiv 1 \pmod{n} = 1$

Second Version $a^{k*f(n)+1} \equiv a \pmod{n} = a$

- The second version of Euler's theorem is used in the RSA cryptosystem.

Examples

- Find the result of:
  - 6^{24} mod 35
  - 20^{62} mod 77

Euclidean Algorithm

- Finds the GCD of 2 numbers.
- EEA - Let a, b be positive integers. It also calculates the coefficients, x & y such that

  ax + by = gcd (a, b).

Pseudo Code of the Algorithm-

Step 1: **Let** `a, b` **be the two numbers**

Step 2: `a mod b = R`

Step 3: **Let** `a = b` **and** `b = R`

Step 4: **Repeat Steps 2 and 3 until** `a mod b` **is greater than 0**

Step 5: **GCD = b**

Step 6: Finish

Examples

- Calculate gcd (25,60).
- Let a=210 and b=45, calculate GCD.
  - Divide 210 by 45, and get the result 4 with remainder 30, so $210=4\cdot45+30$.
  - Divide 45 by 30, and get the result 1 with remainder 15, so $45=1\cdot30+15$.
  - Divide 30 by 15, and get the result 2 with remainder 0, so $30=2\cdot15+0$.

- The greatest common divisor of 210 & 45: 15.
- Calculate gcd (120,295)

Question?

- Compute the result of:
  - $2^5 = ?$
  - $3^? = 129140163$
    - Answer: 17
- We can also write,
  - $5 = \log_2(32)$ & $17 = \log_3(129140163)$
- In general,
  - $p_i = q$
  - $i = \log_p(q)$

Modular Arithmetic

- Compute the result of:

  $? = 3^4 \bmod 12$

  - Answer: 9
- What is the value of exponent i?

  $26067 = 14^i \pmod{29867}$

  - Answer: 12
  - To find i, we use discrete logarithm.
- We can also write,
  - $4 = \text{dlog}_{3,12}(9)$
  - $12 = \text{dlog}_{14,29867}(26067)$
- In general, $a = b^i \pmod m$, $i = \text{dlog}_{b,m}(a)$

Examples - Modular Arithmetic

- Compute the result of:

  $2^5 \bmod 3 = ?$

  - Answer: 2

  $4^4 \bmod 11 = ?$

  - Answer: 3

  $8 = 5^i \bmod 13$, find i (disc. log).

  $i = \text{dlog}_{5,13}(8)$

  - Answer: 3
  - $8 = 5^3 \bmod 13$

- Compute

  $? = 5^7 \bmod 13$ & $? = 5^{11} \bmod 13$

  - Answer: 8
  - i (discrete log), shouldn't have duplicates.

Primitive Roots

$a = b^i \pmod m$,

$i = \text{dlog}_{b,m}(a)$

- We can get a unique value of i, if b is a primitive root of prime modulo m. (i.e., ‘i’ will have distinct values)
- What is primitive root?

Primitive Roots (Contd…

$a = b^i \pmod m$, consider m = 13

b^{m-1}

b	b^1	b^2	b^3	b^4	b^5	b^6	b^7	b^8	b^9	b^{10}	b^{11}	b^{12}
1	1	1	1	1	1	1	1	1	1	1	1	1
2	2	4	8	3	6	12	11	9	5	10	7	1
3	3	9	1	3	9	1	3	9	1	3	9	1
4	4	3	12	9	10	1	4	3	12	9	10	1
5	5	12	8	1	5	12	8	1	5	12	8	1
6	6	10	8	9	2	12	7	3	5	4	11	1
7	7	10	5	9	11	12	6	3	8	4	2	1
8	8	12	5	1	8	12	5	1	8	12	5	1
9	9	3	1	9	3	1	9	3	1	9	3	1
10	10	9	12	3	4	1	10	9	12	3	4	1
11	11	4	5	3	7	12	2	9	8	10	6	1
12	12	1	12	1	12	1	12	1	12	1	12	1

Verifying all rows, row 2, 6, 7, 11 has no duplicates & hence, 2, 6, 7, 11 are the primitive roots of 13.

Total = 4

m-1

Problems

- Compute the result of:
  - $? = \text{dlog}_{17,2111}(1922)$
  - $1922 = 17^{i} (\text{mod } 2111)$
- Answer: 12

Example – Primitive Root Calculation

- Calculate the primitive root of 157 (prime number).
- Also, calculate number of primitive roots of a prime.
  - If any of the residues is 1, then ‘a’ is not a primitive root, otherwise primitive root.
  - 5 is the primitive root of 157

Primitive Root (in Diffie-Hellman)

Primitive root of a prime number n is an integer r between [1,n-1] such that the values of r^x(mod n) where x is in range[0,n-2] are different.

Table 9.5 shows the result of $a^i \equiv x \pmod 7$ for the group $G = \langle Z_7^*, \times \rangle$. In this group, f(7) = 6

Table 9.5 *Example 9.50*

	$i = 1$	$i = 2$	$i = 3$	$i = 4$	$i = 5$	$i = 6$
$a = 1$	x: 1	x: 1	x: 1	x: 1	x: 1	x: 1
$a = 2$	x: 2	x: 4	x: 1	x: 2	x: 4	x: 1
Primitive root → $a = 3$	x: 3	x: 2	x: 6	x: 4	x: 5	x: 1
$a = 4$	x: 4	x: 2	x: 1	x: 4	x: 2	x: 1
Primitive root → $a = 5$	x: 5	x: 4	x: 6	x: 2	x: 3	x: 1
$a = 6$	x: 6	x: 1	x: 6	x: 1	x: 6	x: 1

Generating Primes

- A number in the form $M_p=2^p - 1$ is called a Mersenne number and may or may not be a prime.

$$
\begin{aligned}
M_2 &= 2^2 - 1 = 3 \\
M_3 &= 2^3 - 1 = 7 \\
M_5 &= 2^5 - 1 = 31 \\
M_7 &= 2^7 - 1 = 127 \\
M_{11} &= 2^{11} - 1 = 2047 \qquad \textbf{Not a prime } (\mathbf{2047 = 23 \times 89}) \\
M_{13} &= 2^{13} - 1 = 8191 \\
M_{17} &= 2^{17} - 1 = 131071
\end{aligned}
$$

Thank You!

Next Class

- ☞ Mandatory reading for the next class
  - ☞ https://www.math.upenn.edu/~mlazar/math170/notes06.pdf

The Chinese Remainder Theorem

The Chinese Remainder Theorem (CRT) is used to solve a set of different congruent equations with one variable but different moduli which are relatively prime as shown below:

$X \equiv a_1 \pmod{m_1}$

$X \equiv a_2 \pmod{m_2}$

. . .

$X \equiv a_n \pmod{m_n}$

CRT states that the above equations have a unique solution of the moduli are relatively prime.

The Chinese Remainder Theorem

The Chinese Remainder Theorem (CRT) is used to solve a set of different congruent equations with one variable but different moduli which are relatively prime as shown below:

$X \equiv a_1 \pmod{m_1}$

$X \equiv a_2 \pmod{m_2}$

. . .

$X \equiv a_n \pmod{m_n}$

CRT states that the above equations have a unique solution of the moduli are relatively prime.

$X = (a_1 M_1 M_1^{-1} + a_2 M_2 M_2^{-1} + \ldots + a_n M_n M_n^{-1}) \bmod M$

The Chinese Remainder Theorem

Example 1: Solve the following equations using CRT

$X \equiv 2 \pmod 3$

$X \equiv 3 \pmod 5$

$X \equiv 2 \pmod 7$

Solution:

$$X = (a_1 M_1 M_1^{-1} + a_2 M_2 M_2^{-1} + a_3 M_3 M_3^{-1}) \bmod M$$

The Chinese Remainder Theorem

$X \equiv a_1 \pmod{m_1}$	$X \equiv 2 \pmod{3}$
$X \equiv a_2 \pmod{m_2}$	$X \equiv 3 \pmod{5}$
$X \equiv a_3 \pmod{m_3}$	$X \equiv 2 \pmod{7}$

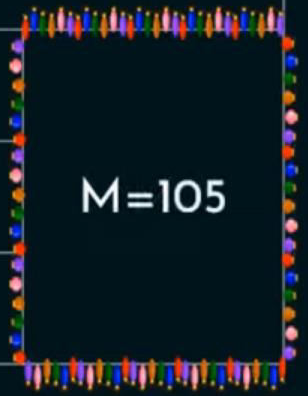
Solution:

$X = (a_1M_1 M_1^{-1} + a_2M_2M_2^{-1} + a_3M_3M_3^{-1}) \bmod M$

Given		To Find		
$a_1 = 2$	$m_1 = 3$	M_1	M_1^{-1}	M
$a_2 = 3$	$m_2 = 5$	M_2	M_2^{-1}	
$a_3 = 2$	$m_3 = 7$	M_3	M_3^{-1}	

The Chinese Remainder Theorem

Given		To Find	
$a_1 = 2$	$m_1 = 3$	M_1	M_1^{-1}
$a_2 = 3$	$m_2 = 5$	M_2	M_2^{-1}
$a_3 = 2$	$m_3 = 7$	M_3	M_3^{-1}



Solution:

$M = m_1 \times m_2 \times m_3$

$M = 3 \times 5 \times 7$

$M = 105$

The Chinese Remainder Theorem

Given		To Find		
$a_1 = 2$	$m_1 = 3$	$M_1 =$	M_1^{-1}	$M = 105$
$a_2 = 3$	$m_2 = 5$	$M_2 =$	M_2^{-1}	
$a_3 = 2$	$m_3 = 7$	$M_3 =$	M_3^{-1}	

$$M_1 = \frac{M}{m_1}$$

$$M_1 = \frac{105}{3}$$

$$M_1 = 35$$

$$M_2 = \frac{M}{m_2}$$

$$M_2 = \frac{105}{5}$$

$$M_2 = 21$$

$$M_3 = \frac{M}{m_3}$$

$$M_3 = \frac{105}{7}$$

$$M_3 = 15$$

The Chinese Remainder Theorem

Given		To Find		
$a_1 = 2$	$m_1 = 3$	$M_1 = 35$	M_1^{-1}	
$a_2 = 3$	$m_2 = 5$	$M_2 = 21$	M_2^{-1}	$M=105$
$a_3 = 2$	$m_3 = 7$	$M_3 = 15$	M_3^{-1}	

$M_1 \times M_1^{-1} = 1 \bmod m_1$

$35 \times M_1^{-1} = 1 \bmod 3$

$35 \times 2 = 1 \bmod 3$

$M_1^{-1} = 2$

$M_2 \times M_2^{-1} = 1 \bmod m_2$

$21 \times M_2^{-1} = 1 \bmod 5$

$21 \times 1 = 1 \bmod 5$

$M_2^{-1} = 1$

$M_3 \times M_3^{-1} = 1 \bmod m_3$

$15 \times M_3^{-1} = 1 \bmod 7$

$15 \times 1 = 1 \bmod 7$

$M_3^{-1} = 1$

The Chinese Remainder Theorem

Example 1: Solve the following equations using CRT

$X \equiv 2 \pmod 3$

$X \equiv 3 \pmod 5$

$X \equiv 2 \pmod 7$

$a_1 = 2$	$m_1 = 3$	$M_1 = 35$	$M_1^{-1} = 2$	$M=105$
$a_2 = 3$	$m_2 = 5$	$M_2 = 21$	$M_2^{-1} = 1$	
$a_3 = 2$	$m_3 = 7$	$M_3 = 15$	$M_3^{-1} = 1$	

Solution:

$X = (a_1M_1 M_1^{-1} + a_2M_2M_2^{-1} + a_3M_3M_3^{-1}) \bmod M$

$= (2x35x2 + 3x21x1 + 2x15x1) \bmod 105$

$= 233 \bmod 105$

$X = 23$

The Chinese Remainder Theorem

Example 1: Solve the following equations using CRT:

$4X \equiv 5 \pmod 9$

$2X \equiv 6 \pmod{20}$

Rewrite the question as follows:

$4X \equiv 5 \pmod 9$

Multiply by 4^{-1} on both sides

$4^{-1} \times 4X \equiv 4^{-1} \times 5 \pmod 9$

$X \equiv 4^{-1} \pmod 9 \times 5 \pmod 9$

$X \equiv 7 \times 5 \pmod 9$

$X \equiv 35 \pmod 9$

$2X \equiv 6 \pmod{20}$

$2X \equiv 2 \times 3 \pmod{20}$

$X \equiv 3 \pmod{20}$

The Chinese Remainder Theorem

Example 1: Solve the following equations using CRT:

$X \equiv 8 \pmod 9$

$X \equiv 3 \pmod{20}$

$X \equiv a_1 \pmod{m_1}$	$X \equiv 8 \pmod 9$
$X \equiv a_2 \pmod{m_2}$	$X \equiv 3 \pmod{20}$

Solution:

$X = (a_1 M_1 M_1^{-1} + a_2 M_2 M_2^{-1}) \bmod M$

Given		To Find		
$a_1 = 8$	$m_1 = 9$	M_1	M_1^{-1}	M
$a_2 = 3$	$m_2 = 20$	M_2	M_2^{-1}	

The Chinese Remainder Theorem

Given		To Find		
$a_1 = 8$	$m_1 = 9$	$M_1 = 20$	M_1^{-1}	$M = 180$
$a_2 = 3$	$m_2 = 20$	$M_2 = 9$	M_2^{-1}	

$$M_1 = \frac{M}{m_1}$$

$$M_1 = \frac{180}{9}$$

$$M_1 = 20$$

$$M_2 = \frac{M}{m_2}$$

$$M_2 = \frac{180}{20}$$

$$M_2 = 9$$

The Chinese Remainder Theorem

Given		To Find		
$a_1 = 8$	$m_1 = 9$	$M_1 = 20$	$M_1^{-1} = 5$	M=180
$a_2 = 3$	$m_2 = 20$	$M_2 = 9$	$M_2^{-1} = 9$	

$M_1 \times M_1^{-1} = 1 \bmod m_1$

$20 \times M_1^{-1} = 1 \bmod 9$

$20 \times 5 = 1 \bmod 9$

$M_1^{-1} = 5$

$M_2 \times M_2^{-1} = 1 \bmod m_2$

$9 \times M_2^{-1} = 1 \bmod 20$

$9 \times 9 = 1 \bmod 20$

$M_2^{-1} = 9$

The Chinese Remainder Theorem

Example 1: Solve the following equations using CRT:

$X \equiv 8 \pmod 9$

$X \equiv 3 \pmod{20}$

Given		To Find		
$a_1 = 8$	$m_1 = 9$	$M_1 = 20$	$M_1^{-1} = 5$	$M=180$
$a_2 = 3$	$m_2 = 20$	$M_2 = 9$	$M_2^{-1} = 9$	

Solution:

$X = (a_1M_1M_1^{-1} + a_2M_2M_2^{-1}) \bmod M$

$= (8x20x5 + 3x9x9) \bmod 180$

$= (800 + 243) \bmod 180$

$= 1043 \bmod 180$

$X = 143$

Dr. Sivaraman E

Computer Science and Engineering

PES University, Bengaluru

PESU Center for Information Security, Forensics and Cyber Resilience

PESU IoT

PESU Center for Internet of Things

PESU Center for
Information Security,
Forensics and
Cyber Resilience

PES
UNIVERSITY

Welcome to

PES University

Ring Road Campus, Bengaluru

PESU Center for
Information Security,
Forensics and
Cyber Resilience

PES
UNIVERSITY

APPLIED CRYPTOGRAPHY

Lecture 30

Modular Arithmetic

Introduction

- In integer arithmetic, if we divide a by n, we can get q and r. The relationship between these four integers can be shown as:

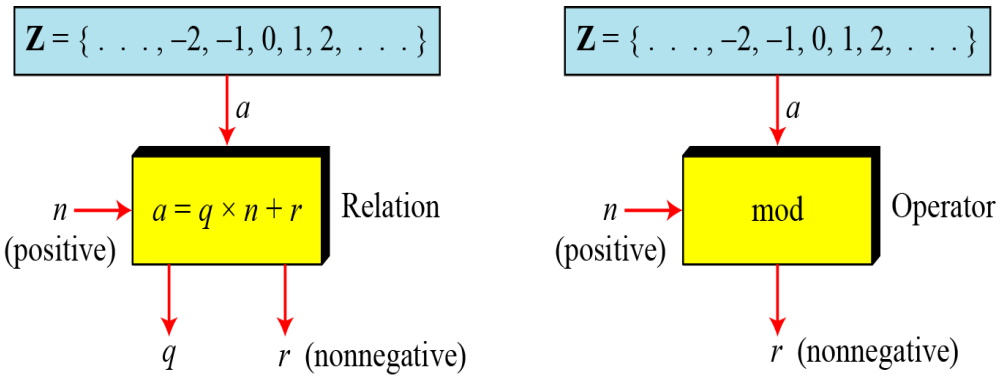
$$a = q \times n + r$$

- The division relationship has:
  - two inputs (a & n) and two outputs (q & r).
- In modular arithmetic, we are interested in only one of the outputs, the remainder r.

Modulo Operator

- The modulo operator is shown as **mod**.
- The second input (n) is called the modulus.
- The output r is called the residue.

Division algorithm and Modulo operator



Example

Calculate the results of:

- 27 mod 5
- 36 mod 12
- −18 mod 14
- −7 mod 10

Solution:

- Dividing 27 by 5 results in r = 2
- Dividing 36 by 12 results in r = 0.
- Dividing −18 by 14 results in r = −4. After adding the modulus, r = 10
- Dividing −7 by 10 results in r = −7. After adding the modulus to −7, r = 3.

Set of residues, Z_n

- a mod n = result (always non-negative and less than n)
- The modulo operation creates a set, which in modular arithmetic is referred to as the set of least residues modulo n, or Z_n.

Some Z_n sets

$$\mathbf{Z}_n = \{ 0, 1, 2, 3, \ldots, (n-1) \}$$

$$\mathbf{Z}_2 = \{ 0, 1 \}$$

$$\mathbf{Z}_6 = \{ 0, 1, 2, 3, 4, 5 \}$$

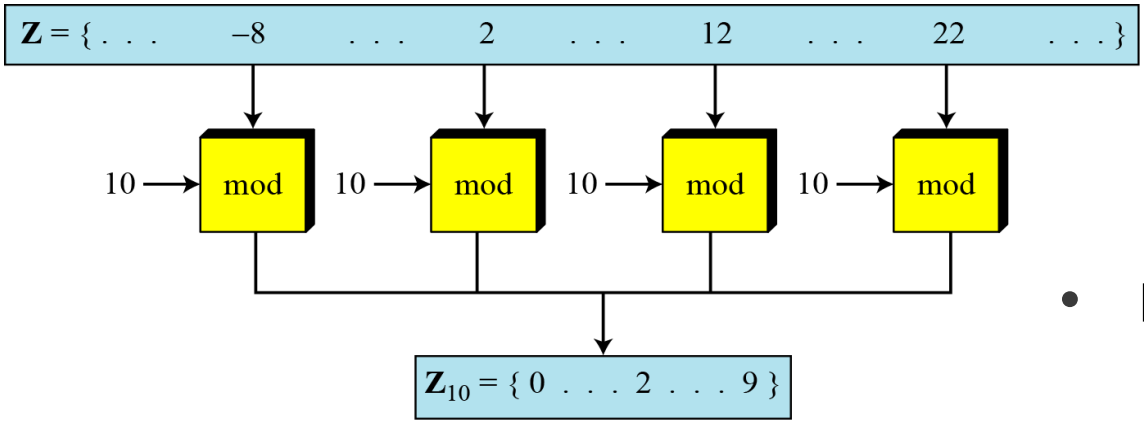
$$\mathbf{Z}_{11} = \{ 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10 \}$$

Congruence (≡)

- To show that two integers are congruent, we use the congruence operator (≡).
- If a and b have same reminder when divided by n (always, n>1), it can be expressed as, **a ≡ b (mod n).**
- In other words, **a mod n = b mod n**
- For example: $12 \equiv 2 \pmod 5$

  $1000 \equiv 1 \pmod{37}$
- Also, **a = k.n + b**
- Or, n | (a-b) i.e., a-b = **k.n**

Concept of Congruence



$-8 \equiv 2 \equiv 12 \equiv 22 \pmod{10}$

Congruence Relationship

- Result of:

  2 mod 10 = 2
  12 mod 10 = 2
  22 mod 10 = 2
  -8 mod 10 = 2

  $-8 \equiv 2 \equiv 12 \equiv 22 \pmod{10}$

Continued..

- a = k.n + r, so, by division algorithm, $0 \leq r \leq n$, i.e., possible values of r are 0,1,2,……,(n-1).
- A residue class of r denoted by [r] or $[r]_n$ is the set of integers congruent modulo n.

  $[r] = \{x : x \text{ is an integer such that } x \equiv r \pmod{n}\}$

  $[r] = \{x : x \text{ is an integer such that } x = n \cdot k + r\}$

  [r] = {x : r is remainder when we divide integer x by n}
- Since there are n-1 possibilities of r, we get n-1 residue classes as [0], [1], [2], …. , [n-1] modulo n.
- We denote the set of all residue classes modulo n by Z_n.

Illustration

- Let n = 5, then we divide any integer by n, the possible values of remainders are 0,1,2,3,4.

[0] = {x : x is an integer such that it leaves remainder 0 when divided by 5}

= { …., -10, -5, 0, 5, 10, ….}

[1] = {….., -9, -4, 1, 6, 11, …..,}

[2] = {….., -8, -3, 2, 7, 12, …..,}

[3] = {….., -7, -2, 3, 8, 13, …..,}

[4] = {….., -6, -1, 4, 9, 14, …..,}

Residue Classes

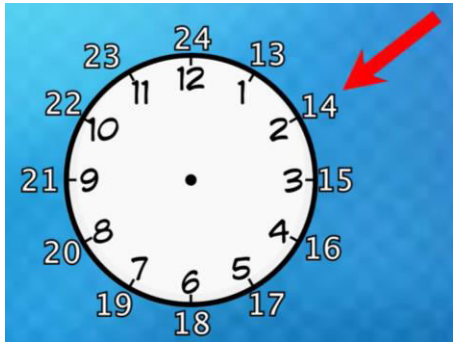
$$
\begin{aligned}
[0] &= \{\ldots, -15, -10, -5, 0, 5, 10, 15, \ldots\} \\
[1] &= \{\ldots, -14, -9, -4, 1, 6, 11, 16, \ldots\} \\
[2] &= \{\ldots, -13, -8, -3, 2, 7, 12, 17, \ldots\} \\
[3] &= \{\ldots, -12, -7, -5, 3, 8, 13, 18, \ldots\} \\
[4] &= \{\ldots, -11, -6, -1, 4, 9, 14, 19, \ldots\}
\end{aligned}
$$

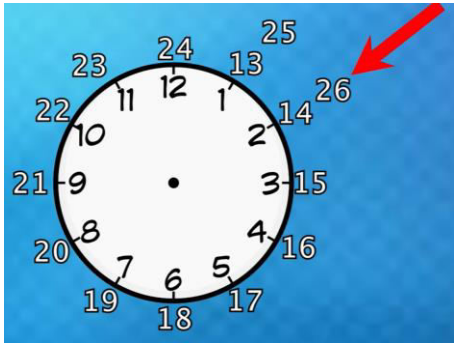
Continued .. Example

- We use modular arithmetic in our daily life; e.g., we use a clock to measure time.
- Our clock system uses modulo 12 arithmetic.
- However, instead of a 0, we use the number 12.

$14 \equiv 2 \pmod{12}$

$26 \equiv 14 \pmod{12}$ ➡ $26 \equiv 2 \pmod{12}$

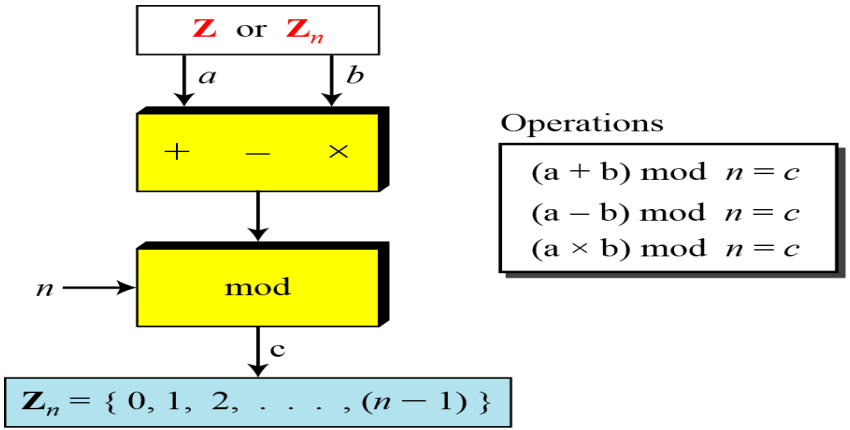




Binary Operations on Z_n

- The three binary operations that we discussed for the set Z can also be defined for the set Z_n. The result may need to be mapped to Z_n using the mod operator.

Binary operations in Z_n



Operations

$(a + b) \bmod n = c$

$(a - b) \bmod n = c$

$(a \times b) \bmod n = c$

Binary Operations on Z_n

Add:	$5 + 9 = 14$	$(-5) + 9 = 4$	$5 + (-9) = -4$	$(-5) + (-9) = -14$
Subtract:	$5 - 9 = -4$	$(-5) - 9 = -14$	$5 - (-9) = 14$	$(-5) - (-9) = +4$
Multiply:	$5 \times 9 = 45$	$(-5) \times 9 = -45$	$5 \times (-9) = -45$	$(-5) \times (-9) = 45$

Four cases for each operation.

Perform the following operations (the inputs come from Zn):

a) Add 7 to 14 in Z_{15}.
b) Subtract 11 from 7 in Z_{13}.
c) Multiply 11 by 7 in Z_{20}.

$$(14 + 7) \bmod 15 \rightarrow (21) \bmod 15 = 6$$
$$(7 - 11) \bmod 13 \rightarrow (-4) \bmod 13 = 9$$
$$(7 \times 11) \bmod 20 \rightarrow (77) \bmod 20 = 17$$

Example-2

Perform the following operations (the inputs come from either Z or Z_n):

a) Add 17 to 27 in Z_{14}.

b) Subtract 43 from 12 in Z_{13}.

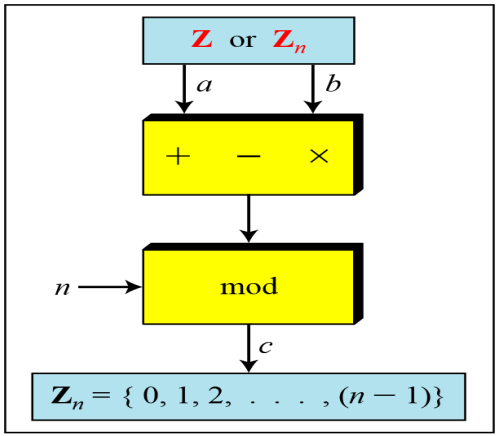
c) Multiply 123 by −10 in Z_{19}.

Properties of Congruence

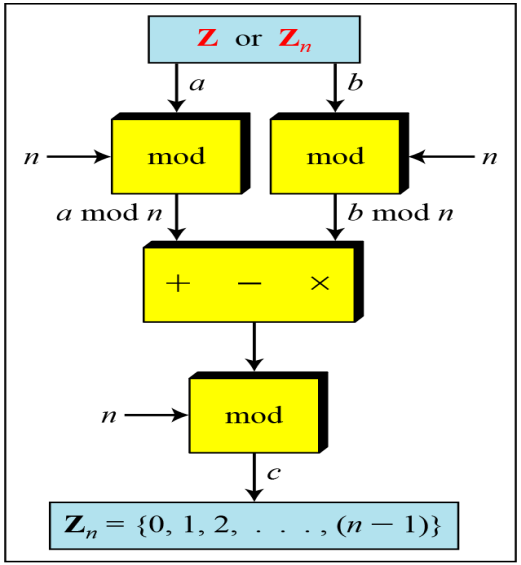
First Property:	$(a + b) \bmod n = [(a \bmod n) + (b \bmod n)] \bmod n$
Second Property:	$(a - b) \bmod n = [(a \bmod n) - (b \bmod n)] \bmod n$
Third Property:	$(a \times b) \bmod n = [(a \bmod n) \times (b \bmod n)] \bmod n$

1. (1,723,345 + 2,124,945) mod 11 = (8 + 9) mod 11 = 6
2. (1,723,345 − 2,124,945) mod 16 = (8 − 9) mod 11 = 10
3. (1,723,345 × 2,124,945) mod 16 = (8 × 9) mod 11 = 6

Properties of mode operator



a. Original process



b. Applying properties

Inverses

- When we are working in modular arithmetic, we often need to find the inverse of a number relative to an operation.
- We are normally looking for an additive inverse (relative to an addition operation) or a multiplicative inverse (relative to a multiplication operation).

Additive Inverse

- In Z_n, two numbers a and b are additive inverses of each other if

$$a + b \equiv 0 \pmod{n}$$

- In modular arithmetic, each integer has an additive inverse. The sum of an integer and its additive inverse is congruent to 0 modulo n.

Example: Find all additive inverse pairs in Z_{10}.

The six pairs of additive inverses are

(0, 0), (1, 9), (2, 8), (3, 7), (4, 6), and (5, 5).

Multiplicative Inverse

- In Z_n, two numbers a and b are the multiplicative inverse of each other if: $a \times b \equiv 1 \pmod{n}$
- In modular arithmetic, an integer may or may not have a multiplicative inverse. When it does, the product of the integer and its multiplicative inverse is congruent to 1 modulo n.
- Example: Find all multiplicative inverse pairs in Z_{11}.

There are seven pairs: (1, 1), (2, 6), (3, 4), (5, 9), (7, 8), (9, 9), and (10, 10).

The inverse of b exists only if gcd (n, b) = 1 (mod n)

Example

1) Find the multiplicative inverse of 11 in Z_{26}.

2) Find the multiplicative inverse of 8 in Z_{10}.

Solution:

- There is no multiplicative inverse because gcd (10, 8) = 2 ≠ 1.
- In other words, we cannot find any number between 0 and 9 such that when multiplied by 8, the result is congruent to 1.

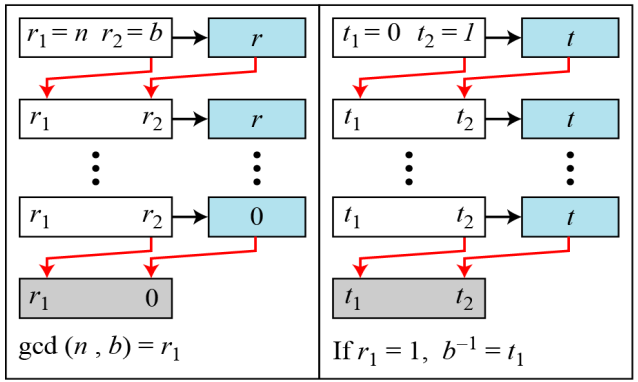
3) Find all multiplicative inverses in Z_{10}.

Solution:

- There are only three pairs: (1, 1), (3, 7) and (9, 9).
- The numbers 0, 2, 4, 5, 6, and 8 do not have a multiplicative inverse.

Extended Euclidean Algorithm

- The extended Euclidean algorithm finds the multiplicative inverses of b in Z_n when n and b are given and gcd (n, b) = 1.
- The multiplicative inverse of b is the value of t after being mapped to Z_n.



a. Process

$r_1 \leftarrow$ n; $r_2 \leftarrow b$;
$t_1 \leftarrow 0$; $t_2 \leftarrow 1$;

while $(r_2 > 0)$
{
$q \leftarrow r_1 / r_2$;

$r \leftarrow r_1 - q \times r_2$;
$r_1 \leftarrow r_2$; $r_2 \leftarrow r$;

$t \leftarrow t_1 - q \times t_2$;
$t_1 \leftarrow t_2$; $t_2 \leftarrow t$;

}
if $(r_1 = 1)$ then $b^{-1} \leftarrow t_1$

b. Algorithm

Example

Find the multiplicative inverse of 11 in Z_{26}

q	r_1	r_2	r	t_1	t_2	t
2	26	11	4	0	1	–2
2	11	4	3	1	–2	5
1	4	3	1	–2	5	–7
3	3	1	0	5	–7	26
	1	0		–7	26	

The gcd (26, 11) is 1; the inverse of 11 is -7 or 19.

Example

Find the multiplicative inverse of 23 in Z_{100}.

q	r_1	r_2	r	t_1	t_2	t
4	100	23	8	0	1	−4
2	23	8	7	1	−4	19
1	8	7	1	−4	9	−13
7	7	1	0	9	−13	100
	1	0		−13	100	

The gcd (100, 23) is 1; the inverse of 23 is -13 or 87.

Example

Find the inverse of 12 in Z_{26}.

q	r_1	r_2	r	t_1	t_2	t
2	26	12	2	0	1	−2
6	12	2	0	1	−2	13
	2	0		−2	13	

The gcd (26, 12) is 2; the inverse does not exist.

Addition and Multiplication Tables

	0	1	2	3	4	5	6	7	8	9
0	**0**	1	2	3	4	5	6	7	8	9
1	1	2	3	4	5	6	7	8	9	**0**
2	2	3	4	5	6	7	8	9	**0**	1
3	3	4	5	6	7	8	9	**0**	1	2
4	4	5	6	7	8	9	**0**	1	2	3
5	5	6	7	8	9	**0**	1	2	3	4
6	6	7	8	9	**0**	1	2	3	4	5
7	7	8	9	**0**	1	2	3	4	5	6
8	8	9	**0**	1	2	3	4	5	6	7
9	9	**0**	1	2	3	4	5	6	7	8

Addition Table in $\mathbf{Z}_{10}$

	0	1	2	3	4	5	6	7	8	9
0	0	0	0	0	0	0	0	0	0	0
1	0	**1**	2	3	4	5	6	7	8	9
2	0	2	4	6	8	0	2	4	6	8
3	0	3	6	9	2	5	8	**1**	4	7
4	0	4	8	2	6	0	4	8	2	6
5	0	5	0	5	0	5	0	5	0	5
6	0	6	2	8	4	0	6	2	8	4
7	0	7	4	**1**	8	0	2	9	6	3
8	0	8	6	4	2	0	8	6	4	2
9	0	9	8	7	6	5	4	3	2	**1**

Multiplication Table in $\mathbf{Z}_{10}$

Different Sets

- We need to use Z_n when additive inverses are needed; we need to use Z_n^* when multiplicative inverses are needed.

Some Z_n and Z_{n^*} sets

$\mathbf{Z}_6 = \{0, 1, 2, 3, 4, 5\}$

$\mathbf{Z}_6^* = \{1, 5\}$

$\mathbf{Z}_7 = \{0, 1, 2, 3, 4, 5, 6\}$

$\mathbf{Z}_7^* = \{1, 2, 3, 4, 5, 6\}$

$\mathbf{Z}_{10} = \{0, 1, 2, 3, 4, 5, 6, 7, 8, 9\}$

$\mathbf{Z}_{10}^* = \{1, 3, 7, 9\}$

Two more sets

- Cryptography often uses two more sets: Z_p and Z_p^*. The modulus in these two sets is a prime number.

$$Z_{13} = \{0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12\}$$
$$Z_{13}^* = \{1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12\}$$

Thank You!

Next Class

- ☞ Mandatory reading for the next class
  - ☞ https://pdfs.semanticscholar.org/40f3/95bf05420ea57578ed209c408e1f2e309094.pdf

Dr. Sivaraman E

Computer Science and Engineering

PES University, Bengaluru

PESU Center for Information Security, Forensics and Cyber Resilience

PESU IoT

PESU Center for Internet of Things

PESU Center for
Information Security,
Forensics and
Cyber Resilience

PES
UNIVERSITY

Welcome to

PES University

Ring Road Campus, Bengaluru

PESU Center for
Information Security,
Forensics and
Cyber Resilience

PES
UNIVERSITY

APPLIED CRYPTOGRAPHY

Lecture 31

Polynomials

Introduction

- A polynomial of degree $n-1$ is an expression of the form

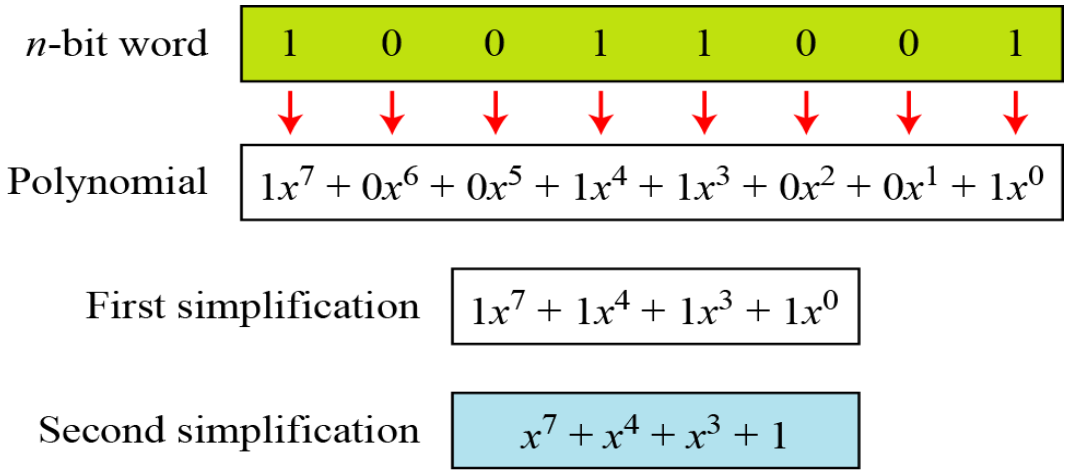
$$f(x) = a_{n-1}x^{n-1} + a_{n-2}x^{n-2} + \cdots + a_1x^1 + a_0x^0$$

Where,
a_i is called coefficient of the ith term.

Introduction

- Figure show how we can represent the 8-bit word (10011001) using a polynomials.

Representation of an 8-bit word by a polynomial



Example

- To find the 8-bit word related to the polynomial $x^5 + x^2 + x$, we first supply the omitted terms. Since $n = 8$, it means the polynomial is of degree 7. The expanded polynomial is: $0x^7 + 0x^6 + 1x^5 + 0x^4 + 0x^3 + 1x^2 + 1x^1 + 0x^0$

- This is related to the 8-bit word 00100110.

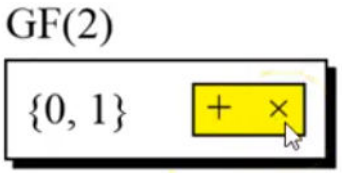
GF(2^n) Fields

- In cryptography, we often need to use operations (add, sub, mul, div). In other words, we need to use fields.
- Galois Field, GF(p^n) – Galois showed that for a field to be finite, the number of elements should be p^n.
  - where, p is a prime and n is a positive integer.
- We will study about the two special cases.

  (n=1), the prime field: GF(p), (n>1 and p is 2), the binary field: GF(2^n)
- Polynomials representing *n*-bit words use two fields:
  - One for the coefficients: GF(2)
  - One for the polynomials: GF(2^n)

Example 1

A very common field in this category is GF(2) with the set {0, 1} and two operations, addition and multiplication.



+	0	1
0	0	1
1	1	0

Addition

×	0	1
0	0	0
1	0	1

Multiplication

a	0	1
−a	1	0

a	0	1
a^{-1}	—	1

Inverses

Example 1

We can define GF(5) on the set Z5 (5 is a prime) with addition and multiplication operations.

GF(5)

{0, 1, 2, 3, 4} + ×

+	0	1	2	3	4
0	0	1	2	3	4
1	1	2	3	4	0
2	2	3	4	0	1
3	3	4	0	1	2
4	4	0	1	2	3

Addition

×	0	1	2	3	4
0	0	0	0	0	0
1	0	1	2	3	4
2	0	2	4	1	3
3	0	3	1	4	2
4	0	4	3	2	1

Multiplication

Additive inverse

a	0	1	2	3	4
−a	0	4	3	2	1

a	0	1	2	3	4
a^{-1}	—	1	3	2	4

Multiplicative inverse

GF(7) – Addition Modulo

+	0	1	2	3	4	5	6
0	0	1	2	3	4	5	6
1	1	2	3	4	5	6	0
2	2	3	4	5	6	0	1
3	3	4	5	6	0	1	2
4	4	5	6	0	1	2	3
5	5	6	0	1	2	3	4
6	6	0	1	2	3	4	5

(a) Addition modulo 7

w	$-w$	w^{-1}
0	0	—
1	6	1
2	5	4
3	4	5
4	3	2
5	2	3
6	1	6

(c) Additive and multiplicative inverses modulo 7

GF(7) – Multiplication Modulo

×	0	1	2	3	4	5	6
0	0	0	0	0	0	0	0
1	0	1	2	3	4	5	6
2	0	2	4	6	1	3	5
3	0	3	6	2	5	1	4
4	0	4	1	5	2	6	3
5	0	5	3	1	6	4	2
6	0	6	5	4	3	2	1

(b) Multiplication modulo 7

w	$-w$	w^{-1}
0	0	—
1	6	1
2	5	4
3	4	5
4	3	2
5	2	3
6	1	6

(c) Additive and multiplicative inverses modulo 7

Irreducible Polynomial

- For the sets of polynomials in GF(2^n), a group of polynomials of degree *n* is defined as irreducible polynomials.
- If a polynomial can't be factored into a product of polynomials of lower degree.

List of irreducible polynomials

Degree	*Irreducible Polynomials*
1	$(x + 1), (x)$
2	$(x^2 + x + 1)$
3	$(x^3 + x^2 + 1), (x^3 + x + 1)$
4	$(x^4 + x^3 + x^2 + x + 1), (x^4 + x^3 + 1), (x^4 + x + 1)$
5	$(x^5 + x^2 + 1), (x^5 + x^3 + x^2 + x + 1), (x^5 + x^4 + x^3 + x + 1),$ $(x^5 + x^4 + x^3 + x^2 + 1), (x^5 + x^4 + x^2 + x + 1)$

- 7 is prime because it can't be factored into a product of smaller numbers.
- $x^2 - 3$ is "prime" because it can't be factored into a polynomials of lower degree.

Addition and Subtraction operation

- Addition and subtraction operations on polynomials are the same operation.
- Let us do $(x^5 + x^2 + x) \oplus (x^3 + x^2 + 1)$ in $GF(2^8)$. We use the symbol to show $\oplus$ that we mean polynomial addition. The following shows the procedure:

$$
\begin{array}{lll}
0x^7 + 0x^6 + 1x^5 + 0x^4 + 0x^3 + 1x^2 + 1x^1 + 0x^0 & \oplus & \\
0x^7 + 0x^6 + 0x^5 + 0x^4 + 1x^3 + 1x^2 + 0x^1 + 1x^0 & & \\
\text{---------- (50 dashes) ----------} & & \\
0x^7 + 0x^6 + 1x^5 + 0x^4 + 1x^3 + 0x^2 + 1x^1 + 1x^0 & \rightarrow & x^5 + x^3 + x + 1
\end{array}
$$

Example

- There is also another short cut. Because the addition in GF(2) means the exclusive-or (XOR) operation.
- So, we can exclusive-or the two words, bits by bits, to get the result.
- In the previous example, $x^5 + x^2 + x$ is 00100110 and $x^3 + x^2 + 1$ is 00001101.
- The result is 00101011 or in polynomial notation $x^5 + x^3 + x + 1$.

Operations on Irreducible Polynomials

- Addition and subtraction operations on polynomials are the same operations.
- GF(2^8) is a field in which every element is its own opposite. This implies subtraction is the same as addition.
- Say, -x equal to x

Multiplication

1. The coefficient multiplication is done in GF(2).
2. The multiplying x^i by x^j results in x^{i+j}.
3. The multiplication may create terms with degree more than $n - 1$, which means the result needs to be reduced using a modulus polynomial.

Example for Multiplication

- Find the result of $(x^5 + x^2 + x) \otimes (x^7 + x^4 + x^3 + x^2 + x)$ in GF(2^8) with irreducible polynomial $(x^8 + x^4 + x^3 + x + 1)$.
- Note that we use the symbol $\otimes$ to show the multiplication of two polynomials.

$$P_1 \otimes P_2 = x^5(x^7 + x^4 + x^3 + x^2 + x) + x^2(x^7 + x^4 + x^3 + x^2 + x) + x(x^7 + x^4 + x^3 + x^2 + x)$$
$$P_1 \otimes P_2 = x^{12} + x^9 + x^8 + x^7 + x^6 + x^9 + x^6 + x^5 + x^4 + x^3 + x^8 + x^5 + x^4 + x^3 + x^2$$
$$P_1 \otimes P_2 = (x^{12} + x^7 + x^2) \bmod (x^8 + x^4 + x^3 + x + 1) = x^5 + x^3 + x^2 + x + 1$$

To find the result, divide the polynomial of degree 12 by the polynomial of degree 8 (the modulus) and keep only the remainder.

Example for Division

$$
\begin{array}{r|l}
 & x^4 + 1 \\
\hline
x^8 + x^4 + x^3 + x + 1 & x^{12} + x^7 + x^2 \\
 & x^{12} + x^8 + x^7 + x^5 + x^4 \\
\hline
 & x^8 + x^5 + x^4 + x^2 \\
 & x^8 + x^4 + x^3 + x + 1 \\
\hline
\text{Remainder} & \boxed{x^5 + x^3 + x^2 + x + 1}
\end{array}
$$

Continued.. Example

In GF (2^4), find the inverse of ($x^2 + 1$) modulo ($x^4 + x + 1$).

Solution: The answer is ($x^3 + x + 1$) as shown in Table.

Euclidean algorithm

q	r_1	r_2	r	t_1	t_2	t
$(x^2 + 1)$	$(x^4 + x + 1)$	$(x^2 + 1)$	(x)	(0)	(1)	$(x^2 + 1)$
(x)	$(x^2 + 1)$	(x)	(1)	(1)	$(x^2 + 1)$	$(x^3 + x + 1)$
(x)	(x)	(1)	(0)	$(x^2 + 1)$	$(x^3 + x + 1)$	(0)
	(1)	(0)		$(x^3 + x + 1)$	(0)	

Continued .. Example

In GF(2^8), find the inverse of (x^5) modulo ($x^8 + x^4 + x^3 + x + 1$).

Solution: The answer is ($x^5 + x^4 + x^3 + x$) as shown in Table

Euclidean algorithm

q	r_1	r_2	r	t_1	t_2	t
(x^3)	$(x^8+x^4+x^3+x+1)$	(x^5)	(x^4+x^3+x+1)	(0)	(1)	(x^3)
$(x+1)$	(x^5)	(x^4+x^3+x+1)	(x^3+x^2+1)	(1)	(x^3)	(x^4+x^3+1)
(x)	(x^4+x^3+x+1)	(x^3+x^2+1)	(1)	(x^3)	(x^4+x^3+1)	$(x^5+x^4+x^3+x)$
(x^3+x^2+1)	(x^3+x^2+1)	(1)	(0)	(x^4+x^3+1)	$(x^5+x^4+x^3+x)$	(0)
	(1)	(0)		$(x^5+x^4+x^3+x)$	(0)	

Multiplication Using Computer

- The computer implementation uses a better algorithm, repeatedly multiplying a reduced polynomial by *x*.
- Eg: instead of finding the result of ($x^2 \otimes P2$), the program finds the result of ($x \otimes (x \otimes P2)$).

Example

- Find the result of multiplying $P_1 = (x^5 + x^2 + x)$ by $P_2 = (x^7 + x^4 + x^3 + x^2 + x)$ in GF(2^8) with irreducible polynomial $(x^8 + x^4 + x^3 + x + 1)$ using the algorithm described above.

Solution:

We first find the partial result of multiplying x^0, x^1, x^2, x^3, x^4, and x^5 by P_2. Note that although only three terms are needed, the product of $x^m \otimes P_2$ for m from 0 to 5 because each calculation depends on the previous result.

Continued ..

An efficient algorithm

Powers	*Operation*	*New Result*	*Reduction*
$x^0 \otimes P_2$		$x^7 + x^4 + x^3 + x^2 + x$	No
$x^1 \otimes P_2$	$\boldsymbol{x} \otimes (x^7 + x^4 + x^3 + x^2 + x)$	$x^5 + x^2 + x + 1$	**Yes**
$x^2 \otimes P_2$	$\boldsymbol{x} \otimes (x^5 + x^2 + x + 1)$	$x^6 + x^3 + x^2 + x$	No
$x^3 \otimes P_2$	$\boldsymbol{x} \otimes (x^6 + x^3 + x^2 + x)$	$x^7 + x^4 + x^3 + x^2$	No
$x^4 \otimes P_2$	$\boldsymbol{x} \otimes (x^7 + x^4 + x^3 + x^2)$	$x^5 + x + 1$	**Yes**
$x^5 \otimes P_2$	$\boldsymbol{x} \otimes (x^5 + x + 1)$	$x^6 + x^2 + x$	No
$\mathbf{P_1 \times P_2} = (x^6 + x^2 + x) + (x^6 + x^3 + x^2 + x) + (x^5 + x^2 + x + 1) = \boldsymbol{x^5 + x^3 + x^2 + x + 1}$			

Example

- Repeat previous example using bit patterns of size 8.
- Multiplication of polynomials in GF(2^n) can be achieved using shift-left and exclusive-or operations.

Solution: P1 = 000100110, P2 = 10011110, modulus = 100011010 (nine bits).

Powers	*Shift-Left Operation*	*Exclusive-Or*
$x^0 \otimes P_2$		10011110
$x^1 \otimes P_2$	00111100	(00111100) $\oplus$ (00011010) = **00100111**
$x^2 \otimes P_2$	01001110	**01001110**
$x^3 \otimes P_2$	10011100	10011100
$x^4 \otimes P_2$	00111000	(00111000) $\oplus$ (00011010) = 00100011
$x^5 \otimes P_2$	01000110	**01000110**
$P_1 \otimes P_2$ = (00100111) $\oplus$ (01001110) $\oplus$ (01000110) = 00101111		

When shifting left, the most-significant bit is lost, and a 0 bit is inserted on the other end.

Continued ..

- The GF(2^3) field has 8 elements. We use the irreducible polynomial ($x^3 + x^2 + 1$) and show the addition and multiplication tables for this field.
- We show both 3-bit words and the polynomials. Note that there are two irreducible polynomials for degree 3.
- The other one, ($x^3 + x + 1$), yields a totally different table for multiplication.

Addition table for GF(2^3)

$\oplus$	000 (0)	001 (1)	010 (x)	011 ($x+1$)	100 (x^2)	101 x^2+1	110 (x^2+x)	111 (x^2+x+1)
000 (0)	000 (0)	001 (1)	010 (x)	011 ($x+1$)	100 (x^2)	101 (x^2+1)	110 (x^2+x)	111 (x^2+x+1)
001 (1)	001 (1)	000 (0)	011 ($x+1$)	010 (x^2)	101 (x^2+1)	100 (x^2+x)	111 (x^2+x+1)	110 (x^2+x)
010 (x)	010 (x)	011 ($x+1$)	000 (0)	001 (1)	110 (x^2+x)	111 (x^2+x+1)	100 (x^2+x)	101 (x^2+1)
011 ($x+1$)	011 ($x+1$)	010 (x)	001 (1)	000 (0)	111 (x^2+x+1)	110 (x^2+x)	101 (x^2+1)	100 (x^2)
100 (x^2)	100 (x^2)	101 (x^2+1)	110 (x^2+x)	111 (x^2+x+1)	000 (0)	001 (1)	010 (x)	011 ($x+1$)
101 (x^2+1)	101 (x^2+1)	100 (x^2)	111 (x^2+x+1)	110 (x^2+x)	001 (1)	000 (0)	011 ($x+1$)	010 (x)
110 (x^2+x)	110 (x^2+x)	111 (x^2+x+1)	100 (x^2)	101 (x^2+1)	010 (x)	011 ($x+1$)	000 (0)	001 (1)
111 (x^2+x+1)	111 (x^2+x+1)	110 (x^2+x)	101 (x^2+1)	100 (x^2)	011 ($x+1$)	010 (x)	001 (1)	000 (0)

Multiplication table for GF(2^3)

$\otimes$	000 **(0)**	001 **(1)**	010 (x)	011 $(x+1)$	100 (x^2)	101 (x^2+1)	110 (x^2+x)	111 (x^2+x+1)
000 (0)	000 (0)	000 (0)	000 (0)	000 (0)	000 (0)	000 (0)	000 (0)	000 (0)
001 (1)	000 (0)	001 (1)	010 (x)	011 $(x+1)$	100 (x^2)	101 (x^2+1)	110 (x^2+x)	111 (x^2+x+1)
010 (x)	000 (0)	010 (x)	100 (x)	110 (x^2+x)	101 (x^2+1)	111 (x^2+x+1)	001 (1)	011 $(x+1)$
011 $(x+1)$	000 (0)	011 $(x+1)$	110 (x^2+x)	101 (x^2+1)	001 (1)	010 (x)	111 (x^2+x+1)	100 (x)
100 (x^2)	000 (0)	100 (x^2)	101 (x^2+1)	001 (1)	111 (x^2+x+1)	011 $(x+1)$	010 (x)	110 (x^2+x)
101 (x^2+1)	000 (0)	101 (x^2+1)	111 (x^2+x+1)	010 (x)	011 $(x+1)$	110 (x^2+x)	100 (x^2)	001 (1)
110 (x^2+x)	000 (0)	110 (x^2+x)	001 (1)	111 (x^2+x+1)	010 (x)	100 (x^2)	011 $(x+1)$	101 (x^2+1)
111 (x^2+x+1)	000 (0)	111 (x^2+x+1)	011 $(x+1)$	100 (x^2)	110 (x^2+x)	001 (1)	101 (x^2+1)	010 (x)

Using a Generator to represent GF(2^n)

- Sometimes it is easier to define the elements of the GF(2^n) field using a generator.
- If g is a generator element, then every element of GF(2^n), except for the 0 element, can be expressed as some power of g.

$$\{0, g, g, g^2, \ldots, g^N\}, \text{ where } N = 2^n - 2$$

Example

- Generate the elements of the field GF(2^4) using the irreducible polynomial $f(x) = x^4 + x + 1$.

- **Solution:** The elements 0, g^0, g^1, $g2$, and g^3 can be easily generated, because they are the 4-bit representations of 0, 1, x^2, and x^3. Elements g^4 through g^{14}, which represent x^4 though x^{14} need to be divided by the irreducible polynomial. To avoid the polynomial division, the relation $f(g) = g^4 + g + 1 = 0$ can be used.

Continued..

0	$=$	0	$=$	0	$=$	0	$\longrightarrow$	0	$=$	(0000)
g^0	$=$	g^0	$=$	g^0	$=$	g^0	$\longrightarrow$	g^0	$=$	(0001)
g^1	$=$	g^1	$=$	g^1	$=$	g^1	$\longrightarrow$	g^1	$=$	(0010)
g^2	$=$	g^2	$=$	g^2	$=$	g^2	$\longrightarrow$	g^2	$=$	(0100)
g^3	$=$	g^3	$=$	g^3	$=$	g^3	$\longrightarrow$	g^3	$=$	(1000)
g^4	$=$	g^4	$=$	g^4	$=$	$g+1$	$\longrightarrow$	g^4	$=$	(0011)
g^5	$=$	$g\,(g^4)$	$=$	$g\,(g+1)$	$=$	g^2+g	$\longrightarrow$	g^5	$=$	(0110)
g^6	$=$	$g\,(g^5)$	$=$	$g\,(g^2+g)$	$=$	g^3+g^2	$\longrightarrow$	g^6	$=$	(1100)
g^7	$=$	$g\,(g^6)$	$=$	$g\,(g^3+g)$	$=$	g^3+g+1	$\longrightarrow$	g^7	$=$	(1011)
g^8	$=$	$g\,(g^7)$	$=$	$g\,(g^3+g+1)$	$=$	g^2+1	$\longrightarrow$	g^8	$=$	(0101)
g^9	$=$	$g\,(g^8)$	$=$	$g\,(g^2+1)$	$=$	g^3+g	$\longrightarrow$	g^9	$=$	(1010)
g^{10}	$=$	$g\,(g^9)$	$=$	$g\,(g^3+g)$	$=$	g^2+g+1	$\longrightarrow$	g^{10}	$=$	(0111)
g^{11}	$=$	$g\,(g^{10})$	$=$	$g\,(g^2+g+1)$	$=$	g^3+g^2+g	$\longrightarrow$	g^{11}	$=$	(1110)
g^{12}	$=$	$g\,(g^{11})$	$=$	$g\,(g^3+g^2+g)$	$=$	g^3+g^2+g+1	$\longrightarrow$	g^{12}	$=$	(1111)
g^{13}	$=$	$g\,(g^{12})$	$=$	$g\,(g^3+g^2+g+1)$	$=$	g^3+g^2+1	$\longrightarrow$	g^{13}	$=$	(1101)
g^{14}	$=$	$g\,(g^{13})$	$=$	$g\,(g^3+g^2+1)$	$=$	g^3+1	$\longrightarrow$	g^{14}	$=$	(1001)

Example

- The following show the results of addition and subtraction operations:

a. $g^3 + g^{12} + g^7 = g^3 + (g^3 + g^2 + g + 1) + (g^3 + g + 1) = g^3 + g^2 \rightarrow (1100)$

b. $g^3 - g^6 = g^3 + g^6 = g^3 + (g^3 + g^2) = g^2 \rightarrow (0100)$

- The following show the result of multiplication and division operations:

a. $g^9 \times g^{11} = g^{20} = g^{20 \bmod 15} = g^5 = g^2 + g \rightarrow (0110)$

b. $g^3 / g^8 = g^3 \times g^7 = g^{10} = g^2 + g + 1 \rightarrow (0111)$

Summary

- The finite field GF(2^n) can be used to define four operations of addition, subtraction, multiplication and division over *n*-bit words.
- The only restriction is that division by zero is not defined.

Thank You!

Next Class

- ☞ Mandatory reading for the next class
  - ☞ https://iacr.org/archive/eurocrypt2001/20450451.pdf

Dr. Sivaraman E

Computer Science and Engineering

PES University, Bengaluru

PESU Center for Information Security, Forensics and Cyber Resilience

PESU IoT

PESU Center for Internet of Things

PESU Center for
Information Security,
Forensics and
Cyber Resilience

PES
UNIVERSITY

Welcome to

PES University

Ring Road Campus, Bengaluru

PESU Center for
Information Security,
Forensics and
Cyber Resilience

PES
UNIVERSITY

APPLIED CRYPTOGRAPHY

Lecture 32

Diffie-Hellman Key exchange algorithm

Generating secret key at sender's side and receiver's side

Exchanging Keys between Communicating Parties

Sam

Tom

Definition of Diffie-Hellman

- Whitfield Diffie & Martin Hellman in 1970s.
- A key-exchange protocol, also known as exponential key exchange.
- Enables 2 parties communicating over public channel to establish a mutual secret without it being transmitted over the Internet.
- Enables the two to use a public key to encrypt and decrypt their data using symmetric cryptography.

Transmitting Keys without Sharing

Tom and Sam agree upon an **arbitrary starting color key** that does not need to be kept secret but, should be different every time.

Tom

Sam

Transmitting Keys without Sharing

Tom and Sam each select the color key that they keep to themselves in this case **orange** and **blue cream**.

Tom

Sam

Transmitting Keys without Sharing

The crucial part of the process is that Sam and Tom each mix their own secret color key together with their mutually shared colored key resulting in **orange tan** and **light blue** mixtures.

Tom

Sam

Transmitting Keys without Sharing

Then publicly exchange the two mixed colors key.

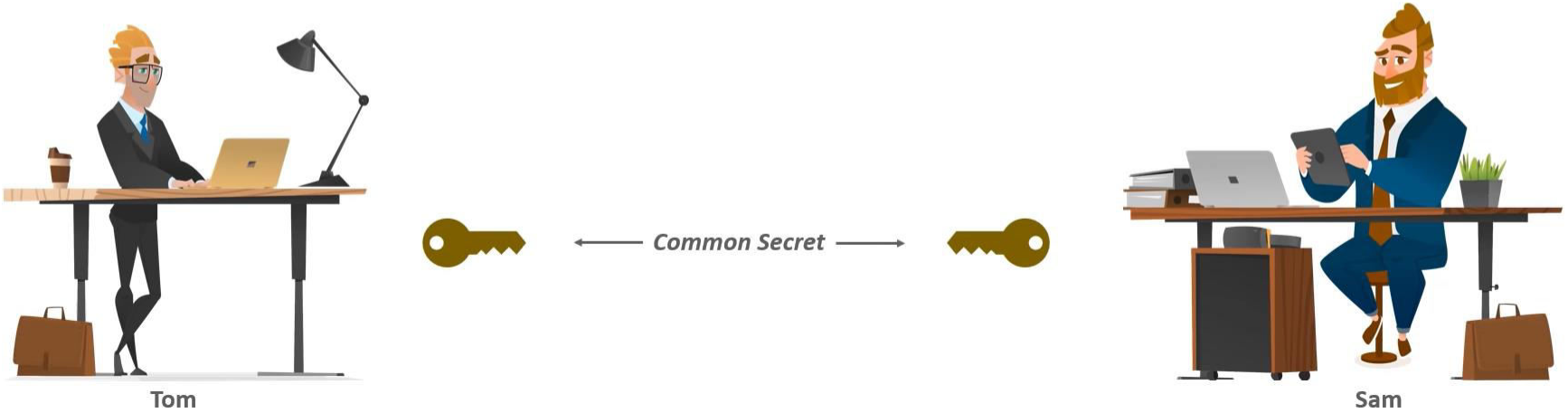
Tom

Sam

Transmitting Keys without Sharing

Finally, each of the two mixed colored keys are received from the partner and the result is **yellow brown**.

If a third party listens to the exchange it would be **computationally difficult** for this party to determine the secret colors.



Agree upon two numbers:

P Prime Number 13

G Generator of P 6

Private = 5

Randomly generate a Private Key

Private = 4

(6^{5}) MOD 13

(7776) MOD 13

Public = 2

Calculate Public Key:

$(G^{Private})$ MOD P

(6^{4}) MOD 13

(1296) MOD 13

Public = 9

Exchange Public Keys

(9^{5}) MOD 13

(59049) MOD 13

Shared Secret = 3

Calculate the Shared Secret

$(Shared\ Public^{Private})$ MOD P

(2^{4}) MOD 13

(16) MOD 13

Shared Secret = 3

PRACTICAL NETWORKING .NET

Working of Diffie-Hellman

- Finding private and public key of sender and receiver.
- Working steps:
  - ✓ Select/Assume two numbers P and Q, where P is a prime number and Q is primitive root of P (Q < P).
  - ✓ Let ‘a’ and ‘b’ be private key for Alice & Bob respectively (a&b < P).
  - ✓ Alice computes shared key as $A = Q^a \bmod P$ and sends A to Bob.
  - ✓ Bob computes shared key as $B = Q^b \bmod P$ and sends B to Alice.
  - ✓ Compute/Generate shared, private common key as follows:
    - $K_a = B^a \bmod P$, $K_b = A^b \bmod P$
  - ✓ Algebraically, it can be shown that $k_a = k_b$

Illustration

- Alice and Bob get public numbers.
  - ✓ P = 23, Q = 9
- Alice and Bob randomly select private key.
  - ✓ a = 4, b = 3
- Alice and Bob compute public key values.
  - ✓ A = 9^4 mod 23 = 6561 mod 23 = 6
  - ✓ B = 9^3 mod 23 = 729 mod 23 = 16
- Alice and Bob exchange public keys (i.e., 6 & 16).
- Alice and Bob compute symmetric keys as follows:
  - ✓ $K_a = B^a \bmod p = 16^4 \bmod 23 = 9$
  - ✓ $K_b = A^b \bmod p = 6^3 \bmod 23 = 9$
- Alice and Bob now can talk securely!
- So, the key used for encryption is 9 by both parties.

Example 2

Suppose Alice and Bob choose P=227 and Q=5. If Alice's secret number is 12 and Bob's is 3, what is the shared secret key?

- ✓ A generates key = 5^{12} mod 227 = 82, sent to B.
- ✓ Similarly, B's key = 5^{3} mod 227 = 125, sent to A.
- ✓ A computes secret key, $K_a = 125^{12}$ mod 227
- ✓ B computes secret key, $K_b = 82^{3}$ mod 227

- ✓ K_a & K_b will be same.

Example 3

Alice and Bob unwisely choose P = 211 for their Diffie-Hellman protocol, along with Q = 2. Eve sees the transmission $Q^n \pmod P = 155$ and the transmission $Q^m \pmod P = 96$.

What is the shared secret key Q^(mn)(mod P)?

Uses of Diffie-Hellman

- Communication can take place through an insecure channel.
- Public Key Infrastructure (PKI)
- Secure Socket Layer (SSL)
- Transport Layer Security (TLS)
- Secure Shell (SSH)
- Internet protocol security (IPsec)

Limitations of Diffie-Hellman

- Does not authenticate either party involved in the exchange.
- It cannot be used:
  - for asymmetric exchange
  - to encrypt messages
  - to digital signature
- Sensitive to man-in-the-middle attacks.

Thank You!

Next Class

- Mandatory reading for the next class
  - https://www.researchgate.net/publication/220709516_A_Comparative_Study_of_Elgamal_Based_Cryptographic_Algorithms

Demo Video!

Dr. Sivaraman E

Computer Science and Engineering

PES University, Bengaluru

PESU Center for Information Security, Forensics and Cyber Resilience

PESU IoT

PESU Center for Internet of Things

PESU Center for
Information Security,
Forensics and
Cyber Resilience

PES
UNIVERSITY

Welcome to

PES University

Ring Road Campus, Bengaluru

PESU Center for
Information Security,
Forensics and
Cyber Resilience

PES
UNIVERSITY

APPLIED CRYPTOGRAPHY

Lecture 33

ElGamal Cryptosystem

ElGamal Encryption Algorithm

- Presented by Taher Elgamal in 1984 based on D-H.
- Uses asymmetric key encryption for communication.
- Based on difficulty of computing discrete logarithms (DL).
- However, it is a hard problem to solve for large primes p.
- One-way function, separate functions for encryption and decryption.

ElGamal Encryption Algorithm

- We will also take a look at the ElGamal public key cipher system for a number of reasons:
  - To show that RSA is not the only public key system.
  - To exhibit a public key system based on a different one-way function.
  - ElGamal is the basis for several well-known cryptographic primitives.

ElGamal Cryptosystem: Definition

- We characterize the cryptosystem as the 5-tuple (M, C, K, E, D) where $M \in \sum^*$ is the plaintext message Alice wants to transmit to Bob.
- The plaintext message can be split in numerous blocks indicated as m_i.
- The ciphertext $C \in \sum^*$ is the result of an encryption $E_k(P)$: C, which takes the plaintext and an extra key K, and computes the ciphertext from it utilizing a satisfactory algorithm.
- Bob, who gets C, utilizes it with a decryption function $D_K(C)$: P, which under use of the key K results in the plaintext.

Mathematical Step 1 - Key Generator

In ElGamal, only the receiver needs to create a key in advance and publish it.

Bob generates the public/private key pair.

1) Chooses a large prime **p** and generator **g** of the finite cyclic group (e.g., G = $(Z_p)^*$) of the integers modulo **p**. ***Note:*** **g** is a primitive root of **p**.
2) Private key selection - Select a random integer **b**, $1 \leq a \leq p - 2$
3) Public key assembling - Calculate public key A of Bob $\mathbf{g^b}$ **mod p** which is a triplet (p, g, g^b) and his private key is b.
4) Bob publishes his public key in key directory;

Mathematical Step 2 – Encryption

Alice encrypts a message M to Bob (using Bob's public key).

1) Obtain Bob's authentic public key part (p, g, g^b) from the key directory/server.
2) Prepare the message **M** for encoding as a set of integers (m1,m2, ….) in the range of 1,….., p − 1.
3) Select a random exponent (integer) **k**, 1 ≤ k ≤ p − 2. Randomness is a crucial factor here.
4) Compute public key to transmit random exponent **k** to Bob, $\mathbf{C_1 = g^k \bmod p}$
5) Encrypt the plaintext **M** to ciphertext **C.** Iterates over the set created in Step 2 and calculates for each the m_i, $\mathbf{c_i = m_1 * (g^b)^k}$.
6) The ciphertext **C** is the set of all c_i with $0 < i \leq |M|$ is sent to Bob together with the public key $\mathbf{g^k \bmod p}$ derived from k.

Mathematical Step 3 – Decryption

After receiving the ciphertext C and the randomized public key $\mathbf{g^k}$, Bob must use the encryption algorithm to read the plaintext M.

1) Compute shared secret key (combination of Bob's private exponent b and Alice's random exponent) and it is defined by the following: $(g^k)^{p-1-b} = (g^k)^{-b} = b^{-bk}$

2) Decryption: For each of the ciphertext parts c_i Bob now computes the plaintext using, $m_i = (g^k)^{-b} \star c_i \; mod \; p$

3) After combining all, the m_i back to M he can read the message sent by Alice.

Illustration

1. Setup

- Let **p** = **23**, select a generator **g** = **11**
- Choose a private key **b** = 6
- Compute A = 11^6 (mod **23**) = 9
- Public key: **9**; Private key: **6**

2. Encryption

To encrypt M = 10 using public key **9**.

- Generate a random number **k** = **3**.
- Compute

  $C_1 = 11^3 \bmod 23 = 20$

  $C_2 = 10 * 9^3 \bmod 23 = (10*16) \bmod 23 = 22$
- Ciphertext C = (20, 22)

2. Decryption

To decrypt C = (20, 22)

- Compute $20^6 = 16 \bmod 23$
- Compute 22/16 = 10 mod **23**
- Plaintext = 10

Problem 2

- Let **p** = 19, **g** = 10
- Alice chooses **b** = 5,
- Bob wants to sent message **M**=17, selects a random key **k**=6.
- Calculate $\mathbf{C_1}$ & $\mathbf{C_2}$. Also prove that the plaintext is 15 from the calculated ciphertext.

Solution:

- C_1=11, C_2=5

Thank You!

Next Class

- ☞ Mandatory reading for the next class
  - ☞ https://psmag.com/news/prime-numbers-keep-our-information-safe

Dr. Sivaraman E

Computer Science and Engineering

PES University, Bengaluru

PESU Center for Information Security, Forensics and Cyber Resilience

PESU IoT

PESU Center for Internet of Things

PESU Center for
Information Security,
Forensics and
Cyber Resilience

PES
UNIVERSITY

Welcome to

PES University

Ring Road Campus, Bengaluru

PESU Center for
Information Security,
Forensics and
Cyber Resilience

PES
UNIVERSITY

APPLIED CRYPTOGRAPHY

Lecture 34

Prime Factorization

Prime Numbers

- Prime numbers only have divisors of 1 and self.
- They cannot be written as a product of other numbers
- **Note:** 1 is prime, but is generally not of interest for e.g.: 2,3,5,7 are prime, 4,6,8,9,10 are not prime numbers are central to number theory.
- List of prime number less than 200 is:

2 3 5 7 11 13 17 19 23 29 31 37 41 43 47 53 59 61 67 71 73 79 83 89 97 101 103 107 109 113 127 131 137 139 149 151 157 163 167 173 179 181 191 193 197 199

Prime Factorization

- To factor a number n is to write it as a product of other numbers: n=a x b x c.
- Note that factoring a number is relatively hard compared to multiplying the factors together compared to multiplying the factors together to generate the number
- The prime factorisation of a number n is when its written as a product of primes.
- e.g. 91 = 7 x 13 ; 3600 = 24 x 32 x 52

Relatively Prime Numbers and GCD

- Two numbers a, b are relatively prime if have no common divisor apart from 1.
- Ex. 8 & 15 are relatively prime since factors of 8 are 1,2,4,8 and of 15 are 1,3,5,15 and 1 is the only common factor.
- Conversely can determine the greatest common divisor by comparing their prime factorizations and using least powers.

e.g., 300 = 21x31x52 18=21x32

Hence, GCD (18,300)=21 x 31 x 50 = 6

Euler Totient Function ø(n)

- When doing arithmetic modulo n
- Complete set of residues is: 0..n-1
- Reduced set of residues is those numbers (residues) which are relatively prime to n.

  e.g. for n=10,
- Complete set of residues is {0,1,2,3,4,5,6,7,8,9}.
- Reduced set of residues is {1,3,7,9}.
- Number of elements in reduced set of residues is called the Euler Totient Function ø(n).

Euler's phi-function

Euler's phi-function, f (n), which is sometimes called the Euler's totient function plays a very important role in cryptography.

1. $\phi(1) = 0$.
2. $\phi(p) = p - 1$ if p is a prime.
3. $\phi(m \times n) = \phi(m) \times \phi(n)$ if m and n are relatively prime.
4. $\phi(p^e) = p^e - p^{e-1}$ if p is a prime.

Continued..

We can combine the above four rules to find the value of f(n). For example, if n can be factored as $\mathbf{n = p_1^{e1} \times p_2^{e2} \times \ldots \times p_k^{ek}}$ then we combine the third and the fourth rule to find

$$\phi(n) = (p_1^{e_1} - p_1^{e_1-1}) \times (p_2^{e_2} - p_2^{e_2-1}) \times \cdots \times (p_k^{e_k} - p_k^{e_k-1})$$

The difficulty of finding f(n) depends on the difficulty of finding the factorization of *n*.

Examples

What is the value of $\phi(13)$?

Solution :

Because 13 is a prime, $f(13) = (13 - 1) = 12$.

What is the value of $\phi(10)$?

Solution :

We can use the third rule: $f(10) = f(2) \times f(5) = 1 \times 4 = 4$, because 2 and 5 are primes.

Examples

What is the value of f(240)?

Solution :

We can write $240 = 2^4 \times 3^1 \times 5^1$. Then, $f(240) = (2^4 - 2^3) \times (3^1 - 3^0) \times (5^1 - 5^0) = 64$

Can we say that f(49) = f(7) × f(7) = 6 × 6 = 36?

Solution :

No. The third rule applies when *m* and *n* are relatively prime.

Here $49 = 7^2$.

We need to use the fourth rule: $f(49) = 7^2 - 7^1 = 42$.

Examples

What is the number of elements in Z_{14}^*?

Solution :

The answer is f(14) = f(7) × f(2) = 6 × 1 = 6. The members are 1, 3, 5, 9, 11, and 13.

Interesting point: If $n > 2$, the value of $\phi(n)$ is even.

Thank You!

Next Class

- Mandatory reading for the next class
  - https://www.ijcsmc.com/docs/papers/June2013/V2I6201330.pdf

Dr. Sivaraman E

Computer Science and Engineering

PES University, Bengaluru

PESU Center for Information Security, Forensics and Cyber Resilience

PESU IoT

PESU Center for Internet of Things

PESU Center for
Information Security,
Forensics and
Cyber Resilience

PES
UNIVERSITY

Welcome to

PES University

Ring Road Campus, Bengaluru

PESU Center for
Information Security,
Forensics and
Cyber Resilience

PES
UNIVERSITY

APPLIED CRYPTOGRAPHY

Lecture 35

RSA Public Key Cryptosystem

RSA Algorithm

- Invented in **1978** by Ron **R**ivest, Adi **S**hamir and Leonard **A**dleman.
- **Published as** R L Rivest, A Shamir, L Adleman, "*On Digital Signatures and Public Key Cryptosystems*", Communications of the ACM, vol 21 no 2, pp120-126, Feb 1978.
- Based on the hardness of factoring large numbers.
- RSA uses two exponents, e and d, where e is public, and d is private.
- Encryption and decryption use modular exponentiation.

RSA Cryptosystem

Key Generation (Bob):

1) Select 2 large prime numbers of about the same size, **p** & **q**.
   - Typically, each p, q has between 512 and 2048 bits.
2) Compute **n = pq**, and **Φ(n) = (q-1)(p-1)**, here Φ is Euler' Totient function
3) Select **e**, 1<e< Φ(n), such that **gcd(e, Φ(n)) = 1**
   - Typically, e=3 or e=65537
4) Compute **d**, 1< d< Φ(n) s.t. **e*d ≡ 1 mod Φ(n)** or **$d=e^{-1} \bmod \Phi$**
   - Knowing Φ(n), d easy to compute.

- **Public key: tuple (e, n)**
- **Private key: d**

RSA Cryptosystem – Key Generation Algorithm

RSA_Key_Generation

{

Select two large primes p and q such that $p \neq q$.

$n \leftarrow p \times q$

$\phi(n) \leftarrow (p-1) \times (q-1)$

Select e such that $1 < e < \phi(n)$ and e is coprime to $\phi(n)$

$d \leftarrow e^{-1} \bmod \phi(n)$ // d is inverse of e modulo $\phi(n)$

Public_key $\leftarrow$ (e, n) // To be announced publicly

Private_key $\leftarrow d$ // To be kept secret

return Public_key and Private_key

}

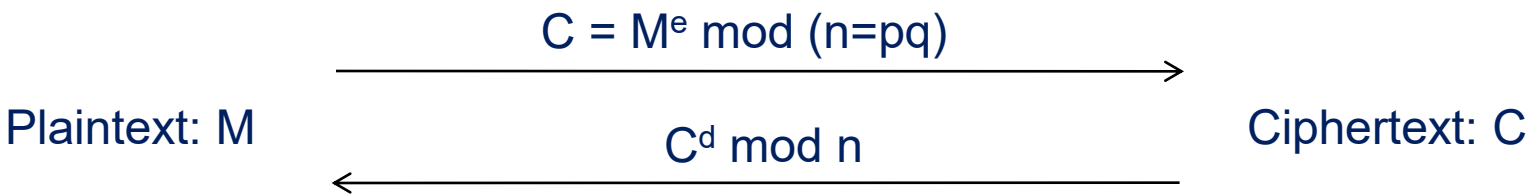
RSA Cryptosystem

Encryption:

- Given a message M, $0 < M < n$
- Use public key (e, n)
- Compute $C = M^e \bmod n$

Decryption:

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



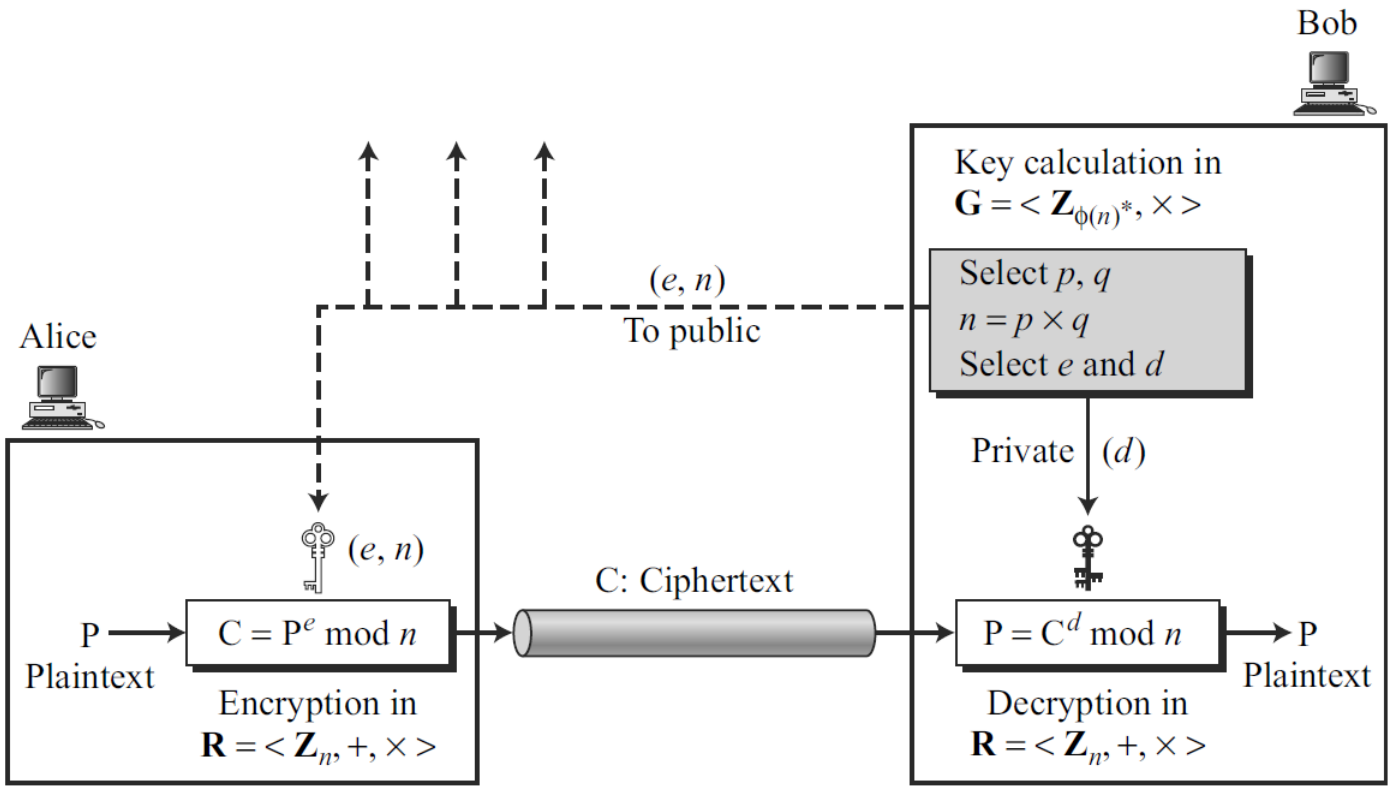
RSA Cryptosystem – Encryption & Decryption Algorithm

```
RSA_Encryption (P, e, n)                    // P is the plaintext in Z_n and P < n
{
    C ← Fast_Exponentiation (P, e, n)       // Calculation of (P^e mod n)
    return C
}
```

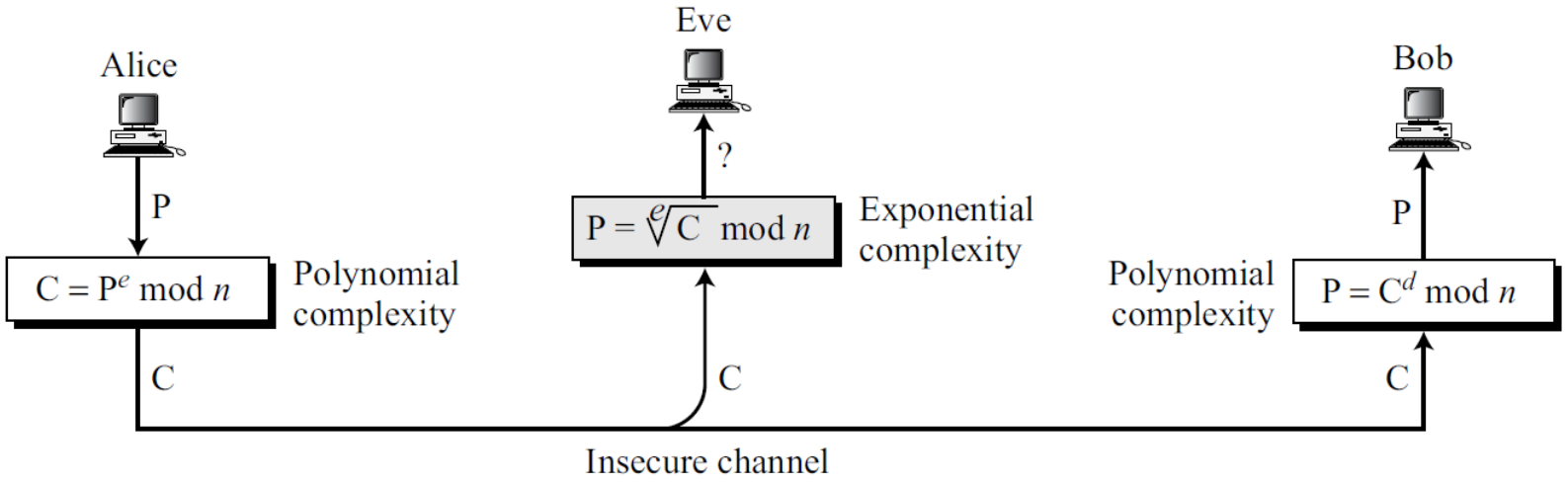
```
RSA_Decryption (C, d, n)                    //C is the ciphertext in Z_n
{
    P ← Fast_Exponentiation (C, d, n)       // Calculation of (C^d mod n)
    return P
}
```

In RSA, *p* and *q* must be at least 512 bits; *n* must be at least 1024 bits.

RSA Cryptosystem – Procedure



Complexity of operations in RSA



- **Alice can encrypt in polynomial time (e is public), Bob also can decrypt in polynomial time (because he knows d), but Eve cannot decrypt because she would have to calculate the *e*th root of C using modular arithmetic.**
- **In other words, Alice uses a one-way function (modular exponentiation) with a trapdoor known only to Bob. Eve, who does not know the trapdoor, cannot decrypt the message.**

RSA uses modular exponentiation for encryption/decryption;
To attack it, Eve needs to calculate $\sqrt[e]{C} \bmod n$.

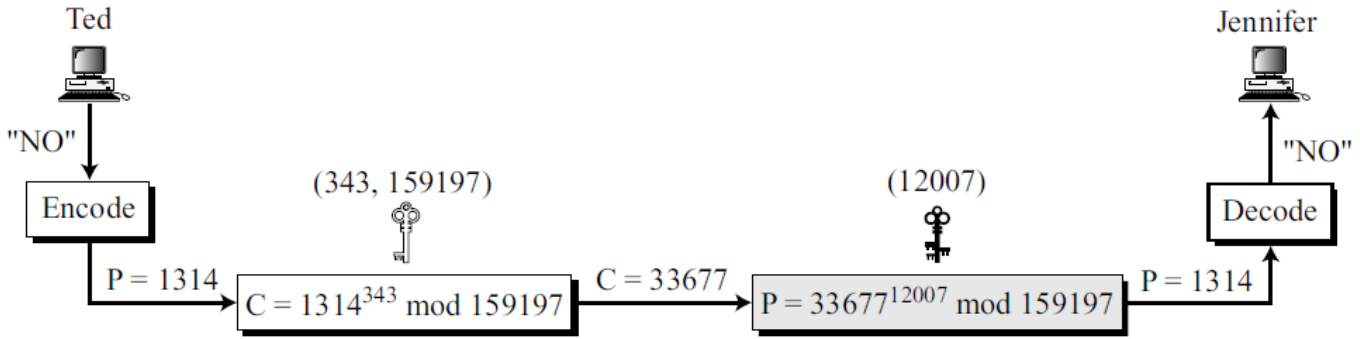
RSA Example 1

- Bob chooses p = 11, q = 7, n = 77, $\Phi(n) = 60$
- d = 13, e = 37 (ed = 481; ed mod 60 = 1)
  - d & e are inverses of each other
- Let M = 15. Then $C \equiv M^e \bmod n$
  - $C \equiv 15^{37} \pmod{77} = 71$
- $M \equiv C^d \bmod n$
  - $M \equiv 71^{13} \pmod{77} = 15$

RSA Example 2

Jennifer creates a pair of keys for herself. She chooses p = 397 and q = 401. She calculates n = 397 × 401= 159197. She then calculates ϕ(n) = 396 × 400 = 158400. She then chooses e = 343 and d = 12007. Show how Ted can send a message 'NO' to Jennifer if he knows e and n.

Solution:



Limitations

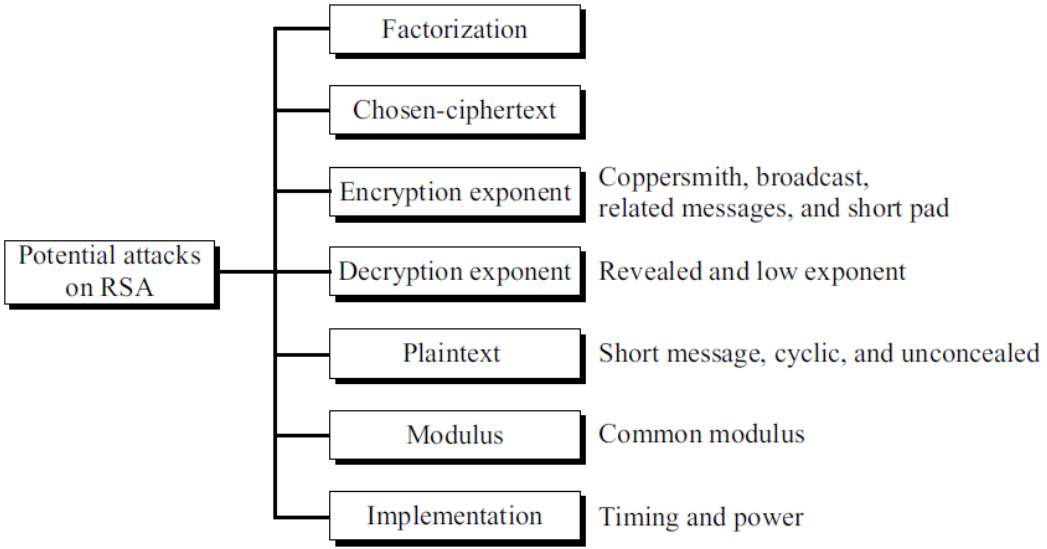
- Not using small primes.
- Not using primes that are very close.
- Two people must not use the same N.
- Message should not be observable of eth power.

Attacks against RSA

- Brute Force
  - Try all possible keys
- Mathematical Attacks
  - Factor n
  - Calculate Φ(n)
- Timings Attacks
  - Use the running time of the algorithm to determine d, the decryption key
- Protocol Attacks

Taxonomy of Potential Attacks (Assignment)

- No devastating attacks on RSA have been yet discovered.
- Several attacks have been predicted based on the **weak plaintext**, **weak parameter selection**, or **inappropriate implementation**.



Thank You!

Next Class

- ☞ Mandatory reading for the next class
  - ☞ https://people.csail.mit.edu/rivest/Rsapaper.pdf

Dr. Sivaraman E

Computer Science and Engineering

PES University, Bengaluru

PESU Center for Information Security, Forensics and Cyber Resilience

PESU IoT

PESU Center for Internet of Things

PESU Center for
Information Security,
Forensics and
Cyber Resilience

PES
UNIVERSITY

Welcome to

PES University

Ring Road Campus, Bengaluru

PESU Center for
Information Security,
Forensics and
Cyber Resilience

PES
UNIVERSITY

APPLIED CRYPTOGRAPHY

Lecture 36

Applications

One-Way Functions and Permutations

DEFINITION 7.66 *A function $f : \{0,1\}^* \to \{0,1\}^*$ whose output length is polynomially-related to its input length*[9] *is* one-way *if the following two conditions hold:*

1. Easy to compute: *There exists a polynomial-time algorithm that on input $x \in \{0,1\}^*$ outputs $f(x)$.*

2. Hard to invert: *Consider the following experiment for a given algorithm $\mathcal{A}$ and parameter n:*

   The inverting experiment $\mathsf{Invert}_{\mathcal{A},f}(n)$

   (a) Choose input $x \leftarrow \{0,1\}^n$. Compute $y := f(x)$.
   (b) $\mathcal{A}$ is given y as input, and outputs x'.
   (c) The output of the experiment is defined to be 1 if $f(x') = y$, and 0 otherwise.

   Then it is required that for all probabilistic, polynomial-time algorithms $\mathcal{A}$ there exists a negligible function negl *such that*

$$\Pr[\mathsf{Invert}_{\mathcal{A},f}(n) = 1] \leq \mathsf{negl}(n).$$

Continued..

DEFINITION 7.69 *A tuple* $\Pi = (\mathsf{Gen}, \mathsf{Samp}, f)$ *of probabilistic, polynomial-time algorithms is a* family of functions *if the following hold:*

1. *The* parameter generation algorithm Gen, *on input* 1^n, *outputs parameters* I *with* $|I| \geq n$. *Each value of* I *output by* Gen *defines sets* $\mathcal{D}_I$ *and* $\mathcal{R}_I$ *that constitute the domain and range, respectively, of a function we define next.*

2. *The* sampling algorithm Samp, *on input* I, *outputs a uniformly distributed element of* $\mathcal{D}_I$ *(except possibly with probability negligible in* $|I|$*).*

3. *The deterministic* evaluation algorithm f, *on input* I *and* $x \in \mathcal{D}_I$, *outputs an element* $y \in \mathcal{R}_I$. *We write this as* $y := f_I(x)$.

Π *is a* family of permutations *if, for each value of* I *output by* $\mathsf{Gen}(1^n)$, *it additionally holds that* $\mathcal{D}_I = \mathcal{R}_I$, *and the function* $f_I : \mathcal{D}_I \to \mathcal{D}_I$ *is a bijection.*

Continued..

DEFINITION 7.70 *Let* $\Pi = (\mathsf{Gen}, \mathsf{Samp}, f)$ *be a family of functions.* Π *is* one-way *if for all probabilistic, polynomial-time algorithms* $\mathcal{A}$ *there exists a negligible function* negl *such that*

$$\Pr[\mathsf{Invert}_{\mathcal{A},\Pi}(n) = 1] \leq \mathsf{negl}(n).$$

CONSTRUCTION 7.71

- *Parameter-generation algorithm* Gen: on input 1^n, run $\mathsf{GenRSA}(1^n)$ to obtain (N, e, d) and output $I = \langle N, e \rangle$. We will have $\mathcal{D}_I = \mathbb{Z}_N^*$.
- *Sampling algorithm* Samp: on input $I = \langle N, e \rangle$, choose a random element of $\mathbb{Z}_N^*$.
- *Evaluation algorithm* f: on input $I = \langle N, e \rangle$ and $x \in \mathbb{Z}_N^*$, outputs $[x^e \bmod N]$.

A family of one-way permutations (assuming the RSA problem is hard relative to GenRSA).

Constructing Collision-Resistant Hash Functions

CONSTRUCTION 7.72

Let $\mathcal{G}$ be as described in the text. Define (Gen, H) as follows:

- *Key generation algorithm* Gen: On input 1^n, run $\mathcal{G}(1^n)$ to obtain $(\mathbb{G}, q, g)$ and then select $h \leftarrow \mathbb{G}$. Output $s = \langle \mathbb{G}, q, g, h \rangle$.
- *Hash algorithm* H: On input $s = \langle \mathbb{G}, q, g, h \rangle$ and message $(x_1, x_2) \in \mathbb{Z}_q \times \mathbb{Z}_q$, output $g^{x_1} h^{x_2} \in \mathbb{G}$.

A fixed-length hash function.

Thank You!

Next Class

- Mandatory reading for the next class
  - https://seedsecuritylabs.org/Labs_16.04/Crypto/Crypto_RSA/

Dr. Sivaraman E

Computer Science and Engineering

PES University, Bengaluru

PESU Center for Information Security, Forensics and Cyber Resilience

PESU IoT

PESU Center for Internet of Things