Cryptography and cyber Law

Assignment - 05

Abida Sultana

IT-21032

Q1) Prove Fermat's Little Theorem and use it to compute $a^{p-1} \bmod p$ for given values of $a=7$, $p=13$. Then, discuss how this theorem is useful in cryptographic algorithms like RSA.

Answer:

If p is a prime number and a is any integer such that $\gcd(a,p)=1$, then:

$$a^{p-1} \equiv 1 \bmod p$$

Proof:

Let $S=\{1,2,3,\ldots,p-1\}$ be the set of integers modulo p, excluding 0.

Multiply each element by a, where $\gcd(a,p)=1$, to form a new set:

$$S'=\{a\cdot 1, a\cdot 2, \ldots, a\cdot(p-1)\} \bmod p$$

Since a has an inverse modulo p, all elements in S' are distinct modulo p, and S' is just a rearrangement of S.

Thus:

$$a \cdot 2a \cdot 3a \cdots (p-1)a \equiv 1 \cdot 2 \cdot 3 \cdots (p-1) \bmod p$$

$$a^{p-1} \cdot (p-1)! \equiv (p-1)! \bmod p$$

Cancelling $(p-1)!$ from both sides (allowed since $\gcd((p-1)!, p) = 1$):

$$a^{p-1} \equiv 1 \bmod p$$

This proves Fermat's Theorem.

Using the Theorem to compute $a^{p-1} \bmod p$:

Given: $a = 7, p = 13$

Since 13 is a prime and $\gcd(7, 13) = 1$, by Fermat's Little Theorem:

$$7^{13-1} = 7^{12} \equiv 1 \bmod 13$$

$$\Rightarrow 7^{12} \bmod 13 = 1$$

Importance in Cryptographic Algorithms like RSA :-

Fermat's Little Theorem is foundational in number theory and crucial for cryptography, especially in the RSA algorithm. Here's how :

- RSA Encryption/Decryption uses modular exponentiation :

$$c = m^e \mod n, \quad m = c^d \mod n$$

The correctness relies on the fact that :

$$m^{ed} \equiv m \mod n$$

when $ed \equiv 1 \mod \phi(n)$, where $\phi(n)$ is Euler's totient function.

- Fermat's Theorem (as a special case of Euler's Theorem) helps ensure that :

$$m^{p-1} \equiv 1 \mod p$$

which is used when n is a product of two

primes p and q.

- key Generation and Modular inverses: The theorem helps in computing modular inverses which are required to find the decryption key d.

Fermat's Little Theorem simplifies exponentiation in modular arithmetic and forms the mathematical backbone of public-key ~~cryptor~~ cryptographic algorithms like RSA ensuring secure data transmission.

Q2. Euler Tootient function : compute $\phi(n)$ for $n = 35, 45, 100$. Prove that if a and n are comprime, then $a^{\phi(n)} \equiv 1 \bmod n$.

Answer :- Given that,

$$n = 35, \quad n = 45, \quad n = 100$$

And prove that if a and n are comprime then : $a^{\phi(n)} \equiv 1 \bmod n$

Step 1: Compute Euler's Totient Function $\phi(n)$;

Euler's totient function for :

$$n = p_1^{k_1} \cdot p_2^{k_2} \cdot \cdots \cdot p_r^{k_r} \text{ is}$$

$$\phi(n) = n\left(1 - \frac{1}{p_1}\right)\left(1 - \frac{1}{p_2}\right) \cdots \left(1 - \frac{1}{p_r}\right)$$

For $n = 35$:

• Prime factorization; $35 = 5 \cdot 7$

$$\phi(35) = 35\left(1 - \frac{1}{5}\right)\left(1 - \frac{1}{7}\right) = 35 \cdot \frac{4}{5} \cdot \frac{6}{7} = 24$$

For $n = 45$

• Prime factorization :

$$45 = 3^2 \cdot 5$$

$$\phi(45) = 45\left(1-\frac{1}{3}\right)\left(1-\frac{1}{5}\right) = 45 \cdot \frac{2}{3} \cdot \frac{4}{5} = 24$$

For $n = 100$:

prime factorization: $100 = 2^2 \cdot 5^2$

$$\phi(100) = 100\left(1-\frac{1}{2}\right)\left(1-\frac{1}{5}\right) = 100 \cdot \frac{1}{2} \cdot \frac{4}{5} = 40$$

Step 2: Proof of Euler's theorem

Theorem: If $\gcd(a,n) = 1$, then:

$$a^{\phi(n)} \equiv 1 \bmod n$$

- Let $A = \{x \in \mathbb{Z}_n^* \mid \gcd(x,n) = 1\}$ be the group of units modulo n.
- Then A has $\phi(n)$ elements
- Since $a \in A$, the function $b(x) = ax \bmod n$ is a bijection on A.
- So the product $\prod_{x \in A} x \equiv \prod_{x \in A} ax \bmod n$
- $\prod_{x \in A} ax = a^{\phi(n)} \prod_{x \in A} x$
- Cancel $\prod x$ from both sides (it's invertible mod n):

$$a^{\phi(n)} \equiv 1 \bmod n$$

Q3. Solve the system of congruences using the Chinese Remainder Theorem and prove that x congruent to 11 on mod $N = 3 \times 4 \times 5 = 60$

$$x \equiv 2 \bmod 3 \quad x \equiv 3 \bmod 4 \quad x \equiv 1 \bmod 5$$

Given, $x \equiv 2 \bmod 3$

$x \equiv 3 \bmod 4$

$x = 1 \bmod 5$

Let: $n_1 = 3, n_2 = 4, n_3 = 5$

$N = n_1 \cdot n_2 \cdot n_3 = 3 \cdot 4 \cdot 5 = 60$

Step 1 :- Compute $N_i = \frac{N}{n_i}$

- $N_1 = \frac{60}{3} = 20$
- $N_2 = \frac{60}{4} = 15$
- $N_3 = \frac{60}{5} = 12$

Step 2 : Find y_i such that:

$N_i \cdot y_i \equiv 1 \bmod n_i$

• $20 \cdot y_i \equiv 1 \mod 3 \rightarrow 20 \equiv 2 \mod 3$, so

solve $2y_1 \equiv 1 \mod 3 \rightarrow y_1 = 2$

• $15 \cdot y_2 \equiv 1 \mod 4 \rightarrow 15 \equiv 3 \mod 4$,

so solve $3y_2 \equiv 1 \mod 4 \rightarrow y_2 = 3$

• $12 \cdot y_3 \equiv 1 \mod 5 \rightarrow 12 \equiv 2 \mod 5$,

so solve $2y_3 \equiv 1 \mod 5 \rightarrow y_3 = 3$

Step 3: compute the solution:

$$x \equiv a_1 N_1 y_1 + a_2 N_2 y_2 + a_3 N_3 y_3 \mod N$$

where:

• $a_1 = 2, a_2 = 3, a_3 = 1$

$x \equiv (2)(20)(2) + (3)(15)(3) + (1)(12)(3) \mod 60$

$x \equiv 80 + 135 + 36 = 251 \mod 60$

$x \equiv 251 \mod 60 \Rightarrow x \equiv 11 \mod 60$

• $11 \mod 3 = 2$

• $11 \mod 4 = 3$

• $11 \mod 5 = 1$

$\therefore x \equiv 11 \mod 60$

Q4. Find whether 561 is a Carmichael number by checking its divisibility and Fermat's test.

Answer:

A Carmichael number is a composite number n such that:

$$a^n \equiv a \mod n$$

for all integers a such that gcd(a,n)=1

It passes Fermet's Little Theorem for all such a, even though it is not prime.

* Step 1:- check it 561 is composite

We factor 561: $561 = 3 \times 11 \times 17$

So, 561 is composite

Step 2: check it 561 is square-free

A number is square-free it no prime factor repeats.

$$561 = 3^1 \cdot 11^1 \cdot 17^1$$

Each prime has exponent 1 → 561 is square-free

Step 3:- Apply Fermat's Test for Each Prime Factor:

Let's check whether for each prime factor p of 561.

$(p-1) \mid (561-1 = 560)$

- For $p = 3$, $p-1 = 2 \Rightarrow 2 \mid 560$
- For $p = 11$, $p-1 = 10 \Rightarrow 10 \mid 560$
- For $p = 17$, $p-1 = 16 \Rightarrow 16 \mid 560$

All prime divisors satisfy $(p-1) \mid 560$

For each prime p dividing 561, $p-1 \mid 560$

Therefore, 561 is a Carmichael numbe

Q5. Find a generator (primitive root) of the multiplicative group modulo 17.

Answer:

We want to find a primitive root modulo 17, that is, a number g such that:

$$\{g^1, g^2, g^3, \ldots, g^{16}\} \bmod 17$$

produces all numbers from 1 to 16 without repetition.

Step 1:- Euler's Totient Function

Since 17 is a prime number,

$$\phi(17) = 17 - 1 = 16$$

So, the order of any primitive root modulo 17 must be 16

Step 2: Prim factors of 16

$16 = 2^4 \Rightarrow$ Prime factor is 2

To test whether a number g is a primitive root modulo 17, check:

$$g^{16/2} = g^8 \not\equiv 1 \mod 17$$

Step 3: Try $g = 2$

$2^8 = 256 \Rightarrow 256 \bmod 17 = 1$

So, $g = 2$ is not a primitive root becaa
its order is 8

Step 4: Try $g = 3$

$3^8 = 6561 = 3^8 \bmod 17 = 16 \neq 1$

Now, let's compute all powers of 3 modul
17:

$3^1 \equiv 3 \bmod 17$

$3^2 \equiv 9 \bmod 17$

$3^2 \equiv 10 \bmod 17$

$3^4 \equiv 13 \bmod 17$

$3^5 \equiv 5 \bmod 17$

$3^6 \equiv 15 \bmod 17$

$3^7 \equiv 11 \bmod 17$

$3^8 \equiv 16 \bmod 17$

$3^9 \equiv 14 \bmod 17$

$3^{10} \equiv 8 \bmod 17$

$3^{11} \equiv 7 \bmod 17$

$3^{12} \equiv 4 \mod 17$

$3^{13} \equiv 12 \mod 17$

$3^{14} \equiv 2 \mod 17$

$3^{15} \equiv 6 \mod 17$

$3^{16} \equiv 1 \mod 17$

All $\{1, 2, \ldots 16\}$ appeared once $\Rightarrow$ order is 16.

$\therefore$ 3 is a primitive root modulo 17

Q6. Solve the Discrete Logarithm Problem:
Find x such that $3^x \equiv 13 \mod 17$

Answer:
To solve the discrete logarithm problem $3^x \equiv 13 \mod 17$, we need to find the value of x.
We can do this by computing the powers of 3 modulo 17 until we reach 13.

$3^1 \equiv 3 \mod 17$

$3^2 \equiv 9 \mod 17$

$3^3 \equiv 3 \cdot 9 \equiv 27 \equiv 10 \mod 17$

$3^4 \equiv 3 \cdot 10 \equiv 30 \equiv 13 \mod 17$

From the calculations, we can see that $3^4 \equiv 13 \mod 17$.

Therefore, $x = 4$

Q7. Discuss the role of the discrete logarith in the Diffie-Hellman key Exchange.

Answer:

Role of Discrete logarithm in Diffie-Hellman key Exchange –

1. Public Parameters: Large prime p, generator g.

2. key Exchange:

- Alice sends $A = g^a \mod p$
- Bob sends $B = g^b \mod p$
- Shared key: $s = g^{ab} \mod p$

3. Discrete Logarithm Problem (DLP):

- Hard to find a from $A = g^a \bmod p$
- This difficulty ensures security

4. Attacker's challenge:

- Cannot compute shared key without solving DLP
- DLP is computationally hard for large p

Q8

Here's clear comparison of substitution cipher, Transposition cipher, and playfair cipher based on encryption mechanism, key space and vulnerability to frequency analysis along with an example:

1. Substitution cipher:

Encryption Mechanism:- Each letter is replaced by another letter.

Example :- Caesar cipher shifts each letter by a fixed number.

Key space :- For monoalphabetic : $26! \approx 4 \times 10^{26}$

Frequency Analysis vulnerability :

- Highly vulnerable : Letter frequencies remain unchanged.

Example : Plaintext : Hello
key : Caesar shift by 3
Ciphertext : KHOOK

2. Transposition Cipher :

Encryption Mechanism :

- Letters are rearranged based on a pattern or key
- No change to actual letters.

Key Space :

- Depends on message/block length :
- for length n, key space is n!

Frequency Analysis Vulnerability:
less vulnerable; Frequencies preserved, but letter positions change.

Example:

plaintext: Hello
key: Rearranged as 3-1-4-2-5
→ Rearranged to LHOEL

3. Playfair Cipher:

Encryption Mechanism:
- Encrypt digraphs (pairs of letters) using a 5×5 matrix
- Uses rules: same row, column or rectangle.

Key space: Based on 5×5 grid of letters (excluding 'J') → 25! ≈ 1.55×10^{25}

Example: HELLO → digraphs: HE, LX, LO

key: Matrix from keyword MONARCHY.

Using rules, suppose result is : ZFGRKV

Comparision between them:

Feature	Substitution	Transposition	Playfair
Mechanism	Replace letters	Rearrange letters	Encrypt letter pairs
Key space	26!	Depends on message size	25!
Frequency Analysis	Highly vulnerable	Moderately vulnerable	less vulnerable
Example (Hello)	KHOOR	LHOEL	ZFGRKV (sample.

Q9. Given the Affine Cipher encryption function $E(x) = (ax + b) \bmod 26$, where $a = 5$ and $b = 8$, (a) Encrypt the plaintext "Dept of ICT, MBSTU", (b) Derive the decryption function and decrypt the ciphertext.

Answer:

Given,

Affine Cipher encryption function:

$$E(x) = (a \cdot x + b) \bmod 26$$

Where

$a = 5$

$b = 8$

Plaintext: "Dept of ICT, MBSTU"

Step A:- Encryption

I. Preprocessing the Plaintext:

Remove punctuation and spaces, convert to uppercase:

Plaintext = "DEPTOFICTMBSTU"

2. Convert letters to numbers (A=0 to Z=25)

D=3, E=4, P=15, T=19, O=14, F=5, I=8, C=2, T=19, M=12, B=1, S=18, T=19, U=20.

3. Apply the encryption function $E(x)=(5x+8) \bmod 26$:-

Letter	x	E(x)	cipher
D	3	$(5\times3+8)\%26=23$	X
E	4	$(5\times4+8)\%26=2$	C
P	15	$(5\times15+8)\%26=1$	B
T	19	$((5\times19)+8)\%26=21$	V
O	14	$(5\times14+8)\%26=0$	A
F	5	$(5\times5+8)\%26=7$	H
I	8	$(5\times8+8)\%26=22$	W
C	2	$(5\times2+8)\%26=18$	S
T	19	$15\times19+8)\%26=21$	V
M	12	$((5\times12)+8)\%26=16$	Q

B	1	$(5\times1+8)\%26=13$	N
S	18	$(5\times18+8)\%26=16$	Q
T	19	$(5\times19)+8)\%26=21$	V
U	20	$(5\times20+8)\%26=6$	G

Step B:- Decryption

The decryption function of Affine Cipher is:

$$D(y) = a^{-1} \cdot (y-b) \bmod 26$$

Where a^{-1} is the modular inverse of $a = 5$ modulo 26.

Since: $5 \cdot 21 \equiv 105 \equiv 1 \bmod 26 \Rightarrow a^{-1} = 21$

So, the decryption function becomes:

$$D(y) = 21 \cdot (y-8) \bmod 26$$

2. Apply decryption on ciphertext:

Ciphertext: XCBVAHWSVQNQVG

converts letters to numbers:

X=23, C=2, B=1, V=21, A=0, H=7, W=22, S=18, V=21, Q=16, N=13, Q=16, V=21, G=6

Apply $D(y) = 21 \cdot (y-8) \bmod 26$;

letter	y	D(y)	Plain
X	23	$21 \times (23-8) \% 26 = 3$	D
C	2	$21 \times (2-8) \% 26 = 4$	E
B	1	$21 \times (1-8) \% 26 = 15$	P
V	21	$21 \times (21-8) \% 26 = 19$	T
A	0	$21 \times (0-8) \% 26 = 14$	O
H	7	$21 \times (7-8) \% 26 = 5$	F
W	22	$21 \times (22-8) \% 26 = 8$	I
S	18	$21 \times (18-8) \% 26 = 2$	C
V	21	$21 \times (21-8) \% 26 = 19$	T
Q	16	$21 \times (16-8) \% 26 = 12$	M
N	13	$21 \times (13-8) \% 26 = 1$	B
Q	16	$21 \times (16-8) \% 26 = 12$	S
V	21	$21 \times (21-8) \% 26 = 19$	T
G	6	$21 \times (6-8) \% 26) = 20$	U

• Encryption function: $E(x) = (5x+8) \bmod 26$

• Decreption function: $D(y) = 21(y-8) \bmod 26$

• plaintext : "Dept of ICT, MBSTU".

• Ciphertext: XCB VAH WS VQ NQVG.

• Decrypted Text: DEPTOFICTMBSTU.

Q10. Design a simple novel cipher (using a combination of substitution and permutation techniques). Describe its encrytion and decreption processes. Then, perform a basic cryptanalysis on your cipher to identify its potential ~~vunbulner~~ abilities. You may use your own PRNG technique.

Answer :-

Here's a simple novel cipher that uses a combination of substitution and permutation techniques. It also uses a custom pseudo-random number generator (PRNG) for added complexity.

Cipher Name : Sub-Perm Cipher (SPC)

Overview :

- Substitution : Each character is substituted using a keyed Caesar shift.
- Permutation : Blocks of text are permuted using a PRNG-based shuffle.
- PRNG : Custom linear congruential generator (LCG).

key :

- k_1 : Integer (used for Caesar shift)
- k2 : Seed value for PRNG
- Block size : fixed block size.

Encryption process :

Step 1 : Substitution :-

Each character c in plaintext is shifted forward using a Caesar-like method with a varying shift based on the PRNG

PRNG : $X_{n+1} = (aX_n + c) \bmod m$

parameters : $a = 17$, $c = 43$, $m = 256$

For each character c_i, compute :

$Shift_i = PRNG(k2) \bmod 26$

$c_i = (c_i + Shift_i + k_1) \bmod 26$

Step 2 : Permutation

Split the substituted ciphertext into blocks of size N.

For each block :

1. Use PRNG (same seed k2) to generate a permutation of indices
2. Permute the block accordingly.

Decryption Process :

step 1 :- Reverse Permutation

Using the same PRNG and Block size, rearrange the permutation pattern

and reverse it for each block.

Step 2 : Reverse Substitution

For each character C_i' in the block

$$shift_i = PRNG(k2) \bmod 26$$

$$C_i = (C_i' - shift_i - k1 + 26) \bmod 26$$

Example :-

Inputs :-

- Plaintext : "HELLO"
- k1 = 3, k2 = 7, Block size = 2

Step 1 : Substitution

Let's say PRNG gives shift = [5, 12, 7, 19, 2]

H → H(7) + 5 + 3 = 15 → P

E → E(4) + 12 + 3 = 19 → T

L → L(11) + 7 + 3 = 21 → V

L → L(11) + 19 + 3 = 33 → H (mod 26)

O → O(14) + 2 + 3 = 19 → T

Substituted :- "PTVHT"

Step 2:- Permutation (Block size 2)

Split : [PT] [VH] [T_]

Permutation genarated : [1, 0]

Apply permutation to each block :-

- [PT] → [TP]
- [VH] → [HV]
- [T_] → [_T] (padding with _)

Final ciphertext : "TPHV_T"