SY BTech Semester-IV (AY 2022-23)

Computer Science and Engineering (Cybersecurity and Forensics)

Disclaimer:

a. Information included in these slides came from multiple sources. We have tried our best to cite the sources. Please refer to the references to learn about the sources, when applicable.

b. The slides should be used only for preparing notes, academic purposes (e.g. in teaching a class), and should not be used for commercial purposes.

Unit II: Mathematical Foundations and Public Key Cryptography

Unit: II	**Mathematical Foundations and Public Key Cryptography:** Mathematics for Security: Modular Arithmetic, Euler's theorem, Fermat Theorem, Euclidean Algorithm, Miller-Rabin Algorithm, Primality Test, Chinese Remainder Theorem, Discrete Logarithm, Asymmetric Key Cryptography: RSA algorithms. Hash algorithms: MD5, SHA1	9 Hrs

Laboratory: Lab Assignment

Assign No.	Name of Assignment
4	Write a program using JAVA or Python or C++ to implement RSA asymmetric key algorithm.
5	Write a program using JAVA or Python or C++ to implement integrity of message using MD5 or SHA

Number Theory

- **Prime Numbers**
- **Relative Prime Numbers**

  - Two numbers are called relatively prime if the **greatest common divisor (GCD)** of those numbers is **1**.
  - 8 and 15 are relatively prime number.
  - The factors of 8 are 1, 2, 4, 8 and the factors of 15 are 1, 3, 5, 15.
  - Examples of relatively prime numbers are: (10, 21), (14, 15), (45, 91), ...

- The greatest common divisor (GCD) of two numbers can be determined by comparing their prime factors and selecting the least powers of the factor.
- For example, the two numbers are 81 and 99.

$$81 = 1 * 9 * 9 = 1 * 3 * 3 * 3 * 3 = 1 * 3^4$$

$$99 = 1 * 3 * 33 = 1 * 3 * 3 * 11 = 1 * 3^2 * 11$$

The GCD is the least power of a number in the factors,

So, $$GCD(81,\ 99) = 1 * 3^2 * 11^0 = 9$$

Modular Arithmetic

- *m mod n*
- The mod with respect to *n is (0, 1, 2, ... n - 1).*
- Suppose m = 23 and *n = 9, then*
- 23 mod 9 = 5
- For any value of m, the value of m mod 9 is from (0, 1, 2, ... 8).

1. Addition of modular number

- The addition of two numbers p and q with same modular base n is:

$$(p \bmod n + q \bmod n) \bmod n = (p + q) \bmod n$$

2. Subtraction of modular number

- The subtraction of two numbers p and q with same modular base n is:

  (p mod n - q mod n) mod n = (p - q) mod n

3. Multiplication of modular number

- The multiplication of two numbers p and q with same modular base n is:

  (p mod n * q mod n) mod n = (p * q) mod n

e.g. p = 11, q = 15 and n = 8

[(11 mod 8) + (15 mod 8)] mod 8 = 10 mod 8 = 2, (11 + 15) mod 8 = 26 mod 8 = 2

[(11 mod 8) – (15 mod 8)] mod 8 = – 4 mod 8 = 4, (11 – 15) mod 8 = – 4 mod 8 = 4

[(11 mod 8) x (15 mod 8)] mod 8 = 21 mod 8 = 5, (11 x 15) mod 8 = 165 mod 8 = 5

Example 1: Find the value of 7^7 mod 9.

Note: *m ^a mod n = m^pq mod n*

*where a = p * q =* $(m^p \bmod n)^q \bmod n$

$$7^7 \bmod 9 = (7^2)^3 * 7 \bmod 9$$
$$= (7^2 \bmod 9)^3 \bmod 9 * 7 \bmod 9$$
$$7^2 \bmod 9 = 49 \bmod 9 = 4$$
$$7^6 \bmod 9 = (7^2)^3 \bmod 9 = 4^3 \bmod 9 = 64 \bmod 9 = 1$$
$$7^7 = 7^6 * 7 \bmod 9 = 1 * 7 \bmod 9 = 7$$

Example 2: Find the value of 5^117 mod 19.

Answer: 5^117 mod 19 = (5 * 17 * 16 * 9 * 5) mod 19

= 61200 mod 19

= 1

$$117 = (2^0 + 2^2 + 2^4 + 2^5 + 2^6)$$
$$117 = 1 + 4 + 16 + 32 + 64$$
$$5^{117} \bmod 19 = 5^{(1+4+16+32+64)} \bmod 19$$
$$5^{117} \bmod 19 = (5^1 * 5^4 * 5^{16} * 5^{32} * 5^{64}) \bmod 19$$

5^1 mod 19 = **5**

5^2 mod 19 = (**5^1 * 5^1**) mod 19 = (**5^1 mod 19** * **5^1 mod 19**) mod 19
5^2 mod 19 = (**5 * 5**) mod 19 = **25** mod 19
5^2 mod 19 = 6

5^4 mod 19 = (**5^2 * 5^2**) mod 19 = (**5^2 mod 19 * 5^2 mod 19**) mod 19
5^4 mod 19 = (**6 * 6**) mod 19 = **36** mod 19
5^4 mod 19 = 17

5^8 mod 19 = (**5^4 * 5^4**) mod 19 = (**5^4 mod 19 * 5^4 mod 19**) mod 19
5^8 mod 19 = (**17 * 17**) mod 19 = **289** mod 19
5^8 mod 19 = 4

5^16 mod 19 = (**5^8 * 5^8**) mod 19 = (**5^8 mod 19 * 5^8 mod 19**) mod 19
5^16 mod 19 = (**4 * 4**) mod 19 = **16** mod 19
5^16 mod 19 = 16

5^32 mod 19 = (**5^16 * 5^16**) mod 19 = (**5^16 mod 19 * 5^16 mod 19**) mod 19
5^32 mod 19 = (**16 * 16**) mod 19 = **256** mod 19
5^32 mod 19 = 9

5^64 mod 19 = (**5^32 * 5^32**) mod 19 = (**5^32 mod 19 * 5^32 mod 19**) mod 19
5^64 mod 19 = (**9 * 9**) mod 19 = **81** mod 19
5^64 mod 19 = 5

5^117 mod 19 = (**5^1 * 5^4 * 5^16 * 5^32 * 5^64**) mod 19

5^117 mod 19 = (**5^1 mod 19 * 5^4 mod 19 * 5^16 mod 19 * 5^32 mod 19 * 5^64 mod 19**) mod 19

5^117 mod 19 = (**5 * 17 * 16 * 9 * 5**) mod 19

5^117 mod 19 = **61200** mod 19 = **1**

5^117 mod 19 = 1

Example 3: Find the value of $3^{110} \bmod 9$.

Fermat's Little Theorem

- If p is prime and a is an integer not divisible by p, then . . .

  $a^p \equiv a \pmod{p}$

  $\mathbf{a^{p-1} \equiv 1 \pmod{p}}$

  - **Hence, a^{p-1} mod p = 1** where, p is prime and GCD (a, p) = 1

- E.g. 8^{12} *mod 13 = 1 mod 13 =1*
- 8^{103} *mod 103 = 8 mod 103 = 8*
- This theorem is useful in public key (RSA) and primality testing.

Example 3: Suppose a = 7 and p = 19 then prove Fermat's Little theorem

Example 4: Compute the value of **12345^23456789 mod 101** using Fermat's theorem

Solution By Fermat's Little theorem $n^{p-1} = 1 \bmod p$ where $n = 12345$ and $p = 101$.

$$12345^{(101-1)} \bmod 101 = 1$$

$$12345^{100} \bmod 101 = 1$$

Therefore, $12345^{23456789} \bmod 101 = (12345^{100})^{234567} * 12345^{89} \bmod 101$

$$= 1 * 12345^{89} \bmod 101$$

$$= 12345^{89} \bmod 101$$

But $12345 \bmod 101 = 23$

Therefore, $23^{89} \bmod 101$

$$23 \bmod 101 = 23$$

$$23^2 \bmod 101 = 24$$

Therefore, $23^{89} \bmod 101$

$$23 \bmod 101 = 23$$

$$23^2 \bmod 101 = 24$$

$$23^3 \bmod 101 = 47$$

$$23^4 \bmod 101 = 71$$

$$23^5 \bmod 101 = 17$$

$$23^7 \bmod 101 = 4$$

$$23^{89} \bmod 101 = (23^7)^{12}\, 23^5 \bmod 101$$

$$= 4^{12} * 17 \bmod 101$$

$$= 5 * 17 \bmod 101$$

$$= 85$$

Therefore, the value of $12345^{23456789} \bmod 101 = 85$.

Fermat's little theorem and its congruence

- Suppose a positive integer be p and two integers x and y are congruent mod p.

  Mathematically, $\mathbf{x \equiv y \pmod{p}}$ **if** $\mathbf{p \mid (x-y)}$

For example:

i) $5 \equiv 2 \bmod 3$

ii) $23 \equiv -1 \bmod 12$

Euler Totient Function ø(n)

- ø (n) = how many numbers there are between **1 and *n* - 1** that are **relatively prime to *n*.**
- ø (4) = 2 (1, 3 are relatively prime to 4)
- ø (5) = 4 (1, 2, 3, 4 are relatively prime to 5)
- ø (6) = 2 (1, 5 are relatively prime to 6)
- ø (7) = 6 (1, 2, 3, 4, 5, 6 are relatively prime to 7)
- This theorem generalizes Fermat's theorem and is an important key to the RSA algorithm.

For prime p, ø(p) = p -1 e.g. ø(37) = 36

Two prime p, q with $p \neq q$, ø(n) = ø(p.q) = (p -1) x (q -1) e.g. ø(21) = (3–1) x (7–1) = 2 x 6 = 12

Where 12 integers are [1, 2, 4, 5, 8, 10, 11, 13, 16, 17, 19, 20]

- Find ø (91) ?
- ø (91) = ø (13 * 7)

  = 12 * 6 = 72

- Euler's theorem, for every a and p that are relatively prime:

  $a^{\Phi(p)} \equiv 1 \pmod{p}$ **i.e.** $a^{\Phi(p)} \bmod p = 1$

- In other words, If *a* and *p* are relatively prime, with *a* being the smaller integer, then when we multiply *a* with itself (*p*) times and divide the result by p, the remainder will be 1.

The general formula to compute ø (n)

- *For a prime number p, the totient function is* ø(p) = p - 1 (because all the numbers less p are relatively prime to p)

  Theorem: If p *is a prime and* a *is a positive integer, then*

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

- Find ø (75) ?
- Find ø (200) ?

Euclidean Algorithm

- Suppose *p and q are two numbers.*
- *GCD (p, q) is the largest number that divides evenly both p and q.*
- *Euclidean algorithm* is used **to compute** the greatest common divisor (**GCD**) of two integer numbers.
- Euclid theorem: **GCD(p, *q) = GCD(q, p mod q)***

Example:

1. Compute GCD (997, 366) using Euclid's algorithm

2. Compute GCD (2222, 1234) using Euclid's algorithm.

1. **Compute GCD (997, 366) using Euclid's algorithm**
2. **Note:** Every time divide the divisor by remainder

- 997 = 2 * 366 + 265

  366 = 1 * 265 + 101

  265 = 2 * 101 + 63

  101 = 1 * 63 + 38

  63 = 1 * 38 + 25

  38 = 1 * 25 + 13

  25 = 1 * 13 + 12

  13 = 1 * 12 + 1

  12 = 12 * 1 + 0

GCD (997, 366) = 1

2. Compute GCD (2222, 1234) using Euclid's algorithm

- 2222 = 1 * 1234 + 988

  1234 = 1 * 998 + 246

  998 = 4 * 246 + 4

  246 = 61 * 4 + 2

  4 = 2 * 2 + 0

GCD (2222, 1234) = 2

Extended Euclidean Algorithm

- Suppose *p and q are two integer numbers. There exist two integers x and y such that*

$$xp + yq = GCD(p, q).$$

Extended Euclidean algorithm is used to find the value of *x and y.*

- Write the two linear combinations vertically as shown below and apply Euclid's algorithm to get *g = GCD(p, q) and the values of x and the y to satisfy the equation*
  - $xp + yq = g.$

  - $x = 1.x + 0.y$
  - $y = 0.x+1.y$
  - $r = 1.x + (-z) .y$

- Find integers *p and q such that 51p* + *36q = 3. Also find the GCD* (51, 36)

$51 = 36(1) + 15$ $36 = 15(2) + 6$ $15 = 6(2) + 3(GCD)$ $6 = 3(2) + 0$	$15 = 51 - 36(1)$ $6 = 36 - 15(2)$ $3 = 15 - 6(2)$

- $3 = 15 - 6(2)$
- $3 = 15 - [36 - 15(2)](2)$
- $3 = 15(5) - 36(2)$
- $3 = [51 - 36(1)](5) - 36(2)$
- $3 = 51(5) - 36(5) - 36(2)$
- $3 = 51(5) - 36(7)$
- $3 = 51(5) + 36(-7)$
- Therefore, the values of *p = 5 and q = -7 and GCD = 3.*

Examples:

1. Use the extended Euclidean algorithm to find the multiplicative inverse of 77 mod 5.

2. Use the extended Euclidean algorithm to find the multiplicative inverse of 35 mod 11.

Chinese Remainder Theorem (CRT)

Chinese Remainder Theorem: If $m_1, m_2, .., m_k$ are pairwise relatively prime positive integers, and if $a_1, a_2, .., a_k$ are any integers, then the simultaneous congruences,

$x \equiv a_1 \pmod{m_1}, x \equiv a_2 \pmod{m_2}, ..., x \equiv a_k \pmod{m_k}$ have a solution, and the solution is unique modulo M, where $M = m_1 m_2 \cdots m_k$

Solution To Chinese Remainder Theorem:

1. Find $M = m_1 \times m_2 \times \ldots \times m_k$. This is the common modulus.
2. Find $M_1 = M/m_1, M_2 = M/m_2, \ldots, M_k = M/m_k$ (Formula: $\mathbf{M_i = M/m_i}$)
3. Find the multiplicative inverse of $M_1, M_2, \ldots, M_k$ using the corresponding moduli $(m_1, m_2, \ldots, m_k)$. Call the inverses $M_1^{-1}, M_2^{-1}, \ldots, M_k^{-1}$ (**Formula:** $\mathbf{M_i\, M_i^{-1} = 1 \bmod (m_i)}$)
4. The solution to the simultaneous equations is

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

Note that the set of equations can have a solution even if the moduli are not relatively prime but meet other condition.

Example: Solve the simultaneous congruences $x \equiv 1 \pmod 5$, $x \equiv 1 \pmod 7$, $x \equiv 3 \pmod{11}$.

Chinese Remainder Theorem :

It will determine a no. that will divided by some given divisor leaves given remainder

$x \equiv a_1 \pmod{m_1}$

$x \equiv a_2 \pmod{m_2}$

$x \equiv a_3 \pmod{m_3}$

$\Rightarrow$

$x \equiv 1 \pmod{5}$

$x \equiv 1 \pmod{7}$

$x \equiv 3 \pmod{11}$

$a_1 = 1 \qquad m_1 = 5$

$a_2 = 1 \qquad m_2 = 7$

$a_3 = 3 \qquad m_3 = 11$

$\Rightarrow \quad x = (M_1 X_1 a_1 + M_2 X_2 a_2 + M_3 X_3 a_3) \pmod{M}$

$\Rightarrow \quad M = m_1 \cdot m_2 \cdot m_3$

$= 5 \cdot 7 \cdot 11$

$= 385$

$M_i = \dfrac{M}{m_i}$ (Calculation of M_1, M_2, M_3)

$\therefore M_1 = \dfrac{M}{m_1} = \dfrac{385}{5} = 77$

$M_2 = \dfrac{M}{m_2} = \dfrac{385}{7} = 55$

$M_3 = \dfrac{M}{m_3} = \dfrac{385}{11} = 35$

ASSIGNMENT / TEST Page No.

Calculation of x_1, x_2, x_3 $\longrightarrow$ $M_i X_i \equiv 1 \pmod{m_i}$

$M_1 X_1 \equiv 1 \pmod{m_1}$

$77 X_1 \equiv 1 \pmod 5$

$2 X_1 \equiv 1 \pmod 5$

$3\,(\,2X_1 \equiv 1 \pmod 5)$

$6 X_1 \equiv 3 \pmod 5$

$1 X_1 \equiv 3 \pmod 5$

$\therefore \boxed{X_1 = 3}$

$(77 X_1 \bmod 5) = 1$

$(77/5$ - remainder $= 2$

5	6 ←	2×3
10	11	
15	16	
20	21	
⋮		

$M_2 X_2 \equiv 1 \pmod{m_2}$

$55 X_2 \equiv 1 \pmod 7$

$6 X_2 \equiv 1 \pmod 7$

$6\,(\,6 X_2 \equiv 1 \pmod 7)$

$36 X_2 \equiv 6 \pmod 7$

$1 X_2 \equiv 6 \pmod 7$

$\therefore \boxed{X_2 = 6}$

7	8
14	15
21	22
28	29
35	36

$$M_3 X_3 \equiv 1 \pmod{m_3}$$

$$35 X_3 \equiv 1 \pmod{11}$$

$$2 X_3 \equiv 1 \pmod{11}$$

$$6(2 X_3 \equiv 1 \pmod{11})$$

$$12 X_3 \equiv 6 \pmod{11}$$

$$1 X_3 = 6 \pmod{11}$$

$$\therefore \boxed{X_3 = 6}$$

$$x = (M_1 X_1 a_1 + M_2 X_2 a_2 + M_3 X_3 a_3) \pmod{385}$$

$$= (77 \times 3 \times 1 + 55 \times 6 \times 1 + 35 \times 6 \times 3) \pmod{385}$$

$$\therefore x = 1191 \pmod{385}$$

$$\therefore x = 36$$

∴ Original example will be.

$36 \bmod 5 = 1$

$36 \bmod 7 = 1$

$36 \bmod 11 = 3$

Example 2: Solve the simultaneous congruences $x \equiv 6 \pmod{11}$, $x \equiv 13 \pmod{16}$, $x \equiv 9 \pmod{21}$, $x \equiv 19 \pmod{25}$.

Ans: 89469

Example 3: Find the smallest multiple of 10 which has remainder 1 when divide by 3, remainder 6 when divided by 7 and remainder 6 when divided by 11.

Discrete Logarithms

- $2^5 \bmod 3 = 2$
- $4^4 \bmod 11 = ?$
- $8 = 5^i \bmod 13$, Determine i ?
- Check, $? = 5^7 \bmod 13$, and $? = 5^{11} \bmod 13$
- The inverse problem to exponentiation is to find the **discrete logarithm** of a number modulo m. i.e. to find i such that $\mathbf{a \equiv b^i \ (mod\ m)}$ where, $0 \le i \le (m - 1)$
- This is written as $\mathbf{i = dlog_b\ a\ (mod\ m)}$
- if **b is a primitive root of m** then it always exists, otherwise it may not.
- used in Diffe-Hellman and the digital signature algorithm.

- **Primitive Root:** If b is a primitive root of m where m is a prime number then b^1 mod m, b^2 mod m , b^3 mod m , b^{m-1} mod m are distinct values.

  - Check 3 is primitive root of 5?
  - Check 4 is primitive root of 5?
  - Check 3 is primitive root of 13?

$a \equiv b^i \pmod{m}$ → **We have to select value of b so that we will get different value of i**

$b \equiv a^i \pmod{m}$

Put a = 1, 2, 3,…12

Power of integers, Modulo 13

a	a^2	a^3	a^4	a^5	a^6	a^7	a^8	a^9	a^{10}	a^{11}	a^{12}	$\lvert a \rvert_{13}$
1	1	1	1	1	1	1	1	1	1	1	1	1
2	4	8	3	6	12	11	9	5	10	7	1	12
3	9	1	3	9	1	3	9	1	3	9	1	3
4	3	12	9	10	1	4	3	12	9	10	1	6
5	11	8	1	5	11	8	1	5	11	8	1	4
6	10	8	9	2	12	7	3	5	4	11	1	12
7	10	5	9	11	12	6	3	8	4	2	1	12
8	11	5	1	8	11	5	1	8	11	5	1	4
9	3	1	9	3	1	9	3	1	9	3	1	3
10	9	12	3	4	1	10	9	12	3	4	1	6
11	4	5	3	7	12	2	9	8	10	4	1	12
12	1	12	1	12	1	12	1	12	1	12	1	2

Powers of Integers, Modulo 19

a	a^2	a^3	a^4	a^5	a^6	a^7	a^8	a^9	a^{10}	a^{11}	a^{12}	a^{13}	a^{14}	a^{15}	a^{16}	a^{17}	a^{18}
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
2	4	8	16	13	7	14	9	18	17	15	11	3	6	12	5	10	1
3	9	8	5	15	7	2	6	18	16	10	11	14	4	12	17	13	1
4	16	7	9	17	11	6	5	1	4	16	7	9	17	11	6	5	1
5	6	11	17	9	7	16	4	1	5	6	11	17	9	7	16	4	1
6	17	7	4	5	11	9	16	1	6	17	7	4	5	11	9	16	1
7	11	1	7	11	1	7	11	1	7	11	1	7	11	1	7	11	1
8	7	18	11	12	1	8	7	18	11	12	1	8	7	18	11	12	1
9	5	7	6	16	11	4	17	1	9	5	7	6	16	11	4	17	1
10	5	12	6	3	11	15	17	18	9	14	7	13	16	8	4	2	1
11	7	1	11	7	1	11	7	1	11	7	1	11	7	1	11	7	1
12	11	18	7	8	1	12	11	18	7	8	1	12	11	18	7	8	1
13	17	12	4	14	11	10	16	18	6	2	7	15	5	8	9	3	1
14	6	8	17	10	7	3	4	18	5	13	11	2	9	12	16	15	1
15	16	12	9	2	11	13	5	18	4	3	7	10	17	8	6	14	1
16	9	11	5	4	7	17	6	1	16	9	11	5	4	7	17	6	1
17	4	11	16	6	7	5	9	1	17	4	11	16	6	7	5	9	1
18	1	18	1	18	1	18	1	18	1	18	1	18	1	18	1	18	1

MIT-WPU
॥ विश्वशान्तिर्ध्रुवं ध्रुवा ॥

Discrete Logarithms mod 19

$a \equiv b^i \pmod{m}$

(a) Discrete logarithms to the base 2, modulo 19

a	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
i $\log_{2,19}(a)$	18	1	13	2	16	14	6	3	8	17	12	15	5	7	11	4	10	9

(b) Discrete logarithms to the base 3, modulo 19

a	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
$\log_{3,19}(a)$	18	7	1	14	4	8	6	3	2	11	12	15	17	13	5	10	16	9

(c) Discrete logarithms to the base 10, modulo 19

a	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
$\log_{10,19}(a)$	18	17	5	16	2	4	12	15	10	1	6	3	13	11	7	14	8	9

(d) Discrete logarithms to the base 13, modulo 19

a	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
$\log_{13,19}(a)$	18	11	17	4	14	10	12	15	16	7	6	3	1	5	13	8	2	9

(e) Discrete logarithms to the base 14, modulo 19

a	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
$\log_{14,19}(a)$	18	13	7	8	10	2	6	3	14	5	12	15	11	1	17	16	4	9

(f) Discrete logarithms to the base 15, modulo 19

a	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
$\log_{15,19}(a)$	18	5	11	10	8	16	12	15	4	13	6	3	7	17	1	2	14	9

Primality Test: Miller-Rabin Algorithm

- If we can efficiently test the primality of a number, then we can generate primes fast.
- Deterministic Test and Probabilistic Test
- RSA algorithm based on primality test

Miller-Rabin Algorithm

- Miller-Rabin-Test (n, a) // n is the number; a is the base
- Find m and k such that: $n - 1 = m \times 2^k$

  If $k \leq 1$, Calculate $T \leftarrow a^m \bmod n$

  If ($T = \pm 1$) return "a prime number", otherwise composite number
- If $k > 1$, Calculate $T \leftarrow T^2 \bmod n$

  If ($T = +1$) return " number is composite",

  If ($T = -1$) return " number is prime",

  else, composite number

Examples:

1. Check (27, 2) is prime or not using Miller Rabin algorithm.

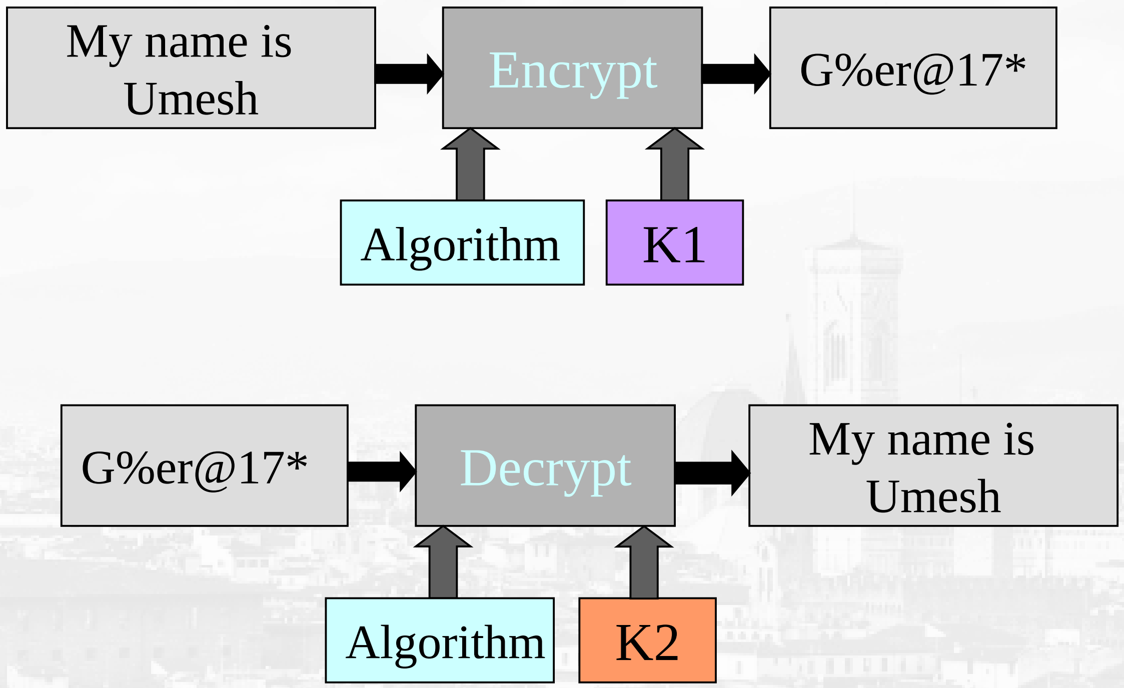
2. Check 29 is prime or not using Miller Rabin algorithm.

3. Check 221 is prime or not using Miller Rabin algorithm.

4. Apply Miller-Rabin Algorithm using base 2 to test whether the number 341 is composite or not.

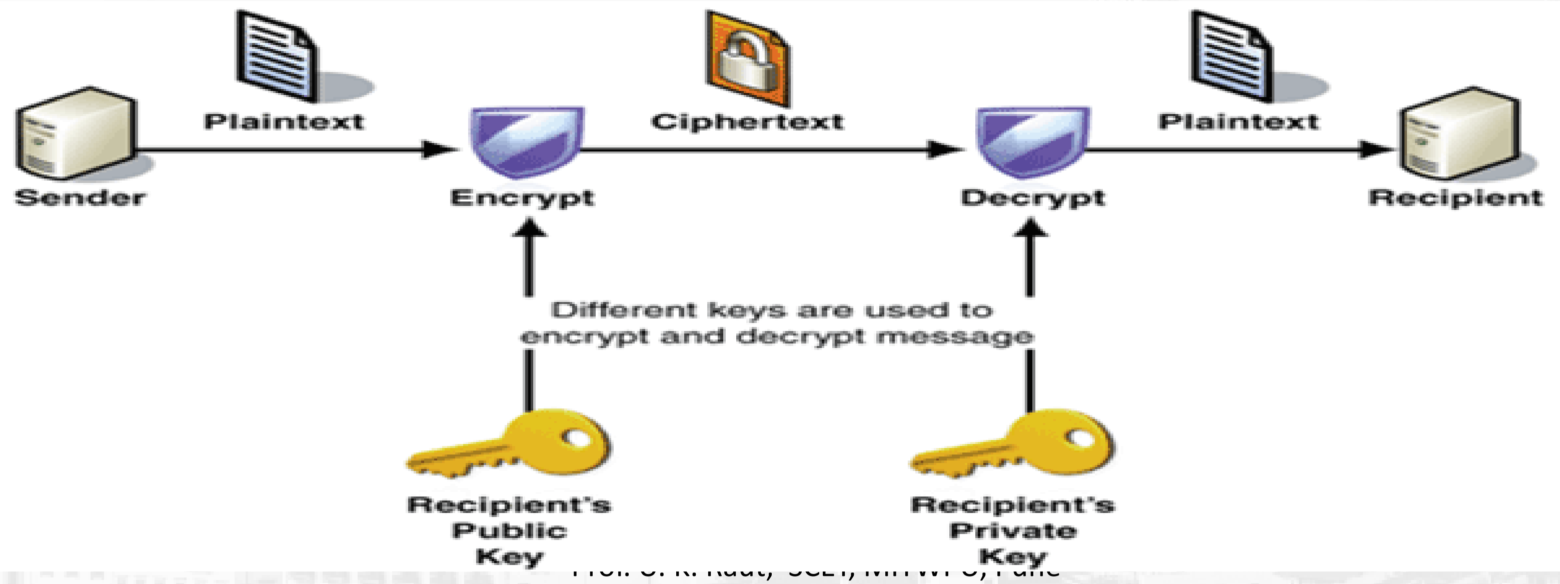
Public Key Cryptography

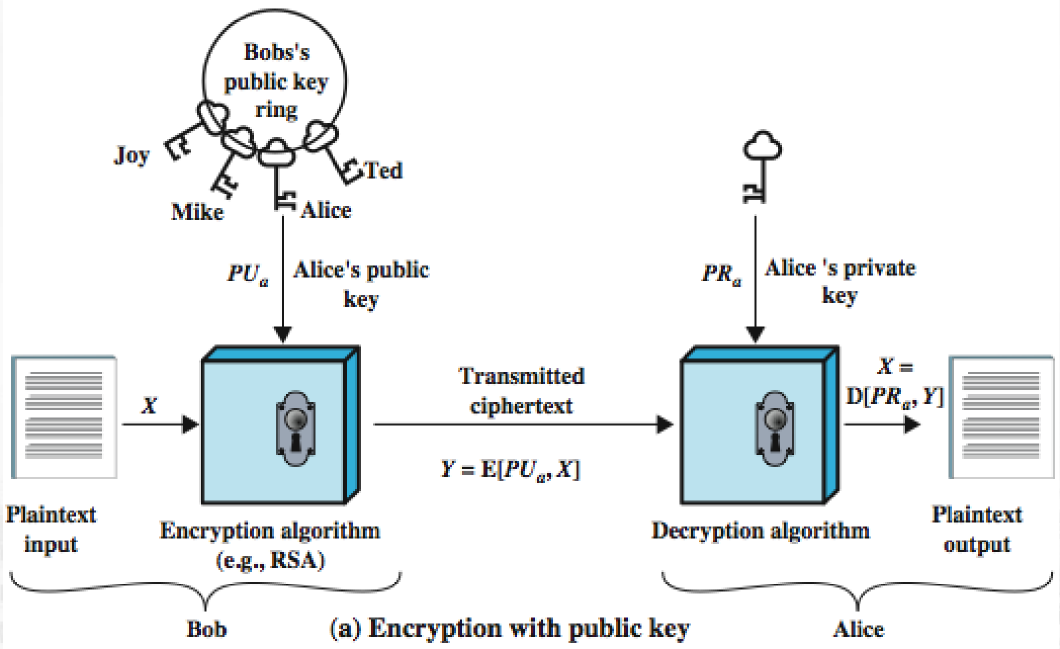
Asymmetric Key Encryption: Example



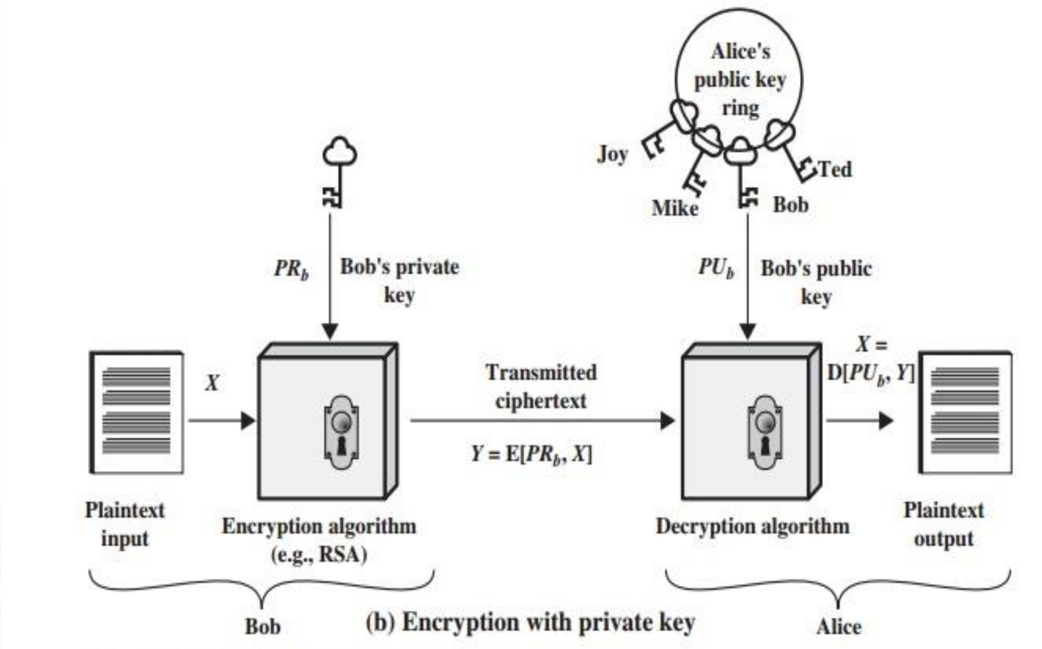
Matrix of Keys

Key details	***A*** **should know**	***B*** **should know**
A’s private key	Yes	No
A’s public key	Yes	Yes
B’s private key	No	Yes
B’s public key	Yes	Yes





(a) Encryption with public key



(b) Encryption with private key

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

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

RSA En/decryption

- by **R**on Rivest, Adi **S**hamir and Leonard **A**dleman of MIT in 1977
- best known & widely used public-key scheme
- based on property of modular exponentiation
- key uses large integers (eg. 1024 bits)

RSA Key Setup

- each user generates a public/private key pair by: selecting two large primes at random: **p, q**
- computing their system modulus **n = (p * q)**
- Compute: **ø(n) = (p -1)(q -1)**
- selecting at random the **encryption key** (public) e, where 1< e < ø(n), gcd(e,ø(n)) = 1
- solve following equation to find decryption key d

  d * e = 1 [mod ø(n)] and $0 \le d \le n$
- publish their public encryption key: **PU = {e, n}**
- keep secret private decryption key: **PR = {d, n}**

- to encrypt a message M the sender:
  - obtains **public key** of recipient PU = {e, n}
  - computes Ciphertext : $\mathbf{C = M^e \bmod n}$, where $0 \le M < n$
- to decrypt the ciphertext C the owner:
  - uses their private key PR = {d, n}
  - computes: $\mathbf{M = C^d \bmod n}$
- note that the message M must be smaller than the modulus n (block if needed)

Key Generation

Select p, q	p and q both prime, $p \neq q$
Calculate $n = p \times q$	
Calcuate $\phi(n) = (p-1)(q-1)$	
Select integer e	$\gcd(\phi(n), e) = 1; 1 < e < \phi(n)$
Calculate d	$d \equiv e^{-1} \pmod{\phi(n)}$
Public key	$PU = \{e, n\}$
Private key	$PR = \{d, n\}$

Encryption

Plaintext:	$M < n$
Ciphertext:	$C = M^e \bmod n$

Decryption

Ciphertext:	C
Plaintext:	$M = C^d \bmod n$

Symmetric Encryption	Asymmetric Encryption
Well-known as secret key encryption	Well-known as public key encryption
Uses a single key for both encryption and decryption	Uses a different key for encryption and decryption
Symmetric encryption is fast in execution	Asymmetric Encryption is slow in execution due to the high computational burden
Size of resulting encrypted text usually same or less than original	Size of resulting encrypted text more than original
Problem of Key Exchange	No Problem of Key Exchange
Used for encrypting small or large message	Used for encrypting small message because its computational time is more
Exemple: DES, 3DES, AES, and RC4	Exemple: Diffie-Hellman, RSA

Symmetric vs Public-Key

Conventional Encryption	Public-Key Encryption
Needed to Work: 1. The same algorithm with the same key is used for encryption and decryption. 2. The sender and receiver must share the algorithm and the key. *Needed for Security:* 1. The key must be kept secret. 2. It must be impossible or at least impractical to decipher a message if no other information is available. 3. Knowledge of the algorithm plus samples of ciphertext must be insufficient to determine the key.	*Needed to Work:* 1. One algorithm is used for encryption and decryption with a pair of keys, one for encryption and one for decryption. 2. The sender and receiver must each have one of the matched pair of keys (not the same one). *Needed for Security:* 1. One of the two keys must be kept secret. 2. It must be impossible or at least impractical to decipher a message if no other information is available. 3. Knowledge of the algorithm plus one of the keys plus samples of ciphertext must be insufficient to determine the other key.

Advantages of RSA

- Can be used for both encryption as well as for digital signature.
- Trapdoor in RSA is in knowing value of n but not knowing the primes of that are factors of n

Disadvantages of RSA

- To protect the encryption, the minimum number of bits in n should be of 2048 bits.

RSA Example - Key Setup

1. Select primes: $p = 17$ & $q = 11$
2. Calculate $n = pq = 17 \times 11 = 187$
3. Calculate $ø(n) = (p - 1)(q - 1) = 16 \times 10 = 160$
4. Select e: gcd(e,160) = 1; choose $e = 7$
5. Determine d: $de = 1 \bmod 160$ and $d < 160$

   Value is d = 23 since 23 x 7 =161 = 10 x 160 + 1
6. Publish public key PU = {7,187}
7. Keep secret private key PR = {23,187}

RSA Example - En/Decryption

- sample RSA encryption/decryption is:

8. given message M = 88 (nb. 88 < 187)

9. encryption:

   $C = 88^7 \bmod 187 = 11$

10. decryption:

   $M = 11^{23} \bmod 187 = 88$

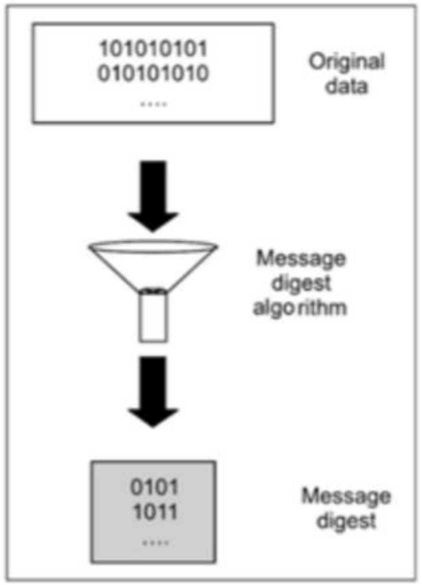
Example 1: The parameters given are $p = 5$, $q = 17$. Find out the possible public keys and private key for RSA algorithm. Also encrypt the message "4".

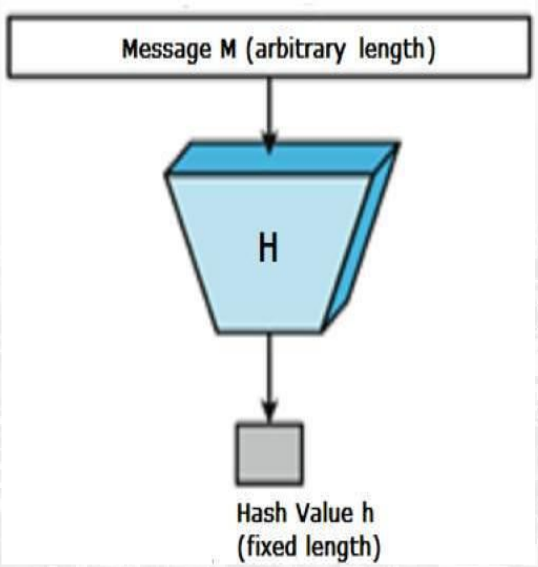
Example 2: Using RSA algorithm to encrypt the message $m =$ "6" use parameters $p = 3$, $q = 17$, $e = 7$, calculate decryption key.

Message Digest: MD 5 and SHA -1

- The digest is sometimes called the "hash" or "fingerprint" of the input.
- Hash value is used to check the integrity of the message.
- MD5 processes a variable-length message into a fixed-length output of 128 bits.

Simple example: 7391743





Algorithm:

- Step -1: Padding
- Step - 2: Append length
- Step - 3: Divide the input into 512-bit blocks
- Step - 4: Initialize chaining variables (4 variables)
- Step - 5: Process blocks

Step 1: Padding

- To make the length of the original message equal to a value, which is 64 bits less than an exact multiple of 512
- **Note:** Padding is always added, even if the original message is already 64 bits less than a multiple of 512

Original Message + Padding (1-512 bits)

⇩

Original Message	**Padding**

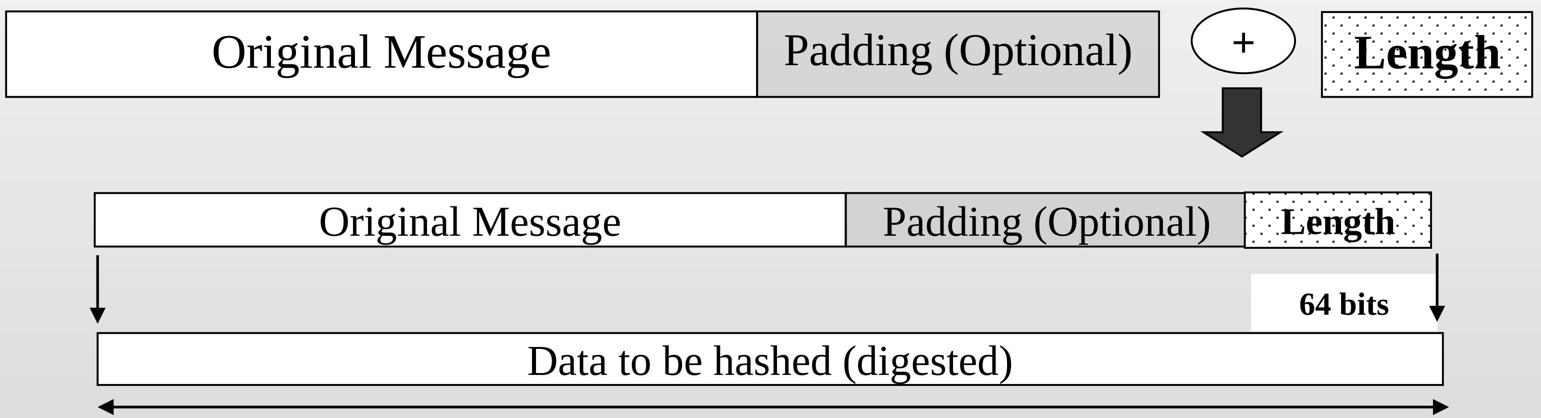
The total length of this should be 64 bits less than a multiple of 512.

For example, it can be 448 bits (448 = 512 – 64) or 960 bits

(960 = [2 x 512] – 64) or 1472 (1472 = [3 x 512] – 64), etc.

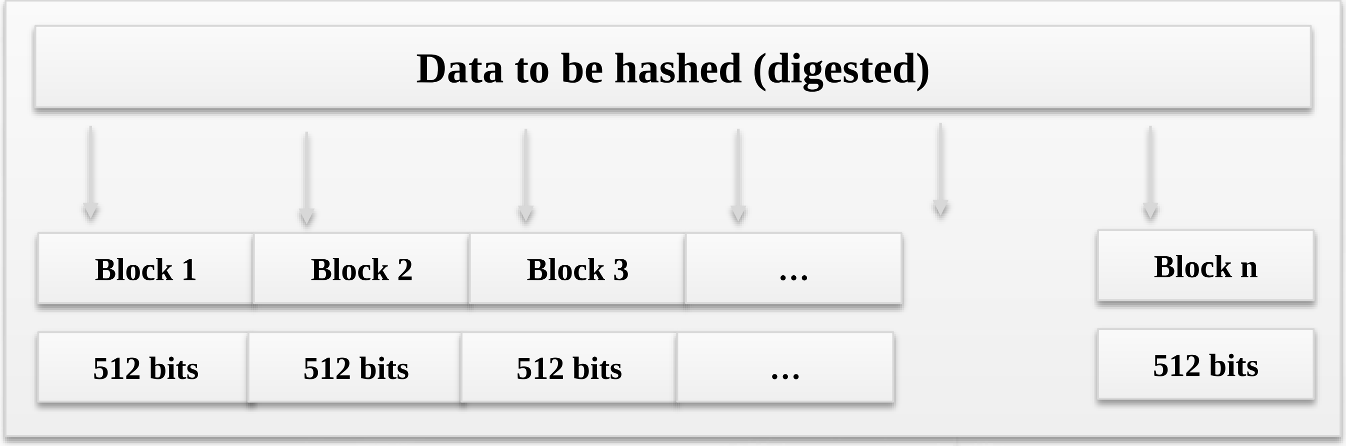
Step 2: Append length

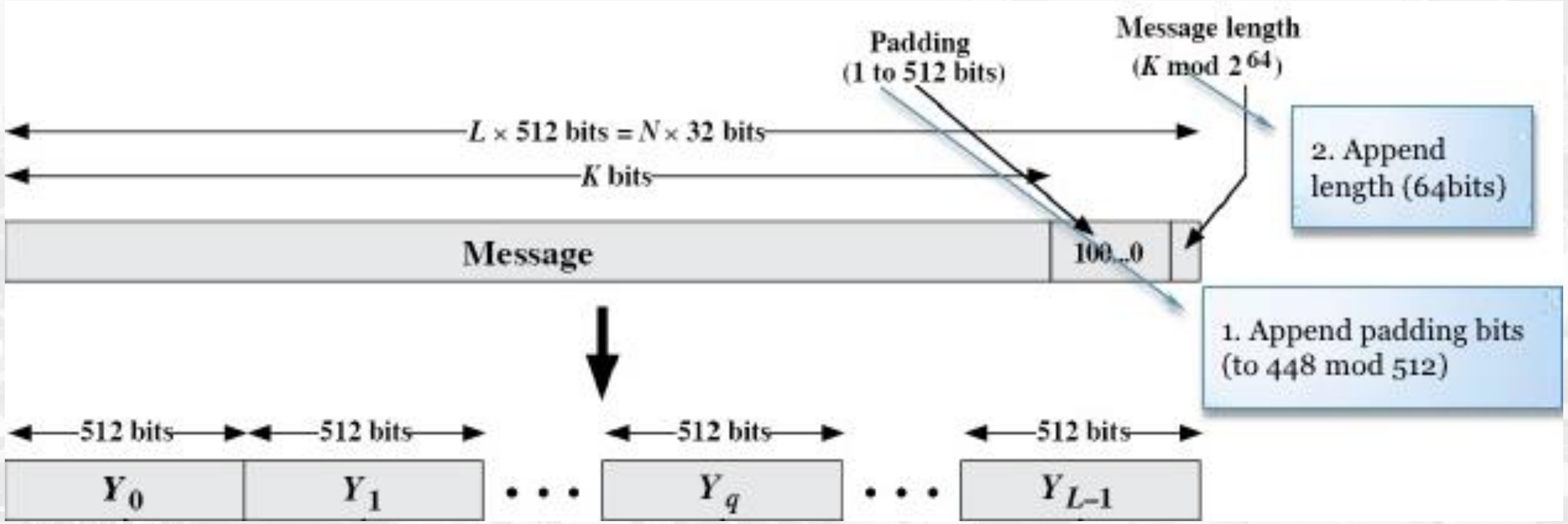
- Add a 64-bit binary-string which is the representation of the message's length
- If the original length is greater than 2^{64}, then only **the low-order 64** bits of the length are used.
- Thus, field contains the length of the original message, modulo 2^{64}.



The length of the original message (excluding the padding) is calculated, and is appended to the end of the message block + padding. This length is expressed in 64 bits. If the length of the message exceeds 64 bits (i.e. it is greater than 2^{64}, then only the last 64 bits of the length are used, i.e. a modulus 64 of the length is taken. After the 64-bit length of the message is appended, this becomes the final message (i.e. the message to be hashed).
The length of the message is now a multiple of 512 bits.

Step 3: Divide the input into 512-bit blocks





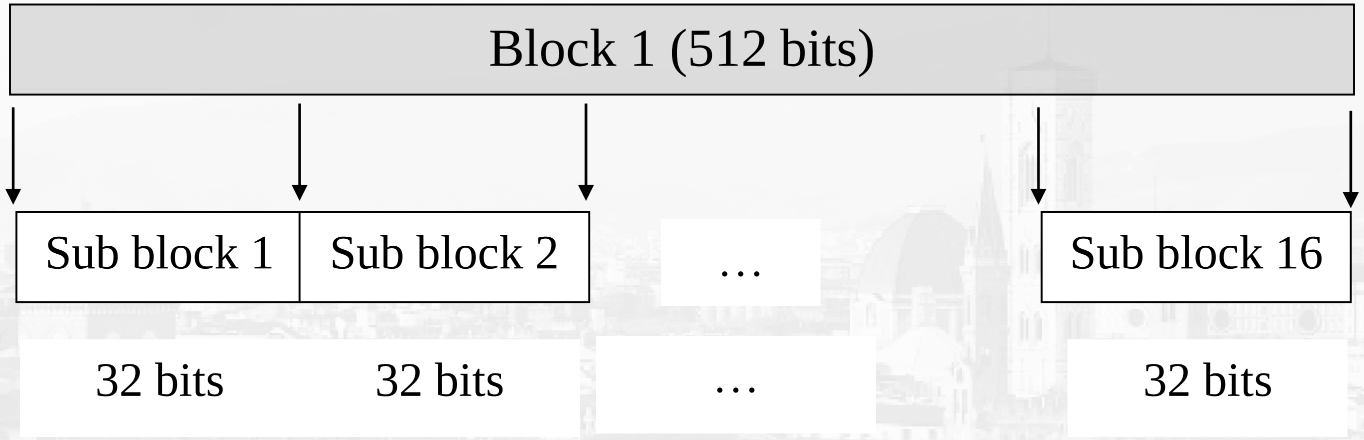
Step 4: Initialize MD buffer

- A four-word buffer (A, B, C, D) is used to compute the message digest.
- Here each of A, B, C, D is a 32 bit register.

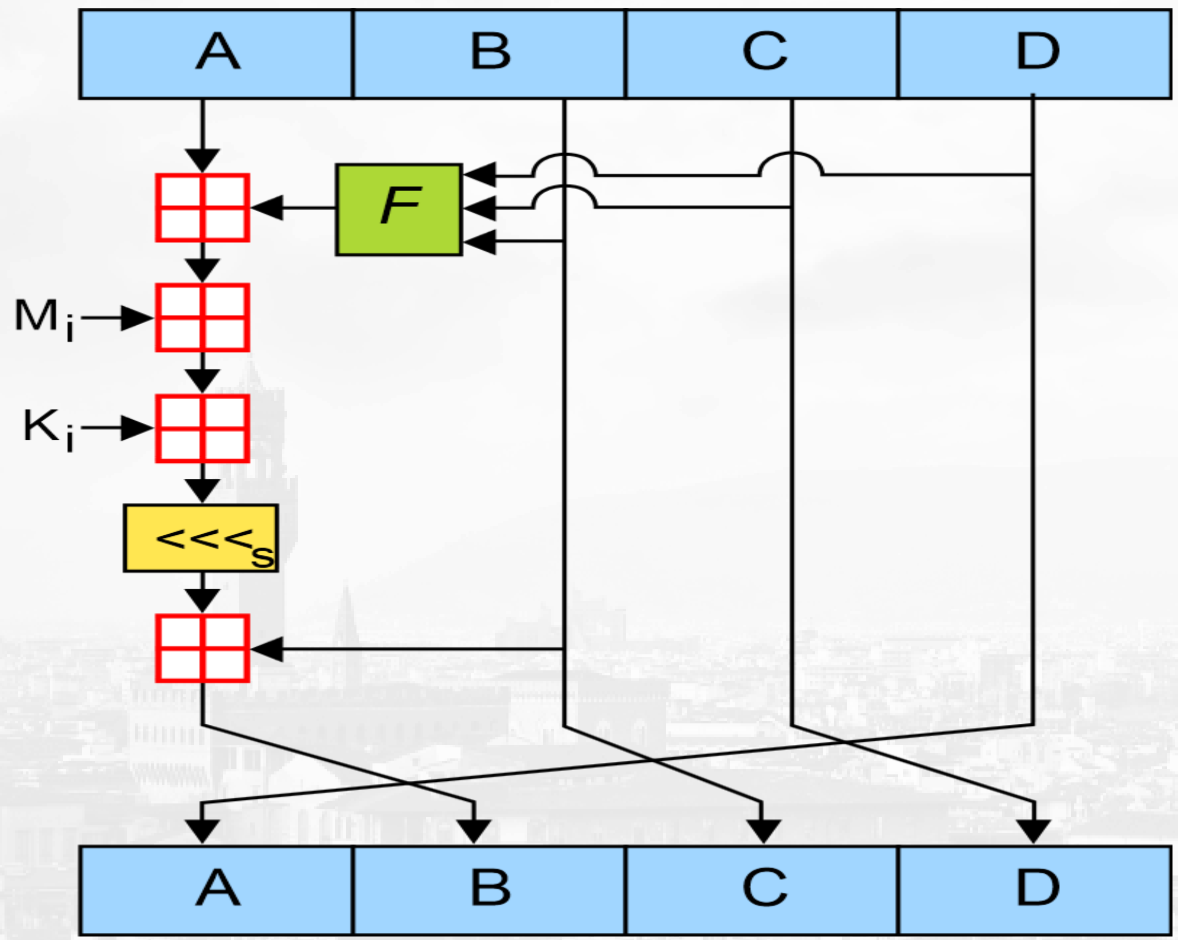
A	01	23	45	67
B	89	AB	CD	EF
C	FE	DC	BA	98
D	76	54	32	10

Step 5: Process Blocks (or message)

- Divide the 512- bit block into 16 sub-blocks.
- Each sub-block undergoes 4 rounds of operations. Total 64 operations are performed.



$$A = B + ((A + \text{Process } F(B, C, D) + M_i + K_i) <<< s)$$



- There are four possible functions F; a different one is used in each round:

Round	**Process F**
1	(B AND C) OR ((NOT B) AND (D))
2	(B AND D) OR (C AND (NOT D))
3	B XOR C XOR D
4	C XOR (B OR (NOT D))

Constants of MD5

T_1 = d76aa478	T_{17} = f61e2562	T_{33} = fffa3942	T_{49} = f4292244
T_2 = e8c7b756	T_{18} = c040b340	T_{34} = 8771f681	T_{50} = 432aff97
T_3 = 242070db	T_{19} = 265e5a51	T_{35} = 6d9d6122	T_{51} = ab9423a7
T_4 = c1bdceee	T_{20} = e9b6c7aa	T_{36} = fde5380c	T_{52} = fc93a039
T_5 = f57c0faf	T_{21} = d62f105d	T_{37} = a4beea44	T_{53} = 655b59c3
T_6 = 4787c62a	T_{22} = 02441453	T_{38} = 4bdecfa9	T_{54} = 8f0ccc92
T_7 = a8304613	T_{23} = d8a1e681	T_{39} = f6bb4b60	T_{55} = ffeff47d
T_8 = fd469501	T_{24} = e7d3fbc8	T_{40} = bebfbc70	T_{56} = 85845dd1
T_9 = 698098d8	T_{25} = 21e1cde6	T_{41} = 289b7ec6	T_{57} = 6fa87e4f
T_{10} = 8b44f7af	T_{26} = c33707d6	T_{42} = eaa127fa	T_{58} = fe2ce6e0
T_{11} = ffff5bb1	T_{27} = f4d50d87	T_{43} = d4ef3085	T_{59} = a3014314
T_{12} = 895cd7be	T_{28} = 455a14ed	T_{44} = 04881d05	T_{60} = 4e0811a1
T_{13} = 6b901122	T_{29} = a9e3e905	T_{45} = d9d4d039	T_{61} = f7537e82
T_{14} = fd987193	T_{30} = fcefa3f8	T_{46} = e6db99e5	T_{62} = bd3af235
T_{15} = a679438e	T_{31} = 676f02d9	T_{47} = 1fa27cf8	T_{63} = 2ad7d2bb
T_{16} = 49b40821	T_{32} = 8d2a4c8a	T_{48} = c4ac5665	T_{64} = eb86d391

$T_i = \lfloor 2^{32} |\sin i| \rfloor$

20

ROUND 1			ROUND 2			ROUND 3			ROUND 4		
i	k s	T[i]	i	k s	T[i]	i	k s	T[i]	i	k s	T[i]
0	0 7	d76aa478	16	1 5	f61e2562	32	5 4	fffa3942	48	0 6	f4292244
1	1 12	e8c7b756	17	6 9	c040b340	33	8 11	8771f681	49	7 10	432aff97
2	2 17	242070db	18	11 14	265e5a51	34	11 16	6d9d6122	50	14 15	ab9423
3	3 22	c1bdceee	19	0 20	e9b6c7aa	35	14 23	fde5380c	51	5 21	fc93a039
4	4 7	f57c0faf	20	5 5	d62f105d	36	1 4	a4beea44	52	12 5	655b59c3
5	5 12	4787c62a	21	10 9	02441453	37	4 11	4bdecfa9	53	3 10	8f0ccc92
6	6 17	a8304613	22	15 14	d8a1e681	38	7 16	f6bb4b60	54	10 15	ffeff47d
7	7 22	fd469501	23	4 20	e7d3fbc8	39	10 23	bebfbc70	55	1 21	85845dd1
8	8 7	698098d8	24	9 5	21e1cde6	40	13 4	289b7ec6	56	8 6	6fa87e4f
9	9 12	8b44f7af	25	14 9	c33707d6	41	0 11	eaa127fa	57	15 10	fe2ce6e0
10	10 17	ffff5bb1	26	3 14	f4d50d87	42	3 16	d4ef3085	58	6 15	a3014314
11	11 22	895cd7be	27	8 20	455a14ed	43	6 23	04881d05	59	13 21	4e0811a1
12	12 7	6b901122	28	13 5	a9e3e905	44	9 4	d9d4d039	60	4 6	f7537e82
13	13 12	fd987193	29	2 9	fcefa3f8	45	12 11	e6db99e5	61	11 10	bd3af235
14	14 17	a679438e	30	7 14	676f02d9	46	15 16	1fa27cf8	62	2 15	2ad7d2bb
15	15 22	49b40821	31	12 20	8d2a4c8a	47	2 23	c4ac5665	63	9 21	eb86d391

Types of Attack on Hashes

- **Preimage:** An attacker has an output and finds an input that hashes to that output
- **2nd Preimage:** An attacker has an output and an input x and finds a 2nd input that produces the same output as x
- **Collision:** An attacker finds two inputs that hash to the same output
- **Length Extension:** An attacker, knowing the length of message M and a digest of M signed by a sender can extend M with an additional message N and can compute the digest of M || N even without the key used to sign the digest of M

Secure Hash Algorithm (SHA)

- SHA is a modified version of MD5. (Published in 1993)
- SHA works any input message less than 2^{64} bits and produces a hash value of 160 bits.
- SHA is designed to be computationally infeasible to:
  - Obtain the original message
  - Find two message producing the same MD.

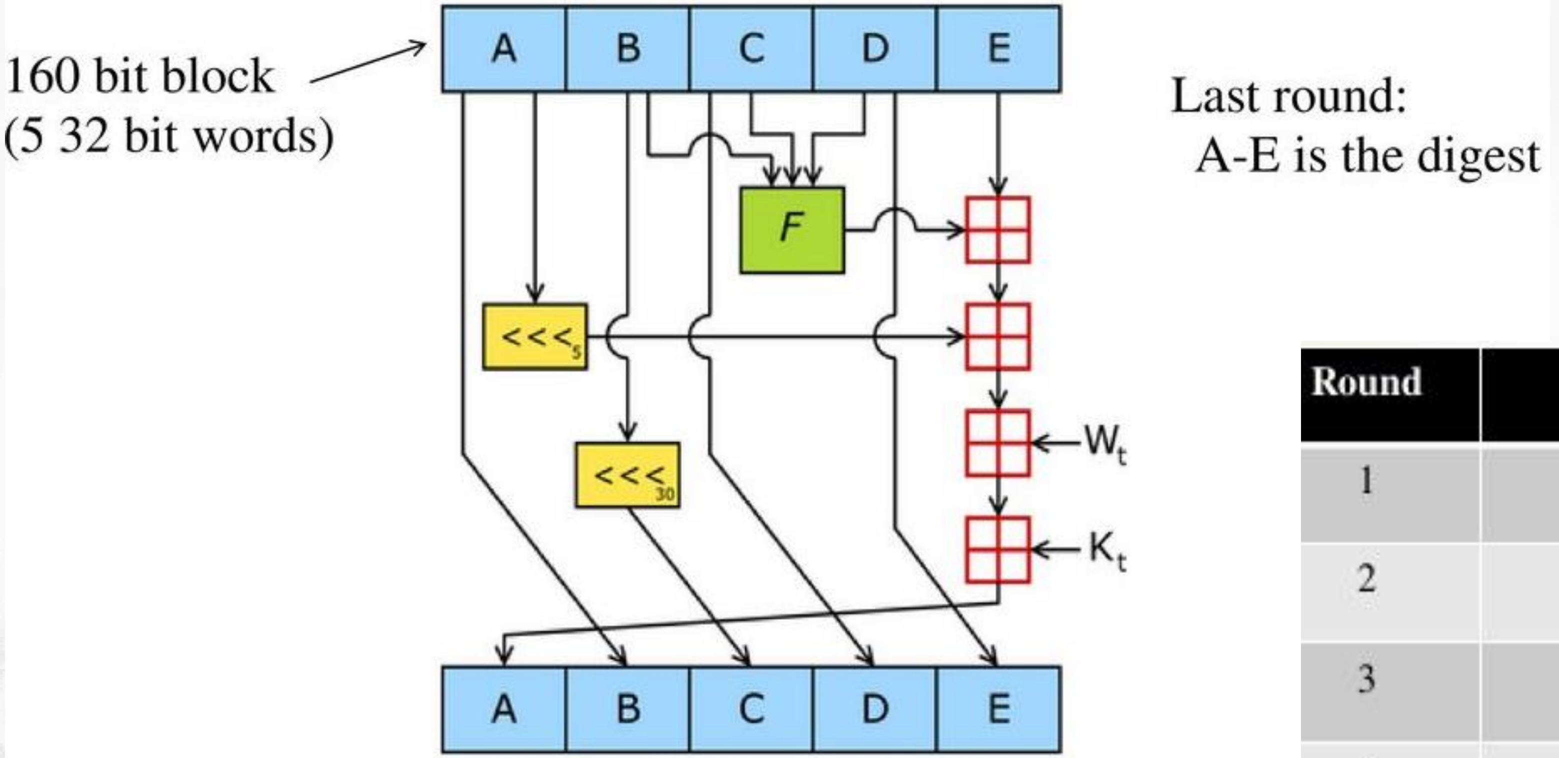
	SHA-1	SHA-256	SHA-384	SHA-512
Message digest size	160	256	384	512
Message size	$<2^{64}$	$<2^{64}$	$<2^{128}$	$<2^{128}$
Block size	512	512	1024	1024
Word size	32	32	64	64
Number of steps	80	64	80	80

Algorithm:

- ❖ Step -1: Padding
- ❖ Step - 2: Append length
- ❖ Step - 3: Divide the input into 512-bit blocks.
- ❖ Step - 4: Initialize chaining variables (5 variables)
- ❖ Step - 5: Process blocks

A	01	23	45	67
B	89	AB	CD	EF
C	FE	DC	BA	98
D	76	54	32	10
E	C3	D2	E1	F0

Process each block with A, B, C, D, E



temp = (A $<<<_5$) + F + E + K_t + w_t
E = D
D = C
C = B $<<<_{30}$
B = A
A = temp

Process F or P
M_i or W_t
K_i or W_t

Round	Process P
1	(b AND c) OR ((NOT b) AND (d))
2	b XOR c XOR d
3	(b AND c) OR (b AND d) OR (c AND d)
4	b XOR c XOR d

Comparison of MD5 and SHA

Point of discussion	MD5	SHA-1
Message digest length in bits	128	160
Attack to try and find the original message given a message digest	Requires 2^{128} operations to break in	Requires 2^{160} operations to break in, therefore more secure
Attack to try and find two messages producing the same message digest	Requires 2^{64} operations to break in	Requires 2^{80} operations to break in
Successful attacks so far	There have been reported attempts to some extent	No such claims so far
Speed	Faster (64 iterations, and 128-bit buffer)	Slower (80 iterations, and 160-bit buffer)
Software implementation	Simple, does not need any large programs or complex tables	Simple, does not need any large programs or complex tables

Thank You !!!!!!

Examples

1. Calculate (36^{106} mod 107) mod 37.

2. If n = 77, find Φ(n).

3. Use the extended Euclidean algorithm to find multiplicative inverse of 77 mod 5.

4. What is the value of d if p =3, q = 11 and e = 7. Use RSA algorithm.

5. How many primitive roots the number 15 has? Calculate all possible primitive roots for 15.