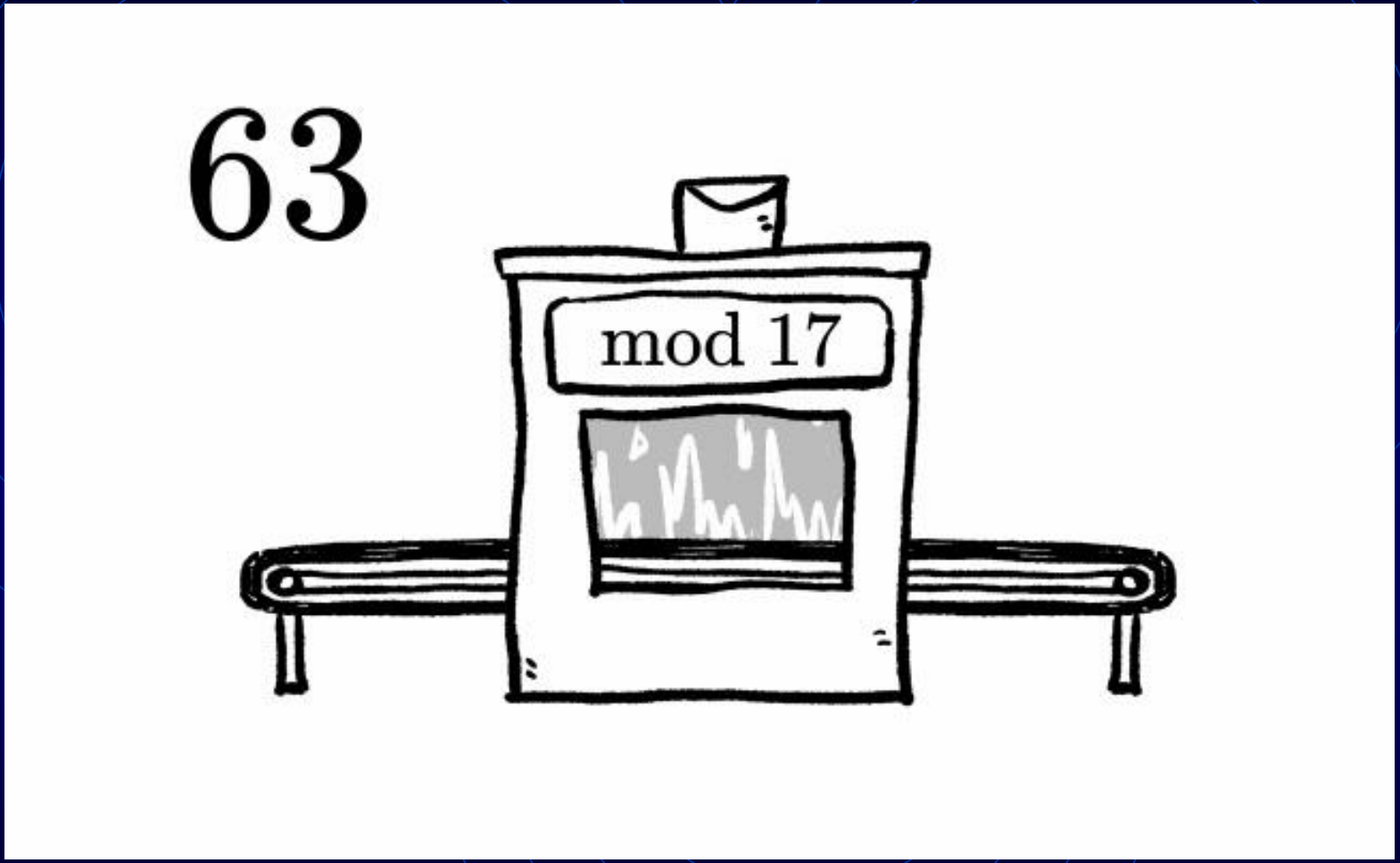
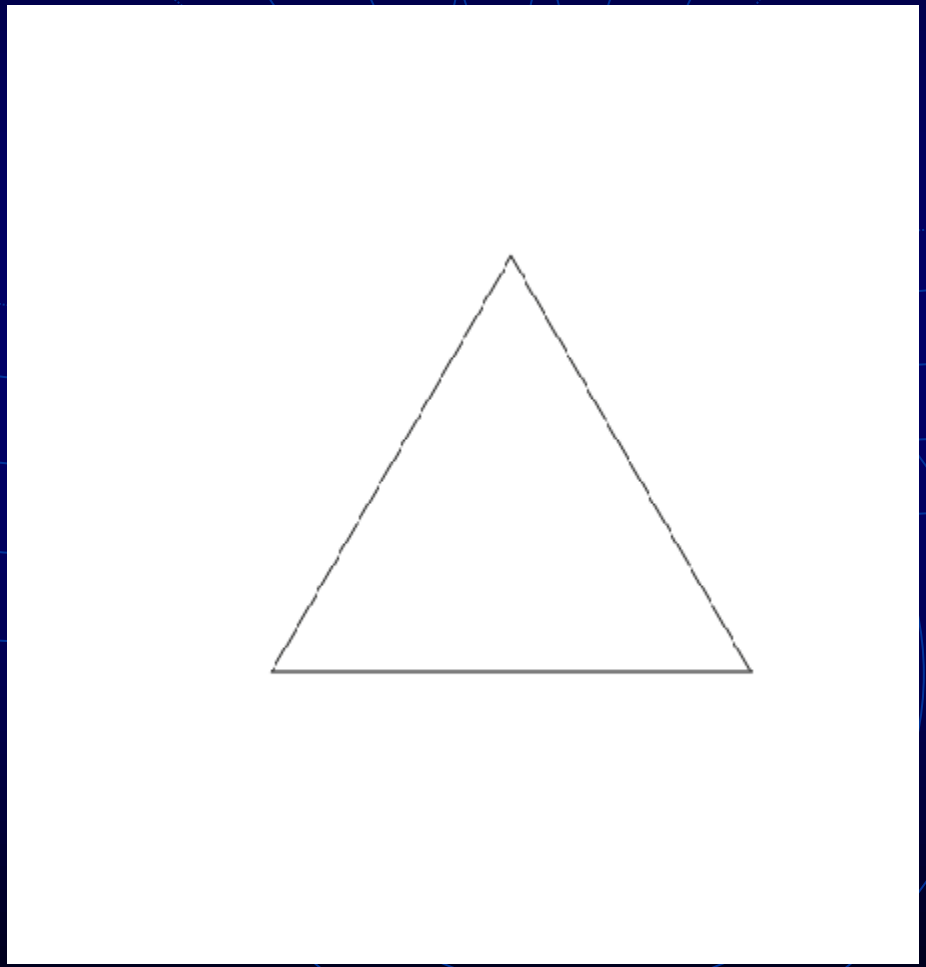
Module - III



Presented by: Arif Mohammad Abdul

Number theory: Divisibility and The Division Algorithm, The Euclidean Algorithm, Modular Arithmetic, Prime Numbers, Fermat's and Euler's Theorems, Testing for Primality, The Chinese Remainder Theorem. Public Key **Cryptography:** Principles of public key cryptosystem, RSA algorithm, security of RSA. Diffie Hellman key exchange.

Divisibility



Divisibility

- We say that a nonzero b divides a

(if ‘a’ is multiple of b) if a = bm for some m, where a, b, and m are integers.

- The notation b | a is commonly used to mean b divides a. Also, if b | a, we say that b is a divisor of a.

E.g. a = 8, b = 2 i.e., 8/2 = 4

➔ b | a if a = b * c ➔ 8 = 2 * 4

Properties of divisibility for integers

- If a | b and b | a, then a = ± b.
- If a | 1, then a = ±1. (b | a if a = b * c)

  1= a*c if a, c =1, and a, c=-1
- Any b ≠ 0 divides 0. ➔ **b | 0 = 0 * b**

  1 | a, a | a
- If a | b and b | c, then a | c:

  11 | 66 and 66 | 198 ➔ 11 | 198
- If b | g and b | h, then b(mg + nh) for arbitrary integers m and n.

Proof

- If a | b and b | a, then a = $\pm$ b.

  a | b ➔ b = a * c

  b | a ➔ a = b * d

Substitute b in a = b * d

a = a * c * d ➔ 1 = cd [1 = 1*1]

[1 = -1*-1]

This holds if c = $\pm$1 or d = $\pm$1

So a = b * ($\pm$1) ➔ a = $\pm$ b

Proof

If a | b and b | c, then a | c:

- a | b ➔ b = a*x
- b | c ➔ c = b*y
- c = a*x*y = (xy)a ➔ a | c

Proof

- If b | g and b | h, then b | (mg + nh) for arbitrary integers m and n.
- b | g ➔ g = b * c

  b | h ➔ h = b * d

Now, mg + nh = (b * c)m + (b * d)h

= b | (mg + nh)

last property example

E.g.

b = 7; g = 14; h = 63; m = 3; n = 2

7 | 14 and 7 | 63.

To show 7 | (3 * 14 + 2 * 63),

we have (3 * 14 + 2 * 63) = 7(3 * 2 + 2 * 9),

and it is obvious that 7 | (7(3 * 2 + 2 * 9))

The Division Algorithm

- Given any positive integer n and any nonnegative integer a, if we divide a by n, we get an integer quotient q and an integer remainder r that obey the following relationship:

$a = 11; \quad n = 7; \quad 11 = 1 \times 7 + 4; \quad r = 4 \quad q = 1$

$a = -11; \quad n = 7; \quad -11 = (-2) \times 7 + 3; \quad r = 3 \quad q = -2$

Figure 2.1b provides another example.

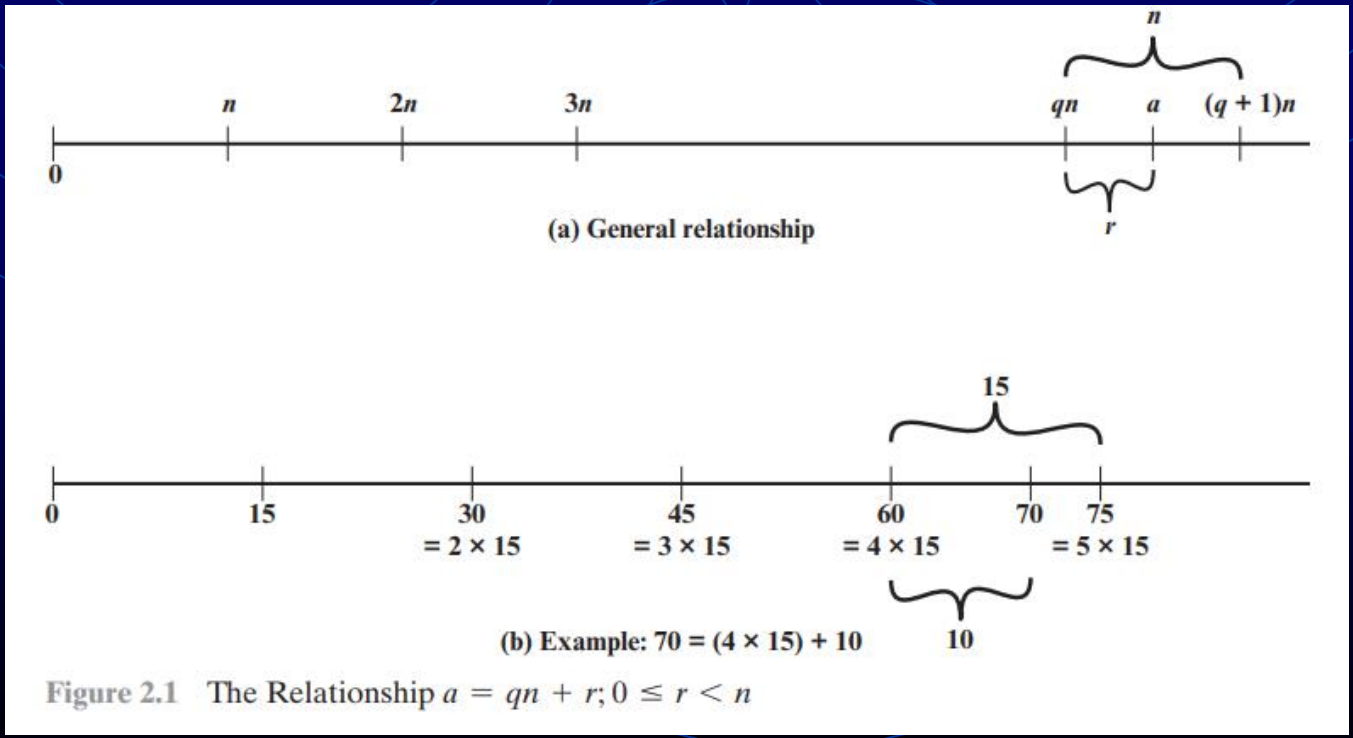


Figure 2.1 The Relationship $a = qn + r; 0 \leq r < n$

Divisibility Tests

A whole number is divisible by

- **2**,

  if its last digit is **0, 2, 4, 6,** or **8**.

  ↓

  196 is divisible by 2

- **3**,

  if the **sum of its digits is divisible by 3**.

  ↓

  117 is divisible by 3 (1+1+7=9!)

Divisibility Tests . . .

A whole number is divisible by

- **5**,

  if the **last digit is 0 or 5**.

  ↓

  2,345 is divisible by 5.

- **10**,

  if its **last digit is 0**.

  ↓

  8,470 is divisible by 10.

Divisibility Tests . . .

A whole number is divisible by

- **4**,

  if the number represented by **the last two digits** of a whole number is **divisible by 4**

  ↓

  2,344 is divisible by 4 (44 is divisible by 4)

- **6**,

  if it is **divisible by 2 and 3**

  ↓

  8,460 is divisible by 6 (check both tests)

Divisibility Tests . . .

A whole number is divisible by

- **8**,

  if the number represented by **the last three digits** of a whole number is **divisible by 8**

  ↓

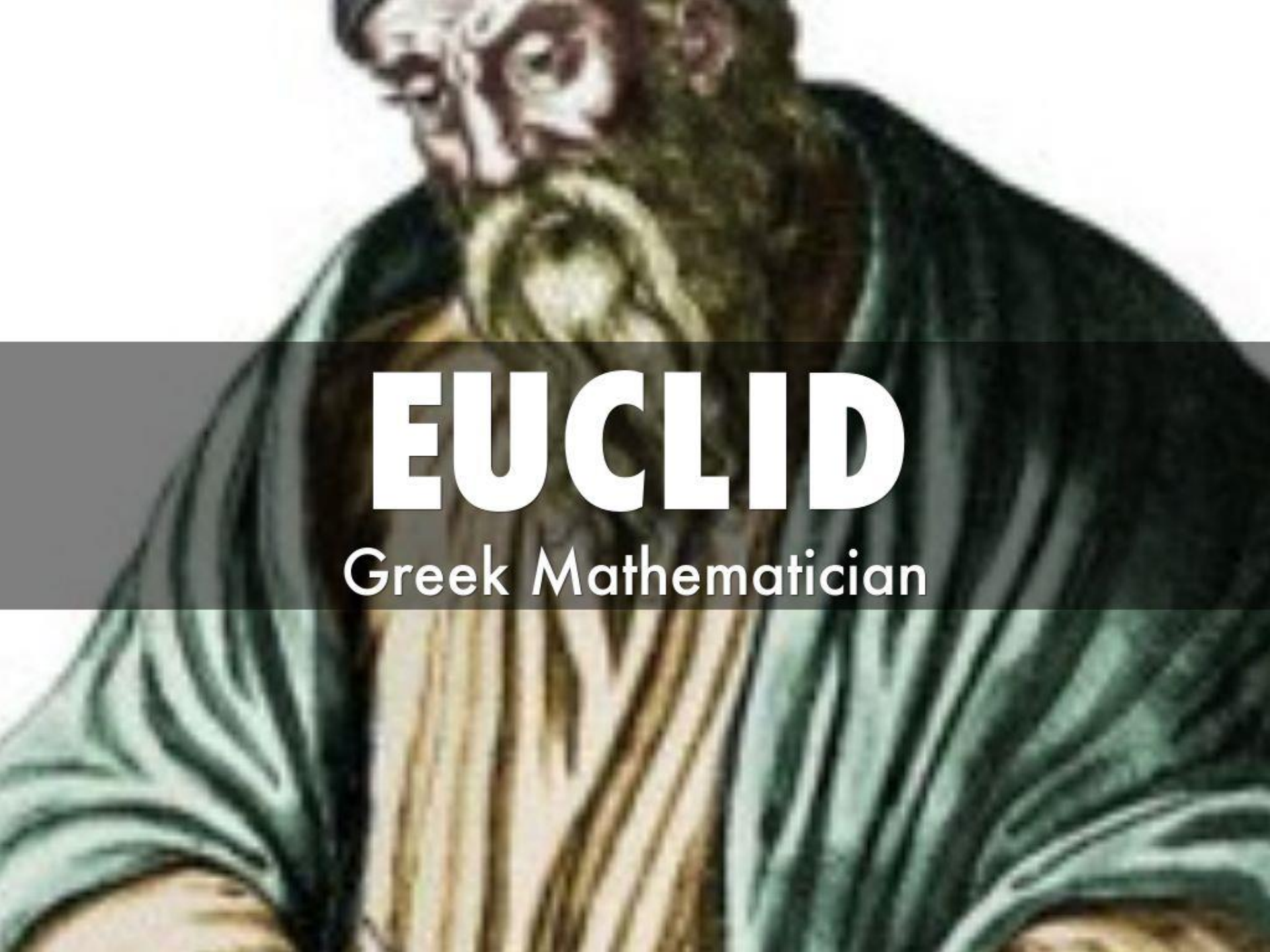
  2,064 is divisible by 8 (064 is divisible by 8)

- **9**,

  if **sum of its digits is divisible by 9**

  ↓

  8,460 is divisible by 9 (8+4+6+0= 18)


EUCLID
Greek Mathematician

Euclidean Algorithm

- **GCD (a,b) = GCD (b, a mod b)**
- **GCD (a,0) = a**
- Eg.

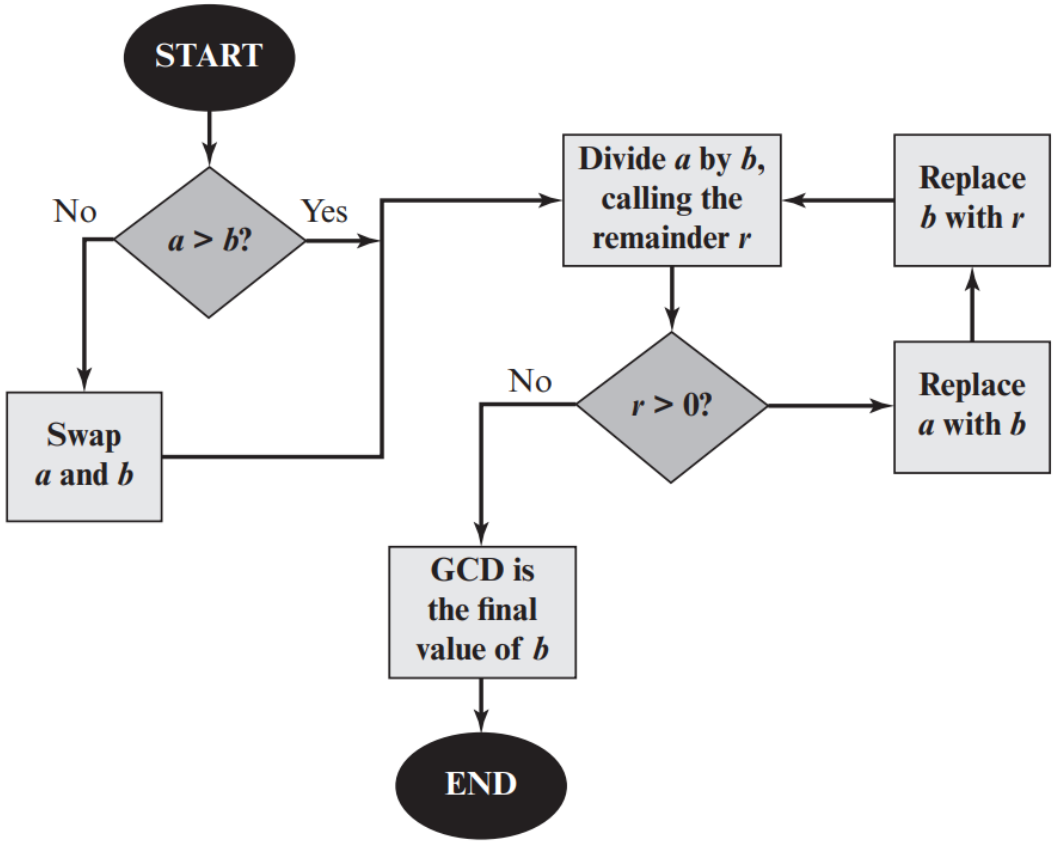
GCD(300,18)	=	GCD (18, 300 mod 18)
	=	GCD (18, 12)
GCD (18, 12)	=	GCD (12, 18 mod 12)
GCD (12, 6)	=	GCD (6, 12 mod 6)
GCD (6, 0)	=	6

Euclidean Algorithm

A simple procedure for determining the **greatest common divisor** of two positive integers.

- Notation **gcd(a, b)** to mean the greatest common divisor of a and b
- The greatest common divisor of a and b is the **largest integer** that **divides** both a and b.
- Two integers are **relatively prime** if and only if their only common positive integer factor is 1.
- The greatest common divisor be positive, gcd(a, b) = gcd(a, **-**b) = gcd(**-**a, b) = gcd(**-**a, **-**b). In general, **gcd(a, b) = gcd(|a| , |b|).**
- All nonzero integers divide 0, we have **gcd(a, 0)** = |a| .
- We define **gcd(0, 0)** = 0

Euclidean Algorithm



Euclidean Algorithm

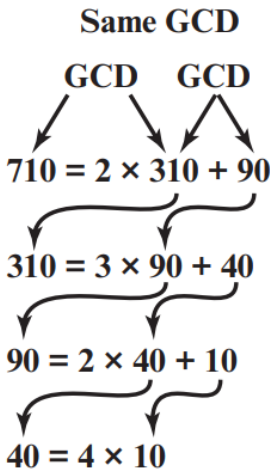


Figure 2.3 Euclidean Algorithm Example: gcd(710, 310)

Euclidean Algorithm

To find $d = \gcd(a, b) = \gcd(1160718174, 316258250)$		
$a = q_1 b + r_1$	$1160718174 = 3 \times 316258250 + 211943424$	$d = \gcd(316258250, 211943424)$
$b = q_2 r_1 + r_2$	$316258250 = 1 \times 211943424 + 104314826$	$d = \gcd(211943424, 104314826)$
$r_1 = q_3 r_2 + r_3$	$211943424 = 2 \times 104314826 + 3313772$	$d = \gcd(104314826, 3313772)$
$r_2 = q_4 r_3 + r_4$	$104314826 = 31 \times 3313772 + 1587894$	$d = \gcd(3313772, 1587894)$
$r_3 = q_5 r_4 + r_5$	$3313772 = 2 \times 1587894 + 137984$	$d = \gcd(1587894, 137984)$
$r_4 = q_6 r_5 + r_6$	$1587894 = 11 \times 137984 + 70070$	$d = \gcd(137984, 70070)$
$r_5 = q_7 r_6 + r_7$	$137984 = 1 \times 70070 + 67914$	$d = \gcd(70070, 67914)$
$r_6 = q_8 r_7 + r_8$	$70070 = 1 \times 67914 + 2156$	$d = \gcd(67914, 2156)$
$r_7 = q_9 r_8 + r_9$	$67914 = 31 \times 2156 + 1078$	$d = \gcd(2156, 1078)$
$r_8 = q_{10} r_9 + r_{10}$	$2156 = 2 \times 1078 + 0$	$d = \gcd(1078, 0) = 1078$
Therefore, $d = \gcd(1160718174, 316258250) = 1078$		

Euclidean Algorithm

Dividend	Divisor	Quotient	Remainder
$a = 1160718174$	$b = 316258250$	$q_1 = 3$	$r_1 = 211943424$
$b = 316258250$	$r_1 = 211943434$	$q_2 = 1$	$r_2 = 104314826$
$r_1 = 211943424$	$r_2 = 104314826$	$q_3 = 2$	$r_3 = 3313772$
$r_2 = 104314826$	$r_3 = 3313772$	$q_4 = 31$	$r_4 = 1587894$
$r_3 = 3313772$	$r_4 = 1587894$	$q_5 = 2$	$r_5 = 137984$
$r_4 = 1587894$	$r_5 = 137984$	$q_6 = 11$	$r_6 = 70070$
$r_5 = 137984$	$r_6 = 70070$	$q_7 = 1$	$r_7 = 67914$
$r_6 = 70070$	$r_7 = 67914$	$q_8 = 1$	$r_8 = 2156$
$r_7 = 67914$	$r_8 = 2156$	$q_9 = 31$	$r_9 = 1078$
$r_8 = 2156$	$r_9 = 1078$	$q_{10} = 2$	$r_{10} = 0$

Modular Arithmetic

The modulus

If **a** is an integer and **n** is a positive integer, we define **a mod n** as the **remainder** when a is divided by n. The integer **n** is called the modulus.

$$a = qn + r \qquad 0 \le r < n; q = \lfloor a/n \rfloor$$

$$a = \lfloor a/n \rfloor \times n + (a \bmod n)$$

11 mod 7 = 4;

-11 mod 7 = 3

Congruent modulo n

Two integers **a** and **b** are said to be congruent modulo **n**, if (a mod n) = (b mod n). This is written as $a \equiv b \ (mod) n$

$73 \equiv 4 \pmod{23}$;

Modular Arithmetic

Properties of Congruences

1. $a \equiv b \pmod{n}$ if $n \mid (a - b)$.
2. $a \equiv b \pmod{n}$ implies $b \equiv a \pmod{n}$.
3. $a \equiv b \pmod{n}$ and $b \equiv c \pmod{n}$ imply $a \equiv c \pmod{n}$

$23 \equiv 8 \pmod{5}$ because $23 - 8 = 15 = 5 \times 3$

$-11 \equiv 5 \pmod{8}$ because $-11 - 5 = -16 = 8 \times (-2)$

$81 \equiv 0 \pmod{27}$ because $81 - 0 = 81 = 27 \times 3$

Modular Arithmetic

Modular arithmetic operations

1. $[(a \bmod n) + (b \bmod n)] \bmod n = (a + b) \bmod n$
2. $[(a \bmod n) - (b \bmod n)] \bmod n = (a - b) \bmod n$
3. $[(a \bmod n) \times (b \bmod n)] \bmod n = (a \times b) \bmod n$

$11 \bmod 8 = 3; 15 \bmod 8 = 7$
$[(11 \bmod 8) + (15 \bmod 8)] \bmod 8 = 10 \bmod 8 = 2$
$(11 + 15) \bmod 8 = 26 \bmod 8 = 2$
$[(11 \bmod 8) - (15 \bmod 8)] \bmod 8 = -4 \bmod 8 = 4$
$(11 - 15) \bmod 8 = -4 \bmod 8 = 4$
$[(11 \bmod 8) \times (15 \bmod 8)] \bmod 8 = 21 \bmod 8 = 5$
$(11 \times 15) \bmod 8 = 165 \bmod 8 = 5$

To find $11^7 \bmod 13$, we can proceed as follows:

$11^2 = 121 \equiv 4 \pmod{13}$
$11^4 = (11^2)^2 \equiv 4^2 \equiv 3 \pmod{13}$
$11^7 = 11 \times 11^2 \times 11^4$
$11^7 \equiv 11 \times 4 \times 3 \equiv 132 \equiv 2 \pmod{13}$

4. Exponentiation is performed by repeated multiplication, as in ordinary arithmetic.

Extended Euclid's Algorithm

- **Find the multiplicative inverse of 7 mod 160?**
- Just try all 160 possibilities for **n** :

$$7 * n \bmod 160 = 1$$

Finally,

$$7 * 23 \bmod 160 = 1$$

23 is the multiplicative inverse of 7

Extended Euclid's Algorithm

$7 * n \bmod 160 = 1$

Q(Quotient)	R1(Divident)	R2 (Divisor)	R3(Remainder)	T1	T2	T=T1-(Q*T2)
22	160	7	6	0	1	-22
1	7	6	1	1	-22	23
6	6	1	0	-22	23	-160
	1	0		23		

Extended Euclid's Algorithm Example

23 * n mod 26 = 1

Q(Quotient)	R1(Divident)	R2 (Divisor)	R3(Remainder)	T1	T2	T=T1-(Q*T2)
1	26	23	3	0	1	-1
7	23	3	2	1	-1	8
1	3	2	1	-1	8	-9
2	2	1	0	8	-9	26
	1	0		-9		

N = q * m + r **-9 = q * 26 + r**

Prime and Composite Numbers

- A **prime number** is a natural number greater than 1 whose **only factors are 1 and itself**. The first few prime numbers are:

  2, 3, 5, 7, 11, 13, 17, 19, 23, 29, 31, 37, ...

- A **composite number** is a natural number greater than 1 that is not prime.

  Examples: 4, 12, 9,…

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
- eg. 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
```

Example

- Is 97 is a prime?

Calculate $\sqrt{97} = 9$

The prime less than 9 are 2,3,5, and 7.

If 97 is divisible by any number 2,3,5, and 7 then 97 is not a prime otherwise prime. Prime

- Is 301 is a prime?

Calculate $\sqrt{\mathbf{301}} = 17$

The prime less than 17 are 2,3,5,7,11, and 13.

If 301 is divisible by any number 2,3,5,7,11,and 13 then 301 is not a prime otherwise prime. Not Prime

Co-primes

- Two positive integers, a and b, are relatively prime or co-prime, if gcd(a,b) = 1. [Two numbers `a, b` are **relatively prime** if have **no common divisors** apart from 1]
- If p is a prime, then all integers 1 to p-1 are relatively prime to p.
- Note: the number 1 is relatively prime with any integer..

Prime Factorisation

- To **factor** a number n is to write it as a product of other numbers: n=a × b × ...
- The **prime factorisation** of a number n is when its written as a product of primes
  - **eg. 91=7×13 ; 3600=$2^4 \times 3^2 \times 5^2$;**
  - **11011 = $7 \times 11^2 \times 13$**

$$a = \prod_{p \in \mathbf{P}} p^{a_p}$$

a base p >=0

Prime Factorisation

- Multiplication of two numbers is equivalent to adding the corresponding exponents:

  k = mn ⟶ $k_p = m_p + n_p$ for all p

  k = 12 * 18 = 216

The integer 12 represented by $\{a_2 = 2, a_3 = 1\}$.

The integer 18 represented by $\{a_2 = 1, a_3 = 2\}$.

$12 = 2^2 * 3^1$ [a base 2 = 2^2]

$18 = 2^1 * 3^2$

$k = 12 \times 18 = 216$
$k_2 = 2 + 1 = 3; k_3 = 1 + 2 = 3$
$216 = 2^3 \times 3^3$

What does a|b mean?

- In terms of prime factors:

$$a|b \rightarrow a_p \leq b_p \qquad \text{for all } p$$

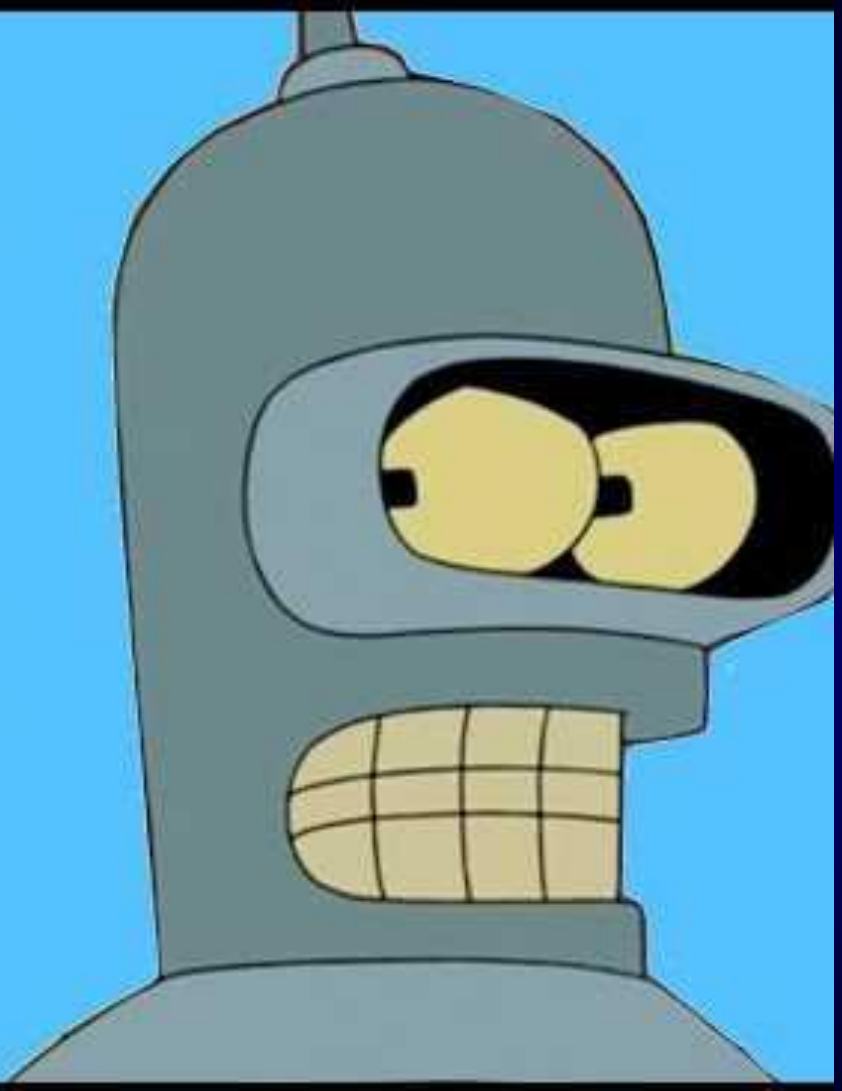
$a = 12; b = 36; 12|36; 12 = 2^2 \times 3; 36 = 2^2 \times 3^2$
$a_2 = 2 = b_2$
$a_3 = 1 \leq 2 = b_3$

Relatively Prime Numbers & GCD

- Two numbers a, b are **relatively prime** if have **no common divisors** apart from 1
  - eg. 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
  - eg. $300=2^1\times3^1\times5^2$ $18=2^1\times3^2$ hence **GCD(18,300)** $=2^1\times3^1\times5^0=6$

13 666

FERMAT'S LITTLE THEOREM

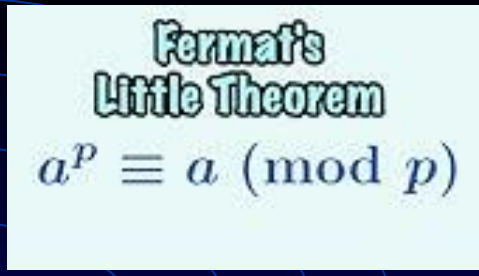


Congruence

- We use the notation **$a \equiv b \pmod{m}$** to indicate that a is congruent to b modulo m.

- In other words:

  $a \equiv b \pmod{m}$ if and only if **$a \bmod m = b \bmod m$**.



Fermat's Theorem

- **If p is prime and a is a positive integer not divisible by p,** then

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

- $a^{p-1} \bmod p = 1$
  - where p is prime and $\gcd(a,p)=1$
- also known as **Fermat's Little Theorem**
- useful in **public key**.

Example

$a = 7, p = 19$
$7^2 = 49 \equiv 11 \bmod 19$
$7^4 \equiv 121 \equiv 7 \bmod 19$
$7^8 \equiv 49 \equiv 11 \bmod 19$
$7^{16} \equiv 121 \equiv 7 \bmod 19$
$a^{p-1} = 7^{18} = 7^{16} \times 7^2 \equiv 7 \times 11 \equiv 1 \bmod 19$

- a = 7 , p = 19

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

- **Alternative of Fermat's.**

If **p** is prime and **a** is any positive integer, then

$a^p \equiv a \bmod p \qquad a<p$

(i) P=5, a= 3 (ii) p =5 , a =10

$$a^p \equiv a \bmod p$$

$p = 5, a = 3, 3^5 = 243 \equiv 3 \bmod 5$
$p = 5, a = 10, 10^5 = 100000 \equiv 10 \bmod 5 \equiv 0 \bmod 5$

Euler's totient function

Euler Totient Function ∅(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
  - Eg for n =4, complete residue set is {0,1,2,3} then reduced residue set is {1,3}.
  - How? gcd(1,4) =1, gcd(2,4) = 2 , gcd (3,4) = 1.
  - 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)** i.e., **ø(4) = 2 , ø(10) = 4**

Euler Totient Function ø(n)

- To compute ø(n) need to count number of elements to be excluded
- In general need prime factorization, but
  - ø(p) = p-1 if p is a prime.
  - ø(p x q) = ø(p)x ø(q) =(p-1)x(q-1)

    Relatively prime — Prime
  - ø(1) = 0
  - $ø(p^e) = p^e - p^{e-1}$ **or**
  - **If a and b are composite then φ(n) = n *(1 - 1/p_1)* ... (1 - 1/p_n)**

$9 = 3^2$, $\varphi(9) = 3^2 - 3^{2-1} = 9 - 3 = 6$ or

φ(n) = n *(1 - 1/p_1)* ... (1 - 1/p_n) ➔ $9 = 3^2$, $\varphi(9) = 9 * (1-1/3) = 6$

$15 = 3*5$, $\varphi(15) = 15 * (1-1/3)*(1-1/5) = 15*(2/3)*(4/5) = 8$

Example

- eg.
  - ø(37) = 37-1 37 is a prime
  - ø(21) = ø(3)x ø(7) = 3-1 * 7-1
    = 2 * 6 = 12
    21 is not a prime
  - ø(77) = ?
  - ø(143) = ?
  - ø(35) = ?
  - ø(81) = ?

Example

n	$\phi(n)$
1	1
2	1
3	2
4	2
5	4
6	2
7	6
8	4
9	6
10	4

n	$\phi(n)$
11	10
12	4
13	12
14	6
15	8
16	8
17	16
18	6
19	18
20	8

n	$\phi(n)$
21	12
22	10
23	22
24	8
25	20
26	12
27	18
28	12
29	28
30	8


Euler's Theorem

Euler's Theorem

- A generalisation of Fermat's Theorem
- $a^{\phi(n)} \equiv 1 \bmod n$

  we can write $a^{\phi(n)} \bmod n = 1$
  - where $\gcd(a,n)=1$
- eg.
  - $a=3; n=10;\ \phi(10)=4;$
  - hence $3^4 = 81 = 1 \bmod 10$
  - $a=2; n=11;\ \phi(11)=10;$
  - hence $2^{10} = 1024 = 1 \bmod 11$

Chinese Remainder Theorem

$$x \equiv 2 \mod 3$$
$$x \equiv 3 \mod 5$$
$$x \equiv 2 \mod 7$$

The Chinese Remainder Theorem

- Used to speed up modulo computations
- Working modulo a product of numbers
  - eg. mod $M = m_1 m_2 .. m_k$
- Chinese Remainder theorem lets us work in each moduli m_i separately
- Since computational cost is proportional to size, this is faster than working in the full modulus M

The Chinese Remainder Theorem

- **Step 1: Find $M = m_1 \times m_2 \times \ldots\ldots m_k$.**
- **Step 2: Find $M_1 = M/m_1$, $M_2 = M/m_2$, ……, $M_k = M/m_k$.**
- **Step 3: Find multiplicative inverse of $M_1, M_2 \ldots M_k$**
- **Step 4: Compute solution using**

$$X = (M_1 \times M_1^{-1} \times a_1 + M_2 \times M_2^{-2} \times a_2 + \ldots M_k \times M_k^{-k} \times a_k) \bmod M$$

CRT Example

- **$x \equiv 5 \pmod 7$, $x \equiv 3 \pmod{11}$, $x \equiv 10 \pmod{13}$**
- **We can rewrite as**

  $x \bmod 7 = 5$

  $x \bmod 11 = 3$

  $x \bmod 13 = 10$
- Step 1: $m_1 = 7$, $m_2 = 11$, $m_3 = 13$

  $M = m_1 \times m_2 \times m_3$

  $M = 7 \times 11 \times 13 = 1001$
- Step 2: $M_1 = M/m_1$, $M_2 = M/m_2$, $M_3 = M/m_3$

  $M_1 = 1001/7 = 143$, $M_2 = 1001/11 = 91$, $M_3 = 1001/13 = 77$

Example of CRT

- Step 3: Multiplicative inverse of M_1,M_2 & M_3

  $M_1^{-1}=143^{-1} \bmod 7 = 3^{-1} \bmod 7 = 5$ ($3 * n \bmod 7 = 1$)

  $M_2^{-1}=91^{-1} \bmod 11 = 3^{-1} \bmod 11 = 4$ ($3 * n \bmod 11 = 1$)

  $M_3^{-1}=77^{-1} \bmod 13 = 12^{-1} \bmod 13 = 12$ ($12 * n \bmod 13 = 1$)

- **Step 4:**

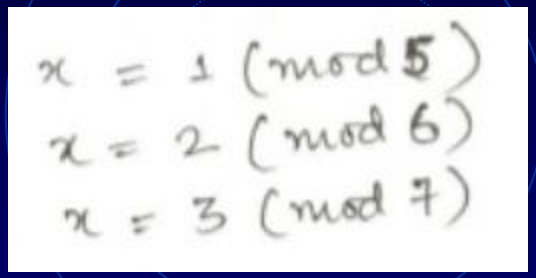
  $$X = (M_1 \times M_1^{-1} \times a_1 + M_2 \times M_2^{-2} \times a_2 + \ldots M_k \times M_k^{-k} \times a_k) \bmod M$$

  $$a_1 = 5,\ a_2 = 3,\ a_3 = 10$$

- $x = (143 \times 5 \times 5 + 91 \times 4 \times 3 + 77 \times 12 \times 10) \bmod 1001$
  $= 13907 \bmod 1001 = 894$

CRT Example

- $x \equiv 1 \pmod 5$,
- $x \equiv 1 \pmod 7$,
- $x \equiv 3 \pmod{11}$



$x \equiv 4 \pmod 5$

$x \equiv 2 \pmod 7$

$x \equiv 5 \pmod 6$

Private-Key Cryptography

- Traditional **private/secret/single key** cryptography uses **one** key
- Key is shared by both sender and receiver
- **If the key is disclosed communications are compromised**
- **Also known as symmetric, both parties are equal**
  - hence does not protect sender from receiver forging a message & claiming is sent by sender

Public-Key Cryptography

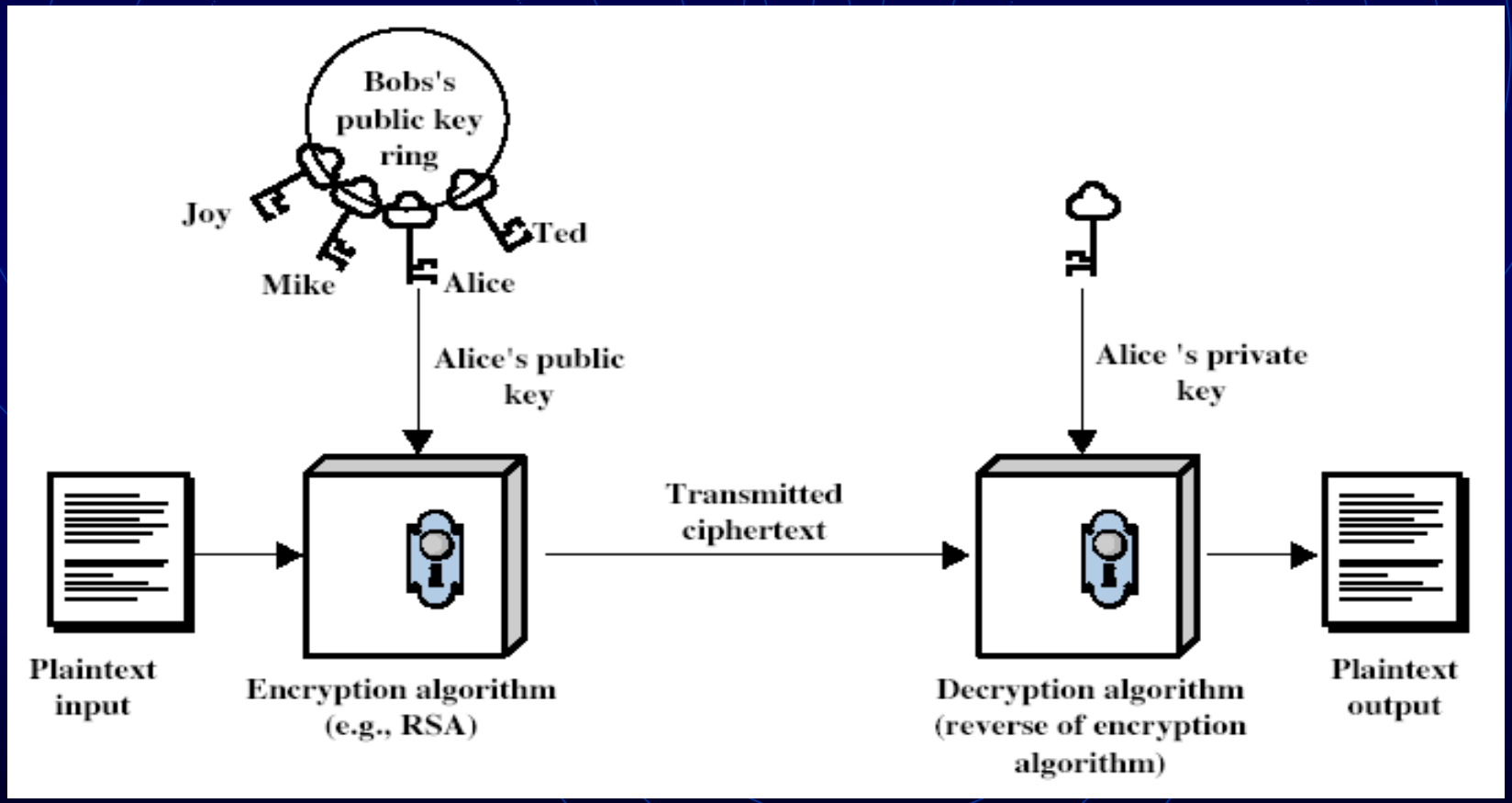
- **Probably most significant advance in the 3000-year history of cryptography**
- **Uses two keys – a public key and a private key**
- **Asymmetric since parties are not equal**
- **Uses clever application of number theory concepts to function**
- **Complements rather than replaces private key cryptography**

Public-Key Cryptography

Public-Key Cryptography

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

Public-Key Cryptography



Why Public-Key Cryptography?

- Developed to address two key issues:
  - **Key distribution** – how to have secure communications in general without having to trust a KDC with your key
  - **Digital signatures** – how to verify a message comes intact from the claimed sender

Public-Key Characteristics

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

Public-Key Cryptosystems

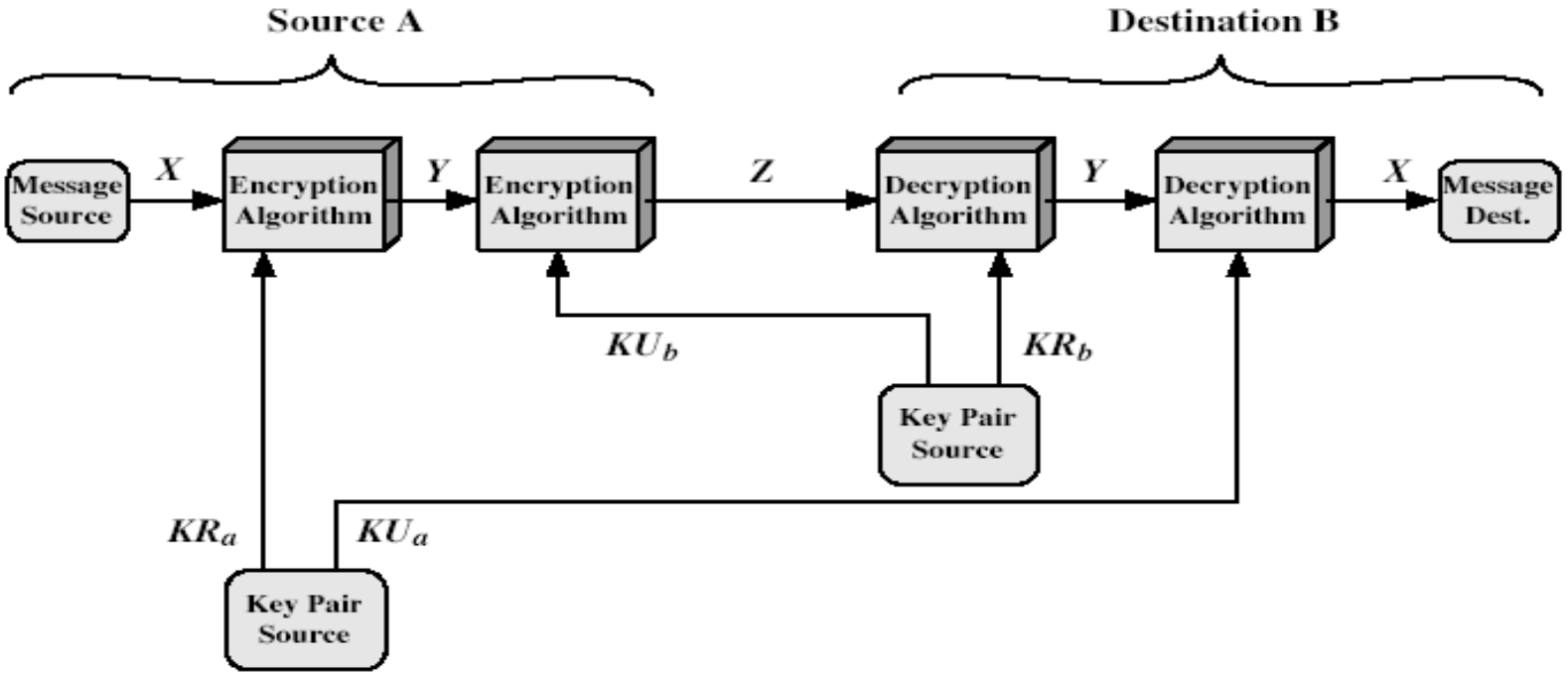


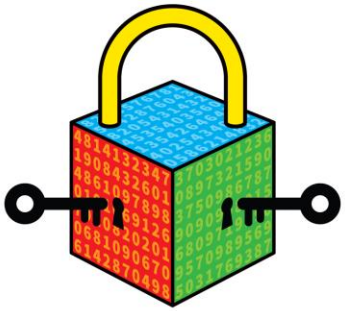
Figure 9.4 Public-Key Cryptosystem: Secrecy and Authentication

Public-Key Applications

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

Security of Public Key Schemes

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



Asymmetric encryption

Alice's private key

Alice's public key

Principles of Public-key Cryptosystem

- **Public-key Cryptosystem** evolved from an attempt to attack of two of most difficult problems.
    1. Key Distribution
    2. Digital Signatures

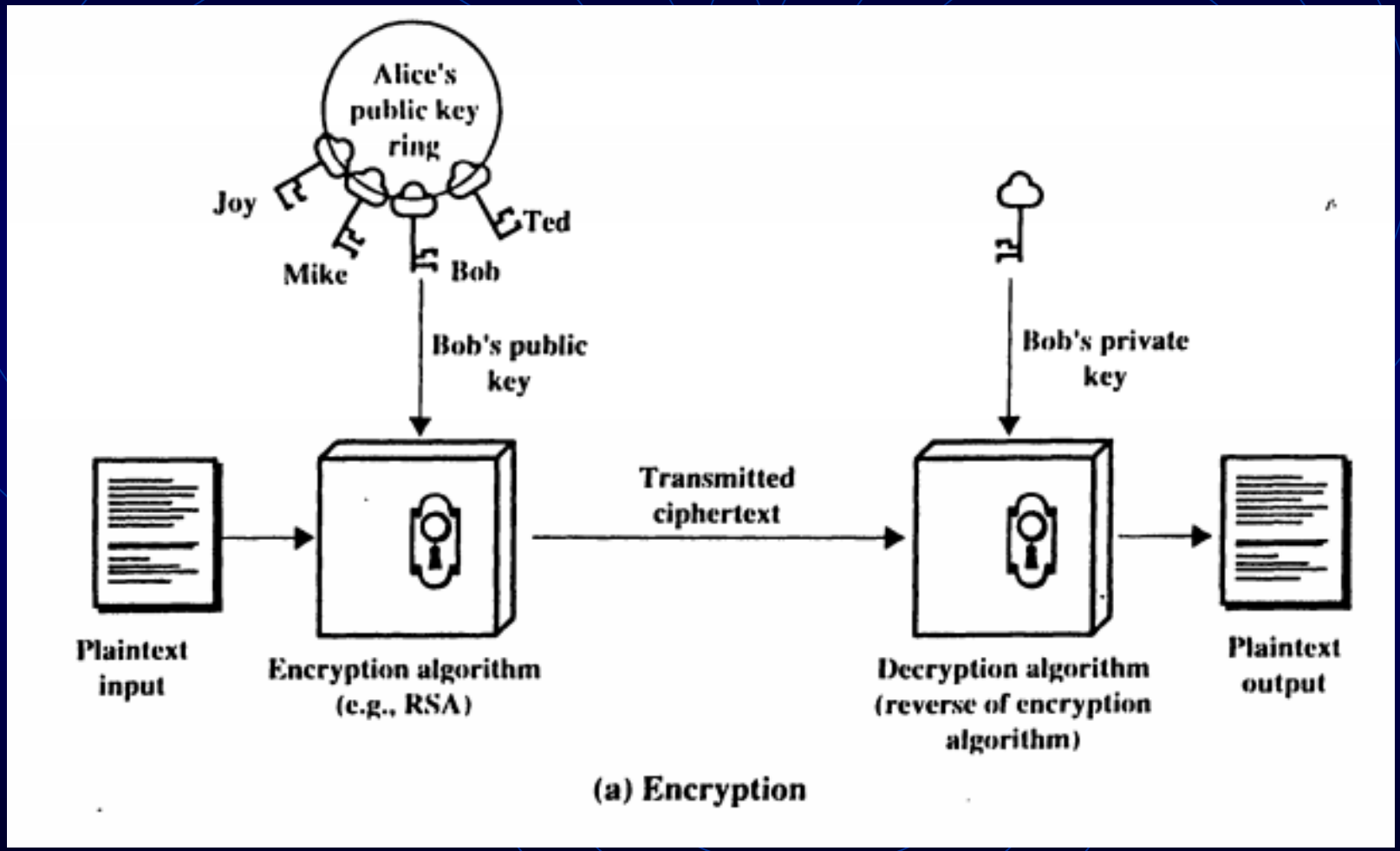
Public-key Cryptosystem

- **Public-key Cryptosystem** scheme has six ingredients:
  1. Plaintext
  2. Encryption Algorithm
  3. Public and Private key
  4. Ciphertext
  5. Decryption Algorithm
  6. secret key

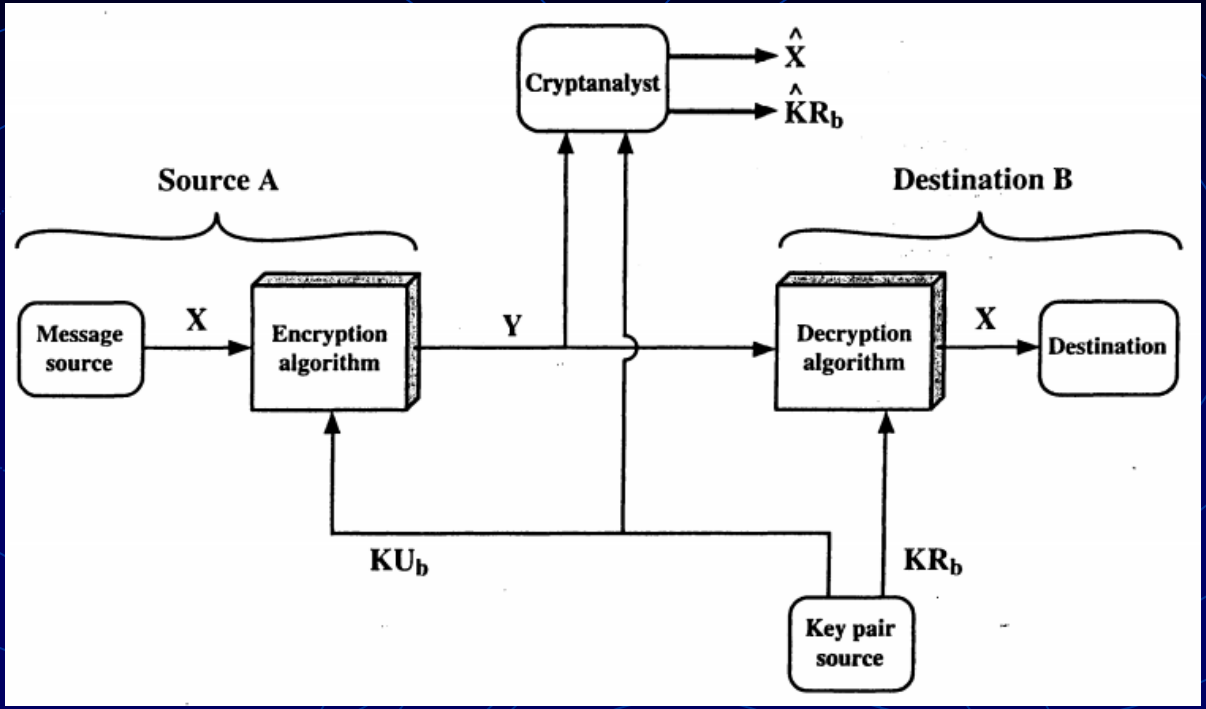
Essential steps of Public-key Cryptosystem

1. Each user generate a pair of keys.
2. Each user places one of the two keys in a public register.
3. If Bob wishes to send confidential message to Alice, Bob encrypts the message using Alice's public key.
4. When Alice receives the message, she decrypts using her private key.
5. No one can decrypt other than Alice because Alice only knows private key.

Essential steps of Public-key Cryptosystem



Public-key Cryptosystem



1. With the message X and encryption key Ku_b as input, Y is ciphertext.

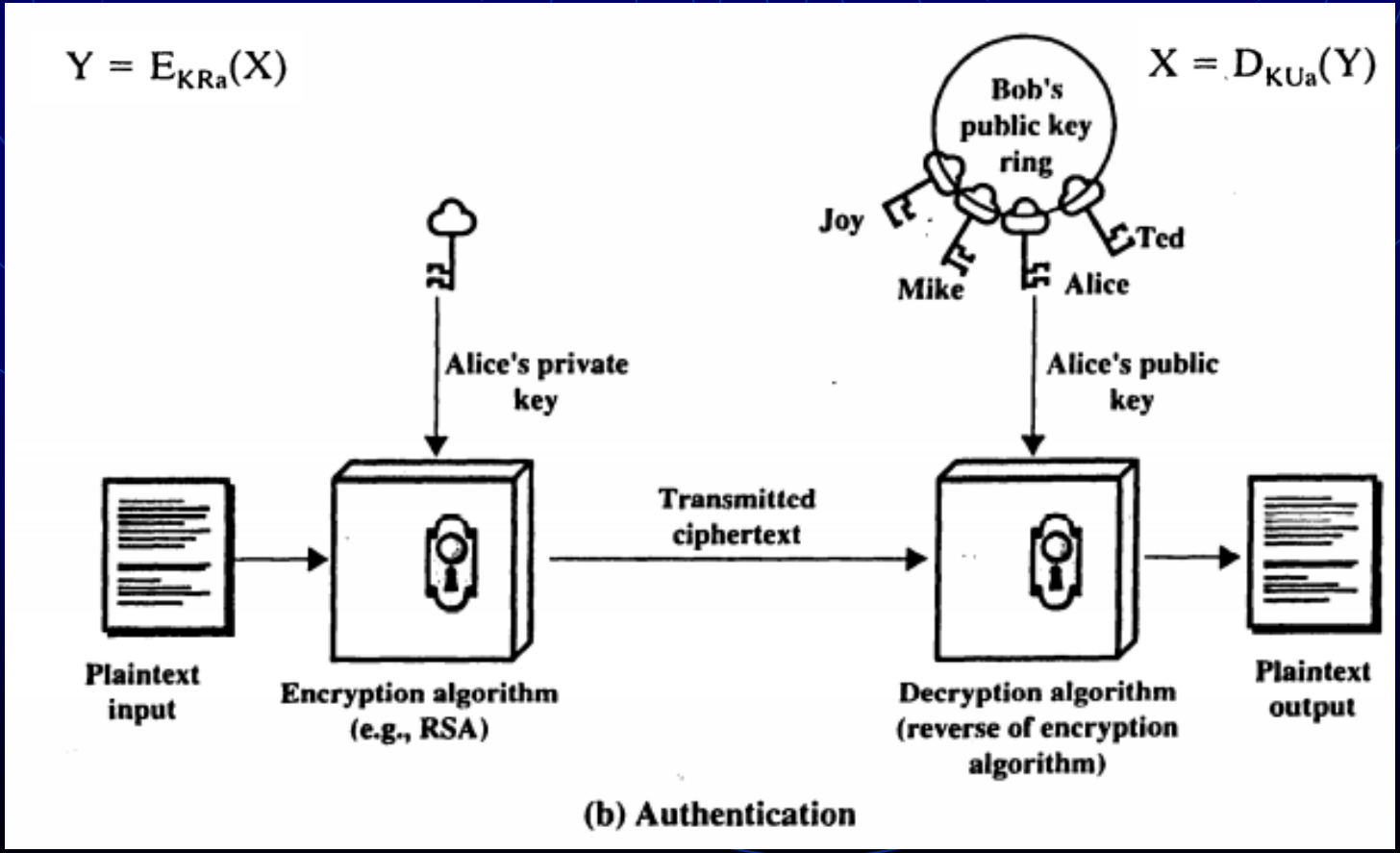
$$Y = E_{KUb}(X)$$

2. The intended receiver, in possession of the matching private key, is able to invert the transformation.

$$X = D_{KRb}(Y)$$

Public-key Cryptosystem

Use of **Public-key encryption** to provide authentication. **Entire encrypted message serves as a digital signature.**



Public-key Cryptosystem: Secrecy (Authentication and Confidentiality)

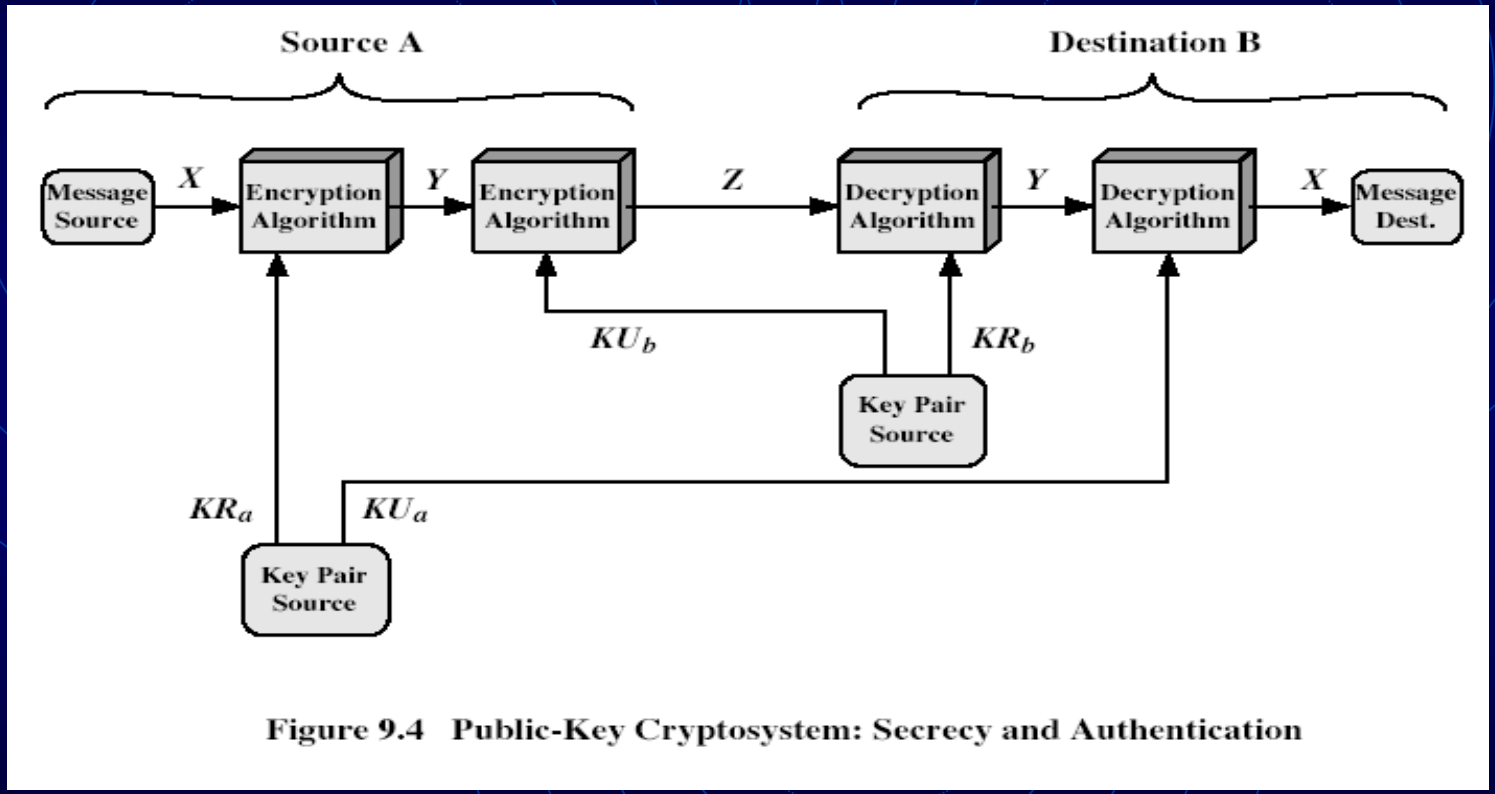


Figure 9.4 Public-Key Cryptosystem: Secrecy and Authentication

$$Z = E_{KUb}[E_{KRa}(X)]$$

$$X = D_{KUa}[D_{KRb}(Z)]$$

Application of Public-key Cryptosystem

1. Encryption/Decryption
2. Digital Signature
3. Key Exchange

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

Requirements of Public-key Cryptosystem

1. It is computationally easy for a party B to generate a pair (public key KU_b, private key KR_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:

$$C = E_{KUb}(M)$$

3. It is computationally easy for the receiver B to decrypt the resulting ciphertext using the private key to recover the original message:

$$M = D_{KRb}(C) = D_{KRb}[E_{KUb}(M)]$$

4. It is computationally infeasible for an opponent, knowing the public key, KU_b, to determine the private key, KR_b.
5. It is computationally infeasible for an opponent, knowing the public key, KU_b, and a ciphertext, C, to recover the original message, M.

We can add a sixth requirement that, although useful, is not necessary for all public-key applications:

6. The encryption and decryption functions can be applied in either order:

$$M = E_{KUb}[D_{KRb}(M)]$$

Trapdoor

A One-way function: Easy to calculate in one direction and infesible to calculate in the other direction unless knows some addition information.

$Y = f_k(X)$	easy, if k and X are known
$X = f_k^{-1}(Y)$	easy, if k and Y are known
$X = f_k^{-1}(Y)$	infeasible, if Y is known but k is not known

RSA

RIVEST
SHAMIR
ADELMAN

RSA

- by Rivest, Shamir & Adleman of MIT in 1977
- best known & widely used public-key scheme
- based on exponentiation in a finite (Galois) field over integers modulo a prime
  - exponentiation takes $O((\log n)^3)$ operations (easy)
- uses large integers (eg. 1024 bits or 309 decimal digits)
- security due to cost of factoring large numbers
  - factorization takes $O(e^{\log n \log \log n})$ operations (hard)

RSA Key Setup

Key Generation

Select p, q	p and q both prime
Calculate $n = p \times q$	
Calculate $\phi(n) = (p-1)(q-1)$	
Select integer e	$\gcd(\phi(n), e)=1; 1 < e < \phi(n)$
Calculate d	$d=e^{-1} \bmod \phi(n)$
Public key	$KU=\{e,n\}$
Private key	$KR=\{d,n\}$

Encryption

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

Decryption

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

RSA Key Setup

- each user generates a public/private key pair by:
- selecting two large primes at random - p, q
- computing their system modulus $N=p.q$
  - note $\phi(N)=(p-1)(q-1)$
- selecting at random the encryption key e
    - where $1<e<\phi(N)$, $\gcd(e,\phi(N))=1$
- solve following equation to find decryption key d
  - $e.d=1 \mod \phi(N)$ and $0 \le d \le N$
- publish their public encryption key: KU={e,N}
- keep secret private decryption key: KR={d,p,q}

RSA Use

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

RSA Example

1. Select primes: $p=17$ & $q=11$
2. Compute $n = pq = 17 \times 11 = 187$
3. Compute $\phi(n) = (p-1)(q-1) = 16 \times 10 = 160$
4. Select e : $\gcd(e,160)=1$; choose $e=7$
5. Determine d: $de=1 \mod 160$ and $d < 160$
   Value is $d=23$ since $23 \times 7 = 161 = 10 \times 16 + 1$

   7 * n mod 160

1. Publish public key $KU=\{7,187\}$
2. Keep secret private key $KR=\{23,17,11\}$

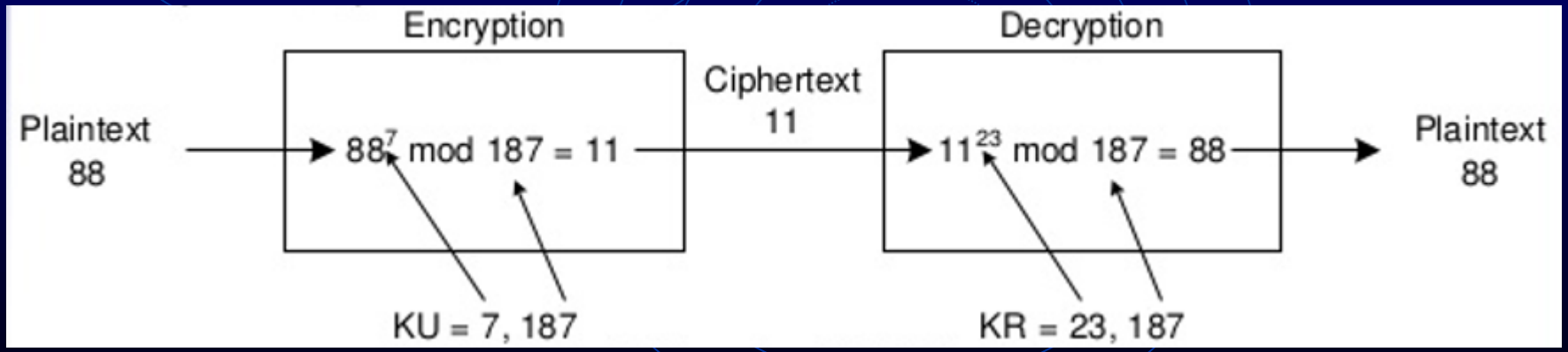
RSA Example

- sample RSA encryption/decryption is:
- given message $M = 88$ ($88<187$)
- encryption:

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

- decryption:

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



RSA Encryption

- Encryption: $C = 88^7 \bmod 187$

$C = [(88^4 \bmod 187)(88^2 \bmod 187)(88^1 \bmod 187)] \bmod 187$

$88^1 \bmod 187 = 88$

$88^2 \bmod 187 = 7744 \bmod 187 = 77$

$88^4 \bmod 187 = [77 * 77] \bmod 187 = 132$

$88^7 \bmod 187 = [88*77*132] \bmod 187 = 11$

RSA Decryption

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

$M = [(11^1 \bmod 187)(11^2 \bmod 187)(11^4 \bmod 187)(11^8 \bmod 187)(11^8 \bmod 187)] \bmod 187$

$11^1 \bmod 187 = 11$

$11^2 \bmod 187 = 121 \bmod 187 = 121$

$11^4 \bmod 187 = [11 * 121] \bmod 187 = 55$

$11^8 \bmod 187 = [55*55] \bmod 187 = 33$

$11^8 \bmod 187 = [55*55] \bmod 187 = 33$

$11^{23} \bmod 187 = [11* 121* 55* 33* 33] \bmod 187 = 88$

RSA Example

1. Select primes: $p=7$ & $q=17$
2. Compute $n = pq =$
3. Compute $\phi(n)=(p-1)(q-1)=$
4. Select e : $\gcd(e,)=1$; choose $e=$
5. Determine d:
6. Publish public key $KU=\{,\}$
7. Keep secret private key $KR=\{,,\}$

RSA Example

1. Select primes: $p=7$ & $q=11$
2. Compute $n = pq =$
3. Compute $\phi(n)=(p-1)(q-1)=$
4. Select e : $\gcd(e,)=1$; choose $e=$
5. Determine d:
6. Publish public key $KU=\{,\}$
7. Keep secret private key $KR=\{,,\}$

- Plaintext = 5

RSA Example

1. Select primes: $p=3$ & $q=11$
2. Compute $n = pq =$
3. Compute $\phi(n)=(p-1)(q-1)=$
4. Select e : $\gcd(e,)=1$; choose $e=7$
5. Determine d:
6. Publish public key $KU=\{,\}$
7. Keep secret private key $KR=\{,,\}$

- Plaintext M = 5

RSA Example

1. Select primes: $p=5$ & $q=11$
2. Compute $n = pq =$
3. Compute $\phi(n)=(p-1)(q-1)=$
4. Select e : $\gcd(e,)=1$; choose $e=3$
5. Determine d:
6. Publish public key $KU=\{,\}$
7. Keep secret private key $KR=\{,,\}$

- Plaintext M = 9

RSA Example

1. Select primes: $p=7$ & $q=11$
2. Compute $n = pq =$
3. Compute $\phi(n)=(p-1)(q-1)=$
4. Select e : $\gcd(e,)=1$; choose $e=17$
5. Determine d:
6. Publish public key $KU=\{,\}$
7. Keep secret private key $KR=\{,,\}$

- Plaintext M = 8

RSA Efficiency

- Both encryption and decryption in RSA involve raising an integer to an integer power, mod n.
- If the exponentiation is done over the integers and then reduced modulo n.

  [(a mod n) * (b mod n)] mod n = (a * b) mod
- we can reduce intermediate results modulo n. This makes the calculation practical.
- RSA dealing with potentially large exponents.

Efficiency of Exponentiation

- To see how efficiency might be increased, consider that we wish to compute x^{16}.
- A straightforward approach requires 15 multiplications:

$$x^{16} = x * x * x * x * x * x * x * x * x * x * x * x * x * x * x$$

- **Also, we can achieve Fermat's theorem in the form of x^2, x^4, x^6, x^8, x^{16}**

Exponentiation Example

- Calculate x^{11} mod n for some integers x and n.

$$x^{11} = (x)(x^2)(x^8) = x^{1+2+8}.$$

- we compute x mod n, x^2 mod n, and x^8 mod n and then calculate [(x mod n) * (x^2 mod n) * (x^8 mod n)] mod n.

$a^b \bmod n$

- More generally, suppose we wish to find the value $a^b \bmod n$ with a, b, and n positive integers.
- If we express b as a binary number $b_k b_{k-1} \ldots b_0$, then we have

$$b = \sum_{b_i \neq 0} 2^i$$

EFFICIENT OPERATION USING THE PUBLIC KEY

Example: $a^b \bmod n$

a = 7, b = 560 = 1000110000, and n = 561

b = 1 0 0 0 1 1 0 0 0 0

i = 9 8 7 6 5 4 3 2 1 0

```
c ← 0; f ← 1
for i ← k downto 0
    do  c ← 2 × c
        f ← (f × f) mod n
    if  b_i = 1
        then c ← c + 1
             f ← (f × a) mod n
return f
```

i	9	8	7	6	5	4	3	2	1	0
b_i	1	0	0	0	1	1	0	0	0	0
c	1	2	4	8	17	35	70	140	280	560
f	7	49	157	526	160	241	298	166	67	1

EFFICIENT OPERATION USING THE PRIVATE KEY

- $M = C^d \bmod n$
- If a small value of d is vulnerable to a brute-force attack and other forms of cryptanalysis.
- So increase the ‘d’ value. If the ‘d’ value is increased, computation also increases.
- There is a way to speed up computation using Fermat’s theorem and the CRT.

EFFICIENT OPERATION USING THE PRIVATE KEY

- We wish to compute the value $M = C^d \bmod n$
- Let us define the following intermediate results

$$V_p = C^d \bmod p \quad V_q = C^d \bmod q$$

Following the CRT using Equation (8.8), define the quantities

$$X_p = q \times (q^{-1} \bmod p) \quad X_q = p \times (p^{-1} \bmod q)$$

The CRT then shows, using Equation (8.9), that

$$M = (V_p X_p + V_q X_q) \bmod n$$

EFFICIENT OPERATION USING THE PRIVATE KEY

- **Example:** $M = C^d \bmod n$

$C = 3$, $d = 101$ and $n = 77$ [7 * 11]

$3^{101} \bmod n$

Using Fermat's $a^{p-1} = 1 \bmod p$

$v_p = C^d \bmod p$ $(p = 7)$ $\quad v_q = C^d \bmod q$ $\quad (q = 11)$

$v_p = 3^{101} \bmod 7 \quad v_q = 3^{101} \bmod 11$

$$V_p = C^d \bmod p = C^{d \bmod (p-1)} \bmod p \quad V_q = C^d \bmod q = C^{d \bmod (q-1)} \bmod q$$

$v_p = 3^{101 \bmod 6} \bmod 7 \quad v_q = 3^{101 \bmod 10} \bmod 11$

$v_p = 3^5 \bmod 7 \quad v_q = 3^{1 \bmod 10} \bmod 11$

$v_p = 5 \qquad v_q = 3$

EFFICIENT OPERATION USING THE PRIVATE KEY

Using CRT:

$$X_p = q \times (q^{-1} \bmod p) \quad X_q = p \times (p^{-1} \bmod q)$$

$X_p = 11 * (11^{-1} \bmod 7)$ $X_q = 7 * (7^{-1} \bmod 11)$

Multiplicative inverse: a * n mod m = 1

$X_p = 11 * (11 * n \bmod 7)$ $X_q = 7 * (7 * n \bmod 11)$

$X_p = 11 * (11 * 2 \bmod 7)$ $X_q = 7 * (7 * 8 \bmod 11)$

$X_p = 11 * 2$ $X_q = 7 * 8$

$$M = (V_p X_p + V_q X_q) \bmod n$$

M = (5 * 22 + 3 * 56) mod 77 = (110 + 161) mod 77

M = 278 mod 77 = 47

RSA Security

- Five approaches to attacking RSA:
  - Brute force key search (infeasible given the size of numbers)
  - Mathematical attacks (based on the difficulty of computing ø(N), by factoring modulus N)
  - Timing attacks (on running time of decryption)
  - Hardware fault-based attack: This involves inducing hardware faults in the processor that is generating digital signatures.
  - Chosen ciphertext attacks: This type of attack exploits the properties of the RSA algorithm.

What is the Diffie-Hellman key exchange?

The Problem of Key Exchange

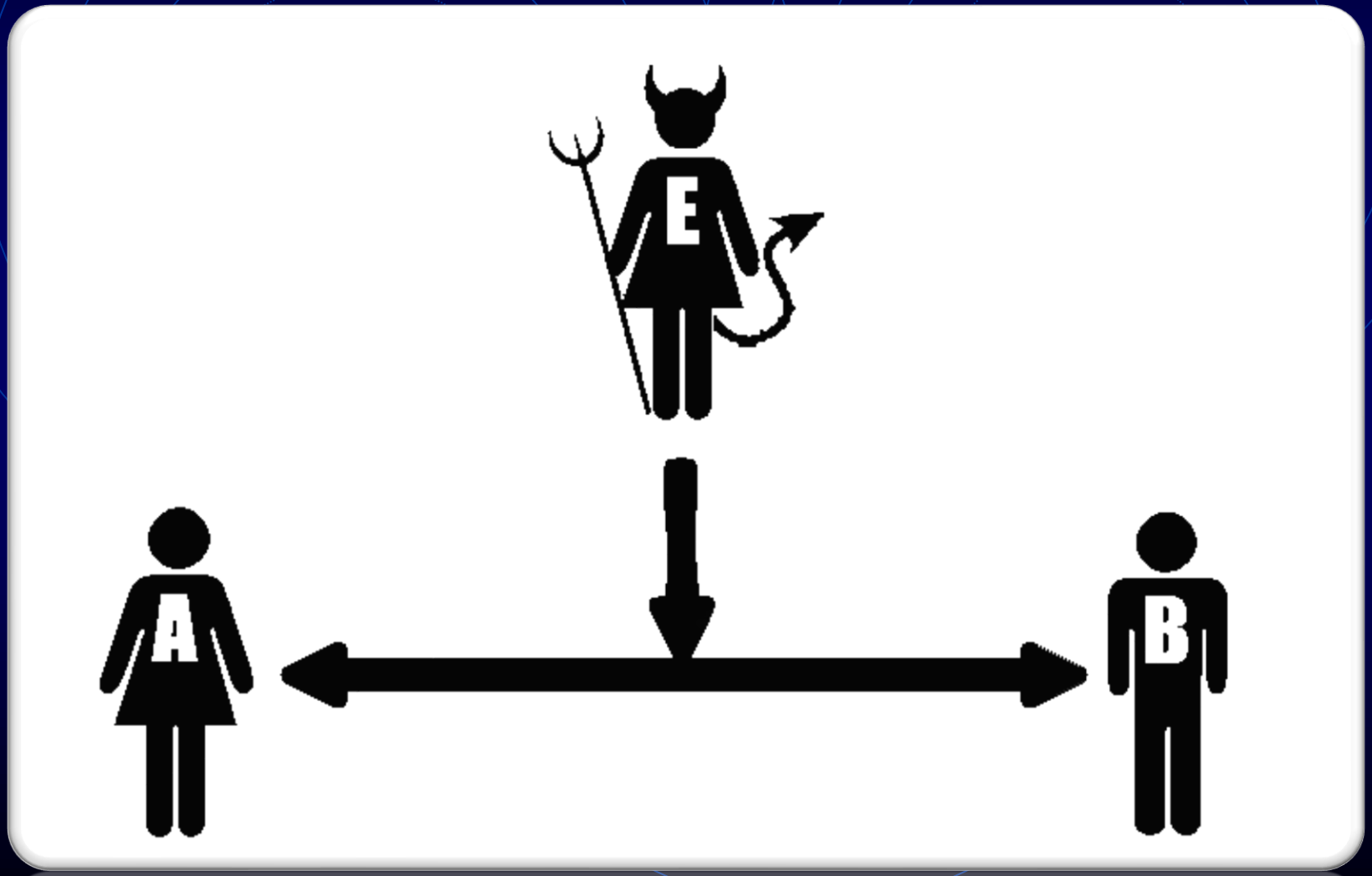
- One of the main problems of symmetric key encryption is it requires a secure & reliable channel for the shared key exchange.

- The Diffie-Hellman Key Exchange protocol offers a way in which a public channel can be used to create a confidential shared key.

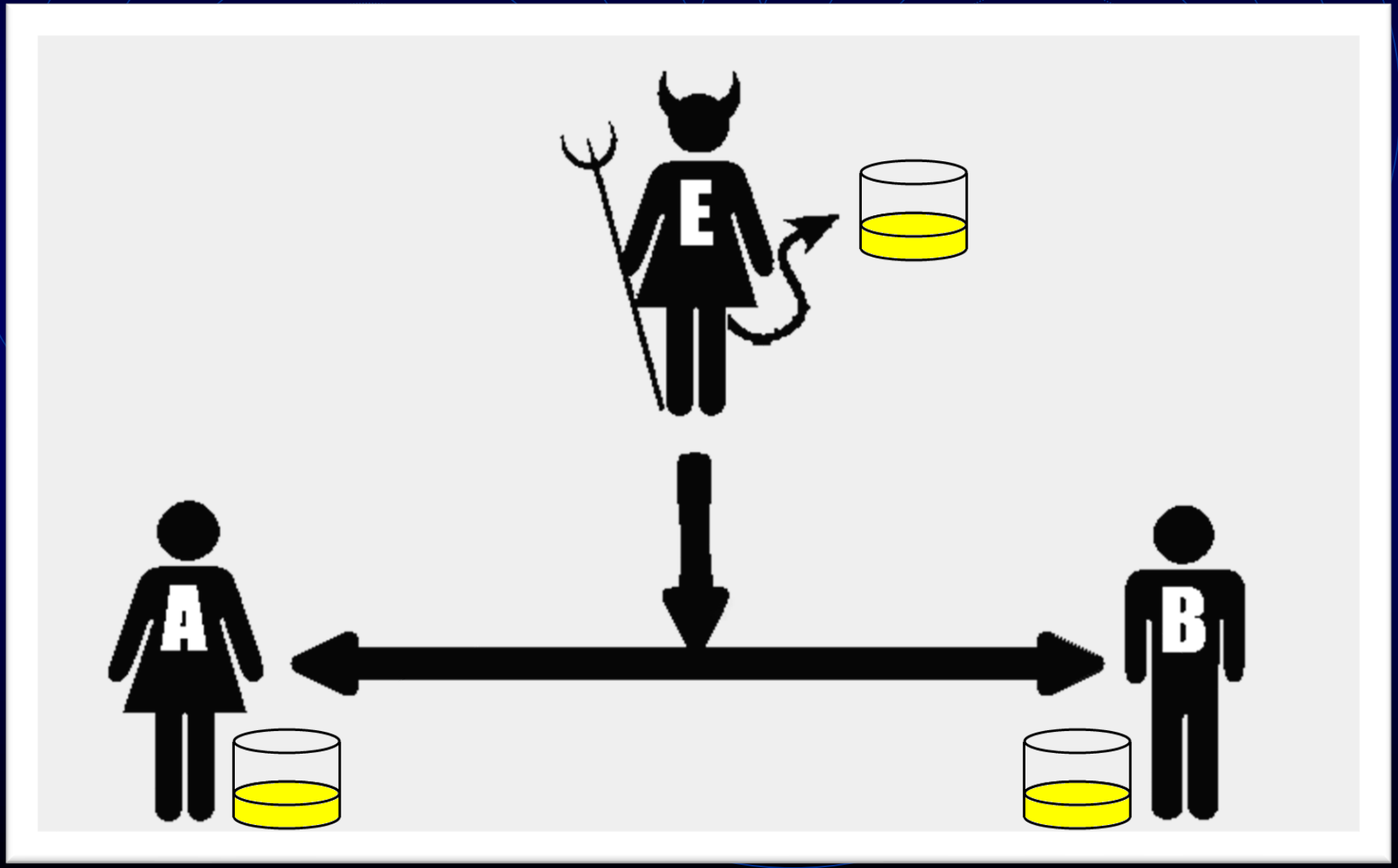
A Difficult One-Way Problem

- The first thing we require is a simple real-world operation that is easy to ***Do*** but hard to ***Undo***.
  - You can ring a bell but not unring one.
  - Toothpaste is easy to squeeze out of a tube but famously hard to put back in.
- In our example we will use *Mixing Colors*.
  - Easy to mix 2 colors, hard to unmix

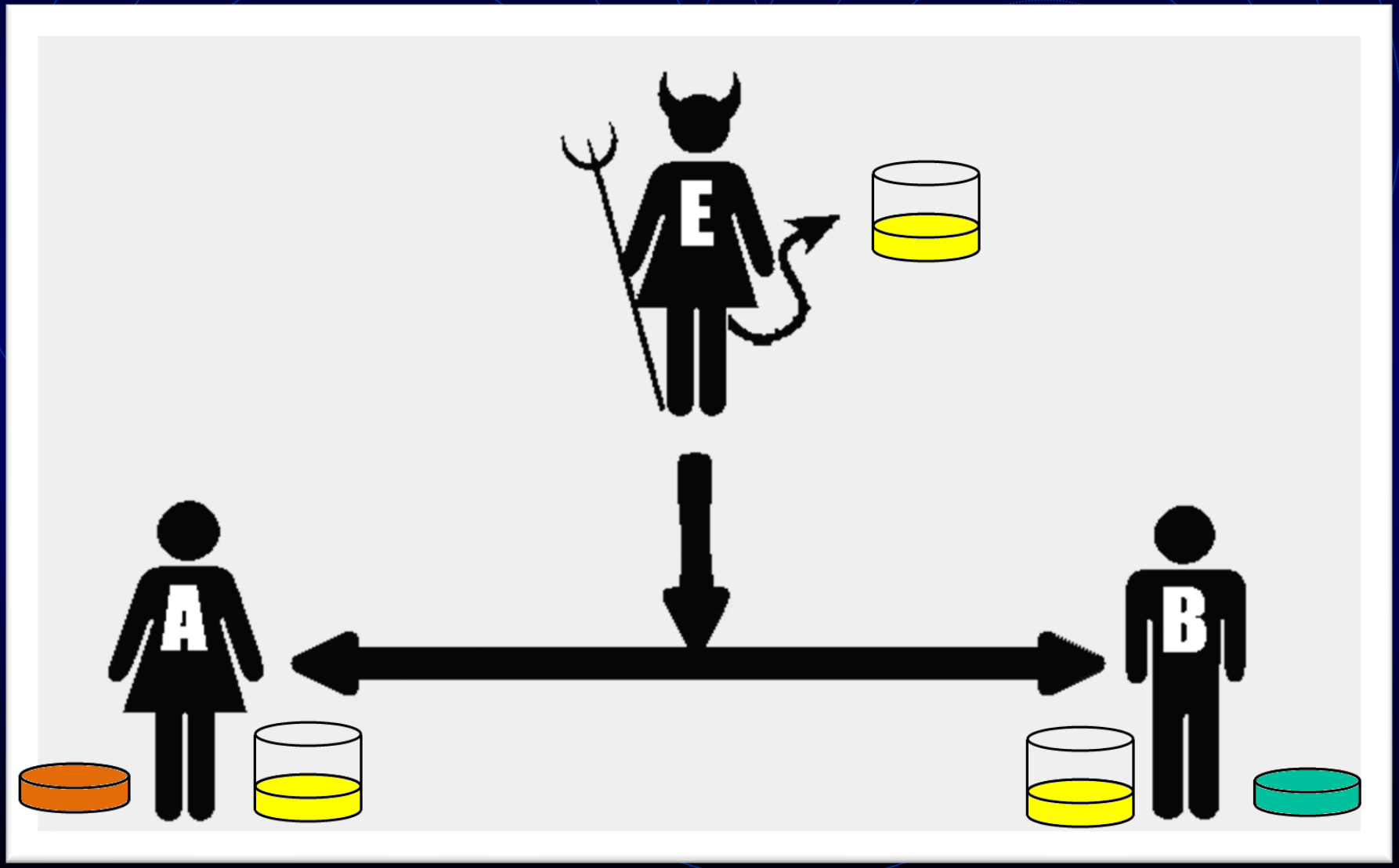
Alice & Bob with Eve listening wish to make a secret shared color



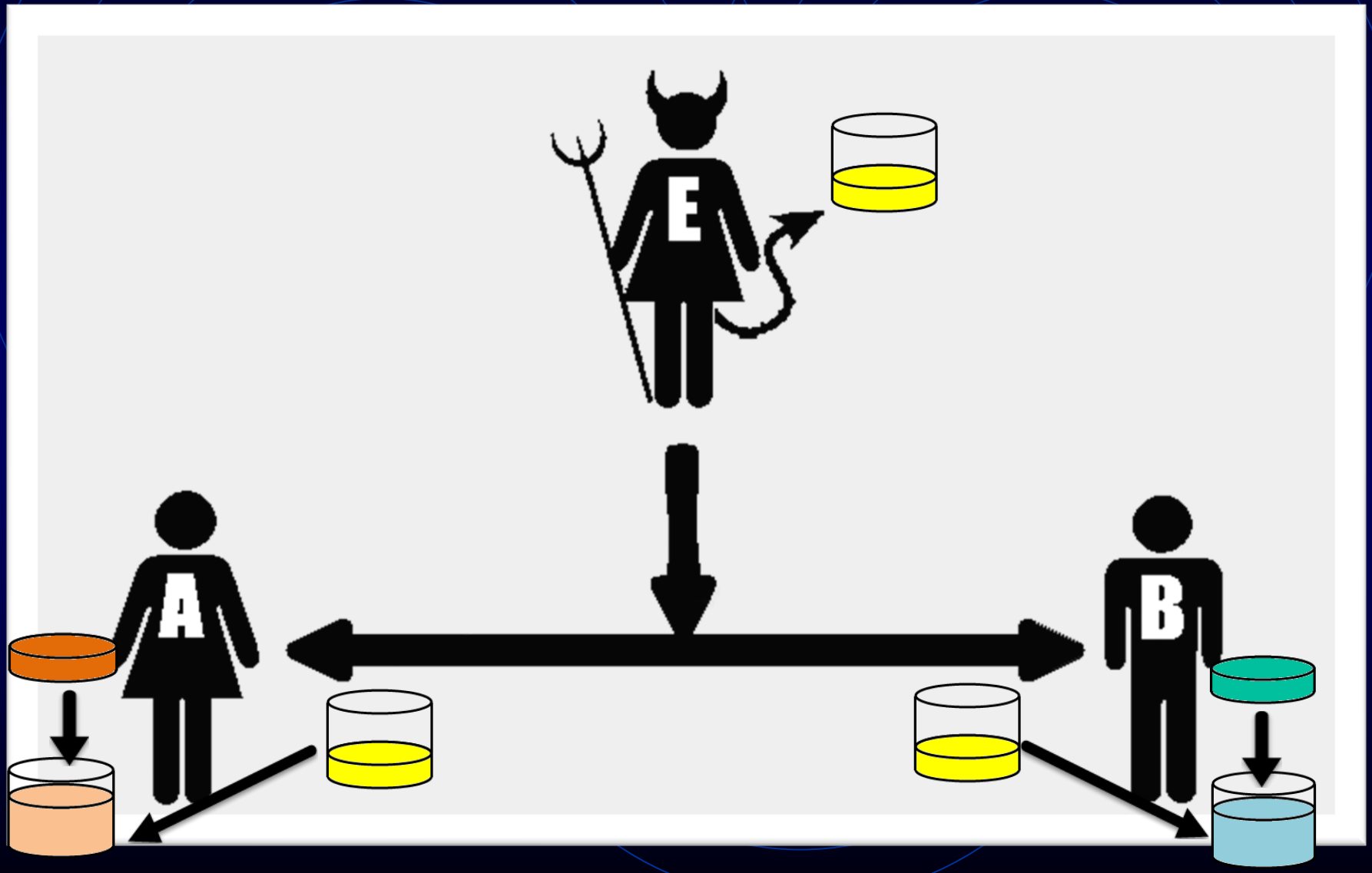
Step 1 - Both publicly agree to a shared color



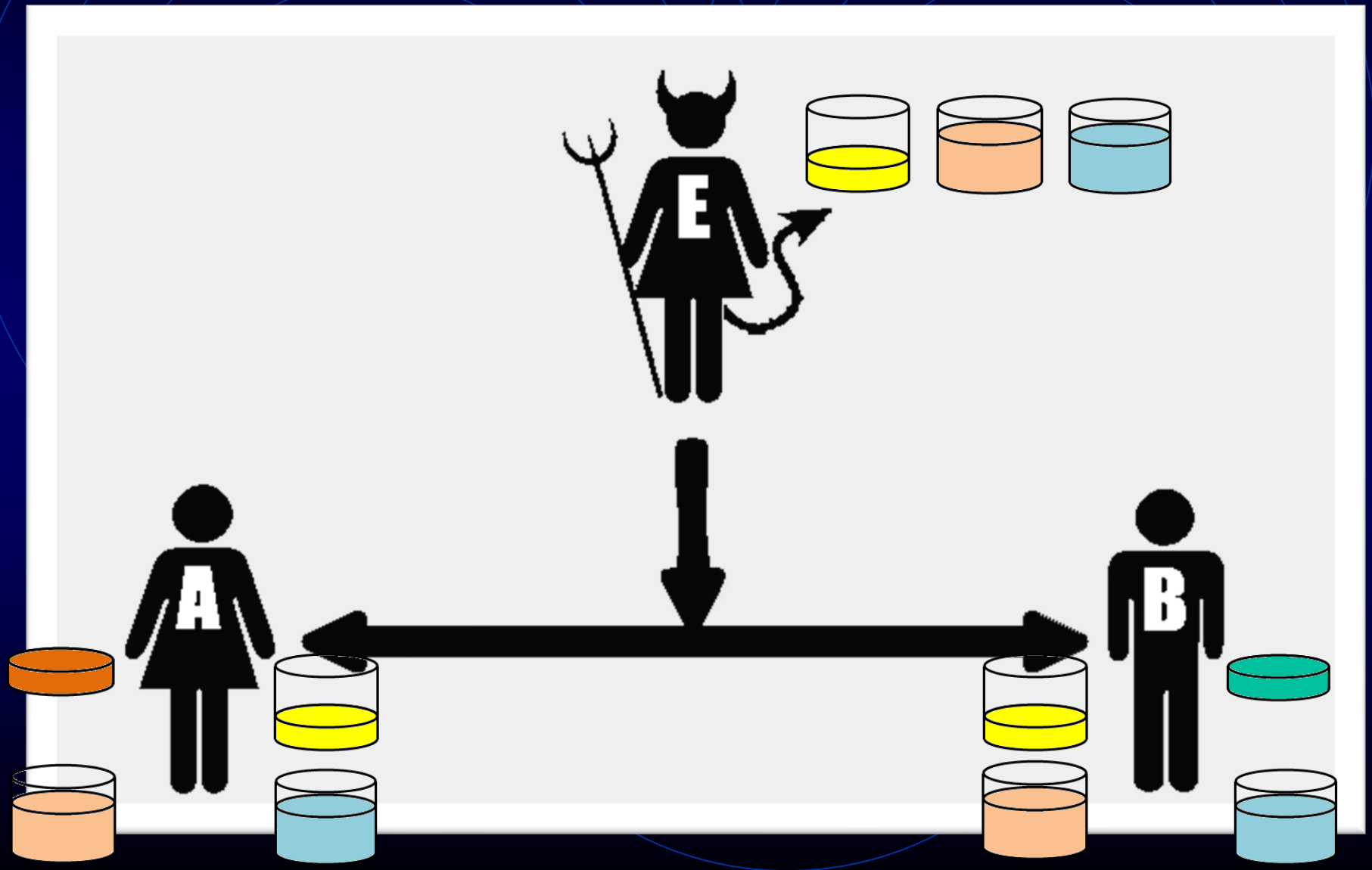
Step 2 - Each picks a secret color



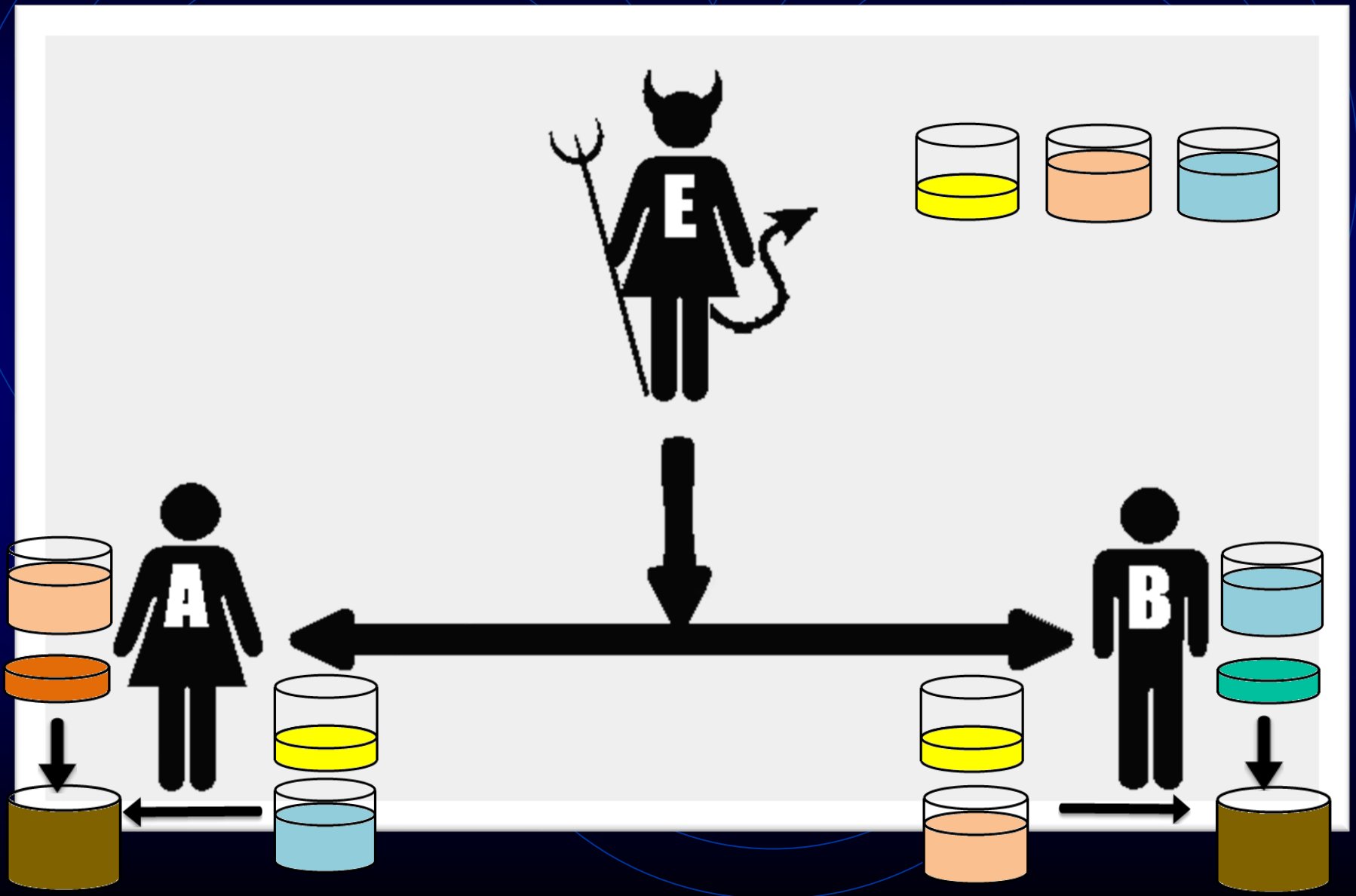
Step 3 - Each adds their secret color to the shared color



Step 4 - Each sends the other their new mixed color



Each combines the shared color from the other with their own secret color



Alice & Bob have agreed to a shared color unknown to Eve

- How is it that Alice & Bob's final mixtures are identical?

- Alice mixed
  - [(Yellow + Green) from Bob] + Orange

- Bob mixed
  - [(Yellow + Orange) from Alice] + Green

Alice & Bob have agreed to a shared color unknown to Eve

- How is it that Alice & Bob's final mixture is secret?
- Eve never has knowledge of the secret colors of either Alice or Bob
- Unmixing a color into its component colors is a hard problem

Let’s get back to math

The Diffie Hellman Key Exchange Algorithm

Global Public Elements

q	prime number
α	$\alpha < q$ and α a primitive root of q

User A Key Generation

Select private X_A	$X_A < q$
Calculate public Y_A	$Y_A = \alpha^{XA} \bmod q$

User B Key Generation

Select private X_B	$X_B < q$
Calculate public Y_B	$Y_B = \alpha^{XB} \bmod q$

Calculation of Secret Key by User A

$$K = (Y_B)^{XA} \bmod q$$

Calculation of Secret Key by User B

$$K = (Y_A)^{XB} \bmod q$$

The Diffie-Hellman Key Exchange Algorithm

Primitive Root

- $q = 11$
- $\alpha = 2$

Range: 1 to q-1

a^1 Mod q,

a^2 Mod q,

……

a^n Mod q

2^1 Mod 11 = 2

2^2 Mod 11 = 4

2^3 Mod 11 = 8

2^4 Mod 11 = 5

2^5 Mod 11 = 10

2^6 Mod 11 = 9

2^7 Mod 11 = 7

2^8 Mod 11 = 3

2^9 Mod 11 = 6

2^{10} Mod 11 = 1

- $q = 11$
- $\alpha = 3$

3^1 Mod 11 = 3

3^2 Mod 11 = 9

3^3 Mod 11 = 5

3^4 Mod 11 = 4

3^5 Mod 11 = 1

3^6 Mod 11 = 3

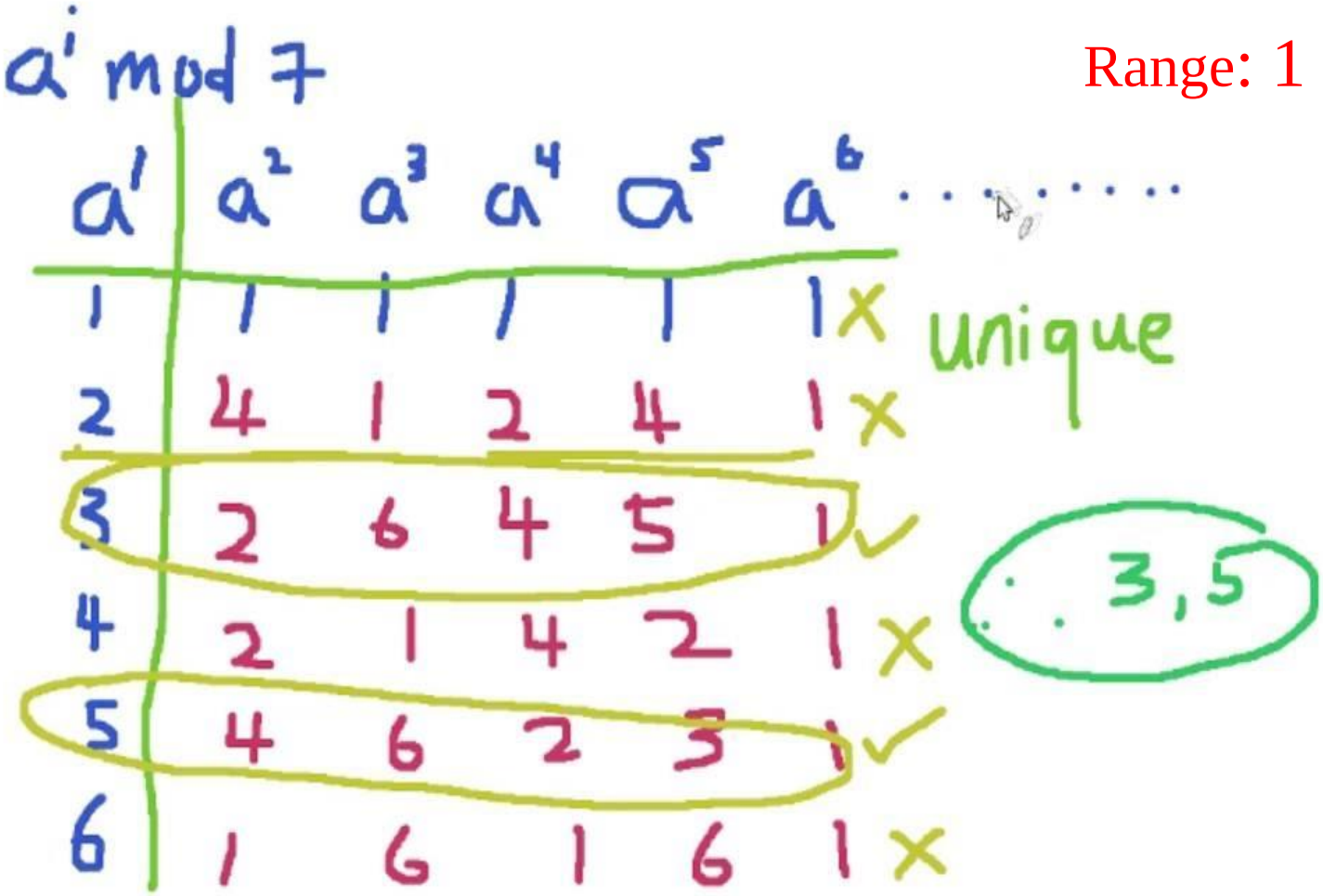
3^7 Mod 11 = 9

3^8 Mod 11 = 5

3^9 Mod 11 = 4

3^{10} Mod 11 = 1

Find Primitive Roots of q =7?



Range: 1 to q-1

Example

User A Key Generation	
Select private X_A	$X_A < q$
Calculate public Y_A	$Y_A = \alpha^{XA} \bmod q$

$X_A = 6$ $X_A < q$ $(q = 11)$

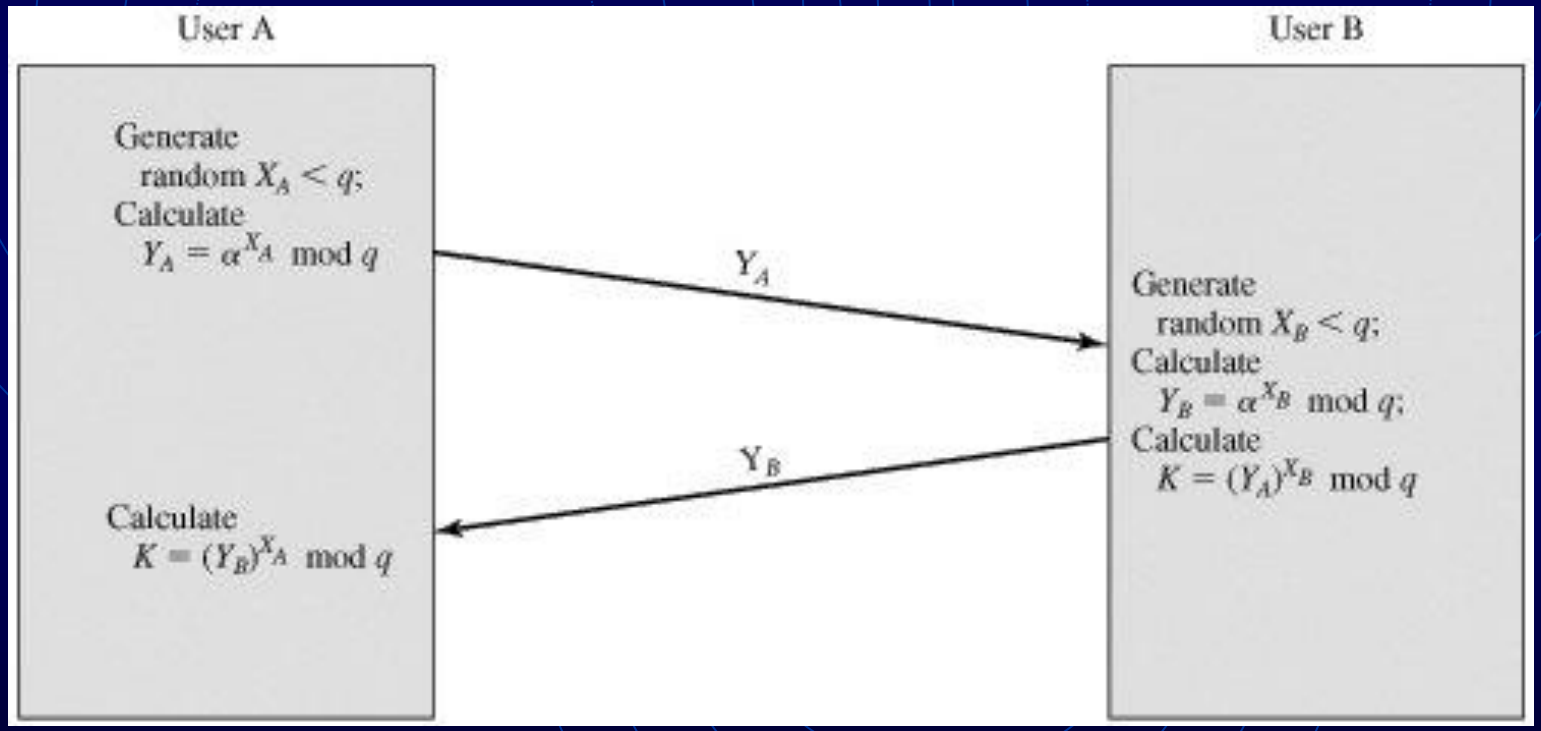
$Y_A = 2^6 \bmod 11 = 9$

User B Key Generation	
Select private X_B	$X_B < q$
Calculate public Y_B	$Y_B = \alpha^{XB} \bmod q$

$X_B = 8$ $X_B < q$ $(q = 11)$

$Y_B = 2^8 \bmod 11 = 3$

The Diffie Hellman Key Exchange Algorithm



Key Exchange

Calculation of Secret Key by User A

$$K = (Y_B)^{XA} \bmod q$$

$K = 3^6 \text{ Mod } 11 = 3$

Calculation of Secret Key by User B

$$K = (Y_A)^{XB} \bmod q$$

$K = 9^3 \text{ Mod } 11 = 3$

$$\begin{aligned} K &= (Y_B)^{X_A} \bmod q \\ &= (\alpha^{X_B} \bmod q)^{X_A} \bmod q \\ &= (\alpha^{X_B})^{X_A} \bmod q \\ &= \alpha^{X_B X_A} \bmod q \\ &= (\alpha^{X_A})^{X_B} \bmod q \\ &= (\alpha^{X_A} \bmod q)^{X_B} \bmod q \\ &= (Y_A)^{X_B} \bmod q \end{aligned}$$

The Diffie Hellman Key Exchange Example

DIFFIE-HELLMAN EXAMPLE

- users Alice & Bob who wish to swap keys:
- agree on prime $q=353$ and $a=3$
- select random secret keys:
  - A chooses $x_A=97$, B chooses $x_B=233$
- compute respective public keys:
  - $y_A=3^{97} \mod 353 = 40$ (Alice)
  - $y_B=3^{233} \mod 353 = 248$ (Bob)
- compute shared session key as:
  - $K_{AB}= y_B^{x_A} \mod 353 = 248^{97} = 160$ (Alice)
  - $K_{AB}= y_A^{x_B} \mod 353 = 40^{233} = 160$ (Bob)

Step 1 –Publicly shared information

- Alice & Bob publicly agree to a large prime number called the modulus, or *p*.
- Alice & Bob publicly agree to a number called the generator, or α, which has a primitive root relationship with *p*.
- In our example we'll assume
  - *p* = 17
  - α = 3
- Eve is aware of the values of *p* or α.

Step 2 – Select a secret key

- Alice selects a secret key, which we will call *a*.
- Bob selects a secret key, which we will call *b*.
- For our example assume:
  - $a = 54$
  - $b = 24$
- Eve is unaware of the values of *a* or *b*.

Step 3 – Combine secret keys with public information

- Alice combines her secret key of *a* with the public information to compute *A*.
  - $A = \alpha^{a} \bmod p$
  - $A = 3^{54} \bmod 17$
  - $A = 15$

Step 3 – Combine secret key with public information

- Bob combines his secret key of *b* with the public information to compute *B*.
  - $B = \alpha^b \bmod p$
  - $B = 3^{24} \bmod 17$
  - $B = 16$

Step 4 – Share combined values

- Alice shares her combined value, *A*, with Bob. Bob shares his combined value, *B*, with Alice.
- Sent to Bob
  - $A = 15$
- Sent to Alice
  - $B = 16$
- Eve is privy to this exchange and knows the values of *A* and *B*

Step 5 – Compute Shared Key

- Alice computes the shared key.
- Bob computes the shared key.

Alice & Bob have a shared encryption key, unknown to Eve

- Alice & Bob have created a shared secret key, s, unknown to Eve.

- The shared secret key can now be used to encrypt & decrypt messages by both parties.