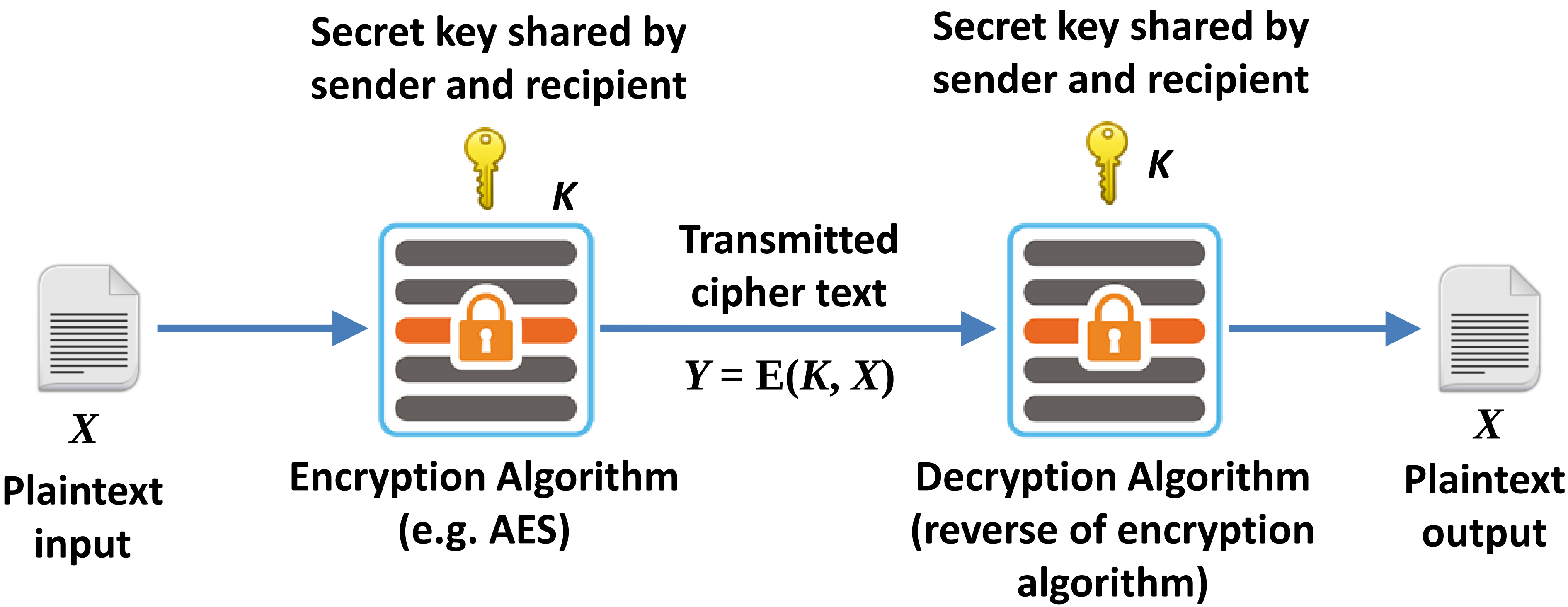
Asymmetric Ciphers

Outline

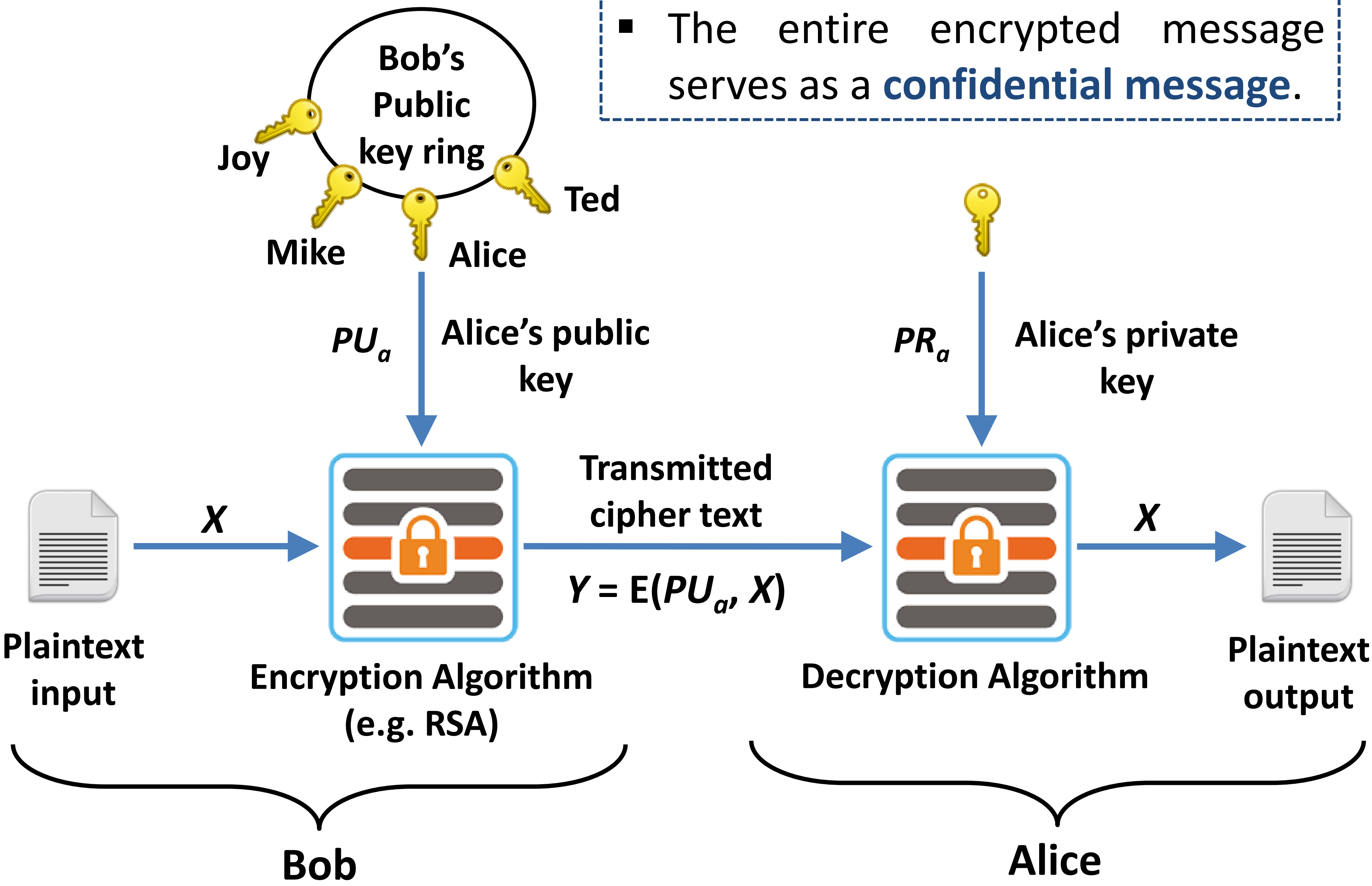
- Public Key Cryptosystems with Applications
- Requirements and Cryptanalysis
- RSA algorithm
- RSA computational aspects and security
- Diffie-Hillman Key Exchange algorithm
- Man-in-Middle attack

Symmetric Key Encryption



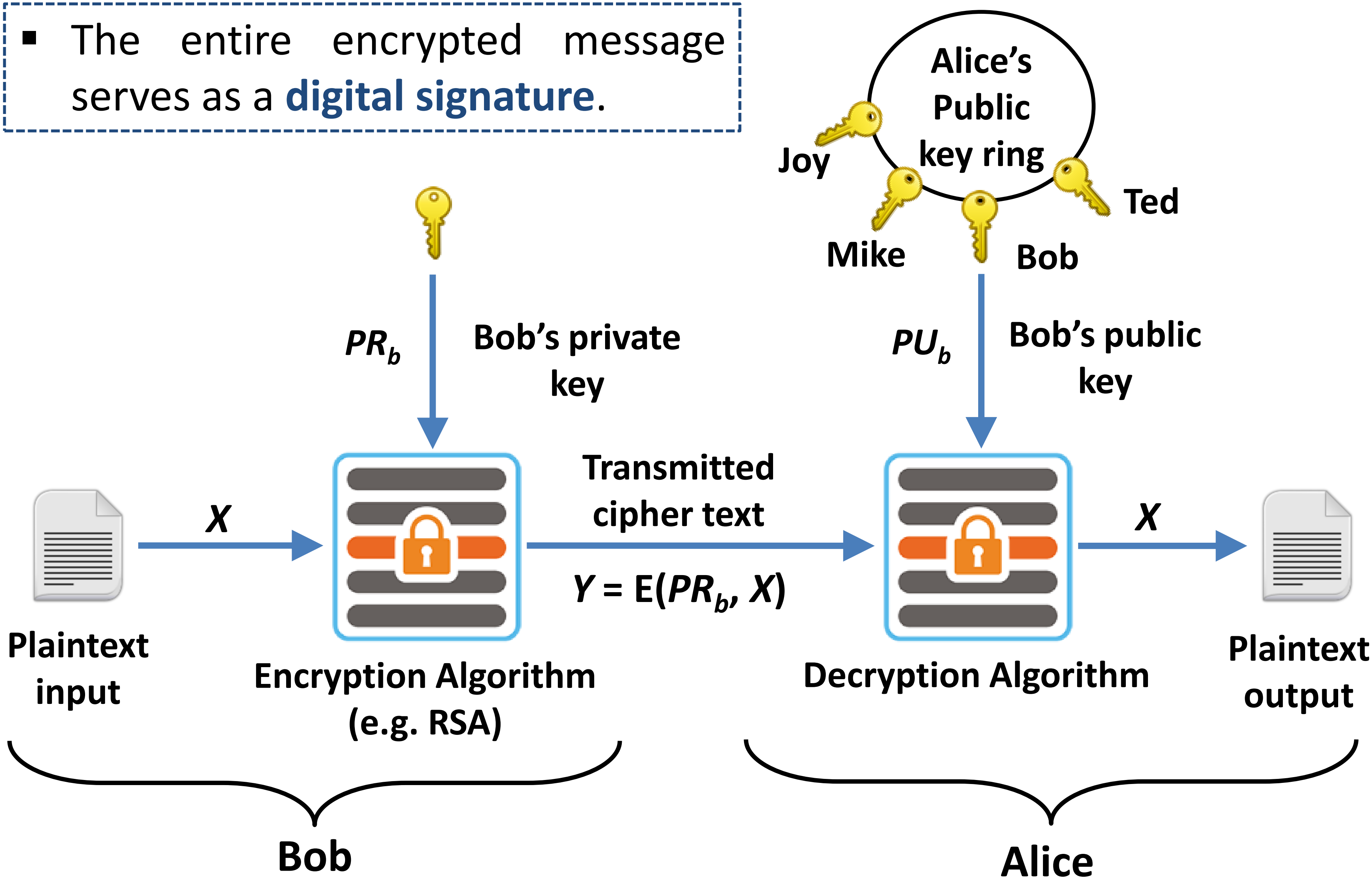
Asymmetric Key Encryption with Public Key

- The entire encrypted message serves as a **confidential message**.

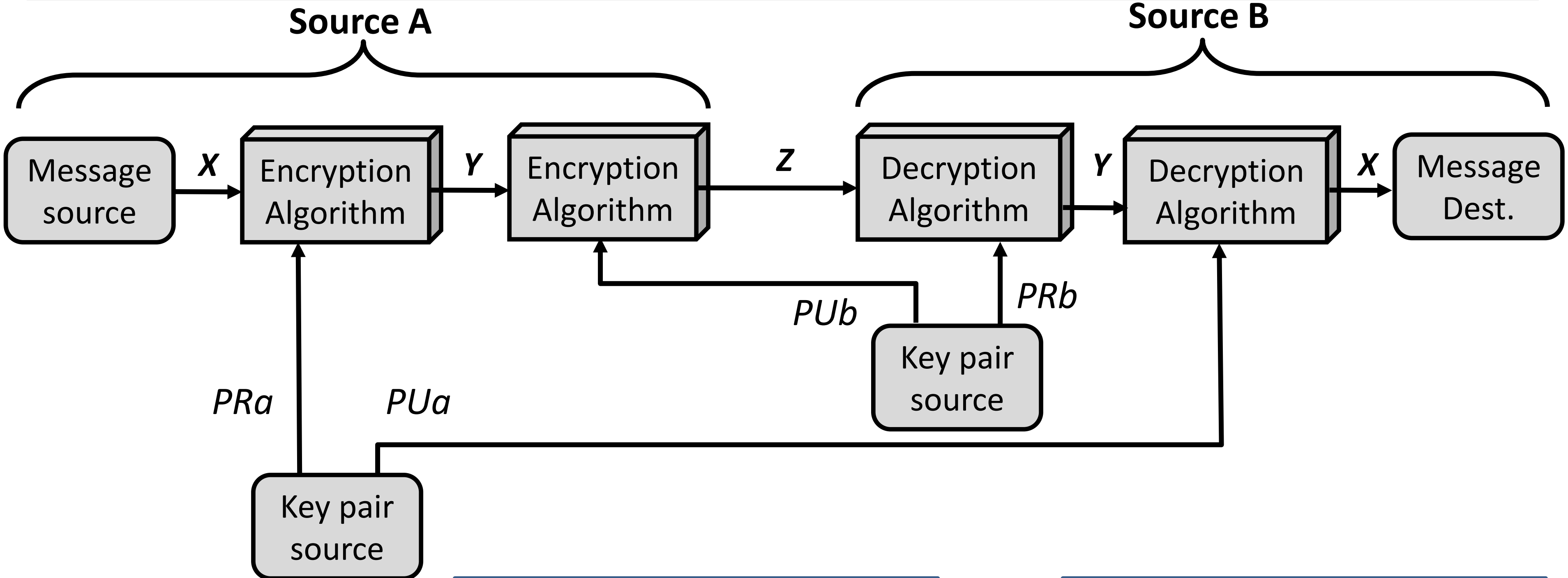


Asymmetric key Encryption with Private Key

- The entire encrypted message serves as a **digital signature**.



Authentication and Confidentiality



$Z = \mathrm{E}(PU_b, \mathrm{E}(PR_a, X))$

$X = \mathrm{D}(PU_a, \mathrm{D}(PR_B, Z))$

Applications for Public-Key Cryptosystems

- **Encryption/decryption:** The sender encrypts a message with the recipient's public key.
- **Digital signature:** The sender "signs" a message with its private key. Signing is achieved by a cryptographic algorithm applied to the message or to a small block of data that is a function of the message.
- **Key exchange:** Two sides cooperate to exchange a session key. Several different approaches are possible, involving the private key(s) of one or both parties. E.g. Diffie–Hellman key exchange scheme

RSA Algorithm

- **RSA** is a block cipher in which the Plaintext and Ciphertext are represented as integers between **0** and **n-1** for some n.
- Large messages can be broken up into a number of blocks.
- Each block would then be represented by an integer.

Step-1: Generate Public key and Private key

Step-2: Encrypt message using Public key

Step-3: Decrypt message using Private key

Step-1: Generate Public key and Private key

- Select two large prime numbers: **p** and **q**
- Calculate modulus : $\mathbf{n = p * q}$
- Calculate Euler's totient function : $\boldsymbol{\phi(n) = (p-1) * (q-1)}$
- Select **e** such that **e** is **relatively prime** to $\boldsymbol{\phi(n)}$ and $\boldsymbol{1 < e < \phi(n)}$

Two numbers are relatively prime if they have no common factors other than 1.

- Determine **d** such that $\boldsymbol{d * e \equiv 1 \pmod{\phi(n)}}$
- Publickey : $\mathbf{PU = \{ e, n \}}$
- Privatekey : $\mathbf{PR = \{ d, n \}}$

Step-1: Generate Public key and Private key

- Select two large prime numbers: **$p = 3$** and **$q = 11$**
- Calculate modulus : **$n = p * q$, $n = 33$**
- Calculate Euler's totient function : **$\phi(n) = (p-1) * (q-1)$**

  $\phi(n) = (3 - 1) * (11 - 1) = 20$
- Select **e** such that **e** is **relatively prime** to **$\phi(n)$** and **$1 < e < \phi(n)$**
- We have several choices for **e : 7, 11, 13, 17, 19** Let's take **$e = 7$**
- Determine **d** such that **$d * e \equiv 1 \pmod{\phi(n)}$**
- **$? * 7 \equiv 1 \pmod{20}$, $3 * 7 \equiv 1 \pmod{20}$**

- Public key : **PU = { e, n } , PU = { 7, 33 }**
- Private key : **PR = { d, n }, PR = { 3, 33 }**

- This is equivalent to finding d which satisfies $de = 1 + j.\phi(n)$ where j is any integer.
- We can rewrite this as $d = (1 + j.\ \phi(n)) / e$

*Find Modular Multiplicative Inverse using Extended Euclidean algorithm

Step-2 : Encrypt Message

- Encryption Using Public key: $C = M^e \bmod n$

Ciphertext

Input Message

Publickey

PU = { e, n } , PU = { 7, 33 }

For message M = 14

$C = 14^7 \bmod 33$

$C = [(14^1 \bmod 33) \text{ X } (14^2 \bmod 33) \text{ X } (14^4 \bmod 33)] \bmod 33$

$C = (14 \text{ X } 31 \text{ X } 4) \bmod 33 = 1736 \bmod 33$

$C = 20$

Step-3 : Decrypt Message

- Encryption Using Public key: $M = C^d \bmod n$

Plaintext Message

Cipher Message

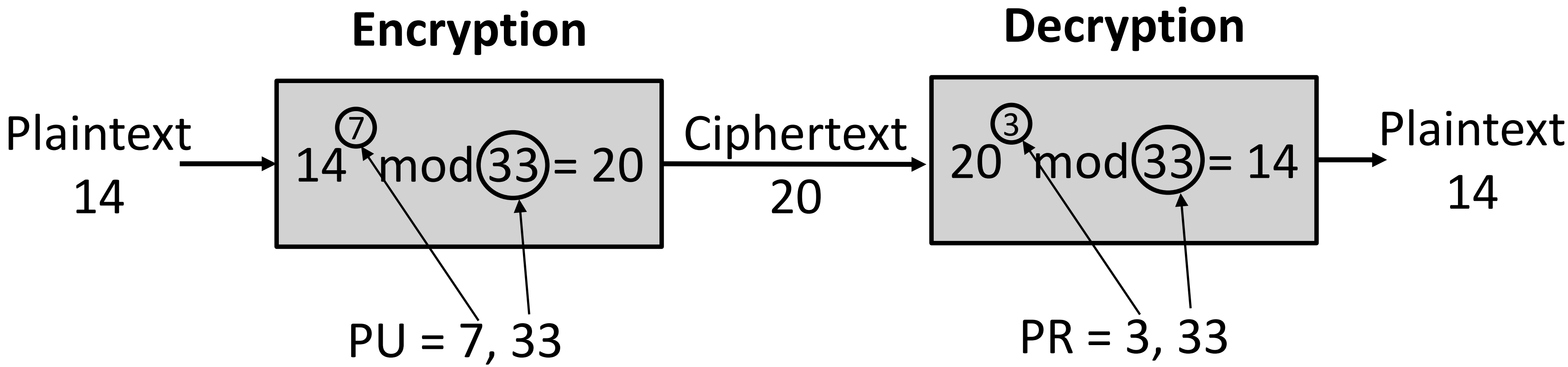
Privatekey

PR = { d, n } , PR = { 3, 33 }

For Ciphertext C = 20

$M = 20^3 \bmod 33$
$M = [(20^1 \bmod 33) \text{ X } (20^2 \bmod 33)] \bmod 33$
$M = (20 \text{ X } 4) \bmod 33 = 80 \bmod 33$
$M = 14$

Example RSA Algorithm



RSA Example

- Find n, $\phi(n)$, e, d for p=7 and q= 19 then demonstrate encryption and decryption for M = 6

$n = p * q = 7 * 19 = 133$

$\phi(n) = (p - 1) * (q - 1) = 108$

Finding e relatively prime to 108

e = 2 => GCD(2, 108) = 2 (no)

e = 3 => GCD(3, 108) = 3 (no)

e = 5 => GCD(5, 108) = 1 (Yes)

- Finding d such that (d * e) mod $\phi(n)$ = 1
- We can rewrite this as d = (1 + j . $\phi(n)$) / e

j = 0 => d = 1 / 5 = 0.2 ← integer ? (no)

j = 1 => d = 109 / 5 = 21.8 ← integer ? (no)

j = 2 => d = 217 / 5 = 43.4 ← integer ? (no)

j = 3 => d = 325 / 5 = 65 integer ? (yes)

*OR Find Modular Multiplicative Inverse using Extended Euclidean algorithm

Public key :

PU = { e, n } = {5, 133}

Private key :

PR = { d, n } = {65, 133}

RSA Example – cont…

- Encryption:

  $C = M^e \bmod n$

  $PU = \{ e, n \}$, $PU = \{ 5, 133 \}$

  For message $M = 6$

  $C = 6^5 \bmod 133$

  $C = 7776 \bmod 33$

  $C = 62$

- Decryption:

  $M = C^d \bmod n$

  $PR = \{ d, n \}$, $PU = \{ 65, 133 \}$

  For $C = 62$

  $M = 62^{65} \bmod 133$

  $M = 2666 \bmod 33$

  $M = 6$

RSA Example

- P and Q are two prime numbers. P=7, and Q=17. Take public key E=5. If plain text value is 10, then what will be cipher text value according to RSA algorithm?
- $n = 119$
- $\phi(n) = 96$
- $e = 5$
- $d = 77$
- $PU = \{ 5, 119 \}$
- $PR = \{77, 119\}$
- $C = 10^5 \bmod 119 \Rightarrow C = 40$

RSA Security

- possible approaches to attacking RSA are:
  - brute force key search - infeasible given size of numbers
  - mathematical attacks - based on difficulty of computing ø(n), by factoring modulus n
  - timing attacks - on running of decryption
  - chosen ciphertext attacks - given properties of RSA

Mathematical Attack

- mathematical approach takes 3 forms:
  - factor $n=p.q$, hence compute $\varnothing(n)$ and then d
  - determine $\varnothing(n)$ directly and compute d
  - find d directly
- currently assume 1024-2048 bit RSA is secure

Timing Attacks

- Developed by Paul Kocher in mid-1990's
- Exploit timing variations in operations
  - E.g. multiplying by small vs large number
- Infer operand size based on time taken
- Infer time taken in exponentiation
- Countermeasures
  - use constant exponentiation time
  - add random delays
  - blind values used in calculations

Chosen Ciphertext Attacks

- RSA is vulnerable to a Chosen Ciphertext Attack (CCA)
- Attackers can choose ciphertexts & get decrypted plaintext back
- Countermeasure with random pad of plaintext

Primitive root

- Let $\boldsymbol{p}$ be a prime number
- Then $\boldsymbol{a}$ is a primitive root for $\boldsymbol{p}$, if the powers of $\boldsymbol{a}$ modulo $\boldsymbol{p}$ generates all integers from **1 to $\boldsymbol{p-1}$** in some permutation.

$$a \bmod p, a^2 \bmod p, \dots, a^{p-1} \bmod p$$

- Example: p = 7 then primitive root is 3 because powers of 3 mod 7 generates all the integers from 1 to 6

$$\begin{aligned}
3^1 &= 3 \equiv 3 \pmod 7 \\
3^2 &= 9 \equiv 2 \pmod 7 \\
3^3 &= 27 \equiv 6 \pmod 7 \\
3^4 &= 81 \equiv 4 \pmod 7 \\
3^5 &= 243 \equiv 5 \pmod 7 \\
3^6 &= 729 \equiv 1 \pmod 7
\end{aligned}$$

Discrete Logarithm

- For any integer $\boldsymbol{b}$ and a primitive root $\boldsymbol{a}$ of prime number $\boldsymbol{p}$, we can find a unique exponent $\boldsymbol{i}$ such that

$$b = a^i (mod\ p) \ \ where\ 0 \leq i \leq (p-1)$$

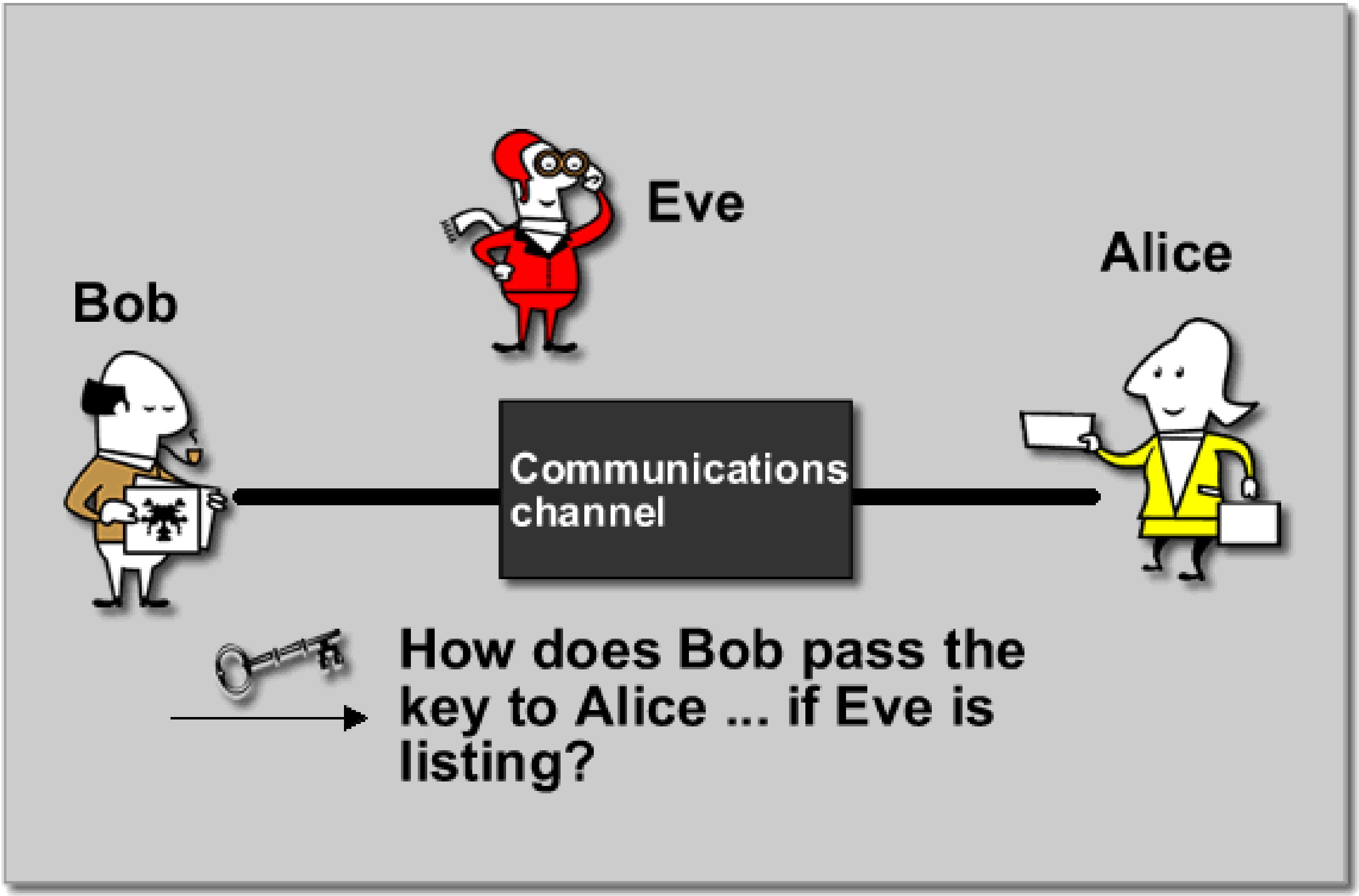
- The exponent $\boldsymbol{i}$ is referred as the discrete logarithm of $\boldsymbol{b}$ for the base $\boldsymbol{a}$, mod $\boldsymbol{p}$. It expressed as below.

$$\mathrm{dlog}_{a,p}(b)$$

Key Establishment Problem

- Securing communication requires that the data is encrypted before being transmitted
- Associated with encryption and decryption are keys that must be shared by the participants.
- The problem of securing the data then becomes the problem of securing the key establishment
- Task: If the participants do not physically meet, then how do the participants establish a shared key?

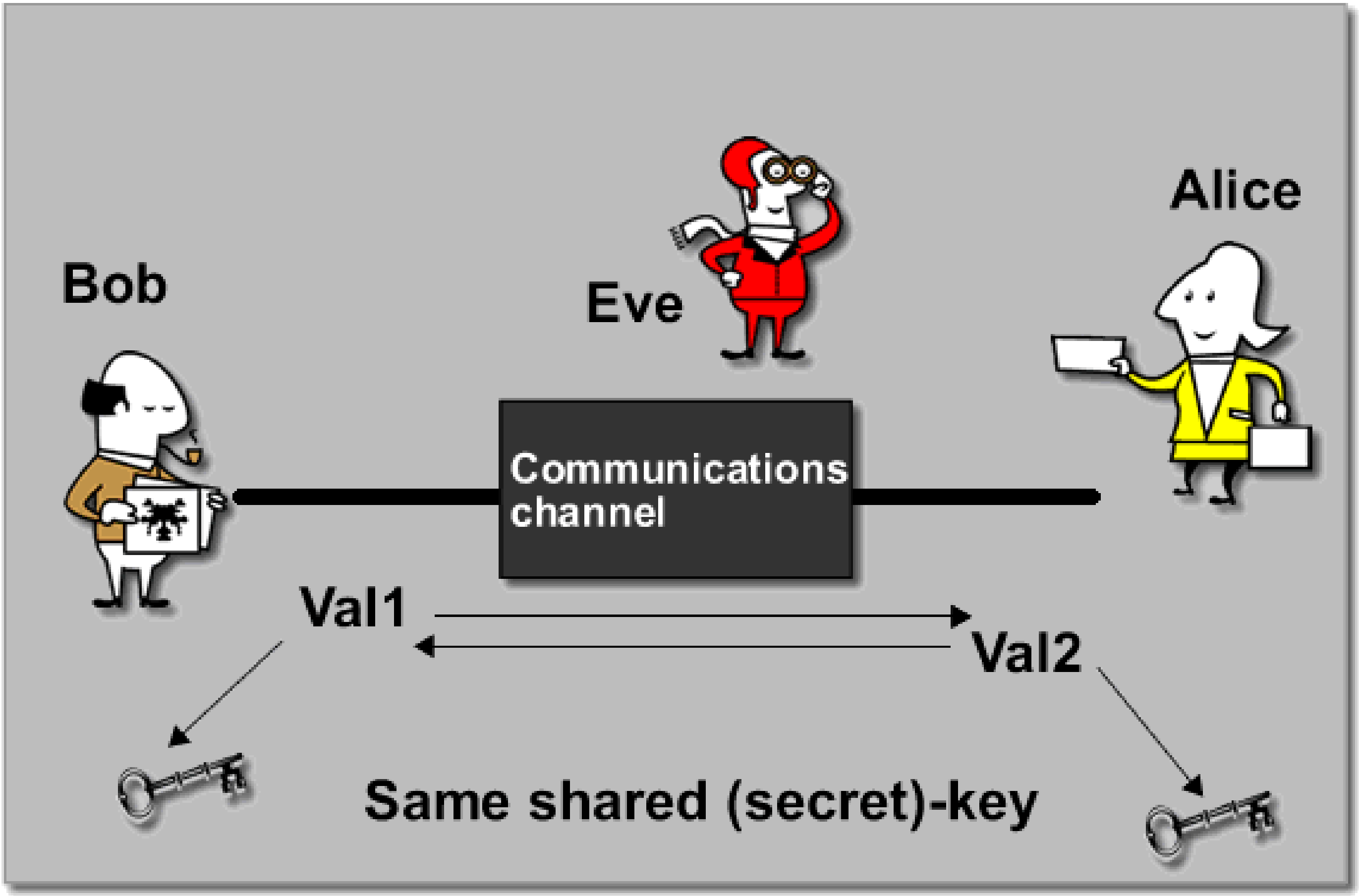
Key Establishment Problem (cont.)



Diffie-Hellman Key Agreement

- Discovered by Whitfield Diffie and Martin Hellman
- Diffie-Hellman key agreement protocol
  - Exponential key agreement
  - Allows two users to exchange a secret key
  - Requires no prior secrets
  - Real-time over an **untrusted** network

Diffie-Hellman Key Agreement (cont.)



Diffie-Hellman Algorithm

- Requires two large numbers, one prime p, and generator g ($2 <= g <= p-2$), a primitive root of p, (p and g are both publicly available numbers).
- Users pick random private values x ($x < p$) and y ($y < p$)
- Compute public values (keys)
  - $R1 = g^x \bmod p$
  - $R2 = g^y \bmod p$
- Keys R1 and R2 are exchanged
- Compute shared, private key
  - $k_{alice} = (R2)^x \bmod p$
  - $k_{bob} = (R1)^y \bmod p$
- Algebraically it can be shown that $k_{alice} = k_{bob}$
  - Users now have a symmetric secret key to encrypt

Proof

- We know

  $R1 = g^x \bmod p$

  $R2 = g^y \bmod p$

- $k_{alice} = (R2)^x \bmod p$

  $= (g^y \bmod p)^x \bmod p$

  $= (g^y)^x \bmod p$

  $= (g)^{yx} \bmod p$

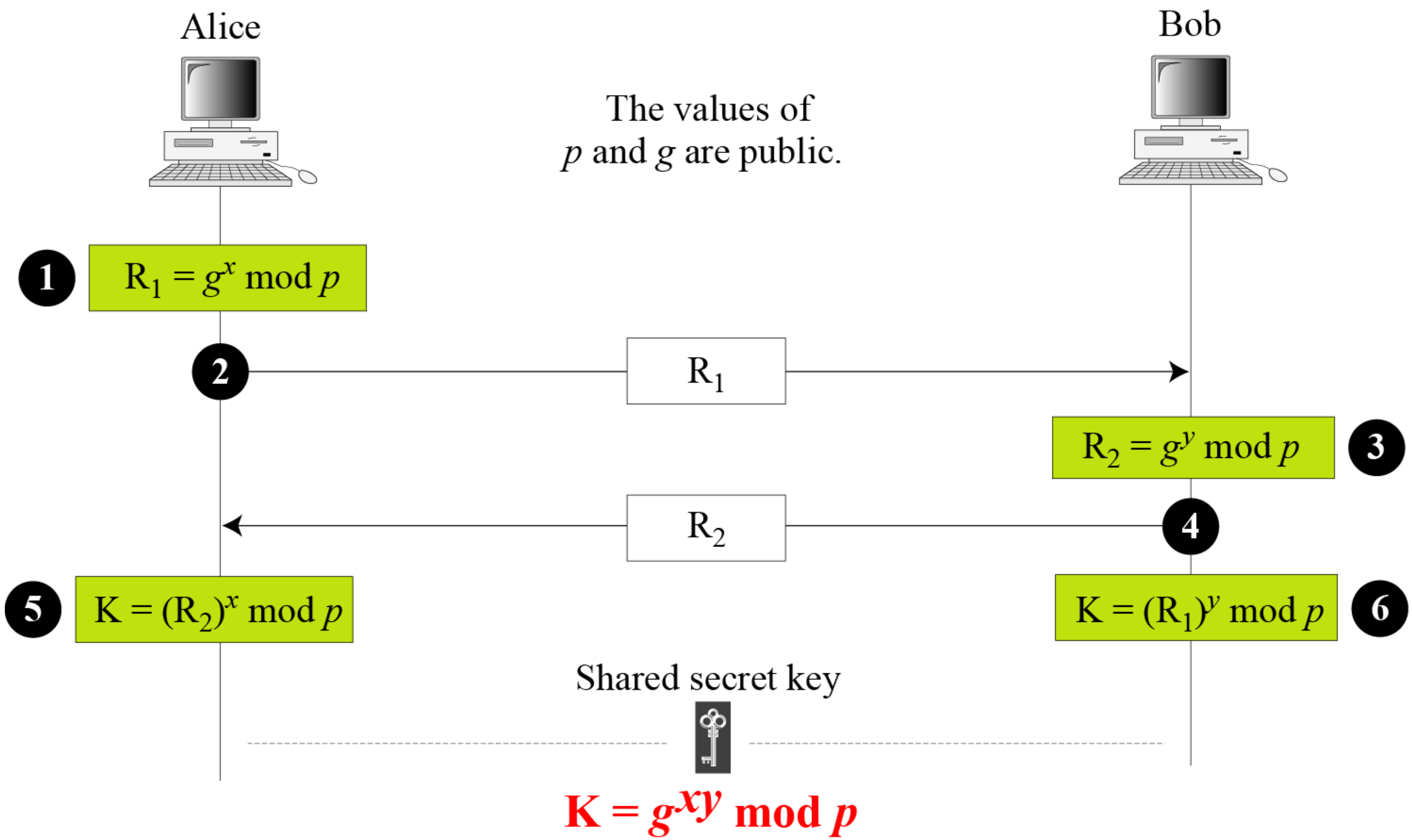
  $= (g^x)^y \bmod p$

  $= (g^x \bmod p)^y \bmod p$

  $= (R1)^y \bmod p$

  $= k_{bob}$

Key Exchange



Example

- Alice and Bob get public numbers
  - P = 19, G = 3 [Primitive roots of modulus 19 are 2,3,10,13,14,15]
- Alice and Bob pick private values x=15 & y=10 respectively
- Alice and Bob compute public values
  - R1 = 3^{15} mod 19 = 12
  - R2 = 3^{10} mod 19 = 16
  - Alice and Bob exchange public numbers
- Alice and Bob compute symmetric keys
  - kalice = $(R2)^x$ mod p = $(16)^{15}$ mod 19 = 7
  - kbob = $(R1)^y$ mod p = $(12)^{10}$ mod 19 = 7
- Alice and Bob now can talk securely!

Example: P=23 and G=5 ?

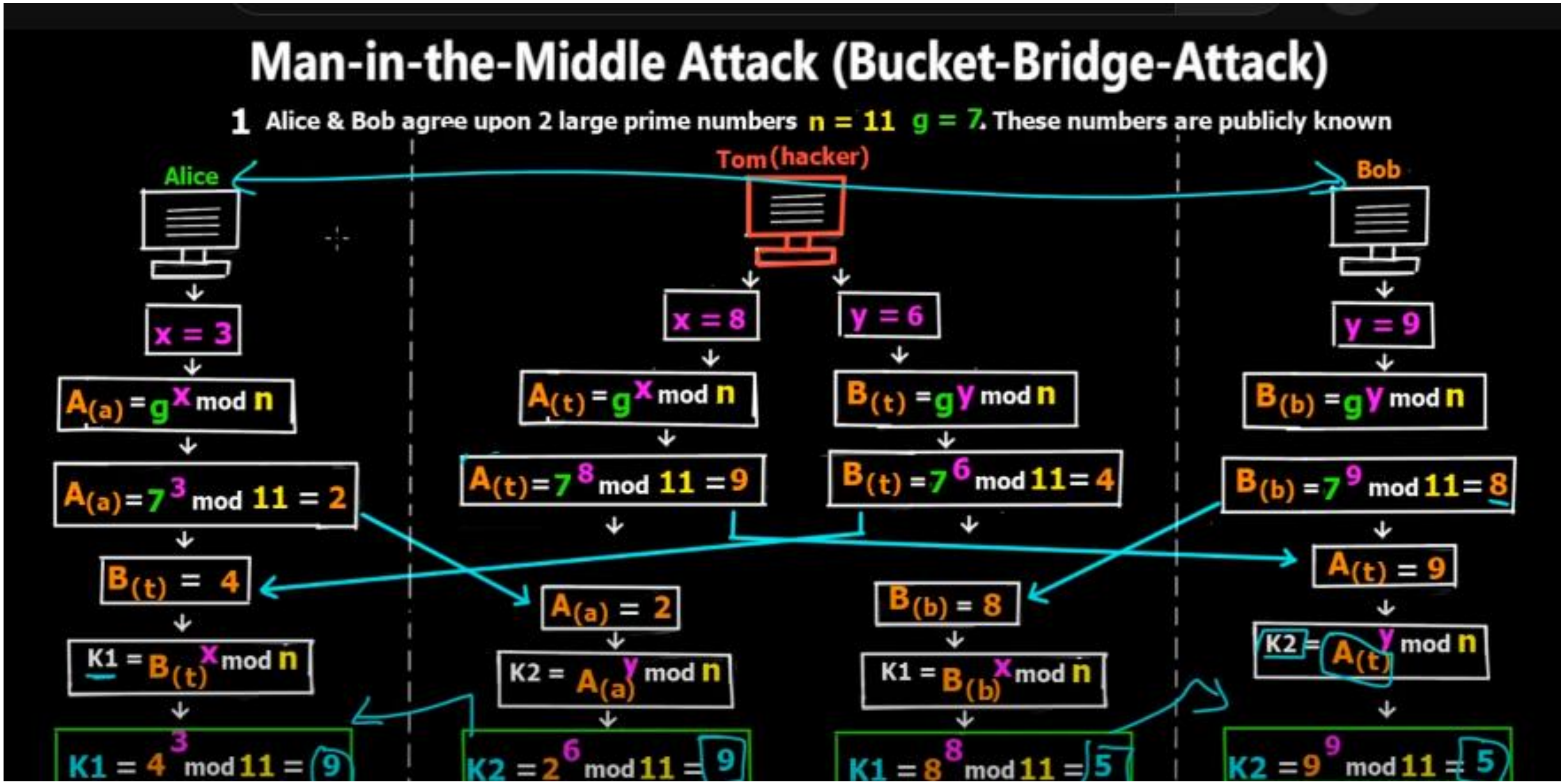
Security of Diffie-Hellamn

- This protocol susceptible to two attacks:
  - The Man-in-the-middle attack
  - The Discrete logarithmic attack

Man in the middle attack

- Suppose Alice and Bob wish to exchange keys, and Darth is the adversary.

1. Darth prepares for the attack by generating two random private keys X_{D1} and X_{D2} and then computes corresponding public keys Y_{D1} and Y_{D2}.
2. Alice transmits Y_A to Bob.
3. Darth intercepts Y_A and transmits Y_{D1} to Bob. Darth also calculates $K_2=(Y_A)^{XD2} \bmod q$.
4. Bob receives Y_{D1} and calculates $K_1 = (Y_{D1})^{XB} \bmod q$.
5. Bob transmits Y_B to Alice.
6. Darth intercepts Y_B and transmits Y_{D2} to Alice. Darth calculates $K_1=(Y_B)^{XD1} \bmod q$.
7. Alice receives Y_{D2} and calculates $K_2 = (Y_{D2})^{XA} \bmod q$.


Man-in-the-Middle Attack (Bucket-Bridge-Attack)
1 Alice & Bob agree upon 2 large prime numbers n = 11 g = 7. These numbers are publicly known
Alice
Tom (hacker)
Bob
x = 3
x = 8
y = 6
y = 9
A(a) = g x mod n
A(t) = g x mod n
B(t) = g y mod n
B(b) = g y mod n
A(a) = 7 3 mod 11 = 2
A(t) = 7 8 mod 11 = 9
B(t) = 7 6 mod 11 = 4
B(b) = 7 9 mod 11 = 8
B(t) = 4
A(a) = 2
B(b) = 8
A(t) = 9
K1 = B(t) x mod n
K2 = A(a) y mod n
K1 = B(b) x mod n
K2 = A(t) y mod n
K1 = 4 3 mod 11 = 9
K2 = 2 6 mod 11 = 9
K1 = 8 8 mod 11 = 5
K2 = 9 9 mod 11 = 5

ElGamal Encryption Algorithm

- **ElGamal encryption** uses asymmetric key encryption for communicating between two parties and encrypting the message. It is based on the Diffie–Hellman key exchange.
- This cryptosystem is based on the difficulty of finding **discrete logarithm** in a cyclic group that is even if we know g^a and g^k, it is extremely difficult to compute g^{ak}.
- ElGamal is generally used to encrypt <u>only the symmetric key</u> (Not the plaintext). This is because asymmetric cryptosystems like ElGamal are usually slower than symmetric ones

ElGamal Encryption

- **Keys & parameters**
  - Domain parameter = {p, g}
  - Choose $x \in [1, p-1]$ and compute $y = g^x \bmod p$
  - Public key (p, g, y)
  - Private key x

- **Encryption**: $m \rightarrow (C_1, C_2)$
  - Pick a random integer $k \in [1, p-1]$
  - Compute $C_1 = g^k \bmod p$
  - Compute $C_2 = m \times y^k \bmod p$

- **Decryption**
  - $m = C_2 \times C_1^{-x} \bmod p$
  - $C_2 \times C_1^{-x} = (m \times y^k) \times (g^k)^{-x} = m \times (g^x)^k \times (g^k)^{-x} = m \bmod p$

ElGamal Example

- Keys & parameters
  - Let prime number p=23 and generator g=7
  - Choose x=9 and $y= g^x \bmod p= 7^9 \bmod 23=15$
  - Publick key: {23, 7, 15}
  - Private key=9
- Encryption for m=20
  - Choose random k=3
  - $C1= 7^3 \bmod 23=21$
  - $C2=20 \times 15^3 \bmod 23=20 \times 17 \bmod 23=18$
  - Send (C1,C2)=(21,18) as a ciphertext
- Decryption
  - $M=18 \times 21^{-9} \bmod 23 = 20$