COMPSCI4062&5063: Cyber Security Fundamentals

Topic 5: Cryptography II

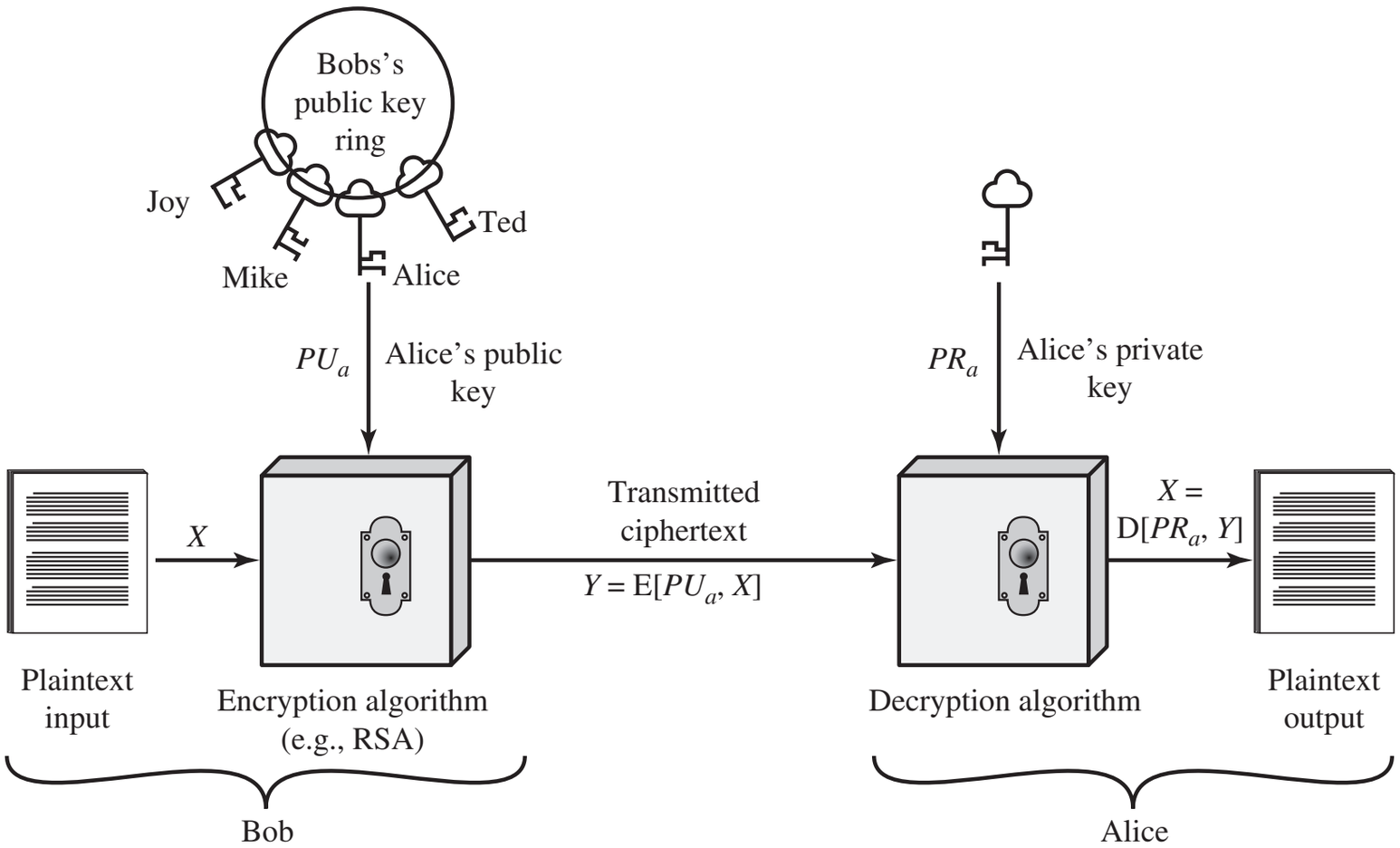
Dr. Dongzhu Liu

Email: dongzhu.liu@glasgow.ac.uk
Office: SAWB 510 (b)

Overview

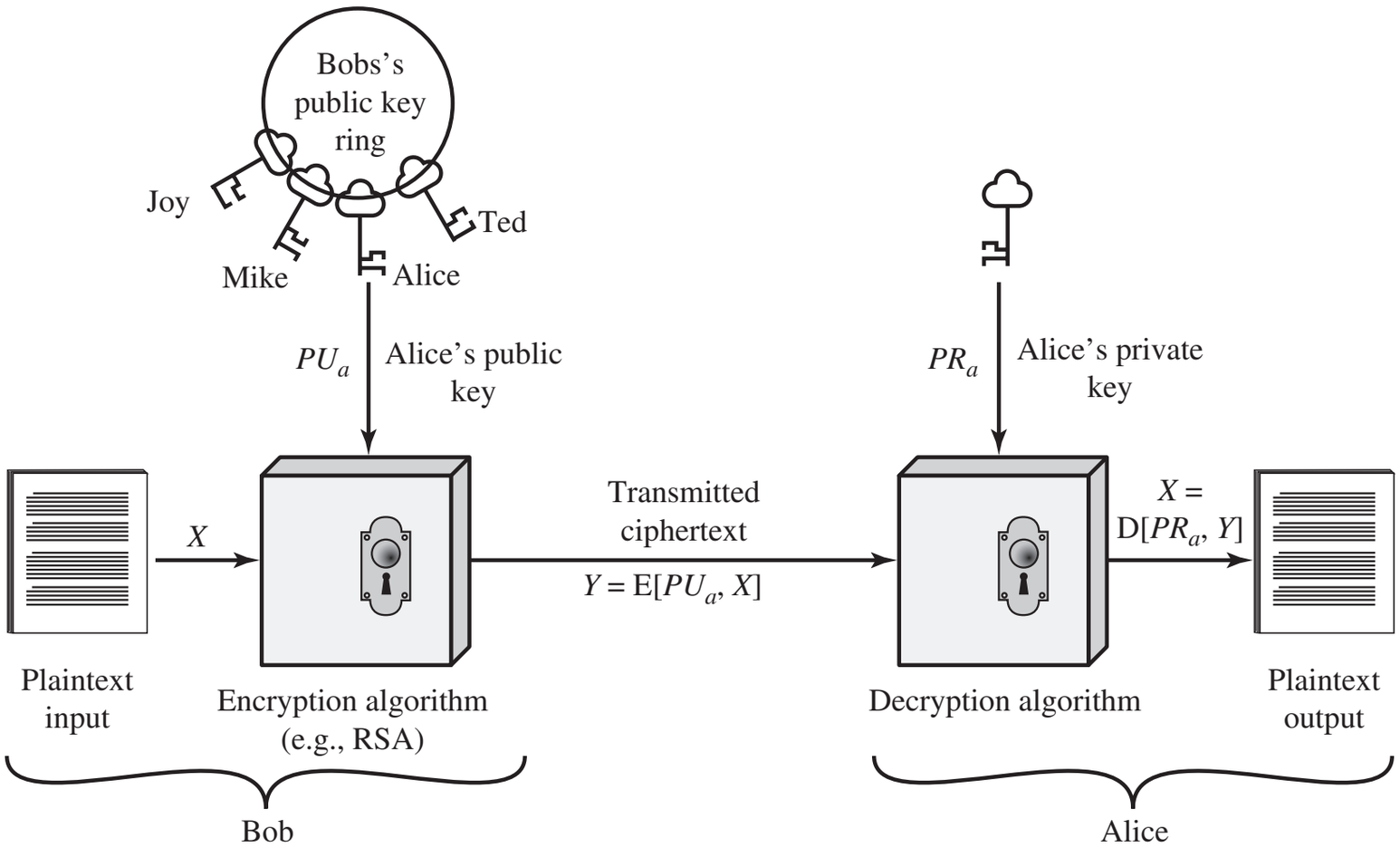
- Asymmetric Encryption
  - Confidentiality vs. Authentication
  - Requirements
- Algorithms
  - RSA
  - Diffie-Hellman Key Exchange
- Tutorial Questions

Asymmetric Encryption



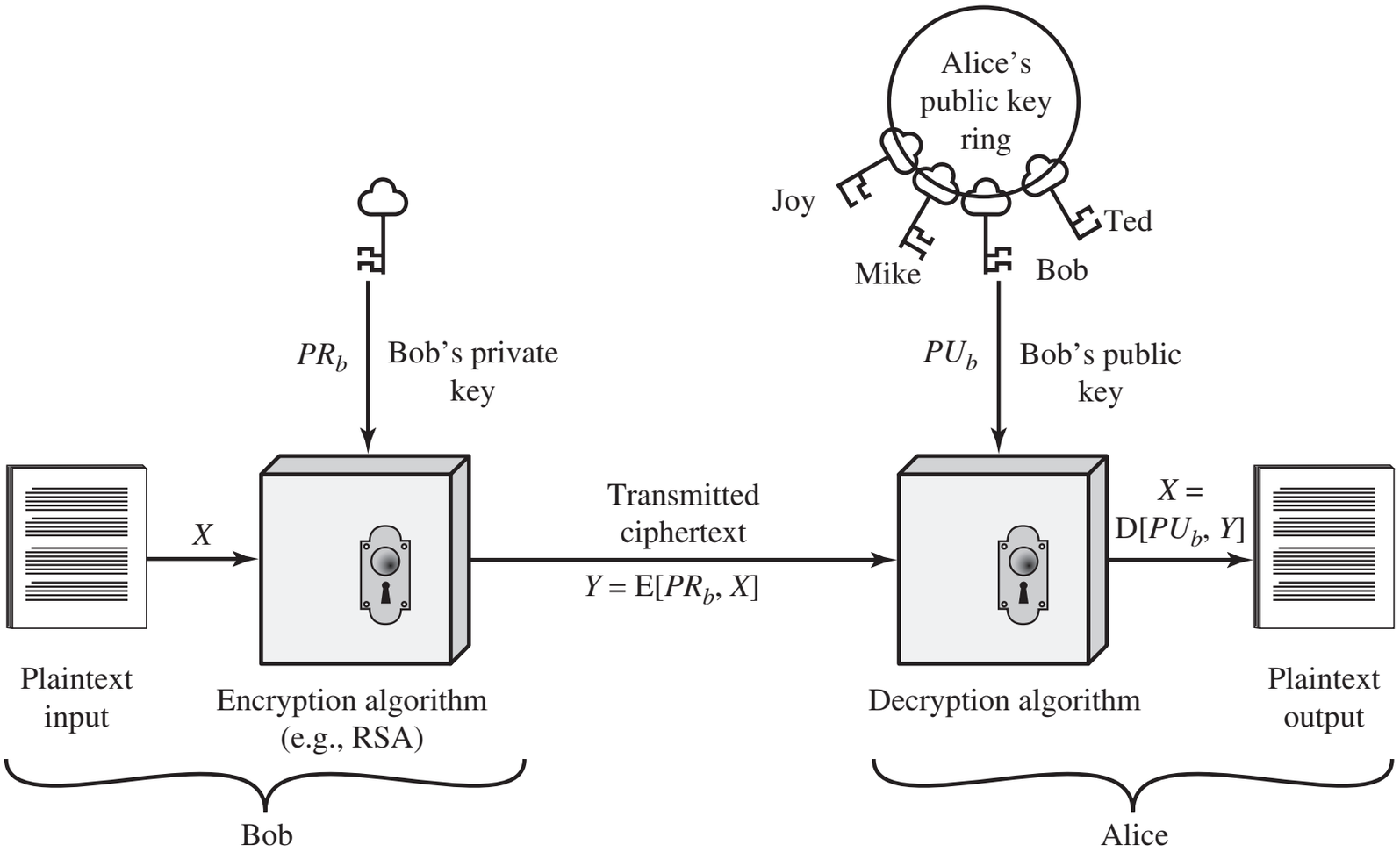
- User generates a pair of keys
- Places public key in accessible files, keep private the other key
- Send a private message to "Alice" — encryption using Alices' public key
- Only Alice can decrypt the message with private key

Asymmetric Encryption



Confidentiality

Asymmetric Encryption



Authentication

- A user is able to recover the plaintext using Bob's public key
- Only Bob encrypted the plaintext — authentication

Asymmetric Encryption: Conditions

- Computationally easy for a party B to generate a pair of keys
- Computationally easy for a sender A, knowing the public key PU_b and the plain text M, to generate the corresponding cipher-text

$$C = \mathrm{E}(PU_b, M)$$

- Computationally easy for a receiver B to decrypt the cipertext using the private key PR_b

$$M = \mathrm{D}(PR_b, C) = \mathrm{D}[PR_b, \mathrm{E}(PU_b, M)]$$

Asymmetric Encryption: Conditions

- Computationally infeasible for an opponent, knowing the public key, to determine the private key
- Computationally infeasible for an opponent, knowing the public key, and a ciphertext, to recover the plaintext

Asymmetric Encryption: Conditions

- Either of the two related keys can be used for encryption, with the other used for decryption

$$M = \mathrm{D}[PU_b, \mathrm{E}(PR_b, M)] = \mathrm{D}[PR_b, \mathrm{E}(PU_b, M)]$$

Algorithms: RSA

- Developed in 1977 by Ron ***R***ivest, Adi ***S***hamir, and Len ***A***dleman
- Asymmetric encryption
- Block Cipher
- Application: Secure Sockets Layer (SSL), Transport Layer Security (TLS)

Algorithms: RSA

- Encryption $C = M^e \mod n$
- Decryption $M = C^d \mod n = (M^e)^d \mod n = M^{ed} \mod n$

Plaintext block M

Ciphertect block C

Public key	$PU = \{e, n\}$	Sender: e, n
Private key	$PR = \{d, n\}$	Receiver: e, n, d

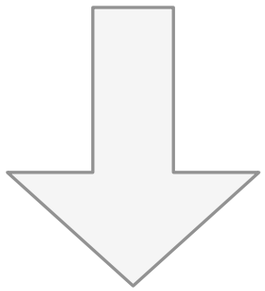
Requirement $M^{ed} \mod n = M$

Algorithms: RSA

Requirement $M^{ed} \bmod n = M$

Public key $PU = \{e, n\}$

Private key $PR = \{d, n\}$



The relationship holds if e and d are multiplicative inverses modulo $\phi(n)$, where $\phi(n)$ is the Euler totient function.

Algorithms: RSA

- Preliminary

1. e and d are multiplicative inverses mod $\phi(n)$

$$\Updownarrow$$

$$ed \bmod \phi(n) = 1$$

$$\Updownarrow$$

$$\gcd(\phi(n), d) = 1 \quad \text{and} \quad \gcd(\phi(n), e) = 1$$

greatest common divisor examples: $\gcd(12, 13) = 1$

$\gcd(12, 6) = 6$

$\gcd(14, 12) = 2$

Algorithms: RSA

- Preliminary

2. $\phi(n)$ is the Euler totient function.

For p, q prime, n=pq, $\phi(pq) = (p-1)(q-1)$

$\phi(n)$ is the number of positive integers less than n and relatively prime to n.

Algorithms: RSA

- Key generation

Select p, q	p and q both prime, $p \neq q$
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	$de \bmod \phi(n) = 1$
Public key	$KU = \{e, n\}$
Private key	$KR = \{d, n\}$

Example: $p = 2, q = 7$

$n = 14$

$\phi(n) = 6$

$e = 5$

$d = 17$ (multiple choices)

Algorithms: RSA

- Key generation

Example: $de \mod \phi(n) = 1$

$$5d \mod 6 = 1$$

Use Euler's algorithm to express 1 as an integer combination of 5 and 6

$$6 = 1(5) + 1$$

Therefore

$$1 = 6 - 1(5)$$

This gives us:

$$5d \mod 6 = [6 - 1(5)] \mod 6$$

$$\rightarrow 5d = -1(5),\ d = -1 \longrightarrow d = -1 + 6N \text{ for any integer } N$$

Algorithms: RSA

- Encryption

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

Example: $n = 14, e = 5, M = 2$

$$C = 2^5 \mod 14 = 4$$

- Decryption

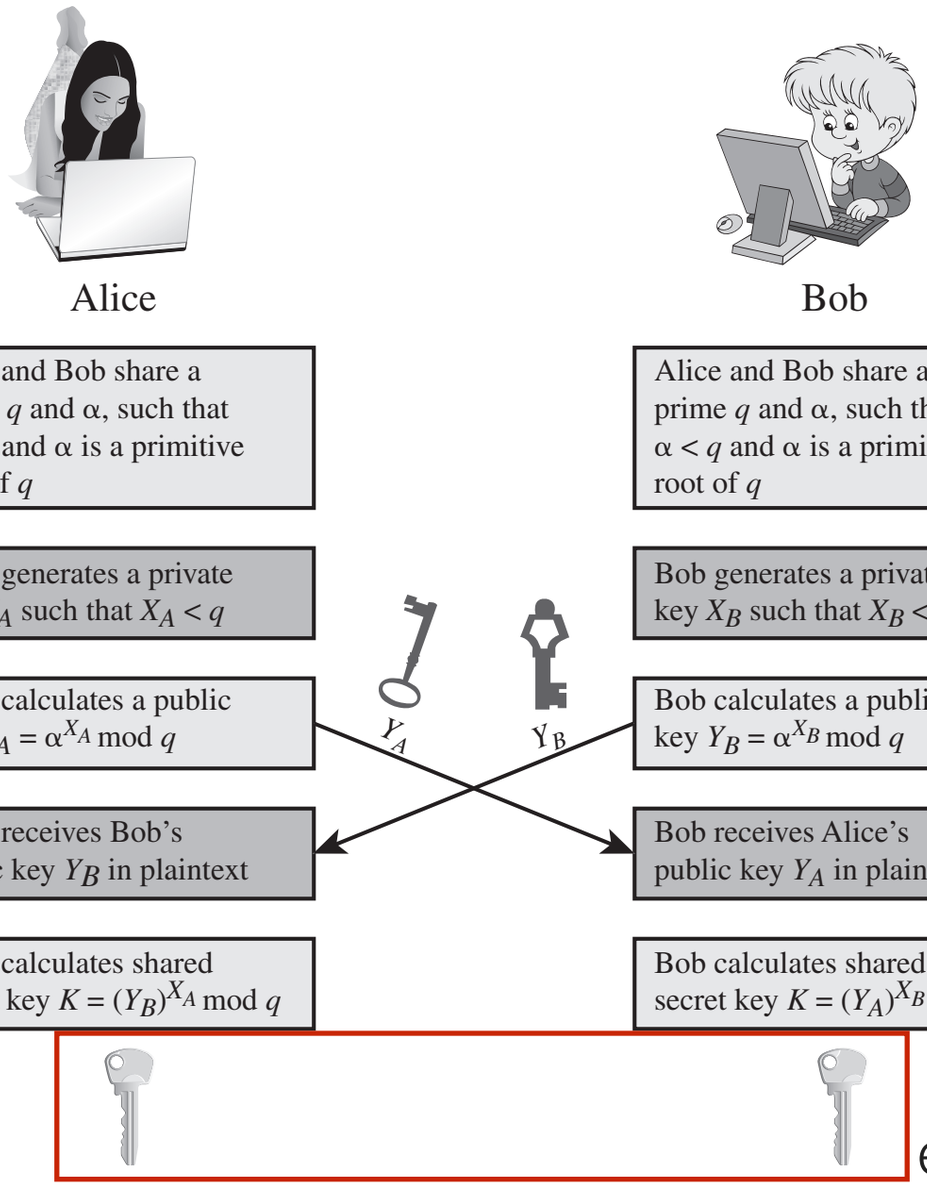
Ciphertext:	C
Plaintext:	$M = C^d \pmod{n}$

Example: $d = 17, n = 14, C = 4$

$$M = 4^{17} \mod 14 = 2$$

Algorithms: Diffie-Hellman Key Exchange

- Purpose: Two users exchange a secret key securely that can then be used for subsequent encryption of messages.



e.g., for a symmetric cipher

Algorithms: Diffie-Hellman Key Exchange

Global Public Elements	
q	Prime number
α	$\alpha < q$ and α a primitive root of q

Example: $q = 13, \alpha = 2$

$$
\begin{aligned}
2^0 &\mod 13 = 1 \\
2^1 &\mod 13 = 2 \\
2^2 &\mod 13 = 4 \\
2^3 &\mod 13 = 8 \\
2^4 &\mod 13 = 3 \\
2^5 &\mod 13 = 6 \\
2^6 &\mod 13 = 12 \\
2^7 &\mod 13 = 11 \\
2^8 &\mod 13 = 9 \\
2^9 &\mod 13 = 5 \\
2^10 &\mod 13 = 10 \\
2^11 &\mod 13 = 7
\end{aligned}
$$

1 to q-1

Algorithms: Diffie-Hellman Key Exchange

Global Public Elements	
q	Prime number
α	$\alpha < q$ and α a primitive root of q

Example: $q = 13, \alpha = 2$

2 is a primitive root mod 13

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

2 is not a primitive root mod 7

$2^0 \mod 7 = 1$
$2^1 \mod 7 = 2$
$2^2 \mod 7 = 4$
$2^3 \mod 7 = 1$
$2^4 \mod 7 = 2$
$2^5 \mod 7 = 4$
$2^6 \mod 7 = 1$

Algorithms: Diffie-Hellman Key Exchange

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

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

Example: $q = 13, \alpha = 2$

$$X_A = 3 \rightarrow Y_A = 2^3 \mod 13 = 8$$

$$X_B = 5 \rightarrow Y_B = 2^5 \mod 13 = 6$$

Algorithms: Diffie-Hellman Key Exchange

Generation of Secret Key by User A

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

Generation of Secret Key by User B

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

Example: $q = 13, \alpha = 2$

$$X_A = 3 \rightarrow Y_A = 2^3 \mod 13 = 8 \qquad K = 6^3 \mod 13 = 8$$

$$X_B = 5 \rightarrow Y_B = 2^5 \mod 13 = 6 \qquad K = 8^5 \mod 13 = 8$$

Algorithms: Diffie-Hellman Key Exchange

- Public $q = 13, \alpha = 2, Y_A = 8, Y_B = 6$
- User A $q = 13, \alpha = 2, X_A = 3, Y_A = 8, Y_B = 6$
- User B $q = 13, \alpha = 2, X_B = 5, Y_B = 6, Y_A = 8$

$$K = (Y_B)^{X_A} \bmod q \;=\; (Y_A)^{X_B} \bmod q$$

How to get K with the public information?

Algorithms: Diffie-Hellman Key Exchange

- Public $q = 13, \alpha = 2, Y_A = 8, Y_B = 6$
- User A $q = 13, \alpha = 2, X_A = 3, Y_A = 8, Y_B = 6$
- User B $q = 13, \alpha = 2, X_B = 5, Y_B = 6, Y_A = 8$

$$K = (Y_B)^{X_A} \bmod q \;=\; (Y_A)^{X_B} \bmod q$$

Attack Example: Find X_a or X_B by solving

$2^x \bmod 13 = 8$ or $2^x \bmod 13 = 6$ by brute force approach

The problems becomes impractical with larger numbers!

Algorithms: Diffie-Hellman Key Exchange

- Man-in-the-middle attack

Alice

adversary

Bob

Private keys X_{D_1}, X_{D_2}

transmits Y_A to Bob	intercepts Y_A and transmits Y_{D_1} to Bob calculates $K_2 = (Y_A)^{X_{D_2}} \mod q$	receives Y_{D_1} $K_1 = (Y_{D_1})^{X_B} \mod q$
receives Y_{D2} $K_2 = (Y_{D_2})^{X_A} \mod q$	intercepts Y_B and transmits Y_{D2} to Alice calculates $K_1 = (Y_B)^{X_{D_1}} \mod q$	transmits Y_B to Alice

Bob and Alice think that they share a secret key

Bob-Adversary share K_1 Alice-Adversary share K_2

Algorithms: Diffie-Hellman Key Exchange

- Man-in-the-middle attack

Alice

adversary

Bob

sends an encrypted message

$E(K_2, M)$

intercepts and decrypts by K_2

recovers M

sends Bob $E(K_1, M)$ or $E(K_1, M')$

Asymmetric Encryption

- Is public-key encryption more secure than symmetric encryption?

Asymmetric Encryption

- Is public-key encryption more secure than symmetric encryption?
  - No
- Security of encryption depends on
  - The length of the key
  - The computational work involved in breaking a cipher

Asymmetric Encryption

- Is symmetric encryption outdated?

Asymmetric Encryption

- Is symmetric encryption outdated?
  - No — Symmetric encryption use less computational overhead
  - Examples: AES is used to encrypt data at rest and in transit, e.g., SSD, google cloud.

Asymmetric Encryption

- Is public key distribution trivial?
  - No — need a central agent and protocols

Tutorial Question 1

Problem 1. For any block cipher, the fact that it is a nonlinear function is crucial to its security. To see this, suppose that we have a linear block cipher EL that encrypts 128-bit blocks of plaintext into 128-bit blocks of ciphertext. Let $EL(k, m)$ denote the encryption of a 128-bit message m under a key k (the actual bit length of k is irrelevant). Thus

$$EL(k, [m_1 \oplus m_2]) = EL(k, m_1) \oplus EL(k, m_2) \text{ for all 128-bit patterns } m_1, m_2 \quad (1)$$

Describe how, with 128 chosen ciphertexts, an adversary can decrypt any ciphertext without knowledge of the secret key k. (A "chosen ciphertext" means that an adversary has the ability to choose a ciphertext and then obtain its decryption. Here, you have 128 plaintext/ciphertext pairs to work with and you have the ability to chose the value of the ciphertexts.)

Chosen ciphertext:	Paired plaintext
$c_1 = [1, 0, 0, \cdots, 0]$	m_1
$c_2 = [0, 1, 0, \cdots, 0]$	m_2
$\vdots$	$\vdots$
$c_{128} = [0, 0, 0, \cdots, 1]$	m_{128}

Any non-zero string (ciphertext) can be represented by using XOR of c_i

Tutorial Question 1

$$EL(k, [m_1 \oplus m_2]) = EL(k, m_1) \oplus EL(k, m_2) \text{ for all 128-bit patterns } m_1, m_2$$

Chosen ciphertext:	Paired plaintext
$c_1 = [1, 0, 0, \cdots, 0]$	m_1
$c_2 = [0, 1, 0, \cdots, 0]$	m_2
$\vdots$	$\vdots$
$c_{128} = [0, 0, 0, \cdots, 1]$	m_{128}

Example: $[1, 0, 1, \cdots, 0] = c_1 \oplus c_3$

plaintext: $m_1 \oplus m_3$

Tutorial Question 1

$EL(k, [m_1 \oplus m_2]) = EL(k, m_1) \oplus EL(k, m_2)$ for all 128-bit patterns m_1, m_2

For all-zero string EL[k, 0]= 0 why?

Suppose EL[k,0]=1 then we will have the following contradiction

set plaintext as $m_1 = m_2 = 0$

$m_1 \oplus m_2 = 0$

$EL[k, m_1 \oplus m_2] = 1 \neq EL[k, m_1] \oplus EL[k, m_2]$

Tutorial Question 2

Problem 2. What RC4 key value will leave S unchanged during initialization? That is, after the initial permutation of S, the entries of S will be equal to the values from 0 through 255 in ascending order.

```
/* Initialization */
for i = 0 to 255 do
S[i] =i;
T[i] = K[i mod keylen];
```

```
/* Initial Permutation of S */
j = 0;
for i = 0 to 255 do
   j = (j + S[i] + T[i]) mod 256;
   Swap (S[i], S[j]);
```

To make it simple, we use a key of length 256, T is a copy of K (key).

Design T[i] (K[i]) such that `(j + S[i] + T[i]) mod 256 = i`

$$K[0] = K[1] = 0 \qquad K[2] = 255; K[3] = 254; \ldots K[255] = 2$$

Tutorial Question 3

Problem 3. RC4 has a secret internal state which is a permutation of all the possible values of the vector S and the two indices i and j.

(a) Using a straightforward scheme to store the internal state, how many bits are used?

(b) Suppose we think of it from the point of view of how much information is represented by the state. In that case, we need to determine how may different states there are, then take the log to the base 2 to find out how many bits of information this represents. Using this approach, how many bits would be needed to represent the state?

(a) i for 0 to 255, use 8 bits, same for j, S[0], S[1], …, S[255] use 8*256 bits

(b) Initialization of 0-255 —> 256! Swap 256*256 $\log_2(256! \times 256^2) \approx 1700$

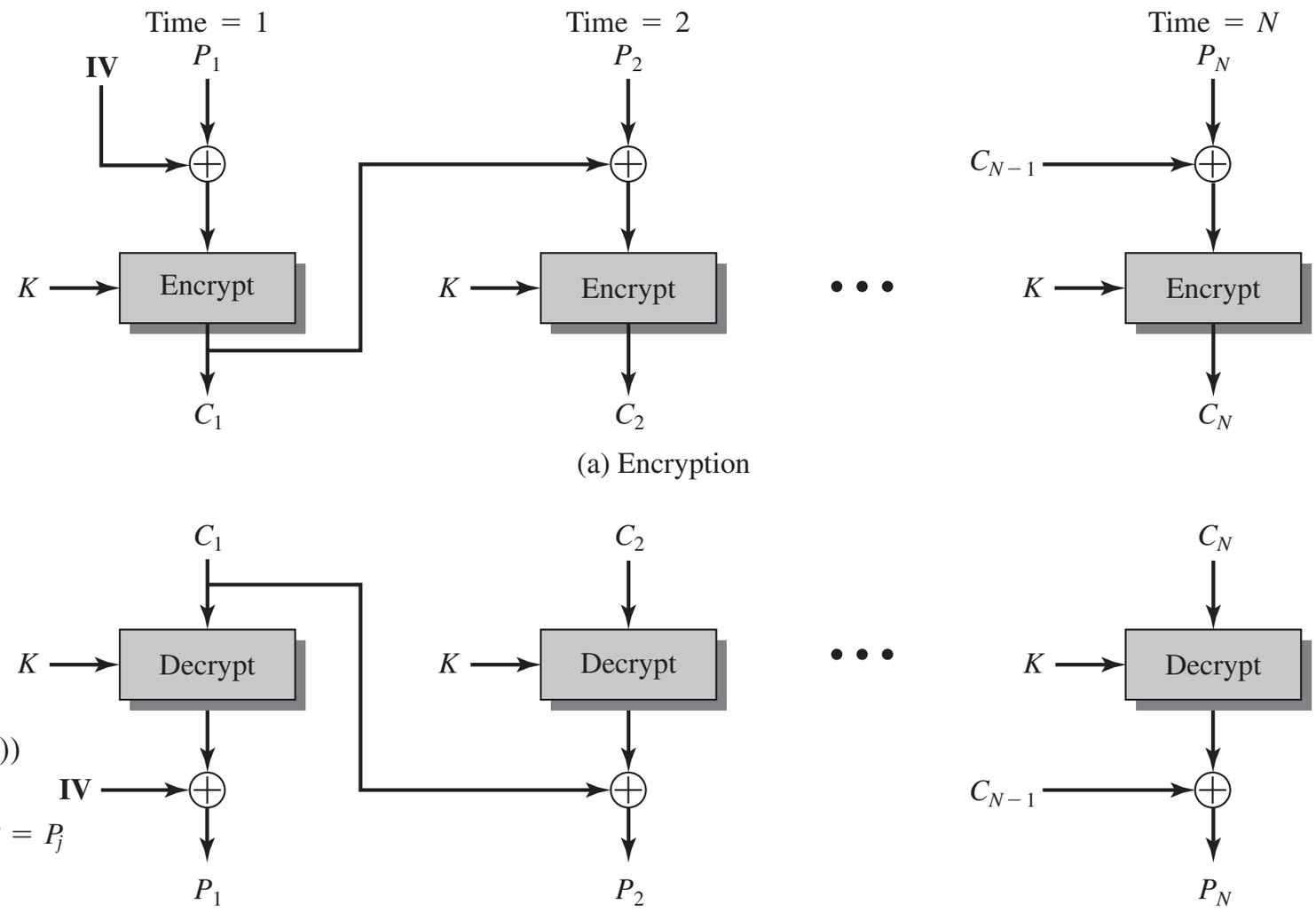
```
/* Initialization */
for i = 0 to 255 do
S[i] =i;
T[i] = K[i mod keylen];
```

```
/* Initial Permutation of S */
j = 0;
for i = 0 to 255 do
   j = (j + S[i] + T[i]) mod 256;
   Swap (S[i], S[j]);
```

Tutorial Question 4

Problem 4. Suppose an error occurs in a block of ciphertext on transmission using Cipher Block Chaining (CBC). What effect is produced on the recovered plaintext blocks?

$C_j = \mathrm{E}(K, [C_{j-1} \oplus P_j])$



(a) Encryption

$\mathrm{D}(K, C_j) = \mathrm{D}(K, \mathrm{E}(K, [C_{j-i} \oplus P_j]))$

$\mathrm{D}(K, C_j) = C_{j-1} \oplus P_j$

$C_{j-1} \oplus \mathrm{D}(K, C_j) = C_{j-1} \oplus C_{j-1} \oplus P_j = P_j$

(b) Decryption

Solution: If an error occurs in transmission of ciphertext block C_i, then this error propagates to the recovered plaintext blocks P_i and P_{i+1}.

Thank You

Quiz Time

- 15 minutes