

10/21/26

COMP 435:



Computer Security Concepts

Lecture 24: Public Key Encryption



The University
of North Carolina
at Chapel Hill

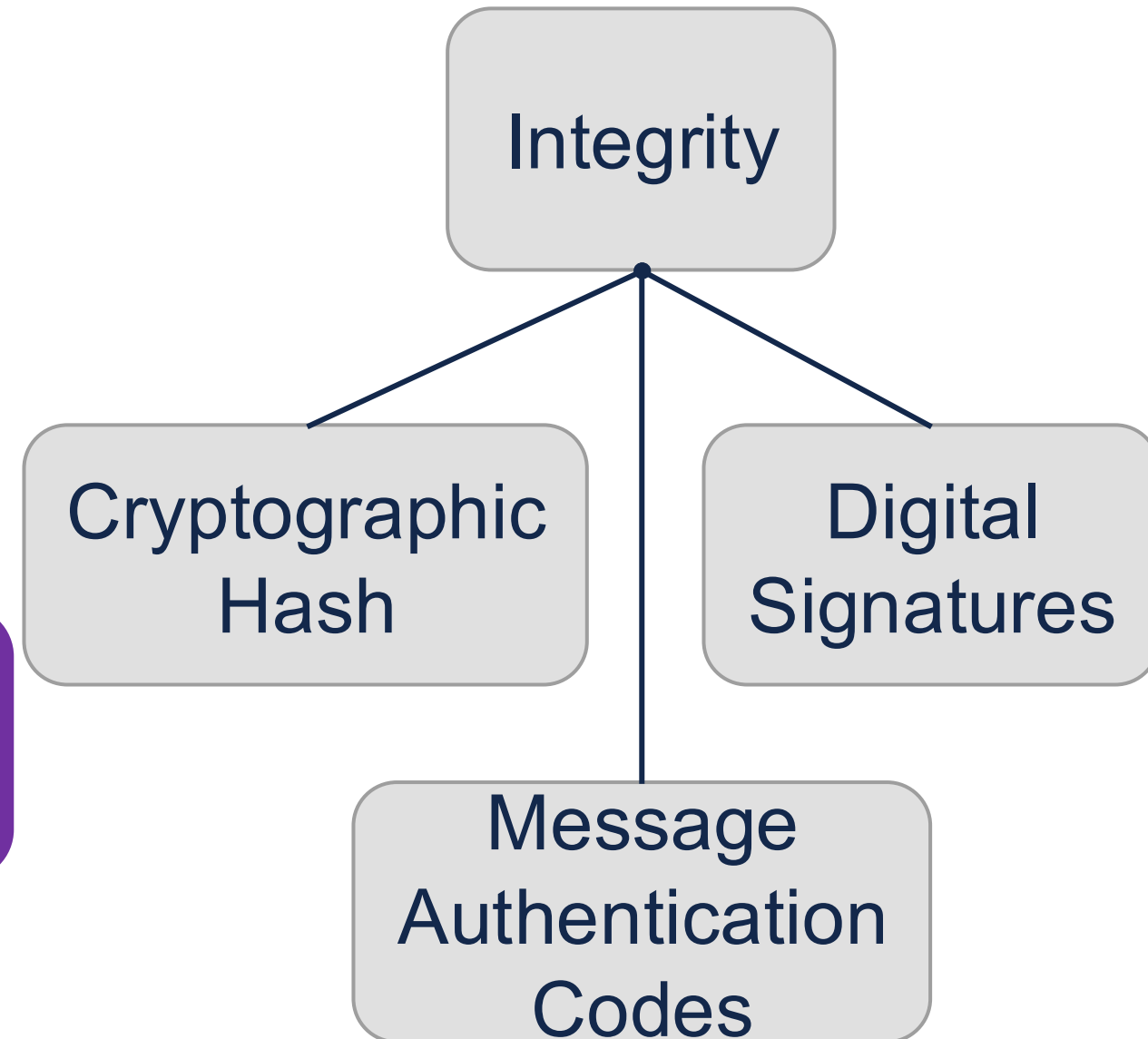
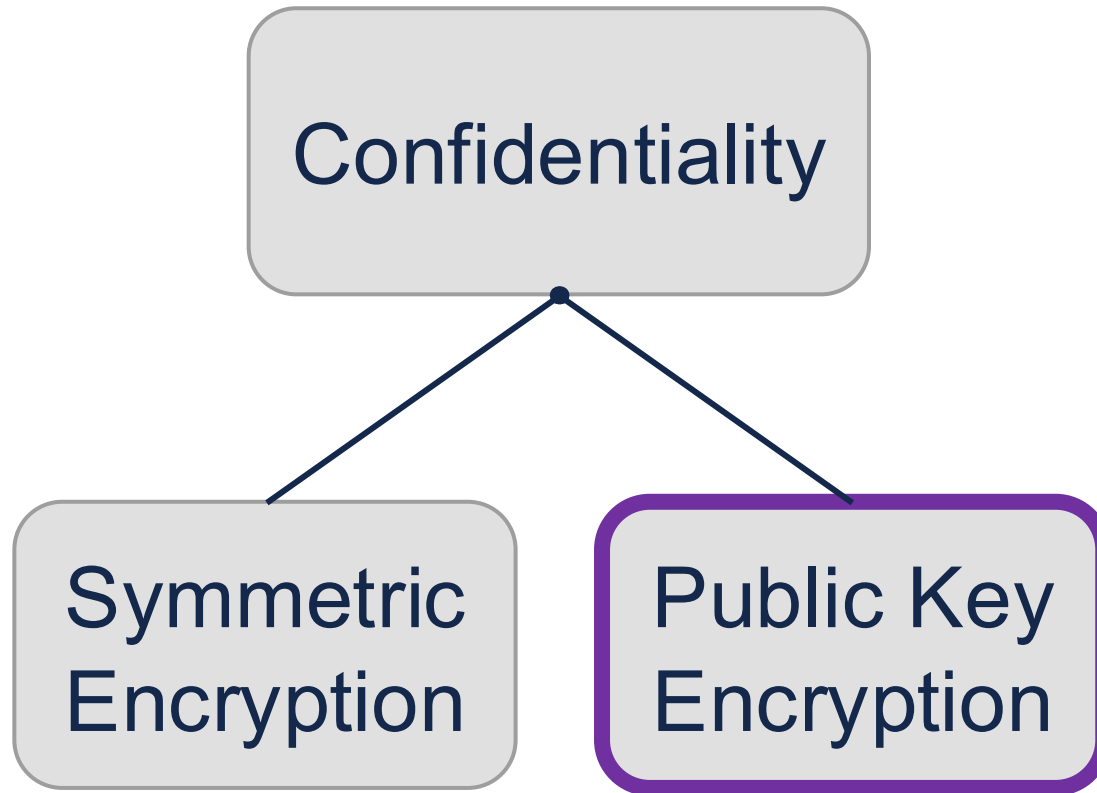
<https://tinyurl.com/comp435-fa26>

Logistics Update



- Project Proposal due Today
 - Working in groups is optional!
- WA4 on Friday. Crypto in Practice. I will assign topics after class!
 - Part 1 due thurs at 11:59pm
 - Part 2 in-class

Roadmap



Public Key Encryption

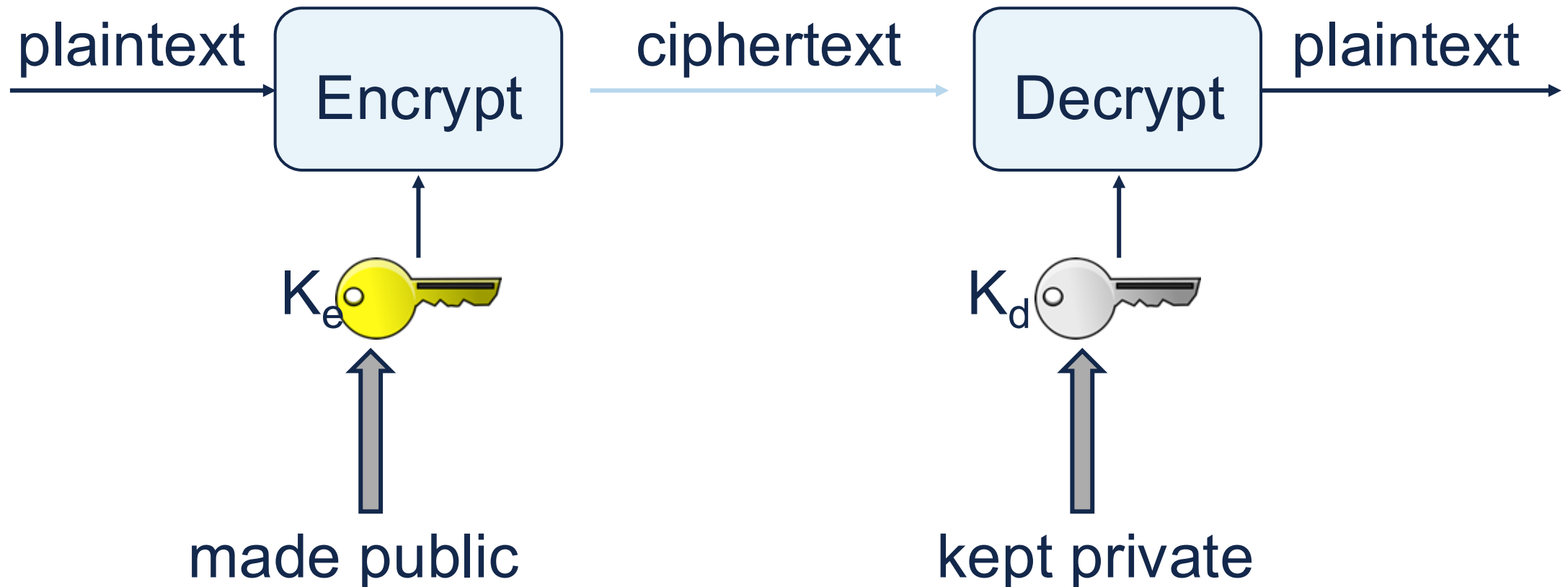


Public Key Encryption



Public Key Encryption

$$c = \text{Enc}_{K_e}(\text{msg})$$
$$\text{msg} = \text{Dec}_{K_d}(c)$$



msg

Dear Bob,
The secret
word is
cookie.
Sincerely,
Alice

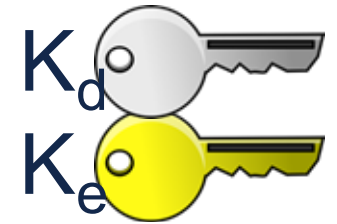
Alice



Hello, World.
This is my
public key.



Bob



msg

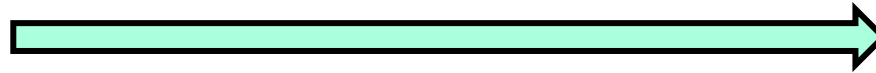
Dear Bob,
The secret
word is
cookie.
Sincerely,
Alice

Alice

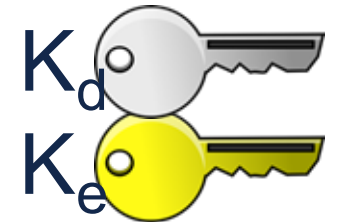


$$c = \text{Enc}_{K_e}(\text{msg})$$

c



Bob



$$\text{msg} = \text{Dec}_{K_d}(c)$$



msg

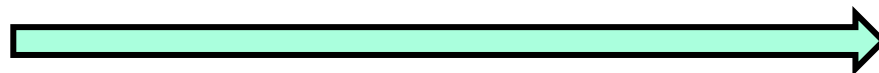
Dear Bob,
The secret
word is
cookie.
Sincerely,
Alice

Alice

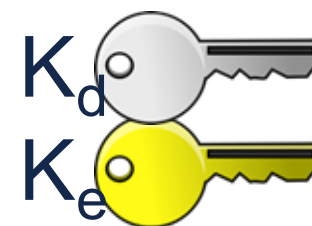


$$c = \text{Enc}_{K_e}(\text{msg})$$

c



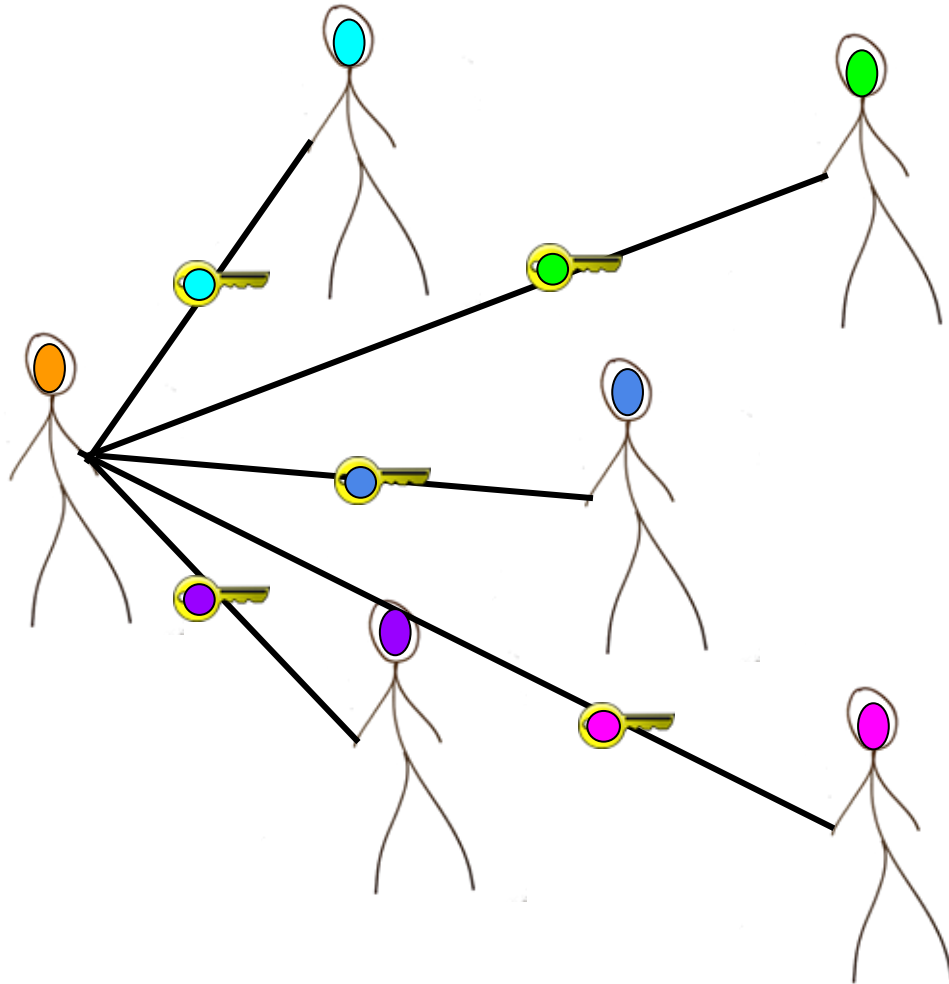
Bob



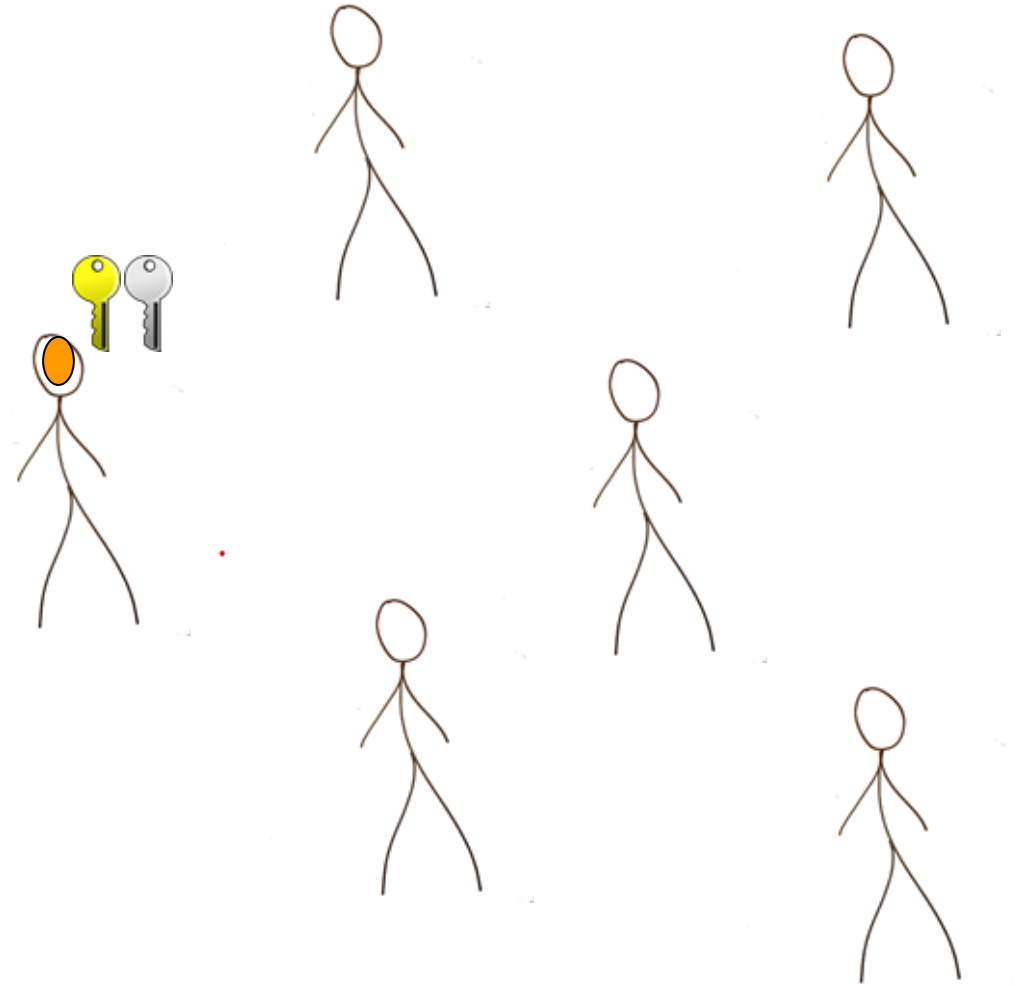
$$\text{msg} = \text{Dec}_{K_d}(c)$$



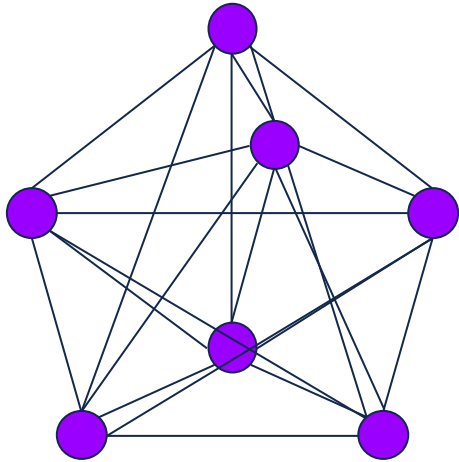
Symmetric Encryption



Public Key Encryption

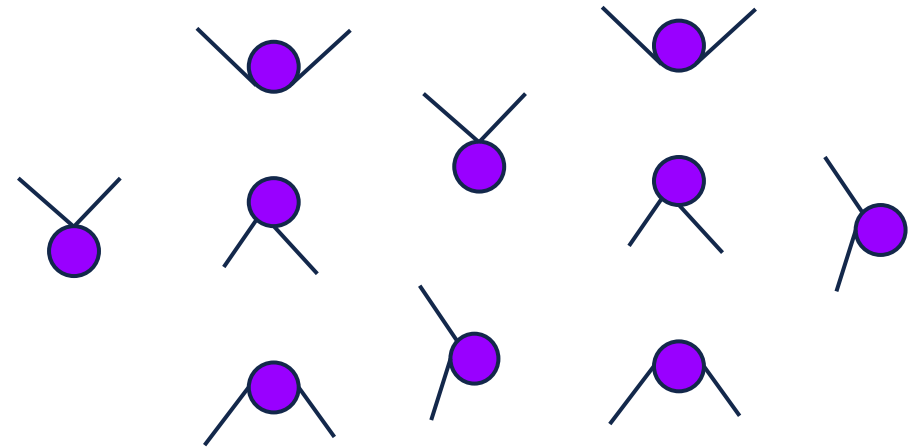


Symmetric Encryption



$$\frac{n(n-1)}{2} \text{ keys}$$

Public Key Encryption



$$2n \text{ keys}$$

RSA



- Ron Rivest, Adi Shamir, Leonard Adleman
- 1978
- Turing Award (2002)
- Based on the hardness of factoring
- Keys are 1024-, 2048-bits long



RSA Key Generation

1. Input parameters: Given $l > 2$ and odd $e > 2$:
2. Generate prime p s.t.
 - p is prime, is l -bits long, and $\gcd(e, p-1) = 1$
3. Generate prime q s.t.
 - $q \neq p$, q is prime, is l -bits long, and $\gcd(e, q-1) = 1$
4. Compute N , where $N := pq$. N is the RSA modulus
5. Compute private exponent, d , s.t. $e \cdot d \equiv 1 \pmod{(p-1)(q-1)}$



RSA Key Generation

1. Input parameters: Given $l > 2$ and odd $e > 2$:
2. Generate prime p s.t.
 - p is prime, is l -bits long, and $\gcd(e, p-1) = 1$
3. Generate prime q s.t.
 - $q \neq p$, q is prime, is l -bits long, and $\gcd(e, q-1) = 1$
4. Compute N , where $N := pq$. N is the RSA modulus
5. Compute private exponent, d s.t. $e \cdot d \equiv 1 \pmod{(p-1)(q-1)}$

The public key
is (N, e)

The private
key is (N, d)

RSA

large primes



$$N = pq$$
The equation $N = pq$ is shown. Two purple curved arrows originate from the text 'large primes' and point to the variables 'p' and 'q' respectively, indicating that both are large primes.

public key $\longrightarrow K_e = (e, N)$

private key $\longrightarrow K_d = (d, N)$

RSA

large primes



$$N = pq$$
The equation $N = pq$ is shown. Two curved purple arrows originate from the text 'large primes' and point to the variables p and q in the equation.

public key $\longrightarrow K_e = (e, N)$

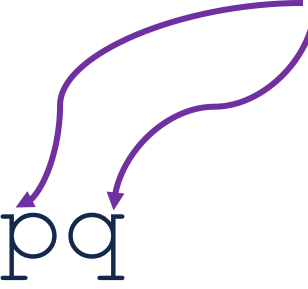
private key $\longrightarrow K_d = (d, N)$

$$ed \equiv 1 \pmod{\phi(N)}$$
A thick purple arrow points upwards from the congruence equation $ed \equiv 1 \pmod{\phi(N)}$ to the variable d in the private key definition $K_d = (d, N)$.

RSA

large primes



$$N = pq$$


public key $\longrightarrow K_e = (e, N)$

private key $\longrightarrow K_d = (d, N)$

$$\gcd(e, \phi(N)) = 1$$

Euler
totient
function

$\longrightarrow \phi(N) = (p-1)(q-1)$

$= \# \text{ pos integers } < N \text{ relatively prime to } N$





RSA Encryption and Decryption

- Encryption: $C = \text{msg}^e \bmod N$
- Decryption: $\text{msg} = C^d \bmod N$

Given N , e and msg^e , it is difficult to compute msg

Given N , e , msg^e , and msg , it is difficult to compute d



RSA Correctness

- $\text{msg} = (\text{msg}^e)^d \bmod N$
- Thm.
For all primes $p, q, p \neq q$,
Let $N = pq$,
 $ed \equiv 1 \pmod{(p-1)(q-1)} \rightarrow$
 $\forall x \in \mathbb{Z}, x^{ed} \equiv x \pmod{N}$.

RSA Security



- $c = \text{msg}^e \bmod N$
- The attacker's challenge:
Given N , e , msg^e , find msg
- If the attacker can factor N , they win

One-Way Functions



- a function $F: X \rightarrow Y$ is one-way if F is easy to compute but hard to invert
- multiplying two large primes is one-way
 - multiplying $pq = N$ is easy to compute
 - factoring N to find p, q is hard



Trap-Door Functions

- a one-way function $F: X \rightarrow Y$ is a trap-door function if there exists a secret to make inversion easy
- inverting $x^e \bmod N$ is hard
 - d is the trap door
 - computing $x^{ed} \bmod N$ is easy

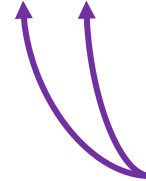
RSA



Public Key $K_e: (N, e)$

Private Key $K_d: (N, d)$

$$N = pq$$



must remain secret!

An Example



$$p = 5$$

$$q = 7$$

$$N = 35$$

$$(p-1)(q-1) = \phi(N) = 24$$

$$e = 5$$

$$d = 5$$

message: 2

$$E_{ke} = 2^5 \bmod N = 32$$

ciphertext: 32

$$E_{kd} = 32^5 \bmod N = 2$$

WS Q1





- $\frac{n(n-1)}{2}$ keys
- key distribution
- + fast algorithms

Symmetric Encryption

- + $2n$ keys
- + publish public key
- slow algorithms

Public Key Encryption

WS Q2





**Establishing a shared
secret key! 😊**

Key Establishment



Def'n: a means of agreeing on a shared secret

E.g., Two communicating parties establish the symmetric key to be used for AES encryption

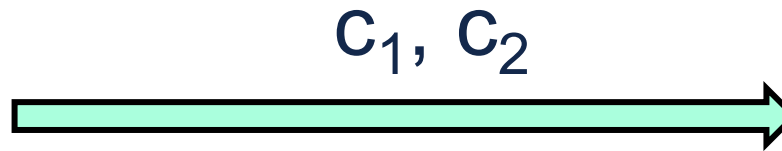


Hybrid Encryption for Key Establishment



$$c_1 = \text{PK-Enc}_{\text{Ke}}(k)$$

$$c_2 = \text{S-Enc}_k(\text{msg})$$



$$k = \text{PK-Dec}_{K_d}(c_1)$$

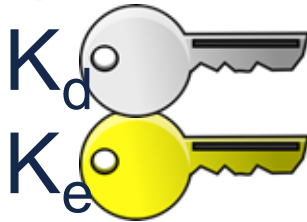
$$\text{msg} = \text{S-Dec}_k(c_2)$$

Purple encrypts session key S with Orange's public key! Then, they use S for symmetric encryption.

Hybrid Encryption



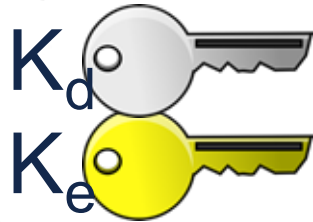
I'm Purple. Let's use: $\text{RSA-Enc}_{K_e}(S)$



Hybrid Encryption: Replay Attack



I'm Purple. Let's use: $Enc_{K_e}(S)$



The attacker records the setup message and sends it again later, claiming to be purple.

Orange may accept old key as a new session key. Encryption alone doesn't establish the sender's identity or message freshness.



I'm Purple. Let's use: $Enc_{K_e}(S)$



Nonce



Def'n: a number used only once as part of a cryptographic protocol

- random number used to provide freshness, message continuity
- sequence number to demonstrate uniqueness
- timestamp to demonstrate timeliness

E.g., IV used in CBC, OFB, Ctr modes of block ciphers



Diffie-hellman Key Agreement

Diffie-Hellman Key Agreement

group $\sim$ set

$$n = g^i \bmod p$$

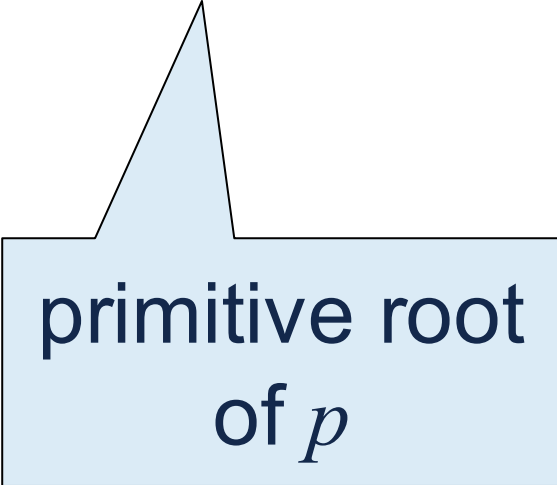
prime

primitive root
of p

For all n , $1 \leq n < p$,
there exists an integer i such that

$$g^i \bmod p = n$$

$$n = g^i \bmod p$$



primitive root
of p

Diffie-Hellman Key Agreement



$$a < p$$
$$A := g^a \bmod p$$



$$b < p$$
$$B := g^b \bmod p$$

$\longleftrightarrow g, p$

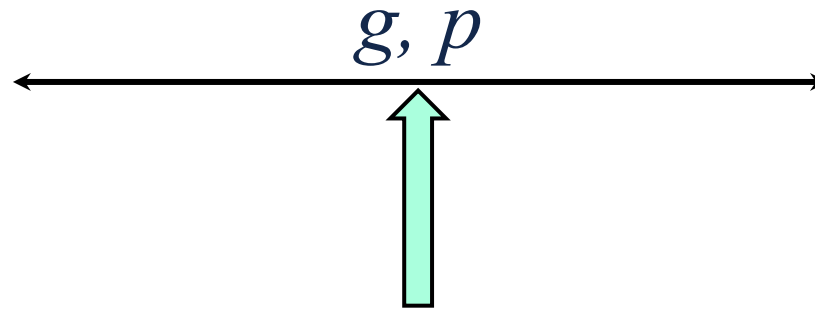
$\xrightarrow{A}$

$\xleftarrow{B}$

$$K := B^a \bmod p$$
$$= (g^b)^a \bmod p$$
$$= g^{ba} \bmod p$$

$$K := A^b \bmod p$$
$$= (g^a)^b \bmod p$$
$$= g^{ab} \bmod p$$

Diffie-Hellman Key Agreement



Diffie-Hellman Key Agreement



$$a < p$$

$$A := g^a \bmod p$$

$$g, p$$



$$b < p$$

$$B := g^b \bmod p$$

$$A$$

$$B$$

Diffie-Hellman Key Agreement



g, p



$$a < p$$

$$A := g^a \bmod p$$

$$b < p$$

$$B := g^b \bmod p$$

A

B

$$\begin{aligned} K &:= B^a \bmod p \\ &= (g^b)^a \bmod p \\ &= g^{ba} \bmod p \end{aligned}$$

$$\begin{aligned} K &:= A^b \bmod p \\ &= (g^a)^b \bmod p \\ &= g^{ab} \bmod p \end{aligned}$$

Diffie-Hellman Key Agreement



g, p



$$a < p$$

$$A := g^a \bmod p$$

$$b < p$$

$$B := g^b \bmod p$$

A

B

$$\begin{aligned} \textcircled{K} &:= B^a \bmod p \\ &= (g^b)^a \bmod p \\ &= g^{ba} \bmod p \end{aligned}$$

$$\begin{aligned} \textcircled{K} &:= A^b \bmod p \\ &= (g^a)^b \bmod p \\ &= g^{ab} \bmod p \end{aligned}$$



DH Correctness

- $K = A^b = (g^a)^b = B^a = (g^b)^a \bmod p$
- Both sides have computed the same value for key K



DH Security

- $K = A^b = (g^a)^b = (g^b)^a = B^a \bmod p$
- The attacker's challenge:
 - given $g, p, A, B,$
 - find K
- If the attacker can calculate the discrete log of A or B , they win

One-Way Function



- a function $F: X \rightarrow Y$ is one-way if F is easy to compute but hard to invert
- modular exponentiation is one-way
 - computing $g^a \bmod p = A$ is easy
 - computing $\text{dlog}_{g,p}(A) = a$,
the discrete log of $A \bmod p$ with base g , is hard

Adversary-in-the-Middle attack on DH



$$a$$
$$A := g^a \bmod p$$

$$c, d$$
$$C := g^c \bmod p$$
$$D := g^d \bmod p$$

$$b$$
$$B := g^b \bmod p$$

$\xrightarrow{A}$

$\xrightarrow{C}$

$\xleftarrow{D}$

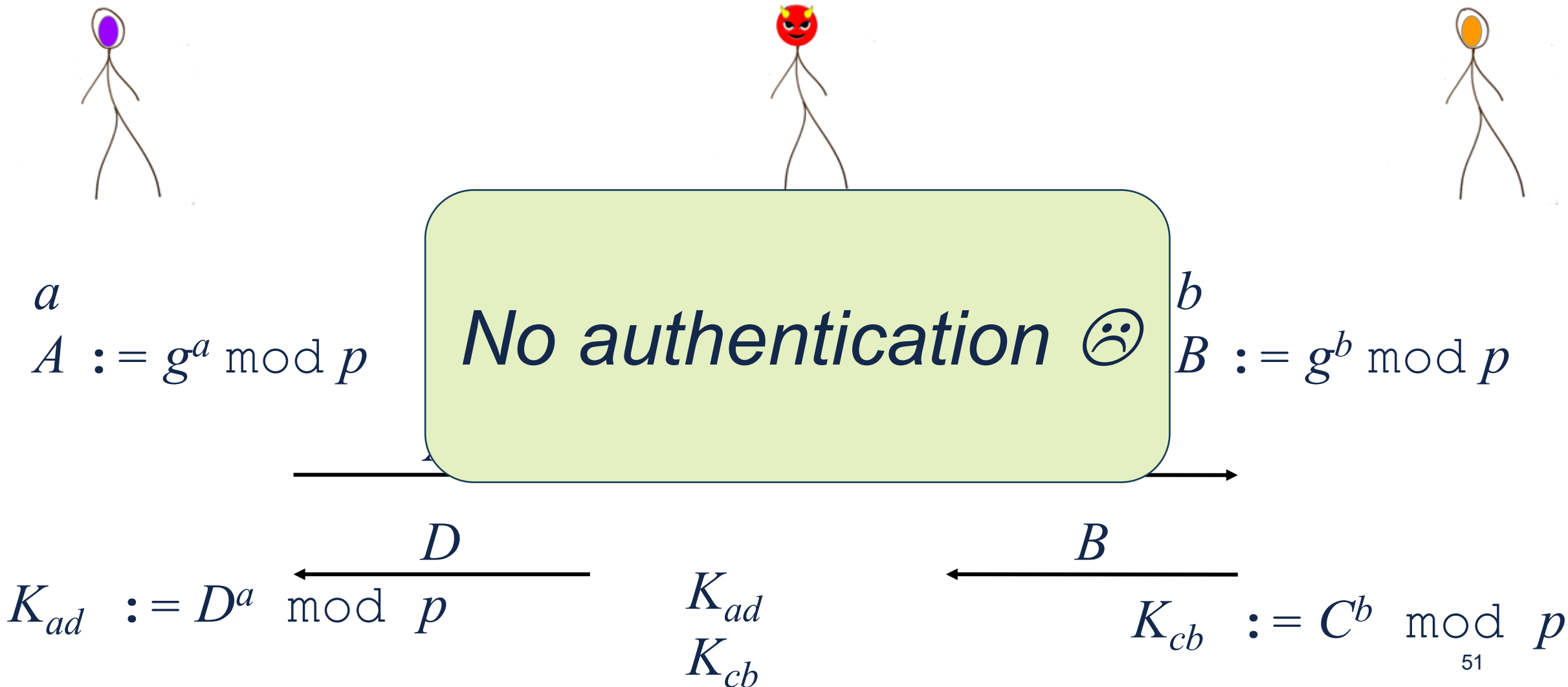
$\xleftarrow{B}$

$$K_{ad} := D^a \bmod p$$

$$K_{ad}$$
$$K_{cb}$$

$$K_{cb} := C^b \bmod p$$

Adversary-in-the-Middle attack on DH

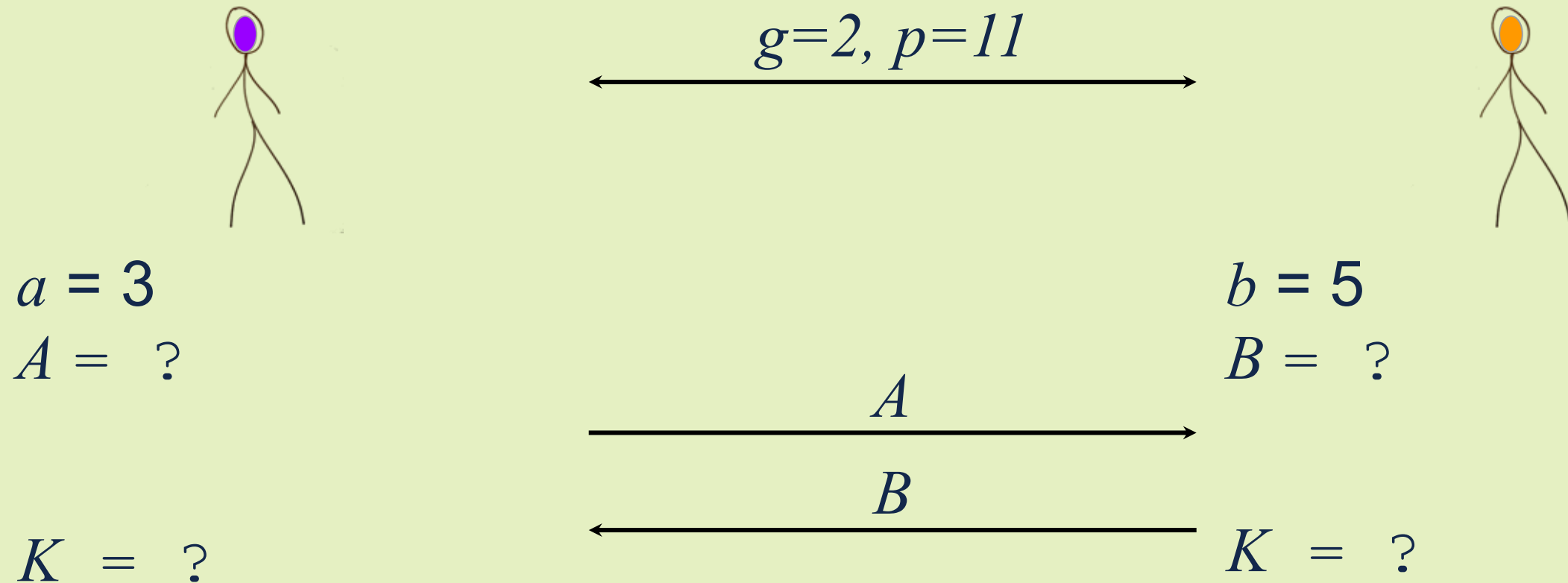




Machine in the Middle Attack (MitM) (or Adversary in the Middle (AitM))

Def'n: two parties, believing that they are communicating with each other, are in fact communicating with an attacker

Diffie-Hellman Key Agreement (Q2)





Cryptographic Hash Functions

Roadmap

