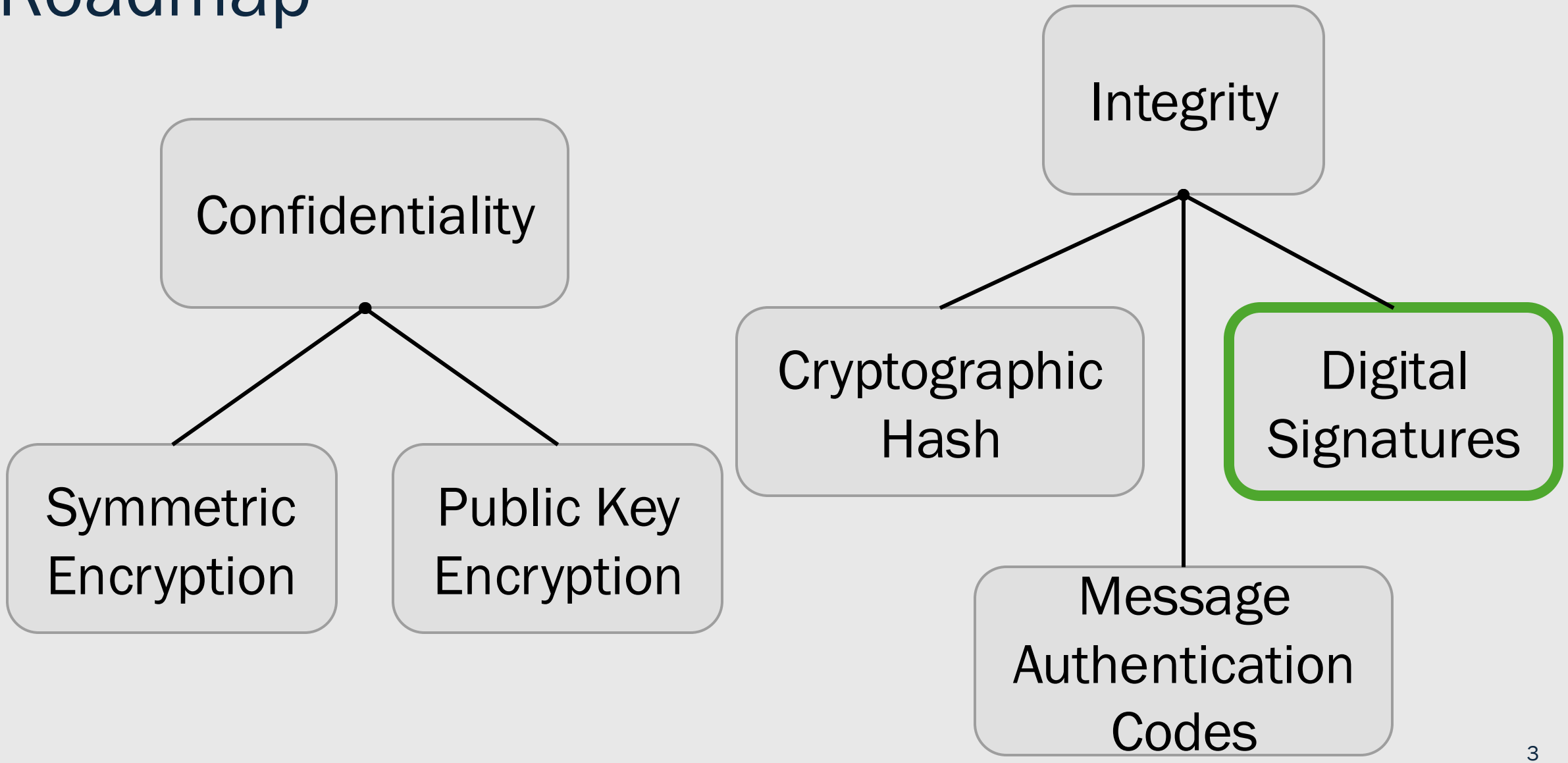
COMP435: *SECURITY CONCEPTS!*

Lecture 10: Digital Signatures + DH Key Exchange

tinyurl.com/comp435-fa25

DIGITAL SIGNATURES

Roadmap


Confidentiality
Symmetric Encryption
Public Key Encryption
Integrity
Cryptographic Hash
Digital Signatures
Message Authentication Codes

Digital Signature

Def’n: a means to ensure a given message comes from a given source.

Properties of a Digital Signature

- Authentication
- Integrity
- Accountability

Properties of a Digital Signature

- **Authentication**  Authentication of data origin
- Integrity
- Accountability

Properties of a Digital Signature

- Authentication
- **Integrity**  Integrity of data
- Accountability

Properties of a Digital Signature

- Authentication
- Integrity
- **Accountability**

Non-repudiation. A signer cannot deny having signed.

Digital Signature Requirements

- Unforgeable
- Authentic
- Unalterable
- Not reusable

Digital Signature Requirements

- **Unforgeable** 
- Authentic
- Unalterable
- Not reusable

No party can produce a viable signature by outside means

Digital Signature Requirements

- Unforgeable
- **Authentic**
- Unalterable
- Not reusable

A signature is tied to a single party

Digital Signature Requirements

- Unforgeable
- Authentic
- **Unalterable**
- Not reusable

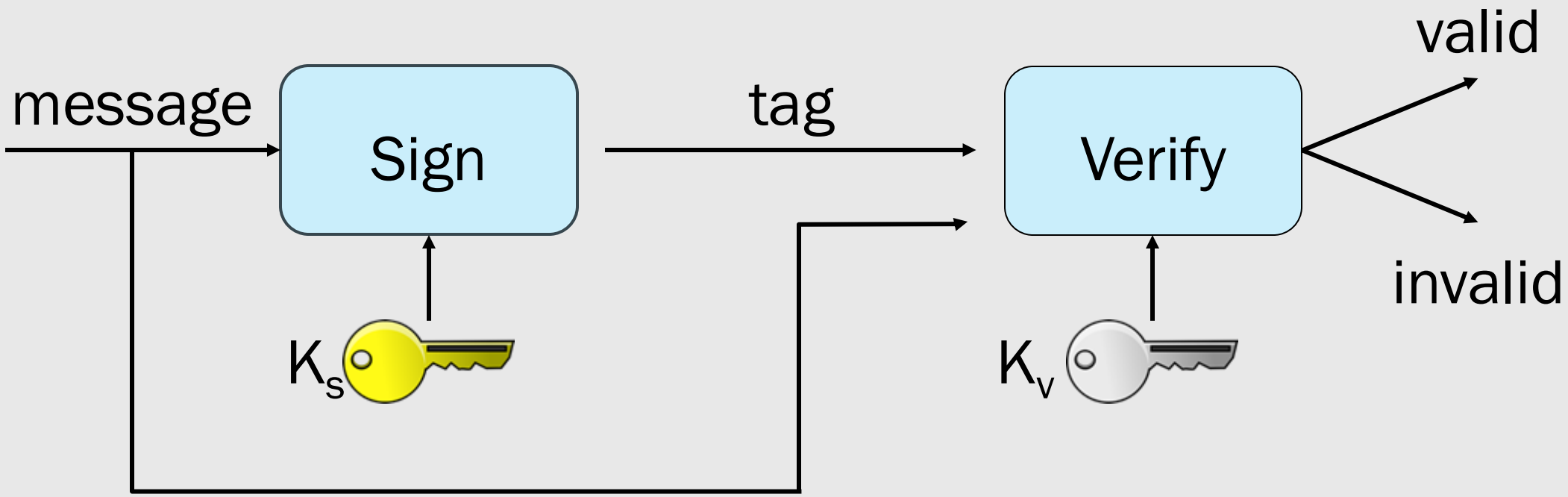
A signed message cannot be altered without invalidating the signature

Digital Signature Requirements

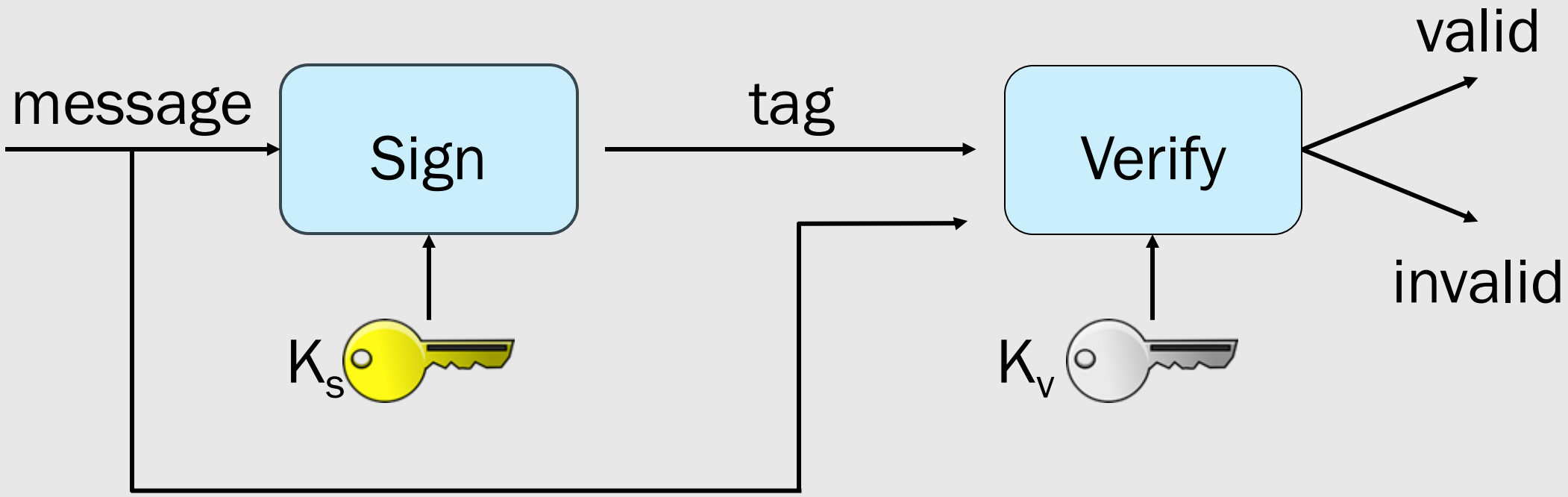
- Unforgeable
- Authentic
- Unalterable
- **Not reusable**

A signature is tied to the signed message

Public Key Signatures



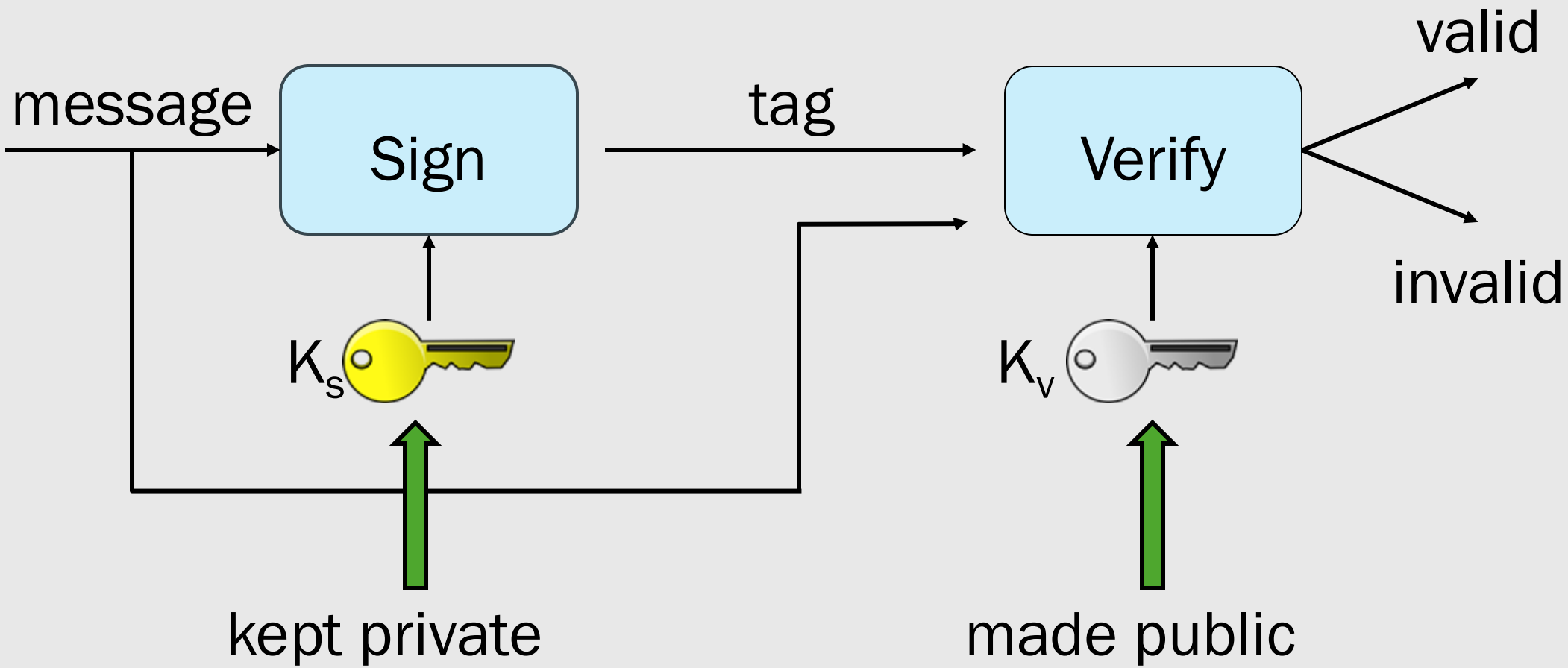
Public Key Signatures

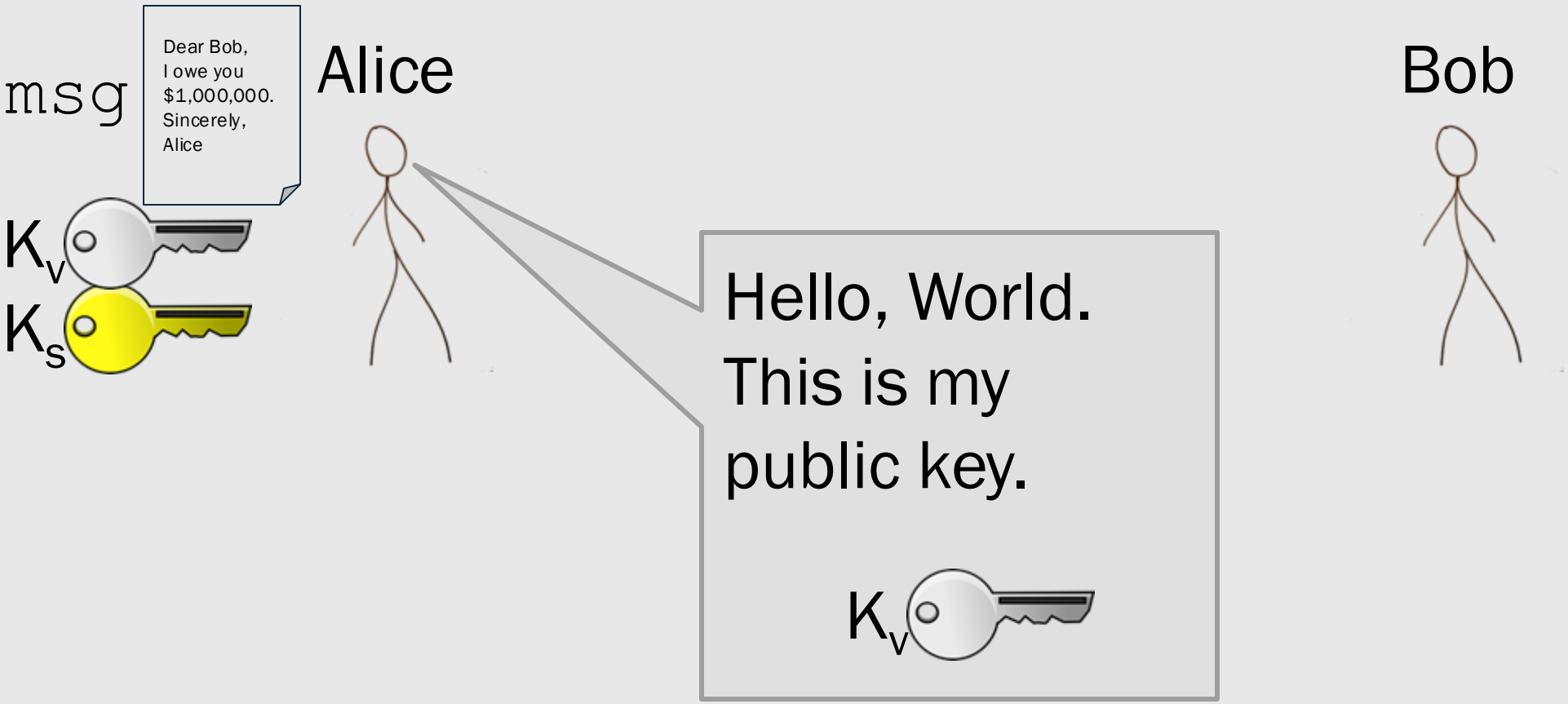


$$tag = Sign_{Ks}(msg)$$

$$isValid = Verify_{Kv}(msg, tag)$$

Public Key Signatures

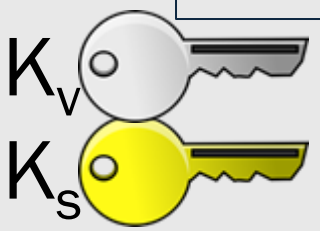



msg
Dear Bob,
I owe you
$1,000,000.
Sincerely,
Alice
Alice
Bob
K_v
K_s
Hello, World.
This is my
public key.
K_v

msg

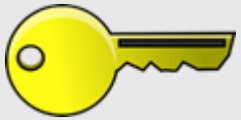
Dear Bob,
I owe you
$1,000,000.
Sincerely,
Alice

Alice



Bob

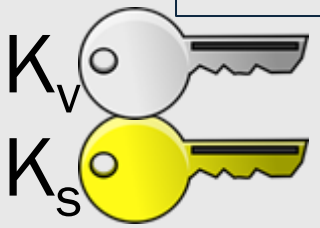
$tag := Sign_{Ks}(msg)$



msg

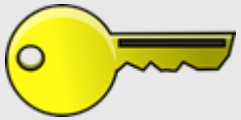
Dear Bob,
I owe you
$1,000,000.
Sincerely,
Alice

Alice

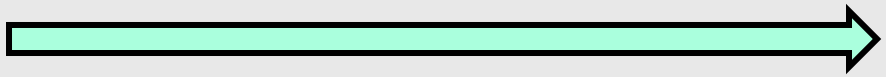


Bob

$tag := Sign_{Ks}(msg)$



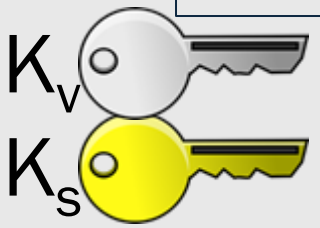
(tag, msg)



msg

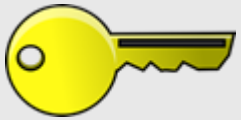
Dear Bob,
I owe you
$1,000,000.
Sincerely,
Alice

Alice

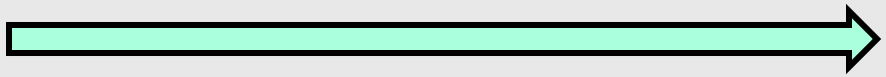


Bob

$tag := Sign_{Ks}(msg)$



(tag, msg)



$isValid = Verify_{Kv}(msg, tag)$

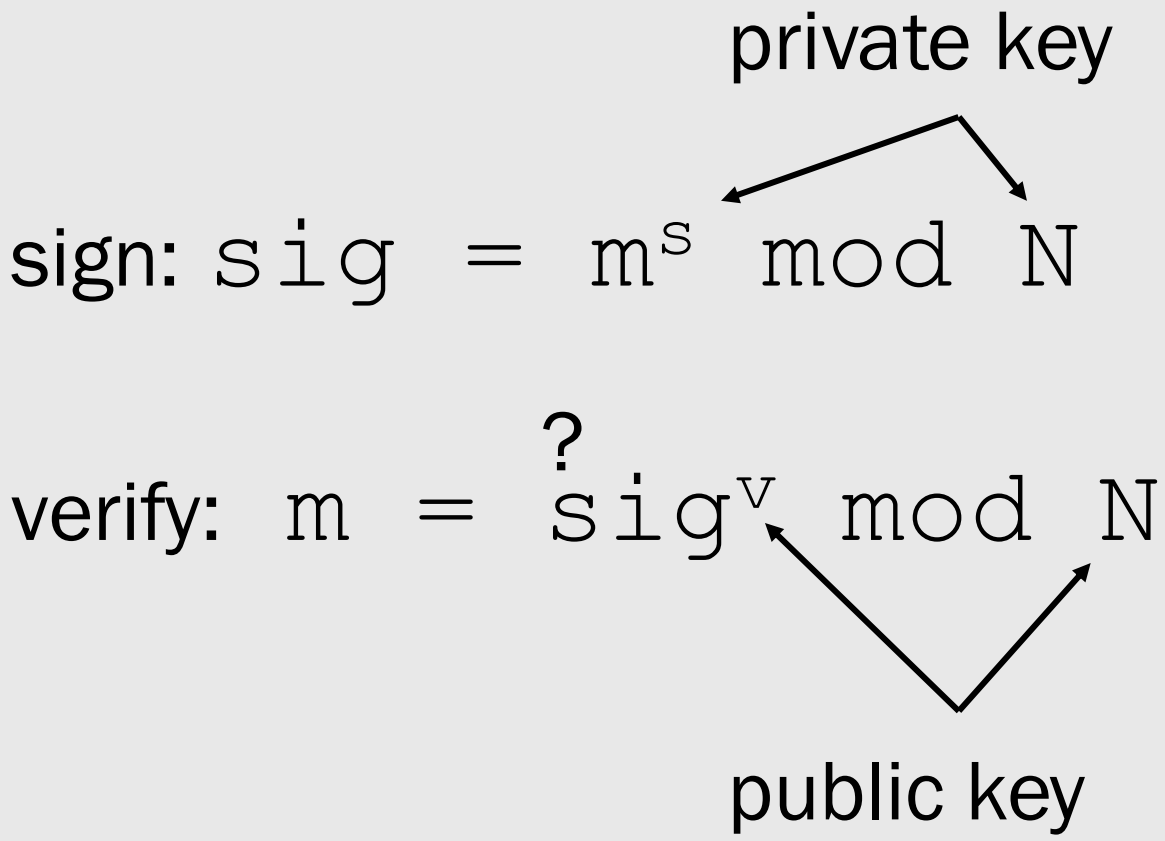
RSA FOR DIGITAL SIGNATURES

RSA for digital signatures

sign: $sig = m^s \bmod N$

verify: $m \stackrel{?}{=} sig^v \bmod N$

RSA for digital signatures



(N, v) → K_v

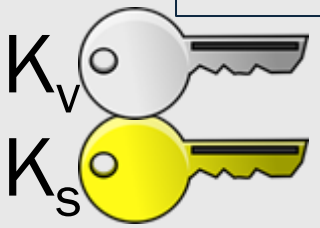
(N, s) → K_s

```
sig := Rand(N)
msg := sig^v mod N
```

RSA does not provide unforgeability

msg

Dear Bob,
I owe you
$1,000,000.
Sincerely,
Alice



Alice

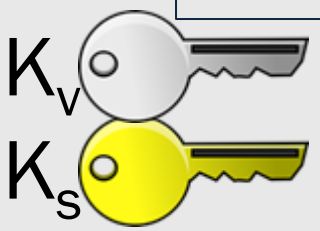
Hello, World.
This is my
public key.

Bob

msg

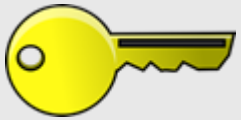
Dear Bob,
I owe you
$1,000,000.
Sincerely,
Alice

Alice



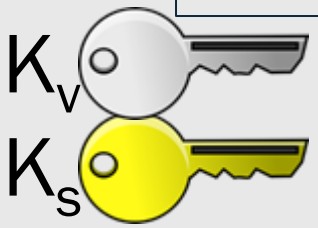
Bob

$tag := Sign_{Ks}(h(msg))$



msg

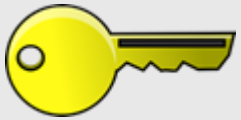
Dear Bob,
I owe you
$1,000,000.
Sincerely,
Alice



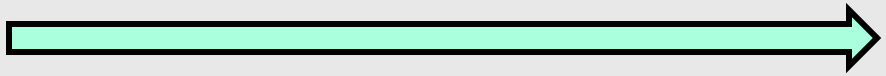
Alice

Bob

$tag := Sign_{Ks}(h(msg))$



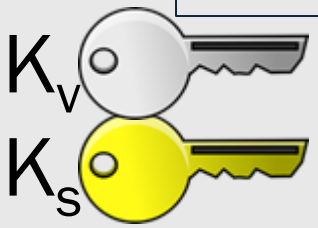
(tag, msg)



msg

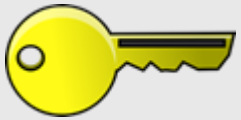
Dear Bob,
I owe you
$1,000,000.
Sincerely,
Alice

Alice

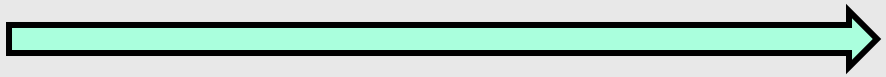


Bob

$tag := Sign_{Ks}(h(msg))$

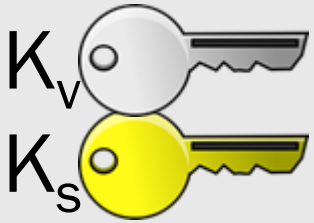


(tag, msg)



$isValid = Verify_{Kv}(h(msg), tag)$

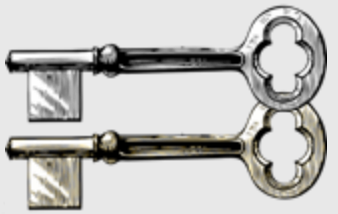
Encrypt then Sign

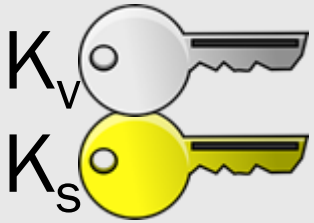


Alice

Hello, World. This is my public key for verifying my signature.

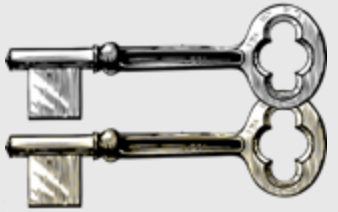
Bob



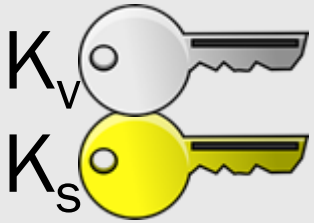


Alice

Bob



Hello, World.
This is my public key for encrypting.

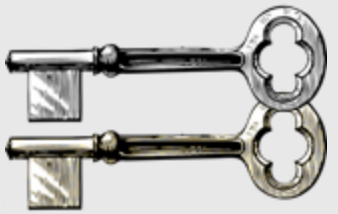


Alice

msg

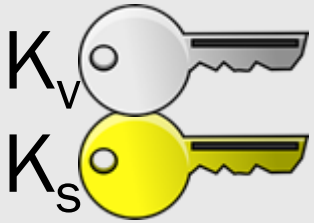
Dear Bob,
I owe you
$1,000,000.
Sincerely,
Alice

Bob



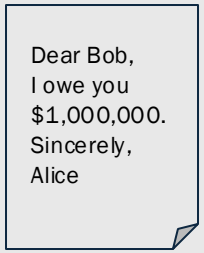
$$c := Enc_{Ke}(msg)$$

$$tag := Sign_{ks}(c)$$

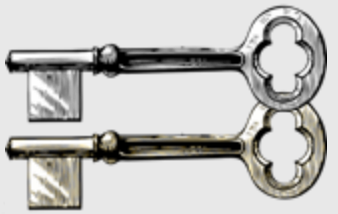


Alice

msg



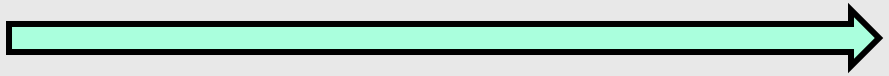
Bob

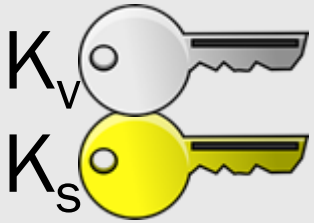


$c := Enc_{Ke}(msg)$

$tag := Sign_{ks}(c)$

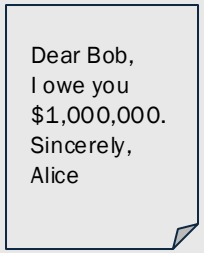
(c,tag)



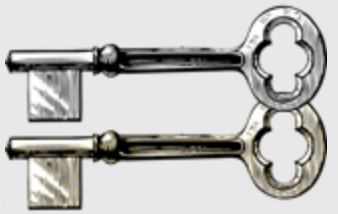


Alice

msg



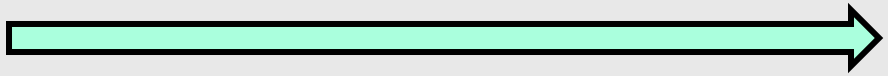
Bob



$c := Enc_{Ke}(msg)$

$tag := Sign_{ks}(c)$

(c,tag)



$isValid := Verify_{Kv}(c,tag)$

$msg := Dec_{kd}(c)$

A Note of Caution

- Beware of determinism
- Use unique keys for encryption, signing, generating MACs
- Use appropriate algorithms for the desired property
- Do not roll your own

ESTABLISHING A SHARED SECRET KEY! ☺

Key Establishment

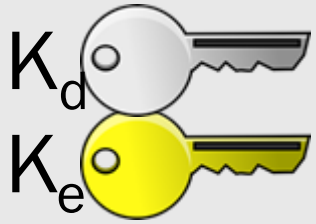
Def’n: a means of agreeing on a shared secret

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

HYBRID ENCRYPTION FOR KEY ESTABLISHMENT

k msg

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



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

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

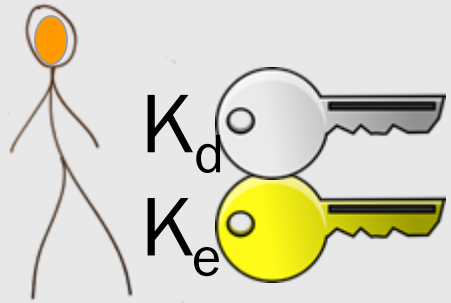
c_1, c_2

$k = PK\text{-}Dec_{Kd}(c_1)$

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

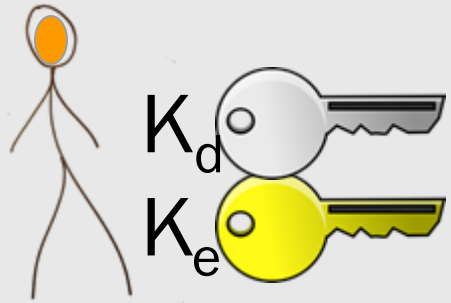
Hybrid Encryption

I’m Purple. Let’s use: RSA-$\mathrm{Enc}_{Ke}(S)$

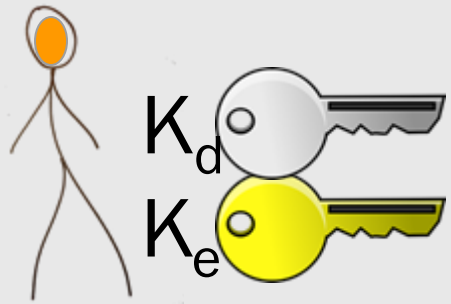


Hybrid Encryption: Replay Attack

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



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



Nonce

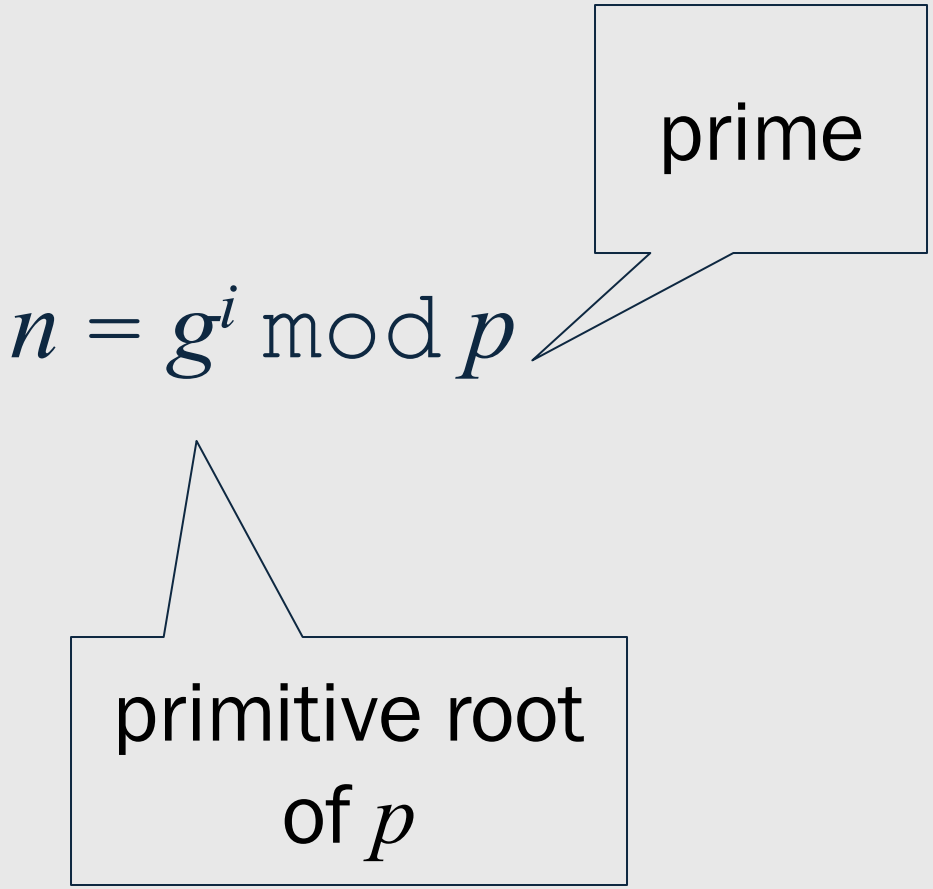
Def’n: a number used only once as part of a cryptographic protocol

- 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



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

g, p

$a < p$

$A := g^a \bmod p$

$b < p$

$B := g^b \bmod p$

A

B

$K := B^a \bmod p$

$= (g^b)^a \bmod p$

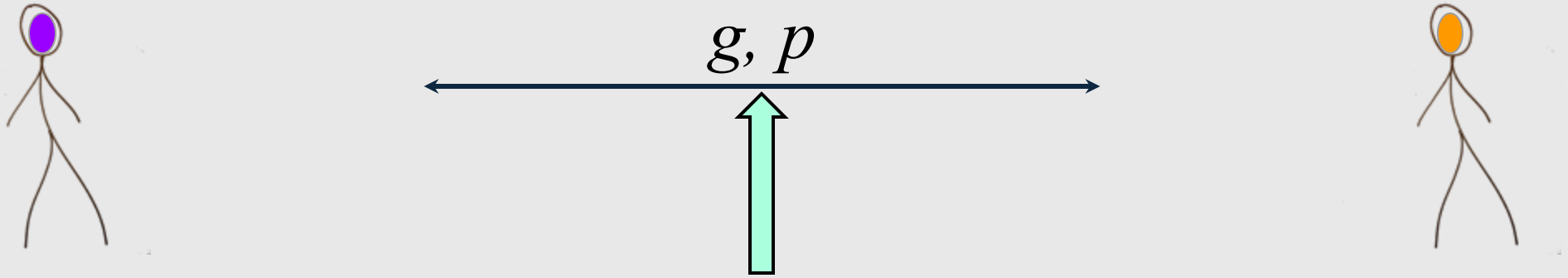
$= g^{ba} \bmod p$

$K := A^b \bmod p$

$= (g^a)^b \bmod p$

$= g^{ab} \bmod p$

Diffie-Hellman Key Agreement



Diffie-Hellman Key Agreement

g, p

$a < p$

$A := g^a \bmod p$

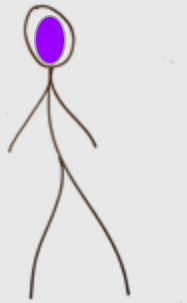
$b < p$

$B := g^b \bmod p$

A

B

Diffie-Hellman Key Agreement



g, p

$a < p$

$A := g^a \bmod p$

$b < p$

$B := g^b \bmod p$

A

B

$K := B^a \bmod p$

$= (g^b)^a \bmod p$

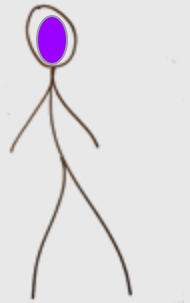
$= g^{ba} \bmod p$

$K := A^b \bmod p$

$= (g^a)^b \bmod p$

$= g^{ab} \bmod p$

Diffie-Hellman Key Agreement



g, p

$a < p$

$A := g^a \bmod p$

$b < p$

$B := g^b \bmod p$

A

B

$K := B^a \bmod p$

$= (g^b)^a \bmod p$

$= g^{ba} \bmod p$

$K := A^b \bmod p$

$= (g^a)^b \bmod p$

$= g^{ab} \bmod p$

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 $F: X \longrightarrow Y$ 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 mod 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

$B := g^b \bmod p$

A →

← D

C →

← B

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

K_{ad}

K_{cb}

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

Adversary-in-the-Middle attack on DH

No authentication ☹

a

$A := g^a \bmod p$

b

$B := g^b \bmod p$

D

B

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

K_{ad}

K_{cb}

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

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

$g=2,\ p=11$

$a = 3$

$A = \ ?$

$b = 5$

$B = \ ?$

A

B

$K \ = \ ?$

$K \ = \ ?$