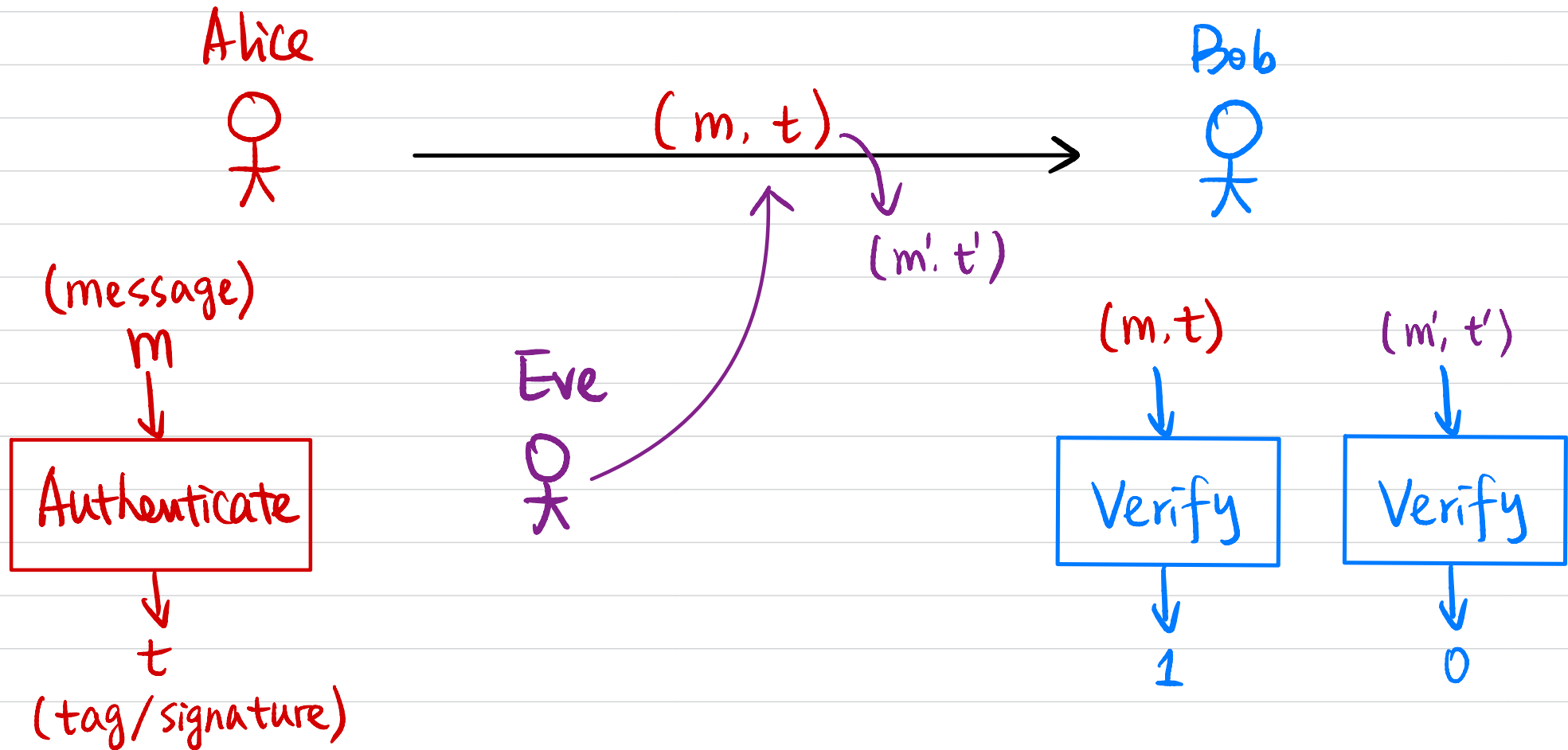


CSCI 1515 Applied Cryptography

This Lecture:

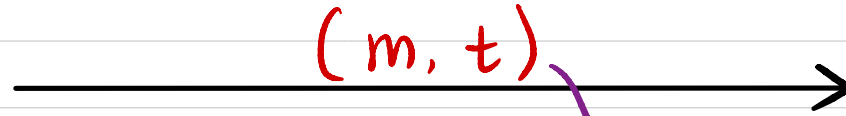
- Message Integrity & CMA Security
- RSA Signature Scheme
- Authenticated Encryption
- Cryptographic Hash Function

Message Integrity



Message Authentication Code (MAC)

Alice



Bob



(m', t')

Eve



(message)

m

k



t

(tag)

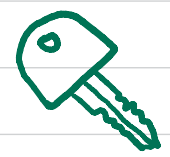


(m, t)

k



0/1



Digital Signature

Alice



(message)

m

sk

Authenticate



σ

(signature)



(secret)

(m, σ)

(m', σ')

Eve



Bob



(m, σ)

pk

Verify



0/1



(public)

Syntax

Message Authentication Code (MAC) Scheme $\Pi = (\text{Gen}, \text{Mac}, \text{Vrfy})$

$$k \leftarrow \text{Gen}(1^\lambda)$$

$$t \leftarrow \text{Mac}_k(m)$$

$$0/1 := \text{Vrfy}_k(m, t)$$

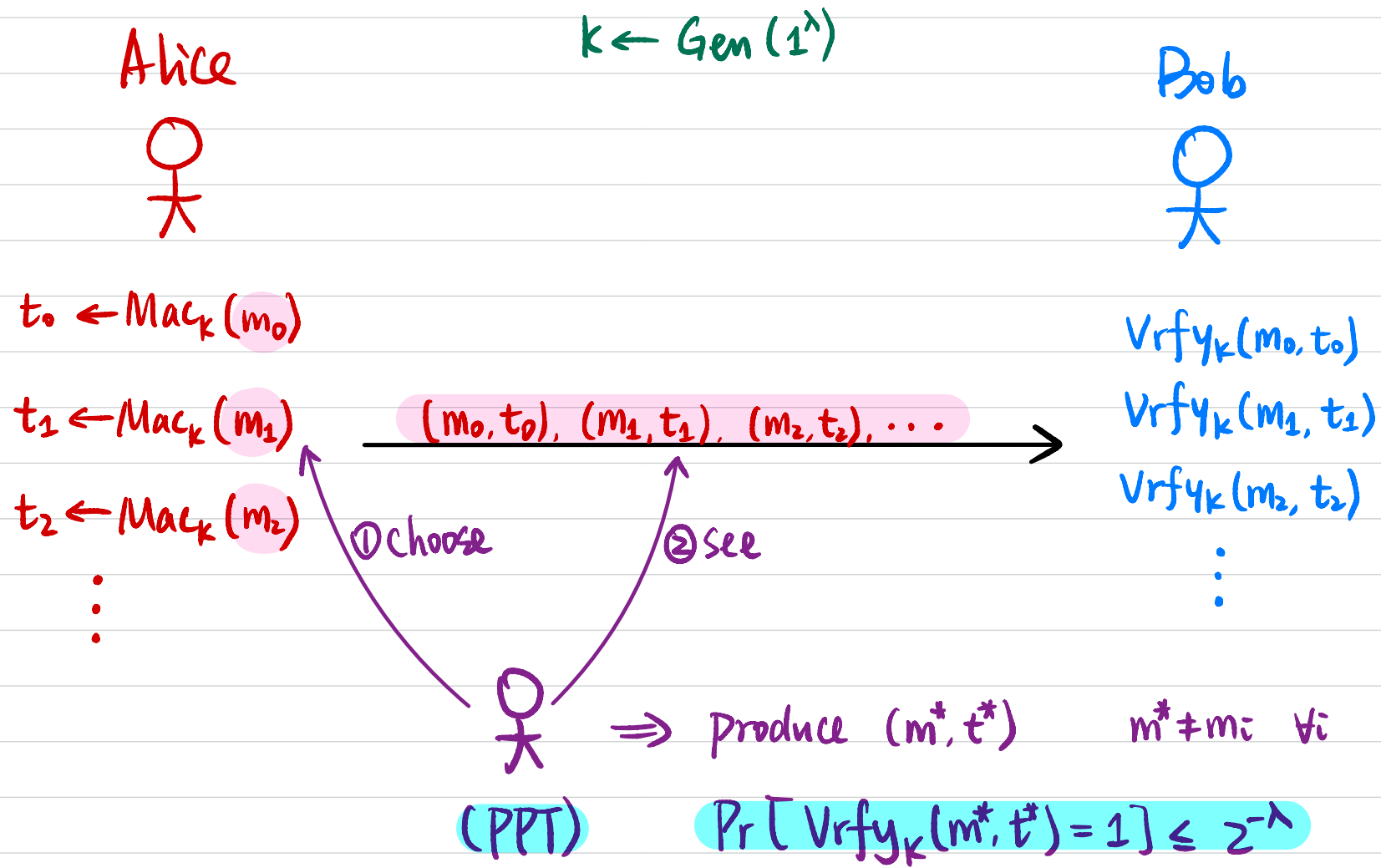
Digital Signature Scheme $\Pi = (\text{Gen}, \text{Sign}, \text{Vrfy})$

$$(pk, sk) \leftarrow \text{Gen}(1^\lambda)$$

$$\sigma \leftarrow \text{Sign}_{sk}(m)$$

$$0/1 := \text{Vrfy}_{pk}(m, \sigma)$$

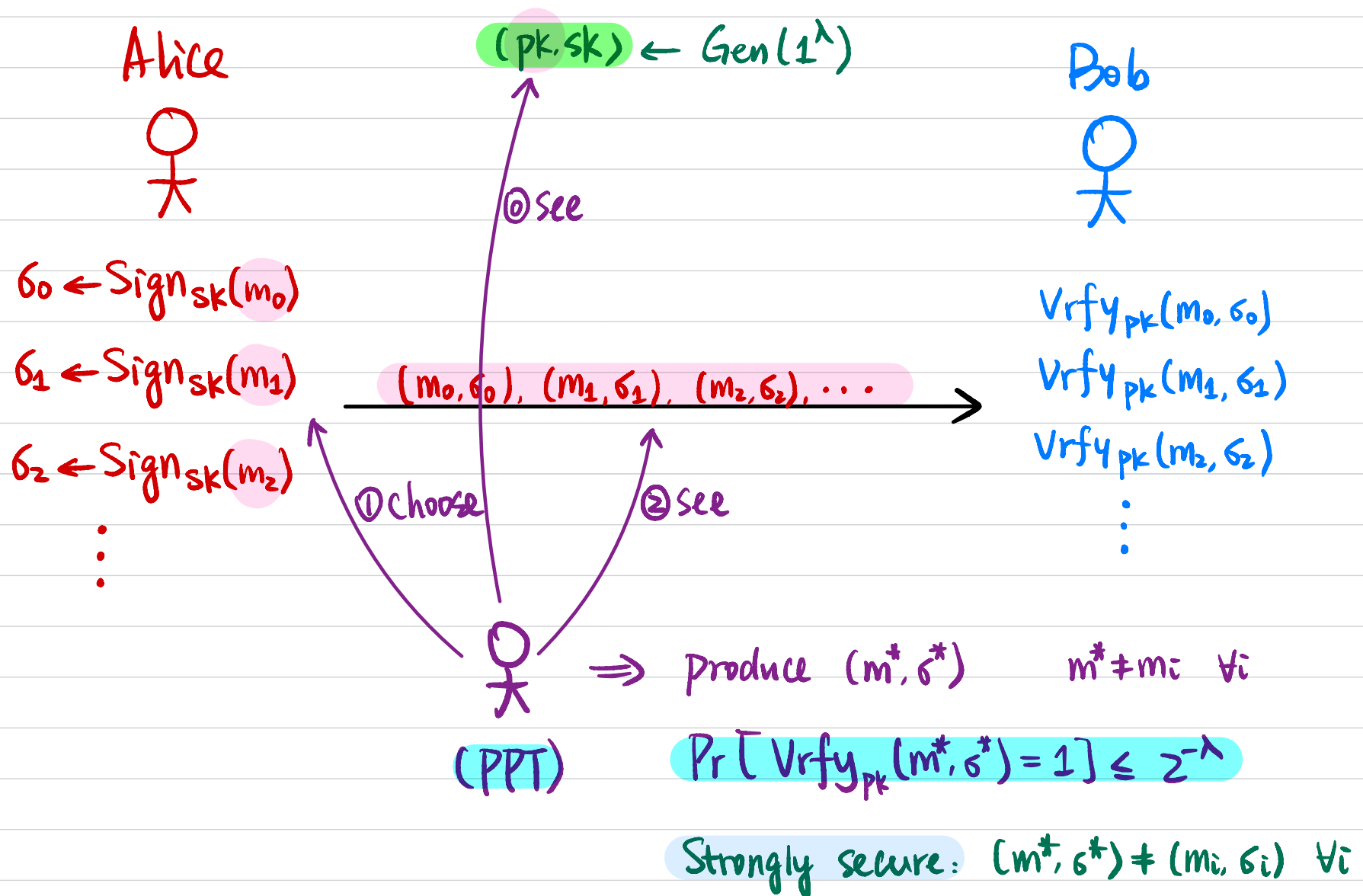
Chosen-Message Attack (CMA)



Strongly secure: $(m^*, t^*) \neq (m_i, t_i) \quad \forall i$

Does Mac have to be randomized?

Chosen-Message Attack (CMA)



Does Sign have to be randomized?

RSA Signature

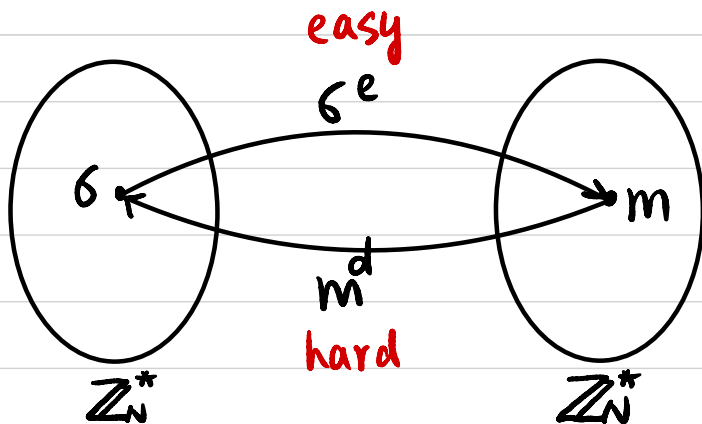
Generate two n -bit primes p, q ($p \neq q$)

Compute $N = p \cdot q$, $\phi(N) = (p-1)(q-1)$

Choose e s.t. $\gcd(e, \phi(N)) = 1$

Compute $d = e^{-1} \bmod \phi(N)$.

Given (N, e) & a random $y \xleftarrow{\$} \mathbb{Z}_N^*$, it's computationally hard to find x s.t.
 $x^e \equiv y \pmod{N}$



$SK = d$ $PK = (N, e)$

$Sign_{sk}(m) = m^d \bmod N$

$Verfy_{pk}(m, \sigma): \sigma^e \stackrel{?}{\equiv} m \pmod{N}$

CMA Secure?

$\sigma^* \in \mathbb{Z}_N^*$
 $m^* := (\sigma^*)^e$
output (m^*, σ^*)

$m_0 \rightarrow \sigma_0$

$m^* := m_0^2$

$\sigma^* := \sigma_0^2$

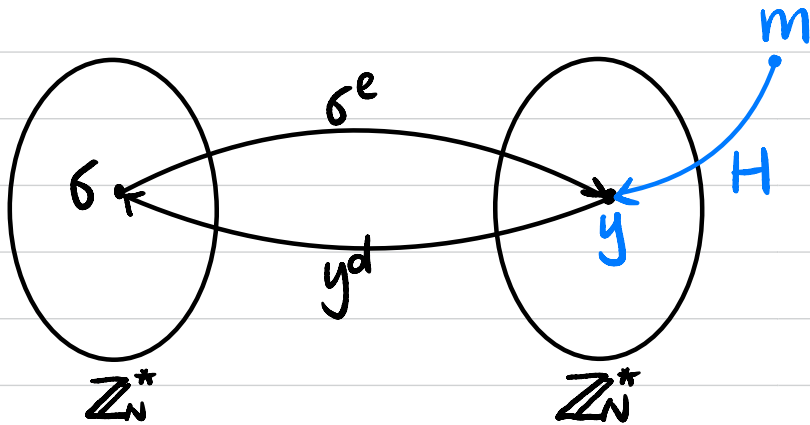
$m_1 \rightarrow \sigma_1$

$m^* := m_0 \cdot m_1$

$\sigma^* := \sigma_0 \cdot \sigma_1$

RSA Signature

Given (N, e) & a random $y \leftarrow \mathbb{Z}_N^*$, it's computationally hard to find x s.t.
 $x^e \equiv y \pmod{N}$



$$sk = d \quad pk = (N, e)$$

$$\text{Sign}_{sk}(m) = H(m)^d \pmod{N}$$

$$\text{Vrfy}_{pk}(m, \sigma): \sigma^e \stackrel{?}{\equiv} H(m) \pmod{N}$$

$$H: \{0, 1\}^* \rightarrow \mathbb{Z}_N^*$$

↑
public, deterministic function
that gives a (pseudo)random output

CMA Secure? Yes!

Constructions for Digital Signature

RSA Signature : RSA Assumption

Schnorr / DSA Signature : Discrete Logarithm Assumption

Lattice-Based Signature Schemes (Post-Quantum Security)

Constructions for MAC

From block cipher: CBC-MAC

From hash function: HMAC

Computational Assumption: "The construction is secure"

Summary

Symmetric-Key

Public-Key

Message
Secrecy

Primitive: SKE

Construction: block cipher

Primitive: PKE

Constructions: RSA / ElGamal

Message
Integrity

Primitive: MAC

Constructions: CBC-MAC / HMAC

Primitive: Signature

Constructions: RSA / DSA

Key Exchange

Construction: Diffie-Hellman

Important
Tool

Primitive: Hash function

Construction: SHA

What happens in practice

Alice



k

Diffie-Hellman Key Exchange

Bob



k

Symmetric-Key Encryption



Message Secrecy & Integrity?

Authenticated Encryption (AE)

Def An authenticated encryption scheme is a symmetric-key encryption that is CCA-secure & unforgeable.

Construction

CPA-secure SKE
+
CMA-secure MAC $\Rightarrow$ AE $\begin{cases} \text{CCA-secure} \\ \text{Unforgeable} \end{cases}$

- ① Encrypt-and-MAC
- ② MAC-then-Encrypt
- ③ Encrypt-then-MAC

Chosen-Ciphertext Attack (CCA) Security

$k \leftarrow \text{Gen}(1^\lambda)$

Alice



$C_0 \leftarrow \text{Enc}_k(m_0)$

$C_1 \leftarrow \text{Enc}_k(m_1)$

$C_2 \leftarrow \text{Enc}_k(m_2)$

$\vdots$

$m_0, m_1 \in \{0, 1\}^n$

$b \leftarrow \{0, 1\}$

$C \leftarrow \text{Enc}_k(m_b)$

① Choose

② See

Choose

See

④ See

③ Choose

(PPT)

guess b'

$$\Pr[b' = b] \leq \frac{1}{2} + 2^{-\lambda}$$

Bob



$m_0 := \text{Dec}_k(C_0)$

$m_1 := \text{Dec}_k(C_1)$

$m_2 := \text{Dec}_k(C_2)$

$\vdots$

$m'_0 := \text{Dec}_k(C'_0)$

$m'_1 := \text{Dec}_k(C'_1)$

$m'_2 := \text{Dec}_k(C'_2)$

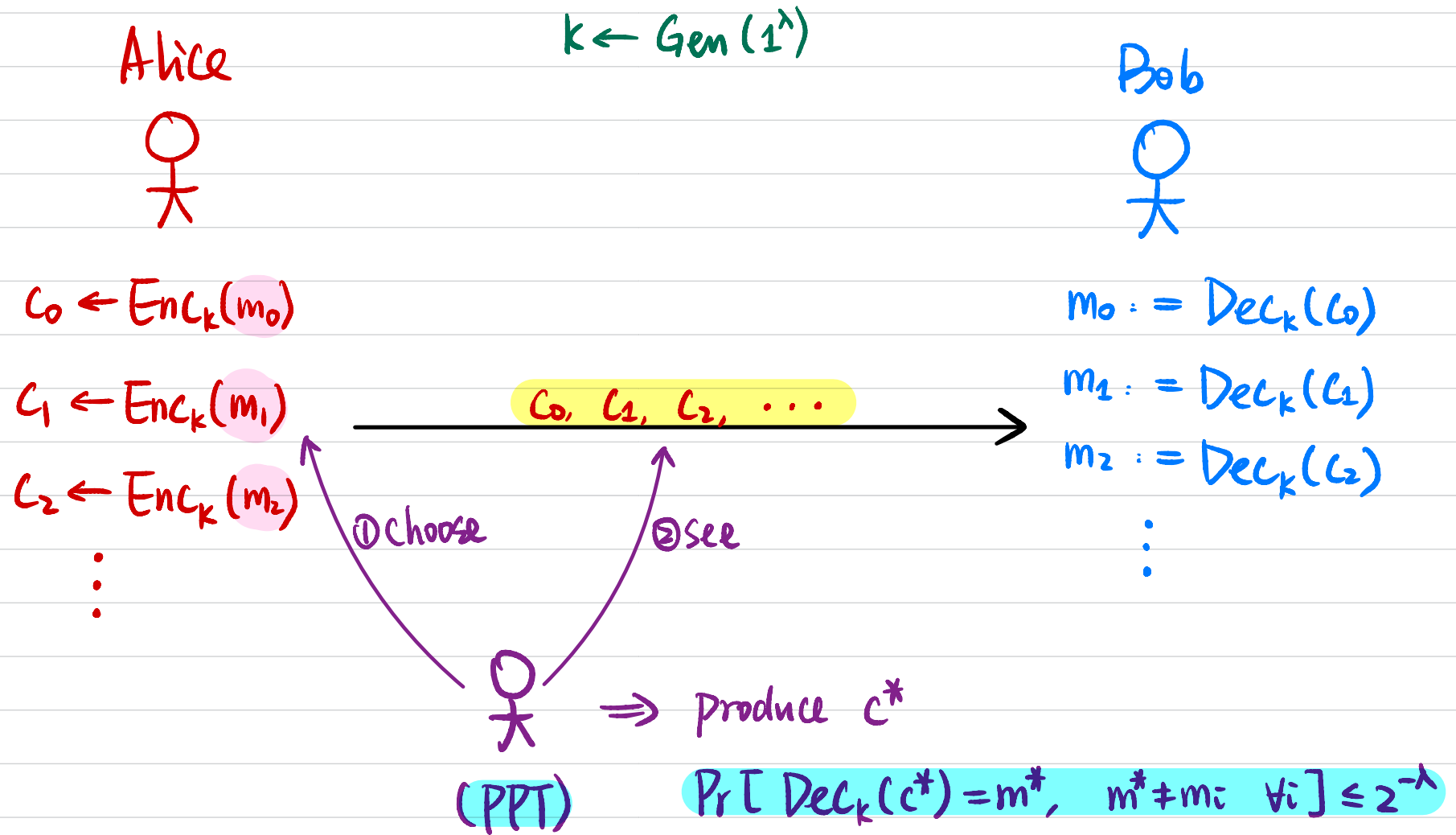
$\vdots$

$C_0, C_1, C_2, \dots$

C

$C'_0, C'_1, C'_2, \dots (\neq C)$

Unforgeability



Encrypt-and-MAC

Given a CPA-secure SKE scheme $\Pi_1 = (\text{Gen}_1, \text{Enc}_1, \text{Dec}_1)$

& a strongly CMA-secure MAC scheme $\Pi_2 = (\text{Gen}_2, \text{Mac}_2, \text{Vrfy}_2)$

Construct an AE scheme $\Pi = (\text{Gen}, \text{Enc}, \text{Dec})$:

Gen(1^λ):

$k_1 \leftarrow \text{Gen}_1(1^\lambda)$

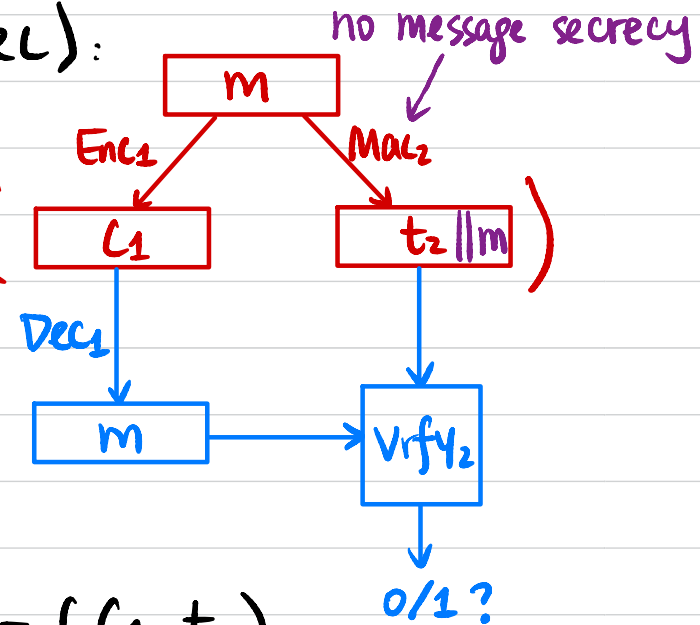
$k_2 \leftarrow \text{Gen}_2(1^\lambda)$

Output $k = (k_1, k_2)$

Unforgeable:

$C^* = (c_1^*, t_2^*)$
 $\downarrow$
 $m^* \rightarrow 0$

$C = (c_1, t_2 \parallel m)$



Enc_k(m):

$c_1 \leftarrow \text{Enc}_1(k_1, m)$

$t_2 \leftarrow \text{Mac}_2(k_2, m)$

output $C = (c_1, t_2)$

Dec_k(C): $C = (c_1, t_2)$

$m := \text{Dec}_1(k_1, c_1)$

$b := \text{Vrfy}_2(k_2, (m, t_2))$

If $b=1$, output m
Otherwise output $\perp$

MAC-then-Encrypt

Gen(1^λ):

$$k_1 \leftarrow \text{Gen}_1(1^\lambda)$$

$$k_2 \leftarrow \text{Gen}_2(1^\lambda)$$

Output $k = (k_1, k_2)$

Enc_k(m):

$$t_2 \leftarrow \text{Mac}_2(k_2, m)$$

$$c_1 \leftarrow \text{Enc}_1(k_1, m \parallel t_2)$$

output $C = c_1$

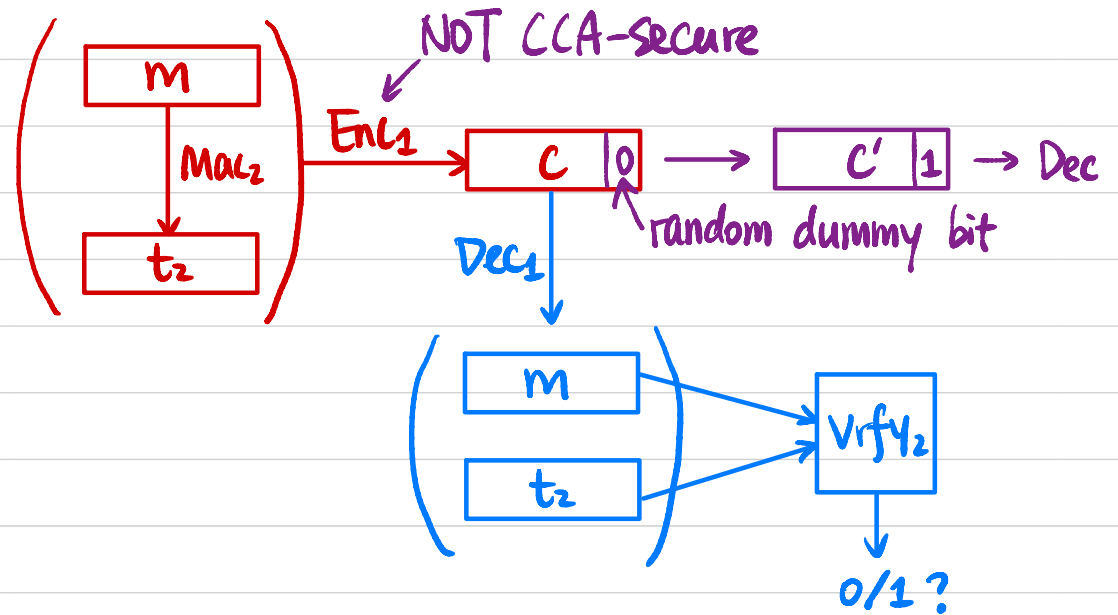
Dec_k(C): $C = c_1$

$$m \parallel t_2 := \text{Dec}_1(k_1, c_1)$$

$$b := \text{Vrfy}_2(k_2, (m, t_2))$$

If $b=1$, output m

Otherwise output $\perp$



Encrypt-then-MAC

Gen(1^λ):

$k_1 \leftarrow \text{Gen}_1(1^\lambda)$

$k_2 \leftarrow \text{Gen}_2(1^\lambda)$

Output $k = (k_1, k_2)$

Enc $_k(m)$:

$c_1 \leftarrow \text{Enc}_1(k_1, m)$

$t_2 \leftarrow \text{Mac}_2(k_2, c_1)$

output $C = (c_1, t_2)$

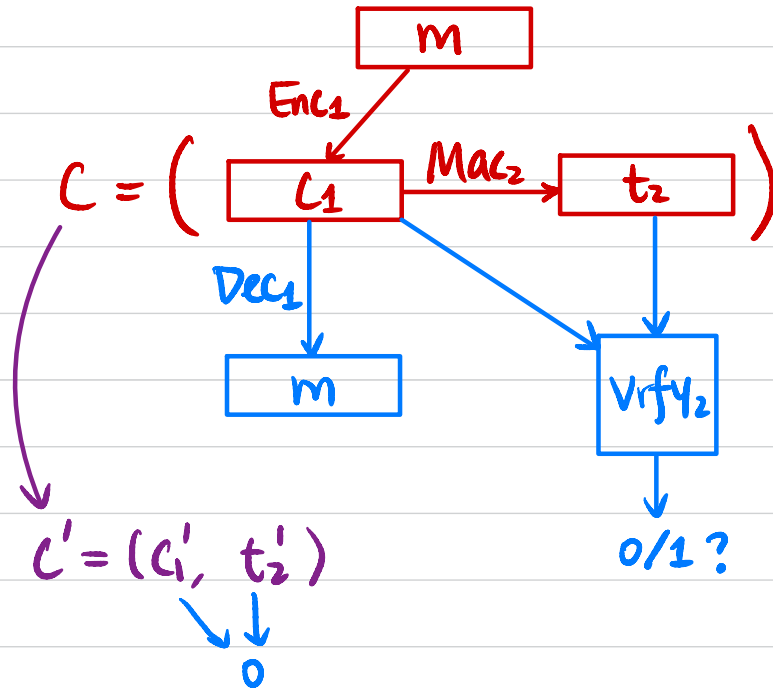
Dec $_k(C)$: $C = (c_1, t_2)$

$m := \text{Dec}_1(k_1, c_1)$

$b := \text{Vrfy}_2(k_2, (c_1, t_2))$

If $b=1$, output m

Otherwise output $\perp$



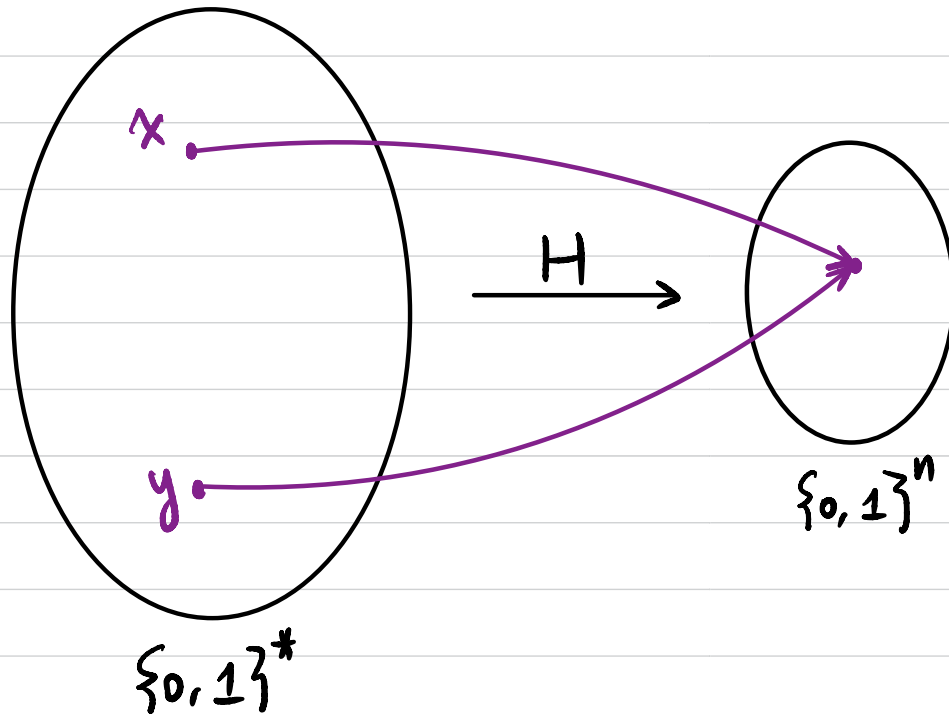
Summary

	Symmetric-Key	Public-Key
Message Secrecy	Primitive: SKE Construction: block cipher	Primitive: PKE Constructions: RSA / ElGamal
Message Integrity	Primitive: MAC Constructions: CBC-MAC / HMAC	Primitive: Signature Constructions: RSA / DSA
Secrecy & Integrity	Primitive: AE Construction: Encrypt-then-MAC	
Key Exchange		Construction: Diffie-Hellman
Important Tool	Primitive: Hash function Construction: SHA	

Hash Function

$$H: \{0,1\}^* \rightarrow \{0,1\}^n \quad \theta(\lambda)$$

$$\lambda = 128$$
$$n = 256$$



Collision-Resistant Hash Function (CRHF):

It's computationally hard to find $x, y \in \{0,1\}^*$ s.t.

$$x \neq y, \quad H(x) = H(y) \quad (\text{collision})$$