

CSCI 1515 Applied Cryptography

This Lecture:

- CBC-MAC
- Password-Based Authentication
- Two-Factor Authentication (2FA)

Summary

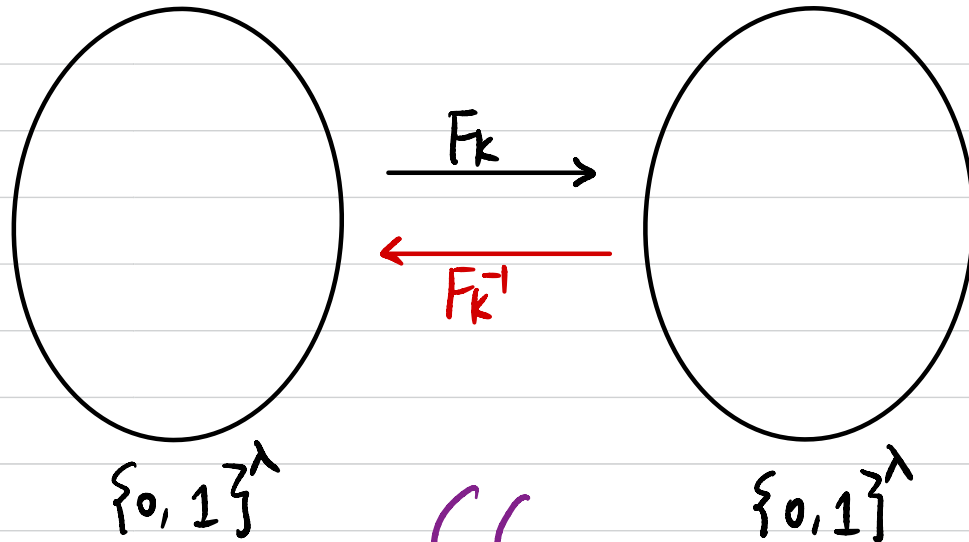
	Symmetric-Key	Public-Key
Message Secrecy	Primitive: SKE Construction: block cipher	Primitive: PKE Constructions: RSA / ElGamal
Message Integrity	Primitive: MAC Constructions: CBC-MAC / HMAC	Primitive: Signature Constructions: RSA / DSA
Secrecy & Integrity	Primitive: AE Construction: Encrypt-then-MAC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

Block Cipher

$F: \{0, 1\}^\lambda \times \{0, 1\}^\lambda \rightarrow \{0, 1\}^\lambda$ ^{← 128} pseudorandom permutation (PRP).

$$k \leftarrow \{0, 1\}^\lambda$$

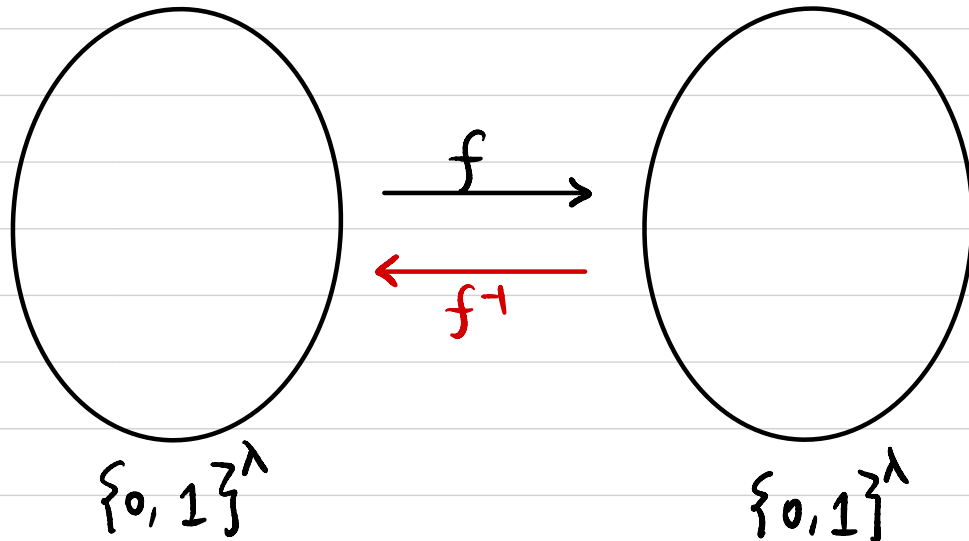
$F_k:$



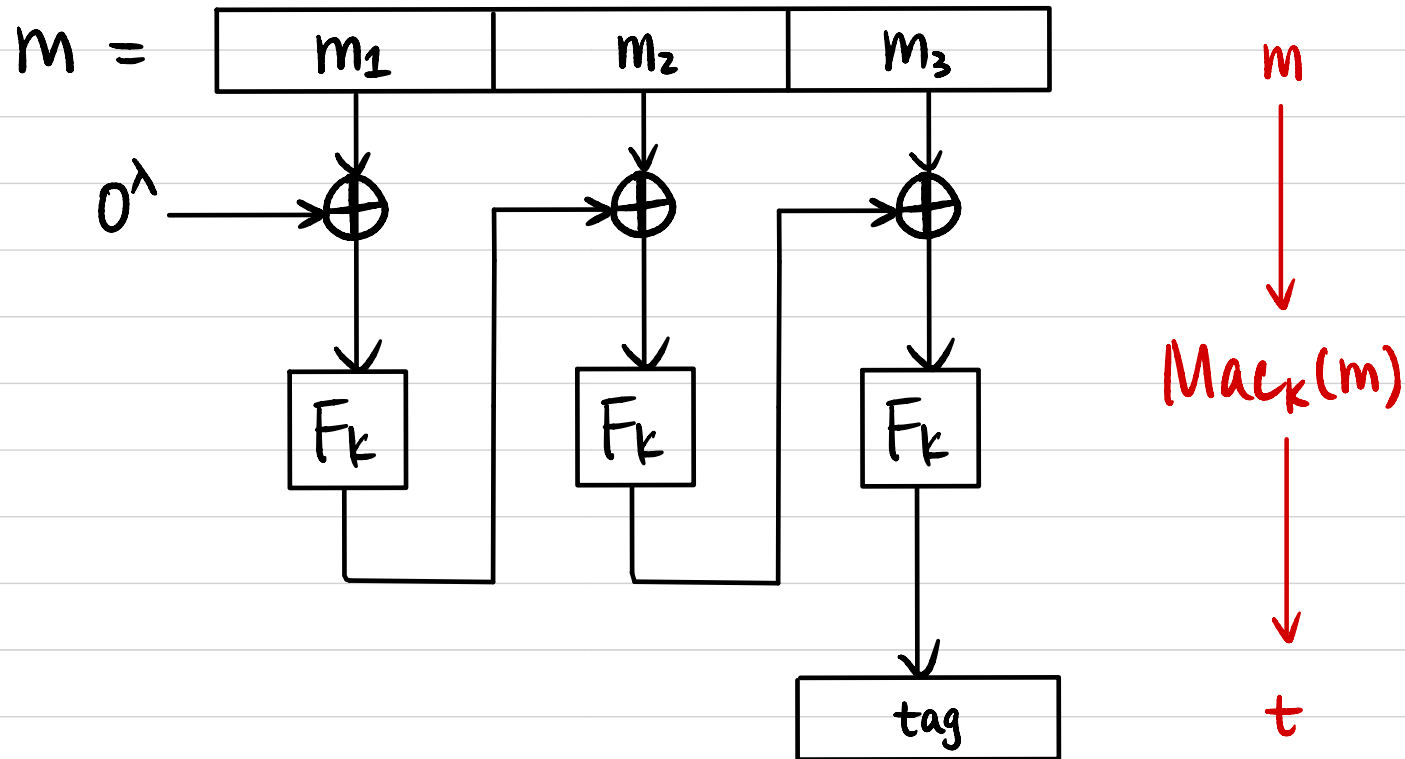
$\mathcal{F}$

$$f \leftarrow \{ F \mid F: \{0, 1\}^\lambda \rightarrow \{0, 1\}^\lambda, \\ F \text{ is bijective} \}$$

$f:$



CBC-MAC

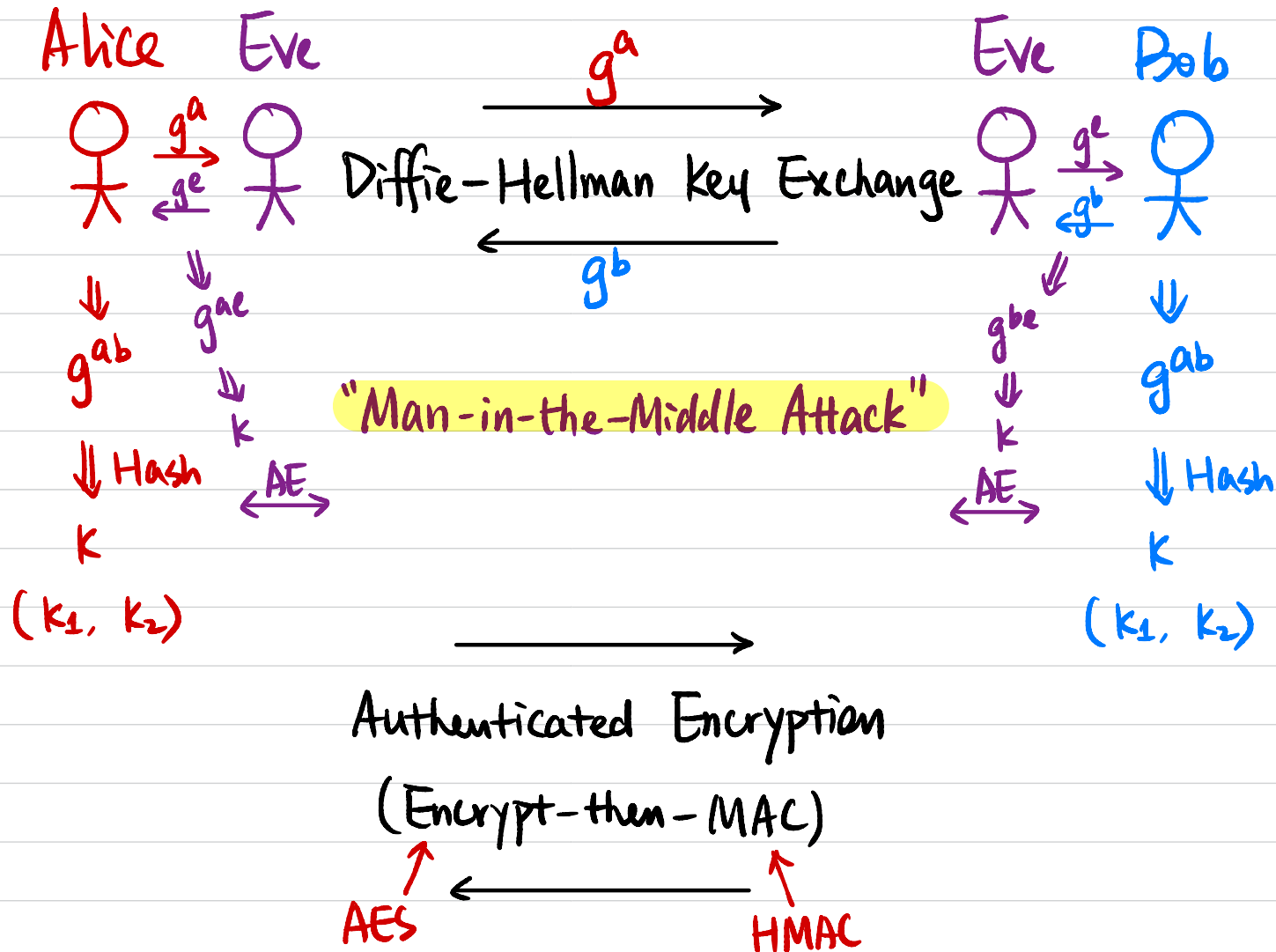


How to verify? $Vrfy_k(m, t): Mac_k(m) \stackrel{?}{=} t$

CMA (Chosen Message Attack) Secure?

- Fixed-length messages of length $3 \cdot \lambda$ Yes!
- Arbitrary-length messages No! Query $m_1 \rightarrow$ Get t_1
Query $t_1 \rightarrow$ Get t_2
Output $m^* = m_1 || 0^\lambda, t^* = t_2$

Putting it All Together: Secure Communication



Any security issue?

Authenticated Key Exchange

Signature Scheme

public
 $(VK_A, SK_A) \leftarrow \text{Gen}(1^\lambda)$

public
 $(VK_B, SK_B) \leftarrow \text{Gen}(1^\lambda)$

Alice

$$\sigma_A \leftarrow \text{Sign}_{SK_A}(g^a)$$



$$g^a, \sigma_A$$



$$\text{Vrfy}_{VK_A}(g^a, \sigma_A) \stackrel{?}{=} 1$$

Diffie-Hellman Key Exchange



$$g^b, \sigma_B$$

$$\sigma_B \leftarrow \text{Sign}_{SK_B}(g^b)$$

Bob



$$g^{ab}$$

$$\text{Vrfy}_{VK_B}(g^b, \sigma_B) \stackrel{?}{=} 1$$

Hash

K

$$(K_1, K_2)$$

$$g^{ab}$$

Hash

K

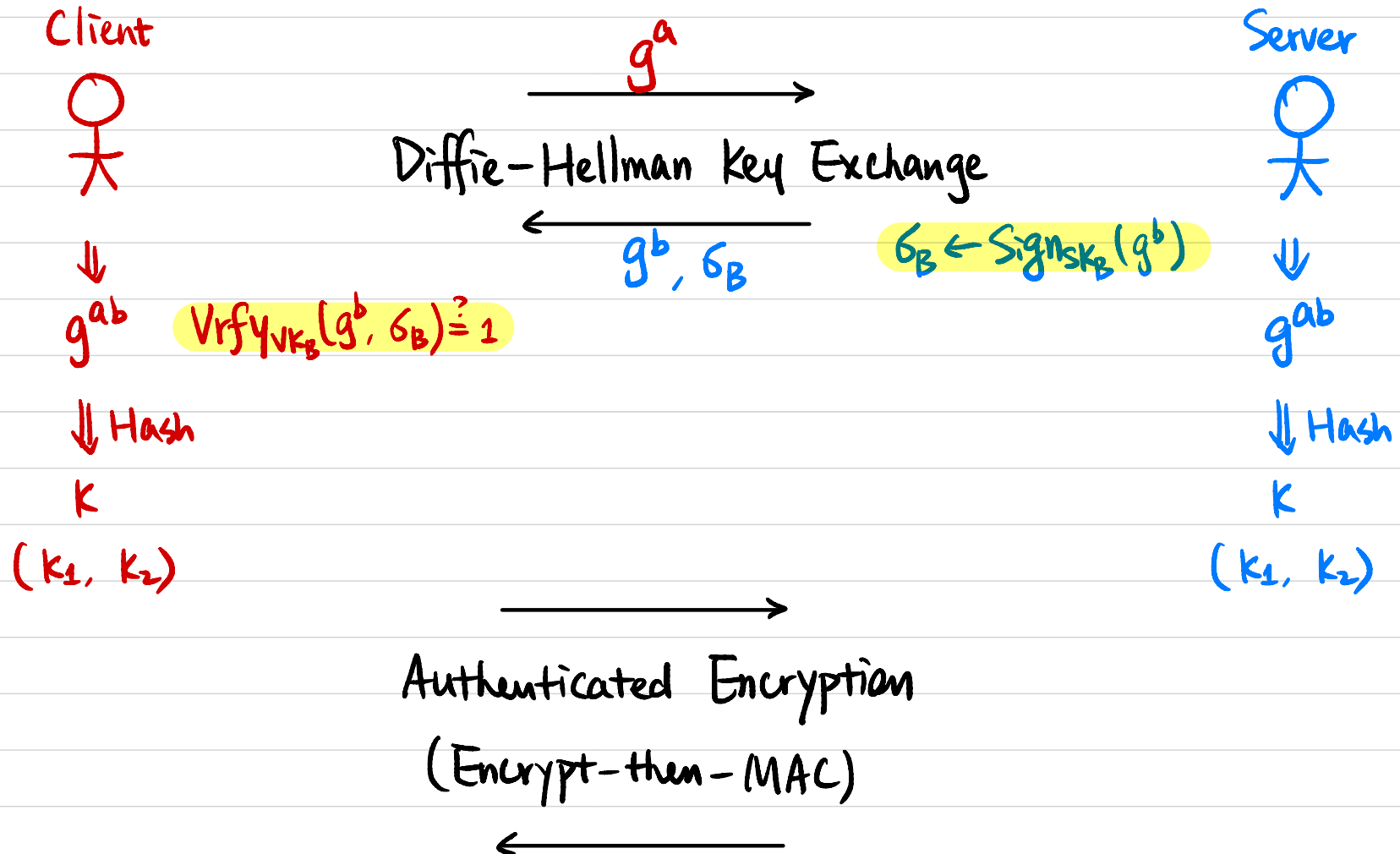
$$(K_1, K_2)$$

Authenticated Encryption

(Encrypt-then-MAC)

One-Sided Secure Authentication

public
↓
 $(VK_B, SK_B) \leftarrow \text{Gen}(1^\lambda)$



Password-Based Authentication

User



Server



ID, password

$h = H(\text{password})$

Signup

ID, h
(Encrypted)

Store (ID, h) (Encrypted)

Login

ID, h
(Encrypted)

check

Password Security ?

Online Dictionary Attack

User



Server



ID, password

$h = H(\text{password})$

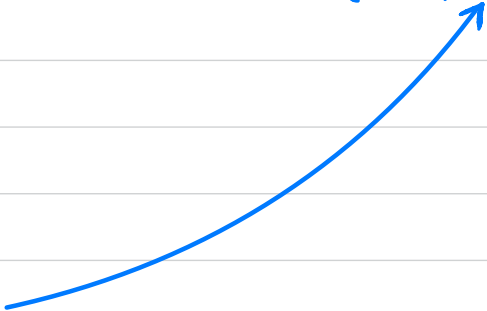
Signup

ID, h
(Encrypted)

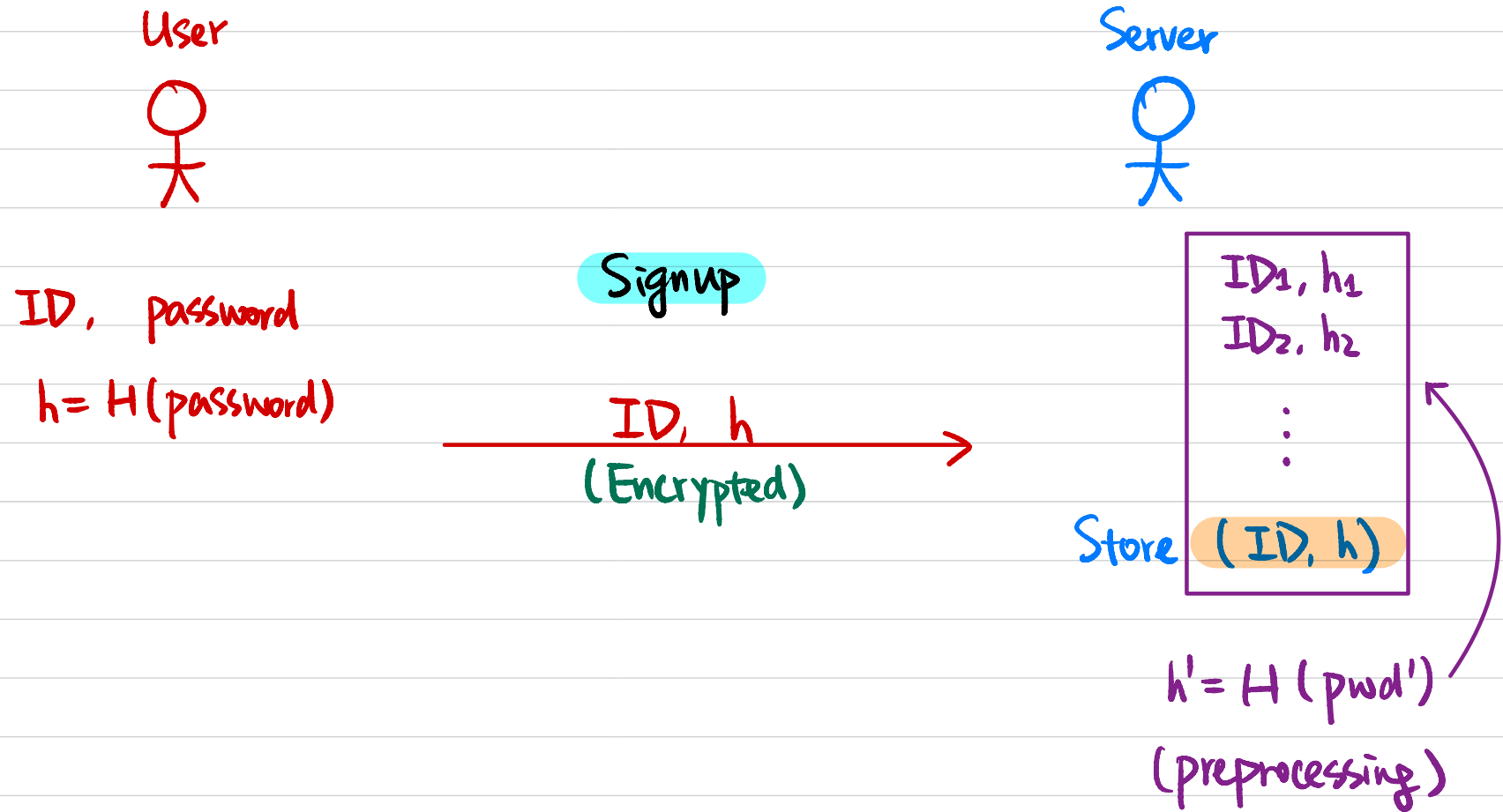
Store (ID, h) (Encrypted)

Login

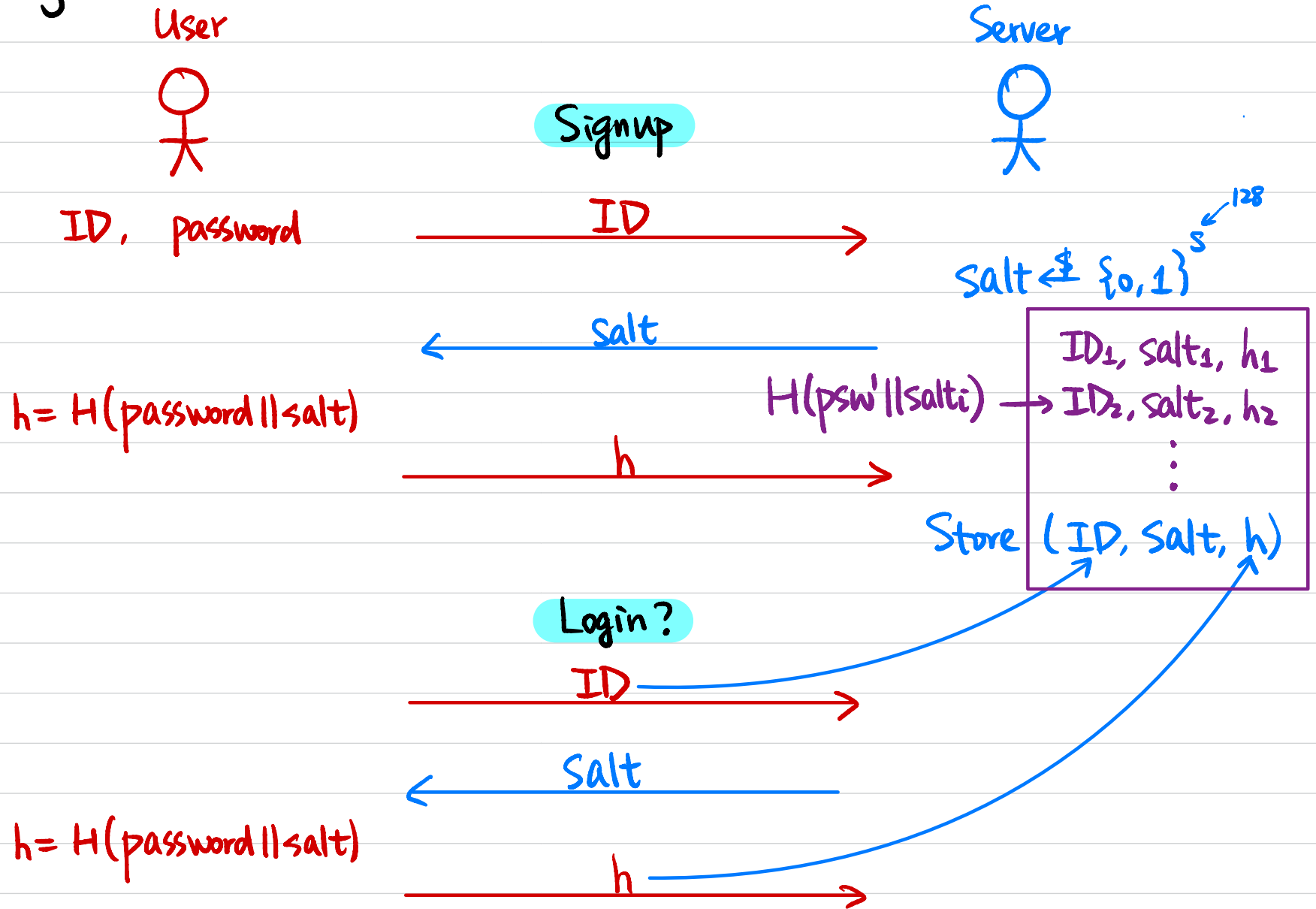
ID, $h' = H(\text{pwd}')$
(Encrypted)



Offline Dictionary Attack



Salting



Why does it help?

Salt & Pepper

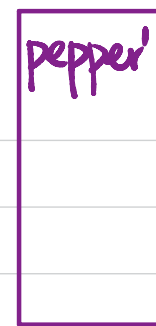
User



ID, password

Signup

$$H(H(\text{psw}' \parallel \text{salt}_i) \parallel \text{pepper}')$$



256

Server



$ID_1, \text{salt}_1, h_1^*$
 $ID_2, \text{salt}_2, h_2^*$
 $\vdots$

ID

salt

$$h = H(\text{password} \parallel \text{salt})$$

h

$$\text{salt} \leftarrow \{0,1\}^S \quad S \leftarrow 128$$

$$\text{pepper} \leftarrow \{0,1\}^P \quad P \leftarrow 8$$

$$h^* = H(h \parallel \text{pepper})$$

Store (ID, salt, h^*)

Login?

ID

salt

$$h = H(\text{password} \parallel \text{salt})$$

h

$$\forall \text{pepper}' \in \{0,1\}^8 \quad 2^8 = 256$$

$$h' := H(h \parallel \text{pepper}')$$

Why does it help?

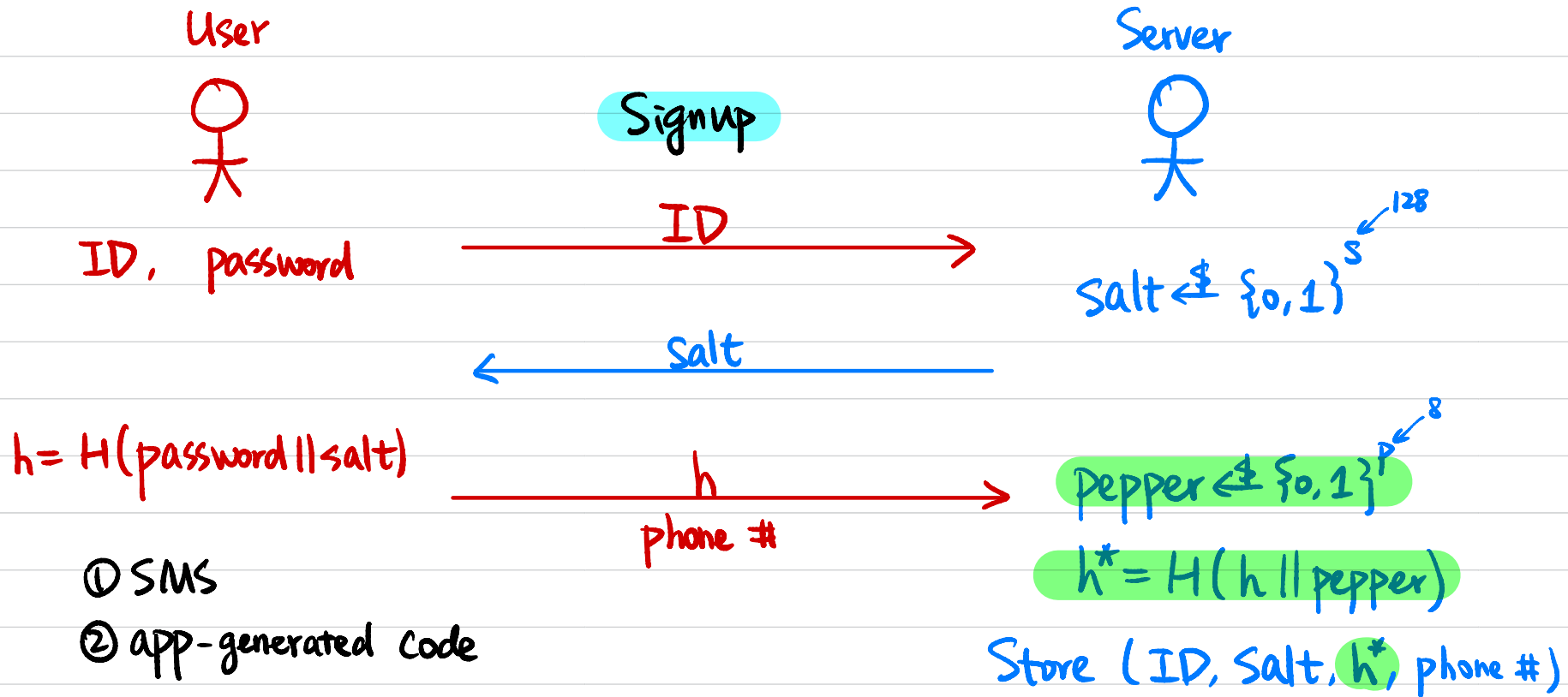
Slow Hash Functions

- Computation-heavy hash function
 - ↳ compose SHA256 in a certain way.

Application-Specific Integrated Circuit (ASIC) → blockchain mining

- Memory-hard hash functions
 - ↳ Scrypt

Two-Factor Authentication (2FA)



How would you design it?