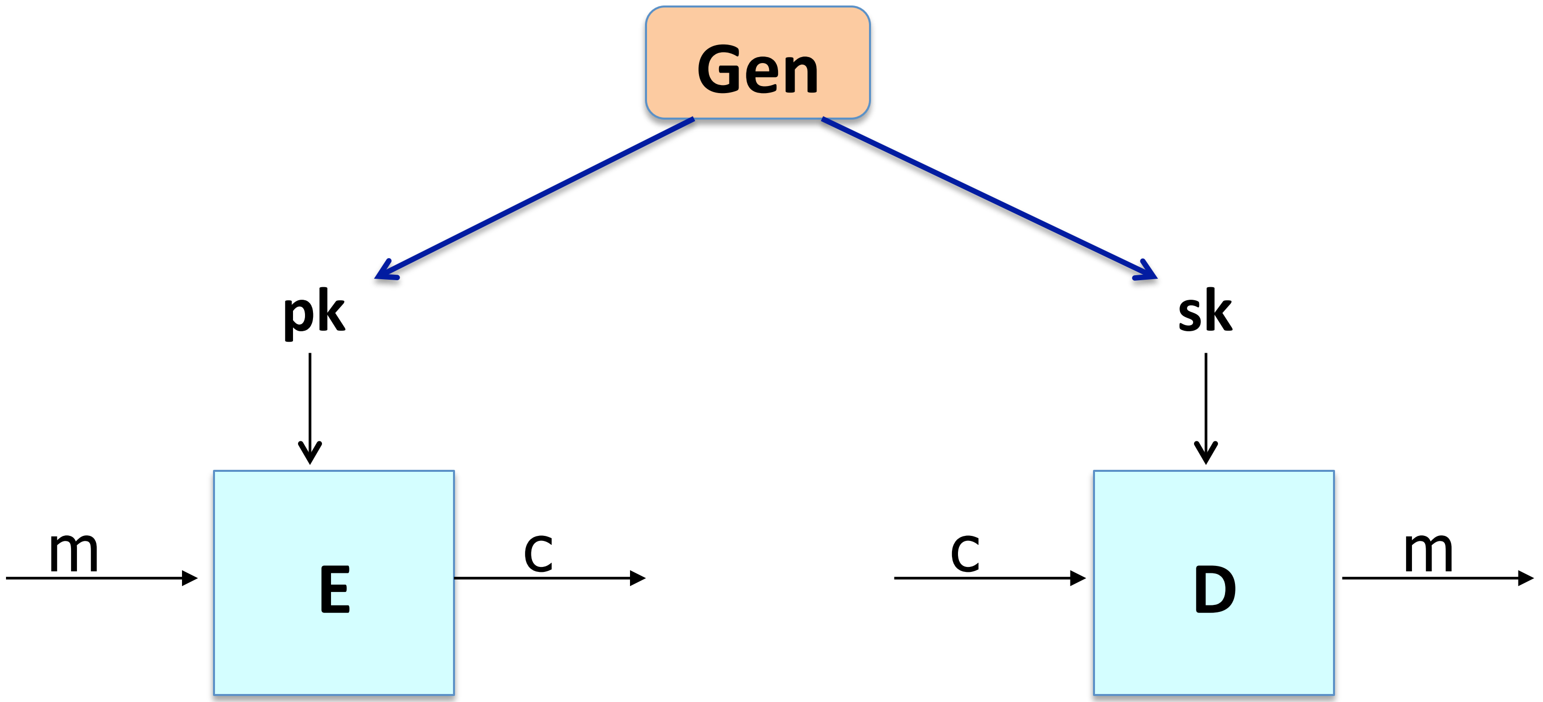
Public key encryption from Diffie-Hellman

The ElGamal Public-key System

Recap: public key encryption: (Gen, E, D)

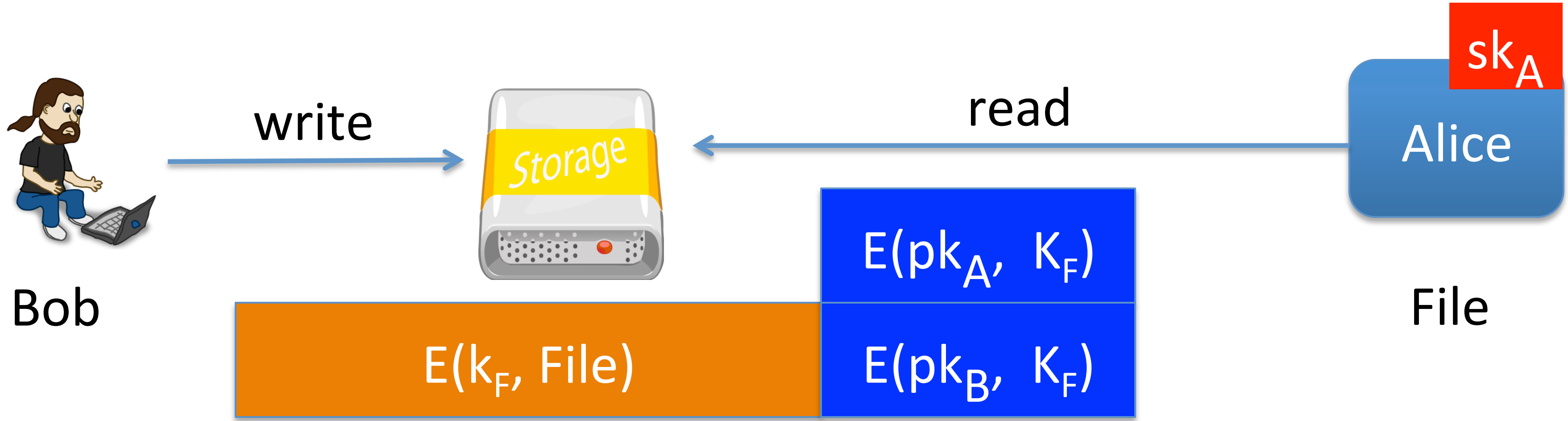


Recap: public-key encryption applications

Key exchange (e.g. in HTTPS)

Encryption in non-interactive settings:

- Secure Email: Bob has Alice's pub-key and sends her an email
- Encrypted File Systems

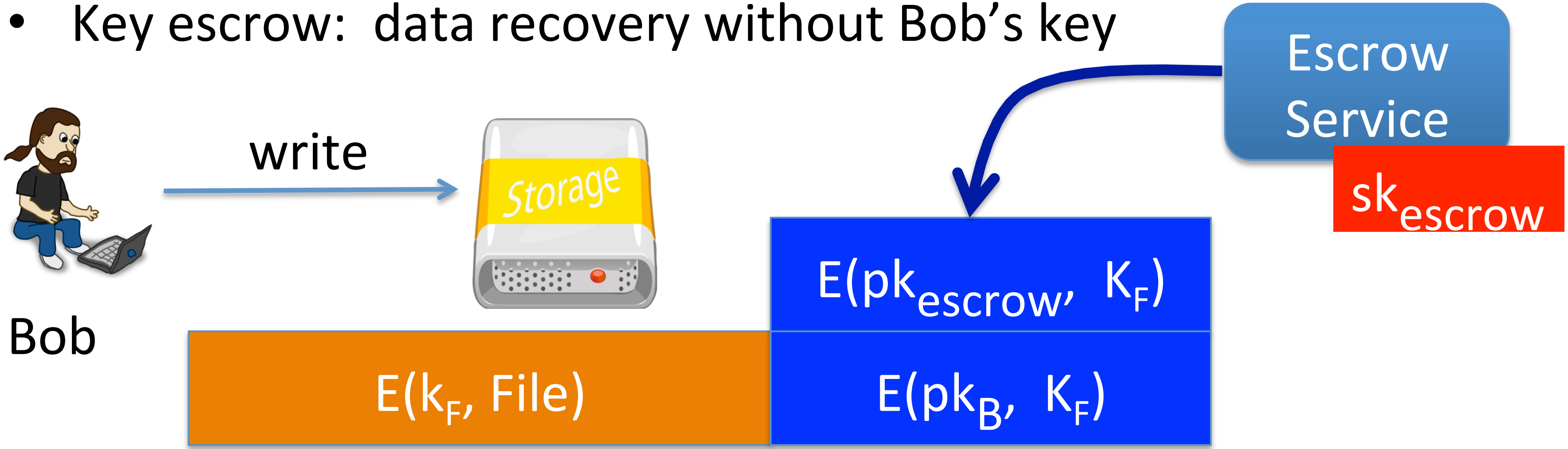


Recap: public-key encryption applications

Key exchange (e.g. in HTTPS)

Encryption in non-interactive settings:

- Secure Email: Bob has Alice's pub-key and sends her an email
- Encrypted File Systems
- Key escrow: data recovery without Bob's key



Constructions

This week: two families of public-key encryption schemes

- Previous lecture: based on trapdoor functions (such as RSA)
  - Schemes: ISO standard, OAEP+, …
- This lecture: based on the Diffie-Hellman protocol
  - Schemes: ElGamal encryption and variants (e.g. used in GPG)

Security goals: chosen ciphertext security

Review: the Diffie-Hellman protocol (1977)

Fix a finite cyclic group G (e.g $G = (Z_p)^*$) of order n

Fix a generator g in G (i.e. $G = \{1, g, g^2, g^3, \ldots, g^{n-1}\}$)

Alice | **Bob**

choose random **a** in {1,…,n} | choose random **b** in {1,…,n}

$A = g^a$ →

← $B = g^b$

$\mathbf{B^a} = (g^b)^a = \mathbf{k_{AB} = g^{ab}} = (g^a)^b = \mathbf{A^b}$

ElGamal: converting to pub-key enc. (1984)

Fix a finite cyclic group G (e.g $G = (Z_p)^*$) of order n

Fix a generator g in G (i.e. $G = \{1, g, g^2, g^3, \dots, g^{n-1}\}$)

Alice

choose random **a** in {1,…,n}

$A = g^a$

Treat as a public key

Bob

ndom **b** in {1,…,n}

compute $g^{ab} = A^b$,

derive symmetric key k ,

ct = [$B = g^b$, encrypt message m with k]

ElGamal: converting to pub-key enc. (1984)

Fix a finite cyclic group G (e.g $G = (Z_p)^*$) of order n

Fix a generator g in G (i.e. $G = \{1, g, g^2, g^3, \dots, g^{n-1}\}$)

Alice

choose random **a** in {1,…,n}

$A = g^a$

Treat as a public key

Bob

ndom **b** in {1,…,n}

compute $g^{ab} = A^b$,
derive symmetric key k ,
encrypt message m with k

ct = [$B = g^b$, encrypt message m with k]

To decrypt:
compute $g^{ab} = B^a$,
derive k, and decrypt

The ElGamal system (a modern view)

- G: finite cyclic group of order n
- (E_s, D_s) : symmetric auth. encryption defined over (K,M,C)
- $H: G^2 \longrightarrow K$ a hash function

We construct a pub-key enc. system (Gen, E, D):

- Key generation Gen:
  - choose random generator g in G and random a in Z_n
  - output sk = a , pk = (g, $h=g^a$)

The ElGamal system (a modern view)

- G: finite cyclic group of order n
- (E_s, D_s) : symmetric auth. encryption defined over (K,M,C)
- $H: G^2 \longrightarrow K$ a hash function

E(pk=(g,h), m) :

$b \xleftarrow{R} Z_n$, $u \longleftarrow g^b$, $v \longleftarrow h^b$

$k \longleftarrow H(u,v)$, $c \longleftarrow E_s(k, m)$

output (u, c)

D(sk=a, (u,c)) :

$v \longleftarrow u^a$

$k \longleftarrow H(u,v)$, $m \longleftarrow D_s(k, c)$

output m

ElGamal performance

E(pk=(g,h), m) :

$b \leftarrow Z_n$, $u \leftarrow g^b$, $v \leftarrow h^b$

D(sk=a, (u,c)) :

$v \leftarrow u^a$

Encryption: 2 exp. (fixed basis)

- Can pre-compute $[\, g^{(2^i)}\,,\ h^{(2^i)}$ for $i=1,\ldots,\log_2 n \,]$
- 3x speed-up (or more)

Decryption: 1 exp. (variable basis)

Next step: why is this system chosen ciphertext secure?
under what assumptions?

End of Segment

Public key encryption from Diffie-Hellman

ElGamal Security

Computational Diffie-Hellman Assumption

G: finite cyclic group of order n

Comp. DH (CDH) assumption holds in G if: $g,\ g^a,\ g^b \not\Rightarrow g^{ab}$

for all efficient algs. A:

$$\Pr\left[\ A(g, g^a, g^b) = g^{ab}\ \right] < \text{negligible}$$

where $g \longleftarrow \{\text{generators of } G\}$, $a, b \longleftarrow Z_n$

Hash Diffie-Hellman Assumption

G: finite cyclic group of order n , $H: G^2 \longrightarrow K$ a hash function

Def: Hash-DH (HDH) assumption holds for (G, H) if:

$$\left(g,\ g^a,\ g^b,\ H(g^b, g^{ab})\right) \approx_p \left(g,\ g^a,\ g^b,\ R\right)$$

where $g \longleftarrow$ {generators of G} , $a, b \longleftarrow Z_n$, $R \longleftarrow K$

H acts as an extractor: strange distribution on G^2 ⇒ uniform on K

Suppose $K = \{0,1\}^{128}$ and

$H: G^2 \longrightarrow K$ only outputs strings in K that begin with 0

(i.e. for all x,y: msb(H(x,y))=0)

Can Hash-DH hold for (G, H) ?

- ○ Yes, for some groups G
- ○ No, Hash-DH is easy to break in this case
- ○ Yes, Hash-DH is always true for such H

ElGamal is sem. secure under Hash-DH

KeyGen: $g \longleftarrow$ {generators of G} , $a \longleftarrow Z_n$

output $pk = (g, h=g^a)$, $sk = a$

E(pk=(g,h), m) : $b \longleftarrow Z_n$

$k \longleftarrow H(g^b, h^b)$, $c \longleftarrow E_s(k, m)$

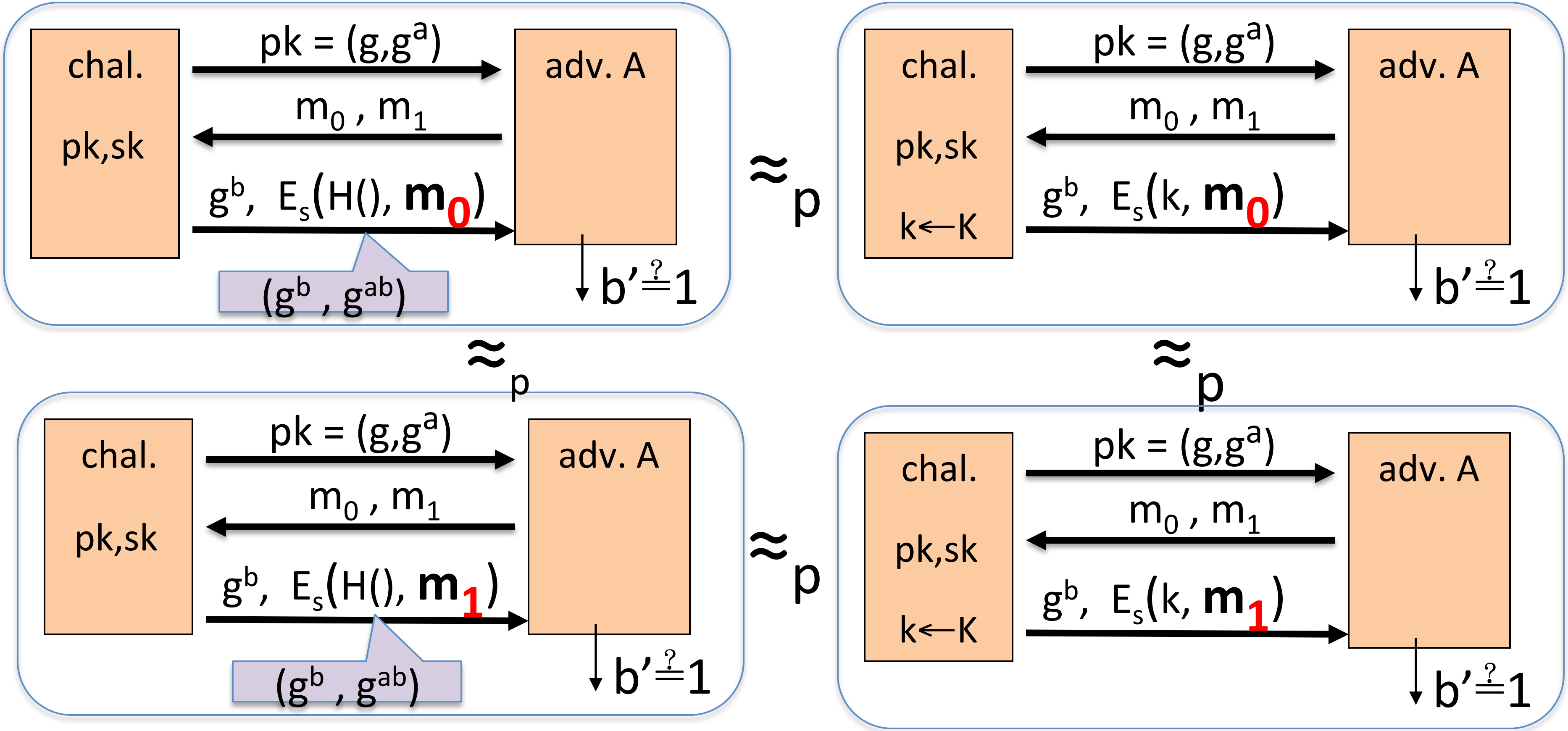
output (g^b, c)

D(sk=a, (u,c)) :

$k \longleftarrow H(u, u^a)$, $m \longleftarrow D_s(k, c)$

output m

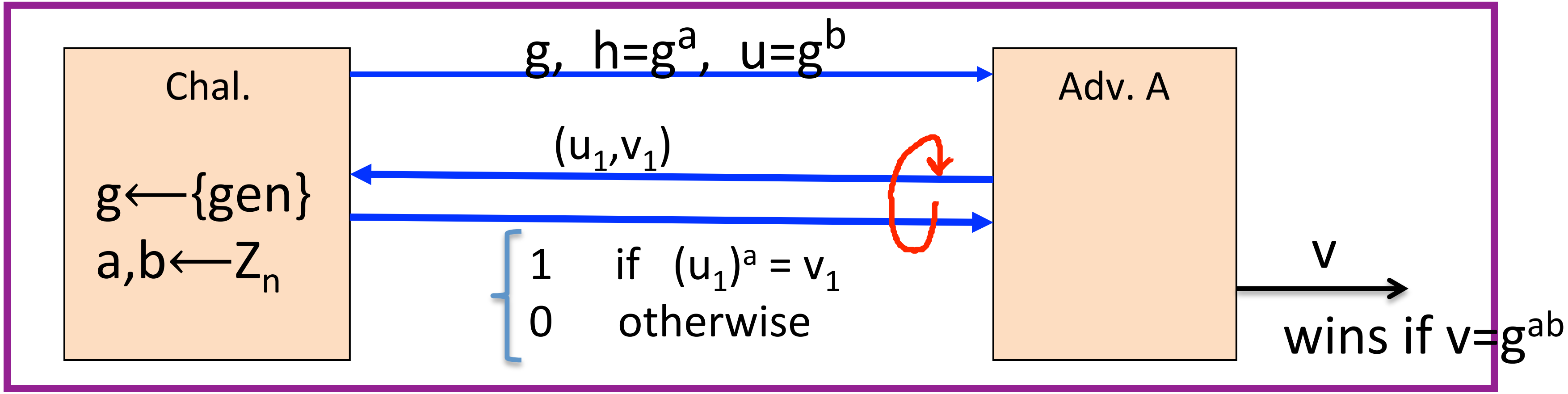
ElGamal is sem. secure under Hash-DH



ElGamal chosen ciphertext security?

To prove chosen ciphertext security need stronger assumption

Interactive Diffie-Hellman (IDH) in group G:



IDH holds in G if: **∀ efficient A: Pr[A outputs g^{ab}] < negligible**

ElGamal chosen ciphertext security?

<u>Security Theorem</u>:

If **IDH** holds in the group G, **(E_s, D_s)** provides auth. enc.

and **H:** $G^2 \longrightarrow K$ is a "random oracle"

then **ElGamal** is CCA^{ro} secure.

Questions: (1) can we prove CCA security based on CDH?

(2) can we prove CCA security without random oracles?

End of Segment

Public key encryption from Diffie-Hellman

ElGamal Variants With Better Security

Review: ElGamal encryption

KeyGen: $g \longleftarrow$ {generators of G} , $a \longleftarrow Z_n$

output $pk = (g, h=g^a)$, $sk = a$

E(pk=(g,h), m) : $b \longleftarrow Z_n$

$k \longleftarrow H(g^b,h^b)$, $c \longleftarrow E_s(k, m)$

output (g^b, c)

D(sk=a, (u,c)) :

$k \longleftarrow H(u,u^a)$, $m \longleftarrow D_s(k, c)$

output m

ElGamal chosen ciphertext security

Security Theorem:

If **IDH** holds in the group G, **(E_s, D_s)** provides auth. enc.

and **H:** $G^2 \longrightarrow K$ is a "random oracle"

then **ElGamal** is CCA^{ro} secure.

Can we prove CCA security based on CDH $(g,\ g^a,\ g^b \not\rightarrow\ g^{ab})$?

- Option 1: use group G where CDH = IDH (a.k.a bilinear group)
- Option 2: change the ElGamal system

Variants: twin ElGamal [CKS'08]

KeyGen: $g \longleftarrow$ {generators of G} , $a1, a2 \longleftarrow Z_n$

output $pk = (g, h_1 = g^{a1}, h_2 = g^{a2})$, $sk = (a1, a2)$

E(pk=(g,h₁,h₂), m) : $b \longleftarrow Z_n$

$k \longleftarrow H(g^b, h_1{}^b, h_2{}^b)$

$c \longleftarrow E_s(k, m)$

output (g^b, c)

D(sk=(a1,a2), (u,c)) :

$k \longleftarrow H(u, u^{a1}, u^{a2})$

$m \longleftarrow D_s(k, c)$

output m

Chosen ciphertext security

__Security Theorem__:

If **CDH** holds in the group G, **(E_s, D_s)** provides auth. enc.

and **H:** $G^3 \longrightarrow K$ is a "random oracle"

then **twin ElGamal** is CCA^{ro} secure.

Cost: one more exponentiation during enc/dec

– Is it worth it? No one knows …

ElGamal security w/o random oracles?

Can we prove CCA security without random oracles?

- Option 1: use Hash-DH assumption in “bilinear groups”
  - Special elliptic curve with more structure [CHK’04 + BB’04]

- Option 2: use Decision-DH assumption in any group [CS’98]

Further Reading

- The Decision Diffie-Hellman problem. D. Boneh, ANTS 3, 1998
- Universal hash proofs and a paradigm for chosen ciphertext secure public key encryption. R. Cramer and V. Shoup, Eurocrypt 2002
- Chosen-ciphertext security from Identity-Based Encryption.
  D. Boneh, R. Canetti, S. Halevi, and J. Katz, SICOMP 2007
- The Twin Diffie-Hellman problem and applications.
  D. Cash, E. Kiltz, V. Shoup, Eurocrypt 2008
- Efficient chosen-ciphertext security via extractable hash proofs.
  H. Wee, Crypto 2010

Public key encryption from Diffie-Hellman

A Unifying Theme

One-way functions (informal)

A function $f: X \longrightarrow Y$ is one-way if

- There is an efficient algorithm to evaluate $f(\cdot)$, but
- Inverting f is hard:

  for all efficient A and $x \longleftarrow X$:

$$\Pr[\, f(A(f(x))) = f(x) \,] < \text{negligible}$$

Functions that are not one-way: $f(x) = x$, $f(x) = 0$

Ex. 1: generic one-way functions

Let $f: X \longrightarrow Y$ be a secure PRG (where $|Y| \gg |X|$)

(e.g. f built using det. counter mode)

Lemma: f a secure PRG $\Rightarrow$ f is one-way

Proof sketch:

$$A \text{ inverts } f \quad \Rightarrow \quad B(y) = \begin{cases} 0 & \text{if } f(A(y)) = y \\ 1 & \text{otherwise} \end{cases} \quad \text{is a distinguisher}$$

Generic: no special properties. Difficult to use for key exchange.

Ex 2: The DLOG one-way function

Fix a finite cyclic group G (e.g $G = (Z_p)^*$) of order n

g: a random generator in G (i.e. $G = \{1, g, g^2, g^3, \dots, g^{n-1}\}$)

Define: $f: Z_n \longrightarrow G$ as $f(x) = g^x \in G$

Lemma: Dlog hard in G $\Rightarrow$ f is one-way

Properties: $f(x), f(y) \Rightarrow f(x+y) = f(x) \cdot f(y)$

$\Rightarrow$ key-exchange and public-key encryption

Ex. 3: The RSA one-way function

- choose random primes p,q ≈1024 bits. Set **N=pq**.
- choose integers **e , d** s.t. **$e \cdot d = 1 \pmod{\varphi(N)}$**

Define: f: $\mathbb{Z}_N^* \to \mathbb{Z}_N^*$ as f(x) = x^e in $\mathbb{Z}_N$

Lemma: f is one-way under the RSA assumption

Properties: $f(x \cdot y) = f(x) \cdot f(y)$ and **f has a trapdoor**

Summary

Public key encryption:

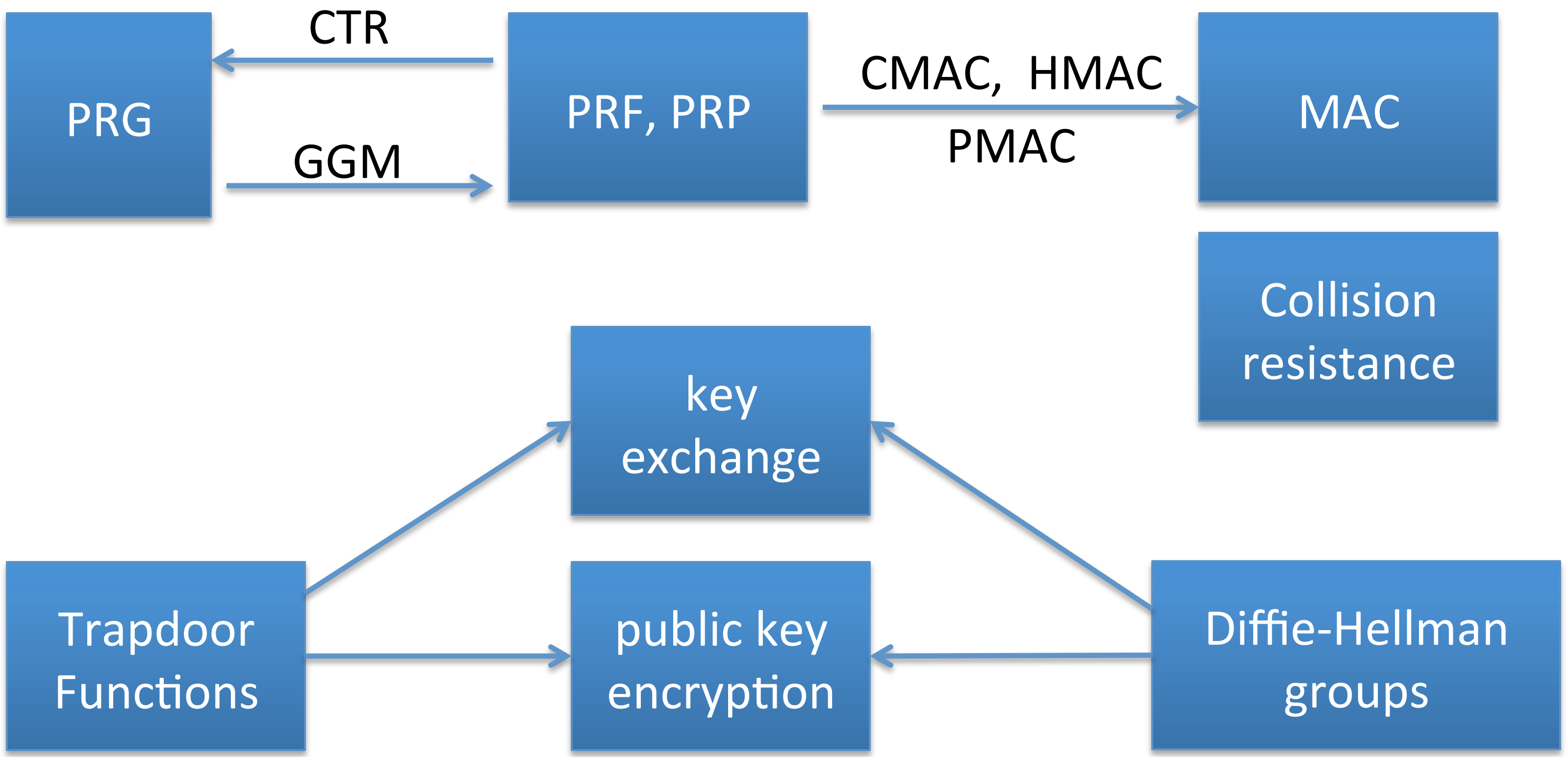
made possible by one-way functions
with special properties

homomorphic properties and trapdoors

End of Segment

Farewell (for now)

Quick Review: primitives



Quick Review: primitives

To protect non-secret data: (data integrity)

- using small read-only storage: use collision resistant hash
- no read-only space: use MAC ... requires secret key

To protect sensitive data: only use authenticated encryption

(eavesdropping security by itself is insufficient)

Session setup:

- Interactive settings: use authenticated key-exchange protocol
- When no-interaction allowed: use public-key encryption

Remaining Core Topics (part II)

- Digital signatures and certificates
- Authenticated key exchange
- User authentication:

  passwords, one-time passwords, challenge-response

- Privacy mechanisms
- Zero-knowledge protocols

Many more topics to cover …

- Elliptic Curve Crypto
- Quantum computing
- New key management paradigms:
  identity based encryption and functional encryption
- Anonymous digital cash
- Private voting and auction systems
- Computing on ciphertexts: fully homomorphic encryption
- Lattice-based crypto
- Two party and multi-party computation

Final Words

Be careful when using crypto:

- A tremendous tool, but if incorrectly implemented:

  system will work, but may be easily attacked

Make sure to have others review your designs and code

Don't invent your own ciphers or modes

End of part I