COL759:

CRYPTOGRAPHY AND COMPUTER SECURITY

2022-23 (SEMESTER 1)

LECTURE 29: PUBLIC KEY ENC

REVIEW: $\mathbb{Z}_p^*$ and its prime-order subgroup $\mathbb{G}$

$\mathbb{Z}_p^* = \langle g \rangle$ where g : generator of $\mathbb{Z}_p^*$

$\mathbb{Z}_p^*$

$|\mathbb{Z}_p^*| = p - 1 = 2q$

Almost 50% of elements in group are generators of $\mathbb{Z}_p^*$

For any element $a \in \mathbb{Z}_p^*$, $|\langle a \rangle| \in \{1, 2, q, 2q\}$. There are at most 2 elements s.t. $a^2 = 1 \mod p$. Similarly, at most q elements s.t. $a^q = 1 \mod p$. Rest must be generators of $\mathbb{Z}_p^*$

Common Properties

- contain 1
- Closed under multiplication mod p
- Every element has an inverse

DISCRETE LOG problem : hard

$\mathbb{G} = \langle g^2 \rangle$: prime-order subgroup of $\mathbb{Z}_p^*$

$|\mathbb{G}| = q$

Almost every element in group is a generator of $\mathbb{G}$

Every element $a \in \mathbb{G}, a \neq 1$ is a gen. of $\mathbb{G}$. (Why?)

Decision Diffie-Hellman problem : hard

REVIEW: DLOG and DDH over prime-order group $\mathbb{G}$

DISCRETE LOG problem

Given random generators (g, h)
compute a such that $h = g^a$

DECISION DIFFIE-HELLMAN problem

Distinguish the following distributions

$$\mathscr{D}_0 = \left\{(g, g^a, g^b, g^{a \cdot b})\right\}_{g \leftarrow \mathbb{G},\ a,b \leftarrow \mathbb{Z}_q} \qquad \mathscr{D}_1 = \left\{(g, g^a, g^b, g^c)\right\}_{g \leftarrow \mathbb{G},\ a,b,c \leftarrow \mathbb{Z}_q}$$

Given the first three elements from a $\mathscr{D}_0$ sample,
the fourth one is uniquely determined.

$\mathscr{D}_1$ is statistically close to a uniform distribution over $\mathbb{G} \times \mathbb{G} \times \mathbb{G} \times \mathbb{G}$.
(Why is it not identical to uniform dist. over $\mathbb{G} \times \mathbb{G} \times \mathbb{G} \times \mathbb{G}$?)

PLAN FOR TODAY'S LECTURE

- A collision resistant hash fn using DLOG
- Intro to public key encryption
- A secure public key enc. scheme using DDH

COLLISION RESISTANCE USING PRIME-ORDER GROUP $\mathbb{G}$

Prime order group $\mathbb{G}$ of size q

Hash key: two elements in $\mathbb{G}$

$H_k : \mathbb{Z}_q \times \mathbb{Z}_q \to \mathbb{G}$

$$H_{(x,y)}(a,b) = (x^a) \times_p (y^b)$$

mult mod p

Looks very similar to the insecure construction we saw last time

Goal: H is secure, assuming hardness of DISCRETE LOG

CRHF CONSTRUCTION USING PRIME-ORDER GROUP $\mathbb{G}$

$$H_k : \mathbb{Z}_q \times \mathbb{Z}_q \to \mathbb{G} \qquad H_{(x,y)}(a,b) = (x^a) \times_p (y^b)$$

Goal: secure CRHF assuming DLOG

Proof sketch:
Suppose there exists a collision for this hash function.
Then $H_{(x,y)}(a,b) = H_{(x,y)}(a',b')$ and $(a,b) \neq (a',b')$.

First, note that since $(a,b) \neq (a',b')$, and this is a valid collision, $a \neq a'$ and $b \neq b'$ (why?)

$$\left[x^a \cdot y^b \mod p = x^{a'} \cdot y^{b'} \mod p\right]$$

$$\implies \left[x^{(a-a') \mod q} \mod p = y^{(b'-b) \mod q} \mod p\right] \text{ (why?)}$$

Let β be the unique integer in $\mathbb{Z}_q$ such that $\beta \cdot (b' - b') = 1 \mod q$.
Raise both sides of above equation to β^{th} power. This gives
$y = x^{(a-a')\cdot\beta \mod q} \mod p$, and hence we have computed the discrete log of y (w.r.t. x).

Use this to give a formal reduction.
(Where did we use the fact that q is prime? And why is this important ?)

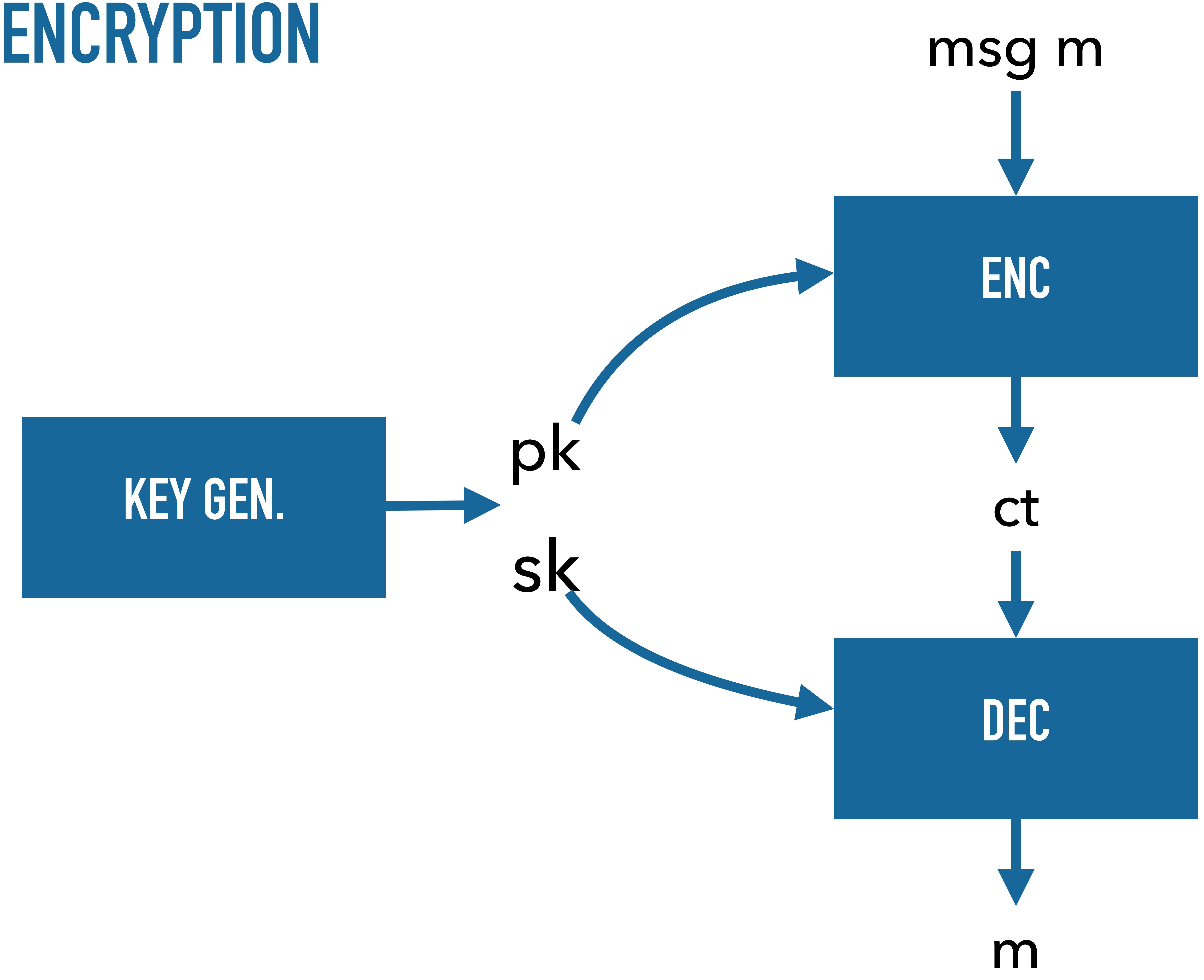
H.W.

Modify previous construction to make it secure, assuming DLOG is hard over $\mathbb{Z}_p^*$

MACs
Auth. Enc.
CRHFs
PRFs
Key Agreement
Commitments
PRGs
DDH over prime order groups
PUBLIC KEY ENCRYPTION

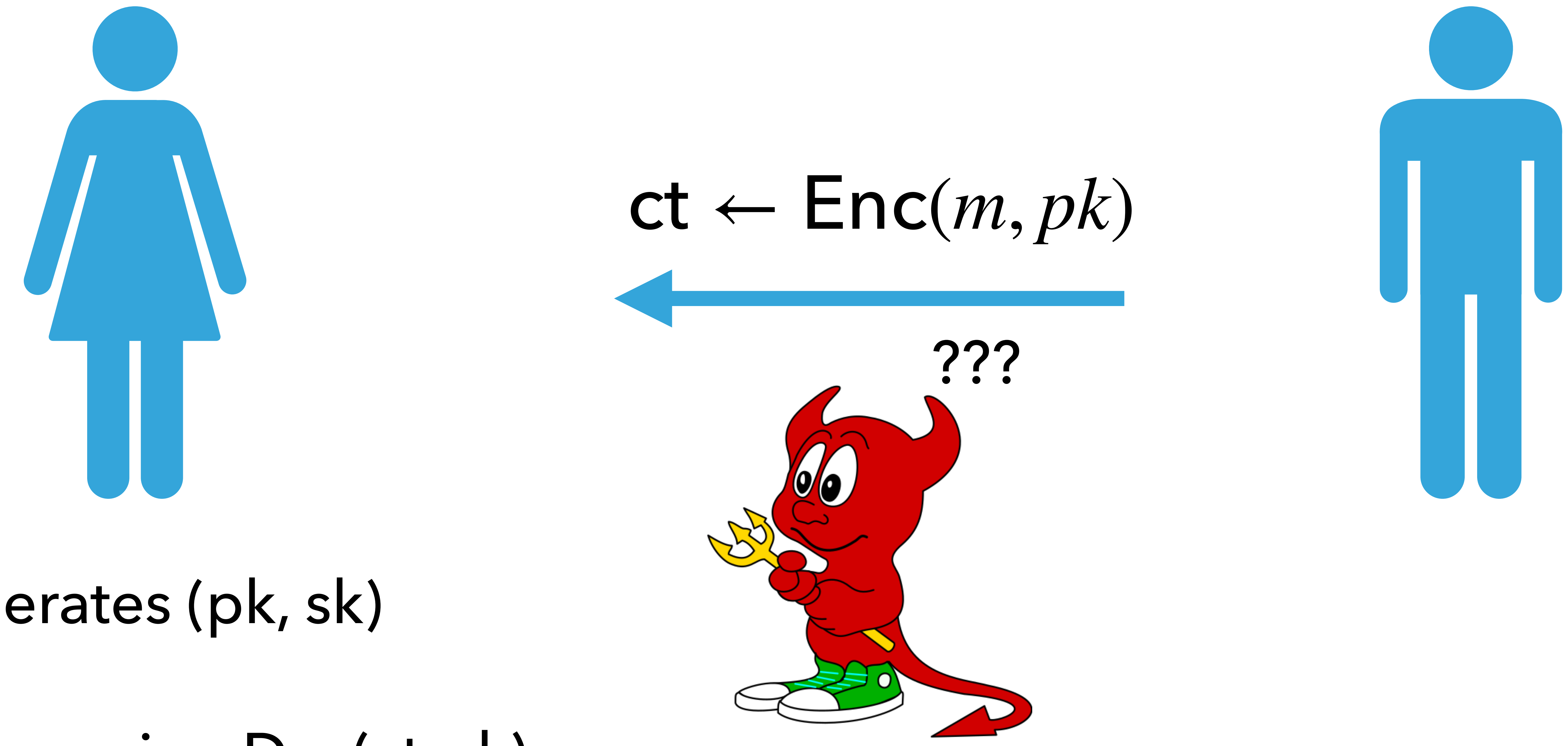
The PRG construction follows from Slide 3, where we defined distributions $\mathscr{D}_0$ and $\mathscr{D}_1$. Define fn. $G : \mathbb{G} \times \mathbb{Z}_q \times \mathbb{Z}_q \rightarrow \mathbb{G} \times \mathbb{G} \times \mathbb{G} \times \mathbb{G}$ as follows: $G(g, a, b) = (g, g^a, g^b, g^{ab})$

PUBLIC KEY ENCRYPTION



PUBLIC KEY ENCRYPTION

pk: known to everyone



ct ← Enc(m, pk)

???

Generates (pk, sk)

Recovers m using Dec(ct, sk)

DEFINING SECURITY FOR PUBLIC KEY ENCRYPTION

Security in the symmetric key setting

We defined two separate security definitions for passive security in the symmetric key setting. The first definition (no query semantic security) allowed the adversary to send one challenge pair, the challenger sends encryption of one of them, and the adversary must guess.

Next, we discussed general semantic security, where the challenger first chooses a key k and a bit b. The adversary then sends polynomially many pairs of messages, and receives encryption of one of the messages from each pair, depending on the bit b.

In the symmetric key setting, these first notion is strictly weaker than the second one. We saw encryption schemes (such as Shannon's OTP, or the PRG based construction) which satisfy the first definition, but do not satisfy the general semantic security definition.

PASSIVE SECURITY: ONE-TIME SEMANTIC SECURITY

Chall.

Adv.

Chooses keys (pk, sk)

pk

Setup

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

$\mathrm{ct}^* \leftarrow \mathrm{Enc}(\mathrm{m}_b^*, pk)$

m_0^* m_1^*

ct^*

Chall.

b′

Guess

PASSIVE SECURITY: MANY-TIME SEMANTIC SECURITY

Chall.

Adv.

Chooses keys (pk, sk)

Chooses bit b

pk

Setup

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

$\mathrm{ct}^* \leftarrow \mathrm{Enc}(\mathrm{m}^*_{i,b}\ , pk)$

$m^*_{i,0}$ $m^*_{i,1}$

ct^*_i

Poly. many
Chall. queries

b'

Guess

Is many-time semantic security stronger than one-time semantic security?

A natural first guess, based on our symmetric key experience, is that one-time security for PKE is strictly weaker than many-time security for PKE. However, these two notions are equivalent in the PKE setting.

MANY-TIME SEMANTIC SECURITY ≡ ONE-TIME SEMANTIC SECURITY

Key idea: Hybrid technique

We will illustrate this using two queries. The general case follows similarly.

Why does this argument not work in the symm. key setting?

WORLD 0

Challenger | Adversary

$(pk, sk) \leftarrow$ KeyGen

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

$\xrightarrow{\quad pk \quad}$

$\xleftarrow{\quad (m_{1,0}\ ,\ m_{1,1}) \quad}$

$\mathsf{ct}_1 \leftarrow \mathsf{Enc}(m_{1,0}, pk)$

$\xrightarrow{\quad \mathsf{ct}_1 \quad}$

$\xleftarrow{\quad (m_{2,0}\ ,\ m_{2,1}) \quad}$

$\mathsf{ct}_2 \leftarrow \mathsf{Enc}(m_{2,0}, pk)$

$\xrightarrow{\quad \mathsf{ct}_2 \quad}$

Indist. due to one-time SS. If an adv. has different win probability in world-0 and hyb-world, then there exists reduction that wins one-time SS.

HYB-WORLD

Challenger | Adversary

$(pk, sk) \leftarrow$ KeyGen

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

$\xrightarrow{\quad pk \quad}$

$\xleftarrow{\quad (m_{1,0}\ ,\ m_{1,1}) \quad}$

$\mathsf{ct}_1 \leftarrow \mathsf{Enc}(m_{1,1}, pk)$

$\xrightarrow{\quad \mathsf{ct}_1 \quad}$

$\xleftarrow{\quad (m_{2,0}\ ,\ m_{2,1}) \quad}$

$\mathsf{ct}_2 \leftarrow \mathsf{Enc}(m_{2,0}, pk)$

$\xrightarrow{\quad \mathsf{ct}_2 \quad}$

Indist. due to one-time SS. If an adv. has different win probability in world-1and hyb-world, then there exists reduction that wins one-time SS.

WORLD 1

Challenger | Adversary

$(pk, sk) \leftarrow$ KeyGen

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

$\xrightarrow{\quad pk \quad}$

$\xleftarrow{\quad (m_{1,0}\ ,\ m_{1,1}) \quad}$

$\mathsf{ct}_1 \leftarrow \mathsf{Enc}(m_{1,1}, pk)$

$\xrightarrow{\quad \mathsf{ct}_1 \quad}$

$\xleftarrow{\quad (m_{2,0}\ ,\ m_{2,1}) \quad}$

$\mathsf{ct}_2 \leftarrow \mathsf{Enc}(m_{2,1}, pk)$

$\xrightarrow{\quad \mathsf{ct}_2 \quad}$

ATTACKS NOT CAPTURED BY SEMANTIC SECURITY GAME

Malleability attacks.

Bob has Alice's pk, encrypts a message using this pk. But adversary can tamper with the ciphertext, resulting in Alice receiving a different message on decryption. We will discuss this soon (hopefully lecture 30/31)

How to ensure Bob receives Alice's public key?

A serious issue in practice, we will discuss this towards the end of the semester.

’74 - Diffie-Hellman propose key agreement protocol

Introduce the notion of public key encryption

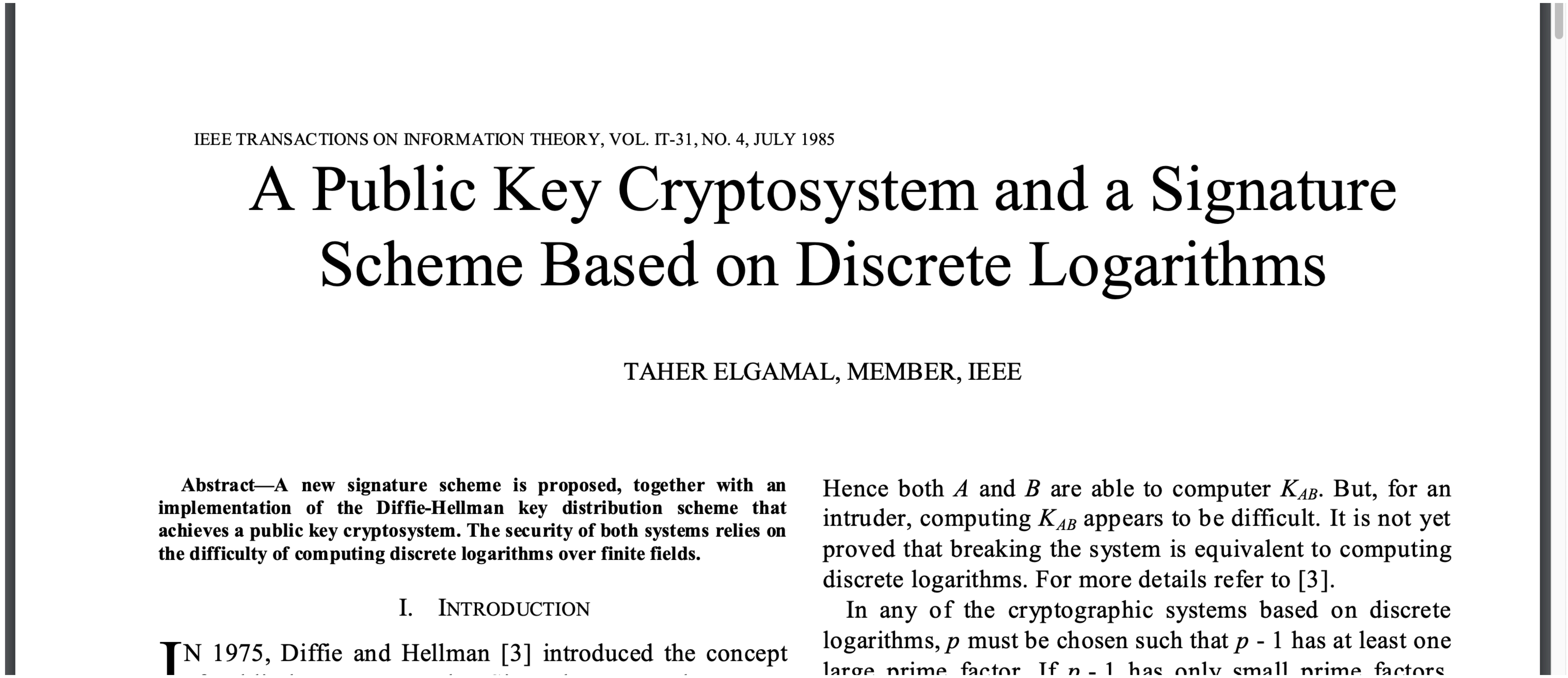
’77 - Rivest-Shamir-Adleman: first PKE scheme

’82 - Goldwasser-Micali: formal definitions of PKE security

’85 - Elgamal: PKE scheme very similar to DH key exchange

DDH-based PKE scheme

Taher Elgamal



IEEE TRANSACTIONS ON INFORMATION THEORY, VOL. IT-31, NO. 4, JULY 1985

A Public Key Cryptosystem and a Signature Scheme Based on Discrete Logarithms

TAHER ELGAMAL, MEMBER, IEEE

Abstract—A new signature scheme is proposed, together with an implementation of the Diffie-Hellman key distribution scheme that achieves a public key cryptosystem. The security of both systems relies on the difficulty of computing discrete logarithms over finite fields.

I. Introduction

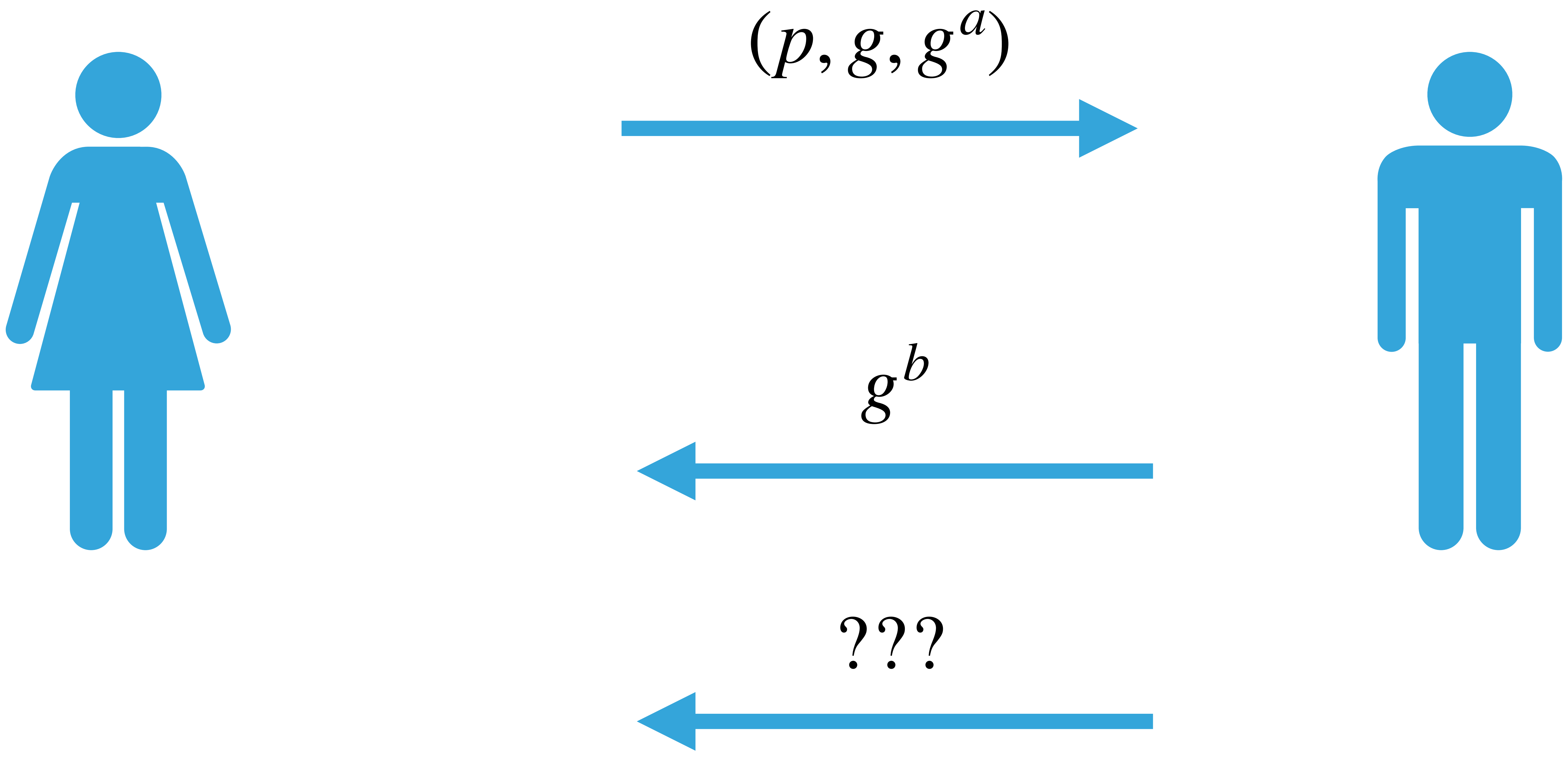
IN 1975, Diffie and Hellman [3] introduced the concept

Hence both A and B are able to computer K_{AB}. But, for an intruder, computing K_{AB} appears to be difficult. It is not yet proved that breaking the system is equivalent to computing discrete logarithms. For more details refer to [3].

In any of the cryptographic systems based on discrete logarithms, p must be chosen such that p - 1 has at least one large prime factor. If p - 1 has only small prime factors,

DDH-based PKE scheme: The Elgamal Encryption Scheme

Toy scenario: Bob wants to send exactly one secret message m to Alice



(p, g, g^a)

g^b

$???$

Bob can send $m \cdot g^{ab}$

Given this and g^a

Alice can recover m

DDH-based PKE scheme: The Elgamal Encryption Scheme

Construction uses prime order group $\mathbb{G} = \langle g \rangle$

Message space: $\mathbb{G}$

Key Generation: $a \leftarrow \mathbb{Z}_q, \quad pk = g^a, sk = a$

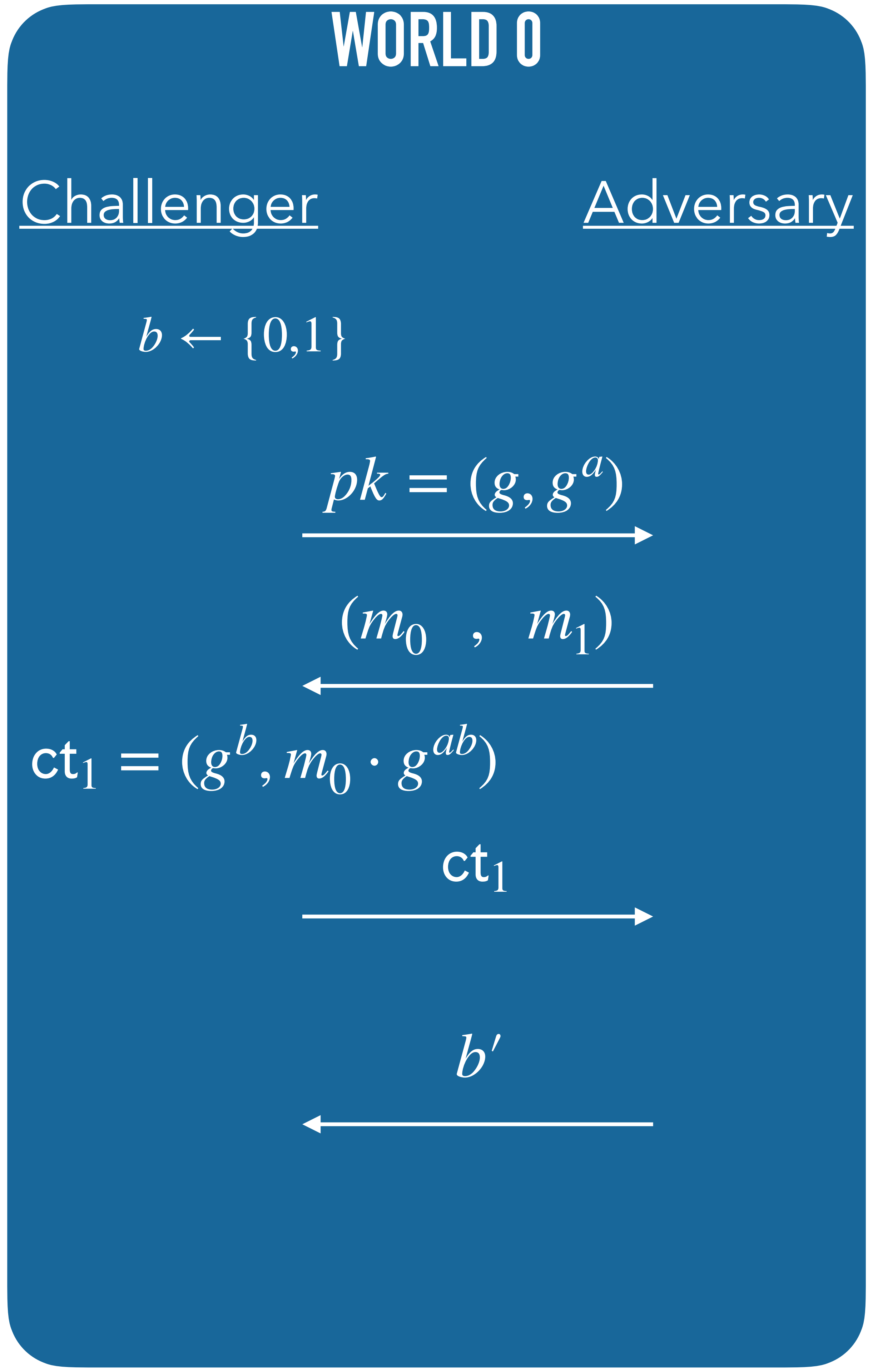
Enc (m, pk) : $\quad b \leftarrow \mathbb{Z}_q, \quad ct_1 = g^b, ct_2 = m \cdot (pk)^b$

Dec $((ct_1, ct_2), sk)$: $\quad$ Output $ct_2 \cdot (ct_1^{q-sk})$

H.W.

How to encrypt longer messages?

DDH-based Proof of Security



WORLD 0

Challenger Adversary

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

$\xrightarrow{pk = (g, g^a)}$

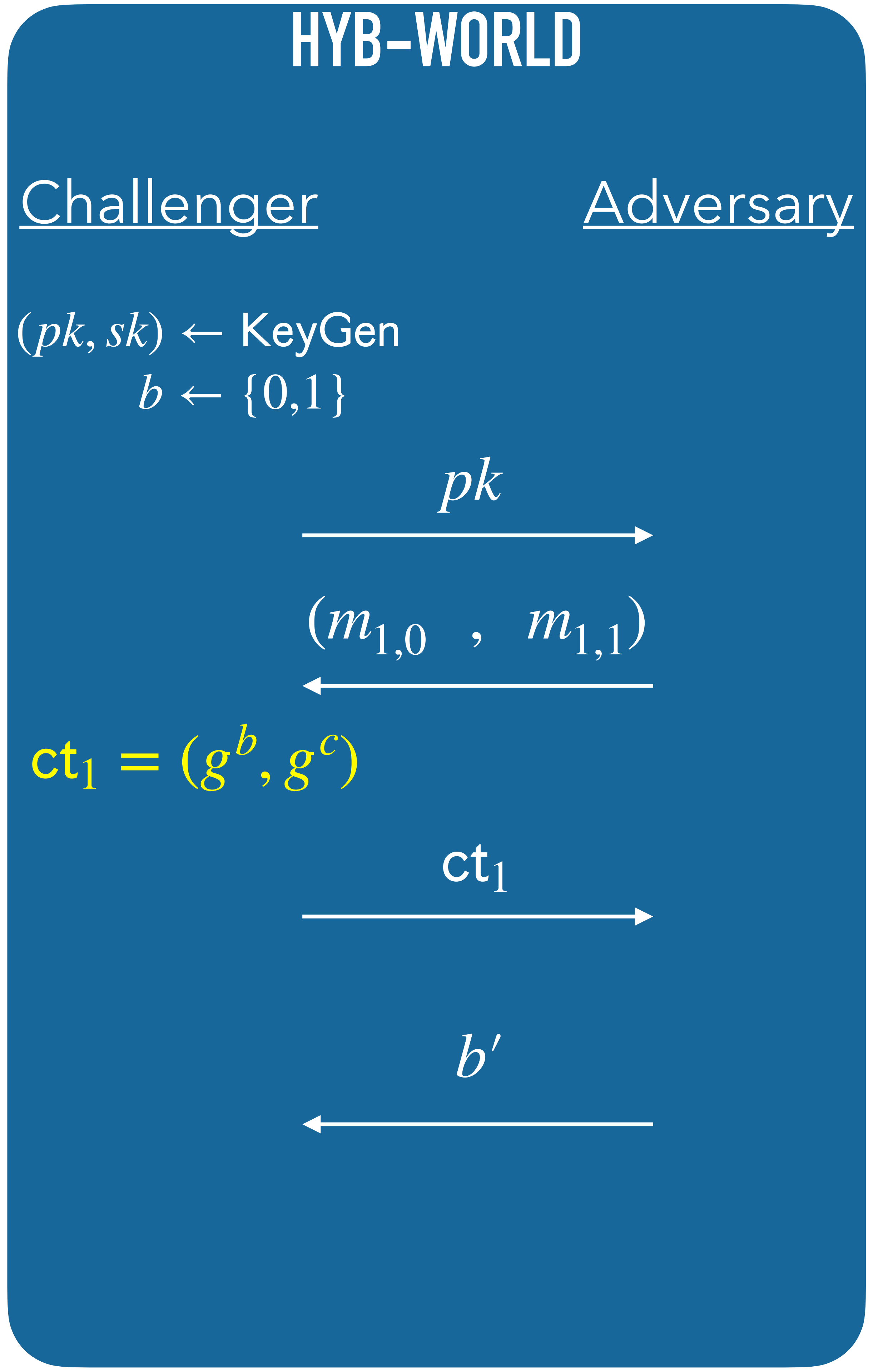
$\xleftarrow{(m_0 \;,\; m_1)}$

$\mathsf{ct}_1 = (g^b, m_0 \cdot g^{ab})$

$\xrightarrow{\mathsf{ct}_1}$

$\xleftarrow{b'}$

Indist. due to DDH assumption. If an adv. has different win probability in world-0 and hyb-world, then there exists reduction that breaks DDH.



HYB-WORLD

Challenger Adversary

$(pk, sk) \leftarrow \mathsf{KeyGen}$

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

$\xrightarrow{pk}$

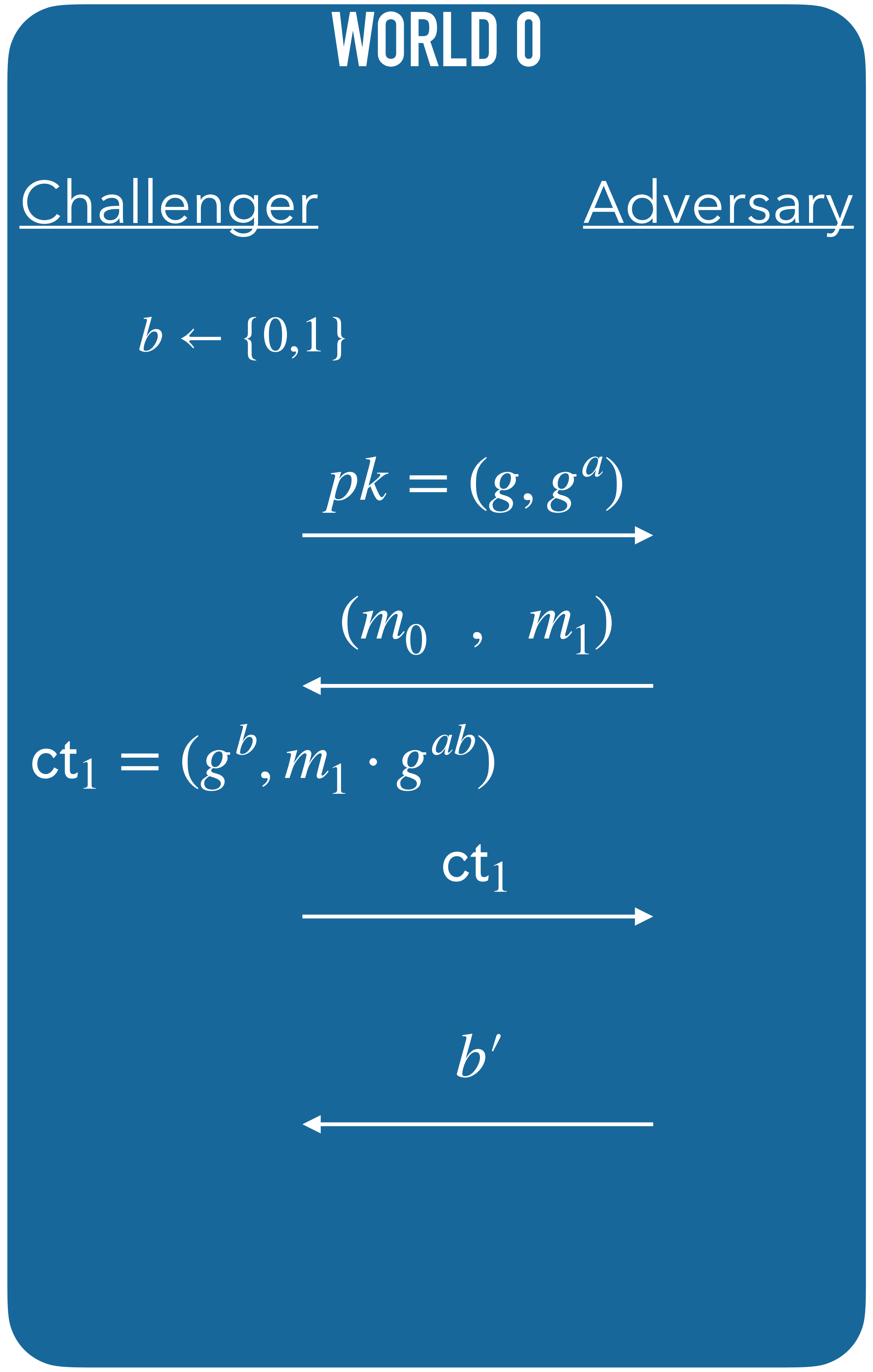
$\xleftarrow{(m_{1,0} \;,\; m_{1,1})}$

$\mathsf{ct}_1 = (g^b, g^c)$

$\xrightarrow{\mathsf{ct}_1}$

$\xleftarrow{b'}$

Same as before



WORLD 0

Challenger Adversary

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

$\xrightarrow{pk = (g, g^a)}$

$\xleftarrow{(m_0 \;,\; m_1)}$

$\mathsf{ct}_1 = (g^b, m_1 \cdot g^{ab})$

$\xrightarrow{\mathsf{ct}_1}$

$\xleftarrow{b'}$

End of lecture.

Exercises

Hash the message using Qn. 3 in assgn 4. and then encrypt.

1. Construct a public key encryption scheme with unbounded message space, whose security can be proven assuming the DDH problem is hard.

2. Let $p = 2q + 1$ be a safe prime where $p = \Theta(2^n)$. Assume the Discrete Log problem is hard over prime-order subgroups of $\mathbb{Z}_p^*$ (that is, for any p.p.t. algorithm B, $\Pr[B(h, h^b) = b : h$ is a random element of $\mathbb{Z}_p^*$ s.t. $|\langle h \rangle| = q, b \leftarrow \mathbb{Z}_q] \leq \text{negl}(n))$.

   Show that the Discrete Log problem is also hard over $\mathbb{Z}_p^*$. That is, for any p.p.t. algorithm B, the following probability is also bounded by a negligible function in n: $\Pr[B(g, g^a) = a : g$ is a random generator of $\mathbb{Z}_p^*$, $\; a \leftarrow \mathbb{Z}_p]$

   What about the converse?

Exercises

Easy hain

3. Assume DDH is hard over prime-order group $\mathbb{G}$. Show that the following distributions are indistinguishable:

$$\mathscr{D}_0 = \left\{(g, g^a, g^b, g^c, g^{a\cdot b}, g^{a\cdot c})\right\}_{g\leftarrow\mathbb{G},\ a,b,c\leftarrow\mathbb{Z}_q} \qquad \mathscr{D}_1 = \left\{(g, g^a, g^b, g^c, g^d, g^e)\right\}_{g\leftarrow\mathbb{G},\ a,b,c,d,e\leftarrow\mathbb{Z}_q}$$

Hint: hybrid technique

4 (**). Let $\mathbb{G}$ be a prime-order group, and suppose there exists a p.p.t. algorithm that can solve the DDH problem with probability 0.51. Discuss how to boost the success probability to 0.99.

Hint: you will need Chernoff's bound for this. As a stepping stone, consider the following simpler problem: you are given a sample, which is either a DDH sample, or a non-DDH sample. Using this sample, generate two independent samples, such that the two samples are DDH samples if and only the original sample was a DDH sample.