MATH/CMSC 456 :: UPDATED COURSE INFO

Instructor: Gorjan Alagic (galagic@umd.edu)

Guest instructor: Carl Miller (camiller@umd.edu), ATL 3100K

Textbook: *Introduction to Modern Cryptography*, Katz and Lindell;

Current readings:

Mar 3: 359-372, 375-382, 387-399

Mar 5: pp. 399-432 (skip subsections 11.4.3 and 11.5.5)

Webpage: alagic.org/cmsc-456-cryptography-spring-2020/

Piazza: piazza.com/umd/spring2020/cmsc456

ELMS: active, slides and reading posted there.

Gradescope: active, access through ELMS.

TAs (Our spot: shared open area across from **AVW 4166**)

- Elijah Grubb (egrubb@cs.umd.edu) 11am-12pm TuTh (AVW);
- Justin Hontz (jhontz@terpmail.umd.edu) 1pm-2pm MW (AVW);

Additional help:

- Chen Bai (cbai1@terpmail.umd.edu) 3:30-5:30pm Tu (2115 ATL – inside **JQI**)
- Bibhusa Rawal (bibhusa@terpmail.umd.edu) 3:30-5:30pm Th (2115 ATL – inside **JQI**)

RECAP: EFFICIENT OPERATIONS MOD q

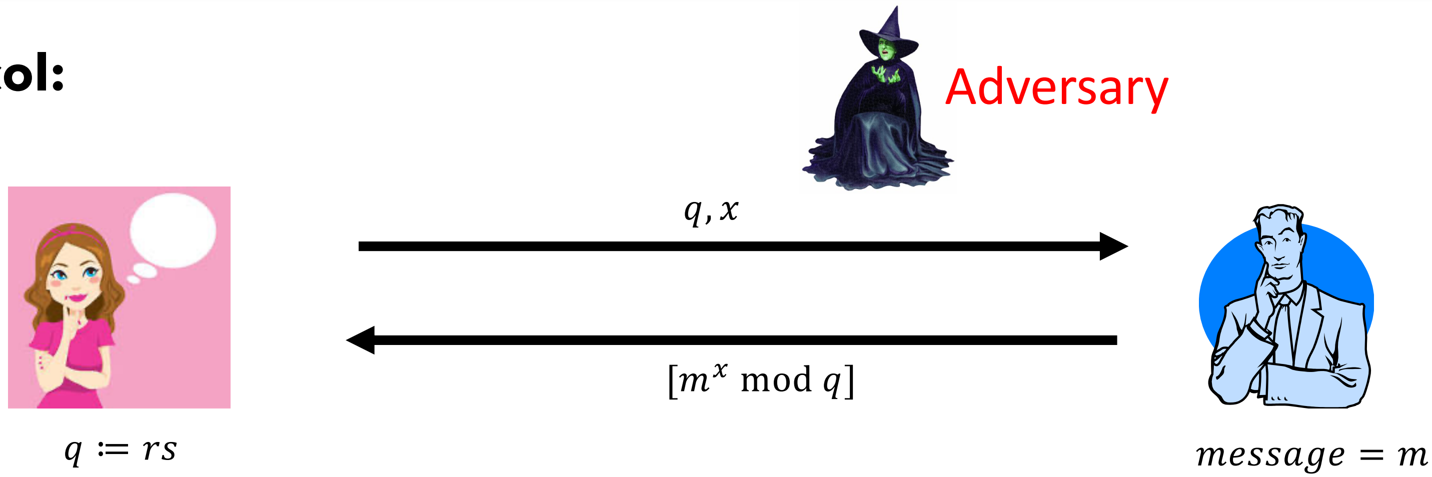
$\boxed{\mathbb{Z}_q = \text{the set of remainders mod q.}}$

	Efficient to compute?	**Efficient to <u>invert</u>?**
Addition	YES	YES
Multiplication	YES	YES
Exponentiation	YES	**?**

We found that exponentiation is efficient to invert if q is prime.
If q is not prime, it may be very difficult.

RECAP: A TOY VERSION OF RSA ENCRYPTION

Protocol:



1. Alice generates random $q = rs$ (r,s = primes) and random $x \in \{1,2, \ldots, \phi(q) - 1\}$.
2. She computes $y = x^{-1} \text{ mod } \phi(q)$. (If it doesn't exist, restart.)
3. Bob transmits ciphertext $\text{c} = [m^x \text{ mod } q]$.
4. Alice computes "plaintext" $c^y = m^{xy} = m^1 \, mod \; q$.

$\phi(q)$ = # of elements $a \in \mathbb{Z}_q$ such that $\gcd(a, q) = 1$.

Idea: There is no obvious way for the Adv. to compute y.

PLAN FOR THIS WEEK

1. Diffie-Hellman key-exchange.
2. Formal models of public-key encryption.
3. RSA encryption revisited.
4. The impact of Shor's algorithm on cryptography.

TODAY

DIFFIE-HELLMAN KEY EXCHANGE

SOME REMARKS ON MOTIVATION

With RSA, we used multiplication in $\mathbb{Z}_q$ to build a cryptosystem. Why can't we just use a different algebraic structure instead?

"$\boldsymbol{abstraction}$:"

Merriam-Webster definition:

"the art or process of abstracting"

Well, that was helpful. Trying again:

"$\boldsymbol{abstract}$:"

"expressing a quality apart from an object."

Today we'll define a large class of algebraic structures (groups).

We'll use them to define a cryptosystem (Diffie-Hellman) which is related to, but different from, RSA.

GROUPS

A group **G** is a set with a binary operation ("$\cdot$") which has "multiplication-like" properties. Specifically, it has:

- Associativity: $(a \cdot b) \cdot c = a \cdot (b \cdot c)$
- Identity: There exists e such that $a \cdot e = a$ for all a.
- Inverses: For every a, there exists b such that $a \cdot b = e$.

Examples:

- The real numbers (under addition).
- The set $\mathbb{Z}_q$ (under addition).
- Is $\mathbb{Z}_q$ under multiplication a group?

  No – but the set of all elements of that have multiplicative inverses ($\mathbb{Z}_q^*$), is!

CYCLIC GROUPS

We'll write ab for $(a \cdot b)$, and a^n for $a \cdot a \cdot \cdots \cdot a$ (n times).

An group G is a **cyclic group** if there is a single a such that all elements in G can be expressed as a^i for some i.

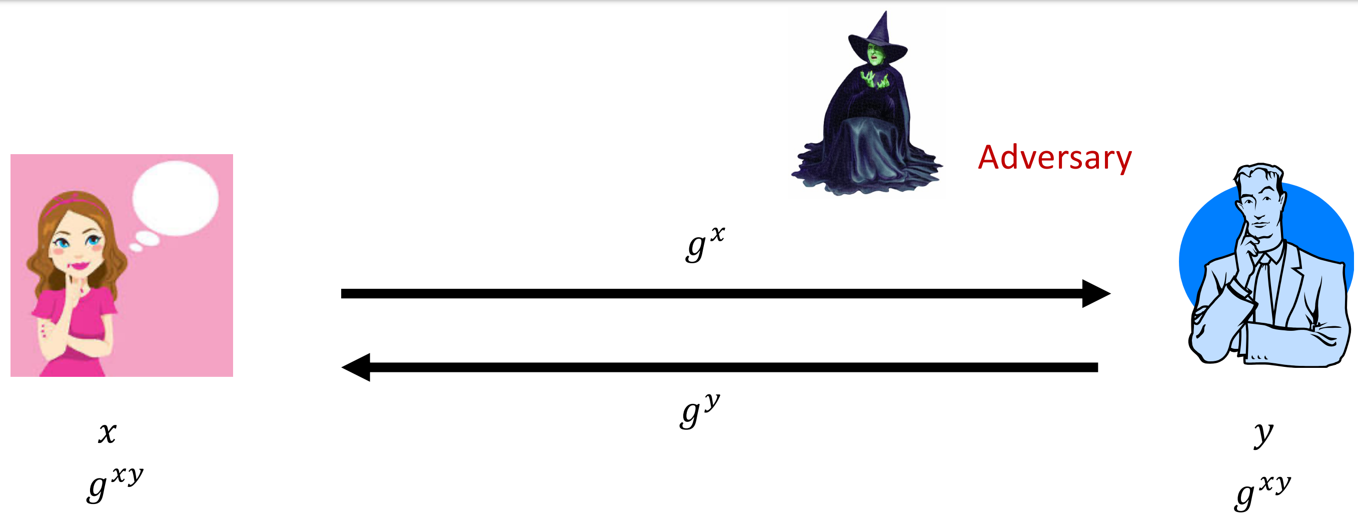
Example: We know (from last week) that the set $\mathbb{Z}_{11}^*$ is cyclic ($a = 2$).

Exercise: Find some q such that $\mathbb{Z}_q^*$ is not cyclic.

KEY EXCHANGE

In this paradigm, Alice and Bob are merely trying to generate a shared random key through public communication.

DIFFIE-HELLMAN KEY EXCHANGE



Let G be a cyclic group, of size t, with generator g. (Public.)

Alice chooses random $x \in \{1,2,\ldots,t\}$ and sends g^x to Bob.

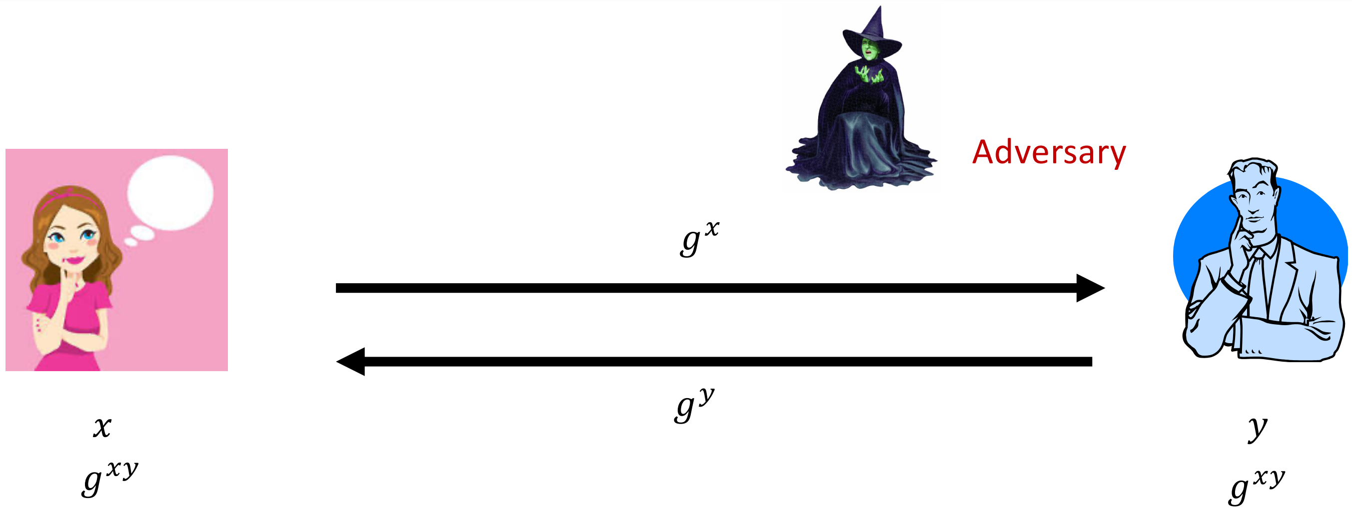
Bob chooses random $y \in \{1,2,\ldots,t\}$ and sends g^y to Alice.

Alice computes $(g^y)^x = g^{xy}$. Bob computes $(g^x)^y = g^{xy}$.

They now have a shared secret!

Exercise:
Compute these values for $G = \mathbb{Z}_{11}^*$, g=2,x=3,y=4.

DIFFIE-HELLMAN KEY EXCHANGE



When is this protocol secure?

Better question: What would we have to assume in order to make this secure?

THE DECISIONAL DIFFIE-HELLMAN PROBLEM

Let $\mathcal{G}$ be an oracle that, on input 1^n, generates a cyclic group (G,q). (t := size of G.)

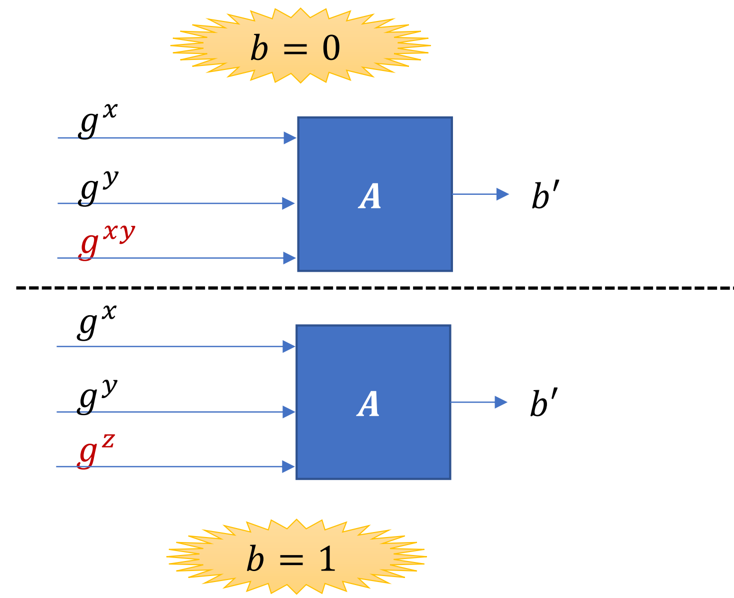
$n =$ "security parameter."
$1^n = 11 \ldots 1$ (n times)

Why?

To satisfy this standard definition:
"**efficient**" = "**polynomial time in the <u>length</u> of the input.**"

THE DECISIONAL DIFFIE-HELLMAN PROBLEM

Let $\mathcal{G}$ be an oracle that, on input 1^n, generates a cyclic group (G,g). (t := size of G.)



Experiment:

1. Draw random $b \leftarrow \{0,1\}$ and $x, y, z \leftarrow \{1, \ldots, t\}$.
2. If $b = 0$, give g^x, g^y, g^{xy} to $\boldsymbol{A}$ (adversary);
3. If $b = 1$, give g^x, g^y, g^z to $\boldsymbol{A}$;
4. $\boldsymbol{A}$ returns $b' \in \{0,1\}$.

Definition. The DDH problem is hard relative to $\mathcal{G}$ if, for any PPT $\boldsymbol{A}$,

$$|\Pr[\boldsymbol{A} = 1 | \mathrm{b} = 0] - \Pr[\boldsymbol{A} = 1 | b = 1]| \leq \mathrm{negl}(n).$$

DIFFIE-HELLMAN KEY EXCHANGE

Chapter 10 proves that if the DDH problem is hard, then the Diffie-Hellman Key Exchange protocol is secure (short proof).

Although the only group we've really worked with so far is $\mathbb{Z}_q^*$, DH can be done with other groups (such as elliptic curves, subsection 8.3.4).

FORMAL MODELS OF PUBLIC-KEY ENCRYPTION

PHILOSOPHY

We want to show that our cryptosystems are secure in a wide range of scenarios.

Therefore, we set up an "experiment," giving the adversary a lot of power in attempting to break the cryptosystem, and ask whether it is still secure.

The adversary always has:

- **Polynomial-time computation ability.**
- **Full knowledge of protocol design.**
- **Access to all public information.**

In some circumstances, we give the adversary even more freedom.

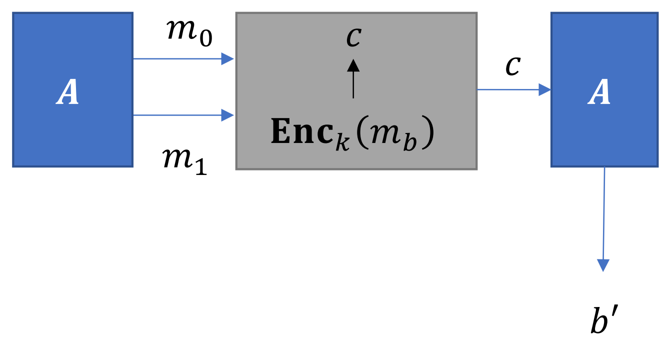
RECALL: "IND" SECURITY FOR SECRET KEY ENCRYPTION

Say our secret-key protocol is $(\mathbf{KeyGen}, \mathbf{Enc}, \mathbf{Dec})$.

Indistinguishability experiment (IND).

1. Sample $k \leftarrow \mathbf{KeyGen}$ and $b \leftarrow \{0,1\}$;
2. A outputs two equal-length messages m_0, m_1;
3. Give $\boldsymbol{A}$ the ciphertext $c \leftarrow \mathbf{Enc}_k(m_b)$;
4. $\boldsymbol{A}$ outputs a bit b'.

We say $\boldsymbol{A}$ wins if $b = b'$.

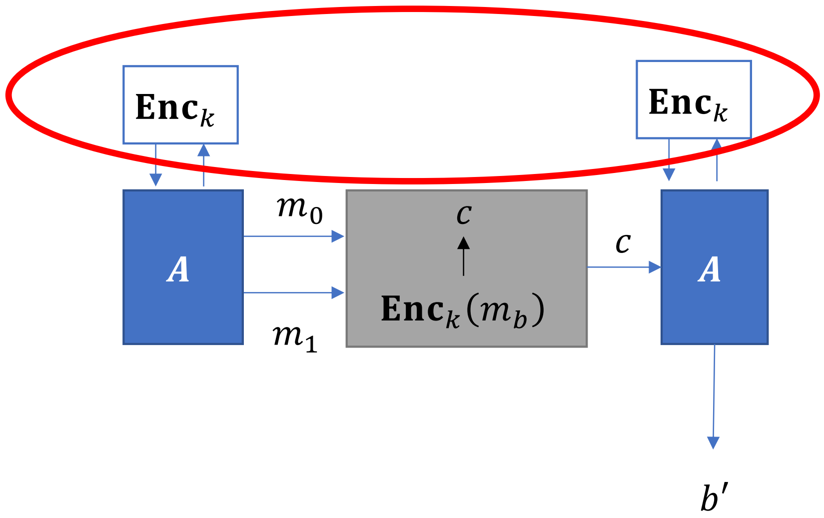


Definition. Our scheme has **indistinguishable ciphertexts** if, for every PPT adversary $\boldsymbol{A}$,

$$\Pr[\boldsymbol{A}\text{ wins}] \leq \frac{1}{2} + \operatorname{negl}(n).$$

RECALL: IND-CPA SECURITY FOR SECRET KEY ENCRYPTION

We give A access to $\mathbf{Enc}_k$ (as an oracle).



Indistinguishability experiment (IND).

1. Sample $k \leftarrow \mathbf{KeyGen}$ and $b \leftarrow \{0,1\}$;
2. A outputs two equal-length messages m_0, m_1;
3. Give $\boldsymbol{A}$ the ciphertext $c \leftarrow \mathbf{Enc}_k(m_b)$;
4. $\boldsymbol{A}$ outputs a bit b'.

We say $\boldsymbol{A}$ wins if $b = b'$.

Definition. Our scheme is **IND-CPA secure** if, for every PPT adversary $\boldsymbol{A}$,

$$\Pr[\boldsymbol{A} \text{ wins}] \leq \frac{1}{2} + \text{negl}(n).$$

IND-CPA SECURITY FOR PUBLIC KEY ENCRYPTION

We give A the public key.

Indisting

1. Samp 1};
2. A out s m_0, m_1;
3. Give
4. $\boldsymbol{A}$ out

We say A

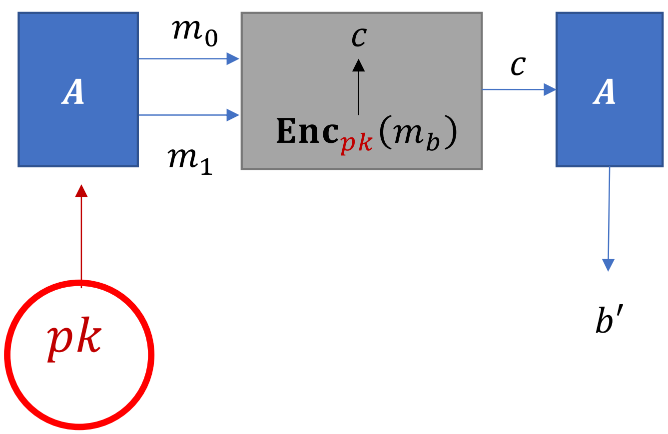
Definit **cure** if, for every PPT adversary $\boldsymbol{A}$,

$$\leq \frac{1}{2} + \text{negl}(n).$$

Question: Why doesn't this strategy always win?:

- $\boldsymbol{A}$ computes $\mathbf{Enc}_{pk}(m_0)$. If it's equal to c she returns $b' = 0$; otherwise, $b' = 1$.

Answer: Enc may use randomness and may encrypt m_0 in multiple ways.



PHILOSOPHY (CONTINUED)

The nuances of the experiment matter.

Sometimes different experiments turn out to be equivalent. Sometimes, not.

CPA = "chosen plaintext attack"

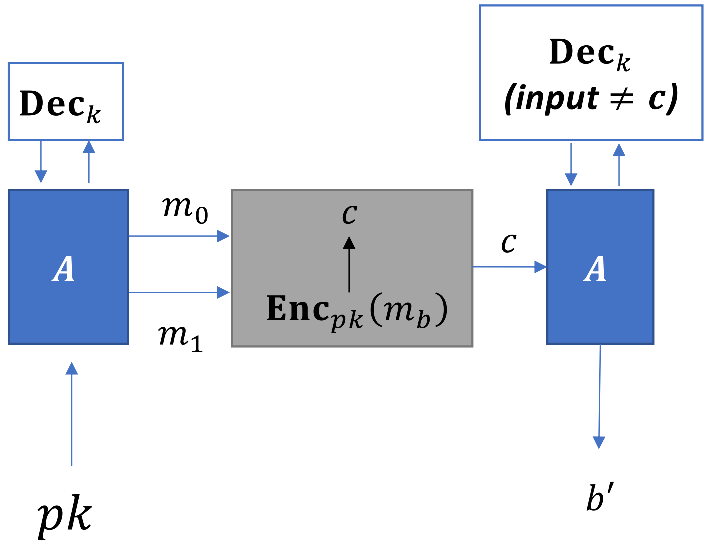
CCA = "chosen ciphertext attack"

IND-CCA SECURITY FOR PUBLIC KEY ENCRYPTION

A has access to a decryption oracle.

1. Sample $pk, sk \leftarrow \mathbf{KeyGen}$ and $b \leftarrow \{0,1\}$;
2. Give pk to $\boldsymbol{A}$, who returns equal-length messages m_0, m_1;
3. Give $\boldsymbol{A}$ the ciphertext $c \leftarrow \mathbf{Enc}_k(m_b)$;
4. $\boldsymbol{A}$ outputs a bit b'.

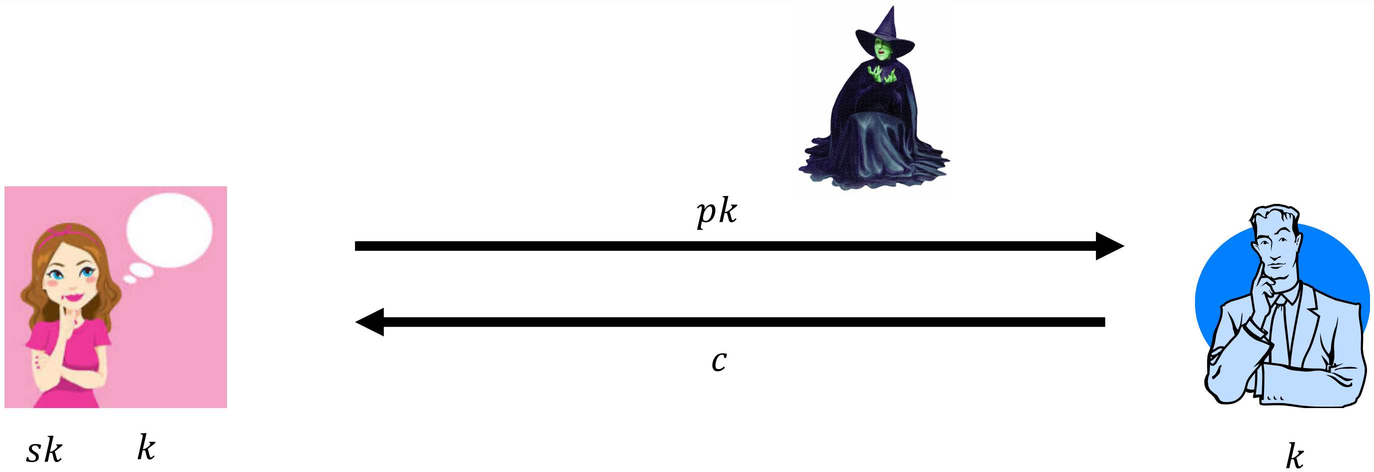
(A is not allowed to decrypt c.)



Definition. Our scheme is **IND-CCA secure** if, for every PPT adversary $\boldsymbol{A}$,

$$\Pr[\boldsymbol{A} \text{ wins}] \leq \frac{1}{2} + \text{negl}(n).$$

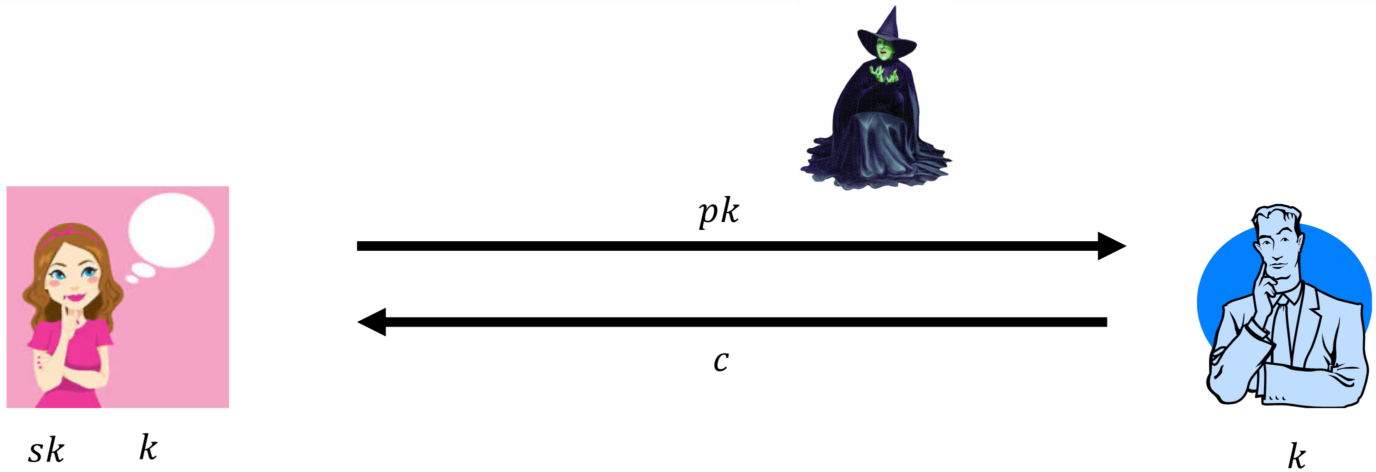
NEW TASK: KEY ENCAPSULATION



Goal: Through public dialogue, share a bit string k that is uniformly random from the perspective of the adversary.

Gen	*input* = 1^n	output = keypair (pk, sk)
Encaps	*input* = 1^n and pk	output = key (k) and ciphertext (c)
Decaps	*input* = sk and c	output = key (k)

NEW TASK: KEY ENCAPSULATION

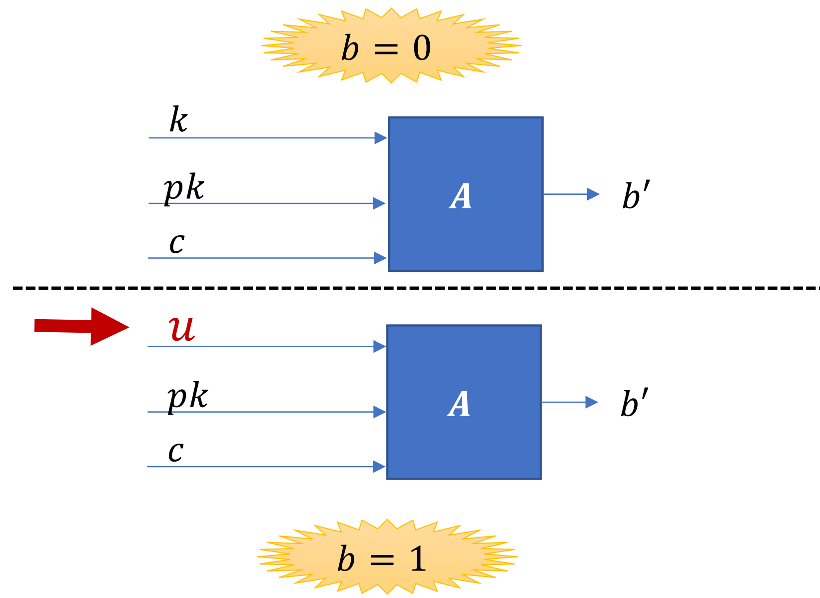


KEM = "key encapsulation mechanism"

One can do KEM → secret-key encryption. The effect is similar to public-key encryption, and can be more efficient.

CPA SECURITY FOR KEMs

1. Carry out the KEM to obtain pk, sk, c, k;
2. Draw random $b \leftarrow \{0,1\}$;
3. If $b = 0$, give k, c, pk to $\boldsymbol{A}$;
4. If $b = 1$, generate a uniformly random bit string u (same length) and give u, c, pk to $\boldsymbol{A}$;
5. $\boldsymbol{A}$ returns b'.



We say that $\boldsymbol{A}$ "wins" if $b = b'$.

Definition. The scheme is **CPA secure** if, for every PPT adversary $\boldsymbol{A}$,

$$\Pr[\boldsymbol{A} \text{ wins}] \leq \frac{1}{2} + \text{negl}(n).$$

SUMMING UP

- We "abstracted" the underlying hardness of RSA encryption, and defined the concept of a "group."
- We defined Diffie-Hellman key exchange (a general framework).
- We stated various formal definitions of security for public-key encryption.

Coming up: A deeper look at RSA.