MATH/CMSC 456 :: UPDATED COURSE INFO

Instructor: Gorjan Alagic (galagic@umd.edu); ATL 3102, office hours: by appointment

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

Webpage: alagic.org/cmsc-456-cryptography-spring-2020/ (slides, reading);

Piazza: piazza.com/umd/spring2020/cmsc456

ELMS: active, slides and reading posted there, assignments will be as well.

Gradescope: active, access through ELMS.

Check these setups asap, and let me know if you run into issues!

TAs (Our spot: shared open area across from IRB 5234)

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

Additional help:

- Chen Bai (cbai1@terpmail.umd.edu) 3:30-5:30pm Tu (2115 ATL, starting Feb 4)
- Bibhusa Rawal (bibhusa@terpmail.umd.edu) 3:30-5:30pm Th (2115 ATL, starting Feb 6)

FIRST HOMEWORK

First homework

- will be posted tonight on ELMS;
- due in one week (11:59pm Thursday February 13th.)

Submission:

- through Gradescope (accessed through ELMS?)
- soon: do a "trial submit" to make sure the system works and you know how to use it;
- replace it with your solutions before the deadline.

Do this ASAP:

- read the problem set completely;
- spend a few minutes thinking about each problem.

This will help you gauge how much time you need to allocate, and how much help you might need.

HOMEWORK RULES AND GUIDELINES

Rules

- collaboration ok, solutions must be written up by yourself, in your own words;
- late homeworks will not be accepted (*no exceptions*, but lowest grade will be dropped.)

Explanations and proofs

- correct answers with no explanation will get a zero score;
- explain your ideas clearly and completely;
- write in complete sentences, use correct and complete mathematical notation (as in lectures and book);
- proofs need to be rigorous, clear, and complete (consider all cases, prove counterexamples, etc.)

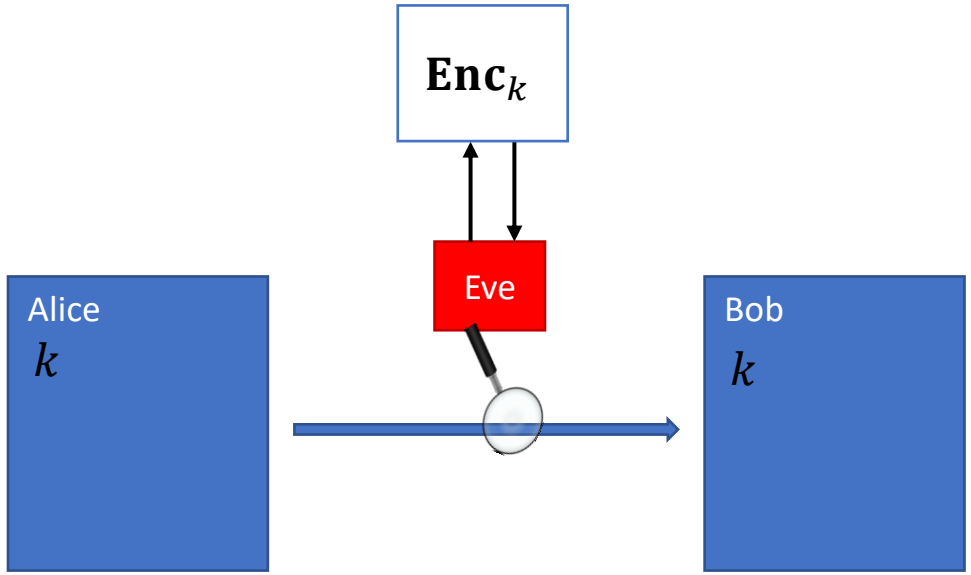
Suggestions

- work on your own at least some of the time for each assignment
- work in 25+ minute chunks of uninterrupted, distraction-free, device-free time
- develop intuition: try lots of examples, ask yourself questions, “play” with the concepts

RECAP: EXPANDING OUR MODEL

So far...

- our model still grants adversary very little power;
- they are only a passive observer;
- in real world, they can do much more!



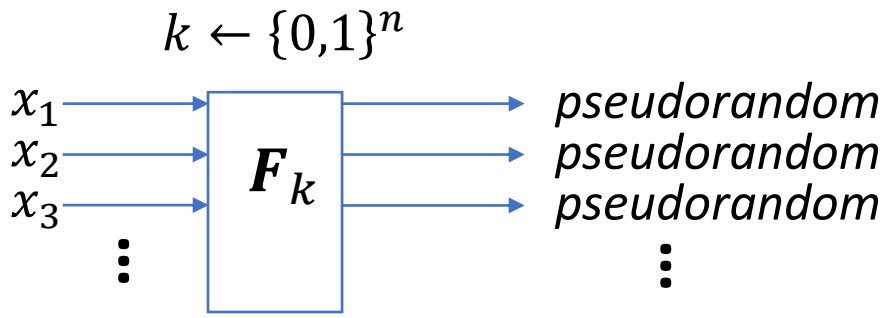
For example: they can *interrogate systems*.

- try to connect to some authorized system;
- guess passwords and see what happens;
- send transmissions and see if they decrypt to something;
- use real world power over parties to get them to send encrypted messages.

How do we capture things like this in our framework? ***Adversaries with oracles.***

RECAP: PSEUDORANDOM FUNCTIONS

A more powerful primitive: pseudorandom *functions*.



It's a PT-computable family:

key:
- choose uniformly at random
- keep secret!

$$F: \{0,1\}^n \times \{0,1\}^m \rightarrow \{0,1\}^\ell$$

input: choose any way you want

output: will look pseudorandom

Given a key $k \in \{0,1\}^n$, we get a function like this: $F_k: \{0,1\}^m \rightarrow \{0,1\}^\ell$.

Trivial: construct **PRG** from **PRF**.

RECAP: PRFs from PRGs

Can you build a PRF from a PRG?

Let $\boldsymbol{G}: \{0,1\}^n \to \{0,1\}^{2n}$ be a PRG, and define:

$\boldsymbol{G}_0: \{0,1\}^n \to \{0,1\}^n$ by $\boldsymbol{G}_0(x) = \boldsymbol{G}(x)|_1^n$

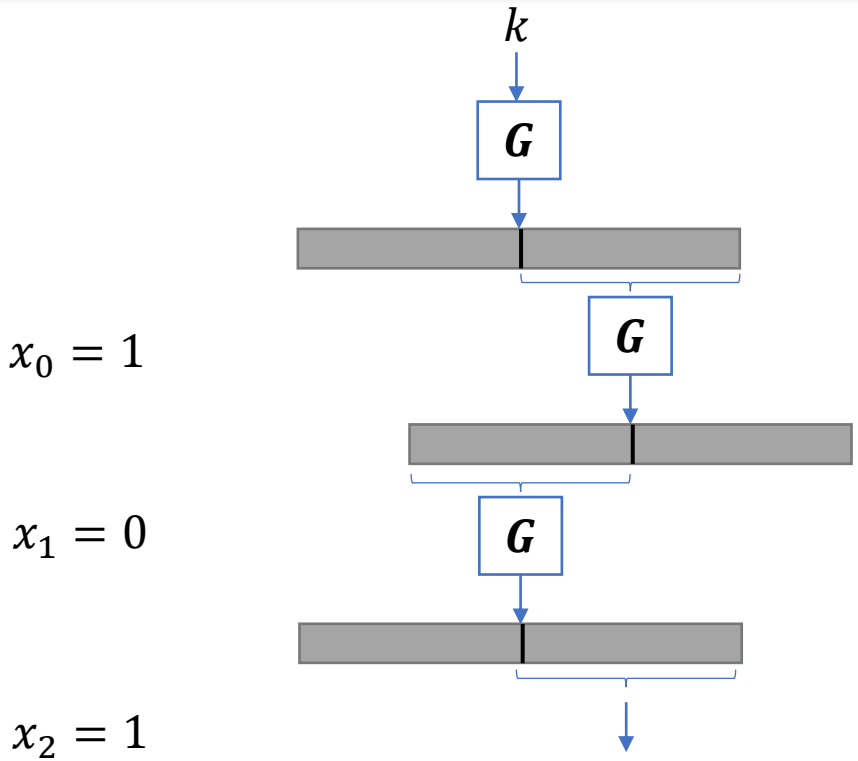
$\boldsymbol{G}_1: \{0,1\}^n \to \{0,1\}^n$ by $\boldsymbol{G}_0(x) = \boldsymbol{G}(x)|_{n+1}^{2n}$

"GGM PRF"

$$\boldsymbol{F}_k(x) = G_{x_n}(G_{x_n}(\cdots(G_{x_1}(G_{x_0}(k)))\cdots)$$

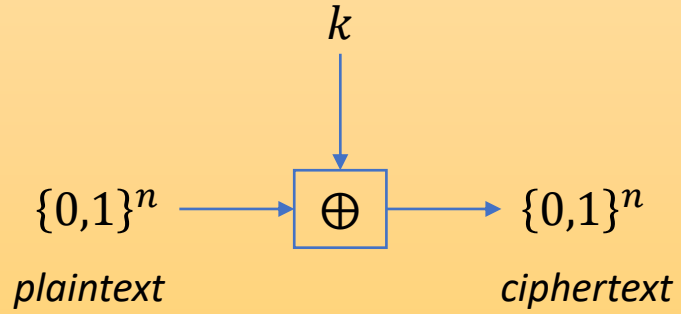
Example:

- suppose n=3
- compute $\boldsymbol{F}_k(101)$.



RECAP: PRF ENCRYPTION

One-time pad

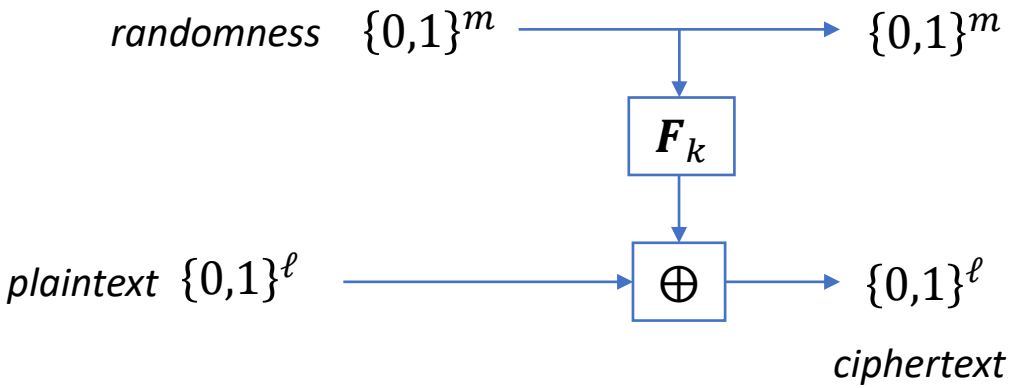


What's a PRF good for?

Lots of things! Like really powerful encryption:

Construction (PRF encryption). Let $\boldsymbol{F}: \{0,1\}^n \times \{0,1\}^m \to \{0,1\}^\ell$ be a PRF. Define a scheme:

- **KeyGen**: sample a PRF key $k \leftarrow \{0,1\}^n$;
- **Enc**: on input a message $m \in \{0,1\}^\ell$, sample $r \leftarrow \{0,1\}^m$ and output $(r, \boldsymbol{F}_k(r) \oplus m)$;
- **Dec**: on input a ciphertext (r, c), output $c \oplus \boldsymbol{F}_k(r)$.



Some properties

- at its core, there's still OTP
- can send arbitrarily-many messages!
- encryption is now a *randomized* algorithm

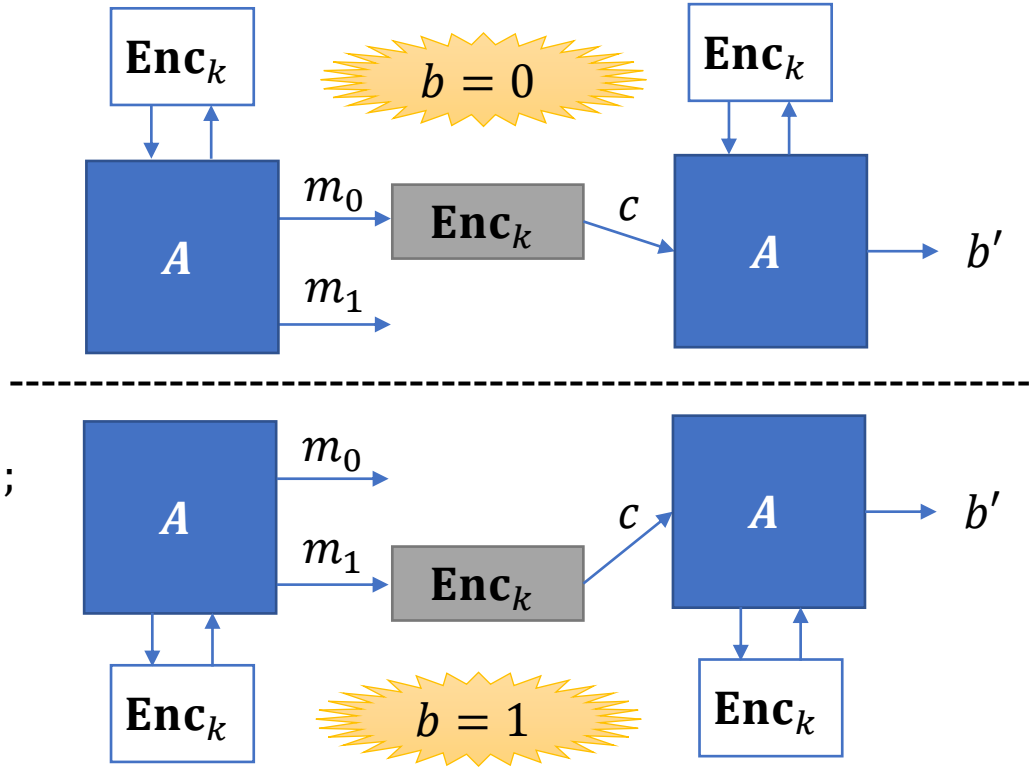
RECAP: IND-CPA

Indistinguishability under Chosen Plaintext Attack.

INDCPA experiment:

1. Sample a key $k \leftarrow \mathbf{KeyGen}$;
2. Give adversary oracle access to $\mathbf{Enc}_k$;
3. $\boldsymbol{A}^{\mathbf{Enc}_k}$ outputs two messages m_0, m_1 with $|m_0| = |m_1|$;
4. Sample a coin $b \leftarrow \{0,1\}$; give $\boldsymbol{A}$ ciphertext $c \leftarrow \mathbf{Enc}_k(m_b)$;
5. $\boldsymbol{A}^{\mathbf{Enc}_k}$ outputs a bit b'.

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



Definition. An encryption scheme $(\mathbf{KeyGen}, \mathbf{Enc}, \mathbf{Dec})$ is **IND-CPA** if, for every PPT adversary $\boldsymbol{A}$,

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

WHAT DOES IT MEAN?

Indistinguishability under Chosen Plaintext Attack.

Definition. An encryption scheme $(\mathbf{KeyGen}, \mathbf{Enc}, \mathbf{Dec})$ is **IND-CPA** if, for every PPT adversary $\boldsymbol{A}$,

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

- this is the "gold standard" : encryption schemes used on the Internet must satisfy it;
- but why? What does IND-CPA mean?
- IND-CPA experiment: just ***one*** possible interaction between an attacker and the scheme;
- what about other ways that "attacker vs scheme" could play out in the real world?
- why don't we have to worry about all those too?

Answer: *semantic security.*

SEMANTIC SECURITY

Recall: "semantic security" is a meaningful, intuitive notion;

It says something like this:

> "observing the ciphertext doesn't help the adversary learn anything ***new*** about the plaintext."

A bit more carefully:

> "no matter what the adversary already knows about the plaintext…
> … observing the ciphertext doesn't help him learn anything more."

Things to capture formally:

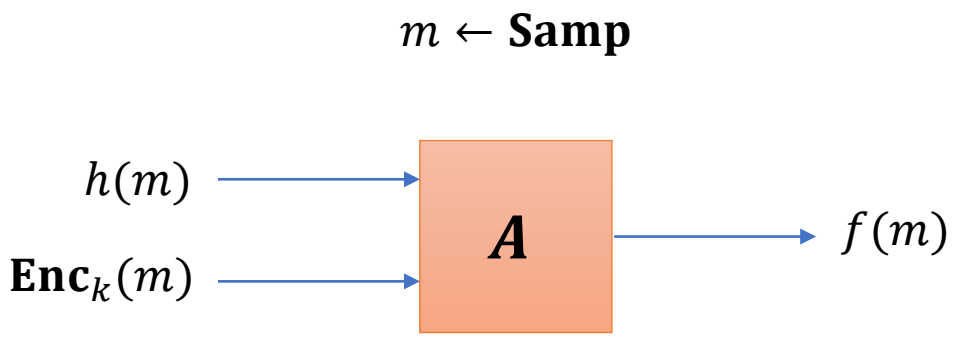
- existing knowledge;
- new knowledge;
- "doesn't help".

SEMANTIC SECURITY

A bit more formally. Pick some scheme $(\mathbf{KeyGen}, \mathbf{Enc}, \mathbf{Dec})$.

An attacker $\boldsymbol{A}$ says they can break it, like this:

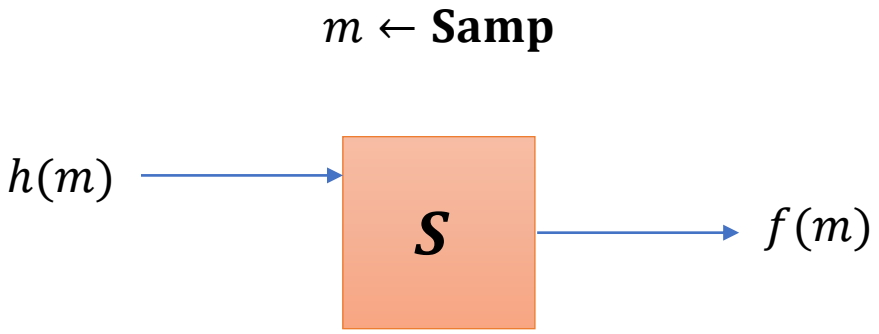
1. A plaintext m is generated somehow;
2. $\boldsymbol{A}$ receives some info $h(m)$ about m;
3. $\boldsymbol{A}$ then observes the ciphertext in transit;
4. Finally, $\boldsymbol{A}$ figures out some info $f(m)$ about m.



Security will mean that:

There exists a "simulator" $\boldsymbol{S}$ such that:

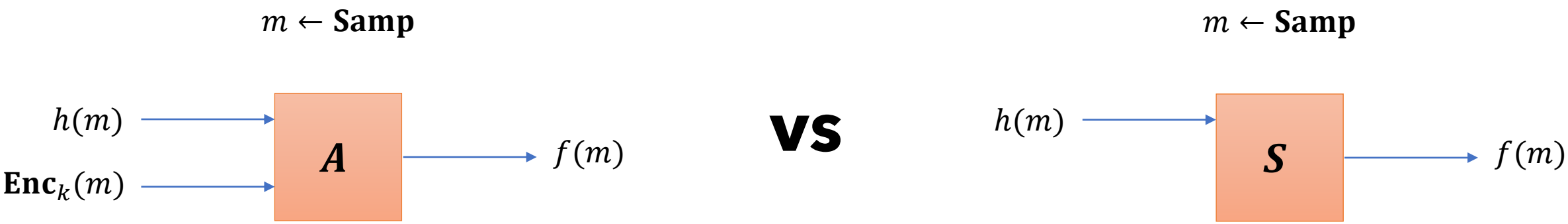
1. If the plaintext m is generated in the same way;
2. and $\boldsymbol{S}$ receives some info $h(m)$ about m;
3. then $\boldsymbol{S}$ also figures out the info $f(m)$ about m.



SEMANTIC SECURITY

Formally:

Definition. An encryption scheme $(\mathbf{KeyGen}, \mathbf{Enc}, \mathbf{Dec})$ is **semantically secure** if, for every PPT algorithm $\boldsymbol{A}$ ("adversary") there exists a PPT algorithm $\boldsymbol{S}$ ("simulator") such that the following holds.



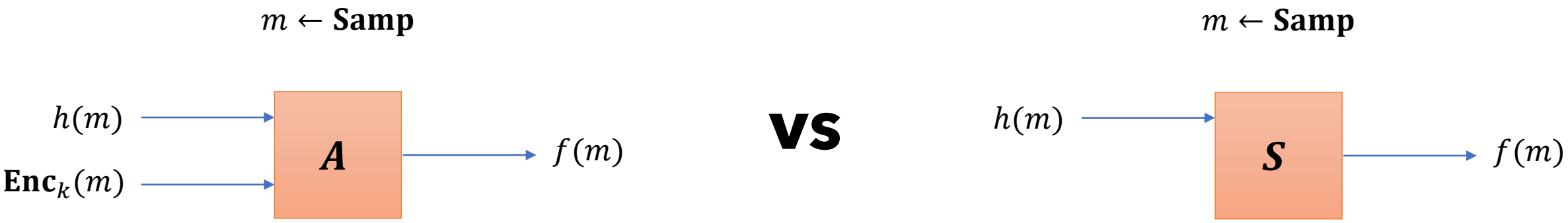
SEMANTIC SECURITY

Formally:

Definition. An encryption scheme $(\mathbf{KeyGen}, \mathbf{Enc}, \mathbf{Dec})$ is **semantically secure** if, for every PPT algorithm $\boldsymbol{A}$ ("adversary") there exists a PPT algorithm $\boldsymbol{S}$ ("simulator") such that the following holds.

For every PPT algorithm **Samp** and every pair of poly-time computable functions h and f,

$$\left|\Pr_{m \leftarrow \mathbf{Samp}}[\boldsymbol{A}(h(m), \mathbf{Enc}_k(m)) = f(m)] - \Pr_{m \leftarrow \mathbf{Samp}}[\boldsymbol{S}(h(m)) = f(m)]\right| \leq \mathrm{negl}(n).$$



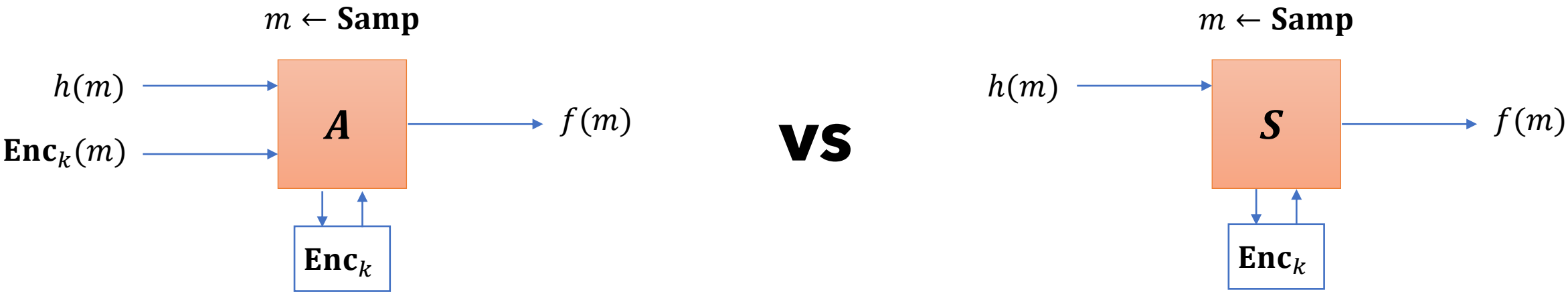
SEMANTIC SECURITY under CHOSEN PLAINTEXT ATTACK

Formally:

Definition. An encryption scheme $(\mathbf{KeyGen}, \mathbf{Enc}, \mathbf{Dec})$ is **SEM-CPA secure** if, for every PPT algorithm $\boldsymbol{A}$ ("adversary") there exists a PPT algorithm $\boldsymbol{S}$ ("simulator") such that the following holds.

For every PPT algorithm **Samp** and every pair of poly-time computable functions h and f,

$$\left|\Pr_{m \leftarrow \mathbf{Samp}}[\boldsymbol{A}^{\mathbf{Enc}_k}(h(m), \mathbf{Enc}_k(m)) = f(m)] - \Pr_{m \leftarrow \mathbf{Samp}}[\boldsymbol{S}^{\mathbf{Enc}_k}(h(m)) = f(m)]\right| \leq \mathrm{negl}(n).$$



SEM-CPA vs IND-CPA

Definition. An encryption scheme $(\mathbf{KeyGen}, \mathbf{Enc}, \mathbf{Dec})$ is **SEM-CPA** if, for every PPT algorithm $\boldsymbol{A}$ ("adversary") there exists a PPT algorithm $\boldsymbol{S}$ ("simulator") such that the following holds.

For every PPT algorithm $\mathbf{Samp}$ and every pair of poly-time computable functions h and f,

$$\left|\Pr_{m \leftarrow \mathbf{Samp}}[\boldsymbol{A}^{\mathbf{Enc}_k}(h(m), \mathbf{Enc}_k(m)) = f(m)] - \Pr_{m \leftarrow \mathbf{Samp}}[\boldsymbol{S}^{\mathbf{Enc}_k}(h(m)) = f(m)]\right| \leq \mathrm{negl}(n).$$

This is really messy. IND-CPA is way simpler to work with.

Theorem. **IND-CPA ⇔ SEM-CPA**.

Totally awesome:

- we get the real, meaningful security strength promised by semantic security…
- … but we can use the much simpler and cleaner indistinguishability definition.
- (for example, when doing proofs!)

SEM-CPA = IND-CPA

Theorem. **IND-CPA** ⇔ **SEM-CPA**.

How to prove something like this?

Break it up into:

1. **IND-CPA** ⇒ **SEM-CPA**
2. **SEM-CPA** ⇒ **IND-CPA** .

- claim: #2 is the less interesting direction;
- we want to know that, when we use **IND-CPA**, this is "good enough";
- intuitively, SEM captures a wide range of "adversary vs scheme" experiments…
- … and the IND experiment is just one special case;
- so #2 is also not very surprising.

So let's talk about #1.

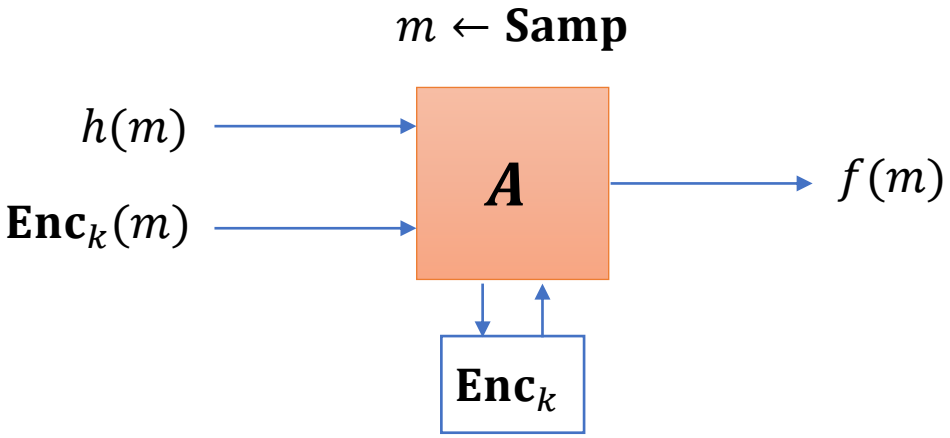
SEM-CPA = IND-CPA

Claim: **IND-CPA** ⇒ **SEM-CPA**.

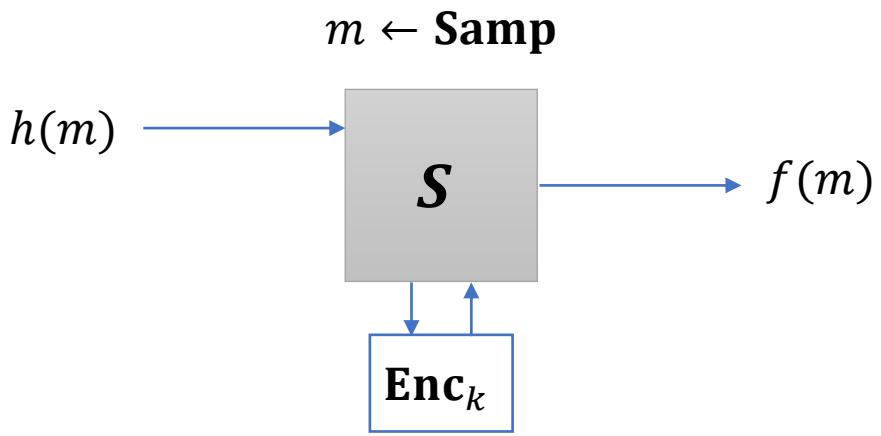
- suppose a scheme is **IND-CPA**;
- let's prove that it must also be **SEM-CPA**;
- direct approach: show how to, given any adversary $\boldsymbol{A}$, construct a simulator $\boldsymbol{S}$.

In pictures:

We have to turn this:



Into this:



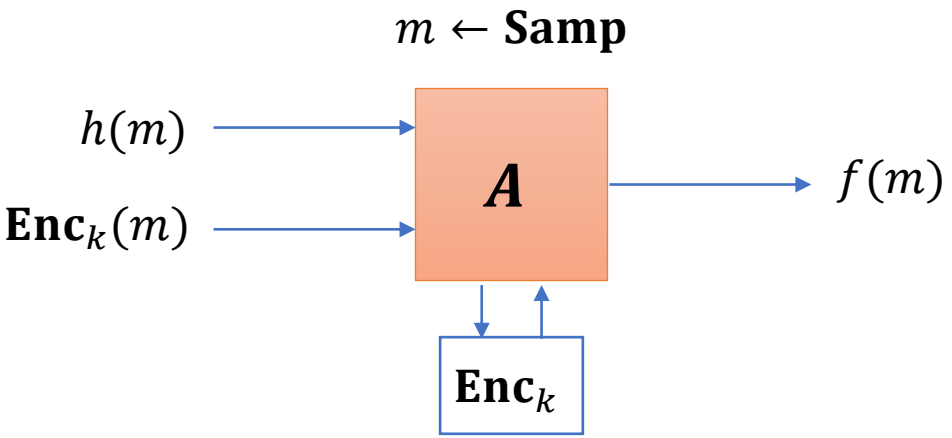
SEM-CPA = IND-CPA

Claim: **IND-CPA** ⇒ **SEM-CPA**.

- suppose a scheme is **IND-CPA**;
- let's prove that it must also be **SEM-CPA**;
- direct approach: show how to, given any adversary $\boldsymbol{A}$, construct a simulator $\boldsymbol{S}$.

In pictures:

We're given this:



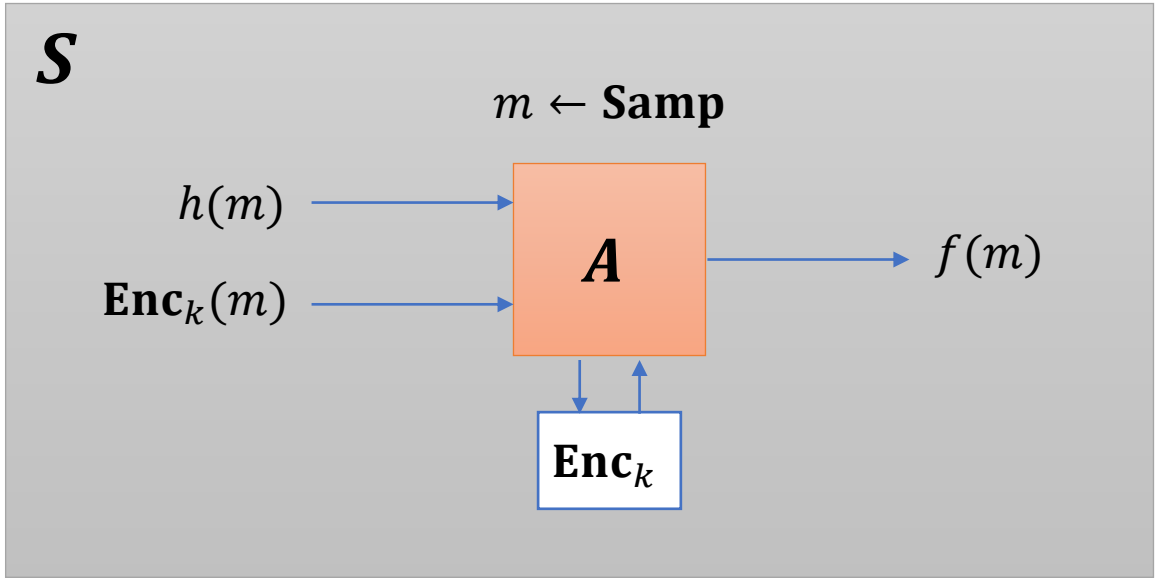
SEM-CPA = IND-CPA

Claim: **IND-CPA** ⇒ **SEM-CPA**.

- suppose a scheme is **IND-CPA**;
- let's prove that it must also be **SEM-CPA**;
- direct approach: show how to, given any adversary $\boldsymbol{A}$, construct a simulator $\boldsymbol{S}$.

In pictures:

We're given this:



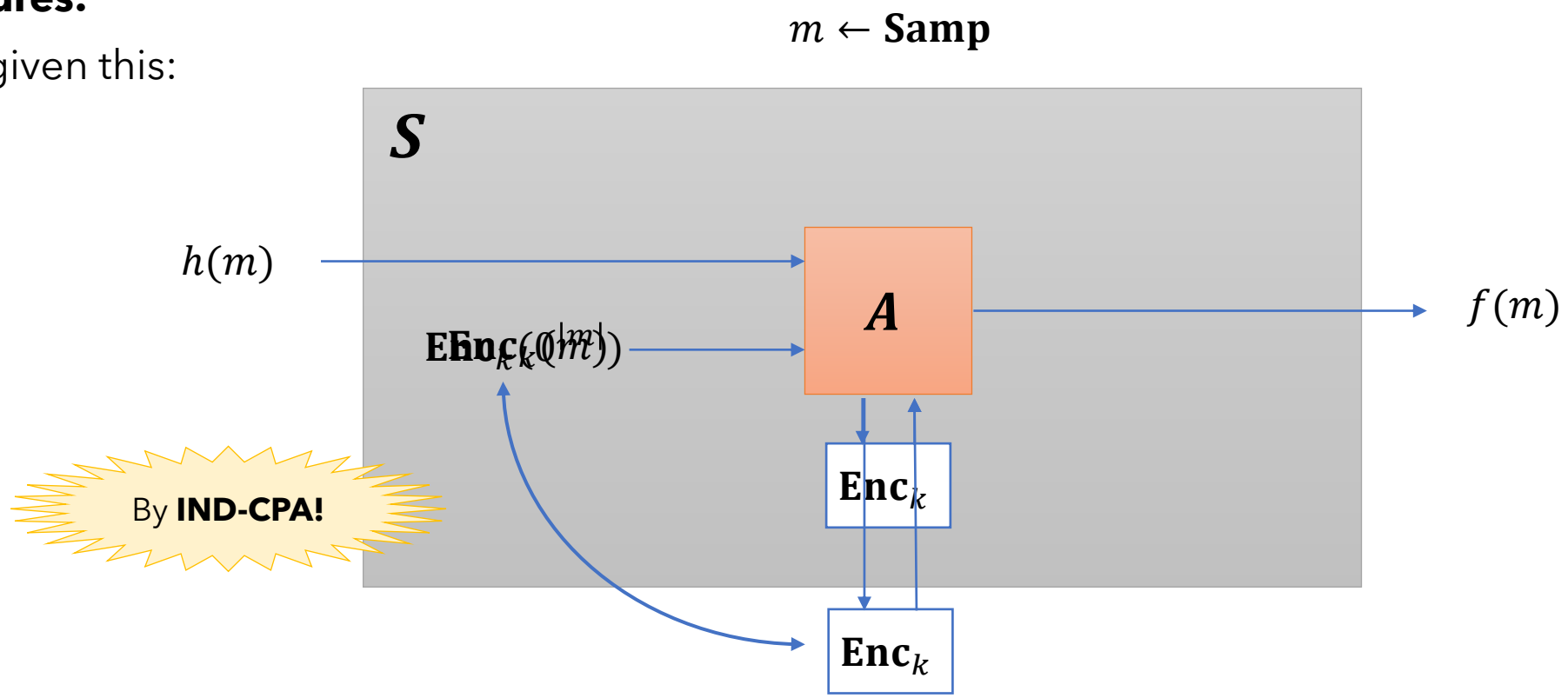
SEM-CPA = IND-CPA

Claim: **IND-CPA** ⇒ **SEM-CPA**.

- suppose a scheme is **IND-CPA**;
- let's prove that it must also be **SEM-CPA**;
- direct approach: show how to, given any adversary $\boldsymbol{A}$, construct a simulator $\boldsymbol{S}$.

In pictures:

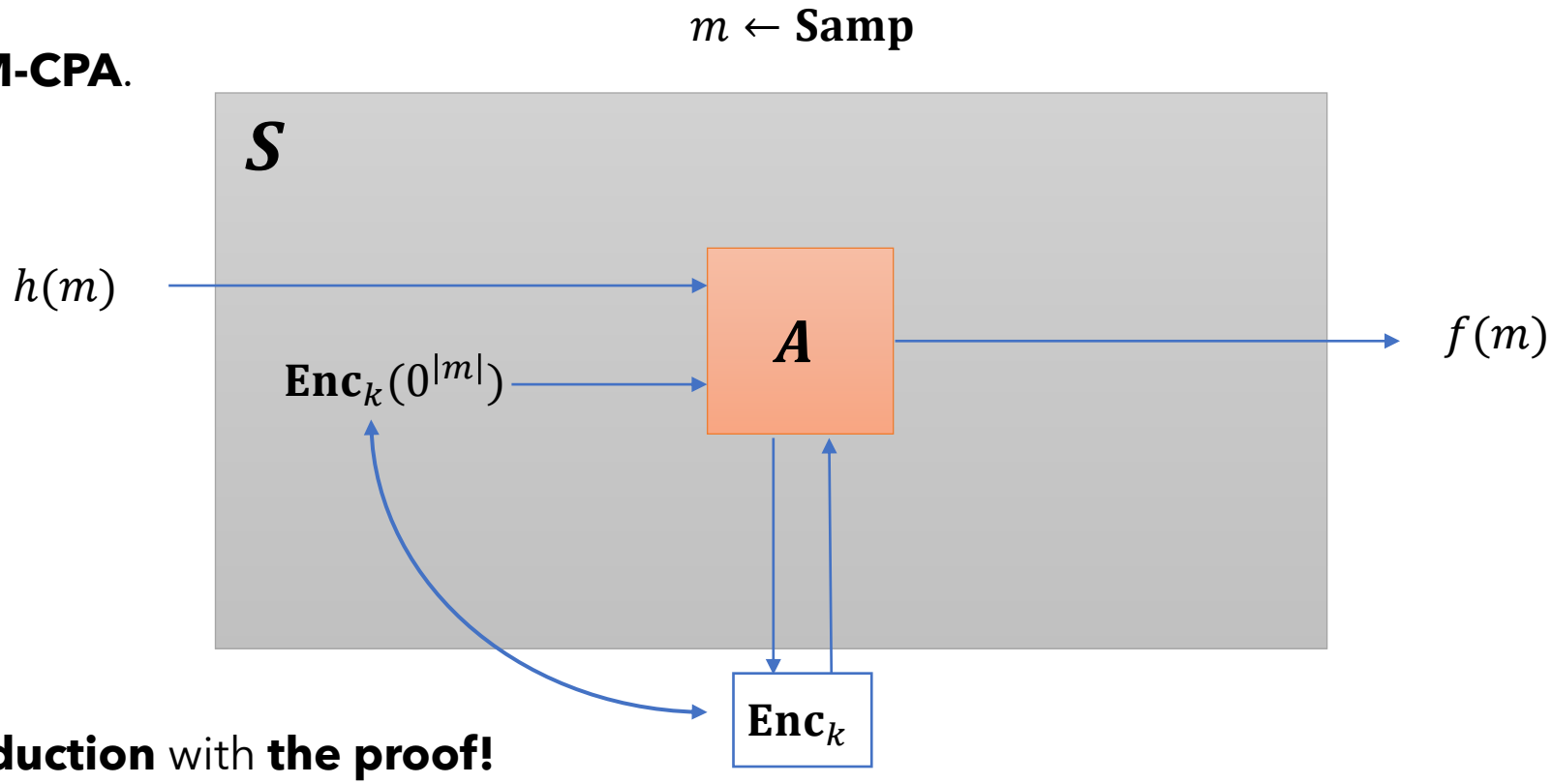
We're given this:



SEM-CPA = IND-CPA

Claim: **IND-CPA** ⇒ **SEM-CPA**.

The reduction:

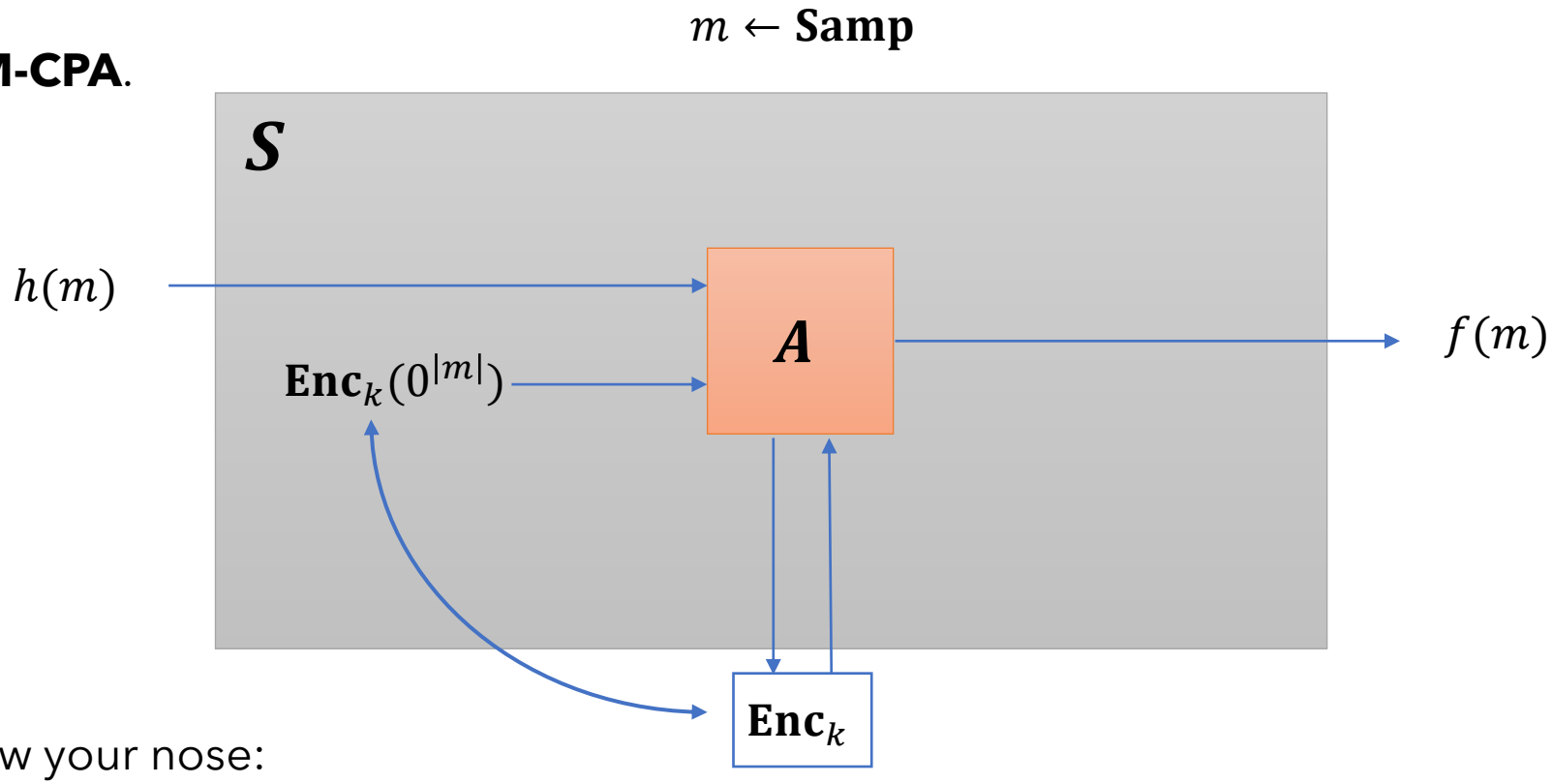


- don't confuse **the reduction** with **the proof!**
- crucial step missing: why does **IND-CPA** allow us to do the ciphertext replacement?
- one way to do this formally: show that, if $\boldsymbol{A}$ can tell the difference,
- i.e., if $\boldsymbol{S}$ and $\boldsymbol{A}$ have noticeably different success probabilities…
- … then you can turn $\boldsymbol{A}$ into a winning **IND-CPA** adversary.

SEM-CPA = IND-CPA

Claim: **IND-CPA** ⇒ **SEM-CPA**.

The reduction:



- for that, you can follow your nose:
- the challenge plaintexts should be m and $0^{|m|}$;
- the post-challenge algorithm gets $h(m)$, and then checks if the output is indeed $f(m)$;
- if YES, guess: challenge was m. If NO, guess: challenge was $0^{|m|}$;

Plenty of details left to check, but you get the idea.

□

SEM-CPA = IND-CPA

Ok, so now we have:

Theorem. **IND-CPA** ⇔ **SEM-CPA**.

Recap why this is awesome:

- we can work with our simple, convenient security definition (**IND-CPA**)…
- … and know that we are capturing the full power of intuitive, meaningful security (**SEM-CPA**).

Actually, **IND-CPA** is even more fantastic than that!

- I tricked you…
- the challenge in IND-CPA only has *one* ciphertext in it;
- what if you want to send lots of messages?
- this is even more apparent when you look at **SEM-CPA**!

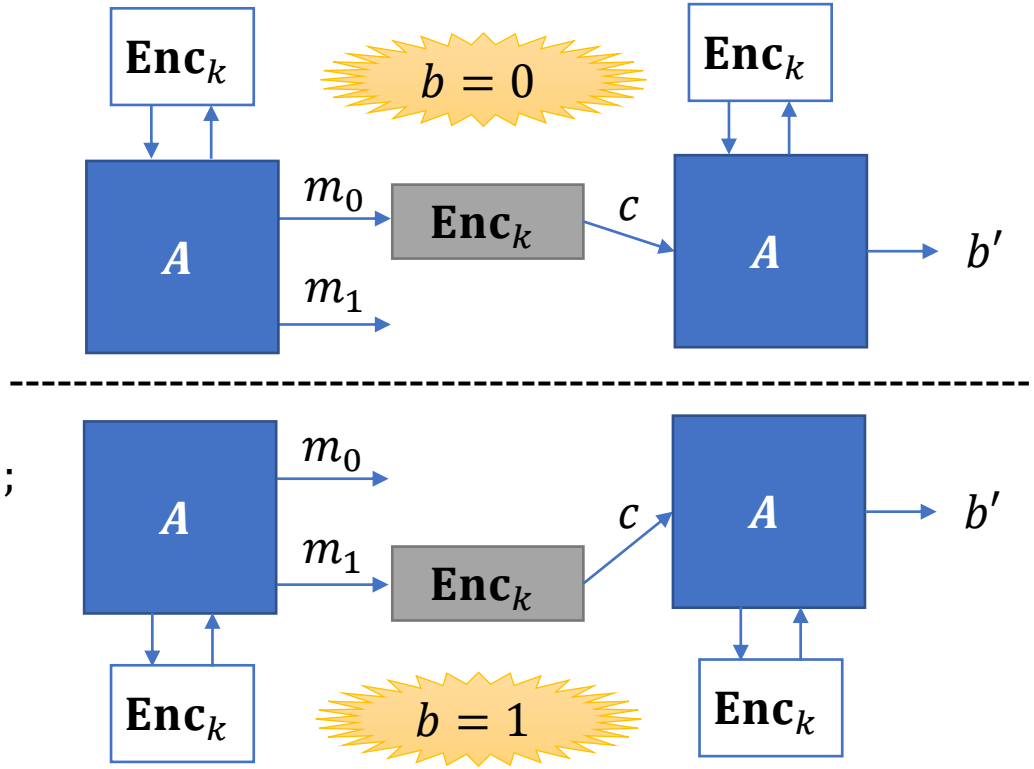
IND-CPA-mult (oh no...)

Indistinguishability under Chosen Plaintext Attack.

INDCPA experiment:

1. Sample a key $k \leftarrow \mathbf{KeyGen}$;
2. Give adversary oracle access to $\mathbf{Enc}_k$;
3. $\boldsymbol{A}^{\mathbf{Enc}_k}$ outputs two messages m_0, m_1 with $|m_0| = |m_1|$;
4. Sample a coin $b \leftarrow \{0,1\}$; give $\boldsymbol{A}$ ciphertext $c \leftarrow \mathbf{Enc}_k(m_b)$;
5. $\boldsymbol{A}^{\mathbf{Enc}_k}$ outputs a bit b'.

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



Definition. An encryption scheme $(\mathbf{KeyGen}, \mathbf{Enc}, \mathbf{Dec})$ is **IND-CPA** if, for every PPT adversary $\boldsymbol{A}$,

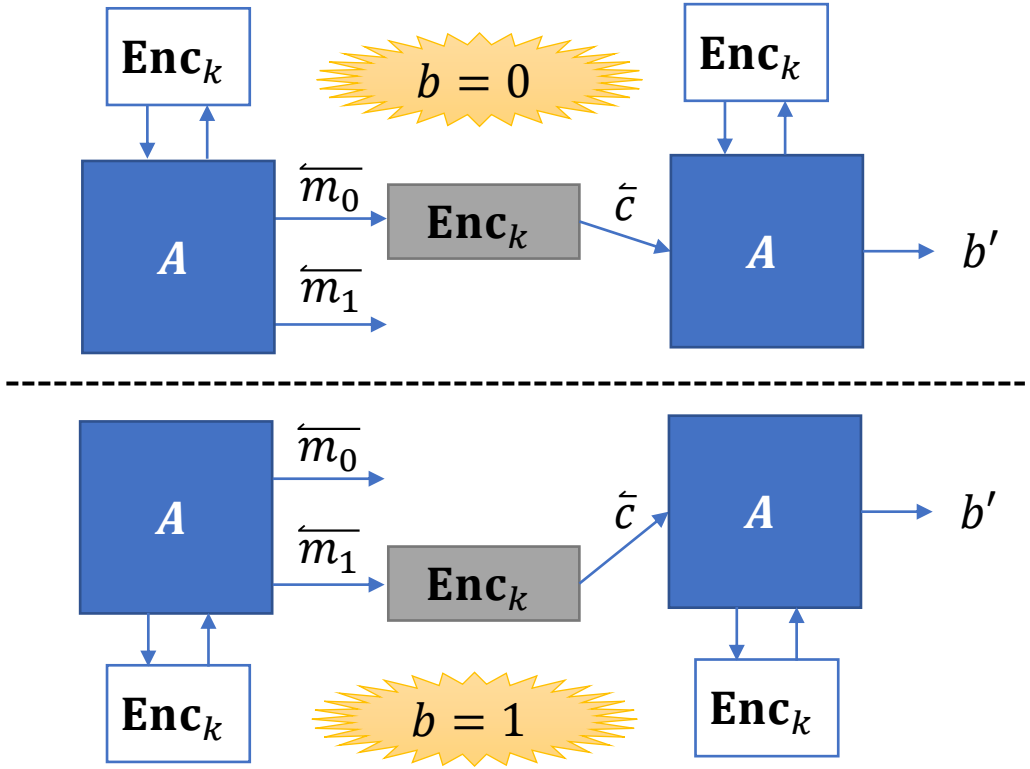
$$\Pr[\boldsymbol{A} \text{ wins INDCPA experiment}] \leq \frac{1}{2} + \text{negl}(n).$$

IND-CPA-mult (oh no...)

INDCPA-mult experiment:

1. Sample a key $k \leftarrow \mathbf{KeyGen}$;
2. Give adversary oracle access to $\mathbf{Enc}_k$;
3. $\boldsymbol{A}^{\mathbf{Enc}_k}$ outputs two message lists $\overrightarrow{m_0}, \overrightarrow{m_1}$;
4. Sample a coin $b \leftarrow \{0,1\}$;
   give $\boldsymbol{A}$ ciphertexts $\vec{c} \leftarrow \mathbf{Enc}_k(\overrightarrow{m_b})$;
5. $\boldsymbol{A}^{\mathbf{Enc}_k}$ outputs a bit b'.

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



Definition. An encryption scheme $(\mathbf{KeyGen}, \mathbf{Enc}, \mathbf{Dec})$ is **IND-CPA-mult** if, for every PPT adversary $\boldsymbol{A}$,

$$\Pr[A \text{ wins } \textbf{INDCPA-mult} \text{ experiment}] \leq \frac{1}{2} + \text{negl}(n).$$

IND-CPA IS GREAT

Easy theorem:

Theorem. **IND-CPA** $\Longrightarrow$ **IND-CPA-mult**.

Yet another reason to love **IND-CPA.**

- ok, let's (finally) do something.
- let's show the PRF scheme is **IND-CPA.**
- by what we just discussed, this will mean that the PRF scheme is also **SEM-CPA** and **IND-CPA-mult.**

REMINDER: PRF ENCRYPTION

Construction (PRF encryption). Let $\boldsymbol{F}: \{0,1\}^n \times \{0,1\}^m \to \{0,1\}^\ell$ be a PRF. Define a scheme:

- $\mathbf{KeyGen}$: sample a PRF key $k \leftarrow \{0,1\}^n$;
- $\mathbf{Enc}$: on input a message $m \in \{0,1\}^\ell$, sample $r \leftarrow \{0,1\}^m$ and output $(r, \boldsymbol{F}_k(r) \oplus m)$;
- $\mathbf{Dec}$: on input a ciphertext (r, c), output $c \oplus \boldsymbol{F}_k(r)$.

Theorem. The PRF scheme is **IND-CPA**.

What's the proof idea?

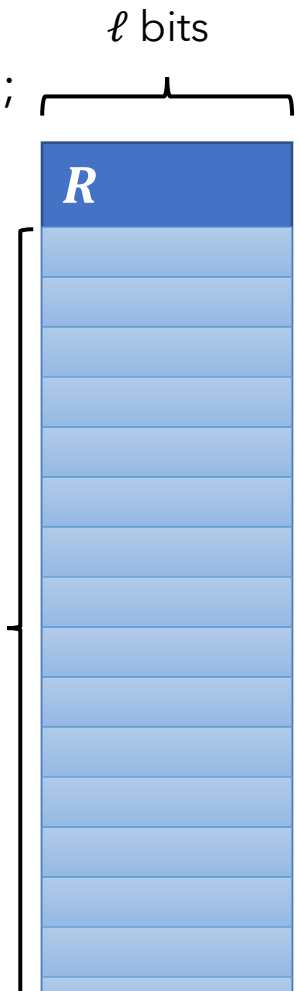
1. by the PRF property, I should be able to replace $\boldsymbol{F}_k$ above with a totally random function $\boldsymbol{R}$…
   … without the adversary noticing the difference.
2. but after that replacement, look at how the scheme works:
   - for each message m…
   - … we pick a random string of the same length as m (specifically, $\boldsymbol{R}(r)$);
   - and we use it to one-time pad m!
3. this is perfectly secret!

PRF SCHEME IS IND-CPA

Theorem. The PRF scheme is **IND-CPA**.

So we need to prove two things:

1. We can replace $\boldsymbol{F}_k: \{0,1\}^m \to \{0,1\}^\ell$ in the PRF scheme with uniformly random $\boldsymbol{R}: \{0,1\}^m \to \{0,1\}^\ell$;
2. The PRF scheme with totally random $\boldsymbol{R}$ is **IND-CPA**.



Important caveats:

- this "$\boldsymbol{R}$-scheme" is **not** an efficient scheme;
- indeed, it takes $\ell \cdot 2^m$ space to store $\boldsymbol{R}$!
- but this is okay: this scheme is *only a proof device*. It does **not** need to be realizable.

Also:

- the "$\boldsymbol{R}$-scheme" isn't really a huge one-time pad…
- … if we happen to sample the same r twice, we will end up reusing the OTP key (bad!)
- but this is ok too: it can only happen with exponentially small probability.

PRF SCHEME IS IND-CPA

PRF scheme $\Pi_{\mathbf{PRF}}$:

- **KeyGen**: sample a PRF key $k \leftarrow \{0,1\}^n$;
- **Enc**: on input a message $m \in \{0,1\}^\ell$, sample $r \leftarrow \{0,1\}^m$ and output $(r, \boldsymbol{F}_k(r) \oplus m)$;
- **Dec**: on input a ciphertext (r, c), output $c \oplus \boldsymbol{F}_k(r)$.

RF scheme $\Pi_{\mathbf{RF}}$

- **KeyGen**: sample a uniformly random function $\boldsymbol{R}: \{0,1\}^m \rightarrow \{0,1\}\hat{}\ell$.
- **Enc**: on input a message $m \in \{0,1\}^\ell$, sample $r \leftarrow \{0,1\}^m$ and output $(r, \boldsymbol{R}(r) \oplus m)$;
- **Dec**: on input a ciphertext (r, c), output $c \oplus \boldsymbol{R}(r)$.

Claim 1. For every PPT $\boldsymbol{A}$,

$$|\Pr[\boldsymbol{A} \text{ wins INDCPA experiment vs } \Pi_{\mathbf{PRF}}] - \Pr[\boldsymbol{A} \text{ wins INDCPA experiment vs } \Pi_{\boldsymbol{RF}}]| \leq \text{negl}(n).$$

Think: "gee, if $\boldsymbol{A}$ really can break $\Pi_{\mathbf{PRF}}$, then they can **also** break $\Pi_{\boldsymbol{RF}}$!"

(and then we'll later show this is impossible as $\Pi_{\boldsymbol{RF}}$ is basically OTP)

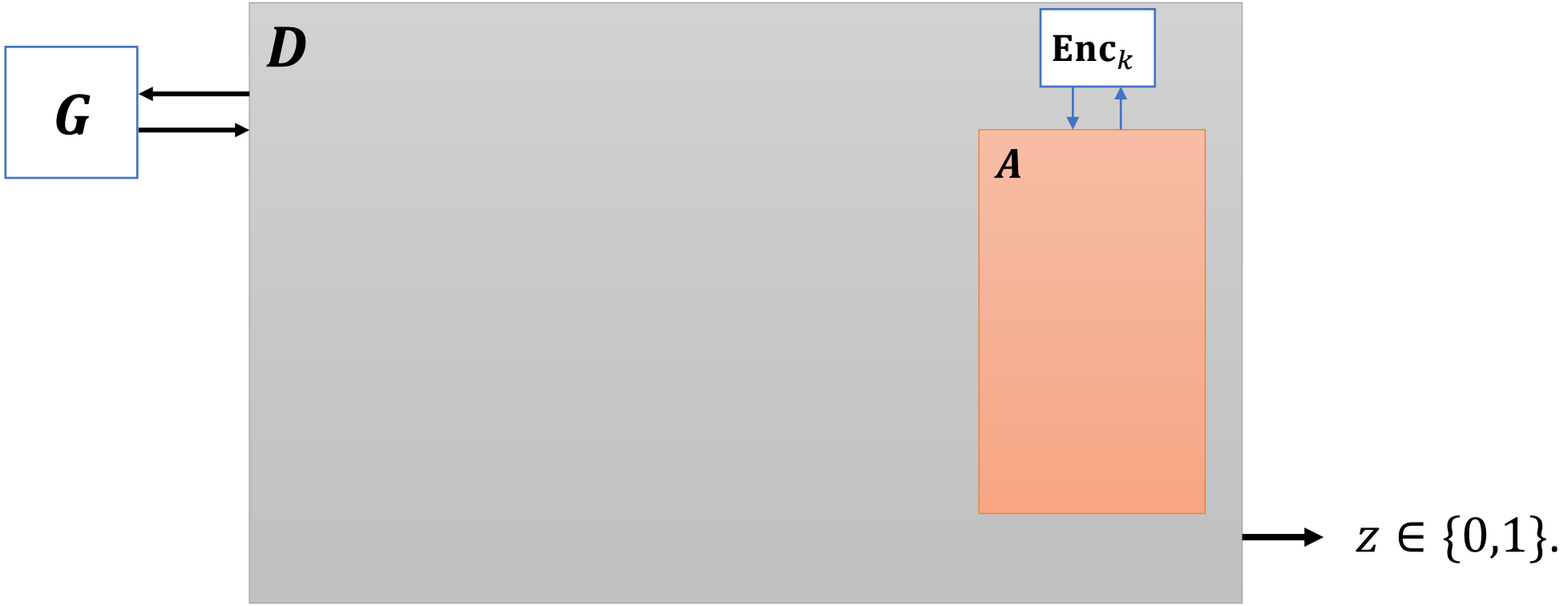
PRF SCHEME IS IND-CPA

Claim 1. For every PPT $\boldsymbol{A}$,

$$|\Pr[\boldsymbol{A} \text{ wins INDCPA experiment vs } \Pi_{\mathbf{PRF}}] - \Pr[\boldsymbol{A} \text{ wins INDCPA experiment vs } \Pi_{\boldsymbol{RF}}]| \leq \text{negl}(n).$$

Strategy: if claim is false, then we can build a distinguisher between **PRF** and **RF** (& violate **PRF** property!)

What's the distinguisher? *It's a simulation of the INDCPA experiment vs* $\boldsymbol{A}$*!*

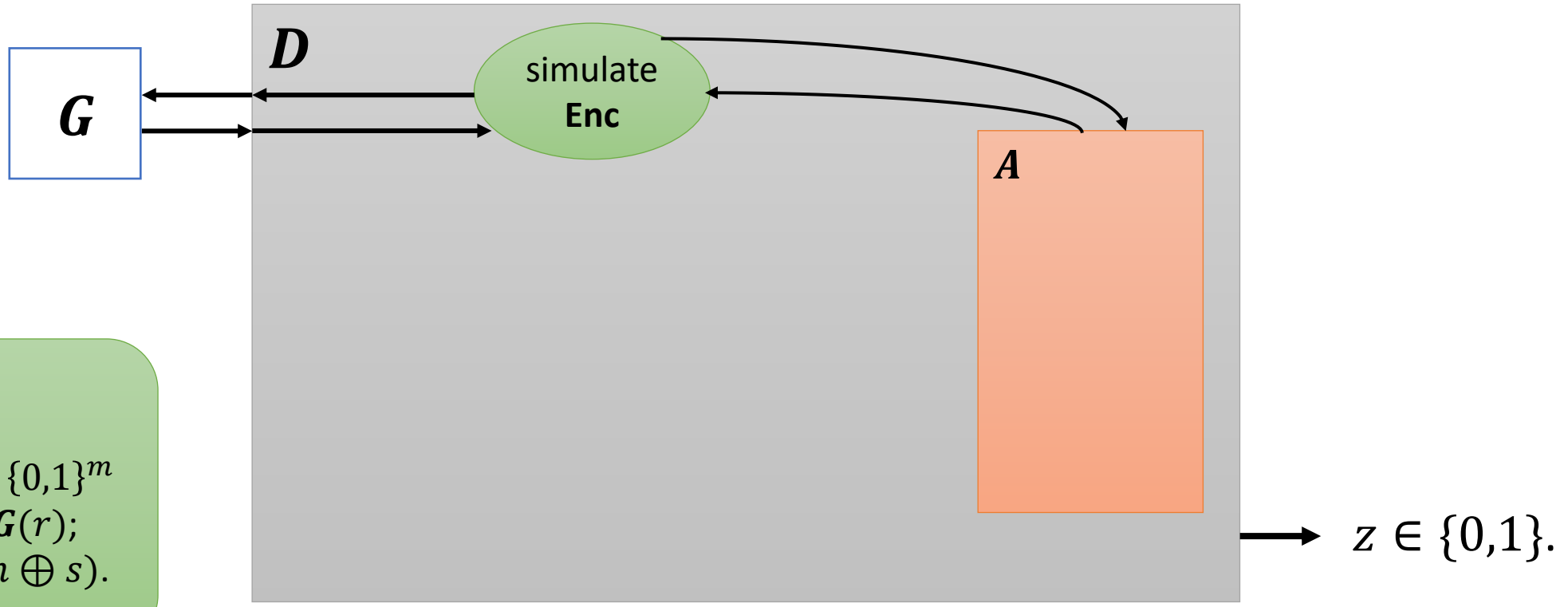


PRF SCHEME IS IND-CPA

Claim 1. For every PPT $\boldsymbol{A}$,
$|\Pr[\boldsymbol{A} \text{ wins INDCPA experiment vs } \Pi_{\mathbf{PRF}}] - \Pr[\boldsymbol{A} \text{ wins INDCPA experiment vs } \Pi_{\boldsymbol{RF}}]| \leq \text{negl}(n)$.

Strategy: if claim is false, then we can build a distinguisher between **PRF** and **RF** (& violate **PRF** property!)

What's the distinguisher? *It's a simulation of the INDCPA experiment vs* $\boldsymbol{A}$*!*



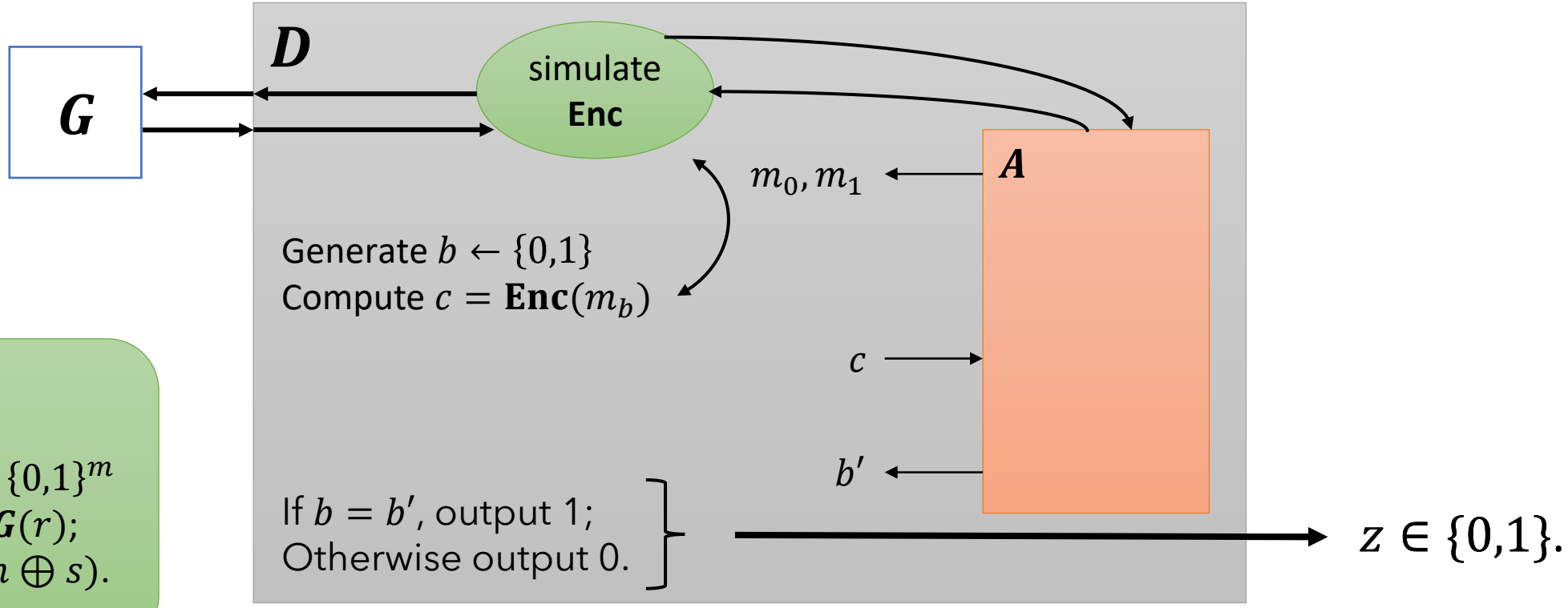
simulate **Enc**:
- on input m;
- sample $r \leftarrow \{0,1\}^m$
- query: $s = \boldsymbol{G}(r)$;
- output $(r, m \oplus s)$.

PRF SCHEME IS IND-CPA

Claim 1. For every PPT $\boldsymbol{A}$,
$|\Pr[\boldsymbol{A} \text{ wins INDCPA experiment vs } \Pi_{\mathbf{PRF}}] - \Pr[\boldsymbol{A} \text{ wins INDCPA experiment vs } \Pi_{\boldsymbol{RF}}]| \leq \text{negl}(n)$.

Strategy: if claim is false, then we can build a distinguisher between **PRF** and **RF** (& violate **PRF** property!)

What's the distinguisher? *It's a simulation of the INDCPA experiment vs* $\boldsymbol{A}$*!*



simulate **Enc**:
- on input m;
- sample $r \leftarrow \{0,1\}^m$
- query: $s = \boldsymbol{G}(r)$;
- output $(r, m \oplus s)$.

PRF SCHEME IS IND-CPA

Claim 1. For every PPT $\boldsymbol{A}$,
$|\Pr[\boldsymbol{A} \text{ wins INDCPA experiment vs } \Pi_{\mathbf{PRF}}] - \Pr[\boldsymbol{A} \text{ wins INDCPA experiment vs } \Pi_{\boldsymbol{RF}}]| \leq \text{negl}(n)$.

Strategy: build a distinguisher between **PRF** and **RF.**

Check:

- If $\boldsymbol{G} = \boldsymbol{F}_k$ for $k \leftarrow \{0,1\}^n$, $\boldsymbol{D}$ correctly simulates the INDCPA experiment vs $\Pi_{\mathbf{PRF}}$…
- … so in that case, $\Pr[\boldsymbol{z} = 1] = \Pr[\boldsymbol{A} \text{ wins vs } \Pi_{\mathbf{PRF}}]$.
- If $\boldsymbol{G} = \boldsymbol{R}$ for $\boldsymbol{R} \leftarrow \{\text{functions from } \{0,1\}^m \text{ to } \{0,1\}^\ell\}$, $\boldsymbol{D}$ correctly simulates the INDCPA experiment vs $\Pi_{\mathbf{RF}}$;
- … so in that case, $\Pr[\boldsymbol{z} = 1] = \Pr[\boldsymbol{A} \text{ wins vs } \Pi_{\mathbf{PR}}]$.

It follows that the distinguishing advantage of $\boldsymbol{D}$ is

$$\left|\Pr[\boldsymbol{D}^{\boldsymbol{F}_k} = 1] - \Pr[\boldsymbol{D}^{\boldsymbol{R}} = 1]\right| = |\Pr[\boldsymbol{A} \text{ wins vs } \Pi_{\mathbf{PRF}}] - \Pr[\boldsymbol{A} \text{ wins vs } \Pi_{\mathbf{RF}}]|$$

which must be negligible by **PRF** property.

PRF SCHEME IS IND-CPA

Claim 2. For every PPT $\boldsymbol{A}$,

$$|\Pr[\boldsymbol{A} \text{ wins INDCPA experiment vs } \Pi_{\boldsymbol{RF}}]| \leq \frac{1}{2} + \text{negl}(n).$$

Strategy?

- consider a run of the INDCPA experiment:
- it involves a polynomial number $p(n)$ of calls to $\mathbf{Enc}_{\boldsymbol{R}}$; each call queries $\boldsymbol{R}$ at a random input r;
- one of these calls is the *challenge*; let's call the random input for that r^*.
- **Event $\boldsymbol{E}$**: for some r, $r = r^*$.
  - occurs with negligible probability: $\Pr[\boldsymbol{E}] \leq p(n)/2^m$.
  - so we can ignore it.
- **Event $\overline{\boldsymbol{E}}$**: for all r, $r \neq r^*$.
  - occurs with probability almost 1 (since $\Pr[\overline{\boldsymbol{E}}] = 1 - \Pr[E]$)
  - good for us: $\boldsymbol{R}(r^*)$ is uniformly random and independent of rest of experiment…
  - … which means the challenge was OTP-encrypted!

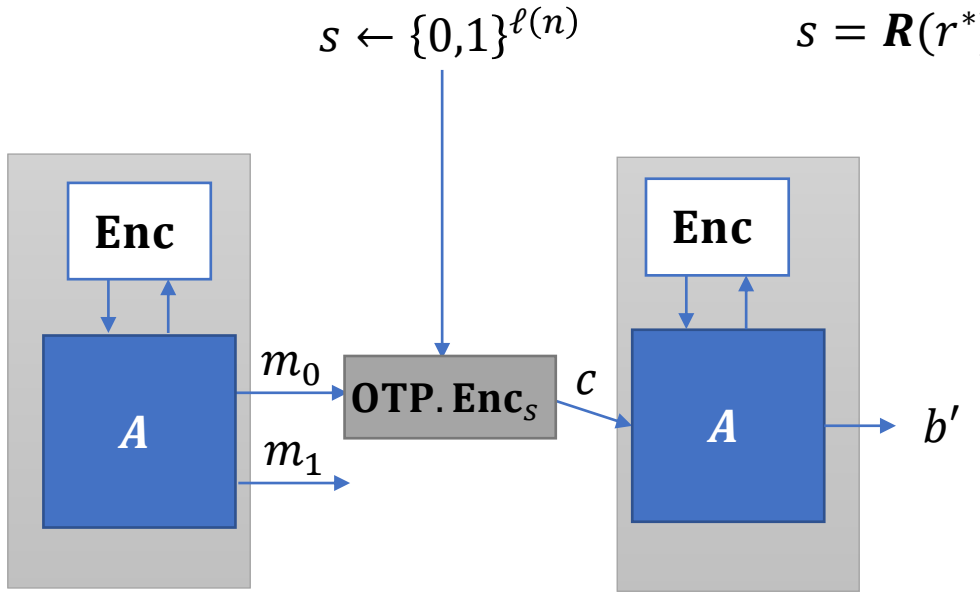
PRF SCHEME IS IND-CPA

Claim 2. For every PPT $\boldsymbol{A}$,

$$|\Pr[\boldsymbol{A} \text{ wins INDCPA experiment vs } \Pi_{\boldsymbol{RF}}]| \leq \frac{1}{2} + \text{negl}(n).$$

Event $\bar{\boldsymbol{E}}$: for all r, $r \neq r^*$.

- occurs with probability almost 1 (since $\Pr[\bar{\boldsymbol{E}}] = 1 - \Pr[E]$)
- good for us: $\boldsymbol{R}(r^*)$ is uniformly random and independent of rest of experiment…
- … which means the challenge was OTP-encrypted!



Conclusion:

$$\Pr[\boldsymbol{A} \text{ wins}|\bar{\boldsymbol{E}}] = 1/2.$$

PRF SCHEME IS IND-CPA

Claim 2. For every PPT $\boldsymbol{A}$,

$$|\Pr[\boldsymbol{A} \text{ wins INDCPA experiment vs } \Pi_{\boldsymbol{RF}}]| \leq \frac{1}{2} + \text{negl}(n).$$

Putting things together:

$$\begin{aligned}
|\Pr[\boldsymbol{A} \text{ wins}]| &= \Pr[\boldsymbol{A} \text{ wins} \mid \boldsymbol{E}] \cdot \Pr[\boldsymbol{E}] + \Pr[\boldsymbol{A} \text{ wins} \mid \bar{\boldsymbol{E}}] \cdot \Pr[\bar{\boldsymbol{E}}] \\
&\leq 1 \cdot \frac{p(n)}{2^m} + \frac{1}{2} \cdot \left(1 - \frac{p(n)}{2^m}\right) \\
&\leq \frac{1}{2} + \frac{p(n)}{2^m} - \frac{p(n)}{2^{m+1}} \\
&\leq \frac{1}{2} + \text{negl}(n).
\end{aligned}$$

PRF SCHEME IS IND-CPA

Claim 1. For every PPT $\boldsymbol{A}$,

$$|\Pr[\boldsymbol{A} \text{ wins INDCPA experiment vs } \Pi_{\mathbf{PRF}}] - \Pr[\boldsymbol{A} \text{ wins INDCPA experiment vs } \Pi_{\boldsymbol{RF}}]| \leq \text{negl}(n).$$

Claim 2. For every PPT $\boldsymbol{A}$,

$$|\Pr[\boldsymbol{A} \text{ wins INDCPA experiment vs } \Pi_{\boldsymbol{RF}}]| \leq \frac{1}{2} + \text{negl}(n).$$

Putting the two claims together:

$$|\Pr[\boldsymbol{A} \text{ wins vs } \Pi_{\mathbf{PRF}}]| \leq |\Pr[\boldsymbol{A} \text{ wins vs } \Pi_{\mathbf{RF}}]| + |\Pr[\boldsymbol{A} \text{ wins vs } \Pi_{\mathbf{PRF}}] - \Pr[\boldsymbol{A} \text{ wins vs } \Pi_{\boldsymbol{RF}}]|$$

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

It follows that the PRF scheme is **IND-CPA**.

□