CIS 5560

Cryptography
Lecture 3

Announcements

- **HW 0 is out;** due Friday, Jan 30 at 5PM on Gradescope
  - Covers modular arithmetic, basic probability, Caesar cipher
- Office Hours:
  - Pratyush: Monday Mondays 10-11AM, Wednesday 12-1PM

Recap of Last Lecture

- Secure communication Threat Model
- Symmetric-key definition
- Perfect secrecy
- Perfect indistinguishability
- Probability review

Shannon's Perfect Secrecy Definition

$\forall m \in \mathscr{M}, \forall c \in \mathscr{C}, M$ is adversary's guess

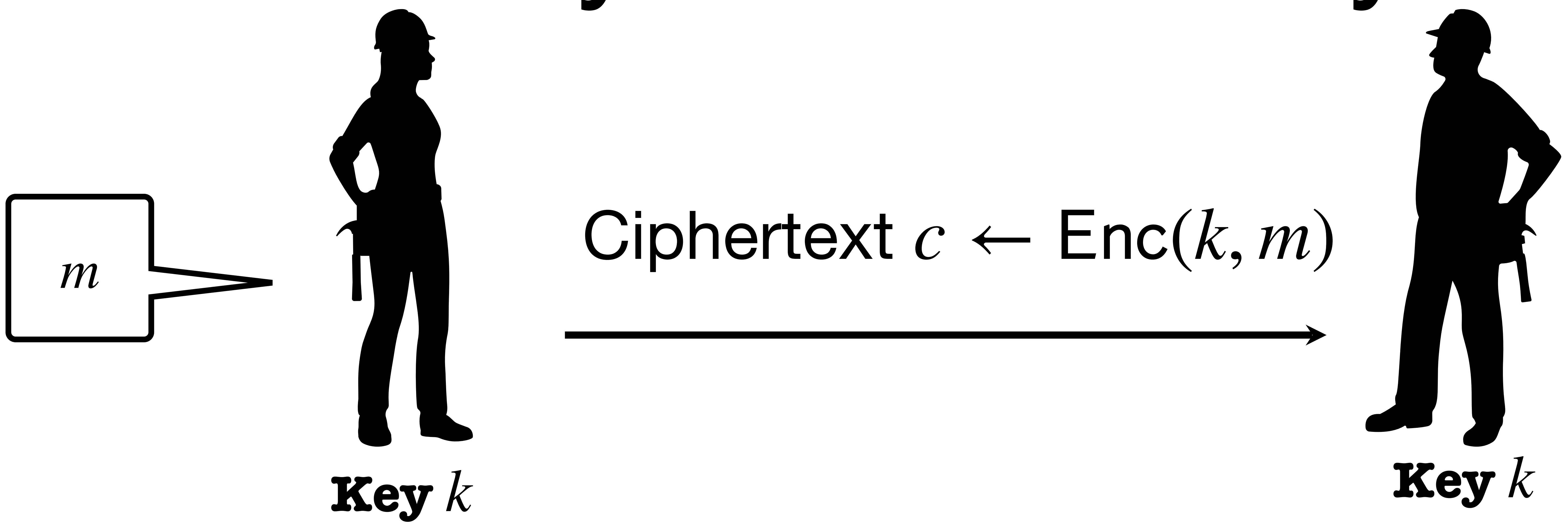
$$\Pr[M = m \,|\, \mathsf{Enc}(\mathscr{K}, m) = c] = \Pr[M = m]$$

after before

✓ **CT reveals no info about PT**

But this def is difficult to work with:
How to prove that ciphertext reveals no info?

Key notion: Symmetric-Key Encryption



<u>**Three (possibly randomized) polynomial-time algorithms:**</u>

Key Generation Algorithm: Gen$(1^\lambda) \rightarrow k$

Has to be randomized (why?)

Encryption Algorithm: Enc$(k, m) \rightarrow c$

Decryption Algorithm: Dec$(k, c) \rightarrow m$

One-Time Pad

The One-time Pad Construction:

Gen: Choose an n-bit string k at random, i.e. $k \leftarrow \{0,1\}^n$

Enc(k, m) with $\mathscr{M} = \{0,1\}^n$: Output $c = m \oplus k$

Dec(k, c): Output $m = c \oplus k$

Perfect Secrecy has its Price

THEOREM: For any perfectly secure encryption scheme,

$$|\mathscr{K}| \geq |\mathscr{M}|$$

Today's Lecture

- Indistinguishability
- Negligible functions
- Pseudorandom generators
- Semantic security
- PRGs → Semantically-secure encryption

Defn II: Perfect Secrecy'

For every m, m'

Probability that c encrypts m (with random key k)

=

Probability that c encrypts m' (with diff. key k')

Hence every ciphertext is equally likely to decrypt to a given message

$$\forall m, m' \in \mathcal{M}, c \in \mathcal{C}$$
$$\Pr_{k \leftarrow \mathcal{K}}[\text{Enc}(k, m) = c] = \Pr_{k' \leftarrow \mathcal{K}}[\text{Enc}(k', m') = c]$$

Defn I ⇒ Defn II

Intuition:

If a ciphertext reveals no information about plaintext, it can equally likely be an encryption for m or m'

Defn III: Perfect Indistinguishability

For every m, m', and for every "distinguishing" predicate ϕ

Output of ϕ on encryption of m

=

Output of ϕ on encryption of m'

$$\Pr_{k \leftarrow \mathcal{K}} [\phi(\mathsf{Enc}(k, m)) = 1] = \Pr_{k \leftarrow \mathcal{K}} [\phi(\mathsf{Enc}(k, m')) = 1]$$

Defn II ⇒ Perfect Indistinguishability

Define $S = \{c \mid \phi(c) = 1\}$.

Then, $\Pr_{k \leftarrow \mathcal{K}}[\phi(\text{Enc}(k, m)) = 1]$

$$= \sum_{c \in S} \Pr_{k \leftarrow \mathcal{K}}[\text{Enc}(k, m) = c]$$

$$= \sum_{c \in S} \Pr_{k \leftarrow \mathcal{K}}[\text{Enc}(k, m') = c]$$

$$= \Pr_{k \leftarrow \mathcal{K}}[\phi(\text{Enc}(k, m')) = 1]$$

From now on, we will work with indistinguishability-style definitions

Perfect Secrecy has its Price

THEOREM: For any perfectly secure encryption scheme,

$$|\mathscr{K}| \geq |\mathscr{M}|$$

- Exchanging large keys is difficult
- Need to keep large keys secure for a long time
- Generating truly random bits is kinda expensive!

So what can we do?

The Key Idea: Computationally Bounded Adversaries

Q: So far, we assumed that Eve is unbounded and all-powerful.

Is this reasonable?

A: No! Universe is not infinite!

So, in the real world, resources are bounded.

The Axiom of Modern Crypto

Feasible Computation

=

randomized polynomial-time* algorithms

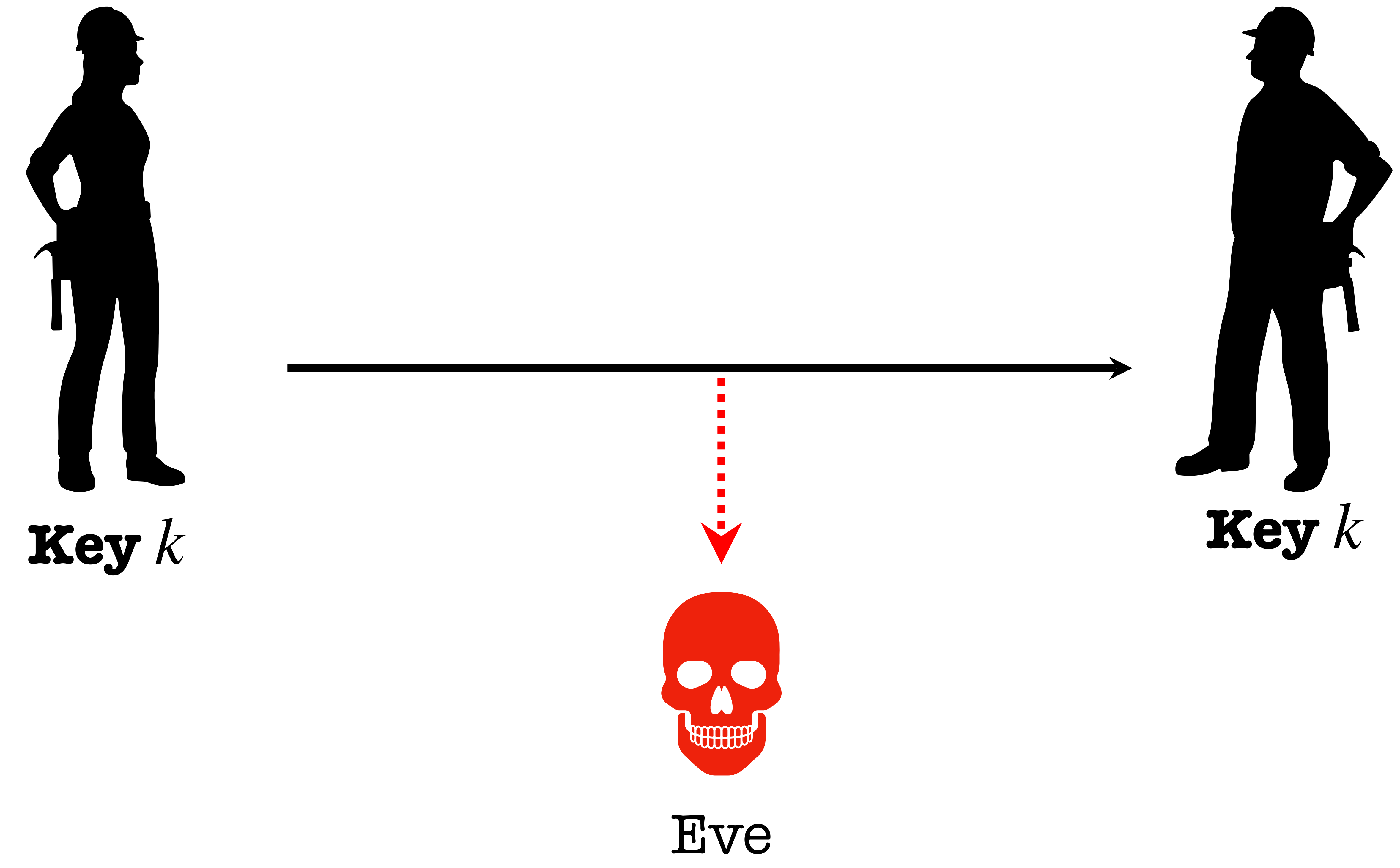
(p.p.t. = Probabilistic Polynomial-Time)

Polynomial in what?
- Size of message?
- Size of key?

Answer: Polynomial in “security parameter” λ. Can think of message and key lengths as upper bounded by λ.

*** in recent years, quantum polynomial-time**

Secure Communication



Running time of Alice and Bob?

Fixed p.p.t. (e.g., run in time $O(n^2)$)

Running time of Eve?

Arbitrary p.p.t. (e.g., run in time $O(n^2)$ or $O(n^4)$ or $O(n^{1000})$)

Computational Indistinguishability

(take 1)

For every m, m', and for every PPT "distinguishing" predicate ϕ

Output of ϕ on encryption of m

=

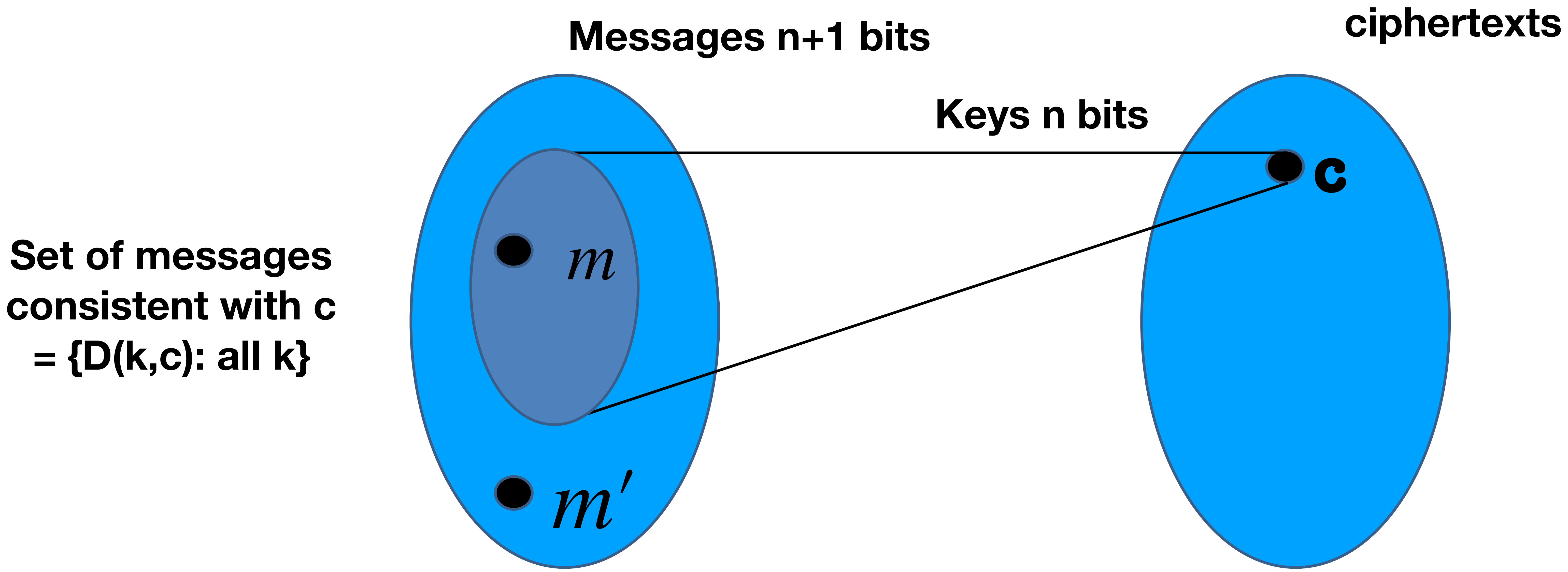
Output of ϕ on encryption of m'

$$\Pr_{k \leftarrow \mathcal{K}}[\phi(\mathsf{Enc}(k, m)) = 1] = \Pr_{k \leftarrow \mathcal{K}}[\phi(\mathsf{Enc}(k, m')) = 1]$$

Is this enough?

No!

Still subject to Shannon's impossibility!



Consider ϕ that picks a random key k and

outputs 1 if $\mathsf{Dec}(k, c) = m$

outputs 0 if $\mathsf{Dec}(k, c) = m'$

and a random bit if neither holds.

Still subject to Shannon's impossibility!

Consider ϕ that picks a random key k and
outputs 1 if $\mathsf{Dec}(k, c) = m$
outputs 0 if $\mathsf{Dec}(k, c) = m'$
and a random bit if neither holds.

When encrypting m, probability of 1 is $1/2 + 1/2^n$

When encrypting m', probability of 1 is $1/2$

$$\Pr_{k \leftarrow \mathcal{K}}[\phi(\mathsf{Enc}(k, m)) = 1] \neq \Pr_{k \leftarrow \mathcal{K}}[\phi(\mathsf{Enc}(k, m')) = 1]$$

What do we do?

Relax guarantees further!

Computational Indistinguishability

(take 1)

For every m, m', and for every **PPT** “distinguishing” predicate ϕ

Output of ϕ on encryption of m

=

Output of ϕ on encryption of m'

$$\Pr_{k \leftarrow \mathcal{K}}[\phi(\mathsf{Enc}(k, m)) = 1] - \Pr_{k \leftarrow \mathcal{K}}[\phi(\mathsf{Enc}(k, m')) = 1] = 0$$

Computational Indistinguishability

(take 2)

For every m, m', and for every **PPT** "distinguishing" predicate ϕ

Output of ϕ on encryption of m

"is close to"

Output of ϕ on encryption of m'

$$\left| \Pr_{k \leftarrow \mathscr{K}} [\phi(\mathsf{Enc}(k, m)) = 1] - \Pr_{k \leftarrow \mathscr{K}} [\phi(\mathsf{Enc}(k, m')) = 1] \right| = \varepsilon$$

How small should ε be?

- In practice:
  - Non-negligible (too large): $1/2^{30}$
  - Negligible: $1/2^{128}$
- In theory, we care about asymptotics:
  - Non-negligible: $\varepsilon > 1/n^2$
  - Negligible: $\varepsilon < 1/p(n)$ for every poly p

New Notion: Negligible Functions

Functions that grow slower than $1/p(n)$ for any polynomial p.

Definition: A function $\varepsilon : \mathbb{N} \to \mathbb{R}$ is negligible if
for every polynomial function p,
there exists an n_0 s.t.
for all $n > n_0$:

$$\varepsilon(n) < \frac{1}{p(n)}$$

<u>Key property:</u> Events that occur with negligible probability look *to poly-time algorithms* like they *never* occur.

Security Parameter: λ

Definition: A function $\varepsilon : \mathbb{N} \rightarrow \mathbb{R}$ is negligible if
for every polynomial function p,
there exists an n_0 s.t.
for all $n > n_0$

$$\varepsilon(n) < \frac{1}{p(n)}$$

- **Runtimes & success probabilities are measured as a function of λ.**
- ***Want***: Honest parties run in time (*fixed*) polynomial in λ.
- ***Allow***: Adversaries to run in time (*arbitrary*) polynomial in λ,
- ***Require***: adversaries to have success probability negligible in λ.

Computational Indistinguishability

For every m, m', for every PPT "distinguishing" predicate ϕ **(take 3)**

Output of ϕ on encryption of m

"is negligibly close to"

Output of ϕ on encryption of m'

That is, for all PPT ϕ, there exists a negligible function ε such that for all m, m'

$$\left| \Pr_{k \leftarrow \mathcal{K}} [\phi(\mathsf{Enc}(k, m)) = 1] - \Pr_{k \leftarrow \mathcal{K}} [\phi(\mathsf{Enc}(k, m')) = 1] \right| = \varepsilon(\lambda)$$

Shannon's impossibility?

Consider ϕ that picks a random key k and
outputs 1 if $\mathsf{Dec}(k, c) = m$
outputs 0 if $\mathsf{Dec}(k, c) = m'$
and a random bit if neither holds.

When encrypting m, probability of 1 is $1/2 + 1/2^n$

When encrypting m', probability of 1 is $1/2$

Negligible!

$$\Pr_{k \leftarrow \mathcal{K}}[\phi(\mathsf{Enc}(k, m)) = 1] - \Pr_{k \leftarrow \mathcal{K}}[\phi(\mathsf{Enc}(k, m')) = 1] = 1/2^n$$

Can we achieve this definition?

Yes!

Our First Crypto Tool:
Pseudorandom Generators (PRG)

Pseudorandom Generators

Informally: Deterministic Programs that stretch a "truly random" seed into a (much) longer sequence of "seemingly random" bits.

seed → PRG G → b1 b2 b3 ...

Q1: How to define "seemingly random"?

Q2: Can such a G exist?

How to Define a Strong Pseudo Random Number Generator?

Def 1 [Indistinguishability]

"No polynomial-time algorithm can distinguish between the output of a PRG on a random seed vs. a truly random string"

= "as good as" a truly random string for all practical purposes.

Def 2 [Next-bit Unpredictability]

"No polynomial-time algorithm can predict the $(i+1)^{th}$ bit of the output of a PRG given the first i bits, better than chance"

PRG Def 1: Indistinguishability

Definition [Indistinguishability]:

A **deterministic** polynomial-time computable function

$$G : \{0,1\}^n \rightarrow \{0,1\}^m \text{ is a PRG if:}$$

(a) It is **expanding**: $m > n$ and

(b) for every PPT algorithm D (called a distinguisher) if there is a negligible function ε such that:

$$\left| \Pr[D(G(U_n)) = 1] - \Pr[D(U_m) = 1] \right| = \varepsilon(\lambda)$$

Notation: U_n (resp. U_m) denotes the random distribution on n-bit (resp. m-bit) strings; m is shorthand for $m(n)$.

PRG Def 1: Indistinguishability

Definition [Indistinguishability]:

A deterministic polynomial-time computable function

$G : \{0,1\}^n \to \{0,1\}^m$ **is a PRG if:**

(a) It is expanding: $m > n$ **and**

(b) for every PPT algorithm D **(called a distinguisher) if there is a negligible function** ε **such that:**

$$\left| \Pr_{s \leftarrow \mathcal{U}_n} [D(G(s)) = 1] - \Pr_{s' \leftarrow \mathcal{U}_m} [D(s') = 1] \right| = \varepsilon(\lambda)$$

PRG Def 1: Indistinguishability

WORLD 1:

The Pseudorandom World

$$y \leftarrow G(U_n)$$

WORLD 2:

The Truly Random World

$$y \leftarrow U_m$$

PPT Distinguisher gets y but cannot tell which world she is in

Why is this a good definition

Good for all Applications:

As long as we can find truly random seeds, can replace true randomness by the output of PRG(seed) in ANY (polynomial-time) application.

If the application behaves differently, then it constitutes a (polynomial-time) statistical test between PRG(seed) and a truly random string.

PRG $\Rightarrow$ Overcoming Shannon's Conundrum

(or, How to Encrypt $n+1$ bits using an n-bit key)

Gen$(1^\lambda) \rightarrow k$:

1. Sample an n-bit string at random.

Enc$(k, m) \rightarrow c$:

1. Expand k to an $n+1$-bit string using PRG: $s = G(k)$
2. Output $c = s \oplus m$

Dec$(k, c) \rightarrow m$:

1. Expand k to an $n+1$-bit string using PRG: $s = G(k)$
2. Output $m = s \oplus c$

Correctness:

Dec(k, c) **outputs** $G(k) \oplus c = G(k) \oplus G(k) \oplus m = m$

PRG $\Rightarrow$ Overcoming Shannon's Conundrum

Security: Define distinguisher $D_m(s) = E(s \oplus m)$. Then,

$$= \Pr_{k \leftarrow \mathcal{K}}[E(G(k) \oplus m) = 1] = \Pr_{k \leftarrow \mathcal{K}}[D_m(G(k)) = 1]$$

$$\approx \Pr_{s \leftarrow \mathcal{U}_{n+1}}[D_m(s) = 1] = \Pr_{s \leftarrow \mathcal{U}_{n+1}}[E(s \oplus m) = 1]$$

$$= \Pr_{s \leftarrow \mathcal{U}_{n+1}}[E(s \oplus m') = 1] = \Pr_{s \leftarrow \mathcal{U}_{n+1}}[D_{m'}(s) = 1]$$

$$\approx \Pr_{k \leftarrow \mathcal{U}_n}[D_{m'}(G(k)) = 1] == \Pr_{k \leftarrow \mathcal{U}_n}[E(\mathsf{Enc}(k, m')) = 1]$$