CIS 5560

Cryptography
Lecture 2

Course website:

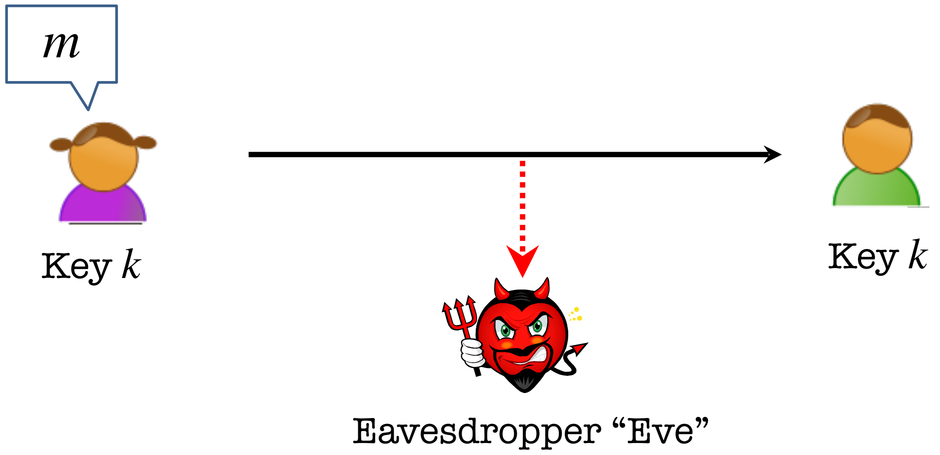
pratyushmishra.com/classes/cis-5560-s24/

Slides adapted from Dan Boneh and Vinod Vaikuntanathan

Announcements

- **HW 1 is out;** due Monday, Jan 29 at 5PM on Gradescope
    - Covers OTPs and negligible functions (this class)
    - Get started today and make use of office hours!
- Course website is up!

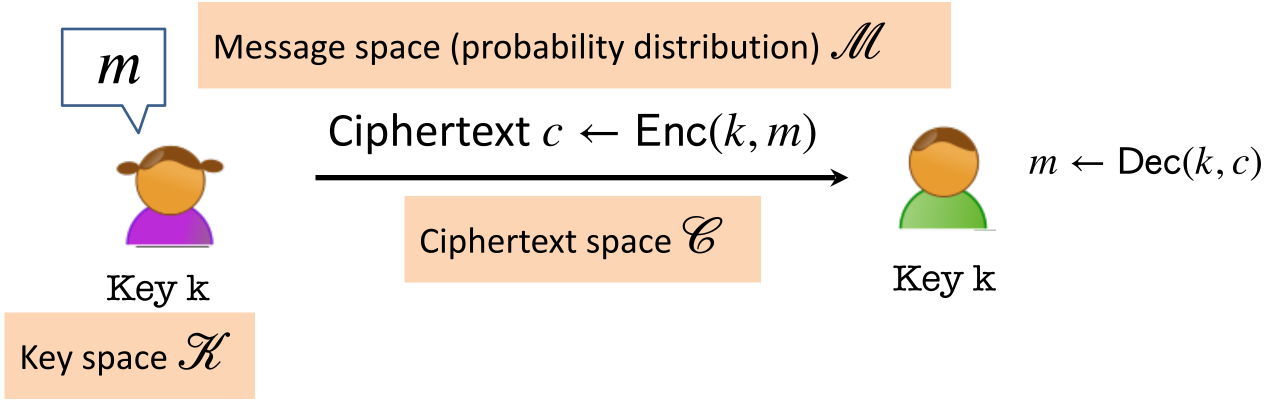
Secure Communication



Alice wants to send a message m to Bob without revealing it to Eve.

Key Notion: Secret-key Encryption

(or Symmetric-key Encryption)



Three (possibly randomized) polynomial-time algorithms:

- **Key Generation Algorithm:** $\mathsf{Gen}(1^k) \rightarrow k$
- **Encryption Algorithm:** $\mathsf{Enc}(k, m) \rightarrow c$
- **Decryption Algorithm:** $\mathsf{Dec}(k, c) \rightarrow m$

Key Property: Security

Perfect Secrecy

What Eve knows after looking at c
=
What Eve knew before looking at c

$$\forall m \in \mathcal{M}, \forall c \in \mathcal{C}, M \text{ is a RV} \sim \mathcal{M}$$

$$\underbrace{\Pr[M = m \,|\, \mathsf{Enc}(\mathcal{K}, m) = c]}_{\text{after}} = \underbrace{\Pr[M = m]}_{\text{before}}$$

Perfect Indistinguishability

Eve cannot distinguish between encryptions of m, m'

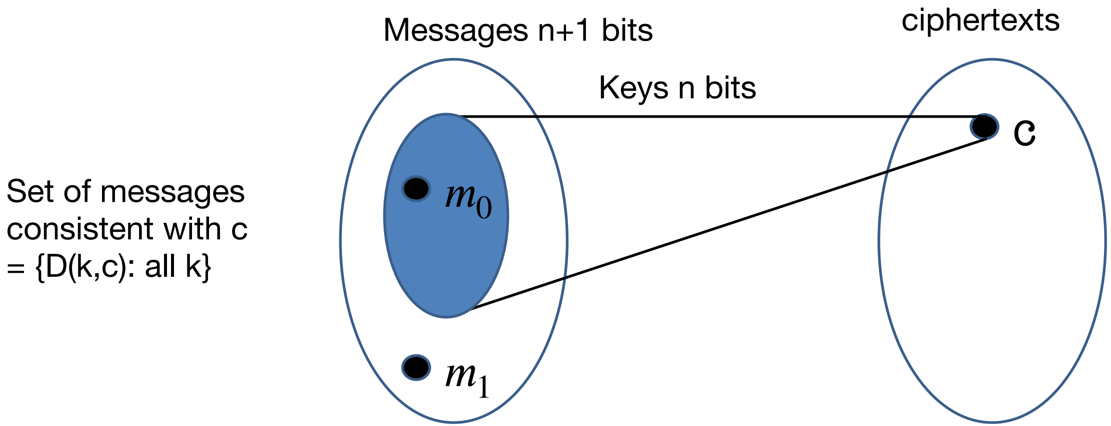
$$\forall m, m' \in \mathcal{M}, c \in \mathcal{C}$$

$$\Pr[\mathsf{Enc}(\mathcal{K}, m) = c] = \Pr[\mathsf{Enc}(\mathcal{K}, m') = c]$$

Perfectly secure encryption scheme

- **One-time Pad**: $\text{Enc}(k, m) = k \oplus m$
- **However**: Keys are as long as Messages
- **WORSE, Shannon's theorem**:
  for **any** perfectly secure scheme, $|\mathcal{K}| \geq |\mathcal{M}|$.

Shannon’s impossibility!



Each cipher text can correspond to at most 2^n messages, but message space contains 2^{n+1} possible messages!

So it is possible (and likely!) that a given cipher text can *never* decrypt to m_1!

$$\Pr[\mathsf{Enc}(\mathscr{K}, m_1) = c] = 0$$

Why is this bad?

- 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?

Let’s look at our definition in more detail…

Why Perfect Indistinguishability?

For all m_0, m_1, c: $\Pr[E(\mathscr{K}, m_0) = c] = \Pr[E(\mathscr{K}, m_1) = c]$

Why do we call it indistinguishability?

World 0:	World 1:
$k \leftarrow \mathscr{K}$	$k \leftarrow \mathscr{K}$
$c = \text{Enc}(k, m_0)$	$c = \text{Enc}(k, m_1)$

For all m_0, m_1, c : $\Pr[\text{world 0}] = \Pr[\text{world 1}]$

Perfect Indistinguishability: a Turing test

For all m_0, m_1, c: $\Pr[E(\mathcal{K}, m_0) = c] = \Pr[E(\mathcal{K}, m_1) = c]$

Why do we call it indistinguishability?

World 0:

$k \leftarrow \mathcal{K}$

$c = \mathsf{Enc}(k, m_0)$

World 1:

$k \leftarrow \mathcal{K}$

$c = \mathsf{Enc}(k, m_1)$

Eve is an **all-powerful** **distinguisher**.

She needs to decide whether c came from World 0 or World 1.

For every Eve and all m_0, m_1,

$$\Pr[\text{Eve says that we are in world 0}]$$

$$= \Pr[\text{Eve says that we are in world 1}]$$

Perfect Indistinguishability: a Turing test

For all m_0, m_1, c: $\Pr[E(\mathscr{K}, m_0) = c] = \Pr[E(\mathscr{K}, m_1) = c]$

Why do we call it indistinguishability?

World 0:

$k \leftarrow \mathscr{K}$

$c = \text{Enc}(k, m_0)$

World 1:

$k \leftarrow \mathscr{K}$

$c = \text{Enc}(k, m_1)$

Eve is an **all-powerful distinguisher**.

She needs to decide whether c came from World 0 or World 1.

For every Eve and all m_0, m_1,

$$\Pr\left[\text{Eve}(c) = 0 \,\middle|\, \begin{array}{r} k \leftarrow \mathscr{K} \\ c = \text{Enc}(k, m_0) \end{array}\right] = \Pr\left[\text{Eve}(c) = 1 \,\middle|\, \begin{array}{r} k \leftarrow \mathscr{K} \\ c = \text{Enc}(k, m_1) \end{array}\right]$$

Perfect Indistinguishability: a Turing test

World 0:

$k \leftarrow \mathcal{K}$

$c = \mathsf{Enc}(k, m_0)$

World 1:

$k \leftarrow \mathcal{K}$

$c = \mathsf{Enc}(k, m_1)$

Eve is an **all-powerful** **distinguisher**.

She needs to decide whether c came from World 0 or World 1.

For every Eve and m_0, m_1, $\Pr\left[\mathsf{Eve}(c) = b \;\middle|\; \begin{array}{r} k \leftarrow \mathcal{K} \\ b \leftarrow \{0,1\} \\ c = \mathsf{Enc}(k, m_b) \end{array} \right] = \dfrac{1}{2}$

So what can we do with this framing?

The Key Idea:

Computationally Bounded Adversaries

The Axiom of ~~Modern Crypto~~ Life

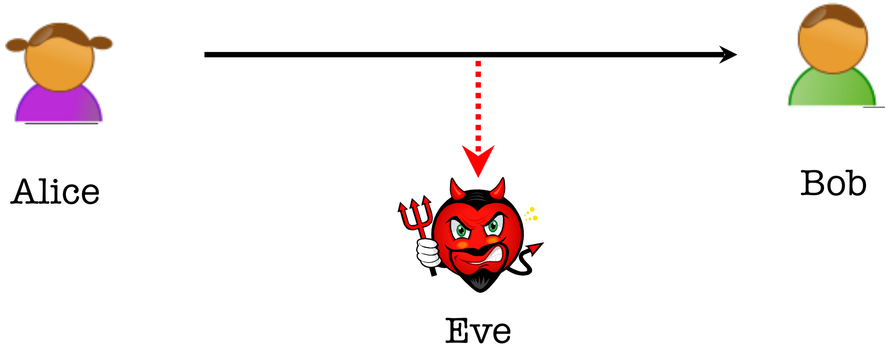
Feasible Computation = randomized polynomial-time* algorithms

(**p.p.t.** = Probabilistic polynomial-time)

(polynomial in a security parameter n)

* 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)

World 0:

$k \leftarrow \mathcal{K}$

$c = \mathsf{Enc}(k, m_0)$

World 1:

$k \leftarrow \mathcal{K}$

$c = \mathsf{Enc}(k, m_1)$

Eve is a **PPT distinguisher**.

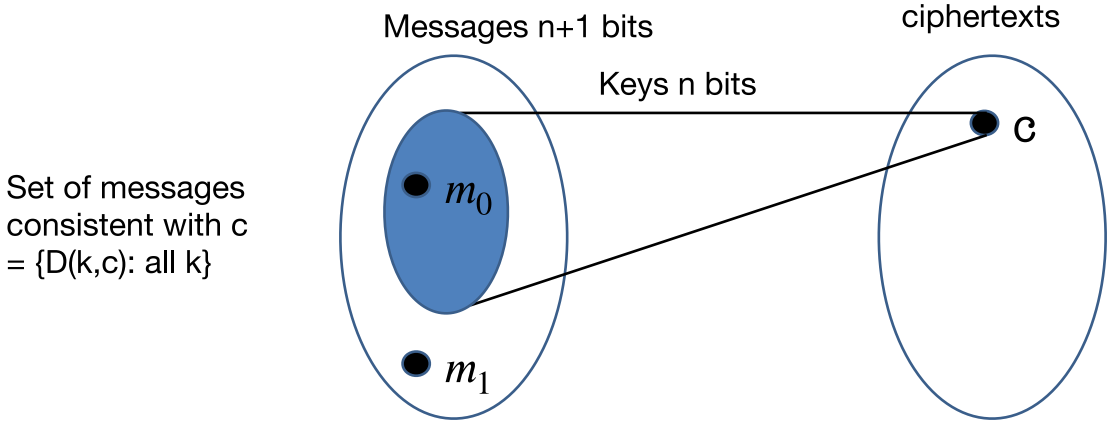
She needs to decide whether c came from World 0 or World 1.

For every **PPT** Eve and m_0, m_1, $\Pr\left[\mathsf{Eve}(c) = b \;\middle|\; \begin{array}{r} k \leftarrow \mathcal{K} \\ b \leftarrow \{0,1\} \\ c = \mathsf{Enc}(k, m_b) \end{array}\right] = \frac{1}{2}$

Is this enough?

No!

Still subject to Shannon's impossibility!



Consider Eve that picks a random key k and

outputs 0 if D(k,c) = m_0 **w.p** $\geq 1/2^n$

outputs 1 if D(k,c) = m_1 **w.p = 0**

and a random bit if neither holds.

Bottomline: Pr[EVE succeeds] $\geq$ 1/2 + $1/2^n$

What do we do?

Relax guarantees further!

Computational Indistinguishability (take 2)

World 0:

$k \leftarrow \mathcal{K}$

$c = \mathsf{Enc}(k, m_0)$

World 1:

$k \leftarrow \mathcal{K}$

$c = \mathsf{Enc}(k, m_1)$

Eve is arbitrary **PPT distinguisher**.

She needs to decide whether c came from World 0 or World 1.

For every **PPT** Eve and m_0, m_1, $\Pr\left[\mathsf{Eve}(c) = b \;\middle|\; \begin{array}{r} k \leftarrow \mathcal{K} \\ b \leftarrow \{0,1\} \\ c = \mathsf{Enc}(k, m_b) \end{array} \right] = \frac{1}{2} + \varepsilon$

Idea: Eve can only do ε better than random guessing.

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.

Why is this the right notion?

Let Eve's ε be non-negligible $1/n^2$

(i.e. distinguishes wp$1/2 + 1/n^2$)

Eve can distinguish for $1/n^2$ fraction of keys!

Formalization: Negligible Functions

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

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 $n > n_0$:

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

Question: Let $\varepsilon(n) = 1/n^{\log n}$. Is ε negligible?

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 $n > n_0$:

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

Question (PS1) Let $\varepsilon(n)$ be a negligible function and $q(n)$ a polynomial function. Is $\varepsilon(n)q(n)$ a negligible function?

Security Parameter: n *(sometimes λ)*

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 $n > n_0$:

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

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

Computational Indistinguishability (take 2)

World 0:

$k \leftarrow \mathcal{K}$

$c = \mathsf{Enc}(k, m_0)$

World 1:

$k \leftarrow \mathcal{K}$

$c = \mathsf{Enc}(k, m_1)$

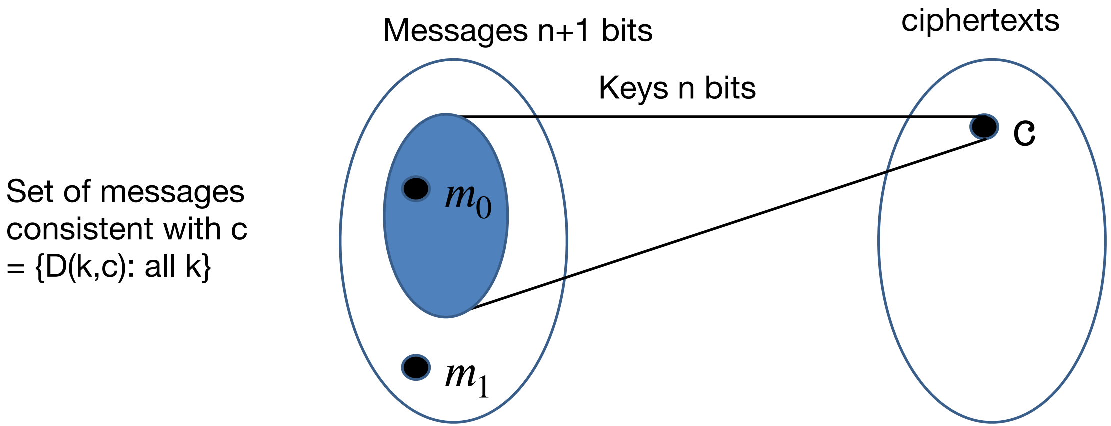
Eve is arbitrary **PPT** **distinguisher**.

She needs to decide whether c came from World 0 or World 1.

For every **PPT** Eve, there exists a negligible fn ε, st for all m_0, m_1,

$$\Pr\left[\mathsf{Eve}(c) = b \;\middle|\; \begin{array}{r} k \leftarrow \mathcal{K} \\ b \leftarrow \{0,1\} \\ c = \mathsf{Enc}(k, m_b) \end{array}\right] < \frac{1}{2} + \varepsilon(n)$$

What about Shannon's impossibility?



Consider Eve that picks a random key k and

outputs 0 if D(k,c) = m_0 **w.p $\geq 1/2^n$**

outputs 1 if D(k,c) = m_1 **w.p = 0**

and a random bit if neither holds.

Bottomline: Pr[EVE succeeds] $\geq$ 1/2 + $1/2^n$

Negligible!

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.

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 distingu… between the output of a PRG on a random seed vs. a… andom string"

= "as good as" a truly random strin… practical purposes.

Def 2 [Next-bit Unpredic…y]

"No polynomial-time a… can predict the (i+1)th bit of the output of a PRG gi… first i bits, better than chance"

ALL THREE DEFS EQUIVALENT!

Def 3 [Incon…essibility]

"No polynomial-time algorithm can compress the output of the PRG into a shorter string"

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:

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

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

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.