Today’s class

- So far:
  - One-way functions
  - Hard core predicates
  - Hard core predicates for any OWF

- Today:
  - Computational Indistinguishability
  - Pseudorandom Generators

- Scribes notes volunteers?
Randomness

- Your computer needs “randomness” for many tasks every day!
- Examples:
  - encrypting a session-key for an SSL connection (login)
  - encrypting your hard-drive for secure backup
- How does your computer generate this randomness?
  - true randomness is difficult to get
  - often, a lot of it is required (e.g. disk encryption)
Randomness

- Common sources of randomness:
  - key-strokes
  - mouse movement
  - power consumption
  - ...

- These processes can only produce so much true randomness
Can we “expand” few random bits into many random bits?

- Many heuristic approaches; good in many cases, e.g., primality testing
- But not good for cryptography, such as for data encryption
- For crypto, need bits that are “as good as truly random bits”
Pseudorandomness

- Suppose you have \( n \) uniformly random bits: \( x = x_1 \| \ldots \| x_n \)
- Find a **deterministic** (polynomial-time) algorithm \( G \) such that:
  - \( G(x) \) outputs a \( n + 1 \) bits: \( y = y_1 \| \ldots \| y_{n+1} \)
  - \( y \) looks “as good as” a truly random string \( r = r_1 \| \ldots \| r_{n+1} \)
- \( G : \{0, 1\}^n \rightarrow \{0, 1\}^{n+1} \) is called a **pseudorandom generator** (PRG)
- Think: What does “as good as truly random” mean?
As good as truly random

- Should have no obvious patterns
- Pass all statistical tests that a truly random string would pass
  - Number of 0’s and 1’s roughly the same
  - ...
- **Main Idea:** No efficient computer can tell $G(x)$ and $r$ apart!
- Distributions:

$$\left\{ x \leftarrow \{0, 1\}^n : G(x) \right\} \quad \text{and} \quad \left\{ r \leftarrow \{0, 1\}^{n+1} : r \right\}$$

are “computationally indistinguishable”
Roadmap

- **New crypto language:**
  - Computational Indistinguishability
  - Prediction Advantage

- Defining pseudorandomness using the above

- A complete test for pseudorandom distributions: Next-bit Prediction

- Pseudorandom Generators
  - Small expansion
  - Arbitrary (polynomial) expansion
**Distribution**: $X$ is a distribution over sample space $S$ if it assigns probability $p_s$ to the element $s \in S$ s.t. $\sum_s p_s = 1$

**Definition**

A sequence $\{X_n\}_{n \in \mathbb{N}}$ is called an ensemble if for each $n \in \mathbb{N}$, $X_n$ is a probability distribution over $\{0, 1\}^*$. Generally, $X_n$ will be a distribution over the sample space $\{0, 1\}^{\ell(n)}$ (where $\ell(\cdot)$ is a polynomial).
Computational Indistinguishability

- Captures what it means for two distributions \( X \) and \( Y \) to “look alike” to any efficient test
- Efficient test = efficient computation = non-uniform PPT
- No non-uniform PPT “distinguisher” algorithm \( D \) can tell them apart
- i.e. “behavior” of \( D \) on \( X \) and \( Y \) is the same
- Think: How to formalize?
Computational Indistinguishability

- Scoring system: Give $D$ a sample of $X$:
  - If $D$ say "Sample is from $X$" it gets $+1$ point
  - If $D$ say "Sample is from $Y$" it gets $-1$ point
- $D$’s output can be encoded using just one bit:
  1 = "Sample is from $X$" and 0 = "Sample is from $Y$"
- Want: Average score of $D$ on $X$ and $Y$ should be roughly same

$$\Pr \left[ x \leftarrow X; D(1^n, x) = 1 \right] \approx \Pr \left[ y \leftarrow Y; D(1^n, y) = 1 \right] \implies$$

$$\left| \Pr \left[ x \leftarrow X; D(1^n, x) = 1 \right] - \Pr \left[ y \leftarrow Y; D(1^n, y) = 1 \right] \right| \leq \mu(n).$$
Computationally Indistinguishability: Definition

**Definition (Computationally Indistinguishability)**

Two ensembles of probability distributions $X = \{X_n\}_{n \in \mathbb{N}}$ and $Y = \{Y_n\}_{n \in \mathbb{N}}$ are said to be *computationally indistinguishable* if for every non-uniform PPT $D$ there exists a negligible function $\nu(\cdot)$ s.t.:

$$\left| \Pr [x \leftarrow X_n; D(1^n, x) = 1] - \Pr [y \leftarrow Y_n; D(1^n, y) = 1] \right| \leq \nu(n).$$
Another way to model that $X$ and $Y$ “look the same”:

- Give $D$ a sample, either from $X$ or from $Y$, and ask it to guess
- If $D$ cannot guess better than $1/2$, they look same to him
- For convenience write $X^{(1)} = X$ and $X^{(0)} = Y$. Then:

**Definition (Prediction Advantage)**

$$\max_{\mathcal{A}} \left| \Pr[b \leftarrow \{0, 1\}, t \sim X^n_b : \mathcal{A}(t) = b] - \frac{1}{2} \right|$$

- Computational Indistinguishability $\iff$ Negl. Prediction Advantage
Proof of Equivalence

\[
\begin{align*}
&\left| \Pr [ b \leftarrow \{0, 1\}; z \leftarrow X^{(b)}; D(1^n, z) = b ] - \frac{1}{2} \right| \\
&= \left| \Pr_{x \leftarrow X^1}[D(x) = 1] \cdot \Pr[b = 1] + \Pr_{x \leftarrow X^0}[D(x) = 0] \cdot \Pr[b = 0] - \frac{1}{2} \right| \\
&= \frac{1}{2} \cdot \left| \Pr_{x \leftarrow X^1}[D(x) = 1] + \Pr_{x \leftarrow X^0}[D(x) = 0] - 1 \right| \\
&= \frac{1}{2} \cdot \left| \Pr_{x \leftarrow X^1}[D(x) = 1] - (1 - \Pr_{x \leftarrow X^0}[D(x) = 0]) \right| \\
&= \frac{1}{2} \cdot \left| \Pr_{x \leftarrow X^1}[D(x) = 1] - \Pr_{x \leftarrow X^0}[D(x) = 1] \right| \\
\rightarrow \quad \text{Equivalent within a factor of 2}
\end{align*}
\]
Lemma (Prediction Lemma)

Let \( \{X^0_n\} \) and \( \{X^1_n\} \) be ensembles of probability distributions. Let \( D \) be a n.u. PPT that \( \varepsilon(\cdot) \)-distinguishes \( \{X^0_n\} \) and \( \{X^1_n\} \) for infinitely many \( n \in \mathbb{N} \). Then, \( \exists \) n.u. PPT \( A \) s.t.

\[
\Pr[b \leftarrow \{0, 1\}, t \sim X^b_n : A(t) = b] - \frac{1}{2} \geq \frac{\varepsilon(n)}{2}
\]

for infinitely many \( n \in \mathbb{N} \).
Properties of Computational Indistinguishability

- **Notation**: \( \{X_n\} \approx_c \{Y_n\} \) means computational indistinguishability

- **Closure**: If we apply an efficient operation on \( X \) and \( Y \), they remain indistinguishable. That is, \( \forall \) non-uniform-PPT \( M \)

\[
\{X_n\} \approx_c \{Y_n\} \implies \{M(X_n)\} \approx_c M\{Y_n\}
\]

*Proof Idea*: If not, \( D \) can use \( M \) to tell them apart!

- **Transitivity**: If \( X, Y \) are indistinguishable with advantage at most \( \mu_1 \); \( Y, Z \) with advantage at most \( \mu_2 \); then \( X, Z \) are indistinguishable with advantage at most \( \mu_1 + \mu_2 \).

*Proof Idea*: use \( |a - c| \leq |a - b| + |b - c| \) (triangle inequality)
Lemma (Hybrid Lemma)

Let $X^1, \ldots, X^m$ be distribution ensembles for $m = \text{poly}(n)$. Suppose $D$ distinguishes $X^1$ and $X^m$ with advantage $\varepsilon$. Then, $\exists i \in [1, \ldots, m - 1]$ s.t. $D$ distinguishes $X_i, X_{i+1}$ with advantage $\geq \frac{\varepsilon}{m}$

Used in most crypto proofs!
Uniform distribution over \( \{0, 1\}^{\ell(n)} \) is denoted by \( U_{\ell(n)} \)

**Intuition:** A distribution is pseudorandom if it looks like a uniform distribution to any efficient test

**Definition (Pseudorandom Ensembles)**

An ensemble \( \{X_n\} \), where \( X_n \) is a distribution over \( \{0, 1\}^{\ell(n)} \), is said to be pseudorandom if:

\[
\{X_n\} \approx \{U_{\ell(n)}\}
\]
Pseudorandom Generators (PRG)

A computer program to convert few random bits into many random bits.

Definition (Pseudorandom Generator)
A deterministic algorithm $G$ is called a pseudorandom generator (PRG) if:

- $G$ can be computed in polynomial time
- $|G(x)| > |x|$
- $\left\{ x \leftarrow \{0, 1\}^n : G(x) \right\} \approx_c \left\{ U_{\ell(n)} \right\}$ where $\ell(n) = |G(0^n)|$

The stretch of $G$ is defined as: $|G(x)| - |x|$

First goal: construct a PRG with just 1-bit stretch.
Next-Bit Test

- Here is another interesting way to talk about pseudorandomness
- A pseudorandom string should pass all efficient tests that a (truly) random string would pass
- **Next Bit Test**: for a truly random sequence of bits, it is not possible to predict the “next bit” in the sequence with probability better than 1/2 even given all previous bits of the sequence so far
- A sequence of bits *passes the next bit test* if no efficient adversary can predict “the next bit” in the sequence with probability better than 1/2 even given all previous bits of the sequence so far
Next-bit Unpredictability

Definition (Next-bit Unpredictability)

An ensemble of distributions \( \{X_n\} \) over \( \{0, 1\}^{\ell(n)} \) is next-bit unpredictable if, for all \( 0 \leq i < \ell(n) \) and n.u. PPT \( A \), \( \exists \) negligible function \( \nu(\cdot) \) s.t.:

\[
\Pr[t = t_1 \ldots t_{\ell(n)} \sim X_n : A(t_1 \ldots t_i) = t_{i+1}] \leq \frac{1}{2} + \nu(n)
\]

Theorem (Completeness of Next-bit Test)

If \( \{X_n\} \) is next-bit unpredictable then \( \{X_n\} \) is pseudorandom.
Next-bit Unpredictability $\iff$ Pseudorandomness

\[ H_n^{(i)} := \{ x \sim X_n, u \sim U_n : x_1 \ldots x_i u_{i+1} \ldots u_{\ell(n)} \} \]

- **First Hybrid:** $H_n^0$ is the uniform distribution $U_{\ell(n)}$
- **Last Hybrid:** $H_n^{\ell(n)}$ is the distribution $X_n$
- Suppose $H_n^{(\ell(n))}$ is next-bit unpredictable but not pseudorandom
- $H_n^{(0)} \not\approx H_n^{(\ell(n))} \implies \exists i \in [\ell(n) - 1] \text{ s.t. } H_n^{(i)} \not\approx H_n^{(i+1)}$
- Now, next bit unpredictability is violated
- **Exercise:** Do the full formal proof. (This proof is very similar to one of the proofs we will do in the next class)
Next class

- Construction of a 1-bit stretch PRG
- Stretching to many bits
- Pseudorandom Functions