- **Correctness (Intuitive):** Does the receiver (Bob) recover the intended plaintext when decrypting the ciphertext?

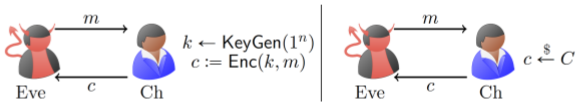
Claim

For all $k, m \in \{0,1\}^n$, it holds that $\mathsf{Dec}(k, \mathsf{Enc}(k, m)) = m$.

Proof. For all $k, m \in \{0,1\}^n$, we have:

$$\begin{aligned}\mathsf{Dec}(k, \mathsf{Enc}(k,m)) &= \mathsf{Dec}(k, (k \oplus m)) \\ &= k \oplus (k \oplus m) \\ &= 0^n \oplus m \\ &= m\end{aligned}$$

Consider the following two interactions between Eve and a challenger.



- Interaction with a *challenger* helps us model what Eve can see during encryption, and what remains hidden.
- We say that an encryption scheme is secure if for any m chosen by Eve, the above two scenarios seem identical to Eve.

Comparing Both Security Notions

Theorem

If an encryption scheme achieves one-time uniform ciphertext security, then it also achieves one-time perfect security.

We are given that for each $m \in \mathcal{M}$ (where $\mathcal{M}$ is the message space), the following distributions are identical:

1. $\mathcal{D}_1 := \{c := \mathsf{Enc}(k, m); k \leftarrow \mathsf{KeyGen}(1^n)\}$
2. $\mathcal{D}_2 := \left\{c \xleftarrow{\$} \mathcal{C}\right\}$

We want to show that for each $m_0, m_1 \in \mathcal{M}$, the following distributions are also identical:

1. $\mathcal{D}_1' := \{c := \mathsf{Enc}(k, m_0); k \leftarrow \mathsf{KeyGen}(1^n)\}$
2. $\mathcal{D}_2' := \{c := \mathsf{Enc}(k, m_1); k \leftarrow \mathsf{KeyGen}(1^n)\}$

Negligible Functions: Examples

Problem

Let $f(\cdot)$ and $g(\cdot)$ be a negligible functions. Show that $f(n) + g(n)$ is negligible.

We need to show that $\forall c$, $\exists n_0$, such that $\forall n > n_0$, $f(n) + g(n) \leqslant \frac{1}{n^c}$.

- Since $f(\cdot)$ and $g(\cdot)$ are both negligible functions, we know that $\exists n_f, n_g$ corresponding to $c+1$, such that $\forall n > n_f$, $f(n) \leqslant \frac{1}{n^{c+1}}$ and $\forall n > n_g$, $g(n) \leqslant \frac{1}{n^{c+1}}$.

For a given c, let $n_0 = \max(n_f, n_g, 2)$. $\forall n > n_0$:

$$\begin{aligned}f(n) + g(n) &\leqslant \frac{1}{n^{c+1}} + \frac{1}{n^{c+1}} \\ &\leqslant \frac{2}{n^{c+1}} \\ &\leqslant \frac{n}{n^{c+1}} \text{ (Since } n \geqslant n_0 \geqslant 2) \\ &\leqslant \frac{1}{n^c}\end{aligned}$$

Example (Double OTP)

Prove uniform ciphertext security of the following scheme:

- $\mathsf{KeyGen}(1^n)$: $k_1 \xleftarrow{\$} \{0,1\}^n$, $k_2 \xleftarrow{\$} \{0,1\}^n$ and output (k_1, k_2)
- $\mathsf{Enc}((k_1, k_2), m)$: $c_1 = k_1 \oplus m$, $c_2 = k_2 \oplus m$ and output (c_1, c_2).
- $\mathsf{Dec}((k_1, k_2), (c_1, c_2))$: Output $m = k_1 \oplus c_1$.

We need to show that for each m, the following distributions are identical:

1. $\{c_1 = k_1 \oplus m, c_2 = k_2 \oplus m; k_1 \leftarrow \mathsf{KeyGen}(1^n), k_2 \leftarrow \mathsf{KeyGen}(1^n)\}$
2. $\left\{(c_1, c_2) \xleftarrow{\$} \{0,1\}^{2n}\right\}$

We consider the following set of distributions called **hybrids**.

$\mathcal{H}_1$: $\{c_1 = k_2 \oplus m, c_2 = k_2 \oplus m; k_1 \leftarrow \mathsf{KeyGen}(1^n), k_2 \leftarrow \mathsf{KeyGen}(1^n)\}$

$\mathcal{H}_2$: $\left\{c_1 \xleftarrow{\$} \{0,1\}^n, c_2 = k_2 \oplus m; k_2 \leftarrow \mathsf{KeyGen}(1^n)\right\}$

$\mathcal{H}_3$: $\left\{c_1 \xleftarrow{\$} \{0,1\}^n, c_2 \xleftarrow{\$} \{0,1\}^n\right\}$

Encryption: One-Time Perfect Security

One-Time Perfect Security

We say that an encryption scheme is one-time perfectly secure if $\forall m_0, m_1 \in \mathcal{M}$ chosen by Eve, the following distributions are identical:

1. $\mathcal{D}_1 := \{c := \mathsf{Enc}(k, m_0); k \leftarrow \mathsf{KeyGen}(1^n)\}$
2. $\mathcal{D}_2 := \{c := \mathsf{Enc}(k, m_1); k \leftarrow \mathsf{KeyGen}(1^n)\}$

As earlier, from adversary's viewpoint, the ciphertext carries no information about the plaintext.

Negligible Functions: Examples

Problem

Let $\nu(\cdot)$ be a negligible function and $p(\cdot)$ be a polynomial s.t. $p(n) \geqslant 0$, $\forall n > 0$. Show that $\nu(n) \cdot p(n)$ is negligible.

We need to show that $\forall c$, $\exists n_0$, such that $\forall n > n_0$, $\nu(n) \cdot p(n) \leqslant \frac{1}{n^c}$.

- Since $p(\cdot)$ is a polynomial function, we know that $\exists n_p, c_p$, such that, $\forall n > n_p$, $p(n) \leqslant n^{c_p}$.
- Since $\nu(\cdot)$ is a negligible function, we know that $\exists n_\nu$ corresponding to $c + c_p$, such that $\forall n > n_\nu$, $\nu(n) \leqslant \frac{1}{n^{c+c_p}}$.

For a given c, let $n_0 = \max(n_\nu, n_p)$. $\forall n > n_0$:

$$\begin{aligned}\nu(n) \cdot p(n) &\leqslant \frac{1}{n^{c+c_p}} \cdot n^{c_p} \\ &\leqslant \frac{1}{n^{c+c_p-c_p}} \\ &\leqslant \frac{1}{n^c}\end{aligned}$$

Distributions & Ensembles

- Recall: X is a distribution over sample space $\mathcal{S}$ if it assigns probability p_s to the element $s \in \mathcal{S}$ s.t. $\sum_s p_s = 1$

Ensemble

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)

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 $\mathcal{A}$ there exists a negligible function $\nu(\cdot)$ s.t.:

$$\left|\Pr\left[x \leftarrow X_n; \mathcal{A}(1^n, x) = 1\right] - \Pr\left[y \leftarrow Y_n; \mathcal{A}(1^n, y) = 1\right]\right| \leqslant \nu(n).$$

- The quantity $\left|\Pr\left[x \leftarrow X_n; D(1^n, x) = 1\right] - \Pr\left[y \leftarrow Y_n; D(1^n, y) = 1\right]\right|$ is called the **advantage** or bias of $\mathcal{A}$ in distinguishing X and Y.
- Therefore, X and Y are computationally indistinguishable if all non-uniform PPT $\mathcal{A}$ have negligible advantage in distinguishing them.

Computational Indistinguishability

- $\mathcal{A}$'s output can be encoded using just one bit: 1 = "from X" and 0 = "from Y"
- We want $\mathcal{A}$ to output 1, with "almost similar" probability in both the above scenarios. $\Pr\left[x \leftarrow X; \mathcal{A}(1^n, x) = 1\right] \approx \Pr\left[y \leftarrow Y; \mathcal{A}(1^n, y) = 1\right] \implies$

$$\left|\Pr\left[x \leftarrow X; \mathcal{A}(1^n, x) = 1\right] - \Pr\left[y \leftarrow Y; \mathcal{A}(1^n, y) = 1\right]\right| \leqslant \nu(n).$$

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 $\mathcal{A}$ there exists a negligible function $\nu(\cdot)$ s.t.:

$$\left|\Pr\left[x \leftarrow X_n; \mathcal{A}(1^n, x) = 1\right] - \Pr\left[y \leftarrow Y_n; \mathcal{A}(1^n, y) = 1\right]\right| \leqslant \nu(n).$$

- The quantity $\left|\Pr\left[x \leftarrow X_n; D(1^n, x) = 1\right] - \Pr\left[y \leftarrow Y_n; D(1^n, y) = 1\right]\right|$ is called the **advantage** or bias of $\mathcal{A}$ in distinguishing X and Y.
- Therefore, X and Y are computationally indistinguishable if all non-uniform PPT $\mathcal{A}$ have negligible advantage in distinguishing them.

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 computationally 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, $\mathcal{A}$ can use M to tell them apart!

- **Transitivity:** If X, Y are computationally indistinguishable, and Y, Z are computationally indistinguishable; then X, Z are also computationally indistinguishable.

Generalizing Transitivity: Hybrid Lemma

Lemma (Hybrid Lemma)

Let $X^1, \ldots, X^m$ be distribution ensembles for $m = \mathsf{poly}(n)$. If for every $i \in [m-1]$, X^i and X^{i+1} are computationally indistinguishable, then X^1 and X^m are computationally indistinguishable.

This is the hybrid technique, stated more generally, in the computational setting.

Used in most crypto proofs!

Pseudorandom Generators (PRG)

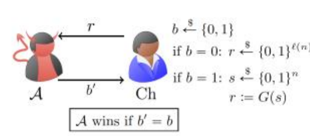
Definition (Pseudorandom Generator)

A deterministic algorithm G is called a pseudorandom generator (PRG) if:

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

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

Game Based Definition of PRG



$$\Pr[b' = 1 | b = 1] \approx \Pr[b' = 1 | b = 0]$$

$$\left|\Pr[b' = 1 | b = 1] - \Pr[b' = 1 | b = 0]\right| \leqslant \nu(n)$$

$$\left|\Pr[\mathcal{A}(1^n, r) = 1 |\ s \xleftarrow{\$} \{0,1\}^n, r := G(s)] - \Pr[\mathcal{A}(1^n, r) = 1 | r \xleftarrow{\$} \{0,1\}^{\ell(n)}]\right| \leqslant \nu(n)$$

Security of Pseudorandom OTP

Lemma

Pseudorandom OTP satisfies one-time computational security.

Proof. We need to show that $\forall m_0, m_1 \in \{0,1\}^{\ell(n)}$ chosen by an adversary, the following two distrbutions are computationally indistinguishable:

1. $\mathcal{D}_1 := \{c := m_0 \oplus G(k);\ k \leftarrow \{0,1\}^n\}$
2. $\mathcal{D}_2 := \{c := m_1 \oplus G(k);\ k \leftarrow \{0,1\}^n\}$

Consider the following hybrids:

1. $\mathcal{H}_1 := \left\{c := m_0 \oplus G(k);\ k \xleftarrow{\$} \{0,1\}^n\right\}$
2. $\mathcal{H}_2 := \left\{c := m_0 \oplus r;\ r \xleftarrow{\$} \{0,1\}^{\ell(n)}\right\}$
3. $\mathcal{H}_3 := \left\{c := m_1 \oplus r;\ r \xleftarrow{\$} \{0,1\}^{\ell(n)}\right\}$
4. $\mathcal{H}_4 := \left\{c := m_1 \oplus G(k);\ k \xleftarrow{\$} \{0,1\}^n\right\}$

Security of Pseudorandom OTP

1. $\mathcal{H}_1 := \left\{c := m_0 \oplus G(k);\ k \xleftarrow{\$} \{0,1\}^n\right\}$
2. $\mathcal{H}_2 := \left\{c := m_0 \oplus r;\ r \xleftarrow{\$} \{0,1\}^{\ell(n)}\right\}$
3. $\mathcal{H}_3 := \left\{c := m_1 \oplus r;\ r \xleftarrow{\$} \{0,1\}^{\ell(n)}\right\}$
4. $\mathcal{H}_4 := \left\{c := m_1 \oplus G(k);\ k \xleftarrow{\$} \{0,1\}^n\right\}$

- $\mathcal{H}_1 \approx_c \mathcal{H}_2$: From the security of PRG, we know that

$$\{G(k);\ k \xleftarrow{\$} \{0,1\}^n\} \approx_c \{r;\ r \xleftarrow{\$} \{0,1\}^{\ell(n)}\}$$

From closure property of computational indistinguishability, we get

$$\{m_0 \oplus G(k);\ k \xleftarrow{\$} \{0,1\}^n\} \approx_c \{m_0 \oplus r;\ r \xleftarrow{\$} \{0,1\}^{\ell(n)}\}$$

Security of Pseudorandom OTP

1. $\mathcal{H}_1 := \left\{c := m_0 \oplus G(k);\ k \xleftarrow{\$} \{0,1\}^n\right\}$
2. $\mathcal{H}_2 := \left\{c := m_0 \oplus r;\ r \xleftarrow{\$} \{0,1\}^{\ell(n)}\right\}$
3. $\mathcal{H}_3 := \left\{c := m_1 \oplus r;\ r \xleftarrow{\$} \{0,1\}^{\ell(n)}\right\}$
4. $\mathcal{H}_4 := \left\{c := m_1 \oplus G(k);\ k \xleftarrow{\$} \{0,1\}^n\right\}$

- $\mathcal{H}_2 \equiv \mathcal{H}_3$: $\mathcal{H}_2$ is an OTP encryption of m_0 and $\mathcal{H}_3$ is an OTP encryption of m_1. Therefore, $\mathcal{H}_2$ and $\mathcal{H}_3$ are identical because of the one-time perfect security of OTP.
- $\mathcal{H}_3 \approx_c \mathcal{H}_4$: Similar to $\mathcal{H}_1 \approx_c \mathcal{H}_2$.

$$\boxed{\mathcal{H}_1 \approx_c \mathcal{H}_2 \equiv \mathcal{H}_3 \approx_c \mathcal{H}_4}$$

By hybrid lemma, $\mathcal{H}_1$ is computationally indistinguishable to $\mathcal{H}_4$.

One-bit stretch PRG $\implies$ Poly-bit stretch PRG

- We will now show that once you can construct a PRG with tiny stretch (even 1 bit), you can also construct arbitrary polynomial stretch PRG.
- Intuition: Iterate the one-bit stretch PRG poly times

Construction of $G_{poly} : \{0,1\}^n \to \{0,1\}^{\ell(n)}$

Let $G : \{0,1\}^n \to \{0,1\}^{n+1}$ be a one-bit stretch PRG.

$$\begin{aligned} s &= x_0 \\ G(x_0) &= x_1 \| b_1 \\ &\vdots \\ G(x_{\ell(n)-1}) &= x_{\ell(n)} \| b_{\ell(n)} \end{aligned}$$

$G_{poly}(s) := b_1 \dots b_{\ell(n)}$

Pseudorandomnes of G_{poly}

- We want to show $\left\{G_{poly}(s);\ s \xleftarrow{\$} \{0,1\}^n\right\} \approx_c \left\{r \xleftarrow{\$} \{0,1\}^{\ell(n)}\right\}$
- Consider the following hybrid experiments:

Experiment $\mathcal{H}_1$	Experiment $\mathcal{H}_2$	Experiment $\mathcal{H}_{\ell(n)}$
$s = x_0$	$s = x_0$	$s = X_0$
$G(x_0) = x_1 \| b_1$	$s_1 \| u_1 = x_1 \| u_1$	$s_1 \| u_1 = x_1 \| u_1$
$G(x_1) = x_2 \| b_2$	$G(x_1) = x_2 \| b_2$	$s_2 \| u_2 = x_2 \| u_2$
$\dots$	$\dots$	$\dots$
$\dots$	$\dots$	$\dots$
$G(x_{\ell(n)-1}) = x_{\ell(n)} \| b_{\ell(n)}$	$G(X_{\ell(n)-1}) = x_{\ell(n)} \| b_{\ell(n)}$	$s_{\ell(n)} \| u_{\ell(n)} = x_{\ell(n)} \| u_{\ell(n)}$
Output $G(s) := b_1 b_2 \dots b_{\ell(n)}$	Output $G(s) := u_1 b_2 \dots b_{\ell(n)}$	Output $G(s) := u_1 u_2 \dots u_{\ell(n)}$

- In order to show that G_{poly} is a PRG, it suffices to show that $\mathcal{H}_1 \approx_c \mathcal{H}_{\ell(n)}$.

Pseudorandomnes of G_{poly}

Experiment $\mathcal{H}_1$	Experiment $\mathcal{H}_2$	Experiment $\mathcal{H}_{\ell(n)}$
$s = x_0$	$s = x_0$	$s = X_0$
$G(x_0) = x_1 \| b_1$	$s_1 \| u_1 = x_1 \| u_1$	$s_1 \| u_1 = x_1 \| u_1$
$G(x_1) = x_2 \| b_2$	$G(x_1) = x_2 \| b_2$	$s_2 \| u_2 = x_2 \| u_2$
$\dots$	$\dots$	$\dots$
$\dots$	$\dots$	$\dots$
$G(x_{\ell(n)-1}) = x_{\ell(n)} \| b_{\ell(n)}$	$G(X_{\ell(n)-1}) = x_{\ell(n)} \| b_{\ell(n)}$	$s_{\ell(n)} \| u_{\ell(n)} = x_{\ell(n)} \| u_{\ell(n)}$
Output $G(s) := b_1 b_2 \dots b_{\ell(n)}$	Output $G(s) := u_1 b_2 \dots b_{\ell(n)}$	Output $G(s) := u_1 u_2 \dots u_{\ell(n)}$

- $\mathcal{H}_1 \approx_c \mathcal{H}_2$: From the security of PRG, we know that

$$\{G(s); s \xleftarrow{\$} \{0,1\}^n\} \approx_c \{s_1 \| u_1 \xleftarrow{\$} \{0,1\}^{n+1}\}$$

Indistinguishability of $\mathcal{H}_1$ and $\mathcal{H}_2$ follows from the closure property of computational indistinguishability.

- Similarly, $\forall i \in [\ell(n) - 1]$, $\mathcal{H}_i \approx_c \mathcal{H}_{i+1}$.
- By Hybrid lemma, $\mathcal{H}_1 \approx_c \mathcal{H}_{\ell(n)}$.

Contrapositive Point of View

- So far, we have only considered security proofs in the "forward" direction.
- A more classical (although initially potentially confusing) way is to prove security by arriving at a contradiction.
- First, we establish the following definitions.

Definition (Non-Negligible Functions)

A function $\nu(n)$ is non-negligible if $\exists c$, such that $\forall n_0$, $\exists n > n_0$, $\nu(n) \geqslant \frac{1}{n^c}$.

Lemma (Alternate way to state Hybrid Lemma)

Let $X^1, \dots, X^m$ be distribution ensembles for $m = \mathsf{poly}(n)$. Suppose there exists a distinguisher/adversary $\mathcal{A}$ that distinguishes between X^1 and X^m with probability μ. Then $\exists i \in [m-1]$, such that $\mathcal{A}$ distinguishes between X^i and X^{i+1} with advantage at least μ/m.

Contrapositive Point of View

- So far, we have proved statements of the following form.

  "If G is a one-bit stretch PRG, then G_{poly} is a poly-bit stretch PRG."

- Let's now think about the **contrapositve** of these statements.

  *"If G_{poly} is a **not** poly-bit stretch PRG, then G is **not** a one-bit stretch PRG."*

- If G_{poly} is not a PRG, then there exists a n.u. PPT adversary $\mathcal{A}$ who can distinguish between its output on a random input and a uniformly sampled string with some **non-negligible** advantage μ.

Contrapositive Point of View

- We just proved security of G_{poly} using a sequence of hybrids.
- If we assume that an adversary $\mathcal{A}$ exists, who has non-negligible advantage in breaking the security of G_{poly}, then at least one of the steps of our previous proof must break down.
- By hybrid lemma, $\mathcal{A}$ can distinguish between at least 1 pair of consecutive hybrids (say $\mathcal{H}_i$ and $\mathcal{H}_{i+1}$) with at least $\mu/\ell(n)$ advantage.

Lemma (Alternate way to state Hybrid Lemma)

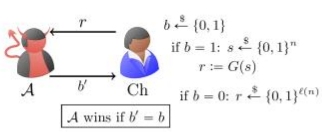
Let $X^1, \ldots, X^m$ be distribution ensembles for $m = \mathsf{poly}(n)$. Suppose there exists a distinguisher/adversary $\mathcal{A}$ that distinguishes between X^1 and X^m with probability μ. Then $\exists i \in [m-1]$, such that $\mathcal{A}$ distinguishes between X^i and X^{i+1} with advantage at least μ/m.

Contrapositive Point of View

- We just proved security of G_{poly} using a sequence of hybrids.
- If we assume that an adversary $\mathcal{A}$ exists, who has non-negligible advantage in breaking the security of G_{poly}, then at least one of the steps of our previous proof must break down.
- By hybrid lemma, $\mathcal{A}$ can distinguish between at least 1 pair of consecutive hybrids (say $\mathcal{H}_i$ and $\mathcal{H}_{i+1}$) with at least $\mu/\ell(n)$ advantage.
- In our previous proof, we relied on the security of G to argue indistinguishability of each pair of consecutive hybrids.
- We will now use $\mathcal{A}$ that has non-negligible advantage in distinguishing between $\mathcal{H}_i$ and $\mathcal{H}_{i+1}$, to construct another adversary $\mathcal{B}$ to break security of G.
- However, since G is a secure PRG, no such n.u. PPT $\mathcal{A}$ should exist. This will give us a contradiction and imply that our assumption was incorrect. G_{poly} is in fact secure.

Proof via Reduction

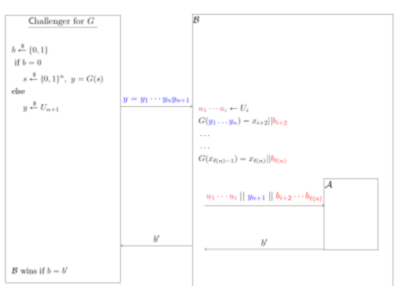
- How do we construct $\mathcal{B}$?
- We consider the game-based definition of PRG.



$$\left|\Pr[b' = 1 | b = 1] - \Pr[b' = 1 | b = 0]\right| \leqslant \nu(n)$$

Proof by Reduction

- This proof technique is also called **proof by reduction**.



Proof by Reduction

- If y is pseudorandom, i.e., sampled as $y = G(s)$, then the input to $\mathcal{A}$ is distributed identically to the output of $\mathcal{H}_i$.
- Otherwise, i.e., y is (truly) random, and therefore the input to $\mathcal{A}$ is distributed identically to the output of $\mathcal{H}_{i+1}$.
- Hence, $\mathcal{B}$ has the **same advantage** in distinguishing between the output of G and a pseudorandom string that $\mathcal{A}$ has in distinguishing between $\mathcal{H}_i$ and $\mathcal{H}_{i+1}$.
- Moreover, since $\mathcal{A}$ is n.u. PPT, so is $\mathcal{B}$. This is a contradiction!
- Hence, $G_{\mathsf{poly}(n)}$ is a PRG.

Proof by Reduction: Key Points

- These are four important things that you must work through for a valid reduction:
  1. **Input Mapping:** How to map the input that "outer adversary" $\mathcal{B}$ recieves from the challenger to an input to the "internal adversary" $\mathcal{A}$?
  2. **Input Distribution:** Does the input mapping provide the right distribution of inputs that $\mathcal{A}$ expects?
  3. **Output Mapping:** How do we map the output that $\mathcal{A}$ provides to an output for $\mathcal{B}$?
  4. **Probability:** When we assume existence of $\mathcal{A}$, we also assume that $\mathcal{A}$ wins with non-negligible advantage. What is the probability/advantage that $\mathcal{B}$ wins, given the mappings above?

One Way Functions: Definition

Definition (One Way Function)

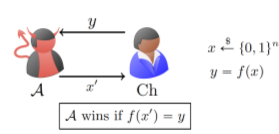
A function $f : \{0,1\}^* \to \{0,1\}^*$ is a one-way function (OWF) if it satisfies the following two conditions:

- **Easy to compute:** there is a polynomial-time algorithm $\mathcal{C}$ s.t. $\forall x \in \{0,1\}^*$,
$$\Pr\big[\mathcal{C}(x) = f(x)\big] = 1.$$
- **Hard to invert:** there exists a negligible function $\nu : \mathbb{N} \to \mathbb{R}$ s.t. for every non-uniform PPT adversary $\mathcal{A}$ and $\forall n \in \mathbb{N}$:
$$\Pr\left[x \xleftarrow{\$} \{0,1\}^n, x' \leftarrow \mathcal{A}(1^n, f(x)) : f(x') = f(x)\right] \leqslant \nu(n).$$

- The above definition is also called **strong** one-way functions.

One Way Functions: Game Based Definition

It is also instructive to think of that definition in this game-based form.



We say that $f : \{0,1\}^* \to \{0,1\}^*$ is a one-way function if there exists a negligible function $\nu : \mathbb{N} \to \mathbb{R}$ s.t. for every n.u. PPT adversary $\mathcal{A}$ and $\forall n \in \mathbb{N}$:
$$\Pr[\mathcal{A} \text{ wins}] \leqslant \nu(n).$$

Noticeable Functions

Let us start by formally defining noticeable functions. These are functions that are **at most polynomially small**.

Definition (Noticeable Function)

A function $\nu(n)$ is noticeable if $\exists c, n_0$ such that $\forall n > n_0$, $\nu(n) \geqslant \frac{1}{n^c}$.

Note that a non-negligible function is not necessarily a noticeable function. Example:

$$f(n) = \begin{cases} 1 & \text{if } n \text{ is even} \\ 2^{-n} & \text{if } n \text{ is odd} \end{cases}.$$

Weak One Way Functions

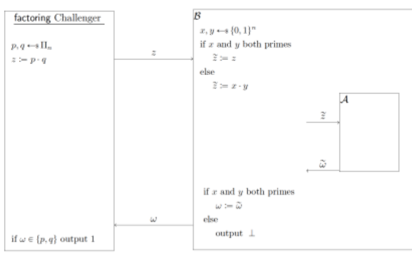
Definition (Weak One Way Function)

A function $f : \{0,1\}^* \to \{0,1\}^*$ is a *weak one-way function* if it satisfies the following two conditions:

- **Easy to compute:** there is a polynomial-time algorithm $\mathcal{C}$ s.t. $\forall x \in \{0,1\}^*$,
$$\Pr\left[\mathcal{C}(x) = f(x)\right] = 1.$$
- **Somewhat hard to invert:** there is a noticeable function $\varepsilon : \mathbb{N} \to \mathbb{R}$ s.t. for every non-uniform PPT $\mathcal{A}$ and $\forall n \in \mathbb{N}$:
$$\Pr\left[x \leftarrow \{0,1\}^n, x' \leftarrow \mathcal{A}(1^n, f(x)) : f(x') \neq f(x)\right] \geqslant \varepsilon(n).$$

Proof via Reduction

Goal: Given an adversary $\mathcal{A}$ that breaks weak one-wayness of $f_\times$ with probability *at least* $1 - \frac{1}{q(n)}$, we will construct an adversary $\mathcal{B}$ that breaks the factoring assumption with noticeable probability



Weak to Strong OWFs

Theorem

For any weak one-way function $f : \{0,1\}^n \to \{0,1\}^n$, there exists a polynomial $N(\cdot)$ s.t. the function $F : \{0,1\}^{n \cdot N(n)} \to \{0,1\}^{n \cdot N(n)}$ defined as

$$F(x_1, \ldots, x_N(n)) = (f(x_1), \ldots, f(x_N(n)))$$

is strongly one-way.

Weak to Strong OWFs: Intuition

- Recall: OWFs only guarantee average-case hardness
- GOOD inputs: hard to invert, BAD inputs: easy to invert
- A OWF is weak when the fraction of BAD inputs is **noticeable**.
- In a strong OWF, the fraction of BAD inputs is **negligible**
- To convert weak OWF to strong, use the weak OWF on **many** (say N) inputs independently
- In order to successfully invert the new OWF, adversary must invert ALL the N outputs of the weak OWF
- If N is sufficiently large and the inputs are chosen independently at random, then the probability of inverting all of them should be small

Hard Core Predicate

- A **hard core predicate** for a OWF f
  - is a function over its inputs $\{x\}$
  - its output is a single bit (called "hard core bit")
  - it can be easily computed given x
  - but "hard to compute" given only $f(x)$
- <u>Intuition</u>: f may leak many bits of x but it does not leak the hard-core bit.
- In other words, learning the hardcore bit of x, even given $f(x)$, is "as hard as" inverting f itself.
- <u>Think</u>: What does "hard to compute" mean for a single bit?
  - you can always guess the bit with probability 1/2.

Hard Core Predicate: Definition

- Hard-core bit cannot be learned or "predicted" or "computed" with probability $> \frac{1}{2} + \nu(|x|)$ even given $f(x)$ (where ν is a negligible function)

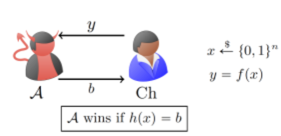
Definition (Hard Core Predicate)

A predicate $h : \{0,1\}^* \to \{0,1\}$ is a hard-core predicate for $f(\cdot)$ if h is efficiently computable given x and there exists a negligible function ν s.t. for every non-uniform PPT adversary $\mathcal{A}$ and $\forall n \in \mathbb{N}$:

$$\Pr\left[x \leftarrow \{0,1\}^n : \mathcal{A}(1^n, f(x)) = h(x)\right] \leqslant \frac{1}{2} + \nu(n).$$

Hard Core Predicate: Game Based Definition

It is also instructive to think of that definition in this game-based form.



We want that for all n.u. PPT adversary $\mathcal{A}$, the adversary wins with probability only at most negligible more than 1/2.

$$\Pr[\mathcal{A} \text{ wins}] \leqslant \frac{1}{2} + \nu(n).$$

Hard Core Predicate: Construction

- Can we construct hard-core predicates for general OWFs f?
- Define $\langle x, r\rangle$ to be the **inner product** function mod 2. I.e:,

$$\langle x, r\rangle = \left(\sum_i x_i r_i\right) \mod 2$$

Theorem (Goldreich-Levin)

Let f be a OWF. Define function

$$g(x, r) = (f(x), r)$$

where $|x| = |r|$. Then g is a OWF and

$$h(x, r) = \langle x, r\rangle$$

is a hard-core predicate for f

Warmup Proof (1)

- Assumption: Given $g(x, r) = (f(x), r)$, adversary $\mathcal{A}$ **always** (i.e., with probability 1) outputs $h(x, r)$ correctly
- Inverter $\mathcal{B}$:
  - Compute $x_i^* \leftarrow \mathcal{A}(f(x), e_i)$ for every $i \in [n]$ where:

$$e_i = (\underbrace{0, \ldots, 0}_{(i-1)\text{-times}}, 1, \ldots, 0)$$

  - Output $x^* = x_1^* \ldots x_n^*$

Warmup Proof (2)

- Assumption: Given $g(x, r) = (f(x), r)$, adversary $\mathcal{A}$ outputs $h(x, r)$ with probability $3/4 + \varepsilon(n)$ (over choices of (x, r))
- **Main Problem:** Adversary may not work on "improper" inputs (e.g., $r = e_i$ as in previous case)
- **Main Idea:** Split each query into two queries s.t. each query individually looks random
- Inverter $\mathcal{B}$:
  - Let $a := \mathcal{A}(f(x), e_i + r)$ and $b := \mathcal{A}(f(x), r)$, for $r \xleftarrow{\$} \{0,1\}^n$
  - Compute $c := a \oplus b$
  - $c = x_i$ with probability at least $\frac{1}{2} + \varepsilon$ (Union Bound)
  - Repeat and take majority to obtain x_i^* s.t. $x_i^* = x_i$ with prob. $1 - \mathsf{negl}(n)\,(n)$

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 $0 \leqslant i < \ell(n)$ and n.u. PPT $\mathcal{A}$, $\exists$ negligible function $\nu(\cdot)$ s.t.:

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

Theorem (Completeness of Next-bit Test)

If $\{X_n\}$ is next-bit unpredictable then $\{X_n\}$ is pseudorandom.

PRG with 1-bit stretch

- Let $f : \{0,1\}^* \to \{0,1\}^*$ be a **OWP**
- Let $h : \{0,1\}^* \to \{0,1\}$ be a hardcore predicate for f
- **Construction:** $G(s) = f(s) \,\|\, h(s)$

Theorem (PRG based on OWP)

G is a pseudorandom generator with 1-bit stretch.

- Think: Proof?
- Proof Idea: Use next-bit unpredictability. Since first n bits of the output are uniformly distributed (since f is a permutation), any adversary for next-bit unpredictability with non-negligible advantage $\frac{1}{p(n)}$ must be predicting the $(n+1)$th bit with advantage $\frac{1}{p(n)}$. Build an adversary for hard-core predicate to get a contradiction.

Random Functions

There are two ways to define a random function:

- **First method**: A random function F from n bits to n bits is a function selected uniformly at random from all 2^{n2^n} functions that map n bits to n bits
- **Second method:** Use a randomized algorithm to describe the function. Sometimes more convenient to use in proofs
  - randomized program M to implement a random function F
  - M keeps a table T that is initially empty.
  - on input x, M has not seen x before, choose a random string y and add the entry (x, y) to the table T
  - otherwise, if x is already in the table, M picks the entry corresponding to x from T, and outputs that
- M's output distribution identical to that of F.

Pseudorandom Functions

- Keep the description of PRF **secret** from D?
  - Security by obscurity not a good idea (Kerckoff's priniciple)
- Solution: PRF will be a keyed function. Only the key will be secret, and the PRF evaluation algorithm will be public
- **Security via a Game based definition**
  - Players: a **challenger** Ch and D. Ch is randomized and efficient
  - Game starts by Ch choosing a random bit b. If $b = 0$, Ch implements a random function, otherwise it implements a PRF
  - D send queries $x_1, x_2, \ldots$ to Ch, one-by-one
  - Ch answers by correctly replying $F(x_1), F(x_2), \ldots$
  - Finally, D outputs his guess b' (of F being random or PRF)
  - D wins if $b' = b$
- PRF Security: No D can win with probability better than $1/2$.

Pseudorandom Functions: Definition

Definition (Pseudorandom Functions)

A family $\{F_k\}_{k \in \{0,1\}^n}$ of functions, where : $F_k : \{0,1\}^n \to \{0,1\}^n$ for all k, is pseudorandom if:

- **Easy to compute:** there is an efficient algorithm M such that $\forall k, x : M(k, x) = F_k(x)$.
- **Hard to distinguish:** for every non-uniform PPT D there exists a negligible function ν such that $\forall n \in \mathbb{N}$:

$$|\Pr[D \text{ wins } \mathsf{GuessGame}] - 1/2| \leqslant \nu(n).$$

where $\mathsf{GuessGame}$ is defined below

Pseudorandom Functions: Game Based Definition

GuessGame(1^n) incorporates D and proceeds as follows:

- The games choose a PRF key k and a random bit b.
- It runs D answering every query x as follows:
- If $b = 0$: (answer using PRF)
  - output $F_k(x)$
- If $b = 1$: (answer using a random F)
  - (keep a table T for previous answers)
  - if x is in T: return $T[x]$.
  - else: choose $y \leftarrow \{0,1\}^n$, $T[x] = y$, return y.
- Game stops when D halts. D outputs a bit b'

D **wins** GuessGame **if** $b' = b$.

Remark: note that for any b only one of the two functions is ever used.

PRF from PRG

Theorem (Goldreich-Goldwasser-Micali (GGM))

If pseudorandom generators exist then pseudorandom functions exist

- **Notation:** define G_0 and G_1 as
$$G(s) = G_0(s) \| G_1(s)$$
i.e., G_0 chooses left half of G and G_1 chooses right half
- Construction for n-bit inputs $x = x_1 x_2 \ldots x_n$
$$F_k(x) = G_{x_n}(G_{x_{n-1}}(\ldots(G_{x_1}(k))..)$$

Proof Strategy (contd.)

Two layers of hybrids:

- First, define hybrids over the n levels in the tree. For every i, H_i is such that the nodes up to level i are random, but the nodes below are pseudorandom.
- Now, hybrid over the nodes in level $i+1$ that are "affected" by adversary's queries, replacing each node one by one with random
- Use PRG security to argue indistinguishability

Discrete Logarithm Problem: Definition

Definition (Discrete Logarithm Problem)

Let $(G, \cdot)$ be a cyclic group of order p (where p is a safe prime) with generator g, then for every non-uniform PPT adversary $\mathcal{A}$, there exists a negligible function ε such that

$$\Pr[a \xleftarrow{\$} \{0,\ldots,p-1\}, a' \leftarrow \mathcal{A}(G,p,g,g^a) : a = a'] \leqslant \varepsilon$$

Computational Diffie-Hellman Assumption: Definition

Definition (Computational Diffie-Hellman Assumption)

Let $(G, \cdot)$ be a cyclic group of order p (where p is a safe prime) with generator g, then for every non-uniform PPT adversary $\mathcal{A}$, there exists a negligible function ε such that

$$\Pr[a,b \xleftarrow{\$} \{0,\ldots,p-1\}, y \leftarrow \mathcal{A}(G,p,g,g^a,g^b) : g^{ab} = y] \leqslant \varepsilon$$

Decisional Diffie-Hellman Assumption

- Let $(G, \cdot)$ be a cyclic group of order p with generator g, where p is an n-bit safe prime number.
- Pick $b \xleftarrow{\$} \{0,1\}$
- If $b = 0$, send (g, g^a, g^b, g^{ab}), where $a, b \xleftarrow{\$} \{0,\ldots,p-1\}$
- If $b = 1$, send (g, g^a, g^b, g^r), where $a, b, r \xleftarrow{\$} \{0,\ldots,p-1\}$
- Adversary has to guess b
- Effectively: $(g, g^a, g^b, g^{ab}) \approx (g, g^a, g^b, g^r)$, for $a, b, r \xleftarrow{\$} \{0,\ldots,p-1\}$ and any g

Decisional Diffie-Hellman Assumption: Definition

Definition (Decisional Diffie-Hellman Assumption)

Let $(G, \cdot)$ be a cyclic group of order p (where p is a safe prime) with generator g, then the following two distributions are computationally indistinguishable:

- $\{a, b \xleftarrow{\$} \{0,\ldots,p-1\} : (G,p,g,g^a,g^b,g^{ab})\}$
- $\{a, b, r \xleftarrow{\$} \{0,\ldots,p-1\} : (G,p,g,g^a,g^b,g^r)\}$

Key Agreement: Construction (Diffie-Hellman)

- Let $(G, \cdot)$ be a cyclic group of order p (where p is a safe prime) with generator g.
- Alice picks $a \xleftarrow{\$} \{0,\ldots,p-1\}$ and sends g^a to Bob
- Bob picks $b \xleftarrow{\$} \{0,\ldots,p-1\}$ and sends g^b to Alice
- Alice outputs $(g^b)^a$ and Bob outputs $(g^a)^b$
- Adversary sees: (g^a, g^b)
- Correctness?
- Security? Use DDH to say that g^{ab} is hidden from adversary's view
- Think: Is this scheme still secure if the adversary is allowed to modify the messages?

Multi-message Secure Encryption

Definition (Multi-message Secure Encryption)

A secret-key encryption scheme $(\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec})$ is multi-message secure if for all n.u. PPT adversaries $\mathcal{A}$, for all polynomials $q(\cdot)$, there exists a negligible function $\mu(\cdot)$ s.t.:

$$\Pr\left[\begin{array}{r} s \xleftarrow{\$} \mathsf{Gen}(1^n), \\ \{(m_0^i, m_1^i)\}_{i=1}^{q(n)} \leftarrow \mathcal{A}(1^n), \\ b \xleftarrow{\$} \{0,1\} \end{array} : \mathcal{A}\left(\{\mathsf{Enc}(m_b^i)\}_{i=1}^{q(n)}\right) = b\right] \leqslant \frac{1}{2} + \mu(n)$$

- Think: Security against *adaptive* adversaries (who may choose message pairs in an adaptive manner based on previously seen ciphertexts)?

Encryption using PRFs

Let $\{f_s : \{0,1\}^n \to \{0,1\}^n\}$ be a family of PRFs

- $\mathsf{Gen}(1^n)$: $s \xleftarrow{\$} \{0,1\}^n$
- $\mathsf{Enc}(s, m)$: Pick $r \xleftarrow{\$} \{0,1\}^n$. Output $(r, m \oplus f_s(r))$
- $\mathsf{Dec}(s, (r, c))$: Output $c \oplus f_s(r)$

Theorem (Encryption from PRF)

$(\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec})$ *is a multi-message secure encryption scheme*

- Think: Proof?

Proof of Security

Proof via hybrids:

- H_1: Real experiment with $m_0^1, \ldots, m_0^{q(n)}$ (i.e., $b = 0$)
- H_2: Replace f_s with random function $f \xleftarrow{\$} \mathcal{F}_n$
- H_3: Switch to one-time pad encryption
- H_4: Switch to encryption of $m_1^1, \ldots, m_1^{q(n)}$
- H_5: Use random function $f \xleftarrow{\$} \mathcal{F}_n$ to encrypt
- H_6: Encrypt using f_s. Same as real experiment with $m_0^1, \ldots, m_0^{q(n)}$ (i.e., $b = 1$)

Think: Non-adaptive vs adaptive queries

Semantic Security

Definition (Semantic Security)

A secret-key encryption scheme $(\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec})$ is semantically secure if there exists a PPT simulator algorithm $\mathcal{S}$ s.t. the following two experiments generate computationally indistinguishable outputs:

$$\left\{\begin{array}{r} (m, z) \leftarrow M(1^n), \\ s \leftarrow \mathsf{Gen}(1^n), \\ \text{Output } (\mathsf{Enc}(s, m), z) \end{array}\right\} \approx \left\{\begin{array}{l} (m, z) \leftarrow M(1^n), \\ \text{Output } S(1^n, z) \end{array}\right\}$$

where M is a machine that randomly samples a message from the message space and arbitrary auxiliary information.

- Indistinguishability security $\Leftrightarrow$ Semantic security

Definition

- **Syntax:**
  - $\mathsf{Gen}(1^n) \to (pk, sk)$
  - $\mathsf{Enc}(pk, m) \to c$
  - $\mathsf{Dec}(sk, c) \to m'$ or $\perp$

  All algorithms are polynomial time
- **Correctness:** For every m, $\mathsf{Dec}(sk, \mathsf{Enc}(pk, m)) = m$, where $(pk, sk) \leftarrow \mathsf{Gen}(1^n)$
- **Security:** ?

Security

Definition ((Weak) Indistinguishability Security)

A public-key encryption scheme $(\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec})$ is weakly indistinguishably secure under chosen plaintext attack (weak IND-CPA) if for all n.u. PPT adversaries $\mathcal{A}$, there exists a negligible function $\mu(\cdot)$ s.t.:

$$\Pr\left[\begin{array}{r} (pk, sk) \xleftarrow{\$} \mathsf{Gen}(1^n), \\ (m_0, m_1) \leftarrow \mathcal{A}(1^n), \\ b \xleftarrow{\$} \{0,1\} \end{array} : \mathcal{A}(pk, \mathsf{Enc}(pk, m_b)) = b\right] \leqslant \frac{1}{2} + \mu(n)$$

- Think: Semantic security style definition?

Security (contd.)

A stronger definition:

Definition (Indistinguishability Security)

A public-key encryption scheme $(\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec})$ is indistinguishably secure under chosen plaintext attack (IND-CPA) if for all n.u. PPT adversaries $\mathcal{A}$, there exists a negligible function $\mu(\cdot)$ s.t.:

$$\Pr\left[\begin{array}{r} (pk, sk) \xleftarrow{\$} \mathsf{Gen}(1^n), \\ (m_0, m_1) \leftarrow \mathcal{A}(1^n, pk), \\ b \xleftarrow{\$} \{0,1\} \end{array} : \mathcal{A}(pk, \mathsf{Enc}(m_b)) = b\right] \leqslant \frac{1}{2} + \mu(n)$$

- Think: IND-CPA is stronger than weak IND-CPA