Lecture 12: Pseudorandom Functions

1. PRGs & Encryption
2. PRF Definition
3. Adaptive Adversary
4. PRF Construction & Proof

1. PRGs & Encryption

Random $s_0 \xrightarrow{\text{long, random-looking}} R = r \cdot s_i$

Use now, e.g. as OTP

Seed to get pseudorandom string later

\[
\begin{align*}
\text{Alice} & \quad s_0 = s \\
(1, m_1 \oplus r_1) & \\
(2, m_2 \oplus r_2) & \\
\vdots & \\
(i, m_i \oplus r_i) & \\
\end{align*}
\]

We want: On input $s, i$, compute $r_i$ in time that is $\text{poly}(|s|, \log i)$.

2. PRF Definition

Def. (in progress). $F : \{0, 1 \}^k \times \{0, 1 \}^k \rightarrow \{0, 1 \}^k$

is a pseudorandom function if $\forall$ ppt oracle TMs $A$,

$|\mathbb{E}_{F, A}(k) - \mathbb{E}_{R, A}(k)| \leq \nu(k)$ for negligible $\nu$. 

Recall: $\gamma_{F,A}(k) = \Pr[ s \leftarrow \{0,1\}^k ; b' \leftarrow A(1^k) : b' = 0 ]$

$\text{TM } F:$

```
   s
state controls
   F(s, i)
   i
```

Query tape

```
i
F(s, i)
   ... work tape
```

$\text{TM } A:$

```
state controls
   ...
```

work tape

Also: $\gamma_{R,A}(k) = \Pr[ R \leftarrow \{ \text{all functions} \} ; b' \leftarrow A(1^k) : b' = 0 ]$

$\text{TM } R:$

```
a_1\ a_2\ ...
state controls
   a_n
i_1\ a_1\ ...
i_n\ a_n
```

work tape

random tape

Query tape

```
...
i_n\ a_n
```

$\text{TM } A:$

```
state controls
   ...
```

work tape
A random function \( R \) can be described by a table:

\[
\begin{array}{c|c}
0 & \text{[random } k \text{ bits for } R(0)] \\
1 & \text{[random } k \text{ bits for } R(1)] \\
\vdots & \vdots \\
2^k - 1 & \text{[random } k \text{ bits for } R(2^k - 1)] \\
\end{array}
\]

\( 2^k \cdot k \) bits to describe \( R \)

We can think of breaking a PRF as distinguishing between two experiments:

- \( F \) experiment
  - \( x \)
  - \( F_x \)
  - \( s \)
  - \( F_s(x) \)
  - \( U \)

- \( R \) experiment
  - \( x \)
  - \( R \)
  - \( r \), completely random but well-defined for \( x \)
  - \( U \)

From \( U \)'s perspective, its oracle looks like a gnome in a box. \( U \) wants to determine whether the gnome is computing \( F_s \) or \( R \).

3. Adaptive Adversary

\( U \) can query its oracle/gnome polynomially many times, and can do so adaptively.

Suppose we have \( F \) s.t. \( F(s, x) = s \circ F'(s_2, x) \), where \( s = s_1(s) s_2 \).

By asking \( 2^k \) questions, we can check for \( s_1 \) in front. So \( F \) cannot be a PRF.

Note that \( G(s) = s \circ G'(s_2) \) is a PRG if \( G' \) is a PRG.

More examples:

- \( F(s, x) = F'(s, x) \circ F'(s_2, x_2) \)
- \( F(s, x) = F'(s, 0^k) \)
- \( F(s, x) = \begin{cases} 
F'_s(x) & \text{if } x \neq F'_s(0^k) \\
0^k & \text{if } x = F'_s(0^k) 
\end{cases} \)
4. PRF Construction & Proof

GGM PRF from PRG $G_i : \{0, 1^k \rightarrow \{0, 1\}^{2^k}$

$G_i(s) = G_{i-1}(s) \oplus G_1(s)$

$k$ bits \hspace{1cm} $k$ bits \hspace{1cm} $k$ bits

---

$\cdots$

$S_0 = G_0(s_0)$

$S_1 = G_1(s_0)$

$S_{u+1} = G_{u+1}(S_u, \ldots, S_0)$

for $u = u_1, \ldots, u_k$

---

Observation: To compute $S_x$ for k-bit $x$, we only need to compute $S_{x_i}$ where $x_i$ is the i-bit prefix of $x$.

$S_{x_i} = G_i^{i^{\text{th}} \text{ bit of } x}(S_{x_{i-1}})$

---

The PRF $F_k(x) = S_x$ is computed iteratively according to the observation.

---

Alice

$F_{\text{sen}}$

$(x, m \oplus F_{\text{sen}}(x))$

$F_{\text{auth}}$

---

Bob

$F_{\text{sen}}$

$F_{\text{auth}}$

---

Eve

$(x_i \text{ altered } c)$

---