1 Introduction

Today we will construct PRFs (Pseudorandom Functions) from PRGs (Pseudorandomness Generators). We will go on to define Pseudorandom and Strongly Pseudorandom Permutations. We end by introducing an example of a strongly pseudorandom permutation through Feistel Permutations.

2 Constructing PRFs from PRGs

We begin by recalling the following definition and property from the previous class. The latter will be instrumental in our proof to follow.

Definition 1 PRF $F$ is pseudorandom if $\forall$ ppt $D$:
$$|Pr[D^{F_s}(1^k) = 1] - Pr[D^{F_k}(1^k) = 1]|$$

is negligible in $k$.

Proposition 2 For a PRG $G$ acting on inputs of size $k$ with expansion factor $l$, we have:
$$G(U_k^1)...G(U_k^p) \equiv U_p^l(k)$$

We now begin the construction. Let $G$ be a PRG with expansion factor $l(k) = 2k$. Denote $G_0, G_1$ as the first, last $k$ bits of $G$’s output, respectively (i.e. $G(s) = G_0(s)||G_1(s)$ where $|s| = |G_0(s)| = |G_1(s)| = k$).

Now, define we will define our pseudorandom function $F$ as follows. Let $f_s$ be defined by:
$$\forall s \in \{0, 1\}^k f_s(x_1...x_k) := G_{x_k}(G_{x_{k-1}}(...G_{x_1}(s)...))$$

And so:
$$F = \{F_k\}_{k \in \mathbb{N}} \text{ with } F_k = \{f_s | s \leftarrow \{0, 1\}^k\}$$

Theorem 3 $F$ is a PRF.

Proof: We begin by pictorially representing the $f_s$. We can think of $f_s$ as a labelled tree of $k$ levels with the root being labelled as $s$. For a node $s_m$ at level $i$, it has two children in level $i+1$; the left child is labelled with the result of applying $G_0$ on $s_m$ and the right child is labelled with the result of applying $G_1$ on $s_m$. The leaves of this tree are then the outputs of $f_s$. We see this in the following example for $k = 4$:
For example, the leaf $s_{011}$ is associated with output of the call:

$$f_s(011) = G_1(G_1(G_0(s))) = G_1(s_{01}) = s_{011}$$

Now, we proceed with the proof. Assume (towards contradiction) that there exists PPT $D$ such that $D$ distinguishes $F$ from $U$ with probability $\delta(k)$ non-negligible. We proceed by a hybrid argument in order to create a distinguisher for $G$.

**Remark 4** It is tempting to use the leaves of the tree to construct the hybrid. Namely, we would fill a number of slots with randomness, and fill the remaining slots with the leaves from tree above. However, we quickly see that there are $2^k$ leaves, so the hybrids would need to be at least $2^k$ in size which is far too large. Instead, we consider the following argument.

In light of the above, we will consider hybrids of levels in the tree instead of hybrids of leaves. More specifically, $H_k^{(i)}$ is defined by selecting the first $i$ levels in the tree uniformly at random from $U_k$ (i.e. for any $r < i$, we have that the value associated with the node $s_r$ is produced from (re)sampling from $U_k$ for a total of $|r|$ times). For all levels greater than $i$, we apply $G_0$ or $G_1$ to the result from the previous level as described in the tree above. Pictorially, can observe the example $H_3^{(1)}$ as represented as below ($s_r \leftarrow \$" means "sample from $U_k$):

We can now make the following observation.

**Observation 5** $H_k^{(0)} = F_k, H_k^{(k)} = U_K$
Using the standard procedure for a hybrid argument, we note that there exists \( i \) such that \( D \) distinguishes \( H_k^{(i)} \) and \( H_k^{(i+1)} \) with probability greater than \( \frac{\delta(k)}{k} \) for some \( \delta(k) \in \text{poly}(k) \).

We now would like to construct a \( D' \) that attacks \( G \) (actually, in this case, \( G \) applied on \( q \) inputs of length \( k \)) and succeeds with non-negligible probability. We first note that \( D \) asks at most \( q(k) \) queries. On input \( z^{(1)}, ..., z^{(q)} \), \( D' \) will first pick \( j \leftarrow \{1, 2, ..., q\} \). \( D' \) will then run \( D \) and simulate its oracle by using \( z^{(j)} \) to answer the queries which \( D \) dispatches. We outline how \( z^{(j)} \) is used to answer the queries from \( D \) below.

Given some query \( q_i \), \( D' \) will answer by using the following procedure based on traversing the tree mentioned at the beginning of this proof. The first \( j \) levels of the trees will be traversed according to the first \( j \) bits of \( q_i \). More specifically, for \( t < j \), \( D' \) takes the left branch from the \( t \)-th level if the \( t \)-th bit in \( q_i \) is 0 and take the right branch otherwise. From the \( j \)-th level to the \( (j+1) \)-th level, we label the two children of the \( j \)-th node (i.e. the children are at the \( (j+1) \)-th level) with \( z^{(j)} \) (or \( z^{(j+1)} \) if we had already performed \( l-1 \) labellings at this level) if the pair of children at the \( (j+1) \)-th level are unlabelled. Otherwise, we simply use the labelling already provided. We then use this string when computing the values going downwards in the tree and where the paths are selected by the remaining \( k-j \) bits in \( q_i \). For example, if \( j=1 \) and \( q_1 = 0110 \), we would traverse down the left branch and then the right branch in the tree (corresponding to 01 in \( q_1 \)). If there is no labelling on the children of this node, we would then label them as \( z^{(j)} \). From this point, we can return the result of the query as \( G_0(G_1(z^{(j)})) \) because the last 2 bits of \( q_1 \) are 10. Then for the query \( q_2 = 0110 \), we hit the same labelling at level \( j+1 \), so we return \( G_0(G_0(z^{(j)})) \) as the answer to \( q_2 \).

Now, from here we see that:

\[
\begin{align*}
z^{(j)} &\sim \mathcal{U}_{q(k)} \Rightarrow D' \text{ sees } H_k^{(i+1)} \\
z^{(j)} &\sim G(\mathcal{U}_k) \Rightarrow D' \text{ sees } H_k^{(i)}
\end{align*}
\]

And by the hybrid argument and Property 2, above, the theorem follows. \( \square \)

3 Pseudorandom Permutations

Now we turn our attention to pseudorandom permutations.

Definition 6 A permutation ensemble is an ensemble \( \mathcal{P} = \{\mathcal{P}_k\}_k \) where \( \mathcal{P}_k \) is a distribution over permutations \( f : \{0,1\}^k \to \{0,1\}^k \).

Definition 7 The uniform permutation ensemble is the ensemble \( \Pi = \{\Pi_k\}_k \) where \( \Pi_k \) is uniform over permutations \( f : \{0,1\}^k \to \{0,1\}^k \).

Let \( A \equiv B \) mean "\( A \) is oracle indistinguishable with respect to \( B \)."

Definition 8 \( \mathcal{P} \) is pseudorandom if \( \mathcal{P} \equiv \Pi \).

Lemma 9 \( \mathcal{P} \equiv \Pi \Rightarrow \mathcal{P} \equiv \mathcal{U} \)
Proof: (Sketch) We note that \( \Pi \overset{O}{=} \mathcal{U} \) because:

\[
\Pr[D^{\mathcal{U}}(1^k) \text{ finds a collision}] \leq \frac{\text{time}(D)^2}{2^k}
\]

From here, we can apply indistinguishability between \( \mathcal{P} \) and \( \Pi \), and the lemma follows.

Now we will examine the properties required for permutations to be useful in a cryptographic sense.

**Definition 10** A permutation ensemble \( \mathcal{P} \) is efficiently computable and invertable if:

1. \( \exists \text{ppt } S \text{ such that } S(1^k) = P_k \)
2. \( \exists \text{ppt } E \text{ such that } \forall f \in S(1^k), E(1^k, f, x) = f(x) \)
3. \( \exists \text{dpt } I \text{ such that } \forall f \in S(1^k), I(1^k, f, y) = f^{-1}(y) \)

Now we get to a notion of pseudorandomness for permutations that is stronger than the ones defined earlier.

**Definition 11** \( \mathcal{P} \) is strongly pseudorandom if:

\( (\mathcal{P}, \mathcal{P}^{-1}) \overset{O}{=} (\Pi, \Pi^{-1}) \)

and \( \forall \text{ppt } D, \)

\[
|\Pr[D^{P_k,P_k^{-1}}(1^k) = 1] - \Pr[D^{\Pi_k,\Pi_k^{-1}}(1^k) = 1]| \text{ is negligible in } k
\]

4 Feistel Permutation

We will soon construct a strongly pseudorandom permutation. We begin with the construction of Feistel Permutations, as follows.

Given a function \( f : \{0,1\}^k \rightarrow \{0,1\}^k \), let \( g_f : \{0,1\}^{2k} \rightarrow \{0,1\}^{2k} \) be defined by:

\[
g_f(x||y) = y||f(y) \oplus x
\]

We note that \( g_f \) is indeed a permutation. For we have:

\[
g_f^{-1}(z||w) = f(z) \oplus w||z
\]

and we see that:

\[
g_f^{-1}(g_f(x||y)) = g_f^{-1}(y||f(y) \oplus x) = f(y) \oplus (f(y) \oplus x)||y = x||y
\]

It should also be clear that both \( g_f \) and \( g_f^{-1} \) are efficiently computable given \( f \). \( g_f \) is called a **Feistel Permutation**. Compositions of \( g_f \)'s for selections of \( f \)'s are called **Feistel Networks**.

Now, to get a strong pseudorandom permutation we compose \( g_f \)'s (for different \( f \)'s). However, we make the following observations.
Observation 12 \textit{g}_f \textit{ is not} a pseudorandom permutation. We see this by the fact that:

\[ g_f(x||0^k) = 0^k || f(0^k) \oplus x \]

for any \( x \in \{0,1\}^k \).

Observation 13 For any \( f_1, f_2 \), we have that \( g_{f_2} \circ g_{f_1} \) is also \textbf{not} a pseudorandom permutation. We see that:

\[(g_{f_2} \circ g_{f_1})(0^k0^k) = f_1(0^k)||... \]

and that:

\[(g_{f_2} \circ g_{f_1})(1^k0^k) = f_1(0^k) \oplus 1^k||... \]

But now we see that \( f_1(0^k) \oplus 1^k = f_1(0^k) \), and so the fact that \( g_{f_2} \circ g_{f_1} \) is not pseudorandom follows.

Remark 14 For any \( f_1, f_2, f_3 \), we have that \( g_{f_3} \circ g_{f_2} \circ g_{f_1} \) is pseudorandom but not strongly pseudorandom. Showing this is much more difficult than for the above two cases, so it is omitted.

5 Luby-Rackoff Construction (1988)

Consider some function ensemble \( \mathcal{F} = \{ \mathcal{F}_k \}_k \). Consider \( \mathcal{G} = \{ \mathcal{G}_k \}_k \) where:

\[ \mathcal{G}_k = \{ g_{f_4} \circ g_{f_3} \circ g_{f_2} \circ g_{f_1} | f_1, f_2, f_3, f_4 \in \mathcal{F}_k \} \]

Theorem 15 \( \mathcal{F} \overset{\circ}{=} \mathcal{U} \Rightarrow (\mathcal{G}, \mathcal{G}^{-1}) \overset{\circ}{=} (\Pi, \Pi^{-1}) \)

Proof: Define \( \mathcal{R} = \{ R_k \}_k \) where \( R_k = \{ g_{u_4} \circ g_{u_3} \circ g_{u_2} \circ g_{u_1} | u_1, u_2, u_3, u_4 \in U_k \} \). Our proof will consist of two parts:

(S1) \((\mathcal{G}, \mathcal{G}^{-1}) \overset{\circ}{=} (\mathcal{R}, \mathcal{R}^{-1}) \)

(S2) \((\mathcal{R}, \mathcal{R}^{-1}) \overset{\circ}{=} (\Pi, \Pi^{-1}) \)

We will prove S1 now and prove S2 tomorrow.

Proof of S1 We will use a hybrid argument, so assume there exists some machine \( D \) that can distinguish \( \mathcal{G} \) from \( \mathcal{R} \) with non-negligible advantage. We proceed by defining 5 hybrids \( H_k^{(0)}, H_k^{(1)}, ..., H_k^{(4)} \). We explicitly write out the first two hybrids:

\[ H_k^{(0)} = \{ g_{f_4} \circ g_{f_3} \circ g_{f_2} \circ g_{f_1} | f_1, f_2, f_3, f_4 \in \mathcal{F}_k \} = \mathcal{G}_k \]
\[ H_k^{(1)} = \{ g_{f_4} \circ g_{f_3} \circ g_{f_2} \circ g_{u_1} | f_2, f_3, f_4 \in \mathcal{F}_k, u_1 \in U_k \} \]

The remaining hybrids are defined similarly, and we observe that \( H_k^{(4)} = \mathcal{R}_k \).

Now, by an averaging argument, there exists \( i \) such that \( (H_k^{(i)}, (H_k^{(i)})^{-1}) \) and \( (H_k^{(i+1)}, (H_k^{(i+1)})^{-1}) \)
we will now be able to define a machine $\overline{D}^A$ that is able to distinguish neighboring hybrids (where $A$ is an oracle which is either $F_k$ or $U_k$).

For simplicity, let us define $\overline{D}^A$ for the case where $i = 2$. The cases where $i = 0, 2, 3, 4$ follow an identical construction. Let $\overline{D}^A(1^k)$ be defined by:

1. $f_4 \leftarrow F_k$
2. $f_3 \leftarrow A$
3. $u_2, u_1 \leftarrow U_k$
4. Construct the Feistel Network $g_{f_4} \circ g_{f_3} \circ g_{u_2} \circ g_{u_1}$
5. Run $D$ on the Feistel Network constructed in (4)

We can deduce that:

$$A \sim F_k \Rightarrow \overline{D}^A \text{ sees } (H_k^{(i)}, (H_k^{(i)})^{-1})$$

$$A \sim U_k \Rightarrow \overline{D}^A \text{ sees } (H_k^{(i+1)}, (H_k^{(i+1)})^{-1})$$

From here, it is straightforward to apply the remainder of the hybrid argument and we obtain our result (S1).