**CSA E0 235: Cryptography** March 2, 2016

# Scribe for Lecture 11

*Instructor: Arpita Patra* *Submitted by: Sameer Shah*

## Goal -

The main goal in this lecture is to construct a PRF using a PRG.
This is important in the 'roadmap' picture. We assumed existence of OWFs(- one way functions) and have proved existence of PRGs(- pseudo random generators). Today we shall explicitly construct a PRF(- pseudo random function) from a given PRG and hence prove if PRGs exist then so do PRFs.

## Idea -

Intuitively, a random function can be thought of as a very long random string. If we fix an ordering (natural here) on domain then the mapping could be the respective blocks of the long string. The same notion could be used for pseudo-random function and a pseudo-random string. But the problem here is, generating such long strings is not easy. Expansion factors in pseudo-random generators are very hard to increase keeping the same level of security. To solve this, we shall recursively use a length doubling PRG and cleverly define required mapping of a PRF.

## Recall -

Recall the security definitions for PRG and PRF. In the indistinguishability security definition of PRG, no adversary should be able to distinguish the output of the PRG from a truly random string of the same length with non-negligible probability. And for PRF security, no adversary should be able to distinguish the output of a PRF from the output of a truly random function with non-negligible probability. The output of a truly random function is equivalent to a truly random string. So we need to define output of PRF in terms of output of PRG such that they are indistinguishable from truly random strings. We will see the exact probability expressions when needed in the proof.

## Construction -

Let's now explicitly construct a PRF from a PRG. Let $G$ be a length doubling PRG on n-bit strings. Using this we shall get a PRF $F$.

$$G : \{0,1\}^n \rightarrow \{0,1\}^{2n} \qquad \Longrightarrow \qquad F : \{0,1\}^n \times \{0,1\}^n \rightarrow \{0,1\}^n$$

We first need to understand what it means to construct a PRF? There are two conditions. First, we need to define the mapping, an n-bit output string for every input we give to $F$. There are $2^n$ such mappings for fixed n-bit key $k$ of $F$. And second condition is that this mapping of $F$ should be poly computable for all $x$, i.e. $F_k(x)$ should be obtained in polynomial time.

We shall use a fundamental structure from graphs to define our function - A complete binary tree.

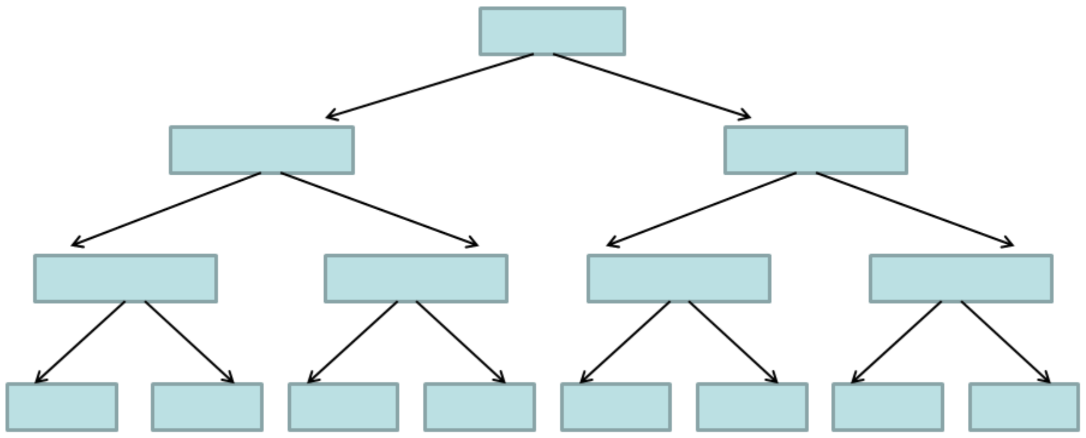
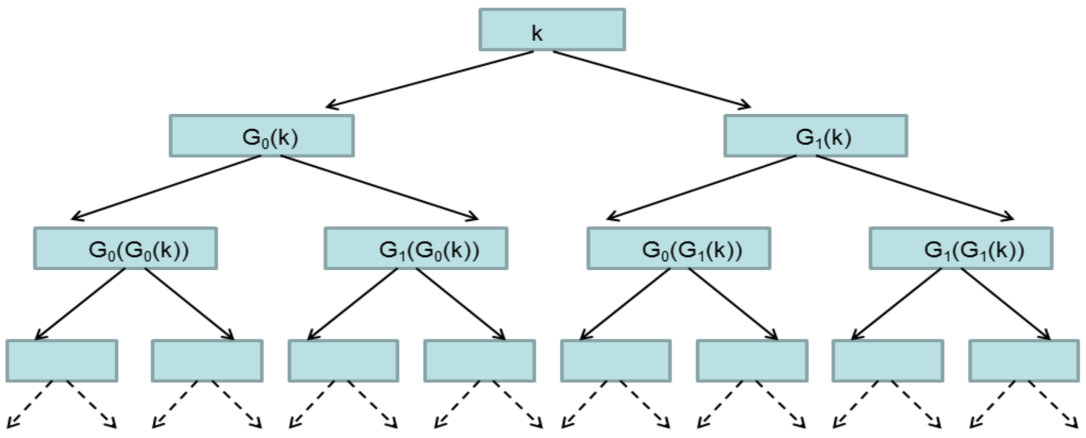


Figure 1: Complete Binary Tree of depth n = 3

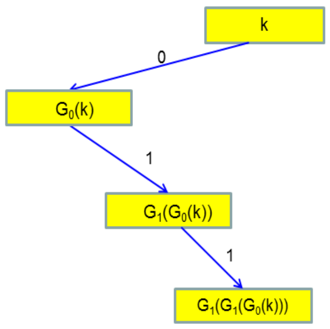
Let us briefly understand some basics of this structure. Considering the arrows shown in the figure as 'directed edges', we define some terms. The box or the 'node' with no edges coming towards it, shall be called as the 'root'. And a node with no edges going out of it shall be called as a 'leaf'. In the special case of a complete binary tree, there is a unique root, and $2^n$ many leaves where $n$ is the 'depth' of the tree. Depth of the tree can be perceived as the length of the path from the root to the leaf. All the leaves are at the same level.

We can naturally assign each leaf with an n-bit string as follows. Suppose at every node going to the left node in next level corresponds to 0 and going to right node in next level corresponds to 1, then the path from the root to any leaf will correspond to an n-bit string in a complete binary tree of depth n. Each leaf will have a unique path and thus a unique number corresponding to it. Also every number can be thought of as a path and will correspond to a unique leaf. This establishes a bijection between n-bit strings and the leaves of a complete binary tree. Now we fill this tree in a certain way with iterates of $G$, and $x$ will be mapped to the value in the leaf it corresponds to. And that is how we will define our $F$.

The key $k$ for a PRF is equivalent to the seed $s$ of the PRG. We fix the root of the tree to be the key $k$, same as the seed $s$ of the PRG $G$. As $G$ is length doubling, and we need the output of $F$ to be n-bit, we split the output of $G$ in two parts. Lets denote first n-bits by $G_0$ and the rest n-bits by $G_1$. The filling of tree is done level-wise. For a given n-bit string, we follow the procedure shown in figure to fill the values in the nodes of the tree.



Let us see an example for better understanding. Computation of $F_k(110)$ is shown below.



Notice that for every input, we do exactly n iterates of PRG to get the output, which is poly in $n$. Let $G^x(k) = G_{b_1}(G_{b_2}(...(G_{b_n}(k))...))$ be the specific composition corresponding to $x$ where $x = b_1 b_2 ... b_n$ is the n-bit string and $b_i$ are bits. Then

$$F_k(x) := G^x(k)$$

completely defines the function $F$. Now we need to prove that the function $F$ we get by this construction is indeed a PRF.

**Proof -**

We shall give the proof with the help of two intermediate lemmas and using hybrid arguments.

***Theorem :*** If $G : \{0,1\}^n \rightarrow \{0,1\}^{2n}$ is a PRG, i.e.,

$$|Pr[D(G(s)) = 1] - Pr[D(r) = 1]| \leq negl_G(n)$$

then $F$ constructed as above is a PRF, i.e.,

$$|Pr[D^{F_k}(1^n) = 1] - Pr[D^f(1^n) = 1]| \leq negl_F(n)$$

*Proof :* We break the proof of this theorem into two parts.

**Lemma 1 :** If $G : \{0,1\}^n \rightarrow \{0,1\}^{2n}$ is a PRG, i.e.,

$$|Pr[D(G(s)) = 1] - Pr[D(r) = 1]| \leq negl_G(n)$$

then for $r_1, ..., r_t \in_R \{0,1\}^{2n}$ and $s_1, ..., s_t \in_R \{0,1\}^n$ random strings or respective lengths

$$|Pr[A(r_1, ..., r_t) = 1] - Pr[A(G(s_1), ..., G(s_t) = 1] \leq negl_1(n)$$

Proof of Lemma 1 : We need to play an indistinguishability game between $(r_1, ..., r_t)$ and $(G(s_1), ..., G(s_t))$. But this does not fit into any of our already existing games directly. So we construct hybrids. Consider $H_i$ to be $(G(s_1), ..., G(s_i, r_{i+1}, ..., r_t)$ for $i = 1, ..., t-1$ with $H_0$ as $(r_1, ..., r_t)$ and $H_t$ as $(G(s_1), ..., G(s_t))$. Note that any two consecutive hybrids differ in only place. Consider $H_i$ and $H_{i-1}$. They have $G(s_i)$ and $r_i$ at their $i^{th}$ places respectively with all the other entries same. As $G$ is a PRG, we get

$$|Pr[A(G(s_i) = 1] - Pr[A(r_i) = 1]| \leq negl_G(n)$$

Now the simulation of the game is trivial and there is no loss of probability, thus we get

$$|Pr[A(G(s_1), ..., G(s_i), r_{i+1}, ..., r_t) = 1] - Pr[A(G(s_1), ..., G(s_{i-1}), r_i, ..., r_t) = 1]| \leq negl_G(n)$$

$$i.e. |Pr[A(H_i = 1] - Pr[A(H_{i-1}) = 1]| \leq negl_G(n)$$

This holds true for $i = 1, ..., t$. Using triangle inequality we get

$$\begin{aligned}|Pr[A(r_1, ..., r_t) = 1] - Pr[A(G(s_1), ..., G(s_t) = 1]| &= |Pr[A(H_t = 1] - Pr[A(H_0) = 1]| \\ &\leq \sum_{i=1}^{t} |Pr[A(H_i = 1] - Pr[A(H_{i-1}) = 1]| \\ &\leq t \cdot negl_G(n) \leq negl_1(n)\end{aligned}$$

This completes the proof of the first lemma.

**Lemma 2 :** If $G : \{0,1\}^n \to \{0,1\}^{2n}$ is such that

$$|Pr[A(r_1, ..., r_t) = 1] - Pr[A(G(s_1), ..., G(s_t) = 1] \leq negl_1(n)$$

then $F$ constructed as above is a PRF, i.e.,

$$|Pr[D^{F_k}(1^n) = 1] - Pr[D^f(1^n) = 1]| \leq negl_F(n)$$

Proof of lemma 2 : We again are going to use hybrid arguments. Defining hybrids in this case is more subtle. The hybrid $H_i$ is taken to be the distribution obtained on leaves when the nodes in level $i$ are filled with random strings with $H_0$ corresponding to the distribution on leaves when only the root was randomly selected, and $H_n$ corresponding to all the leaves themselves being randomly chosen. Note that $H_0$ is also equivalent to the uniform distribution on all the keyed functions and $H_n$ is to the uniform distribution on ALL the functions. The difference between any two consecutive hybrids, say $H_i$ and $H_{i-1}$ is that $H_i$ has truly random strings at level $i$ and $H_{i-1}$ has pseudo-random strings at that level. We need to do polynomial queries in this case to simulate the PRF game. By using Lemma 1 we get

$$|Pr[A^{H_i}(1^n) = 1] - Pr[A^{H_{i-1}}(1^n) = 1]| \leq negl_2(n)$$

And this is true for all the steps. Using triangle inequality we get

$$|Pr[D^{F_k}(1^n) = 1] - Pr[D^f(1^n) = 1]| = |Pr[A^{H_0}(1^n) = 1] - Pr[A^{H_n}(1^n) = 1]|$$

$$\leq \sum_{i=1}^{n} |Pr[A^{H_i}(1^n) = 1] - Pr[A^{H_{i-1}}(1^n) = 1]|$$

$$\leq n \cdot negl_2(n) \leq negl_F(n)$$

This completes the proof of the lemma 2 and thus the proof for the theorem.

## Conclusion -

With this theorem we conclude the journey of implications. We have constructed PRFs with which we built many security schemes in the beginning. These results have profound theoretical value though these can not be directly implemented practically.