COM S 6830 – Cryptography Oct 4, 2011

Lecture 11: Probabilistic Random Functions

Instructor: Rafael Pass *Scribe: Shentong Wang*

1 Encryption Definition

1.1 Shannon's Definition

As discussed before, Shannon's definition for security of an encryption algorithm ($\boldsymbol{Gen}, \boldsymbol{Enc}, \boldsymbol{Dec}$) over message space $\boldsymbol{M}$ and key space $\boldsymbol{K}$ involve indistinguishability over all computable programs defined as follows:

$$\forall m_i, m_j \in \boldsymbol{M}, \{k \leftarrow \boldsymbol{Gen} : \boldsymbol{Enc}(m_i)\} = \{k \leftarrow \boldsymbol{Gen} : \boldsymbol{Enc}(m_j)\}$$

1.2 Refined Definition

This definition can be refined to take into account the assumption that potential attackers have access to polynomial resources. Thus, an encryption algorithm ($\boldsymbol{Gen}, \boldsymbol{Enc}, \boldsymbol{Dec}$) over message space $\boldsymbol{M}$ and key space $\boldsymbol{K}$ is said to be single message secure if:

$$\forall \mathcal{A} \in \boldsymbol{n.u.p.p.t.}, \exists \epsilon \in \boldsymbol{neg}, \forall n \in N, \forall m_i, m_j \in \boldsymbol{M} : |m_i| = |m_j|,$$
$$|Pr(k \leftarrow \boldsymbol{Gen} : \mathcal{A}(1^n, Enc_k(m_i)) = 1) - Pr(k \leftarrow \boldsymbol{Gen} : \mathcal{A}(1^n, Enc_k(m_j)) = 1)| \leq \epsilon(n)$$

It is important to note that the 1^n input parameter for $\mathcal{A}$ is optional, because $\boldsymbol{Enc}$ can not shrink the input m. If it did, then $\boldsymbol{Dec}$ would not be able to deterministically decrypt the output.

2 Single Message Secure Encryption

From this definition of secure encryption, an encryption algorithm ($\boldsymbol{Gen}, \boldsymbol{Enc}, \boldsymbol{Dec}$) over message space $\boldsymbol{M}$ and key space $\boldsymbol{K}$which is secure over single messages can be defined as follows:
Let $\boldsymbol{G}$ be a length

1. $\boldsymbol{Gen}(1^n) : k \leftarrow \{0, 1\}^n$
2. $\boldsymbol{Enc}_k(m) = m \oplus r$, where $r = \boldsymbol{G}(k)$
3. $\boldsymbol{Dec}_k(c) = c \oplus r$, where $r = \boldsymbol{G}(k)$

The advantage of this encryption algorithm is that unlike the one-time pad which satisfies Shannon's definition of secure encryption, this encryption algorithm uses a key which is half as long as the message itself. One can see that this example can be extended to create encryption algorithms which uses keys much smaller than the message.
The proof that this encryption algorithm satisfies the refined definition for secure encryption is straight forward, but should be instructive in illustrating how to prove computational indistinguishability:
Proof. Assume for contradiction:

$$\exists \mathcal{A} \in \boldsymbol{n.u.p.p.t.}, \boldsymbol{p} \in \boldsymbol{poly}, \text{for infinitely many n}, \exists m_i, m_j \in \boldsymbol{M},$$
$$\mathcal{A} \text{ distinguishes, w.p. } \frac{1}{p(n)}, \{k \leftarrow \boldsymbol{Gen} : m_i \oplus \boldsymbol{G}(k)\} \text{ and } \{k \leftarrow \boldsymbol{Gen} : m_j \oplus \boldsymbol{G}(k)\}$$

The first step to approaching this sort of problem would be to define hybrids which link the two distributions in question, $\{k \leftarrow \boldsymbol{Gen} : m_i \oplus \boldsymbol{G}(k)\}$ and $\{k \leftarrow \boldsymbol{Gen} : m_j \oplus \boldsymbol{G}(k)\}$. Thus, define the following hybrid distributions, H_n^1, H_n^2, H_n^3, and H_n^4 as follows:

1. $H_n^1 = \{k \leftarrow \boldsymbol{Gen} : m_i \oplus \boldsymbol{G}(k)\}$
2. $H_n^2 = \{r \leftarrow U_n : m_i \oplus r\}$
3. $H_n^3 = \{r \leftarrow U_n : m_j \oplus r\}$
4. $H_n^4 = \{k \leftarrow \boldsymbol{Gen} : m_j \oplus \boldsymbol{G}(k)\}$

From the hybrid lemma, it can be said that if $\mathcal{A}$ distinguishes H_n^1 and H_n^4, there exists one i, where $\mathcal{A}$ distinguishes H_n^i and H_n^{i+1} with probability greater than or equal to $\frac{1}{3p(n)}$. Thus, each of the H_n^i and H_n^{i+1} cases can be examined individually.

1. $i \neq 1$, because $\{\boldsymbol{G}(U_{\frac{n}{2}})\}_n \approx \{U_n\}_n$, and $m \oplus r$ is a polynomial time operation. Therefore, by the closure under efficient operations lemma, we see that $\mathcal{A}$ can't distinguish H_n^i and H_n^{i+1}, because $H_n^i = \{\boldsymbol{G}(U_{\frac{n}{2}}) \oplus m_i\}_n$ and $H_n^{i+1} = \{U_n \oplus m_i\}_n$.
2. $i \neq 2$, because $\{r \leftarrow U_n : m_i \oplus r\} = \{r \leftarrow U_n : m_j \oplus r\}$
3. $i \neq 3$, because the proof for the $i \neq 1$ case can be extended to m_j.

3 Multi-Message Encryption

After defining single message encryption, an attempt can be made to define the same properties for multiple messages. Thus, an encryption algorithm $(\boldsymbol{Gen}, \boldsymbol{Enc}, \boldsymbol{Dec})$ over message space $\boldsymbol{M}$ and key space $\boldsymbol{K}$is said to be multi-message secure if:

$$\forall \mathcal{A} \in \boldsymbol{n.u.p.p.t.}, \exists \boldsymbol{q} \in \boldsymbol{poly}, \exists \epsilon \in \boldsymbol{neg}, \forall n \in N, \forall m_1 ... m_{q(n)}, m'_1 ... m'_{q(n)} \in \boldsymbol{M},$$
$$|Pr(k \leftarrow \boldsymbol{Gen} : \mathcal{A}(1^n, \mathcal{A}(Enc_k(m_1), Enc_k(m_2)...Enc_k(m_{q(n)})) = 1)) = 1) -$$
$$Pr(k \leftarrow \boldsymbol{Gen} : \mathcal{A}(Enc_k(m'_1), Enc_k(m'_2)...Enc_k(m'_{q(n)})) = 1)| \leq \epsilon(n)$$

An important aspect to notice is that the same key value $k \leftarrow \boldsymbol{Gen}$ will be used for all messages $[m_0...m_{q(n)}]$. This invalidates the previous encryption scheme proposed for single message security. This can be shown when the same message exist in a stream. Because the single message secure encryption algorithm defined earlier is stateless and deterministic, $Enc_k(m)$ is unique. Thus, a distinguisher which distinguishes message streams with repeated messages can be created by detecting repeats in the encrypted message blocks.
Thus, it can be shown that any deterministic and stateless encryption scheme is insecure for that very reason.

Theorem 1 *There does not exist a deterministic and stateless multi-message secure encryption algorithm* $(\mathbf{Gen}, \mathbf{Enc}, \mathbf{Dec})$ *over message space* $\mathbf{M}$ *and key space* $\mathbf{K}$.
Proof. *In such an encryption algorithm,* $Enc_k(m_i) = Enc_k(m_j)$ *if* $m_i = m_j$*. Thus, all message streams where there exists* m_i *and* m_j*, where* $i \neq j \wedge m_i = m_j$*, can be distinguished from all message streams where each message in the stream is different from all other messages in the streams, by a turing machine which checks for repeats for all block repeats in the encrypted stream.*

Thus, it becomes imperative to discover an encryption algorithm which is either stateful or non-deterministic.

3.1 First Attempt[Stateful]

Recall the construction of $\boldsymbol{G}$, which used a OWP and the hardcore bit of the OWP to create an PRG. Thus, using an initial key $k \leftarrow \boldsymbol{Gen}$, we can a stream of pseudo-random bits can be generated. Thus, a block index can be kept within the $\boldsymbol{Enc}$ which generates the next block for each call to $\boldsymbol{Enc}$.
The proof of the security of the scheme will not be given in great detail. The intuition is to form a hybrid between $G_i(k)$, where G_i is the i^{th} iteration of G and the uniform distribution.
But this solution is not preferrable as keeping consistent state is sometimes undesirable.

3.2 Second Attempt

One can extend the first attempt by instead of keeping an i as state within the OWP G, a random i can be generated at each point. An important property of this scheme is that only the sender needs to keep state, since the sender can send the randomly generated index i appended to the message for the receiver to process.

The main flaw with this approach exists when a random i can repeat itself. Thus, in order to decrease the possibility of an repeated i, the range of i will have to be super polynomial. This runs into the polynomial bound constraint for G. Since i is within a super-polynomial range, doing the linear approach as in the previous attempt can result in potentially an super-polynomial running time.

Thus, define a Pseudo Random Function, $\boldsymbol{F} : \{0,1\}^n \rightarrow \{0,1\}^n$, which computes the "$i^{th}$" block of G in polynomial time. After which, ***Enc*** becomes:

$$Enc_k(m) = \{i \leftarrow \{0,1\}^n; m \oplus \boldsymbol{F}(i)\}$$