Encryption

MAR 16

abhi shelat
A **predicate** is a function $b : \{0, 1\}^* \to \{0, 1\}$

A **hard-core predicate** for a function $f$

A predicate is a function $b : \{0, 1\}^* \to \{0, 1\}$

A hard-core predicate for a function $f$
A predicate is a function \( b : \{0, 1\}^* \to \{0, 1\} \)

A hard-core predicate for a function \( f \)

\[
\begin{array}{ccc}
\text{x} & \xrightarrow{\text{easy}} & \text{f(x)} & \text{k bits} \\
\downarrow & & \downarrow & \\
\text{easy} & \text{hard} & \text{b(x)} & \text{1 bit}
\end{array}
\]

\[
\Pr[x \leftarrow U_n; \ A(1^n, f(x)) = b(x)] < \frac{1}{2} + \epsilon(n)
\]
“for any pair of messages $m_1, m_2$, $Eve$ cannot tell whether $c = Enc_k(m_i)$.”
PRIVATE KEY ENCRYPTION

<table>
<thead>
<tr>
<th>Gen</th>
<th>Enc</th>
<th>Dec</th>
<th>( \mathcal{M} )</th>
<th>( \mathcal{K} )</th>
</tr>
</thead>
<tbody>
<tr>
<td>3 algorithms</td>
<td>2 sets</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Gen (key generation)**

\( k \leftarrow \text{Gen}(1^n) \text{ s.t. } k \in \mathcal{K} \)

**Enc (encryption)**

\( c \leftarrow \text{Enc}_k(m) \text{ for } k \in \mathcal{K}, m \in \mathcal{M} \)

**Dec (decryption)**

\( \forall m \in \mathcal{M}, k \in \mathcal{K} \)

\( \Pr[\text{Dec}_k(\text{Enc}_k(m)) = m] = 1 \)
PERFECT SECRECY

$(\text{Gen}, \text{Enc}, \text{Dec}, \mathcal{M}, K)$

is said to be perfectly secret if

$$\forall m_1, m_2 \in \mathcal{M} \text{ s.t. } |m_1| = |m_2|, \forall c$$

$$\Pr[k \leftarrow \text{Gen} : \text{Enc}_k(m_1) = c] = \Pr[k \leftarrow \text{Gen} : \text{Enc}_k(m_2) = c]$$
PERFECT SECRECY

(Gen, Enc, Dec, $\mathcal{M}$, $\mathcal{K}$)
is said to be perfectly secret if

\[ \forall m_1, m_2 \in \mathcal{M} \text{ s.t. } |m_1| = |m_2|, \forall c \]

\[ \{ k \leftarrow \text{Gen}(1^n) : \text{Enc}_k(m_1) \} \]

\[ = \]

\[ \{ k \leftarrow \text{Gen}(1^n) : \text{Enc}_k(m_2) \} \]
SECURE ENCRYPTION
(For one message)

Def: An enc scheme is computationally secret if
\[ \forall m_0, m_1 \in \mathbb{Z}_0, 1^{\ell_n}, \text{ we have that} \]
\[ \exists K \leftarrow \text{Gen}(1^n) : \text{Enc}_K(m_1) \approx_\text{n} \text{Enc}_K(m_2) \]
\[ \approx_\text{n} \]
\[ \exists K \leftarrow \text{Gen}(1^n) : \text{Enc}_K(m_2) \approx_\text{n} \]
AN ENCRYPTION SCHEME (private-key)

\[ \text{Gen}(1^n) : \quad k \leftarrow \mathcal{U}_{1^{n/2}} \]

\[ \text{Enc}(k, m) : \quad \text{Output} \quad G(k) \oplus m \]

\[ \text{PRG that expands } \frac{n}{2} \rightarrow n \]

\[ \text{Dec}(k, c) : \quad \text{Output} \quad G(k) \oplus c \]

like one-time pad except w/PRG.
AN ENCRYPTION SCHEME

Gen(1^n)

\( k \leftarrow U_{n/2} \)  
(key generation)

Enc_k(m)

(encryption)

Dec_k(c)

Dec_k(c)

\( r \leftarrow G(k) \quad |r| = n \)
(output)

\( m \oplus r \)
(decryption)
SECURITY PROOF

\[ \{ G(k) \} \approx \{ \mathcal{U} \} \quad \text{by security of PRG} \]

1. \[ \{ G(k) \oplus m_0 \} \approx \{ \mathcal{U} \oplus m_0 \} \quad \text{by closure} \]

   \| \quad \text{by perfect security of one-time pad.} \]

2. \[ \{ G(k) \oplus m_1 \} \approx \{ \mathcal{U} \oplus m_1 \} \quad \text{by closure} \]

\[ \implies \text{By hybrid lemma, } 0 \approx \psi. \]
WORKS VERY WELL FOR ONE MESSAGE
SECURITY FOR
MULTIPLE MESSAGES
MANY-MESSAGE SECURITY (private key)

(\text{Gen, Enc, Dec, } \mathcal{M}, K)

is said to be \textit{many message secure} if 

\begin{align*}
\text{for every polynomial } \ell(n) & \quad \text{and } \\
\left( m_0, m_1, \ldots, m_{\ell(n)} \right), \left( m'_0, m'_1, \ldots, m'_{\ell(n)} \right) & \quad \text{st. } |m_i| = |m'_i| \leq n \\
\end{align*}

\begin{align*}
\exists K \in \text{Gen}(1^n), \quad & \text{Enc}_K(m_0), \text{Enc}_K(m_1), \ldots, \text{Enc}_K(m_{\ell(n)}) \\
\therefore & \approx \\
\exists K \in \text{Gen}(1^n), \quad & \text{Enc}_K(m'_0) \ldots \text{Enc}_K(m'_{\ell(n)})
\end{align*}
Thus: There are no deterministic, stateless encryption schemes.
BUILDING MANY-MESSAGE SECURE SCHEMES
PRG GENERATE
PSEUDO-RANDOM
STRINGS

POLY NUMBER OF BITS
RANDOM FUNCTIONS

\[ R : \{0,1\}^n \rightarrow \{0,1\}^n \]

(length preserving)

<table>
<thead>
<tr>
<th>In</th>
<th>Out</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>010...00</td>
</tr>
<tr>
<td>1</td>
<td>110..110</td>
</tr>
<tr>
<td>2</td>
<td>001..100</td>
</tr>
<tr>
<td>3</td>
<td>110...001</td>
</tr>
<tr>
<td>...</td>
<td></td>
</tr>
<tr>
<td>(2^n - 1)</td>
<td>100...111</td>
</tr>
</tbody>
</table>

Encr(k, m): \( r \in R\)

\[ c \leftarrow R_k(r) \oplus m \]

output \((r, c)\)
**HOW MANY RANDOM FUNCTIONS ARE THERE?**

\[ R : \{0, 1\}^n \rightarrow \{0, 1\}^n \]

<table>
<thead>
<tr>
<th>In</th>
<th>Out</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>010...00</td>
</tr>
<tr>
<td>1</td>
<td>110...110</td>
</tr>
<tr>
<td>2</td>
<td>001..100</td>
</tr>
<tr>
<td>3</td>
<td>110...001</td>
</tr>
<tr>
<td>...</td>
<td></td>
</tr>
<tr>
<td>(2^n-1)</td>
<td>100...111</td>
</tr>
</tbody>
</table>

- **Description length of a RF:**
  \[ \frac{n2^n}{n2^n} \]
- \# of RF is \[2^{n2^n}\]

\[ \text{proving } 2^n \text{ vs } 2^{n2^n} \]
**HOW MANY RANDOM FUNCTIONS ARE THERE?**

\[ R : \{0, 1\}^n \to \{0, 1\}^n \]

<table>
<thead>
<tr>
<th>In</th>
<th>Out</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>010...00</td>
</tr>
<tr>
<td>1</td>
<td>110...110</td>
</tr>
<tr>
<td>2</td>
<td>001..100</td>
</tr>
<tr>
<td>3</td>
<td>110...001</td>
</tr>
<tr>
<td>...</td>
<td></td>
</tr>
<tr>
<td>(2^{n-1})</td>
<td>100...111</td>
</tr>
</tbody>
</table>

total space of table:
**HOW MANY RANDOM FUNCTIONS ARE THERE?**

\[ R : \{0, 1\}^n \rightarrow \{0, 1\}^n \]

<table>
<thead>
<tr>
<th>In</th>
<th>Out</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>010...00</td>
</tr>
<tr>
<td>1</td>
<td>110...110</td>
</tr>
<tr>
<td>2</td>
<td>001..100</td>
</tr>
<tr>
<td>3</td>
<td>110...001</td>
</tr>
<tr>
<td>...</td>
<td></td>
</tr>
<tr>
<td>(2^n-1)</td>
<td>100...111</td>
</tr>
</tbody>
</table>

**total space of table:** \(2^n n\)
**HOW MANY RANDOM FUNCTIONS ARE THERE?**

Let $R : \{0, 1\}^n \to \{0, 1\}^n$

<table>
<thead>
<tr>
<th>In</th>
<th>Out</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>010...00</td>
</tr>
<tr>
<td>1</td>
<td>110...110</td>
</tr>
<tr>
<td>2</td>
<td>001..100</td>
</tr>
<tr>
<td>3</td>
<td>110...001</td>
</tr>
<tr>
<td>...</td>
<td></td>
</tr>
<tr>
<td>$2^{n-1}$</td>
<td>100...111</td>
</tr>
</tbody>
</table>

**total space of table:** $2^n n$

**total # of random f:** $2^{2^n n}$
DEFINING
PSEUDO-RANDOM
FUNCTIONS
FAMILIES OF FUNCTIONS

\{ RF_n \}_n \text{ uniform distribution over all } n\text{-bit random functions}

\{ F_n \}_n \text{ uniform distribution over all } n\text{-bit random functions}
DEFINING A PRF

distinguisher is given one of these two functions and must guess which
DEFINING A PRF

distinguisher is given one of these two functions and must guess which

"give" a function???
distinguisher is given oracle access to one of these two functions and must guess which
DEFINING A PRF

distinguisher is given oracle access to one of these two functions and must guess which

\[
\Pr[k \in U_n : D^{f_k}(1^n) = 1] - \Pr[k \in U_n : D^{RF}(1^n) = 1] < \epsilon(n)
\]
DEFINING A PRF

A distinguisher is given oracle access to one of these two functions and must guess which

\[
\Pr[f_k \leftarrow F_n; \ D^{f_k}(1^n) = 1] - \Pr[R \leftarrow RF_n; \ D^R(1^n) = 1] < \epsilon(n)
\]
A function family \( F_n = \{ f_k \} \) is a pseudo-random function family if 

\[
\forall \mathcal{D} \exists \mathcal{R} \quad \text{such that} \quad \Pr[f_k \leftarrow F_n : D^{f_k}(1^n) = 1] - \Pr[R \leftarrow R_{\mathcal{F}_n} : D^R(1^n) = 1] < \epsilon(n)
\]
INTUITION BEHIND CONSTRUCTION OF PRFs

Start with

\[ g(s) = g_0(s) \| g_1(s) \]

\[ F_{K_i}(i) : \{0,1\}^n \rightarrow \{0,1\}^{2n} \]

\[ G(i) : 2n \text{ bit string} \]

n-bit seed

n-bit

\[ 011010110 \uparrow \]

50,000
INTUITION BEHIND CONSTRUCTION OF PRFs

start with

\[ G(s) : \{0, 1\}^n \rightarrow \{0, 1\}^{2n} \]

\[ f_K(b) = \gamma \oplus \oplus_{i=1}^{2n} \]
INTUITION BEHIND CONSTRUCTION OF PRFs

start with

\[ G(s) : \{0, 1\}^n \to \{0, 1\}^{2n} \]

\[ G(s) \to G_0(s) \parallel G_1(s) \]
\[ |G_0(s)| = |G_1(s)| \]

This is a 1-bit PRF.

\[ F_k(b) = G_{b'}(k) \]
FIRST STEP

\[ f_k : \{0, 1\} \rightarrow \{0, 1\}^n \]

\[ k \leftarrow U_n \]

\[ f_k(b) = G_b(k) \]

\[ f_K(b, b_2) = \]
FIRST STEP

\[ f_k : \{0, 1\} \rightarrow \{0, 1\}^n \]

\[ k \leftarrow U_n \]

\[ f_k(b) = G_b(k) \]
FIRST STEP

\[ f_k : \{0, 1\} \rightarrow \{0, 1\}^n \]

\[ k \leftarrow U_n \]

\[ f_k(b) = G_b(k) \]
FIRST STEP

$$f_k : \{0, 1\} \rightarrow \{0, 1\}^n$$

$$k \leftarrow U_n$$

$$f_k(b) = G_b(k)$$
SECOND STEP

\[ f_k : \{0, 1\}^2 \rightarrow \{0, 1\}^n \]

\[ k \leftarrow U_n \]

\[ f_k(b_1 b_2) = \left( g_{b_1}(g_{b_2}(k)) \right) \]
\[ f_k(b_1 b_2) = G_{b_1} \left( G_{b_2}(k) \right) \]
\[ f_k(b_1 b_2) = G_{b_1} \left( G_{b_2}(k) \right) \]
GENERAL STEP

\[ f_k : \{0, 1\}^n \rightarrow \{0, 1\}^n \]

\[ k \leftarrow U_n \]

\[ f_k(b_1 \ldots b_n) = G_{b_1} (G_{b_2} (\ldots G_{b_n}(k))) \]
RANDOM FUNCTION

\[ b_1, b_2, \ldots, b_n \]
PROOF: SUPPOSE NOT
PROOF: SUPPOSE NOT

\[ \Pr[f_k \leftarrow F_n; D^f_k(1^n)] = 1 - \Pr[R \leftarrow RF_n; D^R(1^n)] > \frac{1}{p(n)} \]
PROOF: SUPPOSE NOT

\[
\Pr[f_k \leftarrow F_n; D^{f_k}(1^n)] = 1 - \Pr[R \leftarrow RF_n; D^R(1^n)] > \frac{1}{p(n)}
\]
1. Hybrid argument by level
2. Hybrid argument by query