Cryptography

Lecture 4:
From OWFs to PRGs and PRFs

Gil Segev
Recall: Lecture 3

Do PRFs exist? Can we construct a PRF?

PRF \rightarrow PRG

CPA-secure symmetric-key encryption \rightarrow IND-secure symmetric-key encryption
Practical Heuristics: Block Ciphers

• In practice, block ciphers are designed to be secure instantiations of pseudorandom permutations (PRPs)
• A block cipher is an efficiently-computable keyed permutation

\[ F: \{0,1\}^n \times \{0,1\}^\ell \rightarrow \{0,1\}^\ell \]

\[ F_k: \{0,1\}^\ell \rightarrow \{0,1\}^\ell \text{ is a permutation for any key } k \]

• Concrete security rather than asymptotic security
• A block cipher is considered “secure” if the best known “attack” requires time roughly \(2^n\) (\(\approx\)brute-force search for the key)
Practical Heuristics: Block Ciphers

**DES: The Data Encryption Standard**
- Developed in the 1970s by IBM (with help from the NSA), adopted in 1977
- Key length is 56 bits, block length is 64 bits

**Round function**
- Input: Plaintext (64 bits)
- Process:
  - Initial permutation (IP)
  - 16 rounds of substitution and permutation
  - Final permutation (FP)
- Output: Ciphertext (64 bits)

**Key schedule**
- Input: Key (64 bits)
- Process:
  - PC1
  - Subkeys (48 bits)
  - PC2
- Output: S1, S2, S3, ..., S16
Practical Heuristics: Block Ciphers

**DES: The Data Encryption Standard**
- Developed in the 1970s by IBM (with help from the NSA), adopted in 1977
- Key length is 56 bits, block length is 64 bits
- Best known attack in practice is essentially brute-force key search ($\approx 2^{56}$)
- However, no longer considered secure due to its short key length
- Remains widely-used in the strengthened form of 3DES:

$$3\text{DES}_{k_1,k_2,k_3}(x) = \text{DES}_{k_1}\left(\text{DES}_{k_2}^{-1}\left(\text{DES}_{k_3}(x)\right)\right)$$

3×56-bit keys but can be broken in time $2^{2\times56}$

...and also slower than DES
Practical Heuristics: Block Ciphers

AES: The Advanced Encryption Standard

- In 1997 NIST published a call for candidate block ciphers to replace DES
- 15 candidates were proposed by different teams from all over the world
- Each candidate extensively analyzed by the public and by the other teams
- The winner ("Rijndael") was announced in late 2000 (based on security, efficiency, performance in hardware,...)
- Key length is 128/192/256 bits, block length is 128 bits
- To date, no known practical attacks better than brute-force key search

Various design paradigms with insightful structures

Can we do more than relying on heuristics?
This Week

One-Way Functions

PRF

CPA-secure symmetric-key encryption

PRG

IND-secure symmetric-key encryption

Problem set 2
One-Way Functions (OWFs)

• Easy to compute, but hard to invert
One-Way Functions (OWFs)

Definition:
A poly-time computable function $f: \{0,1\}^* \rightarrow \{0,1\}^*$ is one way if for every PPT algorithm $\mathcal{A}$ there exists a negligible function $\nu(\cdot)$ such that

$$\Pr[\mathcal{A}(f(x)) \in f^{-1}(f(x))] \leq \nu(n)$$

where $x \leftarrow \{0,1\}^n$.

- $f$ is length preserving if $|f(x)| = |x|$.
- $f$ can even be a permutation over $\{0,1\}^n$. 
Candidate OWFs

• We don’t know how to prove that OWFs exist (would imply $P \neq NP$)
• We conjecture their existence based on computational problems that have received much attention

Example 1: Based on the average-case hardness of integer factorization
$f(x, y) = x \cdot y$ for equal-length primes $x$ and $y$

Example 2: Based on the average-case hardness of subset sum

$$f(x_1, \ldots, x_n, S) = \left( x_1, \ldots, x_n, \sum_{i \in S} x_i \right)$$

Example 3: Based on the average-case hardness of discrete logarithm

$$f_{p, g}(x) = g^x \mod p$$
Theorem:
If $P = NP$ then there are no one-way functions

Proof idea:
• Given a function $f$ consider the following $NP$ language

$$L_f \overset{\text{def}}{=} \{(x^*, y) \mid \exists x \text{ s.t. } f(x) = y \text{ and } x^* \text{ is a prefix of } x\}$$

• If $P = NP$ then $L_f \in P$
• Given $y = f(x)$ and a polynomial-time algorithm for deciding membership in $L_f$, can find some $x' \in f^{-1}(y)$ bit by bit
Hard-Core Predicates

• OWF: Given $f(x)$ it is hard to compute $x$
• But what about learning some partial information about $x$?

$$g(x_1, x_2) = (x_1, f(x_2))$$ where $|x_1| = |x_2| = n/2$

If $f$ is one-way then $g$ is one-way, but can always recover half of $g$’s input...

• Is there a predicate $h$ such that it is hard to compute $h(x)$ given $f(x)$?
Definition:
A poly-time computable function $hc: \{0,1\}^* \rightarrow \{0,1\}$ is a hard-core predicate for a function $f$ if for every PPT algorithm $\mathcal{A}$ there exists a negligible function $\nu(\cdot)$ such that

$$\Pr_{x \leftarrow U_n}[\mathcal{A}(f(x)) = hc(x)] \leq \frac{1}{2} + \nu(n)$$

• What about $hc(x) = \bigoplus_{i \in [n]} x_i$? Not hard-core for $g(x) = (f(x), \bigoplus_{i \in [n]} x_i)$.

• Assuming the existence of OWFs, for every predicate $hc$ there exists a OWF for which $hc$ is not a hard-core predicate.

Claim (Problem Set 2):
If a one-to-one function $f$ has a hard-core predicate, then $f$ is a one-way function.
A Hard-Core Predicate for Any OWF

• If $f(x)$ is OWF then it hides the exclusive-or of a random subset of the bits of $x$

**Theorem (Goldreich-Levin ‘89):**
Let $f$ be a OWF and define $g(x,r) = (f(x), r)$. Then $gl(x,r) = \bigoplus_{i \in [n]} x_i \cdot r_i$ is a hard-core predicate for $g$.

• Note: If $f$ is permutation then $g$ is a permutation
Theorem:
Let $f$ be a OWP with a hard-core predicate $hc$. Then $G(s) = (f(s), hc(s))$ is a PRG with expansion $\ell(n) = n + 1$.

Proof idea:
• $f$ is a permutation $\Rightarrow f(s)$ is uniformly distributed
• $hc$ is hard-core $\Rightarrow hc(s)$ is pseudorandom even given $f(s)$
From OWPs to PRGs

Proof:
• Assume towards a contradiction that there exists a PPT $\mathcal{D}$ and a non-negligible $\epsilon(n)$ such that

$$\left| \Pr_{s \leftarrow \{0,1\}^n} [\mathcal{D}(f(s), hc(s)) = 1] - \Pr_{r \leftarrow \{0,1\}^{n+1}} [\mathcal{D}(r) = 1] \right| \geq \epsilon(n)$$

• We show a PPT $\mathcal{A}$ that predicts $hc(s)$ given $f(s)$ with a non-negligible advantage

Step 1: Show that $\mathcal{D}$ distinguishes $(f(s), hc(s))$ from $(f(s), \overline{hc(s)})$

Step 2: Consider $\mathcal{A}$ that on input $f(s)$ samples $z \leftarrow \{0,1\}$ and executes $\mathcal{D}(f(s), z)$

Claim:

$$\left| \Pr_{s \leftarrow \{0,1\}^n} [\mathcal{D}(f(s), hc(s)) = 1] - \Pr_{s \leftarrow \{0,1\}^n} [\mathcal{D}(f(s), \overline{hc(s)}) = 1] \right| \geq 2\epsilon(n)$$
From OWPs to PRGs

Claim:
\[
\left| \Pr_{s \leftarrow \{0,1\}^n}[\mathcal{D}(f(s), hc(s)) = 1] - \Pr_{s \leftarrow \{0,1\}^n}[\mathcal{D}(f(s), \overline{hc(s)}) = 1] \right| \geq 2\varepsilon(n)
\]

Proof:
- \( f \) is a permutation \( \Rightarrow f(U_n) \equiv U_n \)

\[
\Pr_{r \leftarrow \{0,1\}^{n+1}}[\mathcal{D}(r) = 1] = \Pr_{s \leftarrow \{0,1\}^n, z \leftarrow \{0,1\}}[\mathcal{D}(f(s), z) = 1]
\]

\[
= \frac{1}{2} \cdot \Pr_{s \leftarrow \{0,1\}^n}[\mathcal{D}(f(s), hc(s)) = 1] + \frac{1}{2} \cdot \Pr_{s \leftarrow \{0,1\}^n}[\mathcal{D}(f(s), \overline{hc(s)}) = 1]
\]

\[
\varepsilon(n) \leq \left| \Pr_{s \leftarrow \{0,1\}^n}[\mathcal{D}(f(s), hc(s)) = 1] - \Pr_{r \leftarrow \{0,1\}^{n+1}}[\mathcal{D}(r) = 1] \right|
\]

\[
= \frac{1}{2} \cdot \left| \Pr_{s \leftarrow \{0,1\}^n}[\mathcal{D}(f(s), hc(s)) = 1] - \Pr_{s \leftarrow \{0,1\}^n}[\mathcal{D}(f(s), \overline{hc(s)}) = 1] \right|
\]
From OWPs to PRGs

Claim:

\[
\left| \Pr_{s \leftarrow \{0,1\}^n} [\mathcal{D}(f(s), hc(s)) = 1] - \Pr_{s \leftarrow \{0,1\}^n} [\mathcal{D}(f(s), \overline{hc(s)}) = 1] \right| \geq 2\epsilon(n)
\]

• Assume without loss of generality that

\[
\Pr_{s \leftarrow \{0,1\}^n} [\mathcal{D}(f(s), hc(s)) = 1] > \Pr_{s \leftarrow \{0,1\}^n} [\mathcal{D}(f(s), \overline{hc(s)}) = 1]
\]

The algorithm \( \mathcal{A} \):

• On input \( f(s) \) sample \( z \leftarrow \{0,1\} \) and compute \( b = \mathcal{D}(f(s), z) \)
• If \( b = 1 \) output “hc(s) is z”, and otherwise output “hc(s) is 1 − z”
From OWPs to PRGs

The algorithm $A$:
- On input $f(s)$ sample $z \leftarrow \{0,1\}$ and compute $b = \mathcal{D}(f(s), z)$
- If $b = 1$ output “hc(s) is $z$”, and otherwise output “hc(s) is $1 - z$”

$$\Pr_{s \leftarrow \{0,1\}^n}[A(f(s)) = \text{hc}(s)]$$

$$= \frac{1}{2} \cdot \Pr_{s \leftarrow \{0,1\}^n}[\mathcal{D}(f(s), z) = 1 \mid z = \text{hc}(s)] + \frac{1}{2} \cdot \Pr_{s \leftarrow \{0,1\}^n}[\mathcal{D}(f(s), z) = 0 \mid z \neq \text{hc}(s)]$$

$$= \frac{1}{2} \cdot \Pr_{s \leftarrow \{0,1\}^n}[\mathcal{D}(f(s), \text{hc}(s)) = 1] + \frac{1}{2} \cdot \left(1 - \Pr_{s \leftarrow \{0,1\}^n}[\mathcal{D}(f(s), \overline{\text{hc}(s)}) = 1]\right)$$

$$= \frac{1}{2} + \frac{1}{2} \cdot \left(\Pr_{s \leftarrow \{0,1\}^n}[\mathcal{D}(f(s), \text{hc}(s)) = 1] - \Pr_{s \leftarrow \{0,1\}^n}[\mathcal{D}(f(s), \overline{\text{hc}(s)}) = 1]\right)$$

$$\geq \frac{1}{2} + \frac{1}{2} \cdot 2\epsilon$$
**Theorem:**
Let $G$ be a PRG with $\ell(n) = n + 1$. Then, for any polynomial $p(n)$ there exists a PRG $G'$ with expansion $\ell(n) = p(n)$.

**Hybrid $\mathcal{H}_i$:**
Replace $s_0, \ldots, s_i, \sigma_1, \ldots, \sigma_i$ with uniform $\mathcal{H}_0 = G'(U_n)$

$$
\begin{align*}
\mathcal{H}_0 &= G'(U_n) \\
\vdots \\
\mathcal{H}_{p(n)} &= U_{p(n)}
\end{align*}
$$

$G'(s) = \sigma_1 \cdots \sigma_{p(n)}$
The Case $p(n) = 2$

- $s_0 = s$
- $s_1 = G(s_0)_{1..n}$
- $s_2 = G(s_1)_{1..n}$

Distinguish between $G(s_0)$ and $U_{n+1}$

Distinguish between $G(s_1)$ and $U_{n+1}$
\( \mathcal{H}_0 \) vs. \( \mathcal{H}_1 \)

- Assume towards a contradiction that there exists a PPT \( \mathcal{D} \) and a non-negligible \( \epsilon(n) \) such that

\[
\left| \Pr_{\sigma_1\sigma_2 \leftarrow \mathcal{H}_0} [\mathcal{D}(\sigma_1\sigma_2) = 1] - \Pr_{\sigma_1\sigma_2 \leftarrow \mathcal{H}_1} [\mathcal{D}(\sigma_1\sigma_2) = 1] \right| \geq \epsilon(n)
\]

- We show a PPT \( \mathcal{A} \) that distinguishes \( G(s_0) \) and \( U_{n+1} \) with advantage \( \epsilon(n) \)

**The distinguisher \( \mathcal{A} \):**

- On input \( w \in \{0,1\}^{n+1} \) let
  \( s_1 = w_{1..n}, \sigma_1 = w_{n+1} \), and
  \( \sigma_2 = G(s_1)_{n+1} \)

- Output \( \mathcal{D}(\sigma_1\sigma_2) \)

\[
\Pr_{s_0 \leftarrow \{0,1\}^n} [\mathcal{A}(G(s_0)) = 1] = \Pr_{\sigma_1\sigma_2 \leftarrow \mathcal{H}_0} [\mathcal{D}(\sigma_1\sigma_2) = 1]
\]

\[
\Pr_{r \leftarrow \{0,1\}^{n+1}} [\mathcal{A}(r) = 1] = \Pr_{\sigma_1\sigma_2 \leftarrow \mathcal{H}_1} [\mathcal{D}(\sigma_1\sigma_2) = 1]
\]
Assume towards a contradiction that there exists a PPT $\mathcal{D}$ and a non-negligible $\epsilon(n)$ such that

$$\Pr_{\sigma_1\sigma_2 \leftarrow \mathcal{H}_1} [\mathcal{D}(\sigma_1\sigma_2) = 1] - \Pr_{\sigma_1\sigma_2 \leftarrow \mathcal{H}_2} [\mathcal{D}(\sigma_1\sigma_2) = 1] \geq \epsilon(n)$$

We show a PPT $\mathcal{A}$ that distinguishes $G(s_1)$ and $U_{n+1}$ with advantage $\epsilon(n)$.

The distinguisher $\mathcal{A}$:

- On input $w \in \{0,1\}^{n+1}$
- sample $\sigma_1 \leftarrow \{0,1\}$, and let $\sigma_2 = w_{n+1}$
- Output $\mathcal{D}(\sigma_1\sigma_2)$

$$\Pr_{s_1 \leftarrow \{0,1\}^n} [\mathcal{A}(G(s_1)) = 1] = \Pr_{\sigma_1\sigma_2 \leftarrow \mathcal{H}_1} [\mathcal{D}(\sigma_1\sigma_2) = 1]$$

$$\Pr_{r \leftarrow \{0,1\}^{n+1}} [\mathcal{A}(r) = 1] = \Pr_{\sigma_1\sigma_2 \leftarrow \mathcal{H}_2} [\mathcal{D}(\sigma_1\sigma_2) = 1]$$
Theorem:
Let $G$ be a PRG with expansion $2n$, then there exists a PRF $F$ mapping $n$-bit inputs to $n$-bit outputs.

The construction:
- Denote $G(s) = G_0(s)G_1(s)$ where $|G_0(s)| = |G_1(s)| = |s|$
- Define $F_k(x) = G_{x_n}(\cdots G_{x_1}(k)\cdots)$ where $x = x_1 \cdots x_n$
From PRGs to PRFs

Hybrid $\mathcal{H}_i$:
- Values on level $i$ are sampled uniformly and independently
- Values on all levels $\geq i$ are computed using $G$

Problem: Level $i$ has $2^i$ values....

Solution:
- Given $\mathcal{D}$ that runs in time $t = t(n)$, need at most $t$ values for the $i$th level
- Reduce to distinguishing $G(s_1) \cdots G(s_t)$ from $U_{2tn}$

$\mathcal{H}_0 = F_k$
$\vdots$
$\mathcal{H}_n = \text{random } f$
The World of Crypto Primitives

One-Way Functions

PRF

CPA-secure symmetric-key encryption

PRG

IND-secure symmetric-key encryption

Problem set 2
Recommended Reading


Problem set 2 is available on-line