Symmetric-Key Encryption: One-Way Functions

Lecture 6
PRG from One-Way Permutations
Story So far

PRG (i.e., a Stream Cipher) for one-time SKE
- “Mode of operation”: msg ⊕ pseudorandom pad
PRF (i.e., a Block Cipher) for full-fledged SKE
- Many standard modes of operation: OFB, CTR, CBC, ...
- All provably CPA-secure if the Block Cipher is a PRF (or PRP with trapdoor, for CBC).
  CTR mode is recommended (most efficient)
In practice, fast/complex constructions for Block Ciphers
But in principle, a PRF can be securely built from a PRG
Can build a PRG from a one-bit stretch PRG, $G_k: \{0,1\}^k \rightarrow \{0,1\}^{k+1}$

Can use part of the PRG output as a new seed

Stream cipher: the intermediate seeds are never output, can keep stretching on demand (for any “polynomial length”)
One-Way Function

- $f_k: \{0,1\}^k \rightarrow \{0,1\}^{n(k)}$ is a one-way function (OWF) if
  - $f$ is polynomial time computable
  - For all (non-uniform) PPT adversary, probability of success in the “OWF experiment” is negligible
  - Note: $x$ may not be completely hidden by $f(x)$
One-Way Function Candidates

- Integer factorization:
  - $f_{\text{mult}}(x,y) = x \cdot y$

- Input distribution: $(x,y)$ random k-bit primes

- Fact: taking input domain to be the set of all k-bit integers, with input distribution being uniform over it, will also work (if k-bit primes distribution works)

- In that case, it is important that we require $|x|=|y|=k$, not just $|x \cdot y|=2k$ (otherwise, 2 is a valid factor of $x \cdot y$ with $3/4$ probability)
One-Way Function Candidates

Solving Subset Sum:

\[ f_{\text{subsum}}(x_1...x_k, S) = (x_1...x_k, \sum_{i \in S} x_i) \]

Input distribution: \( x_i \) k-bit integers, \( S \subseteq \{1...k\} \). Uniform

Inverting \( f_{\text{subsum}} \) known to be NP-hard, but assuming that it is a OWF is “stronger” than assuming \( P \neq NP \)

Note: \((x_1,...,x_k)\) is “public” (given as part of the output to be inverted)

**OWF Collection**: A collection of subset sum problems, all with the same \((x_1,...,x_k)\) (and independent \( S \))
One-Way Function Candidates

Goldreich’s Candidate:

$$f_{\text{Goldreich}}(x, S_1, \ldots, S_n, P) = (P(x|_{S_1}), \ldots, P(x|_{S_n}), S_1, \ldots, S_n, P)$$

- $$x \in \{0,1\}^k$$, $$S_i \subseteq [k]$$ with $$|S_i| = d$$, $$P: \{0,1\}^d \to \{0,1\}$$,
- and $$x|_S$$ stands for $$x$$ restricted to indices in $$S$$

Input distribution: uniformly random with the requisite structure

**OWF Collection**: $$(S_1, \ldots, S_n, P)$$ forms the index
One-Way Function Candidates

- **Rabin OWF**: $f_{\text{Rabin}}(x; n) = (x^2 \mod n, n)$, where $n = pq$, and $p$, $q$ are random $k$-bit primes, and $x$ is uniform from $\{0...n\}$

  - **OWF collection**: indexed by $n$

- More: e.g., **Discrete Logarithm** (uses as index: a group & generator), **RSA function** (uses as index: $n=pq$ & an exponent $e$).

- Later
Hardcore Predicate

- OWFs provide no hiding property that can be readily used.
- E.g. every single bit of (random) \( x \) may be significantly predictable from \( f(x) \), even if \( f \) is a OWF
  
  [Exercise]

- Hardcore predicate associated with \( f \): a function \( B \) such that \( B(x) \) remains “completely” hidden given \( f(x) \)
Hardcore Predicates

- For candidate OWFs, often hardcore predicates known

- e.g. if $f_{\text{Rabin}}(x;n)$ is a OWF, then $\text{LSB}(x)$ is a hardcore predicate for it

- **Reduction**: Given an algorithm for finding $\text{LSB}(x)$ from $f_{\text{Rabin}}(x;n)$ for random $x$, one can use it (efficiently) to invert $f_{\text{Rabin}}$
Goldreich-Levin Predicate

Given any OWF $f$, can slightly modify it to get a OWF $g_f$ such that
- $g_f$ has a simple hardcore predicate
- $g_f$ is almost as efficient as $f$; is a permutation if $f$ is one

$g_f(x,r) = (f(x), r)$, where $|r| = |x|$

Input distribution: $x$ as for $f$, and $r$ independently random

GL-predicate: $B(x,r) = \langle x,r \rangle$ (dot product of bit vectors)

Can show that a predictor of $B(x,r)$ with non-negligible advantage can be turned into an inversion algorithm for $f$

Predictor for $B(x,r)$ is a “noisy channel” through which $x$, encoded as $(\langle x,0 \rangle, \langle x,1 \rangle, ... \langle x,2^{|x|}-1 \rangle)$ (Walsh-Hadamard code), is transmitted. Can efficiently recover $x$ by error-correction (local list decoding).
PRG from One-Way Permutations

- One-bit stretch PRG, $G_k$: $\{0,1\}^k \rightarrow \{0,1\}^{k+1}$

  \[ G(x) = f(x) \circ B(x) \]

  Where $f: \{0,1\}^k \rightarrow \{0,1\}^k$ is a one-way permutation, and $B$ a hardcore predicate for $f$

- Claim: $G$ is a PRG

  For a random $x$, $f(x)$ is also random (because permutation), and hence all of $f(x)$ is next-bit unpredictable.

  $B$ is a hardcore predicate, so $B(x)$ remains unpredictable after seeing $f(x)$
Summary

- OWF: a very simple cryptographic primitive with several candidates

- Every OWF/OWP has a hardcore predicate associated with it (Goldreich-Levin)

- PRG from a OWP and a hardcore predicate for it

  - A PRG can be constructed from a OWF too, but more complicated. (And, some candidate OWFs are anyway permutations.)

- Last time: PRF from PRG

- PRG can be used as a stream-cipher (for one-time CPA secure SKE), and a PRF can be used as a block-cipher (for full-fledged CPA secure SKE)