CS 127/CSCI E-127: Introduction to Cryptography

Prof. Salil Vadhan — Fall 2013

Lecture Notes 11:

PRG Length Expansion, Encrypting Multiple Messages

Reading.

- Katz–Lindell 2nd edition §7.4.2, 3.4 OR 1st edition §6.4.2, 3.4.

1 Length Expansion for PRGs

Last time, we saw how to construct a PRG G that stretches by one bit from any one-way permutation. How can we get a PRG with larger expansion?

First attempt: run G with many independent seeds.

Theorem 1 *Let $G : \{0,1\}^n \to \{0,1\}^{n+1}$ be a PRG. Then $G'(s_1 s_2 \cdots s_\ell) = G(s_1)G(s_2)\cdots G(s_\ell)$ is a PRG for any $\ell \leq \mathrm{poly}(n)$.*

Proof: "Hybrid technique". For $i = 0, \ldots, \ell$, define the *hybrid* $H_i = R_1 R_2 \cdots R_i G(S_{i+1}) \cdots G(S_\ell)$, where $R_j \stackrel{\mathrm{R}}{\leftarrow} \{0,1\}^{n+1}$ and $S_j \stackrel{\mathrm{R}}{\leftarrow} \{0,1\}^n$. Then $H_0 \equiv G'(U_{\ell n})$ and $H_\ell \equiv U_{\ell n+\ell}$.

Suppose that G' is not a PRG: there exists a PPT D such that:

$$\Pr\left[D(G'(U_{\ell n})) = 1\right] - \Pr\left[D(U_\ell) = 1\right] > \varepsilon$$

where ε is nonnegligible. This inequality can be rewritten using the hybrids H_i:

$$\sum_{i=0}^{\ell-1} \left(\Pr\left[D(H_i) = 1\right] - \Pr\left[D(H_{i+1}) = 1\right]\right) > \varepsilon,$$

so we have:

$$\Pr\left[D(H_I) = 1\right] - \Pr\left[D(H_{I+1}) = 1\right] > \frac{\varepsilon}{\ell},$$

where the probabilities are taken over $I \stackrel{\mathrm{R}}{\leftarrow} \{0, \ldots, \ell-1\}$, H_I or H_{I+1}, and the coin tosses of D.

Then the PPT $D'(x) = D(R_1 \cdots R_I x G(S_{I+2}) \cdots G(S_\ell))$ distinguishes $G(S_{I+1}) \equiv G(U_n)$ from $R_{I+1} \equiv U_{n+1}$ with advantage ε/ℓ. ■

Better approach: composition.

Theorem 2 *Let $G : \{0,1\}^n \to \{0,1\}^{n+1}$ be a PRG. Define $G_\ell(s_0) = b_1 b_2 \cdots b_\ell$, where $s_{i+1} b_{i+1} \stackrel{\mathrm{def}}{=} G(s_i)$ for $i = 0, \ldots, \ell-1$. Then, for any $\ell \leq \mathrm{poly}(n)$, G_ℓ is a PRG with expansion ℓ.*

Proof: Intuition: $G(s_0) = (s_1, b_1)$ looks random & independent, so $(G(s_1), b_1) = (s_2, b_2, b_1)$ looks random & independent, etc. To formalize this, we will use the hybrid technique. For $i = 0, \ldots, \ell$, define hybrid
$H_i =$

As above, if G_ℓ is not a PRG, then there exists a PPT D such such that

$$\Pr\left[D(H_I) = 1\right] - \Pr\left[D(H_{I+1}) = 1\right] > \frac{\varepsilon}{\ell},$$

where ε is nonnegligible.

To violate the pseudorandomness of G, define the PPT $D'(y)$ as follows (on input $y \in \{0,1\}^{n+1}$):

If $y \leftarrow G(U_n)$, then D is fed with $b_1 \cdots b_\ell \leftarrow H_i$.
If $y \leftarrow U_{n+1}$, then D is fed with $b_1 \cdots b_\ell \leftarrow H_{i+1}$.

Thus,

$$\Pr\left[D'(G(U_n)) = 1\right] - \Pr\left[D'(U_{n+1}) = 1\right] > \frac{\varepsilon}{\ell}$$

ε is nonnegligible and ℓ is a polynomial so $\frac{\varepsilon}{\ell}$ is nonnegligible, contradicting the assumption that G is a pseudorandom generator. ■

2 PRGs from One-Way Permutations

Last time, we saw

Theorem 3 *If there exists a one-way permutation f, then there exists a one-way permutation f' with a hardcore bit b. (Namely, $f'(x,r) = (f(x), r)$ and $b(x,r) = \langle x, r\rangle$.)*

Theorem 4 *If f is a one-way permutation with hardcore bit b, then $G(s) = f(s)b(s)$ is a pseudorandom generator.*

Combining these two theorems with Theorem 2, we get:

Theorem 5 *If one-way permutations exist, then pseudorandom generators exist (for any expansion function $\ell(n) = \mathrm{poly}(n)$).*

(In fact, it can be shown that pseudorandom generators exist if (and only if) one-way functions exist, but the construction from general one-way functions is much more complicated and beyond the scope of this course.)

If f is a one-way permutation with hardcore bit b, then the generator we obtain is

$$G(x) = b(x)b(f(x))b(f(f(x))) \cdots b(f^\ell(x)).$$

Note:

- The bits can be computed *on-line*, if we remember the current value of $s_i = f^i(s_0)$. To output a new bit, we output $b(s_i)$ and update $s_{i+1} \leftarrow f(s_i)$.
- The construction does not depend on ℓ : the stretch doesn't have to be determined in advance. (Note that the security degrades linearly with the number of bits produced, i.e. the adversary's advantage increases)
- This construction also works for families of one-way permutations.

$$G(r_1, r_2) = b_i(x)b_i(f_i(x)) \cdots b_i(f_i^{\ell}(x))$$

  where r_1 are the coin tosses used to select $i \xleftarrow{\text{R}} \mathsf{Gen}(1^n)$ and r_2 are the coin tosses to sample $x \xleftarrow{\text{R}} \mathcal{D}_i$. The proofs are similar to the proofs above with the modification that we give the key i to the adversary since it has to be able to evaluate the function f_i.

Concrete Instantiations

1. RSA:
   - Use the seed to pick a function from the family, i.e. pick random n-bit primes p, q ($N = pq$), $e \leftarrow \mathbb{Z}^*_{\phi(N)}$, $x \xleftarrow{\text{R}} \mathbb{Z}^*_N$
   - Output: $\text{lsb}(x), \text{lsb}(x^e \bmod N), \text{lsb}(x^{e^2} \bmod N), \text{lsb}(x^{e^3} \bmod N), \ldots$
2. Rabin:
   - Use the seed to choose $p \equiv q \equiv 3 \pmod 4$ (we need one-way permutations) and $x \xleftarrow{\text{R}} \mathbb{Z}^*_N$.
   - Output: $\text{lsb}(x^2 \bmod N), \text{lsb}(x^{2^2} \bmod N), \text{lsb}(x^{2^3} \bmod N), \ldots$
   - If the Factoring Assumption holds, the above construction is a pseudorandom generator.
3. Modular Exponentiation:
   - Use the seed to generate (p, g, x).
   - Output: $(\text{half}_{p-1}(x), \text{half}_{p-1}(g^x \bmod p), \text{half}_{p-1}(g^{g^x \bmod p} \bmod p), \ldots$.
4. All of the above secure if output $O(\log n)$ bits per iteration. Unproven (but conjectured) if output $n/2$ bits per iteration.

3 Putting Everything Together

Putting together everything we've done over the past few weeks, we have seen how to construct encryption schemes with indistinguishable encryptions based on simple and believable complexity assumptions such the hardness of factoring. This has been done through a series of implications:

Factoring Assumption, Discrete Log Assumption, Hardness of Random Subset Sum

$\Downarrow$

One-way Permutations

$\Downarrow$

One-way Permutations with Hardcore Predicates

$\Downarrow$

$$\text{PRGs with expansion } n+1$$
$$\Downarrow$$
$$\text{PRGs with arbitrary expansion } \ell(n) = \text{poly}(n)$$
$$\Downarrow$$
$$\text{Encryption scheme with key length } n \text{ and message length } \ell(n)$$

This is quite a remarkable achievement.

Furthermore, all of the above implications are proven using explicit reductions, which enable us to convert any adversary $\mathcal{A}$ that distinguishes encryptions of two messages into an algorithm $\mathcal{A}'$ that, say, factors large random integers $N = pq$ with nonnnegligible probability. We analyzed these reductions in the asymptotic framework, showing that if $\mathcal{A}$ is a PPT algorithm, then so is $\mathcal{A}'$. But the reductions can also be analyzed more precisely, enabling us to convert concrete hardness assumptions into concrete security guarantees for the resulting encryption scheme.

As described above, the pseudorandom generators G and encryption schemes $\mathsf{Enc}_k(m) = G(k) \oplus m$ are not very complicated or inefficient. However, in practice, people prefer to use PRGs and private-key encryption schemes that are extremely fast, and thus do not use "provable" schemes as above. We'll talk more about the constructions that are used in practice, and their relation to the theory we've seen, in a couple of lectures.