CS 127/CSCI E-127: Introduction to Cryptography

Prof. Salil Vadhan Fall 2013

Lecture Notes 6:

Stream Ciphers

Reading.

- Katz-Lindell 3.3, 3.4

1 Definition

Motivation: The one-time pad is secure if the two parties use a random key k but the length of the key shared is impractical. Instead, the two parties will share a single *short* seed k and use it to generate a long key k' that "looks random", i.e. it is *pseudorandom*.

What does it mean to "look random"? The following are some properties we expect a pseudorandom sequence to have:

- Each bit is almost unbiased.
- Fraction of 1's is $\approx 50\%$.
- Longest run of 1's is $O(\log n)$.
- Incompressible.
- ...

Linear congruential generators Let s_0 be the seed. Define $s_{i+1} = as_i + b \pmod m$ and output $(s_0, s_1, s_2, \cdots)$ These classical linear congruential generators and variants (e.g. where only parts of the s_i's are output or where a, b, m are part of the seed) have all been "broken": there exists a sophisticated algorithm which efficiently predicts the next bit of the sequence after seeing a short prefix.

We want a strong enough definition to guarantee that the stream cipher is secure (i.e. has indistinguishable encryptions).

Definition 1 $G : \{0,1\}^* \to \{0,1\}^*$ *is a* pseudorandom generator *if*

- *(efficiency) G can be computed in polynomial time (deterministically).*
- *(expansion) $|G(x)| = \ell(|x|)$ for some* expansion function *satisfying $\ell(n) > n$,*
- *(pseudorandomness) For every PPT D ("*distinguisher*"), there exists a negligible function $\varepsilon(\cdot)$ s.t.*

$$\left|\Pr\left[D(G(U_n)) = 1\right] - \Pr\left[D(U_{\ell(n)}) = 1\right]\right| \leq \varepsilon(n) \qquad \forall n,$$

where the probabilities are taken over $U_n \stackrel{R}{\leftarrow} \{0,1\}^n$, $U_{\ell(n)} \stackrel{R}{\leftarrow} \{0,1\}^{\ell(n)}$, and the coin tosses of D.

Input to G is called the seed.

This definition implies that all *efficiently testable* statistical properties of truly random sequences are satisfied by pseudorandom sequences. For example:

Proposition 2 *If G is a pseudorandom generator, then*

$$\Pr[\text{fraction of 1's in } G(U_n) \text{ is } < 1/3] \leq \text{neg}(n).$$

More generally, the outcome of any *efficient* procedure/application that uses pseudo-random bits instead of random ones *will not be affected*. This is a quite remarkable feature!

NB: we still need high-quality random bits for the seed of the pseudorandom generator, which are typically obtained by collecting and "hashing" presumably unpredictable physical data (e.g. system clock, disk movements, etc.). This can be challenging to get right in practice, and there have been a number of practical attacks on widely used cryptography implementations based on poor seeding of the random-number generator.

2 PRGS $\Longrightarrow$ Encryption

Given a pseudorandom generator G, we can obtain an encryption scheme $(\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec})$:

- The key k is the random seed for G.
- We generate pseudorandom bits to encrypt like one-time-pad: Let $k' = G(k)$ and define $\mathsf{Enc}_k(m) = k' \oplus m$

Theorem 3 *If G is a PRG, then* $(\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec})$ *has indistinguishable encryptions.*

Proof: Intuition: by pseudorandomness, the above encryption scheme is indistinguishable from using a truly random one-time pad, which we know to be perfectly secure. Formally, we should be able to convert any adversary breaking the scheme into a distinguisher for the pseudorandom generator.

■

3 Next Couple of Lectures

We have just seen how to obtain secure encryption from a pseudorandom generator. Our goal in the next few lectures will be to see how to replace the assumption that there exist pseudorandom generators with a lower-level/more tangible assumption (arguably it will be easier to believe in).

The assumption that we will rely on is the existence of functions that are "easy" to evaluate, but "hard" to invert. Such functions are called *one-way functions*. (In a previous lecture, we have already seen an example of a conjectured one-way function, namely integer multiplication. We will elaborate more on that example, and more.) Eventually, we will have established the following sequence of implications:

$$\textbf{OWF} \Longrightarrow \textbf{PRG} \Longrightarrow \textbf{ENC}.$$

The second implication ($\textbf{PRG} \Longrightarrow \textbf{ENC}$) is what we have seen today. Establishing the first implication ($\textbf{OWF} \Longrightarrow \textbf{PRG}$) is more involved and will require a couple of lectures (actually, we will prove a weaker implication – namely that one-way *permutations* imply pseudorandom generators).