EE 720: An Introduction to Number Theory and Cryptography (Spring 2018)

Lecture 6 — January 24, 2018

Instructor: Saravanan Vijayakumaran *Scribe: Saravanan Vijayakumaran*

1 Lecture Plan

- Finish proof that perfect adversarial indistinguishability is equivalent to perfect secrecy
- Motivate computational secrecy

2 Perfect adversarial indistinguishability

Lemma. *Encryption scheme* $\Pi = (\texttt{Gen}, \texttt{Enc}, \texttt{Dec})$ *is perfectly secret if and only if it is perfectly indistinguishable.*

Proof. Finish up proof of reverse direction. □

3 Computational Security

- To avoid the limitations of perfect secrecy, the weaker notion of computational secrecy is used in modern cryptography.
- Concrete guarantees of security are difficult to provide
- In the asymptotic approach, the cryptographic schemes as well as the involved parties are parametrized by an integer-valued *security parameter* n (typically the key length)
- Computational security allows two relaxations
  - Security is only guaranteed against adversaries with randomized attack algorithms with running time which is polynomial in n.
  - The adversary is allowed to succeed with *negligible* probability, i.e. the success probability is *asymptotically smaller than any inverse polynomial in* n.

Definition (Page 48 of KL)**.** *A function f from the natural numbers to the non-negative real numbers is **negligible** if for every positive polynomial p there is an N such that for all integers $n > N$ it holds that $f(n) < \frac{1}{p(n)}$.*

Examples: $2^{-n}, 2^{-\sqrt{n}}, n^{-\log n}$

- Negligible success probabilities obey certain closure properties.

Proposition (Page 49 of KL). *Let $\mathtt{negl}_1$ and $\mathtt{negl}_2$ be negligible functions. Then,*

1. *The function $\mathtt{negl}_3$ defined by $\mathtt{negl}_3(n) = \mathtt{negl}_1(n) + \mathtt{negl}_2(n)$ is negligible.*
2. *For any positive polynomial p, the function $\mathtt{negl}_4$ defined by $\mathtt{negl}_4(n) = p(n) \cdot \mathtt{negl}_1(n)$ is negligible.*

- The second part of the proposition implies that if a certain event occurs with only negligible probability in a certain experiment, then the event occurs with negligible probability even if the experiment is repeated polynomially many times.
- *General framework of any computational security definition:* A scheme is secure if for every probabilistic polynomial-time adversary $\mathcal{A}$ carrying out an attack of a formally specified type, the probability that $\mathcal{A}$ succeeds in the attack (where success is also formally specified) is negligible.
- Necessity of the relaxations
  - Exclude brute-force attackers
  - Exclude pure-guess attackers who succeed with exponentially small probability

3.1 Defining Computationally Secure Encryption

- We need to introduce the security parameter n in our syntax of private-key encryption.
- We assume $\mathcal{M} = \{0,1\}^*$.
- We allow the decryption algorithm to output an error in case it is presented with an invalid ciphertext.

Definition. *A* ***private-key encryption scheme*** *is a tuple of probabilistic polynomial-time algorithms ($\mathtt{Gen}$, $\mathtt{Enc}$, $\mathtt{Dec}$) such that:*

1. *The key-generation algorithm takes 1^n as input and gives key k, i.e. $k \leftarrow \mathtt{Gen}(1^n)$.*
2. *For $m \in \{0,1\}^*$, $c \leftarrow \mathtt{Enc}_k(m)$.*
3. *For ciphertext c, $\mathtt{Dec}_k(c) = m$ or error indicator $\perp$.*

It is required that for every n, c, k, we have $\mathtt{Dec}_k(\mathtt{Enc}_k(m)) = m$.

3.2 Indistinguishability in the presence of an eavesdropper

- We consider the ciphertext-only attack where the adversary observes a single ciphertext.
- Our definition will resemble the perfect adversarial indistinguishability definition except for two differences:
  - The experiment is parametrized by n

- We require the adversary to output equal length messages m_0, m_1. (See exercise 3.2 of KL)

- Consider the following experiment $\texttt{PrivK}^{\texttt{eav}}_{\mathcal{A},\Pi}(n)$:

  1. The adversary $\mathcal{A}$ is given 1^n and outputs a pair of arbitrary messages $m_0, m_1 \in \mathcal{M}$ with $|m_0| = |m_1|$.
  2. A key k is generated using $\texttt{Gen}$, and a uniform bit $b \in \{0,1\}$ is chosen. Ciphertext $c \leftarrow \texttt{Enc}_k(m_b)$ is computed and given to $\mathcal{A}$. This ciphertext c is called the *challenge ciphertext.*
  3. $\mathcal{A}$ outputs a bit b'.
  4. The output of the experiment is defined to be 1 if $b' = b$, and 0 otherwise. We write $\texttt{PrivK}^{\texttt{eav}}_{\mathcal{A},\Pi}(n) = 1$ if the output of the experiment is 1 and in this case we say that $\mathcal{A}$ succeeds.

Definition. *A private-key encryption scheme $\Pi = (\texttt{Gen}, \texttt{Enc}, \texttt{Dec})$ has* ***indistinguishable encryptions in the presence of an eavesdropper****, or is* ***EAV-secure****, if for all probabilistic polynomial-time adversaries $\mathcal{A}$ there is a negligible function $\texttt{negl}$ such that, for all n,*

$$\Pr\left[\texttt{PrivK}^{\texttt{eav}}_{\mathcal{A},\Pi}(n) = 1\right] \leq \frac{1}{2} + \texttt{negl}(n).$$

4 Pseudorandom Generators

- It is not known how to construct computationally secure encryption schemes without making any assumptions. We need to assume the existence of pseudorandom generators.

- A pseudorandom generator is an efficient (polynomial-time), deterministic algorithm for transforming a short, uniform bitstring called the *seed* into a longer, "uniform-looking" or "pseudorandom" output string.

- Pseudorandomness is a property of a *distribution* on strings.

- Some desirable properties of a pseudorandom generator:

  - Any bit of the output should be equal to 1 with probability close to $\frac{1}{2}$.
  - The parity of any subset of the output bits should be equal to 1 with probability close to $\frac{1}{2}$.

- A good pseudorandom generator should pass all efficient statistical tests, i.e. for any efficient statistical test or *distinguisher* D, the probability that D returns 1 given the output of the pseudorandom generator should be close to the probability that D returns 1 when given a uniform string of the same length.

Definition. *Let l be a polynomial and let G be a deterministic polynomial-time algorithm such that for any n and $s \in \{0,1\}^n$, the result $G(s)$ is a string of length $l(n)$. We say that G is a* ***pseudorandom generator*** *if the following conditions hold:*

1. ***Expansion:*** *For every n it holds that $l(n) > n$.*

2. ***Pseudorandomness:*** *For any PPT algorithm D, there is a negligible function* $\mathtt{negl}$ *such that*

$$\left|\Pr\left[D\left(G(s)\right)=1\right]-\Pr\left[D(r)=1\right]\right|\leq \mathtt{negl}(n),$$

where the first probability is taken over uniform choice of $s\in\{0,1\}^n$ and the randomness of D, and the second probability is taken over uniform choice of $r\in\{0,1\}^{l(n)}$ and the randomness of D.

We call l the ***expansion factor*** *of G.*

5 References and Additional Reading

- Sections 2.3,3.1,3.2,3.3 from Katz/Lindell