CSA E0 235: Cryptography (August 26, 2019)

Lecture 4

Instructor: Arpita Patra *Submitted by: Anirban Chakrabarti*

1 Inherent Drawbacks of perfect security

Any perfectly-secret encryption scheme must have a key space that is at least as large as the message space. The key must be at least as long as the message. All the perfectly-secret encryption schemes suffer severely due to large key length. Thus, despite the mathematical appeal of perfect security, in the practical world, we try to develop computationally secured ciphers. We stress that although we cannot obtain perfect security, this does not mean that we do away with the rigorous mathematical approach. Our proof techniques have to be still formally stated, with assumptions etc.

2 Computational security

The computational approach incorporates two relaxations of the notion of perfect security regarding 'computational power' and 'success rate' of the attacker.

- Make the adversary bounded by polynomial time.
- Allow the break with some negligible probability.

The goal is to design a cipher which cannot be broken within "reasonable time" and also with a "reasonable probability of success". Modern symmetric key encryption schemes can be broken given enough time and computation. But the amount of computation needed to break these encryption schemes would take more than many lifetimes to carry out even using the fastest available supercomputers and also the probability of success is negligible. For all practical purposes, this level of security suffices. To obtain a meaningful theory, we need to precisely define what is meant by polynomially bounded and negligible probability. There are two common approaches for quantifying the security in modern scheme. One is the asymptotic approach and the other is the concrete approach. We explain these now.

3 Asymptotic approach

In asymptotic approach, we evaluate the performance of an algorithm in terms of input size. We don't measure the actual running time. We calculate how does the time taken by an algorithm increases with the input size.

3.1 Probabilistic Polynomial time (PPT) algorithm

A probabilistic polynomial time algorithm is an algorithm that runs in polynomial time and may use randomness to produce non-deterministic results. PPT adversary uses algorithm

that runs in polynomial time of input size n.

A function $f(n) : \mathbb{Z}^+ \to \mathbb{R}$ *is polynomial in n if there exist c,d in* $\mathbb{R}^+$ *such that f(n)* $\leq n^c + d$ *for all* $n \geq 0$. *Example:* n^3

If the key size is n bit long then total number of possible key are 2^n which is exponential of n. An adversary cannot brute force all the keys by running a polynomial time algorithm.

3.2 Negligible function

Brute force is not an option for polynomially bounded adversary. Instead of brute force an adversary can guess the key and try polynomial times. If the success probability for an individual guess is a reciprocal of a polynomial function, then adversary can try a polynomial amount of guesses and succeed with high probability. In sum then, if the overall success rate is 1/poly(n) then we consider this a feasible attack and the scheme is not secured. So, we require that the success probability must be less than the reciprocal of every polynomial function.

The negligible function $f(n) : \mathbb{Z}^+ \to \mathbb{R}$ is defined as - for **every polynomial in n, p(n),** there exists **some positive integer N**, such that $f(n) < \frac{1}{p(n)}$, for all $n > N$.

Examples of negligible function: $\frac{1}{2^n}, \frac{1}{2^{n/2}}$.
Examples of non-negligible functions include $\frac{1}{2}, \frac{1}{logn}, \frac{1}{n^2}$
Below function is not negligible

$$f(n) = \begin{cases} \frac{1}{2} & \text{n is even} \\ \frac{1}{2^{n/2}} & \text{n is odd} \end{cases}$$

Proposition 1: Let p1 and p2 be polynomials in n. Then,

1. p1 + p2 is also polynomial.
2. p1 * p2 is also polynomial.

Proposition 2: Let negl1 and negl2 be negligible functions in n. Then,

1. negl1 + negl2 is a negligible function.
2. p(n). negl1 is a negligible function for any poly p(n)

.

3.3 Security Parameter

Security parameter n is a tunable parameter that tunes how difficult it is to break a cryptosystem. Usually the key size is set to n. The security parameter is publicly known as part of the scheme. The running time of an adversary, running time of an honest party

and adversary's success probability are all functions of n. Out of these three, first two are polynomial functions and last one is negligible function of n. Typically n is the size of secret-key and it is essential to choose the correct value of n. Consider a scheme where an adversary, running for n^3 time, can break a scheme with probability at most $2^{40}2^{-n}$. If the value of n is less than 40 then an adversary working for 40^3 minutes (6 weeks) can break the scheme with probability 1. So for the value of n less than 40, the scheme is not at all secured. But if we set the value of n to 500 for the adversary to run for more than 200 years to break with a probability of 2^{-460} then scheme is definitely acceptable. But we cannot set arbitrarily large n which will increase the user's time for encryption and decryption.

4 Concrete approach

The concrete approach quantifies the security of a given cryptographic scheme by bounding the maximum success probability of any adversary running for at most some specified amount of time. Let t,ε be positive constants with $t <= 1$. A scheme (t, ε) is secure if every adversary running for time at most t succeeds in breaking the scheme with probability at most ε. As an example, one might want to use a scheme with the guarantee that no adversary running for 5 years on 4GHz machine can break the scheme with probability better than 2^{-60}. From a theoretical standpoint, the concrete security approach is disadvantageous due to the following reason. A scheme (t, ε) can be secure but there is no clear answer to the question - for what ranges of t, ε should we say that a (t, ε) is a secure scheme. As one example, if it is claimed that no adversary running for 5 years can break a given scheme with probability better than ε, we still must ask what type of computing power (e.g., desktop PC, supercomputer, network of 100s of computers) does this assume? Does this take into account future advances in computing? We cannot conclude about the success probability of an adversary running for 10 years when it is given that the success probability of an adversary running for 2 years is ε.

5 Refining the definition for Computational Security

A private-key encryption scheme is a tuple of probabilistic polynomial-time algorithms (Gen; Enc; Dec) such that:

1. **The key-generation algorithm Gen(1^n)** *takes as input the security parameter* 1^n *and outputs a key k; we write this as* $k \leftarrow Gen(1^n)$*. Gen is randomized and it outputs a key k chosen according to some probability distribution.*

2. **The encryption algorithm $Enc_k(m)$** *takes as input a key k and a plaintext message* $m \in \{0,1\}^*$*, and outputs a cipher text c. Since Enc may be randomized, we write this as* $c \leftarrow Enc_k(m)$*.*

3. **The decryption algorithm $Dec_k(c)$** *takes as input a key k and a cipher text c and outputs a message m. We assume without loss of generality that Dec is deterministic, and so write this as* $m := Dec_k(c)$*.*

6 Indistinguishability Experiment

If a scheme, Π (Gen, Enc, Dec) - poly(n), is computationally secure then an adversary φ cannot succeed in breaking Π with probability that is not negligible. The adversary might determine the encrypted message with probability negligibly better than $\frac{1}{2}$. Below is an experiment between a challenger and an adversary that defines the security.

1. PPT Adversary φ is free to choose any pair of messages m_o, m_1 of same length.
2. Challenger generates the key by running Gen(1^n) and randomly selecting a bit $b \leftarrow \{0,1\}$. The cipher text $c \leftarrow Enc_k(m_b)$ is computed and given to φ.
3. φ guesses the message and outputs a bit $b^{'} \in (0,1)$.
4. The output of the experiment is defined to be 1 if $b = b^{'}$, and 0 otherwise. If $PrivK^{ind}_{\varphi,\Pi}(n) = 1$, we say that φ has succeded.

A private key encryption scheme = (Gen,Enc,Dec) has indistinguishable encryptions if for all probabilistic polynomial adversaries there exists a negligible function negl such that

$$Pr\left[PrivK^{ind}_{\varphi,\Pi}(n) = 1\right] \leq \frac{1}{2} + negl(n) \tag{1}$$

where the probability is taken over the randomness used by Adversary and the challenger. This means that an adversary cannot detect which plaintext was encrypted other than taking a random guess.

Remark The SKE, that leaks length, is not insecure. Challenger can advertise the message length ($|m_0| = |m_1|$) in ind-security.

7 An equivalent formulation

An equivalent way of formalizing the definition is to state that every adversary behaves the same way when it receives an encryption of m_0 and when it receives an encryption of m_1 (for any m_0, m_1 of the same length). Since φ outputs a single bit, behaving the same way means that it outputs 1 with almost the same probability in each case. To formalize this, define $PrivK^{ind}_{\varphi,\Pi}$(n,b) to be same as above, except that the fixed bit b is used (rather than being chosen at random). The following definition essentially states that φ cannot determine whether it is running experiment PrivKeav $PrivK^{ind}_{\varphi,\Pi}$(n,0) or experiment $PrivK^{ind}_{\varphi,\Pi}$(n,1)

$$\left|Pr\left[output(PrivK^{ind}_{\varphi,\Pi}(n,0)) = 1\right] - Pr\left[output(PrivK^{ind}_{\varphi,\Pi}(n,1)) = 1\right]\right| \leq negl(n) \tag{2}$$

The equation (1) and equation (2) are equivalent.

8 Parity prediction: An implication of PPT ind-security

8.1 Parity prediction experiment

If a scheme, Π(Gen, Enc, Dec) - poly(n), is PP secure then an adversary φ cannot succeed in predicting the parity of the plaintext after seeing the cipher text with probability not more than guessing. The adversary might determine the parity of the encrypted message with probability negligibly better than $\frac{1}{2}$. Below is an experiment between a challenger and an adversary that defines the security.

1. Challenger choose an arbitrary message m from message-space. b is the parity of the un-encrypted message.
2. Challenger generates the key by running Gen(1^n). The cipher text $c \leftarrow Enc_k(\mathrm{m})$ is computed and given to φ.
3. φ breaks the parity of the message and outputs a bit $b^{'} \in (0, 1)$.
4. The output of the experiment is defined to be 1 if $b = b^{'}$, and 0 otherwise. If $PrivK^{pp}_{\varphi,\Pi}(n) = 1$, we say that φ has succeeded.

A private key encryption scheme =(Gen,Enc,Dec) is PP secure if for all probabilistic polynomial adversaries there exists a negligible function negl such that

$$Pr\left[PrivK^{pp}_{\varphi,\Pi}(n) = 1\right] \leq \frac{1}{2} + negl(n) \tag{3}$$

where the probability is taken over the randomness used by adversary and the challenger. This means that an adversary cannot detect the parity of the plaintext from the ciphertext other than taking a random guess.

8.2 Ind-security implies pp-security

Theorem 1 *If a private-key encryption scheme, Π(Gen, Enc, Dec) - poly(n), has indistinguishable encryptions then it is also a pp-secure scheme.*

Proof We first assume a SKE scheme, Π(Gen, Enc, Dec) - poly(n), is ind-secure but not pp-secure. That means there exists an adversary A_{pp} who can break the pp-security of the scheme. If it is not pp-secure then by using reduction technique, we can prove that there exist another adversary A_{ind} who can break the ind-security of the scheme with help of A_{pp}. This contradicts our assumption that the scheme Π is ind-secure. That means if Π is ind-secure then Π is also pp-secure. ■

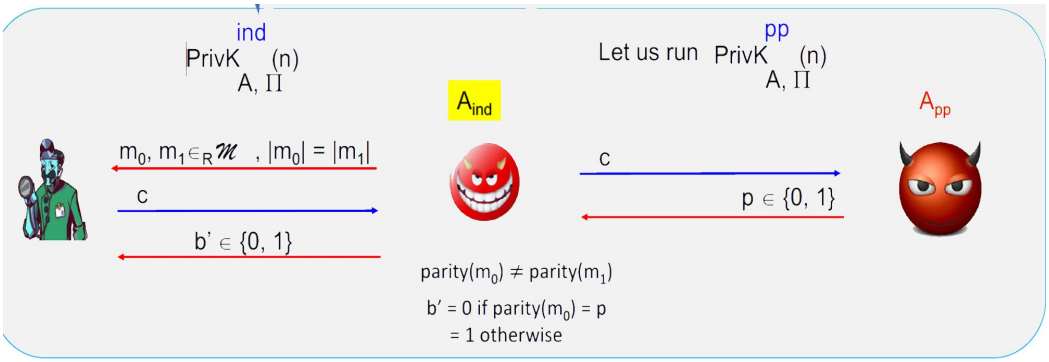
8.2.1 Reduction :

1. Adversary A_{ind} choose two messages m_0, m_1 of different parity and send it to challenger for generation of cipher. parity(m_0) ≠ parity(m_1)
2. Challenger encrypts one of the messages and sends back the cipher c to A_{ind} for identifying the message.

3. Now adversary A_{ind} passes the ciphertext c to another adversary A_{pp} who can break the parity of the message.

4. After receiving the ciphertext c, A_{pp} has come back with the parity bit p $\in$ (0,1) of the message.

5. Adversary A_{ind} calculates $b^{'} \in$ (0,1) based on the response p. A_{ind} assigns $b^{'}$ to 0 if parity(m_0) is p and 1 otherwise.



It is clear that if there is an adversary who can break pp-security then there is also an adversary who can break ind-security. It concludes ind-security implies pp-security.

9 Message recovery: An implication of PPT ind-security

If a scheme, Π(Gen, Enc, Dec) - poly(n), is mr-secure then an adversary φ cannot recover the message from cipher text. Only she can guess the message with probability $\frac{1}{|\mathscr{M}|}$ where $\mathscr{M}$ is message space. An experiment between a challenger and an adversary that defines the security is -.

1. Challenger choose an arbitrary message m from message-space $\mathscr{M}$ and generates the key by running Gen(1^n).

2. Challenger passes the cipher text $c \leftarrow Enc_k$(m) to adversary φ.

3. φ recovers the message and outputs $m^{'}$.

4. The output of the experiment is defined to be 1 if $m = m^{'}$, and 0 otherwise. If $PrivK^{mr}_{\varphi,\Pi}(n) = 1$, we say that φ has succeeded.

A private key encryption scheme =(Gen,Enc,Dec) is mr-secure if for all probabilistic polynomial adversaries there exists a negligible function negl such that

$$Pr\left[PrivK^{mr}_{\varphi,\Pi}(n) = 1\right] \leq \frac{1}{|\mathscr{M}|} + negl(n) \tag{4}$$

Where the probability is taken over the randomness used by Adversary and the challenger.
Remark If scheme Π (Gen, Enc, Dec) - poly(n), is ind-secure then it is also mr-secure.

10 Security for multiple message encryption :

An appropriate experiment that is defined for an encryption scheme,Π(Gen, Enc, Dec) - poly(n), and adversary φ, and a security parameter n:

1. The adversary φ is given input 1^n, and outputs a pair of vectors of messages $\vec{m}_0$,$\vec{m}_1$ such that each vector contains the same number of messages, and for all i it holds that $\mid m_0^i \mid = \mid m_1^i \mid$, where m_b^i denotes the i^{th} element of $\vec{m}_b$.

2. φ random key k is generated by running Gen(1^n), and a random bit b $\leftarrow$(0,1) is chosen. For all i, the ciphertext $c \leftarrow Enc_k(m_b^i)$ is computed and the vector of ciphertexts $\vec{c}$ is given to φ.

3. φ outputs a bit $b^{'}$ by predicting which vector encrypted.

4. The output of the experiment is defined to be 1 if $b^{'}$ = b, and 0 otherwise. If $PrivK_{\varphi,\Pi}^{mult}(n) = 1$, we say that φ has succeeded.

A private key encryption scheme =(Gen,Enc,Dec) is mult-secure if for all probabilistic polynomial adversaries there exists a negligible function negl such that

$$Pr\left[PrivK_{\varphi,\Pi}^{mult}(n) = 1\right] \leq \frac{1}{2} + negl(n) \tag{5}$$

Where the probability is taken over the randomness used by Adversary and the challenger.

Claim 2 *A crucial point is that security for a single encryption does not imply security under multiple encryptions. There exists private-key encryption schemes that are secure with respect to equation 1 but that are not secure with respect to equation 5.*

Any deterministic encryption scheme must be insecure for multiple message encryption. In deterministic encryption if the same message is encrypted multiple times then the same ciphertext results each time.

11 Semantic Security

The idea behind the definition of semantic security is given prior information about message, the cipher text leaks no additional information about the message. This concept is the computational complexity analogue to Shannon's concept of perfect secrecy. Perfect secrecy means that the cipher text is impossible to break, whereas semantic security implies that cipher text is infeasible to break with high probability.

Theorem 3 *A private-key encryption scheme has indistinguishable encryptions if and only if it is semantically secure.*

12 Application

Onetime-pad is a perfectly secure encryption scheme where key length should be same as message length. In computational security we can use same scheme with smaller size of key, - only padding is required when the key length is smaller than message length. In computational security, pseudorandom string is enough for PPT adversary. Whereas in perfect security truly random padding is required for shorter key. Hence shorter key for bigger message is one of the biggest advantage of computational security.

References

[1] *Jonathan Katz and Yehuda Lindell Introduction to Modern Cryptography, second edition.CRC Press, 2014.*