E0 235: Cryptography

Naman Mishra

August 2024

Contents

The course

Course webpage

MS Teams:

Lecture 1.
Tuesday
August 6

Timeline

Symmetric/Private/Shared key cryptography will be covered by Prof. Arpita Patra till 18th September, and then resume near the end of November. **Asymmetric/Public key cryptography** will be covered by Prof. Sanjit Chatterjee in the middle.

Evaluation

Both professors may have different evaluation techniques, each with a 50% weightage.

Prof. Arpita Patra

TBD

Prof. Sanjit Chatterjee

- Midterm:
- Two assignments:
- Endterm:

Chapter 1

Introduction

1.1 History

The 1980s mark a transition from classical to modern cryptography. Classical cryptography dealt only with secure communication, with ad-hoc codes and ciphers. When a code is broken, it is fixed, or another one is created. Creative, intellectually challenging, but not scientifically rigorous.

Modern cryptographic problems:

- Authenticated message transmission: An adversary tries to tamper with the message. The receiver should be able to detect tampering.
- Electronic voting and auctions: Return the highest bid without revealing the other bids. In elections, make the votes anonymous and secret till the end.
- Activism with safety: Deniable encryption
- Secure storage, secret sharing, broadcast, zero-knowledge proofs
- Secure information retrieval
- Secure outsourcing to the cloud
- **Secure computation:** The holy grail of cryptography. An abstraction of everything else.

1.2 Secure (multiparty) computation

Year	**Technique**
110 BC	Caesar Cipher
WWII	Enigma
1980s	Boom!

Table 1.1: Timeline of cryptography

Problem 1.1. There are n parties, $P_1, P_2, \ldots, P_n$, who do not trust each other. Each party P_i has a private input x_i to a common n-input function f.

The goal is to compute $f(x_1, x_2, \ldots, x_n)$ while revealing nothing about the inputs except the output.

Applications:

- Satellite collision avoidance
- Secure set intersection
- Privacy-preserving everything

Modern cryptography follows three principles:

- A *formal definition* of security capturing requirement.
- *Precise* and *well-studied* assumptions.
- Mathematical proof of security.

Examples.

- Prime factorization
- Subset sum: Given a set of integers S and a target t, is there a subset of S that sums to t?

Our primary objective will be secure communication: turning insecure channels into secure ones. We expect (i) privacy, (ii) integrity and (iii) authenticity.

Integrity is the ability to detect tampering. Authenticity is the ability to detect impersonation.

Lecture 1: Cryptography in ye olden days

1.3 Classical examples

In the private key setting, a message m is encrypted with a key k to get a ciphertext c. The ciphertext is decrypted with the same key k to get the message m. We will use the following notation:

- The key-generation algorithm Gen is a randomized algorithm that outputs a key k according to some probability distribution.
- The encryption algorithm Enc_k is parameterized by the key k, and takes a message m to output a ciphertext c. This may be deterministic or randomized.
- The decryption algorithm Dec_k is parameterized by the key k, and takes a ciphertext c to output a message m.
- The key space $\mathcal{K}$ is the set of all possible keys.
- The message space $\mathcal{M}$ is the set of all possible messages.
- The cipher-text space $\mathcal{C}$ is the set of all possible ciphertexts.

Example (Caesar cipher). The Caesar cipher has $\mathcal{M} = \mathcal{C} = \{0, 1, \ldots, 25\}$. $\mathrm{Enc}(m) = m + 3 \bmod 26$ and $\mathrm{Dec}(c) = c - 3 \bmod 26$. It is a keyless cipher.

A keyed cipher needs to be private.

> The security of a cryptographic system should not depend on the secrecy of the algorithm, but only on the secrecy of the key. – Kerckhoffs, 1883

Here are some arguments for Kerckhoffs' principle:

- Maintaining the secrecy of the algorithm is far more difficult than maintaining the secrecy of the key.
- Replacing the algorithm is infinitely harder than replacing the key.
- For cryptography to be an everyday tool, secret algorithms would need to be devised for every pair of communicating parties.

Example (Shift cipher). The obvious generalization of the Caesar cipher using a key k.

26 is an embarrassing size for a key space.

Example (Mono-alphabetic substitution). The key is a permutation of the alphabet. $\mathrm{Enc}_k = k$ and $\mathrm{Dec}_k = k^{-1}$.

The key space has size 26!. Brute force is infesible, but *frequency analysis* destroys it.

Example (Vigenère cipher). The key is a random t-tuple of shifts. The message is divided into blocks of length t, and the key is applied to each block using the shift cipher.

The key space has size 26^t, but if t is known, the key can be broken by frequency analysis. Consider all the characters at positions $i \pmod t$. These are all encrypted with the same key, so frequency analysis works. Repeat to get the entire word.

Even if we take the key to be a tuple of t permutations, the same method works.

Lecture 2. Thursday August 8

Learnings from classical SKE

- Algorithms of secure key encryption (SKE), and in all of cryptography, must be public.
- The key space must be huge.
- Definitions and proofs.

Today, we will formulate a formal definition (the threat and break model) and

Lecture 2: Security!

Chapter 2

Secret Key Encryption

Definition 2.1. A *secret key encryption scheme* is a tuple of algorithms $(\mathrm{Gen}, \mathrm{Enc}, \mathrm{Dec})$ together with three sets $\mathcal{K}, \mathcal{M}, \mathcal{C}$ called the *key space*, *message space* and *ciphertext space* where:

- $\mathrm{Gen}\colon 1 \to \mathcal{K}$ is a probabilistic algorithm;
- $\mathrm{Enc}\colon \mathcal{K} \times \mathcal{M} \to \mathcal{C}$ and $\mathrm{Dec}\colon \mathcal{K} \times \mathcal{C} \to \mathcal{M}$ are deterministic algorithms

such that

$$\forall m \in \mathcal{M}(\mathrm{Dec}_k(\mathrm{Enc}_k(m))) = m.$$

We write Enc_k and Dec_k to denote Enc and Dec partially applied to the key k.

Remark. Notice that this implies Enc_k is injective for each $k \in \mathcal{K}$.

2.1 A formal definition of security

We consider the computationally omnipotent adversary $\mathcal{R}$ (for Ravana). Suppose they have a capability to listen to all ciphertexts. This is called *ciphertext only attack* (COA).

We will also assume that $\mathcal{R}$ can use randomness. Encryption schemes often use randomness, so why not the adversary?

Thus, the adversary has the following characteristics:

- Randomized
- All-powerful
- Ciphertext only

Let us attempt to define “security”. Here are some possibilities:

	a	b	c	d
0				
1				
2				

Table 2.1: Example encryption scheme

- A scheme is secure if it does not leak the secret key. Nope, $m \mapsto m$ would be considered secure.
- A scheme is secure if it does not leak the *entire* message. Nope, revealing even a single bit is bad.
- A scheme is secure if it does not leak *any* bit. Nope. It could be that the scheme reveals the parity of the message modulo 3.
- A scheme is secure if it does not leak *any* digit in *any* base. Hmm, interesting.
- A scheme is secure if it does not leak *any* reasonable information about the message. How do we formalize this?

2.1.1 Probability review

Theorem 2.2 (Law of total probability)**.** *Let $E_1, E_2, \dots, E_n$ be a partition of the sample space. Then for any event A,*

$$\Pr(A) = \sum_{i=1}^{n} \Pr(A \mid E_i) \Pr(E_i).$$

Theorem 2.3 (Bayes' theorem)**.** *Let A and B be events with $\Pr(B) > 0$. Then*

$$\Pr(A \mid B) = \frac{\Pr(B \mid A)\Pr(A)}{\Pr(B)}.$$

2.2 Formalizing SKE

Exercise 2.4. *Consider the encryption scheme in table 2.1 with distributions*

$$K \sim \begin{pmatrix} 0 & 1 & 2 \\ 1/2 & 1/4 & 1/4 \end{pmatrix} \qquad M \sim \begin{pmatrix} a & b & c & d \\ & & & \end{pmatrix}$$

Lecture 2: Security!

Definition 2.5 (perfect security)**.**
An encryption scheme $(\mathrm{Gen}, \mathrm{Enc}, \mathrm{Dec})$ is *perfectly secure* if for every random variable M over $\mathcal{M}$ and every $m \in \mathcal{M}$, $c \in \mathcal{C}$,

$$\Pr[M = m \mid C = c] = \Pr[M = m].$$

Remark. This is probably the first formal definition of security, by Claude E. Shannon in Bell Systems Technical Journal, 28(4): 656–715, 1949.

Proposition 2.6. *The following are equivalent.*

Theorem 2.7 (Vernam cipher)**.** *Let* $\mathcal{K} = \mathcal{M} = \mathcal{C} = \{0,1\}^n$*. Let* Gen *draw uniformly at random from* $\mathcal{K}$*, and let* $\mathrm{Enc} = \mathrm{Dec} = \oplus$ *(bitwise XOR).*

Then $(\mathrm{Gen}, \mathrm{Enc}, \mathrm{Dec})$ *is perfectly secure.*

Proof. It is easy to see that $\mathrm{Dec}_k \circ \mathrm{Enc}_k = \mathrm{id}$.

Now for any $c \in \mathcal{C}, m \in \mathcal{M}$, we have

$$\Pr[C = c \mid M = m] = \Pr[K = c \oplus m] = 2^{-n}.$$

Thus

$$\Pr[C = c] = \sum_{m \in \mathcal{M}} \Pr[C = c \mid M = m] \Pr[M = m] = 2^{-n}.$$

This gives $\Pr[C = c \mid M = m] = \Pr[C = c]$ and so C and M are independent. ■

Problems with the Vernam cipher:

- Can we reuse the key for multiple messages? Nope. We can XOR two consecutive messages to get the XOR of the two messages.
- The key is as long as the message.
- Coin tossing does not scale well.

Fact 2.8. *Key length and key reusability is an issue in* any *perfectly secure encryption scheme.*

Exercise 2.9. *Prove that definition 2.5 is equivalent to the following:*

$$\Pr[C = c \mid M = m] = \Pr[C = c \mid M = m'] \quad \forall\, m, m' \in \mathcal{M}, c \in \mathcal{C}.$$

Solution. Suppose definition 2.5 holds. Then

$$\Pr[C = c \mid M = m] = \frac{\Pr[M = m \mid C = c] \Pr[C = c]}{\Pr[M = m]} = \Pr[C = c]$$

Thus the given condition holds. ■

Announcement: Class on Friday from 9:30–11:00 am.

Lecture 3.
Tuesday
August 13

Theorem 2.10 (Shannon's theorem)**.** *A scheme* $(\mathrm{Gen}, \mathrm{Enc}, \mathrm{Dec})$ *with* $|\mathcal{K}| = |\mathcal{C}| = |\mathcal{M}|$ *is perfectly secure if and only if*

(1) Gen *is a uniform distribution over* $\mathcal{K}$*.*

(2) For every $m \in \mathcal{M}$ *and* $c \in \mathcal{C}$*, there is a unique key* $k \in \mathcal{K}$ *such that* $\mathrm{Enc}_k(m) = c$*.*

Why do we care about $|\mathcal{K}| = |\mathcal{C}| = |\mathcal{M}|$? We will see $|\mathcal{K}| \geq |\mathcal{M}|$ is necessary for security. Equality is the optimal case.

Since Enc_k needs to be injective for each $k \in \mathcal{K}$, $|\mathcal{M}| \leq |\mathcal{C}|$. Equality is the optimal case.

Proof. Suppose $(\mathrm{Gen}, \mathrm{Enc}, \mathrm{Dec})$ satisfies the conditions. We need to show $\Pr[C = c \mid M = m] = \Pr[C = c \mid M = m']$. Since there is a unique key for which $\mathrm{Enc}_k(m) = c$, $\Pr[C = c \mid M = m] = \Pr[K = k] = 1/|\mathcal{K}|$. This is independent of M, so the two probabilities are equal.

Now suppose $(\mathrm{Gen}, \mathrm{Enc}, \mathrm{Dec})$ is perfectly secure. Fix a $c \in \mathcal{C}$. Let $\mathcal{K}_m = \{k \in \mathcal{K} \mid \mathrm{Enc}_k(m) = c\}$.

Claim. $\mathcal{K}_m \neq \varnothing$ *and* $\mathcal{K}_m \cap \mathcal{K}_{m'} = \varnothing$ *for* $m \neq m'$*.*

Suppose there is some message m_0 for which $\Pr[C = c \mid M = m_0] > 0$. By perfect security, $\Pr[C = c \mid M = m] > 0$ for each $m \in \mathcal{M}$. If $\mathcal{K}_m = \varnothing$ for some m, then $\Pr[C = c \mid M = m] = 0$.

Since Enc_k is injective for each $k \in \mathcal{K}$, the same key cannot encrypt two different messages to the same ciphertext. Thus $\mathcal{K}_m \cap \mathcal{K}_{m'} = \varnothing$ for $m \neq m'$.

Since

$$\Pr[K = k_m] = \Pr[C = c \mid M = m] = \Pr[C = c \mid M = m'] = \Pr[K = k_{m'}],$$

we have $\Pr[K = k_m] = 1/|\mathcal{K}|$. ■

We now have three equivalent definitions of perfect security: definition 2.5, exercise 2.9, and theorem 2.10.

Chapter 3

Secret sharing

Lecture 4. Friday August 16

Definition 3.1 (Secret sharing)**.** A *secret-sharing scheme* for an *access structure* Γ over $\mathcal{P} = \{P_1, \ldots, P_n\}$ is a pair of public algorithms

$$\mathrm{Sh}\colon \mathcal{K} \to \mathcal{S}^n \quad \text{and} \quad \mathrm{Rec}\colon$$

such that for all $s \in \mathcal{K}$ with $\mathrm{Sh}(s) = (s_1, \ldots, s_n)$ and $B = \{P_{i_1}, \ldots, P_{i_k}\} \subseteq \mathcal{P}$,

(1) if $B \in \Gamma$, then $\mathrm{Rec}(B, s_{i_1}, \ldots, s_{i_k}) = s$.

(2) if $B \notin \Gamma$, then $\mathrm{Rec}(B, s_{i_1}, \ldots, s_{i_k}) = \perp$.

Lecture 7. Thursday August 22

Security parameter n determines how secure the scheme is.

The parties and the adversary are assumed to be computationally bounded by a polynomial in n.

We want the probability of breaking the scheme to be *negligible* in n.

3.1 Choosing the parameter

Suppose there is a sheme which an adversary can break with probability $2^{40}2^{-n}$ after running for n^3 minutes.

- $n \leq 40$ is dumb. Within 2^{40} minutes, approximatly 6 weeks, the adversary can break the scheme with probability 1.
- $n = 50$ sounds nice, but 2^{10} is only 1024. An adversary can break the scheme with probability 1/1000 within 6 weeks.
- $n = 128$ works.

Finally, we are allowed to leak the *length* of the message. We will say that an SKE that leaks the length of the message is secure. Why?

Exercise 3.2. *Show that there is no scheme which realises the above definition of security without leaking the length of the message.*

Definition 3.3 (Parity secure)**.** The experiment $\mathrm{PrivK}^{pp}_{A,\Pi}(n)$ returns 1 if the adversary A can predict the parity of the message, and 0 otherwise. We say that Π is pp-secure if for every PPT adversary A,

$$\Pr[\mathrm{PrivK}^{pp}_{A,\Pi}(n) = 1] \leq \frac{1}{2} + \mathrm{negl}(n).$$

3.2 Proofs by reduction

Proofs from this point on will be highly conditional.

- If A holds then Π is x-secure.
- If Π is x-secure then A holds.
- If A_1 holds then A_2 holds.
- If Π that is x-secure, then Π' is y-secure.

We keep the last one in mind as we try to look at general techniques to prove such statements.

- **Proof by contradiction/contrapositive:** Suppose Π' is not y-secure. Show that Π is not x-secure.

  That is, there exists a PPT adversary A' that breaks Π' with non-negligible probability $f(n)$. We need to construct a PPT adversary A that breaks Π with non-negligible probability $g(n)$. This is *reduction* of breaking Π to breaking Π'.

  Given a challenge for Π, we have to simulate a challenge for Π' (in polynomial time). Using the broken challenge for Π' (with substantial probability), we need to break the challenge for Π (with substantial probability).

  The product of two substantial probabilities is substantial.

Proposition 3.4. *If* Π *is ind-secure, then* Π *is pp-secure.*

Proof. Suppose Π is not pp-secure. Then there exists a PPT adversary A that predicts the parity of the message with probability $1/2 + f(n)$, where $f(n)$ is non-negligible.

That is, there is some polynomial $p(n)$ such that $f(n) \geq 1/p(n)$ for infinitely many n. To summarize,

$$\Pr[\mathrm{PrivK}^{pp}_{A,\Pi}(n) = 1] \geq \frac{1}{2} + \frac{1}{p(n)}.$$

Lecture 7

Let us construct a PPT adversary A' that breaks the ind-security of Π. Select two messages m_0 and m_1 of the same length, but with $\text{parity}(m_0) = 0$ and $\text{parity}(m_1) = 1$. Pass the challenge to A. Let p be the reply from A. Output p. Then

$$\text{PrivK}^{pp}_{A,\Pi}(n) = 1 \iff \text{PrivK}^{ind}_{A',\Pi}(n) = 1.$$

Thus

$$\Pr[\text{PrivK}^{ind}_{A',\Pi}(n) = 1] = \Pr[\text{PrivK}^{pp}_{A,\Pi}(n) = 1] \geq \frac{1}{2} + \frac{1}{p(n)}$$

is substantial. ■

Quiz: Thursday, 29th

3.3 Shamir Sharing

A polynomial of decree t

$$f(x) = s + a_1x + a_2x^2 + \cdots + a_tx^t.$$

where s is the secret. Any t pairs $(x, f(x))$ fail to determine the polynomial. Any $t+1$ pairs $(x, f(x))$ determine the polynomial.

Consider a new secret sharing scheme as follows:

Problem 0.1 (replicative secret sharing)**.** $\mathcal{P} = \{P_1, \ldots, P_n\}$. *For every subset $T \subseteq \mathcal{P}$ of size $n - t$, there is a share S_T held by all $P_i \in T$. The secret is defined as $S = \sum_T S_T$.*

(1) Is this scheme secure if an adversary is allowed to corrupt t of the parties?

Lecture 9.
Thursday
September 5

Recap:

- Pseudorandomness and pseudorandom generators
- An ind-secure secret key encryption and proof

Today, we will try to reduce our key size, and see if a key can be reused.

3.4 Multi-message security

We will prove that multi-message ind-security is stronger than single-message ind-security.

The multi-message game is as follows:

Definition 3.5. An adversary $\mathcal{A}$ picks a vector M_0 of messages $(m_0^{(1)}, \ldots, m_0^{(q)})$ and a vector M_1 of messages $(m_1^{(1)}, \ldots, m_1^{(q)})$ of the same length, where each $m_b^{(i)}$ has the same length. The challenger chooses a $b \in \{0, 1\}$ uniformly at random, encrypts each message in M_b using the same key k under the scheme Π and gives the resulting ciphertexts to $\mathcal{A}$. $\mathcal{A}$ outputs a guess b'.

Π is multi-ind-secure if for all PPT adversaries $\mathcal{A}$,

$$\Pr[b = b'] \leq \frac{1}{2} + \operatorname{negl}(n),$$

Proposition 3.6. *Multi-ind-security is* strictly *stronger than ind-security.*

Proof. We know that the pseudo one-time pad is ind-secure. However, the encryption algorithm is deterministic, so it fails multi-ind-security.

Construct an adversary $\mathcal{A}$ that picks $M_0 = (a, a)$ and $M_1 = (a, b)$ where $a \neq b \in \mathcal{M}$. $\mathcal{A}$ will output b' as 0 if the two ciphertexts are the same, and 1 otherwise. Then

$$\Pr[b = b'] = 1.$$

■

3.5 Onion routing

3.6 PRG with poly expansion factor

Theorem 3.7. *If there exists a PRG with expansion factor $\ell(n) = n + 1$, then for each polynomial $p(n)$, there exists a PRG with expansion factor $\ell'(n) = n + p(n)$.*

Proof. Let $G\colon 2^n \to 2^{n+1}$ be a PRG. For any $s \in 2^n$, let $G(s) = (I(s), L(s))$ where $I(s) \in 2^n$ and $L(s) \in 2$.

Starting with a seed $s \in 2^n$,

$$\begin{aligned} s_1 &= (Is, Ls), \\ s_2 &= (IIs, LIs, Ls), \\ s_3 &= (IIIs, LIIs, LIs, Ls), \end{aligned}$$

and so on, where we apply G to the first n bits of the previous output. ■