Cryptography and Embedded System Security
CRAESS_I

Xiaolu Hou

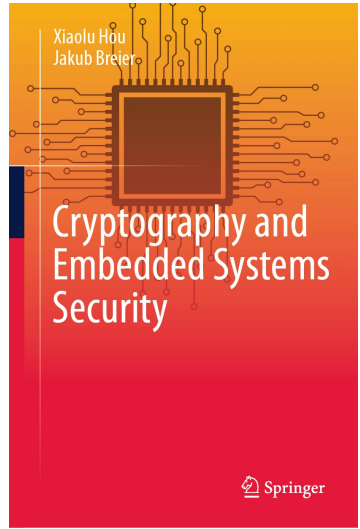
FIIT, STU
xiaolu.hou @ stuba.sk

Course Outline

- Abstract algebra and number theory
- Introduction to cryptography
- Symmetric block ciphers and their implementations
- RSA, RSA signatures, and their implementations
- Probability theory and introduction to SCA
- SPA and non-profiled DPA
- Profiled DPA
- SCA countermeasures
- FA on RSA and countermeasures
- FA on symmetric block ciphers
- FA countermeasures for symmetric block cipher
- Practical aspects of physical attacks
  - Invited speaker: Dr. Jakub Breier, Senior security manager, TTControl GmbH

Recommended reading

- Textbook
  - Sections
    - 2.1;
    - 2.2.1, 2.2.2, 2.2.6, 2.2.7;
    - 2.3.



Lecture Outline

- Cryptography
- Cryptographic Primitives
- Cryptosystems
- Classical Ciphers
- Encryption Modes
- Some exercises

Introduction to cryptography

- Cryptography
- Cryptographic Primitives
- Cryptosystems
- Classical Ciphers
- Encryption Modes
- Some exercises

Definition

Definition

Cryptography studies techniques that allow secure communication in the presence of adversarial behavior. These techniques are related to information security attributes such as *confidentiality*, *integrity*, *authentication*, and *non-repudiation*.

Next, we look at information security attributes that can be achieved by using cryptography

Confidentiality

- *Confidentiality* aims at preventing unauthorized disclosure of information.
- There are various technical, administrative, physical, and legal means to enforce confidentiality.
- In the context of cryptography, we are mostly interested in utilizing various encryption techniques to keep information private.

Confidentiality

Eve

Alice

Eavesdropping

Bob

Confidentiality aims at preventing unauthorized disclosure of information.

Integrity

- *Integrity* aims at preventing unauthorized alteration of data to keep them correct, authentic, and reliable.
- Similarly to confidentiality, while there are many means of ensuring data integrity, in cryptography we are looking at hash functions and message authentication codes.

Integrity

Eve

Alice

Trying to change the message

Bob

Integrity aims at preventing unauthorized alteration of data to keep them correct, authentic, and reliable.

Authentication

- *Authentication* aims at determining whether something or someone is who they claim they are.
- In communication, the entities should be able to identify each other.
- Similarly, the properties of the exchanged information, such as origin, content, and timestamp, should be authenticated.
- In cryptography, we are mostly interested in two aspects: entity authentication and data origin authentication.
- For these purposes, signatures, and identification primitives are used.

Authentication

Eve

Alice

Eve is evil

Bob

It is not me!

Authentication aims at determining whether something or someone is who they claim they are.

Non-repudiation

- *Non-repudiation* aims at assuring that the sender of the information is provided with proof of delivery, and the recipient is provided with proof of the sender's identity so that neither party can later deny the actions taken.
- Similarly to authentication, signatures, and identification primitives are cryptographic means of supporting non-repudiation.

Remark

- CIA Triad is a widely utilized information security model, where the abbreviation stands for confidentiality, integrity, and availability.
- Why we did not mention the availability?
- The answer is rather simple – there are no techniques within cryptography that could contribute in one way or another to ensure availability.
- Availability attribute ensures that information is consistently and readily accessible for authorized entities.
- One needs to look into other means of supporting this attribute.

Introduction to cryptography

- Cryptography
- Cryptographic Primitives
- Cryptosystems
- Classical Ciphers
- Encryption Modes
- Some exercises

Categorization of cryptographic primitives

Cryptographic primitives are the tools that can be used to achieve the goals listed above

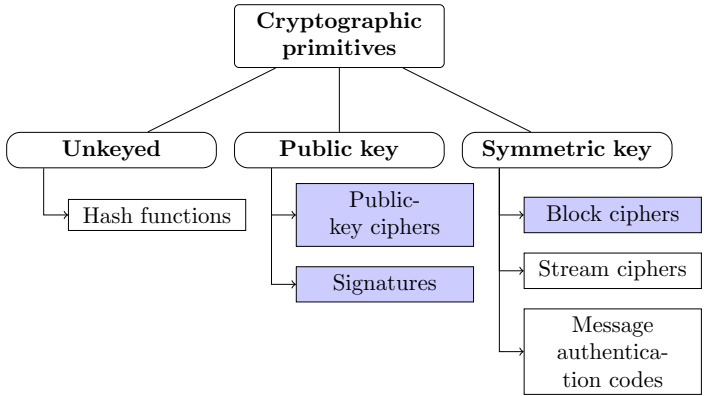


Figure: Categorization of cryptographic primitives. The ones highlighted in blue color will be discussed in this course.

Hash functions

- Hash functions map data of arbitrary length to a binary array of some fixed length called *hash values* or *message digests*
- The following are the properties that should be met in a properly designed cryptographic hash function:
  (a) it is quick to compute a hash-value for any given input;
  (b) it is computationally infeasible to generate an input that yields a given hash value (a preimage);
  (c) it is computationally infeasible to find a second input that maps to the same hash value when one input is already known (a second preimage);
  (d) it is computationally infeasible to find any pair of different messages that produce the same hash value (a collision).

Hash functions

- Cryptographic hash functions are mostly used for integrity and digital signatures.
- Message integrity use case of hash functions works as follows.
  - The user creates a message digest of the original message at some point in time.
  - At a later time (e.g., after a transmission), the digest is calculated again to check whether there have been any changes to the original message.
- In digital signatures, it is common to first create a message digest that is afterwards digitally signed, rather than signing the entire message which can be slow in case the message is large.

Public-key ciphers

- Public-key (or asymmetric) ciphers use a pair of related keys.
- This pair consists of a *private* key and a *public* key.
- These keys are generated by cryptographic algorithms that are based on mathematical problems called *one-way functions*.
- A one-way function is a function that is easy to compute on every input, but it is hard to compute its inverse[1].

[1] It is worth noting that the existence of one-way functions is an open conjecture and depends on $P \neq NP$ inequality.

Signatures

- Digital signatures provide means for an entity to bind its identity to a message.
- This normally means that the sender uses their private key to sign the (hashed) message.
- Whoever has access to the public key can then verify the origin of the message.

Symmetric block ciphers

- Block ciphers are cryptographic algorithms operating on blocks of data of a fixed size (generally multiples of bytes for modern cipher designs).
- They use the same secret key for the encryption and decryption of data.
- One of the main focus of this course
- We will see three block ciphers: DES, AES, PRESENT

Stream ciphers

- Stream ciphers are symmetric key ciphers that combine plaintext digits (usually bits) with the *keystream*, which is a stream of pseudo-random digits generated by the cipher.
- The combination is normally done by a bitwise XOR operation.
- The idea of stream ciphers comes from the one-time pad.

Message authentication codes (MACs)

- A message authentication code is a piece of information that is used to authenticate the origin of the message and to protect its integrity.
- MAC algorithms are commonly constructed from other cryptographic primitives, such as hash functions and block ciphers.

Introduction to cryptography

- Cryptography
- Cryptographic Primitives
- Cryptosystems
- Classical Ciphers
- Encryption Modes
- Some exercises

Insecure communication

- We have mentioned three types of ciphers: public-key ciphers, block ciphers, and stream ciphers.
- Ciphers are also called cryptosystems.
- When we use ciphers, we normally assume insecure communication.
- A popular example setting is that Alice would like to send messages to Bob but Eve is also listening to the communication.

Physical attacks on cryptographic primitives

We will be focusing on physical attacks on

- Symmetric block ciphers
- Public key ciphers

Physical attacks also apply to other cryptographic primitives

- Hemme, Ludger, and Lars Hoffmann. "Differential fault analysis on the SHA1 compression function." 2011 Workshop on Fault Diagnosis and Tolerance in Cryptography. IEEE, 2011.
- Hao, Ronglin, et al. "Algebraic fault attack on the SHA-256 compression function." International Journal of Research in Computer Science 4.2 (2014): 1-9.
- Kahri, Fatma, et al. "Fault Attacks Resistant Architecture for KECCAK Hash Function." Journal of Advanced Computer Science and Applications 5.8 (2017): 237-243.
- ...

Insecure communication channel

Eve

Alice

Bob

Encryption and decryption

- When we use ciphers, we normally assume insecure communication.
- A popular example setting is that Alice would like to send messages to Bob but Eve is also listening to the communication.
- The goal of Alice is to make sure that even if Eve can intercept what was sent, she will not be able to find the original message.
- To do so, Alice will first *encrypt* the message, or the *plaintext*, and send the *ciphertext* to Bob, instead of the original message.
- Bob will then *decrypt* the ciphertext to get the plaintext.
- For this communication to work, there must be a *key* for encryption and decryption.
- It is clear that the decryption key should be secret from Eve
- Also, a basic requirement is that the algorithm for *encryption/decryption* should be designed in a way that Eve cannot easily brute force the plaintext with the knowledge of the ciphertext.

Cryptosystem

Definition

A *cryptosystem* is a tuple $(\mathcal{P}, \mathcal{C}, \mathcal{K}, \mathcal{E}, \mathcal{D})$ with the following properties.

- $\mathcal{P}$ is a finite set of plaintexts, called *plaintext space*.
- $\mathcal{C}$ is a finite set of ciphertexts, called *ciphertext space*.
- $\mathcal{K}$ is a finite set of keys, called *key space*.
- $\mathcal{E} = \{ E_k : k \in \mathcal{K} \}$, where $E_k : \mathcal{P} \to \mathcal{C}$ is an *encryption function*.
- $\mathcal{D} = \{ D_k : k \in \mathcal{K} \}$, where $D_k : \mathcal{C} \to \mathcal{P}$ is a *decryption function*.
- For each $e \in \mathcal{K}$, there exists $d \in \mathcal{K}$ such that $D_d(E_e(p)) = p$ for all $p \in \mathcal{P}$.

If $e = d$, the cryptosystem is called a *symmetric key cryptosystem*. Otherwise, it is called a *public-key/asymmetric cryptosystem*.

Cryptosystem

- Take any $c_1 = E_e(p_1), c_2 = E_e(p_2)$ from the ciphertext space $\mathcal{C}$, where $e \in \mathcal{K}$.
- Let $d \in \mathcal{K}$ be the corresponding decryption key for e.
- If $c_1 = c_2$, then by definition,

$$p_1 = D_d(c_1) = D_d(c_2) = p_2.$$

  Thus, E_e is an injective function.
- We also note that if $\mathcal{P} = \mathcal{C}$, then E_k is a permutation of $\mathcal{P}$ – a bijective function $\mathcal{P} \to \mathcal{P}$.
- There are mainly two types of symmetric ciphers: *block ciphers* and *stream ciphers*.

Block ciphers

Definition (Block cipher)

A *block cipher* is a symmetric key cryptosystem with $\mathcal{P} = \mathcal{C} = \mathcal{A}^n$ for some alphabet $\mathcal{A}$ and positive integer n. n is called the *block length*.

- For classical ciphers that we will see today, $\mathcal{A} = \mathbb{Z}_{26}$.
- For modern cryptosystems that we will discuss next week, $\mathcal{A} = \mathbb{F}_2 = \{ 0, 1 \}$.

Block ciphers

- If we have a long plaintext $\boldsymbol{p} = p_1 p_2 \ldots$, where each $p_i \in \mathcal{A}^n$ is one block of plaintext, and a key k, using a block cipher, we can obtain ciphertext string $\boldsymbol{c}$ as follows:

$$\boldsymbol{c} = c_1 c_2 \cdots = E_k(p_1) E_k(p_2) \ldots .$$

- Such an encryption mode is called an ECB mode, more encryption modes will be introduced at the end of this lecture

Stream ciphers

- For a *stream cipher*, $\mathcal{P} = \mathcal{C} = \mathcal{A}$ are single digits.
- Encryptions are computed on each digit of the plaintext.
- In particular, suppose we have a plaintext string $\boldsymbol{p} = p_1 p_2 \dots$ (where $p_i \in \mathcal{A}$) and a key k.
- We first compute a key stream $\boldsymbol{z} = z_1 z_2 \dots$ using the key k, then the ciphertext is obtained as follows:

$$\boldsymbol{c} = c_1 c_2 \cdots = E_{z_1}(p_1) E_{z_2}(p_2) \dots$$

- A stream cipher is said to be *synchronous* if the key stream only depends on the chosen key k but not on the encrypted plaintext.
- In this case, the sender and the receiver can both compute the keystream synchronously.
- We will see a classical synchronous stream cipher called *one-time pad*.

Converting message to plaintext

- An important aspect to clarify is how the message that Alice intends to send is represented as plaintext.
- For classical ciphers, which we will discuss in a while, we will only consider messages consisting of English letters (A – Z), and we map each letter to an element in $\mathbb{Z}_{26}$.
- The plaintext spaces are vector spaces over $\mathbb{Z}_{26}$.

A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T
0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19
								U	V	W	X	Y	Z						
								20	21	22	23	24	25						

Table: Converting English letters to elements in $\mathbb{Z}_{26}$.

Converting message to plaintext – modern cipher

- In modern computers, we store data in binary digits (bits).
- An $8-$bit binary string is called a byte
- Computers often operate on a few bytes at a time.
- For example, a 64-bit processor operates on eight bytes at a time.
- In computer architecture, a *word* is defined as the unit of data of (at most) a certain bit length that can be addressed and moved between storage and the processor.
  - For a 64-bit processor, the word length is 64 bits.

Converting message to plaintext – modern cipher

- We have discussed that a byte can be represented as a decimal number between 0 and 255 or as a hexadecimal number between 00_{16} and $\mathtt{FF}_{16}$.
- When modern cryptographic algorithms are used, the messages are converted to plaintexts which are n-bit binary strings (i.e. vectors in $\mathbb{F}_2^n$), where n is a multiple of 8.

A	01000001	41
B	01000010	42
a	01100001	61
b	01100010	62
?	00111111	3F

(a) ASCII

Á	1100001110000001	C381
Ä	1100001110000100	C384
Í	1100001110001101	C38D
×	1100001110010111	C397
÷	1100001110110111	C3B7

(b) UTF-8

Table: Examples of methods for converting message symbols to bytes. The second column in each table is the binary representation of the byte value and the third column is the corresponding hexadecimal representation.

Kerckhoffs' principle

When the security of a cryptosystem is analyzed, *Kerckhoffs' principle* is always followed.

Definition (Kerckhoffs' principle)

The security of a cryptosystem should depend only on the secrecy of the key.

In other words, everything is public knowledge except for the secret key.

Attack assumptions

- To discuss the security of cryptosystems, we should also specify the attack assumptions.
- Normally, they consist of the attacker's knowledge and the attacker's goal.
- *Ciphertext-only attack*: the attacker has access to a collection of ciphertexts.
- *Known plaintext attack*: the attacker has a collection of plaintext and ciphertext pairs.
- *Chosen plaintext attack*: the attacker has access to the encryption mechanism such that they can choose plaintexts and obtain the corresponding ciphertexts.
- The attacker's goal can be the recovery of the plaintext or the recovery of the key.
- By Kerckhoffs' principle, we assume the attacker has the knowledge of the cipher design and communication context, e.g. the sender is a student and might use words like "exam," "assignment," etc.

Ciphertext-only attack

- Most realistic one
- Weakest attacker model
- An intercepted encrypted network traffic

Known plaintext attack

- Cryptanalysis of Enigma during World War II.
- There were situations when the German military broadcast the same message encrypted by different cryptosystems – for some recipients, it was encrypted by a so-called *dockyard cipher* (a manual cipher, relatively easy to cryptanalyze), and for some, it was encrypted by Enigma.
- If both messages were intercepted, the allies would possess both the plaintext and the ciphertext, thus making it a known plaintext attack on Enigma.

Enigma

- Used during WWII by German armies
- Broken by Polish mathematicians and Alan Turing
- It is believed that breaking Enigma shortened the war by two years

src: https://en.wikipedia.org/

Chosen plaintext attack

- An encryption device is captured and the attacker can send queries to it and receive the ciphertexts.
- As the key would normally be stored in secure storage, the attacker needs to use the plaintext-ciphertext pairs to recover it.

Security of a cipher

- In this course, we say a cipher is *broken* if the secret key is recovered.
  - In a more general sense, breaking a cipher means finding a weakness in the cipher algorithm that can be exploited with a complexity less than brute-force.
- *Perfectly secure*: in a ciphertext-only attack setting, the attacker cannot obtain any information about the plaintext no matter how much computing power they have.
- *Secure in practice*: no known feasible attack exists.
- *Computationally secure*: no efficient (polynomial-time) adversary can break it except with negligible probability.
- Modern cryptosystems that are popular today are considered to be computationally secure.
- Most of the ciphers are designed in a way that the effort taken to break them grows exponentially with the number of bits of the secret key, which is called *key length*.
- Key length is an important factor in the security of modern ciphers.

Introduction to cryptography

- Cryptography
- Cryptographic Primitives
- Cryptosystems
- Classical Ciphers
- Encryption Modes
- Some exercises

Congruence class

Definition

For any $a \in \mathbb{Z}$, the *congruence class of a modulo n*, denoted $\overline{a}$, is given by

$$\overline{a} := \{\, b \mid b \in \mathbb{Z}, b \equiv a \mod n \,\}.$$

Lemma

Let $\mathbb{Z}_n$ denote the set of all congruence classes of $a \in \mathbb{Z}$ modulo n. Then $\mathbb{Z}_n = \{\, \overline{0}, \overline{1}, \ldots, \overline{n-1} \,\}$.

Example

Let $n = 5$. We have $\overline{1} = \overline{6} = \overline{-4}$. $\mathbb{Z}_5 = \{\, \overline{0}, \overline{1}, \overline{2}, \overline{3}, \overline{4} \,\}$.

Addition and multiplication in $\mathbb{Z}_n$

Define addition on the set $\mathbb{Z}_n$ as follows:

$$\overline{a} + \overline{b} = \overline{a + b}.$$

Example

- Let $n = 7$, $\overline{3} + \overline{2} = \overline{5}$.
- Let $n = 4$, $\overline{2} + \overline{2} = \overline{4} = \overline{0}$.

Define multiplication on $\mathbb{Z}_n$ as follows

$$\overline{a} \cdot \overline{b} = \overline{ab}.$$

Example

Let $n = 5$,

$$\overline{-2} \cdot \overline{13} = \overline{3} \cdot \overline{3} = \overline{9} = \overline{4}$$

$\mathbb{Z}_n$

Theorem

$(\mathbb{Z}_n, +, \cdot)$, the set $\mathbb{Z}_n$ together with addition multiplication defined just now is a commutative ring.

Remark

For simplicity, we write a instead of $\overline{a}$ and to make sure there is no confusion we would first say $a \in \mathbb{Z}_n$. In particular, $\mathbb{Z}_n = \{ 0, 1, 2, \ldots, n-1 \}$. Furthermore, to emphasize that multiplication or addition is done in $\mathbb{Z}_n$, we write $ab \mod n$ or $a + b \mod n$.

Example

Let $n = 5$, we write

$$4 \times 2 \mod 5 = 8 \mod 5 = 3, \text{ or } 4 \times 2 \equiv 8 \equiv 3 \mod 5.$$

Converting message to plaintext

- We focus on the case when messages consist of English letters.
- Those letters are identified with elements in $\mathbb{Z}_{26}$.
- For simplicity, we will not distinguish letters and elements in $\mathbb{Z}_{26}$.
  - For example, when the message is `A`, we may say that the plaintext is `A` or the plaintext is 0, similarly for ciphertext.

A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T
0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19
								U	V	W	X	Y	Z						
								20	21	22	23	24	25						

Table: Converting English letters to elements in $\mathbb{Z}_{26}$.

Shift cipher

Definition (Shift cipher)

Let $\mathcal{P} = \mathcal{C} = \mathcal{K} = \mathbb{Z}_{26}$. For each $k \in \mathcal{K}$, define

$$E_k : \mathbb{Z}_{26} \to \mathbb{Z}_{26}, \quad p \mapsto p + k \mod 26; \quad D_k : \mathbb{Z}_{26} \to \mathbb{Z}_{26}, \quad c \mapsto c - k \mod 26.$$

The cryptosystem $(\mathcal{P}, \mathcal{C}, \mathcal{K}, \mathcal{E}, \mathcal{D})$, where $\mathcal{E} = \{ E_k : k \in \mathcal{K} \}$, and $\mathcal{D} = \{ D_k : k \in \mathcal{K} \}$, is called the *shift cipher*.

- We have seen that $\mathbb{Z}_{26}$ is a commutative ring with addition and multiplication modulo 26.
- Subtracting k corresponds to adding the additive inverse of k

Shift cipher – Example

Shift cipher

For each $k \in \mathcal{K}$,

$$E_k : \mathbb{Z}_{26} \to \mathbb{Z}_{26}, \quad p \mapsto p + k \mod 26; \quad D_k : \mathbb{Z}_{26} \to \mathbb{Z}_{26}, \quad c \mapsto c - k \mod 26.$$

Example

Let $k = 2$, we have

$$-k = -2 \mod 26 = 24 \mod 26.$$

Suppose the message is `A`, then the corresponding plaintext is 0. The ciphertext is given by

$$E_k(\texttt{A}) = 0 + 2 \mod 26 = 2 \mod 26 = \texttt{C}.$$

When we decrypt the ciphertext using the same key, we get our original message:

$$D_k(\texttt{C}) = 2 - 2 = 2 + 24 \mod 26 = 0 \mod 26 = \texttt{A}.$$

Shift cipher – Example

Shift cipher

For each $k \in \mathcal{K}$,

$$E_k : \mathbb{Z}_{26} \to \mathbb{Z}_{26}, \quad p \mapsto p + k \mod 26; \quad D_k : \mathbb{Z}_{26} \to \mathbb{Z}_{26}, \quad c \mapsto c - k \mod 26.$$

We note that encrypting using a key k is the same as shifting the letters by k positions, hence the name "shift cipher".

Example

Let $k = 5$,

$$E_k(\mathtt{A}) = ? \quad E_k(\mathtt{Z}) = ?$$

A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T
0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19
								U	V	W	X	Y	Z						
								20	21	22	23	24	25						

Shift cipher – Example

Example

Let $k = 5$,

$$E_k(\mathtt{A}) = 0 + 5 \mod 26 = \mathtt{F}, \quad E_k(\mathtt{Z}) = 25 + 5 \mod 26 = 4 \mod 26 = \mathtt{E}.$$

To encrypt a message, we can use the following table and replace letters in the first row with those in the second row.

A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T
F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y

U	V	W	X	Y	Z
Z	A	B	C	D	E

Suppose the message is

I STUDY IN BRATISLAVA

Then the corresponding ciphertext (omitting the white spaces) is ?

Shift cipher

We note that encrypting using a key k is the same as shifting the letters by k positions, hence the name "shift cipher".

Example

Let $k = 5$, To encrypt a message, we can use the following table and replace letters in the first row with those in the second row.

A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T
F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y

U	V	W	X	Y	Z
Z	A	B	C	D	E

Suppose the message is

I STUDY IN BRATISLAVA

Then the corresponding ciphertext (omitting the white spaces) is

NXYZIDNSGWFYNXQFAF.

Caeser cipher

- When $k = 3$, the encryption and decryption algorithms are called the *Caesar Cipher*
- Used by Julius Caesar around 50 B.C.
- It is unknown how effective the Caesar cipher was at the time.
- Likely to have been reasonably secure
  - Most of Caesar's enemies would have been illiterate
  - Might have assumed the messages were written in an unknown foreign language.

src: https://www.shutterstock.com/

Security of shift cipher

Exhaustive key search

- Try each possible key value from $\mathcal{K} = \mathbb{Z}_{26}$

Example

- Ciphertext: `NXYZIDNSGWFYNXQFAF`
- Try different key values $k = 1, 2, \ldots$ until we find a sentence that makes sense
- $k = 1$: `MWXYHCMRFVEXMWPEZE`
- $k = 2$: ?

A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T
B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U

U	V	W	X	Y	Z
V	W	X	Y	Z	A

Security of shift cipher

Exhaustive key search

- try each possible key value from $\mathcal{K} = \mathbb{Z}_{26}$

Example

- Ciphertext: `NXYZIDNSGWFYNXQFAF`
- $k = 1$: `MWXYHCMRFVEXMWPEZE`
- $k = 2$: `LVWXGBLQEUDWLVODYD`
- $k = 3$: `KUVWFAKPDTCVKUNCXC`
- $k = 4$: `JTUVEYJOCSBUJTMBWB`
- $k = 5$: `ISTUDYINBRATISLAVA`

We have demonstrated that with an exhaustive key search, we can *break* the shift cipher, i.e. find the key.

$\mathbb{Z}_n^*$

- Recall that $\mathbb{Z}_n^*$ is the set of elements $x \in \mathbb{Z}_n$ such that $\gcd(x, n) = 1$.

Example

$n = 6$, $\mathbb{Z}_6^* = \{ 1, 5 \}$

Affine cipher

Definition (Affine cipher)

Let $\mathcal{P} = \mathcal{C} = \mathbb{Z}_{26}$ and $\mathcal{K} = \{ (a, b) \mid a \in \mathbb{Z}_{26}^*, b \in \mathbb{Z}_{26} \}$. For each key (a, b), define

$$E_{(a,b)} : \mathbb{Z}_{26} \to \mathbb{Z}_{26}, \quad p \mapsto ap+b \mod 26; \quad D_{(a,b)} : \mathbb{Z}_{26} \to \mathbb{Z}_{26}, \quad c \mapsto a^{-1}(c-b) \mod 26.$$

The cryptosystem $(\mathcal{P}, \mathcal{C}, \mathcal{K}, \mathcal{E}, \mathcal{D})$, where $\mathcal{E} = \{ E_{(a,b)} : (a, b) \in \mathcal{K} \}$, $\mathcal{D} = \{ D_{(a,b)} : (a, b) \in \mathcal{K} \}$, is called the *affine cipher*.

When $a = 1$, we have a shift cipher.

Affine cipher – compute decryption key

- Given a key (a, b), to find $a^{-1} \mod 26$, we can apply the extended Euclidean algorithm

Recall – Extended Euclidean algorithm for finding inverse

- $a \in \mathbb{Z}_n^*$
- Using the extended Euclidean algorithm, we can find $x, y \in \mathbb{Z}$ s.t.

$$1 = ax + ny$$

- So $ax \equiv 1 \mod n$, and $a^{-1} = x \mod n$

Example

Suppose the key for affine cipher is $(3, 1)$, by the extended Euclidean algorithm

$$26 = 3 \times 8 + 2, 3 = 2 + 1 \Longrightarrow 1 = 3 - (26 - 3 \times 8) = 3 \times 9 - 26 \Longrightarrow 3^{-1} \mod 26 = 9.$$

Affine cipher – example

Recall affine cipher definition

Let $\mathcal{P} = \mathcal{C} = \mathbb{Z}_{26}$ and $\mathcal{K} = \{ (a, b) \mid a \in \mathbb{Z}_{26}^*, b \in \mathbb{Z}_{26} \}$. For each key (a, b), define

$$E_{(a,b)} : \mathbb{Z}_{26} \to \mathbb{Z}_{26}, \quad p \mapsto ap+b \mod 26; \quad D_{(a,b)} : \mathbb{Z}_{26} \to \mathbb{Z}_{26}, \quad c \mapsto a^{-1}(c-b) \mod 26.$$

Example

Encrypt the word `STROM` using affine cipher with the key $k = (3, 1)$

A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T
0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19

U	V	W	X	Y	Z
20	21	22	23	24	25

Affine cipher – example

Example

Encrypt the word STROM using affine cipher with the key $k = (3, 1)$

A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T
0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19

U	V	W	X	Y	Z
20	21	22	23	24	25

$$3 \times 18 + 1 = 55 \equiv 3 \mod 26, \quad 3 \times 19 + 1 = 58 \equiv 6 \mod 26,$$
$$3 \times 17 + 1 = 52 \equiv 0 \mod 26, \quad 3 \times 14 + 1 = 43 \equiv 17 \mod 26,$$
$$3 \times 12 + 1 = 37 \equiv 11 \mod 26 \implies \text{ciphertext is DGARL.}$$

S	T	R	O	M
18	19	17	14	12
3	6	0	17	11
D	G	A	R	L

Security of affine cipher

Recall

Euler's totient function:

$$\varphi(n) = |\mathbb{Z}_n^*|.$$

For any $n \in \mathbb{Z}$, $n > 1$,

$$\text{if} \quad n = \prod_{i=1}^{k} p_i^{e_i}, \quad \text{then} \quad \varphi(n) = n \prod_{i=1}^{k} \left(1 - \frac{1}{p_i}\right),$$

where p_i are distinct primes.

What is $\varphi(26)$?

Security of affine cipher

Since

$$26 = 2 \times 13,$$

we have

$$\varphi(26) = 26 \times \left(1 - \frac{1}{2}\right)\left(1 - \frac{1}{13}\right) = 12.$$

Recall

Affine cipher key space $\mathcal{K} = \{ (a, b) \mid a \in \mathbb{Z}_{26}^*, b \in \mathbb{Z}_{26} \}$.

- There are 12 possible values for $a \in \mathbb{Z}_{26}^*$.
- There are 26 possible values for $b \in \mathbb{Z}_{26}$.
- Then the total number of possible keys (a, b) is $12 \times 26 = 312$.
- Similarly to shift cipher, knowing a ciphertext, we can try each of the 312 keys until we find a plaintext that makes sense.
- Thus we can break affine cipher by exhaustive key search!

Cryptanalysis

- kryptós: hidden
- analýein: to analyze
- decrypts the message without knowing the key
- successful cryptanalysis recovers the plaintext or even the key

Definition (Kerckhoffs' principle)

The security of a cryptosystem should depend only on the secrecy of the key.

In other words, everything is public knowledge except for the secret key.

- We assume we know the plaintext is an English text.
- We also know the cipher used for communication.
- We assume a ciphertext-only attacker model
- We will show how to recover both the plaintext and the key using *frequency analysis* for affine cipher

Frequency analysis

- Plaintext is an English text
- Analyze the probabilities for the appearance of each letter in a standard English text
- `E` has the highest probability and the second most common letter is `T`
- Similarly, the most common two consecutive letters are `TH`, `HE`, `IN`, ...; and the most common three consecutive letters are `THE`, `ING`, `AND`, ...
- Given a ciphertext that is encrypted using an affine cipher, we expect a permutation of the letters in the ciphertext to have similar frequencies as in the table
  - Because one letter is mapped to one letter

A	0.082	N	0.067
B	0.015	O	0.075
C	0.028	P	0.019
D	0.043	Q	0.001
E	0.127	R	0.060
F	0.022	S	0.063
G	0.020	T	0.091
H	0.061	U	0.028
I	0.070	V	0.010
J	0.002	W	0.023
K	0.008	X	0.001
L	0.040	Y	0.020
M	0.024	Z	0.001

Frequency analysis – affine cipher

Recall

For each key $k = (a, b)$,

$$E_k : \mathbb{Z}_{26} \to \mathbb{Z}_{26}, \quad p \mapsto ap + b \mod 26.$$

Example

Suppose we have the following ciphertext

VCVIRSKPOFPNZOTHOVMLVYSATISKVNVLIVSZVR.

V	S	I	O	R	K	P	N	Z	T	L	C	F	H	M	Y	A
8	4	3	3	2	2	2	2	2	2	2	1	1	1	1	1	1

- The most frequent letter is V and the second most frequent is S.
- Thus, it makes sense to assume V is the ciphertext corresponding to E and S to T.
- Let the key be (a, b), what do we know about a and b?

Frequency analysis – affine cipher

Example

Ciphertext

VCVIRSKPOFPNZOTHOVMLVYSATISKVNVLIVSZVR.

Let the key be (a, b), we have the following equations:

$$4a + b = 21 \mod 26, \quad 19a + b = 18 \mod 26,$$

which gives $15a = 23 \mod 26$. By the extended Euclidean algorithm,

$$26 = 15 \times 1 + 11,\ 15 = 11 \times 1 + 4,\ 11 = 4 \times 2 + 3,\ 4 = 3 + 1,$$

$$\begin{aligned} 1 &= 4 - 3 = 4 - (11 - 4 \times 2) = -11 + 4 \times 3 = -11 + (15 - 11) \times 3 \\ &= 15 \times 3 - 11 \times 4 = 15 \times 3 - (26 - 15) \times 4 = 15 \times 7 - 26 \times 4. \end{aligned}$$

Hence, we have $15^{-1} \mod 26 = 7$ and $a =?$, $b =?$

Frequency analysis – affine cipher

Example

Ciphertext

VCVIRSKPOFPNZOTHOVMLVYSATISKVNVLIVSZVR.

Let the key be (a, b), we have the following equations:

$$4a + b = 21 \mod 26, \quad 19a + b = 18 \mod 26,$$

which gives $15a = 23 \mod 26$. By the extended Euclidean algorithm, $15^{-1} \mod 26 = 7$, we have

$$a = 23 \times 15^{-1} \mod 26 = 23 \times 7 \mod 26 = 5 \mod 26.$$

$$b = 21 - 4a \mod 26 = 21 - 4 \times 5 \mod 26 = 1.$$

To decrypt the message, we compute the decryption key by finding $a^{-1} \mod 26 = 5^{-1} \mod 26 = ?$

Frequency analysis – affine cipher

A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T
0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19

U	V	W	X	Y	Z
20	21	22	23	24	25

Example

Ciphertext

VCVIRSKPOFPNZOTHOVMLVYSATISKVNVLIVSZVR.

Let the key be (a, b), we have computed $a = 5, b = 1$.
To decrypt the message, we compute the decryption key by finding $a^{-1} \mod 26 = 5^{-1} \mod 26$:

$$26 = 5 \times 5 + 1 \Longrightarrow 1 = 26 - 5 \times 5 \Longrightarrow 5^{-1} \mod 26 = -5 \mod 26 = 21.$$

Apply the decryption key $(21, 1)$ to the ciphertext, what is the plaintext?

Frequency analysis – affine cipher

Example

Ciphertext

VCVIRSKPOFPNZOTHOVMLVYSATISKVNVLIVSZVR.

A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T
0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19

U	V	W	X	Y	Z
20	21	22	23	24	25

Apply the decryption key $(21, 1)$ to the ciphertext we get the following plaintext

EVERYTHING IS KNOWN EXCEPT FOR THE SECRET KEY.

Block ciphers

Definition (Block cipher)

A *block cipher* is a symmetric key cryptosystem with $\mathcal{P} = \mathcal{C} = \mathcal{A}^n$ for some alphabet $\mathcal{A}$ and positive integer n. n is called the *block length*.

- For classical ciphers that we have seen (shift cipher, affine cipher), $\mathcal{A} = \mathbb{Z}_{26}$.
- For modern cryptosystems that we will discuss next week, $\mathcal{A} = \mathbb{F}_2 = \{ 0, 1 \}$.
- The block length of a an affine cipher is 1.

Stream ciphers

- For a *stream cipher*, $\mathcal{P} = \mathcal{C} = \mathcal{A}$ are single digits.
- Encryptions are computed on each digit of the plaintext.
- In particular, suppose we have a plaintext string $\boldsymbol{p} = p_1 p_2 \dots$ (where $p_i \in \mathcal{A}$) and a key k.
- We first compute a key stream $\boldsymbol{z} = z_1 z_2 \dots$ using the key k, then the ciphertext is obtained as follows:

$$\boldsymbol{c} = c_1 c_2 \cdots = E_{z_1}(p_1) E_{z_2}(p_2) \dots$$

- A stream cipher is said to be *synchronous* if the key stream only depends on the chosen key k but not on the encrypted plaintext.
- In this case, the sender and the receiver can both compute the keystream synchronously.
- We will see a classical synchronous stream cipher called *one-time pad*.

One-time pad

Definition (One-time pad)

Given a positive integer n, let $\mathcal{P} = \mathcal{C} = \mathcal{K} = \mathbb{F}_2^n$. For any $k \in \mathcal{K}$, define

$$E_k : \mathbb{F}_2^n \to \mathbb{F}_2^n, \quad p \mapsto p \oplus k \qquad D_k : \mathbb{F}_2^n \to \mathbb{F}_2^n, \quad c \mapsto c \oplus k$$

The cryptosystem $(\mathcal{P}, \mathcal{C}, \mathcal{K}, \mathcal{E}, \mathcal{D})$, where $\mathcal{E} = \{ E_k : k \in \mathcal{K} \}$, $\mathcal{D} = \{ D_k : k \in \mathcal{K} \}$, is called the *one-time pad*.

- Recall that vector addition in $\mathbb{F}_2^n$ is defined as bitwise `XOR`, denoted by $\oplus$
- If the attacker has the knowledge of one pair of plaintext p and its corresponding ciphertext c, they can recover the key by computing $p \oplus c = p \oplus p \oplus k = k$.
- Thus each key can be used only once.
- Synchronous stream cipher

Security of one-time pad

- One distinct feature of the one-time pad from the previously introduced classical ciphers is that it achieves perfect secrecy
  - In a ciphertext-only attack setting, the attacker cannot obtain any information about the plaintext no matter how much computing power they have.
- We also note that brute force of the key does not work for one-time pad – by brute force, the attacker can obtain any plaintext of the same length as the original plaintext.

Key management of one-time pad

- Key management is the bottleneck of one-time pad.
- With a plaintext of length n, we will also need a key of length n.
- Each key can only be used once.
- Necessary to share a key of the same length as the message each time before the communication.
- Impractical to use one-time pad.

Introduction to cryptography

- Cryptography
- Cryptographic Primitives
- Cryptosystems
- Classical Ciphers
- Encryption Modes
- Some exercises

Encrypting long messages

- When we use a symmetric block cipher with block length n to encrypt a long message, we first divide this long message into blocks of plaintexts of length n
- Then we apply certain *encryption mode* to encrypt the plaintext blocks.
- If the last block has a length of less than n, padding might be required.
- Different methods exist for padding, e.g, using a constant, or using a random number.

ECB mode

- We have seen a few examples of classical block ciphers.
- For messages that are longer than the block length, the way we encrypted them can be described by

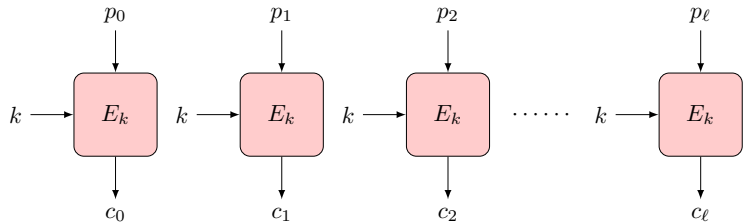


Figure: ECB mode for encryption.

- simplest encryption mode
- *electronic codebook (ECB) mode*

ECB mode

- ECB mode is easy to use, but the main drawback is that the encryption of identical plaintext blocks produces identical ciphertext blocks.
- For an extreme case, if the plaintext is either all 0s or all 1s, it would be easy for the attacker to deduce the message given a collection of plaintext and ciphertext pairs.
- Due to this property, it is also easy to recognize patterns of the plaintext in the ciphertext, which makes statistical attacks easier (e.g. frequency analysis).

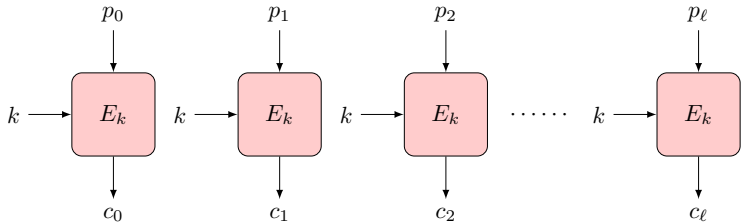


Figure: ECB mode for encryption.

ECB mode – identical plaintext blocks produces identical ciphertext blocks

- The encryption of identical plaintext blocks produces identical ciphertext blocks.
- It is easy to recognize patterns of the plaintext in the ciphertext

(a) Original picture (b) ECB encrypted

ECB mode – encryption and decryption

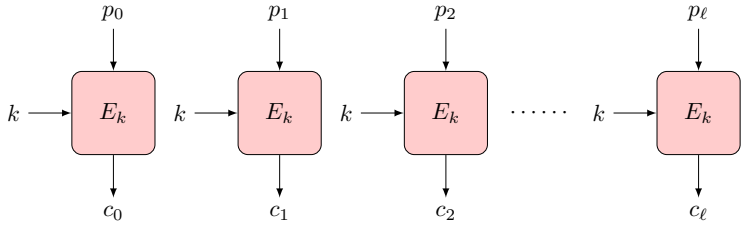


Figure: ECB mode for encryption.

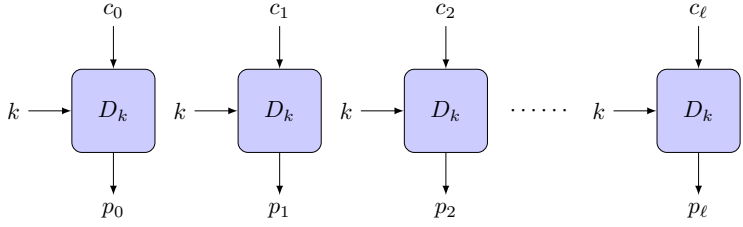


Figure: ECB mode for decryption

CBC mode – encryption

- To avoid the problem of seeing patterns of plaintext in ciphertext, we can use the *cipherblock chaining (CBC) mode*.
- IV stands for *initialization vector*.
- An IV has the same length as the plaintext block and is public.
- We can see that with CBC, the same plaintext is encrypted differently with different IVs.

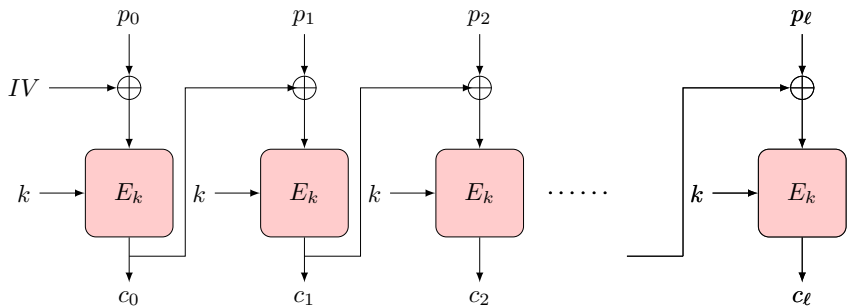
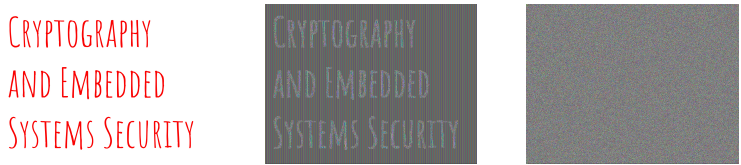


Figure: CBC mode for encryption.

ECB and CBC



(a) Original picture (b) ECB encrypted (c) CBC encrypted

Figure: Original picture and encrypted picture with ECB and CBC modes.

CBC mode – encryption

- If a plaintext block is changed, the corresponding ciphertext block will also be changed, affecting all the subsequent ciphertext blocks.
- Hence CBC mode can also be useful for authentication.

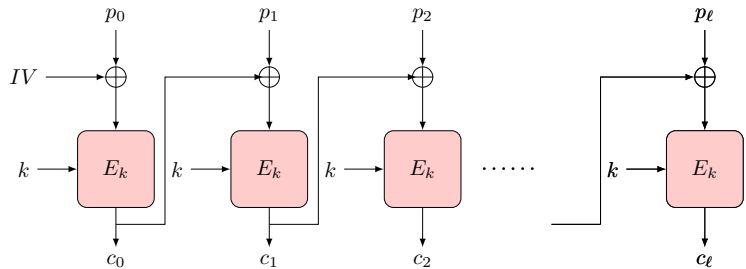
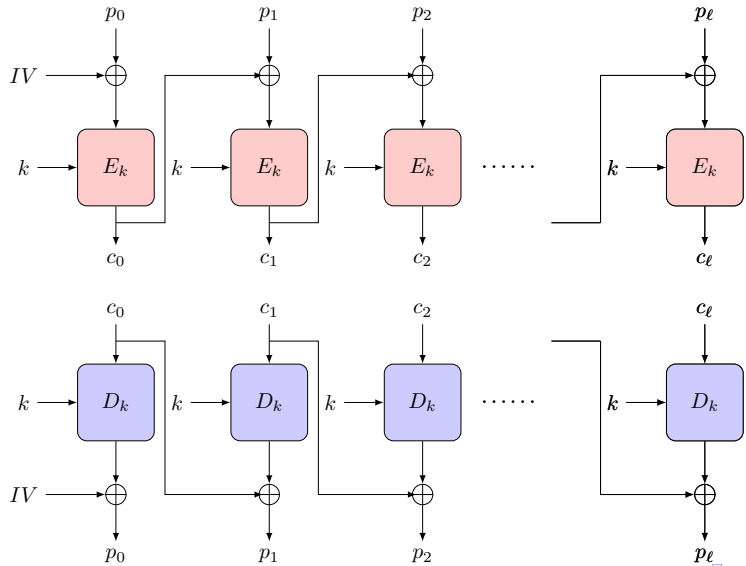


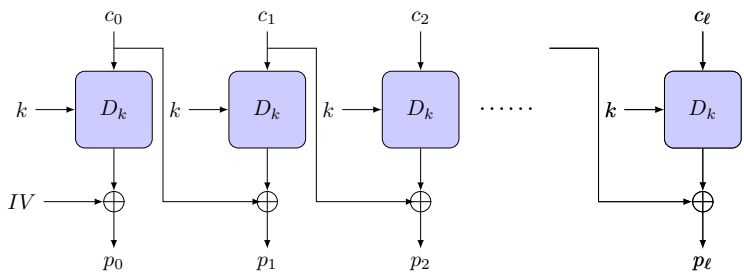
Figure: CBC mode for encryption.

CBC mode – encryption and decryption



CBC mode – decryption

- The receiver needs to wait for the previous ciphertext block to arrive to decrypt the next ciphertext block.
- In real-time applications, *output feedback (OFB) mode* can be used to make communication more efficient.



OFB mode

- The encryption function is not used for encrypting the plaintext blocks, rather it is used for generating a key sequence.
- Ciphertext blocks are computed by `XOR`ing the plaintext blocks and the key sequence.
- Such a design allows the receiver and sender to generate the key sequence simultaneously before the ciphertext is sent.

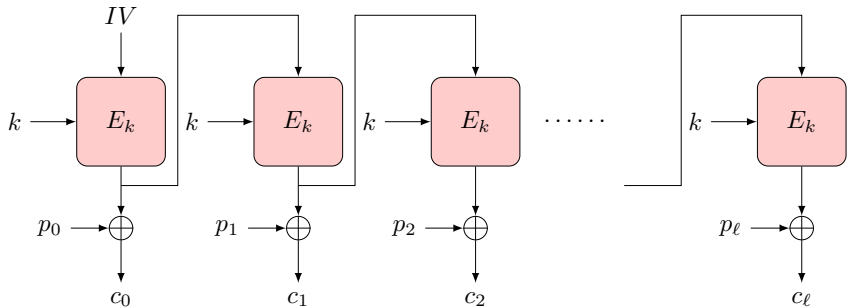


Figure: OFB mode for encryption.

OFB mode

- In a way, OFB mode can be considered as a synchronous stream cipher.
- Another advantage of OFB mode is that padding is not needed.
- However, the encryption of a plaintext block does not depend on the previous blocks, which makes it easier for the attacker to modify the ciphertext blocks.

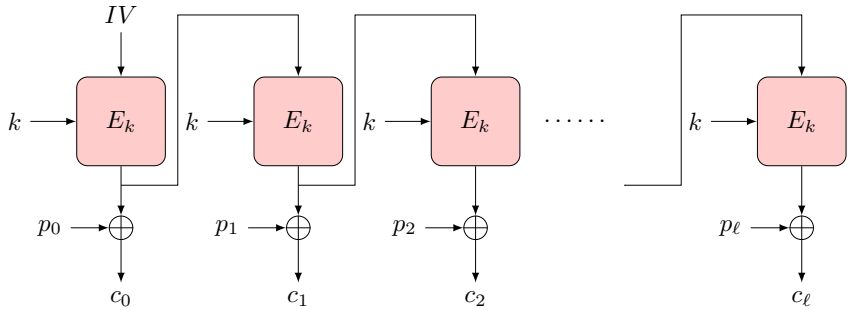
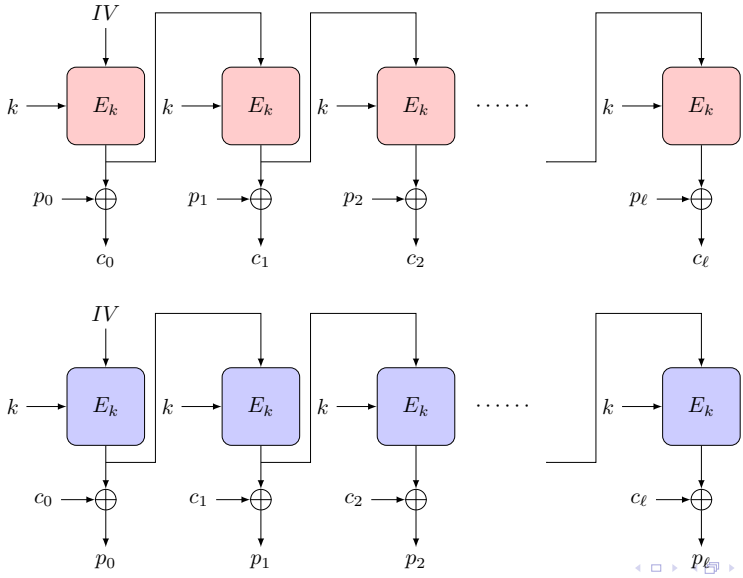


Figure: OFB mode for encryption.

OFB mode – encryption and decryption



Introduction to cryptography

- Cryptography
- Cryptographic Primitives
- Cryptosystems
- Classical Ciphers
- Encryption Modes
- Some exercises

Modular multiplicative inverse

Example

Let

$$p = 5, \quad q = 7$$

Find

$$p^{-1} \mod q = ? \quad q^{-1} \mod p = ?$$

Modular multiplicative inverse

Example

$$p = 5, \quad q = 7$$

By the extended Euclidean algorithm

$$7 = 5 \times 1 + 2, \quad 5 = 2 \times 2 + 1$$

$$1 = 5 - 2 \times 2 = 5 - (7 - 5) \times 2 = 5 \times 3 - 7 \times 2$$

$$5^{-1} \mod 7 = 3, \quad 7^{-1} \mod 5 = -2 \mod 5 = 3.$$

Modular multiplicative inverse

Example

Let

$$p = 7, \quad q = 47$$

Find

$$p^{-1} \mod q = ? \quad q^{-1} \mod p = ?$$

Modular multiplicative inverse

Example

$$p = 7, \quad q = 47$$

By the extended Euclidean algorithm

$$47 = 7 \times 6 + 5, \quad 7 = 5 \times 1 + 2, \quad 5 = 2 \times 2 + 1$$

$$\begin{aligned} 1 &= 5 - 2 \times 2 = 5 - (7 - 5) \times 2 = 5 \times 3 - 7 \times 2 = (47 - 7 \times 6) \times 3 - 7 \times 2 \\ &= 47 \times 3 - 7 \times 20 \end{aligned}$$

$$7^{-1} \bmod 47 = -20 \bmod 47 = 27, \quad 47^{-1} \bmod 7 = 3.$$

Solving linear congruences

Example

Solve the following system of simultaneous linear congruences

$$\begin{aligned} x &\equiv 2 \bmod 5 \\ x &\equiv 1 \bmod 7 \\ x &\equiv 5 \bmod 11 \\ x &\equiv ? \bmod 385 \end{aligned}$$

Solving linear congruences

Example

With the formula we have seen in the lecture

$$m_1 = 5, \quad m_2 = 7, \quad m_3 = 11, \quad a_1 = 2, \quad a_2 = 1, \quad a_3 = 5,$$

$$m = 5 \times 7 \times 11 = 385, \quad M_1 = 77, \quad M_2 = 55, \quad M_3 = 35.$$

$$M_1 \equiv 77 \equiv 2 \mod 5, \quad M_2 \equiv 55 \equiv 6 \mod 7, \quad M_3 \equiv 35 \equiv 2 \mod 11.$$

Using the extended Euclidean algorithm, we compute the modular inverses:

$$y_1 = M_1^{-1} \mod 5 = 3, \quad y_2 = M_2^{-1} \mod 7 = 6, \quad y_3 = M_3^{-1} \mod 11 = 6.$$

And

$$\begin{aligned} x &= \sum_{i=1}^{3} a_i y_i M_i \mod m = 2 \times 3 \times 77 + 1 \times 6 \times 55 + 5 \times 6 \times 35 \mod 385 = \\ &= 1842 \mod 385 = 302. \end{aligned}$$

Solving linear congruences

Example

Solve the following system of simultaneous linear congruences

$$\begin{aligned} x &\equiv 2 \mod 5 \\ x &\equiv 5 \mod 7 \\ x &\equiv 4 \mod 9 \\ x &\equiv ? \mod 315 \end{aligned}$$

Solving linear congruences

Example

With the formula we have seen in the lecture

$$m_1 = 5, \quad m_2 = 7, \quad m_3 = 9, \quad a_1 = 2, \quad a_2 = 5, \quad a_3 = 4,$$

$$m = 5 \times 7 \times 9 = 315, \quad M_1 = 63, \quad M_2 = 45, \quad M_3 = 35.$$

$$M_1 \equiv 63 \equiv 3 \mod 5, \quad M_2 \equiv 45 \equiv 3 \mod 7, \quad M_3 \equiv 35 \equiv 8 \mod 9.$$

Using the extended Euclidean algorithm, we compute the modular inverses:

$$y_1 = M_1^{-1} \mod 5 = 2, \quad y_2 = M_2^{-1} \mod 7 = 5, \quad y_3 = M_3^{-1} \mod 9 = 8.$$

And

$$\begin{aligned} x &= \sum_{i=1}^{3} a_i y_i M_i \mod m = 2 \times 2 \times 63 + 5 \times 5 \times 45 + 4 \times 8 \times 35 \mod 315 = \\ &= 2497 \mod 315 = 292. \end{aligned}$$

Assignment 2

- Read textbook