Technische Hochschule Rosenheim

Computer Science Fundamentals

Cryptography – Classical Methods

Technische Hochschule Rosenheim
Winter 2021/22
Prof. Dr. Jochen Schmidt

Overview/Motivation

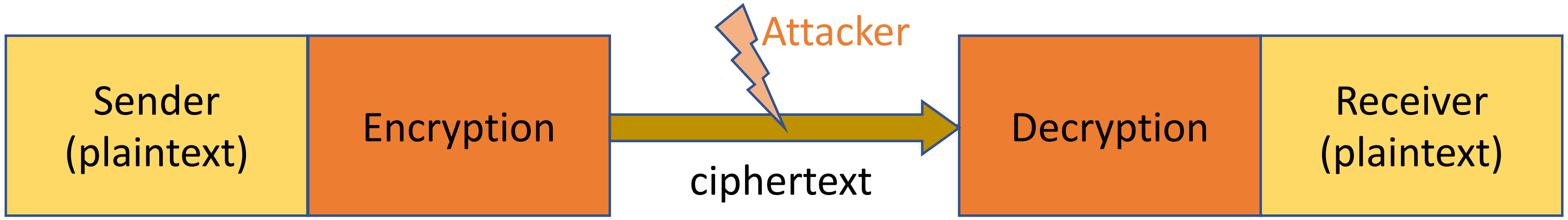
- What is a cryptosystem?
- Some classical methods of cryptography

Encrypted transmission of messages is of great interest

- not only for the military and secret services
- but also for companies (e.g., transmission of confidential information on new products)
- and individuals (e.g., Online-Banking $\longrightarrow$ https)

Principle

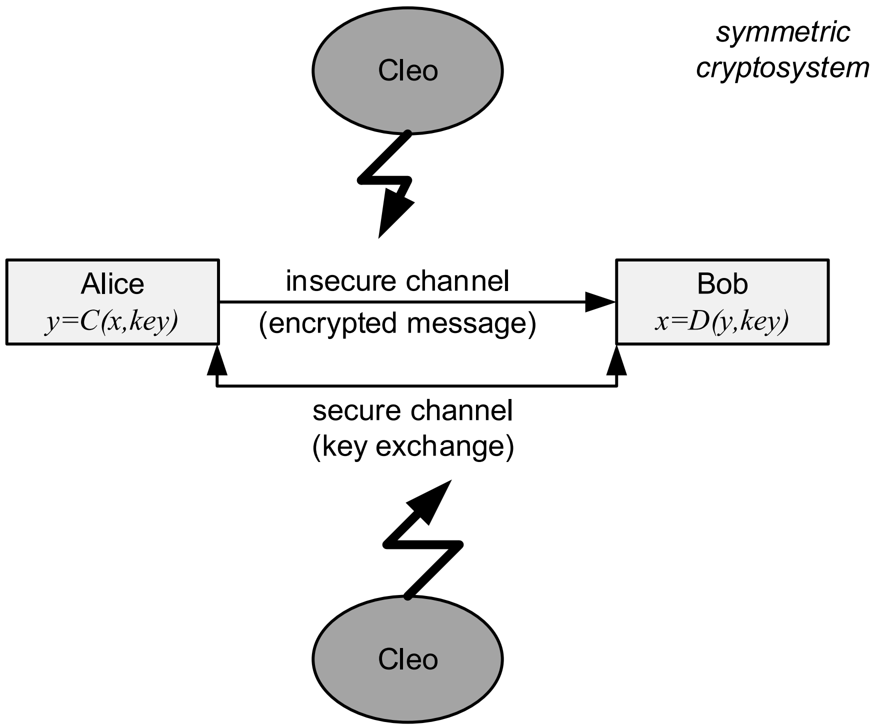
Process



- Encryption of the message (called plaintext) into a ciphertext
  - Use of encryption algorithm
  - and key parameters
- Sender sends ciphertext to the recipient
- Decryption of the ciphertext by recipient
  - Use of a suitable decryption algorithm
  - and the same key parameters
- Recipient gets the plaintext of the message

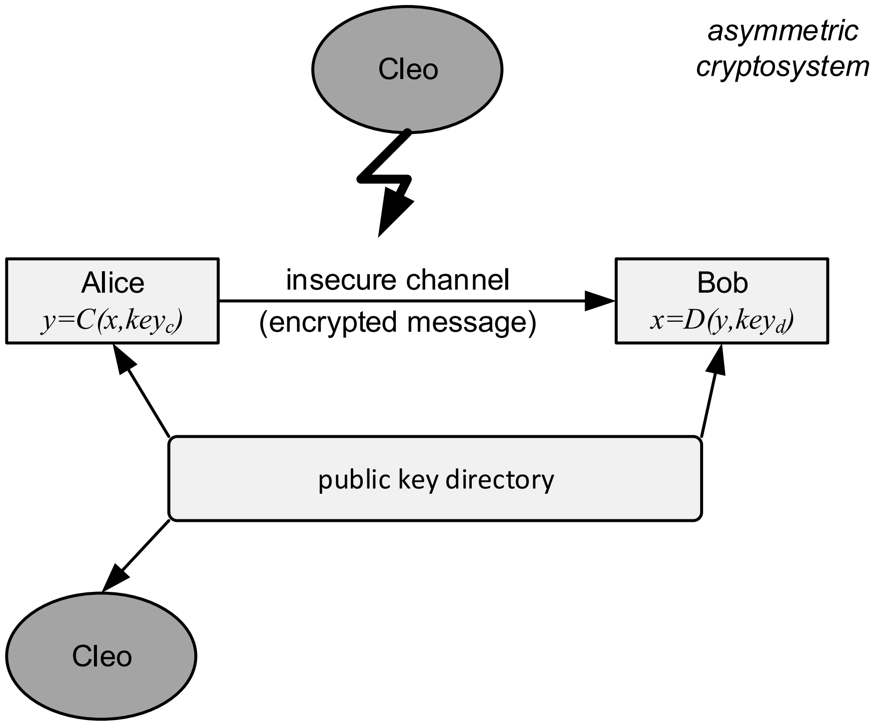
Main Classes of Encryption Methods

Symmetric Encryption



- same secret key for encryption & decryption
- key exchange via secure channel

Asymmetric Encryption



- encryption using public key of recipient
- decryption using private (secret) key of recipient

Kerckhoff‘s Principle

- Formulated 1883
- Principle of all modern cryptographic methods

- Security of a method
  - must **not** be based on secrecy of the algorithm
  - but on secrecy of the key

- This means
  - No „Security through Obscurity“
  - Algorithms are public

Classical Ciphers

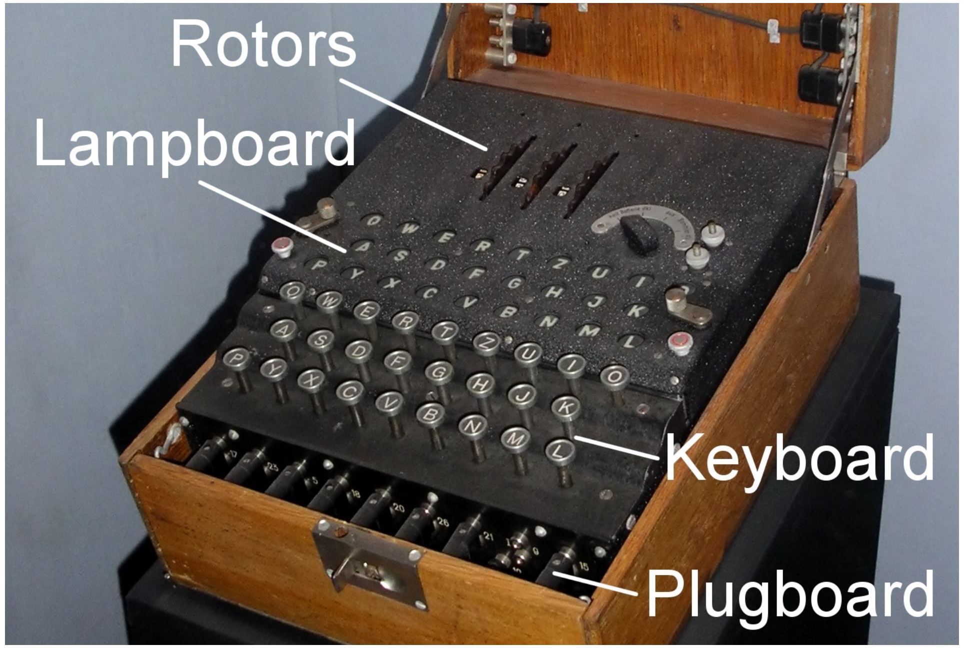
- classical = developed before 1950
- presented here to illustrate the basic encryption principles: substitution ciphers
- classical methods in pure form are no longer in use today
  - but they are a part of modern ciphers like AES

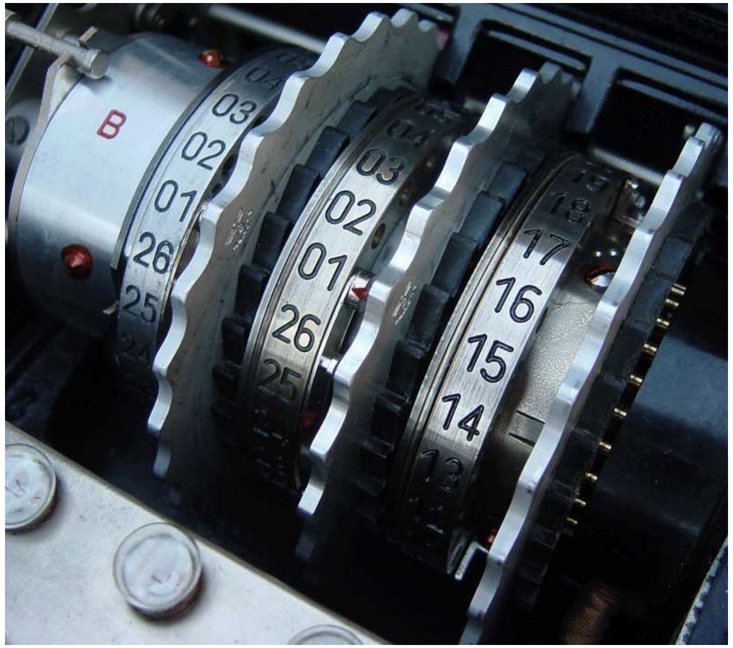
Substitution Ciphers

Technische Hochschule Rosenheim

- Idea: Replace (“substitute”) units of plaintext by corresponding units of ciphertext
- Types of substitution ciphers:
  - **simple** substitution: replace each letter of the plaintext alphabet by (always the same) corresponding letter of the ciphertext alphabet (bijective mapping).
  - **homophonic** substitution: replace each letter of the plaintext alphabet by one of the corresponding symbols of the ciphertext alphabet; one plaintext letter can be mapped to many different ciphertext symbols (homophones).
  - **polyalphabetic** substitution: like “simple”, but different mappings are used for each position in the plaintext using a defined algorithm, e.g., periodically.
  - **polygram** substitution: instead of single letters, replace whole blocks of letters.
- Breakthrough of these methods with the availability of electromechanical encryption machines (like the Enigma, which uses polyalphabetic substitution)

Enigma





Simple Substitution Ciphers

- Example: Use an affine transformation
  - Symbols x_i of an alphabet A containing a total of n symbols are mapped to the same alphabet using

$$x_i \longrightarrow x_{(k \cdot i + d) \bmod n}$$

  - **k** is the multiplicative key
  - **d** is the additive key

- Special cases
  - $k = 1 \longrightarrow$ Caesar Code
  - $d = 0 \longrightarrow$ Product Ciphers

Simple (Affine) Substitution Ciphers

Technische Hochschule Rosenheim

- Alphabet A with n symbols
- symbol in ciphertext =
  - Multiply the position i of a symbol in the alphabet by key k,
  - add d,
  - and reduce mod n.
- Arbitrary combination of key parameters not possible for unambiguous mapping
- Example: $k = 4$, $n = 26$, $d = 0$

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

- Repetitions occur – unsuitable!

Simple (Affine) Substitution Ciphers

- For a suitable combination (k, n):
  - k and n must be relatively prime: gcd(k, n) = 1
  - Only these keys k are suitable, as they have a modular inverse k^{-1} with $k \cdot k^{-1} \bmod n \equiv 1$
  - For computing the modular inverse use
    - extended Euclidean algorithm
    - Euler's/Fermat's theorem
    - details see appendix/maths course

- Example: relatively prime to n = 26 are {1, 3, 5, 7, 9, 11, 15, 17, 19, 21, 23, 25}
  - Therefore: k = 7 is suitable
  - Inverse mod 26: $7^{-1} \equiv 15$
  - Test: $7 \cdot 15 \bmod 26 \equiv 105 \bmod 26 \equiv 1$

Simple (Affine) Substitution Ciphers – Example

Encryption of plaintext *COMPUTER*

- with multiplicative key k=7

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

- and additive key d=5

Encryption

- plaintext : C O M P U T E R
- multiplication (k=7): O U G B K D C P
- shift (d=5): T Z L G P I H U

Decryption with inverse operations

- ciphertext: T Z L G P I H U
- shift (–d = –5): O U G B K D C P
- multiplication (k^{-1}=15): C O M P U T E R

Vigenère Ciphers

- This is a polyalphabetic substitution cipher
- Generalization of Caesar cipher: use multiple substitution tables (with different shifts)

- Define a key that is built from letters of the plaintext alphabet
  - The key defines the parameter d that is used as offset for shifting each letter
- To obtain the letter index of the ciphertext
  1. match positions of repeated key and plaintext
  2. for each position:
     - the index of the plaintext letter is added to
     - the index of the letter in the key

Example:
Latin alphabet (0-25), mod 26, key = `BCD`

- plaintext : `SECRETTEXT`
- key (shift): `BCDBCDBCDB`
- ciphertext: `TGFSGWUGAU`

From Vigenère Ciphers to One-time Pads

- Breaking the Vigenère cipher is getting harder the longer the key

- If the key
  - has the same length as the plaintext,
  - is completely random,
  - and never reused.

  we get the Vernam cipher or one-time pad

One-time Pads

- For one-time pads we typically use the alphabet {0, 1}
  - the plaintext is converted to binary,
  - the key is a random sequence of bits,
  - the mod 2 addition becomes a simple bitwise XOR of plaintext/ciphertext and key for en-/decryption

- Example:

Encryption

- plaintext : 1 0 0 1 1 0 0 1
- key: 0 1 1 1 0 0 1 1 **XOR**
- ciphertext: 1 1 1 0 1 0 1 0

Decryption

- ciphertext: 1 1 1 0 1 0 1 0
- key: 0 1 1 1 0 0 1 1 **XOR**
- plaintext: 1 0 0 1 1 0 0 1

One-time Pads

- One-Time-Pads offer perfect secrecy
  - The encrypted data do not allow any conclusions to be drawn about the plain text except for its length
  - **One-time pads cannot be broken** – no matter how much computing power is invested
  - Proof by Shannon 1949

- Practical limitations
  - this only applies if the key is generated from real random numbers
    - pseudo-randomness is not sufficient
  - the key is as long as the data and has to be exchanged via a secure channel

  ⟶ used very rarely

Appendix: Modular Inverse

Euclidean Algorithm for Greatest Common Divisor

- gcd(a, b) = gcd(b, a mod b)
  - for a >= b
  - Stop if b = 0
  - then a is the gcd

- Examples:
  - gcd(26, 13) = gcd(13, 0) ⟶ gcd = 13
  - gcd(26, 7) = gcd(7, 5)
    = gcd(5, 2)
    = gcd(2, 1)
    = gcd(1, 0) ⟶ gcd = 1

Extended Euclidean Algorithm

For determining the modular inverse

- $\gcd(a, b) = s \cdot a + t \cdot b$
  - s, t are integers
  - if $\gcd(a, b) = 1 \Rightarrow t$ is the (multiplicative) modular inverse of b (mod a)
- Example: modular inverse of 7 mod 26

$$
\begin{array}{llll}
26 = 3 \cdot 7 + 5 & \longrightarrow 5 & = 26 - 3 \cdot 7 & \\
7 = 1 \cdot 5 + 2 & \longrightarrow 2 & = 7 - 1 \cdot 5 & = 7 - (26 - 3 \cdot 7) = \\
 & & = -26 + 4 \cdot 7 & \\
5 = 2 \cdot 2 + 1 & \longrightarrow 1 & = 5 - 2 \cdot 2 & \\
 & & = 26 - 3 \cdot 7 - 2 \cdot (-26 + 4 \cdot 7) & \\
 & & = 3 \cdot 26 - 11 \cdot 7 & \\
2 = 1 \cdot 2 + 0 & & &
\end{array}
$$

Inverse exists

and equals –11 = 15 mod 26

Euler's ϕ-Function

- The function's value is the number of natural numbers
  - that are smaller than n
  - and are relatively prime to n
  - $\phi(n) = | \{1 \leq x \leq n \mid \gcd(x, n) = 1\} |$

- Computation (p, q are prime numbers $p \neq q$)
  - $\phi(p) = p - 1$ all integers from 1 to $p - 1$ are relatively prime to p
  - $\phi(pq) = \phi(p)\phi(q) = (p-1)(q-1)$
  - $\phi(p^i) = p^{i-1}(p-1)$
  - $\phi(p^i q^j) = \phi(p^i)\phi(q^j) = p^{i-1}(p-1)\, q^{j-1}(q-1)$

- Examples
  - $\phi(5) = 4$
    - there are four numbers < 5 that are relatively prime to 5, namely 1, 2, 3, 4
  - $\phi(15) = \phi(3 \cdot 5) = \phi(3)\phi(5) = 2 \cdot 4 = 8$
  - $\phi(27) = \phi(3^3) = 3^2 \cdot (3-1) = 9 \cdot 2 = 18$
    - the numbers that are relatively prime to 27 are: 1, 2, 4, 5, 7, 8, 10, 11, 13, 14, 16, 17, 19, 20, 22, 23, 25, 26
  - $\phi(72) = \phi(2^3 \cdot 3^2) = 2^2 \cdot (2-1) \cdot 3^1 \cdot (3-1) = 4 \cdot 3 \cdot 1 \cdot 2 = 24$

Euler‘s/Fermat‘s Theorem – Modular Inverse

- **Euler’s** theorem:
  for all $x \in \mathbb{Z}, n \in \mathbb{N}, \mathrm{ggT}(x, n) = 1$:

$$x^{\phi(n)} \bmod n = 1$$

- Special case: n is a prime number $p \longrightarrow$ **Fermat’s** little theorem:

$$x^{p-1} \bmod p = 1$$

- It holds:

$$x \cdot x^{\phi(n)-1} \bmod n = 1$$

  and therefore:

$$x^{-1} = x^{\phi(n)-1} \bmod n$$

  or, for a prime:

$$x^{-1} = x^{p-2} \bmod p$$

Modular Inverse/Euler – Example

- Using a prime number as module: $p = 31$
  - wanted: modular inverse for $x = 2$
  - it holds: $2^{-1} = 2^{31-2} \bmod 31 = 2^{29} \bmod 31 = 16$
  - Test: $2 \cdot 16 = 32 \bmod 31 = 1$

- With $n = 26$
  - wanted: modular inverse for $x = 7$
  - determine $\phi(26)$, i.e., the **number** of positive integers that are relatively prime to 26.
    - Prime factorization:
      $26 = 13 \cdot 2$
    - therefore: $\phi(26) = \phi(13)\phi(2) = 12 \cdot 1 = 12$
  - it holds: $7^{-1} = 7^{12-1} \bmod 26 = 7^{11} \bmod 26 = 15$
  - Test: $7 \cdot 15 = 105 \bmod 26 = 1$