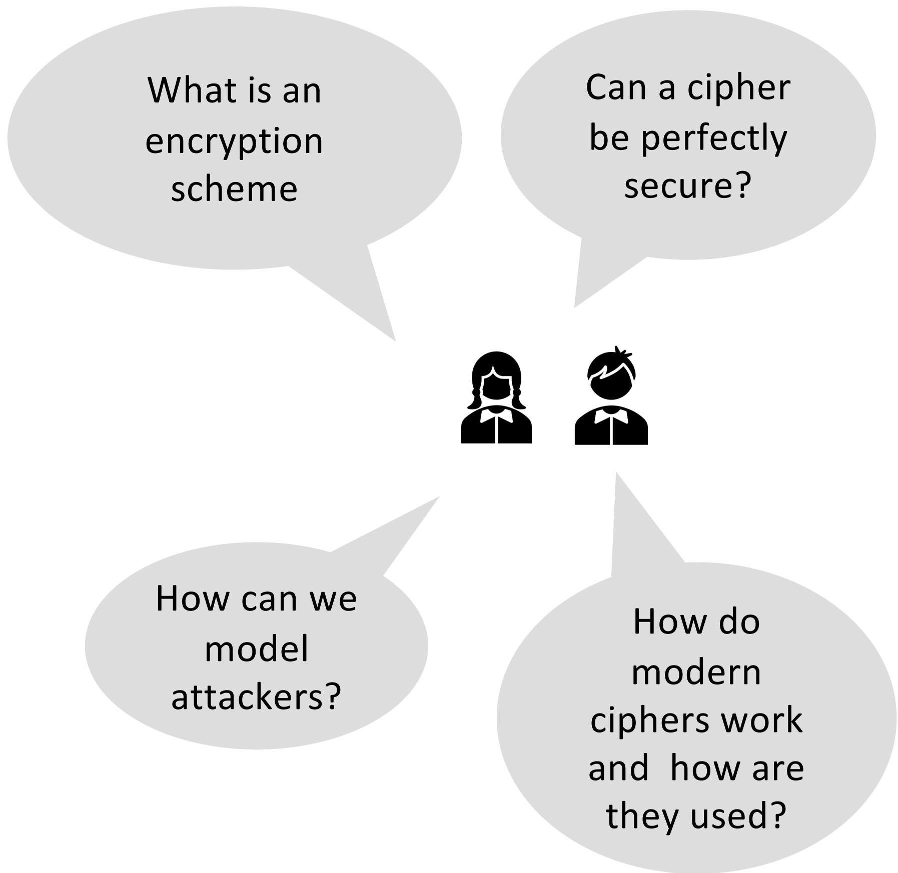
IT-Security

Chapter 2: Symmetric Encryption

Prof. Dr.-Ing. Ulrike Meyer

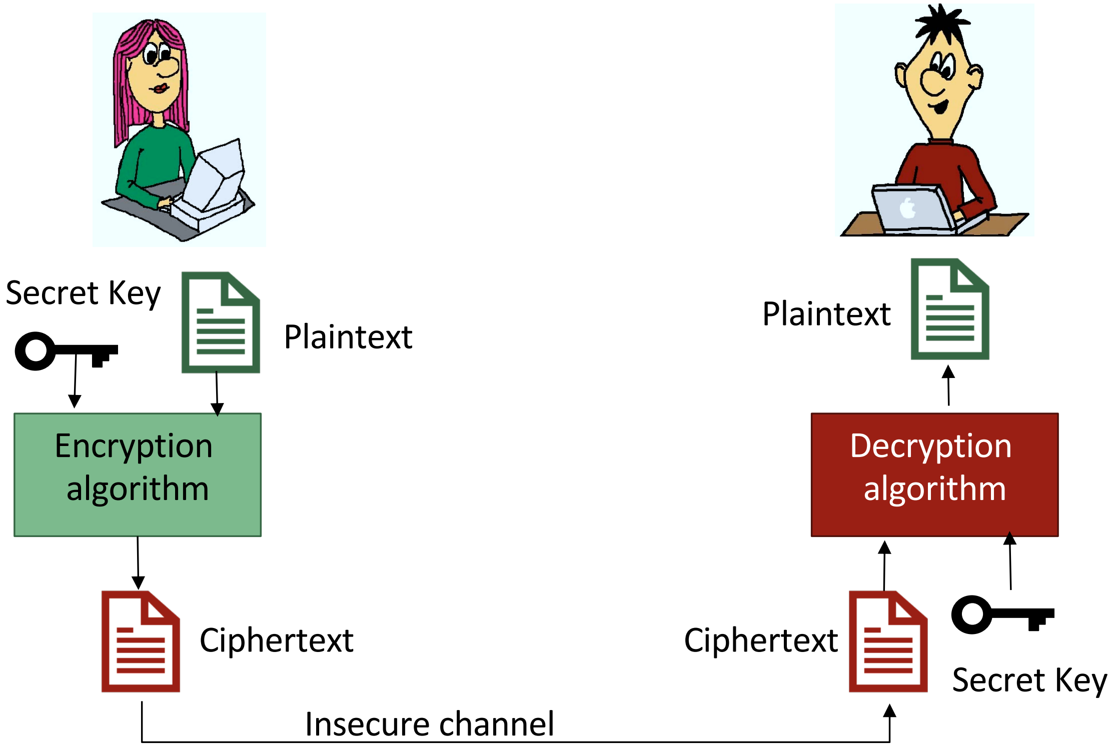
Overview

- **Introduction**
  - Intuition
  - Formal definition
  - Historic examples
- **Computational Security**
  - Attacker models
    - Knowledge
    - Goal
    - Strategy
- **Perfect Secrecy**
  - Definition
  - Shanon’s theorem
  - One-time-pad
- **Practical Schemes**
  - Stream ciphers
  - Block ciphers
  - Modes of encryption



Intuition on Symmetric Ciphers

- Alice wants to send a **confidential** plaintext to Bob
- Alice and Bob **share** a **secret key**
- Alice uses the key to **encrypt** plaintext to ciphertext
- Bob uses the key to **decrypt** ciphertext to plaintext
- Decryption is **"difficult"** without the key



Formal Definition of Encryption Scheme

- **An encryption scheme is a five-tuple ($\mathcal{P}$,$\mathcal{C}$,$\mathcal{K}$,$\mathcal{E}$,$\mathcal{D}$) consisting of**
  - The plaintext space $\mathcal{P}$ of plaintexts (e.g., $\mathcal{P} = \{0,1\}^n$ for some n $\in \mathbb{N}$)
  - The cipher space $\mathcal{C}$ of ciphertexts (e.g., $\mathcal{C} = \{0,1\}^m$ for some m $\in \mathbb{N}$)
  - A key space $\mathcal{K}$ of keys (e.g., $\mathcal{K} = \{0,1\}^k$ for some k $\in \mathbb{N}$)
  - A family $\mathcal{E} = \{E_K : K \in \mathcal{K}\}$ of functions $E_K : \mathcal{P} \rightarrow \mathcal{C}$ called encryption functions
  - A family $\mathcal{D} = \{D_K : K \in \mathcal{K}\}$ of functions $D_K : \mathcal{C} \rightarrow \mathcal{P}$ called decryption functions
- **Such that for any $K_1 \in \mathcal{K}$ there is a $K_2 \in \mathcal{K}$ such that**
  - For all $P \in \mathcal{P}$ it holds that $D_{K_2}(E_{K_1}(P)) = P$
- **In a symmetric encryption scheme the encryption and decryption keys are the same**
- **Note that this definition does not cover any notion of security yet**

Kerckhoff’s Principle 1883

A cryptosystem should be secure even if everything about the system, **except the key,** is public knowledge

- **In contrast:**
  - Keeping the design of a cryptosystem secret is often referred to as **“security by obscurity”**

Example Caesar Cipher

- **The cipher**
  - Plaintext space = ciphertext space = {A,…, Z}, Key space = {1,…,25}
  - Replace each plaintext letter with the one k letters after it. E.g., for k = 4

Plaintext	A	B	C	D	E	F	G	H	I	J	K	L	M
Ciphertext	E	F	G	H	I	J	K	L	M	N	O	P	Q

Plaintext	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
Ciphertext	R	S	T	U	V	W	X	Y	Z	A	B	C	D

- **Security of the Caesar cipher**
  - Assume a message has been encrypted letter by letter using the Cesar cipher
  - Try out each of the 25 keys and check if the resulting plaintext makes sense
    - Requires recognizable plaintext
  - The key space is too small!

⚠ **A secure cipher requires a large key space**

Brute Force Attack on the Caesar Cipher

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

SECURITY

WIGYVMXC

	WIGYVMXC
k=1?	VHFXULWB
k=2?	UGEWTKVA
k=3?	TFDVSJUY
k=4	SECURITY
k=5?	RDBTQHSX
…	

- If the message is **short**, **multiple keys** may lead to **sense making plaintexts**
- If the message is long enough, on **average** key found after ½ $|\mathcal{K}|$ tries
- Brute force attacks are also known as exhaustive search attacks

Monoalphabetic Substitution Cipher

- **Idea**
  - Replace each plaintext letter with one specific other letter according to a substitution table
  - Plaintext space = ciphertext space = {A,…Z}
  - Key space = all permutations of the letters A,…, Z
  - Size of the key space: $|\mathcal{K}| = 26! = 4.0329146 \cdot 10^{26}$
- **Example**

Plaintext	**A**	**B**	**C**	**D**	**E**	**F**	**G**	**H**	**I**	**J**	**K**	**L**	**M**
Ciphertext	D	H	C	E	Z	W	V	S	J	M	L	O	Q

Plaintext	**N**	**O**	**P**	**Q**	**R**	**S**	**T**	**U**	**V**	**W**	**X**	**Y**	**Z**
Ciphertext	P	A	F	K	G	N	B	R	T	Y	I	X	U

- **Trying out each possible key is quite time consuming!**

Exhaustive Search for Monoalphabetic Ciphers

- **Let's assume we**
  - Can decrypt 5 characters per ms
  - Need to decrypt 100 characters to be sure we found the right key

Difficulty of exhaustive search depends on
- **size of key space**
- **resources of attacker**

- **Then we will on average need** $\frac{1}{2} \cdot \frac{100}{5} \cdot |\mathcal{K}| = \frac{1}{2} \cdot 20 \cdot |\mathcal{K}|$ **ms to find the right key**
  - That is $10 \cdot 4.0329146 \cdot 10^{26}$ ms = $4.0329146 \cdot 10^{27}$ ms = $4.0329146 \cdot 10^{24}$ s = $6.7215243 \cdot 10^{22}$ min = $1.2788288 \cdot$ **10^{17} years**
- **Let's assume we**
  - Can decrypt 500 000 characters per ms and still need to decrypt 100 characters in order to be sure
- **Then we will on average need** $\frac{1}{2} \cdot \frac{100}{500\,000} \cdot |\mathcal{K}| = \frac{1}{2} \cdot \frac{1}{5\,000} \cdot |\mathcal{K}|$ **ms to find the right key**
  - That is $10^{-4} \cdot 4.0329146 \cdot 10^{26}$ ms = $4.0329146 \cdot 10^{22}$ ms = $4.0329146 \cdot 10^{19}$ s = $6.7215243 \cdot 10^{17}$ min = $1.2788288 \cdot$ **10^{12} years**

Example Letter Frequencies

- **For any given language and text basis one can determine the relative letter frequencies**

Letter	ENG	GER
A	**8.167%**	6.516%
B	1.492%	1.886%
C	2.782%	2.732%
D	4.253%	5.076%
E	**12.702%**	16.396%
F	2.228%	1.656%
G	2.015%	3.009%
H	6.094%	4.577%
I	**6.966%**	6.550%

Letter	ENG	GER
J	0.153%	0.268%
K	0.772%	1.417%
L	4.025%	3.437%
M	2.406%	2.534%
N	6.749%	9.776%
O	**7.507%**	2.594%
P	1.929%	0.670%
Q	0.095%	0.018%
R	5.987%	7.003%

Letter	ENG	GER
S	6.327%	7.270%
T	**9.056%**	6.154%
U	2.758%	4.166%
V	0.978%	0.846%
W	2.360%	1.921%
X	0.150%	0.034%
Y	1.974%	0.039%
Z	0.074%	1.134%

Top 5 letters in English texts

Letter	ENG
E	**12.702%**
T	**9.056%**
A	**8.167%**
O	**7.507%**
I	**6.966%**

- **Other useful frequencies include, Bigrams, double letters, etc.**

Frequency Analysis

- **Can be used to**
  - Break any cipher that **preserves frequencies**
    - As long as enough ciphertext is available that has been produced by the same key
- **E.g., Monoalphabetic Substitution Ciphers can be broken this way**

A large key space is necessary but does not guarantee a secure cipher

So, how can we get a secure cipher and what does secure mean anyway

Frequency Analysis

- Given a (long) ciphertext in a known language
- Count the frequency of each letter occurring in the ciphertext
- Replace them according to their frequency in the natural language
- Check if the resulting plaintext makes sense

Example Frequency Analysis on Monoalphabetic Substitution Cipher

Top 5
E
T
A
O
I

- **Ciphertext C**
  - JW XAR DGZ FDGDPAJE XAR HZOJZTZ BSDB D TZGX ZTJO DBBDCLZG JN ARB BA VZB XAR
  - JW XAR DGZ FDGDPAJE XAR HZOJZTZ BSDB D TZGX ZTJO DBBDCLZG JN ARB BA VZB XAR
  - I? ?O? A?E ?A?A?OI? ?O? ?E?IE?E T?AT A ?E?? E?I? ATTA??E? I? O?T TO ?ET ?O?

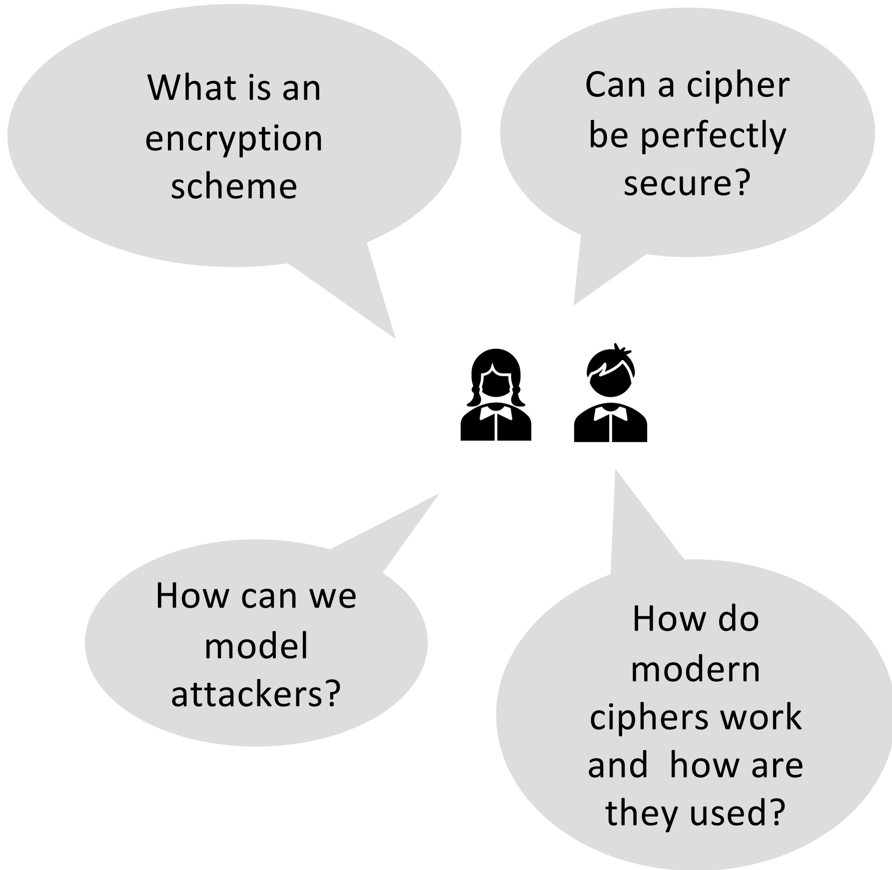
Letter in C	Z	B	D	A	J	G	C	L	X	R	W	F	P	E	D	T	O	N	V	H
Frequency	8	7	7	6	5															
Replace with	E	T	A	O	I	**R**	**C**	**K**	**Y**	**U**	F	P	N	D	H	V	L	S	G	B

  - I? YOU ARE ?ARA?OI? YOU ?E?IE?E T?AT A ?ERY E?I? ATTACKER I? OUT TO ?ET YOU
  - IF YOU ARE PARANOID YOU BELIEVE THAT A VERY EVIL ATTACKER IS OUT TO GET YOU

- **Gives us 20 letters for which the mapping is known, i.e. 76,9% of the key**

Overview

- **Introduction**
  - Intuition
  - Formal definition
  - Historic examples
- **Computational Security**
  - Attacker models
    - Knowledge
    - Goal
    - Strategy
- **Perfect Secrecy**
  - Definition
  - Shanon’s theorem
  - One-time-pad
- **Practical Schemes**
  - Stream ciphers
  - Block ciphers
  - Modes of encryption



Perfect Secrecy

- **Idea of Shanon**
  - A ciphertext should not reveal any new information on the plaintext

Definition:

An encryption scheme is said to provide **perfect secrecy** if

Given a probability distribution Pr on $\mathcal{P}$, and $\Pr(P) > 0$ for all plaintexts P

For each $P \in \mathcal{P}, C \in \mathcal{C}$ and $K \in \mathcal{K}$ chosen uniformly at random $\Pr(\boldsymbol{P}|\boldsymbol{C}) = \Pr(\boldsymbol{P})$

> Whether or not C is observed, P is as likely as its occurrence in the plaintext space

- **This implies: $|\mathcal{K}| \geq |\mathcal{C}| \geq |\mathcal{P}|$ for a perfectly secure encryption scheme**
  - $|\mathcal{C}| \geq |\mathcal{P}|$ holds for any encryption scheme as the encryption functions need to be injective
  - If $|\mathcal{K}| < |\mathcal{C}|$ would hold, then for any $P \in \mathcal{P}, \{E_k(P) \mid k \in \mathcal{K}\} \neq \mathcal{C}$, i.e., there is a $C \in \mathcal{C}$ that does not occur as ciphertext of P such that $\Pr(P|C) = 0$ for this C
  - As we assume $\Pr(P) > 0$, this contradict the perfect forward secrecy

Equivalent Formulations for Perfect secrecy

Definition:

Given a probability distribution Pr on $\mathcal{P}$, and $\Pr(P) > 0$ for all plaintexts P

An encryption scheme is said to provide **perfect secrecy** if

For each $P \in \mathcal{P}, C \in \mathcal{C}$ and $K \in \mathcal{K}$ chosen uniformly at random

$\Pr(\boldsymbol{P}|\boldsymbol{C}) = \Pr(\boldsymbol{P})$

Equivalent

1. Pr(C|P) = Pr(C)
2. $\Pr(C|P_1) = \Pr(C|P_2)$

Proof of 1.:

"$\Leftarrow$": Assume $\Pr(C|P) = \Pr(C)$, then $\frac{\Pr(C|P)\Pr(P)}{\Pr(C)} = \Pr(\mathrm{P})$

as $\Pr(C|P)P(P) = \Pr(P|C)\Pr(C)$ it follows that $\Pr(P)$

$= \Pr(P|C)$

"$\Rightarrow$": Symmetrical argument

Equivalent Formulations for Perfect secrecy

Definition:

Given a probability distribution Pr on $\mathcal{P}$, and $\Pr(P) > 0$ for all plaintexts P

An encryption scheme is said to provide **perfect secrecy** if

For each $P \in \mathcal{P}, C \in \mathcal{C}$ and $K \in \mathcal{K}$ chosen uniformly at random

$\Pr(\boldsymbol{P}|\boldsymbol{C}) = \Pr(\boldsymbol{P})$

Equivalent

1. Pr(C|P) = Pr(C)
2. $\Pr(C|P_1) = \Pr(C|P_2)$

Proof of 2.:

"$\Longrightarrow$ ": Follows directly from 1.

If $\Pr(C|P) = \Pr(C)$ for any $P \in \mathcal{P}, C \in \mathcal{C}$

then $\Pr(C|P_1) = \Pr(C|P_2)$ for any $P_1, P_2 \in \mathcal{P}, C \in \mathcal{C}$

Proof of 2.:

"$\Leftarrow$ ": If $\Pr(C|P_1) = \Pr(C|P_2) = x$ for any $P_1, P_2 \in \mathcal{P}$, $C \in \mathcal{C}$, then

$\Pr(C) = \sum_P \Pr(C|P) \Pr(P) = x \sum_P Pr(P) = x = \Pr(C|P)$

Shannon's Theorem 1949

Shannon's Theorem:

Let $|\mathcal{P}| = |\mathcal{C}| = |\mathcal{K}|$, and $\Pr(P) > 0$ for all plaintexts P.

Then an encryption scheme provides **perfect secrecy** ⇔

1. **K chosen uniformly at random** for **each plaintext** to encrypt and
2. **for each** $\boldsymbol{P} \in \mathcal{P}$ **and** $\boldsymbol{C} \in \mathcal{C}$ **there is exactly one** $\boldsymbol{K} \in \mathcal{K}$ **with** $\boldsymbol{E_K(P) = C}$

A cipher providing perfect secrecy cannot be broken by an attacker.

Not even by one with infinite computational resources and infinite time

Proof Sketch for Shanon's Theorem

Proof

"$\Longrightarrow$ "Assume encryption scheme is perfectly secure

- Let $P \in \mathcal{P}$ and assume there is a $C \in \mathcal{C}$ such that there is no K with $E_K(P) = C$,
- then $\Pr(P|C) = 0$ and thus $\Pr(P) \neq \Pr(P|C)$ which contradicts the perfect secrecy.
- Consequently, there must be at least one K such that $E_K(P) = C$. As there are as many keys as ciphertexts, there must be exactly one such K for each P and C.
- If K was not chosen uniformly, then given C, there would be some plaintexts that is more likely, than others. This again contradicts the perfect secrecy.

"$\Longleftarrow$" Assume each key is equally likely and for each P, C and there is exactly one K such that $E_K(P) = C$.

- Then, $\Pr(C|P) = \frac{1}{|\mathcal{K}|}$ such that for any C and P_1, P_2 it holds that $\Pr(C|P_1) = \Pr(C|P_2) = \frac{1}{|\mathcal{K}|}$, such that the second equivalent definition of perfect secrecy holds

The One-Time-Pad (OTP)

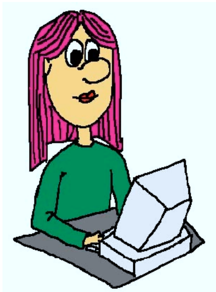
- **Plaintext space, ciphertext space, key space**
  - $\mathcal{P} = \mathcal{C} = \mathcal{K} = \{0,1\}^n$ for some $n \in \mathbb{N}$,
- **Key Generation:**
  - Pick $K \in \mathcal{K}$ **uniformly at random** for each $P \in \mathcal{P}$ to encrypt
- **Encryption:**

$$C = P \oplus K$$

- **Decryption**

$$C \oplus K = P \oplus K \oplus K = P$$

Also Known as
Vernam Cipher or
Vernam's one-time-pad



$$\begin{aligned} P &= 10111101 \\ &\oplus \\ K &= 00110010 \\ &\parallel \\ C &= 10001111 \end{aligned}$$

$$\begin{aligned} C &= 10001111 \\ &\oplus \\ K &= 00110010 \\ &\parallel \\ P &= 10111101 \end{aligned}$$

Perfect Secrecy of the One-Time-Pad

Theorem:

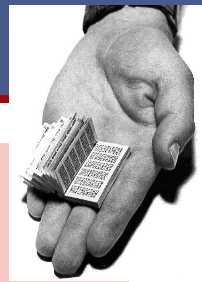
The One-Time-Pad provides perfect secrecy

Proof:

- Follows directly from Shannon's Theorem:
  - As $|\mathcal{P}| = |\mathcal{C}| = |\mathcal{K}|$ per definition of the OTP, we can apply Shannon's Theorem
  - Key is selected uniformly at random in one-time pad $\Longrightarrow$ each key is equally likely
  - Given any pair C, P of ciphertext and plaintext there is a key K that encrypts P to C, namely $K = P \oplus C$:

    $$E_K(P) = P \oplus K = P \oplus (P \oplus C) = C$$

Properties of the One-Time-Pad



Advantages

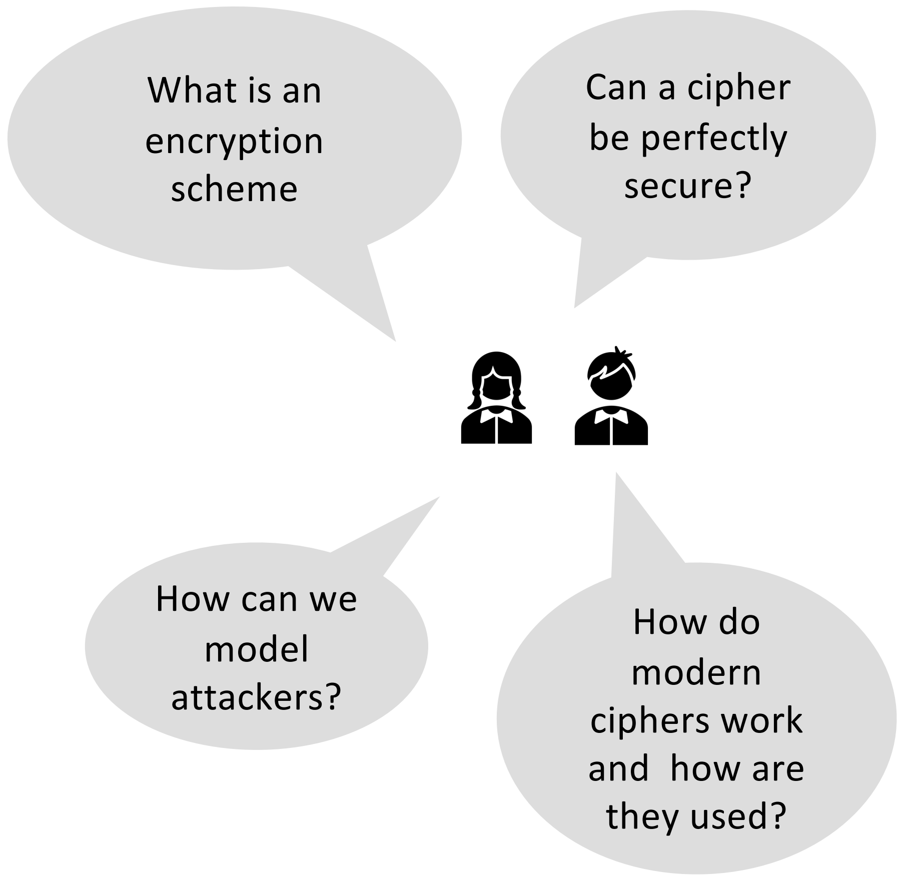
- **Easy to compute**
  - Encryption and decryption are the same operation
  - Bitwise XOR is very cheap to compute
- **As secure as theoretically possible**
  - Given a ciphertext, all plaintexts are equally likely
  - Security independent on the attacker's computational resources

Disadvantages

- **Key must be as long as plaintext**
  - Impractical in most realistic scenarios
  - Still used for diplomatic and intelligence traffic
- **Does not guarantee integrity**
  - One-time pad only guarantees confidentiality
  - Attacker cannot recover plaintext, but can easily change it to something else without being detected
- **Insecure if keys are reused**
  - Attacker can obtain XOR of plaintexts
- **Obviously not practical for all applications**

Overview

- **Introduction**
  - Intuition
  - Formal definition
  - Historic examples
- **Computational Security**
  - Attacker models
    - Knowledge
    - Goal
    - Strategy
- **Perfect Secrecy**
  - Definition
  - Shanon’s theorem
  - One-time-pad
- **Practical Schemes**
  - Stream ciphers
  - Block ciphers
  - Modes of encryption



Practical Modern Encryption Schemes

- **Most encryption schemes used in practice do not provide perfect secrecy**
  - Stream ciphers try to simulate the OTP based on a small random seed
  - Block cipher encrypt complete blocks of plaintexts instead of single bits
- **When do we call such encryption schemes secure?**

Computational Security

An encryption scheme is called **computationally secure** if

- All known attacks against the cipher are computationally infeasible
- I.e., theoretically possible but would take too much time to be practical for any (reasonable) amount of resources

Attacker Models

General assumption in any attack

- Attacker knows which cipher is used
- In line with Kerckhoff's principle

Attack result

- (Partial) key recovery
  - Attacker tries to retrieve (part of) the key
- (Partial) plaintext recovery
  - Attacker tries to retrieve (part of) the plaintext

Key recovery implies plaintext recovery but not the other way round

Power of attacker

Strength of attacker increases

- Cipher-text-only attack
  - Attacker knows only ciphertext
- Known-plaintext attack
  - Knows some pairs of plaintext and ciphertext
- Chosen-plaintext attack — **Chapter 4**
  - Can obtain ciphertext for plaintexts of his choice
- Chosen-ciphertext attack — **Chapter 4**
  - Can obtain plaintext for ciphertexts of his choice before target ciphertext is known

Illustration of Ciphertext-only Attack

- **A classical eavesdropper has access to ciphertext**
- **Thus, he can collect ciphertext(s) and try to**
  - Recover the key and/or
  - Recover the plaintext

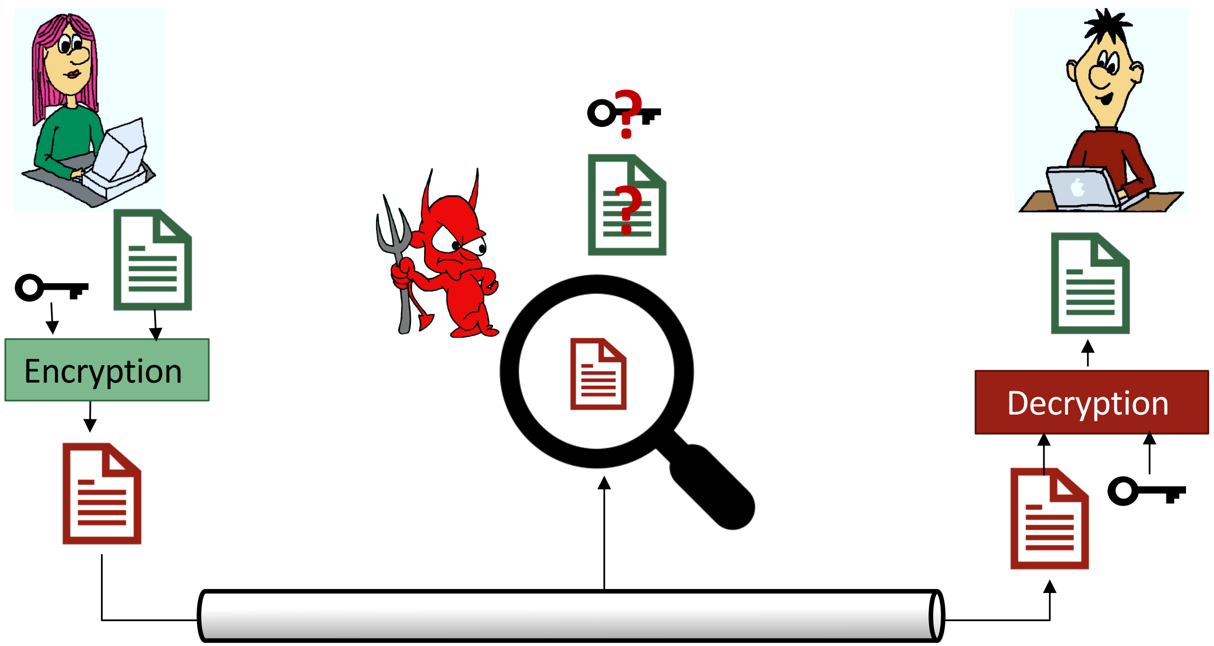
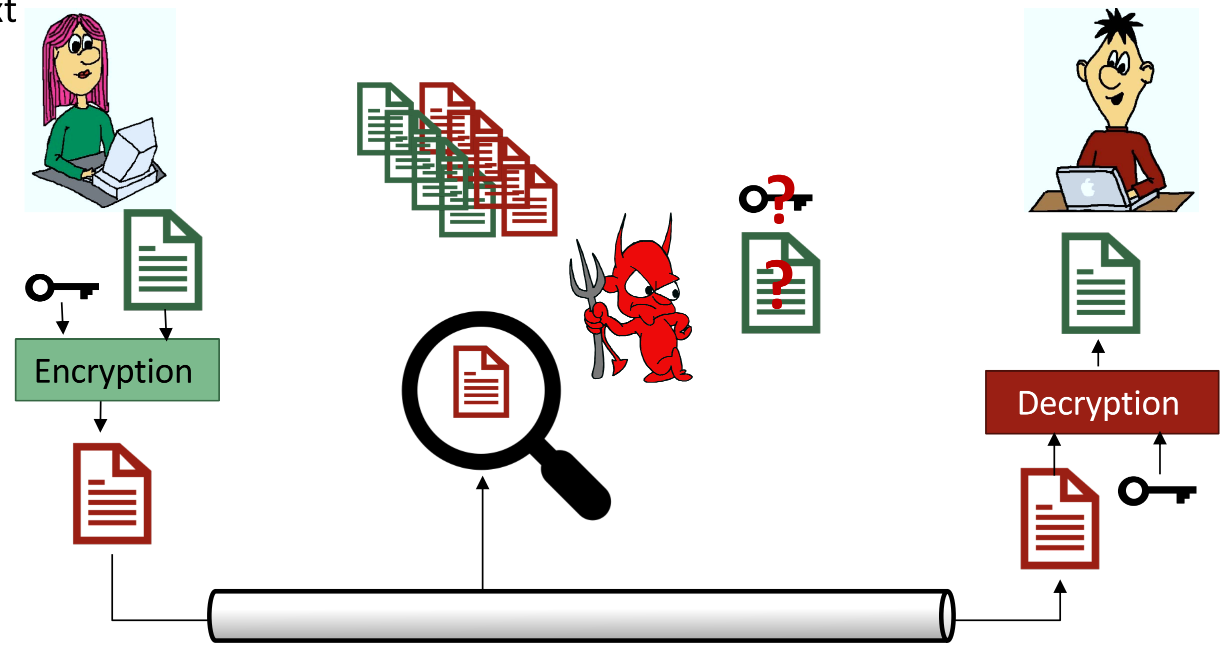


Illustration of Known-Plaintext Attack

- **Attacker observes ciphertext and has access to one or more pair of plaintext and ciphertext**
  - E.g., as he is able to guess plaintext for some ciphertexts
    - E.g., due to Bob's reaction on receiving the ciphertext
  - Tries to recover key and/or plaintext

Example:

- Substitution cipher vulnerable to a known plaintext attack
- One pair of plaintext / ciphertext sufficient to break (part of) the key



Example: Exhaustive Key Search

- **Try out all possible keys from the key space**
  - Ciphertext-only setting
    - Try out each key to decrypt the ciphertext and check if resulting plaintext "makes sense"
    - Only works if valid plaintexts are recognizable for the attacker
  - Known-plaintext setting
    - Try out each key to decrypt the ciphertext
    - Check if it decrypts to the known plaintext
- **Ciphertext-only setting is more difficult for the attacker**
  - Consequently: being secure against a ciphertext-only attack is easier to achieve
- **Security in a chosen-ciphertext setting is hardest to achieve**

Difficulty of Known-Plaintext Brute Force Attack

- **Difficulty of exhaustive key search is proportional to the key size**
  - On average attacker will have to try out $\frac{|\mathcal{K}|}{2}$ keys
- **And proportional to the resources of the attacker**

Key Size (bits)	Number of Alternative Keys	Time required at 1 decryption/µs	Time required at 10^6 decryptions/µs
32	$2^{32} = 4.3 \times 10^9$	2^{31} µs = 35.8 minutes	2.15 milliseconds
56	$2^{56} = 7.2 \times 10^{16}$	2^{55} µs = 1142 years	10.01 hours
128	$2^{128} = 3.4 \times 10^{38}$	2^{127} µs = 5.4×10^{24} years	5.4×10^{18} years
168	$2^{168} = 3.7 \times 10^{50}$	2^{167} µs = 5.9×10^{36} years	5.9×10^{30} years
26 characters (permutation)	$26! = 4 \times 10^{26}$	2×10^{26} µs = 6.4×10^{12} years	6.4×10^6 years

Other Attack Strategies besides Brute Force and Frequency Analysis

- **Time-memory trade-off**
  - Can be used to accelerate known-plaintext attacks
  - Exploits a trade-off between time, memory and key space size

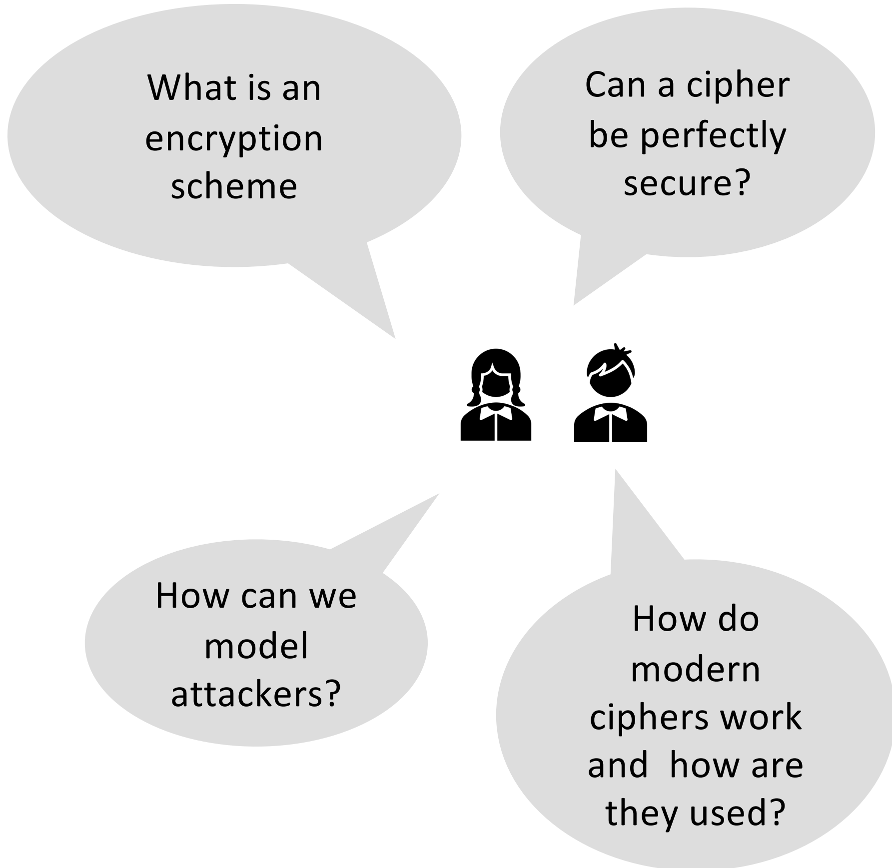
- **Differential cryptoanalysis**
  - Chosen-plaintext attack
  - Attacker tries to recover key using known differences between plaintexts and comparing them to the differences in the ciphertexts

- **Algebraic attacks**
  - Reduces breaking a cipher to solving a system of linear equations with the key bits as unknowns
  - Can work very well in a known-plaintext setting

- **Related key attacks**
  - Chosen-plaintext attack
  - Assumes attacker has access to chosen plaintext encrypted with keys
  - Attacker knows relations between keys

Overview

- **Introduction**
  - Intuition
  - Formal definition
  - Historic examples
- **Computational Security**
  - Attacker models
    - Knowledge
    - Goal
    - Strategy
- **Perfect Secrecy**
  - Definition
  - Shanon’s theorem
  - One-time-pad
- **Practical Schemes**
  - Stream ciphers
  - Block ciphers
  - Modes of encryption



Stream Ciphers

- **The one-time pad $C = P \oplus K$ is perfectly secure**
  - If the key is chosen uniformly at random for each P
- **Idea of stream cipher**
  - Replace K with pseudo-random bit-generator PRBG
    - Seed PRBG with "truly random" key $\boldsymbol{K}$
    - Include a fresh initialization vector $\boldsymbol{IV}$ for each $\boldsymbol{P}$
  - Encryption/Decryption very fast
    - Key stream can be pre-generated
- **The PRNG should be cryptographically secure**
  - We typically cannot proof that a PRBG is cryptographically secure, we assume it is if no attack is known

Stream cipher

For each plaintext **P** select a fresh **IV** and set

$\boldsymbol{C} = \boldsymbol{E_K}(\boldsymbol{P}) = \boldsymbol{IV} \parallel \boldsymbol{P} \oplus$ **PRBG**$(\boldsymbol{IV}, \boldsymbol{K})$.

PRBG(IV,K) is also referred to as **key stream**

The same key $\boldsymbol{K}$ is used for multiple plaintexts

A PRBG is said to be cryptographically secure iff

There is no polynomial-time algorithm which on input of the first k bits of the output of PRBG can predict the next bit with probability > ½ . I.e., it passes the next bit test.

General Stream Cipher Weakness

- **If the IV is ever reused with the same key**
  - Stream ciphers are vulnerable to a known-plaintext attack
- **Why?**
  - Assume attacker known P_1, C_1
    - As $C_1 = E_K(P_1) = IV \parallel P_1 \oplus \text{PRBG}(IV, K)$ attacker knows IV and $\text{PRBG}(IV, K)$
    - Thus, if IV and K are reused to encrypt P_2, and attacker observes C_2
    - Then he can decrypt P_2 by $C_2 \oplus IV \parallel \text{PRBG}(IV, K) = 0 \parallel P_2$
- **As, e.g., been used to attack the security architecture WPA2 for WLAN**
  - Known as KRACK attack

Examples for Stream Ciphers

- **Well-known insecure stream ciphers**
  - RC4
    - Before its break used in WLAN, TLS, …
  - A5/1, A5/2
    - Supported by GSM (2G mobile networks)
  - E0
    - Supported by old Bluetooth versions
  - …

- **Well-known (yet) unbroken stream ciphers**
  - SNOW 3G
    - Supported by 3G/LTE/5G networks
  - CHACHA20
    - Supported by TLS, IPSec,…
  - Unbroken Block ciphers in CTR Mode
    - Supported by LTE/5G networks
    - Supported by TLS, IPSec,…
  - …

- **Any cipher that only provides computational security can break at any point in time**
  - We need to be prepared and always ensure that we can easily switch from one cipher to another

Block Ciphers

- **Operate on plaintext blocks of a specific length**
  - Called the block length **b** $\in \mathbb{N}$ of the cipher
  - Plaintext space $\mathcal{P} = \{0,1\}^b$ and ciphertext space $\mathcal{C} = \{0,1\}^b$
  - For each key K in the key space $\mathcal{K} = \{0,1\}^k$, $E_K : \mathcal{P} \rightarrow \mathcal{C}$

- **Typically need to be used in a specific mode of encryption**
  - Specifies how plaintexts of length $> b$ bits are encrypted

Later in this Chapter

Examples for Block Ciphers

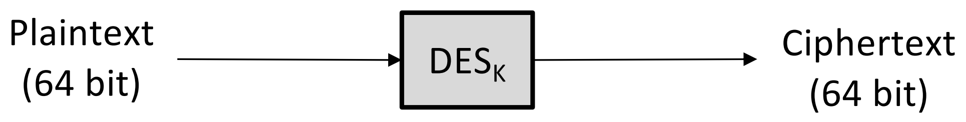
- **Well-known insecure block ciphers**
  - DES
    - Before its break used in IPSec, TLS, …
  - IDEA
  - …

- **Well-known (yet) unbroken block ciphers**
  - KASUMI
    - Supported by 3G/LTE/5G networks
  - AES
    - Supported by TLS, IPSec,…
  - Camellia
    - Supported by TLS
  - …

- **Any cipher can break at any point in time**
  - We need to be prepared and always ensure that we can easily switch from one cipher to another

Example Block Cipher: DES

- **Published in 1977 by the National Bureau of Standards***
  - Designed by IBM and the NSA
- **Uses a 64-bit key K and a block length of 64 bit**
  - But: 8 bits of the key are used as parity bits
- **Effective key size is 56 bits**



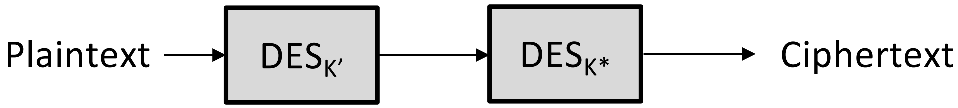
*** called National Institute of Standards and Technology (NIST) since 1988**

Security of DES

- **January 13th, 1999: DES key broken within 22 hours and 15 minutes**
  - In a contest sponsored by RSA Labs using
  - Brute force key search using
  - the Electronic Frontier Foundation's Deep Crack custom DES cracker ...
  - … and the idle CPU time of around 100,000 computers
- **Since then, DES is considered insecure**
- **Biggest weakness still is the key length of 56 bits only!**

First Proposed Fix: 2DES

- **First idea to increase the key size of DES**
  - Use DES twice with two independently chosen keys



- **Problem: this does not double the key size!**

Meet in the middle attack on 2DES

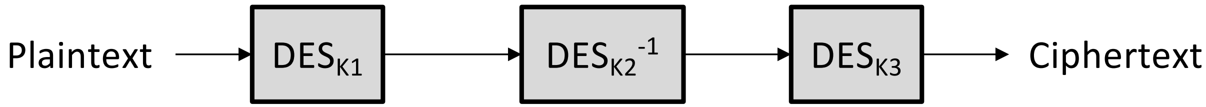
- Assume attacker has access to (P, C), where $C = \text{DES}_{K*}(\text{DES}_{K'}(P))$
- Attacker can encrypt P with any possible key (2^{56} DES operations)
  - And thus, create lookup table $E_K(P) = Z_K$ for $K \in \{0,1\}^{56}$ of intermediate ciphertext
- Attacker can decrypt C with all possible keys (at most 2^{56} DES operations)
  - And compute $D_K(C) = X_K,\ K \in \{0,1\}^{56}$ until $X_{K_i} = Z_{K_j}$ is found in the lookup table
- Then $K_j = K'$ and $K_i = K^*$ with high probability

Complexity of the attack:

- $2 \cdot 2^{56} = 2^{57}$ DES operations
- Effective key size only increased by one!

3DES = ”Triple DES”

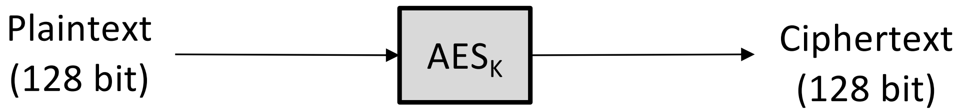
- **Use DES three times in a row**



- **Variants**
  - 3-key DES: K1, K2, and K3 are pairwise different
    - Provides an effective key size of 112 bit according to NIST
  - 2-key DES: K1 = K3
    - Provides and effective key size of 80 bit according to NIST
  - Both variants use encryption with K1, decryption with K2 and encryption with K3
    - Setting K1=K2=K3 this allows 3DES-only capable senders to communicate with DES-only capable receivers

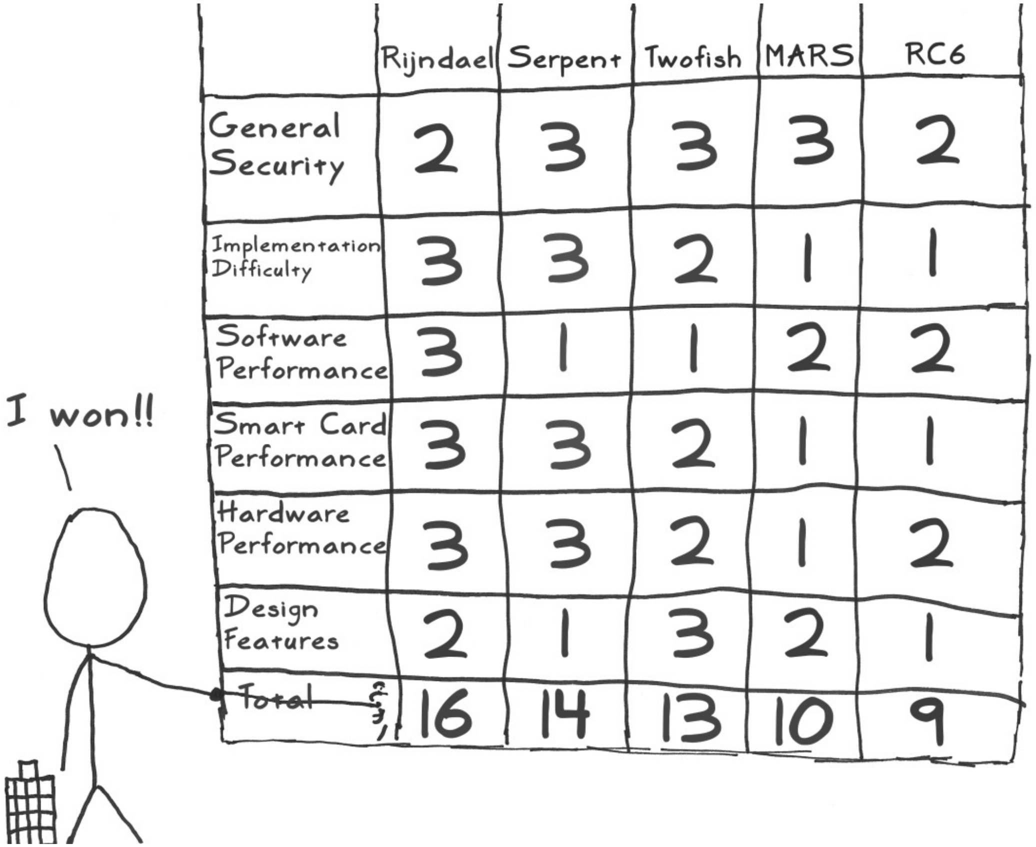
The Advanced Encryption Standard (AES)

- **Goals of the NIST Call for AES**
  - More secure than 3DES
  - More efficient than 3DES
  - Support different key lengths
    - 128, 192, and 256 bit
  - The block length of the cipher is 128 bit
    - Regardless of the key length



- **Timeline of AES Selection**
  - Jan. 1997 NIST-call published
  - Aug. 1998: 15 candidates presented
    - Cast-256, Crypton, DEAL, DFC, E2, Frog, HPC, Loki97, Magenta, MARS, RC6, Rijndael, SAFER+, Serpent, Twofish
    - Broken shortly afterwards (or during presentation)
      - DEAL, Frog, HPC, Loki97, Magenta
  - Aug. 1999 finalists announced
    - MARS, RC6, Rijndael, Serpent, Twofish
  - Oct. 2000 **Rijndael selected as AES**
  - Nov. 2001 AES standardized in FIPS 197

Selection Criteria



Taken from http://www.moserware.com/2009/09/stick-figure-guide-to-advanced.html

Structure of AES

- **AES operates in rounds**
  - Input and output of each round represented as 4x4 byte matrices

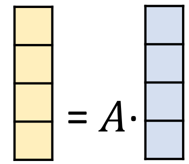
i_0	i_4	i_8	i_{12}
i_1	i_5	i_9	i_{13}
i_2	i_6	i_{10}	i_{14}
i_3	i_7	i_{11}	i_{15}

s_{00}	s_{01}	s_{02}	s_{03}
s_{10}	s_{11}	s_{12}	s_{13}
s_{20}	s_{21}	s_{22}	s_{23}
s_{30}	s_{31}	s_{32}	s_{33}

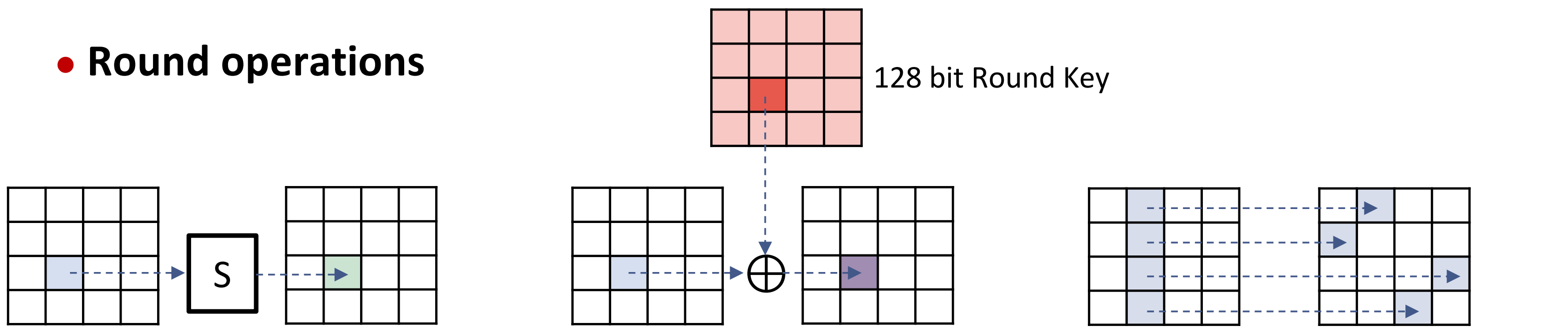
o_0	o_4	o_8	o_{12}
o_1	o_5	o_9	o_{13}
o_2	o_6	o_{10}	o_{14}
o_3	o_7	o_{11}	o_{15}

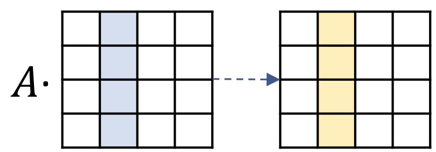
$$A = \begin{bmatrix} 02 & 03 & 01 & 01 \\ 01 & 02 & 03 & 01 \\ 01 & 01 & 02 & 03 \\ 03 & 01 & 01 & 02 \end{bmatrix}$$

Multiplication in GF(2^8)



- **Round operations**





Substitute Byte (SB) · Round Key Addition (KA) · Shift Row (SR) · Mix Column (MC)

Reminder: Multiplication in GF(2^8) with $x^8 + x^4 + x^3 + x + 1$ as irreducible Polynomial

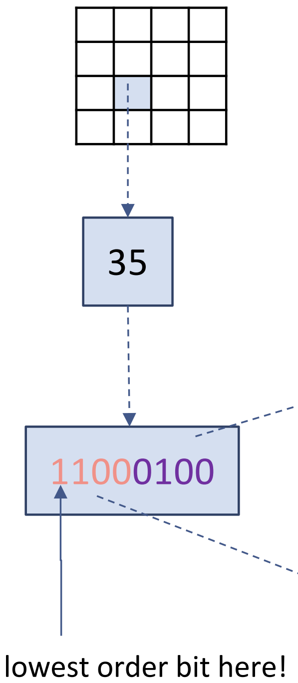
- **For example, (in hex notation) 57 • 83 = c1 in GF(2^8) because**
  - $57 = 01010111 \backsimeq x^6 + x^4 + x^2 + x + 1$
  - $83 = 10000011 \backsimeq x^7 + x + 1$
  - $(x^6 + x^4 + x^2 + x + 1)\,(x^7 + x + 1) \;=\; x^{13} + x^{11} + x^9 + x^8 + x^7 + x^7 + x^5 + x^3 + x^2 + x + \mathrm{x}^6 + \mathrm{x}^4 + \mathrm{x}^2 + \mathrm{x} + 1 = x^{13} + x^{11} + x^9 + x^8 + x^6 + x^5 + x^4 + x^3 + 1$
  - $x^{13} + x^{11} + x^9 + x^8 + x^6 + x^5 + x^4 + x^3 + 1$ modulo $x^8 + x^4 + x^3 + x + 1 \;=\; x^7 + x^6 + 1$

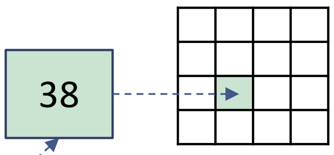
    $\backsimeq 1100\ 0001$

    $= c1$

Substitute Byte (SB)

Each byte $b = b_0b_1b_2b_3b_4b_5b_6b_7$ is replaced with byte in **S** in **column** $b_0b_1b_2b_3$ and **row** $b_4b_5b_6b_7$





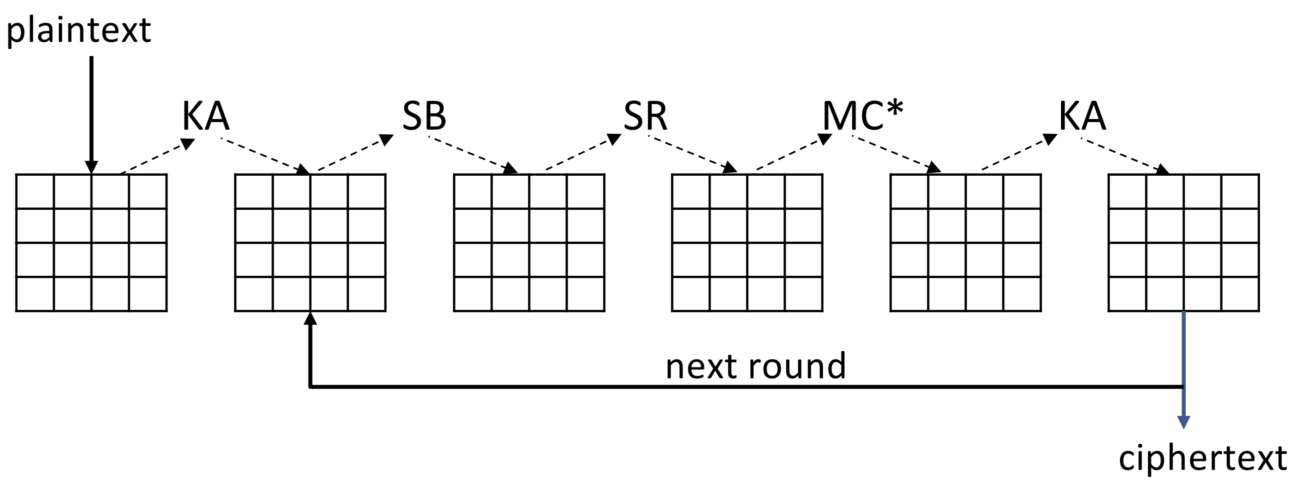
Column 4

Row 2

S-Box

99	124	119	123	242	107	111	197	48	1	103	43	254	215	171	118
202	130	201	125	250	89	71	240	173	212	162	175	156	164	114	192
183	253	147	38	54	63	247	204	52	165	229	241	113	216	49	21
4	199	35	195	24	150	5	154	7	18	128	226	235	39	178	117
9	131	44	26	27	110	90	160	82	59	214	179	41	227	47	132
83	209	0	237	32	252	177	91	106	203	190	57	74	76	88	207
208	239	170	251	67	77	51	133	69	249	2	127	80	60	159	168
81	163	64	143	146	157	56	245	188	182	218	33	16	255	243	210
205	12	19	236	95	151	68	23	196	167	126	61	100	93	25	115
96	129	79	220	34	42	144	136	70	238	184	20	222	94	11	219
224	50	58	10	73	6	36	92	194	211	172	98	145	149	228	121
231	200	55	109	141	213	78	169	108	86	244	234	101	122	174	8
186	120	37	46	28	166	180	198	232	221	116	31	75	189	139	138
112	62	181	102	72	3	246	14	97	53	87	185	134	193	29	158
225	248	152	17	105	217	142	148	155	30	135	233	206	85	40	223
140	161	137	13	191	230	66	104	65	153	45	15	176	84	187	22

AES Operation Overall



MC*: no mix column operation in the last round

- **The round key is always 128 bit key**
  - Different for each round, generated from the secret key
- **Number of rounds depends on the key size**
  - 128 bit key: 10 rounds 192 bit key: 12 rounds 256 bit key: 14 rounds

Modes of Encryption

- **Block ciphers of block length b**
  - Allow us to encrypt a plaintext $\boldsymbol{P}$ of $\boldsymbol{b}$ bit
  - How can we encrypt longer plaintexts?

- **Mode of encryption**
  - Let $\boldsymbol{P} = \boldsymbol{P_1} \parallel \boldsymbol{P_2} \parallel \boldsymbol{P_3} \parallel \boldsymbol{P_4} \parallel \cdots \parallel \boldsymbol{P_n}$ with $\boldsymbol{P_i} \in \{\boldsymbol{0}, \boldsymbol{1}\}^{\boldsymbol{b}}$ for $i = 1, \dots, n-1$ and $\boldsymbol{P_n} \in \{\boldsymbol{0}, \boldsymbol{1}\}^{\boldsymbol{l}}$ for some $0 < l \leqq b$
  - A mode of encryption specifies how to encrypt plaintext $\boldsymbol{P}$ based on a $\mathbf{b}$ bit block cipher $\boldsymbol{E_K}(\cdot)$

- **Modes we cover here**
  - Electronic Code Book (ECB) mode
  - Cipher Block Chaining (CBC) mode
  - Counter Mode (CTR)
- **Modes we may cover in exercises**
  - Cipher Feedback Mode (CFB)
  - Output Feedback Mode (OFB)
- **AEAD Modes** **Chapter 3**
  - Authenticated Encryption with Associated Data (AEAD) Modes
    - E.g., Gallois Counter Mode (GCM)

Electronic Codebook Mode (ECB)

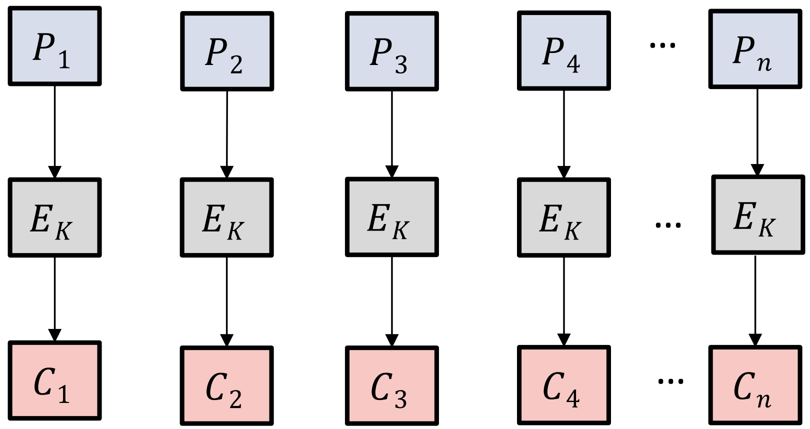
ECB Mode

Encryption: $C_i = E_k(P_i)$ for $i = 1, \dots, n$

Decryption: $P_i = D_k(C_i)$ for $i = 1, \dots, n$

Requires **padding** of P_n to b bit

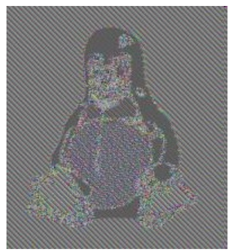
Illustration of encryption in ECB Mode



- **Problem**
  - Same P_i leads to same C_i
  - Thus, patterns in plaintext lead to patterns in ciphertext
  - **ECB mode should not be used!**

Plaintext

ECB-encrypted



Cipher Block Chaining Mode (CBC)

CBC Mode

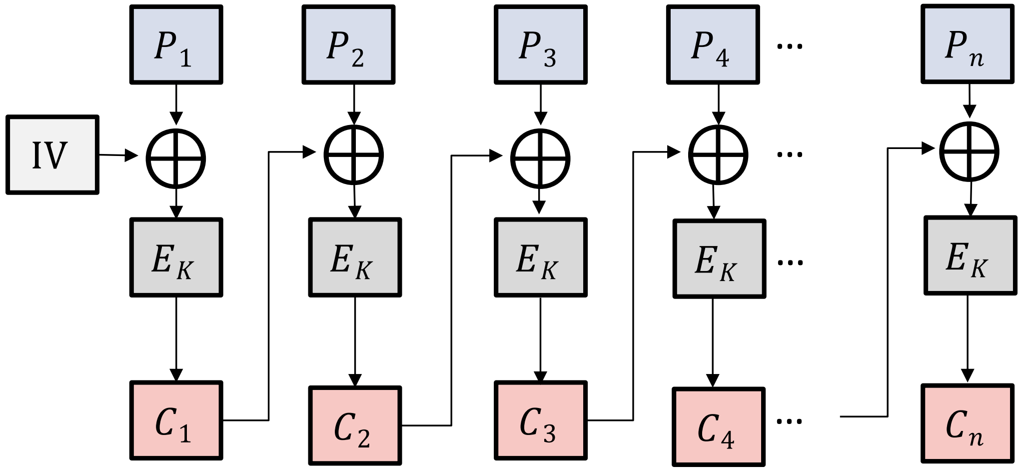
$\mathrm{IV} := \mathrm{C}_0$

Encryption: $\mathrm{C_i} = \mathrm{E_k}(\mathrm{P_i} \oplus \mathrm{C_{i-1}})$ for $i = 1, \ldots, n$

Decryption: $\mathrm{P_i} = \mathrm{D_k}(\mathrm{C_i}) \oplus \mathrm{C_{i-1}}$ for $i = 1, \ldots, n$

Requires **padding** of $\mathrm{P_n}$ to b bit

Illustration of encryption in CBC Mode



- **Requires a fresh IV for each plaintext to encrypt**
  - If same IV is reused on P and P^*
    - then C_1 and C_1^* reveal, whether $P_1 = P_1^*$
  - Is **vulnerable** to a so-called **padding-oracle attack**

Should not be used anymore

Counter Mode (CTR)

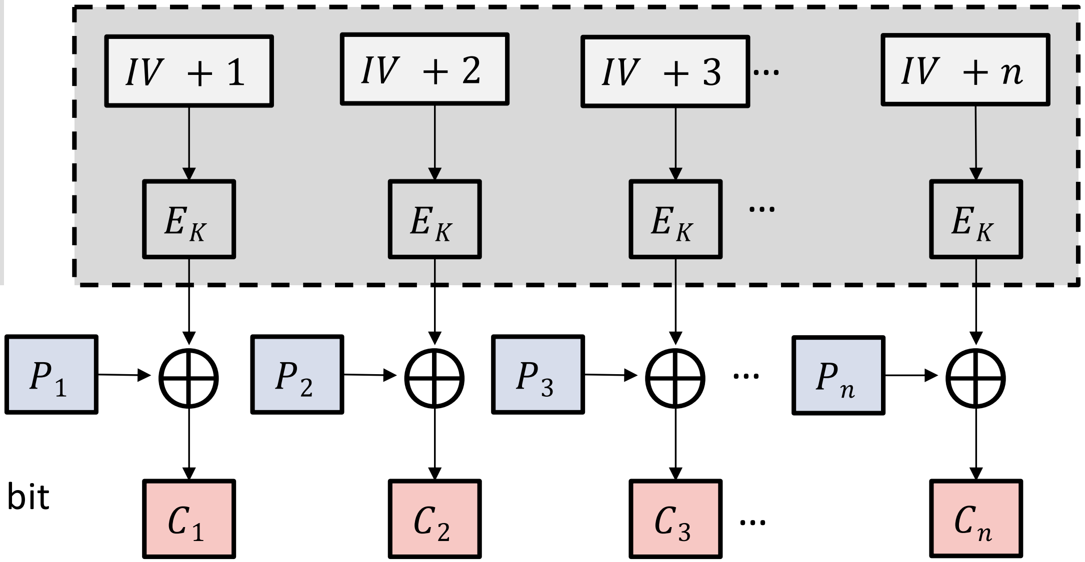
CTR Mode

IV public, fresh for each plaintext

Encryption: $\mathrm{C_i} = \mathrm{E_k}(\mathrm{IV} + i) \oplus P_i$ for $i = 1, \dots, n$

Decryption: $\mathrm{P_i} = C_i \oplus \mathrm{E_k}(\mathrm{IV} + i)$for $i = 1, \dots, n$

Illustration of encryption in CTR Mode



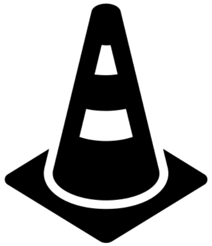
Properties of CTR Mode

- CTR Mode **does not require padding** of $\boldsymbol{P_n}$ to $\boldsymbol{b}$ bit
- Ciphertext is of the same size as plaintext
- CTR Modes **turns a block cipher** into a **stream cipher**
- CTR mode encryption and decryption can be parallelized

Summary

- **Symmetric Encryption Schemes provide confidentiality**
  - Require a secret key shared between the communicating entities
- **Perfect secrecy can be obtained by the one-time-pad**
  - Requires key chosen uniformly at random and as long as the plaintext for each plaintext
  - Impractical to use in many situations
- **Practical encryption schemes only provide computational security**
  - Can in theory always be broken with a brute force attack in a known plaintext setting
    - Require long keys to make brute force attack practically impossible
- **Different attacker models make different assumptions with respect to**
  - The knowledge of the attacker (ciphertext-only, known plaintext,…)
  - The goal of the attacker (plaintext recovery, key recovery)
  - The approach the attacker takes (brute force, frequency analysis, differential analysis…)

Summary



- **Practical symmetric encryption schemes can be divided into**
  - Stream ciphers, e.g., ChaCha20
  - Block ciphers, e.g., AES
- **Stream ciphers encrypt a plaintext by xoring it with a key stream**
  - Key stream is generated by
    - a (longer term) secret key that is reused for multiple plaintext
    - and fresh IV for each plaintext to encrypt
  - Should never reuse IVs with the same key
- **Block ciphers require the use of a mode of encryption**
  - Specifies how to encrypt plaintext that are longer than one block-length of the block cipher
  - These modes have a strong influence of the security of the encryption scheme
    - Used with in an insecure mode, a secure block cipher may become insecure
  - The effective key size of a block cipher cannot be doubled by applying the cipher twice

References

- **More details on symmetric encryption**
  - Johannes Buchmann, Einführung in die Kryptographie, 6. Auflage, Springer Verlag 2016
    - Kapitel 3 - Kapitel 6
  - W. Stallings, Cryptography and Network Security: Principles and Practice, 8^{th} edition, Pearson 2022
    - Chapters 3, 4, 6, and 7

- **Standard Documents**
  - FIPS 197: Advanced Encryption Standard
    - https://nvlpubs.nist.gov/nistpubs/FIPS/NIST.FIPS.197-upd1.pdf
  - FIPS 46-3: Data Encryption Standards (DES)
    - https://csrc.nist.gov/files/pubs/fips/46-3/final/docs/fips46-3.pdf