Information Security

Introduction and History

Srđan Čapkun

What is this course about?

- Part I: Fundamentals of cryptography (Srdjan Capkun)
  - Symmetric, asymmetric encryption, hash functions, randomness, signatures, …
- Part II: Protocols and Systems (David Basin)

- Goal: understand concepts, read + understand literature, evaluate solutions

Administrative

- Assistants:
  - Part 1

    Dr. Kari Kostiainen, Dr. AbdelRahman Abdou, Karl Wüst
  - Part 2
    - Dr. Lucca Hirschi, Sven Hammann, Bhargav Bhatt

- Web:
  - http://www.infsec.ethz.ch/education/ss2018/infsec.html

Exercises (Part 1)

- Two weekly exercise sessions (starting next week)
  - Wed, 15-18 **HG F 26.5**
  - Thu, 15-18 **ML F 36**

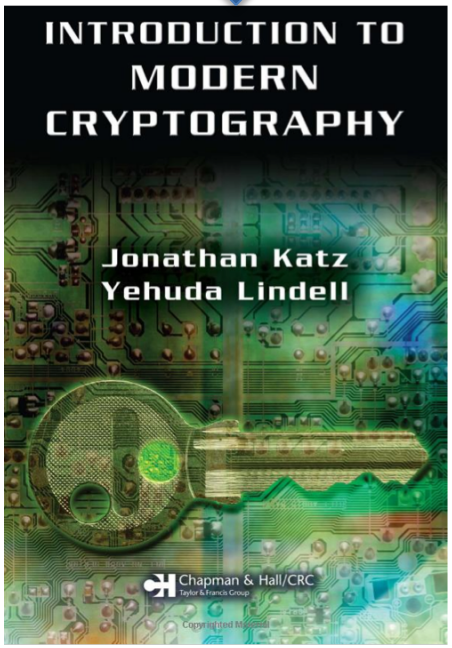
- Exercises are posted on the course webpage on Friday
  - **Try to solve them first on your own**
- Master solutions posted on Monday
  - Check them before the exercise session
- Exercise sessions
  - We explain master solutions, Q&A, discuss alternatives
- Exercises are **not graded**, but solving them is recommended
  - Gives good idea of what to expect in exam ☺

Exercises (Part 2)

- Same exercise sessions (Wed and Thu) but different process…
- Check the course web page for details!

Literature (1st par

Caveat Emptor
- Sometimes quite involving
- Do not get distracted by turgid notation

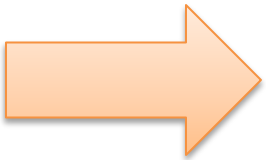


- (Optional) Textbook
  - "Introduction to Modern Cryptography"
    J. Katz and Y. Lindell
    Chapman and Hall/CRC, ISBN 1584885513
  - Slides are partially(!) based on this book
- (Additional) Reading
  - "Cryptography: Theory and Practice"
    D. R. Stinson
    Chapman and Hall/CRC, ISBN 1584882069
  - "Applied Cryptography: Protocols, Algorithms, and Source Code in C"
    Bruce Schneier
    Wiley, ISBN 0471117099
  - …there is many others, e.g., Stallings, Vanstone, Anderson etc.

Disclaimer

- Slides are (partially) recycled from Stefan Dziembowski's and Erik-Oliver Blass .

Plan

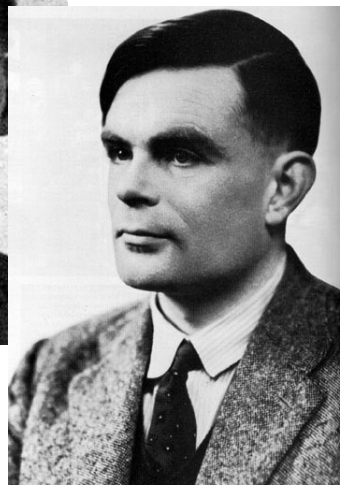


1. Introduction
2. Historical ciphers
3. Information-theoretic security
4. Computational security

Historical cryptography

cryptography ≈ encryption

main applications: **military and diplomacy**



Modern cryptography

cryptography = much more than encryption!

tossing a coin over a telephone

mental poker

signature schemes

key agreement

electronic auctions

e-cash

electronic voting

zero-knowledge

public-key cryptography

multiparty-computations

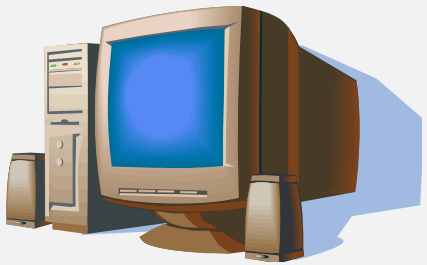
sevenites

now

What happened in the seventies?

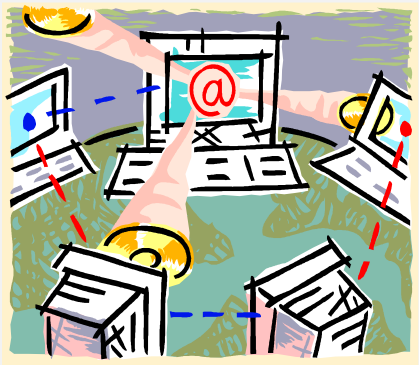
Technology

afforadable hardware



Demand

companies and individuals start to do business electronically



Theory

the computational complexity theory is born

this allows researchers to reason about security in a **formal way**.

Cryptography

In the past:

the **art** of encrypting messages (mostly for the military applications).

Now:

the **science** of securing digital communication and transactions (encryption, authentication, digital signatures, e-cash, auctions, etc..)

Three components of the course

- practical aspects
- mathematical foundations
- new horizons

Practical aspects

- symmetric encryption: block ciphers (DES, AES) and stream ciphers (RC4)
- hash functions (MD5, SHA1,…), message authentication (CBC-MAC)
- elements of number theory
- asymetric encrypion (RSA, ElGamal, Rabin,...)
- signature schemes (RSA, ElGamal,…)

Mathematical foundations

- What makes us believe that the protocols are secure?
- Can we formally define “security”?
- Can security be proven?
- Is there an “unbreakable” cipher?

New horizons

- Advanced cryptographic protocols, such as:
- zero-knowledge
- multiparty computations
- private information retrieval

Example: Bitcoin

- What does Bitcoin depend on?
  - Security of Digital signatures
  - Pre-image resistance of Hash function
  - Future: Zero-Knowledge Proofs of Knowledge

Terminology

constructing secure systems

breaking the systems

Cryptology = cryptography + cryptanalysis

This convention is **slightly artificial** and often ignored.

Common usage:

“cryptanalysis of X” = “breaking X”

Common abbreviation: **“crypto”**

Cryptography – general picture

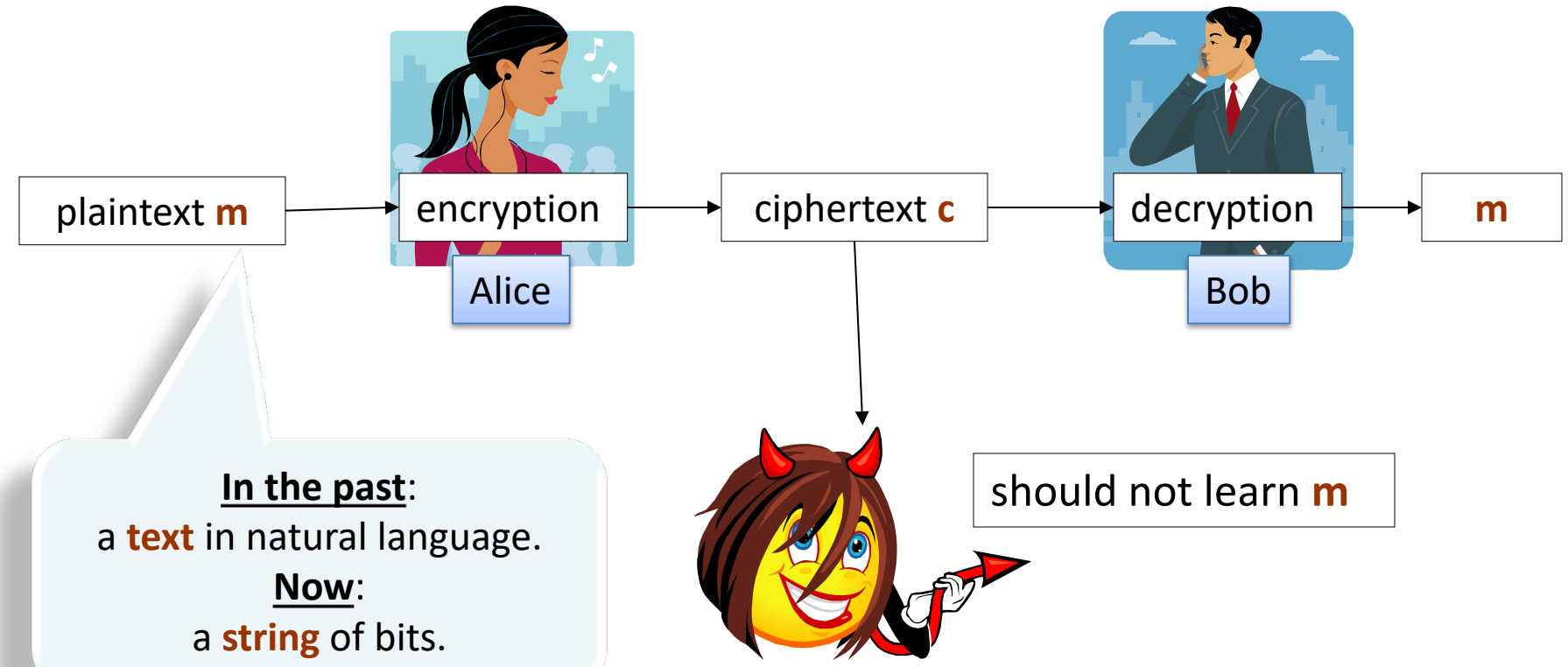
plan of the course:

	encryption	**integrity**
private key	**1** private key encryption	**2** private key authentication
public key	**3** public key encryption	**4** signatures

5 advanced cryptographic protocols

Encryption schemes (a very general picture)

Encryption scheme (cipher) = encryption & decryption



Art vs. science

In the past:

lack of precise definitions, ad-hoc design, usually insecure – an arms race.

Today:

formal definitions, systematic design, very secure constructions.

Provable security

We want to construct schemes that are **provably secure**.

But...

- **why** do we want to do it? Why is it necessary?
- **how** to define it?
- and is it **possible** to achieve it?

Provable security – the motivation

In many areas of computer science formal proofs are **not essential**.

For example, instead of proving that an algorithm is efficient, we can just simulate it on a *"typical* input".

In **cryptography** it's **not true**, because

there cannot exist an experimental proof that a scheme is secure.

Why?

Because a notion of a

"*typical* adversary"

does not make sense.

Security definitions are useful also because they allow us to construct schemes in a modular way...

Kerckhoffs' principle

Auguste Kerckhoffs (1883):

The enemy knows the system

The cipher should remain secure even if **the adversary knows the specification of the cipher.**

The only thing that is **secret** is a

short key **k**

that is **usually chosen uniformly at random**

Kerckhoffs' principle – the motivation

1. In commercial products it is unrealistic to assume that the design details remain secret (**reverse-engineering!**)
2. Short keys are easier to **protect**, **generate** and **replaced**.
3. The design details can be discussed and **analyzed in public**.

Not respecting this principle
=
``**security by obscurity**”.

Kerckhoffs' principle – maybe needs to be refined

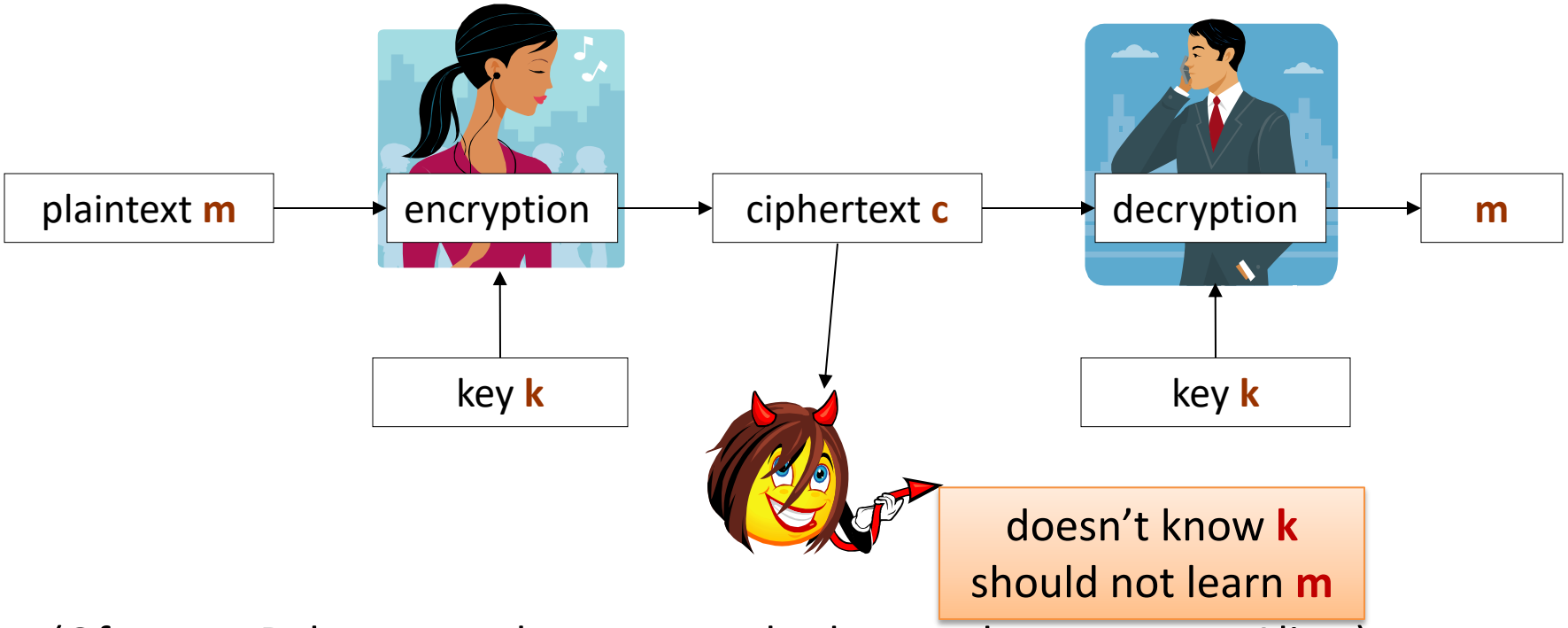
What if the manufacturing process does not allow the production of identical circuits?

Physically Unclonable Functions (PUFs)

Chip1(X) != Chip2(X)

The attacker **cannot replicate the design (even if she holds the chip)**

A more refined picture



(Of course Bob can use the same method to send messages to Alice.)
(That's why it's called the **symmetric setting**)

Let us assume that **k** is chosen uniformly random

A mathematical view

$\mathcal{K}$ – **key** space

$\mathcal{M}$ – **plaintext** space

C - **ciphertext** space

An **encryption scheme** is a pair **(Enc,Dec,(Gen?))**, where

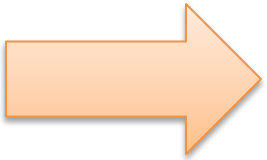
- **Enc :** $\mathcal{K} \times \mathcal{M} \rightarrow C$ is an **encryption** algorithm,
- **Dec :** $\mathcal{K} \times C \rightarrow \mathcal{M}$ is an **decryption** algorithm.

We will sometimes write $\mathbf{Enc_k(m)}$ and $\mathbf{Dec_k(c)}$ instead of **Enc(k,m)** and **Dec(k,c)**.

Correctness

for every **k** we should have $\mathbf{Dec_k(Enc_k(m)) = m}$.

Plan



1. Introduction
2. Historical ciphers
3. Information-theoretic security
4. Computational security

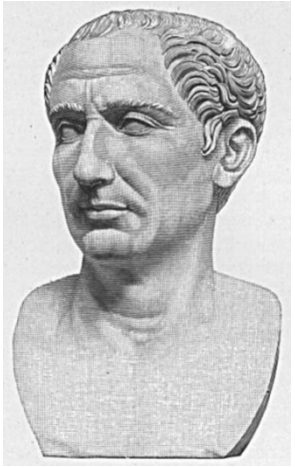
Caesar's Shift cipher

$\mathcal{M}$ = words over alphabet {A,…,Z} ≈ {0,…,25}

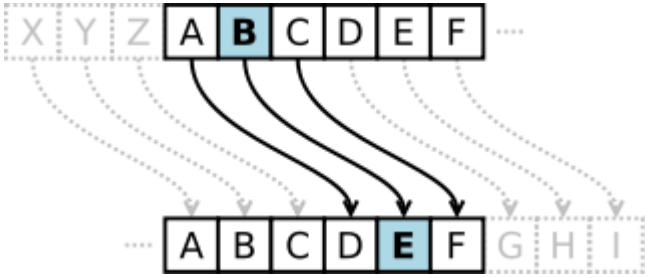
$\mathcal{K}$ = {0,…,25}

$Enc_k(m_0,\ldots,m_n) = (k+m_0 \bmod 26,\ldots, k+m_n \bmod 26)$

$Dec_k(c_0,\ldots,c_n) = (c_0-k \bmod 26,\ldots, c_n-k \bmod 26)$



Caesar: **k = 3**



[De Vita Caesaurum, Divus Iulius, 110 C.E.]

Security of the shift cipher

How to break the shift cipher?

Check all possible keys!

Let **c** be a ciphertext.

For every **$k \in \{0,\ldots,25\}$** check if **$Dec_k(c)$** "makes sense".

Most probably only one such **k** exists.

Thus **$Dec_k(c)$** is the message.

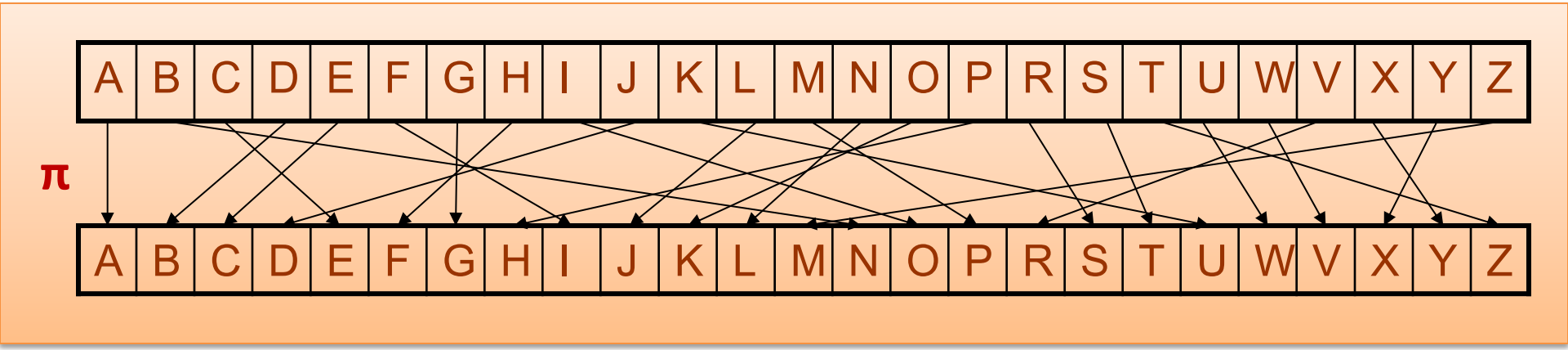
This is called a **brute force attack**.

Moral: the key space needs to be large!

Generalization: Substitution cipher

$\mathcal{M}$ = words over alphabet {A,…,Z} ≈ {0,…,25}

$\mathcal{K}$ = a set of permutations of {0,…,25}



$Enc_\pi(m_0,\ldots,m_n) = (\pi(m_0),\ldots, \pi(m_n))$

$Dec_\pi(c_0,\ldots,c_n) = (\pi^{-1}(c_0),\ldots, \pi^{-1}(c_n))$

How to break the substitution cipher?

- Use **statistical patterns** of the language
  - **For example**: the **frequency tables**
  - Ciphertext has same frequency distribution as plaintext
  - Computer frequency table for ciphertext and correlate with plaintext
- (Obviously useful against Caesar…)

Letter	Frequency
E	0.127
T	0.097
I	0.075
A	0.073
O	0.068
N	0.067
S	0.067
R	0.064
H	0.049
C	0.045
L	0.040
D	0.031
P	0.030
Y	0.027
U	0.024
M	0.024
F	0.021
B	0.017
G	0.016
W	0.013
V	0.008
K	0.008
X	0.005
Q	0.002
Z	0.001
J	0.001

Figure 7 - Frequency Table

Blaise de Vigenère
(1523 - 1596)

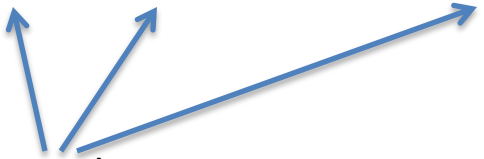
Vigenere Cipher

- Problem of "mono-alphabetic" substitution: same letter always encrypted identically
- Vigenere's idea: "poly-alphabetic" substitution
  - key is multiple letters $k = k_1, k_2, \ldots, k_d$
  - effectively multiple Caesar ciphers: i^{th} letter of key specifies i^{th} shift to use
  - repeat from start after d letters in message
  - $Enc_k(m_1,\ldots,m_n) =$
    $m_1+k_1 \bmod 25,\ldots, m_d+k_d \bmod 25, m_{d+1}+k_1 \bmod 25, \ldots$
  - Decrypt accordingly

Example of Vigenère Cipher

- Write plaintext out
- Write keyword repeated
- Shift plaintext letter by cipher key
- Example using keyword “deceptive”

```
key:          de cep tivedecept ived eceptive
plaintext:    we are discovered save yourself
ciphertext:   ZI CVT WQNGRZGVTW AVZH CQYGLMGJ
```



Result:
Letters are encrypted differently each time

Cryptanalysis: Vigenere's Cipher (1/2)

- Observation: if d is known, use d-times frequency attack on substitution cipher (complexity?)
  - How to find d? Kasiski&Babbage, Friedman
- Idea: two identical l-grams m_i, m_{i+1},…, m_{i+l} are encrypted to **same** ciphertext c_i, c_{i+1},…, c_{i+l}, if their distance is n•d (n ε N)
  - Find identical l-grams, compute distances d_1, d_2, …
  - n•d = gcd (d_1, d_2, …)

```
key:          de cep tivedecept ived eceptive
plaintext:    we are discovered save yourself
ciphertext: ZI CVT WQNGRZGVTW AVZH CQYGLMGJ
```

Cryptanalysis: Vigenere's Cipher (2/2)

- Alternative: Index of coincidence
  - Probability that any two randomly chosen letters are equal
- Observation
  - ciphertext letters with distance d are Shift encrypted
  - For $\{c_i, c_{i+d}, c_{i+2d} \ldots\}$, I_c equals the plaintext's I_c (English language)
- For j=1,2,…
  - Select $C=\{c_1, c_{1+j}, c_{1+2j}, \ldots\}$
  - For C compute individual letter probabilities q_i
  - Check $\sum_{i=0}^{25} q_i^2 \overset{?}{\approx} 0.065 \approx \sum_{i=0}^{25} p_i^2$

Occurrences of letter i in m

$$I_c(m) = \frac{\sum_{i=0}^{25} n_i \cdot (n_i - 1)}{n \cdot (n-1)} \approx \sum_{i=0}^{25} p_i^2$$

Plaintext of length n

For large n

Relative letter probability

$$\sum_{i=0}^{25} p_i^2 \approx 0.065$$

Other famous historical ciphers

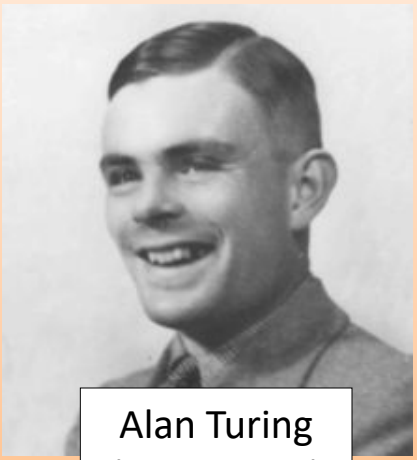
Scytale
(7th century BC)

Transposition Cipher
– permute m

Leon Battista Alberti
(1404 – 1472)

Enigma

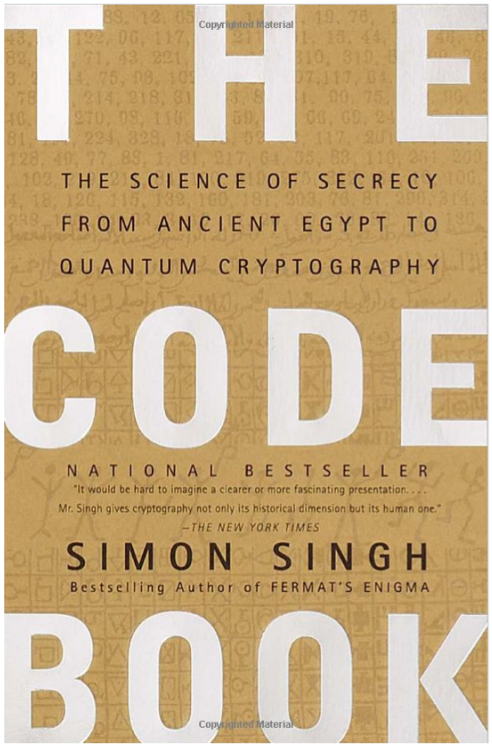
Marian Rejewski
(1905 - 1980)



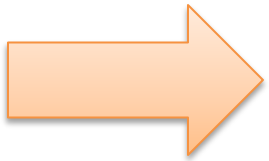
Alan Turing
(1912-1954)

Interesting Read

- "The Code Book: The Science of Secrecy from Ancient Egypt to Quantum Cryptography"
  Simon Singh
  Anchor, ISBN 0385495323



Plan



1. Introduction
2. Historical ciphers
3. Information-theoretic security
4. Computational security

Why formalization?

In contemporary cryptography, ciphers are designed in a **systematic way**.

Main goals:

1. define meaning of security
2. construct schemes that are “provably secure”

Defining “security of an encryption scheme” is not trivial.

consider the following experiment

(**m** – a message)

1. the key **K** is chosen uniformly at random
2. **C := $Enc_K(m)$** is given to the adversary

how to define security

?

Idea 1

(**m** – a message)

1. the key **K** is chosen uniformly at random
2. **C := $Enc_K(m)$** is given to the adversary

An idea

"The adversary should not be able to compute **K**."

A problem

the encryption scheme that "doesn't encrypt":

$Enc_K(m) = m$

satisfies this definition!

Idea 2

(**m** – a message)

1. the key **K** is chosen uniformly at random
2. **C := $Enc_K(m)$** is given to the adversary

An idea

"The adversary should not be able to compute **m**."

A problem

What if the adversary can compute, e.g., the first half of **m**?

m_1	...	$m_{\lvert m \rvert/2}$	?	...	?

Idea 3

(**m** – a message)

1. the key K is chosen uniformly at random
2. **c := $Enc_k(m)$** is given to the adversary

An idea

"The adversary should not learn any information about **m**."

A problem

But he may already have some a priori information about **m**!

For example, he may know that **m** is a sentence in English...

Idea 4

(**m** – a message)

1. the key **K** is chosen randomly
2. **C := $Enc_K(m)$** is given to the adversary

An idea

"The adversary should not learn any additional information about **m**."

This makes much more sense.

But how to formalize it?

Example

Eve knows that m follows a prob. **distribution**:

m :=
- *"I love you"* with prob. **0.1**
- *"I don't love you"* with prob. **0.7**
- *"I hate you"* with prob. **0.2**

c := $Enc_K(m)$

Eve **still** knows that m follows same prob. distribution:

m :=
- *"I love you"* with prob. **0.1**
- *"I don't love you"* with prob. **0.7**
- *"I hate you"* with prob. **0.2**

How to formalize “Idea 4”?

“The adversary should not learn any **additional** information about **m**.”

also called: **information-theoretically** secret or **unconditionally** secret

An encryption scheme is **perfectly secret** if

for random variables **M, C**

and every **m** $\in \mathcal{M}$ and **c** $\in C$ such that **P(C = c) > 0**

$$\mathbf{P(M = m) = P(M = m \mid C = c)}$$

→ **Distributions of random variables M and C are independent**

More Equivalent Definitions

For **M, C** we have that: **M** and **C** are independent

"the distribution of **C** does not depend on **M**"

$P(C = c) = P(C = c \mid M = m)$

for every $\mathbf{m_0}$ and $\mathbf{m_1}$ we have that
$\mathbf{Enc(K,m_0)}$ and $\mathbf{Enc(K,m_1)}$
have the same distribution

A perfectly secret scheme: one-time pad

t – a parameter

$\mathcal{K} = \mathcal{M} = \{0,1\}^t$

component-wise **xor**

Vernam's cipher:

$\mathbf{Enc_k(m) = k \text{ xor } m}$

$\mathbf{Dec_k(c) = k \text{ xor } c}$

Gilbert Vernam (1890 –1960)

Correctness is trivial:

$\mathbf{Dec_k(Enc_k(m)) = k \text{ xor } (k \text{ xor } m)}$

$\mathbf{= m}$

Proof: One-Time-Pad Perfect Secrecy

Perfect secrecy of one time pad is straightforward:
What is the probability for a specific (m, c) pair?

Arbitrary, but fixed

$$P(C = c \mid M = m) \;?!$$

$$= P(M \text{ xor } K = c \mid M = m)$$

$$= P(m \text{ xor } K = c) = P(K = m \text{ xor } c) = 2^{-t}$$

$$= P(C=c \mid M=m_0) = P(C=c \mid M=m_1)$$

Keys are chosen uniformly from random. Regardless of M's&C's distributions

For all m_0, m_1

Observation

One time pad can be **generalized** as follows.

Let **(G,~)** be a group. Let $\mathcal{K} = \mathcal{M} = C = G$.

The following is a perfectly secret encryption scheme:

- **Enc(k,m) = m ~ k**
- **Dec(k,c) = c ~ k^{-1}**

Why the one-time pad is not practical?

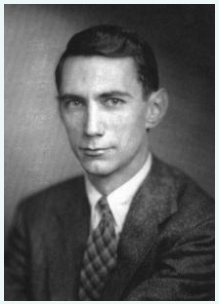
1. **The key has to be as long as the message.**
2. **The key cannot be reused**

This is because:

$$\mathrm{Enc}_k(m_0) \text{ xor } \mathrm{Enc}_k(m_1) = (k \text{ xor } m_0) \text{ xor } (k \text{ xor } m_1)$$

$$= m_0 \text{ xor } m_1$$

Question: can there be a more efficient **perfectly secret** cipher? No… ☹



Theorem (Shannon 1949)

("One time-pad is optimal in the class of perfectly secret schemes.")

Claim: In every perfectly secret encryption scheme

$$\mathbf{Enc} : \mathcal{K} \times \mathcal{M} \to \mathcal{C}\ ,\ \mathbf{Dec} : \mathcal{K} \times \mathcal{C} \to \mathcal{M}$$

we have $|\mathcal{K}| \geq |\mathcal{M}|$.

Proof

Perfect secrecy implies that the distribution of **Enc(K,m)** does not depend on **m.** Hence for every m_0 and m_1 we have

$$\{Enc(k,m_0)\}_{k \in \mathcal{K}} = \{Enc(k',m_1)\}_{k' \in \mathcal{K}}$$

denote this set with $\mathcal{C}'$

Two keys could map to same ciphertext

Observation: $|\mathcal{K}| \geq |\mathcal{C}'|$.

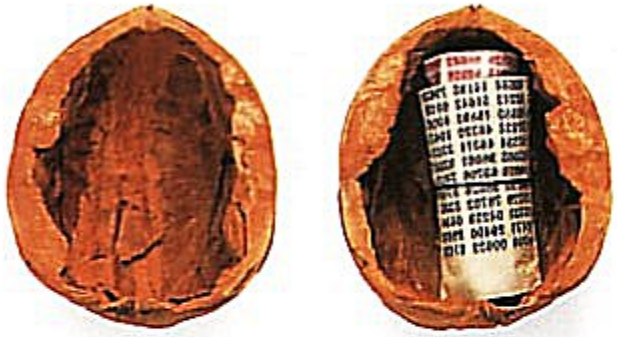
Fact: we always have that $|\mathcal{C}'| \geq |\mathcal{M}|$.
This is because for every **k** we have that
$Enc_k : \mathcal{M} \to \mathcal{C}'$ is an injection
(otherwise we wouldn't be able to decrypt).

$|\mathcal{K}| \geq |\mathcal{M}|$

Practicality?

Generally, the **one-time pad** is **not very practical**, since:

- the key has to be as long as the **total** length of the encrypted messages,
- it is hard to generate truly random strings.



a **KGB** one-time pad hidden in a walnut shell

However, it is sometimes used (e.g. in the **military applications**), because of the following advantages:

- **perfect secrecy**,
- short messages can be encrypted using **pencil and paper** .

In the 1960s the Americans and the Soviets established a hotline that was encrypted using the one-time pad.(**additional advantage**: they didn't need to share their secret encryption methods)

Venona project (1946 – 1980)

Ethel and Julius Rosenberg

National Security Agency decrypted **KGB** messages that were transmitted in the 1940s.

That was possible because the Soviets reused the keys in the one-time pad scheme.

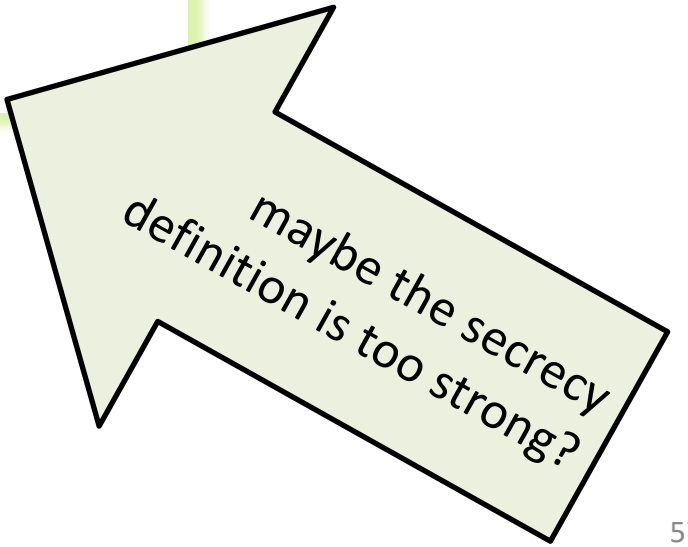
One-Time-Pad: Conclusion

We constructed a perfectly secret encryption scheme

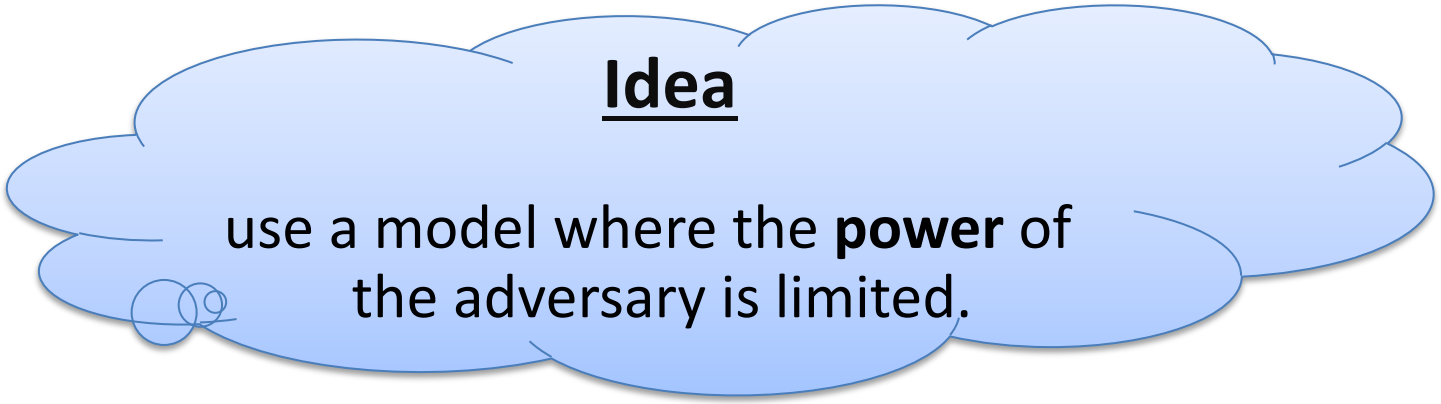
Our scheme has certain drawbacks ($|\mathcal{K}| \geq |\mathcal{M}|$).

But, by Shannon's theorem, this is unavoidable.

Can we go home and relax?



What to do?



How?

Classical (computationally-secure) cryptography:
bound his computational power.

Alternative options:
quantum cryptography, bounded-storage model,...
(not too practical)

Quantum cryptography

Stephen Wiesner (1970s), Charles H. Bennett and Gilles Brassard (1984)

Quantum indeterminacy: quantum states **cannot** be measured without disturbing the original state → detectable by Bob.

Hence **Eve** cannot read the bits in an unnoticeable way.

Good technique to exchange key (for One-Time-Pad).

Quantum cryptography

Advantage:

security is based on the laws of quantum physics

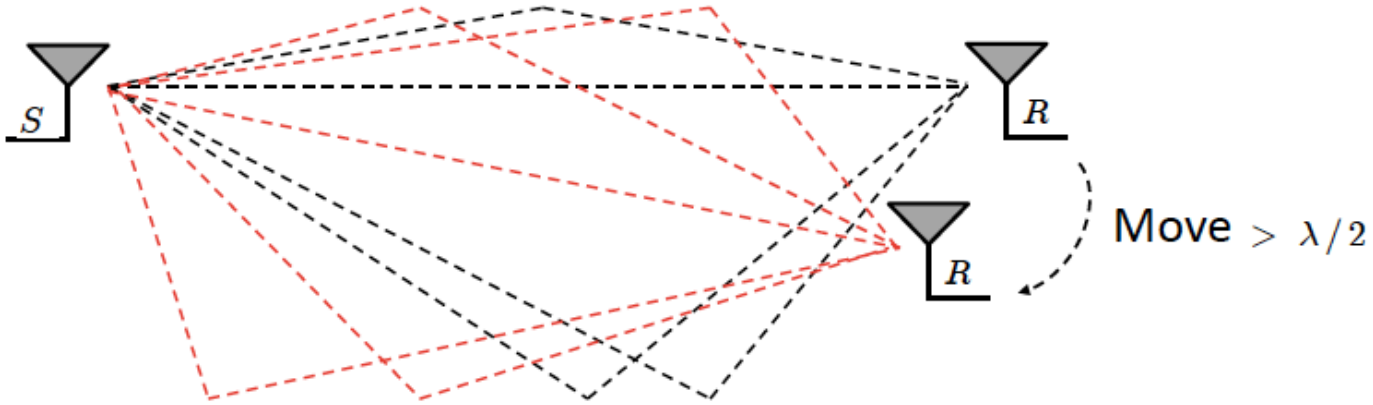
Disadvantage:

needs a dedicated equipment, expensive.
authentication?

Practicality?

Currently: successful transmissions for distances of length around **150 km**.
Commercial products are available.

Wireless Phy-Layer Key Establishment



In a complex, multipath-rich environment, channels exhibit time-varying, stochastic and ***reciprocal*** fading.

For receivers that are > λ/2 away, channels are not correlated.

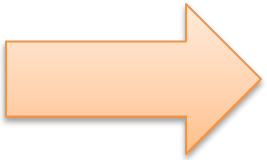
=> the channel between S and R will be 'random' and ***will not be known to the attacker*** but S and R will be able to measure it

=> a natural "wiretap" channel

Authentication?

Plan

1. Introduction
2. Historical ciphers
3. Information-theoretic security
4. Computational security



How to reason about the bounded computing power?

perfect secrecy:
M and **$Enc_K(M)$**
are independent

"unconditionally" –
to strong?

It is enough to require that

M and **$Enc_K(M)$**

are independent

"***from the point of view of a computationally-limited adversary with high probability***''.

How can this be formalized?

We will use the **complexity theory**!

Why is this good?

Eve is computationally-bounded

We construct schemes that in **principle can be broken,** if the adversary has huge computing power (and/or a lot of luck).

For example, the adversary will be able to break the scheme by enumerating all possible secret keys.

(this is called a "**brute force attack**")

Computationally-bounded adversary

Eve is computationally-bounded

But what does it mean?

Ideas:

1. "She has can use at most 1000
   Intel Core 2 Extreme X6800 Dual Core Processors
   for at most **100** years..."

2. "She can buy equipment worth **1 million euro** and use it for **30** years..".

it's hard to reason
formally about it

A better idea (Concrete Approach)

"The adversary has access to a **Turing Machine** that can make at most 2^{90} steps."

More generally, we could have definitions of a type:

"a system **X is (t,ε)-secure** if every **Turing Machine**

that operates in time **t**

can break it with probability at most **ε**."

This would be quite precise, **but...**

- What is t?
- What is ε?
- What is a Turing machine? (Different tapes, random access memory,...)

What to do? Asymptotic Approach

Idea:

t steps of a Turing Machine → "**efficient computation**"

ε → a value "**very small (close to zero)**".

How to formalize it?

Use basic complexity **asymptotic notation**!
…t and ε will depend on parameter n.

Efficiently computable?

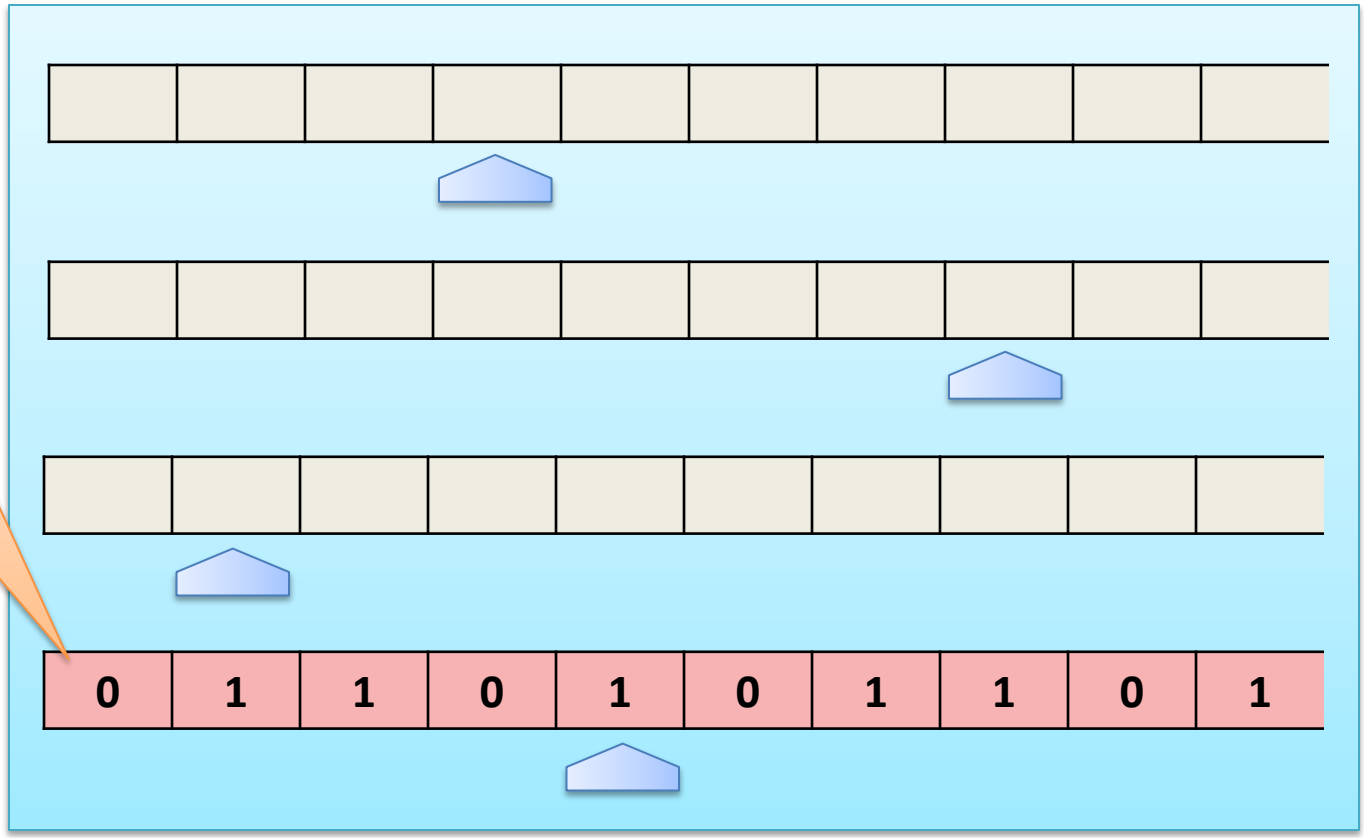
"efficiently computable" = "polynomial-time computable on a **Probabilistic Turing Machine**"

that is: running in time $O(n^c)$ (for some c)

Here we assume that the **poly-time Turing Machines** are the right model for the real-life computation ("Church-Turing Assumption").

Probabilistic Turing Machines

A standard ("deterministic") Turing Machine has some number of tapes:



A **probabilistic** Turing Machine has an additional tape with random bits.

There is a difference between a Probabilistic Turing Machine and a Non-Deterministic Turing Machine!

Some notation

If **M** is a Turing Machine then

M(X)

is a **random variable** denoting the **output** of **M**
assuming that
the content of the random tape was chosen
uniformly at random.

Even more notation

$$Y \leftarrow M(X)$$

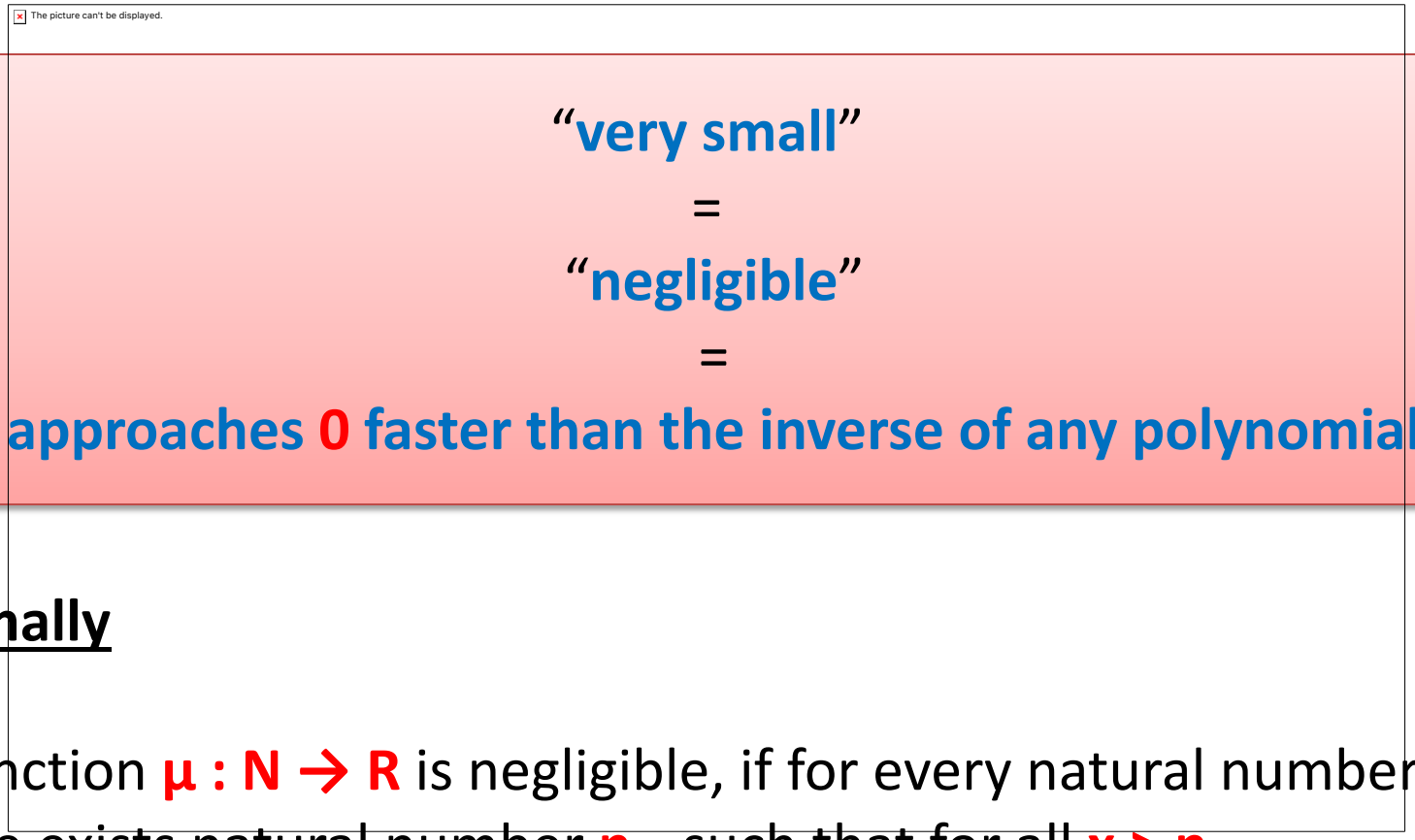
means that the variable **Y** takes the value that **M** outputs (probabilistically) on input **X** (assuming the random input is chosen uniformly).

If $\mathcal{A}$ is a set then

$$Y \leftarrow \mathcal{A}$$

means that **Y** is chosen uniformly at random from the set $\mathcal{A}$.

How small is “very small”?



“very small”

=

“negligible”

=

approaches 0 faster than the inverse of any polynomial

Formally

A function **μ : N → R** is negligible, if for every natural number **c** there exists natural number n_0, such that for all $x > n_0$

$$|\mu(x)| < \frac{1}{x^c}$$

Negligible or not?

no $f(n) := \frac{1}{n^2}$

yes $f(n) := 2^{-n}$

yes $f(n) := 2^{-\sqrt{n}}$

yes $f(n) := n^{-\log n}$

no $f(n) := \frac{1}{n^{1000}}$

Security parameter required

Typically, we will say that a **scheme X is secure** if

$\forall$

(probabilistic) polynomial-time Turing Machine M

P (M breaks the scheme X) is **negligible**

Asymptotic notation: The terms "**negligible**" and "**polynomial**" make sense only if **X** (and the **adversary**) take an additional input $\mathbf{1^n}$ called a **security parameter.**

In other words: we consider an infinite sequence

X(1),X(2),...

of schemes.

Security Parameter: Example

security parameter **n** = the length of the secret key **k**

in other words: **k** is always a random element of $\{0,1\}^n$

The adversary can always **guess k** with probability 2^{-n}.

This probability **is negligible**.

He can also **enumerate all possible keys k** in time 2^n.
(the "brute force" attack)

This time is exponential (not negligible).

Is this the right approach?

Advantages

All types of **Turing Machines** are “equivalent” up to a “**polynomial reduction**”.

Therefore we do need to specify the details of the model.

Disadvantage

Asymptotic results don’t tell us anything about security of the **concrete systems**.

However

Usually one can prove **formally** an asymptotic result and then argue **informally** that “the constants are reasonable”

(and can be calculated if one really wants).

How to change the security definition?

Reminder **Perfect Secrecy**: C=Enc(K, M) and M are independent.

we will require that $\mathbf{m_0, m_1}$ are "chosen" by a **poly-time adversary**
(reflects: adversary knows distribution M)

An encryption scheme is **perfectly secret,** if for every $\mathbf{m_0, m_1} \in \mathcal{M}$

$$\mathbf{Distribution_{Enc(K,\, m_0)} = Distribution_{Enc(K,\, m_1)}}$$

we will require that no **poly-time adversary** can distinguish distributions
$\mathbf{Enc(K, m_0)}$ from $\mathbf{Enc(K, m_1)}$
with non-negligible probability

A game

(Enc,Dec) – an encryption scheme

security parameter
1^n

adversary
(polynomial-time probabilistic Turing machine)

oracle

chooses m_0, m_1 such that
$|m_0| = |m_1|$

m_0, m_1

1. selects **k** randomly from $\{0,1\}^n$
2. chooses a random **b = 0,1**
3. computes
   $c := Enc(k, m_b)$

has to guess **b**

c

Alternative name: **semantically secure encryptions**
(sometimes we will say: "is **computationally-secure**", if the context is clear)

Security definition:
We say that **(Enc,Dec)** is **indistinguishable encryption,** if any **polynomial time** adversary guesses **b** correctly with probability at most **0.5 + ε(n),** where **ε** is negligible.

Testing the definition

Suppose the adversary can compute **k from Enc(k,m).** Can he win the game?

YES!

Suppose the adversary can compute **some bit of m** from **Enc(k,m).** Can he win the game?

YES!

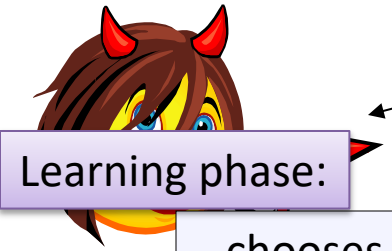
Multiple messages

In real-life applications we need to encrypt **multiple messages with one key**.

The adversary may learn something by looking at ciphertexts $c_1,\ldots,c_t$ of some messages $m_1,\ldots,m_t$.

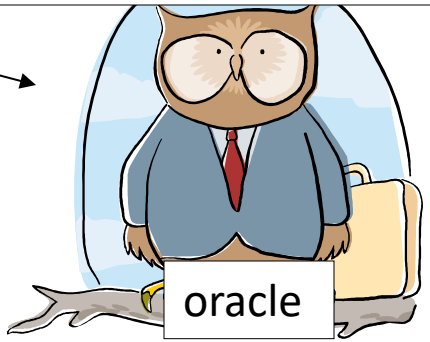
How are these messages chosen?
let's say: the adversary can choose them!
(good tradition: be as pessimistic as possible)
"Chosen Plaintext Attacker"

A chosen-plaintext attack (CPA)



security parameter 1^n

selects random $k \in \{0,1\}^n$



Learning phase:

chooses m'_1

m'_1

$c_1 = Enc(k,m'_1)$

chooses m'_t

m'_t

$c_t = Enc(k,m'_t)$

the interaction continues . . .

challenge phase:

chooses m_0,m_1

m_0,m_1

chooses a random $b = 0,1$

$c = Enc(k,m_b)$

has to guess b

CPA-security

Alternative name: CPA-secure

Security definition

We say that **(Enc,Dec) has indistinguishable encryptions under a chosen-plaintext attack (CPA)** if

every **randomized polynomial time** adversary
guesses **b** correctly
with probability at most **0.5 + ε(n),** where **ε** is negligible.

Observation

Every **CPA-secure** encryption has to be

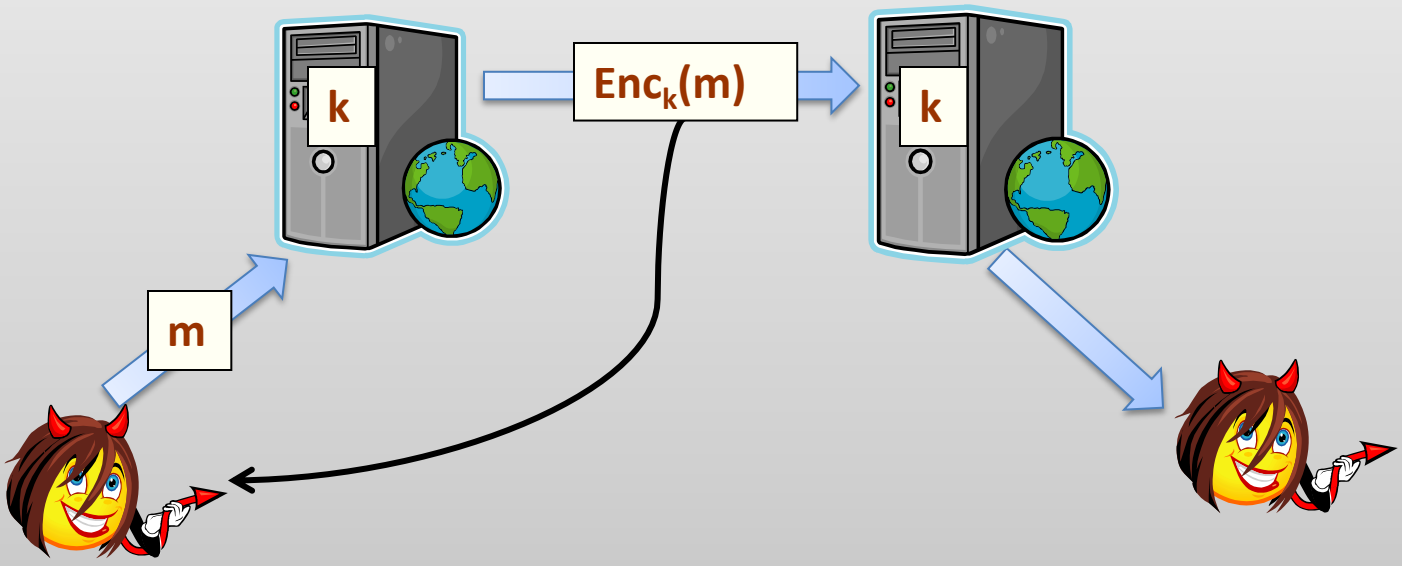
- randomized, or
- "have a state".

CPA in real-life

Q: Aren't we too pessimistic?

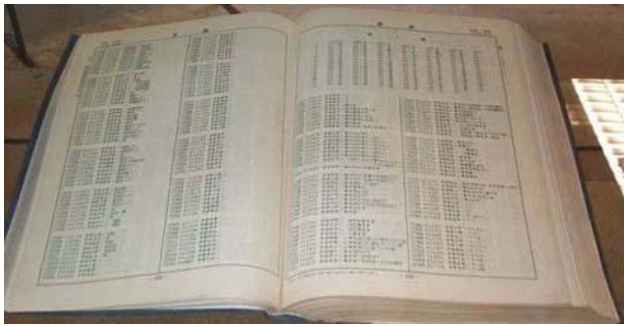
A: **No!** **CPA** can be implemented in practice.

Example: routing



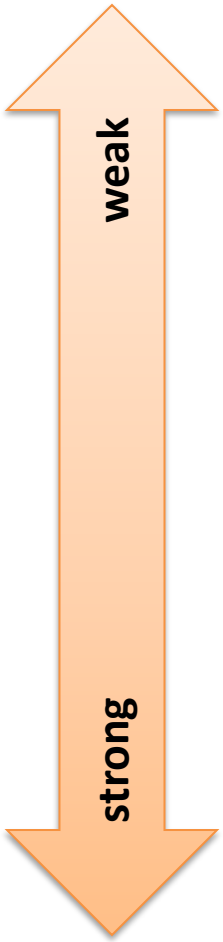
Real-World Chosen-Plaintext-Attack

- US analysis of Japanese naval code “JN-25”
  - World War II, 1942, Pacific battle of Midway
- Substitution cipher
  - Code book to encode words to 5 digit number
  - Encrypt numbers by adding random numbers from 2nd code book
  - Key = page number in 2nd book



- Analysts sent (fake) message in clear: “water shortage in Midway”
  - Intercepted by nearby Japanese
  - Japanese encrypted message with JN-25 and sent to Japan
- Ciphertext for word “Midway” recognized in message for June 4th fleet attack on Midway

Other attacks known in the literature



- **ciphertext-only attack** – the adversary has no information about the plaintext
- **known plaintext attack** – the plaintext are drawn from some distribution that the adversary does not control
- **batch chosen-plaintext attack** – like the **CPA** attack, but the adversary has to choose $\mathbf{m_1,\ldots,m_t}$ at once.

("our" **CPA**-attack is also called the "**adaptive CPA-attack**")

- **chosen ciphertext attack** – we might discuss it later…

End of Class