History of Cryptography

Levente Buttyán

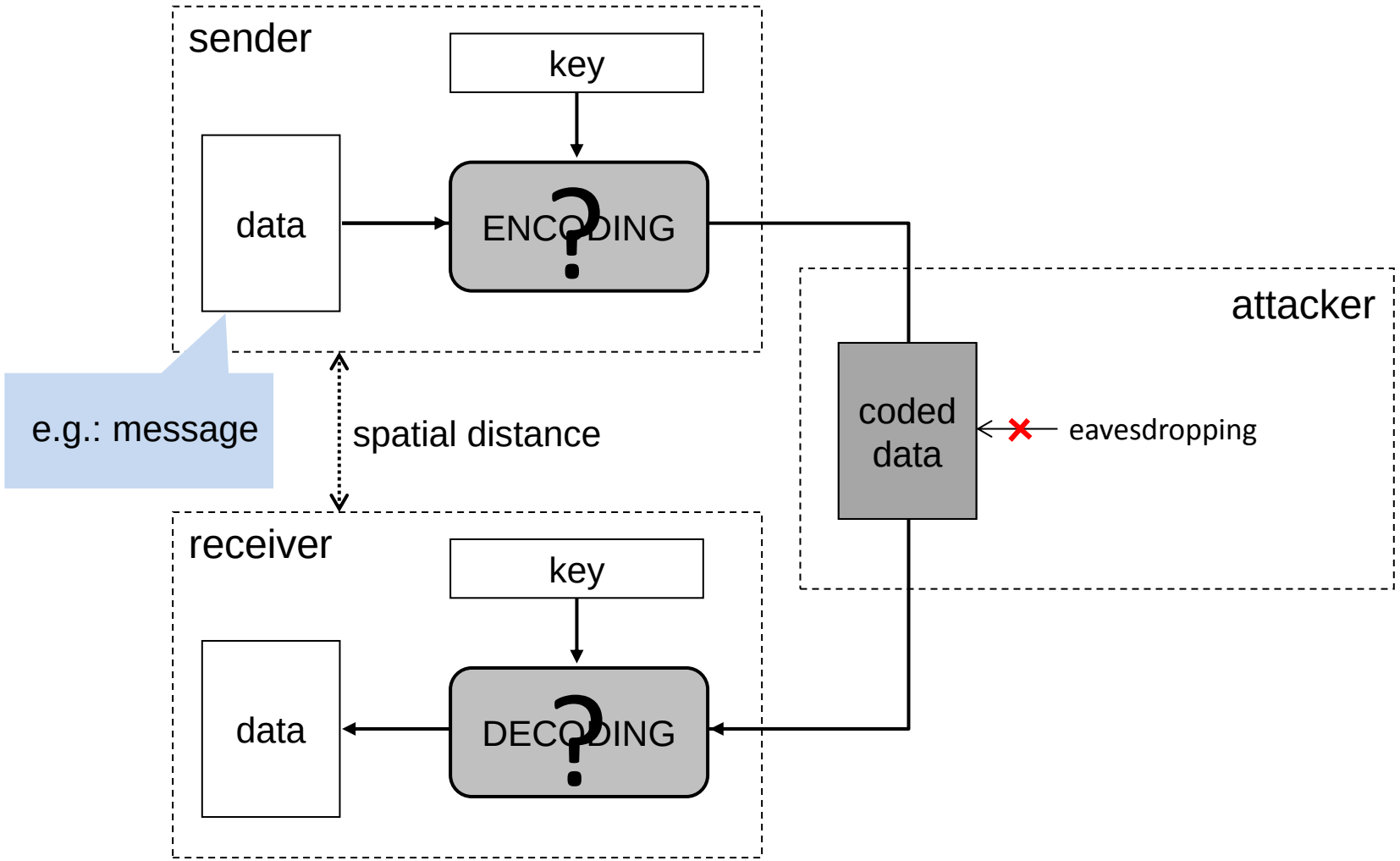
CrySyS Lab, BME

www.crysys.hu

History of crypto in a nutshell

- until the second half of the 20th century:
  - cryptography = encryption, ciphers
  - almost exclusively used in military and diplomacy
- from the second half of the 20th century:
  - cryptography is increasingly used in business applications (banking, electronic funds transfer)
  - besides confidentiality, integrity protection, authentication, and non-repudiation becomes important too
- from the end of the 20th century:
  - cryptography is used in everyday life of people (although they may be unaware of that)
    - » SSL/TLS – secure web transactions
    - » GSM/3G security – subscriber authentiocation, encryption on the air interface
    - » WiFi, Bluetooth, smart cards, ...

Basic model


sender
key
data
ENCODING
?
attacker
coded data
eavesdropping
e.g.: message
spatial distance
receiver
key
data
DECODING
?

Histrorical ciphers

- Skytale from Sparta
- Caesar cipher
- Vigenère cipher (le chifre indéchiffrable)
- German Enigma from WWII

Skytale

- used by the Spartans in the 3rd century BC
- *transposition cipher* (mixes letters of the plaintext)
- encoding and decoding:

- the key is the (diameter of the) rod
- key space is small → easy to break

Caesar cipher

- used by Julius Caesar
- *substitution cipher* (replaces letters of the plaintext)
- each letter is replaced by the letter at some fixed number of positions (e.g., 3) down the alphabet

plain:	**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**
cipher:	**D 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**

example: **CRYPTOGRAPHY → FUBSWRJUDSKB**

- the key is the value of the shift (of the alphabet)
- size of the key space is 26-1 = 25 → easy to break

Monoalphabetic substitution

- generalization of the Caesar cipher
- replacement of letters is determined by a permutation

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

example: **CIPHER → KJNSUV**

- the key is the permutation
- the key space is huge: $26! \sim 1.56 * 2^{88}$

  - » time left until the next ice age ..... 2^{39} sec
  - » time left until the Sun becomes a supernova ..... 2^{55} sec
  - » age of the Earth ..... 2^{55} sec
  - » age of the Universe ..... 2^{59} sec

Breaking monoalphabetic substitutions

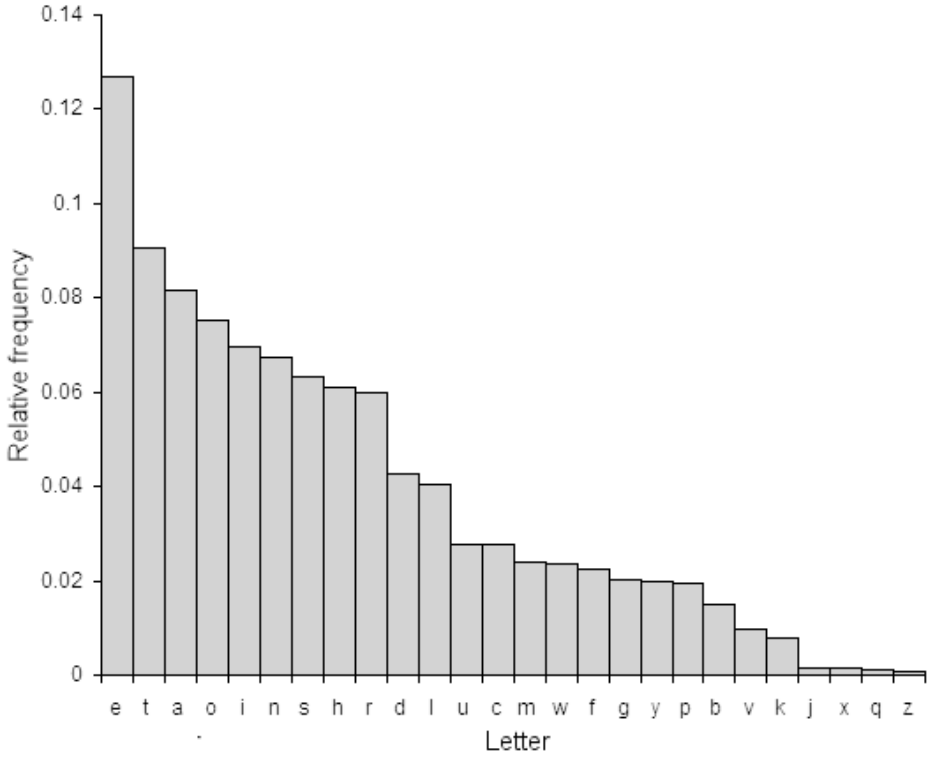
- every language has its own letter statistics
  - there are letters that are more frequently encountered than others

    e.g., in English:

    e → 12.7%, t → 9.1%
  - and letters that are less frequent

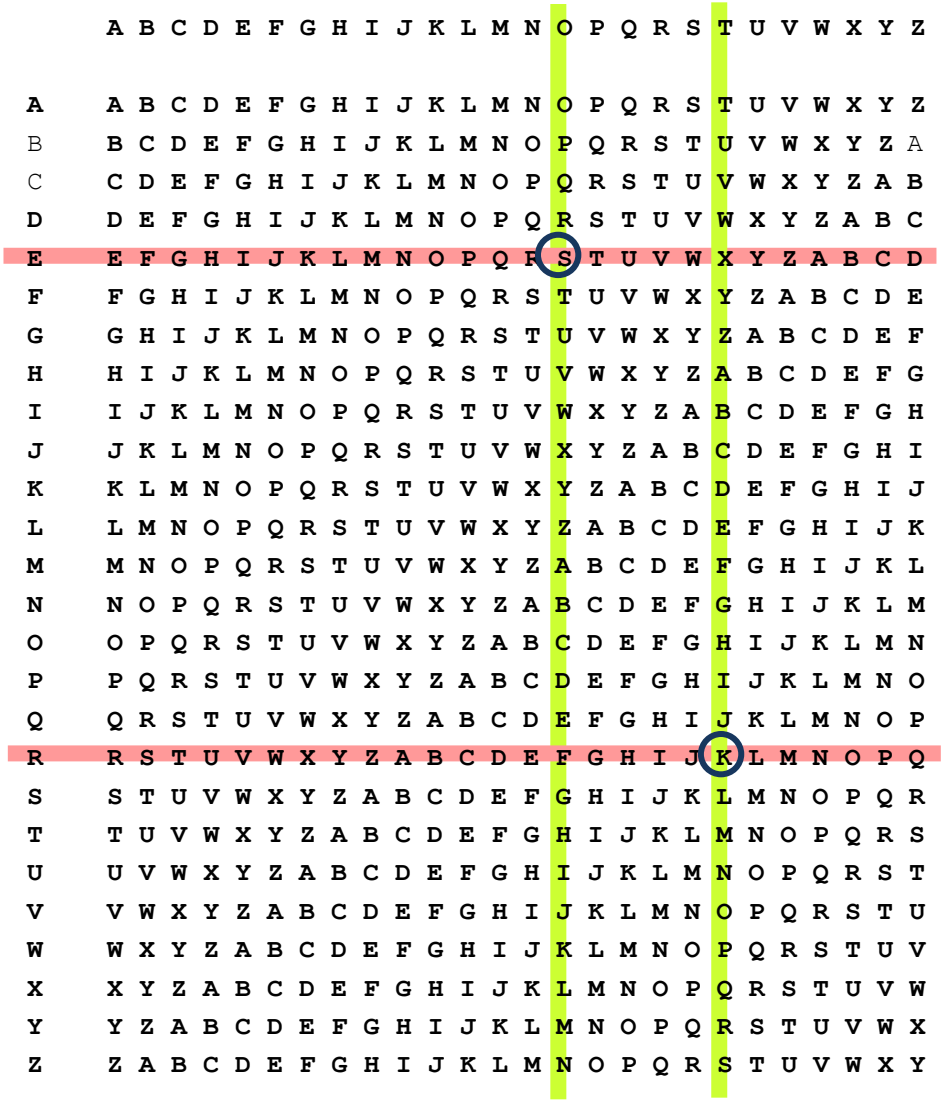
    e.g., in English:

    z → 0.1%, j → 0.2%



- in case of monoalphabetic substitution, the ciphertext preserves the letter statistics of the original plaintext!
  - after decoding the most frequent and least frequent letters, the rest of the text can be figured out much like solving a crossword puzzle

Polyalphabetic substitution (Vigenère)



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

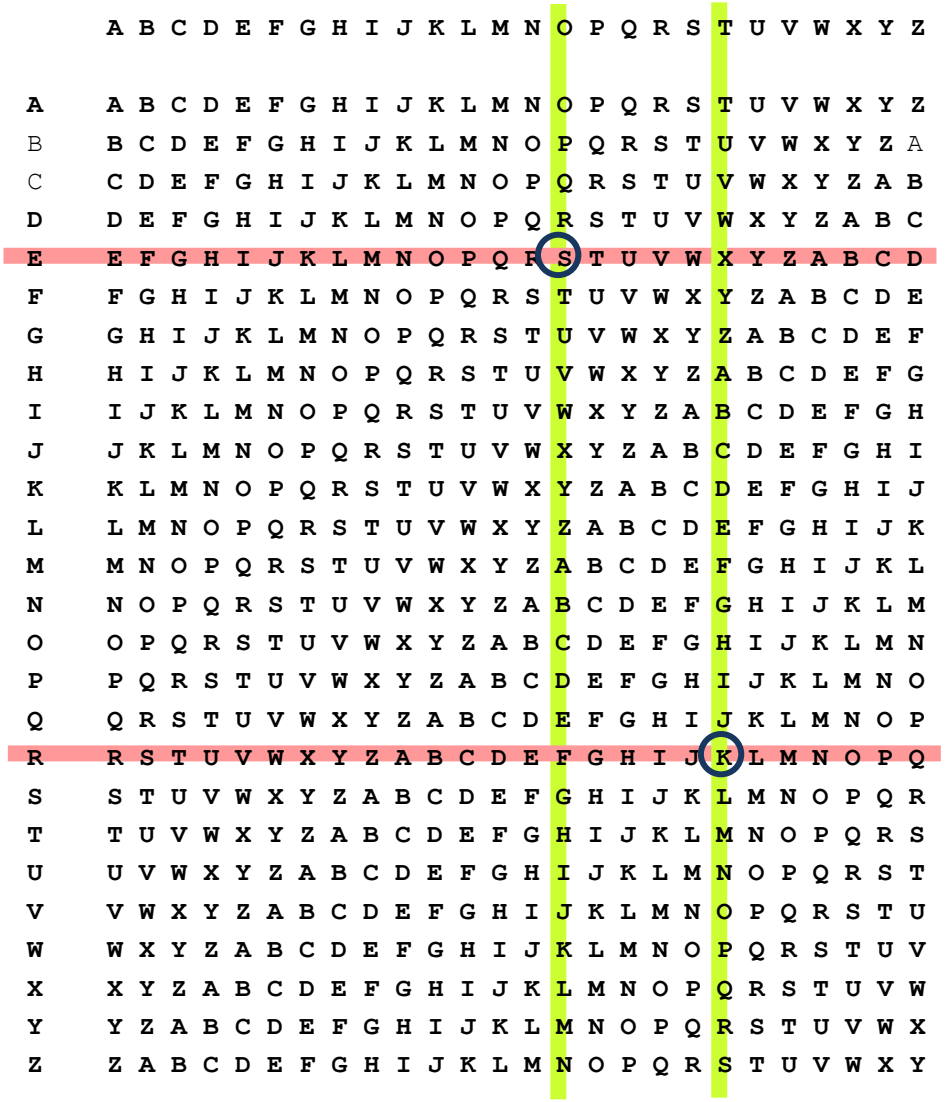
coding:

key: RELAT IONSR ELA

plaintext: TOBEO RNOTT OBE

ciphertext: KSMEH ZBBLK SME

Polyalphabetic substitution (Vigenère)



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

coding:

```
      key: RELAT IONSR ELA
plaintext: TOBEO RNOTT OBE
ciphertext: KSMEH ZBBLK SME
```

decoding:

```
      key: RELAT IONSR ELA
ciphertext: KSMEH ZBBLK SME
plaintext: TOBEO RNOTT OBE
```

The Enigma

- first electro-mechanical cipher
- patented by Arthur Scherbius in 1918
- adopted by the German Army in 1926

Main components of the Enigma

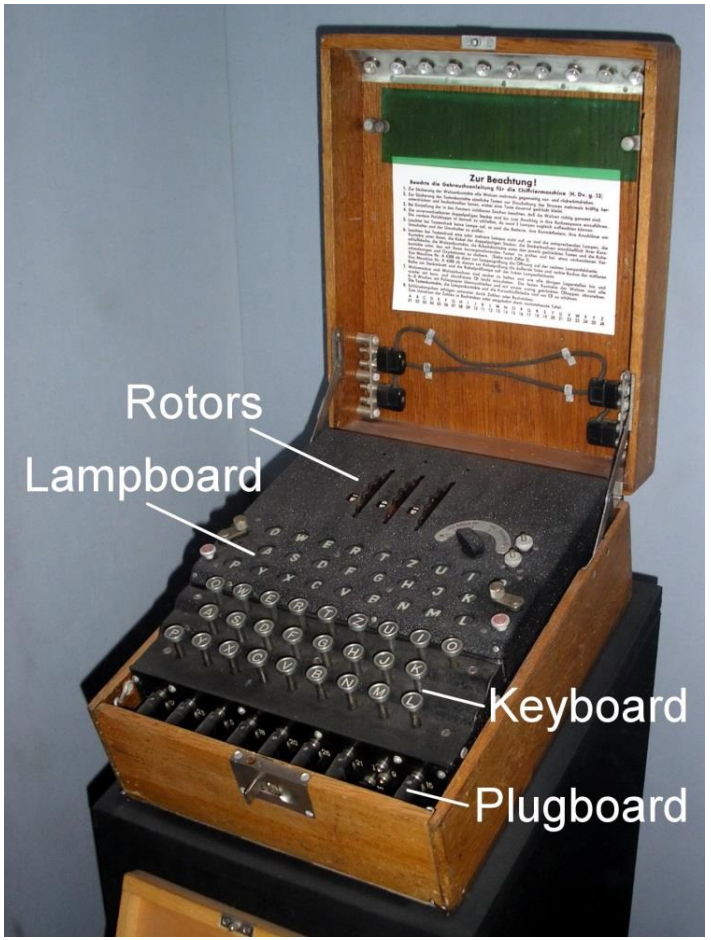
- four main components:
  - keyboard

    for input of the plaintext / ciphertext
  - lampboard

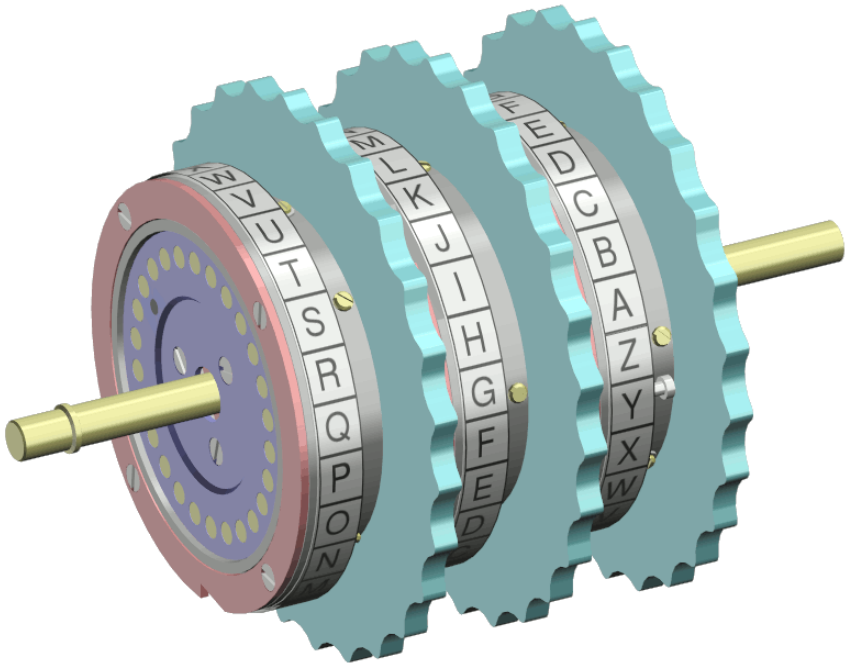
    for display of the ciphertext / plaintex
  - plugboard

    for swapping some input letter pairs
  - scrambler unit (including the rotors)

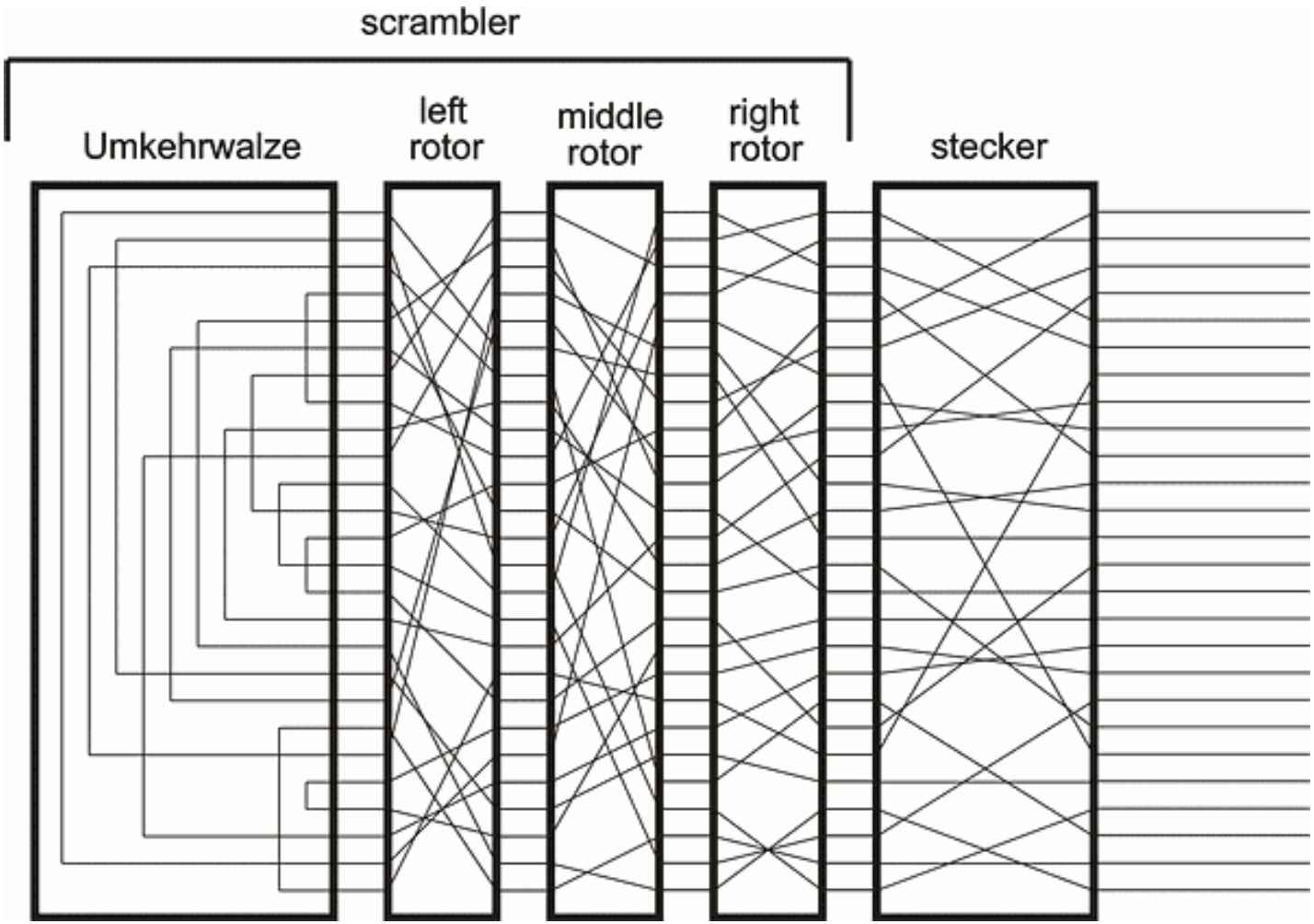
    producing the ciphertext from the plaintext (and vice versa)



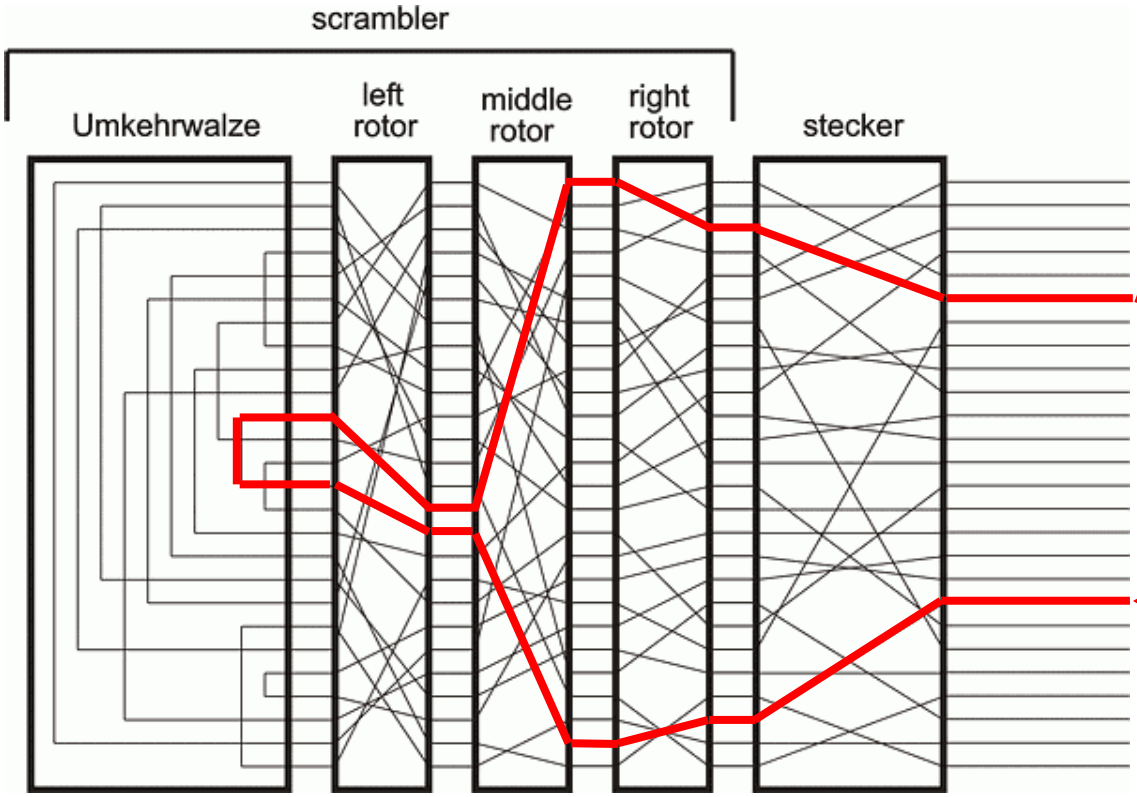
The rotors



The scrambler unit and the plugboard



Enigma in action



set rotors

rotor advances automatically

Enigma key space

- the key consists of the following basic settings:
  - letter pairs swapped (e.g., A/L – P/R – T/D – B/W – K/F – O/Y)
  - order of rotors in the slots (e.g., II – III – I)
  - initial position of the rotors (e.g., R – D – D)

- key space size:

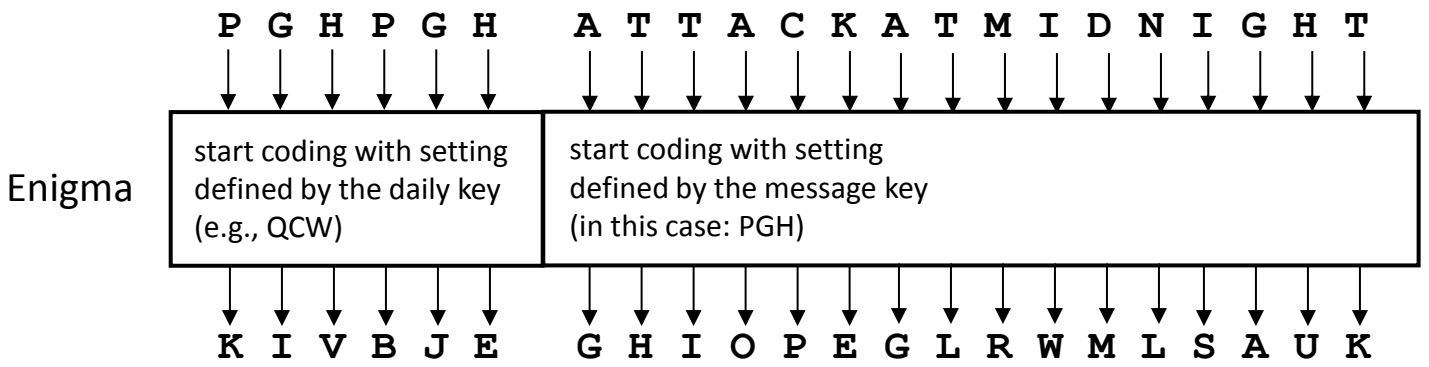
  $100391791500 \times 6 \times 26^3 \sim 2^{53}$

- yet, Enigma was broken by the Allies in WWII
  - exploiting protocol weaknesses and weak keys
  - code breaking was partly automated → birth of first computers
  - credit goes to Marian Rejewski and Alan Turing

Breaking the Enigma

- every morning, the Germans distribute a daily key to their units to be used with Enigma
- however, they do not directly use the daily key to encrypt messages
- instead:
  - they generate a fresh message key for every message
  - they encrypt the message key with the daily key, and send this at the beginning of the communication
  - then they encrypt the message with the message key, and send it to the receiver
  - the receiver first decrypts the message key with the daily key and then decrypts the message with the message key
- in order to cope with errors during transmission, the message key is repeated twice at the beginning of the message!

- example:



Breaking the Enigma

- Rejewski thought that the repetition of the message key at the beginning of the message is a weakness that may be exploited
  - a guess for the daily key can be confirmed by checking if decoding with the guessed key produces a repeating letter triplet at the beginning of the decoded message
- the Polish codebreakers built a machine that tried different guesses for the daily key in an automated way
  - the machine consisted of 6 Enigma copies (each corresponding to one of the 6 possible rotor orders)
  - the machine continuously modified the position setting of the rotors, and attempted decrypting some intercepted message, until it found the daily key
- from 1933, Poland was able to routinly break encrypted German communications

Breaking the Enigma

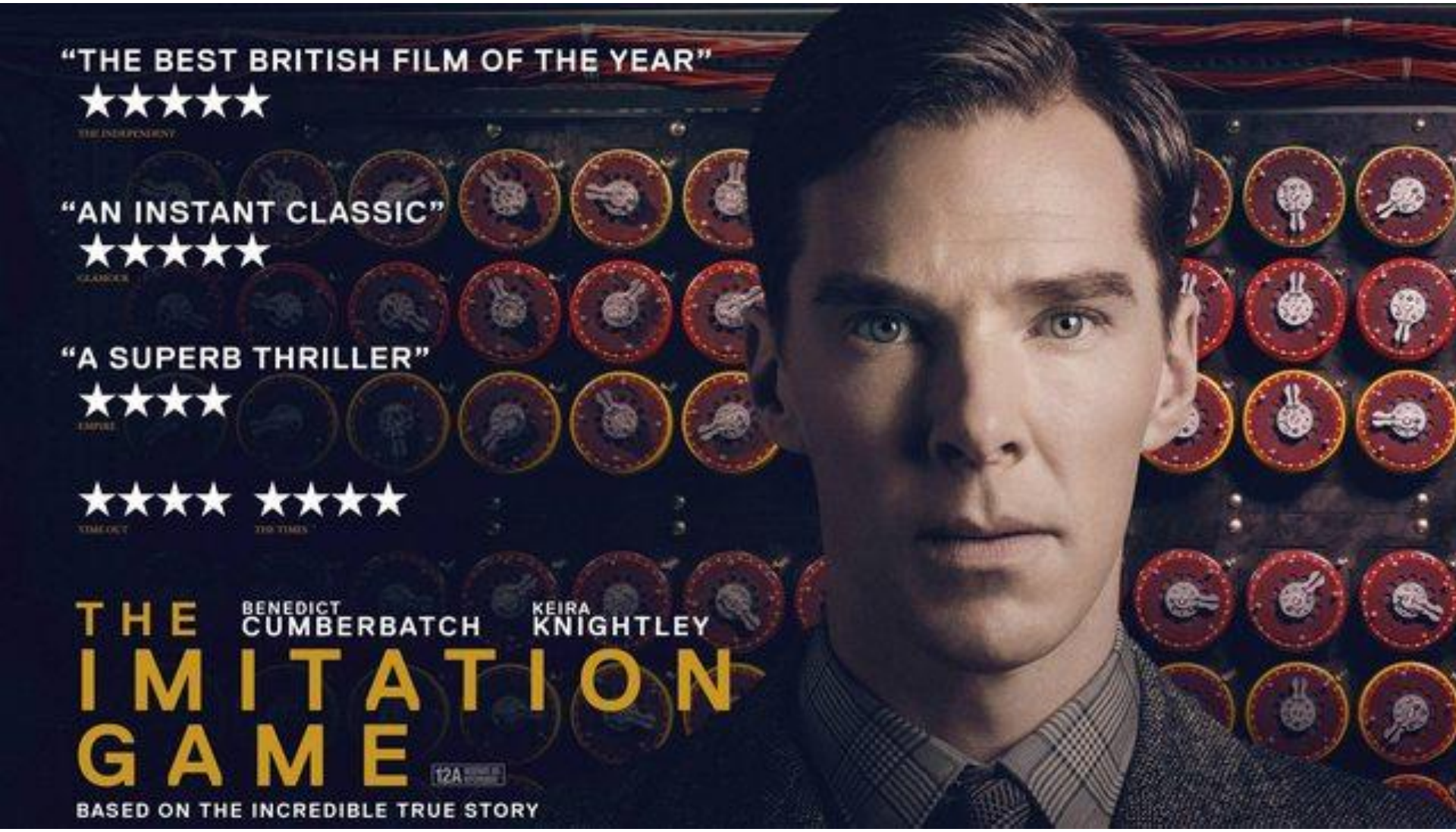
- in December 1938, the Germans increase the security of the Enigma
  - they introduce 2 new rotors (operators have to choose 3 rotors out of 5, and the order in which they are put in the machine → this increases possible rotor placements from 6 to 60)
  - they increase the number of letter pairs swapped on the plugboard from 6 to 10
  - key space grows to ~2^{66}
- in April 1939, Hitler breaks the non-aggression treaty with Poland
- in July 1939, Poland reveals their Enigma breaking capability to England
- on August 16, 1939, the design documents of the Enigma breaking machine are transferred to London
- on September 1, 1939, Germany invades Poland

Breaking the Enigma

- some weaknesses exploited by the British
  - cillies
    - » German Enigma operators sometimes used very weak (far from random) message keys (e.g., QWE, BNM)
    - » an operator always used the same message key (C.I.L.) – perhaps the initials of his wife or girl friend?
    - » these weak keys were called *cillies* (~silly)
  - Germans had usage constraints that actually weakened their system
    - » rotors had to be changed every day, and the same rotor must not be placed in the same slot on two consecutive days
    - » e.g., after I-II-V, they could not use III-II-IV
    - » this actually reduced the size of the key space that the British had to search over

Breaking the Enigma

- in September 1939, Alan Turing joins the code breakers in Bletchley Park
- his task is to find a new method for breaking the cipher that does not rely on the repetition of the message key at the beginning of the coded message
- Turing invents a new method that is essentially an attack known today as the known-plaintext attack
  - German messages are well structured
  - some messages contain guessable words at guessable locations
  - e.g., every morning at 6am, they send a weather forecast, which includes the world ”wetter” always at the same position within the message
- the British build new Enigma breaking machines (Victory, Agnus Dei) based on the plans of Turing in 1940
- indeed, Germans change their message key sending protocol in May 1940, but this does not affect the cryptanalytic capabilities of the British anymore

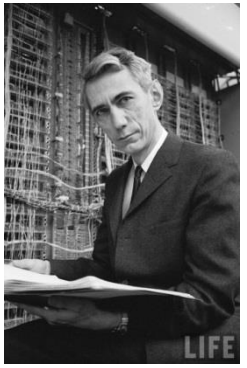

"THE BEST BRITISH FILM OF THE YEAR"
"AN INSTANT CLASSIC"
"A SUPERB THRILLER"
BENEDICT CUMBERBATCH
KEIRA KNIGHTLEY
THE IMITATION GAME
12A
BASED ON THE INCREDIBLE TRUE STORY

Modern cryptography

- Shannon's work on information theoretical characterization of encryption [1948]
- substitution-permutation ciphers and the Data Encryption Standard (DES) [1970's]
- the birth of public key cryptography [1976-78]
- quantum cryptography [1980's]

The birth of modern cryptography

- first theoretically sound formulation of the notion of security of an encryption algorithm
  - used information theory to define the concept of perfect secrecy
  - gave necessary conditions for a cipher to be perfectly secure
  - proved that the one-time pad provides perfect secrecy



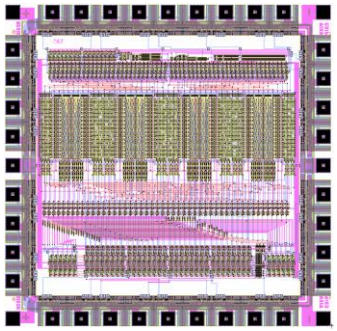
Claude E. Shannon

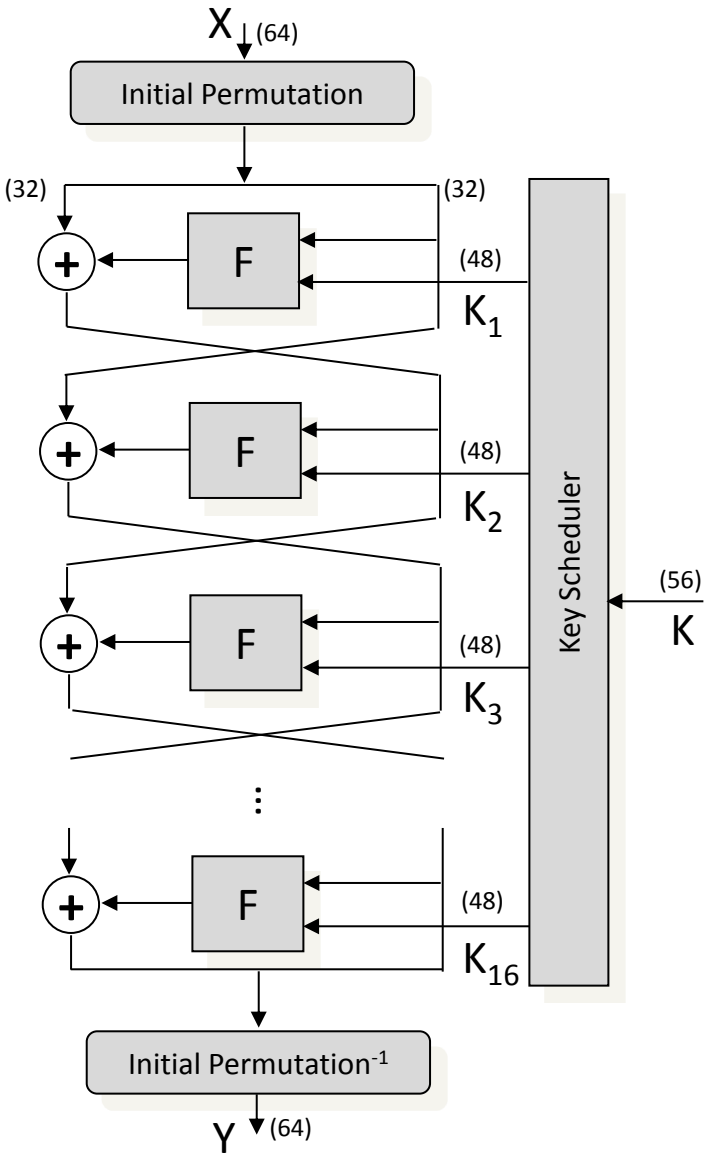
- ideas to build strong block ciphers usable in practice
  - create a complex cipher by repeated use of otherwise simple transformations
  - none of the simple transformations alone would be sufficiently strong, but their repeated use and the large number of iterations would ultimately result in a strong cipher (aka. product ciphers)

Data Encryption Standard (DES)

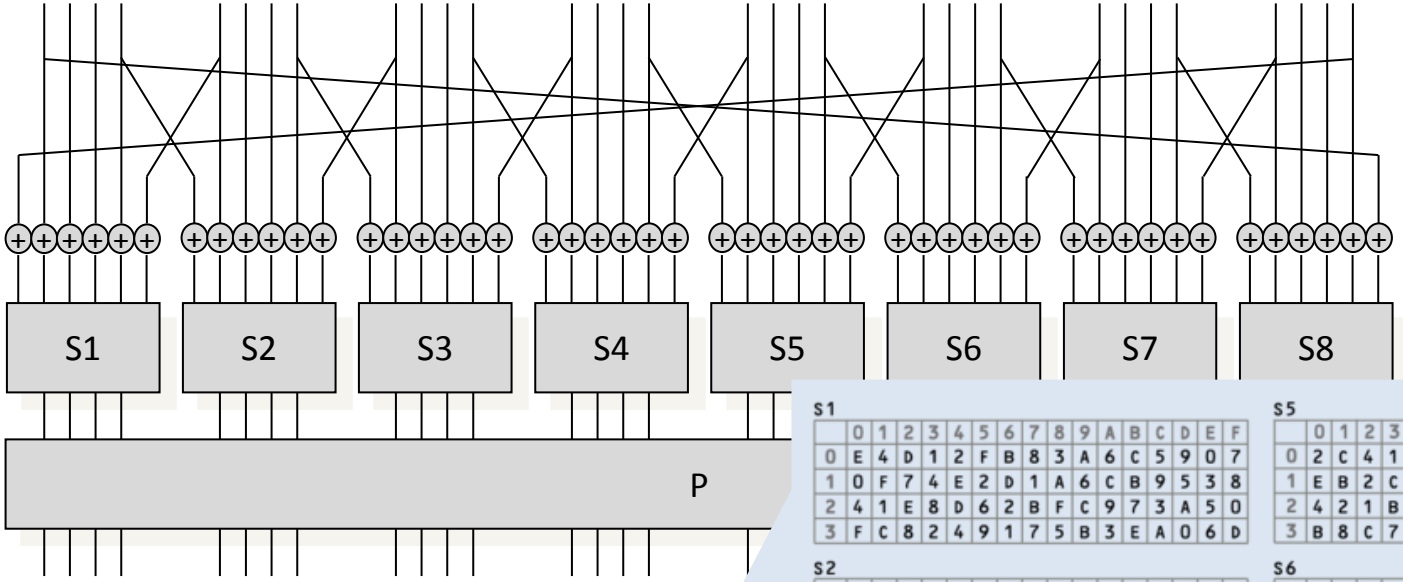
- based on Lucifer, a cipher developed by IBM in the 70's
- symmetric key **block cipher**
- features:
  - Feistel structure (same structure can be used for encoding and decoding)
  - number of rounds: 16
  - input block size: 64 bits
  - output block size: 64 bits
  - key size: 56 bits

HW implementation:
DES chip





DES round function F



- Si – substitution box (S-box)
  - » non-linear look-up tables
- P – permutation box (P-box)
  - » linear bit permutation

S1

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	E	4	D	1	2	F	B	8	3	A	6	C	5	9	0	7
1	0	F	7	4	E	2	D	1	A	6	C	B	9	5	3	8
2	4	1	E	8	D	6	2	B	F	C	9	7	3	A	5	0
3	F	C	8	2	4	9	1	7	5	B	3	E	A	0	6	D

S2

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	F	1	8	E	6	B	3	4	9	7	2	D	C	0	5	A
1	3	D	4	7	F	2	8	E	C	0	1	A	6	9	B	5
2	0	E	7	B	A	4	D	1	5	8	C	6	9	3	2	F
3	D	8	A	1	3	F	4	2	B	6	7	C	0	5	E	9

S3

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	A	0	9	E	6	3	F	5	1	D	C	7	B	4	2	8
1	D	7	0	9	3	4	6	A	2	8	5	E	C	B	F	1
2	D	6	4	9	8	F	3	0	B	1	2	C	5	A	E	7
3	1	A	D	0	6	9	8	7	4	F	E	3	B	5	2	C

S4

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	7	D	E	3	0	6	9	A	1	2	8	5	B	C	4	F
1	D	8	B	5	6	F	0	3	4	7	2	C	1	A	E	9
2	A	6	9	0	C	B	7	D	F	1	3	E	5	2	8	4
3	3	F	0	6	A	1	D	8	9	4	5	B	C	7	2	E

S5

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	2	C	4	1	7	A	B	6	8	5	3	F	D	0	E	9
1	E	B	2	C	4	7	D	1	5	0	F	A	3	9	8	6
2	4	2	1	B	A	D	7	8	F	9	C	5	6	3	0	E
3	B	8	C	7	1	E	2	D	6	F	0	9	A	4	5	3

S6

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	C	1	A	F	9	2	6	8	0	D	3	4	E	7	5	B
1	A	F	4	2	7	C	9	5	6	1	D	E	0	B	3	8
2	9	E	F	5	2	8	C	3	7	0	4	A	1	D	B	6
3	4	3	2	C	9	5	F	A	B	E	1	7	6	0	8	D

S7

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	4	B	2	E	F	0	8	D	3	C	9	7	5	A	6	1
1	D	0	B	7	4	9	1	A	E	3	5	C	2	F	8	6
2	1	4	B	D	C	3	7	E	A	F	6	8	0	5	9	2
3	6	B	D	8	1	4	A	7	9	5	0	F	E	2	3	C

S8

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	D	2	8	4	6	F	B	1	A	9	3	E	5	0	C	7
1	1	F	D	8	A	3	7	4	C	5	6	B	0	E	9	2
2	7	B	4	1	9	C	E	2	0	6	A	D	F	3	5	8
3	2	1	E	7	4	A	8	D	F	C	9	0	3	5	6	B

Security of DES

- average complexity of a brute force attack is 2^{55}
  - was suspected breakable by NSA back in the 70's
  - definitely became breakable by the late 90's by distributed computing
  - new standard AES was accepted in 2001

- algebraic attacks
  - DES has never been broken in a practical sense
  - best known attacks:
    - » linear cryptanalysis (LC)
      - requires ~2^{43} known plaintext – ciphertext pairs
    - » differential cryptanalysis (DC)
      - requires ~2^{47} chosen plaintexts (and corresponding ciphertexts)
  - DC and LC were discovered in the late 80's and early 90's
  - it was revealed in the late 90's that the designers of DES had known about DC, and optimized the DES S-boxes such that DES provides maximum resistance against DC

A breakthrough in modern cryptography

Whitfield Diffie and Martin Hellman. New Directions in Cryptography. *IEEE Transactions on Information Theory*, 1976

Raplh Merkle, Martin Hellman, and Whitfield Diffie

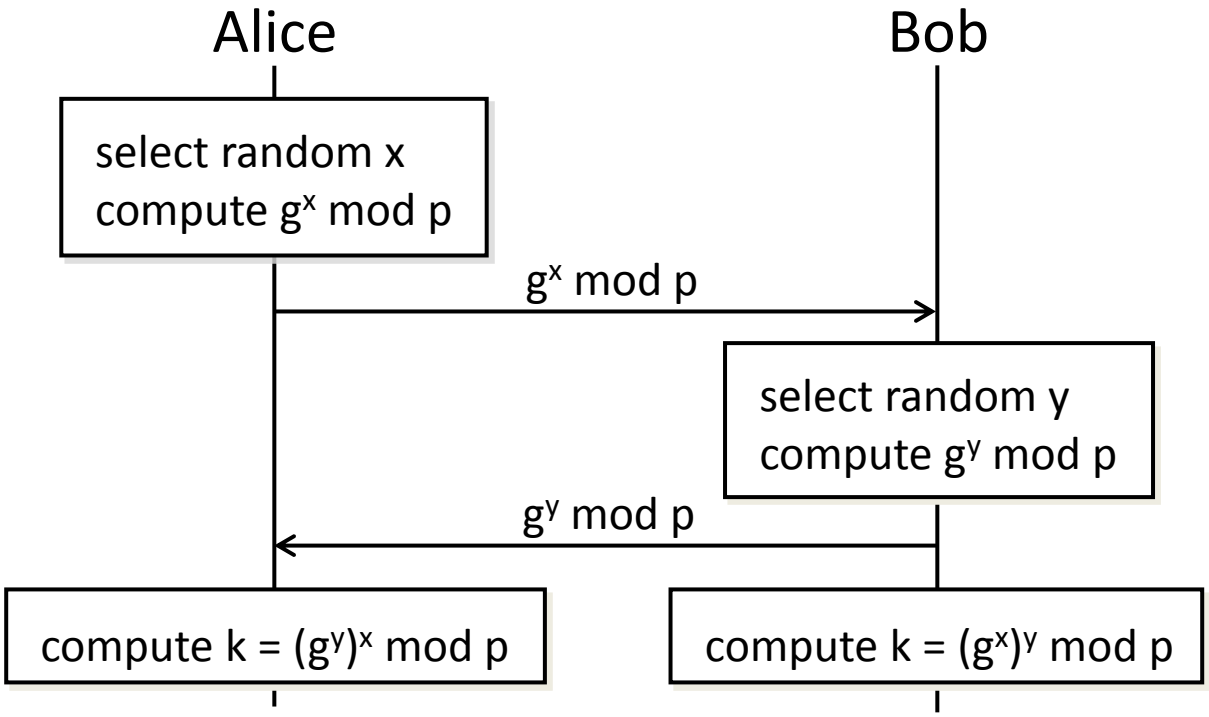
The key exchange problem

- by the 70's digital computers and telecommunication networks were increasingly used in the financial sector
- banks could use symmetric key ciphers, such as Lucifer and later DES, to encrypt sensitive data
- but they faced a practical question: how to setup a shared DES key between two end points (e.g., two remote branches of the same bank) ???
  - in case of earlier military and diplomatic applications, keys were transferred by agents in a physically secure way
  - this was expensive and inflexible for banks

The Diffie-Hellman key exchange protocol

public parameters:

a large prime p and a generator element g of $Z_p^* = \{1, 2, \ldots, p-1\}$



The Diffie-Hellman key exchange protocol

- if an attacker can only eavesdrop the communications between Alice and Bob, then he has only $g^x \bmod p$ and $g^y \bmod p$
- to compute $g^{xy} \bmod p$, he would need x or y
- it is hard to compute x from $g^x \bmod p$
  - this is the so called ”discrete logarithm” problem
  - no polynomial time algorithm is known to solve it
  - if p is large, then computing discrete logarithm (mod p) is practically infeasible

- there seem to exist one way functions:
  - given x, it is easy to compute f(x)
  - given y, it is hard to find an x for which y = f(x)
- can we use such functions to realize a sort of asymmetric key cryptography ???

The idea of asymmetric key cryptography

- encoding and decoding keys are not the same (unlike in symmetric key cryptography)
- computing the decoding key from the encoding key is hard (infeasible in practice)
- encoding key can be made public, decoding key should be kept secret
  - anybody can obtain the public encoding key of Alice, and send an encrypted message to her
  - only Alice can decrypt the message with the private decoding key
  - an attacker cannot compute the private key from the public key
  - aka. public key cryptography
  - solves the key exchange problem (but has other issues to solve)

The RSA cryptosystem

Ronald Rivest, Adi Shamir, Leonard Adleman. A Method for Obtaining Digital Signatures and Public-Key Cryptosystems, 1978

Adi Shamir, Ronald Rivest, and Leonard Adleman

The RSA cryptosystem

- key-pair generation algorithm:
  - choose two large primes p and q (easy)
  - $n = pq$, $\phi(n) = (p-1)(q-1)$ (easy)
  - choose e, such that $1 < e < \phi(n)$ and $\gcd(e, \phi(n)) = 1$ (easy)
  - compute the inverse d of e mod $\phi(n)$, i.e., d such that $ed \bmod \phi(n) = 1$ (easy if p and q are known)
  - output **public key: (e, n)** (public exponent and modulus)
  - output **private key: d** (private exponent)
- encryption algorithm:
  - represent the plaintext message as an integer $m \in [0, n-1]$
  - compute the ciphertext $\mathbf{c = m^e \bmod n}$
- decryption algorithm:
  - compute the plaintext from the ciphertext c as $\mathbf{m = c^d \bmod n}$

Security of asymmetric key algortihms

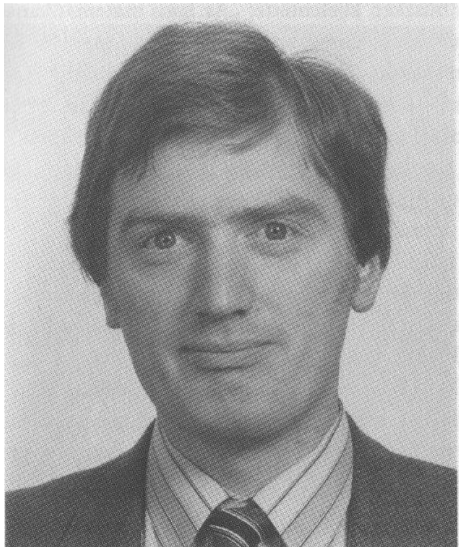
- security is typically related to the difficulty of solving some hard mathematical problem
  - e.g., factoring or discrete logarithm
- provable security by reduction proofs:
  - we show that any efficient algorithm that breaks our crypto scheme could be used to efficiently solve a believed to be hard mathematical problem
  - this means that breaking our crypto scheme is at least as hard as solving the hard mathematical problem
- there exist provably secure crypto systems, but most of them are not efficient (fast) enough for practical applications
- most of the public key crypto schemes that we use in pracitce are not provably secure (or only partial proofs exist)

Example: Security of the RSA crypto system

- factoring integers is believed to be a hard problem
  - given a composit integer n, find its prime factors
  - true complexity is unknown
  - it is believed that no polinomial time algortihm exists to solve it
- computing d from (e, n) is equivalent to factoring n
- computing m from c and (e,n) may not be equivalent to factoring n (this is known as the RSA problem)
  - if the factors p and q of n are known, then one can easily compute d, and using d, one can also compute m from c
  - we don’t know if one could factor n, given that he can efficiently compute m from c and (e,n)

The secret story of public key cryprography

James Ellis



Clifford Cocks

Malcolm Williamson

The secret story of public key cryprography

- Ellis, Cocks, and Williamson worked for GCHQ (British security agency)
- in 1969, Ellis defined the general model of asymmetric key cryptography (called it non-secret key coding)
  - public and private keys
  - (trap-door) one way functions
- in 1973, Cocks invented a cryptosystem same as RSA
  - he was introduced to the idea of non-secret key crypto
  - he worked in the field of number theory, and immediately thought of using factoring as a hard problem
- in 1974, Williamson (a friend of Cocks) invented a key exchange protocol same as the Diffie-Hellman protocol
- by 1975, Ellis, Cocks, and Williamson worked out all the major results of public key cryptography, which were (re)invented some years later
- the story was made pulic only in 1997

Pretty Good Privacy (PGP)

- Phil Zimmermann
  - a peace activist in the 1980s during the Nuclear Weapons Freeze campaign
  - saw the need to develop what would later become PGP
    - » for protecting human rights overseas
    - » for protecting grassroots political organizations in the US

- US Senate Bill 266 of 1991
  - Congressional discussion on requiring that all communications equipment and services have a “trap door” in them to permit government anti-criminal and counterterrorism activities
  - familiar?
    - » U.S.A. P.A.T.R.I.O.T. Act of 2001 signed into law by G. W. Bush
    - » extension by 4 years in 2011 by B. Obama

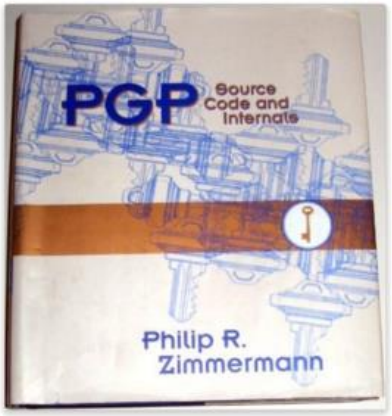
- first working version of PGP arrived in 1991 (when it was still legal)
  - free software that used strong encryption (e.g., RSA)
  - strong crypto available to ordinary people for the first time in history
  - new opportunities for human rights organizations and other users concerned with privacy

Patent and export problems with PGP

- the RSA algorithm was patented in the US by MIT, and was licensed to RSA Data Security Inc.
  - years of disagreement about the permission to use RSA in PGP
  - finally, RSADSI created the RSAREF library for use in freeware and shareware, and PGP 2.5 used RSAREF (in the US)
  - an “international” version of PGP, developed completely outside of the US, used the original implementation of the RSA algorithm
- Public Key Partners filed a complaint in 1992 with US Customs, complaining that Zimmermann was exporting cryptography without the appropriate licenses
  - until 1997, international regulation considered cryptography a weapon
  - free and open cryptosystems were regulated as munitions in the US
  - a criminal investigation of Zimmermann was started in 1992
  - printed books were and are exempt from the export controls
  - the investigation of Zimmermann was dropped in 1996
  - export controls on cryptography were radically liberalized in 2000

PGP and the crypto war

- publication of *PGP Source Code and Internals* (MIT Press, 1995)



PGP: Source Code and Internals Hardcover – June 9, 1995
by Philip R. Zimmermann (Author)
1 customer review

See all formats and editions

Hardcover
from $285.00

9 Used from $285.00
3 New from $1,008.50

- later, Pretty Good Privacy Inc. published the source code of PGP in a more sophisticated set of books
  - also included specialized software tools optimized for easy optical character recognition (OCR) scanning of C source code
  - this made it easy to export unlimited quantities of cryptographic source code, rendering the export controls moot

Quantum and post-quantum crypto

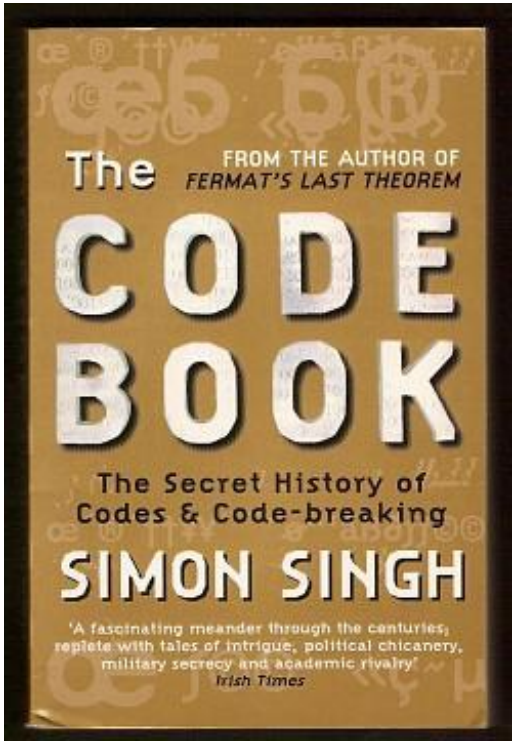
- quantum cryptography (started in the 1980's)
  - using quantum effects to solve traditional problems in new ways
    - » e.g., quantum key exchange using polarized photons
  - using quantum computers to break modern ciphers efficiently
    - » e.g., the Schor factorization algortihm to break RSA

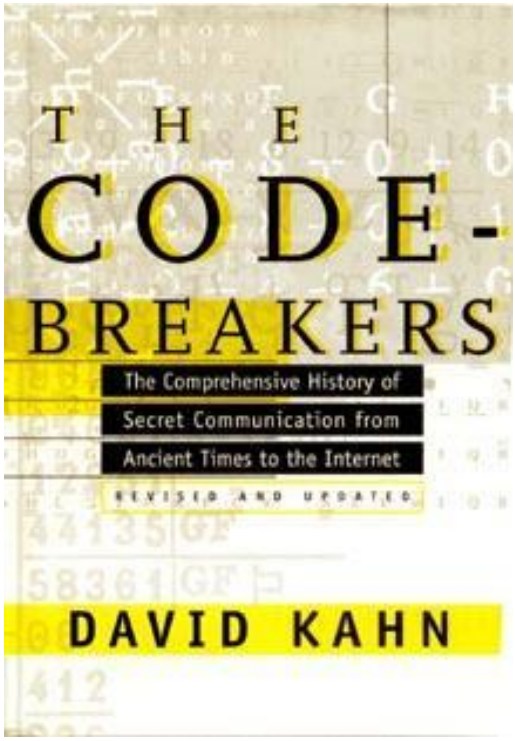
- post-quantum cryptography
  - developing cryptographic algorithms that resist even attacks by a quantum computer
    - » see http://pqcrypto.org/

Practical applications of cryptography

- today, national and international laws, regulations, and expectations about privacy, data governance, and corporate governance either imply or require the widespread use of strong cryptography
  - secure communication over public channels / networks
    - » WWW (https / TLS)
    - » WiFi (WPA, WPA2)
    - » GSM/3G
    - » Bluetooth
  - secure data storage
    - » disk encryption (TrueCrypt, BitLocker, ...)
    - » encrypted cloud strage (Tresorit, CipherCloud, ...)
  - authentication
    - » smart cards (e.g., bank cards)
    - » ignition keys of cars
    - » electronic tickets in public transport (automated fare collection systems)
  - software authentication and integrity protection
    - » digitally signed code (e.g., drivers, applets, Android packages)
  - ...

Further readings





Avatao challenges

https://avatao.com/

- Goal: Mastering Cryptographic Engineering
- Module: Challenges for a Cryptographic Protocols course
- Challenges:
  - Breaking the Nihilist historical cipher
  - Trithemius cipher
  - Four-Square game