数据隐私方法伦理和实践
Methodology, Ethics and Practice of Data Privacy

8.密码学
Introduction to Cryptography

张兰
中国科学技术大学 计算机学院
2020春季

1. History of Cryptography

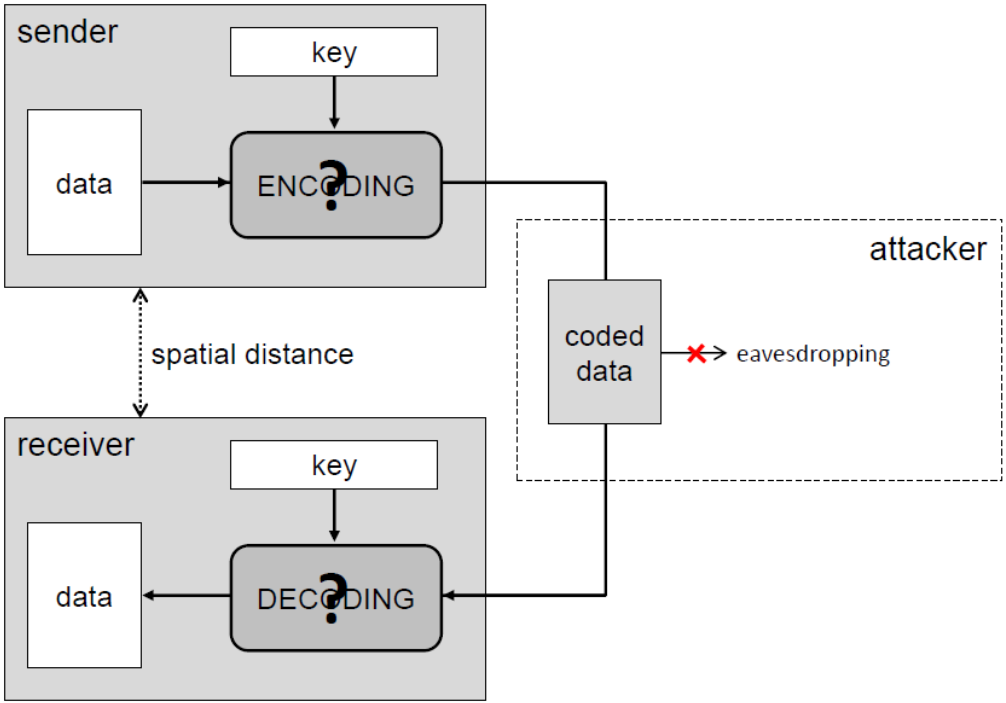
There is a tremendous flow of data among many sources that needs to be somehow kept secret and cryptography is the way to achieve this.

Some slides are from Levente Buttyán, CrySyS Lab, BME

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 (bank)
  - Besides confidentiality, integrity, authentication, and repudiation becomes important too
- From the end of the 20th century:
  - Cryptography is used in everyday life of people
  - SSL/TLS- secure web transactions
  - GSM/.../4G/5G security-subscriber authentication, encryption
  - Wi-Fi, Bluetooth, smart cards, disk encryption,...

Basic Model

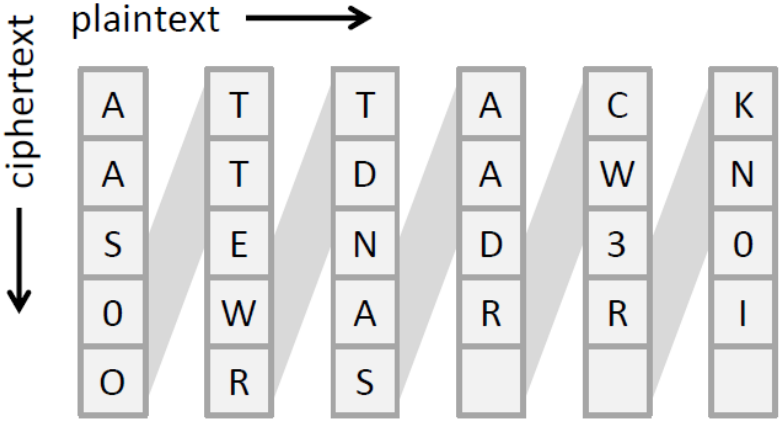

sender
key
data
ENCODING
?
attacker
coded data
eavesdropping
spatial distance
receiver
key
data
DECODING
?

Greek Cryptography – The Scytale

- Used by the Spartans in the 3rd century BC
- Transposition cipher (mixes letters of the plaintext)

Greek Cryptography – The Scytale

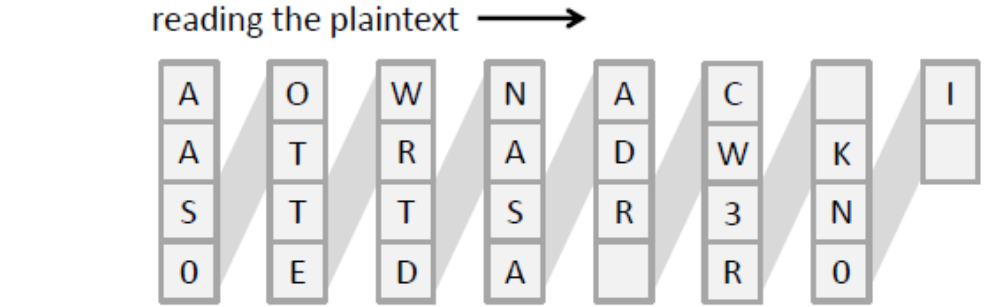
- Used by the Spartans in the 3rd century BC
- Transposition cipher (mixes letters of the plaintext)
- What is the key?



Greek Cryptography – The Scytale

- Key: the diameter of the rod
- Key space is small- easy to break

using a rod with a wrong diameter:



Caesar(shift) 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

Cipher: D E F G H I J K L M N O P Q R S

Attack→?

Caesar(shift) 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**

Cipher: **D E F G H I J K L M N O P Q R S**

Attack→ DWWDN

- The key is the value of the shift
- **What is the size of the key space?**

Caesar(shift) Cipher

- Elegant mathematical description of the cipher.

  Let $k,x,y \in \{0,1,\ldots,25\}$
  Encryption: $y=e_k(x) \equiv x+k \bmod 26$
  Decryption: $x=d_k(y) \equiv y-k \bmod 26$

- Q: Is the shift cipher secure?

- A: No! several attacks are possible, including:
  - Brute force search (key space is only 26)
  - Letter frequency analysis

Affine Cipher

- Extension of the shift cipher: rather than just adding the key to the plaintext, we also multiply by the key

We use for this a key consisting of two parts: k = (a, b)

Let $k,x,y \in \{0,1,\ldots,25\}$
Encryption: $y=e_k(x) \equiv ax+b \bmod 26$
Decryption: $x=d_k(y) \equiv a^{-1}(y-b) \bmod 26$

- Since the inverse of *a* is needed for inversion, we can only use values for *a* for which: gcd(*a*, 26) = 1
- There are 12 values for *a* that fulfill this condition. From this follows that the key space is only 12 x 26 = 312

Monoalphabetic Substitution

- Generalization of the Caesar cipher
- Replacement of letters is determined by permutation

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

  Cipher: H T K C U O I S J Y A R G M Z N B V F P

  Attack→ HPPHKA

- The key is the permutation
- **What is the size of the key space?**

Monoalphabetic Substitution

- Generalization of the Caesar cipher
- Replacement of letters is determined by permutation

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

  Cipher: H T K C U O I S J Y A R G M Z N B V F P

  Attack→ HPPHKA

- The key is the permutation
- **What is the size of the key space?**

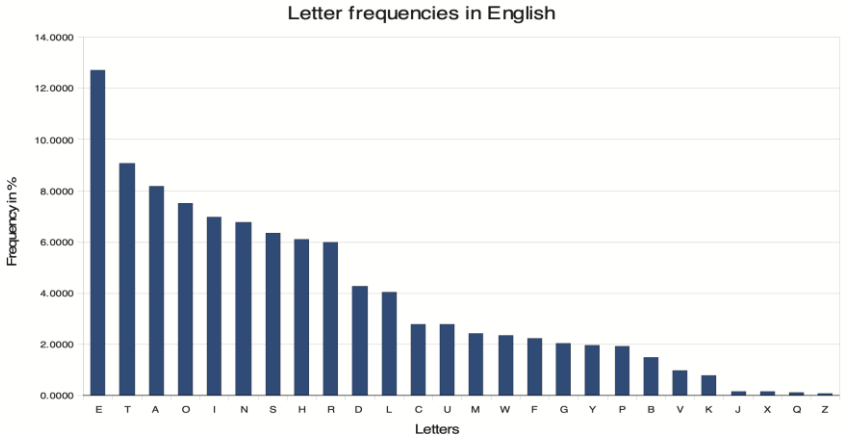
$$26! \sim 1.56 * 2^{88}$$

Age of the earth .....2^{55} sec

Age of the Universe .....2^{59} sec

Breaking Monoalphabetic Substitution

- Every language has its own letter statistics
  - Letter frequencies are independent of the actual text
  - There are letters that are more frequent than others and letters that are less frequent
  - E.g., e→12.7% , t→9.1%, z→0.1%, j→0.2%



Breaking Monoalphabetic Substitution

- Every language has its own letter statistics
  - Letter frequencies are independent of the actual text
  - There are letters that are more frequent than others and letters that are less frequent
  - E.g., e→12.7% , t→9.1%, z→0.1%, j→0.2%

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

Breaking Monoalphabetic Substitution

» Let's return to our example and identify the most frequent letter:

iq ifcc vqqr fb rdq vfllcq na rdq cfjwhwz hr bnnb hcc

hwwhbsqvqbre hwq vhlq

» We replace the ciphertext letter q by E and obtain:

iE ifcc vEEr fb rdE vfllcE na rdE cfjwhwz hr bnnb hcc

hwwhbsEvEbre hwE vhlE

» By further guessing based on the frequency of the remaining letters we obtain the plaintext:

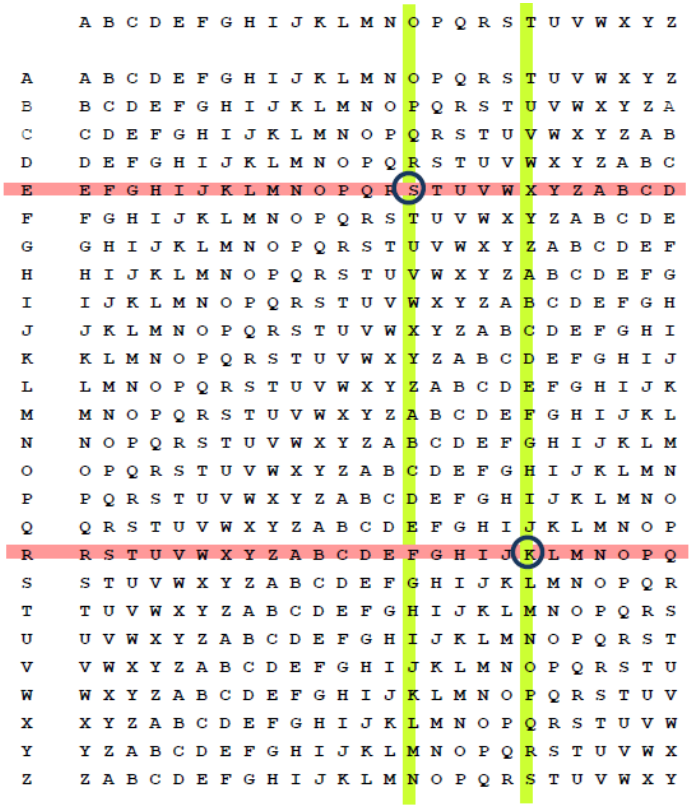
WE WILL MEET IN THE MIDDLE OF THE LIBRARY AT NOON ALL ARRANGEMENTS ARE MADE

Breaking Monoalphabetic Substitution

- If you wanna try...

```
RHINT TLHYF EIXQF AXTNK AFYFV QTRNF EHFEI XQFAD
TXXTE ZAFBA IRRHI NTLQR IAIMT EQNIA ITEFR HINTM
IASGI NXSET EIACR HINTX EIARH FEXRT EOQNT RNFMH
IAUKT ICWSI EXTET CQAKI BXKFE ATNIY QGFEA QISAQ
XTCRX IXTRQ XBIRG FSACT CQADU XTRYI MFXFE RNTFI
ACGFE MTEHI UHIYT AXETH ETATS ETYFA MSRZB QXKXK
TVFIY FGNET IXQAV XKTXT NKAFY FVQTR XFETC SNTRH
INTXE IARHF EXIXQ FANFR XRIAC TAIDY TXKTN FYFAQ
JIXQF AFGMI ERQXK IRCTO TYFHT CXKTG IYNFA QIACG
IYNFA QLYIS ANKOT KQNYT RDFXK CTRQV ATCXF DTETS
RIDYT IACXK TCEIV FARHI NTNEI GXBKQ NKQRG YFBAQ
AXFFE DQXDU XKTGI YNFAQ LYISA NKOTK QNYTX FRSHH
YUXKT QAXTE AIXQF AIYRH INTRX IXQFA QRRBQ XKNIE
VFIMI AATCO TERQF AFGCE IVFAQ RQACT OTYFH MTAXR
HINTL QRGSA CTCDU VFOTE AMTAX RSDRQ CQTRI ACNFA
XEINX RBQXK MSYXQ HYTTA XQXQT RRHIN TLRIN KQTOT
MTAXR QANYS CTXKT GQERX HEQOI XTYUG SACTC YQWSQ
CHEFH TYYIA XEFNZ TXGIY NFAXF ETINK FEDQX XKTGQ
ERXHE QOIXT YUGSA CTCNF MHIAU XFRSN NTRRG SYYUY
ISANK FEDQX IACET NFOTE IRHIN TNEIG XCEIV FAIAC
XKTGQ ERXHE QOIXT NFMHI AUXFR TACIR HINTN EIGXC
EIVFA XFXKT QRRXK TYISA NKFGR TROQQ QBIRX KTGQE
RXRHI NTLCT YQOTE UQAXF VTFRU ANKEF AFSRF EDQXB
...
```

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
```

Flatten the distribution of letters.

Polyalphabetic Substitution (Vigenère)

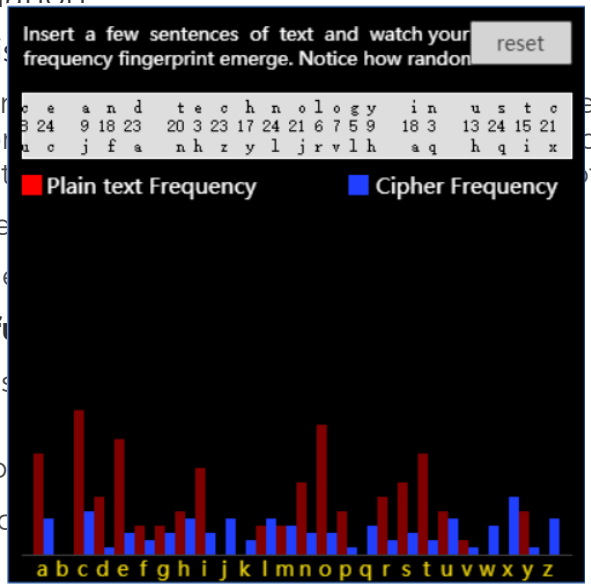
Flatten the distribution of letters.

One-Time Pad

- For over 400 years, the problem remained: how could Alice design a cipher that hides her fingerprint, thus stopping the leak of information?
- The answer is : randomness!
  - Imagine Alice rolled a 26 sided die to generate a long list of random shift (as long as the secret plaintext to avoid any repetition) and shared this with Bob instead of a code word.
  - Alice encrypt the secret using this list .
  - Bob decrypt the message using this list too.
- **Two powerful properties:**
  - 1. The shifts never fall into a repetitive pattern.
  - 2. The encrypted message will have a uniform frequency distribution.
  - The power of combinatorial explosion: 26^k

One-Time Pad

- For over 400 years, the problem remained: how could Alice design a cipher that hides her fingerprint, thus stopping the leak of information?
- The answer i
  - Imagi
  - Alice e
  - Bob d
- **Two powerf**
  - 1. The
  - 2. The
  - The p

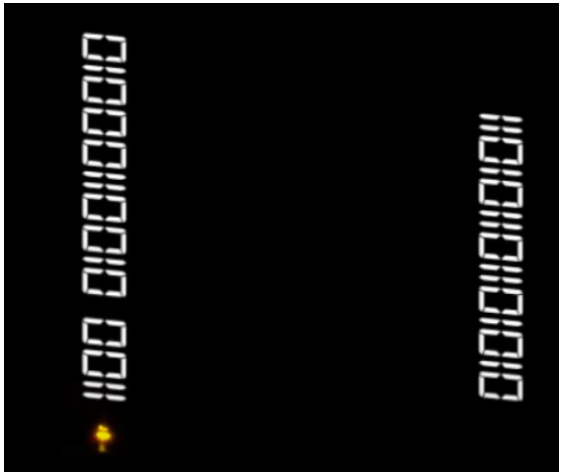


a long list of
o avoid any
f a code word.

frequency

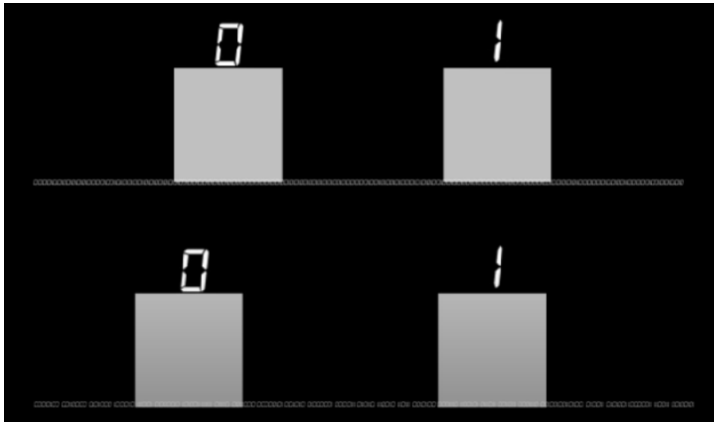
Frequency Stability Property

- Can you tell the difference between actions based upon flipping a coin and those based upon blind guessing or simulating randomness?



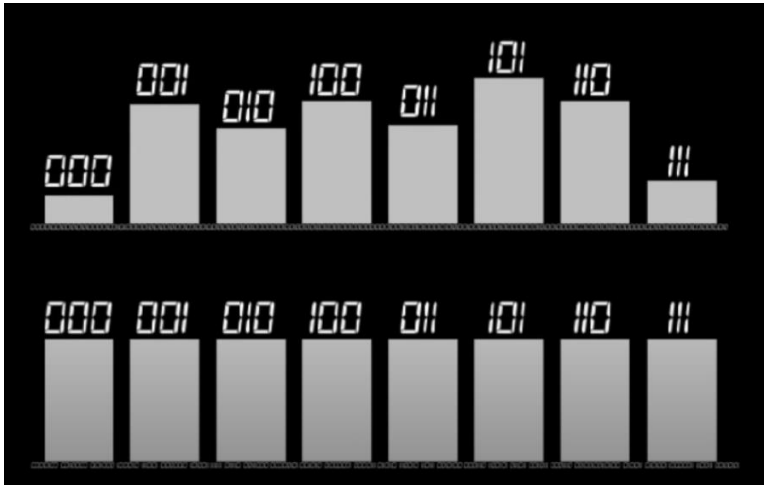
Frequency Stability Property

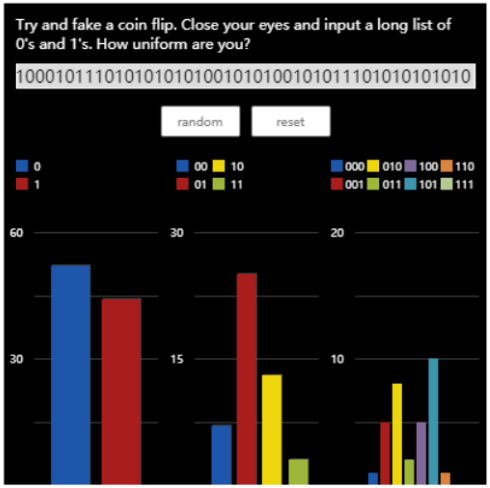
» Can you tell the difference between actions based upon flipping a coin and those based upon blind guessing or simulating randomness?



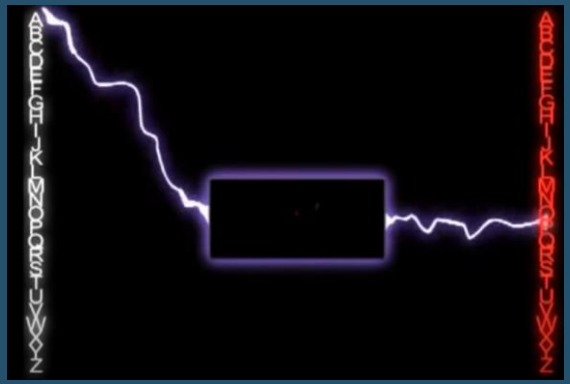
Frequency Stability Property

» Can you tell the difference between actions based upon flipping a coin and those based upon blind guessing or simulating randomness?





How to automate the one-time pad using an encryption machine?



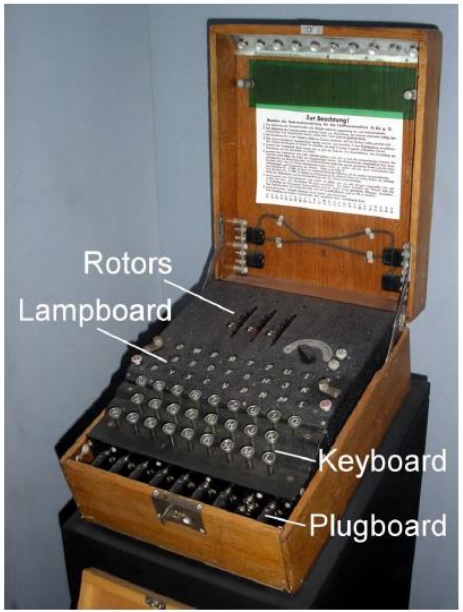
Enigma

- First electro-mechanical cipher
  - Substitution and permutation
- Adopted by the German Army in 1926
- Cryptanalytic effort involved development and use of early computers (Colossus).

Enigma

- Identical machines: always predictable and repeatable.
- **Goal: identical machines → output a scrambled sequence of shifts which took a long time to repeat.**
- Then, Alice and Bob can share identical machines and agree on an initial position and cycle through the identical operations to achieve identical sequences.

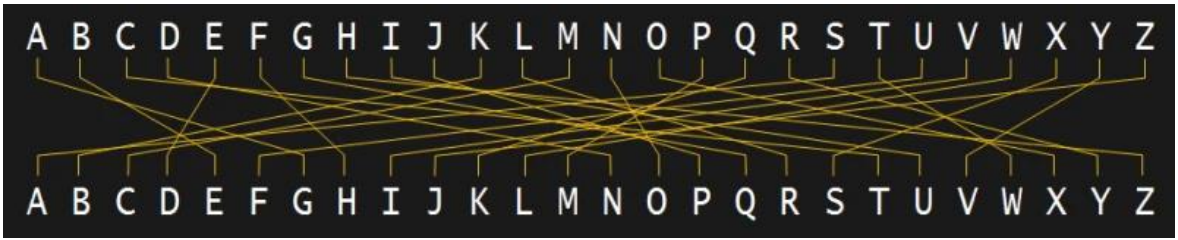
Enigma



- Four main components:
  - Key board: for input of the plaintext/ciphertext
  - Lamp board: for display
  - Plug board: for swapping some input letter pairs
  - Scrambler unit (including the rotors): producing the ciphertext from the plaintext (and vice versa)

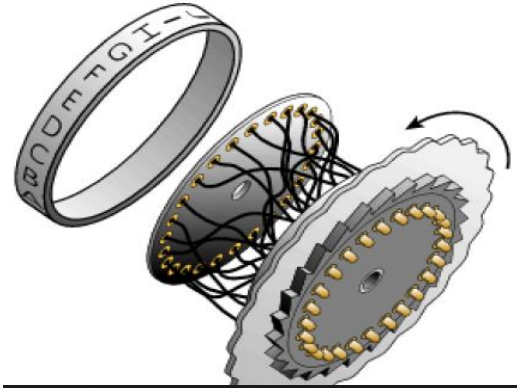
Enigma-rotor

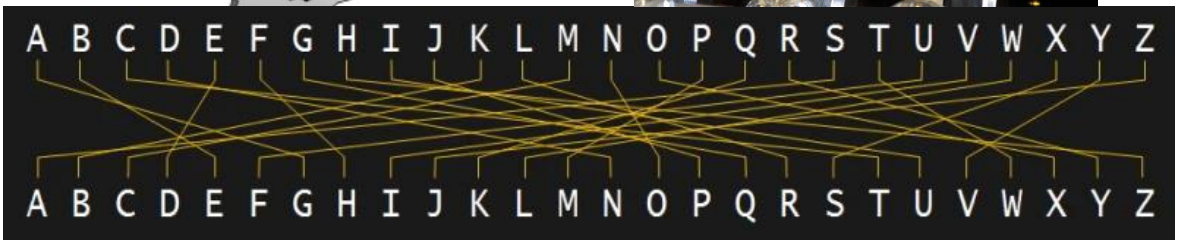
Simple substitution cipher with wires electrically connecting each pair, which opens up the possibility of easily changing how the substitution is mapped through to the ciphertext.



Enigma-rotor

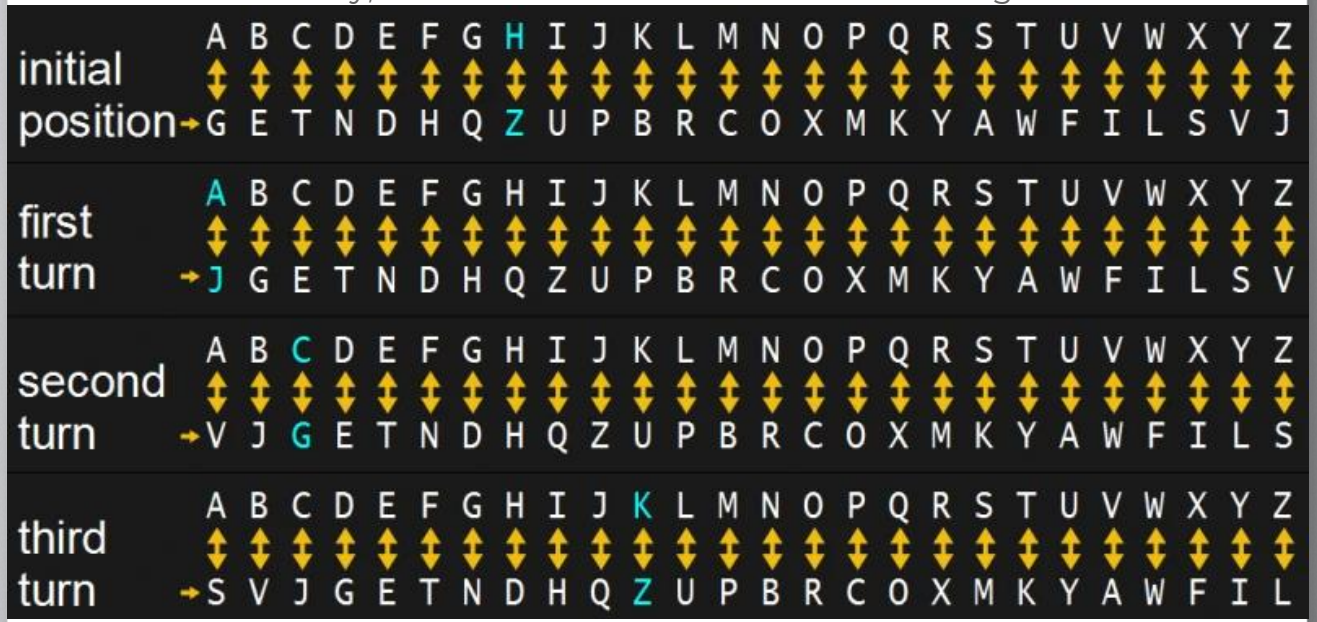
Simple substitution cipher with wires electrically connecting each pair, which opens up the possibility of easily changing how the substitution is mapped through to the ciphertext.





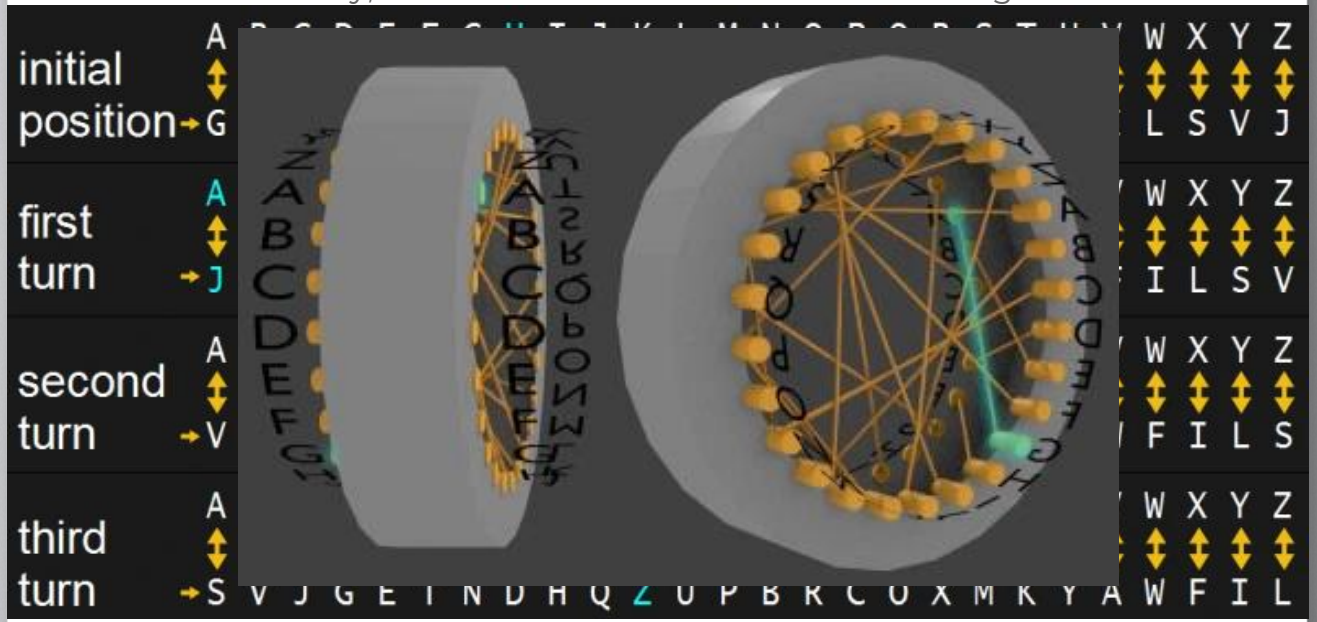
Enigma-rotor

The substitutions change each time a letter is encoded.
Embed all that wiring in a wheel/rotor. By turning the rotor while leaving the letters stationary, the connections between letters change.



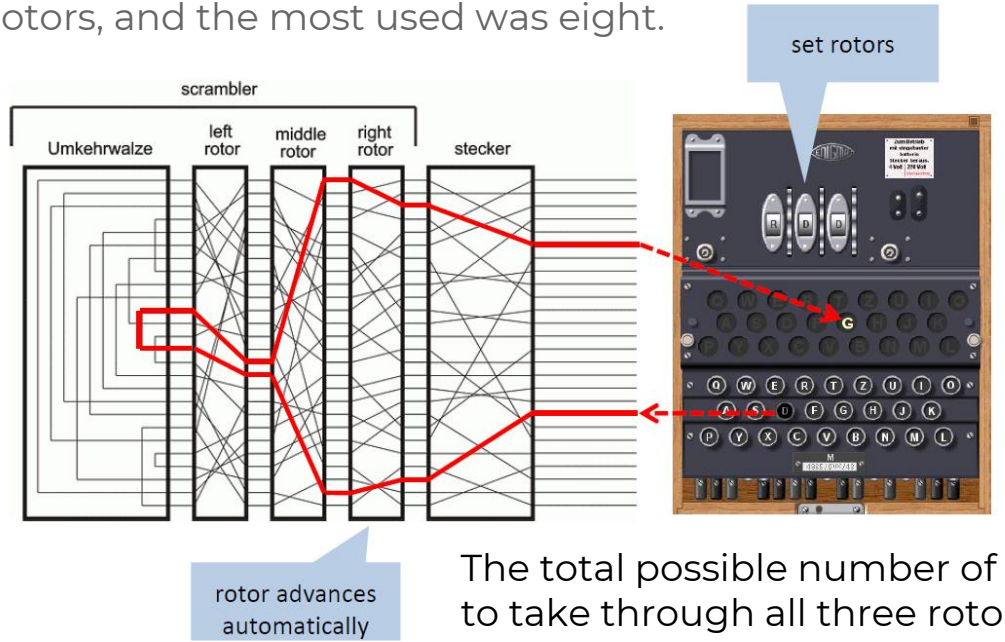
Enigma-rotor

The substitutions change each time a letter is encoded.
Embed all that wiring in a wheel/rotor. By turning the rotor while leaving the letters stationary, the connections between letters change.



Enigma-rotor

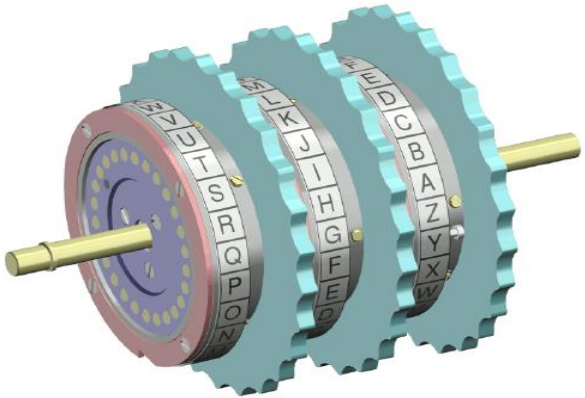
Each of the rotors are wired differently i.e. each rotor contains a different cipher. In some Enigma machines there were three rotors, and the most used was eight.



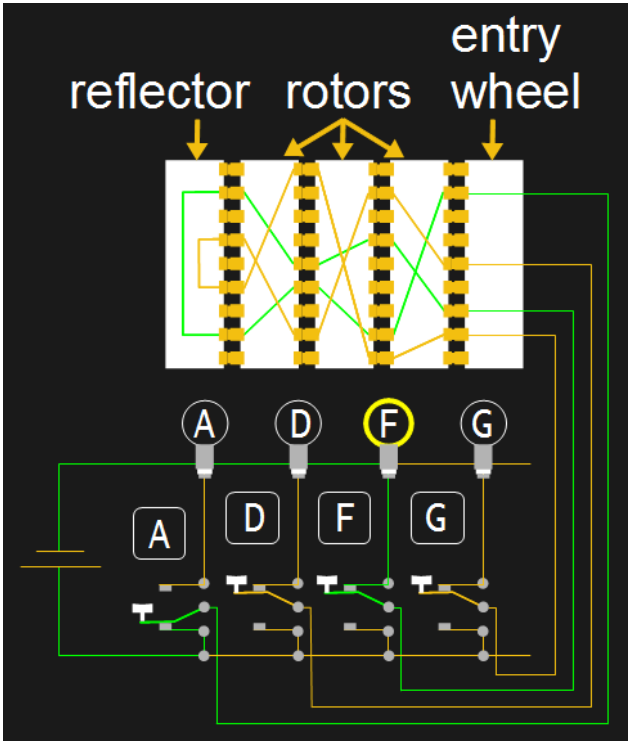
The total possible number of paths for A to take through all three rotors is 26^3.

Enigma-rotor

- To further complicate things for anyone who might have such a machine and be trying to decrypt a message, the rotors are allowed to be moved around before use.
- There are 3×2 possible ways to choose the three rotors.



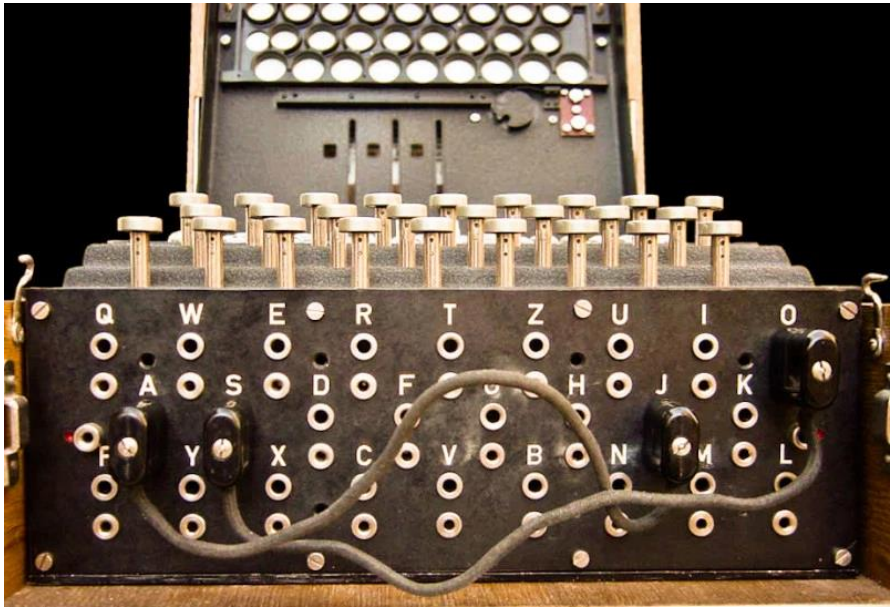
Enigma-rotor



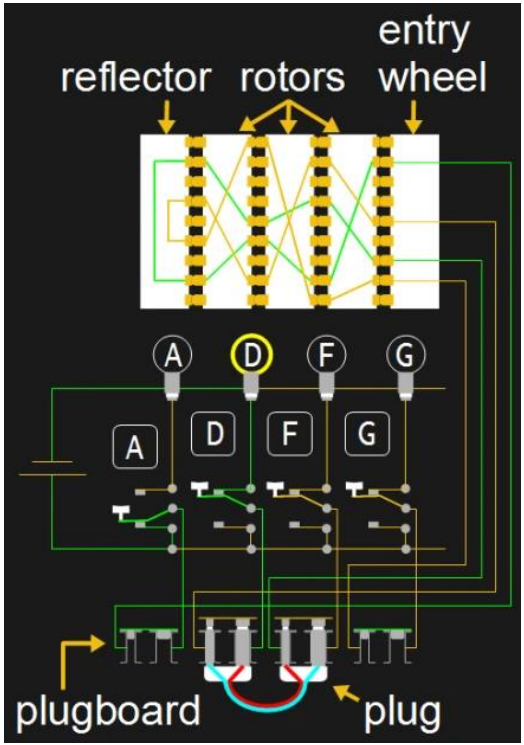
- To make the Enigma easy to use, a circuit including a battery, keyboard, and light-up letters for a display were added.
- The lit F means that the key that was pressed, A, should be substituted with the letter F.
- The first wheel would turn one letter position for each keypress, as illustrated above with the ciphers.

Enigma-plugboard

- In 1930, German army versions added a plugboard, or steckerbrett. This allows letters to be swapped.



Enigma



- In the circuit shown, D and F are swapped. When the A is pressed, the output (with the plugboard) is D.
- Since there are 26 letters, up to 13 swaps could be done.
- The number of possible ways to connect the plugboard is 150,738,274,937,250.

» http://www.louisedade.co.uk/enigma.html

» http://people.physik.hu-berlin.de/~palloks/js/enigma/index_en.html

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**:

$$\frac{\binom{26}{2}\binom{24}{2}\binom{22}{2}\binom{20}{2}\binom{18}{2}\binom{16}{2}}{6!}$$

  - $100391791500 \times 6 \times 26^3 \sim 2^{53}$
  - Security: size of the key space and the randomness of the key setting
- 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 Alan Turing and Marian Rejewski

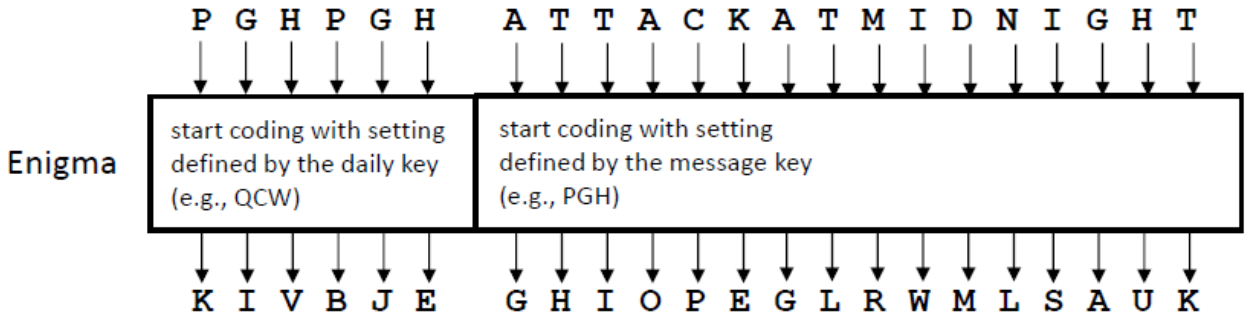
Breaking the Enigma

- every morning, the Germans distribute a daily key to their units to be used with Enigma
  - 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!

Breaking the Enigma

- every morning, the Germans distribute a daily key to their units to be used with Enigma
  - 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



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 →increase 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
  - 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?
  - Enigma doesn't allow letters to be encrypted to themselves.
- 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
- 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 based on the plans of Turing in 1940, which was multiple Enigma rotors chained together, allowing it to rapidly test different key settings.
- indeed, Germans change their message key sending protocol in May 1940, but this does not affect the cryptanalytic capabilities of the British

Breaking the Enigma

- in September 1939, Alan Turing joins the code breakers in Bletchley Park
- h ot rely on
- T day as th
- ons
- ich
- in the
- t ei) ba
- i y 1940, bu anymore

Summary

- The initial attempt at automating the one-time pad failed.
- If the operators had instead rolled dice to decide their initial rotor positions, the starting point in the sequence could have been uniformly distributed.
- If Enigma allowed letters to be encrypted to themselves.
- If there were no usage constraints.

- The outcome of WW II could've been drastically different.

Modern cryptography

- Shannon's work on information theoretical characterization of encryption [1948]

- Substitution-permutation ciphers and the Data Encryption Standard (DES) [1970's]
  - All encryption schemes from ancient times until 1976 were **symmetric** ones.

- The birth of public key cryptography [1976-78]
  - In 1976 public-key (or **asymmetric**) cryptography was openly proposed by Diffie, Hellman and Merkle.

- 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
- 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)

Claude E. Shannon

Perfect Secrecy

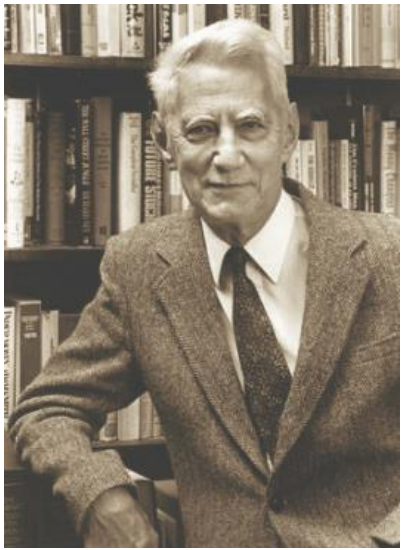
- **A game:**
  - Eve instructs Bob to go into a room, with only a empty of box, a deck of cards, and several locks.
  - Bob needs to select a card from the deck and hide it as best he can.
  - Rule: Bob cannot leave the room with anything, and he can put at most one card in the box.

Perfect Secrecy

- **A game:**
  - Bob selects a card, returns his card to the deck and shuffles the deck to randomize it.
  - Shuffle is the best lock, leaving no information about the choise.
  - The best Eve can do is guess, even we give him/her unlimited compuation power.

Perfect Secrecy

» **Claude Shannon's idea of perfect secrecy: no amount of computational power can help improve your ability to break the one-time pad.**



Communication Theory of Secrecy Systems*

By C. E. SHANNON

1 INTRODUCTION AND SUMMARY

The problems of cryptography and secrecy systems furnish an interesting application of communication theory[1]. In this paper a theory of secrecy systems is developed. The approach is on a theoretical level and is intended to complement the treatment found in standard works on cryptography[2]. There, a detailed study is made of the many standard types of codes and ciphers, and of the ways of breaking them. We will be more concerned with the general mathematical structure and properties of secrecy systems.

The treatment is limited in certain ways. First, there are three general types of secrecy system: (1) concealment systems, including such methods as invisible ink, concealing a message in an innocent text, or in a fake covering cryptogram, or other methods in which the existence of the message is concealed from the enemy; (2) privacy systems, for example speech inversion, in which special equipment is required to recover the message; (3) "true" secrecy systems where the meaning of the message is concealed by cipher, code, etc., although its existence is not hidden, and the enemy is assumed to have any special equipment necessary to intercept and record the transmitted signal. We consider only the third type—concealment system are primarily a psychological problem, and privacy systems a technological one.

Secondly, the treatment is limited to the case of discrete information where the message to be enciphered consists of a sequence of discrete symbols, each chosen from a finite set. These symbols may be letters in a language, words of a language, amplitude levels of a "quantized" speech or video signal, etc., but the main emphasis and thinking has been concerned with the case of letters.

The paper is divided into three parts. The main results will now be briefly summarized. The first part deals with the basic mathematical structure of secrecy systems. As in communication theory a language is considered to be represented by a stochastic process which produces a discrete sequence of

* The material in this paper appeared in a confidential report "A Mathematical Theory of Cryptography" dated Sept 1, 1946, which has now been declassified.

[1] Shannon, C. E., "A Mathematical Theory of Communication," Bell System Technical Journal, July 1948, p.623.

[2] See, for example, H. F. Gaines, "Elementary Cryptanalysis," or M. Givierge, "Cours de Cryptographie."

Perfect Secrecy

- **Claude Shannon's idea of perfect secrecy: no amount of computational power can help improve your ability to break the one-time pad.**

- Example: Alice sends a message to Bob, 20 letters long, using one-time pad. This is equivalent to picking one specific message from a complete collection of all possible 20 letter messages.
  - The size of the message space equals the size of the key space, equals the size of the cipher text space.
  - Given a cipher text, every message is equally likely.

Shannon's Theorem

- **THEROEM 8.1 (Shannon's theorem)** Let (**Gen**,**Enc**,**Dec**) be an encryption scheme over a message space M for which $|M| = |K| = |C|$. This scheme is perfectly secret if and only if:
- 1. Every key $k \in K$ is chosen with equal probability $1/|K|$ by algorithm Gen.
- 2. For every $m \in M$ and every $c \in C$, there exists a single key such that $k \in K$ outputs c.

Uses of Shannon's Theorem

- Theorem 8.1 gives a complete characterization of perfectly-secret encryption schemes.
- the theorem implies that if there exists an encryption scheme that provides perfect secrecy for a specific probability distribution over M then it actually provides perfect secrecy in general (i.e., for all probability distributions over M).
- Shannon's theorem is extremely useful for proving whether a given scheme is or is not perfectly secret.

One-Time Pad

- **(One-Time Pad) A key can be used to encrypt only one plaintext.**
- Secret keys will be strings of bits ,use the variable λ to refer to the length (# of bits) of the secret key in a scheme, keys are elements of the set $\{0, 1\}^\lambda$
- The specific **KeyGen**, **Enc**, and **Dec** algorithms for one-time pad are given below:

Construction 1.1
(One-time pad)

$$\underline{\text{KeyGen:}}$$
$$k \leftarrow \{0,1\}^\lambda$$
$$\text{return } k$$

$$\underline{\text{Enc}(k, m \in \{0,1\}^\lambda):}$$
$$\text{return } k \oplus m$$

$$\underline{\text{Dec}(k, c \in \{0,1\}^\lambda):}$$
$$\text{return } k \oplus c$$

One-Time Pad

- Recall that Key $\{0,1\}^\lambda$ means to sample k uniformly from the set of λ-bit strings.
- All of its security stems from this choice of using the uniform distribution; keys that are chosen differently do not provide equivalent security.

Construction 1.1 (One-time pad)

$$\underline{\text{KeyGen:}}$$
$$k \leftarrow \{0,1\}^\lambda$$
$$\text{return } k$$

$$\underline{\text{Enc}(k, m \in \{0,1\}^\lambda):}$$
$$\text{return } k \oplus m$$

$$\underline{\text{Dec}(k, c \in \{0,1\}^\lambda):}$$
$$\text{return } k \oplus c$$

One-Time Pad-Example

» Encrypting the following 20-bit plaintext m under the 20-bit key k using OTP results in the ciphertext c below:

$$\begin{array}{lll} & \texttt{11101111101111100011} & (m) \\ \oplus & \texttt{00011001110000111101} & (k) \\ \hline & \texttt{11110110011111011110} & (c = \mathrm{Enc}(k, m)) \end{array}$$

Construction 1.1 (One-time pad)

$$\underline{\mathrm{KeyGen}:}$$
$$k \leftarrow \{\texttt{0},\texttt{1}\}^{\lambda}$$
$$\text{return } k$$

$$\underline{\mathrm{Enc}(k, m \in \{\texttt{0},\texttt{1}\}^{\lambda}):}$$
$$\text{return } k \oplus m$$

$$\underline{\mathrm{Dec}(k, c \in \{\texttt{0},\texttt{1}\}^{\lambda}):}$$
$$\text{return } k \oplus c$$

One-Time Pad

- **Correctness:** For all k,m $\in \{0,1\}^{\lambda}$, it is true that Dec(k, Enc(k,m)) = m.
- **Proof :** This follows by substituting the definitions of OTP Enc and Dec, then applying the properties of xor. For all k,m $\in \{0,1\}^{\lambda}$, we have:

$$\begin{aligned}
\mathrm{Dec}(k, \mathrm{Enc}(k, m)) &= \mathrm{Dec}(k, k \oplus m) \\
&= k \oplus (k \oplus m) \\
&= (k \oplus k) \oplus m \\
&= 0^{\lambda} \oplus m \\
&= m.
\end{aligned}$$

One-Time Pad

- **Security:** we have to think about what Alice and Bob do, but from the eavesdropper's point of view.

  From Eve's perspective,

$$\frac{\textsc{eavesdrop}(m \in \{0,1\}^\lambda):}{\begin{array}{l} k \leftarrow \{0,1\}^\lambda \\ c := k \oplus m \\ \text{return } c \end{array}}$$

One-Time Pad

» **Security (Example):**

eavesdrop is a randomized algorithm. You should think of it as a probability distribution over output values. Let's take $\lambda = 3$.

$EAVESDROP(010)$:

Pr	k	output $c = k \oplus 010$
$1/8$	000	010
$1/8$	001	011
$1/8$	010	000
$1/8$	011	001
$1/8$	100	110
$1/8$	101	111
$1/8$	110	100
$1/8$	111	101

$EAVESDROP(111)$:

Pr	k	output $c = k \oplus 111$
$1/8$	000	111
$1/8$	001	110
$1/8$	010	101
$1/8$	011	100
$1/8$	100	011
$1/8$	101	010
$1/8$	110	001
$1/8$	111	000

One-Time Pad

- **Security:** For every m $\in \{0,1\}^\lambda$, the distribution eavesdrop(m) is the uniform distribution on $\{0,1\}^\lambda$. Hence, for all m, m'$\in\{0,1\}^\lambda$,the distributions eavesdrop(m) and eavesdrop(m') are identical.
- **Proof:** Arbitrarily fix m, c $\in \{0,1\}^\lambda$. We will calculate the probability that eavesdrop(m) produces output c. That event happens only when $c = k \oplus m \iff k = m \oplus c.$

The equivalence follows from the properties of xor. That is,

$$\Pr[\textsc{eavesdrop}(m) = c] = \Pr[k = m \oplus c]$$

One-Time Pad

» **Proof (continued)** The equivalence follows from the properties of xor. That is, $\Pr[\text{EAVESDROP}(m) = c] = \Pr[k = m \oplus c]$

where the probability is over uniform choice of $k \leftarrow \{0,1\}^\lambda$.

Since k is chosen uniformly from $\{0,1\}^\lambda$, the probability of choosing the particular value $k = m \oplus c$ is $1/2^\lambda$.

In summary, for every m and c, the probability that eavesdrop(m) outputs c is exactly $1/2^\lambda$. This means that the output of eavesdrop(m), **for any m, is distributed uniformly.**

One-Time Pad

» **Limitations:**

- One-time pad is incredibly limited in practice. Most notably: If two users want to privately convey a λ-bit message, they first need to privately agree on a λ-bit string. A key can be used to encrypt only one plaintext (hence, “one-time” pad) .

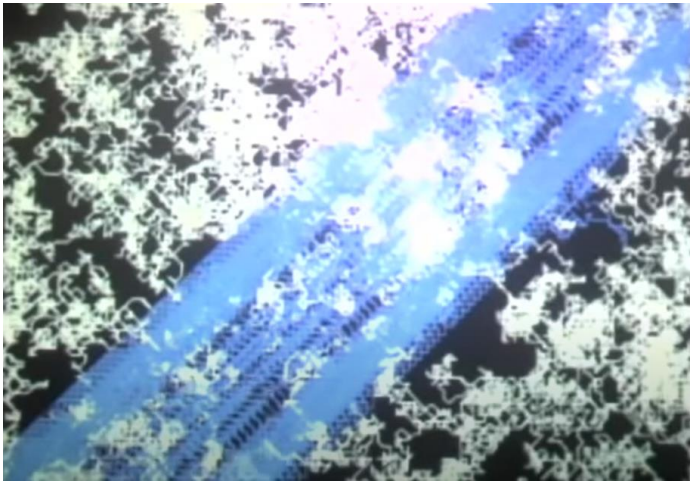
How to share these long keys in advance?

Pseudo-randomness

- There are random fluctuations in the physical world everywhere.
- Random processes are nondeterministic, since they are impossible to determine in advace.
- Machine are deterministic. There operations are predictable and repeatable.
- Neumann develop an algorithm (middle-squares) to mechanically simulate randomness :
  - 1) select a truly random number, called the “seed”;
  - 2) this seed is provided as input to a simple calculation. Multiply the seed by itself. And output the middle of this result.
  - 3) Use the output as the next seed and repeat the process as many times as needed.

Pseudo-randomness

- Random vs. pseudorandom
  - Random walk: the pseudorandom sequence must eventually repeat when the algorithm reaches a seed that has been used.
  - The length before the pseudorandom sequence repeats is called the period (strictly limited by the length of the initial seed).



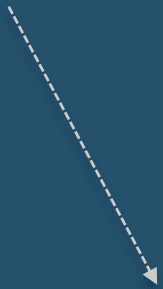
Pseudo-randomness

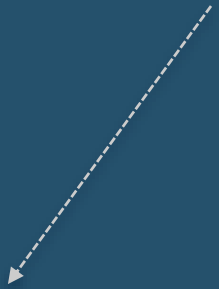
- Random vs. pseudorandom
  - When we move from random to pseudorandom shifts. We shrink the key space into a much smaller seed-space.
  - For a pseudorandom sequence to be indistinguishable from a random sequence, it must be impractical for a computer to try all seeds and look for a match.
  - Possible vs. possible in a reasonable amout of time.

- Now, Alice and Bob can share a ramdom seed in advace.

Kerckhoffs' principle

one-time pad (OTP)

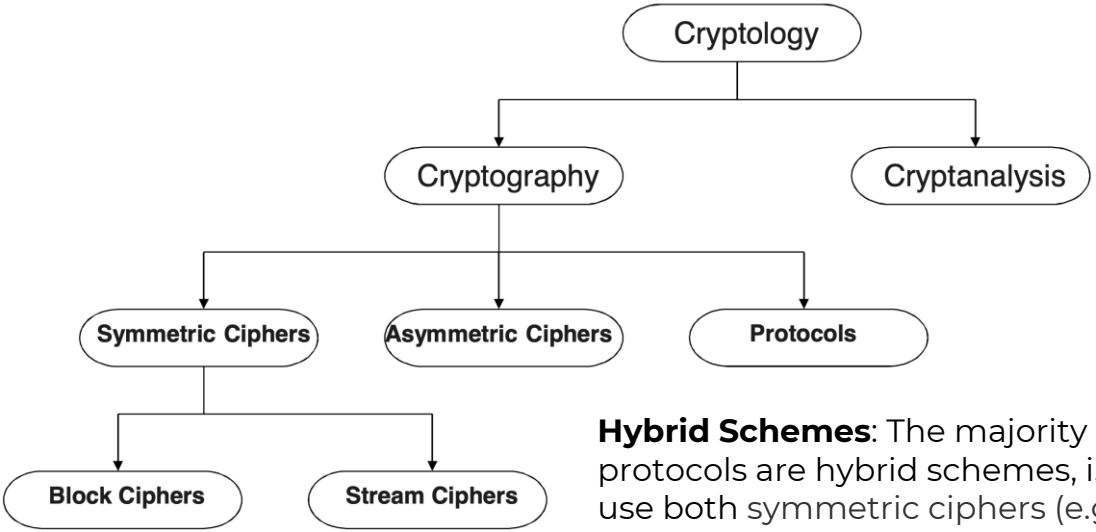




only 2 useful ideas from before 1900

Classification of the Field of Cryptology

Overview on the field of cryptology



Hybrid Schemes: The majority of today's protocols are hybrid schemes, i.e., the use both symmetric ciphers (e.g., for encryption and message authentication) and asymmetric ciphers (e.g., for key exchange and digital signature).

Why do we need Cryptanalysis?

- There is no *mathematical proof of security* for any practical cipher
- The only way to have assurance that a cipher is secure is to try to break it (and fail) !
- **Kerckhoff's Principle** is paramount in modern cryptography:

> A cryptosystem should be secure even if the attacker (Oscar) knows all details about the system, with the exception of the secret key.

- In order to achieve Kerckhoff's Principle in practice: **only use widely known ciphers that have been cryptanalyzed for several years by good cryptographers!** (*Understanding Cryptography* only treats such ciphers)

Cryptanalysis: Attacking Cryptosystems

- **Classical Attacks:**
  - Mathematical Analysis
  - Brute-Force Attack

- **Implementation Attack:** Try to extract key through reverse engineering or power measurement, e.g. , for a banking smart card.

- **Social Engineering:** E.g., trick a user into giving up her password

Brute-Force Attack (or Exhaustive Key Search)

- Treats the cipher as a black box
- Requires (at least) 1 plaintext-ciphertext pair (x_0,y_0)
- Check all possible keys until condition is fulfilled:

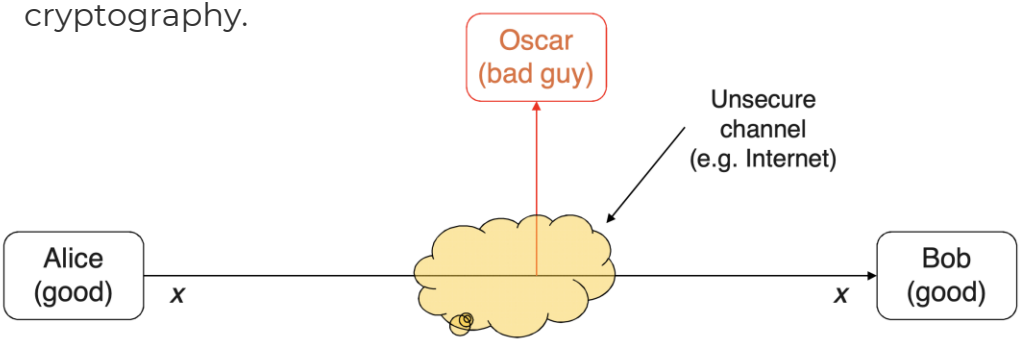
$$d_k(y_0) = x_0$$

- How many keys to we need?

Key length in bit	Key space	Security life time (assuming brute-force as best possible attack)
64	2^{64}	**Short term** (few days or less)
128	2^{128}	**Long-term** (several decades in the absence of quantum computers)

Symmetric Cryptography

» Alternative names: **private-key**, **single-key** or **secret-key** cryptography.



» **Problem Statement:**

- Alice and Bob would like to communicate via an unsecure channel (e.g., WLAN or Internet).
- A malicious third party Oscar (the bad guy) has channel access but should not be able to understand the communication.

Symmetric Cryptography

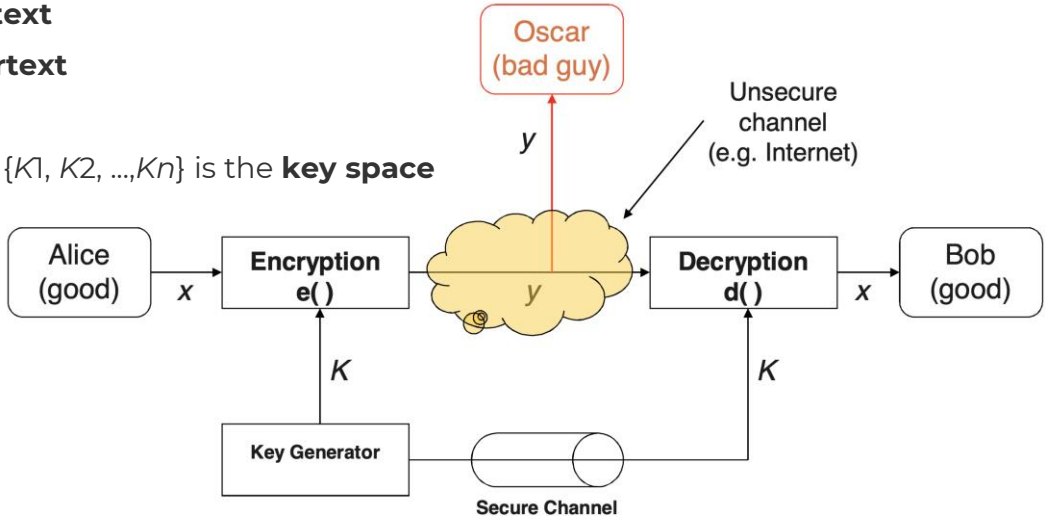
- Solution: Encryption with symmetric cipher.
- Oscar obtains only ciphertext y, that looks like random bits

x is the. **plaintext**

y is the **ciphertext**

K is the **key**

Set of all keys {*K*1, *K*2, ...,*Kn*} is the **key space**



Symmetric Cryptography

Encryption equation	$y=e_k(x)$
Decryption equation	$x=d_k(y)$

- Encryption and decryption are **inverse operations** if the same key K is used on both sides:

$$d_k(y)= d_k\,(e_k(x))=x$$

- Important: the key must be transmitted via a **secure channel** between Alice and Bob.
- However, the system is only secure if an attacker does not learn the key K!
- **The problem of secure communication is reduced to secure transmission and storage of the key K.**

THANKS!

Any questions?

You can find me at:

- zhanglan@ustc.edu.cn