What is cryptography?

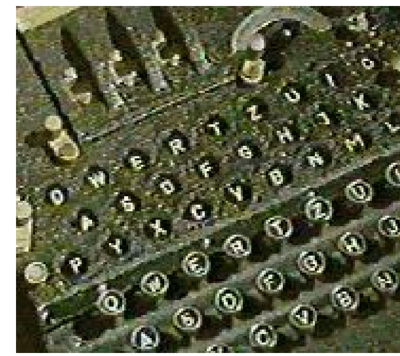
- Cryptography: from Greek kryptos, hidden, and graphein, to write, used to be the “art of secret writing”
- Today, it is a formalized discipline in the fields mathematics and information theory
- Cryptography is commonplace (in your favorite videogame console; in your cellphone; in your browser; in your car; on the underground)

Hystory of crypto

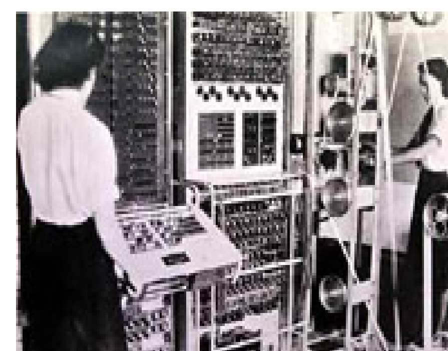
- In ancient history, no need (writing was already a “secret technique”)
- In the Greek society writing became more common, and hidden writing became a need. E.g. the “scytale”, the wand of command of the commanders of Athens. According to Plutarco, in use since the IX century b.C.
- Medieval studies (up to and including Gabriele Lavinde, who wrote a manual in 1379, a copy is available at the Vatican archives). A large number of contributions from Italy (L.B.Alberti, G.B.Porta, G.B.Bellaso, G.Cardano).

Crypto as military stuff

- Obvious military interest in encrypting things, since the Scytale times!
- Italian Army General **Luigi Sacco** wrote a famous “Nozioni di crittografia” book in 1925, one of the last “non-formalized” exercises in cryptography



- During WW II, **Alan Turing**, father of formal computer science, worked at Bletchley Park to break Axis ciphers, dealing a serious blow to the enemy by breaking the Enigma cipher



- Birth of the first universal computers was stimulated by this effort
- “Cryptonomicon” by Stephenson

Key concept of formalization of cryptosystems

- They were formalized by Claude Shannon (the inventor of modern information theory), in his 1949 paper “Communication theory of secrecy systems”
- We call “cryptosystem” a system which takes in input a message (known as plaintext) and transforms it into a ciphertext with a reversible function/algorithm which usually takes a key as a further input
- The function should not be reversible without the key (more formal on the next slide)
- The use of “text” is historical, and today we mean “a string of bits”

Kerckhoffs principle

- Published in the book “La criptographie militaire” in 1883. Also used by Shannon, so sometimes it's called “Kerchoffs/Shannon”
- The security of a cryptosystem relies only on the secrecy of the key, and never on the secrecy of the algorithm (i.e. the algorithm should always be supposed public and known to the attacker)
- This means that in a secure cryptosystem we cannot retrieve the plaintext from the ciphertext without the key, and we cannot retrieve the key itself

Symmetric ciphers

- First basic form of encryption
- The same key is used to encrypt and decrypt messages
- Synonims: shared key ciphers; secret key ciphers
- Issue: how do we agree on the key?
- Off-band transmission mechanism needed!

Building a cipher: substitution

- Replacing each byte with another
- Example (Cesar cipher): replace a letter with the one following it by *n* positions

```
ABCDEFGHIJKLMNOPQRSTUVWXYZ
XYZABCDEFGHIJKLMNOPQRSTUVW
```

- E.g.: the word “SECURE” becomes “VHFXUH”
- The “key” is 3, if the cipher is known, bruteforcing it takes 25 attempts at most, 13 on average. Keyspace way too small! This is a toy example.
- Repetitions and structure show up! Monoalphabetic cipher, could be polyalphabetic (positional)

Building a cipher: transposition or diffusion

- Example: characters in a matrix, written by rows, read by columns.
- The “key” is the dimensions (in this case K=[3,5])
- Not to be trivial, the product must be smaller than the message

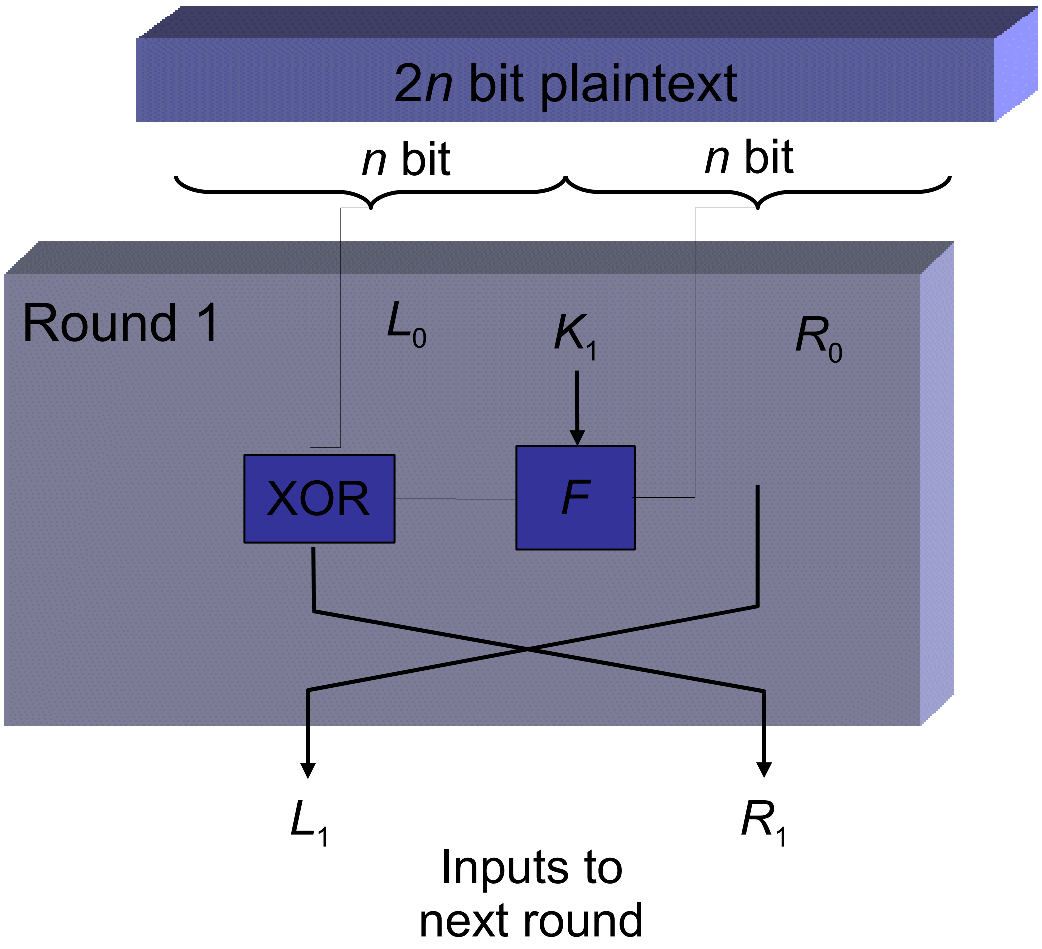
- Example
- CIAO A TUTTI (row-wise)
- CATI IAT OU T (column-wise)

C	I	A	O	
A		T	U	T
T	I			

Modern symmetric ciphers

- Modern ciphers mix diffusion and substitution
- Some well known ciphers
  - Feistel (1973)
  - DES (Data Encryption Standard, 1977), 3DES
  - IDEA (1991)
  - BlowFish (1993)
  - RC5 (1994)
  - CAST-128 (1997)
  - Rijndael (since 2000 it is the AES, Advanced Encryption Standard)

What is a Feistel cipher



F: round function

K_1: round subkey, derived from *main key* *K* through a *scheduling function*

Decryption works by inverting the flow

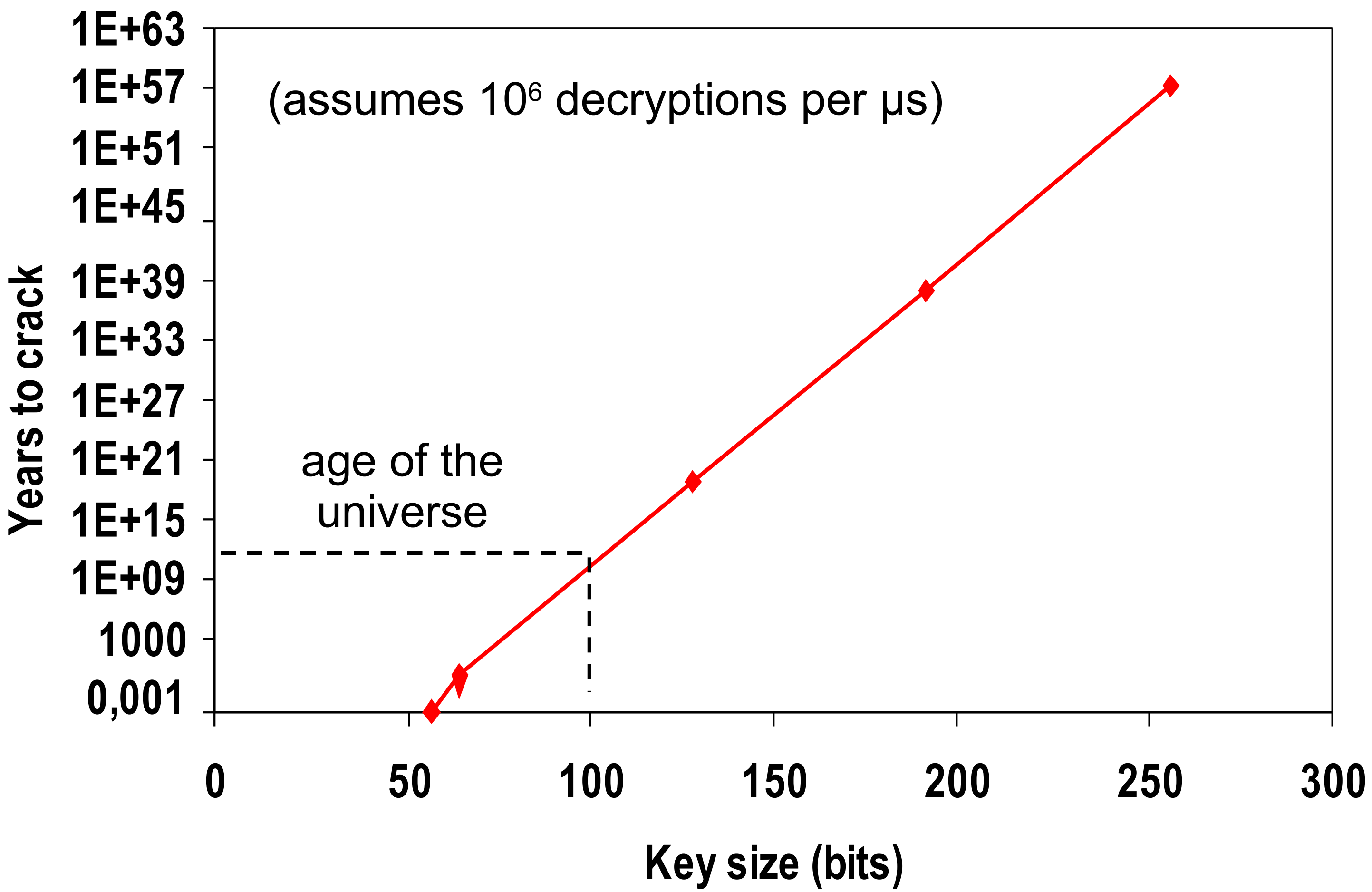
DES: Data Encryption Standard (1977)

- Originally designed in 1970 by an IBM subsidiary
- In 1977 it became a US government standard
- It's a Feistel cipher working on 64-bit blocks, using 16 rounds. Function F is an S-box (a simple substitution)
- S-Boxes were “redesigned” by the NSA
- It uses a 56 bit key (2^{56} combinations), it is actually a 64-bit long string with parity checks
- It was discovered long after that the S-box combination suggested by NSA made the DES specifically vulnerable to a then-unknown cryptanalitic technique dubbed *differential cryptanalisis*
- But more importantly, today DES is insecure because of the limited key size (3DES solves this by encrypting, decripting and encrypting again with 2 or 3 different keys, for a total keysize of 112 or 168 bits)

Keyspace and brute force attacks

- We have seen the concept of a brute force attack: trying all possible keys
- We will see that bruteforcing is applicable against any real world algorithm. So, an algorithm is broken *if there is an attack faster than brute force which breaks the system*
- The smaller the keyspace, the easier the attack (remember: the algorithm itself is known!)
    - We have seen it in the toy Cesar cipher example
- Brute force attack time is generally measured in “bits of keyspace”, and the time is exponential on the number of bits (i.e. 33 bits need twice the time of 32)
- This creates a need to balance key length vs. computational power

Key Security



Asymmetric cryptography

- Introduced in 1976 by W.Diffie and M.Hellmann
- The key concept is having a cipher which uses two keys: what is encrypted with key 1 can be decrypted only with key 2 (and not with key 1 itself), and viceversa
- The keys cannot be retrieved from each other
- Also called “public key cryptography”, because the idea is that one of those two keys is kept private by the subject, and the other can be publicly disclosed
- This solves, as we will see, the problem of key exchange
- They use a one-way function with a trapdoor
- They are usually computation-intensive, so they are often combined with symmetric ciphers as we will see

A few common asymmetric ciphers

- Diffie-Hellman (1976)
- RSA (1977, Ron Rivest, Adi Shamir, Len Adleman)
- DSS (1991, FIPS PUB 186)
- ECC (IEEE P1363, elliptic curve cryptography)

A simple example to understand: Diffie-Hellman

- D-H can be used by Alice and Bob to agree on a secret key over an insecure channel
- The one-way trapdoor is the modular logarithm
- If $y=a^x$ then $x=\log_a y$ – Maths 101
- Well, it turns out that given x, a, p it is easy to compute $y=a^x \bmod p$, but knowing that $y = a^x \bmod p$ it is *difficult* to compute x
- When we say difficult, we mean computationally very intensive, so intensive that the problem is not solvable for all practical purposes and it requires *bruteforce*

Diffie Hellman: the trick

- *p*:prime, *a*:primitive root of p, known, public

A --> B her public key Y_A ($Y_A = a^{X_A} \bmod p$) computed from her private key X_A, which she keeps secret

B --> A his public key Y_B ($Y_B = a^{X_B} \bmod p$), likewise

- A computes $K_A = (Y_B)^{X_A} \bmod p$
- B computes $K_B = (Y_A)^{X_B} \bmod p$
- $K_A = (Y_B)^{X_A} = (a^{X_B})^{X_A} = (a^{X_A})^{X_B} = (Y_A)^{X_B} = K_B$!!!!!
- K can be computed either using the private key of A and the public key of B, or viceversa, but not from the two public keys!
- To retrieve the private key from the public key I would need to break the modular log problem

Diffie Hellman: example with small numbers

- $p = 7$: prime, public
- X chosen in (1, 2, ..., p-1), secret
- $a = 3$: prime root of p (also public), i.e.

$$a^X \bmod p = (1, 2, ..., p-1)$$

$$3^X \bmod 7 = 1, 2, ..., 6 \quad . \; X \in (1,2, ..., 6)$$

$3^1 mod7=3$, $3^2 mod7=2$, $3^3 mod7=6$, $3^4 mod7=4$, $3^5 mod7=5$, $3^6 mod7=1$

- A: $X_A=3$ secret, $Y_A= a^{X_A} \bmod p = 3^3 \bmod 7 = 6$, public
- B: $X_B=1$ segreto, $Y_B= a^{X_B} \bmod p = 3^1 \bmod 7 = 3$, public
- A -> B $Y_A=6$, B -> A $Y_B=3$
- A gets $K = (Y_B)^{X_A} \bmod p = (3)^3 \bmod 7 = 6$
- B gets $K = (Y_A)^{X_B} \bmod p = (6)^1 \bmod 7 = 6$

equal

The RSA algorithm (hints)

- Same trick, different base problem
- Given two large primes, p and q, computing $n=pq$ is easy, whereas given n it is painfully slow to get p and q
- A different problem from mod log, but it can be shown that they are related (so if one is ever solved, the other will follow closely)
- Breaking n in p and q is slower for larger n, but also the computation time for encryption is slower (linearly, whereas the complexity is exponential)
- At the moment of writing anything larger than 600 bits is not practical to factor

Message Digest

- A message digest is a “check code” to verify the integrity of a message
- Also called “checksum” for historic reason
- The digest is computed by applying a one-way function called *hash* to the message
- Commonly used functions are SHA-1 (160 bit hashes) and MD5 (128 bit hashes)
- Examples: md5sum or sha1sum command line utilities under most Linux distributions

Hash Function characteristics

- An hash function H produces a fixed-size output from an arbitrary-size input
- For each x, $H(x)$ must be computationally light
- It must be computationally infeasible to find:
  - x s.t. $H(x) = h$, a specific digest
  - y s.t. $y \neq x$ and $H(y) = H(x)$, with a given x
  - couples $\{x,y\}$ s.t. $H(x) = H(y)$
  - This last property is known as "collision-free property", but an hash function cannot (obviously!) be collision-free (because it maps a larger domain onto a smaller domain!)

How crypto breaks

Cryptanalysis

- Discipline which tests the robustness of crypto algorithms, and which tries to break them
- Mathematics tells us that there is only one absolutely secure cipher
- Which, unsurprisingly, it's almost impossible to use...
- Note: the following slides are not mathematically complete and sound, they are just a sketch of a more complete demonstration

Some basic definitions

- Let's model the message generation as a stochastic process
  - M = random variable which models the choice of message m
  - C = random variable which models sending ciphertext c
  - K = random variable which models the choice of secret key k
- The *process characteristics* obviously depend on the sender
  - E.G. frequence of letters, or of n-grams, in the language; use of certain specific words with more frequency; format of the messages in a given network protocol...

Definition of a perfect cipher

- $P(M = m)$ = probability that the sender wants to send message m (prior probability)
- $P(M=m \mid C = c)$ = probability that the sender has sent message m , given that c was observed
- The attacker knows
  - The *a priori* probability with which messages are sent
  - The encryption and decryption functions
  - He's missing only the secret key k
- The cipher is a perfect cipher if and only if

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

Shannon's theorem on perfect ciphers

In a perfect cipher the number of keys must be greater or equal to the number of possible messages

Sketch of proof:

- Obviously, $|Criptograms| \geq |Msg|$.
- Also for each m, $P(M = m) > 0$.
- Proof by absurd: suppose that $|Keys| < |Msg|$, then it follows $|Keys| < |Criptograms|$.
  - Intuitively, taken a message m, since there are less keys than cryptograms, there are some c_m which are not images of m.
  - If we observe c_m , it follows that the message is not m.

$$P(M=m \mid C = c) = 0 \qquad [\neq \; P(M = m)\,]$$

One-time pad

- A perfect and minimal "stream cipher"
- XOR of a message m and a random key k as long as m (n bits)
- *Minimal* since |Keys| = |Msg|
- *Perfect* since:
  - $P(M = m \mid C = c) = P(M=m, C=c)/P(C=c)$
  - $P(M=m, C=c) = P(M=m, C=c, K=k_{m,c}) =$

$$= P(C=c \mid M=m, K=k_{m,c}) * P(M=m \mid K=k_{m,c}) * P(K=k_{m,c})$$

$$= 1 * P(M=m) * (1/2)^n$$

  - $P(C=c) = \Sigma_m P(C=c, M=m) = \Sigma_m P(M=m)*(1/2)^n = (1/2)^n$
  - $P(M = m \mid C = c) = P(M=m)*(1/2)^n / (1/2)^n = P(M=m)$
- Still quite useless except in very special cases...

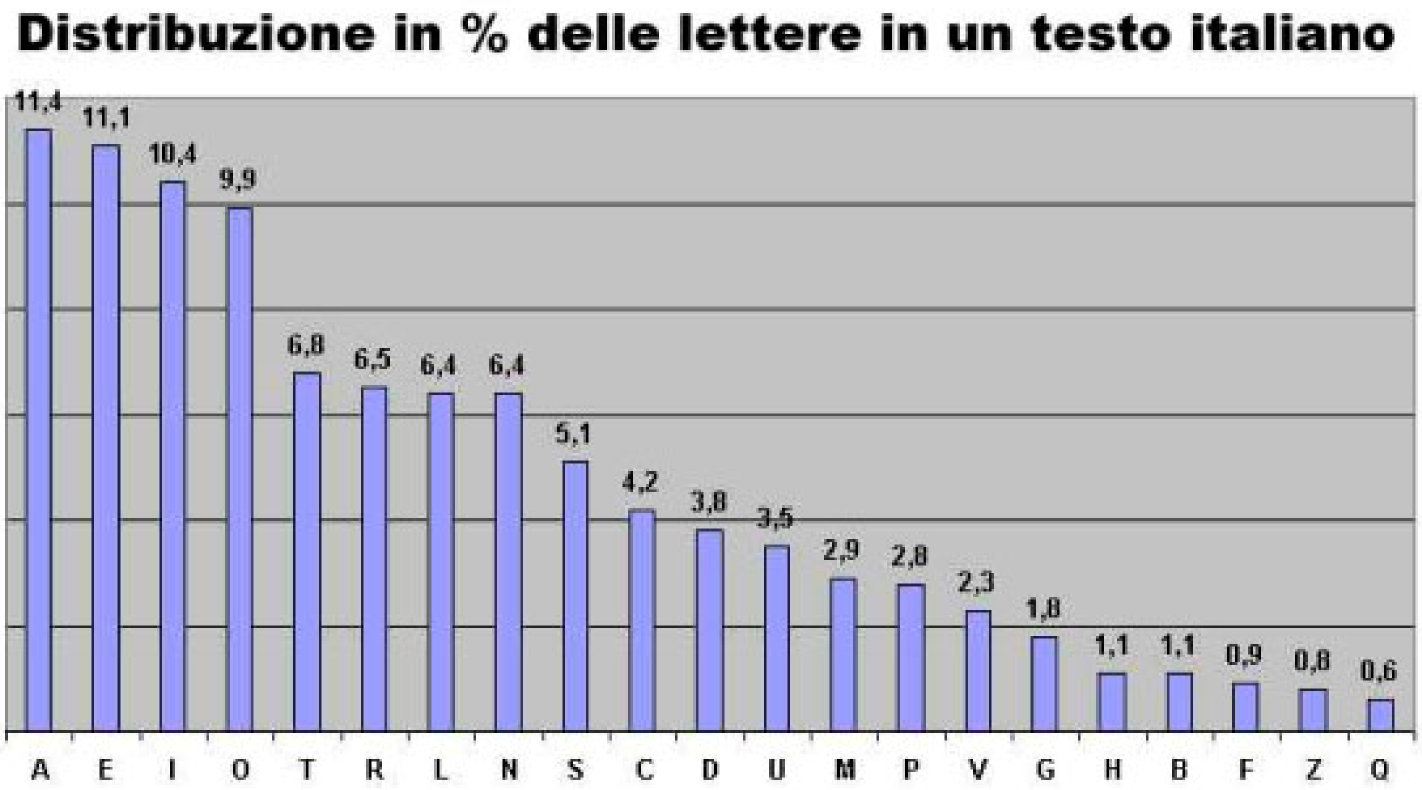
Cryptanalysis basics

- Cryptosystems try to approximate a one-time pad, but they aren't really one time
- Each pair ciphertext/plaintext gives a small amount of information
- Bruteforcing is possible (it isn't with OT pads)
- A real (non perfect) cryptosystem is broken if there is a way to break it which is faster than bruteforcing
- Types of attacks:
  - Cipher Text Attack (the analyst has only ciphertexts)
  - Known Plain-text Attack (the analyst has a set of pairs plain/ciphertext)
  - Chosen Plain-Text Attack (the analyst can choose plaintexts and get them back encripted)

Examples of attack techniques

- Statistical Cryptanalysis: based on distribution of characters. Obviously with polyalphabetic substitution and diffusion significantly less effective
- Linear Cryptanalysis: based on linear relations between input and output bits
  - If we find a relationship which holds in more than 50% of the cases, let p = .5 + 1/M this probability, the key can be derived in M^2 messages
- Differential Cryptanalysis: based on relationships between the bit-per-bit xor of different plaintexts and the corresponding ciphertexts

Example of statistical cryptanalysis



Possible attacks to hash functions

- If we are able to find:
  - x s.t. $H(x) = h$, a specific digest, or equivalently y s.t. $y \neq x$ and $H(y) = H(x)$, with a given x
    - Then we have an arbitrary collision, or preimage attack
    - It must happen more easily than the attempts it takes to do it randomly with a n-sized hash function (2^{n-1})
  - couples $\{x,y\}$ s.t. $H(x) = H(y)$ more easily than maths would predict, we have a simplified collision attack
    - In this case, the random event happens in ($2^{n/2}$) for the birthday paradox

Key points

- Security of a cryptosystem is based on the robustness of the algorithm
- No algorithm save the one-time-pad is invulnerable
- An algorithm is broken if there is at least one attack faster than bruteforcing
- There is no way to prove robustness of a cipher save by trying to break it: open source and sharing are fundamental for cryptography
- Secret algorithms are insecure. Security is transparency.

Looking at the problem in the proper perspective

"You have probably seen the door to a bank vault… 10-inch thick, hardened steel, with large bolts… We often find the digital equivalent of such a vault door installed in a tent. The people standing around it are arguing over how thick the door should be, rather than spending their time looking at the tent."

(Niels Ferguson & Bruce Schneier, *Practical Cryptography*)