T-79.4501
Cryptography and Data Security

Lecture 2:

2.1 Classical cryptosystems

2.2.Introduction to modern cryptographic primitives

- Birthday Paradox

Text book: Chapter 2

2.1 Classical Cryptosystems

Ceasar Cipher, or Shift Cipher

Plain: `meet me after the toga party`

Cipher: `PHHW PH DIWHU WKH WRJD SDUWB`

Alphabets

Plain: `abcdefghijklmnopqrstuvwxyz`

Cipher: `ABCDEFGHIJKLMNOPQRSTUVWXYZ`

Or, Plain: `0123456789...      24,25`

Cipher: `0123456789...      24,25`

Substitution is a mapping from ”Plain” to ”Cipher”

Caesar cipher

p = plaintext letter, $\{0,1,2,\ldots,25\} \ni p$

C = ciphertext letter , $\{0,1,2,\ldots,25\} \ni C$

Caesar substitution E

E: $C = E(p) = (p + 3) \bmod 26$

0 -> 3; 1 -> 4; …

22 -> 25; 23 -> 0; 24 -> 1; 25 -> 2

Caesar substitution, inverse transformation D

D: $p = D(C) = (C - 3) \bmod 26$

0 -> 23; 1 -> 24; 2-> 25; 3 -> 0; … ; 25 -> 22

Brute force cryptanalysis of shift cipher

Shift cipher: *E: C* = *E*(*p*) = (*p* + *K*) mod 26

K = key; {0,1,2,3,…,25} э *K*

We need only some piece of ciphertxt to do exhaustive search for the key

K	`PHHW PH DIWHU WKH WRJD SDUWB`
1	`oggv ..`
2	`nffu ..`
3	`meet me after the toga party`

Monoalphabetic substitution

Alphabets

Plain: `abcdefghijklmnopqrstuvwxyz`

Cipher: `ABCDEFGHIJKLMNOPQRSTUVWXYZ`

Key = permutation of the 26 characters

Size of key space 26! $\cong 4 \times 10^{26}$

Not possible to do exhaustive search over the key space

Cryptanalysis is based on statistical properties of the plaintext

Relative Frequency of Letters in English

A	8.167
B	1.492
C	2.782
D	4.253
E	12.702
F	2.228
G	2.015
H	6.094
I	6.996
J	0.153
K	0.772
L	4.025
M	2.406

N	6.749
O	7.507
P	1.929
Q	0.095
R	5.987
S	6.327
T	9.056
U	2.758
V	0.978
W	2.360
X	0.150
Y	1.974
Z	0.074

Ciphertext obtained from a Substitution Cipher

```
YIFQF MZRWQ FYVEC FMDZP CVMRZ
WNMDZ VEJBT XCDDU MJNDI FEFMD
ZCDMQ ZKCEY FCJMY RNCWJ CSZRE
XCHZU NMXZN ZUCDR JXYYS MRTME
YIFZW DYVZV YFZUM RZCRW NZDZJ
JXZWG CHSMR NMDHN CMFQC HZJMX
JZWIE JYUCF WDJNZ DIR
```

Frequency table

A	0
B	1
C	15
D	13
E	7
F	11
G	1
H	4
I	5
J	11
K	1
L	0
M	16

N	9
O	0
P	1
Q	4
R	10
S	3
T	2
U	5
V	5
W	8
X	6
Y	10
Z	20

Simple substitution: frequency analysis cont'd

The most frequent character: `Z`

The most frequent character in English: e

Guess: D(`Z`) = e

The next most frequent characters

`{M,C,D,F,J,R,Y,N}`

The next most frequent characters in English

`{t,a,o,i,n,s,h,r}`

The most frequent digrams with `Z` are:

`DZ`, `ZW` (4 times); `NZ,ZU` (3 times);

`RZ,HZ,XZ,FZ,ZR,ZV,ZC,ZD` (two times each)

Using common digrams…

`NZ` is common but `ZN` occurs only once;

guess D(`N`) = h

`ZW` is common and `WZ` not at all and `W` is rare;

guess D(`W`) = d

`DZ`(4 times) and `ZD` (2 times) are both common we guess `{r,s,t}` ∍ D(`D`)

`ZRW` and `RZW` occur, and `RW` occurs, and `R` is frequent we guess D(`R`) = n

Now we have

```
       end           e     ne dh  e
YIFQF MZRWQ FYVEC FMDZP CVMRZ WNMDZ VEJBT
        h         e     e           nh d
XCDDU MJNDI FEFMD ZCDMQ ZKCEY FCJMY RNCWJ
  en     e  h  eh      n        n       ed
CSZRE XCHZU NMXZN ZUCDR JXYYS MRTME YIFZW
   e    e   ne nd he e    ed      n h   h
DYVZV YFZUM RZCRW NZDZJ JXZWG CHSMR NMDHN
       e     ed         d  he   n
CMFQC HZJMX JZWIE JYUCF WDJNZ DIR
```

`ne_ndhe` suggests that D(`C`) = a

```
       end      a    e  a  ne dh  e
YIFQF MZRWQ FYVEC FMDZP CVMRZ WNMDZ VEJBT
 a      h         ea     ea    a    nhad
XCDDU MJNDI FEFMD ZCDMQ ZKCEY FCJMY RNCWJ
a en   a e  h  eh   a n        n       ed
CSZRE XCHZU NMXZN ZUCDR JXYYS MRTME YIFZW
   e    e   neand he e    ed  a   n h   h
DYVZV YFZUM RZCRW NZDZJ JXZWG CHSMR NMDHN
a   a  e     ed      a  d  he   n
CMFQC HZJMX JZWIE JYUCF WDJNZ DIR
```

nh_ encrypts to RNM suggests that $D(\mathtt{M})$ = i or o

We have:

```
      iend      a  i e  a ine dhi e
YIFQF MZRWQ FYVEC FMDZP CVMRZ WNMDZ VEJBT
 a    i h      i  ea i   ea    a i  nhad
XCDDU MJNDI FEFMD ZCDMQ ZKCEY FCJMY RNCWJ
a en   a e  hi eh   a n       in i     ed
CSZRE XCHZU NMXZN ZUCDR JXYYS MRTME YIFZW
   e    e i neand he e    ed  a  in hi  h
DYVZV YFZUM RZCRW NZDZJ JXZWG CHSMR NMDHN
ai  a  e i   ed      a  d  he   n
CMFQC HZJMX JZWIE JYUCF WDJNZ DIR
```

Guess {D,F,J,Y} $\ni$ E(o), then Y is the most likely

```
o     iend   o  a  i e  a ine dhi e
YIFQF MZRWQ FYVEC FMDZP CVMRZ WNMDZ VEJBT
 a    i h      i  ea i   ea o  a io nhad
XCDDU MJNDI FEFMD ZCDMQ ZKCEY FCJMY RNCWJ
a en   a e  hi eh e a n   oo  in i  o  ed
CSZRE XCHZU NMXZN ZUCDR JXYYS MRTME YIFZW
 o e  o e i neand he e    ed  a  in hi  h
DYVZV YFZUM RZCRW NZDZJ JXZWG CHSMR NMDHN
ai  a  e i   ed    o a  d  he   n
CMFQC HZJMX JZWIE JYUCF WDJNZ DIR
```

Remaining {D, F, J} possibly decrypt to {r, s, t }

Try D(D)= s, D(F)= r, D(J)= t

```
o r r iend  ro  a rise  a ine dhise   t
YIFQF MZRWQ FYVEC FMDZP CVMRZ WNMDZ VEJBT
 ass  iths  r ris easi   ea   rati  nhadt
XCDDU MJNDI FEFMD ZCDMQ ZKCEY FCJMY RNCWJ
a en   a e  hi eh   asn t oo   in i o red
CSZRE XCHZU NMXZN ZUCDR JXYYS MRTME YIFZW
so e   re i neand heset   ed  a  in his h
DYVZV YFZUM RZCRW NZDZJ JXZWG CHSMR NMDHN
air a  eti  ted   to ar dsthe s n
CMFQC HZJMX JZWIE JYUCF WDJNZ DIR
```

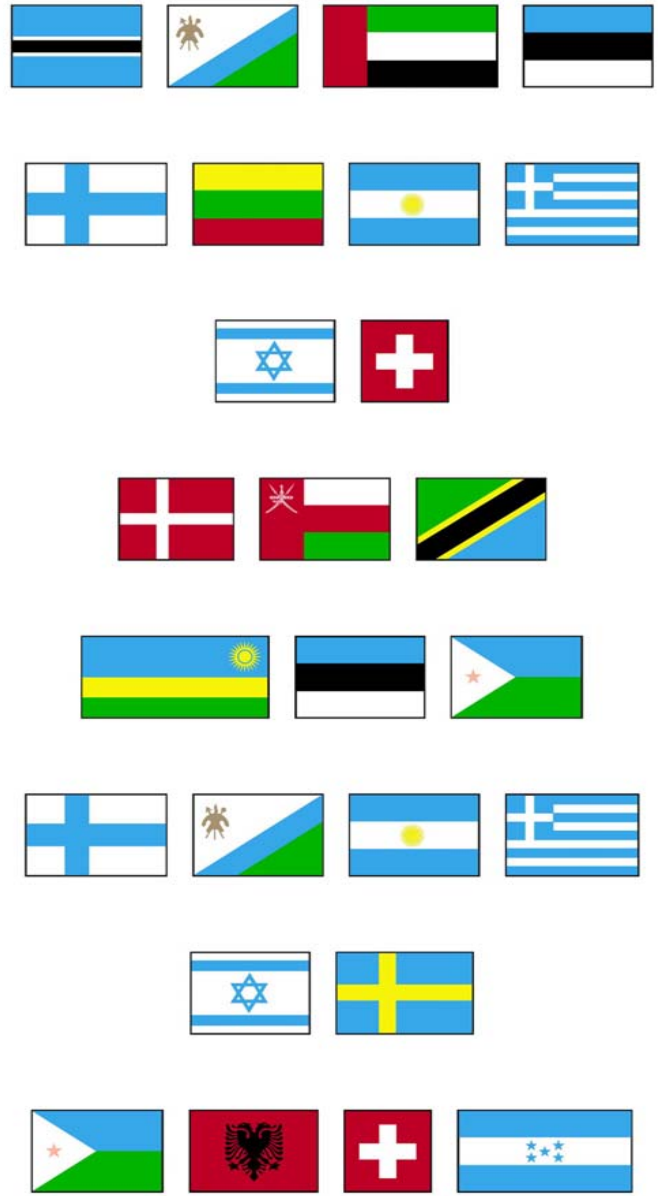
Try *D*(`Q`) = `f` and so on ..

```
o rfr iendf ro  a rise  a ine dhise   t
YIFQF MZRWQ FYVEC FMDZP CVMRZ WNMDZ VEJBT
 ass  iths  r ris easif  ea o ratio nhadt
XCDDU MJNDI FEFMD ZCDMQ ZKCEY FCJMY RNCWJ
a en   a e  hi eh   asn t oo  in i  o red
CSZRE XCHZU NMXZN ZUCDR JXYYS MRTME YIFZW
so e  ore i neand heset   ed  a  in his h
DYVZV YFZUM RZCRW NZDZJ JXZWG CHSMR NMDHN
airfa  eti  ted   to ar dsthe s n
CMFQC HZJMX JZWIE JYUCF WDJNZ DIR
```

```
ourfr iendf ro  a rise  a ine dhise   t
YIFQF MZRWQ FYVEC FMDZP CVMRZ WNMDZ VEJBT
 ass  ithsu r ris easif ea  o ratio nhadt
XCDDU MJNDI FEFMD ZCDMQ ZKCEY FCJMY RNCWJ
a en   a e  hi eh e asn t oo  in i  oured
CSZRE XCHZU NMXZN ZUCDR JXYYS MRTME YIFZW
so e  ore i neand heset t ed  a  in his h
DYVZV YFZUM RZCRW NZDZJ JXZWG CHSMR NMDHN
airfa  eti  tedu  to ar dsthe sun
CMFQC HZJMX JZWIE JYUCF WDJNZ DIR
```

```
ourfr iendf rom a rise  amine dhise m t
YIFQF MZRWQ FYVEC FMDZP CVMRZ WNMDZ VEJBT
 ass  ithsu r ris easif ea  o ratio nhadt
XCDDU MJNDI FEFMD ZCDMQ ZKCEY FCJMY RNCWJ
a en   a e  hi eh e asn t oo  in i  oured
CSZRE XCHZU NMXZN ZUCDR JXYYS MRTME YIFZW
somem ore i neand heset t ed  a  in his h
DYVZV YFZUM RZCRW NZDZJ JXZWG CHSMR NMDHN
airfa  eti  tedu  to ar dsthe sun
CMFQC HZJMX JZWIE JYUCF WDJNZ DIR
```

ourfr iendf rompa risex amine dhise mptyg
YIFQF MZRWQ FYVEC FMDZP CVMRZ WNMDZ VEJBT
lassw ithsu rpris easif evapo ratio nhadt
XCDDU MJNDI FEFMD ZCDMQ ZKCEY FCJMY RNCWJ
akenp lacew hileh ewasn tlook ingip oured
CSZRE XCHZU NMXZN ZUCDR JXYYS MRTME YIFZW
somem orewi neand heset tledb ackin hisch
DYVZV YFZUM RZCRW NZDZJ JXZWG CHSMR NMDHN
airfa cetil tedup towar dsthe sun
CMFQC HZJMX JZWIE JYUCF WDJNZ DIR



Playfair Cipher

Key: MONARCHY

is put first in a 5x5 matrix, which is then filled out with the remaining letters of the alphabet (i = j)

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

The encryption rules

Plaintext formatting

`oo -> oxo`

Same row or column

`ar -> RM`

`mu -> CM`

Regular case

`hs -> BP`

`ea -> IM`

Hill Cipher

$$\begin{pmatrix} c_1 \\ c_2 \\ c_3 \end{pmatrix} = \begin{pmatrix} k_{11} & k_{12} & k_{13} \\ k_{21} & k_{22} & k_{23} \\ k_{31} & k_{32} & k_{33} \end{pmatrix} \times \begin{pmatrix} p_1 \\ p_2 \\ p_3 \end{pmatrix} \mod 26$$

Plaintext: triples of numbers in {0,1,2,…,25}

Ciphertext: triples of numbers in {0,1,2,…,25}

Key: 3x3 matrices with entries in {0,1,2,…,25}

Arithmetic as in shift cipher plus multiplication mod 26

Polyalphabetic ciphers: Vigenère

Plaintext and Ciphertext:

finite sequences of characters in {0,1,2,…,25}

Key has period q : $k_1\, k_2\, k_3 \ldots k_{q-1}\, k_q$

sequence of length q of characters from {0,1,2,…,25}

Encryption:

$c_1 = (p_1 + k_1) \bmod 26 \qquad c_{q+1} = (p_{q+1} + k_1) \bmod 26$

$c_2 = (p_2 + k_2) \bmod 26 \qquad c_{q+2} = (p_{q+2} + k_2) \bmod 26$

... ...

$c_q = (p_q + k_q) \bmod 26 \qquad c_{2q} = (p_{2q} + k_q) \bmod 26$

and so on..

Polyalphabetic ciphers: Vigenère

Example

Key: `spring`

```
ourfr iendf rompa risex amine dhise mptyg
sprin gspri ngspr ingsp rings pring sprin
GJINE OWCUN EU... ..
```

Note the repetition of a two character string resulting from a repetition in the plaintext!

Kasiski's method to determine the period

- Many strings of characters repeat themselves in natural languages.
- Assume the interval between occurence of a string is a multiple of the length of the period.
- Then a repetition of a character string of the same length occurs in the ciphertext.
- By detecting repetitions of strings in the ciphertext one can find the period as the greatest common divisor (GCD) of the repetition intervals
- Their may be false repetitions. The longer the repeating string the more significant it is. Sometimes only strings of length ≥ 3 are considered.

One Time Pad

- Claude Shannon laid (1949) the information theoretic fundamentals of secrecy systems.
- Shannon's pessimistic bound: For perfect secrecy, the length of the key is at least as large as the length of the plaintext.
- An example of a cipher which achieves perfect secrecy is the One Time Pad

$$c_i = (p_i + k_i) \bmod 26$$

where a new fresh random key $k_1 k_2 k_3 \dots k_i \dots$ is chosen for each new plaintext $p_1 p_2 p_3 \dots p_i \dots$

- Practical ciphers do not provide perfect secrecy

2.2 Introduction to contemporary cryptographic primitives

- Secret key (symmetric) primitives
  - Block cipher
  - Stream cipher
  - Integrity primitives
    - Message authentication code
    - Hash functions
- Public key (asymmetric) primitives
  - Public key encryption scheme
  - Digital signature scheme

Primitives and protocols

- Cryptographic primitives and functions are used as building blocks for cryptographic protocols
- For example,
  - A stream cipher primitive is the basic building block for an encryption protocol
  - A message authentication code is the basic building block for an authentication protocol

Different design approaches

- information theoretic
  - security is measured in terms of probabilities
- complexity theoretic
  - security measured in terms of computational and memory requirements
- quantum cryptology
- system based
  - security measured in terms of used cryptanalysis methods

Different assumptions:

- capabilities of an opponent
- cryptanalytic success
- definition of security (e.g., unconditional security, computational security)

Man-made vs. Math-made

Symmetric primitives

- are based on man-made constructions;
- are fast and easy to implement in software and/or hardware:
- use short keys

Asymmetric (public key primitives)

- are based on mathematical construction and their security is derived from infeasibility of some computationally hard problem.
- are slow and difficult to implement (both in software and hardware)
- have long keys and parameters

It would be possible to construct symmetric primitives based on mathematics, but they are not used in practise because they are not efficient compared to symmetric primitives

Constraints

Why not all algorithms are secure?

- public – proprietary
- weak – strong
- crypto competence
- export control
- economic reasons
- degradation over time (Moore’s Law, quantum threat)

Life Cycle of a Cryptographic Algorithm

DEVELOPMENT

- Construction
- Security proofs and arguments
- Evaluation
- Publication of the algorithm
- Independent evaluation

USE

- Implementation
- Embedding into system
- Key management
- Independent evaluation

END

- Break; or
- Degradation by time
- Implementation attacks
- Side channel attacks

Block ciphers

Sender: (Message , Secret key) $\longrightarrow$ Ciphertext

Receiver: (Ciphertext, Secret key) $\longrightarrow$ Message

Confidentiality primitive

- Threat: retrieve the plaintext from the ciphertext without the knowledge of the key.
- Security goal: protect against this threat.

Plaintext *P*: strings of bits of fixed length *n*

Ciphertext *C*: strings of bits of the same length *n*

Key *K*: string of bits of fixed length *k*

Encryption transformations: For each fixed key the encryption operation E_K is one-to-one (invertible) function from the set of plaintexts to the set of ciphertext. That is, there exist an inverse transformation, decryption transformation D_K such that for each *P* and *K* we have: $D_K(E_K(P)) = P$

Block ciphers, security

- Security is measured in terms of time: How long it takes to break the cipher using available resources.
- Upperbound of security: The time complexity of exhaustive key search, which is equal to 2^k, with key length of k bits.
- A second upperbound: $2^{n/2}$, with block length n (due to Birthday paradox, see next page): If we see two equal ciphertexts, then we know that the plaintexts are equal.
- If an attack leads to a break, in time 2^t, where $t < k$, then the cipher is said to be *theoretically broken,* and that the *effective key length* of the cipher is reduced to t. (This does not mean that the cipher is broken in practise unless t is very small.)

Birthday paradox

- No paradox, just somewhat counterintuitive
- Assume that numbers are randomly (with equal probability) picked with replacement from a set of N different numbers.
- Question: How many numbers must be picked until the probability of getting at least one number twice is at least 0.5 ?
- Answer: Approximately $1.17\sqrt{N}$
- See Stallings Appendix 11A, p. 340.

Birthday paradox: Derivation

Let k be the number drawn from the set of N elements with replacement. Then

P = Pr[at least one match] = 1 – Pr[no match] =

$$1-1\cdot\left(1-\frac{1}{N}\right)\cdot\left(1-\frac{2}{N}\right)\cdots\left(1-\frac{k-1}{N}\right)=1-\prod_{i=1}^{k-1}\left(1-\frac{i}{N}\right)$$

Since N is large, we can approximate, for all $i = 1,\ldots, k$-1

$\left(1-\frac{i}{N}\right)\approx e^{-\frac{i}{N}}$ and get:

$$P\approx 1-\prod_{i=1}^{k-1}e^{-\frac{i}{N}}=1-e^{-\frac{1}{N}\sum_{i=1}^{k}i}=1-e^{-\frac{k(k-1)}{2N}}\approx 1-e^{-\frac{k^2}{2N}}$$

Then $P\approx\frac{1}{2}=e^{-\ln 2}$ if $k^2\approx 2N\ln 2$, that is, $k\approx\sqrt{2N\ln 2}\approx 1.17\sqrt{N}$ as desired.

Attack on block ciphers

Let n be the block length in bits. Then if the block cipher encryption operation is used about $2^{n/2}$ times with the same key on any randomly generated data as plaintext, then by Birthday Paradox, the probability of having two equal ciphertexts is about ½. Then one knows that the two corresponding input data are equal.

Block ciphers, design principles

- The ultimate design goal of a block cipher is to use the secret key as efficiently as possible.
- Confusion and diffusion (Shannon)
- New design criteria are being discovered as response to new attacks.
- A state-of-the-art block cipher is constructed taking into account all known attacks and design principles.
- But no such block cipher can become provably secure, it may remain open to some new, unforeseen attacks.
- Common constructions with iterated round function
  - Substitution permutation network (SPN)
  - Feistel network

Attack on block ciphers

- *Ciphertext only attack:* The attacker has access to some amount of ciphertext and also knows something about the nature of the plaintext, which is not perfectly random.
- *Known plaintext attack:* The attacker has access to some amount of plaintext and the corresponding ciphertext.
- *Chosen plaintext attack:* The attacker is able to choose some amount of plaintext and obtains the corresponding ciphertext.
- *Adaptively chosen plaintext attack.* The attacker is able to choose some amount of plaintext in parts, and obtain the corresponding ciphertext, where the choice of each new part of plaintext is influenced by all previously obtained ciphertext.
- *Chosen ciphertext and adaptive chosen ciphertext attacks:* Similar to the chosen plaintext attacks but now with the roles of plaintext and ciphertext reversed.

Stream ciphers

- Stream ciphers are generally faster than block ciphers, especially when implemented in hardware.
- Stream ciphers have less hardware complexity.
- Stream ciphers can be adapted to process the plaintext bit by bit, or word by word, while block ciphers require buffering to accumulate the full plaintext block.
- Synchronous stream ciphers have no error propagation; encryption is done character by character with keys K_i that are independent of the data

$$C_i = E_{Ki}(P_i)$$

- Function E is simple, the function which computes the key sequence is complex
- Example: Vigenère cipher, One Time Pad

$$C_i = (P_i + K_i)\mathrm{mod}\ 26$$

Stream cipher encryption

Sender:

Secret key $\longrightarrow$ Key stream

(Key stream , Message) $\longrightarrow$ Ciphertext

Receiver:

Secret key $\longrightarrow$ Key stream

(Ciphertext, Key stream) $\longrightarrow$ Message

Stream ciphers: Security

- Known plaintext gives known key stream. Chosen plaintext gives the same but nothing more.
- Chosen ciphertext attack may be a useful method for analysing a self-synchronising stream cipher.
- The attacker of a stream cipher may try to find one internal state of the stream cipher to obtain a functionally equivalent algorithm without knowing the key.
- Distinguishing a keystream sequence from a truly random sequence allows also the keystream to be predicted with some accuracy. Such attack is also called prediction attack.

Requirements:

- Long period
- A fixed initialisation value the stream cipher generates a different keystream for each key.

Stream ciphers: Designs

Linear feedback shift register (LFSR). LFSRs are often used as the running engine for a stream cipher.

Stream cipher design based on LFSRs uses a number of different LFSRs and nonlinear Boolean functions coupled in different ways. Three common LFSR-based types of stream cipher can be identified:

- *Nonlinear combination generators*: The keystream is generated as a nonlinear function of the outputs of multiple LFSRs
- *Nonlinear filter generators*: The keystream is generated as a nonlinear function of stages of a single LFSR.
- *Clock controlled generators*: In these constructions, the necessary nonlinearity is created by irregular clocking of the LFSRs. The GSM encryption algorithm A5/1 is an example of a stream cipher of this type.

Message authentication codes (MAC)

Sender: (Secret key , Message) $\longrightarrow$ MAC

Receiver: (Secret key , Message) $\longrightarrow$ MAC

- A MAC of a message P of arbitrary length is computed as a function $H_K(P)$ of P under the control of a secret key K.
- The MAC length m is fixed.
- Security requirement: it must be infeasible, without the knowledge of the secret key, to determine the correct value of $H_K(P)$ with a success probability larger than $1/2^m$. This is the probability of simply guessing the MAC value correctly at random. It should not be possible to increase this probability even if a large number of correct pairs P and $H_K(P)$ is available to the attacker.

Message authentication codes (MAC)

- Similarly as block ciphers, MAC algorithms operate on relatively large blocks of data. Most MACs are iterated constructions. The core function in the MAC algorithm is a compression function. At each round the compression function takes a new data block and compresses it together with the compression result from the previous rounds. Hence the length of the message to be authenticated determines how many iteration rounds are required to compute the MAC value.

- Given a message X and its MAC value H, it can be verified by anybody in possession of the secret key K and the MAC computation algorithm.

Hash functions

Sender: Message $\longrightarrow$ Hash code

Receiver: Message $\longrightarrow$ Hash code

- A hash code of a message P of arbitrary length is computed as a function $H(P)$ of P . The hash length m is fixed.
- Security requirements:
  1. *Preimage resistance:* Given h it is impossible to find P such that $H(P) = h$
  2. *Second preimage resistance:* Given P it is impossible to find P' such that $H(P') = H(P)$
  3. *Collision resistance: I*t is impossible to find P and P' such that $P \neq P'$ and $H(P') = H(P)$

Hash functions

- Similarly as MAC algorithms, hash functions typically operate on relatively large blocks of data. Most hash functions are iterated constructions. The core function in a hash function is a compression function. At each round the compression function takes a new data block and compresses it together with the compression result from the previous rounds. Hence the length of the message to be authenticated determines how many iteration rounds are required to compute the MAC value.

- Hash function is public: Given a message P anybody can compute the hash code of P.

Public key encryption

Sender: (Message , Public key) $\longrightarrow$ Ciphertext

Receiver: (Ciphertext , Private key) $\longrightarrow$ Message

- Slow, usually used to encrypt short messages in more complex protocols than just bulk message encryption: data authentication, key agreement etc.
- Because of the mathematical structures involved, complex message formatting rules (with hash functions) are required.
- Chosen ciphertext attacks maybe an essentially more serious threat than chosen plaintext (for symmetric block ciphers they are about the same). We will see an example later.
- RSA, ElGamal in different groups, Pairing based techniques …

Digital signatures

Sender: (Message , Private key) → Signature

Receiver: (Signature , Public key) → Validity (1 bit)

- Important primitive; the only one to provide non-repudiation.
- Slow, message are signed by applying the digital signature operation on a fixed length hash of the message.
- Used for
  - message authentication protocols
  - non-repudiation protocols
  - authentication and key agreement
  - commitment schemes
  - …
- RSA, ElGamal in different groups, Schnorr, DSA, Pairing based techniques