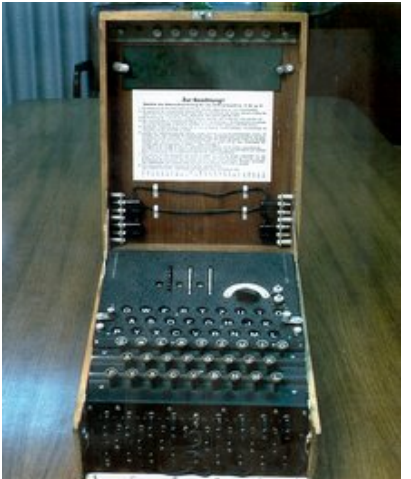
4 PKI – Public Key Infrastructure

4.1 PKI history

- Classical cryptography
  - dates back thousands of years
  - symmetric key

Example form II WW: Enigma



Symmetric key cryptography has a good property: possible to devise **secure** but at the same time – **fast** algorithms**!**

- Public key cryptography:
  - non-symmetric key
  - Clifford Cocks ca 1973 secretly (until 1997, *(The Government Communications Headquarters (GCHQ) – British intelligence agency responsible for providing signals intelligence (SIGINT) and information assurance)*);
  - in parallel **R**ivest, **S**hamir and **A**dleman 1976 (their picture) (`http://people.csail.mit.edu/rivest/photos/Len-Adi-Ron.jpg`)
    * **RSA algoritm** or
    * **Cocks/RSA**)

One-way functions

Definition. Function $f : X \to Y$ is a **one-way function** if it is easy to calculate $y=f(x)$, but knowing only y it is computationally impossible to find efficiently x such that $f(x)=y$. Function f has a backdoor $\boldsymbol{C}$, if knowing the information $\boldsymbol{C}$ it is possible to invert the function f efficiently.

- Existence of one-way functions not proven mathematically
  - but suitable candidates exist
- one of the best-known candidates: calculation of number x^b modulo n (1977)
- It is easy to find solution to the equation $x^b \equiv y \bmod n$ explicitly trying all the values of x but it is computationally inefficient if n is large

4.2 RSA

Definition. For a given positive integer n, two integers a and b are called congruent modulo n, written

$a \equiv b \bmod n$

if $a - b$ is divisible by n (or equivalently if a and b have the same remainder when divided by n)

Example.

$7 \equiv 2 \mod 5$

RSA algoritm main idea based on an example:

$2^1 \equiv 2 \mathrm{mod} 19$

$2^2 \equiv 4 \mathrm{mod} 19$

$2^3 \equiv 8 \mathrm{mod} 19$

$2^4 \equiv 16 \mathrm{mod} 19$

$2^5 \equiv 13 \mathrm{mod} 19$

$2^6 \equiv 7 \mathrm{mod} 19$

$2^7 \equiv 14 \mathrm{mod} 19$

$2^8 \equiv 9 \mathrm{mod} 19$

$2^9 \equiv 18 \mathrm{mod} 19$

$2^{10} \equiv 17 \mathrm{mod} 19$

$2^{11} \equiv 15 \mathrm{mod} 19$

$2^{12} \equiv 11 \mathrm{mod} 19$

$2^{13} \equiv 3 \mathrm{mod} 19$

$2^{14} \equiv 6 \mathrm{mod} 19$

$2^{15} \equiv 12 \mathrm{mod} 19$

$2^{16} \equiv 5 \mathrm{mod} 19$

$2^{17} \equiv 10 \mathrm{mod} 19$

$2^{18} \equiv 1 \mathrm{mod} 19$

RSA is a Block-algorithm: both the original text M as well as the encrypted text C consist of blocks of length 0 to $n-1$.

1. Choose 2 large random distinct prime numbers $p \neq q$. Calculate $n = pq$

2. Calculate $\varphi = (p-1)(q-1)$ (Euler's totient function)

3. Choose integer e ($1 < e < \varphi$), which is coprime to φ, i.e. $gcd(e, \varphi) = 1$

(Definition. Integer and its coprime have their greatest common divisor 1. e.g., 6 and 35 are coprimes, 6 and 27 are not coprimes (divisible by 3)

(How to determine GCD? Euclidean algorithm (**Python** code:)

```
def gcd(a, b):
  if b == 0: return a
  else: return gcd(b,a%b) )
```

4. Find d, for which $d\,e \equiv 1 \bmod \varphi$. (i.e. $(d\,e-1) \bmod \varphi = 0$);

Here: n and e – public key , n and d – private key

Encrypting: M transformed into the ciphertext: $C = M^e \bmod n$

Decrypting: $M = C^d \bmod n$.

4.3 RSA help algorithms

Example: Euclidean algorithm and its extension

Euclidean algorithm:

$2107=2*896+315$

$896=2*315+266$

$315=1*266+49$

$266=5*49+21$

$49=2*21+7$

=>GCD(2107,896)=7

Perform back-substitution:

$7=49-2*21$

$=49-2*(266-5*49)$

$=-2*266+11*49$

$=-2*266+11*(315-266)$

$=11*315-13*266$

$=11*315-13*(896-2*315)$

$=-13*896+37*315$

$=-13*896+37*(2107-2*986)$

$=(37)*2107+(-87)*896$

- Therefore, for implementing RSA algorithm steps ***3.*** and ***4.*** extended Euclidean algorithm can be used:

```
def exeu(a, b): # (Extended Euclidean algorithm)
  q=0L; r=0L;
  x = [0L,0L,0L]
  y = [0L,0L,0L]
  if not b: return [1,0]
  else:
   x[2] = 1; x[1] = 0
   y[2] = 0; y[1] = 1
   while (b>0):
     q=a/b
     r=a-q*b
     x[0]=x[2]-q*x[1];
     y[0]=y[2]-q*y[1]
     a,b=b,r
     x[2]=x[1];x[1]=x[0];
     y[2]=y[1];y[1]=y[0];
   return [x[2],y[2]]
```

Extended Euclidean algorithm gives: $\mathrm{GCD}(a,b) = x*a + y*b$

Example 1: Let $p = 3$ and $q = 11$ => $n = 3*11 = 33$

$\varphi(n) = (3-1)*(11-1) = 20$

Choose randomly e such that GCD(e,20)=1. Let $e = 7$, GCD(7,20)=1

Using the function above: `exeu(20,7)` gives: `[-1,3]`, i.e.

1=-1*20+3*7

=> $d = 3$

(We can say that $d\,e = 1 \mathrm{mod} \varphi$. (i.e. $(d\,e-1)\mathrm{mod}\varphi = 0$)

Public key is: $(n = 33,\, e = 7)$

Secret key: $(n = 33,\, d = 3)$

Let $M = 2$. Then $C = 2^7 \mathrm{mod} 33 = 29$

and $M = 29^3 \mathrm{mod} 33 = 2$.

Example 2: Let $p = 47$ and $q = 71$ => $n = p * q = 3337$

$\varphi(n) = (p-1)*(q-1) = 3220$

Choose randomly e such that GCD(e,3220)=1. Let $e = 79$, GCD(79,3220)=1

Using the extended Euclidean algorithm `exeu(3220,79)`

gives: `[-25,1019]`,

i.e. 1=-25*3220+1019*79

=> $d = 1019$

(We can say that $d\,e = 1 \bmod \varphi$. (i.e. $(d\,e-1) \bmod \varphi = 0$)

Public key is: ($n = 3337$,$e = 79$)

Secret key is: ($n = 3337$,$d = 1019$)

Let $M = 582$. Then $C = 582^{79} \bmod 3337 = 776$

and $M = 776^{1019} \bmod 3337 = 582$

Helping tools for implementing RSA algorithm

Fermat's primality test – fast, gives right answer for most of composite numbers with proability greater than 1/2

Miller-Rabin's test – slower but much more accurate identifying some extremely bad composite numbers – Carmichael's numbers

Fermat's prime number test

Fermat's Little Theorem: If p is a prime number then for any integer a $a^p - a$ will be evenly divisible by p.

i.e.

if p is a prime and a is an integer coprime to p, then $a^{p-1} - 1$ will be evenly divisible by p:

- $a^{p-1} \equiv 1 (\bmod p)$

For testing: choose random integer $a < p$ and perform above check

Let n be composite number. There exist such values of a for which $a^{n-1} \equiv 1 (mod\, n)$ – called Fermat's liars (in this case n is called Fermat pseudoprime to base a), rest are Fermat's witnesses to number n

Carmichael numbers – composite numbers n, for which Fermat's liars are all numbers a for which GCD(n, a)=1.

Carmichael numbers:

	2821	29341	63973	162401	294409	410041
561	6601	41041	75361	172081	314821	449065
1105	8911	46657	101101	188461	334153	488881
1729	10585	52633	115921	252601	340561	512461
2465	15841	62745	126217	278545	399001	....

Carmichael number multipliers:

561 : [3, 11, 17]
1105 : [5, 13, 17]
1729 : [7, 13, 19]
2465 : [5, 17, 29]
2821 : [7, 13, 31]
6601 : [7, 23, 41]
8911 : [7, 19, 67]
10585 : [5, 29, 73]
15841 : [7, 31, 73]
29341 : [13, 37, 61]
41041 : [7, 11, 13, 41]
46657 : [13, 37, 97]
52633 : [7, 73, 103]
62745 : [3, 5, 47, 89]
63973 : [7, 13, 19, 37]
75361 : [11, 13, 17, 31]
101101 : [7, 11, 13, 101]
115921 : [13, 37, 241]
126217 : [7, 13, 19, 73]
162401 : [17, 41, 233]
172081 : [7, 13, 31, 61]
188461 : [7, 13, 19, 109]
252601 : [41, 61, 101]
278545 : [5, 17, 29, 113]
294409 : [37, 73, 109]
314821 : [13, 61, 397]
334153 : [19, 43, 409]
340561 : [13, 17, 23, 67]
399001 : [31, 61, 211]
410041 : [41, 73, 137]
449065 : [5, 19, 29, 163]
488881 : [37, 73, 181]
512461 : [31, 61, 271]

Miller-Rabin's test

1. Choose odd n => $n-1=2^s \times d$, where s -- integer, d -- odd.

2. Choose random integer $a < n$.

3. If $a^d \bmod n = 1$: return 'probably non-prime!'

4. for j in $[0:s-1]$:
   if ($a^{2^j d} \bmod n = -1$: return 'probably prime!'
5. return 'non-prime!'

If the test returns "probably prime" for t tests, then probability that n is prime is $1-4^{-t}$. Example, for $t=10$ we have n as prime with probability of 0.99999 . Usually, the choice for $t \approx 40$.

4.4 Other algorithms

Other public key algorithms

- ElGamal
- DSS
- Diffie-Hellmann
- LUC
- XTR

4.5 Public Key Publishing

Distributing public keys

- How to make authentic copy of a public key to all potential checkers of the signature?
- Draw-back:
  - Modification of a key can be used for man-in-the-middle attacks

Idea: Public keys in a phonebook

- Idea by Diffie and Hellman, in 1976
- “Phonebook” available in the Internet as on-line service

Certificate idea

- 1978 – Kohnfelder:
  - global online service for public key distribution not feasible due to high need for communication
  - idea of **certificates**: each phone-book entry (ceritficate) separately signed by trusted service provider

Certificate: document connecting a person with his/her public key

- *Owner's ID*
- *Owner's public key*
- *Certificate issuer*
- *Validity period*

- etc.

Benefits with certificates:

- can be distributed independently
- no need for “phonebook”-checks each time
- => reduced communication needs; communication channels load more even: no bottle-neck near the “phonebook”

Terms and abbreviations associated with certificates

Certification Authority (**CA**)

Certificate policy (**CP**)

Certification path

Certification Practice Statement (**CPS**)

Certificate revocation list (**CRL**)

Issuing certification authority (***issuing*** **CA**)

Public Key Certificate (**PKC**) – data structure containing end user [/device/computer] public key together with some other useful information digitally signed with the secret key of the issuing **CA**

Registration authority (**RA**) – enterprise performing certificate identification, authorisation, but does not issue or sign certificates. **RA** fullfills tasks delegated by **CA**.

How are certificates issued?

- Certificates are issued by trusted party – **CA** (Certification Authority)
- CA has **CP** (Certification Policy, documented set of rules) regulating whom to, how and for what purposes certificates are issued
- CA has to reliably protect its secret key
- CA has to perform regular audits for its actual security situation

Using a certificate:

It is enough if all know the CA public key!

Merkle's tree structure is used

Merkle's tree

(also, *the Tree Hash Exchange (THEX) format*)

```
                ROOT=IC(E+F)
                 /          \
                /            \
        E=IC(A+B)        F=IC(C+D)
         /    \            /     \
        /      \          /       \
A=EC(S1) B=EC(S2) C=EC(S3) D=EC(S4)
```

EC - end certificate

IC - interim certificate

Storing the certificates

Public key is made available to everybody

Secret keys are kept in a CA typically on a separate storage device:

- online (ID-card (or similar), usually special FIPS 3 cryptodevice)
- in an isolated storage (e.g. USB flash memory kept away in a safe)

Usually CA activities on a separate machine isolated from the network

Eestonian Grid policy:

- key protected with over 15 character password
- CA is backed up (CA database as well as public-secret key)
- All activities are getting logged
- CRL is published at webpage Baltic Grid webpage (`http://ca.balticgrid.org`) (Estonian Grid webpage (`http://grid.eenet.ee/CA`))

Additional information on CA and its activities:

- EuroPKI (`http://www.europki.org`)