Crittografia classica

- Trasmissione di segreti.

Protocolli a due, codici simmetrici.

Crittografia moderna

- Autenticazione
  - Verifiche di identità
    * Frima digitale
    * Controllo login
  - Verifiche di integrità
    * Integrità file
    * Integrità messaggi

Protocolli pubblici, codici asimmetrici.

Trasmissione segreta

chiaro	Codifica	cifrato	Decodifica	chiaro
m	$\rightarrow\boxed{\mathcal{C}}\rightarrow$	x	$\rightarrow\boxed{\mathcal{D}}\rightarrow$	m

Protocollo simmetrico

$$\text{m} \quad \rightarrow\boxed{\mathcal{C}^s_K}\rightarrow \quad x \quad \rightarrow\boxed{\mathcal{D}^s_K}\rightarrow \quad m$$

Si usa una sola chiave, nota sia a chi deve cifrare sia a chi deve decifrare

Protocollo asimmetrico

$$m \quad \rightarrow\boxed{\mathcal{C}_{K_c}}\rightarrow \quad x \quad \rightarrow\boxed{\mathcal{D}_{K_d}}\rightarrow \quad m$$

Le due chiavi sono distinte, una può essere resa pubblica mentre la seconda rimane segreta

Uso misto

I protocolli simmetrici sono piú efficienti; i due sistemi vengono usati in coppia. Il protocollo asimmetrico cifra una chiave *effimera*.

$$
\begin{array}{ccccccccccc}
 & & K_s & \to & \boxed{\mathcal{C}_{K_c}} & \to & k & & & & \\
 & & \downarrow & & & & \downarrow & & & & \\
m & \to & \boxed{\mathcal{C}^s_{K_s}} & \to & x & \to & (k,x) & \to & x & \to & \boxed{\mathcal{D}^s_{K_s}} \to m \\
 & & & & & & \downarrow & & & & \uparrow \\
 & & & & & & k & \to & \boxed{\mathcal{D}_{K_d}} & \to & K_s
\end{array}
$$

Firma digitale

$$
\begin{array}{ccccccccc}
\text{testo} & & \text{Firma} & & \text{firmato} & & \text{Verifica} & & \\
m & \to & \boxed{\mathcal{F}_{K_f}} & \to & (m,f) & \to & \boxed{\mathcal{V}_{K_v}} & \to & \text{Sí/No}
\end{array}
$$

Due primi p, q; usiamo il prodotto $n = pq$.
Messaggi in $M = \mathbb{Z}/_{n\mathbb{Z}}$.
Due esponenti $cd \equiv 1 \pmod{\phi(n)}$.

$[\text{PUB}]\quad K = (n, c)$

$[\text{ PRI }]\quad K = (n, d)$

Cifrati in $X = \mathbb{Z}/n\mathbb{Z}$.

Trasmissione segreta

c

d , *fattorizzazione* $n = pq$

[CIF] $\mathcal{C}_{K_c}(m) \to x \equiv m^c \pmod{n}$

[DEC] $\mathcal{D}_{K_d}(x) \to x^d \equiv m^{cd} \equiv m \pmod{n}$

$$M \ni m \quad \to \boxed{\mathcal{C}_{K_c}} \to \quad m^c \quad \to \boxed{\mathcal{D}_{K_d}} \to \quad m^{cd}$$

Il protocollo RSA è il piú semplice da descrivere e il piú usato dei protocolli asimmetrici, ma **non** il piú sicuro.

Due primi p, q; usiamo il prodotto $n = pq$.
Messaggi in $M = \mathbb{Z}/_{n\mathbb{Z}}$.
Due esponenti $cd \equiv 1 \pmod{\phi(n)}$.

$$[\text{PUB}] \quad K = (n, c)$$

$$[\text{PRI}] \quad K = (n, d)$$

Firme in $F = \mathbb{Z}/n\mathbb{Z}$.

Firma digitale

$$_v \qquad = K_c$$

$$_f \qquad = K_d$$

[FIR] $\mathcal{F}_{K_f}(m) \to f \equiv m^d \pmod{n}$

[VER] $\mathcal{V}_{K_v}(m, f)$ verifica se $m \equiv f^c \pmod{n}$

In realtà non è necessario che venga trasmesso anche il messaggio, può essere ricavato dalla firma f.

$$m \quad \to \boxed{\mathcal{F}_{K_f}} \to \quad m^d \quad \to \boxed{\mathcal{V}_{K_v}} \to \quad m^{dc} \equiv m$$

Dimostrazione funzionamento

Dal protocollo sappiamo che

- $n = pq$
- $cd \equiv 1 \pmod{\phi(n)}$

Per la definizione della ϕ di Eulero

- $\phi(n) = \phi(p)\phi(q) = (p-1)(q-1)$

Per il piccolo teorema di Fermat:

- $\forall m \in \mathbb{N},\ m^p \equiv m^{\phi(p)+1} \equiv m \pmod{p}$
- $\forall m \in \mathbb{N},\ m^q \equiv m^{\phi(q)+1} \equiv m \pmod{q}$

Per il teorema cinese dei resti

- $m^{cd} \equiv m^{\phi(p)\phi(q)+1} \equiv m \pmod{pq}$

In realtà è sufficiente $\lambda(n) = (\phi(p), \phi(q))$

Un solo primo p. Un generatore $\langle g \rangle = {\mathbb{Z}/p\mathbb{Z}}^*$.
Un elemento $\alpha \equiv g^a \pmod{p}$.
Messaggi in $M = {\mathbb{Z}/p\mathbb{Z}}^*$.

[PUB] $K = (p, \alpha, g)$

[PRI] $K = (a)$

Cifrati in $X = {\mathbb{Z}/p\mathbb{Z}}^2$.

Trasmissione segreta

c

d

[CIF] $b \in {\mathbb{Z}/p\mathbb{Z}}^*$, $\mathcal{C}_{K_c}(m) \to x = (g^b, m\alpha^b)$

[DEC] $\mathcal{D}_{K_d}(x) \to (m\alpha^b)(g^b)^{-a} \equiv m \pmod p$

Il protocollo ElGamal, a pari condizioni è piú sicuro dell'RSA. Se al posto di p si usa $n = pq$ la maggior sicurezza è dimostrabile.

(p)

(p)

Un solo primo p. Un generatore $\langle g \rangle = {\mathbb{Z}/p\mathbb{Z}}^*$.
Un elemento $\alpha \equiv g^a \pmod{p}$.
Messaggi in $M = {\mathbb{Z}/p\mathbb{Z}}^*$.

$[\,\mathrm{PUB}\,]\quad K = (p, \alpha, g)$

$[\,\mathrm{PRI}\,]\quad K = (a)$

Firma digitale

$_v$ $= K_c$

$_f$ $= K_d$

[FIR] $r \in \mathbb{Z}_{/(p-1)\mathbb{Z}}{}^*, \quad x \equiv g^b \pmod{p},$
$m \equiv xa + ry \pmod{p-1},$
$\mathcal{F}_{K_f}(m) \rightarrow f = (x, y)$

[VER] $\mathcal{V}_{K_v}$ verifica che $\alpha^x x^y \equiv g^m \pmod{p}$

La verifica funziona perché
$\alpha^x x^y \equiv g^{K_d x} g^{ry} \equiv g^{K_d x + ry} \equiv g^m \pmod{p}$
ed r è invertibile modulo $p - 1$.

Obiettivo

rendere sicura la gestione remota di un account.

- controllare l'identità della macchina
  Per evitare che la password vada al server sbagliato e difendere da *man in the middle*.
- cifrare il resto della connessione
  La password non passa in chiaro, neppure altre password di collegamenti in cascata.
- evitare che la connessione venga *scippata*

- può comprimere la connessione
- può trasferire finestre X

File

```
bodrato@socrate 12:07:36 ~>ls .ssh
authorized_keys  identity         known_hosts
config           identity.pub     random_seed
```

Prima connessione

```
bodrato@socrate 11:39:58 ~>ssh gauss.dm.unipi.it
Host key not found from the list of known hosts.
Are you sure you want to continue connecting? no
```

Attacco o amnesia

```
bodrato@socrate 11:42:04 ~>ssh posso.dm.unipi.it
@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
@      WARNING: HOST IDENTIFICATION HAS CHANGED!       @
@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
IT IS POSSIBLE THAT SOMEONE IS DOING SOMETHING NASTY!
Someone could be eavesdropping on you right now
(man-in-the-middle attack)! It is also possible
that the host key has just been changed.
Please contact your system administrator.
Add correct host key in /home/bodrato/.ssh/known_hosts to
get rid of this message. Agent forwarding is disabled to
avoid attacks by corrupted servers. X11 forwarding is
disabled to avoid attacks by corrupted servers.
Are you sure you want to continue connecting (yes/no)? no
```

SSH: autenticazione RSA


```
@socrate>ssh-keygen
Initializing random number generator...
Generating p:  .........++ (distance 260)
Generating q:  ...++ (distance 64)
Computing the keys...
Testing the keys...
Key generation complete.
Enter file in which to save the key
  (/home/bodrato/.ssh/identity):
Enter passphrase:
Enter the same passphrase again:
Your identification has been saved in
 /home/bodrato/.ssh/identity.
Your public key is:
1024 33 152589...644337 bodrato@socrate
Your public key has been saved in
 /home/bodrato/.ssh/identity.pub
@socrate>scp .ssh/identity.pub posso:
bodrato@posso.dm.unipi.it's password:

@posso>cat identity.pub >>.ssh/authorized_keys
@posso>rm identity.pub

@socrate>ssh posso
Last login: Mon Mar 13 11:15:43 2000 from linuz
@posso>
```

Posta remota

sotto KDE

/.kde/share/applnk/apps/postaremota.kdelnk

```
SwallowExec=ssh -n cibs xbiff -name cibsbiff
SwallowTitle=cibsbiff
Exec=ssh -C -t -x cibs pine
Terminal=1
```

senza KDE

```
ssh -f -n cibs xbiff
xterm -e ssh -C -t -x cibs pine
```

- Nessuna dimostrazione di sicurezza per protocolli asimmetrici
- La segretezza dell'algoritmo **non** aumenta la sicurezza
- Buoni protocolli in un programma scritto male non danno sicurezza
- Un buon programma usato male non dà sicurezza