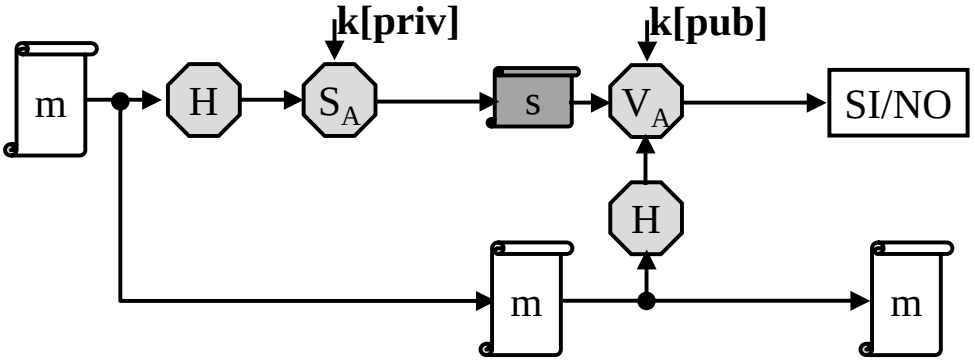
# Meccanismi con sicurezza basata sul problema del logaritmo discreto

- Scambio di Diffie-Hellman
- Cifrario di ElGamal
- **Firma di ElGamal**
- **Digital Signature Standard**

# Algoritmi di firma con appendice

**Messaggi di lunghezza arbitraria**
**Appendice: S(H(m),k[privA])**

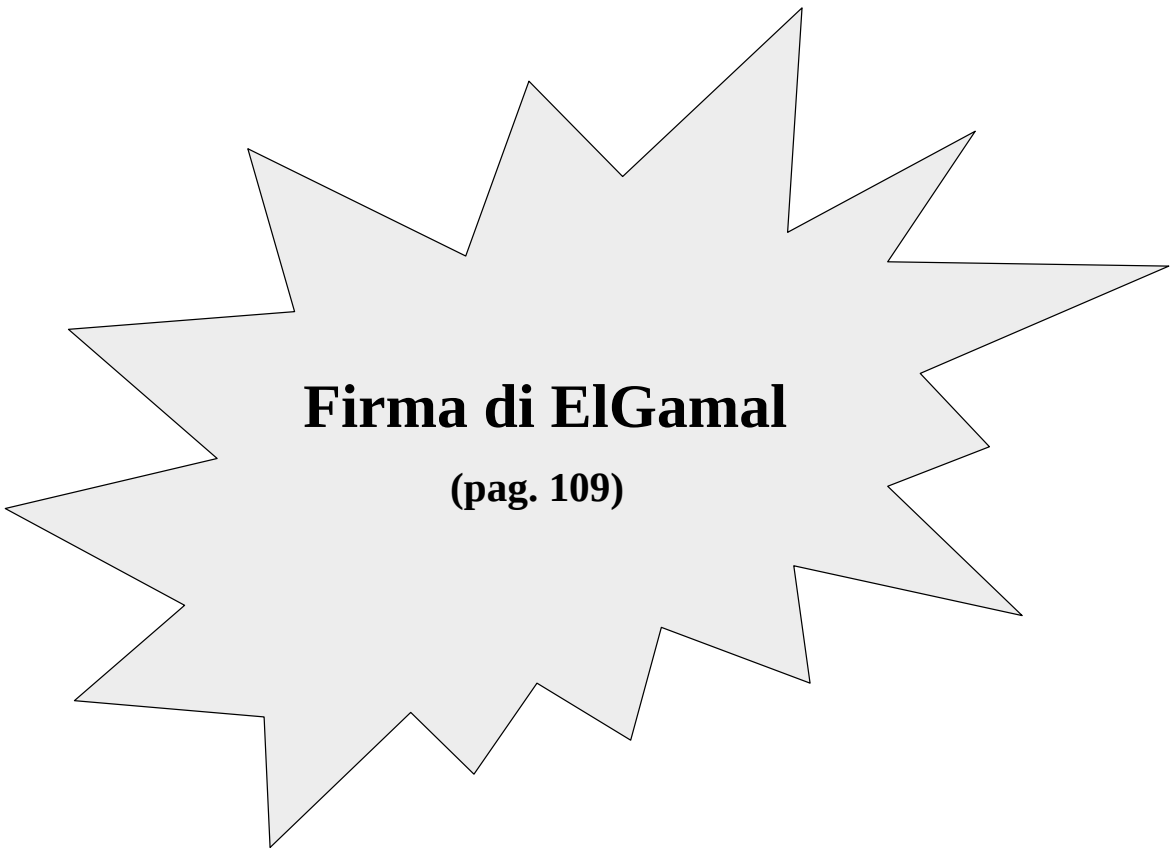


# ElGamal, Kravitz e NIST (DSS)

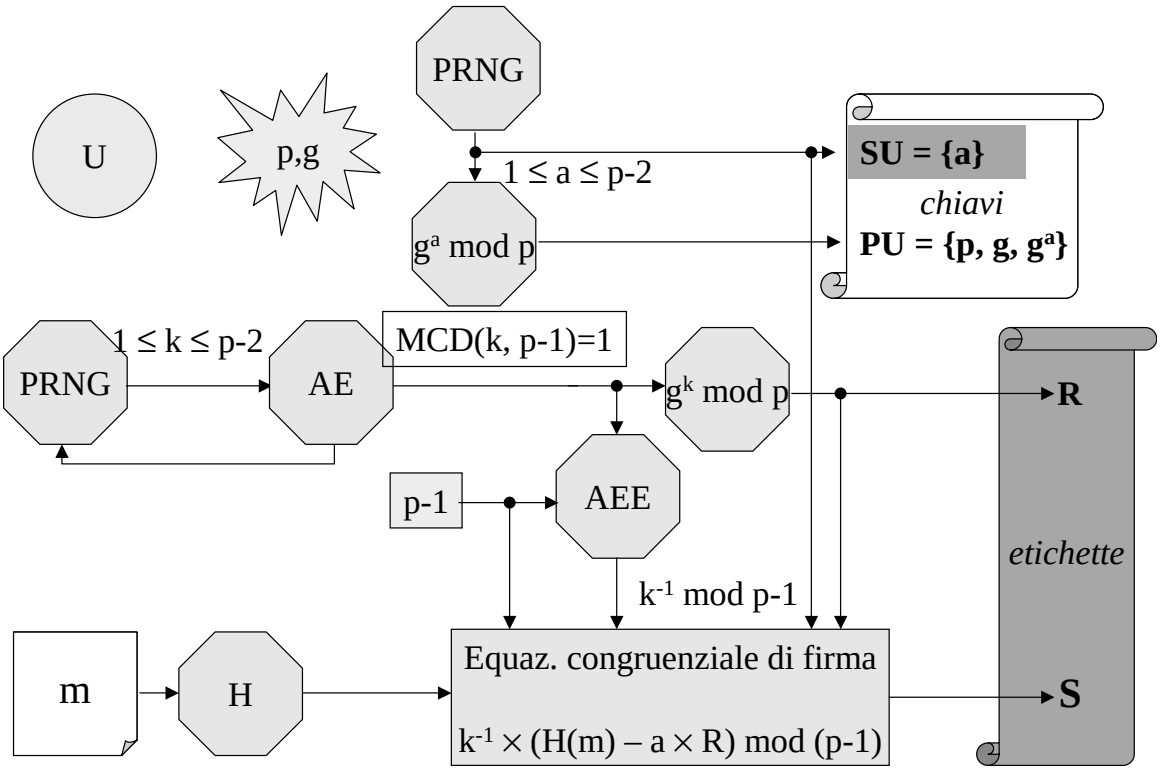
**algoritmo probabilistico con appendice**

**Sicurezza:problema del logaritmo discreto**

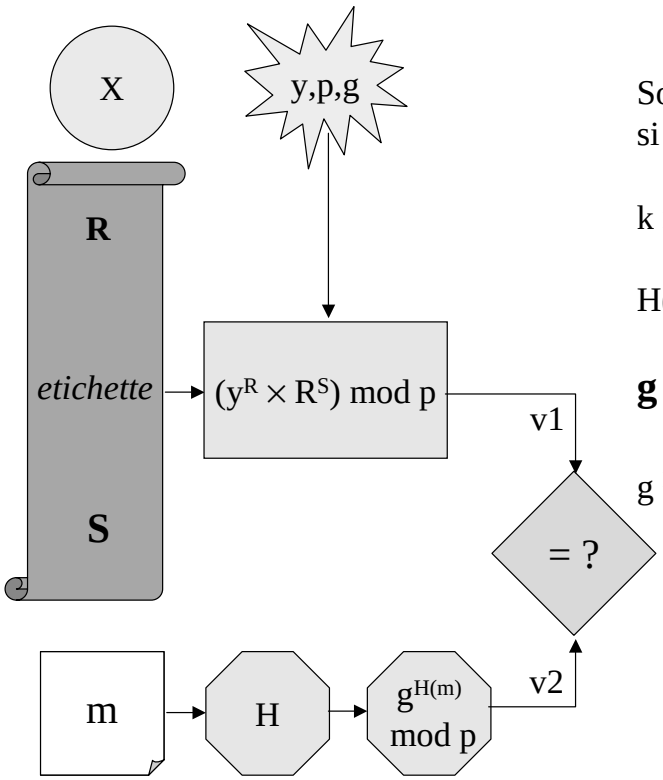
- ElGamal (1985)
- Kravitz (1991)
- NIST (1994)



# L'algoritmo di firma di ElGamal




# L'algoritmo di verifica di ElGamal



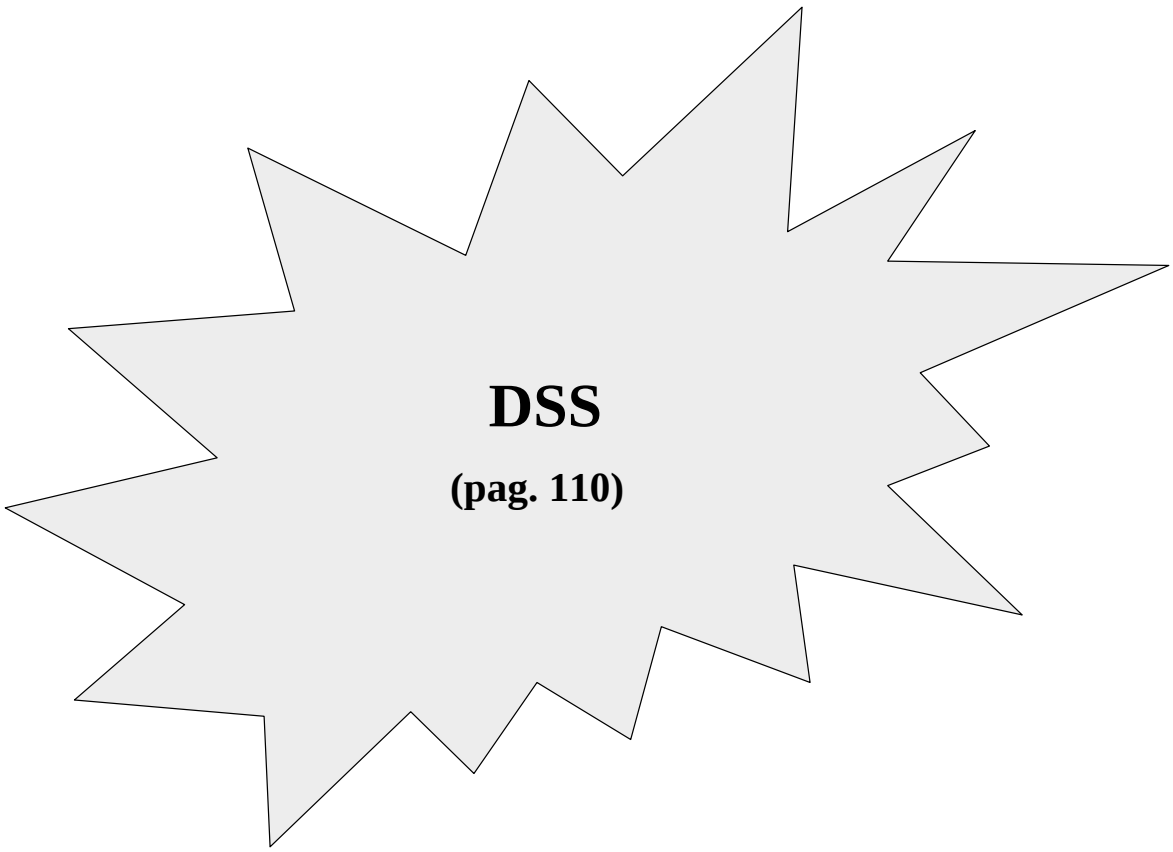

Solo per un messaggio firmato da U si ha:

$k \times S \equiv H(m) - a \times R \pmod{(p-1)}$

$H(m) \equiv a \times R + k \times S \pmod{(p-1)}$

$\mathbf{g^{H(m)} \equiv g^{a \times R + k \times S} \pmod{p}}$

$g^{H(m)} \equiv y^R \times R^S \pmod{p}$




# Digital Signature Algorithm: key generation

Parametri (anche di dominio pubblico e comuni a più utenti):
**p** : numero primo nell'intervallo $2^{512} \div 2^{1024}$ *lunghezza*: n×64 bit
**q** : fattore primo di (p-1) 160 bit
**g** > 1 : *generatore del solo gruppo ciclico di ordine q in* $Z^*_p$
*tale che* $\mathbf{g = h^{(p-1)/q} \bmod p}$ , con 1 < h < p-1

Calcoli segreti del firmatario:
**a**: intero scelto a caso con $1 \leq a \leq q-1$
$\mathbf{y = g^a \bmod p}$

Chiave pubblica: **PU** = (**p**, **q**, **g**, **y**) 1024,160,1024,1024 bit
Chiave privata: **SU = a** 160 bit

# DSS: firma e verifica

Firma :
*Input*: (**p**, **q**, **g**, **a**), **M**
*1. scegli a caso intero*
**k < q**
*2. calcola*
**R = (g$^{k}$ mod p) mod q**
*3. calcola*
**H(M)** *con **H**: SHA-1*
*4. calcola*
**S = k$^{-1}$×(H(M) + a × R) mod q**

Verifica:
*Input*: (**p**, **q**, **g**, **y**) **(m, R, S)**
*1. calcola*
**w ≡ S$^{-1}$ (mod q)**
*2. calcola*
**u$_1$ = [H(m) × w] mod q**
*3. calcola*
**u$_2$ = [R × w] mod q**
*4. calcola*
**v = [(g$^{u1}$ × y$^{u2}$) mod p] mod q**

*Output*: **M,R,S**
*Appendice*: **(R,S)**
**160 + 160 bit**

Se **(v = R)** allora la firma è **OK**

# DSS: giustificazione

*Equazione congruenziale di firma:*
**H(M) = (k × S - a × R) mod q**

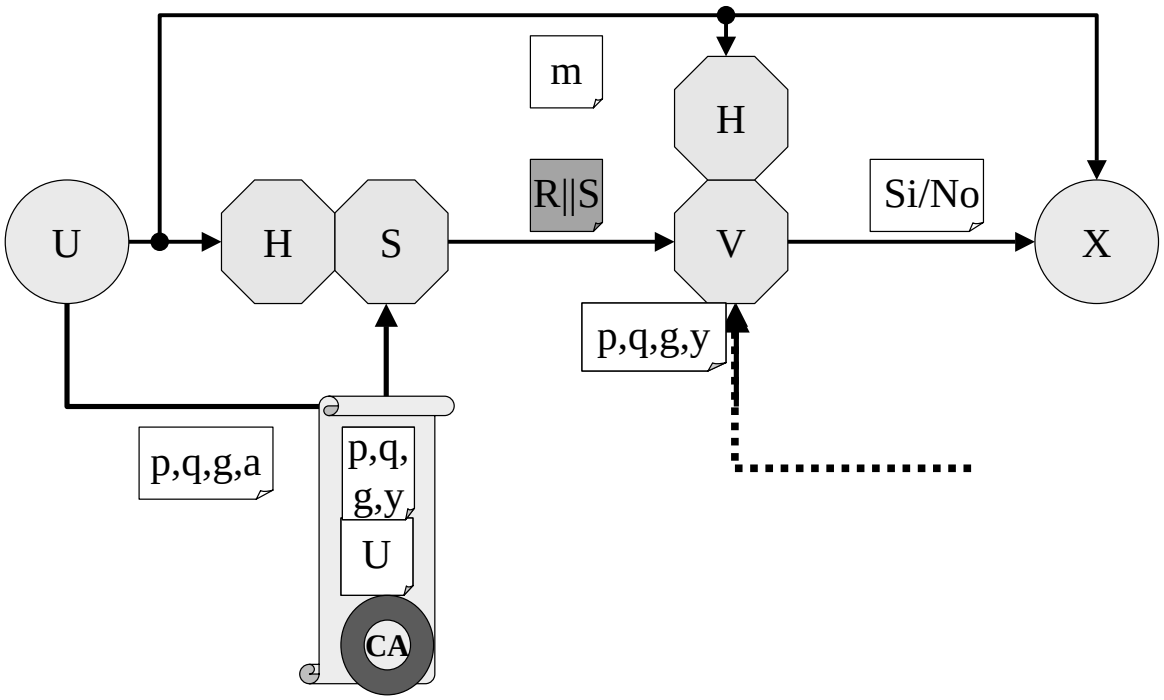
*Moltiplicazione per w ≡S$^{-1}$ (mod q):*
**[w × H(m)] mod q = (w × k × S) mod q – (a × w × R) mod q**
**= k – (a × w × R) mod q**

*e quindi*
**[w × H(m)] mod q + [a × w × R] mod q = k**
**(u$_1$ + a × u$_2$ ) ≡ k (mod q)**

*Esponenziazione con base g*
**g$^{(u1+a.u2) \bmod q}$ mod p = g$^{k \bmod q}$ mod p**

*Tenendo conto che g = h$^{(p-1)/q}$ mod p*
**[(g$^{u1}$ × (g$^{a}$)$^{u2}$) mod p] mod q= (g$^{k}$ mod p) mod q**
**[g$^{u1}$ × y$^{u2}$ mod p] mod q = [g$^{k}$ mod p] mod q**
**v = R**

# Firma con DSS




## Lo Schema di firma RSA

**(pag. 108)**

# Proprietà di reversibilità di RSA

**Reversibilità delle chiavi (impiego di SU al posto di PU e viceversa:**
**$E_{SU}(m) = c = m^{SU} \bmod n$** **messaggio non riservato**
**$D_{PU}(c) = (m^{SU})^{PU} \bmod n = m$** **ma con origine verificabile**

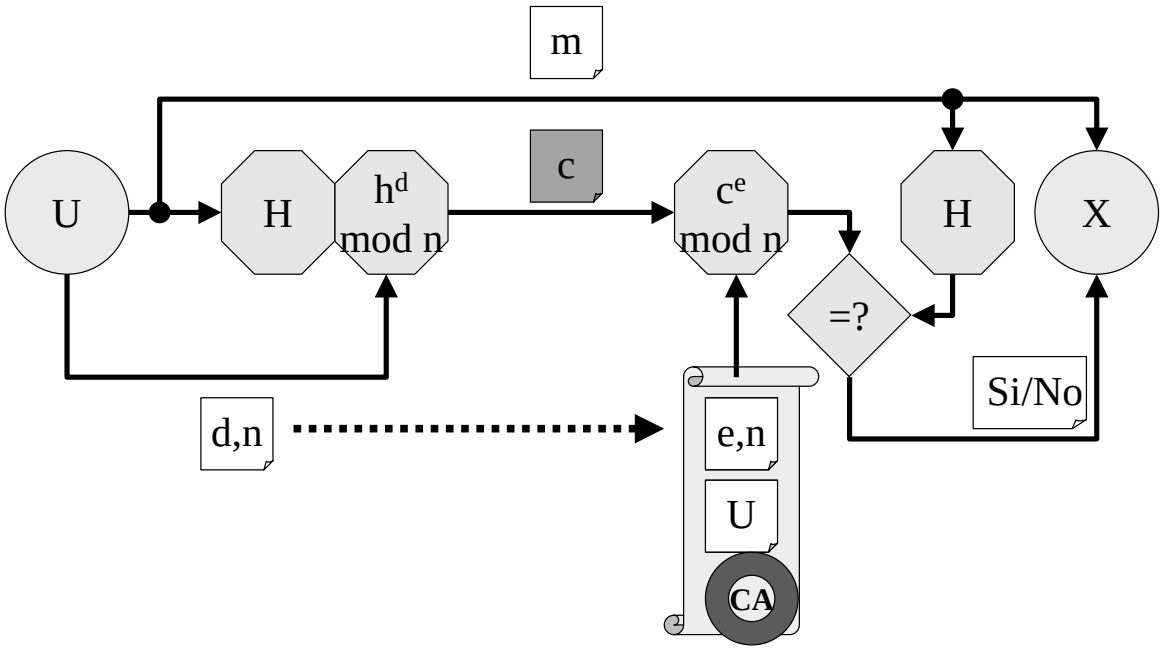
**<u>Schema di firma con recupero</u>** **inefficiente se m > n**

**<u>Schema di firma con appendice</u>**: $\lceil \log_2 n \rceil$ bit di etichetta

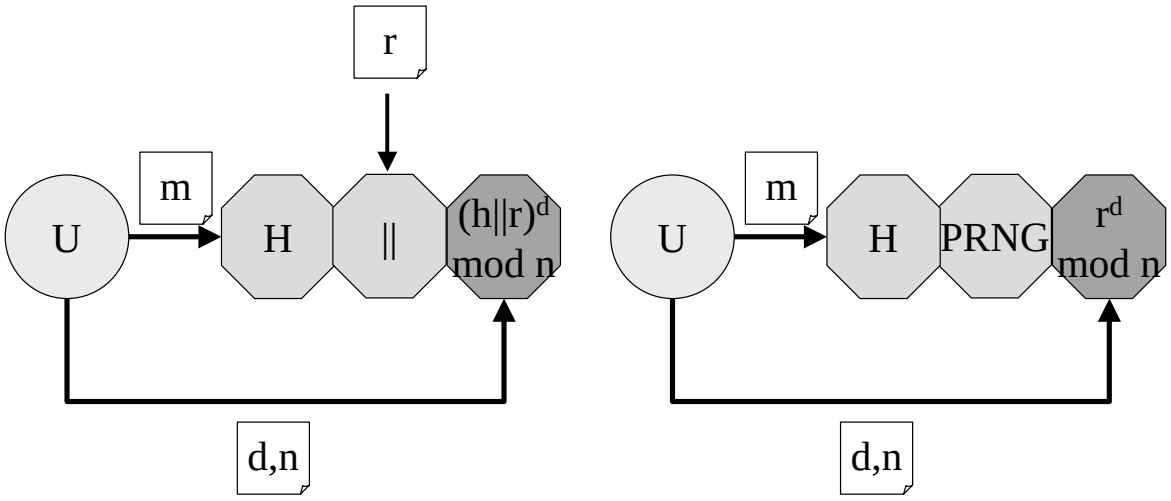
**1. FIRMA**
**$S_{SU}(H(m)) = (H(m))^{SU} \bmod n$** **autenticazione del messaggio**
**$m \| S_{SU}(H(m))$** **comunicazione del messaggio**

**2: VERIFICA**
- **calcolo di H(m)**
- **calcolo di $(S_{SU}(H(m))^{PU} \bmod n$**
- **confronto**

# Firma con RSA



# Padding




PKCS#1

Ferguson-Schneier

# Proprietà moltiplicativa di RSA

Sia $\mathbf{m = m_1 \times m_2 < nU}$
Firma di **m** da parte di **U**:
$\mathbf{c = m^{dU} \bmod nU = (m_1 \times m_2)^{dU} \bmod nU}$
$\mathbf{= ((m_1^{dU} \bmod nU) \times (m_2^{dU} \bmod nU)) \bmod nU}$

Proprietà moltiplicativa dell'algoritmo RSA:
***il testo cifrato (con chiave pubblica o con chiave privata) del prodotto di due testi in chiaro***
***è congruo (mod n) al prodotto dei due testi cifrati***

# Autenticazione di un messaggio oscurato

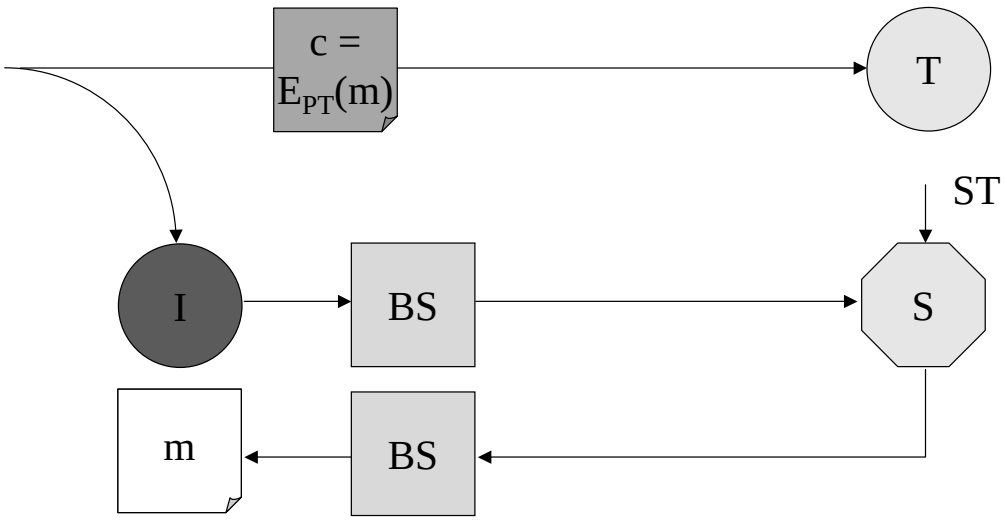
**X vuole farsi autenticare da T un messaggio m senza che T possa conoscerne il contenuto**

Autorizzazioni per voto elettronico, commercio elettronico, ecc.

Autenticazione "a occhi chiusi" di un messaggio **m**

1. **X** sceglie a caso un numero **r**
2. invia a **T** il testo cifrato **c1 = m × $r^{eT}$ mod nT**
3. **T** firma **c1** e restituisce a **X**
   **c2 = $m^{dT}$ × $r^{eT\,dT}$ mod nT** = (**$m^{dT}$ × r) mod nT**
4. **X** moltiplica **c2** per **$r^{-1}$**
   **c3 = ($m^{dT}$ × r × $r^{-1}$) mod nT = $m^{dT}$ mod nU**
5. Il destinatario di **m** può verificare che è autenticato da T
   **m = $(c3)^{eT}$ mod nT = m**

# Attacco con la blind signature




- R30: chi impiega RSA per firmare e per decifrare deve utilizzare due differenti coppie di chiavi


# Altri problemi difficili

**P4: problema del fusto** - Dato un insieme di **n** interi positivi

$$\{a_1, a_2, .., a_n\}$$

ed un intero positivo **s** determinare se esiste o meno un sottoinsieme di $a_j$ la cui somma è **s**.
**Cifrario di Merkle-Hellman, Cifrario di Chor-Rivest**

**P5: problema della radice quadrata modulare** - Dato un **n** composto ed un elemento $a \in Q_n$ (insieme dei residui quadratici modulo n) trovare la sua radice quadrata modulo n, cioè
un intero $x \in Z^*_n$ tale che $x^2 \equiv a \pmod n$.
**Cifrario di Rabin**

**P6: problema della residuosità quadratica** - Dato un **n** composto e dispari ed un elemento $a \in J_p$ (insieme degli interi con simbolo di Jacobi = 1) determinare se **a** è o meno un residuo quadratico modulo **n**.
**Cifrario di Goldwasser-Micali**

# Logaritmo discreto su curva ellittica

**P7: problema del logaritmo discreto su una curva ellittica** - Data la curva ellittica formata da punti le cui coordinate x,y soddisfano l'equazione

$$y^2 = x^3 + ax + b \mod p$$, con p primo,

e dati due suoi punti **P**, **Q** tali che $Q = n \times P$,

determinare **n**.

Complessità degli attuali algoritmi di rottura
$O(\exp(\frac{1}{2}(\log p)))$
160-180 bit

**ECC(Ellipltical Curve Cryptography)**
Certicom.com