Crittografia a chiave pubblica

Cenni alla teoria dei numeri

Numeri Primi

- I numeri primi hanno come divisori solo se stessi e l'unità
  - Non possono essere ottenuti dal prodotto di altri numeri
- eg. 2,3,5,7 sono primi, 4,6,8,9,10 non lo sono
- I numeri primi svolgono un ruolo fondamentale nella teoria dei numeri
- Ecco l'elenco dei numeri primi inferiori a 200:

```
2 3 5 7 11 13 17 19 23 29 31 37 41 43 47 53 59
61 67 71 73 79 83 89 97 101 103 107 109 113 127
131 137 139 149 151 157 163 167 173 179 181 191
193 197 199
```

Fattorizzazione in numeri primi

- Ogni intero può essere diviso in fattori in modo univoco
- N.B.: Fattorizzare un numero è alquanto più complicato che ottenere il numero stesso mediante moltiplicazione dei fattori
- Esempi di fattorizzazione:

$$91 = 7 \times 13 \; ; \; 3600 = 2^4 \times 3^2 \times 5^2$$

Fattorizzazione in numeri primi

- In generale, dato un intero *a,* vale la relazione

$$a=p_1^{a1}p_2^{a2}p_3^{a3}.....$$

ossia

$$a = \prod_{p \in P} p^{a_p}, \qquad a_p \geq 0$$

ove *P* è l'insieme dei numeri primi

Fattorizzazione in numeri primi

- Il valore di un intero positivo lo si può specificare semplicemente indicando i valori degli esponenti non nulli, ossia:

  $12=\{a_2=2, a_3=1\}$, $18=\{a_2=1, a_3=2\}$

- La moltiplicazione di due numeri equivale alla somma degli esponenti:
- $K=mn \rightarrow k_p=m_p+n_p$, per ogni $p \in P$

Numeri Relativamente primi & GCD

- I numeri a,b sono **relativamente primi** se, a parte l'unità, non hanno divisori comuni, ovvero GCD(a,b)=1
  - e.g. 8 & 15 sono relativamente primi
- Il GCD può essere calcolato dalla scomposizione in fattori primi
  - eg. $300=2^1\times3^1\times5^2$ $18=2^1\times3^2$, da cui: $GCD(18,300)=2^1\times3^1\times5^0=6$

Il Piccolo Teorema di Fermat

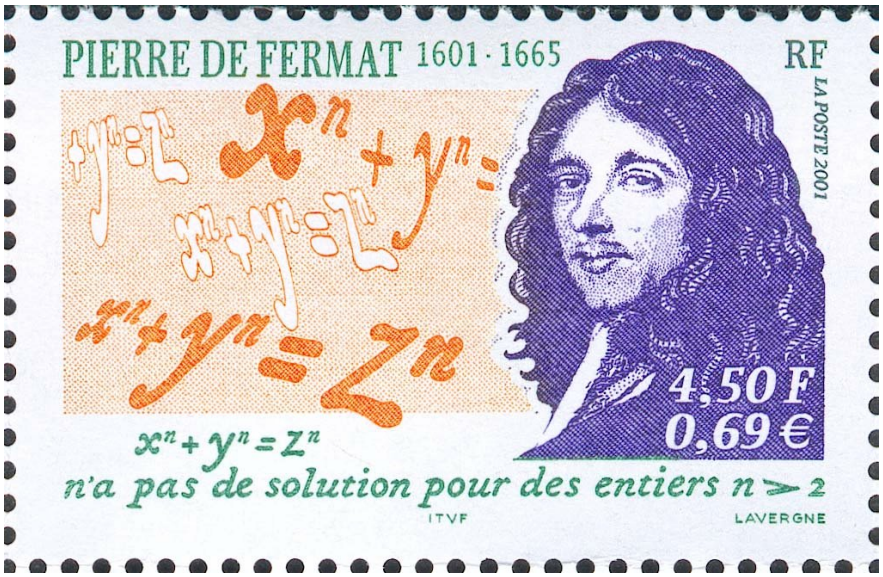
Pierre de Fermat

Born: **17.09.1601 in Beaumont-de-Lomagne**

Died: **12.01.1665 in Castres**

Dato un numero primo p e un intero a tale che gcd(a,p)=1, è:

$$a^{p-1} \bmod p = 1$$



Il Piccolo Teorema di Fermat: Prova (1/2)

- Poichè p è primo, $Z_p=\{0, \ldots, p-1\}$ è tale che moltiplicando modulo p gli elementi di Z_p per a, si riottengono gli stessi elementi di Z_p in ordine diverso;
- Moltiplicando gli elementi di entrambi gli insiemi (ad eccezione dello 0) si ottiene

  $1 \times 2 \times \ldots \times (p-1) \bmod p = (p-1)! \bmod p$

  $a \times 2a \times \ldots \times (p-1)a \bmod p = a^{p-1}(p-1)! \bmod p$

Il Piccolo Teorema di Fermat: Prova (2/2)

- Pertanto si ha
  - $(p-1)!a^{p-1} \bmod p = (p-1)! \bmod p$
- Semplificando (p-1)! (lo si può fare, perchè??) si ha la tesi.

Una formulazione equivalente del teorema è ovviamente

$$a^p \bmod p = a$$

La Funzione Toziente di Eulero $\Phi(n)$

- Dato un intero positivo n, la funzione toziente di Eulero $\Phi(n)$ è pari al numero di interi positivi minori di n e primi relativi con n, ovvero

$$\Phi(n)=\text{card}(\{x \in Z_n : \gcd(x,n)=1\})$$

- Esempi:

  $\Phi(3)=2$, $\Phi(5)=4$, $\Phi(6)=2$.

- Nota: $\Phi(p)=p-1$, per ogni p numero primo

La Funzione Toziente di Eulero

Table 8.2 Some Values of Euler's Totient Function $\phi(n)$

n	$\phi(n)$	n	$\phi(n)$	n	$\phi(n)$
1	1	11	10	21	12
2	1	12	4	22	10
3	2	13	12	23	22
4	2	14	6	24	8
5	4	15	8	25	20
6	2	16	8	26	12
7	6	17	16	27	18
8	4	18	6	28	12
9	6	19	18	29	28
10	4	20	8	30	8

La Funzione Toziente di Eulero $\Phi(n)$

- Il calcolo di $\Phi(n)$ richiede in genere la verifica e la conta esaustiva degli elementi di Z_n relativamente primi con n
- Valgono le seguenti eccezioni
  - Se p è primo $\Phi(p) = p-1$
  - Se p e q sono primi $\Phi(pq) = (p-1)(q-1)$
- eg.
  - $\Phi(37) = 36$
  - $\Phi(21) = (3-1)\times(7-1) = 2\times6 = 12$

Teorema di Eulero

- È una generalizzazione del teorema di Fermat

  Se a e n sono primi relativi, allora

  $$a^{\Phi(n)} \bmod n = 1$$

- Esempi
  - $a=3; n=10;\ \Phi(10)=4;$
  - quindi $3^4 = 81 = 1 \bmod 10$
  - $a=2; n=11;\ \Phi(11)=10;$
  - hence $2^{10} = 1024 = 1 \bmod 11$

Teorema di Eulero: Prova (1/3)

- Se n è primo, $\Phi(n)=n-1$ e la prova è banale (th. di Fermat)
- Se n non è primo, si considerino i $\Phi(n)$ interi positivi minori di n e relativamente primi con n: $R=\{x_1, \ldots , x_{\Phi(n)}\}$
- Si consideri ora l'insieme S dato da

$$S=\{ax_1 \bmod n , \ldots , ax_{\Phi(n)} \bmod n\}$$

Teorema di Eulero: Prova (2/3)

- L'insieme S coincide con R in quanto
  - a e x_i sono entrambi primi relativi con n, e quindi lo è anche ax_i. Pertanto gli elementi di S sono tutti interi minori di n e primi relativi di n
  - in S non vi sono duplicati in quanto se $ax_i \bmod n = ax_j \bmod n$, allora $x_i=x_j$.

Teorema di Eulero: Prova (3/3)

Ma allora, se R=S, si ha

$$\prod_{i=1}^{\Phi(n)} (ax_i \bmod n) = \prod_{i=1}^{\Phi(n)} x_i \bmod n$$

da cui segue banalmente la tesi.

Una forma alternativa del teorema è

$$\mathbf{a^{\Phi(n)+1} \bmod n = a}$$

Perchè studiamo questa roba??

- Siano p e q due primi, sia $n=pq$ e sia $0<m<n$
- Vale la relazione

$$\mathbf{m^{\Phi(n)+1} = m^{(p-1)(q-1)+1} \bmod n = m \bmod n}$$

  Se m ed n sono primi relativi, la relazione è vera per il teorema di Eulero. In realtà, tale relazione si dimostra essere valida anhe se m ed n non sono primi relativi
- Tale relazione è alla base del funzionamento dell'algoritmo RSA di crittografia a chiave pubblica

Test di primalità

- Nella crittografia, c’è bisogno di procedure per generare numeri primi grandi
- Dato un numero generato in maniera pseudocasuale, bisogna stabilire se è un numero primo
- Una procedura sistematica consiste nel provare a dividere per tutti i numeri primi minori della radice quadrata del numero sotto test
- Chiaramente, al crescere di n tale approccio è inutilizzabile

Test di primalità

- Si ricorre quindi ad un approccio statistico, ovvero si usano test che garantiscono che un numero sia primo “con una certa affidabilità,” ovvero in senso probabilistico
- Tali test si basano sulla verifica di una proprietà
  - Soddisfatta da tutti i numeri primi
  - **... Ma usualmente soddisfatta anche da qualche numero composto**

Test di Fermat

- Il teorema di Fermat afferma che se n è primo, per ogni a<n tale che gcd(a,n)=1 si ha:

$$\mathbf{a^{n-1}=1 \bmod n}$$

  Esempio: consideriamo 341=11 x 31

  $2^{340}=1 \bmod 341$

  $3^{340}=56 \bmod 341 \Rightarrow$ 341 è composto

- N.B.: Il test di Fermat permette di stabilire con certezza che un numero è composto, ma non può provare che esso sia primo
- Inoltre, il test di Fermat prova che un numero è composto, ma non fornisce la sua scomposizione in fattori primi

I numeri di Carmichael

- Consideriamo il numero 561=3 x 11 x 17
- E’ possibile verificare che, comunque si scelga a < 561 e primo con 561, il test di Fermat non riesce a dimostrare la non primalità di 561
- Sfortunatamente, esistono infiniti numeri di tal tipo
- Essi sono detti *numeri di Carmichael*

I numeri di Carmichael

- 561 è il più piccolo numero di Carmichael
- E’ stato dimostrato che esistono infiniti numeri di Carmichael
- 561, 1105, 1729, 2465, 2821, 6601, 8911, 10585, 15841, 29341, ... sono tutti numeri di Carmichael
- Di conseguenza, il test di primalità di Fermat non è affidabile!

Test di Miller Rabin

- E’ basato sul teorema di Fermat
- Diamo l’algoritmo senza dimostrazione

TEST (n) is:

1. Find integers k, q, $k > 0$, q odd, so that $(n-1)=2^k q$
2. Select a random integer a, $1<a<n-1$
3. **if** $a^q \bmod n = 1$ **then** return (“maybe prime"), STOP;
4. **for** $j = 0$ **to** $k - 1$ **do**
   5. **if** $(a^{2^j q} \bmod n = n-1)$
      then return(" maybe prime "), STOP
6. return ("composite"), STOP

Test di Miller Rabin

Esempio: n=29; n-1=28=2^2 x 7; si prenda a=10;

10^7 mod 29=17 che è diverso da 1 e –1

$10^{7 \times 2}$ mod 29=28 $\Rightarrow$ **MAYBE PRIME**

Si riprovi con a=2

2^7 mod 29=12 che è diverso da 1 e –1

$2^{7 \times 2}$ mod 29=28 $\Rightarrow$ **MAYBE PRIME**

Di fatto, il risultato “maybe prime” si ottiene con tutti gli interi a compresi tra 1 e 28.

Test di Miller Rabin

Esempio: $n=221=13 \times 17$; $n-1=220=2^2 \times 55$; si prenda $a=5$;

$5^{55} \bmod 221=112$ che è diverso da 1 e –1

$5^{55 \times 2} \bmod 221=168 \Rightarrow$ **COMPOSITE**

Se avessimo scelto $a=21$ si sarebbe avuto

$21^{55} \bmod 221=200$ che è diverso da 1 e –1

$21^{55 \times 2} \bmod 221=220 \Rightarrow$ **MAYBE PRIME**

Di fatto, il risultato “maybe prime” si ottiene con 6 dei 220 interi compresi tra 1 e 220.

Considerazioni Probabilistiche

- Se l'esito del test di Miller-Rabin è "composite" il numero è **sicuramente** non primo
- In caso contrario, **potrebbe** essere primo
- È stato mostrato che, se n è composto, scegliendo a caso la base a, la probabilità che il test dia come output "maybe prime" è < ¼
- Ripetendo il test t volte con basi a scelte a caso ho che la probabilità che il test dia un output "maybe prime" in presenza di un numero composto è 4^{-t}
- Un numero composto viene identificato quindi con probabilità $1-4^{-t}$
  - E.g., per t=10 tale probabilità è > 0.99999

Distribuzione dei numeri primi

- DOMANDA: Generato un numero dispari n grande, con che probabilità questo è un numero primo???
- Il “teorema dei numeri primi” stabilisce che, per n grande, i numeri primi sono spaziati mediamente ogni `ln(n)` interi
- Poichè non si considerano i numeri pari e quelli che terminano per 5, in pratica c’è bisogno in media di `0.4 ln(n)` tentativi per poter identificare un numero primo

Distribuzione dei numeri primi

- Nota però che questi sono risultati validi in media....

  ESEMPI:

  – Gli interi consecutivi $10^{12}+61$ e $10^{12}+63$ sono entrambi primi

  – I numeri 1001!+2, 1001!+3, ... 1001!+1001 è una sequenza di mille interi consecutivi e non primi

Il Teorema Cinese del Resto

- È utilizzato per velocizzare i calcoli dell'aritmetica modulare
- Si supponga di dover lavorare modulo un certo numero M che è a sua volta prodotto di numeri a coppia relativamente primi
  - eg. mod $M = m_1 m_2 .. m_k$
- Il Teorema Cinese del Resto ci permette di lavorare con singolarmente modulo ciascuno dei fattori m_i
- Questo permette di risparmiare notevolmente in termini di risorse computazionali

Il Teorema Cinese del Resto

- Il teorema afferma che è possibile ricostruire gli interi di un determinato intervallo a partire dai loro resti modulo una coppia di numeri relativamente primi
- ESEMPIO: Gli elementi in Z_{10} possono essere rappresentati dai loro resti modulo 2 e 5

Il Teorema Cinese del Resto

- Sia $M=m_1 m_2 \ldots m_k$, con $\gcd(m_i,m_j)=1$ per ogni $i \neq j$
- Per ogni intero $A \in Z_M$, posto $a_i = A \bmod m_i$, la corrispondenza $A \leftrightarrow (a_1, a_2, \ldots, a_k)$ è biunivoca

  Si tratta in realtà di un mappaggio da Z_M a $Z_{m1} \times Z_{m2} \times \ldots \times Z_{mk}$

Il Teorema Cinese del Resto

- Si supponga di voler calcolare (A mod M)
- Dapprima si calcolano separatamente gli a_i =(A mod m_i) e poi si combinano insieme secondo le formule:

$$c_i = M_i \mathrm{x} \left(M_i^{-1} \bmod m_i \right) \qquad \text{per } 1 \le i \le k$$

$$A = \left(\sum_{i=1}^{k} a_i c_i \right) \bmod M$$

ove M_i=M/m_i

Il Teorema Cinese del Resto

ESEMPIO:

M=1813=37 x 49, A=973

a_1=973 mod 37=11, a_2=973 mod 49=42

M_1=49, M_2= 37

$(49)^{-1}$ mod 37=34, $(37)^{-1}$ mod 49=4

c_1 =49 x 34=1666, c_2 =4 x 37=148

A=(1666 x 11 + 148 x 42) mod M=973

Radici Primitive

- Sappiamo dal teorema di Eulero che per a ed n primi relativi si ha $a^{\Phi(n)} \bmod n = 1$
- Si consideri l'equazione in m: $a^{m} \bmod n = 1$, con gcd$(a,n)=1$
  - Una soluzione è certamente m= Φ(n), ma può esserci una soluzione minore di Φ(n)
  - Le potenze si ripetono poi ciclicamente
- Se la più piccola soluzione è m= Φ(n) allora a è detto una **radice primitiva** di n
- Se n è primo, allora le potenze successive di a generano Z_p ($=Z_n$)

Radici Primitive

ESEMPIO: Potenze di 7 modulo 19

$7^1 \bmod 19 = 7$

$7^2 \bmod 19 = 11$

$7^3 \bmod 19 = 1$

$7^4 \bmod 19 = 7$

$7^5 \bmod 19 = 11$

.....

Radici Primitive

- Non tutti gli interi hanno radici primitive
- Gli unici interi con radici primitive sono nella forma 2, 4, p^{α} e $2p^{\alpha}$.

ESEMPIO:

Le radici primitive di 19 sono 2, 3, 10, 13, 14 e 15

Logaritmi Discreti

- Abbiamo trattato dell'esponenziazione modulare
- La funzione inversa dell'esponenziazione modulare è il cosiddetto **logaritmo discreto** di un numero modulo p
- Il logaritmo di b in base a e modulo p è quell'intero x tale che $a^x = b \bmod p$
- Si usa la notazione $x=\log_a b \bmod p$ o $x=\mathrm{ind}_{a,p}(b)$

Logaritmi Discreti

- In generale, non è garantita l'esistenza del logaritmo discreto
- Se a è una radice primitiva di m, allora il logaritmo in base a e modulo m esiste sempre
  - $x = \log_3 4 \bmod 13$ (ovvero trova x tale che $3^x = 4 \bmod 13$) non ha soluzione
  - $x = \log_2 3 \bmod 13 = 4$ per ricerca esaustiva
- N.B.: Mentre l'elevazione a potenza è alquanto semplice, il calcolo del logaritmo discreto è unanimemente riconosciuto non fattibile quando m diventa grande

Sommario

- Ieri abbiamo parlato di:
  - Numeri primi
  - I Teoremi di Fermat e di Eulero
  - Test di primalità (statistici)
  - Il Teorema Cinese del Resto
  - Logaritmi Discreti

Richiami Lezione Precedente

- Il Piccolo Teorema di Fermat

Dato un numero primo p e un intero a tale che gcd(a,p)=1, è:

$$a^{p-1} \bmod p = 1$$

Richiami Lezione Precedente

- Il teorema di Eulero

Se a e n sono primi relativi, allora

$$a^{\Phi(n)} \bmod n = 1$$

Richiami Lezione Precedente

- I Test di primalità
  - Test di Fermat
  - Numeri di Carmichael
  - Test di Miller-Rabin
- Il Teorema cinese del resto

Richiami Lezione Precedente

- Radici primitive:

  a è detta radice primitiva di n se l'equazione $a^m \bmod n=1$, ammette come soluzione m più piccola $m=\Phi(n)$.

  Se n è primo, $a^m \bmod n$ genera, al variare di m, tutti gli elementi di Z_n ad eccezione dello 0

Richiami Lezione Precedente

- Logaritmi discreti
  - Il logaritmo di b in base a e modulo p è quell'intero x tale che $a^x = b \bmod p$
  - In generale, non è garantita l'esistenza del logaritmo discreto
  - Se a è una radice primitiva di m, allora il logaritmo in base a e modulo m esiste sempre
  - Il calcolo del logaritmo discreto di numeri grandi è complicato!!

Oggi parleremo di....

- Crittografia a chiave pubblica
  - L’algoritmo RSA
- Procedure per lo scambio delle chiavi

Crittografia a singola chiave

- Come detto più volte, la **crittografia a chiave segreta** usa un'unica chiave
- La chiave è condivisa da trasmettitore e ricevitore
- Se la chiave è violata, la comunicazione non è più sicura
- La crittografia è simmetrica, ovvero i due attori della comunicazione sono paritetici
- Quindi il ricevitore potrebbe coniare un messaggio, crittografarlo e sostenere che tale messaggio sia stato inviato dal trasmettitore!!

Crittografia a chiave pubblica

- Probabilmente, trattasi della scoperta più importante nella storia millenaria della crittografia
- utilizza **due** chiavi: una è pubblica e una è privata
- È **asimmetrica** poiché i soggetti della comunicazione non sono uguali
- Si basa su concetti avanzati della teoria dei numeri
- Tuttavia, essa complementa e non rimpiazza l'uso della crittografia simmetrica

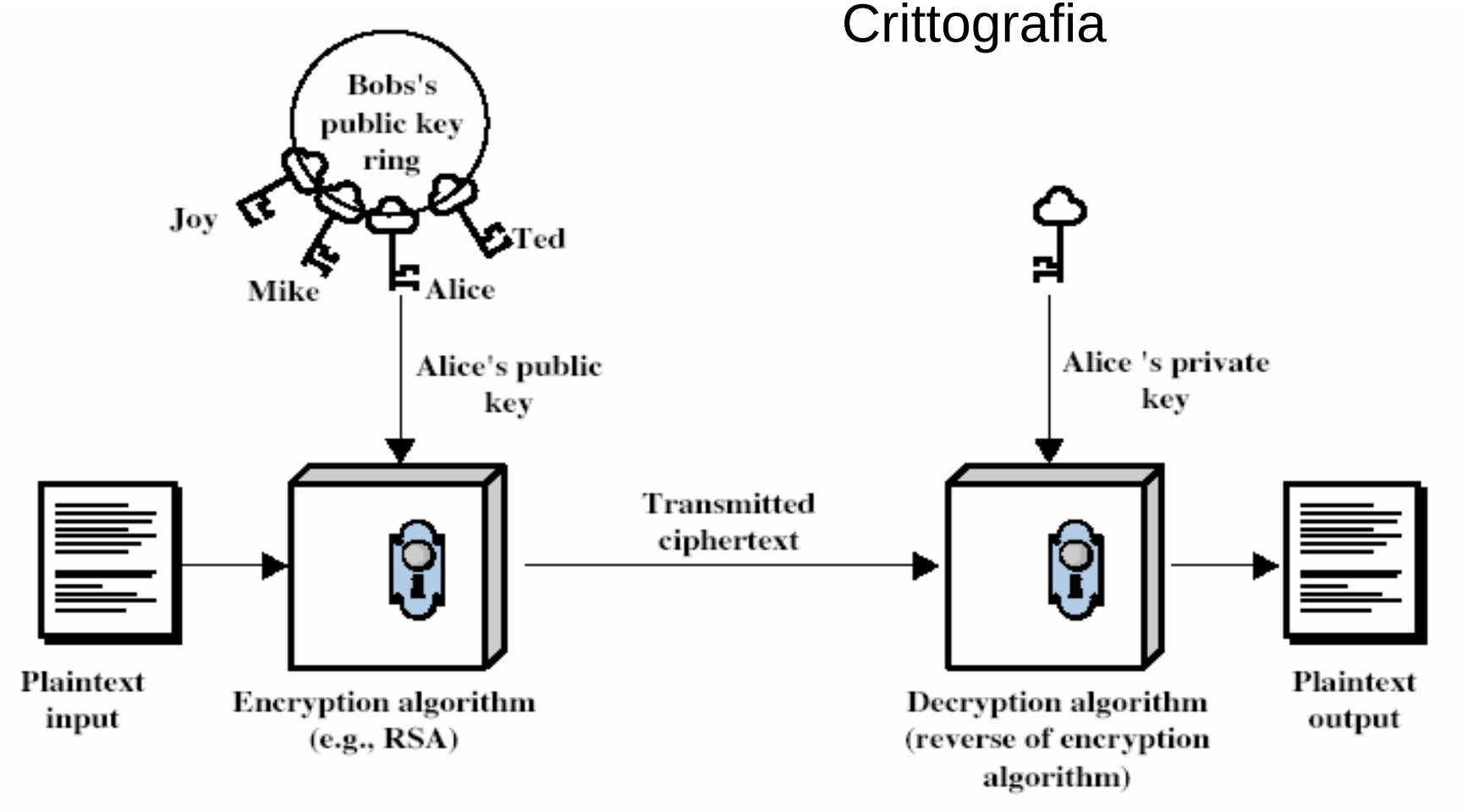
Crittografia a chiave pubblica

- La crittografia **a chiave pubblica/a due chiavi/asimmetrica** si basa sull'uso di **due** chiavi:
  - una **chiave pubblica**, che è pubblicamente nota, ed è usata per **criptare messaggi** e **verificare le firme**
  - una **chiave privata**, nota solo a soggetto della comunicazione, usata per **decriptare i messaggi** e **firmare i messaggi**

Crittografia a chiave pubblica

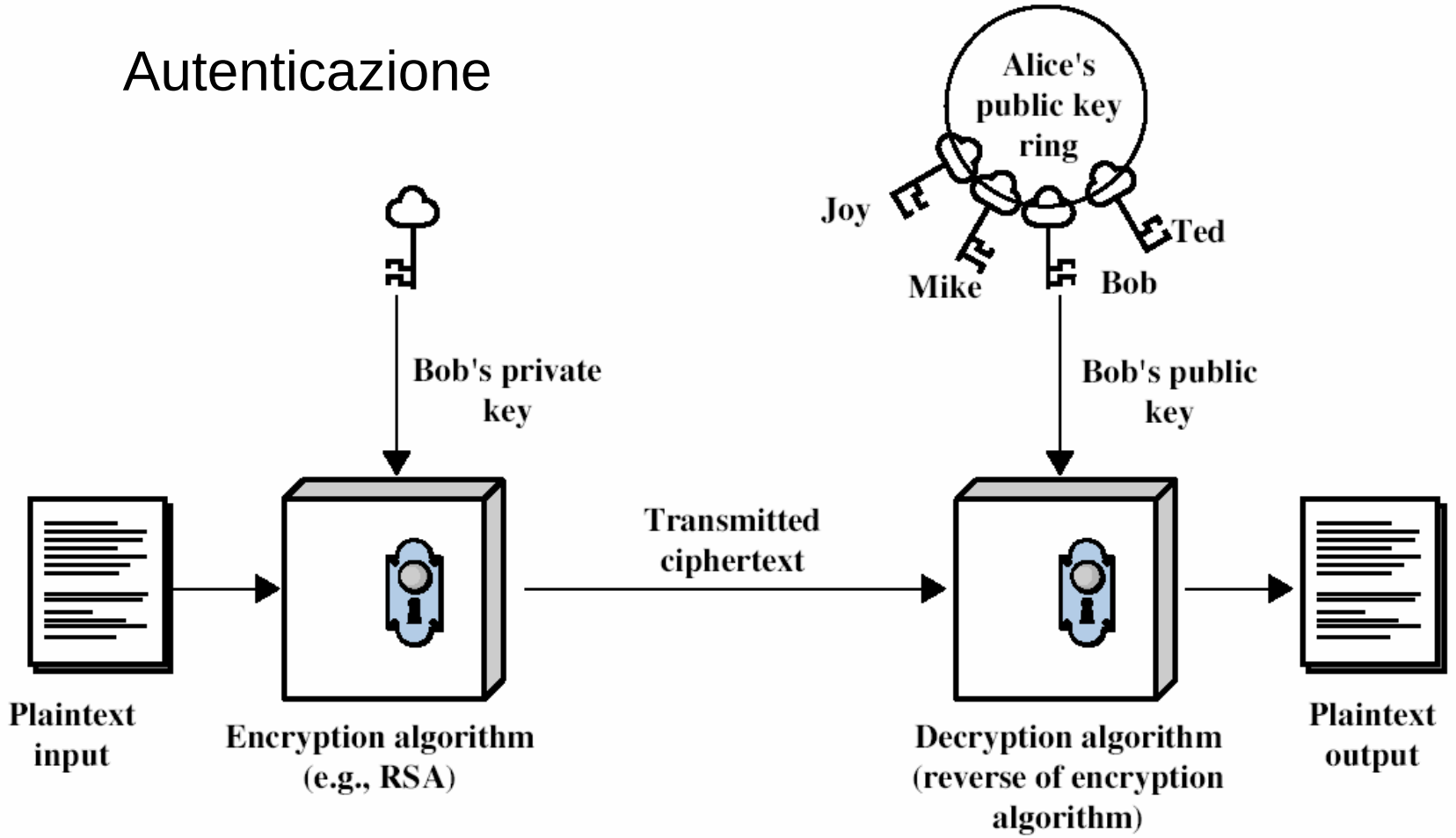
- è **asimmetrica** perchè
  - Coloro che criptano i messaggi o verificano le firme **non possono** decriptare i messaggi o creare le firme

Crittografia a chiave pubblica



Crittografia a chiave pubblica

Autenticazione



Perchè è stata inventata la crittografia a due chiavi??

- Per risolvere due problemi fondamentali:
  - **La distribuzione della chiave** – come possono aversi comunicazioni sicure senza dover porre fiducia in un KDC?
  - **Firme digitali** – come è possibile verificare che un messaggio proviene in forma intatta dal presunto mittente?

Crittografia a chiave pubblica

- Fu introdotta nel 1976 da Whitfield Diffie e Martin Hellman, all'epoca ricercatori presso la Stanford University (California)

Crittografia a chiave pubblica

- Di fatto, il concetto di crittografia a chiave pubblica era già noto alla NSA dalla metà degli anni 60.
- Il lavoro di Diffie&Hellman indusse gli studiosi di crittografia a progettare schemi a chiave pubblica.
- Nel 1977, Ron **R**ivest, Adi **S**hamir e Len **A**dleman del MIT concepirono l'algoritmo RSA

Caratteristiche

- Gli algoritmi a chiave pubblica si basano su due chiavi con le seguenti caratteristiche
  - È computazionalmente irrealizzabile trovare la chiave di decriptazione sulla base della conoscenza dell'algoritmo e della chiave di criptazione
  - È computazionalmente facile criptare e decriptare i messaggi se si conoscono le rispettive chiavi
  - Una delle due chiavi può essere usata per la cifratura, e l'altra deve essere usata per la decifratura (solo però in alcuni schemi)

Combinazione di segretezza e autenticazione

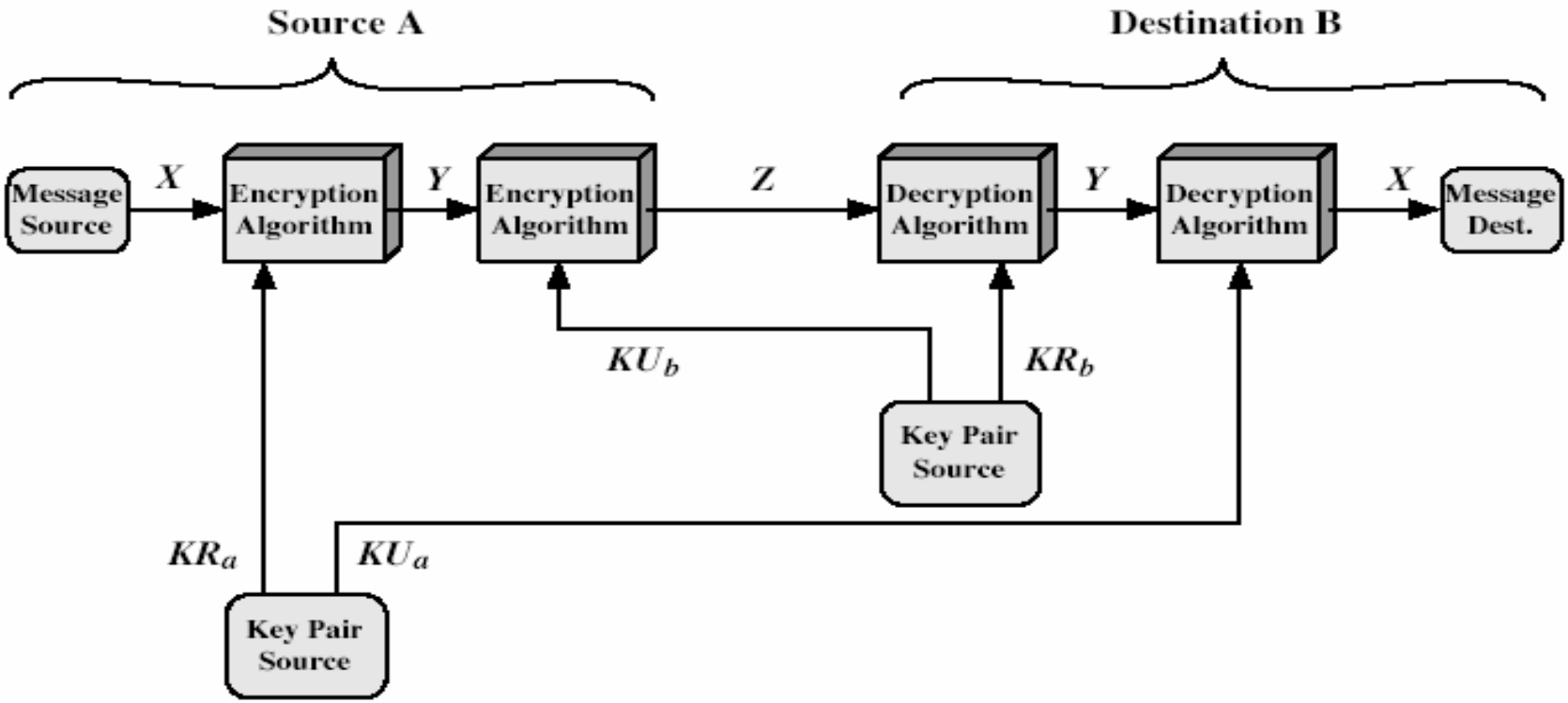


Figure 9.4 Public-Key Cryptosystem: Secrecy and Authentication

Applicazioni della Crittografia a chiave pubblica

- Gli usi rientrano in tre categorie:
  - **Criptazione/decriptazione** (per ottenere la segretezza)
  - **Firma digitale** (per ottenere l'autenticazione)
  - **Scambio di chiavi** (ovvero, di chiavi di sessione)
- Alcuni algoritmi possono essere usati per tutti e tre gli usi, altri no.

Sicurezza degli schemi a chiave pubblica

- Come nella crittografia simmetrica, l'**attacco a forza bruta** è sempre teoricamente possibile
- Si usano chiavi molto lunghe (>512bits)
- La sicurezza si basa sulla difficoltà della criptoanalisi
- In generale, si sa come rompere il codice, solo che ciò è talmente complicato da essere irrealizzabile
- Bisogna utilizzare **numeri molto grandi**
- Gli schemi a chiave pubblica sono **più lenti** degli schemi a chiave segreta

RSA

- by Rivest, Shamir & Adleman del MIT in 1977
- È lo schema a chiave pubblica maggiormente noto ed usato
- È basato sull'elevazione a potenza in aritmetica modulo un intero molto grande
  - N.B. l'esponenziazione richiede $O((\log n)^3)$ operazioni (è facile)
- Utilizza interi dell'ordine di 1024 bit
- La sicurezza è basata sulla difficoltà nel fattorizzare grandi numeri interi
  - N.B. la fattorizzazione richiede $O(e^{\log n \log \log n})$ operazioni (è complicata)

RSA

- Il testo in chiaro è cifrato a blocchi; ogni blocco è un intero minore di un dato n
- Ovvero il blocco è lungo k bit con $2^k<n$
- Detto M il plaintext e C il ciphertext, le operazioni di crittografia e decrittografia si esprimono:

$$C=M^e \bmod n$$

$$M=C^d \bmod n=M^{ed} \bmod n$$

RSA

- n è noto sia al mittente che al destinatario
- Il mittente conosce e (che è pubblica)
- **Solo** il destinatario conosce d
- Quindi KR={d, n}, KU={e, n}

RSA

- Requisiti:
  - È possibile trovare i valori di `n`, `d`, `e` tali che $M^{ed} \bmod n = M$,
  - È relativamente facile calcolare M^e e C^d per tutti i valori di $M<n$
  - È computazionalmente impossibile determinare d sulla base di e ed n

Scelta delle chiavi in RSA (1/2)

- Le due chiavi di ciascun utente sono scelte come segue:
- Si generano due numeri primi molto grandi: p, q
- Si pone poi $n=pq$
  - Nota che $\Phi(n)=(p-1)(q-1)$
- Si seleziona in maniera aleatoria la chiave e
    - ove $1<e<\Phi(n)$, $\gcd(e,\Phi(n))=1$

Scelta delle chiavi in RSA (2/2)

- La chiave di decriptazione d è l'inversa di e modulo $\Phi(n)$
  - $d=e^{-1} \bmod \Phi(n)$ con $0 \leq d \leq n$
- La chiave pubblica di criptazione è: KU={e,n}
- La chiave segreta di decriptazione è: KR={d,p,q}

Uso di RSA

- Per criptare un messaggio M il mittente:
  - Si procura la **chiave pubblica** del destinatario $KU=\{e,n\}$
  - calcola: $C=M^e \bmod n$, ove $0 \leq M < n$
- Per decriptare il testo cifrato C il destinatario:
  - utilizza la sua chiave privata $KR=\{d,p,q\}$ calcolando la quantità $M=C^d \bmod n$
- N.B. Deve essere $M<n$

RSA: Perchè funziona? (1/2)

- Per il teorema di Eulero è: $a^{\Phi(n)} \bmod n = 1$
  - ove è $\gcd(a,n)=1$
- Un corollario del teorema di Eulero, che non dimostriamo, stabilisce che
  - Dati due numeri primi p e q e gli interi $n=pq$, $m<n$, e $k>0$, vale la relazione

$$m^{k\Phi(n)+1} = m \bmod n$$

RSA: Perchè funziona? (2/2)

- in RSA si ha:
  - $n=pq$
  - $\Phi(n)=(p-1)(q-1)$
  - e e d sono selezionati in modo che siano inversi in aritmetica modulo $\Phi(n)$
  - Quindi è $ed=1+k\ \Phi(n)$ per qualche k

- Da cui:

$$C^d = (M^e)^d = M^{1+k\Phi(n)} = M\ (M^{\Phi(n)})^k = M\ (1)^k = M^1 = M \bmod n$$

RSA

- In sintesi, i valori di n ed e sono noti
- d si potrebbe ottenere calcolando l'inverso di e modulo $\Phi(n)$
- Tuttavia $\Phi(n)$ è incalcolabile, a meno che non si sappia che $n = pq$ e che quindi

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

- Ecco perchè la forza di RSA si basa sul fatto che è difficile fattorizzare grandi numeri interi.

RSA: Un esempio

1. Seleziona i numeri primi: $p=17$ & $q=11$
2. Calcola $n = pq = 17 \times 11 = 187$
3. Calcola $\Phi(n)=(p-1)(q-1)=16 \times 10=160$
4. Seleziona e: $\gcd(e,160)=1$; scegli $e=7$
5. Determina d: $de=1 \bmod 160$ e $d < 160$ tale valore è $d=23$ poichè $23 \times 7=161= 10 \times 160+1$
6. La chiave pubblica è $KU=\{7,187\}$
7. La chiave privata è $KR=\{23,17,11\}$

RSA: Un Esempio

- Esempio di criptazione e decriptazione
- Si consideri il messaggio $M = 88$ (n.b. $88<187$)
- criptazione:

  $C = 88^7 \bmod 187 = 11$
- decriptazione:

  $M = 11^{23} \bmod 187 = 88$

Esponenziazione modulare

- Algoritmo “Square and Multiply”
- Permette di effettuare l’esponenziazione in modo veloce ed efficiente
- La base della potenza viene ripetutamente elevata al quadrato
- Vengono poi moltiplicati insieme solo i valori necessari alla formazione del risultato finale
- Si utilizza la rappresentazione binaria dell’esponente
- La complessità è $O(\log_2 n)$
  - eg. $7^5 = 7^4.7^1 = 3.7 = 10 \bmod 11$
  - eg. $3^{129} = 3^{128}.3^1 = 5.3 = 4 \bmod 11$

RSA: Generazione delle chiavi

- Per usare RSA si deve:
  - Determinare in modo aleatorio 2 numeri primi `p` e `q`
  - selezionare `e` o `d` e calcolare l'altro
- I primi `p,q` non devono essere facilmente derivabili dal prodotto `n=p.q`
  - Devono quindi essere sufficientemente elevati
  - Si usa il test di primalità di Miller-Rabin
- Tipicamente `e` si sceglie piccolo, in modo che `d` sia grande

Sicurezza di RSA

- Vi sono 3 strategie di attacco a RSA:
  - Ricerca a forza bruta (non è fattibile perchè la chiave è lunga)
  - Attacchi matematici (cercano di fattorizzare n, o di calcolare $\Phi(n)$)
  - Attacchi a tempo (carpiscono informazioni dai tempi impiegati per la decriptazione)

Attacco matematico

- L’attacco matematico si svolge in 3 modi:
  - Si fattorizza $n=pq$, e quindi si trova $\Phi(n)$ e poi d
  - Si trova $\Phi(n)$ direttamente e quindi d
  - Si cerca direttamente d
- Si ritiene che la difficoltà di tali attacchi sia la stessa
- RSA è sicuro se n è lungo circa 1000 bit

Attacchi a tempo

- Si sono sviluppati dalla metà degli anni ‘90
- Misura i tempi di esecuzione delle esponenziazioni
- Possibili contromisure:
  - Utilizzo di un tempo di esponenziazione costante
  - Introduzione di ritardi aleatori
  - Inserimento di valori spuri nei calcoli