Capitolo 9

Firma digitale

Per anni, si sono usati svariati tipi di firma per associare l'identità delle persone ai documenti. Nel Medioevo, i nobili sigillavano i documenti con uno stampo di cera riportante il loro stemma e si supponeva che il nobile fosse l'unico in grado di riprodurre lo stemma. Nelle transazioni moderne gli scontrini delle carte di credito sono firmati e si suppone che il commerciante ne verifichi la firma confrontandola con quella riportata sulla carta. Con lo sviluppo del commercio elettronico e dei documenti elettronici, questi metodi non sono più sufficienti.

Supponiamo, per esempio, di voler firmare un documento elettronico. Perché non potremmo semplicemente digitalizzare la nostra firma e aggiungerla al documento? Perché chiunque avesse accesso al documento potrebbe semplicemente rimuovere la firma e aggiungerla a un altro documento, per esempio un assegno. Con la firma classica, questa operazione richiederebbe di ritagliare la firma dal documento originale, oppure fotocopiarla, e incollarla sull'assegno. Difficilmente questa firma sarebbe ritenuta valida. Invece, una falsificazione elettronica è abbastanza facile e non può essere distinta dall'originale.

Perciò, abbiamo bisogno di firme digitali che non possono essere separate da un messaggio e incollate altrove. In altre parole, la firma non è solo legata al firmatario, ma anche al messaggio da firmare. Inoltre, la firma digitale deve essere facilmente verificabile da altri soggetti. Pertanto, gli schemi di firma digitale sono composti da due passi distinti: il processo di firma ed il processo di verifica.

Di seguito, vedremo due schemi di firma e discuteremo gli attacchi chiamati "attacchi del compleanno". In generale, non abbiamo intenzione di cifrare il messaggio m, che potrebbe essere un documento legale e pubblico. Se fosse necessario, un messaggio può essere cifrato dopo la firma (come, per esempio, in PGP, vedi Paragrafo 10.6).

9.1 Firma RSA

Per firmare un documento m, Alice compie le seguenti operazioni.

1. Alice genera due numeri primi grandi p, q e calcola $n = pq$. Sceglie e_A, in modo che $1 < e_A < \varphi(n)$ con $\text{MCD}(e_A,\varphi(n)) = 1$ e calcola d_A in modo che $e_A d_A \equiv 1 \pmod{\varphi(n)}$. Alice pubblica (e_A,n) e tiene d_A, p, q privati.

2. La firma di Alice è:
$$y \equiv m^{d_A} \pmod{n}.$$

3. La coppia (m, y) è resa pubblica.

Bob può verificare che il messaggio sia stato veramente firmato da Alice compiendo le seguenti operazioni.

1. Ottiene i parametri (e_A,n) di Alice.

2. Calcola $z \equiv y^{e_A} \pmod{n}$. Se $z = m$, Bob considera valida la firma; altrimenti la firma non è valida.

Supponiamo che Eva voglia incollare la firma di Alice a un altro messaggio m_1. Eva non può semplicemente usare la coppia (m_1, y), perché $y^{e_A} \not\equiv m_1 \pmod{n}$, ma ha bisogno un y_1 tale che $y_1^{e_A} \equiv m_1 \pmod{n}$. Questo è lo stesso problema della decifrazione del messaggio m_1 cifrato con l'algoritmo RSA in modo da ottenere il "testo in chiaro" y_1; ed è un problema che si ritiene difficile.

Un'altra possibilità è che Eva scelga prima y_1 e, di conseguenza, il messaggio $m_1 \equiv y_1^{e_A} \pmod{n}$. Con questo schema di firma, Alice non può negare di avere firmato m_1. Tuttavia è assai improbabile che m_1 sia un messaggio di senso compiuto. Probabilmente sarà una sequenza di caratteri casuali piuttosto che un messaggio che impegni Alice a versare a Eva un milione di euro. Pertanto crederemo ad Alice quando affermerà che il messaggio in questione è falso.

Esiste una variante di questa procedura che permette ad Alice di firmare un messaggio senza conoscerne il contenuto. Supponiamo che Bob abbia fatto una scoperta sensazionale. Bob vuole registrare pubblicamente ciò che ha fatto (in modo da avere la priorità quando saranno assegnati i premi Nobel), ma non vuole che altri sappiano i dettagli della sua invenzione (in modo da sfruttarla economicamente). Bob e Alice eseguono la procedura seguente. Il messaggio da firmare è m.

1. Alice sceglie un modulo RSA n ($n = pq$, prodotto di due numeri primi grandi), un esponente di cifratura e e un esponente di decifrazione d. Alice pubblica n ed e e tiene privati p, q, d. In pratica, può anche cancellarli dal suo computer alla fine della procedura di firma.

2. Bob sceglie un numero intero casuale $k \pmod{n}$ con $\text{MCD}(k,n) = 1$ e calcola $t \equiv k^e m \pmod{n}$, che manda ad Alice.

3. Alice firma t calcolando $s \equiv t^d \pmod{n}$, che rispedisce a Bob.

4. Bob calcola $s/k \pmod{n}$, che è il messaggio firmato m^d.

Per provare che s/k è il messaggio firmato, osserviamo che $k^{ed} \equiv (k^e)^d \equiv k \pmod{n}$, perché si tratta semplicemente della cifratura e poi decifrazione di k nello schema RSA. Pertanto,
$$s/k \equiv t^d/k \equiv k^{ed} m^d/k \equiv m^d \pmod{n},$$
che è il messaggio firmato.
La scelta di k è casuale, quindi $k^e \pmod{n}$ è la cifratura RSA di un numero casuale, quindi è esso stesso un numero casuale e non dà alcuna informazione sul valore di m (tuttavia non nasconderebbe un messaggio come $m = 0$). In questo modo Alice non sa nulla del messaggio che sta firmando.
Una volta che la procedura di firma è conclusa, Bob ha lo stesso messaggio firmato che avrebbe ottenuto con la procedura standard di firma.
Ci sono parecchi pericoli potenziali con questo protocollo. Per esempio, Bob potrebbe far firmare ad Alice una promessa di pagargli un milione di euro. Non discuteremo le salvaguardie necessarie per evitare questi problemi.
Gli schemi come questi, chiamati di **firma cieca** (*blind signature*), sono stati sviluppati da David Chaum, che detiene numerosi brevetti in merito.

9.2 Schema di firma di ElGamal

Il metodo di cifratura di ElGamal, descritto nel Paragrafo 7.5, può essere modificato in uno schema di firma. Una caratteristica che differenzia RSA dal metodo di ElGamal è che in quest'ultimo ci sono differenti firme che sono tutte ugualmente valide per un dato messaggio.
Supponiamo che Alice voglia firmare un messaggio. Per prima cosa sceglie un numero primo p grande e una radice primitiva α. Successivamente, Alice sceglie un numero intero segreto a tale che $1 \le a \le p-2$ e calcola $\beta \equiv \alpha^a \pmod{p}$. I valori di p, α, and β sono resi pubblici. La sicurezza del sistema risiederà nella segretezza del numero a. È difficile per un avversario determinare a conoscendo (p,α,β) perché il logaritmo discreto è considerato un problema difficile.
Alice firma un messagio m in questo modo.

1. Sceglie un numero segreto casuale k tale che $\text{MCD}(k,p-1) = 1$.

2. Calcola $r \equiv \alpha^k \pmod{p}$ con $0 < r < p$.

3. Calcola $s \equiv k^{-1}(m - ar) \pmod{p-1}$.

Il messaggio firmato è la terna (m, r, s).
Bob verifica la firma in questo modo:

1. Scarica la chiave pubblica di Alice (p,α,β).

2. Calcola $v_1 \equiv \beta^r r^s \pmod{p}$ e $v_2 \equiv \alpha^m \pmod{p}$.

3. Considera la firma valida se e solo se $v_1 \equiv v_2 \pmod{p}$.

Per mostrare che la procedura di verifica è corretta, supponiamo che la firma sia valida. Poiché $s \equiv k^{-1}(m - ar) \pmod{p-1}$, abbiamo $sk \equiv m - ar \pmod{p-1}$, quindi $m \equiv sk + ar \pmod{p-1}$. Pertanto (ricordando che una congruenza mod $p-1$ negli esponenti diventa complessivamente una congruenza mod p),

$$v_2 \equiv \alpha^m \equiv \alpha^{sk+ar} \equiv (\alpha^a)^r(\alpha^k)^s \equiv \beta^r r^s \equiv v_1 \pmod{p}.$$

Se Eva dovesse scoprire il valore di a potrebbe eseguire l'algoritmo di firma e produrre la firma di Alice su qualunque documento. Pertanto è importante che a rimanga segreto.

- Se Eva ha un altro messaggio m, ma non conosce a, non può calcolare il corrispondente s. Supponiamo che Eva cerchi di aggirare questo passaggio scegliendo un s che soddisfi l'equazione di verifica:

$$\beta^r r^s \equiv \alpha^m \pmod{p}.$$

Riarrangiando i termini possiamo scrivere $r^s \equiv \beta^{-r}\alpha^m \pmod{p}$, che è un problema di logaritmo discreto; pertanto trovare un s appropriato è difficile. Se invece si sceglie prima s, l'equazione per calcolare r assomiglia a un problema di logaritmo discreto e, generalmente, si suppone che sia difficile da risolvere. Non è noto se ci sia un modo per trovare simultaneamente la coppia r e s, ma sembra improbabile. In definitiva, si ritiene che lo schema di firma sia sicuro finché i logaritmi discreti mod p sono difficili da calcolare (per esempio se $p-1$ non è un prodotto di numeri primi piccoli; vedi il Paragrafo 7.2).

Se Alice vuole firmare un secondo documento deve scegliere un nuovo numero casuale k. Se, invece, sceglie lo stesso k per firmare i messaggi m_1 e m_2, allora in entrambe le firme ci sarà il medesimo r, ed Eva saprà che k è stato usato due volte. I valori di s saranno invece diversi, chiamiamoli s_1 e s_2. Eva sa che

$$s_1 k - m_1 \equiv -ar \equiv s_2 k - m_2 \pmod{p-1}.$$

Quindi

$$(s_1 - s_2)k \equiv m_1 - m_2 \pmod{p-1}.$$

Sia $d = \text{MCD}(s_1 - s_2, p-1)$. La congruenza ha d soluzioni, che possono essere trovate usando la procedura descritta nel Paragrafo 3.3. Di solito d è piccolo e, di conseguenza, i valori che può assumere k sono pochi. Eva calcola α^k per tutti i possibili valori di k finché non trova r. Conoscendo k, Eva risolve

$$ar \equiv m_1 - ks_1 \pmod{p-1}$$

in a. Ci sono $\text{MCD}(r, p-1)$ soluzioni. Eva calcola α^a per ognuna delle soluzioni finché non trova β. A questo punto Eva conosce a, ha completamente violato il sistema e può riprodurre le firme di Alice a suo piacimento.

Esempio. Alice vuole firmare il messaggio $m_1 = 151405$ (che corrisponde alla parola *"one"*, se assumiamo $01 = a, 02 = b, \ldots$) e sceglie $p = 225119$. Una radice primitiva è $\alpha = 11$. Usando il numero segreto a, calcola $\beta \equiv \alpha^a \equiv 18191 \pmod{p}$. Per firmare il messaggio sceglie un numero casuale k, che tiene segreto. Alice calcola $r \equiv \alpha^k \equiv 164130 \pmod{p}$ e poi

$$s_1 \equiv k^{-1}(m_1 - ar) \equiv 130777 \pmod{p-1}.$$

Il messaggio firmato è la terna $(151405, 164130, 130777)$.

Supponiamo che Alice firmi anche il messaggio $m_2 = 202315$ (che corrisponde alla parola *"two"*) ottenendo il messaggio firmato $(202315, 164130, 164899)$. Eva riconosce immediatamente che Alice ha usato lo stesso valore di k, perché il valore di r è uguale nelle due firme. Pertanto scrive la congruenza

$$-34122k \equiv (s_1 - s_2)k \equiv m_1 - m_2 \equiv -50910 \pmod{p-1}.$$

Poiché $\text{MCD}(-34122, p-1) = 2$, ci sono due soluzioni, che possono essere trovare usando il metodo descritto nel Paragrafo 3.3. Dividendo la congruenza per 2 si ottiene:

$$-17061k \equiv -25455 \pmod{(p-1)/2},$$

che ha soluzione $k \equiv 239 \pmod{(p-1)/2}$. In questo modo ci sono due valori di $k \pmod{p}$, e precisamente 239 e $239 + (p-1)/2 = 112798$. Calcolando

$$\alpha^{239} \equiv 164130, \quad \alpha^{112798} \equiv 59924 \pmod{p},$$

Eva osserva che il primo valore di k dà il corretto valore di r e conclude che $k = 239$. A questo punto riscrive $s_1 k \equiv m_1 - ar \pmod{p-1}$ e ottiene

$$164130a \equiv ra \equiv m_1 - s_1 k \equiv 187104 \pmod{p-1}.$$

Poiché $\text{MCD}(164130, p-1) = 2$, ci sono due soluzioni, vale a dire $a = 28862$ e $a = 141421$. Eva calcola

$$\alpha^{28862} \equiv 206928, \quad \alpha^{141421} \equiv 18191 \pmod{p}.$$

Poiché il secondo valore è β, Eva ha trovato $a = 141421$.

Adesso che conosce a, Eva può falsificare la firma di Alice su qualunque documento. ■

Lo schema di firma di ElGamal è un esempio di **firma con appendice**. Il messaggio m non può essere recuperato facilmente a partire dalla firma (r,s) e deve essere incluso nella procedura di verifica. Lo schema di firma RSA, per contro, è uno **schema di recupero del messaggio**. In RSA, infatti, il messaggio può essere ottenuto a partire dalla firma y. Pertanto è sufficiente inviare y perché chiunque possa dedurre m calcolando $y^{e_A} \pmod{n}$. È improbabile che un y casuale dia un messaggio dotato di significato, per cui si può trascurare il pericolo che qualcuno sostituisca un messaggio valido con un messaggio falso sostituendo y.

9.3 Hash e firma

Nei due schemi appena presentati, la firma può essere più lunga del messaggio. Quando il messaggio è lungo, questa proprietà è svantaggiosa, per cui si preferisce applicare lo schema di firma un hash del messaggio, piuttosto che al messaggio stesso.

Sia h una funzione hash pubblica. Dato un messaggio m, Alice calcola lo hash $h(m)$, che è significativamente più piccolo e può essere firmato molto più velocemente dell'intero messaggio, e usa la firma dello hash, $\text{sig}(h(m))$ come firma dell'intero messaggio. La coppia $(m,\text{sig}(h(m)))$ trasmette la stessa informazione dello schema originale, ma ha due vantaggi: è più veloce da creare (data la ragionevole supposizione che l'operazione hash sia veloce) e richiede meno risorse per la trasmissione o memorizzazione.
Ma questo metodo è sicuro? Supponiamo che Eva possieda il messaggio $(m, \text{sig}(h(m))$, firmato da Alice. Per aggiungere la firma di Alice a un altro messaggio m' Eva ha bisogno che $\text{sig}(h(m')) = \text{sig}(h(m))$; in particolare ha bisogno che $h(m') = h(m)$. Se la funzione hash è unidirezionale, Eva avrà difficoltà a trovare un qualunque m' che soddisfi l'equazione e le probabilità che il messaggio m' in suo possesso vada bene sono molto piccole. Inoltre, poiché richiediamo che le nostre funzioni hash siamo fortemente resistenti alle collisioni, è improbabile che Eva possa trovare due messaggi $m_1 \neq m_2$ che ammettono la stessa firma. Se riuscisse, potrebbe far firmare da Alice il messaggio m_1 e trasferire la firma a m_2, ma Alice si insospettirebbe comunque perché è estremamente probabile che m_1 (e m_2) siano messaggi senza senso.
Nel prossimo paragrafo vedremo come Eva può ingannare Alice se la lunghezza dello hash è troppo corta (e vedremo che la funzione di hash non sarà neanche fortemente resistente alle collisioni).

9.4 Attacchi del compleanno alle firme digitali

Alice sta per firmare elettronicamente un documento usando uno schema nel quale si firma un hash del documento. Supponiamo che l'output della funzione hash sia di 50 bit. Alice teme che Fred possa ingannarla e farle firmare anche un contratto aggiuntivo, magari di acquisto di una palude in Florida, tuttavia si sente sicura perché la probabilità che un contratto fraudolento abbia lo stesso hash del documento che sta per firmare sono una su 2^{50}, che è approssimativamente una su 10^{15}. Fred può inventare diverse formulazioni del contratto fraudolento, ma è difficile che ne trovi una con il valore hash corretto. Tuttavia Fred ha studiato il problema del compleanno e può agire in questo modo. Per prima cosa trova nel documento originale 30 punti dove può introdurre leggere modifiche: aggiungere uno spazio alla fine di una riga, riformulare una frase e così via. In ognuno di questi punti, Fred ha due possibilità: tenere la versione originale oppure effettuare la modifica. Perciò Fred può formulare 2^{30} documenti essenzialmente identici al documento originale e contro i quali Alice non avrebbe obiezioni. A questo punto, Fred calcola lo hash di ciascuno di questi 2^{30} documenti e li conserva. Allo stesso modo, Fred formula 2^{30} versioni del contratto fraudolento e ne conserva i corrispondenti hash. Se consideriamo il problema del compleanno generalizzato con $r = 2^{30}$ e $n = 2^{50}$, abbiamo $r = \sqrt{\lambda n}$ con $\lambda = 2^{10} = 1024$. Pertanto, con probabilità circa $1 - e^{-1024} \approx 1$ una versione del documento corretto ha lo stesso hash di una versione del contratto fraudolento. Fred trova questa corrispondenza e chiede ad Alice di firmare la versione buona, con l'intenzione di apporre la firma di Alice anche al contratto fraudolento. Poiché i due documenti hanno lo stesso hash, la firma è valida per entrambi e Fred può sostenere che Alice ha accettato di acquistare la palude. Al momento della firma, Alice, che insegna Italiano, insiste nel rimuovere una virgola da una frase e firma il documento modificato, che ha un hash completamente diverso dal documento redatto da Fred. In questo modo l'attacco è sventato, Fred deve trovare una versione del documento fraudolento che abbia lo stesso hash del documento firmato da Alice, il che è essenzialmente impossibile.
L'attacco che Fred ha cercato di condurre è chiamato "attacco del compleanno". La sua conseguenza pratica è che, a pari sicurezza, è necessario usare funzioni hash con un output di lunghezza almeno doppia rispetto a quella che si ritiene necessaria, perché l'effetto dell'attacco del compleanno è il dimezzamento del numero di bit. Il modo migliore per sventare un attacco di compleanno è fare come Alice: cambiare leggermente un documento elettronico prima di firmarlo.

9.5 Algoritmo DSA

Il NIST (National Institute of Standards and Technology) propose l'algoritmo DSA (Digital Signature Algorithm, algoritmo di firma digitale) nel 1991 e lo adottò come standard nel 1994. Così come il metodo di ElGamal, l'algoritmo DSA è uno schema di firma con appendice. Inoltre, come in altri schemi, la firma è applicata a un digest del messaggio. Nel caso specifico di DSA, la funzione hash genera un output di 160 bit che, nel seguito, indicheremo con m.
La fase di inizializzazione genera le chiavi da usare in DSA.

1. Alice trova un numero primo q lungo 160 bit e sceglie un numero primo p che soddisfi la relazione $q|p-1$ (vedi Esercizio 7). Il problema del logaritmo discreto deve essere difficile da risolvere per questo valore di p. (Nella prima versione p era di 512 bit. Le versioni successive permettono numeri più lunghi.)
2. Sia g una radice primitiva mod p e sia $\alpha \equiv g^{(p-1)/q} \pmod{p}$. Allora $\alpha^q \equiv 1 \pmod{p}$.
3. Alice sceglie un numero segreto a tale che $1 \leq a < q-1$ e calcola $\beta \equiv \alpha^a \pmod{p}$.
4. Alice pubblica (p,q,α,β) e tiene a segreto.

Alice firma un messaggio m eseguendo le seguenti operazioni.

1. Sorteggia un numero intero casuale segreto k tale che $0 < k < q-1$.
2. Calcola $r = (\alpha^k \pmod{p}) \pmod{q}$.
3. Calcola $s \equiv k^{-1}(m + ar) \pmod{q}$.
4. Ottiene la firma (r,s), che invia a Bob insieme al messaggio m.

Per verificare la firma, Bob deve svolgere le seguenti operazioni.

1. Scaricare la chiave pubblica di Alice (p,q,α,β).
2. Calcolare $u_1 \equiv s^{-1}m \pmod{q}$ e $u_2 \equiv s^{-1}r \pmod{q}$.
3. Calcolare $v = (\alpha^{u_1}\beta^{u_2} \pmod{p}) \pmod{q}$.

4. Accettare la firma se e solo se $v = r$.

La procedura di verifica funziona perché, per definizione di s, si ha

$$m \equiv (-ar + ks) \pmod{q},$$

che implica

$$s^{-1}m \equiv (-ars^{-1} + k) \pmod{q}.$$

Perciò

$$\begin{aligned} k &\equiv s^{-1}m + ars^{-1} \pmod{q} \\ &\equiv u_1 + au_2 \pmod{q}. \end{aligned}$$

Quindi $\alpha^k = \alpha^{u_1+au_2} = (\alpha^{u_1}\beta^{u_2} \pmod{p}) \pmod{q}$ da cui $v = r$.
Come nello schema di ElGamal, l'intero a deve essere tenuto segreto. Chiunque conosca a può firmare qualunque documento. Inoltre, se Alice usasse due volte lo stesso valore di k, sarebbe possibile trovare a seguendo la procedura già vista per ElGamal.
Diversamente dallo schema di ElGamal, conoscere r non permette di risalire al valore k, ma solo a $k \mod q$. Ci sono approssimativamente $2^{512-160} = 2^{342}$ numeri mod p che si riducono allo stesso valore mod q.
Qual è il vantaggio di usare $\alpha^q \equiv 1 \pmod{p}$ anziché una radice primitiva? L'algoritmo di Pohlig-Hellman per estrarre il logaritmo discreto trova facilmente informazioni modulo i fattori primi piccoli di $p-1$, mentre è inutile per i fattori grandi, come q.
Nello schema di ElGamal, un intruso potrebbe calcolare facilmente $a \pmod{2^t}$, dove 2^t è la più grande potenza di 2 che divide $p-1$. L'intruso sarebbe ancora lontano dal conoscere il valore di a, ma avrebbe comunque a disposizione un'informazione utile. L'algoritmo DSA evita questo problema eliminando da a tutte le informazioni tranne quella mod q.
Nello schema di ElGamal la verifica richiede tre elevamenti a potenza modulari. Poiché l'elevamento a potenza è l'operazione più lenta nel calcolo, la verifica DSA è modificata in modo da richiederne solo due in modo da ridurre i tempi di verifica nel caso in cui si debbano verificare molte firme in breve tempo.

9.6 Esercizi

1. Supponendo che MCD(r,$p-1$) sia piccolo, mostrare che, chi scopre il valore di k usato in una firma di ElGamal, può determinare anche a.

2. Si supponga che (m,r,s) sia un messaggio firmato usando lo schema di ElGamal. Si scelga h con MCD(h,$p-1$) $= 1$ e si ponga $r_1 \equiv r^h \pmod{p}$. Sia $s_1 \equiv sr_1h^{-1}r^{-1} \pmod{p-1}$.

   (a) Trovare un messaggio m_1 tale che (m_1, r_1, s_1) sia una firma valida.

   (b) Questo metodo permette a Eva di falsificare una firma per il messaggio m_1. Perché è improbabile che questo sia un problema?

3. Siano $p = 11$, $q = 5$, $\alpha = 3$ e $k = 3$. Mostrare che

$$(\alpha^k \pmod{p}) \pmod{q} \neq (\alpha^k \pmod{q}) \pmod{p}.$$

   Questo spiega perché l'ordine delle operazioni è importante in DSA.

4. Si possono ottenere molte varianti allo schema di firma di ElGamal cambiando l'equazione di firma $s \equiv k^{-1}(m - ar) \pmod{p-1}$. Eccone alcune.

   (a) Si consideri l'equazione di firma $s \equiv a^{-1}(m - kr) \pmod{p-1}$. Mostrare che l'equazione $\alpha^m \equiv (\alpha^a)^s r^r \pmod{p}$ permette di verificare la firma.

   (b) Si consideri l'equazione di firma $s \equiv am + kr \pmod{p-1}$. Mostrare che l'equazione $\alpha^s \equiv (\alpha^a)^m r^r \pmod{p}$ permette di verificare la firma.

   (c) Si consideri l'equazione di firma $s \equiv ar + km \pmod{p-1}$. Mostrare che l'equazione $\alpha^s = (\alpha^a)^r r^m \pmod{p}$ permette di verificare la firma.

5. Lo schema di ElGamal si presta al seguente attacco, noto come falsificazione esistenziale (*existential forgery*). Si scelgano u,v tali che MCD(v,$p-1$) $= 1$. Si calcolino $r \equiv \beta^v\alpha^u \pmod{p}$ e $s \equiv -rv^{-1} \pmod{p-1}$.

   (a) Dimostrare che la coppia (r,s) è una firma valida per il messaggio $m = su \pmod{p-1}$ (anche se è improbabile che il messaggio m così ottenuto abbia significato).

   (b) Si supponga di usare una funzione hash h e che la firma debba essere valida per $h(m)$ anziché per m (quindi deve essere $h(m) = su$). Spiegare come questo schema protegga contro la falsificazione esistenziale, ovvero spiegare perché, usando la procedura appena descritta, sia difficile generare un messaggio falso ma con una firma valida.

6. Alice vuole firmare un documento usando lo schema di firma di ElGamal. Si supponga che non sia disponibile un generatore di numeri casuali e che Alice usi $k = a$. Come può Eva scoprire questo fatto e come può determinare i valori di k e a e violare il sistema?

7. **(a)** In molti protocolli crittografici, è necessario scegliere un numero primo p tale che anche $q = (p-1)/2$ sia un numero primo. Un modo può essere scegliere un numero primo q a caso ed effettuare un test di primalità per $2q+1$. Si supponga che q sia stato scelto con 100 cifre decimali. Si supponga inoltre che $2q+1$ sia un numero casuale intero dispari di 100 cifre. (Questa affermazione non è perfettamente accurata perché, per esempio, $2q+1$ non può essere congruente a 1 mod 3; tuttavia è adeguata come stima di massima). Mostrare che la probabilità che $2q+1$ sia primo è approssimativamente 1/115 (usare il teorema dei numeri primi, Paragrafo 6.3). Pertanto, si dovrebbe trovare un numero primo p adeguato dopo avere provato, in media, 115 possibili numeri primi q.

   (b) In una versione dell'algoritmo DSA, Alice ha bisogno di un numero primo q di 160 bit e di un numero primo p di 512 bit, tale che $q|p-1$. Si supponga

che Alice scelga un numero primo q casuale di 160 bit e un numero k casuale pari di 352 bit tale che $qk+1$ sia lungo 512 bit. Mostrare che la probabilità che $qk+1$ sia primo è, approssimativamente, $1/177$. Quindi Alice può trovare abbastanza rapidamente i numeri p e q cercati.

8. Si consideri la seguente variante dello schema di firma di ElGamal. Alice sceglie un numero primo grande p e una radice primitiva α. Alice sceglie anche una funzione $f(x)$ che, dato un intero x con $0 \le x < p$, restituisca un intero $f(x)$ con $0 \le f(x) < p-1$. (Per esempio, $f(x) = x^7 - 3x + 2 \pmod{p-1}$ per $0 \le x < p$.) Alice sceglie un numero intero segreto a e calcola $\beta \equiv \alpha^a \pmod{p}$. I numeri p, α, β e la funzione $f(x)$ vengono resi pubblici.

Per firmare un messaggio m, Alice esegue le seguenti operazioni.

1. Sceglie un numero intero casuale k con $\text{MCD}(k, p-1) = 1$.
2. Calcola $r \equiv \alpha^k \pmod{p}$.
3. Calcola $s \equiv k^{-1}(m - f(r)a) \pmod{p-1}$.

Il messaggio firmato è (m, r, s).

Per verificare la firma, Bob esegue le seguenti operazioni.

1. Calcola $v_1 \equiv \beta^{f(r)} r^s \pmod{p}$
2. Calcola $v_2 \equiv \alpha^m \pmod{p}$
3. Se $v_1 \equiv v_2 \pmod{p}$, la firma è valida.

(a) Mostrare che l'equazione di verifica è corretta.

(b) Si supponga che Alice scelga una funzione costante tale che $f(x) = 0$ per ogni x. Mostrare che Eva può falsificare la firma per qualunque messaggio m_1. Dato un messaggio m_1, dare una combinazione di k, r, s che fornisca una firma valida.

9.7 Problemi al calcolatore

1. Si supponga di usare uno schema di firma di ElGamal con $p = 65539$, $\alpha = 2$, $\beta = 33384$. Si inviano i due messaggi (m, r, s):

$$(809, 18357, 1042)\ (= \text{“hi”}) \quad \text{e} \quad (22505, 18357, 26272)\ (= \text{“bye”}).$$

(a) Mostrare che si è usato lo stesso valore di k in entrambe le firme.

(b) Sfruttando questo fatto, trovare k e trovare un valore di a tale che $\beta \equiv \alpha^a \pmod{p}$.

2. Alice e Bob usano i seguenti parametri RSA:[1]

$$n_A = 1710247041836161097008180669251978415166712 77,$$

[1] Prestare attenzione che i numeri in questo problema sono molto grandi. Per svolgere l'esercizio in MATLAB® occorre usare il motore matematico di Maple®.

$$n_B = 8390735427343693592608713559390626227476331 09,$$

$$e_A = 1571, \qquad e_B = 87697.$$

Bob sa che

$$p_B = 9876345769783456893461 3, \quad q_B = 8495789457893457345793$$

(dove $n_B = p_B q_B$). Alice firma un documento e lo invia a Bob con la **firma** (m,s) (dove $s \equiv m^{d_A} \pmod{n_A}$). Per mantenere la segretezza del documento, lo cifra usando la chiave pubblica di Bob. Bob riceve la firma cifrata $(m_1,s_1) \equiv (m^{e_B}, s^{e_B}) \pmod{n_B}$, dove

$$m_1 = 4187265539970942585779800550613051509405479 56$$

$$s_1 = 7491426496415481015201336347368657528832772 37.$$

Trovare il messaggio m e verificare che proviene da Alice. (La firma è memorizzata nelle variabili `sigpairm1, sigpairs1`. I numeri n_A, n_B, p_B, q_B sono memorizzati nelle variabili `signa, signb, sigpb, sigqb`.)

3. Nel problema precedente,[2] si supponga che Bob abbia scelto i numeri primi $p_B = 7865712896579$ e $q_B = 849578945789345734 5793$. Supponendo di usare gli stessi esponenti di cifratura, spiegare perché, quando Alice gli manda la coppia (m_2, s_2) con

$$m_2 = 1482376523249871234451241871713 0930,$$

$$s_2 = 4317612162846544134011241867206 5063,$$

Bob non riesce a verificarne la firma. Che modifica occorre fare per far funzionare la procedura? (La firma è memorizzata nelle variabili `sigpairm2, sigpairs2`.)

[2] Per svolgere questo esercizio in MATLAB® occorre usare il motore matematico di Maple®.