Sicurezza delle Reti

Prof. Stefano Bregni — **V Appello d'Esame 2021-22 – 13 febbraio 2023**

Cognome e nome: *(stampatello)* *(firma leggibile)*

Matricola:

NB: In ogni esercizio, ogni risposta non giustificata adeguatamente, anche con pochissime parole, avrà valore nullo.

Domanda 1

(svolgere su questo foglio nello spazio assegnato) (7 punti)

Bob adotta il *sistema di firma elettronica di El Gamal* e pubblica $p = 113$, $\alpha = 2$, $\beta = \alpha^a \bmod p$, tenendo segreto l'esponente $a = 34$.

a) Verificare la correttezza dei dati forniti, in base alle ipotesi del metodo di El Gamal. Se $\alpha = 2$ non risultasse una scelta valida, Bob userà invece un valore valido scelto nell'insieme $\alpha = \{3, 4\}$. Se nessuna di queste scelte risultasse valida, Bob rinuncerà a proseguire (e l'esercizio termina qui). Calcolare β.

b) Bob estrae il numero casuale segreto (*nonce*) $k = 13$. Per questo valore di k, calcolare la firma di Bob $A = (r, s)$ del messaggio $P = 30$.

c) Verificare se anche la firma $A' = (r', s') = (47, 0)$ è valida da Bob per lo stesso messaggio $P = 30$.

d) Se è valida, calcolare il valore di k per cui è stata calcolata da Bob, scegliendo il metodo più veloce a disposizione.

a) p primo $1 < a \leq p-2$ $K \perp p-1$ $p-1 = 112 = 2^4 \cdot 7$

Test se α elem. prim. di $\mathbb{Z}_p^*$:
$\alpha^{\frac{p-1}{q_i}} \not\equiv 1 \pmod p$

$3^{56} \equiv -1$
$3^{16} \equiv 49$
$\Rightarrow \alpha = 3$ ($\alpha = 2, 4$ no) ok

$\beta = \alpha^a \bmod p = 3^{34} \bmod 113 = 26$

b) $r = \alpha^k \bmod p = 3^{13} \bmod 113 = 6$

$s = k^{-1}(P - ar) \bmod (p-1) = 69(30 - 34 \cdot 6) \bmod 112 = 90$

$k^{-1} = 34^{-1} \equiv 69 \pmod{112}$ $\Rightarrow$ $A = (6, 90)$

c) $\beta^r \cdot r^s \equiv \alpha^P \pmod p$

$26^{47} \cdot 47^0 \equiv 91 \pmod{113}$
$3^{30} \equiv 91 \pmod{113}$
$\Rightarrow A = (47, 0)$ firma valida di $P = 30$

d) $3^k \equiv 47 \pmod{113}$ $\Rightarrow$ $K = 31$ con BSGS

Sicurezza delle Reti
Prof. Stefano Bregni — **V Appello d'Esame 2021-22 – 13 febbraio 2023**

Cognome e nome: *(stampatello)* *(firma leggibile)*

Matricola:

Domanda 2
(svolgere su questo foglio nello spazio assegnato) (7 punti)

Bob adotta il *sistema di cifratura a chiave pubblica di El Gamal* e pubblica $p = 257$, $\alpha = 6$, $\beta = \alpha^a \bmod p$, tenendo segreto l'esponente $a = 71$.

a) Verificare la correttezza dei dati forniti, in base alle ipotesi del metodo di El Gamal. Se $\alpha = 6$ non risultasse una scelta valida, Bob userà invece un valore valido scelto nell'insieme $\alpha = \{8, 9\}$. Se nessuna di queste scelte risultasse valida, Bob rinuncerà a proseguire (e l'esercizio termina qui). Calcolare β.

b) Alice estrae il numero casuale segreto (*nonce*) $k = 13$ e spedisce il messaggio $P_1 = 60$. Calcolare il messaggio cifrato $C_1 = (r_1, t_1)$.

c) Alice estrae un nuovo numero casuale segreto (*nonce*) k e, usando sempre questo stesso valore, spedisce i messaggi P_2, P_3, P_4. Oscar intercetta i messaggi cifrati $C_2 = (r_2, t_2) = (34, 29)$, $C_3 = (r_3, t_3) = (34, 2)$, $C_4 = (r_4, t_4) = (34, 3)$ e, per altra via, viene a sapere che $P_2 = 99$. Calcolare P_3 e P_4.

d) Calcolare per quale valore di k Alice ha calcolato i messaggi C_2, C_3, C_4 del punto c), applicando l'algoritmo Baby Step Giant Step.

a) p primo $1 < a < p-2$ $p-1 = 256 = 2^8$ Tutte α elem. prim. di $\mathbb{Z}_p^*$

$6^{128} \equiv -1 \pmod{257}$ $\quad \alpha^{(p-1)/q_i} \not\equiv 1 \pmod p$

$\Rightarrow \alpha = 6$ (OK) $(\alpha = 8, 9$ NO$)$ $\quad \beta = \alpha^a \bmod p = 6^{71} \bmod 257 = 20$

b) $r_1 = \alpha^k \bmod p = 6^{13} \bmod 257 = 79$

$t_1 = \beta^k P_1 \bmod p = 20^{13} \cdot 60 \bmod 257 = 82 \Rightarrow \boxed{C_1 = (79, 82)}$

c) $\frac{t_2}{P_2} \equiv \frac{t_3}{P_3} \equiv \frac{t_4}{P_4} \equiv \beta^k \pmod p$ $\quad t_2^{-1} = 29^{-1} \equiv 195 \pmod{257}$

$P_3 \equiv P_2 \frac{t_3}{t_2} \pmod p = 99 \cdot 2 \cdot 195 \equiv \boxed{60} \pmod{257}$

$P_4 \equiv P_2 \frac{t_4}{t_2} \pmod p = 99 \cdot 3 \cdot 195 \equiv \boxed{97} \pmod{257}$

d) $6^k \equiv 24 \pmod{257} \rightarrow \boxed{k = 40}$ (BSGS)

Sicurezza delle Reti
Prof. Stefano Bregni — **V Appello d'Esame 2021-22 – 13 febbraio 2023**

Cognome e nome: *(stampatello)*
(firma leggibile)

Matricola:

Domanda 3

(svolgere su questo foglio nello spazio assegnato) (7 punti)

a) Perché una *funzione di hash* $y = y(x)$ non può mai essere *invertibile*?

Una funzione di hash non può mai essere invertibile perché, per definizione, comprime un messaggio di lunghezza arbitraria (anche molto grande) in un valore di lunghezza fissa.
Questo significa che:
- L'insieme degli input è infinitamente più grande dell'insieme degli output.
- Quindi esistono molti messaggi diversi che producono lo stesso valore di hash (collisioni).
- Non esiste un solo messaggio associato a ogni hash, ma infiniti possibili messaggi.

Di conseguenza, non è possibile risalire in modo univoco all'input a partire dal valore di hash, e quindi la funzione non è invertibile.

b) Spiegare cosa significa affermare che una *funzione di hash* $y = y(x)$ (necessariamente *non invertibile*!) è *non unidirezionale*.

Dire che una funzione di hash non è unidirezionale significa che è facile calcolare un input x a partire da un dato valore di hash y, cioè si può trovare rapidamente un messaggio che produce quell'hash.

Anche se una funzione di hash non è invertibile in senso stretto (perché più messaggi possono produrre lo stesso hash), una buona funzione dovrebbe rendere difficile trovare qualsiasi input x tale che y = y(x).

Se invece è facile trovare x dato y, allora la funzione è vulnerabile e non offre protezione, per esempio contro attacchi che cercano di ricostruire la password a partire dal suo hash.

c) Ti viene offerta una ricompensa, se riesci a dimostrare che una certa *funzione di hash* $y = y(x)$ è resistente alle collisioni (senza specificare meglio). Preferirai tentare di dimostrare che non è *fortemente resistente*, o che non è *debolmente resistente*? Perché?

Preferisco tentare di dimostrare che non è debolmente resistente alle collisioni, perché è la sfida più facile da affrontare.
Infatti:
- Per violare la debole resistenza, basta trovare un messaggio x2 che collide con un hash y noto, cioè trovare x2 tale che y(x2) = y(x1), con x1 noto.

- Per violare la forte resistenza, invece, bisogna trovare due messaggi qualsiasi distinti x1 ≠ x2 tali che y(x1) = y(x2), senza conoscere nessun hash in partenza: questo è molto più difficile.

d) Si consideri una ipotetica funzione di *hash* $h = h(m) = \alpha^m \bmod p$, dove p è un primo tale per cui il problema del logaritmo discreto sia intrattabile in $\mathbb{Z}_p^*$, α non è un elemento generatore di $\mathbb{Z}_p^*$, e m è un intero qualsiasi. Si spieghi perché tale funzione di hash $h = h(m)$ è unidirezionale, ma non resistente alle collisioni (neanche debolmente).
Il fatto che α non sia un elemento generatore indebolisce la unidirezionalità? In che senso?

Unidirezionalità: La funzione h(m) = a^m mod p è unidirezionale perché:
- Calcolare h(m) è facile (basta fare un'esponenziazione modulare).
- Invertire h(m), cioè trovare m dato h(m), equivale a risolvere il logaritmo discreto in Z*_p.
- Questo problema è considerato computazionalmente difficile, quindi la funzione è unidirezionale.

Non resistente alle collisioni (neanche debolmente):
La funzione non è resistente alle collisioni, nemmeno in senso debole, perché:
- a non è un generatore di Z_p, quindi i valori che può produrre (cioè l'immagine della funzione) sono un sottoinsieme ridotto di Z_p.
- Questo significa che ci sono molti valori diversi di m che producono lo stesso valore di h(m).

Inoltre, se l'ordine del sottogruppo generato da a è q, allora per ogni m vale:
h(m) = h(m + k * q) per qualunque intero k.
Quindi è facile trovare collisioni, ad esempio scegliendo:
m1 = m, m2 = m + q → h(m1) = h(m2).

Il fatto che a non sia un generatore indebolisce la unidirezionalità? Parzialmente sì.

Se a non è un generatore, l'esponente m può essere determinato modulo q, dove q è l'ordine del sottogruppo generato da a (q < p-1).

Quindi, la ricerca del logaritmo discreto è più facile, perché è ridotta da modulo p-1 a modulo q.
Questo restringe lo spazio di ricerca e potrebbe rendere la funzione meno resistente agli attacchi di inversione, quindi indebolire la unidirezionalità.

Domanda 4

(svolgere su questo foglio nello spazio assegnato) (4 punti)

La *Trusted Authority* TA adotta lo *Schema di Blom* per distribuire chiavi simmetriche di sessione $K_{ij} = K_{ji}$ a 150 utenti U_k ($k = 1, \ldots, N$) per la comunicazione tra gli stessi. TA sceglie e tiene segreti a, b, c, e pubblica p. Un provider fornisce canali sicuri da TA verso ogni utente, ma a pagamento.

a) Quanti numeri devono essere inviati in tutto da TA agli utenti adottando lo schema di Blom?

300

b) Se invece TA generasse centralmente tutte le possibili chiavi di sessione $K_{ij} = K_{ji}$ e le inviasse ai rispettivi utenti, quanti numeri dovrebbe inviare in tutto?

22350

c) Si consideri il caso di tre soli utenti A, B e C, con identificativi pubblici rispettivamente uguali a $r_A = 101$, $r_B = 104$, $r_C = 110$. TA sceglie e tiene segreti a, b, c, e pubblica $p =$ ~~463~~ 907. Gli utenti A e B però si accordano e si scambiano le informazioni $a_A = 463$, $b_A = 228$, $a_B = 529$, $b_B = 288$.
   - Calcolare i parametri segreti a, b, c.
   - Calcolare le tre chiavi simmetriche distribuite da TA K_{AB}, K_{AC}, K_{BC}.

$$a_A = \begin{cases} a + b\cdot 101 \equiv 463 \pmod{907} \\ a + b\cdot 104 \equiv 529 \pmod{907} \\ b + c\cdot 101 \equiv 228 \pmod{907} \end{cases}$$

$b \cdot 3 \equiv 66 \pmod{907}$

$3^{-1} \equiv 605 \pmod{907}$

$b \equiv 605 \cdot 66 \equiv 22$

$a \equiv 463 - 22 \cdot 101 \equiv 55$

$c \equiv (228 - 22)\, 458 \equiv 20$

$101^{-1} \equiv 458 \pmod{907}$

$K_{AB} = 593$

$K_{AC} = 147$

$K_{CB} = 464$

Domanda 5

(rispondere su questo foglio negli spazi assegnati) (11 punti)
(NB: ogni risposta non giustificata adeguatamente, anche con pochissime parole, avrà valore nullo).

---

1) Cos'è un *elemento primitivo* dell'insieme $\mathbb{Z}_p^*$?

Cos'è un *residuo quàdratico* dell'insieme $\mathbb{Z}_p^*$?

Si calcolino tutti i residui quadratici dell'insieme $\mathbb{Z}_{13}^*$, partendo dalle potenze dell'elemento primitivo $\alpha = 6 \in \mathbb{Z}_{13}^*$. Esaminando i risultati ottenuti, si dica quali sono le radici quadrate di $-3 \pmod{13}$. *(4 punti)*

$6^0 \equiv 1 \pmod{13}$

$6^2 \equiv 10$

$6^4 \equiv 9$

$6^6 \equiv 12$

$6^8 \equiv 3$

$6^{10} \equiv 4$

$\alpha_q = \{1, 3, 4, 9, 10, 12\}$

$\sqrt{10} \equiv \sqrt{-3} \equiv \pm 6 \equiv \{6, 7\}$

---

2) Una tabella raccoglie i valori di *hash*, di lunghezza $L = 25$ bit, calcolati su N file diversi da un archivio. Sia $P(N)$ la probabilità che almeno due di quei file abbiano lo stesso hash in tabella. Quale dovrebbe essere il valore massimo di N affinché $P(N) < 0.01$? *(2 punti)*

$e^{-N^2/2 \cdot 2^{25}} > 0{,}99$

$P \cong 1 - e^{-N^2/2 \cdot 2^L}$

$\frac{N^2}{2^{26}} < \ln \frac{100}{99} \rightarrow N < 821$

Cognome e nome: *(stampatello)*
(firma leggibile)

Matricola:

3) Si consideri un generatore di password consistenti di 12 caratteri casuali scelti nell'alfabeto coreano, che comprende 19 consonanti e 21 vocali. Qual è la quantità di informazione [bit] delle password, se i simboli sono scelti indipendentemente una dall'altro, e la probabilità che siano una consonante o una vocale vale rispettivamente 25% e 75%? *(2 punti)*

$$H(X) = -\left(0{,}25 \cdot \log_2 \frac{0{,}25}{19} + 0{,}75 \log_2 \frac{0{,}75}{21}\right) =$$

$$= 2{,}562 + 3{,}6055 \text{ bit/simbolo} = 5{,}1675 \text{ bit/simbolo}$$

$$H(12 \text{ caratteri}) = 62{,}01 \text{ bit}$$

4) Sono entrato in possesso del file di sistema, in cui sono memorizzati gli hash delle password degli utenti per l'accesso a un server. Se la funzione di hash è debole, posso trovare le password scelte dagli utenti? La presenza di un salt per ogni password può compensare parzialmente la debolezza della funzione di hash? *(3 punti)*

Sì, se la funzione di hash è debole, è possibile trovare le password degli utenti.
Una funzione di hash è considerata debole se:
- Non è unidirezionale (quindi facilmente invertibile),
- Non è resistente agli attacchi a dizionario o con tabelle precalcolate (come le rainbow tables),
- È troppo veloce (adatta a essere sfruttata in attacchi brute force).

In questi casi, un attaccante può provare molte password rapidamente fino a trovare una corrispondenza con l'hash.

Tuttavia, la presenza di un salt per ogni password migliora la sicurezza, anche se la funzione di hash è debole: Il salt è un valore casuale aggiunto alla password prima di calcolare l'hash.

Ogni password ha un hash diverso, anche se due utenti usano la stessa password.

Questo rende inefficaci le tabelle precalcolate (come le rainbow tables), perché bisognerebbe crearne una per ogni possibile valore di salt.