Principles of Cryptocurrency Design

Appunti ed Esercizi

Francesco Pasquale

14 marzo 2024

1 Digital signatures

Si consideri l'algoritmo per la firma digitale mostrato negli esercizi della prima lezione.

Algorithm 1 Hashed RSA

```
1. from Crypto.PublicKey import RSA
2. from hashlib import sha256
3.
4. keyPair = RSA.generate(bits = 1024)
5.
6. msg = b'Benvenuti al corso di Principles of Cryptocurrency Design - AA 23/24!'
7. msg_hash = int.from_bytes(sha256(msg).digest(), byteorder = 'big')
8. msg_sig = pow(msg_hash, keyPair.d, keyPair.n)
9.
10. print("Firma: ", hex(msg_sig))
```

Esercizio 1. Supponiamo che, invece di firmare l'*hash* del messaggio (linee 7 e 8 in Algorithm 1), avessimo firmato direttamente il messaggio:

Algorithm 2 Textbook RSA

```
1. from Crypto.PublicKey import RSA
2.
3. keyPair = RSA.generate(bits = 1024)
4.
5. msg = b'Benvenuti al corso di Principles of Cryptocurrency Design - AA 23/24!'
6. msg_sig = pow(int.from_bytes(msg, byteorder='big'), keyPair.d, keyPair.n)
7.
8. print("Firma: ", hex(msg_sig))
```

1. Scrivere un programma per verificare che la seguente

   `Firma: 0x3379f499aabecf36d6fbb5a516c4c5a8021b31ddd5457362c6019c16f04f024ebd44e51d8400582252f4743cbbb90778d93ba7df6ef4bec490dab61b15d748bde10ce7a0ff7ecdbe0771ff38a1c6477269ff4f643f65d29030935582073bdb9bf5e230ceb7aa7b1519fb0284527f518e216f8ca48641159ddd00054903320e88`

   è una firma valida del messaggio *Benvenuti al corso di Principles of Cryptocurrency Design - AA 23/24!* generata con lo schema in Algorithm 2 con la chiave privata associata alla seguente chiave pubblica

   `n = 0xb6dc6d2f805a5ab512fa20094deed475a3594f600a614a09afea784480fd8c41ab664ed7e90e27e648d38680abaf2574523acbf26b9169b0a75cc74ba024418ca60b79e526f4c613e4`

`cf3910d2f01cf3ac2e0fb2c9587a5ec48e477b761390a79e7b909a1ec899a81d5a824ccc438ed18d4611c53c32bda550823a68744a9be3`

`e = 0x10001`

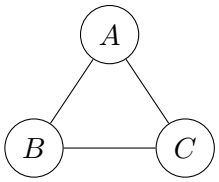
2. Mostrare che lo schema di firma digitale in Algorithm 2 non è sicuro trovando una coppia di numeri interi (`fake_msg`, `fake_sig`) tale che `fake_sig` risulti una firma valida per `fake_msg` relativamente alla chiave pubblica del punto precedente.[1]

3. Notare il motivo per cui l'attacco del punto precedente invece non è attuabile sullo schema di firma digitale in Algorithm 1.

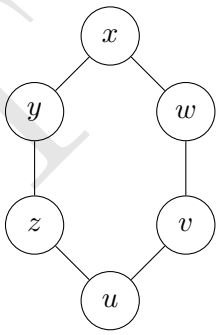
2 Lower bound per Byzantine Broadcast senza PKI

Esercizio 2. Considerate i sistemi G e H qui di seguito.

Sistema G



Sistema H



1. Eseguire il protocollo di Dolev-Strong con $f = 1$ nel sistema G, quando tutti i nodi sono onesti, A è il nodo sorgente e riceve in input 1.

2. Eseguire il protocollo di Dolev-Strong con $f = 1$ nel sistema H in cui x e u sono copie esatte[2] di A e ricevono in input 1, y e v sono copie esatte di B e z e w sono copie esatte di C (tutti i nodi sono onesti e seguono il protocollo).

3. Osservare che le esecuzioni dei due punti precedenti sono equivalenti (A, B e C non possono distinguere se sono nel sistema G o nel sistema H).

4. Eseguire il protocollo di Dolev-Strong con $f = 1$ nel sistema H in cui x e u sono copie esatte di A, ma alla copia di A in x viene dato in input 1 mentre alla copia di A in u viene dato in input 0; y e v sono copie esatte di B e z e w sono copie esatte di C (tutti i nodi sono onesti e seguono il protocollo). Quali sono gli output dei sei nodi?

5. È possibile simulare l'esecuzione in H del punto precedente con una esecuzione equivalente nel sistema G in cui

   (a) B e C sono nodi onesti e A è un nodo corrotto che simula il comportamento dei nodi x, u, v e w?

   (b) A e B sono nodi onesti e C è un nodo corrotto che simula il comportamento dei nodi z, u, v e w?

[1] Hint: Partire da una `fake_sig` arbitraria e trovare un `fake_msg` che ha `fake_sig` come firma.

[2] Ossia entrambe hanno la stessa chiave privata

Esercizio 3. A lezione abbiamo dimostrato che, in assenza di PKI non esistono protocolli per il problema Byzantine Broadcast che soddisfano *validity* e *consistency* per tre nodi, se almeno uno dei tre è corrotto.

Generalizzare la dimostrazione al caso di un numero di nodi arbitrario: dato un n qualunque, in assenza di PKI nessun protocollo per Byzantine Broadcast può soddisfare *validity* e *consistency*, se almeno $n/3$ dei nodi sono corrotti.