04 - Authentication and Digital Signatures

- **Integrity**: il ricevente del messaggio deve essere in grado di verificare che il contenuto corrisponda a quello del messaggio inviato
- **Authentication**: il ricevente del messaggio deve essere in grado di verificare l'*identità* del mittente.
- **Digital signature**: proprietà composta necessaria quando il mittente e ricevente non si fidano uno dell'altro (simile a una firma su carta)

Hash functions

$f : X \to Y$
$|X| = n, |Y| = m,\ n \gg m$
dato $x \in X\ y = f(x) \in Y$

- f : funzione hash
- x : pre-digest
- y : valore hash (detto digest)
- X : dominio di f
- Y : range di f

Proprietà delle funzioni hash

f deve essere *molti a uno* ma è *bilanciata*

$$X_i = \{x \in X : f(x) = y_i\}, 1 \leq i \leq m$$
$$|X_1| \approx |X_2| \approx \ldots \approx |X_m|$$

f è configurata in modo che valori molto vicini in X sono mappati in valori **molto** lontani in Y

Cryptographic hash functions

Una funzione hash usata in crittografia (anche detta *one-way hash function*) è ina funzione hash che soddisfa anche queste proprietà:

1. Per ogni $x \in X$ è semplice calcolare $f(x)$
2. Per ogni $y \in Y$ è computazionalmente infattibile trovare una $x \in X$ tale che $f(x) = y$
3. Dato x_1 è computazionalemnte infattibile trovare un x_2 diverso da x_1 tale che $f(x_1) = f(x_2)$

Si consideri un **blocco di parità da 8 bit**:

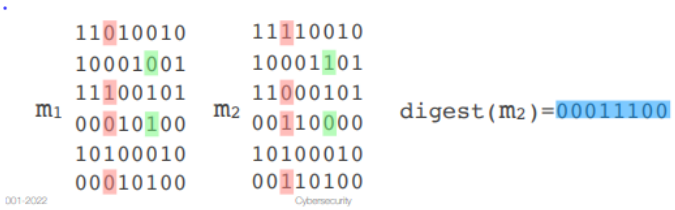
```
m=11010010100010011110010100010100101000100000101

                    b1=110100010
                    b2=100010010
                    b3=111001010
                    b4=000101000
                    b5=101000010
                    b6=000101000
                digest=000111000 (column-wise ⊕)
```

Il blocco di parità da 8 bit soddisfa le proprietà di **bilanciamento** e di **dispersione** delle funzioni hash
Ma non soddisfa la **seconda** e la **terza** proprietà delle funzioni crittografiche hash

Esempio (violazione della seconda proprietà):

- Dato un *digest* è difficile trovare un **pre-digest** che mappi ad esso.
- Esempio (violazione della terza proprietà):
  $m_1 = 11010010100010011110010100010100101000100000101$
  Si trovi un m_2 (diverso da m_1) che abbia lo stesso digest di m_1
  Sappaimo che il digest di m_1 è $\text{digest}(m_1) = 00011100$
  Si possono invertire qualsiasi numero pari di bit in m_1 che sono nella stessa colonna e la parità non cambia



Esempi pratici di funzioni hash:

- MD5 → 128bit
- SHA-0,SHA-1,SHA-2 → lunghezza variabile

MD5

Digest di 128 bit

```
MD5("The quick brown fox jumps over the lazy dog")
  = 9e107d9d372bb6826bd81d3542a419d6

MD5("The quick brown fox jumps over the lazy fog")
  = f0f0996b26d7e959fe3652b4976fc62d
```

Riconosciuto vulnerabile a certi collision attack

Firme digitali

Firme su carta:

- Solo un individuo può generarla
- Non può essere falsificato da altri
- Non può essere riusato (in documenti differenti)
- Il docuemnto firmato non può essere modificato
- Non può essere ripudiato dal firmatario

Firme digitali

- Devono avere le stesse proprietà delle firme su carta
- Essendo all fine una serie di bit possono essere copiate o duplicate perfettamente (al contrario di una firma su carta)

Proprietà

- **Autentica**: Prova che il firmatario e nessun'altro ha volontariamente segnato il documento
- **Non riusabile**: Firma una parte di un singolo documento e non può essere spostata in un altro documento
- **inalterabile**: Dopo che viene firmata il documento non può essere alterato
- **Non può essere rifiutato**: Il firmatario non può affermare di non aver firmato lui il documento

Due operazioni

- Firma: generazione della firma del messaggio m da parte di A
  $\text{Sign}(m, A) \rightarrow \sigma$
- Verifica: verifica che firma σ appartenga ad A
  $\text{Verify}(\sigma, A) \rightarrow \{\text{true}, \text{false}\}$

Protocollo 1
Chiave pubblica (asimmetrica)

```
A: Sign
    s=D(m,k_A[priv])
    send <A,m,s>
B: Verify
    receive <A,m,s>
    m*=C(s,k_A[pub])
    if m*= m
       then true else false
```

Il cifrario deve essere commutativo: $D(C(m)) = C(D(m)) = m$
RSA è commutativa

il messaggio segnato non indica nessun ricevente
Chiunque può verificare il messaggio segnato
Il messaggio non è confidenziale visto che l'atto di verifica del messaggio ne rivela il contenuto
La lunghezza del messaggio inviato è doppia rispetto al messaggio originale
La firma è autentica, non riusabile, inalterabile e irripudiabile da A

Protocollo 2

Aggiunta confidenzialità e destinazione specifica

```
A: Sign
    c=C(m,k_B[pub])    //encrypt
    s=D(c,k_A[priv])   //sign
    send <A,c,s>
B: Verify
    c*=C(s,k_A[pub])   //verify
    m*=D(c*,k_B[priv]) //decrypt
    if m* makes sense
        then true else false
```

Protocollo 3

Basato su criptographic hash functions
Sia $f()$ una funzione hash crittografica:

```
Sign:
    s=D(f(m),k_A[priv]) // sign the digest
    c=C(m,k_B[pub])     // encrypt
    send <A,c,s>

Verify:
    m*=D(c,k_B[priv])       // decrypt
    if f(m*)=C(s,k_A[pub]) // verify
        then true else false
```

Problemi rimasti
Un messaggio firmato può essere mandato nuovamente dopo un po' di tempo:

- "*invia $100 da A a B*"
  Necessario aggiungere i *timestamps*, questo rende difficile per A ottenere la chiave pubblica di B e viceversa. Un semplice scambio di chaivi può essere soggetto ad un attacco **man-in-the-middle**

MAC

Message Authentication Codes

Un **digest** corto e di dimensione prefissata di un messaggio che può essere generato solo da un **sender specifico**
Può essere usato per **autenticare** il sender e verificare l'**integrità** del messaggio
Ottenuto tramite una criptographic hash function assieme a una secret key che vengono condivisi dal sender al ricevente.

Esempio
Dato una funzione crittografica hash $f()$ si può generare il **MAC** del messaggio m applicando $f()$ alla concatenazione di m con una chiave segreta k

$$MAC(m) = f(m|k)$$

Il sender invia la tupla: $(m, MAC(m))$
Il ricevente calcola il MAC del messaggio ricevuto m e lo compara al MAC contenuto nel messaggio
Se coincidono il ricevente ha utenticato il mittente e verificato l'integrità del messaggio visto che nessun'altro avrebbe potuto inviare quella tupla e il contenuto dei messaggio non può essere stato modificato.

MAC con crittografia simmetrica

A e B condividono una chiave privata k

1. A invia $(m, f(m|k))$ a B
2. B riceve (μ, ω)
3. B conosce k quindi può completare $f(\mu|k))$
4. B compara $f(\mu|k))$ con ω
5. Se $f(\mu|k)) = \omega$ allora B può concludere che $\mu = m$ (inegrità) e che il mandante di m è A (autenticazione)

è facile calcolare il MAC di un messaggio ma è difficile calcolare il messaggio dato il MAC.
Esempio di una "*keyed hash function*"
Simile a la firma digitale ma più debole visto che la regola di "non rebudiabilità" non è soddisfatta (il ricevente puù affermare di aver ricevuto un qualsiasi messaggio)
basato su un algoritmo di condivisione di chiavi segrete con tutti i suoi shortcoming associati.