CRITTOGRAFIA

- Identificazione
- Autenticazione
- Firma digitale

Identificazione, autenticazione e firma digitale

Identificazione:

Un sistema di elaborazione, isolato o in rete, deve essere in grado di accertare l'identità di un utente che richiede di accedere ai suo i servizi.

Autenticazione:

Il destinatario di un messaggio deve essere in grado di accertare

- l'identità del mittente
- l'integrità del crittogramma ricevuto.

Identificazione, autenticazione e firma digitale.

Firma digitale

1. MITT non deve poter negare di aver inviato un messaggio m.
2. DEST deve essere in grado di autenticare il messaggio
3. DEST non deve poter sostenere che m' ≠ m è il messaggio inviato da MITT.

Tutto deve essere verificabile da una terza parte.

Relazioni tra le funzionalità

Non sono indipendenti, ma ciascuna estende le precedenti

- L'autenticazione di un messaggio garantisce l'identificazione del mittente.
- L'apposizione della firma garantisce l'autenticazione del messaggio.

Ogni funzionalità è utilizzata per contrastare gli attacchi attivi.

Esistono realizzazioni algoritmiche basate sui **cifrari asimmetrici** e simmetrici.

Funzioni hash

Una funzione hash $f: X \to Y$ è una funzione tale che

$$n = |X| \gg m = |Y|$$

$\exists\ X_1, X_2, \ldots, X_m \subseteq X$ disgiunti t.c.

$$X = X_1 \cup X_2 \cup \ldots \cup X_m$$

$$\forall\ i,\ \forall\ x \in X_i\ ,\quad f(x) = y$$

Una **buona** funzione hash deve assicurare che

I sottoinsiemi $X_1, \ldots, X_m$ abbiano circa la stessa cardinalità

due elementi estratti a caso da X hanno probabilità circa 1/m di avere la stessa immagine in Y

Elementi di X molto "simili" tra loro appartengano a due sottoinsiemi diversi

se X è un insieme di interi, due elementi con valori prossimi devono avere immagini diverse

Gestione delle collisioni

L'algoritmo che impiega la funzione hash dovrà affrontare la situazione in cui più elementi di X hanno la stessa immagine in Y.

Funzioni hash one-way

Se la funzione è applicata in crittografia, deve soddisfare le seguenti proprietà:

1. per ogni $x \in X$ è computazionalmente facile calcolare
$$y = f(x)$$
2. Proprietà one-way: per la maggior parte degli $y \in Y$ è computazionalmente difficile determinare $x \in X$ tale che
$$f(x) = y, \quad \text{i.e., } x = f^{-1}(y)$$
3. Proprietà claw-free: è computazionalmente difficile determinare una coppia di elementi x_1, x_2 in X tali che
$$f(x_1) = f(x_2)$$

Funzioni hash usate in crittografia: MD5 (Message Digest, versione 5)

- Si tratta di una famiglia di algoritmi, quello originale non fu mai pubblicato. Si pubblicarono MD2, seguito da MD4.
- In MD2 e MD4 furono trovate debolezze, e Ron Rivest propose MD5, nel 1992.
- Riceve in input una sequenza S di 512 bit e produce un'immagine di 128 bit: la sequenza è *digerita* riducendone la lunghezza ad un quarto.
- E' stato in seguito dimostrato che MD5 non resiste alle collisioni, e nel 2004 si sono individuate delle debolezze serie.
- Oggi la sua sicurezza si considera severamente compromessa.
- Lo stesso Rivest ha affermato (2005) che MD5 era da considerarsi chiaramente forzata dal punto di vista della resistenza alle collisioni.

Funzioni hash usate in crittografia: RIPEMD-160

- Versione "matura" delle funzioni MD
- Nata nel 1995 nell'ambito di un progetto dell'Unione Europea
- Produce immagini di 160 bit ed è esente dai difetti di MD5

Funzioni hash usate in crittografia: SHA Secure Hash Algorithm

- Progettata da NIST e NSA nel 1993, si adotta quando la proprietà di claw-free è cruciale per la sicurezza del sistema
- Opera su sequenze lunghe fino a 2^{64} bit e produce immagini di 160 bit
- È una funzione crittograficamente sicura: soddisfa i requisiti delle funzioni hash one-way, e genera immagini molto diverse per sequenze molto simili.
- La prima versione pubblicata (SHA-0) conteneva una debolezza, scoperta in seguito da NSA, che portò a una revisione dello standard.

Funzioni hash usate in crittografia: SHA Secure Hash Algorithm

- 1993

  SHA-0: proposta dal NIST nel 1993, presto ritirata a causa di una debolezza interna
- 1995

  SHA-1: progettato da NSA, uso raccomandato dal NIST
- 2001

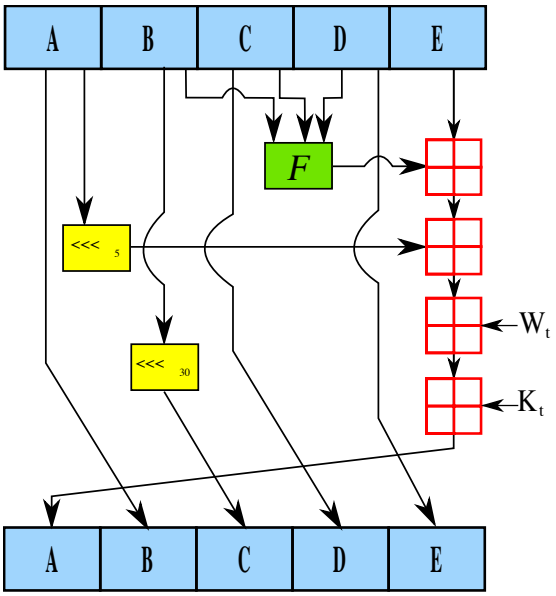
  SHA-2: quattro funzioni della famiglia SHA, progettate da NSA e pubblicate dal NIST, caratterizzate da digest più lunghi
- 2007

  A causa degli attacchi su MD5 e SHA-0, e di attacchi teorici su SHA-1, il NIST ha sollecitato proposte per un nuovo algoritmi hash. Ci sono stati 63 candidati.
- 2012

  Si è conclusa la valutazione, ed è stato selezionato una funzione hash, di progettazione non governativa → SHA-3 (team di analisti italiani e belgi, rilasciata ufficialmente nel 2015).

Funzioni hash usate in crittografia: SHA-1

- Opera su sequenze lunghe fino a 2^{64}-1 bit, e produce immagini di 160 bit.
- E' molto usata nei protocolli crittografici anche se non è più certificata come standard
- Tutte le altre funzioni hanno una struttura molto simile a SHA-1
- Opera su blocchi di 160 bit, contenuti in un buffer di 5 registri di 32 bit ciascuno, in cui sono caricati inizialmente dei valori pubblici
- Il messaggio *m* viene concatenato con una sequenza di padding che ne rende la lunghezza multipla di 512 bit
- Il contenuto dei registri varia nel corso dei cicli successivi in cui questi valori si combinano tra loro e con blocchi di 32 bit provenienti da *m*
- Alla fine del procedimento, i registri contengono SHA-1(m).



Un'iterazione all'interno della funzione di compressione di SHA-1. A, B, C, D ed E sono parole di stato a 32 bit; F è una funzione non lineare che varia; $\lll_n$ denota una rotazione del bit di sinistra di n posti; n varia per ogni operazione. $\boxplus$ denota l'addizione modulo 2^{32}. K_t è una costante.

W_t blocco di 32 bit ottenuto tagliando e rimescolando i blocchi di messaggio

Il contenuto dei registri varia nel corso dei cicli (all'inizio sono caricati valori fissi e pubblici) in cui questi valori si combinano tra loro e con blocchi di 32 bit provenienti dal messaggio W, nonché con alcuni parametri relativi al ciclo.
Alla fine del procedimento (quando è stato letto l'intero messaggio) i registri contengono l'hash SHA1(W)

Identificazione su canali sicuri

ESEMPIO: accesso di un utente alla propria casella di posta elettronica, o a file personali memorizzati su un calcolatore ad accesso riservato ai membri della sua organizzazione.

- l'utente inizia il collegamento inviando in chiaro login e password
- se il canale è protetto in lettura e scrittura, un attacco può essere sferrato solo da un utente locale al sistema:
  - ad esempio l'amministratore che ha accesso a tutti i file memorizzati (oppure un hacker)
- Il meccanismo di identificazione prevede una cifratura delle password, realizzata con funzioni hash one-way.

Cifratura password nei sistemi UNIX

Quando un utente U fornisce per la prima volta una password P

il sistema associa a U due sequenze binarie (che memorizza nel file delle password al posto di P):

- S (seme)

  prodotta da un generatore pseudocasuale

- Q = h (PS)

  h: funzione hash one-way

Cifratura password nei sistemi UNIX

Ad ogni successiva connessione di U, il sistema:

- recupera S dal file delle password,
- concatena S con la password fornita da U
- calcola l'immagine one-way della nuova sequenza: h(PS)
- se h(PS) = Q l'identificazione ha successo.

Un accesso illecito al file delle password non fornisce informazioni interessanti:

è computazionalmente difficile ricavare la password originale dalla sua immagine one-way

Protezione del canale

Se il canale è insicuro, la password può essere intercettata durante la sua trasmissione in chiaro.

Il sistema non dovrebbe mai maneggiare direttamente la password, ma una sua immagine inattaccabile.

Canale insicuro: identificazione

$< e, n >, < d >$ = chiavi pubblica e privata di un utente U che richiede l'accesso ai servizi offerti dal sistema S.

1. S genera un numero casuale $r < n$ e lo invia in chiaro a U.
2. U calcola
   $$f = r^d \bmod n \quad \text{(firma di U su r)}$$
   con la sua chiave privata e lo spedisce a S.
3. S verifica la correttezza del valore ricevuto calcolando e verificando se
   $$f^e \bmod n = r$$
   Se ciò avviene, l'identificazione ha successo

Canale insicuro: identificazione

- Le operazioni di cifratura e decifrazione sono invertite rispetto all'impiego standard nell'RSA
- Possibile perché le due operazioni sono commutative

$$(x^{e} \bmod n)^{d} \bmod n = (x^{d} \bmod n)^{e} \bmod n \ (=x)$$

- f può essere generata solo da U che possiede < d >
- Se il passo 3 va a buon fine, il sistema ha la garanzia che l'utente che ha richiesto l'identificazione sia effettivamente U, anche se il canale è insicuro

Canale insicuro: identificazione

Problema:

S chiede a U di applicare la sua chiave privata a una sequenza r che S stesso ha generato

potrebbe essere stata scelta di proposito per ricavare qualche informazione sulla chiave privata di U.

Protocollo alternativo a "conoscenza zero"

impedisce che da una comunicazione si possa estrarre più di quanto sia nelle intenzioni del comunicatore

Canale insicuro: autenticazione

DEST deve autenticare il messaggio accertando l'identità di MITT e l'integrità di m

MITT e DEST concordano una chiave segreta k.

Canale insicuro: autenticazione

MITT

- allega al messaggio un MAC (Message Authentication Code) A(m, k), allo scopo di garantire la provenienza e l'integrità del messaggio.
- spedisce la coppia < m, A(m, k) > in chiaro,
- oppure, cifra m e spedisce < C (m, k'), A(m, k) >
  - C: funzione di cifratura,
  - k': chiave pubblica o segreta del cifrario scelto.

Canale insicuro: autenticazione

DEST

- entra in possesso di m (dopo averlo eventualmente decifrato)
- essendo a conoscenza di A e k, calcola A(m, k)
- confronta il valore ottenuto con quello inviato da MITT per verificare che il MAC ricevuto corrisponda al messaggio a cui risulta allegato:
  - Se la verifica ha successo il messaggio è autenticato.
  - Altrimenti DEST scarta il messaggio.

Canale insicuro: autenticazione

MAC

- È un'immagine breve del messaggio, che può essere stata generata solo da un mittente conosciuto dal destinatario, previ opportuni accordi.
- Ne sono state proposte varie realizzazioni, basate su cifrari asimmetrici, simmetrici e sulle funzioni hash one-way

MAC con funzioni hash one-way

A(m, k) = h(mk), con h funzione hash one way

Risulta computazionalmente difficile per un crittoanalista scoprire la chiave segreta k

- h è nota a tutti, e m può viaggiare in chiaro o essere scoperto per altra via, ma k viaggia all'interno del MAC
- per recuperare k si dovrebbe invertire h

Il crittoanalista non può sostituire facilmente il messaggio m con un altro messaggio m'

- dovrebbe allegare alla comunicazione di m' il MAC A(m', k) che può produrre solo conoscendo k.

CBC + MAC

- Usando un cifrario a blocchi in modalità CBC, si può usare il blocco finale del crittogramma come MAC
- Il blocco finale è infatti funzione dell'intero messaggio

Firma manuale

1. è autentica e non falsificabile
   prova che chi l'ha prodotta è chi ha sottoscritto il documento;
2. non è riutilizzabile
   è legata strettamente al documento su cui è stata apposta;
3. il documento firmato non è alterabile
   chi ha prodotto la firma è sicuro che questa si riferirà solo al documento sottoscritto nella sua forma originale;
4. non può essere ripudiata da chi l'ha apposta
   costituisce prova legale di un accordo o dichiarazione.

Firma digitale

- Non può consistere semplicemente di una digitalizzazione del documento originale firmato manualmente
  un crittoanalista potrebbe "tagliare" dal documento digitale la parte contenente la firma e "copiarla" su un altro documento.
- Deve avere una forma che dipenda dal documento su cui viene apposta, per essere inscindibile da questo.
- Per progettare firme digitali si possono usare sia i cifrari simmetrici che quelli asimmetrici.

Protocollo 1: messaggio m in chiaro e firmato (Diffie e Hellman)

- U: utente, k_U [priv] e k_U [pub]: chiavi di U,
- C e D : funzioni di cifratura e decifrazione di un cifrario asimmetrico

FIRMA

- U genera la firma f = D (m, k_U [priv]) per m
- U spedisce all'utente V la tripla < U, m, f >

Protocollo 1: messaggio m in chiaro e firmato (Diffie e Hellman)

VERIFICA

- V riceve < U, m, f > e verifica l'autenticità della firma f calcolando e controllando che

$$C\ (f, k_U\ [pub]) = m$$

- L'indicazione del mittente U consente a V di selezionare la chiave pubblica k_U [pub] da utilizzare nel calcolo
- I processi di firma e verifica impiegano le funzioni C e D in ordine inverso a quello standard, C e D devono essere commutative:

$$C\ (D\ (m)) = D\ (C\ (m)) = m$$

Protocollo 1: messaggio m in chiaro e firmato

Il protocollo soddisfa i requisiti della firma manuale

- f è autentica e non falsificabile (1)
  - k_U [priv] è nota solo a U
  - per falsificare la firma occorre conoscere k_U [priv], ma D è one-way
- il documento firmato < U, m, f > non può essere alterato se non da U, pena la non consistenza tra m e f (3)
- Poiché solo U può aver prodotto f, U non può ripudiare la firma (4)
- La firma f non è riutilizzabile su un altro documento m' ≠ m poiché è immagine di m (2)

Protocollo 1: messaggio m in chiaro e firmato

- È definito per un particolare utente U, ma non per un particolare destinatario.
- Chiunque può convincersi dell'autenticità della firma facendo uso solo della chiave pubblica di U.
- È solo uno schema di principio:
  - comporta lo scambio di un messaggio di lunghezza doppia dell'originale poiché la dimensione della firma è paragonabile alla dimensione del messaggio
  - il messaggio non può essere cifrato poiché è ricavabile pubblicamente dalla firma attraverso la verifica di questa.

Protocollo 2:
messaggio m cifrato e firmato

FIRMA E CIFRATURA

- U genera la firma $f = D(m, k_U[priv])$ per m
- calcola il crittogramma firmato $c = C(f, k_V[pub])$
  con la chiave pubblica del destinatario V
  - ➢ si incapsula la firma nel documento cifrato
- spedisce la coppia $< U, c >$ a V.

Protocollo 2:
messaggio m cifrato e firmato

DECIFRAZIONE E VERIFICA

- V riceve la coppia $< U, c >$ e decifra il crittogramma:

$$D(c, k_V[priv]) = f$$

- Cifra tale valore con la chiave pubblica di U ottenendo

$$C(f, k_U[pub]) = C(D(m, k_U[priv]), k_U[pub]) = m$$

- V ricostruisce m, e se m è significativo attesta l'identità di U

  Un utente diverso da U ha probabilità praticamente nulla di generare un crittogramma di significato accettabile se "cifrato" con la chiave pubblica di U.

Algoritmo per il protocollo 2

Cifrario RSA, con

- $< d_U >, < e_U , n_U >$ chiavi di U
- $< d_V >, < e_V , n_V >$ chiavi di V

utente U

- genera la firma del messaggio m: $f = m^{d\_U} \bmod n_U$
- cifra f con la chiave pubblica di V: $c = f^{e\_V} \bmod n_V$
- spedisce a V la coppia $< U, c >$

utente V

- riceve la coppia $< U, c >$ e decifra c: $c^{d\_V} \bmod n_V = f$
- "decifra" f con la chiave pubblica di U: $f^{e\_U} \bmod n_U = m$
- Se m è significativo, conclude che è autentico

Algoritmo per il protocollo 2

Per la correttezza del procedimento è necessario che:

- $n_U \leq n_V$ perché risulti $f < n_V$ e f possa essere cifrata correttamente e spedita a V
- questo impedirebbe a V di inviare messaggi firmati e cifrati a U
- ogni utente stabilisce chiavi distinte per la firma e per la cifratura:
  - si fissa pubblicamente H molto grande, ad esempio $H = 2^{1024}$
  - chiavi di firma < H
  - chiavi di cifratura > H

Algoritmo per il protocollo 2

- Il valore elevato di H assicura che tutte le chiavi possano essere scelte sufficientemente grandi, e quindi inattaccabili in modo esauriente.
- Il meccanismo di firma si presta tuttavia a diversi attacchi

  basati sulla possibilità che un crittoanalista si procuri la firma di un utente su messaggi apparentemente privi di senso.

Protocollo 2: attacco 1

Supponiamo che:

- un utente U invii una risposta automatica (ack) a MITT ogni volta che riceve un messaggio m
- il segnale di ack sia il crittogramma della firma di U su m.

Un crittoanalista attivo X può decifrare i messaggi inviati a U:

- X intercetta il crittogramma c firmato inviato da V a U, lo rimuove dal canale e lo rispedisce a U, facendogli credere che c sia stato inviato da lui.
- U spedisce automaticamente a X un ack
- X usa l'ack ricevuto per risalire al messaggio originale m applicando le funzioni del cifrario con le chiavi pubbliche di V e di U

Protocollo 2: attacco 1

1. V invia il crittogramma c a U:
   - $c = C(f, k_U[pub])$
   - $f = D(m, k_V[priv])$
2. Il crittoanalista X intercetta c, lo rimuove dal canale e lo rispedisce a U (U crede che c arrivi da X)
3. U decifra c

   $f = D(c, k_U[priv])$

   e verifica la firma con la chiave pubblica di X ottenendo un messaggio

   $m' = C(f, k_X[pub])$
4. $m' \neq m$ è un messaggio privo di senso, ma il sistema manda l'ack c' a X

   $f' = D(m', k_U[priv])$

   $c' = C(f', k_X[pub])$

Protocollo 2: attacco 1

X usa c' per risalire al messaggio m

1. decifra c' e trova f'

   $D(c', k_X[priv]) = D(C(f', k_X[pub]), k_X[priv]) = f'$
2. verifica f' e trova m'

   $C(f', k_U[pub]) = C(D(m', k_U[priv]), k_U[pub]) = m'$
3. da m' ricava f

   $D(m', k_X[priv]) = D(C(f, k_X[pub]), k_X[priv]) = f$
4. verifica f con la chiave pubblica di V e trova m

   $C(f, k_V[pub]) = C(D(m, k_V[priv]), k_V[pub]) = m$

Protocollo 2: attacco 1

- Per evitare questo attacco occorre che U blocchi l'ack automatico
- l'ack deve essere inviato solo dopo un esame preventivo di m e scartando i messaggi che non si desidera firmare

Protocollo resistente agli attacchi

- Si evita la firma diretta del messaggio
- Si appone la firma digitale su una immagine del messaggio ottenuta con una funzione hash one-way e pubblica (tipo MD5, SHA)
- La firma non è più soggetta a giochi algebrici

Protocollo 3: messaggio m cifrato e firmato in hash

FIRMA E CIFRATURA

- il mittente U calcola h(m) e genera la firma

$$f = D\,(h(m), k_U\,[priv])$$

- calcola separatamente il crittogramma

$$c = C\,(m, k_V\,[pub])$$

- spedisce a V la tripla < U, c, f >

Protocollo 3: messaggio m cifrato e firmato in hash

DECIFRAZIONE E VERIFICA

- V riceve la tripla < U, c, f >
- decifra il crittogramma c: $m = D\,(c, k_V\,[priv])$
- calcola separatamente h(m) e $C\,(f, k_U\,[pub])$
- se $h(m) = C\,(f, k_U\,[pub])$ conclude che il messaggio è autentico.

Protocollo 3: messaggio m cifrato e firmato in hash

- richiede lo scambio di una maggiore quantità di dati, ma l'incremento è trascurabile
- la firma può essere calcolata più velocemente

Attacchi man in-the-middle

- Debolezza dei protocolli:
  - le chiavi di cifratura sono pubbliche e non richiedono un incontro diretto tra gli utenti per il loro scambio.
- Un crittoanalista attivo può intromettersi proprio in questa fase iniziale del protocollo, pregiudicando il suo corretto svolgimento.
- Attacco attivo chiamato man in-the-middle:

  Il crittoanalista si intromette nella comunicazione tra U e V,
  si comporta agli occhi di U come se fosse V,
  si comporta agli occhi di V come se fosse U,
  intercettando e boccando le comunicazioni di U e V

Attacchi man in-the-middle sulle chiavi pubbliche

Il crittoanalista X devia le comunicazioni che provengono da U e V e le dirige verso se stesso:

- U richiede a V la sua chiave pubblica (attraverso e-mail, pagina web, ...)
- X intercetta la risposta contenente k_V [pub] e la sostituisce con la sua chiave pubblica k_X [pub].
- X si pone in ascolto in attesa dei crittogrammi spediti da U a V, cifrati mediante k_X [pub].
- X rimuove dal canale ciascuno di questi crittogrammi, e lo decritta, lo cifra mediante k_V [pub] e lo rispedisce a V.
- U e V non si accorgeranno della presenza di X se il processo di intercettazione e rispedizione è sufficientemente veloce da rendere il relativo ritardo apparentemente attribuibile alla rete.

Certification Authority

- Un algoritmo crittografico è tanto robusto quanto la sicurezza delle sue chiavi
- lo scambio/generazione della chiave è un passo cruciale
- gli attacchi man-in-the-middle sono i principali problemi di sicurezza da affrontare
- Sono nate le Certification authority

  sono infrastrutture che garantiscono la validità delle chiavi pubbliche e ne regolano l'uso, gestendo la distribuzione sicura delle chiavi alle due entità che vogliono comunicare

Key Certification Authority (CA)

ente preposto alla certificazione di validità delle chiavi pubbliche

- La CA autentica l'associazione <utente, chiave pubblica> emettendo un certificato digitale.
- Il certificato digitale consiste della chiave pubblica e di una lista di informazioni relative al suo proprietario, opportunamente firmate dalla CA.
- Per svolgere correttamente il suo ruolo, la CA mantiene un archivio di chiavi pubbliche sicuro, accessibile a tutti e protetto da attacchi in scrittura non autorizzati.
- La chiave pubblica della CA è nota agli utenti che la mantengono protetta da attacchi esterni e la utilizzano per verificare la firma della CA stessa sui certificati.

Certificato digitale

Un certificato digitale contiene:

- una indicazione del suo formato (numero di versione)
- il nome della CA che lo ha rilasciato
- un numero seriale che lo individua univocamente all'interno della CA emittente
- la specifica dell'algoritmo usato dalla CA per creare la firma elettronica
- il periodo di validità del certificato (data di inizio e data di fine)
- il nome dell'utente a cui questo certificato si riferisce e una serie di informazioni a lui legate
- un'indicazione del protocollo a chiave pubblica adottato da questo utente per la cifratura e la firma: nome dell'algoritmo, suoi parametri, e chiave pubblica dell'utente
- firma della CA eseguita su tutte le informazioni precedenti.

CA

- Se U vuole comunicare con V può richiedere K_V [pub] alla CA, che risponde inviando a U il certificato digitale $cert_V$ di V
- Poiché U conosce K_{CA} [pub], può controllare l'autenticità del certificato verificandone il periodo di validità e la firma prodotta dalla CA su di esso.
- Se tutti i controlli vanno a buon fine, K_V [pub] nel certificato è corretta e U avvia la comunicazione con V
- Un crittoanalista potrebbe intromettersi solo falsificando la certificazione, ma si assume che la CA sia fidata e il suo archivio di chiavi sia inattaccabile

CA

Esistono in ogni stato diverse CA organizzate ad albero

la verifica di un certificato è in questo caso più complicata e si svolge attraverso una catena di verifiche che vanno dalla CA di U alla CA di V

- V invia a U una sequenza di certificati ordinati in accordo alla CA che li ha firmati, da quella che ha certificato V a quella che è alla radice dell'albero delle CA

Ogni utente mantiene localmente al sistema una copia del proprio certificato $cert_U$ e una copia di K_{CA} [pub]

interagisce con la CA una sola volta, e poi la gestione delle chiavi pubbliche diventa decentralizzata

Protocollo 4: messaggio m cifrato, firmato in hash e certificato

FIRMA, CIFRATURA E CERTIFICAZIONE

Il mittente U:

- si procura il certificato $cert_V$ di V
- calcola h(m) e genera la firma

$$f = D\,(h(m), k_U\,[priv])$$

- calcola il crittogramma

$$c = C\,(m, k_V\,[pub])$$

- spedisce a V la tripla $< cert_U, c, f >$
  - $cert_U$ contiene la chiave $k_U\,[pub]$ e la specificazione della funzione h utilizzata

Protocollo 4

VERIFICA DEL CERTIFICATO

- il destinatario V riceve la tripla $< cert_U, c, f >$ e verifica l'autenticità di $cert_U$ (e dunque di K_U[pub]) utilizzando la propria copia di K_{CA}[pub]

DECIFRAZIONE E VERIFICA DELLA FIRMA

- V decifra il crittogramma c mediante la sua chiave privata e ottiene

$$m = D\,(c, k_V\,[priv])$$

- V verifica l'autenticità della firma apposta da U su m controllando che risulti

$$C\,(f, k_U\,[pub]) = h(m)$$

Conclusioni

- Il punto debole di questo meccanismo è rappresentato dai certificati revocati, i.e., non più validi
- Le CA mettono a disposizione degli utenti un archivio pubblico contenente i certificati non più validi
- La frequenza di controllo di questi certificati e le modalità della loro comunicazione agli utenti sono cruciali per la sicurezza
- Attacchi man-in-the-middle possono essere evitati se si stabilisce un incontro diretto tra utente e CA nel momento in cui si istanzia il sistema asimmetrico dell'utente