

Ci siamo finora occupati di come due o più persone possano scambiarsi informazioni in modo non solo riservato ma assolutamente segreto: un accordo precedente fa sì che gli individui comunichino in tranquillità anche alla luce del sole, senza che nessuno sospetti che – in realtà – è in corso una conversazione clandestina. Abbiamo anche considerato attacchi esterni alle procedure steganografiche considerate: un nemico che vuole curiosare in un canale di trasmissione che reputa interessante può disturbare lo scambio di dati o, peggio ancora, intrufolarsi modificando i messaggi. Finora, abbiamo sempre dato per scontata la buona fede delle persone coinvolte: cosa succederebbe se la “talpa” fosse all’interno?

Per ovviare a quest’eventualità, esistono delle tecniche particolari, studiate apposta per garantire che nessuno carpisca con l’inganno informazioni che devono rimanere segrete. Si tratta degli algoritmi speciali per protocolli, che assicurano la limpidezza dei personaggi coinvolti, prima ancora che fra loro abbia inizio uno scambio di informazioni, mediante uno dei metodi steganografici precedentemente analizzati.

4 ALGORITMI SPECIALI PER PROTOCOLLI

4.1 Che cos’è un protocollo?

Un protocollo è costituito da una serie di passaggi (*step*), che coinvolge due o più persone, designate per svolgere un incarico. Esso è dunque una sequenza di direttive, da effettuarsi consecutivamente, dall’inizio alla fine, una alla volta, e nessun passaggio può essere iniziato prima che quello precedente sia terminato. E’ necessaria la presenza di almeno due individui, affinché si possa parlare di protocollo; una persona sola non fa un protocollo. Un singolo individuo è perfettamente in grado di eseguire un elenco di disposizioni, al fine di realizzare un lavoro finale, come la preparazione di una torta, ma si tratta di protocollo solo se qualcun altro assaggia il dolce!

I protocolli hanno le seguenti caratteristiche:

- ❖ ogni persona coinvolta deve conoscere il protocollo e tutti i passaggi da seguire in anticipo;
- ❖ ogni individuo deve concordare con l’esecuzione delle disposizioni del protocollo;
- ❖ il protocollo non deve essere ambiguo; ogni direttiva deve essere chiara, ben definita e non deve esserci alcuna possibilità di equivoco;
- ❖ il protocollo deve essere completo; deve essere contemplata un’azione specifica per ogni possibile situazione.

L’esecuzione di un protocollo procede in modo lineare attraverso le molteplici disposizioni, a meno che non ci siano esplicite istruzioni di saltare ad altri passaggi. Ogni step comporta almeno una delle seguenti operazioni: calcoli effettuati da una o più fra le persone coinvolte

- 130 -

oppure invio e ricezione di messaggi fra le parti in gioco.

Un protocollo crittografico è un protocollo che utilizza la crittografia: infatti, coloro che partecipano alla sua realizzazione possono essere amici ed avere fiducia l’uno dell’altro, oppure avversari e non fidarsi affatto. Un protocollo di questo tipo implica l’utilizzo di algoritmi crittografici, che mirano ad evitare eventuali imbrogli, oltre che a garantire la segretezza.

Analizzeremo ora diversi tipi di protocollo: in alcuni di questi, è possibile che uno dei partecipanti inganni gli altri oppure che un “nemico” invalidi il protocollo, apprendendo informazioni riservate. Alcuni falliscono perché i loro progettisti non sono stati sufficientemente precisi nelle richieste, dando così adito all’insinuazione di equivoci nella gestione; altri, invece, non vengono ultimati, perché non sono state vagliate tutte le possibili alternative nell’analisi preventiva.

4.2 Tipi di protocollo

Per convenzione, Alice e Bob sono coloro che danno inizio ad ogni protocollo, Alice comincia e Bob risponde; se il protocollo richiede la presenza di una terza e una quarta persona, Carol e Dave assumeranno questi ruoli. Altri personaggi, che compaiono nella gestione, sono descritti nella tabella qui sotto, nella quale sono elencate le persone partecipanti all’esecuzione di un protocollo:

Alice	Prima partecipante in ogni protocollo
Bob	Secondo partecipante in ogni protocollo
Carol	Terza partecipante
Dave	Quarto partecipante
Eve	Persona esterna al protocollo che vi ha però accesso
Mallory	Nemico attivo
Peggy	E’ a conoscenza di dati riservati e può dimostrarlo
Victor	Verifica che Peggy conosca davvero informazioni segrete
Trent	“Arbitro” fidato
Walter	Guardiano; controlla Alice e Bob in alcuni protocolli

Tabella 1

Esistono tre tipi di protocollo:

- **protocolli in presenza di un “arbitro”;**
- **protocolli in presenza di un “giudice”;**
- **protocolli self-enforcing.**

L’arbitro presente nel primo tipo di protocollo è una terza persona disinteressata, ma fidata, che non ha alcuna interessenza nell’esecuzione del protocollo né particolari rapporti di fiducia con le parti in gioco. La sua parola però corrisponde sempre a verità ed i partecipanti non possono che sottostare alle sue decisioni.

Nel mondo reale, gli avvocati sono spesso utilizzati come arbitri. Ad esempio, se Alice vuole vendere la propria auto a Bob, un estraneo che vorrebbe pagare con un assegno, ella non ha modo di verificare che l’assegno sia coperto. Così, Alice vuole assicurarsi di essere pagata,

- 131 -

prima di fare la voltura; d’altro canto, Bob non permetterà che Alice incassi l’assegno, prima che la macchina sia intestata a lui. Siamo quindi ad un punto morto: interviene l’avvocato, grazie al quale Alice e Bob possono utilizzare il seguente protocollo, a garanzia del fatto che nessuno dei due inganni l’altro:

1. Alice dà i documenti che attestano il cambio di proprietà all’avvocato;

2. Bob dà l'assegno ad Alice;
3. Alice deposita l'assegno presso la propria banca;
4. se l'assegno è coperto, l'avvocato conclude le trattative fornendo a Bob i documenti di cui ha bisogno; se invece Bob è un truffatore, Alice può dimostrare all'avvocato di non aver potuto incassare l'importo concordato ed egli le restituirà l'atto di proprietà.

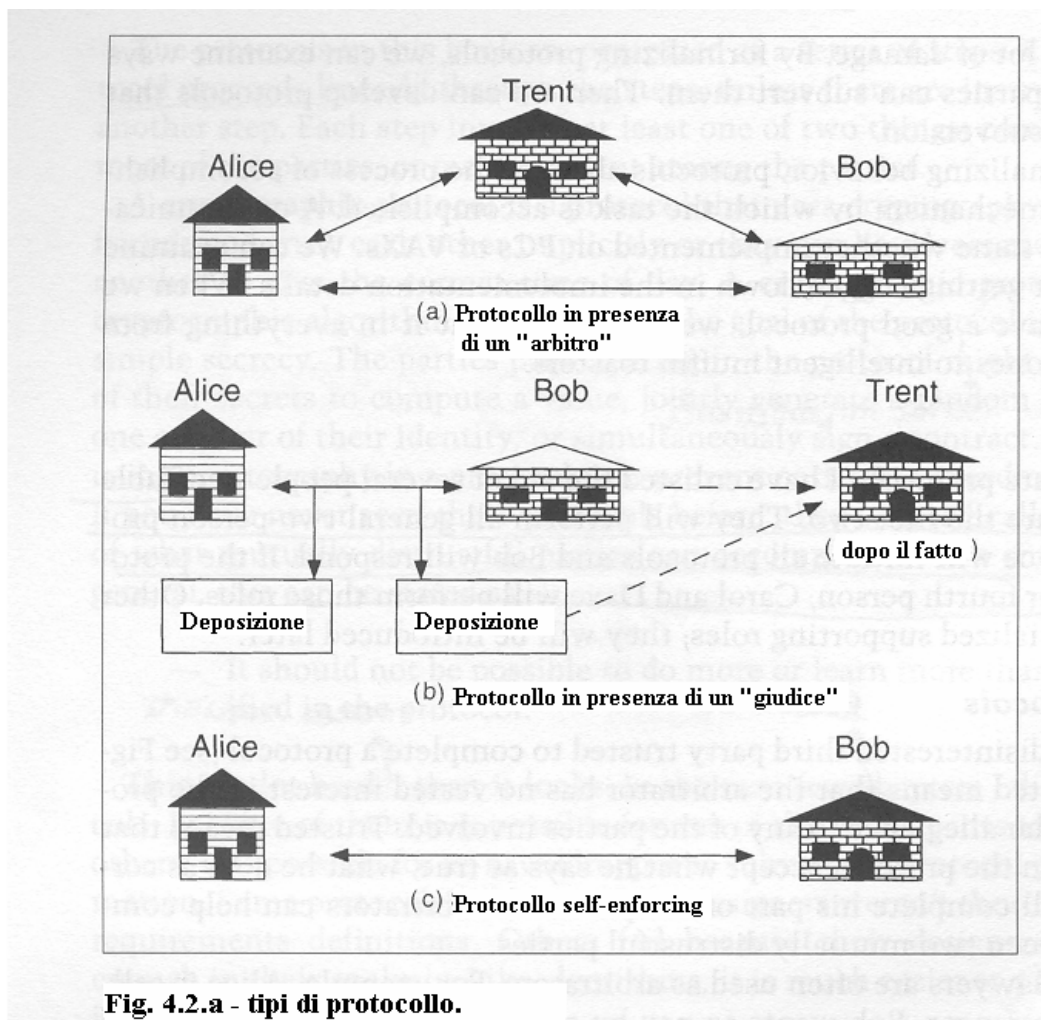


Fig. 4.2.a - tipi di protocollo.

E' evidente che in questo protocollo sia Alice che Bob hanno valide motivazioni per rivolgersi ad una figura *super partes* che tuteli i loro interessi, mentre a quest'ultima non importa che l'assegno sia coperto oppure no, in quanto porterà a termine le proprie mansioni sia nell'uno che nell'altro caso.

Se queste transazioni vengono fatte via computer, la fiducia in chi c'è dall'altra parte del modem si riduce ancora di più ed è facile che due parti in gioco che sospettano l'una dell'altra siano – se possibile – ancora più diffidenti nei confronti di una terza persona, che non cono-

scono e che deve svolgere il ruolo di arbitro. Inoltre, l'arbitro deve essere retribuito per le proprie mansioni e la sua presenza, come descritto nel precedente protocollo, non può che allungare i tempi di conclusione di una trattativa. Infine, poiché i partecipanti ad un protocollo di questo tipo devono fidarsi ciecamente dell'arbitro, egli rappresenta un facile bersaglio per chiunque voglia invalidarne l'esito finale.

I protocolli in presenza di un giudice sono un sottoinsieme di più basso livello dei precedenti, che vengono utilizzati solo in circostanze eccezionali – quando c'è una disputa. Come l'arbitro, anche il giudice è una persona *super partes* disinteressata e fidata, ma, in questo caso, egli non è direttamente coinvolto nell'esecuzione del protocollo: viene chiamato solo per determinare se un protocollo è stato eseguito correttamente.

Se ad esempio, Alice e Bob decidono di stipulare un contratto senza la presenza di un notaio, il giudice non vede il contratto firmato finché uno di loro trascina l'altro in tribunale. Il protocollo può essere così sintetizzato: la prima parte non prevede l'intervento di una terza persona, la seconda sì.

1. Alice e Bob discutono i termini del contratto in sede di negoziazione;
2. Alice firma il contratto;
3. Bob firma il contratto.

Se avviene una disputa fra le parti in gioco, interviene il giudice:

4. Alice e Bob compaiono di fronte ad un giudice;
5. Alice presenta la propria deposizione;
6. Bob espone la propria posizione;
7. il giudice decreta un verdetto finale.

La differenza fra un giudice ed un arbitro è che il primo non sempre è necessario: infatti, la sua presenza è richiesta solo in caso di contenzioso. Un protocollo di questo tipo è finalizzato a scoprire un'eventuale truffa, piuttosto che ad impedirla e questa caratteristica funge da deterrente e scoraggia gli imbrogli.

Il terzo tipo di protocolli è il migliore: in esso è intrinseca l'equità delle operazioni implicate. Non sono necessari né arbitri che completino il protocollo, né giudici che risolvano le eventuali controversie. Se una delle due parti in gioco tenta di ingannare l'altra, questa si accorge immediatamente dell'imbroglio ed interrompe l'esecuzione del protocollo. Sfortunatamente, non esiste un protocollo self-enforcing per ogni situazione.

Eventuali attacchi di tipo crittografico possono essere indirizzati contro le tecniche utilizzate per implementare gli algoritmi di un protocollo, o contro il protocollo stesso. Esistono attacchi passivi, in cui il nemico osserva da fuori e tenta di guadagnare informazioni per lui preziose, ma non può intervenire nell'esecuzione delle disposizioni del protocollo. Gli attacchi attivi, invece, sono quelli in cui il truffatore altera le direttive a proprio vantaggio, spacciandosi per qualcun altro ed introducendo nuovi messaggi nel protocollo, magari dopo aver cancellato quelli reali, o ancora interrompendo un canale di comunicazione o modificando dei dati immagazzinati in un computer.

- 133 -

E' anche possibile che la "talpa" sia interna al protocollo stesso: in tal caso, un doppiogiochista passivo seguirà i passi del protocollo, semplicemente cercando di ottenere dati per lui non ufficialmente disponibili, mentre un nemico attivo potrebbe mentire durante l'esecuzione, compromettendo seriamente l'esito dell'operazione.

4.3 Protocolli di base

Una comune tecnica crittografica consiste nel cifrare ogni singola conversazione con una chiave diversa, detta *chiave di sessione*, proprio perché è utilizzata per un'unica sessione di comunicazioni. Tale chiave è utile, in quanto esiste solo per la durata dello scambio, ma vediamo come gli interlocutori riescono ad ottenerla, per poter avviare la conversazione segreta.

❖ *Scambio della chiave mediante la crittografia simmetrica*

Questo protocollo assume che Alice e Bob, utenti di un network, condividano ciascuno una chiave segreta con il KDC (Key Distribution Center, centro per la distribuzione delle chiavi):

1. Alice chiama Trent e gli richiede una chiave di sessione per comunicare con Bob;
2. Trent genera una chiave di sessione casuale e ne cifra due copie: una diventa la chiave di Alice, l'altra quella di Bob; Trent invia entrambe le copie ad Alice;
3. Alice decifra la propria copia della chiave di sessione;
4. Alice invia a Bob la sua copia della chiave di sessione;
5. Bob decifra la propria copia;
6. Alice e Bob possono usare questa chiave per comunicare in modo sicuro.

Questo protocollo è interamente basato sull'imparzialità di Trent: se il nemico attivo Mallory riuscisse a corrompere Trent, la comunicazione sarebbe compromessa. Mallory disporrebbe di tutte le chiavi private e avrebbe accesso alle conversazioni passate e future fra Alice e Bob. Inoltre, Trent è il punto cruciale, in quanto un suo errore nella distribuzione delle chiavi invaliderebbe il protocollo.

❖ *Scambio della chiave mediante la crittografia a chiave pubblica*

Alice e Bob sfruttano la crittografia a chiave pubblica, per mettersi d'accordo su una chiave di sessione, da utilizzare per cifrare i dati (in alcune implementazioni pratiche, le chiavi pubbliche di Alice e Bob sono disponibili su un database):

1. Alice ricava la chiave pubblica di Bob dal KDC;
2. Alice genera una chiave di sessione casuale, la cifra utilizzando la chiave pubblica di Bob e gliela invia;
3. Bob decifra il messaggio di Alice, usando la propria chiave privata;
4. entrambi cifrano i loro messaggi sfruttando la stessa chiave di sessione.

- 134 -

❖ *L'attacco dell'uomo che sta nel mezzo*

Mallory, l'uomo che sta nel mezzo, non solo può ascoltare i messaggi fra Alice e Bob, ma può anche modificarli, cancellarli ed introdurne di nuovi; egli può imitare Bob quando parla con Alice, o Alice quando comunica con Bob. Tale attacco funziona così:

1. Alice invia a Bob la propria chiave pubblica; Mallory la intercetta ed invia a Bob la propria;
2. Bob invia ad Alice la propria chiave pubblica; Mallory la intercetta ed invia ad Alice la propria;
3. quando Alice manda un messaggio a Bob, cifrato con quella che lei suppone la chiave pubblica di Bob, Mallory lo intercetta, lo decifra con la propria chiave privata, lo ricifra con la chiave pubblica di Bob e glielo invia;
4. quando Bob manda un messaggio ad Alice, cifrato con quella che lui suppone la chiave pubblica di Alice, Mallory lo intercetta, lo decifra con la propria chiave privata, lo ricifra con la chiave pubblica di Alice e lo manda a lei.

L'attacco dell'uomo che sta nel mezzo funziona perché Alice e Bob non hanno modo di verificare che stanno comunicando proprio l'uno con l'altro, non hanno cioè garanzie sull'autenticità dell'altro interlocutore. Se Mallory non provoca alcun fastidio al normale flusso della conversazione, come ritardi eccessivi fra una comunicazione e l'altra, Alice e Bob non hanno motivi per sospettare che un estraneo stia proprio fra di loro ed intercetti tutti i loro messaggi solo apparentemente segreti.

Il **protocollo di sincronizzazione**, inventato da Ron Rivest ed Adi Shamir, è un buon sistema per sventare l'attacco dell'uomo che sta nel mezzo; esso funziona così:

1. Alice invia a Bob la propria chiave pubblica;
2. Bob invia ad Alice la propria chiave pubblica;
3. Alice cifra il suo messaggio, utilizzando la chiave pubblica di Bob e gli invia soltanto metà del messaggio cifrato;
4. Bob cifra il suo messaggio, utilizzando la chiave pubblica di Alice e le invia soltanto metà del messaggio cifrato;
5. Alice manda l'altra metà del messaggio cifrato a Bob;
6. Bob mette insieme le due metà del messaggio di Alice e le decifra con la propria chiave privata; poi invia l'altra metà del suo messaggio cifrato ad Alice;
7. Alice mette insieme le due metà del messaggio di Bob e le decifra con la propria chiave privata.

Il punto cruciale di tale protocollo è che è inutile disporre solo di metà di un messaggio, in quanto esso non può essere decifrato: Bob può leggere il messaggio di Alice solo al sesto passaggio, mentre Alice rileva il messaggio di Bob solo all'ultimo step. Mallory può ancora intercettare le trasmissioni delle chiavi pubbliche fra i due interlocutori e sostituire entrambe con la propria, ma quando intercetta metà del messaggio di Alice al terzo passaggio, egli non

è più in grado di decifrarlo con la propria chiave privata e di ricifrarlo con la chiave pubblica di Bob: deve perciò inventare un nuovo messaggio ed inviarne metà a Bob. Gli si presenta lo stesso problema quando rileva la metà del messaggio di Bob. Quando finalmente egli riesce ad ottenere le seconde metà dei due messaggi agli step 5. e 6., è troppo tardi per sostituire i

nuovi messaggi che è stato costretto ad inventare: la conversazione fra Alice e Bob sarà completamente diversa ed il suo attacco verrà sventato.

Vediamo ora come funzionano alcuni protocolli di autenticazione degli interlocutori e scambio delle chiavi: il più semplice fra questi è il cosiddetto protocollo della **rana dalla bocca larga**, nel quale Alice e Bob condividono una chiave segreta con Trent. Tale condivisione occorre anche nel protocollo **Yahalom**, che funziona così:

1. Alice concatena il proprio nome, A , ed un numero casuale, R_A , ed invia il risultato a Bob; A, R_A
2. Bob concatena il nome di Alice, A , il numero casuale di Alice, R_A , il proprio numero casuale, R_B , cifra il risultato con la chiave che condivide con Trent (E_B) ed invia tutto a Trent, insieme al proprio nome, B ; $B, E_B(A, R_A, R_B)$
3. Trent genera due messaggi: il primo è costituito dal nome di Bob, B , una chiave di sessione casuale K , il numero casuale di Alice e quello di Bob, R_A ed R_B , tutti cifrati mediante la chiave che egli condivide con Alice, E_A ; il secondo è invece composto dal nome di Alice A e da K , cifrati con la chiave che egli condivide con Bob, E_B . Trent invia entrambi i messaggi ad Alice; $E_A(B, K, R_A, R_B), E_B(A, K)$
4. Alice decifra il primo messaggio, estrae K e conferma che R_A ha lo stesso valore che aveva nello step 1. Ella invia a Bob due messaggi: il primo è quello ricevuto da Trent, cifrato con la chiave di Bob; il secondo è R_B , cifrato con la chiave di sessione; $E_B(A, K), E_K(R_B)$
5. Bob decifra il messaggio cifrato con la propria chiave, estrae K e conferma che R_B ha lo stesso valore che aveva nello step 2.

Alla fine, Alice e Bob sono entrambi convinti di avere una conversazione proprio l'uno con l'altro e non con un intruso.

Esistono anche tipi di protocollo basati sulla crittografia a chiave pubblica: citiamo fra questi il DASS (Distributed Authentication Security Service) e il Denning-Sacco, nei quali Alice e Bob dispongono sia di una chiave pubblica che di una privata, mentre Trent possiede un database contenente tutte le chiavi pubbliche.

4.3.1 Crittografia a chiavi pubbliche multiple

Come già visto in precedenza, la crittografia a chiave pubblica utilizza due chiavi: con quella pubblica il messaggio viene cifrato e con quella privata esso viene decifrato. Così, Alice può mettere in codice un messaggio che solo Bob può decriptare e analogamente si comporta Bob. Consideriamo una variante che sfrutta tre chiavi anziché due: K_A , K_B e K_C , distribuite come mostrato nella **tabella 2**.

Alice può cifrare un messaggio con la chiave K_A , in modo che solo Ellen possa decifrarlo mediante le chiavi K_B e K_C . Analogamente, Bob mette in codice un messaggio che solo Frank può leggere e Dave è l'unico possibile lettore dei messaggi cifrati da Carol. Ancora, Dave

può cifrare un messaggio con K_A , in modo che solo Ellen possa decifrarlo o con K_B , in modo che solo Frank possa leggerlo, o anche con entrambe le chiavi K_A e K_B , in modo che solo Carol ne possa venire a capo. Analogamente, si trovano tutte le altre possibili combinazioni, riassunte nella **tabella 3**.

Alice	K_A
Bob	K_B
Carol	K_C
Dave	K_A e K_B
Ellen	K_B e K_C
Frank	K_C e K_A

Tabella 2

Un messaggio cifrato con le chiavi...	...deve essere decifrato con le chiavi
K_A	K_B e K_C
K_B	K_A e K_C
K_C	K_A e K_B
K_A e K_B	K_C
K_A e K_C	K_B
K_B e K_C	K_A

Tabella 3

Questo sistema può essere generalizzato al caso di n chiavi: se un dato sottoinsieme di chiavi di cardinalità k è utilizzato per cifrare un messaggio, sono necessarie le restanti $n - k$ per decifrarlo. Questo approccio è detto appunto **crittografia a chiavi pubbliche multiple**. Sempre limitandoci al caso di tre persone, per semplicità, possiamo distribuire ad Alice le chiavi K_A e K_B , a Bob le chiavi K_B e K_C ed a Carol le chiavi K_C e K_A . Ora è possibile comunicare con chiunque dei tre: se vogliamo ad esempio inviare un messaggio solo ad Alice, lo cifreremo con la chiave K_C , mentre se vogliamo comunicare uno stesso messaggio sia a Bob che a Carol, lo cifreremo mediante le chiavi K_A e K_B .

Ma vediamo in dettaglio l'algoritmo di un sistema crittografico a chiavi pubbliche multiple: si tratta di una generalizzazione dell'RSA, citato in precedenza, nel quale il modulo n è il prodotto di due numeri primi, p e q . Scegliamo t chiavi K_i , tali che

$$K_1 \times K_2 \times \dots \times K_t \equiv 1 \pmod{(p-1)(q-1)}$$

Detti M il messaggio da cifrare e C il messaggio cifrato, si avrà:

$$M^{K_1 \times K_2 \times \dots \times K_t} = C$$

e, supposto ad esempio $t = 5$, un messaggio criptato con le chiavi K_3 e K_5 può essere decifrato con le chiavi K_1 , K_2 e K_4 :

$$C = M^{K_3 \times K_5} \pmod{n}$$

$$M = C^{K_1 \times K_2 \times K_4} \pmod{n}$$

4.3.2 Secret sharing – condivisione dei segreti

La condivisione di un segreto ripartisce a più persone autorizzate delle 'porzioni' di esso, parti che, prese singolarmente, non hanno alcuna utilità, ma ricombinate insieme, consentono di riottenere il dato segreto originario.

Uno schema "m – su – n", noto come crittosistema di condivisione a soglia, decompone il segreto in n parti e le assegna ad n persone, in modo tale da consentire a qualsiasi insieme di sole m persone, con m strettamente minore di n , di ricostruire il segreto riunendo e ricombinando insieme le loro parti. Per esempio, uno schema "3 – su – 5" genera cinque porzioni di segreto: di queste, per creare l'originale, ne bastano solo tre, quelle di un qualsiasi gruppo di tre persone delle cinque che proteggono ognuna una parte del dato segreto spezzettato.

In questo modo, nel caso anche $(n - m)$ parti della chiave (il segreto) andassero distrutte, sarebbe ancora possibile recuperare il segreto condiviso. Inoltre, il nemico non avrebbe vita facile nella necessità di recuperare almeno m parti della chiave e non una singola copia di essa, per procacciarsi i dati illegalmente.

Uno schema di condivisione dei segreti è un sistema atto a garantire la massima sicurezza per quanto riguarda l'accesso a particolari strutture. Consideriamo il caso di una banca in cui lavorano tre cassieri e nella quale si trova una cassaforte. Ad essa è associata una combinazione segreta che ne permette l'apertura. Il principale per sicurezza non affida la combinazione della cassaforte ad uno solo di loro, ma ne assegna un pezzo a ciascuno, in maniera tale che se due cassieri uniscono le loro parti di chiave riescono a ricavare la combinazione per aprire la cassaforte (regola del "2 – su – 3"). In questo caso si è realizzato un sistema di condivisione di segreti, dove il segreto è la combinazione della cassaforte. Assegnando ad ogni cassiere una parte di essa, si evita il tentativo da parte di uno di essi di prelevare indebitamente denaro. Se invece due cassieri si mettono d'accordo, riescono nel loro intento.

❖ **Threshold scheme: definizione di uno schema a soglia (k, n)**

Siano k, n interi positivi tali che $k < n$. Uno schema a soglia (k, n) è un metodo di condivisione di un segreto S tra un insieme di n partecipanti, tali che:

- ❖ ogni gruppo di partecipanti di cardinalità $= k$ riesce a recuperare il segreto S ;
- ❖ ogni gruppo di partecipanti di cardinalità $< k$ non riesce a recuperare alcuna informazione riguardante il segreto S .

Il valore di S è scelto da un partecipante speciale chiamato "dealer".

Il dealer è denotato con D e si assume che D non appartenga a P che è l'insieme dei partecipanti, $P = \{P_i: 1 = i = n\}$.

Quando D vuole distribuire il segreto S fra i partecipanti in P , assegna ad ognuno di loro delle informazioni parziali, dette *share*. Queste vengono distribuite segretamente, in modo che nessun partecipante conosca le share date agli altri.

Per chiarire il concetto, consideriamo un sottoinsieme di partecipanti B di P , che uniscono le loro share per cercare di ricavare il segreto. Se $|B| = k$, allora essi riusciranno a calcolare il segreto S in funzione delle loro porzioni; se $|B| < k$ allora non riusciranno a ricavare alcuna informazione sul segreto S .

Un metodo per costruire uno schema a soglia (k, n) è quello presentato da Shamir nel 1979, chiamato appunto **schema a soglia di Shamir**.

❖ **Il calcolo del segreto – interpolazione polinomiale di Lagrange**

Vediamo come un sottoinsieme di k partecipanti può ricostruire la chiave, in uno schema a soglia (k, n) , costruito mediante equazioni polinomiali in un campo finito. Supponiamo che $P_{i1} \dots P_{ik}$ uniscano le loro share per determinare S . Le share sono $y_{ij} = a(x_{ij})$ per $1 = j = k$, dove $a(x)$ è il polinomio segreto di grado $k - 1$ a coefficienti in Z_p , con p numero primo, scelto dal dealer. Poiché $a(x)$ ha grado $< k$ può essere scritto come segue:

$$a(x) = (a_0 + a_1 x + a_2 x^2 + \dots + a_{k-1} x^{k-1}) \bmod p$$

dove $a_0, a_1, a_2, \dots, a_{k-1}$ sono elementi incogniti in Z_p , e $a_0 = S$ è il segreto.

Poiché $y_{ij} = a(x_{ij})$ per $1 = j = k$, i partecipanti ottengono un sistema di k equazioni in k incognite $a_0, a_1, a_2, \dots, a_{k-1} \in Z_p$. Se le equazioni sono linearmente indipendenti, esisterà un'unica soluzione ed il valore di a_0 rappresenterà il segreto. Il sistema delle equazioni lineari è il seguente:

$$\begin{cases} a_0 + a_1 x_{i_1} + \dots + a_{k-1} x_{i_1}^{k-1} = y_{i_1} \\ a_0 + a_1 x_{i_2} + \dots + a_{k-1} x_{i_2}^{k-1} = y_{i_2} \\ \vdots \\ a_0 + a_1 x_{i_k} + \dots + a_{k-1} x_{i_k}^{k-1} = y_{i_k} \end{cases}$$

Esso può essere scritto in forma matriciale come segue:

$$\begin{pmatrix} 1 & x_{i_1} & x_{i_1}^2 & \dots & x_{i_1}^{k-1} \\ 1 & x_{i_2} & x_{i_2}^2 & \dots & x_{i_2}^{k-1} \\ \vdots & \vdots & \vdots & \dots & \vdots \\ 1 & x_{i_k} & x_{i_k}^2 & \dots & x_{i_k}^{k-1} \end{pmatrix} \begin{pmatrix} a_0 \\ a_1 \\ \vdots \\ a_{k-1} \end{pmatrix} = \begin{pmatrix} y_{i_1} \\ y_{i_2} \\ \vdots \\ y_{i_k} \end{pmatrix}$$

Sia A la matrice $(k \times k)$ costruita, il cui determinante è:

$$\det(A) = \prod_{1 \leq i < t \leq k} (x_{it} - x_{ij}) \bmod p.$$

Ricordiamo che essendo tutte le x_{ij} distinte, ogni termine $(x_{it} - x_{ij}) \neq 0$. Il prodotto è calcolato in Z_p , che è un campo per cui il prodotto di termini diversi da zero non è zero e quindi $\det(A) \neq 0$. Essendo il determinante della matrice diverso da zero, il sistema ammette sempre un'unica soluzione. Questo dimostra che in uno schema a soglia, un gruppo di k partecipanti riuscirà a determinare il segreto S .

Vediamo cosa succede se un gruppo di $k - 1$ partecipanti $P_{i1} \dots P_{ik}$ si unisce nel tentativo di ricavare il segreto condiviso.

Procedendo come prima, si otterrà un sistema di $k - 1$ equazioni in k incognite

$$a_0, \dots, a_{k-1} \in Z_p.$$

Il sistema delle equazioni lineari è il seguente:

$$\begin{cases} a_0 + a_1 x_{i_1} + \dots + a_{k-1} x_{i_1}^{k-1} = y_{i_1} \\ a_0 + a_1 x_{i_2} + \dots + a_{k-1} x_{i_2}^{k-1} = y_{i_2} \\ \vdots \\ a_0 + a_1 x_{i_{k-1}} + \dots + a_{k-1} x_{i_{k-1}}^{k-1} = y_{i_{k-1}} \end{cases} \quad (1)$$

Poiché per ogni polinomio i termini a_1, a_2, \dots, a_{k-1} sono scelti dal dealer in modo casuale e solo a lui noti, segue che i $(k-1)$ partecipanti non riescono a valutare il valore y_0 . Supponiamo che essi ipotizzino un valore y_0 . Poiché il segreto è il termine costante del polinomio $a(x)$ ovvero $S = a_0$, si pone $y_0 = a_0$.

Sia $a_{y_0}(x)$ il polinomio $a(x)$ il cui termine noto è y_0 . Quindi $y_0 = a_{y_0}(0)$ rappresenta la k -esima equazione del sistema. Il risultante sistema è dato dalla (1) e dalla $y_0 = a_{y_0}(x)$. In tal modo il sistema risultante di k equazioni in k incognite ammette di nuovo un'unica soluzione. Per l'arbitrarietà della scelta di y_0 , qualsiasi numero intero potrebbe essere il segreto.

Ricordiamo che per ogni segreto esiste un unico polinomio, poiché il dealer sceglie per ogni segreto un'unica sequenza casuale di $k-1$ elementi a_1, a_2, \dots, a_{k-1} .

Esempio Supponiamo che $p = 17, k = 3$ e $n = 5$, i valori pubblici siano $x_i = i$, per $1 = i = 5$ ed il dealer abbia assegnato a P_1, P_2, P_3, P_4 e P_5 le seguenti share:

$$P_1 \leftarrow 8 = a(1)$$

$$P_2 \leftarrow 4 = a(2)$$

$$P_3 \leftarrow 10 = a(3)$$

$$P_4 \leftarrow 0 = a(4)$$

$$P_5 \leftarrow 11 = a(5)$$

Supponiamo che P_1, P_3 e P_5 vogliano ricavare il segreto condiviso fra i cinque partecipanti. Un polinomio calcolato dal dealer D di grado al più due è il seguente:

$$a(x) = (a_0 + a_1 x + a_2 x^2) \bmod p$$

Dati $a(1), a(3)$ e $a(5)$, essi ottengono le seguenti tre equazioni lineari in tre incognite in \mathbf{Z}_{17} :

$$a_0 + a_1 + a_2 = 8$$

$$a_0 + 3a_1 + 9a_2 = 10$$

$$a_0 + 5a_1 + 8a_2 = 11$$

Tale sistema ammette un'unica soluzione in \mathbf{Z}_{17} :

$$a_0 = 13, \quad a_1 = 10 \quad a_2 = 2.$$

Il segreto condiviso è quindi $S = a_0 = 13$.

Esistono anche schemi a soglia avanzati, nei quali una persona è più importante delle altre, perché dispone di diverse share. Se sono necessari cinque individui per ricreare un messaggio segreto ed uno fra questi ha tre share, mentre tutti gli altri ne hanno una soltanto, allora la persona privilegiata e sole altre due possono ricomporre il dato originario.

4.4 Protocolli intermedi

In molte situazioni, è necessario stabilire l'esatta cronologia delle azioni svolte nell'esecuzione di un'operazione; in ambito digitale, non c'è modo di capire se la data di registrazione di un dato è stata contraffatta o no: una data su un file può, infatti, essere copiata e modificata da chiunque e nessuno è in grado di stabilire il giorno esatto di creazione di un documento digitale. Per ovviare a tali violazioni, esistono degli opportuni protocolli di cronologia digitale con le seguenti proprietà:

- i dati stessi e non solo il documento, l'immagine o il file sonoro che li contengono, devono essere marchiati con la data di creazione;
- deve essere impossibile modificare un singolo bit del prodotto digitale in esame, senza che tale cambiamento risulti evidente;
- deve essere impossibile attribuire ad un documento una data ed un'ora diverse da quelle che già possiede.

Il classico protocollo comporta la presenza dell'arbitro Trent:

1. Alice, che vuole apporre su un proprio documento le corrette informazioni cronologiche, trasmette a Trent una copia del documento;
2. Trent registra la data e l'ora di ricezione del documento e conserva una copia di esso in custodia.

Questo protocollo funziona, ma presenta alcuni inconvenienti, come la totale assenza di privacy, a causa del fatto che Trent dispone di una copia del documento; Alice potrebbe ovviare a tale problema, cifrando il documento, che però sarebbe comunque conservato nel database di Trent, sulla cui sicurezza nessuno può fornire garanzie. Una soluzione è fornita dalla seguente variante del protocollo precedente:

1. Alice produce un'impronta digitale del documento (one-way hash) e la trasmette a Trent;
2. Trent vi appone la data e l'ora di arrivo e poi aggiunge una firma digitale al documento risultante;
3. Trent restituisce ad Alice il documento firmato munito dei dati relativi alla cronologia.

Questa variante risolve il problema della segretezza, in quanto Alice, inviando soltanto l'impronta del messaggio, non rivela nulla che riguardi il contenuto dei dati.

4.4.1 Messaggi subliminali

Supponiamo che Alice e Bob siano stati arrestati e condotti in prigione, in celle separate, e che ci sia un guardiano, Walter, che consente loro di scambiarsi messaggi, a patto che tale comunicazione sia in chiaro: infatti, attraverso messaggi cifrati, essi potrebbero organizzare l'eventuale fuga. I due detenuti allora devono costituire un canale subliminale, una via per scambiarsi informazioni segrete, senza che Walter sospetti che è in corso una conversazione clandestina. Un semplice tipo di scambio subliminale è il numero delle parole di una frase: un numero dispari di termini potrebbe corrispondere ad un "1", mentre un numero pari di essi

indicherebbe uno "0": si tratta però di una forma di steganografia triviale, non c'è una chiave e la sicurezza dipende solo dalla segretezza dell'algoritmo.

Gustavus Simmons introdusse il concetto di canale subliminale in una normale procedura di firma digitale: poiché i messaggi sono nascosti in quelle che sembrano convenzionali sigle digitali, ciò che appare agli occhi di Walter è un normale flusso di messaggi innocui firmati. In generale, il protocollo funziona così:

1. Alice genera un messaggio all'apparenza innocente;
2. utilizzando una chiave segreta condivisa con Bob, ella lo sigla in modo tale da nascondere il proprio messaggio subliminale nella firma;
3. Alice invia il messaggio a Bob tramite Walter;
4. Walter esamina il messaggio, verifica la firma apposta e, non trovando nulla di strano, lo recapita a Bob;
5. Bob verifica la firma, assicurandosi che il messaggio proviene da Alice;
6. Bob ignora il messaggio di copertura e, sfruttando la chiave segreta che condivide con Alice, estrae il messaggio subliminale.

Walter copre il ruolo di guardiano passivo, in quanto egli può sempre impedire che avvenga la comunicazione, ma non ha modo di introdurre messaggi fasulli, perché non è in grado di generare delle firme valide: Bob scoprirebbe l'inganno allo step 5., quando verifica l'autenticità della firma di Alice, per essere certo della provenienza del messaggio.

In alcune implementazioni di questo protocollo, l'informazione segreta di cui Bob necessita per leggere il messaggio subliminale, è la stessa di cui ha bisogno Alice per siglare il messaggio apparentemente innocuo. In tal caso, Bob può spacciarsi per Alice, firmare documenti come se provenissero da lei e non c'è nulla che Alice possa fare a riguardo, eccetto confidare che Bob non abusi della sua chiave privata.

In altre implementazioni, invece, questo problema non si pone: la chiave segreta condivisa da Alice e Bob le permette di inviargli messaggi subliminali, ma non coincide con la sua chiave privata e, per questo motivo, non abilita Bob a siglare messaggi.

Una comune applicazione dell'utilizzo di canali subliminali è nell'ambito dello spionaggio: se in una rete di comunicazione, tutti inviano e ricevono messaggi firmati, nessuno noterà l'inserzione di messaggi subliminali da parte di una spia in documenti siglati.

Inoltre, tramite un canale subliminale, è anche possibile comunicare uno stato di coercizione: Alice potrebbe firmare un documento inserendo il messaggio "Sono stata costretta" e rivelando così la sua posizione di pericolo.

Il canale subliminale progettato da Simmons utilizza lo schema di identificazione **Ong-Schnorr-Shamir**, una procedura di apposizione della firma basata sui polinomi modulo n . Alice sceglie un numero intero grande n , il modulo pubblico, del quale non è necessario conoscere la fattorizzazione, ed un intero casuale k , tali che k ed n siano primi fra loro; k è la chiave privata condivisa da Alice e Bob, il ricevente del messaggio subliminale. Calcoliamo la chiave pubblica in questo modo:

$$h = -k^2 \bmod n$$

Se Alice vuole inviare il messaggio subliminale M , tramite il messaggio innocuo M' , prima si accerta che sia M' ed n che M ed n siano primi fra loro, poi calcola:

$$S_1 = (1/2) \times (M' / M + M) \bmod n$$

$$S_2 = (k/2) \times (M' / M - M) \bmod n$$

La coppia (S_1, S_2) è la firma apposta mediante la procedura di Ong-Schnorr-Shamir ed è il mezzo in cui è incorporato il messaggio subliminale. Bob può verificare l'autenticità del messaggio: egli verifica che

$$S_1^2 - S_2^2 / k^2 \equiv M' \pmod{n}.$$

Se il messaggio è autentico, il ricevente può recuperare il messaggio subliminale utilizzando la seguente formula:

$$M = M' / (S_1 + S_2 k^{-1}) \bmod n.$$

Un secondo tipo di canale subliminale descritto da Simmons è basato sullo schema di apposizione di una firma **ElGamal**, che fonda la propria sicurezza sulla difficoltà di calcolo dei logaritmi discreti in un campo finito. Per generare una coppia di chiavi, scegliamo un numero primo p e due numeri casuali g ed r , tali che siano minori di p ; calcoliamo poi

$$K = g^r \bmod p;$$

K , g e p sono pubblici, la chiave privata è r , condivisa da Alice e Bob. Per inviare un messaggio subliminale M , tramite il messaggio innocuo M' , Alice controlla che M e p ed M e $p - 1$ siano primi fra loro, dopodiché calcola:

$$X = g^M \bmod p$$

e risolve la seguente equazione per Y , utilizzando l'algoritmo di Euclide esteso:

$$M' = rX + MY \bmod (p - 1)$$

La firma sarà costituita dalla coppia (X, Y) . Il guardiano Walter può verificare la firma ElGamal, controllando che

$$K^X X^Y \equiv g^{M'} \pmod{p}.$$

Bob può recuperare il messaggio subliminale, verificando che

$$(g^r)^X X^Y \equiv g^{M'} \pmod{p}$$

e calcolando poi:

$$M = (Y^{-1} (M' - rX)) \bmod (p - 1).$$

4.4.2 Calcolo con dati criptati

Analizziamo ora il **calcolo con dati criptati**: Alice vuole conoscere la soluzione di una funzione $f(x)$, per qualche particolare valore di x , ma il suo computer non funziona; Bob si offre allora di effettuare il calcolo in sua vece, ma ella non si fida a comunicargli il suo x : come può allora affidare il computo a Bob senza che lui sappia il valore di x ? Il problema si risolve con l'aiuto del logaritmo discreto: abbiamo un numero primo grande p ed un generatore g . Alice dispone di un particolare valore per x e vuole conoscere e , in modo che

$$g^e \equiv x \pmod{p}.$$

Le modalità di questo protocollo sono le seguenti:

1. Alice sceglie un numero casuale $r < p$;
2. ella calcola $x' = x g^r \text{ mod } p$;
3. chiede poi a Bob di risolvere l'equazione $g^{e'} \cdot x' \text{ (mod } p)$;
4. egli calcola e' e lo invia ad Alice;
5. Alice recupera e calcolando $e = (e' - r) \text{ mod } (p - 1)$.

4.4.3 Lancio della moneta

Questo protocollo funziona sia utilizzando la crittografia a chiave pubblica, che sfruttando quella simmetrica; l'unica richiesta è che l' algoritmo commuti, cioè che valga:

$$D_{K1}(E_{K2}(E_{K1}(M))) = E_{K2}(M).$$

Analizziamo ora gli step del protocollo:

1. Alice e Bob generano entrambi una coppia di chiavi (pubblica e privata);
 $(E_i, D_i), i = A, B$
2. Alice genera due messaggi: uno che indica "testa", l'altro "croce" – questi messaggi contengono una stringa casuale unica, in modo che ella ne possa verificare l'autenticità negli ultimi step del protocollo – li cifra entrambi con la propria chiave pubblica e li invia a Bob in ordine casuale;
 $E_A(M_1), E_A(M_2)$
3. Bob, che non può leggere alcun messaggio, ne sceglie uno a caso, lo cifra con la propria chiave pubblica e lo restituisce ad Alice;
 $E_B(E_A(M)), M = M_1, M_2$
4. Alice, che non può leggere il messaggio che Bob le ha reinviato, lo decifra con la propria chiave privata e lo rimanda a Bob;

$$D_A(E_B(E_A(M))) = E_B(M_1) \text{ se } M = M_1$$

$$D_A(E_B(E_A(M))) = E_B(M_2) \text{ se } M = M_2$$

5. allora Bob decifra il messaggio con la propria chiave privata per scoprire il risultato del lancio della moneta e manda l'esito ad Alice;

$$D_B(E_B(M_1)) = M_1 \quad D_B(E_B(M_2)) = M_2$$

6. Alice esamina il risultato e verifica che la stringa casuale sia corretta;
7. entrambi rivelano le loro coppie di chiavi, in modo che ciascuno possa controllare che l'altro non abbia barato.

Questo protocollo appartiene alla categoria dei self-enforcing: ogni partecipante può immediatamente scoprire se l'altro non segue le regole, e non è necessaria la presenza di una terza persona di fiducia che completi il protocollo o ne garantisca la limpidezza nell'esecuzione.

4.4.4 Poker

Vediamo ora come possono tre persone giocare a poker per via elettronica, senza che nessuno abbia la possibilità di imbrogliare gli altri: il protocollo seguente, relativo ad una mano di poker che coinvolge tre giocatori, può essere esteso a più persone. Come nel caso precedente, l'algoritmo crittografico deve essere commutativo.

1. Alice, Bob e Carol generano una coppia di chiavi ciascuno, costituita da una chiave pubblica e da una privata; $(E_i, D_i), i = A, B, C$
2. Alice genera 52 messaggi, uno per ogni carta del mazzo (ciascuno contenente un'unica stringa casuale, in modo che ella ne possa verificare l'autenticità negli ultimi step del protocollo) e cifra tutti i messaggi con la propria chiave pubblica, inviandoli successivamente a Bob; $E_A(M_n)$
3. Bob, che non può leggere nessuno dei messaggi, ne sceglie cinque a caso e li cifra con la propria chiave pubblica, rimandandoli indietro ad Alice; $E_B(E_A(M_n))$
4. egli invia poi i restanti 47 messaggi a Carol; $E_A(M_n)$
5. Carol, che non è in grado di leggere nessuno dei messaggi, ne sceglie cinque a caso e li cifra con la propria chiave pubblica, rimandandoli indietro ad Alice; $E_C(E_A(M_n))$
6. Alice, che non può leggere nessuno dei messaggi che le sono stati restituiti, li decifra con la propria chiave privata e poi li rinvia a Bob ed a Carol, rispettivamente;
 $D_A(E_B(E_A(M_n))) = E_B(M_n)$ $D_A(E_C(E_A(M_n))) = E_C(M_n)$
7. Bob e Carol hanno finalmente a disposizione le loro carte, decifrando i messaggi con le loro chiavi private; $D_B(E_B(M_n)) = M_n$ $D_C(E_C(M_n)) = M_n$
8. Carol sceglie cinque ulteriori messaggi fra i restanti 42 e li invia ad Alice; $E_A(M_n)$
9. Alice decifra tali messaggi con la propria chiave privata, ottenendo le carte per poter giocare; $D_A(E_A(M_n)) = M_n$
10. alla fine del gioco Alice, Bob e Carol rivelano le loro carte e loro chiavi, in modo che ciascuno possa controllare che nessuno abbia barato.

Le carte che vengono estratte durante il gioco possono essere assegnate nella stessa maniera. In teoria, l'esecuzione dell'ultimo step non è necessaria e, comunque, non tutti i giocatori dovrebbero essere costretti a scoprire le loro carte: esso ha, infatti, l'unico scopo di individuare eventuali bari, ma ciò che più conta è controllare che il vincitore non abbia imbrogliato. Chiunque altro abbia barato, senza per questo vincere la partita, ha commesso un inganno non rilevante.

4.4.5 "All-Or-Nothing" (Tutto O Niente)

Supponiamo che Alice sia un'agente segreta disoccupata che ha accettato di vendere segreti in suo possesso per guadagnarsi da vivere: chiunque sia disposto a pagare il giusto prezzo può allora ottenere da lei informazioni riservate. Tutti i suoi dati sono immagazzinati in un database, numerati e classificati con etichette allettanti, del tipo "Dove si trova Jimmy Hoffman?", "Chi sta segretamente controllando le disposizioni della Commissione Trilaterale?", ed altre.

Alice deve stare attenta a non rivelare due segreti al prezzo di uno e non deve lasciarsi sfuggire informazioni anche solo parziali riguardanti altri dati privati; d'altro canto, Bob, un potenziale acquirente, non vuole pagare per ottenere dati casuali e vuole assicurarsi che Alice non venga a conoscenza dei segreti di cui egli necessita.

La soluzione è fornita dai protocolli **ANDOS (All-or-Nothing Disclosure Of Secrets)**, rivelazione dei segreti secondo la modalità "tutto o niente", così denominati perché appena Bob ha guadagnato qualsiasi tipo di informazioni inerenti ad uno dei segreti di Alice, egli ha sciupato ogni possibilità di ottenere altri dati riservati. Ma vediamo nei dettagli come funziona questo tipo di protocollo, che permette a diverse persone di acquistare segreti individuali da un unico venditore. Date due stringhe di bit, x ed y , si dice FBI (Fixed Bit Index, indice di bit fissati) di x ed y l'insieme di tutti i bit tali che l' i -esimo bit di x è uguale all' i -esimo bit di y . Ad esempio, se

$$x = 110101001011$$

$$y = 101010000110$$

$FBI(x, y) = \{1, 4, 5, 11\}$ (leggendo i bit da destra verso sinistra e considerando il bit all'estrema destra come lo zeresimo).

Nel seguente protocollo, Alice è la venditrice e possiede k segreti di n bit, S_1, S_2, \dots, S_k . Bob e Carol sono gli acquirenti: lui vuole acquistare il segreto S_b , mentre a lei interessa S_c .

1. Alice genera una coppia di chiavi (una pubblica ed una privata) e comunica a Bob (ma non a Carol) quella pubblica; poi genera un'altra coppia di chiavi (di nuovo, pubblica e privata) e riferisce a Carol (ma non a Bob) quella pubblica.
2. Bob genera k numeri casuali di n bit, B_1, B_2, \dots, B_k e li comunica a Carol, che esegue un'operazione identica, generando k numeri casuali di n bit, C_1, C_2, \dots, C_k e riferendoli a Bob.
3. Bob cifra C_b (S_b è il segreto di cui vuole impossessarsi) mediante la chiave pubblica che gli ha dato Alice, calcola l'FBI di C_b e del risultato che ha appena cifrato ed invia tale valore a Carol; analogamente, quest'ultima cifra B_c (S_c è il segreto che vuole acquistare) mediante la chiave pubblica che le ha dato Alice, calcola l'FBI di B_c e del risultato che ha appena criptato ed invia tale valore a Bob.
4. Bob esamina ciascuno dei numeri ad n bit, B_1, B_2, \dots, B_k , e sostituisce ogni bit il cui indice non è nell'FBI che ha ricevuto da Carol con il suo complementare; infine, invia la nuova lista di numeri ad n bit B'_1, B'_2, \dots, B'_k ad Alice. Analogamente, Carol considera ciascuno dei numeri ad n bit, C_1, C_2, \dots, C_k , e sostituisce ogni bit il cui indice non è nell'FBI che ha ricevuto da Bob con il suo complementare; manda infine la nuova lista di numeri ad n bit C'_1, C'_2, \dots, C'_k ad Alice.
5. Alice decifra tutti i C'_i con la chiave privata di Bob, ottenendo i propri k numeri ad n bit: $C''_1, C''_2, \dots, C''_k$; calcola poi $S_i \oplus C''_i$, per $i = 1 \dots k$ ed invia i risultati a Bob. Poi ella decifra tutti i B'_i con la chiave privata di Carol, ricavando i propri k numeri ad n bit: $B''_1, B''_2, \dots, B''_k$; calcola poi $S_i \oplus B''_i$, per $i = 1 \dots k$ ed invia i risultati a Carol.
6. Bob calcola S_b tramite l'operazione di XOR fra C_b ed il b -esimo numero che ha ricevuto da Alice; analogamente, Carol calcola S_c tramite l'operazione di XOR fra B_c ed il c -esimo numero che ha ricevuto da Alice.

Sfortunatamente, questo protocollo non è completamente sicuro, in quanto se Alice e Carol colludono, possono facilmente scoprire di quale segreto vorrebbe impossessarsi Bob: infatti, conoscendo l'FBI di C_b e l'algoritmo di cifratura di Bob, le due donne sono in grado di ricavare b tale che C_b abbia il corretto FBI. Anche Bob e Carol, mettendo insieme le loro conoscenze, possono ottenere tutti i segreti di Alice.

4.5 Protocolli avanzati

In alcune situazioni si verifica una circostanza curiosa: se Alice è a conoscenza di un segreto, ma nessuno le crede, l'unico modo che ella ha di dimostrare la propria buona fede è rivelare il segreto stesso, mettendo così tutti gli ascoltatori sul suo stesso piano, involontariamente: così, ora tutti conoscono le informazioni riservate che prima erano solo di sua proprietà. Per non cadere in questo trabocchetto, Alice può sfruttare le cosiddette **zero-knowledge proof**, o dimostrazioni senza rivelazioni, che coinvolgono Peggy, colei che conosce il segreto e può dimostrarlo e Victor, colui che deve verificare, mediante opportuni quesiti, che Peggy non stia bluffando. Si tratta di protocolli interattivi, nei quali Victor pone a Peggy una serie di domande: se ella davvero è in possesso delle informazioni riservate, è in grado di rispondere correttamente; se invece sta mentendo, ha il 50% di probabilità di fornire risposte esatte. Il punto cruciale è che, dopo un certo numero di quesiti, Victor è in grado di stabilire se Peggy conosce o no il segreto, ma non ha guadagnato alcun elemento che possa condurlo al segreto stesso.

Analizziamo in dettaglio il protocollo di base, partendo dalla storia della cava. La cava illustrata in **fig. 4.5.a**, nasconde un segreto: chiunque conosca le parole magiche può aprire la porta segreta fra il punto C ed il punto D. Supponiamo che Peggy sia in possesso della formula magica e che voglia dimostrarlo a Victor, senza però rivelargliela.

1. Victor è fermo al punto A;
2. Peggy cammina per la cava fino a raggiungere il punto C oppure il punto D;
3. dopo che Peggy è scomparsa nella cava, Victor raggiunge il punto B;
4. egli le chiede di raggiungerlo
 - a. usando il passaggio di sinistra;
 - b. usando il passaggio di destra;
5. ella esegue, se necessario superando la porta segreta, se conosce le parole magiche;
6. Peggy e Victor ripetono gli step da 1. a 5. n volte, dopo le quali Victor può stabilire se ella è davvero a conoscenza del segreto oppure no.

Questo protocollo funziona perché è impossibile che Peggy possa ripetutamente indovinare da quale passaggio Victor le chiederà di uscire e, se davvero non conosce il segreto, la sua probabilità di successo si riduce in modo esponenziale al crescere del numero di prove. Se le disposizioni del protocollo precedente vengono eseguite una volta sola e Peggy non conosce le parole magiche, ella ha una probabilità del 50% di cavarsela; se la verifica viene fatta due volte, le possibilità di successo si riducono al 25%. In generale, la probabilità che la ragazza ha di beffare Victor in n volte è $1/2^n$. Se egli vuole ripetere il test 16 volte, Peggy ha $1/65536$ possibilità di ingannarlo: se quindi porta a termine con successo tutte le 16 verifiche, egli ha la certezza che la ragazza conosca davvero il segreto.

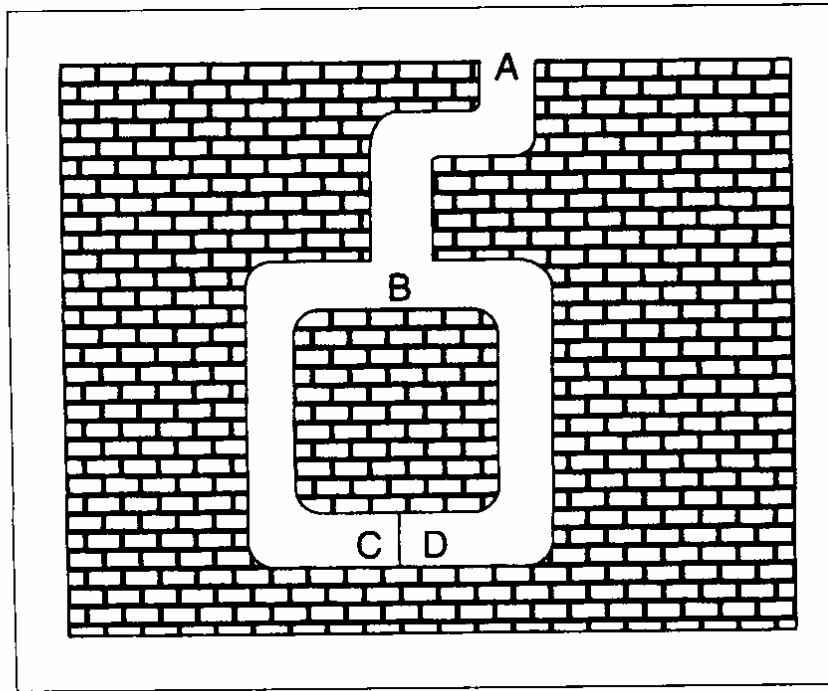


Fig. 4.5.a – la cava

4.5.1 Verifiche di identità

Nel mondo reale, spesso usiamo elementi fisici come prove di identità: i passaporti, le patenti, le carte d'identità e le carte di credito contengono o una foto o una firma o entrambe, ma potrebbe trattarsi dell'impronta digitale di un pollice, della scansione della retina o ancora di una panoramica dentale a raggi X. L'idea di utilizzare le zero-knowledge proof come prove di identità è stata inizialmente proposta Uriel Feige, Amos Fiat ed Adi Shamir: la chiave privata di Alice diventa una funzione della sua identità. Ma vediamo in dettaglio l'algoritmo.

Peggy vorrebbe dimostrare a Victor di conoscere un intero x che soddisfi

$$A^x \equiv B \pmod{p}$$

dove p è un numero primo ed x è un numero casuale relativamente primo con $p - 1$; i numeri A , B e p sono pubblici ed x è segreto. Qui di seguito forniamo i passaggi del protocollo, grazie al quale Peggy può dimostrare di conoscere x senza rivelarlo:

1. Peggy genera t numeri casuali, r_1, r_2, \dots, r_t , dove tutti gli r_i sono minori di $p - 1$;
2. ella calcola $h_i \equiv A^{r_i} \pmod{p}$ per ogni valore di i e li invia a Victor;
3. Peggy e Victor realizzano un protocollo del lancio della moneta per generare t bit: b_1, b_2, \dots, b_t ;
4. per ognuno dei t bit, Peggy esegue le seguenti istruzioni:
 - a. se $b_i = 0$, ella invia a Victor r_i ;
 - b. se $b_i = 1$, ella gli invia $s_i = (r_i - r_j) \pmod{p - 1}$, dove j è il minimo valore per il quale $b_j = 1$.

5. per ognuno dei t bit, Victor verifica
 - a. che $A^{r_i} \equiv h_i \pmod{p}$, se $b_i = 0$;
 - b. che $A^{s_i} \equiv h_i h_j^{-1} \pmod{p}$, se $b_i = 1$;
6. Peggy invia a Victor Z , dove $Z = (x - r_j) \pmod{p-1}$;
7. Victor controlla infine che $A^Z \equiv B h_j^{-1} \pmod{p}$.

La probabilità per Peggy di riuscire ad ingannare Victor è 2^{-t} .

4.5.2 Firma simultanea di un contratto

Esistono diversi modi di stipulare un contratto fra due persone: alcuni di questi non includono la presenza di un “arbitro” e comportano la firma del documento in cui i due contraenti si trovano faccia a faccia, in cui essi sono lontani (telefirma) ed in cui si affidano alla crittografia; l’ultimo, che analizziamo in dettaglio, coinvolge una terza persona di fiducia che supervisioni l’esecuzione del protocollo.

- ❖ **firma faccia a faccia, in assenza di arbitro;**
- ❖ **telefirma, in assenza di arbitro;**
- ❖ **firma crittografica, in assenza di arbitro;**
- ❖ **firma in presenza di un “arbitro”.**

Supponiamo che Alice e Bob vogliano sottoscrivere un contratto: essi concordano sulla formulazione dello stesso, ma nessuno dei due vuole apporre la propria firma prima che l’abbia fatto l’altro. Se si trovano faccia a faccia, possono siglare il documento contemporaneamente, mentre se fra loro esiste una certa distanza, necessitano di una terza persona di fiducia.

1. Alice firma una copia del contratto e la invia a Trent;
2. Bob firma una copia del contratto e la invia a Trent;
3. Trent invia un messaggio sia ad Alice che a Bob per comunicare che l’altro contraente ha firmato il documento;
4. Alice firma due copie del contratto e le invia a Bob;
5. Bob firma entrambe le copie, ne tiene una per sé ed invia l’altra ad Alice;
6. Alice e Bob informano Trent che ora possiedono ciascuno una copia del contratto firmata da entrambi;
7. Trent si libera delle due copie del contratto recanti una firma soltanto.

Questo protocollo funziona, perché Trent impedisce a ciascuna delle due parti di imbrogliare: infatti, se Bob dovesse rifiutarsi di siglare il contratto allo step 5., Alice potrebbe far ricorso a Trent che ne possiede una copia firmata in precedenza proprio da Bob; in modo analogo si comporterebbe Bob se fosse Alice a cambiare idea. Se Trent dovesse ricevere uno solo dei contratti siglati nei primi due step, egli lo straccerebbe eliminando il vincolo del contraente che aveva firmato.

CONCLUSIONE

In questa analisi critto-steganografica, abbiamo analizzato numerose tecniche di inserzione dei dati in diversi tipi di mezzi digitali, partendo dalla compressione dei dati, e valutando il contributo dato all'aumento della casualità ed al conseguente incremento dell'entropia dei file. Abbiamo approfondito algoritmi di compressione testuale, come Huffman, LZW, RLE e GZSteg, metodi di compressione per i formati grafici, tipo GIF, JPEG, JPEG2000, PNG e BMP, ed infine procedure di compressione dei formati audio, fra i quali spicca l'MP3.

In seguito, abbiamo visto come è possibile che informazioni segrete si dissolvano all'interno di un'immagine o di un file sonoro, imitando un pixel o una nota ed adattandovisi perfettamente, procurando al file originale un degrado che rimane sempre al di sotto della percezione umana, sia visiva che uditiva. Abbiamo anche volto brevemente l'attenzione verso tecniche non propriamente steganografiche, ma che meritavano un cenno, quali l'occultamento di messaggi satanici nel rock'n'roll, la tecnica dei Photo-Tile, l'ASCII art ed il cosiddetto "primo illegale".

Successivamente, ci siamo occupati del settore di maggiore applicazione delle procedure steganografiche, quello del copyright digitale. Le richieste per algoritmi di apposizione di un marchio su un prodotto digitale – watermarking e fingerprinting – che garantiscano il rispetto dei diritti d'autore, sono in continuo aumento e, spesso, le aziende che forniscono i pacchetti applicativi di marchiatura sono le stesse che producono i software che li invalidano.

Infine, abbiamo studiato alcuni tipi di protocolli, per assicurare che uno scambio di informazioni tra due o più persone, precedente o successivo all'applicazione di metodi steganografici, sia limpido e non consenta a nessuno fra gli individui coinvolti di carpire dati al cui accesso non è autorizzato. Esistono diversi tipi di protocollo: di base, intermedi ed avanzati, tutti caratterizzati da accorgimenti ingegnosi al fine della conservazione integra di un segreto.

Le procedure di steganografia ed i protocolli non standard analizzati in questa sede rappresentano dunque uno dei campi di sviluppo nel futuro delle comunicazioni. La sua diretta connessione con la matematica ed in particolare con l'algebra è però un'arma a doppio taglio: infatti, se da un lato il supporto matematico è stato fondamentale ai fini dell'elaborazione di algoritmi critto-steganografici, dall'altro è sempre ad un passo dall'invalidamento degli stessi, grazie allo sfruttamento di debolezze di fondo, che prima o poi, vengono individuate. Questo meccanismo a fisarmonica può risultare ostico per chi ha sempre considerato la matematica una disciplina certa, statica, inviolabile: in realtà, ogni applicazione pratica di una certezza astratta non può che incappare nei margini di errore causati dall'uso massiccio del calcolatore, ad esempio, per cui l'individuazione di dati nascosti in un'immagine visualizzata sul video di un computer potrebbe dipendere semplicemente dal livello di risoluzione fornito dalla marca del monitor. Si tratta perciò di una forma di progresso infinito: ogni nuova tecnica, per quanto complicata ed arricchita dalle caratteristiche delle più recenti tecnologie, anche se apparentemente inaccessibile, verrà violata e consentirà così uno sviluppo sempiterno del settore.

Ed ancora una volta non possiamo che apprezzare la matematica e qualificarla come disciplina sublime: inviolabile in teoria, garante di un progresso che non avrà mai fine, in pratica.

Bibliografia e siti Internet visitati

“Crittografia invisibile – Informazioni segrete”, Peter Wayner, Mc Graw Hill, maggio 1997.

“On The Limits of Steganography”, Ross J. Anderson, Fabien A. P. Petitcolas, *IEEE Journal of Selected Areas in Communications*, vol. 16, no. 4, maggio 1998, pp. 474-482.

“Stretching the Limits of Steganography”, Ross Anderson.

“Applied Cryptography – Protocols, algorithms and Source Code in C”, Bruce Schneier (second edition), John Wiley & Sons 1996.

“Attacks on Copyright Marking Systems”, Fabien A. P. Petitcolas, Ross J. Anderson, Markus G. Kuhn, *Second workshop on information hiding*, in vol. 1525 of *Lecture Notes in Computer Science*, Portland, Oregon, USA, 14-17 April 1998, pp. 218-238.

“Steganography – past, present, future”, James C. Judge, November 30th, 2001

“Steganography – a Cryptography Layer”, Vlad Rabinovich, October 26th, 1999;
<http://www.rit.edu/~vrx8205/crypto2/cryptopaper.html>.

“Cracking a Medieval Code”, Ivars Peterson;
http://www.maa.org/mathland/mathtrek_5_4_98.html.

“Steganalysis of Images Created Using Current Steganography Software”, Neil F. Johnson, Sushil Jajodia; <http://www.jitc.com/ihws98/jjgmu.html>.

“Techniques for data-hiding”, Walter Bender, Daniel Gruhl, Norishige Morimoto, Anthony Lu; *IBM System Journal* vol. 35, No. 3&4 MIT Media Lab 1996, pp. 313-336;
<http://www.research.ibm.com/journal/sj/353/sectiona/bender.html>

“Steganography Wing of the Gallery of CSS Descramblers”;
<http://www-2.cs.cmu.edu/~dst/DeCSS/Gallery/SteGo/index.html>.

“Steganografia”, Frank Sinapsi; <http://www.kyuzz.org/anon/steGo.html>

<http://www.diodati.org/scritti/2001/algoritmi/index.asp>

<http://www.cs.sfu.ca/CC/365/li/squeeze/Huffman.html>

<http://www.cs.sfu.ca/CC/365/li/squeeze/LZW.html>

<http://www.luratech.com/>

<http://gailly.net/>

<http://www.alumni.caltech.edu/~madler/>

<http://www.mclink.it>

<http://www-lia.deis.unibo.it/Courses/RetiDiCalcolatori/Progetti00/fortini/project.pdf>

<http://www.cs.uct.ac.za/courses/CS400W/NIS/papers99/dsellars/stego.html>

<http://www.sans.org/rr/steg/steganography4.php>

<http://www.cl.cam.ac.uk/~fapp2/steganography>

http://www.cl.cam.ac.uk/~fapp2/steganography/stego_soft.html

<http://www.ecn.org/crypto/soft/stego.phtml>

<http://ip-service.com/cgi-bin/stego.pl>

www.compris.com/TextHide/en/Overview.html

www.imsa.edu/~keithw/tlex

www.ctgi.net/nicetext/

www.Stegoarchive.com

www.cs.vu.nl/~ast/books/mos2/zebras.html

<http://www.tiac.net/users/korejwa/jsteg.htm>

<ftp://ftp.funet.fi/pub/crypt/steganography>

<http://www.darkside.com.au/gifshuffle>

www.neobytesolutions.com/invsecr

<http://www.webclub.com/tte>

<http://www.cl.cam.ac.uk/~fapp2/steganography/mp3stego/index.html>

<http://www.prismaticsoftware.com/Phototile/PhotoTile.html>

<http://go.to.ascgen>

<http://syscop.igd.fhg.de/>

<http://www.digimarc.com>

<http://www.alphatecltd.com/>

<http://www.signumtech.com/>

<http://www.bluespike.com/giovanni/gdigmark.html>

<http://primes.utm.edu>

Ringraziamenti

Ringrazio il relatore, professor Umberto Cerruti, che mi ha supportato con preziosi consigli, nella ricerca del materiale idoneo alla stesura della tesi, per la Sua solerzia e disponibilità.

I miei ringraziamenti vanno inoltre a parenti ed amici, per avermi accompagnato e sostenuto in ogni momento, gioioso e non, di questa avventura universitaria.

Un ringraziamento particolare va infine al dottor Giancarlo Ramat, che, pur avendo subito recentemente una grave perdita familiare, mi ha fornito materiale utile per apportare ulteriori miglioramenti al lavoro svolto ed un aiuto morale sempre costante.

