Computer and Network Security

Lorenzo Rossi

December 29, 2021

Contents

Part I

Third Midterm

Chapter 1

Secret Sharing

1.1 Trivial Secret Sharing

Supponiamo di avere un segreto e vogliamo dividerne la conoscenza in due persone (dette shareholders). Inoltre, vogliamo si viene a conoscenza del segreto se e solo se entrambe le parti rivelano la loro porzione di segreto. Chi

fornisce il segreto viene detto **dealer**, mentre chi riceve le porzioni del segreto sono detti **share**.
Nel caso in cui avessimo diviso il segreto in parti uguali, è una pessima idea poiché per indovinare il segreto abbiamo $\frac{1}{2^{N_{bit}}}$ probabilità di indovinare la password ed ora, avendo diviso il segreto in parti uguali, abbiamo una probabilità molto maggiore $\frac{1}{2^{\frac{N_{bit}}{2}}}$.

1.1.1 XOR Secret Sharing

Possiamo fare di meglio:

1. Prendi il segreto i.e.0010.1101;
2. Genera una sequenza casuale **key** i.e.1011.0100;
3. XOR il segreto e il valore casuale **one time pad** i.e.1001.1001;
   Fino ad ora abbiamo applicato un *Vernam cipher*.
4. Diamo ad uno share la sequenza casuale, mentre ad un altro diamo il valore dello XOR;
5. L'unione fra gli share da la chiave.

Importante. *Il conoscere la chiave, cioè il valore casuale, non mi da alcuna informazione riguardante la chiave.*
Lo stesso discorso vale per il valore dello XOR poiché, come dimostrato nel ***perfect secrecy****, l'operatore di XOR tra una stringa pseudocasuale e un valore casuale non da informazioni su quale sia la password.*
Questi due aspetti rappresentano un requisito di sicurezza.

1.1.2 Modular Secret Sharing

Un altro possibile schema è quello di utilizzare le somme modulari:

1. Prendi il segreto S in bit, trasformalo in digit i.e.0010.1101$rightarrow$45;
2. Genera $RAND \mod N$ i.e.$RAND \mod 256 \rightarrow 180$;
3. Esegui $S - RAND \mod N$ i.e. $S - RAND \mod 256 \rightarrow 121$;

Importante. *Questo schema è equivalente ad One Time Pad poiché abbiamo sommato un numero pseudocasuale con un numero casuare (in modulo). In altre parole, la probabilità di indovinare S conoscendo il valore casuale o il valore della somma è uguale alla probabilità di indovinare senza sapere nulla.*

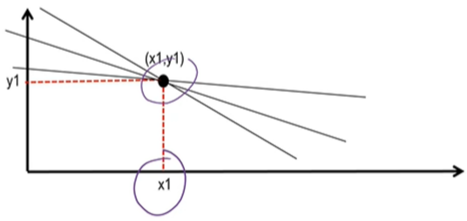
Questo metodo è più facile da implementare per essere condiviso con N shareholders. In particolare, genero 3 quantità truly random ed effettua la differenza tra il segreto e queste 3 quantità modulo N. Nel caso un attacker, riuscisse ad ottenere un numero sufficiente di share non può comunque ottenere la password, ma al più la differenza tra il segreto e le shares non prese.
Da qui è possibile definire il concetto di **perfect secrecy**: un avversario, conoscendo n-1 shares deve ancora possedere la probabilità di indovinare il segreto pari a quella di indovinare il segreto da zero.

1.2 Shamir Secret Sharing

Fino ad ora abbiamo costruito uno schema detto (n,n) secret sharing scheme in cui il primo parametro è il numero delle persone necessarie a rilevare il segreto e il secondo parametor è il numero di parti:il segreto viene rilevato solo se tutte le n parti forniscono il segreto.
Un altro schema è (t,n) secret sharing scheme:il segreto è rilevato quando qualsiasi t delle n parti fornisce il segreto. Questo secondo problema è molto più complicato del trivial secret sharing.

1.2.1 Idea:Schema (2,n)

Il problema è quello di modellare uno schema per cui, conoscendo 2 degli n shareholders, posso ricostruire il segreto. Questo problema è riconducibile a quello di conoscere quanti punti sono necessari per definire una linea:ovviamente 2. Infatti conoscendo un solo punto (shares) ho infinite rette passanti per quel punto e quindi è impossibile ricondurci



al segreto;tuttavia, conoscendo 2 punti (shares), tra essi passa solamente una sola retta e conseguentemente posso conoscere il segreto. Abbiamo comunque mantenuto la proprietà di poter avere un numero maggiore di 2 per ottenere il segreto, ma al minimo sono 2.

1.2.2 Procedura: Schema$(2, n)$

- **Dealer:** costruisce la linea:
  1. Coefficiente a:scelto casualemnte;
  2. Segreto S:noto;

  $$y = S + ax$$

  Per esempio: $a = 15 \quad S = 39$
- Distribuisci le shares ai n partecipanti scegliendo casualmente il valore x_i da introdurre nell'equazione della retta:
  - Shareholder 1: $x_1 = 1 \rightarrow share = (1, 54)$;
  - Shareholder 2: $x_2 = 2 \rightarrow share = (2, 69)$;
  - Shareholder 3: $x_3 = 3 \rightarrow share = (3, 84)$;
  - ...

Importante. *La y viene calcolata in base alla funzione della retta;tuttavia, i punti degli shareholder sono mantenuti con (x,y) e il valore delle x_i possono essere noti a priori a patto che la y sia nascosta.*

1.2.3 Procedura: Ricostruzione (2,n)

- Ricezione di due shares:$P_i = (x_i, y_i) \quad P_j = (x_j, y_j)$;
- Interpola i punti per ricostruire l'equazione della retta:

$$\frac{y - y_i}{y_i - y_j} = \frac{x - x_i}{x_i - x_j}$$

  Ottenendo:

$$y = y_i + \frac{x - x_i}{x_i - x_j}(y_i - y_j)$$

- Bisogna sostituire $x = 0$ per ottenere il segreto $y = S$;

1.2.4 Estensione al caso (t, n)

Estendendo il discorso precedentemente introdotto, ci si riconduce al caso di polinomi di grado $t-1$ unicamente definiti da t punti:

- Linea:2 punti;
- Parabola (quadratic):3 punti;
- Cubiche:4 punti;
- ...

1.2.5 Generalizzazione: Schema (t, n)

- **Dealer**:
  1. Genera un polinomio casuale $p(x)$ di grado $t - 1$;
  2. Imposta il segreto s come il termine noto del polinomio:

$$p(x) = s + a_1x + a_2x^2 + \cdots + a_{t-2}x^{t-2} + a_{t-1}x^t - 1$$

     con s il segreto e i coefficienti delle x generati truly random;
  3. Distribuisci uno share ad ogni shareholders:

$$(x_i, y_i) \rightarrow y_i = p(x_i)$$

- **Ricostruzione**: Colleziona t shares su n disponibili e calcola il segreto utilizzando *l'Interporlazione di Lagrange* con $x = 0$:

$$s = \sum_{shares\ x_i} y_i\Lambda_{x_i} \quad with \quad \Lambda_{x_i} = \Lambda_{x_i}(0) = \prod_{shares\ x_k \neq x_j} \frac{-x_k}{x_i - x_k}$$

  L'interpolazione di Lagrange si basa sul concetto che qualsiasi polinomio di grando t-1 con t punti noti, può essere decomposto come:

$$y = \sum_{i=1}^{t} y_i\Lambda_i(x)$$

  In cui $\Lambda_i(x)$ è la base del polinomio calcolata come:

$$\Lambda_i(x) = \prod_{m=1, m\neq 1}^{l} \frac{x - x_m}{x_i - x_m} \quad \Lambda_i(x_i) = 1; \quad \Lambda_i(x_m) = 0 \quad for\ m \neq i$$

1.2.6 Segretezza

Per discutere di quanto sia sicuro questo schema dobbiamo ricordare che in questo ambito la segretezza è così definita:

Finché si conoscono (t-1) shares non si dovrebbe avere nessuna informazione sul segreto che stiamo condividendo.

Lo schema di Shamir in questo senso non è sicuro poiché se conoscessi a priori il range in cui è compreso il segreto, potrei ciclare su uno share mancante per ottenere un segreto nel range voluto.

***Esempio* 1.** Efftuiamo uno schema (3,4) in cui per conoscere il segreto dobbiamo conoscore almeno 3 share su 4. Dato che utilizziamo l'interpolazione di Lagrange il polinomio sarà di grado $t-1$ e il termine noto sarà s:

$$y = 3x^2 + 52x + 32;$$

Abbiamo 4 shareholders, quindi dobbiamo generare 4 punti,generando un valore casuale x e sostituendolo nell'equazione precedente. Avendo posto rispettivamte i valori $1, 2, 3, 4$, si ottengono i seguenit punti:

$$(1, 87), (2, 148), (3, 215), (4, 288)$$

Ora, occorre calcolare i valori di lambda, supponendo di aver collezionato x_1, x_2, x_3, come

$$\Lambda_i(x) = \prod_{m=1, m\neq 1}^{l} \frac{x - x_m}{x_i - x_m} \quad \Lambda_i(x_i) = 1; \quad \Lambda_i(x_m) = 0 \quad for\ m \neq i$$

Ottenendo:

$$\Lambda_1(x) = \frac{(x - \text{x2})(x - \text{x3})}{(\text{x1} - \text{x2})(\text{x1} - \text{x3})}$$
$$\Lambda_2(x) = \frac{(x - \text{x1})(x - \text{x3})}{(\text{x2} - \text{x1})(\text{x2} - \text{x3})}$$
$$\Lambda_3(x) = \frac{(x - \text{x1})(x - \text{x2})}{(\text{x3} - \text{x1})(\text{x3} - \text{x2})}$$

Ora, per ricostruire il segreto occorre applicare

$$y = \sum_{i=1}^{t} y_i \Lambda_i(x)$$

Quindi:

$$s = y_1\Lambda_{x_1}(0) + y_2\Lambda_{x_2}(0) + y_3\Lambda_{x_3}(0) = 87(30) + 148(-3) + 215(1) = 32$$

Ora supponiamo di non sapere uno share *(d)* e vogliamo verificare se questo schema garantisce secrecy o meno. Sostituendo imponiamo:

$$s = y_1\Lambda_{x_1}(0) + d\Lambda_{x_2}(0) + y_3\Lambda_{x_3}(0) = 476 - 3d$$

Ipotizziamo che il range in cui vive s è noto e compreso tra 0 e 100. Possiamo indovinare il segreto? Si, basta ciclare sulle d:

- Con $d = 125 \to s = 101$;
- Con $d = 126 \to s = 98$;
- Con $d = 127 \to s = 95$;
- Da varie prove si capisce che d è nel range $126 \leq d \leq 158$;

Quindi, conoscere 2 su 3 in uno schema 3 su 4 ci permette di escludere tutti i valori d non ammissibili.

1.2.7 Real Shamir Secret Sharing

Lo schema reale utilizza l'aritmetica modulare (con p numero primo) invece di quella reale e le operazioni effettuate sia con il segreto sia con il polinomio devono essere scelti nel campo dei numeri primi.
L'interpolazione rimane uguale.

Importante. *La regola per scegliere il numero primo p deve essere più grande del dominio del segreto per avere un segreto uniformemente distribuito e non è necessario che sia grande.*

***Esempio* 2.** La nuova costruzione corretta che utilizza il modulo è la seguente.
Supponiamo di avere un segreto $s \in [0, 100]$ in uno schema $(3, 4)$.

1. Scegliamo il primo numero primo maggiore dell'intervallo in cui è compreso s.
$$p = 101$$
2. Il segreto che vogliamo inviare è: $s = 32$.
3. Il polinomio sarà di grado $t - 1$ e con termine noto s:
$$y = Mod[32 + 52x + 3x^2, 101] = (32 + 52x + 3x^2) \mod 101$$
4. Generiamo i valori pergli shareholders:
$$\begin{aligned} x_1 = 1 \rightarrow y_1 = y/.x \rightarrow x_1 = 87 \\ x_2 = 2 \rightarrow y_2 = y/.x \rightarrow x_2 = 47 \\ x_3 = 3 \rightarrow y_3 = y/.x \rightarrow x_3 = 13 \\ x_4 = 6 \rightarrow y_4 = y/.x \rightarrow x_4 = 48 \end{aligned}$$
5. Calcoliamo i valori $\Lambda_i(0)$ presupponendo di conoscere le share di $1, 2, 4$, sostituendo $x = 0$ e ovviamente considerando il modulo:
$$\begin{aligned} \Lambda_1(x) = Mod[(0 - x2) * (0 - x_4) * PowerMod[(x_1 - x2) * (x_1 - x4), -1, 101], 101] = 63 \\ \Lambda_2(x) = Mod[(0 - x1) * (0 - x_4) * PowerMod[(x_2 - x1) * (x_2 - x4), -1, 101], 101] = 49 \\ \Lambda_4(x) = Mod[(0 - x1) * (0 - x_2) * PowerMod[(x_4 - x1) * (x_4 - x2), -1, 101], 101] = 91 \end{aligned}$$
Importante.
$$\Lambda_i(x) = \prod_{m=1, m \neq 1}^{l} \frac{x - x_m}{x_i - x_m} \quad \Lambda_i(x_i) = 1; \quad \Lambda_i(x_m) = 0 \quad for\ m \neq i$$
Questa formula imporrebbe di scrivere il denominatore sotto il segno di frazione, ma questo non è possibile se si effettua il modulo. Quindi, quello che occorre fare è effettuare l'inversa del modulo: in mathematica si utilizza PowerMod. Per esempio per $\Lambda_1(0)$:
$$\Lambda_1(x) = Mod[(0 - x2) * (0 - x_4) * PowerMod[(x_1 - x2) * (x_1 - x4), -1, 101], 101] = \left(\frac{(0 - x2) * (0 - x_4)}{(x_1 - x2) * (x_1 - x4)}\right) \mod p$$
In cui:
$$\frac{1}{(x_1 - x_2) * (x_1 - x_4)} \mod 101 = ((x_1 - x_2)(x_1 - x4))^{-1} \mod 101$$
6. La forma per ricostruire il segreto è la seguente:
$$Mod[y_1\Lambda_1(x) + y_2\Lambda_2(x) + y_3\Lambda_3, 101] = (y_1\Lambda_1(x) + y_2\Lambda_2(x) + y_3\Lambda_3) \mod 101 = 32$$
7. Verifichiamo ora che sia unconditially secure:finché ho anche uno share mancante, allora il segreto potrebbe essere qualsiasi. In particolare, supponiamo che non sia noto $d = y_1$ e cicliamo su d da 0 a 100, sapendo che il segreto è compreso in questo intervallo:
$$Mod[d * \Lambda_1(x) + y_2 * \Lambda_2(x) + y_3 * \Lambda_3(x)/.d \rightarrow Range[0, 100]]$$
Come osserviamo i possibili valori sono molteplici e uniformemente distribuiti tra 0 e 100:
```
Out[•]= {0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13,
  14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24,
  25, 26, 27, 28, 29, 30, 31, 32, 33, 34, 35,
  36, 37, 38, 39, 40, 41, 42, 43, 44, 45, 46,
  47, 48, 49, 50, 51, 52, 53, 54, 55, 56, 57,
  58, 59, 60, 61, 62, 63, 64, 65, 66, 67, 68,
  69, 70, 71, 72, 73, 74, 75, 76, 77, 78, 79,
  80, 81, 82, 83, 84, 85, 86, 87, 88, 89, 90,
  91, 92, 93, 94, 95, 96, 97, 98, 99, 100}
```

36minuti