Condivisione di Segreti Dinamica: Un'implementazione

Leonardo Danella

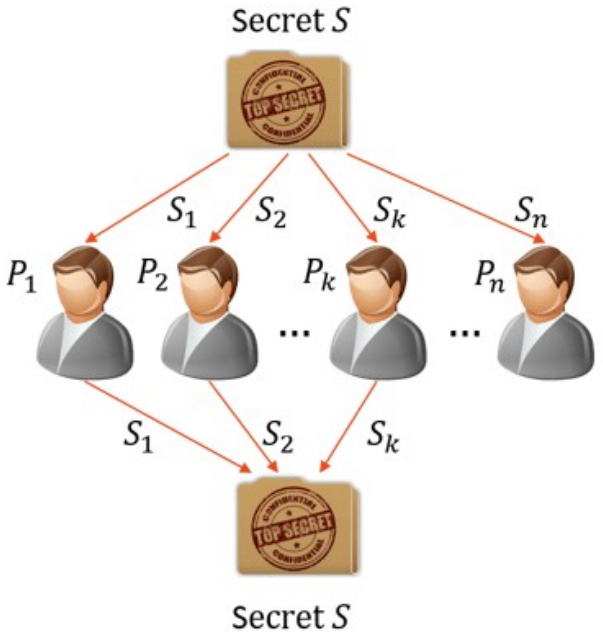
SAPIENZA
UNIVERSITÀ DI ROMA

Obiettivo del tirocinio

Studiare paper su sicurezza e correttezza di Secret Sharing ed Evolving Secret Sharing

Fornire un'implementazione dell'evolving secret sharing che sia:

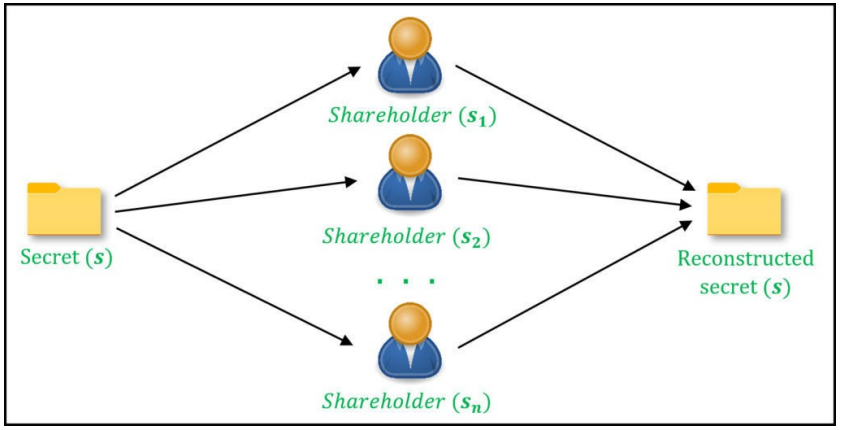
- Sicuro
- Confidenziale
- Con performance paragonabili a uno di SS classico



Secret Sharing

Perché il secret sharing?

- Cos'è e come funziona
- Shamir's & Blakey's SS
- Pro e contro



Secret Sharing

Cos'è e come funziona?

- Algoritmo di condivisione di un segreto

- Dividere il segreto in n parti

Genero un polinomio p di grado (t-1) il cui coefficiente è il segreto da criptare e prendo n punti

- Recuperare il segreto con t < n parti

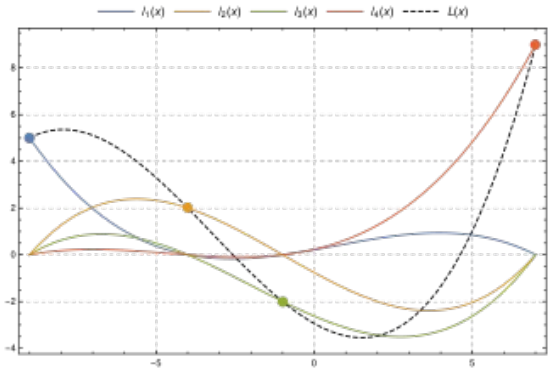
Sfrutto l'interpolazione di Lagrange per recuperare il segreto: t coppie (x,y) serviranno a definire un polinomio fino al (t-1)-esimo ordine

Secret Sharing

Shamir's & Blakey's SS

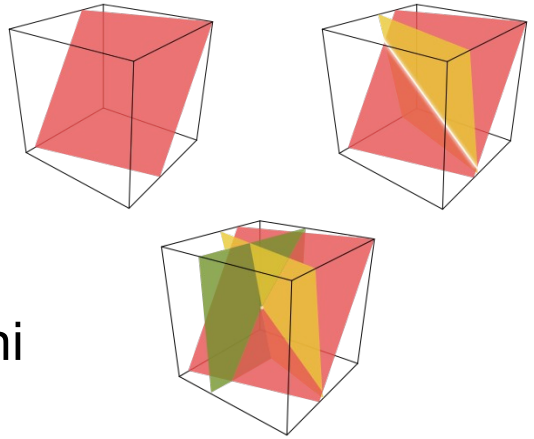
Schema di Shamir:

- polinomio di grado (t – 1)
- t punti
- generare n punti e darne uno a partecipante
- recuperare il segreto sfruttando l'interpolazione di Lagrange



Schema di Blakley:

- iperpiano di grado (n – 1)
- n iperpiani non paralleli
- il segreto giace nell'intersezione dei piani



Secret Sharing

Pro e contro

Pro:

- Confidentiality
- Integrity
- Non c’è un singolo punto di fallimento

Contro:

- No availability
- Grandezza di ogni share ≥ grandezza del segreto

  (Unconditionally secure secret sharing)

Secret Sharing

Computationally secure secret sharing

Algoritmi usati per aumentare le prestazioni di spazio

Due tecniche principali:

- Secret Sharing Made Short

- Crittografare dati con una chiave privata
- Suddividere il ciphertext in n parti (IDA Rabin)
- Fattore di crescita della grandezza di uno share: numero di frammenti / soglia

- AONT-RS

- Trasformazione All or Nothing come IDA
- garantisce che qualsiasi numero di azioni inferiore alla soglia è insufficiente per decriptare i dati

Evolving Secret Sharing

Komargodski, Naor e Yogev [2016]

Differenze tra ESS e SS

- non avere un limite sul numero di share condivisibili

- cambiamenti monotoni sulla struttura d'accesso:
  - Solo aggiunta di partecipanti
  - Insiemi qualificati restano tali

- non avendo in anticipo un limite sugli shares, non si riescono ad ottimizzare i calcoli

Evolving Secret Sharing

Soglia 2: ESS(2, n)

'l' bit di segreto, grandezza share per t-esimo partecipante:

- Diciamo che t ∈ alla generazione g = ⌊log(t)⌋

  $SIZE(g) = 2^g$

- Se 2 partecipanti t1, t2 ∈ ad una stessa generazione possono ricostruire il segreto

- Grandezza di uno share del t-esimo partecipante:

  $\max\{l, \log t\} + \sigma(\log t + 1)$

Evolving Secret Sharing

Soglia k: ESS(k, n)

'l' bit di segreto, grandezza share per t-esimo partecipante:

- Diciamo che t ∈ alla generazione g = $\lfloor \log_k(t) \rfloor$
  $SIZE(g) = k^{g+1} - k^g = (k - 1) \cdot k^g$
- Servono k partecipanti t1, .., t_k ∈ ad una stessa generazione per ricostruire il segreto
- Grandezza di uno share del t-esimo partecipante:
  $kt \cdot \max\{l, \log(kt)\}$

Evolving Secret Sharing

Threshold dinamico: precondizioni

Struttura d'accesso che prende in input valori di soglia t.c.:

- $k1 \leq k2 \leq ..$, per garantire monotonicità, affinché la struttura sia evolutiva
- Al tempo t, sono qualificati solo gli insiemi con cardinalità di almeno k_t

Evolving Secret Sharing

Threshold dinamico: algoritmo

Per ogni partecipante i in una GenSz(g+1):

- Suddivido il segreto $s_{(c0,\ldots,cg)}$ in shares $\Pi_1,.., \Pi_i$
- Ad ogni partecipante in [i] do il rispettivo share

Per ogni $c_{g+1} \in$ [GenSz(g+1)]:

- Generiamo un bit $r_{(c0, \ldots, cg+1)} \leftarrow \{0, 1\}$ randomico
- Condividiamo $r_{(c0, \ldots, cg+1)}$ tra i partecipanti della Gen(g+1)
- Poniamo $s_{(c0 ,\ldots,cg+1)} = s_{(c0,\ldots,cg)} \oplus r_{(c0,\ldots,cg+1)}$

Secret Sharing ed Evolving Secret Sharing

Testing delle implementazioni

Testing			
chars and threshold	Secret Sharing time	Evolving Secret Sharing time	Rapporto ESS/SS
1 and (8, 3)	0.0001468900009058416	0.0003248950015404262	2.211825172148294
1 and (10, 4)	0.0001356089996988885	0.0003544329956639558	2.613639186565439
1 and (15, 7)	0.0003656070002762135	0.0011853360010718461	3.242104227151923
1 and (23, 13)	0.000396353003452532	0.003920939001545776	9.892542676330079
4 and (8, 3)	0.0001572479959577322	0.0004355350174591876	2.769733342587451
4 and (10, 4)	0.0001653709987294860	0.0003919660011888481	2.370222132056095
4 and (15, 7)	0.0003449729993008077	0.0011097449951193994	3.216903923984294
4 and (23, 13)	0.0005766430040239356	0.0037232520116958767	6.456771322489382
16 and (8, 3)	0.0001117869978770613	0.0003153010038658976	2.820551672857807
16 and (10, 4)	0.0001681630019447766	0.0004790980092366226	2.84900961386235
16 and (15, 7)	0.0003610919993661809	0.001227837999977055	3.400346731947147
16 and (23, 13)	0.00079077100235736	0.004249283017998096	5.373594890721332

Il numero di shares influenza di molto le prestazioni poiché l'incremento, all'aumentare degli shares, aumenta linearmente nell'Evolving Secret Sharing, mentre logaritmicamente nel classico Secret Sharing

Conclusioni e sviluppi futuri nell'ESS

- Migliorare le prestazioni
  - temporali
  - spaziali
- Definire la non-malleabilità
- Sfruttare i codici prefisso

SAPIENZA
UNIVERSITÀ DI ROMA

Facoltà di Ingegneria dell'informazione, Informatica e Statistica

Dipartimento di Informatica

Grazie per l'attenzione

Candidato
Leonardo Danella
1885686

Relatore
Daniele Venturi

Anno Accademico 2021-2022