Secret Sharing

A Practical Scheme for Non-interactive Verifiable Secret Sharing by P. Feldman 1987

Sara Adams

sara-adams@gmx.de

Universität Rostock

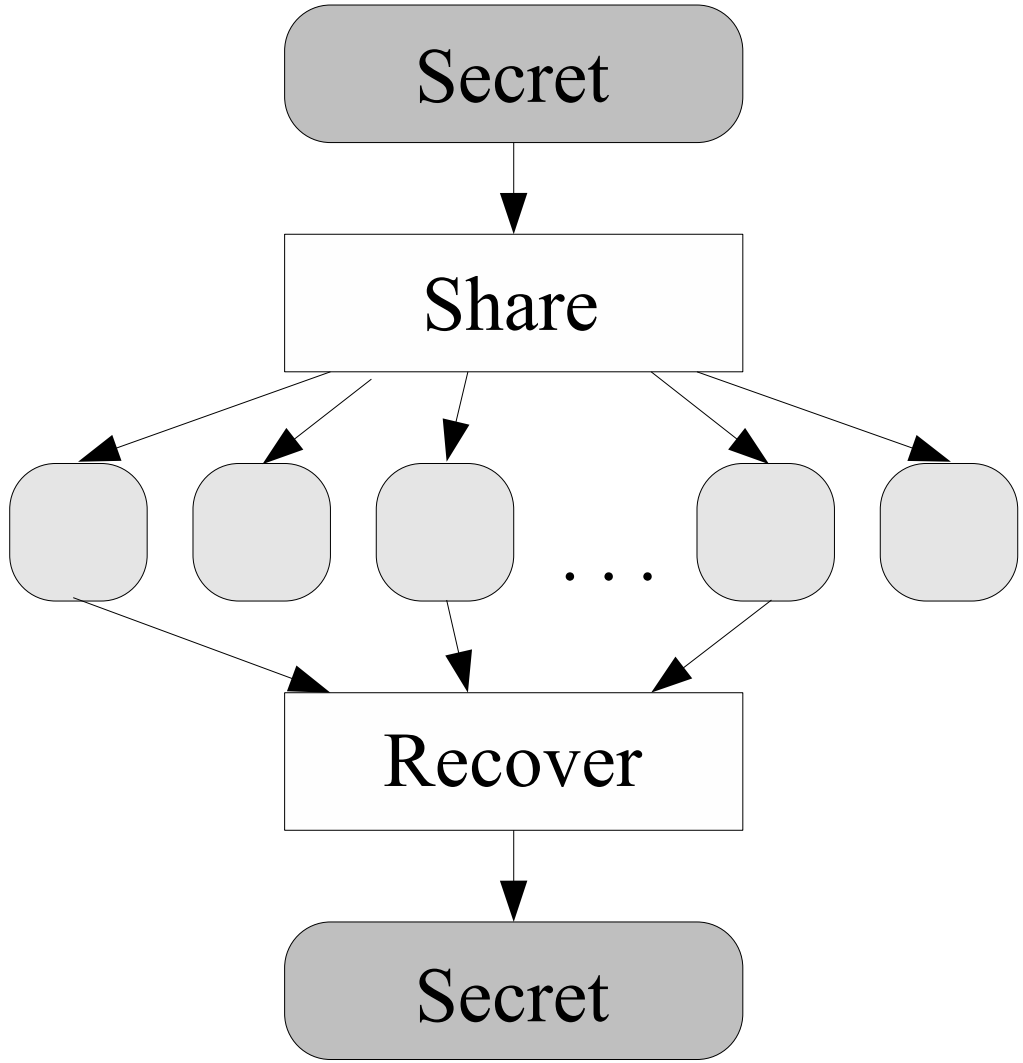
Übersicht

- Motivation
- Gewöhnliches Secret Sharing
  - Beispiel: Polynominterpolation
- Nicht-interaktives, verifizierbares Secret Sharing
- Probabilistische Verschlüsselung
- Homomorphe Verschlüsselung
  - Beispiel: Diskrete Logarithmen
- Zusammenfassung

Motivation

- gegeben: sicherheitskritische Nachricht (das Geheimnis)
- Idee: Übertragen der Verantwortung auf n Personen
- gewünscht: t Personen können das Geheimnis nicht wiederherstellen, u Personen können es jedoch stets
- dabei: $t < u$

Gewöhnliches Secret Sharing



Grundlagen

- Netzwerk mit n Spielern $(1, \ldots, n)$
- Spieler $\widehat{=}$ PPTA (polynomialzeit Algorithmus)
- öffentlicher und privater Kanal
- k Sicherheitsparameter
- $\exists Q_0(n,k), Q_1(n,k)$ Polynome: $\forall$ Spieler $Q_0(n,k) \leq$ max. Schritte je Runde $\leq Q_1(n,k)$
- $MES \hat{=}$*message space* (Definitionsbereich)
- $CIPH \hat{=}$*ciphertext space* (Wertebereich)

Gewöhnliches Secret Sharing

Sei $t < u$.
$(Share, Recover)$ ist (n, t, u)-Secret Sharing $:\Leftrightarrow$

- $\forall k \in \mathbb{N}, \forall \omega \in MES_k, (d_1, \ldots, d_n) =$
  $Share(1^k, \omega) \Rightarrow \ \forall \{a_1, \ldots, a_u\} \subset [n] :$
  $Recover(1^k, (a_1, d_{a_1}, \ldots, (a_u, d_{a_u})) = \omega$

Gewöhnliches Secret Sharing

Sei $t < u$.
$(Share, Recover)$ ist (n, t, u)-Secret Sharing $:\Leftrightarrow$

- $\forall k \in \mathbb{N}, \forall \omega \in MES_k, (d_1, \ldots, d_n) =$
  $Share(1^k, \omega) \Rightarrow \ \forall \{a_1, \ldots, a_u\} \subset [n] :$
  $Recover(1^k, (a_1, d_{a_1}, \ldots, (a_u, d_{a_u})) = \omega$
- $\forall$ PPTAs $A, Guess, \forall c \geq 0 \ \exists k_0 : \ k \geq k_0 \Rightarrow$
  $Pr[\omega = Guess(1^k, (a_1, d_{a_1}), \ldots, (a_t, d_{a_t})) :$
  $(a_1, \ldots, a_t) = A(1^k), \omega = MES_k, (d_1, \ldots, d_n) =$
  $Share(1^k, \omega)] < \frac{1}{|MES_k|} + k^{-c}$

Polynominterpolation (Shamir)

- $p \in \mathbb{P} : \ p \geq 2^n, \ MES = \mathbb{Z}_p$
- $Q(s) = \sum_{i=0}^{t} y_i s^i, \ \omega = y_0$ Geheimnis
- $Share(p, \omega) = (Q(1), \ldots, Q(n)) \mod p$
- $Recover \mathrel{\widehat{=}}$ Polynominterpolation

Polynominterpolation (Shamir)

- $p \in \mathbb{P} : \ p \geq 2^n, \ MES = \mathbb{Z}_p$
- $Q(s) = \sum_{i=0}^{t} y_i s^i, \ \omega = y_0$ Geheimnis
- $Share(p, \omega) = (Q(1), \ldots, Q(n)) \mod p$
- $Recover \mathrel{\widehat{=}}$ Polynominterpolation
- Problem: unehrliche Spieler

Polynominterpolation (Shamir)

- $p \in \mathbb{P} : \ p \geq 2^n, \ MES = \mathbb{Z}_p$
- $Q(s) = \sum_{i=0}^{t} y_i s^i, \ \omega = y_0$ Geheimnis
- $Share(p, \omega) = (Q(1), \ldots, Q(n)) \mod p$
- $Recover \mathrel{\widehat{=}}$ Polynominterpolation

- Problem: unehrliche Spieler
- Lösung: Authentifizierung einführen

Polynominterpolation (Shamir)

- $p \in \mathbb{P} : \ p \geq 2^n, \ MES = \mathbb{Z}_p$
- $Q(s) = \sum_{i=0}^{t} y_i s^i, \ \omega = y_0$ Geheimnis
- $Share(p, \omega) = (Q(1), \ldots, Q(n)) \mod p$
- $Recover \mathrel{\widehat{=}}$ Polynominterpolation

- Problem: unehrliche Spieler
- Lösung: Authentifizierung einführen
- Problem: unehrlicher Geber

Polynominterpolation (Shamir)

- $p \in \mathbb{P} : \ p \geq 2^n, \ MES = \mathbb{Z}_p$
- $Q(s) = \sum_{i=0}^{t} y_i s^i, \ \omega = y_0$ Geheimnis
- $Share(p, \omega) = (Q(1), \ldots, Q(n)) \mod p$
- $Recover \mathrel{\widehat{=}}$ Polynominterpolation

- Problem: unehrliche Spieler
- Lösung: Authentifizierung einführen
- Problem: unehrlicher Geber
- Idee: Verifizieren von Nachrichten

Definitionen

- unehrliche Spieler (*faulty players*)
  - Abweichen vom Protokoll

Definitionen

- unehrliche Spieler (*faulty players*)
  - Abweichen vom Protokoll
- (statischer) t-Gegner auf P (*t-adversary*)
  - Lesen von öffentlichen Nachrichten
  - Bestechen von t Spielern zu Beginn
  - Lesen und Manipulieren der Bänder von bestochenen Spielern

Definitionen

- unehrliche Spieler (*faulty players*)
  - Abweichen vom Protokoll
- (statischer) t-Gegner auf P (*t-adversary*)
  - Lesen von öffentlichen Nachrichten
  - Bestechen von t Spielern zu Beginn
  - Lesen und Manipulieren der Bänder von bestochenen Spielern
- dynamischer t-Gegner auf P
  - Bestechen von bis zu t Spielern
  - Lesen und Manipulieren der Bänder nach Bestechung

Verifizierung

- Erhaltene shares müssen verifiziert werden können.
- Je u verifizierte shares liefern mit $Recover$ das Geheimnis
- Man kann durch t verifizierte shares nicht das Geheimnis in Polynomialzeit raten

Nicht-Interaktivität

$(Share, Recover, Check, Encrypt)$ ist nicht-interaktives (n, t, u)-VSS $:\Leftrightarrow$

- $Share, Recover, Check, Encrypt$ sind PPTAs
- $(Share, Recover)$ ist (n, t, u)-Secret Sharing
- $\forall k \in \mathbb{N}, \forall \omega \in MES_k, \forall 1 \leq j \leq n, Y = Encrypt(1^k, \omega), (d_1, \ldots, d_n) = Share(1^k, \omega) \Rightarrow Check(1^k, Y, j, d_j) = 1$
- $\forall k \in \mathbb{N}, \forall Y \in Range(Encrypt(1^k, .)), \forall I \subset [n] : |I| = u, \forall i \in I \; Check(1^k, Y, i, d_i) = 1 \Rightarrow (1^k, (a_1, d_{a_1}), \ldots, (a_u, d_{a_u})) = \omega$

Zur Verschlüsselung

- Idee: Übermittle eine verschlüsselte Information $E(i)$
  - kein Rückschluß auf die Information i möglich
  - $E(i)$ liefert eine (andere) hilfreiche Information

Zur Verschlüsselung

- Idee: Übermittle eine verschlüsselte Information $E(i)$
  - kein Rückschluß auf die Information i möglich
  - $E(i)$ liefert eine (andere) hilfreiche Information
- deterministische Verschlüsselung
  - einfach zu prüfen, ob
    $\exists \omega : \ Encrypt(1^k, \omega) = Y$

Probabilistische Verschlüsselung

- $H = \{Hard_k : \ k \in \mathbb{N}\}$ Menge aller nicht approximierbaren Prädikate
- nicht approximierbar: $\nexists$ effiziente Mglk. $Hard_k$ auf beliebigen Elementen zu berechnen
- formal:
  $\forall$ PPTAs A, $\forall c > 0 \exists k_0 \in \mathbb{N} : \ \forall k \geq k_0 \Rightarrow$
  $$\left| Pr\big(A(x) = Hard(x) \big| |x| = k\big) - 1/2 \right| < k^{-c}$$
- In Polynomialzeit kann das Prädikat nicht besser vorhergesagt werden als durch Raten

Zwei Ansätze

- Trapdoor Funktionen (Bsp. RSA)
  - schwer berechenbare Funktion, mit Tip einfach zu berechnen

Zwei Ansätze

- Trapdoor Funktionen (Bsp. RSA)
  - schwer berechenbare Funktion, mit Tip einfach zu berechnen
- One-Way Funktionen (Bsp. Hashing)
  - $Enc : MES \rightarrow CIPH$ injektiv
  - Berechnung von Enc einfach, von Enc^{-1} schwer

Zwei Ansätze

- Trapdoor Funktionen (Bsp. RSA)
  - schwer berechenbare Funktion, mit Tip einfach zu berechnen
- One-Way Funktionen (Bsp. Hashing)
  - $Enc : MES \rightarrow CIPH$ injektiv
  - Berechnung von Enc einfach, von Enc^{-1} schwer
  - $Pred : MES \rightarrow \{0,1\},\ Hard = Pred(Enc^{-1})$
    $z = Enc(y), b = Pred(y) \Rightarrow\ Hard(z) = b$
    $\Rightarrow z$ ist probab. Verschlüsselung von b

Homomorphe Verschlüsselung

- Voraussetzungen
  - MES und $CIPH$ Gruppen mit Verknüpfung $\circ$ bzw. $*$
  - $Encrypt(B \circ C) = Encrypt(B) * Encrypt(C)$
  - $Encrypt(\underbrace{B \circ \cdots \circ B}_{c\text{-mal}}) =$

    $\underbrace{Encrypt(B) * \cdots * Encrypt(B)}_{c\text{-mal}} \ \forall c \in \mathbb{Z}$

Beispiel

Diskrete Logarithmen

- $G = \mathbb{Z}_p^*$ multiplikative Gruppe mit Erzeuger g
- $h \in \mathbb{Z}_p^*$ beliebig
- $\text{dlog}_g(h) := k \in \{0, \ldots, p-1\} : \ g^k = h$
- diskreter Log ist schwer zu bestimmen

Diskrete Logarithmen (1)

- $Generator(n, k)$ bestimmt starke Primzahl $p < 2^k$ und Faktorisierung von $p - 1$
  - stark: $p - 1$ hat nicht nur kleine Primfaktoren
- $Generator$ bestimmt Erzeuger g von $\mathbb{Z}_p^*$
- $Enc : \mathbb{Z}_{p-1} \longrightarrow \mathbb{Z}_p^*, y \mapsto g^y \mod p$ homomorph:
  $Enc(x + y) = g^{x+y} = g^x \cdot g^y = Enc(x) \cdot Enc(y)$

Diskrete Logarithmen (2)

- Sei g_q Erzeuger einer großen primen Gruppe der Ordnung q (etwa q Primteiler von $p-1$, Rechnen $\mod p$)
- Wahl von $Hard(x)$ für $x = g_q^k,\ 0 \leq k < q$:
  - $k = (k_n, \ldots, k_1, k_0)$ Binärdarstellung von k
  - $Hard(x) := k_1$
- Wieso ist $Hard(x)$ nicht approximierbar?

Diskrete Logarithmen (3)

Beweisidee:

- Bestimmung von k_0 einfach

Diskrete Logarithmen (3)

Beweisidee:

- Bestimmung von k_0 einfach
- Idee: Berechne $\widetilde{x} = \pm\sqrt{\frac{x}{g^{k_0}}}$
- Ang. wir können k_1 bestimmen
  $\Rightarrow$ Vorzeichen kann bestimmen werden
  $\Rightarrow$ Bestimmung des diskreten Logs möglich
  $\Rightarrow$ $Hard(x)$ so schwierig wie diskreter Log

Zur Vorzeichenbestimmung

Sei r die Wurzel, deren Vorzeichen bestimmt werden soll.

- siehe Voraussetzung: $p \mod 4 \equiv 3$
- $r \equiv 0 \Rightarrow p - r \equiv 3$
- $r \equiv 1 \Rightarrow p - r \equiv 2$
- $r \equiv 2 \Rightarrow p - r \equiv 1$
- $r \equiv 3 \Rightarrow p - r \equiv 0$

- $k_1 = 0 \Rightarrow r \equiv 0, 1 \vee p - r \equiv 0, 1$
- $k_1 = 1 \Rightarrow r \equiv 2, 3 \vee p - r \equiv 2, 3$

Nutzen der Verschlüsselung

- Übermittle verschlüsselte Informationen
  - Verifikation von:
    - eigenen Shares
    - fremden Shares
- Besonderer Vorteil: keine Interaktion nötig
  - Verifikation ohne Nachrichtenaustausch

Verifikation von Shares

- Sei y_0 das Geheimnis und $Q(x) = \sum_{i=0}^{t} y_i x^i$
- Geber veröffentlicht $Y_i := Enc(y_i),\ i = 0, \ldots, t$
- Geber sendet $Q(j)$ an Spieler j
- $Enc(Q(j)) \stackrel{?}{=} \prod_{i=0}^{t} Y_i^{j^i}$, denn:

$$\begin{aligned} Enc(Q(j)) &= Enc(\textstyle\sum_{i=0}^{t} y_i j^i) \\ &= \textstyle\prod_{i=0}^{t} E(j^i y_i) \\ &= \textstyle\prod_{i=0}^{t} E(y_i)^{j^i} = \prod_{i=0}^{t} Y_i^{j^i} \end{aligned}$$

Diskrete Logarithmen (4)

- $y_0 := k_1$ sei Geheimnis und $Q(x) = \sum_{i=0}^{t} y_i x^i$
- Geber veröffentlicht
  $Y_i := Enc(y_i) = g^{y_i} \mod p$
- Geber sendet $Q(j)$ an Spieler j
- Verifikation: $g^{Q(j)} \stackrel{?}{=} \prod_{i=0}^{t} g^{y_i \cdot j^i}$

$$Enc(Q(j)) = g^{Q(j)} = g^{\sum_{i=0}^{t} y_i j^i} = \prod_{i=0}^{t} g^{j^i y_i}$$

Erweiterungen

- bisher: Übermittlung eines einzelnen Bits
- Erweiterung der Prädikate:
  - $Range(Pred) = \{0, \ldots, l\}$
- Erweiterung der Geheimnislänge: Geheimnis durch mehrfache Durchführung verteilen

Zusammenfassung

- Protokoll mit folgenden Eigenschaften:
  - Aufteilung eines Geheimnisses

Zusammenfassung

- Protokoll mit folgenden Eigenschaften:
  - Aufteilung eines Geheimnisses
  - Verifizierbarkeit von Nachrichten:
    - Nachrichten vom Geber
    - Nachrichten von Spielern

Zusammenfassung

- Protokoll mit folgenden Eigenschaften:
  - Aufteilung eines Geheimnisses
  - Verifizierbarkeit von Nachrichten:
    - Nachrichten vom Geber
    - Nachrichten von Spielern
  - Nicht-Interaktivität
    - Geber sendet Nachrichten
    - keine Kommunikation zwischen den Spielern nötig

Zusammenfassung

- Protokoll mit folgenden Eigenschaften:
  - Aufteilung eines Geheimnisses
  - Verifizierbarkeit von Nachrichten:
    - Nachrichten vom Geber
    - Nachrichten von Spielern
  - Nicht-Interaktivität
    - Geber sendet Nachrichten
    - keine Kommunikation zwischen den Spielern nötig
  - Sicherheit gegenüber t-Gegnern