Einführung Code-basierte Kryptografie

Code-basiertes Kryptosystem – McEliece

Fahrplan

Grundlagen

McEliece – Code-basierte Kryptografie

Quellen

Zusammmenfassung

- McEliece asymmetrisches Public-Key-Kryptosystem – 1978 nach Robert McEliece [McE78]
- Grundlegende Idee: Führe absichtliche Fehler in die Chiffre ein
- Verwenden eines allgemeinen fehlerkorrigierende Codes
  - Dekodierung i.A. $\mathcal{NP}$-Hart [Sch07, S. 479], [SP18, S. 353ff]
  - Unterklasse an linearen Codes auch in P lösbar $\rightarrow$ Goppa-Codes
- Angreifer ohne Goppa-Code kann nur in $\mathcal{P}$, also polynomiell viel rechnen
  - Die Entschlüsselung eines zufälligen linearen Codes ist ein $\mathcal{NP}$-Hartes Problem -> [Lju04]
  - Die Generatormatrix eines Goppa-Codes sieht zufällig aus -> [Fau+13]

Fahrplan Grundlagen

Hamming Gewicht

- Das Hamming Gewicht eines Vektors x der Länge n ist definiert als:

$$weight_{\Delta}(x) := \sum_{i=1}^{n} weight_{\Delta}(x_i)$$

mit
$weight_{\Delta}(x_i) = 1 : x_i \neq 0,$
$weight_{\Delta}(x_i) = 0 : x_i = 0$

- Beispiel:

$$weight_{\Delta}(\underline{1}001\underline{1}) = 2$$

Hamming Distanz

- Sei Σ ein diskretes Alphabet und $c_1 = (c_{1_1}, \ldots, c_{1_n})$,$c_2 = (c_{2_1}, \ldots, c_{2_n})$ Codewörter sodass $c_1, c_2 \in C \subseteq \Sigma^n$, wboei C die Menge der gültigen Codeworte darstellt
- Hamming Distanz d zwischen c_1 und c_2 ist definiert als:

$$\Delta(c_1, c_2) := |\{i \in \{1, \ldots, n\} | c_{1_i} \neq c_{2_i}\}$$

- Beispiel:

$$11011001 \oplus 10011101 = 0\underline{1}000\underline{1}00 \implies \Delta(11011001, 10011101) = 2$$

Hamming Distanz

- Für mehr als zwei Worte versteht man das Minimum aller Abstände zwischen verschiedenen Wörtern innerhalb des Codes als deren Hamming Distanz.

$$d = \Delta(C) := \min_{\forall i,j \in \{1,\dots,n\} | i \neq j} \Delta(c_i, c_j)$$

- Beispiel:

$$\begin{aligned} 010 \oplus 011 &= 00\underline{1} \implies \Delta(010, 011) = 1 \\ 010 \oplus 101 &= \underline{111} \implies \Delta(010, 101) = 3 \\ 011 \oplus 101 &= \underline{11}0 \implies \Delta(011, 101) = 2 \end{aligned}$$

$$d = \min\{1, 3, 2\} = 1$$

Galoiskörper

- Ein endlicher Körper der abgeschlossen bezüglich '$+$' und '$*$'.
- Beispiel:
  Die Restklassen modulo 2 bilden den Körper $\mathbb{F}_2 = GF(2)$. [Kun91]
  - Addition: $0+0=0$, $0+1=1$, $1+0=1$, $1+1=0$
  - Multiplikation: $0*0=0*1=1*0=0$, $1*1=1$

Linear-Codes

- Ein binärer Blockcode $C \subseteq GF(2^n) \subseteq \Sigma^n$ heißt linearer Code, wenn gilt:

$$\forall c_1, c_2 \in C\colon c1 \oplus c2 \in C$$

- Bei gegebener Hamming Distanz d wird der Code C auch (n, k, d)-Code genannt.

Goppa-Codes

- Ein Goppa Polynom $g_i \in GF(p^m)$ wird definiert durch:

$$g(x) = g_0 + g_1 x + \ldots + g_t x^t = \sum_{i=0}^{t} g_i x^i$$

- Es sei L eine endliche Untergruppe $GF(p^m)$, wobei p eine Primzahl ist.

$$L = \{\alpha_1, \ldots \alpha_n\} \subseteq GF(p^m)$$

mit $g(\alpha_i) \neq 0 \ \forall \alpha_i \in L$.

(binary) Goppa-Codes

- Ein binärer Goppa-Code $\Gamma(L, g(x))$ ist ein (n, k, d)-Code, der durch ein Generatorpolynom $g(x) \in GF(2^n)$ vom Grad t und einer Sequenz L, über dem endlichen Körper $GF(2^n)$ definiert ist.

Fahrplan Code-basierte Kryptografie

Grundlegende Idee McEliece Kryptosystem

- Transformiere Klartext m (Message) mithilfe einer Generator-Matrix in allgemeinen Goppa-Code
- Multiplikation mit randomisierten Matrizen führt zu allgemeinem linearen Code
  - Gist: Reihe von Matrix-Multiplikationen ist Verschlüsselung
- Retransformation ohne Matrizen in Goppa-Code ist problemtisch: $\mathcal{NP}$-Hart [SP18]
- Öffentlicher Schlüssel:
  - Beinhaltet Generator-Matrix zur Umwandlung in allg. linearen Code
  - Zusätzlich: Anzahl der maximal einbaubaren Fehler in der Chiffre c
  - Fehler sind also die Anzahl der Bits, die invertiert werden sollen
- Privater Schlüssel: Umwandlung des allgemeinen, linearen Codes in Goppa-Code
  - Für performante Retransformation
  - Und Fehlerkorrektur

Parameter Definitionen

- Systemparameter m gibt die Blockgröße an, für zu verschlüsselnde Nachricht
- C sei ein binärer (n, k) Goppa-Code mit t effizient korrigierbaren Fehlern
- t gibt die maximale Anz. eff. korrigierbarer Fehler durch Goppa-Code C [1]
- Daraus ergeben sich:
  - Blocklänge Chiffretext: $n = 2^m$
  - Nachricht Blocklänge $k = n - m \cdot t$
  - Minimale *Hamming-Distance* d des Codes C: $d = 2 \cdot t + 1$

[1] McEliece fixiert $t = 50$, als Maximalwert [McE78]

McEliece als CPA-Sicheres kryptografisches Shema

- Das *McEliece*-Kryptosystem $\Pi := (Gen, Enc, Dec)$
- Wobei:
  - *Gen* Schlüsselerzeugung
  - *Enc* Verschlüsselung
  - *Dec* Entschlüsselung
- Korrekheit: Es muss gelten

$$m = Dec_{priv}(c) = Dec_{priv}(Enc_{pub}(m))$$

Schlüsselerzeugung *Gen*

- Erzeuge Generator-Matrix $G^{k \times n}$ für Goppa-Code C
  - Matrix aus der binärer Klartext mit Länge k die Chiffre der Länge n berechnet werden kann
- Erzeuge zufällige, binäre, nicht singuläre[2] *Scramble-Matrix* $S^{k \times k}$
  - S muss in $\mathbb{Z}_2$ invertierbar sein
- Permutationsmatrix $P^{n \times n}$
  - Binärmatrix, je Zeile genau ein 1 Element enthalten ist
- Berechne: $G'^{k \times n} = S \cdot G \cdot P$
- Schlüssel: $K := (G, S, P, G', t)$
  - Öffentlicher Schlüssel: $K_{pub} := (G', t)$
  - Privater Schlüssel: $K_{priv} := (G, S, P)$

[2] M.a.W. S ist regulär, $\det S \neq 0$; wichtig für Invertierbarkeit

Verschlüsselung *Enc*

- Nachricht in Blöcke, sodass $m \in \mathbb{Z}_2^k$
- Sei $z \in \mathbb{Z}_2^n$ ein belieber Vektor der Länge n, mit maximaler Gewichtung t
  - Gewichtung t: maximale Anzahl Einsen in z
  - Fehlervektor erlaubt es Chiffre an maximal t Stellen zu invertieren
- $Enc_{pub}(m, z) = c = m \cdot G + z$

Entschlüsselung *Dec*

- Berechne $c' = cP^{-1}$
  - $c' = c \cdot P^{-1} = (mG' + z) \cdot P^{-1} = (mG' \cdot P^{-1} + z \cdot P^{-1}) = m(SGP \cdot P^{-1}) + z \cdot P^{-1}$
- Anwenden $decode(c')$ des Goppa-Codes auf c', sodass m' gefunden werden kann
  - Rausrechnen des Fehlervektors z
  - D.h. wir erhalten: $m' = m(SGP \cdot P^{-1}) = m \cdot SG$
  - Hamming-Distanz: $d_H(m'G, c') \leq t$
    - Invertiere mit Generatormatrix G
- Multiplikation mit S^{-1}: $m = m'S^{-1}$
- Kompakt: $dec_{priv}(c) = decode(cP^{-1}) \cdot S^{-1}$

Beispiel McElicece-Kryptosystem

- Kryptosystem (n, k, d) mit Systmeparameter: $n = 7, k = 4, d = 3$
  - 4 Bit Klartext auf 7 Bit Chiffretext
  - Hamming-Distanz $d = 3$
  - Somit lassen sich $t = \frac{d-1}{2} = 1$ Bitfehler korrigieren

Beispiel McElicece-Kryptosystem, cont'

- Schlüsselerzeugung *Gen*: Generator-Matrix erzeugt Hamming-Code statt Goppa-Code
- $G = \begin{pmatrix} 1 & 0 & 0 & 0 & 1 & 1 & 0 \\ 0 & 1 & 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 \end{pmatrix}$

  Da $d = 3$ unterscheidet sich jede Zeile in mindestens drei Werten
- Zufällige Matrizen S und P

$$S = \begin{pmatrix} 1 & 1 & 0 & 1 \\ 1 & 0 & 0 & 1 \\ 0 & 1 & 1 & 1 \\ 1 & 1 & 0 & 0 \end{pmatrix} \qquad P = \begin{pmatrix} 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 \end{pmatrix}$$

Beispiel McElicece-Kryptosystem, cont'

Berechnung des öffentlichen Schlüssels $G' = S \cdot G \cdot P$:

$$G' = \begin{pmatrix} 1 & 1 & 0 & 1 \\ 1 & 0 & 0 & 1 \\ 0 & 1 & 1 & 1 \\ 1 & 1 & 0 & 0 \end{pmatrix} \cdot \begin{pmatrix} 1 & 0 & 0 & 0 & 1 & 1 & 0 \\ 0 & 1 & 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 \end{pmatrix} \cdot \begin{pmatrix} 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 \end{pmatrix} =$$

Beispiel McElicece-Kryptosystem, cont'

Berechnung des öffentlichen Schlüssels $G' = S \cdot G \cdot P$:

$$= \begin{pmatrix} 1 & 1 & 0 & 1 & 1 & 0 & 0 \\ 1 & 0 & 0 & 1 & 0 & 0 & 1 \\ 0 & 1 & 1 & 1 & 0 & 0 & 1 \\ 1 & 1 & 0 & 0 & 0 & 1 & 1 \end{pmatrix} \cdot \begin{pmatrix} 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 \end{pmatrix} = \begin{pmatrix} 1 & 1 & 1 & 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 1 & 1 & 0 & 1 \\ 0 & 1 & 0 & 1 & 1 & 1 & 0 \end{pmatrix}$$

Beispiel McElicece-Kryptosystem, cont'

Der öffentlichen Schlüssels $K_{pub} = (G', t)$:

$$K_{pub} = (G', t) = \left(\begin{pmatrix} 1 & 1 & 1 & 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 1 & 1 & 0 & 1 \\ 0 & 1 & 0 & 1 & 1 & 1 & 0 \end{pmatrix}, 1 \right)$$

Beispiel McElicece-Kryptosystem, cont'

Nachricht $m = (1101)$, Fehlervektor z mit maximalem Gewicht $t = 1$ und Länge $n = 7$:
Wähle $z = (0000100)$

$$Enc_{pub}(m, z) = c = m \cdot G' + z$$

$$\begin{aligned} m &= \begin{pmatrix} 1 & 1 & 0 & 1 \end{pmatrix} \cdot \begin{pmatrix} 1 & 1 & 1 & 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 1 & 1 & 0 & 1 \\ 0 & 1 & 0 & 1 & 1 & 1 & 0 \end{pmatrix} + \begin{pmatrix} 0 & 0 & 0 & 0 & 1 & 0 & 0 \end{pmatrix} \\ &= \begin{pmatrix} 0 & 1 & 1 & 0 & 0 & 1 & 0 \end{pmatrix} + \begin{pmatrix} 0 & 0 & 0 & 0 & 1 & 0 & 0 \end{pmatrix} \\ &= \begin{pmatrix} 0 & 1 & 1 & 0 & 1 & 1 & 0 \end{pmatrix} = c \end{aligned}$$

Beispiel McElicece-Kryptosystem, cont'

Entschlüsselung der Chiffre:
Invertierung der Permuation $c' = cP^{-1}$

$$c' = \begin{pmatrix} 0 & 1 & 1 & 0 & 1 & 1 & 0 \end{pmatrix} \cdot \begin{pmatrix} 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 \end{pmatrix}$$

$$= \begin{pmatrix} 1 & 0 & 0 & 0 & 1 & 1 & 1 \end{pmatrix}$$

Beispiel McElicece-Kryptosystem, cont'

- Dekodierung des Hamming-Codes:
- Berehcne Hamming-Distanz d der Generator-Matrix G: $\begin{pmatrix}1 & 3 & 3 & 2\end{pmatrix}$
- Somit ist $m' = \begin{pmatrix}1 & 0 & 0 & 0\end{pmatrix}$
- Berechne Klartext m

$$
\begin{aligned}
m = m'S^{-1} &= \\
&= \begin{pmatrix}1 & 0 & 0 & 0\end{pmatrix} \cdot \begin{pmatrix}1 & 1 & 0 & 1\\ 1 & 1 & 0 & 0\\ 0 & 1 & 1 & 1\\ 1 & 0 & 0 & 1\end{pmatrix} \\
&= \begin{pmatrix}1 & 1 & 0 & 1\end{pmatrix}
\end{aligned}
$$

Vor- & Nachteile

- The good news: Es gab keine erfolgreichen Angriffe gegen das McEliece-Verfahren
- Verhfahren gilt als *IND-CCA2* [Dot+12] sicher, somit ist es auch *IND-CPA* sicher [Noj+08]
- Angriffe McEliece mit originalen Parametern von 1978 in 1400 Tagen (Einzelne Machine) oder in 7 Tagen mithilfe von 200 CPUs [Bal+16], [CS98]
- Jedoch:
  - Bruce Schneier: McEliece-Kryptosystem etwa 2 bis 3 mal langsamer als RSA [Sch07, S. 479ff]
  - Extrem große öffentliche Schlüssel: $\hat{G}$ ist Matrix $k \times n$
  - Bei Parameter $(1024, 524, 101)$ ist $k \cdot n = 1024 \cdot 524 = 536576$ Bit also etwa $67 kBytes$
  - Chiffretext ist fast doppelt so groß wie Klartext, aus $524 Bit$ klartext werden zu 1024 Bit Chiffre

Quellen I


Marco Baldi u. a. “Enhanced public key security for the McEliece cryptosystem”. In: *Journal of Cryptology* 29.1 (2016), S. 1–27.


Anne Canteaut und Nicolas Sendrier. “Cryptanalysis of the original McEliece cryptosystem”. In: *International Conference on the Theory and Application of Cryptology and Information Security*. Springer. 1998, S. 187–199.


Nico Dottling u. a. “A CCA2 secure variant of the McEliece cryptosystem”. In: *IEEE Transactions on Information Theory* 58.10 (2012), S. 6672–6680.


J. Faugère u. a. “A Distinguisher for High-Rate McEliece Cryptosystems”. In: *IEEE Transactions on Information Theory* 59.10 (2013).


Ernst Kunz. *Endliche Körper (Galois-Felder)*. Vieweg+Teubner Verlag, 1991, S. 185–190.

Quellen II


Ivana Ljubic. "Exact and memetic algorithms for two network design problems". In: *PhD, Technische Universitat Wien, Vienna Austria* (2004).


Robert J McEliece. "A public-key cryptosystem based on algebraic". In: *Coding Thv* 4244 (1978), S. 114–116.


Ryo Nojima u. a. "Semantic security for the McEliece cryptosystem without random oracles". In: *Designs, Codes and Cryptography* 49.1-3 (2008), S. 289–305.


Bruce Schneier. *Applied cryptography: protocols, algorithms, and source code in C*. John Wiley & Sons, 2007.


Douglas Robert Stinson und Maura Paterson. *Cryptography: theory and practice*. CRC press, 2018.