Stromchiffren

Verschlüsseln eines Stroms von Daten m_i (Bits/Bytes) mithilfe eines Schlüsselstroms k_i in die Chiffretexte c_i .

Idee: Im One-Time Pad den zufälligen Schlüssel durch eine pseudo-zufällige Folge k_i ersetzen, welche vom Schlüssel k abhängt.

- Aus kurzem Schlüssel k langen Schlüssel $(\dots, k_i, \dots)$ machen.
- Keine perfekte Sicherheit.
- Komplexitätstheoretische Sicherheit, wenn pseudo-zufällige Folge nicht von einer echt zufälligen Folge effizient unterschieden werden kann.

Aber: Niemals gleichen Schlüsselstrom für verschiedene Nachrichten verwenden! (Fehler wird/wurde in Windows gemacht ...)

1

29. Oktober 2008

Stromchiffren

Synchrone Stromchiffren: k_i abhängig von k , unabhängig von m_i .

- $\sigma_{i+1} = f(\sigma_i, k)$ (Zustände),
- $k_i = g(\sigma_i, k)$,
- $c_i = \mathcal{E}(k_i, m_i)$, $m_i = \mathcal{D}(k_i, c_i)$.

Selbst-synchrone Stromchiffren: k_i abhängig von k und c_i .

- $\sigma_i = (c_{i-t}, \dots, c_{i-1})$ für ein festes t ,
- $k_i = g(\sigma_i, k)$,
- $c_i = \mathcal{E}(k_i, m_i)$, $m_i = \mathcal{D}(k_i, c_i)$.

Meistens $\mathcal{E}(x, y) = \mathcal{D}(x, y) = x \oplus y$.

Wichtiges Beispiel: OFB, CTR, CFB für Blockchiffren.

2

29. Oktober 2008

Stromchiffren

Eigenschaften synchroner Stromchiffren:

- Sender/Empfänger müssen synchronisiert sein.
- Keine Fehlerfortpflanzung.
- Änderung von Chiffretext c_i möglich.

Eigenschaften selbst-synchroner Stromchiffren:

- Nach Übertragungsfehler Selbstsynchronisation, wenn t konsekutive Chiffretexte richtig übertragen werden.
- Beschränkte Fehlerfortpflanzung in maximal t folgende Entschlüsselungen.
- Änderung von Chiffretext c_i leichter erkennbar als bei synchronen Stromchiffren, Einfügen/Ausschneiden von c_i schwerer erkennbar.

In beiden Fällen auf Datenauthenzizität und -integrität extra achten.

3

29. Oktober 2008

Schlüsselstromerzeugung

Für synchrone Stromchiffren wird ein Schlüsselstrom benötigt. Darf nicht von einer zufälligen Bitfolge effizient unterscheidbar sein.

→ Pseudozufallsbitgeneratoren.

- Nimmt kurze Eingabe (seed) und produziert lange, „zufällige“ Ausgabe.
- Statistische Tests von Algorithmen durchführbar.
- Kryptographisch sicher, wenn „alle statistischen Tests“ bestanden werden.

Konstruktionen für Schlüsselstromerzeugung:

- Linear Feedback Shift Register (LFSR, nicht sicher).
- Lineare Kongruenzgeneratoren (nicht sicher).
- Ausgabe von Blockchiffren wie in OFB, CTR (vermutlich sicher).
- ...

4

29. Oktober 2008

Lineare Feedback Shift Register

Zustands- bzw. Schlüsselstromerzeugung durch
Lineare Feedback Shift Register (LFSR):

- Betrachten σ_i aus $\mathbb{F}_2$.
- Startwerte $\sigma_0, \dots, \sigma_{l-1}$.
- Rekursion: $\sigma_j = a_1\sigma_{j-1} + \dots + a_l\sigma_{j-l}$.

Maximale Periode $2^l - 1$ für geeignete Wahl der a_i möglich.
($x^l + a_1x^{l-1} + \dots + a_l$ prim und Nullstelle ist Erzeuger von $(\mathbb{F}_{2^l})^\times$.)

Lineare Komplexität einer Folge $\sigma_0, \sigma_1, \dots$: Kleinstes l , für das die Folge durch ein LFSR mit geeigneten $a_1, \dots, a_l$ entsteht.

LFSR's sind schnell in Hardware!

5

29. Oktober 2008

Lineare Feedback Shift Register

Die Ausgabe eines LFSR ist leicht vorherzusagen (die a_i können mit dem Berlekamp-Massey Algorithmus aus wenigen σ_i berechnet werden).

Daher nicht für kryptographische Zwecke geeignet.

Lösung (teilweise): Mehrere LFSR nicht-linear kombinieren.

Brauchen mindestens:

- Große Periode.
- Große lineare Komplexität.
- Gute statistische Eigenschaften.

Sicherheit bei der Verwendung von LFSR's dann häufig trotzdem etwas vage ...

6

29. Oktober 2008

Lineare Kongruenzgeneratoren

Lineare Kongruenzgeneratoren.

- Rechnen in $\mathbb{Z}/(m)$ bzw. in $\{0, \dots, m-1\}$ modulo m .
- $\sigma_{i+1} = a\sigma_i + b \pmod{m}$.

Eigenschaften:

- Software-geeignet.
- Viel für nicht kryptographische Anwendungen benutzt.
- Allerdings nicht kryptographisch sicher, da leicht vorhersagbar.

7

29. Oktober 2008

Ausgabe von Blockchiffren

Im OFB, CTR und CFC Mode wird der Schlüsselstrom als Ausgabe eines Blockchiffres definiert.

Die Philosophie hier ist ungefähr:

Verhält sich der Blockchiffre wie eine zufällige Permutation, so ergibt dies einen zufälligen Schlüsselstrom.

8

29. Oktober 2008

RC4

RC = Ron's Cipher nach Ron Rivest (Mitgründer von RSA).

RC4 sehr schneller Stromchiffre:

- 1987 entwickelt, 7 Jahre geheim,
- 1994 anonym im Internet veröffentlicht,
- kommerziell (Lizenzgebühren).

In RC4 wird im folgenden speziell $m = 256$ und variables $l \leq 256$ verwendet. Klartext und Schlüsselstrom werden geXORed.

Ist weit verbreitet (Oracle SQL, Windows, SSL, IEEE 802.11 WLAN Standard, ...)

Hat gewisse Sicherheitsschwächen (WEP in IEEE 802.11, Windows).

9

29. Oktober 2008

RC4

Alle Additionen in $\{0, \dots, m-1\}$ modulo m :

- $i, j \in \{0, \dots, m-1\}$, $S_0, \dots, S_{m-1} \in \{0, \dots, m-1\}$.
- Schlüssel k besteht aus $a_i \in \{0, \dots, m-1\}$ für $0 \leq i \leq l-1$.
- Sollte in der Praxis hinreichend zufällig und geheim sein. (!)

Initialisierung (geheimer Schlüssel $\Rightarrow$ initiale Permutation S):

- $S_v \leftarrow v$ für $0 \leq v \leq m-1$.
- $j \leftarrow 0$.
- Für $i = 0, \dots, m-1$: $j \leftarrow j + S_i + a_{i \bmod l}$, vertausche S_i und S_j .

Neues Schlüsselstromglied berechnen (anhand vom aktuellen S):

- Zum Anfang $i \leftarrow 0$, $j \leftarrow 0$.
- Dann: $i \leftarrow i+1$, $j \leftarrow j + S_i$, S_i und S_j vertauschen, $t \leftarrow S_i + S_j$.
- Ausgabe S_t (liefert „zufälligen“ Wert aus $\{0, \dots, m-1\}$).

10

29. Oktober 2008

RC4 Bemerkungen

Untersuchungen der beiden Teilschritte von RC4 in den letzten 15 Jahren haben immer bessere Angriffe produziert, wobei die Initialisierung besondere Schwächen aufweist.

$\Rightarrow$ Sicherheit von RC4 fragwürdig ...

Als Folge der Schwäche der Initialisierung komplett unsicher bei folgender Verwendung (wie in WEP):

- Schlüssel a_i besteht aus festem, geheimen Schlüssel und variablem IV.
- Das erste (die ersten) Ausgabeworte des Schlüsselstroms sind bekannt.

11

29. Oktober 2008

RC4 Bemerkungen

Bessere Verwendung von RC4:

- Schlüssel a_i per Hashfunktion aus festem, geheimen Schlüssel und variablem IV ableiten.
- Die ersten n Worte des Schlüsselstroms verwerfen ($n = 768$, besser $n = 3072$).

Vorschläge für neue Stromchiffren, die RC4 ersetzen könnten:

- eSTREAM Projekt in ECRYPT

12

29. Oktober 2008