Erinnerung Blockchiffre

Definition schlüsselabhängige Permutation

Seien F, F^{-1} pt Algorithmen. F heißt *schlüsselabhängige Permutation* auf ℓ Bits falls

1. F berechnet eine Funktion $\{0,1\}^n \times \{0,1\}^\ell \to \{0,1\}^\ell$, so dass für alle $k \in \{0,1\}^n$ die Funktion $F_k(\cdot)$ eine Bijektion ist.
2. $F_k^{-1}(\cdot)$ berechnet die Umkehrfunktion von $F_k(\cdot)$.

Definition Starke Pseudozufallspermutation (Blockchiffre)

Sei F eine schlüsselabhängige Permutation auf ℓ Bits. Wir bezeichnen F als *starke Pseudozufallspermutation (Blockchiffre)*, falls für alle ppt D gilt

$$\left|\mathrm{Ws}[D^{F_k(\cdot),F_k^{-1}(\cdot)}(1^n)=1] - \mathrm{Ws}[D^{f(\cdot),f^{-1}(\cdot)}(1^n)=1]\right| \leq \mathrm{negl}(n),$$

mit $k \in_R \{0,1\}^n$ und $f \in_R Perm_\ell$.

Angriffe auf Blockchiffren

Angriffe: in aufsteigender Stärke

1. Ciphertext-only: $\mathcal{A}$ erhält $F_k(x_i)$ für unbekannte x_i.
2. Known plaintext: $\mathcal{A}$ erhält Paare $(x_i, F_k(x_i))$
3. Chosen plaintext: $\mathcal{A}$ wählt x_i und erhält $F_k(x_i)$.
4. Chosen ciphertext: $\mathcal{A}$ wählt x_i, y_i und erhält $F_k(x_i), F_k^{-1}(y_i)$.

Sicherheit:

- Jede Pseudozufallspermutation F ist CPA-sicher.
- Jede starke Pseudozufallspermutation F ist CCA-sicher.

Warnung:
Blockchiffren selbst sind **kein sicheres** Verschlüsselungsschema.

Substitutions-Permutations Netzwerk (SPN)

Ziel: Kleine Eingabedifferenzen erzeugen pseudozufällige Ausgaben.

Paradigma Konfusion und Diffusion

Rundeniterierte Vorgehensweise zur Konstruktion einer Blockchiffre

1. **Konfusion:** Permutiere kleine Bitblöcke schlüsselabhängig.
2. **Diffusion:** Permutiere alle Bits.

Bsp: F soll Blocklänge 128 Bits besitzen.

- Konfusion: Definiere schlüsselabhängige Permutation $f_1, \ldots, f_{16}$ auf 8 Bits. Sei $x = x_1 \ldots x_{16} \in (\{0,1\}^8)^{16}$. Definiere
$$F_k(x) = f_1(x_1) \ldots f_{16}(x_{16}).$$
- Diffusion: Permutiere die Bits von $F_k(x)$.
- Iteriere die obigen beiden Schritte hinreichend oft, damit kleine Eingabedifferenzen sich auf alle Ausgabebits auswirken.
- Beschreibungslänge von f_i: $8 \cdot 2^8$ Bits, F : $16 \cdot 8 \cdot 2^8 = 2^{15}$ Bits.
- Länge einer echten Zufallspermutation: $128 \cdot 2^{128} = 2^{135}$ Bits.

Substitutions-Permutations Netzwerk (SPN)

Szenario: Verwende einen Masterschlüssel k.

- Berechne aus dem Masterschlüssel k Rundenschlüssel $k_1, \ldots k_r$ mittels eines sogenannten Keyschedule-Algorithmus.
- Die Permutationsfunktionen $f_1, \ldots, f_m$ werden fest und schlüsselunabhängig gewählt (sogenannte S-Boxen).

Beschreibung Substitutions-Permutations Netzwerk (SPN)

EINGABE: $f_1, \ldots, f_m$, $k \in \{0,1\}^n$, x, ℓ

1. Berechne $k_1, \ldots, k_r \in \{0,1\}^\ell$ aus k. $y \leftarrow x$.
2. For $i \leftarrow 1$ to r
   1. **Schlüsseladdition:** $y \leftarrow y \oplus k_i$. Schreibe $y = y_1 \ldots y_m$.
   2. **Substitution per S-Boxen:** $y \leftarrow f_1(y_1) \ldots, f_m(y_m)$
   3. **Permutation:** $y \leftarrow$ Permutation der Bits von y.

AUSGABE: $F_k(x) := y$

Beobachtung: F ist invertierbar, da jeder Schritt invertierbar ist.

Lawineneffekt

Ziel: Veränderung in Eingabebit wirkt sich auf alle Ausgabebits aus.

Beobachtung Notwendige Eigenschaften für Lawineneffekt

1. **S-Box:** Ändern eines Eingabebits verändert ≥ 2 Ausgabebits.
2. **Permutation:** Ausgabebits einer S-Box werden zu Eingabebits verschiedener S-Boxen.

Beobachtung: Lawineneffekt

- Betrachten ein SPN mit 4 Bit S-Boxen und Blocklänge 128 Bit.
- 1-Bit Eingabedifferenz erzeugt mindestens eine 2-Bit Differenz.
- Eine 2-Bit Differenz resultiert in zwei 1-Bit Differenzen an verschiedenen S-Boxen in der nächsten Runde.
- Diese sorgen für mindestens 4-Bit Differenz, usw.
- D.h. jede Runde verdoppelt potentiell die beeinträchtigen Bits.
- Nach 7 Runden sind alle $2^7 = 128$ Bits von der Veränderung eines Eingabebits beeinträchtigt.

Angriff auf eine Runde eines SPN

Algorithmus Angriff auf eine Runde eines SPN

EINGABE: x, $y = F_k(x)$

1. $y :=$ Invertiere auf y die Permutation und die S-Boxen.
2. Berechne $k := x \oplus y$.

AUSGABE: k

Anmerkungen:

- Die Invertierung in Schritt 1 ist möglich, da sowohl die Permutation als auch die S-Boxen öffentlich sind.
- Nach Invertierung erhält man den Wert $x \oplus k$.

Angriff auf zwei Runden eines SPN

Szenario:

- Wir betrachten ein SPN mit Blocklänge 64 Bit und 2 Runden.
- Der Masterschlüssel $k = k_1 k_2$ ist 128 Bit lang.
- Die Schlüssel $k_1, k_2 \in \{0,1\}^{64}$ sind die zwei Rundenschlüssel.
- Wir schreiben $k_i = k_{i,1} \ldots k_{i,16}$ mit $k_{i,j} \in \{0,1\}^4$.
- Die S-Boxen sind 4 Bit groß. Wir erhalten 6 Paare (x_i, y_i).

Algorithmus Angriff auf zwei Runden eines SPN

EINGABE: (x_i, y_i) für $i = 1, \ldots, 6$

1. $w_1 \ldots w_{16} :=$ Invertiere auf y_i die Permutation und Substitution.
2. For $j = 1$ to 16
   1. w_j wird von Ausgabebits aus höchstens 4 S-Boxen der 1. Runde beeinflusst.
   2. Diese 4 Bits hängen von höchstens vier 4-Bit Werten von k_1 ab. Rate diese 4 Werte. Rate ebenfalls $k_{2,j}$.
   3. Entferne Schlüssel, die nicht mit allen (x_i, y_i) konsistent sind.

AUSGABE: $k = k_1 k_2$

Angriff auf zwei Runden eines SPN

Korrektheit:

- Jeder 4-Bit Block der zweiten Runde hängt von höchstens vier 4-Bit Blöcken der ersten Runde ab.

Laufzeit:

- Schritt 2.2: Raten von 20 Bits, d.h. 2^{20} Möglichkeiten.
- Wir nehmen an, dass ein falscher Schlüssel auf dem j-ten Block eines Paars (x_i, y_i) mit Ws $\frac{1}{2^4}$ korrekt ist.
- D.h. jedes Paar (x_i, y_i) reduziert den Schlüsselraum um Faktor $\frac{1}{2^4}$.
- Nach 6 Paaren (x_i, y_i) verbleibt nur der korrekte Schlüssel.
- Die Gesamtlaufzeit ist beschränkt durch $16 \cdot 2^{20} \cdot 6 < 2^{27}$.
- Man vergleiche mit der Komplexität 2^{128}, um k zu raten.

Distinguisher für 2 Runden

Bsp: Distinguisher für 2 Runden

- Wir betrachten Blocklänge 80 Bit und 4 Bit S-Boxen.
- Wähle x_i, die sich nur im ersten 4-Bit Block unterscheiden.
- Nach 1. Runde: Ausgaben unterscheiden sich in ≤ 4 Blöcken.
- Nach 2. Runde: Ausgaben unterscheiden sich in ≤ 16 Blöcken.
- D.h. nicht alle der 20 Ausgabeblöcke werden verändert.
- Können SPN leicht von Pseudozufallspermutation entscheiden.