06 - Lineare Kryptoanalyse

Luc Spachmann

FSU Jena

30.11.2023

Substitutions-Permutations-Netzwerk

- Designprinzip für Blockchiffren
- Lokale Substitution durch S Boxen
- 'Globale' Permutation
- Schlüsseladdition
- Arbeitet in Runden
- Beispiel: AES

Heutiges SPN

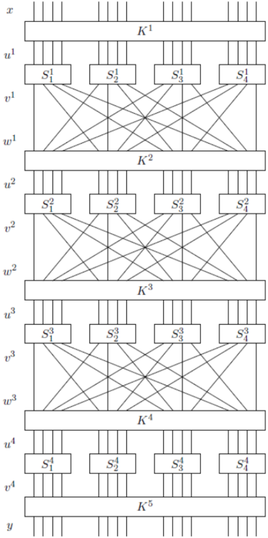
- Das gleiche aus der VL
- 4 Blöcke à 4 Bit
- 4 Runden
- Alle Rundenschlüssel sind gleich
- Alle S-Boxen sind identisch
- S-Box:

z	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
π_S	E	4	D	1	2	F	B	8	3	A	6	C	5	9	0	7

- Permutation:

z	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
π_P	1	5	9	13	2	6	10	14	3	7	11	15	4	8	12	16

Heutiges SPN



Lineare Kryptoanalyse

- Idee: Suche lineare Approximation an S-Boxen
- Sei $a, b \in \{0,1\}^4$, U die gleichverteilte ZV für den Input der S-Box und $V = S(U)$. Dann

$$U_a = \bigoplus_{i=1}^{4} a_i U_i \qquad U_b = \bigoplus_{i=1}^{4} b_i V_i$$

- Suche a, b, c sodass mit hoher Wahrscheinlichkeit gilt:

$$V_b = U_a \oplus c$$

- Bias einer Zufallsvariable X:

$$\varepsilon(X) = Pr[X = 0] - \frac{1}{2}$$

Lineare Kryptoanalyse

- Güte der Approximation:

$$|\varepsilon(U_a \oplus V_b)|$$

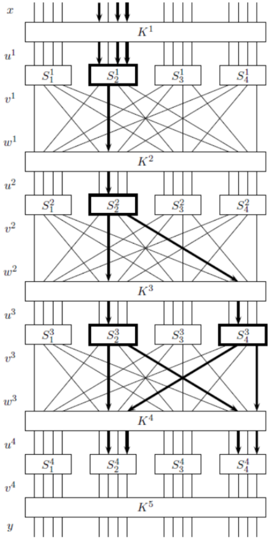
- Gute Approximationen für unsere S-Boxen z.B.:

$$\begin{aligned} T &= U_1 \oplus U_3 \oplus U_4 \oplus V_2 \\ T' &= U_2 \oplus V_2 \oplus V_4 \end{aligned}$$

- Bias der Approximationen:

$$\begin{aligned} \varepsilon(T) &= \frac{1}{4} \\ \varepsilon(T') &= -\frac{1}{4} \end{aligned}$$

Lineare Kryptoanalyse



- 'Verfolgen' Bits durch die Approximation
- Zusammengefasst erhalten wir

$$X_5 \oplus X_7 \oplus X_8 \oplus U_6^4 \oplus U_8^4 \oplus U_{14}^4 \oplus U_{16}^4$$

- Gesamtgüte ungefähr Produkt der Güten der Approximationen

$$\varepsilon \approx (\frac{1}{4})^4 = \frac{1}{32}$$

(Teil-) Schlüsselbestimmung

- Zufälliger Klartext X erfüllt mit hoher oder niedriger (nicht mittlerer) Wahrscheinlichkeit die Approximation

$$X_5 \oplus X_7 \oplus X_8 \oplus U_6^4 \oplus U_8^4 \oplus U_{14}^4 \oplus U_{16}^4 = 0$$

- Benötigen Menge M an Klartext-Kryptotextpaaren (x, y)
- Berechnen für jeden Teilschlüssel-Kandidat (L_1, L_2) für $(K^5_{(2)}, K^5_{(4)})$ Rückwärts benötigte Bits von u_4 (für jedes Paar)
- Für jeden Teilschlüssel-Kandidat berechne Wahrscheinlichkeit der Approximation
- Höchste oder niedrigste W'keit vermutlich richtiger Teilschlüssel (größte Differenz zu $\frac{1}{2}$)
- Teilschlüssel verringert Suchraum für Brute-Force

for $(L_1, L_2) \leftarrow (0,0)$ **to** (F,F) **do**
  $\alpha(L_1, L_2) \leftarrow 0$ **end**
for each $(x, y) \in M$ **do**
  for $(L_1, L_2) \leftarrow (0,0)$ **to** (F, F) **do**
    $v^4_{(2)} \leftarrow L_1 \oplus y_{(2)};\ v^4_{(4)} \leftarrow L_2 \oplus y_{(4)}$
    $u^4_{(2)} \leftarrow \pi_S^{-1}(v^4_{(2)});\ u^4_{(4)} \leftarrow \pi_S^{-1}(v^4_{(4)})$
    if $x_5 \oplus x_7 \oplus x_8 \oplus u^4_6 \oplus u^4_8 \oplus u^4_{14} \oplus u^4_{16} = 0$ **then**
      $\alpha(L_1, L_2) \leftarrow \alpha(L_1, L_2) + 1$
  end
end
$max \leftarrow -1$
for $(L_1, L_2) \leftarrow (0,0)$ **to** (F, F) **do**
  $\beta(L_1, L_2) \leftarrow |\alpha(L_1, L_2) - {}^{t}\!/_{2}|$
  if $\beta(L_1, L_2) > max$ **then**
    $max \leftarrow \beta(L_1, L_2)$
    $maxkey \leftarrow (L_1, L_2)$
end
Ausgabe: maxkey

Aufgaben

- Implementiert die Verschlüsselung des beschriebenen SPN
- Programmname [Input] [Schlüssel] [Output]
- Hier: Schlüssel für jede Runde gleich (16 Bit/ 4 Hexadezimalziffern)
- Input: Folge an Hexadezimalziffern, je 4 ein Block (wie ECB)
- Output analog
- Implementiert Teilschlüsselsuche für gegebene lineare Approximation
- Erzeugt dazu die Klartext-Kryptotextpaare einfach selber
- Wie viele Paare werden benötigt?
- In der Theorie sind es ca $t\varepsilon^{-2} \approx t \cdot 1000$ für kleines t (in VL $t = 8$)
- Programmname [Klartexte] [Kryptotexte]