Datensicherheit, Übung 7

Henry Haustein

Aufgabe 1

Erste Runde: 11000100, damit $L_0 = 1100$ und $R_0 = 0100$, damit $L_1 = R_0 = 0100$ und $R_1 = S(R_0 \oplus k_1) \oplus L_0 = S(0100 \oplus 0110) \oplus 1100 = 0001 \oplus 1100 = 1101 \Rightarrow 01001101$
Zweite Runde: $L_1 = 0100$, $R_1 = 1101$, $L_2 = R_1 = 1101$, $R_2 = S(R_1 \oplus k_2) \oplus L_1 = S(1101 \oplus 1010) \oplus 0100 = S(0111) \oplus 0100 = 1111 \oplus 0100 = 1011 \Rightarrow 11011011$

Entschlüsselung erste Runde: $L_2 = 1101$, $R_2 = 1011$, $R_1 = 1101$, $L_1 = S(L_2 \oplus k_2) \oplus R_2 = S(1101 \oplus 1010) \oplus 1011 = S(0111) \oplus 1011 = 1111 \oplus 1011 = 0100 \Rightarrow 01001101$
Entschlüsselung zweite Runde: $L_1 = 0100$ und $R_1 = 1101$, damit $R_0 = 0100$ und $L_0 = S(L_1 \oplus k_1) \oplus R_1 = S(0100 \oplus 0110) \oplus 1101 = S(0010) \oplus 1101 = 1100 \Rightarrow 11000100$

Aufgabe 2

(a) $S(m) = 0111$, abgespeicherte Werte sind unterstrichen

k_i **bei** $t = 1$	**enc()**	**T()**	**enc()**	**T()**	**enc()**
<u>1010</u>	1101	1011	1100	1001	<u>1110</u>
<u>0101</u>	0010	0100	0011	0110	<u>0001</u>

Wir finden $c = 0011$ nicht in der Tabelle, deswegen $c' = enc(T(c), m) = 0110 \oplus 0111 = 0001$. Dieser Wert findet sich, der Startschlüssel war 0101. Neuberechnung dieser Kette liefert nach der zweiten Iteration den gesuchten Ciphertext von 0011, damit ist der gesuchte Schlüssel $k = 0100$.

(b) $\frac{2 \cdot 3}{2^4} = \frac{3}{8}$

Aufgabe 3

Nachricht bleibt gleich: Eine Runde in DES ist eine Feistel-Chiffre, und damit selbstinvers. Entschlüsselung = Verschlüsselung, wenn der Schlüssel gleich ist, was hier gegeben ist.

Aufgabe 4

(a) 2^{56} Bits zu knacken, mit 2 Millionen Schlüssel pro Sekunde $\frac{2^{56}}{2000000} = 3.6 \cdot 10^{10}$ Sekunden ≈ 1142 Jahre.

(b) Im Schnitt wird der Angreifer nur die Hälfte der Schlüssel testen müssen, er ist also nach 571 Jahren fertig.

(c) $\frac{2^{56}}{65280000000} = 1.1 \cdot 10^6$ Sekunden ≈ 12.78 Tage

(d) Wieder nur die Hälfte, also weniger als eine Woche.

siehe: https://de.wikipedia.org/wiki/Copacobana

Aufgabe 5

Jedes (k, c)-Paar abspeichern, jedes Paar ist $56 + 64 = 120$ Bit, es gibt 2^{56} Paare, damit 1.08 EB.

Zeitaufwand: 2^{56} Verschlüsselungen und sortieren nach c für schnelleres Suchen mittels binärer Suche

Aufgabe 6

Meet-in-the-middle-Angriff, zweifache Verschlüsselung (Aufwand $2^{56} \cdot 2^{56} = 2^{112}$) und einfache Entschlüsselung (Aufwand 2^{56})

Speicheraufwand: 2^{56} nur für die einfache Entschlüsselung reicht.