

Einführung in die Kryptologie

Johannes Köbler



Institut für Informatik
Humboldt-Universität zu Berlin

SS 2022

Produktchiffren

- Produktchiffren erhält man durch die sequentielle Anwendung mehrerer Verschlüsselungsverfahren
- Sie können extrem schwer zu brechen sein, auch wenn die einzelnen Komponenten leicht zu brechen sind

Definition

- Seien $KS_i = (M_i, C_i, E_i, D_i, K_i, S_i)$, $i \in \{1, 2\}$, Kryptosysteme mit $C_1 = M_2$
- Dann ist das **Produktkryptosystem** $KS_1 \times KS_2$ von KS_1 und KS_2 definiert als $(M_1, C_2, E, D, K_1 \times K_2, S)$ mit $S = (S_1, S_2)$ und

$$E(k_1, k_2; x) = E_2(k_2, E_1(k_1, x)) \text{ sowie } D(k_1, k_2; y) = D_1(k_1, D_2(k_2, y))$$
 für alle $x \in M_1$, $y \in C_2$ und $(k_1, k_2) \in K_1 \times K_2$
- Der Schlüsselraum von $KS_1 \times KS_2$ umfasst also alle Schlüsselpaare $(k_1, k_2) \in K_1 \times K_2$, wobei wir voraussetzen, dass die beiden Schlüssel unabhängig gewählt werden (d.h. es gilt $p(k_1, k_2) = p(k_1)p(k_2)$)

Beispiel

- Bilden wir das Produkt $KS = KS^* \times KS^+$ der
 - multiplikativen Chiffre $KS^* = (M, C, \mathbb{Z}_m^*, E^*, D^*)$ mit $M = C = \mathcal{A} = \{a_0, \dots, a_{m-1}\}$ und der
 - additiven Chiffre $KS^+ = (M, C, \mathbb{Z}_m, E^+, D^+)$,

so gilt für jeden Schlüssel $(k_1, k_2) \in K = \mathbb{Z}_m^* \times \mathbb{Z}_m$:

$$E(k_1, k_2; x) = E^+(k_2, E^*(k_1, x)) = k_1x + k_2$$

- Dies bedeutet, dass das Produkt $KS^* \times KS^+ = (M, C, K, E, D)$ der multiplikativen und additiven Chiffre die affine Chiffre ist
- Welche Chiffre erhalten wir, wenn wir das Produkt $KS^+ \times KS^*$ der additiven und der multiplikativen Chiffre bilden?

Beispiel (Fortsetzung)

- Das Produkt $KS^+ \times KS^*$ ist das Kryptosystem $KS' = (M, C, K', E', D')$, in dem für jeden Schlüssel $(k_2, k_1) \in K' = \mathbb{Z}_m \times \mathbb{Z}_m^*$ gilt:

$$E'(k_2, k_1; x) = k_1(x + k_2) = k_1x + k_1k_2 = E(k_1, k_1k_2; x)$$

- Die Abbildung $(k_2, k_1) \mapsto (k_1, k_1k_2)$ ist also eine Bijektion zwischen den Schlüsselräumen $\mathbb{Z}_m \times \mathbb{Z}_m^*$ und $\mathbb{Z}_m^* \times \mathbb{Z}_m$, die jeden Schlüssel $(k_2, k_1) \in \mathbb{Z}_m \times \mathbb{Z}_m^*$ auf einen Schlüssel $(k_1, k_1k_2) \in \mathbb{Z}_m^* \times \mathbb{Z}_m$ mit $E_{(k_1, k_1k_2)} = E'_{(k_2, k_1)}$ abbildet
- Somit können wir auch jeden Schlüsselgenerator \mathcal{S} für $KS^* \times KS^+$ in einen Generator \mathcal{S}' für $KS^+ \times KS^*$ transformieren (und \mathcal{S}' auch wieder zurück in \mathcal{S}), so dass \mathcal{S}' in $KS^+ \times KS^*$ jede Chiffrierfunktion mit der gleichen Wahrscheinlichkeit erzeugt wie \mathcal{S} in $KS^* \times KS^+$
- Daher können die beiden Kryptosysteme $KS^* \times KS^+$ und $KS^+ \times KS^*$ als gleich (oder besser **äquivalent**, siehe Übungen) angesehen werden, d.h. die beiden Kryptosysteme KS^* und KS^+ kommutieren

Definition

- Ein Kryptosystem $KS = (M, C, K, D, E)$ mit $M = C$ heißt **endomorph**
- Ein endomorphes Kryptosystem KS heißt **idempotent**, falls $KS \times KS$ äquivalent zu KS ist (in Zeichen: $KS \times KS = KS$)

Beispiel

- Eine leichte Rechnung zeigt, dass
 - die additive Chiffre,
 - die multiplikative Chiffre und
 - die affine Chiffreidempotent sind
- Dies trifft auch auf
 - die Blocktransposition sowie
 - die Vigenère- und Hill-Chiffrezu (siehe Übungen)

- Will man durch mehrmalige Anwendung (Iteration) derselben Chiffre eine höhere Sicherheit erreichen, so darf diese nicht idempotent sein
- Man kann versuchen, durch sequentielle Ausführung zweier idempotenter Systeme KS_1 und KS_2 ein System $KS = KS_1 \times KS_2$ zu erhalten, das nicht idempotent ist
- Da KS im Fall $KS_1 \times KS_2 = KS_2 \times KS_1$ wegen

$$\begin{aligned}KS \times KS &= (KS_1 \times KS_2) \times (KS_1 \times KS_2) \\ &= KS_1 \times (KS_2 \times KS_1) \times KS_2 \\ &= KS_1 \times (KS_1 \times KS_2) \times KS_2 \\ &= (KS_1 \times KS_1) \times (KS_2 \times KS_2) \\ &= KS_1 \times KS_2 \\ &= KS\end{aligned}$$

idempotent ist, dürfen hierbei KS_1 und KS_2 jedoch nicht kommutieren

- Im Folgenden betrachten wir Blockchiffren über dem Binäralphabet $A = \{0, 1\}$ und auch der Schlüsselraum wird von der Form $\{0, 1\}^k$ sein
- Die Schlüssellänge bezeichnen wir bis auf weiteres mit k und einzelne Schlüssel eines Kryptosystems mit K
- Eine **iterierte Blockchiffre** wird durch eine **Rundenfunktion** (round function) g und einen **Key-Schedule Algorithmus** f beschrieben
- Ist N die Rundenzahl, so erzeugt f bei Eingabe eines Schlüssels K eine Folge $f(K) = (K^1, \dots, K^N)$ von N **Rundenschlüsseln** K^i für g

Iterierte Blockchiffren

- Mit diesen wird ein Klartext $x = w^0$ durch N -malige Anwendung der Rundenfunktion g zu einem Kryptotext $y = w^N$ verschlüsselt:

$$w^1 := g(K^1, w^0)$$

$$\vdots$$

$$w^N := g(K^N, w^{N-1})$$

- Um y wieder zu entschlüsseln, muss die inverse Rundenfunktion g^{-1} mit umgekehrter Rundenschlüsselreihe K^N, \dots, K^1 benutzt werden:

$$w^{N-1} := g^{-1}(K^N, w^N)$$

$$\vdots$$

$$w^0 := g^{-1}(K^1, w^1)$$

- Beispiele für iterierte Chiffren sind der aus 16 Runden bestehende **DES**-Algorithmus und der **AES** mit einer variablen Rundenzahl $N \in \{10, 12, 14\}$, die wir später behandeln werden

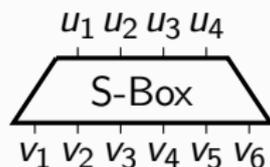
- Als Basisbausteine für die Rundenfunktion von iterierten Blockchiffren eignen sich Substitutionen und Transpositionen besonders gut
- Aus Effizienzgründen sollten die Substitutionen nur eine relativ kleine Blocklänge haben

Definition

- Für ein Wort $u = u_1 \cdots u_n \in \{0, 1\}^n$ und Indizes $1 \leq i \leq j \leq n$ bezeichne $u[i, j]$ das **Teilwort** $u_i \cdots u_j$ von u
- Im Fall $n = ml$ bezeichnen wir das Teilwort $u[(i-1)l + 1, il]$ auch einfach mit $u_{(i)}$, d.h. es gilt $u = u_{(1)} \cdots u_{(m)}$, wobei $|u_{(i)}| = l$ ist

Substitutions-Permutations-Netzwerke

- Sei $\sigma_S : \{0, 1\}^l \rightarrow \{0, 1\}^{l'}$ eine Substitution, die Binärblöcke u der Länge l in Blöcke $v = \sigma_S(u)$ der Länge l' überführt (auch **S-Box** S genannt)



- Für $\sigma_S(u)$ schreiben wir auch einfach $S(u)$
- Durch parallele Anwendung von m Kopien der S-Box S erhalten wir die Substitution $\sigma_{mS} : \{0, 1\}^{ml} \rightarrow \{0, 1\}^{ml'}$ mit

$$\sigma_{mS}(u_1 \cdots u_{ml}) = S(u_{(1)}) \cdots S(u_{(m)})$$

- Auch hier schreiben wir für $\sigma_{mS}(u_1 \cdots u_{ml})$ auch einfach $S(u_1 \cdots u_{ml})$
- Für die Speicherung einer S-Box S mit $\sigma_S : \{0, 1\}^l \rightarrow \{0, 1\}^{l'}$ auf einem Chip werden $l' \cdot 2^l$ Bit Speicherplatz benötigt (im Fall $l = l'$ also $l2^l$ Bit)
- Für $l = l' = 16$ wären dies beispielsweise 2^{20} Bit, was Smartcard-Anwendungen bereits ausschließen würde
- Für eine Transposition P auf $\{0, 1\}^\ell$ bezeichnen wir die zugehörige Permutation auf $[\ell]$ mit π_P oder einfach mit π , falls P aus dem Kontext bekannt ist, d.h. $P(u_1 \cdots u_\ell) = u_{\pi(1)} \cdots u_{\pi(\ell)}$

Definition

- Für natürliche Zahlen $m, l \geq 1$ sei $M = C = \{0, 1\}^\ell$ mit $\ell = ml$
- Ein **Substitutions-Permutations-Netzwerk (SPN)** wird durch eine S-Box S , eine Blocktransposition P mit Blocklänge $\ell = ml$ und durch eine Funktion $f : \{0, 1\}^k \rightarrow \{0, 1\}^{\ell(N+1)}$ beschrieben
- Hierbei realisiert die S-Box S eine Permutation σ_S auf $\{0, 1\}^l$ und N ist die **Rundenzahl** des SPN
- Die Funktion f transformiert einen (externen) Schlüssel $K \in \{0, 1\}^k$ in ein **Key-Schedule** $f(K) = (K^1, \dots, K^{N+1})$ von $N + 1$ **Rundenschlüsseln**
- Unter ihnen wird ein Klartext $x \in \{0, 1\}^\ell$ durch folgenden Chiffrieralgorithmus in einen Kryptotext $y = E_{f,S,P}(K, x) \in \{0, 1\}^\ell$ überführt:

```

1   $w^0 := x$ 
2  for  $r := 1$  to  $N - 1$  do
3       $u^r := w^{r-1} \oplus K^r$ 
4       $v^r := S(u^r)$ 

```

```

5       $w^r := P(v^r)$ 
6       $u^N := w^{N-1} \oplus K^N$ 
7       $v^N := S(u^N)$ 
8       $y := v^N \oplus K^{N+1}$ 

```

Substitutions-Permutations-Netzwerke

Die Chiffrierfunktion $E_{f,S,P}(K, x)$

- Zu Beginn jeder Runde $r \in \{1, \dots, N\}$ wird w^{r-1} zuerst einer XOR-Operation mit dem Rundenschlüssel K^r unterworfen (**round key mixing**) und das Resultat u^r den S-Boxen zugeführt
- Auf die Ausgabe v^r der S-Boxen wird in jeder Runde $r \leq N - 1$ die Transposition P angewendet, was die Eingabe w^r für die nächste Runde $r + 1$ liefert
- Am Ende der letzten Runde $r = N$ wird nicht die Transposition P angewandt, sondern der Rundenschlüssel K^{N+1} auf v^N addiert
- Dies wird **whitening** genannt und bewirkt, dass auch für den letzten Chiffrierschritt der Schlüssel benötigt und somit der Gegner an einer partiellen Entschlüsselung des Kryptotextes gehindert wird
- Zudem wird dadurch eine (legale) Entschlüsselung nach fast demselben Verfahren ermöglicht (siehe Übungen)

```

1   $w^0 := x$ 
2  for  $r := 1$  to  $N - 1$  do
3       $u^r := w^{r-1} \oplus K^r$ 
4       $v^r := S(u^r)$ 
5       $w^r := P(v^r)$ 
6   $u^N := w^{N-1} \oplus K^N$ 
7   $v^N := S(u^N)$ 
8   $y := v^N \oplus K^{N+1}$ 
  
```

- Die S-Boxen sorgen dafür, dass jedes einzelne Bit des Kryptotextes y von mehreren Bits des Klartextes und des Schlüssels abhängt
- Wichtig ist hierbei, dass die Abhängigkeit möglichst komplex ist (also z.B. nicht linear)
- Dadurch werden Angriffe erschwert, die versuchen, Rückschlüsse vom Kryptotext auf den Schlüssel oder Klartext zu ziehen (Shannon nannte diese Eigenschaft **Konfusion**)
- Die Transpositionen über den gesamten Block sorgen dafür, dass sich eine Änderung von einzelnen Bits des Klartextes oder des Schlüssels potentiell auf jedes Kryptotextbit auswirken kann (von Shannon als **Diffusion** bezeichnet)
- Idealerweise wird hierdurch erreicht, dass sich bei Änderung eines einzelnen Eingabe- oder Schlüsselbits jedes Ausgabebit mit Wahrscheinlichkeit $1/2$ ändert

Beispiel

- Wir betrachten ein SPN SP mit Parametern $l = m = N = 4$ und $k = 32$
- Für f wählen wir die Funktion $f(K) = (K^1, \dots, K^5)$ mit
 $K^r = K[4(r-1) + 1, 4(r-1) + 16]$
- Weiter seien $S : \{0, 1\}^4 \rightarrow \{0, 1\}^4$ und $\sigma_P : \{1, \dots, 16\} \rightarrow \{1, \dots, 16\}$ die folgenden Permutationen:

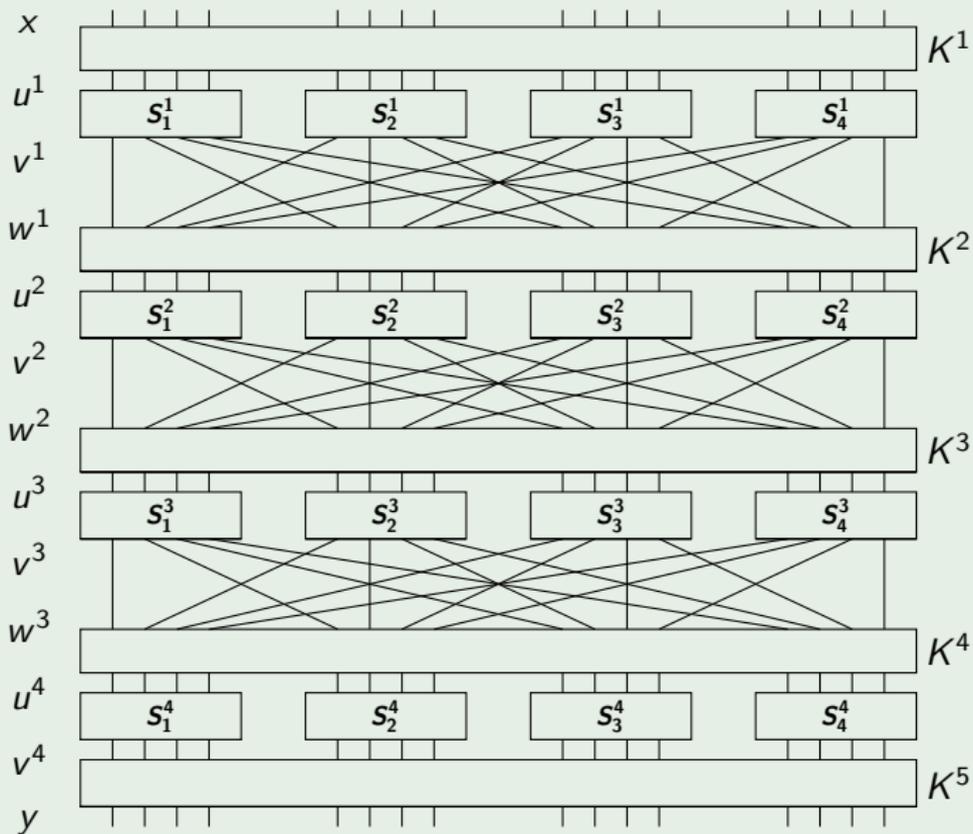
z	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
$S(z)$	E	4	D	1	2	F	B	8	3	A	6	C	5	9	0	7

wobei die Argumente und Werte von S hexadezimal dargestellt sind,
und

i	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
$\sigma_P(i)$	1	5	9	13	2	6	10	14	3	7	11	15	4	8	12	16

Substitutions-Permutations-Netzwerke

Beispiel (Fortsetzung)



Beispiel (Schluss)

- Beispielsweise liefert f für den Schlüssel $K = 0011\ 1010\ 1001\ 0100\ 1101\ 0110\ 0011\ 1111$ die Rundenschlüssel $f(K) = (K^1, \dots, K^5)$ mit

$$K^1 = 0011\ 1010\ 1001\ 0100 \quad K^2 = 1010\ 1001\ 0100\ 1101$$

$$K^3 = 1001\ 0100\ 1101\ 0110 \quad K^4 = 0100\ 1101\ 0110\ 0011$$

$$K^5 = 1101\ 0110\ 0011\ 1111$$

unter denen der Klartext $x = 0010\ 0110\ 1011\ 0111$ die folgenden Chiffrierschritte durchläuft:

$$x = 0010\ 0110\ 1011\ 0111 = w^0$$

$$w^0 \oplus K^1 = 0001\ 1100\ 0010\ 0011 = u^1$$

$$S(u^1) = 0100\ 0101\ 1101\ 0001 = v^1$$

$$P(v^1) = 0010\ 1110\ 0000\ 0111 = w^1$$

$$\vdots$$

$$P(v^3) = 1110\ 0100\ 0110\ 1110 = w^3$$

$$w^3 \oplus K^4 = 1010\ 1001\ 0000\ 1101 = u^4$$

$$S(u^4) = 0110\ 1010\ 1110\ 1001 = v^4$$

$$u^4 \oplus K^5 = 1011\ 1100\ 1101\ 0110 = y$$

Lineare Approximationen

- Sei $f : \{0, 1\}^l \rightarrow \{0, 1\}^{l'}$ eine boolesche Funktion
- Wählen wir die Eingabe $\mathcal{U} = \mathcal{U}_1 \cdots \mathcal{U}_l$ zufällig unter Gleichverteilung, so gilt für die zugehörige Ausgabe $\mathcal{V} = f(\mathcal{U}) = \mathcal{V}_1 \cdots \mathcal{V}_{l'}$ sowie für alle $u \in \{0, 1\}^l$ und $v \in \{0, 1\}^{l'}$:

$$\Pr[\mathcal{V} = v \mid \mathcal{U} = u] = \begin{cases} 1 & f(u) = v \\ 0 & \text{sonst} \end{cases}$$

- Wegen $\Pr[\mathcal{U} = u] = 2^{-l}$ folgt

$$\Pr[\mathcal{V} = v, \mathcal{U} = u] = \begin{cases} 2^{-l} & f(u) = v, \\ 0 & \text{sonst} \end{cases}$$

- Die Funktion f ist **linear**, wenn eine binäre $(l \times l')$ -Matrix A mit $f(u) = uA$ existiert
- In diesem Fall ist jedes Ausgabebit v_j in der Form $v_j = u_{i_1} \oplus \cdots \oplus u_{i_k}$ mit $1 \leq i_1 < \cdots < i_k \leq l$ darstellbar, d.h. $\Pr[\mathcal{V}_j = \mathcal{U}_{i_1} \oplus \cdots \oplus \mathcal{U}_{i_k}] = 1$

Lineare Approximationen

- Für eine lineare Kryptoanalyse benötigen wir Gleichungen der Form

$$\mathcal{V}_{j_1} \oplus \cdots \oplus \mathcal{V}_{j_{k'}} = \mathcal{U}_{i_1} \oplus \cdots \oplus \mathcal{U}_{i_k} \oplus c$$

mit $1 \leq i_1 < \cdots < i_k \leq l$, $1 \leq j_1 < \cdots < j_{k'} \leq l'$ und $c \in \{0, 1\}$, die mit möglichst großer Wahrscheinlichkeit gelten

- Definieren wir für $u, a \in \{0, 1\}^l$ und $v, b \in \{0, 1\}^{l'}$ die xor-Teilsummen

$$u_a = \bigoplus_{i=1}^l a_i u_i \quad \text{und} \quad v_b = \bigoplus_{i=1}^{l'} b_i v_i,$$

so suchen wir also nach Werten für a , b und c , für die das Ereignis $\mathcal{V}_b = \mathcal{U}_a \oplus c$ (oder $\mathcal{U}_a \oplus \mathcal{V}_b = c$) mit großer Wahrscheinlichkeit eintritt

- In diesem Fall lässt sich nämlich der Wert von \mathcal{V}_b bei Kenntnis von \mathcal{U}_a entsprechend gut vorhersagen
- Wegen $\Pr[\mathcal{U}_a \oplus \mathcal{V}_b = c] = 1 - \Pr[\mathcal{U}_a \oplus \mathcal{V}_b = c \oplus 1]$ kommt es nur darauf an, wie stark die Wahrscheinlichkeit $\Pr[\mathcal{U}_a \oplus \mathcal{V}_b = 0]$ von $1/2$ abweicht
- \mathcal{V}_b lässt sich also in Abhängigkeit von \mathcal{U}_a um so besser vorhersagen, je größer der Absolutbetrag $|\Pr[\mathcal{U}_a \oplus \mathcal{V}_b = 0] - 1/2|$ ist

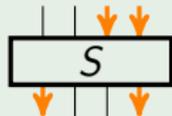
Definition

- Der **Bias** einer Zufallsvariablen \mathcal{X} mit Wertebereich $W(\mathcal{X}) = \{0, 1\}$ ist definiert als $\beta(\mathcal{X}) = \Pr[\mathcal{X} = 0] - 1/2$
- Eine **lineare Approximation** an eine boolesche Funktion $f : \{0, 1\}^l \rightarrow \{0, 1\}^l$ wird durch ein Paar $(a, b) \in \{0, 1\}^l \times \{0, 1\}^l$ beschrieben
- Die **Güte** der durch (a, b) beschriebenen Approximation an f ist der doppelte Absolutbetrag $\gamma_f(a, b) = 2|\beta(\mathcal{U}_a \oplus \mathcal{V}_b)|$ des Bias-Wertes von $\mathcal{U}_a \oplus \mathcal{V}_b$, wobei \mathcal{U} auf $\{0, 1\}^l$ gleichverteilt und $\mathcal{V} = f(\mathcal{U})$ ist

Lineare Approximationen

Beispiel

- Wir betrachten die durch das Paar $(0011, 1001)$ beschriebene lineare Approximation $\mathcal{A} = \mathcal{U}_3 \oplus \mathcal{U}_4 \oplus \mathcal{V}_1 \oplus \mathcal{V}_4$ an die S-Box S aus vorigem Beispiel



- Dann nimmt die Zufallsvariable $(\mathcal{U}_1, \dots, \mathcal{U}_4, \mathcal{V}_1, \dots, \mathcal{V}_4, \mathcal{A})$ die 16 Werte in folgender Tabelle jeweils mit Wahrscheinlichkeit $2^{-4} = 1/16$ an:

u_1	u_2	u_3	u_4	v_1	v_2	v_3	v_4	$u_3 \oplus u_4 \oplus v_1 \oplus v_4$
0	0	0	0	1	1	1	0	1
0	0	0	1	0	1	0	0	1
0	0	1	0	1	1	0	1	1
0	0	1	1	0	0	0	1	1
0	1	0	0	0	0	1	0	0
0	1	0	1	1	1	1	1	1
0	1	1	0	1	0	1	1	1
0	1	1	1	1	0	0	0	1
1	0	0	0	0	0	1	1	1
1	0	0	1	1	0	1	0	0
1	0	1	0	0	1	1	0	1
1	0	1	1	1	1	0	0	1
1	1	0	0	0	1	0	1	1
1	1	0	1	1	0	0	1	1
1	1	1	0	1	0	0	1	1
1	1	1	1	0	0	0	0	1
1	1	1	1	0	1	1	1	1

Beispiel (Fortsetzung)

- Um $\beta(\mathcal{U}_a \oplus \mathcal{V}_b)$ zu berechnen, genügt es, die Anzahl $L(a, b)$ der Eingaben $u \in \{0, 1\}^l$ zu bestimmen, für die $f(u)_b = u_a$ ist
- Dann gilt $\Pr[\mathcal{U}_a \oplus \mathcal{V}_b = 0] = \Pr[\mathcal{U}_a = \mathcal{V}_b] = L(a, b)/16$ und somit

$$\beta(\mathcal{U}_a \oplus \mathcal{V}_b) = L(a, b)/16 - 1/2 = (L(a, b) - 8)/16$$

sowie

$$\gamma(a, b) = |2\beta(\mathcal{U}_a \oplus \mathcal{V}_b)|$$

- Für $a = 0011$ und $b = 1001$ gibt es z.B. $L(0011, 1001) = 2$ Eingaben u mit $u_{0011} = S(u)_{1001}$ ($u = 0100$ und $u = 1001$), d.h. es gilt

$$\beta(\mathcal{U}_{0011} \oplus \mathcal{V}_{1001}) = (L(3, 9) - 8)/16 = (2 - 8)/16 = -3/8$$

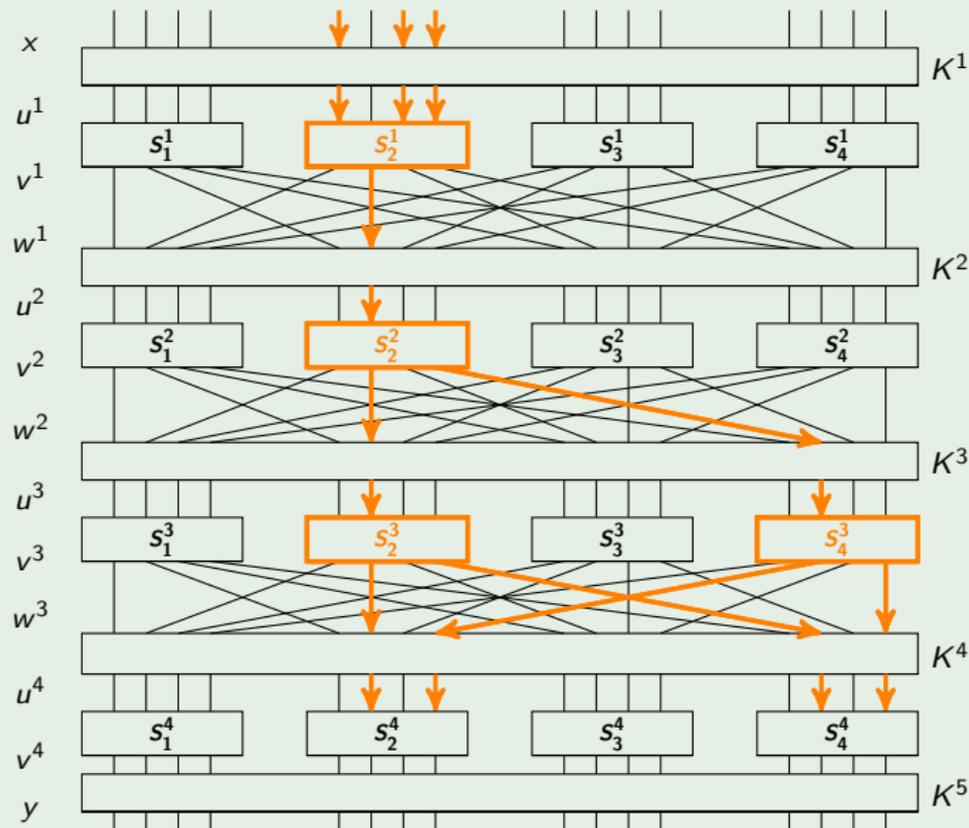
- Die Güte von \mathcal{A} ist also $\gamma(0011, 1001) = |2\beta(\mathcal{U}_{0011} \oplus \mathcal{V}_{1001})| = 3/4$
- Die Tabelle auf der nächsten Folie enthält die Anzahlen $L(a, b)$ für bestimmte Werte von a und b (diese sind hexadezimal dargestellt)

Beispiel (Fortsetzung)

	<i>b</i>															
<i>a</i>	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	16	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8
1	8	8	6	6	8	8	6	14	10	10	8	8	10	10	8	8
2	8	8	6	6	8	8	6	6	8	8	10	10	8	8	2	10
3	8	8	8	8	8	8	8	8	10	2	6	6	10	10	6	6
4	8	10	8	6	6	4	6	8	8	6	8	10	10	4	10	8
								⋮								
B	8	12	8	4	12	8	12	8	8	8	8	8	8	8	8	8
								⋮								
F	8	6	4	6	6	8	10	8	8	6	12	6	6	8	10	8

Unser nächstes Ziel ist, geeignete lineare Approximationen für bestimmte S-Boxen im SPN zu finden, die sich zu einer linearen Approximation der Abbildung $x \mapsto u^4$ zusammenschalten lassen (siehe nächste Folie)

Beispiel (Schluss)



Lineare Kryptoanalyse von SPNs

- Seien K^1, \dots, K^5 die aus K berechneten Rundenschlüssel (diese sind wie K unbekannt, aber konstant)
- Unser erstes Ziel ist, eine lineare Approximation \mathcal{A} an die Abbildung $x \mapsto u^4$ zu finden
- Diese Abbildung benutzt die vier Rundenschlüssel K^1, \dots, K^4 , um den Klartext x in den Block u^4 zu überführen, welcher als Eingabe für die S-Boxen in Runde 4 dient, auf deren Ausgabe $v^4 = S(u^4)$ der Rundenschlüssel K^5 addiert wird, um den Kryptotext $y = v^4 \oplus K^5$ zu erhalten
- Für die Konstruktion von \mathcal{A} verwenden wir die durch $(B, 4)$ und $(4, 5)$ beschriebenen linearen Approximationen

$$\mathcal{T} = \mathcal{U}_1 \oplus \mathcal{U}_3 \oplus \mathcal{U}_4 \oplus \mathcal{V}_2 \quad \text{und} \quad \mathcal{T}' = \mathcal{U}_2 \oplus \mathcal{V}_2 \oplus \mathcal{V}_4$$

an die S-Box S mit den Bias-Werten

- $\beta(\mathcal{T}) = (L(1011, 0100) - 8)/16 = (12 - 8)/16 = 1/4$ und
- $\beta(\mathcal{T}') = (L(0100, 0101) - 8)/16 = (4 - 8)/16 = -1/4$

- Konkret verwenden wir \mathcal{T} für die S-Box S_2^1 :

$$\mathcal{T}_1 = \mathcal{U}_5^1 \oplus \mathcal{U}_7^1 \oplus \mathcal{U}_8^1 \oplus \mathcal{V}_6^1$$

und \mathcal{T}' für die drei S-Boxen S_2^2 , S_2^3 und S_4^3 :

$$\mathcal{T}_2 = \mathcal{U}_6^2 \oplus \mathcal{V}_6^2 \oplus \mathcal{V}_8^2, \mathcal{T}_3 = \mathcal{U}_6^3 \oplus \mathcal{V}_6^3 \oplus \mathcal{V}_8^3 \text{ und } \mathcal{T}_4 = \mathcal{U}_{14}^3 \oplus \mathcal{V}_{14}^3 \oplus \mathcal{V}_{16}^3$$

- Nun schalten wir die vier linearen Approximationen $\mathcal{T}_1, \dots, \mathcal{T}_4$ an die S-Boxen S_2^1 , S_2^2 , S_2^3 und S_4^3 zur linearen Approximation

$$\mathcal{A} = \underbrace{\mathcal{X}_5 \oplus \mathcal{X}_7 \oplus \mathcal{X}_8}_{\mathcal{X}_a \text{ für } a=0x0B00} \oplus \underbrace{\mathcal{U}_6^4 \oplus \mathcal{U}_8^4 \oplus \mathcal{U}_{14}^4 \oplus \mathcal{U}_{16}^4}_{\mathcal{U}_b^4 \text{ für } b=0x0505}$$

an die Abbildung $x \mapsto u^4$ zusammen

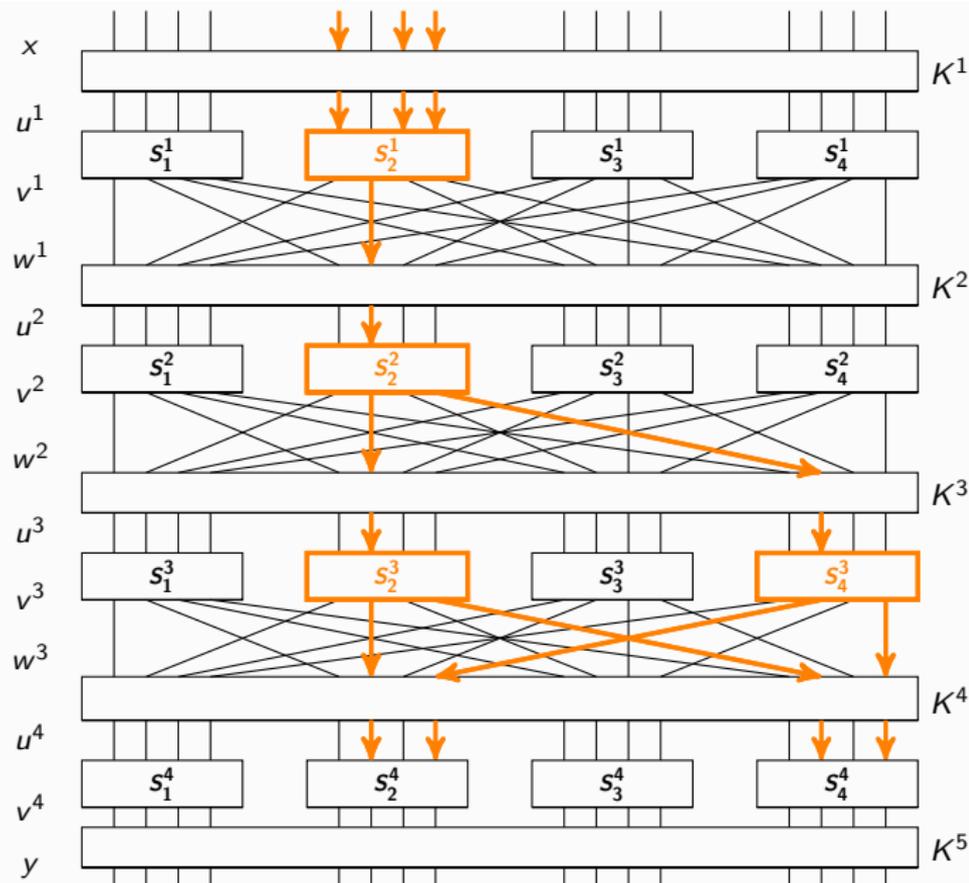
- Dann gilt für ein Bit $c \in \{0, 1\}$ die Gleichung

$$\mathcal{X}_a \oplus \mathcal{U}_b^4 = \mathcal{T}_1 \oplus \mathcal{T}_2 \oplus \mathcal{T}_3 \oplus \mathcal{T}_4 \oplus c$$

An dieser Stelle ergeben sich nun folgende drei Fragen:

- ➊ Warum gilt die Gleichung $\mathcal{X}_a \oplus \mathcal{U}_b^4 = \mathcal{T}_1 \oplus \mathcal{T}_2 \oplus \mathcal{T}_3 \oplus \mathcal{T}_4 \oplus c$?
- ➋ Wie gut ist die durch (a, b) beschriebene lineare Approximation $\mathcal{A} = \mathcal{X}_a \oplus \mathcal{U}_b^4$ an die Abbildung $x \mapsto u^4$?
- ➌ Wie können wir die Approximation \mathcal{A} benutzen, um einzelne Schlüsselbits zu bestimmen?

Warum ist $\mathcal{X}_a \oplus \mathcal{U}_b^4 = \mathcal{T}_1 \oplus \mathcal{T}_2 \oplus \mathcal{T}_3 \oplus \mathcal{T}_4 \oplus c$?



Warum ist $\mathcal{X}_a \oplus \mathcal{U}_b^4 = \mathcal{T}_1 \oplus \mathcal{T}_2 \oplus \mathcal{T}_3 \oplus \mathcal{T}_4 \oplus c$?

- Sei $c_1 = K_5^1 \oplus K_7^1 \oplus K_8^1$, $c_2 = K_6^2$, $c_3 = K_6^3 \oplus K_{14}^3$, $c_4 = K_6^4 \oplus K_8^4 \oplus K_{14}^4 \oplus K_{16}^4$
- Wegen $\mathcal{T}_1 = \mathcal{U}_5^1 \oplus \mathcal{U}_7^1 \oplus \mathcal{U}_8^1 \oplus \mathcal{V}_6^1$, $\mathcal{T}_2 = \mathcal{U}_6^2 \oplus \mathcal{V}_6^2 \oplus \mathcal{V}_8^2$, $\mathcal{T}_3 = \mathcal{U}_6^3 \oplus \mathcal{V}_6^3 \oplus \mathcal{V}_8^3$ und $\mathcal{T}_4 = \mathcal{U}_{14}^3 \oplus \mathcal{V}_{14}^3 \oplus \mathcal{V}_{16}^3$ folgt dann

$$\begin{aligned}
 \mathcal{X}_a &= \mathcal{X}_5 \oplus \mathcal{X}_7 \oplus \mathcal{X}_8 \\
 &= \mathcal{U}_5^1 \oplus \mathcal{U}_7^1 \oplus \mathcal{U}_8^1 \oplus c_1 \\
 &= \mathcal{T}_1 \oplus \mathcal{V}_6^1 \oplus c_1 \\
 &= \mathcal{T}_1 \oplus \mathcal{W}_6^1 \oplus c_1 \\
 &= \mathcal{T}_1 \oplus \mathcal{U}_6^2 \oplus c_1 \oplus c_2 \\
 &= \mathcal{T}_1 \oplus \mathcal{T}_2 \oplus \mathcal{V}_6^2 \oplus \mathcal{V}_8^2 \oplus c_1 \oplus c_2 \\
 &= \mathcal{T}_1 \oplus \mathcal{T}_2 \oplus \mathcal{W}_6^2 \oplus \mathcal{W}_{14}^2 \oplus c_1 \oplus c_2 \\
 &= \mathcal{T}_1 \oplus \mathcal{T}_2 \oplus \mathcal{U}_6^3 \oplus \mathcal{U}_{14}^3 \oplus c_1 \oplus c_2 \oplus c_3 \\
 &= \mathcal{T}_1 \oplus \mathcal{T}_2 \oplus \mathcal{T}_3 \oplus \mathcal{T}_4 \oplus \mathcal{V}_6^3 \oplus \mathcal{V}_8^3 \oplus \mathcal{V}_{14}^3 \oplus \mathcal{V}_{16}^3 \oplus c_1 \oplus c_2 \oplus c_3 \\
 &= \mathcal{T}_1 \oplus \mathcal{T}_2 \oplus \mathcal{T}_3 \oplus \mathcal{T}_4 \oplus \mathcal{W}_6^3 \oplus \mathcal{W}_8^3 \oplus \mathcal{W}_{14}^3 \oplus \mathcal{W}_{16}^3 \oplus c_1 \oplus c_2 \oplus c_3 \\
 &= \mathcal{T}_1 \oplus \mathcal{T}_2 \oplus \mathcal{T}_3 \oplus \mathcal{T}_4 \oplus \underbrace{\mathcal{U}_b^4 \oplus c_1 \oplus c_2 \oplus c_3 \oplus c_4}_{=: c}
 \end{aligned}$$

Wie gut ist die Approximation $\mathcal{A} = \mathcal{X}_a \oplus \mathcal{U}_b^4$?

- Wären die linearen Approximationen $\mathcal{T}_1, \dots, \mathcal{T}_4$ an die vier ausgewählten S-Boxen S_2^1, S_2^2, S_2^3 und S_4^3 (diese werden auch als **aktiv** bezeichnet) unabhängig, so würde uns das **Piling-up-Lemma** (siehe nächste Folie) die folgenden Bias-Werte liefern:

- $\beta(\mathcal{T}_1 \oplus \dots \oplus \mathcal{T}_4) = 2^3(1/4)(-1/4)^3 = -1/32$ und

- $\beta(\mathcal{A}) = (-1)^{c+1}/32$ bzw. $\gamma(\mathcal{A}) = 1/16$

- Sind nämlich $\mathcal{X}_1, \mathcal{X}_2$ unabhängige Zufallsvariablen mit Wertebereich $W(\mathcal{X}_i) = \{0, 1\}$ und Bias $\beta_i = \beta(\mathcal{X}_i)$, $i \in \{1, 2\}$, dann ist

$$\begin{aligned} \Pr[\mathcal{X}_1 \oplus \mathcal{X}_2 = 0] &= \Pr[\mathcal{X}_1 = \mathcal{X}_2 = 0] + \Pr[\mathcal{X}_1 = \mathcal{X}_2 = 1] \\ &= (1/2 + \beta_1)(1/2 + \beta_2) + (1/2 - \beta_1)(1/2 - \beta_2) \\ &= 1/2 + 2\beta_1\beta_2 \end{aligned}$$

- Es gilt also $\beta(\mathcal{X}_1 \oplus \mathcal{X}_2) = 2\beta_1\beta_2$
- Das Piling-up Lemma verallgemeinert dies auf die Summe von mehreren unabhängigen Zufallsvariablen

Wie gut ist die Approximation $\mathcal{A} = \mathcal{X}_a \oplus \mathcal{U}_b^4$?

Lemma (Piling-up Lemma)

Für unabhängige $\{0, 1\}$ -wertige Zufallsvariablen $\mathcal{X}_1, \dots, \mathcal{X}_n$ mit Bias-Werten $\beta_i = \beta(\mathcal{X}_i)$ gilt

$$\beta(\mathcal{X}_1 \oplus \dots \oplus \mathcal{X}_n) = 2^{n-1} \prod_{i=1}^n \beta_i$$

Beweis.

Wir führen den Beweis durch Induktion über n :

- Induktionsanfang ($n \leq 2$): Bereits bewiesen
- Induktionsschritt ($n \rightsquigarrow n+1$): Nach IV hat $\mathcal{Z} = \mathcal{X}_1 \oplus \dots \oplus \mathcal{X}_n$ den Bias

$$\beta(\mathcal{Z}) = 2^{n-1} \beta(\mathcal{X}_1) \cdots \beta(\mathcal{X}_n) = 2^{n-1} \beta_1 \cdots \beta_n$$

und daher folgt

$$\begin{aligned} \beta(\mathcal{X}_1 \oplus \dots \oplus \mathcal{X}_{n+1}) &= \beta(\mathcal{Z} \oplus \mathcal{X}_{n+1}) = 2\beta(\mathcal{Z})\beta_{n+1} \\ &= 2^n \beta_1 \cdots \beta_{n+1} \end{aligned}$$



Wie gut ist die Approximation $\mathcal{A} = \mathcal{X}_a \oplus \mathcal{U}_b^4$?

Beispiel

- Seien $\mathcal{X}_1, \mathcal{X}_2, \mathcal{X}_3$ unabhängige Zufallsvariablen mit $\beta(\mathcal{X}_i) = 1/4$ für $i = 1, 2, 3$
- Dann liefert das Piling-up Lemma die Bias-Werte $\beta(\mathcal{X}_i \oplus \mathcal{X}_j) = 1/8$ für $1 \leq i < j \leq 3$
- Da die beiden Zufallsvariablen $\mathcal{Y} = \mathcal{X}_1 \oplus \mathcal{X}_2$ und $\mathcal{Z} = \mathcal{X}_2 \oplus \mathcal{X}_3$ nicht unabhängig sind, liefert das Piling-up-Lemma nicht den richtigen Bias-Wert für $\beta(\mathcal{Y} \oplus \mathcal{Z})$
- Damit würden wir nämlich den Wert $2 \cdot 1/8 \cdot 1/8 = 1/32$ erhalten, wogegen

$$\beta(\mathcal{Y} \oplus \mathcal{Z}) = \beta(\mathcal{X}_1 \oplus \mathcal{X}_2 \oplus \mathcal{X}_2 \oplus \mathcal{X}_3) = \beta(\mathcal{X}_1 \oplus \mathcal{X}_3) = 1/8$$

ist

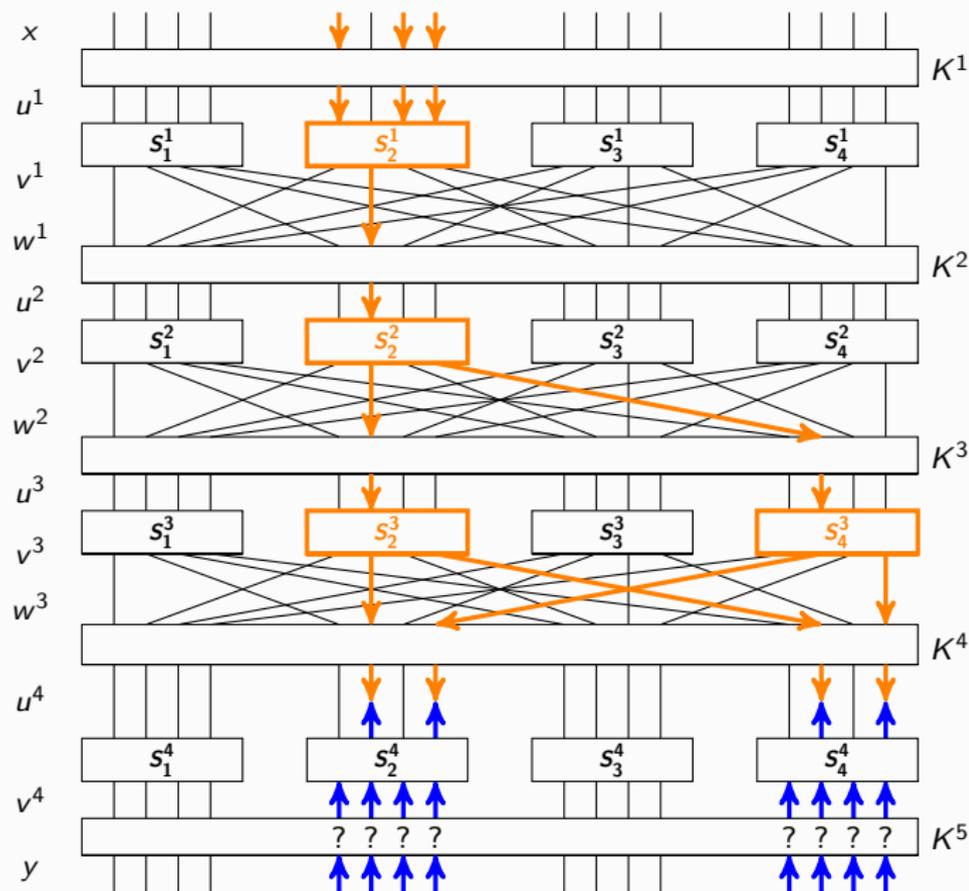
Wie gut ist die Approximation $\mathcal{A} = \mathcal{X}_a \oplus \mathcal{U}_b^4$?

- Zwar sind die Zufallsvariablen \mathcal{T}_i , aus denen eine lineare Approximation $\mathcal{X}_a \oplus \mathcal{U}_b^N = \mathcal{T}_1 \oplus \cdots \oplus \mathcal{T}_k \oplus c$ an die Abbildung $x \mapsto u^N$ gebildet wird, in der Regel nicht unabhängig
- Dennoch zeigt sich in praktischen Anwendungen, dass der tatsächliche Bias von $\mathcal{T}_1 \oplus \cdots \oplus \mathcal{T}_k$ meist nicht sehr von dem **hypothetischen Wert** $2^{k-1} \prod_{i=1}^k \beta(\mathcal{T}_i)$ unterscheidet, welcher sich aus dem Piling-up Lemma ergibt
- Daher können wir in unserem Beispiel

$$\beta(\mathcal{T}_1 \oplus \cdots \oplus \mathcal{T}_4) \approx 2^3(1/4)(-1/4)^3 = -1/32$$

bzw. $\gamma(\mathcal{A}) \approx 1/16$ annehmen

Wie lassen sich mit \mathcal{A} einzelne Schlüsselbits bestimmen? ²⁰⁹



Wie lassen sich mit \mathcal{A} einzelne Schlüsselbits bestimmen?²¹⁰

- Wir nehmen an, dass eine Menge M von Klartext-Kryptotext-Paaren bekannt ist und damit einzelne Schlüsselbits berechnet werden sollen
- Wir betrachten zuerst den (für den Gegner günstigen) Fall, dass die lineare Approximation $\mathcal{X}_a \oplus \mathcal{U}_b^4$ an $x \mapsto u^4$ die Güte $\gamma(\mathcal{X}_a \oplus \mathcal{U}_b^4) = 1$ hat (dies ist z.B. der Fall, wenn die S-Box S **affin** ist, d.h. $S(u) = uA \oplus w$)
- In diesem Fall führt jede Eingabe x auf einen Vektor u^4 mit $x_a \oplus u_b^4 = c$, wobei das Bit c nur von K und nicht von x abhängt
- Da y und S^{-1} bekannt sind, können wir für jeden Kandidaten L für K^5 den zugehörigen u^4 -Block $u^4(y, L) = S^{-1}(L \oplus y)$ berechnen
- Dann wird der Kandidat $L = K^5$ den Test $u^4(y, L)_b = x_a$ für alle $(x, y) \in M$ bestehen (falls $c = 0$ ist) oder für alle $(x, y) \in M$ nicht bestehen (falls $c = 1$ ist)

Wie lassen sich mit \mathcal{A} einzelne Schlüsselbits bestimmen?²¹¹

- Dann wird der Kandidat $L = K^5$ den Test $u^4(y, L)_b = x_a$ für alle $(x, y) \in M$ bestehen (falls $c = 0$ ist) oder für alle $(x, y) \in M$ nicht bestehen (falls $c = 1$ ist)
- Zudem hängt $u^4(y, L)_b$ wegen $b = 0x0505$ nur vom 2. und 4. Teilblock von $u^4(y, L)$ ab, welche wiederum nur von den Teilblöcken $I = L_{(2)}$ bzw. $J = L_{(4)}$ von L abhängen, d.h. $u^4(y, L)_b = u^4(y, I, J)_b$
- Daher genügt es, den Test für alle Kandidaten (I, J) für $(K_{(2)}^5, K_{(4)}^5)$ und alle $(x, y) \in M$ auszuführen
- Der **richtige Kandidat** $(I, J) = (K_{(2)}^5, K_{(4)}^5)$ besteht dann den Test $x_a = u^4(y, I, J)_b$ entweder für jedes oder für kein Paar (x, y) in M
- Dagegen besteht ein **falscher Kandidat** $(I, J) \neq (K_{(2)}^5, K_{(4)}^5)$ diesen Test für etwa die Hälfte aller Paare (x, y) in M
- Daher können wir den richtigen Kandidaten daran erkennen, dass er als einziger den Test für jedes (bzw. kein) Paar (x, y) in M besteht

Wie lassen sich mit \mathcal{A} einzelne Schlüsselbits bestimmen?²¹²

- Nun zum Fall, dass die Güte der linearen Approximation $\mathcal{A} = \mathcal{X}_a \oplus \mathcal{U}_b^4$ an die Abbildung $x \mapsto u^4$ zwar nicht gleich 1 ist, aber auch nicht sehr klein ist
- Ist β der Bias von \mathcal{A} , so besteht der gesuchte Subkey $(K_{(2)}^5, K_{(4)}^5)$ ungefähr einen Anteil von $(1/2 + \beta)$ der durchgeführten Tests, sofern die benutzten Klartext-Kryptotext-Paare hinreichend zufällig sind
- Dagegen besteht ein falscher Kandidat $(I, J) \neq (K_{(2)}^5, K_{(4)}^5)$ nur etwa die Hälfte aller Tests
- Falls also M eine hinreichend repräsentative Menge von genügend vielen Klartext-Kryptotext-Paaren (x, y) ist, können wir den richtigen Subkey daran erkennen, dass die Anzahl der von ihm bestandenen Tests am weitesten von $|M|/2$ abweicht

Algorithmus LinearAttack

- Der Algorithmus LinearAttack (siehe nächste Folie) ermittelt für jeden Subkey-Kandidaten (I, J) die Anzahl $\alpha(I, J)$ der Paare $(x, y) \in M$, für die (I, J) den Gleichheitstest $x_a = u_b^4(y, I, J)$ besteht
- Ausgegeben wird derjenige Kandidat (I, J) , der den Betrag $\hat{\alpha}(I, J)$ der Abweichung der Zahl $\alpha(I, J)$ von $|M|/2$ maximiert
- Im Allgemeinen werden für eine erfolgreiche lineare Attacke circa $t \approx c\beta^{-2}$ Klartext-Kryptotext-Paare benötigt
- Dabei ist c eine „kleine“ Konstante (im aktuellen Beispiel reichen $t \approx 8000$ Paare; d.h. $c \approx 8$, da $\beta^{-2} \approx 1024$ ist)

Algorithmus LinearAttack

```
1  $max := -1$ 
2 for  $(I, J) := (0,0)$  to  $(F,F)$  do
3    $\alpha(I, J) := 0$ 
4   for each  $(x, y) \in M$  do
5      $v_{(2)}^4 := I \oplus y_{(2)}$ 
6      $v_{(4)}^4 := J \oplus y_{(4)}$ 
7      $u_{(2)}^4 := S^{-1}(v_{(2)}^4)$ 
8      $u_{(4)}^4 := S^{-1}(v_{(4)}^4)$ 
9     if  $x_5 \oplus x_7 \oplus x_8 \oplus u_6^4 \oplus u_8^4 \oplus u_{14}^4 \oplus u_{16}^4 = 0$  then
10       $\alpha(I, J) := \alpha(I, J) + 1$ 
11       $\hat{\alpha}(I, J) := |\alpha(I, J) - t/2|$ 
12      if  $\hat{\alpha}(I, J) > max$  then  $(max, maxkey) := (\hat{\alpha}(I, J), (I, J))$ 
13 output $(maxkey)$ 
```

Differentielle Kryptoanalyse von SPNs

- Wie die lineare hat auch die differentielle Kryptoanalyse das Ziel, den unbekanntem Schlüssel K zu finden
- Für die Durchführung wird jedoch **frei wählbarer** Klartext benötigt
- Genauer basiert der Angriff auf einer Menge M von t **Doppelpaaren** (x, x^*, y, y^*) mit der Eigenschaft, dass $E_K(x) = y$ und $E_K(x^*) = y^*$ ist und alle Klartext-Paare (x, x^*) die gleiche Differenz $x' = x \oplus x^*$ bilden

Definition

- Seien $u, u^* \in \{0, 1\}^l$ Eingaben für eine S-Box $S : \{0, 1\}^l \rightarrow \{0, 1\}^l$ und seien $v = S(u)$ und $v^* = S(u^*)$ die zugehörigen Ausgaben
- Dann heißt $u' = u \oplus u^*$ die **Eingabedifferenz (input-xor)** und $v' = S(u) \oplus S(u^*)$ die **Ausgabedifferenz (output-xor)** von (u, u^*)
- Für eine vorgegebene Eingabedifferenz $a' \in \{0, 1\}^l$ sei weiter

$$\Delta(a') = \{(u, u^*) \mid u \oplus u^* = a'\} = \{(u, u \oplus a') \mid u \in \{0, 1\}^l\}$$

die Menge aller Eingabepaare, die die Differenz a' realisieren

- Berechnen wir für alle Paare $(u, u^*) \in \Delta(a')$ die zugehörigen Ausgabedifferenzen, so verteilen sich diese auf die $2^{l'}$ möglichen Werte in $\{0, 1\}^{l'}$
- Man beachte, dass im Fall einer **affinen** S-Box $S(u) = uA \oplus w$ alle Paare $(u, u^*) \in \Delta(a')$ auf dieselbe Ausgabedifferenz $b' = a'A$ führen:

$$S(u) \oplus S(u^*) = (uA \oplus w) \oplus (u^*A \oplus w) = (u \oplus u^*)A = u'A = a'A$$

(hierbei ist A eine Matrix in $\{0, 1\}^{l \times l'}$ und w ein Vektor in $\{0, 1\}^{l'}$)

- Ist S nicht affin, können die 2^l Eingabepaare $(u, u^*) \in \Delta(a')$ zu unterschiedlichen Ausgabedifferenzen $v' = S(u) \oplus S(u^*)$ führen, da in diesem Fall die Ausgabedifferenz nicht nur von der Eingabedifferenz u' , sondern vom Eingabepaar (u, u^*) abhängt
- Um einer differentiellen Kryptoanalyse widerstehen zu können, sollten die auftretenden Ausgabedifferenzen möglichst gleichmäßig verteilt sein

Definition

- Ein **Differential** für eine S-Box $S : \{0, 1\}^l \rightarrow \{0, 1\}^l$ ist ein Paar $(a', b') \in \{0, 1\}^l \times \{0, 1\}^l$
- Dabei heißt a' die **Eingabe-** und b' die **Ausgabedifferenz** des Differentials
- Die Anzahl der Eingabepaare (u, u^*) , die die Eingabedifferenz a' in die Ausgabedifferenz b' überführen, bezeichnen wir mit $D(a', b')$, d.h.

$$D(a', b') = |\{(u, u^*) \in \Delta(a') \mid S(u) \oplus S(u^*) = b'\}|$$

- Der **Weitergabequotient** (engl. **propagation ratio**) von S für ein Differential (a', b') ist der Anteil

$$\rho_S(a', b') = \frac{D(a', b')}{|\Delta(a')|} = 2^{-l} D(a', b')$$

der Eingabepaare (u, u^*) in $\Delta(a')$, die auf die Ausgabedifferenz b' führen

- Der Weitergabequotient $\rho_S(a', b')$ ist die bedingte Wahrscheinlichkeit

$$\begin{aligned}\Pr[\mathcal{V}' = b' | \mathcal{U}' = a'] &= \underbrace{\Pr[\mathcal{V}' = b' \wedge \mathcal{U}' = a']}_{D(a', b')/2^{2l}} / \underbrace{\Pr[\mathcal{U}' = a']}_{|\Delta(a')|/2^{2l}} \\ &= D(a', b')/|\Delta(a')|\end{aligned}$$

dass zwei zufällig mit der Eingabedifferenz $\mathcal{U}' = \mathcal{U} \oplus \mathcal{U}^* = a'$ gewählte Eingaben \mathcal{U} und \mathcal{U}^* die Ausgabedifferenz $\mathcal{V}' = S(\mathcal{U}) \oplus S(\mathcal{U}^*) = b'$ bilden

Differenzielle Kryptoanalyse von SPNs

Beispiel

- Betrachten wir die S-Box $S : \{0, 1\}^4 \rightarrow \{0, 1\}^4$ aus obigem Beispiel, so erhalten wir für $a' = 1011$ folgende Menge von Eingabepaaren

$$\Delta(a') = \{(0000, 1011), \dots, (1111, 0100)\},$$

- Diese führen auf die folgenden Ausgabedifferenzen $v' = \underbrace{S(u)}_v \oplus \underbrace{S(u^*)}_{v^*}$:

u	u^*	v	v^*	v'	u	u^*	v	v^*	v'
0000	1011	1110	1100	0010	1000	0011	0011	0001	0010
0001	1010	0100	0110	0010	1001	0010	1010	1101	0111
0010	1001	1101	1010	0111	1010	0001	0110	0100	0010
0011	1000	0001	0011	0010	1011	0000	1100	1110	0010
0100	1111	0010	0111	0101	1100	0111	0101	1000	1101
0101	1110	1111	0000	1111	1101	0110	1001	1011	0010
0110	1101	1011	1001	0010	1110	0101	0000	1111	1111
0111	1100	1000	0101	1101	1111	0100	0111	0010	0101

Beispiel (Fortsetzung)

- Die Ausgabedifferenz $b' = 0010$ kommt also $D(a', 0010) = 8$ Mal vor, während die Differenzen **0101**, **0111**, **1101** und **1111** je zwei Mal und die übrigen Werte überhaupt nicht vorkommen (siehe Zeile B in nachfolgender Tabelle)
- Führen wir diese Berechnungen für jede der $2^4 = 16$ Eingabedifferenzen $a' \in \{0, 1\}^4$ aus, so erhalten wir die folgenden Werte für die Häufigkeiten $D(a', b')$ der Ausgabedifferenz b' bei Eingabedifferenz a' (a' und b' sind hexadezimal dargestellt)

Beispiel (Schluss)

- Die Tabelle zeigt die Häufigkeiten $D(a', b')$ der Ausgabedifferenzen b' der S-Box S für eine Auswahl von Eingabedifferenzen $a' \in \{0, 1\}^4$

a'	b'															
	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	16	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
1	0	0	0	2	0	0	0	2	0	2	4	0	4	2	0	0
2	0	0	0	2	0	6	2	2	0	2	0	0	0	0	2	0
3	0	0	2	0	2	0	0	0	0	4	2	0	2	0	0	4
⋮								⋮								
B	0	0	8	0	0	2	0	2	0	0	0	0	0	2	0	2
⋮								⋮								
F	0	2	0	0	6	0	0	0	0	4	0	2	0	0	2	0

Differentielle Kryptoanalyse von SPNs

- Wir versuchen nun, für ausgewählte S-Boxen in Runde 1 bis $N - 1$ (diese werden wieder **aktiv** genannt) Differentiale (a', b') zu finden, so dass die (permutierte) Ausgabedifferenz dieser Differentiale in Runde i mit der Eingabedifferenz in Runde $i + 1$ übereinstimmt (siehe nächste Folie)
- Die ausgewählten Differentiale der aktiven S-Boxen lassen sich zu einer sog. **Differentialspur** (**differential trail**) zusammensetzen
- Falls die einzelnen aktiven S-Boxen S_i die vorgegebenen Differenzen unabhängig gemäß den ausgewählten Differentialen (a'_i, b'_i) weitergeben, folgt ein zufällig gewähltes Klartextpaar $(x, x^*) \in \Delta(a')$ dieser Spur mit der Wahrscheinlichkeit $\prod_i \rho(a'_i, b'_i)$
- Auch wenn die Unabhängigkeit meist nicht gegeben ist, weicht der tatsächliche Wert dieser Wahrscheinlichkeit in konkreten Anwendungen nur wenig von diesem hypothetischen Wert ab

Differentielle Kryptoanalyse von SPNs

Beispiel

- Betrachten wir das SPN SP aus dem vorigen Beispiel, so lassen sich folgende Differentiale zu einer Spur kombinieren:
 - für S_2^1 das Differential $(1011, 0010) = (B, 2)$ mit $\rho(B, 2) = 1/2$
 - für S_3^2 das Differential $(0100, 0110) = (4, 6)$ mit $\rho(4, 6) = 3/8$ und
 - für S_2^3 und S_3^3 das Differential $(0010, 0101) = (2, 5)$ mit $\rho(2, 5) = 3/8$
- Gemäß dieser Spur führt also die Klartextdifferenz

$$x' = 0000\ 1011\ 0000\ 0000$$

mit hypothetischer Wahrscheinlichkeit $1/2(3/8)^3$ auf die Differenz

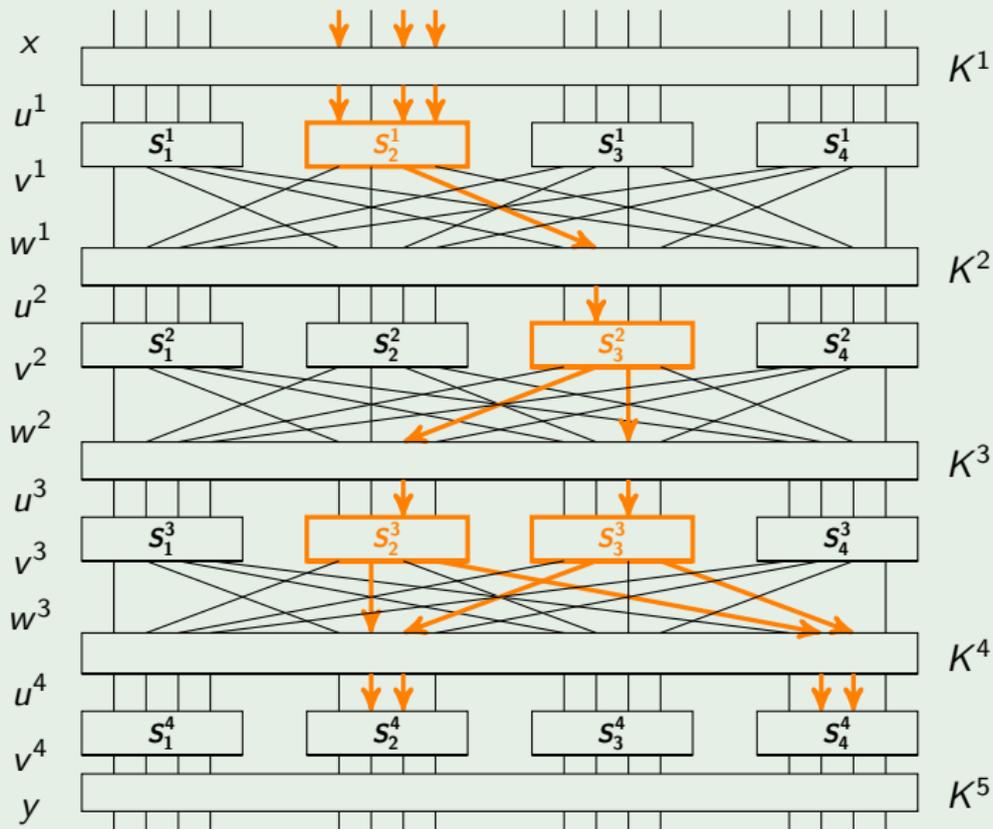
$$(v^3)' = 0000\ 0101\ 0101\ 0000,$$

und diese führt mit Wahrscheinlichkeit 1 auf die Differenz

$$(u^4)' = 0000\ 0110\ 0000\ 0110$$

- Das Differential $(a', b') = (0000\ 1011\ 0000\ 0000, 0000\ 0110\ 0000\ 0110)$ für die Abbildung $x \mapsto u^4$ hat also den hypothetischen Weitergabequotienten $\rho(a', b') = 1/2(3/8)^3 = 27/1024 \approx 0,026$

Beispiel (Fortsetzung)



Differentielle Kryptoanalyse von SPNs

- Sei nun (a', b') ein Differential für die Abbildung $x \mapsto u^N$ mit einem hypothetischen Weitergabequotienten $\rho = \rho(a', b')$
- Weiter sei M eine Menge von t Doppelpaaren (x, x^*, y, y^*) , die
 - alle mit dem gleichen unbekanntem Schlüssel K erzeugt wurden und
 - zusätzlich die Bedingung $x' = x \oplus x^* = a'$ erfüllen
- Dann werden für ca. ein ρ -Anteil dieser Doppelpaare die Klartexte x und x^* in der letzten Runde auf Blöcke u^N und $(u^N)^*$ mit

$$(u^N)' = u^N \oplus (u^N)^* = b'$$

abgebildet (solche Doppelpaare werden als **richtig** bezeichnet)

- Ein Teil der **falschen Doppelpaare** lässt sich daran erkennen, dass die Kryptotext-Differenzen y' nicht die erwarteten 0'-Blöcke aufweisen (im aktuellen Beispiel sind dies die Blöcke $y'_{(1)}$ und $y'_{(3)}$)
- Es empfiehlt sich, diese offensichtlich falschen Doppelpaare auszufiltern, da sie (wie alle falschen Doppelpaare) nur „Hintergrundrauschen“ erzeugen und somit die Bestimmung des Schlüssels behindern

Differentielle Kryptoanalyse von SPNs

Beobachtung

- Sei S_i^N eine aktive S-Box in Runde N und sei $(x, x^*, y, y^*) \in M$
- Dann können wir für jeden Kandidaten J für den Subkey $K_{(i)}^{N+1}$ von K^{N+1} die Ausgabe

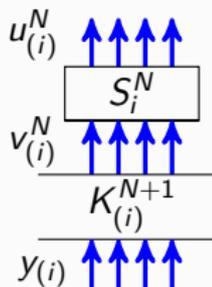
- $v_{(i)}^N(y, J) = y_{(i)} \oplus J$ von S_i^N und die Eingabe
- $u_{(i)}^N(y, J) = S^{-1}(v_{(i)}^N) = S^{-1}(y_{(i)} \oplus J)$ von S_i^N

zurückrechnen

- Falls die S-Box S_i^N nicht affin ist, hängt die aus den Kryptotexten y und y^* zurückgerechnete Eingabedifferenz

$$(u_{(i)}^N)'(y, y^*, J) = S^{-1}(y_{(i)} \oplus J) \oplus S^{-1}(y_{(i)}^* \oplus J)$$

von dem Subkey-Kandidaten J ab



Beobachtung (Fortsetzung)

- Da jedes richtige Doppelpaar (x, x^*, y, y^*) auf die Differenz $(u^N)' = b'$ führt, erfüllt der richtige Kandidat $J = K_{(i)}^{N+1}$ den Test

$$S^{-1}(y_{(i)} \oplus J) \oplus S^{-1}(y_{(i)}^* \oplus J) = b'_{(i)} \quad (*)$$

für alle richtigen Doppelpaare (x, x^*, y, y^*)

- Erfüllt ein Subkey-Kandidat J die Gleichung $(*)$ für ein Doppelpaar (x, x^*, y, y^*) , so sagen wir auch, J ist mit (x, x^*, y, y^*) konsistent
- Gemäß obiger Beobachtung ist der gesuchte Subkey $J = K_{(i)}^{N+1}$ mit allen richtigen Doppelpaaren konsistent
- Dagegen ist ein Subkey $J \neq K_{(i)}^{N+1}$ nur mit einem Teil der richtigen Doppelpaare konsistent
- Ist M hinreichend groß, so lässt sich der gesuchte Subkey daran erkennen, dass er mit mehr Doppelpaaren als jeder andere Kandidat konsistent ist

Wir führen nun mit der Spur aus obigem Beispiel einen Angriff mittels differentieller Analyse auf das SPN SP durch

Beispiel

- Der Algorithmus DifferentialAttack (siehe nächste Folie) erhält eine Menge M von Doppelpaaren (x, x^*, y, y^*) mit $x \oplus x^* = a'$
- Zuerst blendet er alle offensichtlich falschen Doppelpaare aus (Zeile 1)
- Anschließend bestimmt er für jeden Kandidaten (I, J) für den Subkey $(K_{(2)}^5, K_{(4)}^5)$ die Anzahl $\kappa(I, J)$ aller mit (I, J) konsistenten Doppelpaare
- Ausgegeben wird der Kandidat (I, J) mit dem größten κ -Wert
- Im Allgemeinen werden für eine erfolgreiche differentielle Attacke circa $t \approx c\rho^{-1}$ Klartext-Kryptotext-Doppelpaare benötigt
- Dabei ist ρ der Weitergabequotient der Spur und c eine Konstante (im Beispiel reichen $t \approx 80$ Doppelpaare, wobei $\rho^{-1} \approx 38$ ist, d.h. $c \approx 2$) ◀

Algorithmus DifferentialAttack

```

1   $M := \{(x, x^*, y, y^*) \in M \mid y_{(1)} = y_{(1)}^* \wedge y_{(3)} = y_{(3)}^*\}$ 
2   $max := -1$ 
3  for  $(I, J) := (0,0)$  to  $(F,F)$  do
4     $\kappa(I, J) := 0$ 
5    for each  $(x, x^*, y, y^*) \in M$  do
6       $v_{(2)}^4 := I \oplus y_{(2)}$ ;    $v_{(4)}^4 := J \oplus y_{(4)}$ 
7       $u_{(2)}^4 := S^{-1}(v_{(2)}^4)$ ;    $u_{(4)}^4 := S^{-1}(v_{(4)}^4)$ 
8       $(v_{(2)}^4)^* := I \oplus y_{(2)}^*$ ;    $(v_{(4)}^4)^* := J \oplus y_{(4)}^*$ 
9       $(u_{(2)}^4)^* := S^{-1}((v_{(2)}^4)^*)$ ;    $(u_{(4)}^4)^* := S^{-1}((v_{(4)}^4)^*)$ 
10      $(u_{(2)}^4)' := u_{(2)}^4 \oplus (u_{(2)}^4)^*$ ;    $(u_{(4)}^4)' := u_{(4)}^4 \oplus (u_{(4)}^4)^*$ 
11     if  $(u_{(2)}^4)' = 0110 \wedge (u_{(4)}^4)' = 0110$  then  $\kappa(I, J) := \kappa(I, J) + 1$ 
12     if  $\kappa(I, J) > max$  then  $(max, maxkey) := (\kappa(I, J), (I, J))$ 
13 output(maxkey)

```

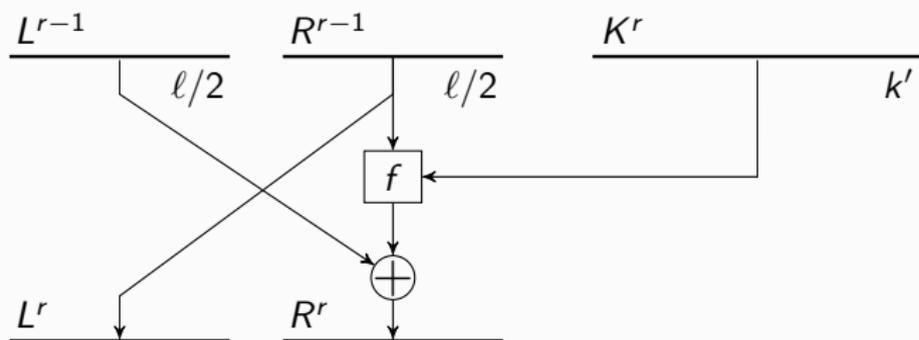
Der Data Encryption Standard (DES)

- Der DES wurde von IBM im Zuge einer Ausschreibung des NBS (National Bureau of Standards; heute National Institute of Standards and Technology, NIST) als ein Nachfolger von Lucifer entwickelt
- Er wurde 1975 veröffentlicht, und 1977 als Verschlüsselungsstandard der US-Regierung für nicht geheime Nachrichten genormt
- Obwohl DES ursprünglich nur für einen Zeitraum von 10 bis 15 Jahren als Standard dienen sollte, wurde er circa alle 5 Jahre (zuletzt 1999) überprüft und als Standard fortgeschrieben
- Bereits 1997 veröffentlichte das NIST eine Ausschreibung für den AES (Advanced Encryption Standard) genannten Nachfolger des DES
- Nach einer mehrjährigen Auswahlprozedur wurde im November 2001 der Rijndael-Algorithmus als AES genormt und im Mai 2002 wurde DES von AES als Standard abgelöst
- Allerdings wurde Triple DES (auch TDES oder 3DES genannt) vom NIST als Standard bis 2030 fortgeschrieben

- Die Rundenfunktion g einer **Feistel-Chiffre** berechnet das Zwischenergebnis $w^r = g(K^r, w^{r-1}) \in \{0, 1\}^\ell$ in Runde r gemäß der Vorschrift

$$L^r = R^{r-1} \quad \text{und} \quad R^r = L^{r-1} \oplus f(K^r, R^{r-1})$$

aus den beiden Hälften L^{r-1} und R^{r-1} von $w^{r-1} = L^{r-1}R^{r-1} \in \{0, 1\}^\ell$:



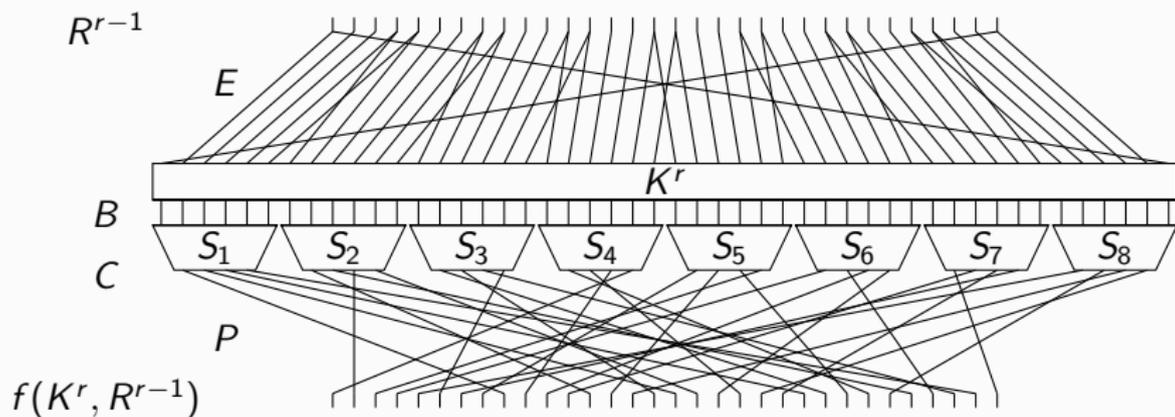
- Hierbei ist $f : \{0, 1\}^{k'} \times \{0, 1\}^{\ell/2} \rightarrow \{0, 1\}^{\ell/2}$ eine beliebige Funktion und k' ist die Länge der Rundenschlüssel K^1, \dots, K^N
- Aus dem letzten Zwischenergebnis $w^N = L^N R^N$ in Runde N wird dann durch Vertauschung von L^N und R^N der Kryptotext $y = R^N L^N$ gebildet

Aufbau der DES-Chiffrierfunktion

- Der DES ist eine Feistel-Chiffre mit einer Blocklänge von $\ell = 64$ Bit und $N = 16$ Runden
- Die 64 Bit Schlüssel haben eine effektive Länge von 56 Bit, da sie acht Paritätsbits (Bit 8, 16, ..., 64) enthalten
- Es gibt somit nur $2^{56} \approx 7.2 \cdot 10^{16}$ verschiedene Schlüssel
- Bei Eingabe eines Schlüssels K und eines Klartextes x führt der DES-Algorithmus die folgenden Chiffrierschritte aus:
 - Zuerst wird der Klartext x einer Initialpermutation $IP: x_1 x_2 \cdots x_{64} \mapsto x_{58} x_{50} \cdots x_7$ unterzogen (siehe nächste Folie)
 - Danach erfolgen 16 Runden mit einer Feistel-Rundenfunktion g unter Verwendung von sechzehn Rundenschlüsseln K^1, \dots, K^{16} (die Berechnung der Rundenschlüssel wird später beschrieben)
 - Aus dem am Ende von Runde 16 ausgegebenen Block $w^{16} = L^{16} R^{16}$ wird durch Vertauschen von L^{16} und R^{16} und Anwendung von IP^{-1} der Kryptotextblock $y = \text{DES}(K, x) = IP^{-1}(R^{16} L^{16})$ gebildet

Graphische Darstellung der DES-Funktion f

- Die DES-Funktion f :



- Initialpermutation IP , Expansion E und Permutation P :

IP							
58	50	42	34	26	18	10	2
60	52	44	36	28	20	12	4
62	54	46	38	30	22	14	6
64	56	48	40	32	24	16	8
57	49	41	33	25	17	9	1
59	51	43	35	27	19	11	3
61	53	45	37	29	21	13	5
63	55	47	39	31	23	15	7

E					
32	1	2	3	4	5
4	5	6	7	8	9
8	9	10	11	12	13
12	13	14	15	16	17
16	17	18	19	20	21
20	21	22	23	24	25
24	25	26	27	28	29
28	29	30	31	32	1

P			
16	7	20	21
29	12	28	17
1	15	23	26
5	18	31	10
2	8	24	14
32	27	3	9
19	13	30	6
22	11	4	25

Berechnung der DES-Funktion f

- Die Funktion $f : \{0, 1\}^{48} \times \{0, 1\}^{32} \rightarrow \{0, 1\}^{32}$ wird wie folgt berechnet:
- Bei Eingabe (K^r, R^{r-1}) wird zuerst der 32-Bit Block R^{r-1} mittels der Expansionsabbildung E (s.u.) auf einen 48-Bit Block $E(R^{r-1})$ erweitert
- Auf diesen wird bitweise der Rundenschlüssel K^r addiert
- Als Ergebnis erhalten wir den 48-Bit Block $B = E(R^{r-1}) \oplus K^r$
- Dieser wird in acht 6-Bit Blöcke $B_{(1)}, \dots, B_{(8)}$ aufgeteilt, die mit den 8 S-Boxen S_1, \dots, S_8 auf 4-Bit Blöcke $C_{(i)} = S_i(B_{(i)})$ verkleinert werden
- Die Konkatenation der von den acht S-Boxen berechneten 4-Bit Blöcke ergibt einen 32-Bit Block $C = C_{(1)} \dots C_{(8)}$
- Zum Schluss wird auf C noch die Transposition P angewandt (siehe vorige Folie; der Werteverlauf der Transpositionen IP , E und P ist dort jeweils zeilenweise dargestellt)

Tabellarische Darstellung der acht DES S-Boxen

- Die Werte $S_i(B_{(j)})$ lassen sich aus folgenden Tabellen wie folgt ablesen:

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
S_1 :	E	4	D	1	2	F	B	8	3	A	6	C	5	9	0	7
	0	F	7	4	E	2	D	1	A	6	C	B	9	5	3	8
	4	1	E	8	D	6	2	B	F	C	9	7	3	A	5	0
	F	C	8	2	4	9	1	7	5	B	3	E	A	0	6	D

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
S_2 :	F	1	8	E	6	B	3	4	9	7	2	D	C	0	5	A
	3	D	4	7	F	2	8	E	C	0	1	A	6	9	B	5
	0	E	7	B	A	4	D	1	5	8	C	6	9	3	2	F
	D	8	A	1	3	F	4	2	B	6	7	C	0	5	E	9

S_3 :	A	0	9	E	6	3	F	5	1	D	C	7	B	4	2	8
	D	7	0	9	3	4	6	A	2	8	5	E	C	B	F	1
	D	6	4	9	8	F	3	0	B	1	2	C	5	A	E	7
	1	A	D	0	6	9	8	7	4	F	E	3	B	5	2	C

S_4 :	7	D	E	3	0	6	9	A	1	2	8	5	B	C	4	F
	D	8	B	5	6	F	0	3	4	7	2	C	1	A	E	9
	A	6	9	0	C	B	7	D	F	1	3	E	5	2	8	4
	3	F	0	6	A	1	D	8	9	4	5	B	C	7	2	E

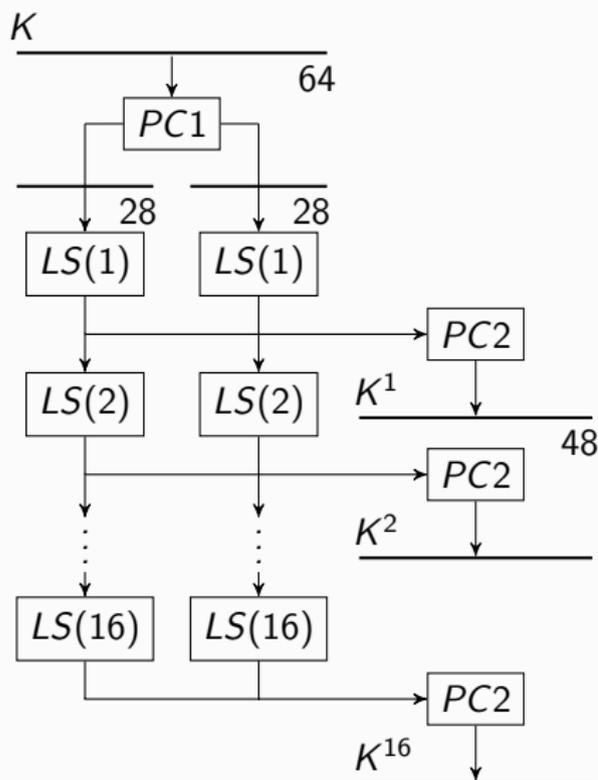
S_5 :	2	C	4	1	7	A	B	6	8	5	3	F	D	0	E	9
	E	B	2	C	4	7	D	1	5	0	F	A	3	9	8	6
	4	2	1	B	A	D	7	8	F	9	C	5	6	3	0	E
	B	8	C	7	1	E	2	D	6	F	0	9	A	4	5	3

S_6 :	C	1	A	F	9	2	6	8	0	D	3	4	E	7	5	B
	A	F	4	2	7	C	9	5	6	1	D	E	0	B	3	8
	9	E	F	5	2	8	C	3	7	0	4	A	1	D	B	6
	4	3	2	C	9	5	F	A	B	E	1	7	6	0	8	D

S_7 :	4	B	2	E	F	0	8	D	3	C	9	7	5	A	6	1
	D	0	B	7	4	9	1	A	E	3	5	C	2	F	8	6
	1	4	B	D	C	3	7	E	A	F	6	8	0	5	9	2
	6	B	D	8	1	4	A	7	9	5	0	F	E	2	3	C

S_8 :	D	2	8	4	6	F	B	1	A	9	3	E	5	0	C	7
	1	F	D	8	A	3	7	4	C	5	6	B	0	E	9	2
	7	B	4	1	9	C	E	2	0	6	A	D	F	3	5	8
	2	1	E	7	4	A	8	D	F	C	9	0	3	5	6	B

- Ist $B_{(j)} = b_1 \cdots b_6$, so findet man $S_i(B_{(j)})$ in der Tabelle für S_i in Zeile $b_1 b_6$ und Spalte $b_2 b_3 b_4 b_5$
- Zum Beispiel ist $S_1(011010) = 1001$, da in Zeile $(00)_2 = 0$ und Spalte $(1101)_2 = D$ die Hexadezimalziffer $9 = (1001)_2$ steht



$PC1$								$PC2$							
57	49	41	33	25	17	9		14	17	11	24	1	5		
1	58	50	42	34	26	18		3	28	15	6	21	10		
10	2	59	51	43	35	27		23	19	12	4	26	8		
19	11	3	60	52	44	36		16	7	27	20	13	2		
63	55	47	39	31	23	15		41	52	31	37	47	55		
7	62	54	46	38	30	22		30	40	51	45	33	48		
14	6	61	53	45	37	29		44	49	39	56	34	53		
21	13	5	28	20	12	4		46	42	50	36	29	32		

r	1	2	3	4	5	6	7	8
$LS(r)$	1	1	2	2	2	2	2	2
r	9	10	11	12	13	14	15	16
$LS(r)$	1	2	2	2	2	2	2	1

- Die 16 Rundenschlüssel K^1, \dots, K^{16} werden wie folgt aus dem externen 64-Bit Schlüssel K berechnet (siehe vorige Folie):
- Zuerst wählt die Funktion $PC\ 1$ (permuted choice 1) aus dem Schlüssel K die 56 kryptografisch relevanten Bits aus und permutiert sie
- Das Resultat wird in zwei 28-Bit Blöcke unterteilt, die in 16 Runden $r = 1, \dots, 16$ jeweils zyklisch um $LS(r) \in \{1, 2\}$ Bit verschoben werden
- Aus den beiden Blöcken nach Runde r bestimmt die Funktion $PC\ 2$ (permuted choice 2) jeweils den Rundenschlüssel K^r und entfernt dabei die 8 Bits an den Positionen 9, 18, 22, 25, 35, 38, 43 und 56

Der Key-Schedule Algorithmus des DES

Definition

Ein DES-Schlüssel K heißt **schwach**, falls alle durch ihn erzeugten Rundenschlüssel gleich sind (d.h. es gilt $\{K^1, \dots, K^{16}\} = \{K^1\}$)

- Der DES hat folgende vier schwache Schlüssel (hexadezimal):

K	erzeugter Rundenschlüssel
0101010101010101	000000000000
1F1F1F1F0E0E0E0E	000000111111
E0E0E0E0F1F1F1F1	111111000000
FEFEFEFEFEFEFEFE	111111111111

- Für jeden von ihnen gilt $\text{DES}(K, \text{DES}(K, x)) = x$ (siehe Übungen)
- Zudem existieren noch sechs sogenannte **semischwache** Schlüsselpaare (K, K') mit der Eigenschaft $\text{DES}(K', \text{DES}(K, x)) = x$ (siehe Übungen)

Sicherheitsbedenken gegen den DES

- Der DES konnte sich nicht sofort nach seiner Veröffentlichung im Jahr 1975 durchsetzen
- Er wurde von manchen Behörden und Banken in den USA zunächst nicht verwendet, da folgende Sicherheitsbedenken gegen ihn bestanden:
 - Die 56-Bit Schlüssellänge bietet eine zu geringe Sicherheit gegen einen Brute-Force Angriff bei bekanntem oder wählbarem Klartext
 - Die Entwurfskriterien für die einzelnen Komponenten, insbesondere die S-Boxen, sind nicht veröffentlicht worden; daher wurde der Verdacht geäußert, dass der DES **Falltüren** besitzt, um einen Angriff durch die National Security Agency (NSA) zu ermöglichen
 - Kryptoanalytische Untersuchungen, die von IBM und der NSA durchgeführt wurden, blieben unter Verschluss
 - Als jedoch Biham und Shamir Anfang der 90er Jahre das Konzept der differentiellen Kryptoanalyse veröffentlichten, gaben die Entwickler bekannt, dass sie diesen Angriff beim Entwurf von DES bereits kannten und sie die S-Boxen dahingehend optimiert haben

- Im Fall von DES ist die lineare Kryptoanalyse effizienter als die differentielle Kryptoanalyse
- Da hierzu jedoch circa 2^{43} Klartext-Kryptotext-Doppelpaare notwendig sind, stellen diese Angriffe keine realistische Bedrohung dar (allein die Generierung der Doppelpaare dauerte bei einem von Matsui, dem Erfinder der linearen Kryptoanalyse, durchgeführten Angriff 40 Tage)
- Dagegen gelang bereits im Juli 1998 mit einer von der Electronic Frontier Foundation (EFF) für 250 000 Dollar gebauten Maschine namens "DES Cracker" eine vollständige Schlüsselsuche in 56 Stunden (dies bedeutete den Gewinn der von RSA Laboratory ausgeschriebenen "DES Challenge II-2")

- Im Jahr 1999 gewann Distributed.Net, eine weltweite Vereinigung von Computerfans, den mit 10 000 Dollar dotierten “DES Challenge III”
- Ihnen gelang es, durch den kombinierten Einsatz des Supercomputers “Deep Crack” und 100 000 PCs, die per Internet kommunizierten, nach 22:15h den Schlüssel zu dem bekannten Klartext „See you in Rome (second AES Conference, March 22-23, 1999)“ zu finden
- Es gibt sogar kommerzielle Angebote im Internet (z.B. `crack.sh`), die bei bekanntem Klartext innerhalb von 26 Stunden eine vollständige Schlüsselsuche auf spezieller Hardware ausführen und alle passenden DES-Schlüssel finden

- Als Vorbereitung zum AES-Algorithmus gehen wir kurz auf die Arithmetik in endlichen Körpern ein
- Diese spielt beim AES eine wichtige Rolle
- Wie wir bereits wissen, bildet \mathbb{Z}_p für primes p einen endlichen Körper der Größe p
- Dieser Körper lässt sich für jede Zahl $n \geq 1$ auf die Größe p^n erweitern
- Da bis auf Isomorphie nur ein Körper der Größe p^n existiert, wird er einfach mit $\mathbb{F}(p^n)$ oder \mathbb{F}_{p^n} bezeichnet
- Um den Körper \mathbb{F}_{p^n} zu konstruieren, betrachten wir zunächst den **Polynomring** $\mathbb{Z}_p[x]$ über \mathbb{Z}_p

Definition. Sei R ein Ring.

- Der **Polynomring** $R[x]$ enthält für alle $n \geq 0$ alle Polynome $p(x)$ in der Variablen x mit Koeffizienten in R , d.h. $p(x)$ hat die Form

$$p(x) = a_n x^n + \cdots + a_1 x + a_0 \quad \text{mit } a_0, \dots, a_n \in R$$

Man sagt, $R[x]$ entsteht aus R durch **Adjunktion der Variablen** x

- Der **Grad von** p (bezeichnet mit $\deg(p)$) ist im Fall $a_n \neq 0$ gleich n und im Fall $n = a_n = 0$ gleich -1

Man überprüft leicht, dass $R[x]$ mit der üblichen Polynomaddition und Polynommultiplikation tatsächlich einen Ring bildet

Definition (Fortsetzung).

- Ein Polynom $q(x)$ teilt ein Polynom $p(x)$ (kurz: $q(x)|p(x)$), falls ein Polynom $d(x) \in R[x]$ existiert mit $p(x) = d(x)q(x)$
- Teilt $q(x)$ die Differenz $f(x) - g(x)$ zweier Polynome, so schreiben wir

$$f(x) \equiv_{q(x)} g(x)$$

und sagen, $f(x)$ ist kongruent zu $g(x)$ modulo $q(x)$

- Weiterhin bezeichne

$$p(x) \bmod q(x)$$

das bei der Polynomdivision von $p(x)$ durch $q(x)$ auftretende **Restpolynom**, also dasjenige Polynom $r(x)$ vom Grad $\deg(r) < \deg(q)$, für das ein Polynom $d(x) \in R[x]$ existiert mit $p(x) = d(x)q(x) + r(x)$

Faktorringe von Polynomringen

- Ähnlich wie beim Übergang von \mathbb{Z} zum Restklassenring \mathbb{Z}_m können wir nun jedem Polynom $p(x) \in \mathbb{Z}_m[x]$ mittels

$$p(x) \mapsto p(x) \bmod m(x)$$

eindeutig ein Polynom vom Grad höchstens $n - 1$ zuordnen, wobei $m(x)$ ein fest gewähltes Polynom und n der Grad von $m(x)$ ist

- Auf diese Weise erhalten wir den Polynomring $\mathbb{Z}_m[x]/m(x)$ aller Polynome vom Grad höchstens $n - 1$
- Dieser endliche Ring heißt **Faktoring von $\mathbb{Z}_m[x]$ modulo $m(x)$**
- Die Addition und Multiplikation sind hierbei wie in $\mathbb{Z}_m[x]$, gefolgt von einer Reduktion modulo $m(x)$, definiert
- In den Übungen werden wir sehen, dass $\mathbb{Z}_m[x]/m(x)$ genau dann ein Körper ist, wenn m prim ist und $m(x)$ nur triviale Teiler besitzt

Irreduzible Polynome

Definition

Ein Polynom $m(x) \in \mathbb{Z}_p[x]$, p prim, vom Grad $n \geq 1$ heißt **irreduzibel** (über \mathbb{Z}_p), falls keine Polynome $p(x), q(x) \in \mathbb{Z}_p[x]$ vom Grad $\deg(p), \deg(q) \geq 1$ existieren mit

$$m(x) = p(x)q(x)$$

Satz

Der Faktorring $\mathbb{Z}_m[x]/m(x)$ ist genau dann ein Körper, wenn $m = p$ prim und $m(x)$ in $\mathbb{Z}_p[x]$ irreduzibel ist.

Beweis.

Siehe Übungen. □