6.1 Die Struktur von Rijndael

Repräsentation der Blöcke

Rijndael operiert auf Oktetten (8-Bit-Bytes), also auf dem $\mathbb{F}_2$-Vektorraum $\mathbb{F}_2^8$. Da an mehreren Stellen die Struktur dieses Vektorraums als Körper mit 256 Elementen ausgenützt wird, liegt es nahe, ihn von vorherein mit diesem Körper $\mathbb{F}_{256}$ zu identifizieren. Wie das genau gemacht wird, steht in Abschnitt 6.2.

Die Blöcke, mit denen Rijndael umgeht, sind zunächst Bit-Blöcke, deren Länge ein Vielfaches von 32 ist; spezifiziert ist der Algorithmus für die Block- und Schlüssellängen 128, 160, 192, 224, 256.

> Hier liegt der einzige Unterschied zwischen Rijndael und AES: AES ist nur für die Blocklänge 128 und die Schlüssellängen 128, 192 und 256 spezifiziert. Der Standardisierungsprozess macht daher keine Aussagen über die Sicherheit von Rijndael bei den übrigen Größen dieser Parameter.

Die 32 Bit werden jeweils als eine Spalte aus 4 Oktetten, also als ein Element des 4-dimensionalen $\mathbb{F}_{256}$-Vektorraums $\mathbb{F}_{256}^4$ aufgefasst. Ein Block der Länge $n = 32 \cdot N_b$ wird dann als N_b-Tupel solcher Spalten, also als $4 \times N_b$-Matrix

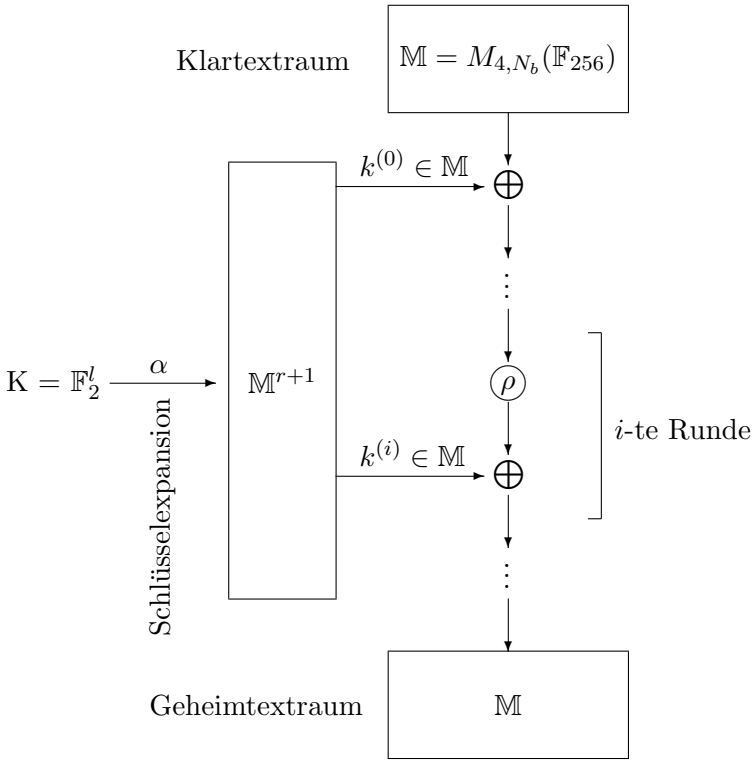
$$a = \begin{pmatrix} a_{0,0} & \dots & a_{0,N_b-1} \\ \vdots & & \vdots \\ a_{3,0} & \dots & a_{3,N_b-1} \end{pmatrix} \in M_{4,N_b}(\mathbb{F}_{256}) =: \mathbb{M}$$

interpretiert. Analog sei die Schlüssellänge $l = 32 \cdot N_k$; entsprechende Matrizen werden aber nicht verwendet, sondern Schlüssel werden erst expandiert und dann in Rundenschlüssel $k^{(i)} \in \mathbb{M}$ zerlegt.

> Nach der Spezifikation von Rijndael ist $N_b, N_k \in \{4, 5, 6, 7, 8\}$, nach der Spezifikation von AES $N_b = 4$, $N_k \in \{4, 6, 8\}$.

Das Iterationsschema

Der Verschlüsselungs-Algorithmus von Rijndael durchläuft r Runden, die jeweils aus der Kernabbildung ρ und der (binären) Addition eines Teilschlüssels (Rundenschlüssel) bestehen; in der letzten Runde ist die Kernabbildung etwas verkürzt – mehr dazu später. Vor Beginn der ersten Runde wird schon ein Teilschlüssel addiert; die Schlüsselexpansion muss also aus den gegebenen l Schlüsselbits $r + 1$ Teilschlüssel aus je $32N_b$ Bits produzieren. Die Kernabbildung ist invertierbar. Sie ist von Runde zu Runde (mit Ausnahme der letzten) gleich.



Die Zahl der Runden ist wie folgt festgelegt:

	N_b				
N_k	4	5	6	7	8
4	10	11	12	13	14
5	11	11	12	13	14
6	12	12	12	13	14
7	13	13	13	13	14
8	14	14	14	14	14

Bei AES mit Schlüssellänge 128, 192 oder 256 ist die Rundenzahl also 10, 12 oder 14, das sind die grün markierten Tabelleneinträge.

Die Kernabbildung

Jede Runde von Rijndael besteht aus den vier Schritten

1. `SubBytes`, einer Substitution, die aus paralleler Anwendung der S-Box auf jedes Oktett besteht, siehe Abbildung 1,

2. `ShiftRows`, einer Permutation jeder Matrixzeile in sich, siehe Abbildung 3,

3. `MixColums`, einer linearen Abbildung (also komplizierter als nur Permutation) jeder Matrixspalte in sich, siehe Abbildung 4,

4. `AddRoundKey`, der Addition des Rundenschlüssels, siehe Abbildung 2.

Die Kernabbildung ρ umfasst also die ersten drei dieser Schritte. In der letzten Runde wird der dritte Schritt `MixColums` weggelassen; dadurch wird für die Umkehrabbildung, den Entschlüsselungsalgorithmus, ein strukturell gleicher Aufbau erreicht, und zur kryptographischen Stärke trägt dieser eine lineare Schritt am Ende ohnehin nichts mehr bei.

Die Abbildungen 1 bis 4 stammen aus der Wikipedia (für den Fall $N_b = N_k = 4$).

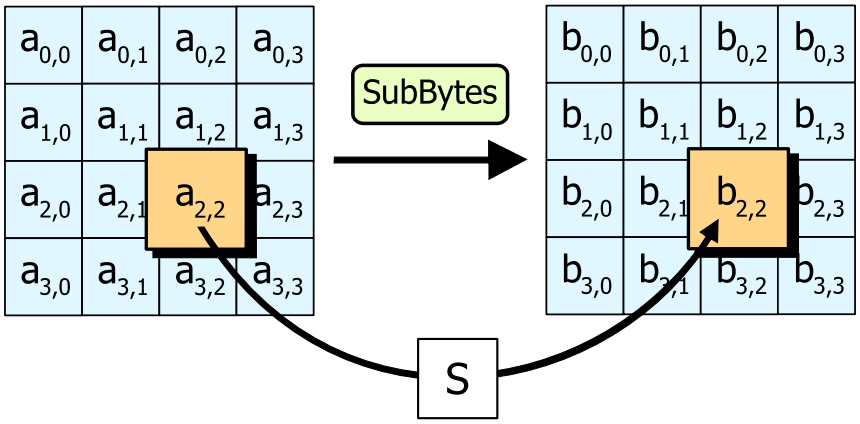


Abbildung 1: Die Wirkung von `SubBytes`

Der Substitutionsschritt `SubBytes`

Auf alle Oktette des aktuellen Zustands, also alle Einträge einer Matrix $b \in \mathbb{M}$, wird die S-Box S_{RD} einzeln angewendet. Diese ist Komposition $\mathrm{S}_{\mathrm{RD}} = f \circ g$ zweier Abbildungen, nämlich der Inversion $g = f_{-1}$ in $\mathbb{F}_{256}$,

$$g\colon \mathbb{F}_{256} \longrightarrow \mathbb{F}_{256}, \quad g(x) = \begin{cases} x^{-1} & \text{für } x \neq 0, \\ 0 & \text{für } x = 0, \end{cases}$$

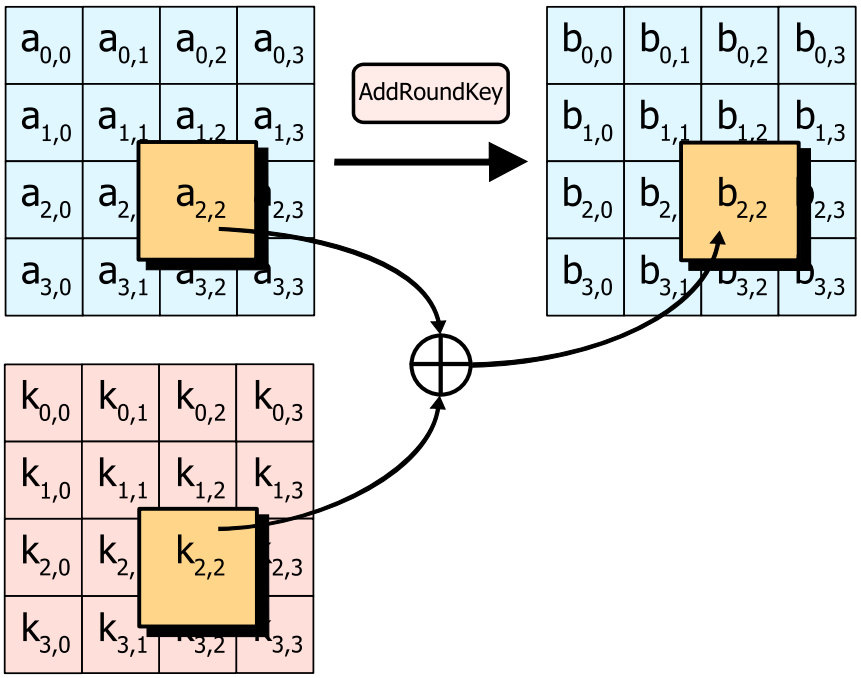


Abbildung 2: Die Wirkung von `AddRoundKey`

deren Nichtlinearitäts-Eigenschaften bestens bekannt sind, sowie der affinen Abbildung

$$f\colon \mathbb{F}_2^8 \longrightarrow \mathbb{F}_2^8, \quad \begin{pmatrix} x_7 \\ x_6 \\ x_5 \\ x_4 \\ x_3 \\ x_2 \\ x_1 \\ x_0 \end{pmatrix} \mapsto \begin{pmatrix} 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 & 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 & 1 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 \\ 1 & 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ 1 & 1 & 0 & 0 & 0 & 1 & 1 & 1 \\ 1 & 1 & 1 & 0 & 0 & 0 & 1 & 1 \\ 1 & 1 & 1 & 1 & 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} x_7 \\ x_6 \\ x_5 \\ x_4 \\ x_3 \\ x_2 \\ x_1 \\ x_0 \end{pmatrix} + \begin{pmatrix} 0 \\ 1 \\ 1 \\ 0 \\ 0 \\ 0 \\ 1 \\ 1 \end{pmatrix},$$

die so gewählt ist, dass

- S_{RD} keinen Fixpunkt hat, also $\mathrm{S}_{\mathrm{RD}}(b) \neq b$ für alle $b \in \mathbb{F}_{256}$,
- S_{RD} keinen „Anti-Fixpunkt“ hat, also $\mathrm{S}_{\mathrm{RD}}(b) \neq \bar{b}$ für alle $b \in \mathbb{F}_{256}$;

dabei ist $\bar{b} = (1 - b_7, \ldots, 1 - b_0)$ das BOOLEsche Komplement, also die bitweise logische Negation.

Nachteilig an dieser Abänderung der Inversionsabbildung ist, dass die involutorische Eigenschaft von f_{-1} verloren geht; für die Entschlüsselung muss daher eine andere S-Box $\mathrm{S}_{\mathrm{RD}}^{-1}$ implementiert werden.

Die Permutation der Zeilen `ShiftRows`

Jede der vier Zeilen des Zustands – der aktuellen $4 \times N_b$-Matrix – wird individuell zyklisch nach links geschoben, und zwar Zeile i um C_i Stellen. Die Werte der C_i hängen wie folgt von der Blocklänge ab:

N_b	C_0	C_1	C_2	C_3
4	0	1	2	3
5	0	1	2	3
6	0	1	2	3
7	0	1	2	4
8	0	1	3	4

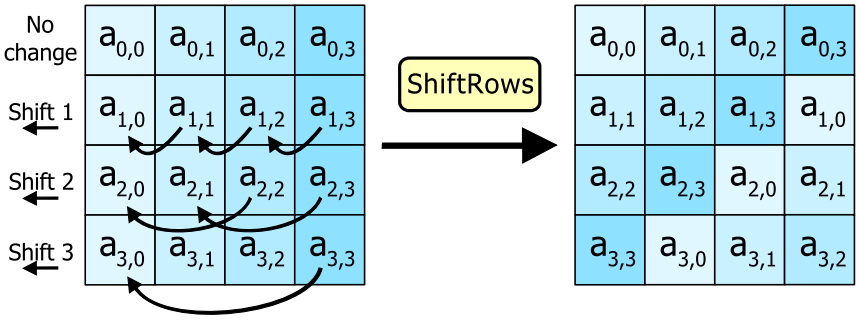


Abbildung 3: Die Wirkung von `ShiftRows`

Die lineare Transformation der Spalten `MixColumns`

Der Zustand wird Spalte für Spalte – d. h. die Zustandsmatrix $\in \mathbb{M}$ wird – von links mit einer festen 4×4-Matrix multipliziert; das Ziel dieses Schritts ist, erhebliche Diffusion zu erzeugen. Es handelt sich also um eine $\mathbb{F}_{256}$-lineare Abbildung

$$\mu \colon \mathbb{F}_{256}^4 \longrightarrow \mathbb{F}_{256}^4,$$

und zwar um die, die durch die Matrix

$$\begin{pmatrix} 02 & 03 & 01 & 01 \\ 01 & 02 & 03 & 01 \\ 01 & 01 & 02 & 03 \\ 03 & 01 & 01 & 02 \end{pmatrix}$$

beschrieben wird; die Einträge sind als Oktette in Hexadezimal-Darstellung zu interpretieren, also z. B. $03 = (00000011) \in \mathbb{F}_2^8$.

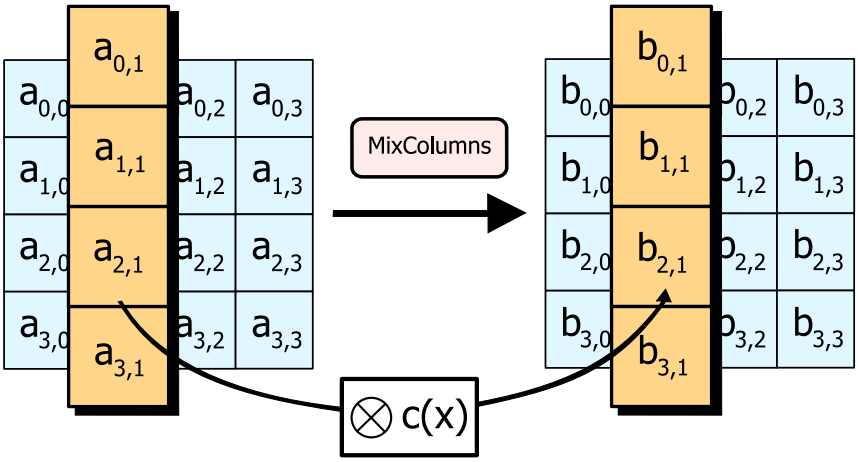


Abbildung 4: Die Wirkung von `MixColumns`

Die Entschlüsselung

Die gesamte Rijndael-Verschlüsselung ist die Komposition

$$F_k = \underbrace{\kappa_r \circ \tau \circ \sigma}_{i=r} \circ \ldots \circ \underbrace{[\kappa_i \circ \mu \circ \tau \circ \sigma]}_{i=1,\ldots,r-1} \circ \ldots \circ \kappa_0$$

von Abbildungen $\mathbb{M} \longrightarrow \mathbb{M}$; dabei ist κ_i die Addition des i-ten Rundenschlüssels, τ = `ShiftRows`, σ = `SubBytes` und μ = `MixColumns`. Die Kernabbildung ist also $\rho = \mu \circ \tau \circ \sigma$.

Für die Entschlüsselung brauchen wir die Umkehrabbildung, die zunächst so aussieht:

$$F_k^{-1} = \kappa_0^{-1} \circ \ldots \circ [\sigma^{-1} \circ \tau^{-1} \circ \mu^{-1} \circ \kappa_i^{-1}] \circ \ldots \circ \sigma^{-1} \circ \tau^{-1} \circ \kappa_r^{-1}.$$

Die versprochene Strukturgleichheit mit F_k ergibt sich aus einigen Umformungen:

- Es ist $\kappa_i^{-1} = \kappa_i$ für alle i.
- Da σ nur auf den Matrix-Einträgen wirkt, und zwar auf allen gleich, ist $\tau \circ \sigma = \sigma \circ \tau$.
- Schließlich ist $\kappa_i \circ \mu(x) = \mu(x) + k^{(i)} = \mu(x + \mu^{-1}(k^{(i)})) = \mu \circ \tilde{\kappa}_i(x)$, da μ linear ist. Also ist $\mu^{-1} \circ \kappa_i = \tilde{\kappa}_i \circ \mu^{-1}$, wobei $\tilde{\kappa}_i$ die binäre Addition von $\mu^{-1}(k^{(i)})$ ist; das wird für $i = 1, \ldots, r$ verwendet.

Damit ergibt sich

$$F_k^{-1} = \kappa_0 \circ \tau^{-1} \circ \sigma^{-1} \circ \ldots \circ [\tilde{\kappa}_i \circ \mu^{-1} \circ \tau^{-1} \circ \sigma^{-1}] \circ \ldots \circ \kappa_r.$$

Die Entschlüsselungsalgorithmus ist also genauso zusammengesetzt wie der Verschlüsselungsalgorithmus, nur dass

- das Schlüsselauswahlschema modifiziert ist: $\tilde{\kappa}_i$ statt κ_i und umgekehrte Reihenfolge der Teilschlüssel,
- `MixColumns` μ durch die lineare Umkehrabbildung μ^{-1} ersetzt wird,
- `Shiftrows` τ durch die Verschiebung nach rechts ersetzt wird,
- die S-Box S_{RD} durch die inverse Abbildung S_{RD}^{-1} ersetzt wird.

Die Schlüsselexpansion

Hier soll nur erwähnt werden, dass die Schlüsselexpansion zyklische Verschiebungen von Bytes innerhalb von Viererblöcken, die S-Box S_{RD}, sowie die Addition fester Konstanten verwendet.