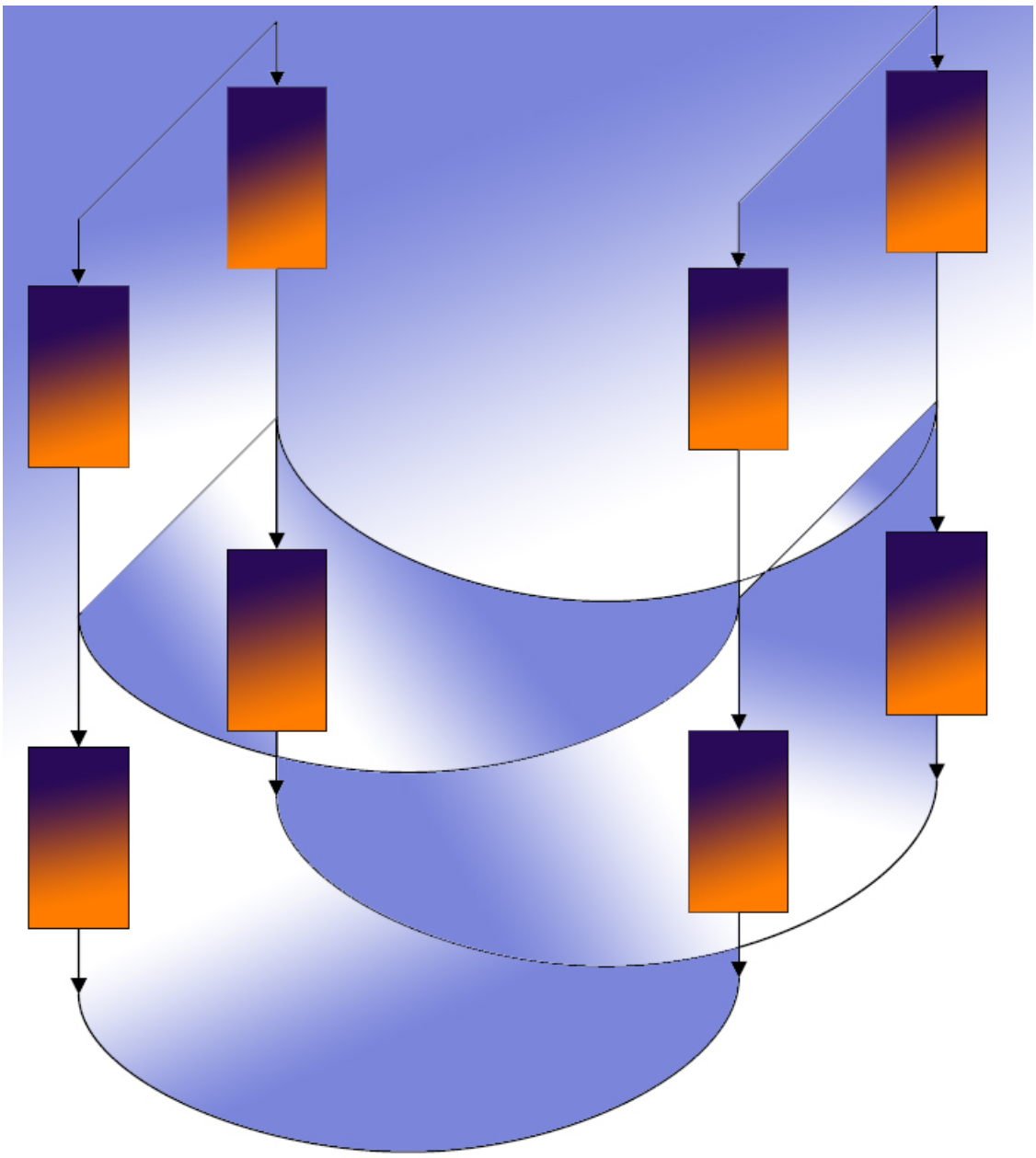
Boomerang- und Rechtecksangriff auf AES

Michael Gorski



Diplomarbeit

Michael Gorski
Universität Paderborn
26. Oktober 2006

Danksagung

Ich danke Professor Dr. Johannes Blömer für das interessante Thema, welches er mir zur Bearbeitung ermöglichte. Zahlreiche Gespräche mit ihm und seinem Mitarbeiter Dipl.-Inform. Volker Krummel halfen mir einen schnellen und theoretisch fundierten Einstieg in ein sehr spannendes Gebiet der Informatik zu erlangen. Auch ihm gebührt mein Dank.
Einen besonderen Dank möchte ich aber an meine Eltern richten, die mich in all den Jahren meines Studiums materiell sowie immateriell in besonderer Weise unterstützt haben.

Erklärung

Hiermit versichere ich, die vorliegende Diplomarbeit selbstständig verfasst und keine anderen Hilfsmittel als die angegebenen Quellen verwendet, sowie Zitate kenntlich gemacht zu haben.

Paderborn, 26. Oktober 2006

Inhaltsverzeichnis

Abbildungsverzeichnis

Einleitung

Die vorliegende Arbeit stammt aus dem Gebiet der Kryptographie, welche in der theoretischen Informatik und Mathematik angesiedelt ist. Die Kryptographie beschäftigt sich unter anderem mit der Verschlüsselung, der Authentifizierung und der Signierung von Daten. Diese Arbeit befasst sich mit der Verschlüsselung von Daten. Seit 1976 lassen sich Verfahren der Kryptographie in symmetrische und asymmetrische Verfahren unterteilen. Diese Einteilung folgt der Arbeit von Diffie und Hellman [16]. Symmetrische Verfahren benutzen für die Ver- und Entschlüsselung denselben Schlüssel, während asymmetrische Verfahren verschiedene Schlüssel verwenden. Generell sind symmetrische Verfahren jedoch im Vergleich zu asymmetrischen wesentlich schneller in der Ausführung. Bei symmetrischen Verfahren ist es allerdings erforderlich, dass der Schlüssel über einen abhörsicheren Kanal dem Kommunikationspartner zugesandt werden kann. Ein abhörsicherer Kanal macht das Mithören an der Kommunikation durch Dritte unmöglich. Eine Möglichkeit eines Schlüsselaustausches für ein symmetrisches Verfahren wäre die Benutzung eines asymmetrischen Verfahrens. Hierbei könnte der Schlüssel des symmetrischen Verfahrens durch die Hilfe eines asymmetrischen Verfahrens an den Kommunikationspartner übertragen werden. Wir beschäftigen uns in dieser Arbeit jedoch ausschließlich mit symmetrischen Verfahren. Dabei spielt ein bestimmtes Verfahren eine tragende Rolle: Die sogenannte Rijndael-Chiffre, welche im Jahre 2000 zum Advanced Encryption Standard oder kurz AES ernannt wurde.

1997 rief das National Institute of Standards and Technology (NIST) der USA einen Wettbewerb ins Leben, der einen neuen symmetrischen Verschlüsselungsstandard hervor bringen sollte. Dieser sollte den bisherigen Standard, den Data Encryption Standard (DES), ablösen, da DES seit 1997 als gebrochen galt. Das bedeutet, dass es Pläne für den Bau einer Maschine gibt, die den Schlüssel einer DES-Anwendung in weniger als drei Tagen bestimmen kann. Zu Beginn des Wettbewerbs standen 15 Kandidaten für den neuen Standard von Firmen und Wissenschaftlern aus 12 Ländern zur Auswahl. Diese Kandidaten wurden auf der ersten AES-Konferenz 1998 diskutiert und auf Schwachstellen untersucht. Eine Auflistung mit den dazugehörigen Erfindern befindet sich in Tabelle 1. Im März 1999 wurde in Rom die zweite AES-Konferenz abgehalten, auf der die Analysen der 15 Kandidaten diskutiert wurden. Im August 1999 veröffentlichte das NIST eine Liste von fünf Kandidaten, welche die Kriterien Sicherheit, Effizienz und Flexibilität am besten erfüllt haben und es somit in die zweite Runde schafften. Die fünf verbliebenen Kandidaten waren Twofish, Mars, RC6, Serpent und Rijndael. Auf der dritten AES-Konferenz im April 2000 wurden diese Kandidaten erneut diskutiert. Am 2. Oktober 2000 wurde Rijndael schließlich als neuer Advanced Encryption Standard gewählt. Der größte Vorteil von Rijndael gegenüber den anderen vier Kandidaten ist seine Flexibilität und effiziente Implementierbarkeit auf Smartcards. AES findet heute unzählige Anwendungen, beispielsweise in der geheimen Datenübertragung des Militärs oder aber auch bei der Verschlüsselung von WLAN-Übertragungen.

Ziel dieser Arbeit ist es, strukturelle Schwächen von AES zu analysieren. Dabei stellt sich die Frage, wie viel ein Angreifer wissen muss, um Teile des geheimen Schlüssel berechnen zu können. Wir werden in der Arbeit verschiedene Ansätze zur Sicherheitsanalyse

Bezeichnung	Entwickler	Art des Entwicklers
CAST-256	Entrust(CA)	Firma
Magenta	Deutsche Telekom (DE)	Firma
Crypton	Future Systems (KR)	Firma
Mars	IBM (USA)	Firma
DEAL	Outerbridge, Knudsen (USA-DK)	Wissenschaftler
RC6	RSA(USA)	Firma
DFC	ENS-CNRS (FR)	Wissenschaftler
Rijndael	Daemen and Rijmen (BE)	Wissenschaftler
E2	NTT(JP)	Firma
SAFER+	Cylink (USA)	Firma
Frog	TecApro (CR)	Firma
Serpent	Anderson, Biham, Knudsen (UK-IL-DK)	Wissenschaftler
HPC	Schroeppel (USA)	Wissenschaftler
Twofish	Counterpane (USA)	Firma
LOKI97	Brown et al. (AU)	Wissenschaftler

Tabelle 1: AES-Kandidaten der ersten Konferenz 1998 (Vgl. [15, S.3].)

von AES diskutieren. Dabei unterstellen wir, dass der Angreifer ein bestimmtes Wissen über das Verfahren besitzt, um damit zu zeigen, dass AES unter bestimmten Annahmen unsicher sein kann. Wir werden neue Techniken beschreiben, die effizientere Angriffe auf AES und somit eine schnelle Schlüsselsuche ermöglichen.

Die Arbeit ist wie folgt aufgebaut: In Kapitel 1 werden wir die wichtigsten mathematischen Grundlagen, die für das Verständnis der Rijndael-Chiffre notwendig sind, erklären. Dabei gehen wir hauptsächlich auf spezielle mathematische Eigenschaften ein, die bei der Rijndael Chiffre zur Anwendung kommen ([15], [31], [32] und [44]). Im anschließenden Kapitel 2 geben wir eine Einführung in die Thematik der Block-Chiffren. Wir werden hierbei den Aufbau und die wichtigsten Komponenten eines Substitutions-Permutations-Netzwerkes erläutern. Dadurch beschreiben wir wichtige Grundlagen für den Aufbau und die Funktionsweise der Rijndael-Chiffre ([36] und [38]). Diese wird in Kapitel 3 ausführlicher beschrieben. Wir gehen ebenfalls auf den Unterschied zwischen dem Advanced Encryption Standard und der Rijndael-Chiffre ein ([15]). Nach den ersten drei einführenden Kapiteln beschreiben wir in Kapitel 4 Techniken, um die Sicherheit von Blockchiffre untersuchen zu können. Wir beschränken uns hierbei auf verschiedene Formen der differentiellen Kryptoanalyse ([23], [41], [45], [43], [26], [3], [1], [25], [27], [24] und [6]). Die wichtigsten Techniken aus Kapitel 4 werden wir in Kapitel 5 auf den Advanced Encryption Standard anwenden ([10] und [24]). Im anschließenden Kapitel 6 zeigen wir, wie sich die verwendeten Techniken auf eine völlig neue Weise kombinieren lassen, um daraus neue Angriffe auf AES entstehen zu lassen. Wir stellen in diesem Kapitel drei von uns entwickelte Angriffe sowie einen Vergleich zu bisher bekannten Angriffen dar. Die Arbeit endet mit einer Zusammenfassung der Ergebnisse und einem Ausblick auf zukünftige Forschungsfelder.

Kapitel 1

Mathematische Grundlagen

In diesem Kapitel werden wir kurz die wichtigsten mathematischen Konzepte, welche für das weitere Verständnis notwendig sind, aufgreifen und erläutern. Hierbei gehen wir neben einer kurzen allgemeinen Einführung speziell auf die Verwendung dieser Konzepte bei der Rijndael-Chiffre ein. Wir orientieren uns weitgehend an der Darstellung der Grundlagen aus [15, S.9-29], [31, S.5-9], [32, S.1-30] und [44, S.23-110, 135-156].

1.1 Grundlegende Definitionen

Gruppen

Als Gruppe wird eine Menge von Elementen G zusammen mit einer Operation $+$, die auf den Elementen G definiert ist, bezeichnet, wenn folgende Eigenschaften gelten:

- Abgeschlossenheit:

$$\forall\ a, b \in G: \quad a + b \in G$$

- Neutrales Element: Es existiert ein $e \in G$, so dass gilt:

$$\forall\ a \in G: \quad a + e = a$$

- Assoziativität:

$$\forall\ a, b, c \in G: \quad (a + b) + c = a + (b + c)$$

- Inverses Element: Für alle $a \in G$ existiert ein $a^{-1} \in G$, so dass gilt:

$$a + a^{-1} = e$$

Eine Gruppe wird als **abelsch** oder **kommutativ** bezeichnet, wenn zusätzlich

$$\forall\ a, b \in G: \quad a + b = b + a$$

gilt.

Ringe

Eine Menge R zusammen mit zwei Operationen $+$ und $\cdot$ auf ihren Elementen wird als Ring bezeichnet, wenn:

- die Menge R zusammen mit der Operation $+$ eine abelsche Gruppe ist,

- die Operation $\cdot$ ist abgeschlossen und assoziativ über R:

$$\forall\, a, b, c \in R: \quad (a \cdot b) \cdot c = a \cdot (b \cdot c),$$

- für die Operation $\cdot$ existiert ein neutrales Element,
- das Distributivgesetz gilt:

$$\forall\, a, b, c \in R: \quad a \cdot (b + c) = a \cdot b + a \cdot c$$

Wir sprechen von einem **kommutativen** Ring, wenn die Operation $\cdot$ kommutativ ist.

Körper

Die Menge F mit den Operationen $+$ und $\cdot$ über F beschreibt genau dann einen Körper, wenn:

- F mit den Operationen $+$ und $\cdot$ einen kommutativen Ring beschreibt.
- Für alle Elemente aus F ein inverses Element unter der Operation $\cdot$ ausgenommen für das neutrale Element der Operation $+$ über F existiert.

Besitzt ein Körper m Elemente, so bezeichnen wir ihn als **endlichen** Körper.

1.2 Besonderheiten von Rijndael

Der Körper $\mathbb{F}_{2^8}$

Besteht ein Körper aus einer primen Anzahl von p Elementen, so können all diese Elemente durch die ganzen Zahlen $0, 1, \ldots, p-1$ dargestellt werden. Sowohl die Addition als auch die Multiplikation können damit über den ganzen Zahlen modulo p ausgeführt werden. Dies ist nicht mehr möglich, sobald die Anzahl der Elemente eine Primzahlpotenz der Form $m = p^n$ mit $n > 1$ beschreibt. In diesem Fall stellen wir Elemente aus $\mathbb{F}_{p^n}$ als Polynom über $\mathbb{F}_p$ dar.

Wir werden im Folgenden auf eine allgemeine Darstellung verzichten und uns stattdessen direkt dem in der Rijndael-Chiffre verwendeten Körper $\mathbb{F}_{2^8}$ zuwenden. Ein Element $a \in \mathbb{F}_{2^8}$ lässt sich als Polynom vom Grad 8 als

$$a(x) = a_7x^7 + a_6x^6 + a_5x^5 + a_4x^4 + a_3x^3 + a_2x^2 + a_1x + a_0$$

schreiben, wobei $a_i \in \mathbb{F}_2$ mit $0 \leq i \leq 7$ ist. Wenn wir im Folgenden die Hexadezimaldarstellung verwenden, so kennzeichnen wir dieses durch den Index 16. Die Binärdarstellung wird mit 2 indiziert.

Beispiel 1.2.1 *Sei $a = 88_{16}$ in Hexadezimaldarstellung, welches $a = 10001000_2$ als Binärdarstellung entspricht. Daraus erhalten wir die Polynomdarstellung*

$$a(x) = x^7 + x^3.$$

Die Addition

Um zwei Elemente $a, b \in \mathbb{F}_{2^8}$ zu addieren, bilden wir zunächst deren Polynomdarstellung und addieren diese anschließend. Dieses erreichen wir durch die Addition der Koeffizienten mit gleichen Exponenten von x, wobei die Addition der Koeffizienten über $\mathbb{F}_2$ erfolgt. Da wir die Addition über $\mathbb{F}_2$ durchführen, können wir auch equivalent das exklusive Oder der Koeffizienten mit gleichen Exponenten von x berechnen:

$$c(x) = a(x) + b(x) = \sum_{i=0}^{7} (a_i \oplus b_i)x^i$$

Die Koeffizientenvektoren können, sofern sie dieselbe Dimension haben, auch direkt per XOR verknüpft werden.

Die Multiplikation

Damit wir die Multiplikation als abgeschlossene Operation über $\mathbb{F}_{2^8}$ durchführen können, definieren wir uns ein Polynom $m(x) \in \mathbb{F}_2[x]$ vom Grad 8. Dieses Polynom soll irreduzibel sein. Das bedeutet, dass es über $\mathbb{F}_2$ keine zwei Polynome $q(x)$ und $w(x)$ mit Koeffizienten in $\mathbb{F}_2$ gibt, so dass

$$m(x) = q(x)w(x)$$

gilt, wobei $q(x)$ und $w(x)$ vom Grad größer Null sind. Anders formuliert: Ein irreduzibles Polynom besitzt nur konstante Polynome und sich selbst als Faktoren. Für die Rijndael Chiffre wurde das irreduzible Polynom

$$m(x) = x^8 + x^4 + x^3 + x + 1$$

gewählt. Die Multiplikation zweier Polynome $a(x)$ und $b(x)$ ist nun definiert als die algebraische Multiplikation der beiden Polynome modulo $m(x)$. Wir erhalten:

$$c(x) \equiv a(x)b(x) \bmod m(x)$$

Nun können wir für jedes Polynom $a(x)$ ein multiplikatives Inverses finden, da $m(x)$ ein irreduzibles Polynom ist. Mit dem erweiterten Euklidischen Algorithmus lässt sich

$$ggT(a(x), m(x)) = 1 = a(x)b(x) + c(x)m(x)$$

berechnen. Dies ergibt modulo $m(x)$:

$$a(x)b(x) \equiv 1 \bmod m(x)$$

Somit erhalten wir mit $b(x)$ für jedes Polynom $a(x)$ ein Inverses. Die Null stellt eine Ausnahme dar, da sie als zu sich selbst invers definiert wird.

Beispiel 1.2.2 *Es seien die binären Darstellungen von a und b durch $a = 01010111_2$ und $b = 10000011_2$ gegeben. Die Multiplikation der entsprechenden Polynome modulo $m(x) = x^8 + x^4 + x^3 + x + 1$ liefert:*

$$\begin{aligned} & (x^6 + x^4 + x^2 + x + 1) \cdot (x^7 + x + 1) \\ = \; & x^{13} + x^{11} + x^9 + x^8 + x^6 + x^5 + x^4 + x^3 + 1 \\ = \; & x^7 + x^6 + 1 \bmod x^8 + x^4 + x^3 + x + 1 \end{aligned}$$

Als Binärdarstellung erhalten wir $c = 11000001_2$.

Die Multiplikation über dem Körper $\mathbb{F}_{2^8}$ modulo $x^8 + 1$

Für die in folgenden Kapiteln erläuterte SubByte Operation benötigen wir die Multiplikation über $\mathbb{F}_{2^8}$ modulo einem Polynom $r(x) = x^8 + 1$, welches nicht irreduzibel ist. Dadurch besitzt nicht jedes Element ein multiplikatives Inverses. Um nun gewährleisten zu können, dass die Multiplikation umgekehrt werden kann, verwenden wir das Polynom

$$b(x) = x^4 + x^3 + x^2 + x + 1,$$

welches zu $r(x)$ teilerfremd ist. Aus dem erweiterten Euklidischen Algorithmus erhalten wir das zu $b(x)$ multiplikativ inverse Polynom:

$$b^{-1}(x) = x^6 + x^3 + x$$

Wir können nun die Reduktion modulo $r(x)$ der Exponenten für alle $i \geq 0$ als

$$x^i \equiv x^{i \bmod 8} \bmod x^8 + 1$$

beschreiben.

Der Ring $\mathbb{F}_{2^8}$

Wir betrachten nun Polynome, deren Koeffizienten Elemente aus $\mathbb{F}_{2^8}$ repräsentieren und einen Grad kleiner gleich 3 besitzen. Diese kommen bei der MixColumns Operation von Rijndael zur Anwendung. Die Addition zweier Polynome $a(x)$ und $b(x)$ verläuft äquivalent zu der Addition, die im vorherigen Abschnitt beschrieben wurde. Hierbei werden die Koeffizienten mit gleichen Exponenten von x addiert.

Für die Multiplikation wird das Polynom

$$t(x) = x^4 + 1$$

zur Reduktion verwendet. Die Reduktion erfolgt hier über die Betrachtung der Exponenten modulo 4. Aus der algebraischen Multiplikation von $a(x)b(x) = c(x)$ reduzieren wir nun die Koeffizienten von $c(x)$ durch:

$$x^i \equiv x^{i \bmod 4} \bmod x^4 + 1$$

Da t(x) nicht irreduzibel über $\mathbb{F}_{2^8}$ ist, können wir nicht für jedes Element ein multiplikatives Inverses finden. Wir benutzen deshalb für die Multiplikation das Polynom

$$g(x) = 03x^3 + 01x^2 + 01x + 02,$$

dessen Koeffizienten in Hexadezimalschreibweise aufgeführt sind und Elemente aus $\mathbb{F}_{2^8}$ darstellen. Das $g(x)$ zu $t(x)$ teilerfremd ist, lässt sich mit dem erweiterten Euklidischen Algorithmus zeigen, da wir auch das multiplikativ inverse Polynom

$$g^{-1}(x) = 0Bx^3 + 0Dx^2 + 09x + 0E,$$

berechnen können.

Kapitel 2

Block Chiffren

2.1 Grundlegende Blockchiffren

Blockchiffren sind Verschlüsselungsverfahren, die aus Klartextblöcken der Länge n-Bit unter Anwendung eines k-Bit-Schlüssels Chiffretextblöcke der Länge n-Bit erzeugen. Wir nennen n die Blocklänge, wobei n einer natürlichen Zahl entspricht. Die Entschlüsselung erfolgt durch Eingabe eines Chiffretextblocks der Länge n-Bit zusammen mit demselben Schlüssel, der auch bei der Verschlüsselung verwendet wurde, um einen Klartextblock gleicher Größe zu erzeugen. Es handelt sich hierbei um symmetrische Blockchiffren, da jeweils zur Ver- und Entschlüsselung derselbe Schlüssel verwendet wird. Abbildung 2.1 schematisiert diesen Zusammenhang. Wir entnehmen die Erläuterungen aus [36, S.223-259], [12, S.68-69], [31, S.13-14] und [38, S.24-26, 54-55].

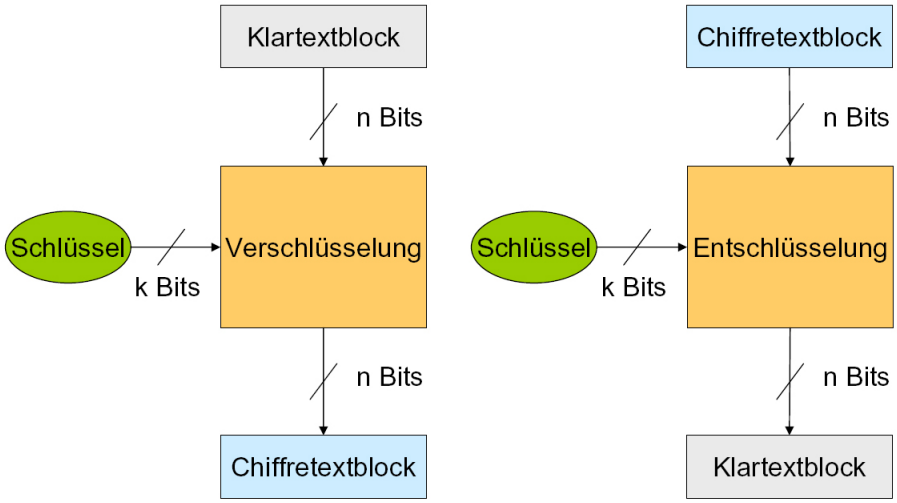


Abbildung 2.1: Ver- und Entschlüsselung mit einer einfachen Blockchiffre

Ist ein zu verschlüsselnder Text größer als die bei einer Blockchiffre verwendeten Klartextblöcke, so wird der Text entsprechend aufgeteilt und gegebenenfalls aufgefüllt, um die passende Eingabegröße der verwendeten Blockchiffre zu gewährleisten.

Damit aus jedem Chiffretextblock der dazugehörige Klartextblock eindeutig rekonstruiert werden kann, muss die Verschlüsselungsfunktion injektiv sein. Nun sind die Verschlüsselungsfunktionen einer Blockchiffre jedoch Permutationen, da eine injektive Abbildung, die n-Bits auf n-Bits abbildet, bijektiv ist. Wir können aus n-Bits 2^n mögliche Klartextblöcke generieren, die auf 2^n mögliche Chiffretextblöcke abgebildet werden. Es existieren genau $2^n!$ solcher bijektiver Abbildungen. Der verwendete Schlüssel wählt nun eine bestimmte dieser Abbildungen aus, um Klartextblöcke in Chiffretextblöcke zu transformieren. Das Ziel einer guten Blockchiffre ist es nun, den Anschein zu erwecken, als würde eine

bestimmte Abbildung rein zufällig durch den Schlüssel gewählt werden. Darüber hinaus sollte es möglich sein mit dem Schlüsselraum, also der Menge aller möglichen Schlüssel, alle $2^n!$ Abbildungen auswählen zu können. Eine Chiffre, die diese Kriterien erfüllt, wird auch als zufällige Chiffre oder „random cipher“ bezeichnet.

Die Konstruktion einer solchen Chiffre ist nicht möglich, da der Schlüsselraum von der Größe $2^n!$ sein müsste. Eine solche Chiffre wäre für die Praxis völlig unbrauchbar, da die Speicherung eines einzelnen Schlüssels sehr viel Speicher benötigt. Um dieses zu erläutern, bringen wir an dieser Stelle ein numerisches Beispiel.

Beispiel 2.1.1 *Haben wir eine Blocklänge von* 64 *Bit, so können wir damit* $2^{64}!$ *bijektive Abbildungen erzeugen, welches gleichzeitig der Größe des Schlüsselraums entspricht. Ein einzelner Schlüssel hätte damit die Länge von*

$$\log\left(2^{64}!\right) \approx 2^{70} \approx 10^{21}\, Bit.$$

Das entspricht etwa 10^{11} *Gigabyte an Speicher.*

In der Praxis haben wir es oft mit k-Bit Schlüssellängen zu tun, wobei $k \ll 2^n!$ ist. Jede der 2^k möglichen Schlüssel bestimmt nun eine Abbildung der Menge der Klartextblöcke auf die Menge der Chiffretextblöcke. Die Wahl der Schlüssellänge lässt allerdings nur einen sehr kleinen Teil der möglichen Abbildungen zur Auswahl zu, nämlich genau:

$$\frac{2^k}{2^n!}$$

Bei $k = n$ ergibt sich $\frac{1}{(2^n-1)!}$.

Aus diesem Grund und da die Erzeugung von echtem Zufall nicht möglich ist um die Abbildung möglichst zufällig auszuwählen, fordert Shannon in [40] ein hohes Maß an Konfusion und Diffusion in die Chiffre zu integrieren. Diese Konzepte bilden die Grundlage für alle modernen Blockchiffren. Unter Konfusion versteht man, dass die statistischen Eigenschaften des Chiffretextes einer Folge aus Zufallszahlen entsprechen sollte. Die Beziehung des Chiffretextblocks zum verwendeten Schlüssel soll damit möglichst schwierig gestaltet werden. Diffusion bedeutet, dass die Bits des Klartextblocks auf möglichst viele Bits des Chiffretextblocks verteilt werden sollen. Dies soll die Zuordnung eines Klartextblocks zu einem Teil des Chiffretextblocks und dem Schlüssel erschweren.

2.2 Iterierte Blockchiffren

Die Komplexität einer Chiffre kann man durch mehrmaliges hintereinander ausführen mit verschiedenen Schlüsseln erhöhen. Es lässt sich auf diese Weise eine bessere Konfusion und eine höhere Diffusion erreichen. Siehe dazu [12, S.69] und [31, S.14-15]. Eine einfache Möglichkeit wäre es, auf den Klartextblock eine Chiffre mehrfach anzuwenden. Dabei wird eine einzelne Chiffre als Rundenfunktion bezeichnet und das System von mehreren Hintereinanderausführungen dieser Rundenfunktionen als iterierte Blockchiffre. Hierbei wird gefordert, dass die Rundenfunktionen jeweils aus mindestens zwei Operationen bestehen, die untereinander nicht vertauschbar sein dürfen. Eine relativ einfache iterierte Blockchiffre ist beispielsweise ein Substitutions-Permutations-Netzwerk, welches wir im Folgenden näher kennenlernen werden.

Je mehr Runden man einsetzt, umso höher wird die Sicherheit der Chiffre. Gleichzeitig sinkt aber auch deren Geschwindigkeit in der Ver- und Entschlüsselung. Hieraus entsteht ein Zielkonflikt zwischen Sicherheit auf der einen und Geschwindigkeit auf der anderen Seite.

2.3 Substitutions Permutations Netzwerke

Eine grundlegende Form, auf deren Struktur viele moderne Blockchiffren aufbauen, darunter auch AES, ist ein Substitutions Permutations Netzwerk(SPN), welches von Feistel in [21] vorgestellt werden. Im Folgenden wird diese Struktur kurz erklärt. Wir entnehmen unsere Erläuterung dabei aus [23, S.3-5] und [42, S.74-79]. Ein SPN besteht aus einer beliebigen Anzahl von Runden, in denen jeweils die gleichen Runden-Algorithmen durchgeführt werden. Die einzelnen Runden bestehen hierbei aus Substitution, Permutation und Schüssel-Addition. Die Funktionsweise dieser Operationen wird nun am Beispiel des in Abbildung 2.2 dargestellten SPN's erklärt.

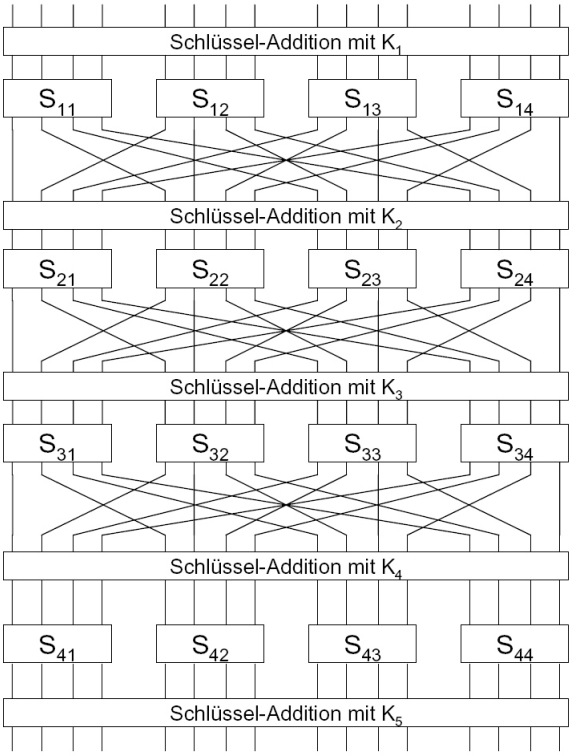


Abbildung 2.2: Darstellung eines Substitutions Permutations Netzwerks

Die Verschlüsselung eines Klartextes erfolgt nun durch die Anwendung einer beliebigen Anzahl von Runden, in denen die im Folgenden dargestellten Schritte erläutert werden. Nach der letzten Runde wird erneut eine Schlüssel-Addition vollzogen, um eine höhere Sicherheit gegen Kryptoanalysen gewährleisten zu können.

Ähnlich wie die Verschlüsselung vollzieht sich auch die Entschlüsselung. Hierbei werden alle Schritte der Verschlüsselung in umgekehrter Reihenfolge durchlaufen. Dabei ist wichtig, dass die Abbildung der Eingabe- auf die Ausgabe-Bits von Substitution und Permutation eine bijektive Abbildung darstellen. Für die Entschlüsselung werden die inversen Substitutionen und Permutationen durch Vertauschung von Ein- und Ausgabe-Bits gesetzt. In Abbildung 2.2 entfällt der Permutationsschritt nach der letzten Runde, um die Struktur der Ver- und Entschlüsselung gleich zu halten.

2.3.1 Substitution

Der Substitutionsschritt wird durch eine sogenannte S-Box ausgeführt. Da die Ausgabe-Bits einer solchen S-Box nicht durch eine lineare Funktion beschrieben werden, stellt sie einen nicht linearen Zusammenhang zwischen den Werten der Ein- und Ausgabe dar. Nebenbei sollte hier bemerkt werden, dass die Substitution die einzige nicht lineare Operation eines SPN's darstellt. Das in Abbildung 2.2 aufgezeigte SPN hat eine Blocklänge von 16

Bit, die vor der Substitution in 4 Bit Blöcke aufgeteilt werden, die als Eingabe einer S-Box dienen. Die nicht-lineare Transformation der S-Boxen wird in diesem Beispiel durch die in Tabelle 2.1 dargestellte Zuordnung der Ein- und Ausgabe-Bits repräsentiert. Dadurch kann jede der 16 S-Boxen durch ein einfaches Table-lookup implementiert werden.[1]

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

Tabelle 2.1: S-Box (Substitution)

2.3.2 Permutation

In diesem Schritt werden die Positionen der Bits nach einer vorgegebenen Permutation ausgetauscht.[2] In unserem Beispiel haben wir 16 Bits, die nach der Permutation in Tabelle 2.2 gemischt werden. Hierbei werden die Ausgaben der vier S-Boxen einer Runde, genau wie der Klartext, als ein zusammenhängender Bit-String aufgefasst.

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

Tabelle 2.2: Permutation

2.3.3 Schüssel-Addition

Viele moderne Blockchiffren, so auch die Rijndael-Chiffre, besitzen einen Schlüssel, aus dem nach einem Schlüsselerzeugungsalgorithmus einzelne Rundenschlüssel extrahiert werden. Ein solcher Rundenschlüssel wird durch ein Bit-weises XOR mit dem entsprechenden Datenblock in die Runden des SPN's integriert.

[1] Eine Chiffre, welche nur aus dem Substitutionsschritt besteht, wird auch als Substitutions-Chiffre bezeichnet. Siehe dazu [42, S.7].

[2] Eine Chiffre, welche nur aus dem Permutationsschritt besteht, wird auch als Permutations-Chiffre bezeichnet. Siehe dazu [42, S.8].

Kapitel 3

Die Rijndael-Chiffre

In diesem Kapitel geben wir eine Beschreibung der Rijndael-Chiffre [15], welche von Daemen und Rijmen entwickelt wurde. Diese Chiffre wurde im Jahre 2001 zum Advanced Encryption Standard (AES) ernannt. Es handelt sich hierbei um eine iterierte Blockchiffre, die auf einer sogenannten Square-Struktur (siehe [14]) basiert. Die Rijndael-Chiffre erlaubt es, Texte mit Block- und Schlüssellängen zwischen 128 und 256 Bit zu verschlüsseln. Dabei müssen diese jeweils einem Vielfachen von 32 Bit in dem eben genannten Intervall entsprechen, um die Struktur der Chiffre beibehalten zu können.[1]

AES unterscheidet sich von der Rijndael-Chiffre dadurch, dass die Blocklänge der Texte auf 128 Bit beschränkt ist. Die Schlüssellängen sind dabei ebenfalls auf 128, 192 oder 256 Bit festgelegt. Wie wir im vorherigen Kapitel gesehen haben, wird ein Klartext zusammen mit einem Schlüssel in einen Chiffretext transformiert. Die Entschlüsselung transformiert einen Chiffretext zusammen mit dem Schlüssel, welcher auch für die Verschlüsselung benutzt wurde, in den ursprünglichen Klartext.

Alle Operationen der Chiffre laufen auf sogenannten Zuständen ab. Ein Zustand wird durch eine Matrix bestehend aus vier Zeilen und N_b Spalten repräsentiert, in welcher jedes Feld genau ein Byte enthält. N_b ist hierbei variabel, wurde bei AES jedoch auf $N_b = 4$ festgelegt. Wir gehen im Folgenden nur noch auf diese AES Konvention ein. Haben wir einen, 16 Byte Klartextblock

$$P = p_0 p_1 p_2 \cdots p_{15},$$

so ergibt sich die entsprechende Transformation in einen Zustand, wie in Abbildung 3.1 dargestellt. Die Bytes des Klartextblocks werden in einen Zustand überführt, indem jede Spalte des Zustands hintereinander mit Bytes des Klartextblocks gefüllt werden.

p_0	p_4	p_8	p_{12}
p_1	p_5	p_9	p_{13}
p_2	p_6	p_{10}	p_{14}
p_3	p_7	p_{11}	p_{15}

Abbildung 3.1: Darstellung eines Zustandes von AES

Bezeichnen wir mit i eine Zeile und mit j eine Spalte der Zustandsmatrix, so können wir formal eine Transformation durch

$$a_{i,j} = p_{i+4j}$$

[1] Es wäre theoretisch auch möglich, die obere Grenze von 256 Bit zu überschreiten; jedoch wird dazu aus Aspekten der Sicherheit im Moment kein Bedarf gesehen.

beschreiben, wobei $0 \leq i < 4$, $0 \leq j < N_b$ ist und $a_{i,j}$ eine Zelle (ein Byte) der Zustandsmatrix repräsentiert. Die Transformation eines Zustands in einen Textblock vollzieht sich durch:

$$p_i = a_{i \bmod 4, \lfloor i/4 \rfloor}$$

Dieses geschieht, unabhängig davon, ob ein Klar- oder Chiffretext transformiert wird. Ein Schlüssel wird auf dieselbe Art und Weise in einen Zustand transformiert. Auch hier hängen die Anzahl der Spalten der Zustandsmatrix von der Anzahl der Schlüsselbits ab.

3.1 Ablauf

Eines der Ziele des Designs von Rijndael war es, die Chiffre möglichst einfach zu halten. Darum wurden vier Operationen definiert, die sich in jeder Runde wiederfinden lassen. Die Rundenanzahl hängt von der Blocklänge der Texte und ihrer Schlüssellänge ab. Gehen wir, wie bei AES, von einer Blocklänge von 128 Bit aus, und einer Schlüssellänge von 128, 192 oder 256 Bit, so beträgt die entsprechende Rundenanzahl $N_r = 10$, 12 oder 14. Tabelle 3.1 zeigt die benötigten Runden abhängig von N_k und N_b, wobei N_k die Anzahl der Spalten der Zustandsmatrix des Schlüssels darstellt wird. Ein 192 Bit Schlüssel besteht aus $N_k = 6$ Spalten.

	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

Tabelle 3.1: Anzahl der Runden

Der Ablauf von Rijndael ist in Abbildung 3.2 dargestellt. Nachdem die Rundenschlüssel mit KeyExpansion erstellt wurden, die wir im Folgenden näher beschreiben, beginnt der eigentliche Ablauf der Chiffre. Vor der ersten Runde wird ein initiales AddRoundKey angewendet. Es folgen $N_r - 1$ Runden, die aus den Operationen SubBytes, ShiftRows, MixColumns und AddRoundKey bestehen. Die letzte Runde der Chiffre unterscheidet sich durch das Fehlen der MixColumns Operation aus Gründen der Symmetrie.

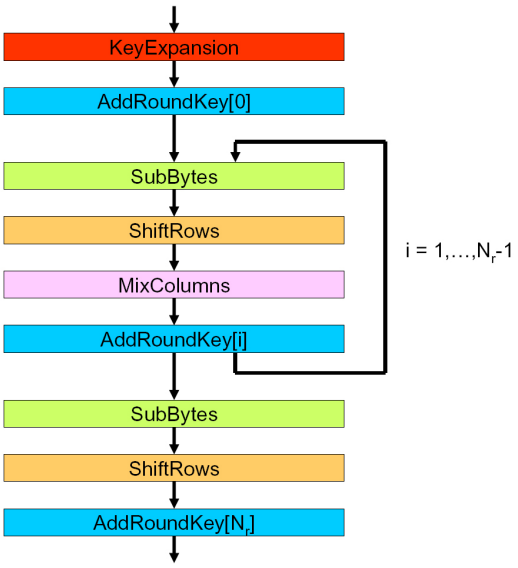


Abbildung 3.2: Der Ablauf der Rijndael-Chiffre (Verschlüsselung)

Wir werden im Folgenden die einzelnen Operationen der Runden detailiert beschreiben. Die KeyExpansion werden wir gesondert betrachten, da es sich hierbei um keine Rundenoperation handelt, sondern um eine Operation, welche die Rundenschlüssel erzeugt.

3.1.1 SubBytes

SubBytes stellt die einzige nicht lineare Operation der Chiffre dar. Hierbei wird jeder Eintrag der Zustandsmatrix, also jedes Byte, durch eine S-Box unabhängig von den anderen Einträgen der Zustandsmatrix transformiert. Um die Chiffre möglichst einfach zu halten, wird die gleiche S-Box für alle Bytes eines Zustandes verwendet.

Jedes Byte g wird als ein Element des Körpers $\mathbb{F}_2[X]/m = \mathbb{F}_{2^8}$ betrachtet und über diesem invertiert. Die Multiplikation über diesem Körper mit dem irreduzible Polynom $m(x)$ haben wir bereits im ersten Kapitel genau betrachtet. Durch dieses lässt sich, wie bereits gezeigt, zu jedem g ein multiplikatives Inverses g^{-1} finden. Als nächstes folgt eine affine Abbildung, bei der ein Byte g als Element des Rings $\mathbb{F}_2[X]/(x^8+1)$ aufgefasst wird. Da $r(x) = x^8 + 1$ nicht irreduzibel in $\mathbb{F}_2[X]$ ist, besitzt nicht jedes g ein multiplikatives Inverses. Wir führen nun folgende Operation durch:

$$c = f(g) = (x^4 + x^3 + x^2 + x + 1)\, g + (x^6 + x^5 + x + 1) \bmod x^8 + 1$$

Die affine Transformation lässt sich auch als Matrixschreibweise repräsentieren:

$$\begin{pmatrix} c_7 \\ c_6 \\ c_5 \\ c_4 \\ c_3 \\ c_2 \\ c_1 \\ c_0 \end{pmatrix} = \begin{pmatrix} 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 & 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 & 1 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 \\ 1 & 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ 1 & 1 & 0 & 0 & 0 & 1 & 1 & 1 \\ 1 & 1 & 1 & 0 & 0 & 0 & 1 & 1 \\ 1 & 1 & 1 & 1 & 0 & 0 & 0 & 1 \end{pmatrix} \times \begin{pmatrix} g_7 \\ g_6 \\ g_5 \\ g_4 \\ g_3 \\ g_2 \\ g_1 \\ g_0 \end{pmatrix} \oplus \begin{pmatrix} 0 \\ 1 \\ 1 \\ 0 \\ 0 \\ 0 \\ 1 \\ 1 \end{pmatrix}$$

Für die Entschlüsselung ist es wichtig, dass SubBytes invertierbar ist. Die inverse Operation zu SubBytes wird als InvSubBytes bezeichnet. Eine Invertierung der affinen Abbildung ist möglich, da die Polynome $r(x) = x^8 + 1$ und $b(x) = x^4 + x^3 + x^2 + x + 1$ teilerfremd sind. Die Invertierung eines Bytes ist ohne Probleme möglich, da diese über einem Körper stattfindet. Als Matrixschreibweise ergibt sich für die invertierte affine Abbildung:

$$\begin{pmatrix} g_7 \\ g_6 \\ g_5 \\ g_4 \\ g_3 \\ g_2 \\ g_1 \\ g_0 \end{pmatrix} = \begin{pmatrix} 0 & 1 & 0 & 1 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 1 & 0 & 0 & 1 \\ 1 & 0 & 0 & 1 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 1 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 & 1 & 0 & 1 \\ 1 & 0 & 0 & 1 & 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 & 1 & 0 & 0 & 1 \\ 1 & 0 & 1 & 0 & 0 & 1 & 0 & 0 \end{pmatrix} \times \begin{pmatrix} c_7 \\ c_6 \\ c_5 \\ c_4 \\ c_3 \\ c_2 \\ c_1 \\ c_0 \end{pmatrix} \oplus \begin{pmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 1 \\ 0 \\ 1 \end{pmatrix}$$

Die Werte von SubBytes und InvSubBytes können einmal berechnet werden und lassen sich bei der weiteren Anwendung durch eine einfache Abfrage bestimmen. Abbildung 3.3 zeigt schematisch die Operationen SubBytes und InvSubBytes für jeweils eine Zelle der Zustandsmatrix (d.h. ein Byte). Ein Byte-Wert b wird durch die S-Box zu c und ein Wert c durch die inverse S-Box zu b transformiert.

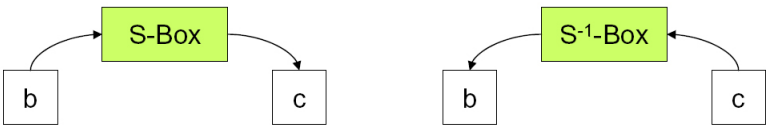


Abbildung 3.3: Die Operationen SubBytes (links) und InvSubBytes (rechts)

Sowohl die S-Box von Rijndael als auch die inverse S-Box sind jeweils in Anhang A.1 und A.2 aufgeführt.

3.1.2 ShiftRows

Die ShiftRows-Operation führt zyklische Shifts nach links auf den Zeilen der Zustandsmatrix aus. Dabei soll ein möglichst hohes Maß an Diffusion erreicht werden, um beispielsweise gegen Angriffe mit verkürzten Differentialen [30] geschützt zu sein. Die Anzahl der Shifts ist dabei für jede Zeile verschieden und hängt von der Blocklänge ab. Die Zeile i wird dabei um C_i Bytes zyklisch nach links geshiftet. Tabelle 3.2 listet die Anzahl der Shifts abhängig von der Blocklänge (hier durch die Anzahl der Spalten der Zustandsmatrix repräsentiert) und der entsprechenden Zeile auf.

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
5	0	1	3	4

Tabelle 3.2: Anzahl der zyklischen Shifts

Die zu ShiftRows inverse Operation ist InvShiftRows. Bei dieser erfolgt ein zyklischer Shift um die in Tabelle 3.2 aufgeführten Werte nach rechts. Die Abbildung 3.4 beschreibt die ShiftRows-Operation speziell für AES. Die Markierung des jeweils ersten Elementes einer Zeile soll dabei lediglich die Shifts verdeutlichen.

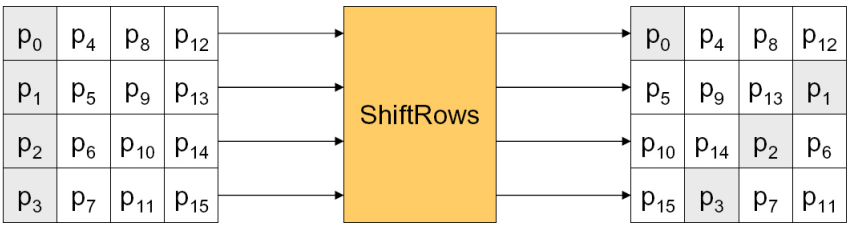


Abbildung 3.4: Die Operationen ShiftRows

3.1.3 MixColumns

MixColumns operiert auf den Spalten der Zustandsmatrix. Da jede Zustandsmatrix genau vier Zeilen besitzt und jeder Matrixeintrag genau einem Byte entspricht, hat jede Spalte eine Größe von 4 Bytes. Diese werden als Polynom $c(x)$ über dem Ring $\mathbb{F}_{2^8}[X]/(x^4+1)$ aufgefasst. Es wird nun ein Polynom

$$g(x) = 03x^3 + 01x^2 + 01x + 02$$

definiert, welches zu $t(x) = x^4 + 1$ teilerfremd ist und ein Element aus $\mathbb{F}_{2^8}[x]$ repräsentiert. Dies ist von Bedeutung, um MixColumns später invertieren zu können. Die Koeffizienten von $g(x)$ sind in Hexadezimaldarstellung geschrieben, dies gilt ebenso für alle Darstellungen in diesem Abschnitt. Das Polynom mit den Koeffizienten einer Spalte der Zustandsmatrix wird nun zu

$$a(x) = g(x)\, c(x) \bmod x^4 + 1$$

transformiert. Als Matrixschreibweise erhalten wir für diese Operation:

$$\begin{pmatrix} a_0 \\ a_1 \\ a_2 \\ a_3 \end{pmatrix} = \begin{pmatrix} 02 & 03 & 01 & 01 \\ 01 & 02 & 03 & 01 \\ 01 & 01 & 02 & 03 \\ 03 & 01 & 01 & 02 \end{pmatrix} \times \begin{pmatrix} c_0 \\ c_1 \\ c_2 \\ c_3 \end{pmatrix}$$

InvMixColumns beschreibt die inverse Operation zu MixColumns. Wir können nun aus

$$g(x)g^{-1}(x) = 01 \bmod x^4 + 1$$

das Polynom

$$g^{-1}(x) = 0Bx^3 + 0Dx^2 + 09x + 0E,$$

wie bereits im ersten Kapitel gezeigt, bilden. InvMixColumns lässt sich nun durch

$$c(x) = g^{-1}(x)\, a(x) \bmod x^4 + 1$$

berechnen. Die entsprechende Matrixschreibweise dazu ist:

$$\begin{pmatrix} c_0 \\ c_1 \\ c_2 \\ c_3 \end{pmatrix} = \begin{pmatrix} 0E & 0B & 0D & 09 \\ 09 & 0E & 0B & 0D \\ 0D & 09 & 0E & 0B \\ 0B & 0D & 09 & 0E \end{pmatrix} \times \begin{pmatrix} a_0 \\ a_1 \\ a_2 \\ a_3 \end{pmatrix}$$

Die Wahl der Koeffizienten des Polynoms $c(x)$ sorgt dafür, dass MixColumns sehr viel schneller als InvMixColumns berechnet werden kann. Durch den Koeffizienten 01 muss keine Multiplikation durchgeführt werden. Bei 02 kann die Multiplikation durch einen zyklischen Shift um ein Bit nach links repräsentiert werden. Der Koeffizient 03 lässt sich durch einen zyklischen Shift um ein Bit nach links und eine zusätzliche XOR-Operation mit dem ursprünglichen Byte repräsentieren.

3.1.4 AddRoundKey

AddRoundKey führt ein bitweises XOR eines Bits der Zustandsmatrix mit einem Bit der Matrix des Rundenschlüssels durch. Die Matrix des Rundenschlüssels hat dabei die gleiche Größe wie die Zustandsmatrix. Abbildung 3.5 verdeutlicht diesen Zusammenhang.

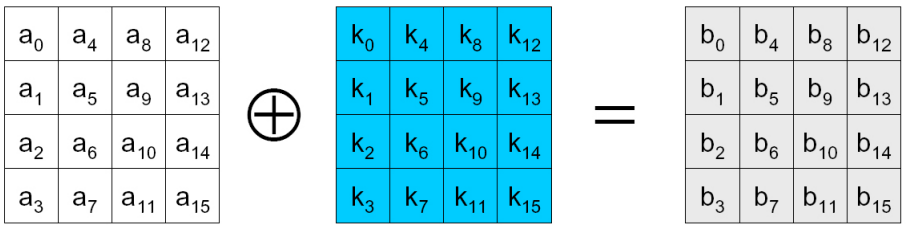


Abbildung 3.5: Die Operationen AddRoundKey

Der Rundenschlüssel wird dabei aus dem Schlüssel mit Hilfe der Funktion ExpandedKey erzeugt. Die genaue Vorgehensweise erläutern wir im nächsten Abschnitt. Da die XOR-Funktion zu sich selbst invers ist, ist auch die Operation AddRoundKey zu sich selbst invers.

3.2 Der Schlüsselplan

In diesem Abschnitt beschreiben wir, wie aus dem Schlüssel die einzelnen Rundenschlüssel abgeleitet werden. Wir benötigen genau $N_r + 1$ Rundenschlüssel, da wir in jeder Runde

einen Rundenschlüssel verwenden und wir bereits vor der ersten Runde eine Schlüsseladdition durchführen. Zunächst wird der Schlüssel expandiert. Dazu wird ein erweiterter Schlüssel W mit 4 Zeilen und $N_b(N_r+1)$ Spalten erzeugt. Aus W werden nun die einzelnen Rundenschlüssel abgeleitet, so dass sich der i-te Rundenschlüssel durch die Spalten iN_b bis $(i+1)N_b - 1$ ergibt. Diese Spalten bilden eine Matrix, welche die gleiche Größe wie die Zustandsmatrix eines Textes besitzt.

Zur Initialisierung von W werden die ersten N_k Spalten mit dem Schlüssel belegt. Alle weiteren Spalten berechnen sich als Kombinationen zweier bereits bekannter Spalten. Dabei unterscheiden sich die Initialisierungsalgorithmen für $N_k > 6$ und $N_k \leq 6$.

Wir beschreiben im Folgenden die Algorithmen zur Erzeugung der Rundenschlüssel bei einer Schlüssellänge von 128, 192 und 256 Bit und einer Textblocklänge von 128 Bit. Algorithmus 3.2.1 beschreibt die Erzeugung des erweiterten Schlüssels. Die folgenden Algorithmen sind aus [35, S.228-229] entnommen.

Algorithmus 3.2.1 GetRoundKeys128

Eingabe: $M = M_0||M_1||M_2||M_3$ mit M_i ist eine Spalte der Größe 32 Bit des Schlüssels

Ausgabe: erweiterter Schlüssel $W_0||\cdots||W_{43}$

SCHRITT 1: **Für** $i = 0$ **bis** 3

$W_i = M_i$

SCHRITT 2: **Für** $j = 4$ **bis** 40

Falls $j \equiv 0 \bmod 4$

$W_j = W_{j-4} \oplus SubBytes(RotByte(W_{j-1})) \oplus Rcon(\frac{j}{4})$

Für $i = 1$ **bis** 3

$W_{i+j} = W_{i+j-4} \oplus W_{i+j-1}$

Mit *SubBytes* wird hierbei die bereits bekannte Rijndael S-Box benutzt. *RotByte* führt einen zyklischen Shift auf den Bytes einer Spalte W_i durch. Besteht eine Spalte W_i aus den Bytes $W_{i0}, W_{i1}, W_{i2}, W_{i3}$, so ergibt sich:

$$RotByte(W_i) = RotByte(W_{i0}, W_{i1}, W_{i2}, W_{i3}) = (W_{i1}, W_{i2}, W_{i3}, W_{i0})$$

Mit anderen Worten, heißt dies, das in einer Spalte alle Bytes um eine Position nach oben zyklisch geshiftet werden, welches dazu führt, dass das erste Byte an die unterste Position gerät. *Rcon* ist eine Funktion, die fest vorgegebene Rundenkonstanten zurück gibt. Eine genaue Darstellung der Rundenkonstanten ist im Anhang A.3 zu finden.

Wird ein Schüssel der Länge 192 Bit verwendet, so benötigen wir $12+1$ Rundenschlüssel. Bei einer Schlüssellänge von 256 Bit sind es $14+1$. Die $+1$ kommt durch das initiale KeyAdd zustande. Die Erzeugung dieser Rundenschlüssel ist in den beiden folgenden Algorithmen 3.2.2 und 3.2.3 dargestellt.

Algorithmus 3.2.2 GetRoundKeys192

Eingabe: $M = M_0||M_1||M_2||M_3||M_4||M_5$ mit M_i ist eine Spalte der Größe 32 Bit des Schlüssels

Ausgabe: erweiterter Schlüssel $W_0||\cdots||W_{51}$

SCHRITT 1: **Für** $i = 0$ **bis** 5

$W_i = M_i$

SCHRITT 2: **Für** $j = 6$ **bis** 48

Falls $j \equiv 0 \bmod 6$

$W_j = W_{j-6} \oplus SubBytes(RotByte(W_{j-1})) \oplus Rcon(\frac{j}{6})$

Für $i = 1$ **bis** 5 und **solange** $i + j < 52$

$W_{i+j} = W_{i+j-6} \oplus W_{i+j-1}$

Algorithmus 3.2.3 GetRoundKeys256

Eingabe: $M = M_0||M_1||M_2||M_3||M_4||M_5||M_6||M_7$ mit M_i ist eine Spalte der Größe 32 Bit des Schlüssels

Ausgabe: erweiterter Schlüssel $W_0||\cdots||W_{59}$

SCHRITT 1: **Für** $i = 0$ **bis** 7

$W_i = M_i$

SCHRITT 2: **Für** $j = 8$ **bis** 56

Falls $j \equiv 0 \bmod 8$

$W_j = W_{j-8} \oplus SubBytes(RotByte(W_{j-1})) \oplus Rcon(\frac{j}{8})$

Für $i = 1$ **bis** 3

$W_{i+j} = W_{i+j-8} \oplus W_{i+j-1}$

$W_{j+4} = W_{j-4} \oplus SubBytes(W_{j+1})$

Für $i = 5$ **bis** 7

$W_{i+j} = W_{i+j-b} \oplus W_{i+j-1}$

Die Rundenschlüssel werden nun aus dem erweiterten Schlüssel nach dem in Abbildung 3.6 exemplarisch für 128 Bit Schlüssel dargestellten Schema errechnet. Dabei besteht der initiale Rundenschlüssel K_0 aus den Spalten W_0, W_1, W_2, W_3 des erweiterten Schlüssels; der erste Rundenschlüssel K_1 aus den Spalten W_4, W_5, W_6, W_7 usw. bis alle Rundenschlüssel erzeugt sind. Siehe dazu [31, S.29-31], [35] und [15, S.43-44].

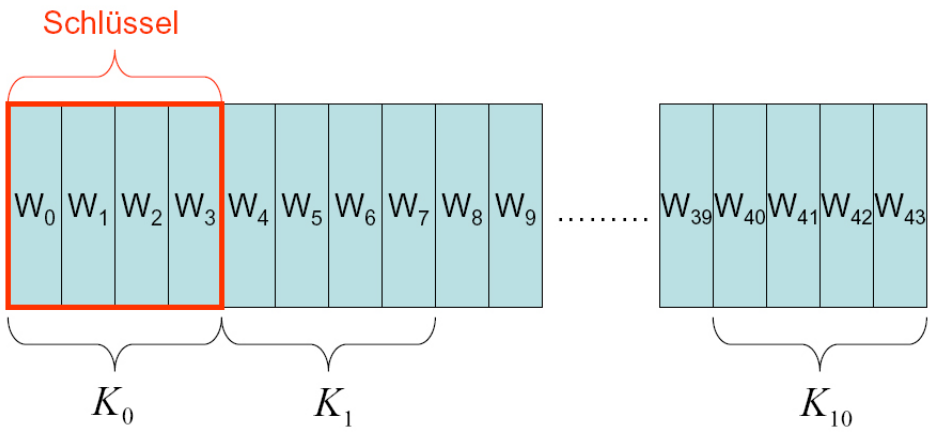


Abbildung 3.6: Erzeugung der Rundenschlüssel für 128 Bit Schlüssel

Die Erzeugung der Rundenschlüssel weist nicht so ein hohes Maß an Konfussion und Diffusion, wie es Shannon in [40] fordert, auf. Dieses Problem wird von May, Henricksen und Millan in [35] im diskutiert. Es werden dabei drei Kriterien für Schlüsselpläne mit hoher Sicherheit ausgearbeitet.

1. Kollisionsresistente Ein-Weg-Funktionen (d.h. Funktionen, die nicht invertierbar sind)
2. Minimale gegenseitige Information zwischen allen Bits der Rundenschlüssel und allen Bits des Generalschlüssels
3. effiziente Implementierung

Der Schlüsselplan von Rijndael weist ein hohes Maß an Nichtlinearität auf. Verglichen mit der Chiffre selbst ist er jedoch deutlich weniger nicht linear, da die S-Box im Schlüsselplan nicht so häufig zur Anwendung kommt, wie in der Chiffre. Dieses begünstigt den Erfolg von Related-key-Angriffen. Es wird deutlich, dass Rijndael nur die dritte Bedingung erfüllt und somit Angriffe über den Schlüsselplan ermöglicht. Bis heute existiert jedoch noch kein Angriff, der AES über alle Runden vollständig bricht. Kennt ein Angreifer Teile bestimmter Rundenschlüssel, so kann er dies nutzen, um auf Bits anderer Rundenschlüssel zu schließen. May, Henricksen und Millan legen in [35] verbesserte Algorithmen zur Erzeugung der Rundenschlüssel dar, mit deren Hilfe alle drei Kriterien erfüllt werden können.

3.3 Entschlüsselung

Die Entschlüsselung eines Chiffretextes lässt sich durch die verwendete Struktur der Chiffre sehr einfach beschreiben. Es werden dabei die Schritte der Verschlüsselung in umgekehrter Reihenfolge durchlaufen. Zu beachten ist allerdings, dass die Rundenschlüssel in umgekehrter Reihenfolge in die Entschlüsselung eingehen. Dieses verdeutlicht Abbildung 3.7.

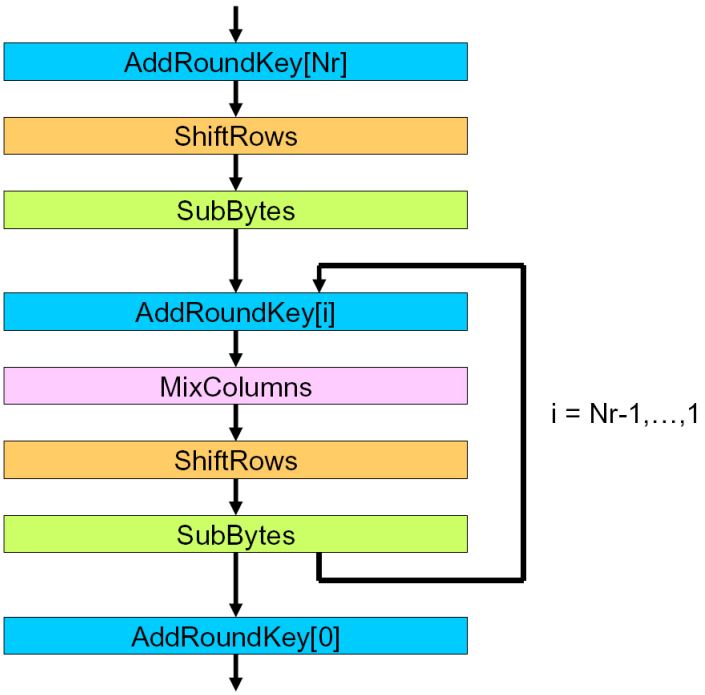


Abbildung 3.7: Der Ablauf der Rijndael-Chiffre (Entschlüsselung)

Kapitel 4

Kryptoanalyse

In diesem Teil der Arbeit betrachten wir verschiedene Angriffe auf Blockchiffren. In [41] und in [37] wird eine Übersicht der verschiedensten Angriffe auf Blockchiffren gegeben, aus der nun einige Auszüge ansprechen werden.

4.1 Differentielle Kryptoanalyse

Differentielle Kryptoanalyse ist ein sehr mächtigen Angriff auf Blockchiffren, welcher von Biham und Shamir in [8] und [9] entwickelt wurde, um Blockchiffren anzugreifen. Diese wurde speziell für Angriffe auf den Data Encryption Standard (DES) entwickelt. Sie lässt sich aber auch auf beliebige andere Blockchiffren anwenden. Wir beschreiben die differentielle Kryptoanalyse speziell für Byte-orientierte Verfahren, wie AES. Die Beschreibungen dazu vollziehen wir in Anlehnung an [23, S.19-29], [41, S.5-8], [45, S.19] und [37, S.11-13]. Eine grundlegende Einführung der differentiellen Kryptoanalyse ist wichtig, da die folgenden Angriffe auf dieser aufbauen. Die differentielle Kryptoanalyse ist ein Angriff mit gewählten Klartexten deren Ziel es ist, einzelne Bits des Rundenschlüssels der letzten Runde zu berechnen. Da die S-Box die einzige nicht lineare Komponente eines SPN's darstellt, wird nun besonders auf die Analyse der S-Box Ein- und Ausgaben Wert gelegt.

Es sei ein 16 Byte Eingabeblock durch $X = [X_1, X_2, \ldots, X_{16}]$ und ein 16 Byte Ausgabeblock durch $Y = [Y_1, Y_2, \ldots, Y_{16}]$ beschrieben. Ein solcher Block entspricht einem Zustand von AES. Wir führen zunächst ein paar grundlegende Begriffe ein.

Definition 4.1.1 *Mit Differenz bezeichnen wir das Bit-weise XOR zweier 16-Byte Texte. Aus X' und X'' berechnet sich eine Differenz ΔX durch:*

$$\Delta X = [X'_1 \oplus X''_1, X'_2 \oplus X''_2, \ldots, X'_{16} \oplus X''_{16}]$$

Dabei berechnet sich das Byte-weise XOR $(X'_i \oplus X''_i)$ als Bit-weises XOR der einzelnen Bits. Das Symbol Δ steht im Folgenden immer für Differenz.

Definition 4.1.2 *Ein Differential einer S-Box ist eine Zuordnung der Differenz einer Ausgabe ΔY zu einer Differenz einer Eingabe ΔX mit einer Wahrscheinlichkeit p. Wir schreiben:*

$$\Delta X \xrightarrow{p} \Delta Y$$

In Worten: Eine Eingabedifferenz ΔX wird mit der Wahrscheinlichkeit p zu einer Ausgabedifferenz ΔY durch das Differential $\Delta X \rightarrow \Delta Y$.

Annahme 4.1.3 *Für alle Texte mit Differenz ΔX, welche das Differential $\Delta X \rightarrow \Delta Y$ durchlaufen, ist die Wahrscheinlichkeit $\Pr(\Delta X \rightarrow \Delta Y)$ gleich.*

Die Wahrscheinlichkeit, dass eine Differenz ΔY auf eine Differenz ΔX folgt, ist für eine zufällige Chiffre genau 2^{-n}, wobei n die Blocklänge darstellt. In der differentiellen Kryptoanalyse wird nun versucht Differentiale zu finden, deren Wahrscheinlichkeit p deutlich größer als 2^{-n} ist.

Annahme 4.1.4 *Ein Differential hat für die Ver- und Entschlüsselung die gleiche Wahrscheinlichkeit.*

Definition 4.1.5 *Hat ein Differential für die Verschlüsselung, eine Wahrscheinlichkeit p, d.h.*

$$\Delta X \xrightarrow{p} \Delta Y,$$

dann folgt mit Annahme 4.1.4 für die Entschlüsselung, dieselbe Wahrscheinlichkeit. Wir schreiben:

$$\Delta X \xleftarrow{p} \Delta Y$$

Definition 4.1.6 *Eine Folge von Differentialen*

$$\Delta_1 X_1 \xrightarrow{p_1} \Delta_2 X_1 \xrightarrow{p_2} \cdots \Delta_r X_1$$

wird als Differential-Charakteristik oder kurz Charakteristik bezeichnet.

Die Differenzen der Ausgabe einer Runde stellen die Differenzen der Eingabe der folgenden Runde dar. Um leichter mit den Charakteristiken rechnen zu können, treffen wir die folgende Annahme.

Annahme 4.1.7 *Die Wahrscheinlichkeiten der Differentiale in einer Charakteristik sind unabhängig voneinander.*

Nach Annahme 4.1.7 können wir die Wahrscheinlichkeit p_{DC} einer Charakteristik als

$$p_{DC} = \prod_{i=1}^{r-1} p_i$$

berechnen, wobei p_i die Wahrscheinlichkeit eines Differentials darstellt.

Definition 4.1.8 *Als passives Byte bezeichnen wir die Differenz zweier Bytes, welche der Null-Differenz entspricht. Alle anderen Werte einer Differenz werden als aktives Byte bezeichnet.*

Beispiel 4.1.9 *Seien zwei 4 Byte Texte in Hexadezimalschreibweise gegeben mit $X' = FF\ 01\ 02\ 03$ und $X'' = FA\ 03\ 02\ 03$, so erhalten wir eine Differenz:*

$$\Delta X = 05\ 02\ 00\ 00$$

Damit sind die ersten zwei Bytes der Differenz ΔX aktiv und die letzten zwei passiv.

Nach den einführenden Definitionen können wir die differentielle Kryptoanalyse wie folgt beschreiben.

1. Wähle eine Klartextdifferenz ΔX.
2. Finde eine Charakteristik mit hoher Wahrscheinlichkeit über die ersten $r-1$ Runden, deren Ausgabedifferenz mit ΔD_C bezeichnet wird.
3. Verschlüssele alle Klartextpaare (X', X'') welche eine Differenz ΔX zu den Chiffretextpaaren (Y', Y'') aufweisen.
4. Initialisiere einen Zähler für jede Kombination von Schlüsselbits der r-ten Runde mit Null.
5. Gehe über alle möglichen Schlüsselbits der aktiven Bytes in der r-ten Runde.
   5.1. Gehe über alle Chiffretextpaare (Y', Y'').
      5.1.1. Entschlüssele das betrachtete Chiffretextpaar (Y', Y'') eine Runde mit den in Schritt 5 gewählten Schlüsselbits und nenne die erhaltene Differenz ΔC.
      5.1.2. Falls $\Delta D_C = \Delta C$, so erhöhe den Zähler auf die verwendeten Schlüsselbits um eins.
6. Gib die Schlüsselbits als korrekt aus, deren Zähler den größten Wert besitzt.

Algorithmus 1: Differentielle Kryptoanalyse

Wir wählen im **Schritt 1** eine beliebige Differenz ΔX, bei der nicht alle Bytes passiv sein dürfen. Es wird nun versucht eine Differential-Charakteristik mit möglichst großer Wahrscheinlichkeit in **Schritt 2** zu finden. Dazu benötigt man zunächst die Wahrscheinlichkeiten aller möglichen Differentiale der verwendeten S-Box.

Definition 4.1.10 *Eine Differential-Verteilungstabelle listet zu jeder Eingabedifferenz die Häufigkeiten der möglichen Ausgabedifferenzen einer S-Box auf.*

Eine Differential-Verteilungstabelle kann für eine bekannte S-Box leicht berechnet werden, indem zu allen Eingabedifferenzen alle möglichen Ausgabedifferenzen errechnet werden. Aus den Häufigkeiten können anschließend die Wahrscheinlichkeiten der Differentiale berechnet werden. Da die Auflistung einer Verteilungstabelle für eine AES S-Box mit 8 Bit Ein- und Ausgabewerten, also 256 Spalten und Zeile zu umfangreich wäre, bringen wir an dieser Stelle ein reduziertes Beispiel, welches wir aus [23, S.21] entnommen haben. Tabelle 4.1 zeigt eine Verteilungstabelle für eine S-Box deren Ein- und Ausgabewerte der Größe 4-Bit entsprechen. Auf die Beschreibung der verwendeten S-Box verzichten wir an dieser Stelle. Das folgende Beispiel orientiert sich an der in Tabelle 4.1 abgebildeten Verteilungstabelle.

Beispiel 4.1.11 *Für dieses Beispiel verwenden wir erneut die Hexadezimalschreibweise. Nehmen wir die Differenzen $\Delta X = B$ und $\Delta Y = 2$, so hat das Differential $\Delta X \xrightarrow{p} \Delta Y$ eine Wahrscheinlichkeit von $p = \frac{8}{16}$. Da es 16 mögliche Werte für ΔY gibt, aber bei $\Delta X = B$ nur acht mal $\Delta Y = 2$ vorkommt, ergibt sich die genannte Wahrscheinlichkeit.*

Es sollte klar sein, dass bei $\Delta X = 0$ und $\Delta Y = 0$ der größte Wert in der Tabelle auftritt, da gleiche Eingabewerte auch gleiche Ausgabewerte nach der S-Box besitzen.

Definition 4.1.12 *Eine ideale S-Box besitzt für jede Kombination aus Ein- und Ausgabedifferenzen den Häufigkeitswert 1.*

Nur mit einer idealen S-Box wären alle Differentiale einer S-Box gleich wahrscheinlich. Ein SPN mit solchen S-Boxen wäre resistent gegen differentielle Kryptoanalyse. Allerdings ist eine ideale S-Box mathematisch unmöglich, welches sich über die Eigenschaften der Differential-Verteilungstabelle begründen lässt. Wir finden hierfür zwei Begründungen.

	Ausgabe															
Eingabe	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
4	0	0	0	2	0	0	6	0	0	2	0	4	2	0	0	0
5	0	4	0	0	0	2	2	0	0	0	4	0	2	0	0	2
6	0	0	0	4	0	4	0	0	0	0	0	0	2	2	2	2
7	0	0	2	2	2	0	2	0	0	2	2	0	0	0	0	4
8	0	0	0	0	0	0	2	2	0	0	0	4	0	4	2	2
9	0	2	0	0	2	0	0	4	2	0	2	2	2	0	0	0
A	0	2	2	0	0	0	0	0	6	0	0	2	0	0	4	0
B	0	0	8	0	0	2	0	2	0	0	0	0	0	2	0	2
C	0	2	0	0	2	2	2	0	0	0	0	2	0	6	0	0
D	0	4	0	0	0	0	0	4	2	0	2	0	2	0	2	0
E	0	0	2	4	2	0	0	0	6	0	0	0	0	0	2	0
F	0	2	0	0	6	0	0	0	0	4	0	2	0	0	2	0

Tabelle 4.1: Differential-Verteilungstabelle (Vgl.[23, S.21].)

1) Die Summe aller Spalten muss 2^n ergeben, wobei n der Länge der Eingaben in Bits entspricht. Dies gilt ebenso für die Summe aller Zeilen. Auf eine Differenz der Eingabe von Null folgt auch eine Differenz der Ausgabe von Null. Damit muss im ersten Feld der Verteilungstabelle der größte Wert stehen und in der entsprechenden ersten Zeile und Spalte an allen anderen Positionen eine Null.

2) Alle Einträge müssen gerade sein, da ein Paar (X', X'') die gleiche Differenz wie ein Paar (X'', X') besitzt.

Durch die Verwendung der Differenzen spielen die Werte der Rundenschlüssel keine Rolle bei der Berechnung der Differentiale. Um dies zu verdeutlichen, schauen wir auf die Auswirkung der Rundenschlüssel auf die Ein- und Ausgaben einer S-Box. Die Rundenschlüssel entsprechen nun ebenfalls einem 16 Byte Zustand der Form $K = K_1, K_2, \ldots, K_{16}$. Wenn wir davon ausgehen, dass für zwei verschiedene Eingaben X' und X'' die Rundenschlüssel gleich sind, erhalten wir:

$$\begin{aligned} \Delta X &= [(X'_1 \oplus K_1) \oplus (X''_1 \oplus K_1), \ldots, (X'_{16} \oplus K_{16}) \oplus (X''_{16} \oplus K_{16})] \\ &= [X'_1 \oplus K_1 \oplus X''_1 \oplus K_1, \ldots, X'_{16} \oplus K_{16} \oplus X''_{16} \oplus K_{16}] \\ \Delta X &= [X'_1 \oplus X''_1, \ldots, X'_{16} \oplus X''_{16}] \end{aligned}$$

Wir sehen, dass die Rundenschlüssel keinen Einfluss auf die Eingabedifferenz haben. Auch die Differential-Verteilungstabelle bleibt in diesem Fall gleich.

Hat die Differenz zweier Texte ein passives Byte, so schreiben wir an die entsprechende Position eine 0. Um die Wahrscheinlichkeit einer Charakteristik berechnen zu können, müssen wir noch die folgende Annahme treffen:

Annahme 4.1.13 *Die Ein- und Ausgabedifferenz einer S-Box ist unabhängig von denen anderer S-Boxen.*

AES arbeitet auf Zuständen der Größe 16 Bytes. Die verwendete S-Box hat 1 Byte Ein- und Ausgabewerte. Die S-Box wird somit auf jedes der 16 Bytes eines Zustandes in einer Runde parallel angewendet. Annahme 4.1.13 sagt nun aus, dass die Ausgabedifferenzen dieser S-Boxen unabhängig voneinander sind. Die Wahrscheinlichkeit eines Differentials berechnet sich damit als

$$p_D = \prod_{i=1}^{16} \beta_i.$$

Hierbei ist β_i die Wahrscheinlichkeit, dass eine bestimmte 1 Byte Differenz zu einer bestimmten 1 Byte Differenz durch die S-Box transformiert wird. Diese Wahrscheinlichkeit ergibt sich, wie zuvor am Beispiel dargestellt, aus der Differential-Verteilungstabelle der S-Box. Nach Annahme 4.1.7 auf Seite 20 wissen wir, dass auch die Wahrscheinlichkeiten der Differentiale in einer Charakteristik unabhängig voneinander sind. Damit können wir die Wahrscheinlichkeit einer Charakteristik durch

$$p_{DC} = \prod_{i=1}^{r-1} \prod_{j=1}^{16} \beta_{ij}$$

errechnen. Hierzu nun ein kurzes schematisches Beispiel.

Beispiel 4.1.14 *Sei eine 16 Byte Differenz gegeben, in der nur die ersten beiden Bytes aktiv sind.*

$$\Delta_0 X = \Delta_0 X_1, \Delta_0 X_2, 0, \cdots, 0$$

Wir suchen eine 3*-Runden-Charakteristik mit hoher Wahrscheinlichkeit, wobei die Differenzen mit der Runde indiziert werden. Um ein Differential mit hoher Wahrscheinlichkeit zu finden, suchen wir in der Verteilungstabelle mögliche Ausgabedifferenzen von* $\Delta_0 X_1$ *und* $\Delta_0 X_2$*, welche eine möglichst hohe Wahrscheinlichkeit aufweisen. Die Wahrscheinlichkeit des Differentials berechnet sich aus dem Produkt der beiden Wahrscheinlichkeiten der 1-Byte-Differentiale. Nehmen wir an, dass wir nach der ersten Runde die Differenz*

$$\Delta_1 X = \Delta_1 X_1, \Delta_1 X_2, 0, \cdots, 0, \Delta_1 X_{16}$$

erhalten haben, so suchen wir erneut die Differentiale mit der größten Wahrscheinlichkeit. Nach der dritten Runde haben wir eine Charakteristik, welche aus drei 16-Byte-Differentialen besteht:

$$\Delta_0 \xrightarrow{p_1} \Delta_1 \xrightarrow{p_2} \Delta_2 \xrightarrow{p_3} \Delta_3$$

Die Wahrscheinlichkeit der Charakteristik berechnet sich nun durch $p_1 \cdot p_2 \cdot p_3$*.*

Als nächstes werden (**Schritt 3**) alle Textpaare (X', X''), welche die geforderte Differenz ΔX erfüllen, zu einem Chiffretextpaar (Y', Y'') verschlüsselt. Wie viele solcher Textpaare wir für einen erfolgreichen Angriff mindestens wählen müssen, werden wir später noch genauer betrachten. Die erhaltene Chiffretext-Differenz besteht nun aus aktiven und passiven Bytes. Von Interesse sind jedoch nur die aktiven Bytes, da wir deren Schlüsselbits berechnen wollen. Wir ordnen nun jeder Kombination von Schlüsselbits der r-ten Runde einen Zähler zu und initialisieren diesen mit Null (**Schritt 4**). Wir betrachten dabei jedoch nicht alle Schlüsselbits, sondern nur die, an den Positionen aktiver Bytes.

Im **Schritt 5** lassen wir eine Schleife über alle möglichen Kombinationen von Schlüsselbits der aktiven Bytes der r-ten Runde laufen. Eine weitere Schleife lassen wir über alle Chiffretextpaare laufen (**Schritt 5.1**). Nun entschlüsseln wir (**Schritt 5.1.1**) das betrachtete Chiffretextpaar mit den gewählten Schlüsselbits genau eine Runde. Anschließend vergleichen wir (**Schritt 5.1.2**), ob die erhaltene Differenz der Differenz der Charakteristik aus Schritt 2 entspricht. Ist dies der Fall, so erhöhen wir den Zähler für die verwendeten Schlüsselbits um eins. Die Schlüsselbits mit dem größten Zähler werden nun am Ende der Schleife in **Schritt 6** als korrekt ausgegeben.

Man kann feststellen, dass die Wahrscheinlichkeit einer Charakteristik um so größer ist, je weniger aktive Bytes vorhanden sind und desto größer die Wahrscheinlichkeiten der einzelnen Differentiale sind. Um die differentielle Kryptoanalyse erfolgreich durchführen zu können, benötigt man etwa

$$N = \frac{c}{p_{DC}} \tag{4.1}$$

Klartextpaare, wobei c eine kleine Konstante und p_{DC} die Wahrscheinlichkeit einer Charakteristik für $r-1$ Runden ist. Die exakte Anzahl von Klartextpaaren ist schwer zu bestimmen, weshalb man diese nur grob abschätzt. Die richtigen Schlüsselbits werden signifikant öfter gezählt als Falsche. Wir erwarten, dass ein Klartextpaar, welches den Zähler auf die richtigen Schlüsselbits erhöht, alle p_{DC}^{-1} Klartextpaare auftritt. Generell ist es empfehlenswert, den Angriff mit einem Vielfachen von p_{DC}^{-1} durchzuführen, damit der Angriff erfolgreich verläuft. Die korrekten Schlüsselbits werden somit signifikant öfter als falsche gezählt, welches zu einer deutlichen Unterscheidung führt.

Um eine Chiffre vor differentieller Kryptoanalyse zu schützen, sollten die Wahrscheinlichkeiten der Charakteristiken möglichst gering sein und eine große Anzahl von aktiven Bytes auftreten. Dazu müssen die Differential-Verteilungstabellen möglichst eine gleichmäßige Verteilung von kleinen Einträgen aufweisen. Wie oben bereits gezeigt, ist die Gestaltung einer idealen S-Box nicht möglich. Darüber hinaus führt auch eine Erhöhung der Runden zu einer Verringerung der Wahrscheinlichkeit der Charakteristik.

Daemen und Rijmen zeigen in [15], dass es für die Rijndael-Chiffre keine 4-Runden Differential-Charakteristiken mit einer Wahrscheinlichkeit größer als 2^{-150} und keine 8-Runden Charakteristiken mit einer Wahrscheinlichkeit größer als 2^{-300} gibt. Rijndael und AES weisen nach den eben genannten Kriterien eine Resistenz gegen differentielle Kryptoanalyse auf.

In den folgenden Abschnitten stellen wir Angriffe vor, welche effizienter als die differentielle Kryptoanalyse zur erfolgreichen Schlüsselsuche eingesetzt werden können. Dabei befassen wir uns zunächst mit sogenannten Unterscheiderangriffen, welche die Texte auf eine bestimmte Art und Weise filtern. Diese Angriffe werden anschließend dazu benutzt, bestimmte Bits des ersten oder letzten Rundenschlüssels zu errechnen. Auf den Boomerang- und Amplified Boomerang-Angriff werden wir sehr ausführlich eingehen. Da sich die anschließenden Angriffe nur in einem sehr geringen Maße von diesen abheben, werden wir lediglich auf dessen Besonderheiten genauer eingehen.

4.2 Der Boomerang - Angriff

"When you send it properly, it always comes back to you." [43]

Der Boomerang-Angriff ist ein auf der differentiellen Kryptoanalyse aufbauender Angriff. Er wurde von Wagner in [43] entwickelt. Wir entnehmen den Angriff aus den Beschreibungen von [26, S78-79], [41, S.13-14], [4, S.2-3], [6, S.507-510] und [37, S.16-17]. Es handelt sich hierbei um einen Angriff mit adaptiv gewählten Klar- und Chiffretexten. Der Boomerang-Angriff erlaubt es, Chiffren anzugreifen, die resistent gegen differentielle Kryptoanalyse sind. Dies liegt daran, dass nun nicht mehr eine lange Charakteristik mit hoher Wahrscheinlichkeit über viele Runden, sondern zwei kurze über weniger Runden mit hoher Wahrscheinlichkeit gesucht werden. Solche kurzen Charakteristiken lassen sich viel leichter finden. Bei der differentiellen Kryptoanalyse wird angenommen, dass man nach Gleichung (4.1) mindestens p_{DC}^{-1} Klartexte benötigt, um eine Chiffre zu brechen. Es stellt sich heraus, dass es mit Hilfe des Boomerang-Angriffs möglich ist, dies mit wesentlich weniger Klartext erreichen zu können. Wenn die beste Charakteristik über die Hälfte der Runden einer Chiffre die Wahrscheinlichkeit r besitzt, benötigen wir $O(r^{-4})$ gewählte Texte, um den Boomerang-Angriff erfolgreich durchführen zu können. Gilt $r^{-4} \ll p_{DC}^{-1}$, so benötigt der Boomerang-Angriff weniger Texte als die differentielle Kryptoanalyse. Wir verwenden im Folgenden nur noch den Begriff Differential, da für uns nur die Ein- und Ausgabedifferenz sowie die gesamte Wahrscheinlichkeit über einer Teilchiffre von Bedeutung ist.

Annahme 4.2.1 *Wir nehmen an, dass sich die betrachtete Chiffre in zwei Teilchiffren E_0 und E_1 aufteilen lässt, welche unabhängig voneinander sind, so dass*

$$E(x) = E_1(E_0(x))$$

gilt. Es existiert ein Differential $\alpha \xrightarrow{p} \beta$ mit der Wahrscheinlichkeit p für E_0 und ein Differential $\gamma \xrightarrow{q} \delta$ für E_1 mit der Wahrscheinlichkeit q.

Wenn wir von der Entschlüsselung einer Teilchiffre sprechen, so schreiben wir E_0^{-1} oder E_1^{-1}. Wenn wir davon sprechen, dass eine Differenz α zu einer Differenz β wird, so handelt es sich dabei immer um die Verschlüsselung mit E_0. Lassen wir eine Differenz β in eine Differenz α umwandeln, so handelt es sich um die Entschlüsselung mit E_0^{-1}. Nach Definition 4.1.5 auf Seite 20 schreiben wir $\alpha \xrightarrow{p} \beta$ um zu verdeutlichen, dass eine Differenz α mit der Wahrscheinlichkeit p zu einer Differenz β nach der Verschlüsselung mit E_0 wird. Außerdem schreiben wir $\alpha \xleftarrow{p} \beta$, um auszudrücken, dass eine Differenz α nach der Entschlüsselung einer Differenz β durch E_0^{-1} mit der Wahrscheinlichkeit p entsteht. Natürlich verschlüsseln wir keine Differenzen, sondern die Klartexte, welche die Differenzen bilden. Dasselbe gilt für γ und δ, entsprechend für E_1. Annahme 4.1.4 auf Seite 20 impliziert, dass

$$\begin{aligned} \Pr(\alpha \rightarrow \beta) &= \Pr(\alpha \leftarrow \beta) \\ \Pr(\gamma \rightarrow \delta) &= \Pr(\gamma \leftarrow \delta) \end{aligned}$$

gilt.

Der Ablauf des Boomerang-Angriffs ist in Abbildung 4.1 dargestellt. Dabei symbolisieren die Vierecke jeweils die Teilchiffren E_0 und E_1. Die Pfeile zeigen die Richtung an, d.h. geht ein Pfeil von oben nach unten, so handelt es sich um eine Verschlüsselung. In umgekehrter Richtung sprechen wir von einer Entschlüsselung. P, P', O und O' sind Klartexte. A, A' sind Texte, die aus den Klartexten P, P' durch Verschlüsselung mit E_0 entstehen, wobei B, B' aus der Entschlüsselung der Chiffretexte D, D' mit E_1^{-1} entstehen. Die Verbindungslinien zwischen jeweils zwei Texten sollen die auftretenden Differenzen verdeutlichen.

Ein grober Ablauf des Angriffs ist folgender. Zwei Klartexte P und P' mit $P \oplus P' = \alpha$ werden mit E_0 verschlüsselt. Es entstehen die Texte A und A', welche mit einer gewissen Wahrscheinlichkeit eine Differenz β aufweisen. Die anschließende Verschlüsselung mit E_1 macht aus den Texten A und A' die Chiffretexte C und C'. Zu diesen werden zwei neue Chiffretexte mit Abstand δ generiert. Diese Chiffretexte D, D' mit $C \oplus D = \delta = C' \oplus D'$ werden mit E_1^{-1} zu B und B' entschlüsselt. Da das Differential $\gamma \leftarrow \delta$ durch E_1^{-1} läuft, haben sowohl A und B als auch A' und B' mit einer gewissen Wahrscheinlichkeit eine Differenz γ. Da A und A' mit einer gewissen Wahrscheinlichkeit eine Differenz β besitzen, gilt diese auch für B und B' mit einer gewissen Wahrscheinlichkeit. Nach der Entschlüsselung von B und B' mit E_0^{-1} haben die entstandenen Klartexte O und O' eine Differenz von α mit einer gewissen Wahrscheinlichkeit. An dieser Stelle soll darauf hingewiesen werden, dass ein Angreifer keinesfalls die Ver- und Entschlüsselungen mit E_0 oder E_1 berechnen kann. Er kann damit auch nicht die Texte A, A', B und B' errechnen. Die Formulierungen hierzu dienen nur zur Verdeutlichung des gedanklichen Ablaufs im betrachteten Angriff. Der Angriff verläuft nun nach den folgenden Schritten:

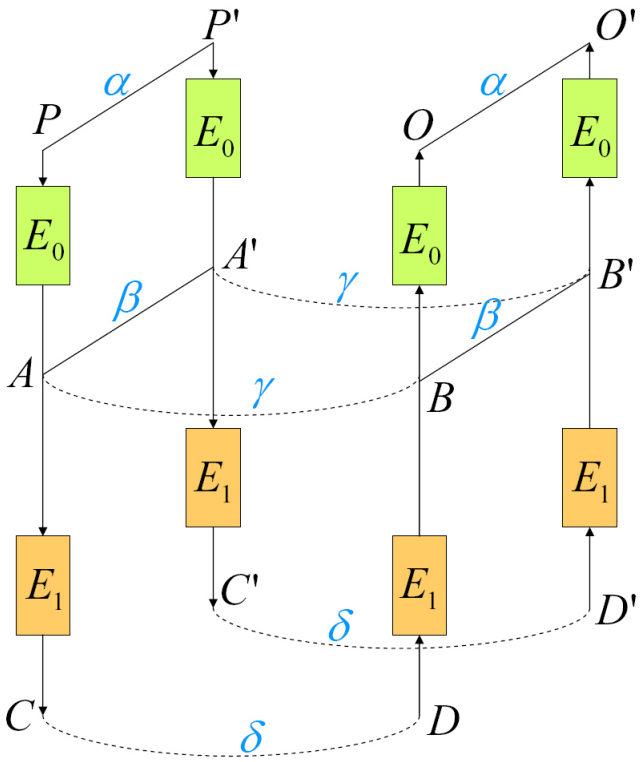


Abbildung 4.1: Darstellung des Boomerang-Angriffs

1. Zwei Klartexte P und P' mit $P \oplus P' = \alpha$ seien gegeben.
2. Lasse P und P' zu C und C' verschlüsseln.
3. Setze $D = C \oplus \delta$ und $D' = C' \oplus \delta$.
4. Lasse D und D' mit $E^{-1} = E_0^{-1}(E_1^{-1}(X))$ zu O und O' entschlüsseln.
5. Prüfe, ob $O \oplus O' = \alpha$ und speichere alle Quartette (P, P', O, O'), die diese Bedingung erfüllen.

Algorithmus 2: Boomerang-Angriff als Unterscheiderangriff

Um den Ablauf des Angriffs zu verstehen, betrachten wir die Wahrscheinlichkeit, dass im **Schritt 5** ein passendes Paar gefunden wird. Offensichtlich wird eine zufällige Permutation die Bedingung in Schritt 5 mit der Wahrscheinlichkeit 2^{-n} erfüllen, wobei n die Blocklänge darstellt. Es werden nur die Quartette gespeichert, welche die Bedingung in diesem Schritt erfüllen. Die Wahrscheinlichkeit, dass ein Paar (P, P') das Differential $\alpha \xrightarrow{p} \beta$ erfüllt, beträgt p. Nach der Erzeugung von D und D' in **Schritt 3** beträgt die Wahrscheinlichkeit, dass (C, D) und (C', D') das Differential $\gamma \xleftarrow{q} \delta$ erfüllen, q^2. Nun gilt $B \oplus B' = \beta$ mit der Wahrscheinlichkeit pq^2, denn zunächst gilt:

$$\begin{aligned} E_0(O) \oplus E_0(O') &= (E_0(P) \oplus E_0(P')) \oplus (E_0(P) \oplus E_0(O)) \oplus (E_0(P') \oplus E_0(O')) \\ &= (E_0(P) \oplus E_0(P')) \oplus (E_1^{-1}(C) \oplus E_1^{-1}(D)) \oplus (E_1^{-1}(C') \oplus E_1^{-1}(D')) \\ &= (A \oplus A') \oplus (A \oplus B)) \oplus (A' \oplus B') \end{aligned}$$

Weiter wissen wir, dass mit der Wahrscheinlichkeit pq^2

$$\beta \oplus \gamma \oplus \gamma = \beta = B \oplus B'$$

gilt. Mit der Wahrscheinlichkeit p erfüllt $B \oplus B' = \beta$ das Differential $\alpha \xleftarrow{p} \beta$. Insgesamt haben wir mit Wahrscheinlichkeit $(pq)^2$ ein Quartett (P, P', O, O') gefunden, welches die

Bedingung in Schritt 5 erfüllt.

Definition 4.2.2 *Wir bezeichnen ein Quartett, welches den Differentialen von E_0 und E_1 genügt und somit die Bedingung in Schritt 5 erfüllt als* korrektes Boomerang-Quartett. *Ein Quartett, welches nur die Bedingungen 5 erfüllt und nicht den Differentialen genügt wird als* falsches Boomerang-Quartett *bezeichnet.*

Je größer der Quotient

$$\frac{pq}{2^{-n/2}}$$

ist, desto anfälliger ist die betrachtete Chiffre für den Boomerang-Angriff. Dieser Ausdruck ist ein Maß dafür, wie weit die Chiffre von einer zufälligen Permutation entfernt ist. Der hier vorgestellte Boomerang-Angriff dient lediglich als Unterscheiderangriff. Ein solcher Angriff kann nur eine Unterscheidung der Menge aller möglichen Quartette in zwei Teilmengen durchführen. Dabei besteht eine erste Menge aus allen korrekten und falschen Boomerang-Quartetten und eine zweite Menge aus allen übrigen Quartetten. Es ist nicht möglich aus der ersten Menge die korrekten Boomerang-Quartette von den falschen Boomerang-Quartetten zu separieren. Ein Unterscheider-Angriff erlaubt lediglich eine Abschätzung über die relativen Häufigkeiten der korrekten Boomerang-Quartette innerhalb der ersten Menge.

Der Leser fragt sich nun berechtigt, was diesen mit der Schlüsselsuche in Verbindung bringt. Nehmen wir noch einmal den Ablauf der differentiellen Kryptoanalyse auf Seite 21 zur Hand, um die Veränderung dieser zum Boomerang-Angriff verdeutlichen zu können. Anstatt ein langes Differential haben wir es beim Boomerang-Angriff mit zwei kurzen zu tun. Wir können den Boomerang-Angriff nun dazu verwenden, am Anfang oder am Ende der Chiffre Teile der Rundenschlüssel zu berechnen. Wir beschränken uns hier auf die Variante, welche Teile des initialen Rundenschlüssels errechnet. Diese wird auch im Kapitel 5 bei der Anwendung auf AES vollzogen. In Kapitel 6 zeigen wir bei den von uns erweiterten Angriffen, wie man Teile der Rundenschlüssel auf beiden Seiten der Chiffre erhält. Wir werden die Verwendung des Boomerang-Angriffs zur Schlüsselsuche hier nur grob beschreiben.

1. Wähle eine Klartextdifferenz α.
2. Benutze den Boomerang-Unterscheiderangriff, um die Menge der korrekten und falschen Boomerang-Quartette zu erhalten. Alle übrigen Quartette werden verworfen.
3. Initialisiere einen Zähler für jede Kombination von Schlüsselbits des initialen Rundenschlüssels.
5. Gehe über alle möglichen Kombination von Schlüsselbits der aktiven Bytes in der 0-ten Runde.
   - 5.1. Gehe über alle Quartette (P_i, P_j, O_i, O_j), die nach der Filterung in Schritt 2 übrig geblieben sind.
     - 5.1.1. Lasse die Paare P, P' und O, O' eines Quartetts mit den gewählten Schlüsselbits eine Runde verschlüsseln und nenne die erhaltenen Differenzen ΔC_1 und ΔC_2.
     - 5.1.2. Falls ΔC_1 und ΔC_2 einer bestimmten Differenz entsprechen, so erhöhe den Zähler auf die verwendeten Schlüsselbits um eins.
6. Gib die Schlüsselbits als korrekt aus, deren Zähler den größten Wert besitzt.

Algorithmus 3: Boomerang-Angriff (Unterscheiderangriff mit Schlüsselsuche)

Es sei an dieser Stelle darauf hingewiesen, dass die sehr unscharfe Formulierung des Schrittes 5.1.2 durchaus beabsichtigt ist. Die bestimmte Differenz, über die gesprochen

wird, hängt von der zu analysierenden Chiffre ab. Es ist damit eine Differenz zwischen zwei Texten gemeint, die nur erhalten wird, wenn die korrekten Schlüsselbits bei einem korrekten Boomerang-Quartett verwendet werden. In den Fällen korrekter Schlüsselbits mit falschem Boomerang-Quartett oder falscher Schlüsselbits mit beliebigem Quartett tritt die gesuchte Differenz nur mit einer sehr geringen Wahrscheinlichkeit auf. Dadurch genügen wenige korrekte Boomerang-Quartette unter den falschen Boomerang-Quartetten, um die korrekten Schlüsselbits signifikant öfter zu zählen als falsche. Die Unterteilung der Chiffre in E_0 und E_1 dient nur der Abschätzung der relativen Häufigkeiten von korrekten Boomerang-Quartetten. Der Angreifer ist nicht in der Lage diese Teilchiffren zu verwenden, wie bereits in diesem Abschnitt erwähnt. Würden wir die Schlüsselsuche über die gesamte Teilchiffre E_0 laufen lassen, so müssten wir zu viele Bits raten, wenn E_0 mehr als eine Runde enthält.

Eine große Schwäche des Boomerang-Angriffs ist, dass er adaptiv gewählte Chiffretexte benötigt. Wir werden sehen, wie dieses Problem gelöst werden kann. Um Chiffren resistent gegen den Boomerang-Angriff zu machen, sollten Differentiale auch für eine reduzierte Anzahl von Runden möglichst eine geringe Wahrscheinlichkeit aufweisen. Die in den folgenden Abschnitten erläuterten Amplified Boomerang- und Rechtecks-Angriffe sind ebenfalls Unterscheiderangriffe und lassen sich ähnlich, wie der hier beschriebene Boomerang-Angriff für die Schlüsselsuche verwenden. Darauf gehen wir allerdings in Kapitel 5 genauer ein.

4.3 Der Amplified Boomerang - Angriff

Der Amplified Boomerang-Angriff stellt eine Erweiterung des Boomerang-Angriffs des vorherigen Kapitels dar und wurde von Kelsey, Kohno und Schneier in [26] vorgestellt. Wir entnehmen die Beschreibung dazu aus [41, S.14], [26, S.79-83], [6, S.509-511], [4, S.2-3], [28, S.245-247] und [3, S.7-9]. Dies ist ein Angriff mit gewählten Klartexten. Im Gegensatz zum einfachen Boomerang-Angriff [43] werden nun keine adaptiv gewählten Chiffretexte mehr benötigt. Wir verwenden erneut die Notation des vorherigen Kapitels. Abbildung 4.2 zeigt den Angriff schematisch; sie unterscheidet sich nur geringfügig von der Abbildung 4.1. Lediglich die Richtung der Pfeile auf der rechten Seite der Abbildung haben sich geändert. Die Annahmen des Boomerang-Angriffs sind weiterhin gültig.

Bei diesem Angriff werden Quartette gebildet, die aus zwei Paaren P, P' und O, O' mit $P \oplus P' = \alpha = O \oplus O'$ bestehen. Nach der Verschlüsselung eines solchen Quartetts mit E_0 erhalten wir mit einer gewissen Wahrscheinlichkeit die Differenz β zwischen den Paaren A, A' und B, B'. Mit einer gewissen Wahrscheinlichkeit stimmen auch die Differenzen von A, B und A', B' überein. Nachdem das Quartett (A, A', B, B') mit E_1 zu (C, C', D, D') verschlüsselt wurde, haben die Paare C, D und C', D' mit einer gewissen Wahrscheinlichkeit die Differenz δ. Der Angriff lässt sich wie folgt beschreiben:

1. Seien (P, P', O, O') Quartette von Klartexten mit $P \oplus P' = \alpha$ und $O \oplus O' = \alpha$.
2. Verschlüssle (P, P', O, O') zu (C, C', D, D').
3. Prüfe, ob $C \oplus D = \delta$ und $C' \oplus D' = \delta$ gilt und speichere alle Quartette (P, P', O, O'), welche diese Bedingung erfüllen.

Algorithmus 4: Amplified Boomerang-Angriff als Unterscheiderangriff

Wie im vorherigen Kapitel nehmen wir an, dass für E_0 ein Differential $\alpha \xrightarrow{p} \beta$ mit Wahrscheinlichkeit p und für E_1 ein Differential $\gamma \xrightarrow{q} \delta$ mit Wahrscheinlichkeit q existiert.

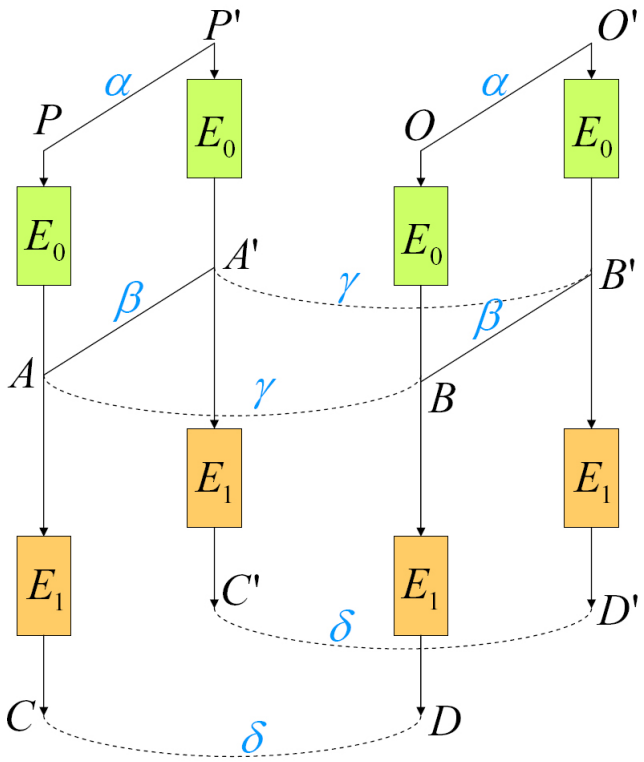


Abbildung 4.2: Darstellung des Amplified Boomerang-Angriffs

Definition 4.3.1 *Wir bezeichnen ein Quartett, welches den Differentialen von E_0 und E_1 genügt und die Bedingung in Schritt 1 und 3 erfüllt, als korrektes Amplified Boomerang-Quartett. Ein Quartett, welches nur die Bedingungen 1 und 3 erfüllt ist ein falsches Amplified Boomerang-Quartett.*

Nun werden wir untersuchen, wie hoch die Wahrscheinlichkeit ist, dass ein Quartett (P, P', O, O') von Klartexten die Bedingung in **Schritt 3** erfüllt. Ebenso wollen wir wissen, wie viele dieser korrekten Quartette wir aus m Paaren mit Differenz α erwarten können.

Haben m Paare (P, P') eine Differenz α, so haben erwartete mp Paare nach der Verschlüsselung mit E_0 eine Differenz von β, da das Differential $\alpha \xrightarrow{p} \beta$ die Wahrscheinlichkeit p besitzt. Damit können wir aus diesen Paaren im Erwartungswert

$$\frac{(mp)^2}{2}$$

Quartette (P, P', O, O') bilden, welche das Differential $\alpha \xrightarrow{p} \beta$ erfüllen.

Annahme 4.3.2 *Die Ausgabe der Verschlüsselung von E_0 sei zufällig über alle möglichen Werte verteilt.*

Nach Annahme 4.3.2 erhalten wir mit der Wahrscheinlichkeit 2^{-n} ein Textpaar (A, B) mit Differenz γ und somit ein Textpaar (A', B') mit $A' \oplus B' = \gamma$ aus:

$$A' \oplus B' \quad = \quad (A \oplus A') \oplus (B \oplus B') \oplus (A \oplus B)$$

Das Textpaar (A, B') hat nach der getroffenen Annahme 4.3.2 ebenso mit Wahrscheinlichkeit 2^{-n} eine Differenz von γ. Damit besitzt ein Textpaar (A', B) eine Differenz von γ. Die Paare (A, A') und (B, B') können somit auf zwei verschiedene Arten mit Wahrscheinlichkeit $2\frac{1}{2^n} = 2^{-n+1}$ als Quartette verwendet werden, wobei eine γ Differenz jeweils zwischen zwei Textpaaren auftritt. Wir erwarteten dadurch

$$\frac{(mp)^2}{2}\, 2^{-n+1}$$

Quartette, welche die eben genannten Eigenschaften besitzen. Da die Wahrscheinlichkeit des Differentials für E_1 q beträgt, erwarten wir

$$m^2\ 2^{-n}(pq)^2$$

korrekte Amplified Boomerang-Quartette, welche alle Differentiale erfüllen. Hiermit können wir mit $m = 1$ folgern, dass die Wahrscheinlichkeit für ein korrektes Amplified Boomerang-Quartett

$$2^{-n}(pq)^2$$

beträgt. Eine zufällige Permutation liefert im Erwartungswert

$$2 \cdot \frac{m^2}{2} \cdot 2^{-n} \cdot 2^{-n} = \frac{m^2}{2} \cdot 2^{-2n+1} = m^2 \cdot 2^{-2n}$$

korrekte Amplified Boomerang-Quartette. Auch hier gilt, je größer

$$\frac{pq}{2^{-n/2}}$$

ist, desto mehr unterscheidet sich die betrachtete Chiffre von einer zufälligen Permutation. Dies sagt aus, wie anfällig die Chiffre für den Angriff ist. Bei diesem Angriff werden zwar die adaptiv gewählten Chiffretexte vermieden, dafür werden im Vergleich zum einfachen Boomerang-Angriff mehr Klartexte für einen erfolgreichen Angriff benötigt, da die Wahrscheinlichkeit eines korrekten Quartetts von $(pq)^2$ auf $2^{-n}(pq)^2$ gesunken ist. Der Grund dafür ist, dass man nicht garantieren kann, dass die Differenz γ in der Mitte der Chiffre tatsächlich erhalten wird. Es müssen also entsprechend viele Quartette untersucht werden, um die gewünschte Differenz in der Mitte der Chiffre erhalten zu können. Die Schlüsselsuche verläuft ähnlich zu der im vorherigen Abschnitt beschriebenen Schlüsselsuche beim Boomerang-Angriff, mit dem Unterschied, dass bei diesem Angriff Schlüsselbits des letzten Rundenschlüssels gesucht werden.

4.4 Der Rechtecks - Angriff

Der Rechtecks-Angriff wurde von Biham, Dunkelman und Keller entwickelt und in [3] vorgestellt. Er stellt eine Erweiterung des im vorherigen Kapitel betrachteten Amplified Boomerang-Angriffs dar. Die Erläuterungen sind aus [20, S.4-5], [5, S.27-30], [6, S.509-511], [4, S.1-9] und [3, S.1-10] entnommen. Da der Ablauf dem des Amplified Boomerang-Angriffs gleicht, werden wir diesen nicht noch einmal beschreiben. Stattdessen können wir uns sofort dessen Verbesserungen zuwenden. Die Idee hierbei ist es, die Differenzen in der Mitte der Chiffre nicht genau zu spezifizieren. Dies führt dazu, dass wir aus m Paaren mit Differenz α mehr Paare erhalten, welche die Bedingung in Schritt 3 des Amplified Boomerang-Angriffs erfüllen. Eine Darstellung des Rechtecks-Angriffs befindet sich in Abbildung 4.3. Hierbei werden Quartette (P, P', O, O') mit Differenz α in den Paaren P, P' und O, O' gesucht. Diese werden durch E_0 und E_1 verschlüsselt. Aus den erhaltenen Chiffretexten C, C', D, D' wird geprüft, ob die Chiffretextpaare C, D und C', D' eine Differenz δ aufweisen.

Wir verwenden nicht mehr das Differential $\alpha \xrightarrow{p} \beta$ und $\gamma \xrightarrow{q} \delta$, sondern alle Differentiale $\alpha \to \beta'$ und $\gamma' \to \delta$ für alle möglichen Werte von β' und γ' mit $\beta' \neq \gamma'$. Dies erhöht gleichzeitig die Wahrscheinlichkeit passender Differentiale, da wir nun anstatt eines Differentials mit hoher Wahrscheinlichkeit viele mit geringen Wahrscheinlichkeiten benutzen können. Sei $\Pr(\alpha \to \beta)^2 = p^2$ die Wahrscheinlichkeit dafür, dass die Paare P, P' und O, O' das Differential $\alpha \to \beta$ erfüllen. Mit anderen Worten: die Paare P, P' und O, O' mit

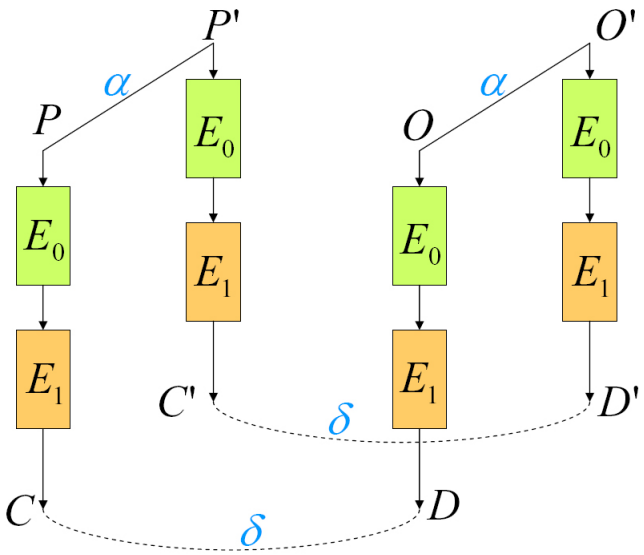


Abbildung 4.3: Darstellung des Rechtecks-Angriffs

Differenz α besitzen nach der Verschlüsselung mit $\alpha \rightarrow \beta$ mit der Wahrscheinlichkeit p^2 eine Differenz β. Wir ersetzen nun diese Wahrscheinlichkeit durch

$$\sum_{\substack{\beta' \\ \alpha \rightarrow \beta'}} \Pr(\alpha \rightarrow \beta')^2,$$

um die Wahrscheinlichkeiten aller möglichen Differentiale für E_0 mit fester Eingabedifferenz α und beliebiger Ausgabedifferenz zu summieren. Eine ähnliche Modifikation können wir für die Differentiale über E_1 durchführen. Zwei Paare A, B und A', B' mit einer Differenz γ (d.h. $A \oplus B = \gamma$ und $A' \oplus B' = \gamma$) besitzen mit der Wahrscheinlichkeit $\Pr(\gamma \rightarrow \delta)^2 = q^2$ nach der Verschlüsselung mit $\gamma \rightarrow \delta$ eine Differenz δ. Diese Wahrscheinlichkeit lässt sich durch

$$\sum_{\substack{\gamma' \\ \gamma' \rightarrow \delta}} \Pr(\gamma' \rightarrow \delta)^2$$

ersetzen. Diese summiert nun die Wahrscheinlichkeiten aller Differentiale für E_1 mit beliebiger Eingabedifferenz und fester Ausgabedifferenz δ. Sei $\hat{p}$ die Wahrscheinlichkeit für das Auftreten der Differentiale $\alpha \rightarrow \beta'$ für alle β' und sei $\hat{q}$ die Wahrscheinlichkeit der Differentiale $\gamma' \rightarrow \delta$ für alle Werte von γ'. Dann ergibt sich:

$$\begin{aligned} \hat{p} &= \sqrt{\sum_{\substack{\beta' \\ \alpha \rightarrow \beta'}} \Pr(\alpha \rightarrow \beta')^2} \\ \hat{q} &= \sqrt{\sum_{\substack{\gamma' \\ \gamma' \rightarrow \delta}} \Pr(\gamma' \rightarrow \delta)^2} \end{aligned}$$

Dementsprechend ist $\hat{p}^2$ die Wahrscheinlichkeit dafür, dass ein Quartett das Differential $\alpha \rightarrow \beta'$ für alle β' erfüllt. Dies gilt analog für $\hat{q}^2$. Die Wahrscheinlichkeit, dass ein Quartett die gewünschten Eigenschaften besitzt und somit als korrekt gilt, ist durch

$$2^{-n/2}\ \hat{p}\hat{q}$$

gegeben. Aus m Paaren mit Differenz α lassen sich nun erwartete

$$2\ m^2\ 2^{-(n+1)}\ (\hat{p}\hat{q})^2 = m^2\ 2^{-n}\ (\hat{p}\hat{q})^2$$

korrekte Rechtecks-Quartette bilden. Der Rechtecks-Angriff benötigt weniger Klartexte als der Amplified Boomerang-Angriff, um einen Angriff erfolgreich durchführen zu können. Es ist allerdings schwierig, alle möglichen Differentiale zu zählen. Wir haben uns an dieser Stelle bewusst auf die Beschreibung der Unterschiede zum Amplified Boomerang-Angriff beschränkt, da der Angriff ansonsten gleich verläuft.

4.5 Related-Key Angriffe

Angriffe mittels Related-Keys wurden von Biham in [1] entwickelt. Knudsen beschrieb in [29] ebenfalls Angriffe dieser Art. Solche Angriffe lassen sich, wie wir in den folgenden Abschnitten sehen werden, mit Varianten der differentiellen Kryptoanalyse kombinieren. Wir entnehmen die Beschreibung der Angriffe aus [1, S.1-10], [37, 21-23], [15, S.157] und [41, S.16-18]. Man zielt bei diesen Angriffen auf Schwächen im Schlüsselplan der Chiffre ab. Bei den meisten Chiffren, wie auch bei AES, beinhaltet der Schlüsselplan ein hohes Maß an Linearität und bietet sich dadurch für Angriffe dieser Art an.

Die einzelnen Rundenschlüssel einer Chiffre werden für gewöhnlich aus einem Schlüssel unter Benutzung eines Schlüsselerzeugungsalgorithmus hergestellt gewonnen. Wenn dieser Algorithmus in allen Runden gleich ist, so können von Differenzen des Schlüssel Informationen über Differenzen der Rundenschlüssel abgeleitet werden.

Definition 4.5.1 *Als Related-Keys bezeichnen wir die Beziehung zwischen zwei Schlüsseln, welche durch eine dem Angreifer bekannte Funktion ausgedrückt werden kann.*

Wir unterscheiden zwei Formen von Angriffen mittels Related-Keys, wobei der Angreifer niemals die Schlüssel selber, sondern nur die Beziehung zwischen ihnen wählt oder kennt.

- Angriff mit gewählter Relation und gewählten Klartexten: Hierbei wählt der Angreifer sowohl die Beziehung zwischen den Schlüsseln als auch zwischen den Klartexten.
- Angriff mit bekannter Relation und gewählten Klartexten: Hierbei kennt der Angreifer die Beziehung zwischen den Schlüsseln, kann sie aber nicht selbst wählen. Er wählt allerdings die Beziehung zwischen den Klartexten.

Um eine Chiffre sicher gegen Related-Key-Angriffe zu machen, muss der Algorithmus zur Generierung von Rundenschlüsseln ein hohes Maß an Nichtlinearität aufweisen. Außerdem sollte bei der Erzeugung der Rundenschlüssel ein hoher Grad an Diffusion in den Rundenschlüsseln auftreten. Lucks diskutiert in [34] die Sicherheitsaspekte von Related-Key-Angriffen.

Der Schlüsselplan der Rijndael-Chiffre weist ein relativ geringes Maß an Nichtlinearität im Vergleich zur Chiffre selbst auf. Damit eignen sich Related-Key-Angriffe, um die Schwächen des Schlüsselplans auszunutzen. Ein Related-Key-Angriff auf neun Runden AES-256 wurde von Ferguson, Kelsey, Schneier, Stay, Wagner und Whiting in [22] vorgestellt. Dieser Angriff ist auf AES mit 256 Schlüssel-Bits ausgelegt und benötigt 2^{72} gewählte Klartexte und 2^{227} Verschlüsselungen, womit er deutlich schneller als eine vollständige Suche ist.

4.5.1 Related-Key Differentielle Kryptoanalyse

Wir werden zunächst eine einfache Variante von Related-Key-Angriffen, welche auf der differentiellen Kryptoanalyse basieren, beschreiben. Diese ist aus [25, S.208-211] entnommen. Sei P_1 fixierter und P_2 ein zufällig gewählter Klartext mit $P_1 \oplus P_2 = \alpha$ und K_1 sowie K_2 zwei Schlüssel mit $K_1 \oplus K_2 = \Delta K$. Ein solcher Angriff auf eine r-Runden Blockchiffre verläuft wie folgt:

1. Wähle eine Eingabedifferenz α, in der nicht alle Bytes passiv sind.
2. Suche ein Related-Key-Differential über $r-1$ Runden, welches eine hohe Wahrscheinlichkeit p besitzt und nenne dessen Ausgabedifferenz β.
3. Verschlüssele alle Klartexte P_i und P_j mit Differenz α mit K_1 und K_2:
$$E_{K_1}(P_i) = C_i, \quad E_{K_2}(P_i) = C_j,$$
um die Chiffretexte C_i und C_j zu erhalten.
4. Initialisiere einen Zähler für alle möglichen Paare von Schlüsseln der r-ten Runde mit Null.
5. Für alle möglichen Paare von Schlüsseln der r-ten Runde (K_1^r, K_2^r)
   5.1. Für alle Chiffretextpaare (C_i, C_j)
      5.1.1. Berechne
      $$d_{K_1^r}(C_i) \oplus d_{K_2^r}(C_j) = \Delta C,$$
      wobei $d_{K_i^r}(C_i)$ die Entschlüsselung der letzten Runde beschreibt.
      5.1.2. Falls $\beta = \Delta C$, erhöhe den Zähler auf das verwendete Schlüsselpaar um eins.
6. Gib das Schlüsselpaar als korrekt aus, welches den größten Zähler aufweist.

Algorithmus 5: Related-Key Differentielle Kryptoanalyse

Damit der Angriff erfolgreich ist, muss sich die Chiffre von einer zufälligen Permutation unterscheiden. Es muss für die Wahrscheinlichkeit p des $r-1$ Runden Differentials
$$p > 2^{-n}$$
gelten, wobei n die Blocklänge der Chiffre ist. Die Anzahl der Durchführungen der Schritte 2 und 3 beträgt:
$$N = \frac{c}{p},$$
wobei c eine kleine Konstante ist. Auch hier ist die Begründung ähnlich der der differentiellen Kryptoanalyse. Wir können die exakte Anzahl an benötigten Texten nicht zählen. Wir erwarten einen erfolgreichen Versuch alle $\frac{1}{p}$ Paare, d.h. ein Erhöhen des Zählers in Schritt 5.1.2. Die Anzahl der Paare, die zu einem erfolgreichen Angriff nötig sind, sollten einem kleinen Vielfachen c von $\frac{1}{p}$ entsprechen. In den folgenden Abschnitten beschreiben wir Related-Key Boomerang- und Rechtecks-Angriffe, die eine Erweiterung der bereits bekannten Angriffe um Related-Keys darstellen. Dabei handelt es sich erneut lediglich um Unterscheiderangriffe.

4.5.2 Related-Key Boomerang-Angriff

Der Related-Key Boomerang-Angriff stellt eine Kombination aus einem Related-Key und dem Boomerang-Angriff dar und ist aus [25, S.208-211], [27, S.123-127], [24, S.368-372] und [6, S.507-512] entnommen. Von zwei Schlüsseln K, K' sei ΔK das bitweise XOR der einzelnen Bits. Die Schlüssel sind damit durch $K = \Delta K \oplus K'$ miteinander verbunden. Die Verschlüsselung eines Klartextes erfolgt wie zuvor durch
$$E_K = E_{1,K}(E_{0,K}(X)).$$
Jedoch werden unterschiedliche Klartexte mit unterschiedlichen Schlüsseln verschlüsselt. Dabei ist nur die Differenz der Schlüssel bekannt, nicht die Schlüssel selbst. Wir nehmen

an, dass für E_0 eine Charakteristik $\alpha \xrightarrow{p} \beta$ mit Wahrscheinlichkeit p existiert, bei der die Schlüssel eine Differenz von ΔK_0 aufweisen. Für E_1 existiert eine Charakteristik $\gamma \xrightarrow{q} \delta$ mit Wahrscheinlichkeit q unter der Differenz ΔK_1 der Schlüssel. Der Angriff ist schema-

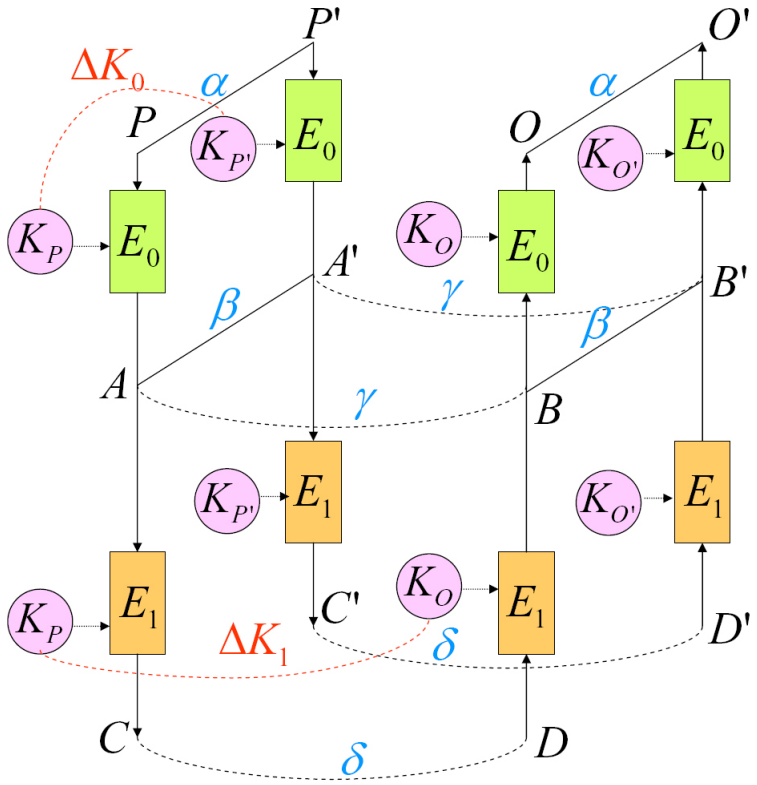


Abbildung 4.4: Darstellung des Related-Key Boomerang-Angriffs (Vgl.[6, S.512].)

tisch in Abbildung 4.4 dargestellt. Die Bezeichnungen K_a, K_b, K_c, K_d in Abbildung 4.4 sollen verdeutlichen, für welche Verschlüsselung welche Schlüssel verwendet werden. In die Teilchiffren E_0 und E_1 gehen dabei aber nur die entsprechenden Rundenschlüssel ein. Der Angriff lässt sich wie folgt durchführen:

1. Seien P und P' zwei Klartexte mit $P \oplus P' = \alpha$.
2. Verschlüssele P mit E_{K_P} zu C und P' mit $E_{K_{P'}}$ zu C'.
3. Setze $D = C \oplus \delta$ und $D' = C' \oplus \delta$.
4. Entschlüssele D mit $E^{-1}_{K_O}$ zu O und D' mit $E^{-1}_{K_{O'}}$ zu O'.
5. Prüfe, ob $O \oplus O' = \alpha$ und speichere alle Quartette (P, P', O, O'), die diese Bedingung erfüllen.

Algorithmus 6: Related-Key Boomerang-Angriff als Unterscheiderangriff

Wie aus der Abbildung 4.4 ersichtlich ist, bestehen zwischen den vier verwendeten Schlüsseln folgende Beziehungen, die dem Angreifer bekannt sind:

$$\begin{aligned} \Delta K_0 &= K_P \oplus K_{P'}, \\ \Delta K_1 &= K_P \oplus K_O, \\ K_{O'} &= K_P \oplus \Delta K_0 \oplus \Delta K_1 \end{aligned}$$

Die Wahrscheinlichkeit, dass ein Paar P, P' die Bedingung in Schritt 5 erfüllt, ist genau wie im einfachen Boomerang-Angriff p^2q^2. Auch hier erfüllt eine zufällige Permutation

mit Wahrscheinlichkeit 2^{-n} diese Bedingung. Der Ablauf gleicht dem des Boomerang-Angriffs aus Abschnitt 4.2 auf Seite 24. Der Unterschied besteht darin, dass im Related-Key Boomerang-Angriff mehr als ein Schlüssel verwendet wird und diese in bestimmten Relationen zueinander stehen. Ob der Related-Key Boomerang-Angriff besser ist, hängt von der Struktur der zu analysierenden Chiffre und deren Schlüsselplan ab. Wir werden auf diese Aspekte im Kapitel 6 genauer eingehen.

4.5.3 Related-Key Rechtecks-Angriff

Auch bei dem Related-Key Rechtecks-Angriff handelt es sich um eine Kombination von einem Related-Key und dem Rechtecks-Angriff, welche in Abbildung 4.5 dargestellt ist. Entnommen wurde sie aus [25, S.208-211], [27, S.123-127], [24, S.368-372] sowie [6, S.507-513]. Der Angriff lässt sich wie folgt beschreiben:

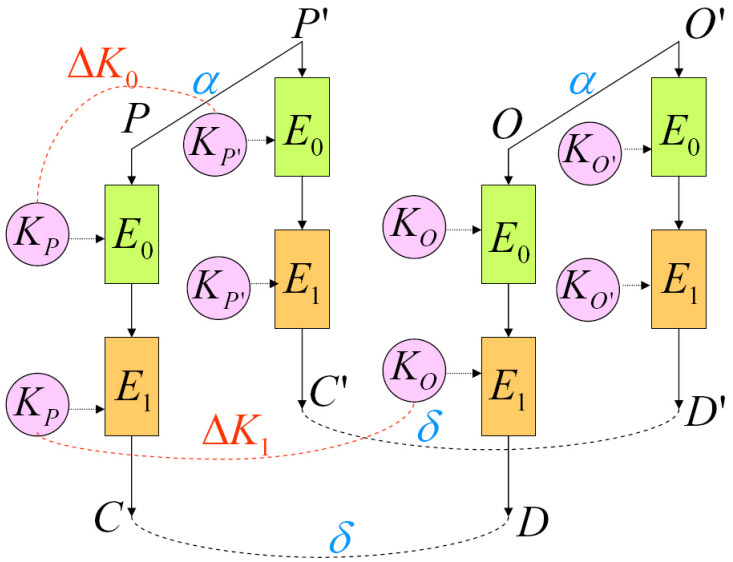


Abbildung 4.5: Darstellung des Related-Key Rechtecks-Angriffs

1. Seien (P, P', O, O') Klartexte mit $P \oplus P' = \alpha$ und $O \oplus O' = \alpha$.
2. Lasse die Klartexte P, P', O, O' mit den Schlüsseln $K_P, K_{P'}, K_O, K_{O'}$ zu (C, C', D, D') verschlüsseln.
3. Prüfe, ob $C \oplus D = \delta$ und $C' \oplus D' = \delta$ und speichere alle Quartette (P, P', O, O'), die diese Bedingung erfüllen.

Algorithmus 7: Related-Key Rechtecks-Angriff als Unterscheiderangriff

Die Beziehungen zwischen den Schlüsseln entsprechen denen aus dem vorherigen Kapitel. Die Analyse des Angriffs ist ähnlich zu der des Rechtecks-Angriffs. Haben m Paare eine Differenz von α und werden diese jeweils mit verschiedenen Schlüsseln verschlüsselt, so erwarten wir

$$m^2\ 2^{-n}\ (\hat{p}\hat{q})^2$$

Quartette, welche die Bedingung in Schritt 3 erfüllen. Dabei sind die Wahrscheinlichkeiten $\hat{p}$ und $\hat{q}$ dem Rechtecks-Angriff entnommen. Auch hier gilt, wie schon zuvor im Related-Key Boomerang-Angriff, dass der Ablauf dem des Rechtecks-Angriff gleicht. Es werden hierbei allerdings mehr als ein Schlüssel verwendet. Der Vorteil dieses Angriffs gegenüber dem Rechtecks-Angriff hängt von der verwendeten Chiffre und der Gestalt dessen Schlüsselplans ab. Im nächsten Kapitel werden wir auf diesen Aspekt genauer eingehen.

Kapitel 5

Anwendung der Angriffe auf AES

5.1 Boomerang-Angriff auf 5 Runden AES-128

Der im Folgenden erläuterte Angriff wurde von Biryukov in [10] beschrieben und basiert auf einer reduzierten Variante von AES mit einer Schlüssellänge von 128 Bits. Hierbei wird der Boomerang-Angriff aus Abschnitt 4.2 eingesetzt, um Teile des Rundenschlüssels der ersten Runde zu bestimmen. Wir entnehmen diesen Angriff und weitere Erläuterungen aus [11], [43], [17, S.1-6], [13], [26, S.88-92] und [19, S.44-53,73-79]. Wir beschreiben zunächst den Angriff in einer einfacheren Variante, um die grundlegenden Konzepte zu verdeutlichen. Anschließend werden wir eine verbesserte Variante dieses Angriffs von Biryukov [10] beschreiben.

Die Chiffre wird in zwei Teilchiffren E_0 und E_1 zerlegt, welche jeweils einem Differential zugeordnet sind. Die Verschlüsselung eines Klartextes x lässt sich als $E(x) = E_1(E_0(x))$ darstellen. Für die Analyse der hier vorgestellten Angriffe auf AES und auch für die Erweiterungen im nächsten Kapitel benötigen wir die folgende wichtige Annahme.

Annahme 5.1.1 *Die S-Box verhält sich wie eine zufällige Permutation.*

Neben der Annahme 5.1.1 gelten weiterhin die Annahmen der vorherigen Kapitel.

Bei diesem Angriff beinhaltet E_0 die Runden $1-3$ und E_1 die Runden 4 und 5. Der Chiffre E_0 wird das Differential $\alpha \to \beta$ und der Chiffre E_1 das Differential $\gamma \to \delta$ zugeordnet. Wie schon im vorherigen Kapitel bedeutet $\alpha \to \beta$ immer die Verschlüsselung mit E_0 und $\alpha \leftarrow \beta$ die Entschlüsselung mit E_0^{-1}. Dies gilt auch hier analog für γ, δ des Differentials für E_1. Vor der ersten Runde kommt ein zusätzliches AddRoundKey

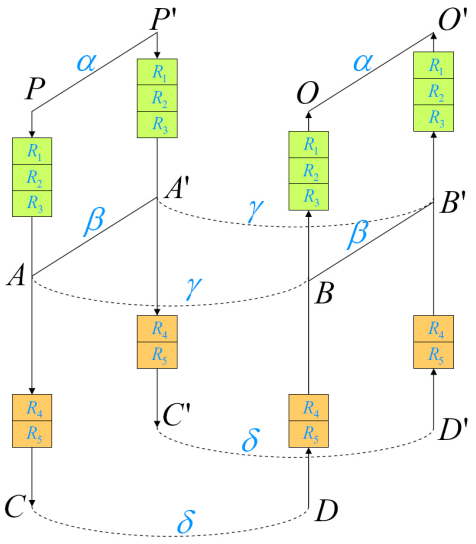


Abbildung 5.1: Übersicht des Boomerang-Angriffs auf 5-Runden AES-128

zur Anwendung. In der fünften Runde entfällt die MixColumns-Operation, wie auch in der 10 Runden Version von AES-128. Die komplette Darstellung des Boomerang-Angriffs auf 5-Runden AES-128 findet sich in der Abbildung 5.1. Dabei sind die Runden mit R_1 bis R_5 beschriftet. Der Ablauf von AES lässt sich durch die folgenden Schritte verkürzt darstellen.

- AddRoundKey(0)
- Von $i = 1$ bis 4
  - SubBytes
  - ShiftRows
  - MixColumns
  - AddRoundKey(i)
- SubBytes
- ShiftRows
- AddRoundKey(5)

Algorithmus 8: AES auf 5 Runden reduziert

5.1.1 Die Differentiale

Neben dem Begriff der Differenzen benötigen wir einen weiteren Begriff für diesen Angriff, den wir zunächst definieren.

Definition 5.1.2 *Ein Muster aktiver Bytes (kurz Muster) zweier Texte ist eine bestimmte Anordnung von aktiven Bytes in der Differenz dieser Texte innerhalb der 16 Byte Zustandsmatrix von AES. Nehmen die aktiven Bytes eines Musters konkrete Werte an, so sprechen wir von Differenzen. Aktive Bytes werden durch ein $*$ Symbol grafisch dargestellt.*

Wenn wir uns mit Mustern von aktiven Bytes beschäftigen, dann spielen die konkreten Werte der aktiven Bytes keine Rolle. Wichtig ist nur, dass die betrachteten Bytes an bestimmten Positionen aktiv sind. Wir verwenden als vereinfachte Schreibweise die folgenden Konventionen: SubBytes (SB), ShiftRows (SR), MixColumns (MC) und AddRoundKey (AK). Um die jeweilige Runde ablesen zu können, indizieren wir die Operationen.

5.1.1.1 Das Differential E_0

Sei α ein Muster zweier Klartexte mit aktiven Bytes an den Positionen 0, 5, 10 und 15, wie in Abbildung 5.2 dargestellt. Es ist wichtig, dass α nun keine konkrete Differenz zwischen zwei Klartexten, sondern nur noch einer bestimmten Anordnung von aktiven Bytes entspricht. Hierbei ist zu beachten, dass nicht die Muster mit AES ver- oder entschlüsselt werden, sondern lediglich die Texte, die zu den Mustern führen. Es wird also beispielsweise ein Text fixiert und ein anderer Text so gewählt, dass dieser zusammen mit dem fixierten Text ein bestimmtes Muster bildet. Die gewählten Texte werden dann durch AES ver- oder entschlüsselt, aber nie die Muster. In diesem Abschnitt werden wir diese Unterscheidung berücksichtigen. In den folgenden Abschnitten werden wir darauf nicht mehr genauer eingehen, sie behält jedoch weiterhin ihre Gültigkeit.

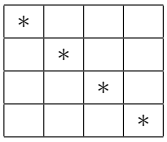


Abbildung 5.2: Muster aktiver Bytes zwischen zwei Klartexten im Boomerang-Angriff

Abbildung 5.3 verdeutlicht das Differential E_0. Für die in diesem Angriff verwendeten Differentiale betrachten wir nur noch Muster von aktiven Bytes.

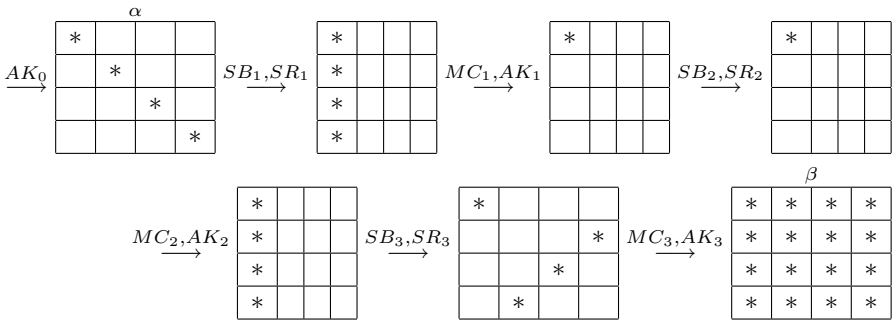


Abbildung 5.3: Das Differential für E_0 der Runden 1 bis 3 im Boomerang-Angriff

Ein Klartextpaar mit Muster α hat nach AK_0, SB_1, SR_1 ein Muster, bei dem sich die vier aktiven Bytes in der nullten Spalte befinden. Aus diesen vier aktiven Bytes wollen wir ein aktives nach MC_1 erhalten. Nach Annahme 5.1.1 auf Seite 36 ist die Ausgabe der S-Box zufällig gleichverteilt. Damit ist auch die Ein- und Ausgabe von MC zufällig gleichverteilt. Wir betrachten nun zwei Klartexte x und y, dessen XOR-Verknüpfung einem Muster α entspricht. Den Wert von x nach MC_1 fixieren wir. Nun wird die Wahrscheinlichkeit dafür, dass

$$Q = MC_1(SR_1(SB_1(x))) \oplus MC_1(SR_1(SB_1(y)))$$

ein aktives Byte an Position 0 besitzt gesucht. Wir müssen nur die erste Spalte von $P = SR_1(SB_1(y))$ betrachten, da MC spaltenweise auf der Zustandsmatrix operiert. Vor MC_1 kann der Text P nun 2^{32} mögliche Bit-Werte in der ersten Spalte annehmen. Von diesen kommen jedoch nur $2^8 - 1$ Werte in Frage, welche zu einem aktiven Byte in Q führen. Für dieses gibt es vier mögliche Positionen innerhalb einer Spalte, so dass wir die Wahrscheinlichkeit

$$\Pr(\text{4 aktive Bytes in 1 aktives mit beliebiger Position}) = 4 \cdot \frac{2^8 - 1}{2^{32}} \approx 2^{-22}$$

erhalten. Da es sich bei dem Angriff um ein heuristisches Verfahren handelt, reicht es oft aus, von genauen Werten zu approximieren, um besser rechnen zu können. Wir weisen ausdrücklich darauf hin, dass wir in den folgenden Analysen auf die genaue Betrachtung der Texte verzichten. Deswegen sprechen wir nur noch davon, dass Muster mit einer gewissen Wahrscheinlichkeit entstehen, was genau genommen auf die konkreten Werte von zwei Texten zurückzuführen ist.

Die Operationen AK_1, SB_2 und SR_2 ändern nichts an der Anzahl der aktiven Bytes. Das anschließende MC_2 erzeugt aus einem aktiven Byte vier aktive mit der Wahrscheinlichkeit:

$$\Pr(\text{1 aktives Byte in 4 Aktive}) = 1$$

Die in MixColumns verwendete Matrix besitzt die MDS (maximum distance separable) Eigenschaft für lineare Codes und erreicht damit exakt die Singleton-Schranke. D.h., seien $N_A \in \{1, \cdots, 4\}$ die Anzahl aktiver Bytes in einer Spalte, so entstehen nach MC mindestens $5 - N_A$ aktive Bytes in dieser Spalte. Siehe dazu [13, S.39-40] und [17, S.2].

Die vier aktiven Bytes der nullten Spalte werden durch SR_3 auf je eine Spalte verteilt. MC_3 sorgt dafür, dass alle 16 Bytes aktiv sind. Das Muster, welches nach AK_3 entsteht nennen wir β.

Wir haben gesehen, dass sich die Wahrscheinlichkeit des Differentials für E_0 durch die Operation MC_1 bestimmt, welches zu

$$\Pr(\alpha \rightarrow \beta) = 2^{-22}$$

führt.

5.1.1.2 Das Differential E_1^{-1}

Das Differential E_1^{-1} verläuft in entgegengesetzter Richtung zu E_0. Es bekommt als Eingabe ein Muster δ, welches ein aktives Byte an der Position 0 besitzt. Eine Übersicht befindet sich in Abbildung 5.4.

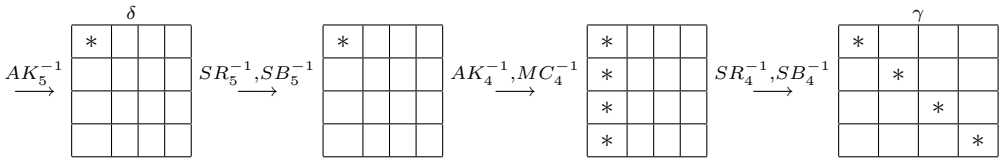


Abbildung 5.4: Das Differential für E_1^{-1} der Runden 4 bis 5 im Boomerang-Angriff

Aus einem Chiffretextpaar mit Muster δ erhalten wir, wie zuvor ermittelt, nach MC_4^{-1} vier aktive Bytes in der nullten Spalte mit der Wahrscheinlichkeit 1. SR_4^{-1} sorgt dafür, dass die aktiven Bytes auf die Positionen 0, 5, 10 und 15 verteilt werden.

Für das Differential E_1^{-1} haben wir eine Wahrscheinlichkeit von

$$\Pr(\gamma \leftarrow \delta) = 1$$

dafür ermittelt, dass aus einem δ Muster ein γ Muster entsteht.

5.1.1.3 Das Differential E_0^{-1}

Das Differential E_0^{-1} ist das Differential E_0 in umgekehrter Richtung. Da wir nur noch mit Mustern und nicht mehr mit konkreten Differenzen arbeiten, gilt nicht zwangsläufig:

$$\begin{aligned} \Pr(\alpha \rightarrow \beta) &= \Pr(\alpha \leftarrow \beta) \\ \Pr(\gamma \rightarrow \delta) &= \Pr(\gamma \leftarrow \delta) \end{aligned}$$

Die Muster sind mit verkürzten Differenzen [30] vergleichbar, bei denen Ver- und Entschlüsselung im Differential ebenfalls unterschiedliche Wahrscheinlichkeiten aufweisen. Eine verkürzte Differenz ist eine Differenz, in der nicht für alle aktiven Bytes die konkrete Differenz bekannt ist. Wir müssen deshalb Ver- und Entschlüsselung getrennt untersuchen, da sich deren Wahrscheinlichkeiten unterscheiden können. Abbildung 5.5 stellt das Differential E_0^{-1} dar, welches wir im Folgenden beschreiben.

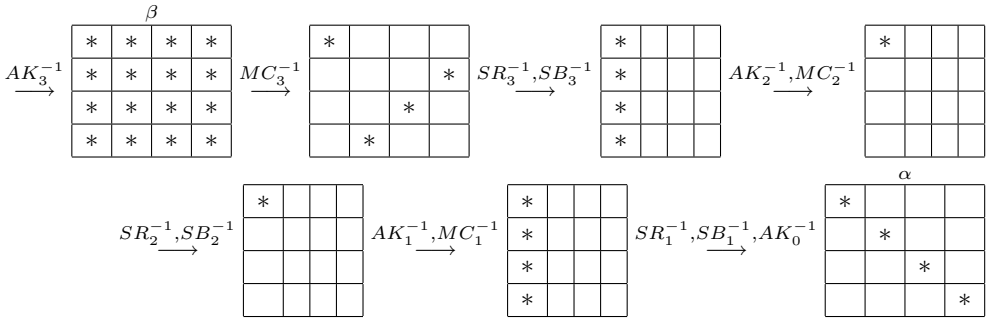


Abbildung 5.5: Das Differential für E_0^{-1} der Runden 3 bis 1 im Boomerang-Angriff

Das MC_3 kann mit einer Wahrscheinlichkeit von eins rückgängig gemacht werden, wenn das Muster β der Eingabe von E_0^{-1} genau dem Muster β der Ausgabe des Differentials E_0 entspricht. An dieser Stelle tritt eine Veränderung in der Betrachtungsweise auf. Wir sehen β zwar immer noch als Muster von aktiven Bytes an, jedoch fordern wir nun zusätzlich, dass nicht nur die Positionen der aktiven Bytes übereinstimmen sondern auch dessen Werte. Mit anderen Worten: An dieser Stelle spielt es eine entscheidende Rolle, dass die konkreten Differenzen der Ausgabe von E_0 und der Eingabe von E_0^{-1} übereinstimmen. Stimmen die Werte der aktiven Bytes überein, so kann das Muster, welches in E_0 vor MC_3 auftrat, in E_0^{-1} nach MC_3^{-1} mit der Wahrscheinlichkeit 1 wiederhergestellt werden. Das folgende Beispiel verdeutlicht den Zusammenhang zwischen den Mustern.

Beispiel 5.1.3 *Sei β_{E_0} das Muster der Ausgabe des Differentials für E_0 und sei $\beta_{E_0^{-1}}$ das Muster der Eingabe des Differentials für E_0^{-1}. Zwei Muster sind gleich, wenn sie die gleiche Anzahl an aktiven Bytes besitzen und diese jeweils auf die gleichen Bytes in der Zustandsmatrix verteilt sind. Betrachten wir folgende Muster*

$$\beta_{E_0} = \begin{array}{|c|c|c|c|}\hline * & * & * & * \\ \hline * & * & * & * \\ \hline * & * & * & * \\ \hline * & * & * & * \\ \hline \end{array} \qquad \beta_{E_0^{-1}} = \begin{array}{|c|c|c|c|}\hline * & * & * & * \\ \hline * & * & * & * \\ \hline * & * & * & * \\ \hline * & * & * & * \\ \hline \end{array}$$

so stimmen die Muster β_{E_0} und $\beta_{E_0^{-1}}$ überein. Sei nun a eine bekannte Differenz und zwei weitere Muster β'_{E_0} und $\beta'_{E_0^{-1}}$ durch

$$\beta'_{E_0} = \begin{array}{|c|c|c|c|}\hline a & a & a & \\ \hline a & a & a & \\ \hline a & & a & \\ \hline & & a & \\ \hline \end{array} \qquad \beta'_{E_0^{-1}} = \begin{array}{|c|c|c|c|}\hline a & a & a & \\ \hline a & a & a & \\ \hline a & & a & \\ \hline & & a & \\ \hline \end{array}$$

gegeben. In diesem Fall stimmen nicht nur die Muster von β'_{E_0} und $\beta'_{E_0^{-1}}$ sondern auch deren konkrete Differenzen der aktiven Bytes überein.

Wie sich sowohl die Gleichheit zweier Muster als auch die Gleichheit deren konkreter Differenzen im Angriff realisieren lässt, werden wir später bei der Analyse des Angriffs zeigen. Zunächst ist aber nur von Bedeutung, dass mit der Wahrscheinlichkeit 1 nach MC_3^{-1} vier aktive Bytes in den Positionen 0, 7, 10 und 13 erzeugt werden. SR_3^{-1} sorgt dafür, dass sich diese aktiven Bytes in der nullten Spalte befinden. Erneut erhalten wir mit einer Wahrscheinlichkeit von 2^{-22} aus vier aktiven Bytes ein aktives nach MC_2^{-1}. Das folgende SR_2^{-1} ändert allenfalls die Position der aktiven Bytes, welche durch MC_1^{-1} zu vier Aktiven mit der Wahrscheinlichkeit 1 werden. Wir erhalten nach AK_0^{-1} ein Muster α mit vier aktiven Bytes in den Positionen 0, 5, 10 und 15.

Das Differential für E_0^{-1} erzeugt mit einer Wahrscheinlichkeit von

$$\Pr(\alpha \leftarrow \beta) = 2^{-22}$$

aus einem β Muster ein α Muster.

5.1.2 Der Ablauf des Angriffs

1. Erzeuge 2^{32} Klartexte $\{P_i\}$, die in den Bytes 0, 5, 10, 15 alle möglichen Kombinationen annehmen und in den restlichen Bytes denselben Wert haben.
2. Lasse $\{P_i\}$ zu den Chiffretexten $\{C_i\}$ verschlüsseln.
3. Wähle ein Muster δ, welche ein aktives Byte an Position 0 besitzt.
4. Berechne die neuen Chiffretexte $\{D_i\}$ durch $D_i = C_i \oplus \delta$.
5. Lasse $\{D_i\}$ zu den Klartexten $\{O_i\}$ entschlüsseln.
6. Benutze nur die Quartette (P_i, P_j, O_i, O_j), für die $P_i \oplus P_j = \alpha$ und $O_i \oplus O_j = \alpha$ gilt.
7. Initialisiere einen Zähler für jede Kombinationen von 32 Schlüsselbits des initialen Rundenschlüssels für die Bytes 0, 5, 10 und 15 mit Null.
8. Gehe über alle möglichen Kombinationen von 32 Schlüsselbits des initialen Rundenschlüssels für die Bytes 0, 5, 10 und 15.
   - 8.1. Prüfe für alle Quartette, welche den Test in Schritt 6 bestanden haben, ob die Paare P_i, P_j und O_i, O_j nach der Verschlüsselung einer Runde mit den in Schritt 8 gewählten Schlüsselbits in den Mustern ein aktives Byte besitzen.
   - 8.2. Erfüllt ein Quartett Schritt 8.1., so erhöhe den Zähler auf die verwendete Kombination von Schlüsselbits um eins.
9. Gib die Schlüsselbits als korrekt aus, deren Zähler einen Wert von zwei oder größer annimmt.

Algorithmus 9: Boomerang-Angriff auf 5-Runden AES-128

5.1.3 Der Boomerang-Angriff

Schritt 1: Für den Angriff benutzen wir eine Menge von 2^{32} Klartexten $\{P_i\}$, die alle möglichen Werte in den Bytes 0, 5, 10 und 15 annehmen, während die restlichen Bytes konstant gehalten werden. Aus diesen 2^{32} Klartexten können wir

$$\frac{2^{32}(2^8 - 1)^4}{2}$$

Klartextpaare mit dem Muster α bilden. Zur Vereinfachung approximieren wir den genauen Wert durch

$$\frac{(2^{32})^2}{2} = 2^{63}.$$

Schritt 2: Alle 2^{32} Klartexte werden nun verschlüsselt. Ein Klartextpaar P_i, P_j mit $P_i \oplus P_j = \alpha$ hat ein Muster β mit der Wahrscheinlichkeit 2^{-22} nachdem es das Differential E_0 durchlaufen hat.

Schritt 3-4: Aus den resultierenden Chiffretexten C_i, C_j werden die neuen Chiffretexte D_i, D_j berechnet, so dass

$$C_i \oplus D_i = \delta \quad \text{und} \quad C_j \oplus D_j = \delta$$

gilt. Dazu wird das Muster δ auf die Chiffretexte C_i, C_j addiert. Genauer gesagt, wird eine Differenz $\Delta \in \delta$ auf die Chiffretexte addiert. Wir vernachlässigen im Folgenden dieses Detail und weißen den Leser an dieser Stelle darauf hin.
Schritt 5: Die neuen Chiffretextpaare D_i, D_j werden zu den neuen Klartextpaaren O_i, O_j entschlüsselt. Wie bereits erläutert, durchläuft das Muster δ das Differential E_1^{-1} mit der Wahrscheinlichkeit 1 und wird dabei zu dem Muster γ transformiert.

Um nun ein Muster α zwischen den neuen Klartexten O_i, O_j zu erhalten, müssen wir Folgendes durchführen. Aus der Verschlüsselung der Klartexte P_i, P_j durch E_0 erhalten wir die Textpaare A_i, A_j mit Muster β. Auf der anderen Seite erhalten wir aus der Entschlüsselung der Chiffretexte D_i und D_j mit E_1^{-1} die Textpaare B_i und B_j. Da C_i, D_i und C_j, D_j jeweils ein Muster δ aufweisen, besitzen A_i, B_i und A_j, B_j ein Muster γ nach E_1^{-1} mit der Wahrscheinlichkeit 1. Da es sich bei γ nicht um eine konkreten Differenz, sondern um ein Muster mit vier aktiven Bytes handelt, müssen wir zunächst sicherstellen, dass

$$A_i \oplus B_i = A_j \oplus B_j$$

gilt, um den Boomerang-Angriff erfolgreich durchführen zu können. Dies trifft mit der Wahrscheinlichkeit

$$\Pr(\gamma_{A_i \oplus B_i} = \gamma_{A_j \oplus B_j}) = 2^{-32}$$

zu. In diesem Fall sind sowohl die Muster als auch deren konkrete Differenzen gleich.

Über eine XOR-Verknüpfung können wir

$$(A_i \oplus B_i) \oplus (A_i \oplus A_j) \oplus (A_j \oplus B_j)$$

berechnen. Mit der Wahrscheinlichkeit $\Pr(\alpha \rightarrow \beta) \cdot \Pr(\gamma \leftarrow \delta)^2$ gilt:

$$\gamma \oplus \beta \oplus \gamma \quad = \quad \beta = (B_i \oplus B_j)$$

Damit erhalten wir das benötigte Muster β für die Eingabe des Differentials E_0^{-1}. Wird das Muster β zur Eingabe des Differentials E_0^{-1} auf die eben beschriebene Weise gebildet, so stimmen die Werte der aktiven Bytes in dem Muster der Ausgabe von E_0 mit dem Muster der Eingabe von E_0^{-1} überein.

Wir können nun die Wahrscheinlichkeit, dass ein Klartextpaar P_i, P_j ein *korrektes Boomerang-Quartett* (P_i, P_j, O_i, O_j) erzeugt, berechnen:

$$\begin{aligned} \Pr(\text{ein korrektes Boomerang-Quartett}) &= \\ \Pr(\alpha \rightarrow \beta) \cdot \Pr(\gamma \leftarrow \delta)^2 \cdot \Pr(\gamma_{A_i \oplus B_i} = \gamma_{A_j \oplus B_j}) \cdot \Pr(\alpha \leftarrow \beta) &= 2^{-22} \cdot 1^2 \cdot 2^{-32} \cdot 2^{-22} \\ &= 2^{-76} \end{aligned}$$

Wir erhalten ein korrektes Boomerang-Quartett mit der Wahrscheinlichkeit:

$$\Pr(\text{ein korrektes Boomerang-Quartett}) \quad = \quad 2^{-76}$$

Schritt 6: Es werden nur die Quartette verwendet, welche aus zwei Paaren mit je einem Muster α bestehen.

Schritt 8 - 8.1: Wir testen alle 2^{32} möglichen Kombinationen von Schlüsselbits der Bytes 0, 5, 10 und 15. Dabei zählen wir (**Schritt 8.2**), wie viele Quartette (P_i, P_j, O_i, O_j) Paare mit einer aktiven S-Box nach MC_1 haben. Zählen wir für eine Wahl der 32 Schlüsselbits mindestens zwei Quartette, die diese Bedingung erfüllen, so geben wir die verwendeten Schlüsselbits als korrekt aus (**Schritt 9**).

5.1.4 Analyse des Boomerang-Angriffs

Damit wir den Angriff durchführen können, verwenden wir 2^{14} Mengen von 2^{63} Klartextpaaren. 2^{14} Menge ist die kleinste Anzahl, um einen erfolgreichen Angriff durchführen zu können und somit ausreichend viele korrekte Boomerang-Quartette zu erhalten. Deshalb erwarten wir

$$\begin{aligned} \#\text{ Mengen} \cdot \#\text{ Quartette} \cdot \Pr(\text{ein korrektes Boomerang Quartett}) &= \\ 2^{14} \cdot 2^{63} \cdot 2^{-76} &= 2 \end{aligned}$$

korrekte Boomerang Quartette. Diese Quartette erfüllen alle verwendeten Differentiale. Für den Angriff benötigen wir damit $2^{46} (= 2^{14} \cdot 2^{32})$ AES-128 Verschlüsselungen über 5 Runden und ebenso viele Entschlüsselungen, welches uns zu einem gesamten Zeitaufwand von 2^{47} AES-128 Verschlüsselungen führt. Der Speicherbedarf beläuft sich auf $2^{33} (= 2 \cdot 2^{32})$ Blöcke, da wir für jeden Klartext auch den dazugehörigen modifizierten Klartext speichern müssen, welcher aus dem Boomerang entstanden ist. Die 2^{33} Blöcke entsprechen 2^{37} Bytes.

Der Boomerang-Angriff auf 5-Runden AES-128 benötigt 2^{47} Verschlüsselungen von 5-Runden AES-128 und 2^{37} Bytes an Speicher. Dies entspricht 128 Gigabyte.

5.2 Verbesserte Variante des Boomerang-Angriffs auf 5-Runden AES-128

Wir können den Angriff erheblich schneller machen, wie es von Biryukov in [10] gezeigt wurde, indem wir das Differential E_0^{-1} ein wenig abändern.

5.2.1 Das verbesserte Differential E_0^{-1}

In Abbildung 5.6 ist eine verbesserte Variante des Differentials E_0^{-1} dargestellt.

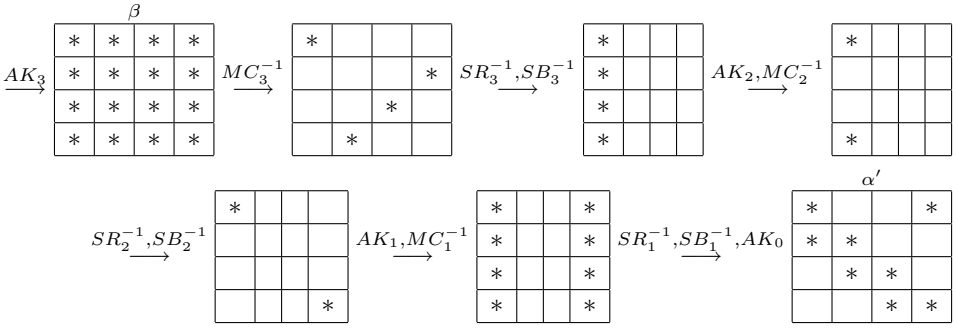


Abbildung 5.6: Das verbesserte Differential E_0^{-1} der Runden 3 bis 1 im Boomerang-Angriff

Auch in diesem Differential lässt sich das Muster, welches in E_0 vor MC_3 bestand, mit der Wahrscheinlichkeit 1 nach MC_3^{-1} in E_0^{-1} erhalten. Dadurch bekommen wir vier aktive Bytes in den Positionen 0, 7, 10 und 13. Das folgende SR_3^{-1} shiftet diese in die

nullte Spalte. Durch MC_2^{-1} lassen wir aus vier aktiven Bytes zwei Aktive entstehen. Da für uns die genaue Position der aktiven Bytes nicht von Bedeutung ist, gibt es für diese sechs mögliche Anordnungen. Die Wahrscheinlichkeit dafür beträgt:

$$\Pr(\text{4 aktive Bytes in 2 Aktive mit beliebigen Positionen}) = 6 \cdot \frac{2^{16} - 3}{2^{32}} \approx 2^{-13.5}$$

Das Muster der Ausgabe des Differentials E_0^{-1} ist nun nicht mehr α sondern α'. Dies weicht von der bekannten Darstellung des Boomerang-Angriffs ab, da der Boomerang nicht mehr in dasselbe α Muster zurückkehrt.

Das Differential E_0^{-1} hat eine verbesserte Wahrscheinlichkeit von

$$\Pr(\alpha' \leftarrow \beta) = 2^{-13.5}$$

aus einem β Muster ein α' Muster zu erzeugen im Vergleich zu dem Differential E_0^{-1} aus Kapitel 5.1.1.3 auf Seite 39, welches eine Wahrscheinlichkeit von 2^{-22} besitzt.

5.2.2 Der Ablauf des Angriffs

Der Ablauf des Angriffs verändert sich ein wenig. Wir beschreiben im Folgenden nur die relevanten Änderungen genauer.

1. Erzeuge 2^{32} Klartexte $\{P_i\}$, die in den Bytes 0, 5, 10, 15 alle möglichen Kombinationen annehmen und in den restlichen Bytes den gleichen Wert haben.
2. Lasse $\{P_i\}$ zu den Chiffretexten $\{C_i\}$ verschlüsseln.
3. Wähle ein Muster δ, welche ein aktives Byte an Position 0 besitzt.
4. Berechne die neuen Chiffretexte $\{D_i\}$ durch $D_i = C_i \oplus \delta$.
5. Lasse $\{D_i\}$ zu den Klartexten $\{O_i\}$ entschlüsseln.
6. Sortiere die $\{O_i\}$, so dass acht Bytes an den Positionen 2, 3, 4, 7, 8, 9, 13 und 14 dieselben Werte annehmen.
7. Benutze nur die Klartextpaare O_i, O_j, welche eine Null Differenz in diesen acht Bytes besitzen. Wenn keine gefunden wurden, dann gehe zurück zum Schritt 1.
8. Initialisiere einen Zähler für jede Kombinationen von 32 Schlüsselbits des initialen Rundenschlüssels für die Bytes 0, 5, 10 und 15 mit Null.
9. Gehe über alle möglichen Kombinationen von 32 Schlüsselbits des initialen Rundenschlüssels für die Bytes 0, 5, 10 und 15.
   - 9.1. Verschlüssele mit den in Schritt 9 gewählten Schlüsselbits die Quartette (P_i, P_j, O_i, O_j) eine Runde.
   - 9.2. Haben P_i, P_j und O_i, O_j ein aktives Byte nach dem MC_1, so erhöhe den Zähler auf den verwendeten Schlüsselbits um eins.
   - 9.3. Entsteht nur bei P_i, P_j ein aktives Byte nach MC_1, so erhöhe den Zähler um $\frac{1}{2}$.
10. Gib die Schlüsselbits als korrekt aus, deren Zähler den größten Wert aufweist.

Algorithmus 10: verbesserter Boomerang-Angriff auf 5-Runden AES-128

5.2.3 Der verbesserte Boomerang-Angriff

Bis zum Schritt 5 gibt es keine Veränderung im Angriff, so dass wir unsere Beschreibung erst mit Schritt 6 beginnen. Allerdings ändert sich die Wahrscheinlichkeit für ein korrektes Boomerang Quartett, welche jetzt

$$\begin{aligned} \Pr(\text{ein korrektes Boomerang-Quartett}) &= \\ \Pr(\alpha \to \beta) \cdot (\Pr(\gamma \leftarrow \delta))^2 \cdot \Pr(\gamma_{A_i \oplus B_i} = \gamma_{A_j \oplus B_j}) \cdot \Pr(\alpha' \leftarrow \beta) &= 2^{-22} \cdot 1^2 \cdot 2^{-32} \cdot 2^{-13.5} \\ &= 2^{-67.5} \end{aligned}$$

beträgt.
Schritt 6-7: Wir sortieren die neu entstandenen Klartexte, so dass die Klartextpaare O_i, O_j eine Null-Differenz in acht Bytes besitzen.
Schritt 9: Nun wählen wir 32 Schlüsselbits an den Bytepositionen 0, 5, 10 und 15 und testen diese für jedes Quartett, welches den Test in Schritt 7 bestanden hat. Dabei unterscheiden wir zwei Fälle:
Schritt 9.2: Haben die Klartextpaare P_i, P_j und O_i, O_j eines korrekten Boomerang-Quartetts (P_i, P_j, O_i, O_j) vier aktive Bytes an den Position 0, 5, 10 und 15, so gilt folgendes: Entweder waren die geratenen Schlüsselbits richtig oder wir erhalten durch Zufall mit der Wahrscheinlichkeit 2^{-22} ein aktives Byte nach dem MC_1. Damit haben wir mit einer Wahrscheinlichkeit von 2^{-44} ein aktives Byte auf beiden Klartextpaaren nach MC_1. Bei 2^{32} möglichen Schlüsselbits erhalten wir für den Fall, dass eine korrekte Wahl von Schlüsselbits den Test besteht die Wahrscheinlichkeit:

$$1 - 2^{-44} \cdot 2^{32} = 1 - 2^{-12}$$

Schritt 9.3: Nun müssen wir allerdings noch den Fall betrachten, dass ein Quartett aus zwei Klartextpaaren besteht, bei dem das Paar O_i, O_j keine aktiven Bytes auf der Diagonale (Bytes 0, 5, 10, 15) besitzt. Wir hatten gesagt, dass es sechs Möglichkeiten gibt, zwei aktive Bytes in einer Spalte anzuordnen. Das Paar O_i, O_j hat nur dann vier aktive Bytes auf der Diagonale, wenn sich eines der beiden aktiven Bytes nach MC_2^{-1} an der Position 0 befindet. Hierfür gibt es genau drei Möglichkeiten, die zwei aktiven Bytes anzuordnen, so dass diese Bedingung gilt. Damit beträgt die Wahrscheinlichkeit, dass zwei Paare eines Quartetts ein Paar mit vier aktiven Bytes an den Positionen 0, 5, 10 und 15 besitzen, genau $\frac{3}{6} = \frac{1}{2}$. Ebenso ist die Wahrscheinlichkeit $\frac{1}{2}$, dass das Paar O_i, O_j keine aktiven Bytes an diesen Positionen besitzt.

Wir nehmen genau die Schlüsselbits als richtig an, für die wir zwei Boomerang Quartetts zählen, wenn es pro Quartett ein Klartextpaar gibt, welches ein aktives Byte nach MC_1 aufweist. In diesem Fall gibt es pro Paar

$$2^{-22} \cdot 2^{32} = 2^{10}$$

mögliche Kombinationen von Schlüsselbits, die den Test bestehen. Deshalb führen wir diesen Test zweimal durch, um die korrekten Schlüsselbits mit einer hohen Wahrscheinlichkeit zu filtern. Die Wahrscheinlichkeit, dass ein Klartextpaar P_i, P_j ein *korrektes Boomerang-Quartett* (P_i, P_j, O_i, O_j) erzeugt, beträgt:

$$\begin{aligned} \Pr(\text{ein korrektes Boomerang-Quartett}) &= \\ \Pr(\alpha \to \beta) \cdot \Pr(\gamma \leftarrow \delta)^2 \cdot \Pr(\gamma_{A_i \oplus B_i} = \gamma_{A_j \oplus B_j}) \cdot \Pr(\alpha' \leftarrow \beta) &= 2^{-22} \cdot 1^2 \cdot 2^{-32} \cdot 2^{-13.5} \\ &= 2^{-67.5} \end{aligned}$$

Die Wahrscheinlichkeit für ein *korrektes Boomerang-Quartett* (P_i, P_j, O_i, O_j) beträgt:

$$\Pr(\text{ein korrektes Boomerang-Quartett}) \;=\; 2^{-67.5}$$

5.2.4 Analyse des verbesserten Boomerang-Angriffs

Für die Durchführung des Angriffs benötigen wir 2^6 Mengen von 2^{63} Klartextpaaren und erhalten damit

$$\begin{aligned}\#\text{Mengen} \cdot \#\text{Quartette} \cdot \Pr(\text{ein korrektes Boomerang-Quartett}) \;&=\\ 2^6 \cdot 2^{63} \cdot 2^{-67.5} \;&\approx\; 3\end{aligned}$$

korrekte Boomerang-Quartette. Für den Angriff benötigen wir $2^{38} (= 2^6 \cdot 2^{32})$ Verschlüsselungen von 5-Runden AES-128 und ebenso viele Entschlüsselungen. Dies führt zu einem gesamten Zeitaufwand von $2^{39} (= 2 \cdot 2^{38})$ Verschlüsselungen. Wir benötigen dafür $2^{33} (= 2 \cdot 2^{32})$ Blöcke, welches 2^{37} Bytes entspricht.

Hat ein Textpaar das Differential E_0^{-1} durchlaufen, so hat es acht aktive Bytes und acht Bytes mit einer Null-Differenz. Ein zufällig gewähltes Quartett (P_i, P_j, O_i, O_j) mit $P_i \oplus P_j = \alpha$ besitzt mit einer Wahrscheinlichkeit von $(2^{-8})^8 = 2^{-64}$ ein Klartextpaar O_i, O_j mit acht Bytes, welches eine Null-Differenz aufweist. Da wir in MC_1^{-1} die Position der zwei aktiven Bytes nicht fixiert haben, können wir die Filterung von MC_1^{-1} nach acht Bytes mit Null-Differenzen als $64 - \log 6$ Filterung auffassen. Insgesamt erwarten wir

$$\begin{aligned}\#\text{ Mengen} \cdot \#\text{ Quartette} \cdot \Pr(\text{ein falsches Boomerang Quartetts}) \;&=\\ 2^6 \cdot 2^{63} \cdot 2^{-64} \cdot 6 \;&=\; 192\end{aligned}$$

falsche Boomerang-Quartette, die nur durch Zufall die gewünschte Differenz α' im Klartextpaar O_i, O_j aufweisen. Der Boomerang-Unterscheiderangriff gibt, wie bereits erwähnt, die Menge der korrekten und falschen Boomerang-Quartette zurück. Nur auf diesen Quartetten wird die Schlüsselsuche vollzogen.

Der verbesserte Boomerang-Angriff auf 5-Runden AES-128 benötigt 2^{39} 5-Runden Verschlüsselungen und 2^{37} Bytes an Speicher.

5.3 Related-Key Rechtecks-Angriff auf 8 Runden AES-192

Der Related-Key Rechtecks-Angriff kombiniert die Related-Key Technik von Biham [1] mit dem Rechtecks-Angriff aus [3]. Diese Art, die beiden Angriffe zu kombinieren, wurde bereits in [27] beschrieben und auf SHACAL angewendet. Der nun folgende Angriff wurde von Kim, Hong, Lee und Preneel in [24] vorgestellt und dabei auf eine 8-Runden reduzierte AES Version mit einer Schlüssellänge von 192 Bit angewandt. Wie schon im vorherigen Kapitel, unterscheidet sich die reduzierte AES Version nur in der Anzahl der Runden. Das initiale AddRoundKey und die letzte Runde ohne MixColumns besteht auch hier. Grafik 5.7 stellt zur Übersicht den Related-Key Rechtecks-Angriff schematisch dar. Für eine genaue Beschreibung verweisen wir auf den Abschnitt 4.5.3 auf Seite 35. Die Klartexte (P_i, P_j, O_i, O_j) werden hierbei mit den Schlüsseln K_a, K_b, K_c und K_d durch die Chiffren E_0 und E_1 verschlüsselt. Es entstehen dadurch die Chiffretexte (C_i, C_j, D_i, D_j). Dabei werden bei der Verschlüsselung jeweils die Rundenschlüssel, welche aus den Schlüsseln K_a, K_b, K_c und K_d berechnet werden, verwendet.

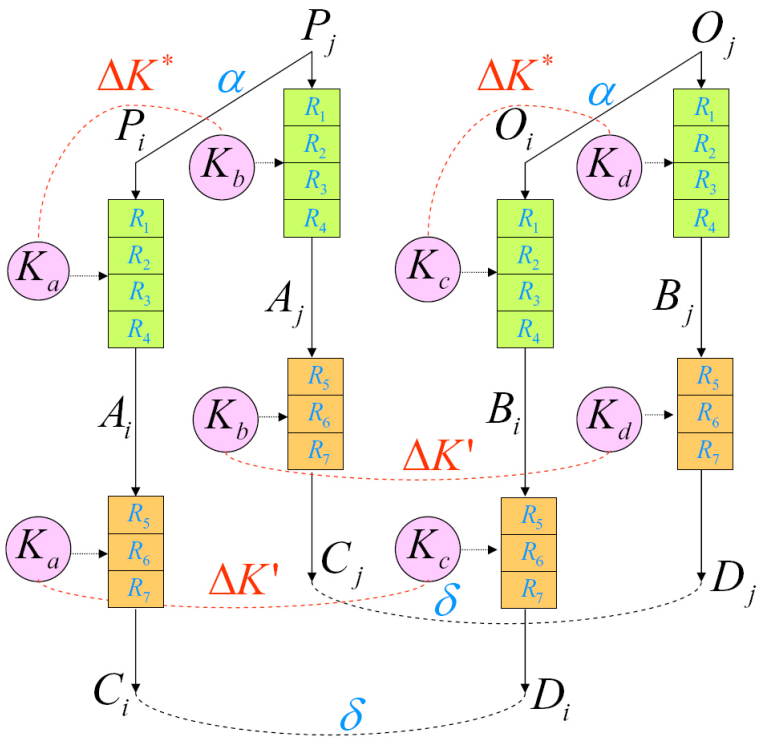


Abbildung 5.7: Darstellung des Related-Key Rechtecks-Angriff auf 8 Runden AES-192

Für diesen Angriff gelten weiterhin die bereits getroffenen Annahmen. Zusätzlich nehmen wir folgendes an.

Annahme 5.3.1 *Die Schlüssel K_a, K_b, K_c, K_d sind durch*

$$K_a \oplus K_b = \Delta K^* = K_c \oplus K_d \quad \textit{und} \quad K_a \oplus K_c = \Delta K' = K_b \oplus K_d$$

miteinander verbunden.

Annahme 5.3.2 *Ein Quartett (P_i, P_j, O_i, O_j) hat die folgende Eigenschaft:*

$$P_i \oplus P_j = \alpha = O_i \oplus O_j$$

5.3.1 Die Related-Keys

Der Angriff verwendet vier Schlüssel K_a, K_b, K_c, K_d, von denen jeweils zwei Schlüssel nach der Annahme 5.3.1 miteinander verbunden sind. Wir werden in diesem Abschnitt die Beziehung zwischen den vier verwendeten Schlüsseln beschreiben. Dabei sei noch einmal darauf hingewiesen, dass der Angreifer die Schlüssel selbst nicht kennt, jedoch deren Beziehungen. Auf die Erzeugung der Rundenschlüssel aus Abschnitt 3.2 wird in diesem Abschnitt zurückgegriffen, um die Differenzen der Rundenschlüssel zu berechnen.

Sei

$$\Delta K^* = \begin{array}{|c|c|c|c|c|c|}\hline & & & & & \\ \hline & & a & a & & \\ \hline & & & & & \\ \hline & & & & & \\ \hline \end{array}$$

eine bekannte Schlüsseldifferenz. Die Differenzen der Rundenschlüssel lassen sich aus den erweiterten Schlüsseln unter Anwendung folgender Regeln erzeugen. Seien W_i und W_i' Spalten der erweiterten Schlüssel von K und K'.

- Falls W_i und W'_i Spalten der erweiterten Schlüssel mit $i \equiv 0 \bmod 6$ sind, so setze:

$$\begin{aligned} W_i \oplus W'_i &= SB(RB(W_{i-1})) \oplus Rcon\left[\frac{i}{6}\right] \oplus W_{i-6} \\ &\quad \oplus SB(RB(W'_{i-1})) \oplus Rcon\left[\frac{i}{6}\right] \oplus W'_{i-6} \\ &= SB(RB(W_{i-1})) \oplus SB(RB(W'_{i-1})) \oplus W_{i-6} \oplus W'_{i-6} \end{aligned}$$

  Hierbei wird RotByte mit RB abgekürzt. Entspricht die Differenz zweier Spalten W_i, W'_i einem passiven Byte, so folgt:

$$\begin{aligned} RB(W_i) \oplus RB(W'_i) &= 0 \\ SB(RB(W_i)) \oplus SB(RB(W'_i)) &= 0 \end{aligned}$$

- Für alle $i \not\equiv 0 \bmod 6$ setze:

$$W_i \oplus W'_i = W_{i-1} \oplus W'_{i-1} \oplus W_{i-6} \oplus W'_{i-6}$$

Algorithmus 11: Erzeugung der Schlüsseldifferenzen

Unter Berücksichtigung dieser Regeln lassen sich die folgenden Differenzen der Rundenschlüssel ableiten:

$\Delta K_0^* =$

		a	a

$\Delta K_1^* =$

$\Delta K_2^* =$

a			

$\Delta K_3^* =$

		a	a

$\Delta K_4^* =$

		b	b
a	a		

$\Delta K_5^* =$

b	b	b	b
a		a	

$\Delta K_6^* =$

b		b	
		a	a
c	c	c	c

$\Delta K_7^* =$

b		b	b
		d	d
c	c	c	

$\Delta K_8^* =$

		b	b
a			
d	d	d	d
c		c	

Hat ein aktives Byte den Wert a, so führt die S-Box, angewendet auf die beiden Werte, die zu einer Differenz a geführt haben, zu einer Differenz b:

$$\begin{aligned} x \oplus y &= a \\ SB(x) \oplus SB(y) &= b \end{aligned}$$

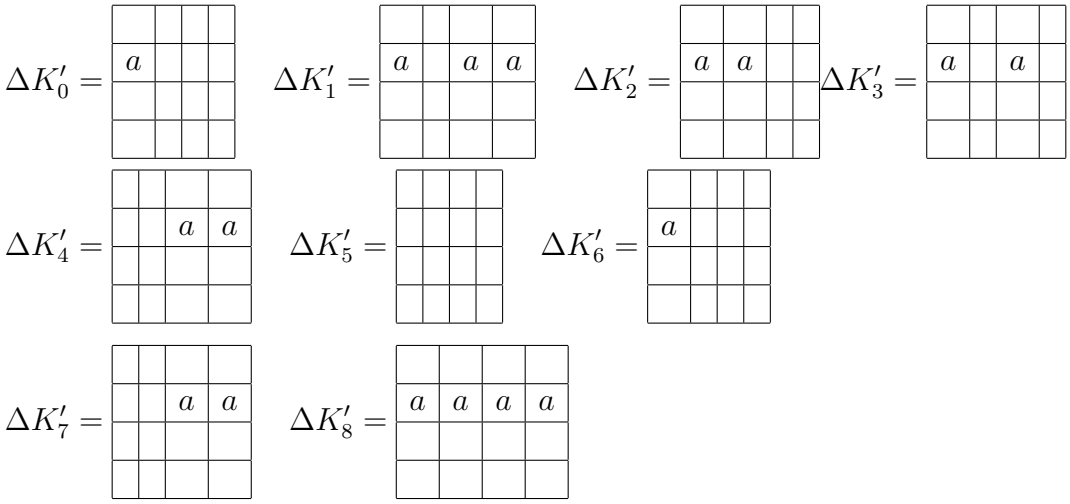
Da a eine bekannte Differenz darstellt, ist b eine unbekannte Differenz, die einen von $2^7 - 1$ Werten annehmen kann. Wir erhalten wegen der Symmetrie-Eigenschaft des XOR 2^7 mögliche Werte, denn $x \oplus y$ ergibt die selbe Differenz wie $y \oplus x$. Die -1 kommt dadurch, dass die Null-Differenz hier nicht auftreten kann, da a nicht der Null-Differenz entspricht. Dies setzt sich mit c und d entsprechend fort. Aus einer unbekannten Differenz b entsteht die unbekannte Differenz c und aus dieser entsteht die ebenfalls unbekannte Differenz d.

Sei außerdem die Differenz

$\Delta K' =$

a				a	

bekannt, die nach Annahme 5.3.1 auf Seite 47 zwischen jeweils zwei Schüsseln besteht. Die Berechnung der Differenzen zwischen den Rundenschlüsseln ergibt:



5.3.2 Das Differential E_0

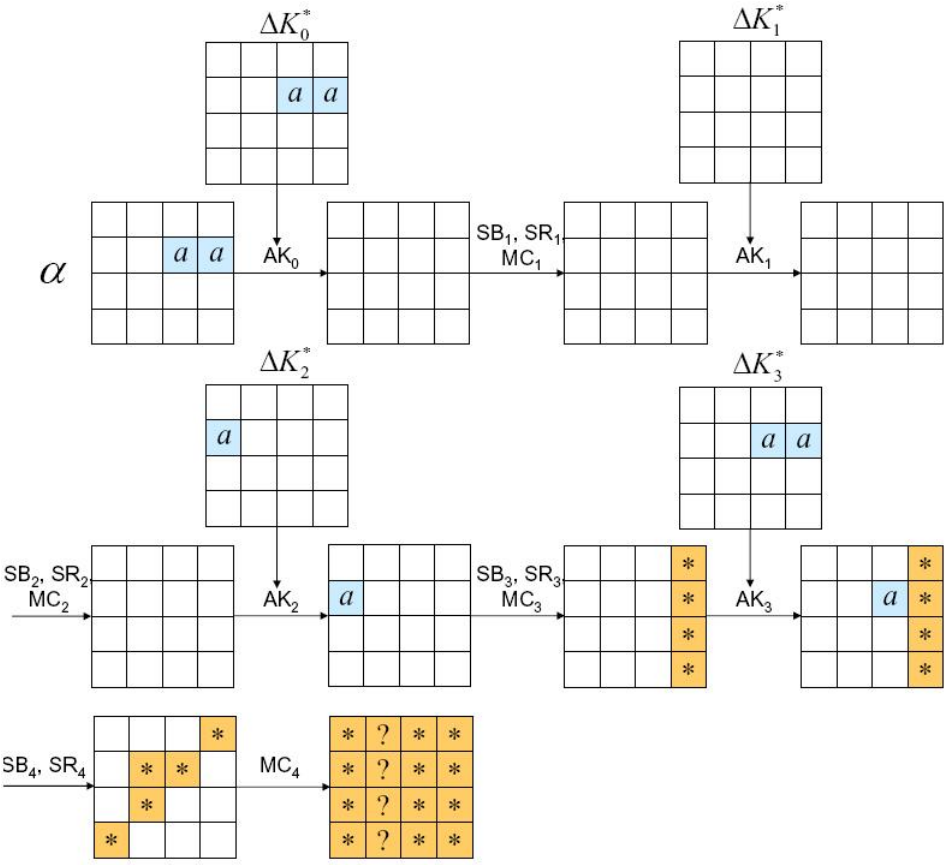


Abbildung 5.8: E_0 der Runden 1 bis 4 im Related-Key Rechtecks-Angriff

In diesem und im nächsten Abschnitt beschreiben wir die Differentiale, welche den Angriff ermöglichen. Sowohl E_0 als auch E_1 haben aufgrund ihrer Struktur eine Wahrscheinlichkeit von 1. Das Differential E_0 verläuft über die Runden 1 bis 4, wobei AK_4 hierbei nicht berücksichtigt wird. Abbildung 5.8 verdeutlicht dieses Differential. Seien P_i und P_j zwei zufällig gewählte Klartexte mit $P_i \oplus P_j = \alpha$. Die Differenz α besteht hier aus zwei aktiven Bytes an den Positionen 9 und 13, die jeweils den bekannten Wert a annehmen. AK_0 führt dazu, dass sich keine aktiven Bytes mehr in der Textdifferenz befinden. Somit führt

das anschließende SB_1, SR_1 und MC_1 zu keiner Veränderung in der Differenz. Auch AK_1 lässt keine aktiven Bytes entstehen, da auch in ΔK_1^* keine aktiven Bytes vorhanden sind. Die Rundenschlüssel sind damit für K_a und K_b in der ersten Runde gleich. Dasselbe gilt auch für K_c und K_d. Erst AK_2 sorgt dafür, dass in der Textdifferenz ein aktives Byte an Position 1 entsteht. SR_3 bringt dieses aktive Byte an Position 13, wonach mit der Wahrscheinlichkeit 1 durch MC_3 vier aktive Bytes entstehen. Nach dem folgenden AK_3 ist auch das Byte an der Position 10 aktiv. SR_4 sorgt dafür, dass jede Spalte mindestens ein aktives Byte enthält. MC_4 aktiviert alle Bytes in Spalte 0, 2 und 3. Die Werte der Spalte 1 lassen wir außer Acht und kennzeichnen diese mit ? in der Abbildung 5.8.

5.3.3 Das Differential E_1

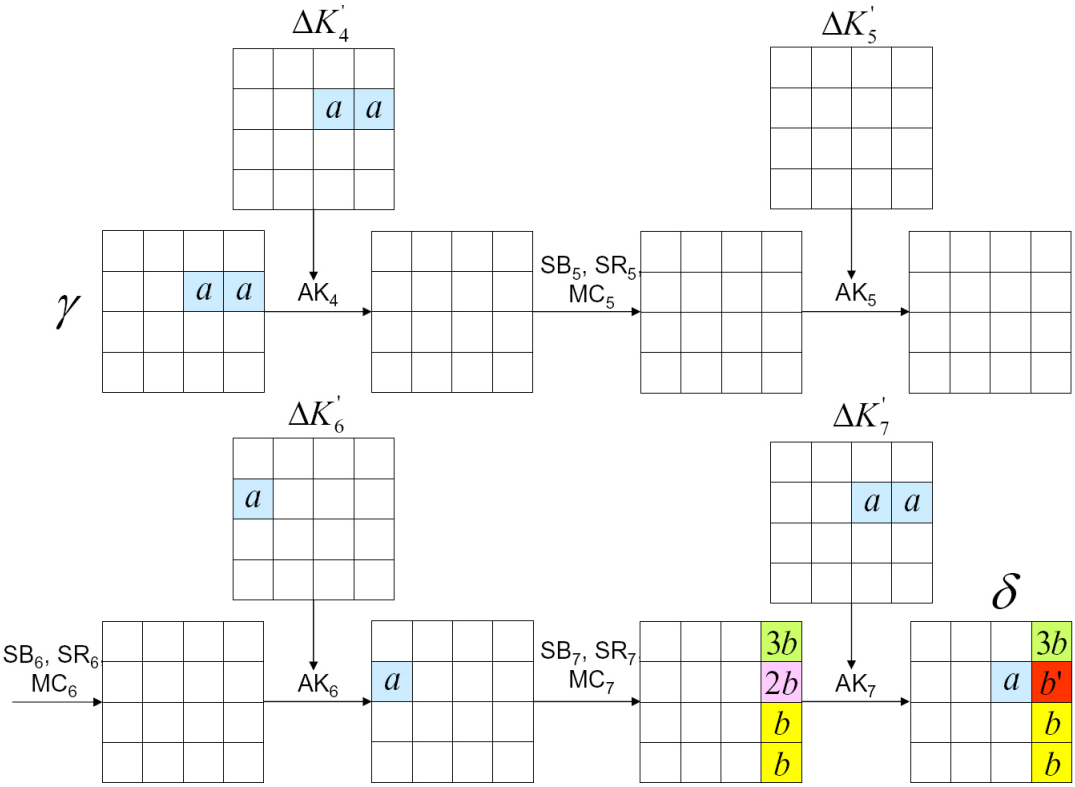


Abbildung 5.9: E_1 der Runden 5 bis 7 im Related-Key Rechtecks-Angriff

Kommen wir nun zur Beschreibung des Differentials E_1, welches ebenfalls eine Wahrscheinlichkeit von 1 besitzt und über die Runden 5 bis 7 verläuft, wobei das AK_4 mit inbegriffen ist. Seien K und K' zwei Schlüssel mit $K \oplus K' = \Delta K'$. Die Klartexte P_i, O_i sowie P_j, O_j werden mit K_a, K_c sowie K_b, K_d durch E_0 verschlüsselt. Wenn $E_{0,K_a}(P_i) \oplus E_{0,K_c}(O_i) = \gamma$ sowie $E_{0,K_b}(P_j) \oplus E_{0,K_d}(O_j) = \gamma$ gilt, so durchlaufen diese Differenzen das Differential E_1 mit der Wahrscheinlichkeit 1. Abbildung 5.8 stellt dieses Differential grafisch dar.

Aus einer γ Differenz mit zwei aktiven Bytes wird nach AK_4 eine Differenz ohne aktive Bytes. Erst AK_6 bringt ein aktives Byte an die Position 1 in die Textdifferenz. Durch SB_7 wird die bekannte a Differenz in eine unbekannte b Differenz transformiert, die einen von $2^7 - 1$ Werten annehmen kann. SR_7 setzt das aktive Byte mit Wert b an die Position 13. Wegen der Eigenschaft von MC setzt MC_7 in der dritten Spalte die Werte $(3b, 2b, b, b)$. Das AK_7 führt nun zu einem Muster, wie es in Abbildung 5.9 dargestellt ist. Wir erhalten aktive Bytes an den Positionen 9, 12, 13, 14 und 15. Dabei erhält das aktive Byte an Position 9 den Wert a, an Position 12 den Wert $3b$, an Positon 13 den Wert $b' = a \oplus 2b$ und an den Positionen 14 und 15 den Wert b. Wir bezeichnen das nach AK_7 entstandene

Muster mit δ. Es handelt sich hierbei um ein Muster, da wir nur die Positionen der aktiven Bytes kennen, nicht aber alle konkreten Ausprägungen.

5.3.4 Der Angriff

Der Angriff verläuft nach den folgenden Schritten.

1. Wähle zufällig $2^{84.5}$ Klartextpaare P_i, P_j und $2^{84.5}$ Klartextpaare O_i, O_j mit:
$$P_i \oplus P_j = \alpha = O_i \oplus O_j$$
2. Lasse das Quartett (P_i, P_j, O_i, O_j) mit den Schlüsseln K_a, K_b, K_c, K_d zu (C_i, C_j, D_i, D_j) verschlüsseln, so dass
$$E_{K_a}(P_i) = C_i,\ E_{K_b}(P_j) = C_j,\ E_{K_c}(O_i) = D_i \text{ und } E_{K_d}(O_j) = D_j$$
gilt.
3. Teste für alle i,j, ob $C_i \oplus D_i \in \Delta O$ und $C_j \oplus D_j \in \Delta O$ gilt.
4. Initialisiere einen Zähler für alle Kombinationen von 5 Bytes der achten Rundenschlüssel $K_{a_8}, K_{b_8}, K_{c_8}, K_{d_8}$ an den Positionen 3, 5, 6, 9 und 12.
5. Gehe über alle Kombinationen von 5 Bytes der achten Rundenschlüssel $K_{a_8}, K_{b_8}, K_{c_8}, K_{d_8}$ an den Positionen 3, 5, 6, 9 und 12.
   5.1. Prüfe für alle Quartette, die den Test in Schritt 3 bestanden haben, ob
   $$D_{K_{a_8}}(C_i) \oplus D_{K_{c_8}}(D_i) \in \delta \quad \text{und} \quad D_{K_{b_8}}(C_j) \oplus D_{K_{d_8}}(D_j) \in \delta$$
   gilt. Wenn ja, dann erhöhe den Zähler auf die verwendeten Schlüsselbits um eins. Hierbei ist $D_{K_8}(\cdot)$ die Entschlüsselung der achten Runde mit den gewählten Schlüsselbits des achten Rundenschlüssels.
6. Hat ein Zähler einen Wert von mindestens 3, so gib die dazugehörigen Schlüsselbits als korrekt aus.

Algorithmus 12: Related-Key Rechtecks-Angriff auf 8 Runden AES-192

Schritt 1: Für die Durchführung des Related-Key Rechtecks-Angriffs wählen wir $2^{84.5}$ Paare P_i, P_j und $2^{84.5}$ Paare O_i, O_j mit:

$$P_i \oplus P_j = \alpha = O_i \oplus O_j$$

Schritt 2: Ein Klartext-Quartett (P_i, P_j, O_i, O_j) wird mit den Schlüsseln K_a, K_b, K_c, K_d in E_0 zu (A_i, A_j, B_i, B_j) durch

$$E_{0,K_a}(P_i) = A_i,\ E_{0,K_b}(P_j) = A_j,\ E_{0,K_c}(O_i) = B_i \text{ und } E_{0,K_d}(O_j) = B_j$$

verschlüsselt. Wir zählen nun alle Differenzen, die nach AK_3 durch das aktive Byte mit Wert a an Position 9 und durch die vier aktiven Bytes an den Positionen 12, 13, 14 und 15 entstehen können. Da SR_4 und MC_4 linear sind, ändern diese nichts an der Anzahl der Differenzen, die als Ausgabe von E_0 entstehen. Mit dieser Zählung können wir die Wahrscheinlichkeit berechnen, dass

$$A_i \oplus A_j = B_i \oplus B_j \tag{5.1}$$

gilt. Diese ergibt sich als:

$$\Pr(A_i \oplus A_j = B_i \oplus B_j) = (2^{-32} \cdot 2^{-7})^2 \cdot (2^7 - 2) \cdot 2^{32} + (2^{-32} \cdot 2^{-6})^2 \cdot 2^{32} \approx 2^{-38.97}$$

Als Eingabedifferenz des Differentials für E_1 muss gelten:

$$A_i \oplus B_i = \gamma = A_j \oplus B_j \tag{5.2}$$

Nach Annahme 4.3.2 auf Seite 29 sind die Ausgaben von E_0 zufällig gleich verteilt. Damit erhalten wir die Wahrscheinlichkeit:

$$\Pr(A_i \oplus B_i = \gamma = A_j \oplus B_j \mid A_i \oplus A_j = B_i \oplus B_j) = 2^{-128}$$

Nun fordern wir, dass Bedingung (5.1) und (5.2) gleichzeitig gelten, um den Angriff erfolgreich durchführen zu können. Die Wahrscheinlichkeit dafür beträgt:

$$\Pr(A_i \oplus A_j = B_i \oplus B_j \text{ und } A_i \oplus B_i = \gamma = A_j \oplus B_j) = 2^{-38.97} \cdot 2^{-128} = 2^{-166.97}$$

Definition 5.3.3 *Ein korrektes Rechtecks-Quartett (P_i, P_j, O_i, O_j) erfüllt Bedingung (5.1), (5.2) und besteht den Test in Schritt 3. Alle Quartette, welche nur den Test in Schritt 3 bestehen, werden als falsche Rechtecks-Quartette bezeichnet.*

Da das Differential für E_1 die Wahrscheinlichkeit 1 besitzt, erhalten wir mit der Wahrscheinlichkeit $2^{-166.97}$ ein Quartett, welches das Muster δ nach E_1 aufweist. Damit der Angriff erfolgreich ist, darf eine zufällige Permutation keine höhere Wahrscheinlichkeit besitzen. Da δ ein Muster und somit eine Menge von möglichen Ausgabedifferenzen beschreibt, benötigen wir für die Berechnung der Wahrscheinlichkeit einer zufälligen Permutation noch einen Zwischenschritt. Sei $|\delta|$ die Anzahl der Differenzen, die sich aus dem Muster δ ergeben. Mit der Wahrscheinlichkeit $|\delta| \cdot 2^{-128}$ trifft ein Paar von Chiffretexten ein Element aus δ. Damit ist die Wahrscheinlichkeit, dass ein zufälliges Quartett zwei Paare aus δ besitzt genau:

$$2^{-256} \cdot |\delta|^2$$

Aufgrund des gewählten Musters ergibt sich $|\delta| = 2^7 - 1$. Im Muster δ befinden sich $2^7 - 1$ Kombinationen von Differenzen, da der Wert a bekannt ist und der Wert b einen von $2^7 - 1$ Werten annehmen kann. Damit liegt die Differenz eines zufälligen Quartetts mit der Wahrscheinlichkeit

$$2^{-256} \cdot (2^7 - 1)^2 \approx 2^{-242}$$

in δ. Der Angriff hat dadurch eine deutlich höhere Wahrscheinlichkeit, ein korrektes Rechtecks-Quartett zu finden als eine zufällige Permutation, da sich AES deutlich von einer zufälligen Chiffre unterscheidet.

Nachdem ein Quartett (P_i, P_j, O_i, O_j) das Differential für E_0 und E_1 durchlaufen hat, erfolgt die achte Runde nach der in Abbildung 5.10 dargestellten Weise. Hierbei führt SR_8 dazu, dass fünf aktive Bytes an die Positionen 3, 5, 6, 9 und 12 verschoben werden. AK_8 fügt zwei weitere aktive Bytes mit dem Wert a an die Positionen 1 und 13 hinzu, so dass ein Muster ΔO entsteht.

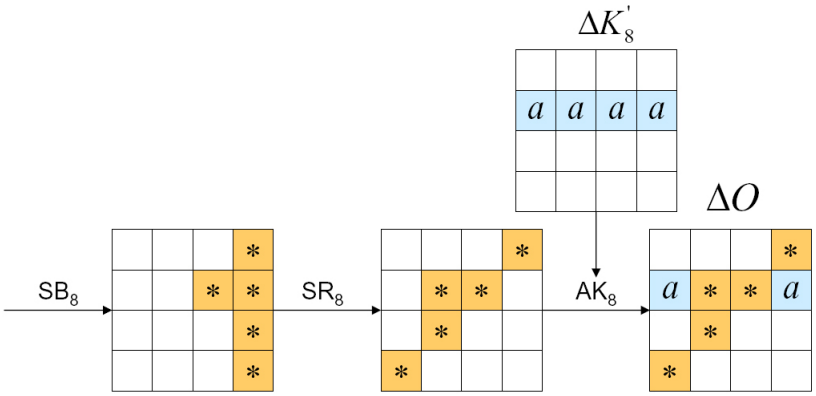


Abbildung 5.10: Runde 8 im Related-Key Rechtecks-Angriff

Schritt 3: Wir speichern uns alle Chiffretext-Quartette (C_i, C_j, D_i, D_j) für die

$$C_i \oplus D_i \in \Delta O \quad \text{und} \quad C_j \oplus D_j \in \Delta O$$

gilt. Mit anderen Worten: Wir speichern uns alle Quartette, die zwei Paare mit dem Muster ΔO besitzen. Da alle Elemente in ΔO genau fünf Bytes mit unbestimmten Werten und elf Bytes mit identischen Werten besitzen, beträgt die Wahrscheinlichkeit, dass ein falsches Quartett von Chiffretexten in ΔO liegt genau:

$$\Pr(\text{ein falsches Quartett in } \Delta O) \;=\; \left(\left(2^{-8}\right)^{11}\right)^2 = 2^{-176}$$

Da die Anzahl der Differenzen des Musters ΔO aus der Anzahl der Differenzen des Musters δ abgeleitet werden kann, entspricht die Wahrscheinlichkeit, dass ein korrektes Rechtecks-Quartett von Chiffretexten aus Elementen in ΔO besteht, genau der des Related-Key Rechtecks-Angriffs. Der Related-Key Rechtecks-Angriff dient hierbei als Unterscheiderangriff. Wie schon zuvor der Boomerang-Angriff, unterscheidet er zwischen zwei Mengen. Eine erste Menge, welche aus allen korrekten Rechtecks-Quartetten in ΔO und allen falschen Quartetten in ΔO besteht. Die zweite Menge vereint alle möglichen Quartette, die nicht in der ersten Menge liegen. Damit haben wir mit der Wahrscheinlichkeit $2^{-166.97}$ die erwartete Anzahl an korrekten Quartetten, deren dazugehörige Chiffretext-Quartette in ΔO liegen.

Aus je $2^{84.5}$ Paaren P_i, P_j und O_i, O_j können wir 2^{169} Quartette bilden. Wir erwarten

$$\begin{aligned} \#\,\text{Mengen} \cdot \Big(\Pr(\text{korrektes Quartett in } \Delta O) + \Pr(\text{falsches Quartett in } \Delta O)\Big) \;&= \\ 2^{169} \cdot \left(2^{-166.97} + 2^{-176}\right) \;&\approx\; 2^2 \end{aligned}$$

korrekte Rechtecks-Quartette. Somit erwarten wir, dass etwa vier korrekte Rechtecks-Quartette den Test in Schritt 3 bestehen.

Schritt 5:(Die Schlüsselsuche) Unser Ziel ist es nun, den eben beschriebenen Related-Key Rechtecks-Angriff zur Suche von 5 Bytes der achten Rundenschlüssel einzusetzen. Wir suchen die 5 Bytes der vier Rundenschlüssel, die in Abbildung 5.10 in der Menge ΔO mit $*$ gekennzeichnet sind. Diese Bytes befinden sich an den Positionen 3, 6, 9 und 12. Für die vier Rundenschlüssel $K_{a8}, K_{b8}, K_{c8}, K_{d8}$ gäbe es

$$\left(\left(2^8\right)^5\right)^4 = 2^{160}$$

mögliche Kombinationen 5 Bytes zu raten, währen die 4 Schlüssel unabhängig. Da es sich wegen der Annahme 5.3.1 auf Seite 47 um Related-Keys handelt, reduziert sich die Anzahl

der möglichen Kombinationen deutlich. Betrachten wir dazu die paarweisen Differenzen der Rundenschlüssel aus Runde 8:

$$\Delta K_8^* = K_{a_8} \oplus K_{b_8} = K_{c_8} \oplus K_{d_8} = \begin{array}{|c|c|c|c|}\hline & & b & b \\ \hline a & & & \\ \hline d & d & d & d \\ \hline c & & c & \\ \hline \end{array}$$

$$\Delta K_8' = K_{a_8} \oplus K_{c_8} = K_{b_8} \oplus K_{d_8} = \begin{array}{|c|c|c|c|}\hline & & & \\ \hline a & a & a & a \\ \hline & & & \\ \hline & & & \\ \hline \end{array}$$

Da jeweils zwei Schlüssel paarweise verbunden sind und die Differenzen in der oben dargestellten Weise auftreten, verbleiben genau

$$\#\text{ Schlüsselkombinationen} = \left(2^8\right)^5 \cdot \left(2^8\right)^2 \cdot (2^7 - 1) = 2^{56} \cdot (2^7 - 1) \approx 2^{63}$$

Möglichkeiten, die 5 gesuchten Bytes für vier verbundene Schlüssel zu belegen. Dies folgt daraus, dass wir für b genau $2^7 - 1$ mögliche Werte erhalten können und für c und d jeweils 2^8 Kombinationen existieren.

Schritt 5.1: Wir benutzen nun alle Quartette, welche den Test in Schritt 3 bestanden haben. Diese lassen wir mit den in Schritt 4 geratenen Schlüsselbits eine Runde entschlüsseln. Sei $D_{K_8}(\cdot)$ die Entschlüsselung der achten Runde mit den Schlüsselbits, welche in Schritt 4 geraten wurden. Wir testen, ob

$$D_{K_{a_8}}(C_i) \oplus D_{K_{c_8}}(D_i) \in \delta \quad \text{und} \quad D_{K_{b_8}}(C_j) \oplus D_{K_{d_8}}(D_j) \in \delta$$

gilt. Ist dies für ein Quartett der Fall, so erhöhen wir einen Zähler auf den verwendeten Schlüsselbits.

Schritt 6: Besitzt ein Zähler einen Wert von 3 oder größer, so werden die entsprechenden Schlüsselbits als korrekt ausgegeben.

5.3.5 Komplexität

Für die Schlüsselsuche benötigen wir

$$2^{63} \cdot 2^2 \cdot 2^2 \cdot \frac{1}{8} \cdot \frac{1}{2} = 2^{63}$$

8-Runden AES-192 Verschlüsselungen. Allerdings dominiert der zweite Schritt, also die Verschlüsselung der zwei Mengen von $2^{84.5}$ Klartextpaaren, die Laufzeit. Damit benötigen wir $2^{86.5}$ 8-Runden AES-192 Verschlüsselungen. Für die Schlüssel benötigen wir 5 Bytes für 4 Schlüssel und damit

$$4 \cdot 5 \cdot 2^{86.5} \approx 2^{90.83}$$

Bytes an Speicher. Dies entspricht $2^{60.83}$ Gigabyte. Die Wahrscheinlichkeit, dass für ein falsches Quartett von Schlüsselbits ein Chiffretext-Quartett den Test in Schritt 5.1 erfüllt, beträgt höchstens 2^{-6}. Dies folgt aus der Tatsache, dass der größte Wert in der Differential-Verteilungstabelle von AES den Wert 4 besitzt. Es tritt auf, falls die Bits von zwei der vier Rundenschlüssel richtig geraten wurden und die Bits der zwei übrigen Rundenschlüssel nur in jeweils einem Byte falsch sind. Die Ausgabe eines falsche Quartetts von Schlüsselbits in Schritt 6 ist in diesem Fall höchstens $(2^{-6})^3 = 2^{-18}$. Es existieren höchstens $2 \cdot 5 \cdot (2^8 - 1)$ falsche Quartette von Schlüsselbits. Die Wahrscheinlichkeit, dass diese falschen Schlüsselbits als korrekt ausgegeben werden beträgt:

$$2 \cdot 5 \cdot (2^8 - 1) \cdot 2^{-18} = 0.0097 \approx 0.01$$

Auf diese Weise können die Wahrscheinlichkeiten aller Kombinationen von falschen Schlüsselbits berechnet werden. Da die Wahrscheinlichkeiten der anderen Fälle jedoch viel kleiner als 0.01 sind, approximieren wir die Wahrscheinlichkeit, dass eine falsche Kombination von Schlüsselbits den Test in Schritt 5.1 besteht mit 0.01. Nun benötigen wir die Wahrscheinlichkeit, dass für ein korrektes Quartett von Schlüsselbits mindestens drei Chiffretext-Quartette den Test in Schritt 5.1 bestehen. Diese berechnet sich durch:

$$\sum_{i=3}^{2^{169}} \binom{2^{169}}{i} \cdot (2^{-166.97})^i \cdot (1 - 2^{-166.97})^{2^{169}-i} \approx 0.77$$

Die Erfolgsrate des Angriffs beträgt damit:

$$0.77 \cdot (1 - 0.01) \approx 0.76 = 76\%$$

Kapitel 6

Erweiterung der Angriffe

In diesem Kapitel beschreiben wir von uns entwickelte Angriffe, welche wir aus den Angriffen des vorherigen Kapitels abgeleitet haben. Die nachfolgende Übersicht zeigt einen Vergleich der von uns entwickelten Angriffe mit bekannten Angriffen.

Angriff	Daten / Verschlüsselungen	Quelle
Square	2^{32} / 2^{184}	[33]
Partial Sums	$19 \cdot 2^{32}$ / 2^{155}	[22]
Partial Sums	$2^{128} - 2^{119}$ / 2^{120}	[22]
Related-Key Impossible	2^{111} / 2^{116}	[25]
Related-Key Impossible	2^{56} / 2^{94}	[7]
Boomerang	$2^{180.5}$ / $2^{180.5}$	neu
Related-Key Boomerang	2^{83} / 2^{83}	neu

Tabelle 6.1: Vergleich mit bestehenden Angriffen auf 7-Runden AES-192

6.1 Boomerang-Angriff auf 7-Runden AES-192

Nun beschreiben wir eine von uns entwickelte Erweiterung des Boomerang-Angriffs auf 5-Runden AES-128 von Biryukov [10]. Diese Erweiterung kann eine auf 7 Runden reduzierte Variante von AES-192 brechen. Für diesen Angriff benutzen wir das Differential E_0 aus Abschnitt 5.1.1.1 auf Seite 37 und das Differential E_0^{-1} aus Abschnitt 5.2.1 auf Seite 43. Wir werden auf diese beiden Differentiale in diesem Kapitel nicht noch einmal eingehen, sondern nur kurz deren Wahrscheinlichkeiten nennen. Der Ablauf des Angriffs bleibt dabei derselbe. Auch die bereits getroffenen Annahmen behalten weiterhin ihre Gültigkeit.

Die Wahrscheinlichkeit des Differentials E_0 aus einem α Muster ein β Muster zu erzeugen beträgt:

$$\Pr(\alpha \rightarrow \beta) = 2^{-22}$$

Das Differential E_0^{-1} hat eine Wahrscheinlichkeit von

$$\Pr(\alpha' \leftarrow \beta) = 2^{-13.5}$$

aus einem β Muster ein α' Muster zu erzeugen.

6.1.1 Das Differential E_1^{-1}

Wir werden nun das Differential für E_1^{-1} beschreiben, welches uns zusammen mit den Differentialen E_0 und für E_0^{-1} einen Angriff auf 7-Runden AES-192 ermöglicht. Abbildung 6.1 zeigt E_1^{-1} schematisch.

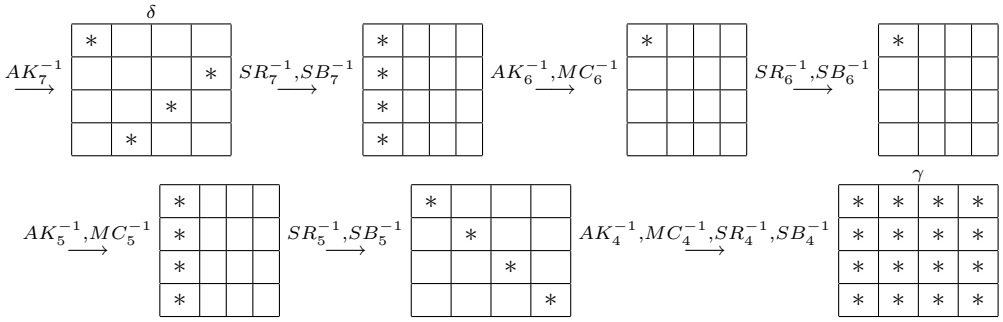


Abbildung 6.1: E_1^{-1} der Runden 7 bis 4 im Boomerang-Angriff auf 7-Runden AES-192

Wir generieren ein Muster δ mit vier aktiven Bytes an den Positionen 0, 7, 10 und 13. AK_7^{-1} ändert nichts an der Anzahl der aktiven Bytes. Allerdings führt SR_7^{-1} dazu, dass alle vier aktiven Bytes in die nullte Spalte verschoben werden. MC_6^{-1} generiert ein aktives Byte in der nullten Spalte mit der Wahrscheinlichkeit 2^{-22}, da hier die genaue Position des aktiven Bytes irrelevant ist. Dieses wird sich nach SR_6^{-1} abhängig von dessen Position in einer der vier Spalten befinden. Die Variante, welche sich in Abbildung 6.1 befindet, stellt nur eine von vier Möglichkeiten dar. Aus diesem aktiven Byte entstehen mit der Wahrscheinlichkeit 1 vier aktive Bytes durch MC_5^{-1}, welche sich nach SR_5^{-1} auf je eine Spalte verteilen. Im nächsten Schritt sorgt MC_4^{-1} dafür, dass alle 16 Bytes aktiviert werden. Auch dies geschieht mit der Wahrscheinlichkeit 1, da die Wahrscheinlichkeit, dass ein aktives Byte einer Spalte zu vier aktiven Bytes wird, genau 1 beträgt.

Das Differential E_1^{-1} erzeugt aus einem Muster δ ein γ Muster mit der Wahrscheinlichkeit

$$\Pr(\gamma \leftarrow \delta) = 2^{-22}.$$

6.1.2 Die Analyse

Mit dem Wissen des vorherigen Kapitels können wir die Wahrscheinlichkeit für ein korrektes Boomerang-Quartett berechnen. Für ein solches Quartett müssen wir sicherstellen, dass das Muster γ von zwei Textpaaren identisch ist. Siehe dazu Abschnitt 5.1.3 auf Seite 41. Die Wahrscheinlichkeit dafür beträgt:

$$\Pr(\gamma_{A_i \oplus B_i} = \gamma_{A_j \oplus B_j}) = (2^{-8})^{16} = 2^{-128}$$

Aus einem Klartextpaar mit Muster α erhalten wir somit ein korrektes Boomerang Quartett mit der Wahrscheinlichkeit:

$$\begin{aligned} \Pr(\text{ein korrektes Boomerang Quartett}) &= \\ \Pr(\alpha \rightarrow \beta) \cdot \Pr(\gamma \leftarrow \delta)^2 \cdot \Pr(\gamma_{A_i \oplus B_i} = \gamma_{A_j \oplus B_j}) \cdot \Pr(\alpha' \leftarrow \beta) &= 2^{-22} \cdot (2^{-22})^2 \cdot 2^{-128} \cdot 2^{-13.5} \\ &= 2^{-207.5} \end{aligned}$$

Hier sei noch einmal der Hinweis erwähnt, dass ein korrektes Boomerang-Quartett alle verwendeten Differentiale erfüllt. Aus $2^{147.5}$ Mengen von 2^{63} Klartextpaaren erhalten wir

$$\begin{aligned} \#Mengen \cdot \#Quartette \cdot \Pr(\text{ein korrektes Boomerang Quartett}) &= \\ 2^{147.5} \cdot 2^{63} \cdot 2^{-207.5} &= 8 \end{aligned}$$

korrekte Boomerang Quartette. Die Schlüsselsuche verläuft analog zum Boomerang-Angriff auf 5-Runden AES-128. Wir benötigen für den Angriff $2^{147.5} \cdot 2^{32} = 2^{179.5}$ 7-Runden AES-192 Verschlüsselungen und ebenso viele Entschlüsselungen. Dies führt zu einem gesamten

Zeitaufwand von $2^{180.5}$ 7-Runden Verschlüsselungen. Der Angriff benötigt 2^{33} Blöcke, welches 2^{37} Bytes entspricht.

Der verbesserte Boomerang-Angriff auf 7-Runden AES-192 benötigt $2^{180.5}$ 7-Runden Verschlüsselungen und 2^{37} Bytes an Speicher.

6.2 Related-Key Boomerang-Angriff auf 7-Runden AES-192

Wir beschreiben in diesem Kapitel den Boomerang-Angriff, der um Related-Keys von uns erweitert wurden ist. Die Idee ist in der Literatur in [43, 6, 10, 1, 27] zu finden, wurde jedoch bisher noch nicht auf AES angewendet. Wir werden uns im folgenden Abschnitt mit der Struktur der Schlüssel beschäftigen und anschließend die verwendeten Differentiale beschreiben. Danach analysieren wir den gesamten Angriff. Abbildung 6.2 verdeutlicht den Angriff sowie die verwendeten Schlüsseldifferenzen. Wir teilen erneut die Chiffre in zwei unabhängige Teilchiffren auf. Die Chiffre E_0 geht dabei über die Runden $1-5$, während die Chiffre E_1 über die Runden $6-7$ verläuft. Der Ablauf lässt sich wie folgt kurz zusammenfassen. Die Klartexte P_i, P_j mit dem Muster α werden mit K_a und K_b zu den Chiffretexten C_i, C_j verschlüsselt. Es wird ein Muster δ ausgewählt, so dass zwei neue Chiffretexte D_i, D_j nach $C_i \oplus \delta$ und $C_j \oplus \delta$ entstehen. Die neuen Chiffretexte D_i, D_j werden mit den Schlüsseln K_c, K_d zu den Klartexten O_i, O_j entschlüsselt. Die Klartexte O_i, O_j sollen nun das Muster α mit einer gewissen Wahrscheinlichkeit besitzen. Wie dies genau zustande kommt, werden wir später im Detail besprechen.

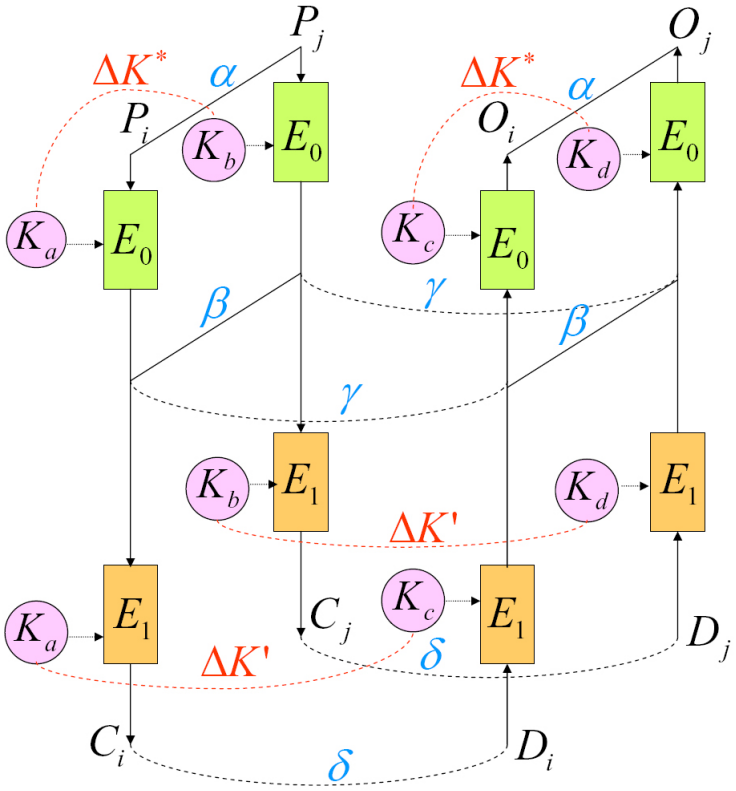


Abbildung 6.2: Übersicht des Related-Key Boomerang-Angriffs

6.2.1 Die Related-Keys

Wie aus Abbildung 6.2 ersichtlich wird, verwenden wir für diesen Angriff vier Schlüssel K_a, K_b, K_c und K_d, deren Werte wir nicht kennen. Wir kennen jedoch die folgenden

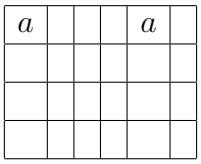
Relationen zwischen den Schlüsseln:

$$\begin{aligned} K_a \oplus K_b &= \Delta K^* \\ K_a \oplus K_c &= \Delta K' \\ K_b \oplus K_d &= \Delta K' \end{aligned}$$

Daraus folgt:

$$\begin{aligned} (K_a \oplus K_c) \oplus (K_a \oplus K_b) \oplus (K_b \oplus K_d) &= \\ \Delta K' \oplus \Delta K^* \oplus \Delta K' &= \\ \Delta K^* &= K_a \oplus K_c \end{aligned}$$

Wir nehmen an, dass wir die folgenden Schlüsseldifferenzen kennen.

$\Delta K^* =$  und $\Delta K' =$ 

Hieraus lassen sich die Differenzen der Rundenschlüssel ableiten. Diese sind in Abbildung 6.3 und 6.4 dargestellt. Da die Erzeugung der Differenzen der Rundenschlüssel analog zu dem Verfahren in Abschnitt 5.3.1 auf Seite 47 des Related-Key Rechtecks-Angriff ist, verweisen wir an dieser Stelle auf diesen.

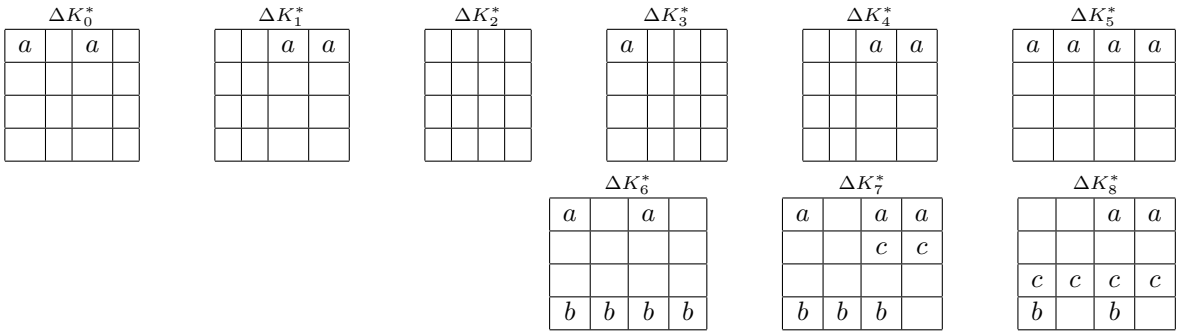


Abbildung 6.3: Die Differenzen der Rundenschlüssel der Schlüsseldifferenz ΔK^*

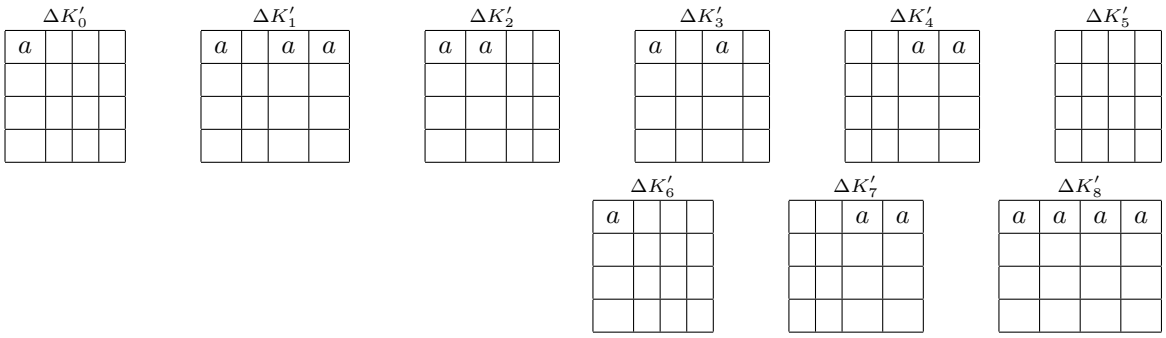


Abbildung 6.4: Die Differenzen der Rundenschlüssel der Schlüsseldifferenz $\Delta K'$

6.2.2 Das Differential E_0

Das Differential E_0 ist in Abbildung 6.5 dargestellt.

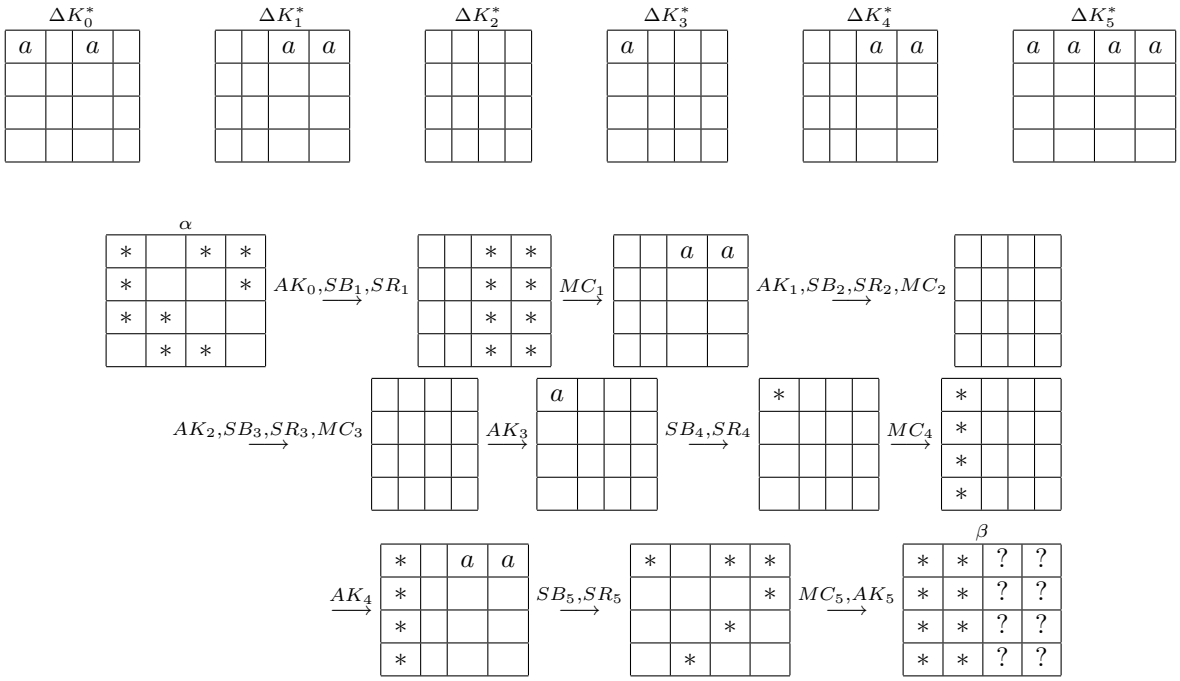


Abbildung 6.5: Das Differential E_0 der Runden 1-5

Zwei Klartexte P_i und P_j mit Muster α werden zu den Chiffretexten C_i und C_j verschlüsselt. Dabei wird P_i mit dem Schlüssel K_a und P_j mit dem Schlüssel K_b verschlüsselt. Es gilt somit

$$E_{K_a}(P_i) = C_i \quad \text{und} \quad E_{K_b}(P_j) = C_j \quad \text{mit } K_a \oplus K_b = \Delta K^*.$$

Das Muster α der Klartexte beinhaltet 9 aktive Bytes an den Positionen 0, 1, 2, 6, 7, 8, 11, 12, und 13. Bei einer zufälligen Wahl der beiden Klartexte mit Muster α sind auch die Werte der aktiven Bytes zufällig verteilt. Das heißt, dass von $2^8 - 1$ möglichen Werten ein aktives Bytes zu erzeugen, jede dieser möglichen Werte mit einer Wahrscheinlichkeit von 2^{-8} auftritt. Zur Vereinfachung schreiben wir allerdings nur $\frac{1}{2^8}$. Damit hat das nullte Byte, bei einer zufälligen Wahl der Klartexte mit Muster α, mit der Wahrscheinlichkeit

$$\Pr(\text{aktives Byte hat Differenz } a) = 2^{-8}$$

eine Differenz a. Wir erhalten mit dieser Wahrscheinlichkeit nach AK_0 eine Null-Differenz in Byte 0. SR_1 verschiebt die verbleibenden acht aktiven Bytes komplett in die Spalten 2 und 3. MC erzeugt aus vier aktiven Bytes ein aktives Byte mit fester Position mit der Wahrscheinlichkeit:

$$\Pr(\text{4 aktive Bytes in ein aktives mit fester Position}) = \frac{2^8 - 1}{2^{32}} \approx 2^{-24}$$

Fixieren wir zusätzlich den Wert des erhaltenen aktiven Bytes, so sinkt die Wahrscheinlichkeit auf:

$$\Pr(\text{4 aktive Bytes in ein aktives mit festem Wert und fester Position}) = 2^{-32}$$

Nach MC_1 erhalten wir damit mit einer Wahrscheinlichkeit von 2^{-64} zwei aktive Bytes an den Positionen 8 und 12 mit dem Wert a. Da die Differenz des ersten Rundenschlüssels ΔK^* ebenfalls zwei aktive Bytes mit dem Wert a an den Positionen 8 und 12 besitzt, gibt es in dem Muster eines Textpaares nach AK_1 keine aktiven Bytes mehr. Die anschließenden Operationen SB_2 bis MC_3 führen dadurch keine Änderung herbei. Zu beachten ist

hierbei, dass die Differenz ΔK_2^* keine aktiven Bytes besitzt und AK_2 somit keine Veränderung im Muster der Textpaare bewirkt. Erst AK_3 bringt ein aktives Byte mit dem Wert a an die Position 0 des Musters. Diese wird zu vier aktiven Bytes durch MC_4 mit der Wahrscheinlichkeit 1. Von diesem Punkt an lassen wir unser Muster frei laufen und erhalten nach AK_5 ein Muster β, welches in den Spalten 0 und 1 jeweils vier aktive Bytes besitzt. Die Wahrscheinlichkeit dafür beträgt 1, da sich vor MC_5 in diesen beiden Spalten nur ein aktives Byte befunden hat. Über die Bytes in den Spalten 2 und 3 machen wir uns keine Gedanken, da sie für die spätere Analyse keine Rolle spielen. Deshalb kennzeichnen wir diese beide Spalten mit ?, wie es in Abbildung 6.5 dargestellt ist. Damit beträgt die Wahrscheinlichkeit des Differentials für E_0:

$$\Pr(\alpha \to \beta) = 2^{-8} \cdot 2^{-64} = 2^{-72}$$

Ein Klartextpaar, welches ein Muster α aufweist, besitzt mit einer Wahrscheinlichkeit von

$$\Pr(\alpha \to \beta) = 2^{-72}$$

ein Muster β nach E_0.

6.2.3 Das Differential E_1^{-1}

Abbildung 6.6 stellt dieses Differential zusammen mit den verwendeten Schlüsseldifferenzen dar.

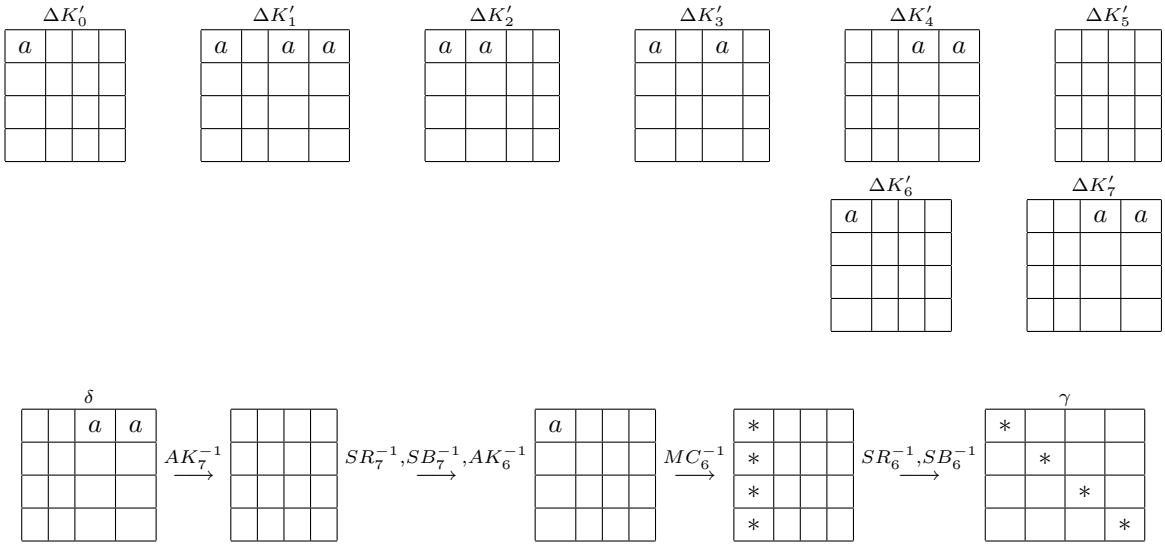


Abbildung 6.6: Das Differential E_1^{-1} der Runden 7-6

Wie wir wissen sind die Chiffretexte C_i und C_j aus der Verschlüsselung der Klartexte P_i und P_j mit den Schlüsseln K_a und K_b entstanden. Aus C_i und C_j werden nun mit $D_i = C_i \oplus \delta$ und $D_j = C_j \oplus \delta$ zwei neue Chiffretexte generiert, welche mit K_c und K_d entschlüsselt werden. Wir erhalten aus

$$E_{K_c}^{-1}(D_i) = O_i \quad \text{und} \quad E_{K_d}^{-1}(D_j) = O_j$$

die neuen Klartexte O_i und O_j, wobei E^{-1} die Entschlüsselung darstellt.

Nun wollen wir die Wahrscheinlichkeit des Differentials für E_1^{-1} berechnen. Sei δ ein Muster, welches zwei aktive Bytes mit dem Wert a an den Positionen 8 und 12 besitzt.

Wir lassen zwei Paare von Chiffretexten (C_i, D_i) und (C_j, D_j) mit Muster δ und Schlüsseldifferenz

$$K_a \oplus K_c = \Delta K' = K_b \oplus K_d$$

durch das Differential E_1^{-1} laufen. Die zwei aktiven Bytes des δ Musters verschwinden nach AK_7^{-1}, da $\Delta K_7'$ ebenfalls zwei aktive Bytes mit dem Wert a an den Positionen 8 und 12 besitzt. SR_7^{-1} und SB_7^{-1} führen zu keiner Veränderung in dem Muster, welches keine aktiven Bytes mehr besitzt. Erst AK_6^{-1} erzeugt ein aktives Byte an der Position 0, welche zu vier aktiven Bytes nach MC_6^{-1} mit der Wahrscheinlichkeit 1 transformiert wird. SR_6^{-1} sorgt dafür, dass die vier aktiven Bytes auf die Positionen 0, 5, 10 und 15 verteilt werden, womit wir schließlich nach SB_6^{-1} ein Muster γ mit vier aktiven Bytes erhalten. Ein δ Muster in der eben beschriebenen Form durchläuft das Differential für E_1^{-1} mit der Wahrscheinlichkeit 1.

Eine Muster δ wird mit der Wahrscheinlichkeit

$$\Pr(\gamma \leftarrow \delta) = 1$$

in ein γ Muster durch das Differential für E_1^{-1} transformiert.

6.2.4 Das Differential E_0^{-1}

Das Differential E_0^{-1} ist in Abbildung 6.7 schematisiert. Wir verwenden für dieses Differential dieselbe Schlüsseldifferenz ΔK^*, welche wir schon für E_0 verwendet haben. Wie

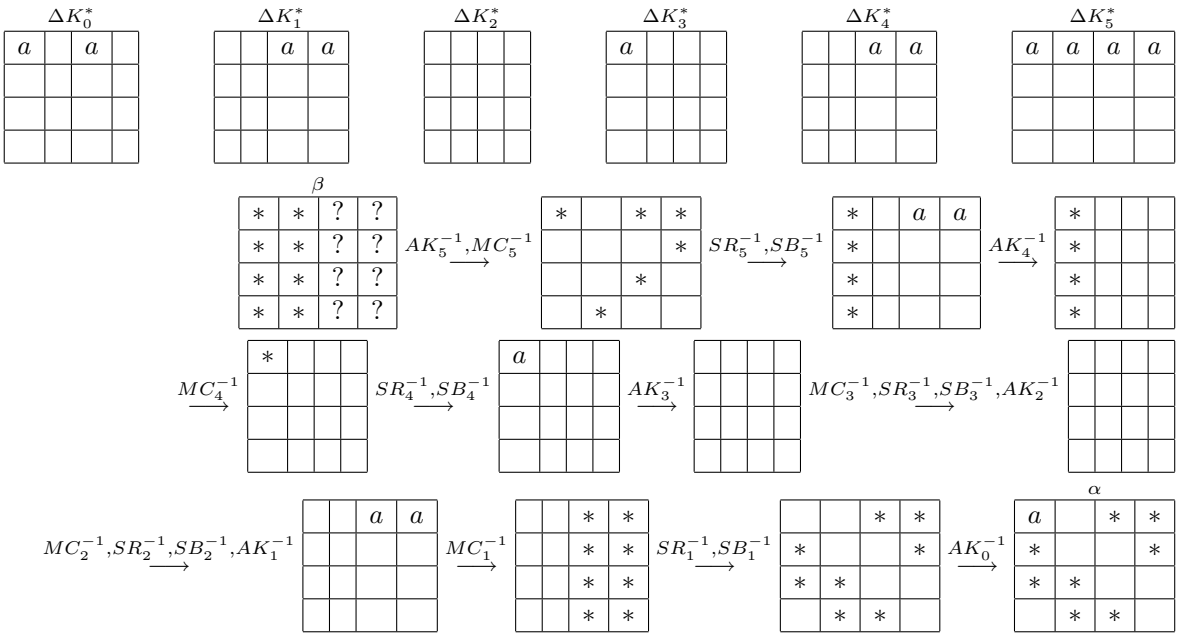


Abbildung 6.7: Das Differential E_0^{-1} der Runden 1-5

wir bisher in allen Ausführungen des Boomerang-Angriffs gesehen haben, können wir sicherstellen, dass die Werte der aktiven Bytes im Muster β bei der Ausgabe nach E_0 gleich den Werten der aktiven Bytes im Muster β bei der Eingabe von E_0^{-1} sind. Damit lässt sich das Muster, welches in E_0 vor MC_5 bestand, mit der Wahrscheinlichkeit 1 nach MC_5^{-1} in E_0^{-1} erzeugen. Dieses führt uns zu sechs aktiven Bytes an den Positionen 0, 7, 8, 10, 12 und 13 nach MC_5^{-1}. SR_5^{-1} verschiebt vier dieser aktiven Bytes in die Spalte 0, wobei die zwei aktiven Bytes an den Positionen 8 und 12 bestehen bleiben. Diese werden durch

SB_5^{-1} in einen Wert a transformiert, was mit einer Wahrscheinlichkeit von $(2^{-8})^2 = 2^{-16}$ geschieht.

Da ΔK_4^* ein Wert a der aktiven Bytes an den Positionen 8 und 12 besitzt, verbleiben nach AK_4^{-1} vier aktive Bytes in der nullten Spalte. MC_4^{-1} generiert aus diesen ein aktives Byte an Position 0 mit der Wahrscheinlichkeit 2^{-24}.[1] Mit einer Wahrscheinlichkeit von 2^{-8} erhalten wir einen Wert a im aktiven Byte an der Position 0 nach SB_4^{-1}. Diese verschwindet nach Anwendung von AK_3^{-1}, da ΔK_3^* ebenfalls ein aktives Byte mit dem Wert a an Position 0 besitzt. Erst AK_1^{-1} bringt zwei aktive Bytes an den Positionen 8 und 12 in das Muster. MC_1^{-1} macht aus diesen acht aktiven Bytes, die sich auf die Spalten 2 und 3 verteilen. Nach SR_1^{-1} haben wir acht aktive Bytes an den Positionen 1, 2, 6, 7, 8, 11, 12 und 13. Das abschließende AK_0^{-1} erzeugt noch zusätzlich ein aktives Byte im Byte 0. Damit haben wir das Muster α mit einer Wahrscheinlichkeit von $\Pr(\alpha \leftarrow \beta) = 2^{-16} \cdot 2^{-24} \cdot 2^{-8} = 2^{-48}$ erhalten.

Das Differential für E_0^{-1} hat eine Wahrscheinlichkeit von

$$\Pr(\alpha \leftarrow \beta) = 2^{-48}$$

ein Muster β in ein Muster α zu wandeln.

6.2.5 Der Angriff

Nachdem wir die Wahrscheinlichkeiten der Differentiale berechnet haben, können wir den Related-Key Boomerang-Angriff auf 7-Runden AES-192 vollständig beschreiben. Zunächst geben wir einen kurzen Überblick des Angriffs, den wir anschließend ausführlich erläutern. Zur Übersicht zeigen wir noch einmal den schematischen Ablauf.

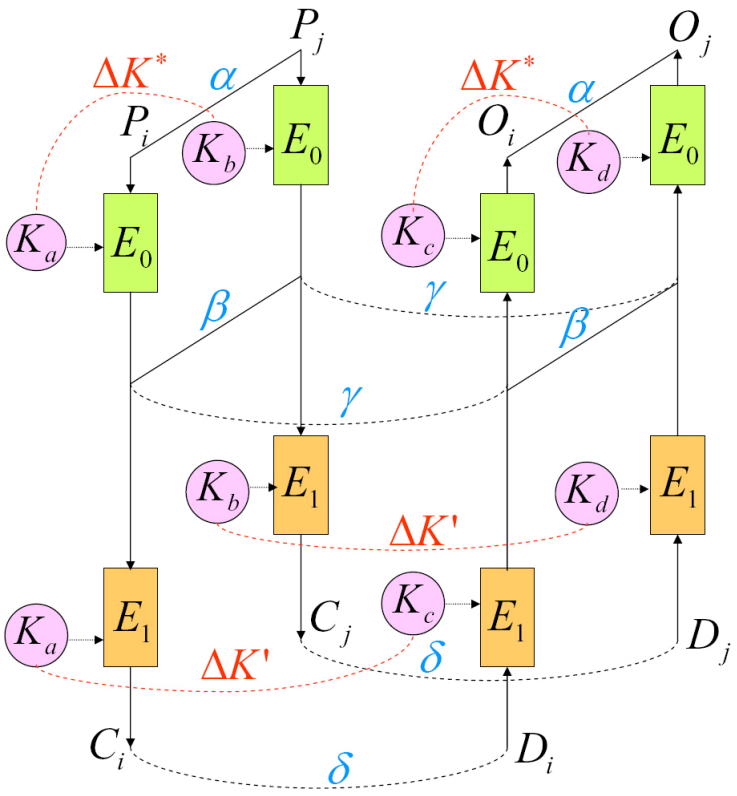


Abbildung 6.8: Übersicht des Related-Key Boomerang-Angriffs

[1] Hierbei haben wir die Position auf Byte 0 fixiert, weswegen die Wahrscheinlichkeit nicht 2^{-22} beträgt.

1. Erzeuge 2^{72} Klartexte $\{P_i\}$, die in den Bytes 0, 1, 2, 6, 7, 8, 11, 12 und 13 alle möglichen Kombinationen annehmen und in den übrigen Bytes denselben Wert haben.
2. Lasse $\{P_i\}$ zu den Chiffretexten $\{C_i\}$ verschlüsseln.
3. Erzeuge ein Muster δ, welches zwei aktive Bytes mit dem Wert a in den Bytes 8 und 12 besitzt.
4. Berechne die neuen Chiffretexte $\{D_i\}$ durch $D_i = C_i \oplus \delta$.
5. Lasse $\{D_i\}$ zu den Klartexten $\{O_i\}$ entschlüsseln.
6. Filtere alle Quartetts (P_i, P_j, O_i, O_j), bei denen die Klartextpaare P_i, P_j und O_i, O_j 9 aktive Bytes an den Positionen 0, 1, 2, 6, 7, 8, 11, 12 und 13 besitzen.
7. Initialisiere einen Zähler für jede Kombination von 4 Schlüsselbits der initialen Rundenschlüssel in den Bytes 1, 6, 11 und 12.
8. Gehe über alle Kombinationen von 4 Schlüsselbits der initialen Rundenschlüssel für die Bytes 1, 6, 11 und 12.
   - 8.1. Benutze die Quartette, welche den Test in Schritt 6 bestanden haben, um deren Klartexte eine Runde mit den gewählten Schlüsselbits zu verschlüsseln.
   - 8.2. Haben P_i, P_j und O_i, O_j ein aktives Byte mit dem Wert a nach MC_1 in dem Bytes 12, so erhöhe einen Zähler auf die benutze Kombination der Schlüsselbits um eins.
9. Werden für eine Kombination von Schlüsselbits mindestens zwei Quartette gezählt, die den Test in Schritt 8.2 bestanden haben, so gebe die geratenen Schlüsselbits als korrekt aus.

Algorithmus 13: Related-Key Boomerang-Angriff auf 7-Runden AES-192

Schritt 1: Aus 2^{72} Klartexten können wir

$$\binom{2^{72}}{2} = \frac{(2^{72})^2}{2} - 2^{71}$$

Klartextpaare mit Muster α bilden. Zur Vereinfachung rechnen wir mit $\frac{(2^{72})^2}{2} = 2^{143}$ Klartextpaaren. Eines dieser Paare hat nach E_0 mit der Wahrscheinlichkeit 2^{-72} ein Muster β.
Schritt 2-5: Auf der anderen Seite des Boomerangs transformiert E_1^{-1} ein Muster δ mit der Wahrscheinlichkeit 1 in ein γ Muster. Damit wir ein passendes Muster als Eingabe für E_0^{-1} erzeugen können, müssen wir sicherstellen, dass zwei Chiffretextpaare C_i, D_i und C_j, D_j mit Muster δ nach E_1^{-1} die gleichen Werte in den aktiven Bytes des γ Musters aufweisen. Da γ vier aktive Bytes an den Positionen 0, 5, 10 und 15 besitzt, geschieht dies mit der Wahrscheinlichkeit:

$$\Pr\left(E^{-1}_{1_{K_a}}(C_i) \oplus E^{-1}_{1_{K_c}}(D_i) = E^{-1}_{1_{K_b}}(C_j) \oplus E^{-1}_{1_{K_d}}(D_j)\right) = 2^{-32}$$

Damit haben wir die Bedingung erzeugt, mit der die Werte der aktiven Bytes im Muster der Ausgabe von E_0 den Werten der aktiven Bytes im Muster der Eingabe von E_0^{-1} entsprechen. E_0^{-1} wandelt in diesem Fall ein Muster β mit der Wahrscheinlichkeit 2^{-48} in ein Muster α.

Wir können jetzt die Wahrscheinlichkeit für ein korrektes Boomerang-Quartett, welches

aus einen Klartextpaar (P_i, P_j) erzeugt wurde, aufstellen und erhalten:

$$\Pr(\alpha \to \beta) \cdot \Pr(\gamma \leftarrow \delta) \cdot \Pr\left(E^{-1}_{1_{K_a}}(C_i) \oplus E^{-1}_{1_{K_c}}(D_i) = E^{-1}_{1_{K_b}}(C_j) \oplus E^{-1}_{1_{K_d}}(D_j)\right) \cdot \Pr(\alpha \leftarrow \beta)$$
$$= 2^{-72} \cdot 1^2 \cdot 2^{-32} \cdot 2^{-48} = 2^{-152}$$

Schritt 8: Wir gehen über alle möglichen Kombinationen von 4 Bytes der initialen Rundenschlüssel an den Positionen 1, 6, 11 und 12. Es gibt genau

$$\left(2^8\right)^4 = 2^{32}$$

Kombinationen die Bytes zu besetzen.
Schritt 8.1: Alle Quartette, welche den Text in Schritt 6 bestanden haben, werden benutzt um zu prüfen, ob diese aus zwei Paaren P_i, P_j und O_i, O_j bestehen, die bei den in Schritt 8 gewählten Schlüsselbits nach MC_1 im Byte 12 eine a Differenz besitzen. Für jedes dieser Quartette erhöhen wir einen Zähler auf den verwendeten Schlüsselbits um eins. Werden für eine Schlüsselbitkombination mindestens zwei Quartette gezählt, welche den Test in Schritt 8.2 bestehen, so wird diese in Schritt 9 ausgegeben.

Die Wahrscheinlichkeit für ein korrektes Boomerang-Quartett beträgt

$$\Pr(\text{ein korrektes Boomerang-Quartett}) = 2^{-152}.$$

6.2.6 Die Analyse

Aus 2^{10} Menge von 2^{143} Klartextpaaren lassen sich

$$\#\text{ Mengen} \cdot \#\text{ Quartette} \cdot \Pr(\text{ein korrektes Boomerang-Quartett}) = 2^{10} \cdot 2^{143} \cdot 2^{-152} = 2$$

korrekte Boomerang-Quartette bilden. Die Wahrscheinlichkeit eines falschen Boomerang-Quartetts ergibt sich als $(2^{-8})^7 = 2^{-56}$. Dies ist genau dann der Fall, wenn die Bytes 3, 4, 5, 9, 10, 14 und 15 zufällig passiv sind. Die erwarte Menge an falschen Boomerang-Quartetten beträgt damit

$$\#\text{ Mengen} \cdot \#\text{ Quartette} \cdot \Pr(\text{ein falsches Boomerang-Quartett}) = 2^{10} \cdot 2^{143} \cdot 2^{-56} = 2^{97}.$$

Da MC_1 mit der Wahrscheinlichkeit 2^{-24} eine a Differenz erzeugt, können wir dies als 24 Bit Filterung bezeichnen. Dieses tritt auf beiden Seiten des Boomerangs auf, womit wir insgesamt eine 48 Bit Filterung haben. Wir erwarten also, dass von 2^{32} möglichen Schlüsselbitkombinationen

$$2^{32} \cdot 2^{-48} = 2^{-16}$$

falsche Kombinationen den Test in Schritt 8.2 bestehen. Damit erhalten wir die korrekten Schlüsselbits mit einer Wahrscheinlichkeit von

$$1 - 2^{-16}.$$

Für die Suche nach den korrekten Boomerang-Quartetten benötigen wir $2^{82} (= 2^{10} \cdot 2^{72})$ 7-Runden AES-192 Verschlüsselungen und ebenso viele Entschlüsselungen, welches zu einem Gesamtaufwand von 2^{83} 7-Runden Verschlüsselungen führt. Da wir 4 Bytes für die 4 Schlüssel suchen und diese Bytes in allen Schlüsseln die selben Werte annehmen, benötigen wir $2^{74} (= 4 \cdot 2^{72})$ Blöcke oder 2^{78} Bytes an Speicher.

Um die Laufzeit der Schlüsselsuche zu berechnen, haben wir zu beachten, dass wir 2^{32} Schlüsselkanditaten an 2^{97} Quartetten testen müssen. Wir müssen somit etwa $2 \cdot 2^{49}$ Klartexte eine Runde 2^{32} mal verschlüsseln. Dies führt uns zu einen Aufwand von

$$\frac{1}{7} \cdot 2^{50} \cdot 2^{32} \approx 2^{79.5}$$

7-Runden AES-192 Verschlüsselungen, allein für die Schlüsselsuche. Die Komplexität des Angriffs wird durch die Suche nach den korrekten Boomerang-Quartetten determiniert.

Der Angriff könnte verbessert werden, wenn es möglich wäre, die Wahrscheinlichkeiten der Differentiale zu erhöhen. Je mehr passive Bytes sich im Muster α befinden, desto geringer wird die Wahrscheinlichkeit für ein falsches Boomerang-Quartett. Dadurch verringert sich die Menge der bei der Schlüsselsuche zu überprüfenden Quartette.

Der Related-Key Boomerang-Angriff findet 4 Bytes der initialen Rundenschlüssel mit einer Erfolgswahrscheinlichkeit von $1 - 2^{-16}$. Dabei benötigt er 2^{83} 7-Runden AES-192 Verschlüsselungen und einen Speicherbedarf von 2^{74} Blöcke, welches 2^{78} Bytes entspricht.

6.3 Related-Key Boomerang-Angriff auf 8-Runden AES-192

Wir können den Angriff des vorherigen Kapitels leicht auf 8 Runden erweitern, indem wir 8 Bytes der Schlüsselbits von AK_8 in der zweiten und dritten Spalte raten. Da wir Related-Keys betrachten ergeben sich insgesamt

$$\#\text{ Schlüsselkombinationen} = 2^{64} \cdot (2^8)^3 = 2^{88}$$

mögliche Schlüsselbits. Der Gesamtaufwand des Angriffs steigt dadurch auf

$$\#\text{ Schlüsselkombinationen} \cdot \#\text{ 7-Runden AES-192 Verschlüsselungen} = 2^{88} \cdot 2^{83} = 2^{171}$$

8-Runden AES-192 Verschlüsselungen und der Speicherbedarf von 2^{80} auf $2^{168} (= 2^{88} \cdot 2^{80})$ Bytes.

Kapitel 7

Zusammenfassung und Ausblicke

In der vorliegenden Arbeit haben wir uns ausführlich mit der Sicherheit von AES beschäftigt. Dabei wurde gezeigt wie Varianten der differentiellen Kryptoanalyse auf reduzierte Versionen von AES angewendet werden können, um Teile von Schlüsselbits schneller als bei einer vollständigen Suche berechnen zu können. Boomerang- und Rechtecks-Angriffe zählen heutzutage zu den stärksten bekannten Angriffen auf AES. In Verbindung mit Related-Keys lassen sich diese Angriffe erheblich verbessern. Nach einer ausführlichen Darstellung der Boomerang- und Rechtecksangriffe in Verbindung mit Related-Keys haben wir einen neuen effizienten Related-Key Boomerang-Angriff auf AES-192 entwickelt. Dieser 7-Runden Angriff verbessert die benötigten Verschlüsselungen des schnellsten bekannten 7-Runden Angriffs um einen Faktor 2^{11}. Sehr gute Ergebnisse haben wir ebenfalls bei der Erweiterung dieses Angriffs auf 8-Runden erzielt, bei dem 2^{83} Klartexte verwendet werden. Eine 9-Runden Variante ist mit diesem Angriff nicht möglich, könnte aber bei der Verwendung von unwahrscheinlichen Differentialen [2] erzielt werden.

Wenn wir von einem in der Theorie schnellen und effizienten Angriff sprechen, so muss dieses in der Praxis nicht zwangsläufig gelten. Betrachten wir den von uns vorgestellten Related-Key Boomerang-Angriff auf 7-Runden AES-192, der 2^{83} Verschlüsselungen und 2^{78} Byte Speicher benötigt. In der Praxis ist es heutzutage maximal möglich etwa 2^{60} Verschlüsselungen durchzuführen. Der Speicherbedarf von 2^{48} Gigabyte lässt sich mit der heutigen Technik ebenfalls nicht realisieren. Wenn ein Verfahren in der Theorie als gebrochen gilt, kann es in der Praxis durchaus weiterhin zur Anwendung kommen. DES gilt, wie bereit erwähnt, als gebrochen. Seine Anwendung ist jedoch immer noch weiter verbreitet, so wird es beispielsweise in E-Mail Programmen zur Verschlüsselung von Nachrichten eingesetzt. Oft auch in etwas robusteren Varianten wie tripple DES. AES bietet mit seiner Struktur und großen Schlüssellänge hervorragende Sicherheitseigenschaften und wird deswegen auf lange Sicht eine praktische Relevanz behalten.

Anhang A

A.1 Darstellung der S-Box

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0	99	124	119	123	242	107	111	197	48	1	103	43	254	215	171	118
16	202	130	201	125	250	89	71	240	173	212	162	175	156	164	114	192
32	183	253	147	38	54	63	247	204	52	165	229	241	113	216	49	21
48	4	199	35	195	24	150	5	154	7	18	128	226	235	39	178	117
64	9	131	44	26	27	110	90	160	82	59	214	179	41	227	47	132
80	83	209	0	237	32	252	177	91	106	203	190	57	74	76	88	207
96	208	239	170	251	67	77	51	133	69	249	2	127	80	60	159	168
112	81	163	64	143	146	157	56	245	188	182	218	33	16	255	243	210
128	205	12	19	236	95	151	68	23	196	167	126	61	100	93	25	115
144	96	129	79	220	34	42	144	136	70	238	184	20	222	94	11	219
160	224	50	58	10	73	6	36	92	194	211	172	98	145	149	228	121
176	231	200	55	109	141	213	78	169	108	86	244	234	101	122	174	8
192	186	120	37	46	28	166	180	198	232	221	116	31	75	189	139	138
208	112	62	181	102	72	3	246	14	97	53	87	185	134	193	29	158
224	225	248	152	17	105	217	142	148	155	30	135	233	206	85	40	223
240	140	161	137	13	191	230	66	104	65	153	45	15	176	84	187	22

Abbildung A.1: AES S-Box (Vgl.[31, S.83].)

A.2 Darstellung der inversen S-Box

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0	82	9	106	213	48	54	165	56	191	64	163	158	129	243	215	251
16	124	227	57	130	155	47	255	135	52	142	67	68	196	222	233	203
32	84	123	148	50	166	194	35	61	238	76	149	11	66	250	195	78
48	8	46	161	102	40	217	36	178	118	91	162	73	109	139	209	37
64	114	248	246	100	134	104	152	22	212	164	92	204	93	101	182	146
80	108	112	72	80	253	237	185	218	94	21	70	87	167	141	157	132
96	144	216	171	0	140	188	211	10	247	228	88	5	184	179	69	6
112	208	44	30	143	202	63	15	2	193	175	189	3	1	19	138	107
128	58	145	17	65	79	103	220	234	151	242	207	206	240	180	230	115
144	150	172	116	34	231	173	53	133	226	249	55	232	28	117	223	110
160	71	241	26	113	29	41	197	137	111	183	98	14	170	24	190	27
176	252	86	62	75	198	210	121	32	154	219	192	254	120	205	90	244
192	31	221	168	51	136	7	199	49	177	18	16	89	39	128	236	95
208	96	81	127	169	25	181	74	13	45	229	122	159	147	201	156	239
224	160	224	59	77	174	42	245	176	200	235	187	60	131	83	153	97
240	23	43	4	126	186	119	214	38	225	105	20	99	85	33	12	125

Abbildung A.2: AES inverse S-Box (Vgl.[31, S.84].)

A.3 Darstellung der Rundenkonstanten

i	0	1	2	3	4	5	6	7
Rcon[i]	00	01	02	04	08	10	20	40
i	8	9	10	11	12	13	14	15
Rcon[i]	80	1B	36	6C	D8	AB	4D	9A
i	16	17	18	19	20	21	22	23
Rcon[i]	2F	5E	BC	63	C6	97	35	6A
i	24	25	26	27	28	29	30	31
Rcon[i]	D4	B3	7D	FA	EF	C5	91	39

Tabelle A.1: AES Rundenkonstanten (Vgl.[15, S.214].)

Index

Literaturverzeichnis

[1] Eli Biham. New types of cryptanalytic attacks using related keys. *J. Cryptology*, 7(4):229–246, 1994.

[2] Eli Biham, Alex Biryukov, and Adi Shamir. Cryptanalysis of skipjack reduced to 31 rounds using impossible differentials. *J. Cryptology*, 18(4):291–311, 2005.

[3] Eli Biham, Orr Dunkelman, and Nathan Keller. The rectangle attack - rectangling the Serpent. In Birgit Pfitzmann, editor, *EUROCRYPT*, volume 2045 of *Lecture Notes in Computer Science*, pages 340–357. Springer, 2001.

[4] Eli Biham, Orr Dunkelman, and Nathan Keller. New results on boomerang and rectangle attacks. In Joan Daemen and Vincent Rijmen, editors, *FSE*, volume 2365 of *Lecture Notes in Computer Science*, pages 1–16. Springer, 2002.

[5] Eli Biham, Orr Dunkelman, and Nathan Keller. Rectangle attacks on 49-round SHACAL-1. In Thomas Johansson, editor, *FSE*, volume 2887 of *Lecture Notes in Computer Science*, pages 22–35. Springer, 2003.

[6] Eli Biham, Orr Dunkelman, and Nathan Keller. Related-key boomerang and rectangle attacks. In Ronald Cramer, editor, *EUROCRYPT*, volume 3494 of *Lecture Notes in Computer Science*, pages 507–525. Springer, 2005.

[7] Eli Biham, Orr Dunkelman, and Nathan Keller. Related-key impossible differential attacks on 8-round AES-192. In David Pointcheval, editor, *CT-RSA*, volume 3860 of *Lecture Notes in Computer Science*, pages 21–33. Springer, 2006.

[8] Eli Biham and A. Shamir. Differential Cryptanalysis of DES-like Cryptosystems. *Journal of Cryptology*, 4(1):3–72, 1991.

[9] Eli Biham and A. Shamir. *Differential Cryptanalysis of the Data Encryption Standard*. Springer-Verlag, Berlin Heidelberg New York, 1993.

[10] Alex Biryukov. The boomerang attack on 5 and 6-round reduced AES. In Dobbertin et al. [18], pages 11–15.

[11] Alex Biryukov, Christophe De Cannière, and Gustaf Dellkrantz. Cryptanalysis of SAFER++. In Dan Boneh, editor, *CRYPTO*, volume 2729 of *Lecture Notes in Computer Science*, pages 195–211. Springer, 2003.

[12] Johannes Buchmann. *Einführung in die Kryptographie*. Springer Verlag, Berlin Heidelberg, 2004.

[13] Jung Hee Cheon, MunJu Kim, Kwangjo Kim, Jung-Yeun Lee, and SungWoo Kang. Improved impossible differential cryptanalysis of Rijndael and crypton. In Kwangjo Kim, editor, *ICISC*, volume 2288 of *Lecture Notes in Computer Science*, pages 39–49. Springer, 2001.

[14] Joan Daemen, Lars R. Knudsen, and Vincent Rijmen. The block cipher Square. In Eli Biham, editor, *Fast Software Encryption*, volume 1267 of *Lecture Notes in Computer Science*, pages 149–165. Springer, 1997.

[15] Joan Daemen and Vincent Rijmen. *The Design of Rijndael: AES - The Advanced Encryption Standard*. Springer Verlag, Berlin Heidelberg, 2002.

[16] Whitfield Diffie and Martin E. Hellman. New Directions in Cryptography. *IEEE Transactions on Information Theory*, IT-22(6):644–654, 1976.

[17] Hans Dobbertin, Lars R. Knudsen, and Matthew J. B. Robshaw. The cryptanalysis of the AES - A brief survey. In Dobbertin et al. [18], pages 1–10.

[18] Hans Dobbertin, Vincent Rijmen, and Aleksandra Sowa, editors. *Advanced Encryption Standard - AES, 4th International Conference, AES 2004, Bonn, Germany, May 10-12, 2004, Revised Selected and Invited Papers*, volume 3373 of *Lecture Notes in Computer Science*. Springer, 2005.

[19] Orr Dunkelman. *Techniques for Cryptanalysis of Block Ciphers*. PhD thesis, Israel Institute of Technology, 2006.

[20] Orr Dunkelman and N. Keller. Boomerang and Rectangle Attacks on SC2000 NES/DOC/TEC/WP3/014/a. *Proceedings of Second Open NESSIE Workshop*, September 12-13, 2001.

[21] H. Feistel. Cryptography and Computer Privacy. *Scientific American*, 228(5):15–23, 1973.

[22] Niels Ferguson, John Kelsey, Stefan Lucks, Bruce Schneier, Michael Stay, David Wagner, and Doug Whiting. Improved cryptanalysis of Rijndael. In Schneier [39], pages 213–230.

[23] H.M. Heys. A Tutorial on Linear and Differential Cryptanalysis. *Cryptologia*, 26:189–221, 2002.

[24] Seokhie Hong, Jongsung Kim, Sangjin Lee, and Bart Preneel. Related-key rectangle attacks on reduced versions of SHACAL-1 and aes-192. In Henri Gilbert and Helena Handschuh, editors, *FSE*, volume 3557 of *Lecture Notes in Computer Science*, pages 368–383. Springer, 2005.

[25] Goce Jakimoski and Yvo Desmedt. Related-key differential cryptanalysis of 192-bit key AES variants. In Mitsuru Matsui and Robert J. Zuccherato, editors, *Selected Areas in Cryptography*, volume 3006 of *Lecture Notes in Computer Science*, pages 208–221. Springer, 2003.

[26] John Kelsey, Tadayoshi Kohno, and Bruce Schneier. Amplified boomerang attacks against reduced-round MARS and serpent. In Schneier [39], pages 75–93.

[27] Jongsung Kim, Guil Kim, Seokhie Hong, Sangjin Lee, and Dowon Hong. The related-key rectangle attack - application to SHACAL-1. In Huaxiong Wang, Josef Pieprzyk, and Vijay Varadharajan, editors, *ACISP*, volume 3108 of *Lecture Notes in Computer Science*, pages 123–136. Springer, 2004.

[28] Jongsung Kim, Dukjae Moon, Wonil Lee, Seokhie Hong, Sangjin Lee, and Seokwon Jung. Amplified boomerang attack against reduced-round SHACAL. In Yuliang Zheng, editor, *ASIACRYPT*, volume 2501 of *Lecture Notes in Computer Science*, pages 243–253. Springer, 2002.

[29] Lars R. Knudsen. Cryptanalysis of Loki91. In Jennifer Seberry and Yuliang Zheng, editors, *ASIACRYPT*, volume 718 of *Lecture Notes in Computer Science*, pages 196–208. Springer, 1992.

[30] Lars R. Knudsen. Truncated and higher order differentials. In Bart Preneel, editor, *Fast Software Encryption*, volume 1008 of *Lecture Notes in Computer Science*, pages 196–211. Springer, 1994.

[31] Volker Krummel. Sicherheit und Anwendungen des Advanced Encryption Standard (AES) Rijndael. Master's thesis, Paderborn University, 2001.

[32] R. Lidl and H. Niederreiter. *Introduction to finite fields and their applications*. Cambridge University Press, 1986.

[33] Stefan Lucks. Attacking seven rounds of Rijndael under 192-bit and 256-bit keys. In *AES Candidate Conference*, pages 215–229, 2000.

[34] Stefan Lucks. Ciphers secure against related-key attacks. In Bimal K. Roy and Willi Meier, editors, *FSE*, volume 3017 of *Lecture Notes in Computer Science*, pages 359–370. Springer, 2004.

[35] Lauren May, Matthew Henricksen, William Millan, Gary Carter, and Ed Dawson. Strengthening the key schedule of the AES. In Lynn Margaret Batten and Jennifer Seberry, editors, *ACISP*, volume 2384 of *Lecture Notes in Computer Science*, pages 226–240. Springer, 2002.

[36] A.J. Menezes, P.C. van Oorschot, and S.A. Vanstone. *Handbook of Applied Cryptography*. CRC Press, 1997.

[37] Y. Nawaz. Cryptanalysis of Block Ciphers (A Survey). *http://scholar.google.de/url?sa=U&q=http://www.dice.ucl.ac.be/ fstandae/CG-2003-2.pdf*, 2004.

[38] Bruce Schneier. *Angewandte Kryptographie*. Addison-Wesley, 1996.

[39] Bruce Schneier, editor. *Fast Software Encryption, 7th International Workshop, FSE 2000, New York, NY, USA, April 10-12, 2000, Proceedings*, volume 1978 of *Lecture Notes in Computer Science*. Springer, 2001.

[40] C. Shannon. Communication theory of secrecy systems. *Bell Syst. Tech. J.*, 28:656–715, 1949.

[41] F.X. Standaert, G. Piret, and J.J. Quisquater. Cryptanalysis of Block Ciphers: A Survey. *citeseer.ist.psu.edu/standaert03cryptanalysis.html*, 2003.

[42] D.R. Stinson. *Cryptography: Theory and Practice*. Chapman & Hall, 2002. 2nd ed.

[43] David Wagner. The boomerang attack. In Lars R. Knudsen, editor, *Fast Software Encryption*, volume 1636 of *Lecture Notes in Computer Science*, pages 156–170. Springer, 1999.

[44] J. Wolfart. *Einführung in die Zahlentheorie und Algebra*. Vieweg & Sohn Verlagsgesellschaft mbH, 1996.

[45] P.D. Yacoumis. On the Security of the Advanced Encryption Standard. *http://ntcourtois.free.fr/yacoumis_aes.pdf*, 2005.