Leibniz Universität Hannover
Fakultät für Elektrotechnik und Informatik
Institut für Theoretische Informatik

Bachelorarbeit

Ununterscheidbarkeit in der Kryptographie

Indistinguishability in Cryptography

Kevin Kässens
Matrikelnummer 3216580

22. September 2019

Erstprüfer: Prof. Dr. Heribert Vollmer
Zweitprüfer: Dr. rer. nat. Arne Meier

Erklärung der eigenständigen Arbeit

Hiermit versichere ich, die vorliegende Bachelorarbeit selbstständig verfasst und keine anderen als die angegebenen Quellen und Hilfsmittel verwendet zu haben. Die Arbeit hat in gleicher oder ähnlicher Form noch keinem anderen Prüfungsamt vorgelegen.

Hannover, den 20. September 2019

Kevin Kässens

Inhaltsverzeichnis

1 Einleitung

Im Folgenden soll ein kurzer Überblick über den Inhalt dieser Arbeit gegeben werden. Die Themen werden im späteren Verlauf genauer beschrieben und erklärt.

Um die Sicherheit eines Kryptosystems bewerten zu können, muss zunächst geklärt werden, was der Begriff „Sicherheit“ im Zusammenhang mit Kryptosystemen überhaupt bedeutet.

Das Kerckhoffs'sche Prinzip besagt, dass die Sicherheit eines Verschlüsselungsverfahrens einzig von der Geheimhaltung des Schlüssels, nicht aber von der Geheimhaltung des verwendeten Verfahrens zur Verschlüsselung abhängig sein darf. Auch muss die Sicherheit unabhängig von der Formatierung des Klartextes gegeben sein. Darüber hinaus muss jedoch auch geklärt werden, ob ein Kryptosystem sicher ist, sobald ein potentieller Angreifer eine gewisse Fähigkeit nicht hat bzw. unsicher ist, sobald er eine andere Fähigkeit besitzt.

Es ist offensichtlich, dass der verwendete Schlüssel nicht aus einem oder mehreren Geheimtexten ableitbar sein darf. Auch ein Erlangen des Klartextes aus dem Geheimtext, ohne in Besitz des Schlüssels zu sein, darf nicht möglich sein. Diese Bedingung gilt nicht nur für den Klartext im Gesamten, sondern auch bereits für Teile dessen, wie z. B. die ersten oder letzten Zeichen, da diese auch nur für sich bereits sensible Informationen enthalten können. Außerdem darf es in einem sicheren Kryptosystem ebenfalls nicht möglich sein, bei gegebenen Geheimtexten zweier verschiedener Klartexte gleicher Länge, diese den Klartexten einzeln zuordnen zu können. Diese Einschränkung wird Ununterscheidbarkeit von Geheimtexten genannt.

Warum ist Ununterscheidbarkeit wichtig? Ein deterministischer Algorithmus liefert bei identischer Eingabe jedes Mal dieselbe Ausgabe. Wird ein solcher deterministischer Algorithmus in einem Kryptosystem zur Verschlüsselung verwendet, bedeutet dies, dass bei Eingabe desselben Klartextes jedes Mal derselbe Geheimtext produziert wird. Dieser Umstand kann jedoch problematisch sein, z. B. wenn ein Angreifer bei einem asymmetrischen Kryptosystem den Klartextraum einschränken kann, z. B. auf lediglich „Ja“ und „Nein“. In einem solchen Fall kann der Angreifer alle Klartexte des Klartextraumes verschlüsseln, seine Ergebnisse mit einem gegebenen Geheimtext vergleichen und schließlich einfach auf dessen Klartext zurückschließen.

Um dies zu verhindern, muss der Algorithmus, der im Kryptosystem zur Verschlüsselung verwendet wird, entweder zustandsorientiert (engl. „stateful“) oder probabilistisch sein.

Die Ununterscheidbarkeit von Geheimtexten (engl. „Ciphertext Indistinguishability“) als Sicherheitsziel besagt, dass ein Angreifer nicht zwischen den Verschlüsselungen eines Klartextpaars zweier verschiedener Klartexte gleicher Länge unterscheiden können darf. Der daraus resultierende Wunsch, dass ein potentieller Angreifer bei einem gegebenen Geheimtext nichts über den Klartext weiß und auch keine weiteren Informationen über diesen aus dem Geheimtext gewinnen kann, ist jedoch nicht realistisch. Auch wird die Einschränkung getroffen, dass dem Angreifer nur realistisch beschränkte Ressourcen, z. B. bezüglich der Anzahl der erlaubten Rechenschritte, zur Verfügung stehen.

Um ein Kryptosystem auf das Sicherheitsziel der Ununterscheidbarkeit von Geheimtexten zu prüfen, sind verschiedene Spiele, auch Experimente genannt, definiert. Die gebräuchlichsten Definitionen sind IND-CPA („indistinguishability under chosen plaintext attack“), sowie IND-CCA („indistinguishability under chosen ciphertext attack“).

Das IND-CPA Spiel für ein symmetrisches Kryptosystem besteht aus einem Widersacher, welcher ein Algorithmus mit beschränkten Ressourcen ist, und einem Herausforderer. Der Widersacher hat zudem Zugriff auf ein Verschlüsselungsorakel.

Im Spiel kann der Widersacher, der keinen Zugriff auf den geheimen Schlüssel des Kryptosystems hat, eine Sequenz von jeweils zwei verschiedenen Klartexten gleicher Länge wählen und diese an das Orakel senden. Der Herausforderer entscheidet zu Beginn des Spiels, ob jeweils nur der erste oder nur der zweite der beiden Klartexte eines Klartextpaars verschlüsselt wird. Das Verschlüsselungsorakel sendet dann dieser Festlegung entsprechend einen Klartext als verschlüsselten Geheimtext an den Widersacher zurück. Die Auswahl der Klartextpaare der Sequenz durch den Widersacher kann entweder adaptiv oder nicht adaptiv erfolgen. Bei der nicht-adaptiven Variante sendet der Widersacher die gesamte Sequenz an den Herausforderer, welcher dann die verschlüsselten Geheimtexte aller Klartextpaare liefert. In der adaptiven Variante sendet der Widersacher jeweils nur ein Klartextpaar und wählt das nächste Klartextpaar erst aus, wenn er den Geheimtext des vorherigen Paars erhalten hat.

Am Ende des Spiels muss schließlich der Widersacher eine Vermutung aufstellen, welcher der beiden Klartexte jeweils verschlüsselt wurde, also wofür sich der Herausforderer zu Beginn entschieden hat.

Der IND-CPA-Vorteil des Widersachers gibt an, wie viel besser seine Vermutungen im Vergleich zu einfachem Raten, also einer Erfolgswahrscheinlichkeit von 50%, sind.

Ein Kryptosystem gilt als sicher gegen einen Angriff mit Klartextwahl (engl. „chosen plaintext attack“, kurz: CPA), wenn kein Angreifer unter Einsatz realistisch beschränkter Ressourcen einen signifikanten Vorteil erzielen kann.

Das IND-CCA Spiel ist eine Erweiterung des IND-CPA Spiels mit einem mächtigeren Widersacher. Das Ziel des Widersachers bleibt identisch zum IND-CPA Spiel, doch erhält er in diesem Spiel zusätzlich ein Entschlüsselungsorakel (engl. „decryption oracle“). Mit Hilfe dieses Entschlüsselungsorakels kann der Widersacher einen Geheimtext entschlüsseln und so den Klartext dazu erhalten, jedoch mit der Einschränkung dass er keine Geheimtexte, die er zuvor vom Herausforderer bekommen hat, entschlüsseln darf. Daher modifiziert der Angreifer gewöhnlich diese zurückbekommenen Geheimtexte und lässt

dann die modifizierten Geheimtexte vom Entschlüsselungsorakel entschlüsseln. Dabei versucht er die Geheimtexte so zu modifizieren, dass die Entschlüsselung Informationen über den ursprünglichen Klartext des unmodifizierten Geheimtextes preisgibt.

Beim IND-CCA Spiel wird zwischen den Definitionen IND-CCA1 und IND-CCA2 unterschieden. IND-CCA1 bezeichnet die „indistinguishability under a non-adaptive chosen ciphertext attack“. Der Widersacher kann seine Geheimtexte also nicht adaptiv wählen. In IND-CCA2 ist eine adaptive Wahl der Geheimtexte hingegen möglich.

In dieser Arbeit werden, wie zu Beginn erwähnt, in den folgenden Kapiteln die in dieser Einleitung vorgestellten Themen genauer betrachtet.

Zunächst werden einige Grundlagen festgelegt (Kapitel 2), daraufhin drei Kryptosysteme speziell vorgestellt (Kapitel 3). Im Anschluss wird das Konzept der Ununterscheidbarkeit detailliert betrachtet und auf die vorgestellten Kryptosysteme beispielhaft angewendet (Kapitel 4). In einem Ausblick (Kapitel 5) werden schließlich noch weitere Definitionen der Ununterscheidbarkeit kurz vorgestellt.

2 Grundlagen

In diesem Kapitel sollen zunächst grundlegende Begriffe, die zum Verständnis der nachfolgenden Kapitel unverzichtbar sind, vorgestellt und definiert werden.

2.1 Kryptosysteme

Die Definitionen zu Kryptosystemen, Ver- und Entschlüsselung, sowie Verschlüsselungsarten folgen den Definitionen von Ralf Küsters und Thomas Wilke [14].

2.1.1 Verschlüsselung und Entschlüsselung

Eine kryptographische Verschlüsselung besteht aus einem Chiffrierschlüssel, kurz auch nur Schlüssel genannt, einer zu verschlüsselnden Nachricht, dem Klartext, sowie einer „neuen“ Nachricht, genannt Chiffre- oder Geheimtext. Beim Verschlüsselungsvorgang, auch Chiffriervorgang genannt, wird schließlich mit Hilfe des Schlüssels aus dem Klartext der Geheimtext berechnet.

Zusätzlich zu der Verschlüsselung existiert natürlich auch die Entschlüsselung. Diese kann mit Hilfe des Dechiffrierschlüssels einen Geheimtext zurück in den Klartext überführen. Je nach Verschlüsselungsart können Chiffrier- und Dechiffrierschlüssel unterschiedlich oder identisch sein.

2.1.2 Verschlüsselungsarten

Bei der symmetrischen Verschlüsselung sind Chiffrier- und Dechiffrierschlüssel identisch, weswegen häufig auch nur von einem einzigen *symmetrischen Schlüssel* die Rede ist. Praktisch bedeutet dies, dass derselbe Schlüssel zum Verschlüsseln des Klartextes, als auch zum Entschlüsseln des Geheimtextes benötigt wird. Daher ist vor Beginn der Nachrichtenübertragung ein Austausch dieses Schlüssels zwischen allen an der Kommunikation teilnehmenden Personen über einen sicheren Kanal nötig.

Bei der asymmetrischen Verschlüsselung sind Chiffrier- und Dechiffrierschlüssel unterschiedlich. Jeder Teilnehmer des Nachrichtenaustausches besitzt ein eigenes Schlüsselpaar, bestehend aus einem öffentlichen Schlüssel, dem Chiffrierschlüssel, und einem privaten Schlüssel, dem Dechiffrierschlüssel. Zur Verschlüsselung eines Klartextes wird der öffentliche Schlüssel des Nachrichtenempfängers verwendet, bei der Entschlüsselung eines Geheimtextes nutzt der Empfänger seinen eigenen privaten Schlüssel.

2.1.3 Kryptosystem

Ein Kryptosystem setzt sich aus einer nicht-leeren, endlichen Menge von Klartexten, genannt *Klartextraum*, einer Menge von Geheimtexten, genannt *Geheimtextraum*, sowie einer nicht-leeren, endlichen Menge von Schlüsseln, genannt *Schlüsselraum*, zusammen. Außerdem wird eine *Chiffrier-*, sowie eine *Dechiffrierfunktion* benötigt.

Die Chiffrierfunktion berechnet für einen Klartext aus dem Klartextraum mit Hilfe eines Schlüssels aus dem Schlüsselraum den zugehörigen Geheimtext aus dem Geheimtextraum. Umgekehrt ermittelt die Dechiffrierfunktion für einen Geheimtext den zugehörigen Klartext.

Definition 2.1 (Kryptosystem)**.** *Ein Kryptosystem ist ein 5-Tupel* $\mathscr{S} = (X, K, Y, e, d)$ *mit einem Klartextraum* X*, einem Schlüsselraum* K *und einem Geheimtextraum* Y*. Zudem existiert eine Chiffrierfunktion* $e\colon X \times K \to Y$*, sowie eine Dechiffrierfunktion* $d\colon Y \times K \to X$*.*

In einem symmetrischen Kryptosystem muss $d(e(x,k),k) = x$ für alle $x \in X, k \in K$, sowie $Y = \{e(x,k) \mid x \in X, k \in K\}$ gelten. Hierbei wird $e(\cdot,k)$, alternativ auch $e_k(\cdot)$ geschrieben, als Chiffre für einen Schlüssel $k \in K$ bezeichnet, $d(\cdot,k)$ als die Umkehrfunktion zu $e(\cdot,k)$.

In einem asymmetrisches Kryptosystem ist der Schlüsselraum K eine Menge von Schlüsselpaaren (k,k'). Ein Schlüsselpaar setzt sich aus einem privaten Schlüssel k und einem öffentlichen Schlüssel k' zusammen. Als $K_{private}$ wird die Menge der privaten Schlüssel, als K_{public} die Menge der öffentlichen Schlüssel bezeichnet.

Bei der Dechiffrierung muss $d(e(x,k'),k) = x$ für alle $x \in X, (k,k') \in K$ gelten.

2.1.4 Angriffsarten auf Kryptosysteme

Die verschiedenen Angriffsarten definieren jeweils ein Angriffsszenarium und die Fähigkeiten, die dem darin enthaltenen Angreifer zur Verfügung stehen. Das Ziel des Angreifers ist es stets, einen Klartext oder den Schlüssel zu erhalten [9].

Die folgenden Angriffsarten sind in ihrer Angriffsstärke aufsteigend. Eine Übersicht wird zudem in Tabelle 2.1 gegeben. Die Definitionen basieren auf den Definitionen von Johannes A. Buchmann [8].

Ciphertext-Only-Angriff

Bei einem *Ciphertext-Only-Angriff* (kurz „COA“) ist dem Angreifer lediglich ein Geheimtext bekannt. Aus diesem versucht er Informationen über den verwendeten Schlüssel oder den zum Geheimtext gehörenden Klartext zu gewinnen.

Ein Beispiel für einen Ciphertext-Only-Angriff ist die *vollständige Suche*, bei der der Angreifer den Geheimtext mit jedem im Schlüsselraum enthaltenen Schlüssel entschlüsselt. Unter den sich daraus ergebenden, sinnvollen Ergebnissen befindet sich auch der Klartext. Dieser Angriff ist jedoch nur praktikabel, wenn der Schlüsselraum nicht zu

Angriffsart	Fähigkeiten und Ziele des Angreifers
Ciphertext-Only-Angriff	Kennt lediglich ein oder mehrere Geheimtexte. Klartexte oder der Schlüssel sollen aufgedeckt werden
Known-Plaintext-Angriff	Kennt Paare aus Klar- und Geheimtexten, weitere Klartexte oder der Schlüssel sollen aufgedeckt werden
Chosen-Plaintext-Angriff	Verschlüsselung selbstgewählter Klartexte, eine adaptive Klartextwahl ist möglich. Ziel ist die Gewinnung von Informationen
Chosen-Ciphertext-Angriff	Verschlüsselung selbstgewählter Klartexte sowie Entschlüsselung selbstgewählter Geheimtexte. Eine adaptive Klar- und Geheimtextwahl ist möglich. Ziel ist die Gewinnung von Informationen

Tabelle 2.1: Übersicht zu Angriffsarten

groß ist: Während der DES-Algorithmus mit einer Schlüssellänge von 56 Bit bereits gegen einen solchen Angriff unsicher ist, gilt der in Abschnitt 3.1 vorgestellte AES-Algorithmus mit einer Schlüssellänge von mindestens 128 Bit als praktisch sicher gegen die vollständige Suche [6].

Ein Angreifer wird als *passiver Angreifer* bezeichnet, wenn er nur zu einem Ciphertext-Only-Angriff fähig ist. Verfügt der Angreifer auch über weitere Angriffsarten, wird er als *aktiver Angreifer* bezeichnet.

Known-Plaintext-Angriff

Im Gegensatz zum Ciphertext-Only-Angriff kennt der Angreifer bei einem *Known-Plaintext-Angriff* (kurz „KPA“) zusätzlich die zu den gegebenen Geheimtexten zugehörigen Klartexte. Mit Hilfe dieser Informationen versucht er dann einen Klartext, den er noch nicht kennt, oder den Schlüssel zu bekommen [9].

Chosen-Plaintext-Angriff

Bei einem *Chosen-Plaintext-Angriff* (kurz „CPA“) kann sich der Angreifer Klartexte, die er zuvor selbst gewählt hat, verschlüsseln lassen und so die zugehörigen Geheimtexte erhalten. Eine Erweiterung dieser Angriffsart ist der *adaptive* Chosen-Plaintext-Angriff. Der Unterschied ist hierbei, dass der Angreifer die Klartexte nacheinander und somit von den Ergebnissen zuvor bereits gesendeter Klartexte, sowie von eigenen, im Laufe des Angriffs getätigter Berechnungen abhängig machen kann.

Da der adaptive Chosen-Plaintext-Angriff die gängigere Variante dieser Angriffsart ist, wird diese oftmals gemeint, auch wenn nur von einem Chosen-Plaintext-Angriff gesprochen wird.

Chosen-Ciphertext-Angriff

Bei einem Chosen-Ciphertext-Angriff (kurz „CCA“) kann sich der Angreifer zusätzlich selbst gewählte Geheimtexte entschlüsseln lassen und dadurch die zugehörigen Klartexte bekommen. Der verwendete Schlüssel wird dem Angreifer dabei jedoch nicht bekannt.

Wie beim Chosen-Plaintext-Angriff existiert auch ein adaptiver Chosen-Ciphertext-Angriff.

2.2 Strom- und Blockchiffren

Eine perfekt sichere Chiffre würde einen Schlüssel mit der gleichen Länge wie der Klartext benötigen, was einen hohen Aufwand bei langen, zu verschlüsselnden Nachrichten bedeutet. Um diesen Aufwand zu umgehen, kann man eine Strom- oder Blockchiffre einsetzen. Diese sind in der Anwendung einfacher umsetzbar und bieten trotzdem eine hohe Sicherheit [6].

Die Definitionen zu Strom- und Blockchiffren folgen [14].

2.2.1 Stromchiffren

Mit Stromchiffren ist es möglich, Klartexte beliebiger Länge zu verschlüsseln [2]. Verschlüsselt wird dabei ein Datenstrom (engl. „stream“) $x := x_1 x_2 \dots x_n \in X^*$, bestehend aus Klartextzeichen $x_i \in X$ mit $1 \leq i \leq n$, in den Strom $y := y_1 y_2 \dots y_n \in Y^*$, bestehend aus Geheimtextzeichen $y_i \in Y$ mit $1 \leq i \leq n$. Verwendet wird dafür ein Schlüsselstrom $k := k_1 k_2 \dots k_n \in K^*$ mit $k_i \in K$ und $1 \leq i \leq n$. Der Klartextstrom wird Zeichen für Zeichen mit dem zugehörigen Schlüssel aus dem Schlüsselstrom verschlüsselt [9].

Definition 2.2 (Stromchiffre)**.** *Eine Stromchiffre verschlüsselt einen aus Klartextzeichen bestehenden Datenstrom $x \in X^*$ zu einem Geheimtextstrom $y \in Y^*$ mittels eines Schlüsselstroms $k \in K^*$ durch*

$$y_i = e(x_i, k_i) \text{ für } i = 1, \dots, n$$

2.2.2 Blockchiffren

Blockchiffren sind ein wichtiges Mittel für symmetrische Verschlüsselungsverfahren. Mit Blockchiffren ist es möglich, Kryptosysteme zu realisieren, die trotz kurzer Schlüssel, das heißt Schlüssellängen von wenigen hundert Bit, eine hohe Sicherheit, zumindest gegen Angreifer mit realistisch beschränkten Ressourcen, bieten können [14].

Eine Blockchiffre ist eine Funktion $E\colon \{0,1\}^\ell \times \{0,1\}^n \to \{0,1\}^n$, die aus einem Schlüssel der Länge ℓ und einem Klartext der Länge n einen Geheimtext, der ebenfalls der Länge n ist, produziert. Die Schlüssellänge ℓ (engl. „key length“) und die Blocklänge n (engl. „block length“) sind dabei feste Werte und Eigenschaften der verwendeten Blockchiffre [2].

Definition 2.3 (Block-Kryptosystem)**.** *Ein Block-Kryptosystem, auch n-Block-Kryptosystem genannt, ist das Kryptosystem* $\mathscr{B} = (\{0,1\}^n, K, \{0,1\}^n, E, D)$ *mit* $K \subseteq \{0,1\}^\ell$ *für ein* $\ell > 0$ *[14]. Die Verschlüsselung eines Klartextblocks* m *mit fester Länge* n *mit einem Schlüssel* $k \in K$ *erfolgt dann durch* $c = e(m,k)$ *[9].*

Übliche Blockgrößen liegen bei 64 Bit oder 128 Bit, übliche Schlüssellängen sind z. B. 56 Bit, 128 Bit und 256 Bit [9].

2.2.3 Blockchiffren-Betriebsmodi

Blockchiffren können nur Nachrichten verarbeiten, die ihrer Blockgröße n entsprechen. Längere Nachrichten müssen daher zuvor in Blöcke dieser Länge n aufgeteilt werden.

Als Betriebsmodus einer Blockchiffre wird das Verfahren bezeichnet, wie die Chiffre mit solchen Nachrichten umgeht. Die Definitionen in diesem Abschnitt folgen [6] und [14].

ECB-Mode

Beim *Electronic-Codebook-Mode*, kurz ECB-Mode, wird der Klartext zunächst in Blöcke der Länge n aufgeteilt. Wenn die Länge des Klartextes nicht restlos durch n geteilt werden kann, wird dieser erweitert, z. B. mit zufälligen Bits, bis restloses Teilen möglich ist. Dieser Vorgang wird als *Padding* (dt. „Auffüllen“) bezeichnet. Die daraus entstehenden Klartextblöcke werden dann einfach unabhängig voneinander verschlüsselt [8]. Der ECB-Mode wird in Abbildung 2.1 dargestellt.

Definition 2.4 (ECB-Betriebsmodus)**.** *Im ECB-Mode werden Klartextblöcke* m_i *fester Länge* n *mittels*

$$c_i = e(m_i, k)$$

für Klartextblock i*, Schlüssel* $k \in K$ *zu Geheimtextblöcken* c_i *verschlüsselt.*

Der finale Geheimtext ist dann die Folge der Geheimtextblöcke, die aus den Klartextblöcken ermittelt wurden. Ebenso wie die Verschlüsselung ist auch die Entschlüsselung der Geheimtextblöcke unabhängig von den anderen Blöcken.

Der ECB-Mode ist ein zustandsloser (engl. „stateless“) Betriebsmodus [2].

CBC-Mode

Auch beim *Cipherblock-Chaining-Mode*, kurz CBC-Mode, wird der Klartext zunächst in Blöcke der Länge n aufgeteilt. Ist kein restloses Aufteilen möglich, wird wie beim ECB-Mode das Padding angewendet. Die Verschlüsselung eines Klartextblockes hängt in diesem Modus jedoch neben dem Schlüssel zusätzlich von den vorhergehenden Blöcken ab.

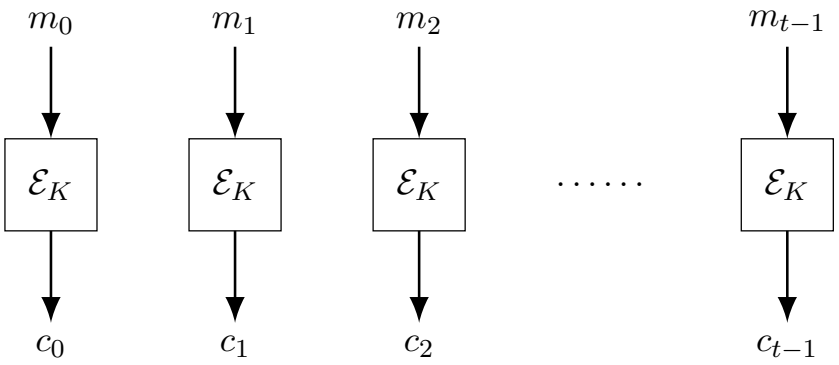

Abbildung 2.1: ECB-Mode

Vor der Verschlüsselung wird der aktuelle Klartextblock zunächst mit dem zuletzt generierten Geheimtextblock verknüpft und diese Verknüpfung schließlich verschlüsselt. Als Verknüpfung wird meist eine XOR-Verknüpfung verwendet.

Da für den ersten zu verschlüsselnden Klartextblock jedoch noch kein Geheimtextblock zum Verknüpfen vorhanden ist, wird für diesen stattdessen ein Initialisierungsvektor $IV \in \{0,1\}^n$ verwendet, welcher vor jeder Verwendung zufällig gewählt wird [8]. Zusätzlich zu den Ergebnissen der Verschlüsselung wird auch der Initialisierungsvektor an den Empfänger übertragen. Der CBC-Mode wird in Abbildung 2.2 dargestellt.

Definition 2.5 (CBC-Betriebsmodus)**.** *Im CBC-Mode erfolgt die Verschlüsselung der Klartextblöcke m_i durch*

$$c_0 = e(m_0 \oplus IV, k)$$

$$c_i = e(m_i \oplus c_{i-1}, k)$$

für den Initialisierungsvektor $IV \in \{0,1\}^n$, $i > 0$ und Schlüssel $k \in K$. m_0 bezeichnet den ersten Klartextblock.

Eine andere Variante des CBC-Mode ist der CBC-C-Mode, bei dem für den Initialisierungsvektor IV statt eines zufälligen Wertes ein Zählerwert verwendet wird. Der Zähler wird bei jeder Nutzung des Initialisierungsvektors hochgezählt, der Startwert des Zählers ist 0 [2].

Da im CBC-Mode jeder Geheimtextblock vom vorigen Geheimtextblock abhängt, ist ein Vertauschen zweier Geheimtextblöcke oder das Fehlen eines Geheimtextblocks beim Entschlüsseln bemerkbar.

Beim Fehlen des i-ten Geheimtextblocks liefert die Entschlüsselung des $i+1$-ten Blocks keinen korrekten Klartext. Alle folgenden Geheimtextblöcke können jedoch wieder korrekt entschlüsselt werden, sofern keine weiteren Blöcke fehlen. Daher ist der CBC-Mode ein *selbstsynchronisierender* Modus.

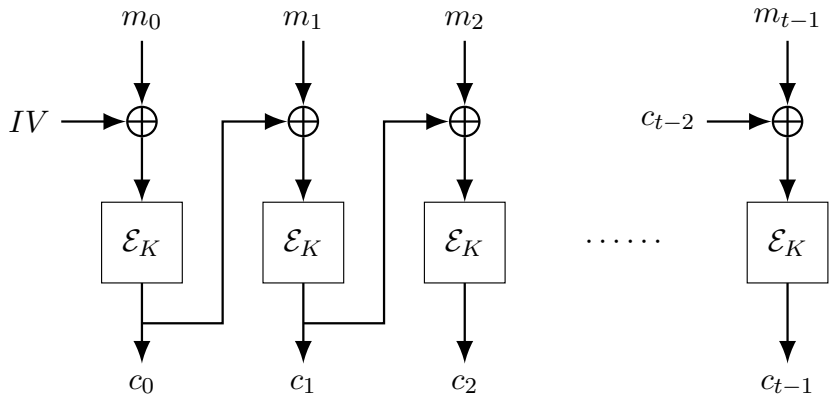

Abbildung 2.2: CBC-Mode

Der CBC-Mode ist ein probabilistischer Betriebsmodus, während der CBC-C-Mode zustandsbasiert (engl. „stateful“) ist [2].

CFB-Mode

Beim *Cipher-Feedback-Mode*, kurz CFB-Mode, wird die Blockchiffre in einer Stromchiffre eingesetzt, weswegen dieser Modus als *stromorientiert* bezeichnet wird. Dadurch ist es auch möglich, die Geheimtextblöcke Zeichen für Zeichen zu entschlüsseln, der gesamte Block muss bei der Entschlüsselung nicht vorliegen. Außerdem ist beim CFB-Mode auch das Verschlüsseln von Blöcken, die kürzer als die Blocklänge n sind, möglich [8].

Die Geheimtextblöcke sind, wie beim CBC-Mode, von den vorigen Geheimtextblöcken abhängig.

Bei einem stromorientierten Betriebsmodus wird der Klartext nicht direkt durch die Blockchiffre verschlüsselt. Zunächst wird, wie beim CBC-Mode, ein Initialisierungsvektor $IV \in \{0,1\}^n$ für den ersten Klartextblock gebildet. Mit der Blockchiffre wird dann stets der $(i-1)$-te Geheimtextblock verschlüsselt und das Ergebnis dieser Verschlüsselung als Schlüsselfolge für eine Stromchiffre verwendet. Der CFB-Mode wird in Abbildung 2.3 dargestellt.

Definition 2.6 (CFB-Betriebsmodus)**.** *Beim stromorientierten CFB-Mode erfolgt die Verschlüsselung der Klartextblöcke m_i durch*

$$c_i = m_i \oplus e(c_{i-1}, k)$$

mit einem Schlüssel $k \in K$. Die Entschlüsselung erfolgt ebenfalls zeichenweise durch

$$m_i = c_i \oplus e(c_{i-1}, k)$$

Beim ersten zu verschlüsselnden Klartextblock wird, wie beim CBC-Mode, der Initialisierungsvektor anstelle eines Geheimtextblocks verwendet.

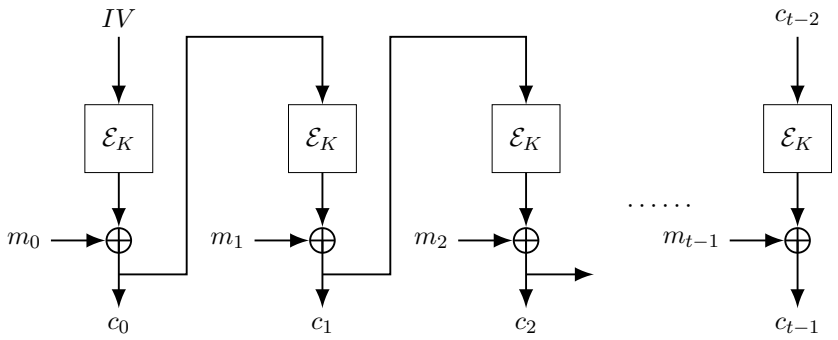

Abbildung 2.3: CFB-Mode

Der CFB-Mode ist selbstsynchronisierend, was bedeutet, dass bei einer fehlerhaften Übertragung eines Geheimtextblockes lediglich der direkt darauffolgende Block inkorrekt entschlüsselt wird. Alle weiteren Geheimtextblöcke werden nicht beeinflusst.

OFB-Mode

Der *Output-Feedback-Mode*, kurz OFB-Mode, ist dem CFB-Mode ähnlich. Auch der OFB-Mode ist ein stromorientierter Modus, die Schlüsselfolge für die Stromchiffre ist jedoch klartextunabhängig. Daher wird der OFB-Mode als *synchroner* Betriebsmodus bezeichnet.

Zunächst wird auch hier ein Initialisierungsvektors IV gebildet, bezeichnet als Startwert s_0. Mit der Blockchiffre werden dann, ausgehend vom Startwert, dessen Iterationen durch Verschlüsselungen gebildet. Da bei der Errechnung keine Abhängigkeiten zu Klar- oder Geheimtexten bestehen, können die Werte theoretisch im Voraus bereits berechnet werden.

Die Klartextblöcke werden dann durch XOR-Verknüpfungen mit den jeweiligen Iterationen des Startwerts verschlüsselt. Der OFB-Mode wird in Abbildung 2.4 dargestellt.

Definition 2.7 (OFB-Betriebsmodus)**.** *Beim OFB-Mode werden zunächst die Iterationen des Startwerts s_0 gebildet:*

$$s_i = e(s_{i-1}, k) \text{ mit Schlüssel } k \in K,\ i \geq 1, s_0 = IV$$

Die Verschlüsselung der Klartextblöcke m_i erfolgt dann mit diesen durch:

$$c_i = m_i \oplus s_i$$

Da es sich beim OFB-Mode um einen synchronen und nicht um einen selbstsynchronisierenden Modus handelt, führt ein Fehlen eines Geheimtextblockes dazu, dass alle folgenden Geheimtextblöcke inkorrekt entschlüsselt werden.

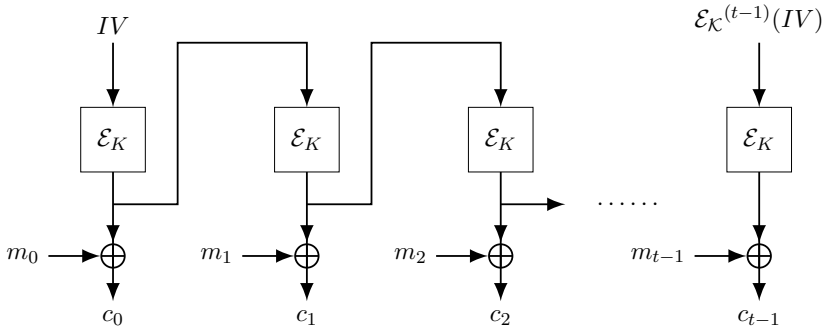

Abbildung 2.4: OFB-Mode

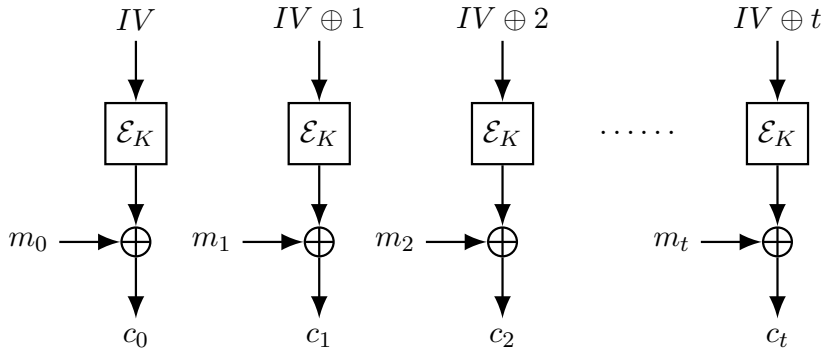

Abbildung 2.5: CTR-Mode

CTR-Mode

Der *Counter-Mode*, kurz CTR-Mode, ist ebenfalls ein stromorientierter Betriebsmodus.

Zunächst wird der Schlüsselstrom gebildet. Der Wert s_0 ist der sogenannte Initialisierungswert und wird zufällig gewählt. Der Klartext wird dann zeichenweise durch eine XOR-Verknüpfung des jeweiligen Klartextzeichens mit dem entsprechenden Element aus dem Schlüsselstrom verschlüsselt. Der CTR-Mode wird in Abbildung 2.5 dargestellt.

Definition 2.8 (CTR-Betriebsmodus)**.** *Beim CTR-Mode wird der Schlüsselstrom durch*

$$s_i = e(s_0 + i, k) \text{ mit } i > 0$$

gebildet. Die Verschlüsselung des Klartextes erfolgt dann zeichenweise durch

$$c_i = m_i \oplus e(c_0 + i, k)$$

Der CTR-Mode ist ein probabilistischer Betriebsmodus und ist, wie der OFB-Mode, synchron, aber nicht selbstsynchronisierend [2].

2.3 Sequenzen von Spielen

Dieses Kapitel bezieht sich auf die Arbeiten von Victor Shoup [16], sowie von Mihir Bellare und Phillip Rogaway [1].

2.3.1 Einleitung

Sicherheitsbeweise in der Kryptographie sind oft kompliziert und komplex, was es erschwert diese zu belegen. Aus diesem Grund hat sich die Darstellung von Sicherheitsbeweisen mit Hilfe von Sequenzen von Spielen zu einer weit verbreiteten Methode entwickelt. Mit dieser Darstellung kann die Komplexität der Beweise im Rahmen gehalten werden, wodurch die Beweise selbst auch weniger fehleranfällig und einfacher zu verifizieren werden. Außerdem ist diese Darstellung in vielen Szenarien einsetzbar und ermöglicht somit eine einheitliche Struktur bei verschiedenen Beweisen.

Es existiert jedoch nicht eine einzige, feste Definition für Beweise mit Sequenzen von Spielen, stattdessen gibt es verschiedene Ausführungen, die sich in Strenge bezüglich Vorschriften und Formalität unterscheiden. Das Standardmodell ist das *Random-Oracle Modell*, eine andere Ausführung ist beispielsweise das *Ideal-Blockcipher Modell*.

2.3.2 Random-Oracle Modell

Beim Random-Oracle Modell werden Sequenzen von Spielen meist in Form eines Angriffsspiels (engl. „attack game“) zwischen einem Widersacher (engl. „adversary“), auch Angreifer genannt, und einem Herausforderer (engl. „challenger“), die miteinander kommunizieren, definiert. Im Rahmen des Beweises wird dieses Angriffsspiel immer weiter verfeinert.

Sowohl dieses Spiel G als auch der Widersacher A sind probabilistische Programme, dargestellt als Sammlung von Prozeduren, bzw. einer einzigen Prozedur im Fall des Widersachers. Diese Programme werden in der Regel in Form von Pseudocode definiert, wobei sowohl einfache Ausdrücke wie *if* und *for* oder Variablenzuweisungen, als auch eine zufällige Variablenzuweisung erlaubt sind. Bei letzterer wird aus einer endlichen Menge S ein zufälliges Element ausgewählt und in einer Variablen gespeichert, formal notiert als $s \xleftarrow{\$} S$, wobei s die Variable ist, in der das gewählte Element aus S gespeichert wird.

Das Spiel G besteht aus einer Initialisierungsprozedur *„Initialize“*, einer Abschlussprozedur *„Finalize“*, sowie weiteren Prozeduren, die als Orakel (engl. „oracle“) bezeichnet werden.

Der Ablauf eines Spiels beginnt mit der Ausführung der *„Initialize“*-Prozedur in G. Diese kann eine Ausgabe, bezeichnet als *inp*, produzieren, die dann an den Widersacher A weitergegeben wird. Im nächsten Schritt wird der Widersacher gestartet, woraufhin dieser die Orakel-Prozeduren in G ausführen und den jeweiligen Rückgabewert dieser erhalten kann. Formal geschrieben $y \leftarrow P(...)$ für eine Orakel-Prozedur P, die in G definiert ist, und deren Rückgabewert in y gespeichert wird.

Sobald der Widersacher A hält, wird die *„Finalize“*-Prozedur in G aufgerufen. Gibt A einen Rückgabewert *out* zurück, so wird dieser an die Prozedur weitergegeben. Schließlich gibt die *„Finalize“*-Prozedur selbst ihren Rückgabewert *outcome* zurück. Dieser wird als Ergebnis des Spiels G bezeichnet.

Sowohl das Spiel G, als auch der Widersacher A müssen eine endliche Laufzeit haben, also nach einer endlichen Zahl Programmschritte terminieren. Diese Bedingung muss unabhängig von eventuellen Zufallsergebnissen oder Orakelrückgaben gelten. Daraus folgt, dass die Anzahl möglicher Zufallsereignisse in beiden Programmen beschränkt ist. Mehr zu probabilistischen Algorithmen findet sich in Kapitel 4.2.2.

Definition 2.9 (Spiel [3])**.** *Ein Spiel $G = (Initialize, P_1, \ldots, P_n, Finalize)$ ist ein Programm, bestehend aus einer bestimmten Menge an Prozeduren, das ein Angriffsspiel zwischen einem Angreifer $\mathcal{A}$ und einem Herausforderer simuliert. Die Prozedurenmenge besteht aus einer Initialisierungs- und einer Abschlussprozedur (Initialize, Finalize), sowie n weiteren Orakelprozeduren $P_1, \ldots, P_n$. Zum Schluss gibt das Spiel einen Rückgabewert outcome zurück.*

2.3.3 Sicherheitsbeweise durch Spielsequenzen

Eine Sicherheitsdefinition enthält stets eine Ganzzahl, die gegen unendlich tendiert und als „Sicherheitsparameter“ bezeichnet wird. Mit dieser lassen sich dann die Begriffe „effizient“ und „vernachlässigbar nah“ definieren.

Definition 2.10 (Effizient)**.** *Eine Laufzeit wird als effizient bezeichnet, wenn diese durch ein Polynom im Sicherheitsparameter beschränkt ist.*

Definition 2.11 (Vernachlässigbar nah)**.** *Eine Funktion $f\colon \mathbb{N} \to \mathbb{R}$ heißt vernachlässigbar (engl. „negligible“), wenn für jedes positive Polynom p ein N existiert, sodass für alle $n > N$ gilt [13]:*

$$f(n) < \frac{1}{p(n)}$$

Ein Wert liegt vernachlässigbar nah (engl. „negligible close“) an einem anderen, wenn die Differenz dieser kleiner als die Inverse eines jeglichen Polynoms im Sicherheitsparameter ist.

Die Sicherheitsdefinition ist außerdem meist abhängig von einem bestimmten Ereignis S. Sicherheit ist dann gegeben, wenn die Wahrscheinlichkeit, mit der dieses Ereignis S eintritt, vernachlässigbar nah an einer festgelegten Zielwahrscheinlichkeit ist. Dieser Wert liegt häufig bei $\frac{1}{2}$ oder 0.

Um diese Sicherheit mit Sequenzen von Spielen beweisen zu können, werden zunächst die Spiele, bezeichnet als Spiel 0 bis Spiel n, konstruiert. Dabei ist Spiel 0 das ursprüngliche Angriffsspiel. Das Ereignis S wird nun als S_0 bezeichnet und daraufhin für $i = 1, \ldots, n$ das Ereignis S_i, das in seiner Definition ähnlich zu S ist, in Spiel i definiert.

Mit Hilfe des Beweises soll gezeigt werden, dass $Pr[S_n]$ vernachlässigbar nah bei der Zielwahrscheinlichkeit liegt oder gleich dieser ist, sowie dass für $i = 0, \ldots, n-1$ gilt:

$Pr[S_i]$ liegt vernachlässigbar nah bei $Pr[S_{i+1}]$. Weil n konstant ist folgt dann, dass auch $Pr[S]$ vernachlässigbar nah bei der Zielwahrscheinlichkeit liegt oder gleich dieser ist. Dadurch ist die Sicherheit dann bewiesen.

Die Unterschiede, auch Übergänge (engl. „transitions“) genannt, zwischen Spielen sollten möglichst klein sein und lassen sich auf drei Arten reduzieren.

Übergänge basierend auf *Ununterscheidbarkeit*: Bei diesen Übergängen wird die Eingabe, die der Angreifer bekommt, verändert. Statt diese aus der ursprünglichen Menge zu entnehmen, wird sie aus einer anderen Menge entnommen. Diese beiden Mengen müssen *ununterscheidbar* sein [10].

Definition 2.12 (Ununterscheidbare Mengen [13])**.** *Seien zwei Mengen X_1, X_2, sowie ein Algorithmus D gegeben. D gibt den Wert 1 bei Eingabe eines Elementes $x \in X_1$ mit einer Wahrscheinlichkeit von $Pr[x \in X_1 \colon D(x) = 1]$ und bei Eingabe eines Elementes $x \in X_2$ mit einer Wahrscheinlichkeit von $Pr[x \in X_2 : D(x) = 1]$ aus.*

X_1 und X_2 sind ununterscheidbar, wenn die Differenz dieser beiden Wahrscheinlichkeiten $|Pr[x \in X_1 \colon D(x) = 1] - Pr[x \in X_2 : D(x) = 1]|$ vernachlässigbar klein ist.

Da beide Mengen, aus denen der Eingabewert des Angreifers entnommen wird, ununterscheidbar sein sollen, folgt aus Definition 2.12 dass die Wahrscheinlichkeit, dass ein Algorithmus diese unterscheiden kann vernachlässigbar klein ist. Da auch der Angreifer ein Algorithmus ist, folgt daraus, dass der Mehrerfolg des Angreifers ebenfalls vernachlässigbar ist.

Übergänge basierend auf einem *Fehlerereignis*: Bei diesen Übergängen verhalten sich zwei aufeinanderfolgende Spiele $i, i+1$ identisch, bis ein bestimmtes Fehlerereignis F auftritt. Die Auftrittswahrscheinlichkeit von F wird dabei als vernachlässigbar klein angenommen [10].

Die Aussage die Spiele i und $i+1$ seien bis zu einem Ereignis F identisch bedeutet: $S_i \wedge \neg F \leftrightarrow S_{i+1} \wedge \neg F$, wobei S_i und S_{i+1} die in den Spielen $i, i+1$ definierten Ereignisse sind. Daraus folgt dann das „Difference Lemma“:

Lemma (Difference Lemma [10])**.** *Sei F ein Fehlerereignis, sodass $S_i \wedge \neg F \leftrightarrow S_{i+1} \wedge \neg F$ gilt. Dann gilt:*

$$|Pr[S_i] - Pr[S_{i+1}]| \leq Pr[F]$$

Aus dem Difference Lemma folgt dann wiederum, dass der Mehrerfolg des Angreifers vernachlässigbar ist.

Überbrückungsschritte: Bei Überbrückungsschritten erfolgt keine Änderung aus Sicht des Angreifers. Änderungen sind nur konzeptuell, daher gilt: $Pr[S_i] = Pr[S_{i+1}]$. Überbrückungsschritte dienen meist als Vorbereitungsschritte für einen darauffolgenden, auf Ununterscheidbarkeit oder einem Fehlerereignis basierenden Schritt.

Ein Sicherheitsbeweis mittels einer Sequenz von Spielen wird in Kapitel 4.6.2 gezeigt.

3 Ausgewählte Kryptosysteme

3.1 AES

Die folgende Beschreibung des AES-Algorithmus basiert auf der Beschreibung von Albrecht Beutelspacher, Heike Neumann und Thomas Schwarzpaul [6].

3.1.1 Geschichte

Das „National Institute of Standards and Technology“ (kurz: NIST) suchte einen Nachfolger für den als nicht mehr sicher geltenden *Digital Encryption Standard* (DES) und suchte daher mit einer am 12. September 1998 veröffentlichten Ausschreibung nach Vorschlägen für einen neuen, sichereren Standard.

Für das neue Block-Kryptosystem, genannt *Advanced Encryption Standard* (AES) wurden insgesamt 25 Kandidaten eingereicht. Am 2. Oktober 2000 [14] wurde schließlich der *Rijndael*-Algorithmus von Joan Daemon und Vincent Rijmen als Sieger der Ausschreibung ausgewählt. Die eingereichten Kandidaten wurden zuvor in mehreren Analyserunden untersucht. Der Rijndael-Algorithmus überzeugte auch auf Grund seiner hohen Effizienz.

AES wird heutzutage sowohl in Behörden, als auch z. B. in Internetprotokollen verwendet [14].

3.1.2 Beschreibung

AES ist ein Block-Kryptosystem. Es existieren verschiedene Varianten, die sich in der Schlüssellänge unterscheiden [14]. Während die Blocklänge auf 128 Bit festgelegt ist, können die Schlüssel Längen von 128, 192 oder 256 Bit annehmen. Zur Unterscheidung dieser Varianten wird daher auch entsprechend der verwendeten Schlüssellänge von AES-128, AES-192 bzw. AES-256 gesprochen.

Der Algorithmus besteht im Wesentlichen aus mehreren Runden. Die Anzahl dieser hängt dabei von der benutzten Schlüssellänge ab: Bei 128 Bit werden 10 Runden, bei 192 Bit 12 Runden und bei 256 Bit 14 Runden ausgeführt.

Definition 3.1 (Byte)**.** *Ein Byte ist ein Bitvektor, bestehend aus 8 Bit.*

Definition 3.2 (Wort [8])**.** *Ein Wort ist ein Bitvektor, bestehend aus 32 Bit.*

Der Klartext wird im AES-Verfahren als (4×4)-Byte-Matrix M gespeichert, die in jeder Zelle 1 Byte des Klartextes enthält. Rundenschlüssel werden in selbiger Form gespeichert.

Jede Runde des Algorithmus besteht aus vier Schritten, genannt *SubBytes*, *ShiftRow*, *MixColumn* und *AddRoundKey*. Die Ausnahme bildet lediglich die letzte Runde, dort wird der *MixColumn*-Schritt ausgelassen.

Vor der ersten Runde wird zudem eine „nullte“ Runde ausgeführt. In dieser wird der Klartext mit dem „nullten Rundenschlüssel“ mittels XOR verknüpft. Durch den Einsatz dieses geheimen Schlüssels wird verhindert, dass ein Angreifer die ersten drei Schritte einfach nachmachen kann.

Der AES-Algorithmus arbeitet auf dem Galoiskörper $GF(2^8)$, somit sind alle Bytes Elemente dieses Körpers. Dadurch ist es möglich, Bytes nicht nur zu addieren, sondern auch zu multiplizieren und zu invertieren, sofern sie ungleich 0 sind.
Dazu werden die Bytes als Polynome vom Grad ≤ 7 mit Koeffizienten aus $GF(2)$ dargestellt.

Satz 3.1. *Sei $GF(2)[x]$ der Ring aller Polynome mit Koeffizienten aus $GF(2)$ und m ein irreduzibles Polynom aus $GF(2)[x]$ mit Grad 8. Dann ist der aus 256 Elementen bestehende Körper $GF(2)[x]/(m)$ die Menge aller Polynome mit Grad ≤ 7 [6].*

Im AES-Algorithmus wird m gewöhnlich definiert als

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

Die Zuordnung eines Bytes zu einem Polynom erfolgt indem die Bits als Koeffizienten des Polynoms verwendet werden. So entspricht z. B. das Byte 1001 1001 dem Polynom $x^7 + x^4 + x^3 + 1$.

Die Addition zweier Bytes erfolgt dann, wie die gewöhnliche Addition über $GF(2^8)$, indem die Koeffizienten der Bytes jeweils mit XOR verknüpft werden [11]. Bei der Multiplikation ist zu beachten, dass stets mit modulo $m(x)$ gerechnet werden muss.

Definition 3.3 (AES-Algorithmus)**.** *Der AES-Algorithmus ist ein Substitutions-Permutations-Netzwerk, das auf dem Galoiskörper $GF(2^8)$ arbeitet. Abhängig von der Schlüssellänge werden 10, 12, oder 14 Runden ausgeführt, die mit Ausnahme der letzten Runde aus den Schritten SubBytes, ShiftRow, MixColumn und AddRoundKey bestehen. In der letzten Runde entfällt der MixColumn-Schritt.*

SubBytes

Der *SubBytes*-Schritt ist eine Substitutionsabbildung auf $GF(2^8)$ und der einzige nichtlineare Teil des Algorithmus.

Auf jedes Byte a werden dabei zwei Operationen angewandt. Zunächst wird die Inverse gebildet, anschließend die gebildete Inverse mit einer binären Matrix A multipliziert, sowie ein Byte b addiert.

Definition 3.4 (SubBytes)**.** *Im SubBytes-Schritt wird jedes Byte a der Matrix neu berechnet durch:*

$$a \leftarrow A \cdot a^{-1} \oplus b$$

Die Matrix A, sowie das Byte b werden durch AES vorgegeben [8]:

$$A = \begin{pmatrix} 1 & 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ 1 & 1 & 0 & 0 & 0 & 1 & 1 & 1 \\ 1 & 1 & 1 & 0 & 0 & 0 & 1 & 1 \\ 1 & 1 & 1 & 1 & 0 & 0 & 0 & 1 \\ 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 & 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 & 1 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 \end{pmatrix}, b = \begin{pmatrix} 1 \\ 1 \\ 0 \\ 0 \\ 0 \\ 1 \\ 1 \\ 0 \end{pmatrix}$$

Die Nichtlinearität der Substitution ist ein wichtiger Faktor bezüglich der Sicherheit der Chiffre.

ShiftRow

Der *ShiftRow*-Schritt ist eine Permutation auf den Zeilen der Matrix und besteht ausschließlich aus Linksshifts.

Dabei werden die Bytes in der Matrix zyklisch nach links verschoben. Die Anzahl der Positionen, um die verschoben wird, hängt dabei von der Zeile, in der sich das jeweilige Byte in der Matrix befindet, ab: Die erste Zeile wird nicht verschoben, die zweite Zeile hingegen um 1 Byte, die dritte Zeile um 2 Byte und die vierte Zeile um 3 Byte.

Definition 3.5 (ShiftRow)**.** *Im ShiftRow-Schritt wird jede Zeile i, $0 \leq i \leq 3$ der Matrix um i Positionen nach links verschoben.*

MixColumn

Der *MixColumn*-Schritt operiert auf den Spalten der Matrix. Jede dieser Spalten wird dafür als Polynom in $GF(2^8)$ mit Grad ≤ 3 interpretiert, wobei die vier in der jeweiligen Spalte enthaltenen Bytes die Koeffizienten des Polynoms sind. Diese Polynome der Spalten werden dann mit einem weiteren, fest definierten Polynom $c(x)$ multipliziert.

Definition 3.6 (MixColumn)**.** *Im MixColumn-Schritt wird zunächst jede Spalte s_i, mit $0 \leq i < 4$, der Matrix als Polynom*

$$a_{3i}x^3 + a_{2i}x^2 + a_{1i}x + a_{0i}$$

interpretiert, wobei die Koeffizienten $a_{0i}, a_{1i}, a_{2i}, a_{3i}$ die Bytes der Spalte s_i sind.

Anschließend wird jede Spalte neu berechnet durch:

$$s_i \leftarrow ((s_i \cdot c(x)) \mod (x^4 + 1))$$

Das Polynom $c(x)$ ist fest definiert als

$$c(x) = \{03\} \cdot x^3 + \{01\} \cdot x^2 + \{01\} \cdot x + \{02\}$$

$\{01\}, \{02\}, \{03\}$ sind dabei sogenannte Halbbytes. Die erste Zahl gibt die ersten vier und die zweite Zahl die letzten vier Bits an, so entspricht z. B. $\{02\}$ dem Byte 0000 0010.

Da $x^4 + 1$ und $c(x)$ teilerfremde Polynome sind, lässt sich die Multiplikation mit $c(x)$ bei der Entschlüsselung wieder rückgängig machen.

AddRoundKey

Im *AddRoundKey*-Schritt wird der Rundenschlüssel mittels XOR-Verknüpfung mit der Matrix verknüpft.

Definition 3.7 (AddRoundKey)**.** *Die Matrix wird im AddRoundKey-Schritt bitweise mit dem aktuellen Rundenschlüssel mittels einer XOR-Verknüpfung verknüpft:*

$$M \leftarrow M \oplus k_i$$

k_i gibt den i-ten Rundenschlüssel, der entsprechend in Runde i verwendet wird, an.

3.1.3 Schlüsselexpansion

Jeder Rundenschlüssel muss dieselbe Länge wie die Matrix haben. Es werden also insgesamt $(\textit{Rundenzahl} + 1) \cdot 128$ Bit Schlüsselbits benötigt. Bei einer Schlüssellänge von 128 Bit wären also $(10 + 1) \cdot 128 = 1408$ Bit benötigt, was 44 Wörtern entspricht.
Da der gegebene Schlüssel jedoch nicht genügend Bits liefert, muss dieser zunächst erweitert werden.

Bei der Schlüsselexpansion werden aus vorhandenen Schlüsselteilen neue Schlüsselteile zusammengesetzt. Der Vorgang ist dabei abhängig von der gewählten Schlüssellänge. Für die Schlüssellänge 128 Bit erfolgt dies folgendermaßen:

Der 128 Bit lange Schlüssel besteht aus 4 Worten W_0, W_1, W_2, und W_3. Das sechste Wort W_5 wird dann beispielsweise errechnet durch $W_5 = W_2 \oplus W_4$. Die weiteren Worte werden nach demselben Schema berechnet: $W_i = W_{i-3} \oplus W_{i-1}$.

Ausnahmen sind die Worte W_i mit $i = 4, 8, 12, 16, \ldots$. Für diese gilt, dass zunächst W_{i-1} um 4 Stellen rotiert, dann *SubBytes* angewendet und schließlich die Rundenkonstante addiert wird.

Die Wörter werden der Reihenfolge nach als Rundenschlüssel verwendet.

3.2 RSA

Die folgende Beschreibung des RSA-Algorithmus basiert auf der Beschreibung von Albrecht Beutelspacher, Heike Neumann und Thomas Schwarzpaul [6].

3.2.1 Überblick

RSA ist ein asymmetrisches Kryptosystem, das 1978 von Ron Rivest, Adi Shamir und Len Adleman entwickelt und nach diesen auch benannt wurde.

Es erreicht seine Sicherheit vor allem durch die hohe Schwierigkeit Zahlen zu faktorisieren: Es ist leicht Zahlen zu multiplizieren, jedoch ist kein effizienter Algorithmus bekannt, der die Primfaktorzerlegung einer Zahl angeben kann.

Neben der Verwendung als asymmetrisches Verschlüsselungsverfahren wird RSA auch für digitale Signaturen eingesetzt.

3.2.2 Schlüsselerzeugung

Jeder Teilnehmer des Nachrichtenaustausches wählt als Erstes zwei Primzahlen p und q und bildet aus diesen das Produkt

$$n = p \cdot q$$

Bei der Wahl der Primzahlen sollte beachtet werden, dass die Faktorisierung dieser möglichst schwierig sein sollte. Daher sollten die Zahlen möglichst groß sein, um ein Ausprobieren aller Teiler von n unmöglich zu machen. Aus diesem Grund werden heutzutage üblicherweise Primzahlen mit Längen von 1024 Bit verwendet, was Zahlen der Größenordnung 2^{1024} entspricht.

Die gewählten Primzahlen p und q müssen geheim gehalten werden.

Anschließend bestimmt jeder Teilnehmer mit der Eulerschen Phi-Funktion die Anzahl der zu n teilerfremden Zahlen, die kleiner als n sind.

Definition 3.8 (Eulersche Phi-Funktion)**.** *Sei n eine natürliche Zahl. Die Eulersche Phi-Funktion $\phi(n)$ gibt die Anzahl der zu n teilerfremden Zahlen, die kleiner als n sind, an.*

Satz 3.2 ([17])**.** *Seien n, m teilerfremde, natürliche Zahlen. Da die Eulersche Phi-Funktion multiplikativ ist, gilt: $\phi(m \cdot n) = \phi(m) \cdot \phi(n)$.*

Satz 3.3 ([17])**.** *Für eine Primzahl p gilt: $\phi(p) = p - 1$.*

Da p und q Primzahlen sind, sind sie auch teilerfremd. Somit gilt:

$$\phi(n) = \phi(p \cdot q) = \phi(p) \cdot \phi(q) = (p-1) \cdot (q-1)$$

Daraufhin wählt jeder Teilnehmer zwei Zahlen e und d, mit

$$ed \equiv 1 \mod \phi(n)$$

Die Zahl e kann prinzipiell frei gewählt werden, muss aber teilerfremd zu $\phi(n)$ sein. Zahl d kann dann durch Anwendung des *erweiterten Euklidschen Algorithmus* auf e und $\phi(n)$ berechnet werden.

Der öffentliche Schlüssel setzt sich dann aus e und n zusammen, während der private Schlüssel d ist.

Definition 3.9 (RSA-Schlüsselerzeugung)**.** *Jeder Teilnehmer wählt zunächst zwei Primzahlen p, q und berechnet anschließend*

$$n = p \cdot q,\ \phi(n) = (p-1) \cdot (q-1)$$

Daraufhin werden e, d so gewählt, dass

$$ed \equiv 1 \mod \phi(n)$$

gilt, wobei e teilerfremd zu $\phi(n)$ sein muss.

Der öffentliche Schlüssel setzt sich schließlich aus e und n, der private Schlüssel ausschließlich aus d zusammen.

3.2.3 Verschlüsselung

Zur Verschlüsselung wird der öffentliche Schlüssel (n, e) des Nachrichtenempfängers verwendet. Die Verschlüsselung einer Nachricht m erfolgt dann durch

$$c = m^e \mod n$$

3.2.4 Entschlüsselung

Die Entschlüsselung eines Geheimtextes c geschieht durch

$$m = c^d \mod n$$

Dazu verwendet der Empfänger seinen eigenen privaten Schlüssel, also die bei der Schlüsselerzeugung berechnete Zahl d. Die Korrektheit der Entschlüsselung wird dabei durch den *Satz von Euler* garantiert. Der Satz von Euler besagt, dass für zwei Zahlen e, d mit $ed \equiv 1 \mod \phi(n)$ gilt:

$$m^{ed} \mod n = m$$

falls m und n teilerfremd sind.

3.3 ElGamal

Die folgende Beschreibung des ElGamal-Algorithmus basiert auf der Beschreibung von Ralf Küsters und Thomas Wilke [14].

3.3.1 Überblick

Das asymmetrische ElGamal-Kryptosystem wurde von Taher ElGamal entwickelt und 1984 veröffentlicht. Die Grundlage des Kryptosystems ist die Diffie-Hellman-Schlüsselvereinbarung.

3.3.2 Diffie-Hellman-Schlüsselvereinbarung

Um einen gemeinsamen Schlüssel zu vereinbaren, wählen die Teilnehmer A und B zuerst eine zyklische Gruppe $\mathcal{G}$. Außerdem muss ein Erzeugerelement g aus $\mathcal{G}$ gewählt werden. Das Element g ist ein Erzeuger, wenn sich jedes Element aus $\mathcal{G}$ als Potenz von g darstellen lässt.

Teilnehmer A wählt dann die geheime Zahl $a \in \{0, \ldots, |\mathcal{G}| - 1\}$ und sendet g^a an Teilnehmer B. Dieser wählt ebenso eine geheime Zahl $b \in \{0, \ldots, |\mathcal{G}| - 1\}$ und sendet g^b an Teilnehmer A.

Nach dem Empfang der jeweiligen Zahlen können A und B schließlich einen gemeinsamen Schlüssel berechnen. Beide potenzieren jeweils den empfangenen Wert mit ihrer geheimen Zahl. Teilnehmer A berechnet $(g^b)^a$ und Teilnehmer B $(g^a)^b$. Das Ergebnis beider Berechnungen ist identisch:

$$(g^b)^a = (g^a)^b = g^{ab}$$

Für geeignete Gruppen $\mathcal{G}$ ist die Berechnung von g^{ab} mittels g^a und g^b ein schweres Problem. Für einen Angreifer mit beschränkten Ressourcen ist es dann nicht realistisch möglich g^{ab} zu bestimmen.

Definition 3.10 (Diffie-Hellman-Schlüsselvereinbarung)**.** *Sei $\mathcal{G}$ eine zyklische Gruppe, g ein Erzeugerelement von $\mathcal{G}$. Teilnehmer A wählt $a \in \{0, \ldots, |\mathcal{G}| - 1\}$ und bestimmt g^a, Teilnehmer B wählt $b \in \{0, \ldots, |\mathcal{G}| - 1\}$ und bestimmt g^b. Der gemeinsame Schlüssel von A, B ist dann g^{ab}.*

3.3.3 Verschlüsselung

Möchte Teilnehmer A eine Nachricht $m \in \mathcal{G}$ verschlüsseln und an Teilnehmer B senden, wählen diese zunächst, wie im letzten Abschnitt beschrieben, ihre privaten Schlüssel a, b und berechnen anschließend jeweils ihre öffentlichen Schlüssel g^a und g^b.

Der Geheimtext $c = (c_1, c_2)$ setzt sich aus zwei Teilen zusammen. Teil c_1 ist der Schlüsselanteil von Teilnehmer A und entspricht dem öffentlichen Schlüssel. Teil c_2 ist die Verschlüsselung des Klartextes m mit dem gemeinsamen Schlüssel.

Definition 3.11 (ElGamal-Verschlüsselung)**.** *Sei $\mathcal{G}$ eine zyklische Gruppe, g ein Erzeugerelement von $\mathcal{G}$. Ein Klartext m wird mittels des öffentlichen Schlüssels g^a und des gemeinsamen Schlüssels g^{ab} zu Geheimtext c verschlüsselt:*

$$c = (c_1, c_2), c_1 = g^a, c_2 = m \cdot g^{ab}$$

3.3.4 Entschlüsselung

Um einen empfangenen Geheimtext $c = (c_1, c_2)$ zu entschlüsseln und den Klartext m zu bekommen, muss Teilnehmer B seinen privaten Schlüssel b einsetzen.

Definition 3.12 (ElGamal-Entschlüsselung)**.** *Ein Geheimtext $c = (c_1, c_2)$ wird mit Hilfe des privaten Schlüssels b zu Klartext m entschlüsselt durch:*

$$m = c_2 \cdot ((c_1)^b)^{-1}$$

Mit $((c_1)^b)^{-1}$ wird das inverse Element zu $(c_1)^b$ in $\mathcal{G}$ bezeichnet.

4 Ununterscheidbarkeit

4.1 Motivation

Die Vertraulichkeit zählt, neben der Authentizität und der Integrität, zu den Zielen der Kryptographie. Ein „sicheres“ Kryptosystem sollte dieses Sicherheitsziel also erfüllen [6].

Doch bevor es möglich ist die Sicherheit eines Kryptosystems zu überprüfen, muss der Begriff „Sicherheit“ zunächst genauer definiert werden.

Das Kerckhoffs'sche Prinzip besagt, dass die Sicherheit eines Verschlüsselungsverfahrens einzig von der Geheimhaltung des Schlüssels, nicht aber von der Geheimhaltung des verwendeten Verfahrens zur Verschlüsselung abhängig sein darf. Daher sollte bei einer Sicherheitsanalyse stets angenommen werden, dass ein Angreifer das Verfahren kennt [14].

Daraus folgt ebenfalls, dass der verwendete Schlüssel logischerweise nicht aus einem oder mehreren Geheimtexten ableitbar sein darf. Selbiges gilt auch für einen Klartext: Dieser darf nicht ohne den Schlüssel aus einem Geheimtext reproduzierbar sein. Aber ist es bereits ausreichend sicherzustellen, dass lediglich der gesamte Geheimtext nicht ohne Schlüssel entschlüsselt werden kann? Es ist denkbar, dass bereits Teile des Klartextes, z. B. die ersten oder die letzten Zeichen der Nachricht, sensible Informationen enthalten können. Daher muss ein sicheres Kryptosystem sicherstellen, dass es auch nicht möglich ist, nur Teile der Nachricht zu entschlüsseln [14].

Auch der Angreifer, gegen den das Kryptosystem sicher sein soll, muss festgelegt werden. Zu welchen Angriffen auf das Kryptosystem soll dieser fähig sein? Welche Ressourcen stehen ihm dabei zur Verfügung? [14]

Zwei Eigenschaften sind also bei der Betrachtung der Sicherheit wichtig: Erstens das Szenario, in dem ein potentieller Angriff ausgeführt wird, und zweitens die Menge der Informationen, die ein Angreifer mit einem solchen Angriff gewinnen kann [6].

Es soll im Folgenden das Sicherheitsmodell der Ununterscheidbarkeit (engl. „indistinguishability“, kurz IND) betrachtet werden.

4.2 Sicherheit durch Ununterscheidbarkeit

Zunächst sollen die zwei zuvor genannten Sicherheitseigenschaften, Szenario und Informationsmenge, bezüglich des Modells der Ununterscheidbarkeit betrachtet werden. Die Betrachtungen folgen den Arbeiten von Mihir Bellare und Phillip Rogaway [2], sowie Albrecht Beutelspacher, Heike Neumann und Thomas Schwarzpaul [6].

Das Szenario betrachtet einen Nachrichtenaustausch zwischen zwei Teilnehmern, die entweder ein symmetrisches oder asymmetrisches Kryptosystem verwenden. Beide Varianten werden später betrachtet. Das bereits erwähnte Kerckhoffs'sche Prinzip soll zudem als erfüllt angenommen werden, was bedeutet dass auch dem Angreifer das verwendete Kryptosystem bekannt ist.

Außerdem wird angenommen, dass der Angreifer übertragene Geheimtexte abfangen kann. Aus diesen kann er dann versuchen Informationen zu gewinnen.

Bezüglich der Informationsmenge, die ein Angreifer bei einem Angriff gewinnen kann, ist es logischerweise erstrebenswert, dass das geprüfte Kryptosystem auch bei einem starken Angriff möglichst wenig Informationen offenbart. Weder der gesamte Klartext, noch Teile dessen dürfen aus dem Geheimtext rekonstruierbar sein.

Theoretisch ist es gar erstrebenswert, dass ein Angreifer nichts über den Klartext weiß und auch keinerlei neue Informationen gewinnen kann. Praktisch werden dem Angreifer jedoch stets „a priori" Informationskenntnisse zugesprochen. Grund dafür ist, dass es wahrscheinlich ist, dass ein Angreifer diese „a priori" Informationen, wie z. B. ein eventuelles Headerformat der Nachricht, in einem realen Szenario besitzen würde.

Die Länge des Klartextes wird ebenfalls als dem Angreifer bekannt vorausgesetzt. Daher muss diese Information von der Verschlüsselung nicht geschützt werden.

In einem sicheren Kryptosystem darf es nicht möglich sein, die Geheimtexte zweier Klartexte gleicher Länge diesen zuzuordnen. Dieser Umstand wirkt sich direkt auf die Art und Weise, wie Nachrichten verschlüsselt werden sollten aus: Wird eine Nachricht stets zum selben Geheimtext verschlüsselt, kann ein Angreifer z. B. zuvor bereits gesendete Geheimtexte wiedererkennen. Dies ist ein Problem, wenn diese bereits entschlüsselt worden sind.

Dass ein Klartext nicht stets zum selben Geheimtext verschlüsselt wird, sondern mehrere Geheimtexte haben muss, bedingt, dass die Verschlüsselung in einem in diesem Konzept sicheren Kryptosystem nicht deterministisch, sondern zufalls- oder zustandsgesteuert sein muss. Erfolgreiche Angriffe gegen deterministische Verschlüsselungen in sowohl symmetrischen, als auch asymmetrischen Kryptosystemen werden in den Abschnitten 4.5.2 und 4.6.1 gezeigt.

Deterministische und zufallsgesteuerte Algorithmen sollen im Folgenden betrachtet werden.

4.2.1 Deterministische Algorithmen

Ein deterministischer Algorithmus gibt bei Eingabe x stets dieselbe Ausgabe y aus, da diese durch eine bestimmte Anzahl zuvor festgelegter Schritte aus der Eingabe x ermittelt wird. Daher ist es möglich, einen solchen Algorithmus als einfache mathematische Abbildung $A\colon X \to Y$ zu schreiben, wobei A der deterministische Algorithmus, X die Menge der Eingaben und Y die Menge der Ausgaben ist [9].

4.2.2 Probabilistische Algorithmen

Die Definition probabilistischer Algorithmen folgt [14] und [9].

Ein probabilistischer, also zufallsgesteuerter Algorithmus unterscheidet sich gegenüber einem deterministischen Algorithmus dadurch, dass sein Verhalten zum Teil durch zufallsbasierte Ergebnisse beeinflusst wird. Das bedeutet die Ausgabe y ist neben der Eingabe x zusätzlich von den Ergebnissen einer endlichen Menge zufälliger „Experimente“ abhängig. Dadurch kann ein probabilistischer Algorithmus trotz identischer Eingabe x bei verschiedenen Durchläufen verschiedene Ausgaben liefern.

Diese Experimente werden als Simulationen von Münzwürfen (engl. „coin tossing“) dargestellt. Dabei wird das Ergebnis eines Wurfes einer idealen Münze betrachtet, was bedeutet, dass die Auftrittswahrscheinlichkeit beider Seiten der Münze, also Kopf oder Zahl bzw. 0 oder 1, bei $\frac{1}{2}$ liegt und somit identisch ist. Zudem sind verschiedene, aufeinanderfolgende Münzwürfe voneinander unabhängig [8].
In Algorithmen werden solche Münzwürfe durch die Funktion $flip()$ dargestellt. Die $flip()$-Funktion simuliert einen solchen Münzwurf und liefert dadurch ein Zufallsbit zurück. Zudem wird die Funktion $flip(l)$ für $l \geq 0$ definiert, welche einen zufälligen Bitvektor der Länge l zurückgibt.

Ein probabilistischer Algorithmus kann während seiner Laufzeit eine endliche Menge an Münzwürfen simulieren. Dabei kann entweder ein nachfolgender Schritt oder auch die Anzahl der insgesamt simulierten Münzwürfe vom Ergebnis eines zuvor simulierten Münzwurfs abhängig sein. Die Laufzeit des Algorithmus ist jedoch durch eine vom Eingabewert x abhängigen Konstante t_x beschränkt. Da die Ausführung der $flip()$-Funktion genau einen und die Ausführung der $flip(l)$-Funktion entsprechend genau l Programmschritte benötigt, kann der Algorithmus also insgesamt t_x Zufallsbits generieren. Die Konstante t_x wird auch einfach als t bezeichnet.
Für die Umsetzung dieser Zufallsereignisse steht einem probabilistischen Algorithmus eine Folge $\alpha \in \{0,1\}^t$, bestehend aus t Zufallsbits, zur Verfügung. Die Funktionen $flip()$ und $flip(l)$ lesen den nächsten Bit bzw. die nächsten l Bits von α aus und geben diese zurück. Dadurch wird kein Zufallsbit aus α mehrfach zurückgegeben, gleichzeitig ist aber auch sichergestellt, dass in jedem Fall ausreichend Zufallsbits zur Verfügung stehen, da $|\alpha| = t$ ist und, wie beschrieben, die Zahl der maximal benötigten Zufallsbits durch die Laufzeit in t beschränkt ist. Somit können vom Algorithmus nicht mehr als die in α enthaltenen t Zufallsbits angefordert werden.

In Algorithmus 1 ist ein Beispiel für einen probabilistischen Algorithmus gegeben, welcher zunächst zwei Zufallsbits x, y generiert und abhängig dieser entweder 0 oder einen zufälligen Bitvektor z der Länge 3 zurückgibt.

Algorithmus 1 Beispiel: Probabilistischer Algorithmus

```
function ProbAlgorithm()
    x ← flip()
    y ← flip()

    if x = y then
        return 0
    else
        z ← flip(3)                          ▷ flip(l) mit l = 3
        return z
    end if
end function
```

4.2.3 Probabilistische Algorithmen in der Kryptographie

In der Kryptographie werden häufig probabilistische statt deterministischer Algorithmen eingesetzt, sowohl bei Verschlüsselungsalgorithmen, wie z. B. dem One-Time-Pad, als auch bei der Sicherheitsanalyse, wie z. B. bei Algorithmen für Angreifer. [9]

Basierend auf zufalls- oder zustandsgesteuerten Algorithmen wird nun die Definition für Kryptosysteme angepasst. Solche Kryptosysteme werden auch als algorithmische Kryptosysteme bezeichnet. Die Definition dieser folgt [2].

Algorithmische Kryptosysteme

Ein symmetrisches, algorithmisches Kryptosystem $\mathcal{SE}$ ist definiert als 3-Tupel $\mathcal{SE} = (\mathcal{K}, \mathcal{E}, \mathcal{D})$ bestehend aus drei Algorithmen:

Der zufallsgesteuerte Schlüsselgenerierungsalgorithmus $\mathcal{K}$ liefert einen Schlüssel k. Die Generierung dieses Schlüssels wird mit $k \xleftarrow{\$} K$ bezeichnet und erfolgt durch mehrere interne Münzwürfe. Die Menge aller Schlüssel, die $\mathcal{K}$ liefern kann, wird mit $\mathit{Keys}(\mathcal{SE})$ bezeichnet.

Der zufalls- oder zustandsgesteuerte Chiffrieralgorithmus $\mathcal{E}(x, k) = y$, kurz $\mathcal{E}_k(x) = y$, mit $x \in \{0,1\}^*$ ein Klartext, $k \in \mathit{Keys}(\mathcal{SE})$, liefert einen Geheimtext $y \in \{0,1\}^* \cup \{\perp\}$.

Der deterministische Dechiffrieralgorithmus $\mathcal{D}(y, k) = x$, kurz $\mathcal{D}_k(y) = x$, mit $k \in \mathit{Keys}(\mathcal{SE})$, $y \in \{0,1\}^*$ ein Geheimtext, liefert einen Klartext $x \in \{0,1\}^* \cup \{\perp\}$.

Die Laufzeiten von $\mathcal{E}$ und $\mathcal{D}$ sind dabei polynomiell in der Länge von x bzw. y beschränkt.

Die Dechiffrierbedingung $\mathcal{D}(\mathcal{E}(x, k), k) = x$ muss für alle $x \in \{0,1\}^*$ und alle $k \in \mathit{Keys}(\mathcal{SE})$ gelten. [14]

Ein asymmetrisches, algorithmisches Kryptosystem $\mathcal{AE}$ ist ebenso definiert als 3-Tupel $\mathcal{AE} = (\mathcal{K}, \mathcal{E}, \mathcal{D})$ bestehend aus drei Algorithmen:

Der zufallsgesteuerte Schlüsselgenerierungsalgorithmus $\mathcal{K}$ liefert, statt eines einzelnen Schlüssels, ein Schlüsselpaar (pk, sk), bestehend aus öffentlichem Schlüssel pk und

privatem Schlüssel sk. Die Generierung eines Schlüsselpaares wird mit $(pk, sk) \xleftarrow{\$} K$ bezeichnet.

Der zufallsgesteuerte Chiffrieralgorithmus $\mathcal{E}(x, pk) = y$ mit pk ein öffentlicher Schlüssel aus einem Schlüsselpaar $(pk, sk) \in \mathit{Keys}(\mathcal{AE})$, x ein Klartext, liefert einen Geheimtext y.

Der deterministische Dechiffrieralgorithmus $\mathcal{D}(y, sk) = x$ mit sk ein privater Schlüssel aus einem Schlüsselpaar $(pk, sk) \in \mathit{Keys}(\mathcal{AE})$, y ein Geheimtext, liefert einen Klartext x.

Definition 4.1 (Algorithmische Kryptosysteme)**.** *Ein symmetrisches, algorithmisches Kryptosystem $\mathcal{SE} = (\mathcal{K}, \mathcal{E}, \mathcal{D})$ bzw. ein asymmetrisches, algorithmisches Kryptosystem $\mathcal{AE} = (\mathcal{K}, \mathcal{E}, \mathcal{D})$ besteht aus einem Schlüsselgenerierungsalgorithmus $\mathcal{K}$, einem zufalls- oder zustandsgesteuerten Chiffrieralgorithmus $\mathcal{E}$, sowie einem deterministischen Dechiffrieralgorithmus $\mathcal{D}$.*

4.3 Ununterscheidbarkeit bei symmetrischen Kryptosystemen

Wie im Laufe dieser Arbeit bereits beschrieben, darf ein Angreifer aus einem Geheimtext keine Informationen über den Klartext gewinnen können. Daraus folgt, dass ein Angreifer auch keine zwei Geheimtexte den zugehörigen Klartexten zuordnen können darf, denn für diese Zuordnung müsste er zumindest geringe Informationen über den Klartext aus dem Geheimtext gewinnen können. Wenn es für den Angreifer bei der Zuordnung also nicht möglich ist, signifikant erfolgreicher zu sein als bei einer zufälligen Zuordnung der Geheimtexte, nennt man das verwendete Kryptosystem „sicher im Sinne der Ununterscheidbarkeit“.

Das Sicherheitsziel der Ununterscheidbarkeit besagt also, dass ein Angreifer nicht in der Lage sein darf, Geheimtextpaare anhand der Klartexte unterscheiden zu können [6].

Die in diesem Kapitel folgenden Definitionen und Beschreibungen richten sich nach den Arbeiten von Mihir Bellare und Phillip Rogaway [2, 4].

4.3.1 Indistinguishability Under A Chosen-Plaintext-Attack

Ein Kryptosystem soll auf „left-or-right indistinguishability under a chosen-plaintext-attack“, vereinfacht auch nur „indistinguishability under a chosen-plaintext-attack“ oder kurz „IND-CPA“, geprüft werden.

Die Idee dazu ist, ein Spiel zwischen einem Angreifer und einem Herausforderer zu spielen. Der Angreifer wählt zwei Nachrichten M_0, M_1 gleicher Länge, somit nach Kapitel 2.1.4 ein Chosen-Plaintext-Angriff, von denen dann eine durch ein Verschlüsselungsorakel mit dem geheimen, dem Angreifer nicht bekannten Schlüssel verschlüsselt wird. Der produzierte Geheimtext wird anschließend vom Orakel an den Angreifer zurückgegeben. Die Aufgabe des Angreifers ist es schließlich zu bestimmen, welche der Nachrichten M_0, M_1 verschlüsselt wurde.

Das verwendete Kryptosystem ist sicher im Sinne der Ununterscheidbarkeit, wenn dem Angreifer diese Bestimmung schwer fällt.

Um die gewährleistete Sicherheit noch weiter zu erhöhen, soll der Angriff verstärkt werden. Dies wird erreicht, indem dem Angreifer mehr Möglichkeiten eingeräumt werden.

Statt eines einzelnen Nachrichtenpaars (M_0, M_1) kann der Angreifer nun eine Sequenz $(M_{0,1}, M_{1,1}), \ldots, (M_{0,q}, M_{1,q})$ bestehend aus q Nachrichtenpaaren wählen. Die Nachrichten in jedem Paar $(M_{0,i}, M_{1,i})$, $1 \leq i \leq q$, müssen jeweils gleicher Länge sein, sprich $|M_{0,i}| = |M_{1,i}|$ für $1 \leq i \leq q$.

Das Orakel liefert dem Angreifer dementsprechend auch statt eines einzelnen Geheimtextes eine Sequenz von Geheimtexten $C_1, \ldots C_q$ zurück. Dabei gilt: Es wird entweder immer die erste oder immer die zweite Nachricht eines Paares verschlüsselt, sprich C_i ist entweder die Verschlüsselung von $M_{0,i}$ für alle $1 \leq i \leq q$ oder die Verschlüsselung von $M_{1,i}$ für alle $1 \leq i \leq q$. Das Orakel verwendet für die Berechnung aller Geheimtexte denselben Schlüssel, die Ergebnisse von Münzwürfen bei zufallsgesteuerten oder Zustandswerte bei zustandsgesteuerten Verschlüsselungen können jedoch verschieden sein.

Um den Angriff noch weiter zu verstärken wird dem Angreifer ein adaptiver Chosen-Plaintext-Angriff ermöglicht. Statt alle q Nachrichtenpaare der Sequenz gleichzeitig zu wählen, darf der Angreifer diese nacheinander wählen und Orakelantworten abwarten: Erst nachdem der Angreifer das Nachrichtenpaar $(M_{0,i}, M_{1,i})$ gewählt und das Orakel C_i zurückgegeben hat, wählt der Angreifer das nächste Nachrichtenpaar $(M_{0,i+1}, M_{1,i+1})$ und übergibt dieses an das Orakel.

Wie in Kapitel 2.1.4 bereits erwähnt, ist der adaptive Chosen-Plaintext-Angriff auch in diesem Fall die gängigere Variante, weswegen im Weiteren dieser gemeint sein soll, wenn von einem Chosen-Plaintext-Angriff gesprochen wird.

4.3.2 Left-Or-Right Verschlüsselungsorakel

Das im vorigen Kapitel bereits erwähnte Verschlüsselungsorakel soll nun genauer betrachtet werden.

Ein Left-Or-Right-Verschlüsselungsorakel, kurz LOR-Orakel, erhält als Eingabe ein Nachrichtenpaar (M_0, M_1), sowie das Bit b. Die Funktionsweise des Verschlüsselungsorakels kann folgendermaßen beschrieben werden:

Wenn $b = 0$ dann berechne $C \leftarrow \mathcal{E}_K(M_0)$, wenn $b = 1$ dann berechne $C \leftarrow \mathcal{E}_K(M_1)$. Gebe anschließend C zurück.

Definition 4.2 (LOR-Verschlüsselungsorakel)**.** *Sei $\mathcal{SE} = (\mathcal{K}, \mathcal{E}, \mathcal{D})$ ein Kryptosystem, $K \leftarrow \mathcal{K}$ ein Schlüssel. Ein Left-Or-Right-Verschlüsselungsorakel ist definiert als*

$$\mathcal{E}_K(LR(\cdot, \cdot, b)) \text{ mit } b \in \{0, 1\}$$

Ein LOR-Verschlüsselungsorakel ist in Algorithmus 2 [2, S.103] gegeben.

Algorithmus 2 LOR-Verschlüsselungsorakel

function ORACLE($LR(M_0, M_1, b)$) ▷ $b \in \{0,1\}$ und $M_0, M_1 \in \{0,1\}^*$
  if $|M_0| \neq |M_1|$ **then return** $\perp$
  end if

  $C \xleftarrow{\$} \mathcal{E}_K(M_b)$
  return C
end function

4.3.3 IND-CPA

Das Spiel, mit dem ein Kryptosystem auf Sicherheit im Sinne der Ununterscheidbarkeit gegen einen Chosen-Plaintext-Angriff geprüft wird und zuvor in Kapitel 4.3.1 beschrieben worden ist, soll nun formalisiert werden.

Gegeben sei ein symmetrisches Kryptosystem $\mathcal{SE} = (\mathcal{K}, \mathcal{E}, \mathcal{D})$, sowie ein Angreifer $\mathcal{A}$, welcher ein Programm mit Zugriff auf ein LOR-Verschlüsselungsorakel sein soll. Das Verschlüsselungsorakel verschlüsselt eine Nachricht des eingegebenen Nachrichtenpaars entsprechend der Beschreibung aus dem letzten Kapitel abhängig von Bit b.
Es werden zwei Welten betrachtet:

- Welt 0: $b = 0$, das Orakel, das der Angreifer bekommt ist $\mathcal{E}_K(LR(\cdot, \cdot, 0))$. Das Orakel liefert dem Angreifer stets die Verschlüsselung von M_0, sofern $|M_0| = |M_1|$.
- Welt 1: $b = 1$, das Orakel, das der Angreifer bekommt ist $\mathcal{E}_K(LR(\cdot, \cdot, 1))$. Das Orakel liefert dem Angreifer stets die Verschlüsselung von M_1, sofern $|M_0| = |M_1|$.

Die Aufgabe des Angreifers im Spiel ist es, zu ermitteln, in welcher Welt er sich befindet. Der Wert von b, und damit auch die Welt, wird nur ein einziges Mal zu Beginn festgelegt.

Die Prüfung des Kryptosystems $\mathcal{SE} = (\mathcal{K}, \mathcal{E}, \mathcal{D})$ auf Sicherheit im Sinne der Ununterscheidbarkeit wird mittels zweier Spiele formuliert: $\mathbf{Left}_{\mathcal{SE}}^{\text{ind-cpa}}$ und $\mathbf{Right}_{\mathcal{SE}}^{\text{ind-cpa}}$.

Spiel $\mathbf{Left}_{\mathcal{SE}}^{\text{ind-cpa}}$

procedure Initialize
$K \xleftarrow{\$} \mathcal{K}$
procedure LR(M_0, M_1)
return $C \xleftarrow{\$} \mathcal{E}_K(M_0)$
procedure Finalize(b')
return b'

Spiel $\mathbf{Right}_{\mathcal{SE}}^{\text{ind-cpa}}$

procedure Initialize
$K \xleftarrow{\$} \mathcal{K}$
procedure LR(M_0, M_1)
return $C \xleftarrow{\$} \mathcal{E}_K(M_1)$
procedure Finalize(b')
return b'

Jedes dieser Spiele spiegelt eine Welt wieder. Das Spiel $\mathbf{Left}_{\mathcal{SE}}^{\text{ind-cpa}}$ die Welt 0, das Spiel $\mathbf{Right}_{\mathcal{SE}}^{\text{ind-cpa}}$ die Welt 1. Entsprechend der durch Bit b festgelegten Welt wird dem Angreifer $\mathcal{A}$ das jeweilige Spiel gegeben.

In beiden Spielen wird in der *Initialize*-Prozedur, die ein einziges Mal zu Spielbeginn ausgeführt wird, ein Schlüssel K zufällig ausgewählt. Mit der *LR*-Prozedur kann der Angreifer dann Anfragen an das Verschlüsselungsorakel stellen und das Nachrichtenpaar (M_0, M_1) an dieses übergeben. Entsprechend der repräsentierten Welt wird dann in beiden Spielen eine Nachricht mit Hilfe des in der Initialize-Prozedur gewählten Schlüssels K verschlüsselt und der errechnete Geheimtext C an den Angreifer zurückgegeben. Die LR-Prozedur kann vom Angreifer bei Bedarf mehrmals ausgeführt werden.

Mit b' gibt der Angreifer schließlich seine Vermutung, welches Spiel er spielt, also in welcher Welt er sich befindet, an die *Finalize*-Prozedur ab. Diese Vermutung des Angreifers ist gleichzeitig auch das Ergebnis des Spiels.

Definition 4.3 (IND-CPA)**.** *Sei $\mathcal{SE} = (\mathcal{K}, \mathcal{E}, \mathcal{D})$ ein symmetrisches Kryptosystem und Angreifer $\mathcal{A}$ ein Algorithmus. Basierend auf Bit $b \in \{0, 1\}$ wird $\mathcal{A}$ das Spiel $\boldsymbol{Left}_{\mathcal{SE}}^{ind\text{-}cpa}$ ($b = 0$) oder das Spiel $\boldsymbol{Right}_{\mathcal{SE}}^{ind\text{-}cpa}$ ($b = 1$) zugeteilt.*

Im Laufe des Spiels kann $\mathcal{A}$ ein oder mehrere Klartextpaare mit Klartexten gleicher Länge an ein Verschlüsselungsorakel $\mathcal{E}_K(LR(\cdot, \cdot, b))$ senden, um einen Geheimtext zu erhalten. Zum Ende des Spiels muss $\mathcal{A}$ eine Vermutung b' über Bit b abgeben.

IND-CPA-Vorteil

Wie im vorigen Kapitel beschrieben, ist es die Aufgabe des Angreifers $\mathcal{A}$ zu bestimmen, in welcher Welt er sich befindet. Der Erfolg, den $\mathcal{A}$ dabei hat, wird als IND-CPA-Vorteil (engl. „IND-CPA advantage“) bezeichnet.

Definition 4.4 (IND-CPA-Vorteil)**.** *Sei $\mathcal{SE} = (\mathcal{K}, \mathcal{E}, \mathcal{D})$ ein symmetrisches Kryptosystem und Angreifer $\mathcal{A}$ ein Algorithmus. Der IND-CPA-Vorteil $\mathsf{Adv}_{\mathcal{SE}}^{\text{ind-cpa}}(\mathcal{A})$ des Angreifers $\mathcal{A}$ gegen $\mathcal{SE}$ ist definiert als*

$$\mathsf{Adv}_{\mathcal{SE}}^{\text{ind-cpa}}(\mathcal{A}) = Pr[\boldsymbol{Right}_{\mathcal{SE}}^{ind\text{-}cpa}(\mathcal{A}) = 1] - Pr[\boldsymbol{Left}_{\mathcal{SE}}^{ind\text{-}cpa}(\mathcal{A}) = 1]$$

Dabei ist $Pr[\mathbf{Left}_{\mathcal{SE}}^{ind\text{-}cpa}(\mathcal{A}) = 1]$ bzw. $Pr[\mathbf{Right}_{\mathcal{SE}}^{ind\text{-}cpa}(\mathcal{A}) = 1]$ die Wahrscheinlichkeit, mit der der Angreifer $\mathcal{A}$ in Spiel $\mathbf{Left}_{\mathcal{SE}}^{\text{ind-cpa}}$ bzw. Spiel $\mathbf{Right}_{\mathcal{SE}}^{\text{ind-cpa}}$ die Vermutung $b' = 1$ abgibt.

Der IND-CPA-Vorteil gibt an, wie erfolgreich der Angreifer $\mathcal{A}$ insgesamt bei der Bestimmung der Welt ist. Je näher der Vorteil bei 0 liegt, desto schlechter ist der Angreifer dabei. Bei einem großen Vorteil, also einem Wert gegen 1, kann der Angreifer scheinbar bestimmen, in welcher Welt er sich befindet. Das geprüfte Kryptosystem ist somit nicht sicher im Sinne der Ununterscheidbarkeit.

Damit ein Kryptosystem diese Sicherheit erlangt, muss der Vorteil stets klein sein, unabhängig von der Strategie des Angreifers.

Der Angreifer ist bei der Betrachtung der Sicherheit im Sinne der Ununterscheidbarkeit jedoch bezüglich seiner Ressourcen auf realistische Möglichkeiten beschränkt. Diese Beschränkungen umfassen dabei die Laufzeit und die Anzahl der Orakelanfragen. Bei der Laufzeit wird zudem stets die Worst-Case-Laufzeit angenommen, unabhängig von Ergebnissen von Münzwürfen oder Orakelantworten.

4.3.4 Alternative Definition für IND-CPA

Neben der im vorigen Kapitel vorgestellten Definition mit zwei Spielen, von denen dem Angreifer je nach gewählter Welt eins gegeben wird, existiert eine weitere Definition für IND-CPA. In dieser alternativen Definition wird dem Angreifer statt je nach Welt eines der Spiele $\mathbf{Left}_{\mathcal{SE}}^{\text{ind-cpa}}$ und $\mathbf{Right}_{\mathcal{SE}}^{\text{ind-cpa}}$ in beiden Welten das Spiel $\mathbf{Guess}_{\mathcal{SE}}^{\text{ind-cpa}}$ gegeben.

Statt zweier Spiele wird nur ein einziges Spiel benötigt. Die Definition dieses Spiels $\mathbf{Guess}_{\mathcal{SE}}^{\text{ind-cpa}}$ ist im Folgenden gegeben.

Spiel $\mathbf{Guess}_{\mathcal{SE}}^{\text{ind-cpa}}$

procedure Initialize $b \xleftarrow{\$} \{0,1\}$; $K \xleftarrow{\$} \mathcal{K}$
procedure LR(M_0, M_1) return $C \xleftarrow{\$} \mathcal{E}_K(M_b)$
procedure Finalize(b') return $(b == b')$

Zu Beginn des Spiels wird, wie vor Beginn der Spiele $\mathbf{Left}_{\mathcal{SE}}^{\text{ind-cpa}}$ und $\mathbf{Right}_{\mathcal{SE}}^{\text{ind-cpa}}$, das Bit b zufällig festgelegt, woraufhin dann der Angreifer in der Welt b spielt. Zum Schluss des Spiels gibt der Angreifer eine Vermutung b' für b ab und gewinnt das Spiel, wenn $b' = b$, seine Vermutung also korrekt, ist. Das Spiel liefert dann die Ausgabe 1, bei inkorrekter Vermutung die Ausgabe 0.

Der Vorteil des Angreifers ist in dieser Variante definiert als:

$$\mathsf{Adv}_{\mathcal{SE}}^{\text{ind-cpa}}(\mathcal{A}) = 2 \cdot Pr[\mathbf{Guess}_{\mathcal{SE}}^{\text{ind-cpa}}(\mathcal{A}) = 1] - 1$$

Dabei ist $Pr[\mathbf{Guess}_{\mathcal{SE}}^{\text{ind-cpa}}(\mathcal{A}) = 1]$ die Wahrscheinlichkeit, mit der der Angreifer den Wert von b korrekt bestimmt.

Durch zufälliges Raten kann der Angreifer das Spiel auch ohne weitere Informationen bereits mit einer Wahrscheinlichkeit von $Pr[\mathbf{Guess}_{\mathcal{SE}}^{\text{ind-cpa}}(\mathcal{A}) = 1] = \frac{1}{2}$ gewinnen. In der Definition des Vorteils wird die Wahrscheinlichkeit des Angreifers daher mit $2 \cdot Pr[\mathbf{Guess}_{\mathcal{SE}}^{\text{ind-cpa}}(\mathcal{A}) = 1] - 1$ skaliert. Der Vorteil beschreibt dadurch auch in dieser Variante den Erfolg des Angreifers relativ zum Erfolg beim Raten und ist daher mit

Ergebnissen, die mittels der Spiele $\mathbf{Left}_{\mathcal{SE}}^{\text{ind-cpa}}$ und $\mathbf{Right}_{\mathcal{SE}}^{\text{ind-cpa}}$ aufgestellt worden sind, vergleichbar.

Wie in der zuvor vorgestellten Definition, sind die Ressourcen des Angreifers auch in dieser Variante bezüglich Laufzeit und Anfragen an das Verschlüsselungsorakel auf realistische Möglichkeiten beschränkt.

Definition 4.5 (alt. IND-CPA)**.** *Sei $\mathcal{SE} = (\mathcal{K}, \mathcal{E}, \mathcal{D})$ ein symmetrisches Kryptosystem und Angreifer $\mathcal{A}$ ein Algorithmus. Zu Beginn des Spiels $\boldsymbol{Guess}_{\mathcal{SE}}^{ind\text{-}cpa}$ bestimmt der Herausforderer das Bit $b \xleftarrow{\$} \{0,1\}$, sowie den Schlüssel $K \xleftarrow{\$} \mathcal{K}$. Der Angreifer $\mathcal{A}$ kann an das Verschlüsselungsorakel $\mathcal{E}_K(LR(\cdot,\cdot,b))$ Anfragen in der Form von Paaren gleichlanger Klartexte senden und muss zum Ende eine Vermutung b' über das Bit b ausgeben.*

Der Vorteil von $\mathcal{A}$ ist definiert als

$$\mathsf{Adv}_{\mathcal{SE}}^{\text{ind-cpa}}(\mathcal{A}) = 2 \cdot Pr[\boldsymbol{Guess}_{\mathcal{SE}}^{ind\text{-}cpa}(\mathcal{A}) = 1] - 1$$

4.3.5 IND-CCA

Um eine noch höhere Sicherheit, als IND-CPA bieten kann, zu erlangen, wird der Angreifer verstärkt. Daher wird ihm statt eines Chosen-Plaintext-Angriffs der in Kapitel 2.1.4 beschriebene, stärkere Chosen-Ciphertext-Angriff ermöglicht.

Dazu benötigt der Angreifer neben dem Verschlüsselungsorakel zusätzlich ein Entschlüsselungsorakel, das bei Eingabe eines Geheimtextes den zugehörigen Klartext ausgibt.

Dabei wird dem Angreifer bezüglich der Nutzung des Entschlüsselungsorakels eine Beschränkung auferlegt: Würde er das Nachrichtenpaar (M_0, M_1) an das Verschlüsselungsorakel senden und den dann erhaltenen Geheimtext C wiederum mit dem Entschlüsselungsorakel entschlüsseln können, würde er entweder M_0 oder M_1 zurückerhalten. Da ihm diese Nachrichten bekannt sind, da er sie selbst gewählt hat, wäre eine Bestimmung der Welt für den Angreifer somit unabhängig vom eingesetzten Kryptosystem problemlos möglich.

Vom Verschlüsselungsorakel erhaltene Geheimtexte dürfen daher nicht mit dem Entschlüsselungsorakel wieder entschlüsselt werden. Eine solche Anfrage mit einem vom Verschlüsselungsorakel erhaltenen Geheimtext wird als *unzulässige* (engl. „illegitimate“) Anfrage an das Entschlüsselungsorakel bezeichnet.

Die Beschränkungen des Angreifers bezüglich Ressourcen aus der IND-CPA Definition bleiben bestehen. Zusätzlich sind bei IND-CCA auch die Anfragen an das Entschlüsselungsorakel beschränkt.

Wie bei IND-CPA werden auch bei IND-CCA zwei Welten betrachtet:

- Welt 0: $b = 0$, der Angreifer bekommt das Verschlüsselungsorakel $\mathcal{E}_K(LR(\cdot,\cdot,0))$, sowie das Entschlüsselungsorakel $\mathcal{D}(\cdot)$.
- Welt 1: $b = 1$, der Angreifer bekommt das Verschlüsselungsorakel $\mathcal{E}_K(LR\cdot,\cdot,1))$, sowie das Entschlüsselungsorakel $\mathcal{D}(\cdot)$.

Es bleibt das Ziel des Angreifers herauszufinden, in welcher Welt er sich befindet.

Bei einem Chosen-Ciphertext-Angriff sendet der Angreifer in der Regel zunächst ein Nachrichtenpaar (M_0, M_1) an das Verschlüsselungsorakel $\mathcal{E}_K(LR(\cdot,\cdot,b))$ um einen Geheimtext C zu bekommen. Diesen Geheimtext C modifiziert er dann zu einem Geheimtext C', welchen er an das Entschlüsselungsorakel $\mathcal{D}(\cdot)$ senden kann. Die Modifikation von C zu C' versucht er dabei so zu gestalten, dass er aus dem erhaltenen Klartext des modifizierten Geheimtextes Informationen gewinnen kann, die ihm helfen, die Welt zu bestimmen.

Wie bei IND-CPA werden zwei Spiele betrachtet, $\mathbf{Left}_{\mathcal{SE}}^{\text{ind-cca}}$ und $\mathbf{Right}_{\mathcal{SE}}^{\text{ind-cca}}$, von denen jedes Spiel eine Welt abbildet. Entsprechend Bit b wird dem Angreifer eines der Spiele gegeben.

Spiel $\mathbf{Left}_{\mathcal{SE}}^{\text{ind-cca}}$

procedure Initialize $K \xleftarrow{\$} \mathcal{K}$
procedure LR(M_0, M_1) return $C \xleftarrow{\$} \mathcal{E}_K(M_0)$
procedure D(C) return $M \leftarrow \mathcal{D}_K(C)$
procedure Finalize(b') if $\mathcal{A}$ queried an illegitimate ciphertext return 0 else return b'

Spiel $\mathbf{Right}_{\mathcal{SE}}^{\text{ind-cca}}$

procedure Initialize $K \xleftarrow{\$} \mathcal{K}$
procedure LR(M_0, M_1) return $C \xleftarrow{\$} \mathcal{E}_K(M_1)$
procedure D(C) return $M \leftarrow \mathcal{D}_K(C)$
procedure Finalize(b') if $\mathcal{A}$ queried an illegitimate ciphertext return 0 else return b'

Bei der Bezeichnung von IND-CCA wird, im Gegensatz zu IND-CPA, zwischen adaptivem und nicht-adaptivem Angriff unterschieden. Bei nicht-adaptivem Angriff mit Geheimtextwahl wird die Bezeichnung IND-CCA1, bei adaptivem Angriff mit Geheimtextwahl die Bezeichnung IND-CCA2 verwendet. Beim nicht-adaptiven Angriff hat der Angreifer nur während er die Sequenz der Klartextpaare, die er an das Verschlüsselungsorakel übergeben wird, wählt Zugriff auf das Entschlüsselungsorakel. Nach dem Erhalt der Geheimtextsequenz vom Verschlüsselungsorakel, also während der Festlegung des Bits b', hat er keinen Zugriff mehr.

Die in diesem Abschnitt beschriebene Variante ist somit IND-CCA2. Da der adaptive Angriff stärker ist, als der nicht-adaptive Angriff, ist IND-CCA2 auch ein stärkerer Sicherheitsbegriff als IND-CCA1 [5]. IND-CCA2 ist die gebräuchlichere IND-CCA-Definition und ist daher gemeint, wenn in dieser Arbeit von IND-CCA gesprochen wird.

Definition 4.6 (IND-CCA)**.** *Sei $\mathcal{SE} = (\mathcal{K}, \mathcal{E}, \mathcal{D})$ ein symmetrisches Kryptosystem und Angreifer $\mathcal{A}$ ein Algorithmus. Basierend auf einem Bit $b \in \{0, 1\}$ wird $\mathcal{A}$ das Spiel $\boldsymbol{Left}^{ind\text{-}cca}_{\mathcal{SE}}$ $(b = 0)$ oder das Spiel $\boldsymbol{Right}^{ind\text{-}cca}_{\mathcal{SE}}$ $(b = 1)$ zugeteilt.*

Der Angreifer $\mathcal{A}$ kann Anfragen in der Form von Paaren aus Klartexten gleicher Länge an ein Verschlüsselungsorakel $\mathcal{E}_K(LR(\cdot, \cdot, b))$ und Anfragen in der Form eines Geheimtextes an ein Entschlüsselungsorakel $\mathcal{D}(\cdot)$ stellen. Zum Ende muss $\mathcal{A}$ eine Vermutung über das Bit b ausgeben.

Eine IND-CCA-Sicherheit impliziert stets eine IND-CPA-Sicherheit. Ist ein Kryptosystem sicher im Sinne der Ununterscheidbarkeit gegen einen Chosen-Ciphertext-Angriff, ist es auch sicher im Sinne der Ununterscheidbarkeit gegen einen Chosen-Plaintext-Angriff. Ebenso impliziert eine IND-CCA2-Sicherheit stets eine IND-CCA1-Sicherheit.

IND-CCA-Vorteil

Wie bei IND-CPA wird auch im IND-CCA-Szenario ein Vorteil des Angreifers, genannt IND-CCA-Vorteil $\mathsf{Adv}^{\text{ind-cca}}_{\mathcal{SE}}(\mathcal{A})$, bestimmt.

Definition 4.7 (IND-CCA-Vorteil)**.** *Sei $\mathcal{SE} = (\mathcal{K}, \mathcal{E}, \mathcal{D})$ ein symmetrisches Kryptosystem und Angreifer $\mathcal{A}$ ein Algorithmus. Der IND-CCA-Vorteil des Angreifers $\mathcal{A}$ ist definiert als*

$$\mathsf{Adv}^{\text{ind-cca}}_{\mathcal{SE}}(\mathcal{A}) = Pr[\boldsymbol{Right}^{ind\text{-}cca}_{\mathcal{SE}}(\mathcal{A}) = 1] - Pr[\boldsymbol{Left}^{ind\text{-}cca}_{\mathcal{SE}}(\mathcal{A}) = 1]$$

Ein Kryptosystem ist sicher im Sinne der Ununterscheidbarkeit gegen einen Chosen-Ciphertext-Angriff, wenn ein Angreifer bei der Bestimmung der Welt maximal einen vernachlässigbar kleinen Vorteil erlangen kann.

4.4 Ununterscheidbarkeit bei asymmetrischen Kryptosystemen

Die Beschreibungen in diesem Abschnitt folgen der Arbeit von Mihir Bellare und Phillip Rogaway [2, 4].

4.4.1 IND-CPA

Die Definition von IND-CPA für asymmetrische Kryptosysteme ist im Wesentlichen identisch zu der Definition für symmetrische Kryptosysteme.

Der Herausforderer wählt zu Beginn des Spiels zufällig das Schlüsselpaar (K, K'), bestehend aus privatem Schlüssel K und öffentlichem Schlüssel K'. Letzterer wird auch an den Angreifer übergeben. Außerdem legt der Herausforderer den Wert für das Bit $b \in \{0, 1\}$ fest.

Der Angreifer kann dann adaptiv die Nachrichtenpaare $(M_{0,1}, M_{1,1}), \ldots, (M_{0,q}, M_{1,q})$ wählen und diese jeweils an das Verschlüsselungsorakel senden. Das Orakel berechnet

dann bei Eingabe eines Nachrichtenpaars $(M_{0,i}, M_{1,i})$ entsprechend dem Bit b den Geheimtext C_i mit $C_i \leftarrow \mathcal{E}_{K'}(M_{b,i})$ für $1 \leq i \leq q$ und gibt diesen an den Angreifer zurück.

Wie im symmetrischen Szenario werden dazu zwei Welten betrachtet:

- Welt 0: $b = 0$, der Angreifer bekommt das Verschlüsselungsorakel $\mathcal{E}_{K'}(LR(\cdot, \cdot, 0))$, also stets die Verschlüsselung von Nachricht M_0.
- Welt 1: $b = 1$, der Angreifer bekommt das Verschlüsselungsorakel $\mathcal{E}_{K'}(LR(\cdot, \cdot, 1))$, also stets die Verschlüsselung von Nachricht M_1.

Ziel des Angreifers ist es, herauszufinden in welcher Welt er sich befindet, sprich welche Nachrichten jeweils verschlüsselt worden sind.

Ein asymmetrisches Kryptosystem gilt als sicher im Sinne der Ununterscheidbarkeit gegen einen Chosen-Plaintext-Angriff, wenn die Erfolgsrate des Angreifers bei dieser Aufgabe maximal vernachlässigbar größer als der erwartete Erfolg bei einem Versuch durch zufälliges Raten der Welt ist.

Wie im symmetrischen Szenario ist für jede Welt ein Spiel definiert, an dessen Ende der Angreifer jeweils seine Vermutung über die Welt in der Variable b' ausgibt:

Spiel $\mathbf{Left}^{\text{ind-cpa}}_{\mathcal{AE}}$

procedure Initialize
$(K, K') \xleftarrow{\$} \mathcal{K}$
return K'

procedure LR(M_0, M_1)
return $C \xleftarrow{\$} \mathcal{E}_{K'}(M_0)$

procedure Finalize(b')
return b'

Spiel $\mathbf{Right}^{\text{ind-cpa}}_{\mathcal{AE}}$

procedure Initialize
$(K, K') \xleftarrow{\$} \mathcal{K}$
return K'

procedure LR(M_0, M_1)
return $C \xleftarrow{\$} \mathcal{E}_{K'}(M_1)$

procedure Finalize(b')
return b'

Da der Angreifer im Besitz des öffentlichen Schlüssels ist, kann er, neben den Anfragen an das Orakel, mit Hilfe dieses Schlüssels eigenständig weitere Klartexte verschlüsseln.

Definition 4.8 (IND-CPA)**.** *Sei $\mathcal{AE} = (\mathcal{K}, \mathcal{E}, \mathcal{D})$ ein asymmetrisches Kryptosystem und Angreifer $\mathcal{A}$ ein Algorithmus. Basierend auf Bit $b \in \{0, 1\}$ wird $\mathcal{A}$ das Spiel $\boldsymbol{Left}^{ind\text{-}cpa}_{\mathcal{AE}}$ ($b = 0$) oder das Spiel $\boldsymbol{Right}^{ind\text{-}cpa}_{\mathcal{AE}}$ ($b = 1$) zugeteilt.*

Im Laufe des Spiels kann $\mathcal{A}$ ein oder mehrere Klartextpaare mit Klartexten gleicher Länge an ein Verschlüsselungsorakel $\mathcal{E}_{K'}(LR(\cdot, \cdot, b))$ senden und einen Geheimtext erhalten. Außerdem kann $\mathcal{A}$ mit dem öffentlichen Schlüssel K' selbständig einzelne Klartexte verschlüsseln. Zum Ende muss $\mathcal{A}$ eine Vermutung über Bit b abgeben.

IND-CPA-Vorteil

Die Definition des IND-CPA-Vorteils $\mathsf{Adv}^{\text{ind-cpa}}_{\mathcal{AE}}(\mathcal{A})$ im asymmetrischen Szenario ist identisch zu der Definition im symmetrischen Szenario.

Definition 4.9 (IND-CPA-Vorteil)**.** *Sei $\mathcal{AE} = (\mathcal{K}, \mathcal{E}, \mathcal{D})$ ein asymmetrisches Kryptosystem und Angreifer $\mathcal{A}$ ein Algorithmus. Der IND-CPA-Vorteil $\mathsf{Adv}^{\text{ind-cpa}}_{\mathcal{AE}}(\mathcal{A})$ des Angreifers $\mathcal{A}$ ist definiert als*

$$\mathsf{Adv}^{\text{ind-cpa}}_{\mathcal{AE}}(\mathcal{A}) = Pr[\boldsymbol{Right}^{ind\text{-}cpa}_{\mathcal{AE}}(\mathcal{A}) = 1] - Pr[\boldsymbol{Left}^{ind\text{-}cpa}_{\mathcal{AE}}(\mathcal{A}) = 1]$$

Die Beschränkungen der Ressourcen des Angreifers auf realistische Möglichkeiten bezüglich Laufzeit und Anfragen an das Verschlüsselungsorakel, wie sie im entsprechenden Kapitel für symmetrische Kryptosysteme beschrieben worden sind, gelten auch im asymmetrischen Spiel.

4.4.2 Alternative Definition für IND-CPA

Auch für asymmetrische Kryptosysteme soll die alternative Definition für IND-CPA gegeben werden. Statt der Spiele $\mathbf{Left}^{\text{ind-cpa}}_{\mathcal{AE}}$ und $\mathbf{Right}^{\text{ind-cpa}}_{\mathcal{AE}}$ bekommt der Angreifer in beiden Welten das Spiel $\mathbf{Guess}^{\text{ind-cpa}}_{\mathcal{AE}}$ gegeben.

Die Definition des Spiels $\mathbf{Guess}^{\text{ind-cpa}}_{\mathcal{AE}}$ ist im Wesentlichen identisch zu der Definition des Spiels für symmetrische Kryptosysteme und ist im Folgenden gegeben.

Spiel $\mathbf{Guess}^{\text{ind-cpa}}_{\mathcal{AE}}$

procedure Initialize $b \xleftarrow{\$} \{0,1\}$; $(K, K') \xleftarrow{\$} \mathcal{K}$ return K'
procedure LR(M_0, M_1) return $C \xleftarrow{\$} \mathcal{E}_{K'}(M_b)$
procedure Finalize(b') return $(b == b')$

Zu Beginn des Spiels wird das Bit b zufällig festgelegt, woraufhin dann der Angreifer in der Welt b spielt. Zum Schluss des Spiels gibt der Angreifer eine Vermutung b' für dieses Bit b ab. Er gewinnt das Spiel, wenn $b' = b$, seine Vermutung also korrekt, ist. Das Spiel liefert dann die Ausgabe 1, bei inkorrekter Vermutung die Ausgabe 0.

Der Vorteil des Angreifers ist in dieser Variante definiert als:

$$\mathsf{Adv}^{\text{ind-cpa}}_{\mathcal{AE}}(\mathcal{A}) = 2 \cdot Pr[\mathbf{Guess}^{\text{ind-cpa}}_{\mathcal{AE}}(\mathcal{A}) = 1] - 1$$

Dabei ist $Pr[\mathbf{Guess}_{\mathcal{AE}}^{\text{ind-cpa}}(\mathcal{A}) = 1]$ die Wahrscheinlichkeit, mit der der Angreifer den Wert von b korrekt bestimmt.

Auch in dieser Variante sind die Ressourcen des Angreifers bezüglich Laufzeit und Anfragen an das Verschlüsselungsorakel auf realistische Möglichkeiten beschränkt.

Definition 4.10 (alt. IND-CPA)**.** *Sei $\mathcal{AE} = (\mathcal{K}, \mathcal{E}, \mathcal{D})$ ein asymmetrisches Kryptosystem und Angreifer $\mathcal{A}$ ein Algorithmus. Zu Beginn des Spiels $\boldsymbol{Guess}_{\mathcal{AE}}^{ind\text{-}cpa}$ bestimmt der Herausforderer das Bit $b \xleftarrow{\$} \{0,1\}$, sowie das Schlüsselpaar $(K, K') \xleftarrow{\$} \mathcal{K}$. Der Angreifer $\mathcal{A}$ kann an das Verschlüsselungsorakel $\mathcal{E}_{K'}(LR(\cdot,\cdot,b))$ Anfragen in der Form von Paaren gleichlanger Klartexte senden und muss zum Ende eine Vermutung b' über das Bit b ausgeben.*

Der Vorteil von $\mathcal{A}$ ist definiert als

$$\mathsf{Adv}_{\mathcal{AE}}^{\text{ind-cpa}}(\mathcal{A}) = 2 \cdot Pr[\boldsymbol{Guess}_{\mathcal{AE}}^{ind\text{-}cpa}(\mathcal{A}) = 1] - 1$$

4.4.3 IND-CCA

Auch für asymmetrische Kryptosysteme soll eine stärkere Sicherheitsdefinition als IND-CPA aufgestellt werden. Statt eines Chosen-Plaintext-Angriffs wird ein Chosen-Ciphertext-Angriff betrachtet.

Der Herausforderer wählt zu Beginn des Spiels zufällig das Schlüsselpaar (K, K'), sowie das Bit $b \in \{0,1\}$. Der Angreifer erhält Zugriff auf den öffentlichen Schlüssel K', ein Verschlüsselungsorakel $\mathcal{E}_{K'}(LR(\cdot,\cdot,b))$ und ein Entschlüsselungsorakel $\mathcal{D}_K(\cdot)$. Der private Schlüssel K bleibt dem Angreifer unbekannt.

Die zwei Welten sind definiert als:

- Welt 0: $b = 0$, der Angreifer bekommt das Verschlüsselungsorakel $\mathcal{E}_K(LR(\cdot,\cdot,0))$, sowie das Entschlüsselungsorakel $\mathcal{D}_K(\cdot)$.
- Welt 1: $b = 1$, der Angreifer bekommt das Verschlüsselungsorakel $\mathcal{E}_K(LR(\cdot,\cdot,1))$, sowie das Entschlüsselungsorakel $\mathcal{D}_K(\cdot)$.

Der Angreifer muss bestimmen, in welcher Welt er sich befindet. Die Beschränkungen auf realistische Ressourcen bezüglich Laufzeit und Anfragen an das Verschlüsselungs-, sowie das Entschlüsselungsorakel, bleiben für den Angreifer auch in diesem Szenario bestehen.

Wie bei der Definition von IND-CCA für symmetrische Kryptosysteme, wird auch bei asymmetrischen Kryptosystemen zwischen adaptivem und nicht-adaptivem Angriff unterschieden. Bei nicht-adaptivem Angriff mit Geheimtextwahl wird die Bezeichnung IND-CCA1, beim adaptiven Angriff mit Geheimtextwahl, wie in diesem Abschnitt beschrieben, die Bezeichnung IND-CCA2 verwendet.

Für jede der zuvor beschriebenen Welten wird ein Spiel definiert. Die Definitionen der Spiele $\mathbf{Left}_{\mathcal{AE}}^{\text{ind-cca}}$ und $\mathbf{Right}_{\mathcal{AE}}^{\text{ind-cca}}$ sind im Folgenden gegeben:

Spiel $\mathbf{Left}_{\mathcal{AE}}^{\text{ind-cca}}$

procedure Initialize $K \xleftarrow{\$} \mathcal{K}$ return K'
procedure LR(M_0, M_1) return $C \xleftarrow{\$} \mathcal{E}_{K'}(M_0)$
procedure D(C) return $M \leftarrow \mathcal{D}_K(C)$
procedure Finalize(b') if $\mathcal{A}$ queried an illegitimate ciphertext return 0 else return b'

Spiel $\mathbf{Right}_{\mathcal{AE}}^{\text{ind-cca}}$

procedure Initialize $K \xleftarrow{\$} \mathcal{K}$ return K'
procedure LR(M_0, M_1) return $C \xleftarrow{\$} \mathcal{E}_{K'}(M_1)$
procedure D(C) return $M \leftarrow \mathcal{D}_K(C)$
procedure Finalize(b') if $\mathcal{A}$ queried an illegitimate ciphertext return 0 else return b'

Die Beschränkung mit dem Entschlüsselungsorakel keine Geheimtexte, die vom Verschlüsselungsorakel zurückgegeben worden sind, entschlüsseln zu dürfen, bleibt bestehen.

Definition 4.11 (IND-CCA)**.** *Sei $\mathcal{AE} = (\mathcal{K}, \mathcal{E}, \mathcal{D})$ ein asymmetrisches Kryptosystem und Angreifer $\mathcal{A}$ ein Algorithmus. Basierend auf einem Bit $b \in \{0, 1\}$ wird $\mathcal{A}$ das Spiel $\boldsymbol{Left}_{\mathcal{AE}}^{ind\text{-}cca}$ $(b = 0)$ oder das Spiel $\boldsymbol{Right}_{\mathcal{AE}}^{ind\text{-}cca}$ $(b = 1)$ zugeteilt.*

Der Angreifer $\mathcal{A}$ kann Anfragen in der Form von Paaren aus Klartexten gleicher Länge an ein Verschlüsselungsorakel $\mathcal{E}_{K'}(LR(\cdot, \cdot, b))$ und Anfragen in der Form eines Geheimtextes an ein Entschlüsselungsorakel $\mathcal{D}_K(\cdot)$ stellen. Zum Ende muss $\mathcal{A}$ eine Vermutung über das Bit b ausgeben.

IND-CCA-Vorteil

Der IND-CCA-Vorteil $\mathsf{Adv}_{\mathcal{AE}}^{\text{ind-cca}}(\mathcal{A})$ wird auch bei asymmetrischen Kryptosystemen bestimmt.

Definition 4.12 (IND-CCA-Vorteil)**.** *Sei $\mathcal{AE} = (\mathcal{K}, \mathcal{E}, \mathcal{D})$ ein asymmetrisches Kryptosystem und Angreifer $\mathcal{A}$ ein Algorithmus. Der IND-CCA-Vorteil des Angreifers $\mathcal{A}$ ist definiert als*

$$\mathsf{Adv}_{\mathcal{AE}}^{\text{ind-cca}}(\mathcal{A}) = Pr[\boldsymbol{Right}_{\mathcal{AE}}^{ind\text{-}cca}(\mathcal{A}) = 1] - Pr[\boldsymbol{Left}_{\mathcal{AE}}^{ind\text{-}cca}(\mathcal{A}) = 1]$$

Ein asymmetrisches Kryptosystem gilt als sicher im Sinne der Ununterscheidbarkeit gegen einen Chosen-Ciphertext-Angriff wenn ein Angreifer bei einem solchen Angriff nur einen vernachlässigbar kleinen Vorteil erlangen kann.

4.5 Beispielanwendung auf symmetrische Kryptosysteme

4.5.1 Sicherheit von AES

Wichtige Eigenschaften der Sicherheit einer Blockchiffre sind Konfusion und Diffusion. Die Konfusion wird bestimmt durch die Abhängigkeit der statistischen Verteilung der Geheimtexte von der Verteilung der Klartexte. Sie ist groß, wenn diese Abhängigkeit von einem Angreifer nicht ausgenutzt werden kann. Die Diffusion ist abhängig davon, wie viele Bits des Geheimtextes die einzelnen Bits des Klartextes und des Schlüssels beeinflussen [8].

Durch den ShiftRow-, sowie den MixColumn-Schritt verfügt der AES-Algorithmus über hohe Konfusion und Diffusion. Darüber hinaus ist AES gegen alle modernen Angriffe auf Blockchiffren sicher [6]. Bei Verschlüsselungen von Klartexten, die länger als die Blocklänge von AES sind, muss aber ein Betriebsmodus, wie in Kapitel 2.2.3 beschrieben, eingesetzt werden. Je nach eingesetztem Betriebsmodus können dann auch beim eigentlich sicheren AES-Algorithmus Unsicherheiten entstehen.

4.5.2 Chosen-Plaintext-Angriff gegen den ECB-Mode

Es soll die Sicherheit im Sinne der Ununterscheidbarkeit des in Kapitel 2.2.3 beschriebenen ECB-Mode einer Blockchiffre untersucht werden. Die Untersuchungen folgen [2].

Es soll wie in Kapitel 4.3.3 beschrieben das Spiel $\mathbf{Left}^{\text{ind-cpa}}_{\text{AES-ECB}}$ bzw. $\mathbf{Right}^{\text{ind-cpa}}_{\text{AES-ECB}}$ gegen einen Angreifer $\mathcal{A}_{ECB}$ gespielt werden. Dazu bestimmt der Herausforderer zunächst den Schlüssel K und legt außerdem den Wert von Bit $b \in \{0, 1\}$ fest, um dem Angreifer das entsprechende Spiel zu geben.

Satz 4.1. *Gegeben sei das AES-Kryptosystem im ECB-Betriebsmodus. Es existiert ein Angreifer $\mathcal{A}_{ECB}$, der bei einem Chosen-Plaintext-Angriff gegen AES im ECB-Mode einen Vorteil $\mathsf{Adv}^{\text{ind-cpa}}_{AES\text{-}ECB}(\mathcal{A}_{ECB})$ von 1 erzielen kann.*

Beweis. Der Angreifer $\mathcal{A}_{ECB}$, der Zugriff auf das Verschlüsselungsorakel $\mathcal{E}_K(LR(\cdot, \cdot, b))$ besitzt und den geheimen Schlüssel K nicht kennt, soll im Folgenden konstruiert werden.

Die grundlegende Idee des Angreifers ist es auszunutzen, dass die AES-Verschlüsselung deterministisch und die Geheimtextblöcke im ECB-Mode unabhängig voneinander sind. Das bedeutet, dass aus identischen Klartextblöcken auch identische Geheimtextblöcke erzeugt werden. Ebenso ergeben unterschiedliche Klartextblöcke auch stets unterschiedliche Geheimtextblöcke.

Der Angreifer $\mathcal{A}_{ECB}$ wählt das Klartextpaar (M_0, M_1), das er später an das Verschlüsselungsorakel sendet, also so, dass ihm dieses Wissen hilft, sein Ziel, die Bestimmung der Welt, zu erreichen.

Wie in Kapitel 3.1 beschrieben ist beim AES-Algorithmus die Blockgröße $n = 128$ Bit. Die Nachrichten M_0, M_1 sollen mit einer Länge von zwei Blöcken, also $2 \cdot n = 256$ Bit,

konstruiert werden, wobei die Blöcke von M_0 unterschiedlich und die Blöcke von M_1 identisch sein sollen:

$$M_0 \leftarrow 0^n \| 1^n, M_1 \leftarrow 0^{2n}$$

Sendet der Angreifer $\mathcal{A}_{ECB}$ nun dieses Nachrichtenpaar (M_0, M_1) an das Verschlüsselungsorakel $\mathcal{E}_K(LR(\cdot, \cdot, b))$, so erhält er entsprechend der Definition einen Geheimtext C, welcher die Verschlüsselung von M_b ist, zurück. Dieser Geheimtext besteht, wie die Klartexte M_0, M_1, aus zwei Blöcken der Größe n:

$$\text{Welt 0: } C = \mathcal{E}_K(0^n) \| \mathcal{E}_K(1^n)$$

$$\text{Welt 1: } C = \mathcal{E}_K(0^n) \| \mathcal{E}_K(0^n)$$

Aus der obigen Überlegung folgt, dass der Angreifer $\mathcal{A}_{ECB}$ den Geheimtext C nun einem der Klartexte aus dem Klartextpaar (M_0, M_1) eindeutig zuordnen kann. $X(i)$ soll im Folgenden den i-ten Block eines Klar- oder Geheimtextes X bezeichnen.

Für M_0 muss gelten, dass $C(1) \neq C(2)$ ist, da $M_0(1) \neq M_0(2)$ ist.

Für M_1 muss gelten, dass $C(1) = C(2)$ ist, da $M_1(1) = M_1(2)$ ist.

Daraus folgt für den Angreifer $\mathcal{A}_{ECB}$, dass wenn $C(1) = C(2)$ gilt, er sich in Welt 1 befindet, sonst in Welt 0.

Ein Algorithmus für den Angreifer $\mathcal{A}_{ECB}$ ist in Algorithmus 3 gegeben.

Algorithmus 3 Angreifer $\mathcal{A}_{ECB}$

```
function A_ECB()
    M_0 = 0^n || 1^n
    M_1 = 0^{2n}

    C = LR(M_0, M_1)                    ▷ Anfrage an das Verschlüsselungsorakel

    if C(1) = C(2) then return 1
    else
        return 0
    end if
end function
```

Da der Angreifer $\mathcal{A}_{ECB}$ somit die Welt eindeutig bestimmen kann, ergibt sich

$$Pr[\mathbf{Left}^{\text{ind-cpa}}_{\text{AES-ECB}}(\mathcal{A}_{ECB}) = 1] = 0$$

$$Pr[\mathbf{Right}^{\text{ind-cpa}}_{\text{AES-ECB}}(\mathcal{A}_{ECB}) = 1] = 1$$

und somit für den Vorteil $\mathsf{Adv}^{\text{ind-cpa}}_{AES\text{-}ECB}(\mathcal{A}_{ECB})$ des Angreifers $\mathcal{A}_{ECB}$

$$\begin{aligned}\mathsf{Adv}^{\text{ind-cpa}}_{AES\text{-}ECB}(\mathcal{A}_{ECB}) &= Pr[\mathbf{Right}^{\text{ind-cpa}}_{\text{AES-ECB}}(\mathcal{A}_{ECB}) = 1] - Pr[\mathbf{Left}^{\text{ind-cpa}}_{\text{AES-ECB}}(\mathcal{A}_{ECB}) = 1] \\ &= 1 - 0 \\ &= 1\end{aligned}$$

□

Da es mit $\mathcal{A}_{ECB}$ also einen Angreifer gibt, der einen Vorteil von 1 erreichen kann, ist auch eine sichere Blockchiffre wie AES im ECB-Mode nicht sicher im Sinne der Ununterscheidbarkeit.

4.5.3 Chosen-Plaintext-Angriff gegen den CBC-C-Mode

Es soll die Sicherheit im Sinne der Ununterscheidbarkeit des in Kapitel 2.2.3 beschriebenen CBC-Mode mit Zähler-IV einer Blockchiffre untersucht werden. Die Untersuchungen folgen [2].

Es soll wie in Kapitel 4.3.3 beschrieben das Spiel $\mathbf{Left}^{\text{ind-cpa}}_{\text{AES-CBCC}}$ bzw. $\mathbf{Right}^{\text{ind-cpa}}_{\text{AES-CBCC}}$ gegen einen Angreifer $\mathcal{A}_{CBCC}$ gespielt werden. Dazu bestimmt der Herausforderer zunächst den Schlüssel K und legt außerdem den Wert von Bit $b \in \{0,1\}$ fest, um dem Angreifer das entsprechende Spiel zu geben.

Satz 4.2. *Gegeben sei der AES-Algorithmus im CBC-C-Betriebsmodus. Es existiert ein Angreifer $\mathcal{A}_{CBCC}$, der bei einem Chosen-Plaintext-Angriff gegen AES im CBC-C-Mode einen Vorteil $\mathsf{Adv}^{\text{ind-cpa}}_{AES\text{-}CBCC}(\mathcal{A}_{CBCC})$ von 1 erzielen kann.*

Beweis. Der Angreifer $\mathcal{A}_{CBCC}$, der Zugriff auf das Verschlüsselungsorakel $\mathcal{E}_K(LR(\cdot,\cdot,b))$ besitzt und den geheimen Schlüssel K nicht kennt, soll im Folgenden konstruiert werden.

Sei n die Blocklänge, welche im AES-Algorithmus 128 Bit groß ist. Der Angreifer wählt zwei Nachrichtenpaare $(M_{0,1}, M_{1,1})$ und $(M_{0,2}, M_{1,2})$ folgendermaßen:

$$M_{0,1} \leftarrow 0^n, M_{1,1} \leftarrow 0^n \text{ und } M_{0,2} \leftarrow 0^n, M_{1,2} \leftarrow 0^{n-1}1$$

Beide Nachrichtenpaare sendet der Angreifer daraufhin an das Verschlüsselungsorakel $\mathcal{E}_K(LR(\cdot,\cdot,b))$ und erhält dadurch die Geheimtexte C_1 und C_2 mit den jeweilig zugehörigen Initialisierungsvektoren zurück.

Für den Angreifer ist es entscheidend, wie sich die Geheimtexte abhängig von der Welt, in der er sich befindet, unterscheiden. In beiden Welten werden diese wie folgt vom Verschlüsselungsorakel berechnet:

$$\langle IV_1, C_1\rangle \leftarrow \mathcal{E}_K(LR(M_{0,1}, M_{1,1}, b))$$
$$\langle IV_2, C_2\rangle \leftarrow \mathcal{E}_K(LR(M_{0,2}, M_{1,2}, b))$$

Da der Initialisierungsvektor im CBC-C-Mode wie ein Zähler funktioniert, gilt in beiden Welten $IV_1 = 0$ und $IV_2 = 1$. Mit dieser Information kann der Angreifer nun entscheidende Informationen über die Welt erlangen.

Betrachtet wird zunächst Welt 0. Verschlüsselt wird in der ersten Anfrage an das Verschlüsselungsorakel die Verknüpfung von $M_{0,1}$ und IV_1, deren Werte dem Angreifer beide

bekannt sind. Die Verknüpfung ergibt $M_{0,1} \oplus IV_1 = 0^n \oplus 0 = 0$. Somit gilt $C_1 = \mathcal{E}_K(0)$. In der zweiten Anfrage wird äquivalent die Verknüpfung von $M_{0,2}$ mit IV_2 verschlüsselt. $M_{0,2} \oplus IV_2 = 0^n \oplus 1 = 1$, wodurch in Welt 0 für C_2 gilt, dass $C_2 = \mathcal{E}_K(1)$ ist. Auf Grund der deterministischen Verschlüsselung des AES-Algorithmus gilt damit in Welt 0 stets, dass $C_1 \neq C_2$ ist.

Für Welt 1 gilt, dass in der ersten Anfrage an das Verschlüsselungsorakel die Verknüpfung von $M_{1,1}$ und IV_1 verschlüsselt wird. $M_{1,1} \oplus IV_1 = 0^n \oplus 0 = 0$, somit ist $C_1 = \mathcal{E}_K(0)$. In der zweiten Anfrage wird die Verknüpfung von $M_{1,2}$ und IV_2 verschlüsselt. $M_{1,2} \oplus IV_2 = 0^{n-1}1 \oplus 1 = 0$, wodurch in Welt 1 gilt, dass $C_2 = \mathcal{E}_K(0)$ ist. Auf Grund der deterministischen Verschlüsselung des AES-Algorithmus gilt somit in Welt 1 stets, dass $C_1 = C_2$ ist.

Ein Algorithmus für den Angreifer $\mathcal{A}_{CBCC}$ ist in Algorithmus 4 gegeben.

Algorithmus 4 Angreifer $\mathcal{A}_{CBCC}$

```
function A_CBCC()
    M_{0,1} = 0^n
    M_{1,1} = 0^n
    M_{0,2} = 0^n
    M_{1,2} = 0^{n-1}1

    C_1 = LR(M_{0,1}, M_{1,1})      ▷ Erste Anfrage an das Verschlüsselungsorakel
    C_2 = LR(M_{0,2}, M_{1,2})      ▷ Zweite Anfrage an das Verschlüsselungsorakel

    if C_1 = C_2 then return 1
    else
        return 0
    end if
end function
```

Der Angreifer $\mathcal{A}_{CBCC}$ kann somit durch ein Vergleichen der Geheimtexte C_1 und C_2 eindeutig bestimmen, in welcher Welt er sich befindet.

Auf Grund dieser Tatsache ergibt sich:

$$Pr[\mathbf{Left}^{\text{ind-cpa}}_{\text{AES-CBCC}}(\mathcal{A}_{CBCC}) = 1] = 0$$

$$Pr[\mathbf{Right}^{\text{ind-cpa}}_{\text{AES-CBCC}}(\mathcal{A}_{CBCC}) = 1] = 1$$

Damit erzielt der Angreifer $\mathcal{A}_{CBCC}$ einen Vorteil von

$$\begin{aligned}\mathsf{Adv}^{\text{ind-cpa}}_{AES\text{-}CBCC}(\mathcal{A}_{CBCC}) &= Pr[\mathbf{Right}^{\text{ind-cpa}}_{\text{AES-CBCC}}(\mathcal{A}_{CBCC}) = 1] \\ &\quad - Pr[\mathbf{Left}^{\text{ind-cpa}}_{\text{AES-CBCC}}(\mathcal{A}_{CBCC}) = 1] \\ &= 1 - 0 \\ &= 1\end{aligned}$$

□

Da es mit $\mathcal{A}_{CBCC}$ einen Angreifer gibt, der einen Vorteil von 1 erreichen kann, ist AES im CBC-C-Mode nicht sicher im Sinne der Ununterscheidbarkeit.

4.5.4 Chosen-Ciphertext-Angriff gegen den CBC-Mode

Im Folgenden soll die Sicherheit des CBC-Mode mit zufällig gewähltem Initialisierungsvektor IV betrachtet werden. Der CBC-Mode ist, im Gegensatz zum CBC-C-Mode, sicher im Sinne der Ununterscheidbarkeit gegen Chosen-Plaintext-Angriffe [7], daher soll nun stattdessen die Sicherheit gegen einen Chosen-Ciphertext-Angriff untersucht werden.
Die weiteren Betrachtungen folgen [2].

Es soll wie in Kapitel 4.3.5 beschrieben das Spiel $\mathbf{Left}^{\text{ind-cca}}_{\text{AES-CBC}}$ bzw. $\mathbf{Right}^{\text{ind-cca}}_{\text{AES-CBC}}$ gegen einen Angreifer $\mathcal{A}_{CBC}$ gespielt werden. Dazu bestimmt der Herausforderer zunächst den Schlüssel K und legt außerdem den Wert von Bit $b \in \{0,1\}$ fest, um dem Angreifer das entsprechende Spiel zu geben. Die Rückgabe des Verschlüsselungsorakels bei Eingabe eines Nachrichtenpaars setzt sich aus dem Geheimtext und dem Initialisierungsvektor zusammen. Ebenso übergibt der Angreifer bei Anfragen an das Entschlüsselungsorakel neben einem Geheimtext auch einen Initialisierungsvektor.

Satz 4.3. *Gegeben sei der AES-Algorithmus im CBC-Betriebsmodus. Es existiert ein Angreifer $\mathcal{A}_{CBC}$, der bei einem Chosen-Ciphertext-Angriff gegen AES im CBC-Mode einen Vorteil $\mathsf{Adv}^{\text{ind-cca}}_{AES\text{-}CBC}(\mathcal{A}_{CBC})$ von 1 erzielen kann.*

Beweis. Der Angreifer $\mathcal{A}_{CBC}$, der sowohl Zugriff auf das Verschlüsselungsorakel $\mathcal{E}_K(LR(\cdot,\cdot,b))$, als auch auf das Entschlüsselungsorakel $\mathcal{D}_K(\cdot)$ besitzt und den geheimen Schlüssel K nicht kennt, soll im Folgenden konstruiert werden.

Wie in Kapitel 4.3.5 beschrieben, wird der Angreifer versuchen eine vom Verschlüsselungsorakel erhaltene Rückgabe so zu modifizieren, dass diese ihm, bei Eingabe in das Entschlüsselungsorakel, Informationen liefert.

Dazu wählt er zunächst ein Nachrichtenpaar (M_0, M_1). Sei n die Blocklänge, im AES-Algorithmus 128 Bit. Dann ist

$$M_0 \leftarrow 0^n \text{ und } M_1 \leftarrow 1^n$$

Sendet der Angreifer $\mathcal{A}_{CBC}$ dieses Nachrichtenpaar (M_0, M_1) an das Verschlüsselungsorakel $\mathcal{E}_K(LR(\cdot,\cdot,b))$, erhält er von diesem $\langle IV, C\rangle$ zurück. Somit wäre $\langle IV, C\rangle$ eine unzulässige Anfrage an das Entschlüsselungsorakel.

Statt den Geheimtext C zu verändern, verändert der Angreifer den Initialisierungsvektor IV zu IV'. In IV' sollen alle Bits von IV invertiert sein, somit wird IV' festgelegt als

$$IV' \leftarrow IV \oplus 1^n$$

Diesen modifizierten Initialisierungsvektor gibt der Angreifer zusammen mit dem Geheimtext C an das Entschlüsselungsorakel. Da $\langle IV', C\rangle$ nicht identisch zu $\langle IV, C\rangle$ ist und vom Verschlüsselungsorakel noch keine weiteren Werte zurückgegeben worden sind, ist $\langle IV', C\rangle$ eine zulässige Anfrage. Vom Entschlüsselungsorakel erhält er somit einen Klartext M zurück.

Für den Angreifer ist es entscheidend zu betrachten, wie sich dieser Klartext M in den zwei Welten unterscheidet, da er mit dieser Information Wissen über die Welt gewinnen kann.

Zunächst die Betrachtung von M in Welt 0. Vom Verschlüsselungsorakel verschlüsselt wird Nachricht M_0, Geheimtext C ist somit

$$C = \mathcal{E}_K(M_0 \oplus IV) = \mathcal{E}_K(0^n \oplus IV)$$

Da die Entschlüsselungsfunktion $\mathcal{D}_K$ die Umkehrfunktion zur Verschlüsselungsfunktion $\mathcal{E}_K$ ist, gilt

$$M = \mathcal{D}_K(\langle IV', C\rangle) = \mathcal{E}_K^{-1}(C) \oplus IV'$$

Und somit

$$M = \mathcal{E}_K^{-1}(C) \oplus IV' = \mathcal{E}_K^{-1}(\mathcal{E}_K(0^n \oplus IV) \oplus IV' = (0^n \oplus IV) \oplus IV'$$

Mit der Definition von IV' ergibt sich

$$M = (0^n \oplus IV) \oplus IV' = (0^n \oplus IV) \oplus (IV \oplus 1^n) = 1^n = M_1$$

In Welt 0 gibt das Entschlüsselungsorakel $\mathcal{D}_K(\cdot)$ bei Eingabe $\langle IV', C\rangle$ also stets M_1 zurück.

Für Welt 1 können äquivalente Beobachtungen festgestellt werden. Statt M_0 verschlüsselt das Verschlüsselungsorakel nun M_1.

$$\begin{aligned} M &= \mathcal{D}_K(\langle IV', C\rangle) = \mathcal{E}_K^{-1}(C) \oplus IV' \\ &= \mathcal{E}_K^{-1}(\mathcal{E}_K(1^n \oplus IV) \oplus IV' = (1^n \oplus IV) \oplus IV' \\ &= (1^n \oplus IV) \oplus (IV \oplus 1^n) \\ &= 0^n = M_0 \end{aligned}$$

In Welt 1 gibt das Entschlüsselungsorakel $\mathcal{D}_K(\cdot)$ bei Eingabe $\langle IV', C\rangle$ also stets M_0 zurück.

Ein Algorithmus für den Angreifer $\mathcal{A}_{CBC}$ ist in Algorithmus 5 gegeben.

Algorithmus 5 Angreifer $\mathcal{A}_{CBC}$

```
function A_CBC()
    M_0 = 0^n
    M_1 = 1^n

    ⟨C, IV⟩ = LR(M_0, M_1)          ▷ Anfrage an das Verschlüsselungsorakel
    IV' = IV ⊕ 1^n

    M = D(C, IV')                   ▷ Anfrage an das Entschlüsselungsorakel
    if M = M_0 then return 1
    else
        return 0
    end if
end function
```

Durch einen Vergleich der erhaltenen Nachricht M mit den Nachrichten M_0 und M_1 aus dem selbst gewählten Nachrichtenpaar kann der Angreifer $\mathcal{A}_{CBC}$ somit eindeutig bestimmen, in welcher Welt er sich befindet. Damit ergibt sich:

$$Pr[\mathbf{Left}^{\text{ind-cca}}_{\text{AES-CBC}}(\mathcal{A}_{CBC}) = 1] = 0$$

$$Pr[\mathbf{Right}^{\text{ind-cca}}_{\text{AES-CBC}}(\mathcal{A}_{CBC}) = 1] = 1$$

Der Angreifer $\mathcal{A}_{CBC}$ erzielt somit einen Vorteil von

$$\begin{aligned}\mathsf{Adv}^{\text{ind-cca}}_{AES\text{-}CBC}(\mathcal{A}_{CBC}) &= Pr[\mathbf{Right}^{\text{ind-cca}}_{\text{AES-CBC}}(\mathcal{A}_{CBC}) = 1] - Pr[\mathbf{Left}^{\text{ind-cca}}_{\text{AES-CBC}}(\mathcal{A}_{CBC}) = 1] \\ &= 1 - 0 \\ &= 1\end{aligned}$$

□

Mit $\mathcal{A}_{CBC}$ existiert somit ein Angreifer, der einen Vorteil von 1 erzielen kann. Der CBC-Mode ist daher nicht sicher im Sinne der Ununterscheidbarkeit gegen einen Chosen-Ciphertext-Angriff.

4.5.5 Zusammenfassung

In den letzten Abschnitten wurde die Sicherheit im Sinne der Ununterscheidbarkeit des ECB-, sowie des CBC-C-Modes gegen einen Chosen-Plaintext-Angriff, sowie die Sicher-

heit im Sinne der Ununterscheidbarkeit des CBC-Mode gegen einen Chosen-Ciphertext-Angriff genauer betrachtet. Im Folgenden sollen die weiteren, noch nicht genauer behandelten Betriebsmodi, die in Kapitel 2.2.3 vorgestellt wurden, ebenfalls kurz betrachtet werden.

Im Allgemeinen bietet keine deterministische, zustandslose Verschlüsselung Sicherheit im Sinne der Ununterscheidbarkeit. Dies gilt logischerweise auch für deterministische, zustandslose Betriebsmodi [2].
Der CFB-, sowie der OFB-Mode bieten Sicherheit im Sinne der Ununterscheidbarkeit gegen einen Chosen-Plaintext-Angriff, jedoch nur bei einem zufällig gewähltem Initialisierungsvektor [7]. Auch der CTR-Mode bietet Sicherheit im Sinne der Ununterscheidbarkeit gegen einen Chosen-Plaintext-Angriff [2].

Keiner der vorgestellten Betriebsmodi bietet Sicherheit im Sinne der Ununterscheidbarkeit gegen einen Chosen-Ciphertext-Angriff [2].

Von den in Kapitel 2.2.3 vorgestellten Betriebsmodi wird allgemein der CTR-Mode als der beste Betriebsmodus empfohlen. Obwohl keine Sicherheit im Sinne der Ununterscheidbarkeit gegen Chosen-Ciphertext-Angriffe gegeben ist, bietet der CTR-Mode nicht weniger Sicherheit als die anderen Betriebsmodi und ist zudem im Vergleich simpler und performanter [15].

Um Sicherheit im Sinne der Ununterscheidbarkeit gegen Chosen-Ciphertext-Angriffe erreichen zu können, sollte ein *Authenticated-Encryption Mode* eingesetzt werden [15].

4.6 Beispielanwendung auf asymmetrische Kryptosysteme

4.6.1 Chosen-Plaintext-Angriff gegen RSA

Es soll das in Kapitel 3.2 vorgestellte RSA-Kryptosystem auf Sicherheit im Sinne der Ununterscheidbarkeit gegen einen Chosen-Plaintext-Angriff untersucht werden.

Nach der Definition aus Kapitel 4.4.1 wird mit einem Angreifer $\mathcal{A}_{RSA}$ das Spiel $\mathbf{Left}_{\text{RSA}}^{\text{ind-cpa}}$ bzw. $\mathbf{Right}_{\text{RSA}}^{\text{ind-cpa}}$ gespielt.
Dazu bestimmt der Herausforderer als Erstes das Bit b, mit $b \in \{0, 1\}$ und gibt dem Angreifer dann entsprechend das Spiel $\mathbf{Left}_{\text{RSA}}^{\text{ind-cpa}}$ bei $b = 0$, bzw. das Spiel $\mathbf{Right}_{\text{RSA}}^{\text{ind-cpa}}$ bei $b = 1$.

Satz 4.4. *Gegeben sei das in Kapitel 3.2 vorgestellte RSA-Kryptosystem. Es existiert ein Angreifer $\mathcal{A}_{RSA}$, der bei einem Chosen-Plaintext-Angriff gegen das RSA-Kryptosystem einen Vorteil $\mathsf{Adv}_{RSA}^{\text{ind-cpa}}(\mathcal{A}_{RSA})$ von 1 erzielen kann.*

Beweis. Im ersten Schritt des Spiels, der *Initialize*-Prozedur, bestimmt der Herausforderer das Schlüsselpaar (K, K'), wobei K der private Schlüssel und K' der öffentliche Schlüssel ist.

Der Angreifer $\mathcal{A}_{RSA}$ bekommt daraufhin den öffentlichen Schlüssel K' als Eingabe übergeben und zudem Zugriff auf das jeweils in den Welten definierte Verschlüsselungsorakel $\mathcal{E}_{K'}(LR(\cdot, \cdot, b))$.

Die Arbeitsweise des Angreifers $\mathcal{A}_{RSA}$ wird im Folgenden beschrieben und ist außerdem im Algorithmus 6 gegeben.

1. Wähle das Nachrichtenpaar (M_0, M_1), mit $M_0 \neq M_1$ und $|M_0| = |M_1|$
2. Berechne $C_0 \leftarrow \mathcal{E}_{K'}(M_0)$ und $C_1 \leftarrow \mathcal{E}_{K'}(M_1)$ mit Hilfe des öffentlichen Schlüssels K'
3. Sende (M_0, M_1) an das Verschlüsselungsorakel $\mathcal{E}_{K'}(LR(\cdot, \cdot, b))$ und erhalte von diesem den Geheimtext C zurück
4. Wenn $C = C_0$, dann setze $b' = 0$. Wenn $C = C_1$, dann setze $b' = 1$.
5. Gebe b' aus.

Algorithmus 6 Angreifer $\mathcal{A}_{RSA}$

function $\mathcal{A}_{RSA}(K')$
  $M_0 \in \{0,1\}^x$
  $M_1 \in \{0,1\}^x$ ▷ $M_0 \neq M_1$ und $|M_0| = |M_1|$

  $C_0 = \mathcal{E}_{K'}(M_0)$
  $C_1 = \mathcal{E}_{K'}(M_1)$

  $C = \mathbf{LR}(M_0, M_1)$ ▷ Anfrage an das Verschlüsselungsorakel

  if $C = C_0$ **then return** 0
  end if
  if $C = C_1$ **then return** 1
  end if
end function

Die in Kapitel 3.2 beschriebene Verschlüsselung ist deterministisch. Dieser Umstand hat, wie in Kapitel 4.2.1 beschrieben, zur Folge, dass ein Klartext x stets zum selben Geheimtext y verschlüsselt wird.

Da der Angreifer Zugriff auf den öffentlichen Schlüssel K' hat, kann er sich einen beliebigen Klartext selbst verschlüsseln und somit auch die Geheimtexte C_0 und C_1 der Nachrichten M_0 und M_1 erhalten. Der Geheimtext C, welchen er vom Verschlüsselungsorakel zurückbekommt, ist ebenfalls die Verschlüsselung von einer der Nachrichten aus

M_0 und M_1. Da die Verschlüsselung deterministisch ist, gilt außerdem $C \in \{C_0, C_1\}$. C ist also definitiv identisch zu einem der vom Angreifer selbst berechneten Geheimtexte.

Durch einfaches Vergleichen der eigenen Geheimtexte mit dem erhaltenen Geheimtext kann der Angreifer $\mathcal{A}_{RSA}$ folglich bestimmen, mit welchem der eigenen Geheimtexte der erhaltene Geheimtext C übereinstimmt und somit die Welt, in der er sich befindet, eindeutig bestimmen.

Dieser Umstand bedeutet, dass für Angreifer $\mathcal{A}_{RSA}$ das Spiel $\mathbf{Left}_{\text{RSA}}^{\text{ind-cpa}}$ stets 0 ausgibt, während Spiel $\mathbf{Right}_{\text{RSA}}^{\text{ind-cpa}}$ stets 1 ausgibt. Daraus folgt:

$$Pr[\mathbf{Left}_{\text{RSA}}^{\text{ind-cpa}}(\mathcal{A}_{RSA}) = 1] = 0 \text{ und } Pr[\mathbf{Right}_{\text{RSA}}^{\text{ind-cpa}}(\mathcal{A}_{RSA}) = 1] = 1$$

Der Vorteil des Angreifers $\mathcal{A}_{RSA}$ liegt somit bei

$$\begin{aligned} \mathsf{Adv}_{RSA}^{\text{ind-cpa}}(\mathcal{A}_{RSA}) &= Pr[\mathbf{Right}_{\text{RSA}}^{\text{ind-cpa}}(\mathcal{A}_{RSA}) = 1] - Pr[\mathbf{Left}_{\text{RSA}}^{\text{ind-cpa}}(\mathcal{A}_{RSA}) = 1] \\ &= 1 - 0 \\ &= 1 \end{aligned}$$

□

Dadurch ist gezeigt, dass mit $\mathcal{A}_{RSA}$ ein Angreifer existiert, der mit begrenzten Ressourcen arbeitet und einen IND-CPA-Vorteil $\mathsf{Adv}_{RSA}^{\text{ind-cpa}}(\mathcal{A}_{RSA}) = 1$ erzielen kann.

Somit ist der RSA-Algorithmus, wie er in Kapitel 3.2 beschrieben ist, nicht sicher im Sinne der Ununterscheidbarkeit.

RSA-OAEP

Das RSA-Verfahren, wie es in Kapitel 3.2 beschrieben wird, wird auf Grund dieser Unsicherheit auch als „Lehrbuch-RSA“ bezeichnet [14]. In der Praxis werden daher stattdessen modifizierte Varianten eingesetzt, die eine hohe Sicherheit gewährleisten können.

Eine solche sichere Variante ist RSA-OAEP, welches das von Bellare und Rogaway entwickelte OAEP-Verfahren („Optimal Asymmetric Encryption Padding“) einsetzt und im Standard PKCS#1 enthalten ist.

RSA-OAEP soll im Folgenden kurz vorgestellt werden. Die Beschreibungen hierzu basieren auf denen von Johannes A. Buchmann [8].

Gegeben seien der öffentliche Schlüssel (e, n), sowie der private Schlüssel d des RSA-Algorithmus.

Als Erstes wird ein $t \in \mathbb{N}$ so gewählt, dass die Laufzeit eines Angreifers stets deutlich kleiner als 2^t ist. Sei k die Länge des RSA-Moduls n mit $n = p \cdot q$. Dann ist $k \geq 1024$ und ℓ definiert als $\ell = k - t - 1$.

Es werden zwei Funktionen G, H benötigt. $G\colon \{0,1\}^t \to \{0,1\}^\ell$ ist die Expansionsfunktion, $H\colon \{0,1\}^\ell \to \{0,1\}^t$ die Kompressionsfunktion. Beide Funktionen sind öffentlich bekannt.

Verschlüsselung Zur Verschlüsselung eines Klartextes $m \in \{0,1\}^\ell$ wird zunächst eine Zufallszahl $r \in \{0,1\}^t$ gewählt und mit dieser der Klartext m zu $(m \oplus G(r))$ randomisiert. Der genutzte Zufallswert r wird dann als $(r \oplus H(m \oplus G(r)))$ maskiert und an den randomisierten Klartext angehängt. Der Geheimtext c wird schließlich durch Verschlüsselung der entstandenen Nachricht mit dem RSA-Algorithmus und dem öffentlichen Schlüssel (e, n) erzeugt.

Definition 4.13 (RSA-OAEP Verschlüsselung)**.** *Gegeben seien der öffentliche Schlüssel (e, n), die öffentlichen Funktionen $G\colon \{0,1\}^t \to \{0,1\}^\ell$ und $H\colon \{0,1\}^\ell \to \{0,1\}^t$, sowie ein Klartext $m \in \{0,1\}^\ell$ und eine Zufallszahl $r \in \{0,1\}^t$. Die Verschlüsselung von m zu einem Geheimtext c erfolgt durch*

$$c = x^e \mod n \text{ mit } x = (m \oplus G(r)) \| (r \oplus H(m \oplus G(r)))$$

Entschlüsselung Die Entschlüsselung eines Geheimtextes c erfolgt in drei Schritten. Als Erstes wird mit der RSA-Entschlüsselung aus c die Verknüpfung aus randomisiertem Klartext und maskierter Zufallszahl r wiederhergestellt. Anschließend kann mit Hilfe der öffentlichen Funktionen G und H die Zufallszahl r berechnet und schließlich der Klartext m ermittelt werden.

Definition 4.14 (RSA-OAEP Entschlüsselung)**.** *Gegeben seien der private Schlüssel d, die öffentlichen Funktionen $G\colon \{0,1\}^t \to \{0,1\}^\ell$ und $H\colon \{0,1\}^\ell \to \{0,1\}^t$, sowie ein Geheimtext c. Die Entschlüsselung des Geheimtextes c zum Klartext m erfolgt durch*

$$(m \oplus G(r)) \| (r \oplus H(m \oplus G(r))) = c^d \mod n$$
$$r = (r \oplus H(m \oplus G(r))) \oplus H(m \oplus G(r))$$
$$m = (m \oplus G(r)) \oplus G(r)$$

Zusammenfassung Der Determinismus, welcher das Lehrbuch-RSA-Kryptosystem unsicher macht, wird in RSA-OAEP gelöst, indem der Klartext m mit $G(r)$ zu einer von Zufallswerten abhängigen Nachricht $(m \oplus G(r))$ transformiert wird.

RSA-OAEP ist sicher im Sinne der Ununterscheidbarkeit gegen Chosen-Plaintext-Angriffe (IND-CPA). Auch die Sicherheit gegen nicht-adaptive (IND-CCA1), sowie adaptive Chosen-Ciphertext-Angriffe (IND-CCA2) konnte bewiesen werden [12].

4.6.2 Beweis der IND-CPA-Sicherheit von ElGamal

Im Folgenden soll die IND-CPA-Sicherheit des in Kapitel 3.3 vorgestellten ElGamal-Kryptosystems bewiesen werden. Der Beweis erfolgt mittels einer Sequenz von Spielen, wie es in Kapitel 2.3.3 vorgestellt wurde.

Dieses Kapitel folgt [16].

Decisional-Diffie-Hellman-Problem (DDH)

Da das *Decisional-Diffie-Hellman-Problem*, kurz DDH, ein elementarer Bestandteil des folgenden Sicherheitsbeweises ist, soll dieses zunächst kurz erläutert werden.

Definition 4.15 (Decisional-Diffie-Hellman-Problem)**.** *Sei $\mathcal{G}$ eine zyklische Gruppe der Ordnung q, $g \in \mathcal{G}$ ein Erzeugerelement und x, y, z zufällige Elemente aus $\mathbb{Z}_q$. Beim Decisional-Diffie-Hellman-Problem muss ein Angreifer zwischen den Tripeln (g^x, g^y, g^{xy}) und (g^x, g^y, g^z) unterscheiden.*

Der DDH-Vorteil gibt den Vorteil des Angreifers bei dieser Unterscheidung gegenüber einfachem Raten an.

Definition 4.16 (DDH-Annahme)**.** *Die DDH-Annahme für eine Gruppe $\mathcal{G}$ besagt, dass der DDH-Vorteil eines jeden effizienten Algorithmus vernachlässigbar klein ist.*

Sicherheitsbeweis

Im Folgenden soll der Sicherheitsbeweis mittels einer Sequenz von Spielen dargestellt werden.

Entsprechend Kapitel 2.3.3 wird zunächst das Spiel 0 festgelegt. Dieses Spiel entspricht dem ursprünglichen Angriffsspiel und ist daher identisch zur Definition des Spiels $\mathbf{Guess}_{\mathrm{EG}}^{\text{ind-cpa}}$. Die Schlüsselwahl in der Initialize-Prozedur, sowie die Verschlüsselung in der LR-Prozedur entsprechen den Beschreibungen aus Kapitel 3.3. Die gewählte zyklische Gruppe $\mathcal{G}$ sei eine Gruppe mit primer Ordnung q.

Aus Spiel 0 wird schließlich mit Hilfe der beschriebenen Übergänge das Spiel 1 konstruiert. Sowohl Spiel 0, als auch Spiel 1 werden im Folgenden gegeben.

Spiel 0
procedure Initialize
$b \xleftarrow{\$} \{0,1\}$
$x \leftarrow \mathbb{Z}_q, X \leftarrow g^x$
return X
procedure LR(M_0, M_1)
$y \leftarrow \mathbb{Z}_q, c_1 \leftarrow g^y$
$Z \leftarrow X^y$
$c_2 \leftarrow M_b \cdot Z$
return (c_1, c_2)
procedure Finalize(b')
return $(b = b')$

Spiel 1
procedure Initialize
$b \xleftarrow{\$} \{0,1\}$
$x \leftarrow \mathbb{Z}_q, X \leftarrow g^x$
return X
procedure LR(M_0, M_1)
$y \leftarrow \mathbb{Z}_q, c_1 \leftarrow g^y$
$z \leftarrow \mathbb{Z}_q, Z \leftarrow g^z$
$c_2 \leftarrow M_b \cdot Z$
return (c_1, c_2)
procedure Finalize(b')
return $(b = b')$

Spiel 0. Das zugehörige Ereignis S_0 sei die Ausgabe von $b = b'$. Wie in Kapitel 2.3.3 beschrieben, entspricht S_0 dem Ereignis S, sprich $S = S_0$.

Der Vorteil des Angreifers von $2 \cdot Pr[S_0] - 1$ in Spiel 0 ergibt sich direkt aus der Definition des Spiels $\mathbf{Guess}_{\mathrm{EG}}^{\text{ind-cpa}}$. Da die Sicherheit bewiesen werden soll, soll der Vorteil des Angreifers gegen 0 gehen. Daraus folgt eine Zielwahrscheinlichkeit $\mathcal{P} = \frac{1}{2}$ für den Beweis.

Spiel 1. Das zugehörige Ereignis S_1 sei die Ausgabe von $b = b'$ in Spiel 1. Die Unterschiede im Vergleich zu Spiel 0 sind zur Markierung grau hinterlegt. Der Übergang von Spiel 0 ist ein Übergang basierend auf Ununterscheidbarkeit.

Der Unterschied zwischen Spiel 0 und Spiel 1 ist, dass der Wert von Z in Spiel 1 von einer dritten zufälligen Zahl z abhängt und der Geheimtext c somit unterschiedlich berechnet wird. In Spiel 0 ergibt sich für den Geheimtext $c = (g^y, m_b \cdot g^{xy})$, in Spiel 1 ergibt sich $c = (g^y, m_b \cdot g^z)$.

Um den Beweis der IND-CPA-Sicherheit für das ElGamal-Kryptosystem abschließen zu können, müssen zuvor noch zwei Behauptungen bewiesen werden.

Satz 4.5. *Der Übergang von Spiel 0 zu Spiel 1 ist ein Übergang basierend auf Ununterscheidbarkeit. Es gilt $|Pr[S_0] - Pr[S_1]| = \varepsilon$ und somit ist $Pr[S_0]$ vernachlässigbar nah an $Pr[S_1]$.*

Beweis. Bei einem Übergang basierend auf Ununterscheidbarkeit wird nach Kapitel 2.3.3 die Menge, aus der die Eingabe für den Angreifer stammt, ausgetauscht. Die ursprüngliche und die neue Menge müssen ununterscheidbar sein. Um dies zu zeigen, wird der Algorithmus $D^{\mathcal{A}}$, welcher zwischen diesen beiden Mengen unterscheidet, konstruiert.

Bei Eingabe eines Tripels (g^x, g^y, g^{xy}) verhält sich $D^{\mathcal{A}}$ entsprechend Spiel 0, bei Eingabe eines Tripels (g^x, g^y, g^z) entsprechend Spiel 1.

Daraus folgt:

$$Pr[x, y \leftarrow \mathbb{Z}_q \colon D^{\mathcal{A}}(g^x, g^y, g^{xy})] = Pr[S_0]$$

$$Pr[x, y, z \leftarrow \mathbb{Z}_q \colon D^{\mathcal{A}}(g^x, g^y, g^z)] = Pr[S_1]$$

Aus der Definition des Decisional-Diffie-Hellman-Problems folgt:

$$\mathsf{Adv}^{\text{ddh}}_{\mathcal{G},g,q}(D^{\mathcal{A}}) = |Pr[S_0] - Pr[S_1]|$$

Für die Gruppe $\mathcal{G}$, für die die DDH-Annahme gilt, ergibt sich somit, dass die Differenz $|Pr[S_0] - Pr[S_1]|$ vernachlässigbar klein ist.

$$\mathsf{Adv}^{\text{ddh}}_{\mathcal{G},g,q}(D^{\mathcal{A}}) = |Pr[S_0] - Pr[S_1]| = \varepsilon$$

Beim Übergang von Spiel 0 zu Spiel 1 handelt es sich somit um einen korrekten Übergang basierend auf Ununterscheidbarkeit. Damit gilt, dass $Pr[S_0]$ vernachlässigbar nah an $Pr[S_1]$ ist. □

Satz 4.6. *$Pr[S_1]$ ist gleich oder vernachlässigbar nah an der Zielwahrscheinlichkeit $\mathcal{P}$. Es gilt $|Pr[S_1] - \mathcal{P}| \leq \varepsilon$.*

Beweis. Der Angreifer $\mathcal{A}_{EG}$ kann in Spiel 1 mit dem Geheimtext $c = (c_1, c_2)$ keine Informationen über den Wert von b gewinnen, da Z ein zufälliges Gruppenelement und unabhängig von X ist. Z dient bei der Verschlüsselung als One-Time-Pad für den Klartext m_b. Da c_2 durch $c_2 \leftarrow m_b \cdot Z$ errechnet wird, ist auch c_2 ein zufälliges Gruppenelement und unabhängig von X, c_1 und b.

Ohne weitere Informationen bleibt dem Angreifer nur das Raten und somit eine Wahrscheinlichkeit $Pr[S_1] = \frac{1}{2}$. Daraus folgt $|Pr[S_1] - \mathcal{P}| = 0 \leq \varepsilon$. Somit gilt, dass $Pr[S_1]$ gleich oder vernachlässigbar nah an der Zielwahrscheinlichkeit $\mathcal{P}$ ist. □

Abschluss

Satz 4.7. *Gegeben sei das in Kapitel 3.3 vorgestellte ElGamal-Kryptosystem. Sei $\mathcal{A}_{EG}$ ein Angreifer mit realistisch beschränkten Ressourcen. Bei einem Chosen-Plaintext-Angriff gegen das ElGamal-Kryptosystem kann $\mathcal{A}_{EG}$ maximal einen vernachlässigbar kleinen Vorteil erzielen. Es gilt*

$$\mathsf{Adv}_{EG}^{\text{ind-cpa}}(\mathcal{A}_{EG}) = \varepsilon$$

Beweis. Aus den zuvor erbrachten Beweisen geht hervor, dass $|Pr[S_0] - Pr[S_1]| = \varepsilon$ und somit vernachlässigbar klein ist. Außerdem wurde beweisen, dass $Pr[S_1] = \frac{1}{2} = \mathcal{P}$ ist. Daraus folgt

$$\mathsf{Adv}_{EG}^{\text{ind-cpa}}(\mathcal{A}_{EG}) = 2 \cdot Pr[S_0] - 1 = \varepsilon$$

□

Damit ist gezeigt, dass das ElGamal-Kryptosystem bei Verwendung einer Gruppe $\mathcal{G}$, für die die DDH-Annahme gilt, sicher im Sinne der Ununterscheidbarkeit gegen einen Chosen-Plaintext-Angriff ist.

5 Ausblick

5.1 Weitere Definitionen der Ununterscheidbarkeit

Es existieren insgesamt vier verschiedene Sicherheitsdefinitionen zu Ununterscheidbarkeit bei einem symmetrischen Kryptosystem $\mathcal{SE}$. Jede Definition kann jeweils gegen einen Angriff mit Klartextwahl (Chosen-Plaintext-Angriff) und einen Angriff mit Geheimtextwahl (Chosen-Ciphertext-Angriff) getestet werden. Diese Arbeit konzentrierte sich auf die Left-Or-Right Indistinguishability, kurz IND oder auch LOR. Im Folgenden sollen drei weitere Definitionen vorgestellt und die Beziehungen dieser untereinander und mit der Left-Or-Right Indistinguishability betrachtet werden.

Für die Betrachtung der Sicherheit werden auch bei diesen Definitionen Spiele zwischen einem Angreifer und einem Herausforderer definiert. Im Rahmen dieser Arbeit sollen diese Spiele jedoch nur kurz beschrieben werden.

Wie bei der Definition der Left-Or-Right Indistinguishability werden stets Angreifer mit realistisch beschränkten Ressourcen bezüglich Laufzeit und erlaubter Anzahl Anfragen an die Orakel betrachtet.

Die Beschreibungen folgen [4] und [2].

5.1.1 Real-Or-Random Indistinguishability

Wie bei der IND-Definition wird ein Spiel zwischen einem Angreifer $\mathcal{A}$ und einem Herausforderer betrachtet. Der Unterschied bei der *Real-Or-Random Indistinguishability* (kurz ROR) ist, dass der Angreifer statt eines Nachrichtenpaares nur einen Klartext M an das Verschlüsselungsorakel sendet. Abhängig vom gewählten Bit $b \in \{0, 1\}$ gibt dieses, statt der Verschlüsselung eines Klartextes aus dem Nachrichtenpaar, entweder die Verschlüsselung des Klartextes M oder die einer anderen, zufällig produzierten Nachricht gleicher Länge zurück.

Das Verschlüsselungsorakel $\mathcal{E}_K(RR(\cdot, b))$ mit Eingabe eines Klartextes M kann also wie folgt beschrieben werden:

1. Wenn $b = 1$, dann berechne $C \leftarrow \mathcal{E}_K(M)$, wenn $b = 0$, dann berechne $C \leftarrow \mathcal{E}_K(r)$, wobei $r \xleftarrow{\$} \{0, 1\}^{|M|}$ ist
2. Gebe C zurück

Es wird ROR-CPA für Chosen-Plaintext-Angriffe und ROR-CCA für Chosen-Ciphertext-Angriffe definiert. Zugriff auf das Verschlüsselungsorakel $\mathcal{E}_K(RR(\cdot, b))$ bekommt der

Angreifer sowohl bei ROR-CPA, als auch bei ROR-CCA. Bei letzterem bekommt der Angreifer zusätzlich Zugriff auf ein Entschlüsselungsorakel $\mathcal{D}_K(\cdot)$, an das keine Rückgaben des Verschlüsselungsorakels gegeben werden dürfen.

Es werden, wie bei IND, zwei Spiele betrachtet, von denen eins, abhängig vom gewählten Bit b, gespielt wird. Die Spiele sowohl für ROR-CPA, als auch für ROR-CCA sind im Folgenden gegeben.

ROR-CPA

Spiel $\mathbf{Left}_{\mathcal{SE}}^{\text{ror-cpa}}$

procedure Initialize
$K \xleftarrow{\$} \mathcal{K}$
procedure RR(M)
$r \xleftarrow{\$} \{0,1\}^{\lvert M\rvert}$
return $C \xleftarrow{\$} \mathcal{E}_K(r)$
procedure Finalize(b')
return b'

Spiel $\mathbf{Right}_{\mathcal{SE}}^{\text{ror-cpa}}$

procedure Initialize
$K \xleftarrow{\$} \mathcal{K}$
procedure RR(M)
return $C \xleftarrow{\$} \mathcal{E}_K(M)$
procedure Finalize(b')
return b'

ROR-CCA

Spiel $\mathbf{Left}_{\mathcal{SE}}^{\text{ror-cca}}$

procedure Initialize
$K \xleftarrow{\$} \mathcal{K}$
procedure RR(M)
$r \xleftarrow{\$} \{0,1\}^{\lvert M\rvert}$
return $C \xleftarrow{\$} \mathcal{E}_K(r)$
procedure D(C)
return $M \leftarrow \mathcal{D}_K(C)$
procedure Finalize(b')
if $\mathcal{A}$ queried an illegitimate ciphertext
return 0
else return b'

Spiel $\mathbf{Right}_{\mathcal{SE}}^{\text{ror-cca}}$

procedure Initialize
$K \xleftarrow{\$} \mathcal{K}$
procedure RR(M)
return $C \xleftarrow{\$} \mathcal{E}_K(M)$
procedure D(C)
return $M \leftarrow \mathcal{D}_K(C)$
procedure Finalize(b')
if $\mathcal{A}$ queried an illegitimate ciphertext
return 0
else return b'

ROR-Vorteil Ziel des Angreifers ist es, wie bei der Left-Or-Right Indistinguishability, den Wert von b zu bestimmen. Der Vorteil, den der Angreifer dabei erzielt, wird bei ROR-CPA mit $\mathsf{Adv}_{\mathcal{SE}}^{\text{ror-cpa}}(\mathcal{A})$ und bei ROR-CCA mit $\mathsf{Adv}_{\mathcal{SE}}^{\text{ror-cca}}(\mathcal{A})$ bezeichnet.

Definition 5.1 (ROR-Vorteil)**.** *Sei $\mathcal{SE} = (\mathcal{K}, \mathcal{E}, \mathcal{D})$ ein symmetrisches Kryptosystem und Angreifer $\mathcal{A}$ ein Algorithmus. Der ROR-CPA-Vorteil $\mathsf{Adv}_{\mathcal{SE}}^{\text{ror-cpa}}(\mathcal{A})$ des Angreifers $\mathcal{A}$ ist definiert als*

$$\mathsf{Adv}^{\text{ror-cpa}}_{\mathcal{SE}}(\mathcal{A}) = Pr[\boldsymbol{Right}^{ror\text{-}cpa}_{\mathcal{SE}}(\mathcal{A}) = 1] - Pr[\boldsymbol{Left}^{ror\text{-}cpa}_{\mathcal{SE}}(\mathcal{A}) = 1]$$

Der ROR-CCA-Vorteil $\mathsf{Adv}^{\text{ror-cca}}_{\mathcal{SE}}(\mathcal{A})$ *des Angreifers* $\mathcal{A}$ *ist definiert als*

$$\mathsf{Adv}^{\text{ror-cca}}_{\mathcal{SE}}(\mathcal{A}) = Pr[\boldsymbol{Right}^{ror\text{-}cca}_{\mathcal{SE}}(\mathcal{A}) = 1] - Pr[\boldsymbol{Left}^{ror\text{-}cca}_{\mathcal{SE}}(\mathcal{A}) = 1]$$

5.1.2 Find-Then-Guess Security

Bei der *Find-Then-Guess Security* (kurz FTG) arbeitet der Angreifer in zwei Phasen.

In der ersten Phase, der *Find*-Phase, versucht er zwei Nachrichten (M_0, M_1) gleicher Länge zu finden, bei denen er meint, die Verschlüsselungen unterscheiden zu können. Dabei sammelt er Zustandsinformationen s, die ihm später bei der Unterscheidung helfen sollen.

In der folgenden *Guess*-Phase erhält der Angreifer den Geheimtext C, welcher die Verschlüsselung einer der Nachrichten aus dem Nachrichtenpaar (M_0, M_1) ist, zusammen mit zugehörigen Zustandsinformationen s. In dieser Phase muss der Angreifer eine Vermutung abgeben, zu welchem der Klartexte M_0, M_1 der Geheimtext C gehört.

In beiden Phasen hat der Angreifer sowohl bei FTG-CPA, also auch bei FTG-CCA Zugriff auf das Verschlüsselungsorakel $\mathcal{E}_K(\cdot)$, bei FTG-CCA zudem Zugriff auf das Entschlüsselungsorakel $\mathcal{D}_K(\cdot)$. Das Verschlüsselungsorakel erhält in dieser Definition, anders als bei IND und ROR, nur einen Klartext als Eingabe und gibt lediglich die Verschlüsselung dieses Klartextes zurück, ohne dem Angreifer den Schlüssel zu offenbaren.

Es werden, wie bei IND, für FTG-CPA und FTG-CCA je zwei Spiele betrachtet, von denen eins, abhängig vom gewählten Bit b, gespielt wird. Die Spiele sind im Folgenden gegeben.

FTG-CPA

Experiment $\mathbf{Exp}^{\text{ftg-cpa-1}}_{\mathcal{SE}}(\mathcal{A})$	Experiment $\mathbf{Exp}^{\text{ftg-cpa-0}}_{\mathcal{SE}}(\mathcal{A})$
$K \xleftarrow{\$} \mathcal{K}$	$K \xleftarrow{\$} \mathcal{K}$
$(M_0, M_1, s) \leftarrow \mathcal{A}^{\mathcal{E}_K(\cdot)}_{cpa}(find)$	$(M_0, M_1, s) \leftarrow \mathcal{A}^{\mathcal{E}_K(\cdot)}_{cpa}(find)$
$C \leftarrow \mathcal{E}_K(M_1)$	$C \leftarrow \mathcal{E}_K(M_0)$
$b' \leftarrow \mathcal{A}^{\mathcal{E}_K(\cdot)}_{cpa}(guess, C, s)$	$b' \leftarrow \mathcal{A}^{\mathcal{E}_K(\cdot)}_{cpa}(guess, C, s)$
return b'	return b'

FTG-CCA

Experiment $\mathbf{Exp}^{\text{ftg-cca-1}}_{\mathcal{SE}}(\mathcal{A})$	Experiment $\mathbf{Exp}^{\text{ftg-cca-0}}_{\mathcal{SE}}(\mathcal{A})$
$K \xleftarrow{\$} \mathcal{K}$	$K \xleftarrow{\$} \mathcal{K}$
$(M_0, M_1, s) \leftarrow \mathcal{A}^{\mathcal{E}_K(\cdot),\mathcal{D}_K(\cdot)}_{cca}(find)$	$(M_0, M_1, s) \leftarrow \mathcal{A}^{\mathcal{E}_K(\cdot),\mathcal{D}_K(\cdot)}_{cca}(find)$
$C \leftarrow \mathcal{E}_K(M_1)$	$C \leftarrow \mathcal{E}_K(M_0)$
$b' \leftarrow \mathcal{A}^{\mathcal{E}_K(\cdot),\mathcal{D}_K(\cdot)}_{cca}(guess, C, s)$	$b' \leftarrow \mathcal{A}^{\mathcal{E}_K(\cdot),\mathcal{D}_K(\cdot)}_{cca}(guess, C, s)$
return b'	return b'

In den Spielen zu FTG-CCA ist jeweils der erhaltene Geheimtext C die unzulässige Eingabe in das Entschlüsselungsorakel $\mathcal{D}_K(\cdot)$.

FTG-Vorteil Der Angreifer gewinnt das Spiel, wenn seine Vermutung über den Klartext korrekt ist. Seine Siegchancen dürfen in einem sicheren Kryptosystem nicht wesentlich höher sein, als bei einfachem Raten. Der Vorteil, den der Angreifer dabei erzielt, wird bei FTG-CPA mit $\mathsf{Adv}_{\mathcal{SE}}^{\text{ftg-cpa}}(\mathcal{A})$ und bei FTG-CCA mit $\mathsf{Adv}_{\mathcal{SE}}^{\text{ftg-cca}}(\mathcal{A})$ bezeichnet.

Definition 5.2 (FTG-Vorteil)**.** *Sei $\mathcal{SE} = (\mathcal{K}, \mathcal{E}, \mathcal{D})$ ein symmetrisches Kryptosystem und Angreifer $\mathcal{A}$ ein Algorithmus mir realistisch beschränkten Ressourcen. Der FTG-CPA-Vorteil $\mathsf{Adv}_{\mathcal{SE}}^{\text{ftg-cpa}}(\mathcal{A})$ des Angreifers $\mathcal{A}$ ist definiert als*

$$\mathsf{Adv}_{\mathcal{SE}}^{\text{ftg-cpa}}(\mathcal{A}) = Pr[\boldsymbol{Exp}_{\mathcal{SE}}^{\textit{ftg-cpa-1}}(\mathcal{A}) = 1] - Pr[\boldsymbol{Exp}_{\mathcal{SE}}^{\textit{ftg-cpa-0}}(\mathcal{A}) = 1]$$

Der FTG-CCA-Vorteil $\mathsf{Adv}_{\mathcal{SE}}^{\text{ftg-cca}}(\mathcal{A})$ des Angreifers $\mathcal{A}$ ist definiert als

$$\mathsf{Adv}_{\mathcal{SE}}^{\text{ftg-cca}}(\mathcal{A}) = Pr[\boldsymbol{Exp}_{\mathcal{SE}}^{\textit{ftg-cca-1}}(\mathcal{A}) = 1] - Pr[\boldsymbol{Exp}_{\mathcal{SE}}^{\textit{ftg-cca-0}}(\mathcal{A}) = 1]$$

5.1.3 Semantische Sicherheit

Die Definition der *Semantischen Sicherheit* (kurz SEM) stammt von Goldwasser und Micali und übertragt Shannons Idee der perfekten Sicherheit in ein Szenario, in dem die mögliche Rechenleistung eines Angreifers berücksichtigt wird. Sie besagt, dass ein sicheres Kryptosystem alle Informationen über einen Klartext im Geheimtext verstecken muss. Das bedeutet, dass Informationen über den Klartext, die ein Angreifer mit realistischem Aufwand aus einem Geheimtext entnehmen kann, auch ohne den Geheimtext für den Angreifer zu erhalten sein sollten. Umgekehrt folgt daraus, dass ein Angreifer bei Besitz eines Geheimtextes, keine weiteren Informationen gewinnen können darf, als er ohne diesen schon besitzt.

Der Angreifer arbeitet in zwei Phasen. In der ersten Phase, der *Select*-Phase, sucht er eine Nachrichtenverteilung $\mathcal{M}$, in der alle Nachrichten die selbe Länge haben und die, seiner Meinung nach, für ihn vorteilhaft für das Spiel ist.

Der Herausforderer entnimmt nach der Select-Phase aus dieser Nachrichtenverteilung $\mathcal{M}$ zwei zufällig gewählte Nachrichten M und M'. Entsprechend eines Bits $b \in \{0, 1\}$ wird der Geheimtext C aus einer dieser Nachrichten erzeugt: Für $b = 0$ gilt $C \leftarrow \mathcal{E}_K(M')$, für $b = 1$ gilt $C \leftarrow \mathcal{E}_K(M)$. Der Geheimtext C wird dann an den Angreifer weitergegeben.

In der darauffolgenden zweiten Phase des Angreifers, der *Predict*-Phase, wählt dieser eine Funktion $f\colon \mathcal{M} \to \{0,1\}^*$ und versucht mit Hilfe des erhaltenen Geheimtextes C den Funktionswert $y = f(M)$ zu berechnen. Er gewinnt das Spiel, wenn er diesen Wert korrekt bestimmt.

In beiden Phasen, *Select* und *Predict*, steht dem Angreifer bei SEM-CPA und SEM-CCA das Verschlüsselungsorakel $\mathcal{E}_K(\cdot)$ zur Verfügung. Bei SEM-CCA zudem in beiden

Phasen das Entschlüsselungsorakel $\mathcal{D}_K(\cdot)$. Der erhaltene Geheimtext C ist eine unzulässige Eingabe an das Entschlüsselungsorakel in den SEM-CCA-Spielen.

Im Folgenden sind die Spiele zu SEM-CPA und SEM-CCA gegeben.

SEM-CPA

Experiment $\mathbf{Exp}_{\mathcal{SE}}^{\text{sem-cpa-1}}(\mathcal{A})$	Experiment $\mathbf{Exp}_{\mathcal{SE}}^{\text{sem-cpa-0}}(\mathcal{A})$
$K \xleftarrow{\$} \mathcal{K}$	$K \xleftarrow{\$} \mathcal{K}$
$(\mathcal{M}, s) \leftarrow \mathcal{A}_{cpa}^{\mathcal{E}_K(\cdot)}(select)$	$(\mathcal{M}, s) \leftarrow \mathcal{A}_{cpa}^{\mathcal{E}_K(\cdot)}(select)$
$M \leftarrow \mathcal{M}; M' \leftarrow \mathcal{M}$	$M \leftarrow \mathcal{M}; M' \leftarrow \mathcal{M}$
$C \leftarrow \mathcal{E}_K(M)$	$C \leftarrow \mathcal{E}_K(M')$
$(f, y) \leftarrow \mathcal{A}_{cpa}^{\mathcal{E}_K(\cdot)}(predict, C, s)$	$(f, y) \leftarrow \mathcal{A}_{cpa}^{\mathcal{E}_K(\cdot)}(predict, C, s)$
return $y = f(M)$	return $y = f(M)$

SEM-CCA

Experiment $\mathbf{Exp}_{\mathcal{SE}}^{\text{sem-cca-1}}(\mathcal{A})$	Experiment $\mathbf{Exp}_{\mathcal{SE}}^{\text{sem-cca-0}}(\mathcal{A})$
$K \xleftarrow{\$} \mathcal{K}$	$K \xleftarrow{\$} \mathcal{K}$
$(\mathcal{M}, s) \leftarrow \mathcal{A}_{cpa}^{\mathcal{E}_K(\cdot), \mathcal{D}_K(\cdot)}(select)$	$(\mathcal{M}, s) \leftarrow \mathcal{A}_{cpa}^{\mathcal{E}_K(\cdot), \mathcal{D}_K(\cdot)}(select)$
$M \leftarrow \mathcal{M}; M' \leftarrow \mathcal{M}$	$M \leftarrow \mathcal{M}; M' \leftarrow \mathcal{M}$
$C \leftarrow \mathcal{E}_K(M)$	$C \leftarrow \mathcal{E}_K(M')$
$(f, y) \leftarrow \mathcal{A}_{cpa}^{\mathcal{E}_K(\cdot), \mathcal{D}_K(\cdot)}(predict, C, s)$	$(f, y) \leftarrow \mathcal{A}_{cpa}^{\mathcal{E}_K(\cdot), \mathcal{D}_K(\cdot)}(predict, C, s)$
return $y = f(M)$	return $y = f(M)$

SEM-Vorteil Wenn $b = 0$ ist, steht dem Angreifer nur die Verschlüsselung einer anderen Nachricht M' und somit keine Informationen über M, die ihm beim Ermitteln des Funktionswertes $y = f(M)$ helfen können, zur Verfügung. Semantische Sicherheit ist daher gegeben, wenn der Angreifer in beiden Szenarien, sprich $b = 0$ und $b = 1$, unabhängig von den gewählten $\mathcal{M}$ und f, gleich häufig Erfolg hat. Der Vorteil des Angreifers wird bei SEM-CPA mit $\mathsf{Adv}_{\mathcal{SE}}^{\text{sem-cpa}}(\mathcal{A})$ und bei SEM-CCA mit $\mathsf{Adv}_{\mathcal{SE}}^{\text{sem-cca}}(\mathcal{A})$ bezeichnet.

Definition 5.3 (SEM-Vorteil)**.** *Sei $\mathcal{SE} = (\mathcal{K}, \mathcal{E}, \mathcal{D})$ ein symmetrisches Kryptosystem und Angreifer $\mathcal{A}$ ein Algorithmus. Der SEM-CPA-Vorteil $\mathsf{Adv}_{\mathcal{SE}}^{\text{sem-cpa}}(\mathcal{A})$ des Angreifers $\mathcal{A}$ ist definiert als*

$$\mathsf{Adv}_{\mathcal{SE}}^{\text{sem-cpa}}(\mathcal{A}) = Pr[\boldsymbol{Exp}_{\mathcal{SE}}^{\textit{sem-cpa-1}}(\mathcal{A}) = 1] - Pr[\boldsymbol{Exp}_{\mathcal{SE}}^{\textit{sem-cpa-0}}(\mathcal{A}) = 1]$$

Der SEM-CCA-Vorteil $\mathsf{Adv}_{\mathcal{SE}}^{\text{sem-cca}}(\mathcal{A})$ des Angreifers $\mathcal{A}$ ist definiert als

$$\mathsf{Adv}_{\mathcal{SE}}^{\text{sem-cca}}(\mathcal{A}) = Pr[\boldsymbol{Exp}_{\mathcal{SE}}^{\textit{sem-cca-1}}(\mathcal{A}) = 1] - Pr[\boldsymbol{Exp}_{\mathcal{SE}}^{\textit{sem-cca-0}}(\mathcal{A}) = 1]$$

5.1.4 Beziehungen der Definitionen

In [4] wird gezeigt, dass es sowohl eine sicherheitsbewahrende Reduktion von Left-Or-Right Indistinguishability (IND) zu Real-Or-Random Indistinguishability (ROR), geschrieben IND $\Rightarrow$ ROR, als auch eine sicherheitsbewahrende Reduktion mit geringem Faktor $n = 2$ von Real-Or-Random Indistinguishability (ROR) zu Left-Or-Right Indistinguishability (IND), geschrieben ROR $\Rightarrow$ IND, existiert.

Definition 5.4 (IND $\Rightarrow$ ROR)**.** *Sei n ein Parameter, der die Ressourcenbeschränkungen eines Angreifers angibt. Für jedes symmetrische Kryptosystem $\mathcal{SE} = (\mathcal{K}, \mathcal{E}, \mathcal{D})$ gilt:*

$$\mathsf{Adv}^{\text{ror-cpa}}_{\mathcal{SE}}(n) \leq \mathsf{Adv}^{\text{ind-cpa}}_{\mathcal{SE}}(n)$$
$$\mathsf{Adv}^{\text{ror-cca}}_{\mathcal{SE}}(n) \leq \mathsf{Adv}^{\text{ind-cca}}_{\mathcal{SE}}(n)$$

Definition 5.5 (ROR $\Rightarrow$ IND)**.** *Sei n ein Parameter, der die Ressourcenbeschränkungen eines Angreifers angibt. Für jedes symmetrische Kryptosystem $\mathcal{SE} = (\mathcal{K}, \mathcal{E}, \mathcal{D})$ gilt:*

$$\mathsf{Adv}^{\text{ind-cpa}}_{\mathcal{SE}}(n) \leq 2 \cdot \mathsf{Adv}^{\text{ror-cpa}}_{\mathcal{SE}}(n)$$
$$\mathsf{Adv}^{\text{ind-cca}}_{\mathcal{SE}}(n) \leq 2 \cdot \mathsf{Adv}^{\text{ror-cca}}_{\mathcal{SE}}(n)$$

Da IND $\Rightarrow$ ROR, als auch ROR $\Rightarrow$ IND gilt, sind IND und ROR äquivalent. Das bedeutet, dass ein Kryptosystem, dessen Sicherheit für IND bewiesen ist, auch in der ROR-Definition sicher ist. Umgekehrt ist ebenso ein Kryptosystem dessen Sicherheit für ROR bewiesen ist, auch in der IND-Definition sicher.

Ebenso sind Find-Then-Guess Security und Semantische Sicherheit äquivalent. Sicherheitsbewahrende Reduktionen von Find-Then-Guess Security (FTG) zu Semantischer Sicherheit (SEM), FTG $\Rightarrow$ SEM mit geringem Faktor $n = 2$, sowie von Semantischer Sicherheit (SEM) zu Find-Then-Guess Security (FTG), SEM $\Rightarrow$ FTG, werden ebenfalls in [4] gezeigt.

Definition 5.6 (SEM $\Rightarrow$ FTG)**.** *Sei n ein Parameter, der die Ressourcenbeschränkungen eines Angreifers angibt. Für jedes symmetrische Kryptosystem $\mathcal{SE} = (\mathcal{K}, \mathcal{E}, \mathcal{D})$ gilt:*

$$\mathsf{Adv}^{\text{ftg-cpa}}_{\mathcal{SE}}(n) \leq \mathsf{Adv}^{\text{sem-cpa}}_{\mathcal{SE}}(n)$$
$$\mathsf{Adv}^{\text{ftg-cca}}_{\mathcal{SE}}(n) \leq \mathsf{Adv}^{\text{sem-cca}}_{\mathcal{SE}}(n)$$

Definition 5.7 (FTG $\Rightarrow$ SEM)**.** *Sei n ein Parameter, der die Ressourcenbeschränkungen eines Angreifers angibt. Für jedes symmetrische Kryptosystem $\mathcal{SE} = (\mathcal{K}, \mathcal{E}, \mathcal{D})$ gilt:*

$$\mathsf{Adv}^{\text{sem-cpa}}_{\mathcal{SE}}(n) \leq 2 \cdot \mathsf{Adv}^{\text{ftg-cpa}}_{\mathcal{SE}}(n)$$
$$\mathsf{Adv}^{\text{sem-cca}}_{\mathcal{SE}}(n) \leq 2 \cdot \mathsf{Adv}^{\text{ftg-cca}}_{\mathcal{SE}}(n)$$

Die Äquivalenz bedeutet auch hier, dass ein Kryptosystem, dessen Sicherheit für FTG bewiesen ist, auch in der SEM-Definition sicher ist und umgekehrt ein Kryptosystem dessen Sicherheit für SEM bewiesen ist, auch in der FTG-Definition sicher ist.

Zudem wird bewiesen, dass zwar eine sicherheitsbewahrende Reduktion von IND zu FTG (IND $\Rightarrow$ FTG) existiert, jedoch keine von FTG zu IND.

Definition 5.8 (LOR $\Rightarrow$ FTG)**.** *Sei n ein Parameter, der die Ressourcenbeschränkungen eines Angreifers angibt. Für jedes symmetrische Kryptosystem $\mathcal{SE} = (\mathcal{K}, \mathcal{E}, \mathcal{D})$ gilt:*

$$\mathsf{Adv}^{\text{ftg-cpa}}_{\mathcal{SE}}(n) \leq \mathsf{Adv}^{\text{lor-cpa}}_{\mathcal{SE}}(n)$$
$$\mathsf{Adv}^{\text{ftg-cca}}_{\mathcal{SE}}(n) \leq \mathsf{Adv}^{\text{lor-cca}}_{\mathcal{SE}}(n)$$

Daraus folgt, dass ein Kryptosystem, dessen Sicherheit für IND bewiesen ist, auch sicher in der FTG-Definition ist. Ein Kryptosystem, dessen Sicherheit für FTG bewiesen ist, muss jedoch nicht sicher in der IND-Definition sein. Die IND- und ROR-Definitionen gewährleisten somit eine höhere Sicherheit, als die FTG-Definition [4].

Da FTG und SEM äquivalent sind, bietet ein Kryptosystem, dessen Sicherheit in der IND-Definition bewiesen ist, auch stets Sicherheit in Modell der semantischen Sicherheit (SEM).

5.2 Perfekte Sicherheit

Die im Abschnitt zur semantischen Sicherheit bereits erwähnte *perfekte Sicherheit* soll kurz betrachtet werden. Dieser Abschnitt folgt [8].

Die perfekte Sicherheit ist ein von Claude Shannon entwickeltes Konzept für perfekte Geheimhaltung bei Verschlüsselungen. Es wird vorausgesetzt, dass jeder Schlüssel nur ein Mal verwendet wird. Da starke Angriffe wie Chosen-Plaintext-Angriffe und Chosen-Ciphertext-Angriffe jedoch eine mehrfache Verwendung von Schlüsseln voraussetzen, ist es im Konzept der perfekten Sicherheit ausreichend nur Ciphertext-Only-Angriffe zu berücksichtigen. Auch deterministische Verschlüsselungen sind mit dieser Einschränkung kein Sicherheitsproblem.

Definition 5.9 (Perfekte Sicherheit)**.** *Ein Kryptosystem $\mathscr{S} = (X, K, Y, e, d)$ ist perfekt sicher, wenn die Auftrittswahrscheinlichkeit eines Klartextes und eines Geheimtextes unabhängig sind. Für alle Klartexte $x \in X$ und alle Geheimtexte $y \in Y$ gilt:*

$$Pr[x|y] = Pr[x]$$

Daraus folgt: Liegen zwei Klartexte x_1, x_2 und ein Geheimtext y vor, dann ist

$$Pr[x_1|y] = Pr[x_2|y]$$

Somit gewinnt ein Angreifer aus einem Geheimtext keine Informationen über den Klartext.

Die perfekte Sicherheit ist für den praktischen Einsatz jedoch eher ungeeignet. Schlüssel werden in der Praxis selten nur ein einziges Mal verwendet, sondern mehrfach. Die bloß einmalige Nutzung eines Schlüssels ist aber Voraussetzung im Konzept der perfekten Sicherheit.

Daher wurden Konzepte wie die bereits beschriebene semantische Sicherheit entwickelt. Diese sind zwar eine Abschwächung der perfekten Sicherheit, da es genügt, wenn ein Angreifer die Auftrittswahrscheinlichkeiten von Geheimtexten mit realistisch beschränkten Ressourcen nicht unterscheiden können. Im Gegensatz zur perfekten Sicherheit sind diese aber auch anwendbar auf Szenarien, in denen Schlüssel mehrfach verwendet werden.

6 Zusammenfassung und Fazit

Zusammenfassung. Zum Abschluss sollen die Themen, die in dieser Arbeit betrachtet worden sind, noch einmal kurz zusammengefasst werden.

In der Kryptographie wird zwischen symmetrischen und asymmetrischen Verschlüsselungen unterschieden. Diese Verschlüsselungs-, sowie die zugehörigen Entschlüsselungsfunktionen, werden in Kryptosystemen zusammengefasst und formalisiert. Für Kryptosysteme sind verschiedene Sicherheitsziele und -merkmale definiert. Das zentrale Thema dieser Arbeit war das Sicherheitsmodell der Ununterscheidbarkeit von Geheimtexten.

Im Rahmen der Ununterscheidbarkeit wurde vor allem die „Left-Or-Right-Indistinguishability“, kurz IND, und die Sicherheit dieser gegen Chosen-Plaintext-Angriffe (IND-CPA) und Chosen-Ciphertext-Angriffe (IND-CCA) betrachtet. Dafür wurde ein Beweismodell mittels einer Sequenz von Spielen und festgelegten Übergängen zwischen diesen Spielen vorgestellt. Für sowohl symmetrische, als auch asymmetrische Kryptosysteme wurden schließlich Spiele für einen Chosen-Plaintext-Angriff oder einen Chosen-Ciphertext-Angriff gegen die Sicherheit im Sinne der Ununterscheidbarkeit definiert.

Bei symmetrischen Kryptosysteme wurden der AES-Algorithmus und verschiedene Betriebsmodi, in denen Blockchiffren wie AES bei Nachrichten, die ihre festgelegte Blockgröße überschreiten, arbeiten können, vorgestellt. Mittels der für symmetrische Kryptosysteme definierten Spiele wurde gezeigt, dass die Betriebsmodi ECB und CBC-C auch bei Verwendung einer sicheren Blockchiffre wie AES, nicht sicher im Sinne der Ununterscheidbarkeit gegen Chosen-Plaintext-Angriffe sind. Außerdem wurde die Sicherheit im Sinne der Ununterscheidbarkeit gegen Chosen-Ciphertext-Angriffe des CBC-Mode widerlegt. Die bewiesenen Unsicherheiten zeigten, dass eine eigentlich sichere Blockchiffre wie AES bei einigen Betriebsmodi unsicher wird und die Wahl des Betriebsmodus daher von hoher Bedeutung ist.

Bei asymmetrischen Kryptosystemen wurden das RSA- und das ElGamal-Kryptosystem vorgestellt. Mittels der für asymmetrische Kryptosysteme vorgestellten Spiele wurde gezeigt, dass das RSA-Kryptosystem auf Grund von deterministischer Verschlüsselung keine Sicherheit im Sinne der Ununterscheidbarkeit bietet. Mit dem RSA-OAEP Verfahren wurde eine Modifikation des RSA-Kryptosystems vorgestellt, die Sicherheit im Sinne der Ununterscheidbarkeit auch bei adaptiv-gewählten Chosen-Ciphertext-Angriffen (IND-CCA2) bietet.

Mittels einer Sequenz von Spielen wurde bewiesen, dass das ElGamal-Kryptosystem Sicherheit im Sinne der Ununterscheidbarkeit gegen Chosen-Plaintext-Angriffe bieten kann, die Sicherheit jedoch von der gewählten Gruppe $\mathcal{G}$ abhängt.

Darüber hinaus wurden mit der Real-Or-Random-Indistinguishability, der First-Then-Guess -Security und der semantischen Sicherheit weitere Definitionen der Ununterscheidbarkeit, die neben der Left-Or-Right-Indistinguishability existieren, kurz vorgestellt und die Beziehungen dieser untereinander betrachtet. Schließlich wurde in einem kurzen Ausblick das Konzept der perfekten Sicherheit betrachtet.

Fazit. Die in der Arbeit vorgestellte Ununterscheidbarkeit und die semantische Sicherheit sind Sicherheitskonzepte, die die perfekte Sicherheit nicht erreichen. In einem realistischen Szenario ist die perfekte Sicherheit jedoch auf Grund der nur einmal verwendbaren Schlüssel oftmals nicht realisierbar oder auch gar nicht nötig. Die Ressourcen, die einem Angreifer tatsächlich zur Verfügung stehen, sind in der Realität niemals uneingeschränkt. Daher ist die Sicherheit, die die Konzepte der Ununterscheidbarkeit und der semantischen Sicherheit gewährleisten, in realistischen Szenarien absolut ausreichend.

Das Sicherheitsmodell der Ununterscheidbarkeit von Geheimtexten impliziert, wie in der Arbeit beschrieben wurde, semantische Sicherheit. Ununterscheidbarkeit bietet zudem mit der Möglichkeit des Beweises oder der Widerlegung der Sicherheit mittels der Spiele und Sequenzen von Spielen eine im Vergleich zu anderen kryptographischen Methoden simple Möglichkeit die Sicherheit für ein Kryptosystem zu beweisen oder zu widerlegen.

Ein Thema, das zwar im Laufe dieser Arbeit erwähnt, aber nicht weiter betrachtet worden ist, sind *Authenticated-Encryption Modes*. Diese Betriebsmodi für Blockchiffren nutzen *Authenticated Encryption* (kurz „AE“) und können, wie in Kapitel 4.5.5 bereits erwähnt, im Gegensatz zu den in dieser Arbeit vorgestellten, klassischen Betriebsmodi, Sicherheit im Sinne der Ununterscheidbarkeit auch gegen adaptiv-gewählte Chosen-Ciphertext-Angriffe (IND-CCA2) bieten [14].

Ein anderes Thema, das in dieser Arbeit nicht behandelt wurde, sind das *Legendre-Symbol* und das *Jacobi-Symbol*. Mit Hilfe dieser können z. B. im ElGamal-Kryptosystem bei bestimmten Gruppen $\mathcal{G}$ Informationen über den Klartext aus dem Geheimtext gewonnen werden. Die Sicherheit des ElGamal-Kryptosystems ist somit bei diesen Gruppen nicht mehr gegeben [14].

Ein weiteres, nicht in dieser Arbeit betrachtetes Thema ist das Konzept der *Non-Malleability*, kurz „NM“. Dieses besagt, dass es einem Angreifer nicht möglich sein darf, aus einen gegebenen Geheimtext y einen anderen Geheimtext y' zu konstruieren, sodass die zugehörigen Klartexte x und x' ähnlich zu einander sind. Eine solche Ähnlichkeit von Klartexten kann z. B. ein einfacher mathematischer Zusammenhang wie $x' = x + 1$ sein [5].

Literaturverzeichnis

[1] Mihir Bellare and Phillip Rogaway. “Code-Based Game-Playing Proofs and the Security of Triple Encryption”. In: *IACR Cryptology ePrint Archive* 2004 (2004), p. 331. URL: http://eprint.iacr.org/2004/331.

[2] Mihir Bellare and Phillip Rogaway. “Introduction to modern cryptography”. In: *Ucsd Cse* 207 (2005), p. 207.

[3] Mihir Bellare and Phillip Rogaway. “The game-playing technique”. In: *International Association for Cryptographic Research (IACR) ePrint Archive: Report* 331 (2004), p. 2004.

[4] Mihir Bellare et al. “A Concrete Security Treatment of Symmetric Encryption”. In: *38th Annual Symposium on Foundations of Computer Science, FOCS '97, Miami Beach, Florida, USA, October 19-22, 1997*. 1997, pp. 394–403. DOI: 10.1109/SFCS.1997.646128. URL: https://doi.org/10.1109/SFCS.1997.646128.

[5] Mihir Bellare et al. “Relations among notions of security for public-key encryption schemes”. In: *Annual International Cryptology Conference*. Springer. 1998, pp. 26–45.

[6] Albrecht Beutelspacher, Heike Neumann, and Thomas Schwarzpaul. *Kryptografie in Theorie und Praxis - mathematische Grundlagen für elektronisches Geld, Internetsicherheit und Mobilfunk*. Vieweg, 2005. ISBN: 978-3-528-03168-8.

[7] Dobre Blazhevski et al. “Modes of operation of the AES algorithm”. In: (2013).

[8] Johannes A. Buchmann. *Einführung in die Kryptographie, 6. Auflage*. Springer, 2016. ISBN: 978-3-642-39774-5. DOI: 10.1007/978-3-642-39775-2. URL: https://doi.org/10.1007/978-3-642-39775-2.

[9] Hans Delfs and Helmut Knebl. *Introduction to Cryptography - Principles and Applications*. Information Security and Cryptography. Springer, 2007. ISBN: 978-3-540-49243-6. DOI: 10.1007/3-540-49244-5. URL: https://doi.org/10.1007/3-540-49244-5.

[10] Alexander W Dent. “A Note On Game-Hopping Proofs.” In: *IACR Cryptology ePrint Archive* 2006 (2006), p. 260.

[11] Wolfgang Ertel. *Angewandte Kryptographie*. Fachbuchverlag Leipzig im Carl Hanser Verlag, 2003.

[12] Eiichiro Fujisaki et al. “RSA-OAEP is secure under the RSA assumption”. In: *Annual International Cryptology Conference*. Springer. 2001, pp. 260–274.

[13] Oded Goldreich. *The Foundations of Cryptography - Volume 1: Basic Techniques*. Cambridge University Press, 2001. ISBN: 0-521-79172-3. DOI: 10.1017/CBO9780511546891. URL: http://www.wisdom.weizmann.ac.il/%5C%7Eoded/foc-vol1.html.

[14] Ralf Küsters and Thomas Wilke. *Moderne Kryptographie - Eine Einführung*. Viweg + Teubner, 2011. ISBN: 978-3-519-00509-4.

[15] Phillip Rogaway. "Evaluation of some blockcipher modes of operation". In: *Cryptography Research and Evaluation Committees (CRYPTREC) for the Government of Japan* (2011).

[16] Victor Shoup. "Sequences of games: a tool for taming complexity in security proofs". In: *IACR Cryptology ePrint Archive* 2004 (2004), p. 332. URL: http://eprint.iacr.org/2004/332.

[17] Jochen Ziegenbalg. *Elementare Zahlentheorie*. Springer, 2002. ISBN: 978-3-658-07170-7.