Sicherheitsmodelle für das Ajtai-Dwork-Kryptosystem

Diplomarbeit am Fachbereich Mathematik von

Sebastian Pape

vorgelegt bei

Prof. Dr. Johannes Buchmann

Fachgebiet Theoretische Informatik – Kryptographie und Computeralgebra
Fachbereich Informatik
Technische Universität Darmstadt

Januar 2004

Vorwort

Mein Dank gilt allen, die bei der Entstehung und Fertigstellung dieser Diplomarbeit beteiligt waren.

Professor Dr. Johannes Buchmann danke ich für die Hilfe bei der Wahl des Themas und die Vergabe dieser interessanten Diplomarbeit.
Besonderer Dank gilt Christoph Ludwig für die hervorragende Betreuung, die sich insbesondere durch zahlreiche Diskussionen, Anregungen und nicht zuletzt Kritik auszeichnete.
Außerdem danke ich Ralf Bendel, Laura Dietz, Björn Egner und Tilman Laschinger für ihre zahlreichen Anmerkungen und Korrekturen.
Mein größter Dank gilt jedoch meinen Eltern, die mich während meines gesamten Studiums in vielerlei Hinsicht unterstützt haben.

Für verbliebene Fehler und sonstige Versehen bitte ich den Leser um Nachsicht und freue mich auf konstruktive Kritik.

Darmstadt, im Januar 2004 *Sebastian Pape*

Inhaltsverzeichnis

Kapitel 1

Einleitung

Mit der Verbreitung von Computern und Netzwerken nimmt die Notwendigkeit von sicheren Public-Key-Verschlüsselungen und digitalen Signaturen immer schneller zu. Dabei beruht die Sicherheit der meisten Kryptosysteme auf der Tatsache, dass es bis heute keinen klassischen Algorithmus gibt, der in der Lage ist, in polynomieller Zeit zu faktorisieren oder diskrete Logarithmen zu berechnen. Ein ernstzunehmendes Problem stellt allerdings die Entwicklung von Quantencomputern (siehe z.B. [CN00]) in absehbarer Zeit dar – z. B. auf Basis von kernmagnetischer Resonanz (NMR) (siehe [VSB+01]). Shor gab Algorithmen auf Quantencomputern an, die in Polynomialzeit sowohl faktorisieren, als auch diskrete Logarithmen berechnen können ([Sho97]). Ein besonderes Augenmerk muss deswegen darauf liegen, möglichst viele verschiedene „schwere Probleme“ zu finden, auf deren Schwierigkeit sich Kryptosysteme aufsetzen lassen.
Ein Ansatz dafür ist, Kryptosysteme basierend auf schweren Gitterproblemen zu entwerfen. Gitter sind diskrete Untergruppen des euklidischen Raumes und haben mittlerweile in vielen Bereichen der Mathematik und Informatik ein breites Anwendungsgebiet. War ihre Hauptverwendung in der Kryptographie anfangs beim Brechen von Kryptosystemen zu finden (z. B. um Kryptosysteme, die auf dem Rucksack-Problem basieren, zu brechen – wie in [Odl91] beschrieben), so führten einige komplexitätstheoretische Entdeckungen Ajtais ([Ajt96] und [Ajt98]) zu einem wahren Schwung an neuen Erkenntnissen über die Komplexität von Gitterproblemen, über die Cai in [Cai99] und [Cai00] einen Überblick gibt. Ajtai konnte eine Verbindung zwischen der Worst-Case-Komplexität und der Average-Case-Komplexität von bekannten Gitterproblemen herstellen. Basierend auf dieser Entdeckung entwickelte er erstmals ein Public-Key-Kryptosystem, bei dem eine zufällig gewählte Instanz zu brechen genauso schwer ist, wie eine Worst-Case-Instanz des zugrundeliegenden Problems zu lösen ([AD97]). Genauer gesagt zeigte er, dass ein Angreifer, der für eine zufällige Instanz des Kryptosystems – in

polynomieller Zeit – Verschlüsselungen von Null und Eins[1] mit einer Wahrscheinlichkeit von mindestens $\frac{1}{2}$ auseinander halten kann, in der Lage ist, das Worst-Case n^8-unique Shortest-Vector-Problem zu lösen.
Ein Sicherheitsproblem für das Ajtai-Dwork-Kryptosystem stellt jedoch die stetige Verbesserung der Gitterbasisreduktionsalgorithmen und die Verbesserung der Approximationen des Shortest-Vector-Problems dar ([NS98], [NS99]). Das Ziel von Gitterreduktionen ist es, eine interessante Darstellung für Gitter zu finden. Diese besteht aus einer Basis mit kurzen Vektoren, die fast orthogonal zueinander sind. Die Reduktionstheorie von Gittern reicht zurück zur Reduktionstheorie quadratischer Formen – u. a. entwickelt durch Lagrange ([Lag73]), Gauss ([Gau01]), Hermite ([Her50]), Korkin und Solotarew ([KS72], [KS73]) – sowie zu Minkowski, der in seiner Geometrie der Zahlen ([Min96]) fundamentale zahlentheoretische Probleme in geometrische transformierte und sie so lösen konnte. Vor dem Hintergrund der algorithmischen Zahlentheorie lebte dieser Forschungszweig um 1980 wieder auf. In der Folge gaben Lenstra, Lenstra und Lovász einen Algorithmus an, der Gitterbasen effizient reduziert und gleichzeitig eine Approximation des Shortest-Vector-Problems berechnet ([LLL82]). Durch den LLL-Algorithmus wurde der Reduktionstheorie von Gittern neue Aufmerksamkeit zuteil. Im folgenden wurde der LLL-Algorithmus – hauptsächlich von Schnorr ([Sch87], [Sch88], [SE91], [Sch03]) – weiter verbessert und auch die Theorie der Quantencomputer wurde für die Approximation von kürzesten Vektoren berücksichtigt ([Reg02], [Lud03]).
Regev entwickelte jedoch ein Public-Key-Kryptosystem, für das er die Sicherheit auf die Worst-Case-Schwierigkeit des $n^{1,5}$-unique Shortest-Vector-Problems reduzieren konnte ([Reg03b]). Im Gegensatz zum Ajtai-Dwork-Kryptosystem reicht diese Sicherheit möglicherweise aus, um gegenüber Approximationen des Shortest-Vector-Problems bestehen zu können.

Da Ajtai und Dwork sich in ihrer Arbeit auf den Reduktionsbeweis beschränkt hatten, stellt sich die Frage, in welchen Sicherheitsmodellen ihr System als sicher gelten kann. Sicherheitsmodelle treffen Annahmen über Absichten und Fähigkeiten eines potentiellen Angreifers und bieten eine formale Grundlage, um die Sicherheit eines Kryptosystems beurteilen zu können. Das Ziel dieser Diplomarbeit ist es, sich einen Überblick über verschiedene Sicherheitsmodelle zu verschaffen und die geeigneten darauf zu untersuchen, ob sie vom Ajtai-Dwork-Kryptosystem erfüllt werden. In den Fällen, in denen dies nicht der Fall ist, soll festgestellt werden, ob das Kryptosystem durch einfache Modifikationen „repariert“ werden kann.

Der Aufbau der Diplomarbeit ist wie folgt:
In Kapitel 2 soll eine Einführung in die Komplexitätstheorie gegeben werden,

[1] Die Verschlüsselung im Ajtai-Dwork-Kryptosystem erfolgt bitweise.

in der weniger formale Definitionen und Abgrenzungen der Komplexitätsklassen im Vordergrund stehen. Vielmehr soll eine Anschauung davon vermittelt werden, welche Probleme (effizient) berechenbar sind. Kapitel 3 führt in die Grundlagen der Kryptographie ein und geht dabei auf verschiedene Absichten und Fähigkeiten potentieller Angreifer ein, welche abschließend in verschiedene formale Definitionen von Sicherheitsmodellen gefasst werden. Eine kurze Einführung in die Gittertheorie sowie eine Schilderung bekannter Gitterprobleme gibt Kapitel 4. Daran anschließend folgt im 5. Kapitel eine detaillierte Beschreibung der Varianten des Ajtai-Dwork-Kryptosystems sowie eine kurze Beschreibung des alternativen Kryptosystems von Regev, das ebenfalls auf der Worst-Case-Schwierigkeit des unique Shortest-Vector-Problems beruht. Hauptsächlich auf den vorangegangenen Kapiteln basierend werden in Kapitel 6 Angriffe auf die verschiedenen Varianten des Ajtai-Dwork-Kryptosystems beschrieben. Den Abschluss der Arbeit bildet die Zusammenfassung und ein kurzer Ausblick in Kapitel 7.

Kapitel 2

Grundlagen der Komplexitätstheorie

In diesem Kapitel wollen wir eine intuitive Vorstellung davon gewinnen, welche Algorithmen effizient berechenbar sind. Dabei soll es weniger darum gehen, bereits bekannte Definitionen zu betrachten, die sich auch in [HU79], [Weg93] und [Sch97] finden, sondern darum, am Ende des Kapitels eine Idee davon zu haben, was in der Praxis durchführbar ist und was eher nicht. Dazu werfen wir zuerst einen kurzen Blick auf die Fragestellung, was überhaupt berechenbar ist, bevor wir die Klasse der berechenbaren Probleme näher betrachten. Den Abschluss des Kapitels bildet dann eine Reihe von Definitionen und Notationen, die wir in den folgenden Kapitel benötigen werden.

2.1 Berechenbarkeit

Beschäftigt man sich ausführlicher mit Rechnern, so stößt man unweigerlich auf die Frage, welche Probleme sich überhaupt automatisch lösen lassen. Die naheliegendste Vorstellung *intuitiver Berechenbarkeit* ist, dass Rechner die Probleme lösen können, für die es einen Algorithmus gibt, der nach endlich vielen Schritten hält.
Nachdem GÖDEL Anfang der 30er Jahre seine Arbeit über die Unvollständigkeit der Mathematik ([Göd31]) veröffentlich hatte, schuf TURING ein simples Modell von Maschine, das er *Universal Machine* nannte ([Tur36]) und später nach ihm *Turing-Maschine* genannt wurde. Zur selben Zeit entwickelte CHURCH mit dem *Lambda-Kalkül* einen mathematischen Formalismus, der Funktionen beschreibt ([Chu36]) und KLEENE arbeitete eine Klasse mathematische Objekte aus, die sogenannten *rekursiven Funktionen* ([Kle35], [Kle36a]). Etwas später definierte POST einen Mechanismus, der Symbole manipuliert und nannte ihn *Produktionssystem* ([Pos43]). Gemeinsam konnten sie schließlich zeigen, dass diese Formalismen gleichwertig sind, d. h. dass dieselbe Klasse von Problemen damit lösbar ist ([Kle36b], [Pos36], [Tur37]). Diese Äquivalenz bringt die unabhängig von CHURCH und TURING entwickelte *Church-Turing-These* zum Ausdruck. Eine Version dieser mittlerweile allgemein akzeptierten Vermutung ist, dass die Klasse der intuitiv berechenbaren Probleme identisch mit der Klasse der Turing-berechenbaren Probleme ist. Ein Indiz für die Richtigkeit dieser These ist, dass auch für alle später entwickelten Formalismen die Gleichwertigkeit zu den bereits aufgeführten gezeigt werden konnte.
Dieser historische Überblick soll uns soweit als Einstieg genügen. Ohne uns in Details verlieren zu wollen, halten wir fest, dass der intuitive Begriff des Berechenbaren sich formal fassen lässt. Für uns interessant sind in erster Linie die Probleme, die von Rechnern gelöst werden können, d. h. genau die eben angesprochene Problemklasse. Schließlich wollen wir Kryptographie auf Rechnern betreiben. Wir bemerken jedoch, dass theoretisch Berechenbares in der Praxis trotzdem nicht erreichbar sein kann. Ein Algorithmus, der zwar die richtige Lösung ausgibt, dafür aber „zu lange“ braucht, wird uns in der Praxis nicht besonders nützlich sein. Es ist also notwendig, die Problemklasse der berechenbaren Probleme detaillierter zu untersuchen, was wir im nächsten Kapitel tun wollen.

2.2 Komplexität

Das Modell der Turing-Maschinen beschreibt klassische Rechner. Wie wir in der Einleitung bereits erwähnt haben, gibt es auch Rechner, die von den Gesetzen der Quantenmechanik Gebrauch machen – die *Quantencomputer*. Leider können auch Quantenrechner den Begriff der Berechenbarkeit nicht

erweitern. Sie lassen sich auf klassischen Rechner simulieren – wenn auch nicht sonderlich effizient. Wie Quantencomputer genau funktionieren, würde den Rahmen dieser Arbeit sprengen, weswegen wir es bei einem Verweis auf [CN00] und [Fen03] belassen wollen. Vorerst lassen wir sie außen vor und wenden uns klassischen Rechnern zu.
Um die praktische Relevanz eines Algorithmus zu beurteilen, gibt es hauptsächlich zwei Ressourcen, die wir betrachten müssen: Zeit und Speicherplatz. Komplexitätstheorie beschäftigt sich damit, die minimale Zeit und den minimalen Speicherplatz von Algorithmen zu erfassen, die ein Problem lösen. Wir wollen uns dabei auf Worst-Case-Analysen konzentrieren, d. h. wir wollen die Rechenzeit und den Speicherverbrauch erfassen, mit denen sich *jede Instanz* eines Problems lösen lässt. Der wichtigste Parameter für diese Beurteilung ist die *Größe der Probleminstanz*, mit der wir die Länge einer natürlichen Darstellung des Problems in Bits meinen. Sollen wir bspw. eine Datenbank sortieren, deren n Einträge jeweils m Bits haben, so ist die Größe der Probleminstanz $n \cdot m$.
Aus der Vielzahl der Komplexitätsklassen (siehe z. B. [Aar03]) wollen wir uns die gängigsten etwas genauer ansehen, dabei beginen wir mit den klassichen Modellen.

2.2.1 P

Probleme der Problemklasse P (auch $PTIME$ genannt) kann ein Algorithmus auf einem klassischen Rechner in *Polynomialzeit* lösen. In Polynomialzeit heißt, dass der Algorithmus jede Instanz des Problems in einer Zeit löst, die durch die Größe der Instanz polynomiell beschränkt ist. Es ist offensichtlich, dass ein Algorithmus, der in Polynomialzeit läuft auch nur polynomiell viel Speicher benutzen kann.
Ein Beispiel für ein Problem dieser Klasse sind Sortierprobleme.
Wir werden in späteren Kapiteln Algorithmen betrachten, die sich aus mehreren Stufen[1] zusammensetzen. Ein Algorithmus, der sich aus mehreren Stufen zusammensetzt, liegt in P, wenn jede seiner Stufen in P liegt.

2.2.2 PSPACE

Algorithmen, die ein Problem auf einem klassischen Rechner mit von der Probleminstanz polynomiell begrenzten Speicherplatz lösen können, liegen in der Klasse $PSPACE$. Die Zeit, die der Algorithmus dazu benötigen darf, wird nicht eingeschränkt. Offensichtlich gilt $P \subseteq PSPACE$, es ist jedoch noch nicht bekannt, ob P eine echte Teilmenge von $PSPACE$ ist.

[1] maximal jedoch polynomiell viele Stufen

2.2.3 NP

Eine weitere wichtige Problemklasse ist NP, die Klasse der nicht deterministischen, polynomiell zeitbegrenzten Algorithmen. Ein typisches Problem dieser Klasse ist, für einen boolschen Ausdruck zu entscheiden, ob es eine Belegung gibt, so dass dieser wahr wird (*Satisfiability* oder abgekürzt SAT genannt). Bis heute ist kein Polynomialzeit-Algorithmus für dieses Problem bekannt. Allerdings ist es möglich, für jede Belegung in Polynomialzeit zu überprüfen, ob sie wahr ist. Für jede Lösung existiert also ein Zertifikat, das sich in Polynomialzeit überprüfen lässt. Das heißt, falls wir – nicht deterministisch – eine richtige Lösung geraten haben, so sind wir in Polynomialzeit in der Lage, ihre Richtigkeit zu verifizieren. Betrachtet man für SAT den vollständigen Entscheidungsbaum, so können wir uns Nicht-Determinismus so vorstellen, dass der Algorithmus einen richtigen Weg von der Wurzel bis zu einem Blatt „weiß“, falls es diesen Weg gibt. Das heißt, der Algorithmus wählt orakelhaft an jedem Knoten die richtige Kante.
Mittlerweile wurde gezeigt, dass $P \subseteq NP \subseteq PSPACE$ gilt. Auch hier ist nicht bekannt, ob es sich um echte Teilmengen handelt. Die Frage, ob $P \neq NP$ gilt, was allgemein angenommen wird, ist eine *der* offenen Fragen der theoretischen Informatik.

Eine besondere Teilmenge der Probleme aus NP sind die *NP-harten* Probleme. Sie sind mindestens so schwer wie alle anderen Probleme in NP. Mindestens so schwer heißt, dass es einen Algorithmus aus P gibt, der einen Algorithmus für die Lösung eines NP-harten Problems in die Lösung eines beliebigen anderen Problems aus NP umformen kann. Diese Umformung wollen wir mit *polynomieller Reduktion* bezeichnen. Auf die mächtigere *randomisierte Reduktion* wollen wir hier nicht eingehen, eine Definition findet sich bspw. in [CR90].

NP-vollständige Probleme (NPC) sind NP-harte Probleme, die zusätzlich noch in NP liegen. Sie haben gemeinsam, dass sie Lösungen in exponentieller Zeit besitzen, aber bislang weder gezeigt werden konnte, dass es einen Algorithmus in Polynomialzeit gibt, der sie löst, noch dass es diesen Algorithmus nicht geben kann. Für das SAT-Problem konnte 1971 Cook die NP-Vollständigkeit direkt nachweisen ([Coo71]) – für alle anderen NP-Vollständigkeitsnachweise genügt es, die Reduktionen *auf* ein anderes und *von* einem anderen NP-vollständigen Problem zu zeigen.

2.2.4 BPP

Ein Entscheidungsproblem ist ein Problem, bei dem es zu gegebener Eingabe nur zwei mögliche Ausgaben gibt: 0 oder 1 bzw. *ja* oder *nein*. BPP[2] ist die

[2] [engl.] Bounded-error Probabilistic Polynomial-time

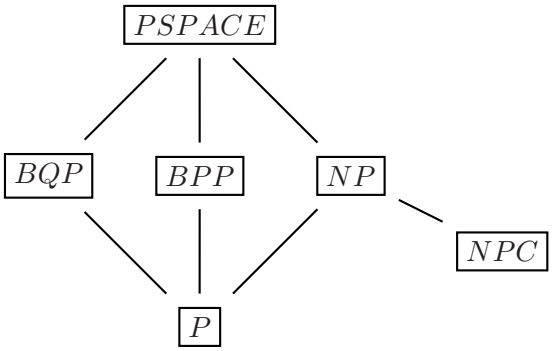

Abbildung 2.1: Inklusionsbeziehungen der gängigsten Komplexitätsklassen

Klasse der *Entscheidungsprobleme*, für die klassische Algorithmen mit Hilfe eines Zufallsgenerators die richtige Antwort in polynomiell beschränkter Zeit mindestens mit einer Wahrscheinlichkeit von $\frac{3}{4}$ finden. In der Literatur finden sich auch ähnliche Definitionen mit anderen Wahrscheinlichkeiten. In der Tat ist $\frac{3}{4}$ willkürlich gewählt. Jede Definition mit einer Wahrscheinlichkeit, die echt größer als $\frac{1}{2}$ ist, beschreibt dieselbe Problemklasse. Die Idee bei der Benutzung von Algorithmen dieser Klasse ist, den Algorithmus mehrfach mit derselben Eingabe laufen zu lassen und das Ergebnis durch Mehrheitsentscheid der Durchgänge zu bestimmen.
Offensichtlich gilt $P \subseteq BPP$, die Beziehung von BPP zu NP ist bisher unbekannt.

2.2.5 BQP

Nachdem wir uns die wichtigsten klassischen Komplexitätsklassen angesehen haben, wollen wir diese Auflistung mit einer Komplexitätsklasse für Quantenrechner beenden.
BQP[3] ist die Klasse der Entscheidungsprobleme, die ein Algorithmus auf einem Quantencomputer in polynomieller Laufzeit mit einer Wahrscheinlichkeit von mindestens $\frac{3}{4}$ richtig löst. Sie ist das Analogon zur Klasse BPP auf klassischen Rechnern. Da Quantencomputer durch ihre Konstruktion unabänderlich probabilistisch sind, existiert kein Analogon zu P für Quantencomputer.
Wie wir bereits in der Einleitung erwähnten, sind Faktorisierung und Ziehen diskreter Logarithmen Probleme aus BQP ([Sho97]). Es ist bekannt, dass $P \subseteq BQP \subseteq PSPACE$ gilt, wie sich BQP jedoch zu BPP und NP verhält ist nicht bekannt.

[3] [engl.] Bounded-error Quantum Polynomial-time

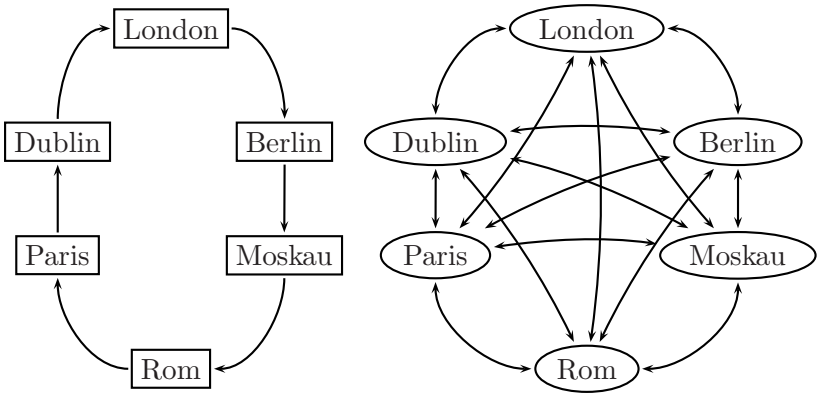

Abbildung 2.2: Zwei Instanzen des Travelling-Salesman-Problems

2.2.6 Fazit

Definition 2.1. *Als Klasse der* effizient *oder* mit vertretbarem Aufwand berechenbaren Probleme auf klassischen Rechnern *wollen wir Probleme ansehen, die in BPP liegen.*

Definition 2.2. *Als Klasse der* effizient *oder* mit vertretbarem Aufwand berechenbaren Probleme auf Quantenrechnern *wollen wir Probleme ansehen, die in BQP liegen.*

Bemerkung 2.3. Wir bemerken ausdrücklich, dass dieser Klasseneinteilung Worst-Case-Analysen zu Grunde liegen und diese nur bedingt auf die Praxis übertragbar sind. Ein Problem, dass im Worst-Case schwierig lösbar ist, muss nicht in jedem Fall schwer zu lösen sein. Betrachten wir dazu als Beispiel die schematische Darstellung zweier Instanzen des *Travelling-Salesman-Problems* (*TSP*) in Abbildung 2.2. Beim Travelling-Salesman-Problem will ein Handlungsreisender eine Rundreise durch n Städte unternehmen. Die Entfernung der Städte zueinander wird durch eine $n \times n$ Matrix gegeben und es soll entschieden werden, ob es eine Rundreise gibt, deren Länge kleiner als k ist. Sind die Städte – ähnlich Perlen auf einer Kette – so angeordnet, dass es für die Rundreise nur eine Möglichkeit gibt, so addiert man alle Weglängen und kann einfach entscheiden, ob deren Summe kleiner als k ist. Gibt es hingegen von jeder Stadt zu jeder anderen einen direkten Weg, so kann dieses Entscheidungsproblem durchaus recht schwierig sein.
Es können also für Probleme mit einer schlechten Worst-Case-Laufzeit durchaus Algorithmen existieren, die im Average-Case polynomielle Laufzeit haben, da die Instanzen, die eine hohe Worst-Case-Laufzeit „verursachen“ selten auftreten oder pathologische Instanzen des jeweiligen Algorithmus sind.

Auch können Algorithmen mit polynomieller Laufzeit für die in der Praxis interessanten Instanzen durchaus länger laufen als Algorithmen mit schlechterer Worst-Case-Laufzeit. So wird bspw. bei der Lösung linearer Optimierungsprobleme häufig der Simplex-Algorithmus von DANTZIG ([Dan63])verwendet, der eine exponentielle Worst-Case-Laufzeit aufweist, obwohl mittlerweile ein Algorithmus von KHACHIYAN mit polynomieller Laufzeit bekannt ist ([Kha79]).

2.3 Definitionen und Notationen

Zum Abschluss dieses Kapitels wollen wir noch einige Definitionen und Notationen angeben, die wir im Verlauf dieser Arbeit brauchen werden.

Definition 2.4. *Als* Orakel *wollen wir eine Black-Box bezeichnen, die ein bestimmtes Problem löst.* Black-Box *heißt dabei, dass genau definiert ist, welche Ausgabe bei welcher Eingabe erfolgt, wir aber nicht wissen was innerhalb der Black-Box passiert oder es für unsere Betrachtungen unwesentlich ist. Orakel wollen wir im Folgenden mit $\mathcal{O}$ notieren, wobei wir zur Unterscheidung der Orakel oder zum Hinweis auf das zu Grunde liegende Problem Indizes verwenden.*

Definition 2.5. *Für einen* Algorithmus $A(\cdot)$ *mit* Eingabe x *und* Ausgabe y *schreiben wir* $y \leftarrow A(x)$. *Hat der Algorithmus Zugriff auf ein Orakel, d. h. darf er für seine Berechnungen Anfragen an ein Orakel $\mathcal{O}$ stellen, so notieren wir dies mit* $y \leftarrow A^{\mathcal{O}}(x)$. *Wollen wir hervorheben, dass es sich bei dem Algorithmus A um einen probabilistischen Algorithmus handelt, so schreiben wir auch* $y \xleftarrow{R} A(x)$. *Mit* probabilistischen Algorithmen *meinen wir klassische Algorithmen, die Zugriff auf einen Zufallsgenerator haben, d. h. mit* $r \in \{0,1\}^*$ *berechnen sie* $y \leftarrow A(x,r)$.
Bei probabilistischen Funktionen fassen wir $f(x)$ als Abkürzung von $f(x,r)$ auf, wenn der Zufallsparameter r für die Betrachtungen unwesentlich ist oder aus dem Zusammenhang klar hervorgeht.

Definition 2.6. *Sei $f(\cdot)$ eine Funktion, $A(\cdot)$ ein Algorithmus und X eine Menge, dann definieren wir* $A(X) \stackrel{def}{=} \{y \mid \exists x \in X \text{ mit } y = A(x)\}$ *bzw.* $f(X) \stackrel{def}{=} \{y \mid \exists x \in X \text{ mit } y = f(x)\}$ *als das* Bild *von X.*

Definition 2.7. *Mit $Pr[E]$ wollen wir die* Wahrscheinlichkeit *bezeichnen, mit der das Ereignis E eintritt. Die* bedingte Wahrscheinlichkeit *von E_1 unter der Bedingung E_2 notieren wir mit $Pr[E_1|E_2]$.*

Definition 2.8. *Eine Funktion* $f(\cdot) : \mathbb{R}^+ \to \mathbb{R}$ *ist* vernachlässigbar, *wenn ihr Betrag schneller als der Kehrwert jedes Polynoms gegen 0 konvergiert. Das heißt, für jede Konstante* $c \in \mathbb{R}^+$ *existiert ein* n_c, *so dass* $|f(n)| \leq n^{-c}$ *für alle* $n > n_c$ *gilt.*

Bemerkung 2.9. Die Idee vernachlässigbarer Funktionen ist, formal angeben zu können, welche Funktionen (asymptotisch) unbedeutend sind. Dies ist in der Kryptographie meistens die Erfolgschance des Angreifers (sein Ziel zu erreichen), die als Funktion in Abhängigkeit des Sicherheitsparameters des Kryptosystems gemessen wird. Eine ausführliche Abhandlung über vernachlässigbare Funktionen und ihre Bedeutung in der Kryptographie gibt BELLARE in [Bel97].

Definition 2.10. *Außerdem werden wir in den folgenden Kapiteln eine Möglichkeit brauchen, Funktionen nach ihrer Größenordnung zu klassifizieren. Wir definieren dazu nach [Sch97] die* O-Notation *wie folgt:*

$$O(f(n)) \stackrel{def}{=} \{g : \mathbb{N} \to \mathbb{N} \mid \exists c, n_0 \forall n \geq n_0 . g(n) \leq c \cdot f(n)\}$$

Dadurch können wir – von konstanten Faktoren abgesehen – eine asymptotische obere Schranke für Funktionen angeben.

Kapitel 3

Kryptographische Grundlagen

Es gibt verschiedene Absichten, die Nutzer eines Kryptosystems verfolgen können. Ein sehr naheliegendes Ziel ist beispielsweise, dass Angreifer keine (nützliche) Information aus Schlüsseltexten erhalten können (Indistinguishability). Ein anderes Ziel könnte sein, dass ein Angreifer keinerlei Möglichkeit besitzt, Schlüsseltexte derart abzuändern, dass die beiden zugehörigen Klartexte „in bedeutungsvollem Zusammenhang[1]“ stehen (Non-Malleability). Dem gegenüber stehen die verschieden starken Attacken, die potentiellen Angreifern zur Verfügung stehen: z. B. (adaptive) Chosen-Plaintext-Attacken und (adaptive) Chosen-Ciphertext-Attacken. Um herauszufinden, wie sicher ein Kryptosystem ist, muss man die möglichen Ziele des Nutzers den Möglichkeiten des Angreifers gegenüberstellen. Als Leitfaden dient dabei die Idee von Bellare, Desai, Pointcheval und Rogaway. Hierbei werden Sicherheitsziele und Attacken auf das Kryptosystem als voneinander unabhängig betrachtet und jedes Paar, bestehend aus einem speziellen Angriff und einem speziellen Ziel, als ein Sicherheitsmodell bezeichnet ([BDPR98], [BDPR01]).
Wie Bellare, Desai, Pointcheval und Rogaway zeigen, bestehen zwischen diesen Sicherheitsmodellen Abhängigkeiten. So lässt sich beispielsweise zeigen, dass ein Kryptosystem, das Non-Malleability erfüllt, auch Indistinguishability erfüllen muss. Außerdem werden wir – Bellare und Sahai folgend ([BS99]) – sehen, dass sich Non-Malleability auf Indistinguishability unter einem bestimmten Angriff zurückführen lässt.

[1] aus dem englischen von „meaningful related“

3.1 Public-Key-Kryptographie

Zu allererst werden wir uns genauer mit dem Vorgang der Informationsübertragung befassen:

Definition 3.1. *Die an der Kommunikation Beteiligten werden wir wie folgt bezeichnen:*

- *Ein* Teilnehmer *ist jemand, der Informationen sendet, empfängt oder manipuliert.*
- *Ein* Sender *ist der Teilnehmer, der der legitime Überträger der Informationen ist.*
- *Ein* Empfänger *ist der Teilnehmer, für den die vom Sender übertragenen Informationen bestimmt sind.*
- *Ein* Angreifer *ist ein Teilnehmer, der weder Sender noch Empfänger ist und versucht, die Informationen, die vom Sender zum Empfänger übertragen werden, zu kompromittieren. Man unterscheidet dabei zwischen* passiven Angriffen*, bei denen der Angreifer lediglich Informationen vom unsicheren Kanal (siehe Definition 3.2) liest und* aktiven Angriffen*, bei denen der Angreifer zusätzlich in der Lage ist, Informationen im unsicheren Kanal zu löschen, zu manipulieren und in ihm zu senden. Während ein passiver Angreifer nur die Sicherheit der übertragenen Daten bedroht, gehen von einem aktiven Angreifer noch weitere Gefahren aus: Er bedroht die Unversehrtheit und Echtheit der Daten.*

Definition 3.2. *Den Weg, über den Informationen zwischen Sender und Empfänger ausgetauscht werden, wollen wir mit* Kanal *bezeichnen, dabei unterscheiden wir:*

- Unsichere Kanäle*, auf denen ein Angreifer Informationen lesen, neu anordnen, löschen oder manipulieren kann.*
- Sichere Kanäle*, auf denen der Angreifer Informationen nicht lesen, nicht neu anordnen, nicht löschen und nicht manipulieren kann.*
- Physikalisch sichere Kanäle*, auf die der Angreifer physikalisch nicht zugreifen kann, wie beispielsweise persönlichen Kontakt, vertrauenswürdige Kuriere oder eine dedizierte Leitung.*

Es sei hier auf den feinen Unterschied zwischen sicheren und physikalisch sicheren Kanälen verwiesen: Sichere Kanäle schließen physikalisch sichere Kanäle mit ein, es ist jedoch auch möglich, einen Kanal mit kryptographischen Mitteln zu sichern.

Bevor wir uns nun den Public-Key-Verfahren zuwenden, deren Idee von DIFFIE und HELLMAN ([DH76]) stammt, wollen wir Kryptosysteme im Allgemeinen definieren. Wir bedienen uns dazu der allgemein üblichen Definition nach BUCHMANN ([Buc99]), die wir allerdings – nach den Ideen von GOLDWASSER und HALEVI ([GM84]) – für probabilistische Verschlüsselung erweitert haben:

Definition 3.3. *Ein* Verschlüsselungsverfahren *oder* Kryptosystem *ist ein Sechstupel* $\Pi = (\mathcal{P}, \mathcal{C}, \mathcal{K}, \mathcal{G}, \mathcal{E}, \mathcal{D})$ *mit folgenden Eigenschaften:*

1. $\mathcal{P}$ *ist eine Menge. Sie heißt* Klartextraum*. Ihre Elemente heißen* Plaintexte *oder* Klartexte*.*
2. $\mathcal{C}$ *ist eine Menge. Sie heißt* Ciphertextraum*. Ihre Elemente heißen* Ciphertexte *oder* Schlüsseltexte*.*
3. $\mathcal{K}$ *ist eine Menge. Sie heißt* Schlüsselraum*. Ihre Elemente heißen* Schlüssel*.*
4. *Der Keygenerator* $\mathcal{G}$ *ist ein probabilistischer Algorithmus, der einen Sicherheitsparameter erhält und ein Paar* (d, e) *zusammengehöriger Ent- und Verschlüsselungsschlüssel (in Polynomialzeit) liefert.*
5. $\mathcal{E} = \{E_e : e \in \mathcal{K}\}$ *ist eine Familie von (probabilistischen) Funktionen* $E_e : \mathcal{P} \times \{0,1\}^* \to \mathcal{C}$*. Ihre Elemente heißen* Verschlüsselungsfunktionen*.*
6. $\mathcal{D} = \{D_d : d \in \mathcal{K}\}$ *ist eine Familie von (deterministischen) Funktionen* $D_d : \mathcal{C} \to \mathcal{P}$ *(bzw.* $D_d(x) = \perp$ *falls* $x \notin \mathcal{C}$*). Ihre Elemente heißen* Entschlüsselungsfunktionen*.*
7. *Für jedes* $e \in \mathcal{K}$ *gibt es ein* $d \in \mathcal{K}$*, so dass für fast alle* $p \in \mathcal{P}$ *(im Sinne von Bemerkung 3.6),* $r \in \{0,1\}^*$ *die Gleichung* $D_d(E_e(p, r)) = p$ *gilt.*

Bemerkung 3.4. Gehen die Klar-, Cipher- und Schlüsseltextmengen aus dem Zusammenhang hervor oder spielen sie für die Betrachtung nur eine untergeordnete Rolle, so wollen wir statt $(\mathcal{P}, \mathcal{C}, \mathcal{K}, \mathcal{G}, \mathcal{E}, \mathcal{D})$ auch abkürzend $(\mathcal{G}, \mathcal{E}, \mathcal{D})$ schreiben.

Bemerkung 3.5. Oft spricht man auch von *Verschlüsselungsalgorithmen* statt von *Verschlüsselungsfunktionen*. Betrachtet man Funktionen nämlich im mathematisch definierten Sinne hieße das, dass es für jeden Klartext zu einem gegebenen Schlüssel nur eine mögliche Verschlüsselung geben kann. Dies wollen wir jedoch nicht fordern; vielmehr wollen wir die Verschlüsselungsfunktion als probabilistische Funktion ansehen. So macht es – wie wir in Abschnitt 3.2.1 sehen werden – durchaus Sinn, bei mehrfachen Anwendungen des Verschlüsselungsalgorithmus (mit demselben Schlüssel) auf denselben

Klartext jeweils einen anderen Ciphertext zu erhalten. Selbstverständlich beeinträchtigt dies die Entschlüsselung nicht: Sie soll eine „echte“ Funktion sein und damit eindeutig. Der Verschlüsselungsalgorithmus darf also bei jeder Iteration mit demselben Klartext einen anderen Ciphertext liefern, er darf jedoch niemals denselben Ciphertext für verschiedene Klartexte liefern.

Bemerkung 3.6. Außerdem wollen wir Fehler im Kryptosystem zulassen, nämlich derart, dass die Verschlüsselung – für einige „wenige“ Klartexte p – Ciphertexte ausgeben kann, die nicht korrekt zum ursprünglichen Klartext entschlüsselt werden.

Wesentlich bei dieser Definition von Kryptosystemen sind die Schlüssel für die Ver- und Entschlüsselung. Man kann argumentieren, dass sich auch Kryptosysteme ohne Schlüssel konstruieren lassen, was jedoch aus zwei Gründen nicht sinnvoll ist: Zum einen muss, falls ein Kryptosystem ohne Schlüssel kompromittiert ist, ein komplett neues konstruiert werden, wohingegen es bei einem Kryptosystem mit Schlüssel genügt, den Schlüssel zu wechseln (vorausgesetzt es ist nur der Schlüssel kompromittiert und nicht das Kryptosystem als solches). Zum anderen führt dies zu sogenannter „security through obscurity“[2] – das Kryptosystem ist nur dann sicher, solange niemand unbefugtes weiß, wie es funktioniert. Dies ist jedoch keine realistische Annahme – zumindest nicht, wenn man das Kryptosystem über einen längeren Zeitraum verwenden will: Mitarbeiter scheiden aus Unternehmen aus und Experten können möglicherweise sehr gute Vermutungen anstellen wie das Kryptosystem funktioniert. Wir folgen deswegen Kerckhoffs Prinzip ([Ker83]), nach dem ein Kryptosystem bekannt sein und die Sicherheit ausschliesslich vom Schlüssel abhängen sollte. Später wurde diese sinnvolle Forderung auch als Shannon's Maxime ([Sha49]: „the enemy knows the system being used“[3]) postuliert.

Man unterscheidet nun zwei Arten von Kryptosystemen:

Definition 3.7. *Gegeben sei ein Kryptosystem* Π *mit einer Ent- bzw. Verschlüsselungsfunktion aus* $\{E_e : e \in \mathcal{K}\}$ *bzw.* $\{D_d : d \in \mathcal{K}\}$*. Dann ist* Π *ein* symmetrisches Kryptosystem*, wenn sowohl d aus e als auch e aus d effizient berechenbar sind.*

Bemerkung 3.8. In der Praxis ist es bei symmetrischen Kryptosystemen häufig der Fall, dass die beiden Schlüssel d und e gleich sind, wodurch sich auch der Name „symmetrisch“ begründet. Man beachte, dass zwei Kommunikationspartner, die mittels eines symmetrischen Kryptosystems miteinander kommunizieren wollen, die Schlüssel vorher über einen sicheren Kanal austauschen und anschließend geheim halten müssen.

[2] Sicherheit durch Unklarkeit

[3] Der Angreifer kennt das verwendete System.

Definition 3.9. *Gegeben sei ein Kryptosystem K mit einer Ent- bzw. Verschlüsselungsfunktion aus* $\{E_e : e \in \mathcal{K}\}$ *bzw.* $\{D_d : d \in \mathcal{K}\}$*. Dann ist K ein* asymmetrisches *oder* Public-Key-Kryptosystem*, wenn es nicht mit vertretbarem Aufwand möglich ist, d aus e zu berechnen.*

Bemerkung 3.10. Wir wollen noch eine übliche Feinheit bei der Benennung der Entschlüsselungsschlüssel einführen: In Public-Key-Kryptosystemen soll er *privater Schlüssel* heißen. Bei symmetrischen Kryptosystemen hingegen soll er als *geheimer Schlüssel* bezeichnet werden. Die Benennung folgt der Idee, dass man mindestens zwei Teilnehmer benötigt, um sich ein Geheimnis zu teilen, wohingegen auf etwas wirklich privates nur ein Teilnehmer Zugriff hat.

Da der Entschlüsselungsschlüssel nicht (mit vertretbarem Aufwand) aus dem Verschlüsselungsschlüssel zu errechnen ist, ist es bei asymmetrischen Kryptosystemen möglich, den Verschlüsselungsschlüssel – z. B. in einem öffentlichen Verzeichnis ähnlich einem Telefonbuch – bekannt zu geben. Man nennt ihn deswegen auch *öffentlichen Schlüssel*. Das heißt, möchte man einer Person eine geheime Nachricht schicken, so besorgt man sich zunächst deren öffentlichen Schlüssel und kann dann mit dessen Hilfe die Nachricht verschlüsseln. Der Empfänger benutzt dann seinen privaten Schlüssel, um die Nachricht zu entschlüsseln. Damit muss man nicht mehr mit jedem Kommunikationspartner ein geheimes Schlüsselpaar austauschen, sondern es genügt, wenn jeder Kommunikationsteilnehmer seinen öffentlichen Schlüssel frei zugänglich macht. Dafür handelt man sich bei Public-Key-Systemen ein anderes Problem ein: Man muss sichergehen, dass der öffentliche Schlüssel, mit dem man verschlüsseln will, auch wirklich von der Person stammt, an die die Nachricht gehen soll. Gelingt es nämlich einem Angreifer, dem Sender statt dem Schlüssel des Empfängers seinen eigenen Schlüssel unterzuschieben, so kann er die Nachrichten lesen. Verschlüsselt er nun die Nachricht mit dem eigentlichen Schlüssel des Empfängers und schickt sie weiter, so bemerkt dieser unter Umständen den Angriff nicht (dazu muss der Angreifer allerdings die Nachricht des Senders nicht nur lesen, sondern auch abfangen). Man kann diese „Man-in-the-middle“ genannte Attacke allerdings verhindern, indem man elektronische Signaturen und Authentifizierungssysteme einsetzt. Dies ist jedoch ein Angriff auf das Protokoll und nicht auf das Kryptosystem an sich, da der Angreifer die Verschlüsselung nicht bricht sondern aushebelt, weswegen wir hier nicht weiter darauf eingehen wollen. Weiterführende Literatur findet sich bei Buchmann ([Buc99]) und Menezes, van Orschott und Vanstone ([MvOV97]). Wir wollen uns nun vielmehr den Angriffen auf Kryptosysteme an sich zuwenden.

3.2 Sicherheitsziele

Wie in der Einleitung bereits erwähnt, gibt es verschiedene Sicherheitsmerkmale, die ein Benutzer von einem Kryptosystem erwarten kann. Die klassische Idee der sicheren Benutzung von Kryptosystemen ist natürlich, das Geheimnis der übermittelten Nachricht zu bewahren (*One-Wayness*). Es gibt jedoch noch andere Anforderungen, die man an ein Kryptosystem stellen kann:

3.2.1 Indistinguishability

Die Idee der Ununterscheidbarkeit oder Indistinguishability (IND) von Verschlüsselungen wurde erstmals von Goldwasser und Micali in Zusammenhang mit Chosen-Plaintext-Angriffen (siehe Abschnitt 3.3) formuliert ([GM84]). Wir wollen hier jedoch der darauf beruhenden Darstellung von Bellare, Desai, Pointcheval und Rogaway ([BDPR98], [BDPR01]) folgen.
Dazu betrachten wir folgendes Experiment: Der Angreifer wählt zwei Klartexte p_0 und p_1 gleicher Länge aus unserem Klartextraum und wir verschlüsseln einen der beiden zum Ciphertext z, das heißt entweder gilt $z = E_e(p_0)$ oder es gilt $z = E_e(p_1)$. Der Angreifer bekommt nun das Tripel (p_0, p_1, z) und soll entscheiden, ob z die Verschlüsselung von p_0 oder von p_1 ist. Hat er – mit den ihm zur Verfügung stehenden Mitteln $\mathcal{M}$ – keine bessere Chance als eine Münze zu werfen oder zu raten, so wollen wir das Kryptosystem als *unter $\mathcal{M}$ ununterscheidbar (indistinguishable)* bezeichnen. Offensichtlich ist dies gleichbedeutend damit, dass der Angreifer aus dem Schlüsseltext keinerlei Information gewinnt (siehe auch [MRS87]). Dies kann man daran erkennen, dass die Wahrscheinlichkeit, dass er den richtigen Klartext benennen kann *bevor* er den Ciphertext bekommt, genauso groß ist wie die Wahrscheinlichkeit, dass er den richtigen Klartext benennen kann *nachdem* er den Ciphertext erhalten hat. In Abschnitt 3.4.2 werden wir eine formale Definition von Indistinguishability geben, die dann auch die Fähigkeiten des Angreifers mit einschließt, die wir hier zunächst nicht berücksichtigt haben.

3.2.2 Non-Malleability

Ein Kryptosystem ist – vereinfacht gesagt – *unmodifizierbar* (*non-malleable*), wenn ein Angreifer einen gegebenen Ciphertext nicht in einen anderen Ciphertext ändern kann, so dass die beiden Ciphertexte „in bedeutungsvollem Zusammenhang“ zueinander stehen. Die ursprüngliche Definition von Non-Malleability (NM), die wir *Simulation Based Non-Malleability* nennen wollen, geht auf Dolev, Dwork und Naor ([DDN91], [DDN95], [DDN00]) zurück. Später gaben Bellare, Desai, Pointcheval und Rogaway ei-

ne auf den ersten Blick anders erscheinende Formalisierung an ([BDPR98], [BDPR01]), die wir mit *Comparison Based Non-Malleability* bezeichnen wollen. Bellare und Sahai konnten jedoch zeigen, dass die beiden Formalisierungen ein gleichwertiges Sicherheitsmodell beschreiben ([BS99]). Viel interessanter ist jedoch ein Zwischenergebnis des Äquivalenzbeweises, bei dem sie zeigen, dass sich Non-Malleability auf eine bestimmte Art der Indistinguishability zurückführen lässt, was wir uns – wie auch die formale Definition von Non-Malleability, für die wir noch verschiedene Angriffsmodelle benötigen – in Abschnitt 3.4.2 näher ansehen werden. Ein wesentlicher Vorteil der auf Indistinguishability basierenden Formalisierung von Non-Malleability ist, dass sie – im Gegensatz zu den beiden zuvor bekannten Formalisierungen – relativ einfach zu benutzen ist. Auch ohne die formale Definition zu kennen, können wir uns jedoch den Unterschied der beiden Definitionen von Non-Malleability einmal näher anschauen:
Beide Definitionen beinhalten eine Relation R zwischen Klartexten und einen Angreifer, der einen Vektor bestehend aus Klartexten ausgibt. Der Angreifer erhält dabei einen Ciphertext z des Klartextes p als Eingabe. Er soll nun einen Vektor von Ciphertexten $\overrightarrow{z}$ zugehörig zu dem Vektor der Klartexte $\overrightarrow{p}$ ausgeben, so dass $R(p, \overrightarrow{p})$ wahr ist. Das Kryptosystem ist non-malleable, wenn die Chance des Angreifers bei dieser Aufgabe genauso groß ist $R(p, \overrightarrow{p})$ zu erfüllen, wie seine Chance $R(p, \overrightarrow{p})$ zu erfüllen wäre, wenn er den Ciphertext z nicht kennen würde und keine weitere Information über den Klartext p außer der Relation R hätte. Die beiden Definitionen unterscheiden sich nun in der genauen Art, wie ein Erfolg des Angreifers gemessen wird: Wie der Name schon andeutet, basiert *Simulation Based Non-Malleability* auf einer Simulation: Ein Kryptosystem ist non-malleable, wenn es für jeden Angriff eine Simulation gibt, die – ohne Information über den Ciphertext z – in etwa dasselbe leistet wie der Angreifer. Bei der Definition von *Comparison Based Non-Malleability* wird für die Non-Malleability eines Kryptosystems verlangt, dass die Erfolgschancen des Angreifers bei einer „echten “ Herausforderung genauso groß sind wie bei einer „erfundenen“. Das Ergebnis von Bellare und Sahai ist um so überraschender, da es noch einen weiteren wesentlichen Unterschied der beiden Definitionen gibt: Während der Angreifer bei der Comparison Based Non-Malleability Zugriff auf ein Entschlüsselungsorakel hat, das ihm Ciphertexte entschlüsselt, darf er dies bei der Simulation Based Non-Malleability nicht benutzen.

3.2.3 Plaintext Awareness

Plaintext Awareness wurde 1994 von Bellare und Rogaway ([BR94], [BR95a]) definiert und 1998 von Bellare, Desai, Pointcheval und Rogaway ([BDPR98], [BDPR01]) verfeinert: Ein Kryptosystem erfüllt genau dann die Anforderungen für *Plaintext Awareness*, wenn ein Angreifer für jeden von ihm generierten Verschlüsselungstext auch den dazugehörigen Klar-

text „kennen“ muss.
Sowohl die formale Definition von Plaintext Awareness als auch alle bisher bekannten Kryptosysteme, die Plaintext Awareness bieten, beruhen fundamental auf dem Random Oracle Modell ([BR95b]). Das Random Oracle Modell ist jedoch ein recht abstraktes Modell, das ideale Hash-Funktionen benutzt, während unser Hauptuntersuchungsgegenstand, das Ajtai-Dwork Kryptosystem, von Hash-Funktionen keinen Gebrauch macht. Wir wollen uns deswegen nicht näher mit dem Random Oracle Modell und der zugehörigen Definition von Plaintext Awareness befassen.
Herzog, Liskov und Micali geben allerdings in [HLM03] eine Definition von *Plaintext Awareness via Key Registration*, die ohne das Random Oracle Modell auskommt. Ihre Definition beruht auf einer öffentlichen Schlüssel-Registrierungsstelle, so dass man nicht nur ein Public-Key-Kryptosystem, sondern auch ein Protokoll benötigt, um mit der Schlüssel-Registrierungsstelle zu kommunizieren. Ihr Modell erfordert des weiteren, dass nicht nur der Empfänger, sondern auch der Sender, einen öffentlichen Schlüssel besitzt. Ähnlich der Definition von Plaintext Awareness im Random Oracle Modell ist die Idee nun, dass jeder Angreifer Ciphertexte, die er erstellt hat, entschlüsseln kann – vorausgesetzt, der Sender hat seinen öffentlichen (Sende)-Schlüssel ordnungsgemäß registriert. Herzog, Liskov und Micali zeigen für ihr Modell, dass nur die Sicherheit der Plaintext Awareness von der Ehrlichkeit der Schlüssel-Registrierungsstelle abhängt, nicht jedoch die Sicherheit der Verschlüsselung. Da sich diese Definition allerdings – wie wir in Kapitel 5 sehen werden – auch nicht so recht auf das Ajtai-Dwork-Kryptosystem anwenden lässt, unter anderem weil noch ein Protokoll benötigt wird, um mit der Schlüssel-Registrierungsstelle zu kommunizieren, wollen wir auch das Thema *Plaintext Awareness via Key Registration* hier nicht vertiefen.

3.3 Angriffsmodelle

Um genauere Aussagen darüber machen zu können, wie sicher ein Kryptosystem ist, ist es erforderlich, sich damit auseinander zu setzen, welche Mittel dem Angreifer möglicherweise zur Verfügung stehen. Dabei wollen wir hauptsächlich Angriffe auf Public-Key-Kryptosysteme betrachten, führen jedoch der Vollständigkeit halber auch einige Angriffe auf symmetrische Kryptosysteme auf.

Ciphertext-Only-Angriff: Bei einem *Ciphertext-Only-Angriff* steht dem Angreifer nur Ciphertext zur Verfügung. In der Regel wird der Angreifer bemüht sein, aus dem Ciphertext den Klartext oder gar den Schlüssel zu gewinnen. Ein Kryptosystem, bei dem dies gelingt, ist total unsicher.

Known-Plaintext-Angriff: Oft muss davon ausgegangen werden, dass der Angreifer nicht nur Ciphertext, sondern möglicherweise auch dazugehörigen Klartext kennt, bspw. weil er den Kontext weiß, in dem die Nachricht steht. Dies ermöglicht einen *Known-Plaintext-Angriff*, bei dem davon ausgegangen wird, dass der Angreifer Plaintext und den dazugehörigen Ciphertext besitzt.

Non-adaptive Chosen-Plaintext-Angriff: Hat der Angreifer nicht nur Paare von Klartext und zugehörigem Ciphertext, sondern kann sich zu einem Klartext den zugehörigen Ciphertext selbst erzeugen, so spricht man von einem *Chosen-Plaintext-Angriff*.

Adaptive Chosen-Plaintext-Angriff (CPA): Ein *adaptive Chosen-Plaintext-Angriff* ist ein Chosen-Plaintext-Angriff, bei dem der Angreifer die zu verschlüsselnden Klartexte in Abhängigkeit vom Ergebnis seiner vorherigen Verschlüsselungen wählen darf.

Bemerkung 3.11. Wir sehen sofort, dass ein Angreifer auf ein Public-Key-Kryptosystem immer die Möglichkeit eines adaptive Chosen-Plaintext-Angriffs hat. Er muss sich dazu lediglich den öffentlichen Schlüssel seines Opfers besorgen und kann dann selbst beliebige Klartexte verschlüsseln, ohne dass sein Opfer etwas davon bemerkt.

Non-adaptive Chosen-Ciphertext-Angriff (CCA1): Beim *Chosen-Ciphertext-Angriff*, dessen formale Definition von Naor und Yung ([NY90] bzw. [NY95]) stammt, hat der Angreifer Zugriff auf ein Entschlüsselungsorakel, das ihm beliebige Ciphertexte entschlüsselt. Der Angreifer hat jedoch nur solange Zugriff auf das Orakel, bis er den Ciphertext erhält, den er entschlüsseln oder manipulieren will. Danach sind ihm keine Anfragen an das Orakel mehr erlaubt. Eine Möglichkeit für einen solchen Angriff ist, dass sich ein Angreifer Zugriff auf das System des Opfers verschafft, der Schlüssel jedoch fest in der Systemumgebung eingebettet ist, so dass dem Angreifer ein direkter Zugriff darauf verwehrt bleibt. Umgangssprachlich wird dieser Angriff auch oft als „lunchtime attack“, „lunch-break attack“ oder „midnight attack“ bezeichnet ([BDPR98]). Die Bezeichnung spielt darauf an, dass ein Angreifer die Abwesenheit seines Opfers – beispielsweise in der Mittagspause – nutzen kann, um – in der Hoffnung, dass ihm dies bei weiteren Angriffen von Nutzen ist – Zugang zu dessen Rechner und insbesondere der Entschlüsselung zu erlangen.

Adaptive Chosen-Ciphertext-Angriff (CCA2): Eine stärkere Form des Chosen-Ciphertext-Angriffs geht auf Rackoff und Simon ([RS92]) zurück: Beim *adaptive Chosen-Ciphertext-Angriff* ist es dem Angreifer erlaubt, das Entschlüsselungsorakel auch nach dem Erhalt des Ciphertextes,

den er entschlüsseln oder manipulieren will, noch zu befragen. Dabei darf der vom Orakel zu entschlüsselnde Ciphertext auch vom zu entschlüsselnden bzw. manipulierenden Ciphertext abhängen, nicht jedoch mit ihm identisch sein.

Parallel-Angriff (PA0, PA1, PA2): Der *Parallel-Angriff* ist eine Erweiterung des Chosen-Plaintext- bzw. der Chosen-Ciphertext-Angriffe. Dabei hat der Angreifer, nachdem er den Ciphertext erhalten hat, noch die Möglichkeit, Anfragen an ein Entschlüsselungsorakel zu stellen, wobei die Anfragen jedoch nicht von den Ergebnissen vorheriger Anfragen abhängen dürfen. Man kann sich dies so vorstellen, dass der Angreifer alle Anfragen an das Entschlüsselungsorakel *parallel* durchführen muss – daher auch der Name. Dabei bezeichnen steigende Ziffern stärkere Angriffe, d. h. PA0 (PA1 bzw. PA2) liegt CPA (CCA1 bzw. CCA2) zugrunde.
Dieses Modell wurde von Bellare und Sahai entwickelt, um Non-Malleability auf eine bestimmte Form der Indistiguishability zurückzuführen ([BS99]). Wir wollen dies hier nicht weiter ausführen, sondern werden uns dies in Abschnitt 3.4.2 näher anschauen.

Bemerkung 3.12. Es ist offensichtlich, dass PA2 und CCA2 dasselbe Angriffsmodell beschreiben. Da der Angreifer beim Adaptive Chosen-Ciphertext-Angriff beliebige Anfragen an das Entschlüsselungsorakel stellen darf, nachdem er den Ciphertext erhalten hat, bringt ihm eine zusätzliche parallele Anfrage keinerlei Vorteile. Die Benennung des Angriffmodells PA2 ist also rein systematischer Natur.

Ciphertext-Verification-Angriff: Das Szenario des *Ciphertext-Verification-Angriff* wurde 1999 von Halevi und Krawczyk ([HK99]) für Public-Key-Kryptosysteme definiert. Beim Ciphertext-Verification-Angriff hat der Angreifer den öffentlichen Schlüssel und Zugriff auf ein Plaintext-Checking Oracle, welches für ein Paar (p, z) – bestehend aus Plaintext p und Ciphertext z – Auskunft darüber gibt, ob z ein gültiger Ciphertext von p ist. Diese Attacke wurde in [NP02] von Nguyen und Pointcheval auch als *Plaintext-Checking-Attack* bezeichnet.

Bemerkung 3.13. Wir sehen leicht, dass für den Fall der bitweisen Verschlüsselung ein Ciphertext-Verification-Angriff gleichwertig zu einem Chosen-Ciphertext-Angriff ist, da es für den Klartext nur zwei Möglichkeiten gibt. Betrachtet man nämlich den Ciphertext z, so gibt es für die Entschlüsselung nur drei Möglichkeiten: 0, 1 bzw. $\perp$ – falls z kein gültiger Ciphertext war, so dass man z mit maximal zwei Anfragen an das Plaintext-Checking Oracle entschlüsseln kann.

Reaktions-Angriff: Hall, Goldberg und Schneier schlagen in [HGS99] ein Szenario vor, bei dem der Angreifer sich Entschlüsselungs-

fehler des Kryptosystems zu Nutze macht. Der *Reaktions-Angriff* befasst sich nicht mit dem (oder den) dem Kryptosystem zugrundeliegenden Problem(en), sondern erhält seine Information über den Schlüssel oder den Klartext dadurch, dass er die Reaktion des Empfängers beim Entschlüsseln des Ciphertextes beobachtet. Dies kann entweder durch die Beobachtung eines geschützen Systems wie bspw. einer Smartcard passieren oder aber durch Social Engineering. Beim Social Engineering bringt der Angreifer sein Opfer durch gezielte Täuschung dazu, benötigte Informationen direkt oder indirekt preiszugeben. Eine ausführlichere Darstellung von Social Engineering findet sich in [MS02]. Wie der Angriff exakt erfolgt, hängt vom Einfallsreichtum des Angreifers beziehungsweise der Naivität des Opfers ab.

3.4 Sicherheitsmodelle

Wie wir in den letzten beiden Abschnitten gesehen haben, gibt es verschiedene Angriffe auf und verschiedene Sicherheitsziele für Kryptosysteme. Deswegen läßt sich keine eindeutige Definition von Sicherheit finden, sondern es gibt verschiedene Abstufungen davon. Der Sinn von Sicherheitsmodellen besteht nun darin, eine formale Grundlage zur Einstufung der Sicherheit verschiedener Kryptosysteme zu bieten.
In der Literatur finden sich verschiedene Definitionen von Sicherheitsmodellen. Diese sollen zunächst vorgestellt werden, um dann die für unsere Zwecke am besten geeigneten genauer zu betrachten.

3.4.1 Von der perfekten bis zur Ad-Hoc-Sicherheit

Eine grobe Klassifizierung von Sicherheit nehmen Menezes, van Orschott und Vanstone vor und schlagen in [MvOV97] fünf Sicherheitsmodelle vor:

Perfekte Sicherheit: Wie der Name schon vermuten lässt, ist dies das strengste der fünf Sicherheitsmodelle. Ein Angreifer soll trotz unbeschränkter Ressourcen (Rechenkraft und Speicher) nicht in der Lage sein, das Kryptosystem zu brechen. Dazu ist es notwendig, dass ein Angreifer keinerlei Information aus dem Schlüsseltext gewinnen kann, d. h. ähnlich wie bei der Indistinguishability darf sich für den Angreifer die Wahrscheinlichkeit, dass er den richtigen Klartext benennen kann, nicht dadurch verbessern, dass er den Schlüsseltext kennt. Man kann sogar noch weiter gehen und die Information, dass überhaupt Nachrichten übertragen wurden, durch das Festlegen von „leeren Nachrichten“ verhindern. Für *perfekte Sicherheit* ist es allerdings erforderlich, dass der Schlüssel mindestens so lange wie der Klartext ist. Daraus folgt natürlich auch, dass jeder Schlüssel nur einmal benutzt werden darf. Ein Kryptosystem, das dieses Sicherheitskriterium erfüllt, ist zum Beispiel das Vernam-One-Time-Pad (siehe [Buc99]). Trotz der Tatsache, dass

One-Time-Pads theoretisch perfekt sicher sind, besteht in der Praxis durchaus eine Chance sie anzugreifen[4], was allerdings auf ungenaue Benutzung zurückzuführen ist. Der Schlüssel muss *wirklich* zufällig gewählt werden und außerdem geheim gehalten werden. Im Allgemeinen bieten Kryptosysteme keine perfekte Sicherheit. So kann z. B. ein Public-Key-Kryptosystem nicht perfekt sicher sein: Hat man einen Schlüsseltext z gegeben, so ist es mit unbegrenzter Rechenkraft möglich, alle Klartexte zu verschlüsseln bis man den Schlüsseltext z beim Verschlüsseln erhält. Dies ist auch möglich, wenn es sich um ein probabilistisches Kryptosystem handelt, hier muss man jedoch für jede mögliche Wahl des Zufallsparameters alle Klartexte verschlüsseln und mit dem Schlüsseltext z vergleichen. Mit unbeschränkten Ressourcen bleibt dies allerdings immer noch durchführbar.

Komplexitätstheoretische Sicherheit: Sie hat nicht immer einen praktischen Nutzen, kann aber zu einem besseren Verständnis von Sicherheit an sich führen. Komplexitätstheoretische Analysen sind unabdingbar für das Formulieren von Prinzipien und Festigen der Intuition. Man führt *komplexitätstheoretische Sicherheitsnachweise*, indem man ein passendes Berechenbarkeitsmodell definiert und dem potentiellen Angreifer in diesem Modell von den Sicherheitsparametern abhängige polynomielle Rechenkraft (bzw. Speicherplatz) gewährt. Meistens benutzt man hierzu asymptotische und Worst-Case-Analysen, weswegen sich die Ergebnisse – wie schon erwähnt – nicht unbedingt auf die Praxis übertragen lassen. So kann es durchaus sein, dass im Modell durchführbare Berechnungen in der Praxis nicht realisierbar sind.

Beweisbare Sicherheit: Ein Kryptosystem gilt als *beweisbar sicher*, wenn gezeigt werden kann, dass es genauso schwierig zu brechen ist, wie ein bekanntes mutmaßlich schweres Problem zu lösen ist. Meistens handelt es sich um ein zahlentheoretisches Problem, beispielsweise um Faktorisierung oder die Berechnung eines diskreten Logarithmus. Dabei ist zu beachten, dass der Nachweis der Schwierigkeit des zugrunde liegenden Problems nicht gefordert ist, sondern dass die Vermutung dessen für dieses Sicherheitsmodell ausreicht. Deswegen sprachen wir auch von einem „mutmaßlich schweren Problem“. Beweisbare Sicherheit kann – wie wir gleich sehen werden – als Spezialfall der berechenbaren Sicherheit angesehen werden.
Ein Beispiel für beweisbare Sicherheit ist RSA (siehe [RSA77] und [RSA78]), wenn man als Zielstellung des Angreifers annimmt, dass er den privaten Schlüssel seines Opfers erhalten will. Es lässt sich nämlich zeigen, dass es genauso schwer ist den privaten Schlüssel zu finden wie eine Primfaktorzer-

[4] „As a practical person, I've observed that one-time pads are theoretically unbreakable, but practically very weak. By contrast, conventional ciphers are theoretically breakable, but practically strong.“ – Steve Bellovin

legung des entsprechenden RSA-Moduls vorzunehmen.

Berechenbare Sicherheit: Hierbei betrachtet man den Aufwand an Rechenzeit und Speicher, der benötigt wird, um das Kryptosystem mit dem besten bekannten Algorithmus zu brechen. Man geht davon aus, dass das entsprechende System ausreichend genau untersucht wurde, um die Algorithmen, mit denen das System gebrochen werden könnte, zu kennen. Ein Kryptosystem heißt *berechenbar sicher*, wenn der Aufwand es zu brechen die Ressourcen des Angreifers deutlich übersteigt. Die meisten Kryptosysteme dieser Klasse beruhen auf schwierigen Problemen, bei denen (noch) kein Äquivalenzbeweis – wie für beweisbar sichere Kryptosysteme gefordert – gelungen ist. Kryptosysteme dieser Klasse werden oft auch als praktisch sicher bezeichnet.
Auch hier können wir wiederum RSA als Beispiel nehmen, allerdings müssen wir diesmal davon ausgehen, dass der Angreifer nicht den privaten Schlüssel ermitteln will, sondern es darauf abgesehen hat, einen Schlüsseltext zu entziffern. Es ist ein offenes Problem, ob der Angreifer dazu den privaten Schlüsseln benötigt oder ob er andere Möglichkeiten hat den Ciphertext zu entschlüsseln.

Ad-Hoc-Sicherheit: Das Modell der *Ad-Hoc-Sicherheit* basiert auf einer beliebigen Vielfalt plausibler Argumente, dass eine erfolgreiche Attacke die Ressourcen eines bestimmten Angreifers übersteigt. Auch wenn dies ein oft benutzter „Nachweis“ der Sicherheit ist, so ist es natürlich auch der am wenigsten zufriedenstellende, da man nie ausschließen kann, dass bisher unbekannte Attacken eine Bedrohung des Systems darstellen.

Bemerkung 3.14. Teile dieser Klassifizierung gehen bereits auf Shannon ([Sha49]) zurück, der die Begriffe der perfekten, beweisbaren und Ad-Hoc-Sicherheit, wenn auch teilweise unter anderem Namen, bereits 1949 einführte.
Goldwasser und Micali geben außerdem unter dem Namen *semantische Sicherheit* eine Variante von Shannons perfekter Sicherheit an, bei der der Angreifer polynomiell beschränkt ist ([GM84]). Es gibt noch andere Klassifizierungen von Sicherheit wie z. B. *polynomielle Sicherheit*, die ebenfalls von Goldwasser und Micali stammt ([GM84]), und *Y-Sicherheit*, die auf Yao ([Yao82]) zurückgeht. Da Micali, Rackoff und Sloan in [MRS87] jedoch zeigen, dass diese ein äquivalentes Sicherheitsmodell beschreiben, wollen wir auf die einzelnen Formalisierungen nicht weiter eingehen.

Wie wir in Kapitel 5 sehen werden, fällt das Ajtai-Dwork-Kryptosystem in die Klasse der beweisbar sicheren Kryptosysteme. Für unsere Zwecke ist diese Klassifizierung jedoch zu allgemein. Wir wollen uns deswegen einer detaillierteren Einstufung zuwenden.

3.4.2 Orthogonalität von Ziel und Angriff

Mit der jetzigen Kenntnis von Angriffsmodellen, können wir uns – Bellare, Desai, Pointcheval und Rogaway folgend ([BDPR98] bzw. [BDPR01]) – den formalen Definitionen von Indistinguishability und Non-Malleability zuwenden:

Indistinguishability

Nachdem wir bereits informell aus Abschnitt 3.2.1 wissen, was Ununterscheidbarkeit eines Kryptosystems heißt, wollen wir nun die formale Definition betrachten, die wir bei der Analyse des Ajtai-Dwork-Kryptosystems benötigen werden. Wir wollen dazu Bezug auf drei Angriffsmodelle nehmen: adaptive Chosen-Plaintext-Angriffe, non-adaptive Chosen-Ciphertext-Angriffe und adaptive Chosen-Ciphertext-Angriffe.
Für die Definition benötigen wir folgendes Experiment: Ein Keygenerator $\mathcal{G}$ des Kryptosystems erzeugt ein Schlüsselpaar (d, e). Algorithmus A_1 – die erste Stufe des Algorithmus bzw. Angreifers A – gibt nun zwei verschiedene Klartexte p_0 und p_1 derselben Länge und die Statusinformation s, die für die zweite Stufe benötigt wird, aus. Danach wird zufällig einer der beiden Klartexte gewählt und zum Ciphertext z verschlüsselt. Nun erhält Algorithmus A_2 – die zweite Stufe von A – die beiden Klartexte p_0 und p_1, die Statusinformation s der ersten Stufe und den Ciphertext z. Algorithmus A_2 soll entscheiden welcher der beiden Klartexte p_0 und p_1 zu z verschlüsselt wurde. Entscheidend für einen Erfolg des Angreifers ist nun, ob seine Chance wesentlich besser als zu raten ist.
Da der einzige Unterschied zwischen den Definitionen bezüglich der Angriffsmodelle im Zugriff der Algorithmen A_1 und A_2 auf das Entschlüsselungsorakel besteht, definieren wir der Einfachkeit und Übersicht halber alle drei Versionen zusammen. Mit $\mathcal{O}_d$ meinen wir dabei ein Entschlüsselungsorakel zum privaten Schlüssel d; mit ϵ – wie auch in den folgenden Abschnitten in diesem Kapitel – die Funktion, die bei jeder Eingabe, den leeren String ϵ zurückliefert.

Definition 3.15. [IND-CPA, IND-CCA1, IND-CCA2] *Sei $\Pi = (\mathcal{G}, \mathcal{E}, \mathcal{D})$ ein Public-Key-Kryptosystem und $A = (A_1, A_2)$ ein Algorithmus (Angreifer). Für die Angriffe $atk \in \{cpa, cca1, cca2\}$ und $k \in \mathbb{N}$ sei die Chance des Angreifers*

$$\boldsymbol{Adv}_{A,\Pi}^{ind-atk}(k) \stackrel{def}{=} Pr[\boldsymbol{Exp}_{A,\Pi}^{ind-atk-1}(k) = 1] - Pr[\boldsymbol{Exp}_{A,\Pi}^{ind-atk-0}(k) = 1]$$

wobei für $b \in \{0, 1\}$ das Experiment $\boldsymbol{Exp}_{A,\Pi}^{ind-atk-b}(k) = o$ wie folgt definiert sei:

(d, e)	$\xleftarrow{R}$	$\mathcal{G}(k)$;	*Erzeugung der Schlüssel*
(p_0, p_1, s)	$\leftarrow$	$A_1^{\mathcal{O}_1}(e)$;	*1. Stufe von A*
z	$\leftarrow$	$\mathcal{E}_e(p_b)$;	*Verschlüsselung des Klartextes*
o	$\leftarrow$	$A_2^{\mathcal{O}_2}(p_0, p_1, s, z)$	*2. Stufe von A*

mit:

$\mathcal{O}_1(\cdot) = \epsilon$	*und*	$\mathcal{O}_2 = \epsilon$	*für atk=cpa*
$\mathcal{O}_1(\cdot) = \mathcal{O}_d(\cdot)$	*und*	$\mathcal{O}_2 = \epsilon$	*für atk=cca1*
$\mathcal{O}_1(\cdot) = \mathcal{O}_d(\cdot)$	*und*	$\mathcal{O}_2 = \mathcal{O}_d(\cdot)$	*für atk=cca2*

Wie bereits in der informellen Definition von Indistinguishability beschrieben, gehen wir davon aus, dass der Angreifer zwei Klartexte derselben Länge wählt: $|p_0| = |p_1|$. Im Falle von CCA2 fordern wir außerdem, dass der Angreifer das Orakel $\mathcal{O}_2$ nicht nach der Entschlüsselung von z fragt.
Ist nun die Chance $\boldsymbol{Adv}_{A,\Pi}^{ind-atk}(k)$ eines polynomiell beschränkten Angreifers vernachlässigbar, so nennen wir das Kryptosystem Π sicher im Sinne von IND-ATK.

Bemerkung 3.16. Wir gehen davon aus, dass die erste Stufe des Algorithmus, A_1, zwei verschiedene Klartexte p_0 und p_1 wählt. Andernfalls gäbe es ja keine Möglichkeit die beiden zu unterscheiden und A würde trivialerweise immer richtig liegen, da ja z dann sowohl eine Verschlüsselung von p_0 als auch von p_1 wäre. Dies heißt insbesondere für eine bitweise Verschlüsselung, dass es „nur“ darauf ankommt, Verschlüsselungen von Null von Verschlüsselungen von Eins zu unterscheiden.

Bemerkung 3.17. Das Sicherheitsmodell IND-CPA ist auch als *semantische Sicherheit* bekannt. Es beschreibt – wie schon erwähnt – eine Variante der perfekten Sicherheit, bei der der Angreifer polynomiell begrenzt ist.

Diese Definition wurde im Jahr 2000 von Baudron, Pointcheval und Stern als Single-User-Indistinguishability bezeichnet. Sie konnten außerdem zeigen, dass ein Angreifer keinerlei Vorteile davon hat, wenn er mehrere öffentliche Schlüssel besitzt, indem sie die Äquivalenz dieser Definition und einer Multi-User-Indistinguishability ([BPS00]) bewiesen.

Non-Malleability

Wir haben bereits in Abschnitt 3.2.2 bemerkt, dass es zwei verschiedene Definitionen von Non-Malleability gibt und werden uns deswegen zunächst die entsprechenden formalen Definitionen von Comparison Based Non-Malleability und Simulation Based Non-Malleability ansehen. Da wir noch die Definition der Indistinguishability im Gedächtnis haben und die Definition der Comparison Based Non-Malleability ihr recht ähnlich ist, wollen wir mit dieser anfangen:

Comparison Based Non-Malleability: Für die Definition benötigen wir folgendes Experiment: Ein Keygenerator $\mathcal{G}$ des Kryptosystems erzeugt

ein Schlüsselpaar (d, e). Algorithmus A_1 – die erste Stufe des Algorithmus A – gibt nun einen durch den Algorithmus P beschriebenen Klartextraum $\mathcal{P}$ und die Statusinformation s, die für die zweite Stufe benötigt wird, aus. $\mathcal{P}$ muss ein zulässiger Nachrichtenraum sein, d. h. es darf Wahrscheinlichkeiten größer Null nur für Klartexte bestimmter Länge geben. Algorithmus A_2 – die zweite Stufe von A – erhält nun die Verschlüsselung z eines zufälligen Klartextes p_1 aus $\mathcal{P}$ (sowie die Statusinformationen s der ersten Stufe und die Beschreibung des Klartextraumes P) und soll eine Relation R sowie einen Vektor $\overrightarrow{z}$, der in keiner Komponente z enthält, zurückgeben. Das Ziel des Algorithmus A_2 ist, dass $R(p_1, \overrightarrow{p})$ mit $\overrightarrow{p} \leftarrow \mathcal{D}_d(\overrightarrow{z})$ wahr ist. Der Angriff ist erfolgreich, wenn $A = (A_1, A_2)$ dies – mit einer nicht vernachlässigbaren Wahrscheinlichkeit – besser schafft, als $R(p_0, \overrightarrow{p})$ mit einem zufällig gewählten p_0 erfüllt wird.
Da auch hier der einzige Unterschied zwischen den Definitionen bezüglich der Angriffsmodelle im Zugriff der Algorithmen A_1 und A_2 auf das Entschlüsselungsorakel besteht, definieren wir der Einfachkeit und Übersicht halber alle drei Versionen zusammen:

Definition 3.18. [CNM-CPA, CNM-CCA1, CNM-CCA2] *Sei $\Pi = (\mathcal{G}, \mathcal{E}, \mathcal{D})$ ein Public-Key-Kryptosystem, R eine Relation und $A = (A_1, A_2)$ ein Algorithmus (Angreifer). Für die Angriffe $atk \in \{cpa, cca1, cca2\}$ und $k \in \mathbb{N}$ sei die Chance des Angreifers*

$$\boldsymbol{Adv}^{cnm-atk}_{A,\Pi}(k) \stackrel{def}{=} Pr[\boldsymbol{Exp}^{cnm-atk-1}_{A,\Pi}(k) = 1] - Pr[\boldsymbol{Exp}^{cnm-atk-0}_{A,\Pi}(k) = 1]$$

wobei für $b \in \{0, 1\}$ das Experiment $\boldsymbol{Exp}^{cnm-atk-b}_{A,\Pi}(k) = o$ wie folgt definiert sei:

(d, e)	$\stackrel{R}{\leftarrow}$	$\mathcal{G}(k)$;	*Erzeugung der Schlüssel*
(P, s)	$\leftarrow$	$A_1^{\mathcal{O}_1}(e)$;	*1. Stufe von A*
p_0, p_1	$\leftarrow$	P;	
z	$\leftarrow$	$\mathcal{E}_e(p_1)$;	*Verschlüsselung des Klartextes*
$(R, \overrightarrow{z})$	$\leftarrow$	$A_2^{\mathcal{O}_2}(P, s, z)$;	*2. Stufe von A*
$\overrightarrow{p}$	$\leftarrow$	$\mathcal{D}_d(\overrightarrow{z})$;	

falls $z \notin \overrightarrow{z} \wedge \perp \notin \overrightarrow{p} \wedge R(p_b, \overrightarrow{p})$ dann $o \leftarrow 1$; sonst $o \leftarrow 0$
mit:

$\mathcal{O}_1(\cdot) = \epsilon$	*und*	$\mathcal{O}_2 = \epsilon$	*für atk=cpa*
$\mathcal{O}_1(\cdot) = \mathcal{O}_d(\cdot)$	*und*	$\mathcal{O}_2 = \epsilon$	*für atk=cca1*
$\mathcal{O}_1(\cdot) = \mathcal{O}_d(\cdot)$	*und*	$\mathcal{O}_2 = \mathcal{O}_d(\cdot)$	*für atk=cca2*

Wie bereits im vorherigen Absatz beschrieben, gehen wir davon aus, dass der Angreifer einen zulässigen Klartextraum $\mathcal{P}$ wählt, d. h. dass alle Klartexte die mit einer größeren Wahrscheinlichkeit als Null auftreten, dieselben Länge haben: $|p_0| = |p_1|$. Im Fall von CCA2 fordern wir außerdem, dass der Angreifer das Orakel $\mathcal{O}_2$ nicht nach der Entschlüsselung von z fragt.

Wir nennen das Kryptosystem Π sicher im Sinne von CNM-ATK*, wenn die Chance* $\boldsymbol{Adv}_{A,\Pi}^{cnm-atk}(k)$ *für jeden polynomiell beschränkten Algorithmus A, der einen gültigen in Polynomialzeit erfassbaren – durch P beschriebenen – Klartextraum* $\mathcal{P}$ *und eine in Polynomialzeit berechenbare Relation R ausgibt, vernachlässigbar ist.*

Bemerkung 3.19. Die Vorausetzung, dass M ein zulässiger Klartextraum sein muss, trägt der Tatsache Rechnung, dass Verschlüsselungen nicht notwendigerweise die Länge des Klartextes verbergen müssen. Außerdem wollen wir dem Algorithmus A die Triviallösung – nämlich das Kopieren des Ciphertextes z – nicht erlauben, weswegen wir fordern, dass z nicht in $\overrightarrow{z}$ enthalten sein darf.

Simulation Based Non-Malleability: Auch bei der Simulation Based Non-Malleability betrachten wir wieder ein Experiment, dazu sei $R(p, \overrightarrow{p}, P, s_1)$ eine in Polynomialzeit berechenbare Funktion, die vier Argumente erhält, wobei wir mit $\overrightarrow{p}$ einen Vektor mit einer frei wählbaren Zahl an Komponenten meinen und mit P wie gehabt die Beschreibung eines Klartextraumes $\mathcal{P}$. Auch hier läuft der Algorithmus $A = (A_1, A_2)$ in zwei Stufen ab. Die erste Stufe des Algorithmus, A_1, berechnet eine Verteilung von Klartextnachrichten sowie die Statusinformationen s_1, die die Relation R erhält, und s_2. Die zweite Stufe des Algorithmus, A_2, erhält nun die Verschlüsselung z eines zufällig gewählten Klartextes p aus $\mathcal{P}$ und die Statusinformation s_2 von A_1. Die Aufgabe von A_2 ist es nun einen Vektor $\overrightarrow{z}$ mit Ciphertexten derart auszugeben, dass $R(p, \overrightarrow{p}, P, s_1)$ erfüllt ist, $z \notin \overrightarrow{z}$ und $\perp \notin \overrightarrow{p}$ mit $\overrightarrow{p} = \mathcal{D}_d(\overrightarrow{z})$ gilt. Das Kryptosystem soll nun als sicher gelten, wenn es für jeden polynomiell beschränkten Algorithmus (Angreifer) A und jede in Polynomialzeit auswertbare Relation R einen polynomiell beschränkten Algorithmus (Simulator) S gibt, der ohne den Ciphertext z zu kennen, in etwa dieselbe Erfolgswahrscheinlichkeit wie A hat. Je nach Angriffsmodell erhalten die jeweiligen Stufen von A Zugriff auf das Entschlüsselungsorakel $\mathcal{O}_d$. Der Simulator S erhält jedoch in keinem der Fälle Zugriff auf ein Entschlüsselungsorakel – unabhängig davon, ob A Anfragen stellen darf.
Wie bei den vorangegangenen Definitionen, definieren wir der Einfachkeit und Übersicht halber die Versionen für die drei Angriffsmodelle zusammen:

Definition 3.20. [SNM-CPA, SNM-CCA1, SNM-CCA2] *Sei* $\Pi = (\mathcal{G}, \mathcal{E}, \mathcal{D})$ *ein Public-Key-Kryptosystem, R eine Relation,* $A = (A_1, A_2)$ *ein Algorithmus (Angreifer) und* $S = (S_1, S_2)$ *ein zweiter Algorithmus (der „Simulator“). Für die Angriffe* $atk \in \{cpa, cca1, cca2\}$ *und* $k \in \mathbb{N}$ *sei die Chance des Angreifers*

$$\boldsymbol{Adv}_{A,S,\Pi}^{snm-atk}(k) \stackrel{def}{=} Pr[\boldsymbol{Exp}_{A,\Pi}^{snm-atk}(k) = 1] - Pr[\boldsymbol{Exp}_{S,\Pi}^{snm-atk}(k) = 1]$$

wobei das Experiment $\boldsymbol{Exp}^{snm-atk}_{A,\Pi}(k) = o$ *wie folgt definiert sei:*

(d, e)	$\xleftarrow{R}$	$\mathcal{G}(k)$;	*Erzeugung der Schlüssel*
(P, s_1, s_2)	$\leftarrow$	$A_1^{\mathcal{O}_1}(e)$;	*1. Stufe von A*
p	$\leftarrow$	P;	
z	$\leftarrow$	$\mathcal{E}_e(p)$;	*Verschlüsselung des Klartextes*
$\overrightarrow{z}$	$\leftarrow$	$A_2^{\mathcal{O}_2}(s_2, z)$;	*2. Stufe von A*
$\overrightarrow{p}$	$\leftarrow$	$\mathcal{D}_d(\overrightarrow{z})$;	

falls $z \notin \overrightarrow{z} \wedge \perp \notin \overrightarrow{p} \wedge R(p, \overrightarrow{p}, P, s_1)$ *dann* $o \leftarrow 1$*; sonst* $o \leftarrow 0$

analog dazu sei das Experiment $\boldsymbol{Exp}^{snm-atk}_{S,\Pi}(k) = o$ *wie folgt definiert:*

(d, e)	$\xleftarrow{R}$	$\mathcal{G}(k)$;	*Erzeugung der Schlüssel*
(M, s_1, s_2)	$\leftarrow$	$S_1(e)$;	*1. Stufe von S*
p	$\leftarrow$	P;	
$\overrightarrow{z}$	$\leftarrow$	$S_2(s_2)$;	*2. Stufe von S*
$\overrightarrow{p}$	$\leftarrow$	$\mathcal{D}_d(\overrightarrow{z})$;	

falls $\perp \notin \overrightarrow{p} \wedge R(p, \overrightarrow{p}, P, s_1)$ *dann* $o \leftarrow 1$*; sonst* $o \leftarrow 0$

mit:

$\mathcal{O}_1(\cdot) = \epsilon$	*und*	$\mathcal{O}_2 = \epsilon$	*für atk=cpa*
$\mathcal{O}_1(\cdot) = \mathcal{O}_d(\cdot)$	*und*	$\mathcal{O}_2 = \epsilon$	*für atk=cca1*
$\mathcal{O}_1(\cdot) = \mathcal{O}_d(\cdot)$	*und*	$\mathcal{O}_2 = \mathcal{O}_d(\cdot)$	*für atk=cca2*

Auch hier gehen wir davon aus, dass der Angreifer einen zulässigen Klartextraum $\mathcal{P}$ *wählt, d. h. dass alle Klartexte die mit einer größeren Wahrscheinlichkeit als Null auftreten, dieselben Länge haben:* $|p_0| = |p_1|$*. Im Fall von CCA2 fordern wir außerdem, dass der Angreifer das Orakel* $\mathcal{O}_2$ *nicht nach der Entschlüsselung von z fragt.*
Wir nennen das Kryptosystem Π sicher im Sinne von SNM-ATK*, wenn für jede in Polynomialzeit berechenbare Relation R und jeden polynomiell beschränkten Algorithmus A, der einen gültigen in Polynomialzeit erfassbaren – durch P beschriebenen – Klartextraum* $\mathcal{P}$ *ausgibt, ein Algorithmus S, der in Polynomialzeit läuft, existiert, so dass die Chance* $\boldsymbol{Adv}^{snm-atk}_{A,S,\Pi}(k)$ *vernachlässigbar ist.*

Bemerkung 3.21. Bemerkung 3.19 gilt auch für Simulation Based Non-Malleability.

Indistinguishability unter parallelen Attacken: Wie wir im nächsten Absatz sehen werden, lässt sich Non-Malleability auf Indistinguishability unter einer parallelen Attacke zurückführen, welche unserem Modell von Indistinguishability unter Chosen-Plaintext- und Chosen-Ciphertext-Angriffen sehr ähnlich ist. Die einzige Veränderung besteht – wie bereits in Abschnitt 3.3 beschrieben – darin, dass der Angreifer eine zusätzliche Anfrage an ein Entschlüsselungsorakel stellen darf, nachdem er den zu entschlüsselnden Ciphertext erhalten hat. Formal bilden wir dies wie folgt ab:

Die erste Stufe des Algorithmus A läuft wie gehabt ab, ebenso die zufällige Wahl eines Klartextes und seine Verschlüsselung. Die zweite Stufe des Algorithmus, A_2, wird in einen Frageteil $A_{2,q}$ und einen Entscheidungsteil $A_{2,g}$ unterteilt. Der Frageteil $A_{2,q}$ erhält die beiden Klartexte p_0, p_1, den Verschlüsselungstext z einer der beiden Klartexte und die Statusinformationen s_1 der ersten Stufe des Algorithmus, A_1, als Eingabe und gibt nun seinerseits Statusinformationen s_2 sowie einen Vektor $\overrightarrow{z}$ mit höchstens polynomiell vielen Ciphertexten aus. Dieser wird komponentenweise von einem Entschlüsselungsorakel $\mathcal{O}_{d'}$ zu einem Vektor $\overrightarrow{p}$ mit den jeweils zugehörigen Klartexten entschlüsselt. Der Entscheidungsteil $A_{2,q}$ erhält nun diesen Klartextvektor $\overrightarrow{p}$ sowie die zuvor gespeicherten Statusinformationen s_2 als Eingabe und soll entscheiden, welcher der beiden Klartexte p_0 und p_1 zu z verschlüsselt wurde.
Der Unterschied zwischen den einzelnen Definitionen liegt auch hier erneut nur im Zugriff der Algorithmen A_1 und A_2 auf das Entschlüsselungsorakel $\mathcal{O}_d$, weswegen wir auch hier – wie gehabt – alle Definitionen zusammen geben wollen:

Definition 3.22. [IND-PA0, IND-PA1, IND-PA2] *Sei $\Pi = (\mathcal{G}, \mathcal{E}, \mathcal{D})$ ein Public-Key-Kryptosystem und $A = (A_1, A_2)$ ein Algorithmus (Angreifer). Für die Angriffe $atk \in \{pa0, pa1, pa2\}$ und $k \in \mathbb{N}$ sei die Chance des Angreifers*

$$\boldsymbol{Adv}^{ind-atk}_{A,\Pi}(k) \stackrel{def}{=} Pr[\boldsymbol{Exp}^{ind-atk-1}_{A,\Pi}(k) = 1] - Pr[\boldsymbol{Exp}^{ind-atk-0}_{A,\Pi}(k) = 1]$$

wobei für $b \in \{0,1\}$ das Experiment $\boldsymbol{Exp}^{ind-atk-b}_{A,\Pi}(k) = o$ wie folgt definiert sei:

(d, e)	$\stackrel{R}{\leftarrow}$	$\mathcal{G}(k)$;	*Erzeugung der Schlüssel*
(p_0, p_1, s_1)	$\leftarrow$	$A_1^{\mathcal{O}_1}(e)$;	*1. Stufe von A*
z	$\leftarrow$	$\mathcal{E}_e(p_b)$;	*Verschlüsselung des Klartextes*
$(\overrightarrow{z}, s_2)$	$\leftarrow$	$A_{2,q}^{\mathcal{O}_2}(p_0, p_1, s_1, z)$	*2. Stufe von A (Entscheiden)*
$\overrightarrow{p}$	$\leftarrow$	$\mathcal{O}_{d'}(\overrightarrow{z})$;	*Entschlüsselung der Ciphertexte $\overrightarrow{z}$*
o	$\leftarrow$	$A_{2,g}^{\mathcal{O}_2}(\overrightarrow{p}, s_2)$	*2. Stufe von A (Raten)*

mit:

$\mathcal{O}_1(\cdot) = \epsilon$	*und*	$\mathcal{O}_2 = \epsilon$	*für atk=pa0*
$\mathcal{O}_1(\cdot) = \mathcal{O}_d(\cdot)$	*und*	$\mathcal{O}_2 = \epsilon$	*für atk=pa1*
$\mathcal{O}_1(\cdot) = \mathcal{O}_d(\cdot)$	*und*	$\mathcal{O}_2 = \mathcal{O}_d(\cdot)$	*für atk=pa2*

Wie gehabt, gehen wir davon aus, dass der Angreifer zwei Klartexte derselben Länge wählt: $|p_0| = |p_1|$. Im Vektor $\overrightarrow{z}$ der Ciphertexte, die das Orakel $\mathcal{O}_{d'}$ entschlüsselt, darf z nicht enthalten sein; außerdem fordern wir im Fall von PA2, dass der Angreifer das Orakel $\mathcal{O}_2$ nicht nach der Entschlüsselung von z fragt.
Ist nun die Chance $\boldsymbol{Adv}^{ind-atk}_{A,\Pi}(k)$ eines polynomiell beschränkten Angreifers vernachlässigbar, so nennen wir das Kryptosystem Π sicher im Sinne von IND-ATK.

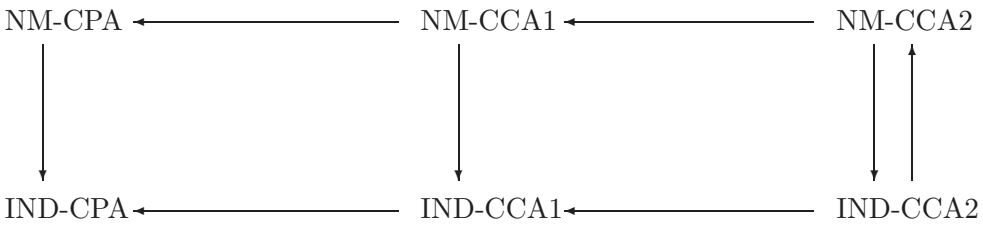

Abbildung 3.1: Beziehungen der Sicherheitsmodelle. Ein Pfeil von $\mathcal{SM}_1$ zu $\mathcal{SM}_2$ im Diagramm existiert genau dann, wenn $\mathcal{SM}_1 \Rightarrow \mathcal{SM}_2$ gilt.

Äquivalenzen: Bellare und Sahai konnten durch einen Ringschluss die Äquivalenz der drei Sicherheitsmodelle SMN, CMN und IND-PAx zeigen. Genauer gesagt zeigten sie, CNM-{CPA, CCA1, CCA2} impliziert SNM-{CPA, CCA1, CCA2}, aus SNM-{CPA, CCA1, CCA2} folgt IND-{PA0, PA1, PA2} und aus IND-{PA0, PA1, PA2} lässt sich CNM-{CPA, CCA1, CCA2} folgern. Die jeweiligen Implikationen finden sich in [BS99].
Auch bei der Non-Malleability gingen Baudron, Pointcheval und Stern der Frage nach, ob ein Angreifer Vorteile davon hat, wenn er mehrere öffentliche Schlüssel besitzt ([BPS00]). Indem sie die Äquivalenz zwischen Comparison Based Non-Malleability und der von ihnen definierten Multi-User-Non-Malleability bewiesen, konnten sie zeigen, dass dies nicht der Fall ist.

3.4.3 Beziehungen zwischen IND-ATK und NM-ATK

Wir haben bereits bei der Definition von Indistinguishability und Non-Malleability die Idee aus [BDPR98] bzw. [BDPR01] von Bellare, Desai, Pointcheval und Rogaway benutzt, die vorschlagen, die Angriffe auf Kryptosysteme von den Zielen losgelöst zu betrachten und jeweils die paarweisen Kombinationen als ein Sicherheitsmodell zu betrachten. Sie komplettierten ausserdem die bestehenden Beziehungen der Sicherheitsmodelle untereinander zu dem in Abbildung 3.1 gezeigten Graph. Die einzelnen Beweise finden sich in den eben genannten Dokumenten, wir wollen hier vorerst nicht näher darauf eingehen, weil wir durch das Ergebnis aus den Äquivalenzbeweisen von IND-{PA0, PA1, PA2} und NM-{CPA, CCA1, CCA2} den Graph aus Abbildung 3.1 – zu dem in Abbildung 3.2 gezeigten Graph – vereinfachen können und die entsprechenden Beweise durch die Äquivalenzen deutlich leichter werden. Wir sehen leicht, dass die Implikationen aus Abbildung 3.2 direkt aus den Definitionen folgen: Ist ein beliebiges Sicherheitsmodell in dem Graph erfüllt, so sind trivialerweise auch alle Sicherheitsmodelle erfüllt bei dem dem Angreifer weniger Fähigkeiten zugestanden werden. Es ist also lediglich noch zu zeigen, dass der Graph auch alle

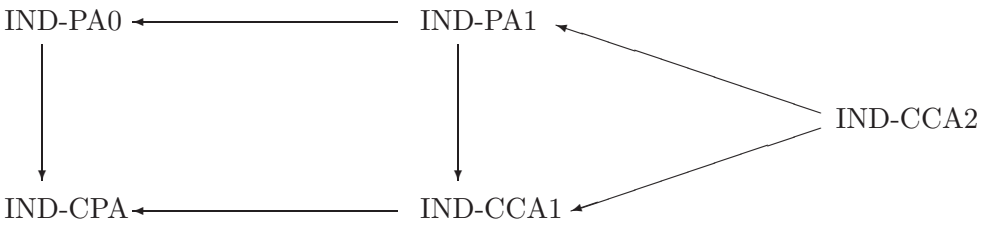

Abbildung 3.2: Einbeziehung des Ergebnisses von Bellare und Sahai. Ein Pfeil von $\mathcal{SM}_1$ zu $\mathcal{SM}_2$ im Diagramm existiert genau dann, wenn $\mathcal{SM}_1 \Rightarrow \mathcal{SM}_2$ gilt.

Implikationen enthält. Dazu wollen wir exemplarisch zeigen, dass NM-CPA nicht IND-CCA1 impliziert und verweisen für die weiteren Abgrenzungen auf [BDPR01] und [BS99].

Satz 3.23. *NM-CPA $\not\Rightarrow$ IND-CCA1 Falls es ein Kryptosystem Π gibt, das sicher im Sinne von NM-CPA ist, dann existiert ein Kryptosystem Π', das sicher im Sinne von NM-CPA, aber nicht sicher im Sinne von IND-CCA1 ist.*

Beweis. Man könnte auf den ersten Blick aus folgendem Grund annehmen, dass Sicherheit im Sinne von NM-CPA Sicherheit im Sinne von IND-CCA2[5] impliziert:
Um zu entscheiden, ob der Ciphertext z die Verschlüsselung des Klartextes p_0 oder p_1 ist, wenn man das Entschlüsselungsorakel $\mathcal{O}_d$ nicht nach z fragen darf, erscheint die folgende Strategie als einzige aussichtsreiche. Man ändert den Ciphertext z in einen – durch die Relation R in Zusammenhang stehenden – Ciphertext z' ab, fragt das Orakel $\mathcal{O}_d$ nach z' und erhält so Informationen über z, mit denen man eine Aussage treffen kann, welcher der beiden Klartexte p_0 und p_1 verschlüsselt wurde. Ist aber nun das entsprechende Kryptosystem non-malleable, so kann man keinen Ciphertext z' schaffen, der zum Ciphertext z in „bedeutungsvollem Zusammenhang“ steht.
Diese Argumentation ist aber trügerisch, da man, um zu entscheiden, ob der Ciphertext z die Verschlüsselung eines Klartextes p_0 oder p_1 ist, auch Informationen über den privaten Schlüssel d erlangen kann, so dass man selber in der Lage ist, den Ciphertext z zu entschlüsseln. Man muss also nicht zwangsläufig in „bedeutungsvollem Zusammenhang“ stehende Ciphertexte z' entschlüsseln, um Informationen über z zu erhalten.
Wir gehen im Folgenden davon aus, dass ein Kryptosystem Π existiert,

[5] und damit auch IND-CCA1

dass sicher im Sinne von NM-CPA ist – andernfalls hätten wir nichts mehr zu zeigen. Um zu zeigen, dass das Kryptosystem $\Pi' = (\mathcal{P}', \mathcal{C}', \mathcal{K}', \mathcal{G}', \mathcal{E}', \mathcal{D}')$ non-malleable, aber nicht sicher im Sinne von IND-CCA1 sein kann, werden wir das Kryptosystem $\Pi = (\mathcal{P}, \mathcal{C}, \mathcal{K}, \mathcal{G}, \mathcal{E}, \mathcal{D})$ derart modifizieren, dass es zwar non-malleable bleibt, jedoch nicht sicher im Sinne von IND-CCA1 ist. Dazu erweitern wir den Schlüsseltextraum um ein Element z^*, so dass $z^* \notin E_e(\mathcal{P})$ ist. Die Verschlüsselungsfunktion D'_d soll für $D'_d(z^*)$ den privaten Schlüssel d liefern und ansonsten wie D_d funktionieren. Hat nun der Angreifer ein Entschlüsselungsorakel $\mathcal{O}_d$, so kann er $\mathcal{O}_d$ nach z^* fragen und erhält somit den privaten Schlüssel, so dass Π' nicht sicher im Sinne von IND-CCA1 sein kann. Um nun aber nach wie vor zeigen zu können, dass unser neues Kryptosystem Π' sicher im Sinne von NM-CPA ist, müssen wir es etwas mehr ändern und beschreiben dazu die Algorithmen $\mathcal{G}'$, E' und D' von Π':

Algorithmus $\mathcal{G}'(k)$	Algorithmus $E'_{(e,u)}(p)$	Algorithmus $D'_{(d,u,v)}((b,z))$ mit $b \in \{0,1\}$
$(d,e) \xleftarrow{R} \mathcal{G}(k)$ $u, v \leftarrow \{0,1\}^k$ $e' \leftarrow (e,u)$ $d' \leftarrow (d,u,v)$ return (d', e')	$z \leftarrow E_e(p)$ return $(0,z)$	falls $b = 0$ return $D_d(z)$ falls $b = 1 \wedge z = u$ return v falls $b = 1 \wedge z = v$ return d sonst return $\perp$

Dahinter steckt die Idee, dass der private Schlüssel d nur als Antwort auf einen zufällig gewählten String v ausgegeben wird, wobei v wiederum nur als Antwort von D' auf den bekannten String u erhalten werden kann. Durch diese Weiterleitung bleibt dem Angreifer v verborgen, so dass das Kryptosystem intuitiv weiterhin sicher im Sinne von NM-CPA scheint, während es leicht zu zeigen ist, dass es nicht sicher im Sinne von IND-CCA1 sein kann:

Hilfsatz 3.24. *Das Kryptosystem Π' ist nicht sicher im Sinne von IND-CCA1.*

Beweis. Die erste Stufe A_1 des Algorithmus A bekommt den öffentlichen Schlüssel e, kann daraufhin das Orakel $\mathcal{O}_d$ nach der Entschlüsselung von u fragen und erhält so v. Indem A_1 sich nun vom Orakel $\mathcal{O}_d$ v entschlüsseln lässt, erhält er den privaten Schlüssel d. Damit kann er zwei beliebige Klartexte p_0 und p_1 wählen und als Statusinformation s den privaten Schlüssel d ausgeben. Für die zweite Stufe A_2 des Algorithmus ist es somit ein Leichtes zu entscheiden, welcher der beiden Klartexte p_0 und p_1 die Entschlüsselung von z ist. Offensichtlich laufen sowohl A_1 als auch A_2 in Polynomialzeit. □

Nun bleibt noch zu zeigen, dass das neue Kryptosystem Π' immer noch sicher im Sinne von NM-CPA ist.

Hilfsatz 3.25. *Das Kryptosystem Π' ist sicher im Sinne von NM-CPA.*

Für den technischen Beweis dieses Hilfsatzes verweisen wir auf die Veröffentlichung von [BDPR01] von Bellare, Desai, Pointcheval und Rogaway. Damit ist gezeigt, dass das Kryptosystem Π' die gesuchten Anforderungen erfüllt. ■

Kapitel 4

Gitter und Gitterprobleme

Im folgenden Kapitel betrachten wir einige Grundlagen der diskreten Geometrie, die uns in den weiteren Kapiteln nützliche Dienste erweisen werden. Eine ausführliche Darstellung der diskreten Geometrie kann und soll hier nicht erfolgen. Sie findet sich in zahlreichen Werken, wie bspw. [MG02] und [NW99]. Wir werden uns im Folgenden weitestgehend an die Definitionen und Notationen von Ajtai und Dwork ([AD97]) halten.
Außerdem stellen wir einige Gitterprobleme vor, so z. B. das Shortest-Vector-Problem, das Closest-Vector-Problem und inbesondere das für das Ajtai-Dwork-Kryptosystem wichtige unique Shortest-Vector-Problem.
Wir wollen dabei den $\mathbb{R}$-Vektorraum mit dem üblichen Skalarprodukt $a \cdot b$ zu Grunde legen.

4.1 Fakten und Notationen

Definition 4.1. *Ein* Gitter *im* $\mathbb{R}^n$ *ist eine diskrete additive Untergruppe des* $\mathbb{R}^n$ *der Form*

$$L = L(b_1, \ldots, b_n) = \{\sum_{i=1}^{n} \lambda_i b_i | \lambda_i \in \mathbb{Z}, i = 1, ..., n\},$$

wobei $b_1, \ldots, b_n$ *eine Basis im* $\mathbb{R}^n$ *ist.* $(b_1, \ldots, b_n)$ *bezeichnen wir dann auch als* Basis *von* L. *Sei* $B = [b_1, \ldots, b_n]$ *die Matrix mit den Spaltenvektoren* $b_1, \ldots, b_n$, *dann nennen wir* B *eine* Basismatrix *von* L. *Die* Dimension *oder der* Rang *des Gitters ist* n.

Definition 4.2. *Sei* B *eine nichtsinguläre* $n \times n$*-Matrix, dann ist* B *in* Hermite-Normalform, *wenn gilt:*

1. B *ist untere Dreiecksmatrix, d. h.* $b_{ij} = 0$ *für* $i < j$;
2. $b_{ii} > 0$ *für* $i = 1, \ldots, n$ *und*
3. $0 \leq a_{ij} < a_{ii}$ *für* $j < i$.

Satz 4.3. *Die Hermite-Normalform einer Basis ist eindeutig bestimmt.*

Bemerkung 4.4. Zu jedem Gitter L gibt es unendlich viele Basen, da die Wahl einer Basis für L nicht eindeutig ist. Seien $GL_n(\mathbb{Z})$ die multiplikative Gruppe der ganzzahligen unimodularen Transformationen in $\mathbb{Z}^n$ mit Determinate ± 1 und $[b_1, \ldots, b_n]$ bzw. $[b'_1, \ldots, b'_n]$ zwei Basismatrizen. Dann gilt $L(b_1, \ldots, b_n) = L(b'_1, \ldots, b'_n)$ genau dann, wenn es eine Matrix $T \in GL_n$ gibt, so dass $[b_1, \ldots, b_n] = [b'_1, \ldots, b'_n] \cdot T$.
Es gibt allerdings wegen Satz 4.3 nur eine Darstellung der Basis in Hermite-Normalform.[1]

Definition 4.5. *Als* Länge $\|b\|$ eines Vektors $b = (b_1, \ldots, b_n)^T$ *wollen wir* $(b_1^2 + \cdots + b_n^2)$ *definieren. Die* Länge einer Basis B *soll die Länge des längsten Vektors der Basis sein:* $\max_{i=1}^{n} \|b_i\|$.

Bemerkung 4.6. Die Definition der Länge weicht von der weitaus üblicheren euklidischen Norm ($\|b\| = \sqrt{(b_1^2 + \cdots + b_n^2)}$) ab und erfüllt auch nicht alle Anforderungen einer Norm.[2] Wir halten uns hier an die Definition aus [AD97], was uns in Kapitel 5 bei der Beschreibung des Ajtai-Dwork-Kryptosystems lästiges Umrechnen erspart. Dies ist zulässig, da diese Form der Längenberechnung für komplexitätstheoretische Zwecke besser geeignet ist und wir ansonsten in den meisten Fällen nur Näherungswerte berechnen könnten.

[1] Es gibt jedoch verschiedene Definitionen der Hermite-Normalform, die in Punkt 3 von unserer Definition abweichen. Da sich diese leicht ineinander ueberführen lassen, sind sie für unsere Betrachtungen äquivalent.

[2] Es gilt nicht $\|\lambda \cdot b\| = |\lambda| \cdot \|b\|$ für alle $\lambda \in \mathbb{R}$ und $b \in \mathbb{R}^n$.

Definition 4.7. *Seien $a_1, \ldots, a_n \in \mathbb{R}^n$ linear unabhängige Vektoren, dann bezeichen wir mit $\mathcal{P}^-(a_1, \ldots, a_n)$ das von $a_1, \ldots, a_n$ aufgespannte, halboffene* Parallelepiped[3]

$$\mathcal{P}^-(a_1, \ldots, a_n) = \{\sum_{i=1}^{n} \gamma_i a_i | 0 \leq \gamma_i < 1, i = 1, ..., n\}.$$

Geht aus dem Kontext hervor, um welche Vektoren a_i es sich handelt, wollen wir der Übersichtlichkeit halber $\mathcal{P}^-(a_1, \ldots, a_n)$ mit $\mathcal{P}'$ bezeichnen. Mit „reduziert modulo $\mathcal{P}'$ "wollen wir die Reduktion modulo $(a_1, \ldots, a_n)$ in das Gitter-Parallelepiped $\mathcal{P}'$ bezeichnen.

Definition 4.8. *Als* Grundmasche *eines Gitters L mit Basis $b_1, \ldots, b_n$ wollen wir das Parallelepiped $\mathcal{P}^-(b_1, \ldots, b_n)$ bezeichnen.*

Definition 4.9. *Seien $a_1, \ldots, a_n \in \mathbb{R}^n$, dann definieren wir die* Breite *$width(a_1, \ldots, a_n)$* des Parallelepiped *$\mathcal{P}^-(a_1, \ldots, a_n)$ als das Maximum der Entfernungen zwischen einem Eckpunkt a_i (mit i=1,. . . ,n) des Parallelepiped $\mathcal{P}^-$ und dem Unterraum, der durch die Vektoren $\{a_j | i \neq j\}$ aufgespannt wird.*

Definition 4.10. *Die* Gitterdeterminante *eines Gitters L mit einer Basis $b_1, \ldots, b_n$ hat den absoluten Wert der Determinante des Parallelepipeds mit den Seiten $b_1, \ldots, b_n$.*

$$\det(L) \stackrel{def}{=} |det(b_1, \ldots, b_n)|$$

Satz 4.11. *Die Determinante eines Gitters ist von der Basiswahl unabhängig.*

Beweis. Da sich zwei Basismatrizen B und B' nur durch eine unimodulare ganzzahlige Transformation T mit Determinante ± 1 unterscheiden, gilt nach dem Determinanten-Multiplikationssatz:

$$|detB| = |det(B'T)| = |detB' detT| = |detB'|$$

□

Definition 4.12. *Seien L, $L' \subseteq \mathbb{R}^n$ Gitter, dann heißt L'* Untergitter von *L, genau dann wenn $L' \subseteq L$.*

Definition 4.13. *Untergitter von $\mathbb{Z}^n$ heißen* ganzzahlig.

Definition 4.14. *Das zum Gitter L* duale Gitter *L^* definieren wir wie folgt:*

$$L^* = \{x \in \mathbb{R}^n | x^T y \in \mathbb{Z} \text{ für alle } y \in L\}.$$

[3] Andere gebräuchliche Namen für Parallelepiped sind Parallelotop oder Parallelflach.

Zwischen den Gitterbasen der Gitter L und L^* besteht folgende Beziehung: Ist $(b_1, \ldots, b_n)$ eine Basis von L, dann ist $(a_1, \ldots, a_n)$ eine Basis für L^* mit:

$$a_i^T b_i = \begin{cases} 1 & \text{wenn } i = j \\ 0 & \text{wenn } i \neq j \end{cases}$$

Die zu $B = (b_1, \ldots, b_n)$ gehörige Basis des dualen Gitters L^* bezeichnen wir als zu B duale Basis $B^* = (b_1^*, \ldots, b_n^*)$.

Bemerkung 4.15. Für die beiden Basismatrizen B und B^* gilt:

$$B^* = \left(B^{-1}\right)^T$$

Definition 4.16. *Sei $n \in \mathbb{N}$, $M, d \in \mathbb{R}^+$ und $L \subseteq \mathbb{Z}^n$ ein ganzzahliges Gitter, das ein $(n-1)$-dimensionales Untergitter L' mit folgenden Eigenschaften besitzt:*

1. *L' besitzt eine Basis, die höchstens die Länge M hat.*

2. *H sei der $(n-1)$-dimensionale Unterraum von $\mathbb{R}^n$, der L' enthält und $H' \neq H$ sei eine Nebenklasse von H, die L schneidet. Dann ist der Abstand zwischen H und H' mindestens d.*

Dann bezeichnen wir L als ein (d,M)-Gitter. *Ist $d > M$, so ist L' eindeutig bestimmt und wir schreiben $L^{(d,m)}$ für L'. Den minimalen Abstand zwischen H und einer Nebenklasse von H, die L schneidet, bezeichnen wir mit d_L.*

4.2 Gitterprobleme

4.2.1 Suchen von Gitterpunkten in einem Würfel

In [Ajt96] gibt Ajtai eine Prozedur an, mit der man Gitterpunkte gleichverteilt in einem Würfel zufällig wählen kann. Dies wollen wir uns kurz anschauen, da wir das entsprechende Verfahren in Kapitel 5 benötigen werden.

Satz 4.17. *Es sei $L \subseteq \mathbb{Z}^n$ ein Gitter, das durch eine Basis $B = (b_1, \ldots, b_n)$ repräsentiert wird. Dann gibt es eine reele Konstante c_1, so dass für alle $c \geq c_1$ eine Konstante c_2 existiert, so dass es einen polynomiell beschränkten Algorithmus gibt, der – mit L als Eingabe – zufällige Gitterpunkte in einem Würfel $KU^{(n)}$ in $\mathbb{R}^n$ mit Kantenlänge $K = n^c \|B\|$ ausgibt, deren Verteilung höchstens $2^{-n^{c_2}}$ von der Gleichverteilung entfernt ist.*

Beweisskizze: Wir suchen einen Parallelepiped $\mathcal{P}$, dessen Ecken Gitterpunkte aus L sind, so dass der Würfel $KU^{(n)}$ in dem Parallelepiped $\mathcal{P}$ enthalten ist und die Anzahl der Punkte in $\mathcal{P}$ die Anzahl der Punkte in dem

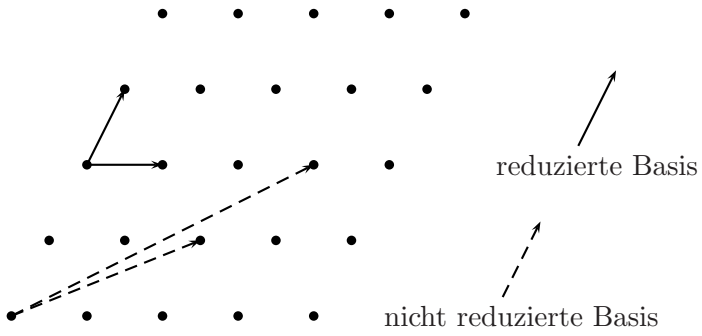

Abbildung 4.1: Ziel der Gitterbasisreduktion

Würfel $KU^{(n)}$ höchstens polynomiell übersteigt. Nun können wir einfach zufällige Punkte in $\mathcal{P}$ wählen, wobei wir diejenigen verwerfen, die nicht in $KU^{(n)}$ liegen. □

Diese Beweisskizze soll uns hier genügen. Für den ausführlichen Beweis einschließlich der entsprechenden Details verweisen wir auf [Ajt96].

4.2.2 Gitterbasisreduktion

Bei der *Gitterbasisreduktion* wird versucht eine möglichst interessante Darstellung des Gitters zu finden. Der Begriff Reduktion beschreibt die Tatsache, dass man eine Basis mit kurzen Vektoren finden möchte, d. h. die Länge der Basisvektoren soll reduziert werden. Außerdem ist man daran interessiert, dass die Basisvektoren möglichst orthogonal zueinander stehen. Die Definition von *reduzierten Basen* ist dabei nicht eindeutig, so gibt es unter anderem Minkowski-reduzierte ([Min11]), Hermite-Korkin-Solotarew-reduzierte ([Her50], [KS72], [KS73]) und Lenstra-Lenstra-Lovász-reduzierte Basen ([LLL82]). Wir wollen hier jedoch ebensowenig auf die Unterschiede eingehen, wie wir konkrete Algorithmen zur Gitterbasisreduktion betrachten wollen. Uns soll hier genügen eine ungefähre Vorstellung des Ergebnisses dieser Algorithmen zu haben, das wir schematisch in Abbildung 4.1 veranschaulichen.

4.2.3 Closest-Vector-Problem

Das *Closest-Vector-Problem* (*CVP*) besteht darin, bei gegebenem Gitter L einen Gittervektor in L zu finden, der einem gegebenen Vektor a am nächsten liegt. Das Closest-Vector-Problem ist NP-hart. Details dazu finden sich bspw. in [Cai99] und [Cai00]. Desweiteren zeigen Dinur, Kindler und Safra in [DKS98], dass die Approximierung des Closest-Vector-Problems

bis zu einem quasipolynomiellen Faktor von $2^{(\log n)^{1-\varepsilon}}$ – mit $\varepsilon = (\log \log n)^{-c}$ für jede Konstante $c < \frac{1}{2}$ – ebenfalls NP-hart ist.

4.2.4 Shortest-Vector-Problem

Im Jahre 1842 wurde das *Shortest-Vector-Problem* (SVP) erstmals von Dirichlet formuliert. Sei eine Basis B des Gitters $L \in \mathbb{Q}^n$ gegeben, dann finde einen vom Nullvektor verschiedenen Minimalvektor in L.
Ajtai zeigt in [Ajt98] die NP-Härte des Shortest-Vector-Problems unter randomisierter Reduktion. Ob das Shortest-Vector-Problem unter deterministischen Reduktionen ebenfalls NP-hart ist, ist ein offenes Problem der Gittertheorie.
Des Weiteren konnte gezeigt werden, dass das Shortest-Vector-Problem nicht schwieriger approximierbar ist als das Closest-Vector-Problem. Außerdem bewies Ajtai, dass die Berechnung einer Approximation von $1 + 1/2^{d^\varepsilon}$ des Shortest-Vector-Problems NP-hart ist. Im Folgenden konnte dieser Faktor von Cai und Nerurkar auf $1 + 1/d^\varepsilon$ und später von Micciancio auf $\sqrt{2}$ gehoben werden.
Der erste Algorithmus in Polynomialzeit für die Approximation des Shortest-Vector-Problems war der LLL-Algorithmus von Lenstra, Lenstra und Lovász ([LLL82]), der eine $2^{n/2}$-Approximation des SVP berechnet. In der Folge wurde der LLL-Algorithmus hauptsächlich durch Schnorr mehrfach verbessert ([Sch87], [Sch88], [SE91]). Schnorr entwickelte eine ganze Hierarchie von Algorithmen, die durch eine variable Schrittzahl Kompromisse zwischen Approximationsfaktor und Laufzeit bietet. So bietet der *Sampling-Reduction* genannte Algorithmus für eine Schrittzahl k mit $2 \leq k \leq \frac{n}{2}$ eine $(\frac{k}{6})^{n/2k}$-Approximation in der asymptotischen Laufzeit $O(n^3(\frac{k}{6})^{k/4} + n^4)$ ([Sch03]). Benutzt man Grovers Quantum-Search-Algorithmus ([Gro96]), so lässt sich – mit Hilfe von Quantencomputern – die asymptotische Laufzeit auf $O(n^3(\frac{k}{6})^{k/8} + n^4)$ senken ([Lud03]).
Für die exakte Berechnung des Shortest-Vector-Problems existiert ein Algorithmus von Kannan ([Kan87]), der von Helfrich ([Hel85]) verbessert wurde, aber eine exponentielle Laufzeit besitzt.

4.2.5 Unique Shortest-Vector-Problem

Das *unique Shortest-Vector-Problem* ($uSVP$) ist eine Variante des Shortest-Vector-Problems. Wiederum ist bei einem gegebenen Gitter der kürzeste, vom Nullvektor verschiedene Vektor v zu finden. Dieser ist jedoch eindeutig in dem Sinne, dass alle anderen Vektoren, die eine Maximallänge von $n^c\|v\|$ haben, parallel zu ihm sind. Da sich das n^c-unique Shortest-Vector-Problem mit fallendem c der Problemstellung des Shortest-Vector-Problems immer

weiter annähert, ist der Exponent c für die Schwierigkeit entscheidend. Je größer der Exponent c ist, desto kleiner ist die Klasse der Gitter, die es „zu durchsuchen“ gilt. Damit nimmt die Schwierigkeit des n^c-unique Shortest-Vector-Problems mit fallendem c zu.

4.2.6 Hidden-Hyperplane-Assumption

Sei $5 < c \in \mathbb{R}$ und $\mathcal{L}$ eine Verteilung auf der Menge der (d,M)-Gitter, für die $d > n^c M$ und $d \leq d_L \leq 2d$ gilt. Dann wollen wir als *Hidden-Hyperplane-Assumption* die Annahme bezeichnen, dass es keinen Algorithmus gibt, der mit einer Basis eines zufällig gewählten (d,M)-Gitters aus $\mathcal{L}$ als Eingabe, das versteckte Gitter $L^{(d,M)}$ effizient berechnen kann.
Die Hidden-Hyperplane-Assumption hängt eng mit dem unique Shortest-Vector-Problem zusammen. Sei L ein (d, M)-Gitter, dann besitzt das zu L duale Gitter L^* einen kürzesten Vektor v mit einer Länge von $1/d_L \leq 1/d$. v ist eindeutig im Sinne des unique Shortest-Vector-Problems bis auf einen Faktor von n^c. Der Vector v ist orthogonal zu H, dem $(n-1)$-dimensionalen Unterraum von $\mathbb{R}^n$, der $L^{(d,M)}$ enthält. Ist $L^{(d,M)}$ und eine Lösung des Hidden-Hyperplane-Problems für L gegeben, so ist es leicht, v zu finden, so dass eine Lösung des unique Shortest-Vektor-Problems für L^* berechnet werden kann.

Kapitel 5

Das Ajtai-Dwork-Kryptosystem

In diesem Kapitel werden wir zuerst der grundlegenden Abhandlung von Ajtai und Dwork ([AD97]) folgen und ihr ursprüngliches Kryptosystem einschließlich der Beweisskizzen einiger Reduktionsbeweise vorstellen. Das Ajtai-Dwork-Kryptosystem erregte Aufsehen, da es das erste war, bei dem es gelang, für das dem Kryptosystem zu Grunde liegende Basisproblem eine Worst-Case- / Average-Case-Äquivalenz zu zeigen. Es ist allerdings ein probabilistisches Kryptosystem, das mit einer (geringen) Wahrscheinlichkeit Fehler bei der Verschlüsselung macht, deswegen wollen wir uns danach ansehen, wie Goldreich, Goldwasser und Halevi das Originalsystem abändern, um die Entschlüsselungsfehler zu beseitigen ([GGH97]). Den Abschluss des Kapitels bildet die Vorstellung eines Kryptosystems von Regev ([Reg03b]) dessen Sicherheit ebenfalls auf dem unique Shortest-Vector-Problem beruht.

Bevor wir uns den verschiedenen Varianten des Kryptosystems von AJTAI und DWORK widmen können, benötigen wir noch einige kleinere Hilfsmittel:

Bemerkung 5.1. Wir werden im Folgenden des öfteren Vektoren mit Gleichverteilung aus der n-dimensionalen Kugel $S^{(n)}(R)$ mit Radius R wählen müssen. Eine Möglichkeit solche Vektoren zu wählen ist – AJTAI und DWORK folgend ([AD97]) – induktiv eine Koordinate nach der anderen, beginnend mit der n-ten Koordinate, zu berechnen. Dabei betrachen wir folgende Wahrscheinlichkeitsdichte, die die Wahrscheinlichkeit die k-te Koordinate in einer k-dimensionalen Kugel mit Radius R_k zu wählen, angibt: Die Wahrscheinlichkeit, die k-te Koordinate mindestens mit einem Wert $\frac{x}{R_k}$ zu wählen sei $\int_x^1(\sqrt{1-x^2})^{k-1}dx/\int_0^1(\sqrt{1-x^2})^{k-1}dx)$. Dann setzen wir $R_n := R$ und berechnen, wenn wir die k-te Koordinate mit Wert x_k gewählt haben, die folgenden Koordinaten gleichverteilt in der $(k-1)$-dimensionalen Kugel mit dem Radius $R_{k-1} := R_k\sqrt{1-x_k^2}$.

Definition 5.2. *Als* Störung *oder* Rauschen *wollen wir folgende randomisierte Funktion betrachten: die Zufallsvariable* $pert(R,m)$[1] *sei die Summe von* m *unabhängig voneinander gewählten, in der Kugel mit dem Radius* R *um* 0 *gleichverteilten (siehe z. B. [LW92]) Vektoren.*

Definition 5.3. *Um reelle Zahlen im Rechner darstellen zu können, müssen wir eine geeignete* Rundung *definieren: Sei* $x \in \mathbb{R}$ *und* $\alpha > 0$*, dann ist* $round_\alpha(x) = i\alpha$*, wobei* $i = \max\{i \in \mathbb{Z} \mid i\alpha \leq x\}$*. Und falls* $x = \langle x_1, \ldots, x_n\rangle \in \mathbb{R}^n$ *dann soll* $round_\alpha(x) = \langle round_\alpha(x_1), \ldots, round_\alpha(x_n)\rangle$ *sein.*

Definition 5.4. *Seien* L *ein Gitter,* $K > 0$*,* $R > 0$ *reelle Zahlen und* $U^{(n)}$ *der Einheitswürfel im* $\mathbb{R}^n$*. Dann definieren wir* $\xi_{L,K,R}$ *wie folgt: Zuerst wählen wir einen zufälligen (Gitter-)Punkt* v *nach der Gleichverteilung aus* $KU^{(n)} \cap L$*. Dann berechnen wir eine Störung* w *mit* $w = pert(R,m)$*.* $\xi_{L,K,R}$ *ist dann die Summe von* v *und* w*.*
η_K *sei ein nach der Gleichverteilung zufälliger Punkt aus* $KU^{(n)}$*.*

Definition 5.5. *Mit* $\mathcal{K}(n)$ *wollen wir die Funktion* $2^n \log n$ *bezeichnen, um die Darstellung übersichtlicher und die Ergebnisse leicht anpassbar zu halten.* AJTAI *und* DWORK *erwähnen ausdrücklich, dass sie keinen Versuch unternommen haben, die Funktion* $\mathcal{K}(n) = 2^{n\log n}$ *zu optimieren, dieser Versuch soll hier auch nicht erfolgen.*

Definition 5.6. *Es seien* $u \in \mathbb{R}^n$*,* $0 < \|u\| \leq 1$*,* $R > 0$ *und* X *die Menge aller* $x \in \mathcal{K}(n)U^{(n)}$*, so dass* $x \cdot u \in \mathbb{Z}$ *ist. Wir wählen nun einen zufälligen Punkt* v *aus* X *und davon unabhängig eine Störung* $w = pert(R,m)$*. Dann ist* $\mathcal{H}_{u,R,m} = round_{2^{-n}}(v+w)$*.*

[1] von [engl.] pertubation – Rauschen, Störung

5.1 Beschränkte Variante

Im Folgenden gehen wir davon aus, dass die Parameter n, $m \geq 4n$ und M bereits gewählt sind. AJTAI und DWORK treffen keine Aussage über eine optimale Wahl, diese soll auch hier nicht erfolgen.

5.1.1 Funktionsweise des Kryptosystems

Algorithmus 5.7. Das Generieren der Schlüssel

1. Wähle zufällig ein $(n-1)$-dimensionales Gitter L' mit der Basis $(b_1, \ldots, b_{n-1})$, so dass $\|b_i\| \leq M$ ist. Mit H bezeichnen wir dann die $(n-1)$-dimensionale Hyperebene, die L' enthält.
2. Wähle $d \geq n^{c_{Alg}} M$ mit $c_{Alg} > 5$.
3. Wähle einen Zufallsvektor b_n der Entfernung $d \leq d_L \leq 2d$ von H.
4. Der *private Schlüssel* ist eine beliebige Basis des Untergitters $L^{(d,M)}$ (oder gleichwertiger Weise dazu eine beliebige Basis für H).
5. Konstruiere eine zufällige Basis B‘ für $L = L(B)$. Dann ist der *öffentliche Schlüssel* $(B‘, M)$.

Bemerkung 5.8. Statt eine zufällige Basis B‘ für den öffentlichen Schlüssel zu konstruieren, die die Gefahr birgt, Fehler im Zufallsgenerator zu haben, so dass Schlüsse auf die Basis von $L^{(d,M)}$ möglich sind, kann es von Vorteil sein, eine Hermite-Normalform der Basis anzugeben. Nach NEMHAUSER, WOLSEY ([NW99]) und MICCIANCIO, WARINSCHI ([MW00]) lässt sich die Hermite-Normalform effizient berechnen. Da ein möglicher Angreifer ebenfalls in der Lage ist, die Hermite-Normalform, der ihm gegebenen Gitterbasis, zu bestimmen, bringt es ihm keinen Vorteil, wenn er die Basis direkt in dieser Art erhält (siehe dazu auch MICCIANCIO [Mic01]).

Algorithmus 5.9. Die Verschlüsselung
Die Verschlüsselung erfolgt bitweise: Um eine Null (Eins) zu verschlüsseln, wählen wir zufällig $\xi_{L,K,R}$ (η_K) wie in Definition 5.4 beschrieben mit $K \geq 2^n d$ und $R = n^3 M$. Es gilt also: $E(0) = \xi_{L,K,R}$ bzw. $E(1) = \eta_K$.

Algorithmus 5.10. Die Entschlüsselung
Sei u_H ein zur Hyperebene H orthogonaler Einheitsvektor und d_L der Abstand zwischen benachbarten Hyperebenen. Um den Schlüsseltext z zu entschlüsseln, berechnet der Empfänger den gebrochenen Anteil von $(u_H \cdot z)/d_L$. Ist er im Bereich $\frac{mR}{d_L}$ von 0 oder 1, wird z als Null entschlüsselt, ansonsten als Eins.

Bemerkung 5.11. Wir haben beim Generieren der Schlüssel in Algorithmus 5.7 als privaten Schlüssel eine Basis von $L^{(d,M)}$ bzw. H festgelegt. Damit ist es uns leicht möglich den orthogonalen Einheitsvektor u_H zu bestimmen.

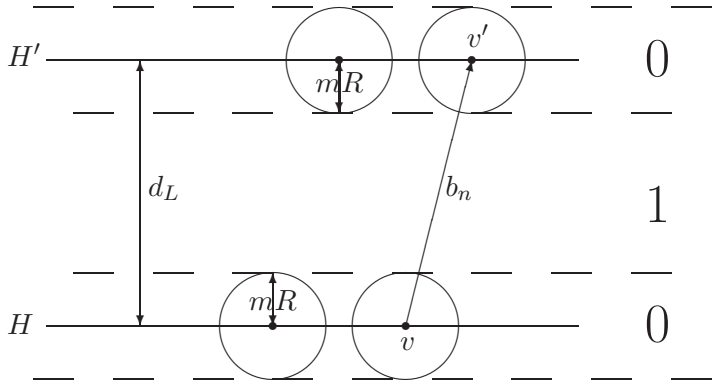

Abbildung 5.1: (Un)beschränktes AJTAI-DWORK-Kryptosystem

Auch den Abstand d_L zur nächsten Hyperebene können wir leicht bestimmen: Da die öffentliche Basis ein n-dimensionaler Raum, die private Basis aber eine $(n-1)$-dimensionale Hyperebene ist, müssen wir lediglich einen Vektor b_n der Dimension, die die beiden Räume unterscheidet, finden. Dieser muss zur nächsten Nebenklasse von $L^{(d,M)}$, die L schneidet, führen, womit wir seinen Abstand d_L zur Hyperebene H berechnen können.

Bemerkung 5.12. Wir sehen leicht, dass eine Null immer als Null entschlüsselt wird, wohingegen eine Eins fälschlicherweise als Null entschlüsselt werden kann, wenn man bei der Verschlüsselung zufällig einen Punkt nahe bei einer der Ebenen H' erhält. Die Wahrscheinlichkeit für einen Entschlüsselungsfehler ist: $\frac{2mR}{d_L} \leq 4n^{4-c_{Alg}}$, wie man aus Abbildung 5.1 ablesen kann. Wie man diese Verschlüsselungsfehler vermeiden kann, wird in Abschnitt 5.4.1 behandelt.

5.1.2 Reduktionsbeweis

Falls es ein Orakel $\mathcal{O}_{DIS}$ gibt, das die Verschlüsselungen von Null und Eins unterscheiden kann, so lässt sich der private Schlüssel (eine Basis des Gitters $L^{(d,m)}$ bzw. von H) mit Hilfe des Orakels bestimmen:

Satz 5.13. *Es seien n, d, M, K, R positive ganze Zahlen, die folgende Ungleichungen erfüllen:*

1. $\log d + \log M + \log K + \log R < n^{c_B}$,
2. $d > n^{c_1} M$,
3. $R > n^{c_2} M$,
4. $2^{c_3 n} d > K > 2^{c_4 n} d$.

$\mathcal{L}$ sei eine Verteilung von (d, M)-Gittern in $\mathbb{Z}^n$, deren Basislänge $n^{c_B} d_L$ nicht überschreitet und für die $d_L > n^5 M$ und $d \leq d_L \leq 2d$ gilt. $\mathcal{O}_{DIS}$ sei ein Orakel, das $\xi_{L,K,R}$ und η_K auf $\mathcal{L}$ mit einer Wahrscheinlichkeit von mindestens $\frac{1}{2} + n^{-c_{DIS}}$ unterscheidet. Dann existieren $c_1, c_2, c_3, c_4 > 0$, so dass für alle $c_B, c_{DIS} > 0$ ein c_{con} und ein Algorithmus $\mathcal{A}_{con}$ existiert, der – mit Hilfe des Orakels $\mathcal{O}_{DIS}$ – das Gitter $L^{(d,m)}$ aus $\mathcal{L}$ mit einer Wahrscheinlichkeit von mindestens $1 - 2^{-n}$ in der Zeit $n^{c_{con}}$ findet.

Bevor wir zum Beweis kommen, überzeugen wir uns noch kurz davon, dass die beschränkte Variante des AJTAI-DWORK-Kryptosystems die Anforderungen von Satz 5.13 erfüllt:

- Ungleichung 5.13.1 stellt sicher, dass d, M, K und R in polynomieller Zeit gelesen werden können, wovon wir auch ausgehen wollen, da wir diese Konstanten auch bei der Generierung des Schlüssels und bei der Verschlüsselung benötigen.

- Ungleichung 5.13.2 lässt sich wegen Bedingung 5.7.2 dadurch erfüllen, dass wir $c_1 < c_{Alg}$ wählen.

- Ungleichung 5.13.3 wird durch $c_2 < 3$ erfüllt, da wir bei der Verschlüsselung 5.9 $R = n^3 M$ wählen.

- Auch für Ungleichung 5.13.4 lassen sich wegen $K \geq 2^n d$ aus 5.9 geeignete Werte für c_3 und c_4 finden.

- Nach Schritt 1 von Algorithmus 5.7 gilt für alle $i \in \{1, \ldots, n-1\}$: $\|b_i\| \leq M$ und somit $\|b_i\| \leq d n^{-c_{Alg}} \leq d_L n^{-c_{Alg}}$ weswegen die Basislänge $n^{c_B} d_L$ nicht überschreitet.

- $d \leq d_L \leq 2d$ lässt sich unmittelbar aus 5.7.3 ablesen, weswegen mit 5.7.2 auch $d_L > n^5 M$ folgt.

Nachdem wir uns davon überzeugt haben, dass Algorithmus 5.7 alle notwendigen Voraussetzungen von Satz 5.13 erfüllt, können wir uns nun seines Beweises widmen, den wir durch Angabe des Algorithmus $\mathcal{A}_{con}$ führen:

Beweisskizze: Wir setzen $K' = n^{c_2} d_L$ und wählen polynomiell (in n) viele Gitterpunkte $(p_1, \ldots, p_{m'})$ zufällig in $K'U^{(n)}$. Hierbei verwenden wir die Annahme, dass die Basis $(b_1, \ldots b_n)$ von L durch Vektoren beschrieben wird, die höchstens polynomiell (in n) grösser als d sind, so dass wir den in Kapitel 4.2.1 vorgestellten Algorithmus nutzen können. Nun bilden wir die Differenzen sämtlicher gewählter Gitterpunkte und setzen $a_{ij} = p_i - p_j$ für $1 \leq i < j \leq m$. H sei die Hyperebene, in der $L^{(d,m)}$ liegt. Da die Abstände der Nebenklassen H' untereinander den Abstand d_L haben, schneiden maximal n^{c_2} Nebenklassen H' den Würfel $K'U^{(n)}$. Sei nun H' eine Nebenklasse von H deren Schnittmenge mit $K'U^{(n)} \cap L$ maximal ist,

dann beträgt die Anzahl der Differenzen a_{ij}, bei denen p_i und p_j beide in H' sind, mindestens $(\frac{1}{n^{c_2}})^2(\frac{m'(m'-1)}{2})$. Also ist ein polynomieller Bruchteil der Differenzen a_{ij} in H. Die Idee ist nun $\mathcal{O}_{DIS}$ zu nutzen, um herauszufinden, welche der Differenzen a_{ij} in H liegen und so – falls m' groß genug ist – eine Basis für H unter ihnen zu finden.
Wenden wir uns nun dem Test zu, wie wir mit Hilfe des Orakels $\mathcal{O}_{DIS}$ herausfinden können, welche der Differenzen a_{ij} in H enthalten sind:

Sei u eine Zufallsvariable mit Gleichverteilung gewählt aus $KU^{(n)} \cap L$, v eine der Differenzen a_{ij}, w die Zufallsvariable $pert(n^3M, m)$ und α eine Zufallsvariable mit Gleichverteilung aus dem Intervall $[0, 1]$. Wir bilden die neue Zufallsvariable $u + \alpha v + w$ und betrachen folgende Fallunterscheidung:

- Ist $av \in H$, dann ist $u + \alpha v \in H'$, wobei H' die Nebenklasse von H ist, die $u \in L$ enthält. In diesem Fall hat $u + \alpha v + w$ im Wesentlichen dieselbe Wahrscheinlichkeitsverteilung wie $\xi_{L,K,R}$. Die Entfernung von H' hängt nur vom Wert der Zufallsvariable w ab.
- Ist nun $v \notin H$ dann sind u und v mit exponentiell kleiner Wahrscheinlichkeit in verschiedenen Nebenklassen von H. Da nämlich u aus $KU^{(n)} \cap L$ und v aus $K'U^{(n)} \cap L$ gewählt wurde, gibt es mindestens $2^{c_4 n-1}$ Nebenklassen von H in denen u liegen kann[2], jedoch – wie wir bereits gesehen haben – maximal n^{c_2} Nebenklassen von H, in denen v liegen kann. Gehen wir also davon aus, dass u und v nicht in derselben Hyperebene H' liegen, dann ist die mit Vorzeichen versehene Entfernung von $u + \alpha v$ zur nächsten Nebenklasse von H auf $(-\frac{d_L}{2}; \frac{d_L}{2}]$ gleichverteilt. Damit hat $u + \alpha v + w$ im Wesentlichen dieselbe Verteilung wie η_K.

Somit können wir mit Hilfe des Orakels $\mathcal{O}_{DIS}$, das zwischen $\xi_{L,K,R}$ und η_K unterscheiden kann, herausfinden, ob v in H liegt oder nicht und damit eine Basis für die Hyperebene H (den privaten Schlüssel) finden. □

Diese Beweisskizze soll uns soweit genügen. Der ausführliche Beweis mit den exakten Wahrscheinlichkeitsverteilungen findet sich in [AD97].

5.2 Unbeschränkte Variante

5.2.1 Funktionsweise des Kryptosystems

Wir wollen nun die Einschränkung der Basislänge durch $n^{c_B} d_L$ entfernen. Da diese bei der Benutzung der beschränkten Variante des AJTAI-DWORK-

[2] Dies ergibt sich aus der Abschätzung $K > 2^{c_4 n} d \geq 2^{c_4 n-1} d_L$.

Kryptosystems nicht verwendet wurde, ändern sich unser Kryptosystem und dessen Algorithmen – bis auf das Entfallen der Beschränkung – nicht.

5.2.2 Reduktionsbeweis

Satz 5.14. *Es seien n, d, M, K, R positive ganze Zahlen, die folgende Ungleichungen erfüllen:*

1. $\log d + \log M + \log K + \log R < n^{c_B}$,
2. $n^{c_1} M > d > n^{c_2} M$,
3. $R > n^{c_3} M$,
4. $K > 2^{c_4 n} d$.

$\mathcal{L}$ sei eine Verteilung von (d, M)-Gittern in $\mathbb{Z}^n$, repräsentiert durch eine Basis, deren Vektoren in einem Würfel der Größe $2^{n^{c_B}} d$ liegen. $\mathcal{O}_{DIS}$ sei ein Orakel, das $\xi_{L,K,R}$ und η_K auf $\mathcal{L}$ mit einer Wahrscheinlichkeit von mindestens $\frac{1}{2} + n^{-c_{DIS}}$ unterscheidet. Dann existieren $c_1, c_2, c_3, c_4 > 0$, so dass für alle $c_B, c_{DIS} > 0$ ein c_{unc} und ein Algorithmus $\mathcal{A}_{unc}$ existiert, der – mit Hilfe des Orakels $\mathcal{O}_{DIS}$ – das Gitter $L^{(d,m)}$ aus $\mathcal{L}$ mit einer Wahrscheinlichkeit von mindestens $1 - 2^{-n}$ in der Zeit $n^{c_{unc}}$ findet.

Bemerkung 5.15. Da M nur eine obere Schranke der Länge von $L^{(d,M)}$ ist, schränkt die Bedingung, dass $n^{c_1} M > d$ ist, $\mathcal{L}$ nicht ein.

Da die Länge der Basis nun nicht mehr eingeschränkt ist, können wir somit auch keine Gitterpunkte mehr nach Satz 4.17 in einem Würfel suchen, wie wir dies bei der beschränkten Variante des Ajtai-Dwork-Kryptosystems getan haben. Wir können dieses Problem jedoch wie folgt lösen:
Wir suchen uns zufällige Vektoren, die nahe bei H liegen und verlängern diese mit Hilfe des Orakels $\mathcal{O}_{DIS}$ systematisch, so dass sie nach wie vor nahe bei H sind. Danach benutzen wir die so erhaltenen „langen Vektoren“, um H zu approximieren und so schliesslich den zu H orthogonalen Vektor u_H zu approximieren. Wenn die „langen Vektoren“ dicht genug an H waren, kann so der Orthogonalvektor zu H durch Runden der Approximation von u_H gefunden werden.

Beweisskizze: Mit σ wollen wir die Menge der Punkte in $\mathbb{R}^n$ mit einer Entfernung von maximal d zu H bezeichnen. Jede Iteration des Algorithmus hat als Startpunkt s, für den bei der ersten Iteration der Ursprung gewählt wird. $S^{(n)}(2\sqrt{n}d, s)$ sei die Kugel mit dem Radius $2\sqrt{n}d$ und Mittelpunkt s. Das Ziel ist nun einen Punkt v in $\sigma \cap S^{(n)}(2\sqrt{n}d, s)$ zu finden, der noch in σ liegt, aber weiter vom Ursprung entfernt ist als s. Dann nehmen wir v als neuen Startpunkt und starten die nächste Iteration.
Aber schauen wir uns zunächst einmal an, wie wir Punkte, die in σ liegen, von Punkten, die außerhalb von σ sind, auseinanderhalten können. Dazu

gehen wir ähnlich wie bei der Beweisskizze von Satz 5.13 vor:
u sei eine Zufallsvariable, gleichverteilt gewählt aus $KU^{(n)} \cap L$, $w = pert(n^3 M, m)$ und r ein nach der Gleichverteilung zufällig gewählter Punkt aus $KU^{(n)} \cap \mathbb{R}^n$, der den Abstand d_r zu einer L schneidenden Nebenklasse von $L^{(d,M)}$ hat. Wir erinnern uns, dass $\xi_{L,K,R} = u + w$ ist und setzen $\xi'_{L,K,R,z} = r + w$. Nun bilden wir die Zufallsvariable $\delta_v = u + \alpha v + w$ mit α gleichverteilt zufällig aus $[0, 1]$ und $v \in \mathbb{R}^n$ und betrachten folgenden Hilfssatz:

Hilfsatz 5.16. *Für alle $c_5 > 0$ existiert ein $c_6 > 0$ und $d_r \geq R/n^{c_6}$, derart dass*

1. *falls v einen kleineren Abstand als d_r zu H hat,*

$$|Pr[\mathcal{O}_{DIS}(\delta_v) = 1] - Pr[\mathcal{O}_{DIS}(\xi_{L,K,R})]| < \frac{1}{n^{c_5}}$$

   ist.

2. *falls v einen kleineren Abstand als d_r zu einer Nebenklasse $H' \neq H$ in L hat,*

$$|Pr[\mathcal{O}_{DIS}(\delta_v) = 1] - Pr[\mathcal{O}_{DIS}(\eta_K)]| < \frac{1}{n^{c_5}}$$

   ist.

Auf den detaillierten Beweis wollen wir hier verzichten und verweisen erneut auf [AD97], wir bemerken jedoch noch folgendes:

zu 1. Falls v innerhalb des Abstandes d_r von H ist, so auch αv. Damit hat $\delta_v = u + \alpha v + w$ im Wesentlichen dieselbe Verteilung wie $\xi'_{L,K,R,z}$, die nahe der Verteilung $\xi_{L,K,R}$ sein muss. Dies lässt sich durch Betrachten der Verteilung $\xi'_{L,K,R,0}$ und des Abstandes zu den Verteilungen $\xi_{L,K,R}$ und $\xi'_{L,K,R,z}$ zeigen.

zu 2. Falls v innerhalb des Abstandes d_r von $H' \neq H$ in L ist, ist der Abstand von $u + \alpha v$ zur nächsten L schneidenden Nebenklasse auf dem Intervall $[d_r, \frac{d}{2}]$ gleichverteilt – unter der Bedingung, dass αv nicht im Abstand d_r zu einer solchen Nebenklasse ist. Hat αv höchstens den Abstand d_r zu einer solchen Nebenklasse, dann hat $u + \alpha v + w$ im Wesentlichen dieselbe Verteilung wie $\xi'_{L,K,R,z}$. Daraus lässt sich – mit einigem Aufwand – zeigen, dass δ_v in etwa dieselbe Verteilung wie η_K hat.

Damit haben wir gezeigt, dass man mit Hilfe eines Orakels $\mathcal{O}_{DIS}$, das die Bedingungen aus Satz 5.14 erfüllt, zwischen Punkten, die nahe bei H sind und Punkten, die nahe bei einer Nebenklasse $H_1 \neq H$ in L sind, unterscheiden kann.

Haben wir passende Vektoren $v_1, \ldots, v_{i-1}$ mit mindestens der Länge l und höchstens Abstand d zu H gefunden, so suchen wir v_i im $(n-i+1)$-dimensionalen Untervektorraum V^{n-i+1} von $\mathbb{R}^n$, der orthogonal zu den bisher gefundenen Vektoren $v_1, \ldots, v_{i-1}$ ist, so dass v_i nahe bei $V^{n-i+1} \cap H$ ist. Nun bleibt noch anzugeben, wie man den Startpunkt für die nächste Iteration beim Suchen der „langen Vektoren“ findet:
Wähle einen zufälligen Vektor $v' \in S^{(n)}(2\sqrt{n}d)$, so dass $\|s+v'\| > \|s\|$. Um zu testen, ob $s+v'$ in σ liegt, testen wir jeden der Punkte $s+v'$, $s+\frac{n^{c_6}-1}{n^{c_6}}v'$, $\ldots$, $s+\frac{1}{n^{c_6}}v'$ durch die letzten Abschnitt beschriebene Prozedur. Sind einer oder mehrere Tests positiv, verwerfen wir v' und beginnen mit einem neuen v' von vorne. Sind alle Tests negativ, so berechnen wir die Länge $\|s+v'\|$. Ist sie kleiner als l, so ist unser nächster Startpunkt $s := 2(s+v')$. Ist die Länge von $s+v'$ größer als l, so setzen wir $v_i := s+v'$ und können uns der Berechnung von v_{i+1} zuwenden.
Durch den folgenden Satz aus [AD97] ist sichergestellt, dass der hier vorgestellte Algorithmus erfolgversprechend ist:

Hilfsatz 5.17. *Es existieren Konstanten $c_7, c_8, c_9 > 0$, so dass der eben beschriebene Algorithmus – von einem Startpunkt s, der höchstens den Abstand $2d$ zu H hat, gestartet – mit der Wahrscheinlichkeit $1 - n^{-c_7}$ und höchstens n^{c_8} Iterationen einen Punkt v ausgibt, der mit der Wahrscheinlichkeit $1 - n^{-c_9}$ in σ liegt.*

Damit haben wir gezeigt, wie man mit Hilfe des Orakels $\mathcal{O}_{DIS}$ das Gitter $L^{(d,m)}$ bestimmen kann. □

Auch hier finden sich die Details dieses Beweises, auf die wir hier verzichtet haben, in [AD97].

5.3 Hauptvariante

5.3.1 Funktionsweise des Kryptosystems

In der Hauptvariante des Ajtai-Dwork-Kryptosystems enthält der öffentliche Schlüssel kein Gitter mehr, sondern eine Menge zufällig gewählter Punkte in der Nähe der Hyperebene, die durch einen zufällig gewählten Vektor u – den privaten Schlüssel – erzeugt wird.

Algorithmus 5.18. Das Generieren der Schlüssel

1. Der private Schlüssel ist ein zufälliger Vektor u, gleichverteilt gewählt aus der Menge $\{x \in \mathbb{R}^n \mid \|x\| \leq 1\}$.

2. Der öffentliche Schlüssel besteht aus m unabhängig voneinander gewählten Werten von $\mathcal{H}_{u,R,m}$, die wir im Folgenden mit $v_1, \ldots, v_m$ bezeichnen wollen. Der öffentliche Schlüssel ist also eine Menge von

Werten v_i, die jeweils „verrauschte Gitterpunkte“ der durch u erzeugten Hyperebenen sind. Ausserdem benötigt der Sender einer Nachricht den kleinsten Index i_0, so dass $width(v_{i_0+1}, \ldots, v_{i_0+n})$ mindestens $n^{-2}\mathcal{K}(n)$ ist. Wir erinnern uns, dass wir $\mathcal{K}(n) = 2^{n \log n}$ gesetzt haben. Da der Wert von i_0 nicht von der zu verschlüsselnden Nachricht abhängt, können wir ihn als Teil des öffentlichen Schlüssels ansehen. Die so ausgezeichneten Vektoren wollen wir im Folgenden mit $w_1 = v_{i_0+1}, \ldots, w_n = v_{i_0+n}$ bezeichnen.

Algorithmus 5.19. Die Verschlüsselung
Die Verschlüsselung erfolgt bitweise:

- Um eine Null zu verschlüsseln, wählen wir m Zufallswerte $\delta_i, \ldots, \delta_m$ aus der Menge $\{0, 1\}$ (je mit der Wahrscheinlichkeit $\frac{1}{2}$) und berechnen den Vektor $z' = \sum_{j=1}^{m} \delta_i v_i$. Die Verschlüsselung z ist nun die Reduktion von z' modulo $w_1, \ldots, w_n$ in das Parallelepiped $\mathcal{P}^-(w_1, \ldots, w_n)$, d. h. der eindeutige Vektor z in $\mathcal{P}^-$, so dass $z' - z$ eine ganzzahlige Linearkombination der $w_1, \ldots, w_n$ ist.

- Um eine Eins zu verschlüsseln, wählen wir nach der Gleichverteilung einen zufälligen Vektor aus der Menge $\mathcal{P}^-(w_1, \ldots, w_n) \cap 2^{-n}\mathbb{Z}^n$, wobei wir mit $2^{-n}\mathbb{Z}^n$ alle Vektoren der Form $2^{-n}b$ mit $b \in \mathbb{Z}^n$ meinen.

Algorithmus 5.20. Die Entschlüsselung
Um den Schlüsseltext z zu entschlüsseln, berechnet der Empfänger den gebrochenen Anteil von $u \cdot z$. Ist er näher bei 0 oder 1 als $\frac{1}{n}$, wird z als Null entschlüsselt, ansonsten als Eins.

Bemerkung 5.21. Wie bereits bei der beschränkten und unbeschränkten Variante sehen wir auch hier leicht, dass eine Null immer als Null entschlüsselt wird, wohingegen eine Eins fälschlicherweise als Null entschlüsselt werden kann. Wir sehen sofort, dass dies mit einer Wahrscheinlichkeit von etwa $\frac{2}{n}$ der Fall ist.

5.3.2 Reduktionsbeweis

Satz 5.22. *Für alle $c_1, c_2, c_3, c_4 > 0$ existiert ein c_5 und ein probabilistischer Algorithmus $\mathcal{A}_{main}$, der ein Orakel $\mathcal{O}_{DIS}$ benutzt, so dass für alle hinreichend großen n, Bedingung (1) Bedingung (2) impliziert:*

(1) Sei $\mathcal{O}_{DIS}$ ein probabilistischer Schaltkreis[3] der Größe n^{c_1}, so dass – wenn $u, v_1, \ldots, v_m$ so wie in Algorithmus 5.18 beschrieben gewählt werden – mit einer Wahrscheinlichkeit von mindestens n^{c_1} folgendes gilt:

[3] Wir können hier den probabilistischen Schaltkreis mit Größe n^{c_1} als einen Orakelaufruf mit Kosten n^{c_1} betrachten.

$\mathcal{O}_{DIS}$ kann die Zufallsvariablen $\mathcal{S}_{v1,\ldots,v_m}$ und $\mathcal{E}_{v1,\ldots,v_m}$ mit Hilfe von $v_1, \ldots, v_m$ mindestens mit der Wahrscheinlichkeit von $\frac{1}{2} + n^{-c_3}$ unterscheiden.

(2) Der Algorithmus $\mathcal{A}_{main}$ kann, mit $\mathcal{O}_{DIS}$ als Orakel, jede beliebige Instanz mit einer maximalen Größe von n^{c_4} des n^8-unique Shortest-Vector-Problem in der Zeit n^{c_5} mindestens mit der Wahrscheinlichkeit $1 - 2^{-n}$ lösen.

Für den anspruchsvollen Beweis verweisen wir auf [AD97].

5.4 Variante nach Goldreich, Goldwasser und Halevi

Wir wir bereits in den vergangenen Abschnitten gesehen haben, gibt es in allen bisher vorgestellten Varianten des Ajtai-Dwork-Kryptosystems Verschlüsselungsfehler – wenn auch mit einer relativ geringen Wahrscheinlichkeit. Goldreich, Goldwasser und Halevi modifizierten nun die Hauptvariante des Ajtai-Dwork-Kryptosystems derart, dass keine Verschlüsselungsfehler mehr auftreten ([GGH97]). Ihnen gelang es außerdem, die Sicherheit des Kryptosystems auf die Schwierigkeit des n^7-unique Shortest-Vector-Problem zu verbessern.

5.4.1 Funktionsweise des Kryptosystems

Der Unterschied zur Hauptvariante des Ajtai-Dwork-Kryptosystems liegt darin, dass Klartexte von Eins nicht mehr zufällig verschlüsselt werden. Stattdessen wird die Ver- und Entschlüsselung so abgeändert, dass der gebrochene Anteil des Skalarproduktes $u \cdot z$ für Klartexte von Eins nun „nahe bei “ $\frac{1}{2}$ liegt – die Verschlüsselung von Klartexten von Null bleibt unverändert.

Algorithmus 5.23. Das Generieren der Schlüssel

1. Der private Schlüssel ist genau wie im Originalsystem ein zufälliger Vektor u, gleichverteilt gewählt aus der Menge $\{x \in \mathbb{R}^n \mid \|x\| \leq 1\}$.

2. Die m unabhängig voneinander erzeugten Werte von $v_1, \ldots, v_m$ werden – genau wie im Originalsystem – als Zufallswerte von $\mathcal{H}_{u,R,m}$ gewählt, d. h. jeder Vektor v_i ist die Summe aus einem Gitterpunkt v_i' und einer Störung w_i. Ebenso wird der kleinste Index i_0 wie im Originalsystem so gewählt, dass $width(v_{i_0+1}, \ldots, v_{i_0+n})$ mindestens $n^{-2}\mathcal{K}$ ist. Auch hier wollen wir die so ausgezeichneten Vektoren mit $w_1 = v_{i_0+1}, \ldots, w_n = v_{i_0+n}$ bezeichnen.
   Die Modifikation besteht nun darin, dass wir zusätzlich einen Index i_1 zufällig mit Gleichverteilung aus allen Indizes wählen, für die $u \cdot v_i'$ ungerade ist. Mit der Wahrscheinlichkeit von etwa $1 - 2^{-m}$ muss es

mindestens einen solchen Index geben.
Der öffentliche Schlüssel besteht nun aus den Vektoren $v_1, \ldots, v_m$, sowie den beiden Indizes i_0 und i_1.

Algorithmus 5.24. Die Verschlüsselung
Die Verschlüsselung erfolgt bitweise:

- Um eine Null zu verschlüsseln, wählen wir – wie gehabt – m Zufallswerte $\delta_i, \ldots, \delta_m$ aus der Menge $\{0, 1\}$ (je mit der Wahrscheinlichkeit $\frac{1}{2}$) und berechnen den Vektor $z' = \sum_{j=1}^{m} \delta_i v_i$. Die Verschlüsselung z ist nun die Reduktion von z' modulo $w_1, \ldots, w_n$ in das Parallelepiped $\mathcal{P}^-(w_1, \ldots, w_n)$, d. h. der eindeutige Vektor z in $\mathcal{P}^-$, so dass $z' - z$ eine ganzzahlige Linearkombination der $w_1, \ldots, w_n$ ist.

- Um eine Eins zu verschlüsseln, wählen wir – wie bei der Verschlüsselung von Null – m Zufallswerte $\delta_i, \ldots, \delta_m$ aus der Menge $\{0, 1\}$ (je mit der Wahrscheinlichkeit $\frac{1}{2}$) und berechnen den Vektor $z' = \sum_{j=1}^{m} \delta_i v_i$. Die Verschlüsselung z ist dann die Reduktion von $z' + \frac{1}{2} v_{i_1}$ modulo $w_1, \ldots, w_n$ in das Parallelepiped $\mathcal{P}^-(w_1, \ldots, w_n)$.

Algorithmus 5.25. Die Entschlüsselung
Um den Schlüsseltext z zu entschlüsseln, berechnet der Empfänger den gebrochenen Anteil von $u \cdot z$. Ist er näher bei 0 oder 1 als $\frac{1}{4}$, wird z als Null entschlüsselt, ansonsten als Eins.

5.4.2 Reduktionsbeweis

Goldreich, Goldwasser und Halevi zeigen in [GGH97], dass ihre Variante des Ajtai-Dwork-Kryptosystems sicher ist, solange das n^7-unique Shortest-Vector-Problem nicht effizient berechnet werden kann. Die Verbesserung der Sicherheit kommt dadurch zu Stande, dass durch das modifizierte System der Abstand zwischen Verschlüsselungen von Null und Eins größer wird. Dies kann man ausnutzen und den, durch die Störung *pert* verursachten, „Fehler“ bei der Erzeugung des öffentlichen Schlüssels derart vergrößern, dass sich die Sicherheit im beschriebenen Rahmen erhöht.

5.5 Alternatives Kryptosystem von Regev

In [Reg03b] zeigt Regev ein der Variante von Goldreich, Goldwasser und Halevi ähnliches Modell.

5.5.1 Funktionsweise des Kryptosystems

Algorithmus 5.26. Das Generieren der Schlüssel

1. Wähle eine große Zahl $N \in \mathbb{Z}$

2. Der private Schlüssel d besteht aus einer zufällig aus dem Bereich $[\sqrt{N}, 2\sqrt{N}[$ gewählten Zahl h.

3. Der öffentliche Schlüssel e besteht aus einer Menge von $m = O(\log N)$ Zahlen $a_1, \ldots, a_m$ aus dem Bereich $\{0, 1, \ldots, N-1\}$, die „nahe“ bei ganzzahligen Vielfachen von N/h sind[4]. Außerdem beinhaltet der öffentliche Schlüssel einen Index $i_0 \in \{1, 2, \ldots, m\}$, so dass a_{i_0} ein ungerades Vielfaches von N/h ist.

Algorithmus 5.27. Die Verschlüsselung
Die Verschlüsselung erfolgt bitweise:

- Um eine Null zu verschlüsseln bilden wir die Summe aus einer zufällig gewählten Teilmenge von $\{a_1, \ldots, a_m\}$ modulo N.

- Um eine Eins zu verschlüsseln bilden wir ebenso die Summe aus einer zufällig gewählten Teilmenge von $\{a_1, \ldots, a_m\}$ modulo N. Allerdings addieren wir noch den ganzzahligen Teil von $a_{i_0}/2$ zum Ergebnis.

Algorithmus 5.28. Die Entschlüsselung
Um den Ciphertext z zu entschlüsseln, betrachten wir den Rest der Divsion von $z/(N/h)$. Ist er klein, so entschlüsseln wir zu Null, ist er groß, so entschlüsseln wir zu Eins.
Da alle a_i aus $\{a_1, \ldots, a_m\}$ nahe bei ganzzahligen Vielfachen von N/h sind, muss auch jede Verschlüsselung von Null nahe bei einem ganzzahligen Vielfachen von N/h sein. Ebenso kann keine Verschlüsselung von Eins nahe bei einem ganzzahligen Vielfachen sein, da $a_{i_0}/2$ nicht nahe bei einem ganzzahligen Vielfachen ist.

5.5.2 Reduktionsbeweis

Regev zeigt in [Reg03b] eine Reduktion von der Unterscheidung zwischen einer Gleichverteilung und einer Verteilung nach obigen Algorithmen auf das $n^{\frac{3}{2}}$-unique Shortest-Vector-Problem. Er geht davon aus, dass diese Reduktion auch andere praktische Anwendungen in der Zukunft findet, da sie unter anderem eine Reduktion von n Dimensionen auf eine Dimension beinhaltet. Mit Hilfe dieser Reduktion konstruiert Regev außerdem eine Hash-Funktion, deren Sicherheit ebenfalls auf dem $n^{\frac{3}{2}}$-unique Shortest-Vector-Problem basiert. Wir wollen dies jedoch nicht vertiefen, verweisen für den Beweis auf [Reg03b] und wenden uns nun dem nächsten Kapitel zu – der Sicherheit des Ajtai-Dwork-Kryptosystems.

[4] Es ist nicht erforderlich, dass h ein Teiler von N ist.

Kapitel 6

Die Sicherheit des Ajtai-Dwork-Kryptosystems

Nachdem wir uns in den vorangegangenen Kapiteln die nötigen Kenntnisse angeeignet haben, können wir nun das Ajtai-Dwork-Kryptosystem daraufhin untersuchen, welche Sicherheitsmodelle es erfüllt. Dazu betrachten wir bei jeder der vier Varianten jeweils die Sicherheitsmodelle IND-CPA, NM-CPA und IND-CCA1. Wir können zeigen, dass keine der Varianten sicher im Sinne von NM-CPA und IND-CCA1 ist. Für die Sicherheit im Sinne von IND-CPA können wir zeigen, dass sie bei realitätsnahen Parametern nicht gegeben ist.
Zum Abschluß des Kapitels übetragen wir einen Angriff gegen das Ajtai-Dwork-Kryptosystem auf das Kryptosystem von Regev und zeigen, dass dieses nicht sicher im Sinne von NM-CPA ist.

6.1 Beschränkte Variante

6.1.1 IND-CPA

Aus Satz 5.13 wissen wir, dass die Sicherheit der beschränkten Variante des AJTAI-DWORK-Kryptosystems von der Hidden-Hyperplane-Assumption abhängt. Die Frage ist also, ob bzw. für welche Dimensionen sich die Hidden-Hyperplane-Assumption als haltbar erweist.
Wie wir in Abschnitt 4.2.6 festgestellt haben, müssen wir im schlechtesten Fall die Lösung des n^5-unique Shortest-Vector-Problems des dualen Gitters finden. Dazu wollen wir den in Abschnitt 4.2.4 angesprochenen Sampling-Reduction-Algorithmus verwenden. Wir wählen eine Schrittweite von $k = \frac{n}{10}$ und erhalten einen Approximationsfaktor von $(\frac{n}{60})^5$ mit Laufzeit $O(n^3(\frac{n}{60})^{n/40} + n^4)$. Zwar wächst die Laufzeit exponentiell, doch durch die kleinen Faktoren in Basis und Exponenten wächst die Funktion erst ab „großen“ Werten für n schnell.
Betrachten wir nun die Schlüssel- und Ciphertextgröße in Abhängigkeit der Dimension n. Für den öffentlichen Schlüssel müssen wir n Vektoren der Dimension n mit Maximallänge $n^{c_B} d_L$ speichern. Dazu benötigen wir $O(n^2 \log_2 n)$ Bits. Ein Bit wird verschlüsselt durch einen n-dimensionalen Vektor aus dem Würfel mit der Kantenlänge $K \geq 2^n d$ – also mit $O(n^2)$ Bits.
Nun müssen wir die Schlüssel- und Ciphertextgröße noch in Relation zu der Laufzeit des Gitterbasisreduktionsalgorithmus setzen und haben in Tabelle 6.1 einige Werte der angesprochenen Funktionen aufgelistet. Untersuchen wir die Funktionswerte näher, so sehen wir, dass der exponentielle Anteil der Laufzeit der Approximation erst ab einer Dimension von etwa 100 zu nimmt. Dabei stellen wir allerdings auch fest, dass wir eine Schlüsselgröße von etwa 66.000 Bit vielleicht noch hinnehmen könnten, die Länge der Ciphertexte jedoch unverhältnismässig groß wird. Für die Verschlüsselung eines Bits werden 10.000 Bits benötigt. Trotzdem erscheint im Zeitalter der Gigahertz-Rechner eine Laufzeit von 10^8 Operationen durchführbar, so dass wir den geheimen Schlüssel für Instanzen mit realitätsnahen Sicherheitsparametern berechnen können. Die beschränkte Variante des AJTAI-DWORK-Kryptosystems ist also *nicht sicher im Sinne von IND-CPA*. Da es allerdings Instanzen höherer Dimensionen gibt, für die wir den privaten Schlüssel nicht derart bestimmen können, wollen wir trotzdem noch die Sicherheit des AJTAI-DWORK-Kryptosystems im Sinne von NM-CPA und IND-CCA1 untersuchen.

6.1.2 IND-CCA1

Um einen non-adaptive Ciphertext-Angriff auf die Ununterscheidbarkeit der beschränkten Variante des AJTAI-DWORK-Kryptosystems durchzuführen,

n	$n^2 \log_2 n$	n^2	$(\frac{n}{60})^{n/40}$	$n^3(\frac{n}{60})^{n/40} + n^4$
5	58	25	0,73	$7,1 \cdot 10^2$
10	332	100	0,56	$1,0 \cdot 10^4$
25	2.902	625	0,58	$3,9 \cdot 10^5$
50	14.109	2500	0,80	$6,3 \cdot 10^6$
100	66.438	10.000	3,6	$1,0 \cdot 10^8$
250	497.861	62.500	7476	$1,2 \cdot 10^{11}$
500	2.241.446	250.000	$3,2 \cdot 10^{11}$	$4,0 \cdot 10^{19}$

Tabelle 6.1: Wachstum von Funktionen

nutzen wir die in Satz 5.13 aufgeführte Reduktion: Gibt es ein Orakel $\mathcal{O}_{DIS}$, das die Schlüsseltexte $\xi_{L,K,R}$ und η_K auf $\mathcal{L}$ mit einer Wahrscheinlichkeit von mindestens $\frac{1}{2}+n^{-c_{DIS}}$ unterscheidet, so existiert ein Algorithmus $\mathcal{A}_{con}$, der – mit Hilfe des Orakels $\mathcal{O}_{DIS}$ – das Gitter $L^{(d,m)}$ aus $\mathcal{L}$ mit einer Wahrscheinlichkeit von mindestens $1-2^{-n}$ in der Zeit $n^{c_{con}}$ findet. Nach Voraussetzung von IND-CCA1 hat die erste Stufe des Algorithmus A_1 Zugriff auf ein (perfektes) Orakel $\mathcal{O}_d$, das Ciphertexte (sicher) entschlüsseln und somit auch auseinander halten kann. Unser Angriff benutzt direkt die zugehörigen Beweisskizze:
Algorithmus A_1 erhält den öffentlichen Schlüssel e und berechnet daraus eine Basis für die geheime Hyperebene H, in der $L^{(d,m)}$ liegt, was – nach Satz 5.13 – mit einer Wahrscheinlichkeit von mindestens $1-2^{-n}$ in der Zeit $n^{c_{con}}$ – also Polynomialzeit – gelingt. Wir sind nun im Besitz der Basis von H und können somit auch einen zur Hyperebene H orthogonalen Einheitsvektor u_H sowie die Entfernung d_L zur nächsten L schneidenden Nebenklasse von H bestimmen.[1] Als Statusinformation s geben wir nun den Einheitsvektor u_H sowie die Entfernung d_L an A_2 weiter. Somit kann A_2 den Ciphertext z entschlüsseln und den zugehörigen Klartext p_b bestimmen. Da wir den privaten Schlüssel mit einer Wahrscheinlichkeit von mindestens $1-2^{-n}$ finden können, ist die beschränkte Variante des AJTAI-DWORK-Kryptosystems somit *nicht sicher im Sinne von IND-CCA1.*

6.1.3 NM-CPA / IND-PA0

Betrachten wir den Angriff im vorherigen Abschnitt etwas genauer, so stellen wir fest, dass wir ihn problemlos zu einem Angriff im Sinne von IND-PA0 ändern können. Wir erinnern uns, dass wir in der Beweisskizze zu Satz 5.13 das Orakel $\mathcal{O}_d$ benutzen, um herauszufinden, welche der Differenzen a_{ij} in H liegen. Da wir lediglich daran interessiert sind, die Differenzen zu finden, die in H liegen, hängen unsere Anfragen an das Orakel nicht voneinander ab. Dies erlaubt es uns, sie parallel durchzuführen: Statt nun das Orakel

[1] siehe dazu auch Bemerkung 5.11

$\mathcal{O}_d$ in der ersten Stufe des Algorithmus, A_1, zu benutzen, wählen wir die benötigten Abfragen in der zweiten Stufe des Algorithmus, $A_{2,q}$, aus und lassen den Vektor mit Ciphertexten $\overrightarrow{z}$ durch das Orakel $\mathcal{D}_d$ entschlüsseln. Der Entscheidungsteil der zweiten Stufe des Algorithmus, $A_{2,g}$, kann nun wie gehabt die geheime Basis finden, damit den Ciphertext z entschlüsseln und somit bestimmen, welcher der Klartexte p_0 und p_1 verschlüsselt wurde. Auch hier sind wir mit einer Wahrscheinlichkeit von mindestens $1-2^{-n}$ erfolgreich, so dass die beschränkte Variante des AJTAI-DWORK-Kryptosystems *nicht sicher im Sinne von IND-PA0* bzw. *NM-CPA* ist.

6.1.4 Zusammenfassung

Die beschränkte Variante des AJTAI-DWORK-Kryptosystems ist nicht sicher im Sinne von IND-CCA1, NM-CPA und stärkeren Sicherheitsmodellen. Es resultiert direkt aus dem Reduktionsbeweis, dass ein potentieller Angreifer, der in der Lage ist, Verschlüsselungen von Null und Eins auseinander zu halten, auch immer in der Lage ist, den privaten Schlüssel zu bestimmen. In den eben genannten Sicherheitsmodellen steht ihm jedoch immer ein Entschlüsselungsorakel zur Verfügung, so dass ihn das Unterscheiden der Verschlüsselungen vor keine großen Probleme stellt.
Da die Reduktion für diese Variante des Kryptosystems fundamental ist, ist eine „Reparatur“ durch einfache Modifikationen nicht möglich.
Die Sicherheit der beschränkten Variante des AJTAI-DWORK-Kryptosystems im Sinne von IND-CPA ist nur sehr bedingt gegeben. Verwendet man realistische Sicherheitsparameter, so hat ein Angreifer mit Gitterbasisreduktionsalgorithmen gute Chancen die geheime Hyperebene und damit den privaten Schlüssel zu berechnen.
Eine „Reparatur“ könnte also dadurch erfolgen, dass man die Effizienz steigert, um die Schlüssel- und Ciphertextlänge zu verkürzen und man somit Gitter höherer Dimension verwenden könnte. Ob dies mit einfachen Modifikationen möglich ist, erscheint jedoch fraglich.

6.2 Unbeschränkte Variante

6.2.1 IND-CPA

Ebenso wie bei der beschränkten Variante des AJTAI-DWORK-Kryptosystems können wir auch bei der unbeschränkten Variante Gitterbasisreduktionsalgorithmen anwenden, um die geheime Hyperebene zu finden. Der einzige Unterschied besteht darin, dass nun die Länge der Basisvektoren nicht mehr polynomiell beschränkt ist, sondern durch den Würfel der Größe $2^{n^{cB}}d$. Dies wiederum führt zu einer Vergrößerung des öffentlichen Schlüssels für dessen Speicherung wir nun $O(n^3)$ Bits benötigen. Andere Parameter ändern sich nicht. Auch bei dieser Variante

hängt die Durchführbarkeit der Gitterbasisreduktion jedoch direkt von der Dimension n des Gitters ab. Wie wir in Abschnitt 6.1.1 festgestellt haben, erscheint die Reduktion für realistische Sicherheitsparameter durchaus durchführbar.

6.2.2 IND-CCA1

Auch hier können wir uns den entsprechenden Reduktionsbeweis zu Nutze machen. Aus Satz 5.14 geht hervor, dass ein Algorithmus $\mathcal{A}_{unc}$ existiert, der – mit Hilfe eines Orakels $\mathcal{O}_{DIS}$ – das Gitter $L^{(d,m)}$ aus $\mathcal{L}$ mit einer Wahrscheinlichkeit von mindestens $1 - 2^{-n}$ in polynomieller Zeit findet. Wiederum hat die erste Stufe A_1 unseres Algorithmus Zugriff auf ein (perfektes) Orakel $\mathcal{O}_d$. Analog zum non-adaptive Ciphertext-Angriff auf die Ununterscheidbarkeit der beschränkten Variante nutzen wir in der ersten Stufe des Algorithmus, A_1, die Beweisidee zu Satz 5.14, um die geheime Hyperebene und somit den privaten Schlüssel zu finden. Wir bemerken allerdings, dass wir beim „Entwickeln der langen Vektoren“ Anfragen an das Orakel $\mathcal{O}_d$ stellen müssen, die vom Ergebnis vorheriger Anfragen abhängen. Die erste Stufe A_1 unseres Algorithmus kann also – mit Hilfe des Orakels $\mathcal{O}_d$ – das Gitter $L^{(d,m)}$ aus $\mathcal{L}$ mit einer Wahrscheinlichkeit von mindestens $1 - 2^{-n}$ in der Zeit $n^{c_{unc}}$ – also Polynomialzeit – finden. Damit kann A_1 den Orthogonalvektor u_H sowie den Abstand d_L zwischen den benachbarten Hyperebenen H und H' bestimmen und gibt diese Werte als Statusinformation s an die zweite Stufe des Algorithmus A_2 weiter. Somit kann A_2 – genau wie bei der beschränkten Variante – den Ciphertext z entschlüsseln und so auch p_b mindestens mit der Wahrscheinlichkeit $1 - 2^{-n}$ bestimmen. Damit ist die beschränkte Variante des Ajtai-Dwork-Kryptosystems *nicht sicher im Sinne von IND-CCA1.*

6.2.3 NM-CPA / IND-PA0

Da der Reduktionsbeweis für Satz 5.14 Anfragen an das Orakel $\mathcal{O}_d$ enthält, die zu vorherigen Anfragen in Abhängigkeit stehen, benötigen wir für IND-PA0 eine andere Attacke. Wir wollen dabei ausnutzen, dass Punkte, die nahe zusammen sind, wahrscheinlich Verschlüsselungen von demselben Klartext (Bit) sind.
Betrachtet man die Entschlüsselung genauer, so stellt man nämlich fest, dass alle Punkte im Abstand von mR von einer der Hyperebenen H' zu Null entschlüsselt werden und alle anderen Punkte zu 1 (siehe auch Abbildung 5.1). Dies kann man sich nun zu Nutze machen, indem man das Orakel nach benachbarten Punkten von z fragt. Stimmen in allen n Dimensionen die Entschlüsselungen der benachbarten Punkte von z überein, so muss z denselben Wert haben.
Alle wesentliche Arbeit wird in den beiden Teilen der zweiten Stufe $A_{2,q}$ und $A_{2,g}$ erfolgen; die erste Stufe des Algorithmus, A_1, muss nicht einmal

Statusinformation wie bspw. den Schlüssel weitergeben. Algorithmus $A_{2,q}$ erhält also die beiden Klartexte p_0 und p_1 sowie den Ciphertextvektor z. Sei nun $(e_1, \ldots, e_n)$ die kanonische Basis des $\mathbb{R}^n$, dann bilden wir den Vektor $\overrightarrow{z} = (z'_1, \ldots, z'_{2n})$ aus Ciphertexten z'_i wie folgt: $z'_i := z + \varepsilon e_i$ falls $i \leq n$ und $z'_i := z - \varepsilon e_{i-n}$ falls $i > n$.
Wobei für $\varepsilon > 0$ ein möglichst kleiner Wert wie bspw. die Rechengenauigkeit des Systems gewählt werden soll. Algorithmus $A_{2,g}$ erhält nun den Vektor aus Klartexten $\overrightarrow{p} = (p'_1, \ldots, p'_{2n})$ sowie die Statusinformation s_2, die die beiden Klartexte p_0 und p_1 enthält, als Eingabe. Betrachten wir nun den Quader Q_z mit den Eckpunkten z'_i und Mittelpunkt z und machen folgende Fallunterscheidung:

- Der zu z gehörige Klartext sei Null. Gehen wir nun o. B. d. A. davon aus, dass H' die Hyperebene sei, zu der z den maximalen Abstand mR hat und bezeichnen die Menge der Punkte, die den maximalen Abstand mR zu H' haben mit Q_0. Wir wollen nun die Wahrscheinlichkeit abschätzen, dass alle Punkte des Quaders Q_z in Q_0 liegen und gehen dabei der Einfachkeit halber von einer Gleichverteilung der Ciphertexte z in Q_0 aus[2]. Entscheidend ist nur die Dimension in Richtung von u_H, so dass wir zu folgender Wahrscheinlichkeit kommen:

$$Pr[Q_z \in Q_0 \mid z = E_e(0)] \geq \frac{2mR - 2\varepsilon}{2mR} = 1 - \frac{\varepsilon}{mR}$$

- Der zu z gehörige Klartext sei Eins. Gehen wir nun o. B. d. A. davon aus, dass z zwischen den Hyperebenen H' und H'' liegt und bezeichnen die Menge der Punkte zwischen den beiden Hyperebenen H' und H'', die zu jeder der beiden Hyperebenen mindestens den Abstand mR haben, mit Q_1. Ähnlich wie im Fall, dass der zu z gehörige Klartext Null ist, sind wir an der Wahrscheinlichkeit, dass alle Punkte des Quaders Q_z in Q_1 liegen, interessiert. Wir vernachlässigen nun den Verschlüsselungsfehler[3] und gehen davon aus, dass z in Q_1 gleichverteilt ist. Dann ist die gesuchte Wahrscheinlichkeit:

$$Pr[Q_z \in Q_1 \mid z = E_e(1)] \geq \frac{d_L - 2mR - 2\varepsilon}{d_L - 2mR} = 1 - \frac{2\varepsilon}{d_L - 2mR} \geq 1 - \frac{\varepsilon}{mR}$$

Das heißt mit mindestens der Wahrscheinlichkeit $1 - \frac{\varepsilon}{mR}$ werden alle Punkte des Quaders Q_z zu demselben Wert wie z entschlüsselt.
Sind also alle p'_i aus $\overrightarrow{p}$ gleich, so kann sich $A_{2,g}$ sicher sein, dass z die Verschlüsselung von p'_1 ist, und muss nur noch den passenden Klartext p_b

[2] Durch die genaue Wahrscheinlichkeitsverteilung von $pert(\cdot)$ sind die Punkte nahe der Ebene H' wahrscheinlicher als Punkte, die weiter von ihr entfernt sind. Dies hätte jedoch einen positiven Einfluss auf unsere Abschätzung, so dass wir dies getrost ignorieren können.

[3] Dies können wir ohne weiteres tun, da der Empfänger des Ciphertextes bei einem Verschlüsselungsfehler beim Entschlüsseln denselben Fehler machen würde.

bestimmen. Für den seltenen Fall, dass in $\overrightarrow{p}$ verschiedene Werte enthalten sind, gibt $A_{2,g}$ den Wert aus, den mindestens die Hälfte aller p'_i haben. A läuft in Polynomialzeit, der Vektor $\overrightarrow{z}$, der an das Orakel $\mathcal{O}_d$ geschickt wird, hat die Länge $2n$ und die Wahrscheinlichkeit, dass $A_{2,g}$ richtig liegt, beträgt mindestens $1-\frac{\varepsilon}{mR}$. Damit ist die unbeschränkte Variante des AJTAI-DWORK-Kryptosystems *nicht sicher im Sinne von IND-PA0* bzw. *NM-CPA*.

Bemerkung 6.1. Man kann auch, wenn man statt dem kompletten Quader Q_0 nur einen Eckpunkt z'_i von Q_0 betrachtet, mit einer Anfrage an das Orakel $\mathcal{D}_d$ auskommen. Ein Eckpunkt des Quaders hat in Richtung u_H maximal die Entfernung ε. Ist ε klein genug, so wird $z \pm \varepsilon e_i$ zu demselben Wert wie z entschlüsselt: $(u_H \cdot (z \pm \varepsilon e_i))/d_L$ kann wegen der Bilinearität des Skalarproduktes zu $(u_H \cdot z)/d_L + (u_H \cdot (\pm\varepsilon e_i))/d_L$ umgeformt werden. Nun ist $\mid u_H \cdot (\pm\varepsilon e_i) \mid < \varepsilon$, ist nun $\frac{\varepsilon}{d_L}$ klein und $(u_H \cdot z)$ nicht nahe bei mR, dann sind die beiden Entschlüsselungen von z'_i und z identisch.
Die Wahrscheinlichkeit, dass die Entschlüsselungen von z und z'_i gleich sind, sind ähnlich wie beim gerade geschilderten Angriff. Allerdings würde man einen Fehler nicht bemerken.

6.2.4 Zusammenfassung

Ebenso wie die beschränkte Variante des AJTAI-DWORK-Kryptosystems ist die unbeschränkte Variante nicht sicher im Sinne von IND-CCA1, NM-CPA und stärkeren Sicherheitsmodellen. Es resultiert im Falle von IND-CCA1 wiederum erneut aus dem Reduktionsbeweis, dass ein potentieller Angreifer, der in der Lage ist, Verschlüsselungen von Null und Eins auseinander zu halten, auch immer in der Lage ist, den privaten Schlüssel zu bestimmen. Im Falle von IND-CCA1 steht dem Angreifer jedoch immer ein Entschlüsselungsorakel zur Verfügung, so dass ihn das Unterscheiden der Verschlüsselungen vor keine großen Probleme stellt.
Im Gegensatz zur beschränkten Variante des AJTAI-DWORK-Kryptosystems können wir jedoch aus dem Reduktionsbeweis nicht direkt herleiten, dass die unbeschränkte Variante unsicher im Sinne von NM-CPA ist. Dies liegt daran, dass im Reduktionsbeweis die Anfragen an das Orakel $\mathcal{O}_{DIS}$ von Ergebnissen vorheriger Anfragen abhängen und uns derartige Anfragen im Sicherheitsmodell IND-PA0 nicht erlaubt sind. Wir nutzen stattdessen die Lokalität der Ciphertexte aus: Ciphertexte, die nahe beieinander liegen, sind mit hoher Wahrscheinlichkeit Ciphertexte desselben Klartextes. Dies können wir nutzen, da wir das Orakel $\mathcal{D}_d$ fragen dürfen, nachdem wir den Ciphertext z erhalten haben.
Sowohl die Reduktion für diese Variante des Kryptosystems als auch die Eigenschaft, dass Ciphertexte, die nahe zusammen liegen, mit hoher Wahrscheinlichkeit Verschlüsselungen desselben Klartextes sind, sind jedoch fundamentale Eigenschaften dieser Variante. Deshalb kann eine „Reparatur“

durch einfache Modifikationen – auch hier – nicht gelingen.
Eine Möglichkeit der „Reparatur“, um Sicherheit im Sinne von IND-CPA zu gewährleisten, wäre auch bei der unbeschränkten Variante eine Effizienzsteigerung. Allerdings erscheint es auch bei dieser Variante als sehr fraglich, ob dies mit einfachen Modifikationen möglicht ist – insbesondere da der öffentliche Schlüssel größer als bei der beschränkten Variante ist.

6.3 Hauptvariante

6.3.1 IND-CPA

Nguyen und Stern präsentieren in [NS98] sowie [NS99] einen Angriff auf die Indistinguishability der Variante nach Goldreich, Goldwasser und Halevi des Ajtai-Dwork-Kryptosystems, der sich leicht auf die Hauptvariante übertragen lässt. Dazu wurden Gitterbasisreduktionsalgorithmen benutzt, um das Closest-Vector-Problem zu approximieren und so den privaten Schlüssel zu errechnen. Wir wollen hier kurz die Idee beschreiben und verweisen für die Details auf die oben genannten Quellen.
Um den privaten Schlüssel bestimmen zu können, stellen wir zuerst fest, dass jeder Ciphertext z aus der Summe einiger v_i's abzüglich einer ganzzahligen Linearkombination der w_i's besteht. Da das Parallelepiped $\mathcal{P}^-(w_1, \ldots, w_n)$ nicht zu flach ist, sind die Koeffizienten der w_i's relativ klein. Nun ist aber auch jede Linearkombination der v_i's und w_i's mit kleinen Koeffizienten nahe einer der geheimen Hyperebenen. Dies ermöglicht es uns ein $(n+m)$-dimensionales Gitter zu konstruieren, so dass jede Verschlüsselung von Null nahe bei dem Gitter liegt und umgekehrt alle Punkte, die einen geringen Abstand zum Gitter haben Verschlüsselungen von Null sind. Damit können wir mit einer Approximation des Closest-Vector-Problems Verschlüsselungen von Null und Eins unterscheiden. Goldreich, Goldwasser und Halevi schlagen auch eine Variante vor, die ähnliche Ideen benutzt und eine Shortest-Vector-Problem-Aproximation benutzt.
Wir überlegen uns nun, wie wir den privaten Schlüssel u bestimmen können. Dazu erinnern wir uns, dass jedes Skalarprodukt $v_i \cdot u$ ist nahe bei einer unbekannten ganzen Zahl V_i ist. Nun lässt sich zeigen, dass eine ausreichend kurze Linearkombination der v_i's Informationen über die ganzzahligen V_i Preis gibt. Sind nämlich die Koeffizienten der v_i's ausreichend niedrig, so ist $\sum_i \lambda_i v_i$ hinreichend kurz und wir können näherungsweise $\sum_i \lambda_i V_i = 0$ setzen. Finden wir genug dieser Gleichungen, so haben wir ein lineares Gleichungssystem aufgestellt, dessen Lösung uns eine Aproximation des geheimen Schlüssels u verrät. Um die V_i zu finden, benutzen wir Giterbasisreduktionsalgorithmen um viele kurze Linearkombinationen $\sum_i \lambda_i v_i$ mit kleinen λ_i zu finden. Es läßt sich nachweisen, dass statistisch gesehen genug dieser Linearkombinationen existieren, um einen deartigen Angriff durchzuführen und so den privaten Schlüssel u zu aproximieren.

Goldreich, Goldwasser und Halevi beschreiben ein Experiment, bei dem sie dies für die Dimension $n = 32$ erfolgreich durchführen konnten. Bereits bei dieser Dimension benötigt die Schlüssellänge, die bei dieser Variante in $O(n^5 \log(n))$ liegt, etwa 20 Megabyte und jede Verschlüsselung eines Bits wird zu einem Ciphertext der Länge 6144.

6.3.2 IND-CCA1

In [HGS99] formulieren Hall, Goldberg und Schneier einen Reaktionsangriff auf die Variante nach Goldreich, Goldwasser und Halevi des Ajtai-Dwork-Kryptosystems. Ihr Idee dabei ist, eine Nachricht mit – möglicherweise falsch – berechneter Prüfsumme an das Opfer zu schicken, an Hand seiner Reaktion die Entschlüsselung einzelner Bits zu erhalten und daraus den privaten Schlüssel zu berechnen. Praktisch gesehen ist dies eine non-adaptive Chosen-Ciphertext-Attacke mit einem Vorschlag, wie man ein Entschlüsselungsorakel $\mathcal{O}_d$ schaffen kann. Diese Attacke wollen wir uns nun näher in Abschnitt 6.4.2 ansehen und stellen hier eine für die Hauptvariante des Ajtai-Dwork-Kryptosystems angepasste Version vor.
Wir werden zeigen, dass die erste Stufe des Algorithmus, A_1, mit Hilfe des Orakels $\mathcal{O}_D$ den privaten Schlüssel approximieren kann. Mit diesem ist es dann der zweiten Stufe des Algorithmus A_2 ein Leichtes, die Ciphertexte z zu entschlüsseln. Für die Approximation betrachten wir Ciphertextvektoren der Form $z' = (z'_1, \ldots . z'_n)^T$ mit $z'_i = x$ und $z'_j = 0$ für alle $i \neq j$, mit denen wir jeweils eine Komponente von u berechnen können. Dazu wollen wir der Übersichtlichkeit halber die Abfrage $\mathcal{O}_D(z')$, die $u \cdot z'$ ausrechnen und auswerten würde, mit $\mathcal{O}_i(x)$ bezeichnen. Es ist offensichtlich, dass $\mathcal{O}_i(x)$ $u_i x$ berechnet und falls $u_i x$ (nicht) im Bereich $\frac{1}{n}$ einer ganzen Zahl ist, mit Null (bzw. Eins) antwortet. Wir betrachten nun die Binärdarstellung von $|u_i| = \beta_0, \beta_1 \ldots \beta_r$ und können mit folgendem Algorithmus die Binärstellen von $|u_i|$ bestimmen:

Algorithmus 6.2. Bestimmung von $|u_i|$

1. Setze $j := 1$.
2. Setze $d_j := \mathcal{O}_i(\frac{2^j}{n} \cdot (1 + \sum_{i=0}^{j-1} 2^i \beta_i)^{-1})$
3. Setze $j := j + 1$ Falls $j \leq r$ gehe zu Schritt 2.

Bemerkung 6.3. Da $|u| \leq 1$ gilt und der Fall, dass $|u_i| = 1$ ist, extrem unwahrscheinlich und außerdem leicht festzustellen ist, gehen wir davon aus, dass $\beta_0 = 0$ ist. Mit Hilfe des Algorithmus können wir somit $|u|$ bis auf 2^{-r} bestimmen.

Die Idee dabei ist folgende: Da wir nur entscheiden können, ob ein Wert x näher als $\frac{1}{n}$ bei einer ganzen Zahl ist oder nicht, projizieren wir die Frage, ob β_i Null oder Eins ist, auf die Frage, ob ein bestimmter Wert näher bei

Null liegt als $\frac{1}{n}$ oder nicht. Dabei skalieren wir mit dem Faktor $\frac{2^j}{n}$, während wir mit $(1 + \sum_{i=0}^{j-1} 2^i \beta_i)^{-1}$ bereits bekannte Stellen berücksichtigen und da $\frac{2^j}{n} \cdot (1 + \sum_{i=0}^{j-1} 2^i \beta_i)^{-1} > \frac{1}{n}$ ist, ist unsere Eingabe nie kleiner als die kleinste in unserem System darstellbare Zahl 2^{-r}.

Bemerkung 6.4. Außerdem bemerken wir, dass ein potentieller Angreifer in der Wahl seines Koordinatensystems vollkommen ungebunden ist. Es ist keinesfalls erforderlich, dass der Angreifer sich an das Koordinatensystem hält, das sein Opfer benutzt. Dadurch wird es – insbesondere im Hinblick auf die Möglichkeit des Angreifers ein Chosen-Plaintext-Orakel durch verwenden von Prüfsummen zu erhalten – für das Opfer unmöglich eine solche Attacke zu entdecken und diese – bspw. durch das Verweigern von Antworten auf Nachrichten, die Ciphertexte der Form $z' = (z'_1, \ldots . z'_n)^T$ mit $z'_i = x$ und $z'_j = 0$ für alle $i \neq j$ enthalten – zu verhindern.

Nachdem wir den Algorithmus 6.2 auf alle Dimensionen angewendet haben, kennen wir nun alle $|u_i|$, allerdings müssen wir noch das jeweilige Vorzeichen bestimmen. Da der private Schlüssel $-u$ zu u gleichwertig ist, gehen wir o. B. d. A. davon aus, dass der erste Wert u_i, dessen Betrag größer als Null ist, positiv ist. Betrachten wir die beiden unterschiedlichen Summen $u_i + u_j$ – deren Binärstellen wir im Folgenden mit γ_i bezeichen wollen – für beide Vorzeichen von u_j, dann wollen wir mit k das erste Bit bezeichnen, in dem sich die beiden Summen unterscheiden. Fragen wir nun das Orakel $\mathcal{O}_D$ nach dem Wert für den Vektor, der in der i-ten und j-ten Komponente den Eintrag $(\frac{2^k}{n} \cdot (1 + \sum_{i=0}^{k-1} 2^i \gamma_i)^{-1})$ hat und in allen anderen Komponenten aus Nullen besteht, so können wir durch das Ergebnis das k-te Bit von $u_i + u_j$ bestimmen und somit das Vorzeichen von u_j.

Damit ist die erste Stufe des Algorithmus, A_1, in der Lage mit Hilfe des Orakels $\mathcal{O}_D$ den privaten Schlüssel u in Polynomialzeit auszurechnen. Ergo ist die Hauptvariante des Ajtai-Dwork-Kryptosystems *nicht sicher im Sinne von IND-CCA1.*

6.3.3 NM-CPA / IND-PA0

Ähnlich wie bei der unbeschränkten Variante des Ajtai-Dwork-Kryptosystems können wir auch hier die Lokalität der Ciphertexte ausnutzen. Dazu können wir erneut einen Quader Q_z um z mit den Eckpunkte z'_i wie in Abschnitt 6.2.3 konstruieren. Den optimalen Wert für ε können wir diesmal direkt aus dem Verschlüsselungsalgorithmus ablesen: 2^{-n}. Vergleichen wir nun die Entschlüsselungen von z und einem z'_i. Um z zu entschlüsseln berechnen wir den gebrochenen Anteil von $u \cdot z$ und schauen, ob er grösser oder kleiner als $\frac{1}{n}$ ist. Wir richten unser Augenmerk nun auf das Skalarprodukt von z'_i und u: $u \cdot z'_i = u \cdot z + 2^{-n} u_i$. Damit ist der Betrag der Differenz der beiden Skalarprodukte $2^{-n} u_i \leq 2^{-n}$. Wir sehen uns nun wieder die beiden Fälle an, dass eine Null bzw. eine Eins verschlüsselt wurde:

- Der zu z gehörige Klartext sei Null: D. h. der gebrochene vorzeichenbehaftete Anteil von $u \cdot z$, den wir mit μ bezeichen wollen, ist im Intervall $]-\frac{1}{n};\frac{1}{n}[$. Gehen wir der Einfachkeit halber von einer Gleichverteilung von $u \cdot z$ im Intervall $]-\frac{1}{n};\frac{1}{n}[$ aus.[4] Also können wir die Wahrscheinlichkeit, dass alle Punkte des Quaders Q_z zu 0 entschlüsselt werden, wie folgt abschätzen:

$$Pr[\|\mu\| < \frac{1}{n} \mid z = E_e(0)] \geq \frac{\frac{1}{n} - 2 \cdot 2^{-n}}{\frac{1}{n}} = 1 - 2^{1-n}n$$

- Der zu z gehörige Klartext sei Eins: Wir vernachlässigen nun den Verschlüsselungsfehler[5] und können die Wahrscheinlichkeit, dass alle Punkte des Quaders Q_z zu 1 entschlüsselt werden, wie folgt abschätzen:

$$Pr[\|\mu\| > \frac{1}{n} \mid z = E_e(1)] \geq \frac{(1 - \frac{1}{n}) - 2 \cdot 2^{-n}}{1 - \frac{1}{n}} \geq 1 - 2^{1-n}n$$

Das heißt mit mindestens der Wahrscheinlichkeit $1 - 2^{1-n}n$ werden alle Punkte des Quaders Q_z zu demselben Wert wie z entschlüsselt.
Damit ergeben sich auch schon die Aufgaben für die einzelnen Stufen des Algorithmus A. A_1 muss keinerlei Statusinformation übergeben und nur die beiden Klartexte p_0 und p_1 wählen. Der Frageteil der zweiten Stufe $A_{2,q}$ erhält nun die beiden Klartexte p_0 und p_1 sowie den Ciphertext z. Sei nun $(e_1, \ldots, e_n)$ die kanonische Basis des $\mathbb{R}^n$, dann gibt $A_{2,q}$ den Vektor $\overrightarrow{z} = (z'_1, \ldots, z'_{2n})$ aus Ciphertexten z'_i – mit $z'_i := z + 2^{-n} \cdot e_i$ falls $i \leq n$ und $z'_i := z - 2^{-n} \cdot e_{i-n}$ falls $i > n$ – aus.
$A_{2,g}$ erhält nun den Vektor $\overrightarrow{p} = (p'_1, \ldots, p'_{2n})$ aus den zu $\overrightarrow{z}$ gehörigen Klartexten, die mindestens mit der Wahrscheinlichkeit $1 - 2^{1-n}n$ alle zu demselben Wert wie z entschlüsselt wurden.
Sind alle p'_i aus $\overrightarrow{p}$ gleich, so kann sich $A_{2,g}$ sicher sein, dass z die Verschlüsselung von p'_1 ist, und muss nur noch den passenden Klartext p_b bestimmen. Für den seltenen Fall, dass in $\overrightarrow{p}$ verschiedene Werte enthalten sind, gibt $A_{2,g}$ den Wert aus, den mindestens die Hälfte aller p'_i haben. A läuft in Polynomialzeit, der Vektor $\overrightarrow{z}$, der an das Orakel $\mathcal{O}_d$ geschickt wird, hat die Länge $2n$ und die Wahrscheinlichkeit, dass $A_{2,g}$ richtig liegt, beträgt mindestens $1 - 2^{1-n}n$. Damit ist die Hauptvariante des AJTAI-DWORK-Kryptosystems *nicht sicher im Sinne von IND-PA0* bzw. *NM-CPA*.

[4] Durch die genaue Wahrscheinlichkeitsverteilung von $\mathcal{H}_{u,R,m}$ bzw. $pert(\cdot)$ sind Werte von $u \cdot z$ nahe bei 0 wahrscheinlicher als Werte nahe bei $\pm\frac{1}{n}$. Dies hätte jedoch einen positiven Einfluss auf unsere Abschätzung, so dass wir dies getrost außen vor lassen können.

[5] Dies können wir ohne weiteres tun, da der Empfänger des Ciphertextes bei einem Verschlüsselungsfehler beim Entschlüsseln denselben Fehler machen würde.

6.3.4 Zusammenfassung

Wie wir gesehen haben, existieren für die Hauptvariante des AJTAI-DWORK-Kryptosystems Angriffe, so dass sie nicht sicher im Sinne von IND-CCA1, NM-CPA und stärkeren Sicherheitsmodellen ist. Wiederum werden für die Angriffe grundlegende Eigenschaften des Kryptosystems genutzt, so dass eine „Reparatur“ mit einfachen Modifikationen nicht möglich ist. Viel schwerer wiegt allerdings, dass es auch Chosen-Plaintext-Angriffe gibt, die bei realistisch gewählten Parametern Erfolg versprechen. Auch hier könnte eine Reperatur durch eine bessere Effizienz des Kryptosystem erfolgen, da diese es ermöglichen würde die Dimension des zu Grunde liegenden Gitters zu erhöhen. Allerdings ist fraglich, ob eine Reperatur mit einfachen Modifikationen erfolgen kann. CAI und CUSICK geben in [CC99] eine Variante mit einer geringeren Ciphertextlänge an, indem sie das AJTAI-DWORK-Kryptosystem mit Rucksackproblemen verbinden. Sicherheitsbeweise wie für das AJTAI-DWORK-Kryptosystem führen sie jedoch nicht und TROMER konnte ihr Cryptosystem im Jahr 2002 brechen [Tro02].

6.4 Variante nach Goldreich, Goldwasser und Halevi

6.4.1 IND-CPA

Der Chosen-Plaintext-Angriff auf die Indistinguishability erfolgt analog des Angriffes auf die Hauptvariante, den wir in Abschnitt 6.3.1 vorgestellt haben.

6.4.2 IND-CCA1

Wir wollen uns nun den in Abschnitt 6.3.2 angesprochenen Reaktions- bzw. non-adaptive Chosen-Plaintext-Angriff von HALL, GOLDBERG und SCHNEIER ([HGS99]), aus dem wir noch zwei kleine Fehler entfernt haben, ansehen: Auch hier werden wir zeigen, dass die erste Stufe des Algorithmus, A_1, mit Hilfe des Orakels $\mathcal{O}_D$ den privaten Schlüssel approximieren kann. Mit diesem ist es dann der zweiten Stufe des Algorithmus, A_2, ein Leichtes, die Ciphertexte z zu entschlüsseln. Wiederum betrachten wir Ciphertextvektoren der Form $z' = (z'_1, \ldots . z'_n)^T$ mit $z'_i = x$ und $z'_j = 0$ für alle $i \neq j$, mit denen wir jeweils eine Komponente von u berechnen können. Die Anfrage $\mathcal{O}_i(x)$ an das Orakel sei wie in Abschnitt 6.3.2 definiert und ebenso sei die Binärdarstellung von $|u_i| = \beta_0, \beta_1 \ldots \beta_r$. Der Algorithmus, mit dem wir die Binärstellen von $|u_i|$ bestimmen, können lautet nun:

Algorithmus 6.5. Bestimmung von $|u_i|$

1. Setze $j := 0$.
2. Falls $\mathcal{O}_i(2^{j+1}) = 0$ dann $\beta_j\beta_{j+1} \in \{00, 11\}$

$$\text{sonst } \beta_j\beta_{j+1} \in \{01, 10\}$$

3. Setze $j := j + 1$. Falls $j \leq r$ gehe zu Schritt 2.

4. Setze $\beta_0 := 0$ und bilde die übrigen β_j entsprechend der jeweiligen Ergebnisse aus Schritt 2.

Bemerkung 6.6. Auf Grund der Analogie zu Algorithmus 6.2 gelten die Bemerkungen 6.3 und 6.4 selbstverständlich auch hier.

Nun müssen wir auch hier noch die Vorzeichen der jeweiligen u_i bestimmen. Da die privaten Schlüssel u und $-u$ gleichwertig sind, können wir auch hier o. B. d. A. davon ausgehen, dass der erste Wert u_i, der einen positiven Betrag hat, größer als Null ist.
Betrachten wir die beiden unterschiedlichen Summen $u_i + u_j$ für beide Vorzeichen von u_j, dann wollen wir mit k das erste Bit bezeichnen, in dem sich die beiden Summen unterscheiden. Fragen wir nun das Orakel $\mathcal{O}_D$ nach dem Wert für den Vektor, der in der i-ten und j-ten Komponente den Eintrag 2^{k-1} hat und in allen anderen Komponenten aus Nullen besteht, so können wir durch das Ergebnis das Vorzeichen von u_j bestimmen. Betrachten wir nämlich die Binärdarstellung der Summe, so stellen wir fest, dass $2^{k-1}(\beta_0, \beta_1 \dots \beta_r \pm \beta'_0, \beta'_1 \dots \beta'_r) = (\beta_0 \dots \beta_k, \beta_{k-1} \dots \beta_r \pm \beta'_0 \dots \beta'_k, \beta'_{k-1} \dots \beta'_r)$ ist. Die Abfrage an das Orakel verrät uns nun, ob $(0, \beta_{k-1} \dots \beta_r + 0, \beta'_{k-1} \dots \beta'_r)$ näher an einer ganzen Zahl als $\frac{1}{4}$ liegt oder nicht. Nach der Voraussetzung, dass sich die beiden Summen im k-ten Bit unterscheiden, ist genau eine der beiden Summen näher an einer ganzen Zahl als $\frac{1}{4}$, wodurch wir die richtige Summe und somit das Vorzeichen von u_j herausbekommen.
Damit ist die erste Stufe des Algorithmus A_1 in der Lage mit Hilfe des Orakels $\mathcal{O}_D$ den privaten Schlüssel u in Polynomialzeit auszurechnen. Ergo ist die Variante nach Goldreich, Goldwasser und Halevi des Ajtai-Dwork-Kryptosystems *nicht sicher im Sinne von IND-CCA1.*

6.4.3 NM-CPA / IND-PA0

Der Angriff auf die Non-Malleability der Variante nach Goldreich, Goldwasser und Halevi des Ajtai-Dwork-Kryptosystems kann genau so wie der Angriff auf die Non-Malleability der Hauptvariante erfolgen.

6.4.4 Zusammenfassung

Ebenso wie für die Hauptvariante des Ajtai-Dwork-Kryptosystems existieren auch für die Variante nach Goldreich, Goldwasser und Halevi Angriffe, so dass sie nicht sicher im Sinne von IND-CCA1, NM-CPA und stärkeren Sicherheitsmodellen ist. Auch hier kann eine „Reparatur„mit einfachen Modifikationen nicht erfolgen. Viel schwerer wiegt allerdings auch

hier, dass es Chosen-Plaintext-Angriffe gibt, die bei realistisch gewählten Parametern Erfolg versprechen.

6.5 Alternatives Kryptosystem von Regev

Im Rahmen dieser Arbeit wurde nicht explizit nach Angriffen auf das Kryptosystem von Regev gesucht. Da sich jedoch Chosen-Plaintext-Angriffe auf die Non-Malleability auf Grund der Ähnlichkeit zur Variante nach Goldreich, Goldwasser und Halevi des Ajtai-Dwork-Kryptosystems leicht übertragen lassen, stellen wir hier noch den entsprechenden Angriff vor.

6.5.1 NM-CPA / IND-PA0

Genau wie bei den zuvor erfolgten Angriffen, können wir wieder die Lokalität der Verschlüsselungen ausnutzen, d. h. dass Punkte, die nahe beieinander liegen mit hoher Wahrscheinlichkeit Verschlüsselung desselben Klartextes sind. Schauen wir uns dazu die Entschlüsselung näher an. Um den Ciphertext z zu entschlüsseln, betrachten wir den Rest der Divsion von $z/(N/h)$. Ist er in $]-\frac{1}{4};\frac{1}{4}[$, so entschlüsseln wir zu Null, ist sein Betrag größer als $\frac{1}{4}$, so entschlüsseln wir zu Eins.
Wir können also auch hier das Entschlüsselungsorakel $\mathcal{O}_D$ nach der Entschlüsselung benachbarter Punkte von z fragen. Sei z der gegebene Ciphertext, dann fragen unser Orakel nach den Punkten $z_1 = z+\frac{1}{N}$ und $z_2 = z-\frac{1}{N}$. Betrachten wir nun den Fall, dass eine Null verschlüsselt wurde.[6] Der Einfachkeit halber gehen wir davon aus, dass der Rest der Divsion von $z/(N/h)$ bei Verschlüsselungen von Null im Intervall $]-\frac{1}{4};\frac{1}{4}[$ gleichverteilt ist.[7] Damit können wir die Wahrscheinlichkeit, dass unsere beiden Anfragen an das Orakel ebenfalls zu Null entschlüsselt werden, wie folgt abschätzen:

$$Pr[z_1 = 0, z_2 = 0 | z = 0] \leq \frac{\frac{1}{2} - 2 \cdot \frac{1}{N}}{\frac{1}{2}} = 1 - \frac{4}{N}$$

Damit ergeben sich sofort die Aufgaben für die einzelnen Stufen des Algorithmus A. A_1 muss keinerlei Statusinformation übergeben und nur die beiden Klartexte p_0 und p_1 wählen. Der Frageteil der zweiten Stufe $A_{2,q}$ erhält nun die beiden Klartexte p_0 und p_1 sowie den Ciphertext z. Er gibt den Vektor $\overrightarrow{z} = (z + \frac{1}{N}, z + \frac{1}{N})$ aus Ciphertexten aus. $A_{2,g}$ erhält nun den Vektor $\overrightarrow{p}$ aus den zu $\overrightarrow{z}$ gehörigen Klartexten, die mindestens mit der Wahrscheinlichkeit $1 - \frac{4}{N}$ alle zu demselben Wert wie z entschlüsselt

[6] Analoges gilt für Verschlüsselungen von Eins.

[7] Diese Vereinfachung ist auch hier zulässig, da durch die Konstruktion der Verschlüsselung Werte nahe bei 0 wahrscheinlicher sind, als Werte nahe bei $\pm\frac{1}{4}$.

wurden. $A_{2,g}$ kann nun mit hoher Wahrscheinlichkeit den richtigen der beiden Klartexte p_0 und p_1 bestimmen. Damit ist das Kryptosystems *nicht sicher im Sinne von IND-PA0* bzw. *NM-CPA*.

Kapitel 7

Zusammenfassung und Ausblick

Ziel der Diplomarbeit war es, zu untersuchen, welche Sicherheitsmodelle durch das Ajtai-Dwork-Kryptosystem erfüllt werden. Dazu wurden verschiedene Absichten und Fähigkeiten eines potentiellen Angreifers vorgestellt. Als zu untersuchende Sicherheitsmodelle kristallisierten sich die von Bellare, Desai, Pointcheval und Rogaway in [BDPR98] bzw. [BDPR01] vorgeschlagenen Sicherheitsmodelle heraus. Dies lag zum einen daran, dass andere Angriffe – wie der Ciphertext-Verification-Angriff von Halevi und Krawczyk ([HK99]) oder der Reaktionsangriff von Hall, Goldberg und Schneier ([HGS99]) – sich auf Chosen-Plaintext-Angriffe zurückführen ließen. Zum anderen konnten andere Sicherheitsziele wie bspw. Plaintext-Awareness nicht in passender Weise auf das Ajtai-Dwork-Kryptosystem angewandt werden.
Also wurden die verschiedenen Varianten des Ajtai-Dwork-Kryptosystems daraufhin untersucht, ob sie Indistinguishability oder Non-Malleability unter Chosen-Plaintext-Angriffen und (non-)adaptive Chosen-Ciphertext-Angriffen bieten können. Dabei stellte sich heraus, dass keine der Varianten sicher im Sinne von Non-Malleability unter Chosen-Plaintext-Angriffen (NM-CPA) oder sicher im Sinne von Indistinguishability unter non-adaptive Chosen-Ciphertext-Angriffen (IND-CCA1) ist.
Im Falle der beschränkten Variante reichte es dazu aus, die Idee des Reduktionsbeweises von Ajtai und Dwork zu betrachten. Der Reduktionsbeweis zeigt, dass ein Angreifer, der in der Lage ist, Verschlüsselungen von Null und Eins zu unterscheiden, auch in der Lage ist, den privaten Schlüssel zu errechnen. Da sich Non-Malleability unter Chosen-Plaintext-Angriffen auf Indistinguishability unter Parallel-Angriffen zurückführen lässt, steht dem Angreifer in beiden Fällen ein Entschlüsselungsorakel zur Verfügung. Damit kann er Verschlüsselungen von Null von Verschlüsselungen von Eins unterscheiden und somit den privaten Schlüssel erlangen.

Bei der unbeschränkten Variante war dies nur für Indistinguishability unter non-adaptive Chosen-Ciphertext-Angriffen möglich. Wir konnten jedoch einen Parallel-Angriff auf die Indistinguishability der unbeschränkten Variante zeigen. Dieser macht sich zu Nutze, dass Ciphertexte Punkte im n-dimensionalen Raum sind und Punkte, die dicht beieinander liegen, mit hoher Wahrscheinlichkeit als Verschlüsselungen desselben Klartextes betrachtet werden können. Der Angreifer kann also das Entschlüsselungsorakel nach entsprechenden Punkten fragen und so Informationen über den zu entschüsselnden Ciphertext erhalten.
Der Reduktionsbeweis der Hauptvariante des Ajtai-Dwork-Kryptosystems konnte jedoch nicht direkt für einen non-adaptive Chosen-Ciphertext-Angriff auf die Indistinguishability genutzt werden. Stattdessen konnten wir aber den von Hall, Goldberg und Schneier in [HGS99] gezeigten Reaktionsangriff auf die Variante nach Goldreich, Goldwasser und Halevi in einen erfolgreichen Angriff auf die Hauptvariante ändern. Die Idee des Angriffes ist, durch Anfragen an das Entschlüsselungsorakel die Länge des privaten Schlüsselvektors zu approximieren. Ist die Länge jeder Dimension bekannt, so kann durch weitere Anfragen an das Orakel das Vorzeichen der jeweiligen Dimension und so der private Schlüssel herausgefunden werden. Chosen-Plaintext-Angriffe auf die Non-Malleability ließen sich ähnlich den Angriffen in der unbeschränkten Variante durchführen.
Für non-adaptive Chosen-Ciphertext-Angriffe auf die Indistinguishability der Variante nach Goldreich, Goldwasser und Halevi konnte der eben beschriebene Reaktionsangriff von Hall, Goldberg und Schneier direkt verwendet werden. Allerdings mußten noch zwei Fehler aus der Darstellung in [HGS99] beseitigt werden. Auch bei dieser Variante ließen sich Chosen-Plaintext-Angriffe auf die Non-Malleability ähnlich den Angriffen in der unbeschränkten Variante durchführen.
Da alle hier gezeigten Angriffe elementare Eigenschaften des Ajtai-Dwork-Kryptosystems nutzen, ist für keine der vier Varianten eine „Reparatur“ des Kryptosystems durch einfache Modifikationen möglich.
Wie in der Einleitung schon angedeutet, wiegt jedoch schwerer, dass auch Chosen-Plaintext-Angriffe auf das Ajtai-Dwork-Kryptosystem von Nguyen und Stern ([NS98], [NS99]) gefunden wurden. Dazu wurden Gitterbasisreduktionsalgorithmen benutzt, um das Shortest-Vector-Problem zu approximieren und so den privaten Schlüssel zu errechnen. Wie Nguyen und Stern aufzeigen, benötigt man dadurch so immens große Schlüssel, dass der praktische Einsatz des Ajtai-Dwork-Kryptosystems nicht in Frage kommt.
Als Ausblick verweisen wir auf das von Regev entwickelte Kryptosystem ([Reg03b]), dessen Sicherheit auf der Worst-Case Schwierigkeit des $n^{1,5}$-unique Shortest-Vector-Problem beruht. Eine gründliche Untersuchung dieses Systems konnte im Rahmen dieser Diplomarbeit nicht erfolgen. Der Angriff auf die Non-Malleability unter Chosen-Plaintext-Angriffen konnte jedoch vom Ajtai-Dwork-Kryptosystem auf Regevs System übertragen

werden. Offen ist, ob das von REGEV vorgeschlagene System sicher im Sinne der Indistinguishability ist. Da die Sicherheit des Kryptosystems von REGEV auf schwierigere Instanzen des unique Shortest-Vector-Problem als das AJTAI-DWORK-Kryptosystem reduziert wurde, besteht die Möglichkeit, dass es nicht mittels Reduktion von Gitterbasen gebrochen werden kann.

Literaturverzeichnis

[Aar03] Aaronson, Scott: *The Complexity Zoo – 377 classes.* `http://www.cs.berkeley.edu/~aaronson/zoo.html`, 2003. Version vom 2. Januar 2004.

[AD97] Ajtai, Miklós und Cynthia Dwork: *A Public-Key Cryptosystem with Worst-Case/Average-Case Equivalence.* Technischer Bericht Revision 01 of ECCC Report TR96-065, Electronic Colloquium on Computational Complexity, Trier, May 1997.

[Ajt96] Ajtai, Miklós: *Generating Hard Instances of Lattice Problems.* Technischer Bericht ECCC Report TR96–007, Electronic Colloquium on Computational Complexity, Trier, 1996.

[Ajt98] Ajtai, Miklós: *The shortest vector problem in L_2 is NP-hard for randomized reductions (extended abstract).* Technischer Bericht Revision 01 of ECCC Report TR97–047, Electronic Colloquium on Computational Complexity, Trier, November 1998.

[AR03] Aharonov, Dorit und Oded Regev: *A Lattice Problem in Quantum NP.* In: *Proc. 44th Annual IEEE Symp. on Foundations of Computer Science (FOCS)*, 2003.

[Art91] Artmann, Benno: *Lineare Algebra.* Birkhäuser Skripten. Birkhäuser Verlag, Basel, 3. überarbeitete und erweiterte Auflage, 1991.

[BDPR98] Bellare, Mihir, Anand Desai, David Pointcheval und Phillip Rogaway: *Relations Among Notions of Security for Public-Key Encryption Schemes.* In: Krawczyk, H. (Herausgeber): *Advances in Cryptology – Crypto 98 Proceedings*, Band 1462 der Reihe *Lecture Notes in Computer Science*, Seiten 26–46, Berlin – Heidelberg, 1998. Springer Verlag.

[BDPR01] Bellare, Mihir, Anand Desai, David Pointcheval und Phillip Rogaway: *Relations Among Notions of Security for Public-Key Encryption Schemes.* Full paper, Juni 2001.

[Bel97] Bellare, Mihir: *A Note on Negligible Functions*. Technischer Bericht TR 97–529, Department of Computer Science and Engineering, University of California, San Diego, März 1997.

[BPS00] Baudron, Olivier, David Pointcheval und Jacques Stern: *Extended Notions of Security for Multicast Public Key Cryptosystems*. In: Montanari, U., J. D. P. Rolim und E. Welzl (Herausgeber): *Proceedings of the 27th International Colloquium on Automata, Languages and Programming 2000*, Band 1853 der Reihe *Lecture Notes in Computer Science*, Seiten 499–511, Berlin – Heidelberg, Februar 2000. Springer Verlag.

[BR94] Bellare, Mihir und Phillip Rogaway: *Optimal Asymmetric Encryption – How to Encrypt with RSA*. In: Santis, A. de (Herausgeber): *Advances in Cryptology – Eurocrypt 94 Proceedings*, Band 950 der Reihe *Lecture Notes in Computer Science*. Springer Verlag, 1994.

[BR95a] Bellare, Mihir und Phillip Rogaway: *Optimal Asymmetric Encryption – How to Encrypt with RSA*. Technischer Bericht, 1995.

[BR95b] Bellare, Mihir und Phillip Rogaway: *Random Oracles are Practical: A Paradigm for Designing Efficient Protocols*. In: *First ACM Conference on Computer an Communications Security*. ACM, Februar 1995.

[BS99] Bellare, Mihir und Amit Sahai: *Non-Malleable Encryption: Equivalence between Two Notions, and an Indistinguishability-Based Characterization*. In: Wiener, Michael J. (Herausgeber): *Advances in Cryptology – Crypto 99 Proceedings*, Band 1666 der Reihe *Lecture Notes in Computer Science*, Seiten 519–536, Berlin – Heidelberg, Februar 1999. Springer Verlag.

[Buc99] Buchmann, Johannes: *Einführung in die Kryptographie*. Springer Lehrbuch. Springer Verlag, Berlin – Heidelberg – New York, 1999.

[Cai99] Cai, Jin-yi: *Some Recent Progress on the Complexity of Lattice Problems*. Electronic Colloquium on Computational Complexity, (ECCC Report TR99–006), 1999.

[Cai00] Cai, Jin-yi: *The complexity of some lattice problems*. In: *In Proceedings of Algorithmic Number Theory Symposium IV*, Band 1838 der Reihe *Lecture Notes in Computer Science*, Seiten 1–32, Berlin – New York, 2000. Springer Verlag.

[CC99] Cai, Jin-yi und Thomas Cusick: *A lattice-based public-key cryptosystem.* Information and Computation archive, 151(1-2):17–31, May 1999.

[Chu36] Church, Alonzo: *An unsolvable problem of elementary number theory.* American Journal of Mathematics, 58:345–363, April 1936.

[CN00] Chuang, Isaac und Michael Nielsen: *Quantum Computation and Quantum Information.* Cambridge Series on Information. Cambridge University Press, 1 Auflage, 2000.

[Coo71] Cook, Stephen: *The complexity of theorem-proving procedures.* In: *Proceedings of the Third Annual ACM Symposium on Theory of Computing*, Seiten 151–158, New York, 1971. ACM.

[CR90] Chang, Richard und Pankaj Rohatgi: *On Unique Satisfiability and Random Reductions.* Current Trends in Theoretical Computer Science, 42:494–503, 1990.

[Dam92] Damgård, Ivan: *Towards practical public key cryptosystems secure against chosen ciphertext attacks.* In: Feigenbaum, J. (Herausgeber): *Advances in Cryptology – Crypto 91 Proceedings*, Band 576 der Reihe *Lecture Notes in Computer Science*, Seiten 445–456, Berlin – Heidelberg, 1992. Springer Verlag.

[Dan63] Dantzig, George: *Linear programming and extensions.* Princeton University Press, 1963.

[DDN91] Dolev, Danny, Cynthia Dwork und Moni Naor: *Non-Malleable Crypotography.* In: *Proceedings of the 23rd ACM Symposium of Theory of Computing 1991*, Seiten 542–552. ACM, 1991.

[DDN95] Dolev, Danny, Cynthia Dwork und Moni Naor: *Non-Malleable Crypotography.* Technischer Bericht CS95–27, Weizmann Institute of Science, 1995.

[DDN00] Dolev, Danny, Cynthia Dwork und Moni Naor: *Non-Malleable Crypotography.* Society for Industrial and Applied Mathematics Journal on Computing, 30(2):391–437, März 2000.

[DH76] Diffie, Whitfield und Martin Hellman: *New Directions in Cryptography.* IEEE Transactions on Information Theory, IT-22(6):644–654, 1976.

[DKS98] Dinur, Irit, Guy Kindler und Shmuel Safra: *Approximating-CVP to Within Almost-Polynomial Factors*

is NP-Hard. In: *IEEE Symposium on Foundations of Computer Science*, Seiten 99–111, 1998.

[Fen03] Fenner, Stephen: *A Physics-Free Introduction to the Quantum Computation Model*. Bulletin of the European Association for Theoretical Computer Science, 79:69–85, Februar 2003.

[Fis95] Fischer, Gerd: *Lineare Algebra*. Vieweg Studium, Grundkurs Mathematik. Friedr. Vieweg & Sohn Verlagsgesellschaft mBH, Braunschweig – Wiesbaden, 10., vollständig neu bearbeitete und erweiterte Auflage, 1995.

[Gau01] Gauss, Carl Friedrich: *Disquisitiones arithmeticæ*. Gerh. Fleischer Iun., 1801.

[GGH96] Goldreich, Oded, Shafi Goldwasser und Shai Halevi: *Collision-Free Hashing from Lattice Problems*. Technischer Bericht ECCC Report TR96–042, Electronic Colloquium on Computational Complexity, Trier, July 1996.

[GGH97] Goldreich, Oded, Shali Goldwasser und Shai Halevi: *Eliminating Decryption Errors in the Ajtai-Dwork Cryptosystem*. Technischer Bericht ECCC Report TR97–018, Electronic Colloquium on Computational Complexity, Trier, May 1997.

[GM84] Goldwasser, Shafi und Silvio Micali: *Probabilistic Encryption*. Journal of Computer and System Sciences, 28:270–299, 1984.

[Göd31] Gödel, Kurt: *Über formal unentscheidbare Sätze der Principia Mathematica und verwandter Systeme I*. Monatshefte für Mathematik und Physik, 38:173–198, 1931.

[Gro96] Grover, Lov: *A fast quantum mechanical algorithm for database search*. In: *Proceedings of the Twenty-Eighth Annual ACM Symposium on Theory of Computing*, Seiten 212–219, Philadelphia, Mai 1996.

[Hel85] Helfrich, Bettina: *Algorithms to construct Minkowski reduced and Hermite reduced lattice bases*. Theoretical Computer Science, 41:125–139, 1985.

[Her50] Hermite, Charles: *Extraits de lettres de M. Hermite à M. Jacobi sur différents objets de la théorie des nombres, deuxième lettre*. Journal fur die Reine und Angewandte Mathematik, 40:270–290, 1850.

[HGS99] Hall, Chris, Ian Goldberg und Bruce Schneier: *Reaction Attacks against Several Public-Key Cryptosystems*. In: *In Proceedings of International Conference on Information and Communications Security '99*, Lecture Notes in Computer Science, Seiten 2–12, Berlin – Heidelberg, 1999. Springer Verlag.

[HK99] Halevi, Shai und Hugo Krawczyk: *Public-Key Cryptography and Password Protocols*. ACM Transactions on Information and System Security, 2(3):25–60, 1999.

[HLM03] Herzog, Jonathan, Moses Liskov und Silvio Micali: *Plaintext Awareness via Key Registration*. Crypto 03, 2003.

[HU79] Hopcroft, John und Jeffrey Ullman: *Introduction to Automata Theory, Languages and Computation*. Addison Wesley, 1979.

[JS98] Joux, Antoine und Jacques Stern: *Lattice Reduction: A Toolbox for the Cryptanalyst*. Journal of Cryptology: the journal of the International Association for Cryptologic Research, 11(3):161–185, Summer 1998.

[Kan87] Kannan, Ravi: *Minkowski's convex body theorem and integer programming*. Mathematics of operation research, 12(3):415–440, August 1987. Vorherige Version in ACM Symposium on Theory of Computing 1983.

[Ker83] Kerckhoffs, Auguste: *La cryptographie militaire*. Journal des sciences militaires, IX:161–191, Februar 1883.

[Kha79] Khachiyan, Leonid: *A polynomial algorithm for linear programming*. Doklady Academiia Nauk USSR, 244:1093–1096, 1979. Englische Übersetzung in Soviet Mathematics Doklady 20:191-194, 1979.

[Kle35] Kleene, Stephen: *A Theory of Positive Integers in Formal Logic*. American Journal of Mathematics, September 1935.

[Kle36a] Kleene, Stephen: *General recursive functions of natural numbers*. Mathematische Annalen, 112:727–742, 1936.

[Kle36b] Kleene, Stephen: *Lambda-definability and recursiveness*. Duke Mathematical Journal, 2:240–353, 1936.

[Koy99] Koy, Henrik: *Angriffe auf das GGH-System mittels Gitterreduktion in Blöcken*. Diplomarbeit vorgelegt bei Prof. Dr. Claus-Peter Schnorr, Lehrstuhl für Mathematische Informatik, Fachbereich Informatik der Johann-Wolfgang-Goethe-Universität Frankfurt, 1999.

[KS72] Korkin, Aleksandr und Jegor Solotarew: *Sur les formes quadratiques positives ternaires.* Mathematische Annalen, 5:581–583, 1872.

[KS73] Korkin, Aleksandr und Jegor Solotarew: *Sur les formes quadratiques positives ternaires.* Mathematische Annalen, 6:336–389, 1873.

[Lag73] Lagrange, L. J.: *Recherches d'arithmétique.* Nouv. Mém. Acad. Roy. Soc. Belles Lettres, Seiten 265–312, 1773.

[LLL82] Lenstra, Arjen, Hendrik Lenstra und Laszlo Lovász: *Factoring polynomials with rational coefficients.* Mathematische Annalen, 261(515–534), 1982.

[Lud03] Ludwig, Christoph: *A Faster Lattice Reduction Method Using Quantum Search*, 2003.

[LW92] Lehn, Jürgen und Helmut Wegmann: *Einführung in die Statistik.* Teubner Studienbücher: Mathematik. B. G. Teubner, Stuttgart, 2. überarbeitete Auflage, 1992.

[MG02] Micciancio, Daniele und Shafi Goldwasser: *Complexity of Lattice Problems: A Cryptographic Perspective*, Band 671 der Reihe *The Kluwer International Series in Engineering and Computer Science.* Kluwer Academic Publishers, Boston – Massachusetts, März 2002.

[Mic01] Micciancio, Daniele: *Improving Lattice Based Cryptosystems Using the Hermite Normal Form.* In: Silverman, Joseph H. (Herausgeber): *Cryptography and Lattices*, Band 2146 der Reihe *LNCS*, Seiten 126–145, Providence, RI, USA, 2001. Springer Verlag.

[Min96] Minkowski, Hermann: *Geometrie der Zahlen.* Teubner-Verlag, Leipzig, 1896.

[Min11] Minkowski, Hermann: *Gesammelte Abhandlungen*, Band I und II. Teubner-Verlag, Leipzig, 1911.

[MRS87] Micali, Silvio, Charles Rackoff und Bob Sloan: *The Notion of Security for Probabilistic Cryptosystems.* In: *Advances in Cryptology – Crypto 86 Proceedings*, Band 263 der Reihe *Lecture Notes in Computer Science*, Seiten 381–392, Berlin – Heidelberg, 1987. Springer Verlag.

[MS02] Mitnick, Kevin und William Simon: *The Art of Deception: Controlling the Human Element of Security.* John Wiley and Sons, Oktober 2002.

[MvOV97] Menezes, Alfred, Paul van Oorschott und Scott Vanstone: *Handbook of applied cryptography*. Discrete mathematics and its applications. CRC Press, Boca Raton – New York – London – Tokyo, 1997.

[MW00] Micciancio, Daniele und Bogdan Warinschi: *A Linear Space Algorithm for Computing the Hermite Normal Form*. Electronic Colloquium on Computational Complexity (ECCC), (074), 2000.

[NP02] Nguyen, Phong und David Pointcheval: *Analysis and Improvement of NTRU Encryption Paddings*. In: Yung, M. (Herausgeber): *Advances in Cryptology – Crypto 02 Proceedings*, Band 2442 der Reihe *Lecture Notes in Computer Science*, Seiten 210–225, Berlin – Heidelberg, 2002. Springer Verlag.

[NS98] Nguyen, Phong und Jacques Stern: *Cryptanalysis of the Ajtai-Dwork Cryptosystem*. In: Krawczyk, H. (Herausgeber): *Advances in Cryptology – Crypto 98 Proceedings*, Band 1462 der Reihe *Lecture Notes in Computer Science*, Seiten 223–242, Berlin – Heidelberg, 1998. Springer Verlag.

[NS99] Nguyen, Phong und Jacques Stern: *A Converse to the Ajtai-Dwork Security Proof and its Cryptographic Implications*. Technischer Bericht Revision 1 of ECCC Report TR98–010, Electronic Colloquium on Computational Complexity, Trier, June 1999.

[NS00] Nguyen, Phong und Jacques Stern: *Lattice Reduction in Cryptology: An Update*. In: *In Proceedings of Algorithmic Number Theory Symposium IV*, Band 1838 der Reihe *Lecture Notes in Computer Science*, Seiten 85–112, Berlin – New York, Juli 2000. Springer Verlag.

[NW99] Nemhauser, George und Laurence Wolsey: *Integer and combinatorial optimization*. Wiley-Intersience Series in Discrete Mathematics and Optimization. Wiley-Interscience, New York, 1999.

[NY90] Naor, Moni und Moti Yung: *Public-Key Cryptosystems Provably Secure against Chosen Ciphertext Attacks*. In: *Proceedings of the 22nd ACM Symposium of Theory of Computing 1990*, Seiten 427–437. ACM, 1990.

[NY95] Naor, Moni und Moti Yung: *Public-Key Cryptosystems Provably Secure against Chosen Ciphertext Attacks*. Technischer Bericht, Juli 1995.

[Odl91] Odlyzko: *The Rise and Fall of Knapsack Cryptosystems.* In: *Proceedings of the 42th Symposium in Applied Mathematics, American Mathematical Society*, Band 42, Seiten 75–88, 1991.

[Pos36] Post, Emil: *Finite combinatory processes-formulation I.* Journal of Symbolic Logic, 1:103–105, 1936.

[Pos43] Post, Emil: *Formal reductions of the general combinatorial decision problem.* American Journal of Mathematics, 65:197–214, 1943.

[Reg02] Regev, Oded: *Quantum computation and lattice problems.* In: *Proc. 43rd Annual IEEE Symp. on Foundations of Computer Science (FOCS)*, Seiten 520–529, 2002.

[Reg03a] Regev, Oded: *Improved inapproximability of lattice and coding problems with preprocessing.* In: *Proc. of 18th IEEE Annual Conference on Computational Complexity (CCC)*, Seiten 363–370, 2003.

[Reg03b] Regev, Oded: *New Lattice Based Cryptographic Constructions.* In: *Proc. 35th ACM Symp. on Theory of Computing (STOC)*, Seiten 407–416, 2003.

[RS92] Rackoff, Charles und Daniel Simon: *Non-Interactive Zero-Knowledge Proof of Knowledge and Chosen Ciphertext Attack.* In: Feigenbaum, J. (Herausgeber): *Advances in Cryptology – Crypto 91 Proceedings*, Band 576 der Reihe *Lecture Notes in Computer Science*, Seiten 433–444, Berlin – Heidelberg, 1992. Springer Verlag.

[RSA77] Rivest, Ronald, Adi Shamir und Leonard Adelman: *On Digital Signatures and Public Key Cryptosystems.* Technischer Bericht, MIT Laboratory for Computer Science Technical Memorandum 82, April 1977.

[RSA78] Rivest, Ronald, Adi Shamir und Leonard Adelman: *A Method for Obtaining Digital Signatures and Public-Key Cryptosystems.* Communications of the ACM, 21(2):120–126, 1978.

[Sch87] Schnorr, Claus-Peter: *A hierarchy of polynomial lattice basis reduction algorithms.* Theoretical Computer Science, 53:201–224, 1987.

[Sch88] Schnorr, Claus-Peter: *A more efficient algorithm for lattice basis reduction.* Journal of Algorithms, 9, 1988.

[Sch97] Schöning, Uwe: *Theoretische Informatik - kurzgefaßt.* Spektrum Akademischer Verlage, Heidelberg – Berlin, 3. Auflage, 1997.

[Sch03] Schnorr, Claus-Peter: *Lattice Reduction by Random Sampling and Birthday Methods.* In: *Proceedings of Symposium of Theoretical Aspects of Computer Science 2003*, Lecture Notes in Computer Science, Seiten 145–156, Berlin, 2003. Springer-Verlag.

[SE91] Schnorr, Claus-Peter und M. Euchner: *Lattice Basis Reduction: Improved Practical Algorithms and Solving Subset Sum Problems.* In: *Fundamentals of Computation Theory*, Seiten 68–85, 1991.

[Sha49] Shannon, Claude: *Communication Theory of Secrecy Systems.* Bell System Technical Journal, 28:656–715, Oktober 1949.

[Sho97] Shor, Peter: *Polynomial-Time Algorithms for Prime Factorization and Discrete Logarithms on a Quantum Computer.* SIAM Journal on Computing, 26(5):1484–1509, 1997.

[Str02] Streicher, Thomas: *Computability Theory.* Skript zur Vorlesung Logik II an der TU Darmstadt im WS 2002, 2002.

[Tro02] Tromer, Eran: *Observations on Cai-Cusick lattice-based public-key cryptosystem (informal notes).* Technischer Bericht, Departement of Computer Science, Weizmann Institute of Science, Rehovot 76100, Israel, Februar 2002.

[Tur36] Turing, Alan: *On computable numbers with an application to the Entscheidungsproblem.* In: *Proceedings of the London Mathematical Society*, Band 42, Seiten 230–265, 1936.

[Tur37] Turing, Alan: *Computability and lambda definability.* Journal of Symbolic Logic, 2:153–163, 1937.

[Val99] Vallentin, Frank: *Zur Komplexität des Shortest Vector Problem und seine Anwendungen in der Kryptographie.* Diplomarbeit vorgelegt bei Prof. Dr. Ingo Wegener, Lehrstuhl für Komplexitätstheorie und Effiziente Algorithmen, Fachbereich Informatik der Universität Dortmund, August 1999.

[VSB+01] Vandersypen, Lieven, Matthias Steffen, Gregory Breyta, COnstantino Yannoni, Mark Sherwood und Isaac Chuang: *Experimental realization of Shor's quantum factoring algorithm using nuclear magnetic resonance.* Nature, 414:883–887, Dezember 2001.

[Weg93] Wegener, Ingo: *Theoretische Informatik*. B.G. Teubner, Stuttgart, 1993.

[Yao82] Yao, Andrew: *Theory and Applications of Trapdoor Functions*. In: *In Proceedings of the 23rd Symposium on the Foundation of Computer Science*, Seiten 80–91, 1982.

Index

Personenverzeichnis

Symbolverzeichnis

Das folgende Symbolverzeichnis ist nicht als Ersatz für die vollständige Definition der Begriffe gedacht, sondern soll vielmehr die Übersichtlichkeit erhöhen und als Gedächtnisstütze dienen. Doppelbelegungen von Symbolen konnten dabei nicht immer vermieden werden. Es geht jedoch aus dem jeweiligen Zusammenhang hervor, welche Bedeutung gemeint ist.

a_i	Vektoren im $\mathbb{R}^n$
$\mathcal{A}$	Algorithmus
A	zweistufiger Algorithmus bestehend aus $A = (A_1, A_2)$
A_1, A_2	erste bzw. zweite Stufe von A
$A_{2,q}$, $A_{2,g}$	Teilalgorithmus von A_2
$A^{\mathcal{O}}$	Algorithmus, der das Orakel $\mathcal{O}$ benutzen darf
α	Zufallsvariable
b_i, b'_i	Vektor im $\mathbb{R}^n$
B, B'	Basis
β_i	ein Bit der Binärdarstellung einer Zahl
c, c_i	Konstante (Skalar)
$\mathcal{C}$	Ciphertextraum
γ_i	Skalar
d	Entschlüsselungsschlüssel
d, d_L, d_r	Abstand
D_d, D'_d	Entschlüsselungsfunktion zum Schlüssel d
δ_i	Zufallsvariable
$det(L)$	Determinante von L
$\mathcal{D}$	Familie von Verschlüsselungsfunktionen
e	Verschlüsselungsschlüssel
e_i	Einheitsvektor der kanonischen Basis
E	Verschlüsselungsfunktion
E_e, E'_e	Verschlüsselungsfunktion zum Schlüssel e
$\mathcal{E}$	Familie von Verschlüsselungsfunktionen
ϵ	Funktion, die den leeren String zurückgibt
ε	Skalar

$\in$	Element von
$\mathcal{G}$, $\mathcal{G}'$	Schlüsselgenerator
$GL_n(\mathbb{Z})$	multiplikative Gruppe der ganzzahligen unimodularen Transformationen in $\mathbb{Z}^n$ mit Determinate ± 1
η_K	Zufallsvariable
H, H'	Hyperebene
i	natürliche Zahl
j	natürliche Zahl
K	Skalar
$\mathcal{K}$, $\mathcal{K}'$	Schlüsselraum
$\mathcal{K}(n)$	Funktion
l	Länge
L	Gitter
$L(B)$	Gitter zur Basis B
$L(b_1, \ldots, b_n)$	Gitter mit $b_1, \ldots, b_n$ als Basis
L^*	zu L duales Gitter
$L^{(d,M)}$	eindeutig bestimmtes (d, M)-Gitter
$\mathcal{L}$	Gitterverteilung
λ_i	Skalar
m	natürliche Zahl
M	Skalar
$\mathcal{M}$	Mittel des Angreifers
n	natürliche Zahl
$\mathcal{O}_i$	Orakel
p, p_i	Klartexte
$\overrightarrow{p}$	Vektor mit Klartexten
$\mathcal{P}$, $\mathcal{P}'$	Klartextraum
$\mathcal{P}^-(b_1, \ldots, b_n)$	von $(b_1, \ldots, b_n)$ aufgespanntes Parallelepiped
$\mathcal{P}$, $\mathcal{P}'$	Parallelepiped
P	algorithmische Beschreibung eines Klartextraumes
$pert(R, m)$	Störung
$Pr[E]$	Wahrscheinlichkeit des Ereignisses E
$Pr[E_1\|E_2]$	bedingte Wahrscheinlichkeit von E_1 unter Bedingung E_2
Π, Π'	Kryptosystem
Q_i	Quader
r	Zufallsparameter
R	Relation
R	Kugelradius
$\mathbb{R}$	Menge der reellen Zahlen
$\mathbb{R}^+$	Menge der positiven reellen Zahlen
$round_\alpha(x)$	Rundung von x mit Genauigkeit α
s	Statusinformationen
s	Startvektor

$S^{(n)}(R)$	n-dimensionale Kugel mit Radius R
$\mathcal{SM}_i$	Sicherheitsmodell
u	Vektor
u_H	orthogonaler Einheitsvektor
u_i	Komponente von u_H
$U^{(n)}$	Einheitswürfel
v, v', v_i	Vektor
w, w_i	Vektor
x	Skalar
$\xi_{L,K,R}$, $\xi'_{L,K,R,z}$	Zufallsvariable mit Parametern
z, z', z^*	Ciphertexte
$\overrightarrow{z}$	Vektor mit Ciphertexten
$\mathbb{Z}$	Menge der ganzen Zahlen
$\|\lambda\|$	Betrag von λ
$\|\|b\|\|$	Länge des Vektors b
$[b_1, \ldots, b_n]$	Matrix mit Spaltenvektoren $b_1, \ldots, b_n$
B^T	Transponierte von B
B^{-1}	Inverse von B
$v \cdot w$	Skalarprodukt
$\perp$	Falsum
$\subseteq$	Teilmenge von
$:=$	algorithmische Wertzuweisung
$\stackrel{def}{=}$	Definition
$y \leftarrow A(x)$	Algorithmus A mit Eingabe x und Ausgabe y
$y \stackrel{R}{\leftarrow} A(x)$	prob. Algorithmus A mit Eingabe x und Ausgabe y

Abbildungsverzeichnis

Erklärung zur Diplomarbeit gemäß §19 Abs. 6 DPO/AT

Hiermit versichere ich, die vorliegene Diplomarbeit ohne Hilfe Dritter nur mit den angegebenen Quellen und Hilfsmitteln angefertigt zu haben. Alle Stellen, die aus den Quellen entnommen wurden, sind als solche kenntlich gemacht worden. Diese Arbeit hat in gleicher oder ähnlicher Form noch keiner Prüfungsbehörde vorgelegen.

Darmstadt, 12. März 2005

Sebastian Pape

Weitergabe der Diplomarbeit

Ich bin damit einverstanden, dass der Betreuer meiner Diplomarbeit diese nach Abschluss der Prüfung zu wissenschaftlichen Zwecken weitergeben kann und ein Exemplar zum Zwecke der Einsichtnahme durch Studenten in die Bibliothek eingestellt wird.

Darmstadt, 12. März 2005

Sebastian Pape