Algorithmische Zahlentheorie*

Seminararbeit im Masterstudiengang Angewandte Informatik
WS 2015/16 Hochschule Hannover †

Marius Rohde‡
Hochschule Hannover
Fakultät IV - Wirtschaft und Informatik
30449 Hannover
Marius.Rohde@stud.HS-Hannover.de

Marcel Reichenbach§
Hochschule Hannover
Fakultät IV - Wirtschaft und Informatik
30459 Hannover
Marcel.Reichenbach@stud.HS-Hannover.de

ZUSAMMENFASSUNG

Diese Arbeit beschäftigt sich mit der theoretischen Grundlage für die asymmetrische Kryptografie. Es werden zunächst die benötigten Grundlagen der Zahlentheorie gegeben, um anschließend einen genaueren Blick auf die Algorithmen zur Primzahlerkennung, des Diffie-Hellmann Austausches und der Anwendung elliptischer Kurven zu geben. Neben der Ausarbeitung verwendeter Algorithmen wird auch die Laufzeit dieser untersucht. Zufallszahlengeneratoren werden nicht betrachtet.

1. EINLEITUNG

Die algorithmische Zahlentheorie bildet die Grundlage der heutigen asymmetrischen Kryptographie und somit auch für einen Großteil des sicheren Datenverkehrs in Netzwerken. Ob Onlinebanking, digitale Signaturen oder virtuelle private Netzwerke, überall finden asymmetrische Verschlüsselungsalgorithmen Anwendung. Obwohl bereits Euklid ca. 300 v. Chr. zahlentheoretische Algorithmen entwickelt hat, konnte erst mit der asymmetrischen Verschlüsselung eine praktische Anwendung der Zahlentheorie gefunden werden. Zu den bedeutendsten Mathematikern die sich mit der Zahlentheorie beschäftigten gehören Euklid, Eratosthenes von Kyrene, Sun-Tse, Pierre de Fermat, Leonhard Euler, Carl Friedrich Gauß und David Hilbert. Trotz der langen Historie sind einige Fragen der Zahlentheorie wie z.B. die Unendlichkeit der Primzahlzwillinge oder die Goldbachsche Vermutung seit Jahrhunderten ungelöst. Erst 2002 konnten die drei indischen Wissenschaftlern Manindra Agrawal, Neeraj Kayal und Nitin Saxena einen Beweis für die Existenz eines deterministischen Primzahltests liefern.

Im folgenden werden die Grundlagen der Zahlentheorie eingeführt, um anschließend ausgewählte Algorithmen, die in der asymmetrischen Kryptographie eingesetzt werden, betrachten zu können. Zum Nachschlagen weiterer Informationen über Persönlichkeiten der Zahlentheorie mit historischer Einordnung, bietet das Buch [15] von Jochen Ziegenbalg einen guten Überblick.

2. GRUNDLAGEN

In diesem Kapitel werden die Grundlagen zur Analyse von Primzahlen und dem diskreten Logarithmus gegeben. Dazu werden im allg. nur die Definitionen und Sätze geliefert. Für Beweise der hier angeführten Sätze sei auf die Bücher [1], [3], [7], [5] **[TODO noch weiter wenn nötig [x],[y],[z]]** verwiesen. Es wird vorausgesetzt das die Rechenvorschrift des Modulo bekannt ist.

Für dies Ausarbeitung wurden viele Quellen unterstützend verwendet. Da es Notations-Unterschiede der einzelnen Quellen gibt führt dies dazu, dass sich nicht an die Notation aller genutzten Quellen gehalten werden kann. Speziell in [3] wird eine multiplikative Gruppe aller Einheiten in G mit G^X bezeichnet. Von dieser Notation wird abgesehen und die aus den anderen Quellen weiter verbreitete Notation G^* verwendet.

2.1 Algebraische Strukturen

In diesem Kapitel werden die algebraischen Strukturen: Halbgruppen, Gruppen, Ringe und Körper vorgestellt. Diese werden für ein späteres Kapitel benötigt. Die algebraischen Strukturen beschreiben ein abstraktes Rechnen mit Zahlen. Dies ermöglicht gezielter nur die Rechenregeln an sich zu untersuchen, unabhängig von der Rechengröße und der jeweiligen Operation. Ein Anwendungsbereich ist u. a. in der Kryptographie zu finden. [8]

2.1.1 Halbgruppen

Eine Halbgruppe ist eine Menge M mit einer assoziativen Operation ∘, geschrieben mit (M, ∘) oder einfach nur M. Zur Erinnerung, das Assoziativgesetz besagt: (a ∘ b) ∘ c = a ∘ (b ∘ c) für alle a, b, c ∈ M. Es gilt für sämtliche Elemente einer

*Genau Betrachtungen der Problemlösenden Algorithmen für ... [TODO]

†ATM ka was man hier noch beschreiben könnte erstmal [TODO]

‡Marius Rohde ... [TODO]

§Marcel Reichenbach ... [TODO]

Halbgruppe. Das Zeichen $\circ$ ist Platzhalter für eine beliebige Operation. Der Wertebereich von $\circ$ ist eine Teilmenge von M so dass, a $\circ$ b $\in$ M für alle a, b $\in$ M. Für das Zeichen $\circ$ werden auch die folgenden Operationszeichen verwendet: $*$, $\cdot$, $+$. Auch muss die Menge nicht zwangsläufig M sein. [3]

Durch das Assoziativgesetz können also Klammern weggelassen werden. Zum besseren Verständnis konkrete Beispiele von Halbgruppen [3]:

- $\mathbb{N}$, $\mathbb{Z}$, $\mathbb{Q}$, $\mathbb{R}$, $\mathbb{C}$ sind Halbgruppen mit der Addition als Operation, ebenso wie mit der Multiplikation.
- Wenn a $\circ$ b = |b - a| für alle a, b $\in \mathbb{Z}$, dann ist ($\mathbb{Z}$, $\circ$) keine Halbgruppe. Da in diesem Fall (1 $\circ$ 2) $\circ$ 3 = 1 $\circ$ 3 = 2 ist, aber 1 $\circ$ (2 $\circ$ 3) = 1 $\circ$ 1 = 0 ist. Ein verändern der Klammerung ergibt unterschiedliche Ergebnisse, somit ist das Assoziativgesetz nicht mehr gewährleistet, wodurch $\mathbb{Z}$ in diesem Fall keine Halbgruppe mehr sein kann.

Sobald es ein neutrales Element in einer Halbgruppe gibt heißt dieses **Monoid**. Ein neutrales Element ist immer dann gegeben wenn es ein e $\in$ M gibt, sodass für alle a $\in$ M gilt: a $\circ$ e = e $\circ$ a = a. Dieses weitere Axiom muss von jeder Halbgruppe erfüllt werden um ein Monoid zu sein. Als Zeichen für ein Monoid wird neben e oft auch 1 verwendet, bei den Operationszeichen $\circ$, $\cdot$, $*$. Wird das $+$ als Operationszeichen verwendet ist oft 0 das neutrale Element. [3]

Wenn ein Monoid auch das folgende Axiom erfüllt, ist es eine **Gruppe**. Existiert für alle a $\in$ G ein b $\in$ G, sodass a $\circ$ b = b $\circ$ a = e gilt, so heißt b invers zu a. [3]

2.1.2 Ringe

In einem Ring als Algebraische Struktur sind mehr als nur eine Operation vorhanden. Eine Menge R mit den zwei Operationen $+$ und $\cdot$ auf R ist genau dann ein Ring wenn folgenden drei Bedingungen gelten [3]:

- (R, $+$) ist eine abelsche Gruppe. (Eine Operation $\circ$ auf einer Menge M heiß kommutativ oder abelsch, wenn: a $\circ$ b = b $\circ$ a für alle a, b $\in$ M gilt.)
- (R, $\cdot$) ist ein Monoid
- Für alle a, b, c $\in$ R gilt: a(b + c) = ab + ac, (a + b)c = ac + bc (Distributivgesetze)

Zusätzlich heißt ein Ring kommutativ, wenn die Operation $\cdot$ kommutativ ist. Ein Ring besitzt also immer eine kommutative Addition und eine nicht notwendigerweise kommutative Multiplikation. Die beiden Distributivgesetze verbinden diese beiden Operationen miteinander. Ein Ring heißt nullteilerfrei wenn a $\cdot$ b = 0 ist und dadurch impliziert wird, dass a = 0 oder b = 0 sein muss, für alle a, b $\in$ R. Wenn es für ein a $\in$ R ein b gibt, so dass ab = ba = 1 gilt, dann ist a invertierbar oder eine Einheit. Alle Elemente im Monoid (R, $\cdot$) wo dies zutrifft sind in einer multiplikativen Gruppe zusammengefasst, bezeichnet wird diese mit R^*. [3]

Es gibt spezielle Ringe, die sogenannten Körper. Ist eine Menge (K, $+$, $\cdot$) ein Ring und ist ($K/\{0\}$, $\cdot$) eine kommutative Gruppe, heißt K ein Körper. Alle von Null verschiedenen Elementen sind Einheiten und es gilt: $K^* = K/\{0\}$. [3]

2.1.3 Integritätsbereiche

Wie in [4] angegeben ist ein Integritätsbereich ein kommutativer, nullteilerfreier Ring R mit Einselement. Aus dieser Definition ergeben sich die Integritätsbereiche der ganzen Gauß'schen Zahlen Z[i] und des Polynomrings in einer Unbestimmten X über einem Körper K der K[X] bezeichnet wird.

$$\mathbb{Z}[i] = \{n + im \in \mathbb{C} : n, m \in \mathbb{Z}\}$$

$$K[X] = \{p(X) = a_0 + a_1X + a_2X^2 + ... + a_nX^n \mid a_i \in K, n \in \mathbb{N}\}$$

Des weiteren wird ein Integritätsbereich mit einer Funktion $\beta : R \longrightarrow \mathbb{N}$, für die gilt:

$$x = qy + r, \quad q, r \in R, \text{mit } r = 0 \text{ oder } \beta(r) < \beta(y)$$

als euklidischer Ring bezeichnet und lässt die Division mit Rest zu. Wie in [4] gezeigt, ist jeder der Ringe Z, Z[i] und K[X], für einen beliebigen Körper K, euklidisch.

Die Teilbarkeit einer Zahl a $\in$ R durch b $\in$ R ohne Rest wird b | a geschrieben. Kann a durch b nicht ohne Rest geteilt werden, wird b $\nmid$ a geschrieben.

2.1.4 Restklassenringe in $\mathbb{Z}$

Restklassenringe $\mathbb{Z}/m\mathbb{Z}$ oder auch $\mathbb{Z}_m$ ergeben sich aus der Definition einer Addition und Multiplikation auf Restklassen. Eine Restklasse bezeichnet zwei Zahlen $\in \mathbb{Z}$ die bei der Division durch m $\in \mathbb{N}$ den gleichen Rest haben. Diese zwei Zahlen sind also in der gleichen Restklasse. Die Anzahl der Restklassen in einem Restklassenring ist gleich m. Die Äquivalenzrelation der beiden Zahlen bezüglich der Restklasse, kann wie folgt definiert werden: Zwei Zahlen x, y $\in \mathbb{Z}$ heißen kongruent modulo m $\in \mathbb{N}$ wenn m | $x - y$. In Zeichen:

$$x \equiv y \ mod \ m$$

Die zugehörigen Beweise und Rechenvorschriften für Restklassenringe finden sich in [4].

2.2 Euklidischer Algorithmus

Der euklidische Algorithmus wird zur Berechnung des größten gemeinsamen Teiler zweier Zahlen benötigt. Der Algorithmus findet in fast allen weiteren Betrachtungen Anwendung und wird deshalb explizit angegeben. Wie in [9] bewiesen kann der Algorithmus folgend definiert werden:

Sind m, n $\in \mathbb{N}$ zwei natürliche Zahlen mit m $\leq$ n und m $\nmid$ n, so gilt:

$$ggT(m, n) = ggT(n \ mod \ m, m).$$

2.3 Euler'sche φ-Funktion

Die Eulersche phi Funktion φ(m) berechnet die Anzahl teilerfremder Zahlen für eine gegebene Zahl m > 1. Also gilt nach [4]:

$$\varphi(m) = Card((\mathbb{Z}/m\mathbb{Z})^*).$$

Mit Card(M) ist die Kardinalität der Menge M gemeint. Die Bezeichnung $(\mathbb{Z}/m\mathbb{Z})^*$ steht für die Menge der Zahlen aus $\mathbb{Z}/m\mathbb{Z}$ die ein multiplikatives Inverses besitzen. Geschrieben sieht die Menge wie folgt aus:

$$(\mathbb{Z}/m\mathbb{Z})^* = \mathbb{Z}_m^* = \{a \in \mathbb{Z}_m \mid ggT(a,m) = 1\}.$$

Die Menge $\mathbb{Z}_m^*$ kann auch Einheitengruppe genannt werden. Ein Beispiel für den Restklassenring modulo 10 ($\mathbb{Z}_{10}^*$) sind die Elemente 1, 3, 7 und 9. In [10] kann gut nachvollzogen werden warum ein multiplikatives Inverses von a existiert, wenn der ggT(a,m) = 1 ist.

2.4 Elliptischen Kurven Grundlagen

In diesem Kapitel sollen nur die Grundlagen von elliptischen Kurven näher gebracht werden, um so die **E**lliptic **C**urve **C**ryptography, kurz **ECC**, verstehen zu können. Der Vorteil beim ECC-Verfahren im Vergleich zum RSA-Verfahren, liegt darin das die Schlüssellänge deutlich kürze ausfallen kann ohne dabei an Sicherheit zu verlieren. Ein RSA-Schlüssel mit 1024 Bit ist etwa so sicher wie ein Schlüssel aus einer elliptischen Kurve mit gerade mal ca. 160 Bit. Dazu kommt das der Rechenaufwand und Speicherbedarf beim ECC-Verfahren wesentlich geringer ist als beim RSA-Verfahren. So kann ECC in Smartcards und Mobiltelefonen genutzt werden.[5]

Um die Funktionsweise der elliptischen Kurven in ihrer vollen Breite und Tiefe zu verstehen ist dafür eine sehr komplexe Mathematik notwendig. Innerhalb dieser Seminararbeit kann dieses Thema nicht Breiter und Tiefer durchleuchtet werden und es sei ihr auf die folgende Literatur verwiesen: [5] und [7].

Eine Elliptische Kurve ist eine ebene Kurve wie in Abbildung 1 gezeigt. Sie wird durch eine Gleichung der Form: $y^2 = x^3 + ax + b$ beschrieben. Damit ist eine Menge Z aller Punkte P(x, y) die auf der elliptischen Kurve liegen definiert. Wichtig dabei ist das die Kurvenparameter a und b so gewählt sind das die partiellen Ableitungen nach x und nach y auf keinem Punkt der Kurve gleichzeitig null sind, dazu später mehr.

Das Addieren von zwei Punkten, die auf der elliptischen Kurve liegen, ergibt wieder einen Punkt welcher ebenfalls auf der Kurve liegt.[5] Mit Addition ist das Verknüpfen von zwei Punkten gemeint, man könnte es auch als Multiplikation bezeichnen. In beiden fällen hat es nichts mit den bekannten Operationen auf Zahlen zu tun. Das Addieren von zwei Punkten ist vielmehr geometrisch definiert, siehe dazu auch Abbildung 1:

> DEFINITION 1. *Durch die gegebenen Punkte P und Q wird eine Gerade gelegt, welche die Kurve in einem dritten Punkt R schneidet. Dieser wird anschließend an der x-Achse gespiegelt. Als Ergebnis erhält man den Punkt S, welcher als Addition von P und Q bezeichnet wird.[5]*

Die so definierte Addition ist kommutativ, zur Erinnerung: P + Q = Q + P. Nicht für alle elliptischen Kurven kann eine Addition von Punkten durchgeführt werden. Wie oben

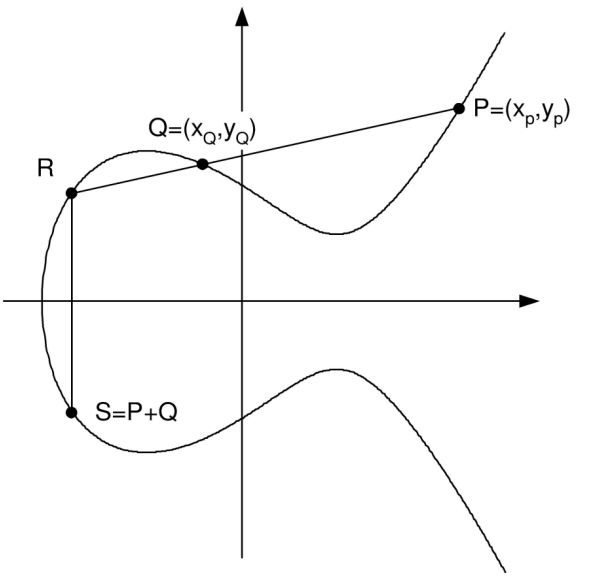

Abbildung 1: Addition von zwei Punkten auf einer elliptischen Kurve [5]

bereits erwähnt dürfen die partiellen Ableitungen nach x und nach y auf keinem Punkt der Kurve gleichzeitig null sein. Anders ausgedrückt die Kurve darf sich nicht selbst schneiden, ansonsten kann die Additionsoperation nicht für beliebige Punkte durchgeführt werden. Zusätzlich muss beachtet werden, dass bei einer Addition von zwei Punkten die nachfolgenden Spezialfälle auftreten können[5]:

- Wenn für die beiden zu Addierenden Punkten Q = P gilt, wird die Tangente an der Kurve im Punkt P verwendet. Dabei entsteht der der Schnittpunkt mit der Kurve in R und durch Spiegelung resultiert daraus S = P + P = 2P.
- Sollten die X-Koordinaten beider zu addierender Punkte gleich sein, so dass ($Q_X = P_X$) gilt, entsteht eine vertikale Gerade und die Kurve wird kein weiteres mal geschnitten. Für diesen Fall wird die elliptische Kurve um einen weiteren Punkt ∞, welcher im Unendlichen liegt, ergänzt. Die Addition von Punk P mit ∞ ist so definiert das man wiederum P als Ergebnis erhält (P + ∞ = P). Somit ist ∞ das neutral Element der Addition. Es gilt also: P + Q = ∞ wenn die x-Koordinaten von P und Q gleich sind. Daraus folgt das Q das inverse Element vo P ist und es gilt: Q = -P.

Das Addieren eines Punktes P mit einem Skalar k $\in$ {1, 2, 3 ...} wird als wiederholte Addition definiert:

$$kP = P1 + P2 + ... + Pk$$

2.4.1 Asymmetrische Verschlüsselung mit Elliptischen Kurven

Um Elliptischen Kurven für Asymmetrische Verschlüsselung einsetzen zu können muss in einem endlichen Körper gerechnet werden um Rundungsfehler zu vermeiden. Bei der Addition und Multiplikation in endlichen Körpern sind diese so definiert, dass das Ergebnis immer wider ein Element des endlichem Körpers ist. Aufgrund dessen muss eine weitere Operation durchgeführt werden: *mod* $|Z|$. Dies stellt sicher das der resultierende Rest ist in jedem Fall wieder ein Element aus Z ist. Für die Addition besitzt jedes Element ein inverses Element -a, damit gilt für die Subtraktion: $b - a = b + (-a)$. Bei der Multiplikation ist das inverse Element a^{-1}, damit gilt für die Division: $b/a = b \cdot a^{-1}$. Für ein konkretes Beispiel sei an dieser Stelle auf S. 154 - 257 in [5] verwiesen.

Um elliptische Kurven für kryptologische Anwendungsfälle zu nutzen, muss die Ordnung eines Punktes der elliptischen Kurve berechnet werden.

DEFINITION 2. *Die Ordnung eines Punktes ist die Anzahl der Punkte, die durch fortwährender Addition dieses Punktes, erzeugt werden.[5]*

Dazu folgendes Beispiel:

$$P + P = 2P \Rightarrow 2P + P = 3P \Rightarrow ... \Rightarrow xP + P = (x + 1)P$$

P ist dabei immer ein Punkt auf der elliptischen Kurve. Irgendwann ist $x_i P = \infty$, somit kein Punkt mehr auf der Kurve, und damit hat der Punkt P die Ordnung x_i, im weiteren verlauf als n bezeichnet.

2.4.2 Schlüsselaustausch mit elliptischen Kurven

Zuerst muss der Körper bestimmt werden und eine elliptische Kurve, dazu wählt man ein große Primzahl und die Kurvenparameter a und b. Weiter wird nun ein Erzeugerpunkt G vereinbart, dabei soll die Ordnung des Punktes G möglichst groß und eine Primzahl sein. A wählt eine geheime ganze Zahl n_A welche kleiner sein muss als n und berechnet daraus den öffentlichen Schlüssel $P_A = n_A \cdot G$. Teilnehmer B mach das gleich jeweils mit n_B und P_B. P_A und P_B können nun über eine unsichere Leitung ausgeschaut werden. Nun kann Teilnehmer A den Schlüssel $K = n_A \cdot P_B$ berechnen. B berechnet ebenfalls K mit n_B und P_A. So habe dabei ein und das selbe geheime K berechnet.

Es folgt der Beweis das A und B wirklich das gleiche K berechnet haben müssen:

BEWEIS 1. *A berechnet $K = n_A \cdot P_B$. P_B wurde ursprünglich von Teilnehmer B berechnet mit $n_B \cdot G$. Dies kann in die Berechnung für K von A an stelle von P_B eingesetzt werden so das daraus folgt: $K = n_A \cdot P_B = n_A \cdot (n_B \cdot G)$. Das gleiche Prinzip angewendet für die Berechnung von K durch Teilnehmer B ergibt: $K = n_B \cdot P_A = n_B \cdot (n_A \cdot G)$. Unter Berücksichtigung des Assoziativgesetzes können diese beiden Gleichungen, gleich gesetzt werden:*

$$n_A \cdot (n_B \cdot G) = n_B \cdot (n_A \cdot G)$$

Asymmetrische Verschlüsselungsverfahren basieren auf Einwegfunktionen. Wobei es nicht allzu schwierig ist $k \cdot P$ zu Berechnen allerdings ist das Berechnen von k aus $k \cdot P$ und P sehr aufwendig. Anzumerken ist das diese aussage allerdings bis heute noch nicht bewiesen wurde.

3. PRIMZAHLEN

Natürliche Primzahlen werden definiert durch Zahlen > 1 die nur durch Eins oder sich selbst teilbar sind. Wie im Kapitel Primfaktorzerlegung gezeigt, können alle natürlichen Zahlen mit einer Multiplikation von Primzahlen erzeugt werden. Sie bilden sozusagen die Bausteine aller natürlichen Zahlen. Die Unberechenbarkeit mit der sie auftreten gibt Mathematikern schon seit Jahrtausenden Rätsel auf und ist ein Grundstein unserer heutigen Verschlüsselungsverfahren. In anderen Zahlensystemen als den natürlichen Zahlen ist die gewohnte Definition von Primzahlen nicht vollständig/korrekt. Sie sagt nur etwas über die Irreduzibelität eines Elements in einem Integritätsbereich aus. Da in den natürlichen Zahlen aber jedes irreduzibles Element auch prim ist reicht diese Definition für $\mathbb{N}$ aus. Für alle Integritätsbereiche gilt für die Primheit folgende Definition nach [4]:

Ein Element $p \in R \setminus (R^* \cup \{0\})$ heißt prim oder Primelement, wenn für alle $a, b \in R \setminus \{0\}$ gilt:

$$p \mid ab \Longrightarrow p \mid a \text{ oder } p \mid b.$$

Mit Hilfe der Primzahlen kann der Restklassenring $\mathbb{Z}/m\mathbb{Z}$ spezialisiert werden. Ist m eine Primzahl p, so ist $\mathbb{Z}/p\mathbb{Z}$ ein Körper und wird auch $\mathbb{F}_p$ bzw. $\mathbb{GF}_p$ bezeichnet.

Neben den normalen Primzahlen gibt es weitere sogenannte Pseudoprimzahlen. Diese Primzahlen verhalten sich bezogen auf einen Algorithmus genauso wie echte Primzahlen, sie sind jedoch zusammengesetzt. Ein Beispiel für solche Zahlen sind Carmichaelzahlen die in einem späteren Kapitel thematisiert sind.

3.1 Primfaktorzerlegung

Jede natürliche Zahl größer als Eins kann als Produkt von Primzahlen geschrieben werden. Dieser zentrale Satz wird als Fundamentalsatz der Zahlentheorie bezeichnet und kann auf jeden euklidischen Ring R angewendet werden. Eine Primfaktorzerlegung wird mit $x \in R$, $u \in R^*$, $e_1, e_2, ..., e_m \in \mathbb{N}$ ist wie folgt definiert:

$$x = u \cdot p_1^{e_1} \cdot p_2^{e_2} \cdot ... \cdot p_m^{e_m},$$

wobei $p_1 < p_2 < ... < p_m$ Primelemente sind.

Neben der Existenz einer solchen Primfaktorzerlegung ist auch die Eindeutigkeit von entscheidender Bedeutung, die durch das Vergleichszeichen kleiner als implizit angegeben ist. In [6] und [4] wird die Existenz und Eindeutigkeit bewiesen.

Die meisten kryptographischen Algorithmen bauen auf die ineffiziente Berechnung der Primfaktoren. Es ist zwar einfach zwei Primzahlen mit einander zu multiplizieren, aber es

ist schwer aus der multiplizierten Zahl die beiden Primfaktoren zurückzugewinnen. Kennt man jedoch eine der beiden Primzahlen so ergibt sich die zweite durch einfache Division. Ein weiterer Baustein der noch fehlt ist ein effizienter Algorithmus der die Frage klärt, wann eine Zahl eine Primzahl ist.

3.2 Satz von Euler und Kleiner Satz von Fermat

Die Vermutung vom kleinen Satz von Fermat wurde von Fermat im 17 Jahrhundert aufgestellt und von Euler bewiesen. Euler konnte zudem zeigen das der kleine Satz von Fermat nur ein Spezialfall für Primzahlen darstellt. Der Satz von Euler lautet:

Sei m $\geq 2 \in \mathbb{N}$. Dann gilt für jede zu m teilerfremde Zahl a $\in \mathbb{N}$

$$a^{\varphi(m)} \equiv 1 \; mod \; m.$$

Da φ(p) = Card$((\mathbb{Z}/\mathrm{p}\mathbb{Z})^*) = |\mathbb{F}_\mathrm{p}^*| = \mathrm{p} - 1$ ist, ergibt sich sofort der kleine Satz von Fermat:

$$a^{p-1} \equiv 1 \; mod \; p$$

Mit dem Satz von Euler und Fermat kann nun ein erster primitiver Primzahltest definiert werden. Für ein beliebiges a $\in \mathbb{N}$ mit $1 <$ a $<$ m gilt:

$$a^{m-1} \not\equiv 1 \; mod \; n \Longrightarrow nicht \; prim$$

Leider ist der fermatsche Primzahltest nur ein negativ Test, der eindeutig zeigen kann das eine Zahl nicht prim ist. Wenn der Test einer Zahl kongruent 1 modulo m ergibt, muss m also nicht prim sein. Eine Zahl die für den Test prim ist, ist entweder eine echte Primzahl oder eine Pseudoprimzahl m zur Basis a.[15]

3.3 Carmichaelzahlen

Da Pseudoprimzahlen die Primalität immer zu einer Basis aufweisen, liegt der Versuch nahe andere Basen zu suchen für die der fermatsche Test nachweisen kann, dass die vermeintliche Primzahl gar keine ist. Dies ist der Grundbaustein für probabilistische Primzahltest. Dennoch gibt es Zahlen für die der fermatsche Test bei allen teilerfremden Basen kongruent eins Äquivalenz feststellt. Diese starken fermatschen Pseudoprimzahlen heißen Carmichaelzahlen.

Carmichaelzahlen werden nach [13] so definiert:

Eine zusammengesetzte Zahl m $\in \mathbb{N}$, m ≥ 3, heißt Carmichaelzahl genau dann, wenn für alle Basen a mit ggT(m, a) = 1 gilt:

$$a^{m-1} \equiv 1 \; mod \; m$$

TODO evtl. Beispiel

3.4 Sieb des Eratosthenes

Jo der einfache kram :D mit Bild

3.5 Effiziente Primzahltests

Die in diesem Kapitel vorgestellten Algorithmen zum Erkennen von Primzahlen sind effiziente Algorithmen im Sinne der Komplexitätsklasse P und im Gegensatz zum fermatschen Test nicht anfällig für starke Pseudoprimzahlen. Dennoch unterscheiden sich die beiden vorgestellten Verfahren deutlich voneinander.

3.5.1 Miller-Rabin-Test

Der Miller-Rabin-Test ist ein probabilistischer Primzahltest und gehört zu den Montecarlo- Algorithmen. Im Gegensatz zum fermatschen Primzahltest ist der Miller-Rabin-Test nicht so anfällig für Carmichaelzahlen.

3.5.2 AKS-Test

Der AKS-TestEntwickelt von ... ist ein deterministischer in polinimieller Zeit durchführbarer Primzahltest.

4. DISKRETER LOGARITHMUS

In diesem Kapitel soll der diskreter Logarithmus betrachtet werden. Es werden Anwendungspeilspiele aufgezeigt und genauer erläutert weswegen sich der diskreter Logarithmus besonders gut in der Kryptografie als Einwegfunktion eignet. Zu beginn wird das grundsätzliche Problem beim diskreten Logarithmus an einem Beispiel praxisnahe erläutert, um im Anschluss auf Algorithmische Lösungen einzugehen wie der diskreten Logarithmus in $\mathbb{Z}_\mathrm{p}^*$ und auf elliptische Kurven berechnet werden kann. Hierfür wird der Baby-Step-Giant-Step-Algorithmus vorgestellt.

4.1 Das Problem des diskreten Logarithmus im Detail

Es existiert eine Primzahl p, ein erzeugendes Element g für $\mathbb{Z}_\mathrm{p}^*$ sowie eine ganze Zahl x. Zu der diskreten Exponentialfunktion $g^x \; mod \; p$ gibt es die diskrete Logarithmusfunktion, die zu einem gegebenen y und g, x beschreibt. Somit ist x der diskrete Logarithmus von y zur Basis g. ($y = g^x \; mod \; p$) Jede Zahl aus $\mathbb{Z}_\mathrm{p}^*$ lässt sich als Potenz von g darstellen, wenn g ein erzeugendes Element von $\mathbb{Z}_\mathrm{p}^*$ ist. Ist dies nicht der Fall, so muss es nicht zu jedem $y \in \mathbb{Z}_\mathrm{p}^*$ einen diskreten Logarithmus geben. [1] Um das eigentliche Problem des diskreten Logarithmus zu verstehen ist es hilfreich, die Logarithmen in $\mathbb{R}$ mit Logarithmen in $\mathbb{Z}_\mathrm{p}^*$ gegenüberzustellen.

$$\begin{aligned} y &= g^x \\ 1024 &= 2^x \\ x &= log_2 1024 \\ x &= 10 \end{aligned} \quad (1) \qquad \begin{aligned} y &= g^x \; mod \; p \\ 10 &= 2^x \; mod \; 11 \\ x &= log_2 10 \\ x &= 3.32193... \; \lightning \end{aligned} \quad (2)$$

Die Gleichungen aus (1) zeigen wie der Logarithmus von x zur Basis g, mit der Logarithmusfunktion berechnet werden kann. Auf diese Weise sind Gleichungen für die positiven reellen Zahlen $\mathbb{R}^+$ immer eindeutig lösbar. Für die Gleichungen aus (2) wird ebenfalls mit der Logarithmusfunktion versucht, den Logarithmus von 10 zu Basis 2 zu erhalten. Als Ergebnis erhält man eine Zahl die nicht in $\mathbb{Z}_\mathrm{p}^*$ enthalten ist. Was auch nicht verwundert, da die Logarithmusfunktion *mod* 11 gar nicht berücksichtigt. Genau hier ist das grundsätzlich Problem beim diskreten Logarithmus. Es gibt keine mathematische Rechenoperation die es ermöglicht den diskreten Logarithmus in einem endlichen Körper mit nur ei-

nem Rechenschritt zu berechnen. Um dennoch eine Lösung zu erhalten, scheint das Enumerationsverfahren das naheliegendste zu sein. Hierbei werden einfach alle Werte die für x in frage kommen durchprobiert. So ist die Lösung der Gleichungen aus (2), $x = 5$. [11] Für sehr große Gruppen, wo x Beispielsweise eine 160 Bit große Zahl ist, gibt es bis heute keine Algorithmen die den diskreten Logarithmus effizient berechnen.[1] Es gibt allerdings eine ganze Reihe von Algorithmen die in der Lage sind den diskreten Logarithmus gezielter zu berechnen als das naive Ausprobieren. Nachfolgend soll der Baby-Step-Giant-Step-Algorithmus genauer betrachtet werden.

4.1.1 Diskreter Logarithmus auf elliptische Kurven

In Kapitel 2.4 wurde die Addition von Punkten und Skalaren auf elliptische Kurven beschrieben. In diesem Unterkapitel soll kurz beschrieben werden was genau der Logarithmus auf einer elliptische Kurven über $\mathbb{Z}_p^*$ ist. Für eine elliptische Kurve E über den Primkörper $\mathbb{Z}_p^*$ mit den Punkten $P, Q \in E(\mathbb{Z}_p^*)$, gibt es ein $k \in \mathbb{Z}_p^*$ sodass die folgende Gleichung erfüllt ist:

$$Q = kP$$

So wird die Zahl k als diskreter Logarithmus von Q zur Basis P bezeichnet. Den diskreten Logarithmus aus $E(\mathbb{Z}_p^*)$ zu bestimmen, ist noch einmal ungleich komplexer als aus $\mathbb{Z}_p^*$. Dieses Berechnungsproblem wird **E**lliptic **C**urve **D**iscrete **L**ogarithmen **P**roblem, kurz **ECDLP**, genannt.

4.2 Baby-Step-Giant-Step-Algorithmus

In der Praxis ist es mit diesem Algorithmus nicht möglich eine Verschlüsselung die auf dem DLP aufbaut zu brechen. Die Komplexität diese Algorithmus liegt in $O(\sqrt{p-1})$ ist somit schon deutlich besser als eine naive vollständige Suche, deren Komplexität in $O(p-1)$ liegt, dennoch zu stark von der Größe der zugrundeliegenden Gruppe abhängig ist. Anzumerken ist, dieser Algorithmus ist ein sogenannter generischer Algorithmus, womit dieser für jede Gruppe funktioniert und nicht von einer speziellen Struktur der Gruppe abhängt. [1]

Zuerst wählt der Algorithmus eine Zahl t $\in \mathbb{N}$, sodass $t \geq \sqrt{p-1}$ ist. Mit einer zweiten Zahl r ($0 \leq r < t$) lässt sich die Gleichung des diskreten Logarithmus schreiben als:

$$x = q \cdot t + r \tag{3}$$

und lässt sich wie folgt umformen:

$$y = g^x = g^{q \cdot t + r} \;\Leftrightarrow\; y \cdot g^{-r} = g^{q \cdot t} \tag{4}$$

Nun müssen die zwei Zahlen r und q gesucht werden, sodass die Gleichung $y \cdot g^{-r} = g^{q \cdot t}$ gilt. Dazu werden zwei Listen angelegt und im Nachhinein werden die Einträge miteinander verglichen.

Baby-Step Liste: $y \cdot g^{-r}$ für alle r mit $0 \leq r < t$

Giant-Step Liste: $g^{q \cdot t}$ für alle q mit $0 \leq q < t$

Sollte ein Eintrag vorhanden sein der in beiden Listen vorkommt, liefert dieser den diskreten Logarithmus x. [1]

4.2.1 Baby-Step-Giant-Step Vorgehen

Zu erst errechnet man t mit $\sqrt{p-1}$. Nun werden alle baby steps benötigt, diese werden mit Gleichung (5) ermittelt:

$$B = \{\ (x \cdot g^{-r}\ mod\ p,\ r)\ :\ 0 \leq r < t\ \} \tag{5}$$

Die Ergebnisse von (5) mit dem dazugehörigen r werden in einer Liste gespeichert. Mit Gleichung (6) werden nun die giant step berechnet.

$$G = \{\ g^{t \cdot q}\ mod\ p\ :\ q = 1, 2, 3, ...\ \} \tag{6}$$

Die so ermittelten Lösungen von (6) werden mit denen aus der Liste der baby steps verglichen. Stimmen baby step und giant step überein, wurde eine Kollision gefunden und die Gleichung aus (4) ist erfüllt. Die so ermittelten Werte für q und r können nun in Gleichung (3) eingesetzt werden und erhält so den diskreten Logarithmus von y.

4.2.2 Baby-Step-Giant-Step zum lösen des ECDLP

Am Grundsätzlichen Vorgehen beim Baby-Step-Giant-Step-Algorithmus auf elliptische Kurven ändert sich nicht viel, weswegen hier nur noch einmal die Unterschiede gezeigt werden. Die baby steps werden nach Gleichung (7) berechnet:

$$B = \{\ (\ Q - rP,\ r\ )\ :\ 0 \leq r < t\ \} \tag{7}$$

Diese werden in einer Liste gespeichert um sie in einem weiteren Schritt mit den giant steps zu vergleichen:

$$G = \{\ (\ qmP,\ q\ )\ :\ 0 \leq q < t\ \} \tag{8}$$

Bei gefundener Kollision kann aus q und r, $k = qm + r$ errechnet werden und man erhält den diskreten Logarithmus von Q.

4.3 Index-Calculus-Algorithmus

Der Index-Calculus-Algorithmus kann das DLP „besser“ lösen als der Baby-Step-Giant-Step-Algorithmus. Es sei an dieser Stell schon vorweggenommen auch dieser ist vom effizienten Lösen noch weit entfernt. Ausgangsbasis ist die schon aus 4.1 bekannte Gleichung (9) zum diskreten Logarithmus aus $\mathbb{Z}_p^*$.

$$y = g^x\ mod\ p \tag{9}$$

Für jedes $h \not\equiv 0\ (mod\ p)$ kann als $h \equiv g^k$ geschrieben werden, womit eindeutig $mod\ p-1$ bestimmt wird. Der diskrete Logarithmus von x wird als $L(x)$ geschrieben, sodass $x = L(x)$ gilt. Daraus folgt:

$$g^{L(h)} \equiv h \quad (mod\ p) \tag{10}$$

Weiter gelten für alle Werte $x_1, x_2 \in \mathbb{Z}_p$:

$$g^{L(h_1 h_2)} \equiv h_1 h_2 \equiv g^{L(h_1) + L(h_2)} \quad (mod\ p) \tag{11}$$

Der diskrete Logarithmus des Produktes aus zwei Zahlen ist die Summe der diskreten Logarithmen der einzelnen Zahlen:

$$L(h_1 h_2) \equiv L(h_1) + L(h_2) \quad (mod\ p) \tag{12}$$

Die Grundidee beim Index-Calculus-Algorithmus ist, aus der Kenntnis des diskreten Logarithmus für einige Zahlen, kann über die passenden Gleichungen der diskrete Logarithmus von beliebigen Zahlen berechnet werden.

Zunächst bildet man eine Faktorbasis Menge B aus kleinen Primzahlen. Als nächstes müssen passende Gleichungen gefunden werden wo die Primfaktorzerlegung von g^x nur aus Potenzen von Elementen aus B besteht. Siehe dazu das Beispiel aus [12] S. 144. Nun kann durch einsetzen der diskrete Logarithmus sämtlicher gefundenen Gleichungen ermittelt werden. Es muss solange ein zufälliger wert für j in Gleichung

$$g^j \cdot x = y \ mod \ p \tag{13}$$

eingesetzt werden bis ein y, das Produkt aus Elementen von B ist. Da die diskreten Logarithmen der Elemente aus B bekannt sind, kann mit dessen Hilfe wie in Gleichung (13) gezeigt der diskrete Logarithmus von y berechnet werden.

5. FAZIT

Fazit zum DLP:

Es gibt noch eine reihe weiterer Algorithmen zum lösen des DLP. Besonders der Index-Calculus-Algorithmus ist den allgemein verwendbaren Algorithmen deutlich überlegen, den dieser hat eine Laufzeit von etwas $O(exp(\sqrt{2 \cdot ln \ p \cdot ln \ ln \ p}))$. Eine genaue Beschreibung diese Algorithmus ist dem Buch [2] zu entnehmen. Dieser ist auch dafür mitverantwortlich weswegen elliptische Kurven gegenüber den multiplikativen Gruppen der endlichem Körper, für kryptografische Anwendungsfälle „sicherer“ sind. Der Baby-Step-Giant-Step-Algorithmus kann sowohl für das DLP sowie für das ECDLP eingesetzt werden, besitzt allerdings nur eine Laufzeit von $O(\sqrt{p-1})$. Der Index-Calculus-Algorithmus hingegen lässt sich nicht auf elliptischen Kurven umschreiben, hier bleiben nur die wesentlich langsameren allgemeinen Verfahren.[14]

6. REFERENCES

[1] A. Beutelspacher, H. B. Neumann, and T. Schwarzpaul. *Kryptografie in Theorie und Praxis*. Vieweg + Teubner and GWV Fachverlage GmbH, Wiesbaden, 2010.

[2] J. Buchmann. *Einführung in die Kryptographie*. Springer, Wiesbaden, 2006.

[3] O. Deiser and C. Lasser. *Erste Hilfe in Linearer Algebra*. Springer Spektrum, Berlin Heidelberg, 2015.

[4] O. Forster. *Algorithmische Zahlentheorie*. Springer Fachmedien, Wiesbaden, 2015.

[5] M. Hufschmid. *Information und Kommunikation*. Teubner Verlag, Wiesbaden, 2006.

[6] R. Schulze-Pillot. *Einführung in Algebra und Zahlentheorie*. Springer, Berlin Heidelberg, 2015.

[7] S. Spitz, M. Pramateftakis, and J. Swoboda. *Kryptographie und IT-Sicherheit*. Vieweg + Teubner Verlag and Springer Fachmedien, Wiesbaden, 2011.

[8] P. D. F. Sprengel. Kryptographie und algorithmen. Technical report, Hochschule Hannover, 2012.

[9] A. Steger. *Diskrete Strukturen Band 1*. Springer, Berlin Heidelberg, 2007.

[10] G. Teschl and S. Teschl. *Mathematik für Informatiker Band 1*. Springer, Berlin Heidelberg New York, 2007.

[11] D. Wallerstorfer. *DLP/ECDLP Probleme und Lösungen*. Fachhochschul für Computer- und Mediensicherheit in Hagenberg, 2006.

[12] L. C. Washington. *Elliptic Curves - Number Theory and Cryptography*. Taylor Francis Group, LLC, University of Maryland, 2008.

[13] K.-U. Witt. *Algebraische und zahlentheoretische Grundlagen für die Informatik*. Springer Fachmedien, Wiesbaden, 2014.

[14] M. Wohlgemuth. *Mathematisch für fortgeschrittene Anfänger*. Springer Spektrum, Wiesbaden, 2010.

[15] J. Ziegenbalg. *Elementare Zahlentheorie*. Springer Spektrum, Wiesbaden, 2015.

7. ANHANG

Anhang ... [TODO]