Kryptosysteme auf Basis imaginärquadratischer Nichtmaximalordnungen

Vom Fachbereich Informatik
der Technischen Universität Darmstadt
genehmigte

Dissertation

zur Erlangung des akademischen Grades
Doctor rerum naturalium (Dr.rer.nat.)

von

Dipl.-Inform. (FH) Detlef Hühnlein

aus Kronach

Referenten:	Prof. Dr. J. Buchmann
	Prof. Dr. T. Takagi
Tag der Einreichung:	29.10.2004
Tag der mündlichen Prüfung:	07.12.2004

Darmstadt 2004

D17

Für Anita und Fritz.

Danksagung

Mein besonderer Dank gilt Prof. Dr. Johannes Buchmann, der die vorliegende Arbeit nicht nur grundsätzlich ermöglicht und die interessante Thematik vorgeschlagen, sondern die Entwicklung der Arbeit mit hilfreichen Gesprächen und fruchtbaren Anregungen stets konstruktiv begleitet hat. Insbesondere möchte ich mich für seinen Mut, sich auf das Abenteuer meiner Promotion einzulassen, und seine Geduld bedanken.

Großer Dank gilt auch Prof. Dr. Tsuyoshi Takagi, der mit vielen Anmerkungen und Verbesserungsvorschlägen die Qualität der Arbeit zu steigern vermochte.

Außerdem möchte ich mich bei allen Mitgliedern der Arbeitsgruppe an der TU Darmstadt und insbesondere bei Michael J. Jacobson, Andreas Meyer, Johannes Merkle, Sachar Paulus und Damian Weber für die konstruktive Zusammenarbeit bei der Erarbeitung unserer gemeinsamen Veröffentlichungen bedanken.

Schließlich möchte ich mich bei allen Familienmitgliedern und Freunden, und insbesondere meiner Liebsten Tina, für die entgegengebrachte Ermunterung und Unterstützung bedanken, ohne die diese Arbeit niemals zustandegekommen wäre. Schließlich danke ich meinem lieben Moritz und vor allem meinem Arbeitgeber, dass ich den Genuss die vorliegende Arbeit anzufertigen so lange auskosten durfte.

Vielen Dank.

Zusammenfassung

Die vorliegende Arbeit behandelt die Konstruktion und effiziente Implementierung von Kryptosystemen auf Basis imaginärquadratischer Nichtmaximalordnungen.

Während die bisher bekannten Kryptosysteme in der Klassengruppe imaginärquadratischer Ordnungen [**BW88**], verglichen mit vielen populären Verfahren, ein höheres Maß an Sicherheit versprechen, so fanden sie in der Praxis bislang kaum Beachtung, da die benötigte Arithmetik nur vergleichsweise ineffiziente Implementierungen erlaubt.

Deshalb wurde in [**HJPT98**] vorgeschlagen, anstatt der bislang verwendeten Maximalordnungen nunmehr imaginärquadratische Nichtmaximalordnungen zur Konstruktion von Verschlüsselungssystemen mit effizienterer Entschlüsselung zu verwenden. Neben der etwa fünffach beschleunigten Entschlüsselung führte dieser Vorschlag insbesondere zur Entwicklung einer völlig neuartigen Klasse von Kryptosystemen – den Verfahren auf Basis imaginärquadratischer Nichtmaximalordnungen.

So führte eine weitere Variation des Systemaufbaus zur Entwicklung der NICE-Kryptosysteme [**PT00, HM00b**], die bei Verwendung der hier vorgestellten Algorithmen [**Hüh00, Hüh01**] eine sehr effiziente Signaturerzeugung und Entschlüsselung erlauben. Entgegen aller populärer Verschlüsselungsverfahren, benötigt NICE zur Entschlüsselung nur quadratische Laufzeit. Die Signaturerzeugung bei den NICE-Schnorr- und NICE-Guillou-Quisquater-Signaturverfahren in $\mathrm{Ker}(\phi_{Cl}^{-1})$ ist bei vermutlich gleicher Sicherheit etwa doppelt so schnell wie bei entsprechenden Analogen zu [**Sch90, GQ90**] in $(\mathbb{Z}/dp^2\mathbb{Z})^*$.

Schließlich kann die in dieser Arbeit vorgestellte Verallgemeinerung der – in [**HT99**] für einen Spezialfall vorgeschlagenen – Reduktion des DL-Problemes von der Klassengruppe der Nichtmaximalordnung auf die kleinere Klassengruppe der Maximalordnung und einigen endlichen Körpern zur Konstruktion eines nichtinteraktiven ElGamal-Verschlüsselungsverfahren [**HJW01**] verwendet werden. Der Vorteil dieses Verfahrens ist, dass gewisse Sicherheitsprobleme der bekannten Verfahren vermieden werden können und die Arbeitslast für die zentrale Schlüsselerzeugungsinstanz vergleichsweise gering ist.

Neben der Diskussion dieser Kryptosysteme samt zugehörigen Sicherheitsaspekten, enthält diese Arbeit die zur effizienten Implementierung benötigten Algorithmen und erste Laufzeitergebnisse.

Symbol- und Abkürzungsverzeichnis

		Seite
$\sim$	Äquivalenz von Idealen	9
$\otimes$	direktes Produkt	
$\oplus$	binäres, exklusives Oder	
R/I	R modulo $I \subset R$	
R^*	Einheitengruppe von R	
$\lceil r \rceil$	kleinste ganze Zahl $\geq r$	
$\lfloor r \rfloor$	größte ganze Zahl $\leq r$	
$\{0,1\}^*, \{0,1\}^l$	Bitstring beliebiger Länge, bzw. der Länge l	
$\left(\frac{\Delta}{p}\right), (\Delta/p)$	Kronecker-Symbol	8
$\left(\frac{a}{p}\right)$	Legendre-Symbol, für p ungerade Primzahl und $a \not\mid p$	
$\mathfrak{a}, \mathfrak{b}, \cdots, \mathfrak{A}, \mathfrak{B} \cdots$	quadratische Ideale	8
$\overline{\alpha}$	konjugierte Zahl	7
BPC	Birthday Paradox Continuation	55
CCA1	Non-Adaptive Chosen-Ciphertext Attack	43
CCA2	Adaptive Chosen-Ciphertext Attack	43
CDHP	Computational Diffie-Hellman Problem	46
CPA	Chosen-Plaintext Attack	43
$Cl(\Delta)$	Klassengruppe	9
CWDHP	Computational Weil Diffie-Hellman Problem	78
DDHP	Decisional Diffie-Hellman Problem	46
DLP	Discrete Logarithm Problem	46
Δ	Diskriminante	7
$E_{[\text{Len01}]}[n,p]$	Funktion zur Beschreibung der ECM-Laufzeit	56
$E[n,p,\tau(p)]$	Funktion zur Beschreibung der ECM-Laufzeit	56
$EC(\mathbb{F}_q)$	Elliptische Kurve über $\mathbb{F}_q$	
$\in_r$	zufällige und gleichverteilte Auswahl eines Elementes	62
FFTC	Fast Fourier Transform Continuation	55
$\text{FP}_{\Delta p^2}$	Faktorisierungsproblem für Zahlen der Form Δp^2	46
$\mathbb{F}_q$	endlicher Körper mit q Elementen	
$\overline{\mathbb{F}_q}$	algebraischer Abschluss von $\mathbb{F}_q$	
$\mathbb{F}_q[X]$	Polynomring über $\mathbb{F}_q$	
$f(X)$	charakteristisches Polynom	23
g, h	Hashfunktionen	
GDHP	Gap Diffie-Hellman Problem	46
$\text{ggT}(a,b)$	größter gemeinsamer Teiler von a und b	
GNFS	General Number Field Sieve	51
$\langle \mathfrak{g} \rangle$	von $\mathfrak{g}$ erzeugte, zyklische Gruppe	
GRH	verallgemeinerte Riemann'sche Vermutung	
G, H	endliche abelsche Gruppen	30
$h(\Delta)$	Klassenzahl, Ordnung der Klassengruppe $Cl(\Delta)$	9
IND	Indistinguishability, Ununterscheidbarkeit	43

		Seite
ISC	Improved Standard Continuation	55
$\mathcal{I}_\Delta$	Menge der invertierbaren Ideale	9
$\mathcal{I}_\Delta(f)$	Menge der zu f primen Ideale	10
JC	Jacobi Symbol Continuation	56
$\mathrm{Ker}(\phi_{Cl}^{-1})$	Kern von ϕ_{Cl}^{-1}	11
$\lambda(n)$	Carmichael-Funktion	58
$L_n[u,v]$	Laufzeit subexponentieller Algorithmen	6
$\log(x)$	natürlicher Logarithmus	
$\log_2(x)$	Logarithmus zur Basis 2	
$\log_{\mathfrak{g}}(\mathfrak{y})$	Diskreter Logarithmus	
l_Δ	Bitlänge der Diskriminante	
l_h	Bitlänge der Hashwerte	
l_m	Bitlänge der Nachrichten	
$\hat{l_m}$	maximale Bitlänge der Nachrichten	
l_q	Bitlänge der Untergruppe	
l_r	Bitlänge der Zufallszahlen	
l_s	Bitlänge der geheimen Exponenten	
$\max\{a_1,\cdots,a_n\}$	Maximum der Elemente $a_1,\cdots,a_n$	
$M(n)$	Aufwand der Multiplikation modulo n	54
MIPS	Million Instructions Per Second	53
MJ	MIPS-Jahr	53
MPQS	Multiple-Polynomial-Quadratic-Sieve	49
NM	Non-Malleability, Nicht-Verformbarkeit	43
$N(\alpha)$	Norm einer quadratischen Zahl α	8
$\mathcal{N}(\mathfrak{a})$	Norm eines Ideales $\mathfrak{a}$	8
$o(f(n))$	o-Notation	6
$O(f(n))$	O-Notation	6
$\mathcal{O}_\Delta$	Quadratische Ordnung der Diskriminante Δ	7
$\mathcal{O}_{\Delta f^2}$	Quadratische Nichtmaximalordnung mit beliebigem Führer f	7
$\mathcal{O}_{\Delta p^2}$	Quadratische Nichtmaximalordnung mit primen Führer p	7
ω	quadratische Wurzel	7
OW	One-Wayness, Einwegeigenschaft	42
PCA	Plaintext-Checking Attack	43
φ	Isomorphismus $\mathcal{I}_\Delta(f)\tilde{\rightarrow}\mathcal{I}_{\Delta f^2}(f)$	10
φ^{-1}	Inverses von φ	10
Φ	Homomorphismus $G \rightarrow H$	30
ϕ_{Cl}^{-1}	Homomorphismus $Cl(\Delta f^2) \rightarrow Cl(\Delta)$	11
$\psi_{\mathrm{Ker}(\phi_{Cl}^{-1})}$	Homomorphismus $(\mathcal{O}_\Delta/f\mathcal{O}_\Delta)^* \rightarrow \mathrm{Ker}(\phi_{Cl}^{-1})$	19
$\psi_{\mathbb{F}}$	Isomorphismus zwischen $(\mathcal{O}_\Delta/f\mathcal{O}_\Delta)^*$ und endl. Körper	24
$\psi_{\mathbb{F}}^{-1}$	Inverses von $\psi_{\mathbb{F}}$	24
ψ_{p^2}	Homomorphismus von endl. Körper nach $\mathrm{Ker}(\phi_{Cl}^{-1})$	26
ψ	Isomorphismus $\mathbb{F}_p^* \tilde{\rightarrow} \mathrm{Ker}(\phi_{Cl}^{-1})$, für $\left(\frac{\Delta}{p}\right)=1$	27
$\mathcal{P}_\Delta$	Menge der invertierbaren Hauptideale	9
$\mathcal{P}_\Delta(f)$	Menge der zu f primen Hauptideale	10
$\mathbb{Q}, \mathbb{Q}_{>0}$	rationale, positive rationale Zahlen	
$\mathbb{Q}(\sqrt{\Delta})$	quadratischer Körper der Diskriminante Δ	7
$\mathbb{R}, \mathbb{R}_{>0}, \mathbb{R}_{\geq 0}$	reelle, positive reelle, nicht-negative reelle Zahlen	
$\rho, \overline{\rho}$	Nullstellen von $f(X)$	23

		Seite
$\rho()$	Reduktionsoperator	11
SC	Standard Continuation	55
$\mathcal{SI}(\Delta p^2)$	Menge der Idealklassen in $Cl(\Delta p^2)$ mit kleiner Norm	11
SNFS	Special Number Field Sieve	51
$\tau(p)$	Funktion zur Beschreibung der ECM-Laufzeit	55
$\mathbb{Z}, \mathbb{Z}_{>0}, \mathbb{Z}_{\geq 0}$	ganze, positive ganze, nicht-negative ganze Zahlen	

Algorithmenverzeichnis

Verzeichnis der Kryptosysteme

Tabellenverzeichnis

Inhaltsverzeichnis

Einleitung

Die durch Diffie und Hellman [**DH76**] eingeführte Public-Key-Kryptographie gehört heute zweifelsohne zu den Schlüsseltechnologien, die für die sichere Kommunikation und für die vertrauenswürdige Abwicklung elektronischer Geschäftsprozesse in offenen Netzen unverzichtbar sind. Neben der **Effizienz** ist vor allem die **Sicherheit** das entscheidende Kriterium bei der Auswahl kryptographischer Verfahren.

Unglücklicherweise beruht die Sicherheit aller populären Public-Key-Verfahren auf unbewiesenen Vermutungen, wie beispielsweise der unterstellten Schwierigkeit des Faktorisierungsproblemes oder des Problemes Diskrete Logarithmen, z.B. in der multiplikativen Gruppe endlicher Körper oder Punktegruppen Elliptischer Kurven über denselben, zu berechnen. Solange niemand garantieren kann, dass diese populären Probleme auch in Zukunft schwer zu lösen, und damit für den kryptographischen Einsatz geeignet, sein werden, ist es – gerade im Hinblick auf den ständig wachsenden geschäftsmäßigen Einsatz – zwingend notwendig, alternative Public-Key-Primitive und Strukturen zu untersuchen, die auch dann noch Sicherheit bieten, falls sich eine populäre Klasse von Kryptosystemen als unsicher erweist. Idealerweise stünde eine Hierarchie schwieriger kryptographischer Probleme in Verbindung mit flexiblen Public-Key-Infrastrukturen zur Verfügung, so dass ein kryptographisches Problem, das sich durch algorithmische Entdeckungen als leicht lösbar erweist, ohne spürbaren Aufwand durch ein schwierigeres Problem ersetzt werden könnte.

Mit dieser Motivation schlugen Buchmann und Williams Ende der 80er Jahre vor, DL-artige Kryptosysteme auf Basis quadratischer Ordnungen $\mathcal{O}_\Delta$ zu konstruieren. In [**BW88**] verwendeten sie die Klassengruppe $Cl(\Delta)$, $\Delta < 0$, einer imaginärquadratischen Ordnung zur Konstruktion eines Diffie-Hellman-Analoges. Die Attraktivität dieses Ansatzes ist darin begründet, dass dieses DL-Problem in $Cl(\Delta)$, $\Delta < 0$, wie in [**Sch82**] gezeigt, mindestens so schwer und im Hinblick auf heute bekannte Algorithmen vermutlich echt schwieriger ist, wie das Faktorisieren der Diskriminante Δ. Leider hat dieser vermeintliche Sicherheitsgewinn seinen Preis; die DL-basierten Kryptosysteme in der Klassengruppe $Cl(\Delta)$ sind – verglichen mit populären Verfahren in $\mathbb{F}_q^*$ oder Elliptischen Kurven über $\mathbb{F}_q$ – relativ ineffizient.

Das Ziel dieser Arbeit ist es, Möglichkeiten zur Steigerung der Effizienz des durch Buchmann und Williams vorgeschlagenen Kryptosystemes [**BW88**] zu untersuchen und neue, effiziente und gleichsam sichere, Kryptosysteme auf Basis imaginärquadratischer Ordnungen zu konstruieren.

Vor diesem Hintergrund wurde in [**HJPT98**] eine ElGamal-Verschlüsselungsvariante auf Basis imaginärquadratischer *Nichtmaximalordnungen* vorgeschlagen, bei der die Entschlüsselung in der geheimgehaltenen Maximalordnung – und somit deutlich schneller – durchgeführt werden kann. Rückblickend bedeutete diese

scheinbar geringe Variation jedoch mehr als die offensichtliche Beschleunigung der Entschlüsselung beim bekannten Verfahren [**BW88**]. Wie die darauf aufbauenden Arbeiten [**PT98, PT00, HMT98, Hüh00, HPT99, HT99, HM00b, Hüh01, BPT99, HJW01, BST02**] belegen, führte die Verwendung von Nichtmaximalordnungen vielmehr zu einer neuartigen Klasse von Public-Key-Kryptosystemen. Es wird sich zeigen, dass die Entschlüsselung bzw. Signaturerzeugung bei diesen Verfahren der NICE[1]-Familie [**PT00, HM00b**] viel effizienter durchgeführt werden kann, als bei vielen populären Public-Key- Kryptosystemen.

Neben der ausführlichen Diskussion von Algorithmen für imaginärquadratische Nichtmaximalordnungen wird die Vorstellung der darauf aufbauenden Kryptosysteme den zweiten Schwerpunkt dieser Arbeit ausmachen.

Im Gegensatz zum reellquadratischen Fall [**BW90**] , besitzt jede Äquivalenzklasse in der Klassengruppe $Cl(\Delta)$, $\Delta < 0$, mit dem effizient berechenbaren, reduzierten Ideal, einen eindeutigen Vertreter. Deshalb wurde in [**BW88**] naheliegenderweise vorgeschlagen, DL-basierte Kryptosysteme in der endlichen, abelschen Gruppe $Cl(\Delta)$ zu konstruieren. Während die Gruppenoperation in $Cl(\Delta)$, wie in [**BB99**] gezeigt, genau wie die modulare Multiplikation, quadratische Laufzeit besitzt, so unterscheiden sich die impliziten Konstanten in der $O()$-Notation dennoch beträchtlich. Experimente belegen, dass die Gruppenoperation, und damit ein Kryptosystem, in $Cl(\Delta)$ etwa fünfzehn mal langsamer ist, als das vergleichbare Verfahren in $\mathbb{F}_p^*$.

Deshalb wurde in [**HJPT98**] vorgeschlagen, anstatt einer Fundamentaldiskriminante Δ eine Nichtfundamentaldiskriminante Δp^2 zur Konstruktion eines ElGamal-artigen Verschlüsselungsverfahren zu verwenden. Hierbei ist der Führer p der Nichtmaximalordnung $\mathcal{O}_{\Delta p^2}$ ein Teil des geheimen Schlüssels und die Entschlüsselung kann, anstatt in der öffentlichen Klassengruppe $Cl(\Delta p^2)$, in der geheimen Klassengruppe $Cl(\Delta)$ der Maximalordnung $\mathcal{O}_\Delta$ durchgeführt werden, was zu einer etwa fünffach beschleunigten Entschlüsselung führt.

Als weitere, wiederum scheinbar geringfügige, aber sehr effektive, Variation dieses Verfahrens wurde sodann von Paulus und Takagi [**PT98, PT00, HPT99**] vorgeschlagen, anstatt eines zufälligen Elementes $\mathfrak{g} \in Cl(\Delta p^2)$, zur Maskierung der Nachricht im ElGamal-Verfahren mit $\mathfrak{g}^r$, ein Element $\mathfrak{g} \in \mathrm{Ker}(\phi_{Cl}^{-1})$ zu verwenden, wobei ϕ_{Cl}^{-1} der surjektive Homomorphismus zwischen $Cl(\Delta p^2)$ und $Cl(\Delta)$ ist. Der Effekt dieses Ansatzes ist, dass zur Entschlüsselung überhaupt keine Exponentiation mehr nötig ist; die Maske $\mathfrak{g}^r$ in einem Schlüsseltext $\mathfrak{c} = \mathfrak{g}^r\mathfrak{m}$ verschwindet allein durch Berechnung von $\phi_{Cl}^{-1}(\mathfrak{c})$. Die Entschlüsselung in diesem NICE-Verfahren benötigt deshalb nur *quadratische Laufzeit*, was diesem Verfahren eine einzigartige Stellung zu bescheren scheint.

Dieses NICE-Verfahren machte deutlich, dass der geschickte Einsatz der Gruppe $\mathrm{Ker}(\phi_{Cl}^{-1}) \subseteq Cl(\Delta f^2)$ zur Konstruktion sehr effizienter Kryptosysteme eingesetzt werden kann. Deshalb war es naheliegend, sich näher mit den Eigenschaften von Elementen in $\mathrm{Ker}(\phi_{Cl}^{-1})$ und möglichen kryptographischen Konsequenzen zu beschäftigen. In [**Hüh00**] wurde gezeigt, dass für Elemente $\mathfrak{g} \in \mathrm{Ker}(\phi_{Cl}^{-1}) \subseteq Cl(\Delta f^2)$, neben der üblichen Standarddarstellung, eine Erzeugerdarstellung $\gamma = x + y\omega \in (\mathcal{O}_\Delta / f\mathcal{O}_\Delta)^*$, so dass $\varphi(\gamma) \sim \mathfrak{g}$, existiert und die relativ ineffiziente Ideal-Arithmetik

[1] NICE ist der, von Sachar Paulus vorgeschlagene, Name des Verschlüsselungsverfahrens [**PT98**] und steht für "**N**ew **I**deal **C**oset **E**ncryption"

in $\mathrm{Ker}(\phi_{Cl}^{-1})$ durch einen surjektiven Homomorphismus

$$\psi_{\mathrm{Ker}(\phi_{Cl}^{-1})} : (\mathcal{O}_\Delta / f\mathcal{O}_\Delta)^* \longrightarrow \mathrm{Ker}(\phi_{Cl}^{-1})$$

auf die effizientere Arithmetik in $(\mathcal{O}_\Delta / f\mathcal{O}_\Delta)^*$, d.h. auf wenige modulare Operationen modulo dem Führer f, zurückgeführt werden kann.

Da nun eine relativ effiziente Arithmetik für die Gruppe $\mathrm{Ker}(\phi_{Cl}^{-1})$ zur Verfgung stand, und bei primem Führer p die Ordnung $\left|\mathrm{Ker}(\phi_{Cl}^{-1})\right| = p-(\Delta/p)$ dieser Gruppe bekannt ist, war man versucht ein DSA-Analog in der Klassengruppe $Cl(\Delta p^2) = \mathrm{Ker}(\phi_{Cl}^{-1})$ einer vollständigen Nichtmaximalordnung, mit $h(\Delta) = 1$, zu konstruieren. Allerdings wurde in [**HT99**] gezeigt, dass neben dem surjektiven Homomorphismus $\psi_{\mathrm{Ker}(\phi_{Cl}^{-1})}$ noch ein Isomorphismus

$$\psi_{\mathbb{F}} : (\mathcal{O}_\Delta / p\mathcal{O}_\Delta)^* \xrightarrow{\sim} \begin{cases} \mathbb{F}_{p^2}^*, & \text{falls } \left(\frac{\Delta}{p}\right) = -1 \\ \mathbb{F}_p^* \otimes \mathbb{F}_p^*, & \text{falls } \left(\frac{\Delta}{p}\right) = 1 \end{cases}$$

existiert. Durch Komposition von $\psi_{\mathbb{F}}^{-1}$ mit $\psi_{\mathrm{Ker}(\phi_{Cl}^{-1})}$ erhält man den surjektiven Homomorphismus

$$\psi_{p^2} = \psi_{\mathbb{F}}^{-1} \circ \psi_{\mathrm{Ker}(\phi_{Cl}^{-1})} : \left\{ \begin{array}{ll} \mathbb{F}_{p^2}^*, & \text{falls } \left(\frac{\Delta}{p}\right) = -1 \\ \mathbb{F}_p^* \otimes \mathbb{F}_p^*, & \text{falls } \left(\frac{\Delta}{p}\right) = 1 \end{array} \right\} \longrightarrow \mathrm{Ker}(\phi_{Cl}^{-1}).$$

Deshalb kann die Arithmetik, und damit natürlich auch das Problem des Diskreten Logarithmus, in $\mathrm{Ker}(\phi_{Cl}^{-1})$ auf $\mathbb{F}_{p^k}^*$, $k \in \{1, 2\}$, zurückgeführt werden. Somit brächte die Verwendung von $Cl(\Delta p^2) = \mathrm{Ker}(\phi_{Cl}^{-1})$ in diesem Fall, entgegen der ursprünglichen Vermutung, keinen Sicherheitsgewinn gegenüber DL-basierten Verfahren in multiplikativen Gruppen endlicher Körper.

Auf der anderen Seite impliziert das Resultat aus [**HT99**], falls $(\Delta/p) = 1$, dass – bei Kenntnis des Führers p – für eine Exponentiation in $\mathrm{Ker}(\phi_{Cl}^{-1})$ lediglich zwei Exponentiationen modulo p notwendig sind. Deshalb wurde in [**HM00b**] vorgeschlagen, ein NICE-Schnorr-Signaturverfahren in der Klassengruppe $Cl(\Delta p^2)$, $|\Delta| \approx p$, einer gewöhnlichen Nichtmaximalordnung zu konstruieren. Somit führen ähnliche Gedankengänge, die zum „Brechen“ des DSA-Analoges in der vollständigen Nichtmaximalordnung führten, zur Konstruktion eines NICE-Schnorr-Analoges mit bemerkenswerten Eigenschaften.

Die – im Hinblick auf die Ausgangssituation – überraschend effiziente Signaturerzeugung beim NICE-Schnorr-Analog, lässt sich abermals signifikant verbessern. In [**Hüh01**] wurde für den Fall $(\Delta/p) = 1$ gezeigt, dass nicht nur ein surjektiver Homomorphismus $\psi_{p^2} : \mathbb{F}_p^* \otimes \mathbb{F}_p^* \to \mathrm{Ker}(\phi_{Cl}^{-1})$ existiert, sondern, was sich wegen $|\mathrm{Ker}(\phi_{Cl}^{-1})| = p - 1$ vermuten lässt, sogar ein Isomorphismus $\psi : \mathbb{F}_p^* \xrightarrow{\sim} \mathrm{Ker}(\phi_{Cl}^{-1})$. Deshalb wird zur Erzeugung einer NICE-Schnorr-Signatur lediglich eine modulare Exponentiation benötigt, was (wegen $p \approx |\Delta| \approx p'^{1/3}$) weniger als ein Viertel der Zeit einer Signaturerzeugung im Originalverfahren [**Sch90**] in $\mathbb{F}_{p'}^*$ beansprucht. Wie in Abschnitt 3.3.1.3 erläutert, wären bei einem Schnorr-Analog in $(\mathbb{Z}/dp^2\mathbb{Z})^*$ zwei Exponentiationen nötig.

In ähnlicher Weise kann das in [**Ham02**, Kapitel 4.1.3] vorgeschlagene Analog des Guillou-Quisquater-Signaturverfahrens [**GQ90**] in der Gruppe $\mathrm{Ker}(\phi_{Cl}^{-1})$ betrieben werden. Auch hier ist die Signaturerzeugung etwa doppelt so schnell wie beim analogen Verfahren in der Gruppe $(\mathbb{Z}/dp^2\mathbb{Z})^*$.

Die in [**HT99**] für den sehr speziellen Fall einer vollständigen Nichtmaximalordnung mit primem Führer p behandelte Reduktion des DL-Problemes, wurde in [**HJW01**] auf den allgemeinen Fall der Klassengruppe $Cl(\Delta f^2)$ einer beliebigen Nichtmaximalordnung $\mathcal{O}_{\Delta f^2}$ übertragen. Dort wurde gezeigt, dass sich das DL-Problem in $Cl(\Delta f^2)$ effizient auf das DL-Problem in $Cl(\Delta)$ und einigen endlichen Körpern zurückführen lässt. Da sich diese Teilprobleme viel leichter lösen lassen, kann somit auch ein vergleichsweise praktikables, identitätsbasiertes, nichtinteraktives Public-Key-Verfahren im Stile von [**MY91**] konstruiert werden.

Ein ähnliches Vorgehen, wie bei der Reduktion des DL-Problemes, kann auch dazu verwendet werden, um das Problem der Quadratwurzelberechnung in $Cl(\Delta f^2)$ auf entsprechende Probleme in $Cl(\Delta)$ und endlichen Körpern zu reduzieren. Dies kann wiederum zur effizienteren Implementierung des in [**HMT98**] vorgeschlagenen Rabin-Analoges eingesetzt werden.

Nach Bereitstellung der in dieser Arbeit benötigten Definitionen und einer kurzen Zusammenfassung der wichtigsten Eigenschaften imaginärquadratischer Ordnungen in Kapitel 1, gliedert sich diese Arbeit in zwei wesentliche Teile.

In Kapitel 2 werden Algorithmen für imaginärquadratische Nichtmaximalordnungen entwickelt, die in Kapitel 3 zur Konstruktion und effizienten Implementierung entsprechender Kryptosysteme eingesetzt werden.

Kapitel 2 gliedert sich folgendermaßen: In Abschnitt 2.1 geben wir konstruktive Versionen der wohlbekannten Abbildungen zwischen Idealen und Idealklassen der Nichtmaximalordnung und der Maximalordnung an. Abschnitt 2.2 ist der Entwicklung einer effizienten Arithmetik für $\mathrm{Ker}(\phi_{Cl}^{-1})$ gewidmet. Wie bereits angedeutet, erfolgte diese Entwicklung in drei Schritten. In Abschnitt 2.2.1 wird mittels $\psi_{\mathrm{Ker}(\phi_{Cl}^{-1})}$ die Arithmetik in $\mathrm{Ker}(\phi_{Cl}^{-1})$ auf Berechnungen in $(\mathcal{O}_\Delta/f\mathcal{O}_\Delta)^*$, für beliebige Führer f, zurückgeführt. In Abschnitt 2.2.2 verwenden wir den Chinesischen Restsatz in $(\mathcal{O}_\Delta/p\mathcal{O}_\Delta)^*$ zu einer weiteren Beschleunigung; durch den Homomorphismus ψ_{p^2} wird die Arithmetik in $\mathrm{Ker}(\phi_{Cl}^{-1})$ auf Berechnungen in $\mathbb{F}_{p^2}^*$, falls $(\Delta/p) = -1$, bzw. $\mathbb{F}_p^* \otimes \mathbb{F}_p^*$, falls $(\Delta/p) = 1$, zurückgeführt. In Abschnitt 2.2.3 beweisen wir schließlich Satz 2.2.11, wonach der Isomorphismus $\psi : \mathbb{F}_p^* \xrightarrow{\sim} \mathrm{Ker}(\phi_{Cl}^{-1})$, sowie sein Inverses ψ^{-1}, effizient berechnet werden kann. Wir beschließen diesen Teil der Arbeit in Abschnitt 2.2.4 mit der Präsentation von Laufzeiten einer ersten Implementierung. In Abschnitt 2.3 zeigen wir, wie DL- und Wurzelprobleme in $Cl(\Delta f^2)$ bei Kenntnis des Führers f auf analoge Probleme in $Cl(\Delta)$ und $\mathrm{Ker}(\phi_{Cl}^{-1})$ zurückgeführt werden können. Nach einer generischen Darstellung der Problemreduktionen für beliebige endliche, abelsche Gruppen G, H mit einem surjektiven Homomorphismus $\Phi : G \to H$ in Abschnitt 2.3.1, werden die Ergebnisse in Abschnitt 2.3.2 auf unseren konkreten Fall $\phi_{Cl}^{-1} : Cl(\Delta f^2) \to Cl(\Delta)$ angewandt.

In Kapitel 3 werden die zuvor eingeführten Algorithmen zur Konstruktion von teilweise sehr effizienten Kryptosystemen auf Basis imaginärquadratischer Nichtmaximalordnungen angewandt. Nach einigen übergreifenden Betrachtungen in Abschnitt 3.1, gehen wir in Abschnitt 3.2 auf das ElGamal-Verschlüsselungsverfahren aus [**HJPT98**] und in Abschnitt 3.3 auf NICE-artige Signaturverfahren ein, die in der Gruppe $\mathrm{Ker}(\phi_{Cl}^{-1}) \subset Cl(\Delta p^2)$ operieren und deshalb eine sehr effiziente

Signaturerzeugung erlauben. Neben dem NICE-Schnorr-Verfahren aus [**HM00b**] wird auch eine effiziente Variante des Guillou-Quisquater-Signaturverfahrens aus [**Ham02**, Kapitel 4.1.3] vorgestellt.

Wir beschließen diese Arbeit, indem wir die hier vorgestellten Höhepunkte, sowie noch nicht befriedigend gelöste Probleme und zukünftige Aufgaben zusammentragen.

KAPITEL 1

Grundlagen

In diesem Kapitel tragen wir die im weiteren Verlauf der Arbeit benötigten mathematischen Grundlagen zusammen.

In Abschnitt 1.1 werden verschiedene Notationen für die Beschreibung der Komplexität von Algorithmen eingeführt.

In Abschnitt 1.2 findet man die grundlegenden Definitionen und Eigenschaften imaginärquadratischer Ordnungen.

Abschnitt 1.3 widmet sich schließlich der – für diese Arbeit sehr wichtigen – Beziehung zwischen imaginärquadratischen Maximal- und Nichtmaximalordnungen.

1.1. Komplexität von Algorithmen

Für die Beschreibung der (asymptotischen) Komplexität der in dieser Arbeit vorgestellten, bzw. referenzierten, Algorithmen verwenden wir die gebräuchlichen $O()$-, $o()$- und $L_n\left[u,v\right]$-Notationen.

Wir beginnen mit der Definition der $O()$-Notation für mehrere Eingangsgrößen.

Sei $k \in \mathbb{Z}_{>0}$, $X, Y \subset \mathbb{Z}_{>0}^k$ und seien $f : X \to \mathbb{R}, g : Y \to \mathbb{R}$ Funktionen. Wir schreiben $f = O(g)$ genau dann, wenn $B, C \in \mathbb{R}_{>0}$ existieren, so dass für alle $(n_1, \cdots, n_k) \in \mathbb{Z}_{>0}^k$ mit $n_i > B$, $1 \leq i \leq k$, $(n_1, \cdots, n_k) \in X \cap Y$ und

$$f(n_1, \cdots, n_k) \leq Cg(n_1, \cdots, n_k). \tag{1.1.1}$$

Neben der $O()$-Notation wird – insbesondere zur Beschreibung der Komplexität subexponentieller Algorithmen – die im Folgenden eingeführte $L_n\left[u,v\right]$-Notation, meist in Verbindung mit der $o()$-Notation, verwendet.

Sei $x \in \mathbb{R}_{>0}$ und $f, g : \mathbb{R}_{>0} \to \mathbb{R}_{>0}$ beliebige Funktionen. Ist $\lim\limits_{x\to\infty} \frac{f(x)}{g(x)} = 0$, so schreiben wir $f(x) = o(g(x))$. Somit ist $f(x)$ eine Funktion, die strikt langsamer wächst als $g(x)$.

Sei $n \in \mathbb{R}_{>e}, v \in \mathbb{R}_{>0}$ und $0 \leq u \leq 1$. Dann definieren wir

$$L_n\left[u,v\right] = e^{v(\log x)^u(\log\log x)^{1-u}}. \tag{1.1.2}$$

Die Komplexität subexponentieller Algorithmen wird üblicherweise durch obere Schranken der Form $L_n\left[u, v + o(1)\right]$ abgeschätzt.

Um den in der Praxis benötigten Berechnungsaufwand in grober Art und Weise abschätzen zu können, vernachlässigt man meist – wie z.B. in [**Odl95, LV01, Len01, Ham02**] – den Einfluss des $o(1)$-Anteiles und unterstellt, dass

$$\frac{L_{n_1}\left[u_1, v_1\right]}{L_{n_2}\left[u_2, v_2\right]} = \frac{\text{Aufwand}_1}{\text{Aufwand}_2}. \tag{1.1.3}$$

1.2. Grundlagen imaginärquadratischer Ordnungen

In diesem Abschnitt wollen wir die benötigten Grundlagen imaginärquadratischer Körper und Ordnungen zusammentragen. Für eine systematische Behandlung der Materie und Beweise der wohlbekannten Eigenschaften imaginärquadratischer Ordnungen verweisen wir auf Standard-Referenzen, wie beispielsweise [**BS66, Bue89, Coh95, Cox89, Hua82, Zag81**].

DEFINITION 1.2.1. *Ist* $\Delta \in \mathbb{Z}, \Delta \equiv 0,1 \pmod 4$*, und kein Quadrat, so nennt man* Δ *eine* (quadratische) Diskriminante*. Ist* Δ *außerdem – abgesehen von einem möglichen Faktor* 4 *– quadratfrei, so nennt man* Δ *eine* Fundamentaldiskriminante*.*

DEFINITION 1.2.2. *Die* quadratische Ordnung *der Diskriminante* Δ *ist als das* $\mathbb{Z}$*-Modul*

$$\mathcal{O}_\Delta = \mathbb{Z} + \frac{\Delta + \sqrt{\Delta}}{2}\mathbb{Z}$$

definiert.

Setzt man

$$\omega = \begin{cases} \frac{\sqrt{\Delta}}{2}, & \text{falls } \Delta \equiv 0 \pmod 4, \\ \frac{1+\sqrt{\Delta}}{2}, & \text{falls } \Delta \equiv 1 \pmod 4, \end{cases} \tag{1.2.1}$$

so ist $\mathcal{O}_\Delta = \mathbb{Z} + \omega\mathbb{Z}$ und jede Zahl $\alpha \in \mathcal{O}_\Delta$ erlaubt die eindeutige Darstellung $\alpha = x + y\omega$, wobei $x, y \in \mathbb{Z}$.

Ist $\Delta < 0$, so nennt man $\mathcal{O}_\Delta$ eine *imaginärquadratische* Ordnung. Man nennt $\mathcal{O}_\Delta$ *Maximalordnung*, wenn $\mathcal{O}_\Delta$ nicht in einer größeren quadratischen Ordnung enthalten ist. Jede Ordnung $\mathcal{O}$ lässt sich eindeutig als

$$\mathcal{O} = \mathbb{Z} + f\mathcal{O}_\Delta, \tag{1.2.2}$$

darstellen, wobei $f \in \mathbb{Z}_{\geq 1}$ und $\mathcal{O}_\Delta$ die Maximalordnung ist. Hierbei nennt man f den *Führer* von $\mathcal{O}$. Falls $f > 1$, nennt man $\mathcal{O}$ eine Nichtmaximalordnung. Nichtmaximalordnungen bezeichnen wir in dieser Arbeit mit $\mathcal{O}_{\Delta f^2}$. Es ist leicht zu sehen, dass $\mathcal{O}_\Delta$ genau dann eine Maximalordnung ist, wenn Δ eine Fundamentaldiskriminante ist. Für eine ausführlichere Diskussion der Beziehung einer imaginärquadratischen Nichtmaximalordnung mit ihrer Maximalordnung verweisen wir auf Abschnitt 1.3.

DEFINITION 1.2.3. *Sei* Δ *eine Fundamentaldiskriminante. Dann ist der* quadratische Körper *der Diskriminante* Δ *als das* $\mathbb{Q}$*-Modul*

$$\mathbb{Q}(\sqrt{\Delta}) = \mathbb{Q} + \sqrt{\Delta}\mathbb{Q}$$

definiert.

Die Maximalordnung $\mathcal{O}_\Delta$ ist der Ring der ganzen Zahlen in $\mathbb{Q}(\sqrt{\Delta})$.

Jede Zahl $\alpha \in \mathbb{Q}(\sqrt{\Delta})$ lässt sich in der *Standarddarstellung*

$$\alpha = \frac{x + y\sqrt{\Delta}}{z}, \tag{1.2.3}$$

mit $x, y, z \in \mathbb{Z}$ angeben. Fordert man weiterhin $\mathrm{ggT}(x, y, z) = 1$ und $z > 0$, so wird diese Darstellung eindeutig. Ist $z = 2$, so ist $\alpha \in \mathcal{O}_\Delta$.

Das *Konjugierte* von α ist durch

$$\overline{\alpha} = \frac{x - y\sqrt{\Delta}}{z}, \tag{1.2.4}$$

und die *Norm* von α durch

$$N(\alpha) = \overline{\alpha}\alpha = \frac{x^2 - y^2\Delta}{z^2} \tag{1.2.5}$$

definiert.

Für $\alpha, \beta \in \mathbb{Q}(\sqrt{\Delta})$ gilt

$$N(\alpha)N(\beta) = N(\alpha\beta). \tag{1.2.6}$$

Eine Zahl $\mu \in \mathcal{O}_\Delta$ nennt man *Einheit*, wenn ein $\mu' \in \mathcal{O}_\Delta$ existiert, so dass $\mu\mu' = 1$. Ist μ eine Einheit, dann folgt aus (1.2.6), dass $N(\mu) = \pm 1$.

In einer imaginärquadratischen Ordnung existieren nur endlich viele Einheiten μ, wobei

$$\mu = \begin{cases} \pm 1, \frac{\pm 1 \pm i\sqrt{3}}{2}, & \text{falls } \Delta = -3, \\ \pm 1, \pm i, & \text{falls } \Delta = -4, \\ \pm 1, & \text{falls } \Delta < -4. \end{cases} \tag{1.2.7}$$

Ein vom Nullideal verschiedenes $\mathcal{O}_\Delta$-Ideal $\mathfrak{a}$ lässt sich in der Standarddarstellung

$$\mathfrak{a} = z\left(a\mathbb{Z} + \frac{b + \sqrt{\Delta}}{2}\mathbb{Z}\right), \tag{1.2.8}$$

wobei $a, b, z \in \mathbb{Z}, a, z > 0, c = (b^2 - \Delta)/(4a) \in \mathbb{Z}, \mathrm{ggT}(a, b, c) = 1$, darstellen. Hierbei ist b modulo $2a$ eindeutig bestimmt. Ist außerdem $\Delta < 0$ und

$$-a < b \leq a, \tag{1.2.9}$$

so nennt man $\mathfrak{a}$ *normal* und die Darstellung (1.2.8) ist eindeutig. Wird im Folgenden von einem $\mathcal{O}_\Delta$-Ideal in Standarddarstellung gesprochen, so meinen wir (1.2.8), wobei zusätzlich (1.2.9) gilt.

Die *Norm* $\mathcal{N}(\mathfrak{a})$ eines Ideales $\mathfrak{a}$ ist definiert als die Ordnung des Ringes $\mathcal{O}_\Delta/\mathfrak{a}$. Für Ideale in Standarddarstellung ist $\mathcal{N}(\mathfrak{a}) = z^2 a$.

Für Primideale $\mathfrak{p}$ von $\mathcal{O}_\Delta$ lässt sich zeigen, dass $\mathfrak{p} \cap \mathbb{Z} = p\mathbb{Z}$, wobei p eine rationale Primzahl ist. Deshalb ist in der Standarddarstellung (1.2.8) eines Primideales $\mathfrak{p}$ entweder $z = p$ prim und $a = 1$, oder $z = 1$ und $a = p$ prim. In beiden Fällen sagen wir, dass $\mathfrak{p}$ *über* p *liegt*.

Nun charakterisieren wir alle Primideale $\mathfrak{p}$ von $\mathcal{O}_\Delta$. Dafür benötigen wir das Kronecker-Symbol.

Definition 1.2.4. *Sei* p *prim und* $\Delta \equiv 0, 1 \pmod 4$ *eine Diskriminante. Dann ist das* Kronecker-Symbol *folgendermaßen definiert:*

$$\left(\frac{\Delta}{p}\right) = \begin{cases} 0 & \text{falls } p \mid \Delta, \\ 1 & \text{falls } p \nmid \Delta \text{ und } b^2 \equiv \Delta \pmod{4p} \text{ lösbar ist}, \\ -1 & \text{falls } p \nmid \Delta \text{ und } b^2 \equiv \Delta \pmod{4p} \text{ nicht lösbar ist}. \end{cases}$$

Es sei angemerkt, dass das Kronecker-Symbol $\left(\frac{\Delta}{p}\right)$ bei ungeraden Primzahlen $p \nmid \Delta$ gleich dem Jacobi-Symbol ist. Aus ästhetischen Gründen schreiben wir gelegentlich auch (Δ/p) für das Kronecker-Symbol.

Das Kronecker-Symbol $\left(\frac{\Delta}{p}\right)$ bestimmt das Zerlegungsverhalten des Ideales $p\mathcal{O}_\Delta$:

1. Ist $\left(\frac{\Delta}{p}\right) \in \{0,1\}$, so besitzt das Ideal $p\mathcal{O}_\Delta$ die Zerlegung
$$p\mathcal{O}_\Delta = \left(p\mathbb{Z} + \frac{b+\sqrt{\Delta}}{2}\mathbb{Z}\right)\left(p\mathbb{Z} + \frac{\overline{b}+\sqrt{\Delta}}{2}\mathbb{Z}\right),$$
wobei $b \equiv -\overline{b} \pmod{4p}$.
   - Falls $\left(\frac{\Delta}{p}\right) = 1$, so ist $b \not\equiv \overline{b} \pmod{4p}$ und die Primzahl p wird *zerlegt* genannt.
   - Falls $\left(\frac{\Delta}{p}\right) = 0$, so ist $b \equiv \overline{b} \pmod{4p}$ und die Primzahl p wird *verzweigt* genannt.
2. Ist $\left(\frac{\Delta}{p}\right) = -1$, so ist $p\mathcal{O}_\Delta$ ein Primideal und man nennt die Primzahl p *träge*.

Eine Teilmenge $\mathfrak{a} \subset \mathbb{Q}(\sqrt{\Delta})$ nennt man *gebrochenes Ideal*, wenn ein $d \in \mathbb{Z}_{>0}$ existiert, so dass $d\mathfrak{a}$ ein $\mathcal{O}_\Delta$-Ideal ist.

Sei m, a, b wie oben und $d \in \mathbb{Z}_{>0}, q = m/d \in \mathbb{Q}$. Dann ist

$$\mathfrak{a} = q\left(a\mathbb{Z} + \frac{b+\sqrt{\Delta}}{2}\mathbb{Z}\right) \tag{1.2.10}$$

die Standarddarstellung eines gebrochenen Ideales. Hier definiert man $\mathcal{N}(\mathfrak{a}) = \frac{1}{N(d)}\mathcal{N}(d\mathfrak{a}) = z^2a/d^2 = q^2a$.

Definition 1.2.5. *Ist $q = 1$, so nennt man das Ideal $\mathfrak{a}$* primitiv*. Ein primitives Ideal einer imaginärquadratischen Ordnung nennt man* reduziert*, wenn*

$$|b| \le a \le c \text{ und } b \ge 0 \text{ falls } a = c. \tag{1.2.11}$$

Das *Produktideal* $\mathfrak{ab}$ zweier Ideale $\mathfrak{a}, \mathfrak{b}$ ist definiert als das kleinste Ideal, das alle Produkte ab, wobei $a \in \mathfrak{a}, b \in \mathfrak{b}$, enthält. Es gilt $\mathcal{N}(\mathfrak{ab}) = \mathcal{N}(\mathfrak{a})\mathcal{N}(\mathfrak{b})$.

Ein Ideal $\mathfrak{a}$ nennt man invertierbar, wenn ein Ideal $\mathfrak{a}^{-1}$ existiert, so dass $\mathfrak{a}\mathfrak{a}^{-1} = \mathcal{O}_\Delta$. Für ein Ideal $\mathfrak{a}$ in Standarddarstellung (1.2.10) ist das *inverse Ideal* als

$$\mathfrak{a}^{-1} = \frac{q}{\mathcal{N}(\mathfrak{a})}\left(a\mathbb{Z} + \frac{-b+\sqrt{\Delta}}{2}\mathbb{Z}\right) \tag{1.2.12}$$

gegeben.

Die Menge der invertierbaren Ideale bezeichnen wir mit $\mathcal{I}_\Delta$. $\mathcal{I}_\Delta$ bildet bezüglich der Idealmultiplikation eine unendliche, abelsche Gruppe mit neutralem Element $\mathcal{O}_\Delta$.

Sei $\gamma \in \mathbb{Q}(\sqrt{\Delta})^*$. Dann nennt man das Ideal $\mathfrak{a} = \gamma\mathcal{O}_\Delta$ *Hauptideal* und γ den *Erzeuger* von $\mathfrak{a}$. Es gilt $\mathcal{N}(\mathfrak{a}) = |N(\gamma)|$.

Die Menge der invertierbaren Hauptideale bezeichnen wir mit $\mathcal{P}_\Delta$. $\mathcal{P}_\Delta$ bildet eine Untergruppe von $\mathcal{I}_\Delta$.

Definition 1.2.6. *Zwei gebrochene Ideale $\mathfrak{a}, \mathfrak{b}$ nennt man* äquivalent*, falls ein $\gamma \in \mathbb{Q}(\sqrt{\Delta})^*$ existiert, so dass $\mathfrak{a} = \gamma\mathfrak{b}$.*

Definition 1.2.7. *Die* Klassengruppe *$Cl(\Delta)$ ist definiert als die Faktorgruppe $Cl(\Delta) = \mathcal{I}_\Delta/\mathcal{P}_\Delta$. Die Ordnung der Klassengruppe $Cl(\Delta)$ nennt man die* Klassenzahl *$h(\Delta)$.*

$Cl(\Delta)$ ist eine endliche abelsche Gruppe.

1.3. Die Beziehung zwischen Maximal- und Nichtmaximalordnungen

In diesem Abschnitt wollen wir einige später benötigte Begriffe und Resultate über die Beziehung zwischen einer Maximalordnung $\mathcal{O}_\Delta$ und Nichtmaximalordnungen $\mathcal{O}_{\Delta f^2}$ mit Führer f zusammentragen. Für eine ausführlichere Darstellung verweisen wir auf [**Cox89**, §7].

Ist $\mathcal{O}_\Delta$ die Maximalordnung, dann lässt sich – wie in Abschnitt 1.2 erwähnt – jede Ordnung im imaginärquadratischen Zahlkörper $\mathbb{Q}(\sqrt{\Delta})$ eindeutig als

$$\mathcal{O} = \mathbb{Z} + f\mathcal{O}_\Delta, \tag{1.3.1}$$

mit einer ganzen Zahl $f \geq 1$ darstellen. Ist der Führer $f > 1$, so ist $\mathcal{O}$ eine *Nichtmaximalordnung*, die wir mit $\mathcal{O}_{\Delta f^2}$ bezeichnen.

Ist die Klassengruppe $Cl(\Delta)$ einer Maximalordnung trivial, also $h(\Delta) = 1$, so wird $\mathcal{O}_{\Delta f^2}$ eine *vollständige Nichtmaximalordnung* genannt.

Wir nennen ein $\mathcal{O}_{\Delta f^2}$-Ideal $\mathfrak{a}$ genau dann *teilerfremd, oder prim, zum Führer* f, wenn $\mathrm{ggT}(\mathcal{N}(\mathfrak{a}), f) = 1$. Alle solchen Ideale sind invertierbar und besitzen eine eindeutige Primidealzerlegung. Sei $\mathcal{I}_{\Delta f^2}$ die Gruppe der $\mathcal{O}_{\Delta f^2}$-Ideale und $\mathcal{I}_{\Delta f^2}(f)$ die Menge der zum Führer teilerfremden $\mathcal{O}_{\Delta f^2}$-Ideale. So ist $\mathcal{I}_{\Delta f^2}(f)$ eine Untergruppe von $\mathcal{I}_{\Delta f^2}$. In $\mathcal{I}_{\Delta f^2}(f)$ gibt es eine weitere Untergruppe - die Untergruppe der zum Führer teilerfremden Hauptideale, die wir mit $\mathcal{P}_{\Delta f^2}(f)$ bezeichnen wollen.

PROPOSITION 1.3.1. *Es gibt einen Isomorphismus*

$$\mathcal{I}_{\Delta f^2}(f) \Big/ \mathcal{P}_{\Delta f^2}(f) \simeq \mathcal{I}_{\Delta f^2} \Big/ \mathcal{P}_{\Delta f^2} = Cl(\Delta f^2) \ .$$

BEWEIS. Siehe [**Cox89**, Proposition 7.19, Seite 143]. □

Das folgende Resultat setzt Ideale in Maximal- und Nichtmaximalordnungen miteinander in Beziehung und ist für alles Folgende von zentraler Bedeutung.

PROPOSITION 1.3.2. *Sei $\mathcal{O}_\Delta$ eine Maximalordnung und $\mathcal{O}_{\Delta f^2}$ eine zugehörige Nichtmaximalordnung mit Führer f.*

1. *Sei $\mathfrak{A}$ ein zum Führer f teilerfremdes $\mathcal{O}_\Delta$-Ideal. Dann ist $\mathfrak{a} = \mathfrak{A} \cap \mathcal{O}_{\Delta f^2}$ ein $\mathcal{O}_{\Delta f^2}$-Ideal und es gilt $\mathcal{N}(\mathfrak{A}) = \mathcal{N}(\mathfrak{a})$.*
2. *Sei $\mathfrak{a}$ ein zum Führer f teilerfremdes $\mathcal{O}_{\Delta f^2}$-Ideal. Dann ist $\mathfrak{A} = \mathfrak{a}\mathcal{O}_\Delta$ ein $\mathcal{O}_\Delta$-Ideal und es gilt $\mathcal{N}(\mathfrak{a}) = \mathcal{N}(\mathfrak{A})$.*
3. *Die Abbildung $\varphi : \mathfrak{A} \mapsto \mathfrak{A} \cap \mathcal{O}_{\Delta f^2}$ induziert einen Isomorphismus $\mathcal{I}_\Delta(f) \xrightarrow{\sim} \mathcal{I}_{\Delta f^2}(f)$. Die inverse Abbildung ist $\varphi^{-1} : \mathfrak{a} \mapsto \mathfrak{a}\mathcal{O}_\Delta$.*

BEWEIS. Siehe [**Cox89**, Proposition 7.20, Seite 144]. □

Außerdem ist die Voraussetzung für die Anwendbarkeit von φ, dass ein Ideal teilerfremd zum Führer sein muss – im Bezug auf Äquivalenzklassen – kein unüberwindbares Hinderniss.

PROPOSITION 1.3.3. *Sei $\mathcal{O}_\Delta$ eine beliebige imaginärquadratische Maximal- oder Nichtmaximalordnung und f eine beliebige ganze Zahl. Dann enthält jede Äquivalenzklasse in $Cl(\Delta)$ ein $\mathcal{O}_\Delta$-Ideal, das zu f teilerfremd ist.*

BEWEIS. Siehe [**Cox89**, Corollary 7.17, Seite 142]. □

Um die in Kapitel 3 ausführlich diskutierten Kryptosysteme auf Basis imaginärquadratischer Nichtmaximalordnungen auch effizient implementieren zu können, werden wir in Abschnitt 2.1 konstruktive Beweise von Proposition 1.3.2 und Proposition 1.3.3 – in Form von Algorithmen – angeben.

BEMERKUNG 1.3.4. Es sei darauf hingewiesen, dass der Isomorphismus φ zwischen den Ideal-Gruppen $\mathcal{I}_\Delta(f)$ und $\mathcal{I}_{\Delta f^2}(f)$ definiert ist – *nicht zwischen den Klassengruppen* $Cl(\Delta)$ und $Cl(\Delta f^2)$. Für äquivalente Ideale $\mathfrak{A}, \mathfrak{B} \in \mathcal{I}_\Delta(f)$ ist *nicht notwendigerweise* auch deren Bild $\varphi(\mathfrak{A}), \varphi(\mathfrak{B})$ in der Nichtmaximalordnung äquivalent.

Auf der anderen Seite bleibt die Äquivalenz bei Anwendung von φ^{-1} gewahrt und es gilt:

PROPOSITION 1.3.5. *Der Isomorphismus* φ^{-1} *induziert einen* surjektiven Homomorphismus $\phi_{Cl}^{-1} : Cl(\Delta f^2) \to Cl(\Delta)$, *wobei* $\mathfrak{a} \mapsto \rho(\varphi^{-1}(\mathfrak{a}))$.

BEWEIS. Das ist Teil der – in [**Neu92**, Satz 12.9, S. 82] bewiesenen – kurzen exakten Sequenz

$$Cl(\Delta f^2) \longrightarrow Cl(\Delta) \longrightarrow 1.$$

□

Hierbei bezeichnet $\rho()$ den „Reduktionsoperator“, der aus einem beliebigen Ideal den eindeutigen, reduzierten Vertreter der Klassengruppe errechnet.

Obwohl ϕ_{Cl}^{-1}, wie in Bemerkung 1.3.4 angedeutet, im Allgemeinen nicht injektiv – also kein Isomorphismus – ist, so existiert – wie von Takagi beobachtet – eine bijektive Korrespondenz der reduzierten Ideale in den jeweiligen Idealklassen, sofern man sich auf Ideale mit *kleiner Norm* beschränkt.

PROPOSITION 1.3.6. *Sei* $\mathfrak{A}$ *ein reduziertes, zum Führer* f *teilerfremdes,* $\mathcal{O}_\Delta$*-Ideal. Dann ist* $\varphi(\mathfrak{A})$ *ein reduziertes* $\mathcal{O}_{\Delta f^2}$*-Ideal.*

BEWEIS. Falls $f = 1$, so ist dies trivial. Sei nun also $f \geq 2$.

Da $\mathfrak{A}$ reduziert ist, folgt aus [**Coh95**, Lemma 5.3.4(1), Seite 227], dass $\mathcal{N}(\mathfrak{A}) \leq \sqrt{|\Delta|/3}$. Durch Proposition 1.3.2 ist $\mathfrak{a} = \varphi(\mathfrak{A})$, wobei $\mathcal{N}(\mathfrak{a}) = \mathcal{N}(\mathfrak{A}) \leq \sqrt{|\Delta|/3} < \sqrt{|\Delta f^2|/4}$. Deshalb ist $\mathfrak{a}$ nach [**Coh95**, Lemma 5.3.4(2)] reduziert. □

PROPOSITION 1.3.7. *Sei* $\mathfrak{a}$ *ein reduziertes, zum Führer* f *teilerfremdes,* $\mathcal{O}_{\Delta f^2}$*-Ideal mit* $\mathcal{N}(\mathfrak{a}) < \sqrt{|\Delta|/4}$. *Dann ist* $\mathfrak{A} = \varphi^{-1}(\mathfrak{a})$ *ein reduziertes* $\mathcal{O}_\Delta$*-Ideal.*

BEWEIS. Nach Proposition 1.3.2 ist $\mathcal{N}(\mathfrak{A}) = \mathcal{N}(\mathfrak{a}) < \sqrt{|\Delta|/4}$. Deshalb ist $\mathfrak{A}$, nach [**Coh95**, Lemma 5.3.4(2)], reduziert. □

Die Menge der Idealklassen in $Cl(\Delta p^2)$, für die die Norm des reduzierten Vertreters $\mathfrak{a}$ die Ungleichung $\mathcal{N}(\mathfrak{a}) < \sqrt{|\Delta|/4}$ erfüllt, bezeichnen wir mit $\mathcal{SI}(\Delta p^2)$.

Ist $\mathfrak{a} \in \mathcal{SI}(\Delta p^2)$, dann gilt insbesondere

$$\varphi(\phi_{Cl}^{-1}(\mathfrak{a})) = \mathfrak{a}. \tag{1.3.2}$$

Im Folgenden wollen wir uns den Kern der in Proposition 1.3.5 definierten Abbildung ϕ_{Cl}^{-1}, also die Beziehung zwischen einer Klasse in $Cl(\Delta)$ und den $|\mathrm{Ker}(\phi_{Cl}^{-1})|$ vielen dazugehörigen Klassen in $Cl(\Delta f^2)$, etwas genauer ansehen.

$\mathrm{Ker}(\phi_{Cl}^{-1})$ wird eine zentrale Rolle für die Kryptosysteme der NICE-Familie spielen. Hierbei wird nämlich im Wesentlichen die Gruppe $\mathbb{F}_p^*$ bei den klassischen ElGamal Signatur- und Verschlüsselungsverfahren durch $\mathrm{Ker}(\phi_{Cl}^{-1})$ ersetzt, was eine sehr effiziente Entschlüsselung und Signaturerzeugung ermöglicht.

PROPOSITION 1.3.8. *Sei $\mathcal{O}_{\Delta f^2}$ eine Ordnung mit Führer f. Dann gibt es den folgenden Isomorphismus*

$$Cl(\Delta f^2) \simeq {\mathcal{I}_{\Delta f^2}(f)}\Big/{\mathcal{P}_{\Delta f^2}(f)} \simeq {\mathcal{I}_{\Delta}(f)}\Big/{\mathcal{P}_{\Delta,\mathbb{Z}}(f)},$$

wobei $\mathcal{P}_{\Delta,\mathbb{Z}}(f)$ die Untergruppe von $\mathcal{I}_{\Delta}(f)$ bezeichnet, die durch Hauptideale $\alpha\mathcal{O}_{\Delta}$ mit $\alpha \equiv a \pmod{f\mathcal{O}_{\Delta}}$, $a \in \mathbb{Z}$, $\mathrm{ggT}(a, f) = 1$, erzeugt wird.

BEWEIS. Siehe [**Cox89**, Proposition 7.22, Seite 145]. □

Mit dieser Interpretation von $Cl(\Delta f^2)$ ist es nun einfach, zu entscheiden welche Hauptideale in $\mathcal{O}_{\Delta}$ nach Abbildung mit φ wiederum Hauptideale in der Nichtmaximalordnung $\mathcal{O}_{\Delta f^2}$ sind.

Das folgende Kriterium folgt sofort aus Proposition 1.3.8:

KOROLLAR 1.3.9. *Sei $\alpha \in \mathcal{O}_{\Delta}$ mit $\mathrm{ggT}(N(\alpha), f) = 1$. Dann ist $\varphi(\alpha\mathcal{O}_{\Delta}) \sim \mathcal{O}_{\Delta f^2}$ genau dann, wenn*

$$\alpha \equiv a \pmod{f\mathcal{O}_{\Delta}},$$

wobei $a \in \mathbb{Z}$ und $\mathrm{ggT}(a, f) = 1$

Somit ist es möglich, die Äquivalenzrelation in der Klassengruppe $Cl(\Delta f^2)$ allein durch Betrachten von Elementen der Maximalordnung zu „modellieren“. Dieser Umstand wird *Ringäquivalenz* genannt, und ist bei der in Abschnitt 2.2 vorgestellten Arithmetik für $\mathrm{Ker}(\phi_{Cl}^{-1})$ – und den daraus resultierenden Reduktionen in Abschnitt 2.3 – von zentraler Bedeutung.

Schließlich wollen wir die exakte Beziehung zwischen den Klassenzahlen $h(\Delta)$ und $h(\Delta f^2)$ angeben:

SATZ 1.3.10. *Sei $\mathcal{O}_{\Delta f^2}$ eine Ordnung mit Führer f im imaginärquadratischen Körper $\mathbb{Q}(\sqrt{\Delta})$ mit Maximalordnung $\mathcal{O}_{\Delta}$. Dann ist*

$$h(\Delta f^2) = \frac{h(\Delta) f}{[\mathcal{O}_{\Delta}^* : \mathcal{O}_{\Delta f^2}^*]} \prod_{p \mid f} \left(1 - \frac{\left(\frac{\Delta}{p}\right)}{p}\right) = |\mathrm{Ker}(\phi_{Cl}^{-1})| \cdot h(\Delta),$$

wobei $\left(\frac{\Delta}{p}\right)$ das Kronecker-Symbol bezeichnet.

BEWEIS. Siehe [**Cox89**, Theorem 7.24, Seite 146]. □

KAPITEL 2

Algorithmen für imaginärquadratische Nichtmaximalordnungen

In diesem Kapitel bezeichnet $\Delta < 0$ eine feste Fundamentaldiskriminante, $\mathcal{O}_\Delta$ eine imaginärquadratische Maximalordnung und $f \in \mathbb{Z}_{>1}$ den Führer der Nichtmaximalordnung $\mathcal{O}_{\Delta f^2}$.

Im Folgenden werden Algorithmen für imaginärquadratische Nichtmaximalordnungen diskutiert, die wir für die Konstruktion und effiziente Implementierung der in Kapitel 3 vorgestellten Kryptosysteme anwenden wollen.

In Abschnitt 2.1 findet man eine konstruktive Version des wohlbekannten Isomorphismus φ zwischen $\mathcal{O}_\Delta$- und $\mathcal{O}_{\Delta f^2}$-Idealen, die teilerfremd zum Führer f sind. Dieser Isomorphismus induziert einen surjektiven Homomorphismus $\phi_{Cl}^{-1} : Cl(\Delta f^2) \to Cl(\Delta)$, dessen Kern für die Konstruktion der NICE-Kryptosysteme von großer Bedeutung ist, da er $\mathbb{F}_p^*$ in klassischen DL-Protokollen ersetzt. In Abschnitt 2.2 diskutieren wir eine neuartige und sehr effiziente Arithmetik für Elemente in $\mathrm{Ker}(\phi_{Cl}^{-1})$, die den in Abschnitt 3.3 vorgestellten Signatursystemen zugute kommt. In Abschnitt 2.3.1 wollen wir zeigen, wie die Berechnung Diskreter Logarithmen und Quadratwurzeln in $Cl(\Delta f^2)$ auf analoge, aber effizienter lösbare, Probleme in $Cl(\Delta)$ und $\mathrm{Ker}(\phi_{Cl}^{-1})$ zurückgeführt werden kann, wobei sich die Probleme in $\mathrm{Ker}(\phi_{Cl}^{-1})$ wiederum auf analoge Probleme in endlichen Körpern zurückführen lassen. Die Reduktion des DL-Problemes wird in Abschnitt 3.2.4 ausgenutzt, um ein vergleichsweise praktikables, nicht-interaktives Public-Key-Kryptosystem zu konstruieren. Die Reduktion des Quadratwurzel-Problemes kann zur effizienten Implementierung des in [**HMT98**] vorgestellten Rabin-Verfahrens eingesetzt werden.

2.1. Abbildungen zwischen Maximal- und Nichtmaximalordnung

In diesem Abschnitt wollen wir Algorithmen für einige Abbildungen zwischen Maximal- und Nichtmaximalordnungen diskutieren.

Wir beginnen mit dem, in Proposition 1.3.2 – für zum Führer teilerfremde Ideale – angegebenen, Isomorphismus φ.

Aus Proposition 1.3.3 wissen wir, dass man in jeder vorgegebenen Idealklasse ein Ideal finden kann, welches teilerfremd zum Führer ist. Im Folgenden wollen wir in Proposition 2.1.2 einen konstruktiven Beweis für diese Aussage liefern und schließlich mit FindIdealPrimeTo einen Algorithmus für dieses Problem angeben.

Wir wollen uns auf den Fall eines primen Führers p beschränken und verweisen für den – etwas weniger effizienten – allgemeinen Fall auf [**Mey97**, Lemma 6.42, S. 136] und [**Lag80**, S. 172 ff].

Zum Beweis von Proposition 2.1.2 benötigen wir ein einfaches Hilfsresultat:

LEMMA 2.1.1. *Sei* $\mathfrak{a} = \mathbb{Z} + \frac{b+\sqrt{\Delta}}{2a}\mathbb{Z}$ *ein gebrochenes Ideal,* $c = (b^2 - \Delta)/(4a)$ *und* $\mathfrak{b} = \mathbb{Z} + \frac{-b+\sqrt{\Delta}}{2c}\mathbb{Z}$. *Dann ist* $\mathfrak{a} = \gamma\mathfrak{b}$, *wobei* $\gamma = \frac{b+\sqrt{\Delta}}{2a}$.

BEWEIS. Es gilt

$$\begin{aligned}
\gamma\mathfrak{b} &= \frac{b+\sqrt{\Delta}}{2a}\left(\mathbb{Z}+\frac{-b+\sqrt{\Delta}}{2c}\mathbb{Z}\right)\\
&= \frac{b+\sqrt{\Delta}}{2a}\mathbb{Z}+\frac{b+\sqrt{\Delta}}{2a}\frac{-b+\sqrt{\Delta}}{2c}\mathbb{Z}\\
&= \frac{b+\sqrt{\Delta}}{2a}\mathbb{Z}+\frac{-b^2+\Delta}{4ac}\mathbb{Z}\\
&= \frac{b+\sqrt{\Delta}}{2a}\mathbb{Z}+\mathbb{Z}\\
&= \mathfrak{a}
\end{aligned}$$

□

PROPOSITION 2.1.2. *Sei $\mathfrak{a}$ ein primitives $\mathcal{O}_\Delta$-Ideal und p eine Primzahl. Dann kann mit $O(\log(\max\{a,|\Delta|\})^2)$ Bitoperationen ein primitives $\mathcal{O}_\Delta$-Ideal $\mathfrak{A}\sim\mathfrak{a}$ berechnet werden, so dass $p\not|\,\mathcal{N}(\mathfrak{A})$.*

BEWEIS. Sei $\mathfrak{a}=(a,b)$ die Standarddarstellung von $\mathfrak{a}$. Wir nehmen an, dass $p\,|\,a$, da andernfalls $\mathfrak{a}$ bereits die gewünschte Eigenschaft aufweisen würde.

Sei $c=(b^2-\Delta)/4a$. Dann zeigen wir, dass entweder das Ideal $\mathfrak{A}_1=(A_1,B_1)=c\mathbb{Z}+\frac{-b+\sqrt{\Delta}}{2}\mathbb{Z}$ oder das Ideal $\mathfrak{A}_2=(A_2,B_2)=(a+b+c)\mathbb{Z}+\frac{-b-2a+\sqrt{\Delta}}{2}\mathbb{Z}$ die gewünschte Eigenschaft aufweist.

Hierfür zeigen wir zuerst, dass zumindest eine der drei Zahlen $a,c,a+b+c$ teilerfremd zu p ist. Nach der Definition der Standarddarstellung eines Ideals $\mathfrak{a}=(a,b)$ in (1.2.8) gilt $\mathrm{ggT}(a,b,c)=1$. Angenommen $p\,|\,a$ und $p\,|\,c$. Dann würde $p\,|\,(a+b+c)$ auch $p\,|\,b$ implizieren. Das wäre ein Widerspruch zu $\mathrm{ggT}(a,b,c)=1$. Somit ist zumindest eine der Zahlen $a,c,a+b+c$ teilerfremd zu p.

Es bleibt schließlich zu zeigen, dass $\mathfrak{A}_1$ und $\mathfrak{A}_2$ primitive Ideale in der Äquivalenzklasse von $\mathfrak{a}$ sind.

Sei $\theta=\frac{b+\sqrt{\Delta}}{2a}$. Dann folgt sofort aus Lemma 2.1.1, dass $\mathfrak{a}=\frac{a\theta}{c}\mathfrak{A}_1$. Außerdem ist

$$\begin{aligned}
\mathfrak{a} &= a\,(\mathbb{Z}+\theta\mathbb{Z})\\
&= a\,(\mathbb{Z}+(\theta+1)\mathbb{Z})\\
&= a\,(\theta+1)\left(\mathbb{Z}+\frac{1}{\theta+1}\mathbb{Z}\right)\\
&= \frac{a\,(\theta+1)}{a+b+c}\left((a+b+c)\mathbb{Z}+\frac{a+b+c}{\theta+1}\mathbb{Z}\right)\\
&= \frac{a\,(\theta+1)}{a+b+c}\left((a+b+c)\mathbb{Z}+\frac{2a(a+b+c)}{b+2a+\sqrt{\Delta}}\mathbb{Z}\right)\\
&= \frac{a\,(\theta+1)}{a+b+c}\left((a+b+c)\mathbb{Z}+\frac{2a(a+b+c)(b+2a-\sqrt{\Delta})}{4a(a+b+c)}\mathbb{Z}\right)\\
&= \frac{a\,(\theta+1)}{a+b+c}\left((a+b+c)\mathbb{Z}+\frac{-b-2a+\sqrt{\Delta}}{2}\mathbb{Z}\right)\\
&= \frac{a\,(\theta+1)}{a+b+c}\mathfrak{A}_2,
\end{aligned}$$

womit die Äquivalenz gezeigt ist. Nun muss noch gezeigt werden, dass für die Koeffizienten $A, B, C = (B^2 - \Delta)/4A$ auch $\mathrm{ggT}(A, B, C) = 1$ gilt, damit es sich bei $\mathfrak{A} = (A, B)$ um die Standarddarstellung eines primitiven Ideales handelt.

$\mathfrak{A}_1 = (c, -b)$ ist offenichtlich ein Ideal in Standarddarstellung, da $\mathrm{ggT}(c, -b, a) = 1$. Nun betrachten wir $\mathfrak{A}_2$. Da $C = ((-b - 2a)^2 - \Delta)/(4(a + b + c)) = a$, zeigt ein analoges Argument wie oben, dass $\mathrm{ggT}(a + b + c, -b - 2a, a) = 1$.

Die aufwendigsten Operationen sind die Berechnung von $c = (b^2 - \Delta)/4a$ und die Versuchsdivisionen mit a und ggf. c. Da $c = O(\max\{a, |\Delta|\})$, benötigt die Berechnung von c, wie auch die Versuchsdivision mit a und möglicherweise auch c, $O(\log(\max\{a, |\Delta|\})^2)$ Bitoperationen . □

Der folgende Algorithmus berechnet bei Eingabe eines Ideals $\mathfrak{a}$ und einer Primzahl p ein zu $\mathfrak{a}$ äquivalentes Ideal, das teilerfremd zu p ist.

Algorithmus 1 FindIdealPrimeTo

Eingabe: Ein primitives $\mathcal{O}_\Delta$-Ideal $\mathfrak{a} = (a, b)$ und eine Primzahl p.
Ausgabe: Ein primitives $\mathcal{O}_\Delta$-Ideal $\mathfrak{A} = (A, B) \sim \mathfrak{a}$, so dass $p \nmid \mathcal{N}(\mathfrak{A})$.

```
if p | a then
  c ← (b^2 − Δ)/4a
  if p | c then {Berechne 𝔄_2}
    A ← a + b + c
    B ← −b − 2a
  else {Berechne 𝔄_1}
    A ← c
    B ← −b
  end if
  return(A, B)
end if
return(a, b)
```

Es sei angemerkt, dass der Algorithmus für Ideale in Maximal- und Nichtmaximal-Ordnungen gleichermaßen eingesetzt werden kann.

Nun wollen wir Algorithmen für die Berechnung des Isomorphismus $\varphi : \mathcal{I}_\Delta(f) \xrightarrow{\sim} \mathcal{I}_{\Delta f^2}(f)$ aus Proposition 1.3.2 und dessen Umkehrung φ^{-1} angeben. Die beiden folgenden Algorithmen sind für beliebige Führer f gültig.

Algorithmus 2 GoToNonMaxOrder

Eingabe: Ein primitives $\mathcal{O}_\Delta$-Ideal $\mathfrak{A} = (A, B)$ und den Führer f.
Ausgabe: Ein primitives $\mathcal{O}_{\Delta f^2}$-Ideal $\mathfrak{a} = \varphi(\mathfrak{B}) = (\mathfrak{B}) \cap \mathcal{O}_{\Delta f^2} = (a, b)$, wobei $\mathfrak{B} \sim \mathfrak{A}$ und $\mathrm{ggT}(\mathcal{N}(\mathfrak{B}), f) = 1$.

```
(a, b) ← FindIdealPrimeTo(𝔄, f)
b ← bf (mod 2a)
return(a, b)
```

Beweis der Korrektheit. Nach der Ausführung von FindIdealPrimeTo haben wir $\mathrm{ggT}(a, f) = 1$ und dürfen φ aus Proposition 1.3.2 berechnen. $\mathcal{N}(\mathfrak{B}) = \mathcal{N}(\mathfrak{a}) = a$ folgt aus Proposition 1.3.2(1). Die Berechnung von b folgt nun aus (1.3.1) und der Eindeutigkeit von $b \pmod{2a}$. □

PROPOSITION 2.1.3. *GoToNonMaxOrder benötigt* $O(\log(\max\{A, |\Delta|\})^2)$ *Bitoperationen.*

BEWEIS. Das folgt sofort aus Proposition 2.1.2. □

Der umgekehrte Schritt von der Nichtmaximal-Ordnung zurück zur Maximalordnung ist ebenso einfach.

Algorithmus 3 GoToMaxOrder

Eingabe: Ein primitives $\mathcal{O}_{\Delta f^2}$-Ideal $\mathfrak{a} = (a, b)$ und den Führer f.
Ausgabe: Ein primitives $\mathcal{O}_\Delta$-Ideal $\mathfrak{A} = \varphi^{-1}(\mathfrak{b}) = (\mathfrak{b})\mathcal{O}_\Delta = (A, B)$, wobei $\mathfrak{b} \sim \mathfrak{a}$ und $\mathrm{ggT}(\mathcal{N}(\mathfrak{b}), f) = 1$.

$(A, B) \leftarrow$FindIdealPrimeTo$(\mathfrak{a}, f)$
Löse $1 = \mu f + \lambda A$, wobei $\mu, \lambda \in \mathbb{Z}$
$B \leftarrow B\mu + A\Delta\lambda \bmod 2A$
return(A, B)

BEWEIS DER KORREKTHEIT. Nach Anwendung von FindIdealPrimeTo haben wir $\mathrm{ggT}(A, f) = 1$ und dürfen deshalb φ^{-1} aus Proposition 1.3.2 anwenden. Nach Proposition 1.3.2(2) haben wir $\mathcal{N}(\mathfrak{b}) = \mathcal{N}(\mathfrak{A}) = A$. Nun ist also noch die Berechnung von B zu betrachten. Wir berechnen μ, so dass $\mu \equiv f^{-1} \pmod{A}$. Diese Inversion ist immer möglich, da $\mathrm{ggT}(A, f) = 1$. Die Form von B ergibt sich nun aus dem Standard Algorithmus für die Ideal-Multplikation (siehe z.B. [**Coh95**, Lemma 5.4.5]). □

Sofern der Führer bekannt ist, kann man also zur Maximalordnung wechseln, wo sich Berechnungen aufgrund kleinerer Koeffizienten effizienter durchführen lassen. Dieser Gedankengang war die ursprüngliche Motivation für die Verwendung von Nichtmaximalordnungen, wie erstmals in [**HJPT98**] vorgeschlagen.

PROPOSITION 2.1.4. *GoToMaxOrder benötigt* $O(\log(\max\{a, |\Delta| f^2\})^2)$ *Bitoperationen.*

BEWEIS. Das folgt aus Proposition 2.1.2 und [**BS96**, Corollary 4.3.3], wonach die ggT-Berechnung auch quadratische Laufzeit hat. □

Für den in der Praxis relevanten Fall eines primen Führers, kann bei reduzierten Idealen – wie von Takagi vorgeschlagen – auf den Aufruf von FindIdealPrimeTo verzichtet werden, sofern der Führer ausreichend groß ist.

LEMMA 2.1.5. *Sei p prim, so dass $\sqrt{|\Delta|} < p$. Dann ist jedes reduzierte Ideal in $Cl(\Delta p^2)$ und $Cl(\Delta)$ teilerfremd zu p.*

BEWEIS. Da $\sqrt{|\Delta|} < p$ sieht man sofort, dass alle reduzierten Ideale in $Cl(\Delta)$ teilerfremd zu p sind. Es bleibt nun, die Behauptung für reduzierte Ideale in $Cl(\Delta p^2)$ zu zeigen.

Sei $\mathfrak{a} = (a, b)$ ein reduziertes Ideal in $Cl(\Delta p^2)$ und $c = (b^2 - \Delta p^2)/4a$. $\mathrm{ggT}(a, p) > 1$ bedeutet $p \mid a$, da p prim ist. Aus $b^2 - 4ac = \Delta p^2$ sehen wir, dass $p \mid b$. Nach [**Coh95**, Lemma 5.3.4] wissen wir, dass $a < \sqrt{|\Delta| p^2/3} = p\sqrt{|\Delta|/3}$, was wiederum $p^2 \nmid a$ impliziert, da wir nach Voraussetzung $\sqrt{|\Delta|/3} < \sqrt{|\Delta|} < p$ haben. Da $p \mid b$ und $4ac = b^2 - \Delta p^2$, sehen wir, dass $p \mid c$. Dies wiederum ist ein Widerspruch zu $\mathrm{ggT}(a, b, c) = 1$. □

Somit erhält man für den in der Praxis meist angewandten Fall eine etwas schärfere Laufzeit-Schranke.

PROPOSITION 2.1.6. *Sei p prim, so dass $\sqrt{|\Delta|} < p$. Sei $\mathfrak{A}$ ein reduziertes $\mathcal{O}_\Delta$-Ideal und $\mathfrak{a}$ ein reduziertes $\mathcal{O}_{\Delta p^2}$-Ideal. Dann benötigen* GoToNonMaxOrder$(\mathfrak{A}, p)$ *und* GoToMaxOrder$(\mathfrak{a}, p)$ *jeweils $O(\log(p)^2)$ Bitoperationen.*

Schließlich wollen wir einen einfachen Algorithmus für den durch φ induzierten surjektiven Homomorphismus $\phi_{Cl}^{-1} : Cl(\Delta f^2) \to Cl(\Delta)$ angeben. Hierbei stützen wir uns auf eine Reduktionsroutine Reduce, die bei Eingabe eines $\mathcal{O}_\Delta$-Ideales $\mathfrak{a} = (a, b)$ in Standarddarstellung den reduzierten Vertreter der Äquivalenzklasse von $\mathfrak{a}$ berechnet. Wie in [**BB99**] gezeigt, benötigt Reduce $O(\log(\max\{a, |\Delta|\})^2)$ Bitoperationen.

Weitere Varianten von Reduce wurden beispielsweise in [**Ham02**, Kapitel 7.1.2], [**Jac99**, Section 2.1] und [**Coh95**, Section 5.4.2] diskutiert.

Algorithmus 4 GoToMaxCl

Eingabe: Ein reduziertes $\mathcal{O}_{\Delta f^2}$-Ideal $\mathfrak{a}$ und den Führer f.
Ausgabe: Ein reduziertes $\mathcal{O}_\Delta$-Ideal $\mathfrak{A} = \phi_{Cl}^{-1}(\mathfrak{a})$.

$\mathfrak{A}' \leftarrow$ GoToMaxOrder$(\mathfrak{a}, f)$
$\mathfrak{A} \leftarrow$ Reduce$(\mathfrak{A}')$
return$(\mathfrak{A})$

Da der Algorithmus Reduce nach [**BB99**] quadratische Laufzeit hat, erhält man sofort die folgende Aussage:

PROPOSITION 2.1.7. *Sei p prim, so dass $\sqrt{|\Delta|} < p$ und $\mathfrak{a}$ ein reduziertes $\mathcal{O}_{\Delta p^2}$-Ideal. Dann benötigt* GoToMaxCl$(\mathfrak{a}, p)$ *$O(\log(p)^2)$ Bitoperationen.*

2.2. Effiziente Arithmetik in der Gruppe $\mathrm{Ker}(\phi_{Cl}^{-1})$

In diesem Abschnitt werden wir eine neuartige und sehr effiziente – in den Arbeiten [**Hüh00, HT99, HM00b, Hüh01**] entwickelte – Arithmetik für die Idealklassen-Gruppe $\mathrm{Ker}(\phi_{Cl}^{-1}) \subseteq Cl(\Delta f^2)$ vorstellen.

Der Ausgangspunkt [**Hüh00**] für die Entwicklung dieser Arithmetik ist der effizient berechenbare, surjektive Homomorphismus $\psi_{\mathrm{Ker}(\phi_{Cl}^{-1})} : (\mathcal{O}_\Delta / f\mathcal{O}_\Delta)^* \to \mathrm{Ker}(\phi_{Cl}^{-1})$, wodurch die Gruppenoperation in $\mathrm{Ker}(\phi_{Cl}^{-1})$ auf die effizientere Gruppenoperation in $(\mathcal{O}_\Delta / f\mathcal{O}_\Delta)^*$, d.h. auf wenige modulare Operationen, reduziert werden kann. Wie in [**HT99, HM00b**] gezeigt wurde, impliziert der Chinesische Restsatz in $(\mathcal{O}_\Delta / p\mathcal{O}_\Delta)^*$, für prime Führer p, den effizient berechenbaren, surjektiven Homomorphismus $\psi_{p^2} : \mathbb{F}_{p^2}^* \to \mathrm{Ker}(\phi_{Cl}^{-1})$, falls $(\Delta/p) = -1$, bzw. $\psi_{p^2} : \mathbb{F}_p^* \otimes \mathbb{F}_p^* \to \mathrm{Ker}(\phi_{Cl}^{-1})$, falls $(\Delta/p) = 1$. Der letztere Fall bedeutet, dass die Signaturerzeugung beim – in Abschnitt 3.3.1 diskutierten – NICE-Schnorr-Verfahren [**HM00b**] im Wesentlichen durch zwei modulare Exponentiationen durchgeführt werden kann. In [**Hüh01**] wurde schließlich gezeigt, dass in diesem Fall sogar ein effizient berechenbarer Isomorphismus $\psi : \mathbb{F}_p^* \xrightarrow{\sim} \mathrm{Ker}(\phi_{Cl}^{-1})$ existiert, wonach zur Signaturerzeugung lediglich *eine* Modulo-Exponentiation nötig ist.

Neben dieser kryptographischen Anwendung zur effizienten Signaturerzeugung in [**HM00b**], führt die in diesem Zusammenhang eingeführte Ideal-Darstellung (2.2.1)

zur – in Abschnitt 2.3 näher diskutierten – Reduktion des DL- bzw. Quadratwurzelproblemes in $Cl(\Delta f^2)$ auf $Cl(\Delta)$ und wenige endliche Körper. Die Reduktion des DL-Problemes werden wir in Abschnitt 3.2.4 zur Konstruktion von identitätsbasierten ElGamal-Varianten verwenden. Die Reduktion des Quadratwurzelproblemes kann zur effizienten Implementierung des Rabin-Analoges aus [**HMT98**] angewandt werden.

Wir werden die Vorstellung der Arithmetik in drei Teile gliedern: In Abschnitt 2.2.1 werden wir zeigen, dass die Arithmetik in $\mathrm{Ker}(\phi_{Cl}^{-1})$ effizient auf die Arithmetik in $(\mathcal{O}_\Delta/f\mathcal{O}_\Delta)^*$ reduziert werden kann. In Abschnitt 2.2.2 verwenden wir den Chinesischen Restsatz in $(\mathcal{O}_\Delta/f\mathcal{O}_\Delta)^*$ zur abermaligen Effizienzsteigerung. In Abschnitt 2.2.3 werden wir schließlich zeigen, dass für $(\Delta/p) = 1$ ein effizient berechenbarer Isomorphismus $\mathrm{Ker}(\phi_{Cl}^{-1}) \tilde{=} \mathbb{F}_p^*$ existiert.

2.2.1. Reduktion der Arithmetik von $\mathrm{Ker}(\phi_{Cl}^{-1})$ auf $(\mathcal{O}_\Delta/f\mathcal{O}_\Delta)^*$

In diesem Abschnitt wollen wir zeigen, dass sich die Arithmetik in $\mathrm{Ker}(\phi_{Cl}^{-1})$ auf Gruppenoperationen in $(\mathcal{O}_\Delta/f\mathcal{O}_\Delta)^*$ zurückführen lässt.

Die initiale Beobachtung ist, dass – wie man sofort aus der Definition von ϕ_{Cl}^{-1} in Proposition 1.3.5 sieht – jedes ganze Ideal $\mathfrak{a} \in \mathrm{Ker}(\phi_{Cl}^{-1}) \subseteq Cl(\Delta f^2)$ mit $\mathrm{ggT}(\mathcal{N}(\mathfrak{a}), f) = 1$ in folgender *Erzeugerdarstellung* angegeben werden kann:

$$\mathfrak{a} = \varphi(\alpha\mathcal{O}_\Delta), \tag{2.2.1}$$

wobei $\alpha = x + y\omega \in \mathcal{O}_\Delta$, $x, y \in \mathbb{Z}$ und ω wie in (1.2.1).

Da φ ein Isomorphismus ist, ist somit auch sofort die Multiplikation von Idealen $\mathfrak{a} = \varphi(\alpha\mathcal{O}_\Delta), \mathfrak{b} = \varphi(\beta\mathcal{O}_\Delta) \in \mathrm{Ker}(\phi_{Cl}^{-1})$ in dieser Darstellung als $\mathfrak{ab} = \varphi(\alpha\beta\mathcal{O}_\Delta)$ erklärt.

Bemerkung 2.2.1. Es sollte angemerkt werden, dass die Multiplikation von Zahlen aus $\mathcal{O}_\Delta$ in aller Regel effizienter ist, als die Multiplikation von $\mathcal{O}_\Delta$-Idealen, da nicht der relativ ineffiziente Erweiterte Euklidische Algorithmus angewandt werden muss.

Da alle in Kapitel 3 diskutierten Kryptosysteme in der Klassengruppe operieren, wollen wir im Folgenden die Reduktion und Äquivalenz von Idealen in obiger Erzeugerdarstellung etwas näher betrachten.

Proposition 2.2.2. *Sei $\alpha, \beta \in \mathcal{O}_\Delta$, wobei $\mathrm{ggT}(N(\alpha), f) = \mathrm{ggT}(N(\beta), f) = 1$ und $\alpha \equiv \beta \pmod{f\mathcal{O}_\Delta}$. Dann ist $\varphi(\alpha\mathcal{O}_\Delta) \sim \varphi(\beta\mathcal{O}_\Delta)$ in $Cl(\Delta f^2)$.*

Beweis. Sei $\alpha, \beta \in \mathcal{O}_\Delta$ mit $\mathrm{ggT}(N(\alpha), f) = \mathrm{ggT}(N(\beta), f) = 1$. Dann bedeutet $\alpha \equiv \beta \pmod{f\mathcal{O}_\Delta}$ nichts anderes als $\alpha = \beta + f\gamma$, wobei $\gamma \in \mathcal{O}_\Delta$. Daraus folgt

$$\alpha = \beta\delta, \text{ wobei } \delta = 1 + f\gamma/\beta \in \mathbb{Q}(\sqrt{\Delta}). \tag{2.2.2}$$

Sei $\overline{\beta}$ das Konjugierte und $N(\beta) = \beta\overline{\beta}$ die Norm von β. Dann ist

$$\begin{aligned} \delta N(\beta) &= (1 + f\gamma/\beta)N(\beta) \\ &= N(\beta) + f\gamma\overline{\beta} \end{aligned}$$

und man sieht sofort, dass $\delta N(\beta) \in \mathcal{O}_\Delta$ und $\delta N(\beta) \equiv N(\beta) \pmod{f\mathcal{O}_\Delta}$. Außerdem erhalten wir aus (2.2.2)

$$\alpha N(\beta) = \beta\delta N(\beta) = \beta(N(\beta) + f\gamma\overline{\beta}).$$

Da $N(\beta) \in \mathbb{Z}$ und nach Voraussetzung $\mathrm{ggT}(N(\beta), f) = 1$, erhalten wir unter Verwendung von Korollar 1.3.9

$$\varphi(\alpha\mathcal{O}_\Delta) \sim \varphi(\alpha N(\beta)\mathcal{O}_\Delta) = \varphi(\beta\delta N(\beta)\mathcal{O}_\Delta) \sim \varphi(\beta\mathcal{O}_\Delta)$$

□

Dies führt zu folgender Repräsentation von Idealklassen in $\mathrm{Ker}(\phi_{Cl}^{-1})$:

DEFINITION 2.2.3. *Sei* $\mathfrak{a} = \varphi(\alpha) \in \mathrm{Ker}(\phi_{Cl}^{-1})$, *wobei* $\alpha = x + y\omega \in (\mathcal{O}_\Delta/f\mathcal{O}_\Delta)^*$. *Dann nennen wir das Paar* (x, y) *eine* Erzeugerdarstellung für die Idealklasse *von* $\mathfrak{a}$.

BEMERKUNG 2.2.4.
1. Da nach Proposition 1.3.3 in jeder Klasse ein zum Führer teilerfremdes (ganzes) Ideal gefunden werden kann, existiert für jede Idealklasse in $\mathrm{Ker}(\phi_{Cl}^{-1})$ (mindestens) eine solche Erzeugerdarstellung.
2. Es bleibt die Frage nach der Eindeutigkeit dieser Erzeugerdarstellung. Betrachtet man umgekehrt die Äquivalenenz $\varphi(\alpha\mathcal{O}_\Delta) \sim \varphi(\beta\mathcal{O}_\Delta)$, so wird aus Korollar 1.3.9 deutlich, dass dies *nicht* $\alpha \equiv \beta \pmod{f\mathcal{O}_\Delta}$ impliziert, sondern lediglich $\alpha \equiv k\beta \pmod{f\mathcal{O}_\Delta}$ für ein $k \in \mathbb{Z}$ mit $\mathrm{ggT}(k, f) = 1$. Deshalb ist die Erzeugerdarstellung einer Klasse aus $\mathrm{Ker}(\phi_{Cl}^{-1})$ durch ein Element $\alpha = x + y\omega \in (\mathcal{O}_\Delta/f\mathcal{O}_\Delta)^*$ *nicht eindeutig*. Im Hinblick auf unsere spätere Anwendung für die effiziente Signaturerzeugung ist das jedoch kein Problem. Außerdem werden wir genau diese Mehrdeutigkeit in Abschnitt 2.2.3 „ausnutzen“, um zu zeigen, wie der Isomorphismus $\mathrm{Ker}(\phi_{Cl}^{-1}) \tilde{=} \mathbb{F}_p^*$, für $(\Delta/p) = 1$ effizient berechnet werden kann.

Nun wollen wir das zentrale Resultat dieses Abschnittes formulieren, das es erlaubt die Arithmetik in $\mathrm{Ker}(\phi_{Cl}^{-1})$ auf die effizientere Arithmetik in $(\mathcal{O}_\Delta/f\mathcal{O}_\Delta)^*$ zurückzuführen.

SATZ 2.2.5. *Die Abbildung* $\psi_{\mathrm{Ker}(\phi_{Cl}^{-1})} : (\mathcal{O}_\Delta/f\mathcal{O}_\Delta)^* \to \mathrm{Ker}(\phi_{Cl}^{-1})$, $\alpha \mapsto \varphi(\alpha\mathcal{O}_\Delta)$ *ist ein surjektiver Homomorphismus.*

BEWEIS. Aus Proposition 2.2.2 und der Darstellung (2.2.1) folgt, dass $\psi_{\mathrm{Ker}(\phi_{Cl}^{-1})}$ ein wohldefinierter Homomorphismus ist und es bleibt zu zeigen, dass es sich um einen surjektiven handelt. Wir nehmen an, dass $\psi_{\mathrm{Ker}(\phi_{Cl}^{-1})}$ nicht surjektiv ist. Das heisst es existiert eine durch ein reduziertes Ideal $\mathfrak{a} \in \mathrm{Ker}(\phi_{Cl}^{-1})$ repräsentierte Idealklasse, die kein Urbild $\alpha \in (\mathcal{O}_\Delta/f\mathcal{O}_\Delta)^*$ unter $\psi_{\mathrm{Ker}(\phi_{Cl}^{-1})}$ besitzt. Für dieses $\mathfrak{a} \in \mathrm{Ker}(\phi_{Cl}^{-1})$ existiert also kein $\alpha \in \mathcal{O}_\Delta$, so dass $\mathrm{ggT}(N(\alpha), f) = 1$ und $\varphi(\alpha\mathcal{O}_\Delta) \sim \mathfrak{a}$. Das ist ein Widerspruch zur Existenz eines zum vorgegebenen Führer teilerfremden Ideales in einer gegebenen Idealklasse (Proposition 1.3.3). □

Die Richtigkeit des folgenden Algorithmus für die Gruppenoperation in $(\mathcal{O}_\Delta/f\mathcal{O}_\Delta)^*$ lässt sich durch einfaches Nachrechnen überprüfen.

Algorithmus 5 Mult-mod-f

Eingabe: Zwei Elemente $\alpha_i = x_i + y_i\omega \in (\mathcal{O}_\Delta/f\mathcal{O}_\Delta)^*$ für $i \in \{1,2\}$ und den Führer f.
Ausgabe: Das Produkt $\beta = x + y\omega = \alpha_1\alpha_2 \in (\mathcal{O}_\Delta/f\mathcal{O}_\Delta)^*$.

if $\Delta \equiv 0 \pmod 4$ **then**
  $x \equiv x_1x_2 + y_1y_2\frac{\Delta}{4} \pmod f$
  $y \equiv x_1y_2 + x_2y_1 \pmod f$
else $\{\Delta \equiv 1 \pmod 4\}$
  $x \equiv x_1x_2 + y_1y_2\frac{\Delta-1}{4} \pmod f$
  $y \equiv x_1y_2 + x_2y_1 + y_1y_2 \pmod f$
end if
return(x, y)

Auch die asymptotische Laufzeit dieser Gruppenoperation ist sofort ersichtlich:

PROPOSITION 2.2.6. *Mult-mod-f benötigt $O(\log(f)^2)$ Bitoperationen.*

In analoger Weise könnte auch eine geringfügig effizientere Routine zur Quadrierung von Elementen in $(\mathcal{O}_\Delta/f\mathcal{O}_\Delta)^*$ angegeben werden. Da hier aber keine signifikanten Laufzeitverbesserungen zu erwarten sind und insbesondere die asymptotische Laufzeit identisch ist, wollen wir auf die explizite Darstellung verzichten und die Routine Mult-mod-f auch zum Quadrieren verwenden.

Da Ideale üblicherweise in der Standarddarstellung (1.2.10) angegeben werden und in dieser Arbeit aber auch die Erzeugerdarstellung (2.2.1) benötigt wird, wollen wir nun noch Konvertierungsroutinen zwischen den beiden Darstellungen angeben.

Hierfür stützen wir uns auf einen Reduktionsalgorithmus ReduceWithGen, der bei Eingabe eines $\mathcal{O}_\Delta$-Ideales $\mathfrak{a} = (a, b)$ in Standarddarstellung zusätzlich zum reduzierten Vertreter $\mathfrak{A} = \gamma\mathfrak{a}$ auch den relativen Erzeuger $\gamma = \frac{x+y\sqrt{\Delta}}{z} \in \mathbb{Q}\sqrt{\Delta}$ in Standarddarstellung (1.2.3) zurückliefert.

Ein solcher Algorithmus, der sich aus der in [**BB99**] analysierten Reduktionsroutine oder [**Jac99**, Algorithm 2.6] ableiten lässt, findet sich beispielsweise in der LiDIA-Bibliothek [**LiD**]. Nach [**BB99**] benötigt ReduceWithGen $O(\log(\max\{a, |\Delta|\})^2)$ Bitoperationen.

PROPOSITION 2.2.7. *Mit Gen2Std und Std2Gen existieren Algorithmen, die es in $O(\log(|\Delta|f^2)^2)$ Bitoperationen erlauben, zwischen einer Erzeugerdarstellung (2.2.1) und der Standarddarstellung (1.2.10) einer Idealklasse $\mathfrak{a} \in \mathrm{Ker}(\phi_{Cl}^{-1})$ zu wechseln.*

BEWEIS. Wir beginnen mit Algorithmus Gen2Std, der aus einer Erzeugerdarstellung die Standarddarstellung berechnet.

Sei (x, y) die gegebene Erzeugerdarstellung der Idealklasse von $\mathfrak{a} = \varphi(\alpha\mathcal{O}_\Delta)$, wobei $\alpha = x + y\omega \in (\mathcal{O}_\Delta/f\mathcal{O}_\Delta)^*$.

Dann berechnen wir zuerst die Standarddarstellung (d, A, B) des Hauptideals $\mathfrak{A}' = d\left(A\mathbb{Z} + \frac{B+\sqrt{\Delta}}{2}\mathbb{Z}\right) = \alpha\mathcal{O}_\Delta$ in der Maximalordnung $\mathcal{O}_\Delta$.

Dies erreichen wir, indem wir x', y', so dass $\alpha = x + y\omega = \frac{x'+y'\sqrt{\Delta}}{2}$, $d = \mathrm{ggT}(y', (x' + y'\Delta)/2) = \lambda y' + \mu(x' + y'\Delta)/2$, mit $\lambda, \mu \in \mathbb{Z}$, und schließlich $A = |x'^2 - \Delta y'^2|/(4d^2)$ und $B = (\lambda x' + \mu(x' + y')\Delta/2)/d \pmod{2A}$ berechnen.

Es bleibt noch die Berechnung von $\varphi(\mathfrak{A}')$ und die Reduktion des Ergebnisses.

Da $\alpha \in (\mathcal{O}_\Delta/f\mathcal{O}_\Delta)^*$, ist $|N(\alpha)| = \mathcal{N}(\alpha\mathcal{O}_\Delta) = Ad^2$ teilerfremd zu f. Da $\mathrm{ggT}(Ad^2, f) = 1$ insbesondere auch $\mathrm{ggT}(d, f) = 1$ impliziert, ist wegen Korollar 1.3.9 $\varphi(\alpha\mathcal{O}_\Delta) = \varphi(\mathfrak{A}') \sim \varphi(\mathfrak{A})$, wobei $\mathfrak{A} = A\mathbb{Z} + \frac{B+\sqrt{\Delta}}{2}\mathbb{Z}$. Deshalb kann $\mathfrak{a}' = \varphi(\mathfrak{A}) \sim \varphi(\mathfrak{A}')$ mit GotoNonMaxOrder berechnet und schließlich mit Reduce zu $\mathfrak{a}$ reduziert werden.

Da $A = O(|\Delta|f^2)$, benötigt die Berechnung der Standarddarstellung des Hauptideales $O(\log(|\Delta|f^2)^2)$ Bitoperationen. Da $\max\{A, |\Delta|\} = O(|\Delta|f^2)$, gilt diese Laufzeitschranke, nach Proposition 2.1.3, auch für GotoNonMaxOrder und nach [**BB99**] schließlich auch für Reduce. Dies zeigt, dass Gen2Std für die Berechnung der Standarddarstellung (1.2.10) bei einer gegebenen Erzeugerdarstellung (2.2.1) $O(\log(|\Delta|f^2)^2)$ Bitoperationen benötigt.

Nun betrachten wir den umgekehrten Weg, bei dem der Algorithmus Std2Gen die Standarddarstellung (a, b) der Idealklasse von $\mathfrak{a} \in \mathrm{Ker}(\phi_{Cl}^{-1})$ als Eingabe erhält und daraus eine Erzeugerdarstellung (x, y), mit $\alpha = x + y\omega \in (\mathcal{O}_\Delta/f\mathcal{O}_\Delta)^*$ und $\varphi(\alpha) \sim \mathfrak{a}$, berechnet.

Zu Beginn wird mittels der in GoToMaxOrder aufgerufenen Routine FindIdealPrimeTo (Algorithmus 1) ein zu $\mathfrak{a}$ äquivalentes Ideal $\mathfrak{a}'$ mit $\mathrm{ggT}(\mathcal{N}(\mathfrak{a}'), f) = 1$ ermittelt. Somit kann die Standarddarstellung des Hauptideales $\mathfrak{A} = \alpha\mathcal{O}_\Delta = \varphi^{-1}(\mathfrak{a}')$ berechnet werden. Durch Anwendung von ReduceWithGen erhalten wir $\alpha = \frac{x'+y'\sqrt{\Delta}}{z'}$ in Standarddarstellung (1.2.3). Schließlich bringen wir α in die gewünschte Form $\alpha = x + y\omega$ und dürfen x und y, nach Proposition 2.2.2, modulo f reduzieren ohne die Äquivalenzklasse von $\mathfrak{a}$ zu verlassen.

Schließlich müssen wir auch hier die Laufzeit von Std2Gen betrachten. Da $\mathfrak{a}$ reduziert – und deshalb $\mathcal{N}(\mathfrak{a}) < |\Delta|f^2$ – ist, benötigt der Aufruf von GoToMaxOrder nach Proposition 2.1.4 $O(\log(|\Delta|f^2)^2)$ Bitoperationen.

Es gilt $\mathcal{N}(\mathfrak{A}) = \mathcal{N}(\mathfrak{a}') = O(|\Delta|f^2|)$. Deshalb benötigt ReduceWithGen nach [**BB99**], wie natürlich auch die abschließende Berechnung von $\alpha \pmod{f\mathcal{O}_\Delta}$, $O(\log(|\Delta|f^2)^2)$ Bitoperationen. □

Algorithmus 6 Gen2Std

Eingabe: Eine Erzeugerdarstellung (x, y) einer Klasse $\mathfrak{a} \sim \varphi(x + y\omega) \in \mathrm{Ker}(\phi_{Cl}^{-1})$, $x + y\omega \in (\mathcal{O}_\Delta/f\mathcal{O}_\Delta)^*$ und den Führer f.

Ausgabe: Die Standarddarstellung (a, b) des reduzierten Ideales $\mathfrak{a} = a\mathbb{Z} + \frac{b+\sqrt{\Delta f^2}}{2}\mathbb{Z}$.

$x \leftarrow 2x$ {Benutze $\frac{x+y\sqrt{\Delta}}{2}$-Form}
if $\Delta \equiv 1 \pmod 4$ **then**
  $x \leftarrow x + y$
end if
Berechne $d \leftarrow \mathrm{ggT}(y, (x + y\Delta)/2) = \lambda y + \mu(x + y\Delta)/2$, wobei $\lambda, \mu \in \mathbb{Z}$
$A \leftarrow |x^2 - \Delta y^2|/(4d^2)$
$B \leftarrow (\lambda x + \mu(x + y)\Delta/2)/d \pmod{2A}$
$\mathfrak{a}' \leftarrow$GotoNonMaxOrder$((A, B), f)$
$\mathfrak{a} \leftarrow$Reduce$(\mathfrak{a}')$
return$(\mathfrak{a})$

Algorithmus 7 Std2Gen

Eingabe: Die Standarddarstellung (a, b) eines reduzierten $\mathcal{O}_{\Delta f^2}$-Ideales $\mathfrak{a} = a\mathbb{Z} + \frac{b+\sqrt{\Delta f^2}}{2}\mathbb{Z} \in \mathrm{Ker}(\phi_{Cl}^{-1})$ und den Führer f.

Ausgabe: Eine Erzeugerdarstellung (x, y) der Klasse $\mathfrak{a} \sim \varphi(x + y\omega) \in \mathrm{Ker}(\phi_{Cl}^{-1})$, wobei $x + y\omega \in (\mathcal{O}_\Delta / f\mathcal{O}_\Delta)^*$.

$\mathfrak{A} \leftarrow$ GoToMaxOrder$(\mathfrak{a}, f)$
$(\mathfrak{g}, (x', y', z')) \leftarrow$ReduceWithGen$(\mathfrak{A})$
if $\mathfrak{g} \neq \mathcal{O}_\Delta$ **then**
  return("Fehler! $\mathfrak{a} \notin \mathrm{Ker}(\phi_{Cl}^{-1})$!")
end if
if $\Delta \equiv 0 \pmod 4$ **then**
  $x \leftarrow x'/2 \pmod f$
  $y \leftarrow y'/2 \pmod f$
else
  $x \leftarrow (x' - y')/2 \pmod f$
  $y \leftarrow y' \pmod f$
end if
return(x, y)

Mit den nun bereitgestellten Mitteln ist es leicht, eine – in Abschnitt 3.3 für die Signaturerzeugung eingesetzte – Exponentiationsroutine für Elemente aus $\mathrm{Ker}(\phi_{Cl}^{-1})$ anzugeben. Die nachfolgende Routine verwendet die klassische „Square & Multiply“ Strategie, die durch ein beliebiges Exponentiationsverfahren für abelsche Gruppen ersetzt werden könnte.

Sei $n \in \mathbb{Z}_{>0}$ und $l = \lfloor \log_2(n) \rfloor$. Dann bezeichnet $(n_l, \cdots, n_0)$ die Binärentwicklung von $n = \sum_{i=0}^{l} n_i 2^i$, wobei $n_i \in \{0, 1\}$ für $0 \leq i \leq l$.

Algorithmus 8 GenExp

Eingabe: Eine Erzeugerdarstellung (x, y) der Klasse $\mathfrak{a} \sim \varphi(x + y\omega) \in \mathrm{Ker}(\phi_{Cl}^{-1})$, wobei $x + y\omega \in (\mathcal{O}_\Delta / f\mathcal{O}_\Delta)^*$, sowie den Führer f und den Exponenten $n = (n_l, \cdots, n_0) \in \mathbb{Z}_{>0}$.

Ausgabe: Die Standarddarstellung (a, b) des reduzierten Vertreters der Klasse von $\mathfrak{a}^n \in \mathrm{Ker}(\phi_{Cl}^{-1})$.

$(x_h, y_h) \leftarrow (x, y)$
for $i = l - 1$ **downto** 0 **do**
  $(x, y) \leftarrow$Mult-mod-f$((x, y), (x, y), f)$ {Square}
  if $n_i = 1$ **then**
    $(x, y) \leftarrow$Mult-mod-f$((x, y), (x_h, y_h), f)$ {Multiply}
  end if
end for
$(a, b) \leftarrow$ Gen2Std$((x, y), f)$
return(a, b)

Die Korrektheit dieser Exponentiationsroutine ist sofort ersichtlich.

PROPOSITION 2.2.8. *Der Algorithmus GenExp benötigt* $O(\log(n)\log(f)^2 + \log(|\Delta| f^2)^2)$ *Bitoperationen.*

BEWEIS. Die Laufzeit ergibt sich aus Proposition 2.2.6 und Proposition 2.2.7. □

2.2.2. Weitere Effizienzsteigerung durch den Chinesischen Restsatz in $(\mathcal{O}_\Delta/f\mathcal{O}_\Delta)^*$

Im letzten Abschnitt (siehe Theorem 2.2.5) zeigten wir, dass sich die Arithmetik in der Idealklassengruppe $\mathrm{Ker}(\phi_{Cl}^{-1})$ auf die effizientere Arithmetik in $(\mathcal{O}_\Delta/f\mathcal{O}_\Delta)^*$ zurückführen lässt. In diesem Abschnitt wollen wir zeigen, dass die Verwendung des Chinesischen Restsatzes für $(\mathcal{O}_\Delta/f\mathcal{O}_\Delta)^*$, verglichen mit der naiven Arithmetik (Mult-mod-f als Gruppenoperation), eine weitere Effizienzsteigerung ermöglicht. Wie in [**HM00b**] erläutert, lässt sich nämlich die Arithmetik in $(\mathcal{O}_\Delta/p\mathcal{O}_\Delta)^*$, p prim mit $\left(\frac{\Delta}{p}\right) = 1$, wiederum auf die noch etwas effizientere Arithmetik in $\mathbb{F}_p^*$ reduzieren. Somit kann eine Exponentiation in $\mathrm{Ker}(\phi_{Cl}^{-1})$ im Wesentlichen durch zwei Exponentiationen in $\mathbb{F}_p^*$ erreicht werden.

Sei $f = \prod_{i=1}^{k} p_i^{m_i}$ die Primzahlzerlegung des Führers und $f\mathcal{O}_\Delta = \prod_{i=1}^{k} p_i^{m_i}\mathcal{O}_\Delta = \prod_{i=1}^{l} \mathfrak{p}_i^{n_i}$ die Primidealzerlegung des Hauptideals $f\mathcal{O}_\Delta$. Dann besagt der Chinesische Restsatz (siehe z.B. [**Lan91**, §4, Seite 11]), dass $\mathcal{O}_\Delta/f\mathcal{O}_\Delta$ folgende Zerlegung aufweist:

$$\mathcal{O}_\Delta\Big/_{f\mathcal{O}_\Delta} \simeq \mathcal{O}_\Delta\Big/_{p_1^{m_1}\mathcal{O}_\Delta} \otimes \cdots \otimes \mathcal{O}_\Delta\Big/_{p_k^{m_k}\mathcal{O}_\Delta} \tag{2.2.3}$$

$$\simeq \mathcal{O}_\Delta\Big/_{\mathfrak{p}_1^{n_1}} \otimes \cdots \otimes \mathcal{O}_\Delta\Big/_{\mathfrak{p}_l^{n_l}}, \tag{2.2.4}$$

wobei $l \geq k$ und Gleichheit genau dann vorliegt, falls alle Primzahlen p_i träge sind.

Nach (2.2.3) kann man sich also bei der Untersuchung von $(\mathcal{O}_\Delta/f\mathcal{O}_\Delta)^*$ auf Führer beschränken, die die Potenz einer Primzahl p sind. Im Hinblick auf die spätere kryptographische Anwendung in Abschnitt 3.3, werden wir im Folgenden nur den Fall betrachten, in dem der Führer eine Primzahl p ist.

LEMMA 2.2.9. *Sei $\mathcal{O}_\Delta$ die Maximalordnung und p prim. Dann gibt es einen Isomorphismus*

$$(\mathcal{O}_\Delta/p\mathcal{O}_\Delta) \simeq \mathbb{F}_p[X]\Big/_{(f(X))},$$

wobei $(f(X))$ das von $f(X) \in \mathbb{F}_p[X]$ erzeugte Ideal und

$$f(X) = \begin{cases} X^2 - \frac{\Delta}{4}, & \text{falls} \quad \Delta \equiv 0 \pmod 4, \\ X^2 - X + \frac{1-\Delta}{4}, & \text{falls} \quad \Delta \equiv 1 \pmod 4, \end{cases} \tag{2.2.5}$$

ist.

BEWEIS. Sei $\rho \in \overline{\mathbb{F}_p}$ eine Nullstelle von $f(X) \in \mathbb{F}_p[X]$, wobei $f(X)$ wie in (2.2.5). Dann besitzt ein Element $\alpha \in \mathbb{F}_p[X]/(f(X))$ eine Darstellung $\alpha = x + y\rho$, wobei $x, y \in \mathbb{F}_p$. Auf der anderen Seite besitzt ein Element $\beta \in \mathcal{O}_\Delta/p\mathcal{O}_\Delta$ eine Darstellung $\beta = \bar{x} + \bar{y}\omega$, wobei ω wie in (1.2.1) und $\bar{x}, \bar{y} \in \mathbb{F}_p$. Setzen wir $x \equiv \bar{x} \pmod p$ und $y \equiv \bar{y} \pmod p$, so erhalten wir sofort die gewünschte bijektive Korrespondenz.

Nun muss noch gezeigt werden, dass es sich bei der Abbildung $\psi_{\mathbb{F}_p[X]} : (\mathcal{O}_\Delta/p\mathcal{O}_\Delta) \to \mathbb{F}_p[X]/\,(f(X))$ mit $x + y\omega \mapsto x + y\rho$ und der inversen Abbildung $\psi_{\mathbb{F}_p[X]}^{-1}$ um Homomorphismen handelt.

Sei $\beta_i = x_i + y_i\omega \in \mathcal{O}_\Delta/p\mathcal{O}_\Delta$ für $i \in \{1,2\}$. Dann ist

$$\begin{aligned}\psi_{\mathbb{F}_p[X]}(\beta_1\beta_2) &= \psi_{\mathbb{F}_p[X]}((x_1+y_1\omega)(x_2+y_2\omega))\\ &= \psi_{\mathbb{F}_p[X]}\left(x_1x_2+(x_1y_2+y_1x_2)\omega+y_1y_2\omega^2\right)\\ &= x_1x_2+(x_1y_2+y_1x_2)\rho+y_1y_2\rho^2\\ &= (x_1+y_1\rho)(x_2+y_2\rho)\\ &= \psi_{\mathbb{F}_p[X]}(\beta_1)\psi_{\mathbb{F}_p[X]}(\beta_2),\end{aligned}$$

wobei $\omega = \frac{\sqrt{\Delta}}{2}$ und $\omega^2 = \frac{\Delta}{4} = \rho^2$, falls $\Delta \equiv 0 \pmod 4$, bzw. $\omega = \frac{1+\sqrt{\Delta}}{2}$ und $\omega^2 = \frac{\Delta+1+2\sqrt{\Delta}}{4} = \rho^2$, falls $\Delta \equiv 1 \pmod 4$.

Dass die inverse Abbildung $\psi^{-1}_{\mathbb{F}_p[X]}$ auch ein Homomorphismus ist, kann analog gezeigt werden, indem man ω durch ρ ersetzt.

□

Durch dieses Resultat lässt sich die Arithmetik in $(\mathcal{O}_\Delta/p\mathcal{O}_\Delta)^*$ auf die Arithmetik in endlichen Körpern zurückführen, wo bereits vielerorts effiziente Implementierungen vorhanden sind.

SATZ 2.2.10. *Es existiert ein Isomorphismus*

$$\psi_{\mathbb{F}} : (\mathcal{O}_\Delta/p\mathcal{O}_\Delta)^* \xrightarrow{\sim} \begin{cases} \mathbb{F}_{p^2}^*, & \text{falls } \left(\frac{\Delta}{p}\right) = -1,\\ \mathbb{F}_p^* \otimes \mathbb{F}_p^*, & \text{falls } \left(\frac{\Delta}{p}\right) = 1,\\ (\mathbb{Z}/p^2\mathbb{Z})^*, & \text{falls } \left(\frac{\Delta}{p}\right) = 0.\end{cases}$$

Falls $(\Delta/p) = 1$ und die beiden Nullstellen $\rho, \bar\rho \in \overline{\mathbb{F}_p}$ von $f(X) \in \mathbb{F}_p[X]$ wie in (2.2.5) gegeben sind, kann der Isomorphismus $\psi_{\mathbb{F}}$, sowie sein Inverses $\psi_{\mathbb{F}}^{-1}$, mit $O((\log p)^2)$ Bitoperationen berechnet werden.

BEWEIS. Aus Lemma 2.2.9 wissen wir, dass die Abbildung $\mathcal{O}_\Delta/p\mathcal{O}_\Delta \xrightarrow{\sim} \mathbb{F}_p[X]/(f(X))$, $x + y\omega \mapsto x + y\rho$, wobei $f(X) \in \mathbb{F}_p[X]$, wie in (2.2.5), ein Isomorphismus ist.

Wir unterscheiden die folgenden drei Fälle:

1. $\left(\frac{\Delta}{p}\right) = -1$:
   In diesem Fall lässt sich zeigen, dass das Polynom $f(X)$ in $\mathbb{F}_p[X]$ irreduzibel ist und wir deshalb sofort den gewünschten Isomorphismus
   $$\mathcal{O}_\Delta\Big/p\mathcal{O}_\Delta \cong \mathbb{F}_p[X]\Big/(f(X)) \cong \mathbb{F}_{p^2}$$
   erhalten. Die Behauptung folgt, wenn wir jeweils die multiplikativen Gruppen der Ringe betrachten.

2. $\left(\frac{\Delta}{p}\right) = 1$:
   In diesem Fall ist das Polynomial $f(X)$ nicht irreduzibel, sondern erlaubt die Zerlegung $f(X) = (X-\rho)(X-\bar\rho) \in \mathbb{F}_p[X]$, wobei $\rho, \bar\rho \in \mathbb{F}_p$ die beiden Nullstellen von $f(X)$ sind; da $(\Delta/p) = 1$ ist $\rho \neq \bar\rho$.

   Falls $\Delta \equiv 0 \pmod 4$ und $D = \Delta/4$, ist also $\rho \in \mathbb{F}_p$, so dass $\rho^2 \equiv D \pmod p$ und $\bar\rho \equiv -\rho \pmod p$. Im anderen Fall ist $\Delta \equiv 1 \pmod 4$ und $\rho = (1+b)/2$,

wobei $b^2 \equiv \Delta \pmod p$ und $\bar\rho = (1-b)/2 \in \mathbb{F}_p$. Deshalb erhalten wir den Isomorphismus

$$(\mathcal{O}_\Delta/p\mathcal{O}_\Delta)^* \cong \Big(\mathbb{F}_p[X]\Big/\big(f(X)\big)\Big)^* \cong \Big(\mathbb{F}_p[X]\Big/(X-\rho)\Big)^* \otimes \Big(\mathbb{F}_p[X]\Big/(X-\bar\rho)\Big)^* \cong \mathbb{F}_p^* \otimes \mathbb{F}_p^*.$$

Sei $a + b\omega \in (\mathcal{O}_\Delta/p\mathcal{O}_\Delta)^*$, dann ist dieser Isomorphismus $\psi_{\mathbb{F}} : (\mathcal{O}_\Delta/p\mathcal{O}_\Delta)^* \to \mathbb{F}_p^* \otimes \mathbb{F}_p^*$ konkret durch $x_1 = \psi_1(\alpha) = a + b\rho \in \mathbb{F}_p^*$ und $x_2 = \psi_2(\alpha) = a + b\bar\rho \in \mathbb{F}_p^*$ gegeben.

Die inverse Abbildung $\psi_{\mathbb{F}}^{-1}$ wird durch Lösen eines kleinen linearen Gleichungssystems berechnet. Man erhält also $a, b \in \mathbb{F}_p^*$ durch Berechnung von $b = \frac{x_2 - x_1}{\bar\rho - \rho}$ und $a = x_1 - b\rho$. Sind $\rho, \bar\rho$, wie angenommen, bereits bekannt, so ist leicht einzusehen, dass die Berechnung von $\psi_{\mathbb{F}}$ und $\psi_{\mathbb{F}}^{-1}$ mit $O(\log(p)^2)$ Bitoperationen durchgeführt werden kann.

3. $\left(\frac{\Delta}{p}\right) = 0$:
Auch in diesem Fall ist das Polynomial $f(X)$ nicht irreduzibel, sondern hat die Zerlegung $f(X) = (X-\rho)^2 \in \mathbb{F}_p[X]$, wobei $\rho \in \mathbb{F}_p$, und man erhält den Isomorphismus

$$\mathcal{O}_\Delta\Big/p\mathcal{O}_\Delta \cong \mathbb{F}_p[X]\Big/\big(f(X)\big) \cong \mathbb{F}_p[X]\Big/\big((X-\rho)^2\big) \cong \mathbb{Z}\Big/p^2\mathbb{Z}.$$

Die Behauptung folgt, wenn wir jeweils die multiplikativen Gruppen der Ringe betrachten.

□

Da wir uns später auf den für die Praxis relevanten Fall $(\Delta/p) = 1$ fokussieren wollen, stellen wir noch eine einfache Routine bereit, die in diesem Fall die Nullstellen von $f(X)$ berechnet. Hierfür stützen wir uns auf eine Routine SqrtFp, die bei Eingabe von $a \in \mathbb{F}_p^*$ eine Quadratwurzel $s \in \mathbb{F}_p^*$, so dass $s^2 \equiv a \pmod p$, bzw. $s \equiv 0$, zurückliefert, falls keine Quadratwurzel von $a \in \mathbb{F}_p^*$ existiert. Algorithmen zur Implementierung von SqrtFp finden sich beispielsweise in [**BS96**, Section 7]. Im folgenden Algorithmus ist die Existenz der Quadratwurzeln übrigens durch $(\Delta/p) = 1$ gesichert.

Algorithmus 9 GetRoots

Eingabe: Eine Diskriminante Δ und eine Primzahl p so dass $\left(\frac{\Delta}{p}\right) = 1$.
Ausgabe: Die Nullstellen $\rho, \bar\rho \in \mathbb{F}_p^*$ von $f(X)$ wie in (2.2.5).

```
if Δ ≡ 0 (mod 4) then
  D ← Δ/4
  ρ ← SqrtFp(D, p)
  ρ̄ ← −ρ (mod p)
else
  b ← SqrtFp(Δ, p)
  ρ ← (1 + b)/2 (mod p)
  ρ̄ ← (1 − b)/2 (mod p)
end if
return(ρ, ρ̄)
```

Durch Komposition des Isomorphismus'

$$\psi_{\mathbb{F}}^{-1} : \left\{ \begin{array}{lll} \mathbb{F}_{p^2}^*, & \text{falls} & \left(\frac{\Delta}{p}\right) = -1 \\ \mathbb{F}_p^* \otimes \mathbb{F}_p^*, & \text{falls} & \left(\frac{\Delta}{p}\right) = 1 \\ (\mathbb{Z}/p^2\mathbb{Z})^*, & \text{falls} & \left(\frac{\Delta}{p}\right) = 0 \end{array} \right\} \xrightarrow{\sim} (\mathcal{O}_\Delta/p\mathcal{O}_\Delta)^*$$

aus Satz 2.2.10 mit dem surjektiven Homomorphismus

$$\psi_{\mathrm{Ker}(\phi_{Cl}^{-1})} : (\mathcal{O}_\Delta/p\mathcal{O}_\Delta)^* \longrightarrow \mathrm{Ker}(\phi_{Cl}^{-1})$$

aus Satz 2.2.5, erhalten wir den surjektiven Homomorphismus

$$\psi_{p^2} = \psi_{\mathbb{F}}^{-1} \circ \psi_{\mathrm{Ker}(\phi_{Cl}^{-1})} : \left\{ \begin{array}{lll} \mathbb{F}_{p^2}^*, & \text{falls} & \left(\frac{\Delta}{p}\right) = -1 \\ \mathbb{F}_p^* \otimes \mathbb{F}_p^*, & \text{falls} & \left(\frac{\Delta}{p}\right) = 1 \\ (\mathbb{Z}/p^2\mathbb{Z})^*, & \text{falls} & \left(\frac{\Delta}{p}\right) = 0 \end{array} \right\} \longrightarrow \mathrm{Ker}(\phi_{Cl}^{-1}). \tag{2.2.6}$$

Für die Konstruktion der folgenden Exponentiationsroutine, deren Korrektheit sofort aus (2.2.6) folgt, beschränken wir uns nun auf den für die Praxis interessantesten Fall $(\Delta/p) = 1$ und setzen die Existenz einer effizienten Exponentiationsroutine in $\mathbb{F}_p^*$ voraus.

Algorithmus 10 GenExpCRT

Eingabe: Eine Erzeugerdarstellung (x, y) der Klasse $\mathfrak{a} \sim \varphi(x + y\omega) \in \mathrm{Ker}(\phi_{Cl}^{-1})$, wobei $x + y\omega \in (\mathcal{O}_\Delta/p\mathcal{O}_\Delta)^*$, den Führer p, wobei $\left(\frac{\Delta}{p}\right) = 1$, die Nullstellen $\rho, \bar{\rho} \in \mathbb{F}_p^*$ von $f(X)$ wie in (2.2.5) und den Exponenten $n \in \mathbb{Z}_{>0}$.

Ausgabe: Die Standarddarstellung (a, b) des reduzierten Vertreters der Klasse von $\mathfrak{a}^n = a\mathbb{Z} + \frac{b+\sqrt{\Delta p^2}}{2}\mathbb{Z} \in \mathrm{Ker}(\phi_{Cl}^{-1})$.

{Exponentiation in $\mathbb{F}_p^* \otimes \mathbb{F}_p^*$}
$x_1 \leftarrow (x + \rho y)^n \pmod p$
$x_2 \leftarrow (x + \bar{\rho} y)^n \pmod p$
{Berechne $\psi_{p^2}(x_1, x_2)$}
$r \leftarrow (\bar{\rho} - \rho)^{-1} \pmod p$
$y \leftarrow (x_2 - x_1)r \pmod p$
$x \leftarrow x_1 - y\rho \pmod p$
$(a, b) \leftarrow$ Gen2Std$((x, y), p)$
return(a, b)

Es sei angemerkt, dass die Berechnung von $r = (\bar{\rho} - \rho)^{-1} \pmod p$ in einer Vorausberechnungsphase gemacht werden kann, da sie unabhängig vom zu exponentierenden Ideal ist. Weiterhin ist leicht ersichtlich, dass – wie bei der naiven Methode GenExp – auch hier die asymptotische Laufzeit gleich $O(\log(n)\log(p)^2 + \log(|\Delta|p^2)^2)$ ist. Ist außerdem $p > \sqrt{|\Delta|}, n$, so hat der Algorithmus eine Laufzeit von $O(\log(p)^3)$ Bitoperationen. Allerdings dokumentieren die Laufzeitmessungen in Abschnitt 2.2.4, dass sich die impliziten Konstanten signifikant unterscheiden.

2.2.3. Der Isomorphismus $\mathrm{Ker}(\phi_{Cl}^{-1}) \tilde{=} \mathbb{F}_p^*$ für $(\Delta/p) = 1$

In diesem Abschnitt wollen wir zeigen, dass für eine Exponentiation in $\mathrm{Ker}(\phi_{Cl}^{-1})$, falls $(\Delta/p) = 1$, lediglich *eine* modulare Exponentiation ausreichend ist.

Um diese – für die praktische Anwendung des NICE-Schnorr-Verfahrens sehr wichtige – Behauptung zu begründen, wollen wir den folgenden Satz beweisen:

SATZ 2.2.11. *Sei $\mathcal{O}_\Delta$ eine imaginärquadratische Maximalordnung der Diskriminante $\Delta < -4$, p prim, $(\Delta/p) = 1$, $\phi_{Cl}^{-1} : Cl(\Delta p^2) \to Cl(\Delta)$ wie in Proposition 1.3.5 und die beiden Nullstellen $\rho, \bar{\rho} \in \mathbb{F}_p^*$ des Polynomes $f(X)$, wie in (2.2.5) gegeben. Dann kann der Isomorphismus*

$$\psi : \mathbb{F}_p^* \xrightarrow{\sim} \mathrm{Ker}(\phi_{Cl}^{-1}),$$

sowie sein Inverses ψ^{-1}, in $O(\log(p)^2)$ Bitoperationen berechnet werden.

BEWEIS. Sei $\left(\frac{\Delta}{p}\right) = 1$. Dann ist – nach Satz 2.2.10 – $(\mathcal{O}_\Delta/p\mathcal{O}_\Delta)^* \tilde{=} \mathbb{F}_p^* \otimes \mathbb{F}_p^*$ und der Isomorphismus $\mathrm{Ker}(\phi_{Cl}^{-1}) \tilde{=} \mathbb{F}_p^*$ folgt aus der exakten Sequenz [**Cox89**, (7.27), Seite 147]

$$1 \longrightarrow \mathbb{F}_p^* \longrightarrow (\mathcal{O}_\Delta/p\mathcal{O}_\Delta)^* \tilde{=} \mathbb{F}_p^* \otimes \mathbb{F}_p^* \longrightarrow \mathrm{Ker}(\phi_{Cl}^{-1}) \longrightarrow 1.$$

Es bleibt eine konstruktive Version dieses Isomorphismus anzugeben und die benötigte Laufzeit zu betrachten.

Sei (x, y) eine Erzeugerdarstellung der Idealklasse von $\mathfrak{a} \sim \varphi(\alpha) \in \mathrm{Ker}(\phi_{Cl}^{-1})$, wobei $\alpha = x + y\omega \in (\mathcal{O}_\Delta/p\mathcal{O}_\Delta)^*$, und $\rho, \bar{\rho}$ die Nullstellen von $f(X)$, wie in (2.2.5). Dann wird $\alpha = x + y\omega \in (\mathcal{O}_\Delta/p\mathcal{O}_\Delta)^*$ durch den – in Satz 2.2.10 angegebenen – Isomorphismus $\psi_{\mathbb{F}} : (\mathcal{O}_\Delta/p\mathcal{O}_\Delta)^* \to \mathbb{F}_p^* \otimes \mathbb{F}_p^*$ auf $(x_1, x_2) \in \mathbb{F}_p^* \otimes \mathbb{F}_p^*$ abgebildet, wobei $x_1 = x + y\rho$ und $x_2 = x + y\bar{\rho}$.

Sei $s \in \mathbb{F}_p^*$, so dass $s(x + y\bar{\rho}) \equiv 1 \pmod{p}$. Aus Korollar 1.3.9 (siehe auch Bemerkung 2.2.4 (2.)) folgt, dass $\varphi(\alpha) \sim \varphi(s \cdot \alpha)$ und deshalb (sx, sy) eine weitere Erzeugerdarstellung der Klasse von $\mathfrak{a} \sim \varphi(\alpha) \sim \varphi(s \cdot \alpha)$ ist. Die Abbildung $\psi_{\mathbb{F}}$ bildet nun $s \cdot \alpha$ auf das Paar $(s(x + y\rho), 1)$ ab und induziert so den gewünschten Isomorphismus $\psi^{-1} : \mathrm{Ker}(\phi_{Cl}^{-1}) \xrightarrow{\sim} \mathbb{F}_p^* \otimes \mathbf{1} \tilde{=} \mathbb{F}_p^*$,

$$\begin{aligned}
\mathfrak{a} &= \varphi(x + y\omega) \\
&\sim \varphi\left(\frac{x + y\omega}{x + \bar{\rho}y}\right) \\
&\mapsto \psi^{-1}\left(\varphi\left(\frac{x + y\omega}{x + \bar{\rho}y}\right)\right) \\
&= \left(\frac{x + \rho y}{x + \bar{\rho}y}, \frac{x + \bar{\rho}y}{x + \bar{\rho}y}\right) \\
&= \left(\frac{x + \rho y}{x + \bar{\rho}y}, 1\right) \\
&\simeq \frac{x + \rho y}{x + \bar{\rho}y}. \qquad (2.2.7)
\end{aligned}$$

Die inverse Abbildung $\psi : \mathbb{F}_p^* \xrightarrow{\sim} \mathrm{Ker}(\phi_{Cl}^{-1})$ ist, wie aus dem Beweis von Satz 2.2.10 und GenexpCRT (Algorithmus 10) ersichtlich, durch

$$\begin{aligned} x &\mapsto \psi(x) \\ &\simeq \psi_{p^2}(x,1) \\ &= \varphi\left(x - \frac{1-x}{\bar\rho-\rho}\rho + \frac{1-x}{\bar\rho-\rho}\omega\right) \\ &= \varphi\left(\frac{x(\bar\rho-\rho)-(1-x)\rho}{\bar\rho-\rho} + \frac{1-x}{\bar\rho-\rho}\omega\right) \\ &= \varphi\left(\frac{x\bar\rho - x\rho - \rho + x\rho}{\bar\rho-\rho} + \frac{1-x}{\bar\rho-\rho}\omega\right) \\ &= \varphi\left(\frac{x\bar\rho-\rho}{\bar\rho-\rho} + \frac{1-x}{\bar\rho-\rho}\omega\right) \\ &\sim \varphi\left(x\bar\rho - \rho + (1-x)\omega\right) \end{aligned} \tag{2.2.8}$$

gegeben.

Sind, wie angenommen, die beiden – wegen $(\Delta/p) = 1$ voneinander verschiedenen – Nullstellen $\rho, \bar\rho \in \mathbb{F}_p^*$ von $f(X)$, wie in (2.2.5), bekannt, so kann der Isomorphismus ψ und sein Inverses in $O(\log(p)^2)$ Bitoperationen berechnet werden. □

Der folgende Algorithmus berechnet $\psi^{-1} : \mathrm{Ker}(\phi_{Cl}^{-1}) \xrightarrow{\sim} \mathbb{F}_p^*$, wie in (2.2.7) beschrieben.

Algorithmus 11 Ker2Fp

Eingabe: Eine Erzeugerdarstellung (x, y) der Klasse $\mathfrak{g} \sim \varphi(x + y\omega) \in \mathrm{Ker}(\phi_{Cl}^{-1})$, wobei $x + y\omega \in (\mathcal{O}_\Delta/p\mathcal{O}_\Delta)^*$, den Führer p, wobei $\left(\frac{\Delta}{p}\right) = 1$ und $\Delta < -4$, sowie die Nullstellen $\rho, \bar\rho \in \mathbb{F}_p^*$ von $f(X)$ wie in (2.2.5).

Ausgabe: Das Element $g = \psi^{-1}(\mathfrak{g}) \in \mathbb{F}_p^*$.

$g \leftarrow (x + \rho y)(x + \bar\rho y)^{-1} \pmod p$
return(g)

Die umgekehrte Abbildung verwendet (2.2.8), um $\psi : \mathbb{F}_p^* \xrightarrow{\sim} \mathrm{Ker}(\phi_{Cl}^{-1})$ zu berechnen.

Algorithmus 12 Fp2Ker

Eingabe: Ein Element $g \in \mathbb{F}_p^*$, den Führer p, wobei $\left(\frac{\Delta}{p}\right) = 1$ und die Nullstellen $\rho, \bar\rho \in \mathbb{F}_p^*$ von $f(X)$ wie in (2.2.5).

Ausgabe: Eine Erzeugerdarstellung (x, y) der Klasse $\mathfrak{g} \sim \varphi(x + y\omega) \in \mathrm{Ker}(\phi_{Cl}^{-1})$, wobei $x + y\omega \in (\mathcal{O}_\Delta/p\mathcal{O}_\Delta)^*$.

$x \leftarrow g\bar\rho - \rho \pmod p$
$y \leftarrow 1 - g \pmod p$
return(x, y)

Mit diesen beiden Routinen für die Berechnung von ψ und ψ^{-1} ist es nun problemlos möglich, eine noch effizientere Exponentiationsroutine für Elemente in $\mathrm{Ker}(\phi_{Cl}^{-1})$ anzugeben.

Algorithmus 13 GenExpIso

Eingabe: Eine Erzeugerdarstellung (x, y) der Klasse $\mathfrak{g} \sim \varphi(x + y\omega) \in \mathrm{Ker}(\phi_{Cl}^{-1})$, wobei $x + y\omega \in (\mathcal{O}_\Delta/p\mathcal{O}_\Delta)^*$, den Führer p, wobei $\left(\frac{\Delta}{p}\right) = 1$, die Nullstellen $\rho, \bar{\rho} \in \mathbb{F}_p^*$ von $f(X)$ wie in (2.2.5) und den Exponenten $n \in \mathbb{Z}_{>0}$.

Ausgabe: Die Standarddarstellung (a, b) des reduzierten Vertreters der Klasse von $\mathfrak{g}^n = a\mathbb{Z} + \frac{b+\sqrt{\Delta p^2}}{2}\mathbb{Z} \in \mathrm{Ker}(\phi_{Cl}^{-1})$.

$g \leftarrow$Ker2Fp$((x, y), p, \rho, \bar{\rho})$
$g \leftarrow g^n \pmod p$
$(x, y) \leftarrow$Fp2Ker$(g, p, \rho, \bar{\rho})$
$(a, b) \leftarrow$ Gen2Std$((x, y), p)$
return(a, b)

Auch hier ist die asymptotische Laufzeit – wie bei den beiden vorher vorgestellten Exponentiationsroutinen GenExp und GenExpCRT auch – gleich $O(\log(n)\log(p)^2 + \log(|\Delta|p^2)^2)$, bzw. $O(\log(p)^3)$ Bitoperationen, falls $p > \sqrt{|\Delta|}, n$.

Allerdings werden wir auch hier sehen, dass sich die praktischen Laufzeiten signifikant unterscheiden.

2.2.4. Laufzeiten verschiedener Arithmetiken für $\mathrm{Ker}(\phi_{Cl}^{-1})$

In diesem Abschnitt stellen wir die praktischen Aspekte der oben entwickelten Arithmetiken für $\mathrm{Ker}(\phi_{Cl}^{-1})$ gegenüber, indem wir die Laufzeiten einer ersten Implementierung präsentieren.

Die Laufzeiten sind in Mikrosekunden angegeben. Es wurde ein Pentium 133 MHz und die LiDIA - Bibliothek [**LiD**] verwendet. Außerdem sei angemerkt, dass Δp^2 mit $|\Delta| \approx p$ gewählt wurde, die jeweiligen Exponentiationen mit einer einfachen Square & Multiply Variante und zufällig gewählten 160 Bit Exponenten realisiert wurden und keine der Implementierungen ausgiebig optimiert ist. Das ist kein Problem, da wir eher am Vergleich der Verfahren, als an den absoluten Zeiten interessiert sind.

Arithmetik	Ideal	GenExp	GenExpCRT	GenExpISO	modular
Bitlänge von	Δp^2	Δp^2	Δp^2	Δp^2	p
600	3182	159	83	42	188
800	4978	234	123	60	302
1000	7349	340	183	93	447
1200	9984	465	249	123	644
1600	15751	748	409	206	1063
2000	22868	1018	563	280	1454

TABELLE 1. Laufzeiten für die Exponentiation in $\mathrm{Ker}(\phi_{Cl}^{-1})$

Die Laufzeiten in Tabelle 1 dokumentieren die eindrucksvolle Verbesserung. Man sieht, dass die Exponentiation mit GenExp etwa zwanzig mal so schnell ist, wie eine Exponentiation mit gewöhnlicher Ideal-Arithmetik. Ist $(\Delta/p) = 1$, so kann durch

den Einsatz von GenExpCRT (Algorithmus 10), wie in [**HM00b**] vorgeschlagen, bzw. GenExpISO (Algorithmus 13), wie in [**Hüh01**] vorgeschlagen, die Exponentiation nochmal um den Faktor zwei, bzw. vier, beschleunigt werden.

Unterstellt man, dass die Berechnung diskreter Logarithmen in einer entsprechend gewählten Untergruppe von $\mathrm{Ker}(\phi_{Cl}^{-1})$ genauso schwierig ist, wie die Berechnung diskreter Logarithmen in einer Untergruppe von $(\mathbb{Z}/n\mathbb{Z})^*$, sofern $|\Delta|p^2 \approx n$, dann ist die Signaturerzeugung beim in Abschnitt 3.3.1 ausführlich erläuterten NICE-Schnorr-Verfahren bei vermutlich gleicher Sicherheit schließlich mehr als vier mal so schnell, wie im Originalverfahren [**Sch90**] mit $n = p'$ prim oder – wie in Abschnitt 3.3.1.3 erläutert – immer noch etwa doppelt so schnell wie bei einem Schnorr-Analog in $(\mathbb{Z}/dp^2\mathbb{Z})^*$.

2.3. Reduktion von Problemen in $Cl(\Delta f^2)$ auf Probleme in $Cl(\Delta)$ und $\mathrm{Ker}(\phi_{Cl}^{-1})$

In diesem Abschnitt wollen wir zeigen, wie die Kenntnis des Führers f zur Reduktion schwieriger Probleme in der Klassengruppe $Cl(\Delta f^2)$ auf analoge Probleme in $Cl(\Delta)$ und $\mathrm{Ker}(\phi_{Cl}^{-1})$ genutzt werden kann. Insbesondere werden wir zeigen, wie die Berechnung Diskreter Logarithmen und Wurzeln in $Cl(\Delta f^2)$ auf entsprechende Probleme in $Cl(\Delta)$ und $\mathrm{Ker}(\phi_{Cl}^{-1})$ reduziert werden kann.

Betrachtet man den für die kryptographische Praxis interessantesten Fall eines primen Führers mit $(\Delta/p) = 1$, so können diese Probleme in $\mathrm{Ker}(\phi_{Cl}^{-1})$, mittels des in Abschnitt 2.2.3 vorgestellten Isomorphismus, in $\mathbb{F}_p^*$ gelöst werden. Die verbleibenden Probleme in $Cl(\Delta)$ und $\mathbb{F}_p^*$ lassen sich bei geschickter Parameterwahl sehr viel leichter lösen, als das ursprüngliche Problem in $Cl(\Delta p^2)$.

Dieser Umstand läßt sich zur Konstruktion und effizienten Implementierung kryptographischer Verfahren nutzen, wobei der Führer p geheimgehalten wird und als Falltürinformation dient.

Zum Einen kann die Reduktion des DL-Problemes – wie in [**HJW01**] vorgeschlagen und in Abschnitt 3.2.4 näher erläutert – zur Konstruktion eines praktikablen, nicht-interaktiven, identitätsbasierten Public-Key Kryptosystemes im Stil von [**MY91**] genutzt werden.

Auf der anderen Seite kann durch die Reduktion des Quadratwurzel-Problemes bei der Implementierung des – in [**HMT98**] vorgeschlagenen Rabin-Verfahrens in $Cl(\Delta p^2)$ auf den ineffizienten Gaußschen Algorithmus (siehe [**Gau01**, Art. 286, S. 328] und [**Mey97, Sch99, Sha71**]) verzichtet werden.

Dieser Abschnitt gliedert sich in zwei Teile. In Abschnitt 2.3.1 betrachten wir die Problemreduktionen für beliebige endliche, abelsche Gruppen. In Abschnitt 2.3.2 wenden wir die generischen Ergebnisse auf den hier benötigten Fall der Klassengruppe $Cl(\Delta p^2)$ an.

2.3.1. Generische Problemreduktionen

In diesem Abschnitt betrachten wir die Reduktion des DL- und Wurzel-Problemes für beliebige endliche, abelsche Gruppen.

Im Folgenden seien G, H endliche, abelsche Gruppen und $\Phi : G \to H$ ein Homomorphismus.

2.3.1.1. Reduktion des DL-Problemes

In diesem Abschnitt wollen wir zeigen, wie sich das DL-Problem in der Gruppe G auf analoge Probleme in H und $\mathrm{Ker}(\Phi)$ zurückführen lässt.

Sei G eine endliche, abelsche Gruppe und $\mathfrak{g}, \mathfrak{a} \in G$. Dann ist das *Problem des Diskreten Logarithmus* (DL-Problem) die Ermittlung des kleinsten positiven $x \in \mathbb{Z}$, so dass $\mathfrak{a} = \mathfrak{g}^x$, bzw. die Entscheidung dass kein solches x existiert.

Wir beweisen den folgenden Satz:

SATZ 2.3.1. *Seien G, H endliche, abelsche Gruppen und $\Phi : G \to H$ ein Homomorphismus. Dann kann das DL-Problem in G durch zwei Anwendungen von Φ, $O(\log(|H|))$ Gruppenoperationen in G und $O(\log(|G|)^2)$ Bitoperationen auf zwei DL-Berechnungen in H und eine DL-Berechnung in $\mathrm{Ker}(\Phi)$ reduziert werden.*

BEWEIS. Sei G, H, Φ wie oben und $\mathfrak{g}, \mathfrak{a} \in G$. Dann ist das DL-Problem die Bestimmung des kleinsten positiven $x \in \mathbb{Z}$, so dass $\mathfrak{a} = \mathfrak{g}^x$.

Um dieses Problem auf analoge Probleme in H und $\mathrm{Ker}(\Phi)$ zu reduzieren, geht man folgendermaßen vor:

Man berechnet $g = \Phi(\mathfrak{g})$ und $a = \Phi(\mathfrak{a})$ und den Diskreten Logarithmus in H als das kleinste positive $x' \in \mathbb{Z}$, so dass $g^{x'} = a$. Existiert kein solches x', so kann – da Φ ein Homomorphismus ist – auch x nicht existieren.

Sei

$$\mathfrak{a} = \mathfrak{g}^{x'}\mathfrak{h}. \tag{2.3.1}$$

Dann sieht man weiterhin, dass der Diskrete Logarithmus x genau dann existiert, wenn $\mathfrak{h} = \mathfrak{a}\mathfrak{g}^{-x'} \in \langle \mathfrak{g} \rangle \cap \mathrm{Ker}(\Phi)$.

Wir bestimmen einen Erzeuger der zyklischen Gruppe $\langle \mathfrak{g} \rangle \cap \mathrm{Ker}(\Phi)$ als die kleinste Potenz $\mathfrak{g}^u$ für die $\mathfrak{g}^u \in \mathrm{Ker}(\Phi)$. Da Φ ein Homomorphismus ist, ist u die Ordnung des Elementes $g \in H$. Diese lässt sich durch eine weitere DL-Berechnung in H bestimmen. Sei $u' \in \mathbb{Z}$ die kleinste positive Zahl, so dass $g^{u'} = g^{-1}$. Dann ist $u = u' + 1$ die gesuchte Ordnung des Elementes $g \in H$.

Schließlich berechnen wir den Diskreten Logarithmus v von $\mathfrak{h}$ zur Basis $\mathfrak{g}^u$ in der Gruppe $\mathrm{Ker}(\Phi)$, so dass

$$\mathfrak{h} = \mathfrak{g}^{uv}. \tag{2.3.2}$$

Aus (2.3.1) und (2.3.2) erhalten wir schließlich $x = uv + x'$. Da u, v, x' jeweils die kleinsten positiven Lösungen der jeweiligen DL-Probleme sind, ist auch x die kleinste positive Lösung für das ursprüngliche DL-Problem.

Um den Aufwand für diese Reduktion abzuschätzen, sind neben der zweimaligen Anwendung von Φ die Gruppenoperationen in G zur Berechnung von $\mathfrak{h} = \mathfrak{a}\mathfrak{g}^{-x'}$ und $\mathfrak{g}^u$ sowie die Berechnung von $x = uv + x'$ zu berücksichtigen.

Da $x', u = O(|H|)$, benötigt man für die Berechnung von $\mathfrak{h} = \mathfrak{a}\mathfrak{g}^{-x'}$ und $\mathfrak{g}^u$ $O(\log(|H|))$ Gruppenoperationen in G. Außerdem ist $x = O(|G|)$ und man benötigt $O(\log(|G|)^2)$ Bitoperationen zur Berechnung von $x = uv + x'$. Daraus folgt die Aussage. □

Wir nehmen an, dass folgende Algorithmen zur Verfügung stehen:

- ComputePhi($\mathfrak{g}$)
  erhält ein Element $\mathfrak{g} \in G$ als Eingabe und liefert das Element $g = \Phi(\mathfrak{g}) \in H$ zurück,

- DLPinH(g, a)
  erhält zwei Elemente $g, a \in H$ als Eingabe und liefert das kleinste positive $x \in \mathbb{Z}$, so dass $a = g^x$, bzw. $x = -1$ zurück, falls kein solches x existiert, und
- DLPinKer$(\mathfrak{g}, \mathfrak{a})$
  erhält zwei Elemente $\mathfrak{g}, \mathfrak{a} \in \mathrm{Ker}(\Phi)$ als Eingabe und liefert das kleinste positive $x \in \mathbb{Z}$, so dass $\mathfrak{a} = \mathfrak{g}^x$, bzw. $x = -1$ zurück, falls kein solches x existiert.

Dann kann das DL-Problem in der Gruppe G durch folgenden Algorithmus gelöst werden.

Algorithmus 14 ReduceDLP

Eingabe: Zwei Elemente $\mathfrak{g}, \mathfrak{a} \in G$.
Ausgabe: Das kleinste positive $x \in \mathbb{Z}$, so dass $\mathfrak{a} = \mathfrak{g}^x$, bzw. $x = -1$, falls kein solches x existiert.

```
{Berechne DL in H}
g ←ComputePhi(g)
a ←ComputePhi(a)
x' ←DLPinH(g, a)
if x' = −1 then
  return(−1)
end if
{Bestimme u, so dass ⟨g^u⟩ = ⟨g⟩ ∩ Ker(Φ)}
u' ←DLPinH(g, g^{−1})
u ← u' + 1
{Berechne DL in Ker(Φ)}
v ←DLPinKer(g^u, a g^{−x'})
if v = −1 then
  return(−1)
end if
{Setze Teilergebnisse zu DL in G zusammen}
x ← uv + x'
return(x)
```

2.3.1.2. Reduktion des Wurzel-Problemes

In diesem Abschnitt wollen wir zeigen, wie sich das e-te Wurzel-Problem in der Gruppe G auf analoge Probleme in H und $\mathrm{Ker}(\Phi)$ zurückführen lässt.

Sei G eine endliche, abelsche Gruppe, $\mathfrak{g} \in G$ und $e \in \mathbb{Z}, e \geq 2$. Dann ist das *($e$-te) Wurzel-Problem* die Ermittlung eines $\mathfrak{r} \in G$, so dass $\mathfrak{r}^e = \mathfrak{g}$, bzw. die Entscheidung dass kein solches $\mathfrak{r}$ existiert. Ist $e = 2$, so sprechen wir vom Quadratwurzel-Problem.

Die hier vorgestellte Reduktion ist insbesondere dann vorteilhaft, wenn die Berechnung e-ter Wurzeln nicht einfach durch Exponentiation mit d, so dass $ed \equiv 1 \pmod{|G|}$, durchgeführt werden kann, weil $|G|$ nicht bekannt oder nicht teilerfremd zu e ist.

Für die folgende Reduktion des Wurzel-Problemes gehen wir im Vergleich zur oben dargestellten Reduktion des DL-Problemes von einer geringfügig veränderten Situation aus. Wir fordern hier, dass der Homomorphismus $\Phi : G \to H$ surjektiv ist und sich effizient Urbilder von Φ berechnen lassen.

Wir beweisen den folgenden Satz:

SATZ 2.3.2. *Seien G, H endliche, abelsche Gruppen, $\Phi : G \to H$ ein surjektiver Homomorphismus und $e \in \mathbb{Z}, e \geq 2$. Dann kann das e-te Wurzel-Problem in G durch eine Anwendung von Φ, die Berechnung eines Urbildes von Φ und $O(\log(e))$ Gruppenoperationen in G jeweils auf die Berechnung einer e-ten Wurzel in H und* $\mathrm{Ker}(\Phi)$ *reduziert werden.*

BEWEIS. Sei G, H, Φ, e wie oben und $\mathfrak{g} \in G$. Dann ist das e-te Wurzel-Problem die Bestimmung eines $\mathfrak{r} \in G$, so dass $\mathfrak{r}^e = \mathfrak{g}$, oder die Entscheidung, dass kein solches $\mathfrak{r}$ existiert.

Um dieses Problem auf analoge Probleme in H und $\mathrm{Ker}(\Phi)$ zu reduzieren, geht man folgendermaßen vor:

Man berechnet $g = \Phi(\mathfrak{g})$ und eine e-te Wurzel $s \in H$, so dass $s^e = g$. Existiert kein solches s, so kann – da Φ ein Homomorphismus ist – auch $\mathfrak{r}$ nicht existieren.

Sei $\mathfrak{s}$, so dass $s = \Phi(\mathfrak{s})$ dann ist

$$\mathfrak{g} = \mathfrak{s}^e \mathfrak{h}, \tag{2.3.3}$$

wobei $\mathfrak{h} = \mathfrak{g}\mathfrak{s}^{-e} \in \mathrm{Ker}(\Phi)$. Aus (2.3.3) ist zu sehen, dass $\mathfrak{g}$ genau dann eine e-te Potenz ist, wenn $\mathfrak{h}$ eine ist.

Nun bestimmen wir schließlich ein $\mathfrak{t}$, so dass $\mathfrak{t}^e = \mathfrak{h}$. Somit folgt aus (2.3.3), dass $\mathfrak{g} = \mathfrak{s}^e\mathfrak{t}^e$ und $\mathfrak{r} = \mathfrak{s}\mathfrak{t}$ eine e-te Wurzel aus $\mathfrak{g}$ ist.

Der Aufwand für diese Reduktion ist sofort ersichtlich. □

Wir nehmen an, dass folgende Algorithmen zur Verfügung stehen:

- ComputePhi($\mathfrak{g}$)
  erhält ein Element $\mathfrak{g} \in G$ als Eingabe und liefert das Element $g = \Phi(\mathfrak{g}) \in H$ zurück,
- InvertPhi(g)
  erhält ein Element $g \in H$ als Eingabe und liefert ein Element $\mathfrak{g} \in G$, so dass $g = \Phi(\mathfrak{g})$, zurück,
- ROOTinH(g, e)
  erhält ein Element $g \in H$ und ein $e \in \mathbb{Z}$ ($e \geq 2$) als Eingabe und liefert ein $r \in H$, so dass $r^e = g$, bzw. $x = -1$ zurück, falls kein solches r existiert und
- ROOTinKer($\mathfrak{g}$)
  erhält ein Element $\mathfrak{g} \in \mathrm{Ker}(\Phi)$ und ein $e \in \mathbb{Z}$ ($e \geq 2$) als Eingabe und liefert ein $\mathfrak{r} \in \mathrm{Ker}(\Phi)$, so dass $\mathfrak{r}^e = \mathfrak{g}$, bzw. $x = -1$ zurück, falls kein solches $\mathfrak{r}$ existiert.

Dann kann die Berechnung e-ter Wurzeln in G mit folgendem Algorithmus durchgeführt werden.

Algorithmus 15 ReduceROOT

Eingabe: Ein Element $\mathfrak{g} \in G$ und $e \in \mathbb{Z}$ ($e \geq 2$).
Ausgabe: Ein Element $\mathfrak{r} \in G$, so dass $\mathfrak{r}^e = \mathfrak{g}$, bzw. -1, falls kein solches $\mathfrak{r}$ existiert.

$g \leftarrow$ComputePhi($\mathfrak{g}$)
$s \leftarrow$ROOTinH(g, e)
if $s = -1$ **then**
  return(-1)
end if
$\mathfrak{s} \leftarrow$InvertPhi(s)
$\mathfrak{h} \leftarrow \mathfrak{g}\mathfrak{s}^{-e}$
$\mathfrak{t} \leftarrow$ROOTinKer($\mathfrak{h}, e$)
if $\mathfrak{t} = -1$ **then**
  return(-1)
end if
$\mathfrak{r} \leftarrow \mathfrak{s}\mathfrak{t}$
return$(\mathfrak{s})$

2.3.2. Anwendung auf den Fall $\phi_{Cl}^{-1} : Cl(\Delta f^2) \to Cl(\Delta)$

In diesem Abschnitt wollen wir die im vorherigen Abschnitt für beliebige endliche, abelsche Gruppen dargestellten Reduktionen auf Klassengruppen imaginärquadratischer Nichtmaximalordnungen anwenden.

Hierzu werden wir zunächst, in Abschnitt 2.3.2.1, die abstrakten Objekte und Methoden aus Abschnitt 2.3.1 konkretisieren und danach, in Abschnitt 2.3.2.2, Laufzeitschranken für unseren Anwendungsfall entwickeln.

2.3.2.1. Instanziierung der abstrakten Objekte und Methoden

In diesem Abschnitt wollen wir den abstrakten Objekten aus Abschnitt 2.3.1 konkrete Strukturen zuweisen und näher auf die verwendeten Methoden eingehen.

In unserem Fall ist

- $G = Cl(\Delta f^2)$ die Klassengruppe einer imaginärquadratischen Nichtmaximalordnung und
- $H = Cl(\Delta)$ die Klassengruppe der zugehörigen Maximalordnung.

Im Folgenden sollen die in Abschnitt 2.3.1 verwendeten Algorithmen etwas näher betrachtet werden.

Zur Reduktion der Probleme von $Cl(\Delta f^2)$ auf $Cl(\Delta)$ und $\mathrm{Ker}(\phi_{Cl}^{-1})$ werden die folgenden Algorithmen benötigt:

- ComputePhi($\mathfrak{g}$)
  berechnet den surjektiven Homomorphismus $\phi_{Cl}^{-1} : Cl(\Delta f^2) \to Cl(\Delta)$ aus Proposition 1.3.5. Bei bekanntem Führer f kann dies mit GoToMaxCl (Algorithmus 4) durchgeführt werden.
- InvertPhi(g)
  berechnet ein Urbild unter ϕ_{Cl}^{-1}. Dies kann mit GoToNonMaxOrder (Algorithmus 2) durchgeführt werden.

Im Folgenden wird näher auf Algorithmen zur Lösung der verbleibenden Probleme in $Cl(\Delta)$ und $\mathrm{Ker}(\phi_{Cl}^{-1})$ eingegangen.

Algorithmen zur Berechnung Diskreter Logarithmen und Wurzeln in $\mathrm{Ker}(\phi_{Cl}^{-1})$

In diesem Abschnitt wollen wir Algorithmen zur Lösung des DL- und Wurzel-Problemes in $\mathrm{Ker}(\phi_{Cl}^{-1})$ näher betrachten. Hier wird zunächst skizziert, wie die in Abschnitt 2.2 entwickelten Ergebnisse zur Reduktion der Probleme von $\mathrm{Ker}(\phi_{Cl}^{-1})$ auf analoge Probleme in arithmetisch besser handhabbaren Strukturen, wie $(\mathcal{O}_\Delta/f\mathcal{O}_\Delta)^*$ bzw. $\mathbb{F}_p^*$, verwendet werden können. Für letzteren Fall existieren schließlich wohlbekannte Algorithmen zur Lösung des DL- und Wurzel-Problemes.

Reduktion der Probleme von $\mathrm{Ker}(\phi_{Cl}^{-1})$ auf $(\mathcal{O}_\Delta/f\mathcal{O}_\Delta)^*$ bzw. $\mathbb{F}_p^*$

Wie in Abschnitt 2.2 ausführlich erläutert, existieren verschiedene Ansätze die Arithmetik in $\mathrm{Ker}(\phi_{Cl}^{-1})$ effizient zu implementieren.

So kann die Arithmetik in $\mathrm{Ker}(\phi_{Cl}^{-1})$ im allgemeinen Fall durch den surjektiven Homomorphismus $\psi_{\mathrm{Ker}(\phi_{Cl}^{-1})} : (\mathcal{O}_\Delta/f\mathcal{O}_\Delta)^* \to \mathrm{Ker}(\phi_{Cl}^{-1})$ aus Satz 2.2.5 auf Berechnungen in $(\mathcal{O}_\Delta/f\mathcal{O}_\Delta)^*$ zurückgeführt werden.

Ist die Faktorisierung des Führers f bekannt, so kann – wie in Abschnitt 2.2.2 näher erläutert – der Chinesische Restsatz zur Reduktion der Arithmetik von $(\mathcal{O}_\Delta/f\mathcal{O}_\Delta)^*$ auf $(\mathcal{O}_\Delta/p^e\mathcal{O}_\Delta)^*$, wobei $p^e \mid f$, verwendet werden. Die Berechnung Diskreter Logarithmen und Wurzeln in $(\mathcal{O}_\Delta/p^e\mathcal{O}_\Delta)^*$ kann unter Verwendung der in [**PH78**] und [**MvOV97**, Algorithm 3.63, Seite 108], bzw. [**Mey97**], erläuterten Strategien auf analoge Probleme in $(\mathcal{O}_\Delta/p\mathcal{O}_\Delta)^*$ zurückgeführt werden.

Ist der Führer eine Primzahl p, so ist $(\mathcal{O}_\Delta/p\mathcal{O}_\Delta)^*$ nach Satz 2.2.10 isomorph zu $\mathbb{F}_p^* \otimes \mathbb{F}_p^*$, falls $\left(\frac{\Delta}{p}\right) = 1$, isomorph zu $(\mathbb{Z}/p^2\mathbb{Z})^*$ falls $\left(\frac{\Delta}{p}\right) = 0$ bzw. isomorph zu $\mathbb{F}_{p^2}^*$, falls $\left(\frac{\Delta}{p}\right) = -1$.

Beschränkt man sich auf den für die kryptographische Praxis relevanten Fall, dass $\Delta < -4$ und $\left(\frac{\Delta}{p}\right) = 1$, so ist nach Satz 2.2.11 $\mathrm{Ker}(\phi_{Cl}^{-1}) \stackrel{\sim}{=} \mathbb{F}_p^*$ und das DL- sowie das Wurzel-Problem in $\mathrm{Ker}(\phi_{Cl}^{-1})$ können auf analoge Probleme in $\mathbb{F}_p^*$ zurückgeführt werden.

Um den Isomorphismus $\psi^{-1} : \mathrm{Ker}(\phi_{Cl}^{-1}) \stackrel{\sim}{\longrightarrow} \mathbb{F}_p^*$ aus Satz 2.2.11 berechnen zu können, müssen die beiden Nullstellen $\rho, \bar{\rho} \in \mathbb{F}_p^*$ des Polynomes $f(X)$ aus (2.2.5) bekannt sein. Die Berechnung dieser Nullstellen kann mit der Routine GetRoots (Algorithmus 9) erfolgen, die ihrerseits eine Routine SqrtFp zur Berechnung von Quadratwurzeln in $\mathbb{F}_p^*$ verwendet.

Algorithmen für DL- und Wurzel-Problem in $\mathbb{F}_p^*$

Zur Lösung der verbleibenden DL- und Wurzelprobleme stehen wohlbekannte Algorithmen zur Verfügung auf die hier nicht weiter eingegangen werden soll.

Nähere Informationen zur Berechnung Diskreter Logarithmen in $\mathbb{F}_p^*$ finden sich z.B. in [**Web96, Web98, JL03**].

Für Algorithmen zur Berechnung von Wurzeln in $\mathbb{F}_p^*$ verweisen wir auf [**BS96**, Section 7].

Algorithmen zur Berechnung Diskreter Logarithmen und Wurzeln in $Cl(\Delta)$

Auch hier stehen wohlbekannte Algorithmen zur Berechnung Diskreter Logarithmen und (Quadrat-) Wurzeln zur Verfügung.

Nähere Informationen zur Berechnung Diskreter Logarithmen in $Cl(\Delta)$ finden sich in [**Jac99, Vol00**].

Die Berechnung e-ter Wurzeln in einer endlichen, abelschen Gruppe G kann, sofern die Gruppenordnung $|G|$ bekannt und teilerfremd zu e ist, bekanntlich durch eine Exponentiation mit d, so dass $ed \equiv 1 \pmod{|G|}$, durchgeführt werden.

Somit müsste zur Berechnung e-ter Wurzeln in $Cl(\Delta)$ die Klassenzahl $h(\Delta)$ bekannt und teilerfremd zu e sein. Ist dies nicht gegeben, so ist lediglich für den Spezialfall $e = 2$ ein (relativ ineffizienter) Algorithmus – der Gaußsche Algorithmus [**Gau01**, Art. 286, S. 328] – bekannt. Wählt man $\Delta = -p$, $p \equiv 3 \pmod 4$ prim und berechnet die Klassenzahl $h(\Delta)$, z.B. mit dem subexponentiellen Algorithmus [**Jac99**], dann kann auch die Berechnung von Quadratwurzeln durch eine einfache Exponentiation erreicht werden.

PROPOSITION 2.3.3. *Sei $q \equiv 3 \pmod 4$ prim, $\Delta = -q$ und $h(\Delta)$ bekannt. Dann ist die Quadratwurzel aus einer beliebigen Klasse $\mathfrak{a} \in Cl(\Delta)$ durch $\mathfrak{s} \sim \mathfrak{a}^{(h(\Delta)+1)/2}$ gegeben.*

BEWEIS. Sei $\Delta = -q$, q prim. Dann ist $h(\Delta)$, gemäß der z.B. in [**Zag81**, Kapitel 12] behandelten Geschlechtertheorie, ungerade. Somit ist jedes Element in $Cl(\Delta)$ ein Quadrat. Betrachtet man den Exponenten $(h(\Delta) + 1)/2 \cdot 2 \equiv 1 \mod h(\Delta)$, so folgt sofort die Aussage. □

2.3.2.2. Laufzeitschranken für Problemreduktionen

In diesem Abschnitt wollen wir die generischen Ergebnisse aus Abschnitt 2.3.1 verwenden, um Laufzeitschranken für die Problemreduktionen im vorliegenden Fall $\phi_{Cl}^{-1} : Cl(\Delta p^2) \to Cl(\Delta)$ zu entwickeln.

Laufzeitschranke für die Reduktion des DL-Problemes

Wir beginnen mit der Laufzeitschranke für die Reduktion des DL-Problemes.

PROPOSITION 2.3.4. *Sei $\Delta < -4$ und $p > \sqrt{|\Delta|}$ eine Primzahl mit $(\Delta/p) = 1$. Ist der Führer p gegeben, so kann das DL-Problem in $Cl(\Delta p^2)$ auf zwei DL-Berechnungen in $Cl(\Delta)$ und eine DL-Berechnung in $\mathbb{F}_p^*$ reduziert werden. Diese Reduktion hat eine erwartete Laufzeit von*

$$O\left(\log\left(|\Delta|p^2\right)^2 \log\left(\sqrt{|\Delta|}\log(\sqrt{|\Delta|})\right) + \log(p)^3\right)$$

Bitoperationen.

BEWEIS. Da der Führer p bekannt ist, kann ϕ_{Cl}^{-1} nach Proposition 2.1.7 in $O(\log(p)^2)$ Bitoperationen berechnet werden.

Nach Satz 2.3.1 kann das DL-Problem in $Cl(\Delta p^2)$ durch $O(\log(h(\Delta)))$ Gruppenoperationen in $Cl(\Delta p^2)$ und $O(\log(h(\Delta p^2))^2)$ Bitoperationen auf zwei DL-Berechnungen in $Cl(\Delta)$ und eine DL-Berechnung in $\mathrm{Ker}(\phi_{Cl}^{-1})$ reduziert werden.

Sei D eine beliebige Diskriminante. Dann ist nach [**Nar74**, Seite 389] $h(D) = O\left(\sqrt{|D|}\log(\sqrt{|D|})\right)$.

Außerdem werden für eine Gruppenoperation in $Cl(\Delta p^2)$ nach [**BB99**] $O(\log(|\Delta|p^2)^2)$ Bitoperationen benötigt. Deshalb werden für diese Reduktion

$$O\left(\log\left(|\Delta|p^2\right)^2 \log\left(\sqrt{|\Delta|}\log(\sqrt{|\Delta|})\right)\right)$$

Bitoperationen benötigt.

Um den Isomorphismus $\psi^{-1} : \mathrm{Ker}(\phi_{Cl}^{-1}) \xrightarrow{\sim} \mathbb{F}_p^*$ aus Satz 2.2.11 in $O(\log(p)^2)$ Bitoperationen berechnen zu können, müssen die beiden Nullstellen $\rho, \bar{\rho} \in \mathbb{F}_p^*$ des Polynomes $f(X)$ aus (2.2.5) bekannt sein. Die Berechnung dieser Nullstellen hat nach [**BS96**, Theorem 7.2.3] eine erwartete Laufzeit von $O(\log(p)^3)$ Bitoperationen. □

Damit hat die Reduktion insbesondere eine erwartete Laufzeit von $O\left(\log\left(|\Delta|p^2\right)^3\right)$ Bitoperationen, was in der Größenordnung einer Exponentiation in $Cl(\Delta p^2)$ liegt, und im Vergleich zur Laufzeit der verbleibenden DL-Probleme in $Cl(\Delta)$ [**Jac99**] und $\mathbb{F}_p^*$ [**Web96, Web98, JL03**] vernachlässigbar ist.

Laufzeitschranke für die Reduktion des Wurzel-Problemes

Die Laufzeit für die Reduktion des e-ten Wurzel-Problemes von $Cl(\Delta p^2)$ auf das Wurzel-Problem in $Cl(\Delta)$ lässt sich folgendermaßen abschätzen:

PROPOSITION 2.3.5. *Sei $\Delta < -4$ und $p > \sqrt{|\Delta|}$ eine Primzahl mit $(\Delta/p) = 1$. Ist der Führer p gegeben, so kann das e-te Wurzel-Problem in $Cl(\Delta p^2)$ auf die Berechnung einer e-ten Wurzel in $Cl(\Delta)$ und die Berechnung einer e-ten Wurzel in $\mathbb{F}_p^*$ reduziert werden, wobei eine Laufzeit von*

$$O\left(\log(e)\log\left(|\Delta|p^2\right)^2 + \log(p)^3\right)$$

Bitoperationen für diese Reduktion erwartet werden kann.

BEWEIS. Da der Führer p bekannt ist, kann ϕ_{Cl}^{-1} (mit GoToMaxCl, nach Proposition 2.1.7) und auch ein Urbild unter ϕ_{Cl}^{-1} (mit GoToNonMaxOrder, nach Proposition 2.1.6) in $O(\log(p)^2)$ Bitoperationen berechnet werden.

Nach Satz 2.3.2 kann das e-te Wurzel-Problem in $Cl(\Delta p^2)$ mit $O(\log(e))$ Gruppenoperationen in $Cl(\Delta p^2)$, die nach [**BB99**] jeweils $O(\log(|\Delta|p^2)^2)$ Bitoperationen benötigen, auf je eine Wurzelberechnung in $Cl(\Delta)$ und $\mathrm{Ker}(\phi_{Cl}^{-1})$ reduziert werden.

Die Wurzelberechnung in $\mathrm{Ker}(\phi_{Cl}^{-1})$ wird mittels des Isomorphismus $\psi^{-1} : \mathrm{Ker}(\phi_{Cl}^{-1}) \xrightarrow{\sim} \mathbb{F}_p^*$ auf eine Wurzelberechnung in $\mathbb{F}_p^*$ zurückgeführt.

Die Berechnung von ψ^{-1} benötigt nach Satz 2.2.11 $O(\log(p)^2)$ Bitoperationen, sofern die Nullstellen $\rho, \bar{\rho}$ zur Verfügung stehen. Die Berechnung der Nullstellen selbst geschieht durch die Berechnung einer Quadratwurzel in $\mathbb{F}_p^*$. Dies hat nach [**BS96**, Theorem 7.2.3] eine erwartete Laufzeit von $O(\log(p)^3)$ Bitoperationen. □

Für den allgemeinen Fall der Berechnung e-ter Wurzeln in $\mathbb{F}_p^*$ verweisen wir auf [**BS96**, Chapter 7] und stellen lediglich den für die Implementierung des Rabin-Verfahrens aus [**HMT98**] benötigten Fall $e = 2$ explizit heraus:

KOROLLAR 2.3.6. *Sei $\Delta < -4$ und $p > \sqrt{|\Delta|}$ eine Primzahl mit $(\Delta/p) = 1$. Ist der Führer p gegeben, so kann das Quadratwurzel-Problem in $Cl(\Delta p^2)$ auf die Berechnung einer Quadratwurzel in $Cl(\Delta)$ reduziert werden, wobei eine Laufzeit von*

$$O\left(\log\left(|\Delta|p^2\right)^2 + \log(p)^3\right)$$

Bitoperationen für diese Reduktion erwartet werden kann.

Wird eine Diskriminante $\Delta = -q$, $q \equiv 3 \pmod 4$ prim verwendet und ist die Klassenzahl $h(\Delta)$ bekannt, so kann die Berechnung der Quadratwurzel in $Cl(\Delta)$ durch eine Exponentiation in $Cl(\Delta)$ durchgeführt werden. Ist außerdem $p \equiv 3 \pmod 4$ so kann nach [**BS96**, Corollary 7.1.2] auch die Berechnung der Quadratwurzel in $\mathbb{F}_p^*$ durch eine Exponentiation in $\mathbb{F}_p^*$ mit dem Exponenten $(p+1)/4$ durchgeführt werden. Das folgende Korollar folgt sofort aus Proposition 2.3.3 und Korollar 2.3.6.

KOROLLAR 2.3.7. *Sei $\Delta = -q$, $q \equiv 3 \pmod 4$ eine Primzahl und $p > \sqrt{|\Delta|}$ eine Primzahl mit $p \equiv 3 \pmod 4$ und $\left(\frac{\Delta}{p}\right) = 1$. Ist der Führer p und die Klassenzahl $h(\Delta)$ gegeben, so kann das Quadratwurzel-Problem in $Cl(\Delta p^2)$ mit*

$$O\left(\log\left(|\Delta|\right)^2 \log\left(h(\Delta)\right) + \log\left(|\Delta| p^2\right)^2 + \log(p)^3\right)$$

Bitoperationen gelöst werden.

Da in diesem Fall die Quadratwurzelberechnung in $Cl(\Delta p^2)$ durch gewöhnliche Exponentiationen in $Cl(\Delta)$ und $\mathbb{F}_p^*$ erreicht werden kann, ist diese Konstellation für die kryptographische Praxis besonders attraktiv.

2.3.2.3. Beispielhafte Problemreduktionen in $Cl(\Delta p^2)$, $\Delta p^2 = -1019 \cdot 23^2$

In diesem Abschnitt wollen wir die oben eingeführten Problemreduktionen am Beispiel der Klassengruppe $Cl(\Delta p^2)$ mit $\Delta p^2 = -1019 \cdot 23^2 = -539051$ veranschaulichen. Beide Klassengruppen $Cl(\Delta p^2)$ und $Cl(\Delta)$ sind zyklisch, wobei $h(\Delta) = 13$ und $h(\Delta p^2) = h(\Delta)\left(p - \left(\frac{\Delta}{p}\right)\right) = 13 \cdot (23 - 1) = 286$.

Reduktion des DL-Problemes

In diesem Abschnitt wollen wir ReduceDLP (Algorithmus 14) zur Lösung eines DL-Problemes in $Cl(\Delta p^2)$ mit $\Delta p^2 = -1019 \cdot 23^2$ verwenden.

Sei $\mathfrak{g} = 15\mathbb{Z} + \frac{-7+\sqrt{-539051}}{2}\mathbb{Z} = (15, -7)$ und $\mathfrak{a} = 11\mathbb{Z} + \frac{9+\sqrt{-539051}}{2}\mathbb{Z} = (11, 9)$ gegeben. Dann ist das DL-Problem das Ermitteln des kleinsten positiven $x \in \mathbb{Z}$, so dass $\mathfrak{g}^x \sim \mathfrak{a}$, bzw. die Entscheidung, dass kein solches x existiert.

Hierfür verwenden wir ReduceDLP (Algorithmus 14) und berechnen zuerst $\mathfrak{G} = \phi_{Cl}^{-1}(\mathfrak{g})$ und $\mathfrak{A} = \phi_{Cl}^{-1}(\mathfrak{a})$, und lösen das DL-Problem $\mathfrak{G}^{x_1} \sim \mathfrak{A}$ in der Klassengruppe $Cl(\Delta)$ der Maximalordnung $\mathcal{O}_\Delta$. Wir erhalten $\mathfrak{G} = 15\mathbb{Z} + \frac{1+\sqrt{-1019}}{2}\mathbb{Z} = (15, 1)$, $\mathfrak{A} = 11\mathbb{Z} + \frac{9+\sqrt{-1019}}{2}\mathbb{Z} = (11, 9)$ und schließlich $x_1 = 9$. In der Praxis verwendet man das Analog [**Jac99, Jac00**] des Quadratischen Siebes für diese DL-Berechnung. Durch eine weitere DL-Berechnung erhalten wir $u = 13$ als die Ordnung von $\mathfrak{G}$ in $Cl(\Delta)$.

Nun wissen wir, dass der gesuchte Diskrete Logarithmus x die Form $x = u \cdot v + x_1 = 13v + 9$ hat; es muss schließlich noch v berechnet werden. Hierbei ist v der Diskrete Logarithmus von

$$\mathfrak{b} = \mathfrak{a}\mathfrak{g}^{-x'} = (311\mathbb{Z} + \frac{277 + 23\sqrt{-1019}}{2}\mathbb{Z}) = (311, 277)$$

zur Basis

$$\mathfrak{c} = \mathfrak{g}^u = (297\mathbb{Z} + \frac{295 + 23\sqrt{-1019}}{2}\mathbb{Z}) = (297, 295)$$

in der Gruppe $\mathrm{Ker}(\phi_{Cl}^{-1})$.

Dieses DL-Problem in $\mathrm{Ker}(\phi_{Cl}^{-1})$ reduzieren wir durch den Isomorphismus ψ^{-1} : $\mathrm{Ker}(\phi_{Cl}^{-1}) \xrightarrow{\sim} \mathbb{F}_p^*$ aus Satz 2.2.11 auf das entsprechende DL-Problem in $\mathbb{F}_p^*$.

Hierzu berechnen wir zunächst mit Std2Gen (Algorithmus 7) die entsprechenden Erzeugerdarstellungen

$$\beta = \mathsf{Std2Gen}(\mathfrak{b}, p) = -8 + \omega = (-8, 1)$$

und

$$\gamma = \mathsf{Std2Gen}(\mathfrak{c}, p) = -7 + \omega = (-7, 1),$$

wobei $\omega = \frac{1+\sqrt{-1019}}{2}$, und bilden diese schließlich mit ψ^{-1} (Ker2Fp, Algorithmus 11) auf Elemente in $\mathbb{F}_p^*$ ab.

Dafür ist es nötig die Nullstellen $\rho, \bar{\rho} \in \mathbb{F}_p^*$ von $f(X) = X^2 - X + \frac{1-\Delta}{4} \pmod p$ zu kennen. Für die Ermittlung der Nullstellen verwenden wir die Routine GetRoots (Algorithmus 9) und erhalten

$$(\rho, \bar{\rho}) = \mathsf{GetRoots}(\Delta, p) \equiv (14, 10) \pmod p$$

Die Berechnung von ψ^{-1} liefert

$$b = \mathsf{Ker2Fp}((x_\beta, y_\beta), p, \rho, \bar{\rho}) = \mathsf{Ker2Fp}((-8, 1), 23, 14, 10) \equiv 3 \pmod{23}$$

und

$$c = \mathsf{Ker2Fp}((x_\gamma, y_\gamma), p, \rho, \bar{\rho}) = \mathsf{Ker2Fp}((-7, 1), 23, 14, 10) \equiv 10 \pmod{23}.$$

Schließlich berechnen wir den Diskreten Logarithmus v, so dass $c^v \equiv b \pmod p$. Wir lösen $10^v \equiv 3 \pmod{23}$ und erhalten $v = 20$.

Setzen wir die beiden Teilergebnisse x_1 und v – wie in ReduceDLP (Algorithmus 14) – zusammen, so erhalten wir mit $x = uv + x_1 = 13 \cdot 20 + 9 = 269$ die Lösung des ursprünglichen DL-Problemes $\mathfrak{g}^x \sim \mathfrak{a}$ in $Cl(\Delta p^2)$.

Reduktion des Wurzel-Problemes

In diesem Abschnitt wollen wir ReduceROOT (Algorithmus 15) zur Lösung eines Quadratwurzel-Problemes in $Cl(\Delta p^2)$ mit $\Delta p^2 = -1019 \cdot 23^2$ verwenden.

Wir wollen die Quadratwurzel von $\mathfrak{g} = 165\mathbb{Z} + \frac{163+23\sqrt{-1019}}{2}\mathbb{Z} = (165, 163)$ berechnen. Hierfür berechnen wir zuerst

$$\mathfrak{G} = \phi_{Cl}^{-1}(\mathfrak{g}) = 11\mathbb{Z} + \frac{-9+\sqrt{-1019}}{2} = (11, -9)$$

.

Im nächsten Schritt muss eine Quadratwurzel aus $\mathfrak{G}$ in der Klassengruppe $Cl(\Delta)$ berechnet werden. Hierfür müsste im Allgemeinen der Gaußsche Algorithmus verwendet werden. Da es sich in unserem Fall aber um eine Primdiskriminante ($|\Delta| = |-1019| \equiv 3 \pmod 4$ prim) handelt, ist die Klassenzahl $h(\Delta)$ ungerade. Da wir $h(\Delta) = 13$ als bekannt voraussetzen, kann nach Proposition 2.3.3 die Quadratwurzel aus $\mathfrak{G}$ durch eine Exponentiation mit $(h(\Delta)+1)/2$ berechnet werden. Wir erhalten also

$$\mathfrak{S} = \mathfrak{G}^{(h(\Delta)+1)/2} = \mathfrak{G}^7 = 5\mathbb{Z} + \frac{-1+\sqrt{-1019}}{2}\mathbb{Z} = (5, -1).$$

Nun berechnen wir

$$\mathfrak{s} = \mathsf{GoToNonMaxOrder}(\mathfrak{S}, p) = 5\mathbb{Z} + \frac{-3 + 23\sqrt{-1019}}{2}\mathbb{Z} = (5, -3)$$

und

$$\mathfrak{h} = \mathfrak{g}\mathfrak{s}^{-2} = 311\mathbb{Z} + \frac{-277 + 23\sqrt{-1019}}{2}\mathbb{Z} = (311, -277)$$

und müssen schließlich eine Quadratwurzel aus $\mathfrak{h} \in \mathrm{Ker}(\phi_{Cl}^{-1})$ berechnen. Dies erledigen wir unter Verwendung des Isomorphismus ψ^{-1} durch die Berechnung einer Quadratwurzel in $\mathbb{F}_p^*$. Wir berechnen zuerst die Erzeugerdarstellung

$$\eta = \mathsf{Std2Gen}(\mathfrak{h}, p) = (x_\eta + y_\eta\omega) = 7 + \omega = (7, 1),$$

wobei $\omega = \frac{1+\sqrt{-1019}}{2}$, und wenden dann den Isomorphismus ψ^{-1} an.

Hierfür benötigen wir, genau wie im obigen Beispiel, die beiden Nullstellen $\rho, \bar{\rho} \in \mathbb{F}_p^*$ von $f(X) = X^2 - X + \frac{1-\Delta}{4} \pmod p$. Wir erhalten

$$(\rho, \bar{\rho}) = \mathsf{GetRoots}(\Delta, p) \equiv (14, 10) \pmod p.$$

Nun berechnen wir

$$h = \mathsf{Ker2Fp}((x_\eta, y_\eta), p, \rho, \bar{\rho}) = \mathsf{Ker2Fp}((7, 1), 23, 14, 10) \equiv 8 \pmod{23}$$

und t, so dass $t^2 \equiv h \pmod p$. Da $p = 23 \equiv 3 \pmod 4$ können wir nach [**BS96**, Corollary 7.1.2] auch die Berechnung der Quadratwurzel in $\mathbb{F}_p^*$, sofern sie existiert, durch eine Exponentiation mit $(p+1)/4 = 6$ durchführen. Wir erhalten schließlich

$$t \equiv h^{(p+1)/4} \equiv 8^6 \equiv 13 \pmod{23}.$$

Es bleibt noch die Rücktransformation des Zwischenergebnisses t in die Gruppe $Cl(\Delta p^2)$ und das Zusammensetzen der Ergebnisse. Wir müssen also zunächst $\mathfrak{t} = \psi(t)$ berechnen. Hierfür berechnen wir mit $\mathsf{Fp2Ker}$ die Erzeugerdarstellung $\tau = x_\tau + y_\tau\omega$, so dass $\varphi(\tau) \sim \mathfrak{t}$. Wir erhalten

$$\tau = \mathsf{Fp2Ker}(t, p, \rho, \bar{\rho}) = \mathsf{Fp2Ker}(13, 23, 14, 10) = 1 - 12\omega = (1, -12)$$

und danach mittels $\mathsf{Gen2Std}$ die Standarddarstellung

$$\mathfrak{t} = \mathsf{Gen2Std}(\tau, p) = \mathsf{Gen2Std}((1, -12), 23) = 257\mathbb{Z} + \frac{-69 + \sqrt{-1019}}{2}\mathbb{Z} = (257, -69).$$

Schließlich erhalten wir die gesuchte Quadratwurzel

$$\mathfrak{r} = \mathfrak{s}\mathfrak{t} = (5, -3)(257, -69) = 171\mathbb{Z} + \frac{-101 + 23\sqrt{-1019}}{2}\mathbb{Z} = (171, -101).$$

KAPITEL 3

Kryptosysteme auf Basis imaginärquadratischer Nichtmaximalordnungen

Dieser Abschnitt ist der Konstruktion von Kryptosystemen auf Basis imaginärquadratischer Nichtmaximalordnungen gewidmet.

Da die Klassengruppe $Cl(\Delta)$ einer beliebigen imaginärquadratischen Ordnung eine endliche, abelsche Gruppe ist, in der jede Klasse durch einen effizient berechenbaren Vertreter eindeutig dargestellt werden kann, ist es naheliegend, DL-basierte Kryptosysteme in $Cl(\Delta)$ zu konstruieren.

Dieser Ansatz wurde erstmals in [**BW88**] verfolgt, wobei – wie auch bei allen in [**Ham02**] betrachteten Verfahren – ausschließlich in der Maximalordnung gearbeitet wurde. Die Attraktivität dieses Ansatzes ist darauf zurückzuführen, dass das zu Grunde liegende DL-Problem mindestens so schwer, und im Hinblick auf existierende Algorithmen vermutlich echt schwieriger, ist, wie das populäre Faktorisierungsproblem.

Leider hat dieses vermeintliche Plus an Sicherheit seinen Preis. Obwohl die zu Grunde liegende Gruppenoperation – wie in [**BB99**] gezeigt – quadratische Laufzeit besitzt, so sind Kryptosysteme auf Basis imaginärquadratischer Maximalordnungen in der Praxis deutlich weniger effizient, als populäre Verfahren in $(\mathbb{Z}/n\mathbb{Z})^*$, $\mathbb{F}_q^*$ oder Elliptischen Kurven $\mathrm{EC}(\mathbb{F}_q)$.

In dieser Arbeit wird ein etwas anderer Ansatz verfolgt. Bei den hier vorgestellten Kryptosystemen auf Basis imaginärquadratischer Nichtmaximalordnungen begnügen wir uns mit dem durch das Faktorisierungsproblem gegebenen Sicherheitsniveau und zielen auf die Steigerung der Effizienz ab.

In [**HJPT98**] wurde vorgeschlagen, anstatt einer Maximalordnung $\mathcal{O}_\Delta$ eine Nichtmaximalordnung $\mathcal{O}_{\Delta p^2}$ für die Konstruktion eines ElGamal-artigen Verschlüsselungssystemes zu verwenden. Hierbei ist der Führer p, und somit auch die Maximalordnung $\mathcal{O}_\Delta$, Teil des geheimen Schlüssels. Deshalb können die in Abschnitt 2.1 diskutierten Algorithmen verwendet werden, um die wesentlichen Teile der *Ent*schlüsselung in der Maximalordnung durchzuführen, was auf Grund der kleineren Koeffizienten zu einer signifikanten Steigerung der Effizienz führt. Außerdem kann durch Anwendung der Problemreduktion aus Abschnitt 2.3.1.1 ein nichtinteraktives identitätsbasiertes ElGamal-artiges Verschlüsselungsverfahren konstruiert werden. Eine andere Variation des ursprünglichen Ansatzes, bei dem, wie in [**PT00**] vorgeschlagen, großteils im Kern der Abbildung $\phi_{Cl}^{-1} : Cl(\Delta p^2) \to Cl(\Delta)$ operiert wird, führt zu äußerst effizienten Kryptosystemen. Bei den Verfahren der NICE-Familie konzentrieren wir uns in Abschnitt 3.3 auf die Signatursysteme und zeigen, wie Schnorr- [**Sch90**] und Guillou-Quisquater-artige [**GQ90**] Verfahren konstruiert werden können, die eine sehr effiziente Signaturerzeugung ermöglichen.

Die Sicherheit aller Verfahren auf Basis imaginärquadratischer Nichtmaximalordnungen beruht in erster Linie nicht mehr auf dem DL- oder Wurzelproblem in

der Klassengruppe $Cl(\Delta p^2)$, sondern – wegen der in Abschnitt 2.3 vorgestellten Problemreduktionen – auf der vermuteten, durch die heute verfügbaren Algorithmen (vgl. Abschnitt 3.1.4.1) und die Rechnerkapazität bestimmte, Schwierigkeit der Faktorisierung von Zahlen der Form Δp^2. Somit wird der mögliche Sicherheitsgewinn bei Verwendung von imaginärquadratischen Ordnungen gegen andere Vorzüge, wie z.B. gesteigerte Effizienz, „eingetauscht".

Bevor wir uns in den Abschnitten 3.2 – 3.3 den verschiedenen Kryptosystemen auf Basis imaginärquadratischer Nichtmaximalordnungen widmen, tragen wir in Abschnitt 3.1 einige grundlegende Aspekte zusammen.

3.1. Grundlegende Aspekte

In diesem Abschnitt tragen wir die übergreifenden Aspekte der Kryptosysteme auf Basis imaginärquadratischer Nichtmaximalordnungen $\mathcal{O}_{\Delta p^2}$ zusammen.

Neben dem verwendeten Sicherheitsmodell und dem generellen Systemaufbau betrachten wir die Schwierigkeit der zu Grunde liegenden Berechnungsprobleme und leiten daraus notwendige Anforderungen an die Sicherheitsparameter ab.

Da der Führer p bei allen hier behandelten Verfahren ein Teil des geheimen Schlüssels ist, kommt dem Faktorisierungsproblem für Zahlen der Form Δp^2 eine besondere Bedeutung zu.

3.1.1. Sicherheitsmodell

In diesem Abschnitt tragen wir die wichtigsten Informationen über das Sicherheitsmodell zusammen, auf das später Bezug genommen werden soll. Wir begnügen uns mit einer informellen Darstellung und verweisen für präzise Definitionen auf [**Poi02**] und die dort referenzierten Arbeiten.

Bevor wir in Abschnitt 3.1.1.3 die wichtigsten Eigenschaften des sogenannten „Random Oracle Models" [**BR93**] skizzieren, werden vorher die benötigten Sicherheitsbegriffe für Verschlüsselungsverfahren (Abschnitt 3.1.1.1) und Signatursysteme (Abschnitt 3.1.1.2) in informeller Art und Weise erläutert.

3.1.1.1. Sicherheitsbegriffe für Verschlüsselungsverfahren

In diesem Abschnitt werden die im weiteren Verlauf der Arbeit benötigten Sicherheitsbegriffe für Verschlüsselungsverfahren eingeführt. Wir begnügen uns mit einer informellen Skizze der Begriffe und verweisen auf [**Poi02**, Section 2.2] und [**BDPR98**] für präzise Definitionen der Sicherheitsbegriffe bzw. der Diskussion der Zusammenhänge zwischen den hier angesprochenen Notationen.

Bei der Einführung der Sicherheitsbegriffe unterscheiden wir zwei verschiedene „Dimensionen". Neben den Zielen des Angreifers, aus denen sich die Sicherheitseigenschaften des Verschlüsselungsverfahrens ergeben, wird der davon unabhängige Angriffstypus betrachtet, der die Mächtigkeit des Angreifers angibt.

Angriffsziele und Sicherheitseigenschaften

- Schlüsselgewinn
  Gelingt es einem Angreifer den privaten Schlüssel zu erlangen, so gilt das Verschlüsselungsverfahren als vollständig gebrochen.
- Einwegeigenschaft (One-Wayness, OW)
  Eine grundlegende Eigenschaft eines Verschlüsselungsverfahrens ist die Einwegeigenschaft. Besitzt ein Angreifer neben dem Schlüsseltext nur öffentlich

zugängliche Informationen, wie beispielsweise öffentliche Schlüssel und Systemparameter, so soll er den Klartext nicht komplett ermitteln können.

- Ununterscheidbarkeit (Indistinguishability, IND)
  Eine sehr viel stärkere Sicherheitseigenschaft für ein Verschlüsselungsverfahren ist die Ununterscheidbarkeit von Verschlüsselungen. Wählt ein Angreifer zwei Klartext-Nachrichten und erhält daraufhin einen zugehörigen Schlüsseltext, so kann er nicht entscheiden, welche der beiden Klartext-Nachrichten zur Erzeugung des Schlüsseltextes verwendet wurde. Der Angreifer ist also nicht in der Lage aus dem Schlüsseltext, abgesehen von der Länge, irgendwelche Informationen über den Klartext zu erhalten. Deshalb wird hier auch von „semantischer Sicherheit“ gesprochen.
- Nicht-Verformbarkeit (Non-Malleability, NM)
  Eine Erweiterung der semantischen Sicherheit führt zu „nicht-verformbaren“ Verschlüsselungsverfahren. Damit ist gemeint, dass ein Angreifer aus einem bekannten Schlüsseltext nicht einen neuen Schlüsseltext ableiten kann, so dass die beiden Klartexte in einer sinnvollen Art und Weise in Beziehung stehen.

Mächtigkeit des Angreifers

Neben dem Ziel des Angreifers zieht man – insbesondere bei der Ununterscheidbarkeit und der Nicht-Verformbarkeit – auch die Mächtigkeit des Angreifers zur Klassifizierung heran.

- Chosen-Plaintext Attack (CPA)
  Bei diesem Angriffstypus darf der Angreifer Klartexte wählen und diese verschlüsseln. Bei Public-Key Verschlüsselungsverfahren kann ein solcher Angriff selbstverständlich durch jeden legitimen Nutzer durchgeführt werden.
- Plaintext-Checking Attack (PCA)
  Bei diesem in [**OP01**] eingeführten Angriffstyp wird unterstellt, dass der Angreifer ein Orakel besitzt, das bei Eingabe eines Klar- und Schlüsseltextpaares entscheidet, ob diese zueinander korrespondieren oder nicht.
- Non-Adaptive Chosen-Ciphertext Attack (CCA1)
  Hat der Angreifer Zugriff auf ein Entschlüsselungsorakel, das beliebige, vor dem Beginn des eigentlichen Angriffs gewählte, Schlüsseltexte entschlüsselt, so spricht man von einem nicht-adaptiven Angriff mit gewählten Schlüsseltexten, der bezeichnenderweise auch „Lunchtime Attack“ genannt wird.
- Adaptive Chosen-Ciphertext Attack (CCA2)
  Darf der Angreifer die Fragen an das Entschlüsselungsorakel auch während des eigentlichen Angriffs spezifizieren, und somit an vorherige Antworten des Orakels anpassen, so spricht man von einem adaptiven Angriff mit gewählten Schlüsseltexten.

Relationen zwischen Sicherheitseigenschaften

Durch die Kombination der Sicherheitseigenschaft mit der Mächtigkeit des Angreifers ergeben sich verschiedene Stufen der Sicherheit.

Wie in [**BDPR98**, Theorem 3.1] gezeigt, impliziert die Nicht-Verformbarkeit (NM) eines Verschlüsselungsverfahren bei den verschiedenen dort betrachteten Mächtigkeiten des Angreifers (CPA, CCA1 und CCA2) die Sicherheitseigenschaft der Ununterscheidbarkeit (IND) und dadurch auch die Einwegeigenschaft (OW).

Umgekehrt wurde in [**BDPR98**, Theorem 3.3] gezeigt, dass IND-CCA2 auch NM-CCA2 impliziert, weshalb diese beiden Sicherheitseigenschaften äquivalent sind. Ein Verschlüsselungsverfahren, das eine dieser beiden Eigenschaften nachweislich besitzt, gilt als besonders sicher.

3.1.1.2. Sicherheitsbegriffe für Signaturverfahren

In ähnlicher Art und Weise werden die Sicherheitsbegriffe für Signaturverfahren definiert. Auch hier unterscheiden wir bei der Betrachtung zwischen den Zielen des Angreifers und dessen Mächtigkeit. Wir begnügen uns mit einer informellen Beschreibung der Begriffe und verweisen auf [**Poi02, GMR88**] für eine formale Behandlung der Thematik.

Angriffsziele und Sicherheitseigenschaften

- Schlüsselgewinn
  Gelingt es einem Angreifer den privaten Schlüssel zu erlangen, so gilt das Signaturverfahren als vollständig gebrochen.
- Universelle Fälschung
  Ein Angreifer kann auch ohne Kenntnis des geheimen Schlüssels versuchen, einen Algorithmus zu finden, der mit großer Wahrscheinlichkeit gültige Signaturen für Nachrichten erzeugt, auf dessen Gestalt er keinen Einfluß hat. Gelingt dies dem Angreifer, so spricht man von einer universellen Fälschung.
- Existenzielle Fälschung
  Ein leichter erreichbares Ziel für einen Angreifer ist die Erstellung eines gültigen Paares aus Nachricht und Signatur, wobei der Angreifer die Nachricht selbst wählen darf.

Mächtigkeit des Angreifers

Neben dem Ziel des Angreifers zieht man auch hier die Mächtigkeit des Angreifers zur Klassifizierung heran.

- No-Message Attack
  Bei diesem Angriff sind dem Angreifer keine Nachrichten und Signaturen bekannt – er kennt lediglich die Systemparameter und den öffentlichen Schlüssel.
- Known-Message Attack
  Bei diesem Angriffstyp kennt der Angreifer eine Reihe von Nachrichten und dazugehörige gültige Signaturen. Allerdings hat er keinen Einfluß auf die Wahl der Nachrichten.
- Non-Adaptive Chosen-Message Attack
  Hier darf der Angreifer die Nachrichten wählen, die er von einem Signaturorakel signiert bekommt. Allerdings muß der Angreifer die Liste der zu signierenden Nachrichten vor dem eigentlichen Angriff spezifizieren.
- Adaptive Chosen-Message Attack
  Darf der Angreifer die Fragen an das Signaturorakel auch während des eigentlichen Angriffs spezifizieren, und somit an vorherige Antworten des Orakels anpassen, so spricht man von einem adaptiven Angriff mit gewählten Nachrichten.

Durch die Kombination der Sicherheitseigenschaft mit der Mächtigkeit des Angreifers ergeben sich verschiedene Stufen der Sicherheit, wobei ein Signatursystem

dann als besonders sicher gilt, wenn selbst bei einem adaptiven Angriff mit gewählten Nachrichten eine existenzielle Fälschung unmöglich ist.

3.1.1.3. Sicherheitsaussagen im „Random Oracle Model“

Um Sicherheitsaussagen für die vorgestellten Verfahren zu erhalten, verwenden wir das in [**BR93**] eingeführte „Random Oracle Model“. Hierbei spezifiziert man ein kryptographisches Protokoll unter Verwendung eines Zufallsorakels[1], beweist Aussagen über dessen Sicherheit und ersetzt das Zufallsorakel schließlich durch eine kryptographische Hashfunktion. Durch diese Vorgehensweise erhält man streng genommen zwar *keinen Beweis* für die Sicherheit des kryptographischen Protokolls, aber dennoch für die Praxis oft wertvolle *heuristische Aussagen*, die von den Sicherheitsaspekten, die mit der verwendeten Hashfunktion zusammenhängen, abstrahieren.

Wird das theoretische Zufallsorakel in der Praxis durch eine ungeeignete Hashfunktion ersetzt, so erweisen sich auch die im idealisierten Modell gültigen Aussagen in der Praxis als falsch. Außerdem wurden in [**CGH98, Nie02, GT03, BBP03**] verschiedene Protokolle betrachtet, die einen Sicherheitsbeweis im idealisierten Random Oracle Model zulassen, aber bei der praktischen Instanziierung des Zufallsorakels mit *beliebigen* Funktionen(-ensembles) die „bewiesene“ Sicherheitseigenschaft verlieren.

3.1.2. Grundsätzlicher Systemaufbau

In diesem Abschnitt erläutern wir den grundsätzlichen Aufbau aller hier vorgestellten Kryptosysteme. Zum Schutz vor $p \pm 1$-Methoden zur Faktorisierung [**Pol74, Wil82, SL84**] verwenden wir „starke Primzahlen“ p, so dass $p-1$ und $p+1$ jeweils einen großen Primfaktor besitzen. Wie in [**Gor85**] gezeigt wurde, ist der Aufwand für das Finden solcher Primzahlen nur unwesentlich höher als bei einer völlig zufälligen Wahl.

Der Systemaufbau geschieht nun in folgenden Schritten:

1. Wähle zufällige, starke Primzahl $d \equiv 3 \pmod 4$, so dass $2^{l_\Delta - 1} < d < 2^{l_\Delta}$.
2. Setze $\Delta = -d$.
3. Wähle zufällige, starke Primzahl p, so dass $d < p < 2^{l_\Delta}$ und $\left(\frac{\Delta}{p}\right) = 1$.
4. Δp^2 ist Teil des öffentlichen Schlüssels.
5. Δ und p sind Teil des geheimen Schlüssels.

Da bei diesem Setup $|\Delta p^2|^{1/6}\sqrt{1/8} < \sqrt{|\Delta|/4}$ gilt, ist jede Klasse, deren reduzierter Vertreter $\mathfrak{a}$ die Ungleichung $\mathcal{N}(\mathfrak{a}) < |\Delta p^2|^{1/6}\sqrt{1/8}$ erfüllt, ein Element der Menge $\mathcal{SI}(\Delta p^2)$ für das (1.3.2) gilt.

3.1.3. Die zu Grunde liegenden Berechnungsprobleme

In diesem Abschnitt soll die grundsätzliche Schwierigkeit der Berechnungsprobleme, die den in dieser Arbeit vorgestellten Kryptosystemen zu Grunde liegen, näher untersucht werden.

Da keine brauchbaren unteren Schranken für die Schwierigkeit kryptographischer Probleme existieren, wollen wir uns hier mit relativen Aussagen begnügen und vergleichen die verschiedenen Probleme insbesondere mit dem Faktorisierungsproblem

[1] Seien m und n positive, ganze Zahlen. Dann ist ein Zufallsorakel R eine Abbildung von $\{0,1\}^m$ nach $\{0,1\}^n$, bei der für alle $x \in \{0,1\}^m$ jedes Bit von $R(x)$ gleichverteilt und unabhängig ist.

für Zahlen der Form Δp^2, das wir mit $\mathrm{FP}_{\Delta p^2}$ bezeichnen wollen. Dieses Problem bietet sich als Referenzproblem an, da es eine natürliche obere Schranke für die Sicherheit der hier vorgestellten Verfahren liefert.

Bei den folgenden Betrachtungen stützen wir uns auf die generischen Aussagen aus [**Ham02**, Kapitel 6.1] und entwickeln diese für unseren Anwendungsfall weiter.

Existiert ein *deterministischer* Algorithmus mit polynomieller Laufzeit, der die Lösung eines Problemes $\mathcal{A}$ auf die Lösung des Problemes $\mathcal{B}$ reduziert, so schreiben wir $\mathcal{B} \longrightarrow \mathcal{A}$. Gilt außerdem $\mathcal{A} \longrightarrow \mathcal{B}$, so schreiben wir $\mathcal{A} \longleftrightarrow \mathcal{B}$ und nennen die beiden Probleme $\mathcal{A}$ und $\mathcal{B}$ (unter deterministischen Polynomialzeitreduktionen) äquivalent.

Sind die zur Reduktion verwendeten Algorithmen nicht deterministisch, sondern *probabilistischer* Natur, so schreiben wir $\mathcal{B} \dashrightarrow \mathcal{A}$, $\mathcal{A} \dashrightarrow \mathcal{B}$ und $\mathcal{A} \dashleftarrow\!\dashrightarrow \mathcal{B}$.

3.1.3.1. Diskrete-Logarithmus- und Diffie-Hellman-Probleme

Hier wollen wir einige im weiteren Verlauf der Arbeit angewandte Diskrete-Logarithmus- und Diffie-Hellman-Probleme definieren und deren Schwierigkeit mit $\mathrm{FP}_{\Delta p^2}$, also dem Faktorisierungsproblem für Zahlen der Form Δp^2, vergleichen.

Für alle folgenden Definitionen sei G eine endliche, abelsche Gruppe.

- Discrete Logarithm Problem (DLP):
  Sei $\mathfrak{g}, \mathfrak{a} \in G$ gegeben. Berechne das kleinste $x \in \mathbb{Z}_{>0}$, so dass $\mathfrak{g}^x = \mathfrak{a}$ oder entscheide, dass kein solches x existiert.
- Computational Diffie-Hellman Problem (CDHP):
  Sei das G-Tripel $(\mathfrak{g}, \mathfrak{g}^a, \mathfrak{g}^b)$ gegeben. Berechne $\mathfrak{g}^{ab}$.
- Decisional Diffie-Hellman Problem (DDHP):
  Sei das G-Quadrupel $(\mathfrak{g}, \mathfrak{g}^a, \mathfrak{g}^b, \mathfrak{g}^c)$ gegeben. Entscheide, ob $c \equiv ab \mod |\langle \mathfrak{g} \rangle|$.
- Gap Diffie-Hellman Problem (GDHP):
  Sei das G-Tripel $(\mathfrak{g}, \mathfrak{g}^a, \mathfrak{g}^b)$ gegeben. Berechne $\mathfrak{g}^{ab}$, wobei ein Orakel für das DDHP zur Verfügung steht.

Betrachten wir die hier definierten Probleme in einer bestimmten Gruppe, so deuten wir dies im Index an. Beispielsweise bezeichnet DLP_G das Diskrete-Logarithmus Problem in der Gruppe G.

Es ist leicht zu sehen, dass unabhängig von der konkreten Gruppe $\mathrm{DLP} \longrightarrow \mathrm{CDHP} \longrightarrow \mathrm{GDHP}$ gilt.

Für beliebige negative, zusammengesetzte Diskriminanten Δ wissen wir nach [**BW88**], dass $\mathrm{CDHP}_{Cl(\Delta)} \dashrightarrow \mathrm{FP}_\Delta$.

Somit lässt sich unser heutiges Wissen über die Schwierigkeit dieser Probleme folgendermaßen zusammenfassen:

$$\begin{array}{ccccc} \mathrm{DLP}_{Cl(\Delta p^2)} & \longrightarrow & \mathrm{CDHP}_{Cl(\Delta p^2)} & \longrightarrow & \mathrm{GDHP}_{Cl(\Delta p^2)} \\ & & \vdots & & \\ & & \mathrm{FP}_{\Delta p^2} & & \end{array} \tag{3.1.1}$$

Es ist nicht bekannt, ob das Faktorisieren von Δp^2 auch bereits auf die Lösung des Gap Diffie-Hellman Problemes in $Cl(\Delta p^2)$ reduziert werden kann.

Außerdem ist es eine offene Frage, ob die Lösung dieser Probleme in gewissen Untergruppen von $Cl(\Delta p^2)$, wie z.B. $\mathrm{Ker}(\phi_{Cl}^{-1})$, möglicherweise leichter ist, als in der gesamten Gruppe.

3.1.3.2. Kern-Probleme

Der Vollständigkeit halber führen wir nun auf informale Weise zwei Probleme ein, die im Zusammenhang mit dem NICE-Verschlüsselungsverfahren [**PT00**] von besonderem Interesse sind. Für formale Definitionen und Beweise sei auf [**BST02**] verwiesen.

Sei $\mathfrak{a}, \mathfrak{b} \in Cl(\Delta p^2)$. Dann nennen wir $\mathfrak{a}$ und $\mathfrak{b}$ *Kern-äquivalent* wenn $\phi_{Cl}^{-1}(\mathfrak{a}) = \phi_{Cl}^{-1}(\mathfrak{b})$.

- Decisional Kernel Problem (DKP)
  Sei eine Diskriminante Δp^2, ein Erzeuger $\mathfrak{g}$ von $\mathrm{Ker}(\phi_{Cl}^{-1})$ und ein Gruppenelement $\mathfrak{a} \in Cl(\Delta p^2)$, so dass $\varphi(\phi_{Cl}^{-1}(\mathfrak{a})) \in \mathcal{S}I(\Delta p^2)$, gegeben. Entscheide, ob $\mathfrak{a} \in \mathrm{Ker}(\phi_{Cl}^{-1})$ oder nicht.
- Smallest Kernel-Equivalent Problem (SKEP)
  Sei eine Diskriminante Δp^2, ein Erzeuger $\mathfrak{g}$ von $\mathrm{Ker}(\phi_{Cl}^{-1})$ und ein Gruppenelement $\mathfrak{a} \in Cl(\Delta p^2)$, so dass $\varphi(\phi_{Cl}^{-1}(\mathfrak{a})) \in \mathcal{S}I(\Delta p^2)$, gegeben. Berechne die Kern-äquivalente Idealklasse, deren reduzierter Vertreter unter allen Kern-äquivalenten Klassen die kleinste Norm besitzt.

Wie in [**BST02**] gezeigt, kann für eine gegebene Klasse $\mathfrak{a} \in Cl(\Delta p^2)$ die kleinste Kern-äquivalente Klasse als $\mathfrak{a}_0 = \varphi(\phi_{Cl}^{-1}(\mathfrak{a}))$ berechnet werden. Hierbei ist φ in Proposition 1.3.2 definiert und kann mit GoToNonMaxOrder (Algorithmus 2) berechnet werden; ϕ_{Cl}^{-1} ist in Proposition 1.3.5 definiert und kann mit GoToMaxCl (Algorithmus 4) berechnet werden.

Für jede Klasse $\mathfrak{a}$ aus $Cl(\Delta p^2)$ existiert eine eindeutige Darstellung $\mathfrak{a} = \mathfrak{a}_0\mathfrak{g}^r$, wobei $\mathfrak{g}$ ein Erzeuger von $\mathrm{Ker}(\phi_{Cl}^{-1})$ und $0 \leq r < p - (\Delta/p)$.

Nach [**BST02**, Theorem 6] gilt

$$\mathrm{FP}_{\Delta p^2} \longrightarrow \mathrm{SKEP} \longrightarrow \mathrm{DKP} \tag{3.1.2}$$

3.1.3.3. Wurzel-Probleme

Wie in Abschnitt 2.3.1.2 angegeben, ist das e-te Wurzelproblem, $e \in \mathbb{Z}, e \geq 2$, in einer endlichen, abelschen Gruppe G bei gegebenem $\mathfrak{g} \in G$ die Berechnung eines Elementes $\mathfrak{r} \in G$, so dass $\mathfrak{r}^e = \mathfrak{g}$, oder die Entscheidung, dass kein solches $\mathfrak{r}$ existiert.

Hier wollen wir diese Definition etwas präzisieren, um zwei verschiedene Wurzelprobleme einzuführen, die zur Konstruktion verschiedener Kryptosysteme verwendet werden können.

Auch hier sei G eine endliche, abelsche Gruppe und $\mathfrak{g} \in G$. Dann sind die beiden Probleme folgendermaßen gegeben:

- Square-Root-Problem (SRP)
  Berechne $\mathfrak{r} \in G$, so dass $\mathfrak{r}^2 = \mathfrak{g}$ oder entscheide, dass kein solches $\mathfrak{r}$ existiert.
- n-Root-Problem (n-RP)
  Sei $n \in \mathbb{Z}$, so dass $\mathrm{ggT}(n, |G|) = 1$. Berechne $\mathfrak{r} \in G$, so dass $\mathfrak{r}^n = \mathfrak{g}$.

Das Square-Root-Problem liegt dem in [**HMT98**] vorgestellten Rabin-Analog zu Grunde. Wie dort, oder beispielsweise auch in [**Mey97**], erläutert, gilt $\mathrm{SRP}_{Cl(\Delta)} \dashleftarrow \mathrm{FP}_{\Delta}$ für beliebige Diskriminanten Δ.

Das n-Root-Problem liegt dem in Abschnitt 3.3.2 vorgestellten NICE-Guillou-Quisquater-Signaturverfahren und dem RSA-Verfahren aus [**HMT98**] zu Grunde.

Hier ist im Allgemeinen nicht bekannt, wie n-$\mathrm{RP}_{Cl(\Delta)}$ mit dem Faktorisierungsproblem FP_{Δ} (direkt) zusammenhängt.

Wie in [**Ham02**, Abschnitt 6.1.5] angedeutet, scheint kein besseres Verfahren zur Lösung des n-Root-Problemes zu existieren, als vorher die Ordnung von $\mathfrak{g}$ in der Gruppe G, oder ein Vielfaches davon, wie z. B. die Gruppenordnung $|G|$ selbst, zu berechnen. Ist mit o die Ordnung von $\mathfrak{g}$ in G gegeben, so kann x so dass $xn \equiv 1 \pmod{o}$ und schließlich mit $\mathfrak{g}^x$ die Lösung des n-Root-Problemes berechnet werden.

Das Problem, bei gegebenem $\mathfrak{g} \in G$ die Ordnung von $\mathfrak{g}$, oder ein Vielfaches davon, zu bestimmen, nennen wir das *Order-Problem* (OP).

Die Lösung des Order-Problemes kann übrigens durch eine DL-Berechnung erreicht werden. Ist x der Diskrete Logarithmus von $\mathfrak{g}^{-1}$ zur Basis $\mathfrak{g}$, d.h. die kleinste, positive Lösung der Gleichung $\mathfrak{g}^x = \mathfrak{g}^{-1}$, so ist die Ordnung von $\mathfrak{g}$ offensichtlich $x+1$. Deshalb gilt $\mathrm{DLP} \longrightarrow \mathrm{OP}$ in beliebigen endlichen, abelschen Gruppen.

Wie oben erläutert, gilt außerdem $\mathrm{OP}_{Cl(\Delta)} \longrightarrow n\text{-}\mathrm{RP}_{Cl(\Delta)}$ und nach [**Ham02**, Proposition 6.1.8] auch $\mathrm{OP}_{Cl(\Delta)} \dashrightarrow \mathrm{FP}_{\Delta}$ für beliebige negative Diskriminanten Δ – eine direkte Relation zwischen FP_{Δ} und $n\text{-}\mathrm{RP}_{Cl(\Delta)}$ ist bisher nicht bekannt.

Etwas anders stellt sich die Situation dar, wenn wir uns – wie beim NICE-Guillou-Quisquater-Signaturverfahren in Abschnitt 3.3.2 – auf Nichtfundamental-Diskriminanten Δp^2 und die Gruppe $\mathrm{Ker}(\phi_{Cl}^{-1}) \subset Cl(\Delta p^2)$ beschränken. In diesem Fall ist nämlich die Gruppenordnung durch $|\mathrm{Ker}(\phi_{Cl}^{-1})| = p - \left(\frac{\Delta}{p}\right)$ gegeben und die Berechnung q-ter Wurzeln in $\mathrm{Ker}(\phi_{Cl}^{-1})$ kann offensichtlich durch einen deterministischen Algorithmus mit polynomieller Laufzeit auf die Faktorisierung von Δp^2 zurückgeführt werden.

Somit läßt sich unser heutiger Kenntnisstand über die Schwierigkeit der Wurzelprobleme folgendermaßen zusammenfassen:

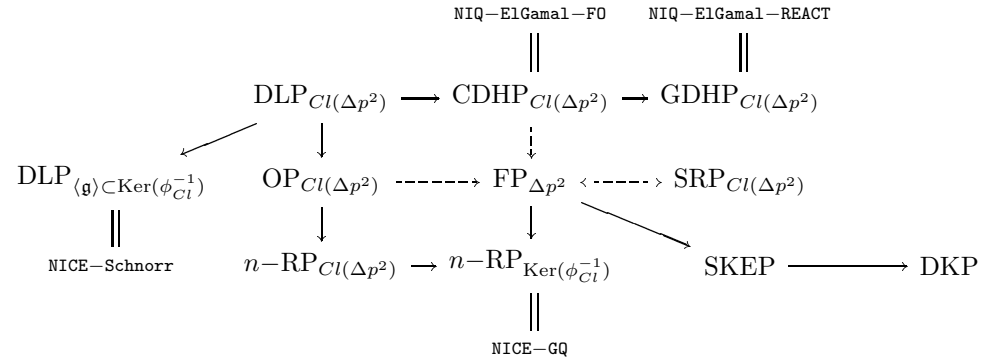
$$\begin{array}{ccccc} \mathrm{DLP}_{Cl(\Delta p^2)} & \longrightarrow & \mathrm{OP}_{Cl(\Delta p^2)} & \longrightarrow & n\text{-}\mathrm{RP}_{Cl(\Delta p^2)} \\ & & \vdots & & \downarrow \\ \mathrm{SRP}_{Cl(\Delta p^2)} & \dashleftarrow & \mathrm{FP}_{\Delta p^2} & \longrightarrow & n\text{-}\mathrm{RP}_{\mathrm{Ker}(\phi_{Cl}^{-1})} \end{array} \tag{3.1.3}$$

3.1.4. Notwendige Sicherheitsparameter

In diesem Abschnitt betrachten wir konkrete Algorithmen für die Lösung der oben eingeführten Berechnungsprobleme und leiten daraus notwendige Eigenschaften der Sicherheitsparameter unserer Kryptosysteme ab.

Der Übersicht halber fassen wir die Aussagen aus (3.1.1), (3.1.2) und (3.1.3) in einem Diagramm zusammen, in dem den Berechnungsproblemen auch die in dieser Arbeit näher betrachteten Kryptosysteme zugeordnet sind:

(3.1.4)



Hierbei ist $\langle \mathfrak{g} \rangle \subset \mathrm{Ker}(\phi_{Cl}^{-1}) \subset Cl(\Delta p^2)$ eine Untergruppe mit Primzahlordnung $q > 2^{160}$, die beim NICE-Schnorr-Signaturverfahren (Abschnitt 3.3.1) verwendet wird und $=\!=$ bedeutet, dass (für gewisse Varianten der Kryptosysteme) entsprechende Äquivalenzbeweise im Random-Oracle-Modell [**BR93**] existieren.

In dieser Darstellung ist leicht erkennbar, dass alle in dieser Arbeit verwendeten Berechnungsprobleme auf die Berechnung Diskreter Logarithmen in $Cl(\Delta p^2)$ zurückgeführt werden können. Erweist sich das DL-Problem als leicht lösbar, so sind alle hier vorgestellten Kryptosysteme unsicher.

Deshalb müssen die grundlegenden Parameter Δ und p zumindest so gewählt werden, dass die Berechnung Diskreter Logarithmen in $Cl(\Delta p^2)$ nicht möglich ist.

Umgekehrt sind keine Algorithmen zur Lösung der Diffie-Hellman- und n-ten Wurzel-Probleme bekannt, die nicht auch (in ähnlicher Form) zur Berechnung Diskreter Logarithmen in $Cl(\Delta p^2)$ eingesetzt werden könnten. In der Praxis löst man die Diffie-Hellman-Probleme direkt durch DL-Berechnungen. Das n-te Wurzel-Problem in $Cl(\Delta p^2)$ löst man durch die Lösung des Order-Problemes, das wiederum durch die Berechnung der Klassenzahl $h(\Delta p^2) = h(\Delta)\left(p - \left(\frac{\Delta}{p}\right)\right)$ gelöst wird. In all diesen Fällen verwendet man in der Praxis eine Variante des Multiple-Polynomial-Quadratic-Sieve (MPQS) [**Jac99**] mit einer erwarteten Laufzeit von $L_{|\Delta|}\left[\frac{1}{2}, 1 + o(1)\right]$.

Außerdem kann durch die in Abschnitt 2.3 vorgestellten Reduktionen das DL- und Wurzelproblem in $Cl(\Delta p^2)$ bei bekanntem Führer p effizient auf analoge Probleme in $Cl(\Delta)$ und $\mathbb{F}_p^*$ reduziert werden.

Deshalb muss Δ und p – auch bei den Kryptosystemen, die originär nicht (nur) auf dem Faktorisierungsproblem basieren – so gewählt werden, dass die Faktorisierung von Δp^2 nicht möglich ist.

Deshalb betrachten wir hier schließlich verschiedene Algorithmen zur Berechnung Diskreter Logarithmen in $Cl(\Delta p^2)$ und zur Faktorisierung von Δp^2, um daraus notwendige Eigenschaften für Δ und p ableiten zu können.

3.1.4.1. Algorithmen zur Faktorisierung von Δp^2

Wie in Abschnitt 2.3 gezeigt, können das DL-Problem und das Wurzel-Problem in $Cl(\Delta p^2)$ *bei Kenntnis des Führers* p effizient auf analoge Probleme in $Cl(\Delta)$ und $\mathbb{F}_p^*$ reduziert werden.

Um diese Problemreduktionen zu verhindern, müssen Δ und p so gewählt werden, dass die Faktorisierung von Δp^2 praktisch nicht möglich ist. Hierfür betrachten wir die heute bekannten Faktorisierungsalgorithmen.

Zahlkörpersieb

Das Zahlkörpersieb [**LL93a**] basiert auf einer Idee von J. M. Pollard [**Pol93**] bei der, wie auch beim quadratischen Sieb [**Pom85**], eine Kongruenz $x^2 \equiv y^2 \pmod{n}$, $x \not\equiv \pm y \pmod{n}$, konstruiert wird, mit der sich durch Berechnung von $\mathrm{ggT}(x-y, n)$ eine möglicherweise nicht-triviale Faktorisierung von n erhalten lässt.

Das Zahlkörpersieb hat eine erwartete Laufzeit von

$$L_n\left[\frac{1}{3}, v + o(1)\right] \tag{3.1.5}$$

und ist deshalb dem quadratischen Sieb mit einer Laufzeit $L_n\left[\frac{1}{2}, u + o(1)\right]$ bei Zahlen ab etwa 120 Dezimalstellen überlegen. Der Parameter v in (3.1.5) hängt von der Form der zu faktorisierenden Zahl n und weiteren Implementierungsdetails ab.

Um die weitere Diskussion zu erleichtern, rufen wir zu Beginn die wesentlichen Phasen des Zahlkörpersiebes ins Gedächtnis zurück. Wir begnügen uns hier mit einer groben Skizze und verweisen auf [**SS02**] für eine umfassendere Behandlung der Thematik.

1. *Initialisierung mit Polynomwahl.* Wähle Polynome $f_i \in \mathbb{Z}[X]$, so dass $f_i(\rho_i) = 0$ und $f_i(m) \equiv 0 \pmod{n}$, so dass es sich bei den Abbildungen $\phi_i : \mathbb{Z}[\rho_i] \to \mathbb{Z}/n\mathbb{Z}, \sum s_j \rho_i^j \mapsto \sum s_j m^j$ um Ringhomomorphismen handelt. In der Praxis wird meist mit einem Polynom f_1 vom Grad
$$d \approx \left(\frac{3 \log n}{\log \log n}\right)^{1/3} \tag{3.1.6}$$
und einem linearen Polynom $f_2 = x - m$ gearbeitet.
2. *Sieb zur Ermittlung von Relationen.* Ermittle Relationen $(a, b) \in \mathbb{Z}^2$, so dass die Elemente $a + b\rho_i$, oder zumindest ein Abbild derselben, über geeignete Faktorbasen zerlegbar sind. In der Praxis geschieht dies durch Siebmechanismen.
3. *Gleichungssystem zur Konstruktion der Quadrate.* Ermittle eine Teilmenge S der soeben ermittelten Relationen, so dass $\alpha_i^2 = \prod_{(a,b)\in S} a + b\rho_i$ ein Quadrat in $\mathbb{Z}[\rho_i]$ ist. In der Praxis geschieht dies durch Lösen eines linearen Gleichungssystems über $\mathbb{F}_2$.
4. *Wurzelberechnung und Abbildung nach $\mathbb{Z}/n\mathbb{Z}$.* Sofern sich α_i nicht bereits aus der Faktorbasendarstellung von α_i^2 ablesen lässt, muss α_i berechnet werden. Daraufhin werden die berechneten α_i nach $\mathbb{Z}/n\mathbb{Z}$ abgebildet und versucht n durch Berechnung von $\mathrm{ggT}(\phi_1(\alpha_1) \pm \phi_2(\alpha_2), n)$ zu faktorisieren.

Es existieren verschiedene Varianten des Zahlkörpersiebes, die sich in der einen oder anderen Phase und in der asymptotischen Laufzeit unterscheiden.

Aus dem ursprünglichen Vorschlag [**Pol93**] für die Faktorisierung von Zahlen der Form $n = x^3 + k$ entstand das *Spezielle Zahlkörpersieb* (Special Number Field Sieve, SNFS) [**LLMP93**], mit dem Zahlen der Form[2]

$$n = r^e - s \tag{3.1.7}$$

mit kleinem r und $|s|$ in $L_n\left[\frac{1}{3}, v_{\text{ SNFS}} + o(1)\right]$ Bit-Operationen mit

$$v_{\text{ SNFS}} = \sqrt[3]{\frac{32}{9}} \approx 1.527 \tag{3.1.8}$$

faktorisiert werden können. Die im Vergleich zu den anderen Varianten des Zahlkörpersiebes günstige asymptotische Laufzeit (vgl. (3.1.9) und (3.1.10)) und die praktischen Vorteile (vgl. Tabelle 2 und Tabelle 3) sind insbesondere[3] darauf zurückzuführen, dass sich auf Grund der Form von n in (3.1.7) in der ersten Phase des Algorithmus auf einfache Art und Weise ein Polynom f_1 wählen lässt, das die Erzeugung der Relationen (Schritt 2) und die Konstruktion der Quadrate (Schritt 3) vergleichsweise leicht macht.

Unter den größten bislang mit dem SNFS faktorisierten Zahlen finden sich folgende:

Zahl	Datum	Referenz
$2^{808} - 1$	Januar 2003	[**FBK**$^+$]
$2^{773} + 1$	November 2000	[**CDL**$^+$**00b**]
$2^{757} - 1$	Dezember 2003	[**LLWG03b**]
$2^{713} - 1$	September 2003	[**LLWG03a**]
$10^{211} - 1)/9$	April 1999	[**CDL**$^+$**99b**]

TABELLE 2. SNFS-Rekorde

Die Verallgemeinerung des SNFS führte zum *Allgemeinen Zahlkörpersieb* (General Number Field Sieve, GNFS) [**BLP93**] mit einer Laufzeit von $L_n\left[\frac{1}{3}, v_{\text{ GNFS}} + o(1)\right]$ mit

$$v_{\text{ GNFS}} = \sqrt[3]{\frac{64}{9}} \approx 1.923, \tag{3.1.9}$$

das zur Faktorisierung beliebig geformter Zahlen mit mindestens zwei Primteilern, also auch Zahlen der Form $n = pq$ oder $n = pq^2$, eingesetzt werden kann.

Die Unterschiede zum SNFS resultieren daraus, dass im ersten Schritt kein „einfaches“ Polynom f_1 von Hand gewählt werden kann, so dass in den weiteren Schritten im Allgemeinen kein „arithmetisch gut beherrschbarer“ Ring $\mathbb{Z}[\rho_1]$

[2] In [**EMS**$^+$**99**] wurde gezeigt, dass das SNFS eigentlich für eine etwas größere Klasse von Zahlen, nämlich der Form $n = ar^e + bs^j$, eingesetzt werden kann.

[3] Da die anderen Phasen des Algorithmus, wie die Konstruktion des algebraischen Quadrates und die Wurzelberechnung, beim GNFS inzwischen vergleichsweise gut gelöst sind, unterscheiden sich viele SNFS-Implementierungen, wie z.B. die in [**Elk96**], nur noch in der ersten Phase von einer Implementierung des Allgemeinen Zahlkörpersiebes.

zur Verfügung steht. Insbesondere würde die Zerlegung der Elemente $a + b\rho_1$ in Primideale unter diesen Voraussetzungen große Schwierigkeiten bereiten. Deshalb verzichtet man beim GNFS darauf und betrachtet lediglich die Norm $\mathrm{N}(a + b\rho_1)$ dieser Elemente, um das Quadrat α_1^2 zu konstruieren und daraus mit dem Algorithmus von Montgomery [**Mon94**], bzw. der Variante von Nguyen [**Ngu98**], die Wurzel α_i zu berechnen.

Abgesehen von der Faktorisierung des 158-stelligen Faktors von $2^{953} + 1$ [**FBK02**], stellten die Faktorisierungen der Zahlen aus der RSA-Challenge den jeweils aktuellen GNFS-Rekord dar:

Zahl	Bitlänge	Datum	Referenz
RSA576	576	Dezember 2003	[**FBK$^+$03**]
RSA160	529	April 2003	[**FBKB03**]
C158 $\mid 2^{953} + 1$	522	Januar 2002	[**FBK02**]
RSA155	512	August 1999	[**CDL$^+$00a**]
RSA140	462	Februar 1999	[**CDL$^+$99a**]
RSA130	429	April 1996	[**CDEH$^+$96**]

TABELLE 3. GNFS-Rekorde

Vergleicht man Tabelle 2 und Tabelle 3, so scheint sich der asymptotische Vorteil des SNFS, in (3.1.5) ist $v \approx 1.53$ statt $v \approx 1.92$, auch in der Praxis deutlich bemerkbar zu machen. Da die unterschiedlichen Laufzeiten insbesondere auf die besseren Polynome beim SNFS zurückzuführen sind, erscheint es lohnenswert, die Polynomwahl beim GNFS näher zu betrachten.

Abgesehen von Montgomery's Methode zur Konstruktion von zwei quadratischen Polynomen (siehe [**Elk96**, Section 5] und [**Mur99**, Section 2.3.1]) scheinen sich alle bislang bekannten Verfahren zur Konstruktion von GNFS-Polynomen auf die sogenannte „Base-m"-Methode zu stützen, bei der die Nullstelle $m = \lfloor n^{1/d} \rfloor$, mit dem Grad d aus (3.1.6), zu Beginn festgelegt wird und man das Polynom

$$f_1(x) = c_d x^d + c_{d-1} x^{d-1} + \cdots + c_1 m + c_0$$

mit $f_1(m) \equiv 0 \pmod{n}$ direkt aus der m-adischen Darstellung von

$$n = c_d m^d + c_{d-1} m^{d-1} + \cdots + c_1 m + c_0$$

ablesen kann. Wie in [**Mur99**] erläutert, lassen sich durch die initiale Variation von m und nachfolgende Transformationen (Rotation und Verschiebung) günstigere (schiefe) Polynome konstruieren. In [**Gow03**] wurde kürzlich gezeigt, wie durch gezielte Rotationen von Polynomen die Existenz von Nullstellen modulo beliebiger Primzahlen erzwungen werden kann, was die Glattheitswahrscheinlichkeit für die Funktionswerte des homogenisierten Polynomes $F_1(x, y) = y^d f_1(x/y)$ erhöht. Wie sich die so verbesserte Polynomwahl auf die praktische Laufzeit des Zahlkörpersiebs auswirkt, bleibt offen. Die Abschätzung in [**LTS$^+$03**, Section 6] deutet an, dass sich die Verbesserung beim Ertrag der Siebphase proportional zu $-\alpha_B(F)$, wie in [**Gow03**, Section 7] oder [**Mur99**] definiert, zu verhalten scheint. Allerdings ändert sich durch diese Verbesserungen nichts an der asymptotischen Laufzeit des Zahlkörpersiebes, so lange $\alpha_B(F)$ eine Konstante ist.

Anders gestaltet sich dies bei der in [**Cop93**] vorgeschlagenen NFS-Variante, die mehrere Zahlkörper verwendet und eine asymptotische Laufzeit von $L_n\left[\frac{1}{3}, v_{\text{ MNFS}} + o(1)\right]$ erreicht, wobei

$$v_{\text{ MNFS}} = \frac{1}{3}\left(92 + 26\sqrt{13}\right)^{1/3} \approx 1.902. \tag{3.1.10}$$

Allerdings ist bislang nicht klar, ob die asymptotisch bessere Laufzeit auch bei kryptographisch relevanten Größenordnungen zu praktischen Verbesserungen führen kann. Wie in [**Hür94**, Kapitel 6.2] erläutert, existieren eine Reihe von praktischen Problemen, beispielsweise im Zusammenhang mit der Behandlung großer Primzahlen und dem gesteigerten Speicherbedarf, die die praktische Eignung dieses Verfahrens fraglich erscheinen lassen.

In [**Ber01**] wurde indessen ein Schaltungsentwurf vorgeschlagen, und in [**LSTT02**] verbessert, der eine – gemessen am Produkt[4] aus Laufzeit und Speicherbedarf – asymptotische Verbesserung der Gleichungssystem-Phase verspricht, weshalb die Schwierigkeit der Faktorisierung großer Zahlen mit dem Zahlkörpersieb ausschließlich bei der Erzeugung der Relationen zu liegen scheint.

Für diese Phase wurde schließlich kürzlich der Entwurf von TWIRL (The Weizmann Institute Relation Locator) [**ST03**] vorgelegt, womit die Siebphase für die Faktorisierung einer 1024 Bit-Zahl in weniger als einem Jahr, mit weniger als $ US 10 Mio. Aufwand, möglich sein soll.

Um zu entsprechenden Abschätzungen für die notwendigen Parameter unserer Kryptosysteme zu gelangen, betrachten wir insbesondere die heute verfügbaren Algorithmen und liefern zusätzlich weitere Informationen, die im Rahmen besonders konservativer Abschätzungen herangezogen werden können.

Unter den hier skizzierten Varianten des Zahlkörpersiebes, die zur Faktorisierung von Zahlen der Form $n = pq^2$ eingesetzt werden können, existieren praktische Erfahrungen mit der Implementierung des „gewöhnlichen“ allgemeinen Zahlkörpersiebes [**BLP93**]. Wie in [**CDL$^+$00a**] berichtet, wurden etwa 8000 MIPS-Jahre[5] (MJ) zur Faktorisierung der 512-Bit-Zahl RSA-155 benötigt.

Unterstellt man nun, wie in (1.1.3), dass der Rechenaufwand etwa proportional zu

$$L_n\left[\frac{1}{3}, 1.923\right] \tag{3.1.11}$$

wächst, so kann aus diesem Datenpunkt – wie in [**Ham02**, Abschnitt 5.2.2] – die Aufwandsabschätzung in Spalte#1 von Tabelle 4 abgeleitet werden.

Führt die asymptotische Verbesserung aus [**Cop93**] zu entsprechenden Effizienzsteigerungen in der Praxis, so ist Spalte#2 der Tabelle maßgebend.

Unterstellt man weiterhin, dass sich für unseren speziellen Fall $n = pq^2$ eine asymptotisch günstigere Variante des Zahlkörpersiebes finden lässt, beispielsweise indem signifikant günstigere Polyome – idealerweise so günstig wie beim SNFS –

[4] In [**Sil00**] wurde eine ähnliche Metrik zur Aufwandsabschätzung, und damit zur Bestimmung notwendiger Paramterlängen, vorgeschlagen. Allerdings ist, wie auch in [**LSTT02**] angemerkt, die gleichberechtigte Betrachtung von Laufzeit und Speicherplatzbedarf eher im Bereich der VLSI-Hardware-Entwicklung als bei zahlentheoretischen Problemen üblich.

[5] MIPS steht für „Million Instructions Per Second“. Beispielsweise hat die DEC VAX 11/780 eine Rechenleistung von 1 MIPS; einem Pentium II - Prozessor mit x MHz Taktfrequenz kann beispielsweise eine Rechenleistung von etwa x MIPS unterstellt werden (vgl. [**LV01**, Section 1.2]).

gewählt werden können, so findet man die zugehörigen Abschätzungen in den Spalten #3-6 von Tabelle 4. Unter diesen Annahmen kann die letzte Spalte als „Potenzial“ verstanden werden, das durch eine verbesserte Polynomwahl beim Zahlkörpersieb erschlossen werden könnte.

Größenordnung von n	Erwarteter Rechenaufwand zur Faktorisierung von n mit dem Zahlkörpersieb (in MJ)						
Spalte #	1	2	3	4	5	6	$\frac{\text{Spalte\#1}}{\text{Spalte\#6}}$
v	1.923 (3.1.9)	1.902 (3.1.10)	1.8	1.7	1.6	1.527 (3.1.8)	
2^{512}	$8.00 \cdot 10^3$						
2^{768}	$4.91 \cdot 10^7$	$4.46 \cdot 10^7$	$2.81 \cdot 10^7$	$1.78 \cdot 10^7$	$1.13 \cdot 10^7$	$8.14 \cdot 10^6$	6.03
2^{1024}	$5.99 \cdot 10^{10}$	$5.04 \cdot 10^{10}$	$2.18 \cdot 10^{10}$	$9.56 \cdot 10^9$	$4.20 \cdot 10^9$	$2.30 \cdot 10^9$	26.0
2^{1536}	$5.97 \cdot 10^{15}$	$4.43 \cdot 10^{15}$	$1.04 \cdot 10^{15}$	$2.51 \cdot 10^{14}$	$6.05 \cdot 10^{13}$	$2.14 \cdot 10^{13}$	297
2^{2048}	$6.98 \cdot 10^{19}$	$4.68 \cdot 10^{19}$	$6.67 \cdot 10^{18}$	$9.89 \cdot 10^{17}$	$1.47 \cdot 10^{17}$	$3.64 \cdot 10^{16}$	$1.92 \cdot 10^3$
2^{3072}	$2.64 \cdot 10^{26}$	$1.50 \cdot 10^{26}$	$9.59 \cdot 10^{24}$	$6.47 \cdot 10^{23}$	$4.63 \cdot 10^{23}$	$6.09 \cdot 10^{21}$	$4.33 \cdot 10^4$
2^{4096}	$5.87 \cdot 10^{31}$	$2.91 \cdot 10^{31}$	$9.69 \cdot 10^{29}$	$3.45 \cdot 10^{28}$	$1.23 \cdot 10^{27}$	$1.07 \cdot 10^{26}$	$5.49 \cdot 10^5$

TABELLE 4. Erwarteter Rechenaufwand für die Faktorisierung mit dem Zahlkörpersieb

Quadratisches Sieb

Da das Zahlkörpersieb nicht nur asymptotisch, sondern bei kryptographisch relevanten Größenordnungen auch in der Praxis leistungsfähiger als das Quadratische Sieb [**Pom85, Sil87, LLD+02**] oder dessen Vorgänger ist, brauchen wir hier nicht explizit auf diese Verfahren einzugehen.

Elliptic Curve Method (ECM)

Während das Zahlkörpersieb [**BLP93**] und das Quadratische Sieb scheinbar nicht von der speziellen Gestalt der Diskriminante Δp^2 profitieren können, so existiert mit der Elliptic Curve Method (ECM) [**Len87**] ein Verfahren, das vorzugsweise die kleinen Primfaktoren einer Zahl findet und somit eine Zahl $n = pqr$, die aus (mindestens) drei Primfaktoren p, q, r zusammengesetzt ist, leichter faktorisieren kann, als eine Zahl $n' = p'q'$ in der Größenordnung von n.

Unter plausiblen Annahmen über die Verteilung der Primfaktoren zufällig gewählter Zahlen, und Gruppenordnungen zufällig gewählter Elliptischer Kurven, lässt sich zeigen, dass Lenstra's ECM [**Len87**] für das Auffinden eines Primfaktors $p \mid n$ einer gegebenen Zahl n eine erwartete Laufzeit von

$$E_{[\text{Len87}]}[n, p] = M(n) L_p \left[\frac{1}{2}, \sqrt{2} + o(1) \right] \tag{3.1.12}$$

Bitoperationen hat, wobei $M(n)$ den Aufwand der Multiplikation modulo n angibt. Beim gewöhnlichen Multiplikationsalgorithmus ist $M(n) = O\left(\log(n)^2\right)$. Bei großen Zahlen können asymptotisch günstigere Algorithmen zur Multiplikation, wie beispielsweise Karatsuba's Methode [**Knu81**, Chapter 4.3.3] mit $M(n) = O\left(\log(n)^{\log_2 3}\right)$, eingesetzt werden. Ab welcher Größenordnung die asymptotisch

günstigeren Algorithmen auch in der Praxis schneller sind, hängt von Implementierungsdetails ab und lässt sich nicht pauschal beantworten. Beispielsweise berichtet Morain [**Mor90**, Ch. 5] (vgl. auch [**Bre99**, Section 5.1]), dass Karatsuba's Methode bei Zahlen mit mehr als 800 Bit effizienter sei.

Im Laufe der Zeit wurden einige praktische Verbesserungen für das ursprüngliche ECM-Verfahren vorgeschlagen, von denen sich die meisten in [**Bre86, Bre99, Ber93, Mon92, Mül95**] wiederfinden dürften. Insbesondere wurde vorgeschlagen, das ursprüngliche Verfahren [**Len87**] durch eine zweite Phase zu ergänzen, die zu einer praktischen Verbesserung führt, da es bei zufällig gewählten Zahlen sehr wahrscheinlich ist, dass genau ein Primfaktor deutlich größer ist, als alle anderen.

Es existieren verschiedene Möglichkeiten die zweite ECM-Phase zu realisieren. In [**Bre99**, Section 3] sind die wesentlichen Aspekte der „Standard Continuation (SC)“, „Improved Standard Continuation (ISC)“ und „Birthday Paradox Continuation (BPC)“ skizziert. Die „FFT-Continuation (FFTC)“ , die bei großen Zahlen besonders günstig sein soll, ist in [**Mon92, Mül95**] ausführlich beschrieben. Die präzise Analyse der erwarteten Laufzeiten und relativen Einsparungen bei den verschiedenen Fortsetzungsvarianten ist keine triviale Aufgabe. In [**Bre99**, Section 4.4] ergab eine heuristische Analyse der ISC-Variante, dass mit einer Laufzeit von etwa

$$E_{[\text{Bre99}]}[n,p] = M(n)L_p\left[\frac{1}{2}, \sqrt{\tau(p)}\right]$$

gerechnet werden kann, wobei $\tau(p)$ für den Bereich $10^{20} < p < 10^{60}$ Werte zwischen $\tau(p) = 1.677$ (für $p \approx 10^{20}$) und $\tau(p) = 1.723$ (für $p \approx 10^{60}$) annimmt und im Unendlichen gegen $\tau(p) = 2$ strebt. In [**Bre86**] wurde argumentiert, dass durch die ISC- oder BPC-Fortsetzung eine Beschleunigung in der Größenordnung von etwa $\log(p)$ erreicht werden kann.

In [**PO96**] wurde eine spezielle ECM-Variante für Zahlen der Form $n = pq^2$ vorgeschlagen. In dieser Variante wird ausgenutzt, dass für alle $x \in \mathbb{Z}$ mit $\text{ggT}(x,n) = 1$

$$\left(\frac{x}{pq^2}\right) = \left(\frac{x}{p}\right) \tag{3.1.13}$$

gilt, und deshalb $\left(\frac{x_1+r}{pq^2}\right) \neq \left(\frac{x_2+r}{pq^2}\right)$ sofort $x_1 \not\equiv x_2 \pmod p$ impliziert. Soll nun bei einer gegebenen Liste von Zahlen $x_1, \ldots, x_m$ entschieden werden, ob $x_i \equiv x_j \pmod p$, so muss nicht mehr für jedes mögliche Paar (x_i, x_j) mit $1 \leq i, j \leq m$ die Berechnung $\text{ggT}(x_i - x_j, pq^2)$ durchgeführt werden, sondern nur noch für diejenigen Paare (x_i, x_j), für die $\left(\frac{x_i+r_k}{pq^2}\right) = \left(\frac{x_j+r_k}{pq^2}\right)$, für eine Liste zufälliger Zahlen r_k mit $1 \leq k \leq \lfloor \log_2(m) \rfloor$.

Ist Q der Punkt auf der zufällig gewählten elliptischen Kurve am Ende der ersten ECM-Phase, so besteht die Fortsetzung in [**PO96**] schlicht daraus, bei m zufällig gewählten Punkten $R_i = (x_i, y_i) \in \langle Q \rangle$, $1 \leq i \leq m$ mittels der oben skizzierten Berechnung von Jacobi-Symbolen zu entscheiden, ob $x_i \equiv x_j \pmod p$.

Eine etwas effizientere Variante für die zweite ECM-Phase für Zahlen der Form $n = pq^2$ wurde in [**Per01**] vorgeschlagen und in [**ET02a, ET02b**] näher untersucht. Hier wählt man nicht vorab m zufällige Gruppenelemente $R_i \in \langle Q \rangle$, sondern erzeugt diese – im Stil von Pollard's Rho-Methode [**Pol75, Bre80, Tes98**] – in einem pseudozufälligen Lauf durch die Gruppe $\langle Q \rangle$. Man setzt $R_0 = Q$, $R_{i+1} = f(R_i)$,

wobei

$$f(Q_i) = \begin{cases} 2R_i, & \text{falls } \left(\frac{x_i}{pq^2}\right) = 1 \\ R_i + Q & \text{sonst.} \end{cases}$$

Somit ist dieses Verfahren eine Variante der Birthday Paradox Continuation (BPC) [**Bre86**, Section 6] und [**Bre99**, Section 3.3], die – wie auch die ECM-Variante in [**PO96**] – die Eigenschaft (3.1.13) ausnutzt. Sei t_n die Ordnung des Punktes Q auf der Kurve E_n modulo n und t_p die Ordnung des Punktes Q auf der entsprechenden Kurve E_p modulo p. Dann ist $t_p \leq t_n$. Der entscheidende Vorteil durch die Verwendung des Jacobi-Symboles hier ist, dass $\left(\frac{x_i}{pq^2}\right)$ nach (3.1.13) nur von p abhängt und die pseudozufällige Sequenz R_i demnach erwartungsgemäß nicht erst nach $O(\sqrt{t_n})$, sondern bereits nach $O(\sqrt{t_p})$ Schritten periodisch wird. Demnach sollte diese Variante Zahlen der Form $n = pq^2$ auch etwas schneller faktorisieren können, als gewöhnliche Fortsetzungen, wie BPC und ISC.

Die Analyse dieser „Jacobi Symbol Continuation (JC)“ in [**Per01**] spricht von einer Verbesserung – gegenüber der ursprünglichen Methode [**Len87**] – um einen Faktor, der etwas größer als $\log(p)$ ist. Vergleicht man dies mit den Aussagen in [**Bre86, Bre99**], die auch von einer Verbesserung in der Größenordnung von $\log(p)$ sprechen, so könnte die JC-Variante das Faktorisieren von Zahlen der Form $n = pq^2$ etwas erleichtern. Während die Analyse in [**ET02a**, Section 4] der JC-Variante asymptotische Vorteile gegenüber der ISC-Variante bescheinigt, so zeigen die Experimente in [**ET02a, ET02b**] jedoch, dass die JC-Variante in der Praxis – zumindest solange die Faktoren p, q gleich groß und in der Größenordnung von 10^{20} sind – langsamer ist, als die verbesserte Standard-Fortsetzung ISC. Es ist eine offene Frage, ab welcher Größenordnung von p die JC-Variante besser ist, als die verschiedenen Standard-Fortsetzungen.

Um die Laufzeit von ECM abschätzen zu können, wurde in [**Len01**] die Funktion

$$E_{[\text{Len01}]}[n,p] = \log_2(n)^2 L_p\left[\frac{1}{2}, \sqrt{2}\right] \tag{3.1.14}$$

verwendet.

Bei allen bisherigen Analysen der Elliptic Curve Method wird unterstellt, dass die Gruppenordnungen zufällig gewählter Elliptischer Kurven „genau so glatt“ sind, wie zufällig gewählte Zahlen der entsprechenden Größenordnung. Betrachtet man die Analysen in [**McK99**], so dürfte sich ECM in der Praxis noch etwas besser verhalten, als es die Laufzeitabschätzungen unter dieser Annahme angeben.

Um diesen Effekt, und die kaum präzise vorhersagbaren Verbesserungen durch die verschiedenen ECM-Fortsetzungen, berücksichtigen zu können, verwenden wir nicht nur die obige Laufzeitfunktion $E_{[\text{Len01}]}[n,p]$ aus (3.1.14) für unsere Abschätzungen, sondern auch die Laufzeitfunktion

$$E[n,p,\tau(p)] = \log_2(n)^{\log_2(3)} L_p\left[\frac{1}{2}, \sqrt{\tau(p)}\right] \tag{3.1.15}$$

für einige Werte $\tau(p)$ mit $1.5 \leq \tau(p) \leq 2$.

Der Vergleichbarkeit halber richten wir unsere Abschätzungen an den in Tabelle 4 ermittelten Aufwänden aus. Als Basis für die Extrapolation dient – wie auch in [**LV01**, Section 5.9] und [**Len01**] – die Beobachtung, dass die Elliptic Curve Method nach einem Rechenaufwand von etwa 6200 MJ mit Wahrscheinlichkeit 0.63 einen 167-Bit-Faktor einer 768-Bit-Zahl findet, sofern ein solcher existiert.

Rechenaufwand in MJ	Nötige Bitlänge für Δp^2							
	GNFS	ECM						
Laufzeitfunktion	(3.1.11)	(3.1.14)	(3.1.15)					
mit $\tau(p) =$	/	/	2.0	1.9	1.8	1.7	1.6	1.5
$8.00 \cdot 10^3$	512	526	523	523	524	524	525	526
$8.15 \cdot 10^5$	640	639	637	641	644	649	653	658
$4.91 \cdot 10^7$	768	748	748	754	761	769	777	787
$2.00 \cdot 10^9$	896	853	854	864	875	886	899	912
$5.99 \cdot 10^{10}$	1024	955	958	971	985	1000	1017	1035
$5.97 \cdot 10^{15}$	1536	1342	1350	1376	1404	1434	1468	1505
$6.98 \cdot 10^{19}$	2048	1701	1715	1753	1795	1841	1891	1947
$2.64 \cdot 10^{26}$	3072	2366	2389	2451	2520	2596	2679	2772
$5.87 \cdot 10^{31}$	4096	2981	3011	3097	3192	3296	3412	3541
Schnittpunkt mit GNFS	/	634	617	647	696	780	934	1208

TABELLE 5. Nötige Bitlängen für $n = \Delta p^2$

Unterstellt man, dass sich die Laufzeit einer tatsächlichen Implementierung der Elliptic Curve Method in der für uns interessanten Größenordnung ($|\Delta| p^2 >> 10^{150}$) proportional zu (3.1.15) mit $1.7 \leq \tau(p) < 2.0$ verhält[6], so muss diese Methode bereits bei 896-Bit-Zahlen nicht mehr gesondert berücksichtigt werden, da das Zahlkörpersieb bei Zahlen dieser Größenordnung bereits deutlich schneller ist – der Schnittpunkt zwischen den Laufzeitfunktionen für GNFS und ECM darf im Intervall von 2^{600} bis 2^{800} vermutet werden. Dies deckt sich auch mit den Abschätzungen aus [**Sil00**].

Gitter-Methode für $n = p^r q$ mit großem r

In [**BDHG99**] wurde ein Verfahren zur Faktorisierung von $n = p^r q$ vorgeschlagen, das bei großen Exponenten r besonders effizient ist. Das Verfahren basiert auf den Arbeiten [**Cop97, HG97**], wonach eine Zahl $n = pq$ (mit p und q in der gleichen Größenordnung) faktorisiert werden kann, sofern die Hälfte der höherwertigen Bits eines Faktors bekannt ist.

Ist $r = (\log p)^\varepsilon$, dann hat das Verfahren eine asymptotische Laufzeit von

$$T_\varepsilon(p) = \gamma \cdot e^{(\log(p))^{1-\varepsilon}},$$

wobei der Faktor γ durch ein Polynom in $\log(n)$ begrenzt ist.

Ist $r \approx \log(p)$, so besitzt die Methode polynomielle Zeit (in $\log n$). Bei $r \approx \sqrt{\log(p)}$ ist $\varepsilon \approx \frac{1}{2}$ und die Methode ist asymptotisch etwas besser als ECM. Bei konstantem r, wie z.B. $r = 2$, besitzt die Methode exponentielle Laufzeit und stellt somit keine Gefahr für unsere Kryptosysteme dar. Das gilt auch für die in [**UK01**] vorgeschlagene Variante, deren praktische Laufzeit um einen konstanten Faktor besser ist.

[6] Die heuristische Analyse der Improved Standard Continuation in [**Bre99**, Section 4.4] gibt $\tau(p) = 1.677$ für $p \approx 10^{20}$ und $\tau(p) = 1.723$ für $p \approx 10^{60}$ an.

Faktorisierungsmethode für $n = p^r q$ mit großem und glattem r

In [**CUS02**] wurde kürzlich eine neuartige Faktorisierungsmethode vorgestellt, bei der eine Zahl $n = p^r q$ durch Versuchsdivision mit möglichen Kandidaten für q faktorisiert wird.

Entscheidend für die Effizienz dieser Methode ist offensichtlich, wieviele dieser Versuchsdivisionen im schlimmsten Fall durchgeführt werden müssen. Dies hängt wiederum davon ab, wie die Kandidaten für q in diesem Verfahren erzeugt werden.

Die in [**CUS02**] vorgeschlagenen Vorgehensweise sieht folgende Schritte vor:

1. Wähle Paare $(l_i^{e_i}, k_i)$ mit Primzahlpotenzen $l_i^{e_i}$ und ganzen Zahlen k_i, so dass
$$\lambda(l_i^{e_i}) \mid r k_i, \tag{3.1.16}$$
wobei $\lambda()$ die Carmichael-Funktion ist, und
$$\prod_i l_i^{e_i} > q.$$
2. Ermittle für alle i die Lösungen der Gleichung
$$x^{k_i} \equiv n^{k_i} \pmod{l_i^{e_i}}. \tag{3.1.17}$$
Da
$$n^{k_i} = (p^r q)^{k_i} = p^{r k_i} q^{k_i} \overset{(3.1.16)}{\equiv} q^{k_i} \pmod{l_i^{e_i}}$$
erkennt man, dass eine der Lösungen der Gleichung (3.1.16) kongruent zu q modulo $l_i^{e_i}$ ist. Leider ist nicht erkennbar, welche der ermittelten Lösungen dies ist.
3. Setze die im vorherigen Schritt ermittelten Lösungen mit dem Chinesischen Restsatz zu den gesuchten Kandidaten für q zusammen.

Die $O(k_i)$ vielen Lösungen aus (3.1.17) werden im letzten Schritt durch den Chinesischen Restsatz zu $O(\prod_i k_i)$ vielen Kandidaten für q zusammengesetzt.

Ist nun r groß und glatt, so dass $k_i = 1$ für alle i, dann muss schließlich nur ein einziger Kandidat für q „ausprobiert“ werden und das Verfahren arbeitet in Polynomialzeit in $\log(n)$.

In unserem Fall ist aber $r = 2$ und deshalb $k_i = \frac{\lambda(l_i^{e_i})}{2}$, weshalb für die Versuchsdivision mit den $O\left(\prod_i \frac{\lambda(l_i^{e_i})}{2}\right)$ Kandidaten für q ein in $\log(n)$ exponentieller Aufwand entsteht. Deshalb stellt diese Faktorisierungsmethode für unsere Kryptosysteme keine Gefahr dar.

$p \pm 1$-Methoden

In [**Pol74**] bzw. [**Wil82**] wurden Faktorisierungsalgorithmen vorgeschlagen, die die Faktorisierung von Zahlen $n = \prod p_i^{e_i}$ ermöglichen, wenn $p_i - 1$ bzw. $p_i + 1$ glatt ist. Um die Faktorisierung von Δp^2 mit diesen Algorithmen grundsätzlich zu verhindern, forderten wir in Abschnitt 3.1.2 die Konstruktion „starker Primzahlen“ p und $d = -\Delta$.

$p - 1$-Analog in $Cl(\Delta p^2)$

In [**SL84**] wurde ein zu Pollard's $p-1$-Algorithmus [**Pol74**] analoges Verfahren in imaginärquadratischen Klassengruppen vorgeschlagen. Dieses Verfahren faktorisiert eine Zahl $\Delta < 0$, wenn ein Element $\mathfrak{g} \in Cl(\Delta)$ gefunden werden kann, dessen Ordnung glatt ist.

Ist $h(\Delta)$ selbst glatt, so führt ein beliebiges Element $\mathfrak{g} \in Cl(\Delta)$ zum Erfolg. In unserem Fall ist aber nach Satz 1.3.10

$$h(\Delta p^2) = h(\Delta)\left(p - \left(\frac{\Delta}{p}\right)\right) = h(\Delta)(p-1)$$

und $p-1$ besitzt, da p eine starke Primzahl ist, einen großen Primfaktor r.

Nun stellt sich die Frage, wie groß die Wahrscheinlichkeit ist, dass ein zufällig gewähltes Element $\mathfrak{g} \in Cl(\Delta p^2)$ unter diesen Voraussetzungen eine glatte Ordnung besitzt.

Ist $r \mid h(\Delta p^2)$ ein Teiler der Gruppenordnung, dann ist nach [**BBHM00**, Theorem 4] die Wahrscheinlichkeit, dass die Ordnung eines zufällig gewählten Elementes von r geteilt wird, gleich $1-\frac{1}{r}$. Da p eine starke Primzahl ist, ist $r \approx p$ und die Wahrscheinlichkeit mit diesem Ansatz erfolgreich zu sein, ist sicherlich vernachlässigbar. Hier sei angemerkt, dass der Faktor r der Gruppenordnung auch bei Verwendung von $\Delta' = k\Delta p^2$ für beliebige $k \in \mathbb{Z}$ erhalten bleibt.

3.1.4.2. Algorithmen zur Berechnung Diskreter Logarithmen in $Cl(\Delta p^2)$

In diesem Abschnitt stützen wir uns auf [**Ham02**, Abschnitt 5.2] und betrachten die verschiedenen Ansätze zur Berechnung Diskreter Logarithmen in $Cl(\Delta p^2)$. Wie in (3.1.4) ersichtlich, können alle Berechnungsprobleme, die den in dieser Arbeit vorgestellten Kryptosystemen zu Grunde liegen, auf die Berechnung Diskreter Logarithmen in $Cl(\Delta p^2)$ zurückgeführt werden. Deshalb verdienen die verschiedenen Ansätze zur Lösung des DL-Problemes in $Cl(\Delta p^2)$ besondere Beachtung.

Reduktion auf DL-Berechnungen in $Cl(\Delta)$ und $\mathbb{F}_p^*$

Wie in Abschnitt 2.3.1.1 und Abschnitt 2.3.2 erläutert, kann die Berechnung Diskreter Logarithmen in $Cl(\Delta p^2)$ auf DL-Probleme in $Cl(\Delta)$ und $\mathbb{F}_p^*$ reduziert werden, sofern der Führer p bekannt ist. Da wir die Diskriminante Δp^2 so wählen, dass ein Angreifer diese nicht faktorisieren kann, kann er diese Problemreduktion nicht anwenden.

Index-Berechnungs-Algorithmen

In [**Vol00**] wurde unter Annahme der Riemannschen Vermutung bewiesen, dass die Berechnung Diskreter Logarithmen in beliebigen imaginärquadratischen Klassengruppen $Cl(\Delta)$ mit $L_{|\Delta|}\left[\frac{1}{2}, \frac{3}{4}\sqrt{2} + o(1)\right]$ Bit-Operationen möglich ist. In der Praxis verwendet man das MPQS-Analog [**Jac99, Jac00**], für das man eine Laufzeit von etwa $L_{|\Delta|}\left[\frac{1}{2}, 1 + o(1)\right]$ erwarten darf. Wie die praktischen Experimente und Abschätzungen in [**Ham02**, Abschnitt 5.2.2] belegen, ist dieses Verfahren nicht nur asymptotisch, sondern für die relevante Größenordnung auch praktisch viel weniger leistungsfähig, als das Zahlkörpersieb. Außerdem lassen es die in [**HB03**] vorgebrachten Argumente unwahrscheinlich erscheinen, dass das Zahlkörpersieb jemals zur effizienten Berechnung diskreter Logarithmen in $Cl(\Delta)$ eingesetzt werden kann.

Die Bitlängen für $|\Delta|$ in folgender Tabelle sind [**Ham02**, Tabelle 5.6] entnommen.

Wie sich anhand dieser Abschätzungen erkennen lässt, können mit dem gleichen Rechenaufwand, verglichen mit dem Faktorisieren mit dem Zahlkörpersieb, nur Diskrete Logarithmen bei sehr viel kleineren Zahlen berechnet werden. Deshalb muss diese Möglichkeit unsere Kryptosysteme zu attackieren unter praktischen Gesichtspunkten nicht näher betrachtet werden.

Rechenaufwand in MJ	Nötige Bitlänge für Δp^2	
	GNFS	MPQS-Analog [**Jac99**]
$4.91 \cdot 10^7$	768	540
$5.99 \cdot 10^{10}$	1024	687
$5.97 \cdot 10^{15}$	1536	958
$6.98 \cdot 10^{19}$	2048	1208
$2.64 \cdot 10^{26}$	3072	1665
$5.87 \cdot 10^{31}$	4096	2084

TABELLE 6. Nötige Bitlängen beim MPQS-Analog

Generische Algorithmen

Wie in [**Ham02**, Abschnitt 5.2.3] angedeutet, könnten zur Berechnung Diskreter Logarithmen in $Cl(\Delta p^2)$ auch generische, d.h. in beliebigen endlichen, abelschen Gruppen G anwendbare, Algorithmen, wie Shank's Baby-Step-Giant-Step-Verfahren [**Sha72**] oder Pollard's ρ- und λ-Algorithmen [**Pol78, Tes98**], eingesetzt werden. Diese Verfahren benötigen $O(\sqrt{|G|})$ Gruppenoperationen, haben demnach exponentielle Laufzeit (in der Eingabelänge $\log(|\Delta|p^2)$) und müssen bei den für uns interessanten Größenordnungen mit mindestens 512 Bit Diskriminanten sicherlich nicht weiter berücksichtigt werden.

Pohlig-Hellman Algorithmus

So ähnlich verhält es sich auch mit dem Pohlig-Hellman-Algorithmus [**PH78**]. Durch dieses Verfahren kann die DL-Berechnung in $Cl(\Delta p^2)$ auf Untergruppen von Primzahlpotenz-Ordnung reduziert werden, sofern die Faktorisierung der Klassenzahl $h(\Delta p^2)$ bekannt ist.

Um die Klassenzahl zu berechnen, würde man im Allgemeinen das MPQS-Analog [**Jac99**] verwenden; da mit dem GNFS effizienter faktorisiert werden kann, wird man in unserem Fall vorerst die Diskriminante Δp^2 faktorisieren und dann die Klassenzahl $h(\Delta)$, und dann schließlich $h(\Delta p^2) = h(\Delta)(p-1)$, berechnen. Insgesamt ist dieser Algorithmus aber nur dann effizient, wenn $h(\Delta p^2)$ glatt ist. Wie bereits bei der Diskussion des $p-1$-Analoges in $Cl(\Delta p^2)$ [**SL84**] festgestellt wurde, ist $h(\Delta p^2)$ nicht glatt – da wir p als „starke Primzahl" gewählt haben, ist ein großer Faktor $r \mid (p-1)$ garantiert. Außerdem ist die Wahrscheinlichkeit, dass ein zufällig gewähltes Element dennoch eine glatte Ordnung hat, durch $\frac{1}{r} \approx \frac{1}{p}$ beschränkt und damit vernachlässigbar.

Deshalb stellt der Pohlig-Hellman Algorithmus in unserem Fall keine Gefahr dar und muss bei der Bestimmung der Parameterlängen nicht weiter berücksichtigt werden.

3.1.4.3. Bestimmung der nötigen Größe für Δ und p

Nachdem oben die verschiedenen Algorithmen zur Faktorisierung von Δp^2 und zur Berechnung Diskreter Logarithmen in $Cl(\Delta p^2)$ betrachtet wurden, wollen wir die wesentlichen Aspekte hier zusammentragen und die notwendigen Parameterlängen für Δ und p bestimmen.

Wie in (3.1.4) ersichtlich, kann ein Angreifer, der Diskrete Logarithmen in $Cl(\Delta p^2)$ berechnen kann, alle Berechnungsprobleme, die den hier vorgestellten Kryptosystemen zu Grunde liegen, lösen.

Wie eben erläutert, müssen die generischen Algorithmen [**Sha72, Pol78, Tes98**] und der Ansatz von Pohlig und Hellman [**PH78**] nicht näher betrachtet werden – bei großen, wie in Abschnitt 3.1.2 gewählten, Diskriminanten Δp^2 ist das MPQS-Analog [**Jac99**] der effizienteste Algorithmus zur Berechnung Diskreter Logarithmen. Da aber auch dieser Algorithmus – wie aus Tabelle 6 ersichtlich – bei vergleichbaren Parametern weniger effizient als das Zahlkörpersieb [**BLP93**] ist, wird ein praktischer Angreifer zuerst die Diskriminante Δp^2 faktorisieren und dann die in Abschnitt 2.3 vorgestellten Problemreduktionen anwenden. Deshalb genügt es, die Parameter so zu dimensionieren, dass das Faktorisieren von Δp^2 nicht möglich ist.

Da $|\Delta|$ und p als starke Primzahlen gewählt wurden, sind die $p \pm 1$-artigen Verfahren [**Pol74, Wil82, SL84**] nicht erfolgsversprechend. Auch die Gitter-Methode aus [**BDHG99**] und das in [**CUS02**] vorgeschlagene Verfahren sind in unserem Fall nicht effizient – es müssen lediglich das Zahlkörpersieb und die verschiedenen ECM-Varianten berücksichtigt werden.

Als untere Schranke für die praktische Sicherheit unserer Verfahren fordern wir, dass das Brechen unserer Verfahren in der Praxis so schwer sein soll, wie das Faktorisieren einer 896-Bit RSA-Zahl $n = pq$ mit dem Zahlkörpersieb.

Wie in Tabelle 5 ersichtlich, wird hier vermutlich ein Aufwand von $2 \cdot 10^9$ MIPS-Jahren benötigt werden. Betrachtet man in Tabelle 5 die nötigen Bitlängen zum Schutz vor Angriffen mit der Elliptic Curve Method, so zeigt sich (selbst bei Unterstellung der für diese Größenordnung sehr optimistischen Laufzeitfunktion (3.1.15) mit $\tau(p) = 1.7$ für ECM), dass das Zahlkörpersieb bei Zahlen Δp^2 mit mindestens 896 Bit effizienter ist, als die Elliptic Curve Method. Deshalb muss bei Zahlen dieser Größenordnung auch die Elliptic Curve Method nicht mehr berücksichtigt werden.

Damit unsere Verfahren in der Praxis vermutlich genau so sicher sind, wie das RSA-Verfahren mit x-Bit, $x \geq 896$, müssen Δ und p wie in Abschnitt 3.1.2 mit einer Bitlänge von $\lceil x/3 \rceil$ gewählt werden.

3.2. ElGamal-Verschlüsselung

In diesem Abschnitt stellen wir verschiedene Varianten des ElGamal-Verschlüsselungsanaloges in der Klassengruppe $Cl(\Delta p^2)$ einer imaginärquadratischen Nichtmaximalordnung $\mathcal{O}_{\Delta p^2}$ vor.

Die Verwendung von Nichtmaximalordnungen zur Konstruktion von DL-basierten Kryptosystemen mit zusätzlicher Falltürinformation wurde erstmals in [**HJPT98**] vorgeschlagen. Der Vorteil dieses Ansatzes gegenüber der konventionellen Konstruktion besteht in der Möglichkeit, die Entschlüsselung (großteils) in der geheimen Klassengruppe $Cl(\Delta)$ anstatt in der öffentlichen Klassengruppe $Cl(\Delta p^2)$ durchzuführen. Da die Koeffizienten der Gruppenelemente in $Cl(\Delta)$ viel kleiner sind, führt das zu einer etwa fünffach beschleunigten Entschlüsselungsroutine.

Während das in [**HJPT98**] und in Abschnitt 3.2.1 näher erläuterte NIQ-ElGamal-HJPT-Verfahren OW-CPA bezüglich $\mathrm{CDHP}_{Cl(\Delta p^2)}$ ist, so wurde in [**JJ00**] gezeigt, dass ein CCA-Angreifer das Verfahren mit großer Wahrscheinlichkeit brechen kann.

Deshalb werden in den Abschnitten 3.2.2 und 3.2.3 IND-CCA2-Varianten vorgestellt, die sich der generischen Tranformationen aus [**FO99, OP01**] bedienen.

In Abschnitt 3.2.4 wird gezeigt, wie die in Abschnitt 2.3.1.1 vorgestellte Reduktion des DL-Problemes zur Konstruktion identitätsbasierter NIQ-ElGamal-Varianten im Stile von [**MY91**] verwendet werden kann.

In Abschnitt 3.2.5 werden die vorgestellten ElGamal-Varianten schließlich zusammenfassend diskutiert.

3.2.1. NIQ-ElGamal-HJPT-Variante

Dieser Abschnitt ist der Diskussion des in [**HJPT98**] vorgestellten ElGamal-Analoges in Klassengruppen imaginärquadratischer Ordnungen gewidmet.

Nach der Vorstellung des ursprünglich vorgeschlagenen Verfahrens in Abschnitt 3.2.1.1, werden in Abschnitt 3.2.1.2 verschiedene Ansätze zur Einbettung von Nachrichten in reduzierte $\mathcal{O}_{\Delta p^2}$-Ideale diskutiert. In den Abschnitten 3.2.1.3 und 3.2.1.4 finden sich Sicherheits- bzw. Effizienzbetrachtungen zu dem Verfahren aus [**HJPT98**].

Da sich bei der Diskussion der Ansätze zur Nachrichteneinbettung in Abschnitt 3.2.1.2 zeigen wird, dass die elegante und beweisbar funktionierende Nachrichteneinbettung ein noch nicht befriedigend gelöstes Problem darstellt, stellen wir in Abschnitt 3.2.1.5 auch eine Variante des Verfahrens vor, bei der auf die Nachrichteneinbettung verzichtet werden kann.

3.2.1.1. Vorstellung des ursprünglich vorgeschlagenen Verfahrens

Bei der folgenden Darstellung des Verfahrens, stützen wir uns auf Algorithmen aus Kapitel 2. Insbesondere werden wir GoToNonMaxOrder (Algorithmus 2) und GoToMaxCl (Algorithmus 4) benötigen.

Wir gehen davon aus, dass eine Routine Embed zur Nachrichteneinbettung existiert, die einen Bitstring $m = \{0,1\}^{l_m}$ auf ein $\mathfrak{m} \in \mathcal{SI}(\Delta p^2)$ abbildet und eine inverse Routine DeEmbed, die diese Abbildung rückgängig macht. Wie sich bei der Diskussion verschiedener Ansätze zur Implementierung von Embed in Abschnitt 3.2.1.2 zeigen wird, ist das scheinbar harmlose Problem der Nachrichteneinbettung noch nicht befriedigend gelöst. Deshalb findet sich in Abschnitt 3.2.1.5 auch eine Variante, bei der auf die Nachrichteneinbettung verzichtet werden kann.

Im Folgenden bezeichnet $r \in_r R$ die zufällige und gleichverteilte Auswahl eines Elementes r aus einer Menge R.

Beweis der Korrektheit von Kryptosystem 1. Um die Korrektheit des Verfahrens zu zeigen, erinnern wir zuerst an Proposition 1.3.5, wonach die Abbildung $\phi_{Cl}^{-1} : Cl(\Delta p^2) \to Cl(\Delta)$, die mittels GoToMaxCl realisiert wird, ein surjektiver Homomorphismus ist. Deshalb impliziert $\mathfrak{a} \sim \mathfrak{b}$ auch $\phi_{Cl}^{-1}(\mathfrak{a}) \sim \phi_{Cl}^{-1}(\mathfrak{b})$. Somit gilt

$$\begin{aligned} \mathfrak{C}_2 &= \phi_{Cl}^{-1}\left(\mathfrak{m}\mathfrak{b}^r\right) \\ &= \phi_{Cl}^{-1}(\mathfrak{m})\phi_{Cl}^{-1}\left(\mathfrak{g}^{rb}\right), \end{aligned}$$

$$\begin{aligned} \mathfrak{C}_1^b &= \phi_{Cl}^{-1}\left(\mathfrak{g}^r\right)^b \\ &= \phi_{Cl}^{-1}\left(\mathfrak{g}^{rb}\right) \end{aligned}$$

Kryptosystem 1 NIQ-ElGamal-HJPT

Systemparameter: Eine imaginärquadratische Nichtmaximalordnung $\mathcal{O}_{\Delta p^2}$, wie in Abschnitt 3.1.2 erläutert, die Bitlänge l der verwendeten Exponenten und ein $\mathfrak{g} \in_r Cl(\Delta p^2)$.

Geheime Schlüssel: Bob's geheimer Schlüssel ist $b \in_r \left[2^{l-1}, 2^l\right[$ und der Führer p.

Öffentliche Schlüssel: Bob's öffentlicher Schlüssel ist $\mathfrak{b} = \mathfrak{g}^b$.

Verschlüsselung: Bob schickt Alice seinen öffentlichen Schlüssel $\mathfrak{b}$. Alice wählt eine Zufallszahl $r \in_r \left[2^{l-1}, 2^l\right[$ und berechnet
$\mathfrak{c}_1 \leftarrow \mathfrak{g}^r$,
$\mathfrak{m} \leftarrow \mathsf{Embed}(m)$,
$\mathfrak{c}_2 \leftarrow \mathfrak{m}\mathfrak{b}^r$.
Der Schlüsseltext ist das Paar $(\mathfrak{c}_1, \mathfrak{c}_2)$.

Entschlüsselung: Um die Nachricht wieder zu entschlüsseln berechnet Bob
$\mathfrak{C}_1 \leftarrow \mathsf{GoToMaxCl}(\mathfrak{c}_1, p)$,
$\mathfrak{C}_2 \leftarrow \mathsf{GoToMaxCl}(\mathfrak{c}_2, p)$,
$\mathfrak{M} \leftarrow \mathfrak{C}_2 \left(\mathfrak{C}_1^b\right)^{-1}$,
$\mathfrak{m} \leftarrow \mathsf{GoToNonMaxOrder}(\mathfrak{M}, p)$,
$m \leftarrow \mathsf{DeEmbed}(\mathfrak{m})$

und

$$\begin{aligned} \mathfrak{M} &= \mathfrak{C}_2 \left(\mathfrak{C}_1^b\right)^{-1} \\ &= \phi_{Cl}^{-1}(\mathfrak{m})\phi_{Cl}^{-1}\left(\mathfrak{g}^{rb}\right)\left(\phi_{Cl}^{-1}\left(\mathfrak{g}^{rb}\right)\right)^{-1} \\ &= \phi_{Cl}^{-1}(\mathfrak{m}). \end{aligned}$$

Da nach Voraussetzung $\mathfrak{m} \in \mathcal{SI}(\Delta p^2)$, ist schließlich nach (1.3.2) $\varphi(\phi_{Cl}^{-1}(\mathfrak{m})) = \mathfrak{m}$. □

Bemerkung 3.2.1. Kennt Bob, wie beim identitätsbasierten Verfahren aus Abschnitt 3.2.4 und [**HJW01, HJW03**], lediglich seinen geheimen Schlüssel b, aber nicht den Führer p, so kann die Entschlüsselung folgendermaßen durchgeführt werden:

Entschlüsselung: Um die Nachricht in diesem Fall wieder zu entschlüsseln berechnet Bob einfach
$\mathfrak{m} \leftarrow \mathfrak{c}_2 \left(\mathfrak{c}_1^b\right)^{-1}$,
$m \leftarrow \mathsf{DeEmbed}(\mathfrak{m})$

3.2.1.2. Einbettung von Nachrichten in Ideale mit beschränkter Norm

Sei Δ eine beliebige negative Diskriminante und $\mathcal{O}_\Delta$ die zugehörige imaginärquadratische Ordnung. Dann widmen wir uns in diesem Abschnitt dem Problem der Einbettung einer Nachricht $m = \{0,1\}^{l_m}$ in ein $\mathcal{O}_\Delta$-Ideal $\mathfrak{m}$ mit beschränkter Norm

$$\mathcal{N}(\mathfrak{m}) < B. \tag{3.2.1}$$

Wählt man beispielsweise die Schranke $B = \sqrt{|\Delta|/4}$, so ist das aus der Nachrichteneinbettung resultierende Ideal $\mathfrak{m}$ reduziert und repräsentiert deshalb ein Gruppenelement von $Cl(\Delta)$. Ist $\Delta = \Delta' p^2$, wie in Abschnitt 3.1.2 gewählt, und $B = |\Delta' p^2|^{1/6}\sqrt{1/8}$, dann ist das resultierende Ideal $\mathfrak{m}$ nicht nur reduziert, sondern es gilt sogar $\mathfrak{m} \in \mathcal{SI}(\Delta' p^2)$. Diese Art der Einbettung wird zur Implementierung von NIQ-ElGamal-HJPT benötigt.

Es wird sich zeigen, dass die vorgeschlagene Nachrichteneinbettung – durch die Berechnung von Quadratwurzeln modulo einer Primzahl a – mit einem vergleichsweise großen Rechenaufwand verbunden ist und „beweisbar funktionierende" Einbettungsvarianten eine etwas größere Bandbreite erfordern. Dies liegt vor allem an fehlenden *Beweisen* für die Präsenz von in $\mathbb{Q}(\sqrt{\Delta})$ zerlegten Primzahlen in kleinen, vorgegebenen Intervallen. Die scheinbar harmlose Nachrichteneinbettung darf somit als ein noch nicht völlig befriedigend gelöstes Problem für zukünftige Arbeiten festgehalten werden.

Sei $\mathfrak{m} = (a, b) = a\mathbb{Z} + \frac{b+\sqrt{\Delta}}{2}\mathbb{Z}$, wobei $c = (b^2 - \Delta)/(4a) \in \mathbb{Z}$, die Standarddarstellung eines Ideales mit $\mathcal{N}(\mathfrak{m}) = a$. Dann existieren offensichtlich verschiedene Möglichkeiten, um einen Bitstring $m = \{0,1\}^{l_m}$, den wir auch als positive ganze Zahl interpretieren wollen, in das Ideal $\mathfrak{m} = (a, b)$ einzubetten.

Diese verschiedenen Möglichkeiten sollen im Folgenden näher betrachtet werden.

Einbettung in den b-Koeffizienten

Bettet man die Nachricht m in den Koeffizienten b ein, so dass $b = m$ dann gilt $4ac = m^2 - \Delta$ und die Anforderung bezüglich der beschränkten Norm des Ideales stellt sich als

$$a = \frac{m^2 - \Delta}{4c} < B \tag{3.2.2}$$

dar. Wäre B ausreichend groß, so dass die Ungleichung (3.2.2) automatisch für $c = 1$ erfüllt ist, dann könnten Nachrichten m auf diese Art und Weise leicht in Ideale eingebettet werden.

Ist aber $B = \sqrt{|\Delta|/4}$ oder, wie für die Implementierung von NIQ-ElGamal-HJPT benötigt, noch kleiner, dann müsste $4ac = m^2 - \Delta$ faktorisiert werden, so dass ein möglicherweise existierendes $a < B$ ermittelt werden kann.

Deshalb ist die Einbettung von Nachrichten in den b-Koeffizienten im Allgemeinen wenig praktikabel.

Einbettung in den a-Koeffizienten

Etwas anders stellt sich die Situation dar, wenn die Nachricht m als Teil des a-Koeffizienten aufgefasst wird. In diesem Fall kann a in Abhängigkeit von m als Primzahl mit $\left(\frac{\Delta}{a}\right) = 1$ gewählt werden und der b-Koeffizient des Ideales $\mathfrak{m}$ lässt sich im Wesentlichen durch die Berechnung einer Quadratwurzel aus Δ modulo a ermitteln. Da in diesem Fall die Norm $\mathcal{N}(\mathfrak{m}) = a$ gezielt gewählt wird, kann die Erfüllung der Ungleichung (3.2.1) leicht berücksichtigt werden.

Somit können Nachrichten beispielsweise folgendermaßen eingebettet werden:

Sei

$$l_{\max} \leq \lfloor \log_2 B \rfloor - 1 \tag{3.2.3}$$

und

$$l_{\max} = 1 + l_m + l_a. \tag{3.2.4}$$

Dann könnte eine Nachricht $m = \{0,1\}^{l_m}$ in ein Ideal $\mathfrak{m} = (a, b)$ eingebettet werden, indem die kleinste Primzahl a größer als $\left(2^{l_m+1} + m\right) 2^{l_a}$ gesucht wird, die auch $\left(\frac{\Delta}{a}\right) = 1$ erfüllt.

Betrachtet man die Binärdarstellung

$$a = \overbrace{1\underbrace{\mathrm{mmm}\cdots\mathrm{mmm}}_{l_m \text{ Bits}}\underbrace{000\cdots000}\underbrace{\mathrm{rr}\cdots\mathrm{rr}}_{l_a \text{ Bits}}}^{l_{\max} \text{ Bits}} \tag{3.2.5}$$

von $a < 2^{l_{\max}+1} \leq B$, so ist die eindeutige Rekonstruktion von m bei gegebenem Ideal $\mathfrak{a} = (a, b)$ trivial.

Daraus ergeben sich folgende Algorithmen zur Einbettung einer Nachricht $m = \{0,1\}^{l_m}$ in ein Ideal $\mathfrak{m}$ mit $\mathcal{N}(\mathfrak{m}) < B$ und der Rekonstruktion von m bei gegebenem Ideal $\mathfrak{m}$.

Wir setzen die folgenden Routinen und Operatoren als gegeben voraus:

- Bitstring2Integer
  erhält einen Bitstring $\{0,1\}^*$ als Eingabe, interpretiert diesen als Binärdarstellung einer positiven ganzen Zahl und liefert diese Zahl schließlich zurück.
- Integer2Bitstring
  sorgt für die Umkehrung, indem eine ganze Zahl in einen Bitstring umgewandelt wird.
- Nextprime
  erhält eine positive ganze Zahl n als Eingabe und liefert die kleinste Primzahl $p \geq n$ zurück.
- Prevprime
  erhält eine positive ganze Zahl n als Eingabe und liefert die größte Primzahl $p \leq n$ zurück.
- SqrtFp
  erhält ein Element $a \in \mathbb{F}_p^*$ und die Primzahl p als Eingabe und liefert ein $s \in \mathbb{F}_p^*$, so dass $s^2 \equiv a \pmod{p}$, zurück, bzw. $s \equiv 0$, falls keine Quadratwurzel von $a \in \mathbb{F}_p^*$ existiert.
- DIV
  bezeichnet schließlich die ganzzahlige Division.

Es bleibt schließlich zu klären, wie groß l_a gewählt werden muss, damit diese Einbettung immer möglich ist.

l_a müsste so gewählt werden, dass die Existenz einer Primzahl

$$a \in \left[(2^{l_m+1} + M)2^{l_a}, (2^{l_m+1} + M)2^{l_a} + 2^{l_a} - 1\right] \tag{3.2.6}$$

mit $\left(\frac{\Delta}{a}\right) = 1$ garantiert werden kann.

Lassen wir für einen Moment die Forderung $\left(\frac{\Delta}{a}\right) = 1$ außer Acht, so führt uns dies zur Frage, ob in einem Intervall bestimmter Größe die Existenz einer Primzahl p garantiert werden kann. In Anlehnung an [**Rib95**, Section 4.II.B] bezeichnen wir mit p_n die n-te Primzahl und $d_n = p_{n+1} - p_n$ die „Lücke“ zwischen den beiden aufeinanderfolgenden Primzahlen p_n und p_{n+1}. Die Frage ist nun, wie groß d_n

Algorithmus 16 Embed-v1

Eingabe: Eine Diskriminante Δ, die Parameter $l_m, l_a \in \mathbb{Z}_{>0}$ sowie die Nachricht $m = \{0,1\}^{l_m}$.
Ausgabe: Ein $\mathcal{O}_\Delta$-Ideal $\mathfrak{m} = (a,b)$, so dass $\mathcal{N}(\mathfrak{m}) \leq 2^{l_m+l_a+2}$.

$M \leftarrow \mathsf{Bitstring2Integer}(m)$
$a \leftarrow \left(2^{l_m+1} + M\right) 2^{l_a}$
repeat
  $a \leftarrow \mathsf{Nextprime}(a)$
until $\left(\frac{\Delta}{a}\right) = 1$
$b \leftarrow \mathsf{SqrtFp}(\Delta, a)$
if $b \not\equiv \Delta \pmod 2$ **then**
  $b \leftarrow a - b$
end if
return(a,b)

Algorithmus 17 DeEmbed-v1

Eingabe: Ein $\mathcal{O}_\Delta$-Ideal $\mathfrak{m} = (a,b)$ und die Parameter $l_m, l_a \in \mathbb{Z}_{>0}$.
Ausgabe: Die Nachricht $m = \{0,1\}^{l_m}$.

$M \leftarrow a \ \mathsf{DIV}\ 2^{l_a} - 2^{l_m}$
$m \leftarrow \mathsf{Integer2Bitstring}(M)$
return(m)

im Verhältnis zu p_n sein kann. Ein altes Resultat von Tschebycheff besagt, dass $d_n < p_n$. Praktische Erfahrungen belegen allerdings, dass diese Schranke viel, viel zu konservativ ist. Cramér *vermutete* bereits im Jahre 1937, dass $d_n = O(\log(p_n)^2)$; beweisen konnte er – selbst unter Annahme der Riemannschen Vermutung – allerdings nur, dass $d_n = O\left(p_n^{1/2}\log(p_n)\right)$. Bis heute ist man diesbezüglich nicht signifikant weiter gekommen. Die heute besten Resultate ohne Riemannsche Vermutung stammen von Baker, Harman und Pintz [**BH96, BHP01**]. Dort wurde bewiesen, dass man für ausreichend großes $x > x_0$ eine Primzahl $p \in [x - x^\theta, x]$ finden kann, wobei in [**BH96**] $\theta = 0.535$ und in [**BHP01**] $\theta = 0.525$ gezeigt wurde.

Ein analoges Resultat von Harman, Kumchev und Lewis [**HKL03**, Theorem 1] besagt, dass man in einem gegebenen imaginärquadratischen Körper für ein ausreichend großes $x > x_0$ ein Primideal $\mathfrak{p}$ mit

$$\mathcal{N}(\mathfrak{p}) \in \left[x - x^{0.53}, x\right] \tag{3.2.7}$$

finden kann.

Aus (3.2.6) und (3.2.7) zusammen erhalten wir die Ungleichung

$$\begin{aligned} 2^{l_a} &> \left[\left(2^{l_m+1} + M\right) 2^{l_a}\right]^{0.53} \\ 2^{\frac{l_a}{0.53}} &> \left(2^{l_m+1} + M\right) 2^{l_a} \\ 2^{l_a\left(\frac{1}{0.53}-1\right)} &> 2^{l_m+1} + M, \end{aligned} \tag{3.2.8}$$

deren Erfüllung die Existenz eines geeigneten Primideales impliziert, sofern

$$x_0 < \left(2^{l_m+1} + M\right) 2^{l_a} + 2^{l_a} - 1 < 2^{l_m+2} \tag{3.2.9}$$

erfüllt ist, wobei die Konstante x_0 vom imaginärquadratischen Körper abhängt.

Da $M < 2^{l_m+1}$, ist die Ungleichung (3.2.8) erfüllt, wenn

$$l_a > \frac{l_m+2}{\frac{1}{0.53}-1} = \frac{53}{47}(l_m+2) \approx 1.13(l_m+2) \tag{3.2.10}$$

Sind die Diskriminanten Δp^2 wie in Abschnitt 3.1.2 gewählt, und wird $B = \sqrt{1/8}|\Delta p^2|^{1/6}$ als Schranke verwendet, so dass die Nachrichtenideale $\mathfrak{m} \in \mathcal{SI}(\Delta p^2)$ sind, dann lässt sich aus (3.2.3–3.2.4) und (3.2.10) die maximal zulässige Nachrichtenlänge $\hat{l_m} \geq l_m$ ableiten.

Aus (3.2.4) und (3.2.10) erhalten wir

$$l_{\max} - l_m - 1 > \frac{53}{47}(l_m+2),$$

was äquivalent ist zu

$$l_m < 0.47\, l_{\max} - 1.53\ . \tag{3.2.11}$$

Zusammen mit (3.2.3) und $B = \sqrt{1/8}|\Delta p^2|^{1/6}$ erhalten wir aus (3.2.11) schließlich

$$l_m < 0.47\ \left\lfloor \log_2\left(\sqrt{1/8}|\Delta p^2|^{1/6}\right)\right\rfloor - 2\ . \tag{3.2.12}$$

Betrachtet man Diskriminanten mit verschiedenen Bitlängen, so erhält man die folgende Tabelle.

Bitlänge der Diskriminante $\log_2\lvert\Delta p^2\rvert$	Maximale Nachrichtenlänge $\hat{l_m}$
1024	77
1280	97
1536	117
1792	137
2048	157
2304	177
2560	197
2816	217
3072	237

TABELLE 7. Maximale Nachrichtenlängen

Nach der in [**HKL03**] bewiesenen Schranke könnte man bei Verwendung einer 1024 Bit Diskriminante also maximal 77 Bit lange Nachrichten sicher einbetten – um 128 Bit Nachrichten einbetten zu können, müsste eine Diskriminante mit mehr als 1669 Bit verwendet werden.

Ist ein derart stark eingeschränkter Nachrichtenraum nicht zu tolerieren, so muss eine andere Einbettungsvariante gewählt werden.

Beispielsweise kann die Nachrichteneinbettung durch die Verwendung eines zusätzlichen Hilfswertes erreicht werden. Man wählt eine zufällige Primzahl $a < 2^{l_{\max}} < B$ mit $\left(\frac{\Delta}{a}\right) = 1$, berechnet den b-Koeffizienten wie oben und zusätzlich den Hilfswert $z = M - a$. Aus diesem Hilfswert und dem Ideal $\mathfrak{m} = (a, b)$ kann

später wieder die ganzzahlige Nachricht $M \in \mathbb{Z}$, und daraus schließlich der Bitstring $m = \{0,1\}^{l_m}$, rekonstruiert werden.

Algorithmus 18 Embed-v2

Eingabe: Eine Diskriminante Δ, den Parameter $l_{\max} \leq \lfloor \log_2(B) \rfloor - 1$ und die Nachricht $m = \{0,1\}^{l_m}$.
Ausgabe: Ein $\mathcal{O}_\Delta$-Ideal $\mathfrak{m} = (a,b)$, so dass $\mathcal{N}(\mathfrak{m}) < B$ und einen Hilfswert $z \in \mathbb{Z}_{>0}$.

$M \leftarrow \mathsf{Bitstring2Integer}(m)$
Wähle Zufallszahl $a \in_r [2^{l_{\max}-1}, 2^{l_{\max}}]$
repeat
  $a \leftarrow \mathsf{Prevprime}(a)$
until $\left(\frac{\Delta}{a}\right) = 1$
$b \leftarrow \mathsf{SqrtFp}(\Delta, a)$
if $b \not\equiv \Delta \pmod 2$ **then**
  $b \leftarrow a - b$
end if
$z \leftarrow M - a$
return(a, b, z)

Algorithmus 19 DeEmbed-v2

Eingabe: Ein $\mathcal{O}_\Delta$-Ideal $\mathfrak{m} = (a,b)$ und den Hilfswert $z \in \mathbb{Z}_{>0}$.
Ausgabe: Die Nachricht $m = \{0,1\}^{l_m}$.

$M \leftarrow z + a$
$m \leftarrow \mathsf{Integer2Bitstring}(M)$
return(m)

Betrachtet man die beiden vorgeschlagenen Lösungsansätze, so zeigt sich, dass die effiziente, elegante und beweisbar funktionierende Nachrichteneinbettung kein triviales Problem zu sein scheint und in zukünftigen Arbeiten nähere Beachtung finden sollte.

3.2.1.3. Sicherheitsbetrachtungen

Im Folgenden betrachten wir unterschiedlich mächtige Angreifer gegen das in Abschnitt 3.2.1.1 vorgestellte Verfahren. Wir werden sehen, dass die Sicherheit des Verschlüsselungsverfahrens signifikant von der Mächtigkeit des Angreifers abhängt. Während das Verfahren immun gegen Chosen Plaintext Attacken (CPA) zu sein scheint, so kann ein mächtigerer Angreifer das Verfahren mit einer Chosen Ciphertext Attacke (CCA) brechen.

Passive Angreifer

Auf der einen Seite ist sofort aus der Darstellung der Verschlüsselungsroutine in NIQ-ElGamal-HJPT (Kryptosystem 1) und den Definitionen der Sicherheitsbegriffe in Abschnitt 3.1.1.1 ersichtlich, dass das Verfahren OW-CPA bezüglich des Computational Diffie-Hellman-Problemes in der Gruppe $Cl(\Delta p^2)$ ($\mathrm{CDHP}_{Cl(\Delta p^2)}$)

ist. Wie in [**BW88**] gezeigt wurde, kann das Faktorisierungsproblem für Δp^2 mit probabilistischen Polynomialzeitreduktionen auf das $\mathrm{CDHP}_{Cl(\Delta p^2)}$ zurückgeführt werden. Deshalb darf NIQ-ElGamal-HJPT bei geeigneter Wahl der Sicherheitsparameter (Δ, p, l) als immun gegen Angriffe betrachtet werden, bei denen vom Angreifer lediglich Klartexte gewählt werden können.

Werden Δ und p wie in Abschnitt 3.1.2 gewählt und ist, wie in Abschnitt 3.1.4.3 näher erläutert, $|\Delta p^2| > 2^{896}$, so scheinen die heute bekannten Algorithmen für das Faktorisierungs- oder DL-Problem in $Cl(\Delta p^2)$ nicht erfolgsversprechend.

Nun muss schließlich noch geklärt werden, welche Bitlänge l für die geheimen Exponenten verwendet werden soll.

Zum Schutz gegen generische Algorithmen zur Berechnung Diskreter Logarithmen in $\langle \mathfrak{g} \rangle \subseteq Cl(\Delta p^2)$, wie Shank's Baby-Step-Giant-Step-Verfahren [**Sha72**] oder Pollard's ρ- und λ-Algorithmen [**Pol78, Tes98**], verlangen wir $l \geq 160$. Wie die Abschätzungen aus [**HM00a**, Section 3.3] zeigen, geht bei dieser Wahl keine Gefahr von diesen generischen Algorithmen aus.

Außerdem muss der Angriff von van Oorschot und Wiener [**vOW96**], d.h. eine Kombination des Pollardschen λ-Algorithmus mit der *partiellen* Pohlig-Hellman Reduktion auf die Primfaktoren von $h(\Delta p^2)$, näher betrachtet werden.

Anders als bei der in [**vOW96**] betrachteten Gruppe $\mathbb{F}_p^*$ ist die Gruppenordnung $h(\Delta p^2)$ nicht a priori bekannt, und kann bei den verwendeten Diskriminanten $|\Delta p^2| > 2^{896}$ auch nicht berechnet werden. Es bliebe also lediglich der Versuch, durch zufällige Wahl von Gruppenelementen $\gamma \in Cl(\Delta p^2)$, mit dem $p-1$-Analog in $Cl(\Delta p^2)$ [**SL84**] glatte *Anteile* von $h(\Delta p^2)$ zu finden.

Da p so gewählt wurde, dass $p-1$ einen großen Primteiler p' hat, ist die Wahrscheinlichkeit, ein zufälliges $\gamma \in Cl(\Delta p^2)$ zu finden, dessen Ordnung glatt – also insbesondere kein Vielfaches von p' – ist, nach [**BBHM00**, Theorem 4] verschwindend gering. Deshalb scheint auch dieser Angriff nicht anwendbar und die Verwendung von 160 Bit-Exponenten ausreichende Sicherheit zu gewährleisten.

Aktive Angreifer

Auf der anderen Seite wurde in [**JJ00**] gezeigt, dass ein aktiver Angriff, bei dem der Angreifer den Klartext zu zwei sorgfältig gewählten Schlüsseltexten erhält, zur Faktorisierung von Δp^2 führt. Somit kann die Problemreduktion aus Abschnitt 2.3.1.1 zur Lösung des DL-Problemes eingesetzt werden und wir betrachten das Verfahren als gebrochen.

Um das Nachrichtenideal $\mathfrak{m}$ eindeutig entschlüsseln zu können, wurde die Verwendung von Nachrichten $\mathfrak{m} \in \mathcal{SI}(\Delta p^2)$, so dass $\mathcal{N}(\mathfrak{m}) < \sqrt{|\Delta/4|}$, gefordert. Der aktive Angreifer missachtet bewusst diese Konvention, verschlüsselt zwei Nachrichten $\mathfrak{m}_i$, $i \in \{1, 2\}$, mit

$$\sqrt{|\Delta|/3} < \mathcal{N}(\mathfrak{m}_i) < \sqrt{|\Delta|}, \tag{3.2.13}$$

und erhält vom ahnungslosen Bob die Nachrichten $m_i' = \mathsf{DeEmbed}(\mathfrak{m}_i')$ zurück, die zu den zwei Klartextidealen $\mathfrak{m}_i'$ korrespondieren. Da $\mathcal{N}(\mathfrak{m}_i) > \sqrt{|\Delta|/3}$, ist $\mathfrak{m}_i \neq \mathfrak{m}_i'$. Außerdem liegen – wegen $\mathcal{N}(\mathfrak{m}_i) < \sqrt{|\Delta|}$ – die Ideale $\mathfrak{m}_i$ und $\mathfrak{m}_i'$ nur einen Reduktionschritt auseinander. Deshalb kann, wie in [**JJ00**] gezeigt, aus zwei solchen entschlüsselten Schlüsseltexten – mit grosser Wahrscheinlichkeit – die Faktorisierung von Δp^2 bestimmt werden.

Um diesen Angriff zu verhindern, muss das zuverlässige Erkennen einer „bösartigen Nachricht“ beim Entschlüsseln garantiert werden. Hierzu könnte man dem Schlüsseltext einen kryptografischen Hashwert der Nachricht anhängen, um beim Entschlüsseln die Zusammengehörigkeit von Schlüsseltext und Nachricht zu überprüfen und den Klartext nur im Erfolgsfall zurückliefern. Die Weiterentwicklung dieser naiven Idee führt zu den generischen Transformationen in [**FO99**] und [**OP01**], die wir in den Abschnitten 3.2.2 und 3.2.3 dazu verwenden wollen, um IND-CCA2-Varianten unseres ElGamal-Verschlüsselungsanaloges in $Cl(\Delta p^2)$ zu entwickeln.

3.2.1.4. Effizienzbetrachtungen

Wie oben bemerkt, liegt der Vorteil dieses Ansatzes in der – durch kleinere Koeffizienten in der Maximalordnung hervorgerufenen – beschleunigten Entschlüsselung.

Deshalb geben wir hier die Laufzeitergebnisse einer ersten Implementierung an, für die die LiDIA-Bibliothek [**LiD**] benutzt wurde. Es sollte angemerkt werden, dass keine der implementierten Varianten ausgiebig optimiert ist; Ziel der Implementierung war vielmehr die relativen Unterschiede der verschiedenen Verfahren herauszustellen.

Die Laufzeiten wurden auf einer SPARC-ultra mit 160 MHz und 256 MB RAM ermittelt, wobei der Durchschnittswert (in Sekunden) für fünf verschiedene Nichtmaximalordnungen und 50 zufällig gewählte Nachrichten angegeben ist. Neben der ElGamal-Verschlüsselung, die in beiden Varianten identisch ist, ist die Entschlüsselungslaufzeit von NIQ-ElGamal-HJPT (Kryptosystem 1) mit der konventionellen Variante und schließlich einer RSA-Entschlüsselung – ohne Anwendung des Chinesischen Restsatzes – verglichen worden.

Bitlänge			Laufzeit der Ver- bzw. Entschlüsselung in s			
Δp^2	Δ	p	Ver-	Ent- in $Cl(\Delta p^2)$	Ent- in $Cl(\Delta)$	RSA-Ent-
768	384	192	8.91	4.48	1.45	0.53
832	384	224	11.05	5.60	1.58	0.67
896	384	256	13.65	6.87	1.70	0.84
960	384	288	16.48	8.30	1.83	1.03
1024	384	320	19.24	9.66	1.95	1.13

TABELLE 8. Laufzeiten für ElGamal-Ver- und Entschlüsselung in $Cl(\Delta p^2)$

Diese – aus [**HJPT98**] entnommene – Tabelle zeigt, dass die Entschlüsselung bei dem vorgeschlagenen Verfahren etwa fünfmal so schnell ist, wie bei der konventionellen Variante ohne Ausnutzen der Falltür-Struktur. Während die RSA-Entschlüsselung immer noch schneller ist, so sind die Laufzeiten nun zumindest grob vergleichbar. Es sollte angemerkt werden, dass in [**HJPT98**] $l_p < l_\Delta$ gewählt wurde; bei der in der Praxis empfohlenen Wahl $l_p \approx l_\Delta$ dürfte die Effizienzsteigerung noch etwas höher liegen.

3.2.1.5. Eine NIQ-ElGamal-HJPT-Variante ohne Nachrichteneinbettung

Wie die Diskussion in Abschnitt 3.2.1.2 gezeigt hat, ist die elegante Nachrichteneinbettung ein noch nicht befriedigend gelöstes Problem. Deshalb soll im Folgenden

eine NIQ-ElGamal-HJPT-Variante vorgestellt werden, bei der auf die Einbettung von Nachrichten verzichtet werden kann.

Hierzu benötigen wir eine Hashfunktion $g : Cl(\Delta p^2) \to \{0,1\}^{l_m}$, wobei l_m die Nachrichtenlänge ist.

Kryptosystem 2 NIQ-ElGamal-HJPT-without-embedding

Systemparameter: Eine imaginärquadratische Nichtmaximalordnung $\mathcal{O}_{\Delta p^2}$, wie in Abschnitt 3.1.2 erläutert, die Bitlänge l der verwendeten Exponenten und ein $\mathfrak{g} \in_r Cl(\Delta p^2)$.

Geheime Schlüssel: Bob's geheimer Schlüssel ist $b \in_r \left[2^{l-1}, 2^l\right[$ und der Führer p.

Öffentliche Schlüssel: Bob's öffentlicher Schlüssel ist $\mathfrak{b} = \mathfrak{g}^b$.

Verschlüsselung: Bob schickt Alice seinen öffentlichen Schlüssel $\mathfrak{b}$. Alice wählt ein zufälliges Element $\mathfrak{r} \in_r \mathcal{SI}(\Delta p^2)$ und einen zufälligen Exponenten $r \in_r \left[2^{l-1}, 2^l\right[$ und berechnet
$\mathfrak{c}_1 \leftarrow \mathfrak{g}^r$,
$\mathfrak{c}_2 \leftarrow \mathfrak{r}\mathfrak{b}^r$,
$c \leftarrow m \oplus g(\mathfrak{r})$.
Der Schlüsseltext ist das Tripel $(\mathfrak{c}_1, \mathfrak{c}_2, c)$.

Entschlüsselung: Um die Nachricht wieder zu entschlüsseln berechnet Bob
$\mathfrak{C}_1 \leftarrow$GoToMaxCl$(\mathfrak{c}_1, p)$,
$\mathfrak{C}_2 \leftarrow$GoToMaxCl$(\mathfrak{c}_2, p)$,
$\mathfrak{R} \leftarrow \mathfrak{C}_1 \left(\mathfrak{C}_2^b\right)^{-1}$,
$\hat{\mathfrak{r}} \leftarrow$GoToNonMaxOrder$(\mathfrak{R}, p)$
$m \leftarrow c \oplus g(\hat{\mathfrak{r}})$
return(m).

Statt dem Nachrichtenideal $\mathfrak{m}$ wird in dieser Variante also ein zufälliges Ideal $\mathfrak{r}$ verschlüsselt, dessen Hashwert $g(\mathfrak{r})$ als geheimer Schlüssel für die One-Time-Pad-Verschlüsselung der eigentlichen Nachricht m verwendet wird. Somit kann auf die problematische Nachrichteneinbettung verzichtet werden.

Bemerkung 3.2.2. Außerdem scheint der Angriff aus [**JJ00**] für diese Variante nicht ohne weiteres anwendbar, da lediglich der Wert $g(\mathfrak{r}_i)$, aber nicht der für den Angriff benötigte Wert $\mathfrak{r}_i$ selbst, ermittelbar ist, sofern es sich bei g um eine Einwegfunktion handelt.

Bemerkung 3.2.3. Kennt Bob, wie beim identitätsbasierten Verfahren aus Abschnitt 3.2.4 und [**HJW01, HJW03**], lediglich seinen geheimen Schlüssel b, aber nicht den Führer p, so kann die Entschlüsselung folgendermaßen durchgeführt werden:

Entschlüsselung: Um die Nachricht in diesem Fall wieder zu entschlüsseln berechnet Bob einfach
$\hat{\mathfrak{r}} \leftarrow \mathfrak{c}_2 \left(\mathfrak{c}_1^b\right)^{-1}$,
$m \leftarrow c \oplus g(\hat{\mathfrak{r}})$

3.2.2. NIQ-ElGamal-FO-Variante

In diesem Abschnitt soll die generische Transformation aus [**FO99**] verwendet werden, um das oben diskutierte NIQ-ElGamal-HJPT-Verfahren so zu modifizieren, dass es nicht mehr nur die Sicherheitseigenschaft OW-CPA, sondern fortan IND-CCA2 erfüllt und somit immun gegen Attacken im Stile von [**JJ00**] ist.

Hierzu benötigen wir zwei Hashfunktionen $g : Cl(\Delta p^2) \to \{0,1\}^{l_m}$ und $h : Cl(\Delta p^2) \times \{0,1\}^{l_m} \to \{0,1\}^{l}$, wobei l_m die Nachrichtenlänge und l die Bitlänge der verwendeten Exponenten ist.

Kryptosystem 3 NIQ-ElGamal-FO

Systemparameter: Eine imaginärquadratische Nichtmaximalordnung $\mathcal{O}_{\Delta p^2}$, wie in Abschnitt 3.1.2 erläutert, die Bitlänge l der verwendeten Exponenten und ein $\mathfrak{g} \in_r Cl(\Delta p^2)$.

Geheime Schlüssel: Bob's geheimer Schlüssel ist $b \in_r \left[2^{l-1}, 2^l\right[$ und der Führer p, sowie die Nullstellen $\rho, \bar{\rho} \in \mathbb{F}_p^*$ von $f(X)$ wie in (2.2.5).

Öffentliche Schlüssel: Bob's öffentlicher Schlüssel ist $\mathfrak{b} = \mathfrak{g}^b$.

Verschlüsselung: Bob schickt Alice seinen öffentlichen Schlüssel $\mathfrak{b}$. Alice wählt ein zufälliges Element $\mathfrak{r} \in_r \mathcal{SI}(\Delta p^2)$ und berechnet
$e \leftarrow h(\mathfrak{r}, m)$,
$\mathfrak{c}_1 \leftarrow \mathfrak{r}\mathfrak{b}^e$,
$\mathfrak{c}_2 \leftarrow \mathfrak{g}^e$,
$c \leftarrow m \oplus g(\mathfrak{r})$. Der Schlüsseltext ist das Tripel $(\mathfrak{c}_1, \mathfrak{c}_2, c)$.

Entschlüsselung: Um die Nachricht wieder zu entschlüsseln berechnet Bob
$\mathfrak{C}_1 \leftarrow$GoToMaxCl$(\mathfrak{c}_1, p)$,
$\mathfrak{C}_2 \leftarrow$GoToMaxCl$(\mathfrak{c}_2, p)$,
$\mathfrak{R} \leftarrow \mathfrak{C}_1 \left(\mathfrak{C}_2^b\right)^{-1}$,
$\hat{\mathfrak{r}} \leftarrow$GoToNonMaxOrder$(\mathfrak{R}, p)$
$\hat{m} \leftarrow c \oplus g(\hat{\mathfrak{r}})$

Integritätscheck: Um zu entscheiden, ob die entschlüsselte Nachricht zurückgeliefert werden darf oder nicht wird überprüft, ob $\mathfrak{c}_1 \stackrel{?}{=} \hat{\mathfrak{r}}\mathfrak{b}^{h(\hat{\mathfrak{r}},\hat{m})}$. Dies geschieht in folgenden Schritten:
$\hat{e} \leftarrow h(\hat{\mathfrak{r}}, \hat{m})$
$\mathfrak{c}_1' \leftarrow$GoToNonMaxOrder$(\mathfrak{C}_1, p)$
$C \leftarrow$ Ker2Fp(Std2Gen$(\mathfrak{c}_1/\mathfrak{c}_1', p), p, \rho, \bar{\rho})$
$\mathfrak{B} \leftarrow$ GoToMaxCl$(\mathfrak{b}, p)$,
$\mathfrak{b}' \leftarrow$ GoToNonMaxOrder$(\mathfrak{B}, p)$,
$B \leftarrow$ Ker2Fp(Std2Gen$(\mathfrak{b}/\mathfrak{b}', p), p, \rho, \bar{\rho})$
if $\mathfrak{C}_1 = \mathfrak{R}\mathfrak{B}^{\hat{e}}$ **and** $C \equiv B^{\hat{e}} \pmod p$ **then**
  $m \leftarrow \hat{m}$
else
  $m \leftarrow -1$
end if

Bemerkung 3.2.4. Kennt Bob, wie beim identitätsbasierten Verfahren aus Abschnitt 3.2.4 und [**HJW01, HJW03**], lediglich seinen geheimen Schlüssel b, aber nicht den Führer p, so kann die Entschlüsselung und der Integritätscheck

folgendermaßen durchgeführt werden:

Entschlüsselung: Um die Nachricht in diesem Fall wieder zu entschlüsseln berechnet Bob einfach
$\mathfrak{r} \leftarrow \mathfrak{c}_2 \left(\mathfrak{c}_1^b\right)^{-1}$,
$m \leftarrow c \oplus g(\hat{\mathfrak{r}})$

Integritätscheck: Um zu entscheiden, ob die entschlüsselte Nachricht zurückgeliefert werden darf wird folgende Überprüfung durchgeführt:
if $\mathfrak{c}_1 = \mathfrak{r}\mathfrak{b}^{h(\hat{\mathfrak{r}},\hat{m})}$ **then**
$m \leftarrow \hat{m}$
else
$m \leftarrow -1$
end if

3.2.2.1. Sicherheitsbetrachtungen

Bezüglich passiver Angreifer weist das Verfahren die gleichen Sicherheitseigenschaften auf, wie das in Abschnitt 3.2.1 diskutierte NIQ-ElGamal-HJPT-Verfahren. Deshalb sei für generelle Sicherheitsaspekte auf Abschnitt 3.2.1.3 verwiesen.

Allerdings erfüllt das hier vorgestellte Verfahren nicht mehr nur OWE-CPA bezüglich des Computational Diffie-Hellman-Problemes in der Gruppe $Cl(\Delta p^2)$ ($\mathrm{CDHP}_{Cl(\Delta p^2)}$), sondern nach [**FO99**, Theorem 14] IND-CCA2 bzgl. $\mathrm{CDHP}_{Cl(\Delta p^2)}$ im Random Oracle Model.

3.2.2.2. Effizienzbetrachtungen

Der wesentliche Unterschied zwischen der naiven Variante NIQ-ElGamal-HJPT und dem hier vorgestellten Kryptosystem 3 (NIQ-ElGamal-FO) liegt in der zusätzlich vorhandenen Integritätsüberprüfung. Durch diese Prüfung können mögliche Manipulationen am Schlüsseltext-Tripel $(\mathfrak{c}_1, \mathfrak{c}_2, c)$ zuverlässig erkannt werden, so dass das Verfahren immun gegen Attacken ist, bei denen der Schlüsseltext vom Angreifer gezielt gewählt wird.

Allerdings hat dieser Sicherheitsgewinn seinen Preis. Beim Integritätscheck aus [**FO99**] muss überprüft werden, ob die Gleichung $\mathfrak{c}_1 = \hat{\mathfrak{r}}\mathfrak{b}^{h(\hat{\mathfrak{r}},\hat{m})}$ erfüllt ist. Bei der Entschlüsselung muss also zusätzlich die „halbe Verschlüsselung“ wiederholt werden. Dies ist umso unglücklicher, da die Verwendung imaginärquadratischer Nichtmaximalordnungen ja auf die Beschleunigung der Entschlüsselung abzielt. Durch die Reduktion der Integritätsberechnungen von $Cl(\Delta p^2)$ auf Berechnungen in $Cl(\Delta)$ und $\mathrm{Ker}(\phi_{Cl}^{-1}) \cong \mathbb{F}_p^*$ kann diese Problematik zwar etwas entschärft, aber nicht gänzlich gelöst werden. Deshalb sind in diesem Kontext Transformationsvarianten, wie beispielsweise die in [**OP01**] vorgestellte „Rapid Enhanced-security Asymmetric Cryptosystem Transform“ (REACT), von Vorteil, durch die ähnliche Sicherheitseigenschaften bei geringerem Aufwand realisiert werden können.

3.2.3. NIQ-ElGamal-REACT-Variante

In diesem Abschnitt soll die generische Transformation aus [**OP01**] verwendet werden, um eine weitere IND-CCA2-Variante von NIQ-ElGamal-HJPT zu konstruieren.

Hierzu benötigen wir wieder zwei Hashfunktionen $g : Cl(\Delta p^2) \to \{0,1\}^{l_m}$ und $h : Cl(\Delta p^2) \times Cl(\Delta p^2) \times Cl(\Delta p^2) \times \{0,1\}^{l_m} \times \{0,1\}^{l_m} \to \{0,1\}^{l}$, wobei l_m die Nachrichtenlänge und l die Bitlänge der verwendeten Exponenten ist.

Kryptosystem 4 NIQ-ElGamal-REACT

Systemparameter: Eine imaginärquadratische Nichtmaximalordnung $\mathcal{O}_{\Delta p^2}$, wie in Abschnitt 3.1.2 erläutert, die Bitlänge l der verwendeten Exponenten und ein $\mathfrak{g} \in_r Cl(\Delta p^2)$.

Geheime Schlüssel: Bob's geheimer Schlüssel ist $b \in_r \left[2^{l-1}, 2^l\right[$ und der Führer p.

Öffentliche Schlüssel: Bob's öffentlicher Schlüssel ist $\mathfrak{b} = \mathfrak{g}^b$.

Verschlüsselung: Bob schickt Alice seinen öffentlichen Schlüssel $\mathfrak{b}$. Alice wählt ein zufälliges Element $\mathfrak{r} \in_r \mathcal{SI}(\Delta p^2)$ und einen zufälligen Exponenten $r \in_r \left[2^{l-1}, 2^l\right[$ und berechnet
$\mathfrak{c}_1 \leftarrow \mathfrak{g}^r$,
$\mathfrak{c}_2 \leftarrow \mathfrak{r}\mathfrak{b}^r$,
$c \leftarrow m \oplus g(\mathfrak{r})$,
$H \leftarrow h(\mathfrak{c}_1, \mathfrak{c}_2, \mathfrak{r}, c, m)$.
Der Schlüsseltext ist das Quadrupel $(\mathfrak{c}_1, \mathfrak{c}_2, c, H)$.

Entschlüsselung: Um die Nachricht wieder zu entschlüsseln berechnet Bob
$\mathfrak{C}_1 \leftarrow$GoToMaxCl$(\mathfrak{c}_1, p)$,
$\mathfrak{C}_2 \leftarrow$GoToMaxCl$(\mathfrak{c}_2, p)$,
$\mathfrak{R} \leftarrow \mathfrak{C}_1 \left(\mathfrak{C}_2^b\right)^{-1}$,
$\hat{\mathfrak{r}} \leftarrow$GoToNonMaxOrder$(\mathfrak{R}, p)$
$\hat{m} \leftarrow c \oplus g(\hat{\mathfrak{r}})$

Integritätscheck: Um zu entscheiden, ob die Nachricht zurückgeliefert werden darf oder nicht wird folgende Prüfung durchgeführt:
if $H = h(\mathfrak{c}_1, \mathfrak{c}_2, \hat{\mathfrak{r}}, c, \hat{m})$ **then**
  $m \leftarrow \hat{m}$
else
  $m \leftarrow -1$
end if
return(m).

Bemerkung 3.2.5. Kennt Bob, wie beim identitätsbasierten Verfahren aus Abschnitt 3.2.4 und [**HJW01, HJW03**], lediglich seinen geheimen Schlüssel b, aber nicht den Führer p, so kann die Entschlüsselung folgendermaßen durchgeführt werden:

Entschlüsselung: Um die Nachricht in diesem Fall wieder zu entschlüsseln berechnet Bob einfach
$\hat{\mathfrak{r}} \leftarrow \mathfrak{c}_2 \left(\mathfrak{c}_1^b\right)^{-1}$,
$m \leftarrow c \oplus g(\hat{\mathfrak{r}})$

Der Integritätscheck ist in diesem Fall identisch zu dem in NIQ-ElGamal-REACT angegebenem.

3.2.3.1. Sicherheitsbetrachtungen

Die generellen Betrachtungen zu den Parameterlängen für dieses Verfahren sind identisch zu den in Abschnitt 3.2.1.3 angestellten.

Im Unterschied zum Originalverfahren aus [**HJPT98**] ist dieses Kryptosystem aber immun gegen Angriffe, bei denen der Schlüsseltext gewählt werden darf.

Wie in [**OP01**, Theorem 13] gezeigt wurde, ist die REACT-Variante des ElGamal-Verschlüsselungsverfahrens im Random Oracle Model IND-CCA2 bezüglich des zugehörigen Gap Diffie-Hellman-Problemes. Das hier vorgestellte NIQ-ElGamal-REACT-Verfahren ist also IND-CCA2 bezüglich $\mathrm{GDHP}_{Cl(\Delta p^2)}$. Somit existiert für dieses Verfahren ein Sicherheitsbeweis, der sich auf das Gap DHP, anstatt auf das Computational DHP, in der Gruppe $Cl(\Delta p^2)$, bezieht.

3.2.3.2. Effizienzbetrachtungen

Vergleicht man das hier vorgestellte NIQ-ElGamal-REACT mit NIQ-ElGamal-HJPT-without-embedding, so ist leicht zu sehen, dass der Unterschied zwischen den beiden Verfahren lediglich darin besteht, dass in der Verschlüsselungs- und Entschlüsselungsroutine der REACT-Variante jeweils zusätzlich ein Hashwert berechnet wird. Dieser Zusatzaufwand ist im Vergleich zu den restlichen Berechnungen vernachlässigbar – die Effizienz von NIQ-ElGamal-REACT ist also vergleichbar mit der des Originalverfahrens aus [**HJPT98**].

3.2.4. Identitätsbasierte NIQ-ElGamal-Verfahren

In diesem Abschnitt wollen wir – wie in [**HJW01, HJW03**] vorgeschlagen – die in Abschnitt 2.3.1.1 diskutierte Reduktion des DL-Problemes in $Cl(\Delta p^2)$, mit $(\Delta/p) = 1$, auf die DL-Berechnung in $Cl(\Delta)$ und $\mathbb{F}_p^*$ zur Konstruktion eines nichtinteraktiven Public-Key-Kryptosystemes anwenden.

Im Vergleich zum bisherigen Ansatz [**MY91, MY96, LL93b**] in $(\mathbb{Z}/n\mathbb{Z})^*$ besitzt das Verfahren in $Cl(\Delta p^2)$ den Vorteil, dass $Cl(\Delta p^2)$ mit großer Wahrscheinlichkeit zyklisch gewählt werden kann und der Aufwand für die Erzeugung individueller Schlüssel durch Einsatz von zwei verschiedenen subexponentiellen Algorithmen zur DL-Berechnung [**Jac99, Web96, Web98, JL03**] vergleichsweise gering ist.

Somit steht neben dem kürzlich von Boneh und Franklin vorgeschlagenen Verfahren auf Basis des Weil-Diffie-Hellman-Problemes [**BF01**] ein zweites vergleichsweise praktikables identitätsbasiertes Verfahren zur Verfügung, das auch dann eingesetzt werden kann, wenn sich das bislang noch wenig erforschte Weil-Diffie-Hellman-Problem als leicht lösbar erweisen sollte.

Bevor wir unseren Ansatz im Detail vorstellen, wollen wir die bisherigen Arbeiten in diesem Bereich kurz zusammenfassen.

3.2.4.1. Verwandte Arbeiten

Ein grundsätzliches Problem aller bislang in dieser Arbeit vorgestellten Public-Key Verfahren ist, dass die Authentizität des öffentlichen Schlüssels durch zusätzliche Maßnahmen sichergestellt werden muss. Üblicherweise wird dieses Problem durch das Ausstellen von Zertifikaten gelöst, die die Verbindung zwischen Bob's öffentlichem Schlüssel $\mathfrak{b}$ und seiner Identität ID_B bestätigen. Dieses Zertifikat $Cert_{\mathfrak{b}} = \{\mathfrak{b}, ID_B, Sig(h(\mathfrak{b}||ID_B), CA)\}$ wird von einer vertrauenswürdigen Zertifizierungsautorität (CA) signiert. Will Alice eine Nachricht für Bob verschlüsseln,

so erhält sie den öffentlichen Schlüssel $\mathfrak{b}$ im Zertifikat $Cert_{\mathfrak{b}}$ von Bob selbst oder einem Verzeichnisdienst. In jedem Fall sind einige Interaktionen nötig, bevor Alice Bob's authentischer öffentlicher Schlüssel zur Verfügung steht.

Identitätsbasierte Kryptosysteme (Shamir)

Deshalb wäre es in diesem Szenario wünschenswert, dass Alice Bob's öffentlichen Schlüssel $\mathfrak{b} = f(ID_B)$ durch eine systemweit bekannte Funktion $f()$ allein aus Bob's Identität ID_B ableiten könnte. Durch diesen identitätsbasierten – erstmals von Shamir [**Sha85**] vorgeschlagenen – Ansatz kann auf das Ausstellen und die Verifikation von Zertifikaten verzichtet werden. In [**Sha85**] wird der öffentliche Schlüssel $\mathfrak{b}$ aus der Identität ID_B und weiteren teilnehmerspezifischen Daten D_B ermittelt. Während diese teilnehmerspezifischen Daten D_B in einem Signatursystem einfach an die Signatur angehängt werden können, so müsste Alice bei einem Verschlüsselungssytem D_B von Bob oder einem Verzeichnisdienst *vor* der Verschlüsselung erhalten; somit böte dieser identitätsbasierte Ansatz [**Sha85**] bei einem Verschlüsselungssystem keine nennenswerten Vorteile gegenüber zertifikatsbasierten Verfahren.

Nicht-interaktive Verschlüsselungsverfahren (Maurer und Yacobi)

Das erste *nicht-interaktive* Public-Key-Verschlüsselungsverfahren, bei dem der öffentliche Schlüssel aus der Identität ID_B allein ableitbar ist, wurde von Maurer und Yacobi [**MY91**] vorgeschlagen. In diesem System wird ein zusammengesetzter Modul n verwendet, dessen Faktorisierung $n = p_1 \cdots p_r$, wobei p_i prim ist, nur der Schlüsselgenerierungsinstanz (KGC) bekannt ist. Der öffentliche Schlüssel $\mathfrak{b}$ wird allein aus Bob's Identität ID_B abgeleitet und der zughörige geheime Schlüssel b wird – nach Wahl eines erzeugenden Elementes $\mathfrak{g}$ – vom KGC durch Lösung des DL-Problemes $b = \log_{\mathfrak{g}} \mathfrak{b}$ ermittelt. Dies ist möglich, da das KGC das DL-Problem modulo der einzelnen Primfaktoren p_i lösen kann. Das Hauptproblem dieses Ansatzes ist die Tatsache, dass $(\mathbb{Z}/n\mathbb{Z})^*$ selbst, bei diesem Setup, *nicht zyklisch* ist und das Bild der Funktion $f()$ eine zyklische Untergruppe von $(\mathbb{Z}/n\mathbb{Z})^*$ sein muss. Wie die Arbeiten [**MY93, MY96, MK99**] belegen, scheint die Konstruktion einer solchen Funktion $f : ID \to \langle \mathfrak{g} \rangle$, $\langle \mathfrak{g} \rangle \subset (\mathbb{Z}/n\mathbb{Z})^*$, die das Faktorisieren von n nicht erleichtert, keine triviale Aufgabe zu sein.

Wir wollen nun kurz die bisherigen Vorschläge für eine solche Funktion f kurz vorstellen:

1. *Quadrierungsmethode:*

   In [**MY91**] wurde vorgeschlagen $f(ID_B) := ID_B^2 \pmod n$ zu verwenden. Allerdings wurde bereits nach kurzer Zeit in [**MY93**] erkannt, dass diese Methode völlig unsicher ist, da ein *einziger Benutzer* mit Wahrscheinlichkeit $1 - 2^{-r+1} \geq 1/2$ einen nicht-trivialen Faktor von $n = p_1 \cdots p_r$ finden kann. Um diese Schwachstelle zu vermeiden, wurde vorgeschlagen, dass das KGC die geheimen Schlüssel mit einem festen Wert $t \in (\mathbb{Z}/\phi(n)\mathbb{Z})^*$ maskiert und alle anderen Berechnungen wie üblich ausführt. Allerdings wurde in [**MK99**] gezeigt, dass dieses Vorgehen mit großer Wahrscheinlichkeit die Faktorisierung von n nicht verhindern kann, wenn *zwei Benutzer* ihre beiden geheimen Schlüssel gegenseitig veröffentlichen. Deshalb sollte diese Methode aus Sicherheitsgründen nicht verwendet werden.

2. *Legendre Symbol Methode:*

   In [**MY91**] wurde diese „alternative, aber weniger praktische Methode ... der Vollständigkeit halber“ vorgeschlagen. Man empfahl, den Modul $n = p_1 p_2$, p_1, p_2 prim, zu verwenden. Ursprünglich schlugen sie vor für $\mathfrak{b} \geq ID_B$ die kleinste Zahl $\mathfrak{b} \geq n$ mit $\left(\frac{\mathfrak{b}}{n}\right) = 1$ als öffentlichen Schlüssel zu verwenden. In [**MY93**] wurde eine einfache Verfeinerung dieses Ansatzes vorgestellt. Es soll $p_1 \equiv 3 \pmod 8$ und $p_2 \equiv 7 \pmod 8$ verwendet werden. Somit ist $\left(\frac{2}{n}\right) = -1$ und die Existenz diskreter Logarithmen kann durch

$$\mathfrak{b} = f(ID_B) = \begin{cases} ID_B & \text{falls} \quad \left(\frac{ID_B}{n}\right) = 1 \\ 2ID_B & \text{falls} \quad \left(\frac{ID_B}{n}\right) = -1 \end{cases}$$

   garantiert werden. In beiden Fällen ist die Methode nur praktikabel[7], wenn $n = p_1 p_2$. Sie schlugen vor $p_1 - 1$ und $p_2 - 1$ „relativ[8] glatt“ zu wählen und die Berechnung diskreter Logarithmen mit dem Pohlig-Hellman Algorithmus [**PH78**] durchzuführen. Es sei angemerkt, dass – im Gegensatz zu subexponentiellen Algorithmen – die Berechnung jedes individuellen Logarithmus hier sehr aufwendig ist. Deshalb ist – wie auch in [**LL93b**] bemerkt – dieser Ansatz für die praktische Anwendung kaum zu gebrauchen. Auf der anderen Seite wäre selbst die Anwendung von subexponentiellen Algorithmen [**Web96, Web98, JL03**] zur DL-Berechnung bei einem Modul der Form $n = p_1 p_2 > 2^{800}$ eine sehr herausfordernde Aufgabe.

Verwendung anormaler elliptischer Kurven (Okamoto und Uchiyama)

Okamoto und Uchiyama [**OU98**] schlugen vor, anstatt der Gruppe $(\mathbb{Z}/n\mathbb{Z})^*$, $n = p_1 p_2$, p_1, p_2 prim, eine „anormale elliptische Kurve“ über dem Ring $\mathbb{Z}/n\mathbb{Z}$ zu verwenden. Dieser Ansatz erschien sehr praktikabel, da diskrete Logarithmen in anormalen Elliptischen Kurven [**SA98, Sem98, Sma99**] in Polynomialzeit berechnet werden können. Leider bemerkten die Autoren noch vor der Veröffentlichung, dass dieses Verfahren komplett gebrochen werden kann, da bei diesem Setup die Faktorisierung von n leicht ermittelt werden kann.

Verwendung von Faktorbasen (Kügler)

In [**Küg98**, Kapitel 3] untersuchte Kügler die Anwendung von Faktorbasen für die Konstruktion eines praktikablen nicht-interaktiven Public-Key-Verfahrens. Während die Schlüsselerzeugung durch das KGC lediglich polynomielle Zeit benötigt, hat dieser Ansatz das große Problem, dass jeder Benutzer die öffentliche Faktorbasis mit mehr als 1 MB speichern oder „im Vorübergehen“ erzeugen muss. Außerdem muss die Anzahl der Elemente in der Faktorbasis mindestens so groß sein, wie die Anzahl der Teilnehmer, damit ein Angriff über die Lösung eines linearen Gleichungssystemes nicht durchgeführt werden kann.

---

[7] In [**MY93**] wurde auch eine Verallgemeinerung für den Fall $n = p_1 \cdots p_r$, $r > 2$, vorgeschlagen. Allerdings ist dieser Ansatz völlig unpraktisch, da Alice und Bob $r - 2$ zusätzliche Exponentiationen durchführen müssen und der übertragene Schlüssel $r - 2$ mal so lang ist.

[8] Da Pollard's $p - 1$ Algorithmus zur Faktorisierung [**Pol74**] die Zahl n effizient faktorisiert, wenn alle Primfaktoren von $p_1 - 1$ oder $p_2 - 1$ glatt sind, sollten die Primteiler $q_i \,|\, p_i - 1$ mit $q \geq 2^{40}$ gewählt werden.

Verwendung der Weil-Paarung (Boneh und Franklin)

Kürzlich wurde von Boneh und Franklin [**BF01**] ein Verfahren auf Basis des Weil Diffie-Hellman Problemes auf einer supersingulären elliptischen Kurve vorgeschlagen.

Sei $p \equiv 2 \pmod 3$ mit $p = 6q - 1$ für p, q prim und $E/\mathbb{F}_p$ die supersinguläre elliptische Kurve $y^2 = x^3 + 1$, $P \in E/\mathbb{F}_p$ ein Punkt mit der Ordnung q auf der Kurve, $\phi(x, y) = (\zeta x, y)$ ein Endomorphismus auf der Gruppe der Punkte auf $E/\mathbb{F}_p$, wobei $1 \neq \zeta \in \mathbb{F}_{p^2}$ eine Lösung der Gleichung $x^3 - 1 \equiv 0 \pmod p$ ist und μ_q die Untergruppe von $\mathbb{F}^*_{p^2}$, die alle Elemente mit Ordnung q enthält.

Dann ist die *Weil-Paarung* eine Abbildung $e : E \times E \to \mu_q$ mit gewissen Eigenschaften auf die hier nicht näher eingegangen werden soll, und das *Computational Weil Diffie-Hellman Problem (CWDHP)* die Berechnung von $W = e(P, \phi(P))^{abc} \in \mathbb{F}_{p^2}$ bei gegebenem Quadrupel (P, aP, bP, cP) für zufällig gewählte $a, b, c \in_r \mathbb{F}^*_q$.

In [**BF01**] wurde nun in einem ersten Schritt ein nicht-interaktives identitätsbasiertes Kryptosystem vorgeschlagen, das die Einwegeigenschaft (OWE) aufweist, sofern das CWDHP schwierig ist. In einem weiteren Schritt wurde dieses grundlegende Verfahren unter Verwendung der generischen Transformation aus [**FO99**] zu einem identitätsbasierten IND-CCA2-Verfahren bzgl. des CWDHP modifziert. Während die Ver- und Entschlüsselung etwa den gleichen Rechenaufwand wie das hier und in [**HJW01**] vorgeschlagene Verfahren hat, so besticht das Kryptosystem von Boneh und Franklin [**BF01**] durch den äußerst geringen Rechenaufwand für die Schlüsselgenerierungsinstanz. Die Berechnung des teilnehmerspezifischen geheimen Schlüssels geschieht durch eine gewöhnliche Punktmultiplikation auf der elliptischen Kurve E/F_p und deshalb insbesondere in *polynomieller Laufzeit* in $\log(p)$.

Auf der anderen Seite ist das Weil Diffie-Hellman Problem ein bislang noch sehr wenig untersuchtes Problem, so dass es sich eines Tages als gefährlich herausstellen könnte, sich allein auf die Schwierigkeit dieses Problemes verlassen zu haben. Während das Decision Diffie-Hellman Problem in vielen in der Kryptographie eingesetzten Gruppen genau so schwierig zu sein scheint, wie das zugehörige Computational Diffie-Hellman Problem, so wurde in [**JN01**] gezeigt, dass dies beim Weil Diffie-Hellman Problem *nicht* der Fall ist.

Bei gegebenem Quadrupel (P, aP, bP, cP) kann nämlich durch die Überprüfung der Gleichung $e(P, \phi(cP)) = e(aP, \phi(bP))$ eindeutig entschieden werden, ob $c = ab$ ist oder nicht. Das Decision Weil DHP ist also in polynomieller Zeit lösbar und deshalb für die Konstruktion von Kryptosystemen, beispielsweise im Stile von [**CS98**], nicht geeignet. Auch wenn das CWDHP derzeit schwierig erscheint, und deshalb das Verfahren von Boneh und Franklin nicht nur äußerst praktisch, sondern auch sicher ist, so erscheint es dennoch geboten, alternative identitätsbasierte Kryptosysteme wie [**HJW01**], die auf anderen kryptographischen Problemen basieren, näher zu untersuchen, da sie auch dann noch eingesetzt werden können, wenn sich das CWDHP als leicht lösbar erweisen sollte.

3.2.4.2. Systemaufbau

Das folgende, nicht-interaktive Verfahren wurde erstmals in [**HJW01**] vorgeschlagen und benutzt die Grundidee aus [**MY91**] – es wird ein DL-basiertes Kryptosystem mit Falltür konstruiert, bei dem die zentrale Schlüsselgenerierungsinstanz KGC durch Kenntnis der Falltürinformation in der Lage ist, die geheimen Teilnehmer-Schlüssel zu berechnen.

Der zentrale Unterschied ist, dass anstatt der Gruppe $(\mathbb{Z}/n\mathbb{Z})^*$ eine *zyklische* Klassengruppe $Cl(\Delta p^2)$ benutzt wird. Dies hat den Vorteil, dass selbst bei der naheliegenden Einbettung $f : ID \to Cl(\Delta p^2)$, bei der die Identität ID als Bitstring interpretiert wird und f einfach durch Embed-v1 (Algorithmus 16) oder Embed-v2 (Algorithmus 18) aus Abschnitt 3.2.1.2 realisiert wird, die Existenz diskreter Logarithmen zu einem gegebenen öffentlichen Schlüssel $\mathfrak{b}$ garantiert werden kann.

Außerdem schlagen wir vor, die DL-Berechnung bei der Schlüsselerzeugung nicht wie in [**MY91, MY93**] mit generischen Algorithmen durchzuführen, sondern zwei verschiedene subexponentielle Algorithmen zu verwenden. Dies hat insbesondere zur Folge, dass – nach einer Vorausberechnungsphase – der Aufwand für die Berechnung individueller Schlüssel vergleichsweise gering ist. Wie in Abschnitt 2.3.1.1 (vgl. Satz 2.3.1 und Proposition 2.3.4) erläutert, kann das DL-Problem in $Cl(\Delta p^2)$, $(\Delta/p) = 1$, vom KGC, dem der Führer p bekannt ist, auf das DL-Problem in $Cl(\Delta)$ und $\mathbb{F}_p^*$ reduziert werden. Die DL-Berechnung in $Cl(\Delta)$ führen wir mit einem Analog des „Self Initializing Quadratic Sieve“ (SIQS) [**Jac99, Jac00**] mit einer asymptotischen Laufzeit von $L_{|\Delta|}\left[\frac{1}{2}, \frac{3}{4}\sqrt{2} + o(1)\right]$ [**Vol00**] durch. Für die DL-Berechnung in $\mathbb{F}_p^*$ schlagen wir vor, das (spezielle) Zahlkörpersieb [**Web96, Web98**] zu verwenden. Wir werden später darauf eingehen, wie die Parameter Δ und p gewählt werden sollten, damit die DL-Berechnung beim KGC (effizient) möglich ist, aber ein Angreifer Δp^2 nicht faktorisieren kann.

Wir werden nun den konkreten Aufbau und die Abläufe bei der Benutzerregistrierung diskutieren. Nach der Benutzerregistrierung steht dem Teilnehmer Bob sein geheimer Schlüssel $b = \log_{\mathfrak{g}} \mathfrak{b}$, wobei $\mathfrak{b} = f(ID_B)$, zur Verfügung und er kann dieses Schlüsselpaar in einem beliebigen ElGamal-artigen Kryptosystem, wie z.B. NIQ-ElGamal-{HJPT, HJPT-without-embedding, FO, REACT} (Kryptosystem 1 - 4) ohne beschleunigte Entschlüsselung (siehe Bemerkung 3.2.1 - 3.2.5), verwenden.

Wie in Abschnitt 3.1.4.3 erläutert, verlangen wir $|\Delta|p^2 \geq 2^{896}$, damit die Faktorisierung von Δp^2 mit einem erwarteten Aufwand von $2 \cdot 10^9$ MIPS-Jahren verbunden wäre und heute unmöglich erscheint. Im Gegensatz zu den anderen in dieser Arbeit vorgestellten Verfahren, können hier keine wesentlich größeren Parameter verwendet werden, da sonst die Berechnung diskreter Logarithmen in $Cl(\Delta)$ und $\mathbb{F}_p^*$ nicht mehr praktikabel ist.

Der Systemaufbau geschieht in folgenden drei Schritten:

1. *Wahl von Δ und Berechnung der Gruppenstruktur von $Cl(\Delta)$*

   Die Schlüsselgenerierungsinstanz KGC wählt eine zufällige, starke Primzahl $d \equiv 3 \pmod 4$ der Bitlänge $l_\Delta \geq \lceil 896/3 \rceil = 299$, setzt $\Delta = -d$ und berechnet $h(\Delta)$, sowie die Gruppenstruktur von $Cl(\Delta)$, mit dem subexponentiellen Algorithmus [**Jac99**]. Der Heuristik von Cohen und Lenstra [**CL84**] zufolge ist die Wahrscheinlichkeit, dass diese Klassengruppe $Cl(\Delta)$ zyklisch ist > 0.97. Falls $Cl(\Delta)$ nicht zyklisch ist, wählt das KGC eine andere Primzahl d bis $Cl(\Delta)$ schließlich zyklisch ist.

2. *Wahl von p, so dass auch $Cl(\Delta p^2)$ zyklisch ist*

   Nun wählt das KGC eine andere zufällige Primzahl p der Bitlänge $l_p \geq 299$, so dass $(\Delta/p) = 1$ und $\mathrm{ggT}(p-1, h(\Delta)) = 1$. Durch diese Wahl ist sichergestellt, dass $\mathrm{Ker}(\phi_{Cl}^{-1}) \cong \mathbb{F}_p^*$ – also das Zahlkörpersieb [**Web96, Web98**] zur DL-Berechnung in $\mathbb{F}_p^*$ verwendet werden kann – und

auch die Klassengruppe $Cl(\Delta p^2)$ zyklisch ist.

3. *Wahl eines Erzeugers von* $Cl(\Delta p^2)$

   Das KGC berechnet schließlich einen Erzeuger $\mathfrak{g}$ von $Cl(\Delta p^2)$ und veröffentlicht ihn zusammen mit der öffentlichen Diskriminante Δp^2. Kennt das KGC – nach der Strukturberechnung von $Cl(\Delta)$ mit [**Jac99**, Algorithm 6.1] – einen Erzeuger $\mathfrak{G}$ von $Cl(\Delta)$, so ist es leicht, einen Erzeuger $\mathfrak{g}$ von $Cl(\Delta p^2)$ zu finden, so dass $\mathfrak{G} = \phi_{Cl}^{-1}(\mathfrak{g})$.

   Dazu wählt das KGC zufällige Werte $\alpha \in (\mathcal{O}_\Delta / p\mathcal{O}_\Delta)^*$, bis es schließlich auf ein $\mathfrak{g} = \varphi(\alpha\mathfrak{G}) \in Cl(\Delta p^2)$ stößt, für das $\mathfrak{g}^{h(\Delta p^2)/d_i} \not\sim \mathcal{O}_{\Delta p^2}$, für alle $d_i \,|\, h(\Delta p^2) = h(\Delta)(p-1)$.

3.2.4.3. Benutzerregistrierung

Die Registrierung individueller Benutzer geschieht in folgenden vier Schritten:

1. *Beantragung des Schlüsselpaares*

   Bob beantragt das zu seiner Identität ID_B, z.B. seiner E-Mail-Adresse, gehörige Schlüsselpaar $(\mathfrak{b} = f(ID_B), b = \log_{\mathfrak{g}} \mathfrak{b})$ beim KGC.
2. *Überprüfung der Identität*

   Das KGC überprüft Bob's Identität durch Vorlage eines Personalausweises und beginnt mit der Schlüsselerzeugung, sofern die Identifikation erfolgreich verlaufen ist.
3. *Ableiten des öffentlichen Schlüssels aus Identität*

   Das KGC verwendet eine geeignete kryptographische Hashfunktion h und berechnet den Hashwert $id = h(ID_B)$ und bettet id – z.B. durch die Algorithmen Embed-v1 (Algorithmus 16) oder Embed-v2 (Algorithmus 18) aus Abschnitt 3.2.1.2 – in ein Gruppenelement $\mathfrak{b} \in Cl(\Delta p^2)$ ein. Falls dieses Gruppenelement bereits durch einen anderen Benutzer belegt ist, wird Bob gebeten eine andere Identität, z.B. eine neue Email-Adresse oder seine Post-Anschrift, zu verwenden.
4. *Berechnung des geheimen Schlüssels durch Lösung des DL-Problemes*

   Schließlich berechnet das KGC unter Verwendung von ReduceDLP (Algorithmus 14) aus Abschnitt 2.3.1.1 den diskreten Logarithmus b, so dass $\mathfrak{g}^b \sim \mathfrak{b}$, und übergibt Bob sein Schlüsselpaar $(\mathfrak{b}, b)$.

Will Alice nun eine Nachricht für Bob verschlüsseln, so berechnet sie Bob's öffentlichen Schlüssel $\mathfrak{b} =$Embed-v2$(h(ID_B))$ und verschlüsselt ihre Nachricht mit einem ElGamal-artigen Verfahren in $Cl(\Delta p^2)$ genau so, als ob es sich bei $Cl(\Delta p^2)$ um eine Maximalordnung handeln würde. Beispielsweise kann NIQ-ElGamal-{HJPT, HJPT-without-embedding, FO, REACT} (Kryptosystem 1 - 4) ohne Anwendung der Falltürinformation zur beschleunigten Entschlüsselung (siehe Bemerkung 3.2.1 - 3.2.5) eingesetzt werden.

3.2.4.4. Sicherheitsbetrachtungen

Sofern ein Angreifer wie gewöhnlich die Systemparameter und öffentlichen Schlüssel als Input erhält und dann beispielsweise CPA- oder CCA-Attacken gegen das eingesetzte Verschlüsselungsverfahren startet, so ist die Ausgangssituation

für den Angreifer identisch zu der bei gewöhnlichen (nicht-identitätsbasierten) Verfahren und man erhält die bereits in den Abschnitten 3.2.1.3, 3.2.2.1 und 3.2.3.1 erörterten Sicherheitseigenschaften.

Dürfte ein Angreifer auch, wie in [**BF01**], beliebige Anfragen zur Schlüsselgenerierung an das KGC stellen, so wären weitere Betrachtungen im Stile von [**BF01**, Lemma 4.6] anzustellen.

3.2.4.5. Effizienzbetrachtungen

Die Effizienz der Ver- und Entschlüsselung ist identisch zu der bei einem gewöhnlichen ElGamal-Analog in der Klassengruppe einer Maximalordnung, so dass lediglich der Aufwand für die Systemerstellung und die Berechnung der geheimen Schlüssel zu beachten ist.

[**HJW03**, Section 6] berichtet von ersten praktischen Erfahrungen mit der Lösung eines DL-Problemes zur Schlüsselerzeugung für Parameter in der relevanten Größenordnung. Betrachtet man die asymptotischen Laufzeitfunktionen der zur DL-Berechnung eingesetzten Algorithmen, so ist leicht zu erkennen, dass der größte Aufwand mit der Berechnung der Gruppenstruktur von $Cl(\Delta)$ und der nachfolgenden Berechnung Diskreter Logarithmen in dieser Gruppe verbunden ist.

Hierfür wurde eine parallele Version des Algorithmus aus [**Jac99**] unter Verwendung von [**LiD**] und [**GBD$^+$94**] auf einem Cluster von 16 Linux-basierten Pentium-III/550 verwendet.

Für die Berechnung eines ersten, für den längerfristigen Einsatz zu wenig Sicherheit bietenden, Beispieles mit 265 Bit $|\Delta|$ und 267 Bit p wurden für die Berechnung der Gruppenstruktur etwa 2 Tage, 6 Stunden und 29 Minuten auf dem Pentium-Cluster benötigt, was rund 36 Tagen auf einem einzelnen Rechner entsprechen würde.

Für die Berechnung der individuellen Schlüssel ist es nötig, Diskrete Logarithmen in $Cl(\Delta)$ und $\mathbb{F}_p^*$ zu berechnen. Die Lösung eines DL-Problemes in $Cl(\Delta)$, unter Verwendung der bei der Berechnung der Gruppenstruktur erzeugten Relationen, dauerte durchschnittlich etwa 3 Minuten und 6 Sekunden. Die Berechnung von Diskreten Logarithmen in $\mathbb{F}_p^*$ mit der GNFS-Implementierung aus [**Web96, Web98**] dauerte etwa 2 Stunden und 18 Minuten.

In einem zweiten, auch für die Praxis tauglichen Beispiel, wurde eine Diskriminante Δ mit 305 Bits verwendet, die nicht völlig zufällig gewählt wurde, sondern zusätzlich die Eigenschaft $(\Delta/r) = 1$ für alle Primzahlen $r < 389$ aufweist. Wie in [**Jac99**] angedeutet, erleichtert der Einsatz einer solchen Diskriminante die praktische Berechnung der Gruppenstruktur in signifikanter Art und Weise. Diskriminanten mit solchen zusätzlichen Eigenschaften können mit Unterstützung der an der University of Manitoba entwickelten Sieving Unit (MSSU) [**LPW95**] leicht gefunden werden. Im konkreten Fall wurde die MSSU zusammen mit der in [**JW00**] beschriebenen Methode eingesetzt. Die Berechnung der Gruppenstruktur von $Cl(\Delta)$ mit dem Pentium-Cluster dauerte etwa 2 Tage, 20 Stunden und 53 Minuten. Die Berechnung individueller Diskreter Logarithmen dauerte lediglich etwa 3 Minuten und 18 Sekunden. Außerdem wurde in diesem Beispiel eine Primzahl p mit $(\Delta/p) = 1$ und 304 Bits verwendet. Die Berechnung Diskreter Logarithmen in $\mathbb{F}_p^*$ auf einem einzelnen Pentium-III/500 dauerte etwa 14 Stunden. Da die Diskriminante Δp^2 der Nichtmaximalordnung 913 Bits aufweist, können die in diesem Beispiel verwendeten Parameter auch in der Praxis eingesetzt werden.

3.2.5. Diskussion der verschiedenen ElGamal-Varianten

In diesem Abschnitt sollen die wesentlichen Ergebnisse dieser Arbeit bezüglich der ElGamal-Verschlüsselungsvarianten zusammengetragen und vergleichend diskutiert werden.

Durch das in [**HJPT98**] und Abschnitt 3.2.1.1 vorgestellte Verfahren (NIQ-ElGamal-HJPT, Kryptosystem 1) ist es möglich, die Entschlüsselung im Vergleich zum naiven ElGamal-Analog in der Gruppe $Cl(\Delta p^2)$ etwa um den Faktor fünf zu beschleunigen. Während durch dieses Verfahren der Grundstein für die kryptographische Verwendung der imaginärquadratischen Nichtmaximalordnungen gelegt wurde, so ist der praktische Einsatz dieser ursprünglichen Variante mit gewissen Problemen behaftet.

Während durch Verwendung der in Abschnitt 3.2.1.5 vorgestellten Variante NIQ-ElGamal-HJPT-without-embedding (Kryptosystem 2) auf die problematische Nachrichteneinbettung (vgl. Abschnitt 3.2.1.2) verzichtet werden kann und der Angriff aus [**JJ00**] nicht anwendbar *scheint* (vgl. Bemerkung 3.2.2), so empfiehlt sich aus Sicherheitsgründen der Einsatz der NIQ-ElGamal-FO- oder NIQ-ElGamal-REACT-Variante (Kryptosystem 3 oder Kryptosystem 4), da hierfür entsprechende Beweise für die Sicherheit (im Random-Oracle-Model) existieren.

Der Unterschied zwischen den beiden Verfahren ist, dass die NIQ-ElGamal-REACT-Variante eine deutlich effizientere Entschlüsselung ermöglicht und sich der zugehörige Sicherheitsbeweis aber „lediglich“ auf das Gap Diffie-Hellman-Problem (GDHP) in $Cl(\Delta p^2)$, anstatt – wie bei der NIQ-ElGamal-REACT-Variante – auf das möglicherweise schwierigere Computational Diffie-Hellman-Problem (CDHP) bezieht. Da das GDHP derzeit genau so schwierig scheint, wie das CDHP, wird man der NIQ-ElGamal-REACT-Variante in der Praxis den Vorzug geben.

Der in [**HJW01**] und Abschnitt 3.2.4 vorgestellte Ansatz zur Konstruktion identitätsbasierter Kryptosysteme ist unabhängig davon, welches DL-basierte Verfahren eingesetzt wird. Da für jeden Benutzer B der Diskrete Logarithmus $b = \log_{\mathfrak{g}}(f(ID_B))$ in der Gruppe $\langle \mathfrak{g} \rangle = Cl(\Delta p^2)$ berechnet wird, kann ein beliebiges DL-basiertes Kryptosystem eingesetzt werden. Deshalb wird man, wie oben erläutert, in der Praxis die NIQ-ElGamal-REACT-Variante einsetzen, wobei die Entschlüsselung wie in Bemerkung 3.2.5 erläutert, ohne Verwendung der Falltürinformation durchgeführt werden muss.

3.3. NICE-Signaturverfahren

In diesem Abschnitt widmen wir uns der Konstruktion von Signaturverfahren, die, wie das NICE-Verschlüsselungsverfahren [**PT00**], in der Gruppe $\mathrm{Ker}(\phi_{Cl}^{-1}) \subset Cl(\Delta p^2)$ operieren.

Neben dem in [**HM00b**] und Abschnitt 3.3.1 vorgestellten Schnorr-Analog, findet sich Abschnitt 3.3.2 auch eine Variante des Guillou-Quisquater Signaturverfahrens [**GQ90**] in der Gruppe $\mathrm{Ker}(\phi_{Cl}^{-1})$.

Da in beiden Fällen der Isomorphismus $\mathrm{Ker}(\phi_{Cl}^{-1}) \tilde{=} \mathbb{F}_p^*$ in der Routine GenExpIso (Algorithmus 13) zur Exponentiation verwendet wird, ist die Signaturerzeugung – bei vermutlich gleicher Sicherheit – mehr als viermal so schnell wie im jeweiligen Originalverfahren.

3.3.1. NICE-Schnorr-Variante

In diesem Abschnitt werden wir das in [**HM00b**] vorgeschlagene NICE-Schnorr-Signatur-Protokoll diskutieren.

Bei diesem Verfahren wird ausgenutzt, dass – im Gegensatz zum DSA-Verfahren[9] [**NIS94**] – beim Schnorrverfahren [**Sch90**] die Gruppenordnung zur Verifikation nicht nötig ist und deshalb geheim gehalten werden kann. Somit kann anstelle der Gruppe $\mathbb{F}_p^*$ im Originalverfahren die Gruppe $\mathrm{Ker}(\phi_{Cl}^{-1})$ verwendet werden. Wie im Original-Verfahren, wird lediglich in einer Untergruppe $\langle \mathfrak{g} \rangle \subset \mathrm{Ker}(\phi_{Cl}^{-1})$ der l_q Bit Primzahl-Ordnung q gearbeitet. Allerdings wird $q \,|\, p - (\Delta/p) = p - 1$ hier nicht veröffentlicht, um das Faktorisieren von Δp^2 nicht möglicherweise zu vereinfachen.

Durch die in Abschnitt 2.2 vorgestellte Arithmetik in $\mathrm{Ker}(\phi_{Cl}^{-1})$ ist es möglich, die Signaturerzeugung sehr effizient zu implementieren. Die Laufzeiten einer ersten Implementierung (siehe Tabelle 1, Seite 29) belegen, dass die Signaturerzeugung – bei vermutlich gleichem Sicherheitsniveau – mehr als viermal so schnell wie im Originalverfahren [**Sch90**] in $\mathbb{F}_p^*$, bzw. etwa doppelt so schnell wie ein Schnorr-Analog in der Gruppe $(\mathbb{Z}/dp^2\mathbb{Z})^*$, ist.

Wir unterstellen, dass Alice eine Signatur für eine beliebige, als Bitstring dargestellte, Nachricht $m \in \{0,1\}^*$ erzeugt und Bob diese verifiziert. Weiterhin setzen wir die Existenz einer kryptographisch geeigneten Hashfunktion $h : \{0,1\}^* \times Cl(\Delta p^2) \to \{0,1\}^{l_h}$ voraus. Bei der folgenden Darstellung machen wir davon Gebrauch, dass ein beliebiger Bitstring einer nicht-negativen Zahl zugeordnet werden kann.

Die Wahl der Sicherheitsparameter l_Δ, l_h, l_q, l_s und l_r wird später näher erläutert.

Bemerkung 3.3.1. Um die Systemparameter $\mathcal{O}_{\Delta p^2}, q$ und $\mathfrak{g}$ zu erzeugen, wählt man zuerst die Primzahl q mit der Bitlänge l_q und Δ, wie in Abschnitt 3.1.2 erläutert, und sucht daraufhin einen passenden Führer p, so dass $q \,|\, p - 1$ und $(\Delta/p) = 1$.

Um $\mathfrak{g}$ mit $|\langle \mathfrak{g} \rangle| = q$ zu erzeugen, kann solange ein zufälliges $\alpha = x + y\omega \in (\mathcal{O}_\Delta / p\mathcal{O}_\Delta)^*$ gewählt und $\beta = \alpha^{(p-1)/q}$, $\mathfrak{g} = \mathsf{Gen2Std}(\beta, p)$ berechnet werden, bis schließlich $\mathfrak{g} \neq \mathcal{O}_{\Delta p^2}$.

[9] Während die Verwendung von *vollständigen Nichtmaximalordnungen* mit $h(\Delta) = 1$ und deshalb $h(\Delta p^2) = p - \left(\frac{\Delta}{p}\right)$ die Konstruktion eines DSA-Analogs in $Cl(\Delta p^2)$ ermöglicht, so impliziert die in Abschnitt 2.3.1.1 oder [**HT99**] vorgestellte Reduktion des DL-Problemes von $Cl(\Delta p^2)$ auf endliche Körper $\mathbb{F}_{p^k}^*$, $k \in \{1,2\}$, dass dieses Verfahren keine Sicherheitsvorteile gegenüber DSA in $\mathbb{F}_{p^k}^*$ besitzt. Deshalb wird dieser Ansatz hier nicht weiter verfolgt.

Kryptosystem 5 NICE-Schnorr

Systemparameter: Eine imaginärquadratische Nichtmaximalordnung $\mathcal{O}_{\Delta p^2}$ (wie in Bemerkung 3.3.1 erläutert), die Hashfunktion $h : \{0,1\}^* \to \{0,1\}^{l_h}$, die Bitlänge des geheimen Schlüssels l_s und der Zufallszahl l_r und ein Element $\mathfrak{g} \in \mathrm{Ker}(\phi_{Cl}^{-1}) \subset Cl(\Delta p^2)$ der Ordnung $q \in_r \left[2^{l_q-1}, 2^{l_q}\right[$, q prim.

Geheime Schlüssel: Alice's geheimer Schlüssel ist ein zufällig gewähltes Element $a \in_r \left[2^{l_s-1}, 2^{l_s}\right[$, die Ordnung q der Untergruppe $\langle \mathfrak{g} \rangle \subset \mathrm{Ker}(\phi_{Cl}^{-1})$, das Bild des Erzeugers $g = \psi^{-1}(\mathfrak{g}) = \mathsf{Ker2Fp}\,(\mathsf{Std2Gen}(\mathfrak{g}, p), p, \rho, \bar{\rho})$ in $\mathbb{F}_p^*$, wobei $(\rho, \bar{\rho}) = \mathsf{GetRoots}(\Delta, p)$, und schließlich der Führer p.

Öffentliche Schlüssel: Alice's öffentlicher Schlüssel ist $\mathfrak{a} = \mathfrak{g}^a$

Signaturerzeugung: Die Signaturerzeugung erfolgt in folgenden Schritten:
Alice wählt eine Zufallszahl $r \in_r \left[2^{l_r-1}, 2^{l_r}\right[$
$r' \leftarrow r \pmod q$ mit $0 \le r' < q$
$R \leftarrow g^{r'} \pmod p$
$\mathfrak{r} \leftarrow \mathsf{Gen2Std}\,(\mathsf{Fp2Ker}(R, p, \rho, \bar{\rho}), p)$
$c \leftarrow h(m, \mathfrak{r})$
$y \leftarrow r + ac$.
Alice's Signatur für die Nachricht m ist das Tripel $(\mathfrak{r}, c, y)$.

Verifikation der Signatur: Um diese Signatur zu verifizieren, berechnet Bob

$$\mathfrak{t} = \mathfrak{g}^y \mathfrak{a}^{-c}$$

und akzeptiert Alice's Unterschrift genau dann wenn $\mathfrak{t} = \mathfrak{r}$.

Beweis der Korrektheit. Die Korrektheit des Verfahrens ist sofort aus

$$\mathfrak{t} = \mathfrak{g}^y \mathfrak{a}^{-c} = \mathfrak{g}^{r+ac} \mathfrak{g}^{-ac} = \mathfrak{g}^r = \mathfrak{g}^{r'} = \mathfrak{r}$$

ersichtlich. □

3.3.1.1. Sicherheitsbetrachtungen

Die Sicherheit des analogen Verfahrens in der Gruppe $(\mathbb{Z}/n\mathbb{Z})^*$, für beliebige n, wurde in [**PS98**] ausgiebig untersucht. Da sich die Sicherheitsbeweise Wort für Wort auf unseren Fall übertragen lassen, zitieren wir lediglich die dortigen Resultate.

Ein Ausdruck f wird vernachlässigbar genannt, wenn f vom Sicherheitsparameter l_Δ abhängt und für eine beliebige Konstante c, und ausreichend großes l_Δ, $f(l_\Delta) < 1/l_\Delta^c$ gilt.

Satz 3.3.2. *Sei T die Anzahl der Nachrichten, die mit einem festen Schlüssel signiert werden. Man nehme an, dass $2^{l_h+l_s}T/2^{l_r}$ und $1/2^{l_h}$ vernachlässigbar sind. Falls die Erzeugung einer gültigen Signatur in Kryptosystem 5 durch einen Angreifer bei einem adaptivem, aktiven Angriff eine nicht-vernachlässigbare Erfolgswahrscheinlichkeit besitzt, so können auch diskrete Logarithmen in $\langle \mathfrak{g} \rangle \subset \mathrm{Ker}(\phi_{Cl}^{-1}) \subset Cl(\Delta p^2)$ in polynomieller Zeit (in l_Δ) berechnet werden.*

Beweis. Ersetzt man die Gruppe $G = (\mathbb{Z}/n\mathbb{Z})^*$ und damit zusammenhängende Begriffe (Multiplikation modulo n, $\lambda(n)$ als größte Ordnung eines Elementes in G, ...) durch die Gruppe $\langle \mathfrak{g} \rangle \subset Cl(\Delta p^2)$ und entsprechende Begriffe, so kann der Beweis von [**PS98**, Theorem 10] Wort für Wort auf unseren Fall übertragen werden. □

Der entscheidende Punkt für die Sicherheit des Verfahrens ist die Größe von $l_r \geq l_h + l_s + \log_2(T)$. In [**PS98**, Section 4.1] wird

$$l_r \geq l_h + l_s + 80 \tag{3.3.1}$$

als ausreichend sicher erachtet.

Es müssen schließlich noch konkrete Werte für l_Δ, l_h, l_s und l_r angegeben werden.

Um die Faktorisierung von Δp^2 und die Berechnung diskreter Logarithmen in $Cl(\Delta p^2)$ mit subexponentiellen Algorithmen zu verhindern, fordern wir, wie in Abschnitt 3.1.4.3 näher erläutert, $l_\Delta \geq 299$. Damit die Erzeugung beliebiger Kollisionen für die Hashfunktion h nicht möglich erscheint, fordern wir $l_h \geq 160$. Damit die DL-Berechnung in $\langle \mathfrak{g} \rangle$ mit einem generischen Algorithmus [**Pol78**] nicht in vertretbarer Zeit möglich ist, wird $l_r \geq l_s = l_q \geq 160$ gefordert. Aus (3.3.1) erhalten wir also $l_r \geq 160 + 160 + 80 = 400$.

Bemerkung 3.3.3. Die Verifikation der Signatur erfolgt vollständig in der Klassengruppe der Nichtmaximalordnung und ist deshalb vergleichsweise ineffizient. Ist, wie vorgeschlagen, $l_h = 160$ und $l_r = 400$, so ist der aufwendigste Teil dieser Verifikation die Berechnung von $\mathfrak{g}^y$. Deshalb wurde in [**HM00b**] vorgeschlagen, anstatt $y = r + ac$ den modulo q reduzierten Wert $y' \equiv r + ac \pmod q$ zurückzuliefern.

Während für diese Variante kein rigoroser Sicherheitsbeweis (auch nicht im Random Oracle Model [**BR93**]) vorhanden ist, so wurde in [**HM00b**, Section 4] zumindest gezeigt, dass der naheliegende Ansatz, den Wert von q durch mehrere Signaturen zu approximieren und schließlich zu raten, mit sehr großer Wahrscheinlichkeit nicht zum Erfolg führt.

Außerdem ist es beim hier vorliegenden Verfahren unklar, ob ein Angreifer von der Kenntnis von q überhaupt profitieren könnte. Hier sei angemerkt, dass ein Angreifer beim in Abschnitt 3.3.1.3 diskutierten Schnorr-Analog in $(\mathbb{Z}/dp^2\mathbb{Z})^*$ den Modul dp^2, unter Verwendung des in [**Mao98**] bzw. [**ML98**] vorgestellten Verfahrens, möglicherweise leichter faktorisieren kann, sofern er q kennt. Da dieser Ansatz die Struktur von $(\mathbb{Z}/n\mathbb{Z})^*$ ausnutzt, scheint er nicht ohne weiteres auf den hier vorliegenden Fall übertragbar.

3.3.1.2. Effizienzbetrachtungen

Bezüglich der Effizienz des Verfahrens muss die Signaturerzeugung und Signaturverifikation getrennt betrachtet werden.

Signaturerzeugung

Da der Führer p ein Teil des geheimen Schlüssels ist, kann anstatt der naiven Idealarithmetik in $Cl(\Delta p^2)$ die deutlich effizientere, in Abschnitt 2.2 und insbesondere in den Abschnitten 2.2.2 und 2.2.3 ausführlich diskutierte, Erzeuger-Arithmetik für $\mathrm{Ker}(\phi_{Cl}^{-1}) \subset Cl(\Delta p^2)$ zur effizienten Signaturerzeugung eingesetzt werden.

Wie in Tabelle 1 (Seite 29) ersichtlich, ist eine Exponentiation in $\mathrm{Ker}(\phi_{Cl}^{-1})$ mit einer $3l_\Delta$ Bit Diskriminante Δp^2 bei Verwendung des Algorithmus GenExpIso (Algorithmus 13) mehr als vier mal so schnell, wie eine modulare Exponentiation in $\mathbb{F}_{p'}^*$ für ein $3l_\Delta$ Bit großes p'.

Unterstellt man, dass die Berechnung Diskreter Logarithmen in einer Untergruppe von $\mathrm{Ker}(\phi_{Cl}^{-1})$ genau so schwer ist, wie in einer Untergruppe von $\mathbb{F}_p^*$, dann

ist die Signaturerzeugung im hier vorgestellten Verfahren bei vermutlich gleicher Sicherheit mehr als vier mal so schnell, wie im Originalverfahren [**Sch90**].

Signaturverifikation

Bei der Signaturverifikation scheinen neben der in Bemerkung 3.3.3 angedeuteten Variante und generischen Ansätzen, keine (naheliegenden) Möglichkeiten zur Effizienzsteigerung bei der Verifikation zu existieren.

3.3.1.3. Vergleich mit dem Schnorr-Analog in $(\mathbb{Z}/dp^2\mathbb{Z})^*$

Der große Vorteil von NICE-Schnorr (Kryptosystem 5) verglichen mit dem Original [**Sch90**] ist, dass die Signaturerzeugung – durch den in Abschnitt 2.2.3 vorgestellten Isomorphismus $\mathrm{Ker}(\phi_{Cl}^{-1}) \cong \mathbb{F}_p^*$ – sehr effizient implementiert werden kann.

In diesem Abschnitt soll schließlich untersucht werden, ob ein naheliegendes Schnorr-Analog in einer Untergruppe von $(\mathbb{Z}/dp^2\mathbb{Z})^*$ der Ordnung q ähnliche Vorteile verspricht.

Sei $n = dp^2$, d, p prim und g ein Erzeuger der Untergruppe der Ordnung q, q prim. Dann zeigt der Chinesische Restsatz, dass $g^q \equiv 1 \pmod{d}$ und $g^q \equiv 1 \pmod{p^2}$, was wiederum $g^q \equiv 1 \pmod{p}$ impliziert. Deshalb muss q mindestens eine der beiden Zahlen $d-1$ oder $p-1$ teilen. Wir erhalten also die folgenden drei Fälle:

1. $q \,|\, (d-1)$ und $q \not| (p-1)$:

   Da g in $(\mathbb{Z}/n\mathbb{Z})^*$ die Ordnung q hat und nach Voraussetzung $q \,|\, (d-1)$ ist, gilt also $g \not\equiv 1 \pmod{d}$. Nun impliziert aber q prim, $g^q \equiv 1 \pmod{p}$ und $q \not| (p-1)$, dass $g \equiv 1 \pmod{p}$ und deshalb $g - 1 = k \cdot p$ für ein $k \in \mathbb{Z}$. Somit kann n, durch Berechnung von $p = \mathrm{ggT}(g-1, n)$, sofort faktorisiert werden. Es sei angemerkt, dass hierfür q nicht bekannt sein muss und dieser Fall bei festgelegtem q und d und zufälliger Wahl von p sehr wahrscheinlich ist. Deshalb muss dieser Fall aus Sicherheitsgründen unbedingt vermieden werden. Das ist besonders schade, da ja $g \equiv 1 \pmod{p}$ ist und in diesem Fall – ähnlich wie bei unserem Schnorr-Analog in $\mathrm{Ker}(\phi_{Cl}^{-1})$ (Kryptosystem 5) – eine einzige Exponentiation in $\mathbb{F}_d^*$ genügt hätte, um die zur Signaturerzeugung notwendige Exponentiation in $(\mathbb{Z}/n\mathbb{Z})^*$ zu realisieren.

2. $q \not| (d-1)$ und $q \,|\, (p-1)$:

   Dieser Fall ist analog zum eben erläuterten. Hier ist $g \not\equiv 1 \pmod{p}$ und $g \equiv 1 \pmod{d}$ und n kann durch Berechnung von $d = \mathrm{ggT}(g-1, n)$ sofort faktorisiert werden.

3. $q \,|\, (d-1)$ und $q \,|\, (p-1)$:

   In diesem Fall ist das Setup ähnlich wie in [**Gir91**] und das Verfahren ist nicht sofort gebrochen. Allerdings ist die Signaturerzeugung hier auch etwas weniger effizient. Um die zur Signaturerstellung notwendige Exponentiation $g^{r'} \pmod{n}$, $n = dp^2$ zu realisieren, wird man den Chinesischen Restsatz verwenden, d.h. die beiden Teilergebnisse $g^{r'} \pmod{d}$ und $g^{r'} \pmod{p^2}$ berechnen und diese später zur Lösung in $(\mathbb{Z}/n\mathbb{Z})^*$ zusammenzusetzen. Wird die Exponentiation modulo p^2 mit dem in [**Tak98**, Section 2.2] vorgestellten Verfahren realisiert, so wird hierfür kaum mehr Aufwand als für eine Exponentiation in $\mathbb{F}_p^*$ benötigt. Zur Signaturerzeugung werden also rund *zwei*

modulare Exponentiationen mit l_Δ Bit langen Zahlen benötigt. Die Signaturerzeugung in diesem Verfahren dauert also etwa doppelt so lang, wie bei dem oben vorgestellten Schnorr-Analog in $\mathrm{Ker}(\phi_{Cl}^{-1})$.

Außerdem wäre es hier möglicherweise gefährlich, wenn der Faktor q der Untergruppe öffentlich bekannt wäre. Denn in diesem Fall wäre es unter Umständen möglich, unter Verwendung der in [**Mao98, ML98**, Section 4] vorgestellten Ideen, die Faktorisierung des Modul dp^2 zu erleichtern.

Diese Diskussion zeigt, dass die Parameter bei einem Schnorr-Analog in $(\mathbb{Z}/n\mathbb{Z})^*$, $n = dp^2$ mit großer Sorgfalt gewählt werden müssen und die Signaturerzeugung dennoch etwa den doppelten Aufwand verursacht, wie bei unserem Schnorr-Analog in $\mathrm{Ker}(\phi_{Cl}^{-1})$. Soll also insbesondere die Erzeugung der Signaturen effizient möglich sein, so wird man das hier vorgestellte Schnorr-Analog NICE-Schnorr (Kryptosystem 5) einem Schnorr-Analog in $(\mathbb{Z}/dp^2\mathbb{Z})^*$ vorziehen.

3.3.2. NICE-GQ-Variante

In diesem Abschnitt zeigen wir, wie eine in $\mathrm{Ker}(\phi_{Cl}^{-1})$ operierende Variante des Guillou-Quisquater-Verfahrens (GQ) [**GQ90**] mit schneller Signaturerzeugung konstruiert werden kann.

Die Verwendung imaginärquadratischer Klassengruppen zur Konstruktion von GQ-Signaturverfahren ist nicht neu. Da bei diesem Verfahren weder der Erzeuger, noch der Prüfer einer Signatur die Gruppenordnung kennen muss, kann nämlich – wie in [**Ham02**] vorgeschlagen – eine beliebige imaginärquadratische Klassengruppe $Cl(\Delta)$ zur Konstruktion eines GQ-Analoges verwendet werden. Bei der hier vorgestellten Variante wird lediglich zusätzlich der nur dem Schlüsselinhaber bekannte Isomorphismus $\mathrm{Ker}(\phi_{Cl}^{-1}) \stackrel{\sim}{=} \mathbb{F}_p^*$ ausgenutzt, um die Signaturerzeugung zu beschleunigen.

Wir unterstellen wieder, dass Alice eine Signatur für eine beliebige, als Bitstring dargestellte, Nachricht $m \in \{0,1\}^*$ erzeugt und Bob diese verifiziert. Weiterhin setzen wir wieder die Existenz einer kryptographisch geeigneten Hashfunktion $h : \{0,1\}^* \times Cl(\Delta p^2) \to \{0,1\}^{l_h}$ voraus.

Kryptosystem 6 NICE-GQ

Systemparameter: Eine imaginärquadratische Nichtmaximalordnung $\mathcal{O}_{\Delta p^2}$ (wie in Abschnitt 3.1.2 erläutert), die Hashfunktion $h : \{0,1\}^* \times Cl(\Delta p^2) \to \{0,1\}^{l_h}$ und die Bitlänge l_n des öffentlichen Parameters n.

Geheime Schlüssel: Alice's geheimer Schlüssel ist ein zufällig gewähltes Element $a \in \mathbb{F}_p^*$, dessen Bild $\mathfrak{a} = \psi(a) = \mathsf{Gen2Std}\,(\mathsf{Fp2Ker}(a, p, \rho, \bar{\rho}), p)$ in $\mathbb{F}_p^*$, wobei $(\rho, \bar{\rho}) = \mathsf{GetRoots}(\Delta, p)$, und schließlich der Führer p.

Öffentliche Schlüssel: Alice's öffentlicher Schlüssel ist das Paar $(n, \mathfrak{n})$, wobei $n \in_r \left[2^{l_n - 1}, 2^{l_n}\right[$ zufällig gewählt ist und $\mathfrak{n} = \mathfrak{a}^{-n}$.

Signaturerzeugung: Alice wählt ein zufälliges Element $r \in_r \mathbb{F}_p^*$ und berechnet
$t \leftarrow r^n \pmod p$,
$\mathfrak{t} \leftarrow \mathsf{Gen2Std}\,(\mathsf{Fp2Ker}(t, p, \rho, \bar{\rho}), p)$,
$e \leftarrow h(m, \mathfrak{t})$,
$s \leftarrow ra^e \pmod p$ und
$\mathfrak{s} \leftarrow \mathsf{Gen2Std}\,(\mathsf{Fp2Ker}(s, p, \rho, \bar{\rho}), p)$.
Alice's Signatur für die Nachricht m ist das Paar $(e, \mathfrak{s})$.

Verifikation der Signatur: Um diese Signatur zu verifizieren, berechnet Bob $\mathfrak{t}' = \mathfrak{s}^n \mathfrak{n}^e$ und akzeptiert Alice's Unterschrift genau dann wenn $h(m, \mathfrak{t}') = e$.

BEWEIS. Die Korrektheit des Verfahrens ist sofort aus $\mathfrak{t}' = \mathfrak{s}^n \mathfrak{n}^e = (\mathfrak{r}\mathfrak{a}^e)^n \mathfrak{a}^{-ne} = \mathfrak{r}^n = \mathfrak{t}$ ersichtlich. □

3.3.2.1. Sicherheitsbetrachtungen

Bezüglich der Sicherheit dieses Verfahrens verweisen wir auf Hamdy's Resultate. Wendet man das generische Resultat aus [**Ham02**, Theorem 6.5.4] auf den hier vorliegenden Fall an, so erhält man die folgende Aussage:

SATZ 3.3.4. *Falls die Erzeugung einer gültigen Signatur in* NICE-GQ *(Kryptosystem 6) durch einen Angreifer bei einem adaptivem, aktiven Angriff eine nichtvernachlässigbare Erfolgswahrscheinlichkeit besitzt, dann kann das Wurzelproblem in* $\mathrm{Ker}(\phi_{Cl}^{-1})$ (n-$\mathrm{RP}_{\mathrm{Ker}(\phi_{Cl}^{-1})}$) *in probabilistischer Polynomialzeit (in* l_Δ*) gelöst werden.*

Es bleibt also die generelle Schwierigkeit des Wurzelproblemes in $\mathrm{Ker}(\phi_{Cl}^{-1})$ zu beleuchten und daraus die notwendigen Parameterlängen für l_h, l_Δ und l_n abzuleiten.

Wie in (3.1.3) und (3.1.4) ersichtlich, ist das Wurzelproblem in $\mathrm{Ker}(\phi_{Cl}^{-1})$ höchstens so schwierig wie das Faktorisierungsproblem für Δp^2 und das generelle Wurzelproblem n-$\mathrm{RP}_{Cl(\Delta p^2)}$. Wie in Abschnitt 3.1.4.3 erläutert, muss deshalb $l_\Delta \geq 299$ gewählt werden. Umgekehrt ist – analog zum RSA-Problem – leider nicht bekannt, ob ein Algorithmus, der n-$\mathrm{RP}_{\mathrm{Ker}(\phi_{Cl}^{-1})}$ löst, auch zur Faktorisierung von Δp^2 verwendet werden kann.

Neben dem Ansatz die Diskriminante zu faktorisieren oder mit dem noch aufwändigeren Algorithmus [**Jac99**] die Klassenzahl $h(\Delta p^2)$ zu bestimmen, könnte ein Angreifer auch versuchen die Berechnung der n-ten Wurzel mit einem generischen Algorithmus, wie z.B. Shank's Baby-Step-Giant-Step-Verfahren [**Sha72**] oder Pollard's ρ- und λ-Algorithmen [**Pol78, Tes98**], durchzuführen. Wählt man $l_n \geq 160$, so ist ein solcher Angriff wenig erfolgsversprechend. Um die Konstruktion

gezielter Kollisionen in h unmöglich zu machen, muss schließlich $l_h \geq 160$ gewählt werden.

3.3.2.2. Effizienzbetrachtungen

Da bei der Signaturerzeugung der Isomorphismus $\psi : \mathbb{F}_p^* \xrightarrow{\sim} \mathrm{Ker}(\phi_{Cl}^{-1})$ verwendet werden kann, müssen lediglich etwa zwei Exponentiationen in $\mathbb{F}_p^*$ durchgeführt werden.

Analog zum Schnorr-Analog in $(\mathbb{Z}/dp^2\mathbb{Z})^*$, das in Abschnitt 3.3.1.3 diskutiert wurde, ist es natürlich auch hier möglich, ein Guillou-Quisquater-Verfahren in $(\mathbb{Z}/dp^2\mathbb{Z})^*$ zu konstruieren. Während ähnliche Sicherheitseigenschaften zu erwarten sind, muss die Exponentiation bei Verwendung des Chinesischen Restsatzes in $\mathbb{F}_d^*$ und $(\mathbb{Z}/p^2\mathbb{Z})^*$ durchgeführt werden. Da man auch hier das Verfahren aus [**Tak98**, Section 2.2] verwenden wird, werden insgesamt etwa zwei Exponentiationen mit l_Δ Bit Zahlen benötigt.

Während die Signaturverifikation beim GQ-Analog in $(\mathbb{Z}/dp^2\mathbb{Z})^*$ rund fünfzehn mal schneller ist, als die Verifikation beim NICE-GQ-Verfahren in $Cl(\Delta p^2)$ (vgl. Tabelle 1, Seite 29), so ist die Signaturerzeugung etwa mit dem doppelten Aufwand verbunden.

3.3.3. Diskussion der beiden NICE-Signatur-Varianten

In diesem Abschnitt sollen die beiden NICE-Signatur-Varianten schließlich zusammenfassend diskutiert werden.

Wie in den vorherigen Kapiteln gezeigt, kann die Signaturerzeugung bei den Verfahren in $\mathrm{Ker}(\phi_{Cl}^{-1})$ etwa doppelt so schnell realisiert werden, wie bei den analogen Verfahren in $(\mathbb{Z}/dp^2\mathbb{Z})^*$. Im Vergleich zu den Originalverfahren in $\mathbb{F}_p^*$ bzw. $(\mathbb{Z}/pq\mathbb{Z})^*$ ist der Performance-Gewinn selbstverständlich noch größer. Dem steht lediglich die ineffizientere Signaturverifikation in der Nichtmaximalordnung $Cl(\Delta p^2)$ entgegen. In Systemen, in denen dieser Nachteil keine Rolle spielt, kann die Verwendung der hier vorgestellten Signaturverfahren also eine interessante Alternative sein.

Ob man das jeweilige Schnorr- oder Guillou-Quisquater-Verfahren bevorzugt, kann nicht pauschal beantwortet werden. Neben generell unterschiedlichen Wurzel- und DL-Problemen auf denen die beiden Verfahren basieren, gibt es gewisse Unterschiede bzgl. der Performanz der Signaturerzeugung und -verifikation. Während die Signaturerzeugung bei den Schnorr-Verfahren etwa doppelt so schnell ist, wie bei den GQ-Verfahren, so ist die Verifikation beim GQ-Verfahren ein wenig effizienter. Außerdem ist es leicht möglich, das GQ-Verfahren identitätsbasiert zu machen, indem $\mathfrak{n}$ aus einem Identitätsstring abgeleitet wird. Ähnliches würde man beim Schnorr-Verfahren nur durch die in Abschnitt 3.2.4 erläuterten Ansätze im Stil von [**MY91**] oder [**HJW01**] erreichen können.

Ausblick

Wir wollen diese Arbeit damit beschließen, kurz auf die hier vorgestellten Höhepunkte und insbesondere weitere Ansätze für zukünftige Arbeiten einzugehen.

In dieser Arbeit wurden, ausgehend vom grundsätzlichen Vorschlag imaginärquadratische Nichtmaximalordnungen [**HJPT98**] zur Konstruktion von Kryptosystemen zu verwenden, verschiedene Verfahren vorgestellt, die im Vergleich zu konventionellen Systemen zum Teil sehr attraktive Eigenschaften aufweisen.

So kann die Signaturerzeugung und Entschlüsselung bei den Kryptosystemen der NICE-Familie [**PT00, HM00b**], die typischerweise auf Chipkarten mit begrenzter Leistungsfähigkeit durchgeführt werden muss, unter Verwendung der hier vorgestellten Algorithmen [**Hüh00, Hüh01**] deutlich effizienter durchgeführt werden, als bei konventionellen Verfahren.

Die in dieser Arbeit vorgestellte Verallgemeinerung der in [**HT99**] vorgeschlagenen DL-Problemreduktion kann zur Konstruktion nicht-interaktiver Verschlüsselungsverfahren [**HJW01, HJW03**] verwendet werden, bei denen Sicherheitsprobleme bei der Nachrichteneinbettung vermieden werden können. Außerdem kann die Arbeitslast für die zentrale Schlüsselerzeugungsinstanz – durch den Einsatz subexponentieller Algorithmen – vergleichsweise gering gehalten werden. Somit steht neben dem identitätsbasierten Verfahren auf Basis des Weil-Diffie-Hellman-Problemes [**BF01**] ein weiteres vergleichsweise praktikables Verfahren zur Verfügung.

Während die hier vorgestellten Kryptosysteme bestimmte praktische Vorteile besitzen, und deshalb die Standardisierung und Vermarktung der Verfahren angeregt werden könnte, zeigt die vorliegende Arbeit aber auch gewisses Forschungspotenzial für zukünftige Arbeiten auf.

Ein bislang unbefriedigend gelöstes Problem ist die elegante und effiziente Einbettung von Nachrichten in reduzierte Ideale. Wünschenswert scheint hier eine Methode, die weniger Redundanz und Rechenaufwand erfordert. Außerdem sollten die in [**Tak01**, Section 2.4.3] skizzierten Untersuchungen, welchen Einfluß die Kenntnis eines Kern-Elementes bei NICE auf die Sicherheit des Verfahren hat, vertieft und möglichst mit Beweisen unterlegt werden. Dies ist eng mit der Frage verbunden, ob das Faktorisierungsproblem $\mathrm{FP}_{\Delta p^2}$ auf die Lösung des Kern-Problemes SKEP reduziert werden kann. Allgemeiner betrachtet könnte der Versuch gestartet werden, die in (3.1.4) aufgezeigten Beziehungen zwischen den verschiedenen Berechnungsproblemen und Kryptosystemen um die eine oder andere weitere Relation zu ergänzen. Besonders interessant scheint die Frage, ob nicht bereits die Lösung des DL-Problemes in einer Untergruppe von $\mathrm{Ker}(\phi_{Cl}^{-1})$ der Ordnung q oder des Wurzelproblemes in $\mathrm{Ker}(\phi_{Cl}^{-1})$ zur Faktorisierung von Δp^2 verwendet werden kann. Dies würde implizieren, dass das Brechen der beiden NICE-Signaturverfahren äquivalent zum Faktorisieren ist.

Während die Signaturerzeugung im NICE-Schnorr-Verfahren – verglichen mit dem Originalverfahren – bereits sehr effizient ist, so scheint es nicht ausgeschlossen zu sein, ein völlig neuartiges Signaturverfahren zu entwickeln, das die vorgegebene Struktur von Maximal- und Nichtmaximalordnung noch besser ausnutzt.

Da die Verfahren der NICE-Familie auf der Schwierigkeit der Faktorisierung von Δp^2 beruhen, sollte zukünftig verstärkt nach Verbesserungen von ECM zum Faktorisieren von derartigen Zahlen gesucht werden. Hier könnte ein erster Ansatz darin bestehen, Elliptische Kurven über $\mathbb{Z}/pq^2\mathbb{Z}$ und deren Zusammenhang mit den gleichen Kurven über $\mathbb{F}_p$, näher zu untersuchen. Außerdem scheint eine Variante des Zahlkörpersiebes, die diese spezielle Struktur ausnutzt, nicht grundsätzlich unmöglich zu sein. Hier scheint es am erfolgsversprechendsten nach für diesen Fall geeigneteren Möglichkeiten zur Polynomwahl zu suchen.

Da das Faktorisieren – zumindest solcher Zahlen – nicht für alle Zeit ein schwieriges Problem bleiben muss, ist es auch ratsam, sich um weitere Effizienzsteigerungen von Verfahren auf Basis von Maximalordnungen zu bemühen.

Ein weiterer, nicht völlig offensichtlicher, Nebenaspekt der in Abschnitt 2.2 diskutierten Zusammenhänge zwischen $\mathrm{Ker}(\phi_{Cl}^{-1})$ und endlichen Körpern, liegt in der Umkehrung der Argumentation. So könnte beispielsweise im Fall $(\Delta/p) = -1$ die DL-Berechnung in $\mathbb{F}_{p^2}^*$ auf die DL-Berechnung in $\mathbb{F}_p^*$ und $\mathrm{Ker}(\phi_{Cl}^{-1})$ zurückgeführt werden. Betrachtet man nun eine Ordnung mit $h(\Delta) = 1$, so ist $Cl(\Delta p^2) = \mathrm{Ker}(\phi_{Cl}^{-1})$. Deshalb könnte es hier interessant sein, Varianten des subexponentiellen Algorithmus [**Jac99**] zu untersuchen, die eine einfachere DL-Berechnung in $Cl(\Delta p^2)$, mit sehr kleinem $|\Delta|$, erlauben.

Ein bislang in der Kryptographie wenig beachteter Aspekt imaginärquadratischer Ordnungen ist die enge Beziehung zu nicht-supersingulären elliptischen Kurven, deren Endomorphismenring isomorph zu $\mathcal{O}_\Delta$, $\Delta < 0$, ist. Hier wäre zu untersuchen, inwieweit die Theorie der komplexen Multiplikation zu neuartigen Attacken gegen Kryptosysteme auf Basis elliptischer Kurven verwendet werden könnte.

Zusammenfassend scheint diese kurze Auflistung offener Probleme zu zeigen, dass in diesem Bereich auch weiterhin *großer Forschungsbedarf* existiert. Außerdem scheinen einige bereits existierende Verfahren auf Basis imaginärquadratischer Ordnungen bereits die „nötige Reife“ für den praktischen Einsatz zu besitzen.

Literaturverzeichnis

[BB99] Biehl, Ingrid und Johannes Buchmann: *An analysis of the reduction algorithms for binary quadratic forms*. In: Engel, Peter und Halyna M. Syta (Herausgeber): *Voronoi's Impact on Modern Science, Kyiv, Ukraine 1998*, Seiten 71–98. National Academy of Sciences of Ukraine, 1999.

[BBHM00] Biehl, Ingrid, Johannes Buchmann, Safuat Hamdy und Andreas Meyer: *A Signature Scheme Based on the Intractability of Extracting Roots*. Technischer Bericht TI-1/00, Technische Universität Darmstadt, Fachbereich Informatik, 2000. http://www.informatik.tu-darmstadt.de/TI/Veroeffentlichung/TR/.

[BBP03] Bellare, Mihir, Alexandra Boldyreva und Adriana Palacio: *An Un-Instantiable Random-Oracle-Model Scheme for a Hybrid-Encryption Problem*. Cryptology ePrint Archive, Report 2003/077, 2003. http://eprint.iacr.org/.

[BCI85] Beth, Thomas, Norbert Cot und Ingemar Ingemarsson (Herausgeber): *Advances in Cryptology*, Band 209 der Reihe *Lecture Notes in Computer Science*. Springer-Verlag, 1985.

[BDHG99] Boneh, Dan, Glenn Durfee und Nick Howgrave-Graham: *Factoring $N = p^r q$ for Large r*. In: Wiener, Michael [**Wie99**], Seiten 326–337.

[BDPR98] Bellare, Mihir Bellare, Anand Desai, David Pointcheval und Phillip Rogaway: *Relations Among Notions of Security for Public-Key Encryption Schemes*. In: Krawczyk, Hugo [**Kra98**], Seiten 26–45.

[Ber93] Berger, Franz-Dieter: *ECM – Faktorisieren mit elliptischen Kurven*. Diplomarbeit, Fachbereich Informatik, Universität des Saarlandes, Saarbrücken, Germany, 1993. http://www.informatik.tu-darmstadt.de/TI/Veroeffentlichung/reports/.

[Ber01] Bernstein, Daniel J.: *Circuits for integer factorization: a proposal*, 2001. http://cr.yp.to/papers/nfscircuit.pdf.

[BF01] Boneh, Dan und Matthew Franklin: *Identity based encryption from the Weil Pairing*. In: tbd (Herausgeber): *Advances in Cryptology – CRYPTO 2001*, Band 2139 der Reihe *Lecture Notes in Computer Science*, Seiten 213–229. Springer-Verlag, 2001.

[BH96] Baker, R. C. und G. Harman: *The difference between consecutive primes*. Proc. London Math. Soc., 3(72):261–280, 1996.

[BHP01] Baker, R. C., G. Harman und J. Pintz: *The difference between consecutive primes II*. Proc. London Math. Soc., 3(83):532–562, 2001.

[BLP93] Buhler, Joe, Hendrik Lenstra und Carl Pomerance: *Factoring integers with the number fields sieve*. In: Lenstra, A.K. und H.W. Lenstra [**LL93a**], Seiten 50–94.

[BPT99] Biehl, Ingrid, Sachar Paulus und Tsuyoshi Takagi: *Efficient Undeniable Signature Schemes based on Ideal Arithmetic in Quadratic Orders*. In: *The Mathematics of Public-Key Cryptography, Toronto 1999*, Seiten 1–17. The Fields Institute for Research in the Mathematical Sciences, 1999.

[BR93] Bellare, Mihir und Paul Rogaway: *Random Oracles are Practical: a Paradigm for Designing Efficient Protocols*. In: *1st ACM Conference on Computer and Communications Security*, Seiten 62–73, 1993.

[Bra90] Brassard, Gilles (Herausgeber): *Advances in Cryptology – CRYPTO '89*, Band 435 der Reihe *Lecture Notes in Computer Science*. Springer-Verlag, 1990.

[Bre80] Brent, Richard P.: *An improved Monte Carlo factorization algorithm*. BIT, 20:176–184, 1980.

[Bre86] Brent, Richard P.: *Some Integer Factorization Algorithms using Elliptic Curves*. Australian Computer Science Communications, 8:149–163, 1986.

[Bre99] Brent, Richard P.: *Factorization of the tenth Fermat number*. Mathematics of Computation, 68:429–451, 1999. http://web.comlab.ox.ac.uk/oucl/work/richard.brent/pd/rpb161.pdf.

[BS66] Borevich, Zenon I. und Igor R. Shafarevic: *Number theory*. Academic Press, 1966.

[BS96] Bach, Eric und Jeffrey Shallit: *Algorithmic Number Theory*, Band 1. MIT Press, 1996.

[BST02] Buchmann, Johannes, Kouichi Sakurai und Tsuyoshi Takagi: *An IND-CCA2 Public-Key Cryptosystem with Fast Decryption*. In: Kim, K. [**Kim02**], Seiten 51–71.

[Bue89] Buell, Duncan A.: *Binary Quadratic Forms: Classical Theory and Modern Computations*. Springer–Verlag, 1989.

[BW88] Buchmann, Johannes und Hugh C. Williams: *A key-exchange system based on imaginary quadratic fields*. Journal of Cryptology, 1(3):107–118, 1988.

[BW90] Buchmann, Johannes und Hugh C. Williams: *A Key Exchange System Based on Real Quadratic Fields*. In: Brassard, Gilles [**Bra90**], Seiten 335–343.

[CDEH+96] Cowie, James, Bruce Dodson, R. Marije Elkenbracht-Huizing, Arjen K. Lenstra, Peter L. Montgomery und Jörg Zayer: *A world wide number field sieve factoring record: On to 512 its*. In: Kim, Kwangjo und Tsutomu Matsumoto (Herausgeber): *Advances in Cryptology – ASIACRYPT '96*, Band 1163 der Reihe *Lecture Notes in Computer Science*, Seiten 382–394. Springer-Verlag, 1996.

[CDL+99a] Cavallar, Stefania, Bruce Dodson, Arjen K. Lenstra, Paul Leyland, Walter M. Lioen, Peter L. Montgomery, Brian Murphy, Herman te Riele und Paul Zimmermann: *Factorization of RSA-140 Using the Number Field Sieve*. In: Lam, Kwok Yun et al. [**LOX99**], Seiten 195–207.

[CDL+99b] Cavallar, Stefania, Bruce Dodson, Arjen K. Lenstra, Paul Leyland, Walter M. Lioen, Peter L. Montgomery, Herman te Riele und Paul Zimmermann: *211-digit SNFS factorization*, 1999. http://ftp.cwi.nl/herman/SNFSrecords/SNFS-211.

[CDL+00a] Cavallar, Stefania, Bruce Dodson, Arjen K. Lenstra, Paul Leyland, Walter M. Lioen, Peter L. Montgomery, Brian Murphy, Herman te Riele und Paul Zimmermann: *Factorization of a 512-bit RSA modulus*. In: Preneel, Bart [**Pre00**], Seiten 1–18.

[CDL+00b] Cavallar, Stefania, Bruce Dodson, Arjen K. Lenstra, Paul Leyland, Walter M. Lioen, Peter L. Montgomery, Herman te Riele und Paul Zimmermann: *233-digit SNFS factorization*, 2000. http://ftp.cwi.nl/herman/SNFSrecords/SNFS-233.

[CGH98] Canetti, Ran, Oded Goldreich und Shai Halevi: *The random oracle methodology, revisited*. In: *Proceedings of the Thirtieth Annual ACM Symposium on the Theory of Computing*, Seiten 209–218, Dallas, Texas, USA, May 23-26, 1998, 1998. Springer-Verlag.

[CL84] Cohen, Henri und Hendrik W. Lenstra, Jr.: *Heuristics on class groups of number fields*. In: Jager, Hendrik (Herausgeber): *Number Theory, Noordwijkerhout 1983*, Band 1068 der Reihe *Lecture Notes in Mathematics*, Seiten 33–62. Springer-Verlag, 1984.

[Coh95] Cohen, Henri: *A Course in Computational Algebraic Number Theory*, Band 138 der Reihe *Graduate Texts in Mathematics*. Springer–Verlag, 1995.

[Cop93] Coppersmith, Don: *Modifications to the Number Field Sieve*. Journal of Cryptology, 6:169–180, 1993.

[Cop97] Coppersmith, Don: *Small solutions to polynomial equations, and low exponent RSA vulnerabilities*. Journal of Cryptology, 10:233–260, 1997.

[Cox89] Cox, David A.: *Primes of the form $x^2 + ny^2$*. John Wiley & Sons, 1989.

[CS98] Cramer, Ronald und Victor Shoup: *A Practical Public Key Cryptosystem Provably Secure against Adaptive Chosen Ciphertext Attack*. In: Krawczyk, Hugo [**Kra98**], Seiten 13–25.

[CUS02] Chida, Koji, Shigenori Uchiyama und Taiichi Saito: *A New Factoring Method of Integers $N = p^r \times q$ for Large r*. IEICE Trans. Fundamentals, E85-A(5):1050–1053, 2002.

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

[Elk96] Elkenbracht-Huizing, R. Marije: *An Implementation of the Number Field Sieve*. Experimental Mathematics, 5:372–385, 1996.

[EMS+99] Elkenbracht-Huizing, R. Marije, Peter L. Montgomery, Robert D. Silverman, Richard Wackerbarth und Samuel S. Wagstaff: *The Number Field Sieve on Many Computers*. In: Gupta, Rajiv und Kenneth Williams (Herausgeber): *Number Theory, Ottawa, Ontario 1996*, Band 19 der Reihe *CRM Proceedings and Lectures Notes*. American Mathematical Society, 1999.

[ET02a] Ebinger, Peter und Edlyn Teske: *Factoring $N = PQ^2$ with the elliptic curve method*. In: Fieker, Claus und David R. Kohel [**FK02**].

[ET02b] Ebinger, Peter und Edlyn Teske: *On the Jacobi Symbol Continuation of ECM (Supplement to: Factoring $N = PQ^2$ with the elliptic curve method)*, 2002. http://www.math.uwaterloo.ca/ eteske/pqqAddendum.ps.

[FBK+] Franke, Jens, Friedrich Bahr, Thorsten Kleinjung, Peter L. Montgomery und year = 2003 note = http://ftp.cwi.nl/herman/SNFSrecords/SNFS-244 Herman te Riele, title = 244-digit SNFS factorization.

[FBK02] Franke, Jens, Friedrich Bahr und Thorsten Kleinjung: *Announcement: Factorization of $C158|2^{953} + 1$ using GNFS*, 2002. http://www.crypto-world.com/announcements/c158.txt.

[FBK+03] Franke, Jens, Friedrich Bahr, Thorsten Kleinjung, Peter L. Montgomery, Herman te Riele, Don Leclair, Paul Leyland und Richard Wackerbarth: *Announcement: Factorization of RSA-576 with GNFS*, 2003. http://www.crypto-world.com/announcements/rsa576.txt.

[FBKB03] Franke, Jens, Friedrich Bahr, Thorsten Kleinjung und M. Lochterand M. Böhm: *Announcement: Factorization of RSA-160 with GNFS*, 2003. http://www.crypto-world.com/announcements/rsa160.txt.

[FK02] Fieker, Claus und David R. Kohel (Herausgeber): *Algorithmic Number Theory, ANTS-V*, Band 2369 der Reihe *Lecture Notes in Computer Science*. Springer-Verlag, 2002.

[FO99] Fujisaki, Eiichiro und Tatsuaki Okamoto: *Secure Integration of Asymmetric and Symmetric Encryption Schemes*. In: Wiener, Michael [**Wie99**], Seiten 537–555.

[Gau01] Gauss, Carl Friedrich: *Disquisitiones Arithmeticae*. Springer–Verlag, 1801. Nachdruck, 1986, ISBN 0-387-96254-9.

[GBD+94] Geist, Al, Adam Beguelin, Jack Dongarra, Weicheng Jiang, Robert Manchek und Vaidy Sunderam: *PVM: Parallel Virtual Machine - a user's guide and tutorial for networked parallel computing*. 1994.

[Gir91] Girault, Marc: *An identity-based identification scheme based discrete logarithm modulo a composite number*. In: Damgård, Ivan (Herausgeber): *Advances in Cryptology – EUROCRYPT '90*, Band 473 der Reihe *Lecture Notes in Computer Science*, Seiten 481–486. Springer-Verlag, 1991.

[GMR88] Goldwasser, Shafi, Silvio Micali und Ron Rivest: *A Digital Signature Scheme Secure Against Adaptive Chosen-Message Attacks*. SIAM Journal of Computing, 17(2):281–308, 1988.

[Gor85] Gordon, John: *Strong primes are easy to find*. In: Beth, Thomas et al. [**BCI85**], Seiten 216–223.

[Gow03] Gower, Jason E.: *Rotations and Translations of Number Field Sieve Polynomials*. In: La, Chi Sung und Colin Boyd [**LB03**], Seiten 302–310.

[GQ90] Guillou, Louis C. und Jean-Jaques Quisquater: *A ParadoxalÏdentity-Based Signature Scheme Resulting from Zero-Knowledge*. In: Goldwasser, Shafi (Herausgeber): *Advances in Cryptology – CRYPTO '88*, Band 403 der Reihe *Lecture Notes in Computer Science*, Seiten 216–231. Springer-Verlag, 1990.

[GT03] Goldwasser, Shafi und Yael Tauman: *On the (In)security of the Fiat-Shamir Paradigm*. Cryptology ePrint Archive, Report 2003/034, 2003. http://eprint.iacr.org/.

[Ham02] Hamdy, Safuat: *Über die Sicherheit und Effizienz kryptographischer Verfahren mit Klassengruppen imaginär-quadratischer Zahlkörper*. Doktorarbeit, Technische Universität Darmstadt, Fachbereich Informatik, Darmstadt, 2002.

[HB03] Hamdy, Safuat und Mark L. Bauer: *On Class Group Computations Using the Number Field Sieve (Extended Abstract)*. In: La, Chi Sung und Colin Boyd [**LB03**], Seiten 311–325.

[HG97] Howgrave-Graham, Nick: *Finding small roots of univariate modular equations revisited*. In: Darnell, Michael (Herausgeber): *IMA Int. Conf.*, Band 1355 der Reihe *Lecture Notes in Computer Science*, Seiten 131–142. Springer-Verlag, 1997.

[HJPT98] Hühnlein, Detlef, Michael J. Jacobson, Jr., Sachar Paulus und Tsuyoshi Takagi: *A cryptosystem based on non-maximal imaginary quadratic orders with fast decryption*. In: Nyberg, Kaisa [**Nyb98**], Seiten 294–307.

[HJW01] Hühnlein, Detlef, Michael J. Jacobson, Jr. und Damian Weber: *Towards practical non-interactive cryptosystems using non-maximal imaginary quadratic orders (extended abstract)*. In: Stinson, Douglas R. und Stafford Tavares (Herausgeber): *Selected Areas in Cryptography, SAC 2000*, Lecture Notes in Computer Science, Seiten 275–287, 2001.

[HJW03] Hühnlein, Detlef, Michael J. Jacobson, Jr. und Damian Weber: *Towards practical non-interactive cryptosystems using non-maximal imaginary quadratic orders*. Designs, Codes and Cryptography, 30(3):281–299, 2003.

[HKL03] Harman, G., A. Kumchev und P. A. Lewis: *The distribution of prime ideals of imaginary quadratic fields*. erscheint in Transactions of the American Mathematical Society, 2003.

[HM00a] Hamdy, Safuat und Bodo Möller: *Security of Cryptosystems Based on Class Groups of Imaginary Quadratic Orders*. In: Matsumoto, Tsutomu (Herausgeber): *Advances in Cryptology – ASIACRYPT 2000*, Band 1976 der Reihe *Lecture Notes in Computer Science*, Seiten 234–247. Springer-Verlag, 2000.

[HM00b] Hühnlein, Detlef und Johannes Merkle: *An efficient NICE-Schnorr-type cryptosystem*. In: Imai, Hideki und Yuliang Zheng (Herausgeber): *Practice and Theory in Public Key Cryptography, PKC 2000*, Band 1751 der Reihe *Lecture Notes in Computer Science*, Seiten 14–27. Springer-Verlag, 2000.

[HMT98] Hühnlein, Detlef, Andreas Meyer und Tsuyoshi Takagi: *Rabin and RSA analogues based on non-maximal imaginary quadratic orders*. In: Rhee, Man Young und Hideki Imai [**RI98**], Seiten 221–240.

[HPT99] Hartmann, Michael, Sachar Paulus und Tsuyoshi Takagi: *NICE – New Ideal Coset Encryption*. In: Koç, Çetin K. und Christof Paar (Herausgeber): *Cryptographic Hardware and Embedded Systems, CHES '99*, Band 1717 der Reihe *Lecture Notes in Computer Science*, Seiten 328–339. Springer-Verlag, 1999.

[HT99] Hühnlein, Detlef und Tsuyoshi Takagi: *Reducing Logarithms in Totally Non-maximal Imaginary Quadratic Orders to Logarithms in Finite Fields*. In: Lam, Kwok Yun et al. [**LOX99**], Seiten 219–231.

[Hua82] Hua, Loo Keng: *Introduction to Number Theory*. Springer–Verlag, 1982.

[Hüh00] Hühnlein, Detlef: *Efficient implementation of cryptosystems based on non-maximal imaginary quadratic orders*. In: Heys, Howard und Carlisle Adams (Herausgeber): *Selected Areas in Cryptography, SAC '99*, Band 1758 der Reihe *Lecture Notes in Computer Science*, Seiten 150–167. Springer-Verlag, 2000.

[Hüh01] Hühnlein, Detlef: *Faster Generation of NICE-Schnorr-type Signatures*. In: Naccache, David [**Nac01**], Seiten 1–12.

[Hür94] Hürter, Michael: *Modifikationen zum Number Field Sieve*. Diplomarbeit, Universität des Saarlandes, 1994.

[Jac99] Jacobson, Michael J. Jr.: *Subexponential Class Group Computation in Quadratic Orders*. Doktorarbeit, Technische Universität Darmstadt, Fachbereich Informatik, Darmstadt, 1999.

[Jac00] Jacobson, Michael J. Jr.: *Computing discrete logarithms in quadratic orders*. Journal of Cryptology, 13(4):473–492, 2000.

[JJ00] Jaulmes, Éliane und Antoine Joux: *A NICE Cryptanalysis*. In: Preneel, Bart [**Pre00**], Seiten 382–391.

[JL03] Joux, Antoine und Reynald Lercier: *Improvements to the general number field sieve for discrete logarithms in prime fields. A comparison with the gaussian integer method*. Mathematics of Computation, 72:953–967, 2003.

[JN01] Joux, Antoine und Kim Nguyen: *Separating Decision Diffie-Hellman from Diffie-Hellman in cryptographic groups*. 2001. http://eprint.iacr.org.

[JW00] Jacobson, Michael J. Jr. und Hugh C. Williams: *The size of the fundamental solutions of consecutive Pell equations*. Experimental Mathematics, 9(4):631–640, 2000.

[Kim02] Kim, K. (Herausgeber): *The 4th International Conference on Information Security and Cryptology, ICISC '01*, Band 2288 der Reihe *Lecture Notes in Computer Science*. Springer-Verlag, 2002.

[Knu81] Knuth, Donald E.: *The Art of Computer Programming — Seminumerical Algorithms*, Band 2. Addison–Wesley, 2 Auflage, 1981.

[Kra98] Krawczyk, Hugo (Herausgeber): *Advances in Cryptology – CRYPTO '98*, Band 1462 der Reihe *Lecture Notes in Computer Science*. Springer-Verlag, 1998.

[Küg98] Kügler, Dennis: *Eine Aufwandsanalyse für identitätsbasierte Kryptosysteme*. Diplomarbeit, Technische Universität Darmstadt, Fachbereich Informatik, 1998. http://www.informatik.tu-darmstadt.de/pub/TI/reports/kuegler.IDCS.diplom.ps.gz.

[Lag80] Lagarias, Jeff C.: *Worst-case complexity bounds for algorithms in the theory of integral quadratic forms*. Journal of Algorithms, 1:142–186, 1980.

[Lan91] Lang, Serge: *Algebraic number theory*, Band 110 der Reihe *Graduate Texts in Mathematics*. Springer–Verlag, 2. Auflage, 1991.

[LB03] La, Chi Sung und Colin Boyd (Herausgeber): *Advances in Cryptology – ASIACRYPT 2003*, Band 2894 der Reihe *Lecture Notes in Computer Science*. Springer-Verlag, 2003.

[Len87] Lenstra, Hendrik W.: *Factoring integers with elliptic curves*. Annals of Mathematics, 126:649–673, 1987.

[Len01] Lenstra, Arjen K.: *Unbelievable Security. Matching AES Security Using Public Key Systems*. In: Boyd, Colin (Herausgeber): *Advances in Cryptology – ASIACRYPT 2001*, Band 2248 der Reihe *Lecture Notes in Computer Science*, Seiten 67–86. Springer-Verlag, 2001.

[LiD] *LiDIA – A C++ Library For Computational Number Theory*. http://www.informatik.tu-darmstadt.de/TI/LiDIA/.

[LL93a] Lenstra, A.K. und H.W. Lenstra (Herausgeber): *The Developement of the Number Field Sieve*, Band 1554 der Reihe *Lecture Notes in Mathematics*. Springer-Verlag, 1993.

[LL93b] Lim, Chae Hoon und Pil Joong Lee: *Modified Maurer-Yacobi's scheme and its applications*. In: Seberry, Jennifer und Zheng Yuliang (Herausgeber): *Advances in Cryptology – AUSCRYPT '92*, Band 718 der Reihe *Lecture Notes in Computer Science*, Seiten 308–323. Springer-Verlag, 1993.

[LLD+02] Leyland, Paul C., Arjen K. Lenstra, Bruce Dodson, Alec Muffett und Sam Wagstaff: *MPQS with Three Large Primes*. In: Fieker, Claus und David R. Kohel [**FK02**], Seiten 446–460.

[LLMP93] Lenstra, Arjen K., Hendrik W. Lenstra, Mark S. Manasse und John M. Pollard: *The number field sieve*. In: Lenstra, A.K. und H.W. Lenstra [**LL93a**], Seiten 11–42.

[LLWG03a] Leyland, Paul, Don Leclair, Richard Wackerbarth und Jeff Gilchrist: *Announcement: Factors of* $2^{713} - 1$, 2003. http://www.nfsnet.org/ann-2_713M_1.html.

[LLWG03b] Leyland, Paul, Don Leclair, Richard Wackerbarth und Jeff Gilchrist: *Announcement: Factors of* $2^{757} - 1$, 2003. http://www.nfsnet.org/ann-2_757M_1.html.

[LOX99] Lam, Kwok Yun, Eiji Okamato und Chaoping Xing (Herausgeber): *Advances in Cryptology – ASIACRYPT '99*, Band 1716 der Reihe *Lecture Notes in Computer Science*. Springer-Verlag, 1999.

[LPW95] Lukes, R.F., C.D. Patterson und H.C. Williams: *Numerical sieving devices: Their history and some applications*. In: *Nieuw Archief voor Wiskunde (4)*, Band 13, Seiten 113–139. 1995.

[LSTT02] Lenstra, Arjen, Adi Shamir, Jim Tomlinson und Eran Tromer: *Analysis of Bernstein's Factorization Circuit*. In: Zheng, Y. (Herausgeber): *Advances in Cryptology – ASIACRYPT 2002*, Band 2501 der Reihe *Lecture Notes in Computer Science*, Seiten 1–26. Springer-Verlag, 2002.

[LTS+03] Lenstra, Arjen, Eran Tromer, Adi Shamir, Wil Kortsmit, Bruce Dodson, James Hughes und Paul Leyland: *Factoring estimates for a 1024-bit RSA modulus*. In: La, Chi Sung und Colin Boyd [**LB03**], Seiten 55–74.

[LV01] Lenstra, Arjen K. und Eric R. Verheul: *Selecting Cryptographic Keysizes*. Journal of Cryptology, 14(4):255–293, 2001.

[Mao98] Mao, Wenbo: *Cryptoanalysis in Prime Order Subgroups of* $\mathbb{Z}_n^*$. IEEE Working Group P1363, 1998. http://www.ieee.org.

[McK99] McKee, James F.: *Subtleties in the distribution of the numbers of points on elliptic curves over a finite field*. Journal of the London Mathematical Society, (2) 59:448–460, 1999.

[Mey97] Meyer, Andreas: *Ein neues Identifikations- und Signaturverfahren über imaginärquadratischen Zahlkörpern*. Diplomarbeit, Technische Universität Darmstadt, Fachbereich Informatik, 1997. via http://www.informatik.tu-darmstadt.de/TI/Welcome.html.

[MK99] Maurer, Markus und Dennis Kügler: *A note on the weakness of the Maurer-Yacobi squaring method*. Technischer Bericht TI-15/99, Technische Universität Darmstadt, Fachbereich Informatik, 1999. http://www.informatik.tu-darmstadt.de/TI/Veroeffentlichung/TR/.

[ML98] Mao, Wenbo und Chae Hoon Lim: *Cryptoanalysis in Prime Order Subgroups of* $\mathbb{Z}_n^*$. In: Ohta, Kazuo und Pei Dingyi (Herausgeber): *Advances in Cryptology – ASIACRYPT '98*, Band 1514 der Reihe *Lecture Notes in Computer Science*, Seiten 214–226. Springer-Verlag, 1998.

[Mon92] Montgomery, Peter L.: *An FFT extension of the elliptic curve method of factorization*. Doktorarbeit, University of California, Los Angeles, 1992. ftp://ftp.cwi.nl/pub/pmontgom/ucladissertation.psl.gz.

[Mon94] Montgomery, Peter L.: *Square roots of products of algebraic numbers*. In: Gautschi, W. (Herausgeber): *Mathematics of Computation 1943-1993, Fifty Years of Computational Mathematics*, Proceedings of Symposia in Applied Mathematics, Seiten 567–571. American Mathematical Society, 1994.

[Mor90] Morain, Francois: *Courbes elliptiques et tests de primalit*. Doktorarbeit, Universit Claude Bernard, Lyon, Frankreich, 1990. ftp://ftp.inria.fr/INRIA/publication/Theses/TU-0144.tar.z.

[Mül95] Müller, Andreas: *Eine FFT-Continuation für die elliptische Kurvenmethode*. Diplomarbeit, Universität des Saarlandes, 1995.

[Mur99] Murphy, Brian A.: *Polynomial Selection for the Number Field Sieve Integer Factorisation Algorithm*. Doktorarbeit, Australian National University, Canberra, Australien, 1999. http://web.comlab.ox.ac.uk/oucl/work/richard.brent/ftp/Murphy-thesis.ps.gz.

[MvOV97] Menezes, Alfred J., Paul C. van Oorschot und Scott A. Vanstone: *Handbook of Applied Cryptography*. CRC Press, 1997.

[MY91] Maurer, Ueli und Yacov Yacobi: *Non-interactive public-key cryptography*. In: Davies, Donald W. (Herausgeber): *Advances in Cryptology – EUROCRYPT '91*, Band 547 der Reihe *Lecture Notes in Computer Science*, Seiten 498–507. Springer-Verlag, 1991.

[MY93] Maurer, Ueli und Yacov Yacobi: *A remark on a non-interactive public-key distribution system*. In: Rueppel, Rainer A. (Herausgeber): *Advances in Cryptology – EUROCRYPT '92*, Band 658 der Reihe *Lecture Notes in Computer Science*, Seiten 458–460. Springer-Verlag, 1993.

[MY96] Maurer, Ueli und Yacov Yacobi: *A non-interactive public-key distribution system*. Designs, Codes and Cryptography, 9:305–316, 1996.

[Nac01] Naccache, David (Herausgeber): *Progress in Cryptology – CT-RSA 2001*, Band 2020 der Reihe *Lecture Notes in Computer Science*. Springer-Verlag, 2001.

[Nar74] Narkiewicz, Wladyslaw: *Elementary and analytic theory of algebraic numbers*. Polish Scientific Publishers, Warszawa, 1974.

[Neu92] Neukirch, Jürgen: *Algebraische Zahlentheorie*. Springer–Verlag, 1992.

[Ngu98] Nguyen, Phong: *A Montgomery-like square root for the number field sieve*. In: Buhler, Joe P. (Herausgeber): *Algorithmic Number Theory, ANTS-III*, Band 1423 der Reihe *Lecture Notes in Computer Science*, Seiten 151–168. Springer-Verlag, 1998.

[Nie02] Nielsen, Jesper Buus: *Separating Random Oracle Proofs from Complexity Theoretic Proofs: The Non-committing Encryption Case*. In: Yung, Moti (Herausgeber): *Advances in Cryptology – CRYPTO 2002*, Band 2442 der Reihe *Lecture Notes in Computer Science*, Seiten 111–126. Springer-Verlag, 2002.

[NIS94] NIST – National Institute of Standards and Technology: *Digital Signature Standard (DSS)*. Federal Information Processing Standards Publication, 186, 1994.

[Nyb98] Nyberg, Kaisa (Herausgeber): *Advances in Cryptology – EUROCRYPT '98*, Band 1403 der Reihe *Lecture Notes in Computer Science*. Springer-Verlag, 1998.

[Odl95] Odlyzko, Andrew M.: *The Future of Integer Factorization*. CryptoBytes, 1(2), 1995. ftp://ftp.rsasecurity.com/pub/cryptobytes/crypto1n2.pdf.

[OP01] Okamoto, Tatsuaki und David Pointcheval:: *REACT: Rapid Enhanced-security Asymmetric Cryptosystem Transform*. In: Naccache, David [**Nac01**], Seiten 159–175.

[OU98] Okamoto, Tatsuaki und Shigenori Uchiyama: *Security of an identity based cryptosystem and related reductions*. In: Nyberg, Kaisa [**Nyb98**], Seiten 546–560.

[Per01] Peralta, Ren: *Elliptic curve factorization using a partially obliviousfunction*. In: Lam, K.-Y., I. Shparlinski, H. Wang und C. Xing (Herausgeber): *Cryptography and Computational Number Theory: Workshop in Singapore 1999*, Band 20 der Reihe *Progress in Computer Science and Applied Logic*. Birkhäuser, 2001.

[PH78] Pohlig, S.C. und Martin E. Hellman: *An improved algorithm for computing logarithms over $GF(p)$ and it's cryptographic significance*. IEEE Transactions on Information Theory, 24:106–110, 1978.

[PO96] Peralta, Renè und Eiji Okamoto: *Faster factoring of integers of a special form*. IEICE Transactions on Fundamentals of Electronics, Communications, and Computer Sciences, E79-A:489–493, 1996.

[Poi02] Pointcheval, David: *Practical Security in Public-Key Cryptography*. In: Kim, K. [**Kim02**], Seiten 1–17.

[Pol74] Pollard, John M.: *Theorems on Factorization and Primality Testing*. In: *Proceedings of Cambridge Philosophy Society*, Band 76. Cambridge University Press, 1974.

[Pol75] Pollard, John M.: *A Monte Carlo method for factorization*. BIT, 15:331–334, 1975.

[Pol78] Pollard, John M.: *Monte Carlo methods for index computation (mod p)*. Mathematics of Computation, 32(143):918–924, 1978.

[Pol93] Pollard, John M.: *Factoring with cubic integers*. In: Lenstra, A.K. und H.W. Lenstra [**LL93a**], Seiten 4–10.

[Pom85] Pomerance, Carl: *The Quadratic Sieve Factoring Algorithm*. In: Beth, Thomas et al. [**BCI85**], Seiten 169–182.

[Pre00] Preneel, Bart (Herausgeber): *Advances in Cryptology – EUROCRYPT 2000*, Band 1807 der Reihe *Lecture Notes in Computer Science*. Springer-Verlag, 2000.

[PS98] Poupard, Guillaume und Jaques Stern: *Security Analysis of a Practical "on the fly" Authentication and Signature Generation*. In: Nyberg, Kaisa [**Nyb98**], Seiten 422–436.

[PT98] Paulus, Sachar und Tsuyoshi Takagi: *A generalization of the Diffie-Hellman Problem and related cryptosystems allowing fast decryption*. In: Rhee, Man Young und Hideki Imai [**RI98**], Seiten 211–220.

[PT00] Paulus, Sachar und Tsuyoshi Takagi: *A New Public-Key Croptosystem over Quadratic Orders with Quadratic Decryption Time*. Journal of Cryptology, 13(2):263–272, 2000.

[RI98] Rhee, Man Young und Hideki Imai (Herausgeber): *The 1st International Conference on Information Security and Cryptology, ICISC '98*. DongKwang Publishing Company, Korea, 1998.

[Rib95] Ribenboim, P.: *The new book on prime number records*. Springer–Verlag, 1995.

[SA98] Satoh, T. und K. Araki: *Fermat quotients and the polynomial time discrete log algorithm for anomalous elliptic curves*. Comm. Math. Univ. Sancti. Pauli, 47:81–92, 1998.

[Sch82] Schoof, Rene J.: *Quadratic fields and factorization*. In: Lenstra, Jr., Hendrik W. und Robert Tijdeman (Herausgeber): *Computational methods in number theory II*, Band 155 der Reihe *Mathematical Centre Tracts*, Seiten 235–286. Mathematisch Centrum, 1982.

[Sch90] Schnorr, Claus P.: *Efficient identification and signatures for smart cards*. In: Brassard, Gilles [**Bra90**], Seiten 239–252.

[Sch99] Schaub, Joachim: *Implementierung von Public-Key-Kryptosystemen über imaginärquadratischen Ordnungen*. Diplomarbeit, Technische Universität Darmstadt, Fachbereich Informatik, 1999.

[Sem98] Semaev, I. A.: *Evaluation of Discrete Logarithms in a Group of p-Torsion Points on an Elliptic Curve in Characteristic p*. Mathematics of Computation, 67(221):353–356, 1998.

[Sha71] Shanks, Daniel: *Gauss' ternary form reduction and the 2-Sylow subgroup*. Mathematics of Computation, 25:837–853, 1971.

[Sha72] Shanks, Daniel: *The infrastructure of a real quadratic field and its applications*. In: *Proceedings of Number Theory Conference, Boulder 1972*, Seiten 217–224, 1972.

[Sha85] Shamir, Adi: *Identity based cryptosystems and signature schemes*. In: Blakley, G. R. und David Chaum (Herausgeber): *Advances in Cryptology*, Band 196 der Reihe *Lecture Notes in Computer Science*, Seiten 47–53. Springer-Verlag, 1985.

[Sil87] Silverman, Robert D.: *The multiple polynomial quadratic sieve*. Mathematics of Computation, 48:329–339, 1987.

[Sil00] Silverman, Robert D.: *A cost-based security analysis of symmetric and asymmetric key lengths*. RSA Laboratories Bulletin, 13, 2000. http://www.rsasecurity.com/rsalabs/bulletins/bulletin13.html.

[SL84] Schnorr, Claus Peter und Hendrik W. Lenstra, Jr.: *A Monte Carlo Factoring Algorithm With Linear Storage*. Mathematics of Computation, 43(167):289–311, 1984.

[Sma99] Smart, Nigel P.: *The Discrete Logarithm Problem on Elliptic Curves of Trace One*. Journal of Cryptology, 12(3):193–196, 1999.

[SS02] Schmidt-Samoa, Katja: *Das Number Field Sieve – Entwicklung, Varianten und Erfolge*. Diplomarbeit, Universität Kaiserslautern, 2002. http://www.informatik.tu-darmstadt.de/KP/staff/samoa/NFS.pdf.

[ST03] Shamir, Adi und Eran Tromer: *Factoring Large Numbers with the TWIRL Device*. In: Boneh, Dan (Herausgeber): *Advances in Cryptology – CRYPTO 2003*, Band 2729 der Reihe *Lecture Notes in Computer Science*, Seiten 1–26. Springer-Verlag, 2003.

[Tak98] Takagi, Tsuyoshi: *Fast RSA-Type Cryptosystem Modulo $p^k q$*. In: Krawczyk, Hugo [**Kra98**], Seiten 318–326.

[Tak01] Takagi, Tsuyoshi: *New public-key cryptosystems with fast decryption*. Doktorarbeit, Technische Universität Darmstadt, Fachbereich Informatik, Darmstadt, 2001. http://elib.tu-darmstadt.de/diss/000104/.

[Tes98] Teske, Edlyn: *New Algorithms for Finite Abelian Groups*. Doktorarbeit, Technische Universität, Darmstadt, 1998. ISBN 3-8265-4045-X.

[UK01] Uchiyama, Shigenori und Naoki Kanayama: *Speeding up the Lattice Factoring Method*. IEICE Trans. Fundamentals, E84-A(1):146–150, 2001.

[Vol00] Vollmer, Ulrich: *Asymptotically Fast Discrete Logarithms in Quadratic Number Fields*. In: Bosma, Wieb (Herausgeber): *Algorithmic Number Theory, ANTS-IV*, Band 1838 der Reihe *Lecture Notes in Computer Science*, Seiten 581–594. Springer-Verlag, 2000.

[vOW96] Oorschot, Paul C. van und Michael J. Wiener: *On Diffie-Hellman Key Agreement with Short Exponents*. In: Maurer, Ueli (Herausgeber): *Advances in Cryptology – EUROCRYPT '96*, Band 1070 der Reihe *Lecture Notes in Computer Science*, Seiten 332–343. Springer-Verlag, 1996.

[Web96] Weber, Damian: *Computing discrete logarithms with the number field sieve*. In: Cohen, Henri (Herausgeber): *Algorithmic Number Theory, ANTS-II*, Band 1122 der Reihe *Lecture Notes in Computer Science*. Springer-Verlag, 1996.

[Web98] Weber, Damian: *The solution of McCurley's discrete log challenge*. In: Krawczyk, Hugo [**Kra98**], Seiten 56–60.

[Wie99] Wiener, Michael (Herausgeber): *Advances in Cryptology – CRYPTO '99*, Band 1666 der Reihe *Lecture Notes in Computer Science*. Springer-Verlag, 1999.

[Wil82] Williams, Hugh C.: *A $p+1$ Method of Factoring*. Mathematics of Computation, 39:225–234, 1982.

[Zag81] Zagier, Don B.: *Zetafunktionen und quadratische Körper*. Springer–Verlag, 1981.

Curriculum Vitae

Werdegang

11.10.1971	geboren in Kronach
09.1977 - 07.1983	Grund- und Teilhauptschule Marktzeuln
09.1983 - 07.1987	Realschule Burgkunstadt
08.1987 - 09.1990	Berufsausbildung als Landwirt
09.1990 - 07.1991	Fachoberschule Kulmbach
10.1991 - 12.1995	Studium der Informatik an der Fachhochschule Würzburg
09.1995 - 10.1997	Freier Mitarbeiter bei Siemens AG, Würzburg Mannesmann Sachs AG, Schweinfurt und secunet Security Networks GmbH, Eschborn
06.1996 - 06.1997	Wissenschaftlicher Mitarbeiter bei GMD GmbH, Darmstadt
11.1997 -	Unternehmensberater bei der secunet Security Networks AG, Eschborn und Essen
03.2000 -	Lehrbeauftragter an den Fachhochschulen Darmstadt und Würzburg

Publikationen

Artikel in internationalen Fachzeitschriften

1. *Towards practical non-interactive public key cryptosystems using non-maximal imaginary quadratic orders*, in Designs, Codes and Cryptography, 30, Kluwer, 2003, Seiten 281-299, (zusammen mit M.J. Jacobson, D. Weber)

Konferenzbeiträge in der LNCS-Reihe

2. *A cryptosystem based on non-maximal imaginary quadratic orders with fast decryption*, Advances in Cryptology, Proceedings of EUROCRYPT '98, LNCS 1403, Springer, 1998, Seiten 249-307, (zusammen mit M.J. Jacobson, S. Paulus und T. Takagi)
3. *Efficient implementation of cryptosystems based on non-maximal imaginary quadratic orders*, in Proceedings of SAC 1999, LNCS 1758, Springer, 2000, Seiten 150-167
4. *Reducing logarithms in totally non-maximal imaginary quadratic orders to logarithms in finite fields*, in Proceedings of ASIACRYPT'99, LNCS 1716, Springer, 1999, Seiten 219-231, (zusammen mit T. Takagi)
5. *Secure and cost efficient electronic stamps*, in Proceedings of CQRE 1999, LNCS 1740, Springer, 1999, Seiten 94-100, (zusammen mit J. Merkle)
6. *An efficient NICE-Schnorr-type signature scheme*, in Proceedings of PKC 2000, Melbourne, LNCS 1751, Springer, 2000, Seiten 14-27, (zusammen mit J. Merkle)

7. *Towards practical non-interactive public key cryptosystems using non-maximal imaginary quadratic orders (extended abstract)*, in Proceedings of SAC 2000, LNCS 2012, 2001, Springer, Seiten 275-287, (zusammen mit M.J. Jacobson, D. Weber)
8. *On the implementation of cryptosystems based on real quadratic number fields*, in Proceedings of SAC 2000, LNCS 2012, 2001, Springer, Seiten 288-302, (zusammen mit S. Paulus)
9. *Faster generation of NICE-Schnorr-Signatures*, in Topics in Cryptology - CT-RSA 2001, The Cryptographer's Track at RSA Conference 2001, LNCS 2020, Springer, 2001, Seiten 1-12
10. *How to qualify electronic signatures and time stamps*, Proceedings of 1st European PKI Workshop, Samos, LNCS 3093, Springer, 2004, Seiten 314-321

Weitere Beiträge zu internationalen Konferenzen

11. *Credential Management and Secure Single Login for SPKM*, Proceedings of ISOC - Symposium on Network- and Distributed System Security, San Diego, 13. - 15. März 1998, ISBN 1-891562-03-7, 1998, Seiten 76-88
12. *Rabin and RSA analogues based on non-maximal imaginary quadratic orders*, Proceedings of ICICS'98, ISBN 89-85305-14-X, 1998, Seiten 221-240, (zusammen mit A. Meyer und T. Takagi)

Sonstige referierte Fachpublikationen

13. *Effiziente Exponentiation und optimale Punktdarstellung für Signatursysteme auf Basis elliptischer Kurven*, in Horster P. (Hrsg.): Tagungsband Digitale Signaturen, DuD-Fachbeiträge, Vieweg, 1996, Seiten 221-236
14. *On the development of a security toolkit for open networks - New security features in SECUDE*, Tagunsband VIS 97, Vieweg, 1997, (zusammen mit U. Faltin, P. Glöckner, U. Viebeg, A. Berger, H. Giehl, S. Andre und T. Surkau)
15. *Implementierung Elliptischer Kurven auf Chipkarten*, in Horster P. (Hrsg.): Tagungsband Chipkarten, DuD-Fachbeiträge, Vieweg, ISBN 3-528-05667-3, 1998
16. *HBCI - Eine sichere Plattform nicht nur für Homebanking*, in Horster P. (Hrsg.): Tagungsband Sicherheitsinfrastrukturen, DuD-Fachbeiträge, Vieweg, ISBN 3-528-05709-2, 1999
17. *A survey of cryptosystems based on imaginary quadratic orders*, in Horster P. (Hrsg.): Tagungsband Systemsicherheit, DuD-Fachbeiträge, Vieweg, 2000, Seiten 370-384
18. *Elliptische Kurven in HBCI - Ein Backup zu RSA*, in Horster P. (Hrsg.): Tagungsband Systemsicherheit, DuD-Fachbeiträge, Vieweg, 2000, Seiten 82-92
19. *Elektronische Zahlungsmechanismen im Überblick*, in Horster P. (Hrsg.): Tagungsband Kommunikationssicherheit im Zeichen des Internet, DuD-Fachbeiträge, Vieweg, 2001, Seiten 180-191, (zusammen mit A. Hitzbleck)
20. *Global Secure E-Mail Interoperability - The Europe-based Bridge-CA initiative*, in Horster P. (Hrsg.): Tagungsband Elektronische Geschäftsprozesse, IT-Verlag, ISBN 3-936052-00-X, 2001, Seiten 22-31, (zusammen mit B. Esslinger)

21. *Public-Key Infrastrukturen für Finanzdienstleister*, in Horster P. (Hrsg.): Tagungsband Elektronische Geschäftsprozesse, IT-Verlag, ISBN 3-936052-00-X, 2001, Seiten 155-167, (zusammen mit A. Jaletzke)
22. *Identrus-enabled applications*, in Horster P. (Hrsg.): Tagungsband Elektronische Geschäftsprozesse, IT-Verlag, ISBN 3-936052-00-X, 2001, Seiten 181-192
23. *Public-Key Infrastrukturen in der Phase der Konsolidierung und Anwendung*, in Horster P. (Hrsg.): Tagungsband Sicherheitsinfrastrukturen in Wirtschaft und Verwaltung, IT-Verlag, 2002, ISBN 3-936052-04-02, Seiten 365-382
24. *IT-Risikomanagement bei Unternehmensfusionen*, in Horster P. (Hrsg.): Tagungsband Enterprise Security, IT-Verlag, ISBN 3-936052-03-4, 2002, Seiten 177-187
25. *Public-Key Infrastrukturen für Sozialversicherungsträger*, in Horster P. (Hrsg.): Tagungsband Elektronische Geschäftsprozesse 2002, IT-Verlag, ISBN 3-936052-07-7, 2002, Seiten 177-198
26. *Towards Harmonizing Identrus with the European Signature Legislation*, in Horster P. (Hrsg.): Tagungsband Elektronische Geschäftsprozesse, IT-Verlag, ISBN 3-936052-07-7, 2002, Seiten 311-319, (zusammen mit I. Alamillo)
27. *Aspekte der Massensignatur*, in Horster P. (Hrsg.): Tagungsband DACH Security, IT-Verlag, ISBN 3-00-010941-2, 2003, Seiten 293-307, (zusammen mit Y. Knosowski)
28. *Rechtliche Betrachtungen zur automatisierten Erzeugung qualifizierter elektronischer Signaturen gemäß §14 UStG und §36 SRVwV*, Proceedings of the 1st Conference on Biometrics and Electronic Signatures of the GI Working Group BIOSIG, LNI 31 GI, ISBN 3-88579-360-1, 2003, Seiten 101-112, (zusammen mit Y. Knosowski und R. Tern)
29. *Intervall-qualifizierte Zeitstempel*, in Horster P. (Hrsg.): Tagungsband Elektronische Geschäftsprozesse, IT-Verlag, ISBN 3-00-014186-3, 2004, Seiten 431-445
30. *Juristische Aspekte der elektronischen Abrechnung*, in Horster P. (Hrsg.): Tagungsband Elektronische Geschäftsprozesse, IT-Verlag, ISBN 3-00-014186-3, 2004, Seiten 134-143, (zusammen mit R. Tern)