Kapitel 5
Verfahren mit diskreten Logarithmen

Die Sicherheit des RSA-Verfahrens hängt zusammen mit der Schwierigkeit der Zerlegung großer Zahlen in ihre Primfaktoren; direkt geht es allerdings um das Problem, aus $x^e \bmod N$ auf den Wert von x zu schließen, also die e-te Wurzel modulo N aus einer Zahl zu ziehen. Für eine Primzahl N ist das für die Exponenten, die wir bei RSA verwenden, problemlos möglich; für eine zusammengesetzte Zahl aber geht es anscheinend nur mit Kenntnis von deren Faktorisierung.

Eine zweite vor allem für elektronische Unterschriften populäre Gruppe von Verfahren baut stattdessen auf die Schwierigkeit, aus der Kenntnis von $a^x \bmod N$ bei bekanntem a auf den Wert von x zu schließen; hier geht es also um die Umkehrung einer modularen Exponentialfunktionen deren Umkehrfunktion bezeichnet man in Analogie zur Umkehrfunktion einer reellen Exponentialfunktion als diskreten Logarithmus (oder Index). Effiziente Verfahren, um ihn auch für große N sind selbst für große Primzahlen N nicht bekannt; da man diskrete Logarithmen modulo einer zusammengesetzten Zahl N leicht nach dem chinesischen Restesatz über die diskreten Logarithmen modulo der Primteiler von N berechnen kann, wählt man bei Kryptoverfahren auf der Basis diskreter Logarithmen N stets als eine Primzahl p.

Wir beginnen mit dem ältesten Beispiel eines solchen Verfahrens:

§1: Schlüsselaustausch nach Diffie und Hellman

Wie wir im letzten Kapitel gesehen haben, waren DIFFIE und HELLMAN mit ihrer Arbeit *New directions in cryptography* die Initiatoren der

asymmetrischen Kryptographie in der akademischen Welt; kurz nach dieser Arbeit entwickelten sie auch ein entsprechendes Verfahren, das zwar nicht zur Verschlüsselung dienen konnte, aber dafür die bis heute wichtigste Aufgabe der Kryptographie mit öffentlichen Schlüssel lösen konnte: Die Vereinbarung eines Schlüssels über eine unsichere Leitung.

Im Gegensatz zum RSA-Verfahren brauchen sie dazu nicht einmal öffentliche Schlüssel: Die beiden Teilnehmer können miteinander sicher kommunizieren ohne zuvor irgendwelche öffentlichen oder privaten Schlüssel zu kennen. Damit ist dieses Verfahren vor allem interessant im privaten Bereich, wo zertifizierte öffentliche Schlüssel oder gelegentlich auch überhaupt die Speicherung von Schlüsseln zu aufwendig wäre.

a) Das Verfahren

Die beiden Teilnehmer einigen sie sich zunächst (über die unsichere Leitung) auf eine Primzahl p und eine natürliche Zahl a derart, daß die Potenzfunktion $x \mapsto a^x$ möglichst viele Werte annimmt. Als nächstes wählt Teilnehmer A eine Zufallszahl $x < p$ und B entsprechend ein $y < p$. A schickt $u = a^x \bmod p$ an B und erhält dafür $y = a^y \bmod p$ von diesem. Sodann berechnet A die Zahl

$$v^x \bmod p = \left(a^y\right)^x \bmod p = a^{xy} \bmod ap$$

und B entsprechend

$$u^y \bmod p = \left(a^x\right)^y \bmod p = a^{xy} \bmod p\,;$$

beide haben also auf verschiedene Weise dieselbe Zahl berechnet, die sie zum Beispiel verwenden können, um daraus einen Schlüssel für ein symmetrisches Kryptosystem zu bestimmen. Verfahren dazu gibt es mehr als genug: Sie könnten etwa die letzten oder sonst irgendwelche Bits dieser Zahl verwenden, aber auch einen irgendwie definierten Hashwert.

Ein Gegner, der den Datenaustausch abgehört hat, kennt die Zahlen p, a, u und v; er kann also problemlos alle möglichen Zahlen modulo p der Art $a^{\alpha x+\beta y} = u^\alpha \cdot v^\beta$ berechnen. Es fällt aber schwer, sich eine Art und Weise vorzustellen, wie er $a^{xy} \bmod p$ finden kann, ohne den diskreten Logarithmus von u oder v zu berechnen. (Bewiesen ist hier, wie üblich, natürlich nichts.)

b) Die *man in the middle attack*

Da der Gegner die Wahl der Mittel hat, muß er nicht notwendigerweise die mathematische Seite des Verfahrens angreifen: Er kann angreifen, was immer er für eine vielversprechende Schwachstelle hält.

Nehmen wir etwa an, der Gegner habe eine gewisse Kontrolle über das Netz, in dem der Datenaustausch stattfindet – beispielsweise, weil er Systemverwalter eines für die betreffende Verbindung unbedingt notwendigen Knotenrechners ist. Dann kann er eine sogenannte *man in the middle attack* durchführen: Er fängt alle Datenpakete zwischen A und B ab und ersetzt sie durch selbstfabrizierte eigene Pakete.

Damit kann er sich gegenüber A als B auszugeben und umgekehrt: Alles, was A an B zu schicken glaubt, geht tatsächlich an den Gegner G, und alles was B von A zu erhalten glaubt, kommt tatsächlich von G. In Gegenrichtung ist es natürlich genauso.

Im einzelnen läuft der Angriff folgendermaßen ab:

Falls die Zahlen a und p nicht ohnehin Konstanten eines Verbunds sind, dem A und B angehören, läßt G die Kommunikation, die zu deren Vereinbarung führt, ungehindert zu: In diesem Stadium beschränkt er sich auf reines Abhören.

Als nächstes wählen A und B ihre Zufallszahlen $x < p$ und $y < p$; gleichzeitig wählt G eine Zufallszahl $z < p$ oder vielleicht auch zwei verschiedene solche Zahlen z_A und z_B für die beiden Teilnehmer.

Wenn A die Zahl $u = a^x \bmod p$ an B schickt, fängt G diese Nachricht ab und ersetzt sie durch $w_B = a^{z_B} \bmod p$; entsprechend fängt er Bs Nachricht $y = a^y \bmod p$ ab und schickt stattdessen $w_A = a^{z_A}$ an A. Dies führt dazu, daß am Ende A und G einen gemeinsamen Schlüssel s_A haben und B und G einen gemeinsamen Schlüssel s_B. Sowohl A als auch B glauben, der ihnen bekannte Schlüssel s_A *bzw.* s_B sei aus $a^{xy} \bmod p$ abgeleitet und senden nun damit verschlüsselte Nachrichten an ihren Partner. Diese Nachrichten fängt G ab, entschlüsselt sie mit dem Schlüssel, den er mit dem Absender gemeinsam hat, und verschlüsselt sie anschließend, gegebenenfalls nach einer seinen Interessen entsprechenden Modifikation, mit dem Schlüssel, den er mit dem Empfänger gemeinsam hat. Auf

diese Weise hat er die gesamte Konversation unter Kontrolle, ohne daß A und B etwas merken.

Die Möglichkeit für diese Attacke kommt natürlich daher, daß sich A und B nicht sicher sein können, den jeweils anderen am anderen Ende der Leitung zu haben. Die kryptographisch einwandfreie Modifikation, die das Verfahren gegen diese Art von Angriff sicher macht, bestünde beispielsweise darin, daß A und B ihre Nachrichten x und y vor dem Versenden unterschreiben – aber dann verschwindet auch wieder der Vorteil, daß sie ohne Kenntnis irgendeines Schlüssels miteinander kommunizieren können: Zur Verifikation einer Unterschrift braucht man schließlich den öffentlichen Schlüssel des Unterschreibenden.

Falls sich A und B hinreichend gut kennen, um die Stimme des jeweils anderen am Telephon einigermaßen sicher zu erkennen, können sie diese Art von Attacke auch dadurch erschweren, daß sie nach dem Austausch von u und v per Telephon über diese Zahlen (z.B. die 317. bis 320. Ziffer) und gegebenenfalls auch noch über Schwänke aus ihrer gemeinsamen Jugendzeit reden; dann müßte der Angreifer zusätzlich noch ein begabter, kundiger und reaktionsschneller Stimmenimitator sein, der auch die Telephonverbindung als *man in the middle* so angreifen kann, daß weder A noch B etwas merkt. Bei Videokonferenzen könnte man auch die Zahlen langsam über den Bildschirm des jeweils anderen laufen lassen. Die volle Sicherheit einer Schlüsselvereinbarung via RSA wird aber nicht erreicht, und da oft zumindest einer der Teilnehmer ein Unternehmen ist, das sich einen zertifizierten RSA-Schlüssel leisten kann, werden Schlüssel für symmetrische Kryptoverfahren in der Praxis sehr viel häufiger via RSA vereinbart als via DIFFIE-HELLMAN.

Im *electronic banking* wird die Idee eines zweiten Kommunikationskanals trotzdem häufig angewandt, hier im allgemeinen dadurch, daß ein Teil des Protokolls via SMS abläuft. Auch die können zwar selbstverständlich manipuliert werden, aber der Aufwand eines Angreifers steigt ganz beträchtlich, wenn er *gleichzeitig* zwei verschiedene Verbindungen manipulieren muß: Die meisten *phishing*-Attacken arbeiten schließlich mit gefälschten Webseiten im Ausland, und selbst im Inland ist es nicht so einfach, Mobilfunkverbindungen in einem größeren Gebiet zu überwachen und zu manipulieren.

§2: Verschlüsselung und elektronische Unterschriften

Zwischen RSA und den Verfahren mit diskreten Logarithmen gibt es einen ganz wesentlichen Unterschied: Wer die Faktorisierung des RSA-Moduls N kennt, kann die sonst schwer zugängliche Umkehrfunktion von $x \mapsto x^e \bmod N$ leicht berechnen, so daß Potenzieren mit e direkt als Verschlüsselung benutzt werden kann.

Bei der modularen „Exponentialfunktion“ $x \mapsto a^x \bmod p$ sind keine speziellen Wahlen von a und p bekannt, die vermöge einer geheimen Information zu einer einfachen Umkehrfunktion führen – diskrete Logarithmen sind für alle gleich schwer zu berechnen.

Die geheime Information bei einem asymmetrischen Verfahren auf der Basis diskreter Logarithmen kann daher nur in der Kenntnis *einzelner* diskreter Logarithmen bestehen: Wer für einen speziellen Wert x die Potenz $u = a^x \bmod p$ berechnet hat, weiß anschließend, daß x der diskrete Logarithmus von u modulo p zur Basis a ist.

Bei diesen sehr viel spezielleren „Geheimnissen“ ist klar, daß Kryptoverfahren auf der Basis von diskreten Logarithmen anders aussehen müssen als RSA.

a) Verschlüsselung nach Elgamal

Im Prinzip könnte man die Schlüsselvereinbarung nach DIFFIE und HELLMAN direkt zu einem Verschlüsselungsverfahren erweitern: Nachdem das gemeinsame Geheimnis $\gamma = a^{xy} \bmod p$ vereinbart ist, können Nachrichtenblöcke m_i mit $0 \le m_i < p-1$ in beide Richtungen verschlüsselt werden als $c_i = \gamma m_i \bmod p$. Da beide Partner den Wert von γ kennen, können sie leicht nach dem erweiterten EUKLIDischen Algorithmus ein δ berechnen, so daß $\gamma\delta \equiv 1 \bmod p$, und die verschlüsselte Information kann einfach entschlüsselt werden als $m_i = \delta c_i \bmod p$.

Solange nur ein einzelner Block m übertragen werden soll, ist dagegen nichts einzuwenden. Sobald aber mehrere Blöcke zu übertragen sind, wird dieses Verfahren verwundbar gegen Angriffe mit bekanntem Klartext: Falls ein Gegner für einen einzigen Chiffreblock c_i den Klartextblock m_i kennt (oder errät), kann er $\delta = m_i/c_i \bmod p$ berechnen und damit den gesamten Klartext entschlüsseln. Um das Verfahren

sicher zu machen, müßte man daher für jeden Block ein eigenes γ vereinbaren und dazu jedes Mal das gesamte DIFFIE-HELLMAN-Protokoll durchlaufen, was sehr aufwendig wäre.

Das Verfahren von ELGAMAL umgeht dieses Problem, indem es exakt dieselbe Mathematik mit einem leicht modifizierten Protokoll zu einem asymmetrischen Kryptoverfahren macht:

Die Parameter a und p sind entweder allgemein bekannte Systemparameter, oder jeder Teilnehmer A wählt sie selbst als Teil seines öffentlichen Schlüssels. Zusätzlich wählt er sich eine geheime Zufallszahl x und veröffentlicht $u = a^x \bmod p$.

Wer immer eine Nachricht $m_1, \ldots, m_r$ an A schicken möchte, erzeugt für jeden Block m_i eine Zufallszahl y_i berechnet daraus $v_i = a^{y_i} \bmod p$ und $c_i = u^{y_i} m_i$. Dann schickt er die Folge der Paare (v_i, c_i) an A. Der Chiffretext ist damit doppelt so lang wie der Klartext, was das Verfahren insbesondere für lange Texte nicht sonderlich attraktiv macht.

A muß zur Entschlüsselung den Multiplikator u^{y_i} kennen; dann kann er m_i als $c_i u^{-y_i}$ berechnen. Da $u^{y_i} \equiv a^{xy_i} \equiv \left(a^{y_i}\right)^x \equiv {v_i}^x \bmod p$ ist, hat er damit keine Probleme.

TAHER ELGAMAL wurde 1955 in Ägypten geboren. Er studierte zunächst Elektrotechnik an der Universität Kairo; nachdem er dort seinen BSc bekommen hatte, setzte er seine Studien fort an den Information Systems Laboratories der Stanford University. In seiner Masterarbeit ging es hauptsächlich um Systemtheorie, jedoch hörte er parallel auch freiwillig viele Mathematikvorlesungen und kam auf diesem Weg zur Kryptographie, die zum Thema seiner Doktorarbeit wurde. Nach dem Studium arbeitete er für eine ganze Reihe von Unternehmen, beispielsweise war er von 1995–1998 als Chefwissenschaftler von Netscape maßgeblich an der Entwicklung von SSL beteiligt. Zeitweise arbeitete er auch in selbst gegründeten Firmen. 2006 wurde er Chief Technology Officer der Tumbleweed Communications Corporation; seitdem diese 2008 von Axway übernommen wurde, ist er deren Chief Security Officer sowie Berater einer Reihe weiterer Unternehmen. Sein Name wird in der Literatur oft auch EL GAMAL oder ELGAMAL geschrieben; die obige Schreibweise ist die, die er selbst im Englischen benutzt.

Der offensichtliche Angriff eines Gegners besteht darin, aus u und a den diskreten Logarithmus x zu ermitteln, was nach derzeitigem Stand der Dinge schwierig erscheint. Ob andere Angriffe zum Erfolg führen könnten, ist (wie üblich) unbekannt – hoffentlich auch unseren Gegnern.

Genau wie bei RSA gibt es aber natürlich auch hier eine ganze Reihe von Möglichkeiten, das Verfahren durch schlechte Parameterwahl oder unsachgemäßen Gebrauch unsicher zu machen; einige davon sind in der am Ende von Kap. 4, §3*b)* zitierten Arbeit zu finden. Insbesondere muß auch hier die Nachricht mit Zufallsbits auf volle Blocklänge gebracht werden; ansonsten hat der Gegner Ansätze zur Entschlüsselung *ohne* Berechnung diskreter Logarithmen.

b) Das Verfahren von Massey-Omura

Bei diesem Verfahren geht es, wie bei der Schlüsselvereinbarung nach DIFFIE-HELLMAN, um Nachrichtenaustausch zwischen zwei Partnern A und B, die über keinerlei gemeinsame Schlüsselinformation verfügen; es gibt auch keine öffentlichen Schlüssel.

Das Verfahren läßt sich am einfachsten verstehen, wenn wir mit einem nichtmathematischen Analogon beginnen: Angenommen, A möchte einen Container mit wichtigen Unterlagen an B schicken, traut aber dem Transporteur nicht. Wenn er B vorher treffen kann, kauft er einfach ein gutes Vorhängeschloß und gibt B einen der beiden Schlüssel. Später kann er dann den Container mit dem Schloß und seinem Schlüssel verschließen, und B kann mit seinem Schlüssel das Schloß wieder entfernen, um den Container zu öffnen.

Wenn A und B keine Möglichkeit zu einem vorherigen Treffen haben, müssen sie umständlicher vorgehen: Jetzt kauft sich jeder der beiden ein Schloß, dessen Schlüssel dann nur er hat. A verschließt den Container mit seinem Schloß und schickt ihn an B. Der kann ihn natürlich nicht öffnen und schickt ihn deshalb ungeöffnet wieder zurück, verschließt ihn aber vorher noch zusätzlich mit *seinem* Schloß. A kann nun *sein* Schloß entfernen und schickt ihn, nun nur noch mit Bs Schloß gesichert, an B. Dieser kann *sein* Schloß entfernen und dann den Container öffnen.

In der digitalen Welten sieht das ganze so aus:

A und B einigen sich auf eine Primzahl p (die auch Konstante eines ganzen Netzwerks sein kann), und jeder erzeugt sich einen (geheimzuhaltenden) Exponenten e_A *bzw.* e_B, der prim ist zu $p-1$. Dazu berechnet er nach dem erweiterten EUKLIDischen Algorithmus ein (ebenfalls geheimzuhaltendes) Inverses modulo $p-1$; diese Inversen seien d_A und d_B. Nach dem kleinen Satz von FERMAT ist somit für jedes $m \in \mathbb{Z}$

$$m^{e_A d_A} \equiv m^{e_B d_B} \equiv m \bmod p\,.$$

Will nun B eine Nachricht m verschlüsselt an A schicken, so schickt er $c_1 = m^{e_B} \bmod p$. Damit kann natürlich weder A noch ein etwaiger Lauscher etwas anfangen: Da niemand außer B die beiden Exponenten e_B und d_B kennt, ist das einfach *irgendeine* Potenz zu *irgendeiner* Basis. Selbst ein BAYESscher Gegner, der alle Kombinationen (M, e) mit $M^e \equiv m^{e_B} \bmod p$ durchprobieren kann, wird dort für große p eine Fülle von potentiellen Klartexten finden, die alle ungefähr gleich wahrscheinlich sind.

A schickt die Nachricht daher gleich wieder zurück, potenziert sie aber vorher mit seinem Exponenten e_A. Was B erhält, ist also $c_2 = m^{e_B e_A}$, eine Nachricht die niemand entschlüsseln kann.

B potenziert diese Nachricht mit seinem Exponenten d_B; dies liefert

$$\left(m^{e_B e_A}\right)^{d_B} = m^{e_B e_B d_B} = m^{e_B d_B e_A} = \left(m^{e_B d_B}\right)^{e_A} \equiv m^{e_A} \bmod p\,.$$

Diese Nachricht schickt er an A, der nun mit seinem Exponenten d_A leicht den Klartext ermitteln kann.

Auch die Sicherheit dieses Verfahrens hängt an diskreten Logarithmen: Ein etwaiger Lauscher kennt die Zahlen

$$m^{e_B} \bmod p, \quad m^{e_B e_A} = \left(m^{e_B}\right)^{e_A} \quad \text{und} \quad m^{e_A} = \left(m^{e_B e_A}\right)^{d_B};$$

falls er in der Lage ist, diskrete Logarithmen modulo p zu berechnen, kann er e_A bestimmen als den diskreten Logarithmus von $m^{e_B e_A}$ zur Basis m^{e_B} und d_B als diskreten Logarithmus von $\left(m^{e_B e_A}\right)^{d_B}$ zur Basis $m^{e_B e_A}$. Damit kann auch er m berechnen, indem er beispielsweise m^{e_B} modulo p mit d_B potenziert. Die Primzahl p muß also auch bei diesem Verfahren so groß sein, daß die Berechnung diskreter Logarithmen modulo p zumindest praktisch undurchführbar ist. Ein *man in the*

middle-Angriff ist hier, im Gegensatz zu ELGAMAL, natürlich in genau der gleichen Weise wie beim Verfahren von DIFFIE-HELLMAN möglich.

JAMES L. MASSEY wurde 1934 in Wauseon, Ohio geboren. Er studierte Elektrotechnik an der University of Notre Dame und am MIT, wo er sich vor allem auf Informations- und Kodierungstheorie konzentrierte. Nach dem Studium war er 14 Jahre lang Professor in Notre Dame, dann kurz am MIT und an der University of California, Los Angelos (UCLA), bis er 180 einem Ruf an die ETH Zürich folgte, wo er bis zu seiner Emeritierung zum 1. April 1999 einen Lehrstuhl für Signal- und Informationsverarbeitung hatte.

JIM K. OMURA studierte in Stanford Elektrotechnik und war dann 15 Jahre lang Professor an der UCLA. Danach gründete er eine eigene Firma namens Cylink (inzwischen von Safenet übernommen) und arbeitete als Berater für verschiedene Firmen und Stiftungen.

c) DSA

Der Zusatzaufwand gegenüber RSA macht sowohl ELGAMAL aus auch MASSEY-OMURA für praktische Anwendungen eher uninteressant; hinzu kommt, daß zumindest bei MASSEY-OMURA ohne einen zusätzlichen Kanal keiner der beiden Partner sicher sein kann, daß er wirklich mit dem anderen kommuniziert. Dasselbe gilt natürlich zumindest für den Empfänger auch bei RSA; dort allerdings liefert das Verfahren selbst eine Möglichkeit für elektronische Unterschriften, die dieses Problem löst. Auch das Verfahren von ELGAMAL kann so modifiziert werden, daß es elektronische Unterschriften realisiert, mit diskreten Logarithmen kann man aber auch noch weiter gehen und sogar relativ kurze, aber trotzdem sichere Unterschriften produzieren.

Der Rechenaufwand pro Byte ist bei asymmetrischer Kryptoverfahren deutlich höher als bei den gängigen symmetrischen Verfahren; andererseits sind zu unterzeichnende Texte oft sehr lang, weil sie beispielsweise von Juristen unter Berücksichtigung aller Eventualitäten abgefaßt wurden. Deshalb wird meist nicht die gesamte Nachricht unterzeichnet, sondern nur ein sogenannten Hashwert. Dabei handelt es sich um eine kurze Bitfolge, die nach Art einer Prüfziffer von der ganzen Nachricht abhängt und diese auch charakterisiert.

Wir werden uns im übernächsten Kapitel genauer mit solchen Hashverfahren beschäftigen; dabei werden wir auch sehen, daß Hashwerte bislang meist 160 Bit hatten und daß man gerade dabei ist, diese zunehmend problematische Länge auf 224 oder besser noch 256 Bit zu erhöhen.

Wenn wir so einen Hashwert mit RSA unterzeichnen, hat die Unterschrift nach derzeitigem Sicherheitsstandard eine Länge von 2048 Bit. Verglichen mit der Länge des zu unterzeichnenden Hashwerts ist das offensichtlich weit übertrieben. Andererseits wäre eine Unterschrift, die auf diskreten Logarithmen in einem Körper mir nur etwa 2^{256} Elementen beruht, ohne großen Aufwand fälschbar.

Der *Digital Signature Algorithm* DSA bietet einen Ausweg aus diesem Dilemma, indem er zwar in einer großen Gruppe rechnet, dabei aber kurze Unterschriften aus einer deutlich kleineren Untergruppe liefert. Dieser Algorithmus wurde im *Digital Signature Standard* DSS der USA spezifiziert und zählt neben RSA auch zu den von der Bundesnetzagentur festgelegten „Geeigneten Algorithmen“.

Als Ordnung der Untergruppe wählt man eine Primzahl q, für die nach den derzeitigen Empfehlungen der Bundesnetzagentur seit Anfang 2010 eine Länge von mindestens 224 Bit notwendig ist; ab Anfang 2016 erhöht sich die Länge auf mindestens 256 Bit. Diese Längen hängen in erster Linie ab von den verwendeten (und zulässigen) Hashverfahren, nicht so sehr von Sicherheitsanforderungen.

Die Sicherheit wird gewährleistet (soweit dies möglich ist) durch eine zweite Primzahl p, die so gewählt wird, daß $p \equiv 1 \bmod q$ ist; für ihre Größe sind mindestens 2048 Bit vorgeschrieben.

Primzahlen $p \equiv 1 \bmod q$ sind nicht schwerer zu finden als beliebige Primzahlen: Falls man bei der Primzahlsuche wirklich auf Nummer sicher geht und Zufallszahlen auf Primalität testet, nimmt man hier einfach Zufallszahlen k und testet $kq + 1$ auf Primalität. Falls man mit ERATOSTHENES arbeitet, kann man das Sieben leicht so modifizieren, daß nur Zahlen der Form $kq + 1$ gesiebt werden. An den Erfolgschancen ändert dies in beiden Fällen nichts: Nach einem Satz von DIRICHLET über Primzahlen in arithmetischen Folgen ist die Dichte der Primzahlen der Form $kq + i$ für jedes i mit $0 < i < q$ dieselbe; in der Größenordnung n ist also weiterhin im Mittel jede $\ln n$-te solche Zahl eine Primzahl. (Tatsächlich sind es sogar geringfügig mehr, denn außer q selbst gibt es

natürlich keine Primzahl der Form $p = kq$. Bei den Größenordnungen von q mit denen wir arbeiten, geht aber der Unterschied zwischen q und $q-1$ definitiv im „Rauschen“ der im Kleinen sehr unregelmäßigen Primzahlverteilung unter.)

Als nächstes muß ein Element g gefunden werden, dessen Potenzen im Körper $\mathbb{F}_p$ eine Gruppe der Ordnung q bilden. Auch das ist einfach: Man starte mit irgendeinem Element $g_0 \in \mathbb{F}_p \setminus \{0\}$ und berechne seine $(p-1)/q$-te Potenz. Falls diese ungleich eins ist, muß sie wegen $g_0^{p-1} = 1$ die Ordnung q haben; andernfalls muß ein neues g_0 betrachtet werden.

Die so bestimmten Zahlen q, p und g werden veröffentlicht und können auch in einem ganzen Netzwerk global eingesetzt werden. Geheimer Schlüssel jedes Teilnehmers ist eine Zahl x zwischen eins und $q-1$; der zugehörige öffentliche Schlüssel ist $u = g^x \bmod p$.

Unterschreiben lassen sich mit diesem Verfahren Nachrichtenblöcke m mit $0 \le m < q$; im allgemeinen wird es sich dabei um Hashwerte der eigentlich zu unterschreibenden Nachricht handeln. Dazu wählt man für jede Nachricht eine Zufallszahl k mit $0 < k < q$ und berechnet

$$r = (g^k \bmod p) \bmod q\,.$$

Da q eine Primzahl ist, hat k ein multiplikatives Inverses modulo q; man kann also modulo q durch k dividieren und erhält eine Zahl s, für die

$$sk \equiv m + xr \bmod q$$

ist; die Unterschrift unter die Nachricht m besteht dann aus den beiden Zahlen r und s zwischen 0 und $q-1$. Sie kann nur erzeugt werden von jemanden, der den geheimen Schlüssel x kennt.

Überprüfen kann die Unterschrift allerdings jeder: Ist t das multiplikative Inverse zu s modulo q, so ist $k \equiv tsk \equiv tm + xtr \bmod q$, also, da g die Ordnung q hat, $g^k \bmod p = g^{tm} g^{xtr} \bmod p = g^{tm} u^{tr} \bmod p$. Modulo q ist die linke Seite gleich r, und auf der rechten Seite können sowohl g^{tm} als auch u^{tr} aus öffentlicher Information und der Unterschrift berechnet werden. Modulo q kann diese Gleichung somit überprüft werden; die Unterschrift wird anerkannt, wenn

$$r \equiv (g^{tm} u^{tr} \bmod p) \bmod q$$

ist. (Die beiden Potenzen und ihr Produkt müssen natürlich zunächst modulo p berechnet werden: Zwei modulo p kongruente Zahlen sind praktisch nie auch kongruent modulo q.)

Ein Angreifer müßte sich nach allem was wir wissen x aus u verschaffen, müßte also ein diskretes Logarithmenproblem modulo der großen Primzahl p lösen, so daß der Sicherheitsstandard dem des diskreten Logarithmenproblems modulo p entsprechen sollte, obwohl die Unterschriften deutlich kürzer sind.

§3: Strategien zur Berechnung diskreter Logarithmen

Genau wie es zahlreiche Ansätze gibt, ganze Zahlen auf mehr oder weniger effiziente Weise zu faktorisieren, gibt es auch die verschiedensten Methoden, diskrete Logarithmen zu berechnen. Obwohl es keinen klaren theoretischen Zusammenhang zwischen den beiden Problemen gibt, zeigt die Erfahrung der letzten Jahren eine erstaunliche Parallelität zwischen den entsprechenden Algorithmen. Da das Interesse an Faktorisierungsalgorithmen zumindest bislang deutlich größer ist, kamen neue Entwicklungen in der letzten Zeit immer von dort; erstaunlicherweise stellte sich aber immer ziemlich schnell heraus, daß ein ähnliches Verfahren mit praktisch demselben Aufwand auch diskrete Logarithmen berechnen kann. Daher benötigen wir, um vergleichbare Sicherheit zu erreichen, bei der Kryptographie mit diskreten Logarithmen Moduln zumindest derzeit dieselben Längen wie bei RSA.

a) Probieren

Am einfachsten und langwierigsten ist das Probieren: Um den diskreten Logarithmus modulo p von a zur Basis g zu bestimmen, berechnet man (analog zur Faktorisierung durch Abdividieren) systematisch alle Potenzen von g, bis man a erhält. Dies erfordert im Mittel $p/2$ Versuche.

b) Gruppentheoretische Formulierung des Problems

Für bessere Verfahren müssen wir das Problem zunächst mathematisch aufbereiten. Dazu dient die Gruppentheorie. Zur Erinnerung seien die wesentlichen Definitionen noch einmal wiederholt:

Definition: Eine Gruppe ist eine Menge G zusammen mit einer Verknüpfung $\star\colon G \times G \to G$, so daß gilt:

- Für alle $g, h, k \in G$ ist $(g \star h) \star k = g \star (h \star k)$ *(Assoziativgesetz)*
- Es gibt ein Element $e \in G$, genannt *Neutralelement*, so daß für alle $g \in G$ gilt: $g \star e = e \star g = g$.
- Zu jedem $g \in G$ gibt es ein *inverses Element* $h \in G$, für das gilt: $g \star h = h \star g = e$.
- Die Gruppe heißt *abelsch*, wenn zusätzlich gilt: $g \star h = h \star g$ für alle $g, h \in G$. *(Kommutativgesetz)*.
- Die Gruppe heißt *endlich*, wenn die Menge G nur endlich viele Elemente hat.
- Eine endliche Gruppe heißt *zyklisch*, wenn es ein Element $g \in G$ gibt, so daß sich jedes $h \in G$ schreiben läßt als $h = \underbrace{g \star \cdots \star g}_{n \text{ mal}} \stackrel{\text{def}}{=} g^n$ mit einer natürlichen Zahl $n \in \mathbb{N}$.

Das inverse Element h zu g schreiben wir kurz als g^{-1}, und für $n \in \mathbb{N}$ soll $g^{-n} = \left(g^{-1}\right)^n$ sein. g^0 bezeichne für jedes $g \in G$ das Neutralelement.

Als erstes müssen wir klären, welche Werte die Potenzen $g^x \bmod p$ überhaupt annehmen können. Klar ist

Lemma: Für jede natürliche Zahl N ist die Menge aller $g^x \bmod N$ mit $x \in \mathbb{N}$ eine zyklische Gruppe bezüglich der Multiplikation modulo N.

Beweis: Das Assoziativgesetz folgt sofort aus dem für die Addition natürlicher Zahlen. Da es nur endlich viele Restklassen modulo N gibt, muß es außerdem zwei Zahlen $r > s$ geben mit $g^r \equiv g^s \bmod N$; mit $m = r - s$ ist daher $g^m \equiv 1 \bmod N$ als Potenz von g darstellbar, und das ist das neutrale Element. Da $g^{x+m} \equiv g^x \bmod N$ für alle x, läßt sich jede Restklasse in der Form $g^x \bmod N$ mit $1 \leq x \leq m$ darstellen; das Inverse dazu ist dann $g^{m-x} \bmod N$. Damit sind alle Gruppenaxiome nachgewiesen, und zyklisch ist die Gruppe nach Konstruktion. ∎

Definition: Die kleinste natürliche Zahl m, für die $g^m \equiv 1 \bmod N$ ist, heißt *Ordnung von g modulo N*.

Das Knacken eines Kryptosystems auf der Basis diskreter Logarithmen ist umso einfacher, je kleiner die Ordnung der Basis g ist. Daher müssen

wir Elemente möglichst großer Ordnung finden. Dazu betrachten wir das Problem gruppentheoretisch:

Definition: Die Ordnung eines Elements a einer (multiplikativ geschriebenen) Gruppe G ist die kleinste natürliche Zahl r, für die a^r gleich dem Neutralelement ist. Falls es keine solche Zahl r gibt, sagen wir, a habe unendliche Ordnung. Die Ordnung einer endlichen Gruppe ist deren Elementanzahl.

Lemma (LAGRANGE): In einer endlichen Gruppe teilt die Ordnung r eines jeden Elements g die Gruppenordnung.

Beweis: Wir führen auf der Gruppe G eine Äquivalenzrelation ein durch die Vorschrift $u \sim v$, falls es ein $s \in \mathbb{N}$ gibt mit $u = vg^s$. Offensichtlich besteht die Äquivalenzklasse eines jeden Elements $u \in G$ aus genau r Elementen, nämlich $u, ug, \ldots, ug^{r-1}$. Da G die Vereinigung aller Äquivalenzklassen ist, muß die Gruppenordnung somit ein Vielfaches von r sein. ∎

JOSEPH-LOUIS LAGRANGE (1736–1813) wurde als GIUSEPPE LODOVICO LAGRANGIA in Turin geboren und studierte dort zunächst Latein. Erst eine alte Arbeit von HALLEY über algebraische Methoden in der Optik weckte sein Interesse an der Mathematik, woraus ein ausgedehnter Briefwechsel mit EULER entstand. In einem Brief vom 12. August 1755 berichtete er diesem unter anderem über seine Methode zur Berechnung von Maxima und Minima; 1756 wurde er, auf EULERs Vorschlag, Mitglied der Berliner Akademie; zehn Jahre später zog er nach Berlin und wurde dort EULERs Nachfolger als mathematischer Direktor der Akademie. 1787 wechselte er an die Pariser Académie des Sciences, wo er bis zu seinem Tod blieb und unter anderem an der Einführung des metrischen Systems beteiligt war. Seine Arbeiten umspannen weite Teile der Analysis, Algebra und Geometrie.

Für eine Primzahl p bilden die Zahlen modulo p bekanntlich einen Körper; wie uns das folgende Lemma zeigt, können wir dann Elemente der größtmöglichen Ordnung $p - 1$ finden:

Lemma: Ist k ein endlicher Körper, so bilden die Elemente von $k \setminus \{0\}$ bezüglich der Multiplikation eine zyklische Gruppe.

Beweis: Da die multiplikative Gruppe eines Körpers mit q Elementen aus allen Körperelementen außer der Null besteht, hat sie die Ordnung $q-1$. Nach LAGRANGE ist daher die Ordnung eines jeden Elements ein Teiler von $q-1$. Wir müssen zeigen, daß es mindestens ein Element gibt, dessen Ordnung *genau* $q-1$ ist.

Für jeden Primteiler p_i von $q-1$ hat die Polynomgleichung

$$x^{(q-1)/p_i} = 1$$

höchstens $(q-1)/p_i$ Lösungen im Körper k; es gibt also zu jedem p_i ein Körperelement a_i mit $a_i^{(q-1)/p_i} \neq 1$.

q_i sei die größte Potenz von p_i, die $q-1$ teilt, und $g_i = a_i^{(q-1)/q_i}$ die $(q-1)/q_i$-te Potenz von a_i. Dann ist

$$g_i^{q_i} = a_i^{q-1} = 1 \quad \text{und} \quad g_i^{\frac{q_i}{p_i}} = a_i^{\frac{q-1}{p_i}} \neq 1\,;$$

g_i hat also die Ordnung q_i. Da die verschiedenen q_i Potenzen verschiedener Primzahlen p_i sind, hat daher das Produkt g aller g_i das Produkt aller q_i als Ordnung, also $q-1$. Damit ist die multiplikative Gruppe des Körpers zyklisch. ∎

In unserer Situation bedeutet dies, daß es mindestens eine Zahl g gibt, so daß die Abbildung

$$\varphi\colon \begin{cases} \mathbb{Z}/(p-1) \to \mathbb{Z}/p \setminus \{0\} \\ x \mapsto g^x \end{cases}$$

bijektiv ist. Solche Zahlen g bezeichnet man als *primitive Wurzeln* modulo p. Kryptoverfahren auf der Basis diskreter Logarithmen sind daher dann am sichersten, wenn die Basis g eine primitive Wurzel modulo p ist; in diesem Fall gibt es die meisten verschiedenen Potenzen $g^x \bmod p$, nämlich $p-1$ Stück.

Nach LAGRANGE ist die Ordnung m eines jeden Elements g ein Teiler m von $p-1$. Ist m ein echter Teiler, so gibt es mindestens einen Primteiler q von $p-1$, so daß m sogar ein Teiler von $(p-1)/q$ ist; wenn wir entscheiden wollen, ob eine gegebene Basis g eine primitive Wurzel ist, genügt es also, für alle Primteiler q von $p-1$ die Potenzen $g^{(p-1)/q}$ zu

berechnen; falls keine davon gleich eins modulo p ist, haben wir eine primitive Wurzel gefunden.

In der Praxis wird dies freilich meist daran scheitern, daß wir $p-1$ nicht faktorisieren können – es sei denn, wir haben p geeignet gewählt. Falls wir beispielsweise eine Primzahl p der Form $p = 2p' + 1$ mit primem p' wählen, ist $p-1 = 2p'$, und g ist genau dann primitive Wurzel modulo p, wenn weder g^2 noch $g^{p'}$ kongruent eins modulo p ist. Da g^2 nur für $g \equiv \pm 1 \bmod p$ kongruent eins ist, ist somit ein g mit $1 < g < p-1$ genau dann primitive Wurzel, wenn $g^{p'} \not\equiv 1 \bmod p$ ist; andernfalls hat es die (kryptographisch fast genauso sichere) Ordnung p'.

c) Anwendung des chinesischen Restesatzes

Die Basis g habe die Ordnung m modulo N und ihre Primfaktorzerlegung sei $m = \prod_{i=1}^{r} q_i^{e_i}$. Wir wollen das diskrete Logarithmenproblem $g^x \equiv a \bmod N$ lösen. Um es auf diskrete Logarithmenprobleme in Gruppen der Ordnungen $q_i^{e_i}$ zurückzuführen, setzen wir $n_i = m/q_i^{e_i}$; für $g^x \equiv a \bmod N$ ist dann auch $g^{n_i x} \equiv a^{n_i} \bmod N$, und $g_i = g^{n_i}$ hat modulo N nur die Ordnung $q_i^{e_i}$.

Falls wir die r diskrete Logarithmenprobleme $g_i^{x_i} \equiv a^{n_i} \bmod N$ lösen können, lassen sich die Lösungen x_i leicht zu einer Lösung des ursprünglichen Problems zusammensetzen: Da die n_i keinen gemeinsamen Primteiler haben, ist ihr ggT gleich eins; es gibt also ganze Zahlen α_i mit $\sum_{i=1}^{r} \alpha_i n_i = 1$, die wir uns mit dem erweiterten EUKLIDischen Algorithmus leicht verschaffen können. Mit $x = \sum_{i=1}^{r} \alpha_i n_i x_i$ ist dann

$$g^x = \prod_{i=1}^{r} g^{n_i x_i \alpha_i} \equiv \prod_{i=1}^{r} a^{n_i \alpha_i} = a^{\sum_{i=1}^{r} \alpha_i n_i} = a \bmod N\,.$$

d) Das Verfahren von Pohlig und Hellman

Tatsächlich reicht es sogar, wenn wir das diskrete Logarithmenproblem statt in Gruppen der Ordnung $q_i^{e_i}$ in Gruppen der Ordnung q_i lösen können. Dazu gehen wir folgendermaßen vor:

Der Einfachheit halber beschränken wir uns auf eine zyklische Gruppe G von Primzahlpotenzordnung q^e und wählen dort ein erzeugendes Ele-

ment g. Gesucht ist der diskrete Logarithmus eines weiteren Elements $a \in G$ zur Basis g.

Diese gesuchte Zahl x liegt zwischen null und $q^e - 1$ (tatsächlich ist sie sogar höchstens $q^e - q^{e-1}$); wenn wir sie in Ziffern zur Basis p schreiben, ist also

$$x = x_0 + x_1 q + x_2 q^2 + \cdots + x_{e-1} q^{e-1} \quad \text{mit} \quad 0 \le x_i \le q\,.$$

Wir wollen uns überlegen, daß wir die „Ziffern“ x_i als diskrete Logarithmen in einer Untergruppe der Ordnung q berechnen können.

Aus $a = g^x$ folgt die Beziehung

$$a^{q^j} = g^{x p^j} = g^{x_0 q^j} \cdot g^{x_1 q^{j+1}} \cdot g^{x_2 q^{j+2}} \cdots g^{x_{e-1} q^{j+e-1}}\,.$$

Da aber $g^{q^e} \equiv 1 \bmod N$ ist, sind modulo N alle Terme, in denen q einen größeren Exponenten als e hat, gleich eins; tatsächlich ist also

$$a^{q^j} \equiv q^{p^j x} \equiv g^{x_0 q^j} \cdot g^{x_1 q^{j+1}} \cdot g^{x_2 q^{j+2}} \cdots g^{x_{e-j-1} q^{e-1}} \bmod N\,.$$

Insbesondere folgt für $j = e - 1$, daß $a^{q^{e-1}} \equiv g^{x_0 q^{e-1}} \bmod N$ ist, x_0 ist also die Lösung eines diskreten Logarithmenproblems in der von $g^{q^{e-1}}$ erzeugten Untergruppe der Ordnung q.

Angenommen, wir haben die „Ziffern“ von x_0 bis x_r bereits bestimmt. Dann schreiben wir

$$a \cdot g^{-x_0 - x_1 q - \cdots - x_r q^r} = g^{x_{r+1} q^{r+1}} \cdots g^{x_{e-1} q^{e-1}}$$

und potenzieren diese Gleichung mit q^{e-2-r}. Modulo N können wir rechts alle Terme außer dem ersten streichen; wir erhalten also die Gleichung

$$a^{q^{e-2-r}} \cdot g^{-q^{e-2-r}(x_0 + x_1 q + \cdots + x_r q^r)} \equiv q^{x_{r+1} q^{e-1}} \bmod N\,,$$

die uns x_{r+1} als diskreten Logarithmus der (bekannten) linken Seite liefert, und zwar wieder in der von $g^{q^{e-1}}$ erzeugten Untergruppe der Ordnung q.

Auf diese Weise können wir nacheinander die sämtlichen x_i berechnen und damit auch x.

Zusammen mit dem oben diskutierten Ansatz über den chinesischen Restesatz ist dies das Verfahren von POHLIG und HELLMAN zur Berechnung diskreter Logarithmen in einer zyklischen Gruppe G der Ordnung n: Sie kann zurückgeführt werden auf die Berechnung diskreter Logarithmen in Untergruppen, deren Ordnungen die Primteiler von n sind.

e) Folgerung für die Sicherheit von Kryptosystemen

Die Diskussion in den beiden vorigen Abschnitten zeigt, daß die Schwierigkeit der Berechnung einen diskreten Logarithmus in einer zyklischen Gruppe der Ordnung m relativ einfach zurückgeführt werden kann auf die Berechnung diskreter Logarithmen in Untergruppen, deren Ordnungen die Primteiler von m sind. Als Faustregel können wir daher festhalten, daß die Sicherheit diskreter Logarithmen in einer zyklischen Gruppe der Ordnung m im wesentlichen nur gleich der Sicherheit in einer Untergruppe der Ordnung q ist, wobei q der größten Primteiler von n ist.

Idealerweise sollte daher die Gruppenordnung m selbst eine Primzahl sein, was allerdings zumindest für die multiplikative Gruppe modulo p nur im kryptographisch völlig uninteressanten Fall $p = 3$ der Fall ist.

Hier empfiehlt sich, eine Primzahl p der Form $2p' + 1$ zu wählen, wobei auch p' eine Primzahl ist. Die multiplikative Gruppe modulo p hat dann die Ordnung $2p'$, und die Sicherheit entspricht der in einer Gruppe der Ordnung p'.

f) Baby step und giant step

Bei der Faktorisierung ganzer Zahlen konnten wir den Aufwand gegenüber dem naiven Abdividieren durch die Monte-Carlo-Methode auf ungefähr die Quadratwurzel der zu faktorisierenden Zahl reduzieren. Auch für die Bestimmung diskreter Logarithmen gibt es entsprechende Verfahren, zum Beispiel den *baby step – giant step* Algorithmus von SHANKS, das auf Kosten von mehr Speicherplatz die Rechenzeit gegenüber reinem Probieren deutlich reduziert: Ist n die Ordnung von g, so ist der Aufwand nicht mehr proportional zu n, sondern nur noch zu $\sqrt{n}$.

DANIEL SHANKS (1917–1996) wurde in Chicago geboren, wo er zur Schule ging und 1937 einen Bachelorgrad in Physik der University of Chicago erwarb. Er arbeitete bis 1950 in verschiedenen Positionen als Physiker, danach als Mathematiker. 1949 begann er ein *graduate* Studium der Mathematik an der University of Maryland, zu dessen Beginn er der erstaunten Fakultät als erstes eine fertige Doktorarbeit vorlegte. Da zu einem *graduate* Studium auch Vorlesungen und Prüfungen gehören, wurde diese noch nicht angenommen, und da er während seines Studiums Vollzeit arbeitete, dauerte es noch bis 1954, bevor er alle Voraussetzungen erfüllte; dann wurde die Arbeit in praktisch unveränderter Form akzeptiert. Erst 1977 entschloß er sich, eine Stelle an einer Universität anzunehmen und war dann bis zu seinem Tod Professor an der University of Maryland.

SHANKS wählt eine natürliche Zahl m die ungefähr gleich $\sqrt{n}\cdot\log_2 n$ ist; falls man n nicht so genau kennt, kann zwar die Effizienz des Verfahrens unter einer schlechten Wahl von m geringfügig leiden, aber solange die Größenordnungen einigermaßen stimmen, ist das nicht so dramatisch. Wichtig ist nur, daß $m > \sqrt{n}$ ist, aber nicht dramatisch größer.

Danach berechnet er die sämtlichen Potenzen g^i von g mit Exponenten $i \le m$; das sind m sogenannte *baby steps*.

Bei den dann folgenden *giant steps* berechnet er, um den diskreten Logarithmus von a zu erhalten, die Elemente $a \cdot g^{-mj}$ für $j = 1, 2, \ldots$ und vergleicht sie mit den Vielfachen aus dem ersten Teil. Ein solcher Vergleich kann etwa über eine binäre Suche oder eine *hash*-Tabelle implementiert werden und hat einen Aufwand proportional $\log_2 n$.

Sobald ein Wert $a \cdot g^{-mj}$ gefunden ist, der mit einer der in den *baby steps* berechneten Potenzen g^i übereinstimmt, gilt $a \cdot g^{-mj} = g^i$ oder $a = g^{mj+i}$; der diskrete Logarithmus von a zur Basis g ist also $mj + i$. Die notwendige Anzahl von *giant steps* liegt im schlimmsten Fall bei $n/m \approx \sqrt{n}$; im Durchschnitt ist sie halb so groß.

g) Zahme und wilde Kängurus

In den Jahren um 1975 entwickelte der britische Mathematiker JOHN M. POLLARD mehrere recht einfache Algorithmen zur Faktorisierung ganzer Zahlen sowie zur Berechnung diskreter Logarithmen, die auch heute noch (teils in verbesserter Form) zu den Standardwerkzeugen der algorithmischen Zahlentheorie gehören. Eine seiner Methoden verwendet eine Strategie zur Jagd auf Kängurus.

JOHN M. POLLARD ist ein britischer Mathematiker, der hauptsächlich bei British Telecom arbeitete. Er veröffentlichte zwischen 1971 und 2000 rund zwanzig mathematische Arbeiten, größtenteils auf dem Gebiet der algorithmischen Zahlentheorie. Bekannt ist er auch für seine Beiträge zur Kryptographie, für die er 1999 den RSA Award erhielt. Außer dem hier vorgestellten Algorithmus entwickelte er unter anderem auch das im letzten Kapitel erwähnte Zahlkörpersieb, dessen Weiterentwicklungen die derzeit schnellsten Faktorisierungsalgorithmen für große Zahlen sind. Seine home page, um die er sich auch jetzt im Ruhestand noch kümmert, ist `jmptidcott.googlepages.com/index.html`.

Da POLLARD kein Australier ist, sondern Brite, sind natürlich auch seine Kängurus britisch. Das geht zwar nicht so weit, daß diese Schlangen an Bushaltestellen bilden, sie springen aber im Gegensatz zu ihren australischen Artgenossen auch nicht völlig ungeordnet durch die Gegend: Sie springen immer geradeaus, und die Sprunglängen sind natürliche Zahlen aus einer endlichen Teilmenge $S \subset \mathbb{N}$, die durch den Startpunkt des Sprungs eindeutig festgelegt sind. Die Position eines Kängurus kann daher durch eine natürliche Zahl $u \in \mathbb{N}$ beschrieben werden, und wenn es von dort aus abspringt, springt es zum Punkt $u + f(u)$, wobei $f\colon \mathbb{N} \to S$ eine bekannte Funktion ist, die wohl in erster Linie von der Bodenbeschaffenheit in den einzelnen Punkten $u \in \mathbb{N}$ abhängen dürfte. Da die Landschaft in Großbritannien sehr variabel ist, können wir in erster Näherung annehmen, daß sich f wie eine Zufallsfunktion verhält; ihr Erwartungswert sei m, das arithmetische Mittel der Elemente von S.

Um ein wildes Känguru zu fangen, überredet POLLARD ein zahmes Känguru, von einem Startpunkt u_0 aus loszuspringen und n Sprünge zu machen. Nach jedem Sprung soll es auf seiner jeweiligen Position ein Loch graben und dieses gut mit Zweigen oder ähnlichem kaschieren. Die Positionen u_i, bei denen es Löcher gräbt, sind rekursiv berechenbar durch die Vorschrift $u_i = u_{i-1} + f(u_{i-1})$.

Falls nun ein wildes Känguru auf derselben Strecke unterwegs ist, kennen wir dessen Startpunkts v_0 natürlich nicht; da es britisch ist, wissen wir aber, wie es springt: Nach i Sprüngen ist es auf einer Position v_i, die über die Rekursion $v_i = v_{i-1} + f(v_{i-1})$ gegeben ist.

Da sich f gemäß unserer Annahme wie eine Zufallsfunktion verhält, fällt das wilde Känguru bei jedem Schritt mit einer Wahrscheinlichkeit von ungefähr $1/m$ in ein Loch und endet dann als Kängurubraten; seine

Chance, bei $r < n$ Sprüngen alle Löcher zu vermeiden, liegt also etwa bei $(1 - 1/m)^r$. Setzen wir $r = am$, so können wir dies auch schreiben als

$$\left(1 - \frac{1}{m}\right)^r = \left(1 - \frac{1}{m}\right)^{ma} \approx e^{-a}$$

für hinreichend große m. Schon für $a = 3$ beträgt diese Chance nur noch knapp 5%, für $a = 6$ ist sie ungefähr $1 : 400$, für $a = 7$ weniger als $1 : 1000$. Das Känguru hat also kaum eine Chance, dem Kochtopf zu entgehen.

Nun gibt es zwar sicherlich sowohl Zahlentheoretiker als auch Kryptanalytiker, die gerne Kängurubraten essen; während der Arbeitszeit interessieren sie sich aber mehr für Fragen wie die Faktorisierung ganzer Zahlen oder die Berechnung diskreter Logarithmen.

Zur Berechnung des diskreten Logarithmus einer Zahl u zur Basis g modulo p können Kängurujäger wie folgt vorgehen: Sie wählen zunächst ein Intervall (A, B), in dem der diskrete Logarithmus liegt. Wenn sie keinerlei spezielle Informationen haben, wird dies zwangsläufig das Intervall $(1, N)$ sein müssen; in Situationen wie beim DSA mit großer Primzahl p und kleiner Primzahl q weiß man aber, daß es bereits eine Lösung im viel kleineren Intervall $(1, q)$ gibt.

Das zahme Känguru startet mit $v_0 = g^A \bmod p$ und springt dann nacheinander die Positionen $v_i = v_{i-1} g^{f(v_{i-1})}$ an, bis es das Suchintervall (A, B) verlassen hat.

Das wilde Känguru startet an der Position $u_0 = u = g^x \bmod p$, wobei x den zu berechnenden diskreten Logarithmus bezeichnet; seine Landepositionen sind die Punkte u_i mit $u_i = u_{i-1} g^{f(u_{i-1})}$, bis es eventuell in ein vom zahmen Känguru gegrabenes Loch fällt. Wenn es alle Fallen vermeidet, war der Ansatz erfolglos; andernfalls haben wir haben wir zwei Indizes i, j mit $u_i = v_j$.

Die Rekursionen für u_i und v_j lassen sich leicht auflösen; wir erhalten die Gleichung

$$g^{B+f(v_0)+f(v_1)+\cdots+f(v_{i-1})} = g^{x+f(u_0)+f(u_1)+\cdots+f(u_{j-1})}$$

und somit den gesuchten diskrete Logarithmus von u als

$$x = B + f(v_0) + f(v_1) + \cdots + f(v_{i-1}) - f(u_0) - f(u_1) - \cdots - f(u_{j-1}) .$$

Um Aufwand und Erfolgschancen der Jagd abzuschätzen, gehen wir davon aus, daß beide sich die Sprungpositionen beider Kängurus wie Zufallsfolgen verhalten. Ist m das arithmetische Mittel von S, braucht das zahme Känguru ungefähr $\alpha = (B - A)/m$ Sprünge, um das Intervall zu durchqueren; da das wilde irgendwo mitten im Intervall startet, können es da erheblich weniger sein. Auf jeden Fall haben wir aber in einem Intervall der Länge $B - A$ mehr als α zufällige Werte; nach dem Geburtstagsparadoxon (mit dem wir uns im Kapitel über Hashfunktionen noch genauer beschäftigen werden) steigt die Chance auf eine Koinzidenz sehr schnell in die Nähe der Eins, sobald $\alpha = (B-A)/m > \sqrt{B - A}$ ist, also $m < \sqrt{B - A}$. Mit m etwas kleiner als $\sqrt{B - A}$ haben wir etwas mehr als $\sqrt{B - A}$ Sprünge des zahmen Kängurus und im Mittel etwa halb so viele für das wilde; der Aufwand ist also in der Größenordnung von $\sqrt{B - A}$ mit einer zwar recht hohen Erfolgswahrscheinlichkeit, aber ohne Erfolgsgarantie.

h) Indexkalkül

Die derzeit besten Faktorisierungsalgorithmen beruhen auf dem quadratischen und dem Zahlkörpersieb; für beide wurden bald nach ihrer Einführung ähnliche Siebalgorithmen gefunden, die zur Berechnung diskreter Logarithmen führen. Wir beschränken uns hier, wie auch schon bei der Faktorisierung, auf das quadratische Sieb, dessen Variante für diskrete Logarithmen als *Indexkalkül* bezeichnet wird nach der in der Zahlentheorie ebenfalls gebräuchlichen Sprechweise Index für den diskreten Logarithmus. Wir beschränken uns auf die einfachste Variante speziell für Primkörper $\mathbb{F}_p$.

Wie beim quadratischen Sieb wird eine Schranke B festgelegt und damit eine Faktorbasis $\mathcal{B}$ definiert; diese besteht hier aus *allen* Primzahlen $q \le B$. Der Algorithmus besteht aus zwei Teilen:

Im ersten Teil berechnet man die diskreten Logarithmen aller Primzahlen q aus der Faktorbasis zur gegebenen Basis a modulo p. Dies mag auf den ersten Blick unsinnig erscheinen, denn schließlich suchen wir

den diskreten Logarithmus *einer* Zahl und beginnen dazu mit der Berechnung der diskreten Logarithmen vieler Zahlen. Die Logarithmen der Primzahlen lassen sich aber simultan wie folgt berechnen: Man berechne viele Potenzen $a^y \bmod p$ und suche diejenigen, die eine Primfaktorzerlegung mit lauter Faktoren aus $\mathcal{B}$ haben. Ist

$$a^y \bmod p = q_1^{e_1} \cdots q_r^{e_r} ,$$

so ist

$$y \equiv e_1 \log_a q_1 + \cdots + e_r \log_a q_r \bmod m ,$$

wobei m die kleinste natürliche Zahl ist mit $a^m \equiv 1 \bmod p$. Für eine primitive Wurzel a modulo p ist $m = p - 1$, ansonsten kann m auch ein echter Teiler davon sein.

Mit genügend vielen Gleichungen dieser Form hat man ein lineares Gleichungssystem für die Logarithmen der $q \in \mathcal{B}$, allerdings leider nicht über einem Körper, sondern modulo der im allgemeinen zusammengesetzten Zahl m. Falls m Produkt von Primzahlen ist, löst man das Gleichungssystem modulo jeder dieser Primzahlen und setzt die Lösungen nach dem chinesischen Restesatz zusammen; wenn auch echte Primzahlpotenzen P^s in m stecken, schreibt man die e_i und die linken Seiten y im Zahlensystem zur Basis P und erhält dann für jede Ziffer ein lineares Gleichungssystem über dem Körper mit P Elementen, aus denen man die Lösung modulo P^s zusammensetzen kann. Dieser erste Schritt ist offensichtlich völlig unabhängig vom Element x, dessen Logarithmus wir suchen; er kann für eine gegebene Basis a und Primzahl p ein für allemal im voraus durchgeführt werden.

Im zweiten Schritt betrachtet man für zufällig gewählte Exponenten y die Elemente $a^y x \bmod p$, bis man eines findet, das nur durch Primzahlen aus der Faktorbasis teilbar ist. Falls etwa $a^y x \bmod p = q_1^{f_1} \cdots q_s^{f_s}$ ist, erhalten wir

$$\log_a x = f_1 \log_a q_1 + \cdots + f_s \log_a q_s - y \bmod m .$$

Leider gibt es kein Siebverfahren, mit dem sich feststellen läßt, welche Werte $a^y \bmod p$ durch eine gegebene Primzahl q teilbar sind; hier muß man also explizit faktorisieren – zumindest so lange, bis alle Faktoren aus $\mathcal{B}$ gefunden sind. Egal ob man hier mit Probedivisionen arbeitet

oder etwa mit POLLARDs ρ-Methode oder mit elliptischen Kurven: Das Verfahren ist für große p deutlich schneller als beispielsweise *baby step – giant step*, und der Aufwand steigt langsamer als exponentiell in der Ziffernzahl von p.

§4: Diskrete Logarithmen in anderen Gruppen

2048 Bit-Zahlen sind bereits ziemlich unhandlich, und in Zukunft wird die Mindestlänge sicherer Moduln garantiert noch weiter wachsen. Daher suchen Kryptologen bereits seit langer Zeit nach Alternativen. Die derzeit interessantesten (und zunehmend bereits in der Praxis eingesetzten) Verfahren beruhen auf einer Verallgemeinerung diskreter Logarithmen:

a) Die abstrakte Situation

Ist G irgendeine Gruppe und $g \in G$, so können wir die Abbildung

$$\varphi_g \colon \mathbb{Z} \to G; \quad n \mapsto g^n$$

betrachten. Falls man in der Gruppe G überhaupt konkret rechnen kann, lassen sich die Werte $\varphi_g(n) = g^n$ nach dem üblichen Verfahren durch Quadrieren und Multiplizieren mit einem Aufwand in der Größenordnung $\log_2 n$ berechnen.

Die Umkehrfunktion $\varphi_g^{-1} \colon \text{Bild}\, \varphi_g \to \mathbb{Z}/\,\text{Kern}\, \varphi_g$ bezeichnen wir auch hier als einen diskreten Logarithmus; falls er hinreichend schwer zu berechnen ist, eignet er sich als Grundlage für Kryptosysteme.

Das Bild von φ_g besteht offensichtlich genau aus den Potenzen von g, ist also eine zyklische Gruppe. Damit können wir uns ohne Beschränkung der Allgemeinheit auf zyklische Gruppen beschränken.

Für die Berechnung diskreter Logarithmen können wir die meisten Verfahren aus dem letzten Paragraphen ohne nennenswerte Änderungen auf die abstrakte Situation verallgemeinern: Für den chinesischen Restesatz sowie die Methode von POHLIG und HELLMAN brauchen wir nur irgendeine zyklische Gruppe, und auch den zahmen und wilden Kängurus ist es gleichgültig, durch welche Gruppe wir sie springen lassen:

Die Sprungweiten beziehen sich schließlich auf den stets ganzzahligen Exponenten.

Anders ist es beim Indexkalkül: Hier haben wir ganz wesentlich benutzt, daß sich die Elemente, deren diskrete Logarithmen wir suchen, als natürliche Zahlen darstellen lassen, so daß wir mit deren Primzerlegung arbeiten können. Die sollte nicht in allen Gruppen funktionieren.

b) Multiplikative Gruppen beliebiger endlicher Körper

Tatsächlich gibt es nicht nur für jede Primzahl p einen Körper mit p Elementen, sondern für jede Primzahl*potenz* $q = p^n$. Vor allem für den Fall $q = 2^8 = 256$ werden wir dies im nächsten Kapitel genauer betrachten.

Wie wir aus §3b wissen, bilden die von Null verschiedenen Elemente eines endlichen Körpers bezüglich der Multiplikation eine zyklische Gruppe; falls ihre Ordnung $q - 1 = p^n - 1$ prim ist oder einen großen Primteiler hat, lassen sich auch so sichere Kryptosysteme aufbauen.

In der Praxis kryptographischen Praxis spielen Potenzen ungerader Primzahlen allerdings kaum eine Rolle: Das Rechnen in den entsprechenden Körpern ist aufwendiger als das in einem vergleichbar großen Körper von Primzahlordnung, ohne daß dies zu einem Sicherheitsgewinn führen würde, da eine Variante des Indexkalküls bei beliebigen endlichen Körpern funktioniert.

Anders steht es mit Körpern von Zweipotenzordnung: Da Computer ohnehin im Zweiersystem rechnen, können sie damit sehr effizient umgehen. Es gibt sogar eine Reihe von Exponenten, für die $2^n - 1$ prim ist; für $n \leq 2\,500$ sind dies 2, 3, 5, 7, 13, 17, 19, 31, 61, 89, 107, 127, 521, 607, 1279, 2203 und 2281.

Grundsätzlich dürfte wohl die Sicherheit diskreter Logarithmen in einem Körper mit 2^n Elementen vergleichbar sein mit der eines Körpers, dessen Elementanzahl eine Primzahl derselben Größenordnung ist, allerdings gibt es sehr viel weniger Zweierpotenzen als Primzahlen, und nicht für alle n hat $2^n - 1$ einen großen Primteiler. Dadurch besteht die Gefahr, daß sich kriminelle Energie (sowie auch Nachrichtendienste) auf die wenigen interessanten Werte von n stürzen und dort mit Spezialhardware

arbeiten, was die Sicherheitssituation zu ihren Gunsten verschiebt. Somit dürften im allgemeinen Primzahlen vorzuziehen sein; hinzu kommt, daß zumindest DSA ohnehin nur für diesen Fall definiert ist.

c) Elliptische Kurven

In der Praxis ist für Kryptoverfahren auf der Basis diskreter Logarithmen abgesehen von den multiplikativen Gruppe der Körper von Primzahlpotenzordnung vor allem noch eine andere Art von Gruppe wichtig: die Gruppe der rationalen Punkte einer elliptischen Kurve über einem endlichen Körper. In bislang eher noch experimentellen Systemen arbeitet man auch mit höherdimensionalen Verallgemeinerungen davon, hauptsächlich den JACOBIschen hyperelliptischer Kurven. Da elliptische Kurven Gegenstand einer eigenen Vorlesung sind, soll hier nur kurz erläutert werden, um welche Art von Gruppe es sich bei einer elliptischen Kurven handelt.

Elliptische Kurven sind keine Ellipsen; sie haben ihren Namen davon, daß bei der Berechnung der Bogenlänge einer Ellipse algebraische Integrale auftreten, deren Integranden solche Kurven definieren.

Eine elliptische Kurve über einem Körper k ist eine ebene Kurve vom Grad drei; sie ist also gegeben durch ein Polynom f vom Grad drei mit Koeffizienten aus k in zwei Variablen x und y. Wir verlangen zusätzlich, daß sich das Polynom f auch über Erweiterungskörpern von k nicht als Produkt eines linearen und eines quadratischen Polynoms schreiben läßt, daß es mindestens eine Nullstelle $(x, y) \in k^2$ hat und daß die partiellen Ableitungen f_x und f_y für keine Lösung (x, y) der Gleichung $f(x, y) = 0$ über irgendeinem Erweiterungskörper von k simultan verschwinden.

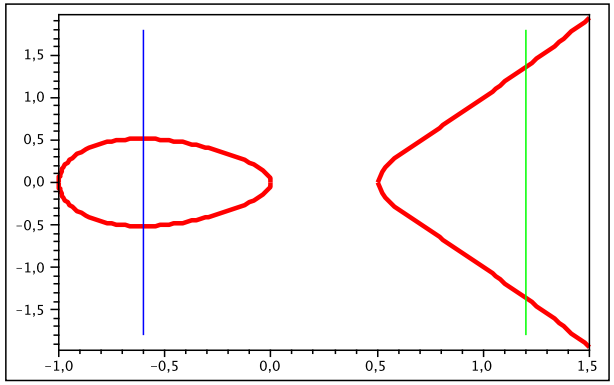
Geometrisch bedeuten diese Forderungen, daß die durch $f(x, y) = 0$ definierte Kurve nicht als Vereinigung einer Geraden und einer anderen Kurve geschrieben werden kann, daß sie mindestens einen Punkt mit Koordinaten aus k hat und daß es in jedem Punkt eine wohldefinierte Tangente gibt.

Schneiden wir eine solche Kurve mit einer Geraden, so erlaubt uns die Geradengleichung die Elimination einer der beiden Variablen x und y; was übrig bleibt ist eine höchstens kubische Gleichung in der anderen.

Damit ist klar, daß eine Gerade eine elliptische Kurve in höchstens drei Punkten schneidet.

Um besser zu sehen, was hier möglich ist, beschränken wir uns auf Kurven mit einer Gleichung der speziellen Form $y^2 = f(x)$, wobei $f(x)$ ein Polynom dritten Grades in x ist und nehmen außerdem an, daß der Körper k nicht den Körper $\mathbb{F}_2$ enthält. (Über einem solchen Körper läßt sich sogar für *jede* elliptische Kurve ein Koordinatensystem finden, in dem sie wie oben geschrieben werden kann; unsere Annahme ist also keine echte Einschränkung.) Die Bedingung, daß die partiellen Ableitungen nach x und y in keinem Kurvenpunkt *beide* verschwinden dürfen, besagt hier einfach, daß $f(x)$ keine mehrfache Nullstelle hat.

Falls $f(x_0)$ für ein $x_0 \in k$ eine Quadratwurzel $y_0 \in k$ hat, ist auch $-y_0$ eine; wenn $f(x_0)$ nicht verschwindet, gibt es also genau zwei Punkte mit x-Koordinate x_0. Die Verbindungsgerade dieser beiden Punkte ist natürlich $x = x_0$, und offensichtlich gibt es keinen dritten Schnittpunkt mit der Kurve.



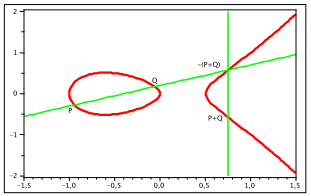
Um dieses Problem zu umgehen, ergänzen wir die Kurve durch einen weiteren Punkt O, der auf jeder Gerade der $x = x_0$ liegen soll; wir bezeichnen O als den „unendlich fernen“ Punkt in Richtung der y-Achse.

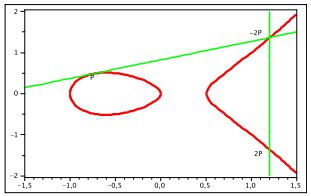
Im Fall zweier Punkte (x_1, y_1) und (x_2, y_2) mit $x_1 \neq x_2$ überzeugt man sich leicht, daß die Verbindungsgerade $y = \frac{y_2 - y_1}{x_2 - x_1}(x - x_1) + y_1$

die Kurve in (mit Vielfachheit gezählt) drei Punkten schneidet, denn schreiben wir die Geradengleichung in der Form $y = mx + b$ und setzen in die Kurvengleichung ein, erhalten wir die Gleichung dritten Grades $f(x) - (mx + b)^2 = 0$. Sie hat x_1 und x_2 als Nullstellen, und nach Division durch $(x - x_1)(x - x_2)$ bleibt eine lineare Gleichung für die dritte Lösung x_3 übrig.

Damit könnte man versuchen, eine Verknüpfung zu definieren, indem je zwei Punkten der dritte Schnittpunkt ihrer Verbindungsgeraden mit der Kurve zugeordnet wird. Diese Verknüpfung erfüllt aber leider abgesehen vom Kommutativgesetz keines der Gruppenaxiome.

Eine leichte Modifikation führt aber zu einer Gruppenstruktur: Wenn P, Q und R auf einer Geraden liegen, soll das nicht bedeuten, daß $R = P + Q$ ist, sondern daß $P + Q + R$ gleich dem Neutralelement O ist; somit ist also $R = -(P + Q)$. Da der unendlich ferne Punkt O so gewählt wurde, daß (x, y), $(-x, y)$ und O auf einer Geraden liegen, ist für $P = (x, y)$ das inverse Element einfach gleich $(-x, y)$; zur Berechnung von $P + Q$ muß also der dritte Schnittpunkt der Geraden durch P und Q noch an der x-Achse gespiegelt werden. (Im Falle $P = Q$ ist unter der Geraden durch P und Q natürlich die Tangente im Punkt P zu verstehen.)





Alle Gruppenaxiome mit Ausnahme des Assoziativgesetzes lassen sich leicht überprüfen; für letzteres ist jedoch einiges an zusätzlicher mathematischer Theorie notwendig, was den Rahmen dieser Vorlesung sprengen würde.

Die Punkte einer elliptischen Kurve können offensichtlich nicht mit ganzen Zahlen identifiziert werden, und von ihrer Primzerlegung können wir auch nicht reden. Damit fällt der Indexkalkül als Strategie zur Berechnung diskreter Logarithmen auf elliptischen Kurven weg.

Natürlich gibt es auch spezielle Methoden für die Berechnung diskreter Logarithmen auf elliptische Kurven; in machen Fällen kann man sie sogar zurückführen auf die Berechnung diskreter Logarithmen in der multiplikativen Gruppe eines nicht sehr viel größeren Körpers. Wer mit den theoretischen Grundlagen der Kryptographie mit elliptischen Kurven vertraut ist, kann aber zumindest die bekannten Angriffsmethoden so erschweren, daß der entsprechende Erweiterungskörper schon für relativ kleine Grundkörper so groß ist, daß dort nach heutigem Kenntnisstand auch klassische diskrete Logarithmenprobleme nicht effizient gelöst werden können. Die Empfehlungen der Bundesnetzagentur sehen hier daher für die Kryptographie mit elliptischen Kurven deutlich kleinere Primzahlen vor als im klassischen Fall. Wie dort gibt es zwei Primzahlen p und q: Die elliptische Kurve ist definiert über einem Körper mit p Elementen, und die Unterschrift liegt in einer Untergruppe der Ordnung q der elliptischen Kurve. Wie beim DSA muß q eine Länge von derzeit mindestens 224 Bit haben, ab 2016 sind es 250. Für die Primzahl p dagegen gibt es keinerlei Einschränkung, außer daß $p \neq q$ sein muß und es auf natürlich auch eine elliptische Kurve über $\mathbb{F}_p$ geben muß, auf der ein Punkt der Ordnung q liegt. Zur Erschwerung der oben erwähnten Angriffsmethode wird außerdem gefordert, daß es kein $r \leq 10^4$ geben darf, so daß q ein Teiler von $p^r - 1$ ist; die Gruppe, in der die Unterschriften liegen, soll sich also nicht in die multiplikative Gruppe einer „kleinen“ Erweiterung des Grundkörpers einbetten lassen. Außerdem soll noch die Hauptordnung, die zum Endomorphismenring der Kurve gehört, eine Klassenzahl von mindestens 200 haben – wie man sieht, steckt in der Kryptographie mit elliptischen Kurve einiges mehr an Mathematik als in den klassischen Verfahren, so daß Einzelheiten im Rahmen dieser Vorlesung nicht behandelt werden können.

§5: Literatur

Diskrete Logarithmensysteme über endlichen Körper werden in denselben Büchern behandelt wie RSA; hierfür sei daher auf die Literaturangaben zum vorigen Kapitel verwiesen. Bücher, die auch Verfahren auf der Grundlage elliptischer und (teilweise) hyperelliptischer Kurven betrachten sind

NEAL KOBLITZ: A Course in Number Theory and Cryptography, *Graduate Texts in Mathematics* **114**, *Springer*, [2]1994

und

NEAL KOBLITZ: Algebraic Aspects of Cryptography, *Springer*, 1998

Als ersten Überblick über elliptische Kurven kann man etwa das Buch

ANNETTE WERNER: Elliptische Kurven in der Kryptographie, *Springer*, 2002,

konsultieren, das eine elementare Einführung in die Theorie elliptischer Kurven unter dem Gesichtspunkt der Kryptographie geht. Beweise sind nicht immer vollständig, und anspruchsvollere Algorithmen werden nur sehr kurz oder gar nicht behandelt.

Deutlich anspruchsvoller und vollständiger sind

DARREL HANKERSON, ALFRED MENEZES, SCOTT VANSTONE: Guide to Elliptic Curve Cryptography, *Springer*, 2004

LAWRENCE C. WASHINGTON: Elliptic Curves – Number Theory and Cryptography, *Chapman & Hall/CRC*, 2003

IAN BLAKE, GADIEL SEROUSSI, NIGEL SMART: Elliptic Curves in Cryptography, *London Mathematical Society Lecture Notes Series* **265**, *Cambridge University Press*, 1999

IAN BLAKE, GADIEL SEROUSSI, NIGEL SMART [HRSG.:] Advances in Elliptic Curve Cryptography, *London Mathematical Society Lecture Notes Series* **317**, *Cambridge University Press*, 2005

Die vollständigste und ausführlichste Information bietet derzeit wohl

HENRI COHEN, GERHARD FREY, ROBERTO AVANZI, CHRISTOPHE DOCHE, TANJA LANGE, KIM NGUYENM FREDERIK VERCAUTEREN: Handbook of Elliptic and Hyperelliptic Curve Cryptography, *Chapman & Hall/CRC*, 2006

Speziell mit Implementierungsfragen beschäftigt sich

MICHAEL ROSING: Implementing Elliptic Curve Cryptography, *Manning*, 1999