Das Faktorisierungsrepräsentationsproblem als Basis kryptographischer Protokolle

Dissertation
zur Erlangung des Doktorgrades
der Naturwissenschaften

vorgelegt beim Fachbereich Mathematik
der Johann Wolfgang Goethe-Universität
in Frankfurt am Main

—

von
Roger Fischlin
aus Offenbach am Main

—

Frankfurt am Main 2002

D F 1

Vom Fachbereich Mathematik der Johann Wolfgang Goethe-Universität
als Dissertation angenommen.

Dekan:	Prof. Dr. J. Baumeister
Gutachter:	Prof. Dr. C.P. Schnorr Prof. Dr. M. Sieveking
Datum der Disputation:	11.September 2002

Zusammenfassung

Okamoto (Crypto 1992) hat die RSA-Repräsentation als Basis eines gegen aktive Angreifer sicheren Identifikationsschemas eingeführt. Eine RSA-Repräsentation von $X \in \mathbb{Z}_N^*$ ist ein Paar $(x, r) \in \mathbb{Z}_e \times \mathbb{Z}_N^*$ mit $X \equiv g^x r^e \pmod{N}$ für vorgegebenes $g \in \mathbb{Z}_N$, RSA-Modul N und primen RSA-Exponenten e. Das zugehörige Repräsentationsproblem, also das Auffinden eines Wertes X samt zweier verschiedener Darstellungen, ist äquivalent zum RSA-Problem, der Berechnung einer e-ten Wurzel von g modulo N. Von Brassard, Chaum und Crépeau (Journal Computing System Science, 1988) sowie Damgård (Journal of Cryptology, 1995) stammt eine analoge Konstruktion der Form $X \equiv g^x r^{2^t} \pmod{N}$ mit $x \in \mathbb{Z}_{2^t}$ für den Spezialfall der Blum-Zahlen als Modul N und gegebenes $t \geq 1$, wo die Möglichkeit, zwei verschiedene Repräsentationen zu berechnen, gleichbedeutend zur Zerlegung des Moduls in die Primfaktoren ist. Im ersten Abschnitt der vorliegenden Arbeit verallgemeinern wir dieses Konzept systematisch auf beliebige (RSA-)Module durch die Einführung eines Anpassungsparameters $\tau := \tau(N)$, so dass $X \equiv g^x r^{2^{\tau+t}} \pmod{N}$ mit $x \in \mathbb{Z}_{2^t}$. Basierend auf dieser als Faktorisierungsrepräsentation bezeichneten Darstellung leiten wir Identifikations-, Signatur- und Blinde-Unterschriften-Verfahren her.

Im zweiten Teil verwenden wir sowohl RSA- als auch Faktorisierungsrepräsentation als Grundlage sogenannter non-malleable Commitment-Schemata zur Hinterlegung (Verbriefung) einer geheimen Nachricht. Bei dem von Dolev, Dwork und Naor (SIAM Journal on Computing, 2000) eingeführten Begriff der Non-Malleability soll ein Angreifer außer Stande sein, die Hinterlegung einer Nachricht m so abzuändern, dass er diese später dann mit einem in Relation zu m stehenden Wert, man denke zum Beispiel an $m+1$, aufdecken kann. Von Dolev, Dwork und Naor stammt ein allgemeiner Ansatz zur Konstruktion von non-malleable Commitment-Schemata aufbauend auf einem sogenannten Knowledge-Extraktor. Für die RSA-Darstellung verfügt das von Okamoto entworfene Protokoll als Proof-Of-Knowledge über einen solchen Extraktor, bei dem im Fall der Faktorisierungsrepräsentation von uns entwickelten Verfahren fehlt allerdings der Extraktor. Aus diesem Grund stellen wir mit Hilfe des Chinesischen Restsatzes ein neues, auf Commitments zugeschnittenes Protokoll mit Knowledge-Extraktor vor, das in Verbindung mit der Faktorisierungsrepräsentation ein effizientes Hinterlegungsschema ergibt. Zum Abschluß wird bei einem Commitment-Verfahren mit abgeschwächter Non-Malleability-Eigenschaft von Di Crescenzo, Katz, Ostrovsky und Smith (Eurocrypt 2001) die RSA- durch die Faktorisierungsrepräsentation ersetzt und das Schema vereinfacht.

Vorwort

Die vorliegende Dissertation bildet den Abschluß der vierjährigen Mitarbeit von 1998 bis 2001 in der Gruppe von Prof. Dr.Schnorr am Fachbereich Mathematik der J.W.Goethe-Universität. Während dieser Zeit sind mit anderen Mitarbeitern des Lehrstuhls drei gemeinsame Arbeiten entstanden:

- mit Jean-Pierre Seifert: *Tensor-Based Trapdoors for CVP and Their Applications to Public-Key Cryptography*, Cryptography and Coding, 1999.
- mit Marc Fischlin: *Efficient Non-Malleable Commitment Schemes*, Crypto 2000.
- mit Marc Fischlin: *The Representation Problem Based on Factoring*, Cryptographer's Track, RSA Konferenz 2002.

Diese Dissertation beruht im wesentlichen auf den von mir beigetragenen Teilen zu den beiden gemeinsamen mit meinem Bruder Marc verfaßten Arbeiten. Das gemeinsame Paper mit Jean-Pierre hat keinen Eingang in die Dissertation gefunden. Die über das Tensor-Produkt konstruierten Gitterbasen mit Geheimtüre für das Closest-Vector-Problem stehen in keinem direkten Zusammenhang mit denen für die non-malleable Hinterlegungsverfahren verwandten RSA-, Faktorisierungs- oder Diskrete-Logarithmus-Repräsentationen.

In Kapitel 1 werden grundlegende, zum Verständnis der beiden anschließenden Kapitel notwendige Definitionen zusammengefaßt und anhand des RSA-Repräsentationsproblems und des davon abgeleiteten Identifikationsschemas von Okamoto verdeutlicht. Ein wichtiger Punkt in Kapitel 2 ist die Einführung der Faktorisierungsrepräsentation und der Beweis, wonach die Möglichkeit, zwei verschiedene Darstellungen eines Wertes zu finden äquivalent zur Zerlegung des Moduls in die Primfaktoren ist. Wir übertragen die von der RSA-Darstellung bekannten Identifikations-, Signatur- und Blinde-Unterschriften-Verfahren und passen diese an die Besonderheiten der Faktorisierungsrepräsentation an. Kapitel 3 steht im Zeichen von non-malleable

Commitment-Schemata. Wir fassen den Non-Malleable-Gedanken und das auf der RSA-Repräsentation basierende Verfahren in eigenen Worten zusammen, um anschließend eine Konstruktion basierend auf dem Chinesischen Restsatz zu geben, mit welcher auch (wie in der Kryptographie üblich) statt der eigentlichen Nachricht ihr Hashwert hinterlegt wird. Dieser Ersatz des Proof-Of-Knowledge erlaubt auch die Faktorisierungsrepräsentation als Ausgangspunkt, obwohl im Gegensatz zur RSA-Darstellung das entsprechende Protokoll kein Proof-Of-Knowledge ist und der für den Non-Malleability-Beweis zentrale Extraktor fehlt.

An dieser Stelle möchte ich mich bei meinem Bruder Marc für die Zusammenarbeit bedanken. Es sollen auch diejenigen erwähnt werden, die ich im Zusammenhang mit meiner Arbeit am Lehrstuhl kennen- und schätzen gelernt habe, sei es als kompetente Gesprächspartner, talentierte Sportler beim traditionellen Crypto-Fußballspiel, oder „einfach“ als Freunde: Stephan Brands, Roland Cramer, Cynthia Dwork, Rosario Gennaro, Shai Halevi, Nick Howgrave-Graham, Danielle Miccancio, Phong Nguyen, Tal Rabin, Omer Reingold, Alon Rosen und Jean-Pierre Seifert. Bedanken möchte ich mich auch bei meiner Familie, Prof. Dr. Schnorr, der letztlich mein Interesse an der Kryptographie geweckt hat, sowie der Hermann-Willkomm-Stiftung für die finanzielle Unterstützung bei Teilnahme an Konferenzen.

Roger Fischlin, Februar 2002.

Inhaltsverzeichnis

Kapitel 1

Einleitung

Cryptography & Intractability:
A match made in heaven.

— Shafi Goldwasser

Wir fassen grundlegende, zum Verständnis der anschließenden Kapitel notwendige Definitionen zusammen und verdeutlichen diese anhand des RSA-Repräsentationsproblems und des davon abgeleiteten Identifikationsschemas von Okamoto.

1.1 Definitionen

Man nennt einen Algorithmus $\mathcal{A}$ *Polynomialzeitverfahren* oder kurz *effizient*, wenn dessen Laufzeit $|\mathcal{A}|$ durch ein Polynom $p(n)$ in der Eingabelänge n beschränkt ist. Die Laufzeit bezieht sich auf die Anzahl der Bitoperationen, so dass zum Beispiel die Addition zweier n-Bit-Zahlen in $\mathcal{O}(n)$ Schritten zu bewerkstelligen ist. Sprechen wir hingegen von arithmetischen Schritten, zählt jede elementare Operation wie u.a. die Addition als ein Schritt.

In der Kryptographie umfaßt die Menge der Polynomialzeit-Algorithmen auch probabilistische Verfahren, bei denen den Algorithmen zusätzlich interne Münzwürfe bzw. Zufallsbits zur Verfügung stehen und die Laufzeit im Erwartungswert durch jeweils ein Polynom $p(n)$ beschränkt ist. Im Fall eines probabilistischen Verfahrens $\mathcal{A}$ steht $|\mathcal{A}|$ für den Erwartungswert der Laufzeit. Dieser als auch Wahrscheinlichkeitsaussagen zur Ausgabe des Algorithmus' beziehen sich (sofern nicht anders vermerkt) jeweils sowohl auf die zufällige Eingabe als auch die internen Münzwürfe. Sind die Zufallsbits ω fixiert, schreiben wir $\mathcal{A}_\omega$ für das dann deterministische Verfahren. Die Wahrscheinlichkeit, dass die Laufzeit eines probabilistischen Polynomialzeit-Algorithmus $\mathcal{A}$ eine vorgegebene Schrittzahl $t(n)$ überschreitet, läßt sich

mittels Markoff-Ungleichung durch den Quotienten $\frac{|\mathcal{A}|}{t(n)}$ nach oben abschätzen. Sollte $\mathcal{A}$ mit Wahrscheinlichkeit π die gewünschte Ausgabe liefern, so ändert sich folglich die Wahrscheinlichkeit maximal um den Faktor $1 - \frac{1}{\delta}$, bricht man die Berechnung nach $t(n) := \delta\pi^{-1}|\mathcal{A}|$ Schritten ab. Auf diese Weise erhalten wir für polynomielles $\pi(n)$ mit $\delta = 2$ aus $\mathcal{A}$ ein probabilistisches Verfahren mit höchstens halbierter Erfolgswahrscheinlichkeit, dessen Laufzeit stets und nicht nur im Erwartungswert polynomiell ist. Man spricht in diesem Fall von *strikter Polynomialzeit*. Zur Vereinfachung unterstellt man, dass ein solcher Algorithmus $\mathcal{A}$ in jedem Schritt ein Zufallsbit verwendet, dementsprechend ω eine Folge von $|\mathcal{A}|$ zufällig und unabhängig gewählten Bits ist.

Die Sicherheit kryptographischer Systeme wird im Durchschnitt betrachtet. Die Laufzeit jedes potentiellen Angreifers $\mathcal{A}$ soll im Erwartungswert polynomiell und dessen Erfolgswahrscheinlichkeit unbedeutend oder vernachlässigbar klein sein. Allgemein nennen wir eine Funktion $\nu : \mathbb{N} \to [0, 1]$ *vernachlässigbar* (engl. negligible), wenn zu jedem positiven Polynom $p(n) \geq 0$ ein $n_0 \in \mathbb{N}$ existiert, so dass für alle $n \geq n_0$ gilt $\nu(n) < \frac{1}{p(n)}$. Man sagt umgangssprachlich, eine solche Funktion wie beispielsweise $\nu(n) := 2^{-n}$ verschwinde (engl. to vanish) schneller als jeder polynomieller Bruchteil. Das Produkt zweier vernachlässigbarer Funktionen ist ebenfalls vernachlässigbar, ebenso das Produkt eines positiven Polynoms $p(n)$ und eines vernachlässigbaren $\nu(n)$. Aus diesem Grund kann man sich bei der Sicherheitsanalyse auf Angreifer $\mathcal{A}$ mit strikter polynomieller Laufzeit beschränken, weil die Transformation eines Algorithmus' in einen mit strikter polynomieller Laufzeit die Klassifizierung der Erfolgswahrscheinlichkeit nicht ändert.

Sollte eine Funktion $\pi(n)$ nicht vernachlässigbar sein, gibt es ein Polynom $p(n)$, so dass für unendlich viele n gilt $\pi(n) \geq \frac{1}{p(n)}$. Diese Eigenschaft bleibt auch nach Subtraktion einer vernachlässigbaren Funktion $\nu(n)$ von $\pi(n)$ erhalten. Durch unabhängige Wiederholungen eines probabilistischen Polynomialzeit-Verfahrens $\mathcal{A}$, das mit Wahrscheinlichkeit $\pi(n) > 0$ die gewünschte Ausgabe liefert, sind wir im Erwartungswert nach $\pi^{-1}|\mathcal{A}|$ Schritten erfolgreich (geometrische Verteilung), was allerdings für ein vernachlässigbares π außerhalb der Polynomialzeit-Schranke liegt. Im Fall der Erfolgswahrscheinlichkeit $\pi(n) = 1 - \nu(n)$ für ein vernachlässigbares $\nu(n)$, sagen wir, $\mathcal{A}$ sei *nahezu immer* (with overwhelming probability) erfolgreich. Allgemein nennen wir zwei Wahrscheinlichkeiten *nahezu gleich*, sofern der Absolutbetrag ihrer Differenz vernachlässigbar klein ist.

Bitstring ist ein Synonym für einen Vektor $\mathbf{a} \in \{0, 1\}^n$ und $\mathbf{a} \oplus \mathbf{b}$ bezeichne die (bitweise) XOR-Verknüpfung von $\mathbf{a}, \mathbf{b} \in \{0, 1\}^n$. Faßt man einen Bitstring als Vektor über $\mathbb{F}_2$ auf, entspricht die XOR-Operation der Addition im Vektorraum $\mathbb{F}_2^n$. Ein Wert $a \in \mathbb{Z}$ heißt n-Bitzahl oder hat die Bitlänge

n, wenn für die Darstellung des Absolutbetrags n Bits hinreichend als auch notwendig sind:

$$2^{n-1} \leq |a| < 2^n.$$

Wir schreiben $\mathbb{Z}_m$ für den Faktorring $\mathbb{Z}/m\mathbb{Z}$ und identifizieren die Restklassen modulo m durch die Zahlen $\{0, 1, \ldots, m-1\}$. $\mathbb{Z}_m^*$ ist die multiplikative Gruppe bestehend aus den zu m teilerfremden Restklassen:

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

Deren Anzahl, die Gruppenordnung von $\mathbb{Z}_m^*$, ist gegeben durch die *Euler-Funktion* $\varphi(m)$. Sei $m = \prod_{i=1}^r p_i^{e_i}$ die Primfaktorzerlegung von m. Aus dem *Chinesischen Restsatz* (CRT)

$$\mathbb{Z}_m^* \simeq \mathbb{Z}_{p_1^{e_1}}^* \times \mathbb{Z}_{p_2^{e_2}}^* \times \cdots \times \mathbb{Z}_{p_r^{e_r}}^*$$

folgt $\varphi(m) = \prod_{i=1}^r \varphi(p_i^{e_i})$. Für Primzahlen $p > 2$ und $e \geq 1$ gilt

$$\varphi(p^e) = p^{e-1}(p-1).$$

Die Gruppe $\mathbb{Z}_{p^e}^*$ ist zyklisch: Es gibt ein erzeugendes Element g (auch als Primitivwurzel oder Generator bezeichnet), so dass modulo p^e die $\varphi(p^e)$ Potenzen $g^0, g^1, g^2, \ldots$ sämtliche Restklassen von $\mathbb{Z}_{p^e}^*$ durchlaufen. Die Ordnung $\mathrm{ord}_m(a)$ eines Elementes $a \in \mathbb{Z}_m^*$ ist das minimale $k \in \mathbb{N}$ mit $a^k \equiv 1 \pmod{m}$. Diese teilt stets die Gruppenordnung.

Eine Restklasse $x \in \mathbb{Z}_m^*$ heißt *quadratischer Rest*, wenn der Repräsentant modulo m als Quadrat darstellbar ist. Diese bilden eine Untergruppe von $\mathbb{Z}_m^*$:

$$\mathrm{QR}_m := \{a^2 \bmod m \mid a \in \mathbb{Z}_m^*\}.$$

Für eine Primzahl $p > 2$ und $e \geq 1$ gibt es modulo p^e genau zwei Wurzeln und folglich ist QR_{p^e} eine Untergruppe mit Index $[\mathbb{Z}_{p^e}^* : \mathrm{QR}_{p^e}] = 2$, denn zu $a^2 \in \mathrm{QR}_{p^e}$ sind — weil $\mathbb{Z}_{p^e}^*$ zyklisch ist — die Wurzeln gegeben durch $a^x \bmod p^e$ mit $2x \equiv 1 \pmod{\varphi(p^e)}$:

Satz 1.1.1 (Lösungsanzahl Kongruenz). *Sei m ein Modul und $a, b \in \mathbb{Z}_m$. Die Kongruenz $ax \equiv b \pmod{m}$ ist genau dann lösbar, wenn $\mathrm{ggT}(a, m) | b$, wobei es exakt $\mathrm{ggT}(a, m)$-viele Lösungen $x \in \mathbb{Z}_m$ gibt.*

Da $\varphi(p^e) = p^{e-1}(p-1)$ gerade ist, hat $2x \equiv 1 \pmod{\varphi(p^e)}$ genau zwei Lösungen. Bei einer zyklischen Gruppe G ungerader Ordnung existiert hingegen exakt ein $x \in [0, |G|)$ mit $2x \equiv 1 \pmod{|G|}$, so dass folglich gilt:

Korollar 1.1.2. *Sei G eine zyklische Gruppe ungerader Ordnung. Dann ist Quadrieren eine Permutation auf G.*

Sei $p > 2$ prim. Für $\mathbb{Z}_{p^e}^*$ sind zu einem Generator $g \in \mathbb{Z}_{p^e}^*$ die quadratischen Reste exakt diejenigen $g^x \bmod p^e$ mit geraden Exponenten x modulo $\varphi(p^e)$. Wegen $2|\varphi(p^e)$ ist daher ein Produkt $ab \in \mathbb{Z}_{p^e}^*$ genau dann ein quadratischer Rest, wenn entweder $a, b \in \mathrm{QR}_{p^e}$ oder $a, b \notin \mathrm{QR}_{p^e}$. Falls -1 kein quadratischer Rest modulo p^e ist, liegt zu jedem $a \in \mathbb{Z}_{p^e}^*$ exakt einer der Werte $\pm a$ in QR_{p^e}. Aufgrund des Euler-Kriteriums ist -1 genau dann ein quadratischer Rest modulo $p > 2$, sofern $p \equiv 1 \pmod 4$. Als Konsequenz entspricht modulo $p \equiv 3 \pmod 4$ Quadrieren auf QR_p einer Permutation.

Mit dem erweiterten Euklidischen oder binären ggT-Algorithmus kann man zu $a, b \in \mathbb{Z}$, $|a|, |b| < 2^n$, Koeffizienten $u, v \in \mathbb{Z}$ mit $ua + vb = \mathrm{ggT}(a, b)$ in $\mathcal{O}(n)$ Iterationen bestimmen [K98, 4.5.2]:

Satz 1.1.3. *Gegeben $a, b \in \mathbb{Z}$ mit $0 < |a|, |b| < 2^n$. Dann sind in $\mathcal{O}(n)$ arithmetischen Schritten auf $\mathcal{O}(n)$-Bit-Zahlen $u, v \in \mathbb{Z}$ mit $ua + vb = \mathrm{ggT}(a, b)$ zu berechnen.*

Sei $N = pq$ das Produkt zweier, verschiedener Primzahlen $p, q > 2$. Nach Chinesischem Restsatz gilt für die Ordnung der Gruppe $\mathbb{Z}_N^* \simeq \mathbb{Z}_p^* \times \mathbb{Z}_q^*$:

$$\varphi(N) = (p-1)(q-1) = N - p - q + 1.$$

Weil $p, q > 2$ prim sind, teilt 2 sowohl $p - 1$ und $q - 1$, so dass $\varphi(N)$ gerade ist. Bei Primzahlen p, q gleicher Bitlänge ist die Wahrscheinlichkeit, dass ein zufälliger Wert a aus $\mathbb{Z}_N$, wir schreiben kurz $a \in_{\mathsf{R}} \mathbb{Z}_N$, nicht in $\mathbb{Z}_N^*$ liegt, vernachlässigbar in der Bitlänge n des Moduls:

$$\mathrm{Ws}_{a \in_{\mathsf{R}} \mathbb{Z}_N}[a \notin \mathbb{Z}_N^*] = 1 - \frac{|\mathbb{Z}_N^*|}{|\mathbb{Z}_N|} = 1 - \frac{N - p - q + 1}{N} \leq \frac{2 \cdot 2^{\frac{n}{2}}}{2^{n-1}} = 2^{2 - \frac{n}{2}}.$$

Die Notation „$a \in_{\mathsf{R}} M$“ verwenden wir im weiteren, falls a uniform und unabhängig aus der endlichen (Multi-)Menge M gewählt wird. Die Schreibweise „$a_1, a_2, \ldots, a_k \in_{\mathsf{R}} M$“ bedeutet, dass wir k Elemente $a_1, a_2, \ldots, a_k$ aus M uniform und unabhängig wählen.

Seien $X = (X_n)_{n \in \mathbb{N}}$ und $Y = (Y_n)_{n \in \mathbb{N}}$ zwei Folgen von Verteilungen (sogenannte *Ensembles*) auf Mengen, deren Elemente jeweils durch in n polynomiell viele Bits darstellbar sind. Wir sagen vereinfachend, die Verteilung X_n habe eine bestimmte Eigenschaft, sofern die zugrundeliegende Folge $(X_n)_{n \in \mathbb{N}}$ aus dem Kontext hervorgeht. So nennt man eine Verteilung X_n *effizient erzeugbar* (polynomially samplable), wenn es einen probabilistischen Polynomialzeit-Algorithmus gibt, der zur Eingabe 1^n (Eins unär kodiert, also n Einsen) ein gemäß X_n verteiltes Element $x \leftarrow X_n$ erzeugt,

genauer die Verteilung von x zumindest statistisch nah (siehe unten) an X_n ist. Beispielsweise ist die uniforme Verteilung $\{0,1\}^n$ effizient erzeugbar, der Algorithmus gibt einfach n Zufallsbits aus.

Die Kryptographie kennt verschiedene Auslegungen des Begriffs der Ununterscheidbarkeit von Verteilungen X_n und Y_n, je nach Grad spricht man von perfekter, statistischer oder Polynomialzeit-Ununterscheidbarkeit. Im Fall von *perfekter Ununterscheidbarkeit* sind beide Verteilungen identisch. X_n und Y_n heißen *statistisch nah* (statistically close) oder *statistisch ununterscheidbar* (statistically indistinguishable), wenn

$$\tfrac{1}{2}\sum_z \left|\mathrm{Ws}[z \leftarrow X_n] - \mathrm{Ws}[z \leftarrow Y_n]\right|$$

vernachlässigbar in n ist (der Faktor $\frac{1}{2}$ dient zur Normierung). Bezogen auf n-Bit-Module N ist beispielsweise die Verteilung des Wertes $(z \bmod N)$ für $z \in_R [0, 2^{2n}]$ statistisch nah an der uniformen Verteilung auf $\mathbb{Z}_N$. Im Fall $N = pq$ für zwei Primzahlen p, q gleicher Bitlänge erhalten wir auf diese Weise eine zur Gleichverteilung auf $\mathbb{Z}_N^*$ statisch nahe Verteilung. Eine schwächere Forderung ist die der Polynomialzeit-Ununterscheidbarkeit. Zwei Verteilungen X_n und Y_n heißen *polynomialzeit-ununterscheidbar* (computational indistinguishable), falls kein probabilistischer Algorithmus (Unterscheider, engl. Distinguisher) $\mathcal{D}$ in der Lage ist, durch $\{0,1\}$-Ausgabe zu entscheiden, ob die Eingabe gemäß X_n oder Y_n verteilt ist, formal der Absolutbetrag

$$\left|\mathrm{Ws}[\mathcal{D}(x) = 1] - \mathrm{Ws}[\mathcal{D}(y) = 1]\right|$$

vernachlässigbar klein ist (wobei sich die Wahrscheinlichkeiten auf die internen Münzwürfe und die Eingabe $x \leftarrow X_n$ bzw. $y \leftarrow Y_n$ beziehen). Statistische (perfekte) Ununterscheidbarkeit impliziert, dass kein Polynomialzeit-Distinguisher beide Verteilungen unterscheiden kann, die Umkehrung gilt hingegen in der Regel nicht [Go98]. Die statistische und insbesondere auch die perfekte Ununterscheidbarkeit sind stärkere Forderungen als die Computational-Indistinguishability.

1.2 RSA-Repräsentation

Eines der ersten Public-Key-Verschlüsselungsschemas stammt von Rivest, Shamir und Adleman [RSA78] aus dem Jahr 1978. Ein sogenannter *RSA-Modul* $N = pq$ ist das Produkt zweier großer, unterschiedlicher Primzahlen p und q gleicher Bitlänge. Ein zugehöriger *RSA-Exponent* $e \in \mathbb{N}$ soll teilerfremd zu $\varphi(N)$ sein, damit die RSA-Funktion

$$\mathrm{RSA}_{N,e}(m) := m^e \bmod N$$

eine Permutation auf der Gruppe $\mathbb{Z}_N^*$ ist. Bei Kenntnis der Faktorisierung oder äquivalent $\varphi(N)$ kann man $d := e^{-1} \bmod \varphi(N)$ berechnen und so die RSA-Funktion invertieren, denn

$$\mathrm{RSA}_{N,e}(m)^d \equiv m^{ed} \equiv m^{1+k\varphi(N)} \equiv m \pmod{N}$$

für $ed = 1 + k\varphi(N)$ und passendes $k \in \mathbb{Z}$. Invertieren der RSA-Funktion bedeutet das Ziehen der e-ten Wurzel modulo N. Dies ist trivialerweise ohne Kenntnis des Secret-Keys d möglich, wenn $e \equiv \pm 1 \pmod{\varphi(N)}$. Der geheime Schlüssel d erlaubt in Verbindung mit dem Public-Key e, über das Vielfache $ed - 1$ von $\varphi(N)$ den Modul zu faktorisieren. Allerdings ist offen, ob Invertieren der RSA-Funktion gleichbedeutend zur Kenntnis des Secret-Keys d ist. Nach Wiener [W90] liefert die Kettenbruchentwicklung zu $\frac{e}{N}$ im Fall geheimer Schlüssel $d < N^{0,25}$ diesen Secret-Key und als Konsequenz die Primfaktorzerlegung des Moduls N. Boneh und Durfee [B00] haben unter heuristischen Annahmen die Schranke auf $N^{0,292}$ verbessert. Bei diesem Angriff erlangt man Kenntnis des geheimen Schlüssels und bricht das komplette Schema. Ein Verfahren von Coppersmith [Co97, B99] liefert in Polynomialzeit die Nachricht m einer Verschlüsselung $\mathrm{RSA}_{N,e}(m)$, wenn man über die obersten $(1 - \frac{1}{e}) \log_2 N$ Bits von m verfügt. Mit Blick auf diesen möglichen Schwachpunkt wird größeren RSA-Exponenten der Vorzug gegeben. Neben der Faktorisierung des Moduls N, auf die wir näher in Abschnitt 2.1 eingehen, sind keine weiteren nennenswerten Angriffe bekannt [B99].

Beim RSA-Problem soll zu gegebenem Tupel (N, e, g) bestehend aus RSA-Modul N, zugehörigem RSA-Exponenten e und $g \in \mathbb{Z}_N^*$ die eindeutig bestimmte e-te Wurzel von g berechnet werden. Abgesehen von den oben erwähnten Exponenten e gilt das RSA-Problem für zufälligen, hinreichend großen RSA-Modul N, beliebigen RSA-Exponenten e zu N und unabhängig, uniform verteiltes $g \in_R \mathbb{Z}_N^*$ als schwierig. Wir beschränken uns für unsere Anwendungen (im Hinblick auf Sicherheitsbeweise) auf prime Exponenten. Sei $\mathrm{RSAIndex}(\cdot)$ ein effizienter Index-Generator, der zur Eingabe 1^n in Polynomialzeit einen n-Bit-RSA-Modul N, einen primen RSA-Exponenten e zu N und ein uniform verteiltes $g \in_R \mathbb{Z}_N^*$ erzeugt: $(N, e, g) \leftarrow \mathrm{RSAIndex}(1^n)$. Bei einer naiven Implementierung des Index-Generators $\mathrm{RSAIndex}(\cdot)$ generiert man einen n-Bit-RSA-Modul N bzw. die beiden Primfaktoren, indem wiederholt zufällige Werte mit $\frac{n}{2}$ Bits einem Primzahltest unterworfen werden. Im Durchschnitt finden wir nach $\mathcal{O}(n)$ vielen Versuchen eine Primzahl [CP01, W96]. Für große Bitlängen ist dieser Ansatz jedoch zu zeitaufwendig, stattdessen konstruiert man iterativ Primzahlen wie bei den Verfahren von Maurer [M95] oder Cramer und Shoup [CrSh00].

Definition 1.2.1 (RSA-Annahme). *Die Erfolgswahrscheinlichkeit jedes pro-*

babilistischen Polynomialzeit-Algorithmus' $\mathcal{A}$ für das RSA-Problem

$$\mathrm{Ws}\left[\mathcal{A}(N,e,g) \equiv g^{\frac{1}{e}} \pmod N\right]$$

sei vernachlässigbar in n, wobei sich die Wahrscheinlichkeit auf die Wahl $(N,e,g) \leftarrow \mathsf{RSAIndex}(1^n)$ und die internen Münzwürfe von $\mathcal{A}$ bezieht.

Eine solche Funktion wie RSA nennt man *Oneway-Funktion*[1], weil zwar die Abbildung

$$(N,e,g^{\frac{1}{e}} \bmod N) \mapsto (N,e,g)$$

effizient zu berechnen ist, unter der RSA-Annahme die Bestimmung eines Urbildes, hier der e-ten Wurzel von g modulo N, in Polynomialzeit allerdings bestensfalls mit vernachlässigbarer Wahrscheinlichkeit bezüglich Eingabe $(N,e,g) \leftarrow \mathsf{RSAIndex}(1^n)$ und interner Münzwürfe gelingt. Man spricht formal korrekterweise von einer Candidated-Oneway-Funktion, weil diese Oneway-Eigenschaft unter einer Komplexitätsannahme, hier der RSA-Annahme, steht.

Mit dem RSA-Repräsentationsproblem von Okamoto [Ok92] beschreiben wir im weiteren eine zu RSA äquivalente Aufgabe, die als Basis kryptographischer Bausteine wie eines Indentifikations-, Unterschriften- und ein Hinterlegungsschema dient. Eine *RSA-Repräsentation* für $X \in \mathbb{Z}_N^*$ bezüglich $(N,e,g) \leftarrow \mathsf{RSAIndex}(1^n)$ ist ein Paar $(x,r) \in \mathbb{Z}_e \times \mathbb{Z}_N^*$ mit:

$$X \equiv g^x r^e \pmod N.$$

Jedes $X \in \mathbb{Z}_N^*$ hat genau e Darstellungen bezüglich (N,e,g), denn zu $x \in \mathbb{Z}_e$ gibt es exakt ein $r \in \mathbb{Z}_N^*$ mit $r^e \equiv Xg^{-x} \bmod N$. Geht der Bezug auf (N,e,g) aus dem Kontext hervor, sagen wir kurz, (x,r) sei eine RSA-Repräsentation von X. Wir nennen zwei RSA-Darstellungen (x_1,r_1) und (x_2,r_2) verschieden, wenn $(x_1,r_1) \neq (x_2,r_2)$. Weil die e-te Wurzel modulo N eindeutig bestimmt ist, folgt aus $x_1 \neq x_2$, dass r_1 und r_2 ebenfalls ungleich sind.

Definition 1.2.2 (RSA-Repräsentationsproblem). *Zu gegebenem Tupel $(N,e,g) \leftarrow \mathsf{RSAIndex}(1^n)$ berechne ein $X \in \mathbb{Z}_N^*$ samt zweier verschiedener Repräsentationen $(x_1,r_1),(x_2,r_2) \in \mathbb{Z}_e \times \mathbb{Z}_N^*$ bezüglich (N,e,g).*

Das RSA-Repräsentationsproblem ist trivial, wenn wir die e-te Wurzel $g^{\frac{1}{e}} \bmod N$ kennen, da für beliebiges $a \in \mathbb{Z}$ gilt:

$$g^x r^e \equiv X \equiv g^{x+a}\left(g^{-\frac{a}{e}} r\right)^e \pmod N.$$

[1] Im Deutschen ist auch die Bezeichnung als Einweg-Funktion üblich, wenngleich es sich übersetzt um eine Einbahnstraße und weniger um eine zum einmaligen Gebrauch bestimmte Abbildung handelt.

Man sagt, das Repräsentationsproblem sei auf RSA reduzierbar, denn mit einem Algorithmus für RSA ist das zugehörige Darstellungsproblem effizient zu lösen. Weil wir dabei das gegebene Verfahren zur Berechnung e-ter Wurzeln modulo N als Blackbox betrachten, also uns nicht für die interne Vorgehensweise interessieren, spricht man von einem *Orakel* für das RSA-Problem.

Definition 1.2.3 (Reduktion kryptographischer Probleme). *Seien Λ und Φ kryptographische Probleme. Λ ist auf Φ reduzierbar, wenn man mit Hilfe eines probabilistischen Polynomialzeit-Algorithmus', der Φ mit nicht vernachlässigbarer Wahrscheinlichkeit löst, das Problem Λ ebenfalls in Polynomialzeit mit nicht vernachlässigbarer Wahrscheinlichkeit löst. Lassen sich die beiden Probleme Λ und Φ wechselseitig aufeinander reduzieren, heißen sie äquivalent.*

Sei Λ auf Φ reduzierbar. Weil umgangssprachlich formuliert Λ nicht schwieriger als Φ ist, verwendet man in der Komplexitätstheorie die anschauliche Notation „$\Lambda \leq \Phi$“ und für äquivalente Probleme „$\Lambda \equiv \Phi$“. Bei Sicherheitsbeweisen führt man das Brechen des neuen Systems auf das Lösen kryptographischer Standardprobleme, von denen man annimmt, sie seien schwierig, zurück. Entscheidend für die Beurteilung der Reduktion aus kryptographischer Sicht sind die Veränderungen von Laufzeit und Erfolgswahrscheinlichkeit [Lu96]. Bleiben beide Charakteristiken zumindest weitgehend erhalten — bis auf moderate konstante Faktoren —, nennen wir die Reduktion $\Lambda \leq \Phi$ *sicherheitserhaltend*. Eine konkrete Sicherheitsannahme, d.h. für das Problem Λ existiere kein Algorithmus mit Laufzeit T und Erfolgswahrscheinlichkeit π, überträgt sich nahezu exakt auf Φ. Bei nicht sicherheitserhaltender Reduktionen muß man hingegen teilweise signifikante Absenkungen des (beweisbaren) exakten Sicherheitsniveaus hinnehmen, wenngleich diese Verfahren asymptotisch sicher sind.

Wir haben bereits gesehen, das RSA-Repräsentationsproblem ist sicherheitserhaltend auf RSA reduzierbar, d.h. die Lösung des Darstellungsproblems kann man auf die Berechnung einer e-ten Wurzel von g modulo N zurückführen. Die Umkehrung gilt auch, beide Probleme sind für prime RSA-Exponenten äquivalent [Ok92]:

Satz 1.2.4 (Okamoto 1992). *Der Algorithmus $\mathcal{A}$ löse das RSA-Repräsentationsproblem zu $(N, e, g) \leftarrow \mathsf{RSAIndex}(1^n)$ mit Wahrscheinlichkeit π. Dann kann man in nahezu gleicher Zeit (siehe unten) und mit gleicher Wahrscheinlichkeit das RSA-Problem zu (N, e, g) lösen, also $g^{\frac{1}{e}} \bmod N$ berechnen.*

Beweis. Seien (x_1, r_1) und (x_2, r_2) zwei verschiedene RSA-Repräsentationen zu $X \in \mathbb{Z}_N^*$. Setze $\Delta x := x_1 - x_2 \neq 0$ und $r := r_2 r_1^{-1} \bmod N$. Es

gilt:

$$g^{\Delta x} \equiv g^{x_1 - x_2} \equiv r_2^e r_1^{-e} \equiv r^e \pmod{N}.$$

Aufgrund $1 \leq |\Delta x| < e$ ist der prime Exponent e teilerfremd zu Δx und es existieren $u, v \in \mathbb{Z}$ mit $u\Delta x + ve = 1$:

$$g \equiv g^{u\Delta x + ve} \equiv (g^{\Delta x})^u \cdot (g^v)^e \equiv (r^u g^v)^e \pmod{N}.$$

Löst $\mathcal{A}$ das RSA-Repräsentationsproblem zu (N, e, g), kann man mit dem erweiterten Euklidischen Algorithmus u, v berechnen und erhält in Form von $r^u g^v$ eine e-te Wurzel von g modulo N. □

Bei der Formulierung von Satz 1.2.4 sprechen wir von nahezu gleicher Laufzeit und umschreiben damit den zusätzlichen Aufwand für den erweiterten Euklidischen Algorithmus und die Exponentationen. Für die Sicherheitsbetrachtung sind diese im Vergleich zur anzunehmenden Komplexität des Algorithmus' $\mathcal{A}$ unbedeutend, weshalb wir bei nachfolgenden Sätzen jeweils die Laufzeit für linear viele zusätzliche arithmetische Operationen auf Zahlen der Bitlänge $\mathcal{O}(n)$ vernachlässigen und von nahezu gleicher Laufzeit sprechen.

Während die Reduktion von RSA auf das zugehörige Repräsentationsproblem für beliebige Exponenten gilt, haben wir nur für prime Exponenten das Repräsentationsproblem auf RSA zurückgeführt. Die im Beweis zu Satz 1.2.4 verwandte Methode mit Hilfe des erweiterten Euklidischen Algorithmus' eine e-te Wurzel zu berechnen, halten wir als Korollar fest:

Korollar 1.2.5. *Sei e prim. Gegeben $a, b \in \mathbb{Z}_N^*$ sowie $\Delta \in \mathbb{Z}$ mit $\Delta \not\equiv 0 \pmod{e}$ und $a^\Delta \equiv b^e \pmod{N}$ kann man ohne Kenntnis der Faktorisierung des n-Bit-Moduls N eine e-te Wurzel von $a \in \mathbb{Z}_N^*$ in $\mathcal{O}(n)$ arithmetischen Schritten berechnen.*

1.3 Identifikationsschema

Ein Identifikationsschema $\langle \mathcal{P}, \mathcal{V} \rangle_{\mathsf{Id}}$ ist ein interaktives Protokoll zwischen zwei Parteien, in dem der sogenannte *Prover* $\mathcal{P}$ die Kenntnis eines Secret-Keys sk zum Public-Key pk und damit seine Identität gegenüber dem *Verifier* $\mathcal{V}$ beweist. Formal sind beide Seiten, in der Kryptographie oftmals mit den Namen Alice und Bob belegt, interaktive, probabilistische Turing-Maschinen mit einem gemeinsamen Kommunikationsband, das wechselseitig beschrieben wird. Man fordert, dass die Laufzeit beider Parteien strikt polynomiell bezogen auf den sogenannten *Sicherheitsparameter* n ist.

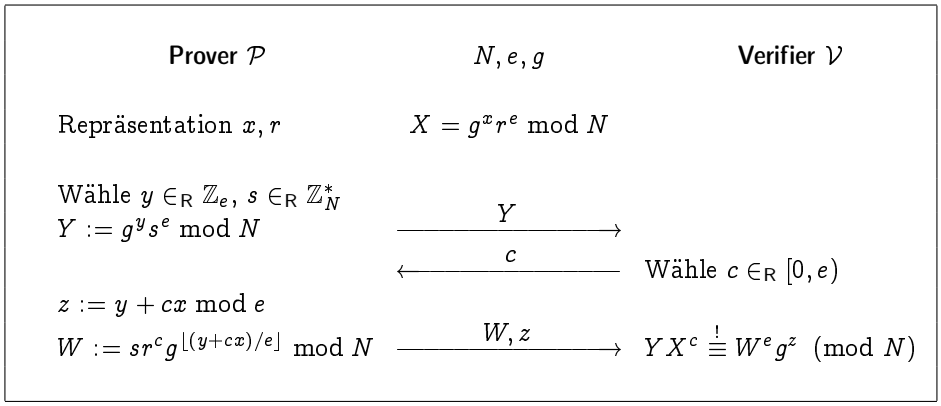
Für die Schlüsselgenerierung gibt eine *Dritte Partei* $\mathcal{T}$ zum Sicherheitsparameter n systemweit allgemeine Daten σ, die sogenannten *Common-* oder *Public-Parameters*, vor. Die Erzeugung des Schlüsselpaares $(\mathsf{sk}, \mathsf{pk})$ mit $\mathsf{sk} \in \mathsf{SecretKey}(\mathsf{pk})$ übernimmt der Prover $\mathcal{P}$ oder wie im folgenden angenommen, ebenfalls die Dritte Partei $\mathcal{T}$. Beide Teilnehmer, $\mathcal{P}$ und $\mathcal{V}$, erhalten als Eingabe die Public-Parameters σ, der Prover $\mathcal{P}$ zusätzlich das Schlüsselpaar $(\mathsf{sk}, \mathsf{pk})$ und der Verifier $\mathcal{V}$ den Public-Key pk. Von der Dritten Partei $\mathcal{T}$ wird vorausgesetzt, dass sie „ehrlich“ (honest) ist und die Werte gemäß Spezifikation wählt. Weil die Teilnehmer der Vorgabe von $\mathcal{T}$ vertrauen, spricht man auch von einer *Trusted-Party*. Der Prover $\mathcal{P}$ erhält von dieser ein Zertifikat, das den Public-Key pk seiner realen Identität, zum Beispiel dem Namen ALICE, zuordnet. Jeder Verifier kann anhand dieser Urkunde nachprüfen, ob es tatsächlich ALICE ist, die im Protokoll ihre Identität nachweist.

Ein Prover $\mathcal{P}(\sigma, \mathsf{pk}, \mathsf{sk})$ *besteht* das Protokoll, wenn ein ehrlicher Verifier $\mathcal{V}(\sigma, \mathsf{pk})$ die Identifikation akzeptiert. Als symbolische Schreibweise für eine erfolgreiche Ausführung schreiben wir $\langle \mathcal{P}(\sigma, \mathsf{pk}, \mathsf{sk}), \mathcal{V}(\sigma, \mathsf{pk}) \rangle_{\mathsf{Id}} \in \mathsf{Accept}$ oder, wenn die Eingabe aus dem Kontext hervorgeht, kurz $\langle \mathcal{P}, \mathcal{V} \rangle_{\mathsf{Id}} \in \mathsf{Accept}$. Die Wahrscheinlichkeit, mit welcher ein Prover das Protokoll besteht, bezieht sich stets auf die Vorgabe durch die Trusted-Party $\mathcal{T}$ als auch interne Münzwürfe. Weicht der Prover respektive der Verifier möglicherweise von der Spezifikation des Protokolls ab, verwenden wir $\tilde{\mathcal{P}}$ und $\tilde{\mathcal{V}}$ als Kennzeichnung. Gezieltes Fehlverhalten ist denkbar bei einem betrügerischen Prover $\tilde{\mathcal{P}}$, um ohne Kenntnis des Secret-Keys das Protokoll zu bestehen, oder einem Verifier $\tilde{\mathcal{V}}$, um Informationen über den geheimen Schlüssel von $\mathcal{P}$ zu erlangen.

Wir beschreiben im folgenden das Identifikationsschema $\langle \mathcal{P}, \mathcal{V} \rangle_{\mathsf{RSA}}$ von OKAMOTO [Ok92], welches auf dem in Abschnitt 1.2 vorgestellten RSA-Repräsentationsproblem basiert. Die Trusted-Party $\mathcal{T}$ gibt als Public-Parameters $\sigma := (N, e, g) \leftarrow \mathsf{RSAIndex}(1^n)$ vor und wählt als geheimen Schlüssel $\mathsf{sk} := (x, r) \in_{\mathsf{R}} \mathbb{Z}_e \times \mathbb{Z}_N^*$. Der Public-Key pk zu (x, r) ist $X := g^x r^e \bmod N$. Mit anderen Worten, zum öffentlichen Schlüssel X entspricht die Menge $\mathsf{SecretKey}((N, e, g), X)$ der geheimen Schlüssel den RSA-Repräsentationen von X bezüglich der Public-Parameters (N, e, g). Der Prover kann, muß aber weder die e-te Wurzel von g noch die Faktorisierung von N kennen. Ohne dieses Wissen können mehrere Teilnehmer (Prover) den gleichen Modul N verwenden. Wir setzen zur Vereinfachung voraus, der Kehrwert $\frac{1}{e}$ des öffentlichen RSA-Exponenten sei vernachlässigbar im Sicherheitsparameter und erläutern später, wie das Schema für andere Exponenten anzupassen ist.

Der Prover $\mathcal{P}$ beweist seine Identität gegenüber dem Verifier $\mathcal{V}$, in-

Abbildung 1.1: Identifikation $\langle \mathcal{P}, \mathcal{V} \rangle_{\mathsf{RSA}}$ basierend auf RSA-Repräsentation



dem beide Parteien das in Abbildung 1.1 angegebene Protokoll $\langle \mathcal{P}, \mathcal{V} \rangle_{\mathsf{RSA}}$ ausführen. Dieses hat die auf FIAT und SHAMIR [FS86, FFS88] zurückgehende Three-Move-Struktur eines Identifikationsschemas:

1. Commitment: $\mathcal{P}$ hinterlegt die beiden Werte (y, s) in Form von Y.
2. Challenge: $\mathcal{V}$ stellt eine zufällige Frage c.
3. Response: $\mathcal{P}$ gibt die Antwort (W, z) auf die Challenge c.

Der Prover $\mathcal{P}$ besteht das Protokoll, wenn die Kontrollgleichung $Y X^c \stackrel{!}{\equiv} W^e g^z \pmod N$ bezüglich Public-Key X, Commitment Y, Challenge c und Response (W, z) erfüllt ist. Damit der Prover $\mathcal{P}$ durch seine Antwort nicht den geheimen Schlüssel offenbart, hängt die Response neben Secret-Key auch von den im ersten Schritt hinterlegten Werten (y, s) ab.

Bei Kenntnis einer RSA-Repräsentation (x, r) von X ist der Prover $\mathcal{P}$ in der Lage, sich gegenüber einem Verifier $\mathcal{V}$, der sich an die Spezifikation des Protokolls $\langle \mathcal{P}, \mathcal{V} \rangle_{\mathsf{RSA}}$ aus Abbildung 1.1 hält, zu identifizieren:

$$\begin{aligned} Y X^c &\equiv (g^y s^e)(g^x r^e)^c \\ &\equiv s^e r^{ce} g^{y+cx} \\ &\equiv (s r^c g^{\lfloor (y+cx)/e \rfloor})^e \cdot g^{(y+cx) \bmod e} \\ &\equiv W^e g^z \pmod N. \end{aligned} \tag{1.1}$$

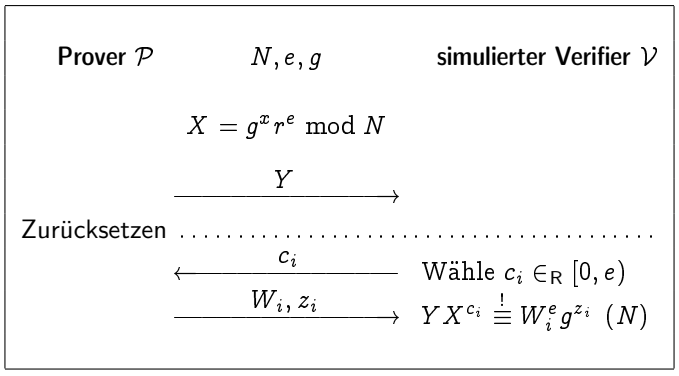
Diese Eigenschaft eines Identifikationsschemas heißt *Completeness*. Umgekehrt soll jeder, der das Protokoll als Prover besteht einen passenden Secret-Key, hier eine RSA-Repräsentation zu X, kennen (*Soundness*). Zu

einem Prover $\tilde{\mathcal{P}}$, der in (strikter) Polynomialzeit das Protokoll mit nicht vernachlässigbarer Wahrscheinlichkeit besteht, können wir eine RSA-Repräsentation zum Public-Key X effizient berechnen, d.h. eine Darstellung aus $\tilde{\mathcal{P}}$ extrahieren.

Satz 1.3.1 (Knowledge-Extraktor, Okamoto 1992). *Der Prover $\tilde{\mathcal{P}}$ bestehe das Identifikationsschema $\langle \mathcal{P}, \mathcal{V} \rangle_{\mathsf{RSA}}$ aus Abbildung 1.1 mit Wahrscheinlichkeit π. Dann existiert ein probabilistischer Polynomialzeit-Algorithmus (Knowledge-Extraktor) $\mathcal{E}$, der eine RSA-Repräsentation zu X bezüglich (N, e, g) in erwarteter Laufzeit $2 \cdot |\tilde{\mathcal{P}}|$ mit Wahrscheinlichkeit $\pi - \frac{1}{e}$ bestimmt.*

Beweis. Wir skizzieren den Beweis [Ok92]. Der Knowledge-Extraktor $\mathcal{E}$ hat volle Kontrolle über den Prover $\tilde{\mathcal{P}}$, weshalb er in der Lage ist, dessen Ausführung schrittweise zu simulieren und die Programmzustände zu speichern, um später den Ablauf an diese Stellen zurückzusetzen. Diesen Vorgang nennt man auch *reseten*, angelehnt an den englischen Ausdruck „Reset“ für Zurücksetzen.

Abbildung 1.2: Knowledge-Extraktor für $\langle \mathcal{P}, \mathcal{V} \rangle_{\mathsf{RSA}}$



Algorithmus $\mathcal{E}$ simuliert eine Protokollausführung mit dem Prover $\tilde{\mathcal{P}}$ und übernimmt die Rolle des Verifiers $\mathcal{V}$, wobei der Zustand nach Senden von Y gespeichert wird. Sollte $\tilde{\mathcal{P}}$ das Protokoll nicht bestehen, dann stoppe. Sonst setze $\tilde{\mathcal{P}}$ an die Stelle zurück, an welcher er im Anschluß an sein Commitment Y auf die Challenge c wartet, und wiederhole die Ausführung mit neuer, unabhängiger Challenge, bis $\tilde{\mathcal{P}}$ erneut erfolgreich ist (Vergleiche Abbildung 1.2). Auf diese Weise erhält man zwei erfolgreiche Protokollausführungen (Y, c_1, W_1, z_1) und (Y, c_2, W_2, z_2) für dasselbe Commitment Y:

$$\begin{aligned} YX^{c_1} &\equiv W_1^e g^{z_1} \pmod{N} \\ YX^{c_2} &\equiv W_2^e g^{z_2} \pmod{N}. \end{aligned} \tag{1.2}$$

Sei $\Delta c = c_1 - c_2$ die Differenz der beiden Challenges. Wir zeigen, wie man unter der Prämisse $\Delta c \neq 0$ aus (1.2) eine RSA-Repräsentation zu X extrahiert. Durch Kombination der beiden Kongruenzen (1.2) erhalten wir mit $\Delta z := z_1 - z_2$:

$$X^{\Delta c} \equiv (W_1 W_2^{-1})^e g^{\Delta z} \pmod{N}. \tag{1.3}$$

Aufgrund $0 < |\Delta c| < e$ ist Δc teilerfremd zu e und es gibt $u, v \in \mathbb{Z}$ mit $u\Delta c + ve = 1$. Wir erheben (1.3) in die u-te Potenz:

$$X^{u\Delta c} \equiv (W_1^u W_2^{-u})^e g^{u\Delta z} \pmod{N}.$$

Setze $x' := (u\Delta z) \bmod e$, so dass $u\Delta z - x' \equiv 0 \pmod{e}$. Wegen $u\Delta c + ve = 1$ gilt

$$\begin{aligned} X &\equiv X^{u\Delta c + ve} \\ &\equiv (W_1^u W_2^{-u} X^v)^e g^{u\Delta z} \\ &\equiv g^{x'} \big(\underbrace{W_1^u W_2^{-u} X^v g^{(u\Delta z - x')/e}}_{:= r'} \big)^e \pmod{N} \end{aligned}$$

und wir erhalten eine RSA-Repräsentation (x', r') zu X. Im Fall $\Delta c = 0$ ist der zuvor beschriebene Knowledge-Extraktor $\mathcal{E}$ nicht erfolgreich.

Analysieren wir zum Abschluß Laufzeit und Erfolgswahrscheinlichkeit des oben beschriebenen Extraktors $\mathcal{E}$. Sei $\pi(\gamma)$ mit $\gamma := (N, e, g, X, \omega)$ die Wahrscheinlichkeit, mit welcher $\tilde{\mathcal{P}}$ für fixierte Public-Parameters (N, e, g), öffentlichen Schlüssel X und interne Zufallsbits ω (umfaßt insbesondere das gesandte Y) erfolgreich ist. Das Tupel γ entspricht einem Präfix des Protokollablaufs und $\pi(\gamma)$ gibt an, mit welcher Wahrscheinlichkeit (bezogen auf die zufällige Wahl der Challenge) $\tilde{\mathcal{P}}$ diese Teilausführung erfolgreich beendet. Für ein gegebenes γ stoppt der Extraktor $\mathcal{E}$ mit Wahrscheinlichkeit $1 - \pi(\gamma)$ nach $|\tilde{\mathcal{P}}|$ Schritten ohne Ausgabe, sonst ist $\pi(\gamma) > 0$ und es schließen sich im Erwartungswert $\pi(\gamma)^{-1}|\tilde{\mathcal{P}}|$ Schritte an. Zusammenfassend gilt für die erwartete Laufzeit des Knowledge-Extraktors $\mathcal{E}$:

$$\begin{aligned} |\mathcal{E}| &= \sum_{\gamma} (|\tilde{\mathcal{P}}| + \pi(\gamma) \cdot \pi(\gamma)^{-1} |\tilde{\mathcal{P}}|) \cdot \mathrm{Ws}[\text{Präfix } \gamma] \\ &= 2 \cdot |\tilde{\mathcal{P}}| \sum_{\gamma} \mathrm{Ws}[\text{Präfix } \gamma] \\ &= 2 \cdot |\tilde{\mathcal{P}}|. \end{aligned}$$

Weil mit Wahrscheinlichkeit $\frac{1}{e}$ die Differenz Δc gleich Null ist, extrahieren wir mit Wahrscheinlichkeit mindestens $\pi - \frac{1}{e}$ eine Repräsentation. □

Im Fall $\pi \leq \frac{1}{e}$ liefert der Extraktor nicht unbedingt eine Repräsentation. Dies ist nicht überraschend, denn $\frac{1}{e}$ entspricht der trivialen Erfolgswahrscheinlichkeit, das Protokoll ohne Kenntnis einer Repräsentation durch Raten der Challenge $c \in [0, e)$ zu bestehen:

1. Im ersten Schritt wählt der betrügerische Prover $\widetilde{\mathcal{P}}$ zusätzlich $\tilde{c} \in_R [0, e)$ und sendet $\tilde{Y} := YX^{-\tilde{c}}$ statt Y.

2. Unter der Voraussetzung, $\tilde{c}$ stimmt mit der Challenge c überein, besteht $\widetilde{\mathcal{P}}$ das Protokoll durch die Antworten $\tilde{y} := y$ und $\widetilde{W} := s$.

Folglich ist es für die Sicherheit des Schemas entscheidend, dass sich $\mathcal{P}$ auf Y vor Kenntnis der Challenge c festlegt. Bei unseren Sicherheitsbetrachtungen schließen wir Exponenten e mit nicht-vernachlässigbarem Kehrwert aus. Für diese wiederholt man das Protokoll über $m = \mathcal{O}(n)$ Runden, die triviale Erfolgswahrscheinlichkeit eines betrügerischen $\widetilde{\mathcal{P}}$ sinkt auf e^{-m} und der Extraktor liefert mit Wahrscheinlichkeit etwa $\pi - e^{-m}$ die Repräsentation [BG92, Theorem 5.2]. Die für kurze Challenges notwendigen mehreren Runden führen im Vergleich zu 1-Runden-Protokollen zu signifikanten Effizienznachteilen.

Die Laufzeit des in Satz 1.3.1 konstruierten Extraktors $\mathcal{E}$ ist im Erwartungswert polynomiell, man spricht von einem *liberalen Knowledge-Extraktor*. Eine generische Transformation des Verfahrens in ein solches mit strikter polynomieller Laufzeit ist zwar möglich, dann sinkt aber die Wahrscheinlichkeit des Auffindens einer Repräsentation. Ein anderes Konzept ist das des ϵ-*Extraktors* [HaMi98], wo für alle vorgegebenen Parameter $\epsilon > 0$ das Verfahren in strikter Polynomialzeit bezüglich n und ϵ^{-1} mit der Erfolgswahrscheinlichkeit π des Provers $\widetilde{\mathcal{P}}$ abzüglich ϵ und eines vernachlässigbaren ν eine Darstellung extrahiert. Über die Vorgabe $\epsilon > 0$ bestimmen wir, wie nah die Extraktionswahrscheinlichkeit an der Erfolgswahrscheinlichkeit des Provers $\mathcal{P}$ liegen soll. Sei γ wie im Beweis zu Satz 1.3.1 definiert und Γ umfasse alle γ mit $\pi(\gamma) \geq \frac{1}{2}\epsilon$. Der Extraktor wiederholt den Challenge-Response-Schritt bis zu $8\epsilon^{-1}\ln(2\epsilon^{-1})$-mal. Für die Analyse verwende die *Chernoff-Schranke*, wonach zu einer Folge $I_1, I_2, \ldots, I_k$ unabhängiger, identisch verteilter Indikatorvariablen mit $\mathrm{Ws}[I_i = 1] \geq p$ für alle $\delta \in (0, 1)$ gilt [MoRa95]:

$$\mathrm{Ws}\left[\sum_{i=1}^{k} I_i < (1-\delta)kp\right] \leq e^{-\frac{1}{2}kp\delta^2}. \tag{1.4}$$

Vorausgesetzt $\gamma \in \Gamma$, ist dann die Wahrscheinlichkeit, dass unter den $k := 8\epsilon^{-1}\ln(2\epsilon^{-1})$ Wiederholungen weniger als zwei Erfolge sind, gemäß Chernoff-Schranke (mit $p = \frac{1}{2}\epsilon$ und $\delta = \frac{1}{2}$) maximal $\frac{1}{2}\epsilon$. Für $\gamma \notin \Gamma$ bestimmt der

Extraktor zwar nicht notwendigerweise eine Repräsentation, aber in diesem Fall ist die Erfolgswahrscheinlichkeit von $\tilde{\mathcal{P}}$ kleiner als $\frac{1}{2}\epsilon$. Addition der Abweichungen in beiden Fällen ergibt die gewünschte Vorgabe:

$$\begin{aligned}
&\mathrm{Ws}[\langle\tilde{\mathcal{P}},\mathcal{V}\rangle_{\mathsf{RSA}} \in \mathsf{Accept}] - \mathrm{Ws}[\mathcal{E}\text{ erfolgreich}] \\
&\leq \mathrm{Ws}\left[\begin{array}{l}\langle\tilde{\mathcal{P}},\mathcal{V}\rangle_{\mathsf{RSA}} \in \mathsf{Accept} \\ \mathcal{E}\text{ nicht erfolgreich}\end{array}\middle|\, \gamma \in \Gamma\right] + \mathrm{Ws}\left[\begin{array}{l}\langle\tilde{\mathcal{P}},\mathcal{V}\rangle_{\mathsf{RSA}} \in \mathsf{Accept} \\ \mathcal{E}\text{ nicht erfolgreich}\end{array}\middle|\, \gamma \notin \Gamma\right] \\
&< (\tfrac{1}{2}\epsilon + \tfrac{1}{e}) + \tfrac{1}{2}\epsilon \\
&= \epsilon + \tfrac{1}{e}.
\end{aligned}$$

Zur Identifikation beweist der Prover $\mathcal{P}$ dem Verifier $\mathcal{V}$, dass $\mathcal{P}$ einen zu den öffentlichen Werten (σ, pk) passenden Secret-Key $\mathsf{sk} \in \mathsf{SecretKey}(\sigma, \mathsf{pk})$ kennt. Diese Beziehung entspricht einer binären Relation R, im Fall von Okamotos Identifikationsschema $\langle\mathcal{P},\mathcal{V}\rangle_{\mathsf{RSA}}$:

$$\mathsf{R} := \left\{ (\langle N, e, g, X\rangle, \langle x, r\rangle) \;\middle|\; X \equiv g^x r^e \pmod{N} \right\}.$$

Das Protokoll $\langle\mathcal{P},\mathcal{V}\rangle_{\mathsf{RSA}}$, Abbildung 1.1, nennt man einen *Proof-Of-Knowledge* für diese Relation [FS90, BG92]. Für die allgemeine Definition eines Proof-Of-Knowledge sei $\mathsf{R} = \{(y,z)\}$ eine effizient überpüfbare Relation, $\mathcal{P}(y,z)$ bezeichne den Prover mit Eingabe (y,z) und $\mathcal{V}(y)$ einen Verifier mit Eingabe y. Sei

$$\mathrm{witness}_{\mathsf{R}}(y) := \{z \mid (y,z) \in \mathsf{R}\}$$

die Menge der sogenannten *Zeugen* (witness) zu y, wobei wir implizit die (in der Länge von y) polynomielle Bitlänge der Zeugen voraussetzen. Im Fall der Identifikation $\langle\mathcal{P},\mathcal{V}\rangle_{\mathsf{RSA}}$ sind die Zeugen zu $y := (N,e,g,X)$ die verschiedenen RSA-Repräsentationen von $X \in \mathbb{Z}_N^*$ bezüglich (N,e,g).

Definition 1.3.2 (Proof-Of-Knowledge). *Ein Proof-Of-Knowledge für die in Polynomialzeit überprüfbare Relation* R *ist ein Protokoll* $\langle\mathcal{P},\mathcal{V}\rangle_{\mathsf{POK}}$ *mit folgenden Eigenschaften:*

a) Completeness: Für $(y,z) \leftarrow \mathcal{T}(1^n)$ *besteht ein Prover* $\mathcal{P}(y,z)$ *das Protokoll* $\langle\mathcal{P}(y,z),\mathcal{V}(y)\rangle_{\mathsf{POK}}$ *nahezu immer, d.h. bis auf vernachlässigbare Wahrscheinlichkeit.*

b) Soundness: Es gibt einen probabilistischen Polynomialzeit-Algorithmus (Knowledge-Extraktor) $\mathcal{E}$*, so dass für alle* $\tilde{\mathcal{P}}$ *und alle* $\tilde{z} := \tilde{z}(y)$ *gilt: Besteht* $\tilde{\mathcal{P}}$ *das Protokoll* $\langle\tilde{\mathcal{P}}(y,\tilde{z}),\mathcal{V}(y)\rangle_{\mathsf{POK}}$ *mit Wahrscheinlichkeit* π*, dann berechnet* $\mathcal{E}$ *zu* $(y,\tilde{\mathcal{P}})$ *einen Zeugen* $z \in \mathrm{witness}_{\mathsf{R}}(y)$ *mit Wahrscheinlichkeit* $\pi - \nu$ *für ein vernachlässigbares* ν*.*

Das Identifikationsschema $\langle \mathcal{P}, \mathcal{V} \rangle_{\mathsf{RSA}}$ von OKAMOTO ist sicher gegen *Angriffe aus dem Stand*, ein Angreifer $\mathcal{A}_{\mathsf{Adhoc}}$ (engl. Adversary), der ad hoc versucht, sich als Prover auszugeben, hat nach Satz 1.3.1 nur eine vernachlässigbare Erfolgswahrscheinlichkeit:

Korollar 1.3.3 (Okamoto 1992). *Unter der RSA-Annahme ist das Identifikationsschema $\langle \mathcal{P}, \mathcal{V} \rangle_{\mathsf{RSA}}$ aus Abbildung 1.1 sicher gegen Angriffe aus dem Stand, denn besteht $\mathcal{A}_{\mathsf{Adhoc}}$ das Protokoll mit Wahrscheinlichkeit π, so kann man in erwarteter Laufzeit $2 \cdot |\mathcal{A}_{\mathsf{Adhoc}}|$ mit Wahrscheinlichkeit mindestens $(1 - \frac{1}{e})(\pi - \frac{1}{e})$ die e-te Wurzel von g mod N berechnen.*

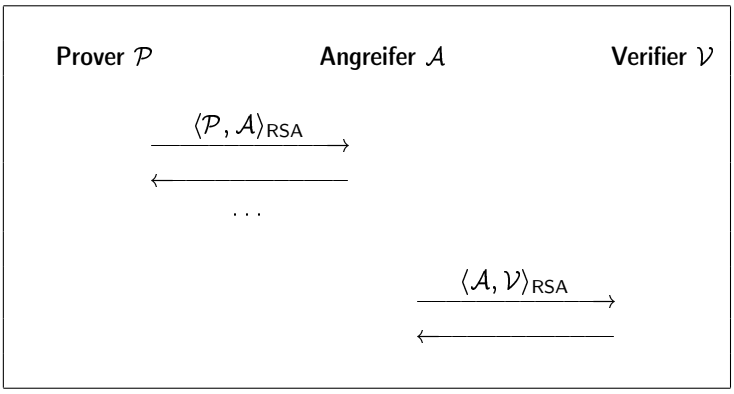
Beweis. Wir geben die Beweisidee [Ok92]. Angenommen, es gäbe einen erfolgreichen Adhoc-Angreifer $\mathcal{A}_{\mathsf{Adhoc}}$. Wir simulieren eine Trusted-Party $\mathcal{T}$: Zu $(N, e, g) \leftarrow \mathsf{RSAIndex}(1^n)$ wähle einen zufälligen Secret-Key $(x_1, r_1) \in_{\mathsf{R}} \mathbb{Z}_e \times \mathbb{Z}_N^*$ und entsprechenden Public-Key $X := g^{x_1} {r_1}^e \bmod N$. Die simulierte Umgebung hat keinen Einfluß auf Verhalten und insbesondere Erfolgswahrscheinlichkeit von $\mathcal{A}_{\mathsf{Adhoc}}$, weil die Verteilung der Daten identisch mit der des realen Umfelds (Setting) ist. Gemäß Satz 1.3.1 extrahiere aus $\mathcal{A}_{\mathsf{Adhoc}}$ eine Repräsentation (x_2, r_2) von X. Der Public-Key X läßt keine Rückschlüsse auf die von uns gewählte Repräsentation zu, die beiden Darstellungen (x_1, r_1) und (x_2, r_2) sind unabhängig. Es gibt exakt e verschiedene RSA-Repräsentationen zu X. Wir verfügen mit Wahrscheinlichkeit $1 - \frac{1}{e}$ in Form von (x_1, r_1) und (x_2, r_2) über zwei verschiedene Repräsentationen zu X, was nach Satz 1.2.4 das Ziehen der e-ten Wurzel von g mod N ermöglicht — Widerspruch zur RSA-Annahme. □

Dieser Sicherheitsbeweis soll auf *aktive Angriffe* erweitert werden. Bei dieser Form eines Angriffs, dargestellt in Abbildung 1.3, übernimmt der Angreifer $\mathcal{A}$ zunächst die Rolle eines Verifiers $\tilde{\mathcal{V}}$ gegenüber dem echten Prover $\mathcal{P}$ und führt das Protokoll $\langle \mathcal{P}, \mathcal{A} \rangle_{\mathsf{RSA}}$ polynomiell oft aus. Im Anschluß an diese „Lernphase“ versucht $\mathcal{A}$, sich gegenüber einem Verifier $\mathcal{V}$ als Prover auszugeben: $\langle \mathcal{A}, \mathcal{V} \rangle_{\mathsf{RSA}}$. Die Beweisidee zu Korollar 1.3.3 ist auf diese Situation übertragbar, denn mit Hilfe des Secret-Keys (x_1, r_1) sind wir in der Lage, die Protokollausführungen mit $\mathcal{A}$ als Verifier zu simulieren und so den aktiven Angriff auf $\mathcal{A}_{\mathsf{Adhoc}}$ zurückzuführen. Nachzuweisen bleibt, dass die extrahierte RSA-Repräsentation (x_2, r_2) unabhängig von der gewählten Darstellung (x_1, r_1) ist, also die Kommunikation des ehrlichen Provers $\mathcal{P}$ mit $\mathcal{A}$ bzw. einem unehrlichen Verifier $\tilde{\mathcal{V}}$ nicht von der gewählten Repräsentation zu X abhängt.

Definition 1.3.4 (View). *Zu einem Protokoll $\langle \mathcal{P}, \mathcal{V} \rangle$ bezeichnet man als View* $\mathrm{view}(\langle \mathcal{P}, \mathcal{V} \rangle)$ *die Zufallsvariable bestehend aus*

- *den von der Trusted-Party vorgegebenen Werten,*

Abbildung 1.3: Aktiver Angriff



- *den internen Münzwürfen beider Parteien sowie*
- *der Kommunikation einer Protokollausführung.*

Beim View der Partei $\mathcal{P}$ bzw. $\mathcal{V}$, $\mathrm{view}_{\mathcal{P}}(\langle \mathcal{P}, \mathcal{V} \rangle)$ *und* $\mathrm{view}_{\mathcal{V}}(\langle \mathcal{P}, \mathcal{V} \rangle)$*, fehlen die internen Münzwürfe und die individuelle Eingabe der anderen Seite.*

Die Views beider Seiten beim Identifikationsschema $\langle \mathcal{P}, \mathcal{V} \rangle_{\mathsf{RSA}}$ aus Abbildung 1.1 lauten:

$$\mathrm{view}_{\mathcal{P}}(\langle \mathcal{P}, \mathcal{V} \rangle_{\mathsf{RSA}}) := (\langle N, e, g, X \rangle, \langle x, r \rangle, \langle y, s \rangle, \langle Y, c, W, z \rangle)$$
$$\mathrm{view}_{\mathcal{V}}(\langle \mathcal{P}, \mathcal{V} \rangle_{\mathsf{RSA}}) := (\langle N, e, g, X \rangle, \langle Y, c, W, z \rangle).$$

Die Witness-Indistinguishability eines Proof-Of-Knowledge respektive eines Identifikationsprotokolls[2] besagt, dass ein Verifier $\tilde{\mathcal{V}}$ nicht unterscheiden kann, welchen Zeugen bzw. Secret-Key der Prover $\mathcal{P}$ verwendet:

Definition 1.3.5 (Witness-Indistinguishability). *Ein Identifikationsschema* $\langle \mathcal{P}, \mathcal{V} \rangle_{\mathsf{Id}}$ *oder ein Proof-Of-Knowledge* $\langle \mathcal{P}, \mathcal{V} \rangle_{\mathsf{POK}}$ *für die Relation* R *heißt witness-indistinguishable, wenn für alle* $\tilde{\mathcal{V}}$ *und alle Zeugen* $z' \in \mathrm{witness}_{\mathsf{R}}(y)$ *die Verteilungen von*

- $\left(z, z', \mathrm{view}_{\tilde{\mathcal{V}}}(\langle \mathcal{P}(y, z), \tilde{\mathcal{V}}(y) \rangle_{\mathsf{POK}})\right)$ *und*
- $\left(z, z', \mathrm{view}_{\tilde{\mathcal{V}}}(\langle \mathcal{P}(y, z'), \tilde{\mathcal{V}}(y) \rangle_{\mathsf{POK}})\right)$

[2] Ein Identifikationsprotokoll muß kein Proof-Of-Knowledge sein, dem in Abschnitt 2.2 konstruierten System fehlt der Knowledge-Extraktor. Der Extraktor ist keine Voraussetzung für Witness-Indistinguishability, daher verwendet man die Bezeichnung auch für Identifikationsschema mit der Relation zwischen Secret- und Public-Key.

ununterscheidbar sind. Je nach Grad der Ununterscheidbarkeit spricht man von perfekter, statistischer oder Polynomialzeit-Witness-Indistinguishability.

Für das Protokoll $\langle \mathcal{P}, \mathcal{V} \rangle_{\mathsf{RSA}}$ gilt die perfekte Witness-Indistinguishability, d.h. die Verteilung von $\mathrm{view}_{\tilde{\mathcal{V}}}(\langle \mathcal{P}, \mathcal{V} \rangle_{\mathsf{RSA}})$ ist unabhängig von der gewählten Repräsentation des Public-Keys X. Man nennt diese Eigenschaft auch *Zeugen-Unabhängigkeit* (Witness-Independence) [Go98].

Lemma 1.3.6 (Okamoto 1992). *Das Identifikationsschema (der Proof-Of-Knowledge) $\langle \mathcal{P}, \mathcal{V} \rangle_{\mathsf{RSA}}$ aus Abbildung 1.1 erfüllt die perfekte Witness-Indistinguishability.*

Beweis. Der Prover $\mathcal{P} = \mathcal{P}(X, x, r)$ verfüge über die Repräsentation (x, r) und der Prover $\mathcal{P}' = \mathcal{P}(X, x', r')$ über die Darstellung (x', r') von X. Wir zeigen, die Kommunikation (Y, c, W, z) entspricht mit gleicher Wahrscheinlichkeit der Protokollausführung von $\mathcal{P}$ und $\tilde{\mathcal{V}}(X)$ als auch der von $\mathcal{P}'$ und $\tilde{\mathcal{V}}(X)$. Setze $\Delta x := x' - x$ und $\Delta r := r(r')^{-1} \bmod N$. Wegen $r^e \equiv X g^{-x} \pmod N$ gilt $(\Delta r)^e \equiv g^{\Delta x} \pmod N$. $\mathcal{P}$ wähle im ersten Schritt $y \in_{\mathsf{R}} \mathbb{Z}_e$ und $s \in_{\mathsf{R}} \mathbb{Z}_N^*$. $\mathcal{P}'$ votiert mit gleicher Wahrscheinlichkeit für

$$y' := y - c \cdot \Delta x \bmod e$$
$$s' := s \cdot \Delta r^c \bmod N.$$

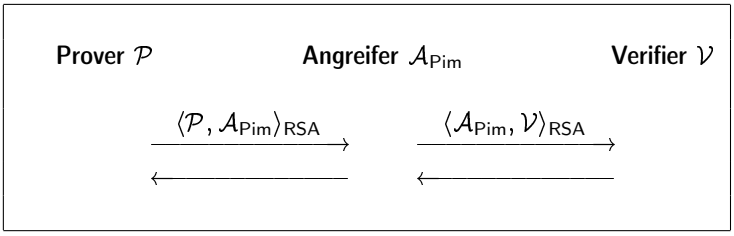
Aufgrund $Y' = Y$ stellt bei beiden Ausführungen $\tilde{\mathcal{V}}$ die Challenge c mit gleicher Wahrscheinlichkeit. Die Antworten (z, W) respektive (z', W') sind durch den Secret-Key, die hinterlegten Werte (y, s) bzw. (y', s') und die Challenge c vorgegeben. Einsetzen der Daten ergibt $(z, W) = (z', W')$. Die Wahrscheinlichkeit, dass bei einer Protokollausführung die Kommunikation (Y, c, W, z) ist, stimmt bei beiden Provern $\mathcal{P}$ und $\mathcal{P}'$ überein. □

Wir erweitern den Sicherheitsbeweis (Korollar 1.3.3, Seite 16) von Adhoc- auf aktive Angreifer. Wie geschildert, simuliere mit Hilfe der gewählten Repräsentation (x_1, r_1) von X zuvor die Protokollausführungen mit dem aktiven Angreifer als Verifier. Dank perfekter witness-indistinguishabillty, siehe Lemma 1.3.6, liefert der Knowledge-Extraktors mit Wahrscheinlichkeit $1 - \frac{1}{e}$ eine weitere Repräsentation von X. Diese Aussage gilt auch, wenn die vorhergehenden Protokollausführungen mit dem Prover $\mathcal{P}$ verschachtelt sind (*Interleaving-Attack*), man spricht dann von *parallelen Angriffen*:

Satz 1.3.7 (Okamoto 1992). *Unter der RSA-Annahme ist das Identifikationsschema $\langle \mathcal{P}, \mathcal{V} \rangle_{\mathsf{RSA}}$ aus Abbildung 1.1 sicher gegen aktive und parallele Angriffe, denn besteht $\mathcal{A}$ das Protokoll mit Wahrscheinlichkeit π, so kann in erwarteter Laufzeit $2 \cdot |\mathcal{A}|$ mit Wahrscheinlichkeit mindestens $(1 - \frac{1}{e})(\pi - \frac{1}{e})$ die e-te Wurzel von g berechnen.*

Der Begriff des parallelen Angriffs im Bezug auf Identifikationsschemas ist nicht kanonisch. Bei FEIGE, FIAT und SHAMIR [FFS88] umfaßt ein paralleler Angriff auch die Kommunikation des Angreifers $\mathcal{A}$ mit mehreren Provern $\mathcal{P}_1, \mathcal{P}_2, \ldots, \mathcal{P}_k$, und im Anschluß versucht der Angreifer sich als einer dieser Prover auszugeben. Diese Attacke kann man auf obigen Sicherheitsbeweis zurückführen. Um das RSA-Repräsentationsproblem zu $(N, g, e) \leftarrow \mathsf{RSAIndex}(1^n)$ zu lösen, wähle zufälliges $m \in_R [1, k]$ und weise $\mathcal{P}_m$ jene Daten als Public-Parameter zu. Aufgrund der Laufzeitschranke von $\mathcal{A}$ ist k ein im Sicherheitsparameter polynomieller Wert und mit nicht vernachlässigbarer Wahrscheinlichkeit $\frac{1}{k}$ versucht der Angreifer, sich als $\mathcal{P}_m$ auszugeben.

Abbildung 1.4: Person-In-The-Middle-Angriff



Wie andere Identifikationsschemas ist das Verfahren $\langle \mathcal{P}, \mathcal{V} \rangle_{\mathsf{RSA}}$ nicht sicher gegen Angriffe des Typs *Person-In-The-Middle* (PIM). Der Angreifer $\mathcal{A}_{\mathsf{Pim}}$ übernimmt bei dieser, auch als *Mafia-Attacke* bekannten Angriffsform die Rolle eines Intermediärs in der Kommunikation zwischen Verifier $\mathcal{V}$ und Prover $\mathcal{P}$ ein. Mit Hilfe der Antworten des Provers $\mathcal{P}$ versucht $\mathcal{A}_{\mathsf{Pim}}$, sich gegenüber $\mathcal{V}$ als $\mathcal{P}$ auszugeben. Der Angreifer $\mathcal{A}_{\mathsf{Pim}}$ stimuliert eine Identifikation mit $\mathcal{P}$ und reicht dessen Wert Y an $\mathcal{V}$ weiter. Seine Challenge c übergibt er seinerseits $\mathcal{P}$, um mit dessen Antwort (z, W) die Identifikation als $\mathcal{P}$ gegenüber $\mathcal{V}$ zu bestehen. Das grundsätzliche Sicherheitsproblem bei PIM-Angriffen ist die fehlende Möglichkeit, die Übernahme anonymer, digitaler Daten zu verhindern. CRAMER und DAMGARD [CrDa97] haben vorgeschlagen, den Verifier $\mathcal{V}$ ebenfalls mit einem Schlüsselpaar auszustatten und so die Identifikation an den Verifier zu binden ($\mathcal{P}$ beweist, er kennt den Secret-Key von $\mathcal{P}$ oder $\mathcal{V}$), mit der Absicht, diese Protokollausführung nicht-übertragbar zu machen.

1.4 Unterschriftenschema

Digitale Unterschriften, eine der zentralen Anwendungen moderner Public-Key-Kryptographie, dienen der öffentlichen Authentifizierung einer Nachricht m. Der Signer $\mathcal{S}$ erzeugt mit Hilfe seines Secret-Keys eine Signatur $\sigma(m)$ zur Nachricht m und jeder Anwender (User) $\mathcal{U}$ kann eigenständig anhand des öffentlichen Schlüssels von $\mathcal{S}$ nachvollziehen, dass dieser die Nachricht m in Form von $\sigma(m)$ digital unterzeichnet hat. Ein einfaches Beispiel ist die RSA-Unterschrift $\sigma(m) = m^{\frac{1}{e}} \bmod N$ [RSA78]. Zur Verifikation vergleiche die e-te Potenz der Signatur mit der Nachricht:

$$\sigma(m)^e \stackrel{!}{\equiv} m \pmod{N}.$$

Zum Unterschreiben längerer Nachrichten verwendet man im allgemeinen das *Hash-&-Sign-Prinzip*, bei dem statt einer Nachricht m nur ihr signifikant kürzerer Hashwert $h(m)$ unterzeichnet wird. Die Urbildmenge der *Hashfunktion* $h(\cdot)$ ist deutlich größer als der Bildbereich, so dass zwar *Kollisionen*, also Paare (x_1, x_2) mit $x_1 \neq x_2$ und $h(x_1) = h(x_2)$, existieren, dennoch soll aus Sicherheitsgründen, weil alle Nachrichten mit identischem Hashwert die gleiche Signatur haben, jeder Polynomialzeit-Algorithmus eine solche Kollision bestensfalls mit vernachlässigbarer Wahrscheinlichkeit finden. Man spricht dann von einer *kollisions-resistenten* Hashfunktion. Beispielsweise stellt unter der RSA-Annahme

$$h_{(N,g,e)}(m, r) := g^m r^e \bmod N$$

mit $(N, e, g) \leftarrow \mathsf{RSAIndex}(1^n)$ eine kollisions-resistente Hashfunktion $h : \mathbb{Z}_N^* \times \mathbb{Z}_e \to \mathbb{Z}_N^*$ dar, denn eine Kollision entspricht einer Lösung des RSA-Repräsentationsproblems. In der Praxis sind solche auf Zahlentheorie basierenden Hashfunktionen allerdings zu komplex bzw. die Implementierung zu zeitaufwendig und man greift auf Funktionen wie MD5 oder SHA zurück [MOV97].

Fiat und Shamir [FS86] haben bei der Vorstellung des ersten Identifikationsschemas mit Three-Move-Struktur vorgeschlagen, die Challenge durch einen Funktionswert zu ersetzen, um auf diese Weise die Interaktion zu eliminieren und ein Unterschriftenschema zu erhalten. Betrachten wir diese generische Transformation anhand Okamotos Identifikationsprotokoll $\langle \mathcal{P}, \mathcal{V} \rangle_{\mathsf{RSA}}$ aus Abbildung 1.1 auf Seite 11:

- Der Prover übernimmt die Rolle des Unterschreibenden (Signers) $\mathcal{S}$,
- die Kommunikation $\sigma(m) := (Y, W, z)$ bezüglich Challenge $c := c(Y, m)$ ist die Unterschrift zu m und

- zum Überprüfen einer Signatur $\sigma(m)$ verifiziere, dass dies in der Tat eine gültige Kommunikation zur Challenge $c(Y, m)$ ist.

Wir haben gesehen, dass ein Betrüger bei Vorabkenntnis der Challenge c das Protokoll über eine geeignete Wahl von Y auch ohne Wissen einer RSA-Repräsentation bestehen kann. Aus diesem Grund bestimmt man die Challenge in Abhängigkeit sowohl von Y als auch der Nachricht m über eine Hashfunktion $c := H(Y, m)$. Während für die beim Sign-&-Hash-Ansatz verwendete Hashfunktion $h(\cdot)$ Kollisions-Resistenz ausreichte, ist die Forderung nach dieser Eigenschaft bezüglich $H(\cdot)$ ungenügend. Die Kollisions-Resistenz schließt nicht aus, dass ein Angreifer $\mathcal{A}$ effizient zu (Y, m) eine Nachricht m^* mit $h(Y, m) \equiv h(Y, m^*) + 1 \pmod{e}$ findet. Dann kann $\mathcal{A}$ die gültige Unterschrift zu m in eine zur Nachricht m^* transformieren, weil u.a. bei OKAMOTOS Identifikation die Werte (W, z) an die um eins erhöhte Challenge ohne weiteres anzupassen sind. Für die Sicherheitsanalyse der Signaturschema wird unterstellt, die Funktion H verhalte sich wie eine echte Zufallsfunktion. Modelliert wird diese Prämisse durch das *Random-Oracle-Modell* [BR93], bei dem die Funktion H als Blackbox gegeben ist und Auswertungen über Fragen an ein Orakel (*Queries*) erfolgen. Insbesondere sind für je zwei verschiedene Commitments Y_1 und Y_2 die Zufallsvariablen $H(Y_1, m)$ und $H(Y_2, m)$ unabhängig und uniform verteilt, was der Wahl der Challenges im Protokoll bei unabhängigen Ausführungen entspricht.

Betrachten wir das aus OKAMOTOS Identifikation $\langle \mathcal{P}, \mathcal{V} \rangle_{\mathsf{RSA}}$ abgeleitete Unterschriftenschema. Um eine Nachricht m zu unterzeichnen, wählt der Signer S zufällige $y \in_{\mathsf{R}} \mathbb{Z}_e$, $s \in_{\mathsf{R}} \mathbb{Z}_N^*$ und bildet die Unterschrift mit Hilfe des geheimen Schlüssels (x, r) analog zum Identifikationsprotokoll

$$\sigma(m) := \big(\underbrace{g^y s^e \bmod N}_{=Y}, \underbrace{y + cx \bmod e}_{=z}, \underbrace{s r^c g^{\lfloor (y+cx)/e \rfloor} \bmod N}_{=W} \big)$$

mit $c := H(Y, m)$. Zur Prüfung einer Unterschrift $\sigma(m) = (Y, z, W)$ verifiziert man anhand des Public-Keys X, ob

$$Y X^{H(Y,m)} \stackrel{!}{=} W^e g^z \pmod{N}.$$

Für die Sicherheit des Identifikationsschemas haben wir die Erfolgswahrscheinlichkeit eines (aktiven) Angreifers $\mathcal{A}$, nach einer Lernphase eine Protokollausführung als Prover zu bestehen, betrachtet. Im Fall eines Unterschriftenschemas erhält der Angreifer $\mathcal{A}$ in der Lernphase die Unterschriften zu beliebigen, mithin adaptiv gewählten Nachrichten, modelliert mit dem Zugang zu einem Signatur-Orakel. Diese Form des aktiven Angriffs auf ein Unterschriftenschema nennt man *Adaptive-Chosen-Message* [GMRi88]. Ziel von

$\mathcal{A}$ ist, eine Unterschrift $\sigma(m)$ zu einer Nachricht m seiner Wahl zu berechnen. Um diese Form der Fälschung, eine sogenannte *Existentical-Forgery*, nicht zu trivialisieren, darf der Angreifer $\mathcal{A}$ keine Unterschrift zu m vom Signatur-Orakel zuvor erfragen. Ist die Erfolgswahrscheinlichkeit des Angreifers vernachlässigbar klein (bezüglich Sicherheitsparameter), nennt man das Schema sicher gegen Existential-Forgery bei Adaptive-Chosen-Message-Angriffen:

Satz 1.4.1 (Okamoto 1992, Pointcheval und Stern 1996). *Das aus dem Identifikationsschema $\langle \mathcal{P}, \mathcal{V} \rangle_{\mathsf{RSA}}$ abgeleitete Unterschriftenschema ist unter der RSA-Annahme im Random-Oracle-Modell bei Adaptive-Chosen-Message-Angriffen sicher gegen Existential-Forgery.*

Wir skizzieren die Grundidee des Beweises [PSt96a, PSt00]. Angenommen, ein Angreifer $\mathcal{A}$ erzeuge eine Unterschrift $\sigma(m)$ zu einer Nachricht m. Wir setzen o.B.d.A. voraus, dass $\mathcal{A}$ das Hash-Orakel nicht mehrfach nach dem Hashwert eines Argumentes fragt. Wie im Beweis der Sicherheit des Identifikationsschemas gegen aktive Angreifer (Satz 1.3.7 auf Seite 18) simulieren wir das Signatur-Orakel mit der von uns gewählten Repräsentation. Beim Zurücksetzen gehen wir an die Stelle, an welcher der Angreifer $\mathcal{A}$ das Hash-Orakel nach $H(Y, m) \in \mathbb{Z}_e$ fragt und ersetzen die Hashfunktion H durch eine, die mit H auf den bisherigen ausgewerteten Stellen übereinstimmt. Dies ist im Random-Oracle-Modell möglich, weil $\mathcal{A}$ nur mittels Fragen ans Hash-Orakel Informationen über Hashwerte erhält und die Verteilungen der „nachfolgenden“ Hashwerte unabhängig von den bisherigen Auswertungen sind. Die beiden Unterschriften zu m entsprechen zwei Protokollausführungen mit gleichem Y-Wert und verschiedenen Hashwerten (Challenges), so dass wir das RSA-Repräsentationsproblem und im Ergebnis RSA lösen. Die Reduktion ist allerdings nicht sicherheitserhaltend. Angenommen, $\mathcal{A}$ gelinge mit Q Hash-Orakel-Fragen in einem Adaptive-Chosen-Message-Angriff eine Existential-Forgery mit Wahrscheinlichkeit $\pi > \frac{7Q}{e}$. Dann erhalten wir in maximal $2^{16,4}Q \cdot |\mathcal{A}|$ Schritten mit Wahrscheinlichkeit π zwei Unterschriften (Y, c_1, z_1, W_1) und (Y, c_2, z_2, W_2) mit $c_1 \neq c_2$, um das RSA-Repräsentationsproblem zu lösen [PSt00, Theorem 10].

Eine wichtige Variante Digitaler Signaturen sind die bereits 1982 von Chaum [Ch82] eingeführten *blinden Unterschriften*. Umgangssprachlich ausgedrückt unterschreibt $\mathcal{S}$ einem Anwender $\mathcal{U}$ einen Wert ohne die eigentliche Nachricht zu kennen. Chaums Idee war, dass in einem Ecash-System von der Bank signierte Nachrichten digitalen Münzen entsprechen und durch die blinde Unterschrift die Anonymität der Kunden beim späteren Bezahlen im Geschäft gewährleistet ist. Die Blindheit verwehrt der Bank die Möglichkeit festzustellen, wem sie die vom Ladenbesitzer eingereichte Münze zuvor

ausstellte, um so beispielsweise ein Konsumprofil der Bankkunden zu generieren. Der Weg der digitalen Münze ist nicht zu verfolgen, man spricht von *Untraceability*.

Wir beschreiben eine Umsetzung des Konzepts der blinden Signaturen zunächst anhand der RSA-Unterschrift $\sigma(m) := m^{\frac{1}{e}} \bmod N$ zu einer Nachricht $m \in \mathbb{Z}_N^*$. Für die blinde Variante wählt $\mathcal{U}$ zu m einen *Randomisierer* $\alpha \in_R \mathbb{Z}_N^*$ und bittet $\mathcal{S}$, die Nachricht $m' := m\alpha^e \bmod N$ zu unterzeichnen. Durch anschließende Multiplikation der von $\mathcal{S}$ erhaltenen Signatur $\sigma(m')$ mit α^{-1} erhält $\mathcal{U}$ die Unterschrift zu m:

$$\sigma(m')\alpha^{-1} \equiv (m\alpha^e)^{\frac{1}{e}}\alpha^{-1} \equiv m^{\frac{1}{e}} \equiv \sigma(m) \pmod N.$$

Bezeichne Σ die Transformation $\Sigma(m', \sigma(m'), \alpha) := \sigma(m')\alpha^{-1}$ der geleisteten Unterschrift $\sigma(m')$ in die zur eigentlichen Nachricht m. Das Paar $(m', \sigma(m'))$ ist uniform verteilt in der Menge aller Nachricht-Unterschrift-Paaren und unabhängig von $(m, \sigma(m))$, weshalb man von perfekter Blindheit spricht. Bei $(m', \sigma(m'))$ handelt es sich um den Signer-View des Signaturprotokolls $\langle \mathcal{S}, \mathcal{U} \rangle_{\mathsf{BISig}}(m)$, wobei m als *beabsichtigter Wert* (engl. intended value) des Protokolls bezeichnet wird. Für die Blindheit fordert man, dass für den Signer $\tilde{\mathcal{S}}$ eine Protokollausführung $\langle \tilde{\mathcal{S}}, \mathcal{U} \rangle_{\mathsf{BISig}}(m)$ nicht dem Paar $(m, \sigma(m))$ bestehend aus Nachricht m und seitens ehrlichem User $\mathcal{U}$ gewonnene Unterschrift $\sigma(m) := \Sigma(\mathrm{view}_{\mathcal{U}}(\langle \tilde{\mathcal{S}}, \mathcal{U} \rangle_{\mathsf{BISig}}(m)))$ zuzuordnen ist [JLO97]. Für die Definition der Blindheit ist der Signer $\tilde{\mathcal{S}}$ zwar nicht an die Protokollspezifikation gebunden, muß aber eine vom ehrlichen User $\mathcal{U}$ akzeptierte Antwort geben, die dieser dann in eine Unterschrift zur beabsichtigten Nachricht m transformieren kann:

Definition 1.4.2 (Blindheit von Unterschriften). *Ein Unterschriften-Schema $\langle \mathcal{S}, \mathcal{U} \rangle_{\mathsf{BISig}}$ heißt blind, wenn für alle $\tilde{\mathcal{S}}$ und alle Nachrichten $m_1 := m_1(\sigma, \mathsf{pk})$, $m_2 := m_2(\sigma, \mathsf{pk})$ die Verteilungen*

- $\begin{pmatrix} m_1, & \mathrm{view}_{\tilde{\mathcal{S}}}(\langle \tilde{\mathcal{S}}, \mathcal{U} \rangle_{\mathsf{BISig}}(m_1)), & \underbrace{\Sigma(\mathrm{view}_{\mathcal{U}}(\langle \tilde{\mathcal{S}}, \mathcal{U} \rangle_{\mathsf{BISig}}(m_1)))}_{=\sigma(m_1)} \\ m_2, & \mathrm{view}_{\tilde{\mathcal{S}}}(\langle \tilde{\mathcal{S}}, \mathcal{U} \rangle_{\mathsf{BISig}}(m_2)), & \end{pmatrix}$ *und*
- $\begin{pmatrix} m_1, & \mathrm{view}_{\tilde{\mathcal{S}}}(\langle \tilde{\mathcal{S}}, \mathcal{U} \rangle_{\mathsf{BISig}}(m_1)), & \underbrace{\Sigma(\mathrm{view}_{\mathcal{U}}(\langle \tilde{\mathcal{S}}, \mathcal{U} \rangle_{\mathsf{BISig}}(m_2)))}_{=\sigma(m_2)} \\ m_2, & \mathrm{view}_{\tilde{\mathcal{S}}}(\langle \tilde{\mathcal{S}}, \mathcal{U} \rangle_{\mathsf{BISig}}(m_2)), & \end{pmatrix}$

ununterscheidbar sind. Je nach Grad der Ununterscheidbarkeit spricht man von perfekter, statistischer oder Polynomialzeit-Blindheit.

Man beachte, dass im Falle eines interaktiven Signaturprotokolls die beide Ausführungen $\langle \tilde{\mathcal{S}}, \mathcal{U} \rangle_{\mathsf{BISig}}(m_1)$, $\langle \tilde{\mathcal{S}}, \mathcal{U} \rangle_{\mathsf{BISig}}(m_2)$ eventuell verschachtelt sind. Die Blindheit ist alternativ durch folgendes Experiment zu charakterisieren. Der Signer $\tilde{\mathcal{S}}$ erhält die zwei Nachrichten m_1 und m_2, die er

dann in zufälliger Reihenfolge zum blinden Unterschreiben vorgelegt bekommt, um anschließend zu entscheiden, welche der beiden Nachrichten er als erstes signiert hat, d.h. welche der beide Ausführungen $\langle \widetilde{\mathcal{S}}, \mathcal{U} \rangle_{\mathsf{BISig}}(m_1)$, $\langle \widetilde{\mathcal{S}}, \mathcal{U} \rangle_{\mathsf{BISig}}(m_2)$ als erste vom User $\mathcal{U}$ initiiert wurde. Für ein blindes Unterschriften-Schema ist der Vorteil gegenüber Raten vernachlässigbar. Bei perfekter Blindheit sinkt der Vorteil sogar auf Null, denn dann ist der View $\mathrm{view}_{\widetilde{\mathcal{S}}}(\langle \widetilde{\mathcal{S}}, \mathcal{U} \rangle_{\mathsf{BISig}}(m))$ unabhängig von Nachricht und Signatur $(m, \sigma(m))$.

Abbildung 1.5: Blinde Unterschrift $\langle \mathcal{S}, \mathcal{U} \rangle_{\mathsf{RSABISig}}$ auf RSA-Repr.

Signer $\mathcal{S}$	(N, τ, t, g)	**User** $\mathcal{U}$
Repräsentation x, r	$X = g^x r^e \bmod N$	
Wähle $y \in_R \mathbb{Z}_e$ Wähle $s \in_R \mathbb{Z}_N^*$ $Y := g^y s^e \bmod N$	$\xrightarrow{Y}$	Wähle $\alpha, \gamma \in_R \mathbb{Z}_e$ Wähle $\beta \in_R \mathbb{Z}_N^*$ $Y' := Y g^\alpha \beta^e X^\gamma \ (N)$ $c := H(Y', m) \in \mathbb{Z}_e$ $c' := c + \gamma \bmod e$ Wähle c'', so dass $c + \gamma = c' + e c''$
	$\xleftarrow{c'}$	
$z' := y + c' x \bmod e$ $W' := s r^{c'} g^{\lfloor (y + c' x)/e \rfloor} \ (N)$	$\xrightarrow{W', z'}$	$Y X^{c'} \stackrel{!}{\equiv} (W')^e g^{z'} \ (N)$ $z := z' - \alpha \bmod e$ Wähle z'', so dass $z' + \alpha = z - e z''$ $W := W' \beta g^{-z''} X^{c''} \ (N)$
	Modulo N gilt $Y' X^c \equiv W^e g^z$	
		$\sigma(m) := (Y', W, z)$

Aus Okamotos Identifikation $\langle \mathcal{P}, \mathcal{V} \rangle_{\mathsf{RSA}}$ bzw. dem Signaturverfahren ist ebenfalls ein Blinde-Unterschriften-Schema abzuleiten [PSt96b, PSt00]. Abbildung 1.5 zeigt das Signaturprotokoll $\langle \mathcal{S}, \mathcal{U} \rangle_{\mathsf{RSABISig}}$. Mit $\sigma(m) := (Y', W, z)$

verfügt $\mathcal{U}$ über eine Unterschrift zu m, denn

$$\begin{aligned} W^e g^z &\equiv (W'\beta g^{-z''} X^{c''})^e g^{z'} g^{z-z'} \\ &\equiv Y X^{c'} (\beta g^{-z''} X^{c''})^e g^{z-z'} \\ &\equiv Y X^{c'+ec''} \beta^e g^{z-ez''-z'} \\ &\equiv Y X^{c+\gamma} \beta^e g^{\alpha} \\ &\equiv Y' X^c \pmod N. \end{aligned} \tag{1.5}$$

Wir rechnen nach, dass (Y, c', W', z') dank der Randomisierer (α, β, γ) unabhängig von (Y', c, W, z) verteilt ist und als Konsequenz das Signaturschema perfekte Blindheit bietet:

Lemma 1.4.3. *Das Unterschriften-Schema $\langle \mathcal{S}, \mathcal{U} \rangle_{\mathsf{RSABlSig}}$ aus Abbildung 1.5 erfüllt die perfekte Blindheit.*

Beweis. Gegeben seien zu den beiden Nachrichten m_1, m_2 zwei blinde Unterschriften $\sigma(m_1) = (Y_1', W_1, z_1)$ und $\sigma(m_2) = (Y_2', W_2, z_2)$ mit $c_1 := H(Y_1', m_1)$ sowie $c_2 := H(Y_2', m_2)$. Wir zeigen, die vorliegende Kommunikation (Y_1, c_1', W_1', z_1') beim Unterschreiben von m_1 wäre mit gleicher Wahrscheinlichkeit beim Signieren von m_2 aufgetreten. Seien $(\alpha_1, \beta_1, \gamma_1)$ die beim Unterzeichnen von m_1 verwandten Randomisierer. Wir benennen passende Werte $(\alpha_2, \beta_2, \gamma_2)$, für welche die Kommunikation (Y_2, c_2', W_2', z_2') im Protokoll $\langle \tilde{\mathcal{S}}, \mathcal{U} \rangle_{\mathsf{RSABlSig}}(m_2)$ mit (Y_1, c_1', W_1', z_1') übereinstimmt, wenn $\tilde{\mathcal{S}}$ in beiden Ausführungen gleiche Zufallsbits verwendet.

Die Wahl von Y_1 im ersten Schritt hängt weder von den von $\mathcal{U}$ später gewählten Werten noch von der Nachricht ab, so dass im Ergebnis $\tilde{\mathcal{S}}$ mit gleicher Wahrscheinlichkeit $Y_2 = Y_1$ vorgibt. Sei $\Delta c := c_1' - c_2$ und $\Delta z = z_2 - z_1'$. Mit der gleichen Wahrscheinlichkeit wie der User beim Unterschreiben von m_1 die Randomisierer $(\alpha_1, \beta_1, \gamma_1)$ wählt, votiert $\mathcal{U}$ für

$$\begin{aligned} \alpha_2 &:= (\Delta z \bmod e) \equiv \alpha_1 + z_1 - z_2 \pmod e \\ \beta_2 &:= W_2 \cdot (W_1')^{-1} \cdot g^{\lfloor \Delta z / e \rfloor} \cdot X^{\lfloor \Delta c / e \rfloor} \pmod N \\ \gamma_2 &:= (\Delta c \bmod e) \equiv \gamma_1 + c_1 - c_2 \pmod e, \end{aligned}$$

denn mit $\alpha_1, \beta_1, \gamma_1$ bzw. α_1, W_1', γ_1 sind auch $\alpha_2, \beta_2, \gamma_2$ uniform und unabhängig verteilt. Für die Randomisierer $(\alpha_2, \beta_2, \gamma_2)$ gilt

$$c_2' \equiv c_2 + \gamma_2 \equiv c_2 + (\gamma_1 + c_1 - c_2) \equiv c_1 + \gamma_1 \equiv c_1' \pmod e,$$

und aus $Y_1 = Y_2$ und den beiden Kontrollgleichungen folgt

$$\begin{aligned} (W_1')^e &\equiv g^{-z_1'} Y_1 X^{c_1'} \pmod N \\ W_2^e &\equiv Y_2' X^{c_2} g^{-z_2} \pmod N, \end{aligned}$$

dass die gewählten Randomisierer in der Tat zum Y_2'-Wert der Unterschrift $\sigma(m_2)$ passen:

$$\begin{aligned} Y_2 g^{\alpha_2} \beta_2^e X^{\gamma_2} &\equiv Y_1 \cdot X^{e\cdot\lfloor \Delta c/e\rfloor+\gamma_2} \cdot g^{e\cdot\lfloor \Delta z/e\rfloor+\alpha_2} \cdot W_2^e (W_1')^{-e} \\ &\equiv Y_1 \cdot X^{\Delta c} \cdot g^{\Delta z} \cdot W_2^e (W_1')^{-e} \\ &\equiv Y_1 \cdot X^{c_1'-c_2} \cdot g^{z_2-z_1'} \cdot W_2^e (W_1')^{-e} \\ &\equiv Y_2' \pmod N. \end{aligned}$$

Die Verteilung der Antwort (W', z') des Signers $\widetilde{S}$ hängt neben den internen Münzwürfen von $\widetilde{S}$ nur von Y und c' ab, wonach der Unterschreibende mit gleicher Wahrscheinlichkeit $(W_2', z_2') = (W_1', z_1')$ vorgibt. Nach Wahl von α_2 folgt aus $z_2' = z_1'$ und $z_1 + \alpha_1 \equiv z_1' \pmod e$, dass $z_2' - \alpha_2$ die Vorgabe z_2 erfüllt:

$$z_2' - \alpha_2 \equiv z_1' - (\alpha_1 + z_1 - z_2) \equiv z_1' - (z_1' - z_2) \equiv z_2 \pmod e.$$

Der verbleibende Nachweis von $W_2 \equiv W_2' \beta_2 g^{-z_2''} X^{c_2''} \pmod N$ anhand der beiden Kontrollgleichungen $Y_2 X^{c_2'} \equiv (W_2')^e g^{z_2'}$ und $X^{c_2} \equiv W_2^e g^{z_2}$ modulo N ist elementar, es handelt sich um eine einfache, algebraische Umformung des Typs (1.5). □

Die Definition einer Existential-Forgery ist nicht unmittelbar auf blinde Unterschriften zu übertragen, denn die Restriktion, dass der Angreifer $\mathcal{A}$ nicht zuvor das Orakel nach der betreffenden Signatur befragt haben darf, widerspricht der Eigenschaft blinder Unterschriften. Pointcheval und Stern [PSt96b, PSt00] haben den Begriff der sogenannten $(\ell, \ell+1)$-*Forgery* oder kurz *One-More-Forgery* eingeführt: Der Angreifer $\mathcal{A}$ soll nach ℓ Interaktionen mit dem Signatur-Orakel am Ende über $\ell+1$ Unterschriften verfügen. Im Beispiel digitaler Münzen ist das Ziel des Angriffs der Besitz von mehr Geld als zuvor von der Bank abgehoben. Weil das digitale Unterschreiben durch ein interaktives Protokoll bewerkstelligt wird, sind für die Sicherheit blinder Signaturen ebenfalls parallele Angriffe relevant. Für das Blinde-Unterschriften-Schema $\langle S, \mathcal{U}\rangle_{\mathsf{RSABlSig}}$ gilt [PSt00, Theorem 26]:

Satz 1.4.4 (Pointcheval und Stern 1996). *Unter der RSA-Annahme ist das Blinde-Unterschriften-Schema $\langle S, \mathcal{U}\rangle_{\mathsf{RSABlSig}}$ aus Abbildung 1.5 im Random-Oracle-Modell bis zu poly-logarithmisch vielen Interaktionen ℓ mit dem Signatur-Orakel sicher gegen (parallele) One-More-Forgery.*

Im Hinblick auf die Sicherheit ist die Güte der Reduktion ernüchternd: Gelingt $\mathcal{A}$ mit Q Hash-Orakel-Fragen eine One-More-Forgery mit Wahrscheinlichkeit $\pi \geq 4Q^{\ell+1}e^{-1}$, lösen wir in $2^{20}(\ell+1)^2 e^{-2} Q|\mathcal{A}|$ Schritten das

RSA-Repräsentationsproblem mit Wahrscheinlichkeit π.[3] Insbesondere die Restriktion von lediglich sub-polynomiell vielen Interaktionen mit dem Signatur-Orakel entspricht nicht dem Sicherheitsgedanken der Kryptographie einer polynomiellen Schranke. Ein theoretisches Resultat von JUELS, LUBY und OSTROVSKY [JLO97] belegt, dass unter kryptographischen Standardannahmen prinzipiell Blinde-Unterschriften-Schemata, welche auch bei polynomiell vielen Interaktionen beweisbar sicher sind, existieren.

1.5 Commitment-Schemata

Commitment-Schemata, sie dienen zur sicheren Hinterlegung von geheimen Daten, sind ein Standardbausteine der Kryptographie [Go98]. Ein solches Verfahren zur Hinterlegung von m setzt sich aus zwei interaktiven Protokollen, auch Phasen genannt, zusammen:

Hinterlegungs- oder Commitment-Phase: Durch die gesandten Daten im Protokoll $\langle \mathcal{S}, \mathcal{R} \rangle_{\mathsf{com}}(m)$ bindet sich der Sender $\mathcal{S}$ gegenüber dem Empfänger (Receiver) $\mathcal{R}$ an die geheime Nachricht m. Die kommunizierten Daten M werden als das Commitment bezeichnet.

Aufdeckungs-, Öffnungs- oder Decommitment-Phase: Der Sender deckt im Protokoll $\langle \mathcal{S}, \mathcal{R}(M) \rangle_{\mathsf{decom}}(m)$ den verbrieften Wert m auf und beweist dem Receiver $\mathcal{R}$, dass er diese Nachricht zuvor in Form von M hinterlegt hatte.

Zur Aufdeckung übergibt der Sender $\mathcal{S}$ dem Empfänger $\mathcal{R}$ in der Regel seine Zufallsbits aus der Commitment-Phase als Nachweis, dass die offenbarte Nachricht m mit der zuvor hinterlegten übereinstimmt. Bei dieser Form ist die Decommitment-Phase nicht-interaktiv. Wird auch die Hinterlegung durch eine Nachricht des Senders $\mathcal{S}$ an den Receiver $\mathcal{R}$ bewerkstelligt, spricht man von einem *nicht-interaktiven Commitment-Schema*. Analog zur digitalen Unterschrift gibt es ein *Hash-&-Commit-Prinzip*: Statt der eigentlichen Nachricht m hinterlegt man ihren kürzeren Hashwert $h(m)$ bezüglich einer kollisions-resistenten Hashfunktion h. NAOR [N91] hat gezeigt, wie aus jedem Pseudozufallsgenerator und im Ergebnis aus jeder Oneway-Funktion [HILL99] ein einfaches, interaktives, polynomialzeit-bindendes Bit-Commitment (d.h. ein einzelnes Bit wird hinterlegt) mit statistischer Geheimhaltung zu konstruieren ist. In der Commitmentphase von NAORS Schema sendet der Receiver $\mathcal{R}$ einen zufälligen Bitstring, und

[3] Für poly-logarithmisches ℓ, RSA-Exponent e mit vernachlässigbarem Kehrwert $\frac{1}{e}$ und polynomielles Q erfüllt jeder polynomielle Bruchteil π die untere Schranke $4Q^{\ell+1}e^{-1}$ für hinreichend großes n.

erhält von $\mathcal{S}$ die Hinterlegung in Form eines Bitstrings gleicher Länge. Durch die Vorgabe des zufälligen Bitstrings durch eine Trusted-Party $\mathcal{T}$ statt $\mathcal{R}$, d.h. als Public-Parameter, erhält man ein nicht-interaktives Commitment-Schema.

Das RSA-Repräsentationsproblem ist die Grundlage eines einfachen, nicht-interaktiven Commitment-Schemas, welches die simultane Hinterlegung mehrere Bits erlaubt. Die Trusted-Party $\mathcal{T}$ stellt den Teilnehmern die öffentlichen Parameter $(N, e, g) \leftarrow \mathsf{RSAIndex}(1^n)$ zur Verfügung. Die Dritte Partei $\mathcal{T}$ sichert beiden Seiten zu, dass die Public-Parameters die Geheimhaltung als auch (unter der RSA-Annahme) die Festlegung gewährleisten. Um $m \in \mathbb{Z}_e$ zu hinterlegen, wählt der Sender $\mathcal{S}$ ein zufälliges $r \in_R \mathbb{Z}_N^*$ und schickt

$$M := \mathrm{com}(m, r) = g^m r^e \bmod N \tag{1.6}$$

an den Empfänger $\mathcal{R}$. In der Aufdeckungsphase publiziert $\mathcal{S}$ neben der Nachricht m auch den zufälligen Wert r aus der Commitment-Phase. Der Receiver $\mathcal{R}$ akzeptiert die Aufdeckung, sofern die beiden Werte zum Commitment M passen, dementsprechend (m, r) eine RSA-Darstellung von M ist. Die traditionelle Sicherheitsanalyse eines Commitment-Schemas umfaßt zwei Aspekte:

Bindung oder Eindeutigkeit: Durch sein Commitment soll der Sender $\mathcal{S}$ an sein hinterlegtes Datum derart eindeutig gebunden sein, dass $\mathcal{S}$ außer Stande ist, später die Hinterlegung erfolgreich mehrdeutig, d.h. mit verschiedenen Nachrichten, zu öffnen.

Geheimhaltung: Der Empfänger $\mathcal{R}$ soll anhand des Commitments keine Hinweise auf den hinterlegten Wert erhalten.

Betrachte das Commitment-Schema (1.6). Unter der RSA-Annahme ist für den Sender $\mathcal{S}$ seine hinterlegte Nachricht verbindlich, da er sonst das zugehörige Repräsentationsproblem lösen könnte. Man spricht in dieser Situation von *Polynomialzeit-Bindung*, denn die Festlegung gilt nur bei unterstellter polynomieller Laufzeit von $\mathcal{S}$. Weil r und somit auch r^e unabhängig von der Nachricht m uniform in $\mathbb{Z}_N^*$ verteilt sind, gilt dies auch für das Produkt $g^m r^e$ modulo N. Der Empfänger $\mathcal{R}$ erhält keine Hinweise über die hinterlegte Nachricht, selbst bei unbeschränkter Laufzeit, weil das Commitment M unabhängig von der hinterlegten Nachricht m ein uniform verteilter Wert in $\mathbb{Z}_N^*$ ist. Man spricht in diesem Fall von *perfekter Geheimhaltung*.

Definition 1.5.1 (Bindung und Geheimhaltung von Commitment-Schemata). *Seien $\langle \mathcal{S}, \mathcal{R} \rangle_{\mathsf{com}}$ und $\langle \mathcal{S}, \mathcal{R} \rangle_{\mathsf{decom}}$ die beiden Protokolle des Commitment-Schemas mit Trusted-Party $\mathcal{T}$.*

- *Das Schema heißt polynomialzeit-bindend, wenn für alle $\tilde{\mathcal{S}}$ und alle Nachrichten $m_1 := m_1(\sigma)$, $m_2 := m_2(\langle\tilde{\mathcal{S}}, \mathcal{R}\rangle_{\mathsf{com}}(m_1))) \neq m_1$ die Wahrscheinlichkeit von*

$$\langle\tilde{\mathcal{S}}, \mathcal{R}(\langle\tilde{\mathcal{S}}, \mathcal{R}\rangle_{\mathsf{com}}(m_1))\rangle_{\mathsf{decom}}(m_2) \in \mathsf{Accept},$$

  vernachlässigbar ist. Im Fall exponentiell kleiner Wahrscheinlichkeit spricht man von statistischer Bindung.

- *Das Schema heißt geheimhaltend, wenn für alle $\tilde{\mathcal{R}}$ und Nachrichten $m_1 := m_1(\sigma)$, $m_2 := m_2(\sigma)$ die Verteilungen*

  - $(m_1, m_2, \mathrm{view}_{\tilde{\mathcal{R}}}(\langle\mathcal{S}, \tilde{\mathcal{R}}\rangle_{\mathsf{com}}(m_1)))$ *und*
  - $(m_1, m_2, \mathrm{view}_{\tilde{\mathcal{R}}}(\langle\mathcal{S}, \tilde{\mathcal{R}}\rangle_{\mathsf{com}}(m_2)))$

  ununterscheidbar sind. Je nach Grad der Ununterscheidbarkeit spricht man von perfekter, statistischer oder Polynomialzeit-Geheimhaltung.

Bei einem statistisch bindenden Commitment liegt die Nachricht bereits durch das Commitment eindeutig fest, wenn im Fall der Polynomialzeit-Bindung zwar mehrere Aufdeckungen existieren (können), allerdings durch die Laufzeitbeschränkung der Sender $\mathcal{S}$ nicht in der Lage ist, eine zweite Öffnung zu generieren. Es gilt [Ok92]:

Satz 1.5.2. *Mit $(N, e, g) \leftarrow \mathsf{RSAIndex}(1^n)$ und $r \in_R \mathbb{Z}_N^*$ ist unter der RSA-Annahme*

$$\mathrm{com}(m, r) := g^m r^e \bmod N \bmod N$$

ein polynomialzeit-bindendes, nicht-interaktives Commitment-Schema mit perfekter Geheimhaltung.

Unter der Annahme, der Sender $\mathcal{S}$ kann das RSA-Problem (N, e, g) effizient nicht lösen, also eine e-te Wurzel von g mod N bestimmen, ist $\mathcal{S}$ an seinen hinterlegten Wert gebunden. Die Kenntnis von $g^{\frac{1}{e}} \bmod N$ versetzt umgekehrt in die Lage, das Commitment mit jeder beliebigen Nachricht zu öffnen. Die e-te Wurzel zu g heißt *Trapdoor*[4] bezüglich der Bindungseigenschaft, ein Hinterlegungsschema mit einer Geheimtüre nennt man *Trapdoor-Commitment.*[5] Trapdoor-Commitments allgemein und das erste explizite

[4] Bildlich gesprochen, wer nicht über diese Information verfügt, fällt beim Versuch, das Commitment mehrdeutig zu öffnen, in die Falltüre (engl. Trapdoor). In der deutschen Literatur wird die Trapdoor-Information auch als Geheimtüre bezeichnet.

[5] Es existieren ebenfalls statistisch bindende Verfahren mit Trapdoors bezüglich der Geheimhaltung, die es dem Simulator ermöglichen, den verbrieften Wert vor der Aufdeckung durch den Sender $\mathcal{S}$ bereits zu bestimmen. Diese Form der Trapdoor ist für unsere Anwendungen allerdings nicht von Bedeutung, so dass im nachfolgenden Text Geheimtüren sich stets auf die Bindung beziehen.

Verfahren gehen auf BRASSARD, CHAUM und CRÉPEAU [BCC88] zurück. BEAVER [Be96] hat in einem generellen Rahmen den Begriff *Equivocable-Commitment* (engl. zweideutige Commitments) für Hinterlegungsschemata, welche bei einer Simulation mehrdeutige Aufdeckung ermöglichen, geprägt. DI CRESCENZO, ISHAI und OSTROVSKY [DIO98] haben die Idee der Equivocable-Commitments für ihr nicht-interaktives, non-malleable Hinterlegung-Schema formalisiert und für eine Konstruktion basierend auf einem Trapdoor-Commitment, genauer NOARS Schema [N91], verwandt.

Für ein Equivocable-Commitment basierend auf der RSA-Repräsentation setzen wir voraus, dass eine Trusted-Party $\mathcal{T}$ die Werte $(N, e, g) \leftarrow \mathsf{RSAIndex}(1^n)$ vorgibt und $g \in_R \mathbb{Z}_N^*$ unabhängig von N, e verteilt ist. In einer realen Ausführung kann der Sender $\mathcal{S}$ unter der RSA-Annahme seine Hinterlegung nicht mehrdeutig öffnen. Bei einer Simulation haben wir jedoch volle Kontrolle über die Trusted-Party $\mathcal{T}$. Der Simulator wählt $(N, e, g) \leftarrow \mathsf{RSAIndex}(1^n)$, ersetzt aber g durch $h^e \bmod N$ für ein zufälliges $h \in_R \mathbb{Z}_N^*$. Erheben in die e-te Potenz modulo N ist eine Permutation, aufgrunddessen h^e ebenfalls uniform verteilt in $\mathbb{Z}_N^*$ ist. Die simulierte Dritte Partei gibt das Tupel (N, e, h^e) vor, dessen Verteilung mit der von $\mathsf{RSAIndex}(1^n)$ erzeugten übereinstimmt. Daher bleibt das Verhalten, so Laufzeit und die Erfolgswahrscheinlichkeit, der übrigen Teilnehmer bei der Simulation erhalten. Gleichzeitig sind wir in der Lage, das Commitment M des Senders $\mathcal{S}$ mit beliebigem m' statt m zu öffnen:

$$(h^e)^m r^e \equiv M \equiv (h^e)^{m'} (h^{m-m'} r)^e \pmod{N}.$$

In Verbindung mit der Möglichkeit, innerhalb der Simulation die Ausführung zurückzusetzen, verwenden wir in Kapitel 3.2 Trapdoor-Hinterlegungen als Bausteine für non-malleable Commitments.

Eine Anwendung von Commitment-Schemata ist das nachstehende *Coin-Flipping-Protokoll*, bei dem sich zwei Parteien $\mathcal{S}$ und $\mathcal{R}$ auf gemeinsamen Bitstring $\mathbf{c} \in \{0,1\}^n$ einigen, der, bei mindestens einer ehrlichen Seite, uniform verteilt in $\{0,1\}^n$ ist [Bl82]:

1. $\mathcal{S}$ wählt einen Zufallsstring $\mathbf{a} \in_R \{0,1\}^n$ und sendet an $\mathcal{R}$ das Commitment $\mathrm{com}(\mathbf{a})$.

2. $\mathcal{R}$ wählt $\mathbf{b} \in_R \{0,1\}^n$ und sendet $\mathcal{S}$ die Zufallsbits $\mathbf{b}$.

3. $\mathcal{S}$ öffnet seine Hinterlegung $\mathrm{com}(\mathbf{a})$ und publiziert $\mathbf{a}$.

4. Beide Parteien verwenden die XOR-Verknüpfung $\mathbf{c} := \mathbf{a} \oplus \mathbf{b}$ als Zufallsstring.

Die XOR-Verknüpfung $\mathbf{a} \oplus \mathbf{b}$ ist uniform in $\{0,1\}^n$ verteilt, sofern einer der beiden Werte $\mathbf{a}$ oder $\mathbf{b}$ zufällig und unabhängig vom anderen Vektor ist. Im Fall des RSA-Commitments, Satz 1.5.2, wählen $\mathcal{S}$ und $\mathcal{R}$ statt eines zufälligen Bitvektors $a, b \in_R \mathbb{Z}_e$ und bestimmen $c := a + b \bmod e$ als gemeinsammen, zufälligen Wert aus $\mathbb{Z}_e$. In Verbindung mit einem Trapdoor-Commitment können wir innerhalb der Simulation das Ergebnis des Coin-Flipping stets an unsere Vorgabe $\mathbf{c}$ durch Öffnen der Hinterlegung mit $\mathbf{c} \oplus \mathbf{b}$ respektive mit $c - b$ modulo e anpassen.

Kapitel 2

Identifikations- und Unterschriftenschemata

. . .in general nothing but frustration
can be expected to come from an attack
on a number of 25 or more digits.

— John Brillhart und John Selfridge (1967)

Does anybody seriously believe
that published attacks represent
the state of the art?

— Arjen. K. Lenstra und Eric R. Verheul

Wir stellen in diesem Kapitel eine Repräsentation vor, für welche das Auffinden zweier verschiedener Darstellungen eines Wertes die Faktorisierung des RSA-Moduls ermöglicht [FF02]. Analog zur RSA-Repräsentation bildet diese Darstellung die Grundlage eines Identifikations-, sowie eines Signatur- und eines Blinde-Unterschriften-Schemas.

2.1 Faktorisierungsrepräsentation

Das RSA- als auch das zugehörige Repräsentationsproblem sind höchstens so sicher wie das Faktorisierungsproblem, denn RSA ist bei Kenntnis der Primfaktorzerlegung des Moduls effizient zu lösen. Es ist eine offene Fragestellung der Kryptographie, ob die Reduktion auch in umgekehrter Richtung gilt. Wenngleich man annimmt, RSA sei nicht in Polynomialzeit zu lösen, gibt eine Arbeit von Boneh und Venkatesan [BV98] Hinweise, wonach die Berechnung einer e-ten Wurzel modulo N einfacher als die Faktorisierung des Moduls N sei.

Eng verbunden mit der Zerlegung eines RSA-Moduls N in die beiden Primfaktoren p und q ist das Wurzelziehen modulo N. Die quadratischen

Reste modulo N

$$\mathrm{QR}_N = \{a^2 \bmod N \mid a \in \mathbb{Z}_N^*\} \simeq \mathrm{QR}_p \times \mathrm{QR}_q$$

bilden eine $\mathbb{Z}_N^*$-Untergruppe vom Index 4. Quadrieren modulo eines RSA-Moduls N ist eine 4:1-Abbildung. Die Gleichverteilung auf QR_N ist effizient erzeugbar, zu zufälligem $a \in_R \mathbb{Z}_N^*$ ist $a^2 \bmod N$ uniform in QR_N verteilt. Modulo der Primzahl p (respektive q) entspricht Quadrieren einer 2:1-Abbildung, die beiden Wurzeln von $x \in \mathrm{QR}_p$ haben die Form $\pm a_p \bmod p$, unterscheiden sich also lediglich im Vorzeichen. Nach Chinesischem Restsatz $\mathbb{Z}_N^* \simeq \mathbb{Z}_p^* \times \mathbb{Z}_q^*$ sind die 4 Wurzeln zu $x \in \mathrm{QR}_N$ von der Form

$$\begin{aligned} a_1 &:= (+a_p, +a_q) & a_3 &:= (-a_p, +a_q) \\ a_2 &:= (-a_p, -a_q) = -a_1 & a_4 &:= (+a_p, -a_q) = -a_3. \end{aligned}$$

Unterscheiden sich zwei Wurzeln $a, b \in \mathbb{Z}_N^*$ zu $x \in \mathrm{QR}_N$ nicht nur im Vorzeichen, also $a \not\equiv \pm b \pmod N$, teilt genau einer der Primfaktoren die Differenz $a - b$ und der andere die Summe $a + b$. Dann erhalten wir die Faktorisierung mit Hilfe des Euklidischen Algorithmus':

$$\mathrm{ggT}(a \pm b, N) = \{p, q\}. \tag{2.1}$$

Mit Hilfe dieser Beobachtung reduziert man die Faktorisierung von N auf Wurzelziehen modulo N [R78]. Sei $\mathcal{A}$ ein Verfahren, dass mit Wahrscheinlichkeit π eine Wurzel zu $x \in_R \mathrm{QR}_N$ bestimmt. Wähle ein zufälliges $b \in_R \mathbb{Z}_N^*$ und berechne mit $\mathcal{A}$ eine Wurzel a zu $x := b^2 \bmod N$. Aus der Eingabe b^2 geht nicht hervor, welche der vier Wurzeln wir gewählt haben, weshalb die Ausgabe a eine von b unabhängige Wurzel ist. In der Hälfte aller Fälle gilt $a \not\equiv \pm b \pmod N$ und wir erhalten die Primfaktoren durch die ggT-Berechnung (2.1). Dank Algorithmus $\mathcal{A}$ faktorisieren wir in nahezu gleicher Zeit mit Wahrscheinlichkeit $\frac{1}{2}\pi$ den Modul. Bei Kenntnis der Primfaktorzerlegung ist Wurzelziehen modulo N nach Chinesischem Restsatz auf die Berechnung modulo p und q reduzierbar, wobei für prime Module effiziente Algorithmen zum Wurzelziehen existieren [MOV97]. Im Ergebnis sind Faktorisierung von N und Wurzelziehen modulo N äquivalente Probleme, wobei die Reduktionen jeweils sicherheitserhaltend sind.

Blum-Zahlen $N = pq$, spezielle RSA-Module des Typs $p, q \equiv 3 \pmod 4$, zeichnen sich dadurch aus, dass wegen $-1 \notin \mathrm{QR}_p$ und $-1 \notin \mathrm{QR}_q$ zu jedem quadratischen Rest exakt eine der Wurzeln modulo N ein quadratischer Rest ist und aufgrunddessen für einen solchen Modul Quadrieren einer Permutation auf QR_N entspricht. Schnorr [S96] hat diese Beobachtung auf beliebige RSA-Module verallgemeinert. Zu einem ungeraden Modul N mit

Primfaktorzerlegung $\prod_{i=1}^{r} p_i^{e_i}$ bezeichne η die kleinste, natürliche Zahl, so dass $2^{\eta+1}$ für $1 \leq i \leq r$ nicht $\varphi(p_i^{e_i})$ teilt. Die 2^η-ten Potenzen

$$\mathrm{HQR}_N := \{a^{2^\eta} \bmod N \mid a \in \mathbb{Z}_N^*\}$$

bilden offenbar eine Untergruppe von $\mathbb{Z}_N^*$. Für BLUM-Zahlen $N = pq$ gilt $\eta = 1$ und $\mathrm{HQR}_N = \mathrm{QR}_N$, denn wegen $p, q \equiv 3 \pmod 4$ sind $\frac{p-1}{2}$ und $\frac{q-1}{2}$ ungerade, so dass für diesen Modultyp Quadrieren auf HQR_N eine Permutation darstellt. Dies trifft auch für beliebige, ungerade Module zu [Ha99, S96]:

Satz 2.1.1. *Für einen ungeraden Modul N permutiert Quadrieren HQR_N.*

Beweis. Sei p^e eine ungerade Primzahlpotenz und η_p die größte ganze Zahl mit $2^{\eta_p} | \varphi(p^e)$. Die Gruppe $\mathbb{Z}_{p^e}^*$ ist zyklisch, somit entspricht die Anzahl der 2^{η_p}-ten Wurzeln modulo p^e

$$\mathrm{ggT}(|\mathbb{Z}_{p^e}^*|, 2^{\eta_p}) = 2^{\eta_p}.$$

Erheben in die 2^{η_p}-te Potenz modulo p^e ist folglich eine 2^{η_p}:1-Abbildung, und die 2^{η_p}-ten Potenzen modulo p^e bilden eine zyklische Untergruppe HQR_{p^e} mit ungerader Ordnung $|\mathbb{Z}_{p^e}^*|/2^{\eta_p}$. Weil auf dieser nach Korollar 1.1.2 von Seite 4 Quadrieren eine Permutation ist, gilt für $\eta \geq \eta_p$:

$$\mathrm{HQR}_{p^e} = \left\{ a^{2^{\eta_p}} \bmod p^e \;\middle|\; a \in \mathbb{Z}_{p^e}^* \right\} = \left\{ a^{2^\eta} \bmod p^e \;\middle|\; a \in \mathbb{Z}_{p^e}^* \right\}.$$

Aus dem Chinesischen Restsatz folgt

$$\mathrm{HQR}_N \simeq \mathrm{HQR}_{p_1^{e_1}} \times \mathrm{HQR}_{p_2^{e_2}} \times \cdots \times \mathrm{HQR}_{p_r^{e_r}}$$

für die Primfaktorzerlegung $\prod_{i=1}^{r} p_i^{e_i}$ des Moduls N. Da nach Wahl von η Quadrieren auf $\mathrm{HQR}_{p_i^{e_i}}$ jeweils bijektiv ist, erhalten wir die Behauptung. □

Der Beweis zu Satz 2.1.1 liefert als alternative Charakterisierung von HQR_N die Menge der $x \in \mathbb{Z}_N^*$ mit ungerader Ordnung $\mathrm{ord}_N(x)$. Weil Quadrieren auf HQR_N eine Permutation ist, folgt induktiv aus Satz 2.1.1, dass dies auch auf Erheben in die 2^k-te Potenz zutrifft. Wir erhalten als Konsequenz:

Korollar 2.1.2. *Seien $a, b \in \mathrm{HQR}_N$ und $k \geq 0$. Aus $a^{2^k} \equiv b^{2^k} \pmod N$ folgt $a \equiv b \pmod N$.*

Da auf $\mathbb{Z}_N^*$ Quadrieren hingegen eine 4:1-Abbildung ist, trifft die Folgerung von Korollar 2.1.2 nicht auf $a, b \in \mathbb{Z}_N^*$ zu, man denke zum Beispiel an $a \equiv -b \pmod N$ mit $a^2 \equiv b^2 \pmod N$. Mit der im Beweis von Satz 2.1.1 verwandten Notation ist die Anzahl der 2^η-ten Wurzeln von $a \in \mathrm{HQR}_N$ gleich $2^{\sum_i \eta_{p_i}}$, und weil Quadrieren auf HQR_N einer Permutation entspricht, gilt diese Anzahl auch für die 2^k-ten Wurzeln mit $k \geq \eta$:

Korollar 2.1.3. *Sei $k \geq \eta$. Dann hängt zu $a \in \mathrm{HQR}_N$ die Anzahl der 2^k-ten Wurzeln modulo N nur von N, aber nicht von a ab.*

Für ungerade n-Bit-Module $N = \prod_{i=1}^{r} p_i^{e_i}$ gilt $1 \leq \eta < n$, da $\varphi(p_i^{e_i}) < N$ gerade ist. Um die uniforme Verteilung auf HQR_N zu erzeugen, wähle ein zufälliges $a \in_R \mathbb{Z}_N^*$ und erhebe es in die 2^n-te Potenz modulo N. Diese Potenz ist gleichverteilt in HQR_N, denn a^{2^η} ist uniform verteilt in HQR_N und nach Satz 2.1.1 sind die weiteren Quadrierungsschritte jeweils 1:1-Abbildungen:

Korollar 2.1.4. *Sei N ein RSA-Modul. Die uniforme Verteilung auf HQR_N ist ohne Kenntnis von η oder der Primfaktorzerlegung von N effizient erzeugbar.*

Analog zum RSA-Problem bezeichne $\mathsf{FactIndex}(\cdot)$ einen effizienten Index-Generator, der zur Eingabe 1^n in Polynomialzeit einen n-Bit-RSA-Modul $N = pq$, einen ganzzahligen Anpassungsparameter $\tau \geq \eta - 1$, $t \geq 1$ und ein unabhängiges, uniform verteiltes $g \in_R \mathrm{HQR}_N$ erzeugt: $(N, \tau, t, g) \leftarrow \mathsf{FactIndex}(1^n)$. Die Summe $\tau + t$ übernimmt später beim Repräsentationsproblem die Rolle des RSA-Exponenten.

Definition 2.1.5 (Faktorisierungsannahme). *Die Erfolgswahrscheinlichkeit jedes probabilistischen Polynomialzeit-Algorithmus' $\mathcal{A}$ für das Faktorisierungsproblem*

$$\mathrm{Ws}\left[\mathcal{A}(N, \tau) = \{p, q\}\right]$$

sei vernachlässigbar in n, wobei sich die Wahrscheinlichkeit auf die internen Münzwürfe von $\mathcal{A}$ und die Wahl $(N, \tau, \cdot, \cdot) \leftarrow \mathsf{FactIndex}(1^n)$ bezieht.

Bei dieser Formulierung der Faktorisierungsannahme erhält $\mathcal{A}$ durch τ eventuell Hinweise auf die Zweierpotenzen in $p - 1$ und $q - 1$. Wegen $1 \leq \eta \leq n$ kann ein Verfahren ohne die zusätzliche Eingabe $\tau \geq \eta - 1$ alle Werte η-Werte ausprobieren, die Laufzeit nimmt maximal um den Faktor n zu.

Die Faktorisierungsannahme ist neben der Prämisse, die Berechnung des Diskreten Logarithmus' sei schwierig, eine der Säulen der modernen Public-Key-Kryptographie. Parallel zur zunehmenden Bedeutung der Kryptographie stieg das Interesse der angewandten Mathematik an schnellen Verfahren, um eine Zahl in ihre Primfaktoren zu zerlegen. Zwar entstanden neue Faktorisierungsalgorithmen wie das *Number-Field-Sieve* (NFS) [LL93] oder die *Elliptic-Curve-Methode* (ECM) [CP01], dennoch kennt man trotz intensiver Forschung bisher nur Algorithmen mit subexponentieller Laufzeit, jedoch kein Polynomialzeit-Verfahren. Für konkrete Anwendungen werden zur Zeit RSA-Module mit Bitlänge 2024 empfohlen. LENSTRA und VERHEUL

[LV01] haben die bisherigen Fortschritte der Entwicklung neuer Faktorisierungsalgorithmen und die erzielten Resultate zusammengetragen, um in ihrer Studie diese Werte in die Zukunft zu extrapolieren. Bezogen auf RSA-Module der Bitlänge 2024 kommen sie zu dem Schluß, dass solche Zahlen erst nach dem Jahr 2020 in ihre Primfaktoren zu zerlegen seien. Mit der Faktorisierung von Modulen der doppelten Länge, also 4048 Bits, sei — sofern keine überraschenden Forschritte erzielt werden — ihrer Untersuchung zufolge erst in der Mitte des 21. Jahrhunderts zu rechnen.

Für das Repräsentationsproblem bezüglich der Faktorisierung ersetze den primen RSA-Exponenten e der RSA-Repräsentation durch die Zweier-Potenz $2^{\tau+t}$ für ein $\tau \geq \eta - 1$. Eine *Faktorisierungsrepräsentation* für $X \in \mathbb{Z}_N^*$ bezüglich $(N, \tau, t, g) \leftarrow \mathsf{FactIndex}(1^n)$ ist ein Paar $(x, r) \in \mathbb{Z}_{2^t} \times \mathbb{Z}_N^*$ mit:

$$X = g^x r^{2^{\tau+t}} \pmod{N}.$$

Für die Darstellung in Form einer Faktorisierungsrepräsentation kommen nur $X \in \mathrm{HQR}_N$ in Frage, denn für alle Paare (x, r) ist zum einen $g^x \in \mathrm{HQR}_N$ und wegen $\tau + t \geq \eta$ gilt ferner $r^{2^{\tau+t}} \in \mathrm{HQR}_N$. Seien (x_1, r_1) und (x_2, r_2) zwei Repräsentationen zu (N, τ, t, g). Im Gegensatz zur RSA-Darstellung impliziert $x_1 = x_2$ nicht, dass die Werte r_1 und r_2 ebenfalls identisch sind. So stellen (x, r) und $(x, -r)$ den gleichen Wert dar und ein unmittelbar abgeleitetes Repräsentationsproblem wäre trivial. Aus diesem Grund sind beim Faktorisierungsrepräsentationsproblem zwei Darstellungen mit unterschiedlichen x-Komponenten zu finden:

Definition 2.1.6 (Faktorisierungsrepräsentationsproblem). *Zu gegebenem Tupel* $(N, \tau, t, g) \leftarrow \mathsf{FactIndex}(1^n)$ *berechne ein* $X \in \mathbb{Z}_N^*$ *samt zweier Repräsentationen* $(x_1, r_1), (x_2, r_2) \in \mathbb{Z}_e \times \mathbb{Z}_N^*$ *bezüglich* (N, τ, t, g) *mit* $x_1 \neq x_2$.

Das Repräsentationsproblem ist auf die Berechnung einer $2^{\tau+t}$-ten Wurzel modulo N bzw. der Primfaktorzerlegung des Moduls reduzierbar, denn in Verbindung mit einer $2^{\tau+t}$-te Wurzel von g gilt für beliebiges $a \in \mathbb{Z}$:

$$g^x r^{2^{\tau+t}} \equiv X \equiv g^{x+a}(g^{-a/2^{\tau+t}} r)^{2^{\tau+t}} \pmod{N}.$$

Der Spezialfall der Faktorisierungsrepräsentation modulo einer Blum-Zahl N (mit $\tau = 0$) sowie $t = 1$ geht auf Brassard, Chaum und Crépeau [BCC88] zurück und wurde später von Damgård [D95] auf beliebige $t \geq 1$ erweitert. Für diese Parameter ist die Zerlegung des Moduls auf das Faktorisierungsrepräsentationsproblem zurückzuführen. Um die Aufgabe von Blum-Zahlen auf beliebige RSA-Module unter Beibehaltung der Äquivalenz zur Faktorisierung zu verallgemeinern, haben wir den modulabhängigen Anpassungsparameter τ eingeführt. Wir zeigen im Anschluß an nachstehendes

Lemma in Satz 2.1.8, dass Faktorisierung auf das zugehörige Repräsentationsproblem reduzierbar ist und als Konsequenz beide Probleme äquivalent sind. Analog zu RSA und dem zugehörigen Repräsentationsproblem, vergleiche Satz 1.2.4 auf Seite 8, sind die Reduktionen ebenfalls sicherheitserhaltend. Während man bei der RSA-Repräsentation zu zwei gegebenen Darstellungen das RSA-Problem direkt löst, verwenden wir bei der Faktorisierungsrepräsentation den Algorithmus, um zwei Darstellungen zu berechnen und dann in einem zweiten Schritt den Modul in die Primfaktoren zu zerlegen.

Lemma 2.1.7. *Der Algorithmus $\mathcal{A}$ löse das Faktorisierungsrepräsentationsproblem zu $(N, \tau, t, g) \leftarrow \mathsf{FactIndex}(1^n)$. Dann kann man eine $2^{\tau+1}$-te Wurzel von g modulo N in nahezu gleicher Zeit mit gleicher Wahrscheinlichkeit berechnen.*

Beweis. Seien (x_1, r_1) und (x_2, r_2) zwei Faktorisierungsrepräsentationen zu $X \in \mathbb{Z}_N^*$ mit $x_1 \neq x_2$. Setze $\Delta x := x_1 - x_2$ und $r := r_2 r_1^{-1} \bmod N$, wobei $1 \leq |\Delta x| < 2^t$. Es gilt:

$$g^{\Delta x} \equiv g^{x_1 - x_2} \equiv r_2^{2^{\tau+t}} r_1^{-2^{\tau+t}} \equiv r^{2^{\tau+t}} \pmod{N}. \tag{2.2}$$

Die Exponenten Δx und $2^{\tau+t}$ müssen nicht teilerfremd sein, der größte gemeinsame Teiler ist eine Zweier-Potenz maximal so groß wie $|\Delta x| < 2^t$. Sei $\mathrm{ggT}(\Delta x, 2^{\tau+t}) = 2^k$ für ein k mit $0 \leq k < t$. Es existieren $u, v \in \mathbb{Z}$ mit $u\Delta x + v2^{\tau+t} = 2^k$:

$$g^{2^k} \equiv g^{u\Delta x + v2^{\tau+t}} \equiv (g^{\Delta x})^u \cdot (g^v)^{2^{\tau+t}} \equiv (r^u g^v)^{2^{\tau+t}} \pmod{N}.$$

Für $b := (r^u g^v)^{2^{t-k-1}}$ gilt (Beachte $t - k \geq 1$):

$$g^{2^k} \equiv (b^{2^{\tau+1}})^{2^k} \pmod{N}. \tag{2.3}$$

Wegen $g, b^{2^{\tau+1}} \in \mathrm{HQR}_N$ folgt nach Korollar 2.1.2 aus dieser Kongruenz $g \equiv b^{2^{\tau+1}} \pmod{N}$ und im Ergebnis ist b eine $2^{\tau+1}$-te Wurzel von g modulo N. □

Wir wollen die Bedeutung des Anpassungsparameters $\tau \geq \eta - 1$ in Beweis zu Lemma 2.1.7 herausstellen. Sei $-1 \in \mathrm{QR}_N$ (insbesondere $\eta \geq 2$), $t \geq 2$ und für die von Algorithmus $\mathcal{A}$ gelieferten beiden Darstellungen gelte $r^2 \equiv -g^{-1} \pmod{N}$ und $\Delta x = -2^{t-1}$. Im Beweis zu Lemma 2.1.7 gilt $\mathrm{ggT}(\Delta x, 2^t) = 2^{t-1}$, demgemäß $k = t - 1$ sowie $u = v = 1$:

$$1 \cdot \Delta x + 1 \cdot 2^t = -2^{t-1} + 2 \cdot 2^{t-1} = 2^{t-1} = 2^k.$$

Dann ist $b := (r^u g^v)^{2^{t-k-1}} \equiv rg \pmod N$. Ohne den Anpassungsparameter τ lautet die Kongruenz (2.3):

$$g^{2^k} \equiv (b^2)^{2^k} \pmod N.$$

Wegen $b^2 \equiv -g \pmod N$ gilt daher $g^2 \equiv b^4 \pmod N$. Die ggT-Berechnung $\mathrm{ggT}(g \pm b^2, N)$ liefert jedoch nur $\{0, N\}$ statt der erhofften Primfaktoren. Durch den Anpassungsparameter $\tau \geq \eta - 1$ ist hingegen gewährleistet, dass b in der Tat eine $2^{\tau+1}$-te Wurzel von g ist und weil aus g nicht hervorgeht, welche $2^{\tau+1}$-te Wurzel a uns bekannt ist, ergibt sich in der Hälfte der Fälle die Faktorisierung des Moduls N:

Satz 2.1.8. *Der Algorithmus $\mathcal{A}$ löse das Faktorisierungsrepräsentationsproblem zu $(N, \tau, t, g) \leftarrow \mathsf{FactIndex}(1^n)$ mit Wahrscheinlichkeit π. Dann kann man N in nahezu gleicher Laufzeit mit Wahrscheinlichkeit mindestens $\frac{1}{2}\pi$ faktorisieren.*

Beweis. Wähle ein zufälliges $a \in_R \mathbb{Z}_N^*$ und ersetze $g := a^{2^{\tau+1}} \bmod N$. Weil $\tau+1 \geq \eta$ und Quadrieren HQR_N permutiert, ist g uniform in HQR_N verteilt. Berechne gemäß Lemma 2.1.7 eine $2^{\tau+1}$-te Wurzel b von g modulo N. Für $c := ab^{-1} \bmod N$ gilt:

$$c^{2^{\tau+1}} \equiv 1 \pmod N. \tag{2.4}$$

Für $i = \tau+1, \tau, \ldots, 1$ teste, ob $\mathrm{ggT}(c^{2^{i-1}} \pm 1, N)$ die Primfaktoren liefert. Wir zeigen, dass für $i = \eta$ dies mit Wahrscheinlichkeit $\frac{1}{2}$ der Fall ist. Da Quadrieren HQR_N permutiert, folgt wegen $\tau + 1 \geq \eta$ und $1, c^{2^\eta} \in \mathrm{HQR}_N$ aus Kongruenz (2.4):

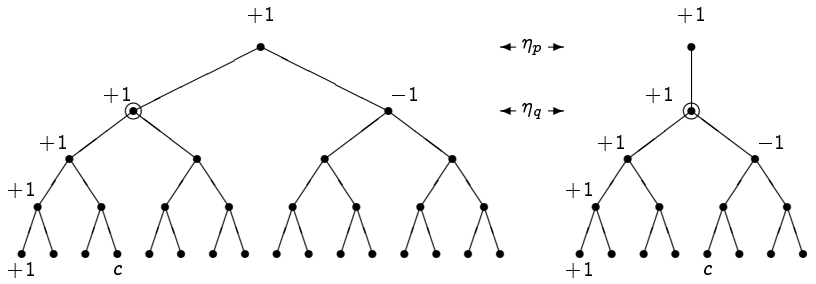
$$c^{2^{\tau+1}} \equiv c^{2^\tau} \equiv \cdots \equiv c^{2^\eta} \equiv 1 \pmod N.$$

Betrachte $c^{2^\eta} \equiv 1 \pmod N$ modulo der beiden Primfaktoren p und q von N. Seien $p - 1 = 2^{\eta_p} p'$ sowie $q - 1 = 2^{\eta_q} q'$ für ungerade p', q', wonach $\eta = \max\{\eta_p, \eta_q\}$. O.B.d.A. gelte $\eta = \eta_p$, also

$$c^{2^\eta} \equiv 1 \pmod p \qquad c^{2^{\eta-1}} \equiv \sigma_p \pmod p$$
$$c^{2^\eta} \equiv 1 \pmod q \qquad c^{2^{\eta-1}} \equiv \sigma_q \pmod q$$

mit $\sigma_p, \sigma_q \in \{\pm 1\}$.[1] Wir zeigen im weiteren, dass mit Wahrscheinlichkeit $\frac{1}{2}$ gilt $\sigma_p \sigma_q = -1$, weshalb dann $c^{2^{\eta-1}} \not\equiv \pm 1 \pmod N$ und die Berechnungen von $\mathrm{ggT}(c^{2^{\eta-1}} \pm 1, N)$ die beiden Primfaktoren des Moduls N liefert.

[1] Während σ_p beide Werte ± 1 annehmen kann, gilt $\sigma_q = +1$ unter der Prämisse $\eta_q < \eta$, denn $c^{2^{\eta-1}} \in \mathrm{HQR}_q$ und $-1 \notin \mathrm{HQR}_q$.



2^η-te Einheitswurzeln modulo p 2^η-te Einheitswurzeln modulo q

Abbildung 2.1: Faktorisierung mit Hilfe von 2^η-ten Einheitswurzeln

Die von $\mathcal{A}$ bestimmte $2^{\tau+1}$-te Wurzel b ist unabhängig von a. Daher sind $(c \bmod p)$ und $(c \bmod q)$ unabhängig und uniform verteilt in der Menge der $2^{\tau+1}$-ten und gleichbedeutend 2^η-ten Einheitswurzeln modulo p respektive modulo q. Als weitere Konsequenz sind σ_p und σ_q ebenfalls unabhängig. Für die Hälfte der 2^η-ten Einheitswurzeln w modulo p ist $w^{2^{\eta-1}}$ gleich -1, für die anderen $+1$. Für die uniform verteilte 2^η-ten Einheitswurzel $(c \bmod p)$ folgt, dass σ_p gleichverteilt in $\{\pm 1\}$ ist. Weil σ_p und σ_q unabhängig sind, besagt dies, dass in der Hälfte aller Fälle $\sigma_p\sigma_q = -1$ gilt. □

Abbildung 2.1 illustriert den Beweis zu Satz 2.1.8. Der linke Baum entspricht dem Quadrieren modulo p, der rechte modulo q. Die Knoten an den Spitzen (Baumwurzeln) der beiden Bäume sind mit $+1$ markiert, die Knoten (Blätter) mit den 2^{η_p}- bzw. 2^{η_q}-vielen 2^η-ten Einheitswurzeln modulo p bzw. q. Jeder Knoten von den Blättern in Richtung der Baumwurzel entspricht dem Quadrieren modulo p und modulo q. Die Abbildung zeigt den Fall $\eta_q = \eta_p - 1$, bei dem der letzte Quadrierungsschritt modulo q bijektiv ist. Jede 2^η-te Wurzel modulo N entspricht unabhängigen Pfaden von den Baumwurzeln zu einem Blatt in jedem Baum. Die linken Pfade in beiden Bäumen, wir bezeichnen sie als 1-Pfade, entsprechen der 2^η-ten Einheitswurzel $+1$ modulo p respektive modulo q. Diesen beiden 1-Pfaden werden über die berechnete 2^η-te Einheitswurzel c zwei zufällige und unabhängige Pfade, die wir als c-Pfade bezeichnen, gegenübergestellt. Man erhält mittels ggT-Berechnung die Faktorisierung, wenn von der Baumwurzel in Richtung der Blätter sich die Pfade in einem Baum trennen und im anderen den gleichen Sohnknoten nehmen. In Abbildung 2.1 ist dies im markierten Knoten auf Stufe η_q der Fall. Es gilt

$$c^{2^{\eta_q}} \equiv 1 \pmod p \qquad c^{2^{\eta_q-1}} \equiv +1 \pmod p$$
$$c^{2^{\eta_q}} \equiv 1 \pmod q \qquad c^{2^{\eta_q-1}} \equiv -1 \pmod q,$$

weshalb $\mathrm{ggT}(c^{2^{\eta_q-1}} \pm 1, N) = \{p, q\}$. Im Beweis zu Satz 2.1.8 haben wir nur die Wahrscheinlichkeit analysiert, mit der sich auf Stufe η_p exakt in einem der Bäume die Pfade verzweigen. Für Nicht-Blum-Zahlen, also $\eta > 1$, besteht die Möglichkeit, dass zwar 1- und c-Pfade die jeweils gleichen Knoten nach der Baumwurzel wählen, aber dann in einem nachfolgenden Knoten in genau einem Baum verzweigen und wir faktorisieren können. Aus diesem Grund ist die Erfolgswahrscheinlichkeit in Satz 2.1.8 für Nicht-Blum-Zahlen sogar größer als $\frac{1}{2}$.

Die Aussage von Satz 2.1.8 gilt mit auch für festes $g \in \mathrm{HQR}_N$, wenn $\tau \geq 1$ und eine 2^τ-te Wurzel $a \in \mathbb{Z}_N^*$ von g öffentlich bekannt ist, wo weder sie noch ihr Negatives ein quadratischer Rest ist: $\pm a \notin \mathrm{QR}_N$. Man quadriere die berechnete $2^{\tau+1}$-te Wurzel b, so dass wegen $\pm a \notin \mathrm{QR}_N$ und $b^2 \in \mathrm{QR}_N$ für $c := ab^{-2}$ gilt $\pm c \notin \mathrm{QR}_N$. Diese bedingt, die 2^τ-te Einheitswurzel c ist modulo genau einer der Primfaktoren p oder q ein quadratischer Rest, also ungleich ± 1 modulo N. Als konkretes Beispiel nennt Halevi [Ha99] eine William-Zahl $N = pq$ mit $p \equiv 3 \pmod 4$ und $q \equiv 7 \pmod 8$ in Verbindung mit $g := 4 \bmod N$ sowie $\tau := \eta = 1$. Es gilt $\pm 2 \notin \mathrm{QR}_N$, ferner $4 \in \mathrm{HQR}_N$, denn $(+2, -2) \in \mathrm{QR}_p \times \mathrm{QR}_q$ ist ein quadratischer Rest modulo N. Als weitere Modifikationen des Faktorisierungsrepräsentationsproblems sind denkbar:

- Modul N: Ersetzt man den RSA Modul N durch einen beliebigen, ungeraden Modul, liefert das Verfahren zu Satz 2.1.8 statt der kompletten Faktorisierung einen nicht-trivialen Primfaktor.

- Parameter t: Statt t vorzugeben, kann man zu gegebenem (N, τ, g) den Parameter $t \geq 1$ selbständig bestimmen, so dass eine Repräsentation von X ein Tupel (x, r, t) mit $X \equiv g^x r^{2^{\tau+t}} \pmod N$ ist. Im Beweis zu Lemma 2.1.7 wähle $r := r_2 r_1^{-2^{t_1-t_2}}$ zu zwei Repräsentationen (x_1, r_1, t_1) und (x_2, r_2, t_2), für die o.B.d.A. $t_1 \geq t_2$ gelte.

- Exponent $2^{\tau+t}$: Teilt $e = \mathcal{O}(\log n)$ die Gruppenordnung $\varphi(N)$, ist die Zweier-Potenz $2^{\tau+t}$ durch $e^{\tau+t}$ zu ersetzen, wobei η die kleinste, ganze Zahl bezeichne, so dass $e^{\eta+1}$ weder $p-1$ noch $q-1$ teilt. Verwende statt HQR_N die Untergruppe $\{x^{e^\eta} \mid x \in \mathbb{Z}_N^*\}$. Nach Ohto und Okamoto [OO88] ist die Berechnung der e-ten Wurzel in diesem Fall äquivalent zur Faktorisierung.

2.2 Identifikation basierend auf Faktorisierung

Aus Okamotos Identifikation $\langle \mathcal{P}, \mathcal{V} \rangle_{\mathsf{RSA}}$ basierend auf der RSA-Repräsentation aus Kapitel 1.3 leiten wir ein Schema $\langle \mathcal{P}, \mathcal{V} \rangle_{\mathsf{Fact}}$ aufbauend auf der

Faktorisierungsdarstellung her, dessen Sicherheit beweisbar auf der Faktorisierungsannahme beruht. OKAMOTO [Ok92] gibt eine Variante eines RSA-basierten Schemas mit $2e$ statt e als Exponenten und formuliert (ohne Beweis), dass die Sicherheit äquivalent zur Faktorisierung des Moduls sei. Um das Protokoll mit Wahrscheinlichkeit $\frac{1}{2}$ zu bestehen, genügt es allerdings, das RSA-Problem zu lösen [FF02].

Beim Schema $\langle \mathcal{P}, \mathcal{V} \rangle_{\mathsf{Fact}}$ gibt die Trusted-Party $\mathcal{T}$ als Public-Parameters $(N, \tau, t, g) \leftarrow \mathsf{FactIndex}(1^n)$ vor und wählt als geheimen Schlüssel $(x, r) \in_{\mathsf{R}} \mathbb{Z}_{2^t} \times \mathbb{Z}_N^*$. Der Public-Key zu (x, r) ist $X := g^x r^{2^{\tau+t}} \bmod N$. Mit anderen Worten, der Secret-Key zum öffentlichen Schlüssel X ist eine Faktorisierungsrepräsentation bezüglich (N, e, g). Der Teilnehmer kann, muß aber weder eine $2^{\tau+t}$-te Wurzel von g noch die Faktorisierung von N kennen. Ohne dieses Wissen können sich mehrere Teilnehmer (Prover) den gleichen Modul N als auch t teilen. Analog zum Schema basierend auf der RSA-Repräsentation wollen wir zur Vereinfachung annehmen, 2^{-t} sei vernachlässigbar, da anderenfalls mehrere Runden des Protokolls auszuführen sind.

Abbildung 2.2: Identifikation $\langle \mathcal{P}, \mathcal{V} \rangle_{\mathsf{Fact}}$ auf Faktorisierungsrepräsentation

Prover $\mathcal{P}$	N, τ, t, g	**Verifier** $\mathcal{V}$
Repräsentation x, r	$X = g^x r^{2^{\tau+t}} \bmod N$	
Wähle $y \in_{\mathsf{R}} \mathbb{Z}_{2^{\tau+t}}$, $s \in_{\mathsf{R}} \mathbb{Z}_N^*$ $Y := g^y s^{2^{\tau+t}} \bmod N$	$\xrightarrow{\quad Y \quad}$	
	$\xleftarrow{\quad c \quad}$	Wähle $c \in_{\mathsf{R}} [0, 2^t)$
$z := y + cx \bmod 2^{\tau+t}$ $W := s r^c g^{\lfloor (y+cx)/2^{\tau+t} \rfloor} \bmod N$	$\xrightarrow{\quad W, z \quad}$	$Y X^c \stackrel{!}{\equiv} W^{2^{\tau+t}} g^z \ (N)$

Abbildung 2.2 zeigt das Protokoll $\langle \mathcal{P}, \mathcal{V} \rangle_{\mathsf{Fact}}$ des Identifikationsschemas. Die direkte Übernahme von OKAMOTOS Protokoll $\langle \mathcal{P}, \mathcal{V} \rangle_{\mathsf{RSA}}$, Abbildung 1.1 auf Seite 11, ist nicht möglich, weil bei der Faktorisierungsrepräsentation der Exponent x aus $\mathbb{Z}_{2^t}$ und nicht $\mathbb{Z}_{2^{\tau+t}}$ ist. Wir übertragen das Protokoll der RSA-Repräsentation mit dem Exponenten $2^{\tau+t}$ statt e, beschränken allerdings den Secret-Key x als auch die Challenge c auf jeweils t Bits. Damit bleibt einerseits die Completeness-Eigenschaft (vergleiche Gleichung

(1.1) auf Seite 11)

$$\begin{aligned} YX^c &\equiv (g^y s^{2^{\tau+t}})(g^x r^{2^{\tau+t}})^c \\ &\equiv s^{2^{\tau+t}} r^{c2^{\tau+t}} g^{y+cx} \\ &\equiv (sr^c g^{\lfloor (y+cx)/2^{\tau+t} \rfloor})^{2^{\tau+t}} \cdot g^{(y+cx) \bmod 2^{\tau+t}} \\ &\equiv W^{2^{\tau+t}} g^z \pmod{N}. \end{aligned}$$

erhalten, andererseits ist die Sicherheit auf die Faktorisierung des Moduls N reduzierbar:

Satz 2.2.1. *Unter der Faktorisierungsannahme ist das Identifikationsschema $\langle \mathcal{P}, \mathcal{V} \rangle_{\mathsf{Fact}}$ aus Abbildung 2.2 sicher gegen Angriffe aus dem Stand, denn besteht $\mathcal{A}_{\mathsf{Adhoc}}$ das Protokoll mit Wahrscheinlichkeit π, so kann man in erwarteter Laufzeit $2 \cdot |\mathcal{A}_{\mathsf{Adhoc}}|$ mit Wahrscheinlichkeit mindestens $\frac{1}{4}(\pi - 2^{-t})$ den Modul N faktorisieren.*

Beweis. Der Beweis orientiert sich am Knowledge-Extraktor zu OKAMOTOS Identifikationsschema basierend auf der RSA-Repräsentation (Satz 1.3.1 auf Seite 12). Wir zeigen, wie mit Hilfe des betrügerischen Provers $\mathcal{A}_{\mathsf{Adhoc}}$ zu $(N, \tau, t, g) \leftarrow \mathsf{FactIndex}(1^n)$ eine $2^{\tau+1}$-te Wurzel von g modulo N in erwarteter Laufzeit $2 \cdot |\mathcal{A}_{\mathsf{Adhoc}}|$ mit Wahrscheinlichkeit $\frac{1}{2}(\pi - 2^{-t})$ zu berechnen ist. Die Aussage des Satzes folgt dann wie im Beweis zu Satz 2.1.8 auf Seite 39, wonach eine $2^{\tau+1}$-te Wurzel von g modulo N in der Hälfte der Fälle die Faktorisierung des Moduls liefert.[2]

Wähle zufälliges $(x, r) \in_{\mathsf{R}} \mathbb{Z}_{2^t} \times \mathbb{Z}_N^*$. Simuliere eine Protokollausführung mit dem Prover $\mathcal{A}_{\mathsf{Adhoc}}$ zu (N, τ, t, g) und $X := g^x r^{2^{\tau+t}} \bmod N$, wobei der Zustand nach Senden von Y gespeichert wird. Sollte $\mathcal{A}_{\mathsf{Adhoc}}$ das Protokoll nicht bestehen, dann stoppe. Sonst setze $\mathcal{A}_{\mathsf{Adhoc}}$ an die Stelle zurück, an welcher er im Anschluß an sein Commitment Y auf die Challenge c wartet, und wiederhole die Ausführung, bis der betrügerische Prover $\mathcal{A}_{\mathsf{Adhoc}}$ das Protokoll erneut besteht. Auf diese Weise erhält man zwei erfolgreiche Ausführungen (Y, c_1, W_1, z_1) und (Y, c_2, W_2, z_2) für dasselbe Commitment Y:

$$\begin{aligned} YX^{c_1} &\equiv W_1^{2^{\tau+t}} g^{z_1} \pmod{N} \\ YX^{c_2} &\equiv W_2^{2^{\tau+t}} g^{z_2} \pmod{N}. \end{aligned}$$

Aus diesen Kontrollgleichungen folgt

$$X^{-c_1} W_1^{2^{\tau+t}} g^{z_1} \equiv Y \equiv X^{-c_2} W_2^{2^{\tau+t}} g^{z_2} \pmod{N}$$

[2] Im Unterschied zu Satz 1.3.1 extrahieren wir keine Repräsentation des Public-Keys, sondern berechnen eine $2^{\tau+1}$-te Wurzel von g.

und durch Umformung erhält man in Verbindung mit $X \equiv g^x r^{2^{\tau+t}} \pmod N$:

$$g^{z_1 - z_2 + x(c_2 - c_1)} \equiv (r^{c_1 - c_2} W_1^{-1} W_2)^{2^{\tau+t}} \pmod N.$$

Mit $\Delta z := z_1 - z_2$ und $\Delta c := c_2 - c_1$ können wir dies schreiben als:

$$g^{\Delta z + x\Delta c} \equiv (r^{c_1 - c_2} W_1^{-1} W_2)^{2^{\tau+t}} \pmod N.$$

Diese Kongruenz hat die Form wie Gleichung (2.2) auf Seite 38 im Beweis zu Lemma 2.1.7. Wir erhalten auf die gleiche Art und Weise in der vorliegenden Situation eine $2^{\tau+1}$-te Wurzel von g, wenn $\mathrm{ggT}(\Delta z + x\Delta c, 2^{\tau+t}) = 2^k$ für ein k mit $0 \leq k < t$. Es genügt im folgenden die Wahrscheinlichkeit, dass der größte gemeinsame Teiler mindestens 2^t ist, nach oben durch $\frac{1}{2}$ zu beschränken. Offenbar gilt:

$$\mathrm{ggT}(\Delta z + x\Delta c, 2^{\tau+t}) \geq 2^t \quad \Longleftrightarrow \quad x \cdot \Delta c \equiv -\Delta z \pmod{2^t}. \tag{2.5}$$

Für feste Δz und Δc kann man — sofern diese Kongruenz lösbar ist — wegen $0 < |\Delta c| < 2^t$ nach Satz 1.1.1 auf Seite 3 die Anzahl der Lösungen angeben mit $2^j := \mathrm{ggT}(\Delta c, 2^t)$ und $0 \leq j < t$. Für die Simulation wählen wir zuerst x, die Werte Δc und Δz werden später festgelegt. Es gilt:

- Der Public-Key $X \equiv g^x r^{2^{\tau+t}} \pmod N$ ist unabhängig von x verteilt, denn $Xg^{-x} \in \mathrm{HQR}_N$ hat für alle x die gleiche Anzahl $2^{\tau+t}$-ter Wurzeln r (Korollar 2.1.3 auf Seite 36).
- Die Challenge c wird bei der Simulation gemäß Protokollspezifikation zufällig und unabhängig aus $\mathbb{Z}_{2^t}$ gewählt. Aus diesem Grund hängt Δc nicht von x ab.
- Die Antwort z hängt nur von (X, s, Y, c) und internen Münzwürfen von $\mathcal{A}_{\mathsf{Adhoc}}$, aber nicht von x ab.

Als Folgerung entspricht die simulierte Situation dem Fall, dass wir zuerst $\Delta c, \Delta z$ wählen und anschließend ein zufälliges $x \in \mathbb{Z}_{2^t}$. Mit Wahrscheinlichkeit 2^{j-t} erfüllt x die Kongruenz

$$x \cdot \Delta c \equiv -\Delta z \pmod{2^t}$$

und aufgrund $j < t$ ist nach Äquivalenz (2.5) die Wahrscheinlichkeit für $\mathrm{ggT}(\Delta z + x\Delta c, 2^{\tau+t}) \geq 2^t$ maximal $\frac{1}{2}$. □

Die Sicherheit des Identifikationsschemas $\langle \mathcal{P}, \mathcal{V} \rangle_{\mathsf{Fact}}$ haben wir im obigen Satz 2.2.1 unmittelbar auf Faktorisierung des Moduls zurückgeführt, ohne wie im Fall des RSA-Schemas eine Repräsentation zu extrahieren. In der

Tat ist das Protokoll $\langle \mathcal{P}, \mathcal{V} \rangle_{\mathsf{Fact}}$ aus Abbildung 2.2 wie das ONG-SCHNORR-Schema [OS90, Sh99] kein Proof-Of-Knowledge. Ein Prover $\widetilde{\mathcal{P}}$, welcher über eine Darstellung (x, r) des Public-Keys X zu (N, τ, t, g) mit $x_1 = 2^t - 1$ verfügt, besteht mit (x_1, r_1) auch das Protokoll bezüglich (N, τ, t, g^2) für die Hälfte der Challenges (genauer den geraden c) *ohne* eine entsprechende Repräsentation zu kennen. Die Fähigkeit, (x_1, r_1) in eine Darstellung (x_2, r_2) bezüglich (N, τ, t, g^2) zu überführen, ist gleichbedeutend mit der Kenntnis der Faktorisierung: Der gleiche Ansatz wie in Lemma 2.1.7 von Seite 38

$$g^{x_1} r_1^{2^{\tau+t}} \equiv g^{2x_2} r_2^{2^{\tau+t}} \pmod{N}$$

liefert $|\Delta x| = |2x_2 - x_1| \leq 2(2^t - 1) - (2^t - 1) < 2^t$ und wir erhalten über eine ggT-Berechnung eine $2^{\tau+1}$-te Wurzel von g, was eine effiziente Zerlegung des Moduls in seine Primfaktoren ermöglicht.

Lemma 2.2.2. *Das Identifikationsschema $\langle \mathcal{P}, \mathcal{V} \rangle_{\mathsf{Fact}}$ aus Abbildung 2.2 erfüllt die perfekte Witness-Indistinguishability.*

Beweis. Wir übertragen den Beweis zu Lemma 1.3.6, Seite 18, vom RSA- auf die Faktorisierungsrepräsentation. Der Prover $\mathcal{P} = \mathcal{P}(X, x, r)$ verfüge über die Repräsentation (x, r), der Prover $\mathcal{P}' = \mathcal{P}(X, x', r')$ über die Darstellung (x', r') von X. Wir zeigen, die Daten (Y, c, W, z) entsprechen mit gleicher Wahrscheinlichkeit der Protokollausführung von $\mathcal{P}$ und $\widetilde{\mathcal{V}}(X)$ als auch der von $\mathcal{P}'$ und $\widetilde{\mathcal{V}}(X)$. Setze $\Delta x := x' - x$ und $\Delta r := r(r')^{-1} \bmod N$. Wegen $r^{2^{\tau+t}} \equiv X g^{-x} \pmod{N}$ gilt $(\Delta r)^{2^{\tau+t}} \equiv g^{\Delta x} \pmod{N}$. $\mathcal{P}$ wähle im ersten Schritt $y \in_{\mathsf{R}} \mathbb{Z}_{2^{\tau+t}}$ und $s \in_{\mathsf{R}} \mathbb{Z}_N^*$. $\mathcal{P}'$ votiert mit gleicher Wahrscheinlichkeit für

$$\begin{aligned} y' &:= y - c \cdot \Delta x \bmod 2^{\tau+t} \\ s' &:= s \cdot \Delta r^c \bmod N. \end{aligned}$$

Aufgrund $Y' = Y$ stellt bei beiden Ausführungen $\widetilde{\mathcal{V}}$ die Challenge c mit gleicher Wahrscheinlichkeit. Die Antworten (z, W) respektive (z', W') sind durch den Secret-Key, die hinterlegten Werte (y, s) bzw. (y', s') und die Challenge c vorgegeben. Einsetzen der Daten ergibt $(z, W) = (z', W')$. Die Wahrscheinlichkeit, dass bei einer Protokollausführung die Kommunikation (Y, c, W, z) ist, stimmt bei beiden Provern $\mathcal{P}$ und $\mathcal{P}'$ überein. □

Aufgrund der Witness-Indistinguishability gilt der Beweis der Sicherheit gegen Angriffe aus dem Stand, Satz 2.2.1, auch, wenn man mit Hilfe der gewählten Repräsentation (x, r) von X zuvor die Protokollausführungen mit dem aktiven Angreifer $\mathcal{A}$ als Verifier simuliert:

Satz 2.2.3. *Unter der Faktorisierungsannahme ist das Identifikationsschema $\langle \mathcal{P}, \mathcal{V} \rangle_{\mathsf{Fact}}$ aus Abbildung 2.2 sicher gegen aktive und parallele Angriffe, denn besteht $\mathcal{A}$ das Protokoll mit Wahrscheinlichkeit π, so kann man in erwarteter Laufzeit $2 \cdot |\mathcal{A}|$ mit Wahrscheinlichkeit mindestens $\frac{1}{4}(\pi - 2^{-t})$ den Modul N faktorisieren.*

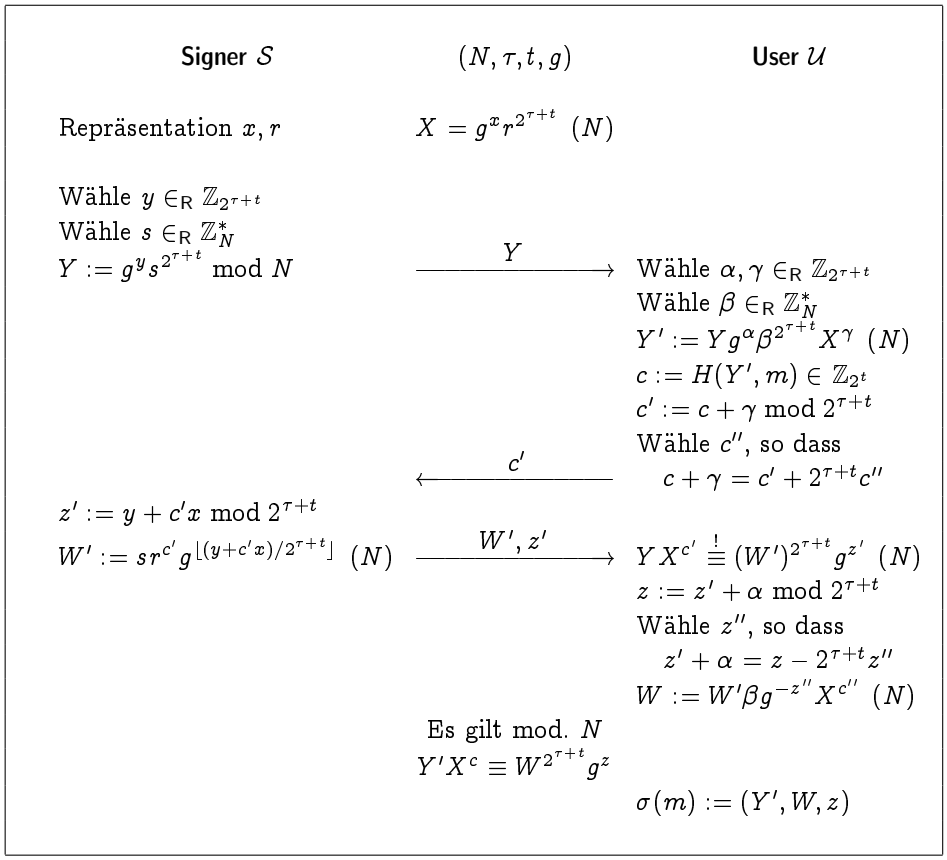
Wir vergleichen das vorgestellte Schema $\langle \mathcal{P}, \mathcal{V} \rangle_{\mathsf{Fact}}$ mit bekannten Identifikationsverfahren (1-Runde und Sicherheitsparameter t), deren Sicherheit ebenfalls auf der Faktorisierungsannahme beruhen. Das Schema von FIAT und SHAMIR [FS86] ist sicher gegen parallele Angriffe, das Protokoll stellt einen Proof-Of-Knowledge für Quadratwurzeln modulo N dar. Nachteilig ist die Länge des Public- bzw. Secret-Keys, die jeweils aus t-vielen $\mathbb{Z}_N^*$-Werten bestehen. Das an dieses Verfahren angelehnte ONG-SCHNORR-Schema [OS90] basiert auf 2^t-ten Wurzeln, weshalb der Schlüssel nur eine Komponente umfaßt. Die Sicherheit gegen sequentielle Angriffe im Fall von BLUM-Zahlen wurde von SHOUP [Sh99] bewiesen, SCHNORR [S96, S97] hat die Situation beliebiger RSA-Module behandelt. Das ONG-SCHNORR-Protokoll ist wie das vorgestellte $\langle \mathcal{P}, \mathcal{V} \rangle_{\mathsf{Fact}}$ kein Proof-Of-Knowledge [Sh99]. Von GIRAULT [Gi91] stammt eine Variante der Diskreten-Logarithmus basierenden SCHNORR-Identifikation, wo der prime Modul durch einen RSA-Modul ersetzt wird und die Modulo-Reduktion der Response entfällt. Die Sicherheitsanalyse stammt von POUPARD und STERN [PoSt98, PoSt99, PoSt00], weshalb man auch vom GIRAULT-POUPARD-STERN- oder kurz GPS-Schema spricht. Die Sicherheit des Identifikationsverfahrens gegen aktive Angriffe beruht auf der Zero-Knowledge-Eigenschaft des Protokolls [FFS88, Go98] und nicht auf Witness-Indistinguishability. Dies hat eine kurze Challenge zur Folge und die notwendigen, mehreren Runden führen zu einem Effizienzverlust. POINTCHEVAL [P00] hat gezeigt, dass man ein einfaches, (statistisch) witness-indistinguishable 1-Runden-Protokoll aus dem GPS-Schema erhält, wenn der vom Prover $\mathcal{P}$ als erstes hinterlegte Wert im Vergleich zur Challenge und Secret-Key hinreichend groß ist.

2.3 Unterschriften basierend auf Faktorisierung

Wie beim Identifikationsschema von OKAMOTO in Kapitel 1.4 leiten wir aus $\langle \mathcal{P}, \mathcal{V} \rangle_{\mathsf{Fact}}$ ein Unterschriftensystem her. Um eine Nachricht m zu unterzeichnen, wählt der Signer $\mathcal{S}$ ein $y \in_R \mathbb{Z}_{2^{\tau+t}}$ sowie $s \in_R \mathbb{Z}_N^*$ und bildet die Unterschrift mit Hilfe des Secret-Keys (x, r) analog zum Identifikationsprotokoll:

$$\sigma(m) := (\underbrace{g^y s^{2^{\tau+t}} \bmod N}_{=Y}, \underbrace{y + cx \bmod 2^{\tau+t}}_{=z}, \underbrace{s r^c g^{\lfloor (y+cx)/2^{\tau+t} \rfloor} \bmod N}_{=W})$$

Abbildung 2.3: Blinde Unterschrift $\langle \mathcal{S}, \mathcal{U} \rangle_{\mathsf{faktBlSig}}$



mit $c := H(Y, m)$. Um eine Unterschrift $\sigma(m) = (Y, z, W)$ zu prüfen, verifiziert man anhand des Public-Keys X, ob

$$Y X^{H(Y,m)} \stackrel{!}{\equiv} W^{2^{\tau+t}} g^z \pmod{N}.$$

Der Sicherheitsbeweis von OKAMOTO [Ok92] als auch derjenige von POINTCHEVAL und STERN [PSt97] im Fall des Signatur-Schemas basierend auf der RSA-Repräsentation übertragen sich unmittelbar:

Satz 2.3.1. *Unter der Faktorisierungsannahme ist das aus dem Identifikationsschema $\langle \mathcal{P}, \mathcal{V} \rangle_{\mathsf{Fact}}$ abgeleitete Unterschriftenschema im Random-Oracle-Modell bei Adaptive-Chosen-Message-Angriffen sicher gegen Existential-Forgery.*

Exakt wie im Fall der RSA-Repräsentation als Basis des zugrundeliegendenden Identifikationsverfahrens gilt für das Blinde-Unterschriften-Schema (vergleiche Lemma 2.3.2 auf Seite 48):

Lemma 2.3.2. *Das Unterschriften-Schema* $\langle \mathcal{S}, \mathcal{U} \rangle_{\mathsf{RSABlSig}}$ *aus Abbildung 2.3 erfüllt die perfekte Blindheit.*

Der Beweis von POINTCHEVAL und STERN [PSt97] zur Sicherheit der blinden Unterschriften überträgt sich auf die Faktorisierungsrepräsentation als Grundlage des Signatur-Schemas:

Satz 2.3.3. *Unter der Faktorisierungsannahme ist das Blinde-Unterschriften-Schema* $\langle \mathcal{S}, \mathcal{U} \rangle_{\mathsf{faktBlSig}}$ *aus Abbildung 2.3 im Random-Oracle-Modell für bis zu poly-logarithmisch vielen Interaktionen* ℓ *mit dem Signatur-Orakel sicher gegen (parallele) One-More-Forgery.*

Das Blinde-Unterschriften-Schema $\langle \mathcal{S}, \mathcal{U} \rangle_{\mathsf{faktBlSig}}$ ist allerdings vom Aufwand her dem der GPS-Variante von POINTCHEVAL [P00] unterlegen.

Kapitel 3

Commitment-Schemata

To Commit Or Not To Commit

— Session-Titel Crypto 2000

If it is provably secure,
it's probably not.

— Lars R. Knudsen

Wir stellen den von Dolev, Dwork und Naor [DDN00] eingeführten Non-Malleable-Begriff bezogen auf Commitment-Schemata vor und geben im Anschluß ein solches Verfahren basierend auf der RSA-Repräsentation und einem Proof-Of-Knowledge wieder [FF00]. Eine Konstruktion mit dem Chinesischen Restsatz eröffnet die Möglichkeit des Hash-&-Commit-Prinzips und der Faktorisierungsrepräsentation als Grundlage des Commitments. In Abschnitt 3.5 werden einfache, auf den Repräsentationsproblemen beruhende Verfahren mit einer abgeschwächten Non-Malleability-Eigenschaft beschrieben [FF02].

3.1 Einfache Hinterlegungsverfahren

Analog zur RSA-Darstellung erhalten wir aus der Faktorisierungsrepräsentation ein Commitment-Schema mit perfekter Geheimhaltung, dessen Bindung auf der Faktorisierungsannahme beruht. Die Geheimhaltung gilt, da $r^{2^{\tau+t}}$ unabhängig von m (bzw. $g^m \in \mathrm{HQR}_N$) uniform in HQR_N verteilt ist.

Satz 3.1.1. *Mit* $(N, \tau, t, g) \leftarrow \mathsf{FactIndex}(1^n)$ *und* $r \in_R \mathbb{Z}_N^*$ *ist unter der Faktorisierungsannahme*

$$\mathrm{com}(m, r) := g^m r^{2^{\tau+t}} \bmod N$$

ein polynomialzeit-bindendes, nicht-interaktives Commitment-Schema mit perfekter Geheimhaltung.

Die Parameter (N, τ, t, g) kann eine Trusted-Party $\mathcal{T}$ systemweit vorgeben oder der Receiver $\mathcal{R}$ stellt diese dem Sender $\mathcal{S}$ zur Verfügung. Durch die Wahl seitens des Receivers stellt dieser sicher, dass $\mathcal{S}$ nicht über die zum mehrdeutigen Aufdecken notwendige Primfaktorzerlegung des Moduls verfügt. Für den Sender $\mathcal{S}$ stellen allerdings eventuell fehlerhafte Parameter die Geheimhaltung in Frage, denn beispielsweise erhält ein betrügerischer $\tilde{\mathcal{R}}$ über ein $g \in \mathbb{Z}_N^*$ mit Jacobi-Symbol -1 durch das Jacobi-Symbol des Commitments das unterste Bit der hinterlegten Nachricht. Um diese Sicherheitslücken zu schließen, schlägt HALEVI [Ha99] (implizit) dem Sender $\mathcal{S}$ vor, nur ungerade N zu akzeptieren und τ durch die Bitlänge n zu ersetzen. Ferner soll $\mathcal{S}$ zunächst g in die 2^n-te Potenz heben. Wegen $g^{2^n} \in \mathrm{HQR}_N$ ist das Commitment dann in jedem Fall ein unabhängig von m uniform verteilter Wert in HQR_N. Um einerseits die Nachrichtenlänge bei t Bits zu belassen und andererseits die Bindungseigenschaft weiterhin auf Faktorisierung zu reduzieren, erhöhe die Zweierpotenz im Exponenten:

$$\mathrm{com}(m, r) := g^{2^n m} r^{2^{2n+t}} \bmod N.$$

Bei diesem Commitment ist die perfekte Geheimhaltung auch bei Vorlage falsch gebildeter Parameter (N, τ, t, g) durch einen betrügerischen Receiver $\tilde{\mathcal{R}}$ bei ungeradem Modul N gewährleistet und der Sender $\mathcal{S}$ ist weiterhin einem ehrlichen Empfänger gegebenüber unter der Faktorisierungsannahme an seine verbriefte Nachricht gebunden.

3.2 Non-Malleability

DOLEV, DWORK und NAOR [DDN00] haben *Non-Malleability* als eine Erweiterung der semantischen Sicherheit von Verschlüsselungsverfahren vorgestellt und auf Commitment-Schemata, genauer deren Geheimhaltung, übertragen. Der Sender $\mathcal{S}$ hinterlege beim Receiver $\mathcal{R}$ die Nachricht m in Form von $\langle \mathcal{S}, \mathcal{R} \rangle_{\mathsf{com}}(m)$. Informell ausgedrückt ist das Schema *malleable* (was wörtlich übersetzt verformbar heißt), wenn das Commitment $\langle \mathcal{S}, \mathcal{R} \rangle_{\mathsf{com}}(m)$ einen Angreifer $\mathcal{A}$ in die Lage versetzt, auch ohne Kenntnis von m diese Hinterlegung in ein Commitment für eine andere, in Beziehung zu m stehende Nachricht m^* abzuändern. Anderenfalls nennt man das Verfahren *non-malleable*. Für Non-Malleability soll das Verändern der Commitments so schwierig sein, dass man gleich mit Hilfe eventueller Seiteninformationen eigenständig aus dem Stand, sprich ad hoc, einen passenden Wert hinterlegen könnte.

Eine einfache Anwendung für non-malleable Commitments sind Auktionen, wo jeder Teilnehmer $\mathcal{S}$ in Form einer Hinterlegung $\langle \mathcal{S}, \mathcal{R} \rangle_{\mathsf{com}}(m)$ ein nicht-öffentliches Gebot m macht. Auch bei perfekter Geheimhaltung ist

nicht auszuschließen, dass ein Mitbieter $\mathcal{A}$ das Commitment $\langle \mathcal{S}, \mathcal{R} \rangle_{\mathsf{com}}(m)$ so verändert, dass dieses dann dem Gebot $m^* := m + 1$ entspricht. Zwar kennt $\mathcal{A}$ zu diesem Zeitpunkt noch nicht m respektive m^*, aber nach Öffnen von $\langle \mathcal{S}, \mathcal{R} \rangle_{\mathsf{com}}(m)$ durch $\mathcal{S}$ gewinnt $\mathcal{A}$ stets bei minimalem Aufwand das Bietverfahren mit $m^* = m + 1$. So scheidet das Commitment-Schema basierend auf der RSA-Repräsentation

$$M := g^m r^e \bmod N$$

aus Satz 1.5.2 aus, denn $M^* := Mg \bmod N$ kann $\mathcal{A}$ später bei Kenntnis von (m, r) mit dem Paar $(m + 1 \bmod e, rg^{\lfloor m+1/e \rfloor})$ aufdecken. Weil die Hinterlegung von m in eine von $a + bm \bmod e$ für $a, b \in \mathbb{Z}$ einfach überführbar ist, spricht man auch von einem *highly malleable* Commitment, also einer in hohem Maße verformbaren Hinterlegung. Ähnliches gilt für das Schema basierend auf der Faktorisierungsrepräsentation aus Satz 3.1.1, es ist ebenso hochgradig verformbar.

Wir formalisieren schrittweise die intuitive Charakterisierung des Non-Malleability-Begriffs. Zur Vereinfachung sei zunächst vorausgesetzt, der Angreifer $\mathcal{A}$ verhalte sich in der Commitmentphase passiv. Die Beziehung zwischen der von $\mathcal{S}$ hinterlegten Nachricht m und derjenigen m^* des Angreifers $\mathcal{A}$ drückt man in Form einer binären Relation R auf der Nachrichtenmenge aus. $\mathcal{A}$ ist erfolgreich, wenn er später sein Commitment M^* durch eine Nachricht m^* mit $(m, m^*) \in \mathrm{R}$ öffnet. Falls $\mathcal{A}$ die Hinterlegung nicht korrekt aufdeckt und im Ergebnis erfolglos ist, sei $m^* := \perp$. Wir haben bisher den trivialen Fall, dass $\mathcal{A}$ die Daten einfach kopiert, d.h. $M^* = M$ und $m^* = m$, nicht beachtet. Diese Form eines Angriffs, das anonyme Kopieren digitaler Daten, ist stets möglich, weshalb diese triviale Attacke per Definition mit der Forderung $(m, m) \notin \mathrm{R}$ für alle Nachrichten m ausgeschlossen wird. Um diese Angriffsart in der Praxis abzuwenden, erweitere zum Beispiel die Nachricht um die Identität des Hinterlegenden, so dass ein Angreifer mit $m = m^*$ nicht erfolgreich ist, weil seine Identität nicht mit der verbrieften übereinstimmt. Fassen wir zusammen: Eine binäre Relation R auf der Nachrichtenmenge erweitert um das Symbol $\perp$ nennen wir *relevant*, wenn:

- Die Relation R ist in Polynomialzeit zu verifizieren.
- Es gilt $(m, \perp) \notin \mathrm{R}$ für alle Nachrichten m.
- Es gilt $(m, m) \notin \mathrm{R}$ für alle Nachrichten m, d.h. R ist irreflexiv.

Der Angreifer $\mathcal{A}$ ist zur relevanten Relation R erfolgreich (bezüglich Aufdeckung), wenn es ihm gelingt, sein Commitment M^* nach der Öffnung von M durch m mit einer Nachricht m^* der Form $(m, m^*) \in \mathrm{R}$ aufzudecken.

Wenngleich das vom Sender $\mathcal{S}$ verbriefte m dem Angreifer $\mathcal{A}$ unbekannt ist, verfügt $\mathcal{A}$ oftmals über Informationen bezüglich des hinterlegten Wertes. So ist im Fall eines Bietverfahrens anzunehmen, dass die Angebote aller konkurrierenden Beteiligten sich in einem wirtschaftlich sinnvollem Rahmen bewegen. Man modelliert diese Seiteninformationen wie folgt [DDN00]:

- $\mathcal{A}$ kennt die Multimenge M der Nachrichten, aus welcher $\mathcal{S}$ eine zufällige Nachricht $m \in_{\mathsf{R}} \mathsf{M}$ hinterlegt oder $\mathcal{A}$ darf sogar M in Abhängigkeit von der Relation R und gegebenenfalls den Public-Parameters vorschlagen.

- $\mathcal{A}$ erhält in Form einer sogenannten *History-Funktion* $\mathrm{Hist}(m)$ Informationen über das gewählte m.

Der Begriff Non-Malleability soll ausdrücken, ein Commitment M sei nicht verformbar: Um eine in Relation stehende Nachricht zu hinterlegen, ist eine Modifikation von M mindestens so schwierig, wie eigenständig nur mit Hilfe von $\mathrm{Hist}(m)$ eine neue Hinterlegung zu erzeugen. Wir haben bisher nur passive Angreifer $\mathcal{A}_{\mathsf{passiv}}$ betrachtet, welche die Commitmentphase nur beobachten. Bei interaktiven Hinterlegungsphasen besteht jedoch auch die Möglicheit eines Person-In-The-Middle-Angriffs (PIM), wo der Angreifer $\mathcal{A}_{\mathsf{Pim}}$ die Rolle eines Intermediärs übernimmt. Während beim passiven $\mathcal{A}_{\mathsf{passiv}}$ die Protokolle $\langle \mathcal{S}, \mathcal{R} \rangle_{\mathsf{com}}(m)$ und $\langle \mathcal{A}_{\mathsf{passiv}}, \mathcal{R} \rangle_{\mathsf{com}}(m^*)$ sequentiell ablaufen, sind beim aktiven Angreifer $\mathcal{A}_{\mathsf{Pim}}$ beiden Ausführungen verschachtelt. Die Form des passiven Angreifers $\mathcal{A}_{\mathsf{passiv}}$ stellt einen Spezialfall einer PIM-Attacke dar, weshalb man den Non-Malleability-Gedanken wie folgt formulieren kann: Existiert ein PIM-Angreifer $\mathcal{A}_{\mathsf{Pim}}$ mit Erfolgswahrscheinlichkeit $\pi(\mathcal{A}_{\mathsf{Pim}}, \mathsf{R}, \mathsf{M})$, dann gibt es einen Angreifer $\mathcal{A}_{\mathsf{Adhoc}}$ aus dem Stand mit mindestens nahezu gleicher Erfolgswahrscheinlichkeit $\pi(\mathcal{A}_{\mathsf{Adhoc}}, \mathsf{R}, \mathsf{M})$.

Definition 3.2.1 (Non-Malleable Commitment-Schema). *Ein Hinterlegungsverfahren heißt non-malleable bezüglich Aufdeckung, wenn zu jedem PIM-Angreifer $\mathcal{A}_{\mathsf{Pim}}$ ein Angreifer $\mathcal{A}_{\mathsf{Adhoc}}$ aus dem Stand existiert, so dass bezüglich aller relevanten Relationen* R *und aller von $\mathcal{A}_{\mathsf{Pim}}$ vorgegebenen Nachrichten-Multimengen* M *für die Erfolgswahrscheinlichkeiten gilt*

$$\pi(\mathcal{A}_{\mathsf{Adhoc}}, \mathsf{R}, \mathsf{M}) \geq \pi(\mathcal{A}_{\mathsf{Pim}}, \mathsf{R}, \mathsf{M}) - \nu$$

für ein im Sicherheitsparameter n vernachlässigbares ν. Je nach Laufzeit von $\mathcal{A}_{\mathsf{Adhoc}}$ unterscheidet man die folgenden Varianten der Non-Malleability:

a) strikte Non-Malleability: $|\mathcal{A}_{\mathsf{Adhoc}}|$ ist strikt polynomiell in n.

b) liberale Non-Malleability: $|\mathcal{A}_{\mathsf{Adhoc}}|$ *ist im Erwartungswert polynomiell in* n*.*

Bei ϵ*-Non-Malleability soll für alle* $\epsilon > 0$*, relevanten Relationen* R *und allen von* $\mathcal{A}_{\mathsf{Pim}}$ *vorgegebenen Nachrichten-Multimengen* M

$$\pi(\mathcal{A}_{\mathsf{Adhoc}}, \mathsf{R}, \mathsf{M}) \geq \pi(\mathcal{A}_{\mathsf{Pim}}, \mathsf{R}, \mathsf{M}) - \epsilon - \nu$$

gelten, wobei $|\mathcal{A}_{\mathsf{Adhoc}}|$ *strikt polynomiell in* n *und* ϵ^{-1} *sei. Im Fall strikter und* ϵ*-Non-Malleability setzen wir voraus, dass die Laufzeit von* $\mathcal{A}_{\mathsf{Pim}}$ *wie die des Adhoc-Angreifers* $\mathcal{A}_{\mathsf{Adhoc}}$ *strikt polynomiell sei.*

Wir schreiben kurz $\pi(\mathcal{A})$ statt $\pi(\mathcal{A}, \mathsf{R}, \mathsf{M})$, wenn Relation R und Nachrichtenmenge M aus dem Kontext hervorgehen. Hinterlegungsverfahren mit strikter Non-Malleability sind bisher nicht bekannt. Die bekannten, auf Knowledge-Extraktoren aufbauenden Verfahren [DDN00, FF00] erreichen liberale als auch ϵ-Non-Malleability. Andere Konstruktionen wie [DIO98, CKOS01] oder die in den Abschnitten 3.4 und 3.5 vorgestellten Schemata basierend auf der Faktorisierungsrepräsentation bieten ϵ-Non-Malleability. Man zieht allgemein die liberale der ϵ-Non-Malleability vor, da $\mathcal{A}_{\mathsf{Adhoc}}$ stets in einen Angreifer mit strikter Polynomialzeit in n und ϵ^{-1} zu transformieren ist. Bei den von uns betrachteten Verfahren mit Polynomialzeit-Bindung sind für ϵ-Non-Malleability als Parameter ϵ nur polynomielle Bruchteile von Bedeutung, weil anderenfalls $\mathcal{A}_{\mathsf{Adhoc}}$ keiner polynomiellen Laufzeitschranke unterliegt und als Konsequenz die Bindung an den hinterlegten Wert gegebenenfalls hinfällig wird.

Die obige Formulierung 3.2.1 der Definition eines non-malleable Commitments unterscheidet sich von der ursprünglichen [DDN00], bei der die Differenz der Erfolgswahrscheinlichkeiten hinreichend nah sein soll. Unsere Charakterisierung des Begriffs erfaßt hingegen auch solche Adhoc-Angreifer $\mathcal{A}_{\mathsf{Adhoc}}$, deren Erfolgswahrscheinlichkeit signifikant über der des PIM-Angreifers $\mathcal{A}_{\mathsf{Pim}}$ liegen. Beide Definitionen sind äquivalent, weil die Erfolgswahrscheinlichkeit eines Adhoc-Angreifer $\mathcal{A}_{\mathsf{Adhoc}}$ gegebenenfalls abgesenkt werden kann. Während Dolev, Dwork und Naor [DDN00] einen Angreifer aus dem Stand mit nahezu gleicher Erfolgswahrscheinlichkeit im Sinn haben (statt eines Adhoc-Angreifers sprechen sie von einem Simulator zu $\mathcal{A}$), folgt unsere Definition der intuitiven Vorstellung, dass jenes, was ein PIM-Angriff gestattet, mindestens bereits aus dem Stand möglich sei.

Für statistisch bindende Commitments existiert nur eine Aufdeckung der Hinterlegung, weshalb der verbriefte Wert unabhängig von der Laufzeit des Senders $\mathcal{S}$ bereits nach der Commitmentphase festliegt, und nur noch offen ist, ob $\mathcal{S}$ anschließend die Aufdeckung gelingt. Für statistisch bindende Schemata gibt es daher eine weitere Auslegung des Non-Malleability-Gedankens

[FF00, F01]: *Non-Malleability bezüglich Hinterlegung* bedeutet, wir sehen einen Angreifer $\mathcal{A}$ bereits dann als erfolgreich an, falls $\mathcal{A}$ ein statistisch bindendes Commitment mit einer zu m in Relation stehenden Nachricht m^* generiert. Diese Forderung ist im Public-Parameter-Modell stärker als Non-Malleability bezüglich Aufdeckung. Unter der Voraussetzung eines statistisch geheimhaltenden Bit-Commitment-Schemas, welches non-malleable bezüglich Aufdecken ist, existiert ein statistisch bindendes Verfahren zur Bit-Hinterlegung, das

a) zwar non-malleable bezüglich Aufdecken aber

b) malleable bezüglich Hinterlegung ist.

Das von Dolev, Dwork und Naor [DDN00] angegebene (statistisch bindende) Commitment-Verfahren erreicht Non-Malleability bezüglich Hinterlegung, allerdings umfaßt das Protokoll zur Hinterlegung eines einzelnen Bits im Sicherheitsparameter logarithmisch viele Runden. Weil sich die Bindungseigenschaft der im weiteren betrachteten Schemata nur auf Polynomialzeit-Algorithmen erstreckt (genauer, durch das Commitment ist die Nachricht m^* nicht eindeutig festgelegt), beziehe sich im folgenden Non-Malleability stets auf Aufdeckung.

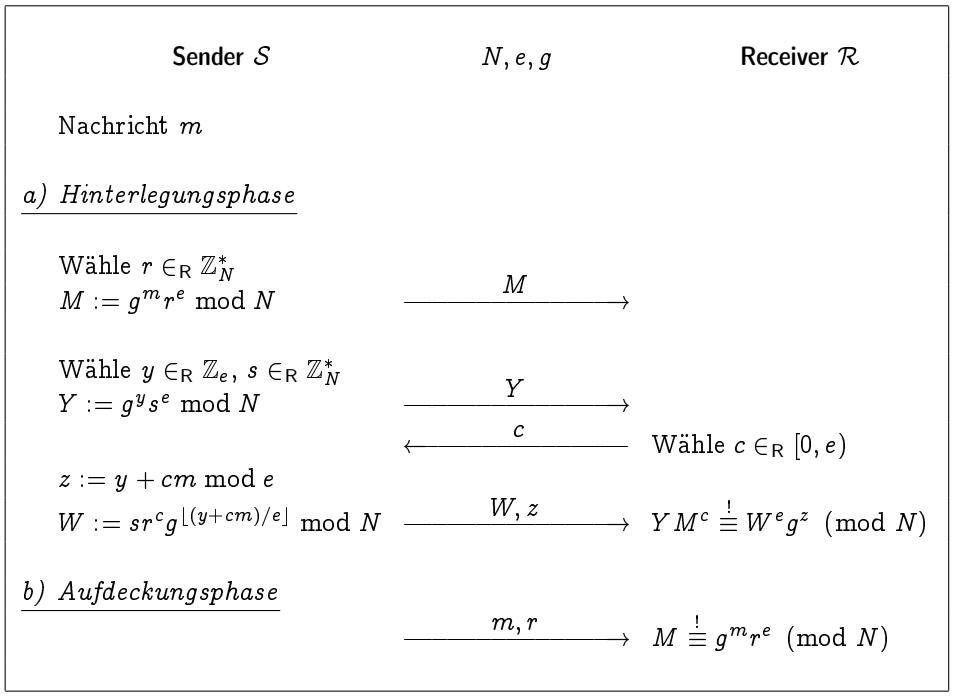
Ein erster Schritt in Richtung eines non-malleable Hinterlegungsverfahrens ist die Erweiterung des eigentlichen Commitments um einen witness-indistinguishable *Proof-Of-Knowledge* (POK) bezüglich der hinterlegten Nachricht. Die Geheimhaltung bleibt gewahrt, der zusätzliche Proof-Of-Knowledge liefert dank Witness-Indistinguishability keine Hinweise auf die hinterlegte Nachricht. Abbildung 3.1 zeigt eine solche Konstruktion basierend auf dem RSA-Repräsentationsproblem und dem Proof-Of-Knowledge von Okamoto aus Kapitel 1.3. Wir konstruieren zum *passiven* Angreifer $\mathcal{A}_{\mathsf{passiv}}$ einen Adhoc-Angreifer $\mathcal{A}_{\mathsf{Adhoc}}$ mit nahezu gleicher Erfolgswahrscheinlichkeit, der auf dem Knowledge-Extraktors des Proof-Of-Knowledge beruht. Unter der Überschrift „Extraction Implies Non-Malleability“ zeigen Dolev, Dwork und Naor [DDN00, Abschnitt 4.4]:

Satz 3.2.2 (Dolev, Dwork, Naor). *Sei $\mathcal{E}$ ein Knowledge-Extraktor für die hinterlegte Nachricht des Commitments. Dann existiert zu allen passiven Angreifern $\mathcal{A}_{\mathsf{passiv}}$ ein Angreifer $\mathcal{A}_{\mathsf{Adhoc}}$ aus dem Stand mit Laufzeit $|\mathcal{E}|$, so dass für alle relevanten Relationen R und alle von $\mathcal{A}_{\mathsf{passiv}}$ vorgegebenen Nachrichten-Multimengen M*

$$\pi(\mathcal{A}_{\mathsf{Adhoc}}, \mathsf{R}, \mathsf{M}) \geq \pi(\mathcal{A}_{\mathsf{passiv}}, \mathsf{R}, \mathsf{M}) - \nu$$

mit einem bezüglich Sicherheitsparameter n vernachlässigbaren ν gilt.

Abbildung 3.1: RSA-Commitment-Schema mit Proof-Of-Knowledge



Beweis. Wir fassen den Beweis [DDN00] in eigenen Worten zusammen. Wendet man auf $\mathcal{A}_{\mathsf{passiv}}$ den Knowledge-Extraktor $\mathcal{E}$ an, steht die extrahierte Nachricht m^* in Beziehung zu m mit Wahrscheinlichkeit

$$\pi(\mathcal{E}) \geq \pi(\mathcal{A}_{\mathsf{passiv}}) - \epsilon \tag{3.1}$$

für ein vernachlässigbares ϵ. Denn eine gelungene Hinterlegungsphase ist Voraussetzung für eine spätere, erfolgreiche Aufdeckung von $\mathcal{A}_{\mathsf{passiv}}$ und die Wahrscheinlichkeit, sein Commitment später mit einer anderen Nachricht zu öffnen, ist vernachlässigbar (anderenfalls erhielte man im Widerspruch zur Bindung zwei verschiedene Aufdeckungen eines Commitments).

Wir definieren mit $\mathcal{A}_{\mathsf{Hybrid}}$ eine Mischform zwischen passivem und Adhoc-Angriff, der zwar die History $\mathrm{Hist}(m)$, aber statt des Commitments von m die Hinterlegung eines von uns gewählten $m_{\mathsf{fake}} \in_R \mathsf{M}$ erhält. $\mathcal{A}_{\mathsf{Hybrid}}$ wird suggeriert, bei der vorliegenden Hinterlegung handele sich um diejenige zu $\mathrm{Hist}(m)$, wenngleich das Commitment gefälscht ist (engl. to fake). Der Adhoc-Angreifer $\mathcal{A}_{\mathsf{Adhoc}}$ geht wie folgt vor: Er bestimmt eine zufällige Nachricht $m_{\mathsf{fake}} \in_R \mathsf{M}$ und ruft $\mathcal{A}_{\mathsf{Hybrid}}$ mit der ihm gegebenen, echten Seiteninformationen $\mathrm{Hist}(m)$ sowie dem Commitment zu m_{fake} auf, um an-

schließend die extrahierte Nachricht zu hinterlegen. Daher gilt:

$$\pi(\mathcal{A}_{\mathsf{Adhoc}}) = \pi(\mathcal{A}_{\mathsf{Hybrid}}). \tag{3.2}$$

Zur Analyse der Erfolgswahrscheinlichkeit $\pi(\mathcal{A}_{\mathsf{Hybrid}})$ betrachte nachstehendes Gedankenexperiment:

> Wähle zwei zufällige Nachrichten $m_1, m_2 \in_R M$. Wir erhalten das Commitment zur Nachricht m_i für ein $i \in_R \{1, 2\}$ und sollen i bestimmen.

Aufgrund der Geheimhaltung ist diese Aufgabe effizient höchstens geringfügig besser als Raten, formal mit Wahrscheinlichkeit $\frac{1}{2} + \xi$ für ein vernachlässigbares $|\xi|$, zu lösen. Betrachte folgenden Unterscheider (Distinguisher) $\mathcal{D}$:

1. Lege in einer Simulation $\mathcal{A}_{\mathsf{passiv}}$ das Commitment zu m_i und $\mathrm{Hist}(m_1)$ vor. Sei m^* die extrahierte Nachricht seiner Hinterlegung. Sollte der Extraktor keinen Wert bestimmen, setze $m^* := \perp$.
2. Falls $(m_1, m^*) \in R$, gib „$i = 1$“ aus, anderenfalls „$i = 2$“.

Unter der Annahme $i = 1$ bestimmt $\mathcal{D}$ genau dann den richtigen Wert, sofern m^* in Relation zu m_i steht, also mit Wahrscheinlichkeit $\pi(\mathcal{E})$. Gilt hingegen $i = 2$, liefert der Unterscheider $\mathcal{D}$ mit Wahrscheinlichkeit $\pi(\mathcal{A}_{\mathsf{Hybrid}})$ eine falsche Ausgabe. Im Ergebnis ermittelt $\mathcal{D}$ das korrekte i mit Wahrscheinlichkeit

$$\tfrac{1}{2}\pi(\mathcal{E}) + \tfrac{1}{2}(1 - \pi(\mathcal{A}_{\mathsf{Hybrid}})) = \tfrac{1}{2} + \underbrace{\tfrac{1}{2}(\pi(\mathcal{E}) - \pi(\mathcal{A}_{\mathsf{Hybrid}}))}_{:=\xi}.$$

Wegen der Geheimhaltung ist $|\xi|$ kleiner als jeder polynomielle Bruchteil, so dass ein vernachlässigbares μ mit $\pi(\mathcal{A}_{\mathsf{Hybrid}}) \geq \pi(\mathcal{E}) - \mu$ existiert. Nach Gleichung (3.2) und Abschätzung (3.1) gilt

$$\pi(\mathcal{A}_{\mathsf{Adhoc}}) = \pi(\mathcal{A}_{\mathsf{Hybrid}}) \geq \pi(\mathcal{E}) - \mu \geq \pi(\mathcal{A}_{\mathsf{passiv}}) - \epsilon - \mu. \tag{3.3}$$

Weil mit den beiden Summanden auch $\nu := \epsilon + \mu$ vernachlässigbar ist, folgt die Behauptung. □

Offenbar genügt für die Aussage von Satz 3.2.2, wenn der Extraktor $\mathcal{E}$ statt des von $\mathcal{A}$ hinterlegten m^* auch eventuell eine äquivalente Nachricht m' findet, wobei gleichwertig bedeutet, dass aus $(m, m') \in R$ nahezu immer $(m, m^*) \in R$ folgt.

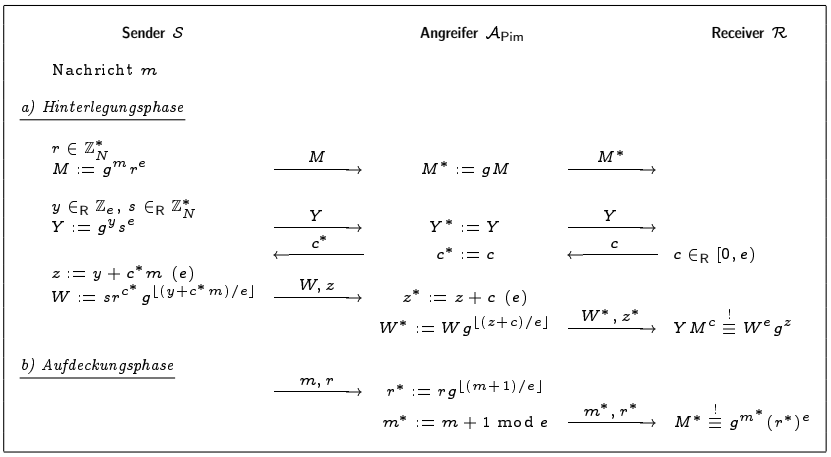
Wir übertragen Satz 3.2.2 auf ϵ-Extraktoren. Abschätzung (3.3) des Beweises gilt auch für nicht vernachlässigbares ϵ, mit $\nu := \mu$ folgt:

Korollar 3.2.3. *Sei $\mathcal{E}$ ein ϵ-Knowledge-Extraktor für die hinterlegte Nachricht des Commitments. Dann existiert zu allen passiven Angreifern $\mathcal{A}_{\mathsf{passiv}}$ ein Angreifer $\mathcal{A}_{\mathsf{Adhoc}}$ aus dem Stand mit Laufzeit $|\mathcal{E}|$, so dass für alle relevanten Relationen* R *und alle von $\mathcal{A}_{\mathsf{passiv}}$ vorgegebenen Nachrichten-Multimengen* M

$$\pi(\mathcal{A}_{\mathsf{Adhoc}}, \mathsf{R}, \mathsf{M}) \geq \pi(\mathcal{A}_{\mathsf{passiv}}, \mathsf{R}, \mathsf{M}) - \epsilon - \nu$$

mit einem bezüglich Sicherheitsparameter n vernachlässigbaren ν gilt.

Abbildung 3.2: PIM-Angriff auf RSA-Commitment mit POK



Die Konstruktion von $\mathcal{A}_{\mathsf{Adhoc}}$ aus Satz 3.2.2 bzw. Korollar 3.2.3 gilt zwar prinzipiell auch für PIM-Angreifer $\mathcal{A}_{\mathsf{Pim}}$, die unmittelbare Anwendung scheitert aber an der Voraussetzung des Knowledge-Extraktors, also der Möglichkeit, die von $\mathcal{A}_{\mathsf{Pim}}$ verbriefte Nachricht bereits in der Hinterlegungsphase zu ermitteln. Einen erfolgreichen PIM-Angriff auf das Schema 3.1 bestehend aus einem Commitment und anschließendem Proof-Of-Knowledge zeigt Abbildung 3.2 (soweit nicht anders angegeben seien alle Kongruenzen modulo N). Der Intermediär $\mathcal{A}_{\mathsf{Pim}}$ wählt $m^* := m + 1 \bmod e$, reicht die Challenge des Proof-Of-Knowledge an den Sender $\mathcal{S}$ weiter und paßt dessen Antworten

$$\begin{aligned} z^* &:= z + c \bmod e \\ W^* &:= W \cdot g^{\lfloor (z+c)/e \rfloor} \bmod N \end{aligned}$$

entsprechend seines Commitments $M^* \equiv gM \pmod N$ an, denn

$$Y(M^*)^{c^*} \equiv Y M^c g^c \equiv W^e g^z g^c \equiv W^e g^{z+c} \equiv (W^*)^e g^{z^*} \pmod N.$$

Weil der Intermediär $\mathcal{A}_{\mathsf{Pim}}$ einfach die Response unabhängig von m^* modifiziert (d.h. der Proof-Of-Knowledge ist malleable), fehlt ihm — informell ausgedrückt — das Wissen über m, weshalb schon sprachlich die Erfolgsaussichten des Knowledge-Extraktors fragwürdig erscheinen. Und in der Tat tritt beim Zurücksetzen des Angreifers innerhalb der Simulation durch den Knowledge-Extraktor ein Problem auf: Man benötigt für die weiteren Challenges jeweils die Antworten des Senders $\mathcal{S}$, die allerdings uns nicht zur Verfügung stehen. Um dieses Hindernis zu umgehen, ist folgende Idee naheliegend:

a) Wir wählen eine „gefälschte“ Nachricht $m_{\mathsf{fake}} \in_{\mathsf{R}} \mathsf{M}$ und geben ihr Commitment M_{fake} als dasjenige zu den vorliegenden Seiteninformationen $\mathrm{Hist}(m)$ der uns unbekannten Nachricht m aus. Wegen der perfekten Geheimhaltung des RSA-Repräsentations-Commitments sind M_{fake} und $\mathrm{Hist}(m)$ nicht widersprüchlich.

b) Die Kenntnis einer Repräsentation von M_{fake} gestattet uns, auch nach Zurücksetzen korrekte Antworten des simulierten Senders $\mathcal{S}$ zu geben, aufgrund der perfekten Witness-Indistinguishability ist die Verteilung der kommunizierten Daten identisch.

Zwar besteht nicht die Gefahr einer Inkonsistenz zwischen Protokollausführung und Seiteninformationen, aber, wie zuvor gesehen, kann $\mathcal{A}_{\mathsf{Pim}}$ durch einfaches Anpassen der Response auch ohne Wissen der Nachricht m bzw. m^* die Hinterlegungsphase erfolgreich abschließen. Der Knowledge-Extraktor liefert in diesem Fall keine in Relation zu m, sondern eine in Beziehung zu m_{fake} stehende Nachricht, im Beispiel aus Abbildung 3.2 nämlich $m^* \equiv m_{\mathsf{fake}} + 1 \pmod{e}$. Die Folgerung aus diesen Überlegungen lautet umgangssprachlich, dass wir in der Simulation den Angreifer $\mathcal{A}_{\mathsf{Pim}}$ mit weniger Informationen versorgen dürfen, als wir von ihm erhalten. Beispielsweise soll beim Zurücksetzen zwischen $\mathcal{S}$ und $\mathcal{A}_{\mathsf{Pim}}$ jeweils die gleiche Challenge gewählt werden, während auf der anderen Seiten $\mathcal{A}_{\mathsf{Pim}}$ verschiedene Challenges erhält, so dass dieser „ohne Hilfe des Senders $\mathcal{S}$“ die passende Response geben muß, mit anderen Worten, ad hoc aus dem Stand. Dieser Idee folgend zeigen wir in Abschnitt 3.3, wie man basierend auf der RSA-Repräsentation und zugehörigem Proof-Of-Knowledge mit Hilfe eines Coin-Flipping-Protokolls ein interaktives, non-malleable Commitment-Schema konstruiert.

3.3 Commitment basierend auf RSA

Unser Ziel ist die Konstruktion eines non-malleable Commitments basierend auf der RSA-Repräsentation. Das Beispiel in Abbildung 3.1 zeigte, dass die

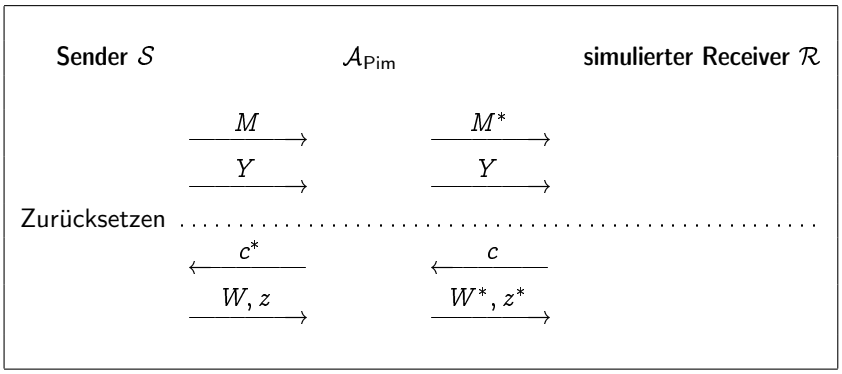
einfache Erweiterung um einen Proof-Of-Knowledge keine Non-Malleability bei Person-In-The-Middle-Angriffen $\mathcal{A}_{\mathsf{Pim}}$ bietet. In der Simulation für den Non-Malleability-Nachweis übernehmen wir wie im Beweis zu Satz 3.2.2 auf Seite 54 die Rolle des Receivers $\mathcal{R}$, der von $\mathcal{S}$ ein Commitment für ein unbekanntes m erhält. Wir versuchen, durch den simulierten Person-In-The-Middle-Angriff über den Knowledge-Extraktor vorab (d.h. vor Aufdeckung der Commitments) Informationen über die von $\mathcal{A}_{\mathsf{Pim}}$ hinterlegte, in Relation zu m stehende Nachricht m^* zu erhalten, um unsererseits durch Verbriefen von m^* einen erfolgreichen Angriff aus dem Stand zu erreichen. Betrachte die Simulation anhand Abbildung 3.3:

1. Der simulierte $\mathcal{A}_{\mathsf{Pim}}$ erhält das Commitment M von $\mathcal{S}$ und hinterlegt seinerseits einen Wert in Form von M^*.

2. Wir richten eine Challenge c_1 an $\mathcal{A}_{\mathsf{Pim}}$, seine Frage c_1^* reicht der Simulator an $\mathcal{S}$ weiter, gibt dessen Antwort (W_1, z_1) zurück an $\mathcal{A}_{\mathsf{Pim}}$ und speichert die Response (W_1^*, z_1^*) des Intermediärs auf die Challenge c_1.

3. Der Simulator setzt $\mathcal{A}_{\mathsf{Pim}}$ an die Stelle zurück, an welcher der Intermediär eine neue Challenge c_2 erhält. Weil wir allerdings keine Kontrolle über den Sender $\mathcal{S}$ haben, ist es nicht möglich, $\mathcal{S}$ ebenfalls zu reseten respektive die Challenge c_2^* von $\mathcal{A}_{\mathsf{Pim}}$ an $\mathcal{S}$ weiterzuleiten. Der Sender $\mathcal{S}$ hat — wie in der Protokollspezifikation festgelegt — die eine Challenge, nämlich c_1^*, bereits beantwortet.

Abbildung 3.3: Zurücksetzen bei Commitment-&-Proof-Of-Knowledge



Um bei der Extraktion dieses Hindernis zu umgehen, modifiziere die Bestimmung der Challenge c innerhalb des Protokolls. Die Wahl seitens $\mathcal{R}$ garantiert, dass, einen ehrlichen Empfänger vorausgesetzt, die Challenge uniform verteilt ist. Diese für die Analyse des Proof-Of-Knowledge verwandte

Charakteristik bleibt erhalten, wenn die Challenge durch ein Coin-Flipping-Protokoll (vergleiche Abschnitt 1.5) zwischen $\mathcal{S}$ und $\mathcal{R}$ bestimmt wird:

- $\mathcal{S}$ hinterlegt $a \in_R \mathbb{Z}_e$.
- $\mathcal{R}$ publiziert $b \in_R \mathbb{Z}_e$
- $\mathcal{S}$ öffnet sein Commitment und beide Parteien wählen $c := a + b \bmod e$ als Challenge[1] für den Proof-Of-Knowledge.

Für die Simulation plazieren wir eine Trapdoor in das im Coin-Flipping-Protokoll verwandte Commitment, um dann in der Simulation die Hinterlegung von $\mathcal{S}$ mit der Differenz $c^* - b$ modulo e zu öffnen. Auf diese Art und Weise sind wir in der Lage, beim Zurücksetzen in Abbildung 3.3 zu erreichen, dass $\mathcal{S}$ und $\mathcal{A}_{\mathsf{Pim}}$ sich stets auf die gleiche Challenge c^* einigen. Dies ermöglicht, die von $\mathcal{S}$ bei der ersten Ausführung gegebene Antwort zu wiederholen, während durch das Zurücksetzen der Angreifer $\mathcal{A}_{\mathsf{Pim}}$ die Daten der vorherigen Wiederholung verliert und folglich nicht bemerkt, dass stets die gleiche Challenge generiert wird.

Eine universelle Trapdoor für das Coin-Flipping-Protokoll stellt uns vor ein neues Problem. Einerseits soll der Simulator in der Lage sein, das Coin-Flipping-Commitment dank einer Geheimtüre mehrdeutig zu öffnen, während auf der andere Seite dem Angreifer $\mathcal{A}_{\mathsf{Pim}}$ genau dies verwehrt sein muß, damit wir dessen Nachricht extrahieren können. Unser Ziel ist daher eine individuelle, auf den Sender $\mathcal{S}$ zugeschnittene Geheimtüre. Wir haben bisher noch nicht verwandt, dass, um überhaupt erfolgreich zu sein, der Angreifer $\mathcal{A}_{\mathsf{Pim}}$ per Definition eine andere Nachricht m^* als diejenige des Senders $\mathcal{S}$ wählen muß. Diesen Unterschied nutzen wir für eine personen- oder genauer nachrichtenbezogene Trapdoor und bestimmen dazu die Parameter des Coin-Flipping-Commitments abhängig von der hinterlegten Nachricht.

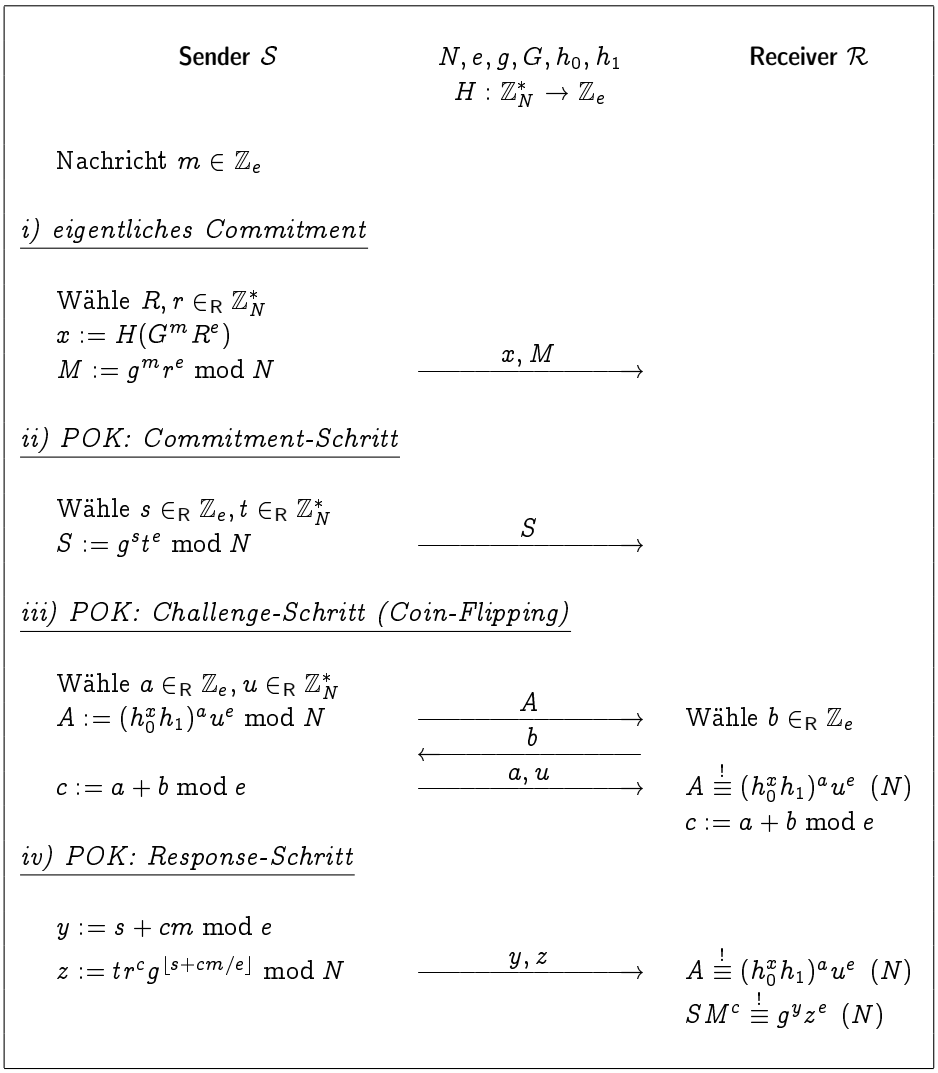
Für das non-malleable Schema gibt die Dritte Partei $\mathcal{T}$ neben $(N, e, g) \leftarrow \mathsf{RSAIndex}(1^n)$ als Public-Parameters zusätzlich $G, h_0, h_1 \in_R \mathbb{Z}_N^*$ vor. Im ersten Schritt hinterlegt der Sender $\mathcal{S}$ die eigentliche Nachricht $m \in \mathbb{Z}_e$ in Form eines RSA-Repräsentations-Commitments bezüglich (N, e, G) in Verbindung mit einer kollisions-resistenten Hashfunktion $H : \mathbb{Z}_N^* \to \mathbb{Z}_e$, welche die Hinterlegung nach $\mathbb{Z}_e$ abbildet:

$$x := H(G^m R^e).$$

Wir gehen auf die Bedeutung von $G^m R^e$ später ein, entscheidend ist zu diesem Zeitpunkt, dass unter der RSA-Annahme für unterschiedliche Nachrichten m^* und m auch bis auf vernachlässigbare Wahrscheinlichkeit die

[1] Wir behalten den Begriff der Challenge bei, wenngleich es eigentlich keine von $\mathcal{R}$ an $\mathcal{S}$ gerichtete Frage ist, sondern sich beide Parteien auf diese einigen.

Abbildung 3.4: Hinterlegungsphase bei non-malleable RSA-Schema



beiden Hashwerte x^* und x nicht identisch sind, denn sonst findet man eine Kollision oder zwei RSA-Darstellungen des gleichen Wertes. Um $a \in \mathbb{Z}_e$ für das Coin-Flipping zu hinterlegen, wählt $\mathcal{S}$ ein $u \in_R \mathbb{Z}_N^*$ und setzt:

$$A := (h_0^x h_1)^a u^e \bmod N.$$

Im Gegensatz zur üblichen Hinterlegung hängt hier die Basis $h_0^x h_1$ vom Hashwert x und letztlich von der Nachricht m ab. Für die Simulation, um aus $\mathcal{A}_{\mathsf{Pim}}$ die hinterlegte Nachricht zu extrahieren, geben wir $h_1 := h_0^{-x} w^e \bmod N$ für ein zufälliges $w \in_R \mathbb{Z}_N^*$ vor. Die Trapdoor w, d.h. die e-ten

Abbildung 3.5: Non-malleable RSA-Commitment-Schema

Sender $\mathcal{S}$	N, e, g, G, h_0, h_1 $H : \mathbb{Z}_N^* \to \mathbb{Z}_e$	Receiver $\mathcal{R}$
Nachricht $m \in \mathbb{Z}_e$		
a) Hinterlegungsphase		
Wähle $R \in_R \mathbb{Z}_N^*$ $x := H(G^m R^e)$		
Wähle $a, s \in_R \mathbb{Z}_e$ Wähle $r, t, u \in_R \mathbb{Z}_N^*$ $M := g^m r^e \bmod N$ $A := (h_0^x h_1)^a u^e \bmod N$ $S := g^s t^e \bmod N$	$\xrightarrow{x, M, A, S}$ $\xleftarrow{b}$	Wähle $b \in_R \mathbb{Z}_e$
$c := a + b \bmod e$ $y := s + cm \bmod e$ $z := t r^c g^{\lfloor s+cm/e \rfloor} \bmod N$	$\xrightarrow{a, u, y, z}$	$c := a + b \bmod e$ $A \overset{!}{\equiv} (h_0^x h_1)^a u^e \ (N)$ $S M^c \overset{!}{\equiv} g^y z^e \ (N)$
b) Aufdeckungsphase		
	$\xrightarrow{m, r, R}$	$M \overset{!}{\equiv} g^m r^e \ (N)$ $x \overset{!}{=} H(G^m R^e)$

Wurzel von $h_0^x h_1$ modulo N, erlaubt dem Simulator das Commitment A beliebig zu öffnen. Dem Angreifer $\mathcal{A}_{\mathsf{Pim}}$ ist hingegen unter der RSA-Annahme diese Möglichkeit verwehrt, denn unterstellt $x^* \neq x$ erhalten wir über zwei verschiedene Aufdeckungen von A^* eine e-te Wurzel w^* von $h_0^{x^*} h_1$ modulo N. Wegen $h_1 \equiv h_0^{-x} w^e \pmod N$ gilt mit $\Delta x := x^* - x \not\equiv 0 \pmod e$

$$(w^* w^{-1})^e \equiv h_0^{x^*} h_1 w^{-e} \equiv h_0^{x^*} h_0^{-x} w^e w^{-e} \equiv h_0^{\Delta x} \pmod N,$$

und wie in Korollar 1.2.5 auf Seite 9 festgehalten, gestatteten Δx und $w^* w^{-1}$ die effiziente Berechnung der e-ten Wurzel von h_0 modulo N im Widerspruch zur RSA-Annahme.

Wir übernehmen das zuvor beschriebene Commitment A für das Coin-Flipping, über welches die Challenge im Proof-Of-Knowledge generiert wird.

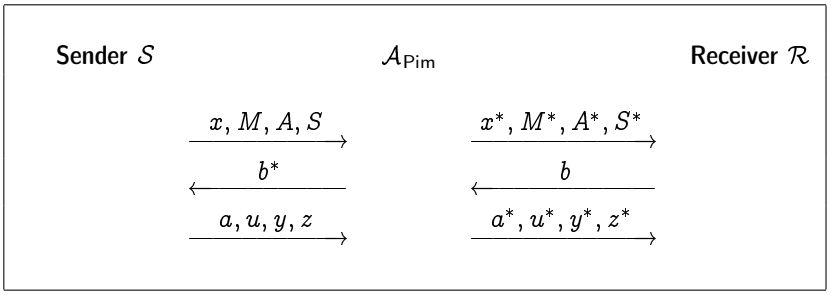
Die gesamte Hinterlegungsphase, wie in Abbildung 3.4 auf Seite 61 dargestellt, umfaßt das eigentliche Commitment, das Coin-Flipping und den Proof-Of-Knowledge. Das Coin-Flipping ist integraler Bestandteil des Proof-Of-Knowledge, denn wie in Kapitel 1.3 erläutert, muß der Commitment-Schritt des Proof-Of-Knowledge vor Vereinbarung der Challenge erfolgen. Die einzelnen Schritte können teilweise zusammengefaßt werden, Abbildung 3.5 zeigt das kompakte non-malleable RSA-Commitment-Schema.

Satz 3.3.1. *Die von $\mathcal{A}_{\mathsf{Pim}}$ vorgegebene Nachrichtenmenge* M *aus $\mathbb{Z}_e$ hänge nur vom öffentlichen Parameter e ab. Dann ist unter der RSA-Annahme und der Voraussetzung einer kollisions-resistenten Hashfunktion das interaktive Hinterlegungsverfahren in Abbildung 3.5 ein polynomialzeit-bindendes, bezüglich Aufdeckung non-malleable Commitment-Schema mit perfekter Geheimhaltung.*

Durch die Restriktion der Wahl von M können wir uns auf den Kern des Non-Malleability-Beweises fokussieren. Um die Einschränkung aufzuheben, modifizieren wir anschließend den Beweis (nicht das Schema) und erhalten die Non-Malleability als Korollar 3.3.2.

Beweis. Der Beweis folgt [FF00, F01]. Die perfekte Geheimhaltung ist die Konsequenz aus der perfekten Geheimhaltung des grundlegenden RSA-Repräsentations-Commitments (Satz 1.5.2 auf Seite 29) und der perfekten Witness-Indistinguishability des Proof-Of-Knowledge (Lemma 1.3.6 auf Seite 18). Die Polynomialzeit-Bindung folgt ebenfalls aus Satz 1.5.2. Nach Satz 3.2.2 von Seite 54 impliziert die Existenz eines Extraktors $\mathcal{E}$, der zu einem Commitment von m die von $\mathcal{A}_{\mathsf{Pim}}$ hinterlegte Nachricht m^* bestimmt, Non-Malleability. Offenbar genügt es, wenn der Extraktor $\mathcal{E}$ eine äquivalente Nachricht m' findet, also aus $(m, m') \in \mathsf{R}$ nahezu immer $(m, m^*) \in \mathsf{R}$ folgt.

Abbildung 3.6: PIM-Angriff auf non-malleable RSA-Commitment



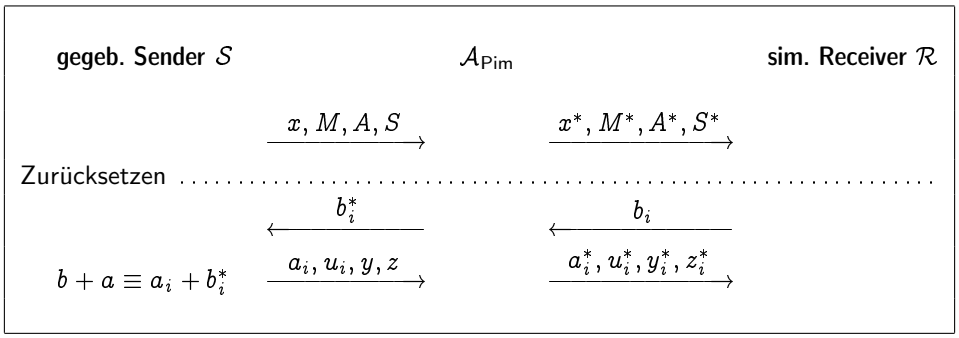
Als Eingabe erhalten wir analog zum Beweis von Satz 3.2.2 auf Seite 54 neben den Public-Parameters zu einer uns unbekannten Nachricht

$m \in_R M$ die Commitmentphase $(x, M, A, S, b, a, u, y, z)$ samt Seiteninformationen Hist(m). Unser Ziel ist, über einen Extraktor $\mathcal{E}$ die von $\mathcal{A}_{\mathsf{Pim}}$ hinterlegte Nachricht m^* (oder ein äquivalentes m') zu extrahieren. Für die Anwendung des Extraktors $\mathcal{E}$ auf $\mathcal{A}_{\mathsf{Pim}}$ bauen wir eine Geheimtüre in einen Public-Parameter ein:

$$h_1 := h_0^{-x} w^e \bmod N.$$

Die übrigen öffentlichen Werte, also (N, e, g, G, h_0), bleiben unberührt. Wegen $h_0^x h_1 \equiv w^e \pmod N$ sind wir im Stande, für diese Public-Parameters dank Geheimtüre das gegebene Coin-Flipping-Commitment A beliebig zu öffnen.

Abbildung 3.7: Simulation $\mathcal{A}_{\mathsf{Pim}}$ beim non-malleable RSA-Commitment



Wir setzen o.B.d.A. die in Abbildung 3.6 auf Seite 63 angegebene Reihenfolge der kommunizierten Daten voraus. Denn die Ordnung der zwischen $\mathcal{A}_{\mathsf{Pim}}$ und $\mathcal{R}$ ausgetauschten Nachrichten ist durch die Spezifikation vorgegeben, die Abfolge der Kommunikation zwischen $\mathcal{S}$ und $\mathcal{A}_{\mathsf{Pim}}$ garantiert für den Angreifer bestmögliche Datenlage unter Einhaltung der definierten Nachrichtenfolge. In der Simulation wird $\mathcal{A}_{\mathsf{Pim}}$, wie in Abbildung 3.7 skizziert, an die Stelle vor Senden des Anteils b zum Coin-Flipping zurückgesetzt. Analog zum Knowledge-Extraktor aus Satz 1.3.1 auf Seite 12 durchlaufen wir in der Simulation eine Hinterlegungsphase. Akzeptiert der simulierte Receiver $\mathcal{R}$ nicht die Hinterlegung von $\mathcal{A}_{\mathsf{Pim}}$ oder gilt $x = x^*$, stoppe. Anderenfalls setze die Ausführung zurück und wiederhole die Challenge-Response-Schritte, bis $\mathcal{A}_{\mathsf{Pim}}$ erneut erfolgreich ist oder für das Coin-Flipping A^* abweichend von der bisherigen Öffnung aufdeckt. Durch die Geheimtüre ist der Simulator in der Lage, zu b_i^* das Coin-Flipping-Commitment A mit dem entsprechenden a_i aufzudecken, so dass $c_i := a_i + b_i^* \bmod e$ modulo e stets gleich der Challenge $a + b$ der uns gegebenen Hinterlegung ist. Im Fall des ersten Abbruch-Kriteriums berechne aus zwei akzeptierten Antworten auf

verschiedene Challenges eine Nachricht m'. Für die Analyse dieses Extraktors $\mathcal{E}$ zeigen wir, dass unter der Voraussetzung der RSA-Annahme sowie der Kollisions-Resistenz von $H(\cdot)$ gilt:

a) Die Wahrscheinlichkeit für einen erfolgreichen $\mathcal{A}_{\mathsf{Pim}}$ mit $x = x^*$ ist vernachlässigbar.

b) Die Wahrscheinlichkeit für einen erfolgreichen $\mathcal{A}_{\mathsf{Pim}}$, der beim Zurücksetzen dann eine alternative Öffnung von A^* gibt, ist vernachlässigbar.

c) Der Extraktor $\mathcal{E}$ liefert mit Wahrscheinlichkeit mindestens $\pi(\mathcal{A}_{\mathsf{Pim}}) - \frac{1}{e}$ eine Aufdeckung des Commitments M^* in Form einer Nachricht m'.

d) Die extrahierte Nachricht m' steht bis auf vernachlässigbare Wahrscheinlichkeit in Relation R zu m.

Falls Behauptung a) nicht zuträfe, simuliere für die von der Dritten Partei uns vorgegebenen Public-Parameters einen regulären Angriff von $\mathcal{A}_{\mathsf{Pim}}$ inklusive Aufdeckung, wenn $\mathcal{S}$ eine von uns gewählte Nachricht $m \in_{\mathsf{R}} \mathsf{M}$ hinterlegt. Sofern $\mathcal{A}_{\mathsf{Pim}}$ mit $x = x^*$ erfolgreich ist, erhalten wir zusätzlich zu (m, R) in der Aufdeckungsphase (m^*, R^*) mit $h(G^m R^e) = h(G^{m^*}(R^*)^e)$. Entweder liefert dies eine Kollision der Hashfunktion oder im Fall $G^m R^e \equiv G^{m^*}(R^*)^e \pmod N$ zwei verschiedene RSA-Repräsentationen und als Konsequenz entgegen der RSA-Annahme die e-te Wurzel von G modulo N.

Träfe Behauptung b) nicht zu, wäre im Widerspruch zur Annahme das RSA-Problem zu $(N, e, h_0) \leftarrow \mathsf{RSAIndex}(1^n)$ mit nicht vernachlässigbarer Wahrscheinlichkeit effizient lösbar. Simuliere einen regulären Angriff von $\mathcal{A}_{\mathsf{Pim}}$ inklusive Aufdeckung, wenn $\mathcal{S}$ eine von uns gewählte Nachricht $m \in_{\mathsf{R}} \mathsf{M}$ hinterlegt, allerdings mit dem modifiziertem Public-Parameter $h_1 := h_0^{-x} w^e \bmod N$ für $w \in_{\mathsf{R}} \mathbb{Z}_N^*$ und $x := H(G^m R^e)$. Falls $\mathcal{A}_{\mathsf{Pim}}$ nicht erfolgreich bezüglich Aufdeckung ist oder $x = x^*$, dann stoppe. Anderenfalls, was gemäß Punkt a) mit Wahrscheinlichkeit nahezu $\pi(\mathcal{A}_{\mathsf{Pim}})$ eintritt, setze $\mathcal{A}_{\mathsf{Pim}}$ an die Stelle, an welcher er auf b wartet, zurück und wiederhole die Ausführung, bis wir neben a_1^* in einer erfolgreichen Commitmentphase von $\mathcal{A}_{\mathsf{Pim}}$ eine zweite Aufdeckung a_2^* von A^* erhalten. Dabei generieren wir dank Geheimtüre jeweils die gleiche Challenge, so dass insgesamt Vorgehen und Umfeld mit dem des Extraktors $\mathcal{E}$ bei $\mathcal{A}_{\mathsf{Pim}}$ übereinstimmen und gleiche Wahrscheinlichkeiten vorliegen. Es gilt

$$(h_0^{x^*} h_1)^{a_1^*}(u_1^*)^e \equiv A^* \equiv (h_0^{x^*} h_1)^{a_2^*}(u_2^*)^e \pmod N$$

und Einsetzen von $h_1 \equiv h_0^{-x} w^e \pmod N$ liefert:

$$h_0^{(x^*-x)a_1^*}(w^{a_1^*} u_1^*)^e \equiv A^* \equiv h_0^{(x^*-x)a_2^*}(w^{a_2^*} u_2^*)^e \pmod N.$$

Mit $\Delta x := x^* - x$ und $\Delta a^* = a_1^* - a_2^*$ können wir dies schreiben als:

$$h_0^{\Delta x \cdot \Delta a^*} \equiv (w^{-\Delta a^*}(u_1^*)^{-1}u_2^*)^e \pmod{N}. \tag{3.4}$$

Aufgrund der Annahme $x \neq x^*$ ist $\Delta x \not\equiv 0 \pmod{e}$ und wegen $a_1^* \not\equiv a_2^* \pmod{e}$ gilt, da e prim ist:

$$\Delta x \cdot \Delta a^* \not\equiv 0 \pmod{e}. \tag{3.5}$$

Nach Korollar 1.2.5 von Seite 9 können wir aus Gleichung (3.4) die e-te Wurzel von h_0 effizient bestimmen.

Für die Aussage c) betrachte den Extraktionsprozeß unter der Bedingung, dass ein erfolgreicher $\mathcal{A}_{\mathsf{Pim}}$ beim Zurücksetzen jeweils die gleiche Öffnung von A^* gibt, also $a_1^* = a_2^* = \ldots$ gilt. Nach Behauptung b) ist diese Annahme nahezu immer erfüllt. Weil $b_1, b_2, \ldots$ zufällig und unabhängig vom simulierten $\mathcal{R}$ gewählt werden, sind die generierten Challenges $a_i^* + b_i \bmod e$ im Proof-Of-Knowledge zwischen $\mathcal{A}_{\mathsf{Pim}}$ und $\mathcal{R}$ ebenso unabhängig und uniform verteilt. Daher können wir den in Satz 1.3.1 auf Seite 12 angeführten Knowledge-Extraktor $\mathcal{E}$ aus Okamotos Identifikationsschema anwenden und erhalten eine Aufdeckung von M^* in Form einer Nachricht m'. Für die Analyse der Erfolgswahrscheinlichkeit des Extraktors $\mathcal{E}$ beachte neben Aussage b), dass $\mathcal{A}_{\mathsf{Pim}}$ die Hinterlegungsphase mindestens mit der Erfolgswahrscheinlichkeit $\pi(\mathcal{A}_{\mathsf{Pim}})$ besteht, denn eine gelungene Commitmentphase ist Voraussetzung für eine erfolgreiche Aufdeckung.

Der Nachweis von Behauptung d) entspricht dem Vorgehen bei Aussage b), denn wäre d) falsch, könnten wir das RSA-Problem zu $(N, e, h_0) \leftarrow \mathsf{RSAIndex}(1^n)$ mit nicht vernachlässigbarer Wahrscheinlichkeit lösen. Simuliere einen regulären Angriff von $\mathcal{A}_{\mathsf{Pim}}$ inklusive Aufdeckung, wenn $\mathcal{S}$ eine von uns gewählte Nachricht $m \in_{\mathsf{R}} \mathsf{M}$ hinterlegt, allerdings mit dem modifizierten Public-Parameter $h_1 := h_0^{-x} w^e \bmod N$ für $w \in_{\mathsf{R}} \mathbb{Z}_N^*$ und $x := H(G^m R^e)$. Über einen erfolgreichen $\mathcal{A}_{\mathsf{Pim}}$ erhalten wir eine Aufdeckung von M^* in Form einer Nachricht m^*, der Extraktor $\mathcal{E}$ liefert bei Erfolg eine weitere Aufdeckung m'. Falls $(m, m') \notin \mathsf{R}$, folgt $m^* \neq m'$ wegen $(m, m^*) \in \mathsf{R}$. Sofern der Unterschied zwischen $\pi(\mathcal{A}_{\mathsf{Pim}})$ und $\pi(\mathcal{E})$ nicht vernachlässigbar ist, erhalten wir zwei RSA-Repräsentationen von M^* und können, wie unter Punkt b) beschrieben, eine e-te Wurzel von h_0 modulo N effizient bestimmen. □

Die Voraussetzung einer kollisions-resistenten Hashfunktion kann abgeschwächt werden, eine Familie H_n *universeller Oneway-Hashfunktion* (Universal Oneway Hash Function, kurz UOWHF) [NY90] ist ausreichend. Statt zu gegebener Hashfunktion einen Hashwert samt zweier verschiedener

Urbilder zu bestimmen, wählt der Algorithmus $\mathcal{A}$ zu 1^n ein x, erhält anschließend $H \in_R \mathsf{H}_n$ und soll in Polynomialzeit ein x' mit $H(x) = H(x')$ berechnen. Für Familien universeller Oneway-Hashfunktionen ist die Erfolgswahrscheinlichkeit von $\mathcal{A}$ bestensfalls vernachlässigbar. Während man UOWHF aus jeder beliebigen Oneway-Funktion konstruieren kann [Ro90], erscheint fraglich, ob die Existenz kollisions-resistenter Hashfunktionen auf das Vorhandensein von Oneway-Permutationen reduzierbar ist [Si98]. Die Aussage des Satzes 3.3.1 gilt auch, wenn H aus einer Familie universeller Oneway-Hashfunktion gewählt wird. Man kann sich sogar überlegen, dass der Beweis seine Gültigkeit behält, wenn für hinreichend große RSA-Exponenten $e \geq N$ statt des Hashwertes x direkt X verwendet wird. Die Präferenz des deutlich kürzeren Hashwertes x statt X dient der schnelleren Berechnung von h_0^x.

Satz 3.3.1 gilt weiterhin, falls x der Hashwert von M statt von $G^m R^e$ ist und man dieses Commitment $G^m R^e$ komplett aus dem Schema entfernt. Im Beweis des Satzes 3.3.1 haben wir aber an mehreren Stellen, genauer in allen Punkten außer a), eine zufällige Nachricht m aus der Menge M gewählt und in den Public-Parameter h_1 eine Geheimtüre zugeschnitten auf dieses m eingebaut. Bei diesem Vorgehen hängt der öffentliche Wert h_1 von der zufälligen Wahl der Nachricht $m \in_R \mathsf{M}$ sowie als Konsequenz von M ab. Ohne die Einschränkung der Wahl von M in Satz 3.3.1 tritt eine Unwägbarkeit auf: Der Angreifer $\mathcal{A}_{\mathsf{Pim}}$ bestimmt M neben der Relation R basierend auf den Public-Parameters, während beim Beweis h_1 in umgekehrter Abhängigkeit gewählt wurde. Dieses Problem lösen wir durch eine weitere Trapdoor im Commitment $G^m R^e$, welche uns gestattet, für die Wahl von h_1 zunächst $m = 0$ anzunehmen und in der Decommitphase die Hinterlegung mit einem erst dann bestimmten $m \in_R \mathsf{M}$ zu öffnen. Wähle $v, w \in_R \mathbb{Z}_N^*$ und ersetze

$$G := v^e \bmod N$$
$$h_1 := h_0^{-x} w^e \bmod N$$

für $x := H(R^e)$. Der Beweis zu Satz 3.3.1 ist unmittelbar übertragbar. Nur Punkt a) haben wir über das Auffinden einer Kollision des Commitments $G^m R^e$ bewiesen, jedoch für die regulären Public-Parameters.

Korollar 3.3.2. *Unter der RSA-Annahme und der Voraussetzung einer kollisions-resistenten Hashfunktion ist das interaktive Commitment-Schema in Abbildung 3.5 ein polynomialzeit-bindendes, bezüglich Aufdeckung non-malleable Commitment-Schema mit perfekter Geheimhaltung.*

Dieses Schema erfüllt neben der liberalen auch die ϵ-Auslegung des Non-Malleability-Begriffs, man ersetze beim konstruierten Adhoc-Angreifer

$\mathcal{A}_{\text{Adhoc}}$ den Knowledge-Extraktor durch die in Kapitel 1.3 beschriebene ϵ-Variante. Das Hash-&-Commit-Prinzip, also die Hinterlegung des Hashwertes $h(m)$ einer längeren Nachricht m, kann allerdings nicht auf diese Verfahren aus Abbildung 3.5 angewandt werden, denn der Extraktor liefert unter diesen Umständen statt einer in Beziehung zu m stehenden Nachricht m' einen in Relation zu $h(m)$ stehenden Hashwert. Die in Abschnitt 3.4 beschriebene Konstruktion basierend auf dem Chinesischen Restsatz ist hier eine Alternative zum benutzten Proof-Of-Knowledge, die auch die Hashwert-Hinterlegung gestattet.

3.4 Commitment basierend auf Faktorisierung

Ein unmittelbare Übertragung des Commitment-Schemas aus Kapitel 3.3 auf die Faktorisierungsrepräsentation scheitert, weil das verwandte Subprotokoll kein Proof-Of-Knowledge ist, genauer der für den Non-Malleability-Nachweis wichtige Extraktor fehlt. Die Konstruktion des Adhoc-Angreifers $\mathcal{A}_{\text{Adhoc}}$ basiert im wesentlichen auf dem Knowledge-Extraktor, während die Möglichkeit, in der Hinterlegungsphase die Antwort auf die Challenge zu verifizieren, nur von untergeordneter Bedeutung ist. Vielmehr unterstellen wir die Korrektkeit der Response, weil dies der Prämisse entspricht, unter welcher eine passende Aufdeckung des Commitments überhaupt vom Receiver $\mathcal{R}$ akzeptiert wird. Aus diesem Grund ist es nicht entscheidend, ob $\mathcal{R}$ die Antwort des Senders $\mathcal{S}$ bereits während der Commitmentphase verifiziert oder dies erst im Rahmen der Aufdeckung geschieht. Im Ergebnis suchen wir eine Methode, bei der

- die Antwort bei Kenntnis der Hinterlegung m zu verifizieren ist,
- die Response die Geheimhaltung von m nicht in Frage stellt,
- die Ratewahrscheinlichkeit zu vernachlässigen ist und
- bei einer Simulation ein nicht-vernachlässigbarer Anteil richtiger Antworten zur Extraktion der Nachricht führt.

Wir konzipieren im folgenden ein solches Verfahren, dessen Extraktor auf dem Chinesischem Restsatz (CRT) aufbaut, anhand des Commitments basierend auf der Faktorisierungsrepräsentation:

$$M := g^m r^{2^{\tau+t}} \bmod N$$

mit $m \in \mathbb{Z}_{2^t}$. Kürze die Länge t der hinterlegten Nachricht m um $2d$ Bits für ein $d < \frac{1}{2}t$, wobei 2^d vernachlässigbar sei. Bezeichne $\text{PrimeGen}_d(c)$ eine

Funktion, die ein Argument $c \in \mathbb{Z}_{2^t}$ auf eine d-Bit-Primzahl abbilde und kollisions-resistent sei. Für die Konstruktion solcher Abbildungen basierend auf kollisions-resistenten Hashfunktionen oder UOWHF verweisen wir auf die Arbeiten von Gennaro, Halevi und Rabin [GeHR99] sowie Cramer und Shoup [CrSh00]. Die hinterlegte Nachricht $m \in \mathbb{Z}_{2^{t-2d}}$ ergänzt der Sender $\mathcal{S}$ um $2d$ zufällige Bits $s \in_{\mathsf{R}} \mathbb{Z}_{2^{2d}}$ zur t-Bit-Zahl

$$m_{\mathsf{pad}} := s + 2^{2d} m,$$

und addiert zu m_{pad} das 2^t-fache der verwandten Zufallsbits $r \in_{\mathsf{R}} \mathbb{Z}_N^*$, so dass aus dem $(t+n)$-Bit-Wert

$$y := m_{\mathsf{pad}} + 2^t r = s + 2^{2d} m + 2^t r$$

Nachricht m samt Zufallswert r unmittelbar hervorgehen. Sender $\mathcal{S}$ und Receiver $\mathcal{R}$ einigen sich auf einen uniform verteilten Wert c und die d-Bit-Primzahl $p := \mathsf{PrimeGen}_d(c)$ stellt die eigentliche Challenge dar. Auf diese antwortet $\mathcal{S}$ mit der Restklasse von y modulo p:

$$y_p \equiv m_{\mathsf{pad}} + 2^t r \equiv s + 2^{2d} m + 2^t r \pmod{p}.$$

Die Verteilung von s modulo der halbsogroßen Zahl p ist statistisch nah an der uniformen Verteilung auf $\mathbb{Z}_p$ und wirkt im Ergebnis wie ein sogenanntes *One-Time-Pad*: Für jede Nachricht m, Zufallsbits r und jede d-Bit-Zahl p liegt die Verteilung von y_p statistisch nah an der Gleichverteilung auf $\mathbb{Z}_p$. Das resultierende Commitment-Schema erzielt statistische Geheimhaltung[2] und die Ratewahrscheinlichkeit der Response ist vernachlässigbar.

Betrachten wir den CRT-Extraktor $\mathcal{E}_{\mathsf{crt}}$, der in einer Simulation aus den Antworten $y_{p_1}, y_{p_2}, \ldots, y_{p_k}$ zu mehreren Challenges $p_1, p_2, \ldots, p_k$ zunächst y und letztlich m zu rekonstruieren versucht. Vorausgesetzt, die Antworten stimmen mit der jeweiligen Restklasse überein, liefert der Chinesische Restsatz die Lösung y zum Kongruenzsystem

$$\begin{aligned} y &\equiv y_{p_1} \pmod{p_1} \\ y &\equiv y_{p_2} \pmod{p_2} \\ &\vdots \\ y &\equiv y_{p_k} \pmod{p_k}. \end{aligned}$$

Sofern $y < \prod_{i=1}^{k} p_i$, erhalten wir die Nachricht m samt des zur Hinterlegung verwandten Zufallswertes r, mit anderen Worten, $\frac{t+n}{d-1}$ Kongruenzen

[2] Für die Geheimhaltung muß der Sender nicht prüfen, ob p tatsächlich eine Primzahl ist. Diese Eigenschaft verwendet der CRT-Extraktor $\mathcal{E}_{\mathsf{crt}}$.

modulo d-Bit-Primzahlen sind hinreichend. Die erfolgreiche Rekonstruktion setzt allerdings voraus, dass wir diejenigen Antworten y_i identifizieren, welche „richtig“ sind, also mit y modulo p_i übereinstimmen. Ist bei einer Kongruenz des ausgewählten Teilsystems die rechte Seite y_i falsch, liefert der Chinesische Restsatz zwar eine Lösung $\tilde{y}$, die allerdings in der Regel eine falsche Nachricht $\tilde{m}$ bzw. Zufallswert $\tilde{r}$ zur Folge hat. Durch Einsetzen können wir aber verifizieren, ob ein Paar (m, r) einer Aufdeckung des Commitments M entspricht. Die Wahrscheinlichkeit mit welcher man auf diese Art eine von der hinterlegten Nachricht abweichende erhält, ist unter der Faktorisierungsannahme vernachlässigbar, weil anderenfalls die beiden Repräsentationen der Hinterlegung M die Primfaktorisierung des Moduls N liefern.

Die hinreichende Anzahl der Kongruenzen k, für welche nahezu immer ein Teilsystem eine Repräsentation des Commitments M ergibt, bestimmen wir später mit Hilfe der Chernoff-Schranke. Bezeichne $K_{\min}$ die zur Konstruktion notwendige, minimale Anzahl „richtiger“ Kongruenzen, so existieren $\binom{k}{K_{\min}}$ Teilsysteme und jedes führt zu einer Lösung, und erst Einsetzen zeigt, ob diese einer Repräsentation von M entspricht. Für konstantes $K_{\min}$ ist der naive Ansatz, alle möglichen Teilsysteme getrennt zu betrachten, möglich, sonst scheitert diese Vorgehensweise jedoch an der vorgegebenen, polynomiellen Laufzeit des Extraktors. Die Aufgabenstellung

- Gegeben: Primzahlen $p_1 < p_2 < \cdots < p_k$, ganze Zahlen $y_{p_1}, y_{p_2}, \ldots, y_{p_k} \in \mathbb{Z}$ und eine Schranke B.
- Gesucht: Alle natürlichen Zahlen $m < B$ mit $m \equiv y_{p_i} \pmod{p_i}$ für mindestens K der Kongruenzen

ist in anderem Zusammenhang als *CRT-List-Decoding* bekannt. Beim Dekodieren fehler-korrigierender Codes, wenn der Übertragungsfehler jenseits der korrigierbaren Fehlerschranke liegt, ist das Codewort nicht eindeutig bestimmt, weshalb entsprechende Verfahren dann eine Liste der nächstgelegenen Codeworte generieren und man von List-Decoding spricht. Die Übermittlung der Restklassen modulo vorgegebener Primzahl entspricht einem linearen Code [Su01], so dass eine Adaption der Verfahren für CRT-List-Decoding naheliegt. Ein früheres Resultat von Goldreich, Ron und Sudan [GoRoSu99] hat Boneh [B00, Corollary 2.1] verbessert und gleichzeitig den Beweis vereinfacht:

Satz 3.4.1 (Boneh 2000). *Gegeben Primzahlen $p_1 < p_2 < \cdots < p_k$, ganze Zahlen $y_{p_1}, y_{p_2}, \ldots, y_{p_k} \in \mathbb{Z}$ und eine Schranke B. Dann kann man in Polynomialzeit alle natürlichen Zahlen $m < B$ mit $m \equiv y_{p_i} \pmod{p_i}$*

für mindestens K dieser k Kongruenzen berechnen, sofern

$$\frac{K}{k} \geq \frac{\log_2 p_k}{\log_2 p_1} \cdot \sqrt{\frac{\log_2 4B}{\log_2 P}}$$

mit $P := \prod_{i=1}^{k} p_i$.

Abbildung 3.8: Hinterlegung non-malleable Faktorisierungs-Commitment

Sender $\mathcal{S}$	$N, \tau, t, g, G, h_0, h_1$ $H : \mathbb{Z}_N^* \to \mathbb{Z}_{2^t}$	**Receiver** $\mathcal{R}$
Nachricht $m \in \mathbb{Z}_{2^{t-2d}}$		
i) eigentliches Commitment		
Wähle $R, r \in_R \mathbb{Z}_N^*$ Wähle $s \in_R \mathbb{Z}_{2^{2d}}$ $m_{\text{pad}} := s + 2^{2d} m$ $x := H(G^m R^{2^{\tau+t}})$ $M := g^{m_{\text{pad}}} r^{2^{\tau+t}} \bmod N$	$\xrightarrow{x, M}$	
ii) Challenge-Schritt (Coin-Flipping)		
Wähle $a \in_R \mathbb{Z}_{2^t}, u \in_R \mathbb{Z}_N^*$ $A := (h_0^x h_1)^a u^{2^{\tau+2t}} \bmod N$	$\xrightarrow{A}$	Wähle $b \in_R \mathbb{Z}_{2^t}$
	$\xleftarrow{\text{b}}$	
$c := a + b \bmod 2^t$	$\xrightarrow{\text{a,u}}$	$A \overset{!}{\equiv} (h_0^x h_1)^a u^{2^{\tau+2t}} \ (N)$
iii) Response-Schritt		
$p := \text{PrimeGen}_d(c)$ $y := m_{\text{pad}} + 2^t r$ $y_p := y \bmod p$	$\xrightarrow{y_p}$	

Wir übertragen die RSA-basierte Konstruktion aus Kapitel 3.3 auf die Faktorisierungsrepräsentation, die Challenge entspricht einer über das Coin-Flipping gewählten Primzahl p und die Response besteht aus der entsprechenden Restklasse y modulo p. Auf den ersten Blick ist man versucht, den Non-Malleability-Nachweis wie in Satz 3.3.1 auf Seite 63 zu führen. Allerdings scheitert dies an der fehlenden Möglichkeit, bereits während der Hin-

terlegungsphase zu verifizieren, ob die Response korrekt ist. Beim Proof-Of-Knowledge verwirft der Extraktor dank Kontrollgleichung ungültige Antworten sofort. Beim vorliegenden Verfahren fehlt hingegen die Möglichkeit, „falsche“ Restklassen vorab zu identifizieren. Stattdessen speichert der CRT-Extraktor $\mathcal{E}_{\mathsf{crt}}$ die Responses und bestimmt am Ende mit Bonehs Algorithmus (Satz 3.4.1) alle möglichen Lösungen, die aus den gegebenen Kongruenzen hervorgehen, und schließt durch Einsetzen falsche Nachrichten aus. Allerdings führt die Vorgehensweise des Knowledge-Extraktors, bis zur Berechnung der Nachricht den Challenge-Response-Schritt zu wiederholen, nicht mehr zur gewünschten, im Erwartungswert polynomiellen Laufzeit. Die Laufzeit von Bonehs Verfahren hängt auch von der Anzahl k der vorliegenden Kongruenzen, also den bisherigen Wiederholungen, ab.[3] Aus diesem Grund wählen wir die ϵ-Variante der Non-Malleability, bei der uns ein Parameter $\epsilon > 0$ vorgibt, wann die Erfolgswahrscheinlichkeit des Adhoc-Angreifers hinreichend nah an der von $\mathcal{A}_{\mathsf{Pim}}$ liegt, und der CRT-Extraktor $\mathcal{E}_{\mathsf{crt}}$ auf weitere Wiederholungen verzichten kann. Abbildung 3.8 zeigt die ausführliche Hinterlegungsphase, das kompakte Schema ist in Abbildung 3.9 zu sehen.

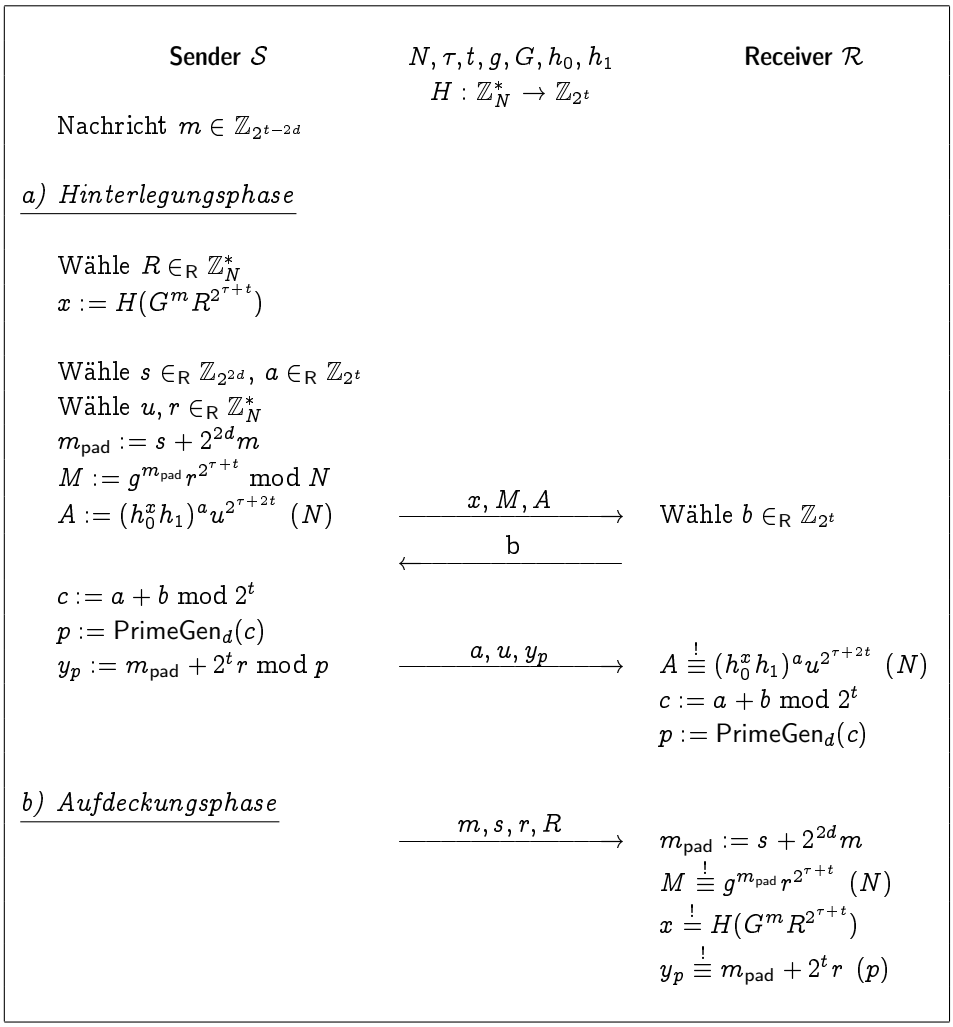
Satz 3.4.2. *Sei $\frac{t+n}{d-1}$ polynomiell in n. Dann ist unter der Faktorisierungsannahme und der Voraussetzung einer kollisions-resistenten Hashfunktion das interaktive Hinterlegungsverfahren aus Abbildung 3.9 ein polynomialzeit-bindendes, bezüglich Aufdeckung ϵ-non-malleable Commitment-Schema mit statistischer Geheimhaltung.*

Beweis. Die statistische Geheimhaltung ist die Konsequenz der perfekten Geheimhaltung des grundlegenden Commitments basierend auf der Faktorisierungsrepräsentation (Satz 3.1.1 auf Seite 49) und des als One-Time-Pad agierenden Wertes s, dessen Verteilung statisch nah an der uniformen Verteilung auf $\mathbb{Z}_p$ ist. Die Polynomialzeit-Bindung des Schemas folgt ebenfalls aus Satz 3.1.1.

Nach Korollar 3.2.3 von Seite 57 impliziert das Vorhandensein eines ϵ-Extraktors, der zu einem Commitment von m die von $\mathcal{A}_{\mathsf{Pim}}$ hinterlegte Nachricht m^* oder eine äquivalente Nachricht m' bestimmt, ϵ-Non-Malleability. Der Extraktor $\mathcal{E}_{\mathsf{crt}}$ folgt dem im Beweis zu Satz 3.3.1 auf Seite 63 konstru-

[3] Zur Erinnerung: Der Knowledge-Extraktor aus Satz 1.3.1 auf Seite 12 wiederholt mit Wahrscheinlichkeit $\pi(\gamma)$ den Challenge-Response-Schritt zum Präfix γ mit jeweils konstantem Aufwand durchschnittlich $\pi(\gamma)^{-1}|\mathcal{A}_{\mathsf{Pim}}|$-mal, so dass wir insgesamt als erwartete Laufzeit $|\mathcal{A}_{\mathsf{Pim}}|$ erhalten. Beim CRT-Extraktor wäre aber der durchschnittliche Aufwand der Wiederholungen mindestens $\sum_{k=1}^{\pi(\gamma)^{-1}} k|\mathcal{A}_{\mathsf{Pim}}|$ und das nicht polynomiell beschränkte $\pi(\gamma)^{-1}$ würde sich beim Mitteln über alle Präfixe nicht wegheben.

Abbildung 3.9: Non-malleable Faktorisierungs-Commitment

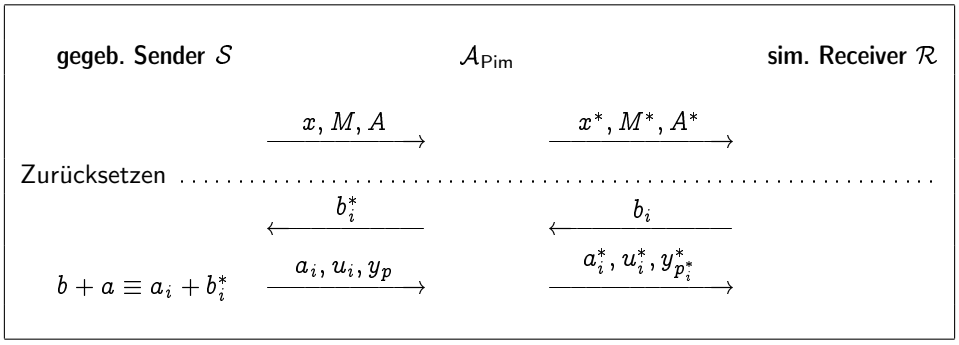


ierten. Für die Analyse setzen wir o.B.d.A.

$$\epsilon < \pi(\mathcal{A}_{\text{Pim}}) \tag{3.6}$$

voraus, denn sonst ist die vorgegebene Wahrscheinlichkeit, mit welcher $\mathcal{E}_{\text{crt}}$ eine in Relation R zu m stehende Nachricht m' ermitteln soll, gleich Null und eine Analyse kann unter diesen Umständen entfallen. Wir nehmen zunächst an, $\mathcal{A}_{\text{Pim}}$ wählt die Nachrichtenmenge M nur in Abhängigkeit von (t, d). Als Eingabe erhalten wir analog zum Beweis von Satz 3.2.2 auf Seite 54 neben den Public-Parameters zu einer uns unbekannten Nachricht $m \in_R$ M die Commitmentphase (x, M, A, b, a, u, y_p) samt Seiteninformationen Hist(m).

Abbildung 3.10: Simulation non-malleable Faktorisierungs-Commitment



Für die Anwendung des Extraktors $\mathcal{E}_{\mathsf{crt}}$ auf $\mathcal{A}_{\mathsf{Pim}}$ baue eine Geheimtüre in den Public-Parameter h_1 ein:

$$h_1 := h_0^{-x} w^{2^{\tau+2t}} \bmod N$$

Die übrigen öffentlichen Werte bleiben unberührt. Die Trapdoor w, eine $2^{\tau+2t}$-ten Wurzel von $h_0^x h_1$ modulo N, erlaubt dem Simulator das Commitment A beliebig zu öffnen. Zwar ist im allgemeinen bei der Aufdeckung von $A \equiv (h_0^x h_1)^a u^{2^{\tau+2t}} \pmod N$ mit a_i der zweite Wert u_i der Repräsentation nicht eindeutig bestimmt, weil wir allerdings in der Simulation stets die $2^{\tau+t}$-ten Wurzel w verwenden, also

$$A \equiv (h_0^x h_1)^a u^{2^{\tau+2t}} \equiv (h_0^x h_1)^{a_i} (w^{a_i - a} u)^{2^{\tau+2t}} \pmod N,$$

legen im vorliegenden Fall (a, u, w) und a_i das vom Simulator gewählte $u_i := w^{a_i - a} u \bmod N$ eindeutig fest. Analog zum Beweis von Satz 3.3.1 ist unter der Faktorisierungsannahme und der Kollisions-Resistenz von $H(\cdot)$ die Wahrscheinlichkeit von $x = x^*$ vernachlässigbar, weshalb $\mathcal{A}_{\mathsf{Pim}}$ außer Stande ist, sein Coin-Flipping-Commitment A^* mehrdeutig zu öffnen. In der Simulation wird $\mathcal{A}_{\mathsf{Pim}}$ wie in Abbildung 3.11 skizziert an die Stelle vor Senden des Anteils b zum Coin-Flipping zurückgesetzt. Sei Γ die Menge der Tupel (Präfixe)

$$\gamma := (\sigma, w, \omega, x, M, A, a, u, c, y_p),$$

für welche bei Public-Parameters σ der PIM-Angreifer $\mathcal{A}_{\mathsf{Pim}}$ mit fixierten Zufallsbits ω, nach Erhalt von b die Hinterlegungsphase korrekt abschließt und dann sein Commitment nach der Aufdeckung von (x, M) mit m mit Wahrscheinlichkeit $\frac{1}{2}\epsilon$ erfolgreich ist, wenn $\mathcal{S}$ sein Coin-Flipping-Commitment A so öffnet, dass der Sender sich mit $\mathcal{A}_{\mathsf{Pim}}$ auf die Challenge c festlegt und

y_p sendet. Mit anderen Worten, für $\gamma \in \Gamma$ gibt (in der Simulation) $\mathcal{A}_{\mathsf{Pim}}$ mit Wahrscheinlichkeit mindestens $\frac{1}{2}\epsilon$ bezogen auf die generierte Challenge c^* die zur späteren Aufdeckung passende Antwort $y^*_{p^*}$ (Weil alle anderen Werte fixiert sind, bezieht sich hier die Wahrscheinlichkeit lediglich auf die Wahl von b, welches der Receiver $\mathcal{R}$ an $\mathcal{A}_{\mathsf{Pim}}$ sendet). Für $\gamma \notin \Gamma$ sinkt die Wahrscheinlichkeit, dass $\mathcal{A}_{\mathsf{Pim}}$ erfolgreich ist, unter $\frac{1}{2}\epsilon$. Aus

$$\mathrm{Ws}[\mathcal{A}_{\mathsf{Pim}} \text{ erfolgr. und } \gamma \notin \Gamma] \leq \mathrm{Ws}_b[\mathcal{A}_{\mathsf{Pim}} \text{ erfolgr.} \,|\, \gamma \notin \Gamma] < \tfrac{1}{2}\epsilon$$

folgt in Verbindung mit der Annahme (3.6),

$$\mathrm{Ws}[\gamma \notin \Gamma \,|\, \mathcal{A}_{\mathsf{Pim}} \text{ erfolgr.}] \leq \frac{\mathrm{Ws}[\mathcal{A}_{\mathsf{Pim}} \text{ erfolgr. und } \gamma \notin \Gamma]}{\mathrm{Ws}[\mathcal{A}_{\mathsf{Pim}} \text{ erfolgr.}]} < \frac{\frac{1}{2}\epsilon}{\pi(\mathcal{A}_{\mathsf{Pim}})} \leq \tfrac{1}{2},$$

so dass bei einem Erfolg von $\mathcal{A}_{\mathsf{Pim}}$ in mindestens der Hälfte aller Fälle $\gamma \in \Gamma$ zutrifft.

Abbildung 3.11: CRT-Extraktor $\mathcal{E}_{\mathsf{crt}}$

1. Fixiere Zufallsbits ω von $\mathcal{A}_{\mathsf{Pim}}$
2. Simuliere erste Schritte der Commitmentphase und Angriff von $\mathcal{A}_{\mathsf{Pim}}$
   Seien x^*, M^*, A^*, die von $\mathcal{A}_{\mathsf{Pim}}$ gesandten Daten
3. Falls $(M \equiv M^*)$ oder $(M \not\equiv M^*$ und $x = x^*)$, dann stoppe ohne Ausgabe
4. $\mathcal{K} = \emptyset$
5. Wiederhole bis zu $\frac{64(t+n)\epsilon^{-2}}{d-1}$-mal:
   i. Simuliere Challenge-Response-Schritt,
      sei $a_i^*, u_i^*, y^*_{p_i^*}$ die Response von $\mathcal{A}_{\mathsf{Pim}}$ auf c_i^* bzw. p_i^*
   ii. Falls a_i^* von bisherigen a^*-Werten abweicht, dann stoppe.
   iii. Füge Kongruenz $y^* \equiv y^*_{p_i^*} \pmod{p_i^*}$ zur Liste $\mathcal{K}$ hinzu.
6. Bestimmte alle Lösungen y' bzw. (m'_{pad}, r') von $\mathcal{K}$
   Gib m' aus, falls (m'_{pad}, r') Repräsentation zu M^*.

Wir zeigen, dass unter der Bedingung $\gamma \in \Gamma$ der Extraktor $\mathcal{E}_{\mathsf{crt}}$ eine in Beziehung zu m stehende Nachricht m' mit Wahrscheinlichkeit $\pi(\mathcal{A}_{\mathsf{Pim}}) - \frac{1}{2}\epsilon - \nu$ für ein vernachlässigbares ν liefert. Selbst wenn für $\gamma \notin \Gamma$ die Bestimmung einer passenden Nachricht fehlschlägt, steht insgesamt der extrahierte Wert m' hinreichend oft in Relation zu m:

$$\begin{aligned}
\pi(\mathcal{A}_{\mathsf{Pim}}) - \pi(\mathcal{E}) &\leq \mathrm{Ws}\left[\begin{array}{l}\mathcal{A}_{\mathsf{Pim}} \text{ erfolgreich} \\ \mathcal{E}_{\mathsf{crt}} \text{ nicht erfolgr.}\end{array}\middle|\, \gamma \in \Gamma\right] + \mathrm{Ws}\left[\begin{array}{l}\mathcal{A}_{\mathsf{Pim}} \text{ erfolgreich} \\ \mathcal{E}_{\mathsf{crt}} \text{ nicht erfolgr.}\end{array}\middle|\, \gamma \notin \Gamma\right] \\
&< (\tfrac{1}{2}\epsilon + \nu) + \tfrac{1}{2}\epsilon \\
&= \epsilon + \nu
\end{aligned}$$

Unter der Voraussetzung der Faktorisierungsannahme sowie der Kollisions-Resistenzen von $H(\cdot)$ und $\mathsf{PrimeGen}_d(\cdot)$ gilt:

a) Die Wahrscheinlichkeit für einen erfolgreichen $\mathcal{A}_{\mathsf{Pim}}$ mit $x = x^*$ und $\gamma \in \Gamma$ ist vernachlässigbar.

b) Unter der Bedingung $\gamma \in \Gamma$ ist die Wahrscheinlichkeit für einen erfolgreichen $\mathcal{A}_{\mathsf{Pim}}$, der nach Zurücksetzen dann eine alternative Öffnung von A^* gibt, vernachlässigbar.

c) Unter der Bedingung $\gamma \in \Gamma$ ist die Wahrscheinlichkeit für einen erfolgreichen $\mathcal{A}_{\mathsf{Pim}}$, der beim Zurücksetzen dann auf die Challenge mit der Restklasse zu einer anderen Nachricht m'_{pad} antwortet und die Hinterlegung erfolgreich aufdeckt, vernachlässigbar.[4]

d) Unter der Bedingung $\gamma \in \Gamma$ liefert der Extraktor $\mathcal{E}_{\mathsf{crt}}$ aus Abbildung 3.11 mit Wahrscheinlichkeit mindestens $\pi(\mathcal{A}_{\mathsf{Pim}}) - \frac{1}{2}\epsilon - \nu$ eine Aufdeckung des Commitments M^* in Form einer Nachricht m' für ein vernachlässigbares ν.

e) Die extrahierte Nachricht m' steht bis auf vernachlässigbare Wahrscheinlichkeit in Relation R zu m.

Zum Nachweis der Aussage a) geht man analog wie im Beweis von Satz 3.3.1 auf Seite 63 vor, für die Analyse beachte, dass beim Erfolg von $\mathcal{A}_{\mathsf{Pim}}$ in mindestens der Hälfte aller Fälle $\gamma \in \Gamma$ gilt.

Träfe Behauptung b) nicht zu, wäre entgegen Faktorisierungsannahme der Modul effizient mit nicht vernachlässigbarer Wahrscheinlichkeit zu faktorisieren. Simuliere einen regulären Angriff von $\mathcal{A}_{\mathsf{Pim}}$ inklusive Aufdeckung, wenn $\mathcal{S}$ eine von uns gewählte Nachricht $m \in_{\mathsf{R}} \mathsf{M}$ hinterlegt, allerdings mit dem modifizierten Public-Parameter $h_1 := h_0^{-x} w^{2^{\tau+2t}} \bmod N$ für $w \in_{\mathsf{R}} \mathbb{Z}_N^*$ und $x := H(G^m R^{2^{\tau+t}})$. Falls $\mathcal{A}_{\mathsf{Pim}}$ nicht erfolgreich bezüglich Aufdeckung ist oder $x = x^*$, dann stoppe. Anderenfalls gilt in mindestens der Hälfte aller Fälle $\gamma \in \Gamma$. Setze $\mathcal{A}_{\mathsf{Pim}}$ an die Stelle, an welche er auf b wartet, zurück und wiederhole die Ausführung, bis wir neben a_1^* in einer erfolgreichen Commitmentphase von $\mathcal{A}_{\mathsf{Pim}}$ eine zweite Aufdeckung a_2^* von A^* erhalten. Dabei generieren wir dank Geheimtüre jeweils die gleiche Challenge, so dass insgesamt Vorgehen und Umfeld mit dem des Extraktors $\mathcal{E}_{\mathsf{crt}}$ bei $\mathcal{A}_{\mathsf{Pim}}$ übereinstimmen. Es gilt

$$(h_0^{x^*} h_1)^{a_1^*} (u_1^*)^{2^{\tau+2t}} \equiv A^* \equiv (h_0^{x^*} h_1)^{a_2^*} (u_2^*)^{2^{\tau+2t}} \pmod{N}$$

[4] In diesem Punkt weicht die Liste der Behauptungen von den entsprechenden Aussagen auf Seite 65 zum RSA-Schema (Satz 3.3.1) ab. Der Grund ist die fehlende Kontrollgleichung für die Antwort auf die Challenge, bei der vorliegenden CRT-Konstruktion kann man die Response erst nach der Aufdeckung verifizieren.

und Einsetzen von $h_1 \equiv h_0^{-x} w^{2^{\tau+2t}} \pmod{N}$ liefert:

$$h_0^{(x^*-x)a_1^*}(w^{a_1^*}u_1^*)^{2^{\tau+2t}} \equiv A^* \equiv h_0^{(x^*-x)a_2^*}(w^{a_2^*}u_2^*)^{2^{\tau+2t}} \pmod{N}.$$

Mit $\Delta x := x^* - x$ und $\Delta a^* = a_1^* - a_2^*$ können wir dies schreiben als:

$$h_0^{\Delta x \cdot \Delta a^*} \equiv (w^{-\Delta a^*}(u_1^*)^{-1}u_2^*)^{2^{\tau+2t}} \pmod{N}.$$

Aufgrund der Annahme $x \neq x^*$ ist $1 \leq |\Delta x| < 2^t$, wegen $a_1^* \neq a_2^*$ und $0 \leq a_1^*, a_2^* < 2^t$ folgt $1 \leq |\Delta a^*| < 2^t$, so dass insgesamt gilt:[5]

$$1 \leq |\Delta x \cdot \Delta a^*| < 2^t \cdot 2^t = 2^{2t}.$$

Wir verfügen über zwei verschiedene Repräsentationen von A^* bezüglich $(N, \tau, 2t, h_0)$. Diese Möglichkeit, das Faktorisierungsrepräsentationsproblem zu $(N, \tau, 2t, h_0) \leftarrow \mathsf{FactIndex}(1^n)$ in Polynomialzeit mit nicht vernachlässigbarer Wahrscheinlichkeit zu lösen, gestattet nach Satz 2.1.8 von Seite 39, die Primfaktoren des Moduls N effizient zu bestimmen.

Träfe Aussage c) nicht zu, wäre ebenso im Widerspruch zur Faktorisierungsannahme der Modul effizient mit nicht vernachlässigbarer Wahrscheinlichkeit zu faktorisieren. Simuliere einen regulären Angriff von $\mathcal{A}_{\mathsf{Pim}}$ samt Aufdeckung m^*_{pad}, wenn $\mathcal{S}$ eine von uns gewählte Nachricht $m \in_{\mathsf{R}} \mathsf{M}$ hinterlegt, allerdings mit dem modifizierten Public-Parameter $h_1 := h_0^{-x} w^{2^{\tau+t}} \bmod N$ für $w \in_{\mathsf{R}} \mathbb{Z}_N^*$ und $x := H(G^m R^{2^{\tau+t}})$. Resete $\mathcal{A}_{\mathsf{Pim}}$ und setze die Ausführung fort, bis die (noch zu bestimmende) maximale Anzahl k der Iterationen erreicht ist oder wir eine von m^*_{pad} abweichende Aufdeckung m'_{pad} erhalten. Dabei generieren wir dank Geheimtüre jeweils die gleiche Challenge, so dass insgesamt Vorgehen und Umfeld während der Hinterlegungsphase mit denen des Extraktors $\mathcal{E}_{\mathsf{crt}}$ bei $\mathcal{A}_{\mathsf{Pim}}$ übereinstimmen. Die Fähigkeit, effizient M^* samt zweier verschiedener Repräsentationen zu berechnen, also das Faktorisierungsrepräsentationsproblem zu $(N, \tau, t, g) \leftarrow \mathsf{FactIndex}(1^n)$ in Polynomialzeit mit nicht vernachlässigbarer Wahrscheinlichkeit zu lösen, gestattet nach Satz 2.1.8 von Seite 39, den Modul N effizient zu faktorisieren.

Für die Aussage d) betrachte den Extraktionsprozeß unter der Bedingung, dass ein erfolgreicher $\mathcal{A}_{\mathsf{Pim}}$ beim Zurücksetzen jeweils die gleiche Öffnung von A^* gibt, also $a_1^* = a_2^* = \ldots$ gilt. Nach Behauptung b) ist diese Annahme nahezu immer erfüllt. Weil $b_1, b_2, \ldots$ zufällig und unabhängig vom

[5] Zum Vergleich: Im Fall des RSA-Schemas, Gleichung (3.5) auf Seite 66, folgt aus $\Delta x, \Delta a^* \not\equiv 0 \pmod{e}$ dass auch das Produkt $\Delta x \cdot \Delta a^*$ teilerfremd zum primen e ist. Weil hier statt eines primen e eine (nicht-prime) Zweier-Potenz vorliegt, ist auf diese Weise nicht auszuschließen, dass das Produkt $\Delta x \cdot \Delta a^*$ just 2^t entspricht. Aus diesem Grund erhöhen wir die Zweier-Potenz der Repräsentation von $2^{\tau+t}$ auf $2^{\tau+2t}$.

simulierten $\mathcal{R}$ gewählt werden, sind die generierten Challenges $a_i^* + b_i \bmod 2^t$ zwischen $\mathcal{A}_{\mathsf{Pim}}$ und $\mathcal{R}$ ebenso unabhängig und uniform verteilt. Aufgrund der Kollisions-Resistenz von $\mathsf{PrimeGen}_d(\cdot)$ genügt es zu zeigen, dass wir bei

$$k := \frac{64(t+n)\epsilon^{-2}}{d-1} \tag{3.7}$$

Wiederholungen des Challenge-Response-Schrittes hinreichend viele Kongruenzen erhalten, um nach Aussage c) die Aufdeckung des Commitments M^* zu bestimmen.

Wegen $\gamma \in \Gamma$ antwortet $\mathcal{A}_{\mathsf{Pim}}$ mit Wahrscheinlichkeit mindestens $\frac{1}{2}\epsilon$ mit der „richtigen“ Restklasse, die zur späteren Aufdeckung paßt. Gemäß Chernoff-Schranke (1.4) von Seite 14 (mit $p = \frac{1}{2}\epsilon$ und $\delta = \frac{1}{2}$) gilt für die Anzahl K der richtigen Kongruenzen

$$K \geq \frac{16(t+n)\epsilon^{-1}}{d-1} \tag{3.8}$$

bis auf Wahrscheinlichkeit deutlich unterhalb $\frac{1}{2}\epsilon$. Um Bonehs Algorithmus anzuwenden, sind die in Satz 3.4.1 auf Seite 70 genannten Bedingung nachzuweisen. In der vorliegenden Situation gilt $\log_2 4B \leq 2(t+n)$ und

$$\log_2 P \geq k(d-1) = 64(t+n)\epsilon^{-2} \qquad \frac{\log_2 p_k}{\log_2 p_1} \leq \frac{d}{d-1} \leq \sqrt{2}.$$

für $d \geq 3$. Die gewählten Parameter (3.7) und (3.8) erfüllen die Voraussetzung:

$$\frac{K}{k} = \frac{\epsilon}{4} = \sqrt{\frac{4}{64\epsilon^{-2}}} \geq \sqrt{\frac{2 \cdot 2(t+n)}{64(t+n)\epsilon^{-2}}} \geq \frac{\log_2 p_k}{\log_2 p_1} \cdot \sqrt{\frac{\log_2 4B}{\log_2 P}}.$$

Wie in Abbildung 3.11 beschrieben, bestimmen wir mit Bonehs Algorithmus alle Lösungen y bzw. möglichen Repräsentationen (m', r') und überprüfen durch Einsetzen, ob diese zu M passen. Die Wahrscheinlichkeit, beim Einsetzen zwei passende Aufdeckungen von M^* zu erhalten, ist unter der Faktorisierungsannahme vernachlässigbar, weil wir anderenfalls N effizient in seine Primfaktoren zerlegen können.

Der Nachweis von Behauptung e) entspricht dem Vorgehen bei Aussage b), denn wäre e) falsch, könnten wir den Modul N effizient mit nicht vernachlässigbarer Wahrscheinlichkeit faktorisieren. Simuliere einen regulären Angriff von $\mathcal{A}_{\mathsf{Pim}}$ inklusive Aufdeckung, wenn $\mathcal{S}$ eine von uns gewählte Nachricht $m \in_{\mathsf{R}} \mathsf{M}$ hinterlegt, allerdings mit dem modifizierten Public-Parameter $h_1 := h_0^{-x} w^{2^{\tau+2t}} \bmod N$ für $w \in_{\mathsf{R}} \mathbb{Z}_N^*$ und $x := H(G^m R^{2^{\tau+t}})$. Über einen erfolgreichen $\mathcal{A}_{\mathsf{Pim}}$ erhalten wir eine Aufdeckung von M^* in Form einer Nachricht m^*_{pad}, der Extraktor $\mathcal{E}$ liefert bei Erfolg eine weitere Aufdeckung m'_{pad}.

Falls $(m, m') \notin \mathsf{R}$, folgt $m^*_{\mathsf{pad}} \neq m'_{\mathsf{pad}}$ wegen $(m, m^*) \in \mathsf{R}$. Sofern der Unterschied zwischen $\pi(\mathcal{A}_{\mathsf{Pim}})$ und $\pi(\mathcal{E})$ nicht vernachlässigbar ist, erhalten wir zwei Faktorisierungsrepräsentationen von M^*, was ermöglicht, wie unter Punkt b) dargestellt, die Primfaktoren des Modul N zu berechnen.

Wir haben zur Vereinfachung die Wahl der Nachrichtenmengen M eingeschränkt. Der Beweis kann analog zum Vorgehen bei Korollar 3.3.2 auf Seite 67 für von $\mathcal{A}_{\mathsf{Pim}}$ vorgegebene Nachrichtenmengen M angepaßt werden. Wir plazieren in das Commitment $G^m R^e$ eine Trapdoor, welche uns gestattet, für die Wahl von h_1 zunächst $m = 0$ anzunehmen und in der Decommitphase die Hinterlegung mit einem erst dann bestimmten $m \in_{\mathsf{R}} \mathsf{M}$ zu öffnen. Wähle $v, w \in_{\mathsf{R}} \mathbb{Z}^*_N$ und ersetze

$$G := v^{2^{\tau+t}} \bmod N$$
$$h_1 := h_0^{-x} w^{2^{\tau+t}} \bmod N$$

für $x := H(R^{2^{\tau+t}})$. Der Non-Malleability Beweis ist unmittelbar übertragbar, lediglich Punkt a) haben wir über das Auffinden einer Kollision des Commitments $G^m R^{2^{\tau+t}}$ bewiesen, jedoch für die regulären Public-Parameters. □

Im Gegensatz zum auf der RSA-Repräsentation basierenden non-malleable Commitment-Schema aus Abschnitt 3.3 ist unklar, ob das Verfahren aus Abbildung 3.9 neben der ϵ- auch die liberale Auslegung des Non-Malleability-Begriffs erfüllt.

Die Konstruktion aufbauend auf dem Chinesischen Restsatz eröffnet die Möglichkeit des Hash-&-Commit-Prinzips, um Nachrichten der Bitlänge ℓ zu hinterlegen. Sei $h : \mathbb{Z}_{2^{\ell+2d}} \to \mathbb{Z}_{2^t}$ eine kollisions-resistente Hashfunktion. Abweichend vom Schema in Abbildung 3.9 bildet der Sender $\mathcal{S}$

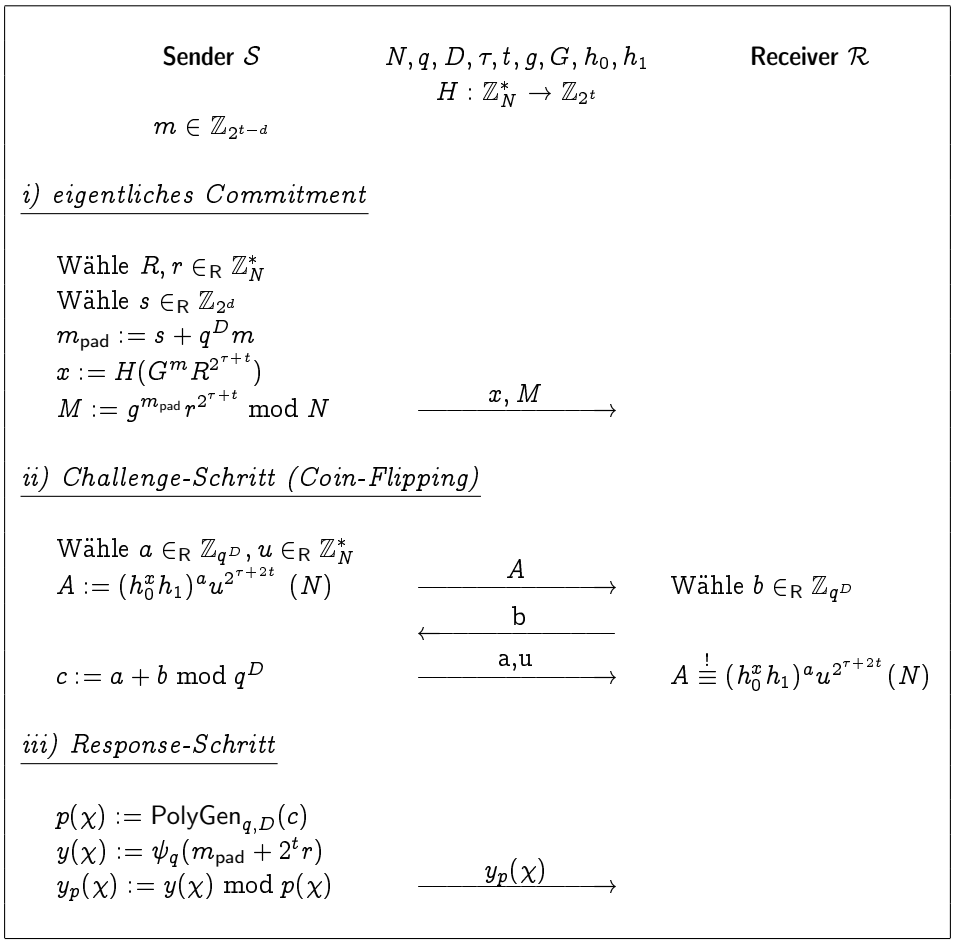
$$m_{\mathsf{pad},h} := h(m_{\mathsf{pad}}) = h(s + 2^{2d} m)$$

und verbrieft diesen Hashwert als $M := g^{m_{\mathsf{pad},h}} r^{2^{\tau+t}} \bmod N$. Auf die Challenge in Form der Primzahl p antwortet $\mathcal{S}$ mit

$$y_p := s + 2^{2d} m + 2^{2d+\ell} r \bmod p.$$

Der Sender $\mathcal{S}$ hinterlegt zwar den Hashwert von m_{pad}, die Response bezieht sich jedoch auf das eigentliche m_{pad}. Unter der Voraussetzung, dass $\frac{2d+\ell+n}{d-1}$ polynomiell in n ist, liefert der im Beweis zu Satz 3.4.2 konstruierte Extraktor $\mathcal{E}_{\mathsf{crt}}$ eine Liste der möglichen (s, m, r) und mit Hilfe der Hashfunktion ist durch Einsetzen eine passende Darstellung von M zu identifizieren. Die Konstruktion eines ϵ-non-malleable Commitments ist auch für die RSA- statt der Faktorisierungsrepräsentation möglich, man erhält

Abbildung 3.12: Hinterlegung bei Polynom-Challenge



auf diese Weise ein (liberales) non-malleable Hinterlegungsverfahren für das Hash-&-Commit-Prinzip.

Betrachtet man das Hinterlegungsschema aus Abbildung 3.9 vom Blickpunkt der Effizienz, ist der Flaschenhals der Hinterlegungsphase die Transformation des zufälligen $c \in \mathbb{Z}_{2^t}$ in eine Primzahl $p := \text{PrimeGen}_d(c)$. Die unmittelbare Verwendung von c statt der Primzahl scheidet aus, weil die Wahrscheinlichkeit, dass zwei „zufällige“ Zahlen teilerfremd sind, mit $\frac{6}{\pi^2} \approx 0,61$ nicht vernachlässigbar ist [K98, Sec. 4.5.2]. Und bei nicht paarweise teilerfremden Modulen ist nachstehende Situation

$$y \equiv 1 \pmod 2 \qquad y \equiv 2 \pmod 3$$
$$y \equiv 2 \pmod 4$$

möglich, in der sich die beiden Kongruenzen auf der linken Seite widersprechen, da $y \equiv 2 \pmod 4$ bedingt $y \equiv 0 \pmod 2$. Ein Ausweg ist der Übergang von $\mathbb{Z}$ zum Polynomring $\mathbb{F}_q[\chi]$ über dem endlichen Körper $\mathbb{F}_q$ mit q Elementen für eine $\sqrt{d}$-Bit-Primzahl q. Zur Vereinfachung nehmen wir an, d sei eine Quadratzahl $d = D^2$. Um eine natürliche Zahl $z \in \mathbb{N}$ in dem Polynomring $\mathbb{F}_q[\chi]$ bzw. $\mathbb{Z}_q[\chi]$ einzubetten, schreibe den Wert in q-ärer Darstellung $z = \sum_{i \geq 0} z_i q^i$ und ordne z das Polynom mit den Koeffizienten $z_0, z_1, \ldots$ zu:

$$\psi_q(z) := \sum_{i \geq 0} z_i \chi^i \in \mathbb{F}_q[\chi]$$

Die Wahrscheinlichkeit, dass zwei zufällige, normierte (nicht-konstante) Polynome aus $\mathbb{F}_q[\chi]$ gleichen Grades relativ prim sind, ist vernachlässigbar: Sie entspricht unabhängig vom Grad $1 - \frac{1}{q}$ [K98, Sec. 4.6.1, Ex. 5]. Als Challenge wählen wir statt einer d-Bit-Primzahl ein zufälliges, normiertes Polynom aus $\mathbb{F}_q[\chi]$ vom Grad D. Dazu ändere die Länge des Arguments c zur Generierung der Challenge von t auf $d < t$ Bits, genauer $c \in \mathbb{Z}_{q^D}$, und wähle einfach das korrespondierende, normierte Polynom vom Grad D:

$$p(\chi) = \mathsf{PolyGen}_{q,D}(c) := \chi^D + \psi_q(c).$$

Die Funktion $\mathsf{PolyGen}_{q,D}(c)$ ist injektiv, verschiedene $c \in \mathbb{Z}_{q^D}$ werden auf verschiedene, normierte Polynome vom Grad D abgebildet. Auf die Challenge $p(\chi)$ antwortet der Sender S mit

$$y_p(\chi) := \underbrace{\psi_q(m_{\mathsf{pad}} + 2^t r)}_{\psi_q(s)+\ldots} \bmod p(\chi)$$

zu $m_{\mathsf{pad}} = s + q^D m$ mit $s \in_{\mathsf{R}} \mathbb{Z}_{q^D}$. Das Polynom $\psi_q(s)$ vom Grad $D - 1$ übernimmt hier die Rolle eines echten One-Time-Pads. Abbildung 3.12 zeigt die resultierende Hinterlegungsphase, wenn man statt Restklassen modulo Primzahlen die Restklassen modulo zufälliger Polynome verwendet. Weil $\psi_q(s)$ uniform verteilt im Faktorring $\mathbb{F}_q[\chi]/(p)$ ist, bietet das Schema perfekte Geheimhaltung. Für die Rekonstruktion „sammelt“ der Extraktor die Restklassen des Polynoms $y(\chi)$ modulo paarweise relativ primer Polynome $p_1(\chi), p_2(\chi), \ldots$ und setzt $y(\chi)$ zusammen. Denn für Polynome über einem Körper $\mathbb{F}$ gilt ebenfalls der Chinesische Restsatz [K98, Sec. 4.6.2, Ex. 3]: Gegeben k paarweise relativ prime Polynome $p_1, p_2, \ldots, p_k \in \mathbb{F}[\chi]$ sowie

$y_{p_1}, y_{p_2}, \ldots, y_{p_k} \in \mathbb{F}[\chi]$ hat das Kongruenzensystem

$$\begin{aligned} y &\equiv y_{p_1} \pmod{p_1} \\ y &\equiv y_{p_2} \pmod{p_2} \\ &\vdots \\ y &\equiv y_{p_k} \pmod{p_k} \end{aligned}$$

eine eindeutig bestimmte Lösung $y \in \mathbb{F}[\chi]$ mit $\operatorname{grad}(y) \leq \sum_{i=1}^{k} \operatorname{grad}(p_i)$. Für konstantes $\frac{n+t}{D}$ kann der Extraktor über Testen aller möglichen Teilsysteme in Polynomialzeit die Nachrichten rekonstruieren, so dass sich der Non-Malleability-Nachweis in Satz 3.4.2 auch auf die Modifikation aus Abbildung 3.12 erstreckt:

Satz 3.4.3. *Sei $\frac{t+n}{D} = \mathcal{O}(1)$. Dann ist unter der Faktorisierungsannahme und der Voraussetzung einer kollisions-resistenten Hashfunktion das interaktive Hinterlegungsverfahren aus Abbildung 3.12 ein polynomialzeit-bindendes, bezüglich Aufdeckung ϵ-non-malleable Commitment-Schema mit perfekter Geheimhaltung.*

Für polynomielles $\frac{t+n}{D}$ oder um das Schema sinnvoll (d.h. mit moderatem D) auf die Hinterlegung von Hashwerten zu erweitern, ist der List-Decoding-Algorithmus für CRT-Codes basierend auf Polynomen statt Primzahlen zu modifizieren. Eine einfache Anpassung von Bonehs Verfahren (Satz 3.4.1 auf Seite 70) scheidet aus, denn bei diesem Verfahren bestimmt man durch Gitterreduktion ein Polynom $Q \in \mathbb{Z}[Y]$ mit kleinen Koeffizienten und $Q(y_i) \equiv 0 \pmod{\prod_i p_i}$, um dann mittels Faktorisierung von $Q(Y)$ über $\mathbb{Z}$ die Nullstellen und die gesuchten Lösungen zu finden [B00]. Im Fall des Polynomringes suchen wir wie allgemein von Sudan [Su01, Sec. 3] beschrieben ein Polynom $Q(\chi, Y) \in \mathbb{F}[\chi, Y]$ kleinen Grades, was entsprechend der Situation wie bei der List-Dekodierung von Reed-Solomon-Codes auf das Lösen eines linearen Gleichungssystems über $\mathbb{F}$ hinausläuft [Su97a, Su97b].

3.5 Nicht-Interaktive Commitments

Für das Public-Parameter-Modell stammt von Di Crescenzo, Ishai und Ostrovsky [DIO98] ein einfaches und zugleich nicht-interaktives Verfahren, welches auf Naors Schema [N91] aufbaut. Jedoch schränken sie den Non-Malleability-Begriff ein: Die History-Funktion Hist(·) sei konstant für alle Nachrichten, d.h. Seiteninformationen fehlen. Diese Restriktion ermöglicht innerhalb der Simulation, mit Hilfe der Trapdoor eine Hinterlegung mehrdeutig ohne Gefahr möglicher Inkonsistenzen zu den Seiteninformationen

aufzudecken. Während man im Fall des Schemas basierend auf der RSA-Repräsentation (Abschnitt 3.3) über den Umweg eines Knowledge-Extraktors bereits in der Commitmentphase die vom Angreifer $\mathcal{A}$ gewählte Nachricht m^* extrahierte, können wir hier die Aufdeckungsphase ebenfalls simulieren, um $\mathcal{A}$ zu bewegen, seinen hinterlegten Wert m^* selbst zu offenbaren. Di Crescenzo, Katz, Ostrovsky und Smith [CKOS01] haben die Commitment-Schemata basierend auf dem Diskreten Logarithmus und RSA [FF00] mit der allgemeinen Konstruktion aus [DIO98] verbunden, und auf diese Weise nicht-interaktive, non-malleable Hinterlegungsverfahren beruhend auf diesen Komplexitätsannahmen erhalten. Wir geben in diesem Kapitel eine Vereinfachung der Konstruktion und ersetzen die RSA- durch die Faktorisierungsannahme [FF02]. Weil wie zuvor skizziert, der Non-Malleability-Beweis keinen Extraktor voraussetzt, entfällt die Unterscheidung zwischen RSA- und Faktorisierungsrepräsentation als Baustein der Schemata.

Abbildung 3.13 gibt das Schema basierend auf der Faktorisierungsannahme wieder. Für die Public-Parameter erweitere das Tupel $(N, \tau, t, g) \leftarrow \mathsf{FactIndex}(1^n)$ um zwei Elemente $h_0, h_1 \in_R \mathrm{HQR}_N$ sowie eine kollisionsresistente Hashfunktion $H : \mathbb{Z}_N^* \to \mathbb{Z}_{2^t}$. Die Hinterlegung besteht aus zwei verbundenen Commitments

$$X := g^m r^{2^{\tau+t}} \bmod N$$
$$M := (h_0^x h_1)^m s^{2^{\tau+2t}} \bmod N$$

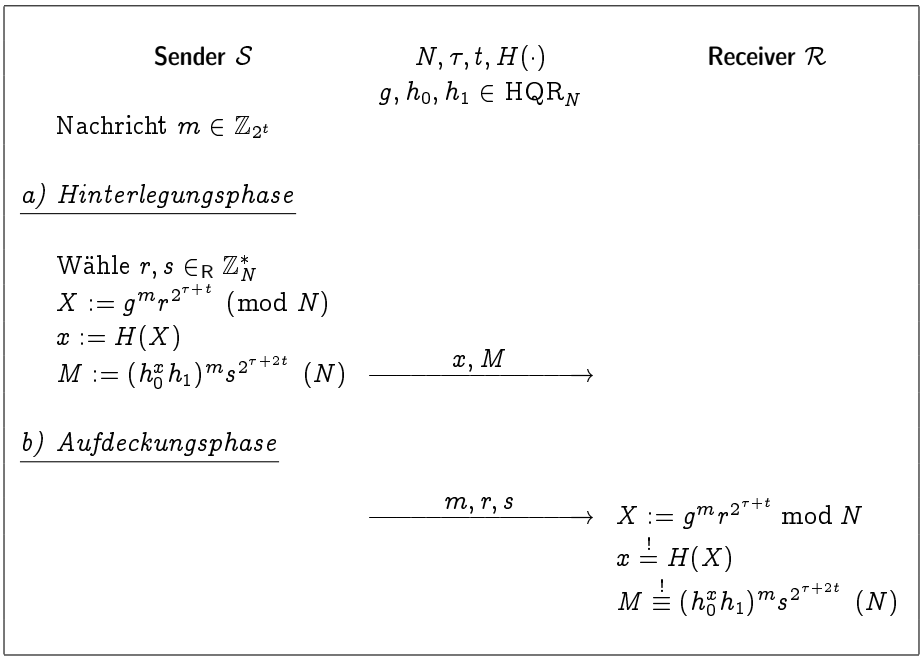
mit $x := H(X)$, und als Commitment schickt der Sender $\mathcal{S}$ das Paar (M, x). In der Aufdeckungsphase erhält der Receiver $\mathcal{R}$ dann die von $\mathcal{S}$ verwandten Werte und verifiziert, ob diese zur Hinterlegung passen. Offenbar bietet das Commitment-Schema perfekte Geheimhaltung sowie unter der Faktorisierungsannahme Polynomialzeit-Bindung.

Die beiden Hinterlegungen sind zwar einzeln betrachtet malleable, in Verbindung erreichen wir jedoch Non-Malleability. Beim nicht-interaktiven Commitment sieht der Angreifer $\mathcal{A}$ die Hinterlegung (x, M) von m durch den Sender $\mathcal{S}$ (d.h. $\mathcal{A}$ ist passiv), bevor er dann seinerseits eine in Relation R zu m stehende Nachricht m^* verbrieft. Im Fall eines erfolgreichen Angreifers $\mathcal{A}$ sind zwei Szenerien denkbar:

a) $\mathcal{A}$ ist in der Lage, sein Commitment (x^*, M^*) mehrdeutig aufzudecken, und auf diese Art und Weise nach Bekanntwerden der Hinterlegung m dann eine in Beziehung stehende Öffnung m^* zu generieren.

b) $\mathcal{A}$ kann zwar sein Commitment nicht mehrdeutig aufdecken, erzeugt aber eine Hinterlegung (x^*, M^*), die er mit einer in Relation zu m stehendenden Nachricht m^* öffnet.

Abbildung 3.13: nicht-interaktives non-malleable Commitment-Schema



Wir zeigen, dass in Fall a) man mit Hilfe von $\mathcal{A}$ den Modul effizient in die Primfaktoren zerlegen und als Konsequenz unter der Faktorisierungsannahme Situation b) unterstellen kann. $\mathcal{A}_{\mathsf{Adhoc}}$ legt dem simulierten $\mathcal{A}$ ein Pseudo-Commitment vor und gibt es als Hinterlegung der Nachricht m aus. Mit Hilfe einer Geheimtüre kann $\mathcal{A}_{\mathsf{Adhoc}}$ seine „Hinterlegung“ beliebig öffnen, und weil $\mathcal{A}$ — um überhaupt erfolgreich sein zu können — sein Commitment, an welches $\mathcal{A}$ nach Voraussetzung gebunden ist, aufdecken muß, erhält $\mathcal{A}_{\mathsf{Adhoc}}$ nach eventuell mehreren Wiederholungen eine in Relation zu m stehende Nachricht m^*.

Satz 3.5.1. *Die History-Funktion* Hist$(\cdot)$ *sei konstant für alle Nachrichten. Dann ist unter der Faktorisierungsannahme und der Voraussetzung einer kollisions-resistenten Hashfunktion das nicht-interaktive Hinterlegungsverfahren aus Abbildung 3.13 ein polynomialzeit-bindendes, bezüglich Aufdeckung ϵ-non-malleable Commitment-Schema mit perfekter Geheimhaltung.*

Beweis. Der Beweis folgt [CKOS01] angepaßt auf den Fall der Faktorisierungsrepräsentation [FF02]. Es ist zu zeigen, dass zu $\mathcal{A}$ ein Angreifer $\mathcal{A}_{\mathsf{Adhoc}}$

aus dem Stand existiert, für dessen Erfolgswahrscheinlichkeit gilt

$$\pi(\mathcal{A}) - \pi(\mathcal{A}_{\mathsf{Adhoc}}) \leq \epsilon - \nu \tag{3.9}$$

für ein vernachlässigbares ν. Zu gegebenem $(N, \tau, t, h_0) \leftarrow \mathsf{FactIndex}(1^n)$ und kollisions-resistenter Hashfunktion $H(\cdot)$ wähle $r, s, u, v \in_{\mathsf{R}} \mathbb{Z}_N^*$ und setze

$$\begin{aligned} g &:= u^{2^{\tau+t}} \bmod N \\ h_1 &:= h_0^{-x} v^{2^{\tau+2t}} \bmod N \\ M &:= s^{2^{\tau+2t}} \bmod N \\ X &:= r^{2^{\tau+t}} \bmod N \end{aligned} \tag{3.10}$$

zu $x := H(X)$. Beachte, dass diese Werte (3.10) unabhängig von einer Nachrichtenmenge M sind, weshalb man vom Angreifer in Relation zu den Public-Parametern vorgegebene Nachrichtenmenge M nicht gesondert behandeln muß. Als Public-Parameters σ werden $(N, \tau, t, H, g, h_0, h_1)$ vorgegeben, wobei deren Verteilung ebenso wie diejenige des Pseudo-Commitments (x, M) identisch mit einem realen Umfeld ist. Wir kennen in Form von v eine $2^{\tau+2t}$-te Wurzel von

$$h_0^x h_1 \equiv h_0^x \cdot h_0^{-x} v^{2^{\tau+2t}} \equiv v^{2^{\tau+2t}} \pmod N$$

und verfügen mit u ebenfalls über eine $2^{\tau+t}$-te Wurzel von X. Dieses Wissen versetzt uns in der Lage, X respektive M mehrdeutig und im Ergebnis auch das Commitment (x, M) mit jeder beliebigen Nachricht m' zu öffnen:

$$\begin{aligned} X &\equiv g^0 r^{2^{\tau+t}} \equiv g^{m'} (u^{-m'} r)^{2^{\tau+t}} \pmod N \\ M &\equiv (h_0^x h_1)^0 s^{2^{\tau+2t}} \equiv (h_0^x h_1)^{m'} (v^{-m'} s)^{2^{\tau+2t}} \pmod N \end{aligned}$$

Im allgemeinen sind bei der Aufdeckung von (X, M) die jeweils verwandten Zufallswerte durch die Nachricht m' nicht eindeutig bestimmt, denn es existieren mehrere $2^{\tau+t}$-te Wurzeln. Im vorliegenden Fall hängt unsere Aufdeckung neben der Nachricht m' auch von u, r und v, s ab, allerdings ist bei fixierten u, r und v, s unsere Öffnung eindeutig durch die Nachricht m' vorgegeben.

Die Trapdoor erlaubt zwar beliebiges Aufdecken, die perfekte Geheimhaltung bleibt aber unberührt. Wir legen $\mathcal{A}$ die Hinterlegung (x, M) als Festlegung auf die (unbekannte) Nachricht m vor, und als Folge gibt $\mathcal{A}$ seinerseits ein Commitment (x^*, M^*). Der Angreifer $\mathcal{A}$ ist erfolgreich, wenn er dieses nach Aufdeckung der Nachricht m durch ein m^* mit $(m, m^*) \in \mathsf{R}$ öffnet. Wir unterscheiden drei Fälle:

a) $X \equiv X^* \pmod N$

b) $X \not\equiv X^* \pmod N$ und $x = x^*$.

c) $X \not\equiv X^* \pmod N$ und $x \neq x^*$

Im Fall a) ist unter der Faktorisierungsannahme die Erfolgswahrscheinlichkeit von $\mathcal{A}$ vernachlässigbar, denn unterstellt $X \equiv X^* \pmod N$, erhält man durch einen erfolgreichen $\mathcal{A}$ über die Aufdeckungen der Hinterlegungen wegen $(m, m) \notin \mathsf{R}$ zwei verschiedene Repräsentationen (m, r) und (m^*, r^*) zu X und im Ergebnis nach Lemma 2.1.7 auf Seite 38 die Primfaktorzerlegung des Moduls N. Die Wahrscheinlichkeit von Fall b) ist aufgrund der Kollisions-Resistenz von $H(\cdot)$ ebenfalls zu vernachlässigen. Daher erreicht $\mathcal{A}_{\mathsf{Adhoc}}$ in den Situationen a) und b) ohne Hinterlegung bereits nahezu die gleiche Erfolgswahrscheinlichkeit wie $\mathcal{A}$.

Abbildung 3.14: Algorithmus $\mathcal{A}_{\mathsf{Adhoc}}$

1. Wähle Public-Parameters $(N, \tau, t, H, g, h_0, h_1)$ und Pseudo-Commitment (x, M) gemäß (3.10)
2. Fixiere Zufallsbits ω von $\mathcal{A}$
3. Sei (x^*, M^*) Commitment von $\mathcal{A}$
4. Falls $(X \equiv X^*)$ oder $(X \not\equiv X^*$ und $x = x^*)$, dann stoppe ohne Ausgabe
5. Wiederhole bis zu $4\epsilon^{-1} \ln(2\epsilon^{-1})$-mal:
   i. Decke (x, M) mit $m' \in_{\mathsf{R}} \mathsf{M}$ und passendem r' auf
   ii. Sei (m^*, r^*) Öffnung von $\mathcal{A}$ zu (x^*, M^*)
   iii. Bei korrekter Aufdeckung von $\mathcal{A}$ gehe zu 6, anderenfalls setze $\mathcal{A}$ an Beginn der Aufdeckungsphase zurück.
6. Gib Commitment zu m^* aus

Abbildung 3.14 zeigt die Konstruktion des Angreifers $\mathcal{A}_{\mathsf{Adhoc}}$ basierend auf $\mathcal{A}$. Wir schreiben (x, M) für das $\mathcal{A}$ vorgelegte Commitment von $\mathcal{S}$ und (x^*, M^*) für die Hinterlegung seitens $\mathcal{A}$. Sei Γ die Menge der Tupel $\gamma := (\sigma, r, s, \omega, (x, M))$, für welche $\mathcal{A}$ mit fixierten Zufallsbits ω sein Commitment (x^*, M^*) zu (x, M) nach der Aufdeckung mit $m \in_{\mathsf{R}} \mathsf{M}$ mit Wahrscheinlichkeit $\frac{1}{2}\epsilon$ korrekt öffnet. Bei fixierten γ hängt die entsprechende Nachricht m^* nur von (x, M) und m ab, wir schreiben $m^* := \mathcal{A}_\omega((x, M), m)$. Bezeichne $\mathcal{A}^{\mathsf{sim}}_\omega((x, M), m')$ die in der Simulation 3.14 aufgedeckte Nachricht m^*, welche der Adhoc-Angreifer $\mathcal{A}_{\mathsf{Adhoc}}$ übernimmt. Falls $\mathcal{A}_{\mathsf{Adhoc}}$ ohne Ausgabe

stoppt, setze $m^* := \perp$. Es gilt:

$$\pi(\mathcal{A}) - \pi(\mathcal{A}_{\mathsf{Adhoc}}) = \mathrm{Ws}\left[\begin{array}{l} m^* := \mathcal{A}_\omega((x,M),m) \\ (m,m^*) \in \mathsf{R} \\ \gamma \in \Gamma \end{array}\right] + \mathrm{Ws}\left[\begin{array}{l} m^* := \mathcal{A}_\omega((x,M),m) \\ (m,m^*) \in \mathsf{R} \\ \gamma \notin \Gamma \end{array}\right]$$
$$- \mathrm{Ws}\left[\begin{array}{l} m^* := \mathcal{A}^{\mathsf{sim}}_\omega((x,M),m') \\ (m,m^*) \in \mathsf{R} \\ \gamma \in \Gamma \end{array}\right] - \mathrm{Ws}\left[\begin{array}{l} m^* := \mathcal{A}^{\mathsf{sim}}_\omega((x,M),m') \\ (m,m^*) \in \mathsf{R} \\ \gamma \notin \Gamma \end{array}\right],$$

wobei sich sämtliche Wahrscheinlichkeiten auf die Wahl von γ und $m, m' \in_{\mathsf{R}} \mathsf{M}$ beziehen. Für $\gamma \notin \Gamma$ steht — nach Definition der Menge Γ — die Nachricht $\mathcal{A}_\omega((x,M),m)$ höchstens mit Wahrscheinlichkeit $\frac{1}{2}\epsilon$ in Relation zu m:

$$\pi(\mathcal{A}) - \pi(\mathcal{A}_{\mathsf{Adhoc}}) \leq \mathrm{Ws}\left[\begin{array}{l} m^* := \mathcal{A}_\omega((x,M),m) \\ (m,m^*) \in \mathsf{R} \\ \gamma \in \Gamma \end{array}\right] + \tfrac{1}{2}\epsilon$$
$$- \mathrm{Ws}\left[\begin{array}{l} m^* := \mathcal{A}^{\mathsf{sim}}_\omega((x,M),m') \\ (m,m^*) \in \mathsf{R} \\ \gamma \in \Gamma \end{array}\right] - 0.$$

Dies können wir schreiben als:

$$\pi(\mathcal{A}) - \pi(\mathcal{A}_{\mathsf{Adhoc}}) \leq \mathrm{Ws}\left[\begin{array}{ll} m_1^* := \mathcal{A}_\omega((x,M),m), & (m,m_1^*) \in \mathsf{R} \\ m_2^* := \mathcal{A}^{\mathsf{sim}}_\omega((x,M),m'), & (m,m_2^*) \notin \mathsf{R} \\ \gamma \in \Gamma & \end{array}\right] + \tfrac{1}{2}\epsilon.$$

Vorausgesetzt $\gamma \in \Gamma$, ist die Wahrscheinlichkeit, dass es bei allen $k := 4\epsilon^{-1}\ln(2\epsilon^{-1})$ Wiederholungen keine gültige Aufdeckung gibt, also dementsprechend $m_2^* = \perp$ gilt, gemäß Chernoff-Schranke (1.4) von Seite 14 (mit $p = \frac{1}{2}\epsilon$ und $\delta = \frac{1}{2}$) höchstens $\frac{1}{2}\epsilon$:

$$\pi(\mathcal{A}) - \pi(\mathcal{A}_{\mathsf{Adhoc}}) \leq \epsilon + \underbrace{\mathrm{Ws}\left[\begin{array}{ll} m_1^* := \mathcal{A}_\omega((x,M),m), & (m,m_1^*) \in \mathsf{R} \\ m_2^* := \mathcal{A}^{\mathsf{sim}}_\omega((x,M),m'), & (m,m_2^*) \notin \mathsf{R} \\ \gamma \in \Gamma, & m_2^* \neq \perp \end{array}\right]}_{=:\nu}.$$

Wir zeigen, dass ν unter der Faktorisierungsannahme vernachlässigbar ist, was aufgrund Abschätzung (3.9) dann den Beweis vervollständigt. Es gilt $m_1^* \neq m_2^*$, was in Verbindung mit $m_2^* \neq \perp$ aus $(m,m_1^*) \in \mathsf{R}$ und $(m,m_2^*) \notin \mathsf{R}$ folgt. Als Konsequenz erhalten wir

$$(h_0^{x^*} h_1)^{m_1^*} (r_1^*)^{2^{\tau+2t}} \equiv M^* \equiv (h_0^{x^*} h_1)^{m_2^*} (r_2^*)^{2^{\tau+2t}} \pmod{N}$$

und Einsetzen von $h_1 \equiv h_0^{-x} v^{2^{\tau+2t}} \pmod{N}$ liefert:

$$h_0^{(x^*-x)m_1^*} (v^{m_1^*} r_1^*)^{2^{\tau+2t}} \equiv M^* \equiv h_0^{(x^*-x)m_2^*} (v^{m_2^*} r_2^*)^{2^{\tau+2t}} \pmod{N}.$$

Mit $\Delta x := x^* - x$ und $\Delta m^* = m_1^* - m_2^*$ können wir dies schreiben als:

$$h_0^{\Delta x \cdot \Delta a^*} \equiv (v^{-\Delta a^*}(r_1^*)^{-1} r_2^*)^{2^{\tau+2t}} \pmod{N}.$$

Aufgrund der Annahme $x \neq x^*$ ist $1 \leq |\Delta x| < 2^t$, wegen $m_1^* \neq m_2^*$ und $0 \leq m_1^*, m_2 < 2^t$ folgt $1 \leq |\Delta m^*| < 2^t$, so dass insgesamt gilt:

$$1 \leq |\Delta x \cdot \Delta m^*| < 2^t \cdot 2^t = 2^{2t}.$$

Wir verfügen über zwei verschiedene Repräsentationen von M^* bezüglich $(N, \tau, 2t, h_0)$. Diese Möglichkeit, das Faktorisierungsrepräsentationsproblem zu $(N, \tau, 2t, h_0) \leftarrow \mathsf{FactIndex}(1^n)$ in Polynomialzeit mit nicht vernachlässigbarer Wahrscheinlichkeit zu lösen, gestattet nach Satz 2.1.8 von Seite 39, die Primfaktoren des Moduls N effizient zu bestimmen. Unter der Faktorisierungsannahme ist daher die Wahrscheinlichkeit ν vernachlässigbar. □

Das Schema in Abbildung 3.13 auf Seite 84 verwendet eine kollisionsresistente Hashfunktion $H(\cdot)$. Diese können wir ebenfalls durch eine Familie universeller Oneway-Hashfunktionen ersetzen, alternativ fordere $t \geq n$ und verwende X direkt statt des Hashwertes als Exponent:

$$X := g^m r^{2^{\tau+t}} \bmod N$$
$$M := (h_0^X h_1)^m s^{2^{\tau+2t}} \bmod N$$

Für den Sicherheitsbeweis beachte, dass $|X|, |X^*| < N < 2^t$ sind. Weil allerdings X deutlich größer als sein Hashwert x ist, steigt der Aufwand der Berechnung von h_0^X spürbar gegenüber h_0^x an. Ersetzt man die Exponenten $2^{\tau+t}$ und $2^{\tau+2t}$ durch einen primen RSA-Exponenten, ist die Sicherheit des Commitment-Verfahrens auf RSA reduzierbar.

Index

Literaturverzeichnis

[AR98] Y. Aumann und M.O. Rabin: *Proof of Plaintext Knowledge and Chosen Ciphertext Attacks*, Manuskript, Invited Talk auf Cypto '98 (nicht identisch mit der im Konferenzband abgedruckten Zusammenfassung).

[Be96] D. Beaver: *Adaptive Zero-Knowledge and Computional Equivocation*, Proceedings des 28. jährlichen ACM Symposium on Theory of Computing (STOC), Seiten 629–638, ACM Press, New York, 1996.

[BG92] M. Bellare und O. Goldreich: *On Defining Proofs of Knowledge*, Advances in Cryptology — Proceedings Crypto '92, Lecture Notes in Computer Science, Band 740, Seiten 390–420, Springer-Verlag, Berlin/Heidelberg, 1993.

[BR93] M. Bellare und P. Rogaway: *Random Oracles are Practical: A Paradigm for Designing Efficient Protocols*, ACM Conference on Computer and Communication Security, Seiten 62–73, ACM Press, New York, 1993.

[Bl82] M. Blum: *Coin Flipping by Telephone: A Protocol for Solving Impossible Problems*, 24-te IEEE Spring Computer Conference, Seiten 133–137, IEEE, Los Alamitos, 1982.

[BV98] D. Boneh und R. Venkatesan: *Breaking RSA may not be Equivalent to Factoring*, Advances in Cryptology — Proceedings Eurocrypt '98, Lecture Notes in Computer Science, Band 1403, Seiten 59–71, Springer-Verlag, Berlin/Heidelberg, 1998.

[B99] D. Boneh: *Twenty Years of Attacks on the RSA Cryptosystem*, Notices of the American Mathematical Society (AMS), Band 46, Nr. 2, Seiten 203–213, 1999.

[B00] D. Boneh und G. Durfee: *Cryptanalysis of RSA With Private Key d Less Than $N^{0.292}$*, IEEE Transactions on Information Theory, Band 46, Nr. 4, Seiten 1339–1349, 2000.

[B00] D. Boneh: *Finding Smooth Integers Using CRT Decoding*, Proceedings des 32. jährlichen ACM Symposium on Theory of Computing (STOC), ACM Press, New York, 2000. Neuere Version erhältlich auf Bonehs Homepage http://crypto.Stanford.EDU/~dabo/.

[B97] S. Brands: *Rapid Demonstration of Linear Relations Connected by Boolean Operators*, Advances in Cryptology — Proceedings Eurocrypt '97, Lecture Notes in Computer Science, Band 1233, Seiten 318–333, Springer-Verlag, Berlin/Heidelberg, 1997.

[BCC88] G. Brassard, D. Chaum und C. Crépeau: *Minimum Disclosure Proofs of Knowledge*, Journal Computing System Science, Band 37, Nr. 2, Seiten 156–189, 1988.

[Ch82] D. Chaum: *Blind Signatures for Untraceable Electronic Cash*, Advances in Cryptology — Proceedings Crypto '82, Seiten 199–203, Plenum, New York, 1983.

[Co97] D. Coppersmith: *Small Solutions to Polynomial Equations and low Exponent RSA Vulnerabilities*, Journal of Cryptology, Band 10, Seiten 233–260, 1997.

[CrDa97] R. Cramer und I. Damgård: *Fast and Secure Immunization Against Adaptive Man-In-The-Middle Impersonation*, Advances in Cryptology — Proceedings Eurocrypt '97, Lecture Notes in Computer Science, Band 1233, Seiten 75–87, Springer-Verlag, Berlin/Heidelberg, 1997.

[CrSh00] R. Cramer und V. Shoup: *Signature Schemes Based on the Strong RSA Assumption*, ACM Transactions on Information and System Security (ACM TISSEC), Band 3, Nr, 3, Seiten 161–185, 2000.

[CP01] R. Crandall und C. Pomerance: **Prime Numbers: A Computational Perspective**, Springer-Verlag, Berlin/Heidelberg, 2001.

[CKOS01] G. Di Crescenzo, J. Katz, R. Ostrovsky und A. Smith: *Efficient and Non-Interactive Non-Malleable Commitment*, Advances in Cryptology — Proceedings Eurocrypt '01, Lecture

Notes in Computer Science, Band 2045, Seiten 40–59, Springer-Verlag, Berlin/Heidelberg, 2001.

[D95] I. DAMGÅRD: *Practical and Provably Secure Release of a Secret and Exchange of Signature*, Journal of Cryptology, Band 8, Seiten 201–222, 1995.

[DIO98] G. DI CRESCENZO, Y. ISHAI und R. OSTROVSKY: *Non-interactive and Non-Malleable Commitment*, Proceedings des 30. jährlichen ACM Symposium on Theory of Computing (STOC), Seiten 141–150, ACM Press, New York, 1998.

[DH76] W.DIFFIE und M.HELLMAN: *New Directions in Cryptography*, IEEE Transaction on Information Theory, Band 22, Seiten 644–654, 1976.

[DDN00] D. DOLEV, C. DWORK und M. NAOR: *Nonmalleable Cryptography*, SIAM Journal on Computing, Band 30, Nr. 2, Seiten 391-437, 2000. Erste Version in Proceedings des 21. jährlichen ACM Symposium on Theory of Computing (STOC), Seiten 542–552, ACM Press, New York, 1991.

[FFS88] U. FEIGE,A. FIAT und A. SHAMIR: *Zero-Knowledge Proofs of Identity*, Journal of Cryptology, Band 1, Nr. 2, Seiten 77–94, 1988.

[FS86] A. FIAT und A. SHAMIR: *How to Prove Yourself: Practical Solutions to Identification and Signature Schemes*, Advances in Cryptology — Proceedings Crypto '86, Lecture Notes in Computer Science, Band 263, Seiten 186–194, Springer-Verlag, Berlin/Heidelberg, 1986.

[FS90] U. FEIGE und A. SHAMIR: *Witness Indistinguishable and Witness Hiding Protocols*, Proceedings des 22. jährlichen ACM Symposium on Theory of Computing (STOC), Seiten 416–426, ACM Press, New York, 1990.

[F01] M. FISCHLIN: *Trapdoor Commitment Schemes and Their Applications*, Dissertation am Fachbereich Mathematik, J.W.-Goethe-Universität Frankfurt/Main, 2001.

[FF00] M. FISCHLIN und R. FISCHLIN: *Efficient Non-Malleable Commitment Schemes*, Advances in Cryptology — Proceedings Crypto 2000, Lecture Notes in Computer Science,

Band 1880, Seiten 414–432, Springer Verlag, Berlin/Heidelberg, 2000. Journal-Version in Vorbereitung.

[FF02] M. Fischlin und R. Fischlin: *The Representation Problem Based on Factoring*, Cryptographer's Track, RSA Conference 2002, Lecture Notes in Computer Science, Band 2271, Seiten 96–113, Springer-Verlag, Berlin/Heidelberg, 2002.

[GeHR99] R. Gennaro, S. Halevi und T. Rabin: *Secure Hash-and-Sign Signatures Without the Random Oracle*, Advances in Cryptology — Proceedings Eurocrypt '99, Lecture Notes in Computer Science, Band 1592, Seiten 123–139, Springer-Verlag, Berlin/Heidelberg, 1999.

[Gi91] M. Girault: *Self-Certified Public Keys*, Advances in Cryptology — Proceedings Eurocrypt '91, Lecture Notes in Computer Science, Band 547, Seiten 490–497, Springer-Verlag, Berlin/Heidelberg, 1992.

[Go98] O. Goldreich: **Foundations of Cryptography (Fragments of a Book)** Version 2.03, Department of Computer Science and Applied Mathematics, Weizmann Institute of Science, Rehovot, Israel, 1998. Im WWW erhältlich auf Goldreichs Homepage `http://www.wisdom.weizmann.ac.il/home/oded/public_html/index.html`.

[GoRoSu99] O. Goldreich, D. Ron und M. Sudan: *Chinese Remainder With Errors*, Proceedings des 31. jährlichen ACM Symposium on Theory of Computing (STOC), Seiten 225–234, ACM Press, New York, 1999.

[GMR89] S. Goldwassser, S. Micali und C. Rackoff: *The Knowledge Complexity of Interactive Proof Systems*, SIAM Journal on Computation, Band 18, Seiten 186–208, 1989.

[GMRi88] S. Goldwassser, S. Micali und R.L. Rivest: *A Digital Signature Schema Secure Against Adaptive Chosen-Message Attacks*, SIAM Journal on Computing, Band 17, Nr. 2, Seiten 186–208, 1988.

[GuQu88] L.C. Guillou und J.-J. Quisquater: *A Practical Zero-Knowledge Protocol Fitted to Security Microprocessors Minimizing Both Transmission and Memory*, Advances in Cryptology — Proceedings Eurocrypt '88, Lecture Notes in

Computer Science, Band 330, Seiten 123–129, Springer-Verlag, Berlin/Heidelberg, 1988.

[HaMi98] S. Halevi und S. Micali: *More On Proofs Of Knowledge*, Manuskript, IACR Cryptology ePrint Archive 1998/015, 1998. Im WWW auf der ePrint-Server `http://eprint.iacr.org/` erhältlich.

[Ha99] S. Halevi: *Efficient Commitment Schemes With Bounded Sender and Unbounded Receiver*, Journal of Cryptology, Band 12, Nr. 2, Seiten 77–90, 1999.

[HILL99] J. Haståd, R. Impagliazzo, L. Levin und M. Luby: *A Pseudorandom Generator From Any One-Way Function*, SIAM Journal on Computing, Band 28, Nr. 4, Seiten 1364–1396, 1999.

[JLO97] A. Juels, M. Luby und R. Ostrovsky: *Security of Blind Digital Signatures*, Advances in Cryptology — Proceedings Crypto '97, Lecture Notes in Computer Science, Band 1294, Seiten 150–164, Springer-Verlag, Berlin/Heidelberg, 1997.

[K98] D. Knuth: **Seminumerical Algorithms**, The Art of Computer Programming, Band 2, 3.te Auflage, Addison Wesley, Reading, 1998.

[KR99] H. Krawczyk und T. Rabin: *Chameleon Hashing and Signatures*, Network and Distributed System Security Symposium 2000, Seiten 143–154, 2000.

[LL93] A.K. Lenstra und H.W. Lenstra, Jr. (Hrsg.): **The Development of the Number Field Sieve**, Lecture Notes on Mathematics, Band 1554, Springer-Verlag, Berlin/Heidelberg, 1993.

[LV01] A.K. Lenstra und E.R. Verheul: *Selecting Cryptographic Key Size*, Journal of Cryptology, Band 14, Nr. 4, Seiten 255-293, 2001.

[Lu96] M. Luby: **Pseudorandomness and Cryptographic Applications**, Princeton Computer Science Notes, Princeton University Press, Princeton, 1996.

[M95] U. Maurer: *Fast Generation of Prime Numbers and Secure Public-Key Cryptographic Parameters*, Journal of Cryptology, Band 8, Seiten 123–155, Springer-Verlag, 1995.

[MOV97] A.J. MENEZES, P.C. VAN OORSCHOT und S.A. VANSTONE: **Handbook of Applied Cryptography**, CRC Press Series on Discrete Mathemathics and its Applications, CRC Press, Boca Raton, 1997.

[MoRa95] R. MOTWANI und P. RAGHAVAN: **Randomized Algorithms**, Cambridge University Press, New York, 1995.

[N91] M. NAOR: *Bit Commitment Using Pseudo-Randomness*, Journal of Cryptology, Band 4, Seiten 151–158, 1991.

[NY90] M. NAOR und M. YUNG: *Universal Oneway Hash Functions and Their Cryptographic Applications*, Proceedings des 21. jährlichen ACM Symposium on the Theory of Computing (STOC), Seiten 33–43, ACM Press, New York, 1989.

[OS90] H. ONG und C.P. SCHNORR: *Fast Signature Generation With a Fiat-Shamir-Like Scheme*, Advances in Cryptology — Proceedings Eurocrypt '90, Lecture Notes in Computer Science, Band 473, Seiten 432–440, Springer-Verlag, Berlin/Heidelberg, 1991.

[OO88] K. OHTA und T. OKAMOTO: *A Modification of The Fiat-Shamir Scheme*, Advances in Cryptology — Proceedings Crypto '88, Lecture Notes in Computer Science, Band 403, Seiten 232–243, Springer-Verlag, Berlin/Heidelberg, 1989.

[Ok92] T. OKAMOTO: *Provable Secure and Practical Identification Schemes and Corresponding Signature Schemes*, Advances in Cryptology — Proceedings Crypto '92, Lecture Notes in Computer Science, Band 740, Seiten 31–53, Springer-Verlag, Berlin/Heidelberg, 1993.

[Pe91] T. PEDERSEN: *Non-Interactive and Information-Theoretical Secure Verifiable Secret Sharing*. Advances in Cryptology — Proceedings of Crypto '91, Lecture Notes in Computer Science, Band 576, Seiten 129–140, Springer-Verlag, Berlin/Heidelberg, 1991.

[PSt96a] D. POINTCHEVAL und J. STERN: *Security Proofs for Signature Schemes*, Advances in Cryptology — Proceedings of Eurocrypt '96, Lecture Notes in Computer Science, Band 1070, Seiten 387–398, Springer-Verlag, Berlin/Heidelberg, 1996.

[PSt96b] D. POINTCHEVAL und J. STERN: *Security Proofs for Signature Schemes*, Advances in Cryptology — Proceedings of Asiacrypt '96, Lecture Notes in Computer Science, Band 1163, Seiten 252–265, Springer-Verlag, Berlin/Heidelberg, 1996.

[PSt97] D. POINTCHEVAL und J. STERN: *New Blind Signatures Equivalent to Factorization*, Proceedings der 4. ACM Conference on Computer and Communications Security (CCS) '97, Seiten 92-99, ACM Press, New York, 1997.

[PSt00] D. POINTCHEVAL und J. STERN: *Security Arguments for Digital Signatures and Blind Signatures*, Journal of Cryptology, Band 13, Nr. 3, Seiten 361–396, 2000. Journal-Version u.a. zu [PSt96a, PSt96b].

[P00] D. POINTCHEVAL: *The Composite Discrete Logarithm and Secure Authentication*, Workshop on Practice and Theory in Public Key Cryptography (PKC) 2000, Lecture Notes in Computer Science, Band 1751, Seiten 113–128, Springer-Verlag, Berlin/Heidelberg, 2000.

[PoSt98] G. POUPARD und J. STERN: *Security Analysis of a Practical "On The Fly" Authentication and Signature Generation*, Advances in Cryptology — Proceedings of Eurocrypt'98, Lecture Notes in Computer Science, Band 1403, Seiten 422–436, Springer-Verlag, Berlin/Heidelberg, 1998.

[PoSt99] G. POUPARD und J. STERN: *On The Fly Signatures Based On Factoring*, Proceedings der 6. ACM Conference on Computer and Communications Security (CCS) '99, Seiten 37–45, ACM Press, New York, 1999.

[PoSt00] G. POUPARD und J. STERN: *Short Proofs of Knowledge for Factoring*, Workshop on Practice and Theory in Public Key Cryptography (PKC) 2000, Lecture Notes in Computer Science, Band 1751, Seiten 147—166, Springer-Verlag, Berlin/Heidelberg, 2000.

[R78] M.O. RABIN: *Digitalized Signatures and Public Key Functions as Intractable as Factorization*, Technical Report Nr. 212, MIT Laboratory for Computer Science, 1978.

[RSA78] R.L. RIVEST, A. SHAMIR und L. ADLEMAN: *A Method for Obtaining Digital Signatures and Public-Key Cryptosystems*, Communications of the ACM, Band 21, Seiten 120–126, 1978.

[Ro90] J. ROMPEL: *One-Way Functions Are Necessary and Sufficient for Digital Signatures*, Proceedings des 22. jährlichen ACM Symposium on Theory of Computing (STOC), Seiten 387–394, ACM Press, New York, 1990.

[S91] C.P. SCHNORR: *Efficient Signature Generation By Smart Cards*, Journal of Cryptology, Band 4, Seiten 161–174, 1991.

[S96] C.P. SCHNORR: *Security of 2^t-Root Identification and Signatures*, Advances in Cryptology — Proceedings Crypto '96, Lecture Notes in Computer Science, Band 1109, Seiten 143–156, Springer-Verlag, Berlin/Heidelberg, 1996.

[S97] C.P. SCHNORR: *Erratum: Security of 2^t-Root Identification and Signatures*, in Proceedings Crypto '97, Lecture Notes in Computer Science, Band 1294, Seite 540, Springer-Verlag, Berlin/Heidelberg, 1997.

[Sh99] V. SHOUP: *On the Security of a Practical Identification Scheme*, Journal of Cryptology, Band 12, Seiten 247–260, 1999.

[Si98] D.R. SIMON: *Finding Collisions on a Oneway Street: Can Secure Hash Functions be Based on General Assumptions*, Advances in Cryptology — Proceedings Eurocrypt '98, Lecture Notes in Computer Science, Band 1403, Seiten 334–345, Springer-Verlag, Berlin/Heidelberg, 1998.

[Su97a] M. SUDAN: *Decoding of Reed Solomon Codes Beyond the Error-Correction Bound*, Journal of Complexity, Band 13, Nr. 1, Seiten 180—193, 1997.

[Su97b] M. SUDAN: *Decoding Reed-Solomon Codes Beyond the Error-Correction Diameter*, Proceedings der 35. jährlichen Allerton Conference on Communication, Control and Computing, Seiten 215–224, 1997. Im WWW erhältlich auf Sudans Homepage `http://theory.lcs.mit.edu/~madhu/`

[Su01] M. SUDAN: *Ideal Error-Correcting Codes: Unifying Algebraic And Number-Theoretic Algorithms*, Proceedings des 14. Symposium on Applied Algebra, Algebraic Algorithms and Error-Correcting (AAECC-14), Lecture Notes in Computer Science, Band 2227, Seiten 36–45, Springer-Verlag, Berlin/Heidelberg, 2001.

[W90] M. WIENER: *Cryptanalysis of Short RSA Secret Exponents*, IEEE Transactions on Information Theory, Band 36, Seiten 553–558, 1990.

[W96] J. WOLFART: **Einführung in die Zahlentheorie und Algebra**, Vieweg-Verlag, Braunschweig/Wiesbaden, 1996.

Lebenslauf

persönliche Daten

Name	Roger Fischlin
Geburtsdatum	04. April 1971
Geburtsort	Offenbach/Main
Staatsbürgerschaft	deutsch
Familienstand	ledig, keine Kinder

schulische Ausbildung

Sept. 1977-Juni 1990	Grundschule, Förderstufe, Gymnasium (Karl-Rehbein-Schule Hanau) Allgemeine Hochschulreife mit Note 1,5 (Leistungskurse Mathematik und Englisch)

universitäre Ausbildung

Okt. 1991–Dez. 1998	Informatik-Studium, Goethe-Universität Frankfurt Nebenfach Mathematik, zusätzl. Nebenfach BWL Diplom mit Gesamtnote „sehr gut“, Diplomarbeit: Lineare Programmierung Betreuer Prof. Dr. Schnitger
April 1995–Dez. 1997	Mathematik-Studium, Goethe-Universität Frankfurt Nebenfach Informatik Diplom mit Gesamturteil „mit Auszeichnung“, Diplomarbeit: kryptogr. Pseudozufallsgeneratoren Betreuer Prof. Dr. Schnorr
ab April 1999	Promotionsstudium, Goethe-Universität Frankfurt

Berufserfahrungen

Okt. 1994–Jan. 1998	Studentische Hilfskraft an Fachbereichen Mathematik und Informatik der Goethe-Universität Frankfurt (mit Unterbrechungen)
ab Februar 1998	Wissenschaftlicher Angestellter, Arbeitsgruppe von Prof. Dr. Schnorr am Fachbereich Mathematik der Goethe-Universität Frankfurt

Sonstiges

Juli 1990–Sept. 1991	Grundwehrdienst/Zivildienst