

Faktorisierungsproblem, Protokolle

1. RSA-Schema

RSA ist sowohl ein Verschlüsselungs- als auch ein Signatur-Schema. Die Bezeichnung RSA leitet sich von den drei Erfindern RIVEST, SHAMIR und ADLEMAN ab [RSA78].

1.1. Schlüsselpaar. Historisch entstand RSA beim Versuch, das Diffie-Hellman-Schema auf Gruppen zu erweitern, deren Ordnung geheim oder unbekannt ist. Ist diese Ordnung ein Teil des geheimen Schlüssels, muss jeder Teilnehmer eine eigene Gruppe verwenden. Im Fall des RSA-Schemas ist dies die Gruppe \mathbb{Z}_N^* für einen RSA-Modul N :

Definition 3.1 (RSA-Modul). *Ein RSA-Modul N ist das Produkt zweier verschiedener Primzahlen $p, q > 2$.*

Mit Aufkommen der modernen Kryptographie haben zahlreiche Forscher an schnellen Faktorisierungsalgorithmen gearbeitet, ohne jedoch effiziente Algorithmen zu finden:

Annahme 3.2 (Faktorisierungsannahme). *RSA-Module $N = pq$ sind effizient nicht zu faktorisieren, wenn p, q hinreichend groß (heute $p, q \geq 2^{1024}$) sind und $p \pm 1, q \pm 1$ jeweils mindestens einen großen Primfaktor (heute $\geq 2^{80}$) haben.*

Nach Chinesischem Restsatz gilt

$$\mathbb{Z}_N^* \simeq \mathbb{Z}_p^* \times \mathbb{Z}_q^*$$

so dass die Anzahl der Gruppenelemente von \mathbb{Z}_N^* gleich (Euler-Funktion)

$$\varphi(N) := |\mathbb{Z}_N^*| = |\mathbb{Z}_p^*| \cdot |\mathbb{Z}_q^*| = \varphi(p) \cdot \varphi(q) = (p-1)(q-1)$$

ist. Gemäß Satz von LAGRANGE gilt daher

$$(1) \quad w^{|\mathbb{Z}_N^*|} = w^{\varphi(N)} = 1 \quad \text{für alle } w \in \mathbb{Z}_N^*.$$

Die Gruppe \mathbb{Z}_N^* ist (im Gegensatz zu \mathbb{Z}_p^* und \mathbb{Z}_q^*) nicht zyklisch. Die maximale Ordnung¹ ist (Carmichael-Funktion)

$$\lambda(N) = \text{kgV}(\varphi(p), \varphi(q)) = \text{kgV}(p-1, q-1).$$

Es gilt $\lambda(N) | \varphi(N)$. Da $p-1$ und $q-1$ gerade Zahlen sind, ist $\lambda(N) \leq \frac{1}{2}\varphi(N)$.

Definition 3.3 (RSA-Exponent). Ein RSA-Exponent e zu einem RSA-Modul N ist eine Zahl $e \in \mathbb{Z}_{\varphi(N)}^*$ mit $e \neq 1 \pmod{\varphi(N)}$.

Definition 3.4 (RSA-Schlüssel). Sowohl Chiffrier- als auch Signatur-Schema verwenden als Schlüsselpaar:

- Public-Key (N, e) : zufälliger RSA-Modul N und Exponent e zu N .
- Secret-Key d : $d := e^{-1} \pmod{\varphi(N)}$ bzw. Faktorisierung von N .

Der Nachrichtenraum ist \mathbb{Z}_N^* .

In einer Übungsaufgabe zeigen wir, wer über $\varphi(N)$ oder d verfügt, kennt auch die Primfaktoren des RSA-Moduls N .

1.2. Verschlüsselungs-Schema. Um eine Nachricht $m \in \mathbb{Z}_N^*$ zu chiffrieren, berechne

$$c := \text{RSA}_{N,e}(m) := m^e \pmod{N},$$

d.h. die Nachricht wird in die e -te Potenz modulo N erhoben. Um den Ciphertext c zu dekodieren, ziehe die e -te Wurzel oder äquivalent erhebe c in die d -te Potenz:

$$c^d = \text{RSA}_{N,e}(c)^d = m^{ed} = m$$

Zu zeigen bleibt $m^{ed} = m$. Nach Wahl von $d := e^{-1} \pmod{\varphi(N)}$ gilt $ed = 1 \pmod{\varphi(N)}$, d.h. es existiert ein $k \in \mathbb{Z}$ mit $ed + k \cdot \varphi(N) = 1$. Aus Satz von Lagrange (1) folgt:

$$m^{ed} = m^{1-k\varphi(N)} = m \cdot \underbrace{(m^{-k})^{\varphi(N)}}_{\in \mathbb{Z}_N^*} = m$$

Die inverse Abbildung $\text{RSA}_{N,e}^{-1}$ ist gegeben durch $\text{RSA}_{N,d}$.

¹Die Ordnung $\text{ord}_N(a)$ von $a \in \mathbb{Z}_N^*$ ist das minimale $k \geq 1$ mit $a^k = 1$. Es gilt $\lambda(N) = \max \{ \text{ord}_N(a) \mid a \in \mathbb{Z}_N^* \}$.

Die Berechnung der $\text{RSA}_{N,e}$ -Funktion ist aufwändig, im Vergleich zu Diskreten-Log-Schemata ist die Bitlänge deutlich größer. Ein beliebiger, universeller RSA-Exponent ist

$$e := 2^{2^4} + 1 = 65537,$$

da 17 Multiplikationen (in \mathbb{Z}_N^*) zur Verschlüsselung ausreichen.

Trotz intensiver Forschung [Sil97] kennt man bis heute neben der Faktorisierung des RSA-Moduls N nur einen weiteren, allgemeinen Angriff auf das RSA-Schema: WIENER [W90] hat gezeigt, dass für $d \leq N^{\frac{1}{4}}$ die Kettenbruchentwicklung zu $\frac{e}{N}$ in Polynomialzeit den geheimen Schlüssel d liefert:

Annahme 3.5 (RSA-Annahme). Für einen RSA-Modul N und einen zugehörigen RSA-Exponenten e ist die $\text{RSA}_{N,e}$ -Funktion effizient nicht zu invertieren, sofern der Modul N die in der Faktorisierungsannahme 3.2 genannten Eigenschaften aufweist und $(e^{-1} \bmod \varphi(N)) > N^{\frac{1}{4}}$ ist.

Die $\text{RSA}_{N,e}$ -Funktion ist ein Homomorphismus, d.h. es gilt

$$\begin{aligned} \text{RSA}_{N,e}(m_1) \cdot \text{RSA}_{N,e}(m_2) &= m_1^e m_2^e \\ &= (m_1 m_2)^e \\ &= \text{RSA}_{N,e}(m_1 m_2) \end{aligned}$$

für alle $m_1, m_2 \in \mathbb{Z}_N^*$. Da eine inverse Funktion zu $\text{RSA}_{N,e}$ existiert, folgt:

Satz 3.6. Sei N ein RSA-Modul und e ein zugehöriger RSA-Exponent. Dann ist $\text{RSA}_{N,e} : \mathbb{Z}_N^* \rightarrow \mathbb{Z}_N^*$ mit $a \mapsto a^e \bmod N$ ein Isomorphismus, die inverse Abbildung ist gegeben durch $\text{RSA}_{N,e}^{-1} = \text{RSA}_{N,d}$ für $d := e^{-1} \bmod \varphi(N)$. Sind die Bedingungen der RSA-Annahme gegeben, handelt es sich unter der RSA-Annahme um eine Oneway-Permutation.

Wer die Faktorisierung des RSA-Moduls N bzw. $\varphi(N)$ kennt, kann $\text{RSA}_{N,e}$ effizient invertieren. Es ist aber offen, ob auch die Umkehrung gilt, d.h. ist die Kenntnis der Faktorisierung bzw. $\varphi(N)$ notwendig, um $\text{RSA}_{N,e}$ zu invertieren?

Das RSA-Encryption-Schema ist nicht semantisch sicher (bereits gegen die schwächste Angriffsform, den sogenannten Chosen-Plaintext-Angriffen). Zu gegebenem $m_0, m_1 \in \mathbb{Z}_N^*$ und $c := \text{RSA}_{N,e}(m_b)$ bestimmen wir $b \in \{0, 1\}$, indem wir die Verschlüsselung $\text{RSA}_{N,e}(m_0)$ berechnet und mit der Challenge c vergleichen:

$$(2) \quad b = 0 \quad \iff \quad c = \text{RSA}_{N,e}(m_0).$$

RSA ist ein *deterministisches* Verschlüsselungsverfahren. Das ElGamal-Schema ist hingegen ein *probabilistisches* Verfahren, beim Chiffrieren verwendet man

Zufallsbits: Verschlüsselt man die gleiche Nachricht mit verschiedenen Zufallsbits, sind auch die Ciphertexte unterschiedlich, so dass eine Äquivalenz der Form (2) nicht gegeben ist.

Basierend auf der Homomorphie-Eigenschaft der $\text{RSA}_{N,e}$ -Funktion kann man zu $c := \text{RSA}_{N,e}(m)$ durch einen Chosen-Ciphertext-Angriff (CCA) das Urbild m bestimmen:

1. Wähle $r \in_{\text{R}} \mathbb{Z}_N^* \setminus \{1\}$.
2. Dekodiere $\text{RSA}_{N,e}(r) \cdot c$ mit Hilfe des Orakels, sei m_r die Antwort. Es gilt $\text{RSA}_{N,e}(r) \cdot c = \text{RSA}_{N,e}(rm)$.
3. Gib $r^{-1}m_r \bmod N$ aus.

Zum Schutz gegen CCA werden Formate für die Nachrichten festgelegt, z.B. PKCS #1. Entspricht der Aufbau nicht der Vorgabe, weil zum Beispiel zwei Ciphertexte multipliziert wurden, wird die Dechiffrierung abgelehnt und der Ciphertext nicht akzeptiert.

1.3. Signatur-Schema. Auf der Symmetrie

$$\text{RSA}_{N,d}^{-1} = (\text{RSA}_{N,e}^{-1})^{-1} = \text{RSA}_{N,e}$$

beruht das RSA-Signatur-Schema. Um $m \in \mathbb{Z}_N^*$ zu unterschreiben, bestimme

$$y := \text{RSA}_{N,e}^{-1}(m) = m^d.$$

Um die Unterschrift $y \in \mathbb{Z}_N^*$ der Nachricht m zu überprüfen, teste

$$\text{RSA}_{N,e}(y) \stackrel{!}{=} m,$$

d.h. ob die RSA-Verschlüsselung von y der Nachricht m entspricht.

Da $\text{RSA}_{N,e}^{-1}$ ebenfalls homomorph ist, kann mittels Chosen-Message-Angriff (CMA) jeder die Unterschrift zu einer (beliebigen) Nachricht $m \in \mathbb{Z}_N^*$ bestimmen.

2. Rabin-Schema

Auf RABIN geht die folgende, RSA-ähnliche Funktion zurück, deren Invertierung nachweislich äquivalent zur Faktorisierung des Moduls ist [Rabin79]. Sei $N = pq$ ein RSA-Modul und

$$\text{QR}_N := \{r^2 \mid r \in \mathbb{Z}_N^*\} \subset \mathbb{Z}_N^*$$

die Untergruppe der quadratischen Reste modulo N . Nach Chinesischem Restsatz gilt

$$\begin{array}{ccc} \mathbb{Z}_N^* & \simeq & \mathbb{Z}_p^* \times \mathbb{Z}_q^* \\ \cup & & \cup \times \cup \\ \text{QR}_N & \simeq & \text{QR}_p \times \text{QR}_q \end{array}$$

wobei $\text{QR}_N, \text{QR}_p, \text{QR}_q$ jeweils (echte) Untergruppen sind.

Lemma 3.7. *Sei $p > 2$ prim. Dann gilt:*

- a) $r^2 \in \text{QR}_p$ hat modulo p genau die beiden Wurzeln $\pm r$, d.h. Quadrieren modulo p ist eine 2:1-Abbildung.
- b) $\text{QR}_p = \left\{ a \in \mathbb{Z}_p^* \mid a^{\frac{p-1}{2}} = +1 \pmod{p} \right\}$ (Euler-Kriterium)
- c) Für $p \equiv 3 \pmod{4}$ sind $\pm a^{\frac{p+1}{4}} \pmod{p}$ die beiden Wurzeln von $a \in \text{QR}_p$.

Beweis. Übungsaufgabe. □

Für $p \equiv 1 \pmod{4}$ existiert ein effizienter, probabilistischer Algorithmus, um Wurzeln modulo p zu berechnen [MOV97, §3.5].

Satz 3.8. *Sei N ein RSA-Modul. Das Quadrieren modulo N ist eine 4:1-Abbildung. Die 4 Wurzeln von $a \in \text{QR}_N$ sind von der Form $\pm r_1, \pm r_2 \in \mathbb{Z}_N^*$ mit $r_1 \neq \pm r_2$.*

Beweis. Seien $\pm r_p, \pm r_q$ die Wurzeln von a modulo p bzw. modulo q . Wegen $\text{QR}_N \simeq \text{QR}_p \times \text{QR}_q$ sind $(\pm r_p, \pm r_q)$ genau die Wurzeln von $a \pmod{N}$. Setze $r_1 := (r_p, r_q) \in \mathbb{Z}_N^*$ und $r_2 := (r_p, -r_q) \in \mathbb{Z}_N^*$. □

Korollar 3.9. *Faktorisieren des RSA-Moduls N ist (unter probabilistischer Polynomialzeit-Reduktion) äquivalent zum Wurzelziehen modulo N .*

Beweis. Mit Hilfe der Faktorisierung des Moduls $N = pq$ reduziere das Wurzelziehen modulo N auf die Bestimmung der Wurzeln modulo der Primzahlen p und q (die wir effizient berechnen können) und setze das Resultat mittels Chinesischem Restsatz zusammen.

Zu zeigen bleibt, dass Faktorisierung des Moduls N auf Wurzelziehen modulo N reduzierbar ist, d.h. man N effizient mit Hilfe eines (beliebigen) SQRT-Algorithmus faktorisieren kann. Wähle $a \in_{\mathbb{R}} \mathbb{Z}_N^*$ zufällig und berechne mit SQRT-Algorithmus eine Wurzel r von a^2 modulo N . Mit Wahrscheinlichkeit $\frac{1}{2}$ (bezogen auf die zufällige Wahl von a) gilt $a \neq \pm r$. Sei $a \neq \pm r \pmod{N}$. $N = pq$ ist dann weder Teiler von $a + r$ noch von $a - r$. Aus

$$(a + r)(a - r) = a^2 - r^2 = 0 \pmod{N}$$

folgt $pq \mid (a + r)(a - r)$. Weil p, q prim sind und $N = pq$ weder $a + r$ noch $a - r$ teilt, teilt p einen der beiden Faktoren und folglich q den anderen. Also

$$\text{ggT}(a \pm r, N) = \{p, q\}.$$

Mit jedem SQRT-Algorithmus können wir mit Wahrscheinlichkeit $\frac{1}{2}$ (bezogen auf die zufällige Wahl von $a \in_{\mathbb{R}} \mathbb{Z}_N^*$) den Modul N faktorisieren. □

Satz 3.10. Sei $N = pq$ eine Blumzahl, d.h. ein RSA-Modul mit $p, q \equiv 3 \pmod{4}$. Dann ist $\text{Rabin}_N : \text{QR}_N \rightarrow \text{QR}_N$ mit $a \mapsto a^2 \pmod{N}$ ein Isomorphismus. Unter der Faktorisierungsannahme ist es eine Oneway-Permutation.

Beweis. Übungsaufgabe. Beachte, dass $-1 \notin \text{QR}_p$ und $-1 \notin \text{QR}_q$. \square

Für das Rabin-Chiffrier- und Rabin-Signatur-Schema schränke den Nachrichtenraum auf QR_N für eine Blum-Zahl N ein und verwende Rabin_N statt $\text{RSA}_{N,e}$. Der Public-Key ist die Blum-Zahl N , der Secret-Key ist die Faktorisierung des Moduls $N = pq$.

3. Fiat-Shamir-Identifikation

Das Identifikationsschema von FIAT und SHAMIR [FS86, FFS88] ist charakterisiert durch

- Sicherheit beruht auf Faktorisierung des RSA-Moduls N
- Sicherheit gegen aktive Angreifer.

Insbesondere ist das Protokoll perfekt zero-knowledge.

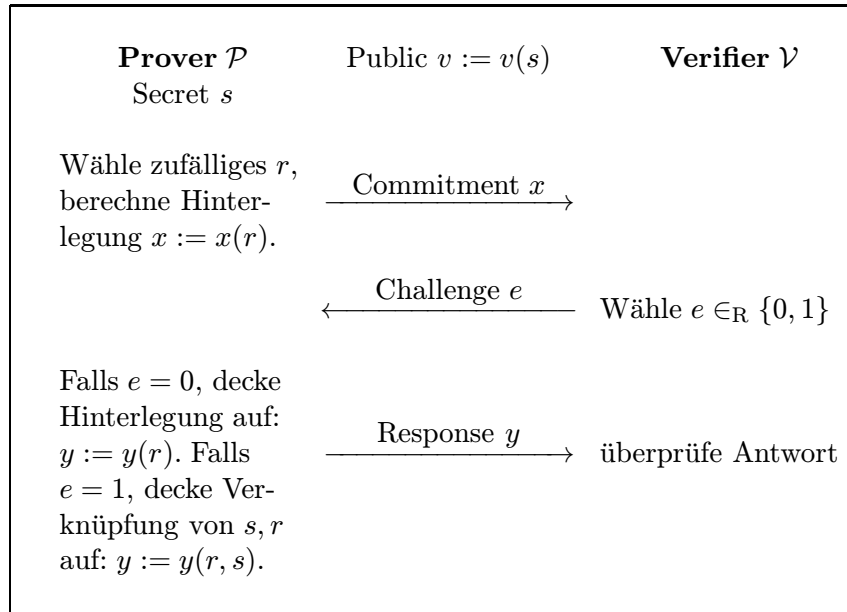
3.1. Vereinfachtes Schema. Wir betrachten zunächst eine einfache Variante des Fiat-Shamir-Identifikation-Schemas. Der öffentliche Schlüssel ist ein RSA-Modul N und $v \in \text{QR}_N$, der geheime Schlüssel ist eine Wurzel s von v modulo N , also $v = s^2$. Besteht der Prover das Protokoll, kennt er eine Wurzel von v modulo N .

Für den Proof-Of-Knowledge für s verwende das übliche Three-Move-Protokoll aus Abbildung 1, das wir bereits von der Schnorr-Identifikation (mit $t = 1$) kennen:

1. Der Prover \mathcal{P} hinterlegt einen Zufallswert r durch ein sogenanntes Commitment $x := x(r)$.
2. Der Verifier \mathcal{V} sendet eine Challenge $e \in_{\mathbb{R}} \{0, 1\}$.
 - Falls $e = 0$, möchte \mathcal{V} überprüfen, ob \mathcal{P} das Commitment x aufdecken kann.
 - Falls $e = 1$, möchte \mathcal{V} überprüfen, ob \mathcal{P} eine Verknüpfung von s und r , die durch v und x gegeben ist, aufdecken kann.²
3. In Abhängigkeit der Challenge e schickt \mathcal{P} eine Antwort y , die der Verifier \mathcal{V} überprüft.

Die *Completeness*-Eigenschaft (ein ehrlicher Prover \mathcal{P} besteht das Protokoll) folgt zumeist durch direktes Nachrechnen. Für den Nachweis der *Soundness*-Eigenschaft (besteht $\tilde{\mathcal{P}}$ das Protokoll, kennt $\tilde{\mathcal{P}}$ auch das Geheimnis s), ist

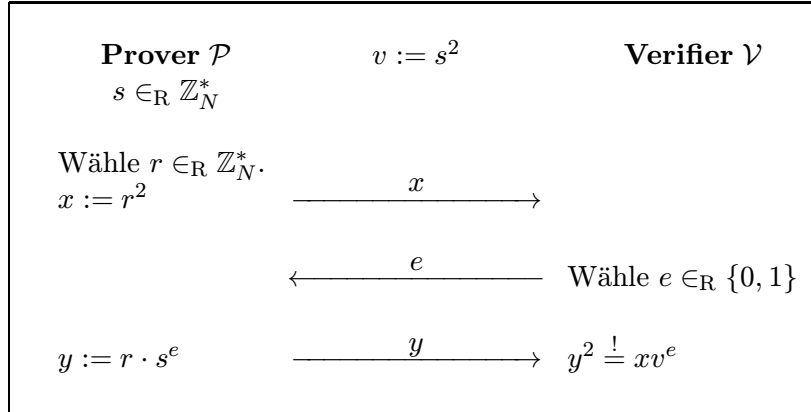
²Der Prover \mathcal{P} deckt nicht das Geheimnis s sondern eine Verknüpfung von s und r auf, um sein Geheimnis s nicht preiszugeben.

ABBILDUNG 1. Three-Move-Protokoll $(\mathcal{P}, \mathcal{V})$ 

die Response y in der Regel so konzipiert, dass man für das gleiche r aus den Antworten $y(r)$ und $y(r, s)$ zu beiden Challenges $e = 0, 1$ das Geheimnis s extrahieren kann. Angenommen, es gibt einen Prover $\tilde{\mathcal{P}}$, der mit hinreichend großer Wahrscheinlichkeit das Protokoll besteht. Der *Knowledge-Extraktor* geht wie folgt vor:

1. Simuliere $\tilde{\mathcal{P}}$ und \mathcal{V} bei einer Ausführung des Protokolls. Speichere dabei den Zustand von $\tilde{\mathcal{P}}$, nachdem dieser seine Hinterlegung $x := x(r)$ geschickt hat.
2. Resete $\tilde{\mathcal{P}}$, setze ihn auf den gespeicherten Zustand zurück. Fahre von dieser Stelle mit der Ausführung fort.

Mit einem Abzähl-Argument (Erfolgsmatrix) schätzt man die Wahrscheinlichkeit ab, dass der $\tilde{\mathcal{P}}$ für das gemachte Commitment x die Challenges beider Ausführungen besteht. Mit Wahrscheinlichkeit $\frac{1}{2}$ sind die Challenges der beiden Ausführungen verschieden, so dass der Extraktor aus beiden Antworten (sofern diese korrekt sind) das Geheimnis mit hinreichend großer Wahrscheinlichkeit ermittelt.

ABBILDUNG 2. (einfachte) Fiat-Shamir-Identifikation $(\mathcal{P}, \mathcal{V})_{\text{FS}_1}$ 

Das Protokoll der vereinfachten Fiat-Shamir-Identifikation $(\mathcal{P}, \mathcal{V})_{\text{FS}_1}$, d.h. der Proof-Of-Knowledge für eine Wurzel s von v modulo N , hat ebenfalls diese Three-Move-Struktur (siehe Abbildung 2). Das Schema basiert auf der homomorphen Funktion $\text{Rabin}_N : (\mathbb{Z}_N^*, \cdot) \rightarrow (\mathbb{Z}_N^*, \cdot)$ mit $a \mapsto a^2 \bmod N$.³

Für die Completeness-Eigenschaft des Fiat-Shamir-Protokolls $(\mathcal{P}, \mathcal{V})_{\text{FS}_1}$ beachte, dass die Response $y = r \cdot s^e$ eine Quadratwurzel von xv^e ist:

$$y^2 = (rs^e)^2 = r^2(s^2)^e = xv^e$$

Ein betrügerischer Prover $\tilde{\mathcal{P}}$ kann das vereinfachte Protokoll mit Wahrscheinlichkeit $\frac{1}{2}$ bestehen:

1. Wähle neben $r \in_{\mathbb{R}} \mathbb{Z}_N^*$ ein $\tilde{e} \in_{\mathbb{R}} \{0, 1\}$ und sende $\tilde{x} := r^2v^{-\tilde{e}}$.
2. Sende als Response $\tilde{y} := r$.

Falls $\tilde{e} = e$, d.h. $\tilde{\mathcal{P}}$ die Challenge geraten hat, besteht $\tilde{\mathcal{P}}$ das Protokoll, denn

$$\tilde{y}^2 = r^2 = r^2v^{-\tilde{e}}v^e = \tilde{x}v^e.$$

Dieser $\tilde{\mathcal{P}}$ besteht das Protokoll mit Wahrscheinlichkeit $\text{Ws}[\tilde{e} = e] = \frac{1}{2}$.

Das Fiat-Shamir-Protokoll $(\mathcal{P}, \mathcal{V})_{\text{FS}_1}$ ist sound, denn kann ein Prover $\tilde{\mathcal{P}}$ zu einer Hinterlegung $x := r^2$ beide Challenges richtig beantworten, kennt er eine Quadratwurzel $y(r, s) \cdot y(r)^{-1}$ von v :

$$\left(\frac{y(r, s)}{y(r)} \right)^2 = \left(\frac{r \cdot s}{r} \right)^2 = s^2 = v.$$

³Zum Vergleich: Die Schnorr-Identifikation beruht auf der homomorphen Funktion $\exp_g : (\mathbb{Z}_q, +) \rightarrow (\langle g \rangle, \cdot)$ mit $a \mapsto g^a$.

Die Fiat-Shamir-Identifikation $(\mathcal{P}, \mathcal{V})_{\text{FS}_1}$ ist sicher gegen passive Angreifer (also Angriffe aus dem Stand), sofern die Berechnung einer nicht-trivialen Wurzel schwierig ist.

Wird das Schlüsselpaar von einer Trusted-Party vorgegeben, kann man Hilfe von $\tilde{\mathcal{P}}$ den Modul N wie folgt faktorisieren:

1. Wähle $s_1 \in_{\mathbb{R}} \mathbb{Z}_N^*$, setze $v := s_1^2$.
2. Bestimme mit Hilfe des Knowledge-Extraktors eine Wurzel $s_2 := y(r, s) \cdot y(r)^{-1}$ von v modulo N .
3. Berechne $\text{ggT}(s_1 \pm s_2, N)$. Falls dies nicht die Faktorisierung liefert, gehe zu Schritt 1.

Das Protokoll $(\mathcal{P}, \mathcal{V})_{\text{FS}_1}$ ist ebenfalls sicher gegen aktive Angreifer \mathcal{A} , d.h. der Angreifer führt zunächst ℓ -mal das Protokoll $(\mathcal{P}, \mathcal{V}_{\mathcal{A}})_{\text{FS}_1}$ aus und versucht anschließend sich als Prover $\tilde{\mathcal{P}}_{\mathcal{A}}$ gegenüber \mathcal{V} auszuweisen. Nach Schritt 1 im obigen Algorithmus simuliere mit Hilfe von s_1 zunächst ℓ Ausführungen des Protokolls $(\mathcal{P}, \mathcal{V}_{\tilde{\mathcal{P}}})_{\text{FS}_1}$.

Die Sicherheit des vereinfachten Fiat-Shamir-Protokolls $(\mathcal{P}, \mathcal{V})_{\text{FS}_1}$ gegen aktive Angreifer folgt alternativ aus der Zero-Knowledge-Eigenschaft. Der Simulator \mathcal{S} geht wie folgt vor:

1. Wähle neben $r \in_{\mathbb{R}} \mathbb{Z}_N^*$ ein $\tilde{e} \in_{\mathbb{R}} \{0, 1\}$. Setze $\tilde{x} := r^2 v^{-\tilde{e}}$.
2. Wähle $e \in_{\mathbb{R}} \{0, 1\}$.
3. Setze $\tilde{y} := r$.
4. Falls $e = \tilde{e}$, gib $(\tilde{x}, e, \tilde{y})$ aus. Sonst gehe zu Schritt 1.

Im Erwartungswert liefert der Simulator nach 2 Iterationen eine Ausgabe $(\tilde{x}, e, \tilde{y})$. Die Verteilung dieser Daten $(\tilde{x}, e, \tilde{y})$ und der Daten (x, e, y) einer Protokoll-Ausführung sind identisch:

- \tilde{x} und x sind zufällige quadratische Reste QR_N .
- \tilde{y} und y sind zufällige Wurzeln von xv^e bzw. $\tilde{x}v^e$.

Das vereinfachte Fiat-Shamir-Protokoll ist perfekt zero-knowledge.

Die Erfolgswahrscheinlichkeit eines trivialen Betrügers (der die Challenge rät) ist 2^{-1} . Um diese auf 2^{-k} zu senken, wiederhole sequentiell das Protokoll k -mal. Dabei bleibt die Zero-Knowledge-Eigenschaft erhalten (wieso?).

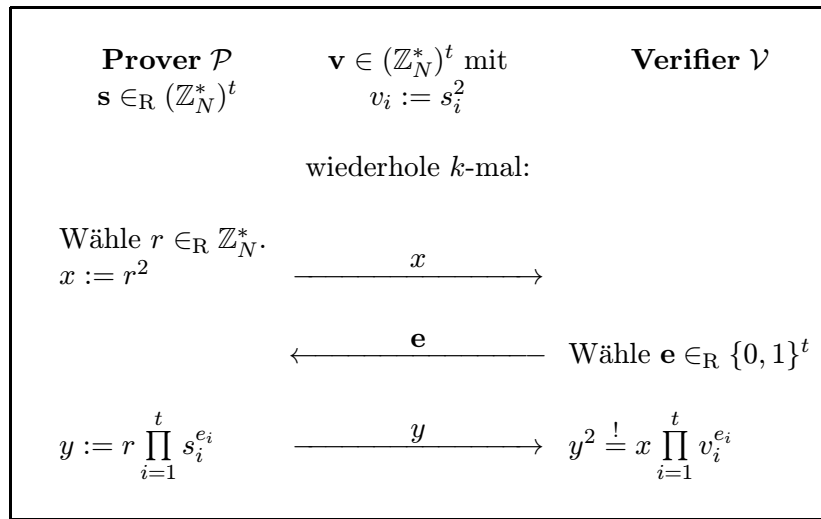
3.2. Schema. Um die Erfolgswahrscheinlichkeit durch Erraten der Challenge auf 2^{-tk} zu senken, ist im Original-Schema $(\mathcal{P}^k, \mathcal{V}^k)_{\text{FS}}$ der Public-Key

\mathbf{v} ein Vektor aus QR_N^t und der Secret-Key \mathbf{s} der Vektor der t Wurzeln:

$$\begin{aligned}\mathbf{s} &:= (s_1, \dots, s_t) \in_{\mathbb{R}} (\mathbb{Z}_N^*)^t \\ \mathbf{v} &:= (v_1, \dots, v_t) \in (\text{QR}_N)^t \quad \text{mit } v_i^2 := s_i^2\end{aligned}$$

Abbildung 3 zeigt die Fiat-Shamir-Identifikation $(\mathcal{P}^k, \mathcal{V}^k)_{\text{FS}}$.

ABBILDUNG 3. Fiat-Shamir-Identifikation $(\mathcal{P}^k, \mathcal{V}^k)_{\text{FS}}$



Durch Raten der Challenges in den k Runden kann ein betrügerischer Prover $\tilde{\mathcal{P}}$ das Protokoll bestehen:

Lemma 3.11. *Es gibt $\tilde{\mathcal{P}}$ für $(\mathcal{P}^k, \mathcal{V}^k)_{\text{FS}}$ mit $\text{ERF}_{\mathbf{v}}(\tilde{\mathcal{P}}, \mathbf{v}) = 2^{-kt}$.*

Sei $\tilde{\mathcal{P}}$ ein Angreifer aus dem Stand auf $(\mathcal{P}^k, \mathcal{V}^k)_{\text{FS}}$.

Satz 3.12. *Es gibt einen probabilistischen Algorithmus $\text{AL} : (\tilde{\mathcal{P}}, \mathbf{v}, N) \rightarrow (\mathbf{c}, \mathbf{s}_{\mathbf{c}})$ mit $E_w |\text{AL}| = \mathcal{O}(|\tilde{\mathcal{P}}|/\epsilon)$, sofern $\tilde{\mathcal{P}}$ mit \mathbf{v} Erfolgswahrscheinlichkeit $\epsilon \geq 2^{-tk+1}$ hat, so dass $\mathbf{c} \in \{\pm 1, 0\}^t \setminus \{0\}$ und $\mathbf{s}_{\mathbf{c}}^2 = \prod_{i=1}^t v_i^{c_i}$.*

Dies entspricht Satz 2 aus Kapitel 2 der DL-Identifikation. $(\mathcal{P}^k, \mathcal{V}^k)_{\text{FS}}$ ist ein Proof-Of-Knowledge einer nicht-trivialen Quadratwurzel modulo N (Soundness).

Damit ist $(\mathcal{P}^k, \mathcal{V}^k)_{\text{FS}}$ sicher gegen passive Angreifer, sofern die Berechnung nicht-trivialer Quadratwurzeln modulo N schwierig ist.

Korollar 3.13. *Es gibt einen probabilistischen Algorithmus $\overline{\text{AL}} : (\tilde{\mathcal{P}}, N) \mapsto \{p, q\}$ mit $E_w |\overline{\text{AL}}| = \mathcal{O}(|\tilde{\mathcal{P}}|/\epsilon)$, sofern $\tilde{\mathcal{P}}$ für beliebiges $\mathbf{v} \in \text{QR}_N^t$ Erfolgswahrscheinlichkeit $\epsilon \geq 2^{-tk+1}$ hat.*

Beweis. $\overline{\text{AL}}$ wählt $\mathbf{s} \in_{\mathbb{R}} (\mathbb{Z}_N^*)^t$ zufällig und bildet $\mathbf{v} = (s_1^2, \dots, s_t^2)$. Berechne $(\mathbf{c}, \mathbf{s}_{\mathbf{c}}) := \text{AL}(\tilde{\mathcal{P}}, \mathbf{v}, N)$. Dann gilt $\mathbf{s}_{\mathbf{c}}^2 = (\prod_{i=1}^t s_i^{c_i})^2$ und

$$\text{Ws}_{\mathbf{s}} \left[\underbrace{\mathbf{s}_{\mathbf{c}} \neq \pm \prod_{i=1}^t s_i^{c_i}}_{\text{ggT}(\mathbf{s}_{\mathbf{c}} \pm \prod_{i=1}^t s_i^{c_i}, N) = \{p, q\}} \right] = \frac{1}{2}$$

□

Die Aussagen von Satz 3.12 und Korollar 3.13 gelten auch für den Fall, dass die Erfolgswahrscheinlichkeit $\epsilon \geq (1 + \delta)2^{-kt}$ für ein $\delta > 0$ ist. Die erwartete Laufzeit ist dann $\mathcal{O}(|\tilde{\mathcal{P}}|/(\delta\epsilon))$.

Der Simulator für das Fiat-Shamir-Protokoll $(\mathcal{P}^k, \mathcal{V}^k)_{\text{FS}}$ hat den Verzögerungsfaktor 2^t :

Satz 3.14. *Das Fiat-Shamir-Protokoll $(\mathcal{P}^k, \mathcal{V}^k)_{\text{FS}}$ ist für $t = \mathcal{O}(\log n)$ perfekt zero-knowledge, wobei n die Bitlänge des Moduls sei.*

Ein aktiver Angreifer \mathcal{A} führt zunächst ℓ -mal $(\mathcal{P}^k, \tilde{\mathcal{V}}_{\mathcal{A}}^k)_{\text{FS}}$ aus und dann $(\tilde{\mathcal{P}}_{\mathcal{A}}^k, \mathcal{V})_{\text{FS}}$. Hat \mathcal{A} die Erfolgswahrscheinlichkeit mindestens 2^{-2tk+1} , dann liefert die Koalition $[\mathcal{P}^k, \mathcal{A}]$ die Faktorisierung des Moduls N :

Satz 3.15. *Es gibt einen probabilistischen Algorithmus $\text{AL} : (\mathcal{A}, N) \mapsto \{p, q\}$ mit $\mathbb{E}_w |\text{AL}| = \mathcal{O}(|\tilde{\mathcal{P}}|/\epsilon)$, sofern \mathcal{A} für beliebiges $\mathbf{v} \in \text{QR}_N^t$ Erfolgswahrscheinlichkeit $\epsilon \geq 2^{-tk+1}$ hat.*

Damit ist $(\mathcal{P}^k, \mathcal{V}^k)_{\text{FS}}$ sicher gegen aktive Angriffe, sofern Faktorisierung von N schwierig ist. Satz 3.15 ist analog zu Satz 4 bezüglich $(\mathcal{P}^k, \mathcal{V}^k)_{\text{OK}}$.

Nachteil der Fiat-Shamir-Identifikation ist die lange Kommunikation von $k(2n + t)$ Bits. \mathcal{P} und \mathcal{V} benötigen jeweils maximal $(t + 1)k$ (und im Mittel $\frac{t+2}{2}k$) Multiplikationen modulo N . Betrachten wir den Rechenaufwand zum Sicherheitsniveau 2^{80} , d.h. $kt = 80$. Für $k = 10$ und $t = 8$ benötigen \mathcal{P} und \mathcal{V} maximal 90 und im Mittel 50 Multiplikationen modulo N .

Das Fiat-Shamir-Schema ist ein Beispiel für ein sogenanntes *Identity-Based-Scheme*. Die Trusted-Party wählt einen RSA-Modul N , der öffentliche Schlüssel eines Teilnehmers A ist der Hashwert $\mathbf{v}_A := H(\text{Identität von } A) \in \text{QR}_N^t$ für eine kollisionsresistente Hash-Funktion H . Mit Hilfe der Primfaktorzerlegung des Moduls bestimmt die Trusted-Party den zugehörigen, geheimen Schlüssel \mathbf{s} mit $v_{A,i} = s_{A,i}^2$. Jeder Teilnehmer kann selbständig den öffentlichen Schlüssel von A berechnen, ohne sich diesen aus eventuell unsicheren Quellen (z.B. übers Internet) besorgen zu müssen.

3.3. Signatur-Schema. Analog zur Schnorr-Identifikation transformiert man die Fiat-Shamir-Identifikation in ein Signatur-Schema, indem die Challenge \mathbf{e} durch einen Hashwert $H(x, m)$ ersetzt wird. Die Fiat-Shamir-Unterschrift zu $m \in \mathbb{Z}_N^*$ besteht aus $(\mathbf{y}, \mathbf{e}) \in (\mathbb{Z}_N^*)^k \times \{0, 1\}^{kt}$ mit

- $\mathbf{e} := H(r_1^2, \dots, r_k^2, m)$ für $\mathbf{r} \in_{\mathbb{R}} (\mathbb{Z}_N^*)^k$.
- $y_j := r_j \prod_{i=1}^t s_i^{e_{j,i}}$.

Zum Überprüfen einer Unterschrift (\mathbf{y}, \mathbf{e}) zu m teste:

$$\mathbf{e} \stackrel{!}{=} H\left(y_1^2 \prod_{i=1}^t v_i^{-e_{1,i}}, \dots, y_k^2 \prod_{i=1}^t v_i^{-e_{k,i}}, m\right).$$

Analog zum Schnorr-Signatur-Schema zeigt man, dass unter der Faktorisierungsannahme das Fiat-Shamir-Signatur-Schema im Random-Oracle-Modell [BR93] sicher gegen Chosen-Message-Angriffe (CMA) ist.

Die FS-Unterschrift ist $k(n+t)$ Bits lang. Vergleichen wir die Anzahl der Multiplikationen für FS-Signaturen mit denen für RSA- und Schnorr-Unterschriften. Der RSA-Modul habe die Bitlänge $n = 1024$ und die Gruppenordnungen q habe die Bitlänge 160:

Signatur-Erzeugung:

- FS: maximal $tk + k = 90$ und im Mittel $\frac{t+2}{k} = 50$ (speziell für $k = 8$ und $t = 10$).
- RSA: maximal $2 \cdot 1024$ und im Mittel $1,5 \cdot 1024$.
- Schnorr: maximal 320 und im Mittel 160.

Signatur-Erzeugung (Online):

- FS: maximal 80 und im Mittel 40.
- RSA: maximal $2 \cdot 1024$ und im Mittel $1,5 \cdot 1024$.
- Schnorr: keine.

Signatur-Prüfung:

- FS: maximal 90 und im Mittel 50.
- RSA: maximal $\log e$ (z.B. 5 für $e = F_4$).
- Schnorr: maximal 400 und im Mittel 280.

Bei Fiat-Shamir- und Schnorr-Signaturen kann $x = r^2$ bzw. $x = g^r$ unabhängig von der zu unterschreibenden Nachricht als Preprocessing bestimmt werden und online ist dann nur noch y zu berechnen.

4. Diskreter Log modulo N und Paillier-Schema

Sei N ein RSA-Modul und $g \in \mathbb{Z}_N^*$. Wie schwierig ist es, den diskreten Logarithmus zur Basis g in der Untergruppe $\langle g \rangle \subseteq \mathbb{Z}_N^*$ zu bestimmen?

Satz 3.16 (Bach 1984, Chor 1985). Sei $N = pq$ ein RSA-Modul, dessen Primfaktoren p, q die gleiche Bitlänge haben. Dann ist das Diskrete-Logarithmus-Problem $\log_g(\cdot)$ in $\langle g \rangle \subseteq \mathbb{Z}_N^*$ für zufälliges $g \in_{\mathbb{R}} \mathbb{Z}_N^*$ mindestens so schwierig wie Faktorisierung des Moduls.

Beweis. Wir skizzieren, wie mit Hilfe eines DLOG-Algorithmus die Faktorisierung zu berechnen ist [HSS93]. Wegen

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

und $g^{\varphi(N)} = 1$ gilt:

$$g^N = g^{N-\varphi(N)} = g^{p+q+1}.$$

Also $\log_g g^N = p + q + 1 \pmod{\text{ord}_N(g)}$. Für zufälliges $g \in_{\mathbb{R}} \mathbb{Z}_N^*$ gilt mit hinreichend großer Wahrscheinlichkeit $\text{ord}_N(g) \geq p + q + 1$. Berechne $s := \log_g g^N$ für zufälliges $g \in_{\mathbb{R}} \mathbb{Z}_N^*$ mit DLOG-Algorithmus. Falls $s = p + q + 1$, sind $\{p, q\}$ die beiden Nullstellen von

$$X^2 + (s+1)X + N = X^2 + pX + qX + N \in \mathbb{R}[X],$$

die man mit der pq -Formel bestimmt. □

Es ist allerdings unklar, ob die Kenntnis der Primfaktoren bei der effizienten Berechnung des diskreten Logarithmus modulo N hilfreich ist.

PAILLIER [Pa99] hat ein Public-Key-Crypto-Schema entworfen, bei dem zum Dechiffrieren der diskrete Logarithmus in der Untergruppe der N -ten Einheitswurzeln modulo N^2 zu bestimmen ist. Sei $N = pq$ ein RSA-Modul mit $p \nmid q-1$ und $q \nmid p-1$, d.h. $\text{ggT}(N, \varphi(N)) = 1$. Es gilt

$$\begin{aligned}\varphi(N) &= \varphi(p) \cdot \varphi(q) = (p-1)(q-1) \\ \lambda(N) &= \text{kgV}(\varphi(p), \varphi(q)) = \text{kgV}(p-1, q-1)\end{aligned}$$

und für den quadrierten Modul:

$$\begin{aligned}\varphi(N^2) &= \varphi(p^2) \cdot \varphi(q^2) = p(p-1)q(q-1) = N \cdot \varphi(N) \\ \lambda(N^2) &= \text{kgV}(\varphi(p^2), \varphi(q^2)) = pq \cdot \text{kgV}(p-1, q-1) = N \cdot \lambda(N)\end{aligned}$$

Zur Vereinfachung setze $\lambda := \lambda(N)$. Identifiziere \mathbb{Z}_N mit $[0, N)$ und \mathbb{Z}_N^2 mit $[0, N^2)$, um \mathbb{Z}_N^* in $\mathbb{Z}_{N^2}^*$ einzubetten. Beachte:

$$\begin{aligned}\text{ggT}(a, N) = 1 &\implies \text{ggT}(a, N^2) = 1 \\ a = b \pmod{N^2} &\implies a = b \pmod{N}.\end{aligned}$$

Wir nennen $a \in \mathbb{Z}_{N^2}^*$ einen N -ten Potenzrest modulo N^2 , falls ein $r \in \mathbb{Z}_{N^2}^*$ mit $r^N = a$ existiert. Die N -ten Potenzreste modulo N^2 bilden eine

Untergruppe von $\mathbb{Z}_{N^2}^*$ der Ordnung $\varphi(N)$, denn jeder N -ter Potenzrest hat genau N viele N -te Wurzeln. Sei

$$\mathcal{E}_N := \{r \in \mathbb{Z}_{N^2}^* \mid r^N = 1 \pmod{N^2}\}$$

die Menge der N -ten Einheitswurzeln modulo N^2 . Da allgemein für $i \in \mathbb{N}$ gilt

$$(1+N)^i = \sum_{j=0}^i \binom{i}{j} 1^{i-j} N^j = 1 + iN + \sum_{j=2}^i \binom{i}{j} N^j = 1 + iN \pmod{N^2}$$

besteht \mathcal{E}_N genau aus den N Werten $1 + kN$ für $k = 0, 1, \dots, N-1$, denn $(1+kN)^N = 1 + kN^2 = 1$.

Sei $g \in \mathbb{Z}_{N^2}^*$ mit $\text{ord}_{N^2}(g) = \lambda N$. Aufgrund $g^{\lambda N} = 1$ ist $g^\lambda \in \mathcal{E}_N$. Sei $g^\lambda = 1 + kN$. Unter welchen Bedingungen können wir den diskreten Logarithmus m von g^m zur Basis g berechnen? Erhebe g^m in die λ -te Potenz:

$$(g^m)^\lambda = (g^\lambda)^m = (1+kN)^m = 1 + mkN \pmod{N^2}$$

Setze

$$L := \frac{(1+mkN) - 1}{kN} = \frac{((g^m)^\lambda - 1) \bmod N^2}{(g^\lambda - 1) \bmod N^2}.$$

Wegen $\lambda N = \text{ord}_{N^2}(g)$ gilt:

$$\lambda \cdot m = L \pmod{\lambda N}$$

Diese Kongruenz hat für $m \in \mathbb{Z}_{\lambda N}$ zwar $N = \text{ggT}(\lambda, \lambda N)$ viele Lösungen, aber modulo N

$$\lambda \cdot m = L \pmod{N}$$

exakt eine, nämlich $m := \lambda^{-1}L \bmod N$.⁴ Bei Kenntnis von λ oder äquivalent der Faktorisierung von N können wir den diskreten Logarithmus $m = \log_g g^m$ berechnen, sofern $m \in \mathbb{Z}_N$.

Satz 3.17. Sei $g \in \mathbb{Z}_{N^2}^*$ mit $\text{ord}_{N^2}(g) = \lambda N$ für einen RSA-Modul N mit $\text{ggT}(N, \varphi(N)) = 1$. Dann ist die Abbildung $\text{Pail}_{N,g} : (\mathbb{Z}_N, +) \times (\mathbb{Z}_N^*, \cdot) \rightarrow (\mathbb{Z}_{N^2}^*, \cdot)$ mit $(m, r) \mapsto g^{m r^N} \bmod N^2$ ein Isomorphismus.

Beweis. Die Funktion ist homomorph

$$\begin{aligned} \text{Pail}_{N,g}(m_1, r_1) \cdot \text{Pail}_{N,g}(m_2, r_2) &= g^{\overbrace{m_1 + m_2}^{\bmod \lambda N} \left(\overbrace{r_1 \cdot r_2}^{\bmod N^2} \right)^N} \\ &= \text{Pail}_{N,g}(\underbrace{m_1 + m_2}_{\bmod N}, \underbrace{r_1 r_2}_{\bmod N}), \end{aligned}$$

⁴Das Inverse $\lambda^{-1} \bmod N$ existiert, da nach Wahl die Primfaktoren p, q keine Teiler von λ sind.

denn die abweichenden Moduli ändern nicht das Ergebnis modulo N :

$$\begin{aligned}(m_1 + m_2 \bmod \lambda N) &= m_1 + m_2 \pmod{N} \\ (r_1 r_2 \bmod N^2) &= r_1 r_2 \pmod{N}\end{aligned}$$

Es genügt der Nachweis der Injektivität, denn beide Mengen haben die gleiche Mächtigkeit:

$$|\mathbb{Z}_N| \cdot |\mathbb{Z}_N^*| = pq \cdot \underbrace{\varphi(p)\varphi(q)}_{=\varphi(N)} = p(p-1)q(q-1) = \underbrace{\varphi(p^2) \cdot \varphi(q^2)}_{=\varphi(N^2)} = |\mathbb{Z}_{N^2}^*|.$$

Sei $\text{Pail}_{N,g}(m_1, r_1) = \text{Pail}_{N,g}(m_2, r_2)$, d.h.

$$(3) \quad g^{m_1 - m_2} (r_1 r_2^{-1})^N = 1 \pmod{N}$$

Wir zeigen, dass dann $(m_1, r_1) = (m_2, r_2)$ gilt. Erhebe (3) in die λ -te Potenz. Da $r^{\lambda N} = 1 \pmod{N^2}$ für alle $r \in \mathbb{Z}_{N^2}^*$, gilt:

$$g^0 = 1 = (g^\lambda)^{m_1 - m_2} (r_1 r_2^{-1})^{\lambda N} = (g^\lambda)^{m_1 - m_2} \pmod{N^2}$$

Aus $\text{ord}_{N^2}(g) = \lambda N$ erhalten wir

$$\lambda(m_1 - m_2) = 0 \pmod{\lambda N},$$

also $m_1 - m_2 = 0 \pmod{N}$. Wegen $m_1, m_2 \in \mathbb{Z}_N$ gilt $m_1 = m_2$. Aus $r_1^N = r_2^N \pmod{N^2}$ folgt

$$r_1^N = r_2^N \pmod{N}.$$

Nach Voraussetzung ist N ein RSA-Exponent zu N , d.h. es gibt genau eine N -te Wurzel modulo N . Es folgt $r_1 = r_2 \pmod{N}$, d.h. $r_1 = r_2$. \square

Wegen $\text{Pail}_{N,g}(m, r)^\lambda = (g^\lambda)^m$ gilt:

Korollar 3.18. Sei $c := \text{Pail}_{N,g}(m, r)$. Dann gilt $m = \lambda^{-1}L(c) \pmod{N}$ mit

$$L(c) = \frac{(c^\lambda - 1) \bmod N^2}{(g^\lambda - 1) \bmod N^2}.$$

Definition 3.19 (Paillier-Schlüssel). Das Schlüssel-Paar zu Pailliers Crypto-Schema bilden:

- *Public-Key* (N, g) : zufälliger RSA-Modul N mit $\text{ggT}(N, \varphi(N)) = 1$ und $g \in \mathbb{Z}_{N^2}^*$ mit $\text{ord}_{N^2}(g) = \lambda N$ für $\lambda := \lambda(N)$.
- *Secret-Key* λ bzw. Faktorisierung von N .

Der Nachrichtenraum ist \mathbb{Z}_N .

Um eine Nachricht $m \in \mathbb{Z}_N$ zu chiffrieren, wähle $r \in_R \mathbb{Z}_N^*$ und setze $c := \text{Pail}_{N,g}(m, r)$. Dekodiere mit Hilfe des Secret-Keys λ gemäß Korollar 3.18. Die Sicherheit basiert auf der Paillier-Annahme, dass $\text{Pail}_{N,g}$ eine Oneway-Permutation ist. Es ist offen, ob die RSA- oder die Faktorisierungs-Annahme die Paillier-Annahme implizieren.

Sei $a_0 \in \mathbb{Z}_{N^2}^*$ ein N -ter Potenzrest, $a_1 \in \mathbb{Z}_{N^2}^*$ kein N -ter Potenzrest und $b \in_{\mathbb{R}} \{0, 1\}$. Die Decisional-Composite-Residuosity-Annahme (DCR-Annahme) besagt, gegeben a_b kann man in Polynomialzeit b nicht besser als mit Wahrscheinlichkeit $\frac{1}{2} + \epsilon$ für ein vernachlässigbar kleines $\epsilon \geq 0$ bestimmen [Pa99].

Satz 3.20. *Das probabilistische Verschlüsselungsschema von Paillier ist unter der DCR-Annahme semantisch sicher gegen Chosen-Plaintext-Angriffe.*

Beweis. Wir skizzieren den Beweis: Gegeben zwei verschiedene Nachrichten $m_0, m_1 \in \mathbb{Z}_N$ und $c := \text{Pail}_{N,g}(m_b, r)$. Genau dann ist $b = 0$, wenn

$$g^{-m_b} c = g^{m_b - m_0} r^N$$

ein N -ter Potenzrest modulo N^2 ist. □

Aufgrund der Homomorphie-Eigenschaften ist Pailliers Schema allerdings nicht sicher gegen Chosen-Ciphertext-Angriffe.

Literaturverzeichnis

- [BG92] M. BELLARE and O. GOLDREICH: *On Defining Proofs of Knowledge*, Advances in Cryptology — Proceedings Crypto '92, Lecture Notes in Computer Science, vol. 740, Seiten 390–420, Springer Verlag, 1993.
- [BR93] M. BELLARE und P. ROGAWAY: *Random Oracles are Practical: a Paradigm for Designing Efficient Protocols*, First ACM Conference on Computer and Communication Security, ACM Press, Seiten 62–73, 1993.
- [FFS88] U. FEIGE, A. FIAT und A. SHAMIR: *Zero Knowledge Proofs of Identity*, Journal of Cryptology, Band 1, Seiten 210–217, 1988.
- [FS86] A. FIAT und A. SHAMIR: *How to prove yourself: Practical Solution of Identity and Signature Problems*, Advances in Cryptography — Crypto '86, Lecture Notes in Computer Science, Band 263 (1987), Springer-Verlag, Berlin/Heidelberg, Seiten 186–194, 1986.
- [GMR85] S. GOLDWASSER, S. MICALI and C. RACKOFF: *The Knowledge Complexity of Interactive Proof Systems*, Proceedings 38. te IEEE Symposium on Foundations of Computer Science (FOCS), Seiten 291–304, IEEE Computer Society Press, 1985.
- [G98] O. GOLDREICH: **Foundations of Cryptography**, Fragments of a Book, Version 2.03, 1998.
- [GM84] S. GOLDWASSER and S. MICALI: *Probabilistic Encryption*, Journal of Computer and System Science, vol. 28, Seiten 270–299, 1984.
- [HSS93] J. HÅSTAD, A.W. SCHRIFT und A. SHAMIR: *The Discrete Logarithm Modulo a Composite Hides $\mathcal{O}(n)$ Bits*, Journal of Computer and System Science, Band 47, Seiten 376–404, 1993.
- [MOV97] A.J. MENEZES, P.C. VAN OORSCHOT und S.A. VANSTONE: *Handbook of Applied Cryptography*, CRC Press, Boca Raton, 1997.
- [Pa99] P. PAILLIER: *Public-Key Cryptosystems Based on Composite Degree Residuosity Classes*, Proceedings Eurocrypt '99, Springer Lecture Notes in Computer Science, Band 1592, Seiten 223–228, 1999.
- [Rabin79] M.O. RABIN: *Digital Signatures and Public-Key Functions as intractable as Factorization*, MIT Laboratory for Computer Science Technical Report LCS-TR 212, Massachusetts Institute of Technology, 1979.

- [RSA78] R. RIVEST, A. SHAMIR and L. ADLEMAN: *A Method for Obtaining Digital Signatures and Public-Key Cryptosystems*, Communication of the ACM, vol. 21(2), Seiten 120–126, 1978.
- [S91] C.P. SCHNORR: *Efficient Signature Generation by Smart Cards*, Journal of Cryptology, Band 4(3), Seiten 161–174, 1991.
- [Sil97] R.D. SILVERMAN: *Fast Generation of Random, Strong RSA Primes*, CryptoBytes, RSA Laboratories, Band 3, Nr. 1 (Frühjahr), Seite 9–13, 1997.
- [W90] M.J. WIENER: *Cryptanalysis of Short RSA Secret Exponents*, IEEE Transaction Information Theory, Band IT-36, Nr. 3 (Mai), Seiten 553–558, 1990.