

# Zahlen mit Zahlen - Teil 1

## Mathematische Algorithmen zum sicheren Bezahlen in elektronischen Datennetzen

### Einleitung

Die fortschreitende Nutzung des Internets für kommerzielle Transaktionen insbesondere die Bezahlung von Waren und Dienstleistungen erfordert dringend die Entwicklung eines Protokollsystems zum digitalen Bezahlen. Derzeit wird weltweit an einer Vielzahl von möglichen Lösungen für dieses Problem gearbeitet. Man muß jedoch feststellen, daß sich bis zum jetzigen Zeitpunkt keine der vorgeschlagenen Lösungen etablieren konnte. Unserer Wettbewerbsbeitrag für den 33. Bundeswettbewerb „Jugend forscht“ 1998 im Fachbereich „Mathematik/Informatik“ soll einen weiteren Beitrag in der Diskussion um einen Standard darstellen. In unserer Arbeit werden grundlegende Kenntnisse der Theorie der asymmetrischen Verschlüsselungssysteme, digitaler Signaturen und des RSA-Algorithmus vorausgesetzt. Eine einfache Darstellung hierzu findet sich beispielsweise in [RS97] oder auch in [BSW95].

Unser Lösungsansatz zeichnet sich gegenüber anderen Vorschlägen durch eine Zweiteilung aus, zum einen können Zahlungen nach dem Prinzip von Kreditkarten beziehungsweise Schecks geleistet werden, zum anderen aber auch mit sogenannten elektronischen Münzen“. Die Sicherheit des Kunden beim kreditkartenbasierten Teil soll durch eine 1024-Bit-RSA-Verschlüsselung mit einigen zusätzlichen Sicherheitsmechanismen gewährleistet werden. Der banknotenbasierte Teil gewährleistet eine konsequente Anonymität des Käufers, bei gleichzeitig hoher Sicherheit für Kunden, Bank und Händler. Unser Konzept simuliert also bildlich gesprochen eine elektronische Geldbörse, bei der wahlweise mit namentlich unterzeichneten Schecks oder mit Anonymität gewährleistenden Münzen bezahlt werden kann.

Der Kunde soll sich gegenüber Händler und Bank durch einen geheimen Schlüssel identifizieren, sowie über seine Signatur verifizierbar sein.

Auch Händler und Bank lassen sich über Signaturen identifizieren. Der geheime Schlüssel des Kunden soll auf einer Speicherchipkarte gespeichert werden, die von einem Trustcenter für den Kunden personalisiert wird.

Der vorliegende Artikel besteht aus zwei Teilen. Im ersten Teil wird unser Konzept vom kreditkartenbasierten Bezahlen erläutert, im zweiten Teil wird das banknotenbasierte Bezahlen beschrieben.

## Ansätze zur Verhinderung kryptographischer Angriffe

### Schlüsselwahl

Um das hier teilweise benutzte RSA-Verfahren gegen kryptoanalytische Angriffe zu schützen, sollten folgende Bedingungen erfüllt sein (siehe auch [Bau95]):

1. Das benutzte Modul  $n = p \cdot q$  sollte mindestens in der Größenordnung von 1024 Bit liegen, so daß eine eventuelle Faktorisierung von  $n$  erschwert wird. Ein Superrechner bräuchte zum Entschlüsseln einer Nachricht mit einem 512 Bit Schlüssel ein knappes Jahr. Trotz des relativ geringen Informationswertes, den ein Jahr alte Transaktionsdaten haben würden, erscheint

eine Schlüssellänge von 512 Bit also zu niedrig. Andererseits muß die geringe Geschwindigkeit des RSA- Algorithmus betrachtet werden, die mit steigender Schlüssellänge weiterhin abnimmt. In Anbetracht dieser Tatsachen erscheint eine Schlüssellänge von 1024 Bit durchaus angebracht, da diese nach heutigem Stand von Wissenschaft (Faktorisierungsalgorithmen) und Technik (Rechenleistung) für einen mittelfristigen Zeitraum als sicher gelten kann.

2. Die benutzten Primzahlen  $p$  und  $q$  unterscheiden sich in ihrer Länge um einige Stellen. Somit wird die exhaustive Suche nach einer Darstellung von  $n$  als Differenz von 2 Quadraten sofort vereitelt:

$n = p \cdot q = [(p + q)/2]^2 - [(p - q)/2]^2$  mit ab  $\sqrt{n}$  laufenden Werten für  $(p + q)/2$ . Allerdings ist auch zu beachten, daß die Differenz zwischen  $p$  und  $q$  auch nicht zu groß werden darf: sonst ist ein Faktor von  $n$  klein und somit wäre die Faktorisierung von  $n$  nicht schwer.

3.  $p - 1$  und  $q - 1$  enthalten nur große Primfaktoren.

4.  $ggT(p - 1, q - 1)$  ist sehr klein.

Die Eigenschaften 3 und 4 sind in idealer Weise bei **sicheren Primzahlen** erfüllt. Dabei heißt eine Primzahl  $P$  genau dann sicher, wenn mit  $P$  auch  $(P - 1)/2$  eine Primzahl ist.

5. Sind auch  $p' = (p - 1)/2$  und  $q' = (q - 1)/2$  sichere Primzahlen, so bezeichnet man  $p$  und  $q$  als **doppelt sichere** Primzahlen. Sind die Bedingungen 3, 4 und 5 nicht gegeben, ist ein *Angriff durch Iteration* möglich: Die mit RSA verschlüsselte Nachricht wird vom Angreifer wiederholt mit dem öffentlichen Schlüssel des Empfängers verschlüsselt. Da alle Operationen modulo  $n$  ablaufen, wird nach endlich vielen Schritten wieder der Klartext erhalten, ohne daß der private Schlüssel  $d$  des Empfängers bekannt ist. Werden die Bedingungen 3 bis 5 nicht erfüllt, so ist die Anzahl der Iterationsschritte entsprechend gering und der Klartext kann schnell ermittelt werden. Wählt man für  $p$  und  $q$  jedoch doppelt sichere Primzahlen aus, so ist die Anzahl der Iterationsschritte so enorm groß, daß der Aufwand dieser Angriffsmethode etwa vergleichbar mit der Faktorisierung von  $n$  ist.

### Random Padding

Tritt bei der Benutzung des RSA-Verfahrens der Fall auf, daß eine Nachricht  $m$  verschlüsselt an verschiedene Personen gesendet wird, so kann ein Angreifer aus Kenntnis dieser verschlüsselten Nachrichten

$$m'_1 = m^{e_1} \bmod n_1, m'_2 = m^{e_2} \bmod n_2, \dots, m'_k = m^{e_k} \bmod n_k$$

mit Hilfe des Chinesischen Restsatzes auf die Originalnachricht  $m$  zurückzuschließen. Um diese Klasse von Angriffen abzuwehren, muß also dafür gesorgt werden, daß keine Nachricht zweimal gleich signiert wird. Zu diesem Zweck bietet es sich an, die betreffende Nachricht immer mit einer Anzahl Zufallsbits aufzufüllen (random padding), um gleiche Signaturen zu verhindern.

Es muß immer mit einer Mindestanzahl von Zufallsbits gepadded werden, sonst könnte ein Angreifer bewußt solche Nachrichten einschleusen, die sehr

kurz sind und somit nicht gepadded werden müssen. Einem möglichen Angreifer können aber durchaus Position und Mindestanzahl der Zufallsbits bekannt sein, dies dürfte ihm keinen Vorteil bei der Kryptoanalyse verschaffen.

Ein positiver Nebeneffekt des Random padding ist, daß auch sogenannte *Replay-Attacks* dadurch vermieden werden, daß es niemals zwei gleiche Nachrichten beziehungsweise Signaturen gibt. (Bei einer Replay-Attacke wird eine Nachricht, die bereits einmal gesendet wurde, erneut in das Protokoll geschleust.)

## Kreditkartenbasiertes Bezahlen

In diesem Teil des Artikels wollen wir uns mit dem Ablauf eines Handelsprozesses beim kreditkartenbasierten Bezahlen beschäftigen. Ein solcher Handelsprozeß beim kreditkartenbasierten Bezahlen läuft nun wie folgt ab:

- Nachdem der Kunde Kaufinteresse bekundet hat, sendet ihm der Händler ein signiertes Preisangebot für diesen Artikel. Signiert im Laufe des Einkaufsprozesses eine Instanz eine Nachricht  $m$  mit ihrem privaten Schlüssel  $d$ , so bestimmt sie

$$\text{sig}(m) = m^d \bmod n.$$

Jeder beliebige Teilnehmer des Systems kann die Signatur überprüfen, indem er den öffentlichen Schlüssel der betreffenden Instanz auf  $\text{sig}(m)$  anwendet:

$$\text{sig}(m)^e \bmod n = m^{ed} \bmod n = m.$$

- Um sicher zu gehen, daß ihm ein korrektes Angebot vorliegt, überprüft der Kunde die Signatur des Händlers mittels dessen öffentlichen Schlüssels. Diesen erfragt er bei einem Trustcenter. Dieses überprüft seinerseits die Echtheit der Kundenkarte mit Hilfe der Angaben von Hashwert sowie Hashwert-Signatur auf der Karte, die der Kunde dem Trustcenter mitteilt. Als Hashfunktion können hier bei der praktischen Umsetzung beispielsweise die Verfahren MD5 oder SHA genutzt werden.
- Um die Authentizität des vom Kunden erfragten Schlüssels zu bestätigen, signiert das Trustcenter auch diese Auskunft. Da der öffentliche Schlüssel des Trustcenters fest auf der Teilnehmerchipkarte gespeichert ist, ist es jedem Teilnehmer zu jedem Zeitpunkt des Prozesses möglich, die Signaturen des Trustcenters zu überprüfen.
- Nachdem sich der Kunde nun von der Authentizität des Angebots überzeugt hat, „unterschreibt“ er gewissermaßen einen digitalen Scheck. Mittels des öffentlichen Schlüssels des Verkäufers verschlüsselt er nun mit einer 1024 Bit RSA-Verschlüsselung eine Nachricht, die den gewünschten Betrag, sowie seinen Namen, BLZ und Kontonummer enthält. Anschließend wird die Nachricht noch signiert. Der Anbieter überprüft nun mittels der Informationen des Trustcenters die Signatur des Kunden.

- Nun reicht der Anbieter den erhaltenen Scheck an die Bank weiter, mit der Bitte, seinem Konto den entsprechenden Betrag gutzuschreiben. Nachdem die Bank die Signatur des Kunden unter dem Scheck sowie die Signatur des Händlers geprüft hat, wird der entsprechende Betrag vom Konto des Kunden abgebucht und dem Konto des Händlers gutgeschrieben.
- Die Bank teilt dem Anbieter nun das Ergebnis der Transaktion mit, so möglicherweise das Mißlingen des EinlöSENS eines Schecks. Der Händler überprüft schließlich die Integrität der Information der Bank mittels ihrer Signatur.

Es wäre nun weiterhin sinnvoll, die Nachrichten jeweils mit einem Zeitstempel zu versehen, um zu verhindern, daß beispielsweise ein Händler einen Scheck zweimal bei der Bank einreicht (replay Attacke). In der Praxis ist ein zuverlässiger Zeitstempel für Nachrichten jedoch schwer zu realisieren, da die Systemzeit beispielsweise auf Personalcomputern frei verstellt werden kann.

Statt dessen kann die Nachricht aber mit Zufallsbits aufgefüllt werden. In diesem Fall spricht man von „random padding“. Man erzielt so, wie bereits im Abschnitt „Ansätze zur Verhinderung kryptographischer Angriffe“ beschrieben, zwei positive Effekte auf einmal: zum einen werden replay Attacken verhindert, zum anderen Angriffe nach dem chinesischen Restsatz.

## Ausblicke

Der zweite Teil dieses Artikels in der nächsten Ausgabe beschäftigt sich mit den Aspekten des banknotenbasierten BezahleNS. Insbesondere wird die Erzeugung und die Verifikation von elektronischen Münzen beschrieben. Außerdem enthält dieser Artikel theoretische Betrachtungen zu Hash-Funktionen.

## Literatur

- [Bau95] Friedrich L. Bauer, *Entzifferte Geheimnisse*, Springer Verlag, Berlin Heidelberg 1995
- [Beu97] Albrecht Beutelspacher, *Geheimschriften*, Verlag C. H. Beck, München 1997
- [BSW95] Albrecht Beutelspacher, Jörg Schwenk, Klaus-D. Wolfenstetter, *Moderne Verfahren der Kryptologie*, Verlag Vieweg, Braunschweig/Wiesbaden 1995
- [RS97] Patrick Reichert, Christoph Seidler, *Rechnen mit großen natürlichen Zahlen am Beispiel der Implementation eines RSA-Kryptosystems*, in: *Wurzel* Heft 9+10/97
- [Sch96] Bruce Schneier, *Angewandte Kryptographie*, Addison-Wesley, Bonn 1996

Patrick Reichert, Christoph Seidler  
Halle(Saale)

## Zahlen mit Zahlen - Teil 2

### Mathematische Algorithmen zum sicheren Bezahlen in elektronischen Datennetzen

#### Einleitung

Die fortschreitende Nutzung des Internets für kommerzielle Transaktionen insbesondere die Bezahlung von Waren und Dienstleistungen erfordert dringend die Entwicklung eines Protokollsystems zum digitalen Bezahlen. Derzeit wird weltweit an einer Vielzahl von möglichen Lösungen für dieses Problem gearbeitet. Man muß jedoch feststellen, daß sich bis zum jetzigen Zeitpunkt keine der vorgeschlagenen Lösungen etablieren konnte. Unserer Wettbewerbsbeitrag für den 33. Bundeswettbewerb „Jugend forscht“ 1998 im Fachbereich „Mathematik/Informatik“ soll einen weiteren Beitrag in der Diskussion um einen Standard darstellen. Unser Lösungsansatz zeichnet sich gegenüber anderen Vorschlägen durch eine Zweiteilung aus, zum einen können Zahlungen nach dem Prinzip von Kreditkarten beziehungsweise Schecks geleistet werden, zum anderen aber auch mit sogenannten „elektronischen Münzen“.

Im ersten Teil dieses Artikels wurde das kreditkartenbasierte Bezahlen näher beschrieben, in diesem zweiten Teil soll das banknotenbasierte Bezahlen im Vordergrund stehen.

#### Hashfunktionen

Vor der Beschreibung der mathematischen Algorithmen zum banknotenbasierten Bezahlen müssen noch zwei Funktionsklassen näher beschrieben werden: symmetrische Verschlüsselungsfunktionen zum einen und Hashfunktionen zum anderen.

Gegeben sei eine Verschlüsselungsfunktion  $c = f(m, k)$ , die eine beliebige Nachricht  $m$  mit Hilfe des Schlüssels  $k$  in den Geheimtext  $c$  verschlüsselt. Die Funktion  $f$  heißt *symmetrisch*, wenn  $m = f(c, k)$  gilt, d.h. wenn zur Entschlüsselung des Geheimtextes  $c$  der gleiche Schlüssel  $k$  benutzt werden kann. Beispiele für symmetrische Verschlüsselungsfunktionen sind die DES oder die einfache XOR-Operation: Gilt für den Geheimtext

$$c = m \text{ XOR } k,$$

so erhält man die ursprüngliche Nachricht wie folgt:

$$c \text{ XOR } k = (m \text{ XOR } k) \text{ XOR } k = m.$$

Eine *kryptographische* Hashfunktion, oft auch als *Einweg-Hashfunktion* bezeichnet, besitzt die folgenden 3 Eigenschaften:

- **Kompressionseigenschaft:** Nachrichten beliebiger Länge werden auf Nachrichten einer festen kurzen Länge (Beispiel MD-5: 128 Bit) komprimiert.
- **Einwegeigenschaft:** Obwohl es zu einem Komprimat im allgemeinen sehr viele zugehörige Datensätze gibt, kann man praktisch zu einem vorgegebenen Komprimat auch nicht einen dieser Datensätze konstruieren, d.h. die Hashfunktion ist nicht invertierbar.

- **Kollisionsfreiheit:** Es ist praktisch unmöglich, zwei verschiedene Nachrichten mit dem gleichen Hashwert zu finden.

Somit können Hashfunktionen in geradezu idealer Weise dazu genutzt werden, eine Nachricht zu signieren und eine Signatur zu authentifizieren. Die in unserem Programmpaket implementierte Hashfunktion MD-5 wurde von Ronald L. Rivest entwickelt und besteht aus einer Folge von logischen Befehlen und anderen bitorientierten Transformationen. Für weitere Beschreibungen der MD-5-Hashfunktion vergleiche man [Sch96].

## Banknotenbasiertes Bezahlen

Im folgenden Abschnitt soll das verwendete Verfahren zum Herstellen, Verifizieren und Einlösen von elektronischen Münzen beschrieben werden.

Dieses Verfahren besitzt folgende grundlegende Eigenschaften:

- **Authentizität:** Nur autorisierte Instanzen (etwa die Zentralbanken) dürfen Geld erzeugen können. Heutzutage besitzen nur die Nationalbanken in Deutschland das Recht, neue Münzen und Geldscheine in Umlauf zu bringen.
- **Verifikation:** Jeder Teilnehmer im System muß in der Lage sein, die Echtheit von elektronischen Münzen überprüfen zu können.
- **Anonymität:** Der Besitzer der Münze bleibt beim Bezahlen vollständig anonym, d.h. wenn eine Münze eingelöst wird, kann die Bank nicht feststellen, an wen sie diese ausgegeben hat.
- **Einmaligkeit** (Schutz gegen mehrfaches Ausgeben): Wenn eine Münze zum zweiten Mal eingelöst werden soll, kann die Bank feststellen, an wen diese Münze ausgegeben wurde.

*Bemerkung:* Im folgenden Abschnitt wird davon ausgegangen, daß sowohl der Kunde als auch der Verkäufer ihre Bankgeschäfte bei dem gleichen Geldinstitut abwickeln. Diese einschränkende Bedingung kann aber leicht verallgemeinert werden, da Transaktionen zwischen verschiedenen Bankinstituten schon heute zur Realität gehören.

### Erzeugung von neuen Münzen

Die Bank besitzt für jeden möglichen Münzwert (z.B. 0,10 DM; 0,50 DM; 1 DM; 5 DM; 10 DM; 50 DM; 100 DM) je ein Münzmodul  $n$ , einen öffentlichen Münzschlüssel  $e$  und einen privaten Münzschlüssel  $d$ . Nur die Zahlen  $n$  und  $e$  werden allen Kunden bereitgestellt, die Zahl  $d$  hält die Bank streng geheim.

Das Verfahren zur Münzerzeugung kann in einem Satz zusammengefaßt werden: Ein vom Kunden erzeugter „Münzrohling“ wird durch die Bank mit Hilfe einer „blinden Signatur“ authentifiziert.

Folgende Schritte sind nun zur Münzerzeugung nötig:

1. Der Kunde wählt eine natürliche Zahl  $m$ , die folgende Informationen enthält: Eine für den Kunden eindeutige Identifizierungsnummer der Münze (32 Bit groß) und seine Identität, bestehend aus Namen, Bankleitzahl

und Kontonummer; diese Informationen befinden sich auf der Chipkarte. (Es muß sichergestellt werden, daß stets  $m < n$  gilt.) Würde die Identifizierungsnummer nicht in den Wert von  $m$  einfließen, dann könnte der Kunde nur eine Münze für jeden Münzwert erzeugen. Würde der Kunde nämlich 2 identische Münzen besitzen, so würde die zweite Münze nicht angenommen werden, da das hier vorgestellte System sicherstellt, daß jede Münze nur einmal eingelöst werden kann.

2. Weiterhin wählt er 32 zufällige Zahlen  $k_1, \dots, k_{32}$  und berechnet 32 Geheimtexte  $c_1, \dots, c_{32}$  mit  $c_i = f(k_i, m)$ . Dabei ist  $f$  eine symmetrische Verschlüsselungsfunktion (z.B. DES oder XOR) mit den Schlüsseln  $k_i$ . (Hinweis: Dieses *private key* - Verfahren benötigt im Gegensatz zu den *public key* - Verfahren nur einen Schlüssel, hier  $k_i$ , zur Chiffrierung und Dechiffrierung.)
3. Anschließend werden die Werte  $k_i$  und  $c_i$  (für  $i = 1 \dots 32$ ) gehasht, d.h. es erfolgt die Bestimmung von  $h(k_i)$  und  $h(c_i)$ . (Zur Berechnung des Hashwertes können z.B. die Verfahren MD5, SHA oder RIP-MP benutzt werden.)
4. Die Hashwerte werden nicht im Klartext zur Bank geschickt, sondern „geblendet“. Dazu wählt der Kunde eine zufällige natürliche Zahl  $z$ , welche ein modulares Inverses  $z'$  bezüglich  $n$  besitzt. (Es muß  $zz' \bmod n = 1$  gelten, solche Zahlenpaare sind mit Hilfe des euklidischen Algorithmus leicht zu finden.) Anschließend werden für jeden Index  $i$  folgende Werte bestimmt:

$$\begin{aligned} \text{blend}(h(k_i)) &= h(k_i) \cdot z^e \bmod n \text{ und} \\ \text{blend}(h(c_i)) &= h(c_i) \cdot z^e \bmod n. \end{aligned}$$

Somit hat die Bank nicht die Möglichkeit, die Identität des Kunden und die Hashwerte  $h(k_i)$  und  $h(c_i)$  zu speichern um somit beim Einlösen der Münze zu bestimmen, an wen sie diese ausgegeben hat.

5. Nun beginnt eine Interaktionsphase mit der Bank, die das Ziel hat, die Identität des Kunden für die Bank zweifelsfrei festzustellen. Der Kunde sendet zunächst Angaben zu seiner Person und den gewünschten Geldbetrag der Münze zur Bank. Die Bank wählt 16 Indizes  $i$  zwischen 1 und 32 zufällig aus und fordert den Kunden auf,  $k_i$  und  $c_i$  offenzulegen. Mit Hilfe der Umkehrfunktion der symmetrischen Funktion  $f$  (siehe Schritt 2) kann die Bank das ursprüngliche  $m$  bestimmen. (Hinweis: Es gilt  $m = f(k_i, c_i)$ , da eine symmetrische Funktion gerade die Eigenschaft hat, mit ihrer Umkehrfunktion übereinzustimmen.) Stimmen die in  $m$  enthaltenen Personaldaten des Kunden mit der in diesem Schritt übermittelten Daten überein, so vertraut die Bank dem Kunden und signiert ihm die restlichen 16 Zahlenpaare  $\text{blend}(h(k_i))$  und  $\text{blend}(h(c_i))$  mit ihrem privatem Münzschlüssel  $d$  *blind*, d.h. der Kunde bekommt die Werte

$$\begin{aligned} \text{sig}(\text{blend}(h(k_i))) &= (h(k_i) \cdot z^e)^d \bmod n = h(k_i)^d \cdot z^{ed} \bmod n \\ &= h(k_i)^d \cdot z \bmod n \text{ und} \\ \text{sig}(\text{blend}(h(c_i))) &= (h(c_i) \cdot z^e)^d \bmod n = h(c_i)^d \cdot z^{ed} \bmod n \\ &= h(c_i)^d \cdot z \bmod n, \end{aligned}$$

da nach der grundlegenden Eigenschaft des RSA-Algorithmus  $z^{ed} \bmod n = z$  ist.

Außerdem erfolgt eine Abbuchung des entsprechenden Geldbetrages vom Kundenkonto.

6. Der Kunde multipliziert die erhaltenen Werte mit  $z'$  und erhält die signierten Hashwerte:

$$\begin{aligned} \text{sig}(\text{blend}(\text{h}(k_i))) \cdot z' \bmod n &= \text{h}(k_i)^d \cdot z z' \bmod n \\ &= \text{h}(k_i)^d \bmod n \\ &= \text{sig}(\text{h}(k_i)) \text{ und} \\ \text{sig}(\text{blend}(\text{h}(c_i))) \cdot z' \bmod n &= \text{h}(c_i)^d \cdot z z' \bmod n \\ &= \text{h}(c_i)^d \bmod n \\ &= \text{sig}(\text{h}(c_i)). \end{aligned}$$

Die elektronische Münze besteht nun also aus den 16 Zahlenpaaren  $(\text{sig}(\text{h}(k_i)), \text{sig}(\text{h}(c_i)))$ .

### Verifikation der Echtheit von elektronischen Münzen

Der Kunde kann den Verkäufer nun durch folgende Schritte von der Authentizität seiner Münzen überzeugen:

1. Der Kunde läßt dem Verkäufer die 16 Zahlenpaare  $(\text{sig}(\text{h}(k_i)), \text{sig}(\text{h}(c_i)))$  zukommen. Mit dem öffentlichen Münzschlüssel  $e$  der Bank kann dieser die Signaturen auflösen und erhält die 16 Zahlenpaare  $(\text{h}(k_i), \text{h}(c_i))$ .
2. Der Verkäufer erzeugt 16 Zufallszahlen aus der Menge  $\{0, 1\}$  und fordert den Kunden bei einer 0 auf, den Wert  $k_i$  preiszugeben bzw. analog bei einer 1 den Wert  $c_i$  zu übermitteln. Der Kunde sendet also entweder das Urbild von  $\text{h}(k_i)$  oder das Urbild von  $\text{h}(c_i)$ .
3. Durch Anwendung der Hashfunktion auf die gesendeten Urbilder kann der Verkäufer überprüfen, ob er eine echte Münze vor sich hat oder nicht, das Urbild ist also der Beweis für die Echtheit der Münze.

*Bemerkung:* Um im praktischen Fall die Anonymität des Kunden gegenüber dem Verkäufer zu gewährleisten, haben wir in unserem Protokollsystem einen „MIX“ zwischen Kunden und Verkäufer geschaltet. Der MIX hat nur die Aufgabe, vom Kunden kommende Informationen anonym an den Verkäufer weiterzuleiten; die Zieladresse wird also behalten, der Absender der Nachricht wird lediglich entfernt. Ferner kann ein MIX dazu benutzt werden, Nachrichten von verschiedenen Kunden in einer anderen Reihenfolge weiterzuleiten. Somit ist es einem eventuellen Beobachter nicht möglich zu entscheiden, welcher Kunde mit welchem Verkäufer kommuniziert hat.

### Einlösen von elektronischen Münzen

1. Der Verkäufer reicht bei der Bank sowohl die Münze, als auch die geoffenbarten Urbilder ein, sendet also die folgenden 16 Zahlentripel zu der Bank:  $(\text{sig}(\text{h}(k_i)), \text{sig}(\text{h}(c_i)), k_i \text{ oder } c_i)$ .

2. Die Bank überprüft, ob diese Münze schon einmal eingereicht wurde. Wenn nicht, schreibt sie dem Verkäufer den entsprechenden Münzbetrag auf sein Konto gut.

Wenn die Bank feststellt, daß diese Münze schon einmal eingereicht wurde, so ist die Wahrscheinlichkeit nur  $1 : 2^{16} = 1 : 65536$ , daß der Verkäufer in dem zweiten Fall der Münzannahme genau die 16 entgegengesetzten Zufallsbits erzeugt hat. Es ist also sehr wahrscheinlich, daß es einen Index  $i$  gibt, so daß der Bank nunmehr sowohl der Schlüssel  $k_i$  als auch der Geheimtext  $c_i$  bekannt ist. Daraus kann sie aber  $m$  bestimmen:  $m = f(k_i, c_i)$  und besitzt somit die Identität des Betrügers.

Die Anzahl der Schlüssel  $k_i$  kann theoretisch von 32 auf jeden beliebigen geraden Wert  $2a$  erhöht werden. Gleichzeitig würde die Wahrscheinlichkeit, eine Münze zweimal unbemerkt einzureichen, auf den Wert  $1 : 2^a$  fallen. Dieser Fakt einer „Restunsicherheit“ stellt also kein grundsätzliches Sicherheitsproblem des Systems dar.

### Zusammenfassung

In unserem Artikel haben wir versucht, unser Konzept der elektronischen Geldbörse darzustellen. Insbesondere haben wir uns bemüht, die Punkte aufzuzeigen, in denen unser Konzept über bestehende Lösungsansätze hinausgeht. In der Präsentation unseres Projektes bei dem Wettbewerb „Jugend forscht“ haben wir unser Programmpaket „Krypton“ vorgestellt, welches das Bezahlen wahlweise mit elektronischen Schecks oder mit elektronischen Münzen ermöglicht. Von uns wurden mehrere Komponenten erstellt, die es ermöglichen, alle Instanzen eines Handelsprozesses zu simulieren. Es können also wahlweise die Aufgaben von Kunde, Händler, Bank sowie dem überwachenden Trustcenter wahrgenommen werden.

Bei einer praktischen Realisierung unseres Konzeptes bleibt noch zu klären, wer die Funktion eines Trustcenters übernehmen kann, welches das Schlüsselmanagement übernimmt und für alle Teilnehmer eine persönliche Speicherchipkarte mit den entsprechenden Daten erstellt und zertifiziert. Wir schlagen hierfür speziell ausgebildete Notare vor. Außerdem sei hier noch auf die bereits existierenden gesetzlichen Grundlagen im Informations- und Kommunikationsdienste-Gesetz (IuKGD) beziehungsweise der Signaturverordnung (SigV) verwiesen.

### Literatur

- [Bau95] Friedrich L. Bauer, *Entzifferte Geheimnisse*, Springer Verlag, Berlin Heidelberg 1995
- [Beu97] Albrecht Beutelspacher, *Geheimschriften*, Verlag C. H. Beck, München 1997
- [BSW95] Albrecht Beutelspacher, Jörg Schwenk, Klaus-D. Wolfenstetter, *Moderne Verfahren der Kryptologie*, Verlag Vieweg, Braunschweig/Wiesbaden 1995

- [RS97] Patrick Reichert, Christoph Seidler, *Rechnen mit großen natürlichen Zahlen am Beispiel der Implementation eines RSA-Kryptosystems*, in: *Wurzel* Heft 9+10/97
- [Sch96] Bruce Schneier, *Angewandte Kryptographie*, Addison-Wesley, Bonn 1996

*Patrick Reichert, Christoph Seidler  
Halle(Saale)*