Einführung in Zahlungssysteme mit Doublespender-Identification

Eric Auer und Andreas Franke

19. Juli 2001

Inhaltsverzeichnis

1 Motivation

1.1 Offlinesysteme mit Chipkarten

Der herkömmliche Bargeldverkehr hat durch die Verwendung nicht unterscheidbarer Münzen in fester Stückelung den Vorteil, völlig anonym ablaufen zu können. Geldscheine haben zwar Seriennummern, diese werden aber nur selten geprüft und insbesondere von der Bank normalerweise nicht in Verbindung mit der Identität des Kunden (der Geld abholt oder einzahlt) aufgezeichnet.

Eine direkte Umsetzung des Münzkonzepts in elektronischer Form sind *elektronische Geldbörsen*, oft in Form von Chipkarten. Intern wird in der einfachsten Form lediglich ein Zähler über den „Kontostand" der Karte geführt. Gestückelte Werte werden nicht verwendet.

Das Problem: Ein betrügerischer Kunde könnte die Karte manipulieren, damit sie immer meldet, sie sei noch mit Guthaben geladen. Betrügereien mit Telefonkarten verschiedener Systeme zeigen, dass ein Gerät, welches ständig im Besitz möglicherweise betrügerischer Kunden ist, nur schwer vor Manipulationen und Fälschung geschützt werden kann.

Eine mögliche Lösung: Die Bank kann eigene Zähler für den Inhalt der jeweiligen Karten führen, damit kann eine Manipulation der karteninternen Zähler nachgewiesen werden. Dazu muss aber jede Karte ihrem Besitzer zuzuordnen sein, Anonymität ist nicht möglich. Ein Beispiel für ein nicht anonymes Offlinesystem ist die *ec-Kartenzahlung per Lastschrift*, wo die Kontobewegung nicht direkt stattfindet, sondern die Karte nur als Nachweis, ein bestimmtes Konto zu besitzen verwendet wird.

Eine andere Lösung: Die Bank kann zur Aufladung der Chipkarten *elektronische Münzen* ausgeben, die mit einer Seriennummer versehen sind. Die Verbindung zwischen Nummern und Personen wird dann aufgezeichnet, um später Betrüger entlarven zu können (jede bei der Bank eingezahlte Münze muss dann überprüft werden, ob sie gerade gültig ist, d.h. ausgegeben aber noch nicht wieder eingezahlt wurde). Auch hier geht die

Anonymität verloren.

1.2 Verbesserte anonyme Offlinesysteme

Aufbauend auf in diesem Text vorgestellten Verfahren kann man ein Offlinesystem konstruieren, bei dem Betrüger entlarvt werden können, ehrliche Benutzer aber anonym bleiben. Um Betrug zunächst grundsätzlich zu erschweren, sollte man natürlich trotzdem die verwendete elektronische Geldbörse in Form einer möglichst schwer zu manipulierenden Chipkarte oder vergleichbar konstruieren.

Die Grundidee ist die der *bedingten Anonymität*: Wird eine Münze mehrfach ausgegeben, lässt sich die darin verschlüsselte Identität des Besitzers berechnen. Ein komplettes System (nach Brands) ist in Abschnitt 14.5.3 des Vorlesungsskriptes ausführlich beschrieben, hier werden nur einige Grundideen und das Schnorr Signatursystem erklärt.

1.3 Skizze eines kompletten Systems

Bei Kontoeröffnung wählt der Kunde selbst ein Zahlenpaar (eine geheime Zufallszahl x und eine öffentliche Zahl $h = g^x$, deren diskreter Logarithmus x im Normalfall nur dem Kunden zugänglich ist).

Um seine Geldbörse zu laden, beweist er der Bank, x zu kennen, und lässt die Bank eine (vom Kunden erzeugte) elektronische Münze *blind signieren* . Die allgemeine Vorgehensweise beim Bezahlen auf Basis blind signierter Münzen ist in Abbildung 1 zu sehen (aus dem Themabremservortrag über auf RSA Signaturen aufbauenden Bezahlsystemen).

Die Münze enthält eine mit x verschlüsselte Nachricht, wobei für denselben Wert immer dieselbe Nachricht verwendet werden kann. Da die Bank eine verschlüsselte Form der Münze signiert (blinde Signatur), kann sie die entschlüsselte Münze wie sie später zum Bezahlen verwendet wird den Kunden nicht mehr zuordnen.

Beim Bezahlen wird der Besitz einer geeigneten Münze bewiesen, indem der Empfänger aus seiner Identität und einem zufälligen Wert eine „Challenge" (Herausforderung) konstruiert. Eine passende Antwort zu berechnen gelingt nur dem, der die Münze entschlüsseln kann, die Erfolgsaussichten anderer Ansätze sind vernachlässigbar schlecht. Solange die entschlüsselte Form der Münze nicht erkennbar ist, kann auch der Bezahler nicht identifiziert werden. Der Zahlungskreislauf im Normalfall ist in Abbildung 2 dargestellt.

Zur Verdeutlichung der Idee, wie blinde Signaturen anonyme Offline-Bezahlsysteme ermöglichen, ein Schaubild aus dem Themabremser-Vortrag über RSA Signaturen in Bezahlsystemen:

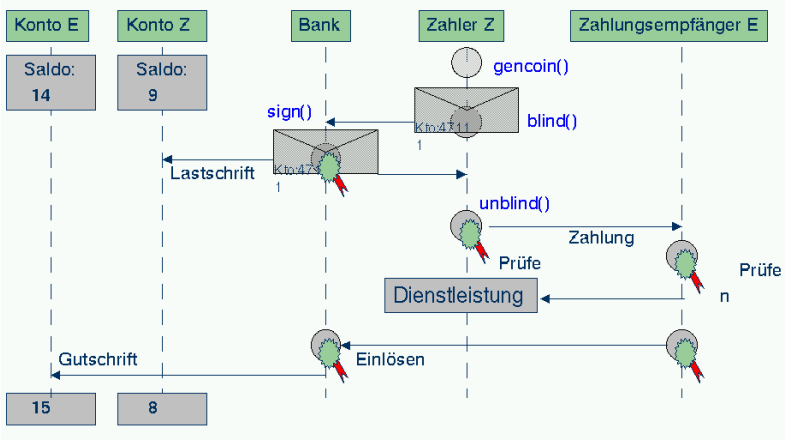

Abbildung 1: Allgemeines Schema: blind signierte elektronische Münzen

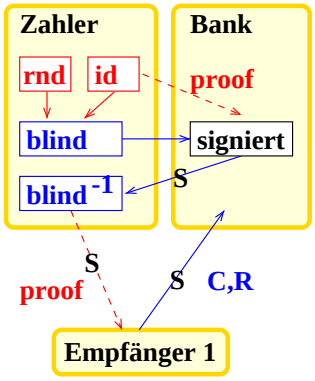

Abbildung 2: Zahlungskreislauf mit anonymem ehrlichen Zahler

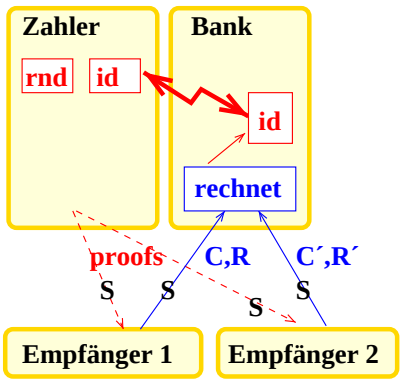

Abbildung 3: Bei Betrug wird die Identität des Betrügers sichtbar

Der Empfänger übermittelt der Bank schliesslich die Challenge und die Antwort (Response) des Bezahlers. Wird nun eine *einmal signierte* Münze mit *mehreren Challenges* verwendet, kann die Münze anhand der Challenge Response Paare entschlüsselt werden, die Identität des Betrügers wird sichtbar. Der mathematische Hintergrund wird im Folgenden grob erklärt, Details finden sich wie immer im Vorlesungsskript. Ein ehrlicher Benutzer muss seine Münzen also *jedesmal neu* signieren lassen (also sein Konto bei der Bank mehrfach belasten), wenn er sie mehrfach verwenden will.

Schematisch ist der Verlust der Anonymität bei Betrug in Abbildung 3 nochmals verdeutlicht.

2 Challenges müssen verschieden sein!

- Szenario: Zahler + zwei Empfänger gegen Bank
  - gleiche Challenge von beiden Empfängern: $C' = C$
  - Bank kann nicht doppelt gutschreiben!
  - Soll einfach der erste das Geld bekommen?
    Nein, sonst:
    - Zahler + Empfänger 2 betrügen Empfänger 1:
    - C wiederverwenden, und schneller zur Bank
- Lösung:
  - Challenges enthalten Id des Zahlungsempfängers
  - Gutschriften nur auf dessen Konto

Noch ein Problem können wir abstrakt behandeln: Was passiert, wenn ein Zahler und zwei Empfänger zusammenarbeiten und dem Zahler dieselben Challenges schicken, d.h. $C' = C$? Die Bank kann das Geld sicherlich nicht beiden Empfängern gutschreiben. Man könnte sagen, daßnur der erste Empfänger der zur Bank geht das Geld bekommt. Aber dann könnten ein Zahler und Empfänger 2 den Empfänger 1 betrügen, indem sie C mehrfach benutzen und schneller zur Bank gehen. Daher wird jede challenge die Identität des Empfängers beinhalten, und eine Gutschrift wird nur möglich sein auf das Konto des Empfängers. (Dies ist ein Spezialfall des „coin key mechanism" aus Abschnitt 14.3.3: Die Response ist gleichzeitig eine Signatur der Identität des Empfängers.)

3 Ideen für Responses

3.1 Kernidee

- *One-time pad*

  1. response: $R = k$ (one-time pad)
  2. response: $R' = id \oplus k$

- gewünschte Eigenschaft ok:
  - R oder R' alleine verrät nichts über id
  - R und R' zusammen offenbaren id vollständig

- Problem:

  nur zwei challenges möglich

3.2 Verfeinerung

- *Lineare Gleichung in zwei Variablen*: $c_1 id + c_2 k$

  $\Longrightarrow$ jetzt viele challenges c_1, c_2 möglich

- gewünschte Eigenschaft ok:
  - eine response: jede id möglich
  - zwei Gleichungen bestimmen beide Variablen eindeutig (falls linear unabhängig)

- Idee verwendet in Brands' system ($\rightsquigarrow$ Ausblick):

zwei Gleichungen in vier Variablen pro Response

$\Longrightarrow$ effiziente Verifikation

Realisierungsideen für die Responses

Eine erste Idee wäre, daßdie erste Antwort R ein one-time pad k ist, und die zweite Antwort $R' = id \oplus k$. Dann gibt offensichtlich jede Antwort alleine für sich noch keine Information über id, aber beide Antworten zusammen offenbaren id vollständig. Dies würde aber nur zwei Challenges erlauben.

Die nächste Idee ist, daßeine Response eine lineare Gleichung in zwei Variablen, wovon eine id ist, z.B. $c_1 id + c_2 k$. Dies ergibt eine große Menge möglicher Challenges (c_1, c_2). Dann ist nach einer Response immer noch jede id möglich, während zwei Gleichungen beide Variablen eindeutig bestimmen (vorausgesetzt die Gleichungen sind linear unabhängig). Im Prinzip wird diese Idee auch in Brands' System verwendet, welches im Script genauer beschrieben wird. Dort liefert allerdings jede Response zwei Gleichungen in vier Variablen — dies erlaubt Brands eine effiziente Verifikation daßeine Response korrekt ist.

4 Schnorr Identifikation

$$|G_q| = q, \qquad G_q < Z_p^*, \qquad < g > = G_q, \qquad x \in Z_q$$

	Prover (Secret: x)	Common input: $q, p, g, h = g^x$	Verifier
1. Random variant	$w \in_R Z_q$ $a := g^w$	$\xrightarrow{a}$	
2. Challenge		$\xleftarrow{c}$	$c \in_R Z_q$
3. Response	$r := w + cx$	$\xrightarrow{r}$	Verify $g^r = ah^c$

Die Schlüsselgenerierung für Schnorr-Signaturen hat als Ausgabeparameter eine Gruppe G_q der Ordnung q, sagen wir eine Untergruppe von Z_p^*, einen Generator $< g >$ von G_q, einen geheimen Schlüssel $x \in Z_q$, und $h = g^x$ als Hauptkomponente des *public key*. Das Signaturschema war ursprünglich erfunden worden als Variante eines Identifikationssystems. In dem Identifikationssystem „beweist" der Besitzer von x „daßer x kennt". (Mehr Details im folgenden.) Das Beweisprotokoll ist oben dargestellt.

Die Idee ist die folgende, und sie wird dieselbe sein in den modifizierten Protokollen:

In Schritt 1 wählt der Besitzer des Geheimnisses, hier x, noch ein anderes Geheimnis derselben Form, hier w. Es wird mehr oder weniger als ein one-time pad für x dienen. Er berechnet auch das Äquivalent des öffentlichen Wertes (hier h) mit w; hier ist es $a = g^w$.

In Schritt 2 sendet der „Verifier" eine Challenge c.

In Schritt 3 muss der „Prover" mit einer Linearkombination zwischen x und w (dem „real secret" und dem „one-time secret"), die von c determiniert ist. Der „Verifier" kann testen, ob dies Linearkombination korrekt ist, indem er das homomorphe Bild für die öffentlichen Werte verifiziert: Falls alles korrekt war, sollte $g^r = g^{w+cx} = g^w(g^x)^c = ah^c$ gelten.

Bemerkung (one-bit challenge, „Zero Knowledge"): In einem anderen, noch grundlegenderen Protokoll ist die Challenge nur ein einziges Bit, aber man muss das Protokoll dann mehrmals wiederholen. Der Vorteil liegt darin, dass die Sicherheit beweisbarer ist (es hat dann eine stärkere Eigenschaft namens „Zero Knowledge", anstatt der informalen Eigenschaft 2, wie sie im folgenden beschrieben ist.

5 Sicherheitsideen

Das Identifikations-Protokoll sollte zwei Eigenschaften haben. Beide sind im Skript nicht formal definiert; es wird auf den Kurs in hoeherer Kryptographie verwiesen.

5.1 „Proof of knowledge"

Prover besteht Verifikation mit signifikanter Wahrscheinlichkeit $\Longrightarrow$ er kennt x

Beweisidee:

- Ann.: Prover hat a gesendet,
  kann zwei challenges c_1 und c_2 beantworten
- Seien r_1, r_2 seine responses
- x und w eindeutig (diskr. Logs von h und a)

- responses r_1 und r_2 werden nur akzeptiert, wenn $r_1 = w + c_1 x$ und $r_2 = w + c_2 x \pmod q$
- auflösen: $(r_1 - r_2) = (c_1 - c_2)x \pmod q$
- $\Longrightarrow$ Prover kennt x.

Falls ein „Prover" die Verifikation mit einer signifikanten Wahrscheinlichkeit bestehen kann, dann mußer x kennen.

In unserem Fall ist die Beweisidee die folgende: Der „Prover" habe bereits a gesendet. Angenommen er kann mindestens zwei von den vielen möglichen Challenges c beantworten, die er jetzt bekommen könnte. Seien diese Challenges c_1 und c_2, und die Antworten die er senden würde seien r_1 bzw. r_2. Die Werte x und w sind eindeutig bestimmt als diskrete Logarithmen von h und a. Nach einem Lemma aus dem Skript (XXX Proposition 1 from Section 5.5.1, wo ist das?) bestehen r_1 und r_2 den Verifikationstest nur dann, wenn tatsächlich $r_1 = w + c_1 x$ und $r_2 = w + c_2 x (modq)$ gelten. Also kann man aus $(c_1, c_2, r_1, r2)$ leicht x berechnen, indem man dieses Gleichungssystem auflöst, d.h. $(r_1 - r_2) = (c_1 - c2)xmodq$. Damit kennt er x. (Dies war nur ein Beweis, denn z.B. Wahrscheinlichkeiten wurden nicht betrachtet, und auch nicht die Frage warum „berechnen können" eine geeigneter Begriff von „Wissen" ist.)

5.2 „Kein nützliches Wissen nach außen"

- auch nach vielen Identifikationen:
  Angreifer erfährt nichts sinnvolles über x
  (x immer gleich, aber w jedesmal verschieden)
- Grundidee:
  für jedes c ist w ein one-time pad auf x
  (in der response r die der Angreifer bekommt)
- aber: kein wirklicher Beweis

Sogar wenn der „Prover" viele solche Beweise durchführt, macht es dem Empfänger nicht leichter x herauszubekommen, oder auch nur irgendwelche sinvollen Dinge mit x anzustellen. Die Grundidee hier ist: für jedes spezifische c ist der Wert w ein one-time pad zu x, in der Response die der Angreifer bekommt. Dies ist jedoch kein wirklicher Beweis, da x nicht komplett unsichtbar ist, da der Angreifer auch a sieht.

6 Vom Ident.-Protokoll zur Signatur

- Gegeben:
  - 3-Schritt Identifikations-Protokoll
  - mit nur einer zufälligen Challenge

  $\Longrightarrow$ Signaturverfahren

- Trick: Signierer berechnet c selbst: $c = hash(m, a)$
  (m: zu signierende Nachricht), wobei $hash$
  - zumindest einweg
  - hoffentlich pseudozufällig
    $\Longrightarrow$ nur dann so gut wie Identifikations-Protokoll
  - öffentlich (wg. Signaturtest)

  $\Longrightarrow$ Sicherheit nicht bewiesen!

- Signaturtest hier: $g^r = ah^c$ mit $c = hash(m, a)$

Es gibt einen allgemeinen Trick von Fiat und Shamir, wie man aus einem 3-Schritt-Identifikationsprotokoll, bei dem der „Verifier" nur eine zufällige Challenge c sendet, ein Signaturverfahren machen kann. Die Qualität dieses Tricks ist aber nicht bewiesen.

Der Trick besteht darin, daßder Signierer c selbst berechnet als $c = hash(m, a)$, wobei m die zu signierende Nachricht ist, und $hash$ eine Hashfunktion, mindestens einweg. Man hofft, daßdie Hashfunktion irgendwie pseudozufällig ist, so daßdas Resultat c genauso gut wie das tatsächlich zufällige c im Identifikationsprotokoll ist. Dies ist jedoch nur heuristisch, da $hash$ öffentlich ist (sonst könnte die Signatur nicht getestet werden), und damit kann es natürlich jeder von einer echten Zufallsfunktion unterscheiden.

Der Signaturtest ist jetzt: $g^r = ah^c$ für $c = hash(m, a)$.

7 Ausblick: Erweiterungen und Probleme

7.1 Erweiterung zu einem kompletten System

Bisher haben wir nur einige Grundbausteine gesehen. In diesem Abschnitt wird kurz erklärt, welche Erweiterungen für ein komplettes Offline-Bezahlsystem mit Anonymität und Doublespender-Identification („Betrügerentlarvung") nötig sind.

Über das Schnorr Identifikationsprotokoll kann eine Partei der anderen beweisen, eine Geheimzahl x zu *kennen*, ohne x selbst preisgeben zu müssen. Nur eine öffentliche Zahl $h = g^x$ ist allen Parteien bekannt, da das Ziehen diskreter Logarithmen schwer ist, lässt sich daraus jedoch x nicht berechnen.

Wir haben gesehen, dass dieses Challenge-Response Verfahren zu einem *Signaturverfahren* umgebaut werden kann, indem die Challenge von der beweisenden Partei (Prover) selbst aus einem Hashwert von zu signierender Nachricht und dem öffentlichen One-Time-Pad $a = g^w$ erzeugt wird (w ist der eigentliche One-Time-Pad, der den Wert x schützt und zufällig für jede Nachricht neu gewählt wird).

Eine verbesserte Variante davon sind die in Abschnitt 14.5.3.B des Folienskriptes beschriebenen *Chaum-Pedersen Signaturen*: Der Signierer übermittelt verschlüsselte Formen der Nachricht (das commitment $z = m^x$) und des One-Time-Pad ($a = g^w$), beide Parteien kennen die verschlüsselte Form des Signierergeheimnisses ($h = g^x$). Der Empfänger antwortet mit einer für den Signierer nicht vorhersehbaren Challenge (c), der Signierer kann am Ende sowohl beweisen, x zu kennen als auch z unter Verwendung von x erzeugt zu haben. Die Signatur *verbindet Nachricht und Besitzer* von x.

Für den Einsatz in anonymen Systemen wird in Abschnitt 14.5.3.C das Protokoll zur *blinden Chaum-Pedersen Signatur* erweitert: Über mehrere zufällig gewählte „Blendfaktoren" (blinding factors, s, t, u, v) wird dem Signierer die Information verborgen, welche Nachricht er wann signiert hat. Der Empfänger blendet die Nachricht und lässt sie in geblendeter Form vom Signierer unterzeichnen, der Empfänger kann daraus eine Signatur der ungeblendeten Nachricht erzeugen.

Für den Einsatz als Bezahlsystem nach Brands werden die Münzen so erzeugt, dass die Identität sowohl in der geblendeten als auch der ungeblendeten Form der Nachricht (Münze) noch soweit enthalten ist, dass bei Betrug die Identität des Betrügers rekonstruierbar ist. Das wird in Abschnitt 14.5.3.D beschrieben.

Zuletzt beschreibt Abschnitt 14.5.3.E die Struktur des fertigen Systems, wie bereits in Abschnitt 1.3 dieses Textes skizziert: Jeder Kunde wählt eine geheime Identität id und erzeugt eine öffentliche Form $h = g^{id}$ daraus. Später kann er mit dem Schnorr Identifikationsprotokoll beweisen, id zu kennen. Ausserdem wählt er für die verschiedenen Stückelungen Kennziffern. Kennziffern und h fliessen in die Münzen ein (jedem Kunden reicht also pro Wert eine Münze!). Wann immer der Zahler Münzen zum Ausgeben braucht, lässt er sich von der Bank gegen Gebühr eine

entsprechende *Anzahl von Signaturen* auf seine Münzen geben. Sobald eine einmal signierte Münze mehrfach ausgegeben wird, kann mit den beim Challenge-Response Beweis der Signatur gewonnenen Informationen (zwei Challenges für dieselben Daten) die Münze entschlüsselt und so der Betrüger entlarvt werden. Im Folienskript sind die genauen Methoden und Gleichungen ausführlich beschrieben – dieser Text vermittelt nur die Grundidee und erklärt das Schnorr Identifikationsprotokoll.

7.2 Verbleibende Probleme

Wie im letzten Abschnitt und in 1.3 bereits angedeutet, wird Betrug aufgedeckt, indem er die *Entschlüsselung der Münze* ermöglicht. Damit kann jeder, der den Betrug nachweisen kann (zum Beispiel die Bank, oder zwei kooperierende Empfänger die dieselbe signierte Münze bekommen haben und so betrogen wurden) sich als ursprünglicher Besitzer der Münze ausgeben.

Eine mehrfach ausgegebene signierte Münze kann damit leicht *weiter vervielfältigt* werden, weiterer Betrug fällt leicht und die weiteren Täter bleiben dennoch unentdeckt (es ist nur zu erkennen, wem die vervielfältigte Münze *ursprünglich* gehört hat).

Da die Bank für einen Kunden und eine Stückelung immer dieselbe Münze neu signiert, ist auch unklar, bei welcher Gelegenheit der Betrug begonnen hat. Eine einmal betrügerisch vervielfältigte Münze kann zu Lasten ihres Besitzers (der immerhin mit dem Betrug angefangen haben muss, sofern seine geheime Identitätsnummer x nicht auf anderem Weg schon bekannt geworden ist) zu weiterem Betrug verwendet werden, ohne dass das *Ausmass* des Schadens direkt feststellbar ist. Der weitere Betrug wird immer dem ursprünglichen Besitzer angelastet, obwohl zu diesem Zeitpunkt auch weitere Personen dafür verantwortlich sein könnten.

7.3 Praktischer Einsatz

Trotz der genannten Probleme stellt das System nach Brands ein sinnvolles System für anonyme Offlinezahlungen dar: Wenn *einzelne* elektronische Geldbörsen manipuliert werden (was durch den Einsatz schwer manipulierbarer Chipkartenkonstruktionen nur wenigen Betrügern gelingen sollte), lassen sich diese identifizieren. Es bricht nicht gleich das *komplette* System zusammen, wie das in anderen, speziell auf symmetrischen Verfahren aufbauenden Chipkartensystemen möglich wäre.

Ein weitgehend auf Brands System aufgebautes Zahlungssystem wurde im CAFE Projekt praktisch implementiert ([BBCM1_94]).