GI-Fachtagung Verläßliche Informationssysteme (VIS'91), März 1991, Darmstadt, Informatik-Fachberichte 271, Springer-Verlag, Heidelberg 1991, 302-320

Unbedingte Unbeobachtbarkeit mit kryptographischer Robustheit

Birgit Pfitzmann, Michael Waidner

Institut für Rechnerentwurf und Fehlertoleranz, Universität Karlsruhe
Postfach 6980, D-W7500 Karlsruhe 1, Bundesrepublik Deutschland

Kurzfassung: Mit dem DC-Protokoll von D. CHAUM sollen Teilnehmer über ein beliebiges Kommunikationsnetz Nachrichten unbeobachtbar senden und empfangen können. Dieses Protokoll hat zwei Schwachpunkte, die im folgenden betrachtet und beseitigt werden:

1. Das Protokoll ist nicht robust: Jeder Teilnehmer kann das DC-Protokoll unbeobachtbar und dauerhaft stören. Wir stellen ein *Fallenprotokoll* vor, das Störer identifiziert und von der weiteren Teilnahme ausschließt. Es setzt ein zuverlässiges Verteilnetz voraus und nimmt an, der Störer verfüge nur über beschränkte Berechnungsfähigkeiten (z.B. könne natürliche Zahlen mit zwei sehr großen Primfaktoren nicht faktorisieren). Unser Protokoll ist eine Verbesserung eines Protokolls von D. CHAUM.

2. Während im DC-Protokoll die Unbeobachtbarkeit des Sendens von keinerlei Annahmen über den möglichen Angreifer abhängt (unbedingte Senderunbeobachtbarkeit), setzt die des Empfangens zuverlässige Verteilung voraus. Diese Annahme ist z.B. in Sternnetzen unrealistisch. Wir stellen ein Protokoll vor, das *Unbeobachtbarkeit* unter der *Zusammenhangsannahme* (= gute Teilnehmer können nicht daran gehindert werden, paarweise miteinander zu kommunizieren) und *kryptographische Robustheit* kombiniert. Dazu ersetzen wir die angenommene zuverlässige Verteilung durch *robuste fail-stop Byzantinische Übereinstimmung*: Diese garantiert zuverlässige Verteilung, falls der Angreifer kryptographisch beschränkt ist und die Zusammenhangsannahme zutrifft; verfügt er über unerwartet große Berechnungsfähigkeiten, so stoppt das Protokoll. Sollen Unbeobachtbarkeit und Robustheit garantiert werden, so ist die Zusammenhangsannahme notwendig.

Im Anhang beschreiben wir ein Protokoll zur *adaptiven Byzantinischen Übereinstimmung*, d.h. Byzantinische Übereinstimmung, die nur gestört werden kann, wenn ein Angreifer sowohl mindestens ein Drittel (bzw. die Hälfte) aller Teilnehmer kontrolliert als auch Signaturen fälschen kann.

Schlagwörter: Byzantinische Übereinstimmung, Kommunikationsnetze, Kryptographie, Fehlertoleranz, Datenschutz, zuverlässige Verteilung, Sicherheit, Unbeobachtbarkeit.

Einige jüngere Kryptographen ...

verbringen einen Abend in einer Diskothek. Sie fühlen sich recht locker und beschließen, einander zu sagen, wen unter ihnen sie für die faszinierendsten Tänzer und besten Kryptographen halten. Alle Hemmungen haben sie aber noch nicht verloren. So beschließen sie, sich diese Meinungen anonym zu sagen, eingedenk dessen, daß ihre Chefs einst in einem drei-Sterne-Restaurant ein vorzügliches Protokoll für solche Zwecke erfanden [Chau_88].

Sie haben Schwierigkeiten: Die Musik ist laut, die Dunkelheit wird nur durch Blitzlichter unterbrochen; die einzige Kommunikationsform ist, einander paarweise in die Ohren zu schreien. Aber der Eintritt war teuer, so daß sie bleiben möchten. Auch fürchten sie, daß manche von ihnen, besorgt, daß niemand sie nennen wird, die Konversation stören möchten. Zum Glück haben ihre Chefs auch Ideen erwähnt, wie Störer vom Eßtisch ausgeschlossen werden können.

Nun wird ein Kryptograph, der sich wenigstens für einen guten Tänzer hielt und daher nicht störte, ausgeschlossen; empört stürmt er aus der Diskothek. In der plötzlichen Stille draußen fällt ihm auf, daß der Lärm der Grund für seine Demütigung sein könnte. Daher entwickelt er ein Protokoll, um die Eßtisch-Unterhaltung in der Diskothek zu simulieren, das ihn in Zukunft vor solchen Erfahrungen

bewahren wird, selbst wenn alle kryptographischen Annahmen falsch sind und beliebig viele seiner Kollegen mogeln.

1 Einleitung

Für öffentliche Netze wird aus Datenschutzgründen oft verlangt, daß Teilnehmer *unbeobachtbar* Nachrichten senden und empfangen können, d.h. daß nicht zu beobachten ist, wer welche Nachricht sendet bzw. empfängt, und insbesondere nicht, wer mit wem kommuniziert. Insbesondere ist dies eine Voraussetzung, wenn Transaktionen, z.B. der Kauf von Informationen aus Datenbanken, *anonym* stattfinden sollen [Chau_85, PfPW_88, PWP_90]. (Daneben würde man die Nachrichteninhalte in solchen Fällen natürlich verschlüsseln.)

Der Ansatz, der die beste Unbeobachtbarkeit verspricht, ist das DC-Protokoll[1] von DAVID CHAUM [Chau_88]. (Allerdings setzt es voraus, daß das zugrundeliegende Kommunikationsnetz wesentlich mehr Bandbreite bietet als die betrachteten Kommunikationsdienste benötigen, ist also vor allem für ein Breitband-ISDN geeignet.)

Wir stellen im folgenden Verbesserungen des DC-Protokolls hinsichtlich seiner Sicherheit vor. In §1.1 führen wir einige Sicherheitsbegriffe ein, die zur Beschreibung der Qualität der folgenden Protokolle benötigt werden. In §1.2 beschreiben wir kurz das ursprüngliche DC-Protokoll und wie es effizient zur Übertragung von Nachrichten verwendet werden kann. §1.3 nennt die betrachteten Probleme und gibt einen Überblick über das Papier.

1.1 Sicherheitsbegriffe

Ein Teilnehmer, der sich streng an sein Protokoll hält und geheime Information für sich behält, wird **gut** genannt, ein Teilnehmer, der dies nicht tut, **böse**. Hinsichtlich ihrer Kooperation wird der schlimmste Fall betrachtet: sie koordinieren ihr Verhalten und tauschen alle geheime Information aus. Deswegen werden sie oft zusammen als „der **Angreifer**“ bezeichnet.

Unbeobachtbarkeit des Sendens bzw. Empfangens einer Nachricht bedeutet, daß (für den gerade betrachteten Angreifer) nicht feststellbar ist, wer diese Nachricht gesendet hat bzw. für wen sie bestimmt ist. **Robustheit** eines Kommunikationsprotokolls bedeutet, daß der Angreifer die guten Teilnehmer nicht dauerhaft am kommunizieren hindern kann.

Man möchte jede Art von Sicherheit (hier also Unbeobachtbarkeit und Robustheit) gern unter möglichst allgemeinen Voraussetzungen garantieren.

Das erste Kriterium ist, ob ein Verfahren, mathematisch betrachtet, überhaupt nicht gebrochen werden kann, oder *im Prinzip* wohl, jedoch nur mit unvertretbar großem Aufwand. Der Aufwand zum Brechen wird dann durch einen wählbaren Sicherheitsparameter bestimmt, und es wird angenommen, der Angreifer könne nur solche Berechnungen ausführen, deren Aufwand polynomial in diesem Sicherheitsparameter ist. Ein solcher Angreifer wird **kryptographisch beschränkt** genannt, einer mit *unbeschränkten* Berechnungsfähigkeiten **kryptographisch unbeschränkt**. Entsprechend bedeutet **(nicht)kryptographische Sicherheit** Sicherheit gegen einen kryptographisch (un)beschränkten Angreifer

Um die Sicherheit eines konkreten Verfahrens nachzuweisen, wird bislang stets eine zusätzliche **kryptographische Annahme** benötigt, z.B. die, daß es in polynomialer Zeit nicht möglich sei, natürliche Zahlen bestehend aus zwei sehr großen Primfaktoren einer bestimmten Gestalt (BLUM-Zahlen) zu faktorisieren [GoMR_88]. Diese **Faktorisierungsannahme** ist zwar allgemein als richtig akzeptiert, aber letztendlich unbewiesen. Für den praktischen Einsatz muß natürlich ein konkreter Sicherheitsparameter gewählt werden, wodurch die asymptotische kryptographische Annahme in eine konkrete verwandelt wird, z.B. daß es zur Zeit praktisch unmöglich ist, BLUM-Zahlen der Länge 800 bit zu faktorisieren.

Ein zweites Kriterium ist, ob der Angreifer **anzahlmäßig beschränkt** sein muß, z.B. ob höchstens die Hälfte aller Teilnehmer böse sein darf.

---

[1] So benannt nach dem motivierenden Beispiel aus [Chau_88], den "*Dining Cryptographers*".

Ein drittes Kriterium ist, welche physischen Annahmen gemacht werden müssen; hier über das Netz, auf dem das DC-Protokoll ausgeführt wird. Insbesondere wird unterschieden, ob zuverlässige Verteilung vorausgesetzt werden muß. **Zuverlässige Verteilung** garantiert folgendes, wenn ein Sender P_s eine Nachricht N senden will: Jeder Teilnehmer P_i empfängt tatsächlich eine Nachricht N_i; falls P_s gut ist, gilt für alle guten Teilnehmer P_i jeweils $N_i = N$ (*Korrektheit*); und auch sonst gilt für alle guten Teilnehmer P_i und P_j jeweils $N_i = N_j$ (*Konsistenz*) [PeSL_80]. Die Annahme zuverlässiger Verteilung ist in vielen Netzen unrealistisch, z.B. in Sternnetzen. Möglichkeiten zur Realisierung zuverlässiger Verteilung werden in §3.1 kurz beschrieben.

Im folgenden nennen wir Sicherheit **unbedingt**, wenn keine dieser Annahmen gemacht werden muß. Insbesondere wird immer davon ausgegangen, daß kein physischer Abhörschutz besteht, d.h. alle Verfahren tolerieren, daß der Angreifer alle Leitungen abhört.

1.2 Das DC-Protokoll

Das **DC-Protokoll** von DAVID CHAUM [Chau_88] sollte auf einem beliebigen Kommunikationsnetz unbeobachtbares Senden und Empfangen von Nachrichten ermöglichen. Es besteht aus zwei Teilen, überlagerndem Senden und zuverlässiger Verteilung.

Überlagerndes Senden: Die Unbeobachtbarkeit des Sendens wird durch *überlagerndes Senden* erreicht, das einen *unbeobachtbaren, additiven Mehrfachzugriffskanal* realisiert:

Das Protokoll wird in **Runden** ausgeführt. Für jede Runde r wählen die Teilnehmer P_i, $i = 1, \ldots, n$, jeweils ein **Nachrichtenzeichen** M_i^r aus einem Alphabet F, einer endlichen abelschen Gruppe.

Gemeinsam berechnen sie die **globale Summe** $S^r = M_1{}^r + \ldots + M_n{}^r$ derart, daß die einzelnen Summanden *geheim* bleiben: Hierfür muß jedes Paar $\{P_i, P_j\}$ von Teilnehmern zuvor über einen sicheren Kanal einen geheimen, zufällig gewählten **Schlüssel** K_{ij}^r $(=K_{ji}^r)$ aus F ausgetauscht haben. Jeder Teilnehmer P_i berechnet damit seine **Ausgabe** $O_i^r = M_i^r + (K_{1i}{}^r + \ldots + K_{i-1,i}{}^r) - (K_{i+1,i}{}^r + \ldots + K_{in}{}^r)$. Diese Ausgaben werden addiert, was wegen

$$O_1{}^r + \ldots + O_n{}^r = M_1{}^r + \ldots + M_n{}^r$$

die globale Summe S^r ergibt.

Die Geheimhaltung der Summanden M_i^r ist unbedingt (nur müssen die Schlüssel $K_{jk}{}^r$ geheimgehalten werden). Selbst ein kryptographisch unbeschränkter Angreifer, der die Kommunikation zwischen allen Teilnehmern abhört und eine beliebige Teilmenge der Teilnehmer kontrolliert, erhält keine Information über die Summanden außer S^r [Chau_88, Pfit_90].

Alle Teilnehmer sollten die globale Summe S^r erhalten. War nur ein Nachrichtenzeichen $M_i^r \neq 0$, so ist $S^r = M_i^r$, d.h. dieses Zeichen wurde *kollisionsfrei* gesendet. Aus der Geheimhaltung der Summanden folgt dann, daß unbeobachtbar ist, welcher Teilnehmer P_i der Sender von M_i^r war.

Wenn in §2 Robustheit gewährleistet werden soll, und wenn ab einer bestimmten Runde sicher ist, daß wenigstens einer von zwei Teilnehmern P_i oder P_j böse ist, so dürfen sie ab dieser Runde ihre gemeinsamen Schlüssel K_{ij}^r nicht mehr verwenden. (Für die Unbeobachtbarkeit ist ein Schlüssel mit einem bösen Teilnehmer ja sowieso nutzlos.) Die jeweils noch verwendeten Schlüssel werden durch einen **Schlüsselgraphen** beschrieben. Darin sind zwei Teilnehmer genau dann durch eine Kante verbunden, wenn ihre gemeinsamen Schlüssel noch verwendet werden.[2]

Zuverlässige Verteilung: Die Unbeobachtbarkeit des Empfangens wird durch zuverlässige Verteilung der globalen Summen S^r sichergestellt.

---

[2] Ist a priori einiges über den Angreifer bekannt, so muß dieser Schlüsselgraph bereits anfangs nicht vollständig vermascht sein. Für die Unbeobachtbarkeit genügt es, daß der Teilgraph aller guten Teilnehmer zusammenhängend ist [Chau_88].

(In [Chau_88] wurde die Zuverlässigkeit der Verteilung nur implizit vorausgesetzt; die Notwendigkeit der Konsistenz der Verteilung für die Unbeobachtbarkeit wurde in [Waid_90, WaPf_89] gezeigt.[3])

Reservierung: Der Zugriff auf den additiven Kanal des DC-Protokolls muß durch ein *Mehrfachzugriffsprotokoll* geregelt werden, das die Unbeobachtbarkeit erhält [BoBo_90, Chau_88, Pfit_85, Pfit_90]. Unsere Protokolle zur Gewährleistung von Robustheit setzen eine *Reservierungstechnik* voraus: Eine feste Anzahl von Zeichen wird zu einer **Nachricht** zusammengefaßt. Jede Nachricht wird in aufeinanderfolgenden Runden übertragen, die zusammen einen **Slot** bilden. Einer Anzahl von Slots geht eine Reservierungsphase voran.

Wesentlich zum Verständnis des folgenden sind zwei Reservierungstechniken: Sei jeweils R die Anzahl der durch eine Reservierungsphase zu reservierenden Slots.

1. Reservierungsleistentechnik: In der einfacheren Variante wird $F = Z_2$ vorausgesetzt[4] [Chau_88]. Eine Reservierungsphase besteht aus R^2 Runden, genannt *Reservierungsrunden*. Für jede zu reservierende Nachricht wählt ein Teilnehmer zufällig einen Index $x \in \{1, \ldots, R^2\}$ einer der R^2 folgenden Reservierungsrunden. In allen so ausgewählten Reservierungsrunden verwendet er das Nachrichtenzeichen 1, in allen anderen 0. Der Reservierungsphase folgen so viele Slots, wie es Reservierungsrunden mit globaler Summe 1 gab. Hatte ein Teilnehmer in der r-ten unter den Reservierungsrunden mit globaler Summe 1 eine 1 gesendet, so gilt der r-te nachfolgende Slot als für ihn reserviert.

Verwendet man statt $F = Z_2$ die Gruppe $F = Z_R$, so werden *Reservierungskollisionen* deterministisch erkannt [Pfit_85]. (Bei Z_2 wird z.B. nicht bemerkt, wenn drei Teilnehmer denselben Slot reserviert haben.)

2. Überlagerndes Empfangen nach BOS und DEN BOER [BoBo_90][5]: Sei $F = \mathrm{GF}(p)$[6], p prim und $p > R$. Eine Reservierungsphase besteht aus R Reservierungsrunden. Um einen Slot zu reservieren, wählt ein Teilnehmer zufällig ein Nachrichtenzeichen aus F, genannt *Reservierungsnachricht*. In der r-ten Reservierungsrunde sendet jeder Teilnehmer die r-te Potenz seiner Reservierungsnachrichten. Aus den damit übertragenen R Potenzsummen kann die Menge aller Reservierungsnachrichten (mit den richtigen Vielfachheiten) berechnet werden. Die arithmetische Ordnung der Reservierungsnachrichten bestimmt die Reihenfolge, in der die nachfolgenden R Slots für die einzelnen Teilnehmer reserviert sind. Die Wahrscheinlichkeit von Reservierungskollisionen wird durch Wahl eines großen Körpers exponentiell klein.

Nutzung eines DC-Netzes: Zur sinnvollen Nutzung des DC-Protokolls benötigt man Verfahren zur *impliziten Adressierung* [PfPW_88, Pfit_90]: Durch sie kann eine Nachricht so codiert werden, daß nur der intendierte Empfänger sie lesen und als an ihn adressiert erkennen kann (N.B.: alle Nachrichten werden verteilt!).

Die implizite Adressierung und erst recht alle „höheren" Protokolle (z.B. wie sich für manche Zwecke die Kommunikationspartner doch wieder identifizieren können) sind von den folgenden Protokollen *unabhängig*, so daß sie nicht betrachtet werden.

1.3 Probleme und Überblick

Das DC-Protokoll hat zwei offensichtliche Schwachstellen, die gemeinsam im folgenden weitgehend beseitigt werden:

---

3 In [Waid_90, WaPf_89, LuPW_90] sind Erweiterungen des DC-Protokolls beschrieben, die ohne zuverlässige Verteilung auskommen, also Unbeobachtbarkeit ohne jede Annahme garantieren. Diese Erweiterungen sind jedoch mit Robustheit nicht vereinbar.

4 Z_k für $k \in \mathbb{N}$ bezeichnet eine zyklische Gruppe der Ordnung k. Z_k kann interpretiert werden als Menge $\{0,1, \ldots, k-1\}$ mit der üblichen Addition modulo k als Gruppenoperation.

5 Eine algorithmisch einfachere Technik zum überlagernden Empfangen ist in [Pfit_90] beschrieben; für unsere Zwecke ist diese allerdings weniger gut geeignet.

6 $\mathrm{GF}(p)$ bezeichnet einen endlichen Körper (GALOIS-Feld) mit p Elementen. Ist p prim, so kann $\mathrm{GF}(p)$ interpretiert werden als Menge $\{0,1, \ldots, p-1\}$ mit der üblichen Addition und Multiplikation modulo p.

1. Robustheit: Jeder Teilnehmer kann das DC-Protokoll unbeobachtbar und dauerhaft *stören*, indem er z.B. für die Runden der von anderen Teilnehmern reservierten Slots statt, wie vorgeschrieben, $M_i^r = 0$ ein zufällig gewähltes Nachrichtenzeichen verwendet.

In [Chau_88 §2.5] wurde bereits ein Protokoll skizziert, das Unbeobachtbarkeit unter der Annahme zuverlässiger Verteilung garantieren sollte, und Robustheit, wenn der Angreifer zusätzlich kryptographisch beschränkt ist. In §2.1 wiederholen wir kurz dieses Verfahren. Wir zeigen, daß es einige Änderungen und Verfeinerungen benötigt, bevor es sicher ist (selbst unter der Annahme zuverlässiger Verteilung). Wir diskutieren einige davon und schlagen in §2.2 ein verbessertes **Fallenprotokoll** vor.

2. Zuverlässige Verteilung: Die Unbeobachtbarkeit des Empfangens wie auch die Fallenprotokolle zur Gewährleistung der Robustheit setzen *zuverlässige Verteilung* voraus. Diese Annahme wird in §3 beseitigt.

Zunächst (§3.1) betrachten wir, was erreicht wird, wenn zuverlässige Verteilung wie üblich durch Byzantinische Übereinstimmung implementiert wird. Die Unbeobachtbarkeit bleibt dabei aber bedingt.

In §3.2 stellen wir zunächst ein Protokoll zur **robusten fail-stop Byzantinischen Übereinstimmung** vor, d.h. ein kryptographisch sicheres Protokoll für zuverlässige Verteilung mit folgender zusätzlichen Eigenschaft: Sobald ein Angreifer über unerwartet große Berechnungsfähigkeiten verfügt und die Übereinstimmung stört, erkennen alle anderen Teilnehmer dies. Dieses Protokoll wird mit dem Fallenprotokoll kombiniert. So erhalten wir Unbeobachtbarkeit ausschließlich unter der Annahme, daß der Angreifer gute Teilnehmer nicht am kommunizieren auf dem zugrundeliegenden Netz hindern kann, und Robustheit, wenn der Angreifer zusätzlich kryptographisch beschränkt ist.

Im Anhang beschreiben wir, aufbauend auf der robusten fail-stop Byzantinischen Übereinstimmung, erstmals ein Protokoll zur **adaptiven Byzantinischen Übereinstimmung**; diese gewährleistet zuverlässige Verteilung, außer wenn der Angreifer sowohl mindestens ein Drittel (bzw. die Hälfte) aller Teilnehmer kontrolliert als auch Signaturen fälschen kann.

Große Teile des folgenden sind ausführlicher in [WaPf_89] beschrieben.

2 Unbeobachtbarkeit mit kryptographischer Robustheit unter der Annahme zuverlässiger Verteilung

Eine Störung des DC-Protokolls kann nicht verhindert werden. Man kann daher nur versuchen, Angreifer im nachhinein zu lokalisieren und von der weiteren Teilnahme am DC-Protokoll auszuschließen[7]. **Robustheit** heißt daher, daß die vom Angreifer kontrollierten, störenden Teilnehmer nach einer begrenzten Anzahl von Störungen mit großer Wahrscheinlichkeit vom DC-Protokoll ausgeschlossen werden.

Durch Verfahren zur Gewährleistung von Robustheit darf die Unbeobachtbarkeit nicht verringert werden. Insbesondere darf kein guter Teilnehmer als Sender oder Empfänger einer (echten) Nachricht identifiziert werden, und kein guter Teilnehmer darf vom DC-Protokoll ausgeschlossen werden (denn hierdurch würde sich zumindest die Unbeobachtbarkeit der übrigen Teilnehmer verringern).

In [Chau_88 §2.5] wurde ein Protokoll vorgeschlagen, das in diesem Sinne kryptographische Robustheit unter Verwendung eines zuverlässigen Verteilnetzes garantieren sollte. In §2.1 wird dieses Protokoll kurz beschrieben und begründet, weshalb es in der ursprünglichen Version unsicher ist. §2.2 beschreibt eine Verbesserung dieses Protokolls (und enthält auch viele Teile, die das ursprüngliche Protokoll ebenfalls benötigen würde.)

---

7 Im folgenden werden wir stets annehmen, daß störende Teilnehmer dies mit Absicht tun, also Angreifer sind. Die Ursache für eine Störung kann aber auch ein Fehler sein. Einerseits sollte deshalb in der Praxis ein als Störer identifizierter Teilnehmer nicht sofort vom Protokoll ausgeschlossen werden, sondern man sollte eine Fehlerbehebung versuchen und ihm zumindest eine zweite Chance geben. Andererseits sollten Fehler möglichst bereits in Ebenen unter dem DC-Protokoll toleriert werden [Pfit_85, Pfit_90].

Generalannahmen: In diesem Kapitel nehmen wir durchweg an, daß Nachrichten *zuverlässig* verteilt werden, d.h. kein Angreifer kann Konsistenz oder Korrektheit der Verteilung manipulieren.

Wie realistisch diese Annahme ist oder gemacht werden kann, wird in §3.1 diskutiert. „P_i veröffentlicht N“ bedeutet, daß P_i die Nachricht N zuverlässig an alle anderen Teilnehmer verteilt.

Für die Unbeobachtbarkeit werden keine weiteren Annahmen gemacht, für die Robustheit wird jedoch ein kryptographisch beschränkter Angreifer angenommen. In gewissem Sinne sind also *zwei verschiedene Angreifer* gleichzeitig zu betrachten!

2.1 Das ursprüngliche Fallenprotokoll nach CHAUM

Könnte man beliebige Runden des DC-Protokolls öffentlich nachrechnen, „aufdecken“, so wäre die Lokalisierung von Störern recht einfach. Dies ist für *sensitive* Runden (etwa Teile echter Nachrichten) mit der Unbeobachtbarkeit natürlich unvereinbar. *Nichtsensitive* Runden könnte man hingegen gefahrlos aufdecken. Man müßte daher zugleich

- nach einer Störung sensitive und nichtsensitive Runden sicher voneinander unterscheiden, und
- verhindern können, daß ein Angreifer gezielt nur sensitive Runden stört.

Die Lösung dieser Probleme beruht darauf, daß gute Teilnehmer gezielt nichtsensitive Nachrichten, **Fallen**, senden, die für alle anderen Teilnehmer von echten, sensitiven Nachrichten nicht zu unterscheiden sind, für die aber nach einer Störung ein eindeutiger Beweis ihrer Nichtsensitivität erbracht werden kann [Chau_88]. Alle folgenden Protokolle zur Gewährleistung der Robustheit basieren auf dieser Idee, weshalb sie kurz **Fallenprotokolle** genannt werden.

Ein erstes Fallenprotokoll wurde in [Chau_88 §2.5] skizziert. Es sollte kryptographische Robustheit gewährleisten. Robustheit und Unbeobachtbarkeit setzten zuverlässige Verteilung voraus.

Ursprüngliches Fallenprotokoll: Damit man eine Basis zum öffentlichen Nachrechnen von DC-Runden hat, werden alle Ausgaben zuverlässig verteilt. Vorausgesetzt wird die Reservierungsleistentechnik mit $F = Z_2$ (vgl. §1.2). Jeder Teilnehmer P_i reserviert in jeder Reservierungsphase genau einen Slot. Vor jeder Reservierungsphase entscheidet er, ob er diesen Slot für eine wirkliche Nachricht oder eine Falle verwenden will. Entscheidet er sich für eine Falle und hat Index x für die Reservierung gewählt, so wählt er eine beliebige Nachricht y und veröffentlicht verschlüsselt die Nachricht (x, y)[8]. Dies wird eine **Fallenankündigung** genannt. Im Slot, der zum Index x gehört, sendet er dann die Nachricht y.

Vor jeder Runde legt sich jeder Teilnehmer auf seine Ausgabe fest, bevor er sie veröffentlicht (genauer vgl. §2.2). Dies verhindert, daß böse Teilnehmer ihre Ausgaben abhängig von den Ausgaben anderer Teilnehmer wählen.

Wenn die von P_i gestellte Falle gestört wird, veröffentlicht P_i einen **Fallenbeweis**, d.h. die Nachricht (x, y) im Klartext, zusammen mit dem verwendeten Schlüssel (und ggf. die bei der Verschlüsselung verwendeten Zufallszahlen [GoMi_84]), und der Angreifer wird in allen Runden der Falle *verfolgt*:

Aufdecken einer Runde: Wird eine Runde r aufgedeckt, so veröffentlicht jeder Teilnehmer P_i sein Nachrichtenzeichen M_i^r und alle Schlüssel K_{ij}^r. Die Summe kann mit seiner öffentlich bekannten Ausgabe O_i^r verglichen werden.

Angenommen, der böse Teilnehmer P_a habe gestört. Gibt P_a sich selbst als böse zu erkennen, z.B. indem seine Summe nicht „paßt“, oder indem er nichts oder sein tatsächlich verwendetes, aber nicht erlaubtes Nachrichtenzeichen M_a^r veröffentlicht, so wird er von der weiteren Teilnahme am DC-Protokoll *ausgeschlossen*. Möchte er sich nicht direkt verraten, so muß er (damit seine Ausgabe O_a^r stimmt) mindestens einen Schlüssel K_{ai}^r falsch angeben. Ist P_i gut, so wird dies wegen $K_{ai}^r \neq K_{ia}^r$ entdeckt und der Schlüssel, der dem Angreifer ohnehin bereits bekannt ist, wird aus dem Schlüsselgraphen *entfernt*. Ist P_i

8 Man kann dies als eine kompakte Form der Nachricht „Ich werde den durch Index x reservierten Slot verwenden, um eine Falle y zu senden“ deuten.

ebenfalls böse, so kann er zwar P_a decken, muß dann aber selbst sein Nachrichtenzeichen oder einen Schlüssel abändern. Letztendlich wird stets entweder ein böser Teilnehmer ausgeschlossen, oder ein Schlüssel eines bösen Teilnehmers entfernt.

Ist der Schlüsselgraph nach einem Angriff partitioniert, so wird das DC-Protokoll in den einzelnen Partitionen jeweils getrennt ausgeführt. (Da Schlüssel zwischen guten Teilnehmern nie entfernt werden, befinden sich alle guten Teilnehmer stets in einer Partition!) Da der Angreifer nur eine beschränkte Anzahl von Schlüsselpaaren und Teilnehmern hat, nimmt spätestens nach dieser Anzahl von verfolgten Angriffen kein Angreifer mehr am DC-Protokoll der guten Teilnehmer teil.

Wegen der Annahme zuverlässiger Verteilung kann jeder gute Teilnehmer lokal über das Entfernen von Schlüsseln und den Ausschluß von Teilnehmern entscheiden, und wird zu demselben Ergebnis kommen wie alle anderen guten Teilnehmer.

Das Aufdeckprotokoll wird unverändert auch in den folgenden verbesserten Fallenprotokollen verwendet. Das ursprüngliche Fallenprotokoll selbst ist jedoch unsicher:

Kamikazeangriff auf das Fallenprotokoll: Selbst ein kryptographisch beschränkter Angreifer kann einen Fallenbeweis für einen beliebigen Slot, den er nicht reserviert hat, fälschen:

Dazu veröffentlicht er die erforderliche verschlüsselte Nachricht (x, y), aber *ohne* den zu x gehörenden Slot s_x zu reservieren. Da eine Reservierungsleiste der Länge n^2 verwendet wird, wird mit Wahrscheinlichkeit $\approx 1/n$ ein guter Teilnehmer den Index x tatsächlich verwenden. In diesem Fall sendet der Angreifer in Slot s_x die angekündigte Nachricht y. (Man beachte, daß der Angreifer die Nachricht y^* des rechtmäßigen Benutzers leicht aus den globalen Summen S des Slots und y berechnen kann: $y^* = S - y$.) Hernach veröffentlicht er seinen Fallenbeweis für Slot s_x, und der rechtmäßige Benutzer wird identifiziert und als vermeintlicher Angreifer ausgeschlossen! Veröffentlicht auch der rechtmäßige Benutzer einen Fallenbeweis, so ist zwar klar, daß einer der beiden Veröffentlicher ein Angreifer war, nicht aber, wer.

Der Name *Kamikazeangriff* rührt daher, daß durch einige naheliegende Ergänzungen des Aufdeckprotokolls nach jedem Angriff wenigstens ein böser Teilnehmer ausgeschlossen oder ein sinnloser Schlüssel entfernt wird. Der Angreifer kann dann zwar immer noch gute Teilnehmer identifizieren, schließt sich damit aber nach und nach selbst aus [WaPf_89].

DAVID CHAUM schlug [in einer Diskussion am 12.4.1989 über den Inhalt von §2] vor, den Kamikazecharakter des Angriffs noch dadurch zu erhöhen, daß *jede* Fallenankündigung geöffnet werden muß. Hatte ein Angreifer eine Fallenankündigung für einen von niemandem genutzten Slot veröffentlicht, so wird er entdeckt und ausgeschlossen.

Andere Reservierungstechnik: Ein analoger Kamikazeangriff ist auch dann möglich, wenn statt einer Reservierungsleistentechnik überlagerndes Empfangen [BoBo_90, Pfit_90] (vgl. §1.2) verwendet wird. Der Index x muß nur überall durch eine Reservierungsnachricht x ersetzt werden.

Der Angriff ist hier allerdings weitaus weniger gefährlich, da die Wahrscheinlichkeit, daß ein guter Teilnehmer gerade die Reservierungsnachricht x des Angreifers wählen wird, exponentiell klein ist in der Länge der Reservierungsphase.

Direkte Verbesserung des Protokolls: Wird überlagerndes Empfangen verwendet und werden zusätzlich einige der Teile und Verfeinerungen des Fallenprotokolls aus §2.2 übernommen, so entsteht ein Protokoll, das etwa so gut und effizient ist wie das in §2.2, außer daß immer eine sehr kleine Wahrscheinlichkeit besteht, daß der Sender einer Nachricht identifiziert wird.

Übernommen werden müßte im wesentlichen: 1. Jeder Teilnehmer muß eine feste Anzahl > 1 von Slots reservieren und davon einen festen Anteil für Fallen nutzen (vgl. 2. in §2.2.2), da ansonsten nach jeder Sendephase, wenn Ankündigungen gestörter Fallen geöffnet wurden, klar ist, daß die sensitiven Nachrichten dieser Phase nicht von den Fallenstellern gesendet wurden. 2. Die Fallenankündigungen dürfen (im Gegensatz zu dem Vorschlag in [Chau_88]) nur eindeutig geöffnet werden können (vgl. 3. in §2.2.2). Andernfalls könnte ein kryptographisch unbeschränkter Angreifer seine Fallenankündigungen so

öffnen, daß sie die Falle für einen Slot ankündigen, den ein guter Teilnehmer reserviert hat, denn zur Zeit des Öffnens hat er die Reservierungen gesehen.

Das verbesserte Fallenprotokoll in §2.2 vermeidet selbst die kleine Erfolgswahrscheinlichkeit des Angreifers und kann mit beiden genannten Reservierungstechniken kombiniert werden. Daher verzichten wir auf eine ausführlichere Darstellung dieser Möglichkeit.

2.2 Verbessertes und verfeinertes Fallenprotokoll

Das folgende Fallenprotokoll verhindert den obigen Kamikazeangriff, indem Fallenankündigungen und Fallen besser miteinander verkettet werden. Es enthält auch einige Details, die für das ursprüngliche Fallenprotokoll ebenfalls notwendig gewesen wären. Wir beginnen mit einem kurzen Überblick.

2.2.1 Überblick

Man nehme an, eine der in §1.2 genannten Reservierungstechniken werde verwendet, und es wurden R Slots reserviert. *Jeder* Teilnehmer P_i, der einen Slot s_i reserviert hat, sendet in Slot s_i eine verschlüsselte **Ankündigung**, und in Slot s_i+R eine **Falle** oder eine **Nichtfalle**, d.h. eine wirkliche Nachricht, gemäß seiner Ankündigung (siehe Bild 1).

Fallenankündigungen und Fallen sind daher durch ihre Slotnummern eindeutig verkettet, d.h. der Angreifer kann nicht mehr a priori einen Fallenbeweis für einen Slot fälschen, der von einem anderen Teilnehmer verwendet wird. Wenn Ankündigungen eindeutig sind, d.h. wenn es keine Nachricht gibt, die sowohl eine Falle als auch eine Nichtfalle ankündigen kann, kann der Angreifer Nichtfallenankündigungen nicht zum Initiieren des Aufdeckprotokolls mißbrauchen.

Daher stellt sich das Problem des ursprüngliche Fallenprotokolls hier nicht.

Wir nennen diese drei Phasen des Protokolls **Reservierungsphase**, **Ankündigungsphase** und **Sendephase**.

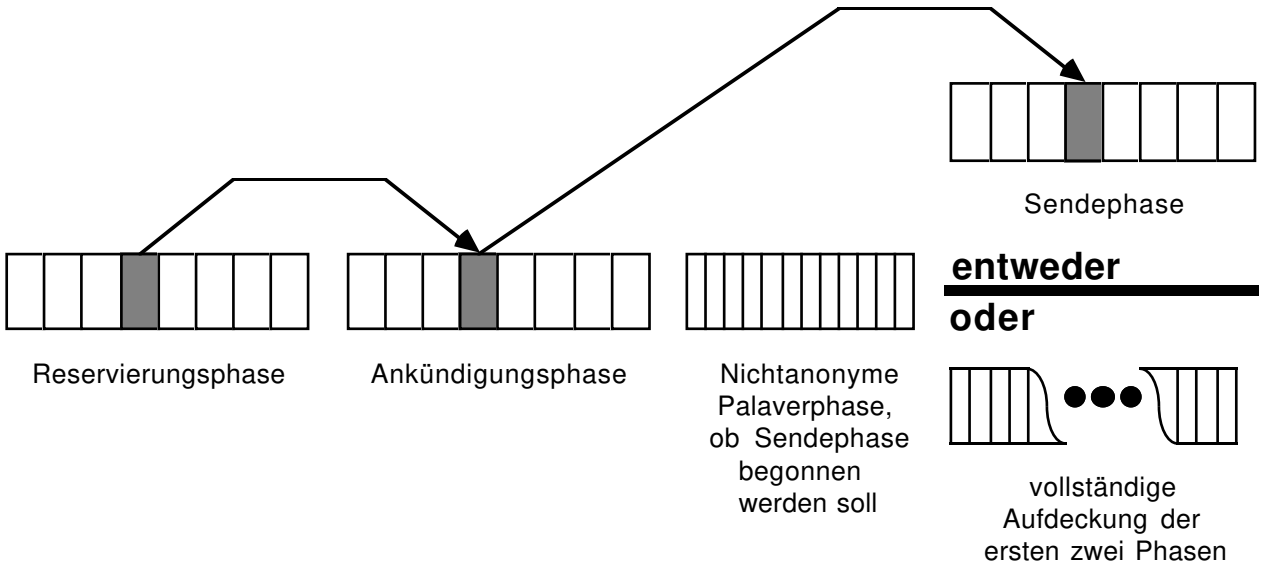

Bild 1 Phasen des verbesserten Protokolls.

Das Reservierungs- und Ankündigungsverhalten der Teilnehmer wird unabhängig von ihren wirklichen Sendewünschen gestaltet (wie in [Chau_88]), so daß die ersten zwei Phasen *vor* Beginn der Sendephase ohne Verlust an Unbeobachtbarkeit aufgedeckt werden können. Vor der Sendephase wird daher eine nichtanonyme **Palaverphase** eingeschoben, in der jeder Teilnehmer eine Störung der ersten beiden Phasen anzeigen und damit den Beginn der Sendephase verhindern kann. Störungen während der Sendephase werden nur aufgedeckt, wenn der passende Fallenbeweis gezeigt wird.

2.2.2 Das Protokoll

0. Allgemeine Ausschlußregel: Ein Teilnehmer, der eine erwartete Nachricht nicht sendet, wird von der weiteren Teilnahme am Protokoll ausgeschlossen.

1. Ausgabenfestlegung: Alle Ausgaben werden wieder *zuverlässig* verteilt. Für manche Runden ist aber zusätzlich wünschenswert, daß die Ausgaben aller Teilnehmer *gleichzeitig* erfolgen: Angenommen, ein böser Teilnehmer erhält die Ausgaben aller anderen Teilnehmer, schon *bevor* er seine eigene Ausgabe veröffentlicht. Er kann dann leicht die korrekte globale Summe berechnen und z.B. die Sendephase durch Zerstören aller offensichtlich sensitiven Nachrichten stören [Chau_88]. (Offensichtlich sensitiv sind z.B. alle Nachrichten, die an böse Teilnehmer adressiert sind. Ob eine Nachricht an einen bösen Teilnehmer adressiert ist, kann bei vielen Adressierungsschemata bereits nach den ersten Runden des sie enthaltenden Slots festgestellt werden.)

Folglich muß sich jeder Teilnehmer auf seine lokale Ausgabe O_i^r für mehrere Runden r festlegen, bevor er die Ausgaben anderer Teilnehmer für irgendwelche dieser Runden erfährt. Diese Festlegung geschieht mittels eines kryptographisch sicheren **Festlegschemas** (*commitment scheme* [BrCC_88], vgl. auch 3.):

Vor Beginn der Kommunikation wählt sich jeder Teilnehmer ein solches Festlegschema. Zur Ausgabenfestlegung veröffentlichen in einer ersten Phase nacheinander alle Teilnehmer ihre Festlegung. In einer zweiten Phase werden die Festlegungen dann in umgekehrter Reihenfolge[9] geöffnet und somit die Ausgaben O_i^r veröffentlicht.

Ein Teilnehmer, dessen Ausgabe nicht seiner Festlegung entspricht, wird als Angreifer ausgeschlossen. Wegen der Annahme zuverlässiger Verteilung kann dies guten Teilnehmern nicht passieren.

2. Reservierungsphase: Um zu ermöglichen, die ersten beiden Phasen vor Beginn der Sendephase ohne Unbeobachtbarkeitsverlust aufzudecken, darf kein Teilnehmer während dieser Phasen sensitive Information verwenden. Das Reservierungsverhalten muß daher unabhängig von den wirklichen Sendewünschen der Teilnehmer sein: Jeder Teilnehmer muß in jeder Reservierungsphase eine feste Anzahl u von Slots reservieren, von denen ein fester Anteil μ für Fallen verwendet wird. (Es sind auch etwas flexiblere Regeln denkbar [WaPf_89].)

Bei allen bekannten Reservierungstechniken (vgl. §1.2) versuchen mit einer gewissen Wahrscheinlichkeit zwei oder mehr Teilnehmer, denselben Slot zu reservieren, d.h. es können *Reservierungskollisionen* auftreten. Am sinnvollsten erscheint die Verwendung von überlagerndem Empfangen nach [BoBo_90], da hier die Wahrscheinlichkeit für Reservierungskollisionen beliebig klein gemacht werden kann.

Um zu verhindern, daß der Angreifer gezielt Reservierungskollisionen mit guten Teilnehmern verursacht, müssen sich alle Teilnehmer vor der Reservierungsphase zunächst auf ihre *gesamten* Ausgaben in der Reservierungsphase festlegen (vgl. 1.)[10]. Verwenden wider Erwarten doch zwei Teilnehmer dieselbe Reservierungsnachricht, so wird dies ignoriert, d.h. mehrfach verwendete Reservierungsnachrichten führen zu keiner Reservierung. Verwenden böse Teilnehmer dieselben Reservierungsnachrichten, so schaden sie sich damit nur selbst.

Die Reservierungsleistentechnik mit $F = Z_2$ ermöglicht spezielle Angriffe (ähnlich §2.1). Daher sollte diese Technik nicht verwendet werden, wenn man verlangt, daß gute Teilnehmer *niemals* als Angreifer betrachtet und ausgeschlossen werden [WaPf_89].

---

9 Die umgekehrte Reihenfolge garantiert, daß ein Teilnehmer seine eigene Festlegung unabhängig von den Ausgaben der anderen Teilnehmer getroffen haben muß. Werden Festlegungen in derselben Reihenfolge getroffen und geöffnet, so wäre es denkbar, daß ein böser Teilnehmer seine Festlegung aus derjenigen eines anderen Teilnehmers so berechnen kann, daß z.B. die Summe der beiden damit festgelegten Ausgaben gerade 0 ergibt und er seine berechnete Festlegung, *nachdem* sein „Vorbild“ geöffnet wurde, ebenfalls öffnen kann.

10 Aus diesem Grunde ist das in [Pfit_90] beschriebene überlagernde Empfangen für unsere Zwecke weniger geeignet: dort ergeben sich die R Reservierungsrunden durch ein *interaktives* Protokoll, so daß es unmöglich ist, die Ausgaben der r-ten Reservierungsrunde festzulegen, bevor die globale Summe der $(r-1)$-ten bekannt ist.

Reservierungsphasen mit *unmöglichem Ergebnis* (d.h. einem, das unter guten Teilnehmern nicht auftreten könnte) können aufgedeckt werden. Hierfür prüft jeder Teilnehmer bei Verwendung

- der *Reservierungsleistentechnik* mit einer großen Gruppe [Pfit_85]: Wurden insgesamt genau $u \bullet n$ Reservierungsversuche vorgenommen?
- des *überlagernden Empfangens*: Hat der Teilnehmer seine eigenen Reservierungsnachrichten korrekt wieder empfangen?

Eine gefundene Störung wird in der Palaverphase angezeigt.

Zeigt einer dieser Tests einen Fehler an, so wird beim Aufdecken mit Sicherheit (unbedingt) eine Protokollabweichung gefunden. Wird keine Protokollabweichung gefunden, so war jeder Teilnehmer, der eine Störung anzeigte, böse und kann ausgeschlossen werden. Dies verhindert, daß der Angreifer es nie zu einer Sendephase kommen läßt.

3. Ankündigungsphase: Ankündigungen werden wiederum durch ein Festlegschema implementiert [BrCC_88], d.h. der Teilnehmer muß sich öffentlich auf einen der Werte „Falle“ oder „Nichtfalle“ festlegen (codiert durch $\{0,1\}$). Im Gegensatz zur Ausgabenfestlegung hat hier die Sicherheit des für die Ankündigungen verwendeten Festlegschemas Auswirkungen nicht nur auf die Robustheit, sondern auch auf die Unbeobachtbarkeit: Wenn ein Angreifer eine Nichtfallenankündigung eines anderen Teilnehmers als Fallenankündigung öffnen könnte, könnte er erreichen, daß der Sender einer beliebigen sensitiven Nachricht identifiziert wird.

Daher muß ein Festlegschema verwendet werden, bei dem jede Ankündigung selbst durch einen kryptographisch unbeschränkten Angreifer nur auf eine Weise geöffnet werden kann (dies heißt in [BrCC_88] „sicher für den Verifizierer“). Die Ununterscheidbarkeit von Fallen- und Nichtfallenankündigungen ist damit zwangsläufig nur kryptographisch sicher.

Wer welches Festlegschema verwendet, darf den ankündigenden Teilnehmer natürlich nicht verraten (im Gegensatz zur Ausgabenfestlegung, wo der Urheber einer Ausgabe ja bekannt sein *soll*). Um dies zu erreichen, gibt es verschiedene Möglichkeiten:

i. Für jede Ankündigung muß der Ankündiger ein neues Festlegschema wählen. Üblicherweise heißt dies aber, daß vor jeder Ankündigung ein recht aufwendiger Erzeugungsalgorithmus (der z.B. zufällig zwei Primzahlen einer vorgegebenen Länge bestimmt und miteinander multipliziert) ausgeführt werden muß.

ii. Für alle Ankündigungen aller Teilnehmer wird ein einziges Festlegschema gewählt. Die Geheimhaltung der Festlegung ist jedoch meist nur sicher gegenüber Teilnehmern, die die Erzeugung des Schemas nicht beobachten konnten (z.B. der im obigen Beispiel die gefundenen Primzahlen vor ihrer Multiplikation nicht sah). Damit scheiden die meisten Schemata aus (möchte man nicht auf sehr aufwendige Verfahren zur *gemeinsamen* Erzeugung zurückgreifen [ChDG_88], analog wie in [PfWa_90]). Übrig bleiben etwa Schemata, deren Geheimhaltungsfähigkeit auf dem Problem des diskreten Logarithmus beruht [BrCC_88 §6.2.2].

iii. Jeder Teilnehmer wählt vor Beginn der Kommunikation jeweils ein Festlegschema. Für alle Ankündigungen werden alle n Festlegschemata verwendet: Um sich auf $b \in \{0,1\}$ festzulegen, wird b zufällig zerlegt in $b = b_1 + \ldots + b_n$ ($b_i \in \{0,1\}$) und sodann mit dem k-ten Festlegschema eine Festlegung auf b_k vorgenommen. Dies ver-n-facht allerdings den Rechen- und Nachrichtenaufwand je Ankündigung.

Für die i. und iii. Möglichkeit kann ein beliebiges Festlegschema der geforderten Art verwendet werden, etwa eines, dessen Sicherheit auf der Faktorisierungsannahme beruht. Ein solches gewinnt man etwa aus dem (asymmetrischen) probabilistischen Kryptosystem aus [BlGo_84]:

Die *Auswahl* eines Festlegschemas besteht in der Auswahl eines Chiffrierschlüssels c, der öffentlich bekannt gegeben wird. Die *Festlegung* auf x geschieht, indem x verschlüsselt und die Verschlüsselung $c(r,x)$ zusammen mit dem Schlüssel c (falls dieser nicht schon bekannt ist) bekannt gegeben wird, wobei r die bei der Verschlüsselung verwendeten Zufallszahlen bezeichne. Die Festlegung wird *geöffnet*, indem (r, x) bekannt gegeben wird [GMW_87]. Da die Entschlüsselung eindeutig ist, kann eine Festlegung $c(r,x)$ wie gefordert nur auf eine Art geöffnet werden.

Dasselbe Festlegschema kann dann auch für die Ausgabenfestlegung verwendet werden.

Die Ankündigungsphase muß nicht durch Ausgabenfestlegung geschützt werden. Jeder gute Teilnehmer P_i, dessen Ankündigung gestört wurde, zeigt dies in der Palaverphase an. Da P_i den entsprechenden Slot reserviert hat, kann er sein Recht, den Slot zu verwenden, während des Aufdeckens der Reservierungsphase beweisen.

4. Palaverphase: Jeder Teilnehmer, der eine Störung der ersten beiden Phasen feststellte, zeigt dies nicht-anonym allen anderen an. (Es genügt, daß in einer festen Reihenfolge jeder Teilnehmer jeweils ein Bit veröffentlicht. Eine 0 stehe für „keine Störung", eine 1 für „Störung".)

5. Aufdecken der Reservierungs- und Ankündigungsphase: Zeigte in der Palaverphase wenigstens ein Teilnehmer eine Störung an, so werden alle Runden der ersten beiden Phasen aufgedeckt (vgl. §2.1) und jeder Teilnehmer prüft lokal das Verhalten jedes anderen Teilnehmers nach: Zuerst werden die Regeln des überlagernden Sendens, dann die Regeln des Reservierungs- und Ankündigungsprotokolls geprüft.

Nach jeder Störung wird entweder ein Schlüssel aus dem Schlüsselgraphen entfernt oder ein böser Teilnehmer ausgeschlossen. Wird keine Protokollabweichung festgestellt, so sind alle Teilnehmer, die eine Störung anzeigten, böse und werden ausgeschlossen. Wegen des zuverlässigen Verteilnetzes, und weil das Verfolgen des Angreifers deterministisch erfolgt, gelangen alle guten Teilnehmer zum selben Ergebnis.

6. Sendephase: Zeigte in der Palaverphase *kein* Teilnehmer eine Störung an, so werden nun entsprechend der Reservierungsphase die Nachrichtenslots verwendet. Wird ein Slot für eine Falle genutzt, so wählen alle, auch der rechtmäßige Benutzer, für alle Runden der Falle als Nachrichtenzeichen die 0. Um zu verhindern, daß der Angreifer anhand eines Teils eines Slots Nachrichten von Fallen unterscheiden kann, werden alle Slots durch Ausgabenfestlegung geschützt. Die Festlegung erfolgt stets für einen ganzen Slot im voraus.

7. Aufdecken von Fallen: In einer weiteren Palaverphase veröffentlicht jeder Teilnehmer die Zahl ($\in \{0, \ldots, \mu \bullet u\}$) seiner gestörten Fallen und reserviert damit ggf. den zum Aufdecken seiner gestörten Fallen benötigten Bandbreitenanteil. Nacheinander öffnen die Fallensteller die entsprechenden Fallenankündigungen und überzeugen damit alle guten Teilnehmer zweifelsfrei davon, daß alle Runden der entsprechenden Slots aufgedeckt werden können.

Da alle Teilnehmer in einer Falle stets eine 0 senden mußten, bereitet das Aufdecken einer Falle keine Probleme.[11] Wird kein Fehlverhalten festgestellt, so wird der Teilnehmer ausgeschlossen, der die Fallenankündigung öffnete.

Fazit: Wenn ein störender Angreifer nicht die Ausgabenfestlegung „entschlüsseln" und Fallen- und Nichtfallenankündigungen unterscheiden kann, so wird er nach einer Störung mit der Wahrscheinlichkeit μ bestraft. Ein geeignetes Festlegschema existiert z.B. unter der Faktorisierungsannahme.[12]

Im Gegensatz zum ursprünglichen Protokoll in §2.1 wird die Unbeobachtbarkeit deterministisch garantiert, d.h. ein Angreifer hat keinerlei Chance, die Unbeobachtbarkeit zu gefährden.

---

11 Sendet der rechtmäßige Benutzer in seiner Falle statt 0 eine beliebige Nachricht, so muß man zwischen ihm und dem Angreifer unterscheiden (beide haben in der Falle etwas gesendet, und es ist keineswegs sicher, daß derjenige Teilnehmer, der eine Fallenankündigung öffnet, der Fallensteller ist). Wurde eine Reservierungsleistentechnik verwendet, so kann man hierzu einfach die Reservierung der Falle aufdecken. Wird überlagerndes Empfangen verwendet, so muß statt dessen der entsprechende Teil der Ankündigungphase aufgedeckt werden.

12 Unser Verfahren kann so modifiziert werden, daß es sowohl Unbeobachtbarkeit als auch Robustheit entweder deterministisch unter der Annahme $t < n/3$ [WaPf_89], oder mit exponentiell kleiner Fehlerwahrscheinlichkeit unter der Annahme $t < n/2$ garantiert. Dazu muß man die Ausgabenfestlegung durch *simultane Verteilung* [CGMA_85] ersetzen und das kryptographisch sichere Festlegschema durch ein nichtkryptographisch sicheres. Beides kann für $t < n/3$ deterministisch mit der Technik aus [BeGW_88] und für $t < n/2$ probabilistisch mit der Technik aus [RaBe_89] erreicht werden. Da ein unerwartet zahlreicher Angreifer Nichtfallenankündigungen als Fallenankündigungen öffnen kann, garantieren diese Modifikationen keine unbedingte Unbeobachtbarkeit.

3 Beseitigen der Annahme zuverlässiger Verteilung

Wird das in §2.2 beschriebene Protokoll in einem *unzuverlässigen* Verteilnetz angewendet, z.B. in einem Sternnetz mit nicht-vertrauenswürdiger Zentrale, so ist die Unbeobachtbarkeit verloren:

Kann der Angreifer gezielt eine Nachricht nur an einen beliebigen Teil der guten Teilnehmer verteilen, so kann er am Auftreten oder Ausbleiben einer Reaktion erkennen, in welchem Teil sich der Empfänger dieser Nachricht befindet. Durch höchstens $\lceil \mathrm{ld}(n) \rceil$-maliges Halbieren der Menge der guten Teilnehmer kann der Empfänger vollständig identifiziert werden [Waid_90, WaPf_89].

Kann ein Angreifer die Ausgabe eines guten Teilnehmers fälschen, so kann er das Aufdeckprotokoll (§2.1, bzw. 5. oder 7. in §2.2.2) verwenden, um diesen auszuschließen oder zumindest einen Schlüssel, den dieser mit einem anderen guten Teilnehmer gemeinsam hat, zu entfernen (das hatten die jungen Kryptographen vergessen). Selbst ein Gericht, das als eine sehr langsame Implementierung eines zuverlässigen Verteilnetzes betrachtet werden kann, kann im nachhinein ursprüngliche und gefälschte Nachrichten nicht voneinander unterscheiden.

Im folgenden werden einige Verfahren diskutiert, um Unbeobachtbarkeit und Robustheit ohne die Annahme zuverlässiger Verteilung miteinander zu kombinieren.[13]

In §3.1 diskutieren wir, wie zuverlässige Verteilung direkt physisch oder durch *Byzantinische Übereinstimmungsprotokolle* implementiert werden kann. Die Unbeobachtbarkeit bleibt jeweils „bedingt".

§3.2.1 beschreibt robuste fail-stop Byzantinische Übereinstimmung, eine neue Erweiterung der Byzantinischen Übereinstimmung, mittels derer in §3.2.2 dann Unbeobachtbarkeit unter der **Zusammenhangsannahme** garantiert werden kann, und Robustheit, wenn der Angreifer zusätzlich kryptographisch beschränkt ist.

Die Zusammenhangsannahme besagt, daß jede Nachricht, die ein guter Teilnehmer an einen anderen sendet, nach beschränkter Zeit tatsächlich ankommt. Sie ist eine *notwendige* Voraussetzung für die Unbeobachtbarkeit, falls gleichzeitig Robustheit garantiert werden soll: Wenn ein böser Teilnehmer einfach nichts mehr sendet, müssen für die Robustheit die guten Teilnehmer weiter kommunizieren. Würde nun ein guter Teilnehmer von allen anderen abgeschnitten, so würde auch ohne ihn weiter kommuniziert. Damit würde die Unbeobachtbarkeit aller Teilnehmer eingeschränkt, da er nicht mehr als Sender von Nachrichten in Frage käme.

3.1 Bekannte Techniken zur Implementierung zuverlässiger Verteilung

Physisch zuverlässige Verteilung: Am angenehmsten zur Implementierung des DC-Protokolls wäre natürlich ein physisches Verteilnetz, das ohne weiteres Zutun garantiert, daß kein Angreifer die Konsistenz oder Korrektheit der Verteilung stören kann. Physisch zuverlässige Verteilnetze sind aber nur sehr schwer zu realisieren:

Da im DC-Protokoll permanent zuverlässig verteilt werden muß, benötigt man eine einigermaßen effiziente Implementierung. Prinzipiell scheiden Netze, in denen Nachrichten *aktiv* durch andere Stationen weitergeleitet werden (und dabei verfälscht werden können!), aus. Dies gilt z.B. für ring-, baum- und sternförmige Netze. Die Annahme, jeder Teilnehmer verfüge über einen eigenen Satelliten, ist sicher unrealistisch, so daß höchstens Funknetze und Busse übrigbleiben.

Störungen in einem physisch zuverlässigen Verteilnetz müssen mit physischen Mitteln lokalisiert werden, und es muß gesichert werden, daß diese ihrerseits nicht mißbraucht werden können, um die Verteileigenschaft zu stören.

---

13 In [GMW_87] wurden sehr allgemeine Techniken für „multi-party computations" eingeführt, die natürlich auch zur Implementierung von Sender- und Empfängerunbeobachtbarkeit verwendet werden können [WaPf_89]. Alle bekannten Techniken dieser Art setzen aber ebenfalls ein zuverlässiges Verteilnetz voraus.

Byzantinische Übereinstimmung: In physisch weniger zuverlässigen Netzen wird zuverlässige Verteilung durch **Byzantinische Übereinstimmungsprotokolle** (BÜPs) [PeSL_80] implementiert.

Unter der Zusammenhangsannahme[14] existieren BÜPs, die

- kryptographisch beschränkte Angreifer tolerieren und daher **kryptographische BÜPs** genannt werden [PeSL_80, DoSt_83], und solche, die
- kryptographisch unbeschränkte, dafür aber anzahlmäßig beschränkte Angreifer tolerieren und daher **nichtkryptographische BÜPs** genannt werden [PeSL_80, DFFL_82, FeMi_88, GoPe_90, BaPW_91].

Welche der zwei möglichen Beschränkungen angenommen wird, muß gewöhnlich a priori entschieden werden. (Im Anhang stellen wir ein neues Protokoll vor, wo dies nicht nötig ist.)

Sei t die Anzahl der bösen Teilnehmer. Alle bekannten kryptographischen BÜPs beruhen auf digitalen Signaturen [DiHe_76]. Sie tolerieren jede Anzahl $t < n$. Durch nichtkryptographische BÜPs kann deterministisch höchstens $t < n/3$ toleriert werden [PeSL_80]. Wenn eine kleine Fehlerwahrscheinlichkeit akzeptabel ist und zuverlässige Verteilung und geheime Kanäle in einer Vorwegphase zur Verfügung stehen, ist sogar $t < n/2$ tolerierbar [BaPW_90]. Für einen Überblick über untere Schranken für BÜ und bekannte Lösungen siehe [Fisc_83, Reis_87, ChDw_89].

3.2 Unbeobachtbarkeit mit kryptographischer Robustheit unter der Zusammenhangsannahme

Das folgende Protokoll garantiert Unbeobachtbarkeit ausschließlich unter der Zusammenhangsannahme (insbesondere ohne Bedingungen an die Anzahl der Angreifer) und kryptographische Robustheit. Wie schon in §2 werden also für Unbeobachtbarkeit und Robustheit zwei unterschiedliche Annahmen gemacht.

Die Grundidee ist, das Protokoll aus §2.2 zu verwenden, die zuverlässige Verteilung aber durch eine spezielle Art von BÜP zu implementieren: In §3.2.1 zeigen wir, wie ein kryptographisches BÜP für $t \leq n-2$ in ein BÜP verwandelt werden kann, das

- BÜ garantiert, falls der Angreifer wie angenommen kryptographisch beschränkt ist, und
- in jedem Fall garantiert, daß alle guten Teilnehmer ihre Teilnahme beenden, sobald die BÜ doch gestört wird.

Wir nennen dies **robuste fail-stop BÜ**.

Treffen die für das Fallenprotokoll (§2.2) und das robuste fail-stop BÜP gemachten kryptographischen Annahmen zu, so sind Robustheit und Unbeobachtbarkeit garantiert. (Im folgenden genügt als einzige kryptographische Annahme die Faktorisierungsannahme.) Trifft jedoch eine verwendete kryptographische Annahme nicht zu, so daß die Robustheit ohnehin nicht mehr garantiert werden kann, so bricht die Kommunikation vollends ab (fail-stop Verhalten), so daß wenigstens die Unbeobachtbarkeit erhalten bleibt (§3.2.2).

3.2.1 Robuste fail-stop Byzantinische Übereinstimmung

Wir konstruieren ein Protokoll für robuste fail-stop BÜ, indem wir drei Änderungen an einem kryptographischen BÜP vornehmen. Der Konkretheit halber verwenden wir das aus [DoSt_83 Theorem 3], im folgenden **DOLEV-STRONG BÜP** genannt; zum Verständnis des folgenden genügt es aber zu wissen, daß es $t \leq n-1$ toleriert, und daß digitale Signaturen darin der einzige auf einer kryptographischen Annahme beruhende Baustein sind.

1. Änderung: Statt konventioneller digitaler Signaturen werden sogenannte **Fail-stop-Signaturen** verwendet, d.h. digitale Signaturen [DiHe_76, GoMR_88] mit der zusätzlichen Eigenschaft, daß Fälschungen bewiesen werden können [Pfit_89, WaPf_89, PfWa_90, PfWa_91]:

---

14 Die Zusammenhangsannahme ist auch eine notwendige Voraussetzung für Byzantinische Übereinstimmung.

Wie üblich sind unter einer kryptographischen Annahme keine Fälschungen möglich. War die kryptographische Annahme falsch, und empfängt P_i eine angeblich von ihm signierte Nachricht, so kann er mit Wahrscheinlichkeit exponentiell nahe 1 effizient einen **Fälschungsbeweis** berechnen. Die Möglichkeit des Berechnens von Fälschungsbeweisen hängt nicht von kryptographischen Annahmen ab. (Eine Wahrscheinlichkeit heißt exponentiell nahe 1, wenn sie für einen Parameter σ kleiner als $2^{-\sigma}$ ist, während der Aufwand der Algorithmen nur polynomial in σ wächst.)

Ein konkretes Fail-stop-Signatursystem, Verstecksystem genannt, wird in [Pfit_89, PfWa_90, PfWa_91] beschrieben. Es kann so implementiert werden, daß unter der Faktorisierungsannahme keine Signaturen gefälscht werden können. ([PfWa_91] ist in diesem Tagungsband enthalten.)

Die Signatur von Teilnehmer P_j unter den Wert v sei mit $s_j(v)$ bezeichnet.[15]

2. Änderung: Um das DOLEV-STRONG BÜP zu stören, muß der Angreifer wenigstens eine Signatur eines guten Teilnehmers fälschen. Damit dieser einen Fälschungsbeweis bilden und an alle guten Teilnehmer versenden kann, wird das DOLEV-STRONG BÜP (das $n-1$ Phasen hat) um zwei Phasen ergänzt:

1. Ergänzung	
[n]	Jeder Teilnehmer P_i sendet alle Signaturen, die er während der vorigen Phasen empfangen hat, zurück an ihre angeblichen Unterzeichner, d.h. er sendet jedes $s_j(v)$ zurück an P_j.
[n+1]	Wenn P_j eine gefälschte Signatur $s_j(v)$ empfangen hat, versucht er, einen Fälschungsbeweis zu berechnen. Wenn dies gelingt, sendet er den Fälschungsbeweis an alle anderen Teilnehmer.[16]
[*End*]	Wenn P_i keinen Fälschungsbeweis gefunden oder empfangen hat, entscheidet er wie im ursprünglichen DOLEV-STRONG BÜP. Andernfalls entscheidet er „Signaturen gebrochen“.

Solange keine Fälschungsbeweise auftreten (also insbesondere, wenn die für die fail-stop Signaturen verwendete kryptographische Annahme zutrifft), entscheidet sich kein guter Teilnehmer für „Signaturen gebrochen“. Die Ergänzung ändert dann nichts am bisherigen DOLEV-STRONG BÜP.

Hat der Angreifer in den ersten $n-1$ Phasen, d.h. während des DOLEV-STRONG BÜPs, Signaturen guter Teilnehmer gefälscht, so erhalten alle guten Teilnehmer in [n+1] einen Fälschungsbeweis und entscheiden sich daher übereinstimmend für „Signaturen gebrochen“.

Problematisch ist nur, wenn in Phase [n+1] manche, aber nicht alle guten Teilnehmer einen Fälschungsbeweis erhalten. Dann ist zwar sicher, daß das DOLEV-STRONG BÜP ungestört war und das richtige Ergebnis lieferte (d.h. kein guter Teilnehmer akzeptiert einen falschen Wert), aber die Konsistenz ist nicht gesichert.

3. Änderung: Um nach einer Unkorrektheit die Verteilung zu stoppen, werden Fälschungsbeweise vom Ende einer BÜ zur nächsten weitergegeben.

Dazu sei angenommen, das Protokoll solle „ad infinitum“ ausgeführt werden, d.h. nach jeder Phase [n+1] beginne eine neue Phase [1]. Jede Protokolldurchführung heiße eine **Verteilung**. Für jede Verteilung

---

15 Das DOLEV-STRONG BÜP verwendet iterierte Signaturen (etwa $s_{i_k}(s_{i_{k-1}}(\ldots s_{i_2}(s_{i_1}(v))\ldots))$). Im Verstecksystem sind Signaturen viel länger als Nachrichten. Der Rechenaufwand beim Signieren und, wenn die Nachrichten so kurz sind, daß sie nicht gehasht zu werden brauchen, auch die Länge der Signaturen, wächst mit der Länge der Nachricht. Es ist somit effizienter, statt dessen *Mengen* von Signaturen zu verwenden (etwa $\{s_{i_k}(v), s_{i_{k-1}}(v), \ldots, s_{i_2}(v), s_{i_1}(v)\}$). Zum Glück funktioniert das DOLEV-STRONG BÜP auch mit Mengen von Signaturen [WaPf_89].

16 Man kann Phase [n+1] auf Kosten von mehr Nachrichten einsparen: In Phase [n] sendet jeder Teilnehmer P_i zusätzlich seinen geheimen Schlüssel an alle anderen Teilnehmer. Dann kann jeder Teilnehmer lokal einen Fälschungsbeweis erzeugen, den er andernfalls in Phase [n+1] empfangen würde. Diese Veröffentlichung stört das DOLEV-STRONG BÜP nicht, weil das schon beendet ist. (Beim Verstecksystem genügt der Teil des Schlüssels, der die Signaturen enthält, die P_i während dieser Verteilung verwendet haben könnte. Deshalb brauchen nicht vor jeder Verteilung neue Schlüssel ausgetauscht zu werden.) In diesem Fall kann der signierte Wert des Senders später Dritten nicht als Beweis gezeigt werden, aber dies ist in unserer Anwendung nicht nötig.

sei vorweg bestimmt, welcher Teilnehmer der Sender ist. Dann fügen wir folgende Regel zu allen Phasen des Protokolls hinzu:

2. Ergänzung: Für jede Phase [k] wird folgende Regel zum obigen Protokoll hinzugefügt:

[k] Wenn P_i in der vorigen Phase erstmals einen Fälschungsbeweis empfangen hat, entscheidet er sofort „Signaturen gebrochen“ für die aktuelle und *alle* folgenden Verteilungen, sendet den Fälschungsbeweis an alle anderen Teilnehmer, und beendet dann das Senden „für immer“.

Entscheidet sich ein guter Teilnehmer in Phase [k] für „Signaturen gebrochen“, so entscheidet sich jeder andere gute Teilnehmer in Phase [k+1] ebenso. Damit haben wir robuste fail-stop BÜ erreicht:

Satz 1: Robuste fail-stop Byzantinische Übereinstimmung. Angenommen, das DOLEV-STRONG BÜP, implementiert mit fail-stop Signaturen und der 1. und 2. Ergänzung, werde polynomial oft in den Sicherheitsparametern des fail-stop-Signatursystems wiederholt.

a) Robustheit: Wenn die für die fail-stop Signaturen verwendete kryptographische Annahme wahr ist, realisiert das Protokoll zuverlässige Verteilung.
b) Fail-stop: Wenn die kryptographische Annahme falsch ist, und die bösen Teilnehmer die t-te Verteilung während der Phasen [1] bis [n] stören, so entscheiden sich alle guten Teilnehmer am Ende der t-ten Verteilung für „Signaturen gebrochen“ und beenden das Senden.
c) Ist die kryptographische Annahme falsch und stören die bösen Teilnehmer die t-te Verteilung während Phase [n+1], so entscheidet sich jeder gute Teilnehmer
   - entweder für „Signaturen gebrochen“ am Ende der t-ten Verteilung,
   - oder er akzeptiert den korrekten Wert am Ende der t-ten Verteilung und beendet das Senden nach Phase [1] der (t+1)-ten Verteilung.

Alle drei Aussagen gelten mit Wahrscheinlichkeit exponentiell nahe 1. ♦

Beweis. a) Da die bösen Teilnehmer keine Signaturen fälschen können, arbeitet das Protokoll wie das ursprüngliche DOLEV-STRONG BÜP und garantiert somit korrekte Übereinstimmung.

b) Dieser Fall bedeutet, daß die Angreifer in den Phasen [1] bis [n–1] die gefälschte Signatur eines guten Teilnehmers P_i an einen anderen guten Teilnehmer senden, oder in Phase [n] an P_i selbst, oder in den Phasen [1] bis [n] einen Fälschungsbeweis an einen guten Teilnehmer P_i. Deshalb empfängt, mit Wahrscheinlichkeit exponentiell nahe 1, jeder gute Teilnehmer in Phase [n+1] der t-ten Verteilung den Fälschungsbeweis von P_i, entscheidet „Signaturen gebrochen“ und beendet das Senden.

c) In diesem Fall senden die bösen Teilnehmer erstmals in Phase [n+1] einen Fälschungsbeweis an einen guten Teilnehmer. Deshalb sind die Entscheidungen gemäß des ursprünglichen DOLEV-STRONG BÜPs korrekt, und jeder gute Teilnehmer empfängt den Fälschungsbeweis spätestens in Phase [1] der (t+1)-ten Verteilung. □

3.2.2 Einsatz von fail-stop Byzantinischer Übereinstimmung für Unbeobachtbarkeit und Robustheit

Zusammen mit dem Fallenprotokoll aus §2.2 (d.h. überall da verwendet, wo das zuverlässige Verteilnetz gebraucht wurde), garantiert das robuste fail-stop BÜP kryptographische Robustheit, während die Unbeobachtbarkeit ausschließlich die Zusammenhangsannahme voraussetzt:

Die einzige Möglichkeit, wie zwei gute Teilnehmer verschieden entscheiden können, ist, daß einer einen Wert v akzeptiert hat, während der andere „Signaturen gebrochen“ entschieden hat (Satz 1). Aber in diesem Fall stoppen alle guten Teilnehmer nach der ersten Phase der nächsten Verteilung (Satz 1c), und diejenigen, die die erste Phase der nächsten Verteilung durchführen, haben den korrekten Wert akzeptiert. Deshalb kann der Angreifer nicht mehr erfahren als eine zusätzliche Ausgabe O_i^t von einem Teilnehmer, der keinen

falschen Wert empfangen hat. Da er diese zusätzliche Ausgabe auch dann empfangen hätte, wenn er nicht gestört hätte, kann ihm dies keine zusätzliche Information geben.

Für die kryptographische Robustheit genügt als einzige kryptographische Annahme z.B. die Faktorisierungsannahme.

Da dieses Verfahren viel weniger effizient ist als die Verfahren aus Fußnote 3, könnte man in der Praxis versuchen, eines von jenen zu verwenden, und nur wenn das ständig gestört wäre, würde man zum hier beschriebenen Verfahren übergehen (von Hand, weil zunächst die Schlüsselverteilung durchgeführt werden muß).

4 Zusammenfassung

Das Ziel dieses Papieres war zu untersuchen, wie auf der Basis des DC-Protokolls (§1.2) unbedingte Sender- und Empfängerunbeobachtbarkeit realisiert und zusätzlich unter gewissen Annahmen Robustheit garantiert werden kann.

Das in [Chau_88] zur Lösung dieses Problems vorgeschlagene Fallenprotokoll (§2.1) kann für einen einfachen aktiven Angriff auf die Unbeobachtbarkeit mißbraucht werden, aber seine Grundidee konnte für die dargestellten sicheren Lösungen verwendet werden (§2.2).

Die Beschränkungen A_{rob} und A_{unb} des Angreifers gegen Robustheit bzw. Unbeobachtbarkeit, die für die verschiedenen Protokolle nötig sind, sind in Bild 2 zusammengefaßt. Ein Strich „–“ bedeutet, daß keine Beschränkung vorliegt. „Krypt. beschr.“ bedeutet, daß der Angreifer nur über beschränkte Berechnungsfähigkeiten verfügt, etwa keine großen BLUM-Zahlen faktorisieren kann.

Zum Vergleich betrachten wir auch die Verfahren aus [Waid_90, WaPf_89]. n ist die Anzahl aller, t die der bösen Teilnehmer.

Protokoll	Abschnitt	A_{rob}	A_{unb}	Unbeobachtbarkeit probabilistisch ?
Überlagerndes Senden	§1.2, [Chau_88]	$t = 0$	zuverl. Verteilung	nein
Fail-stop Verteilung	[Waid_90, §3.2.2.1]	$t = 0$	–	nein
	[Waid_90, §3.2.2.2-3]	$t = 0$	–	ja
Fallenprotokoll und zuverlässige Verteilung	§2.2	Annahme zuverlässiger Verteilung $\wedge$ krypt. beschr.		nein
Fallenprotokoll und Byzantinische Übereinstimmung	§3.1	Zusammenhangsannahme $\wedge$ kryptographisch beschränkt		ja
	Fußnote 12 und §3.1	Zusammenhangsannahme $\wedge\ t < n/3$		nein
	Fußnote 12 und §3.1	Zusammenhangsannahme $\wedge\ t < n/2$		ja
Fallenprotokoll und robuste fail-stop BÜ	§3.2	Zusammenhangsannahme $\wedge$ krypt. beschr.		ja

Bild 2 Zusammenfassung

Unsere in §3.2 beschriebene Lösung basiert auf robuster fail-stop Byzantinischer Übereinstimmung, d.h. kryptographischer BÜ mit der zusätzlichen Eigenschaft daß, sobald ein Angreifer mit unerwarteten Berechnungsfähigkeiten die BÜ stört, alle anderen Teilnehmer dies erkennen.

Im Anhang zeigen wir, wie man unser Protokoll für robuste fail-stop Byzantinische Übereinstimmung mit einem beliebigen nichtkryptographischen BÜP kombinieren kann, so daß das kombinierte BÜP nur von einem Angreifer gestört werden kann, der beide zugrundeliegenden BÜPs stören kann (adaptive BÜ). Dies

heißt zur Zeit, daß er sowohl kryptographisch unbeschränkt sein als auch mindestens ein Drittel (bzw. die Hälfte) aller Teilnehmer kontrollieren muß.

5 Anhang: Adaptive Byzantinische Übereinstimmung

Ein adaptives BÜP sollte funktionieren, wenn entweder kryptographische BÜ oder nichtkryptographische BÜ erreicht werden kann (vgl. §3.1). Die erste Idee hierzu ist natürlich, mit einem kryptographischen BÜP zu beginnen, in dem die Signaturen mit einem Fail-stop-Signatursystem implementiert sind, und ein nichtkryptographisches BÜP in Reserve zu halten.

Im Gegensatz zu robuster fail-stop BÜ braucht man nun einen festen Zeitpunkt, zu dem jeder weiß, ob die Signaturen gebrochen sind und das nichtkryptographische BÜP verwendet werden muß, oder ob die Ergebnisse des kryptographischen BÜPs verwendet werden können. Uneinigkeit hierüber könnte auftreten, wenn die kryptographische Annahme falsch ist und die Angreifer in der letzten Phase einen Fälschungsbeweis an einige, aber nicht an alle guten Teilnehmer gesendet haben. Um zu entscheiden, ob ein Fälschungsbeweis empfangen wurde oder nicht, scheint man wieder eine Art adaptive BÜ zu benötigen. Dies mag wie ein Teufelskreis aussehen. Das Problem kann jedoch gelöst werden, indem diese Entscheidungs-BÜ nicht über die Werte „ja“ oder „nein“, sondern über den Fälschungsbeweis selbst (statt „ja“) stattfindet. Trifft die kryptographische Annahme zu, so kann diese Entscheidungs-BÜ nicht gestört werden, da nur der Wert „nein“ bekannt ist.

Dies legt folgendes Protokoll nahe (nicht das effizienteste, aber leicht zu beweisen):

Adaptives BÜP

[α]	Man führe das DOLEV-STRONG BÜP durch, implementiert mit Fail-stop-Signaturen und der 1. Ergänzung aus §3.2.1.
[β]	Man führe ein nichtkryptographisches BÜP durch.
Für $i = 1,\ldots,n$:	
[γ_i]	Teilnehmer P_i verteilt eine Entscheidungsnachricht D_i mittels des nichtkryptographischen BÜPs. D_i ist entweder ein Fälschungsbeweis, wenn P_i in [*End*] von Schritt [α] einen hatte, oder der Wert „nein“. (In jeder Runde des Protokolls prüft jeder Teilnehmer, ob ein Wert, der ein Fälschungsbeweis sein soll, korrekt ist. Wenn nicht, wird er als „nein“ interpretiert.)
[*End*]	Wenn ein guter Teilnehmer P_j entschieden hat, daß jedes D_i „nein“ ist, verwendet er den Wert aus Schritt [α], andernfalls den Wert aus Schritt [β].

Satz 2: Adaptive Byzantinische Übereinstimmung. Wenn Signaturen unfälschbar sind oder das nichtkryptographische BÜP nicht gestört werden kann, garantiert das adaptive BÜP innerhalb einer festen Anzahl von Phasen BÜ, mit Wahrscheinlichkeit exponentiell nahe 1. ♦

Beweis. Wir unterscheiden drei Fälle:

1. Fall: Die kryptographische Annahme trifft zu, d.h. keine Signaturen werden gefälscht und kein Fälschungsbeweis wird erzeugt.

Dann findet Schritt [α] die korrekte Übereinstimmung (analog Satz 1a). Es bleibt zu zeigen, daß jeder gute Teilnehmer sich in Schritt [*End*] entscheidet, diesen Wert zu verwenden. Dies ist offensichtlich, da in keinem der Schritte [γ_i] jemand einen Fälschungsbeweis senden kann, so daß alle guten Teilnehmer in allen diesen Schritten „nein“ entscheiden.

2. Fall: Die kryptographische Annahme ist falsch, und die bösen Teilnehmer stören Schritt [α], d.h. sie fälschen mindestens eine Signatur eines guten Teilnehmers oder senden einen Fälschungsbeweis an einen guten Teilnehmer.

Dann kann nach Annahme das nichtkryptographische BÜP nicht gestört werden, also findet Schritt [β] die korrekte Übereinstimmung. Es genügt daher zu zeigen, daß jeder gute Teilnehmer den Wert verwendet, den er in Schritt [β] empfangen hat.

Entweder hat ein guter Teilnehmer P_i einen Fälschungsbeweis direkt von den bösen Teilnehmern empfangen, oder in Phase [n+1] von Schritt [α] kann, mit Wahrscheinlichkeit exponentiell nahe 1, mindestens ein guter Teilnehmer P_i einen Fälschungsbeweis berechnen. In beiden Fällen verteilt P_i ihn in Schritt [γ_i] als D_i. Da das nichtkryptographische BÜP nicht gestört werden kann, ist die Verteilung in Schritt [γ_i] korrekt, so daß alle guten Teilnehmer dieses D_i empfangen. Daher verwenden sie in [*End*] alle den in Schritt [β] empfangenen Wert.

3. Fall: Die kryptographische Annahme ist falsch, aber die bösen Teilnehmer verwenden dies nicht, um Schritt [α] zu stören. (Aber vielleicht stören sie die Schritte [γ_i] durch Senden von Fälschungsbeweisen!)

Dann kann nach Annahme das nichtkryptographische BÜP nicht gestört werden. Deshalb sind alle Schritte [γ_i] korrekt. Alle guten Teilnehmer empfangen somit dieselben Nachrichten D_i. Daher verwenden in [*End*] entweder alle guten Teilnehmer die Werte aus Schritt [α], oder sie verwenden alle die Werte aus Schritt [β].

Da das nichtkryptographische BÜP nicht gestört werden kann, ist der Wert, den sie erhalten, wenn sie sich für [β] entscheiden, korrekt. Da Schritt [α] nicht gestört wurde, hat auch Schritt [α] eine korrekte Übereinstimmung gefunden (analog Satz 1c). Jede Entscheidung, die die guten Teilnehmer in [*End*] treffen, ist also korrekt. □

Der 3. Fall mag unnötig kompliziert erscheinen: Wenn beide Schritte [α] und [β] korrekt sind, warum müssen die guten Teilnehmer darin übereinstimmen, welches der Ergebnisse sie verwenden? Dies ist nötig, wenn der Sender böse ist; zwei korrekte Übereinstimmungen können dann verschiedene Werte ergeben!

Bemerkung über Effizienzverbesserungen: Es gibt einige offensichtliche Effizienzverbesserungen:

- Um Nachrichten zu sparen, kann man Schritt [β] aufschieben bis nach den Schritten [γ_i], und nur durchführen, wenn die Entscheidung lautet, daß sein Ergebnis verwendet werden muß. (Der Beweis hat gezeigt, daß alle guten Teilnehmer diesbezüglich übereinstimmen.)
- Man kann über die Korrektheit der Schritte [α] einer beliebigen Anzahl von Verteilungen mit einer Durchführung der Schritte [γ_i] entscheiden. In diesem Fall wählt P_i als D_i einen Fälschungsbeweis, wenn er am Ende irgendeines der Schritte [α] einen hatte.
- Um Phasen zu sparen, kann Schritt [β] parallel mit irgendeinem der anderen Schritte durchgeführt werden.
- Wenn beide Schritte [α] und [β] mit der Verteilung des Wertes des Senders beginnen, genügt eine solche Verteilung (mit Signaturen) für beide Protokolle.
- Die Variante aus Fußnote 16 ist auch hier möglich.

Dank: Wir danken *Andreas Pfitzmann* für viele trickreiche Ideen und *David Chaum* für hilfreiche Diskussionen über den in §2.1 beschriebenen Angriff. Wir danken auch *Birgit Baum-Waidner*, *Manfred Böttger*, *Axel Burandt*, *Klaus Echtle* und den anonymen Gutachtern der VIS '91 für viele wertvolle Hinweise, und der *DFG* für finanzielle Unterstützung.

Literatur

BaPW_90 B. Baum-Waidner, B. Pfitzmann, M. Waidner: Unconditional Byzantine Agreement with Good Majority; Symp. on Theoretical Aspects of Computer Science (STACS), Hamburg 1991; LNCS, Springer-Verlag, Berlin 1991.

BeGW_88 M. Ben-Or, S. Goldwasser, A. Wigderson: Completeness theorems for non-cryptographic fault-tolerant distributed computation; 20th Symposium on Theory of Computing (STOC '88), ACM, New York 1988, 1-10.

BlGo_84 M. Blum, S. Goldwasser: An Efficient Probabilistic Public-Key Encryption Scheme Which Hides All Partial Information; Crypto '84, LNCS 196, Springer-Verlag, Berlin 1985, 289-299.

BoBo_90 J. Bos, B. den Boer: Detection of Disrupters in the DC Protocol; Eurocrypt '89, LNCS 434, Springer-Verlag, Berlin 1990, 320-327.

BrCC_88 G. Brassard, D. Chaum, C. Crépeau: Minimum Disclosure Proofs of Knowledge; Journal of Computer and System Sciences 37 (1988) 156-189.

CGMA_85 B. Chor, S. Goldwasser, S. Micali, B. Awerbuch: Verifiable secret sharing and achieving simultaneity in the presence of faults; 26th Symposium on Foundations of Computer Science (FOCS) 1985, IEEE, 1985, 383-395.

Chau_85 D. Chaum: Security without Identification: Transaction Systems to make Big Brother Obsolete; Communications of the ACM 28/10 (1985) 1030-1044.

Chau_88 D. Chaum: The Dining Cryptographers Problem: Unconditional Sender and Recipient Untraceability; Amsterdam 1985; Journal of Cryptology 1/1 (1988) 65-75.

ChDG_88 D. Chaum, I. Damgård, J. van de Graaf: Multiparty Computations ensuring privacy of each party's input and correctness of the result; Crypto '87, LNCS 293, Springer-Verlag, Berlin 1988, 87-119.

ChDw_89 B. Chor, C. Dwork: Randomization in Byzantine Agreement; JAI Press, Advances in Computing Research Vol. 5, Greenwich (Connecticut) 1989, 443-497.

DFFL_82 D. Dolev, M. Fischer, R. Fowler, N. Lynch, H. Strong: An Efficient Algorithm for Byzantine Agreement without Authentication; Information and Control 52 (1982) 257-274.

DiHe_76 W. Diffie, M. Hellman: New Directions in Cryptography; IEEE Tr. on Information Theory 22/6 (1976) 644-654.

DoSt_83 D. Dolev, H. Strong: Authenticated Algorithms for Byzantine Agreement; SIAM J. Comput. 12/4 (1983) 656-666.

FeMi_88 P. Feldman, S. Micali: Optimal algorithms for byzantine agreement; 20th Symposium on Theory of Computing (STOC) 1988, ACM, New York 1988, 148-161.

Fisc_83 M. Fischer: The Consensus Problem in Unreliable Distributed Systems (A Brief Survey); 4th Conference on Foundations of Computation Theory, 1983, 127–140.

GMW_87 O. Goldreich, S. Micali, A. Wigderson: How to play any mental game - or - a completeness theorem for protocols with honest majority; 19th STOC 1987, ACM, New York 1987, 218-229.

GoMi_84 S. Goldwasser, S. Micali: Probabilistic Encryption; J. of Computer and System Sciences 28 (1984) 270-299.

GoMR_88 S. Goldwasser, S. Micali, R. Rivest: A Digital Signature Scheme Secure Against Adaptive Chosen-Message Attacks; SIAM J. Comput. 17/2 (1988) 281-308.

GoPe_90 O. Goldreich, E. Petrank: The Best of Both Worlds: Guaranteeing Termination in Fast Randomized Byzantine Agreement Protocols; Information Processing Letters 36 (1990) 45-49.

LuPW_90 J. Lukat, A. Pfitzmann, M. Waidner: Effizientere fail-stop Schlüsselerzeugung für das DC-Netz; Datenschutz und Datensicherung DuD 15/2 (1991).

PeSL_80 M. Pease, R. Shostak, L. Lamport: Reaching Agreement in the Presence of Faults; JACM 27/2 (1980) 228-234.

Pfit_85 A. Pfitzmann: How to implement ISDNs without user observability - Some remarks; Fakultät für Informatik, Universität Karlsruhe, Interner Bericht 14/85.

Pfit_89 B. Pfitzmann: Für den Unterzeichner sichere digitale Signaturen und ihre Anwendung; Diplomarbeit am Institut für Rechnerentwurf und Fehlertoleranz der Universität Karlsruhe, 1989.

Pfit_90 A. Pfitzmann: Diensteintegrierende Kommunikationsnetze mit teilnehmerüberprüfbarem Datenschutz; IFB 234, Springer-Verlag, Heidelberg 1990.

PfPW_88 A. Pfitzmann, B. Pfitzmann, M. Waidner: Datenschutz garantierende offene Kommunikationsnetze; Informatik-Spektrum 11/3 (1988) 118-142.

PfWa_90 B. Pfitzmann, M. Waidner: Formal Aspects of Fail-stop Signatures; Fakultät für Informatik, Universität Karlsruhe, Interner Bericht 22/90, Dezember 1990.

PfWa_91 B. Pfitzmann, M. Waidner: Fail-stop-Signaturen und ihre Anwendung; VIS '91; in *diesem* Tagungsband.

PWP_90 B. Pfitzmann, M. Waidner, A. Pfitzmann: Rechtssicherheit trotz Anonymität in offenen digitalen Systemen; Datenschutz und Datensicherung DuD 14/5-6 (1990) 243-253, 305-315.

RaBe_89 T. Rabin, M. Ben–Or: Verifiable Secret Sharing and Multiparty Protocols with Honest Majority; 21st Symposium on Theory of Computing (STOC) 1989, ACM, New York 1989, 73-85.

Rabi_80 M. Rabin: Probabilistic Algorithms in Finite Fields; SIAM J. Comput. 9/2 (1980) 273-280.

Reis_87 R. Reischuk: Konsistenz und Fehlertoleranz in Verteilten Systemen - Das Problem der Byzantinischen Generäle; 17. GI Jahrestagung, IFB 156, Springer-Verlag, Berlin 1987, 65-81.

Waid_90 M. Waidner: Unconditional Sender and Recipient Untraceability in spite of Active Attacks; Eurocrypt '89, LNCS 434, Springer-Verlag, Berlin 1990, 302-319.

WaPf_89 M. Waidner, Birgit Pfitzmann: Unconditional Sender and Recipient Untraceability in spite of Active Attacks – Some Remarks; Fakultät für Informatik, Universität Karlsruhe, Interner Bericht 5/89, März 1989.