1989.09.04

Unconditional Sender and Recipient Untraceability in spite of Active Attacks

Michael Waidner
Institut für Rechnerentwurf und Fehlertoleranz, Universität Karlsruhe
Postfach 6980, D-7500 Karlsruhe 1, F. R. Germany

Abstract. A protocol is described which allows to send and receive messages anonymously using an arbitrary communication network, and it is proved to be unconditionally secure.

This improves a result by DAVID CHAUM: The DC-net guarantees the same, but on the assumption of a reliable broadcast network. Since unconditionally secure Byzantine Agreement cannot be achieved, such a reliable broadcast network cannot be realized by algorithmic means.

The solution proposed here, the DC$^+$-net, uses the DC-net, but replaces the reliable broadcast network by a fail-stop one. By choosing the keys necessary for the DC-net dependently on the previously broadcast messages, the fail-stop broadcast can be achieved unconditionally secure and without increasing the complexity of the DC-net significantly, using an arbitrary communication network.

Categories and Subject Descriptors: C.2.0 [**Computer-Communication Networks**]: General — *Security and protection*, E.3 [**Data Encryption**], F.2.1 [**Analysis of Algorithms and Problem Complexity**]: Numerical Algorithms and Problems

General Terms: Security, Algorithms

Additional Key Words and Phrases: Privacy, Untraceability, Unconditional Security, Fail-Stop Broadcast

1 Overview

In [Chau_88] DAVID CHAUM describes a technique, the DC-net, which should allow to send and receive messages anonymously using an arbitrary communication network.

Section 2 gives a short and slightly generalized description of the sending mechanism of the DC-net, called **superposed sending** here. Some known efficient and anonymity preserving multi-access protocols for using the multi-access channel superposed sending offers are described.

In [Chau_88] the untraceability of senders and recipients of messages is proved to be *unconditional*, but this proof implicitly assumes a *reliable* broadcast network, i.e. each message broadcast by an honest participant is received by each other participant without being changed. Since *unconditional* Byzantine Agreement (i.e. Byzantine Agreement in spite of an attacker with unlimited computational power who may control an arbitrary number of participants) is impossible, such a network cannot be realized by cryptographic means. Thus the assumption may be rather unrealistic.

In Section 3 it is shown how the sending of a specific participant X can be traced by an active attacker who is able to alter the messages received by X and who controls the current communication partner of X. A number of countermeasures, called **fail-stop broadcast** schemes, are proposed, and it is proved that each one will achieve the desired *unconditional* untraceability in spite of active attacks, independent of the underlying communication network. Superposed sending together with fail-stop broadcast is called a **DC$^+$-net**.

Without any further measures, the **serviceability** of the multi-access channel of superposed sending is not good: each faulty or dishonest participant can untraceably and enduringly disturb the channel.

Unfortunately no measures are known which can guarantee serviceability while preserving *unconditional* untraceability using *arbitrary* communication networks. Therefore this problem is not further considered here.

A scheme which guarantees **nearly unconditional untraceability** and computationally secure serviceability (i.e. untraceability if the attacker cannot prevent the honest participants from communicating and serviceability if additionally the attacker is computationally restricted) is described in [WaPf_89, WaPf1_89].

2 Unconditional sender untraceability

Section 2.1 describes the basic mechanisms of the DC-net, superposed sending and broadcast, and defines the notation used throughout this paper. Section 2.2 describes anonymity preserving multi-access protocols for superposed sending, and in Section 2.3 some general remarks on sender untraceability schemes are given.

2.1 Superposed sending

Assume that a number of participants want to exchange messages over an arbitrary communication network. A computationally unlimited attacker, who is able to eavesdrop communication between any two of the participants (e.g. because he collaborates with the network operator) and who controls an arbitrary subset of the participants, tries to trace the messages exchanged between the participants to their senders and recipients.

If all messages are delivered to each participant, the attacker is not able to trace the *intended* recipient of a message. Therefore unconditionally reliable broadcast guarantees **unconditional recipient untraceability**.

It is important to notice that in this section, as in [Chau_88], attackers are assumed to be unable to manipulate the consistency of broadcast.

Sender untraceability is guaranteed by **superposed sending**, which realizes an anonymous multi-access channel:

Let $P = \{P_1, \ldots, P_n\}$ be the set of all participants and let G be an undirected self-loop free graph with nodes P. Let $(F,\oplus)$ be a finite abelian group. The set F is called the **alphabet**.

To be able to perform a single sending step, called a **round**, each pair of participants P_i, P_j who are directly connected by an edge of G choose a key K_{ij} from F randomly[1]. Let $K_{ji} := K_{ij}$. Participants P_i and P_j keep their common key secret. The graph G is called **key graph**, the matrix K of all keys is called **key combination**.

Each participant P_i chooses a message character M_i from the alphabet F, outputs his *local* sum

$$O_i := M_i \oplus \sum_{\{P_i,P_j\} \in G} \mathrm{sign}(i\text{-}j) \bullet K_{ij} \qquad (2.1)$$

and receives as input the *global* sum

[1] In the following, the term "X is randomly chosen from a set M" is abbreviated by "$X \in_R M$". This means that X is a uniformly distributed random variable which is independent of "all other variables". What is meant by "all other variables" should always be clear from the context.

$$S := \sum_{j=1}^{n} O_j \tag{2.2}$$

As usual, the symbolic operation $\pm 1 \bullet K_{ij}$ is defined by $+1 \bullet x := x$ and $-1 \bullet x := -x$.

Superposed sending realizes an **additive multi-access channel** with (digital) collisions, which is stated in the following lemma.

Lemma 2.1 If the local sums are computed according to (2.1), then the global sum defined in (2.2) is equal to the sum of all message characters:

$$S = \sum_{j=1}^{n} M_j \tag{2.3}$$

Proof. In (2.2) each key is both added and subtracted exactly once. □

If exactly one character M_i has been chosen unequal to 0, this character is successfully delivered to all participants. Otherwise a (digital) collision occurs which has to be resolved by a multi-access protocol, cf. Section 2.2.

Superposed sending guarantees **unconditional sender untraceability**. Let A denote the subset of participants controlled by the attacker. If the graph $G \setminus (P \times A)$ is connected, the attacker gets no additional information about the characters M_i besides their sum.

Lemma 2.2 **Superposed sending.** Let A be the subset of participants controlled by the attacker and assume $G \setminus (P \times A)$ to be connected. Let $(O_1, \ldots, O_n) \in F^n$ be the output of a single round.

Then for each vector $(M_1, \ldots, M_n) \in F^n$ which is consistent with the attacker's a priori knowledge about the M_i and which satisfies

$$\sum_{j=1}^{n} O_j = \sum_{j=1}^{n} M_j \tag{2.4}$$

the same number of key combinations exist which satisfy Equation (2.1) and which are consistent with the attacker's a priori knowledge about the K_{ij}.

Hence the conditional probability for $(M_1, \ldots, M_n)$ given the output $(O_1, \ldots, O_n)$ (i.e. the a posteriori probability) is equal to the conditional probability for $(M_1, \ldots, M_n)$ given the sum in (2.4) only (i.e. the a priori probability).

This is stated and proved in [Chau_88] for $F = \mathrm{GF}(2)$ by a technique which can easily be applied to any finite field. In [Pfit_89 Sect. 2.5.3.1] and in the following, Lemma 2.2 is proved for any finite abelian group F. (The general applicability of finite abelian groups was also mentioned in [Pfi1_85].)

Proof. Let $M' := (M'_1, \ldots, M'_n) \in F^n$ be another vector which satisfies (2.4) and which is consistent with the attacker's a priori knowledge about the M_i.

To prove Lemma 2.2, a finite sequence $M^0, M^1, \ldots$ of vectors from F^n is defined, which all satisfy Eq. (2.4) and which differ in only two components. Let $M^k = (M^k{}_1, \ldots, M^k{}_n)$.

Let $\mathrm{M}^0 := (M_1, \ldots, M_n)$, hence M^0 satisfies Eq. (2.4). If $M^k = M'$ then stop. Now assume $M^k \neq M'$. Since both M^k (by induction hypothesis) and M' satisfy Eq. (2.4) there are at least two different indices i, j with $M^k{}_i \neq M'_i$ and $M^k{}_j \neq M'_j$, and since both M^k and M' are consistent with the attacker's a priori knowledge $P_i, P_j \notin A$. Define

$$M^{k+1}{}_i := M'_i$$

$$M^{k+1}{}_j := M^k{}_j \oplus M^k{}_i \oplus -M'_i \tag{2.5}$$
$$M^{k+1}{}_h := M^k{}_h \text{ for all } h \notin \{i, j\}$$

Obviously M^{k+1} satisfies (2.4). After a maximum of n - 1 steps the sequence stops with $M^k = M'$.

Let K^k be the set of all key combinations which satisfy (2.1) for the vector M^k and which are consistent with the attacker's a priori knowledge. Between each pair K^k, K^{k+1} a bijection ϕ^k is defined. Hence $|K^k| = |K^{k+1}|$ for all k and therefore $|K^0| = |K^{n-1}|$ where $M^{n-1} = M'$.

To define ϕ^k consider the equations (2.5). Let $\Delta := M^{k+1}{}_i \oplus -M^k{}_i$. Then $M^{k+1}{}_i = M^k{}_i \oplus \Delta$ and $M^{k+1}{}_j = M^k{}_j \oplus -\Delta$.

Because of the connectivity of $G \setminus (P \times A)$ a path $(P_i = P_{k_1}, \ldots, P_{k_m} = P_j)$ exists with $P_{k_h} \notin A$ and $(P_{k_h}, P_{k_{h+1}}) \in G \setminus (P \times A)$. Let $K \in K^k$. Then $\phi^k(K)$ is defined by changing the keys on this path appropriately:

$$\forall\ h = 1, \ldots, m\text{-}1\text{:}\ \ \phi^k(K)_{k_h k_{h+1}} := K_{k_h k_{h+1}} \oplus -\Delta \bullet \text{sign}(k_h - k_{h+1}),$$
$$\phi^k(K)_{k_{h+1} k_h} := \phi^k(K)_{k_h k_{h+1}}$$

and

$$\forall\ (f, g) \notin \{\ (k_h, k_{h+1}), (k_{h+1}, k_h) \mid h=1, \ldots, m\text{-}1\ \}\text{:}\ \ \phi^k(K)_{fg} := K_{fg}$$

The construction of ϕ^k is depicted in Figure 1.

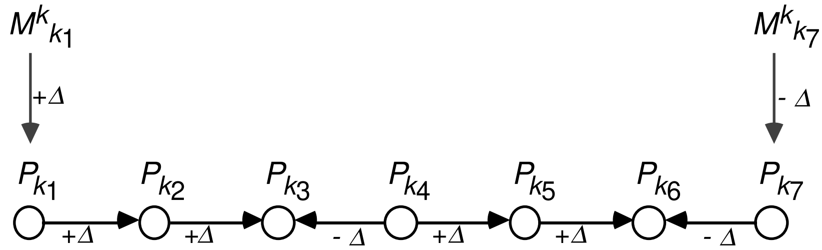
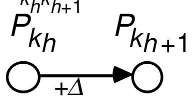


Figure 1 Construction of ϕ^k from a path with m = 7. The vertical arrows indicate the change of $M^k{}_{k_h}$, h=1, 7, the horizontal arrows the numerical order of the k_h, and the $\pm\Delta$ the change of $K_{k_h k_{h+1}}$:



:⇔ $k_h < k_{h+1}$ and therefore $\phi^k(K)_{k_h k_{h+1}} := K_{k_h k_{h+1}} \oplus \Delta$

Obviously, the local outputs of P_{k_h}, h = 1, …, 7, are not changed by ϕ^k.

It can easily be checked that $\phi^k(K)$ satisfies (2.4). Because $\phi^k(K)$ differs from K only in keys which are unknown to the attacker, $\phi^k(K)$ is necessarily consistent with the attacker's a priori knowledge. Since ϕ^k is simply a translation of the group $F^{|G|}$, the bijectivity of ϕ^k is obvious. □

2.2 Efficient and anonymity preserving multi-access protocols

To use the multi-access channel superposed sending offers, it is necessary to regulate the participants' access to the channel by an appropriate, i.e. efficient and anonymity preserving protocol.

For an in depth discussion of possible protocols cf. [Pfit_89 Sect. 3.1.2]. In the following, some protocols are mentioned, and two protocols are described in detail: a reservation map technique and superposed receiving.

The first step for each multi-access protocol is to combine a fixed number c of characters into a **message**. Each message is transmitted in c consecutive rounds, which are called a **slot**.

In the following, rounds are numbered from 1 to a maximum number t_{max}. Parameter t_{max} is necessary for technical reasons only. Usually $\mathrm{ld}(|F|) \bullet t_{max}$, i.e. the maximum number of transmitted bits, can be assumed to be very large, e.g. $\mathrm{ld}(|F|) \bullet t_{max} = 10^{25}$. Even with a rather unrealistic transmission rate of 10^{15} bps this is sufficient for about 317 years of superposed sending.

The character and output of participant P_i in round t are named M_i^t and O_i^t, respectively, the global sum in round t is named S^t.

The simplest protocol is the well known (slotted) **ALOHA** [Chau_88, Tane_88 Sect. 3.2]: If P_i has a message to send, he simply does so in the next slot. If another participant has decided to send a message, too, a collision occurs, which is detected by P_i. After waiting a random number of slots, P_i retransmits his message. Obviously ALOHA preserves anonymity, but wastes the transmission capacity of the network.

2.2.1 Reservation map technique

To avoid collisions of messages, a simple **reservation map technique** can be used: a slot of r rounds, the **reservation frame**, is used to reserve the following up to r slots [Pfi1_85 Sect. 2.2.2.2].

Let m be an arbitrary integer $\geq n$, and let F be the additive group of integers modulo m. For each message P_i plans to send, he chooses an index k from $\{1, \ldots, r\}$ at random and outputs 1 as his k-th character for the reservation frame. The resulting reservation message consists of three classes of characters: 0, indicating an unreserved slot, 1, indicating a reserved slot, and $\{2, \ldots, m\text{-}1\}$, indicating collisions. Since all message slots with corresponding reservation character $\neq 1$ are of no use, they are skipped, i.e. the reservation frame is followed only by as many message slots as there are successful reservations. A slot with reservation character $=1$ is used by the participant who has sent a 1 in the corresponding reservation round.

A similar reservation technique, the **bit-map reservation technique**, is described in [Chau_88]: instead of using a relatively large group F to enable the detection of multiple collisions, the superposition is done in $F = GF(2)$ and a value r in the order of the square of s_{max}, the maximum number of reservations, is used to make multiple collisions of an odd number of reservations rather unlikely. Therefore the scheme requires $s_{max}{}^2$ additional bits per s_{max} sent messages.

2.2.2 Superposed receiving

The following two **collision resolution techniques** are based on the observation that

- collisions on this channel, in contrast to an analog one, carry useful information, namely the sum of all collided messages, and
- it is possible to *compute* s collided message characters from s well-defined "collisions", i.e. sums.

Therefore these techniques are subsumed under the name **superposed receiving**.

The first one is an algorithm suggested by ANDREAS PFITZMANN in [Pfit_89 Sect. 3.1.2] and called **tree-like collision resolution with superposed receiving.** It improves CAPETANAKIS collision resolution algorithm described in [Mass_81] using the fact that the set of s collided characters M_i can be computed from each set of s linearly independent sums of these characters. (A performace evaluation of the following can be found in [Marc_88].)

Let s_{max} be the maximum number of collided messages, e.g. $s_{max} = n$, and $\{0, 1, 2, \ldots, M_{max}\} \subset \mathbb{Z}$ the set of all *allowed* message characters. The alphabet F is chosen to be the ring of integers modulo m where m is greater than $s_{max} \bullet M_{max}$. As usual each character $M \in F$ can be interpreted as an integer. A message consists of two characters: For a participant who has to send a message, the first character is always 1 and the second is his message character. For a nonsending participant both are always 0.

Now assume that a new round of the protocol starts and an a priori unknown number of participants have decided to send a message. Let SP denote the set of all sending participants, Σ the sum of their characters M_i modulo m, and $s := |SP|$.

Thus the first slot contains the pair (s, Σ). A number $s \geq 2$ indicates a (digital) collision. To resolve it, each participant computes the average message $M_A := \lfloor \Sigma / s \rfloor$, which is possible, since the modulus m has been chosen so large that Σ is also the sum of the characters in $\mathbb{Z}$.

This average is used to deterministically divide the set SP into two disjoint subsets SP_1 and SP_2: SP_1 consists of all participants $P_i \in SP$ with $M_i \leq M_A$, SP_2 consists of all other sending participants. For $i = 1, 2$ define s_i and Σ_i in analogy to s and Σ.

All participants $P_i \in SP_1$ immediately repeat their messages $(1, M_i)$ in the next slot, hence each user *receives* the pair (s_1, Σ_1) and can *compute* the pair $(s_2, \Sigma_2) = (s \oplus -s_1, \Sigma \oplus -\Sigma_1)$.

Given the rare case $s_2 = 0$, the protocol terminates after the second slot: each participant $P_i \in SP$ has sent the same character $M_i = M_A$. Otherwise, i.e. $s_2 \neq 0$, the sets SP_1 and SP_2 are both nonempty and the collision resolution procedure is recursively applied to (s_i, Σ_i), $i = 1, 2$.

To resolve a collision of s messages, the protocol deterministically needs a maximum of s slots, i.e. $s \bullet (2 \bullet \log(s_{max}) + \log(M_{max}))$ bits.

The second one is suggested by JURJEN BOS and BERT DEN BOER in [BoBo_89] and called **root-finding collision resolution with superposed receiving**. It is based on the observation that the collision of s different characters can be resolved using the sums of the first s powers of the characters (where F is a sufficiently large finite field).

The collision is resolved by computing the coefficients of a polynomial whose zeros are exactly the collided characters, and by factoring this polynomial. Since factoring is an expensive task [Rab1_80] the computational complexity of this technique is much higher than that of tree-like collision resolution. On the other hand it needs only $s \bullet \log(M_{max})$ bits.

To allow long messages to be sent, either the alphabet F could be made large enough to represent a long message by a single character, or superposed receiving could be used as a reservation technique (**reservation by superposed receiving**):

Each participant willing to send a message chooses a reservation message RM_j at random and sends it in the next possible reservation phase. The collision of all s reservation messages is resolved, after which each P_i sorts the received reservation characters RM_j according to their numerical values. The order of the RM_j naturally defines an order of all reserving participants, according to which each P_j sends his real message in the appropriate one of the next s slots. (To increase the fairness of the reservation scheme, this order can be shifted cyclically by an index randomly chosen by all participants together [BoBo_89].)

The probability of collisions is exponentially small in $\log(|F|)$.

2.3 Some remarks on sender untraceability schemes

Given the very strong assumption of an unlimited attacker (i.e. there may be an arbitrary number of attackers $|A| < |P|$, there are no computational restrictions) the fundamental restrictions of superposed sending as far as performance and reliability are concerned are a consequence of its sender untraceability: In order to make the physical behaviour of a participant meaningless, it is necessary that a participant P_i who is willing to send a character M_i

- does this in an encrypted way,
- each other participant P_j outputs a character, too, and
- the attacker is not able to learn anything about M_i before knowing all the outputs.

Because the attacker is assumed to be an insider it follows from the last fact that the result of such a single sending step cannot contain more information than the last of the participant's output does. Therefore any unconditional sender untraceability scheme realizes a multi-access channel and superposed sending offers the best possible channel capacity as far as only a single round is concerned [Pfit_87].

To guarantee the unconditional sender untraceability, the global output of the realized multi-access channel has to depend on each participant's output, therefore any unconditional sender untraceability scheme can be untraceably disturbed by each participant.

As far as I know, superposed sending is the only *unconditional* sender untraceability scheme which withstands an unlimited attacker.

There are two other untraceability schemes known from literature, the MIX-net [Chau_81] and the concept of physical unobservability [Pfit_84]. Both can only withstand weaker attackers than superposed sending. The first is based on the use of a public-key cryptosystem and the existence of a number of network stations, called MIXes, at least one of which has to be trustworthy. The second assumes that the attacker only controls a very small number of participants.

In [BeGW_88, ChCD1_88, Chau1_89] very general techniques for information theoretically secure fault tolerant distributed computations are described. In general these techniques can be used for implementing a sender and recipient untraceability scheme, but they can only withstand attackers with $3 \cdot |A| < |P|$ and are therefore not further considered here. Also, an untraceability scheme based on such a general technique would be far more expensive than superposed sending with fail-stop key generation described in Section 3.2.

To reduce the tremendous number of randomly chosen keys for superposed sending which have to be exchanged by the participants, one can use keys which are generated by pseudorandom bitgenerators (PRBG). If the PRBG used is cryptographically strong, i.e. if distinguishing the PRBG from a true random source in random polynomial time is provably equivalent to solving a (hopefully) hard problem [VaVa_85], tracing becomes equivalent to this hard problem, too, but the *unconditional* sender untraceability is lost.

Because of the growing importance of public telecommunication networks, it seems necessary to look for efficient implementations of untraceability schemes resulting in networks without user observability. For details about the motivation and the more practical aspects of this task cf. [Cha8_85, Pfi1_85, PfWa_86, PfPW_88, Pfit_89].

3 Active attacks on untraceability

The power of an *active* attack is based on a very simple observation: for services using two-way communication it is impossible to realize unconditional sender untraceability without unconditional recipient untraceability and vice versa.

To see this, assume that one of the participants controlled by the attacker, say P_a, communicates with some honest participant X, and that X will answer a message M by sending a message M'. If the attacker is able to identify the *sender* of M', he can identify the *recipient* of M and vice versa. If the attacker doesn't control P_a, the same is true for light traffic; then the attacker can identify both communication partners.

In general if sending and receiving is correlated (which is usually the case) the attacker can always learn something about recipients from identifying senders and vice versa.

If active attacks are possible, superposed sending doesn't guarantee recipient untraceability and therefore it doesn't guarantee sender untraceability:

Let I_i (I_i^t) be the input character which participant P_i receives (in round t) and which should always be equal to the global sum S (S^t).

Assume that the attacker is able to deliver an arbitrary character I^*_i to each participant P_i instead of the correct character I_i. This may be possible e.g. if the DC-net is implemented using a star whose centre collaborates with the attacker. Further assume that participant P_a, who is controlled by the attacker, communicates with the honest participant X according to some protocol. P_a knows that X will always answer to a received message M within a given time by sending a message M'.

If the attacker delivers message M consecutively to a single participant only, and a meaningless message to all others, he can always identify X by checking whether he receives M' or not. Instead of delivering M to a single participant only, he can deliver it to a subset of the participants. By successively partitioning the participants he can identify X in log(n) rounds, provided that the protocol between X and P_a consists of at least log(n) interactions (on average log(n) / 2 interactions would suffice).

One could argue that this attack can be avoided by using networks for which it is physically more difficult to manipulate the input characters of *all* participants, e.g. trees or rings.

But even if the attacker can only manipulate what a *single* participant P_i receives, at least the unobservability of this participant is essentially decreased, since the attacker can test whether he is communicating with P_i or not.

In case of a network where the participants' stations actively forward other participant's messages (e.g. the ring implementation suggested in [Chau_88]), this attack can be performed by the neighbors of P_i without any technical manipulation. Generally, it can be performed by physically disturbing the channel of P_i, or by physically disconnecting P_i from the real network and connecting it to a simulated one with similar physical characteristics.

(If for the case of a ring network one assumes that this attack is not possible, or that observability of single participants is acceptable, i.e. that generally neither participants attack their neighbors nor attackers are able to manipulate cables between participants, then it is more efficient to use the concept of physical unobservability [Pfit_84] to realize untraceability than to use superposed sending.)

If it were *guaranteed* that in all rounds $t = 1, \ldots, t_{max}$ each participant not controlled by the attacker receives the same input character, then superposed sending would guarantee unconditional sender and recipient untraceability in the presence of arbitrary active attacks. Such a network is called a **DC$^+$ -net**.

For an a priori given number t_{max} of rounds this is the well known problem of **reliable broadcast**. Instead of using a fixed t_{max} one can also try to limit t_{max} adaptively: if in round t two honest participants receive different characters then t_{max} is set to t; this is called **fail-stop broadcast** here.

3.1 Reliable broadcast

Reliable broadcast is defined by the following two properties [PeSL_80]: in each round t

i. every two honest participants P_i and P_j receive the same character, i.e. $I_i^t = I_j^t$, and

ii. if the "sender" X is honest, then each honest participant receives the character sent by X.

If superposition of local sums is done by a central station, e.g. the centre of a star network, which delivers the global sum to all participants, only the centre has the function of a "sender". If each participant receives the local sum of each other and computes the global sum locally for himself, each participant acts as "sender".

Some types of networks, e.g. satellite networks, offer reliable broadcast without any additional protocol, but because of their bandwidth limitations they are not very usual in two-way telecommunication. Also the DC-network is meant to be usable with a variety of underlying communication networks, e.g. rings, therefore a cryptographic solution should be preferred to a physical one.

The problem of achieving reliable broadcast on a network which does not provide it automatically is also known as the Byzantine Generals problem, its solution by protocols as **Byzantine Agreement** [PeSL_80, LaSP_82].

It has been proved that *information-theoretically* secure protocols for reliable broadcast exist iff the number of honest participants is greater than twice the number of dishonest participants, i.e. $|P| > 3 \bullet |A|$, and the attacker is not able to prevent communication between honest participants [LaSP_82]. All protocols for information-theoretically secure reliable broadcast implicitly make use of perfect authentication codes [GiMS_74, Sim3_88] and therefore require a large number of additional secret keys exchanged by the participants.

Based on the existence of secure signatures there are reliable broadcast protocols for arbitrary numbers $|A| < |P|$ [LaSP_82]. An adaptive Byzantine Agreement protocol, i.e. one which withstands an attacker with $3 \bullet |A| < |P|$ *or* $|A| < |P|$ and A is computationally restricted, is described in [WaPf_89, WaPf1_89].

Because of its severe limitation $3 \bullet |A| < |P|$ reliable broadcast does not seem to be a useful technique for the desired unconditional recipient untraceability and is therefore not further considered here.

Fail-stop broadcast combines both advantages: it can be implemented in a more efficient way than reliable broadcast and it is unconditionally secure in spite of arbitrary attackers.

3.2 Fail-stop broadcast

The goal of fail-stop broadcast is to stop message transmission as soon as two honest participants receive different input characters.

If such a difference is detected by an honest participant P_i, the fail-stop can easily be performed: P_i simply disturbs the superposed sending in the subsequent rounds by choosing his outputs randomly from F instead of following Eq. (2.1). Then the global sums of all subsequent rounds are independent of the message characters.

In Section 3.2.1 the most obvious, but inefficient, implementation of this idea by a comparison protocol is discussed.

In Section 3.2.2 fail-stop key generation schemes are described: they generate keys for superposed sending dependent on the received input characters and ensure that two participants who have received different input characters will use completely independent keys (at least with high probability) and thus will stop message transmission.

It is shown that the most efficient key generation scheme (Sect. 3.2.2.2) does not affect the performance and reliability characteristics of pure superposed sending.

3.2.1 Comparison of input characters

To detect a difference, the participants can explicitly compare their input characters using an additional protocol: After each round of superposed sending each participant P_i sends his input character I_i to all participants P_j with $j > i$. If an honest participant P_j receives an input character unequal to I_j from another participant P_i, or if he receives nothing from a P_i with $i < j$, he will disturb superposed sending in all subsequent rounds.

Such test phases are well known from Byzantine Agreement protocols.

To make the tests dependable, communication between P_i and P_j should be protected by a perfect authentication scheme [GiMS_74, Sim3_88], i.e. a scheme which allows the attacker to successfully forge a message with probability at the most $1 / \sqrt{|F|}$, if F is used as key space. An additional message and a secret key are therefore necessary for each test.

The number of tests necessary can be determined according to the attacker's assumed power: define G^* to be an undirected graph whose nodes are the participants. Two participants P_i and P_j are directly connected in G^* iff P_i and P_j compare their input characters. In analogy to superposed sending, the following Lemma 3.1 holds:

Lemma 3.1 Let A be the subset of participants controlled by the attacker and assume $G^* \setminus (P \times A)$ to be connected.

If two honest participants P_i and P_j receive different input characters I_i, I_j, then there exists a pair of honest participants $P_{i'}$ and $P_{j'}$ who are directly connected in G^* and who also receive different input characters.

Hence either $P_{i'}$ or $P_{j'}$ detects the difference and disturbs superposed sending.

Proof. Because of the connectivity of $G^* \setminus (P \times A)$ there exists a path $(P_i = P_{k_1}, \ldots, P_{k_m} = P_j)$ with $P_{k_z} \notin A$ and $(P_{k_z}, P_{k_{z+1}}) \in G^* \setminus (P \times A)$. It is assumed that $I_i \neq I_j$, hence there exists an index z such that $I_{k_z} \neq I_{k_{z+1}}$. Choose $(i', j') = (k_z, k_{z+1})$. □

Obviously the connectivity of $G^* \setminus (P \times A)$ is a necessary condition.

The scheme requires $|G^*|$ additional messages in each round, which is usually in the order of $O(n^2)$. If $G = G^*$, and if it is assumed that for each test message the authentication scheme requires a key chosen from F, the number of privately exchanged keys is increased by a factor of two in comparison with pure superposed sending.

In a physical broadcast environment, the number of test messages can be reduced to $O(n)$ broadcast messages by using a digital signature scheme [DiHe_76, GoMR_88] instead of an authentication scheme. But this results in scheme which is computationally secure only.

3.2.2 Message dependent key generation

3.2.2.1 Deterministic fail-stop key generation

A more efficient realization of fail-stop broadcast is obtained by combining the tasks of detecting differences and stopping the network: if the keys K_{ij} and K_{ji} used for superposed sending depend completely (but not exclusively) on the characters received by P_i and P_j, then a difference between I_i and I_j will automatically disturb superposed sending, thereby stopping message transmission.

Define $\delta_{ij}^t := K_{ij}^t \oplus -K_{ji}^t$ and $\varepsilon_{ij}^t := I_i^t \oplus -I_j^t$ for all i, j, t. A key generation scheme for superposed sending is required which guarantees for all P_i and P_j directly connected in G:

SS *Superposed sending:* If for all rounds $s = 1, \ldots, t\text{-}1$ the equation $I_i^s = I_j^s$ holds, then the keys K_{ij}^t and K_{ji}^t for round t are *equal* and randomly selected from F. More formally:

$$[\forall\ s \in \{1, \ldots, t\text{-}1\}\colon \varepsilon_{ij}^s = 0] \Rightarrow K_{ij}^t \in_R F \text{ and } \delta_{ij}^t = 0$$

Then superposed sending works as usual.

FS *Fail-stop:* If there exists an index $s < t$ with $I_i^s \neq I_j^s$, then the keys K_{ij}^t and K_{ji}^t for round t are *independently* and randomly selected from F. More formally:

$$[\exists\ s \in \{1, \ldots, t\text{-}1\}\colon \varepsilon_{ij}^s \neq 0] \Rightarrow K_{ij}^t \in_R F \text{ and } \delta_{ij}^t \in_R F$$

Superposed sending is disturbed by any such pair, i.e. the global sum is independent of the message characters sent. Because of the connectivity of $G \setminus (A \times P)$ this realizes the fail-stop property according to Lemma 3.1 (with $G = G^*$).

In the rest of Section 3.2.2 an arbitrary, but fixed, key pair (K_{ij}, K_{ji}) with $P_i \notin A$ and $P_j \notin A$ is considered. Therefore indices i, j are often omitted.

The most powerful attacker is assumed: he is able to observe the values of K_{ij}^t and K_{ji}^t for each round t directly and he can deliver arbitrary input characters I_i^t and I_j^t to P_i and P_j. Participants P_i and P_j are assumed to by unsynchronized, hence the attacker can wait for K_{ij}^{t+1} before he delivers I_j^t to P_j.

Let $(F, +, \bullet)$ be a finite field and let $a^1, a^2, \ldots, a^{t_{max}}$ and $b^1, b^2, \ldots, b^{t_{max}-1}$ be two sequences whose elements are randomly selected from F and privately exchanged by P_i and P_j. Define for $t = 1, \ldots, t_{max}$

$$\begin{aligned} K_{ij}^t &:= a^t + \sum_{k=1}^{t-1} b^{t-k} \bullet I_i^k \\ K_{ji}^t &:= a^t + \sum_{k=1}^{t-1} b^{t-k} \bullet I_j^k \end{aligned} \tag{3.1}$$

Lemma 3.2 The key generation scheme defined by Equation (3.1) satisfies the two conditions SS and FS formulated above.

Proof. Since $a^t \in_R F$, and since $\Sigma := K_{ij}^t - a^t$ is independent of a^t, $K_{ij}^t \in_R F$.

Assume $\varepsilon_{ij}^s = 0$ for all $s < t$. Then obviously $\delta_{ij}^t = 0$ and condition SS is satisfied.

Now assume that s is the first round with $\varepsilon_{ij}^s \neq 0$. For simplicity let $\varepsilon^u := \varepsilon_{ij}^u$ and $\delta^u := \delta_{ij}^u$. The differences δ^u are formed according to the following system of linear equations:

$$\delta^u = 0 \text{ for } u = 1, \ldots, s$$

$$\begin{pmatrix} \delta^{s+1} \\ \delta^{s+2} \\ \ldots \\ \delta^{t-1} \\ \delta^t \end{pmatrix} = \begin{pmatrix} \varepsilon^s & 0 & \ldots & 0 & 0 \\ \varepsilon^{s+1} & \varepsilon^s & \ldots & 0 & 0 \\ \ldots & \ldots & \ldots & \ldots & \ldots \\ \varepsilon^{t-2} & \varepsilon^{t-3} & \ldots & \varepsilon^s & 0 \\ \varepsilon^{t-1} & \varepsilon^{t-2} & \ldots & \varepsilon^{s+1} & \varepsilon^s \end{pmatrix} \bullet \begin{pmatrix} b^1 \\ b^2 \\ \ldots \\ b^{t-s-1} \\ b^{t-s} \end{pmatrix}$$

Since $\varepsilon^s \neq 0$, the matrix is regular and defines a bijective mapping. Since all $b^u \in_R F$, all δ^u are uniformly and independently distributed in F. The independence of all $K_{ij}^1, \ldots, K_{ij}^t$ and $\delta^{s+1}, \ldots, \delta^t$ follows from the independence of all $a^1, \ldots, a^t$ and $\delta^{s+1}, \ldots, \delta^t$. □

The additional expenditure of this key generation scheme is given by

- the $2 \bullet t_{max} - 1$ privately exchanged keys a^t, b^t for each pair P_i, P_j directly connected in G (instead of only t_{max} for pure superposed sending),
- the storage of all t_{max}-1 received input characters, and
- the (t-1) field additions and multiplications for computing the key for round t.

From the last fact it follows that the scheme requires an average of $t_{max} / 2$ field additions and multiplications per round. Hence the scheme does not seem to be very practical.

Given the assumption that there is no additional communication between P_i and P_j about their current states the scheme is *optimal* with respect to the number of exchanged keys and additional storage requirements.

Lemma 3.3 The key generation scheme defined by Equation (3.1) is optimal with respect to the number of exchanged keys and additional storage requirements, i.e. each key generation scheme which *deterministically* satisfies conditions SS and FS requires at least

- the storage of all t_{max} - 1 received input characters and
- $2 \bullet t_{max}$ - 1 privately exchanged keys.

Proof. The first limit is obvious: the scheme has to distinguish between all possible sequences of t_{max} input characters, hence all input characters have to be stored.

For proving the second limit, let Z be the secret key shared by P_i and P_j and used for generating the keys $K_{ij}{}^t$, $K_{ji}{}^t$, $t = 1, \ldots, t_{max}$. Let $H(I_i)$, $H(I_j)$, $H(K_{ij}{}^1)$, $H(K_{ij}{}^{(2,t_{max})})$, $H(K_{ji}{}^{(2,t_{max})})$, H(Z) be the entropy of the random variables $I_i = (I_i{}^1, \ldots, I_i{}^{t_{max}})$, $I_j = (I_j{}^1, \ldots, I_j{}^{t_{max}})$, $K_{ij}{}^1$ $(=K_{ji}{}^1)$, $K_{ij}{}^{(2,t_{max})} = (K_{ij}{}^2, \ldots, K_{ij}{}^{t_{max}})$, $K_{ji}{}^{(2,t_{max})} = (K_{ji}{}^2, \ldots, K_{ji}{}^{t_{max}})$, and Z, respectively [Gall_68, Chapter 2].

By applying standard rules of information theory

$$H(K_{ij}{}^1 K_{ij}{}^{(2,t_{max})} K_{ji}{}^{(2,t_{max})} \mid I_i I_j) \leq H(Z K_{ij}{}^1 K_{ij}{}^{(2,t_{max})} K_{ji}{}^{(2,t_{max})} \mid I_i I_j)$$
$$= H(Z \mid I_i I_j) + H(K_{ij}{}^1 K_{ij}{}^{(2,t_{max})} K_{ji}{}^{(2,t_{max})} \mid Z I_i I_j)$$

Since Z is chosen independently of the attacker's input characters $H(Z \mid I_i I_j) = H(Z)$, and since the keys are completely determined by Z and I_i, I_j, $H(K_{ij}{}^1 K_{ij}{}^{(2,t_{max})} K_{ji}{}^{(2,t_{max})} \mid Z I_i I_j) = 0$.

Hence it follows

$$H(Z) \geq H(K_{ij}{}^1 K_{ij}{}^{(2,t_{max})} K_{ji}{}^{(2,t_{max})} \mid I_i I_j)$$

Since only a lower bound is proved, it can be assumed that the attacker chooses $I_i{}^1$ and $I_j{}^1$ differently. Then the keys $K_{ij}{}^1$, and $K_{ij}{}^t$, $K_{ji}{}^s$ for $t, s = 2, \ldots, t_{max}$ are independently chosen, i.e.

$$H(K_{ij}{}^1 K_{ij}{}^{(2,t_{max})} K_{ji}{}^{(2,t_{max})} \mid I_i I_j) = H(K_{ij}{}^1 K_{ij}{}^{(2,t_{max})} K_{ji}{}^{(2,t_{max})})$$
$$= H(K_{ij}{}^1) + H(K_{ij}{}^{(2,t_{max})}) + H(K_{ji}{}^{(2,t_{max})})$$

Hence

$$H(Z) \geq H(K_{ij}{}^1) + H(K_{ij}{}^{(2,t_{max})}) + H(K_{ji}{}^{(2,t_{max})})$$

i.e. Z must consist of at least $1 + (t_{max}\text{-}1) + (t_{max}\text{-}1) = 2 \bullet t_{max}$ - 1 keys. □

3.2.2.2 Probabilistic fail-stop key generation

To get a more efficient key generation scheme, it seems necessary to switch to a probabilistic version of FS: For a given fail-stop mechanism, let $Prob_A$ be the attacker's **probability of success**. The attacker is successful if, in spite of choosing $I_i{}^s \neq I_j{}^s$ for a $s < t_{max}$, there exists an index t, $s < t \leq t_{max}$, such that the global sum S^t and the message characters $M_i{}^t$, $i = 1, \ldots, n$, are *not* independent.

For each $d \in \mathbb{N}$ define

FS_d If two honest participants receive two different input characters in round t, they will disturb superposed sending for the following d rounds.

The maximum number d for which FS_d is satisfied is a random variable with probability distribution Prob(d).

Let $a^1, a^2, \ldots, a^{t_{max}}, b^3, b^4, \ldots, b^{t_{max}}, e$ be randomly and privately selected elements of the finite field F. Let $b^1 = b^2 = 0$ and let $K_{ij}{}^0 = K_{ji}{}^0 = 0$ and $I_i{}^0 = I_j{}^0 = 0$. Then define for $t = 1, \ldots, t_{max}$

$$K_{ij}{}^t := a^t + b^t \bullet K_{ij}{}^{t\text{-}1} + e \bullet I_i{}^{t\text{-}1} \qquad (3.2)$$

$$K_{ji}{}^t := a^t + b^t \bullet K_{ji}{}^{t-1} + e \bullet I_j{}^{t-1}$$

Lemma 3.4 The key generation scheme defined by Equation (3.2) satisfies condition SS. The maximum number d for which FS_d is satisfied is a geometrically distributed random variable:

$$\text{Prob}(d) = \frac{1}{|F|} \bullet \left(1 - \frac{1}{|F|}\right)^{d-1}$$

The attacker's probability of success is

$$Prob_A \leq 1 - \left(1 - \frac{1}{|F|}\right)^{t_{max}}$$

Proof. Since $a^t \in_R F$, and since $\Sigma := K_{ij}{}^t - a^t$ is independent of a^t, $K_{ij}{}^t \in_R F$.

Assume $\varepsilon_{ij}{}^s = 0$ for all $s < t$. Then obviously $\delta_{ij}{}^t = 0$ and condition SS is satisfied.

Now assume that s is the first round with $\varepsilon_{ij}{}^s \neq 0$. For simplicity let $\varepsilon^v := \varepsilon_{ij}{}^v$ and $\delta^v := \delta_{ij}{}^v$.

In the next round $\delta^{s+1} = e \bullet \varepsilon^s$. Since $\delta^v = 0$ for all $v \leq s$ the attacker has no information about the actual value of e before round s+1. By assumption $\varepsilon^s \neq 0$, hence δ^{s+1} is uniformly distributed in F.

Now consider the rounds $s + u + 1$ with $u \geq 1$. If $\delta^{s+u} = 0$, then $\delta^{s+u+1} = e \bullet \varepsilon^{s+u}$. From round s+1 the attacker knows the value of e, hence δ^{s+u+1} is *not* independently distributed in F. If $\delta^{s+u} \neq 0$, then $\delta^{s+u+1} = b^{s+u+1} \bullet \delta^{s+u} + e \bullet \varepsilon^{s+u}$. Since b^{s+u+1} is uniformly distributed in F, δ^{s+u+1} is uniformly distributed, too, and since b^{s+u+1} is only used in that round, δ^{s+u+1} is independent of all other δs.

Therefore the actual value of d is given by the lowest value $d \geq 1$ for which $\delta^{s+d} = 0$. Since δ^{s+1} is uniformly distributed,

$$\text{Prob}(\delta^{s+1} \neq 0) = 1 - \frac{1}{|F|}$$

and since for $\delta^{s+d} \neq 0$, δ^{s+d+1} is uniformly distributed,

$$\text{Prob}(\delta^{s+d+1} \neq 0 \mid \delta^{s+d} \neq 0) = 1 - \frac{1}{|F|}$$

From this it follows

$$\text{Prob}(d) = \frac{1}{|F|} \bullet \left(1 - \frac{1}{|F|}\right)^{d-1}$$

The independence of all $K_{ij}{}^1, \ldots, K_{ij}{}^t$ and $\delta^{s+1}, \ldots, \delta^d$ follows from the independence of all $a^1, \ldots, a^t$ and $\delta^{s+1}, \ldots, \delta^d$.

The probability of success is simply the probability that $s + d \leq t_{max}$:

$$Prob_A = \text{Prob}(d \leq t_{max} - s)$$

Since $s \geq 0$,

$$Prob_A \leq \text{Prob}(d \leq t_{max}) = 1 - \text{Prob}(d > t_{max}) = 1 - \left(1 - \frac{1}{|F|}\right)^{t_{max}}$$

□

Since d is geometrically distributed, the average value of d is $|F|$ [Triv_82 p. 579]. Hence $|F|$ must be chosen considerably larger than t_{max}.

Corollary. Assume the key generation scheme of Eq. (3.2). Then

$$Prob_A \leq 1 - \left(\frac{1}{4}\right)^{t_{max} / |F|}$$

Proof. From Lemma 3.4 it follows

$$Prob_A \leq 1 - \left(1 - \frac{1}{|F|}\right)^{t_{max}} = 1 - \left(1 - \frac{1}{|F|}\right)^{|F| \bullet t_{max} / |F|}$$

The sequence $(1 - \frac{1}{x})^x$ increases monotonously. Since $|F| \geq 2$

$$Prob_A \leq 1 - (\tfrac{1}{4})^{t_{max} / |F|}$$

□

Obviously with a decreasing value of $t_{max} / |F|$ the probability $Prob_A$ vanishes. From the corollary it follows for each $0 \leq L < 1$

$$\frac{t_{max}}{|F|} \leq \frac{1}{2} \bullet \mathrm{ld}(\frac{1}{1-L}) \;\Rightarrow\; Prob_A \leq L$$

E.g. for $L = 10^{-9}$

$$\frac{t_{max}}{|F|} \leq 7 \bullet 10^{-10}$$

is sufficient, which is satisfied e.g. by $|F| = 2^{108}$ and $t_{max} = 10^{23}$. These values allow the transmission of

$$t_{max} \bullet \mathrm{ld}(|F|) = 10^{23} \bullet 108 \text{ bits} \approx 10^{25} \text{ bits}$$

For a transmission speed of 10^{15} bits/s (which is far beyond today's technology) this would be sufficient for about 317 years.

The key generation of Eq. (3.2) requires as many privately exchanged keys as the scheme defined by Eq. (3.1), i.e. $2 \bullet t_{max} - 1$.

To evaluate Eq. (3.2) for round t, it is only necessary to store the last key, K_{ij}^{t-1} (in contrast to the last t-1 keys for Eq. (3.1)) and to perform 2 field additions and multiplications. In contrast to the scheme of Eq. (3.1), only large fields are suitable.

3.2.2.3 Combination of key generation and explicit tests

If the multi-access protocol guarantees that for some slots only one participant is allowed to choose a nonzero message, this participant can test the network:

Assume that superposed sending is stopped after a broadcast inconsistency by one of the key generation schemes described above, i.e. the global sums are randomly distributed. Then each participant P_i who is allowed to use a slot exclusively, and sends a message randomly selected from F^c, will receive a wrong message with probability $1 - |F|^{-c}$. Thus he detects the disturbance with the same probability and can explicitly stop superposed sending by choosing his following output characters randomly from F instead of according to Eq. (2.1).

If it is guaranteed that each participant sends a test message within a fixed number s of slots, and if there are at least two honest participants, this makes it unnecessary to consider more than the last $(s-1) \bullet c$ input characters for key generation: after s-1 slots, superposed sending will be explicitly disturbed with high probability by some honest participant who received a disturbed test message instead of the one he sent.

The required fairness of the multi-access protocol can deterministically be satisfied by superposed receiving and in a probabilistic sense by each reservation technique (Sect. 2.2). If e.g. each participant reserves exactly one test message and at the most one real message in each reservation phase, each participant tests the network within $s = 4 \bullet n$ slots.

Obviously the fairness of this can only be guaranteed if all participants behave fairly, i.e. each unfair (and therefore dishonest) participant can prevent some honest participants from successfully doing their required reservation. Therefore each honest participant who cannot send a message within s slots should disturb superposed sending.

The additional rules do not help the attacker: Assume that an honest participant P_i detects a disturbance, i.e. $I_i^t \neq M_i^t$, and stops sending. Nevertheless the attacker is not able to observe the sending of P_i.

If the disturbance detected by P_i was a consequence of a previous broadcast inconsistency, the sending was stopped anyway, hence there is nothing to show. Otherwise, and if all honest participants receive the same input character, the unobservability of P_i follows from Lemma 2.2, and if the attacker manipulates the broadcast property for round t, sending is stopped by the key generation scheme anyway, independent of P_i's test.

Proper modifications to the key generation schemes will be discussed in the following two sections.

The advantages and disadvantages of the combination are the same in both schemes:

- For key generation, the parameter t_{max} is replaced by $(s\text{-}1)\bullet c$, which decreases the number of additional secret keys from t_{max} to $(s\text{-}1)\bullet c$, and for deterministic key generation the computation complexity from $O(t_{max}{}^2)$ to $O(s^2\bullet c^2)$ operations and from $O(t_{max})$ to $O(s\bullet c)$ required storage.
- Some honest participants may be forced to send meaningless test messages, thus the throughput of the DC-net is decreased. The number of additional test messages depends on the participants' sending rates.

3.2.2.3.1 Combination of deterministic key generation and explicit tests

Assume that the deterministic scheme of Eq. (3.1) is used in combination with explicit tests.

If round u is the first disturbed round, the attacker has no information about the privately exchanged keys b^v, $v = 1, \ldots, u$. After round $u + (s\text{-}1) \bullet c$, it is highly probable that the DC$^+$-net will be disturbed by at least one honest participant who has detected the disturbance. Hence instead of t_{max} - 1 additional keys, a maximum of $(s\text{-}1) \bullet c$ are really necessary:

$$K_{ij}{}^t := a^t + \sum_{k=t-(s-1)\bullet c}^{t-1} b^{\,t-k} \bullet I_i{}^k$$
$$K_{ji}{}^t := a^t + \sum_{k=t-(s-1)\bullet c}^{t-1} b^{\,t-k} \bullet I_j{}^k \qquad (3.3)$$

Lemma 3.5 The key generation scheme defined by Equation (3.3) satisfies condition SS. Together with the additional rules for testing and disturbing it ensures the fail-stop property in a probabilistic sense: Let h be the number of honest participants, $h \geq 2$. Then

$$Prob_A \leq \frac{1}{|F|^{c\bullet(h-1)}}$$

Proof. Since $a^t \in_R F$, and since $\Sigma := K_{ij}{}^t - a^t$ is independent of a^t, $K_{ij}{}^t$ is uniformly distributed in F. Assume $\varepsilon_{ij}{}^u = 0$ for all $u < t$. Then obviously $\delta_{ij}{}^t = 0$ and condition SS is satisfied.

Now assume that u is the first round with $\varepsilon_{ij}{}^u \neq 0$. According to Lemma 3.3 (with $t_{max} = (s\text{-}1)\bullet c\text{-}1$) the global sums of the following $(s\text{-}1)\bullet c\text{-}1$ rounds are all randomly chosen from F. Since it is assumed that during the s slots each participant tests the network, the attacker's only chance is that during the first s-1 slots none of the at least h-1 honest participants detects the disturbance. The probability that a single test doesn't detect a disturbance is $|F|^{-c}$, hence the attacker's probability is less than $|F|^{-c\bullet(h-1)}$. □

The scheme requires only $(s\text{-}1) \bullet c$ additional keys instead of the t_{max}-1 of the key generation scheme of Section 3.2.2.1.

The number of field operations per round is in the order of $(s\text{-}1) \bullet c - 1$. To avoid unnecessarily expensive field computations, $F = \mathrm{GF}(2)$ should be chosen. Here, with $h \geq 2$, $Prob_A \leq 1 / 2^c$.

Since each of the n participants should send a message within s slots, s should be in the order of n. In this case, the scheme requires $\mathrm{O}(n \bullet c)$ operations. For $F = \mathrm{GF}(2)$ and therefore $c \approx -\log(Prob_A)$ this is equal to $\mathrm{O}(n \bullet -\log(Prob_A))$.

3.2.2.3.2 Combination of probabilistic key generation and explicit tests

Assume that the probabilistic scheme of Eq. (3.2) is used in combination with explicit tests.

By the same argumentation as above it follows that instead of $t_{max} - 1$ additional keys, a maximum of $(s\text{-}1) \bullet c$ are really necessary, i.e. it is possible to use the $(s\text{-}1) \bullet c$ keys $b^0, \ldots, b^{(s-1)\bullet c-1}$ cyclically:

Let $a^1, \ldots, a^{t_{max}}, e, b^0, \ldots, b^{(s-1)\bullet c-1}$ be randomly chosen keys. Then

$$\begin{aligned} K_{ij}{}^t &:= a^t + b^{t \bmod (s-1)\bullet c} \bullet K_{ij}{}^{t-1} + e \bullet I_i{}^{t-1} \\ K_{ji}{}^t &:= a^t + b^{t \bmod (s-1)\bullet c} \bullet K_{ji}{}^{t-1} + e \bullet I_j{}^{t-1} \end{aligned} \tag{3.4}$$

Lemma 3.6 The key generation scheme defined by Equation (3.4) satisfies condition SS. Together with the additional rules for testing and disturbing it ensures the fail-stop property in a probabilistic sense:

$$Prob_A \leq 1 - \left(1 - \frac{1}{|F|}\right)^{(s-1)\bullet c}$$

Proof. The first part is proved as in Lemma 3.4. The worst case for the second part, i.e. the best case for an attacker, is that out of all testing participants only the last two are honest. Then the attacker is unsuccessful iff the actual value of d (defined as for Eq. (3.2)) is greater than $(s\text{-}2) \bullet c$, and the test detects the disturbance. Hence

$$\begin{aligned} Prob_A &\leq 1 - \sum_{j=1}^{c-1} \left(\mathrm{Prob}(d=(s-2)\bullet c+j) \bullet \left(1 - \frac{1}{|F|^j}\right)\right) - \mathrm{Prob}(d \geq (s-1)\bullet c) \bullet \left(1 - \frac{1}{|F|^c}\right) \\ &\leq 1 - \mathrm{Prob}(d \geq (s-1)\bullet c) \bullet \left(1 - \frac{1}{|F|^c}\right) \\ &= 1 - \left(1 - \frac{1}{|F|}\right)^{(s-1)\bullet c-1} \bullet \left(1 - \frac{1}{|F|^c}\right) \\ &\leq 1 - \left(1 - \frac{1}{|F|}\right)^{(s-1)\bullet c} \end{aligned}$$

□

Again only large fields F (e.g. $\mathrm{ld}(|F|) \approx 150$) are suitable.

4 Final remarks

Superposed sending together with one of the discussed fail-stop key generation schemes (Sect. 3.2.2) guarantees the desired unconditional sender and recipient untraceability.

If one tries to transform this nice theoretical result into a real communication network, a lot of practical problems must be solved, but none of them becomes really harder if fail-stop broadcast is used in addition to normal superposed sending.

For this, consider the performance of superposed sending measured by

- the number of exchanged keys per message transmitted,
- its communication complexity,
- its computational complexity,
- and the reliability of the scheme.

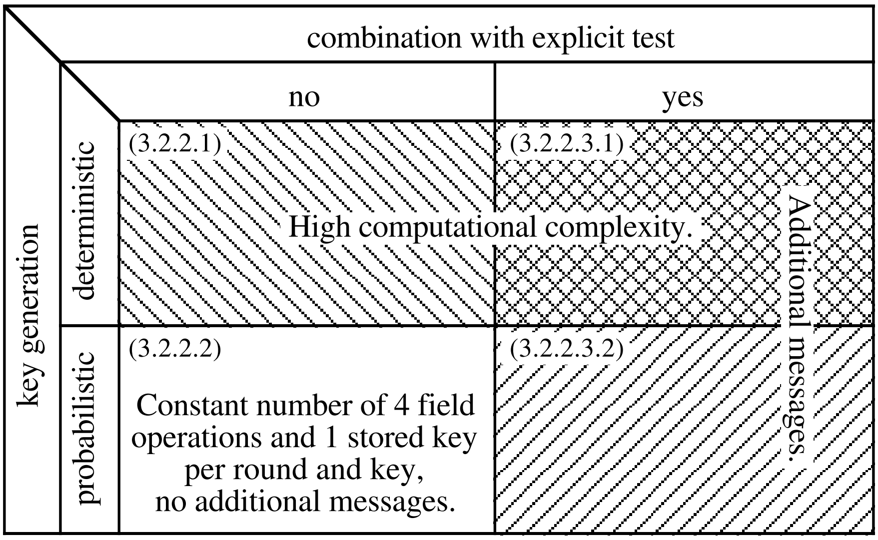


Figure 2 Comparison of fail-stop key generation schemes.

The **number of additional keys** is increased by a factor of two at the most. This was shown to be the optimal value for deterministic key generation schemes without explicit tests. In theory this seems to be acceptable, and in practice pseudorandomly generated keys one will mostly be chosen anyway (and as a result of this, unconditional untraceability will be lost).

Communication complexity (Fig. 2)**.** None of the pure key generation schemes (Sect. 3.2.2.1, 3.2.2.2) requires additional messages to be sent.

If combinations of key generation and explicit tests (Sect. 3.2.2.3) are used, some honest participants may be forced to send meaningless test messages. The number of additional test messages depends on the participants' sending and testing rates. If real messages are encrypted end-to-end, they appear to be randomly selected from F^c, i.e. they can be used instead of explicit test messages.

Computational complexity (Fig. 2)**.** The key generation requires some additional time and memory for each exchanged key. For that reason the schemes with deterministic key generation (Sect. 3.2.2.1, 3.2.2.3.1) seem to be less practical, but if one uses one of the schemes with probabilistic key generation (Sect. 3.2.2.2, 3.2.2.3.2), the computation requires only the storage of the last key and two field additions and multiplications per round and exchanged key.

All schemes except that of Sect. 3.2.2.1 realize *probabilistic* untraceability only, i.e. there is a small probability that an attacker will successfully transmit different messages to different participants. But all four schemes don't rely on any unproved assumptions.

For probabilistic key generation (Sect. 3.2.2.2, 3.2.2.3.2) only large fields F are suitable, but this is no hard restriction:

- Usually the cardinality $|F|^c$ of the set of all transmission units "message" will be relatively large. It doesn't matter whether one uses a small field and a large c or a large field and a small c.
- The reservation map technique and (reservation by) superposed receiving (Sect. 2.2) require a large cyclic group $(F,\oplus)$, anyway. It is important to notice that the group $(F,\oplus)$ used for superposed

sending need not be the additive (or multiplicative) group of the finite field (F,+,•) used for key generation. E.g. one can use the field F = GF(2^m) for key generation and, by interpreting the elements of GF(2^m) as binary encoded integers, the additive group of integers modulo 2^m for superposed sending.

The **transmission delay** introduced by key generation could be decreased by parallelizing the key generation for different rounds. This can be done in two ways.

One can use $k > 1$ DC^+-nets, say DC^+_0, …, DC^+_{k-1}, in a time division technique, i.e. in round t the DC^+-net $DC^+_{t \bmod k}$ is used. To preserve untraceability, each interaction between participants should be completely performed using only a single DC^+-net, i.e. each participant should answer a message only by that DC^+-net by which he has received the message.

The other possibility is to use just one DC^+-net, but to make the keys for round t dependent not on the directly preceding rounds t-i, i = 1, 2, …, t-1, but on the rounds t-i, i = k, k+1, …, t-1 for a $k > 1$. To preserve untraceability, each participant has to wait at least k-1 rounds before he answers to a received character.

Naturally the fail-stop property decreases the **reliability** of the network, since every inconsistent broadcast will immediately stop the network independent of whether it was caused by an attacker or a physical fault. But most transient faults in a network can be tolerated by usual data link protocols [Tane_88 Chapter 4], and if a permanent fault occurs (e.g. if a participant's station is damaged or all links between two participants are cut), superposed sending is disturbed and the network is stopped anyway. Therefore reliability is not essentially reduced by the discussed fail-stop schemes.

The problem of combining untraceability and **serviceability** in spite of active attacks is discussed in [WaPf_89, WaPf1_89].

Hence, the pure probabilistic key generation scheme (Sect. 3.2.2.2) with an appropriately large field F seems to be the most practical choice.

Acknowledgements

I'm pleased to thank Birgit Baum-Waidner and Andreas Pfitzmann for lots of stimulating discussions and Birgit Pfitzmann for her efficacious support, and I'm grateful to Manfred Böttger, Klaus Echtle, and Tina Johnson for many valuable suggestions, and the German Science Foundation (DFG) for financial support.

The results presented here are also contained in [WaPf_89], which was written in cooperation with Birgit Pfitzmann.

References

BeGW_88 M. Ben-Or, S. Goldwasser, A. Wigderson: Completeness theorems for non-cryptographic fault-tolerant distributed computation; 20th STOC, ACM, New York 1988, 1-10.

BoBo_89 J. Bos, B. den Boer: Detection of Disrupters in the DC Protocol; Abstracts of Eurocrypt '89; final version will be published in: Eurocrypt '89, LNCS, Springer-Verlag, Berlin 1989.

Chau_81 D. Chaum: Untraceable Electronic Mail, Return Addresses, and Digital Pseudonyms; Communications of the ACM 24/2 (1981) 84-88.

Cha8_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; J. of Cryptology 1/1 (1988) 65-75 (draft received May 13, 1985).

Chau1_89 D. Chaum: The Spymasters Double-Agent Problem – Multiparty Computations Secure Unconditionally from Minorities and Cryptographically from Majorities; presented at Crypto '89 (draft received July 21, 1989); final version will be published in: Crypto '89, LNCS, Springer-Verlag, Berlin 1990.

ChCD1_88 D. Chaum, C. Crépeau, I. Damgård: Multiparty unconditional secure protocols; 20th Symp. on Theory of Computing, ACM, New York 1988, 11-19.

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

Gall_68 R. G. Gallager: Information Theory and Reliable Communication; John Wiley & Sons, New York 1968.

GiMS_74 E. N. Gilbert, F. J. Mac Williams, N. J. A. Sloane: Codes which detect deception; Bell System Technical J. 53/3 (1974) 405-424.

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

LaSP_82 L. Lamport, R. Shostak, M. Pease: The Byzantine Generals Problem; ACM Trans. on Programming Languages and Systems 4/3 (1982) 382-401.

Marc_88 E. Marchel: Leistungsbewertung von überlagerndem Empfangen bei Mehrfachzugriffsverfahren mittels Kollisionsauflösung; Diplomarbeit am Institut für Rechnerentwurf und Fehlertoleranz, Universität Karlsruhe 1988.

Mass_81 J. L. Massey: Collision-Resolution Algorithms and Random-Access Communications; Multi-User Communication Systems; Edited by G. Longo; CISM Courses and Lectures No. 265, International Centre for Mechanical Sciences; Springer-Verlag, Wien 1981, 73-137.

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

Pfit_84 A. Pfitzmann: A switched/broadcast ISDN to decrease user observability; 1984 International Zürich Seminar on Digital Communications, IEEE, 1984, 183-190.

Pfi1_85 A. Pfitzmann: How to implement ISDNs without user observability - Some remarks; Fakultät für Informatik, Interner Bericht 14/85, Universität Karlsruhe 1986.

Pfit_87 B. Pfitzmann: private communication, 1987.

Pfit_89 A. Pfitzmann: Diensteintegrierende Kommunikationsnetze mit Teilnehmer-überprüfbarem Datenschutz; Dissertation Fakultät für Informatik, Universität Karlsruhe 1989 (will be published in: IFB, Springer-Verlag, Berlin 1989).

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

PfWa_86 A. Pfitzmann, M. Waidner: Networks without user observability -- design options; Eurocrypt '85, LNCS 219, Springer-Verlag, Berlin 1986, 245-253; revised version: Computers & Security 6/2 (1987) 158-166.

Rab1_80 M. O. Rabin: Probabilistic Algorithms in Finite Fields; SIAM J. Comput. 9/2 (1980) 273-280.

Sim3_88 G.J. Simmons: A Survey of Information Authentication; Proc. IEEE 76/5 (1988) 603-620.

Tane_88 A. S. Tanenbaum: Computer Networks; 2nd ed., Prentice-Hall, Englewood Cliffs 1988.

Triv_82 K. S. Trivedi: Probability and Statistics with Reliability, Queuing, and Computer Science Applications; Prentice-Hall, Englewood Cliffs 1982.

VaVa_85 U. V. Vazirani, V. V. Vazirani: Efficient and Secure Pseudo-Random Number Generation (extended abstract); Crypto '84, LNCS 196, Springer-Verlag, Berlin 1985, 193-202.

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

WaPf1_89 M. Waidner, B. Pfitzmann: The Dining Cryptographers in the Disco: Unconditional Sender and Recipient Untraceability with Computationally Secure Serviceability; presented at Eurocrypt '89; Fakultät für Informatik, Universität Karlsruhe 1989.