Secure communication with secret sharing
in static computer networks with partition in mistrust parties

Johannes Viehmann
Fraunhofer Institut FOKUS (MOTION)
Kaiserin-Augusta-Allee 31
D-10589 Berlin, Germany
Johannes.Viehmann@Fokus.Fraunhofer.de

Abstract — This paper presents a solution to ensure arbitrarily secure communication in a large computer network by using secret sharing and multiple parties mistrusting each other instead of relying on some “trusted party” or a “web of trust”. In contrast to other solutions that use a PKI and require asymmetric encryption, this concept can guarantee to provide secure communication even after any possible advance in cryptanalysis and even if unlimited calculation power was available to attack it. But this solution requires the computer network to have special properties. It is mainly intended to be used in the S-Network, a repository for reliable publications.

Secret sharing; trust; mistrust; secure communication; PSMT

I. BASIC NOTATION AND BASE TECHNIQUES

Let x and y be bit sequences. The concatenation of x and y prefixed with their identifiers and lengths is noted as x◦y.

The symmetric encryption of a bit sequence x with key K is noted as EK(x). The corresponding decryption is written as DK(EK(x)). Let P(x) be a function calculating a message authentication code (MAC) of a bit sequence x.

K, EK(x), DK(EK(x)), P(x) and x◦y are bit sequences. Messages are bit sequences, too.

A secure channel between Alice and Bob is a communication channel which allows exchanging messages between Alice and Bob in a finite time so that the secrecy, integrity and authenticity of the messages can be ensured and that the temporal order in which the delivered messages were sent by Alice can be reconstructed by Bob. There are provable secure solutions to keep perfect secrecy [14], but integrity, authenticity and the order can only be ensured with arbitrary high probability: Bit sequences passing tests for these could be guessed.

Secret sharing is a technology to split a secret x into a set of n pieces with the property that you need at least t pieces of the set to be able to reconstruct x from that subset. Any subset with less than the threshold t pieces does not reveal any information about x at all. There are several perfectly secure secret sharing systems known, for an example see [13]. The following notation will be used for the set of pieces of a secret sharing split:

\[ Z_n(x) = \{ T_{n,0}(x), \ldots, T_{n,n-1}(x) \} \]

The inverse operation will be noted as:

\[ x = Z_n^{-1}(M) \mid M \subseteq Z_n(x) \land \#M \geq t \]

The concept introduced in this paper makes use of security technologies like secret sharing that do have a threshold up to which they are secure. To describe a unique security level for the entire system, a security constant \( \Psi \) is defined. \( \Psi \) is a natural number and it must be greater than two.

II. THE PROBLEM

A. Computer networks with strong security requirements

To enable secure communication in a computer network, any two participants should be able to establish a secure channel with each other. The required level of security may vary from application to application. Sometimes, long term security has to be guaranteed, which means, that the cryptographic concept should be secure and practically usable for the future – independent from any possible further technical development. For the following, high long term security requirements are assumed.

B. The difficulty to provide secure channels in big networks

It is possible to build an arbitrarily secure channel between any two participants Alice and Bob which has these strong long term security properties, but that requires that Alice and Bob share an exclusive secret in advance. Alice and Bob have to check their identities and exchange the secret manually.

In a very small network with only a few participants, it is possible to do such a manual procedure for all possible pairs of participants. But the effort grows quadratically with the number of participants. With several thousand or with several million participants, this is definitely not manageable.

III. STATE OF THE ART

A. Secure communication relying on a “trusted party”

“Trusted parties”, sometimes called “trusted third parties”, too, can be used to provide secure communication between any two participants in computer networks. The idea is, that all participants identify themselves only to the “trusted party”. For the further usage of the “trusted party” there are different concepts:

1) Inline usage of a “trusted party”: Each participant shares an exclusive key with the “trusted party” so that a secure channel can be built between each participant and the “trusted party”. Messages between two simple participants Alice
and Bob are first send from Alice to the “trusted party” over a secure channel and then the “trusted party” forwards them to Bob over another secure channel. All messages have to pass the “trusted party”, which makes it likely that the central “trusted party” becomes a bottleneck.

2) **Online usage** of a “trusted party” as key server: The “trusted party” generates a session key for Alice and Bob so that they can build a direct secure channel between them. See [10] for a solution with this approach. With keys of constant length, this approach reduces the workload of the “trusted party”.

3) At least somehow **offline usage** of a “trusted party” or of a hierarchy of several “trusted parties” as certification authority (CA): See [7] for a description of such a public key infrastructure (PKI) and a discussion of the advantages in comparison with a key server. The most important difference for the strong long term security requirement is however, that asymmetric encryption (for example [12]) is required for this solution.

It is possible that in the future all potential asymmetric algorithms can be broken in a relevant short time. For algorithms whose security depends on the assumed difficulty of calculating discrete logarithms or to do prime factorization for large numbers, a theoretical solution for breaking them with quantum computers in polynomial time has already been shown in [15]. For the prime factorization of small numbers, it has been demonstrated that the Shor algorithm really works [9].

If potentially insecure functions are used for creating signatures on certificates, this may be another potential point to attack a PKI. See [16] for an attack that takes advantage of the MD5 cryptographic hash function that has been widely used on certificates, but which is not collision resistant.

Public keys are typically used to encrypt and exchange symmetric session keys so that messages can then be exchanged with a more efficient symmetric cypher like AES [5]. This is called hybrid encryption. However, if the security of the symmetric cypher used for hybrid encryption might eventually be broken, this opens another point to attack. Recent advances in cryptanalysis [3] show that this threat should be taken serious.

So in a typical PKI, there are at least three different potentially insecure algorithms that can be attacked independently. It is enough to break just one of these potential weaknesses to break the entire system’s security.

No matter how “trusted parties” are used – the security of the communication depends on the fair and always correct behavior of the “trusted party”. Why should participants trust the “trusted party”? To control institutions that have so much power is difficult and maybe it is utopian or naive to believe that universal neutrality can at all be enforced in a big network that really matters.

**B. Secure communication relying on a "web of trust"**

To avoid the need to trust in some single party, the “web of trust” offers a decentralized alternative concept [4]. However, with this approach, it is not possible to achieve legal validity and it requires asymmetric encryption, too. Furthermore, the demands for the users are high as they have to decide whom to trust.

In general, it has also to be questioned whether trust is transitive at all.

**C. Secure communication with secret sharing**

Secret sharing can be used to avoid the need to trust a single party, too, by dividing the responsibility for trust related things between several parties. A typical application of secret sharing is to store a secret, for example a secret key.

But it is also possible to use secret sharing for “perfectly secure message transmission” (PSMT) over disjoint paths as shown in [6] (see also [8] and [11]). These solutions require a set of completely separated communication channels (called “wires”) between sender and addressee. But how these disjoint “wires” could be realized is not mentioned, neither how the identities could be checked nor how authentication could work.

In [1] a dynamic method to find separate wires is presented, but it provides only paths with disjunct edges, not with disjoint nodes. Therefore, it is not a solution for PSMT.

**IV. SOLUTION WITH SECRET SHARING AND MISTRUST**

**A. A static network with partition in mistrust parties**

The concept for secure communication presented in this paper requires an applicable legal framework and it requires the computer network in which the secure communication takes place to have the following properties:

The logical addresses of the logical systems within the network must be everlasting, absolute and unique. In the following, such an uniquely addressable logical system will be called an **S-Node**.

S-Nodes added to the network have to be kept accessible by their logical addresses. If an **S-Node** is not accessible because of some failure, it has to be repaired and restored within a finite time. Such a network may be called a static network.

For each **S-Node** there must be exactly one natural or juristic person responsible for it in a legal sense: the **S-Operator**. Each **S-Node** does also have an **owner**. If the **S-Operator** is not also the owner of his **S-Node** but only the administrator, the **S-Operator** must have a contract with the owner.

Let X be the set of all **S-Operators** in a static network. A **partition** of such a static network is the split of X into not empty disjunct subsets so that the union of all subsets is X. The subsets of a partition of a static network are called **parties**.

The solution presented in this paper requires a special partition of the static network so that any two **S-Operators** belonging to two different parties mistrust each other in a way that they will not cooperate for illegal and therefore potentially dangerous manipulations. Such a partition is called a partition **into mistrust parties**.

This mistrust between the parties can be established by a strict geographical, cultural and legal separation, by laws that prohibit certain forms of cooperation explicitly and by active measures to test the correct behavior of the **S-Operators** in the
sence of these laws. Such a test can include fake proposals for building manipulative coalitions, for example. S-Operators have the duty to report illegal offers they get in a standardized fashion. Because any illegal offer could just be a fake for testing the correct reaction, not reporting them might be very risky.

MP is used as abbreviation for mistrust party in general. A certain MP is identified with an index i and noted as MPi. If an S-Operator belongs to MPi, all the S-Nodes he is responsible for belong to MPi, too. #(MPi) is the total number of S-Nodes belonging to mistrust party MPi.

The general idea for the following solution for secure communication between two arbitrary S-Nodes is to split responsibilities among these mistrust parties.

B. Acquaintances, partisan forwarding

Two S-Nodes are called acquaintances, if messages can be exchanged between them over an arbitrary secure channel. Therefore the S-Operators of the acquaintances have to check the identities of each others S-Node's owner and they have to exchange the necessary communication data (including an exclusive secret key). This security critical manual operation is a high effort.

The S-Operators do also have to make sure that data can actually be transmitted between acquaintances in a finite time. Therefore, S-Operators of two S-Nodes that are acquaintances have to negotiate manually appropriate physical channels and they have to provide them to the S-Nodes. For example, one channel could be a direct microwave transmission and the Internet could be used as another single channel between the acquaintances.

Because of the high manual effort, an S-Node cannot have more than just a few acquaintances to be practicable.

Acquaintances do have high responsibility for each other. In order to split responsibilities between the mistrust parties, an arbitrary S-Node Si must get for each mistrust party MP, at least one S-Node belonging to MP, as acquaintance. This ensures that the identity of the owner of Si has to be verified for each different mistrust party at least by one S-Operator belonging to that MP whose S-Node becomes an acquaintance.

But to keep the manual effort on a reasonably low level, each S-Node should not require many more acquaintances than the total number of mistrust parties.

Only acquaintances may communicate directly with each other. If two S-Nodes are not acquaintances, a message can be forwarded between them if there is a series of pairwise acquaintances among them and if the message can be forwarded from one acquaintance to the next acquaintance in that series. Such an indirect connection is called a forwarding. The forwarding S-Nodes between the sender and the addressee are called forwarders.

For the solution presented in this paper, any two S-Nodes must be acquaintances or there must be a forwarding between them. In contrast to the direct communication with an acquaintance, the forwarding communication cannot take place over a secure channel because a sender and an addressee who are not acquaintances do not have an exclusive shared key with each other – they do not even know whether their pretended communication partner exists at all.

To make the communication between S-Nodes which are not acquaintances secure and reliable, there are additional requirements. For any two S-Nodes SA and SB belonging to the same MPi, there must be a connection without any S-Node of all the other mistrust parties involved. This means, that if SA and SB are not acquaintances, there must be a forwarding between them so that all the forwarders belong to MPi. Such a connection within a single MP is called partisan forwarding.

If the network structure within MPi is like a single ring so that each S-Node belonging to MPi has exactly two acquaintances in MPi, there is always a partisan forwarding between any two S-Nodes belonging to MPi; if they are not acquaintances.

C. Partition-routing

For secure communication between any two S-Nodes SA and SB that are not acquaintances the following protocol for partition-routing may be used:

1) Preparation: Let x be the bit sequence to be transmitted. SA creates a bit sequence xA containing a random one-time key KA, the encryption EKA(x) and a message authentication code P(KA◦ x).

\[ x_A = K_A \cdot E_K(x) \cdot P(K_A \cdot x) \]

Let n be \( n \in N \land n \geq \Psi \). SA builds the set of a secret sharing split:

\[ Z_n(x_A) = \{ T_{n,0}(x_A), \ldots , T_{n,n-1}(x_A) \} \]

Let A0 be the address of the addressee SA. Let H be additional required header data including some message number and the current time. SA generates the n split messages \( \tau_i = A_0 \cdot H \cdot T_{n,i}(x_A) \quad \forall i \in N \mid i < n \)

2) Separation: SA sends each \( \tau_i \) over a secure channel to a different acquaintance of SA, not belonging to any of the different parties belonging to SA or SB belong to. SA may not send more than one piece of Zn(xA) into any MP.

3) Check and forwarding: Each forwarding S-Node Si decrypts and checks messages m arriving over secure channels from its acquaintances.

If m is from an acquaintance not belonging to the same MP as Si, this acquaintance is the sender SA, Si generates an identity confirmation IAs containing the address of SA, the name and additional identity data that was manually exchanged and verified when SA and Si became acquaintances. Si adds IAs as proof of authenticity to the message m:

\[ m = \tau_i \cdot I_A \cdot \]

Else if m is from an acquaintance belonging to the same MP as Si, m must already contain an IAs.

Si must forward correct messages m for an addressee SB according to these rules:

3.1) If SB is not an acquaintance of SA, Si forwards m over a secure channel to the next forwarder, who must be one of the acquaintances of SA belonging to the same MP.
as $S_{o}$. The forwarder must be chosen so that the message gets closer to an acquaintance of $S_{o}$ belonging to the same MP as $S_{o}$.

Continue with step 3 for the next forwarder.

3.2) Else if $S_{o}$ is an acquaintance of $S_{a}$, $S_{r}$ forwards $m$ over a secure channel direct to $S_{o}$.

Continue with step 4.

4) **Check and collection:** The addressee $S_{a}$ decrypts and checks messages arriving over secure channels from its acquaintances and extracts $T_{a}(x_{p})$ from $t_{i}$ if possible. Correct arriving parts $T_{a}(x_{p})$ and the according identity confirmations $I_{a}$ are collected and stored together with the information from which MP they actually were forwarded.

Only the sender $S_{a}$ and the addressee $S_{o}$ do get more than one piece of $Z_{a}(x_{p})$ if this protocol is followed properly: In step 2, all the parts $T_{a}(x_{p})$ are distributed over secure channels to different mistrust parties. The forwarding of the loop in step 3.1 between an acquaintance of $S_{a}$ and an acquaintance of $S_{o}$ is a strictly partisan forwarding over secure channels. This means that all the parts $T_{a}(x_{p})$ stay in exactly the MP they were sent to at step 2 until they reach an acquaintance of $S_{o}$. Only then, at step 3.2, all the parts are send to the same MP, but they are directly send over secure channels to the addressee $S_{o}$.

To reconstruct $x_{p}$ from a subset of $Z_{a}(x_{p})$, at least $\Psi$ parts of $Z_{a}(x_{p})$ are required. Any attack to get $x_{p}$ and therefore any manipulation that is more sophisticated than just trying to guess an entire valid bit sequence must affect at least $\Psi$ forwarding S-Nodes in $\Psi$ different mistrust parties.

The identity confirmation $I_{a}$ as proof of authenticity has to be identical from at least $\Psi$ different mistrust parties, too. To cheat requires again that at least $\Psi$ S-Nodes in $\Psi$ different mistrust parties behave incorrect.

### D. Optimization

With the simple ring like network structure shown so far, each S-Node has exactly two acquaintances belonging to the same MP and that is theoretically enough because the required partisan forwarding is possible with that solution. But this is not yet a practicable solution, because there are two major weaknesses:

1) The efficiency is unusably low. Partisan forwarding may need great many S-Nodes as forwarders. In the worst case, a message has to be forwarded by 50% of the $(\#(MP_{i}))$ S-Nodes that belong to $MP_{i}$. With thousands or millions of S-Nodes, this would be terribly slow because the messages are not just forwarded – they have to be decrypted, checked and encrypted with another key. On average each S-Node would have to forward about 25% of all the messages exchanged by forwarding through its MP which can result in an extremely high workload, too.

2) The total system robustness would be very low. If only two S-Nodes belonging to the same MP are temporary not reachable for their acquaintances, the entire ring like network structure can break into two separate segments R and Q so that any partisan forwarding between an S-Node in R and another S-Node in Q would fail.

Robustness against failures in a communication network can be increased by mashing up the network tighter with additional redundant connection possibilities so that alternative routes can be chosen in case of failures [2]. By increasing the number of acquaintances within the same MP per S-Node, alternative routes for the partisan forwarding can be created. But more acquaintances imply also a higher manual effort.

With just a few more carefully chosen acquaintances for each S-Node and with a fitting routing concept, a good robustness can be achieved. By doing so, the length of the most efficient partisan forwarding between any two S-Nodes belonging to the same MP can be reduced to a practical value, too. The solution presented here is completely decentralized.
Requirements, definitions and strategic objectives

The following optimization requires the static address of an S-Node to consist of two independent components – one identifying the mistrust party MP, the S-Node belongs to and the other identifying the S-Node within MP. The last is called the Intra-MP-Address. The Intra-MP-Address must be a natural number and it must be unique within its mistrust party. The S-Node belonging to the same MP, can be sorted by their Intra-MP-Addresses.

For the optimization, for each S-Node $S_i$, belonging to MP, two acquaintances belonging to the same MP, are chosen according to the following rules:

1) The S-Node with the biggest Intra-MP-Address in MP, smaller than the Intra-MP-Address of $S_i$, becomes an acquaintance of $S_i$, if such an S-Node exists.

2) The S-Node with the smallest Intra-MP-Address in MP, bigger than the Intra-MP-Address of $S_i$, becomes an acquaintance of $S_i$, if such an S-Node exists.

3) Additionally, the S-Node belonging to MP, with the smallest Intra-MP-Address in MP, and the S-Node belonging to MP, with the biggest Intra-MP-Address in MP, become acquaintances.

This results again in a ring-like network structure per MP, but the S-Nodes on that ring are now sorted by their Intra-MP-Address.

The ring-distance $R(S_{old}, S_{new})$ between two S-Node $S_{old}$ and $S_{new}$ belonging to the same MP, is the number of S-Node on the sorted ring that are between $S_{old}$ and $S_{new}$ in the shorter direction.

The ring-distance is useful to define an objective for the optimization of the partisan forwarding with additional acquaintances in the same MP:

In the partisan forwarding process of a message between two arbitrary S-Node $S_{old}$ and $S_{new}$, it should be possible to reduce the ring-distance to $S_{new}$ at each forwarding step from $S_{old}$ to $S_{new}$ according to this formula:

$$R(S_{old}, S_{new}) \leq R(S_{old}, S_{new}) - \frac{R(S_{old}, S_{new})}{d} \quad \text{with} \quad d \in \mathbb{N} \cap d > 1$$

If the constant divisor $d$ is 2 for example, the new ring-distance should at least be reduced by 50% of the old ring-distance at each optimized partisan forwarding step.

Let $F_i$ be the number of S-Node required as forwarders between $S_{old}$ and $S_{new}$ in an optimized partisan forwarding in MP, $F_i$ would then be logarithmic with the number of S-Node belonging to MP:

$$F_i \leq (d - 1) \cdot \left\lfloor \log_d(\#(MP_i)) \right\rfloor$$

Let $A_i$ be the number of acquaintances each S-Node $S_i$ needs in his own MP to provide such an efficient partisan forwarding. If the acquaintances are chosen well distributed, the upper bound of $A_i$ for this optimization can be:

$$A_i \leq 2 \cdot \left\lceil \log_d(\#(MP_i)) \right\rceil$$

For $d=2$, $F_i$ becomes minimal, but $A_i$ becomes maximal, so the most acquaintances per S-Node will be required. Because making many acquaintances means a high manual effort, it probably makes sense to chose a higher $d$ and to accept slightly longer routes in the partisan forwarding.

In theory, acquaintances are perfectly distributed if each S-Node $S_i$ belonging to MP, does always have exactly those S-Node belonging to the same MP, as acquaintances that have a ring-distance of $\left\langle d - 1 \right\rangle$ with $f \in \mathbb{N} \cap f < \left\lfloor \log_d(\#(MP_i)) \right\rfloor$ to $S_i$. Because the ring-distances might change whenever a new S-Node is inserted into MP, and making new acquaintances has a high manual effort, for this optimization an approximation to the perfect distribution with enduring well-chosen acquaintances is the best solution.

It is not enough that efficient routes with no more than $F_i$ forwarders for the optimized partisan forwarding just exist: With the help of the Intra-MP-Address of the addressee $S_{old}$, any S-Node must actually be able to chose the best acquaintance a message should be forwarded to in order to bring it closer to $S_{old}$ on the optimal route.

![Figure 2: The optimal distributed acquaintances of S-Node $S_i$ in MP, with $d=2$. The numbers in the small circles that represent the other S-Node in MP, indicate their ring-distances to $S_i$.](image)

Procedure to add S-Node and to make acquaintances in a MP

This procedure creates and inserts new S-Node with their Intra-MP-Addresses into MP, and makes the optimized acquaintances within the same MP.

In advance, two constants $d \in \mathbb{N} \cap d > 1$ and $z \in \mathbb{N}$ have to be defined. Intra-MP-Addresses might have any values of natural numbers between 0 and $d^z$, so $d^z$ should be much bigger than the potential number of S-Node a single MP might actually ever have.

Let $a$ be a variable for the current Intra-MP-Address and let $r$ be an integer to count the rounds, which is initialized with 1.
1) **Initialization**: The first S-Node in MP₁ gets the Intra-MP-Address \( \alpha=0 \). No acquaintances in MP₁ have to be made.

2) **Increment of \( \alpha \)**: As long as an S-Node with the Intra-MP-Address \( \alpha \) exists already in MP₁, \( \alpha \) is incremented with \( d^{j}+d^{f} \).

3) **Check** for new round: If \( \alpha \geq d^{f} \), then \( r \) is incremented by one and \( \alpha \) is set to \( \alpha = d^{f}+d^{f} \).

4) **Insertion**: A new S-Node \( S_{n} \) with the Intra-MP-Address \( \alpha \) is inserted.

5) **Make acquaintances**: \( S_{n} \) should get for each \( f \in \mathbb{N} \), \( f > 0 \land f \leq r \) an acquaintance with the Intra-MP-Address \( \beta_{j} = (\alpha+d^{f}) \mod d^{f} \) and an acquaintance with the Intra-MP-Address \( \beta_{i} = (\alpha-d^{f}) \mod d^{f} \).

   The following sub-steps have to be done for each optimal address \( \beta_{j} \) and for each optimal address \( \beta_{i} \). The notation \( \beta_{k} \) will be used to express that.

   5.1) If there is already an S-Node with the Intra-MP-Address \( \beta_{j} \) in MP₁, then this S-Node \( S_{0j} \) becomes a final optimal acquaintance of \( S_{n} \).

   5.2) Else there is not yet an S-Node with the Intra-MP-Address \( \beta_{j} \) in MP₁.

      Let the address-distance \( \Delta(\phi, \chi) \) between two Intra-MP-Addresses \( \phi \) and \( \chi \) be:
      \[
      \Delta(\phi, \chi) = \min(|\text{abs}(\phi-(\chi+p\cdot d^{f}))| | p \in \mathbb{Z} |)
      \]

      Let \( W \) be the set of all S-Nodes in MP₁ that have a smaller address-distance to \( \alpha \) than \( \Delta(\beta_{i}, \alpha) \).

      If \( W \) is not empty, then the S-Node \( S_{n} \) whose Intra-MP-Address \( \gamma \) has the smallest \( \Delta(\gamma, \beta_{i}) \) of all the S-Nodes in \( W \) becomes a preliminary suboptimal acquaintance of \( S_{n} \).

      Else if \( r = 1 \) then the S-Node with the Intra-MP-Address 0 becomes a preliminary suboptimal acquaintance of \( S_{n} \).

      For the next S-Node to be inserted, continue with step 2.

**Note**: At the end of each round when \( r \) is incremented, all the perfectly distributed acquaintances exist. Whenever an already existing S-Node \( S_{0} \) becomes a new S-Node \( S_{n} \), some preliminary suboptimal acquaintances of \( S_{0} \) might become superfluous.

The average total number of suboptimal and optimal acquaintances made per S-Node in its own MP₁ is less than:
\[
1.5 \ast A_{j} = 3 \ast \log_{2}(\#(\text{MP₁}))
\]

**Foresighted partisan forwarding**

In the process of partisan forwarding each forwarder S-Node being not an acquaintance of the addressee \( S_{n} \) has to identify the acquaintance that would be the next optimal forwarder. In a network constructed the way shown before, that is the acquaintance with the Intra-MP-Address having the lowest address-distance to the Intra-MP-Address of \( S_{n} \).

If some S-Node \( S_{r} \) would be the next optimal forwarder, but the current forwarder \( S_{r} \) cannot reach \( S_{n} \), alternative routes may be tried until \( S_{r} \) is restored and reachable again. Alternative routes are not necessarily less efficient. To find the best alternative route is however more difficult: the address-distances have to be checked further ahead.

Let \( X \) be a set of S-Nodes that belong to MP₁. Let \( B(X) \) be the set of those S-Nodes belonging to the same MP₁, which have at least one acquaintance in set \( X \).

For optimal routing, \( S_{n}^{E} \) has to choose the S-Node in \( B(B(\{S_{1}\})\backslash S_{i})\) as next forwarder that has the Intra-MP-Address with the minimal address-distance to the Intra-MP-Address of \( S_{n} \).

Let \( S_{U} \) be an acquaintance of the addressee \( S_{n} \). If the addressee \( S_{n} \) is not reachable for \( S_{U} \), other acquaintances of \( S_{n} \) may be tried as final forwarders. Each S-Node in \( B(\{S_{1}\})\backslash S_{i} \) that has not yet been tried can be chosen as a preliminary target on an alternative partisan forwarding route. The number of alternative acquaintances for each S-Node in MP₁ is between \( r-1 \) and \( r\ast2-1 \) with \( r=\lceil \log_{2}(\#(\text{MP₁})) \rceil \).

Whenever an S-Node is not reachable and an alternative route is tried, this has to be logged in the message's header to make sure that no message circles around in an endless loop.

**Acquaintances in foreign mistrust parties**

In the protocol for partition-routing, between the first and the last forwarder only partisan forwarding is used to deliver each split message from the sender \( S_{1} \) to the addressee \( S_{n} \). For an efficient routing, it is essential to find an acquaintance of \( S_{U} \) belonging to the MP₁ in which the entire partisan forwarding takes place so that it can be used as preliminary target in the optimized partisan forwarding process.

Therefore, exactly those S-Nodes in different mistrust parties that have the same Intra-MP-Address should become pairwise acquaintances.

If in any MP₁ there is an S-Node \( S_{i} \) with the Intra-MP-Address \( \chi \), but in another MP₁ there is not yet an S-Node having the same Intra-MP-Address \( \chi \), \( S_{i} \) must get some suboptimal preliminary acquaintance in MP₁ because each S-Node must have at least one acquaintance in each MP₁.

Let \( S_{i} \) be the S-Node in MP₁ having the greatest Intra-MP-Address smaller than \( \chi \). Then \( S_{i} \) becomes the suboptimal preliminary acquaintance of \( S_{i} \) in MP₁.

If later an S-Node \( S_{n} \) is added to MP₁, then \( S_{n} \) becomes the optimal acquaintance of \( S_{i} \) in MP₁. The suboptimal preliminary acquaintance \( S_{i} \) becomes superfluous for \( S_{i} \).
If an S-Node $S_\beta$ has already a suboptimal preliminary acquaintance $S_\gamma$ in MP, it would principally be possible to create a better placed suboptimal preliminary acquaintance as soon as a new S-Node $S_\gamma$ is inserted in the same MP, having an Intra-MP-Address $\nu$ that is bigger than $\varphi$ and smaller than $\chi$. That would lead to shorter forwarding routes which are easier to find, but the additional manual effort to make acquaintances is probably too high.

Therefore, if a suboptimal preliminary acquaintance $S_\gamma$ for $S_\gamma$ already exists, it seems to be better to create only one more acquaintance for $S_\gamma$ in MP, which must be the optimal acquaintance $S_\nu$.

Protocol for optimized partition-routing

Let $S_m$ be the sender belonging to MP. Let $S_\beta$ be the addressee belonging to MP, and having the Intra-MP-Address $\beta$.

The following protocol has to be repeated for each split message of the partition-routing protocol. It delivers such a message $m$ from $S_m$ to $S_\beta$. All the forwards must belong to the same MP. Let MP, be that MP. Let $S_\gamma$ be a variable for an S-Node belonging to MP.

1) **Check for common acquaintance**: $S_m$ sends $m$ to an acquaintance $S_\gamma$ belonging to MP. If $S_\gamma$ is also an acquaintance of $S_\beta$, $S_\gamma$ can forward $m$ directly to $S_\beta$ and the protocol ends.

2) **Route to optimal acquaintance**: With the foresighted partisan forwarding, the S-Nodes in MP, try to deliver $m$ to an S-Node $S_\beta$ belonging to MP, and having the same Intra-MP-Address $\beta$ as $S_\beta$. If the S-Node $S_\beta$ exists and can be reached, $m$ can be forwarded in a single step from $S_\beta$ and the protocol ends.

3) **Go to start point for alternative search loop**: $S_\gamma$ is set to the S-Node having the biggest Intra-MP-Address smaller than $\beta$ in MP.

If the foresighted partisan forwarding did not end at $S_\nu$, but at $S_\alpha$, $m$ must be send now to $S_\nu$. This should always be possible in a single forwarding step because $S_\nu$ and $S_\chi$ are at least acquaintances.

4) **Try to reach addressee**: If $S_\nu$ is an acquaintance of $S_\beta$, $m$ is forwarded to $S_\beta$. End of the protocol.

5) **Check if search failed**: If $S_\nu$ has an Intra-MP-Address null, there is no acquaintance of $S_\beta$ in MP. End of the protocol.

6) **Forward to next possible acquaintance**: For any S-Node $S_\gamma$ having the Intra-MP-Address $\chi$ let $\Phi(S_\gamma)$ be the smallest natural number bigger null for that the equation $\chi$ modulo $d^{(r-\Phi(S_\gamma))} = 0$ holds. Note that $\Phi(S_\gamma)$ is identical with the round $r$ in which $S_\gamma$ was created.

Let $L(S_\gamma)$ be a subset of $B(S_\gamma)$ containing only those acquaintances $S_\nu$ that have a $\Phi(S_\nu)$ less or equal to $\Phi(S_\gamma)$.

$S_\gamma$ forwards $m$ to the S-Node of $L(S_\gamma)$ having the biggest Intra-MP-Address that is smaller than the Intra-MP-Address of $S_\nu$.

That S-Node becomes the new $S_\nu$.

Continue with step 4.

Let $F$ be the maximum number of forwarders required for the partition-routing of a message strictly split over $n$ mistrust parties. With the optimized protocol, $F$ is limited by the following formula:

$$ F \leq \sum_{i=0}^{n-1} (2*F_i + 3) = \sum_{i=0}^{n-1} (2*(d-1)*\log_2(\#(MP_i)) + 3) $$

To make communication more robust, the number of acquaintances in other mistrust parties could be increased.

But if the number of mistrust parties is bigger than the security threshold $\Psi$, in case of any disturbance in some MP, it would be possible to avoid MP completely and choose another MP instead to deliver a split message. Or if for the $Z_{\nu}$ $n$ is chosen bigger than $\Psi$, in up to $n-\Psi$ different mistrust parties there may be failures and the communication still works.

Also, secure connections between two acquaintances $S_\nu$ and $S_\chi$ belonging to different mistrust parties are only used for communication having at least one of the S-Nodes $S_\nu$ and $S_\chi$ as sender or addressee. If the direct connection between $S_\nu$ and $S_\chi$ is disrupted, the effect of this failure is rather limited. Only those messages that have $S_\nu$ or $S_\chi$ as sender or addressee are affected.

Therefore, additional redundancy seems to be superfluous for acquaintances in foreign mistrust parties.

Let $q$ be the number of mistrust parties. Any S-Node will not need more than $q-1$ acquaintances in all the other mistrust parties together.

Per S-Node belonging to MP, this leads to a total number $TA_i$ of required optimal acquaintances according to the following formula:

$$ TA_i \leq A_i + q - 1 = 2*\log_2(\#(MP_i)) + q - 1 $$

With the shown procedures, in average additional $SA_i$ suboptimal preliminary acquaintances will be created per S-Node:

$$ SA_i \sim \log_2(\#(MP_i)) + (q-1) \div 2 $$

### V. Possibilities

The solution presented so far offers only secure communication between S-Nodes which should always be online. Clients that are typically often offline cannot be a part of a static network. But the human beings and their client systems using the
static network should be able to communicate secure and reliable with any S-Node, too. There are several ways to do this:

An S-Node could work as a proxy server for its owner. The owner does only have to be able to communicate directly with his own S-Node over a secure channel. This S-Node can forward messages which should be exchange with other S-Nodes – using the partition-routing protocol for those other S-Nodes that are not acquaintances. For users having their own S-Node and who trust in its reliability, this is the preferred solution.

Of course, users could also exchange for each mistrust party MP, the required information for direct communication over a secure channel with at least one responsible S-Operator of an S-Node belonging to MP. Then, they could themselves start the partition-routing protocol. The advantage would be that the user does not need his own proxy S-Node. A failure of such a single S-Node could not hinder the user to communicate with other S-Nodes. The disadvantage would be the higher manual effort.

A. Application

If a computer network is anyway static and if a partition in mistrust parties is required for other reasons than secure communication, too, then the security concept presented here does not produce much additional effort. The S-Network, a trustworthy repository currently developed at Fraunhofer FOKUS, is a good candidate for this concept: The S-Network combines secure long term data storage and preservation in a computer network with non-repudiation and an international applicable legal validity. For the future, the S-Network must be guaranteed to be secure even after any possible technical advance.

The S-Network uses mistrust parties to store backup copies in a distributed way and it requires secure communication between the systems storing the backup copies. With the concept presented in this paper, exactly the security level that is reached for data preservation can be guaranteed for the required message exchange. The same provable secure base technologies like secret sharing can be used, and of course the same mistrust parties, too.

The solution presented here was already successfully implemented in a prototype of the S-Network.

VI. CONCLUSION

This paper presents a practical concept for secure communication in a large computer network without a "trusted party" or a "web of trust" and without relying on assumptions of the complexity theory. Unlike in previous PSMT proposals, a realistic concept to actually create communications paths with disjoint sets of nodes is provided.

Depending on the choice of algorithms used to build secure channels between acquaintances, unlimited calculating power does not help to successfully break the security of this solution. That makes this solution applicable where ever a strong long term security concept is required.

Perfect security is not guaranteed: An attacker could randomly generate a message that passes the integrity tests. The likelihood that a valid message is guessed can be reduced by expanding the MAC.

The security also depends on the choice of $\Psi$: A coordinated manipulation involving at least $\Psi$ S-Nodes in $\Psi$ different mistrust parties can break the security concept. Increasing $\Psi$ and the number of mistrust parties probably only makes sense up to a certain degree. It is really decisive to prevent manipulative cooperation among the mistrust parties.

For a trustworthy repository that has to guarantee secure long term serviceability like the S-Network, the solution presented here seems to be a good choice.

REFERENCES


