Efficient Reliable Communication over Partially Authenticated Networks*

Amos Beimel Lior Malka

Dept. of Computer Science
Ben Gurion University, Beer Sheva 84105, Israel.
Email: beimel,liorma@cs.bgu.ac.il
July 26, 2004

Abstract

Reliable communication between processors in a network is a basic requirement for executing any protocol. Dolev [7] and Dolev et al. [8] showed that reliable communication is possible if and only if the communication network is sufficiently connected. Beimel and Franklin [1] showed that the connectivity requirement can be relaxed if some pairs of processors share authentication keys. That is, costly communication channels can be replaced by authentication keys.

In this work, we continue this line of research. We consider the scenario where there is a specific sender and a specific receiver in a synchronous network. In this case, the protocol of [1] has $n^{O(n)}$ rounds even if there is a single Byzantine processor. We present a more efficient protocol with round complexity of $(n/t)^{O(t)}$, where n is the number of processors in the network and t is an upper bound on the number of Byzantine processors in the network. Specifically, our protocol is polynomial when the number of Byzantine processors is $O(1)$, and for every t its round complexity is bounded by $2^{O(n)}$. The same improvements hold for reliable and private communication. The improved protocol is obtained by analyzing the properties of a "communication and authentication graph" that characterizes reliable communication.

Key Words. Reliable communication, Fault tolerance, Authentication, Incomplete networks.

1 Introduction

Suppose that some processors are connected by an incomplete synchronous network of reliable channels. The processors cooperate to execute some protocol, but some of them are maliciously faulty. Dolev [7] and Dolev et al. [8] proved that if there are t faulty processors, then every pair of processors can communicate reliably if and only if the network is $(2t + 1)$-connected. Beimel and Franklin [1] showed that the connectivity requirement can be relaxed if *some* pairs of processors share authentication keys. That is, costly communication channels can be replaced by authentication keys.

In this paper we consider the problem of "single-pair" reliable communication in partially-authenticated, synchronous networks. In this problem there is a specific sender a who wants to send a message to a specific receiver b, such that any coalition of at most t faulty processors cannot prevent this transmission. The communication channels in the network define a natural "communication graph," with an edge between two vertices for every channel between two processors. The pairs of processors sharing authentication

*A preliminary version of this paper appeared in [2].

keys define a natural "authentication graph," with an edge between two vertices for every shared key. The *partially-authenticated network*, which is the union of the two graphs, is given and known to all of the processors. To enable reliable communication from a to b there must be at least $t+1$ disjoint paths from a to b in the communication graph (otherwise, there are t vertices that can fail-stop, disconnecting a from b). If a and b share an authentication key and there are $t+1$ disjoint communication paths from a to b, then reliable communication from a to b is possible and efficient. But what if a and b do not share an authentication key? We do not want to give a new key to a and b, because distributing and maintaining these keys is expensive. Our goal is to achieve reliable communication from a to b using the existing communication and authentication capabilities of the two graphs.

Beimel and Franklin [1] characterize when reliable communication is possible using these two graphs. Their characterization depends on recursively defined graphs which include all of the edges of the communication graph and some of the edges of the authentication graph. However, the reliable protocol presented by Beimel and Franklin [1] is inefficient; it requires $n^{O(n)}$ rounds, where n is the number of processors in the network. In this paper we present a more efficient protocol obtained by exploiting the properties of the graphs that characterize reliable communication.

Historical Notes. The connectivity requirements for several distributed tasks in several models has been studied in many papers; for example, Byzantine agreement [7, 11], approximate Byzantine agreement [9, 24], reliable message transmission [7, 8], and reliable and private message transmission [19, 8, 21, 23]. Simple impossibility results and references can be found in [11, 18]. We mention that in Byzantine agreement all honest processors should agree on the same message while in reliable communication only the sender and the receiver agree on the message. Beimel and Franklin [1] considered the connectivity requirements in partially-authenticated networks. In addition to the "single-pair" version of the problem, they characterize when reliable transmission is possible in the "all-pairs" version. In this version any sender should be able to reliably transmit a message to any receiver, such that any coalition of at most t faulty processors cannot prevent this transmission. Sayeed and Abu-Amara [20] gave a secure message transmission protocol for asynchronous networks. Kumar et al. [17] studied the secure message transmission problem in the non-threshold setting. Goldreich, Goldwasser, and Linial [14], Franklin and Yung [13], Franklin and Wright [12], and Wang and Desmedt [5] studied secure communication and secure computation in multi-recipient (multicast) models. Wang and Desmedt [6] studied secure computation in directed networks. Blaser et al. [3] characterize some of the functions that can be securely computed in non-2-connected networks.

Our Results. Our main result is a more efficient protocol for "single-pair" reliable communication. The round complexity of our protocol is $(n/t)^{O(t)}$, where n is the number of processors in the network and t is an upper bound on the number of Byzantine processors in the network. Specifically, our protocol is polynomial when the number of Byzantine processors is $O(1)$, and for every t its round complexity is bounded by $2^{O(n)}$. The improved protocol is obtained by analyzing the properties of the graphs that characterize reliable communication. We exploit these properties to obtain a protocol with better round complexity than the protocol of [1]. It remains open whether there is a reliable message transmission protocol with polynomial number of rounds for $t = \omega(1)$.

Our improved protocol for reliable communication implies, using a transformation of [12, 1], an improved protocol for reliable and *private* communication, that is, a protocol in which a message is reliably transmitted and the adversary learns nothing about it (other than the information that a message is being transmitted). Hence, we obtain a protocol for reliable and *private* communication with round complexity $(n/t)^{O(t)}$.

We also give a simple characterization for reliable communication against one Byzantine processor. In this case a simple necessary condition for reliable communication is that the communication graph is 2-

connected between a and b and the union of the communication and authentication graphs is 3-connected between a and b. We show that this condition is sufficient when the communication graph is connected. This characterization implies that reliable communication is symmetric for $t = 1$. However, we show that the natural generalization of this condition to $t \geq 2$ is not sufficient. Finally, we show that reliable communication is not symmetric for $t \geq 2$. That is, there is a partially-authenticated network (i.e., a combination of a communication graph and an authentication graph) for which reliable communication is possible from a to b, but is not possible from b to a. This result is somewhat counter-intuitive as the edges are bi-directional.

Organization. In Section 2, we describe our model, review relevant results from [1], and describe a simplified protocol SIMPLESEND. In Section 3, we study the properties of the "useful communication graph." In Section 4, we use these properties to prove that Protocol SIMPLESEND is efficient. In Section 5, we show how Protocol SIMPLESEND can be transformed to a protocol that achieves fault-restricted reliable communication, and, in Section 6, we show how to use the fault-restricted protocol to achieve private and reliable communication. In Section 7, we discuss the symmetry/asymmetry of reliable communication. Finally, in Section 8, we present some concluding remarks and open problems.

2 Preliminaries

2.1 The Model

In this work we consider a synchronous network in which the communication channels are assumed to be reliable. Specifically, the network is modeled by an undirected graph $G_{\mathrm{C}} = \langle V, E_{\mathrm{C}} \rangle$, where V is the set of processors in the network (i.e., $|V| = n$), and E_{C} describes the communication channels. That is, there is an edge $\langle u, v \rangle$ in E_{C} if and only if there is a communication channel between u and v. We assume that these communication channels are reliable: an adversary that does not control u or v (but may control other vertices in the network) cannot change or delete a message sent on the edge $\langle u, v \rangle$ or insert a message on the channel. (In Section 6 we assume that these channels are also private.) Some pairs of processors share authentication keys. Informally, an authentication scheme enables a sender and a receiver who share a common key to exchange messages such that the receiver can verify that the message was sent by the sender (see Section 2.2 for more details). We describe which pairs of processors share a common authentication key by a graph $G_{\mathrm{A}} = \langle V, E_{\mathrm{A}} \rangle$, in which u and v share a common key, denoted by $k_{u,v}$, if and only if $\langle u, v \rangle \in E_{\mathrm{A}}$. These keys are chosen according to some known probability distribution, and every set of vertices (processors) has no information on the keys of disjoint edges (pairs of processors), except for their a priori probability distribution. A *partially-authenticated network* is a triplet $N = \langle V, E_{\mathrm{C}}, E_{\mathrm{A}} \rangle$, where $G_{\mathrm{C}} = \langle V, E_{\mathrm{C}} \rangle$ is the communication graph and $G_{\mathrm{A}} = \langle V, E_{\mathrm{A}} \rangle$ is the authentication graph.

We consider protocols for message transmission, in which a sender $a \in V$ wants to transmit a message M to a receiver $b \in V$ in a partially-authenticated, synchronous network. There is a global clock in the system and the protocol proceeds in rounds. That is, at the beginning of each round each processor $v \in V$ sends messages to some of its neighbors in the graph G_{C}. These messages get to the neighbors before the beginning of the next round. Furthermore, each processor knows when a round starts and what messages it got from its neighbors in the previous round (and which neighbors did not send a message). Alternatively, the requirement that the network is synchronous can be replaced with the following requirement: the network can be asynchronous, but there is a timeout constant d such that any message sent from any processor to any of its neighbors arrives at the neighbor after at most d time units. The *round complexity* of a protocol is the maximum number of rounds that elapse from its activation to its termination. The *message complexity* of a protocol is the total number of bits in messages exchanged in a round by the non-Byzantine processors, maximized over all of the rounds.

Assumptions. We assume that all processors in the system know the topology of the partially-authenticated network $\langle V, E_{\mathrm{C}}, E_{\mathrm{A}}\rangle$. Furthermore, all the processors in the system know in which round processor a starts to transmit a message to processor b.

Attack model. During the execution there might be Byzantine faults (also known as "active attacks"). An adversary, with unlimited power, controls a subset T of the processors. The adversary knows the protocol, the distribution under which the authentication keys where chosen, and the topology of the network (i.e., G_{C} and G_{A}). For every processor in T, the adversary knows all the messages received by that processor, its random inputs, and its keys. The adversary can choose T adaptively during the execution of the protocol. From the moment a processor is included into T, the adversary determines the messages this processor sends thereafter (possibly deviating from the protocol specification in an arbitrary manner). Once a processor joins the set T of Byzantine processors, it cannot be excluded from T.

Definition 2.1 (**(t,ϵ)-reliable communication**) *Let $N = \langle V, E_{\mathrm{C}}, E_{\mathrm{A}}\rangle$ be a partially-authenticated network, and $a, b \in V$ be a sender and a receiver. A message transmission protocol from a to b in N is* (t,ϵ)-reliable *if for every message M transmitted from a to b by the protocol, when the adversary can control any set T of at most t processors such that $T \subseteq V \setminus \{a, b\}$, the probability that b accepts the message M is at least $1-\epsilon$, where the probability is over the random inputs of the processors, the distribution of the authentication keys, and the random input of the adversary.*

In this paper we consider the problem of fault-restricted reliable communication. We emphasize that fault-restricted reliable communication is a tool for characterizing when t-reliable transmission between a given pair of processors is possible. Furthermore, our protocol for t-reliable transmission uses a fault-restricted reliable transmission protocol as a sub-protocol. In the fault-restricted version at least one of two given sets T_0, T_1, which are not necessarily disjoint, is guaranteed to contain all of the faulty processors. We use the term *suspected vertex* for a vertex from $T_0 \cup T_1$, that is – a vertex that is possibly Byzantine.

Definition 2.2 (**$(\{T_0, T_1\}, \epsilon)$-reliable communication**) *Let $N = \langle V, E_{\mathrm{C}}, E_{\mathrm{A}}\rangle$ be a partially-authenticated network, $a, b \in V$ be a sender and a receiver, and $T_0, T_1 \subseteq V \setminus \{a, b\}$ be a pair of sets. A message transmission protocol from a to b in N is* $(\{T_0, T_1\}, \epsilon)$-reliable *if for every message M transmitted from a to b by the protocol, when the processors controlled by the adversary are contained in at least one of T_0, T_1, the probability that b accepts the message M is at least $1-\epsilon$, where the probability is over the random inputs of the processors, the distribution of the authentication keys, and the random input of the adversary.*

It was proved in [1] that (t,ϵ)-reliable communication is possible if $(\{T_0, T_1\}, \epsilon/\binom{n}{t})$-reliable communication is possible for every pair T_0, T_1 of sets of size t. This (t,ϵ)-reliable protocol executes (in parallel) the $(\{T_0, T_1\}, \epsilon/\binom{n}{t})$-reliable protocol for every pair of sets of size t, and the receiver learns the message that was sent from the sender by analyzing the results of these executions. In particular, if t is constant and the $(\{T_0, T_1\}, \epsilon/\binom{n}{t})$-reliable protocol is efficient for every T_0, T_1 of size at most t, then the resulting (t,ϵ)-reliable protocol is efficient. See details in Section 6.

The reliability of a network is closely related to its connectivity. We consider *vertex* connectivity of *undirected* graphs. Two paths from a to b are *vertex disjoint* if no vertices other than a and b appear on both paths. A path P *passes* through a set T if there is a vertex $u \in T$ in the path. Otherwise, we say that P *avoids* T or P is T*-avoiding*. A graph $G = \langle V, E\rangle$ is (t, u, v)*-connected* if $\langle u, v\rangle \in E$ or if there are t vertex-disjoint paths from u to v. There is an efficient algorithm that checks whether a graph is (t, u, v)-connected (see, e.g., [10]).

2.2 Authentication Schemes

We briefly describe authentication schemes; the reader is referred to, e.g., [22] for more details. Let s be a positive integer and K be a finite set, called the set of keys. An authentication scheme for messages in $\{0,1\}^s$ is a pair $\langle \text{AUTH}, \mu \rangle$ for which $\text{AUTH} : \{0,1\}^s \times K \to \{0,1\}^*$ is a function and μ is a probability distribution on the set of keys K. An authentication scheme $\langle \text{AUTH}, \mu \rangle$ can be used to send messages between two processors, which we call Alice and Bob, in the following way: in the initialization stage, Alice and Bob are given a shared secret key $k \in K$ chosen from the probability distribution μ. To send an authenticated message M to Bob, Alice computes $\alpha = \text{AUTH}(M, k)$, called a *tag*, and sends the pair $\langle M, \alpha \rangle$ to Bob. When Bob receives a pair $\langle M', \alpha' \rangle$, he verifies that $\alpha' = \text{AUTH}(M', k)$, in which case Bob accepts the message M'. Informally, the scheme is ϵ-secure if the probability that an adversary can cause Bob to accept a message that was not sent by Alice is at most ϵ.

We use the notion of an ℓ-adversary in the following definition. Let $\langle \text{AUTH}, \mu \rangle$ be an authentication scheme, and let $k \in K$ be a key chosen from the probability distribution μ. An *ℓ-adversary* is a computationally unlimited adversary who does not know k, but can get ℓ tags $\alpha_i = \text{AUTH}(M_i, k)$ of messages of its choice. The adversary can choose these messages adaptively. That is, its strategy has, without loss of generality, ℓ stages; in the ith stage the adversary chooses a message M_i which may depend on the previous messages and tags, and the adversary gets the tag $\alpha_i = \text{AUTH}(M_i, k)$. If an ℓ-adversary can produce a pair $\langle M, \alpha \rangle$ for which $M \neq M_i$ for every $0 \leq i \leq \ell$ and $\alpha = \text{AUTH}(M, k)$, then the ℓ-adversary *breaks* the scheme, and the pair $\langle M, \alpha \rangle$ is a forgery.

Definition 2.3 (**(ℓ, ϵ)-authentication scheme**) *The scheme $\langle \text{AUTH}, \mu \rangle$ is an (ℓ, ϵ)-authentication scheme if any ℓ-adversary cannot break the scheme with probability greater than ϵ, where the probability is over the distribution of the authentication keys and the random input of the adversary.*

Authentication schemes based on hash functions were presented in [4, 25]. Efficient authentication schemes were presented by Krawczyk [15, 16]. In these schemes, for every message of length s there is an (ℓ, ϵ)-authentication scheme with keys of length $O(\ell \cdot \log \frac{1}{\epsilon} + \log s)$ and tags of length $O(\log(\frac{1}{\epsilon}))$.[1] Note that the length of the tag is independent of the length of the message. We use these schemes throughout the paper.

Remark 2.4 Our definitions of reliable communication and authentication schemes consider an adversary with unlimited computational ability. An alternative approach is to consider a polynomial-time adversary and to require that it cannot break the protocol/scheme in polynomial time. Breaking our reliable protocol implies breaking the underlying authentication scheme. Hence, if we use authentication schemes that are secure against polynomial-time adversaries, then the resulting reliable communication protocol is computationally secure (i.e., secure against polynomial-time adversaries.)

2.3 Simple Examples of our Protocol

To illustrate some of the ideas used in our protocol for fault-restricted reliable communication, we first consider a simple example. Let N_0 be the partially-authenticated network described in Figure 1.[2] In this network, a wants to send a message to b, and both know that at most one of t_0, t_1 is Byzantine. If a shared an authentication key with b, then it could use it to send the authenticated message along the paths $\langle a, t_0, b \rangle$ and $\langle a, t_1, b \rangle$. The authenticated message would then arrive on at least one of these paths, and b would verify

[1] If we settle for computational security, then the length of the key becomes shorter.

[2] Throughout this paper, communication channels are described by solid lines and authentication edges are described by dashed lines.

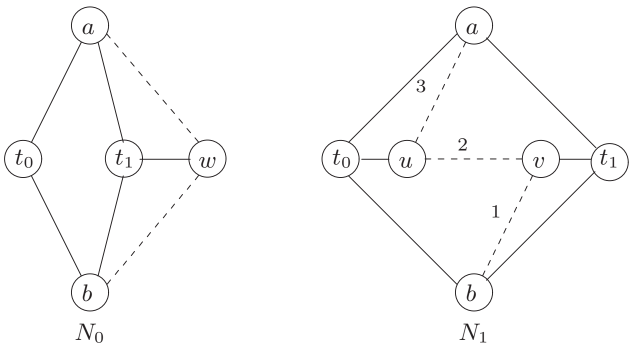

Figure 1: Examples of partially-authenticated networks. The numbers indicate the levels of authentication edges as defined in Definition 2.9.

its authenticity using the shared key. As we assume that the adversary cannot forge authenticated messages sent by the honest processors, b accepts the message sent by a.

However, a and b do not share an authentication key in N_0. Instead, a sends the message M to b on the three paths $\langle a, t_0, b\rangle$, $\langle a, t_1, b\rangle$, and $\langle a, w, b\rangle$. Transmission on $\langle a, w, b\rangle$ is done as follows: a authenticates M using the shared key $k_{a,w}$ and sends the authenticated message to w on the path $\langle a, t_1, w\rangle$. If w receives a valid authenticated message M from a, it uses the shared key $k_{w,b}$ to authenticate M and the authenticated message is sent to b on the path $\langle w, t_1, b\rangle$; otherwise, w sends nothing. If b receives a valid authenticated message from w, it accepts it. Otherwise, b deduces that t_1 corrupted the transmission on either $\langle a, t_1, w\rangle$ or $\langle w, t_1, b\rangle$; thus t_0 is honest and therefore b accepts the message that arrived on $\langle a, t_0, b\rangle$. Notice that the processors use the fact that the network is synchronous only to determine when a message should arrive.

In the protocol for reliable transmission in N_0, for every authentication edge, a path in the communication graph was used to send the authenticated message from one side of the edge to its other side. Let us see what happens if we apply this idea to N_1, described in Figure 1. As before, a sends the message M on the three paths $\langle a, t_0, b\rangle$, $\langle a, t_1, b\rangle$, and $P_{a,b} \stackrel{\text{def}}{=} \langle a, u, v, b\rangle$. To send an authenticated message over the authentication edges $\langle a, u\rangle$ and $\langle v, b\rangle$, we can use the paths $\langle a, t_0, u\rangle$ and $\langle v, t_1, b\rangle$ respectively. To send an authenticated message over the authentication edge $\langle u, v\rangle$, we choose some path from u to v in the communication graph. However, since any path from u to v in the communication graph passes through both t_0 and t_1, if v does not receive a valid authenticated message from u, it does not know which of t_0, t_1 has corrupted this transmission. Hence, b cannot deduce which of t_0, t_1 is Byzantine, and it cannot choose the right message.

To solve this problem, let us reconsider the transmission of M on the path $P_{a,b}$. We use a sub-protocol with 8 rounds to achieve this goal. After this sub-protocol, either b receives the authentic message M or b can detect the Byzantine vertex. We first describe the sub-protocol round by round, and then try to abstract its components.

Round 1 – a, t_0: The sender a sends $m_1 \leftarrow \langle M, \text{AUTH}(M, k_{a,u})\rangle$ to t_0.

Round 2 – t_0, u: Vertex t_0 sends the message m_2, which should be equal to m_1, to u.

Round 3 – u, t_0: Vertex u gets a message $m_2 = \langle M', \alpha'\rangle$ and verifies that $\alpha' = \text{AUTH}(M', k_{a,u})$.

- If u does not receive a message in Round 2 or the authentication fails, then u knows that t_0 is Byzantine, and sends nothing to t_0.
- Otherwise, u sends $m_3 \leftarrow \langle M, \text{AUTH}(M, k_{u,v})\rangle$ to t_0.

Round 4 – t_0, b: Vertex t_0 sends m_4, which should be equal to m_3, to b.

Round 5 – b, t_1: If b does not receive a message from t_0 in Round 4, then it detects that t_0 is Byzantine. Otherwise, it receives m_4. The receiver b sends $m_5 \leftarrow \langle m_4, \text{AUTH}(m_4, k_{b,v})\rangle$ to t_1.

Round 6 – t_1, v: Vertex t_1 sends m_6, which should be equal to m_5, to v.

Round 7 – v, t_1: Vertex v gets a message $m_6 = \langle M'', \alpha''\rangle$ and verifies that $\alpha'' = \text{AUTH}(M'', k_{b,v})$. There are three options:

- If v does not receive a message in Round 6 or the authentication fails, then v knows that t_1 is Byzantine, and sends nothing to t_1.
- If the authentication is valid, then v knows that $m_4 = M''$, which is supposed to be equal to $\langle M, \text{AUTH}(M, k_{u,v})\rangle$. In other words, $M'' = \langle M''', \alpha'''\rangle$. Now, v verifies that $\alpha''' = \text{AUTH}(M''', k_{u,v})$. If the authentication is valid, v knows that $M = M'''$ and sends $m_7 \leftarrow \langle M, \text{AUTH}(M, k_{b,v})\rangle$ to t_1.
- If the authentication of M''' is not valid, v knows that t_0 is Byzantine, and sends to t_1 the message $m_7 \leftarrow \langle$"error", $\text{AUTH}($"error"$, k_{b,v})\rangle$.

Round 8 – t_1, b: Vertex t_1 sends m_8, which should be equal to m_7, to b.

After round 8: The receiver b gets a message $m_8 = \langle M'''', \alpha''''\rangle$ and verifies that $\alpha'''' = \text{AUTH}(M'''', k_{b,v})$. There are three options:

- If b does not receive a message in Round 8 or the authentication fails, then b detects that t_1 is Byzantine.
- Otherwise, if b receive the message "error", then it detects that t_0 is Byzantine.
- Otherwise, b receives a message with valid authentication. This must be the authentic message sent by a.

Finally, if b detects in Round 5 or after Round 8 that t_i is Byzantine, it accepts the message sent on the path $\langle a, t_{1-i}, b\rangle$.

To understand the above protocol, we separate the flow of messages on the path $\langle a, t_0, u, t_0, b, t_1, v, t_1, b\rangle$ to virtual paths. Our goal is to send the message on the path $\langle a, u, v, b\rangle$. In Rounds 1 and 2, we "send" the message on the authentication edge $\langle a, u\rangle$ using the path $\langle a, t_0, u\rangle$. In Rounds 3-6, we "send" the message on the authentication edge $\langle u, v\rangle$ using the path $\langle u, t_0, b, v\rangle$. The authentication edge $\langle b, v\rangle$ appears in this path, and in Rounds 5 and 6 we "send" the message on the authentication edge $\langle b, v\rangle$ using the path $\langle b, t_1, v\rangle$. Finally, in Rounds 7 and 8, we "send" the message on the authentication edge $\langle v, b\rangle$ using the path $\langle u, t_1, b\rangle$. Each time we need to "send" the message on an authentication edge, we choose a path that avoids at least one of t_0, t_1, and send the message, together with its appropriate authentication, on this path. Thus, if the message arrives without the valid authentication we can detect the Byzantine vertex.

Our general protocol follows the same ideas used for N_1. That is, to "send" a message on an authentication edge $\langle u, v\rangle$, an authenticated message is sent from u to v by propagating the message over a path that avoids at least one of the sets T_0, T_1. This way, if the message arrives with invalid authentication, the Byzantine set is detected. This path can contain additional authentication edges and the same idea is used recursively. In this process, we handle two problems. First, we make sure that this recursion is not applied infinitely. Second, we use an appropriate alert mechanism to ensure that the receiver b detects a set T_i that contains all Byzantine vertices.

2.4 Characterizing Fault-Restricted Reliable Communication

In this section we quote the definitions of G^* and confusing pairs from [1]. These definitions characterize when a can (T_0, T_1)-reliably transmit a message to b. We also use them extensively to describe our protocols.

Definition 2.5 (Honest and avoiding paths) *Let $\langle V, E\rangle$ be a graph, u and v be some vertices in V, and T_0, T_1 be subsets of V. A path $\langle u, \ldots, v\rangle$ from u to v is* honest *if it avoids $T_0 \cup T_1$. A path $\langle u, \ldots, v\rangle$ from u to v is* avoiding *if it avoids at least one of the sets T_0,T_1.*

Whereas the above definition can be applied to any graph, the next definitions apply only to partially authenticated networks. To motivate the next definition consider an authentication edge $\langle u, v\rangle$ with a T_i-avoiding path from u to v in G_{C}, and an honest path from v to b in G_{C}. When u wants to send a message M to v, it authenticates M using the shared key $k_{u,v}$ and then sends the authenticated message along the T_i-avoiding path from u to v. If the message does not arrive at v when it is supposed to or if it arrives with invalid authentication, then v immediately knows that the set $T_{\bar{i}}$ is controlled by the adversary.[3] Furthermore, v can share this information with b using the honest path from v to b. This makes the edge $\langle u, v\rangle$ useful. The following definition formalizes the idea of useful edges and how new useful edges can be added iteratively.

Definition 2.6 (The graphs G_j and G^*) *Let $G = \langle V, E_{\mathrm{C}}, E_{\mathrm{A}}\rangle$ be a partially-authenticated network, a be the sender, b be the receiver, and $T_0, T_1 \subseteq V \setminus \{a, b\}$ be a pair of sets. We inductively define a sequence of graphs G_j for $j \geq 0$. First, we define $G_0 = \langle V, E_0\rangle$ where $E_0 = E_{\mathrm{C}}$. For every $j \geq 1$, we define $G_j = \langle V, E_j\rangle$, where E_j is the union of E_{j-1} with the set of all authentication edges $\langle u, v\rangle \in E_{\mathrm{A}}$ for which all of the following properties hold:*

1. $u, v \notin T_0 \cup T_1$,
2. *There is an avoiding path from u to v in $G_{j-1} = \langle V, E_{j-1}\rangle$, and*
3. *There is an honest path in G_{j-1} from either u or v to b.*

Finally, we define $G^* \stackrel{def}{=} G_n$ and $E^* \stackrel{def}{=} E_n$.

Informally, the graph G^* is the useful communication graph, as it contains exactly the edges that are used for reliably transmitting a message from a to b. That is, given a partially-authenticated network $N = \langle V, E_{\mathrm{C}}, E_{\mathrm{A}}\rangle$, our protocols only use the communication graph $\langle V, E_{\mathrm{C}}\rangle$ and the authentication keys $k_{u,v}$ of authentication edges $\langle u, v\rangle \in G^*$. During the execution of our protocols, Property (2) ensures that v learns the Byzantine set if an invalid message arrives from u, and Property (3) ensures that it can tell b about it. Also, as E_{A} is finite, there is a k for which $E_{k+i} = E_k$ for every $i \geq 0$. The graph G^* is defined as G_n since it is proven in [1] that $E_{n+i} = E_n$ for all $i \geq 0$.

Remark 2.7 Property (3) in Definition 2.6 implies that the topology of the graph G^* depends on the receiver b. Thus, changing the receiver may change the topology of G^*, a fact that we use in Section 7 to show that reliable communication is asymmetric.

Remark 2.8 Authenticating a message M over an authentication edge $e = \langle u, v\rangle \in E_{\mathrm{A}}$ is not necessary if there is an honest path from u to v in G_{C}. In such a case, M is reliably transmitted over this path, and e can be discarded. Hence, without loss of generality, we assume throughout the paper that there are no such edges in E_{A}.

[3] Throughout this paper, if $i = 1$ then $\bar{i} = 0$, and if $i = 0$ then $\bar{i} = 1$

We next define the notion of *level* of an edge, which is the stage in which it joins G^*. As we will see, our protocols send messages over edges of level i by using edges of level at most $i-1$. Thus, the higher the level of an edge, the more rounds it takes to send a message over that edge.

Definition 2.9 (Level of edges and paths) *Let $N = \langle V, E_{\mathrm{C}}, E_{\mathrm{A}} \rangle$ be a partially-authenticated network, a be the sender, b be the receiver, and $T_0, T_1 \subseteq V \setminus \{a, b\}$ be a pair of sets. The* level *of an edge $e = \langle u, v \rangle \in E_{\mathrm{C}} \cup E_{\mathrm{A}}$ is* $\mathrm{level}(e) \stackrel{def}{=} \min\{j | e \in E_j\}$. *The level of a path P is* $\mathrm{level}(P) \stackrel{def}{=} \max\{\mathrm{level}(e) | e \in P\}$.

Note that e is a communication edge if and only if it has level 0. Obviously, a path has level 0 if and only if it is a path in G_{C}. Also, for every authentication edge $e = \langle u, v \rangle$ with $\mathrm{level}(e) = j$, there is an honest path from either u or v to b of level at most $j-1$. Therefore, if there is an honest path $P_{v,b}$ from v to b of level at most $j-1$, then the path $P_{u,b} = \langle u, v \rangle, P_{v,b}$ (that is, $P_{u,b}$ is the concatenation of the edge $\langle u, v \rangle$ and the path $P_{v,b}$) is an honest path from u to b of level at most j. We conclude that in G^* there are honest paths from both u and v to b of level at most j.

We use N_1 described in Figure 1 to demonstrate these definitions. In this network we have $\langle v, b \rangle \in E_1$ since $\langle v, t_1, b \rangle$ is an avoiding path from v to b in G_0. Hence, the level of $\langle v, b \rangle$ is 1. Next, $\langle u, v \rangle$ is added to E_2 because $\langle u, t_0, b, v \rangle$ is an avoiding path from u to b in G_1 and $\langle v, b \rangle$ is an honest path from v to b in G_1. Hence, the level of $\langle u, v \rangle$ is 2. Finally, the edge $\langle a, u \rangle$ is added to E_3 and its level is 3. Note that $\langle a, u \rangle$ can be added to G^* only after $\langle v, b \rangle$ and $\langle u, v \rangle$ are added to G^* because we require that there is an honest path from either a or u to b.

Definition 2.10 (Confusing pairs) *Let $N = \langle V, E_{\mathrm{C}}, E_{\mathrm{A}} \rangle$ be a partially-authenticated network and $a, b \in V$ be a sender and receiver. A pair of sets (T_0, T_1) is an (a, b)-confusing pair in N if $T_0, T_1 \subseteq V \setminus \{a, b\}$, and at least one of the following holds:*

1. *There is an index $i \in \{0, 1\}$ such that every path from a to b in G_{C} passes through T_i, or*
2. *Every path from a to b in G^* passes through $T_0 \cup T_1$.*

If Property (1) of Definition 2.10 holds and the adversary controls T_i, then the Byzantine processors in T_i can block the communication from a to b. However, if it does not hold, then for every index $i \in \{0, 1\}$ there is a T_i-avoiding path P_i from a to b in G_{C}. For every message M sent from a to b on both P_0 and P_1, even if b does not know which of T_0, T_1 is Byzantine, it can guess i with probability $\frac{1}{2}$ and accept the message M received on P_i. This implies that if Property (1) does not hold then $(\{T_0, T_1\}, 1/2)$-reliable communication from a to b is possible. The next theorem states that fault-restricted reliable communication from a to b for smaller values of ϵ is possible only if neither properties hold.

Theorem 2.11 ([1]) *Let $N = \langle V, E_{\mathrm{C}}, E_{\mathrm{A}} \rangle$ be a partially-authenticated network and $a, b \in V$ be a sender and receiver. For all $T_0, T_1 \subseteq V \setminus \{a, b\}$ it holds that:*

1. *If (T_0, T_1) is not an (a, b)-confusing pair, then $(\{T_0, T_1\}, \epsilon)$-reliable communication from a to b is possible for every $\epsilon > 0$.*
2. *If (T_0, T_1) is an (a, b)-confusing pair and $0 \leq \epsilon < \frac{1}{2}$, then $(\{T_0, T_1\}, \epsilon)$-reliable communication from a to b is* not *possible.*

The following theorem connects fault-restricted reliable communication with reliable communication. As mentioned before, there is a transformation from [1] that executes the fault-restricted protocol for every pair of sets $T_0, T_1 \subseteq V \setminus \{a, b\}$ and analyzes these executions to achieve reliable communication from a to b (see details in Section 6). Together with Theorem 2.11, this transformation gives an exact characterization when (t, ϵ)-reliable communication is possible:

Theorem 2.12 ([1]) *Let $N = \langle V, E_C, E_A \rangle$ be a partially-authenticated network and $a, b \in V$ be a sender and receiver. There is a (t, ϵ)-reliable communication protocol from a to b for every $\epsilon > 0$ if and only if for every $T_0, T_1 \subseteq V \setminus \{a, b\}$ of size at most t it holds that (T_0, T_1) is not an (a, b)-confusing pair.*

2.5 The Depth of Edges

Beimel and Franklin [1] used the level of edges in order to bound the round complexity of the protocol. The contribution of this paper is a more efficient protocol, and it starts with introducing the notion of the depth of an edge. We use the depth of edges in order to bound the round complexity of the protocol. The depth of an edge is at most the level of an edge, but it can be significantly smaller. Moreover, the level of edges can be as much as $\Omega(n)$ even for $t = 1$, whereas the depth on an edge can be at most t. The depth of an edge of level j is the number of levels $j' \leq j$ for which j' is the first level that has an avoiding path to b from some vertex in precisely one of T_0 and T_1. More formally,

Definition 2.13 (Depth of edges, paths, and networks) *Let $N = \langle V, E_C, E_A \rangle$ be a partially-authenticated network, $a, b \in V$ be a sender and receiver, and $T_0, T_1 \subseteq V \setminus \{a, b\}$ be a pair of sets. We inductively define a sequence of sets $B_j \subseteq T_0 \Delta T_1$ for $j \geq 0$, where $T_0 \Delta T_1 \stackrel{def}{=} (T_0 \cup T_1) \setminus (T_0 \cap T_1)$. First, we define $B_0 = \emptyset$. For every $j \geq 1$, we define B_j to be the set of all $z \in T_0 \Delta T_1$ for which the following properties hold:*

- *For every $0 \leq j' < j$ it holds that $z \notin B_{j'}$, and*
- *For the $i \in \{0, 1\}$ such that $z \in T_i$ there is a $T_{\bar{i}}$-avoiding path from z to b in G_{j-1}.*

We denote $\text{depth}(j) \stackrel{def}{=} |\{j' | B_{j'} \neq \emptyset, 1 \leq j' \leq j\}|$, *and say that an edge e is of depth d if* $\text{depth}(\text{level}(e)) = d$. *For a path P, we define* $\text{depth}(P) \stackrel{def}{=} \max\{\text{depth}(e) | e \in P\}$. *Finally, the depth of N is the maximal depth over all the edges in G^*.*

Note that the following three statements are equivalent: (1) $e \in G_C$, (2) the level of e is 0, and (3) the depth of e is 0. Therefore, the depth of a path P is 0 if and only if P is in G_C. For example, in the network N_2 described in Figure 3 we have $B_1 = \{t_0, t_1, t_2, t_3\}$. Hence, all of the authentication edges are of depth 1 and the depth of N_2 is 1. We next bound the depth of a partially-authenticated network.

Lemma 2.14 *Let $N = \langle V, E_C, E_A \rangle$ be a partially-authenticated network, $a \in V$ be a sender, $b \in V$ be a receiver, and $T_0, T_1 \subseteq V \setminus \{a, b\}$ such that $|T_1|, |T_0| \leq t$. If there is no honest path from a to b in $G_C = \langle V, E_C \rangle$ and G_C is $(t+1, a, b)$-connected, then the depth of N is at most t.*

Proof: Since G_C is $(t+1, a, b)$-connected, there are at least $t+1$ vertex-disjoint paths from a to b in G_C. Fix a set of such $t + 1$ paths. Since there is no honest path from a to b in G_C, none of these paths is honest and there is at least one suspected vertex on each one of them (recall that a vertex v is suspected if $v \in T_0 \cup T_1$). We consider the last suspected vertex on each of these paths. Since from each of these suspected vertices there is a path to b that has no other suspected vertices on it, $|B_1| \geq t + 1$. Thus, there are at most other $2t - (t + 1) = t - 1$ sets B_j for which $B_j \neq \emptyset$, and the depth of N is as asserted. □

2.6 Protocol SimpleSend

Protocol SIMPLESEND(M, u, v), described in Figure 2, transmits a message M on a path from u to v in G^*. For every authentication edge $\langle u', v' \rangle$ on the path from u to v it recursively calls SIMPLESEND(M, u', v') to transmit the message on a path from u' to v'. Protocol SIMPLESEND(M, u, v) does not achieve reliable communication from u to v, it only chooses the paths on which the message is sent. It is a preliminary

PROTOCOL SIMPLESEND(M, u, v)

Parameters: M – message, u – source, v – target.

Choose an avoiding path $u = v_0, v_1, \ldots, v_{\ell-1}, v_\ell = v$ from u to v in G^*.

For $i = 0$ to $\ell - 1$ do $(*\ v_i$ propagates the message to $v_{i+1}\ *)$

If $\langle v_i, v_{i+1}\rangle \in E_{\mathrm{C}}$ then v_i sends M to v_{i+1} on this edge

Otherwise, $\langle v_i, v_{i+1}\rangle \in E_{\mathrm{A}}$: vertex v_i executes SIMPLESEND(M, v_i, v_{i+1})

Endfor.

Figure 2: A protocol for sending a message from u to v.

version of Protocol SEND, discussed in Section 5. We will show that SEND is efficient if SIMPLESEND is efficient, and then we will use Protocol SEND as a sub-protocol of Protocol TRANSMIT, our protocol for fault-restricted reliable communication.

The description of Protocol SIMPLESEND from Figure 2 does not specify how an avoiding path is chosen. Such specification will be given in Section 4, after investigating the special structure of G^* in Section 3. We next give intuition for possible implementations of Protocol SIMPLESEND and their analysis. As observed in [1], since for every authentication edge $\langle u, v\rangle$ of level j in G^* there is an avoiding path from u to v of level at most $j-1$, transmitting a message on this path can be done by at most n transmissions on edges of level at most $j-1$, yielding, by simple induction, a protocol with round complexity $n^{O(n)}$.

The first property that we introduce is of paths that end in b. We prove that for every authentication edge $\langle u, v\rangle$ there is a path from both u and v to b which has at most one edge of each level. Concatenating the path from u to b with the path from b to v results in an avoiding path from u to v that has at most two edges of each level, yielding, by simple induction, a protocol with round complexity $2^{O(n)}$.

Both approaches do not consider the impact of the number of Byzantine processors on the round complexity of the protocol. The main contribution of this paper is the concept of depth. When we send a message from u to b we choose a path from u to b for which the depth of authentication edges does not increase as the path is traversed. We prove an upper bound on the round complexity of sending a message over an authentication edge that is exponential in the depth of the edge. Since the depth of an edge is at most t, the resulting protocol has round complexity $n^{O(t)}$. We next present an example demonstrating the choice of the paths in Protocol SIMPLESEND.

Example 2.15 Consider the partially-authenticated network N_2 described in Figure 3 in which $T_0 = \{t_0, t_2\}$ and $T_1 = \{t_1, t_3\}$. Assume we want to send a message over the authentication edge $\langle a, u_1\rangle$. We can choose the avoiding paths on which we send messages in the following way: To send the message on $\langle a, u_1\rangle$, we use the T_1-avoiding path $\langle a, t_0, b, t_2, u_2, u_1\rangle$. This requires a recursive send on the authentication edge $\langle u_2, u_1\rangle$. To send a message over $\langle u_2, u_1\rangle$, we use the T_0-avoiding path $\langle u_2, u_3, t_3, b, t_1, u_1\rangle$, which requires a recursive send on the authentication edge $\langle u_2, u_3\rangle$. For the edge $\langle u_2, u_3\rangle$ we use the T_1-avoiding path $\langle u_2, t_2, b, u_4, u_3\rangle$ which requires a recursive send on the authentication edge $\langle u_4, u_3\rangle$. Finally, to send a message on $\langle u_4, u_3\rangle$ we use the T_0-avoiding path $\langle u_4, b, t_3, u_3\rangle$ which does not require any recursive calls.

Although this example may seem artificial, we show in Lemma 4.1 that every partially-authenticated network has the structure of N_2, and then we analyze the transmission costs in this structure. We show that

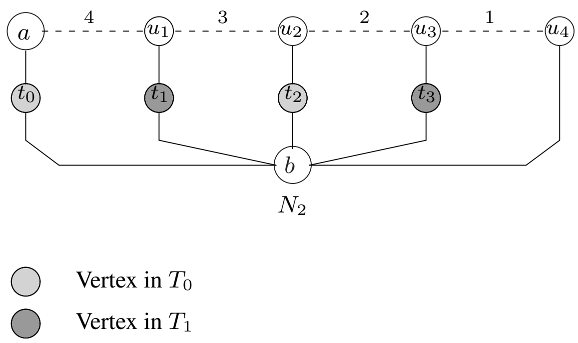

Figure 3: the choice of the paths in Protocol SIMPLESEND. The numbers indicate the level of authentication edges.

these costs are exponential in the depth. The somewhat technical proofs in Section 3 provide the tools that enable the construction of such a structure.

The following lemma, which is used in Section 4, proves that the round complexity of transmitting M from u to v is equal to the round complexity of transmitting M from v to u for all $u, v \in V$. This implies that the round complexity of the protocol could be analyzed regardless of the direction in which M is sent.

Lemma 2.16 *Let $N = \langle V, E_{\mathrm{C}}, E_{\mathrm{A}} \rangle$ be a partially-authenticated network. For all $u, v \in V$, if there is an implementation of the protocol* SIMPLESEND(M, u, v) *that terminates after ℓ rounds, then there is an implementation of* SIMPLESEND(M, v, u) *that terminates after ℓ rounds.*

Proof: The lemma is proved by induction on the depth of the recursive calls. The base case for paths in G_{C} is trivial. For the induction step, let $P_{u,v}$ be the avoiding path chosen in the execution of SIMPLESEND(M, u, v) and note that the reverse path $P_{v,u}$ is an avoiding path as well. Using the induction hypothesis for the authentication edges $\langle u', v' \rangle$ on $P_{u,v}$, we conclude that SIMPLESEND(M, u', v') and SIMPLESEND(M, v', u') require the same number of rounds to terminate. Since SIMPLESEND(M, u, v) terminates after ℓ rounds, SIMPLESEND(M, v, u) terminates after ℓ rounds as well, and the induction follows. □

The fact that Protocol SIMPLESEND is symmetric with respect to the sender and the receiver does not imply that reliable communication is symmetric with respect to the sender and the receiver. The reason is that the alert mechanism added in Protocol TRANSMIT is not symmetric.

3 Properties of the Graph G^*

In this section we analyze the graph G^*. In particular, we show that paths which end in b have additional properties. Our protocol utilizes this analysis to more efficiently transmit a message over an authentication edge. In this section we fix a partially-authenticated network $N = \langle V, E_{\mathrm{C}}, E_{\mathrm{A}} \rangle$, a sender a, a receiver b, and sets $T_0, T_1 \subseteq V \setminus \{a, b\}$ of suspected vertices. Thus, the graph G^* is well defined.

3.1 Monotonous Paths

The first property that we introduce is monotonous paths. Specifically, monotonous paths have only one authentication edge of each level. As explained before, monotonous paths imply a protocol with round complexity $2^{O(n)}$.

Definition 3.1 (Monotonous paths) *A path P is* monotonous *if for all authentication edges e_1 and e_2 in P, whenever e_2 precedes e_1 in the path P, then* level(e_2) *is strictly larger then* level(e_1).

For example, the path $\langle a, u_1, u_2, u_3, u_4, b\rangle$ in N_2 (described in Figure 3) is a monotonous path. Note that P being monotonous implies that the first authentication edge e on P has the highest level over all of the other edges in P. Hence, the level of P is determined by the level of this edge and vice versa. Also, if P is of level 0 (i.e., P is a path in G_C), then P is monotonous.

Lemma 3.2 *For every $w \in V$, if there is an honest path from w to b in G^* of level j, then there is a monotonous honest path from w to b of level at most j.*

Proof: The lemma is proved by induction on j. The base case for $j = 0$ follows from the observation that every path of level 0 is monotonous. For the induction step, assume that for every $w \in V$, if there is an honest path from w to b of level at most $j-1$, then there is a monotonous honest path from w to b of level at most $j-1$. Now, let $P_{w,b}$ be an honest path from w to b of level j. Since the level of $P_{w,b}$ is at least 1, there is at least one authentication edge on $P_{w,b}$, and its level is at most j. Denote the first authentication edge on $P_{w,b}$ by $e = \langle u, v\rangle$. If there is an honest path $P_{u,b}$ from u to b of level at most $j-1$, then concatenating the prefix $\langle w, \ldots, u\rangle$ of $P_{w,b}$ with $P_{u,b}$ yields an honest path $\langle w, \ldots, u\rangle, P_{u,b}$ from w to b of level at most $j-1$, and, by the induction hypothesis, there is a monotonous honest path from w to b of level at most $j-1$. Otherwise, by Property (3) in the definition of the graph G_j, the level of e must be exactly j and there is an honest path from v to b with level at most $j-1$. By the induction hypothesis there is a monotonous honest path $P_{v,b}$ from v to b of level at most $j-1$, which implies that the path $\langle w, \ldots, u\rangle, \langle u, v\rangle, P_{v,b}$ is a monotonous honest path from v to b of level j, and the induction follows. □

3.2 Left Edges and Left Paths

We further introduce left edges. We also define the second property of paths that end in b, namely, left paths. Left paths are used in our implementation of Protocol SIMPLESEND, as they ensure paths with lower depth, resulting with an efficient implementation of Protocol SIMPLESEND.

Definition 3.3 (Left and right edges and paths) *An authentication edge $e = \langle u, v\rangle$ of level j is* left *if the following properties hold:*

1. *There is an honest path from v to b of level at most $j-1$.*
2. *There is an avoiding path $P_{u,v}$ from u to v of level at most $j-1$ with at least one suspected vertex on this path, where for the leftmost suspected vertex t on $P_{u,v}$, the prefix $\langle u, \ldots, t\rangle$ of $P_{u,v}$ is in G_C.*

An edge $\langle u, v\rangle$ is right *if and only if $\langle v, u\rangle$ is left. A path P is left if every authentication edge on P is left.*

For an illustration of a left edge see Figure 4, case (1). As another example, consider the authentication edge $\langle a, u_1\rangle$ of level 4, shown in the graph N_2 (see Figure 3). This edge is left since $\langle a, t_0, b, t_2, u_2, u_1\rangle$ is an avoiding path from a to u_1 with t_0 as its leftmost suspected vertex, and $\langle u_1, u_2, u_3, u_4, b\rangle$ is an honest path from u_1 to b of level 3. Definition 2.6 of the graph G^* implies that there must be an honest path from either u or v to b of level at most $j-1$, and an avoiding path $P_{u,v}$ from u to v of level at most $j-1$. Property (2) in Definition 3.3 requires, in addition, that a suspected vertex must appear on $P_{u,v}$ before any authentication edges that are on $P_{u,v}$. Informally, this vertex provides a shortcut path to b that enables sending messages more efficiently.

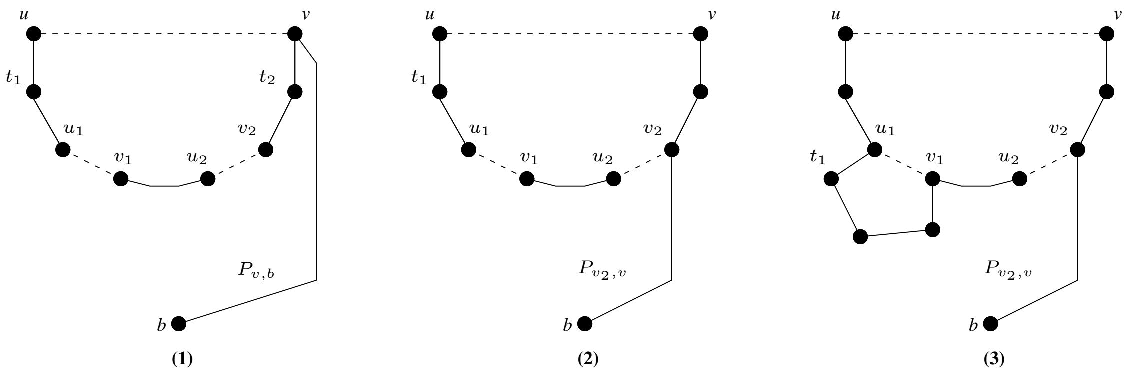

Figure 4: The three cases in the proof of Lemma 3.4.

Lemma 3.4 *Every authentication edge in G^* is either left or right.*

Proof: Let $e = \langle u, v\rangle$ be an authentication edge of level j. We prove by induction on j that e is either left or right. For every edge of level 1 there is an avoiding path from u to v in G_{C}. Remark 2.8 implies that there must be at least one suspected vertex on this path. If there is an honest path from v to b of level 0, then e is left. Otherwise, there is an honest path from u to b of level 0 and e is right.

Assume that every authentication edge of level at most $j-1$ is either left or right. The induction step for j is as follows: Let $e = \langle u, v\rangle$ be an edge of level j. If there is an avoiding path from u to v in G_{C}, then similar arguments to those in the base case hold, and e is either left or right. Otherwise, let P be an avoiding path from u to v chosen with the minimal level among the avoiding paths from u to v. The path P must contain at least one authentication edge. Denote the level of P by j', where $1 \leq j' \leq j-1$, and let $e_1 = \langle u_1, v_1\rangle$ and $e_2 = \langle u_2, v_2\rangle$ be the leftmost and rightmost authentication edges on P respectively (e_1 and e_2 can be the same edge). Denote $P_{u_1,b}$ and $P_{v_2,b}$ to be honest minimal level paths from u_1 and v_2 to b respectively. Define $P_{u,v} \stackrel{\text{def}}{=} \langle u, \ldots, u_1\rangle, P_{u_1,b}, P_{b,v_2}, \langle v_2, \ldots, v\rangle$. Note that, for some $i \in \{0, 1\}$, the path $P_{u,v}$ is a $T_{\bar{i}}$-avoiding path from u to v of level at most j'. Since $P_{u_1,b}, P_{b,v_2}$ is an honest path, if there is a suspected vertex on $P_{u,v}$, then it can appear on $\langle u, \ldots, u_1\rangle$ or $\langle v_2, \ldots, v\rangle$, but not on $P_{u_1,b}, P_{b,v_2}$. There are three cases to consider (see Figure 4), and in each case we construct the paths proving that e is either left or right.

1. There are vertices $t_1, t_2 \in T_i$ such that t_1 is a suspected vertex in $\langle u, \ldots, u_1\rangle$, and t_2 is a suspected vertex in $\langle v_2, \ldots, v\rangle$: Note that there is $w \in \{u, v\}$ for which there is an honest path from w to b of level at most $j-1$. If $w = u$, then e is right. Otherwise, $w = v$ and e is left. See Figure 4, case (1).

2. There is a vertex $t_1 \in T_i$ such that t_1 is a suspected vertex in $\langle u, \ldots, u_1\rangle$, and $\langle v_2, \ldots, v\rangle$ avoids $T_0 \cup T_1$: In this case the prefix $\langle u, \ldots, t_1\rangle$ of $P_{u,v}$ is in G_{C}. Also, the honest paths $\langle v, \ldots, v_2\rangle$ and $P_{v_2,b}$ make an honest path $\langle v, \ldots, v_2\rangle, P_{v_2,b}$ from v to b of level at most $j-1$, which implies that e is left. See Figure 4, case (2). If there is a suspected vertex in $\langle v_2, \ldots, v\rangle$, and $\langle u, \ldots, u_1\rangle$ avoids $T_0 \cup T_1$, then by symmetric arguments e is right.

3. Both $\langle u, \ldots, u_1\rangle$, and $\langle v_2, \ldots, v\rangle$ avoid $T_0 \cup T_1$: By the induction hypothesis, each of e_1 and e_2 is either left or right. If e_1 is right and e_2 is left, then, by Definition 3.3, the level of the path $P_{u_1,b}, P_{b,v_2}$ is at most $j'-1$. See Figure 5. This implies that $P_{u,v}$ is an avoiding path from u to v of level at most $j'-1$, in contradiction to the choice of P with the minimal level. Hence, either e_1 is left or e_2 is right. If e_1 is left, then, by the induction hypothesis, there is an avoiding path P_{u_1,v_1} from u_1 to v_1 of level

at most $j' - 1$, and there is a prefix $\langle u_1, \ldots, t_1 \rangle$ of P_{u_1,v_1} where $t_1 \in T_0 \Delta T_1$ is the leftmost suspected vertex on P_{u_1,v_1}. Note that $\langle u_1, \ldots, t_1 \rangle$ is a path in G_{C}. We construct an avoiding path P' from $P_{u,v}$ by replacing e_1 with P_{u_1,v_1}. See Figure 4, case (3). There is a prefix of P' in which t_1 is the leftmost suspected vertex. Moreover, the level of P' is at most j'. Finally, since $\langle v, \ldots, v_2 \rangle, P_{v_2,b}$ is an honest path from v to b of level at most $j - 1$, the edge e is left. If e_2 is right, then by symmetric arguments e is right.

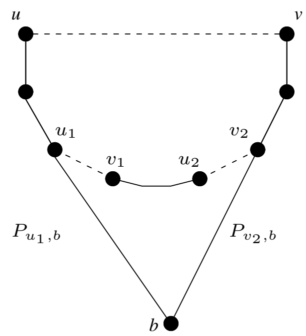

Figure 5: Case 3 in the proof of Lemma 3.4.

Thus, the induction follows. □

The next lemma combines the property of monotonous paths with the property of left paths. Our protocol uses paths that are both monotonous and left to transmit messages efficiently.

Lemma 3.5 *For every left authentication edge $\langle u, v \rangle$ of level j, there is a left, monotonous, honest path from v to b of level at most $j - 1$.*

Proof: We prove the lemma by induction on j. Let $e = \langle u, v \rangle$ be a left authentication edge of level $j \geq 1$. For the base case of the induction, the level of e is 1 and e is left. By Definition 3.3 there is an honest path from v to b of level 0. Since this path is in G_{C}, it is left and monotonous as well.

Assume that the induction hypothesis holds for every authentication edge e of level at most j. For the induction step, let $e = \langle u, v \rangle$ be a left authentication edge of level $j + 1$. By Definition 3.3 there is an honest path from v to b of level at most j. Therefore, there is a minimal $j' \leq j$ for which there is an honest path from v to b of level j'. By Lemma 3.2, there is a monotonous, honest path $P_{v,b}$ from v to b of level j'. We show that there is a left, monotonous, honest path from v to b of level j'. If $j' = 0$, then, by Definitions 3.3 and 3.1, $P_{v,b}$ is a left, monotonous, honest path from v to b. Otherwise, there is a leftmost authentication edge $\langle u', v' \rangle$ on $P_{v,b}$ of level at most j', and by Lemma 3.4, the edge $\langle u', v' \rangle$ is either left or right. If $\langle u', v' \rangle$ is right, then there is an honest path $P_{u',b}$ from u' to b of level at most $j' - 1$, and $\langle v, \ldots, u' \rangle, P_{u',b}$ is an honest path from v to b of level at most $j' - 1$, in contradiction to the choice of $P_{v,b}$ with a minimal level. Therefore, $\langle u', v' \rangle$ is a left edge and $\mathrm{level}(\langle u', v' \rangle) \leq j'$. By the induction hypothesis, there is a left, monotonous, honest path $P_{v',b}$ from v' to b of level at most $\mathrm{level}(\langle u', v' \rangle) - 1$. Therefore, $\langle v, \ldots, u' \rangle, \langle u', v' \rangle, P_{v',b}$ is a left, monotonous, honest path from v to b of level at most j, as asserted. □

In the next lemma we make the first link between depth and left edges.

Lemma 3.6 *For every left authentication edge $e = \langle u, v \rangle$ of depth d there is an avoiding path from u to b of depth at most $d - 1$.*

Proof: Let $e = \langle u, v \rangle$ be a left authentication edge of level j and depth d. Since e is left, there is an avoiding path $P_{u,v}$ from u to v of level at most $j-1$ and there is an honest path $P_{v,b}$ from v to b of level at most $j-1$. Hence, the path $P_{u,v}, P_{v,b}$ is an avoiding path from u to b of level at most $j-1$ and there is a leftmost suspected vertex $z \in T_i \setminus T_{\bar{i}}$ on this path for some $i \in \{0,1\}$. This implies that there is also an avoiding path from z to b of level at most $j-1$ and, since $z \notin T_0 \cap T_1$, the vertex z is in B_k for some $k \leq j$.

Note that the prefix $\langle u, \ldots, z \rangle$ of $P_{u,v}$ avoids $T_{\bar{i}}$. Since $z \in B_k$, there is a $T_{\bar{i}}$-avoiding path $P_{z,b}$ from z to b in G_{k-1}. Also, $B_k \neq \emptyset$ implies that $\text{depth}(k-1) = \text{depth}(k) - 1 \leq \text{depth}(j) - 1 = d - 1$, and we conclude that $\langle u, \ldots, z \rangle, P_{z,b}$ is an avoiding path from u to b of depth at most $d-1$. □

4 Efficient Implementation of Protocol SimpleSend

In this section we utilize the properties of G^* from Section 3 to describe an efficient implementation of Protocol SIMPLESEND. This implementation uses paths of depth $d-1$ to send messages over edges of depth d, which motivates us to express transmission costs in terms of depth. Moreover, the depth of edges in N is at most t, and we prove that the running time of Protocol SIMPLESEND is $n^{O(t)}$, which implies that the protocol is polynomial whenever the number of Byzantine processors is constant.

To specify an implementation for Protocol SIMPLESEND we specify how an avoiding path is chosen in each recursive call. That is, we describe how a message is sent over an authentication edge. This implementation completes the specification of the protocol and it is known to all of the processors in the network. Hence, every processor in the network can execute its part of the protocol.

The following lemma proves an upper bound on the number of rounds required to send a message over an authentication edge of depth $d+1$ by describing an implementation that achieves this bound. During the transmission of a message over an authentication edge of depth $d+1$, this implementation sends the message over paths of depth d, which may be honest or avoiding. The notation $\text{cost}(d)$ denotes the number of rounds required to send a message over a path of depth at most d in this implementation, taken as the maximal over the paths of depth at most d that are used by the protocol in this implementation. Since a path of depth 0 can have at most n edges, all of which are communication edges, we conclude that $\text{cost}(0) \leq n$. Our next lemma is used to upper bound $\text{cost}(d)$ as a function of $\text{cost}(d-1)$. As before, we fix a partially authenticated network $N = \langle V, E_{\text{C}}, E_{\text{A}} \rangle$, a sender a, a receiver b, and sets $T_0, T_1 \subseteq V \setminus \{a, b\}$ of suspected vertices. Thus, the graph G^* is well defined.

Lemma 4.1 *If $\langle u, v \rangle$ is a left authentication edge of depth d and $P_{v,b}$ is a left, monotonous, honest path from v to b with m authentication edges of depth d (and any number of authentication edges of a lower depth), then there is an implementation of* SIMPLESEND(M, u, v) *that terminates after at most* $2(m+1) \cdot \text{cost}(d-1) + mn$ *rounds.*

Proof: Since $\langle u, v \rangle$ is left, Lemma 3.6 implies that there is an avoiding path $P_{u,b}$ from u to b of depth at most $d-1$. By the definition of $\text{cost}(d-1)$, a message M sent from u to b by SIMPLESEND(M, u, b) arrives at b after at most $\text{cost}(d-1)$ rounds. By induction on m, which is the number of authentication edges of depth d on $P_{v,b}$, we prove that SIMPLESEND(M, u, v) terminates after at most $2(m+1) \cdot \text{cost}(d-1) + mn$ rounds. For the base case, since $m = 0$, the honest path $P_{v,b}$ is of depth at most $d-1$. Lemma 2.16 guarantees that SIMPLESEND(M, b, v) requires the same number of rounds as SIMPLESEND(M, v, b). Hence, a message M sent from b to v by SIMPLESEND(M, b, v) arrives at v after at most $\text{cost}(d-1)$ rounds. Therefore, the path $P_{u,b}, P_{b,v}$ is an avoiding path of depth at most $d-1$, and a message M sent by SIMPLESEND(M, u, v) from u to v through b, arrives at v after at most $2 \cdot \text{cost}(d-1)$ rounds.

Assume the induction hypothesis holds for every $m' \leq m$. For the induction step, let $\langle u, v \rangle$ be a left edge of depth d and fix $P_{v,b}$ to be a left, monotonous, honest path from v to b with $m+1$ authentication edges of depth d. Denote $P_{v,b} \stackrel{\text{def}}{=} P_{v,v_{m+1}}, P_{v_{m+1},b}$ where $P_{v,v_{m+1}} = \langle v \rightsquigarrow u_1, v_1 \rightsquigarrow u_2, v_2, \ldots, u_{m+1}, v_{m+1}\rangle$ is a prefix of $P_{v,b}$ with $m+1$ authentication edges $e_\ell = \langle u_\ell, v_\ell\rangle$ for every $1 \leq \ell \leq m+1$ (the notation $\rightsquigarrow$ stands for an honest path in G_{C}), and $P_{v_{m+1},b}$ is a suffix of $P_{v,b}$ of depth at most $d-1$.

Consider the path $P_{v,v_{m+1}}$. This path is also a left, monotonous, honest path from v to v_{m+1}. By Lemma 3.6 there is an avoiding path $P_{u_\ell,b}$ from u_ℓ to b of depth at most $d-1$ for every $1 \leq \ell \leq m+1$ (see Figure 6). This implies that there is an avoiding path $P_{b,v_\ell} \stackrel{\text{def}}{=} P_{b,u_{\ell+1}}, \langle u_{\ell+1} \rightsquigarrow v_\ell\rangle$ from b to v_ℓ of depth at most $d-1$ for every $1 \leq \ell \leq m$, where $P_{b,u_{\ell+1}}$ is the reverse path of $P_{u_{\ell+1},b}$. Let i be such that $P_{u,b}$ is T_i-avoiding. There are two cases:

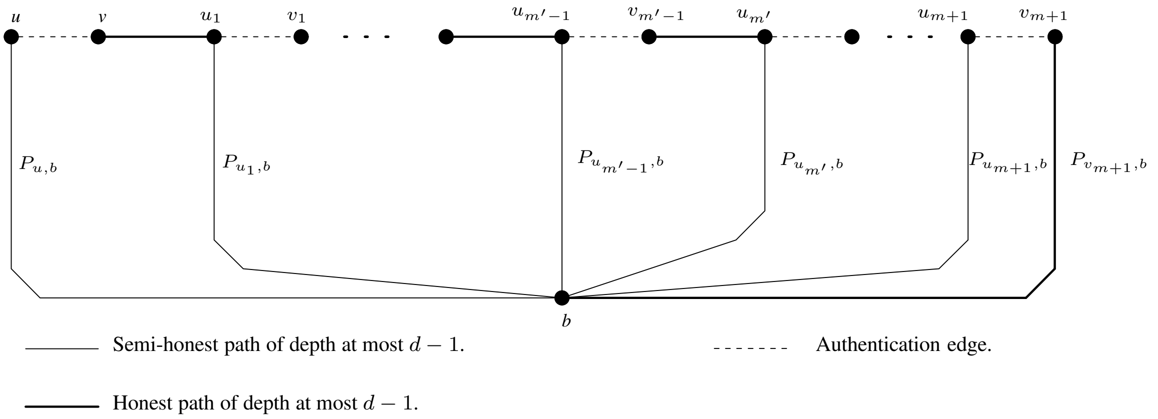

Figure 6: The paths in the induction step of the proof of Lemma 4.1.

First Case. For every $1 \leq \ell \leq m+1$, the path $P_{u_\ell,b}$ from u_ℓ to b is $T_{\bar{i}}$-avoiding: For every $1 \leq \ell \leq m$ consider the path $P_{u_\ell,b}, P_{b,v_\ell}$. This is an avoiding path from u_ℓ to v_ℓ of level at most $d-1$. For the edge e_{m+1}, recall that $P_{v_{m+1},b}$ is an honest path from v_{m+1} to b of depth at most $d-1$. Hence, there is an avoiding path $P_{u_{m+1},b}, P_{b,v_{m+1}}$ from u_{m+1} to v_{m+1} of depth at most $d-1$. This implies that $\textsc{SimpleSend}(M, u_\ell, v_\ell)$ terminates after at most $2 \cdot \text{cost}(d-1)$ rounds for every $1 \leq \ell \leq m+1$.

Consider the avoiding path from u to v:

$$P_{u,b}, P_{b,v_{m+1}}, \langle v_{m+1}, u_{m+1}, \ldots, v_1, u_1 \rightsquigarrow v\rangle.$$

A message M sent from u on $P_{u,b}, P_{b,v_{m+1}}$ arrives at v_{m+1} after at most $2 \cdot \text{cost}(d-1)$ rounds. Since there are at most n communication edges on $P_{v,v_{m+1}}$, each with transmission cost of 1 round, a message M sent from u to v by $\textsc{SimpleSend}(M, u, v)$ arrives at v after at most

$$2 \cdot \text{cost}(d-1) + (m+1) \cdot 2 \cdot \text{cost}(d-1) + n \quad = \quad 2(m+2) \cdot \text{cost}(d-1) + n$$

rounds.

Second Case. There is an ℓ, where $1 \leq \ell \leq m+1$, for which the $P_{u_\ell,b}$ is a T_i-avoiding path: Let m' be the minimal index such that $P_{u_{m'},b}$ is T_i-avoiding. By the choice of m', the path $P_{u_\ell,b}$ avoids $T_{\bar{i}}$ for every $1 \leq \ell \leq m'-1$. As in the previous case, $P_{b,v_\ell} = P_{b,u_{\ell+1}}, \langle u_{\ell+1} \rightsquigarrow v_\ell\rangle$ is a $T_{\bar{i}}$-avoiding path from b to v_ℓ for every $1 \leq \ell \leq m'-2$. Therefore, the path $P_{u_\ell,b}, P_{b,v_\ell}$ is an avoiding path from u_ℓ to v_ℓ of level at most $d-1$ for every $1 \leq \ell \leq m'-2$, which implies that $\textsc{SimpleSend}(M, u_\ell, v_\ell)$ terminates after at most $2 \cdot \text{cost}(d-1)$ rounds for every $1 \leq \ell \leq m'-2$.

Consider the avoiding path from u to v:

$$P_{u,b}, P_{b,u_{m'}}, \langle u_{m'} \rightsquigarrow v_{m'-1}, u_{m'-1}, \ldots, v_1, u_1 \rightsquigarrow v\rangle.$$

By the induction hypothesis for the edge $\langle u_{m'-1}, v_{m'-1}\rangle$ it holds that SIMPLESEND$(M, u_{m'-1}, v_{m'-1})$ terminates after at most $2 \cdot [(m+1) - m' + 1] \cdot \text{cost}(d-1) + [(m+1) - m']n$ rounds. Since there are at most n communication edges on $P_{v,u_{m'-1}}$, each with transmission cost of 1 round, we conclude that a message M sent from u by SIMPLESEND(M, u, v) arrives at v after at most

$$2 \cdot \text{cost}(d-1) + (m'-2) \cdot 2 \cdot \text{cost}(d-1) + 2[m - m' + 2] \cdot \text{cost}(d-1) + [m + 1 - m']n + n$$

rounds. Since this expression is less than $2(m+2) \cdot \text{cost}(d-1) + (m+1)n$, the induction follows. □

We further describe the implementation of Protocol SIMPLESEND from Lemma 4.1 to specify how a message is sent on a path of depth d. Although this path can be either honest or avoiding, we treat it as an avoiding path, because an honest path is an avoiding path. The following lemma describes how these paths are chosen and upper bounds cost(d) for our implementation. Towards this goal, define $\delta_0 \overset{\text{def}}{=} 0$ and $\delta_d \overset{\text{def}}{=} |\{j|\text{depth}(j) = d\}|$ for every $d \geq 1$. That is, δ_d is the number of levels in which the depth of edges is d. Clearly, if the depth of the network is d, then $\delta_0 + \ldots + \delta_d \leq n$.

Lemma 4.2 *There is an implementation of Protocol* SIMPLESEND *for which* $\text{cost}(d) \leq (d+1) \cdot n \prod_{k=0}^{d}(\delta_k + 1)^2$ *for every depth* $d \geq 0$.

Proof: We prove the upper bound by induction on d. For the base case, any path of depth 0 is a path in G_{C}, which implies that cost$(0) \leq n$ and the inequality holds. Assume the induction hypothesis holds for every $d' < d$. For the induction step, let P be an avoiding path from w to b, chosen with a minimal level over these paths, and denote depth$(P) = d$ and level$(P) = j$. Since $d \geq 1$, there is at least one authentication edge on P. Let $\langle u, v\rangle$ be the leftmost authentication edge on P, and let $\langle w, \ldots, u\rangle$ be a prefix of P in G_{C}. If $\langle u, v\rangle$ is right, then there is an honest path P' from u to b of level at most $j-1$, which implies that $\langle w, \ldots, u\rangle, P'$ is an avoiding path from w to b of level at most $j-1$; a contradiction to the choice of P with a minimal level. Therefore, $\langle u, v\rangle$ is left, and, by Lemma 3.5, there is a monotonous, honest path $P_{v,b}$ from v to b of depth d and level at most $j-1$.

Consider the avoiding path $P_{w,b} \overset{\text{def}}{=} \langle w, \ldots, u\rangle, \langle u, v\rangle, P_{v,b}$. Note that the path $\langle u, v\rangle, P_{v,b}$ is a left, monotonous, honest path from u to b of depth at most d. By the definition of δ_d and since $\langle u, v\rangle, P_{v,b}$ is monotonous, there are $m \leq \delta_d$ authentication edges of level d on $\langle u, v\rangle, P_{v,b}$. Let $\langle u_m, v_m\rangle = \langle u, v\rangle$ and define $P_{u,b} \overset{\text{def}}{=} \langle u_m, v_m, \ldots, u_1, v_1\rangle, P_{v_1,b}$ where $e_\ell = \langle u_\ell, v_\ell\rangle$ is an authentication edge of depth d for every $1 \leq \ell \leq m$, and $P_{v_1,b}$ is a path from v_1 to b of depth at most $d-1$.

By the definition of cost$(d-1)$, a message M sent from v_1 to b by SIMPLESEND(M, v_1, b) arrives at b after at most cost$(d-1)$ rounds. In addition, by Lemma 4.1 a message M sent from u_ℓ to v_ℓ by SIMPLESEND(M, u_ℓ, v_ℓ) arrives at v_ℓ after at most $2 \cdot \ell \cdot \text{cost}(d-1) + (\ell-1)n$ rounds for every $m \geq \ell \geq 1$. Finally, since there are at most n communication edges on P_{w,v_1}, we conclude that a message M sent from w by SIMPLESEND(M, w, b) arrives at b after at most $\sum_{\ell=1}^{m}[2 \cdot \ell \cdot \text{cost}(d-1) + (\ell-1)n] + \text{cost}(d-1) + n$ rounds, where $m \leq \delta_d$. Thus:

$$\begin{aligned}
\text{cost}(d) &\leq \sum_{\ell=1}^{m}[2 \cdot \ell \cdot \text{cost}(d-1) + (\ell-1)n] + \text{cost}(d-1) + n \\
&\leq \sum_{\ell=1}^{\delta_d} \ell[2 \cdot \text{cost}(d-1) + n] + \text{cost}(d-1) + n
\end{aligned}$$

$$\begin{aligned}
&\leq \left(\frac{(\delta_d+1)\delta_d}{2}+1\right)[2\cdot \mathrm{cost}(d-1)+n] \\
&\leq (\delta_d+1)^2[d\cdot n\prod_{k=0}^{d-1}(\delta_k+1)^2+n] \qquad (1)\\
&\leq (d+1)\cdot n\prod_{k=0}^{d}(\delta_k+1)^2.
\end{aligned}$$

The inequality in (1) is implied by the induction hypothesis. □

We have specified how Protocol SIMPLESEND sends a message over an authentication edge, and how honest and avoiding paths are chosen. This implementation is well defined, which implies that cost(d) is well defined. We use these results to upper bound the round complexity of a message transmission from a to b on an honest path.

Lemma 4.3 *There is an implementation of* SIMPLESEND(M, a, b) *that terminates after at most* $n^2 \cdot \left(\frac{2n}{t}\right)^{2t}$ *rounds.*

Proof: If there is an honest path from a to b in G_{C}, then SIMPLESEND(M, a, b) terminates after at most n rounds. Otherwise, by Lemma 2.14 the depth of G^* is at most t, which implies, by Lemma 4.2, that there is an implementation of SIMPLESEND(M, a, b) that transmits M over an honest path from a to b and terminates after at most cost(t) $\leq (t+1)\cdot n\prod_{k=0}^{t}(\delta_k+1)^2$ rounds. Let j be the highest level of an edge in G^*, and notice that $\delta_0+\delta_1+\ldots+\delta_t = j$ and that $j \leq n$. Also, $\prod_{k=0}^{t}(\delta_k+1)^2$ is maximal when $\delta_1=\delta_2=\ldots=\delta_t=\frac{j}{t}\leq\frac{n}{t}$. Finally, since $t\leq n-2$, we conclude that:

$$\begin{aligned}
\mathrm{cost}(t) &\leq (t+1)n\prod_{k=0}^{t}(\delta_k+1)^2 \;\leq\; n^2\prod_{k=1}^{t}\left(\frac{n}{t}+1\right)^2 \\
&= n^2\left(\frac{n+t}{t}\right)^{2t} \leq n^2\left(\frac{2n}{t}\right)^{2t}.
\end{aligned}$$

□

5 Fault-Restricted Reliable Communication

In this section we present Protocol SEND, which is an extended version of Protocol SIMPLESEND, and we explain why the two protocols have the same round complexity. We also introduce Protocol TRANSMIT which solves the problem of fault-restricted reliable communication by executing Protocol SEND, and we prove that its round complexity is at most $n+1$ times the round complexity of Protocol SEND. Protocol TRANSMIT borrows ideas from Protocol TRANSMIT of [1]; however, our protocol is simpler and has lower message complexity. In Section 6 we use this protocol to obtain a protocol that solves the problem of (t, ϵ)-reliable communication.

We first describe Protocol SEND and then we explain how Protocol TRANSMIT uses it to achieve fault-restricted reliable communication. Protocol SEND(M, P), described in Figure 7, transmits a message M on a path P. The message M is propagated in the same fashion as in Protocol SIMPLESEND: if $\langle u, v\rangle$ is a communication edge, then u simply propagates M to v. Otherwise, $\langle u, v\rangle$ is an authentication edge, u uses the shared key $k_{u,v}$ to authenticate M, and the authenticated message is sent to v by calling SEND recursively. If v does not receive a valid authenticated message from u (at the appropriate time), it recalls this fact by setting a flag which is later sent to b in Protocol TRANSMIT. Thus, Protocol SEND does not give any reliability guarantees, but the flags that it maintains are used for reporting to b which of T_0, T_1 is Byzantine.

Protocol SEND uses fixed paths that are known to all of the processors in the network. That is, these paths are explicitly specified by Protocol SEND. Let $\langle u, v\rangle$ be an authentication edge of level j. To send a message from u to v, the protocol fixes an avoiding path $\text{PATH}(u, v)$ from u to v. The only requirement from $\text{PATH}(u, v)$ is that its level is at most $j - 1$. For every $w \in V$ with an honest path from w to b, to send a message from w to b the protocol fixes an honest path $\text{PATH\_TO\_}b(w)$ from w to b. The only requirement from $\text{PATH\_TO\_}b(w)$ is that it is a monotonous honest path, whose level is minimal over these paths. Under these requirements, the correctness of the protocol is proved.

By Lemma 4.1 and Lemma 4.2, the paths chosen by Protocol SIMPLESEND satisfy the requirements for the paths fixed by Protocol SEND. Authenticating messages and maintaining a flag do not change the round complexity of Protocol SEND with respect to Protocol SIMPLESEND. Hence, the round complexity of Protocol SEND equals to the round complexity of Protocol SIMPLESEND.

Protocol TRANSMIT(M, a, b), described in Figure 8, *reliably* transmits a message M from a to b. This protocol proceeds in cycles: in the first cycle M is sent from a to b on three paths: P_0, P_1, and P. The paths P_0 and P_1 are paths from a to b in the communication graph that avoid T_0 and T_1 respectively, and M is propagated on these paths from a to b. The path P is a $T_0 \cup T_1$-avoiding path in G^*. The message M is propagated from a to b on P by executing the recursive Protocol SEND(M,P). Once this execution terminates, if M arrives on P, then b accepts it. Otherwise, we prove in Lemma 5.2 that b can analyze the execution of TRANSMIT(M, a, b) and determine an index $i \in \{0, 1\}$ for which T_i is Byzantine. This enables b to conclude that P_i is Byzantine-free, therefore it accepts the message which arrives on P_i.

In the first cycle of Protocol TRANSMIT, Protocol SEND(M, P) is executed and then alert calls are invoked. These calls, executed in parallel in the second cycle, send the flag values to b. Alert calls are recursive calls to Protocol SEND, and they may trigger additional cycles. However, if (and only if) all of the alert calls that are executed in a cycle send messages over paths in G_{C}, then no additional cycle of alert calls is executed, and Protocol TRANSMIT terminates. In Lemma 5.1 we show that such a cycle exists.

The following lemma proves that the round complexity of TRANSMIT is at most $n + 1$ times the round complexity of SIMPLESEND. As before, we fix a partially-authenticated network $N = \langle V, E_{\mathrm{C}}, E_{\mathrm{A}}\rangle$, a sender a, a receiver b, and sets $T_0, T_1 \subseteq V \setminus \{a, b\}$ of suspected vertices. Thus, the graph G^* is well defined.

Lemma 5.1 *If there is an implementation of Protocol* SIMPLESEND(M, w, b) *that terminates after at most τ rounds for every $w \in V$, then there is an implementation of Protocol* TRANSMIT(M, w, b) *that terminates after at most $\tau \cdot (n + 1)$ rounds for every $w \in V$.*

Proof: In every cycle of Protocol TRANSMIT there are parallel executions of Protocol SEND. If an authentication edge participates in an execution of Protocol SEND, then alert calls are invoked and another cycle is executed.

To show that the execution of TRANSMIT(M, w, b) terminates after at most $n + 1$ cycles, we show that sending a message to b on a path of level j using Protocol SEND may only trigger executions of Protocol SEND in which alerts are sent to b on paths of level at most $j - 1$. Since the level of the paths on which alerts are transmitted decreases from one cycle to another, a cycle is eventually reached in which no authentication edges are used and no alerts are invoked; hence Protocol TRANSMIT terminates.

Consider the execution of SEND(M, P) and let $j = \text{level}(P)$. We show that $\text{level}(\text{PATH\_TO\_}b(z)) < j$ for any execution of SEND$(M, \text{PATH_TO_}b(z))$ in the next cycle. For every authentication edge $\langle u, v\rangle$ of level i that participates in SEND(M, P), the protocol fixes the path $\text{PATH}(u, v)$ from u to v of level at most $i - 1$ to propagate the message from u to v. Moreover, by the construction of G^* there is an honest path of level at most i from both u and v to b. From Lemma 3.2 it follows that for every authentication edge $\langle u, v\rangle$

Protocol Send(M,P)

PARAMETERS:

M – a message, $P = v_0, v_1, \ldots, v_\ell$ – a path in G^*.

For $i = 0$ to $\ell - 1$ do   ($*$ M is propagated on P $*$)

Let M' be the message received at v_i.

If no message is received, then $M' \leftarrow$ "error".

If $\langle v_i, v_{i+1} \rangle \in E_{\mathrm{C}}$, then v_i propagates M' to v_{i+1} on this edge.

Otherwise, $\langle v_i, v_{i+1} \rangle \in E_{\mathrm{A}}$:

1. v_i executes $\textsc{Send}(\langle M', \textsc{auth}(M', k_{v_i, v_{i+1}}) \rangle, \textsc{path}(v_i, v_{i+1}))$.
2. If v_{i+1} received $\langle \hat{M}, \hat{\alpha} \rangle$ such that $\hat{\alpha} \neq \textsc{auth}(\hat{M}, k_{v_i, v_{i+1}})$,
   then v_{i+1} sets $\mathrm{FLAG}_{v_i, v_{i+1}} \leftarrow$ FALSE, and $M \leftarrow$ "error".

Endfor.

Figure 7: A sub-protocol for sending a message on a path in G^*.

Protocol Transmit(M,a,b)

Parameters: M – a message, a – sender, and b – receiver.

Initialization: $\mathrm{cycle} \leftarrow 0$, and for every $u, v \in V$ set $\mathrm{FLAG}_{u,v} \leftarrow$ TRUE.

Send M from a to b on a T_0-avoiding path in G_{C}.

Send M from a to b on a T_1-avoiding path in G_{C}.

Execute $\textsc{Send}(M, \textsc{path\_to\_}b(a))$

Repeat

$\mathrm{cycle} \leftarrow \mathrm{cycle} + 1$

Let R be the set of all authentication edges $\langle u, v \rangle$ for which u has sent an authenticated message to v in the previous cycle.

For each $\langle u, v \rangle \in R$ do ($*$ In parallel $*$)

$\textsc{Send}(\langle \mathrm{cycle}, u, v, \mathrm{FLAG}_{u,v} \rangle, \textsc{path\_to\_}b(v))$ ($*$ This is an alert call $*$)

Until $R = \emptyset$

Figure 8: A protocol for *reliable* message transmission from a to b.

of level i there is a monotonous honest path from w to b of level at most i. Hence, authentication edges of level $i \leq j-1$ that participate in the execution of $\text{SEND}(M, P)$ trigger alert calls on paths of level at most $j-1$. Moreover, since P is a monotonous path, only the first edge on this path, denoted $\langle u_1, v_1 \rangle$, is of level j, and the suffix of P is an honest monotonous path from v_1 to b of level at most $j-1$. This implies that the alert call invoked by v_1 is on a path of level at most $j-1$. We conclude that authentication edges that participate in the transmission of a message from w to b on a path of level j trigger alert calls on paths of level at most $j-1$.

Therefore, if $\text{level}(\text{PATH\_TO\_}b(w)) = j$, then in cycle 1 of $\text{TRANSMIT}(M, w, b)$ there are alert transmissions on paths of level at most $j-1$. Repeating this argument we conclude that in cycle at most $j+1$, there are alert transmissions on paths of level 0. That is, alert messages are sent to b on paths in the communication graph, which implies that no authentication edges participate in this cycle. Since there are at most $n+1$ cycles, each of which requires at most τ rounds to terminate, Protocol $\text{TRANSMIT}(M, w, b)$ terminates after at most $\tau \cdot (n+1)$ rounds. □

The next lemma is used to prove the correctness of the protocol.

Lemma 5.2 *For every $w \in V$ with an honest path from vertex w to vertex b and for every message M, if during the execution of* $\text{TRANSMIT}(M, w, b)$ *the adversary has not forged any authenticated messages that were sent from u to v, for every authentication edge $\langle u, v \rangle$ such that both u and v are honest, then either b accepts M or b learns a set that contains all Byzantine vertices.*

Proof: Let P be the honest path from w to b fixed by Protocol TRANSMIT. Since P is Byzantine-free, if P is in G_C then clearly b accepts M. Otherwise, authentication edges on P trigger a second cycle in which alert calls are transmitted. In particular, for every authentication edge $\langle u, v \rangle$ on P, the message M is propagated by v if and only if it arrives with valid authentication. Otherwise, the "error" message is propagated. Note that it is impossible that v accepts a message that was not sent by u because we assume that the adversary has not forged any authenticated messages that were sent from u to v (since both u and v are honest). Hence, if b accepts a message $M' \neq$ "error", then it must be that $M = M'$. If b receives "error", it infers that there is at least one authentication edge that participated in the protocol for which a message sent from one end of this edge was not received with valid authentication on its other end. The rest of the proof shows that b can find an authentication edge from which it can determine a set that contains all Byzantine vertices.

The execution of $\text{TRANSMIT}(M, w, b)$ proceeds in cycles. Alert transmissions are invoked at the end of each cycle, and executed in the next cycle. For every $\text{FLAG}_{u,v}$ sent from v to b, since the alert message contains the cycle number, u, and v, no two alert messages are identical. As before, if b receives the message $\text{FLAG}'_{u,v} \neq$ "error", then it must be that $\text{FLAG}'_{u,v} = \text{FLAG}_{u,v}$. We say that a cycle is successful if, at the end of the cycle, b receives all of the messages that were sent to it during the cycle. Since messages in the last cycle are sent over honest paths in G_C, vertex b receives all of these messages which implies that there is at least one successful cycle.

Consider the first successful cycle and denote it by C. This cycle cannot be the first because $M' =$ "error". Also, if all of the flag values received by b in cycle C are true, then v has received a valid authenticated message from u for every authentication edge $\langle u, v \rangle$ that participated in the previous cycle. This implies that the cycle preceding C is successful; a contradiction to the choice of C. Therefore, b receives all of the flags that are sent to it in C and at least one of these flags has a false value.

Let $\langle u, v \rangle$ be an authentication edge that participates in cycle C for which b receives $\text{FLAG}_{u,v} =$ FALSE, chosen with the minimal level over these edges. This implies that there was a round in which v did not receive a valid authenticated message from u. Recall that to send a message from u to v the protocol fixes an avoiding path $\text{PATH}(u, v)$ from u to v, which is known to all of the processors. By the choice of C, when this cycle ends, b receives $\text{FLAG}_{u',v'}$ for every authentication edge $\langle u', v' \rangle$ on the avoiding path $\text{PATH}(u,v)$. Moreover, for every authentication edge $\langle u',v'\rangle$ on the avoiding path $\text{PATH}(u,v)$, since level$(\langle u',v'\rangle) <$ level$(\langle u,v\rangle)$, by the choice of $\langle u,v\rangle$ it must be that $\text{FLAG}_{u',v'} = \text{TRUE}$. Since b knows that v did not receive a message with valid authentication from u, and that v' received a valid authenticated message from u' for every authentication edge $\langle u',v'\rangle$ on the avoiding path $\text{PATH}(u,v)$, the receiver b can conclude that there is a Byzantine processor on $\text{PATH}(u,v)$. Finally, $\text{PATH}(u,v)$ is a $T_{\bar{i}}$-avoiding path for some $i \in \{0,1\}$ and b detects that T_i contains all Byzantine vertices. □

The next theorem proves that Protocol TRANSMIT achieves $(\{T_0,T_1\},\epsilon)$-reliable communication from a to b if authentication edges that participate in its execution use an $(\ell, \frac{\epsilon}{\ell\cdot n^2})$-authentication scheme, for $\ell = 2n^3 \cdot \left(\frac{2n}{t}\right)^{2t}$. By [16], this can be achieved if authentication keys of length $O(\ell(n+\log\frac{1}{\epsilon}) + \log|M|)$ are used.

Theorem 5.3 *Let $N = \langle V, E_{\text{C}}, E_{\text{A}}\rangle$ be a partially-authenticated network, $a \in V$ be a sender, $b \in V$ be a receiver, and $T_0, T_1 \subseteq V \setminus \{a,b\}$ be a pair of sets. If (T_0,T_1) is not an (a,b)-confusing pair, then, for every $\epsilon > 0$, Protocol TRANSMIT(M,a,b) is a $(\{T_0,T_1\},\epsilon)$-reliable protocol which terminates after at most $O(n^3 \cdot \left(\frac{2n}{t}\right)^{2t})$ rounds provided that authentication edges that participate in the execution of the protocol use an $(\ell, \frac{\epsilon}{\ell\cdot n^2})$-authentication scheme, where $\ell = 2n^3 \cdot \left(\frac{2n}{t}\right)^{2t}$.*

Proof: Since (T_0,T_1) is not an (a,b)-confusing pair, for every $i \in \{0,1\}$ there is a T_i-avoiding path P_i from a to b in G_{C}, and there is an honest path from a to b in G^*. Hence, the execution of Protocol TRANSMIT is well defined. Applying Lemma 5.1 and Lemma 4.3, this protocol terminates after at most $2n^3 \cdot \left(\frac{2n}{t}\right)^{2t}$ rounds.

Assume that during the execution of TRANSMIT(M,a,b) the adversary has not forged any authenticated messages that were sent from u to v, for every authentication edge $\langle u,v\rangle$ such that both u and v are honest. Thus, by Lemma 5.2, either b learns M or b detects that T_i is Byzantine for some $i \in \{0,1\}$, in which case b accepts the message which arrives on P_i. Since in both cases b learns M, we only need to show that the probability that the adversary has forged at least one authenticated message during the execution of TRANSMIT(M,a,b) is at most ϵ. The number of times that an authentication key is used, taken as the maximal over all of the authentication keys, is at most $\ell = 2n^3 \cdot \left(\frac{2n}{t}\right)^{2t}$ (the number of rounds in the protocol). Hence, for every authentication edge $\langle u,v\rangle$ that participates in the execution of SEND(M,P), there are at most ℓ transmissions between u and v, and for each of these, the probability that one of the processors accepts a message that was not sent by the other is at most $\frac{\epsilon}{\ell\cdot n^2}$. Since there are at most n^2 such edges, and since there are at most ℓ transmissions on each such edge, the probability that the adversary forges an authentication of a message sent to an honest vertex is at most $\ell \cdot n^2 \cdot \frac{\epsilon}{\ell\cdot n^2} = \epsilon$. □

6 Reliable Communication

In this section we employ a transformation from [1] that uses Protocol TRANSMIT to achieve (t,ϵ)-reliable communication. We show that (t,ϵ)-reliable communication is efficient whenever t is constant. By [1], this result also translates to private communication.

The next theorem proves that there is a protocol for (t,ϵ)-reliable communication from a to b if authentication edges that participate in its execution use an $(\ell, \epsilon/(\binom{n}{t}\cdot\ell\cdot n^2))$-authentication scheme, for $\ell = 2n^3 \cdot \left(\frac{2n}{t}\right)^{2t}$. By [16], this can be achieved if authentication keys of length $\binom{n}{t}\cdot O(\ell(n+\log\frac{1}{\epsilon}) + \log|M|)$ are used.

Theorem 6.1 *Let N be a partially-authenticated network and $a, b \in V$ be a sender and receiver. If (T_0, T_1) is not an (a,b)-confusing pair in N for every $T_0, T_1 \subseteq V \setminus \{a,b\}$ of size t, then, for every $\epsilon > 0$, there is a protocol for (t, ϵ)-reliable communication from a to b with round complexity at most $2n^3 \cdot \left(\frac{2n}{t}\right)^{2t} \leq 2^{O(n)}$, provided that authentication edges that participate in the execution of the protocol use an $(\ell, \epsilon/(\binom{n}{t} \cdot \ell \cdot n^2))$-authentication scheme, where $\ell = 2n^3 \cdot \left(\frac{2n}{t}\right)^{2t}$.*

Proof: To achieve (t, ϵ)-reliable communication from a to b we follow the transformation from [1]: For every $T_0, T_1 \subseteq V \setminus \{a,b\}$ of size t, we execute Protocol $\textsc{Transmit}(M, a, b)$, assuming that one of T_0, T_1 contains all Byzantine processors. We execute these $\binom{n}{t}^2$ executions in parallel and by Theorem 5.3 the round complexity of this protocol is at most $2n^3 \cdot \left(\frac{2n}{t}\right)^{2t} \leq 2^{O(n)}$ rounds.

Notice that the precondition that one of T_0, T_1 contains all Byzantine processors may not hold in some of the $\binom{n}{t}^2$ executions. However, there is a set T of size t that contains all Byzantine processors, and if for every $T_1 \subseteq V \setminus \{a,b\}$ of size t, the adversary has not forged any valid authentication on a message that was sent on an edge $\langle u, v\rangle$, where both u and v are honest, during the execution of $\textsc{Transmit}(M, a, b)$ for (T, T_1), then, by Lemma 5.2, in each of these $\binom{n}{t}$ executions b accepts M. Hence, b chooses a set T' such that for every $T_1 \subseteq V \setminus \{a,b\}$ of size t it holds that b accepts the same message M' in the execution of $\textsc{Transmit}(M, a, b)$ for (T', T_1). The receiver b accepts M' as the message sent by a. In particular, b accepts M' in the execution of Protocol $\textsc{Transmit}(M, a, b)$ for the pair T', T. Since b accepts M in the execution of Protocol $\textsc{Transmit}(M, a, b)$ for the pair T, T', it must be that $M = M'$, and b accepts the message M sent by a.

We only need to show that the probability that the adversary has forged any valid authentication on a message that was sent on an edge $\langle u, v\rangle$, where both u and v are honest, during the above $\binom{n}{t}$ executions of $\textsc{Transmit}(M, a, b)$ is at most ϵ. Since new authentication keys are selected for every execution of Protocol $\textsc{Transmit}$, the number of times that an authentication key is used, taken as the maximal over all of the authentication keys in all of the $\binom{n}{t}$ executions is at most $\ell = 2n^3 \cdot \left(\frac{2n}{t}\right)^{2t}$. Hence, for every authentication edge $\langle u, v\rangle$ that participates in an execution of $\textsc{Transmit}(M, a, b)$, there are at most ℓ transmissions between u and v, and for each of these, the probability that one of the processors accepted a message that was not sent by the other is at most $\epsilon/(\binom{n}{t} \cdot \ell \cdot n^2)$. Since there are at most n^2 such edges, and since there are at most ℓ transmissions on each such edge, the probability that the adversary has forged at least one authenticated message in at least one of the $\binom{n}{t}$ executions is at most $\binom{n}{t} \cdot \ell \cdot n^2 \cdot \epsilon/(\binom{n}{t} \cdot \ell \cdot n^2) = \epsilon$. $\square$

The next corollary proves that the protocol presented in Theorem 6.1 has polynomial round complexity and polynomial message complexity if t is constant.

Corollary 6.2 *Let $N = \langle V, E_{\mathrm{C}}, E_{\mathrm{A}}\rangle$ be a partially-authenticated network, $a, b \in V$ be a sender and receiver, and t be a constant. If (T_0, T_1) is not an (a,b)-confusing pair for all $T_0, T_1 \subseteq V \setminus \{a,b\}$ of size t, then, for every $\epsilon > 0$, there is a protocol for (t,ϵ)-reliable communication from a to b with polynomial round and message complexity, provided that authentication edges that participate in the execution of the protocol use an $(\ell, \epsilon/(\binom{n}{t} \cdot \ell \cdot n^2))$-authentication scheme, where $\ell = 2n^3 \cdot \left(\frac{2n}{t}\right)^{2t}$.*

Proof: If t is constant, the protocol described in Theorem 6.1 achieves (t,ϵ)-reliable communication in polynomial round complexity. This protocol executes, in parallel, $\binom{n}{t}^2 < n^{O(t)}$ copies of Protocol $\textsc{Transmit}$. Since in every round of each execution of Protocol $\textsc{Transmit}$ there are at most n^2 parallel executions of Protocol $\textsc{Send}$ (also called alert transmissions), at any round of the (t, ϵ)-reliable protocol there are at most $n^2 \cdot n^{O(t)}$ messages in transit. Therefore, it remains to show that the length of any of these messages is polynomial.

We show that for every $w \in V$ with an honest path P from w to b, the length of any message sent during the execution of $\textsc{Send}(M, P)$ is at most $O(|M| + n^2 \log(\frac{1}{\epsilon}))$. Note that in any round of Protocol $\textsc{Send}$, there is at most one message in transit, which is propagated on edges in G^*. If u propagates the message to v and $\langle u, v\rangle \in G_{\mathrm{C}}$, the message does not increase in length. However, if $\langle u, v\rangle$ is an authentication edge, then an authentication tag is attached to the message before it is sent to v on $\textsc{path}(u, v)$. Moreover, it is possible that the authenticated message is authenticated again, this time by an authentication edge $\langle u', v'\rangle$ on $\textsc{path}(u, v)$. However, since level($\textsc{path}(u, v)$) $<$ level($\langle u, v\rangle$) and since the level of an edge can be at most n, this process can repeat itself at most n times before the outermost authentication tag is removed.

By [16], the tag produced by an (ℓ, ϵ')-authentication scheme on a message of length s has length $O(\log(\frac{1}{\epsilon'}))$, which implies that the authenticated message has length at most $s + O(\log \frac{1}{\epsilon'})$. Repeating this process n times yields an authenticated message of length at most $s + n \cdot O(\log \frac{1}{\epsilon'})$. Since messages sent by Protocol $\textsc{Send}(M, P)$ have length either $O(\log n)$ in the case of alert call or $|M|$, the length of any message sent during the execution of $\textsc{Send}(M, P)$ is at most $O(|M| + \log n + n \cdot \log \frac{1}{\epsilon'})$.

Since our (t, ϵ)-reliable protocol uses an $(\ell, \epsilon/(\binom{n}{t} \cdot \ell \cdot n^2))$-authentication scheme, the length of any message sent during the execution of $\textsc{Send}(M, P)$ is at most $O(|M| + n \cdot \log(\binom{n}{t} \cdot \ell \cdot n^2/\epsilon))$. Finally, since $\binom{n}{t}$ and $\ell \leq 2^{O(n)}$, it holds that $\log(\binom{n}{t} \cdot \ell \cdot n^2) = O(n)$. Therefore, the length of any message sent during the execution of the (t, ϵ)-reliable protocol is at most $O(|M| + n^2 \cdot \log \frac{1}{\epsilon})$, as required. $\square$

6.1 Privacy from Reliability

A protocol for private transmission from a to b was offered in [12, 1], under the assumption that communication channels are reliable and *private*. That is, if $\langle u, v\rangle$ is a communication edge and both u and v are honest processors, then the adversary cannot change or delete a message sent from u to v, nor can it insert a message on the channel or *learn anything about the message being sent*. It was proved in [1] that private communication from a to b is possible if and only if reliable communication is possible both from a to b and from b to a. The protocol of [12, 1] for private communication executes Protocol $\textsc{Transmit}$ twice (once from a to b and once from b to a) and the rest of the protocol of [12, 1] is efficient. Thus, an efficient implementation of Protocol $\textsc{Transmit}$ from a to b and from b to a implies an efficient implementation of a private communication protocol from a to b.

Theorem 6.3 *Let N be a partially-authenticated network, $a, b \in V$ be a sender and receiver, and t be an integer. If t-reliable communication is possible both from a to b and from b to a, then there is a t-private protocol from a to b with round complexity at most $n^3 \cdot \left(\frac{2n}{t}\right)^{2t} \leq 2^{O(n)}$ rounds.*

In addition, if private communication from a to b is possible, then a can use the protocol of [12, 1] for private communication to privately transmit a key to b. Once a and b share a secret key, further private communication between a and b is efficient. That is, once a and b share a private key, b can privately send a message to a by first encrypting and authenticating the message and then sending it to a on $t + 1$ vertex-disjoint paths in the communication graph. Since one of the encrypted and authenticated messages must arrive at a on at least one of the communication paths, a can verify its authenticity, decrypt it, and retrieve the original message from b.

7 Is Reliable Communication Symmetric?

In this section we give a simple characterization of reliable communication in the presence of one Byzantine processor and show that $(1, \epsilon)$-reliable communication is symmetric. Furthermore, we show that these results do not apply for $t \geq 2$.

7.1 Characterizing Reliable Communication with One Byzantine Processor

In this section, we consider $(1, \epsilon)$-reliable communication. We prove that if G_C is connected, then a simple necessary and sufficient condition for $(1, \epsilon)$-reliable communication is that the communication graph G_{C} is $(2, a, b)$-connected, and that $G = G_{\mathrm{C}} \cup G_{\mathrm{A}}$ is $(3, a, b)$-connected.

Lemma 7.1 *Let $N = \langle V, E_{\mathrm{C}}, E_{\mathrm{A}} \rangle$ be a partially-authenticated network and $a, b \in V$ be a pair of vertices. If G_{C} is connected and $(2, a, b)$-connected, and $G = G_{\mathrm{C}} \cup G_{\mathrm{A}}$ is $(3, a, b)$-connected, then for every $t_0, t_1 \in V \setminus \{a, b\}$ the pair $(\{t_0\}, \{t_1\})$ is not an (a, b)-confusing pair.*

Proof: Fix any $t_0, t_1 \in V \setminus \{a, b\}$. If there is a path in G_{C} that avoids $\{t_0, t_1\}$, then by Property (1) of Definition 2.10 the pair $(\{t_0\}, \{t_1\})$ is not an (a, b)-confusing pair. Otherwise, every path from a to b in G_{C} has a suspected vertex, t_0 or t_1, on it. Since G_{C} is $(2, a, b)$-connected, there are two vertex-disjoint paths from a to b in G_{C}, and there must be exactly one suspected vertex on each of these paths. Hence, there is a path $P_{t_0,b}$ from t_0 to b that avoids t_1 and there is a path $P_{t_1,b}$ from t_1 to b that avoids t_0. Also, G_{C} is connected and for every $u \in V$ there is a path $P_{u,b}$ from u to b. If $P_{u,b}$ is not honest, then there is an index $i \in \{0, 1\}$ such that the prefix $\langle u, \ldots, t_i \rangle$ of $P_{u,b}$ avoids $t_{\bar{i}}$ and $\langle u, \ldots, t_i \rangle, P_{t_i,b}$ is an avoiding path from u to b. We conclude that for every $u \in V$ there is an avoiding path $P_{u,b}$ from u to b.

Since G is $(3, a, b)$-connected, there is a path P from a to b in G that avoids $\{t_0, t_1\}$. We will prove that P is also a path in G^*, which implies by Definition 2.10 that $(\{t_0\}, \{t_1\})$ is not an (a, b)-confusing pair. Assume towards contradiction that P is not in G^*. Hence, there is an authentication edge $e = \langle u, v \rangle$ and an honest path P' such that $\langle u, v \rangle, P'$ is a suffix of P, the edge e is not in E^*, and P' is in E^*.

We next check when $e \in E^*$ according to Definition 2.6 of G^*. Since P avoids $\{t_0, t_1\}$, the vertices u, v are not in $\{t_0, t_1\}$ and Property (1) holds. Since there is an avoiding path $P_{u,b}$ from u to b, the path $P_{u,b}, P'$ is an avoiding path from u to v and, therefore, Property (2) holds. Finally, the path P' is an honest path from v to b and Property (3) holds. Hence, $e \in E^*$; a contradiction. Thus, P is in G^* and $(\{t_0\}, \{t_1\})$ is not an (a, b)-confusing pair in G. □

Theorem 7.2 *Let $N = \langle V, E_{\mathrm{C}}, E_{\mathrm{A}} \rangle$ be a partially-authenticated network, $a, b \in V$ be a pair of vertices, V' be the connected component of b in G_{C}, and E'_{A} (respectively, E'_{C}) be the set of authentication (respectively, communication) edges that connect vertices in V'. Define $G' = \langle V', E'_{\mathrm{C}} \cup E'_{\mathrm{A}} \rangle$. Then, $(1,\epsilon)$-reliable communication from a to b is possible for every $\epsilon > 0$ if and only if G_{C} is $(2, a, b)$-connected and G' is $(3, a, b)$-connected.*

Proof: By Lemma 7.1, the pair (T_0, T_1) is not an (a, b)-confusing pair for all $T_0, T_1 \subseteq V \setminus \{a, b\}$ of size at most 1. Therefore, by Theorem 2.12, there is a $(1, \epsilon)$-reliable protocol from a to b for every $\epsilon > 0$.

On the other hand, if the conditions that G_{C} is $(2, a, b)$-connected and G' is $(3, a, b)$-connected do not hold, then there is a confusing pair in G', which implies by Theorem 2.12 that (t, ϵ)-reliable communication is not possible for every $\epsilon < \frac{1}{2}$. □

Since the conditions of Theorem 7.2 are symmetric with respect to a and b, reliable communication is symmetric for $t = 1$.

Corollary 7.3 *Let $N = \langle V, E_{\mathrm{C}}, E_{\mathrm{A}} \rangle$ be a partially-authenticated network and $a, b \in V$ be a pair of vertices. Then, $(1, \epsilon)$-reliable communication from a to b is possible for every $\epsilon > 0$ if and only if $(1, \epsilon)$-reliable communication from b to a is possible for every $\epsilon > 0$.*

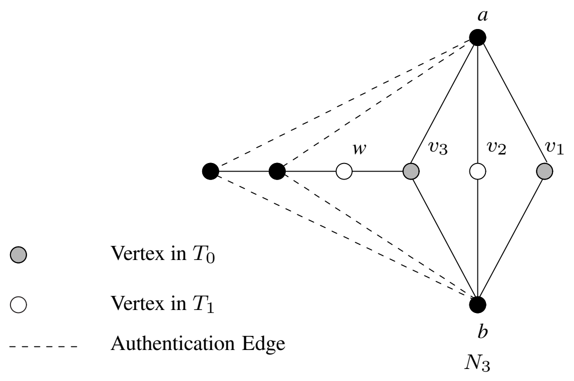

Figure 9: Confusing pairs for $t = 2$.

7.2 Reliable Communication is Not Symmetric for $t \geq 2$

In this section we show that the simple characterization for the case $t = 1$ cannot be applied to the case $t \geq 2$. Moreover, we show that if $t \geq 2$, then (t, ϵ)-reliable communication for every $\epsilon > 0$ is not symmetric.

Lemma 7.4 *For every $t \geq 2$, there a partially-authenticated network $N = \langle V, E_{\mathrm{C}}, E_{\mathrm{A}} \rangle$ such that*

- *$G_C = \langle V, E_{\mathrm{C}} \rangle$ is a connected graph,*
- *G_C is $(t + 1, a, b)$-connected,*
- *$G = G_{\mathrm{C}} \cup G_{\mathrm{A}}$ is $(2t + 1, a, b)$-connected,*
- *however, (t, ϵ)-reliable communication from a to b in N is impossible with $\epsilon < \frac{1}{2}$.*

Proof: For $t = 2$, consider N_3 and the suspected sets described in Figure 9. There is no avoiding path from u to v, for every authentication edge $\langle u, v \rangle$. By Property (2) of Definition 2.6, the graph G^* does not contain any authentication edges. Since G^* is the communication graph G_C, and since there is no honest path from a to b in G^*, by Definition 2.10 of a confusing pair, the pair (T_0, T_1) is an (a, b)-confusing pair in G^*, which implies that $(2, \epsilon)$-reliable communication from a to b in N_3 is impossible with $\epsilon < \frac{1}{2}$. For $t > 2$, consider the partially-authenticated network N_4 described in Figure 10. In N_4 the communication graph is $(2t - 1, b, a)$-connected and the union of the communication graph with the authentication graph is $(2t + 1, b, a)$-connected. Yet, we prove in Theorem 7.5 that (t, ϵ)-reliable communication from b to a is impossible with $\epsilon < \frac{1}{2}$. $\square$

Beimel and Franklin [1] showed an example where fault-restricted reliable communication is possible from a to b, but is impossible from b to a. However, in their example (t, ϵ)-reliable communication for every $\epsilon > 0$ is impossible in both directions. We present a stronger example in which (t, ϵ)-reliable communication from a to b is possible for every $\epsilon > 0$, but impossible from b to a with $\epsilon < \frac{1}{2}$.

Theorem 7.5 *For every $t \geq 2$, there is a partially-authenticated network $N = \langle V, E_{\mathrm{C}}, E_{\mathrm{A}} \rangle$ and vertices $a, b \in V$ such that (t, ϵ)-reliable communication from a to b in N is possible for every $\epsilon > 0$ and reliable communication from b to a in N is impossible with $\epsilon < \frac{1}{2}$.*

Proof: Consider the network N_4 described in Figure 10 with $V = \{a, b, u_1, \ldots, u_4, v_1, \ldots, v_{2t}\}$, $P_1 \stackrel{\text{def}}{=} \langle a, u_4, u_3, b \rangle$, and $P_2 \stackrel{\text{def}}{=} \langle a, u_2, u_1, b \rangle$.[4] We first show that (T_0, T_1) is not an (a, b)-confusing pair in N_4 for

[4] The edge $\langle v_1, v_3 \rangle$ was missing in the preliminary version [2] of this paper, and the proof in [2] is incorrect.

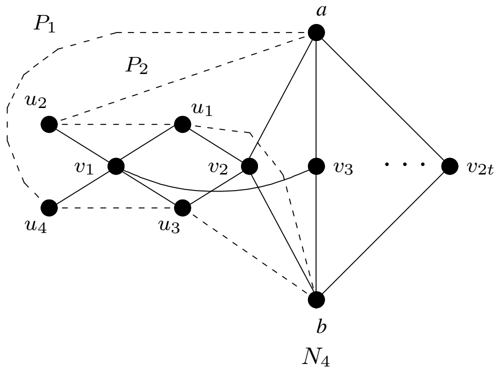

Figure 10: The partially-authenticated network N_4 in which reliable communication is possible from a to b, but impossible from b to a.

all $T_0, T_1 \subseteq V \setminus \{a, b\}$ of size t. Fix any $T_0, T_1 \subseteq V \setminus \{a, b\}$ with size t, and let G_{C} be the communication graph of N. If there is an honest path from a to b in G_{C}, then there is an honest path from a to b in G^*, which implies that (T_0, T_1) is not an (a, b)-confusing pair. Otherwise, the vertices $v_2, \ldots, v_{2t} \in T_0 \cup T_1$ and there are two cases:

1. $v_2 \in T_0 \cap T_1$: Since $|T_0| + |T_1| = 2t$ and $v_2 \in T_0 \cap T_1$, the size of $T_0 \cup T_1$ is at most $2t - 1$. Since $v_2, \ldots, v_{2t}$ are $2t - 1$ suspected vertices, no other vertices, namely u_1, u_2, u_3, u_4, v_1, are suspected and $v_3 \notin T_0 \cap T_1$ (otherwise, $|T_0| + |T_1| > 2t$). Thus, the path $\langle u_3, v_1, v_3, b\rangle$ is an avoiding path from u_3 to b in G_0, and $\langle u_3, b\rangle \in E_1$. Furthermore, the path $\langle u_4, v_1, u_3\rangle$ is an avoiding path from u_4 to u_3 in G_1 and there is an honest path $\langle u_3, b\rangle$ from u_3 to b in G_1, and $\langle u_4, u_3\rangle \in E_2$. Finally, the path $\langle u_4, u_3, b, v_3, a\rangle$ is an avoiding path from u_4 to a in G_2 and there is an honest path $\langle u_4, u_3, b\rangle$ from u_4 to b in G_2, and $\langle u_4, a\rangle \in E_3$. Therefore, P_1 is an honest path from a to b in G^*, and (T_0, T_1) is not an (a, b)-confusing pair.

2. $v_2 \notin T_0 \cap T_1$: Since $|T_0 \cup T_1| \leq 2t$ and $v_2, \ldots, v_{2t}$ are suspected, at most one of the vertices u_1, u_2, u_3, u_4, v_1 is a suspected vertex. Hence, either P_1 or P_2 is an honest path. Assume, without loss of generality, that P_1 is an honest path. Since $v_2 \notin T_0 \cap T_1$, the path $\langle u_3, v_2, b\rangle$ is an avoiding path from u_3 to b in G_{C}, and $\langle u_3, b\rangle \in E_1$. Since $v_1 \notin T_0 \cap T_1$ (otherwise, $|T_0| + |T_1| > 2t$), the path $\langle u_4, v_1, u_3\rangle$ is an avoiding path from u_4 to u_3. Also, there is an honest path $\langle u_3, b\rangle$ from u_3 to b in G_2, and therefore $\langle u_4, u_3\rangle \in E_2$. Finally, in G_2 the path $\langle u_4, u_3, b, v_2, a\rangle$ is an avoiding path from u_4 to a and there is an honest path $\langle u_4, u_3, b\rangle$ from u_4 to b, which implies that $\langle u_4, a\rangle \in E_3$. Hence, P_1 is an honest path from a to b in G^*, and therefore (T_0, T_1) is not an (a, b)-confusing pair.

We conclude that for every $T_0, T_1 \subseteq V \setminus \{a, b\}$ of size t there is an honest path from a to b in G^*, which implies by Theorem 2.12 that (t, ϵ)-reliable communication from a to b is possible for every $\epsilon > 0$.

We now show that (t, ϵ)-reliable communication from b to a is impossible with $\epsilon < \frac{1}{2}$. Fix $T_0 = \{v_1, v_{t+2}, \ldots, v_{2t}\}$ and $T_1 = \{v_2, v_3, \ldots, v_{t+1}\}$. We prove that (T_0, T_1) is a confusing pair in G^* with respect to (b, a) by showing that no authentication edges are added to G^*. First, consider the edge $\langle u_4, a\rangle$. Any path from a to u_4 in G_{C} passes through $v_1 \in T_0$ and through either v_2 or v_3, both in T_1. Thus, there is no avoiding path from u_4 to a in G_0 and $\langle a, u_4\rangle \notin E_1$. Symmetric arguments imply that $\langle a, u_2\rangle \notin E_1$. Moreover, any other authentication edge is not added to E_1 since there is no honest path from any of its endpoints to a. To conclude, there is no honest path from b to a in G^*, and (T_0, T_1) is a (b, a)-confusing pair in G^*, which, by Theorem 2.12, implies that (t, ϵ)-reliable communication from b to a is impossible for every $\epsilon < \frac{1}{2}$. $\square$

This asymmetry result is somewhat surprising because both communication and authentication edges are symmetric. The asymmetry stems from the topology of G, which can be asymmetric with respect to a and b. Recall that constructing G^* from a to b requires honest paths to b, while constructing G^* from b to a requires honest paths to a. These requirements are asymmetric, and the resulting G^* from a to b is different than the resulting G^* from b to a.

8 Conclusions and Open Problems

In this paper we considered reliable message transmission between a pair of processors in partially-authenticated, synchronous networks, that is, networks where some processors are connected by communication channels and some pairs of processors share authentication keys. Beimel and Franklin [1] characterize the partially-authenticated networks where a can reliably transmit a message to b. Our main result is a more efficient protocol than the protocol of [1]. The complexity of our protocol is $(n/t)^{O(t)}$, where n is the number of processors in the network and t is an upper bound on the number of Byzantine processors in the network. Specifically, our protocol is polynomial when the number of Byzantine processors is $O(1)$, and for every t its round complexity is bounded by $2^{O(n)}$. The same improvements hold for reliable and private communication.

We also give a simple characterization for reliable communication against one Byzantine processor and show that reliable communication is symmetric for $t = 1$. On the other hand, we show that reliable communication is not symmetric for $t \geq 2$. That is, there is a partially-authenticated network (i.e., a combination of a communication graph and an authentication graph) for which reliable communication is possible from a to b, but is not possible from b to a. This result is somewhat counter-intuitive as the edges are bi-directional.

The main open problem raised by our work is whether for every number t of Byzantine processors, (t, ϵ)-reliable communication from a to b implies *efficient* (t, ϵ)-reliable communication from a to b. That is,

Open Problem 1 *Let N be a partially-authenticated network with n processors, a be a sender, b be a receiver, and t be an integer. Assume that a can (t, ϵ)-reliably transmit a message to b. Is there such a (t, ϵ)-reliable protocol whose round complexity and message complexity are polynomial in n?*

Our protocol, as well as the protocol of [1], assume that the network is synchronous. That is, a protocol proceeds in rounds; at the end of each round a processor knows if it has not received messages from another processor. In addition, every processor in the network knows when a starts to transmit a message to b. The question is whether these assumptions are necessary. There are some networks where asynchronous message transmission is possible, for example, in a network where the communication graph is $(t+1)$-connected and union of the communication and authentication graphs is $(2t + 1, a, b)$-connected.[5] Thus, the open question is as follows:

Open Problem 2 *Let N be a partially-authenticated network with n processors, a be a sender, b be a receiver, and t be an integer. Assume that a can (t, ϵ)-reliably transmit a message to b when N is synchronous. Is there a (t, ϵ)-reliable protocol if N is asynchronous?*

[5] Simply, send the message on $2t + 1$ vertex-disjoint paths in the union graph; for every authentication edge on these paths send the authenticated message on $t + 1$ vertex-disjoint paths in the communication graph.

References

[1] A. Beimel and M. Franklin. Reliable communication over partially authenticated networks. *Theoretical Computer Science*, 220:185–210, 1999.

[2] A. Beimel and L. Malka. Efficient reliable communication over partially authenticated networks. In *Proc. of the 22nd ACM symp. on Principles of Distributed Computing*, pages 233–242, 2003.

[3] M. Bläser, A. Jakoby, M. Liśkiewicz, and B. Manthey. Private computation – k-connected versus 1-connected networks. In *Advances in Cryptology – CRYPTO 2002*, volume 2442 of *Lecture Notes in Computer Science*, pages 194–209. Springer-Verlag, 2002.

[4] J. Carter and M. Wegman. Universal classes of hash functions. *J. of Computer and System Sciences*, 18:143–154, 1979.

[5] Y. Desmedt and Y. Wang. Secure communication in multicast channels: The answer to Franklin and Wright's question. *J. of Cryptology*, 14(2):121–135, 2001.

[6] Y. Desmedt and Y. Wang. Perfectly secure message transmission revisited. In L. Knudsen, editor, *Advances in Cryptology – EUROCRYPT 2002*, Lecture Notes in Computer Science, pages 502–517. Springer-Verlag, 2002.

[7] D. Dolev. The Byzantine generals strike again. *J. of Algorithms*, 3:14–30, 1982.

[8] D. Dolev, C. Dwork, O. Waarts, and M. Yung. Perfectly secure message transmission. *J. of the ACM*, 40(1):17–47, 1993.

[9] C. Dwork, D. Peleg, N. Pippenger, and E. Upfal. Fault tolerance in networks of bounded degree. *SIAM J. on Computing*, 17(5):975–988, 1988.

[10] S. Even. *Graph Algorithms*. Computer Science press, 1979.

[11] M. J. Fischer, N. A. Lynch, and M. Merritt. Easy impossibility proofs for distributed consensus problems. *Distributed Computing*, 1(1):26–39, 1986.

[12] M. Franklin and R. N. Wright. Secure communication in minimal connectivity models. *J. of Cryptology*, 13(1):9–30, 2000.

[13] M. Franklin and M. Yung. Secure hypergraphs: privacy from partial broadcast. In *Proc. of the 25th ACM Symp. on the Theory of Computing*, pages 36–44, 1993.

[14] O. Goldreich, S. Goldwasser, and N. Linial. Fault-tolerant computation in the full information model. In *Proc. of the 32nd IEEE Symp. on Foundations of Computer Science*, pages 447–457, 1991.

[15] H. Krawczyk. LFSR-based hashing and authentication. In Y. G. Desmedt, editor, *Advances in Cryptology – CRYPTO '94*, volume 839 of *Lecture Notes in Computer Science*, pages 129–139. Springer-Verlag, 1994.

[16] H. Krawczyk. New hash functions for message authentication. In L. C. Guillou and J.-J. Quisquater, editors, *Advances in Cryptology – EUROCRYPT '95*, volume 921 of *Lecture Notes in Computer Science*, pages 301–310. Springer-Verlag, 1995.

[17] M. V. N. A. Kumar, P. R. Goundan, K. Srinathan, and C. P. Rangan. On perfectly secure communication over arbitrary networks. In *Proc. of the 21st ACM symp. on Principles of Distributed Computing*, pages 193–202, 2002.

[18] N. A. Lynch. *Distributed Algorithms*. Morgan Kaufman Publishers, 1997.

[19] T. Rabin and M. Ben-Or. Verifiable secret sharing and multiparty protocols with honest majority. In *Proc. of the 21st ACM Symp. on the Theory of Computing*, pages 73–85, 1989.

[20] H. M. Sayeed and H. Abu-Amara. Perfectly secure message transmission in asynchronous networks. In *Proc. of the 7th IEEE Symposium on Parallel and Distributeed Processing*, pages 100–105, 1995.

[21] H. M. Sayeed and H. Abu-Amara. Efficient perfectly secure message transmission in synchronous networks. *Information and Computation*, 126:53–61, 1996.

[22] G. J. Simmons. A survey of information authentication. In G. J. Simmons, editor, *Contemporary Cryptology, The Science of Information Integrity*, pages 441–497. IEEE Press, 1992.

[23] K. Srinathan, V. Vinod, and C. Pandu Rangan. Efficient perfectly secure communication over synchronous networks. In *Proc. of the 22nd ACM symp. on Principles of Distributed Computing*, pages 252–252, 2003.

[24] E. Upfal. Tolerating a linear number of faults in networks of bounded degree. *Information and Computation*, 115(2):312–320, 1994.

[25] M. Wegman and J. Carter. New hash functions and their use in authentication and set equality. *J. of Computer and System Sciences*, 22:265–279, 1981.