2014 IEEE 33rd International Symposium on Reliable Distributed Systems

Self-Stabilizing Byzantine Broadcast

Alexandre Maurer
UPMC Sorbonne Universités
Alexandre.Maurer@lip6.fr

Sébastien Tixeuil
UPMC Sorbonne Universités
Institut Universitaire de France
Sebastien.Tixeuil@lip6.fr

***Abstract*—We consider the problem of reliably broadcasting messages in a multi-hop network where nodes can fail in some unforeseen manner. We consider the most general failure model: the Byzantine model, where failing nodes may exhibit arbitrary behavior, and actively try to harm the network. Previous approaches dealing with *permanent* Byzantine failures limit either the *number* of Byzantine nodes or their *density*. In dense network, the density criterium is the allowed fraction of Byzantine neighbors per correct node. In sparse networks, density has been defined as the distance between Byzantine nodes.**

In this context, we first propose a new algorithm for networks whose communication graph can be decomposed into cycles: *e.g.*, a torus can be decomposed into square cycles, a planar graph into polygonal cycles, etc. Our algorithm ensures reliable broadcast when the distance between permanent Byzantine failures is greater than twice the diameter of the largest cycle of the decomposition.

Then, we refine the first protocol to make it Byzantine fault tolerant for *transient* faults (in addition to permanent Byzantine faults). This additional property is guaranteed by means of self-stabilization, which permits to recover from any arbitrary initial state. This arbitrary initial state can be seen as the result of every node being Byzantine faulty for a short period of time (hence the *transient* qualification). This second protocol thus tolerates permanent (constrained by density) and transient (unconstrained) Byzantine failures. When the maximum degree and cycle diameter are both bounded, both solutions perform in a time that remains proportional to the network diameter.

I. Introduction

As modern networks grow larger, their individual components become more likely to fail. Indeed, nodes may be subject to crashes, attacks, transient bit flips, etc. To encompass all possible cases, we consider the most general failure model: the *Byzantine* model [13], where failing nodes can exhibit arbitrary (and potentially malicious) behavior. Tolerating Byzantine nodes implies ensuring that they are not able to cause problems in the correct part of the network (if this part is not empty, of course).

We study the problem of reliably broadcasting a message in a multihop network. In the ideal case, a correct node sends a message to its correct neighbors, that in turn send it to their own correct neighbors, and so forth. However, a single Byzantine node, if not neutralized, can forward fake messages (*e.g.* messages not initiated by the correct source node) and lie to the entire network. Our goal in this paper is to design solutions that guarantee reliable broadcast despite the presence of Byzantine nodes.

Related works

Many Byzantine-robust protocols are based on *cryptography* [3], [8]: the nodes use digital signatures to authenticate the sender across multiple hops. However, as the Byzantine nodes are supposed to ignore some cryptographic secrets, their behavior cannot be considered as entirely arbitrary. Besides, a cryptographic system requires a trusted infrastructure to initially distribute cryptographic keys. Therefore, if this initial infrastructure fails due to the initial presence of Byzantine nodes, the whole network fails. In this work, our goal is to design a totally decentralized solution, where *any* element can fail without compromising the whole system. For the aforementioned reasons, we focus on non-cryptographic solutions.

Tolerating permanent Byzantine failures. We now review existing approaches to non-cryptographic solutions to permanent Byzantine faults. A first approach is to assume a maximal *number* of such Byzantine failures can occur (typically, this number depends on n, the size of the network). Most such Byzantine-resilient algorithms are designed for completely connected networks [13], [1], [18]. In [4], it was shown that, to tolerate k Byzantine nodes in a multihop network, it is necessary and sufficient that the network is $(2k+1)$-connected, and that the number of nodes in the system is at least $3k+1$. However, this solution assumes that the topology is known to every node, and that the network is synchronous. Both requirements have been relaxed in [20]: the topology is unknown to the nodes and the scheduling is asynchronous. Yet, this solution retains the $2k+1$ connectivity bound for reliable broadcast and $k+1$ connectivity bound for failure detection.

Another line of work considers the *density* of permanent Byzantine failures. A first set of solutions considers the fraction of Byzantine neighbors per node [12], [2], [21], in the context of regular lattices [12], [2], then in more general graphs [21]. However, these solutions assume a dense network topology (each node has a large number of neighbors), and cannot be applied to sparse networks, such as tori or planar graphs (in those networks, the neighbor density criterium allows only a single permanent Byzantine fault in the entire network). For such networks, density is more adequately measured as the distance between Byzantine failures. It was shown that, for perfect reliable broadcast in a torus network, it is sufficient that the minimal distance between two Byzantine failures is greater than 4 [16]. Other approaches assume a uniform random distribution of permanent Byzantine faults [16], [15], [17] and provides guarantees in terms of communication probability.

Tolerating permanent and transient Byzantine failures. The restriction on the number or density of Byzantine entities is sensible when malicious actions are permanent or intermittent (otherwise, the problem is unsolvable), yet for transient Byzan-

1060-9857/14 $31.00 © 2014 IEEE
DOI 10.1109/SRDS.2014.10

tine failures those limits can be relaxed. Self-stabilization [5] is a versatile approach for recovering from arbitrary transient faults. Indeed, a self-stabilizing protocol guarantees to recover correct behavior in finite time whatever the initial state of the network is. Combining self-stabilization and Byzantine tolerance yields one of the most effective resilience possible, as both global transient and restricted permanent failures of arbitrary nature can be tolerated. Combining these two approaches introduces a major challenge, as Byzantine nodes can actively try to prevent stabilization, and correct nodes are unable to distinguish continuously misbehaving nodes (the permanent Byzantine ones) from transiently misbehaving ones (resulting from incorrect initial state).

Algorithms that are both self-stabilizing and tolerate a limited number of permanent Byzantine failures were proposed for both complete [7] and dense [6] (*i.e.* with $2k+1$ connectivity) networks. None of those approaches can be used in sparse networks, for the same aforementioned reasons. A notable class of self-stabilizing algorithms tolerates permanent Byzantine failures with either space [14], [19], [22] or time [11], [10], [9] locality in arbitrary shaped networks. Space local algorithms contain faults as close as possible to their source. Hence, this can only be applied to the problems where the information from distant nodes is unimportant: vertex coloring, link coloring, dining philosophers, etc. Also, time local algorithms presented so far can hold at most one permanent Byzantine node, and are not able to mask the effect of Byzantine actions for an arbitrary long period of time. Thus, both approaches are not applicable to reliable broadcast.

Our contribution

We consider the problem of reliable broadcast in a sparse network that is subject to Byzantine failures and whose communication graph can be decomposed into cycles: *e.g.*, a torus can be decomposed into square cycles, a planar graph into polygonal cycles, etc. We first propose a new algorithm for networks that allow only a constrained density of permanent Byzantine failures. Our algorithm ensures reliable broadcast when the distance between permanent Byzantine failures is greater than twice the diameter of the largest cycle of the decomposition.

Then, we refine the first protocol to make it Byzantine fault tolerant for unconstrained *transient* faults (in addition to permanent Byzantine faults). This additional property is guaranteed by means of self-stabilization. This arbitrary initial state can be seen as the result of every node being Byzantine faulty for a short period of time (hence the *transient* qualification). This second protocol thus tolerates permanent (constrained by density) and transient (unconstrained) Byzantine failures.

When the maximum degree and cycle diameter are both bounded, both solutions perform in a time that remains proportional to the network diameter.

II. Preliminaries

Let $G = (V, E)$ be the graph representing the network topology (*a.k.a.* the *communication graph*). V is the set of processes (or *nodes*), and E is the set of pairs of nodes (*edges*) that are *neighbors*. Two neighbors nodes can communicate directly through a channel.

Let D be the network *diameter* (the maximal distance between any two nodes), and let Δ be the maximal *degree* (the maximal number of neighbors per node).

We now define the notion of Z*-resilient* network (see Definition 5). For this purpose, we first give some definitions.

Definition 1 (Path)**.** *A sequence of nodes $(u_1, \ldots, u_n)$ is a* path *if, $\forall i \in \{1, \ldots, n-1\}$, u_i and u_{i+1} are neighbors. A path is* correct *if all its nodes are correct.*

Definition 2 (Cycle)**.** *A set of nodes C is a* cycle *if there exists a path $(u_1, \ldots, u_n)$ containing all the nodes of C, such that u_1 and u_n are also neighbors. A cycle is* correct *if all its nodes are correct. The* diameter *of a cycle is $n/2$ if n is even, and $(n-1)/2$ otherwise.*

Definition 3 (Connected set of cycles)**.** *An arbitrary set of cycles S of the network is* connected *if, $\forall \{C, C'\} \subseteq S$, there exists a sequence $(C_1, \ldots, C_n)$ of cycles of S such that $C_1 = C$, $C_n = C'$ and, $\forall i \in \{1, \ldots, n-1\}$, C_i and C_{i+1} have at least two nodes in common. (This definition is novel to this work.)*

To illustrate, the grey sets of cycles of Figure 1 and 2 are connected in the sense of Definition 3.

Definition 4 (Resilient decomposition)**.** *An arbitrary set of cycles S of the network is a* resilient decomposition *if, for each pair of nodes p and q, there exists a connected set $S(p, q) \subseteq S$ of at most Δ cycles such that:*

1) *Each cycle of $S(p, q)$ contains p and not q.*
2) *Each neighbor of p (distinct from q) belongs to a cycle of $S(p, q)$.*

An example of such sets $S(p, q)$ is given in Figure 1 and Figure 2.

Definition 5 (Z-resilient network)**.** *A network is Z-*resilient *if there exists an arbitrary set of cycles S of the network such that S is a resilient decomposition, and the diameter of the cycles of S is at most Z.*

For instance, in a torus network (see Figure 1), the set of square cycles is a resilient decomposition. Thus, a torus is a 2-resilient network. Similarly, in a planar graph network (see Figure 2), the set of polygonal cycles is a resilient decomposition. As the diameter of the largest cycle is 2, this network is also 2-resilient.

More generally, in Theorem 1 (see Section IV), we show that any 3-connected network admits a resilient decomposition.

In the following, we are interested in Z-resilient networks where Z is small compared to the network diameter D. This is the case of networks where the nodes are homogeneously distributed in a bidimensional or tridimensional space, and are neighbors with the closest nodes – for instance, a network of sensors or mobile robots.

III. Algorithms

In this section, we present our hypotheses and the problems. Then, we present algorithms solving these problems.

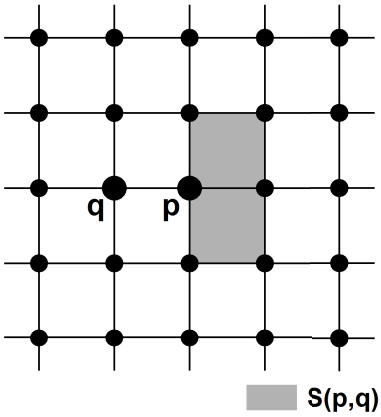

Fig. 1. Exemple of set of cycles $S(p,q)$ on a torus. Here, the cycles are the elementary squares.

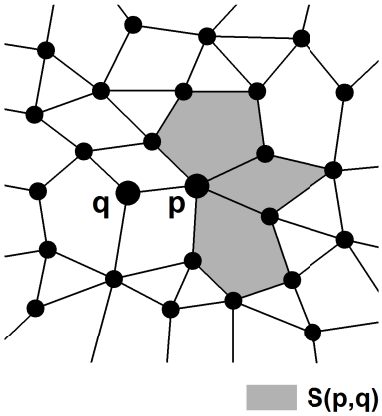

Fig. 2. Exemple of set of cycles $S(p,q)$ on a planar graph. Here, the cycles are the elementary polygons.

A. Hypotheses

We consider a Z-resilient network. Some nodes are *correct* and follow a given algorithm. The other nodes are permanent *Byzantine*, and have an arbitrary malicious behavior. The correct nodes do not know which nodes are Byzantine.

In the following, we assume that the minimal distance between two Byzantine nodes is strictly greater than $2Z$.

A node can send *messages* to its neighbors. We assume *authenticated channels* (or *oral model*): each node has a unique identifier, and knows the identifier of its neighbors. Therefore, when a node receives a message from a neighbor p, it knows that p sent the message.

Each correct node is regularly activated, and executes a given algorithm. We make no hypothesis on the order of activation of nodes. However, we assume that the time between two activations of a same node has an upper bound T_1. Similarly, the time for a message to cross a communication channel has an upper bound T_2. Let $T = max(T_1, T_2)$.

Note that this bound T is unknown to the correct nodes, and that our algorithms do not use any time primitive. This bound is only used to evaluate the global broadcast time.

B. Problem 1: reliable broadcast

We consider the following problem: each correct node p wants to broadcast a message $p.m_0$ to the other correct nodes. We say that a node q *accepts* m *from* p when it considers that m is the message broadcast by p (that is, $m = p.m_0$). The difficulty is that the permanent Byzantine nodes can forward false messages ($m \neq p.m_0$) to destabilize the network. For instance, they can try to make q accept m' from p, with $m' \neq p.m_0$.

We say that an algorithm executed by the correct nodes ensures *reliable broadcast* when, for each pair of correct nodes p and q, we always eventually reach a configuration where q has accepted $p.m_0$ from p, and never accepts another message from p. This guarantee must be independent of any possible behavior of the permanent Byzantine nodes. We present an algorithm that ensures reliable broadcast in III-D.

C. Problem 2: self-stabilizing reliable broadcast

We now consider a stronger version of the previous problem: in addition of tolerating permanent Byzantine nodes, we must also recover from *any* arbitrary initial state (that *e.g.* results from every node being transiently Byzantine). In other words, we now assume that the local memories of nodes and channels can initially contain any element.

More precisely, at time $t = 0$:

- For each correct node, any local memory used in our algorithm can contain *any* element.
- Each communication channel $\{p, q\}$ can contain *any* message sent by p, but not yet received by q.

This arbitrary initial state can represent the temporary violation of the density requirement of permanent Byzantine nodes. We present an algorithm that ensures reliable broadcast despite any arbitrary initial state in III-E.

D. Algorithm 1: reliable broadcast

Let us present an algorithm that ensures reliable broadcast in at most $8D\Delta^2ZT$ time units. The correctness proof of this algorithm is given in Theorem 2 (see Section V).

Informal description

Each correct node p initially multicasts $p.m_0$, and its correct neighbors accept $p.m_0$ from p. Then, the nodes send tuples (p, m, X), where p is the supposed initiator of the message m (that is, $m = p.m_0$), and X is a set recording the nodes visited by this message. This message can visit at most Z nodes.

When a correct node has received (p, m, X) and (p, m, X') through two disjoint set of nodes X and X', it accepts m from p and multicasts $(p, m, \emptyset)$. Then, the same mechanism is repeated until each correct node accepts m from p.

The underlying idea is that two Byzantine nodes can never cooperate to make a correct node accept a false message, as a

message can cross at most Z hops, and the distance between Byzantine nodes is more than $2Z$.

Each correct node p has a memory set $p.Rec$, initially empty, to register the messages received. We say that a node *multicasts* a message when it sends this message to all its neighbors.

Description of algorithm 1

When p is activated, it executes the following algorithm:

1) If p is activated for the first time: multicast $p.m_0$.
2) If m is received from a neighbor q: accept m, multicast $(q, m, \emptyset)$ and never accept another message from q.
3) If (s, m, X) is received from a neighbor q, with $q \notin X$ and $|X| < Z$: add $(s, m, X \cup \{q\})$ to $p.Rec$ and multicast it.
4) When there exists s, m, X and X' such that $X \cap X' = \emptyset$, $(s, m, X) \in p.Rec$ and $(s, m, X') \in p.Rec$: accept m from s, multicast $(s, m, \emptyset)$ and never accept another message from s.

E. Algorithm 2: self-stabilizing reliable broadcast

Let us present an algorithm that ensures reliable broadcast despite any arbitrary initial state in at most $12D\Delta^2ZT$ time units. The correctness proof of this algorithm is given in Theorem 3 (see Section VI).

The interest of algorithm 1 compared to algorithm 2 is that it uses shorter messages. Indeed, algorithm 1 requires the messages to register up to Z node identifiers. Algorithm 2 requires to register up to $K = 4D\Delta^2Z$ node identifiers.

Besides, algorithm 2 does not terminate: indeed, if it was the case, the initial configuration could be a wrong terminal configuration, and we would not have the self-stabilization property. However, the transient faults only cause a finite number of wrong messages.

In particular, as a torus is a 2-resilient network (see Section II), this algorithm ensures reliable broadcast in a torus network despite any arbitrary initial state, provided that the distance between Byzantine failures is greater than 4.

Informal description

The previous algorithm cannot work if we assume an arbitrary initial state. Indeed, in the initial state, the broadcast of false messages can already be initiated, and it becomes impossible to distinguish authentic messages from false messages.

We thus adopt a different strategy. First, we remove the limit of Z nodes identifiers. Therefore, when a message broadcast by a correct node p reaches a correct node q, all the nodes visited by the message are registered.

The node q waits until it receives a same message from p through several different paths. Then, it checks if the fusion of these paths can reconstitute a sequence of cycles $(C_1, \ldots, C_n)$ such as described in Definition 3. We call this a (p, q)-*valid* set of sequences (see Definition 6). If yes, q accepts the message from p.

The interest of this mechanism is the following: as the distance between Byzantine nodes is more than $2Z$, a (p, q)-valid set contains at least one correct path connecting p to q, which ensures that the message is correct. Besides, the fake sequences of node identifiers resulting from the arbitrary initial state are eliminated after a certain time.

Each correct node p has a memory set $p.Rec$, but this set can now initially contain any element. Besides, a correct node q can already have accepted any message from a node p. Later, q can accept another message from p, which replaces the previous one, and so forth.

Definitions

Let $K = 4D\Delta^2Z$.

Definition 6 ((p, q)-valid set of sequences)**.** *Let Ω be a set of sequences $(u_1, \ldots, u_n)$ of node identifiers. Let $G(\Omega)$ be the graph (V, E) such that:*

- *V is the set of node identifiers of the sequences of Ω.*
- *E is the set of pairs of node identifiers $\{p, q\}$ such that there exists $(u_1, \ldots, u_n) \in \Omega$ and $i \in \{1, \ldots, n-1\}$ such that $p = u_i$ and $q = u_{i+1}$.*

*We say that Ω is (p, q)-*valid *if:*

- *$\forall (u_1, \ldots, u_n) \in \Omega$, $u_1 = p$ and $u_n = q$.*
- *$G(\Omega)$ can be decomposed in a sequence $(C_1, \ldots, C_m)$ of cycles of diameter at most Z, such that $p \in C_1$, $q \in C_m$ and $\forall i \in \{1, \ldots, m-1\}$, C_i and C_{i+1} have at least two elements in common.*
- *For each path $(u_1, \ldots, u_n)$ of $G(\Omega)$ such that $u_1 = p$, $u_n = q$ and $n \leq K$, $(u_1, \ldots, u_n) \in \Omega$.*

Description of algorithm 2

When a correct node p is activated, it executes the following algorithm:

1) Multicast $(p, p.m_0, (p))$.
2) If $(s, m, (u_1, \ldots, u_n))$ is received from a neighbor q, with $u_n = q$ and $n \leq K$: add $(s, m, (u_1, \ldots, u_n, p))$ to $p.Rec$ and multicast it.
3) When there exists s, m and a (s, p)-valid set Ω such that, $\forall X \in \Omega$, $(s, m, X) \in p.Rec$: accept m from s, and $\forall (m', X')$ such that $(s, m', X') \in p.Rec$, remove (s, m', X') from $p.Rec$.

IV. Topology properties

In this section, we prove the results that only concern the communication graph, independently of the algorithms.

In IV-A, we show that any 3-connected graph admits a resilient decomposition, as claimed in Section II. In IV-B, we show that the set of correct cycles is connected, which is a key property fort the correctness proofs of our algorithms.

A. Resilient decomposition

Theorem 1. *If the network is 3-connected, there exists a resilient decomposition.*

Proof: Let us suppose that the communication graph G is 3-connected.

Let p and q be two nodes. Let u and v be two neighbors of p. Let G' be the graph obtained by removing the nodes p and q from G. As G is 3-connected, at least 3 nodes must be removed to disconnect the graph. Thus, G' is connected. Let $(u_1, \dots, u_n)$ be the shortest path in G' such that $u_1 = u$ and $u_n = v$. Let $C(u, v) = \{p, u_1, \dots, u_n\}$. According to Definition 2, $C(u, v)$ is a cycle in G.

As G is 3-connected, p has at least 2 neighbors distinct from q. Let $(v_1, \dots, v_m)$ be a sequence containing all the neighbors of p distinct from q, with $m \leq \Delta$. $\forall i \in \{1, \dots, m-1\}$, let $C_i = C(v_i, v_{i+1})$. At last, let $S(p, q) = \{C_1, \dots, C_{m-1}\}$.

Let us show that $S(p, q)$ satisfies the properties of Definition 4:

- $\forall i \in \{1, \dots, m-1\}$, C_i and C_{i+1} have the nodes p and v_{i+1} in common. Thus, according to Definition 3, $S(p, q)$ is connected.
- By definition, $S(p, q)$ contains $m-1$ cycles. As $m \leq \Delta$, $S(p, q)$ contains less than Δ cycles
- Let C_i be a cycle of $S(p, q)$. Then, by definition of $C(u, v)$, C_i contains p, and does not contain q.
- Let v_i be a neighbor of p. If $i = m$, v_i belongs to C_{m-1}. Otherwise, v_i belongs to C_i. Thus, each neighbor of p belongs to a cycle of $S(p, q)$.

Therefore, $S = \bigcup_{\{p,q\} \in V} S(p, q)$ is a resilient decomposition, according to Definition 4. ■

B. Connected sets of cycles

Let us show that the set of correct cycles is connected in the sense of Definition 3. This property in used in the correctness proofs of our algorithms, to show that the correct messages always manage to broadcast between Byzantine nodes.

For this purpose, we first show that each pair of correct nodes is connected by a bounded correct path (Lemmas 1 and 2). Then, we show that each pair of correct cycles is connected by a bounded set of cycles (Lemmas 3 and 4).

Lemma 1. *Let b be a Byzantine node. Let u and v be two neighbors of b. Then, there exists a correct path of at most ΔZ hops connecting u and v.*

Proof: Let q be any node distinct from b, u and v.

First, let us show that b is the only Byzantine node of the cycles of $S(b, q)$. Indeed, let us suppose the opposite: a cycle of $S(b, q)$ contains a Byzantine node $b' \neq b$. Then, the Byzantine nodes b and b' are on the same cycle, and are distant from at most Z hops: contradiction. Therefore, b is the only Byzantine node of the cycles of $S(b, q)$.

Let C (resp. C') be two cycles of $S(b, q)$ such that $u \in C$ (resp. $v \in C'$). As $S(b, q)$ is connected, according to Definition 3, there exists a sequence $(C_1, \dots, C_n)$ of cycles of $S(b, q)$ such that $C = C_1$, $C' = C_n$ and $\forall i \in \{1, \dots, n-1\}$, C_i and C_{i+1} have at least two nodes in common. As $S(b, q)$ contains at most Δ cycles, $n \leq \Delta$. Thus, as b is the only Byzantine node of C_i and C_{i+1}, C_i and C_{i+1} have at least one correct node in common. Let u_i be that node. Let $u_0 = u$ and $u_n = v$.

Let us show the following property $\mathcal{P}_i$ by induction, $\forall i \in \{0, \dots, n\}$: there exists a correct path of at most iZ hops connecting u and u_i.

- $\mathcal{P}_0$ is true, as $u = u_0$.
- Let us suppose that $\mathcal{P}_i$ is true, for $i \in \{0, \dots, n-1\}$. Then, as $\{u_i, u_{i+1}\} \subseteq C_{i+1}$ and b is the only Byzantine node of C_{i+1}, there exists a path of at most Z nodes of C_{i+1} connecting u_i and u_{i+1}. Thus, $\mathcal{P}_{i+1}$ is true.

Therefore, $\mathcal{P}_n$ is true, and there exists a correct path of at most $nZ \leq \Delta Z$ hops connecting u and $u_n = v$. Thus, the result.

■

Lemma 2. *Let p and q be two correct nodes. There exists a correct path of at most $2D\Delta$ hops connecting p and q.*

Proof: As D is the network diameter, let P be a path of n hops connecting p and q, with $n \leq D$. If P is correct, the result follows.

Otherwise, let b be a Byzantine node of P. As the distance between two Byzantine nodes is more than $2Z$, b has two correct neighbors u and v on P. Thus, according to Lemma 1, there exists a correct path of at most ΔZ hops connecting u and v. Thus, we can replace b by this path, and we obtain a new path P' that does not contain b.

We repeat this process for each Byzantine node of P, until we get a fully correct path. As the distance between two Byzantine nodes is more than $2Z$, there are at most $1 + D/2Z \leq D/Z$ Byzantine nodes in P. Thus, the final correct path contains at most $D + \Delta Z D/Z \leq 2D\Delta$ hops.

■

Lemma 3. *Let p be a correct node. Let u and v be two correct neighbors of p. Then, there exists a connected set X containing at most Δ correct cycles of diameter at most Z, such that u and p (resp. v and p) belong to a cycle of X.*

Proof: First, let us choose a particular node q.

If all the nodes at distance Z or less from p are correct, let q be any node distinct from p, u and v.

Otherwise, let b be a Byzantine node at distance Z or less from p. Let us show that b is the only Byzantine node at distance Z or less from p. Indeed, let us suppose that there exists another Byzantine node b' in this situation. Then, b (resp.

b') is at Z hops or less from p, and the distance between b and b' is at most $2Z$: contradiction. Let $q = b$.

Now, let us show that we can take $X = S(p, q)$:

- By definition, $S(p, q)$ is connected, contains at most Δ cycles, and Z is an upper bound of the diameter of the cycles of $S(p, q)$.

- Let C be a cycle of $S(p, q)$. By definition, C does not contain q. As the diameter of C is at most Z, all the nodes of C are at Z hops or less from p. Therefore, as all the nodes at Z hops or less from p and distinct from q are correct, C is correct. Thus, all the cycles of $S(p, q)$ are correct.

- For each neighbor w of p, there exists a cycle of $S(p, q)$ containing w. Therefore, there exists a cycle C (resp. C') of $S(p, q)$ containing u and p (resp. v and p).

Thus, the result, if we take $X = S(p, q)$. ■

Lemma 4. *Let p and q be two correct nodes. Then, there exists a connected set Y containing at most $2D\Delta^2$ correct cycles of diameter at most Z, such that p (resp. q) belongs to a cycle of Y.*

Proof: According to Lemma 2, there exists a correct path $(u_0, \ldots, u_n)$, with $n \leq 2D\Delta$, such that $u_0 = p$ and $u_n = q$. Let us prove the following property $\mathcal{P}_i$ by induction, $\forall i \in \{1, \ldots, n\}$: there exists a connected set X_i, containing at most $i\Delta$ correct cycles of diameter at most Z, such that p (resp. u_i) belongs to a cycle of X_i.

- First, let us show that $\mathcal{P}_1$ is true. Let v be a correct neighbor of p distinct from u_1. Then, according to Lemma 3, there exists a connected set X containing at most Δ correct cycles of diameter at most Z, such that p and u_1 (resp. p and v) belong to a cycle of X. Thus, $\mathcal{P}_1$ is true, if we take $X_1 = X$.

- Now, let us suppose that $\mathcal{P}_i$ is true, for $i \in \{1, \ldots, n\}$. Let C be the cycle of X_i containing u_i. Let v be a correct neighbor of u_i in C. Then, according to Lemma 3, there exists a connected set X containing at most Δ correct cycles of diameter at most Z, such that u_i and v (resp. u_i and u_{i+1}) belong to a cycle of X. Let C' be the cycle of X containing u_i and v. Then, as C and C' share two nodes, $X_i \cup X$ is connected, and contains at most $(i + 1)\Delta$ cycles. Thus, $\mathcal{P}_{i+1}$ is true, if we take $X_{i+1} = X_i \cup X$.

Therefore, $\mathcal{P}_n$ is true. Thus, the result, as $n \leq 2D\Delta$. ■

V. Correctness proof of algorithm 1

Let us show that algorithm 1 ensures reliable broadcast in at most $8D\Delta^2 ZT$ time units. For this purpose, we first show that the correct nodes can only accept correct messages (Lemma 5). Then, we show that they always eventually accept any correct message (Lemma 6 and Theorem 2).

Lemma 5. *Let p and q be two correct nodes. Then, if q accepts m from p, we necessarily have $m = p.m_0$.*

Proof: Let us suppose the opposite, and let q be the first correct node to accept a message $m \neq p.m_0$ from p. The node q cannot accept m from p in step 2 of our algorithm, as p only multicasts $p.m_0$. Thus, it was in step 4. Implying that there exists X and X' such that $X \cap X' = \emptyset$, $(p, m, X) \in q.Rec$ and $(p, m, X') \in q.Rec$.

Let us show that there exists a Byzantine node $b \in X$ at Z hops or less from q. For this purpose, let us suppose the opposite, and let us show the following property $\mathcal{P}_i$ by induction, $\forall i \in \{0, \ldots, |X|\}$: there exists a correct node u_i at i hops or less from q and a set $X_i \subseteq X$ such that u_i multicasts (p, m, X_i) and $|X_i| \leq Z - i$.

- As $(p, m, X) \in q.Rec$, (p, m, X) was added to $q.Rec$ in step 3 of our algorithm. It implies that $|X| \leq Z$ and that p multicasts (p, m, X). Thus, $\mathcal{P}_0$ is true if we take $u_0 = q$ and $X_0 = X$.

- Now, let us suppose that $\mathcal{P}_i$ is true, for $i \in \{0, \ldots, |X| - 1\}$. As u_i multicasts (p, m, X_i), according to step 3 of our algorithm, it implies that there exists a node $v \in X_i \subseteq X$ such that u_i receives $(p, m, X_i - \{v\})$ from v. As v is at $i + 1 \leq |X| \leq Z$ hops or less from q, according to our hypothesis, v cannot be Byzantine. Thus, v is correct, and $\mathcal{P}_{i+1}$ is true if we take $u_{i+1} = v$ and $X_{i+1} = X_i - \{v\}$.

Thus, $\mathcal{P}_{|X|}$ is true, and $u_{|X|}$ multicasts $(p, m, \emptyset)$, which was necessarily in step 2 of our algorithm. It implies that $u_{|X|}$ accepts m from p before q: contradiction. Thus, there exists a Byzantine node $b \in X$ at Z hops or less from q.

Similarly, there exists a Byzantine node $b' \in X'$ at Z hops or less from q. Thus, as $X \cap X' = \emptyset$, there exists two distinct Byzantine nodes b and b' distant from $2Z$ hops or less: contradiction. Thus, the result. ■

Lemma 6. *Let p be a correct node, and let C be a correct cycle of diameter at most Z in the communication graph. Let us suppose that two nodes u and v of C accept $p.m_0$ from p before date t. Then, all nodes of C accept $p.m_0$ from p before date $t + 4ZT$.*

Proof: According to Lemma 5, a correct node can only accept $p.m_0$ from p.

Let w be a node of C such that, if one of the 3 nodes $\{u, v, w\}$ is removed, there still exists a path of nodes of C connecting the two remaining nodes in at most Z hops. Thus, there exists two internally disjoint paths $(u_1, \ldots, u_n)$ and $(v_1, \ldots, v_m)$ of nodes of C, with $u_1 = u$, $v_1 = v$, $u_n = v_m = w$, $n \leq Z$ and $m \leq Z$. Let $X_0 = Y_0 = \emptyset$, and for $i \neq 0$, let $X_i = \{u_1, \ldots, u_i\}$ and $Y_i = \{v_1, \ldots, v_i\}$.

Let us show the following property $\mathcal{P}_i$ by induction, $\forall i \in \{1, \ldots, n\}$: u_i multicasts $(p, p.m_0, X_i)$ before date $t + 2iT$.

- As $u_1 = u$ accepts $p.m_0$ from p before date t, according to our algorithm, u multicasts $(p, p.m_0, \emptyset)$ before date t. Thus, $\mathcal{P}_0$ is true.

- Let us suppose that $\mathcal{P}_i$ is true, for $i \in \{0, \ldots, n-1\}$. As u_i multicasts $(p, p.m_0, X_i)$ before $t + 2iT$, u_{i+1} receives $(p, p.m_0, X_i)$ from u_i before $t + 2(i+1)T$. Thus, as $|X_i| < n \leq Z$, according to step 3 of our algorithm, u_{i+1} multicasts $(p, p.m_0, X_{i+1})$ before $t + 2(i+1)T$, and $\mathcal{P}_{i+1}$ is true.

Therefore, $\mathcal{P}_n$ is true, and $u_n = w$ multicasts $(p, p.m_0, X_n)$ before date $t + 2nT \leq t + 2ZT$. According to step 3 of our algorithm, it implies that we have $(p, p.m_0, X_{n-1}) \in w.Rec$ before $t + 2ZT$. Similarly, we have $(p, p.m_0, Y_{m-1}) \in w.Rec$ before $t + 2ZT$. Thus, as $Y_{m-1} \cap X_{n-1} = \emptyset$, according to step 4 of our algorithm, w accepts $p.m_0$ from p before $t + 2ZT$.

Now, let w' be any correct node of C. If $w' \in \{u, v, w\}$, the result follows. Otherwise, according to the choice of w, we can take two nodes $\{u', v'\} \subseteq \{u, v, w\}$ such that there exists two internally disjoint paths $(u'_1, \ldots, u'_{n'})$ and $(v'_1, \ldots, v'_{m'})$ of nodes of C, with $u'_1 = u$, $v'_1 = v$, $u'_{n'} = v'_{m'} = w'$, $n' \leq Z$ and $m' \leq Z$. Then, by a perfectly similar reasoning, w' accepts $p.m_0$ from p before $t + 4ZT$. Thus, the result.

■

Theorem 2. *Let p and q be two correct nodes. Then, q accepts $p.m_0$ from p before date $8D\Delta^2 ZT$, and never accepts another message from p.*

Proof: According to Lemma 5, a correct node can only accept $p.m_0$ from p.

If p and q are neighbors, according to our algorithm, q accepts $p.m_0$ from p before date $2T$, and then never accepts another message from q. Thus, the result. Now, let us suppose that p and q are not neighbors.

According to Lemma 4, there exists a sequence of correct cycles $(C_1, \ldots, C_n)$ of diameter at most Z in the communication graph, such that $p \in C_1$, $q \in C_n$, $n \leq 2D\Delta^2$ and $\forall i \in \{1, \ldots, n-1\}$, C_i and C_{i+1} have at least two nodes in common.

Let us show the following property $\mathcal{P}_i$ by induction, for $i \in \{1, \ldots, n\}$: all the nodes of C_i accept $p.m_0$ from p before date $2T + i4ZT$.

- Let u and v be the two neighbors of p in C_1. Then, according to step 1 and 2 of our algorithm, u and v accept $p.m_0$ from p before date $2T$. Thus, according to Lemma 6, $\mathcal{P}_1$ is true.

- Let us suppose that $\mathcal{P}_i$ is true, for $i \in \{1, \ldots, n-1\}$. Then, as C_i and C_{i+1} have two nodes in common, according to Lemma 6, $\mathcal{P}_{i+1}$ is true.

Therefore, $\mathcal{P}_n$ is true, and q accepts $p.m_0$ from p before date $2T + n4ZT \leq 2T + 8D\Delta^2 ZT \leq 9D\Delta^2 ZT$. Besides, according to our algorithm, q never accepts another message from p. Thus, the result.

■

VI. Correctness proof of algorithm 2

Let us show that algorithm 2 ensures reliable broadcast despite any arbitrary initial state in at most $12D\Delta^2 ZT$ time units. In VI-A, we show that the effects of the arbitrary initial state dissipate after a certain time. In VI-B, we show that messages are regularly accepted. At last, in VI-C, we show that we always achieve reliable broadcast.

A. Self-stabilization

Let us show that any problematic message resulting from the arbitrary initial state is eliminated after $2KT$ time units.

For this purpose, we introduce the notion of *real sequence* (see Definition 7). A real sequence is a sequence of nodes that corresponds to a correct path after the last Byzantine node. In Lemma 7, we show that after $2KT$ time units, all the sequences attached to messages are real. Therefore, after $2KT$ time units, the lies can only come from the permanent Byzantine nodes, and not from the arbitrary initial state.

Definition 7 (Real sequence)**.** *A sequence of node identifiers $(u_1, \ldots, u_n)$ is* real *if $n \in \{2, \ldots, K-1\}$ and:*

- *Either $(u_1, \ldots, u_n)$ is a correct path is the communication graph.*
- *Or there exists $k \in \{1, \ldots, n-1\}$ such that u_k is a Byzantine node, u_{k+1} is a correct neighbor of u_k, and $(u_{k+1}, \ldots, u_n)$ is a correct path in the communication graph.*

Lemma 7. *Let p and q be two correct nodes. Let us suppose that (p, m, X) is added to $q.Rec$ after date $2KT$. Then, X is necessarily a real sequence. Besides, if X is a correct path in the communication graph, $m = p.m_0$.*

Proof: Let us suppose the opposite: X is not a real sequence. As (p, m, X) is added to $q.Rec$, according to our algorithm, X is necessarily a sequence $(u_1, \ldots, u_n)$, with $n \in \{2, \ldots, K\}$ and $u_n = q$. For $i \in \{1, \ldots, n-1\}$, let $X_i = (u_1, \ldots, u_{n-i})$ and $Y_i = (u_{n-i}, \ldots, u_n)$.

Let us show the following property $\mathcal{P}_i$ by induction, $\forall i \in \{1, \ldots, n-1\}$: Y_i is a correct path in the communication graph and u_{n-i} multicasts (p, m, X_i) after date $2(K-i)T$.

- As (p, m, X) is added to $q.Rec$ after date $2KT > 0$, according to our algorithm, $q = u_n$ necessarily receives (p, m, X_1) from u_{n-1}. It implies that u_{n-1} multicasts (p, m, X_1) after date $2(K-1)T > 0$. If u_{n-1} is Byzantine, according to Definition 7, X is real: contradiction. Thus, u_{n-1} is correct, and $Y_1 = (u_{n-1}, u_n)$ is a correct path. Thus, $\mathcal{P}_1$ is true.

- Now, let us suppose that $\mathcal{P}_i$ is true, for $i \in \{1, \ldots, n-2\}$. As u_{n-i} multicasts (p, m, X_i) after date $2(K-i)T > 0$, according to our algorithm, u_{n-i} necessarily receives (p, m, X_{i+1}) from u_{n-i-1}. It implies that u_{n-i-1} multicasts (p, m, X_{i+1}) after date $2(K-i-1)T > 0$. If u_{n-i-1} is Byzantine, according to Definition 7, X is real: contradiction. Thus, u_{n-i-1} is correct. Therefore, as Y_i is a correct path and u_{n-i-1} is a correct neighbor of u_{n-i}, Y_{i+1} is also a correct path. Thus, $\mathcal{P}_{i+1}$ is true.

Therefore, $\mathcal{P}_{n-1}$ is true, and $X = Y_{n-1}$ is a correct path in the communication graph. Thus, according to Definition 7,

X is a real sequence: contradiction. Thus, the first part of the result.

Besides, in the case where X is a correct path in the communication graph, as $\mathcal{P}_{n-1}$ is true, $u_1 = p$ multicasts $(u_1, m, X_{n-1}) = (p, m, (p))$ before date $2(K - n + 1)i > 0$. Thus, according to our algorithm, we necessarily have $m = p.m_0$.

■

B. Regular acceptance

Let us show that, for each pair of correct nodes p and q, q accepts a message from p each $2KT$ time units (whether or not this message is correct).

For this purpose, we show that q receives messages from p through each path of length K (Lemma 8). Then, we show that these paths reconstitute the sequence of cycles required by our algorithm for acceptance (Lemma 9).

Lemma 8. *Let $X = (u_1, \ldots, u_n)$ be a correct path in the communication graph, with $n \in \{2, \ldots, K-1\}$, and let t be a date. Then, $(u_1, u_1.m_0, X)$ is added to $u_n.Rec$ between t and $t + (2K - 1)T$.*

Proof: Let us show the following property $\mathcal{P}_i$ by induction, $\forall i \in \{1, \ldots, n-1\}$: u_{i+1} multicasts $(u_1, u_1.m_0, X_i)$ between t and $t + (2i + 1)T$, with $X_i = (u_1, \ldots, u_{i+1})$.

- First, let us show that $\mathcal{P}_1$ is true. As u_1 is activated between t and $t+T$, u_1 multicasts $(u_1, u_1.m_0, (u_1))$. Thus, as u_2 receives $(u_1, u_1.m_0, (u_1))$ between t and $t + 3T$, according to step 2 of our algorithm, u_2 multicasts $(u_1, u_1.m_0, (u_1, u_2))$. Thus, $\mathcal{P}_1$ is true.

- Now, let us suppose that $\mathcal{P}_i$ is true, for $i \in \{1, \ldots, n-2\}$. As u_{i+1} multicasts $(u_1, u_1.m_0, X_i)$ between t and $t + (2i + 1)T$, u_{i+2} receives $(u_1, u_1.m_0, X_i)$ between t and $t + (2(i + 1) + 1)T$. Thus, as $|X_i| \leq n \leq K$, according to step 2 of our algorithm, u_{i+2} multicasts $(u_1, u_1.m_0, X_{i+1})$. Thus, $\mathcal{P}_{i+1}$ is true.

Therefore, $\mathcal{P}_{n-1}$ is true, and u_n multicasts $(u_1, u_1.m_0, X)$ between t and $t + (2(n-1)+1)T \leq t + (2K-1)T$. Implying that $(u_1, u_1.m_0, X)$ is added to $u_n.Rec$ between t and $t + (2K - 1)T$. Thus, the result. ■

Lemma 9. *Let p and q be two correct nodes, and let t be a date. Then, between t and $t+2KT$, q accepts a message from p.*

Proof: According to Lemma 4, there exists a sequence $(C_1, \ldots, C_m)$ of correct cycles of diameter at most Z, such that $p \in C_1$, $q \in C_m$, $m \leq 2D\Delta^2$ and $\forall i \in \{1, \ldots, n-1\}$, C_i and C_{i+1} have at least two nodes in common.

Let Ω be the set of paths $(u_1, \ldots, u_n)$ such that $u_1 = p$, $u_n = q$, $n \leq K$, and forall $i \in \{1, \ldots, n\}$, u_i belongs to a cycle of $(C_1, \ldots, C_m)$. As $2Zm \leq K$, each node and each edge of a cycle of $(C_1, \ldots, C_m)$ is contained by at least one path of Ω. Therefore, according to Definition 6, Ω is (p, q)-valid.

According to Lemma 8, $\forall X \in \Omega$, $(p, p.m_0, X)$ is added to $q.Rec$ between t and $t + (2K - 1)T$. Thus, two possibilities:

- Either one of these tuples $(p, p.m_0, X)$ has been removed between t and $t + (2K - 1)T$. According to step 3 of our algorithm, it implies that q accepts a message from p between t and $t + 2KT$.

- Or, at date $t + (2K - 1)T$, $\forall X \in \Omega$, we have $(p, p.m_0, X) \in q.Rec$. As q is activated between $t + (2K - 1)T$ and $t + 2KT$, according to step 3 of our algorithm, as Ω is valid, q accepts $p.m_0$ from p.

Thus, in all cases, q accepts a message from p between t and $t + 2KT$.

■

C. Reliable broadcast

At last, let us show that our algorithm ensures reliable broadcast. For this purpose, we show that after a certain time, a message accepted is necessarily correct (Lemmas 10 and 11). Then, we show that the correct messages are eventually and definitively accepted (Theorem 3).

Lemma 10. *Let p and q be two correct nodes. Let Ω be a (p, q)-valid set such that, $\forall X \in \Omega$, X is a real sequence. Then, at least one of these sequences is a correct path in the communication graph.*

Proof: Let us suppose the opposite: Ω does not contain any sequence that corresponds to a correct path in the communication graph.

As the sequences of Ω are real, according to Definition 7, it implies that $\forall X = (u_1, \ldots, u_n) \in \Omega$, there exists $k \in \{1, \ldots, n-1\}$ such that u_k is a Byzantine node, u_{k+1} is a correct neighbor of u_k, and $(u_{k+1}, \ldots, u_n)$ is a correct path in the communication graph. Thus, these Byzantine nodes u_k form a node cut on $G(\Omega)$, and the subgraph G' containing q after the cut is a subgraph of the communication graph.

As Ω id (p, q)-valid, $G(\Omega)$ can be decomposed in a sequence $(C_1, \ldots, C_m)$ of cycles of diameter at most Z, such that $p \in C_1$, $q \in C_m$ and $\forall i \in \{1, \ldots, m-1\}$, C_i and C_{i+1} have at least two elements in common. Let $i \in \{1, \ldots, n\}$ be the smallest integer such that a node of C_i belongs to G'. As $G(\Omega)$ is 2-connected, C_i contains at least two Byzantine nodes. As the diameter of C_i is at most Z, the distance between these two Byzantine nodes in G' is at most $2Z$. Therefore, as G' is a subgraph of the communication graph, the distance between these two Byzantine nodes is at most $2Z$: contradiction. Thus, the result.

■

Lemma 11. *Let p and q be two correct nodes. After date $4KT$, if q accepts m from p, then we necessarily have $m = p.m_0$.*

Proof: According to Lemma 7, after date $2KT$, if a tuple (p, m, X) is added to $q.Rec$, then X is a real sequence. According to Lemma 9, between dates $2KT$ and $4KT$, q accepts a message from p. According to our algorithm, it implies

that $\forall(m', X')$ such that $(p, m', X') \in q.Rec$, $(p, m'X')$ was removed from $q.Rec$. Therefore, after date $4KT$, $\forall X$ such that $(p, m, X) \in q.Rec$, X is a real sequence.

As q accepts m from p, according to our algorithm, there exists a (p, q)-valid set Ω such that $\forall X \in \Omega$, $(p, m, X) \in q.Rec$. As all the sequences of Ω are real, according to Lemma 10, Ω contains a least one sequence $X_0 = (u_1, \ldots, u_n)$ that is a correct path in the communication graph. Thus, as $(p, m, X_0) \in q.Rec$, according to Lemma 7, $m = p.m_0$. ■

Theorem 3. *Let p and q be two non-neighbors correct nodes. Then, after date $6KT = 12D\Delta^2ZT$, q has accepted $p.m_0$ from p, and never accepts another message from p.*

Proof: According to Lemma 11, after date $4KT$, if q accepts a message from p, this is necessarily $p.m_0$. Thus, according to Lemma 9, between dates $4KT$ and $6KT$, q accepts $p.m_0$ from p, and then never accepts another message from p. ■

VII. Conclusion

In this paper, we generalized the existing distance-based result for Byzantine-tolerant broadcast in two orthogonal directions. A first generalization goes from tori-shaped networks [16] to arbitrary 3-connected networks (or, any network that admits a cycle decomposition). A second generalization was to relax the Byzantine density constraints by allowing every node to behave arbitrarily (and possibly maliciously) for a short period of time, enabling the first permanent and transient Byzantine tolerant reliable broadcast in multihop sparse networks. Quite surprisingly, our positive results show that adding self-stabilization to (permanent) Byzantine tolerance does not come at a high cost with respect to density constraints for permanents Byzantine nodes. Specifically, instantiating our protocol on a torus shaped network, we retain the same density constraint (no two permanent Byzantine nodes may be less that four hops apart) as the non-stabilizing case [16], yet tolerates transient phases during which *every* node may be Byzantine.

Many open challenges remain: allowing network dynamicity (both from a network topology point of view, but also allowing Byzantine nodes to move in the network), and characterizing the exact Byzantine node placement that enables a reliable broadcast solution in a given topology would be major achievements.

References

[1] H. Attiya and J. Welch. *Distributed Computing: Fundamentals, Simulations, and Advanced Topics*. McGraw-Hill Publishing Company, New York, May 1998. 6.

[2] Vartika Bhandari and Nitin H. Vaidya. On reliable broadcast in a radio network. In Marcos Kawazoe Aguilera and James Aspnes, editors, *PODC*, pages 138–147. ACM, 2005.

[3] Miguel Castro and Barbara Liskov. Practical Byzantine fault tolerance. In *OSDI*, pages 173–186, 1999.

[4] D. Dolev. The Byzantine generals strike again. *Journal of Algorithms*, 3(1):14–30, 1982.

[5] Shlomi Dolev. *Self-Stabilization*. MIT Press, 2000.

[6] Shlomi Dolev, Omri Liba, and Elad M. Schiller. Self-stabilizing Byzantine resilient topology discovery and message delivery. In *Proceedings of the International Conference on Networked Systems (NETYS 2013)*.

[7] Shlomi Dolev and Jennifer L. Welch. Self-stabilizing clock synchronization in the presence of Byzantine faults. *J. ACM*, 51(5):780–799, 2004.

[8] Vadim Drabkin, Roy Friedman, and Marc Segal. Efficient Byzantine broadcast in wireless ad-hoc networks. In *DSN*, pages 160–169. IEEE Computer Society, 2005.

[9] Swan Dubois, Toshimitsu Masuzawa, and Sébastien Tixeuil. The impact of topology on Byzantine containment in stabilization. In *Proceedings of DISC 2010*, Lecture Notes in Computer Science, Boston, Massachusetts, USA, September 2010. Springer Berlin / Heidelberg.

[10] Swan Dubois, Toshimitsu Masuzawa, and Sébastien Tixeuil. Bounding the impact of unbounded attacks in stabilization. *IEEE Transactions on Parallel and Distributed Systems (TPDS)*, 2011.

[11] Swan Dubois, Toshimitsu Masuzawa, and Sébastien Tixeuil. Maximum metric spanning tree made Byzantine tolerant. In David Peleg, editor, *Proceedings of DISC 2011*, Lecture Notes in Computer Science (LNCS), Rome, Italy, September 2011. Springer Berlin / Heidelberg.

[12] Chiu-Yuen Koo. Broadcast in radio networks tolerating Byzantine adversarial behavior. In Soma Chaudhuri and Shay Kutten, editors, *PODC*, pages 275–282. ACM, 2004.

[13] Leslie Lamport, Robert E. Shostak, and Marshall C. Pease. The Byzantine generals problem. *ACM Trans. Program. Lang. Syst.*, 4(3):382–401, 1982.

[14] Toshimitsu Masuzawa and Sébastien Tixeuil. Stabilizing link-coloration of arbitrary networks with unbounded Byzantine faults. *International Journal of Principles and Applications of Information Science and Technology (PAIST)*, 1(1):1–13, December 2007.

[15] Alexandre Maurer and Sébastien Tixeuil. Limiting Byzantine influence in multihop asynchronous networks. In *Proceedings of the 32nd IEEE International Conference on Distributed Computing Systems (ICDCS 2012)*, pages 183–192, June 2012.

[16] Alexandre Maurer and Sébastien Tixeuil. On Byzantine broadcast in loosely connected networks. In *Proceedings of the 26th International Symposium on Distributed Computing (DISC 2012)*, volume 7611 of *Lecture Notes in Computer Science*, pages 183–192. Springer, 2012.

[17] Alexandre Maurer and Sébastien Tixeuil. Parameterizable Byzantine broadcast in loosely connected networks. Technical report, UPMC Sorbonne Universités, 2013.

[18] Y. Minsky and F.B. Schneider. Tolerating malicious gossip. *Distributed Computing*, 16(1):49–68, 2003.

[19] Mikhail Nesterenko and Anish Arora. Tolerance to unbounded Byzantine faults. In *21st Symposium on Reliable Distributed Systems (SRDS 2002)*, pages 22–29. IEEE Computer Society, 2002.

[20] Mikhail Nesterenko and Sébastien Tixeuil. Discovering network topology in the presence of Byzantine nodes. *IEEE Transactions on Parallel and Distributed Systems (TPDS)*, 20(12):1777–1789, December 2009.

[21] Andrzej Pelc and David Peleg. Broadcasting with locally bounded Byzantine faults. *Inf. Process. Lett.*, 93(3):109–115, 2005.

[22] Yusuke Sakurai, Fukuhito Ooshita, and Toshimitsu Masuzawa. A self-stabilizing link-coloring protocol resilient to Byzantine faults in tree networks. In *Principles of Distributed Systems, 8th International Conference, OPODIS 2004*, volume 3544 of *Lecture Notes in Computer Science*, pages 283–298. Springer, 2005.