J. Parallel Distrib. Comput. 74 (2014) 3153–3160

Contents lists available at ScienceDirect

J. Parallel Distrib. Comput.

journal homepage: www.elsevier.com/locate/jpdc

Byzantine broadcast with fixed disjoint paths[☆]

Alexandre Maurer[*], Sébastien Tixeuil

Sorbonne Universités, UPMC University of Paris 06, LIP6 - UMR CNRS 7606, Paris, France

HIGHLIGHTS

- We consider the problem of reliably communicating despite Byzantine failures.
- We propose an algorithm designed for sparse networks.
- We give a methodology to determine if two nodes communicate reliably.
- With simulations, we show that our algorithm outperforms previous solutions.

ARTICLE INFO

Article history:
Received 9 March 2014
Received in revised form
11 July 2014
Accepted 29 July 2014
Available online 7 August 2014

Keywords:
Byzantine failures
Multihop networks
Asynchronous networks
Reliable broadcast
Fault tolerance
Distributed computing
Protocol
Random failures

ABSTRACT

We consider the problem of reliably broadcasting a message in a multihop network. We assume that some nodes may be Byzantine, and behave arbitrarily. We focus on cryptography-free solutions.

We propose a protocol for sparse networks (such as grids or tori) where the nodes are not aware of their position. Our protocol uses a fixed number of disjoint paths to accept and forward the message to be broadcast. It can be tuned to significantly improve the number of Byzantine nodes tolerated. We present both theoretical analysis and experimental evaluation.

1. Introduction

As modern networks grow larger, they become more likely to fail, sometimes in unforeseen ways. Indeed, nodes can be subject to crashes, attacks, transient bit flips, etc. Many failure and attack models have been proposed, but one of the most general is the *Byzantine* model proposed by Lamport et al. [15]. The model assumes that faulty nodes can behave arbitrarily. In this paper, we study the problem of reliable communication in a multihop network despite the presence of Byzantine faults. The problem proves difficult since even a single Byzantine node, if not neutralized, can lie to the entire network.

1.1. Related works

Many Byzantine-robust protocols are based on *cryptography* [4,8]: the nodes use digital signatures to authenticate the sender across multiple hops. However, cryptography itself is not unconditionally reliable, as shown by the recent Heartbleed bug [30]. According to the *defense in depth* paradigm [16], a good strategy for critical systems is to use multiple layers of security controls, including non-cryptographic layers. For instance, if the cryptographic security is compromised by a bug or a virus, the non-cryptographic communication layer can be used to safely broadcast a patch or to update cryptographic keys. Another drawback of cryptography is that it requires a centralized infrastructure to initially distribute cryptographic keys. Therefore, if this central weak spot fails, the whole network fails. Here, we would like to have a system where any element can fail independently without compromising the whole system. In this paper, we thus consider non-cryptographic strategies for reliable communication in the presence of Byzantine faults.

Cryptography-free solutions have first been studied in fully connected networks [1,15,18,19,25]: a node can directly communicate

[☆] A preliminary version of this work was presented at the DISC conference Maurer and Tixeuil [23]. The conference version of the paper only provides a simplified version of the protocol, that corresponds to the $(1, H)$ setting of the current protocol (see Section 2.2). This paper provides a fully rewritten text that generalizes the protocol to any setting $(H_1, \ldots, H_n)$, with newly developed theoretical analysis and experimental evaluation.

[*] Corresponding author.
E-mail address: alexandre.maurer@lip6.fr (A. Maurer).

http://dx.doi.org/10.1016/j.jpdc.2014.07.010
0743-7315/© 2014 Elsevier Inc. All rights reserved.

with any other node, which implies the presence of a channel between each pair of nodes. Therefore, these approaches are hardly scalable, as the number of channels per node can be physically limited. We thus study solutions in multihop networks, where a node must rely on other nodes to broadcast messages.

A notable class of algorithms assumes restrictions on the consequences of Byzantine failures either in space [21,26,29] (nodes far away from Byzantine nodes are not impacted by their behavior) or in time [10,9,11,12,20] (a Byzantine node executes only a limited number of malicious actions before being ignored by correct nodes). Space-local solutions are only applicable to problems where the information from distant nodes is unimportant (for example, vertex coloring, link coloring, or dining philosophers). Time-local solutions presented so far can tolerate at most one Byzantine node in the entire network, and are not able to mask the effect of Byzantine actions (that is, all correct nodes may be perturbed by a Byzantine node at least once). Thus, this approach is not applicable to reliable broadcast.

We now present the solutions with the same setting as our contribution: a multihop network where each node has a unique identifier, and where cryptography is not allowed. They tolerate either a certain *number* or a certain *density* of Byzantine failures.

Let us start with the solutions that tolerate a certain *number* of Byzantine failures. It was shown that, for reliable broadcast in the presence of k Byzantine nodes, it is necessary and sufficient that the network is $(2k+1)$-connected [7]. This first solution assumes that every node knows the entire topology, and that the scheduling is synchronous. Both requirements have been relaxed in [27]: the topology is unknown and the scheduling is asynchronous. However, in sparse topologies such as grid-shaped networks (where the connectivity is 4), both approaches tolerate only one Byzantine node, independently of the size of the grid.

Now, we present the solutions that tolerate a certain *density* of Byzantine failures. In dense network (where each node has a large number of neighbors), this density is represented by the *fraction* of Byzantine neighbors per node. Broadcast protocols have been proposed for nodes organized on a lattice [2,14]. However, these solutions require much more than 4 neighbors per node to enable reliable broadcast. These results were later generalized to other topologies [28], assuming that each node knows the global topology. A recent paper [17] proved the optimality of this solution for this setting. However, this approach cannot be applied to sparser networks. For instance, in a grid network, only the 8 nodes surrounding the source may accept its message.

All aforementioned results assume a large connectivity or node degree. Therefore, tolerating more Byzantine failures requires to increase the number of channels per node, which limits scalability. To overcome this problem, a probabilistic approach has been proposed in [22–24]. In this setting, the distribution of Byzantine failures is uniformly random. This hypothesis is realistic if we consider that each node has a given probability to fail or to be corrupted by an adversary [3,5,6]. We can also consider distributed hash tables used in the construction of overlay networks, where the identifier of a node joining the network is attributed randomly (therefore, its location in the overlay is random). With these assumptions, those solutions can tolerate a large number of Byzantine failures [22] with a high communication probability. This approach has been generalized to tolerate a constant rate of Byzantine nodes in an unbounded network, despite a bounded node degree [24]. However, both solutions require a global view of the network: each node must know its position in the communication graph. This strong hypothesis is difficult or impossible to satisfy in many types of networks, such as self-organized wireless sensor networks or peer-to-peer overlays.

1.2. Our contribution

In this paper, we consider the case of sparsely connected networks where the nodes do not know their position. We propose a new protocol that both contains and outperforms previous solutions. Our algorithm uses a fixed number of disjoint paths to accept and forward messages. For instance, when a communication probability of 0.99 is required on a 50×50 torus, we can tolerate a 40 times more Byzantine nodes than previous solutions.

The paper is organized as follows. In Section 2, we describe the principle of our protocol and give the algorithm executed by each correct node. In Section 3, we explain how to determine which nodes always accept correct messages, and only correct messages. In Section 4, we use this method to evaluate and compare the performances of our protocol with simulations.

2. Description of the protocol

2.1. Hypotheses

Let (V, E) be a non-oriented graph representing the topology of the network. V denotes the *nodes* of the network. E denotes the *neighborhood* relationships. A node can only send messages to its neighbors. Some nodes are *correct* and follow the protocol described further. We consider that all other nodes are totally unpredictable (or *Byzantine*) and have an arbitrary behavior.

We assume that any message sent is eventually received, and that in an infinite execution, any process is activated infinitely often. However, we make no hypothesis on synchronicity: the processes can be activated in any order. Finally, we consider the *oral* model: each node has a unique identifier, and when a node receives a message from a neighbor p, it knows that p is the author of the message. Therefore, a Byzantine node cannot forge its own identity.

2.2. Informal description

First, we present an informal description of the problem and of our protocol. Each correct node s wants to broadcast a message $s.m_0$ to the rest of the network. In the ideal case, s sends $s.m_0$ to its neighbors, which in turn transmit $s.m_0$ to their own neighbors—and so forth, until every node receives $s.m_0$. We call this an *unsecured broadcast*.

In our setting however, some nodes can be Byzantine and broadcast false messages (*i.e.* messages that are *not* sent by a correct node) in the network. In the following, we say that a correct node *accepts* m *from* s when it definitively considers that m is the message broadcast by p. We say that m is *correct* if $m = s.m_0$, and *false* otherwise. Our objective is to propose a broadcast protocol that maximizes the number of nodes accepting the correct messages.

To limit the diffusion of false messages, we introduce the following mechanism. First, $s.m_0$ is directly accepted and retransmitted by the neighbors of s. Then, to accept a message, the other correct nodes must receive confirmations from several distinct nodes, through a fixed number of disjoint paths. For instance, in Fig. 1, the right node accepts a message if and only if it is received through 3 disjoint paths of at most $H_1 = 3$ (resp $H_2 = 4$ and $H_3 = 2$) hops. The same requirement stands for every correct node. Once the message is accepted, the node retransmits it for more distant nodes, and the same principle is repeated over and over.

This specific setting of the protocol can be described by the tuple $(H_1, H_2, H_3) = (3, 4, 2)$. More generally, a setting of the protocol is described by a tuple $(H_1, \ldots, H_n)$, each H_i being a positive integer. The integer n (not to confuse with the number of nodes) and the values H_i are assigned arbitrarily: we do not know *a priori* their impact on the global performances, which is studied further in Section 4.

The underlying idea is as follows: if the Byzantine nodes are sufficiently spaced, they cannot cooperate to make a correct node accept a false message. Indeed, with setting $(3, 4, 2)$, a correct node can accept the first false message *only* if there exists 3 distinct Byzantine nodes distant of at most 3 (resp. 4 and 2) hops. This critical case is illustrated in Fig. 2.