Computer Communications 62 (2015) 1–12

Contents lists available at ScienceDirect

Computer Communications

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

Gossip-based data aggregation in hostile environments

Mousa Mousazadeh *, Behrouz Tork Ladani

Department of Computer Engineering, University of Isfahan, Hezar Jerib Ave., Isfahan, Iran

ARTICLE INFO

Article history:
Received 6 January 2014
Received in revised form 5 February 2015
Accepted 7 February 2015
Available online 14 February 2015

Keywords:
Data aggregation
Distributed averaging
Gossip-based algorithms
Security
Pull-Adjust algorithm

ABSTRACT

In the last decade, several gossip-based algorithms have been introduced for data aggregation in distributed systems. The main and common advantage of these algorithms is their robustness in dynamic and fault-prone environments. However, gossip-based algorithms are not robust in hostile environments. In such environments, some malicious nodes intentionally violate the normal execution of the algorithm to distort the aggregate value. In this paper, we focus on gossip-based averaging and try to improve the security of these algorithms. First, we revise existing gossip-based averaging algorithms to present a more secure algorithm called "Pull-Adjust." Then, we develop and examine a light, transparent, and fully distributed protection system to push back malicious nodes. The simulation results show that the proposed system considerably improves the performance of the Pull-Adjust algorithm in hostile environments.

1. Introduction

Computing the aggregate values of scattered data such as the average value, extreme values, and weighted sums can be a non-trivial task in a fully distributed system. This is even more difficult when real-world constraints come into play. In real environments, particularly, the nodes of the system are asynchronous, there is the possibility of packet dropping during transmission, communication channels are noisy, there may be node failures, and finally, each node has only a partial view of the whole system.

However, gossip-based algorithms are amazing solutions to the problem. In these algorithms, the idea of spreading gossip in human societies is used to aggregate data in distributed systems. The works in [1–6] present the most well-known algorithms of this type in which nodes repeatedly send their estimates of the aggregate value to other nodes. During communication of the local estimates, each node uses a predefined function to update its estimate based on the received data. Over time, the local estimates of nodes asymptotically converge to a common value. Although real-world constraints can affect the accuracy of aggregated values, nodes can reach a consensus on an approximation of desired aggregate value. Due to these desired properties, gossip-based algorithms have been used to solve a wide range of problems including load balancing in parallel computing [7], data aggregation in distributed systems [1–6], trust and reputation management in peer-to-peer networks [8], and privacy preserving data mining [9].

More recently, a series of works has been published on the analysis of existing gossip-based algorithms in fault-prone environments and designing more robust ones [10–18]. However, there are only a few works investigating the effect of malicious behaviors (i.e. intentional disturbances) on the integrity of values aggregated by gossip-based algorithms. In a hostile environment, malicious nodes may violate the normal execution of algorithms by sending fake values to other nodes. Sending fake values can affect the integrity of aggregate values, as well as the convergence of algorithms. In this way, the local values of nodes may converge to an incorrect value, or the algorithm may even never converge. This problem has already been investigated analytically for the main category of these algorithms, i.e., averaging algorithms, and it was shown that even a small group of malicious nodes can considerably affect the integrity of aggregate values [19]. In this paper, we go one step further and try to improve the security of gossip-based averaging algorithms in hostile environments.

While gossip-based algorithms usually use a push model for communicating local values, we will conclude that this communication model can worsen the performance of these algorithms from the security point of view. Based on this observation, we revise existing gossip-based averaging algorithm to present an algorithm that uses a pull communication model. Hence, we have called revised algorithm "Pull-Adjust." Then, we propose a protection system for this algorithm to improve its security in hostile environments. In the proposed protection system, each node uses only its local data for identifying and subsequently isolating

* Corresponding author.
E-mail addresses: mousazadeh@eng.ui.ac.ir (M. Mousazadeh), ladani@eng.ui.ac.ir (B. Tork Ladani).

http://dx.doi.org/10.1016/j.comcom.2015.02.002
0140-3664/

suspicious nodes. Therefore, it remains light, transparent, and fully distributed. To make the evaluation process possible, we keep the protection system as simple as possible, and then examine the performance of the algorithm under different attacks. Our experiments show that the proposed scheme considerably improves the accuracy of the aggregated values in hostile environments. However, the proposed scheme is not utopian; the protection system may filter out outliers. This can affect the accuracy of the aggregated values. In some applications outliers are important and of interest. For such applications, this is a limitation of the proposed scheme and we should be cautious about using it.

The rest of the paper is organized as follows: Section 2 surveys main gossip-based averaging algorithms. Section 3 deals with the vulnerabilities of gossip-based averaging algorithms and measuring their performance in hostile environments. In Section 4, the Pull-Adjust algorithm is described and analyzed. The protection system is presented and evaluated in Sections 5 and 6, respectively. We postpone the discussion of related work until Section 7 in order to build up sufficient context to compare our approach to others. Finally, Section 8 concludes the paper.

2. Gossip-based averaging

Each gossip-based algorithm consists of several transactions. In each transaction, the local estimate of at least one node is sent to another node, and also, the local estimate of at least one node is updated. In this paper, we use an asynchronous time model for describing gossip-based algorithms and for the sake of convenience and without loss of generality, we suppose that there is only one gossip transaction in each unit of time. Also, we suppose that each node uses an independent Poisson clock to start a gossip transaction. In other words, the node that starts a gossip transaction at time $t+1$ is independent of the node that starts such a transaction at time t.

Here, we review three basic gossip-based averaging algorithms. Let us start with the pairwise averaging algorithm [2,3]. In this algorithm, when the clock of a node ticks, it randomly selects one of its neighboring nodes and contacts it. After communicating local values, both nodes update their values to be the average of their current values. More formally, suppose that there are N nodes in the system and nodes i and j are the participants of the gossip transaction at $t+1$. Also, suppose that x_k^t is the local estimate of node k at time t where $k = 1, \ldots, N$. Then, the transactions of the pairwise averaging can be formalized as follows:

$$x_k^{t+1} = \begin{cases} \frac{1}{2}x_i^t + \frac{1}{2}x_j^t & \text{if } k = i \text{ or } k = j, \\ x_k^t & \text{otherwise.} \end{cases} \quad (1)$$

If the communication graph of the nodes is connected, then almost surely, the local estimates of the nodes will converge to the average value.

Push-Sum is another gossip-based algorithm which can be used not only for distributed averaging, but also for computing the weighted sums of distributed data. Although this algorithm was initially introduced for data aggregation in systems with the synchronous time model [1], it can easily be adapted for the asynchronous time model as well. In this algorithm, each node has a weight in addition to its value. Let v_k^t and w_k^t be the value and weight of node k respectively at time t. In this algorithm, each node randomly selects one of its neighboring nodes and gives it half of its own value and half of its own weight. Let i be the sending node and j be the receiving node at $t+1$. Then, gossip transactions can be described more formally as follows:

$$v_k^{t+1} = \begin{cases} \frac{1}{2}v_i^t & \text{if } k = i, \\ v_j^t + \frac{1}{2}v_i^t & \text{if } k = j, \\ v_k^t & \text{otherwise,} \end{cases}$$

$$w_k^{t+1} = \begin{cases} \frac{1}{2}w_i^t & \text{if } k = i, \\ w_j^t + \frac{1}{2}w_i^t & \text{if } k = j, \\ w_k^t & \text{otherwise.} \end{cases} \quad (2)$$

For every node k, v_k^t/w_k^t is the local estimate of aggregate value at time t. If all nodes start with weight 1, i.e., $\forall k : w_k^0 = 1$, then the average value of the nodes will eventually be obtained. Also, if only one node starts with weight 1 and others start with weight 0, then the sum of the node values is obtained. See [1] for more details.

There is another averaging algorithm that is not as famous as the above-mentioned algorithms. In this algorithm, each node continually sends its local estimate to others. Simultaneously, each node adjusts its local estimate based on the received data using an adjusting parameter $\alpha \in (0, 1)$. So, we can call this algorithm Push-Adjust. To describe the algorithm more formally, let i be the sending node and j be the receiving node at $t+1$. Then, gossip transactions can be described as follows:

$$x_k^{t+1} = \begin{cases} \alpha x_j^t + (1-\alpha)x_i^t & \text{if } k = j, \\ x_k^t & \text{otherwise.} \end{cases} \quad (3)$$

This algorithm converges almost surely on connected graphs [20]. However, since the sum of the node values (the mass [1]) changes slightly in each transaction, only an approximation of the average value is obtained instead of its exact value. In Section 4, we will investigate this issue in more detail.

3. Hostile environment

"Hostile environment" is an environment in which malicious nodes may violate the normal execution of gossip algorithms [19]. Malicious nodes may do this either by simply sending fake values to distort the aggregate values, or by exploiting the communication model to increase their share in computing the average value. We call the vulnerabilities that lead to the former attack scenarios *intrinsic vulnerabilities* and those that lead to the latter, *extrinsic vulnerabilities*. In the next two subsections we discuss the above types of vulnerability.

3.1. Intrinsic vulnerabilities

In a hostile environment, a group of malicious nodes may mislead honest nodes about the right average value by sending them fake values over time. To explain this better, consider an environment with six nodes in which the initial values of nodes are $0.1, 0.2, 0.3, 0.4, 0.5$, and 0.6. Suppose the nodes run the pairwise averaging algorithm to obtain the average value. If all the nodes are honest and correctly run the algorithm, then, as can be seen in Fig. 1, the local values of the nodes converge to the right average value, i.e., 0.35. In this figure, each dashed curve shows the expected value of one node during 100 gossip transactions when all nodes are honest. We have simulated 10,000 runs of the algorithm and used the average of the results to estimate the expected values of the nodes.

Now, suppose that nodes 1–5 are honest, but node 6 is malicious. The malicious node intends to skew the average value toward its own value, i.e., 0.6. To this end, when the malicious node participates in a gossip transaction, it constantly sends 0.6 to the other participant of the transaction without updating its own value. The result of this scenario can also be seen in Fig. 1. In this figure, the solid curves show the expected values of the nodes for this