Signal Strength as Support to Mobility Detection on Failure Detectors

Antônio Rodrigo D. De Vit
PUCRS
Porto Alegre – RS – Brazil
antonio.vit@acad.pucrs.br

César Marcon
PUCRS
Porto Alegre – RS – Brazil
cesar.marcon@pucrs.br

Raul Ceretta Nunes
UFSM
Santa Maria – RS – Brazil
ceretta@inf.ufsm.br

ABSTRACT

A Failure Detector (FD) is an essential building block to develop reliable applications on Mobile Ad-hoc NETworks (MANETs). To detect faulty nodes correctly the node mobility and disconnections are challenges that must be circumvented by an FD. This paper presents an FD algorithm that detects faulty nodes efficiently in MANETs. This approach explores the signal power intensity on the receiving messages to detect node mobility and improve the failure detector knowledge. The experimental results have shown improvements in the Quality of Service (QoS) when compared with other similar FDs.

CCS Concepts

Descriptors: A.1 [General Literature]: Introductory and Survey; C.4 [Computer Systems Organization]: Performance of Systems—Fault tolerance; modeling techniques; reliability, availability, and serviceability.

Keywords

MANETs, Failure Detector, Mobile nodes.

1. Introduction

A MANET [1] is a network containing self-configurable and mobile nodes with dynamic communication links. The nodes communicate with their neighbors for broadcast radio messages, limited to the transmission range (*tr*) of their radio. Like in other networks, MANETs users require systems with consistency, reliability, availability and security features, and applications require algorithms for coordination and consensus. In this sense, algorithms for node failure detection, a fundamental building block to build dependable distributed applications, have been widely studied for operation in mobile networks. The main challenge to developing failure detection protocols for MANETs is to differentiate a failure node from a mobile or disconnected one. Thus, some approaches were explored to circumvent node mobility in mobile networks [2][3][4][5][6][7][8].

This paper proposes a new failure detection technique for mobile ad hoc networks using the power measurement of the received signal as an element to classify the mobility of a node. The nodes are classified into regions to identify the movement of a node and to generate knowledge of the node movement. The FD uses this knowledge to adjust itself to the node mobility and to improve the detection quality. The proposed solution was evaluated in a simulation environment for MANETs, and it demonstrates good performance to improve the QoS of the FD.

The remainder of this paper is organized as follows. Section 2 presents the system model and failure detectors, including its QoS metrics. Section 3 describes the related work. Section 4 details the proposed failure detector. Section 5 presents the experimental results and Section 6 presents the conclusions.

SAC 2017, April 03-07, 2017, Marrakech, Morocco
© 2017 ACM. ISBN 978-1-4503-4486-9/17/04…$15.00
DOI: http://dx.doi.org/10.1145/3019612.3019903

2. System Model and Failure Detectors

The distributed system considered in this work is asynchronous and composed of a finite set of self-organized mobile nodes $\prod = \{p_i, p_{i+1}, ..., p_n \mid 1 \le i \le n, n > 1\}$. There is no global clock; each node has its memory, processing unit and a local clock. The nodes communicate by radio broadcast with equal and finite transmission range. The power of the reception signal depends on the transmission distance. The network topology is dynamic, and communication channels are bi-directional and reliable (do not change, do not create and do not lose messages). Partitions from nodes movement finish in a finite time. To control failures, each node in the system runs a failure detection service.

The concept of unreliable FD was introduced in [9] to circumvent the inability of solving consensus in an asynchronous system prone to failure [10]. FD encapsulates the problem and provides nodes status for the application. Typically, an FD informs its state perception from a local detection module, and a node p_j is *suspect* when a p_i detector does not receive a message from p_j within a given period. There are some communication strategies to build the notion of state in the detector [11]. Gossip [1] is the most traditional strategy for mobile networks. A Gossip-based FD sends periodic gossip messages and usually carries a list of *id*s and *heartbeat* counters [12]. Thus, the information is forwarded to the nodes that are likely to be moving towards a desired region of the network.

To evaluate FDs, Chen *et al.* [13] proposed a set of metrics that are independent of the FD implementation. The metrics are presented in two sets (primary and secondary), and enable to evaluate performance and accuracy, and can be expressed considering two nodes *p* and *q*. The secondary metrics can be derived from the primary ones. The primary metrics are: (i) *Detection Time* (T_D), which measures the time elapsed from the moment in which *q* fails until the instant when *p* permanently suspects *q*; (ii) *Mistake Recurrence Time* (T_{MR}), which measures the period between two successive wrong suspicions; and (iii) *Mistake Duration Time* (T_M), which measures the time elapsed to an FD detects its error (wrong suspicion) and correct it. A *T-transition* represents the time instant to *Trust* state and *S-transition* represents the time instant to *Suspect* state.

3. Related Work

Several works deal with nodes mobility for FD. To support gaps in receiving FD messages Friedman and Tcharny [3] determined the expected latency of each hop and the number of hops the algorithm needs to overcome the network delay. In a similar way, Hutle [2] reserves network bandwidth to the FD to get bounded jitter and to compute how much time it needs to overcome the network delay. Shridar [4] and Zia et al. [5] inserted a mobility detector that uses an additional round of *broadcast* messages and the dissemination of information lists to circumvent node mobility. Sens *et al.* [6] explore a query-response failure detection technique to detect failures and tolerate the mobility of nodes, instead of using Gossip algorithm and time monitoring. Greve et al. [7] proposed a protocol for dynamic networks with an unknown membership extending the work of Sens et al. by using fault information. Recently, Benkaouha et al. [8] introduced a heartbeat class protocol dedicated for MANETs that consisted of two components: the FD and the disconnection management. Unlikely anything else found in the literature; this paper explores the observed message receiving energy to detect movement, allowing the distinction between node mobility and faulty node.

4. Failure Detector with Mobility Support

This section presents the proposed FD based on the Signal Strength (FD2S) for MANETs.

4.1 System definitions

There are three important definitions related to the proposed FD2S algorithm: (i) the **transmission range** (*tr*) is the maximum distance that a message sent by a node p_i can reach and it is proportional to the power of the transmission signal. All node $p_i \in \prod$ have the same transmission range; (ii) a **Communicating Group** (*CG*) is a set of intercommunicating nodes dynamically composed at each predefined time interval, such that CG_i is the group of nodes containing p_i and its local neighbors, and $\forall\{p_i, p_j\} \in \Pi$: if $p_i \subset CG_j$ then $p_j \subset CG_i$; (iii) a coverage area of a node p_i is split into φ concentric and exclusive **regions of locality**, where $region_i^k$ corresponds to kth power level range inside the *tr* of p_i, such that: $\forall p_i \in \Pi$: $\varnothing = region_i^1 \cap region_i^2 \cap \ldots \cap region_i^{\varphi}$, $\forall\{p_i, p_j\} \in \Pi$: if $p_i \subset region_j^k$ then $p_j \subset region_i^k \mid 1 \leq k \leq \varphi$, $\forall p_i \in \Pi$: $ca_i = region_i^1 \cup region_i^2 \cup \ldots \cup region_i^{\varphi}$.

4.2 FD2S Algorithm

The FD2S algorithm is organized in three concurrent tasks (Fig. 1). The *sending messages task* takes care of gossiping information messages. The *receiving messages task* receives information from other nodes and sets the local data structures properly. The *status check task* decides about the status of monitored nodes. In the beginning, each p_i node suspects all other nodes except itself.

To support mobility, each instance p_i of FD2S has a set of data structures so that nodes can keep information about themselves and their neighbors: $\boldsymbol{NBL_i}$ - list of neighbors it believes to be correct; $\boldsymbol{ML_i}$ - mobility list that contains all nodes it believes in movement; $\boldsymbol{SL_i}$ - suspects list that contains all nodes that do not increment their heartbeat for *Cfail* × *Tgossip* and not in movement; $\boldsymbol{H_i^j}$ - movement history for each p_j node; $\boldsymbol{TL_i}$ - local timestamp of the node; $\boldsymbol{TS_j}$ - timestamp for the last message received from node p_j; $\boldsymbol{Hcount_j}$ - heartbeat vector that contains TL_i and TS_j for each $p_j \in \Pi$; $\boldsymbol{Cfail_j}$ - movement control for each $p_j \in \Pi$, and $\boldsymbol{Pow_j}$ – power measured on the signal received from p_j.

```
$H \leftarrow 0$; $Cfail \leftarrow 0$;
$NBL \leftarrow \{\ \}$;           // Neighbors list
$ML \leftarrow \{\ \}$;            // Mobility list
$SL \leftarrow \{p_i, ..., p_n\}$; // Suspects list
Task 1: sending messages
At every $Tgossip$ do
    $Hcount_i = Hcount_i + 1$
    broadcast($NBL_i$, $Hcount$)
Task 2: receiving messages
Upon receive a message from $p_j$ with ($NBL_j$, $Hcount$)
        update $H_i^j$ with region($Pow_j$)
        if $p_j \in ML_i$ then
            remove $p_j$ from $ML_i$
        if $p_j \in SL_i$ then
            $Cfail_j \leftarrow \lceil (TL_i - TS_j) / Tgossip \rceil$
            add $p_j$ in $NBL_i$
            remove $p_j$ from $SL_i$
        $\forall\ p_k \in NBL_j$ with $i \neq k$ do
            if $Hcount_j^k > Hcount_i^k$ then
                if $p_k \in SL_i$ then
                    $Cfail_k \leftarrow \lceil (TL_i - TS_k) / Tgossip \rceil$
                    add $p_k$ in $NBL_i$
                    remove $p_k$ from $SL_i$
                $TS_k \leftarrow TL_i$
            update $Hcount_i^k$ with Max($Hcount_i^k$, $Hcount_j^k$)
Task 3: status check
At every $TCleanup$ do
        $\forall\ p_j \in ML_i$ with $i \neq j$ do
            if $TL_i - TS_j > Tgossip + (Cfail_j \times Tgossip)$ then
                add $p_j$ in $SL_i$
                remove $p_j$ from $ML_i$
                remove $p_j$ from $NBL_i$
        $\forall\ p_j \in NBL_i$ with $i \neq j$ do
            if $TL_i - TS_j > Cfail_j \times TGossip$ then
                update $H_i^j$ with region(0)
                if $H_i^j[t-1] = H_i^j[t-2]$ then
                    add $p_j$ in $SL_i$
                    remove $p_j$ from $NBL_i$
                else add $p_j$ in $ML_i$
```

Fig. 1. Failure detection algorithm with mobility support.

4.3 FD2S Gossip and Control Mechanism

The gossip-based logic of FD2S determines that every node sends periodic gossip messages and monitors all other nodes (Task 1). At every *Tgossip* interval, each node increments its heartbeat counter and sends by radio broadcast a gossip message to its neighbors. Each message contains a list of heartbeat counters *Hcount* that includes the sender newest heartbeat number, and the newest one it knows about others monitored nodes (gossip information). Upon a node p_i receives a message from other node p_j (Task 2), the $Hcount_j$ is merged with the $Hcount_i$, and the highest *heartbeat* counter (newest) of each node is retained in the receiver node. Besides, as soon as a node receives a message, it updates the region of the sender node on the movement history H_i^j. The region depends on the signal power of the message received (Pow_j). This energy-aware gossip logic allows node discovery and network partition recovery under nodes movement. *Cfail*, ML_i and SL_i allow a node to keep the information about the mobility. Thus, a mobile node outside GC_i may remain as non-suspect (in NBL_i).

To take a decision about failures (Task 3), FD2S monitors the heartbeat counter of each monitored node. According to [14], if the counter does not increment in a range of *Tfail* time units the node is considered suspect and placed on the suspect list just after *Tcleanup* time units. *Tfail* is selected according to the likelihood of an erroneous failure detection, and *Tcleanup* is chosen according to the probability of a gossip message to arrive in an acceptable time that avoids an erroneous suspicion. However, like on [15], to better performance

FD2S simplifies the protocol suspecting nodes after *Tcleanup* without verifying *Tfail*. From [16] it is enough to set *Tcleanup* as a multiple of *Tgossip*. The idea is to match the time required for the information to reach other nodes within the *Tcleanup* time limit [16]. Thus, in this work *Cfail* determines the number of *Tgossip* intervals that must be expected to a monitored node information to reach its monitor node. Likewise, considering the MANET environment, *Cfail* depends on each node movement pattern (trajectory) to overlap the local disconnection time of p_j; there is a *Cfail* for each node $p \in \Pi$.

5. Experimental Results

In the experiments, the simulation model follows [3], and it was set up to operate in a field of 300 meters x 300 meters, with a transmission frequency of 2.412~2.462 GHz (IEEE 802.11g) and a bandwidth of 54 Mbps. The movement model is the *Random Waypoint*, and every node's transmission range was set up to 50 meters. The simulation was performed with Omnet++ Simulator [17], and some values were used for the maximum moving speed (2, 4, 6 and 8 m/s), and quantity of nodes (30, 40, 50 and 60 nodes). Our simulations were executed on a PC running Linux on an Intel i7 with 16 GB of RAM.

The first experiment evaluates the performance of FD2S; i.e., the T_D metric that measures the amount of time spent to detect a real failure. The experiment used scenarios containing some number of nodes (from 30 to 60 nodes) and different speeds (from 2 to 8 m/s). A node was randomly chosen to fail at a given known simulation time to compute the performance metric T_D. Fig. 2 (i) presents the results to T_D on these conditions. The metric T_D decreases both when the number of nodes increases and when the node's movement speed increases. These results show that the FD2S adapts itself to the dynamic characteristics of the network and improves its performance according to the increasing of connections. The information dissemination strategy is effective when the network is well connected, making fast the FD2S failure detection. In a similar way, the increase of node's movement speed reduces the time a node needs to accomplish a given trajectory, thus decreasing *Cfail*. Since the failure detector handles *Cfail* counter with expected intervals to classify a node as faulty, this decreasing on *Cfail* results in better performance of FD2S. To verify how fast the FD2S can react to a failure, we also measure T_D^L, the lower bound of T_D. Fig. 2 (ii) illustrates the minimum time to detect failures (best detection time), and to get stability (small influence of the number and speed of the nodes) from 50 or more nodes in the simulation playground. It happens because FD2S differentiates a fixed node from a mobile one. When a failure occurs in a node that stays in the same region of the transmission range for more than two verification rounds, the FD2S reacts in a similar way as in a fixed network. The performance in the worst case (30 nodes and 2 m/s) it is not so good, but when compared with similar failure detectors we note it also demonstrates good QoS.

The second experiment evaluates the accuracy of FD2S regarding the mistake recurrence time (T_{MR}) and mistake duration time (T_M). A mistake occurs when a working node p_j is considered to be suspect by some other node p_i. In the experiment a node never fails, in such a way, every suspicion by some detector is a mistake. Thus, an accurate FD seeks for high values of T_{MR}, to reduce the recurrence of the error, and for small values of T_M, to recognize its error faster. In other words, an FD should make few mistakes, and when it does one, it should dismiss it in a shorter time. With the same setup from first experiment, Fig. 2 (iii) shows that the T_M values are low when compared with T_{MR}. In order, FD2S makes few mistakes and becomes aware of its mistake quickly, demonstrating good accuracy features. The results also show that increasing the number of nodes decreases T_{MR}, but also decreases the T_M rate. This behavior shows that the FD2S also tends to fit well despite the increasing of the network. Additionally, Fig. 2 (iv) illustrates the behavior of false suspicions considering different number and speed of nodes. The results show that increasing the number and the speed of the nodes rises the number of false suspicious proportionally. This proportionality could also be viewed by T_{MR} in Fig. 2 (iii), and it is derived from the mechanism of the algorithm in handling regions and nodes. Further, on checking the average mistake rate (λ_M = 1/ T_{MR} [13]) it can be observed low values (in average less than 1/100). A low λ_M allows FD2S to be applied on long-live applications, where a mistake (S-Transition) could result in a costly interrupt.

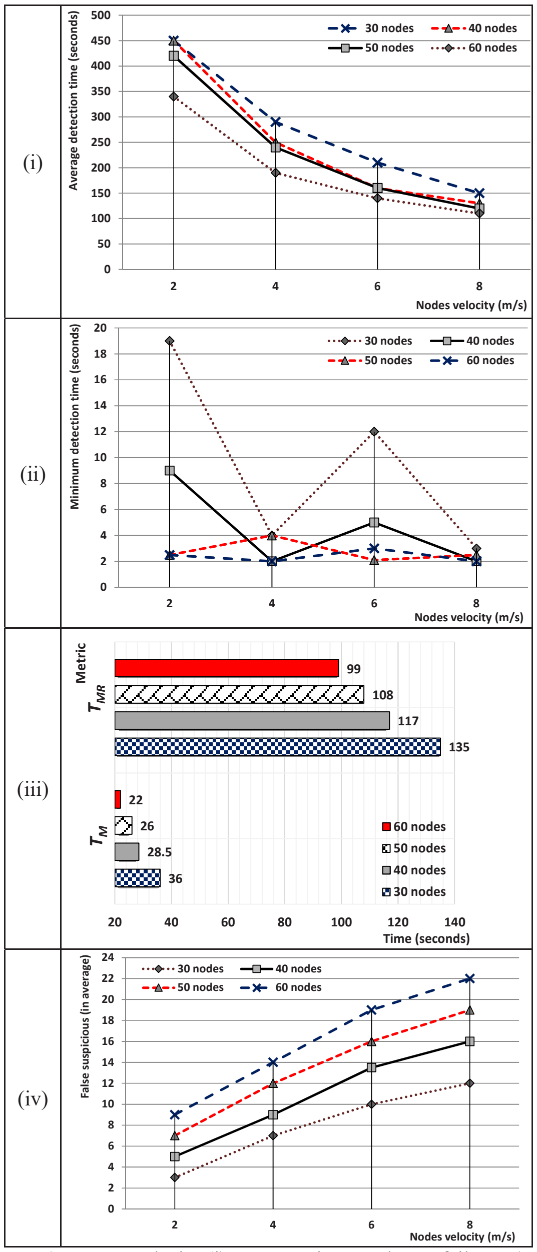


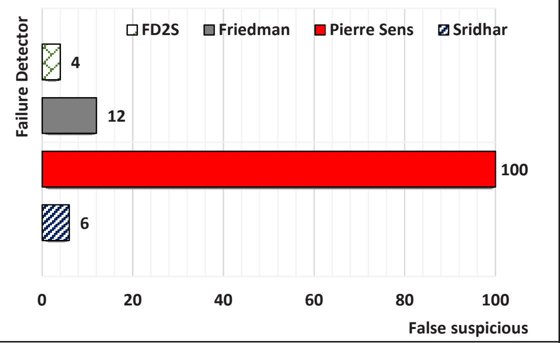
Fig. 2. FD2S QoS analysis: (i) average time to detect failures (T_D); (ii) minimum time to detect failures (T_D^L); (iii) *Mistake recurrence time* – T_{MR} and *Mistake duration time* (T_M); (iv) Number of false suspicions according to the velocity of the nodes.

These two QoS experiments enable us to conclude that the performance and accuracy of the FD2S present good behavior on MANET environment.

In the third experiment, we compare FD2S with other failure detectors. The goal is to compare the number of false suspicions and the speed detection (T_D) obtained by FD2S against that one's generated by similar detectors proposed in [3][4][7]. The tests used a network with 30 nodes moving on 2 m/s.

From Fig. 3 (i), FD2S generates less false suspicions when compared to other detectors. Friedman [3] and Sridhar [4] detectors also generate few false suspicions, if compared with Pierre Sens algorithm [6]. We note after Query messages being disseminated in the network the Pierre Sens algorithm [6] corrects itself, but it needs more time to do it when comparing with others. On the other hand, the FD2S gets few false suspicions due to its fast reaction of movements and by its adequate adjust of *Cfail*. Concerning the time spent to detect a real failure (T_D), Fig. 3 (ii) illustrates that Pierre Sens algorithm [6] results in better detection performance, but that FD2S produces a better result than others do. Thus, this experiment demonstrates that FD2S presents a competitive performance.

(i)



(ii)

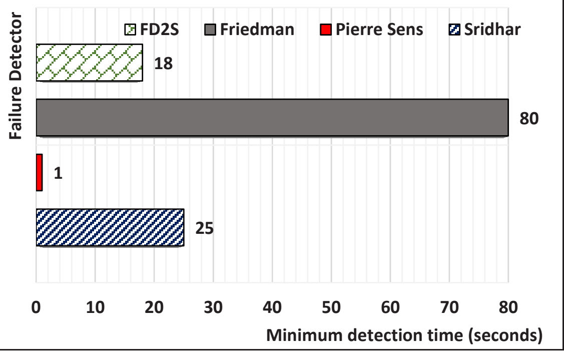


Fig. 3. FD2S quality versus others FDs: (i) FD2S quality compared with other failure detectors to false suspicious; (ii) FD2S quality compared to other failure detectors (minimum T_D).

Together, the analysis of the T_D and the number of false suspicions demonstrates that the FD2S has a high QoS when compared to other similar detectors. This quality derives from the elimination of immediate suspicion of a mobile node p_j coming out of coverage area of p_i. On FD2S, the node notices a movement of a neighbor, which is coming out of its transmission range and adjusts the *Cfail* parameter to wait an additional period before placing the node in the suspects list. It is noted that the perception of the movement was made from a history of regions, which considers the movements in the last two transmission periods. The interval may be sufficient if the node repeats its route in a shorter interval than twice the history update period, but must be reviewed and it is adaptable according to the new findings routes (that take longer). To adapt the periods, FD2S does not need to know how many hops a node is far from the other, or information about the network jitter. Instead of it, the algorithm can adapt and get this information just observing the operation of the network by messages exchange.

6. Conclusions

This paper presented the FD2S, a new failure detector algorithm for MANETs that distinguish mobile nodes from faulty nodes by considering the power of received signal from the incoming messages. The new mechanism does not request new messages and neither increases energy consume. The paper shows that the FD2S has a good performance in detecting faulty nodes, properly adapting its timeout to node movements. For known trajectory, the FD2S adapts itself quickly and avoids false suspicions when a node leaves the transmission range. When compared with similar algorithms the FD2S results in a small number of false detections and presents good QoS.

References

[1] L. Loo, J. Mauri, J. Ortiz. "*Mobile Ad Hoc Networks: current status and futures trends*". CRC Press, 538p., 2012.

[2] M. Hutle. "An efficient failure detector for sparsely connected networks". *International Conference on Parallel and Distributed Computing and Networks*, pp. 369-374, 2004.

[3] R. Friedman, G. Tcharny. "Evaluating failure detection in mobile ad-hoc networks", *International Journal of Wireless and Mobile Computing*, v. 1, pp. 8-23, 2005.

[4] N. Sridhar, "Decentralized local failure detection in dynamic distributed systems". *IEEE Symposium on Reliable Distributed Systems* (SRDS). pp. 143-154, 2006.

[5] H. Zia, N. Sridhar, S. Sastry. "*Failure detectors for wireless sensor-actuator systems*". *Ad Hoc Networks*, v. 7, pp. 1001-1013, 2009.

[6] P. Sens, L. Arantes, M. Bouillaguet, V. Simon, F. Greve, "An Unreliable Failure Detector for Unknown and Mobile Networks". *International Conference on Principles of Distributed Systems (OPODIS)*, pp. 555-559, 2008.

[7] F. Greve et al. "Eventually Strong Failure Detector with Unknown Membership". *Comput. J.*, pp. 1507-1524, 2012.

[8] H. Benkaouha, A. Abdelli, N. Badache, J. Ben-Othman and L. Mokdad, "Towards Improving Failure Detection in Mobile Ad Hoc Networks", *2015 IEEE Global Communications Conference (GLOBECOM)*, San Diego, CA, pp. 1-6, 2015.

[9] T. Chandra, S. Toueg. "Unreliable failure detectors for reliable distributed systems". *Journal of the ACM*, v. 43, n. 2, pp. 225-267, 1996.

[10] M. Fischer, N. Lynch, M. Paterson. "Impossibility of distributed consensus with one faulty process". Journal of the ACM, v. 32, n. 2, pp. 374-382, 1985.

[11] F. Bonnet, M. Raynal. "Anonymous asynchronous systems: The case of failure detectors". *Distributed Computing*, v. 26, n. 3, pp. 141-158, 2013.

[12] R. Friedman *et al.* "Gossiping on MANETs: the Beauty and the Beast", *ACM Operating Systems Review*, pp. 67-74, 2007.

[13] W. Chen, S. Toueg and M. K. Aguilera, "On the quality of service of failure detectors", in *IEEE Transactions on Computers*, vol. 51, no. 5, pp. 561-580, 2002.

[14] R. Van Renesse, Y. Minsky, M. Hayden. "A gossip-style failure detection service", *Middleware'98*, pp. 55-70, 1998.

[15] M. Burns, A. George, B. Wallace. "Simulative Performance Analysis of Gossip Failure Detection for Scalable Distributed Systems". *Cluster Computing*, v. 2, n. 3, pp. 207-217, 1999.

[16] P. Raman et al. "GEMS: Gossip-Enabled Monitoring Service for Scalable Heterogeneous Distributed Systems". *Cluster Computing*, v. 9, n. 1, pp. 101-120, 2006.

[17] Omnet++. "*Discrete Event Simulator*". Available in https://omnetpp.org/, Sep, 2015.