A Twofold Self-Healing Approach for MANET Survivability Reinforcement

Leila Mechtri, Fatiha Djemili Tolba, Salim Ghanemi, Damien Magoni

A Twofold Self-Healing Approach for MANET Survivability Reinforcement

Leila Mechtri* and Fatiha Djemili Tolba

Networks and Systems Laboratory (LRS), Badji Mokhtar University, Computer Science dept., P. O. 12, 23000 Annaba, Algeria
E-mail: mechteri@lrs-annaba.net
E-mail: fatiha.djemili@univ-annaba.org
* Corresponding author

Salim Ghanemi

Badji Mokhtar University, Computer Science dept., P. O. 12, 23000 Annaba, Algeria
E-mail: ghanemisalim@yahoo.com

Damien Magoni

University of Bordeaux
Computer Science Research Laboratory (LaBRI)
351 cours de la Libération, 33405 Talence Cedex
E-mail: magoni@labri.fr

Abstract: Distributed systems are by nature fault-prone systems. The situation becomes more complex in the presence of intrusions that continue to grow in both number and severity, especially in open environments like MANET. In this paper, we present a twofold self-healing approach to reinforce MANET survivability. First, a fault-tolerant IDS is designed by replication of individual agents within MASID to ensure continuous supervision of the network. However, since not all intrusions are predictable, there might have some serious effects on the network before being detected and completely removed. For that, even if the implications of intrusions could be minimized by the intrusion detection system MASID, still the need for the recovery of altered or deleted data is a vital step to ensure the correct functioning of the network. For that, a recovery-oriented approach for a self-healing MANET is also presented. It is based on the ability of MASID-R to assess the damage caused by the detected intrusions and aimed at enabling the supervised network to heal itself of those faults and damages. Simulations using *ns-2* have been performed to study the feasibility and prove the optimality of the proposed approach.

Keywords: Survivability, Fault-tolerance, Intrusion Detection, Self-healing, Replication, Recovery-oriented Approach, MANET.

Reference to this paper should be made as follows: Mechtri, L., Djemili, F.T., Ghanemi, S. and Magoni, D. (201x) ‘A Twofold Self-Healing Approach for MANET Survivability Reinforcement’, *Int. J. International Journal of Intelligent Engineering Informatics*, Vol. X, No. Y, pp.xx-xx.

This paper is a revised and expanded version of a paper entitled ***[An IDS-***

based Self-healing Approach for MANET Survival'] *presented at* ***[International Conference on Intelligent Information Processing, Security and Advanced Communication- IPAC '15, November 23 - 25, 2015, Batna, Algeria]****.*

1 Introduction

In recent years, there has been a growing interest in securing the mobile ad-hoc networks (MANETs). Some researchers developed preventive approaches (Chen and Wu, 2010) to guarantee security while many others prefer the use of secure routing protocols (Abusalah et al., 2013; Patil and Sidnal, 2013). Also, there has been, recently, a great tendency to develop intrusion detection systems (IDS) specifically designed to fit MANET requirements in terms of both security and constraints. However, the obtained security level does not always guarantee that the network is completely free of faults and malfunctioning. More specifically, some intrusions might have some undesirable effects on the nodes or network services being targeted by the intruder(s) before being detected and completely removed. For that, the network should be designed so that to survive such situations and to autonomously heal any potential damages. This has led to the emergence of the so-called self-healing techniques as essential complementary techniques to achieve truly autonomous survivable networks.

In this paper, we will first present a replication framework to enable the intrusion detection system MASID (Multi-Agent System for Intrusion Detection) (Mechtri et al., 2012) to recover from individual and/or multiple agent failures. This is to give it more flexibility, reliability and most importantly high availability which means continuous surveillance of the network, i.e., ensuring permanent protection of the network.

However, the network is not yet that reliable since data lost due to intrusions is not recovered. For that, we would like to improve the reliability and consistency of the network, so as to enable it to heal itself of faults and to better survive malicious attacks. To this end, a new paradigm for a self-healing MANET is presented. It is based on the ability of the adopted intrusion detection system (MASID-R) to assess the damage caused by the detected intrusions. Then, building on this assessment, SH-MASID-R, via its healing agent, initiates and executes the necessary actions to heal the network. The main objective of this approach is to enable the supervised network to heal itself of faults and damages caused by intrusions and to better survive malicious attacks (mainly packet dropping attacks) and malfunctioning, which would considerably improve the resilience and reliability of the network, thus improving its survivability. The resulting system is fully autonomous: it accurately detects intrusions launched against it, appropriately responds to them, and perfectly heals the caused damages.

The rest of the paper is organized as follows. Section 2 provides a brief overview of the related work. Then, section 3 and section 4 detail the proposed approach while section 5 describes the conducted experiments and discusses the obtained results. Finally, section 6 concludes the paper and initiates for possible future work.

2 Related Work

The vulnerabilities of the mobile ad-hoc networks and the proliferation of intrusions and thereby the need for survivability have been widely studied in the literature. For instance, there has been considerable research in the fields of network monitoring, intrusion detection, self-healing, fault-tolerant systems and survivable networks. In this section, we review some interesting works in these areas.

2.1 Fault-tolerant IDS

Kaur et al. (2010) analysed and discussed one of the most challenging issues in the field of intrusion detection, which is the IDS' fault tolerance. For instance, they evaluated some of the widely used fault tolerance mechanisms, namely replication of software agents, employment of redundancy in processing elements, integrity checking for self-healing, use of reconfigurable hardware and restructuring architectures, and fault detection using heartbeat messages in multi-agent systems.

The results of this study show that an IDS must be fault tolerant and that replication techniques, which we will use in this paper, provide the IDS with both high availability and reliability. Example IDS that support fault tolerance are (Sen, 2010; Zhihao and Guanyu, 2012; Sasikumar and Manjula, 2012; Chang and Shin, 2010).

Sen (2010) presented the design of a distributed specification-based IDS that consists of a group of agents cooperating with each other to carry out the task of intrusion detection. These agents use an inference process that uses a Bayesian network of data to model the structures of well-known attack types as well as the network's normal usage patterns. The fault tolerance of the proposed IDS is enhanced through a prompt identification and isolation of compromised hosts in the network through a distributed trust framework. More specifically, the authors integrated a byzantine agreement protocol to enable the proposed IDS to identify compromised nodes within the system.

Also, Zhihao and Guanyu (2012) presented a distributed intrusion detection model inspired by the human immune system, in which mobile agents are used in the hope of increasing its fault tolerance, adaptability, and flexibility. In the proposed IDS, called MADIDS, immune agents are classified as: configuration agents, collect agents, detection agents, and response agents. According to their functions, these agents are either static or roaming on the network to monitor it, collect host and network-related data, and detect and respond to possible intrusions. Although the proposed IDS is made more robust, fault tolerant and flexible through the integration of mobile agents, it still suffer from high false negative rates.

Sasikumar and Manjula (2012) developed a dynamic distributed intrusion detection system (DDIDS) based on mobile agents. In the proposed architecture, each DDIDS system has a connection with other DDIDSs for information sharing as well as problem solving. Each of these systems is composed of three layers: layers consisting of host agents and net agents, mobile agents, and decision making and replication agents. Also, the system has a DDIDS console that has control over every DDIDS agent in the network and that supports for report preparation. The use of decision-making and replication agents helped greatly in increasing DDIDS fault tolerance. Moreover, the use of agents improved the IDS' performance mainly in terms of real-time intrusion detection.

Finally, Chang and Shin (2010) focused on the detection of intrusions at the application layer. Similarly to many other agent-based IDSs, they used a local IDS, consisting on a monitoring and detection agent, a response agent, and a communication agent to detect intrusions at every network node. Their main contribution is the use of mobile agents to augment each node's intrusion-detection capability. Specifically, they equipped the network with a mobile agent server capable of creating and dispatching three types of mobile agents: update, analysis, and verification agents. If a local IDS fails to identify a suspicious behaviour, its response agent will request the mobile agent server to send analysis agents for further investigation. The analysis agent is capable of a more detailed analysis and diagnosis compared to the local IDS as it can launch multi-point network-based anomaly detection. Once the investigation completed, the analysis agent will report the results to the mobile agent server. Hence, if a new attack type is detected or the suspicious activity is judged as a change in the node's behaviour, an update agent will be created to update local IDSs' databases with the new attack signature or normal profile. Further, the mobile agent server periodically checks the status of local IDSs using verification agents. If a vulnerability is detected, it will patch and install programs on the concerned mobile nodes via its update agents. Clearly, mobile agents can overcome network latency and reduce the network load related to intrusion detection. Also, this approach was a step forward in enhancing agent-based IDSs' fault tolerance, but it might lead to further problems. For instance, the mobile agent server might exhaust the node's resources (mainly processing and storage) in addition to being a single point of failure.

2.2 IDS-based self-healing networks

Elsadig and Abdullah (2009) presented a bio-inspired approach to design an intrusion prevention system (IPS) for securing MANET against intrusions. More specifically, this approach implements an analytical computational framework based on the danger theory. Using agents (Sense, Analysis, and Adaptive agents) of multi-layers, the proposed IPS analyses the behaviour of system processes and network traffic to detect harmful events. The prevention process is preceded by a training phase, during which normal and dangerous signatures are specified. A danger signal is then activated upon any match with the dangerous signatures. The potential intrusion will thereby be prevented by disconnecting or blocking the suspected connection and the adopted self-healing mechanism (self-healing agent) will be triggered so that to regenerate the damaged components. For that, the self-healing agent is provided with a knowledge base containing all candidate system components, in addition to the healing function. For instance, whenever a healing message is received from the Analysis Agent, a healing component is immediately identified, deployed and tested to keep the system in function. The designed IPS is autonomous and the network's fault repair ability was considerably enhanced through the adopted self-healing mechanism.

Lee and Suzuki (2008) proposed a decentralized self-healing mechanism that detects and recovers from wormhole attacks in wireless multi-hop sensor networks using connectivity information. This mechanism, denoted SWAT, identifies the locations of malicious nodes, isolates them, and finally recovers the routing structure distorted by them. For that, each sensor node maintains a neighbour list containing the connectivity information about one hop and two hops neighbour nodes. Using this list, a node monitors the connectivity with its neighbours. Anomaly detection within these connections results in the production of a danger signal in the form of a control packet that will trigger the recovery phase in which recovery packets are used to isolate the wormhole nodes and to heal the caused damages within the wormhole sphere based on a pre-established routing tree structure.

Kong et al. (2005) proposed a new intrusion protection mechanism based on the notion of self-healing communities. These communities consist on groups of neighbouring nodes among which a network service is distributed so as to mitigate the adverse actions of selfish and malicious nodes. For each end-to-end connection, a chain of self-healing communities along the shortest path are established based on localized simple schemes. The idea, here, is that a self-healing community is perceived as a big virtual node that replaces the conventional single forwarding node. Thus, data delivery is considered as a combination of conventional node-based data forwarding and community-based healing. At each intermediate community in a route, the most recent control packet forwarder is supposed to be the current data forwarder. If this node fails to forward a packet due to maliciousness, selfishness or network dynamics, members in the same self-healing community will make up. This way, routes can be healed locally with minimal latency. Yet, because such self-healing communities might lose shape due to mobility and network dynamics, their reconfiguration is deemed crucial for the survivability of the proposed solution. For that, the authors used end-to-end probing with a probing interval adapted with respect to network dynamics.

In (Jain et al., 2011) an artificial immune intrusion detection system inspired by idiotypic networks was proposed. More specifically, the pattern recognition technology is adopted to execute the process of intrusion detection. The proposed detection approach is divided into two main phases: (a) a training phase during which, an idiotypic network is built and trained to learn normal patterns and attacks' profiles; and (b) a detection phase to distinguish between normal and abnormal patterns and update the idiotypic network if necessary. The proposed IDS is aimed at detecting and analysing malicious activities, measuring the effects of these activities, and as a final step it triggers the self-healing process. This latter is responsible for the diagnosis, fault identification, and the configuration of anomalous activities. Further, the self-healing system is charged with candidate fix generation, damage repair, self-testing and deployment.

3 Replication for Continuous Protection

The introduction of a multi-agent system to the distributed and cooperative architecture of MASID brought more flexibility and a complete automation of the detection process. Nevertheless, considering multi-agent systems vulnerability to faults and system failures, MASID is deemed unreliable as it adopts no failure recovery mechanism while being a fault-prone IDS. For that, it is necessary to enhance our IDS' fault tolerance so that to guarantee continuous protection of the network. Since replication is a key technique to achieve fault tolerance in distributed and dynamic environments (which is the case of MASID), we use this technique to avoid malfunctioning resulting from the potential failure of one or more agents within the multi-agent system constituting our intrusion detection system as described in the following subsections.

3.1 Agent Replication

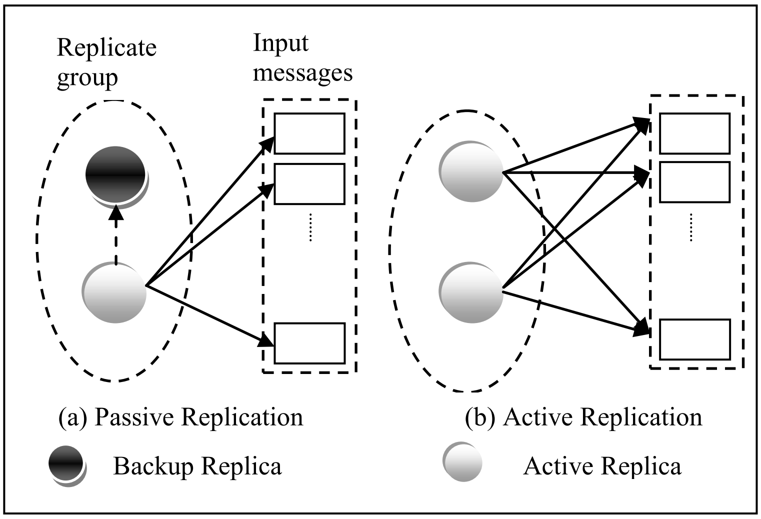
Agent replication (Fedoruk and Deters, 2002) is generally defined as the act of creating duplicates of one or more agents in a multi-agent system. Each of these duplicates performs the same task as the original agent. The group of duplicate agents is referred to as a replicate group and the individual agents within the replicate group are referred to as replicas.

There are two basic types of agent replication: heterogeneous and homogeneous. In heterogeneous replication, replicas are functionally equivalent, but they may have been implemented separately i.e., they are not identical but designed to perform the same action. In homogeneous replication, replicas are exact copies of the original agent. In other words, the replicas are not only functionally equivalent but are copies of the same code.

Furthermore, considering the relation between an agent and its replicas, we can distinguish two categories of replication, namely passive and active replication. In passive replication, also called single-copy passive replication or primary-backup replication (Budhiraja et al., 1993), there exist one active replica (denoted primary) that processes all input messages and periodically updates the other replicas (called backups) in order to maintain coherence and to constitute a recovery point in case of failure. Figure 1 (a) shows a simple example of a passive replication scenario.

Whereas, active replication, also called the state machine approach (Schneider, 1990), is characterized by the existence of several replicas that process concurrently all input messages as illustrated in Figure 1 (b). Since all of the replicas are active at the same time, this category of replication might lead to the overhead of the CPU but it is still the best choice if fast recovery delays are required. On the contrary, passive replication needs more processing time for recovery but it requires less CPU resources as it activates replicas only in case of failures. It is then obvious that the choice of the most suitable replication technique depends on the application and its environment, such as the failure rate, or the available resources.

Figure 1 Active vs. passive replication



3.2 MASID-R

The replication framework presented in this paper concerns all of the agents constituting MASID with the exception of the SNMP (Simple Network Management Protocol) agent which we will not replicate since the SNMP protocol already provides replication to its agents. In summary, the agents concerned with replication are: collector, the detection agent, the collaboration agent and the response agent. In our framework we distinguish two varieties of replication:

3.2.1 Replication at System Initialization

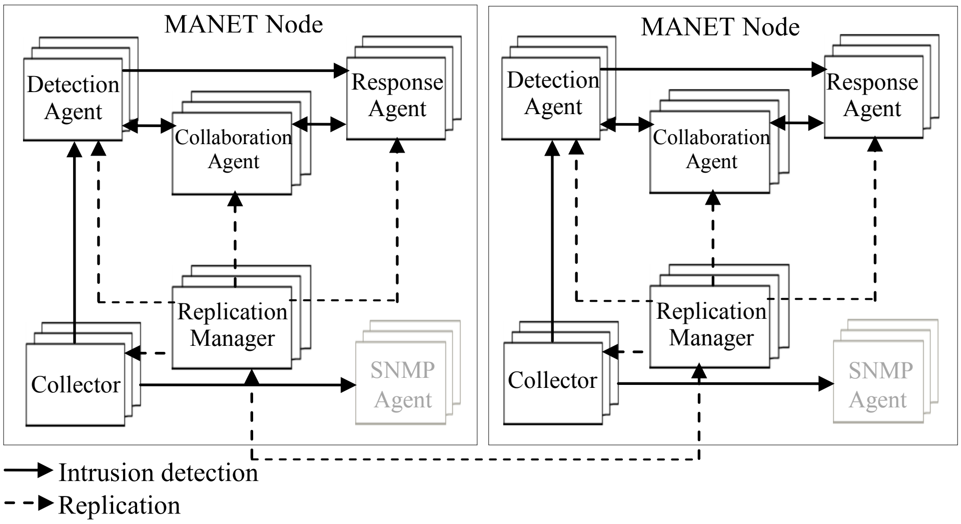
Since the initialization of MASID-R, each of its constituting agents will have a replica ready to take over at any moment. It is, indeed, very difficult to estimate the number of potential failures of an agent and thus the number of its needed replicas. Thus, instead of using several replicas (a replicate group) for each agent, we have introduced a new type of agents, which we call the replication manager. This latter enables us to get rid of the different problems related to replicate groups such as how many replicas to create for each agent, replicas' update and so on, in return of little overhead for the creation of replicas only when failures occur thereby introducing a constant trade-off between consistency and efficiency.

3.2.2 On-Demand Replication

On-demand replication means that it is only when failures occur that a new replica is created. Therefore, in case of an agent failure, simply a new replica is created and the already existing replica will take its place as an active agent in MASID-R, playing the role of the failed agent. This latter will no longer belong to our system i.e., it will be dropped out from the system. Briefly, MASID-R's state is rolled back to the most recent restoration point and restarted from there whenever needed.

In fact, a failed agent might not be able to carry out the replication process by itself. To address this problem, we suggest adding one more agent to MASID, which we call the replication manager. This agent has nothing to do with intrusion detection but it is rather responsible for observing and detecting failures within the multi-agent system constituting our IDS. Additionally, the replication manager dynamically adds or removes replicas, carries out the update of the current replicas and handles failure recovery within each IDS. Nevertheless, the replication manager might fail as well. For that, it also needs to be replicated. In that way, we can guarantee that if the replication manager fails, one of its replicas will continue to supervise the system. To handle the replication of the replication managers and to avoid having a single point of failure, each of them will serve as a supervisor for the replication managers on neighbouring nodes and vice versa. Figure 2 describes the proposed replication framework for MASID.

Figure 2 Replication Framework for MASID



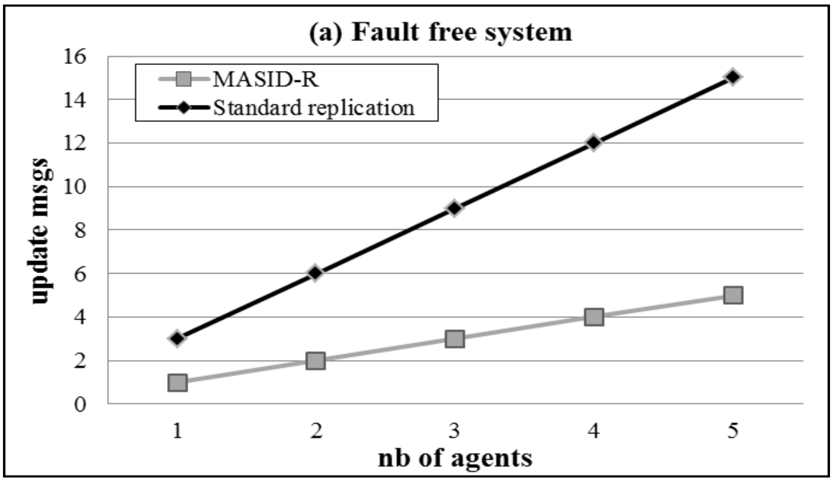
3.2.3 Consistency

The basic problem with replication techniques is that an update to any given logical object (in our case, active agent) must be propagated to all stored copies (replicas) of that object (Oo et al., 2010).

In our framework, we distinguish three types of agents. The first type is performing the required intrusion detection tasks. We call them the active agents. The second type of agents represents the replicas (one replica for each agent). The last type of agents is referred to as the replication manager. It is used to supervise and recover the active agents in case of their failure. These agents communicate using a peer-to-peer message-passing mechanism. As explained earlier, the Replication Manager is an agent that continuously observes the active part of the system, builds a state of the system and handles recovery whenever necessary thereby, guaranteeing consistency amongst active agents and their replicas.

Figure 3 presents a comparison between the cost, in term of the number of generated messages, of updating replicas using our approach and the one generated using standard replication approaches. For the sake of simplicity, we assume that, for the standard replication approach, the number *R* of replicas is the same for all the agents.

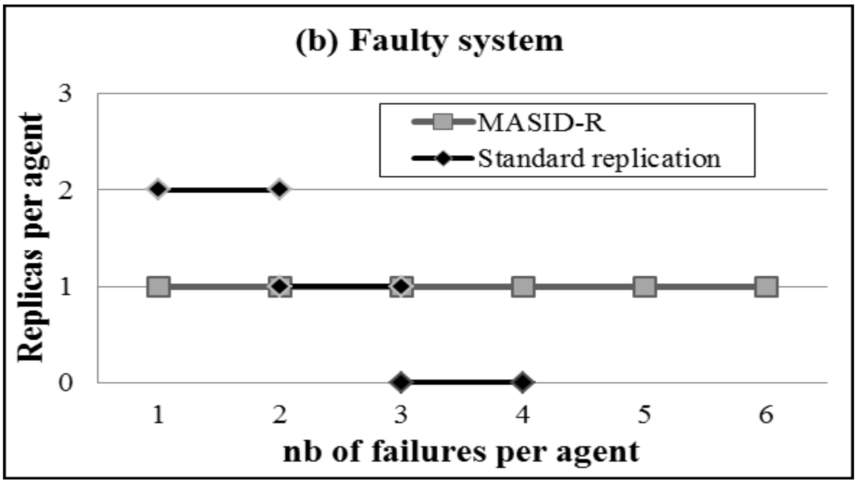
Figure 3 Consistency cost - Fault-free System



Compared to standard replication approaches, where the number of replicas' update messages depends not only on the number of updated agents but also on the number of each agent replicas, the number of update messages generated by MASID-R, is significantly reduced especially in case of large-scale systems. This was achieved through the introduction of replication managers as a new type of agents to substitute the traditional replicate groups. This not only reduces the cost of updating the replicas but takes the replicas' creation (how many?; where to place?; how to communicate?) burden off the system's designer. However, the creation of replicas at runtime is not without cost but still the cost of creating a replica, once a failure occurs, is negligible compared to the cost of updating *R* replicas each time the corresponding active agent's behaviour changes, especially for systems where agents change their behaviour continuously.

Figure 4 shows the impact of failures on the replicas' updating process. Contrary to systems incorporating standard replication approaches, where the system will inevitably crash if the number of failures is greater than the estimated number of replicas, MASID-R continues to work steadily whatever been the number of failures. More specifically, the proposed replication framework enabled our IDS to recover from individual and/or multiple agent failures, thereby guaranteeing permanent protection of the network on which it is installed. However, the network is not yet that reliable since data lost or altered due to intrusions is not recovered. For that, we would like to improve the reliability and consistency of the network, so as to enable it to heal itself of faults and to better survive malicious attacks.

Figure 4 Consistency Cost - Faulty System



In the following section, we present a recovery-oriented approach for a self-healing MANET based on MASID-R.

4 IDS-Based Self-Healing

Self-healing (Ghosh, 2007) can be defined as the property that enables a system to perceive that it is not operating correctly and, without (or with) human intervention, make the necessary adjustments to restore itself to normality. From that perspective, we divided the proposed self-healing process into two main phases. The first phase is Fault Detection and Damage Spread Stopping where the second will be Self-healing or Fault-repair. The next sub-sections discuss these phases.

4.1 Fault detection and damage spread stopping

The proposed healing process concerns the healing of faults and damages caused by intruders. Thus, the detection of faults and damages is dependent on the detection of intrusions. For that, we based our healing approach on the intrusion detection system MASID-R.

MASID-R is an agent-based distributed and cooperative IDS where every network node is equipped with a local IDS (LIDS) consisting of different but complementary agents: collector, the detection agent, collaborator, replication agents, and the response agent). Neighbouring LIDSs can communicate using mobile agents. Upon detection of an abnormal behaviour by the detection agent, the response agent will execute the necessary actions to stop the intrusion(s). These may include: dropping the connectivity to the potential intruder either permanently or for a limited period of time, informing other nodes about the detected intrusion and its potential source, and the update of both normal profiles and known attacks databases in case of unknown intrusions. To finish this phase, the detection agent will trigger the self-healing process by activating the healing agent.

4.2 Self-healing or fault-repair

The healing agent is a new stationary agent added to MASID-R to perform the necessary actions for the healing of the network. It has the ability to communicate with the other agents within the same LIDS. In the self-healing phase, this agent will use the information collected by the detection agent about the detected intrusion(s) (e.g., packet drop ratio, delay, victim node(s)' ID(s), intruder(s)' IDs, detection time, and so on) to measure the damage caused by the intruder(s). Then, building on the estimated level of damage, it will create and execute an appropriate list of actions to heal the network.

In fact, the healing agent will store information (backup information) about network traffic regularly (during the detection phase). Once an abnormal behaviour is detected by the detection agent, or a notification of a detected intrusion is received by the collaboration agent via the network, this will trigger the healing agent to start the healing or recovery process using both its backup data and data collected during the detection phase as illustrated in the example of Table 1.

Table 1 Example of IDS and healing data (case of a blackhole or grayhole attack)

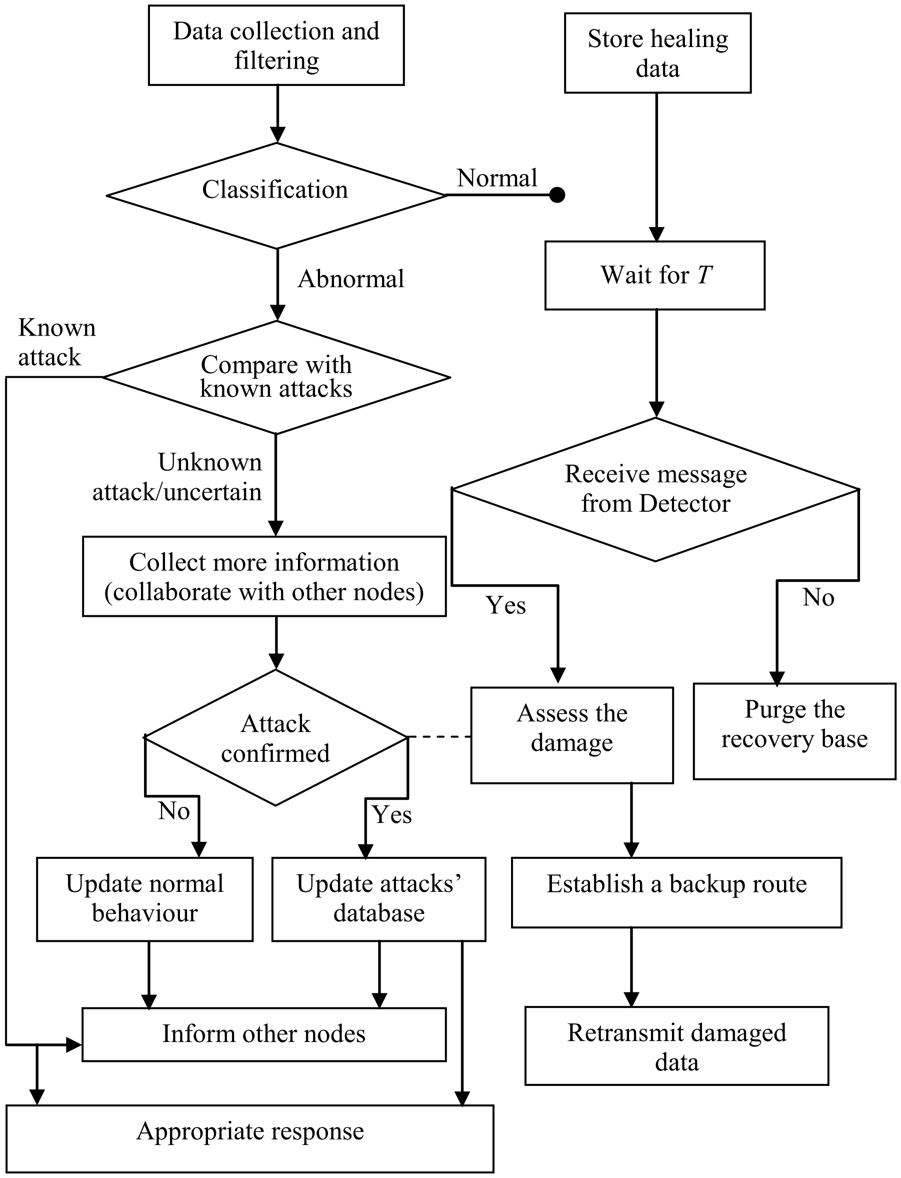
Detection data	Healing data
Node *I* is the intruder	Active routes having node *I* as a member.
x packets were dropped by node *I* during *T* (*T* is the active_detection interval of time).	Source and destination nodes’ IDs for each path (it knows the dropped packets were generated by node *S* and are destined to node *D*).
Detection time	Copy of the packets sent during *T*

The healing agent performs the following tasks:

- The node, on which the healing agent resides, should keep a copy of every sent packet during every active detection interval of time (detection session *T*).
- Receive messages about anomalous events from the detection agent: if no message is received from detector during *T*. Then the healing agent will purge the recovery base (i.e., it will delete the stored packets' copies from the recovery base at the end of the current detection session). Else, it will start the diagnosis and fault identification by using information contained in the received message (e.g., ID of the intruder, intrusion detection time, drop ratio, and so on).
- Repair the damage caused by the detected intrusive activities. This is a twofold task: the healing agent will first establish a new route, not including the intruder and the suspected nodes (if they exist), to replace the damaged route. Then, it will resend the stored packets to their destination via the newly established route.

  Steps for both phases are presented in the flowchart of figure 5.

Figure 5 Intrusion/Fault Detection and Self-healing Process



5 Experiments and Results

In order to evaluate the proposed approach, we carried out a series of simulation experiments using the network simulator *ns-2* (Fall and Varadhan, 2010). In these experiments, the proposed approach is validated against two packet dropping attacks, namely: blackhole and grayhole.

- Blackhole: an active DoS (Denial of Service) attack in which a malicious node exploits the routing protocols such as AODV (Ad-hoc On-demand Distance

Vector) to advertise itself as having a valid and good path to the destination node with the goal of dropping the absorbed packets.

- Grayhole: a variation of the blackhole attack in which the malicious node adopts a selective packet dropping.

The following subsections detail the simulation environment and metrics and discuss the obtained results.

5.1 Simulation Environment and Parameters

In order to evaluate our approach, we simulated a MANET using *ns-2*. It is an object oriented discrete event simulator, written in C++, with an OTcl (Object-oriented Tcl) interpreter as a frontend. It can simulate both wired and wireless network systems. Table 2 summarizes the different parameters related to our experiments.

Table 2 Simulation Parameters

Parameter	**Value**
Simulator	*ns*-2 (version 2.34)
Simulation time	120 s
Number of nodes	50
RP. for legitimate nodes	AODV
RP. for blackhole nodes	blackholeAODV
RP. for grayhole nodes	grayholeAODV
Traffic model	Constant Bit Rate (CBR)
Transport protocol	User Datagram Protocol (UDP)
Terrain area	1000 m × 1000 m
Maximum bandwidth	2 Mbps
Nb. of malicious nodes	Variable

5.2 Evaluation Metrics

To validate the efficiency of our approach, we consider the following metrics:

- Packet delivery ratio (PDR): it designates the ratio between the number of packets originated by the application layer CBR sources and the number of packets received by the final destination. Packet delivery ratio is important as it describes the loss rate that will be seen by the transport protocols. This metric characterizes both the completeness and correctness of our protocol.

$PDR = \sum \textit{received packets} / \sum \textit{sent packets}.$

- Average End-to-End Delay (E2E): it represents the average time taken by a data packet to arrive to its destination. This includes all delays caused during route acquisition and buffering at intermediate nodes. Only data packets that are successfully delivered to their destinations are counted. A lower value of the E2E delay means a better performance of the protocol.

  E2E =∑(arrive_time - send_time)/∑successful_connections

- Protocol Control Overhead (PCO): it is the ratio of the number of protocol control packets transmitted to the number of data packets received. Lower value of PCO means better performance of the studied protocol.

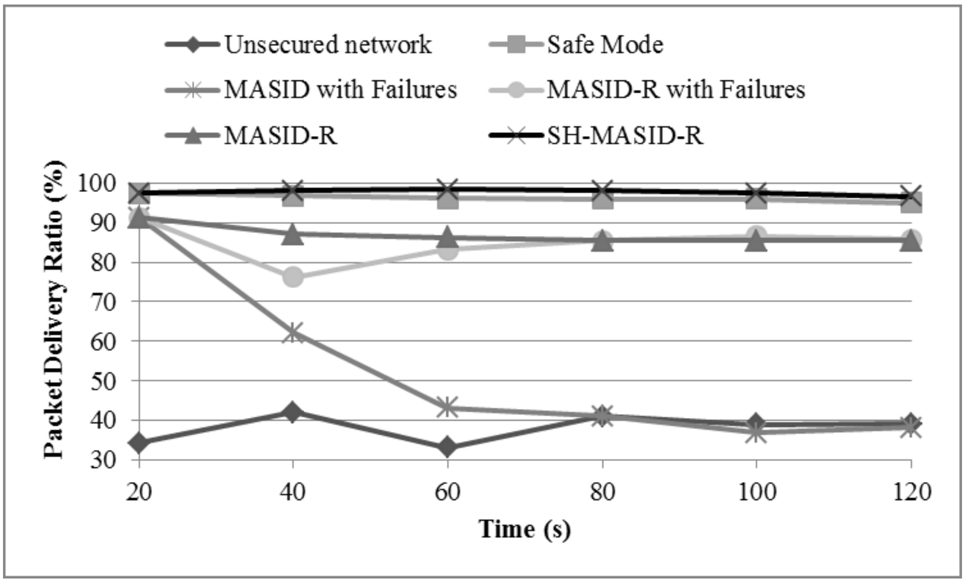
  PCO = ∑control packets/∑data packets

5.3 Experimental Results

In this subsection, we present and discuss the results of our study. As mentioned earlier, we used *ns-2* and simulated an ad-hoc network consisting of 50 mobile nodes. Also, we introduced malicious nodes in the network in the form of Blackhole nodes that drop all the absorbed packets and Grayhole nodes that adopt a random packet drop. We simulated the blackhole and grayhole attacks by modifying the original AODV routing protocol so that to generate a fake RREP (Route REPly) with the highest possible value of the sequence number and the smallest value of hop-count. In our simulations, for instance, we set the value of the sequence number of the RREP packet generated by the malicious nodes to 4294967295, and the hop-count is always set to 1. Each node in the network is assigned an initial position within the simulation dimensions *1000m* × *1000m* and joins the network at random. The MAC layer used for the simulations is IEEE 802.11. The simulation takes place for 120 seconds. Moreover, a failure of MASID and MASID-R is planned at 30 s, in some simulation scenarios, to demonstrate the feasibility of replicating agents within MASID. All the results are averaged over 10 simulation runs.

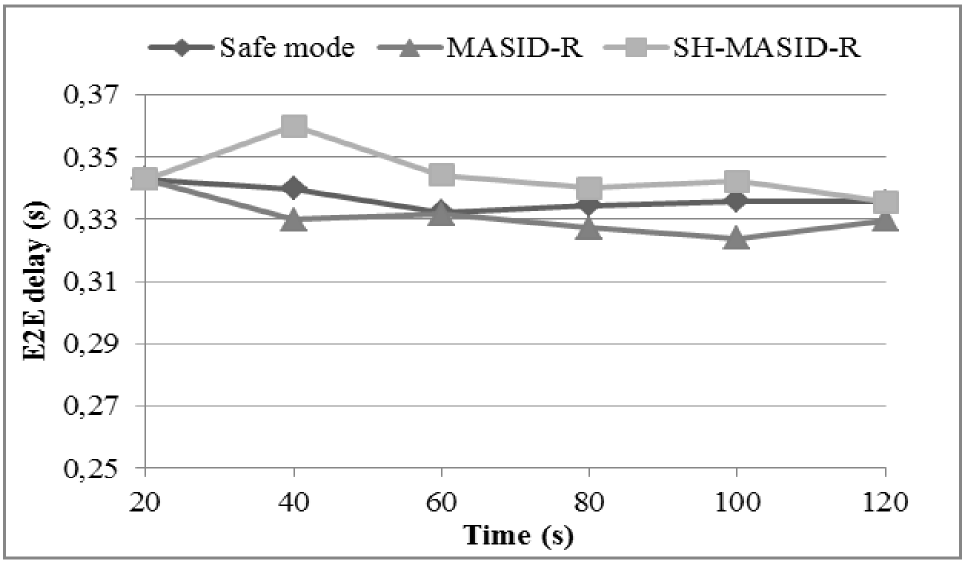
Figure 6 presents the evolution over time of the packet delivery ratio. In the presence of intrusions, PDR has notoriously increased through the use of MASID-R but a more considerable increase was achieved after the integration of the healing agent since even packets that were timed-out or dropped due to congestion could be restored. Upon failure occurrence, however, MASID will no longer be able to perform correctly leading to the success of intruders in dropping considerable amounts of packets whereas, MASID-R could heal itself, using its replication system thereby, guaranteeing a continuous protection of the network.

Figure 6 Packet delivery ratio vs. time



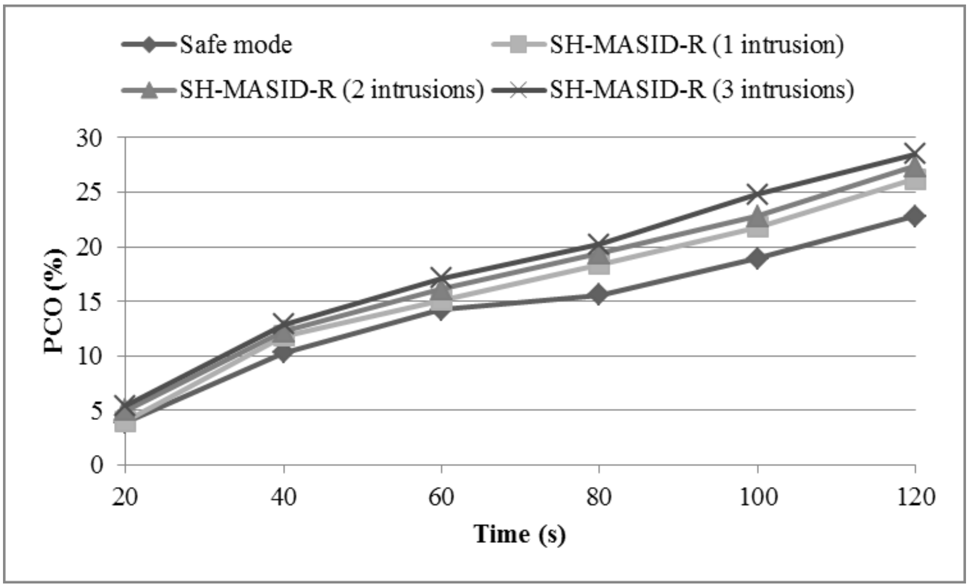
Unfortunately, to achieve these rates, it was necessary to create a kind of trade-off between guaranteeing the delivery of packets and both the overall communication time and the generated control overhead. More accurately, the healing approach tends to increase the ratio of correctly delivered packets at the cost of increased latency in the interrupted communication's delay resulted from the resubmission of the damaged packets as shown in Figure 7.

Figure 7 End-to-End Delay vs. time



In addition to the potential increase in the communications' delays, some traffic overhead may result due to the new route search as shown in Figure 8.

Figure 8 Packet Control Overhead vs. time



Fortunately, this overhead is proportional to the number of intrusions and their distribution over time, i.e., it increases with the increase in the number of intrusions and decreases if the risk disappears.

6 Conclusion

In this paper, we presented a twofold self-healing approach for MANET survivability. First, we designed a fault-tolerant IDS by replication of individual agents within MASID. This IDS, denoted MASID-R, is flexible as it possesses the ability to decide about when and which parts of the IDS should be replicated. Also, replication helps in overcoming individual agent failures which leads to a significant increase in our IDS's availability.

Then, a recovery-oriented approach is proposed to enable the supervised network to heal itself of those potentially caused faults and damages. The network is now sufficiently survivable as it can provide its services correctly even in the presence of intrusions and faults. For instance, a very high packet delivery ratio was achieved in return of few additional traffic overhead and proportional short delivery delays. Also, since recovery data is stored for limited short intervals of time, it will not overload the nodes on which it is stored. Moreover, the design of the recovery base in that flexible way facilitates greatly data manipulation (i.e., storage, extraction, and deletion), thereby minimizing both processing load and time.

As a future work, we tend to improve the healing ability of our intrusion detection system so that to heal the network of all kinds of damages and faults that can be caused by the potential intrusions.

References

Chen, J. and Wu, J. (2010) 'A survey on Cryptography Applied to Secure Mobile ad hoc networks and wireless sensor networks', *Handbook of Research on Developments and Trends in Wireless Sensor Networks: From Principle to Practice*, pp.226–289

Abusalah, L., Khokhar, A. and Guizani, M. (2013) 'A survey of secure mobile ad hoc routing protocols', *IEEE communications surveys & tutorials*, Vol. 10, No. 4, pp.78–93

Patil, J. A. and Sidnal, N. (2013) 'Survey - secure routing protocols of MANET', *International Journal of Applied Information Systems*, Vol. 5, No. 4, pp.8–15

Mechtri, L., Djemili, F.T. and Ghanemi, S. (2012) 'MASID: Multi-Agent System for Intrusion Detection in MANET', *9th International Conference on Information Technology- New Generations (ITNG' 2012), IEEE Computer Society, Washington, DC, USA*, pp. 65–70.

Kaur, P., Rattan, D. and Bhardwaj, A.K. (2010) 'An Analysis of Mechanisms for Making IDS Fault Tolerant', *International Journal of Computer Applications*, Vol. 1, No. 24, pp. 22–25

Sen, J. (2010) 'A robust and fault-tolerant distributed intrusion detection system', *Parallel Distributed and Grid Computing*, pp.123–128

Zhihao, P. and Guanyu, L. (2012) 'An Intelligent Immunity-based Model for Distributed Intrusion Detection', *Journal of Computational Information Systems, Binary Information Press*, Vol. 8, No. 24, pp.10123–10130

Sasikumar, R. and Manjula, D. (2012) 'Dynamic Distributed Intrusion Detection System Based on Mobile Agents with Fault Tolerance', *Journal of Computer Science, Science Publications*, Vol. 8, No. 7, pp.1092–1098

Chang, K. and Shin, K.G. (2010) 'Application-Layer Intrusion Detection in MANETs', *43rd Hawaii International Conference on System Sciences, Honolulu, HI*, pp.1–10.

Elsadig, M. and Abdullah, A. (2009) 'Biological inspired intrusion prevention and selfhealing system for network security based on danger theory', *International Journal of Video & Image Processing and Network Security*, Vol. 9, No. 9, pp.16–28

Lee, C-H. and Suzuki, J. (2008) 'SWAT: a decentralized self-healing mechanism for wormhole attacks in wireless sensor networks', *Handbook on Sensor Networks, Xiao,Y., Chen, H., Li, F. Ed.*, pp.01–21

Kong, J., Hong, X., Yi, Y., Park, J-S., Liu, J. and Gerla, M. (2005) 'A secure adhoc routing approach using localized self healing communities', *MobiHoc'05, USA*, pp.254–265.

Jain, P., Singh, P. K., and Abraham, A. (2011) 'Intrusion detection and self healing model for network security', *7th International Conference on Next Generation Web Services Practices*, pp.320–325.

Fedoruk, A. and Deters, R. (2002) 'Improving Fault-Tolerance by Replicating Agents', *AAMAS'02, ACM Press, Bologna, Italy*, pp.737–744.

Budhiraja, N., Marzullo, K., Schneider, F.B. and Toueg, S. (1993) 'The primary-backup approach', *Distributed Systems 2nd ed., Addison-Wesley Books, ACM Press*, pp.199–215

Schneider, F.B. (1990) 'Implementing Fault-Tolerant Services Using the State Machine Approach: A Tutorial', *ACM Computing Surveys*, Vol. 22, No. 4, pp.299–319

Oo, M.M., Soe, T.T. and Thida, A. (2010) 'Fault Tolerance by Replication of Distributed Database in P2P System using Agent Approach', *International Journal of Computers*, Vol. 4, No. 1.

Ghosh, D., Sharman, R., Rao, H.R. and Upadhyaya, S. (2007) 'Self-healing systems-survey and synthesis', *Decision Support Systems*, Vol. 42, pp.2164–2185

Fall, K. and Varadhan, K. The ns Manual, http://www.isi.edu/nsnam/ns/ns-documentation