An Architecture for Self-protection in Internet of Things

Ruan de A. C. Mello, Admilson de R. L. Ribeiro, Fernando M. de Almeida, Edward D. Moreno
Department of Computing
Federal University Sergipe UFS
São Cristóvão, Brazil
E-mails: ruanmello@gmail.com, admilson@ufs.br, fernando.m.al.91@gmail.com, edwdavid@gmail.com

***Abstract*—To make the Internet of Things a more receptive environment and well regarded by everyone, it is important to invest in security. The devices involved in the Internet of things environment have limited computational resources and expose the network to many threats. With the use of an architecture to be able to detect, classify and mitigate the effects of these threats, it is possible to create a safer environment. This paper proposes a security architecture for the Internet of Things, considering the limited computational resources of the environment in question. This Architecture uses the Dendritic Cells Algorithm (DCA) combined with a neural network to detect attacks and use the White List to ignore the malicious nodes in the network.**

Keywords–Internet of Things; Autonomic Computing; Self-Protection.

I. INTRODUCTION

Next ages of computing will tend to be beyond the traditional work environment. The Internet of Things (IoT) is a recent paradigm that makes part of this new age and its main goal is to create the possibility of communication between people and things and also between things without the need of human intervention [1].

Atzori et al. [2] calls attention to the impact of the fast advance of IoT and the great challenges that follow such as the interoperability between devices, the limited computational resources and especially the security. According to Roman et al. [3] the Internet and the users are under constant attacks, but there are many business models that try to provide the ethical and safe use of the Internet. The IoT environment is considered more vulnerable than the conventional Internet [2]. This occurs because when the wireless network has many nodes, it becomes easier for both physical and logical attacks, since it is not possible to apply one complex security mechanism due to lack of computational resources. There are many malicious models and many others will emerge associated with this environment. The challenge is to prevent the growth of such models or at least to minimize their impact.

Dobson [4] highlights the relevance of the autonomic characteristics, considering the growing amount of devices interconnected in the IoT environment. In 2011, the number of connected devices already exceeded the real number of people around the world and it is estimated that in 2020 this number will reach 24 billion.

The Vice President - Senior of International Business Machines (IBM), Paul Horn, introduced in March 2001, for the first time the use of the term Autonomic Computing [5]. Horn deliberately chose a term with a biological connotation trying to compare in this manifesto the need of self-management in complex systems aimed to reduce the burden on system administrators with the way that the autonomic nervous system regulates the heart beat and body temperature [5]. Thus, the conscious brain will be released from the burden of dealing with these and many other functions that can be considered low-level, but vital for your functioning. In this manifesto, he presented the four properties of self-management: self-configuring, self-optimizing, self-healing and self-protection.

Our work completes the architecture proposed by Almeida et al. [6] addressing the self-protection on the IoT. The proposed architecture consists of five modules. The tasks are distributed among the modules in order to share the responsibilities. By dividing responsibilities between modules, it makes easier to design self-protection systems for the IoT environment. This Architecture uses the dendritic cell algorithm (DCA) combined with a Multilayer Perceptron Neural Network with Limited Weights (MLPLW) to detect attacks in the network. The MLPLW can learn non-linear patterns during the execution, making the attack detection more efficient. In case of attack on the network of IoT, the architecture proposed will discover the type of attack and minimize the damage.

The remainder of this document has seven sections: Section II presents five common types of attacks to the IoT environment that will come to be mitigated with the proposed architecture; Section III introduces some aspects of autonomic computing and presents the autonomic loop MAPE-K; Section IV presents the Dendritic Cells Algorithm; Section V describes the architecture proposed; Section VI presents two related works to Internet security and the system of self-protection; Section VII presents the initial results; Section VIII concludes the paper.

II. ATTACKS IN INTERNET OF THINGS

According to Atzori [2] the characteristics of IoT (low power, limited energy and limited resources) expose the network to many threats. Most of them attack the limited power of the sensors. In other cases these threats modified or deleted some data. Following this section, we will discuss some of the latest and more common attacks on the environment of the IoT and wireless sensor networks [7].

1) Sinkhole: In a Sinkhole attack, the attacker tries to attract all the traffic from neighboring nodes [8]. So, practically, the attacker node listens to all data transmitted from neighboring nodes. Only this attack does not cause too much damage in the network, but together with another type of attack (Selective Forward or Black Hole), can become very powerful.

 ISBN: 978-1-61208-514-2

2) Selective forward: In a Selective Forward attack, the attacker node receives the transmission packets, but refuses to transmit some of them and drops those that it refused to transmit. The attacker must choose which packets to discard according to some standard such as size, destination or origin [9]. In this case, only the packets released by the attacker node can be freely transmitted.

3) Black Hole: In a Black Hole attack, the attacker node receives the transmission packets and drops all packets received, regardless of type, size, origin or destination [9].

4) Flooding: There are vulnerabilities related to the exhaustion memory. One manner to take advantage of this vulnerability is when an opponent sends too many requests trying to connect to the victim, every request makes the victim allocate the resources in an attempt to maintain the connection [10]. Thus, to prevent the total resource depletion is necessary to limit the number of connections. However, this solution also prevents valid nodes to create a connection with the victim, causing problems such as queuing [10].

5) Hello Flood: The attack Hello Flood uses a device with a powerful signal to regularly send some messages; that way, the network is left in a state of confusion [7]. In order to find ad-hoc networks, many protocols use Hello Messages for discovering neighbor nodes and automatically create a network. With the Hello Flood attack, an attacker can use a device with high transmission power to convince every other node in the network that the attacker is its neighbor, but these nodes are far away from the attacker. In this case the power consumption of sensors is significantly increased, because of protocols that depend on exchange information between neighbor nodes for topology maintenance or flow control [7].

Previously, we saw some of the most common attacks on IoT networks and in the next topics will be analyzed the possible strategies to end or to mitigate the damage caused by them.

To stop the damages caused by attacks on a network, first it is necessary to detect these attacks, using an intrusion detection system (IDS). An IDS analyzes network activity and attempts to detect any unusual behavior that may affect the integrity of the network. Based on information provided by IDS, strategies are created to cope the attacks. For example:

- To mitigate Sinkhole - If the geographical locations of the nodes of RPL DODAG are known, the effect of Sinkhole attacks can be mitigated by the use of flow control, making sure, that the messages are traveling to the correct destination. The RPL protocol also supports multiple instances DODAG offering alternative routes to the root DODAG [11].
- To mitigate Hello Flood - A simple solution to this attack, it is perform a bidirectional check for each message "HELLO" [12]. If there is no recognition, the path is assumed to be bad and a different route is chosen. If geographical locations of the nodes of RPL DODAG are known, all packets received from a node that is far beyond of the common network node transmission capacity can be dropped.
- To mitigate Selective Forward - An effective countermeasure against Selective Forward attacks is to ensure that the attacker cannot distinguish the different type of packets, forcing the attacker to send all or none packets [13].

Raza et al. [14] said that the most efficient and fastest way to stop the damage of routing attacks is isolate the malicious node. Some forms to ignore the attacker node were studied. These forms are:

- The Black List: After identifying the nodes and find the attackers, it will be created a list and all the malicious nodes will be added in order to exclude them from the possible routes of traffic data. To ignore the attacker, it will be done a verification in the Black List excluding all nodes found of the typical RPL DODAG that have a root and multiple nodes.
- The Gray List: After identifying the nodes and find the attacker, it will be created a list. The suspicious attacker node will be added to this list with the intention of excluding it from the possible routes of traffic data, for a predetermined time. After the end of the predetermined time the suspicious attacker node is deleted from the list. In this way, if have any doubt about the identification of the attacker node, the node may re-join the network. To ignore the suspicious attacker nodes, when create the routing, it will be done a verification in the Gray List excluding all nodes found of the typical RPL DODAG that have a root and multiple nodes.
- The White List: As in the example of the Black List, it will be created a list after identifying the nodes and find the attacker node. But this time will be added into the White List only the valid nodes and all malicious nodes will be excluded. This way will have a verification stating which nodes are valid and must belong to a typical RPL DODAG with a root and several nodes.

III. AUTONOMIC COMPUTING

The initiative of autonomic computing proposed by IBM, is based on the human nervous system [4]. The idea is, as the body's mechanisms that have functions with self-management and do not demand any conscious act such as heartbeat or intestinal activity, a computer system creates mechanisms that will also allow it to have a self-management [4].

According to the manifest proposed by IBM in 2001, there are four self-managing properties: self-configuration, self-healing, self-optimizing and self-protection [5].

- Self-Configuration Through self-configuring autonomic system will be installed and set up to attend the high level predetermined policies according to user intentions.
- Self-Healing Through this property it is possible for autonomic systems detect, diagnose and repair local problems resulting from bugs or failures in software and hardware.
- Self-Optimization The property of self-optimization is present on systems that can perform some change, proactively, to improve the performance and quality of service.

 ISBN: 978-1-61208-514-2

- Self-Protection The self-protection property is present in the systems that can defend themselves from malicious attacks and unauthorized changes. The autonomic system with self-protection is used to prevent and anticipate security breaches.

Dobson [4] say that self-management mechanisms in the autonomic computing are not independent entities. For example, the success of an attack against the system, requires actions of self-healing, self-configuration and self-optimization initially to ensure, that the system will have a trusted operation. After that, the self-protection would have responsibility for dealing with similar attacks in the future.

A. MAPE-K Autonomic Loop

The MAPE-K loop was presented by IBM as a reference model. Composed of five modules that can be seen in Fig. 1, the MAPEK-K Loop is intended to distribute the tasks of each element of the autonomic computing [5]. The modules that build the MAPE-K Loop are, respectively, monitoring, analysis, planning, execution and knowledge.

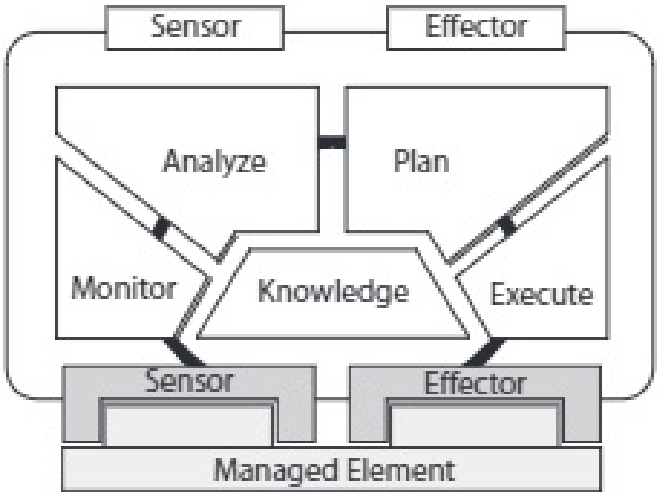

Figure 1. Mape-K Autonomic Control Loop [15].

- The Monitoring module uses sensors to collect data from the managed element, which could be a software or hardware resource, or an autonomic manager itself.
- The Analysis module provides mechanisms to interpret the collected data from the monitoring phase and predict future situations.
- The Planning module builds the necessary actions to achieve the goals.
- The Execution module uses effects to make changes on managed elements.
- The Knowledge module is in charge of keeping relevant data in the memory to accelerate decision making.

IV. DENDRITIC CELL ALGORITHM

The Dendritic Cell Algorithm (DCA) was introduced by Greensmith [16] and is inspired by the Danger Theory regarding the human Immune System. The main elements of DCA are Dendritic Cells (DC), the lymph node and antigens. The DC input signals are signs of danger, safe signals, PAMP signals, signs of inflammation. The DC output signals are: migration signal (costimulatory Molecules-CSM), semi-mature signal and mature signal.

Iteratively, the antigens are presented to DC. All signals increment the migration signal indicating when the dendritic cells will migrate to the lymph node and being processed. The danger signs and PAMP increment the mature signal of the DC while the safe signal increments the semi-mature signal of the DC. The sign of inflammation potentiates the growth of all signals.

When DC reaches the migration threshold is sent to the lymph node, and DC will be labeled how mature if the mature signal is greater than semi-mature or semi-mature otherwise. When the lymph node has a number of DCs, the antigens anomaly index is calculated, the Mature Context Antigen Value (MCAV) from the equation (1) where M is the number of mature DCs and SM the number of semi- mature DCs.

$$MCAV = \frac{M}{(SM + M)} \quad (1)$$

If MCAV has a value greater than the threshold, the DCA detects the presence of an intruder.

V. SELF-PROTECTION ARCHITECTURE

The Self-protection architecture for the IoT proposed in this work has five modules (Monitoring, Analysis, Planning, Executing and Knowledge) and was based on the MAPE-K loop. It is important to remember that the main contribution of this work is the implementation of the missing modules (Planning and Execution). This way, we will be completing the architecture proposed by Almeida et al. [6].

The monitoring and analysis modules are responsible, respectively, for collecting through the sensors, some information of the network that will be analyzed to measure the possibility of being associated with an attack. These two modules are present in the network nodes and in the border router (6BR). The planning and execution modules will be responsible, respectively, for identifying the attacker, the type of attack and to mitigate the damage in the network. The information listed as relevant data for analysis, planning and execution is: type of transport protocol, type of application protocol, time of communication, number of messages sent, number of messages effectively sent and number of messages received.

In Fig. 2, we can see that the border node (6BR) will have all components. This occurs because the 6BR has sensing and also for being the main element of the network. The components of the monitoring module and a part of the analysis module components are present in the network nodes that have the self-protection system. The components of the planning and execution modules are present only in 6BR, but the need to distribute them among the other network nodes can be considered. The five modules present in the architecture will be described more clearly in the following subsections.

1) Monitoring Phase: The components responsible for the monitoring phase are present in all network nodes, especially in 6BR. At this stage, the network information and the nodes are monitored. Some important information collected during the sensing to future analysis is: number of successfully sent

 ISBN: 978-1-61208-514-2

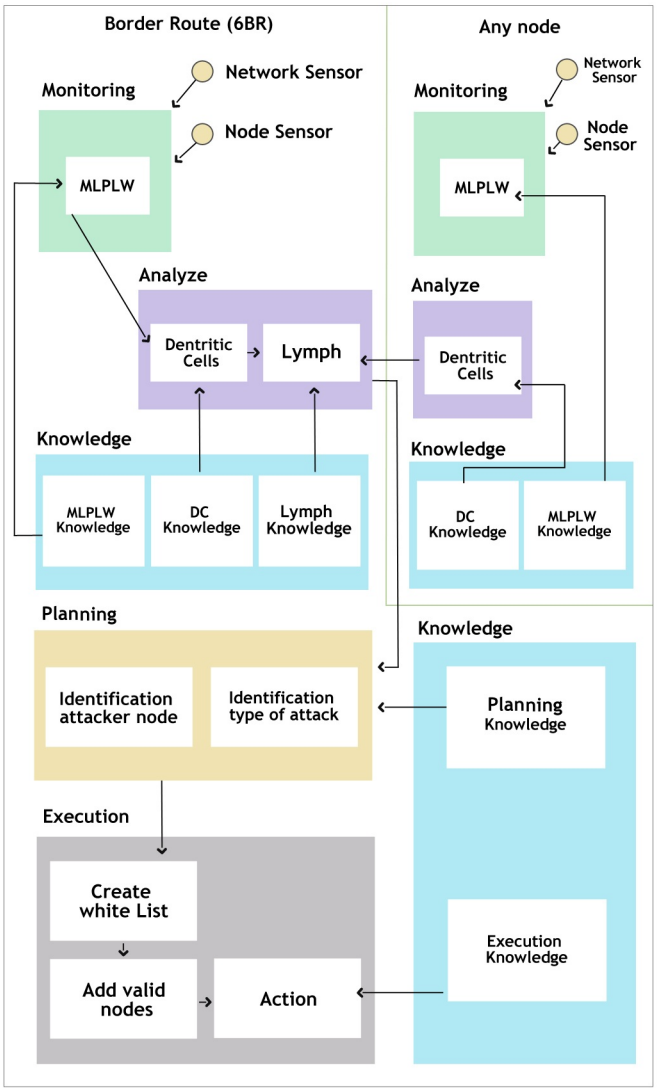

Figure 2. Self-Protection Architecture.

packets, number of sent packets, number of lost packets and the information about routing in DODAG tree. The main component of the monitoring phase, in this architecture proposed, is the Multiplier Perceptron Neural Network with Limited Weights (MLPLW), that can learn non-linear patterns during the execution. The MLPLW in this architecture is used to predict what information received indicates a problem in the node or in the network, for example, the MLPLW can predict the decrease of successfully sent packets from the packet sender address, if the training is enough. The prediction of the MLPLW is sent to the Analysis phase, to the Dendritic Cell Algorithm.

2) Analysis Phase: The component responsible for analysis phase uses the Dendritic Cell Algorithm (DCA), which is inspired by the danger theory of the human immune system. This algorithm has a high distributive potential and a low false positive rate. The DCA is used to analyze the results received from the MLPLW and determine if an attack is happening. The Dendritic Cells collect information from the MLPLW and migrates to the Lymph when the cell becomes mature or semi-mature. The Lymph receives mature and semi-mature dendritic cells and determine if the system is in danger. This information indicates an attack or not and, if there is an attack, it is sent to the Planning phase.

3) Planning Phase: The component responsible for the planning phase needs confirmation that really exist the presence of attack on the network, otherwise it becomes obsolete. Because the main objective is to find out what type of attack occurred on the network and to identify the attacker node through the information acquired by the analysis module. Thus, the execution module will decide the most appropriate action to be taken to mitigate the damage caused by the attack. The type of attack will be classified according to the following set of options: route selection group, packets control group and exhaustion attack group.

Through the obtained results, it is possible to differentiate the type of attack occurred. For example:

- If an attack belongs to the route selection group (Sinkhole), it will be taken into account the flow of messages in the legitimate nodes and in the attackers nodes. In this case, it will be needed to evaluate the influence of attackers in the choice of a better route. For this, it is necessary to compare the flow of messages from legitimate nodes regarding to the attackers nodes, through a medium and total number of messages.
- If an attack that belongs to the packet control group, which includes messages disposal (Black Hole and Selective Forward), it will be taken into account the number of messages sent, messages effectively sent and messages received.
- If an attack that belongs to an exhaustion attack group (Hello Flood and Flooding), it will be taken into account the number of packets sent by the attacker node, which is intended to send a huge number of messages to overload the system.

4) Execution Phase: The component responsible for the execution phase should mitigate or stop the damage caused by the attacks occurred on the network. The type of attack and the identification of the attacker node will be information that will influence in the choice of the predetermined action to mitigate or stop the damage in the network. These two important information will be provided by the Planning Phase. The reason to find out the attacker node is, trying to isolate as quickly as possible and create a new route, thus, avoid further damage to the network. The type of attack will be among one of the three groups mentioned above. According to the group selected there will be a specific action to solve the problem, because at each type of attack causes different types of damage on the network.

The first action to be taken by the components of the execution module, since the attack was detected, it is to ignore the malicious node. To perform this action it is important to be made the identification of network nodes, legitimate and malicious. Raza et al. [14] say that it is necessary to be careful with the way of identifying nodes. If possible it should be avoided the identification by IP address or MAC address, because they can be easily falsified. After making the identification of the nodes, some ways to ignore the attacker

 ISBN: 978-1-61208-514-2

node were studied. Three ways to isolate the malicious node were analyzed before choosing the most convenient. The three ways are: Black List, Gray List and White List.

The way chosen was the White List, because the maintenance of this list is simple. This way, all valid nodes will recalculate their rank in the RPL protocol (DODAG). To recalculate the rank of all valid nodes, it will be necessary to ignore the DODAG Information Object (DIO) of all nodes with higher rank than theirs and of nodes that are not present in the White List. Thus, for a stranger node join in the network, it should be reported as safe and added in the White List.

5) Knowledge Phase: The components of knowledge phase will be responsible for keeping all the knowledge acquired by the system. Knowledge about the planning and execution modules will be at 6BR. The information kept by the Knowledge Module will be used to facilitate and accelerate the discover of the type of attack, the attacker node and the action to be taken to protect the network.

VI. RELATED WORK

The related works listed in this paper not only detect the attack, but also tries to mitigate the damage caused by the attacker. The related works will be described more clearly in the following subsections.

A. CAD

In [9], the authors describe how they developed a way to detect, identify the attacker node and classify the attack in a Wireless Mesh Networks (Wireless Mesh Networks - WMNs). They considered a special case of denial of service (DoS), known as Selective Forward, where the attacker node behaving maliciously pushing forward only a subset of the packets that received but discards the others.

While most studies about Selective Forward focuses on attack detection assuming a wireless channel, error free, the authors Shila et al. [9] considered more interesting the challenging scenario where the fall of the package may be due to an attack or normal loss events, such as the collision of access to the medium or bad quality of the channel.

Specifically, they developed an algorithm that can effectively distinguish the Selective Forward from the packet losses occurring in normal loss events. The developed algorithm is called CAD (Channel Aware Detection) and is based on two strategies, channel estimation and traffic monitoring. If the loss rate monitored in certain jumps exceeds the normal rate of estimated loss, the nodes involved will be identified as attackers. In addition, they perform analytical studies to determine the optimal detection thresholds that minimize the sum of false alarm and missed detection probabilities.

The traffic control procedure works basically that way, each intermediate node, monitors along the path the behavior of the neighbors. Given a path determined by the routing protocol, the CAD sends messages along the way to probe, to detect possible attacks. In the event of a positive detection, CAD, then triggers the underlying routing protocol to activate a process for finding out a new route.

CAD design was used by Network Simulator NS2 Berkeley (v2.29) for simulations. By the end of the simulation, using the CAD algorithm, the results showed that, in the presence of normal loss events or without, CAD can detect and classify the attack, increase the packet delivery rate in the network and finding out attacker nodes.

B. SVELT

SVELT, name given by authors, Raza et al. [14] to Intrusion Detector System for the Internet of Things projected by them, is applied in 6LoWPAN networks that use the RPL routing protocol. The approach of SVELTE is distributed and centralized, inserting modules in the edge router (6BR) and network nodes. The three main modules SVELTE are 6LoWPAN Mapper, Component Detector Intruder and distributed mini - firewall. The Mapper 6LoWPAN (6Mapper) reconstructs the routing tree RPL protocol (DODAG) in 6BR, each network node has a client for the 6Mapper. To rebuild the tree DODAG, you must periodically send requests to the 6Mapper customers, who respond to their corresponding information: NodeID, ParentID, Neighbor IDs and ranks of Neighbors IDs. It should be considered whether the network uses some form of reliable communication authentication, so you can reduce the size of requests and responses.

Once the SVELT detects an attack, the goal is to mitigate its effects and remove the intruder from the network, Raza et al. [14] say that the simplest approach to remove an attacker is to ignore it. Taking this approach requires the identification of the attacker node. The authors adopted, as a way to ignore the malicious node the White List, which would include all valid nodes and would reject malicious nodes. This method is reliable and easy to maintain in the presence of many attackers. As a result, in SVELTE a white list is used. The informed location of the nodes will also help mitigate the SVELTE Sybil and CloneID attacks intended to disrupt the routing information forging identities.

Implementations of SVELTE and mini-firewall were made in ContikiOS, the code is open and available. Raza et al. [14] used the ContikiOS and its RPL implementations of 6LoWPAN and IP stack. The evaluation was made empirically using Cooja, the network simulator of the ContikiOS, measuring the rate of detection, true positives and false positives for each experiment.

The authors concluded that the SVELTE is very effective for Sinkhole and Selective Forward attacks on a network with fewer losses. When the network has more losses, the system has better results when the RPL network becomes stable. Getting rates close to 90% on a network with losses when the RPL network stabilizes and results close to 100% on a network without loss. As for energy consumption, SVELTE solution consumes 30% more energy than using only the RPL. As the consumption of memory, 6Mapper client has 1414 bytes of overhead, the Firewall client has 246 bytes, 6Mapper server varies with the number of nodes and neighbors: 3580 bytes for one node and one neighboring, 3846 bytes into 8 nodes and 1 neighbor, 4152 bytes into 16 nodes and 1 neighbor and 4724 bytes into 16 nodes and 8 neighbors.

C. Comparison of Related Work

The SVELTE, as the authors claim, is the first IDS (Intrusion Detector System) for the IoT. The work presented has a huge contribution to design an IDS with the characteristics of a network for IoT, considering the technologies used in the communications stack, such as 6LoWPAN and RPL routing

 ISBN: 978-1-61208-514-2

protocol. However its approach does not have autonomic characteristics for auto protect the network from further attacks, only the determined attack on the network project level.

TABLE I. COMPARISON OF RELATED WORK.

Qualities	*SVELT*	*CAD*	*Architecture proposal*
Designed for IoT	X	-	X
Autonomicity	-	X	X
Mitigate SinkHole	X	X	X
Mitigate Selective Foward	X	X	X
Mitigate BlackHole	X	X	X
Mitigate Flooding	-	-	X
Mitigate Hello Flood	-	-	X

Like the SVELT, CAD not only detects the attack, but also tries to mitigate the damage caused by the attacker. The CAD is directed to the WMNs and can differentiate between losses occurring in the normal events of a legitimate attack Selective Forward efficiently. In Table 1, we present the differences between the related works and the proposed architecture.

VII. RESULTS

The initial results of this research are about the technique used in the monitoring phase. The chosen technique for the first efforts is an ANN (Artificial Neural Network), a MLPLW based on the neural network with limited precision weights [17].

The MLPLW implemented has 10 neurons in the hidden layer and each weight is represented by a byte. The training technique of the MLPLW is the QBPSS (Quantized Back-Propagation Step-by-Step) [17], a modified version of Back-Propagation for neural network with limited weights.

TABLE II. MEMORY CONSUMPTION.

Resource	*MLP*	*MLPLW*	*MLPLW with training*
ROM Memory	214 bytes	354 bytes	1716 bytes
RAM Memory	3360 bytes	420 bytes	420 bytes

The KDD99 dataset it is widely used in Intrusion Detection Systems. Thus, KDD99 dataset was used with our MLPLW ANN implementation with a stream based training [18]. Each input is used once and the accuracy and false positive rates were measured every thousand inputs. After the first thousand inputs, the MLPLW achieved 97,65% accuracy, but oscillated until the thirty-fourth thousand input. The oscillation of the accuracy rate of the MLPLW is depicted in Fig. 3.

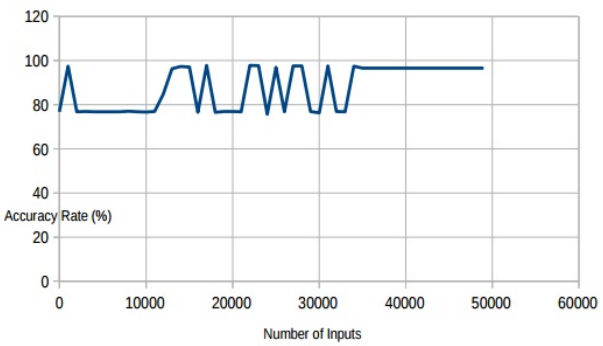

Figure 3. Accuracy rate of MLPLW.

As our proposed architecture shall be used in the Internet of Things context, the used techniques should be in accordance to the use of resources. It is possible to see in Table II, that the MLPLW have a small impact in memory utilization in an ARM Cortex-M3 with 512 KB of Persistent memory and 64 KB of Volatile memory.

VIII. CONCLUSION

The completion of this proposal will help provide a self-protection mechanism for IoT networks, facilitate detection and the classification of possible attacks on smart devices, mitigate the damage of the attacks suffered ensuring better performance and increase the confidence of users when using devices connected to IoT network.

The architecture proposed deals with five different attacks (Sinkhole, Selective forward, Black Hole, Flooding and Hello Flood) bearing in mind the memory consumption and energy due to a lack of resource of the available devices in the IoT environment. This Architecture uses the Dendritic Cell Algorithm combined with a neural network (MLPLW) to detect attacks and use the White List to ignore the malicious nodes in the network.

The MLPLW implemented shows that the monitor phase of the proposed architecture can use less than 1% of the memory of the embedded system and still have a high accuracy rate.

For future work, there will be the implementation of the proposed architecture to validate it. The performance of the system should be evaluated to verify if the proposed architecture has better results than related work. New attacks and new technologies will emerge, and then the work in question may be extended becoming increasingly complete.

ACKNOWLEDGMENT

This work was supported by CAPES and FAPITEC/SE

REFERENCES

[1] J. Gubbi, R. Buyya, S. Marusic, and M. Palaniswami, “Internet of things (iot): A vision, architectural elements, and future directions,” Future Generation Computer Systems, vol. 29, September 2013, pp. 1645–1660, doi:10.1016/j.future.2013.01.010.

[2] L. Atzori, A. Iera, and G. Morabito, “The internet of things: A survey,” Computer networks, vol. 54, October 2010, pp. 2787–2805, doi:10.1016/j.comnet.2010.05.010.

[3] R. Roman, P. Najera, and J. Lopez, “Securing the internet of things,” Computer, vol. 44, September 2011, pp. 51–58, doi:10.1109/MC.2011.291.

[4] S. Dobson, R. Sterritt, P. Nixon, and M. Hinchey, “Fulfilling the vision of autonomic computing,” IEEE Computer, January 2010, doi:10.1109/MC.2010.14.

[5] J. O. Kephart and D. M. Chess, “The vision of autonomic computing,” Computer, vol. 36, January 2003, pp. 41–50, doi:10.1109/MC.2003.1160055.

[6] F. M. de Almeida, A. d. R. L. Ribeiro, and E. D. Moreno, “An architecture for self-healing in internet of things,” UBICOMM 2015, July 2015, p. 89.

[7] D. Martins and H. Guyennet, “Wireless sensor network attacks and security mechanisms: A short survey,” in Network-Based Information Systems (NBiS), 2010 13th International Conference on. IEEE, November 2010, pp. 313–320, doi:10.1109/NBiS.2010.11.

[8] P. Goyal, S. Batra, and A. Singh, “A literature review of security attack in mobile ad-hoc networks,” International Journal of Computer Applications, vol. 9, November 2010, pp. 11–15.

 ISBN: 978-1-61208-514-2

[9] D. M. Shila, Y. Cheng, and T. Anjali, "Mitigating selective forwarding attacks with a channel-aware approach in wmns," Wireless Communications, IEEE Transactions on, vol. 9, May 2010, pp. 1661–1675, doi:10.1109/TWC.2010.05.090700.

[10] A. D. Wood and J. A. Stankovic, "Denial of service in sensor networks," Computer, vol. 35, December 2002, pp. 54–62, doi:10.1109/MC.2002.1039518.

[11] T. Heer, O. Garcia-Morchon, R. Hummen, S. L. Keoh, S. S. Kumar, and K. Wehrle, "Security challenges in the ip-based internet of things," Wireless Personal Communications, vol. 61, September 2011, pp. 527–542, doi:10.1007/s11277-011-0385-5.

[12] C. Karlof and D. Wagner, "Secure routing in wireless sensor networks: Attacks and countermeasures," Ad hoc networks, vol. 1, September 2003, pp. 293–315.

[13] L. Wallgren, S. Raza, and T. Voigt, "Routing attacks and countermeasures in the rpl-based internet of things," International Journal of Distributed Sensor Networks, vol. 2013, June 2013.

[14] S. Raza, L. Wallgren, and T. Voigt, "Svelte: Real-time intrusion detection in the internet of things," Ad hoc networks, vol. 11, November 2013, pp. 2661–2674.

[15] D. Weyns, S. Malek, and J. Andersson, "Forms: a formal reference model for self-adaptation," in Proceedings of the 7th international conference on Autonomic computing. ACM, June 2010, pp. 205–214, doi:10.1145/1809049.1809078.

[16] J. Greensmith, U. Aickelin, and S. Cayzer, "Introducing dendritic cells as a novel immune-inspired algorithm for anomaly detection," in Artificial Immune Systems. Springer, August 2005, pp. 153–167, doi:10.1007/11536444_12.

[17] J. Bao, Y. Chen, and J. Yu, "An optimized discrete neural network in embedded systems for road recognition," Engineering Applications of Artificial Intelligence, vol. 25, June 2012, pp. 775–782.

[18] A. Carvalho, K. FACELI, A. LORENA, and J. GAMA, "Inteligência artificial–uma abordagem de aprendizado de máquina," Rio de Janeiro: LTC, 2011.

 ISBN: 978-1-61208-514-2