STUDYING MALWARE PROPAGATION IN WIRELESS SENSOR NETWORKS WITH CELL-DEVS

Ala'a Al-Habashna and Gabriel Wainer

Systems and Computer Engineering
Carleton University
Ottawa, ON, Canada
{alaaalhabashna, gwainer}@sce.carleton.ca

ABSTRACT
Wireless Sensor Networks (WSNs) are usually susceptible to viruses and malware attacks. This is due to the nature of the forming nodes which are constrained in terms of memory, processing capabilities, battery power, and communication capabilities. Such constraints impose limitations in terms of the employed security features, as sophisticated protection mechanisms would be considered inefficient and hence unsuitable. Therefore, WSNs could be attacked by viruses and malware which interrupt or completely terminate their main functionalities. We present a method to study malware propagation patterns in WSNs and examine the abilities of different Media Access Control (MAC) rules and techniques to limit malware propagation. Two different MAC protocols are considered. We have built different models for wireless communication in WSNs as well as malware propagation using Cell-DEVS. Furthermore, we have implemented our models and ran simulations to study malware propagation.

Keywords: DEVS, Cell-DEVS, Malware, Wireless Sensor Networks.

1 INTRODUCTION

With the emergence of Internet of Things (IoT) and the improvements on embedded devices, Wireless Sensor Networks (WSNs) are regarded as an important technology due to their potential usage in monitoring applications in number of fields including defense, health care, environmental science, or chemical industry. There are still technical challenges that should be overcome to achieve a widespread employment of wireless sensor networks. S*ecurity* is one of the most serious issues to be considered for the deployment and operation of WSNs.

Different research has been conducted on malware and virus attacks on wireless computer networks and wireless telecommunication networks (Karyotis and Papavassiliou 2014; Angel Martín del Rey 2013; Peng, Wang, and Yu 2013; Yu et al. 2015). However, WSNs differ from traditional computer networks and wireless telecommunication networks in various aspects. First, they are highly distributed and normally include a large number of distributed nodes (sensor nodes) with the ability to monitor their surroundings. Second, sensor nodes are limited in power, computational capacities, and memory. Finally, self-organization is a fundamental feature of wireless sensor networks, meaning that they need normal operation with minimal human interaction. Security mechanisms employed in traditional computer networks or wireless telecommunication networks are not suitable, due to the mentioned limitations of WSNs. As such, it is important to come up with methods to analyze the impact and propagation of malware attacks on WSNs and study the performance of different lightweight protection techniques. Furthermore, it is important to develop spatial models and tools that visually simulate the propagation of malware in WSNs, which can help understanding the malware propagation patterns under different operation conditions and protocols.

Cellular Automata (CA)-based models have been proposed to study malware propagation in WSNs (Yurong Song and Guo-Ping Jiang 2008). In such models, malware propagation is modeled using CA theory (Das

2012), a well-established and popular methods for natural computing. CA allows modeling complex natural systems with a large numbers of simple identical components and their local interactions. Cell-DEVS can be identified as an advanced version of CA which uses the Discrete Event System Specifications (DEVS) formalism to enhance the capabilities of conventional CA (Wainer 2009). It allows interactions with external entities and components, and explicit timing delays when computing individual cell states. Hence events that trigger local cell state computation can be fired by entities that are external to the cellular neighborhood. Moreover, advanced Cell-DEVS simulators have allowed convenient behavioral modeling using multiple state variables and multiple ports that define the cells' characteristics.

In this work, we show how the Cell-DEVS methodology can be used to build models for WSNs and malware propagation in WSNs. We have focused on the inherent characteristics of WSNs and the dynamic process of malware propagation. We have also implemented our models using CD++ (Wainer 2009), a simulation engine designed to simulate Cell-DEVS models. The models developed were used to run various simulations to study malware propagation under two Medium Access Control (MAC) protocols. Simulation results revealed that MAC mechanisms of wireless sensor networks have an impact on the speed of malware propagation and can reduce the risk of large-scale malware prevalence in WSNs. The developed WSN models can be used to accurately generate the dynamic behavior of malware propagation in WSNs, in order to understand such behavior and develop robust and efficient defense systems. We present three WSN malware-propagation simulation models. First, we a present a basic Cell-DEVS model that we implement with the CD++ toolkit. Then, we present the implementation of this model with an extended version of CD++. Finally, we updated the simulation model to resemble more realistic WSN, and compare effectiveness of different MAC protocols on malware spread controlling.

2 BACKGROUND

Malware can be defined as a piece of software with malicious intent that is designed to implant itself in a computer or server, and cause some damage (Wikipedia n.d.). It can take the form of scripts, executable code, or other kinds of software and propagates from the infected device to other devices either automatically or through user activities. Malware attacks on wireless computer networks and wireless telecommunication networks have been investigated by many researchers (Karyotis and Papavassiliou 2014; Angel Martín del Rey 2013; Peng, Wang, and Yu 2013; Yu et al. 2015). In (Karyotis and Papavassiliou 2014), the authors model malware spreading in wireless networks. The authors focus on the impact of malware on the network and its effect on node churn. A queuing-based model was adopted for malware spreading for the case of wireless distributed networks. CA-based models have been developed for malware propagation in computer networks. A CA-based model to simulate computer malware spreading was proposed in (Angel Martín del Rey 2013). The authors focus on fast spreading of computer malware that are mainly distributed through bulk e-mailing to the contacts in the address book. An efficient two-dimensional CA model for worm propagation was proposed in (Peng, Wang, and Yu 2013). The model utilizes the epidemic theory and defines a set of suitable local transition rules for the CA model based on this theory.

As mentioned in the previous section, WSNs have fundamental differences from traditional computer networks and wireless telecommunication networks which makes it necessary to study and analyze the propagation of malware in such networks. Several research works have recently given attention to study malware propagation in WSN and proposed a number of techniques to minimize its effects and limit its propagation. Some of the proposed approaches involve application of signal processing techniques to model space-time propagation dynamics of topologically-aware malware in a sensor network with uniformly distributed nodes (Khayam and Radha 2006).

In addition to the work above, CA-based models have been developed to simulate malware propagation in WSNs. In (Yurong Song and Guo-Ping Jiang 2008), a CA-based model has been developed to analyze the propagation of malware in WSNs through multi-hop broadcast protocols. The simulation results show that the density of sensor nodes has a significant impact on malware propagation over the WSN. Results have also shown that malware propagation diffuses continuously from infected nodes toward the outside nodes which are spatially bounded.

In this work, we use Cell-DEVS to develop multiple models for malware propagation in WSNs. It is well established that Cell-DEVS has many advantages over CA (Wainer 2009; Al-Habashna and Wainer 2016). Most importantly, the Cell-DEVS formalisms provide an asynchronous cellular modeling technique including a better definition of timing properties in the model and allowing combining models easily. This also allows modeling complex systems more efficiently. For further detail on the advantages of Cell-DEVS, the reader is referred to (Wainer 2009).

Cell-DEVS is a combination of CA and DEVS with explicit timing delays (Wainer 2009). DEVS provides a formal framework for modeling generic dynamic systems and includes hierarchical, modular, and component-oriented structure. Moreover, it has a formal specifications for defining the structure and behavior of a discrete event model (Zeigler, Praehofer, and Kim 2000). Cell-DEVS (Wainer 2009) extends the DEVS formalisms, allowing us to implement cellular models with explicit timing delays. Once the behavior of a cell is defined, a coupled Cell-DEVS model can be created by interconnecting a number of cells with their neighbors. Each cell is defined as a DEVS atomic model. Each cell uses N inputs to compute its next state. These inputs, which are received through the model's interface, activate a local computing function τ. A delay d can be associated with each cell. A coupled Cell-DEVS model is the resulting array of cells (atomic models) with given dimensions, borders, and zones.

A Cell-DEVS model was developed for large WSNs in (Qela, Wainer, and Mouftah 2009) and used to study the behavior of the network and evaluate topology control algorithms. In (Kazi and Wainer 2017), a simple model has been developed for malware propagation in WSNs. We will show how to build similar models using Cell-DEVS and implementing them using the CD++ toolkit. We present two different models, a simple version, and an extended one using an advanced specification language which reduces the number of cells in the model, and hence, reduces the number of exchanged messages and log-file size, which in turn reduces the execution time. Furthermore, we build an improved version of the model, which employ more complex rules that produce a more realistic behavior of WSNs.

3 MODELING MALWARE IN WSN

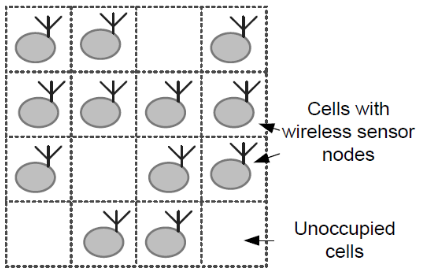
In this section, we start by discussing the model, which captures a few basic characteristics of WSN. As mentioned above, we provide two implementations of this model. The original version provides simple rules for experimentation, while the extended version, which includes a more complex model definition, allows adding more features and functionalities while improving performance.

3.1 Cell-DEVS WSN model

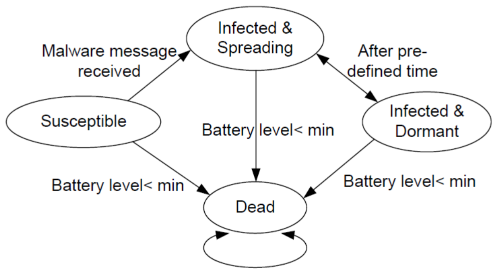
We have modeled the WSN as a 2-dimensional lattice of cells, with each cell may or may not be occupied by a stationary wireless sensor node. The states that each cell can take are shown in Table 1. Figure 1-a explains our Cell-DEVS model. The figure shows a 2-dimentional grid of cells. An empty cell represent a small area that is empty, i.e., area that does not contain a sensor node, or an area with a dead sensor node. Other cells represent areas with sensor nodes. A cell with a sensor node can be in one of three states: *Susceptible*, *Infected & spreading*, or *Infected & dormant*, as shown in Table 1.

Table 1: Malware plane states.

State	Value
Dead or unoccupied	-1
Susceptible	0
Infected & spreading	1
Infected & dormant	2



(a)

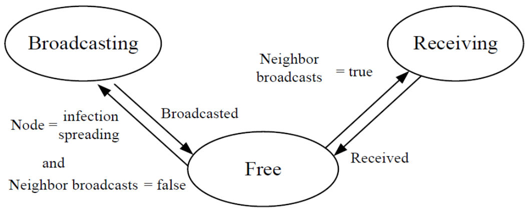


(b)

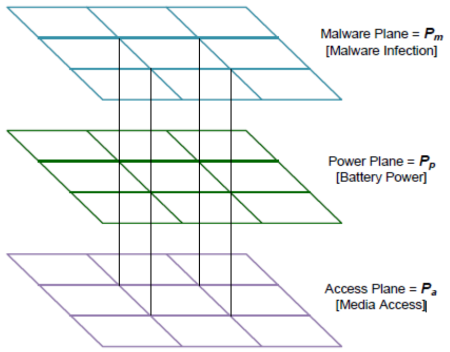
Figure 1: (a) Two-dimensional representation of the studied area. (b) State-transition diagram for malware plane (initial model).

The states of the cells change as per the state-transition diagram shown in Figure 1-b. Initially all the nodes in the lattice are in the *susceptible* state with fixed battery power. If no malware is received, it is assumed that sensor nodes perform their normal functions using relatively less power consumption. Hence, uninfected nodes will take a long time to switch to the *Dead* state. However if a node receives a malware message, it moves to the *infected-&-spreading* state and starts broadcasting malware to its neighbors. Broadcasting consumes a significant amount of power, draining the battery faster and results in moving to the *Dead* state quickly.

Wireless Media Access Control (MAC) protocols are used in WSNs to manage access of nodes to the transmission channel and minimize transmission collisions in an energy-efficient manner (Kabara and Calle 2012). Initially we considered the basic characteristics of media access protocol to guarantee channel access fairness and minimize collisions. Table 2 shows media access states of each node that are used to model wireless data transfer with minimal collisions. Transition of states and assignment of these state values are implemented as per the transition diagram in Figure 2-a. Due to the media access (MAC layer) rules, a node has to wait for a channel to be free, to start broadcasting. Basic characteristics of wireless channel access is modeled by introducing the *infected-&-dormant* state. An *Infected-&-spreading* node waits for random delay before the next malware broadcast by moving to the *infected-&-dormant* state after each broadcast. Finally sensor node battery power is modeled by starting at a battery power level value of 20 and decreasing to the minimum of 10 at different rates for each node depending on the state.


Broadcasting
Receiving
Neighbor broadcasts = true
Broadcasted
Node = infection spreading
and
Neighbor broadcasts = false
Free
Received

(a)


Malware Plane = P_m [Malware Infection]
Power Plane = P_p [Battery Power]
Access Plane = P_a [Media Access]

(b)

Figure 2: (a) State-transition diagram of the access plane. (b) Three planes used to assign three state variables.

In our initial version, we modeled the system in Cell-DEVS using three planes to represent three state variables of a sensor node at a particular time instance: a *malware* plane, a *power* plane and an *access* plane, as shown in Figure 2-b. The Malware plane is used to handle malware related values representing wireless sensor nodes. As we discussed earlier, a cell in the malware plane can be; dead/unoccupied, susceptible, spreading or dormant. The Power plane is used to keep the power state values of the sensor node. An initial power value of 20 represents a full battery level for a sensor node and a value of 10 represents a completely depleted battery of the sensor node. The access plane defines MAC-related states, and has the state variable values *channel free*, *receiving* and *broadcasting*.

3.2 Advanced WSN model

We built an improved version of the original model by including new states and transition rules to resemble a more realistic behavior of WSN, exploiting additional features introduced by an extended version of CD++ (López and Wainer 2004). First, we implemented the model using the extended version of the simulation engine. We preserved most of the model characteristics and simulation rules but implemented them with multiple state variables and ports.

After converting the model with multiple-plane design to a multiple-port design, we further enhanced the model by adding new functionalities. These new modifications can be highlighted as:

- Cells were modified to compute their states by receiving neighbor inputs from three input ports
- A probabilistic approach was followed to model shared wireless medium access
- The characteristics of the Sensor-MAC (S-MAC) (Ye, Heidemann, and Estrin 2004) protocol and the Optimized MAC protocol (Rajesh Yadav 2008) were modeled introducing fixed and variable sleep states

Using multiple state variables and ports reduced the complexity of the multiple-plane model. We used three state variables, namely, *node*, *txrx* and *pwd* to simulate node state changes, data transmission and channel access state changes, and power changes, respectively. The state variables and corresponding values are illustrated in Table 2.

Table 2: Malware plane states.

	Node							**Pwd**			**TxRx**			
Value	-1	0	1	2	3	4	5	20	…	0	0	1	2	3
Represented state	Dead or Occupied	Susceptible sleeping node	Susceptible active node	Infection spreading node	Infected but dormant node	Patched active node	Patched sleeping node	Power level			No channel access	Normal ad-hoc messages	Malware broadcast	Patch messages

Each variable name represents a state variable and its input/output ports. When a change occur to the value of a certain state variable, the new value is sent over the corresponding port to the neighboring cells that receive the sent value over their corresponding input ports. Figure 3 shows the state transition diagram for the node state variable.

The state variable pwd is used to simulate the power consumption. Each nodes battery starts at a value of 25 and decreases during the operation until it reaches 0. The power consumption per 100ms is different for each state. The values are not shown here due to lack of space.

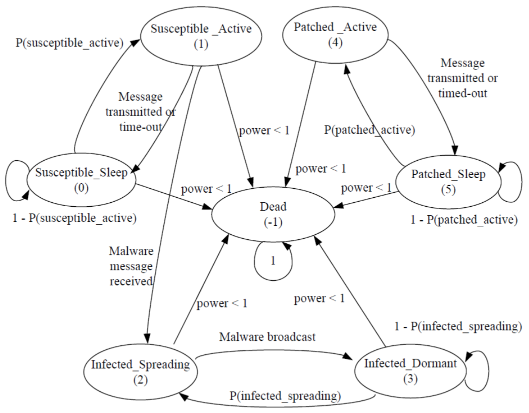


Figure 3: State-transition diagram of the improved model.

3.3 WSN MAC protocols

We implement two Wireless MAC protocols in the improved model, namely, the S-MAC protocol and the Optimized-MAC protocol. The S-MAC protocol is a contention-based MAC protocol (Ye, Heidemann, and Estrin 2004) that is a modification of the IEEE 802.11 protocol specially designed for WSN. In this MAC protocol, a sensor node periodically goes to the fixed listen/sleep cycle. A time frame in S-MAC is divided into two parts: one for a listening session and the other for a sleeping session. Only for a listen period, sensor nodes are able to communicate with other nodes and send some control packets such as SYNC, RTS (Request to Send), CTS (Clear to Send) and ACK (Acknowledgement). By a SYNC packet exchange, all neighboring nodes can synchronize together, and using RTS/CTS, two nodes can communicate with each other. Much energy is still wasted in this protocol during the listen period as the sensor will be awake even if there is no reception/transmission.

In the Optimized MAC protocol (Rajesh Yadav 2008), the sensors duty cycle is changed based on the network load. If the traffic load is high the duty cycle increases and vice versa. The network load is identified based on the number of pending messages in the queue at a particular sensor. The control packet overhead is minimized by reducing the number and size of the control packets as compared to those used in the S-MAC protocol. This protocol may be suitable for applications in which there is need for low latency.

The basic characteristics of Optimized MAC protocol and S-MAC protocols were used to model medium access in WSN. The optimized mac protocol increases the duty cycle based on the total load in the sensor network. Since we have followed a probabilistic approach, this characteristic was modeled by maintaining inversely proportional relationship between ad-hoc message generations and sleeping probabilities. On the other hand, the S-MAC protocol uses fixed duty cycles regardless of the load on the sensor network. Hence we have modeled the S-MAC characteristics by keeping fixed sleeping probability and observing the malware propagation patterns for different message-generation probabilities.

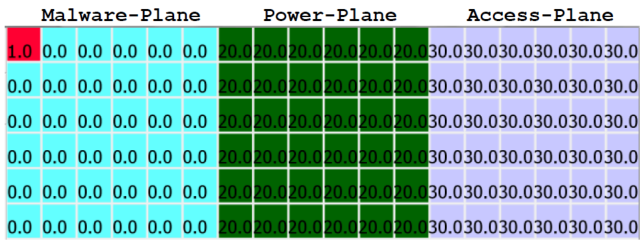
Beside these characteristics, a set of rules was defined to provide shared access to the wireless medium, minimizing collisions. Implementation of collision avoidance algorithms such as CSMA is outside the scope of this work. Hence we simply make a node wait if it detects that the channel is used by its neighbors and only transfer data if none of the neighbors use the channel (i.e. channel is free). However this simple approach does not guarantee collision free data transfer. Therefore if more than one neighbor of a particular cell access the channel at the same time, the node assumes a possible collision and simply ignores the message. Due to lack of space, we will not show the Cell-DEVS rules used to implement the models above.

4 PERFORMANCE ANALYSIS

We ran various simulations with the developed malware propagation models. In this section, we analyze the simulation results obtained from both versions of the model. Furthermore, the impact of MAC protocols on controlling malware propagation is evaluated. We consider the two MAC protocols discussed in the previous section, i.e., the SMAC protocol and the Optimized MAC protocol. which aims to optimize sensor battery usage by changing the duty cycle based on sensor load, and the S-MAC protocol that keeps a fixed duty cycle.

4.1 Output analysis

In this subsection, we study the simulation outputs captured by the basic model at different simulation times. The basic model is conceptually similar to the CA model developed in (Yurong Song and Guo-Ping Jiang 2008). As such, we analyze the results produced by this model for validation purposes. In the following subsections, we analyze the results obtained by the improved model.

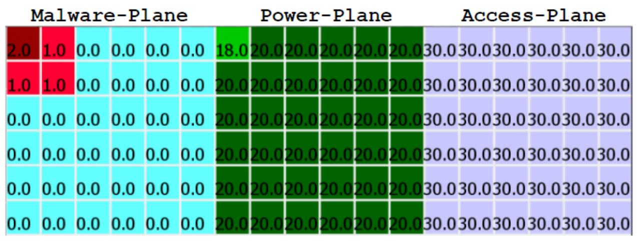


Malware-Plane						Power-Plane						Access-Plane					
1.0	0.0	0.0	0.0	0.0	0.0	20.0	20.0	20.0	20.0	20.0	20.0	30.0	30.0	30.0	30.0	30.0	30.0
0.0	0.0	0.0	0.0	0.0	0.0	20.0	20.0	20.0	20.0	20.0	20.0	30.0	30.0	30.0	30.0	30.0	30.0
0.0	0.0	0.0	0.0	0.0	0.0	20.0	20.0	20.0	20.0	20.0	20.0	30.0	30.0	30.0	30.0	30.0	30.0
0.0	0.0	0.0	0.0	0.0	0.0	20.0	20.0	20.0	20.0	20.0	20.0	30.0	30.0	30.0	30.0	30.0	30.0
0.0	0.0	0.0	0.0	0.0	0.0	20.0	20.0	20.0	20.0	20.0	20.0	30.0	30.0	30.0	30.0	30.0	30.0
0.0	0.0	0.0	0.0	0.0	0.0	20.0	20.0	20.0	20.0	20.0	20.0	30.0	30.0	30.0	30.0	30.0	30.0

Figure 4: Initial values of the cells.

Figure 4 shows the cell values at simulation time 0 ms. Hence it shows the initialization of the cells just after assigning values to cells in the malware plane, power plane and access plane. It can be seen that in the malware plane, all the nodes are in the susceptible state (value 0) except the node (0,0,0), at the top-left corner. We have initialized this node as an infected-&-spreading node (state value 1), to observe the malware propagation behavior. We have set the initial battery power level of every node to its maximum value, i.e., 20. The access plane is initialized to show that the wireless medium is free (state value 30) and none of the nodes are broadcasting (state value 31) or receiving (state value 32). According to the rules, the infected node will start broadcasting after random countdown time. As can be seen in Figure 5, after two time steps, node (0,0,0) has moved to the infected-&-dormant state while its battery power has been reduced to 18. Meanwhile, three neighbors of the infected cell has received the malware message (not shown in the figure) and moved to infection-&-spreading state.

In the next time step, three infected neighboring nodes of (0,0,0) will start broadcasting, following the medium access rules, which are defined to minimize collisions by not allowing many neighbors to broadcast simultaneously. In this version, we assume that multiple channels are available, and hence, up to a certain number of nodes can broadcast simultaneously.

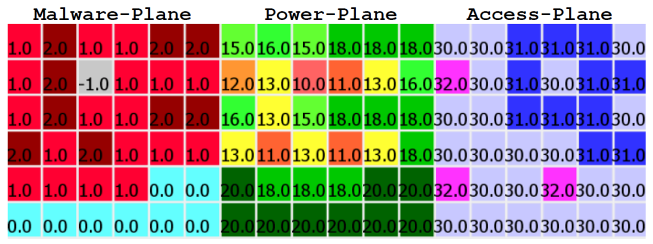


Malware-Plane						Power-Plane						Access-Plane					
2.0	1.0	0.0	0.0	0.0	0.0	18.0	20.0	20.0	20.0	20.0	20.0	30.0	30.0	30.0	30.0	30.0	30.0
1.0	1.0	0.0	0.0	0.0	0.0	20.0	20.0	20.0	20.0	20.0	20.0	30.0	30.0	30.0	30.0	30.0	30.0
0.0	0.0	0.0	0.0	0.0	0.0	20.0	20.0	20.0	20.0	20.0	20.0	30.0	30.0	30.0	30.0	30.0	30.0
0.0	0.0	0.0	0.0	0.0	0.0	20.0	20.0	20.0	20.0	20.0	20.0	30.0	30.0	30.0	30.0	30.0	30.0
0.0	0.0	0.0	0.0	0.0	0.0	20.0	20.0	20.0	20.0	20.0	20.0	30.0	30.0	30.0	30.0	30.0	30.0
0.0	0.0	0.0	0.0	0.0	0.0	20.0	20.0	20.0	20.0	20.0	20.0	30.0	30.0	30.0	30.0	30.0	30.0

Figure 5: Malware propagation to the neighbors.

Another screenshot of the simulations taken at simulation time 14000ms is shown in Figure 6. In this figure we can see the behavior of self-propagating malware. From the access plane we can see broadcast (value 32) message initiation and broadcast receiving (value 31). This gives a good example of minimal simultaneous multiple broadcasts in same neighborhood. From the power plane, we can see that malware-infected nodes consume more battery power for broadcasting and hence die faster due to faster battery drainage.

The simulation continues until the end of experimental time interval or when all sensor nodes move to the dead state. As mentioned in the beginning of this subsection, this basic model is conceptually similar to the CA model in (Yurong Song and Guo-Ping Jiang 2008), which has been already validated. As such, we compare our visual results, presented here, to the visual results obtained in (Yurong Song and Guo-Ping Jiang 2008), in order to provide operational validation of our model. By analyzing the results obtained and shown in Figure 5 and Figure 6, it can be clearly seen that malware continuously propagates from the infected node to the surrounding nodes, in a spatially bounded manner. This means that malware from an infected node does not propagate directly to susceptible nodes that are outside a bounded area. This is expected due to the nature of wireless communication in WSNs where nodes can broadcast on the channel over a spatially limited range. This behavior is different from wired networks, which can allow unbounded malware propagation. Such behavior is similar to the results obtained by the model in (Yurong Song and Guo-Ping Jiang 2008), which provides operational validation for our model.



Malware-Plane						Power-Plane						Access-Plane					
1.0	2.0	1.0	1.0	2.0	2.0	15.0	16.0	15.0	18.0	18.0	18.0	30.0	30.0	31.0	31.0	31.0	30.0
1.0	2.0	-1.0	1.0	1.0	1.0	12.0	13.0	10.0	11.0	13.0	16.0	32.0	30.0	31.0	30.0	31.0	31.0
1.0	2.0	1.0	1.0	2.0	2.0	16.0	13.0	15.0	18.0	18.0	18.0	30.0	30.0	31.0	31.0	31.0	30.0
2.0	1.0	2.0	1.0	1.0	1.0	13.0	11.0	13.0	11.0	13.0	18.0	30.0	30.0	30.0	30.0	31.0	31.0
1.0	1.0	1.0	1.0	0.0	0.0	20.0	18.0	18.0	18.0	20.0	20.0	32.0	30.0	30.0	32.0	30.0	30.0
0.0	0.0	0.0	0.0	0.0	0.0	20.0	20.0	20.0	20.0	20.0	20.0	30.0	30.0	30.0	30.0	30.0	30.0

Figure 6: Intermediary output of the simulation that shows malware propagation.

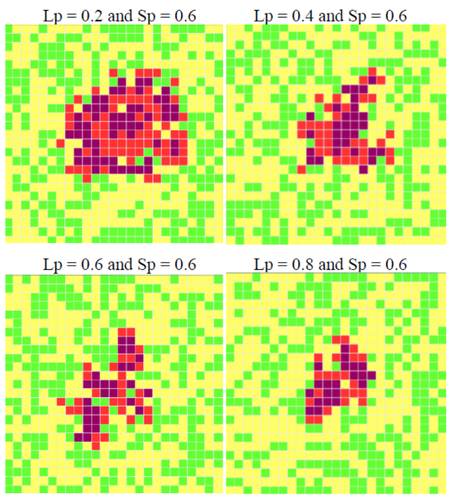
4.2 The Effect of MAC protocols on malware propagation

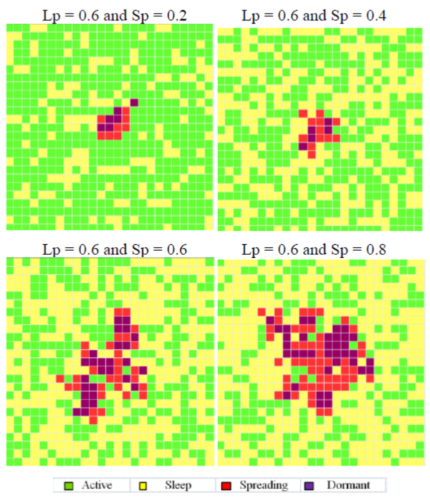
The model version introduced in Section 3.2 was also simulated using the RISE version of parallel CD++ (Al-Zoubi and Wainer 2011) using a 25×25 2-dimensional lattice. We have observed malware propagation patterns with different MAC protocols.

As we have previously discussed, the optimized MAC protocol increases the duty cycle based on the total load of the sensor network. This characteristic was modeled by maintaining inversely proportional relationship between ad-hoc message generations and sleeping probabilities. S-MAC protocol on the other hand, maintains a fixed sleep/active cycles regardless of the load. In order to compare these protocols, we have obtained the number of infected sensor nodes after 2000ms simulation time, for different combinations of ad-hoc message generation and sleep probabilities. We have made further modifications to the existing rules discussed in Section 3.2, as follows:

- We introduced the variable S_p which stand for sleeping probability. This variable control the probability at which an active node goes to sleep after time-out
- We have modified the rules by removing its post-condition, move-to-sleep, to make sure that moving to sleep is only controlled by S_p
- We introduced the variable L_p which stands for load probability. A node transmits with probability L_p each time

Figure 7-a shows the visualization of the simulation results at 2000ms, obtained by keeping S_p constant at 0.6, and changing L_p from 0.2 to 0.8 in 0.2 steps. According to the outputs shown in Figure 7-a, when node load (in terms of messages to be transmitted) is increased, malware spreading speed is decreased. This is because higher node load makes the WSN channel access highly competitive and broadcast collisions prevent faster malware spreading. Further simulations were performed where we kept the load, L_p, constant and varied the sleeping probability, S_p. The results obtained from these simulations are shown in Figure 7-b. The results show that increasing the sleep probability increases the malware-propagation speed. This is because when the nodes go to sleep for longer periods, the channel will be more available for malware broadcast which speeds up malware propagation.



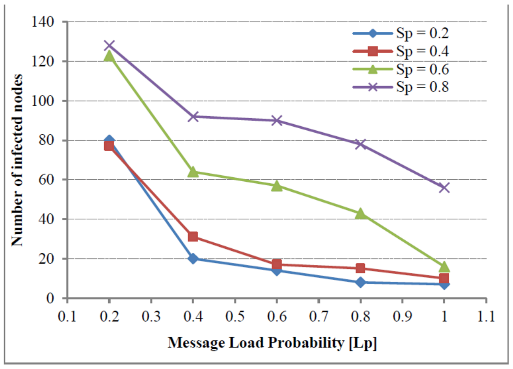


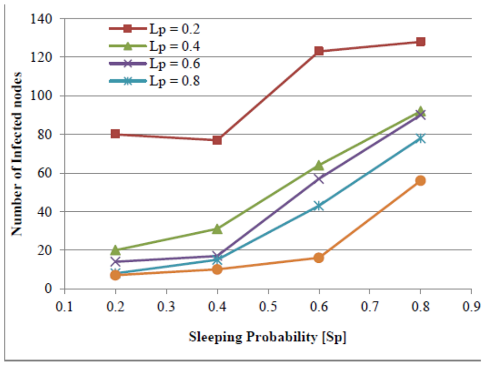
(a) (b)

Figure 7: (a) Malware spreading patterns for constant sleep and increasing load probabilities. (b) Malware spreading patterns for constant node load and different sleep probabilities.

More simulations were executed. In these simulations, the values of both L_p and S_p where varied. The number of infected nodes at simulation time of 2000ms versus L_p and S_p and are shown in Figure 8. The results in the two figures confirm that the propagation speeds decreases with increasing the load probability, and increases by increasing the sleep probability.

Given the two observations above, we can compare the effectiveness of the two considered MAC protocols, i.e., the optimized-MAC and S-MAC protocols, in controlling malware propagation speed. We have previously discussed that the optimized-MAC protocol changes the duty cycle based on sensor node load and S-MAC protocol keeps fixed duty cycles fixed regardless of the load. Hence we can build an argument that, changing the load probabilities while keeping sleep probabilities constant would resemble the operation of S-MAC protocol. Furthermore, making sleep probability inversely proportional to load probability would resembles the operation of the optimized-MAC protocol. Figure 9 shows the number of infected nodes with each MAC protocol. From the graph shown in Figure 9, we can observe that the optimized-MAC protocol is more effective than the S-MAC protocol in controlling malware propagation speeds over WSN. This is because the optimized MAC protocol changes the duty cycle based on sensor node load. As such, when the traffic load increases, the protocol decreases the sleeping cycle, which slows down malware propagation.





(a) (b)

Figure 8: (a) Number of infected nodes at 2000ms vs. sensor node communication load. (b) Number of infected nodes at 2000ms vs. sleeping probability of sensor nodes.

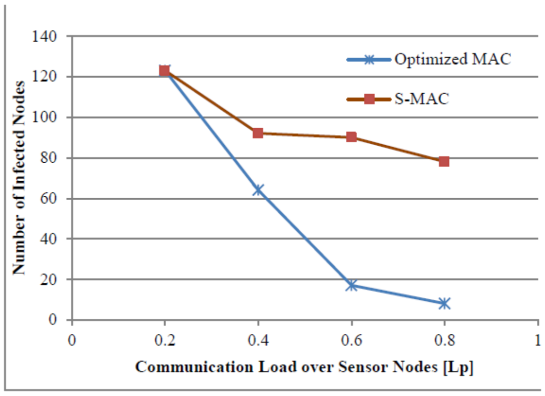


Figure 9: Malware propagation behaviors in Optimized-MAC and S-MAC protocols.

ACKNOWLEDGMENTS

The authors would like to thank Heli Amarasinghe for his valuable efforts and assistance with this research.

5 CONCLUSION

In this paper, we study malware propagation patterns in highly resource-constrained WSN environments. WSN are highly vulnerable to viruses, worms and malicious programs such as self-propagating malware. Due to limitations on processing, memory and battery power, powerful security features are not cost efficient for most WSNs. Hence, suitable Medium Access Control (MAC) protocols are widely regarded as an efficient and cost effective method of restricting malware propagation in WSNs. In this work, we have investigated the impact of two widely used MAC protocols; Optimized-MAC and S-MAC on restraining malware propagation. In future work, we will study the effect of self-propagating patching seeds on malware propagation. We also recognize the importance of carrying out a more detailed study in this area focusing on other popular WSN MAC technologies. Moreover, we are planning to integrate topological details in our future simulations and develop more powerful patching algorithms.

REFERENCES

Al-Habashna, Ala'a, and Gabriel Wainer. 2016. "Modeling Pedestrian Behavior with Cell-DEVS: Theory and Applications." *SIMULATION* 92 (2): 117–39.

Al-Zoubi, Khaldoon, and Gabriel Wainer. 2011. "Distributed Simulation Using RESTful Interoperability Simulation Environment (RISE) Middleware." Intelligence-Based Systems Engineering, 129–57. Springer, Berlin, Heidelberg.

Angel Martín del Rey, Automatá. 2013. "A SIR E-Epidemic Model for Computer Worms Based on Cellular Automata." Berlin, Heidelberg.

Das, Debasis. 2012. "A Survey on Cellular Automata and Its Applications." Communications in Computer and Information Science, 753–62. Springer, Berlin, Heidelberg.

Kabara, Joseph, and Maria Calle. 2012. "MAC Protocols Used by Wireless Sensor Networks and a General Method of Performance Evaluation." *International Journal of Distributed Sensor Networks* 8 (1): 834784.

Karyotis, Vasileios, and Symeon Papavassiliou. 2014. "Evaluation of Malware Spreading in Wireless Multihop Networks with Churn." In *International Conference on Ad Hoc Networks*, 63–74. San Remo, Italy: Springer, Cham.

Kazi, Baha Uddin, and Gabriel A. Wainer. 2017. "Formal Modeling and Simulation to Analyze the Dynamics of Malware Propagation in Networks Using Cell-DEVS." In *Proceedings of the 20th Communications & Networking Symposium*, 1–12. Virginia Beach, USA: Society for Computer Simulation International.

Khayam, S.A., and H. Radha. 2006. "Using Signal Processing Techniques to Model Worm Propagation over Wireless Sensor Networks." *IEEE Signal Processing Magazine* 23 (2): 164–69.

López, Alejandro, and Gabriel Wainer. 2004. "Improved Cell-DEVS Model Definition in CD++." In *Cellular Automata ACRI 2004. Lecture Notes in Computer Science, Vol 3305*, 803–12. Berlin, Heidelberg: Springer.

Peng, Sancheng, Guojun Wang, and Shui Yu. 2013. "Modeling the Dynamics of Worm Propagation Using Two-Dimensional Cellular Automata in Smartphones." *Journal of Computer and System Sciences* 79 (5): 586–95.

Qela, Blerim, Gabriel Wainer, and Hussein Mouftah. 2009. "Simulation of Large Wireless Sensor Networks Using Cell-DEVS." In *Proceedings of the 2009 Winter Simulation Conference (WSC)*, 3189–3200. IEEE.

Rajesh Yadav, Shirshu Varma and N.Malaviya. 2008. "Optimized Medium Access Control for Wireless Sensor Network." *International Journal of Computer Science and Network Security* 8 (2): 334–38.

Wainer, Gabriel A. 2009. *Discrete-Event Modeling and Simulation : A Practitioner's Approach*. Boca Raton: CRC Press.

Wikipedia. n.d. "Malware." Wikipedia. Accessed May 23, 2019. https://en.wikipedia.org/wiki/Malware.

Ye, W., J. Heidemann, and D. Estrin. 2004. "Medium Access Control With Coordinated Adaptive Sleeping for Wireless Sensor Networks." *IEEE/ACM Transactions on Networking* 12 (3): 493–506.

Yu, Shui, Guofei Gu, Ahmed Barnawi, Song Guo, and Ivan Stojmenovic. 2015. "Malware Propagation in Large-Scale Networks." *IEEE Transactions on Knowledge and Data Engineering* 27 (1): 170–79.

Yurong Song, and Guo-Ping Jiang. 2008. "Modeling Malware Propagation in Wireless Sensor Networks Using Cellular Automata." In *2008 International Conference on Neural Networks and Signal Processing*, 623–27. Nanjing, China: IEEE.

Zeigler, Bernard P., Herbert. Praehofer, and Tag Gon. Kim. 2000. *Theory of Modeling and Simulation: Integrating Discrete Event and Continuous Complex Dynamic Systems*. San Diego: Academic Press.

AUTHOR BIOGRAPHIES

ALA'A AL-HABASHNA has obtained his Master of Engineering degree in Electrical and Computer Engineering from Memorial University of Newfoundland, St. John's, Canada, and his PhD in Electrical and Computer Engineering from Carleton University, Ottawa, Canada. Currently, he is a Postdoctoral Fellow at the Department of Systems and Computer Engineering at Carleton University. His email address is alaaalhabashna@sce.carleton.ca.

GABRIEL WAINER is a Professor at the Department of Systems and Computer Engineering at Carleton University. He is a Fellow of SCS. His email address is gwainer@sce.carleton.ca.