Healing Wireless Sensor Networks from Malicious Epidemic Diffusion

Nicola Roberto Zema, Enrico Natalizio, Michael Poss, Giuseppe Ruggeri, Antonella Molinaro

Healing wireless sensor networks from malicious epidemic diffusion

Nicola Roberto Zema*, Enrico Natalizio†, Michael Poss†, Giuseppe Ruggeri*, Antonella Molinaro*

*ARTS Laboratory, University "Mediterranea" of Reggio Calabria - DIIES Department, Italy.
{antonella.molinaro, giuseppe.ruggeri, nicola.zema}@unirc.it

†Lab. Heudiasyc, UMR CNRS 7253, France.
{enrico.natalizio, michael.poss}@hds.utc.fr

***Abstract*—Leveraging the concept of *controlled node mobility* in this paper we develop an algorithm for tracking and controlling proximity malware propagation in Wireless Sensor Networks (WSN). Our proposal aims at: (i) notifying the nodes of malware propagation, (ii) leading a flying robot along a path in the WSN that guarantees the minimum recovery time to (iii) heal the infected nodes. We formulate the *targeted curing problem* as a binary integer problem and determine the optimal solution by a central solver. We use the analytical result as a benchmark to evaluate the recovery time of the proposed solution. The achieved results show a satisfactory performance in terms of tracking the presence of an ongoing epidemic and healing the nodes.**

***Index Terms*—Controlled mobility, Wireless Sensor Network, Malware Removal**

I. Introduction

Proximity malware propagation consists in malicious code injection and diffusion in a wireless distributed network [1], [2]. This has recently become a serious security threat also in Wireless Sensor Networks (WSNs) [3]. Traditionally, sensor nodes were conceived as resource-constrained devices whose behaviour was essentially determined by hard-wired algorithms. This made them naturally immune to malwares. Today, instead, advanced sensing applications exploit more complex sensor devices, whose capabilities are very similar to those of smartphones and microcomputers [4]. For example, a sensor network used for video-surveillance applications shares the typical constraints of a WSN (in terms of energy and deployment in inaccessible fields) but it also requires additional complexity to be capable to handle a video stream and transmit it to a remote destination. The additional complexity required from WSNs to satisfy new applications requirements translates in more complex operating systems and communication protocols, which, in turn, expose the network to security risks. In recent past, several proximity malwares have been detected that use Bluetooth (e.g., Cabir [5]) and WiFi (e.g., iKee [6]) connections to spread among mobile devices. Furthermore, the feasibility of full-scale attacks on networks that rely on short range communications, like WSNs, has been demonstrated [7]. Several defense mechanisms have been proposed to face proximity malware in WSNs [8], [9]. However, so far, the proposed solutions for curing infected nodes have relied on the network infrastructure or a human intervention to deliver the patches and securing the remote access. Unfortunately, these features are not commonly available in the majority of WNS deployments [10]–[12].

In this paper, our main contributions, listed below, aim at fulfilling this lack:

1) We propose to leverage a diffusion mechanism to inoculate a cure to infected nodes (i.e. provide a method to restore the nodes to an original *healthy* state). The diffusion of the cure is performed by a flying robot (which we will refer to as *searcher* in the following), which is capable of autonomously move within the network and deliver node maintenance.
2) We develop a counter-epidemic algorithm that (i) notifies the nodes of the malware spreading, (ii) drives the *searcher* through the network to prevent and hinder the diffusion of a malicious epidemic and (iii) heals the infected nodes.
3) We propose a mathematical model for the Targeted Curing Problem (TCP) to determine the best trajectory for the *searcher* to cure infected nodes and contain the epidemic outbreak. Specifically, the model outputs the lowest delay needed to inoculate a cure by using a counter-epidemic diffusion scheme. We use the model to benchmark the behavior of our proposed approach.

For the purpose of our scenario in Section II we introduce the background research on the subject, followed by our set of working prerequisites in Section III. Our main contribution is presented in Section IV and V, where we first describe our scenario and main proposal, and after we propose a mathematical formulation of the same problem. In Section VI we present our simulative and numerical results while the Section VII concludes our proposal.

II. Related Work

Our work is mainly related to three research topics: the epidemics diffusion and network immunization, the identification of outbreaks, and the path determination for the *searcher*'s movement. In the following subsections, we will discuss the three topics separately.

A. Epidemics diffusion and network immunization

The research on networks epidemics and immunization, with reference to WSN, is recently becoming a popular topic among the networking research community [8], [13]. The most common strategy to limit the effects of an epidemic is to render a part of population immune [14] by means of some vaccine. However, a highly viral computer epidemic needs almost 100% immunization to produce an appreciable effect, and a random immunization strategy could lead to intolerable delays [15]. In the case of a WSN, it is more effective to upload on the infected node a new non protected part of the operating system that would restore the node functionalities and cure the node [16]. We will adopt this solution as part of our curing algorithm. Generally, the abovementioned strategies are tailored for *epidemic prevention*, whereas we aim at controlling epidemic diffusion.

B. Intrusion detection systems for outbreak identification

Often, in a wireless environment, the outbreak identification is associated with the node failure identification [1], [17]. However, the presence of an unwanted epidemic could not be generally seen as a node failure. In fact, an infection does not necessarily hinder basic node functionalities [18]. Furthermore, a malware might be programmed to spread through the network without disabling nodes functionalities in order to maximize the attack impact on the network [19]. In the mentioned cases, Intrusion Detection Systems (IDS) have been proposed [17], [20]. In this paper we specifically refer to host-based Intrusion Detection Systems that, when present on mobile devices, are capable of identifying the presence of malware monitoring continuously a set of relevant parameters and applying several different information extraction techniques to the data gathered. Collecting information on runtime code, resources and network usage, IDSs are capable to detect the anomalous behaviors of nodes caused by malwares without the need of signature databases.

By following the taxonomy presented in [21], our algorithm can be classified as a part of the Service Discovery Protocol (SDP) that uses a *passive* form of search. More specifically, our algorithm can be considered from a SDP's perspective as a service in the network that keeps track of the ongoing epidemics and informs the *searcher* about the malicious nodes when queried.

C. Search and pursuit

Traditionally *search and pursuit* games have been addressed for the definition of strategies that maximize the performance of a *searcher* entity against a worst-case adversary. In these games, one or more searchers try to find and capture one or more evaders. If we consider the epidemic as an evader, we can apply search and pursuit methodologies to find the best trajectories for the *searcher* to move in order to cure and immunize the network. The existing solutions focus on the searcher's motion planning to find worst-case bounds on its performance. Following the taxonomy presented in [22] our algorithm would be deployed in a discrete, finite and scale-free graph [23]. Furthermore, our algorithm runs on a speed-constrained single *searcher*, whose sensor model uses *imperfect detection* with finite range. There are different works where this class of systems is approached [24], [25], however the particular set of conditions on which our proposal is dependent has not been treated in literature.

III. Main assumptions

In this Section we describe the main assumptions on which our work is based, and we give some details regarding the epidemic dynamics and the network model. We suppose that sensors have an operating system, computational and communication resources that make them capable to perform advanced applications [4]. Furthermore, we assume that:

- no infrastructured access to an external network is provided, and the sensors communicate only over short-range radio links;
- the proximity malware is self-contained in a software package that is injected into a set of sensors (the initial malicious spreaders) and spreads across the network by exploiting the short-range wireless communication capabilities of sensor nodes [5], [6];
- when the malware infects a node, it shuts down its communication capabilities, except those useful to keep spreading [19];
- after a certain time that a node has been in contact with an infected neighbor, it will accept to receive the software package containing the malware;
- an IDS, as those mentioned in Section II-B, is put in place in the WSN to detect the presence of the malware [1];
- when a malware outbreak is detected by the IDS, a *searcher* is sent on the infected node to install a software package containing the *cure* that can safely eradicate the infection;
- the cure is always a clean image of the operating system that is re-uploded on the infected node, regardless of the type of malware that has infected it;
- a flying robot is used as the *searcher* in order to consider any kind of WSN deployment, even in harsh and remote scenarios;
- all the messages between the infected nodes and the *searcher* are directly embedded in Layer 2 (L2) frames, because L2 primitives are usually implemented by the network interface card (NIC) hardware. Therefore, they can be considered reliable even if the functioning of the node is heavily compromised;
- it is supposed that the geographical localization of nodes is available in form of predefined coordinates or through the presence of a GPS system into them.
- a secure interface on the sensors is available to guarantee low level access to the wireless interface [26], as explained in the following.

Concerning the last assumption, the secure access could be granted by an operating system that supports protected memory for kernel operations [26]. Furthermore, several works presented the possibility of a secure boot of a node after the

retrieval of a new part of the operating system [27], [28], as also specified by the guidelines of Trusted Computing [29]. Thus, we also assume that sensors are able to replace a part or the whole operating system by downloading some software modules that, in our approach, will be transferred via wireless interface as described in Section IV.

IV. Distributed Curing Algorithm

Our idea is to begin to inoculate the cure in the region with the highest density of infected nodes. In our work the density perceived by a node is intended as the number of reachable neighbors versus the node's sensing range. If we consider that all the nodes are of the same kind, we can just use the total number of neighbors as an indication of the density of a circular area of which the node is the center. Our Distributed Curing Algorithm (DCA) can be split in two concurrent processes. The first process is the *Alerting Process*, where the nodes exchange information about the diffusion of an epidemic. This process is carried out by the infected nodes themselves [17]. The second process, carried out by the *searcher*, is the *Healing Process* which can be split in two phases that are continuously reiterated: (i) the positioning phase where the *searcher*, aware of the epidemic, moves towards it and (ii) the curing phase where the searcher node cures the infected ones. After those phases the algorithm cycles and starts again with phase (i).

A. The Alerting Process

The responsible of the alerting process at each node is the IDS (see Section III). When the IDS detects that some malware is compromising the functioning of the node, it starts to send an *SOS* message every Δt_{SOS} intervals by using the node's short range interface. As explained in Section III, *SOS* messages are directly embedded into L2 frames.

Each *SOS* message contains:

- **Sender id (*SiD*)**: A 32 bit label which uniquely identifies the node that generated the message.
- **Sender position (*SP*):** The GPS position of the node which generates the *message*. Since in our scenario the nodes are stationary this information can be stored in the secure memory at time of node installation. The GPS position is coded through a 64 bit long field.
- **The number of known infected nodes (*nKIN*):** This is the number of infected nodes in the sensing range of the sending node. A sending node can infer this number by hearing to *SOS* message coming from other nodes in a reference time interval $\Delta t_{observ} = 2\Delta t_{SOS}$ If no other *SOS* message are received then this field is set to one.

Every node, both infected and not, participates in re-broadcasting the *SOS* messages. A node which receives a *SOS* message waits for Δt_{observ} seconds before to rebroadcast it. If more than one *SOS* messages are received only the one with the highest *nKIN* is retransmitted. Obviously, local *nKIN* is considered also in the ranking.

B. The Healing Process

During the *Positioning phase*, phase (i), a *searcher* is constantly patrolling the area where the nodes are deployed; it constantly monitors its short range interface looking for *SOS* message. If one of such messages is received the *searcher* moves toward the point indicated in *SP*. If more than one *SOS* message are received the *searcher* moves toward the point indicated in *SP* of the message with the highest *nKIN*. For further reference, we name the movement scheme as *CIPS*, (Critical Infection Point Seeker).

During the *Curing phase*, phase (ii), the cure is actually inoculated to infected nodes. The inoculation consists in an interaction between the *searcher* and the IDS of an infected node. The first transmits a clean image of the operating system to the nodes. This copy is used by the IDS to overwrite the infected one. When the replacement is completed the infected node is rebooted as a healed node. All the communication between the *searcher* and the infected nodes are carried out through L2 messages for the reasons outlined in Section III. Each message from the *searcher* to the nodes is signed so that it can be recognized as genuine by the secure routines of the infected nodes [26]. When the *searcher* arrives at the target starts to advertise itself to the infected nodes by issuing a *"Hello"* message every Δt_{Hello} seconds. The *"Hello"* message contains an identifier of the *searcher* and the security signature. These informations are extracted and stored by the receiving nodes.

The image of the operating system stored by the *searcher* is fragmented in chunks, we assume that each chunk has a size that allows transmitting it in a single L2 frame. This size strongly depends on which L2 technology is used. Chunks are cached by the nodes so they can be shared between many infected nodes that are downloading the image. The download of the entire code is performed through the broadcasting of two message types, the *Interest* and the *Data*. An *Interest* is used to request a specific chunk. Once an infected node receives a *"Hello"* message from a *searcher* it starts the downloading procedure which has the form of an "Interest/Data" messages exchange. When the *searcher* (or any infected node) receives an *Interest*, it broadcasts the correspondent chunk within a *Data* message. The *Interest-Data* exchange continues until the entire image is downloaded. In case of a message loss, the *Interest* is transmitted again. In order to reduce the collision probability arising from multiple simultaneous transmissions (e.g., in the presence of two or more infected nodes that can reply to the same Interest), each node I waits a random defer time T_D before transmitting the Data. During T_D, I monitors the channel to detect the same transmission performed by other nodes. Finally, if no node has issued the Data, I sends the packet.

When the download is complete the *searcher* returns in phase (i) and starts again to search for *SOS* messages.

V. Mathematical formulation

The Targeted Curing Problem (TCP) consists in determining the positions that a *searcher* should successively cover to

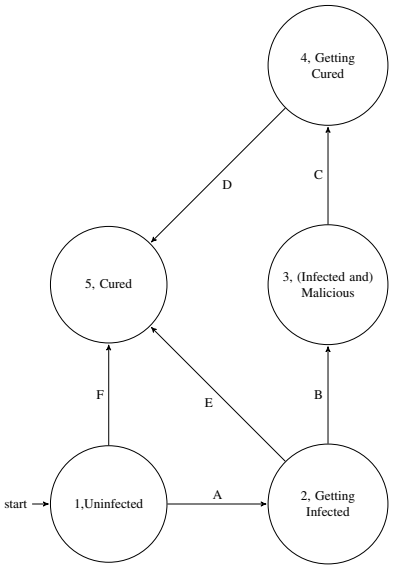


Fig. 1: Nodes states

timely cure the infected nodes and limit the diffusion of a spreading malware in the context of a WSN.

We propose a mathematical formulation of the TCP that is capable of providing an optimal solution. However, the mathematical formulation is not directly applicable to real cases, because it requires the complete *a-priori* knowledge of the network and nodes' characteristics, as well as the complete description of the diffusion dynamics. Hence, in Section VI, we will use the results of the mathematical model as a benchmark for our distributed algorithm.

We provide a mathematical programming formulation for the problem, which involves only linear constraints and binary variables. First, we characterize the states and the transitions among them. Then, we will define the parameters used in the model, the optimization variables and their role. Finally, we will detail the objective function and the model constraints.

States and transitions

In order to characterize the different states in which each node can be, we use the diagram of Fig. 1. In a given time, a node can be in one of the following states:

1) Uninfected
2) Getting Infected
3) (Infected and) Malicious
4) Getting Cured
5) Cured

In Table I we describe the transitions between each couple of states as in Fig. 1. The time intervals used in the Table will be explained in the next subsection.

Parameters

In the following list, we introduce all the parameters used in the mathematical model.

TABLE I: State transitions

Transition	States	Happens when
A	1⇒2	a node in state 1 will switch to state 2 if it has been connected for T_a time units to any number of nodes in state 3
B	2⇒3	a node in state 2 will switch to state 3 if any number of nodes in state 3 have been connected to it for, at least, T_t time units
C	3⇒4	a node in state 3 will switch to state 4 if it has been connected for T_a time units to the *searcher*
D	4⇒5	a node in state 4 will switch to state 5 if any number of nodes in state 1 or 2 or the *searcher* are connected to it for, at least, T_t time units
E	2⇒5	a node in state 2 will switch to state 5 if it has been connected for T_a time units to the *searcher*
F	1⇒5	a node in state 1 will switch to state 5 if it has been connected for T_a time units to the *searcher*

Data

T	set of time units
T^*	set of even time units
T_m	time interval for the *searcher* to choose a new direction
T_t	time needed for a node to download a curing or malicious module
T_a	time needed for the *searcher* to access the secured part of a node
N	node set
S	possible states for each node
E	wireless links, bidirected
E_c	physical path, bidirected
d_e	length of edge e in time units
d_i	transition duration of node i in time units
d_{ij}	time needed by the *searcher* to travel from i to j

Initial conditions

N^*	subset of nodes where the infection starts
t^*	time unit at which the cure is inserted
i^*	node at which the cure is inserted

The parameter T_m (time to move) is the smallest time unit considered in this work. Hence, we consider it as the unitary time unit for the model and the distributed algorithm. The parameter T_t (time to transfer) can vary within the range between 3 and 5 T_m, by following a uniform distribution among the nodes. The parameter T_a (time to access) is the time needed to the *searcher* to access the secured part of the compromised node, after the *searcher* verified that the security key has been successfully received, its value is set to 2 T_m. We tested the system and made a preliminary time parameters tuning by using the ns-2 simulator.

As we devised the use of a flying robot as *searcher* node, there is a physical path between any pair of nodes with the time needed for the *searcher* to travel this distance being d_{ij}. The wireless links used for the model are taken from an adjacency matrix that represents the topologies used in the simulative approach.

Variables

We classify the optimization variables in two groups. The first group contains the variables of the problem, which describe the current state of each node of the network at each time period and the movement of the cure, while the second group contains conditional variables that are used to simplify the model.

State and decision variables

- x_{si}^t equal to 1 iif node i is in state $s \in \{1, 3, 5\}$ at the beginning of period t
- x_{si}^{pt} equal to 1 iif node i is in state $s \in \{2, 4\}$ for p time units at the beginning of period t
- y_i^t equal to 1 iif node i carries the cure at the beginning of period t
- z_{ij}^t equal to 1 iif the cure is starting to travel from i to j at the beginning of period t

Conditional variables

- Y_i^t equal to 1 iif node i is connected to the cure
- M_i^t equal to 1 iif node i is connected to at least a node in state 3 and not to the cure
- C_i^t equal to 1 iif node i is connected to at least a node in state $1, 2$ or to the cure
- S_i^t equal to 1 iif node i is not connected to nodes in states 3 and not connected to the cure
- X^t equal to 1 iif at least one node is infecting or malicious at the beginning of period t

Objective function and constraints

The objective function aims at minimizing the time period at which the infection is extinguished (no malicious or infected nodes left):

$$\min \quad \sum_{t \in T} X^t.$$

The constraints of the problem can be classified in different groups. The constraints in Equation (1), ensure that conditional variables take the correct values. while the constraints in Equation (2) translate mathematically the state transitions introduced in Fig. 1. These constraints are non-linear. However, because all involved variables are binary, we can use a well-known technique to linearize the products. Namely, each product of binary variables $var_1\, var_2$ in the constraints above can be replaced by a new real variable $prod$ that satisfies the following constraints:

$$\begin{aligned} prod &\leq var_1 \\ prod &\leq var_2 \\ prod &\geq var_1 + var_2 - 1. \end{aligned}$$

$$\begin{aligned}
Y_i^t &\geq y_i^t && i \in N, t \in T \\
Y_i^t &\geq y_j^t && ij \in E, t \in T \\
Y_i^t &\leq y_i^t + \sum_{ij \in E} y_j^t && i \in N, t \in T \\
M_i^t &\geq x_{3j}^t - y_i^t - \sum_{ij' \in E} y_{j'}^t && ij \in E, t \in T \\
M_i^t &\leq \sum_{ij \in E} x_{3j}^t && t \in T \\
M_i^t &\leq 1 - y_i^t - \sum_{ij \in E} y_j^t && t \in T \\
C_i^t &\geq x_{1j}^t && ij \in E, t \in T \\
C_i^t &\geq x_{2j}^{pt} && ij \in E, t \in T, p = 0, \ldots, d_j - 1 \\
C_i^t &\geq x_{5j}^t && ij \in E, t \in T \\
C_i^t &\geq y_i^t && i \in N, t \in T \\
C_i^t &\geq y_j^t && ij \in E, t \in T \\
C_i^t &\leq y_i^t + \sum_{ij \in E} (y_j^t + x_{1j}^t + x_{5j}^t + \sum_{p=0}^{d_i - 1} x_{2j}^{pt}) && t \in T \\
S_i^t &\leq 1 - y_i^t && i \in N, t \in T \\
S_i^t &\leq 1 - y_j^t && ij \in E, t \in T \\
S_i^t &\leq 1 - x_{3j}^t && ij \in E, t \in T \\
S_i^t &\geq 1 - \sum_{ij \in E} (y_j^t + x_{3j}^t) - y_i^t && t \in T \\
X^t &\geq \sum_{i \in N} \frac{x_{2i}^t + x_{3i}^t}{2|N|} && t \in T
\end{aligned} \tag{1}$$

$$\begin{aligned}
x_{1i}^t &= x_{1i}^{t-1} S_i^{t-1} && i \in N, t \in T^* \\
x_{1i}^t &= x_{1i}^{t-1} && i \in N, t \in T \backslash T^* \\
x_{2i}^{0t} &= x_{2i}^{0t-1} S_i^{t-1} && i \in N, t \in T \backslash T^* \\
x_{2i}^{0t} &= x_{2i}^{0t-1} S_i^{t-1} + x_{1i}^{t-1} M_i^{t-1} && i \in N, t \in T^* \\
x_{2i}^{pt} &= x_{2i}^{pt-1} S_i^{t-1} + x_{2i}^{p-1t-1} M_i^{t-1} && p = 1, \ldots, d_i - 1, i \in N, t \in T \\
x_{3i}^t &= x_{3i}^{t-1} + x_{2i}^{d_i - 1t-1} M_i^{t-1} && i \in N, t \in T \backslash T^* \\
x_{3i}^t &= x_{3i}^{t-1} (1 - Y_i^{t-1}) + x_{2i}^{d_i - 1t-1} M_i^{t-1} && i \in N, t \in T^* \\
x_{4i}^{0t} &= x_{4i}^{0t-1} (1 - Y_i^{t-1}) && i \in N, t \in T \backslash T^* \\
x_{4i}^{0t} &= x_{4i}^{0t-1} (1 - Y_i^{t-1}) + x_{3i}^{t-1} Y_i^{t-1} && i \in N, t \in T^* \\
x_{4i}^{pt} &= x_{4i}^{pt-1} (1 - C_i^{t-1}) + x_{4i}^{p-1t-1} C_i^{t-1} && p = 1, \ldots, d_i - 1, i \in N, t \in T \\
x_{5i}^t &= x_{5i}^{t-1} + x_{4i}^{d_i - 1t-1} C_i^{t-1} && i \in N, t \in T \backslash T^* \\
x_{5i}^t &= x_{5i}^{t-1} + (x_{1i}^{t-1} + \sum_{i=0}^{d_i - 1} x_{2i}^{pt-1}) Y_i^{t-1} + x_{4i}^{d_i - 1t-1} C_i^{t-1} && i \in N, t \in T^*
\end{aligned} \tag{2}$$

For simplicity, we avoid rewriting the linearized constraints. We describe next the cure movement.

$$
\begin{aligned}
y_i^t &= 0 && i \in N, t \leq t^* - 1 \\
y_{i^*}^{t^*} &= 1 && \\
y_i^{t^*} &= 0 && i \in N \backslash i^* \\
y_i^t &= \sum_{ij \in E_c} z_{ji}^{t-d_{ij}} + y_i^{t-1} - \sum_{ij \in E_c} z_{ij}^{t-1} && \\
& && t \in T \\
\sum_{ij \in E_c} z_{ij}^t &\leq y_i^t ij \in E_c, t \in T &&
\end{aligned}
$$

Finally, the following constraints characterize the initial conditions of the system, and enforce that all optimization variables be $\{0,1\}$-valued.

$$
\begin{aligned}
x_{1i}^0 &= 1 && i \in N \backslash N^* \\
x_{3i}^0 &= 1 && i \in N^* \\
x_{1i}^0 + x_{3i}^0 + x_{5i}^0 + \sum_{p=0}^{d_i - 1} (x_{2i}^{p0} + x_{4i}^{p0}) &= 1 && \\
& && i \in N \\
Y, M, C, S, X, x, y, z &\in \{0,1\} &&
\end{aligned}
$$

VI. Performance Evaluation

In this section we compare the movement scheme computed in our algorithm, *CIPS*, to two other schemes. All three schemes present a common curing phase, i.e. the one described in section IV, but differs on the movement path of the *searcher* through the WSN.

The first alternative is to let the *searcher* move according to Random Way Point model (RWP). This approach represents a simple yet effective approach, where no effort is made to optimize the movements. In the following we will refer to it as the *RWP* approach. The second alternative consists in letting the *searcher* move according to the pattern determined by the analytical model defined in Section V. The latter requires the complete *a-priori* knowledge of the network and nodes' characteristics as well as the complete description of the diffusion dynamics. Nonetheless, it allows us to have an upper bound on the performance of our proposal. In the following we will refer to it as the *MM* (Mathematical Model) approach. We have implemented RWP and CIPS in the ns-2 [30] simulator, while the *MM* has been coded in the FICO Xpress Suite and Mosel language [31].

The reference simulated scenario is 160 m x 160 m square area. Inside this area a variable number of nodes is placed according to a regular grid pattern. The number of nodes in the grid can vary as it follows $[25, 49, 64, 81, 100]$. Hence, the distances between the nodes varies accordingly, specifically they are respectively $[32, 23, 20, 18, 16]$ m.

All the nodes are equipped with a single IEEE 802.11 interface and a Ricean fading model takes into account the multi-path effects due to various obstacles. As shown in Table II, Physical and MAC parameters are based on IEEE 802.11g and transmission power and receiver sensitivity figures are taken from data-sheets of devices available on the market [32]. It is worth to note that the maximum distance considered in our grid (32 m) is equal to the maximum achievable distance to have a reliable transmission in Table II.

TABLE II: 802.11g Simulation Parameters.

PHY Parameter	Value		MAC Parameter	Value
Frequency	2.4 GHz		SlotTime	9 μs
Receive Sensitivity	-86 dBm		SIFS	16 μs
Transmission Power	18 dBm		Preamble Length	96 bit
Max distance	32 m			

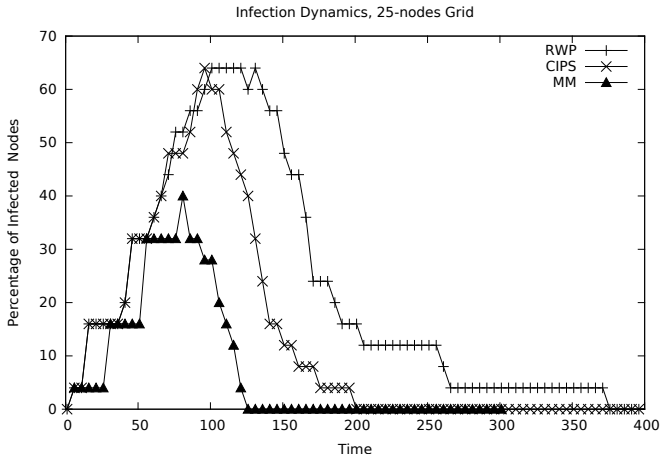


Fig. 2: Percentage of infected elements for the 25-nodes scenario.

The searcher and the first set of infected node are initially placed at random positions within the grid using a uniform distribution. Once the simulation starts, the *searcher* moves according to the mobility pattern computed by the RWP, CIPS or MM approach. The *searcher* speed is set according to the literature in this field [33].

A. Result with 25 nodes

The first simulation set was carried out considering a network of only 25 sensors. This reduced number of sensors was chosen due to the complexity of the mathematical model. Although this number is quite small the results obtained give a clear idea of how CIPS relates with the other two. In the next section we will evaluate how the CIPS and the RWP schemes scale with the number of nodes.

Our main objective is to evaluate the evolution and the steady state number of infect nodes to understand if the system is capable to recover a network from an expanding infection.

In Fig. 2 the percentage of infected nodes versus time is shown.

The MM approach is the best in preventing the epidemic diffusion and is the quickest in extinguish the infection. The RWP scheme presents the lowest performance in terms of both diffusion of the infection and recovery time. The CIPS scheme suffers a high percentage of infected nodes at the infection peak (i.e. when the malware affects the maximum number of nodes). This number is close to the one obtained with RWP. This is due to the high latency associate with the *Alerting Process*. However, in spite of this initial latency, CIPS quickly

TABLE III: 25-nodes Simulation Performances Comparison.

Scheme	Distance [m]	Peak [%]	Peak Time [s]	TTS [s]
RWP	3510	64	101	261
CIPS	959	64	96	161
MM	376	40	81	121

heals the network. The recovery time of CIPS is about $70s$ greater than the time needed by the MM and about $180s$ lower than the time needed by RWP. Consequently, the time spent by CIPS to gather information from the network is well repaid by the better functioning of the movement scheme.

Tables III reports the following metrics: (i) the distance traveled by the *searcher* until 90% of nodes are healed, (ii) the peak infection, i.e. the maximum number of nodes which are infected at the same time, (iii) the time at which the peak is reached, and (iv) the time to settle (TTS), i.e. the time needed to heal 90% of the nodes.

Table III confirms the trend formerly described. The MM approach is successful in both limiting the malware diffusion and quickly healing the network, while the CIPS approach suffers from the finite amount of time required to get the information about the network. Nonetheless, the CIPS approach far outperforms the RWP approach.

The distance traveled by the *searcher* is a metric of particular interest. A shorter traveled distance implies a lower amount of energy consumed by the *searcher*. The CIPS approach requires the *searcher* to travel about 0.27 times the distance required by the RWP approach and about 2.55 times the distance required by the MM approach.

The infection peak gives a measure of how the malware is able to make the network inoperative. A high infection peak means that only few nodes continue to properly function. Unfortunately the CIPS approach has the same peak as the RWP approach. However, the CIPS approach presents a TTS much shorter than the RWP approach and very close to that of the MM approach. This means that, due to the accurate movement pattern computed by CIPS, the aimed cure inoculation is effective for a faster network recovery.

B. Result varying the number of nodes

The high computational load and complexity of the mathematical approach did not allow us to determine a solution for grids containing more than 25 nodes. For this reason, for a higher number of nodes we used only the CIPS and the RWP schemes for comparison.

In Table IV, if we consider the distance traveled by the searcher, it can be observed a different trend between the two compared approaches. The traveled distance roughly increases with the number of nodes when the CIPS approach is considered, whereas it is roughly decreasing when the RWP scheme is considered. This is due to the fact that the CIPS approach leads to a movement scheme that is closely related to the network topology. When the number of nodes increases the *searcher* is forced to prolong its journey to cover the higher number of infected nodes in remote areas. Vice versa, for the RWP movement scheme, when the number of nodes increases, the probability that such an erratic movement produces a positive effect is raised as it travels less distance for curing the same percentage of nodes.

In Fig. 3, we present the percentages of infect nodes versus time for greater number of nodes, specifically for 49 (3a), 64 (3b) and 100 (3c) nodes. These results confirm the trend already found previously. The CIPS approach has a lower peak value and a faster TTS in respect to the RWP approach. We can also notice that a more congested topology affects the performance of the approach. For 25 nodes the infection peak for CIPS and RWP is the same. However, as the number of nodes increases, the CIPS approach obtains a lower infection peak. This is due to the fact that the CIPS approach achieves a quickly delivery of the cure, where it is most needed. The effectiveness of the proposed approach is also confirmed by the shorter TTS.

VII. Conclusions and Future Works

In this paper we addressed the issue of facing a malware spreading in a Wireless Sensor Network. Using the concepts of controlled mobility and information diffusion we defined an algorithm (i) to notify the spreading of the malware, (ii) to lead an autonomous flying robot, which can cure infected nodes, along a path in the sensor network and (iii) to heal the infected nodes. To benchmark the proposed algorithm we developed a mathematical model which defines the best path to be followed by the flying robot to cure the infected nodes in the shortest time. Using the results obtained from the mathematical model as the upper bound and the ones from the application of random waypoint movement as the lower, we have assessed the satisfactory performance of the proposed algorithm. A larger and more comprehensive mathematical model is expected as the most immediate step in order to provide more generalized solutions which can be applied in real scenarios followed by a tuning study of the proposed autonomic approach.

Acknowledgment

This work has been partially supported by: a)the STEM-Net, PRIN-National Italian Project # H21J11000050001, financed by the Italian Ministry of University and Research, and b) by the Labex MS2T,which is funded by the French Government, through the program "Investments for the future", managed by the National Agency for Research (Reference ANR-11-IDEX-0004-02). The research of Nicola Roberto Zema is partially supported by European Union (EU), European Social Fund (ESF) and Calabria Local Goverment.

References

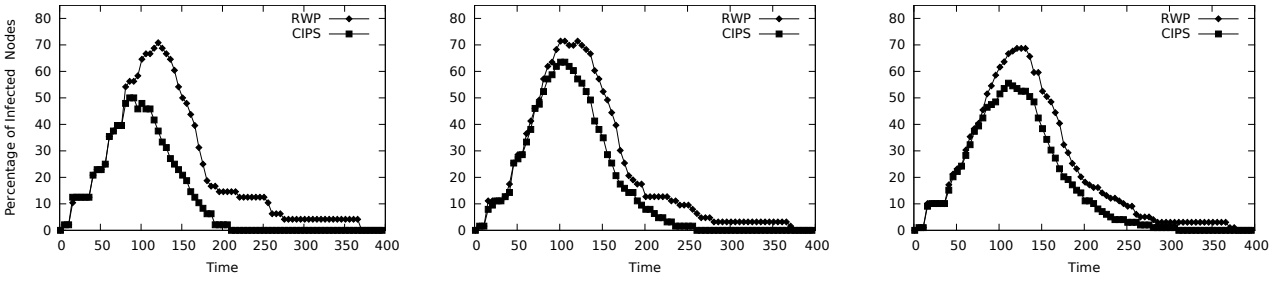
[1] M. La Polla, F. Martinelli, and D. Sgandurra, "A survey on security for mobile devices," *Communications Surveys & Tutorials*, IEEE , vol.15, no.1, pp.446,471, First Quarter 2013.

[2] Y. Wang, S. Wen, Y. Xiang, and W. Zhou, "Modeling the propagation of worms in networks: A survey," *Communications Surveys & Tutorials*, IEEE , vol.PP, no.99, pp.1,19

[3] B. Sun, G. Yan, and Y. Xiao, "Worm propagation dynamics in wireless sensor networks," in *Communications, 2008. ICC'08. IEEE International Conference on*. IEEE, 2008, pp. 1541–1545.

TABLE IV: Performance when the number of nodes varies

Nodes	*CIPS*				*RWP*			
	Distance [m]	**Peak [% of nodes]**	**Peak Time [s]**	**TTS [s]**	**Distance [m]**	**Peak [% of nodes]**	**Peak Time [s]**	**TTS [s]**
25	959	64	96	161	3510	64	101	201
49	1059	50	86	176	3180	70	121	261
64	1215	63.4	101	196	2470	71	101	241
81	1110	68.7	111	201	2894	73	116	261
100	1307	55.5	111	216	2712	68	121	251



(a) 49 Nodes (b) 64 Nodes (c) 100 Nodes

Fig. 3: Percentage of infected elements for the 49, 64 and 100-nodes scenario.

[4] I. F. Akyildiz, T. Melodia, and K. R. Chowdury, "Wireless multimedia sensor networks: A survey," *Wireless Communications, IEEE*, vol. 14, no. 6, pp. 32–39, 2007.

[5] T. M. I. (2004). Symbos cabir.a. Available: http://goo.gl/aHcES

[6] T. M. I. (2006). Ios ikee.a. Available: http://goo.gl/z0j56

[7] W. Shengjun and C. Junhua, "Modeling the spread of worm epidemics in wireless sensor networks," in *Wireless Communications, Networking and Mobile Computing, 2009. WiCom'09. 5th International Conference on*. IEEE, 2009, pp. 1–4.

[8] X. Wang, Z. He, X. Zhao, C. Lin, Y. Pan, and Z. Cai, "Reaction-diffusion modeling of malware propagation in mobile wireless sensor networks," *Science China Information Sciences*, vol. 56, no. 9, pp. 1–18, 2013.

[9] B. Sun, G. Yan, Y. Xiao, and T. Andrew Yang, "Self-propagating malpackets in wireless sensor networks: Dynamics and defense implications," *Ad Hoc Networks*, vol. 7, no. 8, pp. 1489–1500, 2009.

[10] F. Li, Y. Yang, and J. Wu, "Cpmc: An efficient proximity malware coping scheme in smartphone-based mobile networks," in *INFOCOM, 2010 Proceedings IEEE*. IEEE, 2010, pp. 1–9.

[11] A. Houmansadr, S. A. Zonouz, and R. Berthier, "A cloud-based intrusion detection and response system for mobile phones," in *Dependable Systems and Networks Workshops (DSN-W), 2011 IEEE/IFIP 41st International Conference on*. IEEE, 2011, pp. 31–32.

[12] Y. Nadji, J. Giffin, and P. Traynor, "Automated remote repair for mobile malware," in *Proceedings of the 27th Annual Computer Security Applications Conference*. ACM, 2011, pp. 413–422.

[13] S. Tang, "A modified epidemic model for virus spread control in wireless sensor networks," in *Global Telecommunications Conference (GLOBECOM 2011), 2011 IEEE*. IEEE, 2011, pp. 1–5.

[14] L. Sander, C. Warren, I. Sokolov, C. Simon, and J. Koopman, "Percolation on heterogeneous networks as a model for epidemics," *Mathematical Biosciences*, vol. 180, pp. 293–305, 2002.

[15] V. M. EguíLuz and K. Klemm, "Epidemic Threshold in Structured Scale-Free Networks," *Physical Review Letters*, vol. 89, no. 10, p. 108701, Aug. 2002.

[16] S. Brown and C. J. Sreenan, "Software updating in wireless sensor networks: A survey and lacunae," *Journal of Sensor and Actuator Networks*, vol. 2, no. 4, pp. 717–760, 2013.

[17] A. Shabtai, U. Kanonov, and Y. Elovici, "Intrusion detection for mobile devices using the knowledge-based, temporal abstraction method," *J. Syst. Softw.*, vol. 83, no. 8, pp. 1524–1537, Aug. 2010.

[18] P. Wang, M. C. González, C. A. Hidalgo, and A.-L. Barabási, "Understanding the spreading patterns of mobile phone viruses," *Science*, vol. 324, no. 5930, pp. 1071–1076, 2009.

[19] M. Khouzani, S. Sarkar, and E. Altman, "Maximum damage malware attack in mobile wireless networks," *IEEE/ACM Transactions on Networking (TON)*, vol. 20, no. 5, pp. 1347–1360, 2012.

[20] C. Gao and J. Liu, "Modeling and restraining mobile virus propagation," *IEEE Transactions on Mobile Computing* , vol.12, no.3, pp.529,541, March 2013.

[21] A. N. Mian, R. Baldoni, and R. Beraldi, "A survey of service discovery protocols in multihop mobile ad hoc networks," *Pervasive Computing, IEEE*, vol. 8, no. 1, pp. 66–74, 2009.

[22] T. H. Chung, G. A. Hollinger, and V. Isler, "Search and pursuit-evasion in mobile robotics," *Autonomous Robots*, vol. 31, no. 4, pp. 299–316, 2011.

[23] R. Pastor-Satorras and A. Vespignani, "Epidemic Spreading in Scale-Free Networks," *Phys. Rev. Lett.*, vol. 86, pp. 3200–3203, Apr 2001.

[24] M. A. Vieira, R. Govindan, and G. S. Sukhatme, "Scalable and practical pursuit-evasion with networked robots," *Intelligent Service Robotics*, vol. 2, no. 4, pp. 247–263, 2009.

[25] S. LaValle, D. Lin, L. Guibas, J.-C. Latombe, and R. Motwani, "Finding an unpredictable target in a workspace with obstacles," in *IEEE International Conference on Robotics and Automation, 1997. Proceedings, 1997* , vol. 1, 1997, pp. 737–742 vol.1.

[26] A. U. Schmidt, N. Kuntze, and M. Kasper, "On the deployment of mobile trusted modules," in *Wireless Communications and Networking Conference, 2008. WCNC 2008. IEEE*. IEEE, 2008, pp. 3169–3174.

[27] J. Grossschadl, T. Vejda, and D. Page, "Reassessing the tcg specifications for trusted computing in mobile and embedded systems," in *IEEE International Workshop on Hardware-Oriented Security and Trust, 2008. HOST 2008.* . IEEE, 2008, pp. 84–90.

[28] O. Aciicmez, A. Latifi, J.-P. Seifert, and X. Zhang, "A trusted mobile phone prototype," in *5th IEEE Consumer Communications and Networking Conference, 2008. CCNC 2008.* IEEE, 2008, pp. 1208–1209.

[29] TCG. (2011) Tpm main part 1 design principles. Specification Version 1.2 Revision 116.

[30] "Network Simulator- ns (version 2)," available from http://www.isi.edu/nsnam/ns/.

[31] "Fico xpress optimization suite," http://www.fico.com/en/products/fico-xpress-optimization-suite/.

[32] "Cisco Aironet 802.11a/b/g Wireless CardBus Adapter," Data Sheet available on line at. http://www.cisco.com/en/US/prod/collateral/wireless/ps6442/ps4555/ps5818/product_data_sheet09186a00801ebc29.pdf.

[33] E. Natalizio, R. Surace, V. Loscrí, F. Guerriero, and T. Melodia, "Two families of algorithms to film sport events with flying robots," in *The 10th IEEE International Conference on Mobile Ad-hoc and Sensor Systemsoc Networks and Wireless (MASS)*, Hangzhou, China, 2013, pp. 319–323.