EpiNet: A Simulation Framework to Study the Spread of Malware in Wireless Networks *

Karthik Channakeshava ⋆ Deepti Chafekar † Keith Bisset ‡

V.S. Anil Kumar ‡ Madhav Marathe ‡

⋆Bradley Department of Electrical and Computer Engineering
†Department of Computer Science
‡Department of Computer Science and Virginia Bioinformatics Institute
Virginia Tech, Blacksburg VA 24061
Email: {kchannak, chafekar, kbisset, akumar, mmarathe}@vbi.vt.edu

ABSTRACT

We describe a modeling framework to study the spread of malware over realistic wireless networks. We develop (i) methods for generating synthetic, yet realistic wireless networks using activity-based models of urban population mobility, and (ii) an interaction-based simulation framework to study the dynamics of worm propagation over wireless networks. We use the prototype framework to study how Bluetooth worms spread over realistic wireless networks. This required developing an abstract model of the Bluetooth worm and its within-host behavior.

As an illustration of the applicability of our framework, and the utility of activity-based models, we compare the dynamics of Bluetooth worm epidemics over realistic wireless networks and networks generated using random waypoint mobility models. We show that realistic wireless networks exhibit very different structural properties. Importantly, these differences have significant qualitative effect on spatial as well as temporal dynamics of worm propagation. Our results also demonstrate the importance of early detection to control the epidemic.

Categories and Subject Descriptors

I.6.5 [**Simulation and Modeling**]: Model Development—*Modeling methodologies*; I.6.8 [**Simulation and Modeling**]: Types of Simulation—*Discrete event*; K.6.5 [**Management of Computing and Information Systems**]: Security and Protection—*Invasive software (e.g., viruses, worms, Trojan horses)*

Keywords

Malware, Bluetooth, Activity-based, Wireless Epidemics

*This work has been partially supported NSF Nets Grant CNS-062694, HSD Grant SES-0729441, CDC Center of Excellence in Public Health Informatics Grant 2506055-01, NIH-NIGMS MIDAS project 5 U01 GM070694-05, DTRA CNIMS Grant HDTRA1-07-C-0113 and NSF CNS- 0831633.

SIMUTools 2009, Rome, Italy

1. INTRODUCTION

The ever increasing ubiquity of smart digital devices has amplified the opportunities for malware attacks. Cabir [9] and CommWarrior [14] are recent proof-of-concept worms that affected cell phones and PDAs. Though current generation worms do not cause significant harm, increased incidents presents an alarming trend with potential implications in misuse of privacy and identity information of victims and impact to the critical network infrastructure. It is observed that once a certain technology reaches *critical mass* it becomes a target for attacks, and with projected smart phone ownership trends—20.9 million units shipped to North America in 2007 [11]—the problem is bound to cause significant impact. As a result, researchers have begun developing methods and tools to understand the potential impact of malware attack on wireless networks composed of handheld devices.

Currently, analytical models and worm simulations are used to predict the worm spread across wireless networks. Analytical models based on earlier work in mathematical epidemiology provide a natural way to study large systems and have a number of desirable features, such as, closed form expressions for important epidemic quantities, e.g. total number of infected devices. But these models make a number of crucial assumptions, e.g. complete mixing [18, 21], which do not hold in the real world. Detailed simulations that build on well-known network simulators such as NS-2 [1] or Qualnet [16] provide a natural alternative. These simulators allow for a quick and easy implementation and evaluation of the the worm, and have been used in a number of recent studies. Researchers can create different network topologies and observe the growth of an infection with these simulators [19, 21, 18]. Some of these studies use random networks with random mobility models. Although these studies provide significant understanding of the spread, they have two important shortcomings. Firstly, existing tools that conduct a detailed simulation of the protocols do not scale even to small networks, let alone large ones. For example, simulations we conducted to study the spread of Bluetooth worms using NS2 with 500 digital devices in a given location took 2 days to complete. Further, the size of the data sets we are interested in evaluating cannot be handled by such network simulation tools. Secondly, wireless networks formed by smart phones and mobile devices carried by individuals have very different structural properties than the networks formed by random waypoint (RWP) movement models considered in the past. Together, these shortcomings imply that current tools and techniques are not adequate

Digital Object Identifier: 10.4108/ICST.SIMUTOOLS2009.5652
http://dx.doi.org/10.4108/ICST.SIMUTOOLS2009.5652

for studying the potential malware-epidemics over realistic wireless networks.

1.1 Summary of Results

In this paper, we present *EpiNet*: an end-to-end framework for simulating the spread of malware over wireless networks. EpiNet can be used to undertake comprehensive studies related to malware propagation in present and future generation wireless networks. Scalability is an important design consideration. EpiNet can be used for both planning and response phase of a malware epidemic.

As an illustration of the utility of the framework, we simulate the spread of Bluetooth worm over fairly large synthetic, yet realistic networks. The case study is chosen to show how the shortcomings of NS-2 based simulations can be overcome using EpiNet. The synthetic network is derived using a number of innovative modeling techniques, using detailed location and individual data in a US city. To the best of our knowledge, this is the first time detailed activity-based models have been used to study the spreading of malware over wireless networks.

We show that synthetic networks generated from realistic data exhibit features absent in, and significantly different from, networks generated using RWP movement models. We then compare the spatial and temporal dynamics of worm propagation over realistic networks described above and random-waypoint generated models. The results, not surprisingly, show that the dynamics are qualitatively different over these networks. Importantly, those differences highlight the need for early detection for controlling a malware epidemic. Specifically, we observed that most of the infections happen during the first hour of contact with infected device, indicating that coming in contact with the device at inception time can have significant impact in the eventual growth of the infection. This can also potentially impact the way these infections are identified.

Although, we study the spread of Bluetooth worms over a network of devices and use the worm protocol from [19], the framework can be applied to other complicated worm protocols. Since the simulation framework uses an abstracted Bluetooth worm model we are able to obtain orders of magnitude speed-up in comparison with a detailed simulator like NS-2. For the same setting (as with NS-2) EpiNet achieves the results in 14 minutes without parallelization. With parallelization, EpiNet can be used on much larger networks and can help evaluate entire regions in a particular city. We are working on scaling this to entire cities. Simulations of such large scale, city wide networks help in gaining insight into the spread characteristics, and devise intelligent schemes to prevent a widespread digital epidemic.

1.2 Paper Outline

The outline of the paper is as follows: Related work in malware studies is reported in Section 2. Section 3 outlines the details of activity-based mobility model and how wireless networks are built from this model. This section also differentiates the structural measures of the graphs seen using activity-based and the RWP models. The simulation framework and the EpiNet simulator are discussed in Section 4 and Section 5, respectively. Details regarding the experiments, results and analysis are shared in Section 6 and the conclusions are provided in Section 7.

2. RELATED WORK

The study of computer worms in general and Internet worms in particular is not new. Kephart, Chess and White of IBM conducted a study of viral infections in computers using epidemiological models in [12, 13]. Some extremely virulent Internet worms like Code Red have been studied using port scans and computer logs in [22]. [15] models the spread of epidemics using *probabilistic queues* for considering non-homogeneous connectivity distributions that arise in mobile environments. Here the network is modeled as multiple queues emulating the skewed connectivity levels.

Recently, an analytical model of the Bluetooth protocol has been built in [20] and used to study the spread of Bluetooth worms. [19] attempts to study the nature, characteristics and spreading dynamics of such worms through simulations. Small scale NS-2 simulations were performed on a random network and effect of several parameters on worm spreading was observed. [21] studies the effects of different mobility models such as Random Waypoint, Random walk, Random direction, and Random landmark on the spreading characteristics of the Bluetooth worm. Some earlier studies have used realistic data. On a smaller scale, [18] uses a realistic social setting obtained from traces gathered by Bluetooth enabled mobile phones to study worms. The authors in [10] build an event driven simulator for studying malware in mobile devices that are non proximity-based and require human intervention. Such malware, use address books of cell phones to build social networks and spread through VoIP and MMS applications.

Although, such studies provide valuable insight into the spreading characteristics and dynamics, they fail to capture the dynamics of the human population using, interacting and communicating with these devices. Further, random networks lack the structural variability observed in realistic human networks. For example, locations have specific occupancy patterns that impact device network creation, thus, affecting the worm spread. Furthermore, from a service provider's policy standpoint, random networks do not yield a clear understanding of the impact of such worms in actual device (or social) networks. Thus, we seek to address this problem by considering the use of mobility models based on human activity.

3. ACTIVITY-BASED MOBILITY

Similar to human epidemiology studies considering human networks, we study malware propagation in digital devices by constructing human influenced wireless networks. The Bluetooth worm, being proximity-based, fits in well with the human epidemiology model, where, devices in range ($\approx 10\ m$ for a Class II Bluetooth device) are susceptible. We build human contact networks with the Simdemics modeling framework [7, 8] originally built to study the spread of human epidemics and use them to generate device networks. Simdemics uses an individual-based model and maintains an individual's attributes, behaviors and activities. Simdemics consists of several steps as outlined below. **Step 1:** Create synthetic urban population by integrating a variety of databases from commercial and public sources into a common architecture for data exchange that preserves the confidentiality of the original data sets, and yet produces realistic attributes and demographics for the synthetic individuals. A census of our synthetic population yields results that are statistically indistinguishable from the original census data, if they are both aggregated to the block group level [4, 17]. **Step 2:** Use activity-based models for creating human contact networks. A set of activity templates for individuals in the households are determined, based on US census and survey data on activity and time-use surveys [6]. These activity templates describe the sort of activities each household member performs and the time of day they are performed. **Step 3:** Assign detailed route plans to individuals based on the locations where they perform activities and the road network that connects these locations. **Step 4:** Construct detailed movement patterns using a cellular automata based micro-simulation for individuals over the transportation infrastruc-

Digital Object Identifier: 10.4108/ICST.SIMUTOOLS2009.5652
http://dx.doi.org/10.4108/ICST.SIMUTOOLS2009.5652

ture. We do not use steps 3 and 4 for this work.

Thus, essentially the above four steps generate—via a combination of simulation and data fusion techniques—demographic information for each (synthetic) person and location, and a minute-by-minute schedule of each person's activities and the locations where these activities take place. The next step is to generate wireless networks from this raw activity data for simulating the worm spread. We first show how we generate the wireless network using activity information (in Section 3.1) and then discuss the structural measures of the graphs created from this modeling and contrast them with graphs obtained from RWP (in Section 3.2).

3.1 Building Wireless Networks

For simulating the worm spread in a device network, we construct a device network from the raw activity data for each person in the synthetic population. In this section, we describe how we construct this network using *sublocation modeling* within each location. Since wireless links are proximity based and depend on the physical range of the wireless technology, in this paper, we only consider such links. Since we assume Bluetooth, we use the Bluetooth range of 10 m (Class II Bluetooth device). We also do not model effects of noise and fading in indoor environments which effectively decrease the information transmission rates and link availability. We understand that Bluetooth worms can spread through links created based on social contacts (contacts in the address books) through the MMS and SMS applications where physical proximity is not a requirement. We consider this for a future evaluation.

As outlined in Section 3, the synthetic population is configured with activities during the day, involving going to work, shop, school or others for each person in the population. We use these activities to determine the mobility patterns of devices carried by people. The arrival times and durations at activity locations provide the set of the people that potentially come in contact at that location. We call such arrivals at locations for performing an activity *visits*. The visits are generated externally and provided to the simulator during simulation. The visit files contain the following information: person identification number, location identification, arrival time and duration of the activity. For the construction of wireless networks we go through a sublocation modeling process to encode the sublocation information in this visit so that the simulator can construct the wireless network at runtime.

In this paper, we make some assumptions while generating the wireless network: (1) We do not consider the growth of the infection during transit between activities, i.e., we ignore **Step 3** and **Step 4** outlined in Section 3, but use the arrival time at a new location after departing from the previous location. We account for the time spent in transit by modeling activities in *special locations* where the person (or device) is isolated, with no interactions with other devices. (2) We round arrival and departure time instances to 5 minutes (300 s). Though this alters the arrival and departure times of the activity data, it does not cause a significant variation in the overall activity statistics, but can offer some speedup. (3) We are not considering mobility of devices within locations at this time. The activity-based mobility provides a set of devices in a location and the times when the devices arrive and depart. For constructing a wireless network from this input, we model *sublocations* for constructing a coarse grained network and then tune it according to the type of device during simulation to obtain the actual fine grained network, used to propagate the infection.

3.1.1 Modeling Sublocations

We define *Sublocations* as an area within a location where devices interact. We use sublocation modeling to build a *device contact network* within a location. Unlike social contacts of people where physical proximity has no bearing, device network is formed by devices in range. Sublocation modeling only creates a coarse grained device network, i.e., all devices belonging to a sublocation are connected to each other, irrespective of the actual physical distance. Some of these devices may be out of range when considering a particular communication protocol. For instance, Bluetooth devices have a range of 10 m, where as IEEE 802.11 devices may have a range of 500 m. At runtime, the simulator obtains a fine grained device network based on distance. We want to isolate the technology specific aspects from the sublocation modeling and consider them at simulation time. For example, one can also model other signal propagation effects or link availability at runtime. The sublocation modeling is performed outside the simulator as it allows flexibility to use different models for creating sublocations. The results of the sublocation allocation is encoded into the visit information of each person. The *special locations* are also encoded in the visit file with a single sublocation.

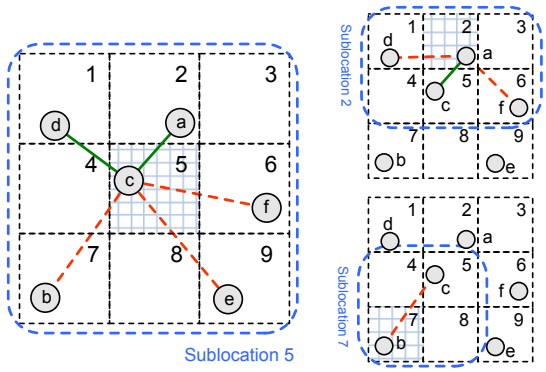
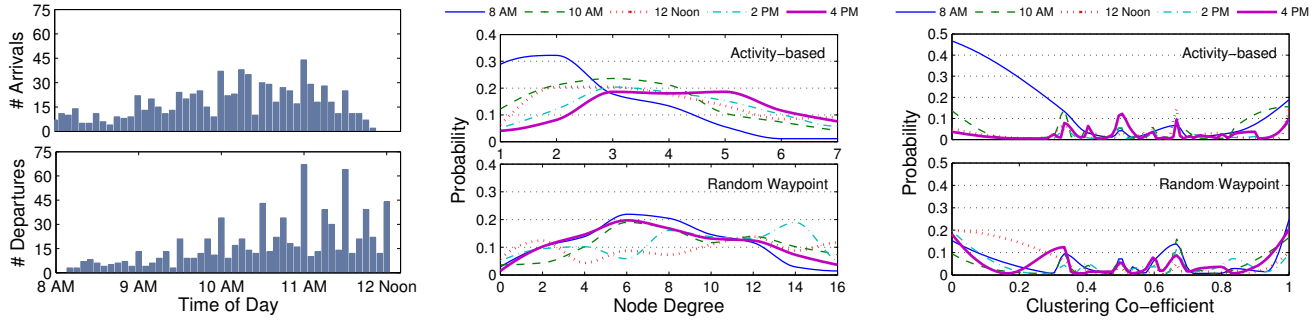


Figure 1: Device network created through sublocation modeling. Solid (green) lines indicate devices in range and dashed (red) lines indicate out of range devices.

The sublocation modeling follows these steps: **Step 1:** *Assign a size to a location.* Using the building occupancy, we assign an area to each location. Since we do not have data for this, we are making up sizes on our own. In the experiments we vary this size to determine its effect on the spread. **Step 2:** *Assign random positions to the devices and determine sublocations.* We assume a certain block size corresponding to the wireless range of the devices. Using this block size, we divide the location into grids of equal size. With the position for each device, we determine the sublocation assignment for that device. **Step 3:** *Form wireless networks based on sublocation modeling.* Since wireless networks are formed based on distance, devices belonging to neighboring sublocations may be in range. So, we model *logical sublocaitons* to include devices belonging to neighboring grids. This sublocation assignment is a coarse grained network and can include devices that are not in range. A fine grained network where links are established based on distance is created at runtime by EpiNet.

Figure 1 shows a location with some blocks (squares with dashed lines) and devices (circles with letters) distributed among the blocks. The solid (green) line indicates the existence of a link based on distance and the dashed (red) line indicates the devices in the logical sublocation but, out of range. 'Sublocation 5' marked in the Figure 1 indicates the sublocation for block 5 and includes all the neighboring blocks $(1, 2, 3, 4, 6, 7, 8$ and $9)$ and the devices present in each of the blocks $(d, a, b, e$ and $f)$. Of these devices a network is

Digital Object Identifier: 10.4108/ICST.SIMUTOOLS2009.5652
http://dx.doi.org/10.4108/ICST.SIMUTOOLS2009.5652



(a) (b) (c)

Figure 2: Graph Characteristics obtained from activity-based mobility models. **(a):** Arrival and departure rates of people (and devices) at a location induced by activity-based mobility models from 8 AM to 12 Noon at every 300 seconds; **(b):** Comparison of degree distribution between activity-based (top) and RWP models (bottom) at different time snap shots during the 8 hour duration of simulation; **(c):** Clustering co-efficient distributions for activity-based (top) and RWP (bottom).

formed only between devices a, c and d. The Sublocation 7 consists of devices b and c with no links being formed. Similarly, Sublocation 2 consisting of devices a, c, d and f forms a link between a and c. Figure 1 shows the devices belonging to the Sublocations 2 and 7 and the device network being formed at these sublocations.

In this section, we have looked at the procedure for creating wireless networks using the proposed sublocation modeling. Next, we need to analyze the structure of these networks and compare them with the structures created when RWP model is used for mobility.

3.2 Graph Characteristics

In this section, we will differentiate the graph structure of the activity-based information and the RWP model used in most wireless network studies. Figure 2 shows the graph metrics derived from the activity-based mobility models we use for the experiments and some differences between RWP model. Figure 2a shows the arrival and departure rates for a 4 hour duration from 8 AM to 12 Noon, at a location when activity-based mobility model is used. We plot the number of new arrivals and departures in steps of 300 s (we consider arrivals and departures at 300 s intervals). Note that the arrival and departure rate vary in a manner that is not easily captured by simple stochastic processes, e.g., Poisson, making simulations necessary for understanding their effect. The RWP mobility model uses number of nodes, area of the location, minimum and maximum speeds and pause times for generating mobility. Since activity-based models do not provide a fixed number of devices in a location, we use the average device occupancy in a location as the total devices in the RWP model, i.e. the number of nodes in RWP is the average occupancy value. So, we use RWP model to generate the mobility information for 28800 s (8 hours), with 0.5 m/s and 1.5 m/s as minimum and maximum speeds and a pause time of 300 s. We take snapshots of the graph every hour and compute the graph metrics on them. The resulting degree distributions and clustering coefficient distributions are shown in Figures 2b and 2c, respectively. Note that the x-axis in Figure 2b is different for each sub-plot. We see that for RWP nodes have highly varied degrees and also has nodes with degree greater than 7 (the maximum degree in case of activity-based models). It is well known that RWP exhibits the property of high device density at the center of an area. This higher degree is due to the clustering of the nodes at the center (shown in Figure 6).

4. THE SIMULATION FRAMEWORK

Our aim in this work is to present a framework for performing simulations containing millions of devices interacting in realistic scenarios, to predict the spread of digital malware. In this paper, we concentrate on Bluetooth worms that have recently affected smart devices and use the worm protocol description in [19]. The same techniques can be extended to other worm protocols by modeling them appropriately and plugging the model into our framework. Our approach requires the following: (1) a means for generating and using realistic mobility patterns for people and/or devices, and, (2) a high level model of the worm protocol. The mobility model provides the environment in which the people interact and the devices come in contact with other devices. We generate the mobility information using activity-based models and conduct the simulation of wireless epidemics on this network. Modeling the worm abstractly improves scalability allowing us to study the system in its entirety and observe the effect of various system level policies. A set of initially infected nodes is assumed to start the infection during simulation. At the end of the simulation we observe the spread of the worm. Figure 3 shows the framework pictorially. We provide details of the various aspects in detail in further sections.

4.1 Modeling the Bluetooth Worm

In this section we describe the design of the Bluetooth worm model. We follow a top down approach: Firstly, we discuss the worm protocol, i.e., the stages of an infected device during each infection cycle. Secondly, we describe the design of the model based on this protocol. Lastly, we discuss the calibration of the model based on actual small scale simulation studies and then compare results from packet level simulations and the model based simulation studies. Table 1 defines some terms used in the worm protocol description.

4.1.1 Worm Protocol

We briefly describe the worm protocol in [19] for completeness. The Bluetooth worm follows four distinct steps during the infection cycle — inquiry phase, page phase, infection phase and idle phase. During the inquiry phase, the infected device obtains information regarding the neighborhood. The inquiring (infected) device sends inquiry requests and waits for responses from other devices. The device continues to perform this till it receives N_{inq}^{to} responses or the request times out in T_{inq}^{to}. The infected device maintains a neighbor

Digital Object Identifier: 10.4108/ICST.SIMUTOOLS2009.5652
http://dx.doi.org/10.4108/ICST.SIMUTOOLS2009.5652

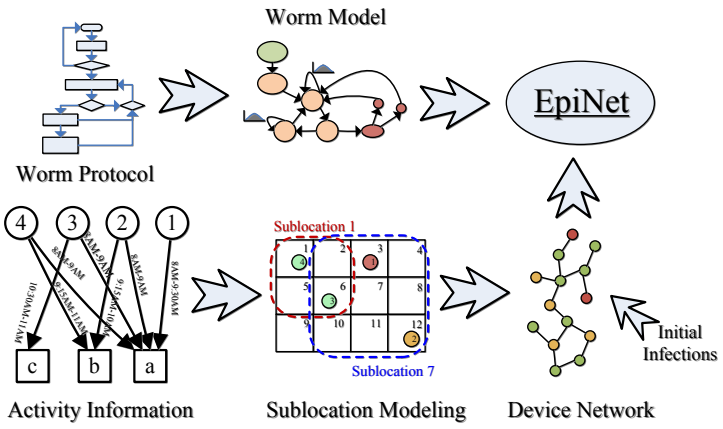


Figure 3: The EpiNet Simulation Framework

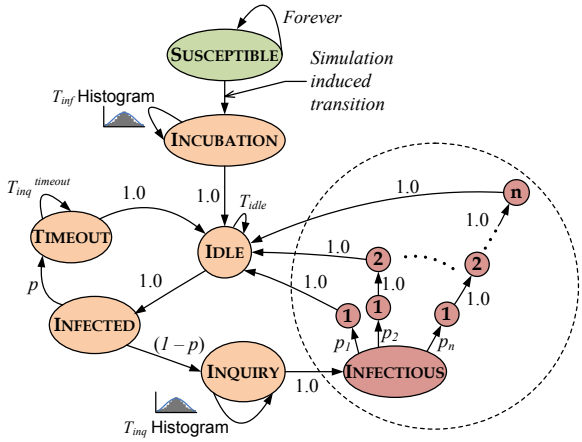


Figure 4: Bluetooth Worm Manifestation Diagram

Terms	Definition
N_{inq}^{to}	Maximum inquiry responses expected by the inquiring device during an inquiry process
N_{inq}	Number of inquiry responses actually received by the inquiring device
T_{inq}^{to}	Inquiry timeout value (inquiring node receives 0 or N_{inq} responses at the end of T_{inq}^{to})
T_{inq}	Total time taken during for an inquiry process
T_{inq}^{n}	inquiry time for the n^{th} inquiry request
T_{inf}	Time taken for the infection
T_{idle}	Idle time between infection cycles

Table 1: Bluetooth Protocol and Worm Parameters

list consisting of nodes that responded to the inquiry. Only devices that are *discoverable* respond to this message. After T_{inq} ($\leq T_{inq}^{to}$) seconds, the infected device enters the infection phase, where it processes N_{inq} neighbors, one at a time. This phase involves sending a *page* request to each neighboring device and on obtaining a page response, sending a message that verifies the condition of this neighbor. The condition of the neighbor can be either susceptible, not susceptible or infected. The infected node then sends the worm packet and payload to a susceptible device. If the neighboring device is not susceptible or already infected, then this packet is sent. These steps are repeated for each device in the neighbor list. The entire process, starting with the inquiry, is repreated after idling for T_{idle}.

Our modeling approach incorporates this worm protocol as a probabilistic timed transition system (PTTS) called the *worm manifestation*. The manifestation represents the various stages of the worm protocol executed by an infected device. The timing of the transitions and the probability of being in particular states of the manifestation is obtained from simulation studies. In these characterization experiments, we conduct exact simulation using packet accurate small scale worm simulation with tools like NS-2 [1]. In Section 4.1.2 we describe the PTTS model for the Bluetooth worm. Further, in Section 6.1 we provide information regarding the validation of the model and compate it to actual simulation results.

4.1.2 Worm Model

The worm model is built to abstract the details of the Bluetooth protocol. Figure 4 shows the worm modeled as a PTTS. Though [20] models the Bluetooth protocol analytically, several modeling assumptions regarding homogeneous distribution of devices and steady state conditions are not realistic for networks we study in the paper, and relaxing these assumptions make modeling them intractible. Thus, we take the simulation approach for studying malware propagation.

Here we ontline the worm model in detail. Initially, all nodes are *susceptible* and remain susceptible until they become infected through the simulation. When conducting the simulation study, we consider some nodes as infected initially and force this transition on these nodes. Once a node becomes infected, the node *incubates* for a certain time denoted by T_{inf}. This is the time taken to actually pass the infection to a new node and is lumped in the *incubating* state. This time is obtained from the T_{inf} histogram. When *incubating* the node cannot infect any other susceptible nodes i.e., the node is not infectious. Once the worm is infected, it begins executing the worm protocol. The protocol begins with an *idle* state where it waits for a certain idle time, T_{idle}, without spreading the infection. T_{idle} is a worm specific parameter that can change based on the worm characteristics and indicates the time between two infection cycles. After this the worm starts the infection cycle by becoming *infected*. In the model, the infected device does not spend any time in the *infected* state and moves either to *timeout* with probability p or to *inquiry* with probability $(1-p)$, where p is the probability that the node's inquiry terminates without receiving any response. This incorporates the conditions when the inquiry terminates with no responses or obtains at least one response. The probability p is obtained from the T_{inq} histogram. Once in the *timeout* state the worm spends $T_{inq}^{timeout}$ time and returns back to the *infected* state. Since there are no neighbors discovered, the node has to perform another inquiry after T_{idle} time. If the node picks the *inquiry* state, after a T_{inq} time it shifts into the *infectious* state when the infectious cycle begins. The time spent in these set of states depends a lot on the Bluetooth protocol and its functioning. Therefore, the probability p_t of choosing a certain cycle time $t : t \in (1, n)$ is deduced from the T_{inf} histogram. This is represented in the disease model as b_t branches with t states in each branch. In each state b_t^i where $i \in (1, t)$, the probability of infecting at time i, p_{inf}^i is derived from the T_{inf} histogram. After the last state in branch b_t the node moves back into the *idle* state and the next infection cycle continues while the node remains infected or the simulation is complete. Here, we make an important assumption: we assume that the probability of infecting a susceptible node at time t_1, $p_{inf}^{t_1}$ is independent of $p_{inf}^{t_2}$ at the next instant t_2. From our validation results in Section 6.1, we find that this seems to work very well.

Digital Object Identifier: 10.4108/ICST.SIMUTOOLS2009.5652
http://dx.doi.org/10.4108/ICST.SIMUTOOLS2009.5652

5. THE EPINET SIMULATOR

We have designed and implemented EpiNet, a simulator for wireless epidemics studies. EpiNet is based on the human epidemic simulator EpiSimdemics [3] with modifications to account for worm time scales and formation of wireless networks. In this section, we provide an overview of EpiNet simulator. EpiNet is a parallel distributed discrete event simulator that models *arrive* and *depart* events at the level of sublocations. When a device arrives at a location l, based on the sublocation s allocated, the simulator generates arrive and depart events at t and $(t + \Delta t)$ respectively. The devices also carry with them the disease model and each device's health state is updated through *disease update* events. The activity schedule determines the time of arrival and departures at different locations and sublocations. All events are sorted according to a global clock, ensuring that all the devices are added into the sublocation (processing arrive events) before the infections are computed. The infection is computed using infection probability p of the infected device's current disease state and a fine grained list of susceptible neighbors. The devices are removed from the sublocation (and location) when the depart events are executed. These events are repeated for every (Δt).

5.1 EpiNet Implementation

EpiNet has three computational components: devices, locations and message brokers. Suppose we have N processing elements (or PEs). The parallelizing strategy of EpiNet is: (1) partition devices and locations into N groups, (2) for each PE implement a person and location manager, PM and LM respectively, and (3) duplicate message brokers into N groups. To optimize the computation and communication effort during the simulation we perform load balancing on the number of devices on each PE by preprocessing the visits.

Algorithm 1: The EpiNet Algorithm

```
for t ← 0; t < T; t ← (t + Δt) do
    foreach i ∈ P_i do
        //send visits to location PEs
1       computeVisits(i, t to t + Δt);
2       sendVisits(MB_i);
    //process visit messages
    Visits ← MB_i.RetrieveMessages();
    synchronize();
    foreach location l_j ∈ L_j do
        //create events and process them
3       makeEvents(Visits);
4       computeInfection();
5       sendOutcomes(MB_i);
    MB_i.RetrieveMessages();
    synchronize();
    //update health state
    foreach i ∈ P_i do
6       updateState();
```

Algorithm 1 shows EpiNet's implementation. The following steps are repeated every Δt until the end of the simulation. The visit files are processed and visits from t to Δt for a device are computed (Line 1) and communicated to the locations PEs (Line 2). Before the further steps are taken, we have to synchronize the PEs so that they have all the arrival information. At each location PE, the message broker receives the visits and creates events for arrival and departure of devices at sublocations inside the location (Line 3). The disease state events are also generated in this step based on the traversal of the disease model. The events are ordered and processed to compute the infections at each sublocation (Line 4). The outcome of this procedure is a set of new infections. These infections are communicated to the respective device PEs (Line 5) and updated (Line 6) so that when they move to other locations the latest disease state is carried along with them.

5.2 Optimizations

We perform several optimizations of the basic EpiNet implementation to make use of the assumptions we have made. Since we round the arrivals and departures to 5 minutes (300 s), i.e., no new arrivals or departures happen with that duration, we can safely perform updates of the location PEs once every 5 minutes. This reduces communication overhead involved in updating the PEs. Since sublocation occupants do not change for this duration, we can use the same events until a change occurs and new devices are received. When the simulator propagates the infection, it computes distance between nodes in the same sublocation. We cache these distances to avoid re-calculating it every Δt. All disease state related events are local to the location PE where the person (device) is performing an activity. So, an update of the disease state is performed on the person PE when a device gets infected and and updated at every Δt. Since special locations do not propagate the infection, we neglect events for them.

6. EXPERIMENTS AND RESULTS

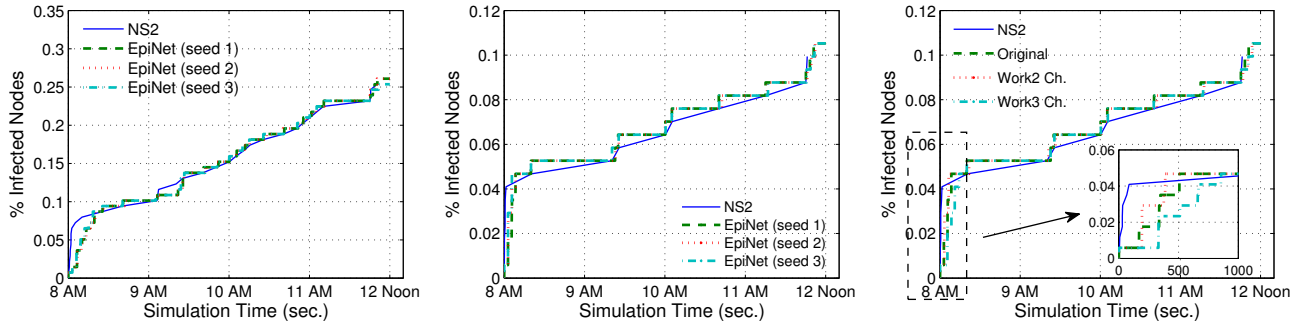
In this section, we describe the experiments we conducted using the EpiNet simulation framework and illustrate the use of the simulator. Firstly, before using the Bluetooth worm model for the study, we validate the model and evaluate its accuracy in Section 6.1. Here we compare the results from the model with detailed simulation studies on a small-scale network and use a single location to evaluate the model. Once validated, we make comparisons with RWP mobility models in Section 6.2 where the difference in outcomes of the infection spread is studied. This shows why we need to use activity-based mobility models. Next, we plug in the worm model into the EpiNet framework and run the simulation studies on large scale networks with more people and locations. We outline the experimental design, experiments we conducted and their results and analysis in Section 6.3.

6.1 Worm Model Validation

We validate the worm model discussed in Section 4.1.2 by comparing the results with detailed packet level simulation using NS-2. For this comparison, we first calibrate the model from an exact simulation with specific parameters—inquiry time and infection time distributions and inquiry timeout. We use the calibrated model for validation. We implemented a Bluetooth worm model in NS-2 using the UCBT's Bluetooth [2] implementation. Since packet level protocol simulations are not scalable, we consider scenarios with 100–400 devices in a single location, with activities over a 4 hour period during the day. Note that we use our activity-based mobility model in NS-2 for the calibration and validation experiments. A worm packet size of 20,000 bytes and probe packet of 27 bytes were used for the simulation studies as in [20].

For all our experiments in this paper, we make the following assumptions with respect to the Bluetooth devices: (1) all devices are discoverable, i.e., they respond to inquiry requests of others, (2) all devices are connectable, i.e., any device can establish a connection with any other device, and (3) all devices are similar and no device heterogeneity is considered. All the assumptions can be relaxed and we can easily make some devices not discoverable or connectable or introduce device heterogeneity and conduct the study. All these can be evaluated with the EpiNet framework without major modifications. We will study these aspects in future.

Digital Object Identifier: 10.4108/ICST.SIMUTOOLS2009.5652
http://dx.doi.org/10.4108/ICST.SIMUTOOLS2009.5652



(a) (b) (c)

Figure 5: Infection CDF comparison with model and actual simulation studies. **(a):** We model the worm's *infectious* state with the probability of infection determined with the histogram H^2_{inf} with x bins of interval size 1 s in location Work2; **(b):** We alter the histogram H^4_{inf} with $\frac{x}{2}$ bins of interval size 2 s in location Work4; **(c):** Using H^2_{inf}, H^3_{inf} and H^4_{inf} for Bluetooth model in location Work4 compared with exact simulation.

Comparison Experiments: Figure 5a shows the comparison between the exact simulation and the Bluetooth worm model for a location. We can see that the simulation using the Bluetooth model tracks the infection CDF for the packet level simulation. The time taken for the NS-2 simulations for this setting was 48 hours (2 days) and for the simulation using the worm model with EpiNet, it was 10 minutes. There is a huge advantage in using the high level simulation in terms of time and the accuracy is very good in comparison with the detailed simulation. For this comparison, we have characterized the models using the T_{inf} histogram for location l (H^l_{inf}) with 1 s as the width of each interval (for example, any T_{inf} between 0 and 1.0 s is counted in bin 1, and so on).

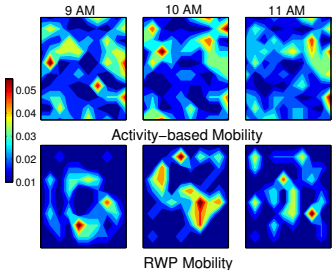


Figure 6: Density distribution snapshots at a location of activity-based mobility model (top) and RWP (bottom).

We also wanted to evaluate the effect of changing some of the histograms on the model and results. First, we wanted to observe how the model performs if the number of sub-states are reduced in the worm model's *infectious* state. For this we construct H^l_{inf} with the interval doubled from 1.0 to 2.0 s (for example, while adding T_{inf} to the bins, we add any T_{inf} value between 0 and 2.0 s into the first bin, and so on). This effectively halves the number of sub-states. We observed how the infection propagates with a change in the model parameters. From Figure 5b, we can see that the model tracks the actual simulation and makes more discrete steps but, follows the trend of the actual results. Second, since we want to evaluate multiple locations simultaneously, we want to use one set of characterizations for all locations. We evaluate whether characterization in one location can be applied to another. So, we compute H^2_{inf}, H^3_{inf} and H^4_{inf} using simulations of respective locations 2, 3 and 4. We then use H^2_{inf} and H^3_{inf} in location 4 and compare the results with the exact and the model with H^4_{inf}. Figure 5c shows the comparison between them. We can see that there is not much difference between the results from actual simulation and the model.

6.2 Comparison with RWP

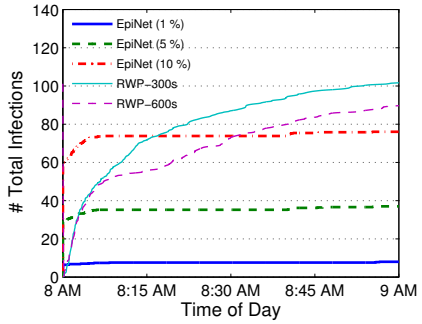
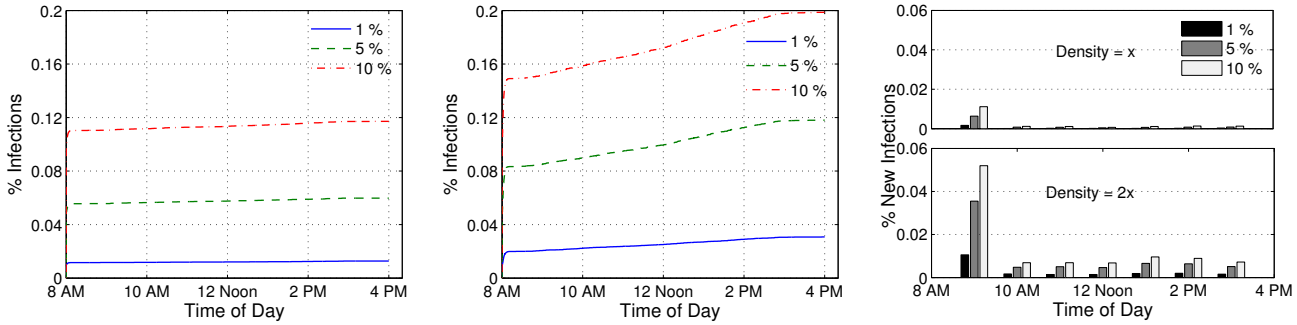


Figure 7: Difference in infection growth between activity-based and RWP models.

In this section, we compare RWP mobility models with activity-based models. We make this comparison for smaller settings and perform exact simulation using NS-2. Making comparisons between activity-based and RWP models is not straight-forward as they depend on diverse parameters. Activity models are governed by arrivals and departures from the location making instantaneous occupancy vary with time, while RWP uses a constant number of devices for the entire duration. In addiion, RWP uses minimum and maximum speeds and maximum pause time for device mobility. So, we consider a single location and use the average instantaneous occupancy from 8 AM to 9 AM (109 devices) as the number of nodes. The instantaneous occupancy of the location in the activity model ranges from 91–147 and devices arrive (or depart) at 300 s intervals and the total occupancy of the location is 572. We consider the same area for both the cases. For RWP, we consider mobility with 0.5 m and 1.5 m as minimum and maximum speeds, respectively, and evaluate two cases with 300 s and 600 s pause

Digital Object Identifier: 10.4108/ICST.SIMUTOOLS2009.5652
http://dx.doi.org/10.4108/ICST.SIMUTOOLS2009.5652



(a) Effect of initial infections (b) Effect of density (2x) (c) Rate of new infections

Figure 8: Results of Experiment I: **(a):** Infection growth with varying initial infection size. **(b):** Infection growth when density is doubled. **(c):** New infections occurring over the previous hour.

times. Here we are interested in comparing how the models differ when these parameters are configured. Figure 6 shows the density (devices/unit area) comparison at the location between activity model (top) and RWP (bottom) at different times of the day. Note that the actual number of devices in the activity models depend on activities and this number increases (or decreases) when people enter (or leave) the location. However, RWP model generates uneven device densities with higher center densities [5].

We simulate the scenario for an hour and observe the number of nodes that are infected by 9 AM. In case of the activity models, we randomly select 1%, 5% and 10% of the location's total occupancy as the initial infection size, and consider a single randomly infected device for RWP. Since there is no direct relationship between devices in RWP and activity models, we cannot select the same device to be infected in both cases. We use NS-2 to simulate the RWP model and EpiNet to simulate the activity-based model. We compare the number of devices infected at the end of the hour. Figure 7 shows the infection growth (averaged over 5 seeds) comparison between Epinet (EpiNet 1%, EpiNet 5% and EpiNet 10%) and RWP models (RWP-300s and RWP-600s). For the RWP model, the infections surge initially and infect almost all of the devices present in the location within the first hour. This can be attributed to the higher degree of the RWP network as seen in Figure 2b (in page). However, the initial surge of infections quickly saturates for activity models. Note, that the initial infection size is different in both cases. This effect of varying infection growth is mainly due to the activities of people in the location. For example, a person carrying an infected device may leave the location without interacting with a lot of people or devices due to a wide variation in density of the devices. Random models as shown in Figure 7 can predict a very high level of growth of the infection when, in reality the growth saturates.

In activity-based models, we can also consider the influence of special activities (for example, football games, public transport system, classrooms, etc.) where density of devices vary across locations and also influence the growth differently. Growth of a malware can be observed and studied both in temporal and spacial terms when realistic mobility is considered.

6.3 Experiment Setup and Design

What are the parameters that impact the spread of the worm? How does the worm's idle time impact the spreading speed? Does the time of initial infection affect the spread? Does the number of initial infections matter to the spread? How do these parameters interact? These are some of the questions we are going to answer by a factorial experiment design using the simulation framework. The parameters we are studying can be divided into worm parameters (T_{idle}), network parameters (location density) and system parameters (number of initial infections and inception time of the worm). All these parameters, individually or collectively impact the spread and we intend to bring these effects out from the simulation experiments. For example, lower idle times between infection cycles can increase the rate of spread locally provided the device density in that location is high. The idle time can cause significant spatial effect if the infected devices were to be mobile, thus triggering infections at different locations. Figure 9 shows the parameters varied in the experiments.

1. **Number of initial infections:** 1%, 5%, 10%
2. **Idle time (T_{idle}):** 20 s, 10 s
3. **Inception time of worm:** 8 AM, 10 AM
4. **Location density:** We varied the density of locations from x to $2x$
5. **Simulator used:** EpiNet framework
6. **Simulation runs:** 5 runs for any combination of input parameters
7. **Simulation duration:** 8 hours from 8 AM to 4 PM
8. **Social network:** Simdemics framework was used to generate the demographic and activities. We consider only locations in 60602 zipcode in Chicago downtown area and neglect activities with duration less than 300 s
9. **Population and location size:** 20000 devices and 30000 locations
10. **Computing resources:** 4 CPUs of 1GHz Pentium III Linux cluster with 1 GB RAM
11. **Average runtime:** 45 hours

Figure 9: Experimental setting and parameters studied

We select a region in Chicago downtown corresponding to the zipcode 60602 and select the locations in this area ($\approx$ 30000 locations). We select a population size of $\approx$ 20000 people and extract their activities for an 8 hour duration, from 8 AM to 4 PM, at these locations. The people in the ages in the range of 20–50 were selected to carry digital devices. We assume that device penetration is 100% and each device is identical and all the devices are susceptible. The arrival and departure times of the people's activities were rounded to 5 minutes. We encode the worm model in a disease manifestation file and provide it as input to the EpiNet. The activity information is provided to the sublocation modeling pro-

Digital Object Identifier: 10.4108/ICST.SIMUTOOLS2009.5652
http://dx.doi.org/10.4108/ICST.SIMUTOOLS2009.5652

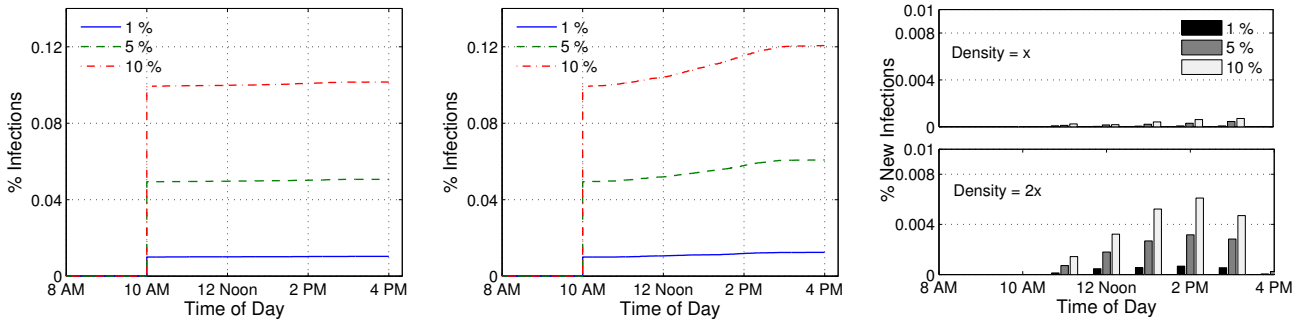


Figure 10: Results of Experiment II: **(a):** Infection growth with varying infection seed time (10 AM). **(b):** Infection growth with double the density than with (a). Here we halved the total area of each location. **(c):** Plots the number of new infections occurring over the previous hour. It is the difference between the infection at the beginning of the next hour and the current time.

gram to look at the activities and perform sublocation allocation and load balancing on the number of devices in each PE. This information is encoded into visit files. The experiments takes ≈ 2 days to complete (45 hours) for 8 hours of simulation time. Each experiment was conducted for 5 seed values. Sections below outline the different experiments, the parameters studied, results and their analysis.

6.3.1 *Experiment I: Varying Initial Infection Size*

In these experiments, we increase the number of the initially infected devices from 1% to 10%. We conduct these experiments with infection starting at 8 AM. We also vary the location sizes (from x) to double the device densities (to $2x$) and observe the spread patterns. We expect the number of infections to be higher and the speed to increase for higher initial infection size. When the density is doubled, this increase is exaggerated further.

Results: Figure 8a shows the difference between the infection growth with change in initial infection size (averaged for the 5 seeds). From this, we can see that the rate of infection is highest for 10% initial infection. In fact, within the first few minutes of the outbreak the rate of new infections is the highest. In all cases, 10% of the new infections happen during the first hour of the outbreak. After this, the infection rate falls and almost remains constant. Figure 8b shows the change in infection rate as density is doubled at each location considered. Here we observe increased rates of new infections, $5\times$ larger, within the first hour of infection. Figure 8c shows the change in rate with respect to the time of day (averaged for 5 seeds). We can see that the rate change is highest for the case with 10% initial infection size. We can also observe that there are slight increases as the day progresses, reducing to the lowest at 12 Noon and again increasing towards mid afternoon and dropping at around 4 PM. These changes can be attributed to the movement of people in and out of a location and change in the topology as a result of it. Overall, the infection seems to spread to almost all the new arrivals and then saturates (as there are no susceptible devices), until the next wave of arrivals and infections.

6.3.2 *Experiment II: Varying the worm seed time*

Here, we consider 8 AM and 10 AM as the seed time of the worm and observe the effect on the growth of the infection. For example, in work locations people arrive at the location around 8 AM with the number slightly growing till about midmorning, remaining constant for about sometime till around noon and then starts decreasing towards noon. Other location types such as restaurants have a different pattern with the number of people in such locations increasing at times when people have lunch or dinner. Thus, the seed time, based on the type of location and arrival patterns, can have significant impact. This cannot be observed in other random mobility models and also present an realistic estimate of the infection size. Here, we wanted to verify if we can say something with change in seed time. Does the seed time impact the growth of the worm? We consider seed time of 10 AM and observe the growth rate of the infection.

Figure 10a shows the infection CDF. We can see that the infection starting at 10 AM has a negative impact on the growth in comparison with the seed at 8 AM (Figure 8a). The final infection size at 4 PM for the seed time of 10 AM is 12% of total devices (a 20% increase), while for a seed at 8 AM it is 20% (an increase of 100%). This clearly shows that the seed time of the infection in a realistic scenario can greatly impact the growth of infection in the devices.

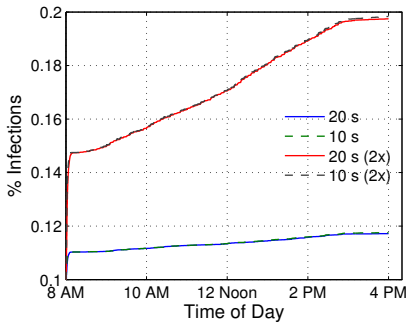


Figure 11: Results of Experiment III: Comparison of effect of reducing idle time from 20 s to 10 s on the growth of the infection.

6.3.3 *Experiment III: Varying the worm idle time*

In this set of experiments, we change the worm's idle time from 20 s to 10 s and observe the growth of the infection. Idle time of the worm is the time spent by the worm in-between infection cycles during which the infected device does not spread the infection. We intend to determine if this is a factor in the growth of the infection.

Figure 11 shows the effect of reducing the idle time from 20 s to 10 s for both the location density values. From this, it appears

Digital Object Identifier: 10.4108/ICST.SIMUTOOLS2009.5652
http://dx.doi.org/10.4108/ICST.SIMUTOOLS2009.5652

that the idle time has very slight impact on the actual growth, in the cases we have considered. When location density is doubled, the idle time results in 16 more infections than with the idle time of 20 s This shows that the idle time of the worm does not make a real impact on the overall number of infections. It will speed up local infections slightly, not enough to impact in the larger setting. This is because, in the setting we have evaluated, unlike in random mobility models, the neighbor set does not change significantly. It can happen that all neighboring nodes are already infected and does not change the total number of infections. To evaluate idle time's effect, we may have to create special activity scenarios that cause extremely high device densities—movie theaters or football games—so that every inquiry process retuns new susceptible neighbors.

7. CONCLUSION AND FUTURE WORK

We presented a EpiNet: new parallel simulation framework for simulating spread of malware over realistic wireless networks. Additional details of the simulation framework and methods for generating realistic wireless networks are omitted due to lack of space. We demonstrate how the simulator can be configured with the activity based mobility information and a disease model (or manifestation) in the form of a probabilistic timed transition system. We have validated and calibrated the model with a detailed small scale simulation and is found to obtain the same results. We have used the framework and conducted simulation involving $20,000$ nodes distributed across a certain region in downtown Chicago.

The following broad conclusions are obtained: (1) The initial size of the infection impacts in the growth, the higher the initial infection size, the faster is the growth of infection. (2) The seed time of the infection also makes a difference. Since we are looking at activity-based models for user mobility and presence at particular locations, the seed time does impact the spread and the growth of infection. Earlier seed times (for example, 8 AM) cause significantly more impact than the other seed time (10 AM) that we considered. Though this cannot be generalized, it does emerge that seed time is important to evaluate the potential growth of any malware after an outbreak. (3) Nothing conclusive can be claimed regarding the alteration of the idle times, and, in the set of experiments we conducted, we do not find a significant impact on the growth of the infection. In the setting of a large area and number of devices considered, it does not seem to make a significant impact to alter the growth. Larger idle times or smaller idle times do not make sense in terms of the Bluetooth worm setting as it will significantly slow down or cause higher power consumption on the infected device, respectively.

We are currently working on (i) mechanisms for tracking and detecting such worms, (ii) developing intervention strategies in the event of an epidemic onset, and (iii) scaling the simulation so as to be able to scale cities with 1 Million+ individuals.

8. REFERENCES

[1] The Network Simulator NS-2. http://www.isi.edu/nsnam/ns/.

[2] UCBT: Bluetooth Extension for NS2 at University of Cincinnatti. http://www.ececs.uc.edu/ cdmc/ucbt/.

[3] C. L. Barrett, K. R. Bisset, S. G. Eubank, X. Feng, and M. V. Marathe. EpiSimdemics: an Efficient Algorithm for Simulating the Spread of Infectious Disease over Large Realistic Social Networks. In *SC '08*, pages 1–12, 2008.

[4] R. J. Beckman, K. A. Baggerly, and M. D. McKay. Creating synthetic base-line populations. Transportation Research Part A – Policy and Practice, 1996.

[5] C. Bettstetter. Smooth is better than sharp: a Random Mobility Model for Simulation of Wireless Networks. In *MSWIM '01*, pages 19–27, 2001.

[6] J. L. Bowman and M. E. Ben-Akiva. Activity-based Disaggregate Travel Demand Model System with Activity Schedules. *Transportation research. Part A, Policy and practice*, 35A, 2001.

[7] S. Eubank, H. Guclu, V. S. A. Kumar, M. V. Marathe, A. Srinivasan, Z. Toroczkai, and N. Wang. Modeling Disease Outbreaks in Realistic Urban Social Networks. *Nature*, 429:180–184, May 2004.

[8] S. Eubank, V. S. A. Kumar, M. V. Marathe, A. Srinivasan, and N. Wang. Structural and algorithmic aspects of massive social networks. In *SODA '04*, pages 718–727, 2004.

[9] P. Ferrie, P. Szor, R. Stanev, and R. Mouritzen. Security Responses: Symbos.cabir. Symantec Corporation, June 2004.

[10] C. Fleizach, M. Liljenstam, and *et. al.*. Can you infect me now?: Malware Propagation in Mobile Phone Networks. In *WORM '07*, pages 61–68, 2007.

[11] R. Kairer. Smartphones Showed Strong Growth in 2007, February 2008. http://www.palminfocenter.com/news/9617/smartphones-showed-strong-growth-in-2007.

[12] J. O. Kephart and S. R. White. Directed-Graph Epidemiological Models of Computer Viruses. *Research in Security and Privacy, 1991. Proceedings., 1991 IEEE Computer Society Symposium on*, 00:343–359, May 1991.

[13] J. O. Kephart, S. R. White, and D. M. Chess. Computers and Epidemiology. *IEEE Spectrum*, 30(5):20–26, 1993.

[14] M. Lactaotao. Security Information: Virus Encyclopedia: Symbos comwar.a. Trend Micro Incorporated, March 2005.

[15] J. W. Mickens and B. D. Noble. Modeling Epidemic Spreading in Mobile Environments. In *WiSe '05: Proceedings of the 4th ACM workshop on Wireless security*, pages 77–86. ACM, 2005.

[16] Scalable Network Technologies. QualNet. http://www.scalable-networks.com/.

[17] P. Speckman, K. Vaughn, and E. Pas. A Continuous Spatial Interaction Model: Application to Home-Work Travel in Portland, Oregon. Transportation Research Board 1997 Annual Meeting, 1997.

[18] J. Su, K. K. W. Chan, A. G. Miklas, *et al.*. A Preliminary Investigation of Worm Infections in a Bluetooth Environment. In *WORM '06: Proceedings of the 4th ACM Workshop on Recurring Malcode*, pages 9–16, 2006.

[19] G. Yan and S. Eidenbenz. Bluetooth Worms: Models, Dynamics, and Defense Implications. In *ACSAC '06*, pages 245–256, Washington, DC, USA, 2006. IEEE Computer Society.

[20] G. Yan and S. Eidenbenz. Modeling Propagation Dynamics of Bluetooth Worms. In *ICDCS*, pages 42–52. IEEE Computer Society, 2007.

[21] G. Yan, H. D. Flores, L. Cuellar, *et al.*. Bluetooth Worm Propagation: Mobility Pattern Matters! In *ASIACCS '07*, pages 32–44, New York, NY, USA, 2007. ACM.

[22] C. C. Zou, W. Gong, and D. Towsley. Code Red Worm Propagation Modeling and Analysis. In *CCS '02*, pages 138–147, 2002.

Digital Object Identifier: 10.4108/ICST.SIMUTOOLS2009.5652
http://dx.doi.org/10.4108/ICST.SIMUTOOLS2009.5652