Optimal Distributed Malware Defense in Mobile Networks with Heterogeneous Devices

Yong Li, *Member*, *IEEE*, Pan Hui, *Member*, *IEEE*,
Depeng Jin, *Member*, *IEEE*, Li Su, and Lieguang Zeng, *Member*, *IEEE*

Abstract—As malware attacks become more frequently in mobile networks, deploying an efficient defense system to protect against infection and to help the infected nodes to recover is important to prevent serious spreading and outbreaks. The technical challenges are that mobile devices are heterogeneous in terms of operating systems, the malware infects the targeted system in any opportunistic fashion via local and global connectivity, while the to-be-deployed defense system on the other hand would be usually resource limited. In this paper, we investigate the problem of how to optimally distribute the content-based signatures of malware, which helps to detect the corresponding malware and disable further propagation, to minimize the number of infected nodes. We model the defense system with realistic assumptions addressing all the above challenges that have not been addressed in previous analytical work. Based on the framework of optimizing the system welfare utility, which is the weighted summation of individual utility depending on the final number of infected nodes through the signature allocation, we propose an encounter-based distributed algorithm based on Metropolis sampler. Through theoretical analysis and simulations with both synthetic and realistic mobility traces, we show that the distributed algorithm achieves the optimal solution, and performs efficiently in realistic environments.

Index Terms—Security threat, mobile malware, distributed algorithm, heterogeneous mobile networks

1 Introduction

The target landscape for malware attacks (i.e., viruses, spam bots, worms, and other malicious software) has moved considerably from the large-scale Internet to the growingly popular mobile networks [1], with a total count of more than 350 known mobile malware instances reported in early 2007 [2]. This is mainly because of two reasons. One is the emergence of powerful mobile devices, such as the iPhone, Android, and Blackberry devices, and increasingly diversified mobile applications, such as multimedia messaging service (MMS), mobile games, and peer-to-peer file sharing. The other reason is the emergence of mobile Internet, which indirectly induces the malware. Malware residing in the wired Internet can now use mobile devices and networks to propagate. The potential effects of malware propagation on mobile users and service providers would be very serious [3]. Understanding the behaviors and damages of mobile malware, and designing an efficient detection and defense system are necessary to prevent large-scale outbreaks [4], [5]; and it should be an urgent and high-priority research agenda.

- *Y. Li, D. Jin, L. Su, and L. Zeng are with the State Key Laboratory on Microwave and Digital Communications, Tsinghua National Laboratory for Information Science and Technology (TNLIST), Department of Electronic Engineering, Tsinghua University, Room-Building 10-202, Beijing 100084, China.*
  E-mail: liyong07@tsinghua.edu.cn, jindp@mailtsinghua.edu.cn.
- *P. Hui is with the Department of Computer Science and Engineering, The Hong Kong University of Science and Technology, Hong Kong, Telekom Innovation Laboratories, Germany, and Aalto University, Finland.*
  E-mail: panhui@cse.ust.hk.

Manuscript received 19 Mar. 2012; revised 10 Sept. 2012; accepted 30 Nov. 2012; published online 12 Dec. 2012.
For information on obtaining reprints of this article, please send e-mail to: tmc@computer.org, and reference IEEECS Log Number TMC-2012-03-0137.
Digital Object Identifier no. 10.1109/TMC.2012.255.

Currently, mobile malware can propagate through two different dominant approaches. Via MMS, a malware may send a copy of itself to all devices whose numbers are found in the address book of the infected handset. This kind of malware propagates in the social graph formed by the address books, and can spread very quickly without geographical limitations. The other approach is to use the short-range wireless media such as Bluetooth to infect the devices in proximity as "proximity malware." Recent work of Wang et al. [1] has investigated the proximity malware propagation features, and finds that it spreads slowly because of the human mobility, which offers ample opportunities to deploy the defense system. However, the approach for efficiently deploying such a system is still an ongoing research problem. In this paper, we are the first to address the challenges of designing a defense system for both MMS and proximity malware. We introduce an optimal distributed solution to efficiently avoid malware spreading and to help infected nodes to recover.

Consider a mobile network where a portion of the nodes are infected by malware. Our research problem is to deploy an efficient defense system to help infected nodes to recover and prevent healthy nodes from further infection. Typically, we should disseminate the content-based signatures of known malware to as many nodes as possible [6], [7]. Consequently, distributing these signatures into the whole network while avoiding unnecessary redundancy is our optimization goal. However, to address the above problem in the realistic mobile environment is challenging for several reasons. First, typically we cannot rely on centralized algorithms to distribute the signatures because the service infrastructure is not always available. Therefore, a sensible way for signature distribution is to use a distributed and cooperative way among users. Second,

mobile devices in general have limited resources, i.e., CPU, storage, and battery power. Although their storage and CPU capacity has been increasing rapidly, it is still very resource-limited compared with desktops. Hence, in the to-be-deployed defense system, we should adequately consider the limitation of resources, especially the memory capacity to store the defense software and signatures. Finally, the mobile devices are heterogeneous in terms of operating systems (OS), and different malware targets different systems. These heterogeneous features as well as the propagation via both local and global connectivity should be taken into consideration in the design of defense system for real use.

In this paper, we propose an optimal signature distribution scheme by considering the following realistic modeling assumptions: 1) the network contains heterogeneous devices as nodes, 2) different types of malware can only infect the targeted systems, and 3) the storage resource of each device for the defense system is limited. These assumptions are usually not addressed in previous analytical works for simplicity reasons [4], [5], [7]. Our contributions are summarized as follows:

- We formulate the optimal signature distribution problem with the consideration of the heterogeneity of mobile devices and malware, and the limited resources of the defense system. Moreover, our formulated model is suitable for both the MMS and proximity malware propagation.
- We give a centralized greedy algorithm for the signature distribution problem. We prove that the proposed greedy algorithm obtains the optimal solution for the system, which provides the benchmark solution for our distributed algorithm design.
- We propose an encounter-based distributed algorithm to disseminate the malware signatures using Metropolis sampler [8]. It only relies on local information and opportunistic contacts. Through theoretical proof and extensive real and synthetic traces driven simulations, we show that our distributed algorithm approaches the optimal system performance.

The rest of the paper is organized as follows: We describe the system model in Section 2, and then formulate the associated optimization problem and give a centralized algorithm in Section 3. In Section 4, we design a distributed algorithm to approach the optimal solution. In Section 5, we introduce the experimental environment for performance evaluation and provide extensive simulation results. Finally, we present the related work in Section 6 and conclude the paper in Section 7.

2 System Description

In this section, we first give an overview description of signature distribution in the defense system, and then present the ordinary differential equation (ODE) model for the studied system.

2.1 System Overview

Mobile malware that spreads in the mobile networks typically exploits both the MMS and opportunistic contacts to propagate from one device to another. In the network, there are different types of handsets and each malware only targets handsets with a specific OS. In the defense system, we use some special nodes named *helper* to distribute the signatures into the network. Generally speaking, the deployed *helpers* can be stationary base stations or access points. However, since mobile nodes are more efficient to disseminate content and information in the network [9], we focus on the case of mobile helpers. Consequently, there is limitation in storage on each mobile device for deploying the defense system. Although currently most smartphones have gigabytes of storage, users usually will not allocate all of them for the usage of malware defense. Our goal is to minimize the malware infected nodes in the system by appropriately allocating the limited storage with the consideration of different types of malware.

2.2 Notations and Malware Spreading Model

We consider a system of N heterogeneous wireless nodes belonging to K types (e.g., type of OS), which can be infected by K types of malware, denoted by set $\mathbb{K}$. In the defense system, we assume that there are S helpers, denoted by set $\mathbb{S}$, storing the signatures to help other nodes with detecting the malware. Furthermore, let A_s denote the maximum number of signatures that can be stored at helper s, and u_k denote the number of helpers for malware k. To describe how the signatures of malware are stored in the helpers, we define $x_{s,k}$ as the indicator whether helper s has the signature to detect malware k. Since different types of malware will infect different classes of nodes, we let v_k denote the maximum number of nodes that malware k can infect, and let v_k^0 denote the number of infected nodes at the starting time.

We first consider the number of nodes infected by malware k in the system at time t, denoted by $\zeta_k(t)$. The dynamic system of malware spreading and defending can be described by the following ordinary differential equation model:

$$\frac{d\zeta_k(t)}{dt} = \lambda_{k1}(v_k - \zeta_k(t))\zeta_k(t) - \lambda_{k2}u_k\zeta_k(t), \tag{1}$$

where $\lambda_{k1}v_k$ and $\lambda_{k2}u_k$ are the malware spreading rate depending on the number of infected nodes, and recovering rate that depends on the number of the helpers with signature of malware k, respectively. We obtain this ordinary differential equation by the epidemic spreading model that is widely used in the malware attack analysis [4], which is based on the SIS model [10]. On the other hand, we note that this equation is based on the fluid model. The use of fluid approximation is a standard tool in modeling the information propagation of Delay Tolerant Network (DTN) [11], [4]. Moreover, we will verify this model by comparing the simulation and theoretical results in the Section 3.2. Here, we summarize the commonly used variables throughout the paper in Table 1.

3 Problem Formulation and Centralized Algorithm

Based on the malware spreading model, we first formulate the problem, and then give a greedy algorithm to achieve the optimal signature distribution.

TABLE 1
List of Commonly Used Variables Throughout the Paper

Variable	*Description*
N	Number of wireless nodes in the system
$\mathbb{K}$	Malware set, which includes K types
K	Number of the types of malware
v_k	Maximum number of nodes that malware k can infect
v_k^0	Number of infected nodes infected by malware k at the starting time
$x_{s,k}$	Indicator whether helper s has the signature to prevent malware k
A_s	Maximum number of signatures that can be stored at helper s
u_k	Number of helpers for malware k
h_k	Number of infected nodes by malware k in the system at time L
$\zeta_k(t)$	Number of infected nodes by malware k in the system at time t
λ_{k1}	Spreading rate of malware k
λ_{k2}	Recovering rate of malware k,
L	Final system time used to count the number of infected nodes in the network
$F_k(x)$	System utility function for defending malware k, introduced in Section III-A
w_k	System welfare factor to weight system contributions of defending malware k, introduced in Section III-C
T	Parameter in Gibbs ditribution named system temperature used in Algorithm 2 of Section IV

3.1 Utility Function

To solve (1), we define $z(t)$ as follows:

$$z(t) = (-\lambda_{k1})\zeta_k(t) + \frac{1}{2}(\lambda_{k1}v_k - \lambda_{k2}u_k). \tag{2}$$

Applying variable substitution and by (1), we have

$$\frac{dz(t)}{dt} = z^2(t) - \frac{1}{4}(\lambda_{k1}v_k - \lambda_{k2}u_k)^2. \tag{3}$$

We assume that $z(t) \neq \pm\frac{1}{2}(\lambda_{k1}v_k - \lambda_{k2}u_k)$ is satisfied,[1] then we have

$$\frac{dz}{z^2 - \frac{1}{4}(\lambda_{k1}v_k - \lambda_{k2}u_k)^2} = dt.$$

Integrating both sides of the above equation, we can get

$$z(t) = \frac{(\lambda_{k1}v_k - \lambda_{k2}u_k)(Ce^{(\lambda_{k1}v_k - \lambda_{k2}u_k)t} + 1)}{2(1 - Ce^{(\lambda_{k1}v_k - \lambda_{k2}u_k)t})}, \tag{4}$$

where the integral constant $C = \frac{z(0) - \frac{1}{2}(\lambda_{k1}v_k - \lambda_{k2}u_k)}{z(0) + \frac{1}{2}(\lambda_{k1}v_k - \lambda_{k2}u_k)}$.

Using (2) and (4), we have

$$\zeta_k(t) = \frac{\lambda_{k1}v_k - \lambda_{k2}u_k}{\lambda_{k1}\left(1 - \frac{\lambda_{k1}v_k^0 - \lambda_{k1}v_k + \lambda_{k2}u_k}{\lambda_{k1}v_k^0}e^{-(\lambda_{k1}v_k - \lambda_{k2}u_k)t}\right)}. \tag{5}$$

1. If $z(t) = \frac{1}{2}(\lambda_{k1}v_k - \lambda_{k2}u_k)$, then $\zeta_k(t) = 0$. On the other hand, if $z(t) = -\frac{1}{2}(\lambda_{k1}v_k - \lambda_{k2}u_k)$, $\zeta_k(t) = \frac{\lambda_{k1}v_k - \lambda_{k2}u_k}{\lambda_{k1}}$, which is a simple case where the number of nodes infected by malware k is proportional to the number of helpers in the deployed defense system. Therefore, we assume $z(t) \neq \pm\frac{1}{2}(\lambda_{k1}v_k - \lambda_{k2}u_k)$ is satisfied in the following discussion.

Define ζ_k^L as the number of infected nodes by malware k at system time L, and define $\zeta_k^L(u_k)$ as the function of the number of infected nodes by malware k with argument u_k. Note that the system time L defined here is just a reference time to observe the system performance. In practice, it can be chosen by the network administrators according to their specific security requirements. We assume that for each malware there is an underlying utility function $G_k(\zeta_k^L)$ that specifies the system utility of defending malware k given the number of infected nodes at time L. It is natural that $G_k(\zeta_k^L)$ is a nonincreasing function of ζ_k^L. According to different network scenarios that reflect the relationship between the system utility increasing rate and malware defense effect, the utility function can be designed flexibly.

We further assume $G_k(\zeta_k^L)$ is a concave function, and denote $F_k(u_k) = G_k(\zeta_k^L(u_k))$ as the system utility of defending malware k given the number of helpers carrying the signature of malware k, u_k. Then, we obtain the following properties about $\zeta_k^L(u_k)$ and $F_k(u_k)$.

Lemma 1. *$\zeta_k^L(u_k)$ is a monotonic decreasing, strictly convex function of the number of helpers in the defense system with the signature of malware k, u_k, when*

$$L \geq \frac{1 + \sqrt{1 + 2Bz}}{z(Bz + 1)}$$

where $z = \lambda_{k2}u_k - \lambda_{k1}v_k$ and $B = \frac{1}{\lambda_{k1}v_k^0}$.

Proof. Perform the variable substitution of $z = \lambda_{k2}u_k - \lambda_{k1}v_k$ and $B = \frac{1}{\lambda_{k1}v_k^0}$, then $\zeta_k^L(u_k)$ becomes

$$\zeta_k^L(z) = \frac{-z}{\lambda_{k1}(1-(1+Bz)e^{zL})}. \quad (6)$$

We investigate the monotonicity and concavity-convexity of $\zeta_k^L(z)$ first. Define function $H(z) = 1-(1+Bz)e^{zL}$, then we get $\zeta_k^L(z) = -z/(\lambda_{k1}H(z))$. Calculate the first- and the second-order derivatives of $\zeta_k^L(z)$, we have

$$\frac{d\zeta_k^L(z)}{dz} = \frac{-H(z)+zH'(z)}{\lambda_{k1}H^2(z)},$$
$$\frac{d^2\zeta_k^L(z)}{dz^2} = \frac{zH''(z)H(z)-2H'(z)(-H(z)+zH'(z))}{\lambda_{k1}H^3(z)}.$$

Define $G_1(z) = -H(z)+zH'(z)$, thus $\frac{d\zeta_k^L(z)}{dz} = \frac{G_1(z)}{\lambda_{k1}H^2(z)}$. Next, we will show $G_1(z)$ is nonpositive for all $z \in \mathbb{R}$. Considering the definition of $H(z)$, we know

$$G_1(z) = -1+e^{zL}(1-zL(1+Bz)),$$

thus

$$\frac{dG_1(z)}{dz} = -e^{zL}(zL^2(1+Bz)+2BLz).$$

Now, it is clear that $\frac{dG_1(z)}{dz} > 0$ if and only if $z \in (-\frac{L+2B}{BL}, 0)$; consequently, $G_1(z)$ is decreasing on interval $(-\infty, -\frac{L+2B}{BL}]$ as well as on $[0,\infty)$, respectively, and $G_1(z)$ is increasing on interval $[-\frac{L+2B}{BL}, 0]$. Combined with the facts that $G_1(0) = 0$ and $\lim_{z\to-\infty} G_1(z) = -1$, we have already proved $G_1(z) \le 0$. Consequently,

$$\frac{d\zeta_k^L(z)}{dz} \le 0,$$

therefore, $\zeta_k^L(z)$ is decreasing for $z \in |\mathbb{R}$.

Similarly, define

$$G_2(z) = zH''(z)H(z)-2H'(z)(-H(z)+zH'(z)),$$

then $\frac{d^2\zeta_k^L(z)}{dz^2} = \frac{G_2(z)}{\lambda_{k1}H^3(z)}$. For simplicity, we focus on the scenario where $z > 0$ and L is large enough. It is easy to achieve $H(z) < 0$ for $z > 0$, then we consider the sign of $G_2(z)$. The analytical expression of $G_2(z)$ is as follows:

$$G_2(z) = -e^{zL}\big(BL^2z^2+(L^2+4BL)z+2B+2L + e^{zL}\big(B^2L^2z^3+2BL^2z^2+(L^2-2BL)z-2B-2L\big)\big).$$

If L satisfies $L \ge \frac{1+\sqrt{1+2Bz}}{z(Bz+1)}$, one can know quadratic function

$$\big(B^2L^2z^3+2BL^2z^2+(L^2-2BL)z-2B-2L\big)$$

of variable L is nonnegative. Combined with the fact that B, L, and z are all positive, it can be obtained that $G_2(z) < 0$ and

$$\frac{d^2\zeta_k^L(z)}{dz^2} > 0.$$

Thus, we have proved that $\zeta_k^L(z)$ is decreasing for $z \in \mathbb{R}$ and convex when $z > 0$ and $L \ge \frac{1+\sqrt{1+2Bz}}{z(Bz+1)}$. At the same time, z is a linear function of u_k with the slope $\lambda_{k2} > 0$, so the monotonicity and concavity-convexity of $\zeta_k^L(z)$ is equivalent with those of $\zeta_k^L(u_k)$. Finally, we arrive at the conclusion that $\zeta_k^L(u_k)$ is decreasing for $u_k \in \mathbb{R}$ and convex when $u_k > 0$ and $L \ge \frac{1+\sqrt{1+2Bz}}{z(Bz+1)}$. □

TABLE 2
Trace Summary

Trace	*Reality*	*Shanghai* Taxi
Network Type	Bluetooth	GPS
Device	Phone	GPS device
Number of devices	98	2, 000
Duration (days)	246	30

Lemma 2. *Based on the condition that $G_k(\zeta_k^L)$ is a nonincreasing and concave function, $F_k(u_k)$ is an increasing and concave function of u_k when $L \ge \frac{1+\sqrt{1+2Bz}}{z(Bz+1)}$.*

Proof. Calculate the second-order derivative of $F_k(u_k)$ according to the chain rule:

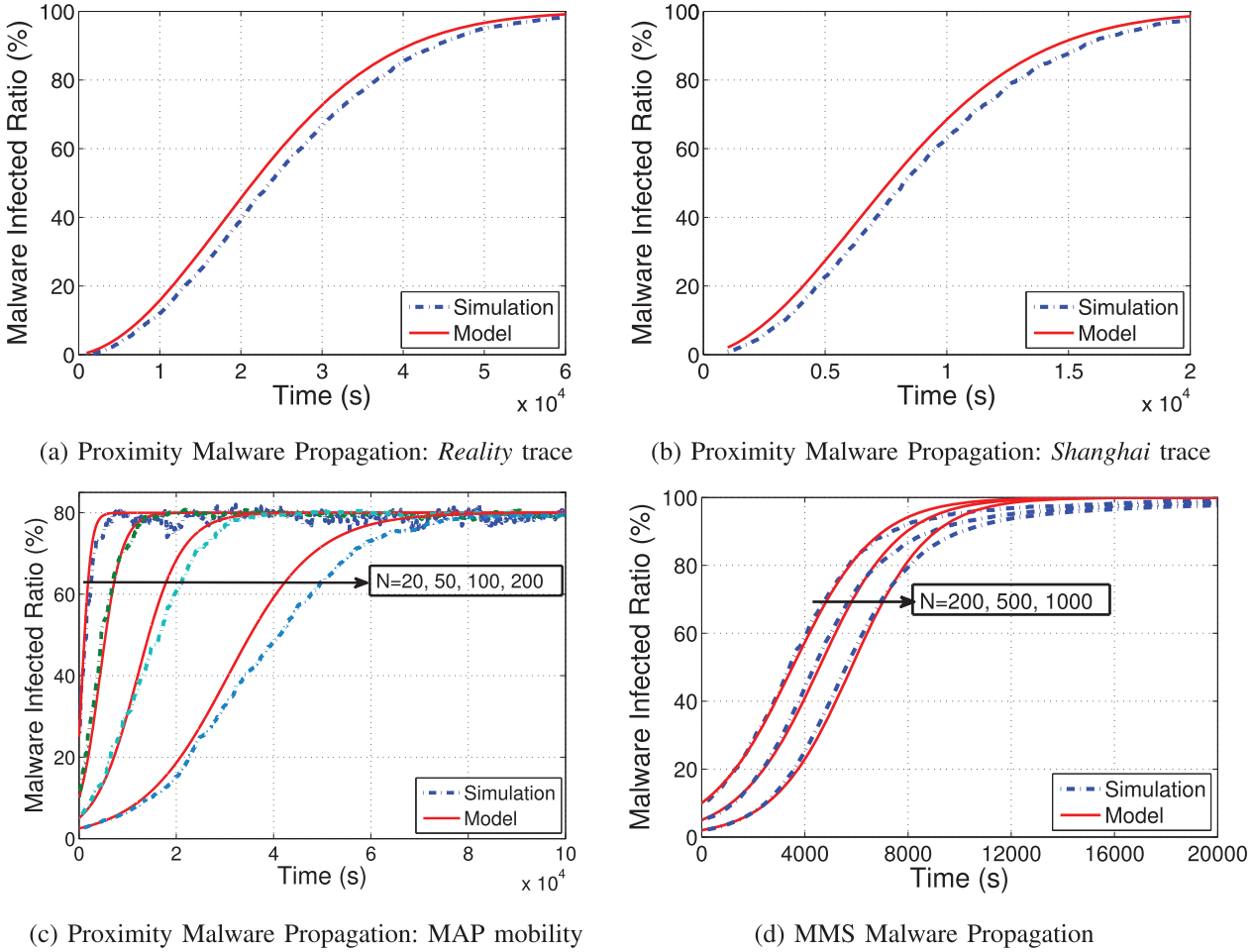
$$\frac{\mathrm{d}^2F_k}{\mathrm{d}u_k^2} = \frac{\mathrm{d}}{\mathrm{d}u_k}\left(\frac{\mathrm{d}G_k}{\mathrm{d}y}\cdot\frac{\mathrm{d}H_k}{\mathrm{d}u_k}\right) = \frac{\mathrm{d}^2G_k}{\mathrm{d}y^2}\left(\frac{\mathrm{d}H_k}{\mathrm{d}u_k}\right)^2 + \frac{\mathrm{d}G_k}{\mathrm{d}y}\frac{\mathrm{d}^2H_k}{\mathrm{d}u_k^2}. \quad (7)$$

Since $\zeta_k^L(u_k)$ is an increasing and strictly convex function of u_k when $L \ge \frac{1+\sqrt{1+2Bz}}{z(Bz+1)}$, and $G_k(\zeta_k^L)$ is a nonincreasing function, $G_k(\zeta_k^L(u_k))$ is an increasing and concave function of u_k, which proves the lemma. □

3.2 Model Validation

Now, we validate the proposed malware spreading model expressed in (1), which is based on the epidemic model for malware spreading and the fluid model in DTN. Since our model characterizes the fraction of the malware infected nodes, we simulate the malware spreading, and compare the simulation results of infected ratio with that obtained by the model. As we have claimed that this model characterizes the MMS and proximity malware spreading, we validate the malware spreading in both the proximity and MMS scenarios.

For proximity malware propagation, we use both realistic mobility trace and synthetic trace for simulations. Related to the realistic mobility, we use two traces. One is the human mobility contact trace from the *Reality* Mining Project of MIT [12], the other is the *Shanghai* taxi GPS trace [13]. The trace information is given by Table 2. From their features, we can observe that the used two traces cover a large diversity of DTN environments, from disperse university campus (*Reality*) to concentrated road site (*Shanghai*), with experiment period from 246 (*Reality*) to 30 days (*Shanghai*). We note that the number of nodes in these two traces is fixed. Consequently, they cannot validate the scalability of the proposed malware spreading model, which needs to change the number of nodes in the system. Therefore, we use another *MAP* trace in The ONE simulator [14], where the number of nodes in a system can be flexibly set. In the *MAP* trace, the nodes move along the streets on an imported map of downtown Helsinki, Finland. The size of the map is $4{,}500 \times 3{,}400\ \mathrm{m}^2$, and the nodes' transmission range is $20\ \mathrm{m}$. Among all nodes, about 15 percent of them



(a) Proximity Malware Propagation: *Reality* trace

(b) Proximity Malware Propagation: *Shanghai* trace

(c) Proximity Malware Propagation: MAP mobility

(d) MMS Malware Propagation

Fig. 1. Model validation of malware MMS and proximity spreading under different mobility trace of *MAP*, *Reality* human mobility trace, and *Shanghai* taxi trace.

are configured to follow predefined routes with the speed uniformly distributed in the range of $[1, 3]$ m/s, which imitates the regularity of certain type of mobility existing in human or vehicular, like public buses that have very regularity mobility trajectories and patterns every day. Other nodes are divided into three groups. For each group, there are points-of-interest (POI), and we assign a different probability for picking the next node from a particular group of POIs to simulate the phenomenon that people visit certain areas of a place more frequently than other areas based on features such as age and profession. The nodes' walking speed is chosen from a uniform distribution with the range of $[0.8, 1.2]$ m/s. Other settings are as default in The ONE simulator. We set the maximum number of nodes that malware can infect varying from 20 to 200.

For the above three introduced simulation scenarios, we set the number of nodes infected by the malware at the beginning of the simulation is 5 ($v_k^0 = 5$) and 10 percent nodes are helpers distributing the signature to prevent the malware propagation, i.e., u_k is set to 10 percent of the system nodes and v_k is set to 90 percent of the system nodes, respectively. Related to the malware spreading rate and killing rate related parameters of λ_{k1} and λ_{k2}, we use the exponential distribution to fit the contact records generated by the trace and use the obtained values of exponential parameter for the above three simulation scenarios. More specifically, we set λ_{k1} as the exponential parameter obtained from the contact records between helpers and general nodes, while set λ_{k2} as the parameter obtained from the general node pairs.

For MMS spreading malware, we use a realistic social network model to obtain the social graph to model the address books in the phone, which is exploited by the malware to propagate. The used social model is presented by Kumpula et al. [15]. In this model, the weights of the edges are generated dynamically and they shape the developing topology. By tuning a model parameter governing the importance of weights, the resulting networks undergo a gradual structural transition from a module free topology to one with communities. We set the average degree of nodes as 20, the number of nodes as 200, 500, and 1,000, and 10 percent nodes as infected at the beginning and no helpers. The other settings are the same as [5].

Figs. 1a and 1b show the simulation and theoretical results of malware infected ratio of proximity malware under the human trace of *Reality* and taxi trace of *Shanghai*, respectively. From these results, we can observe that the malware infected ratios obtained by the simulation (blue curves) and model (red curves) are very similar. Specifically, the average deviations between the theoretical calculations and the simulation results are only about 8.9 and 6.2 percent, respectively, for the *Reality* and *Shanghai* traces. To demonstrate the salability of our model, we show

the results under different number of system nodes in the *MAP* trace in Fig. 1c. We can observe that our model is accurate enough to capture the real system even when the number of nodes varies from 20 to 200. Similarly, Fig. 1d demonstrates the accuracy and scalability of our model to characterize the MMS malware spreading. In these validation, we use the general parameters to verify different kinds of simulation configurations to justify the effectiveness of the proposed model. Therefore, we come to the conclusion that the model used in the paper is accurate enough to characterize the malware propagation.

3.3 Problem Definition

Based on the defined utility function, we use the sum of individual utilities with different weighting factors w_k according to the final number of infected nodes as the system welfare. This is a standard and widely used definition. Consequently, we can specify the studied problem as the following optimization problem:

$$\begin{aligned} \text{maximize} \quad & \sum_{k\in\mathbb{K}} w_k F_k\left(\sum_{s\in\mathbb{S}} x_{s,k}\right) \\ \text{over} \quad & x_{s,k}\in\{0,1\}; \\ \text{subject to} \quad & \sum_{k\in\mathbb{K}} x_{s,k}\le A_s, \end{aligned} \tag{8}$$

where $x_{s,k}=1$ means helper s stores the signature of malware k; otherwise, $x_{s,k}=0$. w_k is the weighting factor, and $F_k(\sum_{s\in\mathbb{S}} x_{s,k})$ is the utility function for defending malware k defined in Section 3.1. Related to w_k, it is used to weigh the system contributions of different malware defending effects under different fairness objectives. One special case is that all malware has the same defending contribution to the system. Then, the utility is obtained by setting the weighting factors w_k with the same value. For example, by setting all $w_k=1$, we have the system welfare utility of $\sum_{k\in\mathbb{K}} F_k(\sum_{s\in\mathbb{S}} x_{s,k})$. Usually, malware that destroys the system more seriously will be assigned with a relatively higher weighting factor. In the formulated problem, we note that the system utility is an increasing and concave function of u_k, and the constraint is convex. Therefore, we can derive the optimal solution by gradient descent algorithm if $x_{s,k}$ is allowed to take real value. However, in the system, $x_{s,k}$ can either take 1 or 0. Therefore, we should design corresponding algorithms to solve this problem.

3.4 The Greedy Algorithm

Now, we give a greedy algorithm described in Algorithm 1 for the formulated problem. The obtained result by Algorithm 1 is the optimal solution, which is proved by Theorem 1. The algorithm repeatedly chooses signatures to store in the helpers: in each step, it tries to select one signature that brings the maximum system utility for a helper that still has enough storage. Therefore, our algorithm is likely to allocate more helpers to store the signatures of malware whose corresponding malware-defending utilities are larger than others, which is achieved by using the heterogeneous features in terms of devices and malware.

Algorithm 1. The Greedy Algorithm to Maximize the System Welfare

1: Set $x_{s,k}=0$, $u_k=0$, $\triangle F_k=0(k\in\mathbb{K}, s\in\mathbb{S})$;
2: Initialize *set* $\Re=\{1,2,\ldots,K\}$ and $\text{sum}=0$;
3: **for** Every malware k that $k\in\Re$ **do**
4: $\quad\triangle F_k\leftarrow w_k(F_k(u_k+1)-F_k(u_k))$;
5: **end for**
6: **while** $\text{sum}\le\sum_{s\in\mathbb{S}} A_s$ and $\Re\neq\emptyset$ **do**
7: $\quad$Select $i=\arg\max_k\{\triangle F_k|k\in\mathbb{K}\}$;
8: $\quad$Select $l=\arg\max_s\{A_s-\sum_{k\in\mathbb{R}} x_{s,k}|x_{s,i}=0,\ s\in\mathbb{S}\}$;
9: $\quad$Set $x_{l,i}=1$;
10: $\quad$Update $u_i\leftarrow u_i+1$, $\text{sum}\leftarrow\text{sum}+1$;
11: $\quad$Update $\triangle F_i\leftarrow w_i(F_i(u_i+1)-F_i(u_i))$;
12: $\quad$**if** $u_i\ge S$ **then**
13: $\quad\quad\Re\leftarrow\Re\backslash\{i\}$;
14: $\quad$**end if**
15: **end while**

Theorem 1. *The optimal solution of problem defined by (8) is obtained by Algorithm 1, whose computational complexity is* $O(K+\sum A_s\cdot\log K)$.

Proof. We first consider the case that Algorithm 1 is without line 8. In this case, there is no constraint that each node can store only one copy of the same signature. Based on this fact and the problem definition in (8), we can refine it as

$$\begin{aligned} \text{maximize} \quad & \sum_{k\in\mathbb{K}} w_k F_k(u_k) \\ \text{subject to} \quad & \sum_{k\in\mathbb{K}} u_k\le\sum_{s\in\mathbb{S}} A_s. \end{aligned} \tag{9}$$

Now, we prove Algorithm 1 without line 8 obtains the optimal solution of problem (9). Based on this formula, we define

$$\Delta_k(i)=w_k(F_k(i+1)-F_k(i)), i\in\mathbb{N}, k\in\mathbb{K}, \tag{10}$$

where $\mathbb{N}=\{0,1,2,S\}$. Sort $\Delta_k(i)$ for all $i\in\mathbb{N}, k\in\mathbb{K}$, and denote the result as

$$\widehat{\Delta_1}\ge\widehat{\Delta_2}\ge\widehat{\Delta_3}\ge\cdots\ge\widehat{\Delta_n}\ge\cdots,$$

where $\widehat{\Delta_1}$ is the maximum of all $\Delta_k(i)$, $\widehat{\Delta_2}$ is the second, and $\widehat{\Delta_j}$ is the jth largest one, respectively. At each step of the greedy algorithm to maximize the system welfare, we calculate

$$\Delta F_k=w_k(F_k(u_k+1)-F_k(u_k))=\Delta_k(u_k), k\in\mathbb{K}, \tag{11}$$

and choose $i=\text{argmax}_k\{\Delta F_k\mid k\in\mathbb{K}\}$, then update $u_i\leftarrow u_i+1$ and add ΔF_i to the objective function; let

$$\widehat{\delta_j}=\max\{\Delta F_k\mid k\in\mathbb{K}, \text{at the } j\text{th step of the algorithm}\}.$$

For simplicity, we denote the corresponding $\Delta_k(i)$ of $\widehat{\Delta_j}$ and $\widehat{\delta_j}$ as follows:

$$\widehat{\Delta_j}=\Delta_{K_1(j)}(U_1(j)), \tag{12}$$

$$\widehat{\delta_j}=\Delta_{K_2(j)}(u_{K_2(j)}), \tag{13}$$

where $K_1(j)$, $U_1(j)$, and $K_2(j)$ are the corresponding indexes. Based on the factor that $F_k(u_k)$ is a concave function of u_k, we have Lemma 3 as follows:

Lemma 3. *In our greedy algorithm, it can be obtained that $\widehat{\delta_j} = \widehat{\Delta_j}$, i.e., the increment of the objective function at the <jth> step is exactly the <jth> largest element among all $\Delta_k(i)$, in which $i \in \mathbb{N}, k \in \mathbb{K}$.*

Proof. First, we prove $\widehat{\delta_j} \geq \widehat{\delta_{j+1}}$. If $K_2(j) = K_2(j+1)$, it can be deduced that $u_{K_2(j+1)} = u_{K_2(j)} + 1$ because the update of $u_{K_2(j)}$ is performed at the <jth> step. From the concavity of $F_k(u_k)$, we know $F_k(u_k+1) - F_k(u_k) \geq F_k(u_k+2) - F_k(u_k+1)$, thus $\Delta_k(u_k) \geq \Delta_k(u_k+1)$; after substituting k and u_k with $K_2(j)$ and $u_{K_2(j)}$, respectively, it is finally achieved that $\widehat{\delta_j} \geq \widehat{\delta_{j+1}}$. If $K_2(j) \neq K_2(j+1)$, we know $u_{K_2(j+1)}$ remains the same between the <jth> and the <$(j+1)$th> step of the algorithm; thus, $\Delta F_{K_2(j+1)}$ is the same for the <jth> and the <$(j+1)$th> step. Since $K_2(j) = \text{argmax}_k\{\Delta F_k \mid k \in \mathbb{K}, \textit{at the } j\text{th } \textit{step}\}$, we know $\Delta F_{K_2(j)}$ is no less than $\Delta F_{K_2(j+1)}$ at the <jth> step. Consequently, we have

$$\widehat{\delta_j} = \Delta F_{K_2(j)} \geq \Delta F_{K_2(j+1)} = \widehat{\delta_{j+1}}. \tag{14}$$

Thus, it is obtained that

$$\widehat{\delta_1} \geq \widehat{\delta_2} \geq \widehat{\delta_3} \geq \cdots \geq \widehat{\delta_n} \geq \cdots$$

and we have $\widehat{\delta_j} \leq \widehat{\Delta_j}$ by the definition of $\widehat{\Delta_j}$.

Next, we obtain $\widehat{\delta_j} = \widehat{\Delta_j}$ by proof of contradiction. If it is not true, we can denote $j_0 = \min\{j \mid \widehat{\delta_j} \neq \widehat{\Delta_j}\}$. Since we have already proved that $\widehat{\delta_j} \leq \widehat{\Delta_j}$, it can be obtained that $\widehat{\delta_{j_0}} < \widehat{\Delta_{j_0}}$. Without loss of generality, we can assume that if $\widehat{\Delta_j} = \widehat{\Delta_{j+1}}$, the indexes will satisfy $K_1(j) < K_1(j+1)$ or $K_1(j) = K_1(j+1)$ and $U_1(j) < U_1(j+1)$. Furthermore, from (8) we have $\widehat{\Delta_{j_0}} = \Delta_{K_1(j_0)}(U_1(j_0))$. From our proof in the previous paragraph, it can be attained that $\Delta_{K_1(j_0)}(q) \geq \Delta_{K_1(j_0)}(U_1(j_0))$ for all $q < U_1(j_0)$. Since j_0 is the minimum and from the assumption of indexes above, we can know $\Delta_{K_1(j_0)}(q),\ 1 \leq q < U_1(j_0)$ all appear in $\widehat{\Delta_j},\ j < j_0$, thus they appear in $\widehat{\delta_j},\ j < j_0$. Consider the <$j_0$th> step of algorithm, it can be deduced that $u_{K_1(j_0)} = U_1(j_0)$ from above analysis, thus $K_1(j_0) = \text{argmax}_k\{\Delta F_k \mid k \in \mathbb{K}, \textit{ at the } j_0\text{th } \textit{step}\}$ because $\Delta F_{K_1(j_0)} = \widehat{\Delta_{j_0}}$ and $\widehat{\Delta_{j_0}} \geq \widehat{\Delta_{j'}}$ when $j' > j_0$. Consequently, from the algorithm we know $\widehat{\delta_{j_0}} = \widehat{\Delta_{j_0}}$, which leads to a contradiction. Thus, we finish the proof of Lemma 3. □

Now, we come back to the validity of our algorithm. From the definition of $\Delta_k(i)$, it can be derived that the objective function becomes

$$\sum_{k\in\mathbb{K}} w_k F_k(u_k) = \left(\sum_{k\in\mathbb{K}} \sum_{i=0}^{u_k-1} \Delta_k(i)\right) + \sum_{k\in\mathbb{K}} w_k F_k(0),$$
$$\text{subject to } \sum_{k\in\mathbb{K}} u_k \leq C,$$

where C is a constant. Since $\sum_{k\in\mathbb{K}} w_k F_k(0)$ is a constant and $(\sum_{k\in\mathbb{K}} \sum_{i=0}^{u_k-1} \Delta_k(i))$ is a sum of at most C terms of $\Delta_k(i)$, it can be deduced that

$$\left(\sum_{k\in\mathbb{K}} \sum_{i=0}^{u_k-1} \Delta_k(i)\right) \leq \sum_{j=1}^{C} \widehat{\Delta_j} = \sum_{j=1}^{C} \widehat{\delta_j}, \tag{15}$$

from the definition of $\widehat{\Delta_j}$ and Lemma 3. From (15), we have already proved that the objective function cannot exceed what we obtained from the greedy algorithm, Thus, we have proved that Algorithm 1 without Line 8 obtained the optimal solution of problem (9).

Now, we prove Algorithm 1 obtains the optimal solution of problem (8). Compared with problem (9), problem (8) has the constraints that each helper has its individual limitation of how many signatures can be stored and one cannot store the same signature twice. Since Line 8 of Algorithm 1 selects the node that has the maximum left storage size and has not stored the selected malware, it guarantees these constraints are satisfied. Therefore, in sorting $\Delta_k(i)$, $i \in \mathbb{N}$, we need to delete these $\Delta_k(i)$ that allows one helper to store the same signature twice. We denote the new sorting result as

$$\widehat{\Delta'_1} \geq \widehat{\Delta'_2} \geq \widehat{\Delta'_3} \geq \cdots \geq \widehat{\Delta'_n} \geq \ldots,$$

where $\widehat{\Delta'_j}$ is the jth largest one and all items are the available set that can be selected by the algorithm. Since at each step of Algorithm 1 we choose the items $i = \text{argmax}_k\{\Delta F_k \mid k \in \mathbb{K}\}$ from the available items, similar to the proof of the case without the constraints, we have proved Algorithm 1 obtain the optimal solution of problem (8).

Related to the computational complexity, we note that Lines 3-5 in the algorithm takes $O(k)$ time, and the next while loop takes $\sum A_s \cdot \log K$ time. Therefore, the computational complexity of the algorithm is $O(K + \sum A_s \cdot \log K)$. □

4 Distributed Algorithm Using Metropolis Sampler

Now, we design a distributed algorithm for the signature distribution problem. The designed algorithm is based on a simulated annealing technique called Metropolis sampler. In the following sections, we first describe the basic notions and the framework of Metropolis sampler (details are available in [8]), then design the distributed algorithm based on simulated annealing with the Metropolis sampler, and finally prove that the proposed algorithm converges to the optimal performance.

4.1 The Metropolis Sampler

Consider a Gibbs distribution, denoted by $\varpi(\cdot)$,

$$\varpi(x) = \frac{1}{Z}\exp\left(\frac{\xi(x)}{T}\right), \tag{16}$$

on the space $\mathbb{X}$. The basic notion introduced as follows comes from the physics. In (16), x is a value named configuration and $x \in \mathbb{X}$ is the set of all configurations, $T > 0$ is a system parameter named *temperature*, $\xi(\cdot)$ is the *energy* function associating a real number $\xi(x)$ with each configuration $x \in \mathbb{X}$, and Z is the normalizing constant. Since $\varpi(x)$ takes its value in $[0, 1]$, necessarily $-\infty \leq \xi(x) < +\infty$. This distribution has an important property that it favors configuration of larger energy, especially when temperature T is small [8]. More specifically, if we change the configuration x according to

certain probability, the distribution of configuration x will converge to the distribution of $\varpi(x)$ after a large number of iterations. Therefore, if we set $\xi(x)$ as our system welfare utility function defined in Section 3.3 as

$$\xi(x) = U(x) = \sum_{k \in \mathbb{IK}} w_k F_k(u_k),$$

where $u_k = \sum_{s \in \mathbb{S}} x_{s,k}$, $\varpi(x)$ is very much concentrated on the large values of $U(x)$. In other words, we are looking for any configuration of $x_0 \in \mathbf{X}$ such that

$$U(x_0) \geq U(x) \text{ for all } x \in \mathbf{X}.$$

Such a configuration is called as the global maximum of objective $U(x)$, and it depends on the global configuration set $\mathbf{X}$. We choose a configuration according to the current configuration x from a subset $\mathbf{Y} \subset \mathbf{X}$, called the neighborhood of x, and then proceed iteratively as follows: we examine a tentative configuration $x' \in \mathbf{Y}$ chosen according to a rule specific to the algorithm of the Metropolis sampler defined afterwards by comparing $U(x')$ and $U(x)$. If $U(x') > U(x)$, the tentative configuration is accepted as the new configuration, and the system transfers according to an irreducible transition matrix. It iterates and eventually produces a solution, which is locally maximized. Generally speaking, the solution is not optimal because the possible existence of local maximum is not necessary the global maximum. However, by the simulated annealing-based Metropolis sampler, the obtained local solution is globally optimal.

Now, we define the Metropolis sampler. Suppose the current configuration is x. At the step of n, a tentative configuration x' is selected according to the probability $\alpha_{x',x}(T_n)$, named replacement probability or acceptance probability, defined as

$$\alpha_{x',x}(T_n) = \min\left(1, \frac{\varpi(x')}{\varpi(x)}\right) = \min\left(1, e^{\left(\frac{U(x')-U(x)}{T_n}\right)}\right), \qquad (17)$$

where T_n is the system temperature; otherwise x' is rejected. At the same time, let $Q = q_{i,j}$ be an irreducible transition matrix on the system states of $\mathbf{X}$. At step n, the current system configuration is denoted by $X_n(T_n)$. Then, the process $\{X_n(T_n)\}_{n \geq 0}$ is a homogeneous Markov chain with state space $\mathbf{X}$ and transition matrix $\mathbf{P}(T_n)$ with each element $p_{i,j} = q_{i,j}\alpha_{i,j}(T_n)$.

4.2 Encounter-Based Distributed Algorithm

Based on the introduction of Gibbs distribution and Metropolis sampler, we now design the distributed algorithm for signature dissemination. We consider every encounter between any two helpers as one step of configuration changing in the algorithm. When nodes i and j meet, each one adjusts its current configuration according to the others. More specifically, one node, says i, randomly chooses a signature in its own buffer, and randomly chooses another one that is not in its buffer but in the buffer of node j to replace the chosen signature, which comes to a tentative configuration. After obtaining the replacement probability depending on the current and tentative configuration, node i decides whether to replace it or not. We assume the current configuration of the system is x, and the configuration of the node i is x_i, and node i chooses the signature c' from the buffer of node j to replace c. Consequently, the tentative new configuration of node i is $x'_i = x_i - 1_{i,c} + 1_{i,c'}$, where $1_{i,c}$ is a vector with the ith value equals 1 and others equal 0, and the system configuration changes to x'. According to these two configurations, we can derive the acceptance probability of the new configuration, as Lemma 4. If $U(x') \geq U(x)$, c' is accepted; whereas, a chance, which diminishes as its deviation from c, is left to the signature c'. This chance is necessary because the solution derived from the local information may deviate from the global optimization. Therefore, the intuition behind our distributed algorithm is that each helper keeps on selecting the signature that gives a higher contribution to the system utility according to the local information. When the nodes encounter again and again, the algorithm approaches to the global optimal solution.

Lemma 4. *The acceptance probability $\alpha_{c',c}$ of tentative configuration x', which replaces the signature c with c', is $\alpha_{c',c} = \min(1, \beta)$, where β is expressed as follows:*

$$\beta = \exp\left(\frac{w_c(\triangle F_c(u_c - 1) + w_{c'}(\triangle F_{c'}(u_{c'} + 1)))}{T_n}\right),$$

where $\triangle F_c(u_c - 1) = F_c(u_c) - F_c(u_c - 1)$, $\triangle F_{c'}(u_{c'} + 1) = F_c{}'(u_{c'}) - F_{c'}(u_{c'} + 1)$ and T_n is defined in Algorithm 2.

Proof. From (17), the definition of Metropolis sampler, we have

$$\beta = \exp\left(\frac{U(x') - U(x)}{T_n}\right).$$

Since $U(x) = \sum_{k \in \mathbb{K}} w_k F_k(u_k)$ and the difference between x' and x only lies in x_i and $x'_i = x_i - 1_{i,c} + 1_{i,c'}$, we have

$$U(x) = \sum_{k \in \mathbb{IK}, k \neq c, c'} w_k F_k(u_k) + w_c F_c(u_c) + w_{c'} F_{c'}(u_{c'}),$$

$$U(x') = \sum_{k \in \mathbb{IK}, k \neq c, c'} w_k F_k(u_k) + w_c F_c(u'_c) + w_{c'} F_{c'}(u'_{c'}),$$

where $u'_c = \sum_{s \in \mathbb{S}} x'_{s,c}$. At the same time, note that $u'_c = u_c - 1$ and $u'_{c'} = u_{c'} + 1$, we have

$$\begin{aligned} U(x') - U(x) = {} & w_c(F_c(u_c) - F_c(u_c - 1)) \\ & + w_{c'}(F_{c'}(u_{c'}) - F_{c'}(u_{c'} + 1)), \end{aligned}$$

which proves the Lemma. □

We note that Algorithm 2 requires nodes i and j to estimate system state related parameters of u_c and $u_{c'}$. In the centralized Algorithm 1, the system state is the number of nodes with signature k, $u(k)$, $k \in \mathbb{IK}$. However, in Algorithm 2, we only need to know the states of the two selected malware signatures c and c'. That is to say, in the distributed algorithm, each node only needs to maintain the state of the signatures that are currently buffered in its local storage, and there is no need to know the global system state. Since the required information is related to the carried signatures, we do not need to obtain this information from the whole network. In the algorithm, we exploit the exponentially weighted moving average (EWMA), which is one of the most effective schemes for online estimation and has been widely used in distributed

algorithms of DTN to maintain and update the global system states or parameters like the number of nodes, contact probability, and so on. EWMA is a simple and low-computation complex approach, which requires only constant storage space.

Algorithm 2. The distributed algorithm of malware signature distribution for Node i to adjust its configuration when encountering Node j, where T_0 is the initial temperature and n is the encounter counter that are set to be 1 at the beginning

1: **if** $x_{i,k} == x_{j,k}$ for all $k \in \mathbb{K}$ **then**
2: End the process;
3: **end if**
4: **if** $\exists k$: $x_{i,k} = 0$ and $x_{j,k} = 1$, which means there is at least one signature existing in node j, but does not exist in node i **then**
5: Set $n \leftarrow n + 1$
6: Select a signature c from the buffer of user i uniform randomly such that $x_{i,c} = 1$, and select a signature c' from the buffer of user j uniform randomly such that $x_{j,c'} = 1$ and $x_{i,c'} = 0$;
7: Set the system temperature $T_n = \frac{T_0}{\log(n-1)}$;
8: Compute the acceptance probability $\alpha_{c',c}(T_n)$;
9: Draw a random number R uniform distributed in $(0, 1]$;
10: **if** $R < \alpha_{c',c}(T_n)$ **then**
11: User i selects signature of c' and drops c with probability of $\frac{1}{SK}\alpha_{c',c}(T_n)$;
12: **end if**
13: **end if**

In the distributed system, each node, says i, maintains values of local u_k, where $k \in \mathbb{K}$ and $x_{i,k} = 1$, and updates through the exponential smoothing when two nodes meet with each other by exchanging local information through the contact. For example, when nodes i encounters node j and their local states are u_k^i and u_k^j for signature k. For all signatures that both nodes i and j carry, node i updates as $u_k^i \leftarrow \beta + (1-\beta)u_k^i$, and for all other signatures $u_k^i \leftarrow (1-\beta)u_k^i$, where β is the exponential decay rates. This mechanism is used widely, and its efficiency is verified by recent works of [16], [17], [18], [19]. It has been demonstrated in [16] that EWMA converges as long as the node mobility, and the convergence speed is exponential [19]. We note that the convergence speed of Algorithm 2 is geometric that will be introduced in the next section, which is much slower than the system state coverages. This ensures each node can obtain a relatively accurate system state to perform the distributed algorithm. At the same time, we will demonstrate the effectiveness of EWMA and the distributed algorithm by simulation.

4.3 Optimality and Convergence

In this section, we give a formal proof that the proposed distributed algorithm achieves the optimal system performance and its convergence is guaranteed. From the introduction of Metropolis sampler and distributed algorithm, we know that the system is evolved as a Markov chain $X_n(T_n)$. Therefore, we first obtain the stationary distribution of $X_n(T_n)$, which is given by the Lemma 5.

Lemma 5. *By iteration, the stationary distribution of the algorithm, denoted by $\pi(T_n)$, uniquely exists and it is given by*

$$\pi(T_n) = \frac{e^{U(x)/T_n}}{e^{\sum_{k\in\mathbb{K}} U(k)/T_n}}. \tag{18}$$

Proof. Since the state space of the Markov process $X(T)$ is finite, which equals to $S \times K$, it is positive recurrent. At the same time, it is irreducible. Therefore, it has a unique stationary distribution.

To verify its stationary distribution, $\pi(T) = \{\pi_i, i \in K\}$ can be expressed by (18), we need to prove the following three points: 1) $\pi_i \geq 0$; 2) $\sum_i \pi_i = 1$; and 3) $\pi(T) \cdot \mathbf{P} = \pi(T)$.

Points 1 and 2 are obvious. Now, we demonstrate point 3. Denote $\theta = \pi(T) \cdot \mathbf{P}$, for $\forall\, j$, we have

$$\gamma_j = \sum_i \pi_i p_{i,j} = \sum_i \pi_i \alpha_{i,j} q_{i,j}.$$

From Algorithm 2, we obtain $q_{i,j} = \frac{1}{SK}$. Substituting $\alpha_{i,j}$ in (17), we have

$$\gamma_j = \sum_{U(i)>U(j)} \frac{\pi_i}{SK} + \sum_{U(i)\leq U(j), i\neq j} \frac{\pi_i}{SK} e^{\frac{U(i)-U(j)}{T}} + \frac{\pi_j}{SK}\left(1 + \sum_{U(i)>U(j)} \left(1 - e^{\frac{U(i)-U(j)}{T}}\right)\right).$$

Since we can obtain

$$\pi_j \cdot e^{\frac{U(i)-U(j)}{T}} = \frac{e^{U(j)/T}}{e^{\sum_{k\in K} U(k)/T}} \cdot e^{\frac{U(j)-U(ji)}{T}} = \pi_i,$$

we have

$$\gamma_j = \sum_{U(i)\leq U(j), i\neq j} \frac{\pi_j}{SK} + \frac{\pi_j}{SK} + \sum_{U(i)>U(j)} \frac{\pi_j}{SK} = \pi_j.$$

Therefore, we have derived $\gamma = \pi$, leading to $\pi(T) \cdot \mathbf{P} = \pi(T)$, which proves the Lemma. □

Now, we give the property that the optimal solution is obtained when the system is in the stationary state.

Lemma 6. *When the system is in the stationary state, its distribution $\pi(T_n)$ concentrates on maximizing the system utility function $U(x)$ if $T_n \to 0$.*

Proof. First, we define the set of global maxima in which the configuration achieves the maximal system utility, as follows:

$$\Omega = \{i \in \mathbf{X};\ U(i) \geq U(j) \text{ for all } j \in \mathbf{X}\}.$$

We define $\omega = \max_{k\in\mathbf{X}} U(k)$. Dividing the numerator and denominator of the stationary distribution $\pi(T_n)$ by $\exp(\omega/T_n)$, we have

$$\pi(T_n) = \frac{\exp\left(\frac{-(\omega-U(x))}{T_n}\right)}{\sum_{k\in\mathbb{K}} \exp\left(\frac{-(\omega-U(k))}{T_n}\right)} = \frac{\exp\left(\frac{-(\omega-U(x))}{T_n}\right)}{|\Omega| + \sum_{k\notin\Omega} \exp\left(\frac{-(\omega-U(k))}{T_n}\right)}.$$

Then, we have

$$\lim_{T_n \to 0} \exp\left(\frac{-(\omega - U(x))}{T_n}\right) = \begin{cases} 1, & \text{if } x \in \Omega, \\ 0, & \text{if } x \notin \Omega. \end{cases}$$

Therefore, we have

$$\lim_{T_n \to 0} \pi(T_n) = \begin{cases} \frac{1}{|\Omega|}, & \text{if } x \in \Omega; \\ 0, & \text{if } x \notin \Omega. \end{cases}$$

In other words, when $T_n \to 0$, the stationary distribution lies in the global maximal set with probability $p = 1$. Therefore, the stationary system state focuses on maximizing the system utility function $U(x)$, which proves the Lemma. □

Now, we conclude that our algorithm obtains the global maximum solution when the system is in the stationary state. In the designed distributed algorithm, we use the iteration steps to obtain the solution. Therefore, we need to demonstrate that the system's stationary state can be approached by the iteration process. This is given in Theorem 2, which proves the system converges to the optimal solution.

Theorem 2. *The solution obtained by Algorithm 2 converges to the system optimal solution.*

Proof. We first prove that $\{X_n(T_n)\}_{n \geq 1}$ is ergodic. Recall that in our algorithm, the acceptance probability for the tentative configuration is

$$\alpha_{x',x}(T_n) = \begin{cases} \exp\left(\frac{U(x') - U(x)}{T_n}\right), & \text{if } U(x') < U(x); \\ 1, & \text{if } U(x') \geq U(x). \end{cases}$$

From the above expression, we can obtain that

$$\lim_{T_n \to 0} \alpha_{x',x}(T_n) \begin{cases} = 0, & \text{if } U(x') < U(x); \\ = 1, & \text{if } U(x') \geq U(x). \end{cases}$$

Thus, when $U(x') = U(x)$, we have $\lim_{T_n \to 0} \alpha_{x',x}(T_n) = 1 > 0$, which indicates the required conditions of [8, Theorem 8.1 of Chapter 7] are satisfied. Applying this theorem, we have $\underline{\alpha}_{x',x}(T_n)$, which is defined as $\underline{\alpha}_{x',x}(T_n) = \inf_{x \in \mathbb{X}, x' \in \mathbb{Y}} \alpha_{x',x}(T_n)$, can be expressed as

$$\underline{\alpha}_{x',x}(T_n) = \inf_{\substack{x \in \mathbb{Y} \\ U(x') > U(x)}} \exp\left(\frac{-(U(x) - U(x'))}{T_n}\right) \geq e^{\frac{-\Delta}{T_n}},$$

where $\Delta = \sup\{U(x) - U(x'),\ x' \in \mathbb{Y}\}$, which is a constant over time.

From Algorithm 2, we can obtain $T_n = \frac{T_0}{\log(n-1)}$ and can always set the initial temperature $T_0 \leq N\Delta$. Therefore, we have

$$\sum_{n=1}^{\infty} \left(\underline{\alpha}_{x',x}(T_{nN})\right)^N \geq \sum_{n=1}^{\infty} \left(\exp\left(\frac{-N\Delta}{T_0} \log(nN - 1)\right)\right)$$
$$\geq \sum_{n=1}^{\infty} \left(\exp\left(\frac{N\Delta}{T_0} \log \frac{1}{nN}\right)\right) \geq \sum_{n=1}^{\infty} \left(\frac{1}{nN}\right) = \infty.$$

From [8, Theorem 8.2 of Chapter 6], we obtain that $X_n(T_n)$ is strongly ergodic. At the same time, $X_n(T_n)$ is irreducible. Therefore, we obtain the limiting distribution of $X_n(T_n)$ is equal to its stationary distribution. According to Lemma 6, we have proved that our algorithm converges to the system's global maxima, which achieve the optimal performance. □

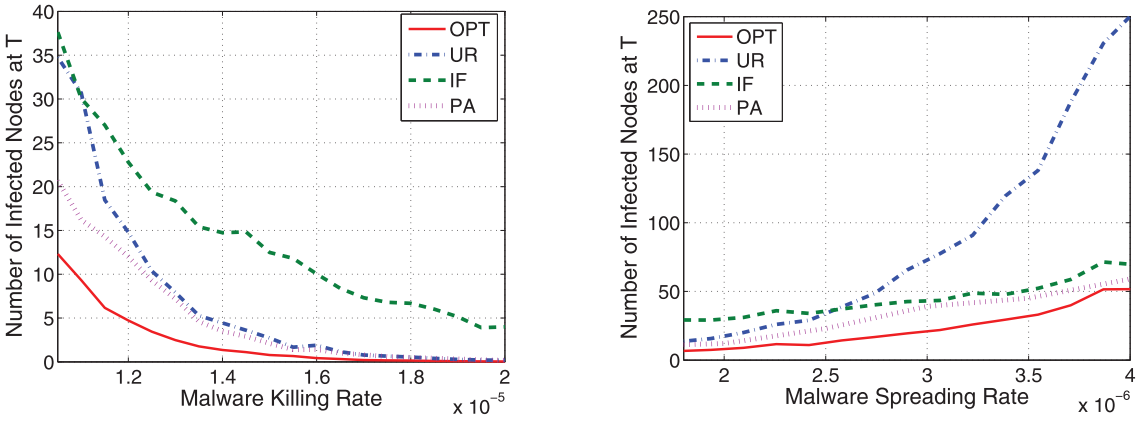
In Theorem 2, we have proved that the proposed algorithm converges to the optimal solution. Another important metric is the convergence speed. From the proof of Theorem 2, we can obtain that the convergence speed is related to how fast the algorithm converges to the stationary state because the optimal solution is obtained when the system approaches to the stationary state. From [8, Section 6.2 of Chapter 7], we obtain that the system converges to the stationary state with geometric rate. Therefore, the designed distributed Algorithm 2 approaches the optimal solution with geometric speed.

5 Performance Evaluation

5.1 Centralized Greedy Algorithm

In this section, we present numerical results with the goal of demonstrating that our greedy algorithm for the signature distribution, denoted OPT, achieves the optimal solution and yields significant enhancement on the system welfare compared with prior heuristic algorithms. Related to the heuristic algorithms, we consider 1) Important First (IF), which uses as many helpers as possible to store the signature of the most popular malware, 2) Uniform Random (UR), where each helper randomly selects the target signatures to store, and 3) Proportional Allocation (PA), which is a heuristic policy that assigns signatures with the uniform distribution proportional to the market sharing and the weights of different malware. To simulate a more realistic scenario, we model the malware in the system according to the market share of different handset OS of 2009. In the simulation, we change the malware killing rate and spreading rate, and consider a system with nodes that can be infected by five different types of malware, which are RIM targeted malware 36 percent; Android targeted 28 percent; iPhone 21 percent; Windows Mobile 10 percent, and others 5 percent. We set $N = 500$ and have 100 helpers with uniform random storage size from one to five signatures to deploy in the antimalware software. In the experimental setup, the number of initial infected nodes is set to be 10 percent of all nodes. Related to the utility function and weighting factors, we set $G_k(\zeta_k^L) = -\zeta_k^L$, $L = 2 \times 10^4$ s and $w = [1/2, 1/4, 1/8, 1/16, 1/16]$ to differentiate the system contributions of different malware defending effects by considering the factor that usually the malware spreading in the largest market sharing OS would result in the most serious damage.

The simulation results are shown in Fig. 2. Fig. 2a shows the number of infected nodes according to the malware recovering rates caused by the signature distribution in the centralized greedy algorithm. We can observe that the number of infected nodes decreases with the increase of recovering rate. Among different algorithms, IF provides the worst performance. Compared with other heuristic algorithms, our OPT algorithm reduces the number of infected nodes by 355.6, 127.3, and 56 percent over the FI, UR, and PA on average, respectively. Fig. 2b shows the number of infected nodes according to the malware spreading rates. Different from Fig. 2a, the number of infected nodes increases with the growth of spreading rate. From these results, we can observe that PA obtains relatively better performance than FI and UR, which are expected underperform. However, this well-organized heuristic policy still performs about 34 percent worse than



(a) (b)

Fig. 2. Performance of different algorithms for the malware defense system with (a) variable malware recovering rate; and (b) variable malware spreading rate.

our OPT algorithm on average. In conclusion, we obtain that OPT has a larger gain on reducing the number of infected nodes than other heuristic algorithms, which shows its efficiency.

5.2 Distributed Algorithm

In this section, we carry out simulations to evaluate the distributed algorithm by addressing the following goals: 1) demonstrating the distributed algorithm converges to the optimal system performance in different environment settings and large scale networks, 2) demonstrating our scheme of deploying the defense system achieves good performance of preventing the malware propagation under the real-world mobility traces. To achieve these two points, we cover a broad set of parameters as follows: 1) extensive mobility models of both real and synthetic traces, in which real traces include both human and taxi mobility traces, while synthetic traces include the Small World In Motion (SWIM) [20] and Self-similar Least Action Walk (SLAW) [21] models, where the number of system nodes can be changed in the simulation, 2) small and large system scale with respect to the number of nodes and kinds of malware, and 3) various compared schemes including our centralized optimal greedy algorithm, UR and IF. According to the malware propagation, we use the opportunistic contacts between nodes to spread the proximity malware, while using the phone books generated by the social model introduced in Section 3.2 to spread the malware via MMS. More specifically, the infected nodes will transmit the malware to the nodes in its phone book one by one, and the time interval and malware transmission and receiving time are set as exponential distribution. But if they encounter other nodes in proximity, these nodes will be infected immediately. In the simulation of the distributed algorithm, instead of simply assuming each node has the exact global knowledge, we run the proposed EWMA method to estimate the global information of the number of nodes and system states in the network. That is to say, the global information is also obtained by the distributed approach in the simulation, which sets up an objective environment to evaluate our proposed distributed algorithm.

5.2.1 Mobility Model Simulation

We now simulate the greedy algorithm and distributed algorithm under the mobility models of SWIM [20] and SLAW [21], which are two of the most realistic synthetic mobility models existing. For the SWIM model, we use the typical settings in [20] that the transmission range of the nodes is $50\ \mathrm{m}$, coefficient $\alpha = 0.95$, and use a power law with slope $a = 1.45$ to generate the waiting time values of nodes when arriving to destination. For the SLAW model, we use the typical settings in [21], where the speed of every user is set to $1\ \mathrm{m/s}$, the transmission range of each node is set to $50\ \mathrm{m}$, and use a truncated Pareto distribution to generate the pause time of which the minimum and maximum values are 30 seconds and 700 minutes, respectively. For both of these two mobility models, we set the network area as $1{,}000 \times 1{,}000\ \mathrm{m}^2$, the number of nodes in the network as 500, and the malware distribution follows the market sharing mentioned above but merges the smallest into the other type. In the simulation, v_0 is set to 50 percent of all nodes, and 10 percent nodes are set as the helpers with uniform random storage for one to four signatures. Since we have proved our greedy algorithm gives the optimal system performance, we use it to compare with the distributed algorithm.

First, we show the deviation of the number of helpers for each kind of malware (u_1 to u_4) between the greedy algorithm and distributed algorithm in Fig. 3. From the results, we can see that with the increase of the time, the deviation converges to 0 in all cases. This demonstrates the convergence of our distribution algorithm to the optimal signature distribution. Among them, the coverage rate of u_4 is the smallest one, which is caused by relatively less helpers to defense this malware because the number of nodes can be infected in the system is smallest caused by the market sharing setting. Besides the same convergence properties observed in both SWIM and SLAW models, we can find the convergence rate of SLAW is a little slower than that of SWIM, which is mainly caused by the different encounter properties in these two mobility models. Second, we measure the malware infected ratio of nodes against time, and the obtained results under the SWIM and SLAW mobility models are shown in Figs. 4a and 4b, respectively.

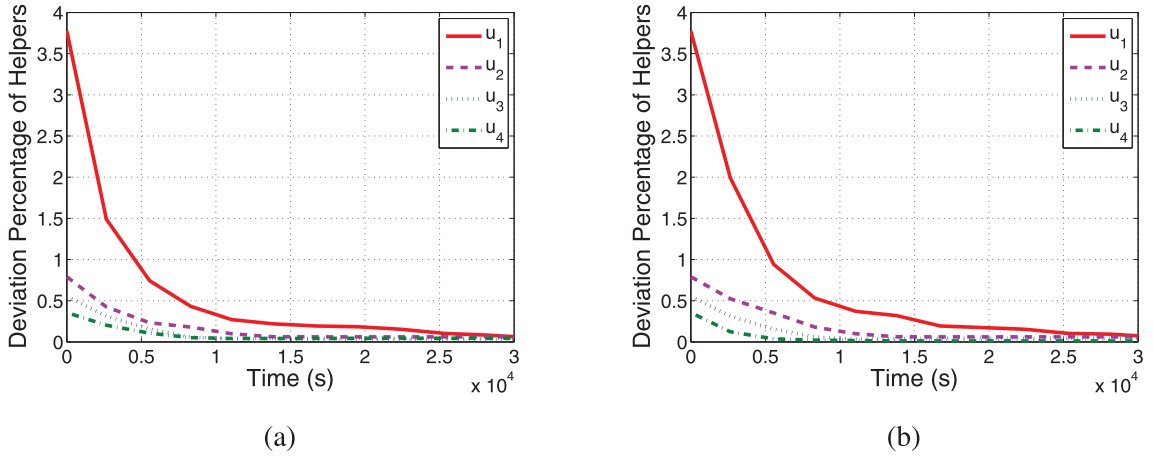


Fig. 3. Convergence of the helper under different mobility models of (a) SWIM, and (b) SLAW.

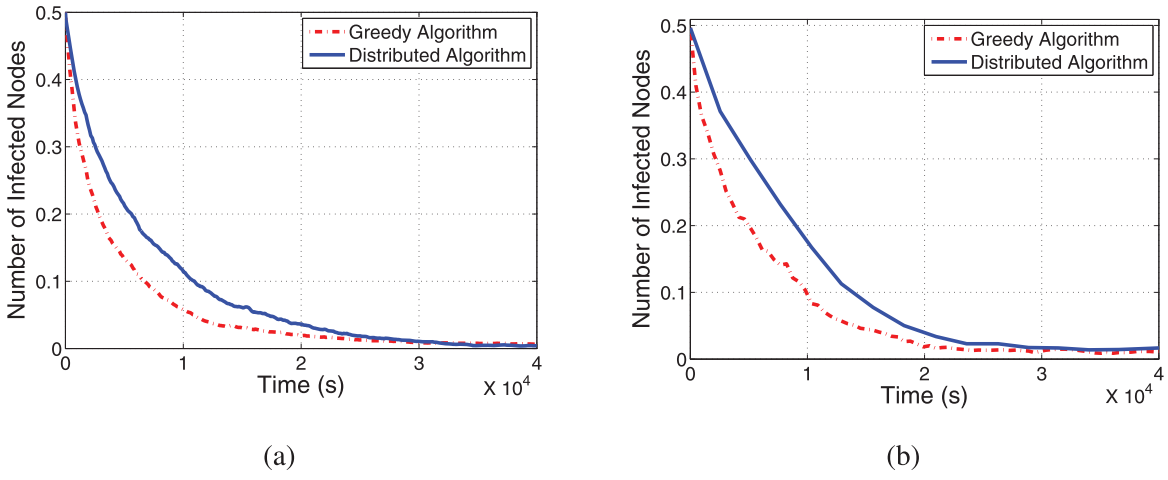


Fig. 4. System performance of malware infected ratio under different mobility models of (a) SWIM, and (b) SLAW.

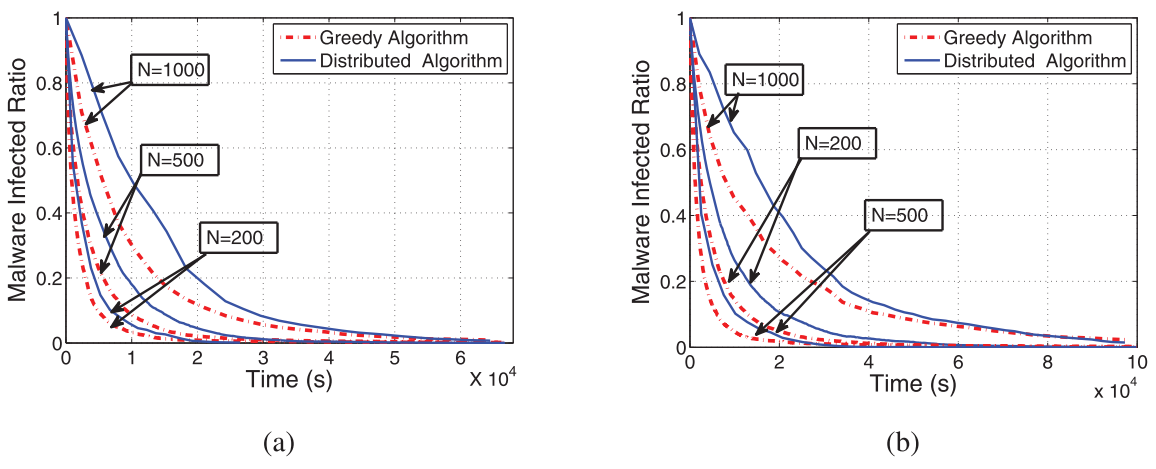


Fig. 5. System performance of malware infected ratio with different network scales under different mobility models of (a) SWIM, and (b) SLAW.

From the results, we can observe that the greedy algorithm performs better than the distributed algorithm when the time is short. But the distributed algorithm approaches the performance of the greedy algorithm with the increase of the time. When the time is longer enough, these two schemes have the same performance. Therefore, we can conclude that our distribution algorithm approaches the optimal system performance.

To show the scalability of our scheme, we set the number of the network nodes to 200, 500, and 1,000, and the corresponding malware types are 10, 15, and 20. We also study the performance of the greedy and distributed algorithms, and the results are shown in Fig. 5. From the results, we can observe that with the increase of the network size, the time needed by the distributed algorithm to converge to the optimal system performance becomes larger. However, under both the SWIM and SLAW model, the distributed algorithm always approaches the optimal system performance provided by the greedy algorithm.

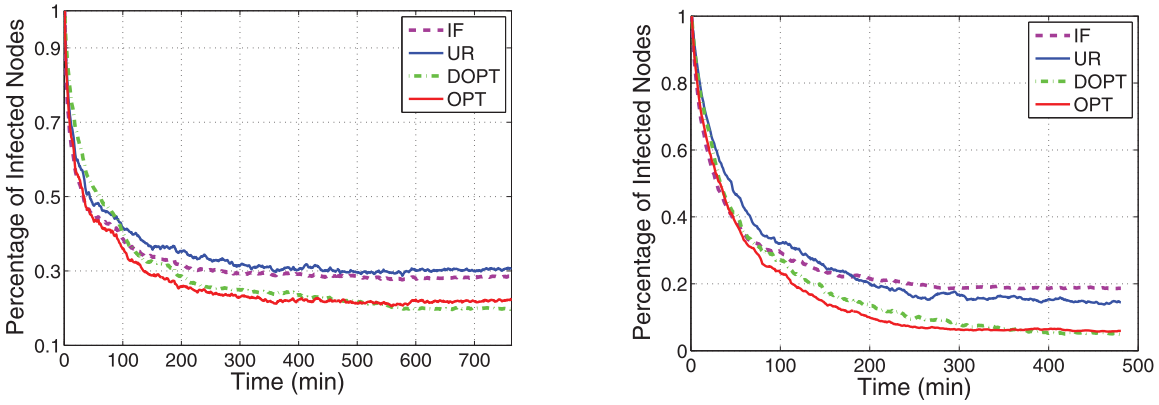


Fig. 6. System performance of malware infected ratio with real trace of (a) *Reality* human trace, and (b) *Shanghai* vehicle trace.

Therefore, it is demonstrated that our distributed scheme is scalable in both small and large networks.

5.2.2 Real Contact Traces

To show the efficiency of our scheme in real-mobility environments, we now use the real traces for simulation, which are the human mobility of *Reality* trace and taxi mobility of *Shanghai* trace introduced in Section 3.2. For the simulated algorithm, we compare our distributed algorithm, denoted DOPT, with OPT, UR, and IF. Similar to the evaluation of the centralized algorithm, we model the malware types according to the market share of different handset OS of 2009, set $G_k(\zeta_k^L) = -\zeta_k^L$, and $w = [1/2, 1/4, 1/8, 1/16, 1/16]$. For the node types, we set all nodes infected at first, and use 15 percent nodes as helpers to distribute the signatures. The results of malware infected ratio according to the time are shown in Fig. 6. From the results, we can see that IF and UR perform worse compared with the greedy algorithm and distributed algorithm DOPT. Comparing OPT and DOPT, we can observe DOPT becomes closer to the optimal system performance provided by OPT with the increase of time. Therefore, we can conclude that the proposed scheme for the signature distribution achieves good performance of preventing malware propagation under the real-world environments.

To reveal how the performance of malware defense system is influenced by the heterogeneous system resources, we study the scenario of changing the malware signature carrying capacity of the helpers, which depends on their buffer size. We use *Reality* and *Shanghai* traces, set the number of malware types to be 25 with the same weight values of $1/25$, change the average number of malware carried by the helpers, and obtain the system performance of malware infected ratio by fixing the final system time $L = 400$ min. The obtained results are shown in Fig. 7, where the distributed algorithm shown by the blue curves performs almost the same as the optimal centralized algorithm. From the results, we can observe that with the increase of the number of carried signatures, the system performance of malware infected ratio decreases significantly, which is from about 65 to 11 percent for *Reality* trace, and from about 53 to 6 percent for *Shanghai* trace. Especially, when the number of carried signatures changes from 5 to 15, the decreasing rate of malware infected ratio is relatively larger. These results reveal that the system performance of malware defense is influenced by the system resource of storage significantly.

6 Related Work

With the growth of SMS/MMS, mobile games, mobile commerce, and mobile peer-to-peer file sharing, a number of studies have demonstrated the threat of malware propagation on mobile phones. They can be generally categorized into two main types. One class of works focuses on analyzing the proximity malware spreading. Yan et al. [22], [23] develop a simulation and analytic model for Bluetooth worms, and show that mobility has a significant impact on the propagation dynamics. The other class focuses on the malware spreading by SMS/MMS. Fleizach et al. [24] evaluate the speed and severity of malware spreading by cell phone address books. Zhu et al. [5] studied the characteristics of slow start and exponential propagation exhibited by MMS malware. Besides, a small amount of works also look at both MMS and proximity malware. For example, Bose and Shin [25] investigate the

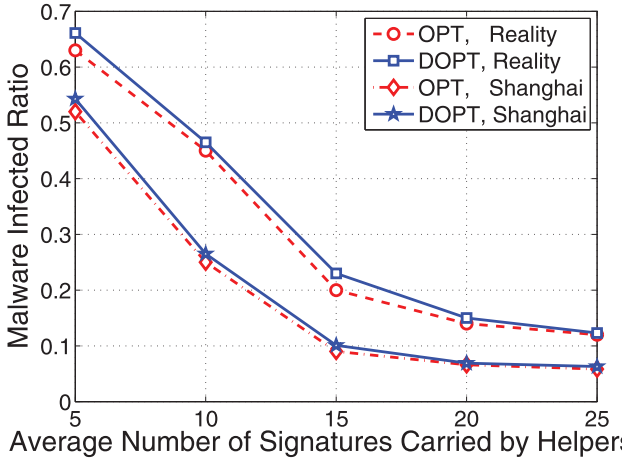


Fig. 7. System performance of malware infected ratio with *Reality* human mobility trace and *Shanghai* vehicle trace when changing the number of signatures carried by the helpers.

propagation of mobile worms and viruses using data from a real-life SMS customer network, and they reveal that hybrid worms using both MMS and proximity scanning can spread rapidly within cellular networks. Wang et al. [1] model the mobility of mobile phone users by analyzing a trace of 6.2 million mobile subscribers from a service provider. They study the fundamental spreading patterns that characterize a mobile virus outbreak and find that the greatest danger is posed by hybrid viruses that take advantage of both proximity and MMS. Obtaining the insights of these two works, our model considers both the MMS and proximity propagation in our defense system design.

For performance evaluation and modeling of mobile malware spreading, the epidemic model, based on the classical Kermack-Mckendrick model [26] traditionally used in wired networks, has been extensively used in [6], [27], [4], [1], and so on. Actually, the system performance of the epidemic model can be approximated by the Ordinary Differential Equations with a well-known technique called fluid model [11], which is widely used to model the epidemic forwarding in DTN [11], [17], [28]. In the fluid model, the solution of the ODE converges in probability to the system's sample paths. These works show that when the number of nodes in a network is large, the deterministic epidemic models can successfully represent the dynamics of malware spreading, which is demonstrated by simulations and matching with actual data. We use an ODE model to analyze and design the signature distribution problem in the malware defense system. Therefore, our model in this work is reasonable.

Recently, some malware coping schemes have been proposed to defend mobile devices against malware propagation. To prevent the malware spreading by MMS/SMS, Zhu et al. [5] propose a counter-mechanism to stop the propagation of a mobile worm by patching an optimal set of selected phones by extracting a social relationship graph between mobile phones via an analysis of the network traffic and contact books. This approach only targets the MMS spreading malware and has to be centrally implemented and deployed in the service provider's network. To defend mobile networks from proximity malware by Bluetooth, Zyba et al. [6] explore three strategies, including local detection, proximity signature dissemination, and broadcast signature dissemination. For detecting and mitigating proximity malware, Li et al. [7] propose a community-based proximity malware coping scheme by utilizing the social community structure reflecting a stable and controllable granularity of security. These two works both target the proximity malware. The former one has the limitations that signature flooding costs too much and the local view of each node constrains the global optimal solution. Although the aftermath scheme integrates short-term coping components to deal with individual malware and long-term evaluation components to offer vulnerability evaluation toward individual nodes, the social community information still need to be obtained in a centralized way. Khouzani et al. [29] investigate the optimal dissemination of security patches in mobile wireless network to counter the proximity malware threat by contact. It uses the SIR model to formulate the tradeoffs between the security risks and resource consumption as optimal control problems under the assumptions of homogeneous network setting. There are significant differences between these works and our work. First, our scheme targets both the MMS and proximity malware at the same time, and considers the problem of signature distribution. Second, all these works assume that malware and devices are homogeneous, we take the heterogeneity of devices into account in deploying the system and consider the system resource limitations. Third, our proposed algorithm is distributed, and approaches to the optimal system solution.

7 Conclusions

In this paper, we investigate the problem of optimal signature distribution to defend mobile networks against the propagation of both proximity and MMS-based malware. We introduce a distributed algorithm that closely approaches the optimal system performance of a centralized solution. Through both theoretical analysis and simulations, we demonstrate the efficiency of our defense scheme in reducing the amount of infected nodes in the system.

At the same time, a number of open questions remain unanswered. For example, the malicious nodes may inject some dummy signatures targeting no malware into the network and induce denial-of-service attacks to the defense system. Therefore, security and authentication mechanisms should be considered. From the aspect of malware, since some sophisticated malware that can bypass the signature detection would emerge with the development of the defense system, new defense mechanisms will be required. At the same time, our work considers the case of OS-targeting malware. Although most of the current existing malware is OS targeted, cross-OS malware will emerge and propagate in the near future. How to efficiently deploy the defense system with the consideration of cross-OS malware is another important problem. We are continuing to cover these topics in the future work.

Acknowledgments

This work was supported by the National Basic Research Program of China (973 Program) (No. 2013CB329105), the National Nature Science Foundation of China (No. 61301080, No. 61171065, and No. 61273214), the National High Technology Research and Development Program (No. 2013AA013501 and No. 2013AA013505), the Chinese National Major Scientific and Technological Specialized Project (No. 2013ZX03002001), and China's Next Generation Internet (No. CNGI-12-03-007). The authors would also like to acknowledge the valuable comments from Prof. P.R. Kumar, University of Illinois at Urbana-Champaign, and Prof. Wei Chen, Tsinghua University. Part of this work was presented at IEEE SECON 2011 (Salt Lake City, Utah), June 2011.

References

[1] P. Wang, M. Gonzalez, C. Hidalgo, and A. Barabasi, "Understanding the Spreading Patterns of Mobile Phone Viruses," *Science*, vol. 324, no. 5930, pp. 1071-1076, 2009.

[2] M. Hypponen, "Mobile Malwar," *Proc. 16th USENIX Security Symp.*, 2007.

[3] G. Lawton, "On the Trail of the Conficker Worm," *Computer*, vol. 42, no. 6, pp. 19-22, June 2009.

[4] M. Khouzani, S. Sarkar, and E. Altman, "Maximum Damage Malware Attack in Mobile Wireless Networks," *Proc. IEEE INFOCOM*, 2010.

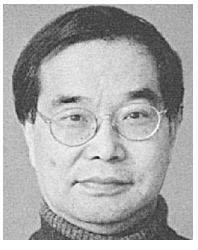
[5] Z. Zhu, G. Cao, S. Zhu, S. Ranjan, and A. Nucci, "A Social Network Based Patching Scheme for Worm Containment in Cellular Networks," *Proc. IEEE INFOCOM,* 2009.
[6] G. Zyba, G. Voelker, M. Liljenstam, A. Méhes, and P. Johansson, "Defending Mobile Phones from Proximity Malware," *Proc. IEEE INFOCOM,* 2009.
[7] F. Li, Y. Yang, and J. Wu, "CPMC: An Efficient Proximity Malware Coping Scheme in Smartphone-Based Mobile Networks," *Proc. IEEE INFOCOM,* 2009.
[8] P. Brémaud, *Markov Chains: Gibbs Fields, Monte Carlo Simulation, and Queues.* Springer Verlag, 1999.
[9] M. Grossglauser and D. Tse, "Mobility Increases The Capacity of Ad-Hoc Wireless Networks," *Proc. IEEE INFOCOM,* pp. 1360-1369, 2001.
[10] R. May and A. Lloyd, "Infection Dynamics on Scale-Free Networks," *Physical Rev. E,* vol. 64, no. 6, p. 066112, 2001.
[11] E. Altman, G. Neglia, F. De Pellegrini, and D. Miorandi, "Decentralized Stochastic Control of Delay Tolerant Networks," *Proc. IEEE INFOCOM,* 2009.
[12] P. Hui, J. Crowcroft, and E. Yoneki, "Bubble Rap: Social-Based Forwarding in Delay Tolerant Networks," *Proc. ACM MobiHoc,* 2008.
[13] Shanghai Jiao Tong Univ., Traffic Information Grid Team, Grid Computing Center, "Shanghai Taxi Trace Data," http://wireless-slab.sjtu.edu.cn/, 2013.
[14] A. Keränen, J. Ott, and T. Kärkkäinen, "The ONE Simulator for DTN Protocol Evaluation," *Proc. Second Int'l Conf. Simulation Tools and Techniques,* pp. 1-10, 2009.
[15] J. Kumpula, J. Onnela, J. Saramäki, K. Kaski, and J. Kertész, "Emergence of Communities in Weighted Networks," *Physical Rev. Letters,* vol. 99, no. 22, p. 228701, 2007.
[16] S. Ioannidis, L. Massoulié, and A. Chaintreau, "Distributed Caching over Heterogeneous Mobile Networks," *Proc. ACM Int'l Conf. Measurement and Modeling of Computer Systems (SIGMETRICS '10),* 2010.
[17] L. Hu, J. Boudec, and M. Vojnovic, "Optimal Channel Choice for Collaborative Ad-Hoc Dissemination," *Proc. IEEE INFOCOM,* 2010.
[18] T. Ning, Z. Yang, and H. Wu, "Counting in Delay-Tolerant Mobile Networks," *Proc. IEEE Int'l Conf. Comm. (ICC),* pp. 1-5, 2010.
[19] Y. Wang and H. Wu, "DFT-MSN: The Delay Fault Tolerant Mobile Sensor Network for Pervasive Information Gathering," *Proc. IEEE INFOCOM,* 2006.
[20] A. Mei and J. Stefa, "SWIM: A Simple Model to Generate Small Mobile Worlds," *Proc. IEEE INFOCOM,* pp. 2106-2113, 2009.
[21] K. Lee, S. Hong, S. Kim, I. Rhee, and S. Chong, "SLAW: A New Mobility Model for Human Walks," *Proc. IEEE INFOCOM,* pp. 855-863, 2009.
[22] G. Yan and S. Eidenbenz, "Modeling Propagation Dynamics of Bluetooth Worms," *IEEE Trans. Mobile Computing,* vol. 8, no. 3, p. 1071, 2008.
[23] G. Yan, H. Flores, L. Cuellar, N. Hengartner, S. Eidenbenz, and V. Vu, "Bluetooth Worm Propagation: Mobility Pattern Matters!" *Proc. ACM Symp. Information, Computer and Comm. Security,* pp. 32-44, 2007.
[24] C. Fleizach, M. Liljenstam, P. Johansson, G. Voelker, and A. Mehes, "Can You Infect Me Now? Malware Propagation in Mobile Phone Networks," *Proc. ACM Workshop Recurring Malcode,* pp. 61-68, 2007.
[25] A. Bose and K. Shin, "On Mobile Viruses Exploiting Messaging and Bluetooth Services," *Proc. Securecomm and Workshops,* pp. 1-10, 2006.
[26] D. Daley and J. Gani, *Epidemic Modelling: An Introduction.* Cambridge Univ, 2001.
[27] J. Mickens and B. Noble, "Modeling Epidemic Spreading in Mobile Environments," *Proc. Fourth ACM Workshop Wireless Security,* pp. 77-86, 2005.
[28] E. Altman, A.P. Azad, and F. De Pellegrini, "Optimal Activation and Transmission Control in Delay Tolerant Networks," *Proc. IEEE INFOCOM,* 2010.
[29] M. Khouzani, S. Sarkar, and E. Altman, "Dispatch then Stop: Optimal Dissemination of Security Patches in Mobile Wireless Networks," *Proc. IEEE 49th Conf. Decision and Control (CDC),* pp. 2354-2359, 2010.

Yong Li received the BS degree from the Department of Electronics and Information Engineering at Huazhong University of Science and Technology, Wuhan, China, in 2007. He is currently working toward the PhD degree in Tsinghua University. He serves as a paper reviewer for international conferences of IEEE ICC, VTC, ICOIN, PIMRC, and APCC. His research interests include mobile delay-tolerant networks, topics including forwarding polices design, buffer management design and performance evaluation, mobility modeling, mobility management in next generation wireless IP networks, topics including mobile IP, SIP, proxy mobile IP, cross-layer design, multicast mobility, modeling for mobility performance evaluation, enhancing handoff performance, and proposing mobility management architecture. He is a member of the IEEE.

Pan Hui received the BEng and MPhil degrees both from the University of Hong Kong and the PhD degree from the Computer Laboratory, University of Cambridge. He is currently a faculty member in the Department of Computer Science and Engineering at the Hong Kong University of Science and Technology, where he directs the HKUST-DT System and Media Lab. He also serves as a distinguished scientist at Telekom Innovation Laboratories (T-labs), Germany, and as an adjunct professor of social computing and networking at Aalto University, Finland. He has published more than 100 research papers and has several granted and pending European patents. He is a member of the IEEE.

Depeng Jin received the BS and PhD degrees from Tsinghua University, Beijing, China, in 1995 and 1999, respectively, both in electronics engineering. He is currently an associate professor at Tsinghua University and the vice chair of the Department of Electronic Engineering. He received the National Scientific and Technological Innovation Prize (Second Class) in 2002. His research interests include telecommunications, high-speed networks, ASIC design, and future internet architecture. He is a member of the IEEE.

Li Su received the BS degree from Nankai University, Tianjin, China, in 1999 and the PhD degree from Tsinghua University, Beijing, China, in 2007, respectively, both in electronics engineering. He is currently a research associate in the Department of Electronic Engineering, Tsinghua University. His research interests include telecommunications, future Internet architecture, and on-chip network.



Lieguang Zeng received the BS degree from Tsinghua University, Beijing, China, in 1970. Since 1970, he has been with the Department of Electronic Engineering, Tsinghua University, Beijing, China, where he is currently a professor. He has received the National Scientific and Technological Innovation Prize (Second Class) three times, in 1987, 1991, and 2002, respectively. He has also received provincial and ministerial level science and technology progress prizes eight times. He has published more than 150 research papers and obtained four patents. His research interests include telecommunications, high-speed networks, ASIC design, and future Internet architecture. He is a member of the IEEE.