Hindawi
Complexity
Volume 2019, Article ID 9057531, 19 pages
https://doi.org/10.1155/2019/9057531

Research Article

CC^2: Defending Hybrid Worm on Mobile Networks with Two-Dimensional Circulation Control

Hailu Yang,[1,2] **Deyun Chen**,[1,2] **Guanglu Sun**,[2] **Xiaoyu Ding**,[3] **and Yu Xin**[4]

[1]*Postdoctoral Research Station of Computer Science and Technology, Harbin University of Science and Technology, Harbin 150080, China*
[2]*School of Computer Science and Technology, Harbin University of Science and Technology, Harbin 150080, China*
[3]*College of Computer Science and Technology, Harbin Engineering University, Harbin 150080, China*
[4]*Faculty of Electrical Engineering and Computer Science, Ningbo University, Ningbo 315211, China*

Correspondence should be addressed to Deyun Chen; chendeyun@hrbust.edu.cn

Received 14 August 2019; Accepted 30 November 2019; Published 14 December 2019

Guest Editor: Raúl Baños

As the hybrid worm can propagate by both personal social interactions and wireless communications, it has been identified as one of the most severe threats to the mobile Internet. This problem is expected to become worse with the boom of social applications and mobile services. In this work, we study the propagation dynamics of hybrid worms and propose a systematic countermeasure. The system maintains a set of community structure which describes the high-speed infection zone of worms and contains worm propagation by distributing the worm signature to the guard nodes selected from the periphery of each community. For those nodes that are geographically close but located in different communities , we evaluate the communication security between them based on the observed infection history and limit communications between insecure ones to avoid the worm spreading across communities. We also design an efficient worm signature forwarding strategy that enables most nodes in the network to reach an immune state before being infected by the worm. Extensive real-trace driven simulations verify the feasibility and effectiveness of the proposed methods.

1. Introduction

With the rapid deployment of mobile Internet technology, smartphone-based social services such as Facebook, Wechat, and LinkedIn have already reached billions of registered users, many of whom choose to incorporate those services into their work and family life. On the positive side, the mobile Internet provides a convenient platform for people to communicate with close friends and interact online. On the negative side, however, it is also a breeding place for the spread of the mobile Internet worm [1].

The propagation of the worm in mobile Internet mainly depends on two dominant patterns [2]. First, the short-range worm infects all Bluetooth or Wi-Fi opened devices within the infection radius, which exhibits a spatial propagation pattern similar to the case of the contact-based disease [3]. Such kinds of infections rely on peer-to-peer communications between sensors with geographical adjacency, which build a geographic interaction networks (short for GINs). The GINs worm always exploits hardware vulnerabilities of the mobile device to crash them. Defending against this type of infection is a challenge due to the lack of effective centralized regulation. On this account, most of the existing methods utilize a distributed coping scheme that allows users to limit the communications with vulnerable devices to insulate the proximity worm [4, 5]. Second, a long-range worm can replicate itself and infect all smartphones whose identifiers are stored in the infected smartphone's contact list, a delocalized propagation mode based on social relations in the social information networks (short for SINs). The SINs worm is similar to the one observed in Multimedia Messaging Service (MMS), both of which exhibit the characteristics of slow start

and exponential propagation [6]. Recent works mostly utilize a partitioning strategy to insulate the SINs worm in several disjoint "islands" [7–9].

Recent research reveals that as mobile phone functions continue to increase, the worm no longer uses a single model to spread [2]. The hybrid worm uses short-range infections as well as long-range ones. The first variant that utilizes the hybrid propagation mechanism is Commwarrior, which spreads from one phone to another via the Bluetooth interface and MMS. Since the message usually comes from friends or family members, Commwarrior has a high probability of being activated. The most recent malware that exhibits the hybrid propagation feature is WannaCry, devastating ransomware that uses the campus network as the short-range propagation route. Although there has been no case of WannaCry infections on mobile phones yet, people have to be vigilant because of its extremely destructive power. Figure 1 gives a brief description of the propagation dynamics of the hybrid worm. The synergetic infection mechanism visibly increases the infection ability of the worm and brings a considerable challenge to the worm containment task.

To solve this problem, one possible method is to integrate the SINs and the GINs into one single network using the dimensionality reduction method [10]. However, the evolution speed of these two networks is quite different, so the integrating process usually does not make any sense. In this paper, we adopt a divide-and-conquer strategy and propose CC^2 (2-dimensional circulation controller). We first model the propagation dynamics of the hybrid worm in the mobile Internet and analyze the vulnerable spot in the worm propagation chain. Next, two components are designed to reduce infection rates. The first one is the SINs containment unit which aims to solve the secondary forwarding caused by acquaintances based on the fact that the propagation of the worm on social networks mainly depends on the closeness of social relationships. Another one is the GINs feedback unit. Some devices are geographic proximity and offer the chance for the worm to propagate across communities with short-range communications. Based on the security history records, the GINs feedback unit restricts the communication with insecure devices with a certain probability. The outputs of these two units serve as the input parameters of each other, forming a cyclic state.

The rest of the paper is organized as follows: Section 2 introduces related works. Section 3 discusses the propagation dynamics of the hybrid worm. Section 4 gives detailed descriptions of CC^2. The proposed methods are evaluated in Section 5. In Section 6, we make conclusions and envision further work.

2. Related Works

Although the worm is well understood on the Internet [6, 11], the worm on mobile Internet, however, has received only limited attention. In simple terms, the existing methods fall into two categories. For the short-range worm containment, Su et al. [12] showed that Bluetooth is an essential interface for worm's propagation. Yan and Eidenbenz [13], Mickens and Noble [14], and Morris-king and Cam [15]

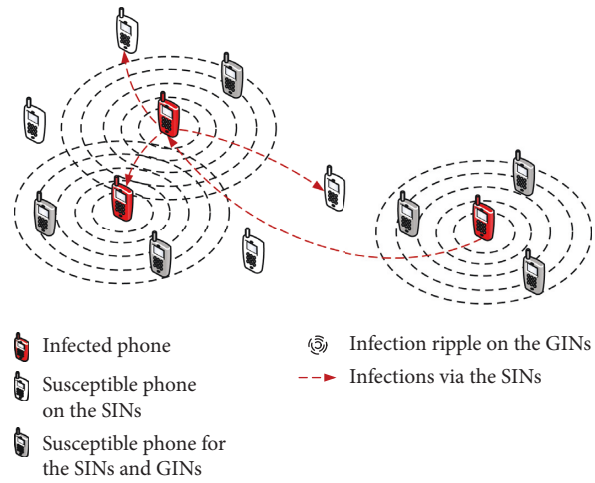

Figure 1: The hybrid spreading behavior of the mobile Internet worm.

confirmed this conclusion by analyzing the propagation dynamics of the worm transferred via the Bluetooth interface. Zyba et al. [4] designed a distributed coping scheme to eliminate the adjacent worm by using the worm signature. However, the time complexity of the algorithm is too challenging to solve a vast network. Yang et al. [16] proposed a sensor worm coping scheme based on graph coloring. The basic principle of this method is to increase the diversity of software version in the network. Li et al. [5] proposed a method to evaluate node vulnerability and control the worm by restricting the communication between vulnerable nodes. Miklas et al. [17] exploited social relations to improve the security of the Bluetooth interface and reduced the propagation speed of the worm by refusing connection requests from strangers. Gao and Liu [18] focused on the impacts of human behaviors on worm propagation and proposed a two-layer network model to protect large-scale dynamic mobile networks. The short-range worm relies on the direct connections between hardware interfaces and currently tends to attack wireless sensor networks [19–21] and vehicular networks [22–24].

For the long-range worm containment, Fleizach et al. [25] verified the differences in propagation characteristic between the Internet and the mobile Internet worm and evaluated the propagation effect of the MMS worm on cellular networks. Meng et al. [7] investigated the credibility of the communications in Short Messaging Service (SMS) by analyzing the trajectory data in the mobile networks. Bose et al. [26] utilized a quarantine method to limit the interaction between vulnerable nodes in MMS networks. Zhu et al. [8] considered that the core nodes in social networks should be immunized first, but this method ignores the transmission route of the worm via the Bluetooth interface, so the worm still has an opportunity to quickly forward. Moreover, the algorithm needs the number of clusters k, which is incognizable for social networks in advance. Zhao et al. [27] integrated the centralized and decentralized patch distribution strategy by constructing a new network layer model. Yang and Yang [28] proposed an evaluation

framework for testing patch distribution efficiency. The key to long-range worm containment is to identify the area of high-speed infection. The latest research studies tend to use community detection [9, 29, 30] and social influence analysis [31, 32] to solve this problem.

The above studies have made remarkable progress in the field of SINs and GINs worm containment, respectively. However, hybrid worm containment on the mobile Internet is still an open issue. The contribution of this study is that we formalized the propagation equation of the hybrid worm and proposed a worm containment scheme based on the mesoscopic analysis. Different from the flooding patching strategy, we preferentially distribute patches to the high-impact nodes in the network and establish a link between historical communication records and security predictions through Bayesian inference.

3. Propagation Dynamics of the Hybrid Worm

Susceptible infected recovered (SIR) model is used to measure the propagation dynamics of contagions within a population under contact infections in epidemiological theory [33]. Inspired by [2], we propose the hybrid-SIR model to depict the propagation characteristics of the hybrid worm by changing the propagation criteria of the SIR model.

Let $S(t)$, $I(t)$, and $R(t)$ represent the number of susceptible, infected, and recovered nodes at time t, respectively. $J(t) = I(t) + R(t)$ calculates the number of infected nodes including immune. Let β, γ, and N, respectively, represent the infection rate, the recovery rate, and the total number of nodes, then the differential equations of the SIR model are given by

$$\begin{cases} \dfrac{\mathrm{d}J(t)}{\mathrm{d}t} = \beta J(t)[N - J(t)], \\ J(t) = N - S(t), \\ \dfrac{\mathrm{d}R(t)}{\mathrm{d}t} = \gamma I(t). \end{cases} \tag{1}$$

For our hybrid-SIR model, all of the interactions between smartphones derive from the SINs and the GINs. Let $I_{\mathrm{SINs}}(t)$ and $I_{\mathrm{GINs}}(t)$, respectively, represent the number of infections via the SINs and the GINs at time t. $I(t) = I_{\mathrm{SINs}}(t) + I_{\mathrm{GINs}}(t)$ calculates the total number of infected nodes at time t. $S(t)$ denotes the total number of susceptible nodes at time t. Then, we have

$$\begin{aligned} I_{\mathrm{SINs}}(t) + I_{\mathrm{GINs}}(t) + S(t) = N, \\ \frac{\mathrm{d}I(t)}{\mathrm{d}t} = \frac{\mathrm{d}I_{\mathrm{SINs}}(t)}{\mathrm{d}t} + \frac{\mathrm{d}I_{\mathrm{GINs}}(t)}{\mathrm{d}t}. \end{aligned} \tag{2}$$

When an infected smartphone intends to propagate the worm through the SINs, it behaves like a traditional SMS virus which sends messages to the one found in the local contact list (reflected in the degree of nodes). Let β_{SINs} denote the probability of the worm being activated, η_{SINs} denote the average degree of nodes in the SINs, the number of susceptible nodes neighboring a start point equals to $\eta_{\mathrm{SINs}}\beta_{\mathrm{SINs}}$, and the total number of susceptible nodes is thus given by

$$S'(t) = S(t)\frac{\eta_{\mathrm{SINs}}\beta_{\mathrm{SINs}}}{N} I(t). \tag{3}$$

Based on equation (3), the equation that depicts the dynamics of infected nodes in the SINs with time is

$$\frac{\mathrm{d}I_{\mathrm{SINs}}(t)}{\mathrm{d}t} = \beta_{\mathrm{SINs}}^2 \frac{\eta_{\mathrm{SINs}} S(t)}{N} I^2(t) - \gamma' I(t), \tag{4}$$

where γ' denotes the recovery rate by means of sending patches.

When an infected device tries to propagate the worm through the GINs, it first detects all adjacent neighbors (related to population density) within its propagation range R. We assume that the smartphones are distributed with density ρ, then the average amount of accessible smartphones η_{GINs} equals to $\rho\pi R^2$.

At the new time step, only the infected smartphone that lies on the circumference of the infection ripple has the chance to exchange messages with the susceptible smartphones and therefore have the possibility to infect them, while the smartphones located in the interior of the infection ripple are not contributing to further spatial infections (that part of nodes are immune or still under infection).

Let $F'(t)$ and $F(t)$ represent the number of smartphones lies on and lies within the circumference of the ripple, respectively; the infected dynamics of the periphery nodes with the incremental infected radius $r(t)$ is described as

$$F'(t) = F(t) - \rho\pi(r(t) - R)^2. \tag{5}$$

Here, $r(t) - R$ calculates the radius of the infection ripple at time $t - 1$. We have $F(t) = \rho\pi r^2(t)$ based on the fact that the number of nodes inside the infection ripple is the sum of neighbors of the center node. Thus, equation (5) can be simplified into

$$F'(t) = 2R\sqrt{\rho\pi}\sqrt{F(t)} - \eta_{\mathrm{GIN}}. \tag{6}$$

We assume that the spatial infection ripple of a start node is generated at time r' and achieves the current infection status $F(r' + s')$ after the duration of s'. Then, the incremental infection at time $r' + s'$ is

$$F'(r', s') = \beta_{\mathrm{GINs}} \frac{2\eta_{\mathrm{GINs}}\left(2R\sqrt{\rho\pi}\sqrt{F(r', s')} - \eta_{\mathrm{GINs}}\right)^2}{3N} S(r' + s'). \tag{7}$$

After simplification, we have

$$F'(r', s') = \beta_{\mathrm{GINs}} \frac{2\eta_{\mathrm{GINs}}^2\left(c\sqrt{F(r', s')} - 1\right)^2}{3N} S(r' + s'). \tag{8}$$

In equations (7) and (8), $2/3N$ indicates that for each smartphone that lies on the circumference of the ripple (e.g., the node v in Figure 2), approximately two-thirds of its neighbors outside the ripple are susceptible. β_{GINs} denotes

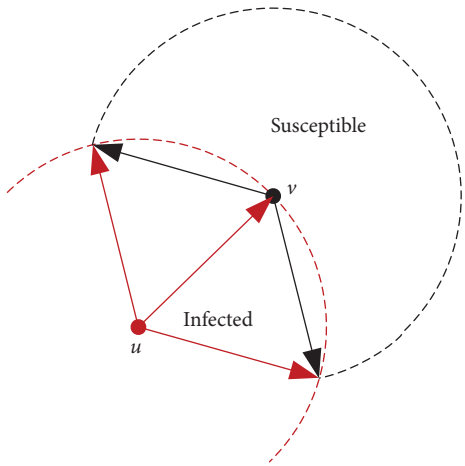

Figure 2: Infections outside the ripple.

the probability of a GIN worm being activated in the susceptible device. $c = 2/R(\rho\pi)^{1/2}$ is the proportionality constant [34]. The incremental infection of all infection ripples at time t is thus given by

$$\frac{dI_{GINs}(t)}{dt} = \int_0^t I'_{SINs}(s)F'(s, t-s)ds. \quad (9)$$

It means that I_{SINs} dominates the spatial infections of the worm. In other words, there are $I'_{SINs}(s)$ infected phones emerging at time s and each one approximately contributes $F'(s, t-s)$ incremental infection at time t.

We roughly verified the propagation performance of the worm via the SINs, the GINs, and the hybrid mode based on the proposed model. Two parameter settings are considered: one is $\rho_{GINs} = 0.75$, $\beta_{SINs} = \beta_{GINs} = 0.1$, and $\eta_{GIN} = \eta_{SIN} = 6$; the other is $\rho_{GINs} = 0.85$, $\beta_{SINs} = \beta_{GINs} = 0.1$, and $\eta_{GINs} = \eta_{SINs} = 10$. Simulation results in Figure 3 indicate that with the increasing number of smartphone's neighbors ($\eta_{GINs} = \eta_{SINs} = 10$), the hybrid model greatly enhances the power of worm infection and brings considerable challenges to the worm containment task.

4. Containment Scheme of the Hybrid Worm

In this section, we present the framework of CC^2 (shown in Figure 4) and demonstrate the basic idea of the system. The system consists of five units. On the top, it starts with the Online Community Manager. In social networks, messages from close friends have a higher probability of being received and further opened. Thus, the infection rate β_{SINs} in equation (4) basically equals to one. We cannot expect the recovery rate γ' to be increased, as the mobile operator requires necessarily time to detect worm and code patches. Also, directly modifying η_{SINs} is unrealistic. However, reducing the average degree of nodes is seeking the sparse representation of the network. Online Community Manager maintains a set of community structures [35], in which nodes are closer to each other than anyone else outside the community. In this light, if the nodes are mapped to the communities, the propagation speed of the worm can be significantly reduced at the mesoscopic level during the exponential growth phase.

node monitor is used to monitor the infection status of nodes, and generate guard nodes lies on the periphery of each community. Once the worm has been found on the network, the system starts generating the worm signature and delivers it to the guard nodes through a data transmitter. If a guard node receives the worm signature before it is infected, it will become immune to the worm. As a result, the worm propagation can be blocked since any malicious message has to go through the guard nodes to reach the adjacent communities.

All the information about the nodes is gathered in the information collector and provides the evidence for the security evaluator. The purpose of security evaluation is to prevent the worm from spreading across communities through location-based infections. For those users with high activity and poor security, we randomly reject the communication with them until their security consciousness is improved. This scheme will reduce η_{GINs} in equation (8) to some extent, while β_{GINs} in the equation is intractable since it describes the density of the population in real space. The output of the security evaluator also guides the community detection process, ensuring security within the community. Section 5 describes more details of CC^2.

5. System Description

5.1. Online Community Manager. Community detection in this component consists of three steps: security evaluation, label propagation [36], and overlapping community combination. Note that we do not intend to detect "high-quality" partitions (i.e., with higher modularity [37]), but to detect a reasonable structure with higher density and internal forwarding efficiency. We use label propagation (LP) algorithm to simulate the process of nodes receiving neighbor messages. In the algorithm, if a sender satisfies the following conditions, the label he delivers is more likely to be accepted: (1) enough security; (2) with coercive power.

For condition 1, we first analyze the communication security of nodes on the SINs (marked as T_{SINs}) and estimate the probability of secure communication θ with the Bayesian formula as follows:

$$P(\theta \mid x) = \frac{f(x \mid \theta)P(\theta)}{\int_\Theta f(x \mid \theta)P(\theta)d\theta}. \quad (10)$$

Let a denote the number of secure communications between node i and its neighbors in n communications (marked as normal in Figure 4), and $b = n - a$ calculates the number of insecure communications in the same situation (marked as infected in Figure 4). Before inference, we assume that the probability of secure communication θ follows uniform distribution $\mathrm{Uni}(0, 1)$, which means $P(\theta) = 1$, $\theta \in (0, 1)$. Estimating communication security is a binomial experiment $\mathrm{Bin}(n, p)$, so the likelihood of a out of n communications being normal is

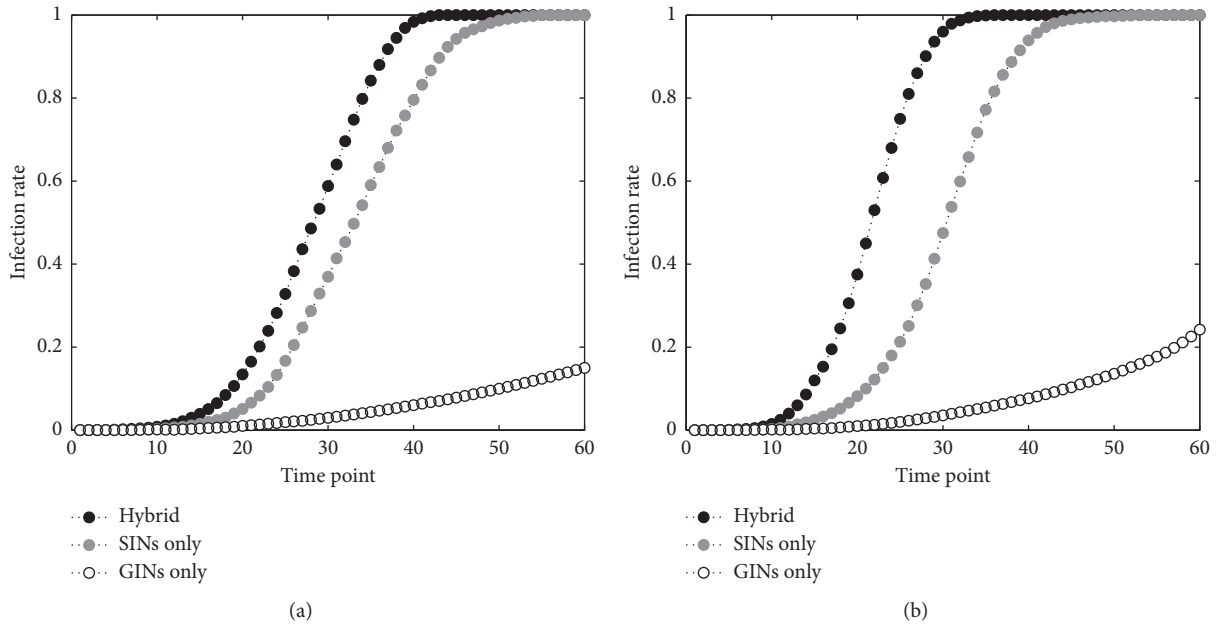

Figure 3: Propagation performance of the worm spreading via the SINs, the GINs, and the hybrid mode: (a) GINs = 0.75 and $\eta_{\text{GINs}} = \eta_{\text{SINs}} = 6$ and (b) GINs = 0.85 and $\_\eta_{\text{GINs}} = \eta_{\text{SINs}} = 10$.

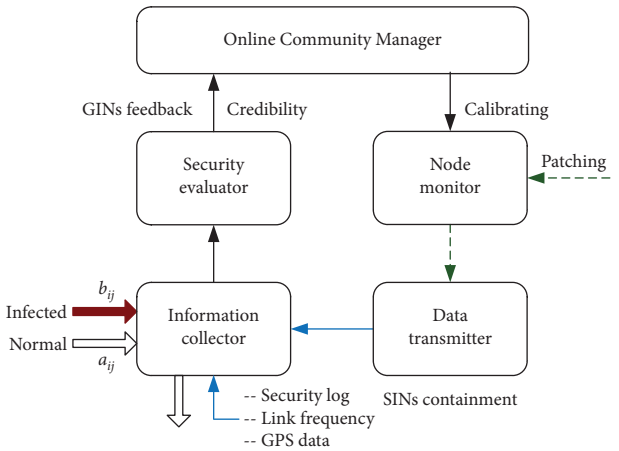

Figure 4: System framework of CC^2.

$$f(x \mid \theta) = \binom{n}{k} \theta^x (1-\theta)^{n-x}. \tag{11}$$

Substituting (equation (11)) into (equation (10)), we have

$$P(\theta \mid x) = \frac{\binom{n}{k} \theta^x (1-\theta)^{n-x}}{\int_0^1 \binom{n}{k} \theta^x (1-\theta)^{n-x} d\theta} = \frac{\theta^x (1-\theta)^{n-x}}{\int_0^1 \theta^x (1-\theta)^{n-x} d\theta}. \tag{12}$$

Given that for any real number α and β, the beta function satisfies

$$B(\alpha, \beta) = \int_0^1 x^{\alpha-1} (1-x)^{\beta-1} dx. \tag{13}$$

Let $x = a$ and $n - x = b$, respectively, represent the number of normal and infected communications, then we have

$$P(\theta \mid x) = \frac{\theta^a (1-\theta)^b}{\int_0^1 \theta^{a+1} (1-\theta)^{b+1} d\theta} = \frac{\theta^a (1-\theta)^b}{B(a+1, b+1)}. \tag{14}$$

Equation (14) indicates that the communication security of node i follows beta distribution with parameters $a+1$ and $b+1$, and its expectation is

$$E(\Theta) = \frac{a+1}{a+b+2}. \tag{15}$$

According to the law of large numbers, we have $T_{\text{SINs}}(i) = E(\Theta)$. It seems that frequent sending of secure messages is an indication of reliable communication. This conclusion, however, does not consider the security of the adjacent environment. We define the user's security evaluation function as

$$\text{se}(i) = T_{\text{SINs}}(i) \times \exp\left(-\frac{1}{d_i} \sum_{j \in \eta(i)} T_{\text{SINs}}(j) \right). \tag{16}$$

Here, d_i denotes the degree of node i. In equation (16), the multiplier represents the security of the adjacent environment. It can be found that when the adjacent

environment is insecure and the user can still send secure messages with higher probability, the user is considered to be relatively reliable.

Similarly, we can define the likelihood of node i sending insecure messages as $uT_{\mathrm{SINs}}(i) = 1 - T_{\mathrm{SINs}}(i)$. For condition 2, if the malicious message sent by a user does not significantly change the security of the adjacent environment, the coercive power (cp) of the user is weak. We define this condition in equation (17), which measures the probability of a security-conscious user receiving a malicious message:

$$\mathrm{cp}(i) = uT_{\mathrm{SINs}}(i) \times \left(1 - \exp\left(\frac{1}{d_i}\sum_{j\in\eta(i)}(\mathrm{se}(j) - 1)\right)\right). \tag{17}$$

Based on equations (16) and (17), the transition probability of messages (whether secure or not) sent by node i on the SINs is defined as $\mathrm{soc}(i) = \max\{\mathrm{se}(i), \mathrm{cp}(i)\}$. Combined with the security evaluation result $\mathrm{geo}(i)$ (equation (27) submitted by the GINs feedback unit, the transition probability of the community label held by node i is defined as

$$L(i) = \frac{\max\{\mathrm{soc}(i), \mathrm{geo}(i)\}}{1 + |\mathrm{soc}(i) - \mathrm{geo}(i)|}. \tag{18}$$

We modify the propagation and acceptance criteria of the LP process and propose a security-based label propagation (SLP) algorithm based on equation (18); the pseudocode is described in Algorithm 1.

The output of Algorithm 1 is a set of overlapping communities, as each node is allowed to have multiple community labels in Step 5. However, this will increase the bandwidth cost of the network, as the more the number of communities, the more the number of guard nodes needs to be monitored. Therefore, we need to execute a merging program to reduce structure redundancy. We define the structure stability of community C as

$$\mathrm{ss}(C) = \left(\frac{1}{|C| - 1}\sum_{i\in C}\frac{1}{|C|}|\eta(i)|\right) \times \sum_{i\in C}\frac{1}{|C|}L(i). \tag{19}$$

Equation (19) measures the tradeoff between structure density and security. What we need to evaluate is whether the overlapping part provides significant stability for the entire community. In our method, any two communities C_A, $C_B \in C$ can be merged if $\mathrm{ss}(C_A \cap C_B) \geq \min(\mathrm{ss}(C_A), \mathrm{ss}(C_B))$. Note that the merging program also provides a way to detect dynamic communities. Given a network G, an initial community structure C, and an incremental update ΔG, we have

$$\mathrm{SLP}(G \cup \Delta G) = \mathrm{Merge}(C, \mathrm{SLP}(\Delta G)). \tag{20}$$

Equation (20) indicates that when the network changes, we only need to execute the SLP algorithm on the newly added nodes in the local environment and then run the merging program to avoid repetitive computation of the existing results. Figure 5 shows a three-step example of the SLP process. Each node chooses to join a security substructure based on the observed contact history. In the algorithm, the label in iteration t is always based on its neighboring labels in iteration $t - 1$ to avoid the oscillations of labels [36].

5.2. Data Transmitter. The function of the data transmitter is twofold: (1) construct the forwarding strategy of worm signature and (2) select the guard nodes (the initial delivery node set). When the worm is found in the network, the data transmitter will send the worm signature to the network operations center (NOC) and then distribute it to each network node. Flooding is a possible forwarding strategy. However, this will bring enormous bandwidth costs and influence normal network communications. Besides, the flooding method considers that all the network nodes have the same delivery priority, and the forwarding capability is weak. We propose a new function to calculate the forwarding capability of nodes, which is described as follows:

$$\varphi(i) = \frac{1}{n}\sum_{j=1}^{n}\left(L(j) \times \exp\left(-\frac{1}{2}h_{ij}\right)^2\right). \tag{21}$$

Here, h_{ij} denotes the hops between node i and node j. Equation (21) shows that the delivery priority of nodes depends on the security and coercive power of its neighbors within 2-hops. The start points of the delivery process include the overlapping nodes and the endpoints of links between communities. This setting will ensure community quarantine [38] as early as possible and prevent worms from spreading across communities. Algorithm 2 describes the pseudocode of the data transmitter.

5.3. Security Evaluator. In the GINs feedback unit, the security evaluator is an essential component which is designed to perform community quarantine procedure and provide security evaluation results for short-range data transmission. Different from social applications, there is not enough evidence to use the Bayesian formula to deduce the communication security of users since short-range interfaces are not used frequently in practice. However, this will not prevent the use of Bayesian thoughts in this component.

Let a'_{ij} and $b'_{ij} = n - a'_{ij}$, respectively, represent the number of normal and infected communications that node i receives from its neighbor j in n rounds. Based on the observed security history, node i will calculate the communication security between neighbor j as

$$\mathrm{cs}_{i,j} = \frac{a_{i,j}}{a_{i,j} + b_{i,j}}. \tag{22}$$

When there is no supervision record, the initial value of both $a_{i,\ j}$ and $b_{i,\ j}$ equals to one. Equation (22) is unlikely to provide a reliable security calculation directly when the sample is insufficient. To solve this problem, we design an uncertainty computing function uc_{ij}, which is defined as

$$\mathrm{uc}_{i,j} = \frac{\min\{a_{i,j}, b_{i,j}\}}{\max^2\{a_{i,j}, b_{i,j}\}}. \tag{23}$$

Require:
Contact history from information collector;
Ensure:
Community structure $C = \{C_1, C_2, \ldots\}$;
Step 1. Initialize the community label. Each node i in graph G contains a feature vector $\text{vec}_t(i) = \{\{l_1, b_1 \times L(i)\}, \{l_2, b_2 \times L(i)\}, \ldots\}$, where l represents the community label, b represents the belonging degree, L represents the transition probability of l, t represents the number of iterations. For example, the feature pair of i is $\text{vec}_0(i) = \{i, 1 \times L(i)\}$ means the initial label of node i is itself and the belonging degree equals to one;
Step 2. Set $t = 1$;
Step 3. Arrange each $i \in G$ in a random order and assign them to V;
Step 4. For each $i \in V$, $j \in \eta(i)$, calculate the belonging degree of the adjacent community label l_x as $b_t(l_x) = (1/|l_x|)\sum_{l_x \in \text{vec}_t(j)}(b_x \times L(j))$;
Step 5. For each $i \in V$, if exists $b_y \in \text{vec}_t(i)$ s.t. $b_y < b_t(l_x)$, then replace b_y with b_x, and remain $L(i)$ unchanged. Else stop and jump to **Final**;
Step 6. Set $t = t + 1$ and go to **Step 3**;
Final.

Algorithm 1: Security-based label propagation algorithm.

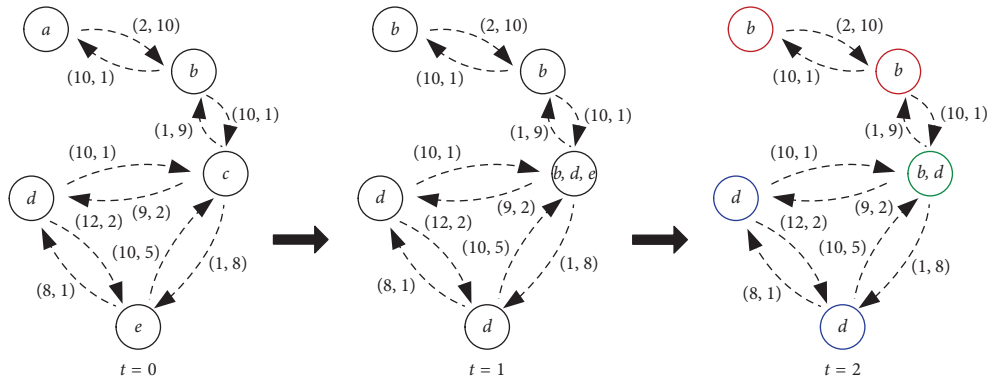

Figure 5: A simple simulation of the SLP process. The numbers in parentheses represent the number of normal and infected communications in turn.

In equation (23), when $a_{i,\ j}$ or $b_{i,\ j}$ dominates, the value of $\text{uc}_{i,\ j}$ declines, which indicates that the security of the next communication will be more predictable. Based on equations (22) and (23), the revised communication security rs_{ij} can be defined as

$$\text{rs}_{i,j} = \text{cs}_{i,j}\left(1 - \text{uc}_{i,j}\right). \tag{24}$$

Another parameter that needs to be considered in the security evaluator is the interaction frequency, which is measured in this paper as the regularity of interaction. We present the concept of circulation connections. As shown in the dashed box in Figure 6, the communication interval can be regarded as continuous communication. Let n_{cont} and n_{cycle}, respectively, represent the number of continuous and circulation connections in a sliding window of size ws, then the interaction frequency q_{ij} is thus given by

$$q_{i,j} = \frac{n_{\text{cont}} + n_{\text{cycle}}}{\text{ws}}. \tag{25}$$

In the four cases shown in Figure 6, $q_{i,\ j}$ equals to 0.9, 0.8, 0.2, and 0, respectively. The first two cases show strong regularity, which probably comes from close friends. While the latter two cases with a low $q_{i,\ j}$ could be the contact history from acquaintances or strangers. From a statistical point of view, a low communication frequency will not provide sufficient evidence to support the communication security evaluation. For example, when $\text{rs}_{ij} = 0.9$ and $q_{ij} = 0.1$, rs_{ij} should be punished appropriately; when $\text{rs}_{ij} = 0.1$ and $q_{ij} = 0.1$, rs_{ij} should be gained slightly. We define the communication security of arc (i, j) on the GINs as

$$T_{\text{GINs}}(i, j) = \text{rs}_{ij} + \varepsilon\left(1 - q_{ij}\right) \times \text{soc}(j). \tag{26}$$

Here, $\varepsilon = \exp(-\text{rs}_{ij}) - \text{rs}_{ij}$ is used to decide whether the security should be gained or punished and the regulation threshold is approximately equal to 0.5671. When $q_{ij} = 0$, we have $\text{rs}_{ij} = 0$. In this case, we use $\text{soc}(j)$ to give a reference value to the GINs security evaluation. Based on equation (26), the security evaluation result of node i in the GINs can be defined as

$$\text{geo}(i) = \frac{1}{|\eta(i)|}\sum_{j \in \eta(i)} T_{\text{GINs}}(j, i). \tag{27}$$

Require:
Graph $G = (V, E)$; community structure $C = (C_1, C_2, \ldots)$;
Ensure:
Guard nodes V_G; signature propagation;
Step 1. Initialize V_G. For each $C_a, C_b \in C$ do $V_G \longleftarrow \{i \mid i \in C_a \cap C_b\}$, $V_G \longleftarrow \{i, j \mid i \in C_a, j \in C_b, (i, j) \in E\}$;
Step 2. For each $i \in V_G$, initialize the worm signature from the NOC in i's buffer as $\omega(i) = \{\omega_1(i), \omega_2(i), \ldots\}$; set initial token τ in i's buffer;
Step 3. For each $i \in G$, $j \in \eta(i)$, if exists τ in i's buffer and j's buffer is null then duplicate and deliver $\omega(i)$ to node j; if exists j s.t. $\varphi(j) \geq \varphi(i)$ then duplicate and deliver token τ to node j;
Step 4. If all nodes have received the worm signature, then stop and jump to **Final**. Else go to **Step 3**;
Final.

Algorithm 2: Data transmitter.

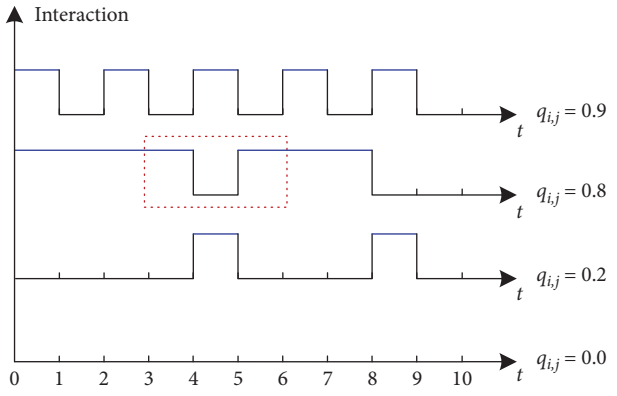

Figure 6: Interaction frequency between node i and j in a sliding window of size 10 time units.

Algorithm 3 describes our idea on community quarantine. For node $i \in V$, $j \in \eta(i)$, if i has not received any worm signature, i will reject j's messages with a certain probability by using equation (26). Considering channel utilization, we first perform the algorithm on users who are located in different communities but geographically adjacent (i.e., within 30 m [39]). Next, these users will broadcast quarantine notification to other community members to achieve global synchronization.

Here, $uT_{\mathrm{GINs}}(i, j) = 1 - T_{\mathrm{GINs}}(i, j)$. We use counter $\mathrm{cnt}(i, j)$ to record the number of consecutive messages on arc (j, i) and increase the rejection probability P_{R} with $\mathrm{cnt}(i, j)$ grows.

5.4. Complexity Analysis. The computation complexity of CC^2 mainly consists of three parts: (1) community detection and combination in Online Community Manager; (2) forwarding capability calculation in Algorithm 2; and (3) security evaluation in Algorithm 3.

For the first part, the time complexity of Algorithm 1 is $O(|\mathrm{vec}|(m + n))$ [40], m is the number of edges, n is the number of nodes, and $|\mathrm{vec}|$ is the average number of communities per node. The time complexity of the merging process is $O(d_{\mathrm{c}}|C|)$, where d_{c} represents the average degree of communities and $|C|$ is the number of communities.

For each community $C_i \in C$, we have

$$|C| < n_{\mathrm{avg}}|C| = \sum_{C_i \in C} |C_i|, \tag{28}$$

where n_{avg} represents the average number of nodes in the community. The SLP algorithm guarantees that each node is at least located in one community, therefore

$$\sum_{C_i \in C} |C_i| = n + \sum_{i<j} |C_i \cap C_j|. \tag{29}$$

It is quite clear that $|C_i \cap C_j| < n$. Let d_{avg} represent the average degree of nodes, then we have

$$\sum_{i<j} |C_i \cap C_j| < n d_{\mathrm{avg}} < m. \tag{30}$$

Based on equations (27)–(30), we have $O(d_{\mathrm{c}}|C|) = O(d_{\mathrm{c}}(m+ n))$. Since $d_{\mathrm{c}} < |\mathrm{vec}|$, the time complexity of the first part is $O(|\mathrm{vec}|(m + n))$. For the adaptive procedure (equation (20)), the time complexity of the SLP process and merging process is $O(|\mathrm{vec}|(\Delta G + n) + |C|(\Delta G + n)) = O(|\mathrm{vec}|(\Delta G + n))$, where ΔG represents the incremental update of the network.

For part 2 and part 3, the time complexity is $O(d_{\mathrm{avg}}n) + O(|\mathrm{ws}|d_{\mathrm{avg}}n) = O(|\mathrm{ws}|d_{\mathrm{avg}}n)$, where ws is the size of the sliding window. We use the KMP algorithm to match circulation connections in the window, and the time complexity is $O(\mathrm{ws})$. For the adaptive procedure, the time complexity of part 2 and part 3 is $O(n\Delta G) + O(|\mathrm{ws}|n\Delta G) = O(|\mathrm{ws}|n\Delta G)$.

In CC^2, all the components are executed sequentially; therefore, the time complexity of the initial phase is $O(|\mathrm{vec}|m + |\mathrm{ws}|d_{\mathrm{avg}}n)$. For the adaptive procedure, the time complexity is $O((|\mathrm{vec}| + |\mathrm{ws}|\Delta G)n)$. Note that in the above analysis, we assume that the system confronts extreme conditions (e.g., we assume that the newly added nodes link to all the existing nodes). In practice, the execution time of CC^2 will be shorter.

6. Experiments and Analyses

In this section, we will present and discuss the experiments of CC^2 on two different kinds of real-world traces including the MIT Reality (consists of students and faculty in the MIT Media Laboratory) (http://crawdad.org/mit/reality) and the Haggle Project (conducted for four days during INFOCOM

Require:
$G = (V, E)$; $C = \{C_1, C_2, \ldots\}$; $T_{\mathrm{GINs}}(i, j)$;
Ensure:
Community quarantine;
Step 1. $V_Q \longleftarrow \{i, j \mid i, j \in V; \mathrm{vec}(i) \cap \mathrm{vec}(j) = \varnothing; \mathrm{distance}(i, j) < 30\,\mathrm{m}\}$;
Step 2. For each $i \in V_Q$, $j \in V$, if $\omega(i) = \varnothing$, i will reject the messages from j (except $\omega(j)$) with probability $P_{\mathrm{R}}(i, j) = uT_{\mathrm{GINs}}(i, j)/1 - \mathrm{cnt}(i, j) \times uT_{\mathrm{GINs}}(i, j)$;
Step 3. For each $i \in V_Q$, i broadcasts quarantine notification to node $\{j \mid \mathrm{vec}(i) \cap \mathrm{vec}(j) \neq \varnothing\}$, j will reject messages from other communities with probability P_{R};
Step 4. For each $i \in V$, if $\omega(i) \neq \varnothing$, then stop and jump to **Final.** Else go to **Step 3**;
Final.

Algorithm 3: Community quarantine.

2006 in Barcelona) (http://crawdad.org/cambridge/haggle). In both datasets, Bluetooth contacts, phone call records, and users' locations were provided to construct the GINs and the SINs layer, respectively. Each Bluetooth contact includes the start time, end time, and the IDs of nodes. Each phone call record includes call logs, cell tower IDs, application usage, and phone status (such as charging and idle).

For each round of the simulation, a portion (default 35%) of the dataset was used as the contact history (including normal and infected communications). At the very beginning, we randomly chose 0.05% of the nodes as the seed set of worm sources to initiate the infection. The propagation of the malicious message follows the hybrid propagation model proposed in Section 3. In the model, the recovery rate is $\gamma' = 0.05$ and the probability of a node sending messages to acquaintances (its top 10 neighbors) in the SINs and the GINs is set to 0.2 and 0.05, respectively and to strangers is set to 0.05 and 0.01, respectively. The activation probability of the worm is set to 0.95 (from acquaintances) and 0.05 (from strangers) without running any coping scheme. To avoid flooding broadcast, each infected node attempts to attack its neighbor nodes only once.

6.1. The Analysis on Community Quality. In this section, we test the quality of the community structure generated by the SLP algorithm (Algorithm 1) in terms of the average community size (marked as ACS), the number of detected communities (marked as #Communities), the structure stability of community structure (equation (19) marked as SS), the EQ function, and the execution time. The EQ function proposed by Shen et al. [41] is widely used in evaluating overlapping communities, which is described in the following equation:

$$\mathrm{EQ} = \frac{1}{D} \sum_{i} \sum_{v \in C_i, w \in C_i} \frac{1}{O_v O_w} \left[A_{vw} - \frac{d_v d_w}{D} \right], \tag{31}$$

where d_v is the degree of node v, $D = \sum_{vw} A_{vw}$ is the total degree of the network nodes, A_{vw} is the element of the adjacency matrix of the network, O_v is the number of communities which the node v belongs to, and C_i is the i-th community in the network. The comparison algorithms include COPRA [40] (based on label propagation), EAGLE [41] (high-performance overlapping detection algorithm), and A3CS [42] (detecting dynamic community structure). We perform the algorithms mentioned above in Haggle and MIT Reality, respectively, and the experiment results are covered in Tables 1 and 2.

In both datasets, the SLP algorithm makes more numbers of communities (10 and 16, respectively) and smaller size of communities (8 and 6, respectively) than others, which means that the detected communities are more granular. When this feature is applied to Algorithm 2, CC^2 will have more opportunities to find high-impact nodes in the local environment and thereby will increase the performance of worm containment and the efficiency of sending patches. The EQ value of the SLP algorithm (0.439 and 0.441, respectively) is slightly lower than that of COPRA and EAGLE. As we discussed in Section 5, the SLP algorithm is not aiming to find high "modularity" partitions [37], but to find communities with higher security awareness inside. So the structure stability (SS score) of SLP (0.486 and 0.533, respectively) is stronger than the other three algorithms. In terms of the execution time, SLP (1.132 and 1.296, respectively) is slightly slower than COPRA and faster than EAGLE and A3CS, which is in line with the online community monitoring task.

6.2. The Performance of SINs Containment Unit. The test in this section mainly considers three scenarios: (1) turn on and off the GINs feedback unit to demonstrate the performance of the SINs containment unit; (2) replace the SLP algorithm with COPRA, EAGLE, and A3CS to verify the performance of the Online Community Manager in the SINs containment unit; and (3) compare with three community-based coping strategies, including Member [30], I2C [29], and TC-based [9]. We primarily focus on the infected ratio which is reflected in the proportion of nodes infected by the worm.

In Figure 7, the line graph marked as "no coping" depicts the infected ratio of worms with time without any coping scheme, which can be used as the benchmark in comparison. When the GINs feedback unit is turned off, the performance of CC^2 is slightly better than Member and I2C and better than COPRA, EAGLE, and A3CS due to the higher security within the community structure. CC^2 performs significantly when the GINs feedback unit is turned on, primarily because

Table 1: The comparison of the four community detection algorithms on Haggle.

Algorithms	#Communities	ACS	EQ	SS	Execution time
COPRA	8	10	0.445	0.450	0.923
EAGLE	5	16	0.472	0.476	1.521
A3CS	6	13	0.423	0.434	1.296
SLP	10	8	0.439	0.486	1.132

Table 2: The comparison of the four community detection algorithms on MIT Reality.

Algorithms	#Communities	ACS	EQ	SS	Execution time
COPRA	12	8	0.458	0.502	1.103
EAGLE	8	13	0.477	0.528	1.618
A3CS	11	9	0.436	0.478	1.425
SLP	16	6	0.441	0.533	1.296

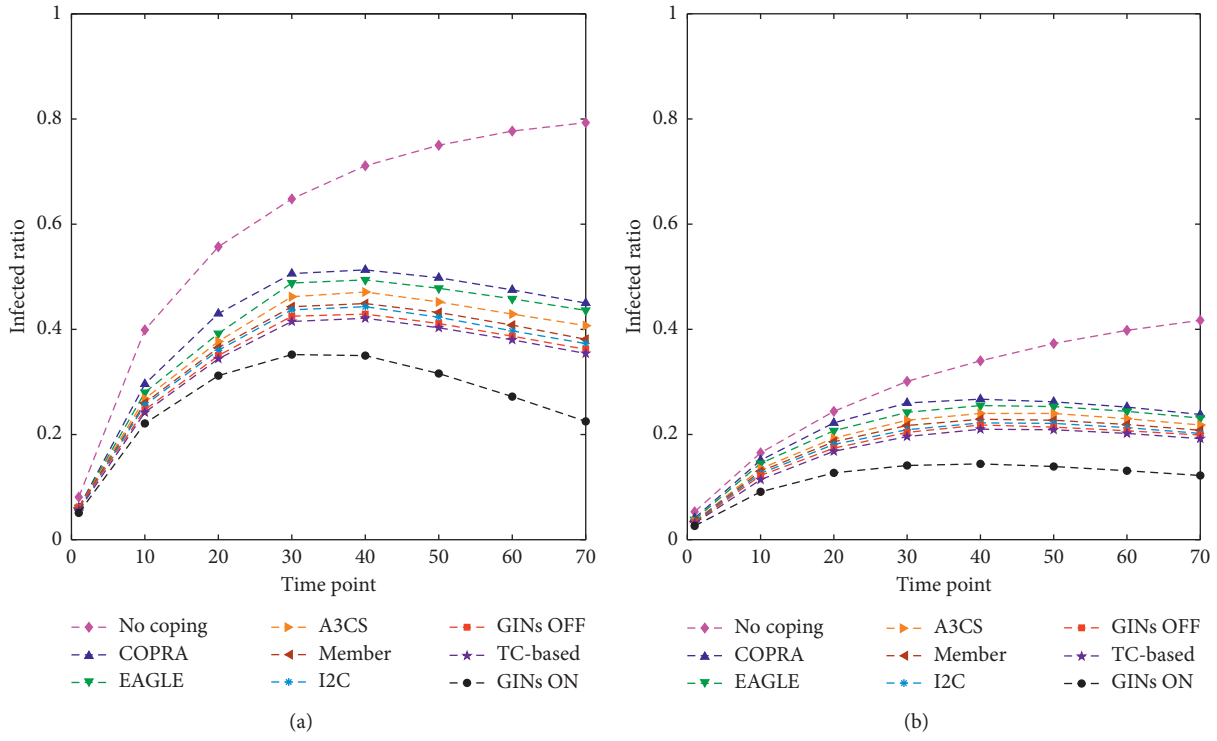

Figure 7: Performance comparison of long-range coping schemes on Haggle (a) and MIT Reality (b).

it decreases worm propagation across communities when the user's location is adjacent. When the recovery rate $\gamma' = 5\%$, CC^2 can constrain the further spread of worms around 30 time units and control the final infected ratio at 22.5% (Haggle) and 12.2% (MIT Reality) at 70 time units. We can find that at the mesoscopic level, the smaller the group size, the higher the internal security of the group, and the better the worm containment performance. Besides, the community quarantine strategy significantly improves the effect of worm containment, approximately equal to the difference between GINs ON and GINs OFF.

6.3. The Performance of GINs Feedback Unit. In this section, the SINs containment unit is turned on and off to demonstrate the performance of the security evaluator in the GINs feedback unit. The comparison methods include (1) a distributed local detection-based scheme [4] (marked as distributed), (2) a proximity signature forwarding-based scheme [4] (marked as proximity), (3) a Bluetooth-based malware coping scheme [21] (marked as hierarchical), (4) a pruning-based proximity malware coping scheme [15] (marked as *K*-distance), (5) a community-based proximity malware coping scheme [5] (marked as centralized), (6) a social network-based patching scheme [8] (marked as socializing), and (7) TC-based (performs well in the former test). The first four methods primary focus on the GINs worm containment task, while the latter three to contain the propagation of the worm on the SINs. We keep the relevant

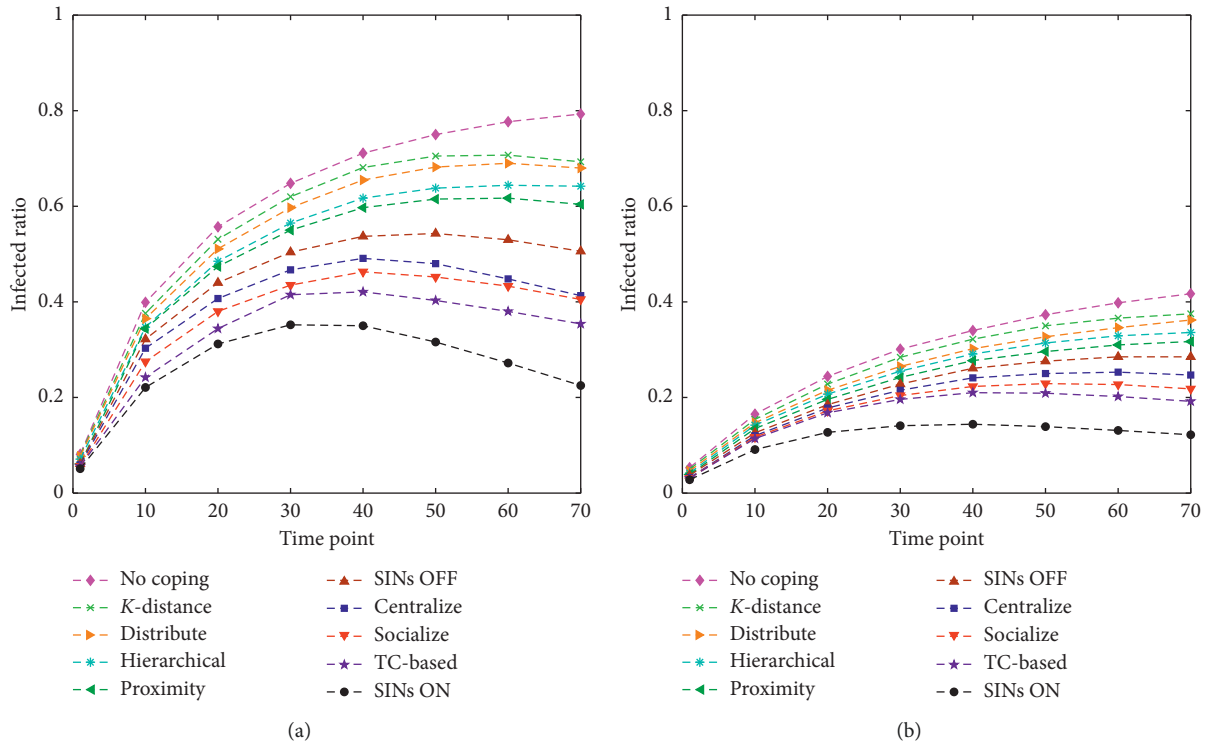

Figure 8: Performance comparison of short-range coping schemes on Haggle (a) and MIT Reality (b).

parameters as well as the original literature setting for comparison convenience.

Figure 8 gives the experiment results. In contrast to Figure 7, the performance of SINs OFF is weaker than GINs OFF, which indicates that the high-speed infection of worm relies on the community structure. The comparison with the other seven methods confirms this conclusion, the scheme using social relationships (centralize, socialize, and TC-based) is superior in performance to the scheme using geographic location (distribute, proximity, *K*-distance, and hierarchical). We also noted that the patching strategy is an effective way to control worms (proximity, centralize, and socialize), and its performance is much higher than that of the structure-based control methods (distribute, *K*-distance, and hierarchical). On the whole, SINs ON performs better than other methods with an average reduction of 14.4% (Haggle) and 5.9% (MIT Reality) in terms of the infected ratio. Another observation is that although the propagation speed of the GINs worm is slow, ignoring the GINs feedback unit, however, will significantly reduce the containment performance, which is approximately equal to the difference between TC-based and SINs ON.

6.4. The Comparison on Worm Containment Capability. This section mainly contains three series of experiments. First, we do tests on the percentage of patched nodes, which is defined as the average number of signatures forwarded in the network. Comparing Figures 8 and 9, we can see that CC^2 only needs to deliver 20.9% (Haggle) and 16.7% (MIT Reality) network nodes to control the infected ratio at 22.5% (Haggle) and 12.2% (MIT Reality). With the same patching ratio, proximity, centralize, socialize, and TC-based can only control the infected ratio around 60.4%, 41.3%, 40.5%, and 35.4% (Haggle) and 31.7%, 24.7%, 21.8%, and 19.2% (MIT Reality), respectively. Distribute does not need to deliver patches, but it also loses the chance to be immune to worms. As a result, the performance of distribute is only slightly better than "no coping" in Figure 8.

In the experiment shown in Figure 9, the start points of the worm signature are automatically generated by Algorithm 2 and grow gradually with time. Next, we manually set the initial immunization ratio (percentage of patched nodes) from 0% to 5% (preferential the guard nodes) and verify the infected ratio at 70 time units. We use a patching threshold μ (initially infected ratio) to decide when to initiate the patching process. The experiments are conducted with $\mu = 5\%$, 10%, 15%, and 20% (beyond 20%, the worm becomes uncontrollable), and the results are described in Figures 10 and 11.

In both cases, CC^2 performs better than the other three containment schemes under the same patching cost. For example, CC^2 is 20.1%, 13.2%, 10.2%, and 7.7% higher than proximity, centralize, socialize, and TC-based on Haggle networks when $\mu = 5\%$. This advantage is even more pronounced when $\mu = 20\%$, with the gap becoming 35.7%, 16.1%, 12.1%, and 10.4%, respectively. We also conclude four observations: (1) the later the worm is discovered (i.e., the higher the threshold μ is), the more difficult it is to control; (2) the control of the worm is proportional to network sparsity; (3) increasing the delivery priority of high-impact nodes can significantly improve worm containment

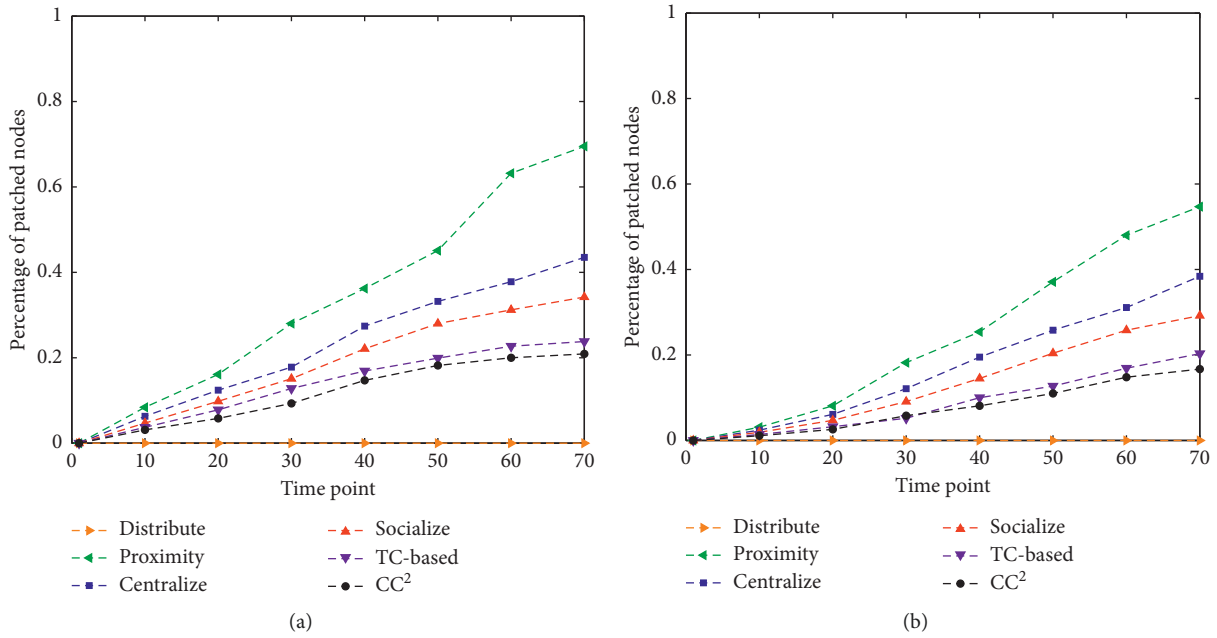

Figure 9: Performance comparison of the percentage of patched nodes on Haggle (a) and MIT Reality (b).

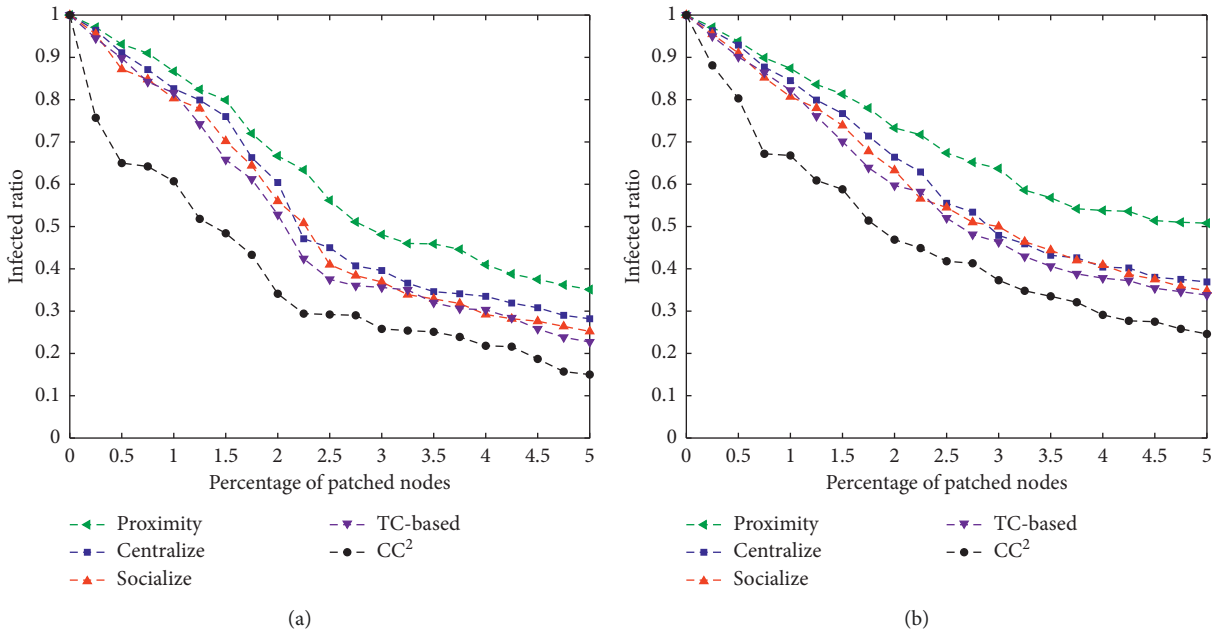

Figure 10: Continued.

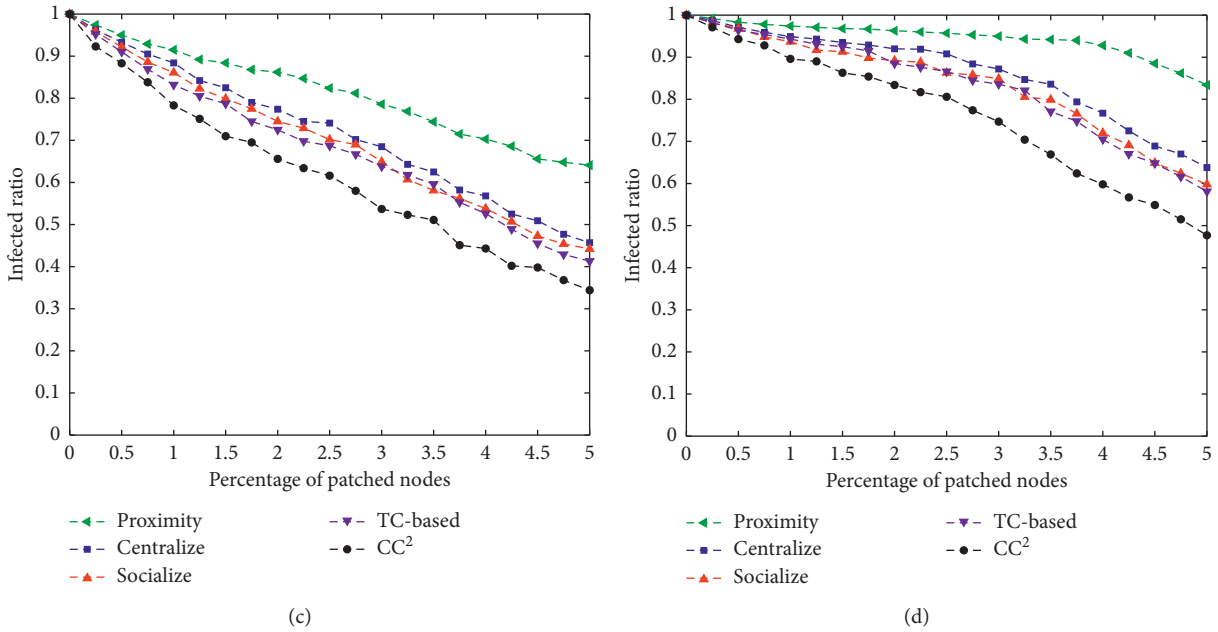

Figure 10: Performance comparison of different patching strategy on the Haggle network with an initially infection ratio μ = 5% (a), 10% (b), 15% (c), and 20% (d), respectively.

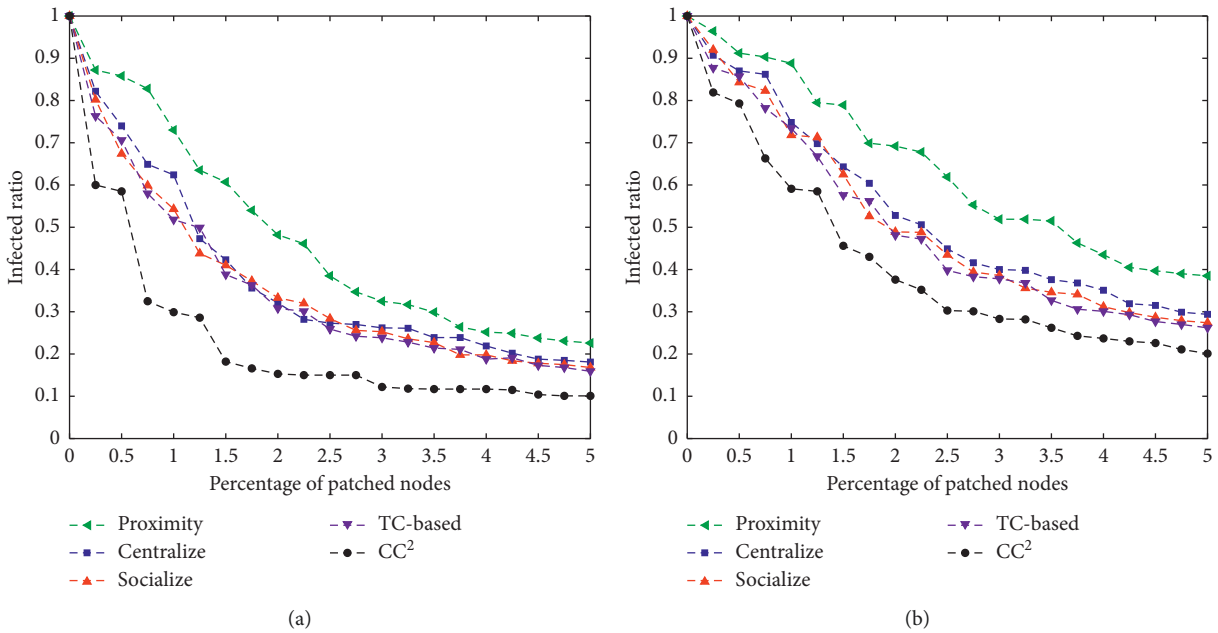

Figure 11: Continued.

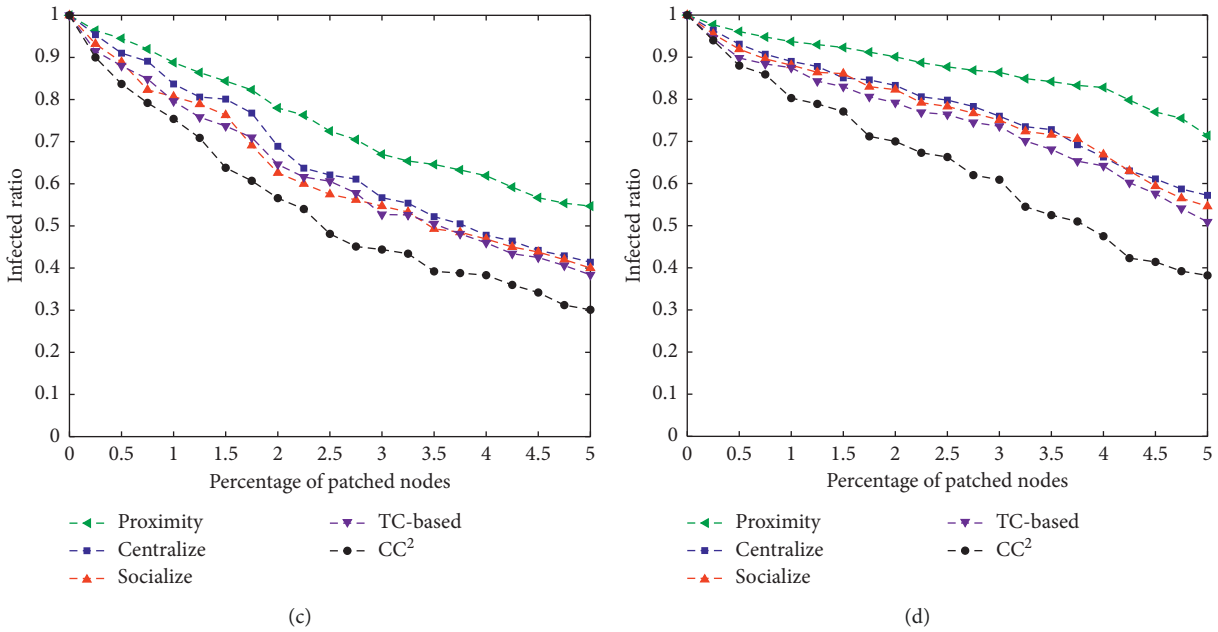

Figure 11: Performance comparison of different patching strategy on the MIT Reality network with an initially infected ratio μ = 5% (a), 10% (b), 15% (c), and 20% (d), respectively.

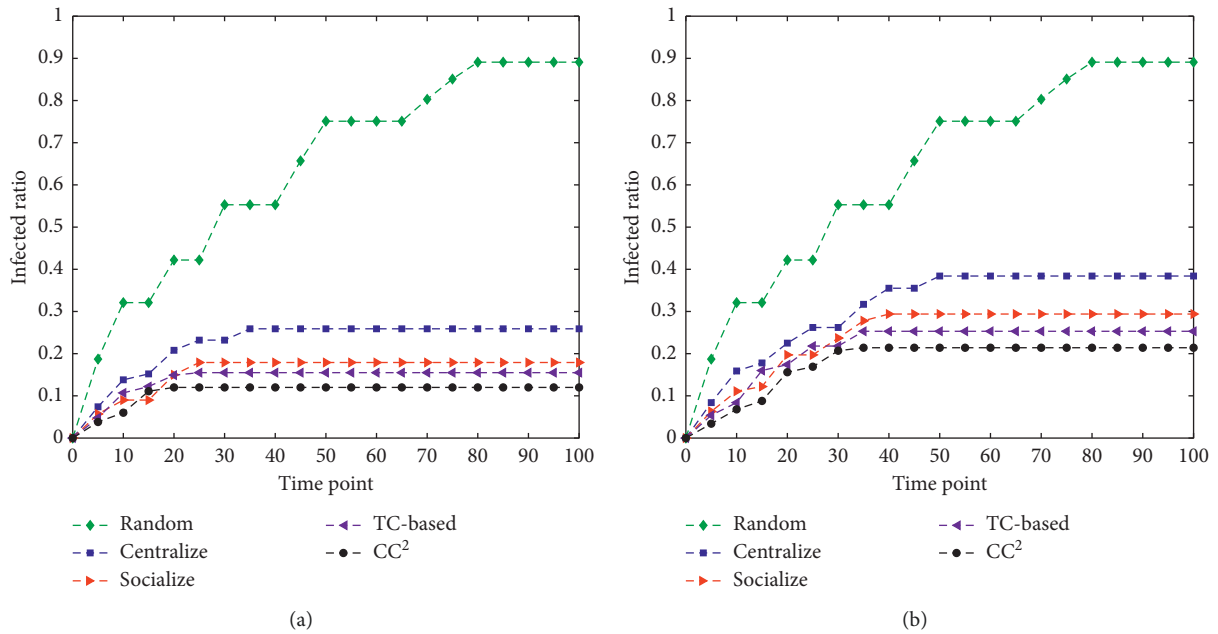

Figure 12: Continued.

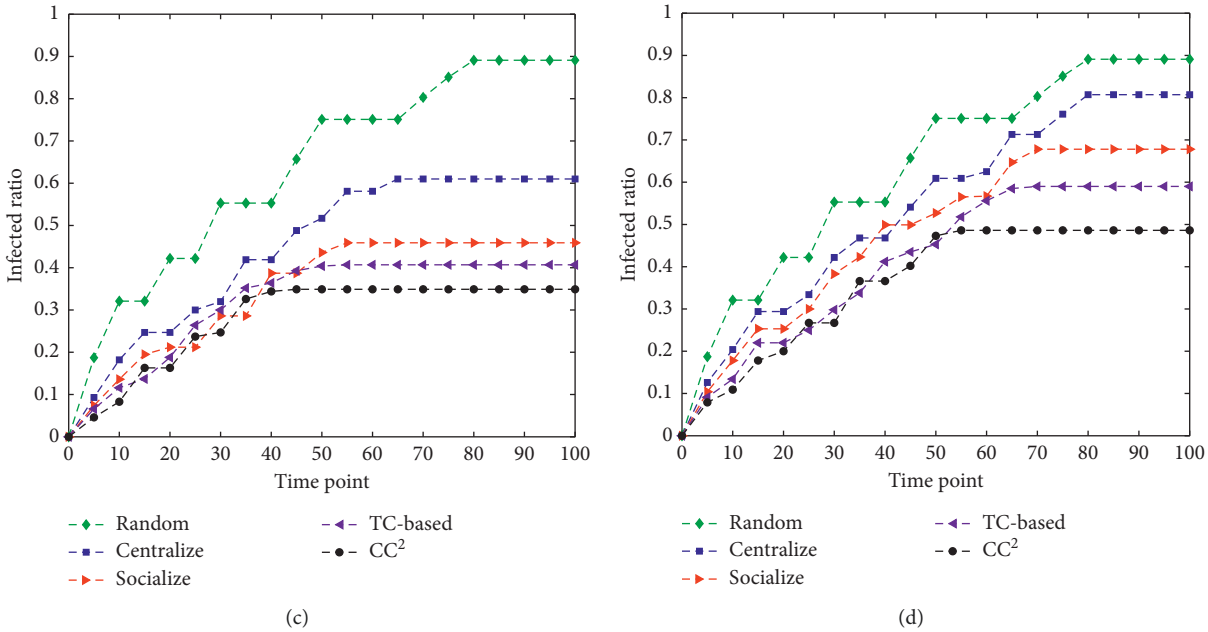

Figure 12: Performance comparison of different coping scheme with 5% patching nodes on the Haggle network with initially infected ratio μ = 5% (a), 10% (b), 15% (c), and 20% (d), respectively.

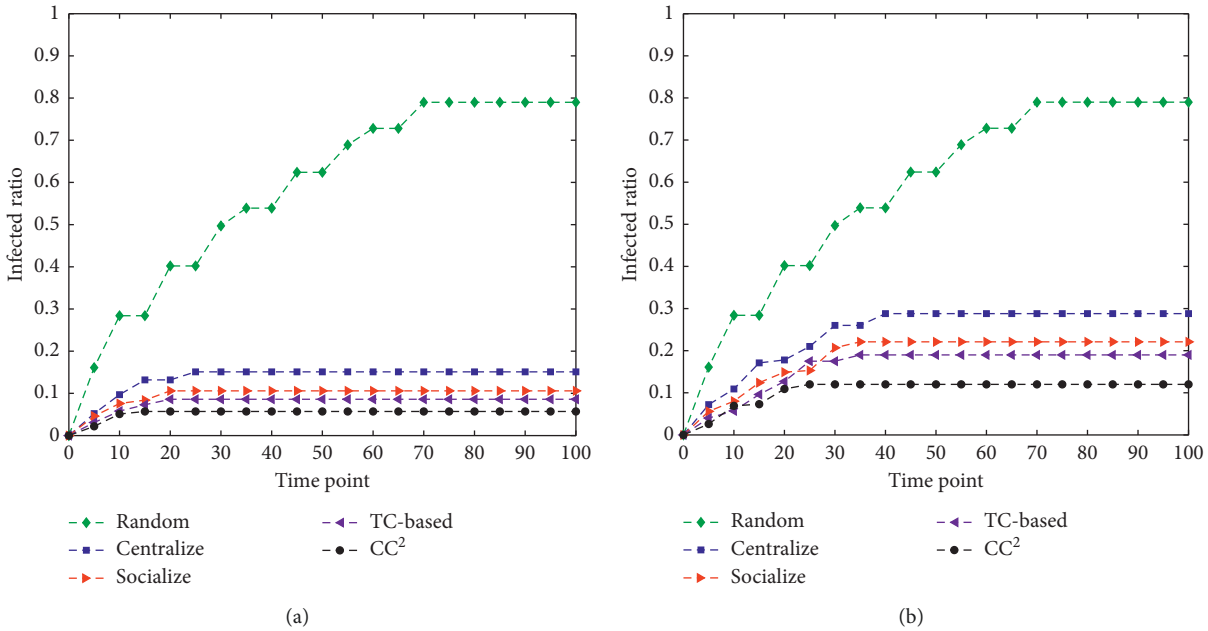

Figure 13: Continued.

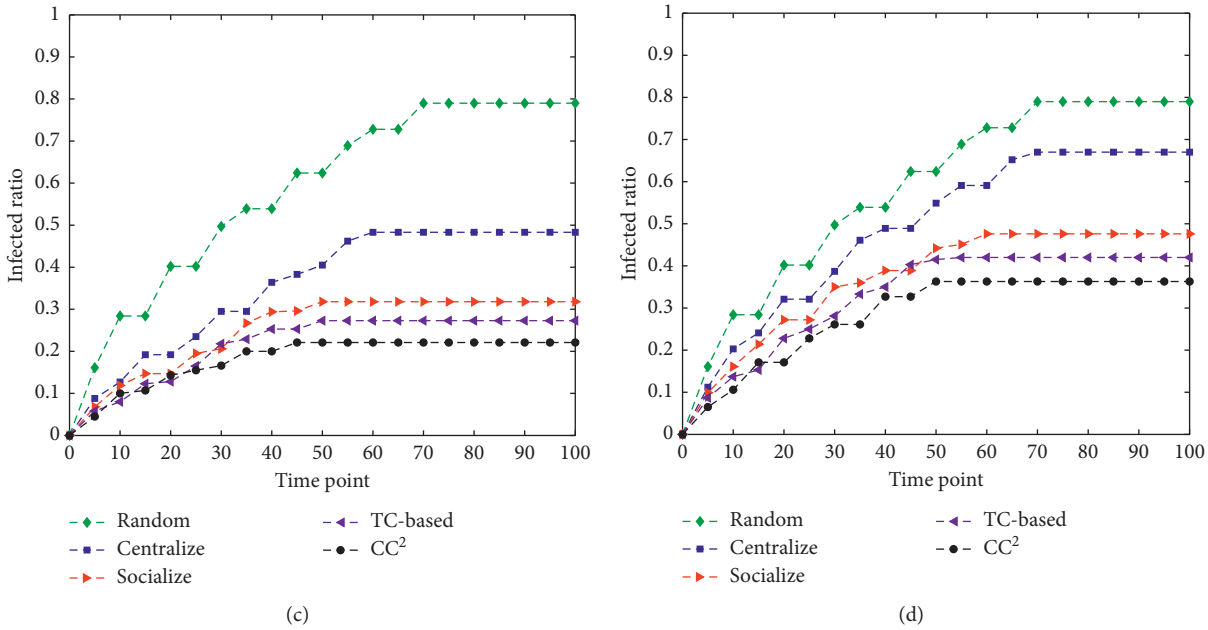

Figure 13: Performance comparison of different coping scheme with 5% patching nodes on the MIT Reality network with initially infected ratio μ = 5% (a), 10% (b), 15% (c), and 20% (d), respectively.

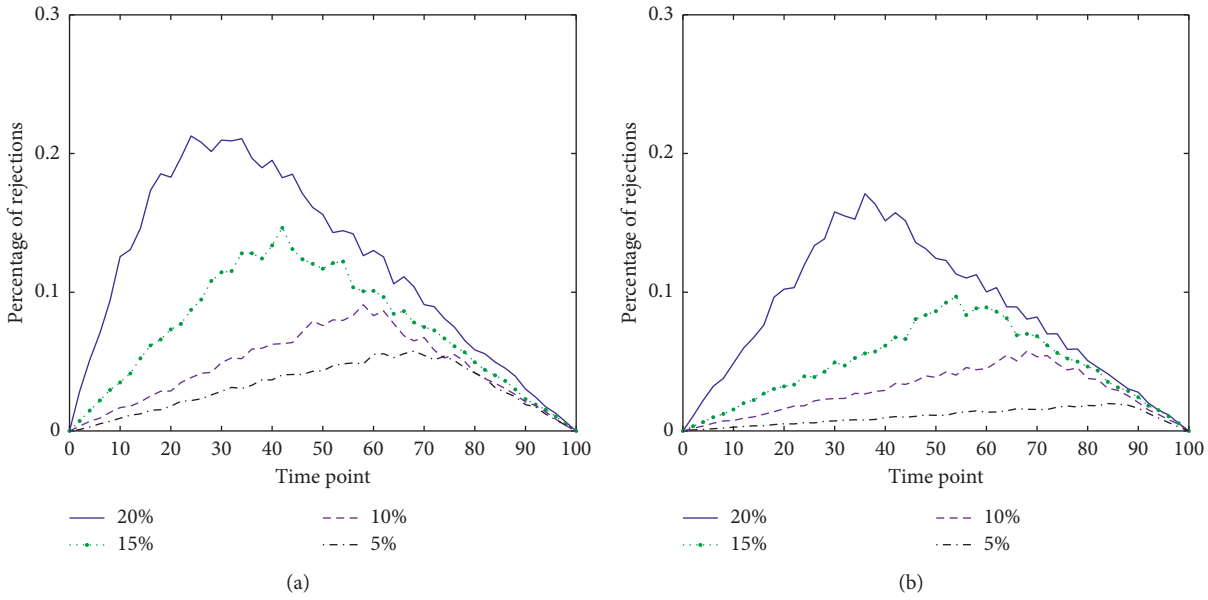

Figure 14: Percentage of rejections with initially infected ratio μ = 5%, 10%, 15%, and 20%: Haggle (a) and MT Reality (b).

performance; and (4) when a critical point is reached (such as 2.5% and 1.5% in Figures 10 and 11 when $\mu = 5\%$), further increasing the number of patched nodes will not effectively reduce the infected ratio.

The third experiment in this part verified the infected ratio with time when μ = 5%, 10%, 15%, and 20%. The comparison methods include (1) random patching (repeated 100 times for consistency), (2) centralize (short-range containment), (3) socialize (long-range containment), and (4) TC-based (community-based coping scheme). To make the results more explicit, we set the initial percentage of patched nodes to 5% (performs well in the previous simulation), and the results are shown in Figures 12 and 13.

During the experiment, the random patching strategy is not sufficient to control the spread of worms. Even for the MIT Reality dataset with a sparse topology, the infected ratio eventually stabilized at 79.1% (around 70 time units). For the Haggle dataset, the value rises to 89.1% (around 80 time units). As expected, the later the worm is discovered, the higher the infected ratio. In essence, the worm signature is

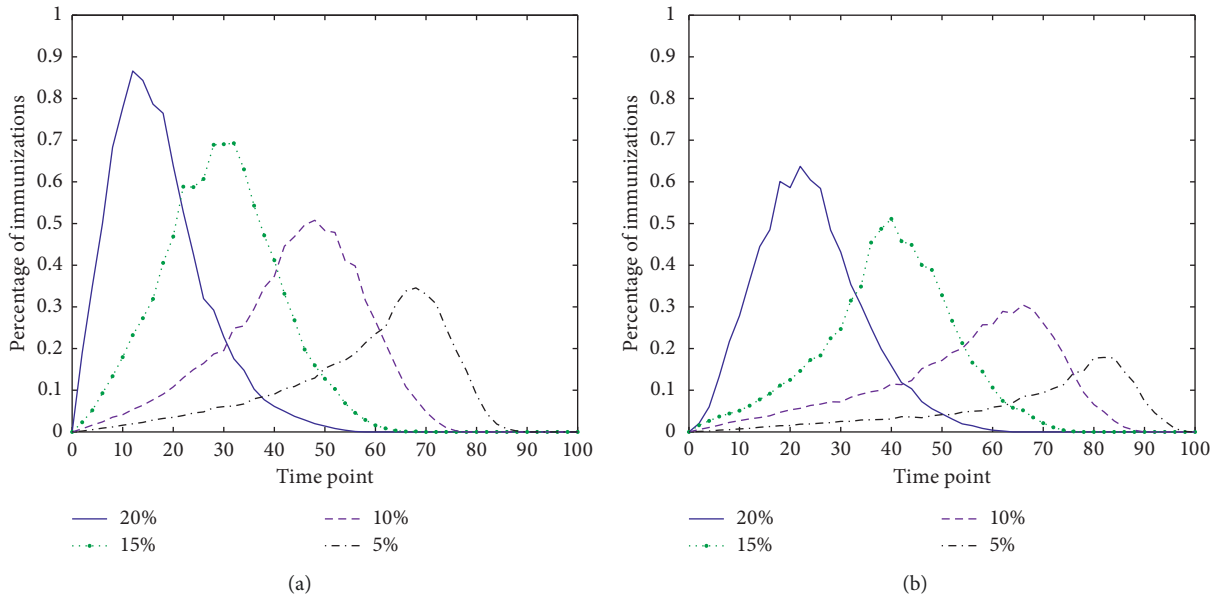

Figure 15: Percentage of immunizations with initially infected ratio μ = 5%, 10%, 15%, and 20%: Haggle (a) and MT Reality (b).

"racing" against the worm. If the worm signature "runs" faster, the containment performance will be better. On both datasets, when $\mu \leq 15\%$, centralize, socialize, TC-based, and CC^2 can control the infected ratio below 50% (around 40 time units). The reason is that the active containment scheme delivers the worm signatures in parallel, while the worm can only randomly move around. When $\mu = 20\%$, only CC^2 can effectively control the further spread of the worm and the network reaches a stable state around 50 time units.

6.5. The Analysis on Rejection and Immunization Number. Finally, we briefly analyze the immunization and rejection strategies of CC^2. We focus on the percentage of rejections and immunizations with initial infected ratio μ = 5%, 10%, 15%, and 20% during the iteration. The results are shown in Figures 14 and 15. On average, the number of immunizations is higher than the number of rejections, suggesting that the immunization strategy dominates the worm containment process. This conclusion will be more conspicuous when the link density becomes larger (i.e., the Haggle network). We also noticed that when the number of immunizations and rejections decayed from the peak, the infected ratio is still at a high rate of growth (compared to Figures 12 and 13), which is related to the two-stage propagation pattern of the worm. The first stage is the "strong-link infection" of top ten mutual friends, which makes the number of attacks reach its peak quickly. The second stage is the "weak-link infection" of nontop ten friends. In this mode, the probability of a node being infected is proportional to the degree of the node. Once a high-degree node is infected, it triggers a broader range of secondary infections, which continues to increase the infected ratio. From this perspective, the idea of priority immunization against high-impact nodes is practical and has certain performance advantages. Besides, the immunization strategy does not affect the normal communication of the network (e.g., the peak is 21.3% and 17.1% when μ = 20% in Figure 14), which ensures the channel utilization of the network to a certain extent.

7. Conclusion

This paper models the spreading dynamics of the hybrid worm and propose several designs for worm containment on the mobile Internet. First, we identify the community structure of the network and control the long-range infection by immunizing the guard nodes that lie on the periphery of each community. Second, a worm signature distribution scheme is presented to quickly deliver each signature to all nodes to prevent the worm from spreading out to a larger population. Third, we design a security evaluation method to help users make favorable decisions when exchanging data using short-range transmission interfaces.

Experimental results show that the key to SINs worm containment is to identify high-impact nodes in the network accurately. Besides, although the GINs worm spread slowly, it can significantly improve the propagation performance of hybrid infections. The method has high performance and efficiency in the location-based social networks. Subsequent research will focus on the multilayer network worm containment task to achieve an accurate perception of the real world.

Data Availability

The data used to support the findings of this study are available from the corresponding author upon request. The

data can also be downloaded from the URLs given in Section 6.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This work was sponsored by the National Natural Science Foundation of China (61402126 and 61602133), Nature Science Foundation of Heilongjiang Province of China (F2016024), Heilongjiang Postdoctoral Science Foundation (LBH-Z15095), University Nursing Program for Young Scholars with Creative Talents in Heilongjiang Province (UNPYSCT-2017094), Scientific Research Foundation for the Overseas Returning Person of Heilongjiang Province of China (LC2018030), and National Training Programs of Innovation and Entrepreneurship for Undergraduates (201810214020).

References

[1] S. Peng, S. Yu, and A. Yang, "Smartphone malware and its propagation modeling: a survey," *IEEE Communications Surveys & Tutorials*, vol. 16, no. 2, pp. 925–941, 2014.

[2] S.-M. Cheng, W. C. Ao, P.-Y. Chen, and K.-C. Chen, "On modeling malware propagation in generalized social networks," *IEEE Communications Letters*, vol. 15, no. 1, pp. 25–27, 2011.

[3] P. Kim and C. H. Lee, "Epidemic spreading in complex networks with resilient nodes: applications to fmd," *Complexity*, vol. 2018, Article ID 5024327, 9 pages, 2018.

[4] G. Zyba, G. M. Voelker, M. Liljenstam, A. Méhes, and P. Johansson, "Defending mobile phones from proximity malware," in *Proceedings of the IEEE INFOCOM 2009—The 28th Conference on Computer Communications*, pp. 1503–1511, IEEE, Rio de Janeiro, Brazil, April 2009.

[5] F. Li, Y. Yang, and J. Wu, "CPMC: an efficient proximity malware coping scheme in smartphone-based mobile networks," in *Proceedings of the IEEE INFOCOM 2010*, pp. 1–9, IEEE, San Diego, CA, USA, March 2010.

[6] Y. Wang, S. Wen, Y. Xiang, and W. Zhou, "Modeling the propagation of worms in networks: a survey," *IEEE Communications Surveys & Tutorials*, vol. 16, no. 2, pp. 942–960, 2014.

[7] X. Meng, P. Zerfos, V. Samanta, S. H. Y. Wong, and S. Lu, "Analysis of the reliability of a nationwide short message service," in *Proceedings of the IEEE INFOCOM 2007—26th IEEE International Conference on Computer Communications*, pp. 1811–1819, IEEE, Barcelona, Spain, May 2007.

[8] Z. Zhu, G. Cao, S. Zhu, S. Ranjan, and A. Nucci, "A social network based patching scheme for worm containment in cellular networks," in *Proceedings of the IEEE INFOCOM 2009—The 28th Conference on Computer Communications*, pp. 1476–1484, IEEE, Rio de Janeiro, Brazil, April 2009.

[9] X. Zhang and G. Cao, "Transient community detection and its application to data forwarding in delay tolerant networks," *IEEE/ACM Transactions on Networking*, vol. 25, no. 5, pp. 2829–2843, 2017.

[10] P.-Y. Chen and A. O. Hero, "Multilayer spectral graph clustering via convex layer aggregation: theory and algorithms," *IEEE Transactions on Signal and Information Processing Over Networks*, vol. 3, no. 3, pp. 553–567, 2017.

[11] C. C. Zou, D. Towsley, and W. Gong, "Modeling and simulation study of the propagation and defense of internet e-mail worms," *IEEE Transactions on Dependable and Secure Computing*, vol. 4, no. 2, pp. 105–118, 2007.

[12] J. Su, K. K. W. Chan, A. G. Miklas et al., "A preliminary investigation of worm infections in a bluetooth environment," in *Proceedings of the 4th ACM workshop on Recurring malcode*, pp. 9–16, ACM, Alexandria, VA, USA, 2006.

[13] G. Yan and S. Eidenbenz, "Modeling propagation dynamics of bluetooth worms (extended version)," *IEEE Transactions on Mobile Computing*, vol. 8, no. 3, pp. 353–368, 2009.

[14] J. W. Mickens and B. D. Noble, "Modeling epidemic spreading in mobile environments," in *Proceedings of the 4th ACM Workshop on Wireless Security*, pp. 77–86, ACM, Cologne, Germany, September 2005.

[15] J. R. Morris-King and H. Cam, "Controlling proximity-malware infection in diverse tactical mobile networks using K-distance pruning," in *Proceedings of the IEEE MILCOM 2016—2016 IEEE Military Communications Conference*, pp. 503–508, Baltimore, MD, USA, November 2016.

[16] Y. Yang, S. Zhu, and G. Cao, "Improving sensor network immunity under worm attacks: a software diversity approach," in *Proceedings of the 9th ACM International Symposium on Mobile Ad Hoc Networking and Computing*, pp. 149–158, ACM, Hong Kong, China, May 2008.

[17] A. G. Miklas, K. K. Gollu, K. K. W. Chan, S. Saroiu, K. P. Gummadi, and E. De Lara, "Exploiting social interactions in mobile systems," in *Proceedings of the 9th International Conference on Ubiquitous Computing*, pp. 409–428, Springer-Verlag, Innsbruck, Austria, 2007.

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

[19] M. S. Haghighi, S. Wen, Y. Xiang, B. Quinn, and W. Zhou, "On the race of worms and patches: modeling the spread of information in wireless sensor networks," *IEEE Transactions on Information Forensics and Security*, vol. 11, no. 12, pp. 2854–2865, 2016.

[20] X. Sun, Z. Lu, X. Zhang, M. Salathe, and G. Cao, "Infectious disease containment based on a wireless sensor system," *IEEE Access*, vol. 4, pp. 1558–1569, 2016.

[21] B. Thompson, J. Morris-King, and R. Harang, "Slowing the spread of bluetooth-based malware in mobile tactical networks," in *Proceedings of the 2016 IEEE Military Communications Conference*, pp. 485–490, Baltimore, MD, USA, November 2016.

[22] O. Trullols-Cruces, M. Fiore, J. M. Barcelo-Ordinas, and Barcelo-Ordinas, "Worm epidemics in vehicular networks," *IEEE Transactions on Mobile Computing*, vol. 14, no. 10, pp. 2173–2187, 2015.

[23] Q. Zhang and A. Boukerche, "A novel infrastructure-based worm spreading countermeasure for vehicular networks," *IEEE Transactions on Intelligent Transportation Systems*, vol. 19, no. 7, pp. 2188–2203, 2018.

[24] A. Boukerche and Q. Zhang, "Countermeasures against worm spreading," *ACM Computing Surveys*, vol. 52, no. 2, pp. 1–25, 2019.

[25] C. Fleizach, M. Liljenstam, P. Johansson, G. M. Voelker, and A. Mehes, "Can you infect me now?: malware propagation in mobile phone networks," in *Proceedings of the 2007 ACM Workshop on Recurring Malcode*, pp. 61–68, ACM, Alexandria, VA, USA, November 2007.

[26] A. Bose, X. Hu, K. G. Shin, and T. Park, "Behavioral detection of malware on mobile handsets," in *Proceedings of the 6th International Conference on Mobile Systems, Applications, and Services*, pp. 225–238, ACM, Breckenridge, CO, USA, June 2008.

[27] D. Zhao, L. Wang, Z. Wang, and G. Xiao, "Virus propagation and patch distribution in multiplex networks: modeling, analysis, and optimal allocation," *IEEE Transactions on Information Forensics and Security*, vol. 14, no. 7, pp. 1755–1767, 2019.

[28] Lu-X. Yang and X. Yang, "A novel virus-patch dynamic model," *PLoS One*, vol. 10, no. 9, Article ID e0137858, 2015.

[29] Z. Lu, X. Sun, Y. Wen, G. Cao, and T. L. Porta, "Algorithms and applications for community detection in weighted networks," *IEEE Transactions on Parallel and Distributed Systems*, vol. 26, no. 11, pp. 2916–2926, 2015.

[30] F. Taghavian, M. Salehi, and M. Teimouri, "A local immunization strategy for networks with overlapping community structure," *Physica A: Statistical Mechanics and Its Applications*, vol. 467, pp. 148–156, 2017.

[31] S. Peng, M. Wu, G. Wang, and S. Yu, "Containing smartphone worm propagation with an influence maximization algorithm," *Computer Networks*, vol. 74, pp. 103–113, 2014.

[32] W. Yang, H. Wang, and Y. Yao, "An immunization strategy for social network worms based on network vertex influence," *China Communications*, vol. 12, no. 7, pp. 154–166, 2015.

[33] D. J. Daley and J. M. Gani, *Epidemic Modelling: An Introduction*, Vol. 15, Cambridge University Press, Cambridge, UK, 2001.

[34] P. De, Y. Liu, and S. K. Das, "An epidemic theoretic framework for vulnerability analysis of broadcast protocols in wireless sensor networks," *IEEE Transactions on Mobile Computing*, vol. 8, no. 3, pp. 413–425, 2009.

[35] J. Xie, S. Kelley, and B. K. Szymanski, "Overlapping community detection in networks: the state-of-the-art and comparative study," *ACM Computing Surveys*, vol. 45, no. 4, p. 43, 2013.

[36] U. N. Raghavan, R. Albert, and S. Kumara, "Near linear time algorithm to detect community structures in large-scale networks," *Physical Review E*, vol. 76, no. 3, p. 036106, 2007.

[37] M. E. J. Newman, "Modularity and community structure in networks," *Proceedings of the National Academy of Sciences*, vol. 103, no. 23, pp. 8577–8582, 2006.

[38] N. A. Christakis and J. H. Fowler, "Social network sensors for early detection of contagious outbreaks," *PLoS One*, vol. 5, no. 9, p. e12948, 2010.

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

[40] S. Gregory, "Finding overlapping communities in networks by label propagation," *New Journal of Physics*, vol. 12, no. 10, p. 103018, 2010.

[41] H. Shen, X. Cheng, K. Cai, and M.-B. Hu, "Detect overlapping and hierarchical community structure in networks," *Physica A: Statistical Mechanics and Its Applications*, vol. 388, no. 8, pp. 1706–1712, 2009.

[42] T. N. Dinh, N. P. Nguyen, and M. T. Thai, "An adaptive approximation algorithm for community detection in dynamic scale-free networks," in *Proceedings of the IEEE INFOCOM*, pp. 55–59, IEEE, Turin, Italy, April 2013.