This full text paper was peer reviewed at the direction of IEEE Communications Society subject matter experts for publication in the IEEE INFOCOM 2010 proceedings
This paper was presented as part of the main Technical Program at IEEE INFOCOM 2010.

CPMC: An Efficient Proximity Malware Coping Scheme in Smartphone-based Mobile Networks

Feng Li
School of Engineering and Technology
IUPUI
Indianapolis, IN 46202

Yinying Yang
Dept. of Comp. Science and Engineering
Florida Atlantic University
Boca Raton, FL 33431

Jie Wu
Dept. of Comp. and Info. Sciences
Temple University
Philadelphia, PA 19122

***Abstract*—Smartphones are envisioned to provide promising applications and services. At the same time, smartphones are also increasingly becoming the target of malware. Many emerging malware can utilize the proximity of devices to propagate in a distributed manner, thus remaining unobserved and making detections substantially more challenging. Different from existing malware coping schemes, which are either totally centralized or purely distributed, we propose a Community-based Proximity Malware Coping scheme, CPMC. CPMC utilizes the social community structure, which reflects a stable and controllable granularity of security, in smartphone-based mobile networks. The CPMC scheme integrates short-term coping components, which deal with individual malware, and long-term evaluation components, which offer vulnerability evaluation towards individual nodes. A closeness-oriented delegation forwarding scheme combined with a community level quarantine method is proposed as the short-term coping components. These components contain a proximity malware by quickly propagating the signature of a detected malware into all communities while avoiding unnecessary redundancy. The long-term components offer vulnerability evaluation towards neighbors, based on the observed infection history, to help users make comprehensive communication decisions. Extensive real- and synthetic-trace driven simulation results are presented to to evaluate the effectiveness of CPMC.**

***Index Terms*—Granularity of security, mobile networks, proximity malware, signature, social network analysis, vulnerability.**

I. Introduction

Malware [1] has become the major impediment in the development of networks. Any new type of network that provide promising applications always becomes the main target of new malware. Among the recent viruses and worms, those propagating in a distributed manner and without central control are the hardest to defend against. In the last year, the rapid outbreak of the Confilcker worm [2], which propagates updates in a distributed peer-to-peer way, clearly indicates the difficulty and importance of coping with distributed malware.

Mobile devices have increasingly penetrated work and family life. With the emergence of powerful new devices, such as the Blackberry, iPhone, and Palm Treo, the smartphone-based mobile network is considered to be a promising branch for the next generation networks. Many revolutionary applications, such as opportunistic podcasting [3], have been proposed for this type of network. The distributed nature of these networks gives the malware the opportunity to propagate through direct pair-wise communications, i.e. bluetooth or Wi-Fi, between nodes in geographical proximity. A typical proximity malware propagation process is shown in Fig. 1. Several newly reported families of worms, including Commwarrior, Cabir, and Lasco [4], belong to the proximity malware category. These worms can easily persist in the network and remain undetected because of the decentralized infection and the dynamic topology. Moreover, since heterogeneous devices exist in the networks, only a portion of the smartphones will have malware detection capabilities. The threats of proximity malware are immediate and rapidly growing due to these networks' growing popularity and convenience.

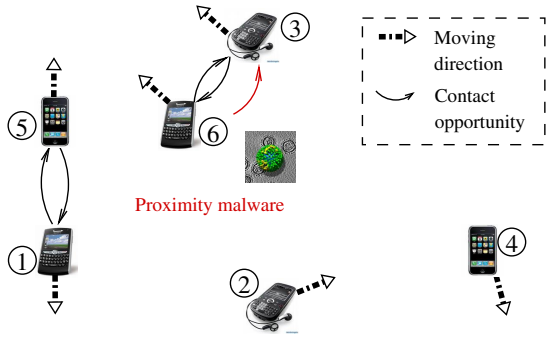


Fig. 1. Proximity malware propagation. Malware can propagate through bluetooth connections when two nodes are in geographical proximity.

Most of the existing malware coping schemes, such as the approaches in response to the MMS/SMS malware [5], [6], utilize centralized methods implemented in the provider network. However, these centralized defense methods are only effective towards malware propagated through the centralized infrastructure. Defending against proximity malware, however, poses very different challenges, since the absence of the provider network and the highly dynamic topology blurs the possible line of defense. In [7], Zyba *et al.* present a purely distributed coping scheme, which allows nodes to detect malware and flood the signature in the network to eliminate the proximity malware. This kind of method also shows limitations: each node's own view is too limited, and the signature flooding is too costly. Since both the centralized and purely distributed schemes need to be improved, we intend to find a novel and suitable granularity in between to facilitate

978-1-4244-5837-0/10/$26.00 ©2010 IEEE

This full text paper was peer reviewed at the direction of IEEE Communications Society subject matter experts for publication in the IEEE INFOCOM 2010 proceedings
This paper was presented as part of the main Technical Program at IEEE INFOCOM 2010.

the malware coping task.

Mobile smartphones are personal in nature, i.e. they stay and travel together with one person most of the time, and thus enter various social contexts of that person. Therefore, nodes' movements in these networks are usually repetitive to a certain extent, and bear the social network properties of their owners. We exploit these social network properties in proposing the coping scheme. In these networks, metrics based on the contact history are usually helpful in depicting the level of closeness among nodes. Ina addition, the social grouping structure [8], [9], [10] has been observed in multiple sets of real Smartphone-based mobile network traces, such as Haggle [11] and Reality Mining [12]. Although the instant topology of the mobile networks is transient, the grouping structure tends to be stable over time.

These observations provide us the ground to offer malware coping strategies at a new granularity in the mobile networks. Based on the concept of community, which is a clique-based social structure that represents the controllable granularity, we propose the **C**ommunity-based **P**roximity **M**alware **C**oping scheme (CMPC for short). It contains two kinds of components, short-term coping and long-term evaluation components, both aiming to fully utilize the structural advantages.

The proximity malware usually adopts a flooding scheme to propagate. To beat a particular malware, the signature of the malware, which will intrigue the malware to be removed on the received nodes, needs to be propagated fast and efficiently. In the short-term component, we combine the efficient signature propagation with a community quarantine. The signature will only be further propagated by nodes that have some social properties to be the better delegates to propagate it in more communities. When a signature reaches a community, all nodes in this communication will reject communications from the original community for a short period of time. Besides the short-term components to coping with one particular type of proximity malware, long-term evaluation components are proposed to measure the vulnerability of each node. Since nodes in the mobile network tend to have different security settings and policies, as well as distinct neighbors, they will appear to be different in vulnerability towards a new type of proximity malware. We first develop a way for nodes to evaluate the vulnerability of their direct neighbors. Because each node's own view is very limited, we propose a consensus scheme that utilizes the information collected from a community to calculate the vulnerability value.

The contributions of this paper are three-fold. First, we exploit a new granularity of security, which is based on the stable neighboring relationships and community structure, to guide the design of the malware coping scheme. Second, instead of the flooding mechanism, we develop an efficient distributed signature propagation process, and combine it with the community quarantine process to make the signature propagate faster than the malware. Third, we develop the vulnerability evaluation scheme to link a node's past history with future predictions, and enlarge each node's view of vulnerability evaluation from community consensus.

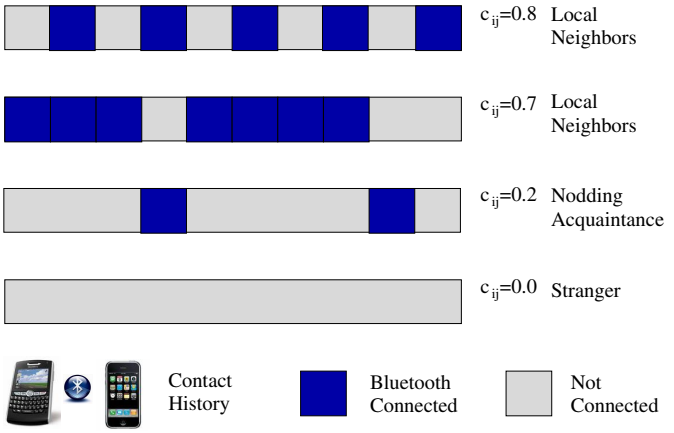


Fig. 2. Relationship abstraction. The bar graph represents the bluetooth contact history between two smartphones. Notions, including local neighbors, nodding acquaintances, and strangers, intuitively depict the corresponding closeness.

II. Preliminary

We focus on using the short range communication capability of the smartphones to form the mobile networks. We assume that a store-and-forward style scheme, which belongs to the main stream forwarding schemes [13], [14] in this kind of challenging environment, is adopted to deliver packets via intermittently connected nodes.

A. Relationship abstraction

In mobile networks, each node can record the encounter time and duration whenever it meets another node. Nodes' original knowledge, which includes both temporal and spacial information, can be abstracted into a single *closeness* metric $c_{ij} \in [0, 1]$ for nodes i and j. Several examples of closeness abstractions are shown in Fig. 2.

To measure c_{ij}, the average separation period $\bar{D}_{ij}$ is measured during a training time window before the abstraction starts. $\bar{D}_{ij}$ is defined as the total separation time between i and j divided by the number of separations during the training window. For example, the $\bar{D}_{ij}$ of the first case in Fig. 2 is 1 unit. $\bar{D}_{ij}$ is a comprehensive metric to start the time-space abstraction since it reflects both the frequency and length of the encounters. Smaller $\bar{D}_{ij}$ indicates shorter communication latency between i and j. We apply the Gaussian similarity function [15] to normalize $\bar{D}_{ij}$ as follows and denote the resulting metric as closeness c_{ij}:

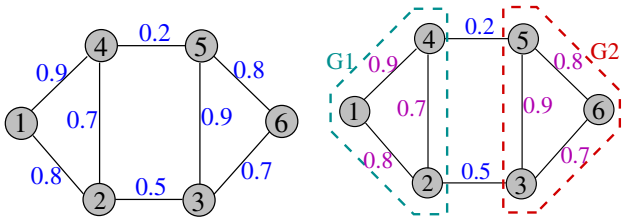
$$c_{ij} = \exp(-\frac{(\bar{D}_{ij})^2}{2\sigma^2}). \quad (1)$$

Here, σ is a scaling parameter [15] for the separation period.

B. Social group identification

Many recent studies [16], [17], [18] based on real mobile traces reveal that the mobile network shows certain social network properties. They apply social network analysis in mobile networks without [16] or with communities [8]. In [8], Hui *et al.* analyze k-clique community structure from mobility

This full text paper was peer reviewed at the direction of IEEE Communications Society subject matter experts for publication in the IEEE INFOCOM 2010 proceedings
This paper was presented as part of the main Technical Program at IEEE INFOCOM 2010.



(a) Before (b) After

Fig. 3. Community formation. Dotted lines enclosed the formed community.

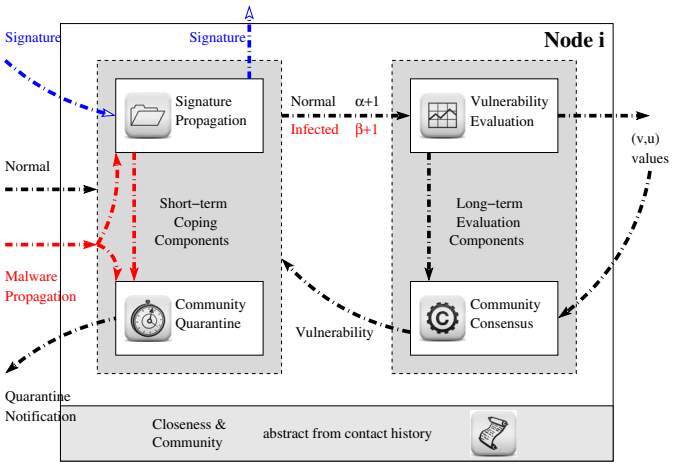


Fig. 4. CPMC scheme overview.

traces and use them for forwarding algorithms. In [17], [18], communities are organized based on location preferences of nodes. In [9], [10], we propose to use the closeness-based community to reflect the locality in mobile networks. The closeness locality concept is suitable for the definition of controllable granularity for security.

In graph theory, a *clique* is a subgraph in which every vertex is connected to every other vertex in the graph. We extend the concept in the CMPC scheme and define a *community* in the mobile networks as follows: For any pair of nodes in the community, a link exists such that the closeness of the link is larger than the threshold value T. As a criterion to determine whether the relationship between two nodes is strong enough to be considered as in the same group, we adopt an adjustable threshold T. If $c_{ij} > T$, we consider nodes i and j potentially members of the same community. One example with $T = 0.6$ is shown in Fig. 3. The non-overlapped community structure can be constructed in a distributed manner using a simplified clique formation algorithm (one example is proposed in our previous work [9]). Each node only needs two hops of local information in the community construction.

III. CPMC Main Scheme

The CPMC scheme integrates both the short-term and long-term components, on top of the social community, to cope with the proximity malware comprehensively.

The short-term coping components, shown in Fig. 4 on the left, including signature propagation and community quarantine, deal with each individual proximity malware. The short-term coping components classify each round of communication as normal (black arrows in Fig. 4) or malware infected (red arrows in Fig. 4). If malware infected, community quarantine starts and a signature (blue arrow on the top of Fig. 4) will be generated and propagated. The short-term coping components also decide whether to accept or reject one round of communication, based on the feedback of the long-term evaluation components (black arrow in the middle of Fig. 4), when in proximity of a particular node.

The long-term evaluation components, shown in Fig. 4 on the right, provide nodes with comprehensive information based on others' history security performances. They also provide the incentive for nodes to enforce strong security policy. We introduce the concept of vulnerability, and evaluate a node's vulnerability based on the results gathered from the short-term components, which are the α and β in Fig. 4. We also offer a method to draw consensus from the opinions originated from nodes in the same community. The community consensus expends an individual node's view and improves the evidence sufficiency. The vulnerability consensus from the long-term components will be considered as the feedback to the short-term components (black arrow in the middle of Fig. 4), and guide the future decisions made by the short-term components.

The short-term and long-term components are closely combined to utilize the designed granularity of security reflected by the community. Both types of components are constructed on top of the relationship-based community structure abstracted from community history, as shown in the bottom of Fig. 4.

A. Short-term coping components

The short-term coping components deal with each individual type of proximity malware. A node in the mobile networks decides whether to reject a communication request based on the result of the short-term coping component.

Signature propagation component. This component is designed to quickly propagate the signature to all the communities without the cost of flooding. A signature will be delegated to nodes that can quickly propagate it to more communities, i.e. if a current delegate i of a signature encounters node j, and j is closer, in terms of social relationship, to one or more communities than i and the nodes that i met before, i should forward the signature to j and allow j to further propagate the signature.

A malware signature consists of the summarized malicious patterns in the malware, which can be included in an alert or a patch. If a node receives the signature before it is infected by a proximity malware, it will become immune towards the specific malware. However, the heterogeneous equipments and distributed environment in the smartphone-based mobile networks decide that only a portion of the devices will have the capability to detect and generate the signature of a proximity

Algorithm 1 Signature delegation forwarding

```
1: Node i encounters node j;
2: for each signature in i's buffer do
3:    Examine {F_1, ... F_X, ... F_Y};
4:    if ∃ X that satisfies c_j(X) > F_X then
5:       if node j never receives this signature before then
6:          Node i duplicates and delegates the signature to j;
7:       end if
8:       for each community Y do
9:          Set F_Y = max{c_j(Y), original F_Y} on nodes i and j;
10:      end for
11:   end if
12: end for
```

Algorithm 2 Community quarantine

```
1: Node i encounters node j that is in another community;
2: if Node i receives a new signature from node j then
3:    Node i starts timer Q_t and rejects communication (except
      signature) with nodes in X(j);
4:    Node i propagates a quarantine notice, which includes current
      Q_t, to nodes in X(i);
5: end if
```

malware. The signature will be generated only when a device i detects that an infected device j in i's proximity is trying to propagate the malware.

Inspired by [19], we propose the efficient community-based delegation scheme to propagate the signature after a signature has been generated on a node. In our community-based signature forwarding process, each copy of a signature maintains a set of forwarding thresholds $\{F_1, \dots F_X, \dots F_Y\}$ which is initialized as the quality of its original node, based on the closeness of the node to communities $\{1, \dots X, \dots Y\}$. For each community X, node i will use the value $c_i(X) = \max_{j \in X}\{c_{ij}\}$ to represent its closeness to the community. This value represents the node's best possible relationship with the community. For the community that node i belongs to, both $c_i(X)$ and F_X will be set to 1. If node i has no previous contact with a community X, $c_i(X)$ will be set to 0.

Whenever node i meets node j, node i delegates the signature to j if and only if the forwarding quality of node j towards one or more communities X exceeds the message's threshold F_X, i.e. $\exists$ community X, where $c_j(X) > F_X$ on i. Here, delegating means that node i will forward a copy of the signature to j and allow j to further propagate the copy. After that, the $\{F_1, \dots F_X, \dots F_Y\}$ of both copies in i and j are set to $F_X = \max\{c_j(X), \text{ original } F_X\}$. In the case that j's quality is better than the i's copy's threshold F but j already has the signature, the signature is not forwarded, but the F values of both copies on i and j will still be set to $F_X = \max\{i\text{'s original } F_X, j\text{'s original } F_X\}$.

The basic idea of this decision is that node i needs to check whether a potential forwarder j has better forwarding quality than node i, and all the delegates that node i met before. Node i will avoid delegating the signature to node j without a better quality. However, in this case, delegate i will still send the signature to j if j does not have a copy already, but j should not further forward the signature. The idea of the signature efficient propagation can be guaranteed through Algorithm 1.

Community quarantine component. This component extends the community concept in biological epidemiology and quarantines a community when a signature from it is received.

Quarantine is defined as the voluntary or compulsory isolation, typically to contain the spread of something dangerous. In existing proximity malware coping schemes [7], [20], when a node is found to be infected, it can immediately be isolated (quarantined) by the neighbors that detect the suspicious behavior. However, this node-oriented isolation is not sufficient in the smartphone-based mobile networks. Since the proximity malware is detected in a distributed manner, the node-oriented quarantine won't stop the malware from propagating to neighbors that cannot detect it. The key problem is deciding an appropriate scope for the quarantine. Since the community is a good reflection of locality in smartphone-based mobile networks, it is a natural candidate for the scope.

A community will enclose nodes that have close relationships, i.e. encounter more frequently or for longer durations. When one node is infected by a proximity malware, these close neighbors also have a larger chance to get contaminated. Algorithm 2 reflects our thoughts on the community quarantine.

In Algorithm 2, $X(i)$ represents the community X that node i belongs to. In this way, the signature propagation will further catch up with a malware's fast infection speed and constrain its propagation. To avoid the high cost of false alarms and interruptions of normal communication, our dynamic scheme will always lift after the timer Q_t expires.

B. Long-term evaluation components

The long-term evaluation components allow nodes to evaluate our nodes' vulnerability based on the security history, thus providing nodes with more information about others' history security performances as well as the incentive for nodes to enforce a strong security policy. The security history is collected from the short-term coping components. The results will guide the nodes in making future communication decisions.

Vulnerability evaluation component. This component aims to offer a rational way for a node i to link its observed security history towards node j with i's prediction of j's future behavior.

Vulnerability is originally defined as the susceptibility to physical or emotional injury or attack. In the smartphone-based mobile networks, we use vulnerability to refer to a node's state of being susceptible to proximity malware. The reasons for vulnerability of a node in the smartphone-based mobile networks include loose security policy, high risk communications, and sensitive stored information. We aim to quantify the vulnerability based on nodes' security histories.

For each round of communication between two nodes i and j, a result of whether a proximity malware is involved can be determined by the short-term coping components. Therefore,

a node i can accumulate its observation results towards node j according to the feedback of the short-term components. In our system, we use α to represent the total number of observed normal communications, and β to represent the total number of observed infections from known proximity malware. When there is no observation, the initial value of both α and β is 1.

We use Bayesian inference to reason from observation results to the vulnerability metric. Bayesian inference is statistical inference in which evidence or observations are used to update or to newly infer the probability that a hypothesis may be true. According to Bayesian inference, the vulnerability metric can be quantified using the portion of evidence that supports the claim that a node is vulnerable. Therefore, node i will calculate node j's vulnerability metric v as:

$$v = \frac{\beta}{(\alpha + \beta)}. \tag{2}$$

The evaluation is limited by i's own observation towards node j. The v metric gives nodes the hint about other nodes' vulnerability. However, there is another dimension of evaluation that should be included, which is node i's certainty towards its prediction. Imagine the following two cases: one is that node i observes that node j has been infected by proximity malware for 50 times out of the past 100 interactions, and another case is 1 observed infection in the only 2 interactions. The resulting v metric will be the same. However, the certainty, i.e. evidence sufficiency, in these two cases are quite different. Therefore, we present our definition of uncertainty u to reflect the difference in this dimension of evaluation:

$$u = \frac{4 \cdot \min\{\alpha, \beta\}}{(\alpha + \beta)^2}. \tag{3}$$

Here, Equation 3 is just one of the possible definitions for u. When α or β dominates, the u value will be low, which reflects the factor that the result of the next communication will be more predictable. When $(\alpha + \beta)$ is larger, the u value will also be low, which reflects the situation with more sufficient evidence. When there is no evidence and $\alpha = \beta = 1$, u should be 1, which represents the maximal uncertainty,

The vulnerability evaluation result (v, u) will guide the short-term coping components in making more comprehensive communication decisions. To communicate with nodes that are highly vulnerable, a node i may use more caution and sometimes directly reject to establish a connection to avoid the risk. The node also has strong incentive to enforce a strong security policy, since being highly vulnerable will make it unwelcome in future communication.

Community consensus component. The vulnerability evaluation based on each node's own information is too limited. With this component, nodes in the same community can draw consensus on vulnerability towards nodes in the community or nodes with direct contact to the community.

Using the vulnerability evaluation component, each node will form its own view (v, u) on its social neighbors' vulnerability. In order to expand the view, nodes need to exchange opinions and form consensus. However, a global consensus on nodes' vulnerability evaluations is both costly and unnecessary. Our consensus component is built on the community level, which allows nodes in the same community to share information. Since the community is constructed to reflect locality, nodes can easily verify each others' claims, and form views towards similar targets.

Each node will collect 2 hops of information before starting the calculation. More specifically, each node will evaluate all of its social neighbors' (v, u) and c value, and send the information to all of its neighbors in the same community. After the information has been exchanged and collected in the community, each node will follow the following two phases to calculate the group consensus.

In the first phase, each node i calculates a consensus vulnerability value towards a node j according to the opinions of all the nodes in the same community as i. Here, j could be a node in the same community as i, or j could be a node that is the social neighbor of some nodes from the same community as i. The vulnerability value of j is calculated as:

$$v_j^{(0)} = \frac{(1 - u_{kj})}{\sum_{k \in X, k \neq j}(1 - u_{kj})} \cdot v_{kj}, \tag{4}$$

where k represents the node in the same community X as node i. (v_{kj}, u_{kj}) represents node k's initial view towards node j. In this phase, the consensus vulnerability value $v_j^{(0)}$ towards a node j is the weighted sum of the direct views from all nodes in the same community as node i. The weight is decided by the node k's certainty towards the vulnerability evaluation. An opinion with larger certainty will gain more weight in the calculation.

After the first phase, we will have a vector $(\overrightarrow{v}^{(0)})^T = \{1, \ldots, v_i^{(0)}, \ldots, v_j^{(0)}\}$, which is the initial consensus view. However, this vector has not included the vulnerability propagation effect. The following are three network related properties that should be taken into consideration in the vulnerability evaluation.

Property 1: If a node is very vulnerable, its close neighbors also have a large chance to be infected.

Property 2: If a node has two neighbors and one is closer than the other, then it will be more dangerous if the closer one is more vulnerable.

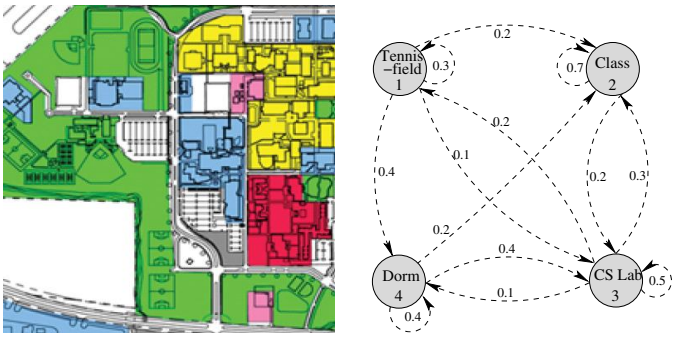
Property 3: A popular node tends to be more vulnerable than a lonely node, if their security policies are the same, and the lonely node's neighbors are the subset of the popular node.

To reflect the vulnerability propagation, we integrate closeness and consider the vulnerability propagation effect as follows:

$$v_j^{(1)} = \frac{1}{\sum_{l \in N(j)} c_{lj}} \cdot (v_j^{(0)} + \sum_{k \in N(j)} c_{kj} \cdot v_k^{(0)}), \tag{5}$$

Here, $N(j)$ represents j's social neighborhood including node j. $v_j^{(1)}$ not only includes the vulnerability from node j's own policy, which is reflected by $v_j^{(0)}$, but also considers the vulnerability propagated from j's neighbors in $N(j)$. For a node $k \in N(j)$, the vulnerability that k propagates to j is

This full text paper was peer reviewed at the direction of IEEE Communications Society subject matter experts for publication in the IEEE INFOCOM 2010 proceedings
This paper was presented as part of the main Technical Program at IEEE INFOCOM 2010.



(a) FAU map (b) Sample Markov chain

Fig. 5. The synthetic mobility traces are generated based on information collected from a university.

reflected by $c_{kj} \cdot v_k^{(0)}$. If k and j are close neighbors, this item will be larger. If k is more vulnerable, this item will also be larger. Therefore, this equation complies with our observation reflected in Properties $1 \sim 3$. The factor $\frac{1}{\sum_{l \in N(j)} c_{lj}}$ is used to normalize $v_j^{(1)}$. We can define $c_{jj} = 1$ and $c_{kj} = 0$ when $k \notin N(j)$. Then the coefficient for each item $\frac{c_{kj}}{\sum_{l \in N(j)} c_{lj}}$ reflects node k's influence in j's vulnerability. We can define C as the matrix $[\frac{c_{ij}}{\sum_{l \in N(i)} c_{li}}]$, and then Equation 5 can also be expressed in the matrix form:

$$\overrightarrow{v}^{(1)} = C \cdot \overrightarrow{v}^{(0)}, \tag{6}$$

After j's vulnerability has been updated, the propagation function should be updated as well. $v_j^{(2)} = \frac{1}{\sum_{l \in N(j)} c_{lj}} \cdot (v_j^{(1)} + \sum_{k \in N(j)} c_{kj} \cdot v_k^{(1)})$. We can continue in this manner, and calculate $\overrightarrow{v} = (C)^n \cdot \overrightarrow{v}^{(0)}$. If the community size is n, node i will get the aggregated views of all nodes in the same community as i, after n rounds of iterations.

However, if the community size n is large, the computational cost will be high. Fortunately, when the closeness matrix C is irreducible and periodic, which is generally true for the cases of the communities, the vulnerability vector $\overrightarrow{v}$ will converge to the same vector, i.e. the left principal eigenvector of C multiple with $\overrightarrow{v}^{(0)}$, for every node i in the same community. Therefore, each node should repeat $\overrightarrow{v}^{(r+1)} = C \cdot \overrightarrow{v}^{(r)}$, until $r = n$ or $||\overrightarrow{v}^{(r+1)} - \overrightarrow{v}^{(r)}|| < \epsilon$. The final vector $\overrightarrow{v}$ will be treated as the community consensus on the vulnerability of each node in the community's view.

With these vulnerability values, a node can make comprehensive decisions. The security policy of the reactions on these values should be tailored according to the needs of a specific application. We offer one example in the simulations, where the communication request will be rejected and the quarantine time will be doubled when a node's v value is larger than 0.7.

IV. Simulation

In our simulations, we compare the effectiveness of our scheme with a distributed local detection based coping scheme (Distributed for short), and a proximity signature dissemination based coping scheme [7] (Proximity for short). In the Distributed scheme, each node makes decisions based on its own information, and it will decline communications with a neighbor if that neighbor is detected as the infected node. In the Proximity scheme [7], when a node detects a malware, it will generate the signature of this malware and propagate it to all the other nodes that come into its proximity.

TABLE I
Characteristics of three mobility datasets

Dataset	*Haggle*	*Reality*	*Synthetic*
Device	iMotes	Phone	N/A
Network type	Bluetooth	Bluetooth	N/A
Duration (days)	3	246	10
Number of nodes	41	97	200
Number of contacts	22,459	54,667	Vary

A. Simulation setup

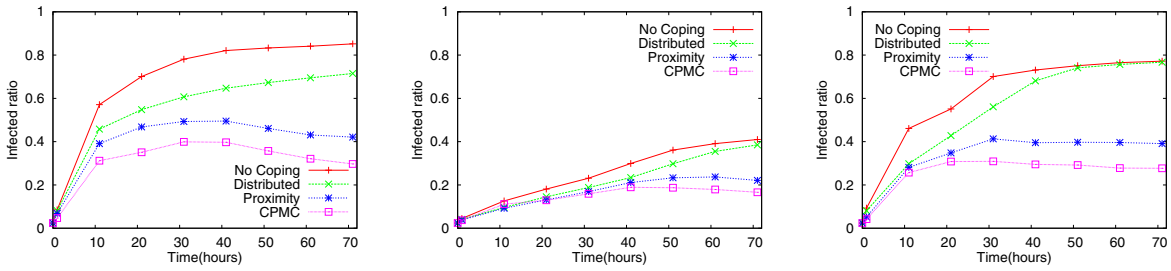
We primarily focus on two parameters: 1) *Malware infection Ratio:* for each specific malware, the infection ratio is reflected in the proportion of nodes infected by the malware; and 2) *Alert overhead:* the alert contains the signature, and the overhead is defined as the average number of alerts forwarded in the mobile network for each malware.

Mobility and contact traces. We ran trace-driven simulations with two different datasets: Haggle project [11] and MIT Reality Mining [12]. In both datasets, bluetooth contacts were logged and provided. Each contact recorded in the datasets includes the start time, end time, and IDs of the nodes in contact. For each round of simulation, a portion (default 30%) of the dataset was used as the contact history. The closeness associated with each neighboring relationship was constructed based on the contact history. The malware was then introduced and combined with the remaining portion to evaluate the effectiveness of the malware coping schemes.

We also adopted a community mobility model extended from [18] and generated synthetic traces based on information collected from the Florida Atlantic University (FAU). The collected information included a map of buildings as shown in Fig. 5(a), and the enrollment information of 250 students from four departments. The trace of a node, which represents a smartphone carried by a student, was generated according to a Markov chain, and an example chain is illustrated in Fig. 5(b). The states and probabilities were determined by the students' class schedules and enrollment information. If two nodes were in the same building at the same time, they had a probability (default 0.6) to setup a bluetooth connection. The contact length follows a power-law distribution.

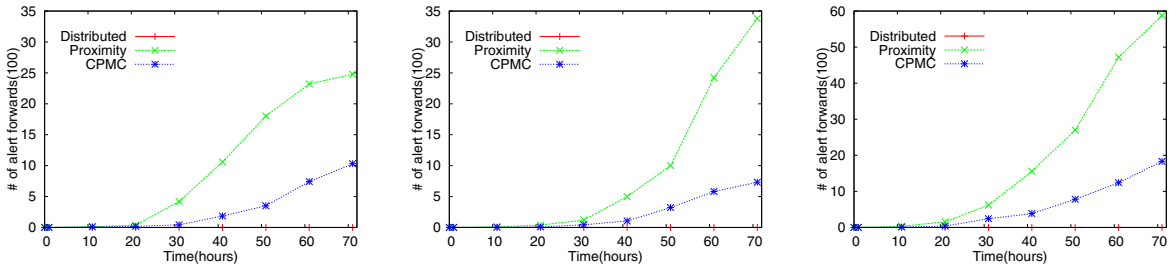
Malware propagation and nodes' policy. We simulated the malware similar to the Cabir or CommWarrior (bluetooth part) [4]. For all these bluetooth-oriented worms, the malware first scans the proximity of the infected node and tries to setup a connection and propagate whenever possible. However, based on the receiver's security policy and the malware's quality of concealment, the infection succeeds only with a

This full text paper was peer reviewed at the direction of IEEE Communications Society subject matter experts for publication in the IEEE INFOCOM 2010 proceedings
This paper was presented as part of the main Technical Program at IEEE INFOCOM 2010.



(a) Haggle (b) Reality (c) Synthetic

Fig. 6. Performance comparison on malware infection ratio.



(a) Haggle (b) Reality (c) Synthetic

Fig. 7. Performance comparison on total number of alert forwards.

certain probability. For each simulation, nodes were uniformly selected to be the origin of a proximity malware (seed), and the number of initial devices infected was one adjustable parameter (default: 1). The proximity malwares were generated by a Poisson process.

Each node was associated with an actual vulnerability factor that complies with the uniform distribution on interval (0, 1]. A node with a smaller vulnerability value indicates the tightened security policy. Each category of malware was also associated with a concealment factor that complies with the uniform distribution on interval (0, 1]. When an infected node meets a susceptible node, the susceptible node becomes infected with a probability equal to its vulnerability factor multiplied with the malware's concealment factor.

Our model is different from traditional epidemic SIR models [21]. First, SIR models are homogeneous, in the sense that an infected node is equally likely to infect any other susceptible nodes. We simulate nodes with different security policies as well as heterogeneous proximity malware in our model, which is the usual case in smartphone-based mobile networks. Second, our simulation is carried on top of realistic network topology abstracted from real and synthetic traces.

Each node knows only its own contact history before the community is formed. Each simulation was repeated 30 times with different random seeds for statistical confidence. The default threshold for forming communities is $T = 0.5$.

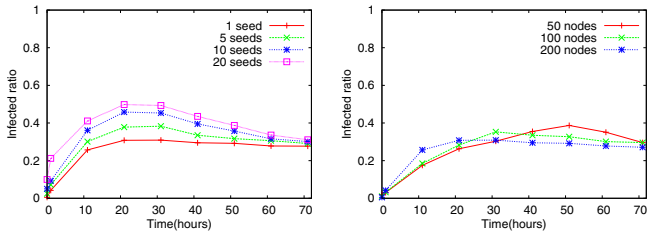
B. Simulation results

We first examine the effectiveness (i.e. malware infection ratio) and the cost (i.e. number of alert forwards) of the three schemes in three very different mobility scenarios in Figs. 6 and 7. The Reality dataset and Haggle dataset show two extreme cases in mobile networks. Haggle has extremely dense contact distribution, while Reality has extremely sparse contact distribution. We adjust key parameters of the mobile networks in the Synthetic dataset to investigate the coping scheme in more different scenarios.

The malware infection rates under different coping schemes are compared in Fig. 6 with accumulated simulation time. The results show that our CPMC scheme successfully controls the spreading of the proximity malware in the mobile network. The CPMC scheme also shows a steady improvement ($5\% \sim 10\%$) in terms of malware infection ratio over the proximity signature dissemination based coping scheme, since it combines the underlining community structure with a deliberated alert forwarding and quarantine plan. This improvement is significant, since the proximity signature dissemination scheme floods the signatures in the network in order to control the spreading of the proximity malware, which incurs huge alert forwarding costs as shown in Fig. 7.

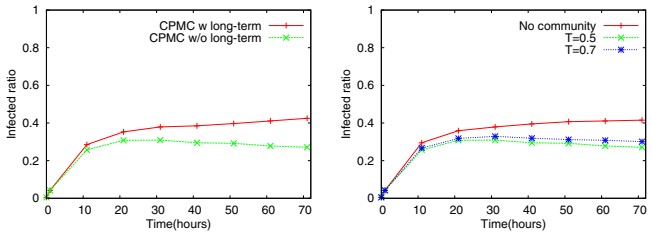
Although the Distributed scheme incurs no alert forwarding cost, as shown in Fig. 7, it also loses the opportunity to exchange the signatures in the mobile networks. That leads to its poor performance illustrated in Fig. 6. This intuitive method only deters the propagation of proximity malware. The proximity signature dissemination scheme is more effective, but the alert flooding will become a heavy burden. As shown in Fig. 7, it may produce $3 \sim 6$ times more alerts forwardings compared to the CPMC scheme. With the help of long-term components, the CPMC scheme can reduce the malware even more effectively than the flooding-based proximity signature dissemination scheme. According to Fig. 6, the differences

This full text paper was peer reviewed at the direction of IEEE Communications Society subject matter experts for publication in the IEEE INFOCOM 2010 proceedings
This paper was presented as part of the main Technical Program at IEEE INFOCOM 2010.



(a) Initial (b) Total

Fig. 8. Malware infection ratio with different a) number of initially infected nodes, b) total number of nodes in Synthetic trace.



(a) Long-term component (b) Community size

Fig. 9. Performance comparison a) with and without long-term component, b) with different community size in Synthetic trace.

are larger especially when the simulation time is between 30 to 72 hours. This is because many nodes have accumulated some evidence to accurately evaluate others' vulnerability. Combining the results in Fig. 7 with the steady improvement in Fig. 6, CPMC outperforms other existing proximity malware coping schemes in both effectiveness and cost efficiency.

To investigate the performances of the malware coping schemes in more cases, we vary two key factors in the synthetic trace: the number of initially infected nodes in Fig. 8(a), and the number of students (nodes) in Fig. 8(b). The differences in the number of students (nodes) create different densities of contact distribution.

Fig. 8(a) presents the difference of the malware infection ratio given a different initial number of seeds. In the original SIR model analysis, the number of initial seeds is considered as one key factor that may decide whether a malware will prevail in the network. However, according to Fig. 8(a), despite the difference in the peak of the malware infection ratio, our CPMC scheme always achieves a tendency to exterminate the malware given enough time. This further proves the effectiveness of the CPMC scheme.

Fig. 8(b) illustrates that the peak of the proximity malware infection ratio depends on the contact density distribution. In a sparser mobile network, such as the case of 50 nodes in Fig. 8(b), the long-term component needs more time to accumulate evidence and provides feedback to the short-term component. In a dense mobile network, the community will be larger and both the short-term and long-term components need less time to react to the proximity malware.

To illustrate the importance of integrating long-term components, we turn the long-term components on and off and compare the performance in Fig. 9(a). At the beginning, the two curves are very close to each other, which is because the long-term components have not accumulated enough evidences and the high uncertainty makes the vulnerability evaluation less instructive. But after 30 hours, the tendency of these two curves divides. Without the long-term component, since new proximity malware continue to occur in the network, the proximity malware persist in the network. The curve for the case with long-term components shows another tendency, since the vulnerability evaluation approaches the actual vulnerability of the nodes. High risk nodes will be treated with more caution in the communications. So, by combining the long-term and short-term components, the coping scheme can achieve a better control over the proximity malware.

Fig. 9(b) indicates the effectiveness of community-based consensus. The first curve shows the case in which each node conducts vulnerability evaluation without exchanging opinions with other nodes in the community. Compared to the second and third curves, the CPMC scheme without forming the community consensus is less effective, since the information that each node has is very limited. Although the difference in the malware infection ratio is around 5%, the difference is still significant if we consider the overall improvement of the long-term components in Fig. 9(a). $T = 0.5$ curve represents the case with a larger community but loosely intra-connected communities, while $T = 0.7$ curve represents the case with a smaller community with stronger intra-connection. In our simulation, the $T = 0.5$ case appears to be better. However, we infer that the optimal community size for the vulnerability evaluation depends on the distribution of contacts in the smartphone-based mobile networks.

Simulation results confirm that, compared with existing proximity malware coping schemes, CPMC has a better performance in controlling the malware infection ratio and a low cost. The results under the different settings of the real and synthetic traces prove the superiority of CPMC, which deliberately utilizes the community structure, a controllable granularity of security, in the mobile networks. It also reduces the alert forwarding overhead.

V. Related Work

Proximity malware. With the inherent convenience and increasing computation and communication power, smartphones and other mobile devices have become natural focus of future network applications. At the same time, the smartphone-based mobile networks will also become the focus of malware. A number of studies have demonstrated the severe threat of proximity malware propagation through Bluetooth. Su *et al.* gather Bluetooth scanner traces and use simulation to demonstrate that one effective way for malware to propagate is via Bluetooth [22]. Yan *et al.* develop a detailed Bluetooth worm model [23]. Bose and Shin show that a worm that uses both SMS/MMS and Bluetooth can propagate faster than by messaging alone [5].

Packet forwarding in mobile networks. In mobile networks, one cost-efficient way to route packets is via short-range communication capabilities of intermittently connected smartphones [13], [14]. The delegation forwarding [19], [24] is one of these routing schemes. In [19], each message copy maintains a forwarding threshold which is initialized as the quality of its source node towards a certain destination, and then updated with new delegate's quality. Forwarding to intermediate nodes with worse quality will be avoided. Our signature propagation component extends this idea on the community level and uses a set of closeness thresholds to decide on delegates.

While early work in mobile networks used a variety of simplistic random *i.i.d.* models, such as random waypoint, recent findings [18] show that these models may not be realistic. Moreover, many recent studies [16], [17], [18] based on real mobile traces reveal that the nodes' mobility processes certain social network properties.

Trust evaluation schemes. We develop our vulnerability evaluation component based on the observation that trust evaluations can be the bridge to link the past experiences with future predictions. Various frameworks [25], [26] have been designed to model trust networks. We can divide them into three main categories. The trust management systems in the first category have a central authority, usually called the trusted third party. In the second category, one global trust value is drawn and published for each node, based on other nodes' opinions of it. EigenTrust [27] is one mechanism in this category. The third category includes the trust management systems that allow each node to have its own view of other nodes [28], [29]. Our vulnerability evaluation is conducted on the community level, which draws trust evaluation results from a new granularity.

VI. Conclusion

In this paper, we propose CPMC, an efficient proximity malware coping scheme based on the social relationships and community structure of the smartphone-based mobile networks. We first propose a short-term community-based delegation forwarding scheme to quickly propagate each signature into all communities while avoiding unnecessary redundancy. A community quarantine method is presented to enhance the difference in signature propagation and malware propagation. We also design a long-term vulnerability evaluation scheme, including community consensus formation, to help users make comprehensive communication decisions. Extensive results of simulations based on real and synthetic traces are provided, which further illustrate the efficiency of the proposed scheme. In the future, we plan to investigate the detection node deployment problem in proximity malware protection by utilizing the community structure.

Acknowledgments

This work is supported in part by NSF grants CNS 0422762, CNS 0626240, CCF 0830289, and CNS 0948184. Emails: fengli@iupui.edu, yyang4@fau.edu, and jiewu@temple.edu.

References

[1] H. Debar, E. Filiol, and G. Jacob. Behavioral detection of malware: from a survey towards an established taxonomy. *Journal in Computer Virology*, pages 251–266, 2008.
[2] G. Lawton. On the trail of the conficker worm. *IEEE Computer*, 42(6):19–22, 2009.
[3] M. May, V. Lenders, G. Karlsson, and C. Wacha. Wireless opportunistic podcasting: implementation and design tradeoffs. In *Proc. of ACM CHANTS*, 2007.
[4] Mobile security threats. *http://www.f-secure.com*, 2009.
[5] A. Bose and K. Shin. On mobile viruses exploiting messaging and bluetooth services. In *Proc. of the ICST Securecomm*, 2006.
[6] Z. Zhu, G. Cao, S. Zhu, S. Ranjany, and A. Nucciy. A social network based patching scheme for worm containment in cellular networks. In *Proc. of the IEEE INFOCOM*, 2009.
[7] G. Zyba, G. Voelker, M. Liljenstam, A. Mehes, and P. Johansson. Defending mobile phones from proximity malware. In *Proc. of the IEEE INFOCOM*, 2009.
[8] P. Hui, J. Crowcroft, and E. Yoneki. Bubble rap: Social-based forwarding in delay tolerant networks. In *Proc. of ACM MobiHoc*, 2008.
[9] F. Li and J. Wu. LocalCom: A community-based epidemic forwarding scheme in disruption-tolerant networks. In *Proc. of IEEE SECON*, 2009.
[10] F. Li and J. Wu. MOPS: Providing content-based service in disruption-tolerant networks. In *Proc. of IEEE ICDCS*, 2009.
[11] J. Scott, R. Gass, J. Crowcroft, P. Hui, C. Diot, and A. Chaintreau. CRAWDAD data set cambridge/haggle (v. 2006-09-15). http://crawdad.cs.dartmouth.edu/cambridge/haggle, September 2006.
[12] N. Eagle and A. Pentland. CRAWDAD data set MIT/reality (v. 2005-07-01). http://crawdad.cs.dartmouth.edu/mit/reality, July 2005.
[13] A. Vahdat and D. Becker. Epidemic routing for partially connected ad hoc networks. In *Technical Report CS-200006, Duke University*, 2000.
[14] J. Burgess, B. Gallagher, D. Jensen, and B. Levine. Maxprop: Routing for vehicle-based disruption-tolerant networks. In *Proc. of IEEE INFOCOM*, 2006.
[15] U. Luxburg. A tutorial on spectral clustering. *Statistics and Computing*, 17(4):395–416, 2007.
[16] E. Daly and M. Haahr. Social network analysis for routing in disconnected delay-tolerant MANETs. In *Proc. of ACM MobiHoc*, 2007.
[17] N. Djukic, M. Piorkowski, and M. Grossglauser. Island hopping: Efficient mobility-assisted forwarding in partitioned networks. In *Proc. of IEEE SECON*, 2006.
[18] W. Hsu, T. Spyropoulos, K. Psounis, and A. Helmy. Modeling time-variant user mobility in wireless mobile networks. In *Proc. of IEEE INFOCOM*, 2007.
[19] V. Erramilli, M. Crovella, A. Chaintreau, and C. Diot. Delegation forwarding. In *Proc. of ACM MobiHoc*, 2008.
[20] C. Zou, W. Gong, and D. Towsley. Worm propagation modeling and analysis under dynamic quarantine defense. In *Proc. of the ACM WORM*, 2003.
[21] D. Chakrabarti, J. Leskovec, C. Faloutsos, S. Madden, C. Guestrin, and M. Faloutsos. Information survival threshold in sensor and p2p networks. In *Proc. of the IEEE INFOCOM*, 2007.
[22] J. Su, K. Chan, A. Miklas, K. Po, A. Akhavan, S. Saroiu, E. de Lara, and A. Goel. A preliminary investigation of worm infections in a bluetooth environment. In *Proc. of the ACM WORM*, 2006.
[23] G. Yan, H. Flores, L. Cuellar, N. Hengartner, S. Eidenbenz, and V. Vu. Bluetooth worm propagation: mobility pattern matters! In *Proc. of the ACM Symposium on Information, Computer and Communications Security (ASIACCS)*, 2007.
[24] X. Chen, J. Shen, T. Groves, and J. Wu. Probability delegation forwarding in delay tolerant networks. In *Proc. of IEEE ICCCN*, 2009.
[25] A. Josang, R. Ismail, and C. Boyd. A survey of trust and reputation systems for online service provision. *Decision Support Systems*, 43(2):618–644, 2007.
[26] A. Srinivasan, J. Teitelbaum, and J. Wu. Drbts: Distributed reputation-based beacon trust system. In *Proc. of IEEE DASC*, 2006.
[27] S. Kamvar, M. Schlosser, and H. Garcia-Molina. The eigentrust algorithm for reputation management in P2P networks. In *Proc. of International World Wide Web Conference(WWW)*, 2003.
[28] S. Buchegger and J. Boudec. Performance analysis of the confidant protocol. In *Proc. of ACM MobiHoc*, pages 226–236, 2002.
[29] F. Li and J. Wu. Mobility reduces uncertainty in MANETs. In *Proc. of IEEE INFOCOM*, 2007.