The Microcosmic Model of Worm Propagation

YINI WANG*, SHENG WEN, SILVIO CESARE, WANLEI ZHOU AND YANG XIANG

School of Information Technology, Deakin University, Melbourne, Australia
*Corresponding author: yiniwang@deakin.edu.au

Each year, large amounts of money and labor are spent on patching the vulnerabilities in operating systems and various popular software to prevent exploitation by worms. Modeling the propagation process can help us to devise effective strategies against those worms’ spreading. This paper presents a microcosmic analysis of worm propagation procedures. Our proposed model is different from traditional methods and examines deep inside the propagation procedure among nodes in the network by concentrating on the propagation probability and time delay described by a complex matrix. Moreover, since the analysis gives a microcosmic insight into a worm’s propagation, the proposed model can avoid errors that are usually concealed in the traditional macroscopic analytical models. The objectives of this paper are to address three practical aspects of preventing worm propagation: (i) where do we patch? (ii) how many nodes do we need to patch? (iii) when do we patch? We implement a series of experiments to evaluate the effects of each major component in our microcosmic model. Based on the results drawn from the experiments, for high-risk vulnerabilities, it is critical that networks reduce the number of vulnerable nodes to below 80%. We believe our microcosmic model can benefit the security industry by allowing them to save significant money in the deployment of their security patching schemes.

Keywords: network security; worm propagation; patch strategy

Received 12 January 2011; revised 26 July 2011
Handling editor: Ing-Ray Chen

1. INTRODUCTION

Worms and their variants are widely believed to be one of the most serious challenges to address in network security research. According to the Symantec Global Internet Security Threat Report [1], the second highest percentage of the top 50 potential malicious code infections for 2009 belonged to worms, which increased from 29% in 2008 to 43% in 2009. Six of the top 10 threats in 2009 had worm components, compared with only four in 2008.

In order to prevent worms from spreading effectively, large amounts of money and labor are spent by the industry on patching vulnerabilities in operating systems and popular software. Wipro Technologies stated in their 2004 patch management costs report [2], ‘Annual per-system patching costs on windows: $297.1 (clients), $416.2 (Non-Database Servers), $682.1 (Database Servers) and on open source software systems: $343.7 (clients), $479.3 (Non-Database Servers), $1020.4 (Database Servers)’. We expect the cost would be greater in 2010 because the sophistication and damage potential of worms have increased enormously. Consequently, it is important to provide a set of optimized and economic patch strategies to deal with the problems of where and how many nodes do we need to patch.

Security experts routinely uncover software vulnerabilities and then issue software patches and upgrades. Sometimes, however, it may cause inadvertent and possibly detrimental effects. Security researcher Dan Kaminsky uncovered a flaw in the Domain Name System (DNS) and published a series of patches before publicly disclosing the specifics of the vulnerability [3]. By looking at the patch, others were able to reverse-engineer the patch, and shortly afterwards code to exploit the newfound weakness had been posted to a website. Some network administrators may initially have been reluctant to patch their systems, fearing that the upgrade itself might cause problems. However, the result is the potential break out of worms before a sufficient number of nodes are patched. Therefore, we need to quantify an appropriate time for patching vulnerabilities.
In order to understand and possibly address: (1) Where to patch; (2) How many nodes do we need to patch; (3) When to patch, it is very important to characterize the worm propagation. Previous work has adopted the classical simple epidemic model [4–7], which simulates two states for all hosts: susceptible and infectious. This is known as the SI model. However, this approach is not suitable for the cases where the infected and infectious nodes are patched or removed. Consequently, the classical general epidemic model [8, 9], also called susceptible-infected-recovered (SIR) models, have been proposed to extend simple epidemic models by introducing a removal process of infectious peers. Continued improvements [10–12] on modeling worm propagation have considered the immunization defense. Zou et al. [10] proposed a two-factor worm model, which developed the general epidemic model by taking into account both the effect of human countermeasures and decreases in the infection rate. They also studied the propagation model for internet email worms [11] by comparing three different types of topology and summarized the immunization strategies. Although these propagation models perform well in predicting the tendency of worms spreading in the network, macroscopic models identify very little information within the propagation procedure. This leads to difficulties to deal with the proposed three problems. In fact, there are five parameters involved in modeling worm propagation: (1) propagation probability; (2) infectious nodes’ distribution; (3) vulnerable nodes’ distribution; (4) patch strategy; (5) time delay. Previous models of worm propagation have failed to address the following issues:

(i) Propagation probability between each pair of nodes so that they cannot locate which set of nodes are easier to be infected (Sections 3.1 and 3.2.).
(ii) Propagation time delay between each pair of nodes so that they cannot estimate the time for each node to be infected from the propagation source (Sections 3.1 and 3.2).
(iii) Worms’ propagation procedure from node to node so that they have weak information to decide an appropriate position and time for patching on each node (Section 3.3).
(iv) Errors caused by reinfection in traditional models so that they cannot avoid overestimation of patching budget (Section 3.4).
(v) The mutual impact between propagation probability and time delay (Section 3.1).

A recent improvement was proposed in [13], which used a logic matrix approach to model the spreading of peer-to-peer worms between each pair of all peers. It adopted two constants of logic type (True or 1, False or 0) as the value of matrix variables. This 0–1 matrix stands for the propagation ability (PA) of nodes, that is whether they can allow the worms to spread or not. Nevertheless, a significant limitation of this model is that it cannot describe the propagation process of some worms, such as local preference worms, since these worms have different spreading probabilities for specific IP address spaces. More importantly, the model does not include temporal factors so that it cannot model dynamic worm propagation procedures.

1.1. Motivation

Compared with a macrocosmic propagation model, a microcosmic model can accurately reflect the distribution of nodes in the network, which is beneficial for describing the propagation procedure. We can examine deep inside the spreading procedure and are able to understand how the current infected states impact on the worm’s propagation in the next step. Modeling a microcosmic propagation procedure can provide defenders useful information to answer the questions of where do we patch, how many nodes do we patch and when do we patch. Moreover, there is little research in microcosmic propagation models from the view of probability. Therefore, we are motivated to present a microcosmic propagation model for simulating worms spreading. The objective of our research is to generate a set of optimized patch strategies to minimize the number of infected peers and provide economic benefits to the industry by selectively deploying security patches.

1.2. Contribution

In this paper, we mainly focus on scanning worms, which scan the entire network and infect targets without regard to topological constraints. It is closely related to the logical features of the network rather than the physical structure. Therefore, our proposed approach is suitable for modeling networks that are susceptible to scanning worms. Our model has several important components:

(i) propagation matrix (PM) are proposed to construct propagation models for worms;
(ii) a propagation source vector (S) is introduced for describing the distribution of initial infectious nodes;
(iii) a vulnerable distribution vector (V);
(iv) a patching strategy vector (Q) accounts for a special deployment of patching nodes;
(v) propagation abilities (PA).

To the best of our knowledge, there are few papers that refer to the microcosmic procedure of worm propagation between nodes in a network. Although papers such as [13] analyzed worm propagation from the view of the microcosm, it adopted a simplified logic matrix to indicate the infected states of the network. This simple logic does not effectively describe the propagation procedure between each pair of nodes, nor does it reflect the spreading effect in each step of a worm’s propagation. The major contributions of this paper are as follows:

(i) We create a microcosmic landscape on worm propagation and successfully provide useful information for the
proposed problems of where, when and how many nodes do we need to patch (Section 4.4).

(ii) Associated with $S$, $V$, $Q$, our model can also help us to evaluate:

(a) The mutual effect of initial infectious states and patch strategies (Section 4.2).
(b) The impact of different distributions of vulnerable hosts has on worm propagation (Section 4.3).

(iii) We introduce a complex matrix to represent the probabilities and time delay between each pair of nodes. These factors lead to the accurate exploration of the propagation procedure and estimation of both infection scale and the effectiveness of defense. The extension from the real field to the complex field reflects the mutual effect between these two factors, which matches the real case well (Section 4.5).

(iv) We carry out extensive simulation studies of worm propagation. Through matrix iteration, we derive a better understanding of dynamic infection procedures in each step. These procedures include:

(a) What is the propagation probability and time delay between each pair of nodes.
(b) How does one node infect another node directly.
(c) How does one node infect another node through a group of intermediate nodes.

The rest of the paper is organized as follows. In Section 2, we review related work. In Section 3, we present the proposed microcosmic propagation model. Next, we conduct an analysis and deduce the result for obtaining an optimized patch strategy in Section 4. Section 5 performs a discussion and open issues. Finally, we conclude the paper in Section 6.

## 2. RELATED WORK

In the area of network security, both macroscopic [7, 10, 11, 14–24] and microcosmic [13] models exist for simulating different worm propagation. Most worm propagation models are based on macroscopic view, such as homogenous worms' model, local preference worms' model and topological worms' model, which mainly describe the overall worms spreading tendency. In contrast, microcosmic models prefer to study the propagation procedure between nodes according to different scenarios of infectious states, vulnerable states and quarantine states.

### 2.1 Macroscopic worm propagation models

#### 2.1.1 Homogenous worms' model

The homogenous worm propagation model is a simple epidemic model that is used in much research [7, 10, 17–20] to model worm propagation for random scanning worms (Code Red [10], Slammer [25]). Variants of random scanning worms (hit-list worms [7], routable worms [26]) are modeled using extensions of this simple epidemic model. The homogenous model is based on the concept of a fully connected graph and is an unstructured worm model that ignores the network topology. The model assumes that each infectious host has an equal probability in spreading the worm to any vulnerable peer in a network. Staniford et al. [7] presented an RCS (Random Constant Spread) model to simulate the propagation of the Code Red II v2 worm, which is almost identical to the homogenous model. Zou et al. [10] introduced a two-factor model, which extended the homogenous model by considering human countermeasures and network congestion. These models focus on analyzing the trends of worm propagation. However, they do not describe the worm propagation from node to node or the infection process when disrupted by patching or immunizing nodes. Thus, they are not suitable for modeling the dynamic process of infection and patching between each pair of nodes. Furthermore, the models are significantly limited for modeling worms that scan IP addresses with differing probabilities and are unable to simulate topology-based worm propagation. Additionally, they did not discuss the different impacts of reinfection and non-reinfection on worm propagation. Rohloff and Basar [18] presented a stochastic density-dependent Markov jump process propagation model. Sellke et al. [19] provided a stochastic Galton–Watson Markov branching process model. These two models are also limited to simulate the propagation tendency, which cannot describe the spreading procedure.

#### 2.1.2 Local preference worms' model

Since vulnerable nodes are not uniformly distributed, some localized scanning worms (Code Red II [14–16], Blaster worm [21]) propagate the virus with a high probability in certain IP addresses for the purpose of increasing their spreading speed. Taking Code Red II as an example, the probability of the virus propagating to the same class A IP address is 3/8; to the same class A and B IP address it is 1/2; and to the random IP address it is 1/8. Thus, the local preference model employs a non-homogenous pattern to simulate worm propagation. Chen et al. [20] presented a LAAWP (Local Analytical Active Worm Propagation) model to take into account the characteristics of local subnet scanning worms' spreading. However, this model assumes that the distribution of vulnerable hosts is uniform in every subnet. They did not consider the impact of vulnerable distribution on worm propagation, which is one of the important parameters on modeling worms spreading. Zou et al. [17] considered the distribution of vulnerable hosts in the IPv4 address space and provided a more accurate method to model local preference scanning worms' propagation. In this model, they suppose only the first $m$ networks have vulnerable hosts. However, they still assume vulnerability distribution is uniform in each subnet. Moreover, although their model introduced the pair-wise rates of infection in local scanning and remote scanning, it is still derived from the homogenous model. Therefore, these models cannot reflect non-uniform
vulnerability distribution on worm propagation and the dynamic process of infection and immunization between each pair of nodes.

2.1.3. Topological worms’ model
Both homogenous model and local preference model reflect unstructured worms’ propagation without regard to topological constraints. However, a topological model describes a structure-dependent propagation of worms, which relies on the topology for the spreading of viruses. Zou et al. [11] considered these two probabilities and compared internet email worm propagation on power law topologies, small world topologies and random graph topologies. In the proposed model, the probability of each user opening a worm attachment can be treated as an infected probability and the distribution of email checking times can represent the propagation probability. However, this model still describes the email worm propagation tendency instead of modeling the dynamic spreading procedure between each pair of nodes. In addition, they discussed the lower bound for non-reinfection case, but their model is not capable of accurately eliminating the errors caused by reinfection.

2.2. Microcosmic worm propagation models
Microcosmic worm propagation models focus on the infection procedure between each pair of nodes. Fan and Xiang [13] employed a logic matrix approach to model the spreading of peer-to-peer worms between each pair of all peers. They discovered the relation between out-degree, vulnerability and coverage rate in power law topologies and simple random graph topologies, respectively. However, they did not consider the propagation probability and infected probability of each node, which has significant impacts on the infection procedure. Additionally, although they do not allow peers’ outbound links to themselves, they cannot avoid propagation loops formed among intermediate nodes.

We propose a novel complex matrix that models worm propagation, and simulates the microcosmic spreading procedure of worms. Using this complex matrix in worms’ propagation simulation forms a major difference between this work and existing work. In our model, we focus on investigating the procedure of worms’ spreading and providing effective patching strategies, which will benefit IT industries and security best practice.

3. PROPAGATION MODEL
In this section, we present the propagation model that is used to simulate the propagation process of worms and to estimate an optimized patch strategy.

3.1. Propagation matrix (PM)
We propose employing an $n$ by $n$ square complex matrix $PM$ with elements $c_{xy}$ to describe a network consisting of $n$ peers.

$$ PM = \begin{bmatrix} c_{11} & \cdots & \cdots & \cdots \\ \cdots & c_{xy} & \cdots & \cdots \\ \cdots & \cdots & \cdots & \cdots \end{bmatrix}_{n \times n}, $$

$$ c_{xy} = p_{xy} + d_{xy}i, \quad c_{xy} = 0 (x = y), $$

$$ \text{Re}(c_{xy}) = p_{xy} = p(N_y|N_x) \quad p_{xy} \in [0, 1], $$

$$ \text{Im}(c_{xy}) = d_{xy} = t(N_x, N_y) \quad d_{xy} \in (0, 1]. $$

Each row of the PM represents the propagation probability ($p_{xy}$) and propagation delay ($d_{xy}$) from one infectious peer to all other peers. Each column represents $p_{xy}$ and $d_{xy}$ from infectious peers to a target peer. We assume that a peer cannot propagate the worm to itself; so the self-propagation $p_{xx}$ and $d_{xx}$ are zero.

Generally, worms scan an IP address space or a hit-list for scanning the entire IP address space or the hit-list.
3.2. Propagation Function (γ)

In real-world conditions, worms could be spread between peers from node \( x \) to node \( y \) via one or more intermediate nodes, as shown in Fig. 1a. In existing worms, it is observed that an infectious peer can propagate worms and a vulnerable peer can also be infected to become a new infectious node for future propagation. In this scenario, we assume that initially every peer in the PM is vulnerable to the worm.

We assume that worm propagation from node \( x \) \( (N_x) \) to node \( y \) \( (N_y) \) is via and only via \( k \) intermediate nodes in a network consisting of \( n \) peers. According to the rule of complex multiplication, as shown in Fig. 1b, the first component \((ac - \beta bd)\) of the result indicates propagation probability from \( N_x \) to \( N_y \). Here we manually insert an impact factor \((\beta)\) to describe the decrease in the propagation probability caused by time delay. It combines the characteristic of the worm itself and the network it operates on. The second component \((\gamma k)\) of the result indicates possible time delay for worm propagation from \( N_x \) to \( N_y \). It is denoted by \( c_{xy}^{(k)} \) and defined in (2):

\[
c_{xy}^{(k)} = \sum_{m=1}^{\infty} \sum_{m \neq x} (1 - \gamma k) c_{my}
\]

\[
= \sum_{m=1}^{\infty} \left( (p_{xy}^{(k)} - \beta t_{xy}^{(k)}) m_{xy} + (p_{xy}^{(k)} + \gamma t_{xy}^{(k)}) m_{xy} \right), \quad k \in [1,n-2], x = 1, \ldots, n, \quad y = 1, \ldots, n.
\]

Since \( N_x \) self-propagation via \( k \) nodes is meaningless in the real world, we define the value of this propagation probability as zero; namely \( c_{xy}^{(k)} = 0 \) when \( x = y \). We introduce a function \( \gamma \) to conduct the iterated procedure as in (3):

\[
\gamma^0(\text{PM}) = \text{PM},
\]

\[
\gamma^k(\text{PM}) = \text{PM} \times \text{PM} \times \cdots \times \text{PM}, \quad k \geq 1.
\]

Operation \( \times \) is the traditional matrix multiplication. Subsequently, the PM can be represented by the following equation when the worm propagation is via and only via \( k \) intermediate nodes, as shown in (4).

\[
\text{PM}^{(k)} = \begin{bmatrix}
c_{11}^{(k)} & \cdots & \cdots \\
\cdots & c_{xy}^{(k)} & \cdots \\
\cdots & \cdots & c_{nn}^{(k)}
\end{bmatrix}_{n \times n} = \gamma^k(\text{PM}).
\]

3.3. Three key factors

In a network, there are three significant factors for worm propagation: infectious state, vulnerability distribution and patch strategy. The infected state represents the state of the peer has being infected or not. Vulnerability distribution identifies vulnerable peers in the network. Patch strategy provides an approach to cure the infected peers. Infected peers cannot be infected after being patched.

3.3.1. Propagation source vector (S)

An initial propagation source vector \((S)\) is defined as shown in (5). An infectious peer that can propagate worms is represented with a probability of one. The probability of zero means that a peer is healthy and does not have the ability to propagate the worm.

\[
S = [s_1, s_2, \ldots, s_x, \ldots, s_n]^T, \quad s_x = 0 \text{ or } 1, \quad x = 1, \ldots, n. \quad (5)
\]

The iterated procedure can be represented as function \( \gamma_s \) in (6):

\[
\gamma^0_s(\text{PM}) = S \& L \text{ PM},
\]

\[
\gamma^k_s(\text{PM}) = \gamma^{k-1}_{s_0}(\text{PM}) \times \text{PM} = (S \& L \text{ PM}) \times \text{PM} \times \cdots \times \text{PM} \quad (k \geq 1).
\]

We define \& \( L \) to indicate a new logic AND operation of a column vector \( A \) and a matrix \( B \), called \( \text{Left Logic AND} \). The result of \( A \& L \text{B} \) is a new logic matrix of the same dimension as \( B \). This operation is used to eliminate non-infectious nodes. Each element in the new matrix is the result of the product of the corresponding elements \( a_{xy} \) and \( b_{xy} \) from each column of matrix \( B \). It is defined in (7):

\[
A \& \text{L} B = \begin{bmatrix}
a_1 \\
\cdots \\
a_n
\end{bmatrix} \& \text{L} \begin{bmatrix}
b_{11} & \cdots & \cdots \\
\cdots & b_{xy} & \cdots \\
\cdots & \cdots & b_{nn}
\end{bmatrix} = \begin{bmatrix}
a_1 \times b_{11} & \cdots & \cdots & a_1 \times b_{1n} \\
\cdots & \cdots & \cdots & \cdots \\
a_n \times b_{n1} & \cdots & \cdots & a_n \times b_{nn}
\end{bmatrix}.
\]

The PM can be represented by the following equation when worm’s propagation is via and only via \( k \) intermediate nodes, as in (8).

\[
\text{PM}^{(0)}_x = \gamma^0_s(\text{PM}),
\]

\[
\text{PM}^{(k)}_x = \gamma^k_s(\text{PM}) = \text{PM}^{(k-1)}_x \times \text{PM} \quad (k \geq 1).
\]

During the propagation process, each intermediate node can be infected and become infectious. We introduce an infected state vector \( I \), as shown in (9):

\[
I = [i_{e_1}, i_{e_2}, \ldots, i_{e_x}, \ldots, i_{e_n}]^T,
\]

\[
i_{e_x} = 0 \text{ or } 1, \quad x = 1, \ldots, n, \quad (9)
\]

\[
i^{(k)}_s = \Gamma(S^T, \text{PM}_x^{(k)}) \quad (k \geq 0),
\]

where \( \Gamma \) function computes each item in infected state vector \( I \) using the formula as shown in (10).

\[
i^{(k)}_s = \sum_y S_y p^{(k)}_{xy} + \sum_y S_y p^{(k)}_{xy} I^{(k)}_y, \quad (10)
\]

\( I^{(k)}_s \) reflects the infected possibility and time delay of each node after worm propagation via \( k \) intermediate nodes under a certain deployment of \( S \).
3.3.2. Vulnerable distribution vector (V)

Under real-world conditions, the vulnerability of a peer is an objective fact. Therefore, a healthy peer without any vulnerability cannot become infectious in the worm’s propagation process. On the basis of this fact, we need to consider the vulnerability distribution in the PM. The vulnerable distribution vector (V) is defined in (11). For an element in V, the value of one represents that a peer is vulnerable. Zero means that the peer is healthy and is not vulnerable.

\[ V = [v_1, v_2, \ldots, v_x, \ldots, v_n]^T, \quad v_x = 0 \text{ or } 1, \quad x = 1, \ldots, n. \]  
(11)

Once nodes are vulnerable, they can become infected and have the ability to infect others. Therefore, the iterated procedure can be represented as function \( \gamma_{sv} \) in (12):

\[ \gamma_{sv}^0(PM) = S \& L \times PM \& R \times V^T, \]
\[ \gamma_{sv}^k(PM) = \gamma_{sv}^{k-1}(PM) \times (V \& L \times PM \& R \times V^T) \quad (k \geq 1). \]  
(12)

We define \& R to indicate a new logic AND operation of a column vector A and a matrix B, called Right Logic AND, which is different from Left Logic AND. The result of \& R is a new logic matrix of the same dimension as B. Each element in the new matrix is the result of the product of the corresponding elements \( a_y \) and \( b_{xy} \) from each row of matrix B. It is defined in (13):

\[
B \& R A = \begin{bmatrix} b_{11} & \cdots & \cdots \\
 \cdots & b_{xy} & \cdots \\
 \cdots & \cdots & b_{nn} \\
\end{bmatrix} \& R \begin{bmatrix} a_1 & \cdots & a_n \\
\end{bmatrix}
= \begin{bmatrix} b_{11} \times a_1 & \cdots & b_{1n} \times a_n \\
\cdots & b_{xy} \times a_y & \cdots \\
\cdots & \cdots & b_{nn} \times a_n \\
\end{bmatrix}.
\]  
(13)

Considering the vulnerability distribution vector, the PM and infected probability vector I can be represented by the following equations, respectively, when the worm propagates via and only via \( k \) intermediate nodes, as in (14).

\[
PM_{xy}^{(k)} = \gamma_{sv}^k(PM) \quad (k \geq 0), \\
I_{xy}^{(k)} = \Gamma(S^T, PM_{xy}^{(k)}) \quad (k \geq 0).
\]  
(14)

3.3.3. Patch strategy vector (Q)

An infected peer can be cured to become a healthy node, becoming unable to spread worms to other peers. Therefore, we need to remove these nodes from the propagation process at that time. We define a patch vector Q in (3). For each element in Q, the value of one represents that a peer has been patched and becomes a healthy node. A value of zero indicates that a peer is still vulnerable.

\[ Q = [q_1, q_2, \ldots, q_x, \ldots, q_n]^T, \quad q_x = 0 \text{ or } 1, \quad x = 1, \ldots, n. \]  
(15)

Once the nodes have been patched, they will become immune to the worms and lose their infectious ability. Thus we should exclude these patched nodes in the matrix for the successive iteration. The iterated procedure can be represented as function \( \gamma_{svq} \) shown in (16):

\[
Q' = V \& Q, \\
\gamma_{svq}^0(PM) = S \& L \times PM \& R \times Q'^T, \\
\gamma_{svq}^k(PM) = \gamma_{svq}^{k-1}(PM) \times (Q' \& L \times PM \& R \times Q'^T) \quad (k \geq 1).
\]  
(16)

We define \& to indicate a new logic AND operation between two elements. The definition for \& operation is shown in Table 1.

After considering the patch strategy vector, the PM and infected probability vector I can be represented by the following equations, respectively, when the worm propagates via and only via \( k \) intermediate nodes, as shown in (17).

\[
PM_{xy}^{(k)} = \gamma_{svq}^k(PM) \quad (k \geq 0), \\
I_{xy}^{(k)} = \Gamma(S^T, PM_{xy}^{(k)}) \quad (k \geq 0).
\]  
(17)

3.4. Error Calibration Vector (E)

We consider two scenarios of infection: reinfection and non-reinfection. Generally, reinfection means a node can be infected repeatedly and non-reinfection indicates a node can be infected only once [14].

In our model, each node can be infected by \( N(N \geq 1) \) nodes and we assume no peer has an outbound link to itself. If a worm belongs to the reinfection type, the earlier-mentioned propagation mechanism is reasonable. However, if a worm belongs to the non-reinfection type, propagation loops will be formed during the spreading procedure, which results in errors in the infected probabilities, as shown in Fig. 2. \( N_1 \) can be infected by infectious nodes via two intermediate nodes: \( S – N_1 – N_2 – N_3 – N_4 – N_5 – N_6 \). Since two loops (3 and 4) begin from the intermediate nodes in the propagation procedure, we cannot eliminate the infinite probability loops by setting the diagonal items in PM to zero. The macroscopic propagation models cannot exclude the errors caused by propagation loops among the intermediate nodes.

In order to avoid the errors in non-reinfection worms, we introduce an error calibration vector E, as shown in (18).

\[
E^{(k)} = \begin{bmatrix} e_1^{(k)} \\
e_2^{(k)} \\
\vdots \\
e_{k-1}^{(k)} \\
e_n^{(k)} \\
\end{bmatrix}, \\
e_i^{(k)} = \sum_{x=1}^{k-1} P_{xi}^{(k-x)} P_{xi}^{(x)}, \quad i = 1, \ldots, n, k \geq 2.
\]  
(18)
where $k$ is the current iteration times. $P^{(k-x)}_{s,i}$ is the propagation probability from node $s$ to node $i$ by $(k - x)$ times’ iteration. $P^{(x)}_{i}$ is the propagation probability from node $i$ to node $i$ by $x$ times’ iteration. Consequently, in the case of non-reinfection worms, we calibrate $I^{(k)}_{svq}$ to be (19):

$$I^{(k)}_{svq} = I^{(k)}_{svq} - E^{(k)}_{svq} \quad (k \geq 2).$$

### 3.5. Propagation ability (PA)

In real-world scenarios, attackers expect to control a significant proportion of a network to enable the worm propagation. The worm PA is related to the number of peers that the worm can propagate to with high probability and related time delay. In consideration of more than one path for the propagating worm, we adopt an accumulative $I$ (AI) to represent the sum of probabilities for the worm propagation between two peers with at most $k$ intermediate nodes. It is defined in (20):

$$AI = [a_{i1}, a_{i2}, \ldots, a_{i1}, \ldots, a_{in}]^T,$$

$$a_{i} = \frac{\sum_{k=0}^{n-2} \text{Re}(i^{(k)})}{n-1} a_{i}, \quad i \in [0, 1),$$

where function $\text{Re}(I^{(k)})$ is used to obtain the real component of $I^{(k)}$, $n$ indicates the number of nodes in the network and $(n-1)$ means the maximum number of intermediate nodes. In the propagation procedure, it is observed that the infected probability gradually decreases when the number of intermediate nodes increases.

Moreover, we define the accumulative time delay AT to represent the estimated time delay for the worm propagation between two peers with at most $k$ intermediate nodes, as shown in (21).

$$AT = [a_{t1}, a_{t2}, \ldots, a_{t1}, \ldots, a_{tn}]^T, a_{t} = \frac{2 \sum_{k=0}^{n-2} \text{Im}(i^{(k)}) \times \text{Re}(i^{(k)})}{(n-1) \sum_{k=0}^{n-2} \text{Re}(i^{(k)})} \quad a_{t}, \quad i \in [0, 1].$$

The condition to terminate propagation is that the matrix iteration count reaches $N - 2$ ($N$ nodes in a network). Since PA is a two-tuples (AI, AT), in order to evaluate the PA, we simply inspect the AI and AT for each node in the network after an iteration of propagation.

### 4. MODEL ANALYSIS

#### 4.1. Experiments environment

Our implementation is in Visual C++ 2008 SP1 and Matlab 7. The random numbers in our experiments are produced by the C++ TR1 library extensions. Experiments are carried out by a series of simulations: (1) we analyze the effect of the main components in our model, including $S$, $V$, $Q$. (2) we analyze the mutual effect from the impact factor $\beta$ between propagation probability and time delay. (3) In this paper we focus on scanning worms that primarily belong to the non-reinfection class of worms. Thus, we evaluate the errors caused by loops in worm propagation, which are normally ignored by macroscopic propagation models.

Some worms, such as Code Red [10], Code Red II [14–16] and Slammer [25], can propagate without a dependency of the topology. This means that an infectious node is able to infect an arbitrary vulnerable peer. Up to now, many researchers have modeled this type of worm propagation. In our experiments, we choose a typical local preference worm on the basis of Code Red II, as shown in Fig. 3. The time delay between each pair of nodes follows the Gaussian Distribution $N(0.5, 0.2^2)$.

In practice, there are problems to overcome in the propagation simulation. It often takes a significant amount of time to perform the experiments—72 h in our case on an Intel (R) Core (TM) i7 CPU 2.67-GHz (4 cores) processor to model 10 000 nodes—to simulate a single run of matrix iteration for one set of components $S$, $V$, $Q$. To identify trends, many such runs need to be performed and the whole simulation process has to be rerun for any parameter changes. The simulation overhead can be prohibitively high in some cases when the simulated network has a larger scale. This leads to the conclusion that all such experiments are intractable in practice. However, according to our practice and observation, we have found two properties of our model that can be used in addressing the difficulties stated earlier: (1) our model is based on matrix computation (see Equation (3)). Matrix multiplication has a computational cost, however, the matrix operations all run with a polynomial time complexity and can
be highly parallelized. Matrix multiplication is the bottleneck in our implementation and is an embarrassingly parallel problem, which means each resultant matrix element can be solved concurrently. Thus the performance of our system will increase significantly with the addition of concurrent computational resources. On a single workstation, we performed the evaluation using 2 × 4 threads (OpenMP threading Library) to improve the speed of matrix computation. The theoretical speedup is linear in an embarrassingly parallel problem such as matrix multiplication for most realistically sized clusters, which means the computational time is reduced linearly as more computational units are utilized. Industry and research organizations have access to significant computation resources and can mitigate the performance obstacles we have described by employing distributed and high performance computing resources such as clusters and clouds; (2) we analyze the impact of changing the matrix dimensionality used in the experiments and find that a larger dimension will not produce significantly different results. In order to show these results clearly, we choose reasonable network sizes (5000 nodes) and examine them under different scenarios.

4.2. Effect of propagation source vector

In this section, we assume that all nodes in the network are vulnerable and no nodes have been patched. According to the Symantec Internet Security Threat Report [27], global malicious activities are not evenly distributed in different ranges of IP addresses. Consequently, we arrange a group of scenarios with practical meaning in Table 2 to describe the different origins of worms. The results are represented by the mean value of PA (E(AI)), the variation of PA (D(AI)), the mean value of

### TABLE 2. Scenarios for analyzing propagation source (S).

<table>
<thead>
<tr>
<th>Scenario</th>
<th>Description (refer to Fig. 3)</th>
<th>Practical meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>IP address range $A_1B_1$ has increasing number of initial infectious nodes</td>
<td>Analyzing the impact of the number of initial infectious sources on the propagation probability in an IP address range such as a specific region</td>
</tr>
<tr>
<td>2</td>
<td>Increasing number of IP address ranges $A_1B_x$ $(x \in [1, g])$ have an initial infectious node</td>
<td>Analyzing the impact of different geographic distribution of initial infectious sources on the propagation probability</td>
</tr>
<tr>
<td>3</td>
<td>Increasing number of IP address ranges $A_xB_1$ $(x \in [1, g])$ have an initial infectious node</td>
<td></td>
</tr>
</tbody>
</table>

### TABLE 3. Results form different scenarios of propagation source (S).

<table>
<thead>
<tr>
<th>Scenario</th>
<th>Infectious node</th>
<th>Propagation probability</th>
<th>Time delay</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>$E(AI)$</td>
<td>$D(AI)$</td>
</tr>
<tr>
<td>1</td>
<td>$A_1B_1$ has 1% initial infectious node</td>
<td>0.0124</td>
<td>$0.0729 \times 10^{-4}$</td>
</tr>
<tr>
<td>2</td>
<td>$A_1B_1$ has 2% initial infectious node</td>
<td>0.0124</td>
<td>$0.0729 \times 10^{-4}$</td>
</tr>
<tr>
<td>3</td>
<td>$A_1B_1$ has 3% initial infectious node</td>
<td>0.0124</td>
<td>$0.0729 \times 10^{-4}$</td>
</tr>
<tr>
<td>2</td>
<td>$A_1B_1$ and $A_1B_2$ have 1% initial infectious node, respectively</td>
<td>0.0124</td>
<td>$0.0723 \times 10^{-4}$</td>
</tr>
<tr>
<td>3</td>
<td>$A_1B_1$, $A_1B_2$, and $A_1B_3$ have 1% initial infectious node, respectively</td>
<td>0.0124</td>
<td>$0.0704 \times 10^{-4}$</td>
</tr>
<tr>
<td>3</td>
<td>$A_1B_1$ and $A_2B_1$ have 1% initial infectious node, respectively</td>
<td>0.0124</td>
<td>$0.0320 \times 10^{-4}$</td>
</tr>
<tr>
<td>3</td>
<td>$A_1B_1$, $A_2B_1$, and $A_3B_1$ have 1% initial infectious node, respectively</td>
<td>0.0124</td>
<td>$0.0183 \times 10^{-4}$</td>
</tr>
<tr>
<td>4</td>
<td>$A_1B_1$ $(x \in [1, 4])$ have 1% initial infectious node, respectively</td>
<td>0.0124</td>
<td>$0.0115 \times 10^{-4}$</td>
</tr>
<tr>
<td>4</td>
<td>$A_1B_1$ has 2% infectious nodes and $A_1B_2$ has 1% infectious node</td>
<td>0.0124</td>
<td>$0.0724 \times 10^{-4}$</td>
</tr>
<tr>
<td>3</td>
<td>$A_1B_1$ has 2% infectious nodes and A2B1 has 1% infectious node</td>
<td>0.0124</td>
<td>$0.0365 \times 10^{-4}$</td>
</tr>
</tbody>
</table>

*Another scheme of deployment on the same quantity.*
propagation time delay (\(E(\text{AT})\)) and the variation of propagation time delay (\(D(\text{AT})\)). In order to describe the differences of each parameter clearly, we cut the first 81 nodes to make figures for some experiments.

4.2.1. Scenario 1

Preparation: We deploy 1% to 3% infectious nodes in \(A_1B_1\) of PM (see Fig. 3). Based on different propagation probabilities, the entire IP space is divided into three ranges:

(i) \(R_1: A_1B_1\)
(ii) \(R_2: A_1B_2 \rightarrow A_1B_x\)
(iii) \(R_3: A_2 \rightarrow A_x\)

Result: The result is listed in Table 3 Scenario 1. We find that the number of initial infectious nodes have no impact on \(E(\text{AI})\) and \(D(\text{AI})\). As shown in Fig. 4, \(\text{AI}\) in different IP ranges \(R_1\), \(R_2\) and \(R_3\) are overlapped respectively when the number of initial infectious nodes increases. In Fig. 4a, \(\text{AI}\) deviates in different IP ranges during the propagation procedure: 
\[\text{AI}(R_1) > \text{AI}(R_2) > \text{AI}(R_3)\]. In Fig. 4b, the difference of \(\text{AI}\) deviates in different IP ranges. Within the first 20 iterations, \(R_1\) and \(R_2\) decline rapidly, while \(R_3\) slightly increases. Afterwards, the difference of \(\text{AI}\) tends to be stable.

In Table 3, the result of time delay reflects temporal properties of the worm propagation in this scenario: an increasing number of initial infectious nodes results in a decrease in \(E(\text{AT})\) and \(D(\text{AT})\). Figure 5 shows the estimated time delay \(\text{AT}\) in different IP ranges when the number of initial infectious nodes increases. During the first nearly 40 iterations, \(\text{AT}(R_1) > \text{AT}(R_2) > \text{AT}(R_3)\); afterwards, \(\text{AT}\) in \(R_3\) goes up quickly: 
\[\text{AT}(R_3) > \text{AT}(R_2) > \text{AT}(R_1)\].

Analysis: Although the number of initial infectious nodes is increasing, their effects are limited in the same IP ranges, which leads to the overall propagation probabilities are not improved. Therefore the value of \(E(\text{AI})\) and \(D(\text{AI})\) stays the same.

When more infectious nodes are involved, the \(E(\text{AT})\) is obviously decreased as the average time for searching the targets is reduced. Meanwhile, a decline of \(D(\text{AT})\) indicates that an increase in the number of initial infected nodes can accelerate propagation speed to all nodes in the network since the time delay is close to the \(E(\text{AT})\). In the early propagation stage, the infected nodes are mainly in IP ranges \(R_1\) and \(R_2\). Thus, \(\text{AT}\) is dominated by the nodes with greater propagation probability in \(R_1\) and \(R_2\). Afterwards, when the number of infected nodes in \(R_3\) increases, the nodes in \(R_3\) have greater contribution to \(\text{AT}\).

4.2.2. Scenario 2

Preparation: We deploy 2 and 3% infectious nodes in PM (see Fig. 3). Based on different propagation probabilities, the entire IP space is divided into three ranges:

(i) \(R_1: A_1B_1 \rightarrow A_1B_2\) (2% infectious nodes)
\(A_1B_1 \rightarrow A_1B_3\) (3% infectious nodes)
(ii) \(R_2: A_1B_1 \rightarrow A_1B_x\) (2% infectious nodes)
(iii) \(R_3: A_2 \rightarrow A_x\)

Result: The result is listed in Table 3 Scenario 2. As shown in Fig. 6, the basic tendency of the curves is similar to Scenario 1. However, more infectious nodes (from 2 to 3%) in the network result in a decrease in \(\text{AI}\) of \(R_1\) and \(R_2\). Additionally, we find that the number of initial infectious nodes have no impact on \(\text{AI}\) in \(R_3\).

In Table 3, temporal properties of time delay in Scenario 2 stay the same with Scenario 1. As shown in Fig. 7, the value of \(\text{AT}\) decreases when the number of initial infectious nodes increases.

Analysis: We analyze the decrease of \(\text{AI}\) in \(R_1\) and \(R_2\) when there are more initial infectious nodes distributed in adjacent IP ranges of network. The reason is that when a new infectious node in \(A_1B_3\) is involved, compared with two infectious nodes...
case, the AI in $A_1B_1$ will increase. However, the sum of all probabilities is equal to one, which means an increase of AI in $A_1B_1$ results in mathematical decrease of AI in other infectious ranges such as $A_1B_1$–$A_1B_2$.

Similar to Scenario 1, $E(AI)$ stays the same (0.0124), only a small decrease of $D(AI)$ (from 0.0723 × 10^{-4} to 0.0704 × 10^{-4}) indicates that more nodes in the network have higher probabilities to be infected. Additionally, Scenario 2 has a same acceleration of propagation time as Scenario 1.

4.2.3. Scenario 3

**Preparation:** We deploy 2% to 4% infectious nodes in PM (See Fig. 3). Based on different propagation probabilities, the entire IP space is divided into three ranges:

(i) $R_1$: $A_1B_1 + A_2B_1$ (2% infectious nodes)
- $A_1B_1 + A_2B_1 + A_3B_1$ (3% infectious nodes)
- $A_1B_1 + A_2B_1 + A_3B_1 + A_4B_1$ (4% infectious nodes)

(ii) $R_2$: $\{A_xB_2 \rightarrow A_xB_g\}_{x=1,2}$ (2% infectious nodes)
- $\{A_xB_2 \rightarrow A_xB_g\}_{x=1,2,3}$ (3% infectious nodes)
- $\{A_xB_2 \rightarrow A_xB_g\}_{x=1,2,3,4}$ (4% infectious nodes)

(iii) $R_3$: $A_4 \rightarrow A_g$ (2% infectious nodes)
- $A_4 \rightarrow A_g$ (3% infectious nodes)
- $A_4 \rightarrow A_g$ (4% infectious nodes)

**Result:** We use 2% infectious nodes case to compare with Scenario 2. In Fig. 8, when the infectious nodes are scattered
in the network, the AI of $R_1$ and $R_2$ decreases. AI of $R_3$ stays the same.

In Table 3, temporal properties of time delay in Scenario 3 stay the same with Scenario 2. In Fig. 9, AT of Scenario 2 is almost same with AT of Scenario 3 in IP ranges $R_1$ and $R_2$.

**Analysis:** We analyze the decrease of AI in $R_1$ and $R_2$ when initial infectious nodes are scattered in different IP ranges of network. The reason is that when infectious nodes are deployed loosely, more nodes have higher probabilities to be infected. Similar to the exceptional decrease of AI in Scenario 2, an increase of AI in $A_2B_1 \rightarrow A_2B_g$ results in mathematical decrease of AI in other infectious ranges.

Additionally, Scenario 3 has a same acceleration of propagation time as Scenario 2 and 1.

### 4.2.4. Scenario 4

**Preparation:** We deploy 3% infectious nodes in PM (see Fig. 3). Based on different propagation probabilities, the entire IP space is divided into several ranges:

1. $R_1$: $A_1B_1$
2. $R_2$: $A_1B_2$ (3% infectious nodes)
3. $A_2B_1$ (3% infectious nodes)
4. $R_3$: $A_1B_3 \rightarrow A_1B_g$ (3% infectious nodes)
5. $A_1B_2 \rightarrow A_1B_g$ (3% infectious nodes)
6. $R_4$: $A_2B_2 \rightarrow A_2B_g$ (3% infectious nodes)
7. $R_5$: $A_3 \rightarrow A_g$

**Result:** The result is listed in Table 3 Scenario 4. In Fig. 10a, two infectious nodes are in $A_1B_1$; another infectious node is in $A_2B_1$. The result shows five IP address ranges have different AI.

In Table 3, scenario setting in Fig. 10a spends slightly more time to infect the nodes in the network than Fig. 10b (0.1732 compared with 0.1722). In Fig. 11, the value of AT is almost same when same proportion of initial infectious nodes are deployed in different IP ranges.

**Analysis:** We analyze the reason of four and five different ranges of AI. An infectious node has a larger effect on its own and adjacent IP ranges. A high density of initial infectious nodes has a greater effect on its own and adjacent IP ranges than other IP ranges with low density. Therefore, in Fig. 10a, $R_1$ (2% initial infectious nodes) has higher AI than $R_2$ (1% initial infectious nodes). In Fig. 10b, $R_3$ that is adjacent to $R_1$ (2% initial infectious nodes) has higher AI than $R_4$ that is adjacent to $R_2$ (1% initial infectious nodes).
4.2.5. Conclusion of propagation source effect

We draw conclusions of the practical meaning from different scenarios of propagation source.

(i) In Scenario 1, increasing number of initial infectious nodes in a specific region has no impact on propagation probability (AI) in the entire network. However, it accelerates the speed of worm propagation considerably.

(ii) Within a certain (20 in Scenario 1) number of intermediate nodes, the vulnerable nodes in adjacent IP ranges of infectious source have a greater probability to be infected.

(iii) In Scenario 2 and 3, different geographic distribution of initial infectious nodes has no impact on the overall AI. However, when the initial infectious nodes are more scattered in the network, they can infect more vulnerable nodes in the adjacent IP address ranges and accelerate the speed of worm propagation considerably in the network.

(iv) In Scenario 4, a high density of initial infectious nodes can infect more vulnerable nodes, which are mainly in adjacent IP address ranges of infectious source.

4.2.6. Inspiration for developing the patch strategy

The experiments on the propagation source vector (S) are mainly used to estimate where we need to patch.

(i) Where: According to the conclusion in this section, the best position for patching are similar or adjacent net blocks to the propagation source. In the real world, however, it is impractical to locate this position since the initial infectious nodes may be scattered and it is difficult to foresee the original propagation sources. On the basis of the conclusion from Scenario 4, the IP ranges with a high density of vulnerable nodes are essential areas in lieu of adjacent IP ranges of propagation source for patching since denser ranges have a greater possibility to be chosen as initial infectious sources. This may warrant collaboration across administrative boundaries when adjacent net blocks are not controlled by the same authority. It may be advantageous for network administrators to have a prior relationship with adjacent network owners to work together in threat intelligence and help prevent worm outbreaks and establish patch priorities in their own networks.

(ii) How many: The number of nodes that require patching is closely related to the different vulnerability distributions in the network. We will discuss it in the conclusion of next section.

(iii) When: Here, we will consider the estimated time of worm propagation in scanning worms. It is closely related to the propagation probability in the target IP ranges, but is unrelated to the geographic distribution of the propagation sources. In our experiments, when the percentage of the initial infectious nodes is from 1 to 4%, the range of propagation time delay is from 0.13t to 0.52ti. i is the scanning time of the entire IP address space.

4.3. Effect of vulnerable distribution vector

In this section, we assume that not all nodes are vulnerable and no nodes have been patched. Symantec examines the types of worms causing potential infections in each region [27]. The increasing regionalization of vulnerabilities is observed from one area to the next when vulnerabilities concern certain languages or localized events. Information about the geographic distribution of vulnerabilities can help network administrators improve their security efforts. Consequently, we arrange a group of scenarios with practical meaning in Table 4 to describe the different distributions of vulnerabilities. The results are represented by the mean value of PA (E(AI)), the variation of PA (D(AI)), the mean value of propagation time delay (E(AT)) and the variation of propagation time delay (D(AT)).

4.3.1. Scenario 1

Preparation: In Scenario 1, we assume vulnerability rate from 5 to 100% and its distribution follows uniform distribution. We fix the initial infectious nodes to 1.

Result: The result is described in Fig. 12. When the vulnerability rate is less than 80%, E(AI), D(AI), E(AT) and D(AT) remain at a low level. The change point is when the vulnerability rate is 80%. The steady AI occurs when the vulnerability rate is lower than 70%.

Analysis: We analyze the reason of the change point being at 80%. Drawing from the conclusions of Section 4.2, the nodes in adjacent IP ranges of the propagation origins have

![FIGURE 11. Propagation time delay in Scenario 4 (the first 81 nodes in 5000 nodes).](image-url)
Therefore, when the vulnerability rate is 80%, seldom nodes in non-adjacent IP ranges of the propagation origins are quickly infected, which contributes to E(AI). This is a reason why E(AI) and D(AI) are larger when the initial infectious nodes are deployed in vulnerability dense IP ranges.

Similar to Scenario 1, an increasing size of infected nodes in the network results in increasing time expenditure for overall worm propagation.

4.3.3. Inspiration of vulnerable distribution effect

The experiments on the vulnerable distribution vector (V) are mainly used to estimate how many nodes we need to patch.

(i) Where: If the threat is from a localized worm that exploits vulnerabilities in a specific region, it is of greater value to patch in the areas with a high density of vulnerabilities since the propagation is accelerated when more nodes are vulnerable.

(ii) How many: If the worm propagation is independent of the geographic region, the worm can infect a large number of nodes when the vulnerability rate is more than 80%. Making sure the vulnerability rate is lower than 80% can prevent the worm from propagating effectively. When the vulnerability rate is lower than 70%, the propagation probability remains stable and is significantly lower. A recommended patch strategy is to ensure the vulnerability rate is lower than 70%.

4.4. Effect of patch strategy vector

A large amount of money and labor are spent on patching the vulnerabilities each year. In order to reduce the cost, we focus on obtaining the most economic tactics for corporations patching their software vulnerabilities. In this section, we analyze the effect of patch strategy vector Q, which is used to eliminate the vulnerabilities in the vector V. Two scenarios are listed in Table 5. The results are represented by the mean value of PA (E(AI)), the variation of PA (D(AI)), the mean value of propagation time delay (E(AT)) and the variation of propagation time delay (D(AT)).

Table 4. Scenarios for analyzing vulnerability distribution (V).

<table>
<thead>
<tr>
<th>Scenario</th>
<th>Description</th>
<th>Practical meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Increasing percentage of vulnerable nodes and the vulnerabilities follow uniform distribution</td>
<td>Analyzing worm propagation when most of nodes are vulnerable without the difference of geographic distribution</td>
</tr>
<tr>
<td>2</td>
<td>Increasing percentage of vulnerable nodes and the vulnerabilities follow a Gaussian distribution. Initial infectious nodes are deployed in an IP address range that is rich of vulnerable nodes</td>
<td>Analyzing the impact of different geographic distribution of vulnerabilities on worm propagation</td>
</tr>
<tr>
<td>3</td>
<td>Increasing percentage of vulnerable nodes and the vulnerabilities follow a Gaussian distribution. Attackers deploy initial infectious nodes in sparse vulnerabilities ranges</td>
<td>Analyzing the impact of different deployment of propagation source under different distribution of vulnerabilities</td>
</tr>
</tbody>
</table>

a greater propagation probability to be infected. When the vulnerability follows a Uniform distribution, the coverage rate of vulnerable nodes in adjacent IP ranges of the propagation origins is small if the entire vulnerability rate is not large enough. Therefore, when the vulnerability rate is < 80%, seldom nodes in adjacent IP ranges are involved in the propagation and the nodes in non-adjacent ranges dominate the value of AI. When the vulnerability rate reaches 80% or more, more vulnerable nodes in adjacent IP ranges may be involved in the propagation, which lead to the E(AI) and D(AI) soar in Fig. 12.

When the vulnerability rate is more than 80%, the vulnerable nodes have a large probability to be infected. Thus, an increasing size of infected nodes in the network results in increasing time expenditure for overall worm propagation.

4.3.2. Scenario 2 and Scenario 3

Preparation: In Scenario 2 and 3, we investigate the impact of different geographic distribution of vulnerabilities on worm propagation. We also observe the impact of different deployments of the propagation source under different distributions of vulnerabilities. Therefore, we assume that vulnerabilities follow a Gaussian distribution from N(1024, 102^2) (10% vulnerability rate) to N(1024, 819^2) (80% vulnerability rate). We deploy one initial infectious node in vulnerability dense or sparse IP ranges.

Result: The result is described in Fig. 13. When more nodes in the network are vulnerable, E(AI) and D(AI) increase gradually in different deployments of initial infectious node. Obviously, if one initial infectious node is in vulnerability dense IP ranges, E(AI) and D(AI) are larger.

From Fig. 13, E(AT) and D(AT) have similar results to E(AI) and D(AI).

Analysis: More nodes in the network are infected when the vulnerability rate increases, which leads to E(AI) increasing smoothly. Since the vulnerabilities follow a Gaussian distribution, there are more vulnerable nodes in some specific IP ranges. If the initial infectious nodes are deployed in vulnerability dense IP ranges, the vulnerable nodes in adjacent IP ranges of the propagation origins are quickly infected, which
4.4.1. Scenario 1

**Preparation:** The intention of patching is to decrease the number of potential vulnerable nodes. When the patching rate increases, the vulnerability rate decreases. Initially, we assume that all nodes are vulnerable and fix one initial infectious node in the network.

**Result:** From Fig. 14, when the patching rate is higher than 20%, there is no obvious change in $E(AI)$. When the patching rate is higher than 30%, $D(AI)$ become steady. The change points of $E(AT)$ and $D(AT)$ are at a 10% patching rate.

**Analysis:** Once the patching rate reaches 20%, there are no obvious outcomes for more patching. Moreover, the outcomes of patching strategy become steady when the patching rate is more than 30%.

4.4.2. Scenario 2

**Preparation:** When vulnerabilities depend on geographic region, some specific IP ranges have more vulnerable nodes. Therefore, we arrange vulnerability to follow a Gaussian distribution. We assume that the vulnerability rate is 50 or 80% and fix one initial infectious node in the network. The patching rate varies between 5 and 40%.

**Result:** From Fig. 15, when the patching rate increases from 5 to 40%, $E(AI)$ and $D(AI)$ in vulnerable dense IP ranges decrease. The increasing patching rate has a greater effect on $E(AI)$ and $D(AT)$ with 80% vulnerability rate. Additionally, $E(AT)$ and $D(AT)$ have similar tendency.

**Analysis:** The objective of this scenario is to investigate the impact of patching rate on the specific vulnerable dense region.
15

FIGURE 13. Vulnerability in Gaussian distribution (Scenario 2 and 3).

TABLE 5. Scenarios for analyzing patching strategy $(Q)$.

<table>
<thead>
<tr>
<th>Scenario</th>
<th>Description</th>
<th>Practical meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Increasing percentage of patching nodes when vulnerabilities follow uniform distribution</td>
<td>Analyzing the effect of patch strategy when most of nodes are vulnerable without the difference of geographic distribution</td>
</tr>
<tr>
<td>2</td>
<td>Increasing percentage of patching nodes when vulnerabilities follow Gaussian distribution</td>
<td>Analyzing the effect of patch strategy when distribution of vulnerabilities depends on geographic region</td>
</tr>
</tbody>
</table>

When the specific region has more vulnerable nodes, the patch strategy has more effect.

4.4.3. Conclusion of patch strategy effect

The experiments on the patch strategy vector $(Q)$ are mainly used to estimate when we need to patch. In accordance with the conclusions regarding $S$ and $V$, we can summarize the patch strategies.

(i) Where: If the propagation sources can be predicted, the best strategy is to patch nodes that have the same class IP address as the infectious sources. However, in real-world scenarios, the propagation sources are
hard to locate. In these situations, the IP ranges with a high density of vulnerable nodes are essential areas for patching because more nodes are infected in these specific regions.

(ii) How many: The most economic patching rate is 20%, however we recommend a 30% patching rate because the outcome of this patch strategy is more stable.

(iii) When: “When do we patch?” is a complicated problem when considering global recommendations because it involves many social factors such as how widely used is the target software or the size of company. However, companies employing vulnerability management services can be given actionable recommendations of when it is critical to patch. For high-risk vulnerabilities, it is critical that networks reduce the number of vulnerable nodes to below 80%. Another actionable result is when to disclose information on the vulnerabilities. Most corporations such as Microsoft issue vulnerability patches for software products with some specific information of the nature of vulnerabilities. This ensures that the users are aware of the reason and necessity for deploying the patches. However, this information may be utilized by hackers to develop exploits for the vulnerabilities. Therefore, increased disclosure of specific vulnerabilities could possibly be delayed until the patching rate reaches at least 20%. Otherwise, the worms that target these vulnerabilities can propagate quickly to infect a large proportion of the network.

FIGURE 14. Patch strategy (Scenario 1).
4.5. Effect of the impact factor $\beta$

The impact factor $\beta$ reflects the impact of propagation time delay on the propagation probability. We introduced this parameter because the propagation time delay is caused by two factors: the worm’s infection strategy and network infrastructure information such as bandwidth. In 2001, Code Red v1 [16] used a static seed for its random number generator and thus generated identical lists of IP addresses on each infected machine. The first version of the worm spread slowly, because each infected machine began to spread the worm by probing the machines that were either infected or impregnable. Then, it was improved in Code Red v2 [16] through generating a random seed variant. This second version shared almost all of its code with the first version, but spread much more rapidly. Each node with an individual IP address may be scanned within a much shorter period of time and consequently the probability of each node to become infected is credibly increased. Therefore, a worm’s infection strategy has a significant effect on the spreading time.

On the other hand, in 2002, the Sapphire worm [28] randomly selected IP addresses to spread and reached its peak scanning rate of over 55 million scans per second across the Internet in under 3 min, but in later stages the rate of growth slowed down because networks became saturated with its scans and there was not enough bandwidth to allow the worm operate unhindered. It is therefore clear that a network environment with more bandwidth will accelerate the infection.

Since we do not know the exact value of $\beta$ for propagation in real worms, we assume that $\beta$ is equal to zero, which indicates the propagation probability cannot be affected by temporal properties in our previous simulations. However, in order to see
how the impact factor $\beta$ affects the propagation probability in worm spreading procedure, we compare the changes of AI with two different $\beta$ by assuming $\beta_1 = 0.25 \times 10^{-6}, \beta_2 = 0.5 \times 10^{-6}$.

**Preparation:** We deploy 1% infectious nodes in $A_1B_1$ of PM (see Fig. 3). We also assume that all nodes are vulnerable and no nodes are patched. Based on the different propagation probabilities, the entire IP space is divided into three ranges:

(i) $R_1: A_1B_1$
(ii) $R_2: A_1B_2 \rightarrow A_1B_g$
(iii) $R_3: A_2 \rightarrow A_g$

**Result:** As shown in Fig. 16, the propagation probabilities are initially almost the same for both $\beta_1$ and $\beta_2$ via 28 intermediate nodes. Later, however, the propagation probabilities decrease gradually. This matches the real spreading tendency in [10] quite well.

We also observe the effect of different impact factors step by step. In Fig. 17, after 60 hops the propagation probability approaches zero, which indicates that the worm theoretically propagates in a limited range of vulnerable nodes. This is in accordance with the real case analysis by [10]: when the propagation time delay largely increases because of network congestion, an infectious node cannot spread worms to the target. Therefore, the propagation probability is close to zero.

**Discussion:** The impact factor $\beta$ is to reflect the mutual impact between the propagation probability and time delay. When $\beta$ increases, from Fig. 16, time delay has a greater impact on the worm’s spreading, which results in a decrease of the propagation probability. If the value of $\beta$ increases continuously, the time delay will increase and worms will not be able to propagate to the target, which reflects real scenarios. Moreover, according to Figs 17 and 4b, we find that an increase of $\beta$ leads to the propagation probability decreasing gradually and tending towards zero. This also indicates that an increase in time delay results in a small propagation probability of the worm’s propagation. However, in the real world, each well-known worm has its own feature for propagation. How to formulate the value of $\beta$ to reflect the characteristic of propagation accurately is an issue of modeling worm’s propagation that we will address in the future.

**4.6. Discussion of the overestimation in macroscopic model**

Scanning worms infect targets by scanning the entire network and probing for vulnerable machines. Many researchers have studied and modeled various worms’ propagation using a variety of approaches and a number of different modeling techniques that address particular problems being examined. In this paper, we generalize previous works, such as [7–10, 12], as macroscopic models and propose our microscopic modeling method. Macroscopic models rely on differential equations to predict worm behavior and can effectively identify worms’ spreading tendency and infection scale along with the elapsed time. Our proposed microscopic model adopts matrix computation and focuses on presenting worms’ propagation procedure. In the remainder of this section, we will analyze the overestimation in traditional macroscopic modeling methods which can be avoided in the microscopic point of view, and thus is a key reason we chose the microscopic modeling approach.

Macroscopic methods model worms’ propagation through observing the current number of infected hosts and identifying the number of possible hosts for immediate and subsequent infection. These methods construct differential equations as a function of time $t$ to calculate the number of possible
hosts that can be infected in each time tick. The propagation analysis of macroscopic models starts from a group of infected nodes and this group is updated by conducting the propagation from infected nodes to uninfected vulnerable nodes, which are again used as initial infected nodes for propagation. This process continues as time elapses, ad infinitum. In our proposed microscopic model, we simulate worms’ propagation by constructing the spreading path from the initially infectious nodes to the targets via some intermediate nodes. According to the microscopic modeling and analysis of the propagation procedure, we have found an important source of inaccuracies in macroscopic modeling caused by propagation cycles (Section 3.4). These propagation cycles lead to overestimation in the macroscopic analysis of worms’ propagation. This is one of the reasons why we believe our microscopic model performs better than previous models.

In this paper, we focus on scanning worms that primarily belong to the non-reinfection class of worms. These types of worms, which include Code Red, can be infected only once in a worm outbreak. According to previous analyses, this leads to overestimation because of propagation cycles among the intermediate nodes. In this section, we use a simple scenario to analyze the errors.

**Preparation:** We deploy 1% infectious nodes in $A_1$ $B_1$ of PM (see Fig. 3). We also set all nodes to be vulnerable and set no patched nodes. Based on the different propagation probabilities, the entire IP space is divided into three ranges:

(i) $R_1: A_1 B_1$
(ii) $R_2: A_1 B_2 \rightarrow A_1 B_g$
(iii) $R_3: A_2 \rightarrow A_g$

**Result:** As shown in Fig. 18a, errors occur in different IP ranges during the propagation procedure: $\text{Errors}(R_1) > \text{Errors}(R_2) >> \text{Errors}(R_3)$. Within the first 20 iterations, $R_1$ increases rapidly, while $R_3$ keeps stable. In our microscopic model, we remove the errors. Fig. 18b shows the propagation probability $AI$ in different IP ranges before and after the removal of errors when the worm’s propagation is via some intermediate nodes. From the curves, we find the noticeable differences.

**Analysis:** Figure 18 demonstrates that non-trivial differences exist between macroscopic and microscopic models. This difference is accounted for by errors introduced by propagation cycles in the macroscopic model. According to (18), errors are mainly composed of two parts: the propagation probability from node $s$ to node $i$ when iterated $(k - x)$ times and the propagation probability from node $i$ to node $i$ when iterated $x$ times. In Fig. 18a, the errors have curves analogous to $AI$, but they are two magnitudes smaller ($10^{-3}$ compared with $10^{-1}$). In the experiments, similar results also exist in other scenarios. In Fig. 18b, we show that when a worm starts to propagate, more intermediate nodes are involved in the worm’s propagation. This results in the continuous and increasing formation of propagation. Thus the errors increase rapidly especially when the worm spreads via the first 20 intermediate nodes. Then, when more vulnerable nodes in the network are infected, the growth of propagation cycles tends to stabilize. Consequently, the errors increase slowly. After eliminating the errors, we find a clear difference in each IP region. Through the inspection of these errors, however, we can eliminate this negative effect by (19).

Moreover, in Fig. 18b, we can see noticeable differences between the macroscopic model and the microscopic model. Although the magnitude of errors is small ($10^{-3}$), we cannot regard them trivially when more initial infectious nodes or a larger network is involved. Especially for security companies, the errors can possibly mislead analysis on predicting the infected scale of the network and even cause a significant economic loss.
5. DISCUSSION AND OPEN ISSUES

Several limitations and open issues are worth discussing. First, the microcosmic model is not a complete substitute for the traditional macroscopic model of worm propagation. In order to give an insight into the change of propagation probability between nodes, the propagation source \( S \) in our model has been constructed according to different initial scenarios. Thus, \( S \) is static. However, in the traditional macroscopic models \([10, 11]\), the infectious state is a function of time \( t \) allowing the traditional models to reflect the changes during propagation. These two approaches model worm propagation from different perspectives and both are useful in worm analysis.

Second, our model employs an \( n \) by \( n \) square complex matrix to describe a network, which makes two arbitrary nodes adjacent. Thus, this representation is suitable for worms that scan the entire network and spread themselves to the target without regard to topological constraints. In the real world, some worms, such as email worms, are dependent on the topology of network in infecting targets. Our model cannot directly simulate these worms. However, if we assume the value of propagation probability in our proposed model as being either one or zero to indicate the existence or non-existence of a directed link between the nodes, then we can extend our model to simulate the topology-dependent worm propagation.

Third, many corporations prioritize the patching of various vulnerabilities on the basis of their own vulnerability ranking system. For example, vulnerabilities in firewalls should be patched as soon as possible because firewalls directly face the internet. Our microscopic model cannot describe this type of context-dependent information. We believe that this issue requires additional knowledge and is out of scope for this investigation.

Fourth, in this paper, we have not thoroughly investigated the impact factor \( \beta \) and the effect of errors. In fact, subtle changes in these may result in perceptible variances. This particularly happens in large-scale worm propagation. However, like the undiscovered parameter \( \alpha \) in [11], we do not know the exact value of \( \beta \) for real world worm propagation. More research and discussion will address these two factors in our future work.

Finally, in the experiments, we found that the overhead for the simulation is high. Given that the industry has existing infrastructure in clouds and cluster environments, accuracy in the worm propagation model is the key component to be addressed compared with the issue of time cost. In future work, we will employ more practical analysis of parallel algorithms to implement our model.

6. CONCLUSION

This paper presents a new exploration on modeling worm propagation processes. We discussed each component of the proposed model and were able to provide a series of recommendations and advice for patch strategies to counter worm propagation. Firstly, if the propagation sources can be predicted, the best strategy is to patch nodes that have IP addresses in the same net block. Otherwise, the IP ranges with a high density of vulnerable nodes are essential areas for patching. Secondly, for high-risk vulnerabilities, it is critical that networks reduce the number of vulnerable nodes to below 80%. Thirdly, increased disclosure of specific vulnerabilities could possibly be delayed until the patching rate reaches at least 20%.

The proposed theoretical design and experiments are based on typical scanning worms such as Code Red II. However, there are also topology-dependent worms that are actively used in the wild. Thus, our future work will mainly focus on extending the microcosmic model to include topology-dependent worms.

REFERENCES


