A Self-Learning Worm Using Importance Scanning

Zesheng Chen
School of Electrical & Computer Engineering
Georgia Institute of Technology
Atlanta, Georgia 30332
zchen@ece.gatech.edu

Chuanyi Ji
School of Electrical & Computer Engineering
Georgia Institute of Technology
Atlanta, Georgia 30332
jic@ece.gatech.edu

ABSTRACT

The use of side information by an attacker can help a worm speed up the propagation. This philosophy has been the basis for advanced worm scanning mechanisms such as hitlist scanning, routable scanning, and importance scanning. Some of these scanning methods use information on vulnerable hosts. Such information, however, may not be easy to collect before a worm is released. Questions then arise whether and how a worm can self-learn and use such information while propagating, and how virulent the resulting worm may be. In this paper, we design a self-learning worm using importance scanning. An optimal yet practical importance-scanning strategy is derived based on a new metric. A self-learning worm is demonstrated to have the ability to accurately estimate the underlying vulnerable-host distribution if a sufficient number of infected hosts are observed. Experimental results based on parameters chosen from Code Red show that after accurately estimating the distribution of vulnerable hosts, a self-learning worm can spread much faster than a random-scanning worm, a permutation-scanning worm, and a Class A routing worm. Some guidelines for detecting and defending against such self-learning worms are also discussed.

Categories and Subject Descriptors

D.4.6 [**Security and Protection**]: [Invasive software]

General Terms

Security

Keywords

worm propagation, self-learning worm, modeling, importance scanning

1. INTRODUCTION

A worm attacks vulnerable computer systems and employs self-propagating method to flood the Internet rapidly. Worms, such as Code Red [10], Slammer [9], and Witty [17], have infected hundreds of thousands of hosts and become a significant threat to network security and management. It is therefore of great importance for defenders to characterize the spread of worms that employ distinct scanning methods and to study countermeasures accordingly.

Different scanning methods have been employed by previous worms. For instance, Morris worm used *topological scanning* that relies on the information contained in the victim host to find new targets. Code Red v2 and Slammer worms employed *random scanning* that selects targets randomly. Code Red II and Nimda worms exercised *localized scanning* that preferentially searches for targets on the "local" address space.

Some advanced scanning mechanisms have been developed based on such a philosophy: *The use of side information by an attacker can help a worm speed up the propagation.* For example, *hitlist scanning* collects a list of vulnerable hosts before the worm is released [20]. *Flash worm* is an extreme case of hitlist-scanning worms, where IP addresses of all vulnerable hosts are known in advance [19]. *Routable scanning* exploits the information provided by BGP routing tables [26, 22]. *Importance scanning* takes advantage of the knowledge of vulnerable-host group distribution, assuming that this distribution is either available or obtainable [2].

In the Internet, however, it may not be easy for attackers to collect information on vulnerable hosts. For example, Windows SQL database servers do not advertise their addresses [9, 26]. It is therefore difficult for Slammer to obtain a list of vulnerable hosts or an underlying vulnerable-host distribution before the worm is released. Nevertheless, future worms can become more intelligent and potentially learn a certain knowledge about, e.g. the vulnerable-host distribution, while propagating. In this work, we focus on self-learning worms and intend to answer the following questions:

- How can a worm self-learn about a vulnerable-host distribution and make use of such information while propagating?
- How fast can a self-learning worm spread?

First, we derive an optimal *static importance scanning* method assuming that a vulnerable-host group distribution is given. The *dynamic importance scanning* proposed in [2] is optimal in terms of worm propagation speed under the same assumption. But such an optimal solution is not realistic, since it requires a lot of information exchange among infected hosts. The proposed static importance scanning

WORM'05, November 11, 2005, Fairfax, Virginia, USA.

is practical as it constrains the information exchange, and optimal as it is derived based on a new metric that characterizes the effectiveness of scanning strategies. This metric reflects the average number of worm scans required until the first scan hits a randomly-chosen vulnerable host. We then show the propagation characteristics of different static importance-scanning methods through an extended Analytical Active Worm Propagation (AAWP) model [3, 2].

Next, we design a self-learning worm without the knowledge of vulnerable-host group distribution before spreading. Such worm intends to use importance scanning but avoids information exchange among infected hosts. A key capability of this worm is to learn an underlying vulnerable-host group distribution. We show that the worm can accurately estimate the group distribution through a simple proportion estimator if a sufficient number of IP addresses of infected hosts can be collected. We consider the group distribution of web servers as an example of vulnerable-host distribution in /8 subnets. We then show that a self-learning worm based on parameters chosen from real measurements can spread far faster than a random-scanning worm, a permutation-scanning worm, and a Class A routing worm after estimating the group distribution of vulnerable hosts.

Finally, we provide a few guidelines for detecting and defending against the self-learning worms: (1) A new application should be uniformly deployed in the future Internet from the view of game theory between a self-learning worm and a defender. (2) Defenders from different domains should share information with each other and cooperate to build up distributed worm detection systems. (3) An information collection and process center of a self-learning worm system, shown in this paper, needs to be detected and disabled early.

The remainder of this paper is structured as follows. In Section 2, we introduce the concept of importance scanning and the notations used in this paper. In Section 3, we characterize the static importance-scanning strategies through theoretical analysis and experiments. We then design a self-learning worm in detail and compare it with a random-scanning worm, a permutation-scanning worm, and a Class A routing worm in Section 4. We further discuss some guidelines for detecting and defending against such self-learning worms in Section 5. We conclude this paper in Section 6 with a brief summary and an outline of future work.

2. IMPORTANCE SCANNING

Importance scanning is inspired by importance sampling in statistics [2]. Importance sampling is used to reduce the sample size for accurately estimating the probability of rare events [18, 7, 5]. Xing et al. employed the principle of importance sampling to measure the size and the growth of the Internet [24]. Chen et al. designed a fast spreading worm based on the spirit of importance sampling [2]. Both works regard probing/scanning as sampling. That is, probing a target to find an information server or a vulnerable host is equivalent to obtaining a sample in IP address space. If servers or vulnerable hosts are distributed non-uniformly, importance sampling can sample the IP address space according to their distributions. This enables importance sampling to reduce the number of samples (probes/scans) needed for accurately estimating the number of servers or quickly attacking a large number of vulnerable hosts.

When worm scanning methods are considered, random scanning is equivalent to a Monte Carlo method, which samples randomly-chosen targets in IP address space. In contrast, importance scanning samples targets according to an underlying group distribution of vulnerable hosts [2]. The division of groups can follow different criteria, such as Domain Name System (DNS) Top-Level Domains, countries, Autonomous Systems, IP prefixes in Classless Inter-Domain Routing (CIDR), first byte of IP addresses (/8 subnets), or first two bytes of IP addresses (/16 subnets). A key observation is that the vulnerable-host distributions in these groups are highly non-uniform [10, 9, 17, 14, 2], which can be exploited by importance-scanning strategy. Importance scanning concentrates on scanning groups that are more likely for worms to find vulnerable hosts.

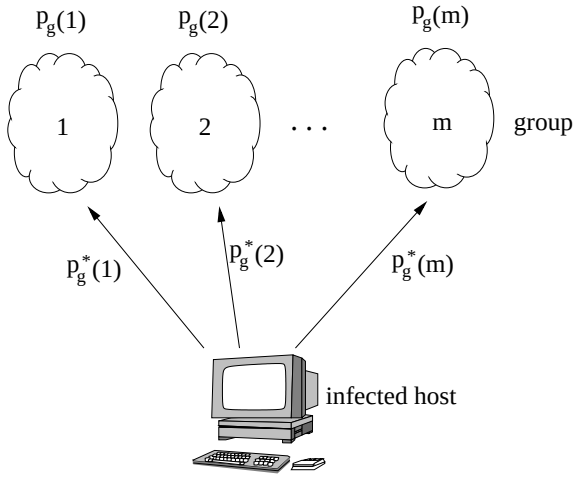


Figure 1: Illustration of importance scanning.

Specifically, we assume that the Internet is composed of m groups, as illustrated in Figure 1. Let N denote the total number of vulnerable hosts in the Internet. Let N_i and Ω_i $(i = 1, 2, \cdots, m)$ denote the number of vulnerable hosts and the size of address space in group i, respectively. Thus, $\sum_{i=1}^{m} N_i = N$ and $\sum_{i=1}^{m} \Omega_i = 2^{32}$. We define *group distribution* of vulnerable hosts, $p_g(i)$ $(i = 1, 2, \cdots, m)$, as the ratio between the number of vulnerable hosts in group i and the total number of vulnerable hosts, i.e., $p_g(i) = \frac{N_i}{N}$. We define *group scanning distribution*, $p_g^*(i)$ $(i = 1, 2, \cdots, m)$, as the probability that a worm scan hits group i. Thus, $\sum_{i=1}^{m} p_g(i) = 1$ and $\sum_{i=1}^{m} p_g^*(i) = 1$. Table 1 shows the notations used throughout this paper.

The choice of $p_g^*(i)$'s is essential to the effectiveness of importance scanning. There are two types of importance scanning: *dynamic importance scanning* if $p_g^*(i)$'s vary with time, and *static importance scanning* if $p_g^*(i)$'s are fixed at all time. In [2], the optimal dynamic importance scanning is derived, assuming that the group distribution and the total number of vulnerable hosts are known in advance. At each time step, this optimal solution forces all infected hosts to concentrate on scanning the group where it is most likely for the worm to find an uninfected vulnerable host. This strategy, however, is not realistic, since it requires each infected host to know the number of uninfected vulnerable hosts in each group at every time step, and thus leads to a lot of information exchange among infected hosts. In this paper, we focus on static importance-scanning strategies, while using the optimal dynamic importance-scanning strategy as a per-

Table 1: Notations used throughout this paper.

Notation	Explanation
N	Total number of vulnerable hosts
m	Number of groups in the Internet
N_i	Number of vulnerable hosts in group i
Ω_i	Size of address space in group i
$p_g(i)$	Group distribution: Percentage of vulnerable hosts in group i
$p_g^*(i)$	Group scanning distribution: Probability of a worm scan hitting group i

formance upper-bound for comparison. Here "performance" refers to the propagation speed of worms. If a worm spreads faster, it has better performance.

3. STATIC IMPORTANCE SCANNING WITH GROUP DISTRIBUTION

In this section, we assume that the group distribution is given. We first derive the optimal static importance-scanning strategy. We then extend the AAWP model [2] to model the spread of static importance-scanning worms. Finally, we use a web-server distribution in /8 subnets as an example of vulnerable-host group distribution to compare the propagation speed of static importance-scanning strategies.

3.1 Optimal Static Importance Scanning

When a worm scan hits group i ($i \in \{1, 2, \cdots, m\}$), Ω_i hosts in this group are targeted by that scan with the same likelihood. That is, when considering a vulnerable host in group i, it has a probability of $\frac{1}{\Omega_i}$ to be hit by a worm scan given that the scan hits the group. Thus, a vulnerable host in group i is hit by an importance-scanning worm scan with probability

$$p_h(i) = p_g^*(i) \cdot \frac{1}{\Omega_i}. \tag{1}$$

Since the events of a vulnerable host being hit are assumed to be independent in static importance scanning, the number of scans required until the first scan hits an appointed vulnerable host in group i, denoted by X_i, follows a geometric distribution [15]

$$P(X_i = j) = p_h(i)(1 - p_h(i))^{j-1}, \quad j = 1, 2, \cdots. \tag{2}$$

Then, the expected number of scans needed until this vulnerable host is hit is

$$E[X_i] = (p_h(i))^{-1} = \frac{\Omega_i}{p_g^*(i)}. \tag{3}$$

Therefore, if we randomly choose a vulnerable host in the Internet, the average number of scans required until the first scan hits this host, denoted by Y, is

$$Y = \frac{1}{N}\sum_{i=1}^{m} N p_g(i)\frac{\Omega_i}{p_g^*(i)} = \sum_{i=1}^{m}\frac{\Omega_i p_g(i)}{p_g^*(i)}, \tag{4}$$

where $Np_g(i)$ is N_i, the number of vulnerable hosts in group i. Intuitively, a good metric to measure the effectiveness of scanning strategies is the average number of scans required for hitting all vulnerable hosts divided by the number of vulnerable hosts. An expression of this metric, however, is complex and difficult to obtain. Instead, Y gives an alternative metric to reflect the effectiveness of scanning strategies. A better static importance-scanning strategy leads to a smaller Y. Thus, the goal of the static importance scanning is to minimize Y. The optimal solution can be found by Lagrangian optimization of Y as shown in the following theorem.

THEOREM 1. *Among all possible static importance-scanning strategies, the group scanning distribution $\tilde{p}_g^*(i)$ is the strategy that minimize Y subject to $\sum_{i=1}^{m} p_g^*(i) = 1$, where*

$$\tilde{p}_g^*(i) = \frac{\sqrt{\Omega_i p_g(i)}}{\sum_{k=1}^{m}\sqrt{\Omega_k p_g(k)}}. \tag{5}$$

PROOF: The optimal static importance-scanning strategy can be found by minimizing Y. Let the Lagrangian objective function be

$$J = \sum_{i=1}^{m}\frac{\Omega_i p_g(i)}{p_g^*(i)} + \lambda\left(\sum_{i=1}^{m} p_g^*(i) - 1\right). \tag{6}$$

For each group i, differentiating with respect to $p_g^*(i)$ and setting the result equal to zero yield $\tilde{p}_g^*(i) = \sqrt{\frac{\Omega_i p_g(i)}{\lambda}}$. The constraint $\sum_{i=1}^{m}\tilde{p}_g^*(i) = 1$ gives $\lambda = \left(\sum_{i=1}^{m}\sqrt{\Omega_i p_g(i)}\right)^2$, which leads to Equation (5). Since $\nabla^2 J\left(p_g^*(i)\right) \geq 0$, $\tilde{p}_g^*(i)$ is the optimal static importance-scanning strategy that minimize Y. ∎

Putting $\tilde{p}_g^*(i)$ into Equation (4), we obtain

$$\tilde{Y}_{min} = \left(\sum_{i=1}^{m}\sqrt{\Omega_i p_g(i)}\right)^2 \leq \left(\sum_{i=1}^{m}\Omega_i\right)\left(\sum_{i=1}^{m} p_g(i)\right) = 2^{32}. \tag{7}$$

The above inequality is derived by Cauchy-Schwarz Inequality and holds when $\frac{p_g(1)}{\Omega_1} = \frac{p_g(2)}{\Omega_2} = \cdots = \frac{p_g(m)}{\Omega_m} = \frac{1}{2^{32}}$, i.e., the vulnerable hosts are uniformly distributed in the Internet. When $\Omega_1 = \Omega_2 = \cdots = \Omega_m = \frac{2^{32}}{m}$, $\tilde{p}_g^*(i) = \frac{\sqrt{p_g(i)}}{\sum_{k=1}^{m}\sqrt{p_g(k)}}$ and $\tilde{Y}_{min} = \frac{2^{32}\times\left(\sum_{i=1}^{m}\sqrt{p_g(i)}\right)^2}{m}$.

3.2 Worm Propagation Model for Static Importance-Scanning Worms

To compare the performance of different static importance-scanning strategies, we need to model the spread dynamics of static importance-scanning worms. The AAWP model, as applied in [3, 14, 2], is extended here for our purpose.

Let I_t denote the total number of infected nodes at time t ($t \geq 0$). Then, during the time period $[t, t+1)$ the number of scans hitting group i is $sI_t p_g^*(i)$, where s is the scanning rate

of the worm. For group i, since each address is scanned with the same likelihood of $\frac{1}{\Omega_i}$, the expected number of infected hosts at time $t+1$ can be derived as:

$$I_{t+1,i} = I_{t,i} + (N_i - I_{t,i}) \left[1 - (1 - \frac{1}{\Omega_i})^{sI_t p_g^*(i)} \right], \quad (8)$$

where $i = 1, 2, \cdots, m$; $I_{0,i}$ is the number of initially-infected hosts in group i; and $I_t = \sum_{i=1}^{m} I_{t,i}$.

For static importance-scanning strategies, assuming that $\Omega_1 = \Omega_2 = \cdots = \Omega_m = \frac{2^{32}}{m}$, we can relate the group scanning distributions $p_g^*(i)$ with the group distributions $p_g(i)$ in the following formula:

$$p_g^*(i) = \frac{(p_g(i))^n}{\sum_{k=1}^{m} (p_g(k))^n} \propto (p_g(i))^n . \quad (9)$$

When $n = \frac{1}{2}$, $p_g^*(i)$ is the optimal static importance-scanning strategy that minimizes Y.

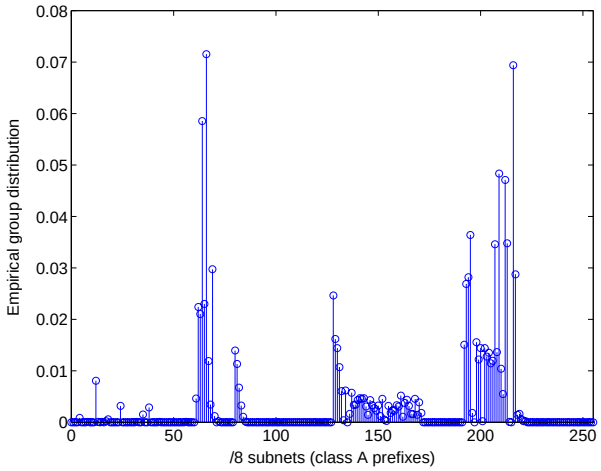
3.3 Comparison of Static Importance-Scanning Strategies

We use the web-server (port 80) distribution as an example of vulnerable-host group distribution to compare the performance of static importance-scanning strategies. To estimate the distribution of web servers, we exploited a random Uniform Resource Locator (URL) generator from UROULETTE (http://www.uroulette.com/) to collect 13,866 IP addresses of web servers on January 24, 2005. An *empirical group distribution* based on the first byte of IP addresses (/8 subnets) is then formed as

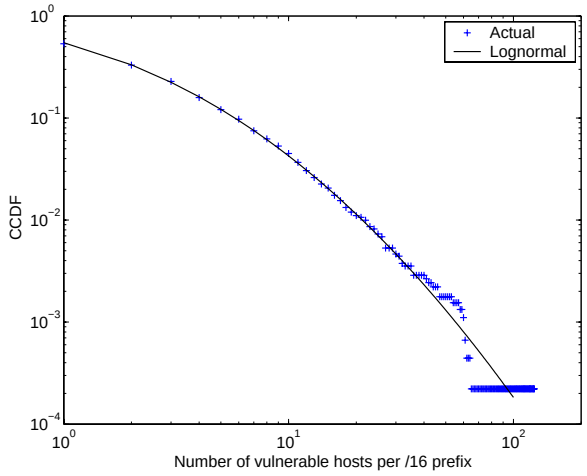
$$p_e(i) = \frac{\text{number of addresses with the first byte equal to } i}{\text{total number of collected addresses}}, \quad (10)$$

where $i = 0, 1, \cdots, 255$. The results are plotted in Figure 2(a), showing that the web-server distribution is far from being uniform. We further plot the Complementary Cumulative Distribution Function (CCDF) of the web-server distribution in /16 subnets in log-log scales in Figure 2(b) for collected data. CCDF is defined as the fraction of the /16 subnets with the number of vulnerable hosts greater than d. We find that a lognormal distribution with mean 0.15 and standard deviation 1.25 closely fits these measurement data. This indicates that the distribution of web servers nearly follows a *power law* distribution. Similar observations are shown in [14]. Since the empirical group distribution $p_e(i)$ gives the relative distribution of web servers as a function of the first byte value of IP addresses, we assume $p_g(i) = p_e(i)$ for our experiments in this paper. This assumption is also applied in [24, 2].

We use Code Red v2 as a worm example, which has a vulnerable population N=360,000 and a scanning rate s=358 per minute [10, 25]. We also assume a hitlist of 10 (*i.e.*, $I_0 = 10$). Equations (8) and (9) are used to model the spread of worms that employ the static importance scanning. Figure 3 shows the propagation speed of different static importance-scanning strategies ($n = \frac{1}{3}, \frac{1}{2}, 1, 2$) as well as the optimal dynamic importance-scanning (IS) method [2]. The experiments stop when 99% vulnerable hosts are infected. As expected, when $n = \frac{1}{2}$, static IS infects 99% vulnerable hosts in the shortest time duration among all static strategies. One interesting observation is that if a static strategy (such as $n = 2$) spreads faster at the early stage, it will propagate slower at the late stage; or vise versa (such as $n = \frac{1}{3}$).



(a) Empirical group distribution of web servers (/8 subnets).



(b) Web-server distribution (/16 subnets) in log-log scale and lognormal curve fitting.

Figure 2: Uneven distribution of web servers.

This is because a static IS uses the same group scanning distribution all the time. Larger n leads to that an IS worm preferentially scans the groups containing more vulnerable hosts at the early stage, but unfavorably probes the groups having more left vulnerable hosts at the late stage. Therefore, attackers may choose a corresponding static IS strategy based on the purpose of attacks, e.g. infecting some amount of hosts as quickly as possible.

4. A SELF-LEARNING WORM WITHOUT GROUP DISTRIBUTION

We now assume that the knowledge of the group distribution is not available before a worm starts to spread. We then focus on a self-learning worm that learns the distribution while propagating.

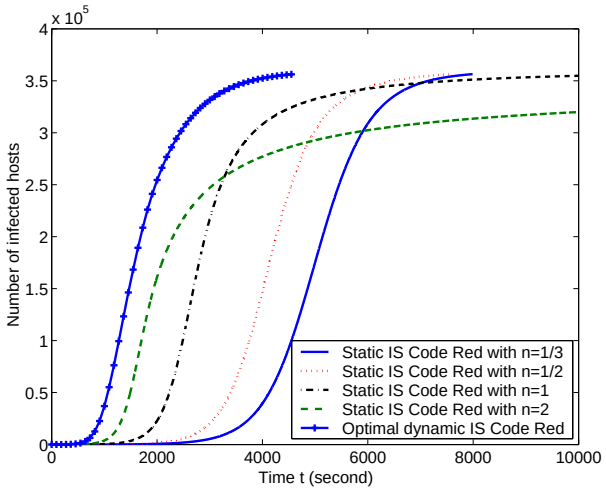


Figure 3: Comparison of static importance-scanning (IS) strategies.

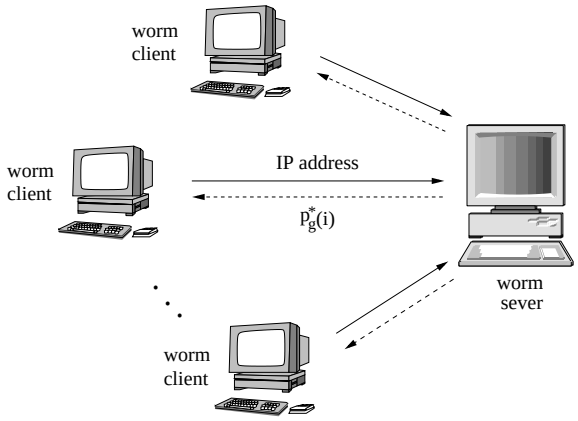


Figure 4: A self-learning worm system.

4.1 A Self-Learning Worm

For practicality, we assume that learning takes place using as less information exchange among hosts as possible. Such a constructed worm system is shown in Figure 4. A host with a high Internet bandwidth capacity, called *worm server*, is responsible for collecting and processing information about the IP addresses of infected hosts. An infected host is called *worm client* and may communicate with the worm server, but not with other infected hosts. If the communication uses Internet Relay Chat (IRC), this worm system forms a Botnet [4, 13].

The propagation process of this self-learning worm can be divided into two stages:

- **Learning stage:** Each infected host (worm client) performs random scanning or routable scanning [26, 22]. Once a vulnerable host is infected and becomes a new worm client, it reports its IP address to the worm server. The worm server records the clients' IP addresses in a list. When the worm server records a sufficient number of IP addresses, it estimates the group distribution of the vulnerable hosts ($p_g(i)$) based on collected data, and sends the corresponding group scanning distribution ($p_g^*(i)$) to all worm clients on the list.
- **Importance-scanning stage:** Upon receiving $p_g^*(i)$, a worm client switches from either random scanning or routable scanning to importance scanning using $p_g^*(i)$. The newly-infected hosts at this stage do not need to communicate with the worm server, but perform importance scanning directly.

This worm system is simple and effective, behaving in a similar way to the query process in Napster peer-to-peer system [16].

4.2 Estimating the Group Distribution

The performance of our designed self-learning worm strongly depends on how the worm server accurately estimates the group distribution of vulnerable hosts. Let L denote the number of clients' IP addresses collected on the worm server. Here, we attempt to answer the question: how large should L be for accurately estimating the group distribution?

Let L_i denote the number of worm clients' IP addresses from group i among all L addresses. Then, a simple proportion estimator for group i distribution is

$$\hat{p}_g(i) = \frac{L_i}{L}. \tag{11}$$

Let Z_j $(j = 1, 2, \cdots, L)$ denote the event that the jth worm client is in group i,

$$Z_j = \begin{cases} 1, & \text{if the } j\text{th worm client is in group } i; \\ 0, & \text{otherwise.} \end{cases}$$

Thus, $\sum_{j=1}^{L} Z_j = L_i$. Since the worm uses random scanning or routable scanning in the learning stage of worm propagation, Z_j follows a Bernoulli distribution with parameter $p_g(i)$. Then, $E[Z_j] = p_g(i)$ and $Var[Z_j] = p_g(i)\,(1 - p_g(i))$. Thus,

$$E[\hat{p}_g(i)] = E\left[\frac{\sum_{j=1}^{L} Z_j}{L}\right] = \frac{1}{L}\sum_{j=1}^{L} E[Z_j] = p_g(i). \tag{12}$$

This means that the estimator is unbiased, which is desirable. When $j \neq k$, $E[Z_j Z_k] = P(Z_j = 1, Z_k = 1) = P(Z_j = 1)P(Z_k = 1|Z_j = 1) = \frac{N_i}{N} \cdot \frac{N_i - 1}{N - 1}$ and $E[Z_j]E[Z_k] = (p_g(i))^2 = (\frac{N_i}{N})^2$. Thus,

$$\begin{aligned} Cov[Z_j, Z_k] &= E[Z_j Z_k] - E[Z_j]E[Z_k] && (13) \\ &= -p_g(i)\frac{1 - p_g(i)}{N - 1}, && (14) \end{aligned}$$

which leads to

$$\begin{aligned} Var[\hat{p}_g(i)] &= Var\left[\frac{\sum_{j=1}^{L} Z_j}{L}\right] && (15) \\ &= \frac{\sum_{j=1}^{L} Var[Z_j] + 2\sum_{j<k} Cov[Z_j, Z_k]}{L^2} && (16) \\ &= \frac{1}{L} \cdot \frac{N - L}{N - 1} \cdot p_g(i)(1 - p_g(i)). && (17) \end{aligned}$$

The estimation error between the actual group distribution and the estimated group distribution is defined as in [1]

$$e = \sum_{i=1}^{m} (\hat{p}_g(i) - p_g(i))^2. \tag{18}$$

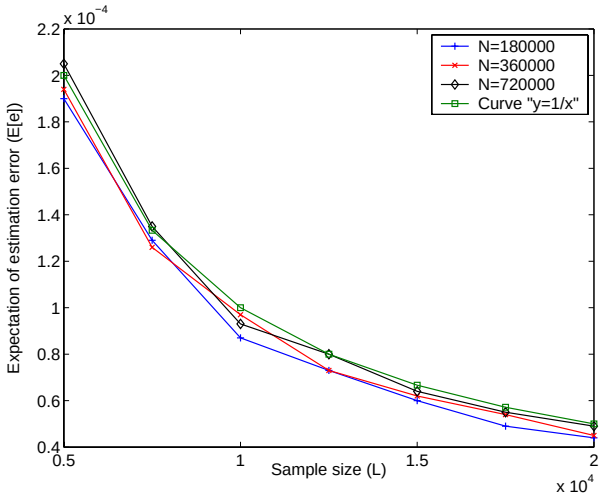


Figure 5: Expected estimation error of proportion estimator.

Then, the expected estimation error is

$$E[e] = E\left[\sum_{i=1}^{m}(\hat{p}_g(i) - p_g(i))^2\right] \quad (19)$$

$$= \sum_{i=1}^{m} Var[\hat{p}_g(i)] \quad (20)$$

$$= \frac{1}{L}\cdot\frac{N-L}{N-1}\cdot\left(1-\sum_{i=1}^{m}p_g^2(i)\right). \quad (21)$$

Since $\sum_{i=1}^{m} p_g^2(i)\cdot\sum_{i=1}^{m} 1^2 \geq \left(\sum_{i=1}^{m} p_g(i)\right)^2$ by Cauchy-Schwarz Inequality, $\sum_{i=1}^{m} p_g^2(i) \geq \frac{1}{m}$. Therefore,

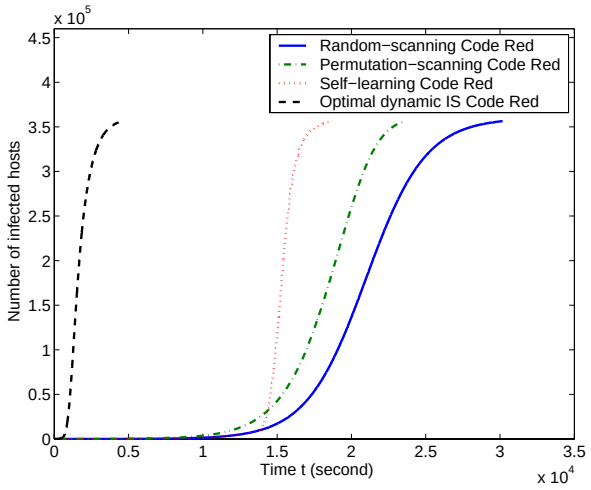
$$E[e] \leq \frac{1}{L}\cdot\frac{N-L}{N-1}\cdot\frac{m-1}{m} \leq \frac{1}{L}. \quad (22)$$

This means that we can choose the number of samples L to achieve a desired accuracy of estimation. For example, if $L \geq 10^4$, we have $E[e] \leq 10^{-4}$.

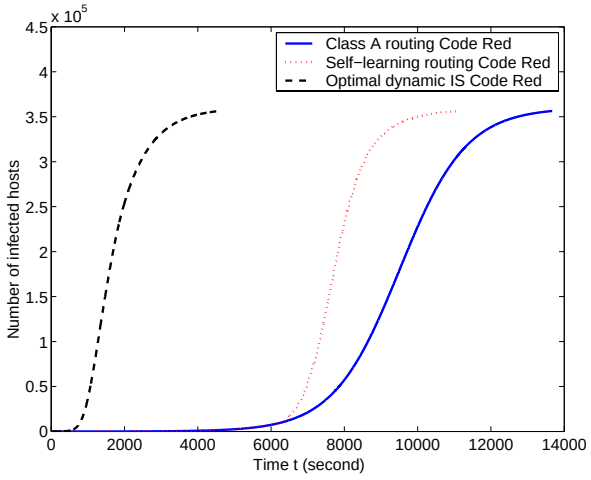
To examine the tightness of the bound, we simulate the random-scanning worm propagation, assuming that the vulnerable hosts follow the same group distribution as web servers shown in Figure 2(a). When L hosts are infected, the number of infected host in each /8 subnet is counted and the proportion estimator is performed using Equation (11). The estimator errors are averaged over 100 runs. The simulation results are shown in Figure 5, where x axis is the sample size (L) and y axis is the expected estimation error ($E[e]$). We compare the results of different sizes of vulnerable population (N=180000, 360000, 720000) with the curve "$y = \frac{1}{x}$". It is shown that the bound on the expected estimation error is tight and rather accurate for a wide range of L.

4.3 Performance Evaluation

How much does the self-learning process help a worm in speeding up the propagation? To answer this question, we compare the propagation speed of self-learning worms with that of an optimal dynamic importance-scanning worm, a random-scanning worm, a permutation-scanning worm, and a Class A routing worm. The simulated worms have the parameters comparable to those of Code Red v2, which has a vulnerable population N=360,000 and a scanning rate s=358 per minute [10, 25]. We assume a hitlist of 10 (i.e., $I_0 = 10$). The experiments stop when 99% vulnerable hosts are infected. We employ the AAWP model [3] to study the propagation of a self-learning worm in the learning stage and the model in Equation (8) to imitate the spread in the importance-scanning stage with the optimal static importance-scanning strategy $p_g^*(i) = \frac{\sqrt{\Omega_i p_g(i)}}{\sum_{k=1}^{m}\sqrt{\Omega_k p_g(k)}}$. In our experiments, the self-learning worm switches from the learning stage to the importance-scanning stage when the worm server observes 10,000 worm clients. Since the expected estimation error of group distribution is less than 10^{-4}, we assume that the worm server can accurately estimate the underlying group distribution $p_g(i)$ at the end of the learning stage.



(a) A self-learning Code Red.



(b) A self-learning routing Code Red.

Figure 6: Performance of self-learning Code Red worms.

Here we also use the web-server distribution in /8 subnets shown in Figure 2(a) as an example of vulnerable-host group distribution.

Figure 6(a) shows the propagation comparison among a self-learning Code Red, a permutation-scanning Code Red, and a random-scanning Code Red. In permutation scanning, all worms share a common pseudo random permutation of the IP address space and coordinate to provide comprehensive scanning [21]. Such a permutation scanning is implemented by Weaver's simulator, which uses a 32-bit, 6-round variant of RC5 to generate all permutations and random number. Compared with a permutation-scanning worm, a self-learning worm spreads slower in the learning stage, but propagates much faster in the importance-scanning stage. Figure 6(b) demonstrates the spread of another self-learning Code Red if the worm uses the Class A routable scanning in the learning stage. A Class A routable-scanning worm reduces the scanning space to 45.3% of the entire IPv4 address space [26]. It is noted that self-learning Code Red worms spend much time on the learning stage to infect the first 10,000 hosts. After collecting the information of 10,000 worm clients, the self-learning worms use only 81 minutes to infect the rest about 350,000 vulnerable hosts in the importance-scanning stage. In comparison, a random-scanning Code Red, a permutation-scanning Code Red, and a Class A routing Code Red need 271 minutes, 194 minutes, and 123 minutes, respectively, to finish the infection. Hence, a simple self-learning process can greatly increase worms' spreading speed.

5. DETECTING AND DEFENDING AGAINST SELF-LEARNING WORMS

How can we detect and defend against self-learning worms? Our study on self-learning worms provides the following guidelines:

- When a new application is introduced to the future Internet, how can we deploy this application? From Equation (4), attackers attempt to minimize Y by choosing the optimal static group scanning distribution $p_g^*(i)$, while defenders endeavor to maximize Y by customizing the group distribution $p_g(i)$. This is a classic two-person zero-sum game [12] between the attackers and the defenders, which leads to

$$Y_{opt} = \min_{p_g^*(i)} \max_{p_g(i)} \{Y\} = \max_{p_g(i)} \min_{p_g^*(i)} \{Y\}. \quad (23)$$

  From the derivation in Section 3.1, we see that the optimal strategy for the defenders is to deploy a new application uniformly in the Internet for any grouping criteria, such as /8 subnets, /16 subnets, and DNS Top-Level Domains [6]. Thus, the self-learning process cannot help the worm in speeding up the propagation. It is a common belief that IPv6 can slow down the spread of scanning worms effectively due to the large address space. An importance-scanning worm, however, can have an astonishing spreading speed, if vulnerable hosts are still distributed in a non-uniform fashion and the group distribution can be obtained. A similar observation has also been pointed out using a different metric in [2]. On the other hand, current traffic engineering requires non-uniform partition of the address space for routing aggregation. How to balance the tradeoff between traffic engineering and security engineering is a challenging task for designing the future Internet.

- Since a self-learning worm has an astounding spreading speed at the importance-scanning stage, defenders need to detect the worm during the learning stage of worm propagation. Scan/probe detection can be combined with content-based anomaly detection to improve the speed and the accuracy of detection. Moreover, a good detection system should be distributed as proposed in [14]. Interestingly, the effectiveness of this worm monitoring system [14] strongly depends on obtaining the information of the underlying vulnerable-host group distribution in /8 subnets and /16 subnets. Thus, the weapon race between the attackers and the defenders relies on how each side can collect and process the information of vulnerable-host distribution. The cooperation between the defenders from different domains provides information sharing, and therefore a possibly more effective detection system [8].

- For the self-learning worm system proposed in this paper, a key issue in defense is to detect and disable the worm server before the importance-scanning stage. One possible method to detect the worm server, for example, is to use host contact graph presented in [23]. After detecting the worm server, different mechanisms can be applied to disable the worm server, for example, putting the IP address of worm server in the *address blacklisting* [11]; providing false information of worm clients to the worm server; or even performing Denial of Service (DoS) attack on the worm server.

6. CONCLUSIONS

In this paper, we characterize a new scan-based worm through both analysis and simulation. This "self-learning worm" has the intelligence to gather and process information while propagating and thus increases the propagation speed. This self-learning ability of worms can also be applied to gain knowledge about other distribution information, such as DNS Top-Level Domains, countries, and Autonomous Systems. The self-learning worms presented in this paper use a simple proportion estimator. As our future work, other learning algorithms are worth investigating. Moreover, it is shown that the worm server can accurately estimate the group distribution with a large sample size. We plan to study the effect of estimation error on the worm propagation with a small or medium sample size. Since collected information is important for both attackers and defenders, we plan to study how the defenders from different domains can share information and build up practical and effective defense systems against future intelligent worms.

7. ACKNOWLEDGEMENTS

The authors would like to thank the anonymous reviewers for their valuable comments. Support from NSF ECS 0300605 is gratefully acknowledged.

8. REFERENCES

[1] P. J. Bickel and K. A. Doksum, "Mathematical Statistics: Basic Ideas and Selected Topics, Vol I (2nd Edition)," *Prentice Hall*, 2001.

[2] Z. Chen and C. Ji, "Importance-Scanning Worm Using Vulnerable-Host Distribution," *to appear in IEEE GLOBECOM 2005*.

[3] Z. Chen, L. Gao, and K. Kwiat, "Modeling the Spread of Active Worms," in *Proc. of INFOCOM 2003*, San Francisco, April, 2003.

[4] E. Cooke, F. Jahanian, and D. McPherson, "The Zombie Roundup: Understanding, Detecting, and Disrupting Botnets," *Proc. of Steps to Reducing Unwanted Traffic on the Internet Workshop (SRUTI '05)*, Boston, 2005.

[5] A. Doucet, N. de Freitas, and N. Gordon, "Sequential Monte Carlo Methods in Practice," *New York: Springer*.

[6] H. Feng, A. Kamra, V. Misra, and A. D. Keromytis, "The Effect of DNS Delays on Worm Propagation in an IPv6 Internet," in *Proc. of INFOCOM 2005*, March 2005.

[7] P. Heidelberger, "Fast Simulation of Rare Events in Queueing and Reliability Models," *ACM Transactions on Modeling and Computer Simulation*, vol.5, no.1 pp. 43-85, Jan. 1995.

[8] M. Locasto, J. Parekh, S. Stolfo, A. Keromytis, T. Malkin, and V. Misra, "Collaborative Distributed Intrusion Detection," *CU Tech Report CUCS-012-04*, 2004.

[9] D. Moore, V. Paxson, S. Savage, C. Shannon, S. Staniford, and N. Weaver, "Inside the Slammer Worm,", *IEEE Security and Privacy*, 1(4):33-39, July 2003.

[10] D. Moore, C. Shannon, and J. Brown, "Code-Red: a case study on the spread and victims of an Internet worm," *ACM SIGCOMM/USENIX Internet Measurement Workshop*, Marseille, France, Nov 2002.

[11] D. Moore, C. Shannon, G. Voelker, and S. Savage, "Internet Quarantine: Requirements for Containing Self-Propagating Code,", in *Proc. of INFOCOM 2003*, San Francisco, April, 2003.

[12] G. Owen, "Game Theory," *Academic Press*, New York.

[13] R. Puri, "Bots & Botnet: An Overview," *SANS Institute 2003*.

[14] M. Abu Rajab, F. Monrose, and A. Terzis, "On the Effectiveness of Distributed Worm Monitoring," *to appear in Usenix Security 2005*.

[15] S. Ross, "Introduction to Probability Models," 7th edition, *New York: Academic Press*, 2000.

[16] S. Saroiu, P. K. Gummadi, and S. D. Gribble, "A Measurement Study of Peer-to-Peer File Sharing Systems," *Proc. of Multimedia Computing and Networking 2002 (MMCN'02)*, January 2002.

[17] C. Shannon and D. Moore, "The Spread of the Witty Worm," *IEEE Security and Privacy*, vol. 2 No 4, Jul-Aug 2004, pp. 46–50, Aug 2004.

[18] P.J. Smith, M. Shafi, and H. Gao, "Quick simulation: a review of importance sampling techniques in communications systems," *IEEE Jour. Selected Areas Commun.*, vol.15, pp.597-613, May 1997.

[19] S. Staniford, D. Moore, V. Paxson, and N. Weaver, "The Top Speed of Flash Worms," in *Proc. ACM CCS WORM*, October 2004.

[20] S. Staniford, V. Paxson, and N. Weaver, "How to 0wn the Internet in Your Spare Time," in *Proc. of the 11th USENIX Security Symposium (Security '02)*, 2002.

[21] N. Weaver, "Warhol Worms, the Potential for Very Fast Internet Plagues," http://www.cs.berkeley.edu/~nweaver/warhol.html.

[22] J. Wu, S. Vangala, L. Gao, and K. Kwiat, "An Effective Architecture and Algorithm for Detecting Worms with Various Scan Techniques," in *Network and Distributed System Security Symposium*, 2004.

[23] Y. Xie, V. Sekar, D. A. Maltz, M. K. Reiter, and H. Zhang, "Worm Origin Identification Using Random Moonwalks," in *Proc. of the IEEE Symposium on Security and Privacy (Oakland 2005)*,Oakland, CA, May 2005.

[24] S. Xing and B.-P. Paris, "Measuring the size of the Internet via importance sampling," IEEE journal on selected areas in communications, 21(6), pages 922-933, August 2003.

[25] C. C. Zou, L. Gao, W. Gong, and D. Towsley, "Monitoring and Early Warning for Internet Worms," *10th ACM Conference on Computer and Communication Security (CCS'03)*, Oct. 27-31, Washington DC, USA, 2003.

[26] C. C. Zou, D. Towsley, W. Gong, and S. Cai, "Routing Worm: A Fast, Selective Attack Worm based on IP Address Information," *19th ACM/IEEE/SCS Workshop on Principles of Advanced and Distributed Simulation* (PADS'05), June 1-3, Monterey, USA, 2005.