Misleading and Defeating Importance-Scanning Malware Propagation

Guofei Gu, Zesheng Chen
Georgia Institute of Technology
Atlanta, Georgia 30332
{guofei@cc, zchen@ece}.gatech.edu

Phillip Porras
SRI International
Menlo Park, California 94025
porras@csl.sri.com

Wenke Lee
Georgia Institute of Technology
Atlanta, Georgia 30332
wenke@cc.gatech.edu

***Abstract*—The *scan-then-exploit* propagation strategy is among the most widely used methods by which malware spreads across computer networks. Recently, a new self-learning strategy for selecting target addresses during malware propagation was introduced in [1], which we refer to as *importance scanning*. Under the importance-scanning approach, malware employs an address sampling scheme to search for the underlying group distribution of (vulnerable) hosts in the address space through which it propagates. The malware utilizes this information to increase the rate at which it locates viable addresses during its search for infection targets. In this paper, we introduce a strategy to combat importance scanning propagation. We propose the use of *white hole* networks, which combine several existing components to dissuade, slow, and ultimately halt the propagation of importance scanning malware. Based on analytical reasoning and simulations using real trace and address distribution information, we demonstrate how the white hole approach can provide an effective defense, even when the deployment of this countermeasure represents a very small fraction of the address space population.**

I. Introduction

Malicious self-propagating code, or malware, is recognized as a significant threat to Internet security. Most typically, malware attempts to infect vulnerable machines on the Internet in an *automatic* fashion [2], [3]. Existing malware families are highly versatile in their ability to attack susceptible hosts through different propagation methods, such as via email attachments, infected P2P media, drive-by download infections, and scan-then-exploit infections. Among these methods, the scan-then-exploit strategy remains one of the most commonly used, and arguably is among the more efficient methods to rapidly spread malware across the Internet. In particular, scan-then-exploit malware operates without any need for *human interaction:* there is no need to wait for an email to open, a website visit, or a P2P application data exchange. In essence, scan-then-exploit malware is able to propagate as fast as it can find new victims. Thus, there is significant incentive for malware developers to continue refining the efficiency with which malware scans yield new candidate addresses.

As a result, malware is becoming smarter in selecting scan targets. In recent years we have seen an evolution in the approaches of scanning strategies from naive random-scanning techniques, to much faster and more evasive propagation methods. While many malware families (especially worms) have utilized random scanning techniques with notable success [4], [5], in general the random scan method is relatively inefficient in searching for victim hosts within the Internet, and its indiscriminate nature makes it highly subject to passive detection [6]–[8]. A key observation is that a random search algorithm is ill-suited for seeking targets when those targets reside in a space at (predictably) non-uniformly distributed locations.

With respect to Internet address occupation, existing research shows that the real (and also vulnerable) machine distribution in the whole IP space is not uniform [9]–[11], and this point has also been noted in some worm studies [5]. Recently, researchers proposed several new propagation strategies for malware families that apply knowledge of the Internet structure. These strategies fall into two categories: list-based scanning, and probability-based scanning.

Malware that employs list-based scanning uses a pre-generated target (IP or subnet) list as the propagation space, so that the infection does not waste time probing dark space. For example, some propagation strategies utilize BGP routing space information [12], some utilize address sampling to uncover live subnets in the address space prior to spreading [13]. Staniford *et al.* [14] studied the top speed of flash worms assuming the whole vulnerable list is available. Since it is generally hard to carry a large list of all IP addresses, list-based worms can aggregate addresses into subnets, e.g., /8 or /16 subnets.

Probability-based scanning worms further utilize the distribution information instead of treating these subnets equally. They scan the subnets with certain probability according to the underlying (vulnerable) machine distribution. A typical example of this strategy is proposed in [1] about self-learning worms using importance-scanning. Although currently observed malware families have not *fully* employed these two advanced strategies, there are already many preliminary attempts to (*partially*) utilize these techniques (e.g., local preference scanning, used by CodeRed [15] among many other malware variants). Recent research [13] also indicates that many botnets (e.g., AgoBot [16]) employ blacklists of well-known monitored IP space or agencies and avoid scanning. Here, we anticipate the future threat landscape will include malware that will ultimately adopt the *full* power of advanced scanning techniques. Thus, it is important for researchers to consider these threats and develop countermeasures now.

In particular, importance scanning techniques attempt to uncover the distribution of live IP addresses (or even real vulnerable machines), and then focus their infection efforts on these targets both to achieve a higher scan-to-infection ratio and to help evade passive monitors by avoiding the indiscriminate scanning of unused IP addresses. In [1], a self-learning worm is proposed, which estimates the vulnerability distribution very early in the infection stage (instead of before spreading). After an initial infection cycle, the attacker [1] will estimate the distribution according to existing victim information. Then all the worms will use this vulnerable-host distribution estimation to adjust their scan probability distribution. Alternatively, importance scanning malware can use the *live-host distribution* information instead of *vulnerable-host distribution* information, which may be harder to obtain. For this purpose, malware can use a two-step infection cycle similar to [13]: it may sample various network segments, followed by a probability-based spreading phase to those segments that appear to contain *live* subnets. In the first phase, the malware sample-scans addresses from an address segment, and upon completion will spread with a probabilistic affinity to those segments that appear to contain targets of interest (e.g., a population of live subnets).

Importance scanning poses some disturbing challenges for malware defense research. One implication of these network-structure-aware infection strategies is that they are *by design* intended to avoid low-occupancy address segments, including darknets that are instrumented with passive worm/malware detection tools, such as Kalman-filter-based detection [7], victim number-based detection [8]), Internet Motion Sensor [6], or network telescopes [17]. Second, importance-scanning malware is shown to provide a faster infection rate than other contemporary naive propagation strategies, suggesting a future of more virulent malware epidemics that combine speed and stealthy behavior. Chen et al. [18] demonstrates that counteracting network-aware malware such as importance-scanning worms is a significant challenge for the strategies that include host-based defense and IPv6.

In this paper we observe that the predictable affinity of importance scanning malware toward densely populated networks can also be viewed as a potential vulnerability. We explore the design space of what we refer to as *white holes*, which are systems that co-occupy populated network segments to increase the difficulty with which legitimate hosts can be targeted. A white hole can turn a legitimate live network segment into a segment that looks anomalously dense to a malicious application attempting to avoid honeynets, and can proactively mask the location of legitimate co-located addresses.

We introduce a defensive white hole approach that can be constructed using components from existing techniques, and analyze their ability to hinder importance-scanning malware propagation. Further, we examine how the incorporation of LaBrea-like mechanisms [19] can make a white hole an effective offensive tool to trap importance scanning malware, and conclude that the very affinity criteria that allows such malware to accelerate their infection rates also increases their susceptibility to our white hole countermeasure. Based upon analytical reasoning and simulations using real trace and address distribution information, we demonstrate how LaBrea mechanisms are far more effective in countering importance-scanning malware propagation, even when these countermeasures are deployed to a small ratio of the address space. We also discuss some challenging design issues and limitations in Section V.

[1]This worm can actually be considered as a simple botnet. The attacker is the botmaster in the botnet.

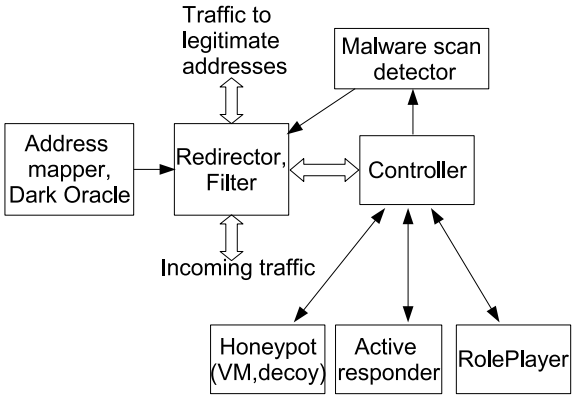


Fig. 1. Architecture of a White Hole

II. White Hole Design

As intelligent as firewalls, content filters, and address translation systems have become, it generally remains a difficult challenge to keep the existence of a live subnet or network segment invisible to attackers. In recognition of this reality, the alternative approach explored here is to hide the hosts of a live network segment within a population of seemingly live phantom addresses. The objective of a white hole service is to blend live targets in among phantom addresses the way a tree may be blended into a forest, or a needle into a needle stack. White holes present interactive responses to malware probes such that from the malware's point of view, the density of responses in the network segment obfuscate the malware's ability to successfully identify potential targets. A successful white hole deployment will effectively prevent the malware from accurately measuring the address distribution of the network segment.

We can assemble a white hole in the network by combining several existing techniques. Figure 1 shows the architecture of the white hole components, which include an address mapper, redirector, controller, malware scan detector, active responder, RolePlayer, VM honeypot, and a decoy honeypot. The following briefly summarizes the purpose and function of each component in the white hole architecture.

- **Address mapper**: Actively collects and updates the unused IP/port segment of the network that the white

hole will occupy. This can be done using an agent-manager based architecture (similar to SNMP network management). A similar technique is used in [20], in which an active mapping method builds profiles of the network topology and the TCP/IP policies of all hosts on the network. Recently, Cooke et al. proposed a system called Dark Oracle [21], which discovers dark addresses by actively participating in allocation, routing, and policy systems, and demonstrated successful operation in several networks. Unused ports can also be observed and emulated, or a strategic subset of service ports can be selected and emulated to enhance the realism of the white hole.

- **Redirector:** Redirects all incoming traffic to unused IP/ports, as specified by the address mapper, to the controller. This component is the first line classifier at the edge router of the protected network. All incoming packets targeting legitimate end hosts are passed through without disruption.
- **Controller:** Decides how redirected traffic will be handled within the white hole space. Streams may be directed to any of the available active responder components (including honeypots) within the white hole or filtered in cases of overload.
- **Responders:** Include a simple active responder, stateless role player, virtual machine honeypot, and physical decoy honeypot to interact with incoming malware scans. Potential **active responders** include applications such as iSink [16], or Honeyd [22] for handling simple scan responses. These scans may contribute most of the traffic. For simple scanning as shown in [13], a simple connection responder may be enough. If the connection needs further interaction, simple connection response may be insufficient to follow further application-level dialog. In such cases, traffic may be redirected to **VM-based honeypots**, or even **decoy honeypots** (real physical machines), to emulate full application response. In this way, we can capture the sampling attacks discussed in [1]. [23] discusses efficient approaches to deploy honeyfarms that can support large numbers of virtual machines. While this partially solves the scalability problem, we may also leverage the similarity of malware dialogs to more efficiently scale to larger malware scan volumes, as discussed by Cui et al. in the RolePlayer system [24], [25]. RolePlayer can achieve the goal of protocol independent adaptive replay of application dialog in a stateless (memory efficient) way. That is, one can use such a lightweight technique to learn existing (captured) malware dialog and then mimic the dialog to provide quick response for further similar connections.
- **Malware scan detector:** Includes among other techniques a detection algorithm such as threshold random walk (TRW) [26], [27]. We also envision white hole collaboration, allowing detectors from different white hole spaces to corroborate scanning patterns, similar to the Worminator [28] and DOMINO [29] architectures. Every detector will record scanner addresses in a Bloom filter. By exchanging these Bloom filters, we can achieve a privacy-preserving way for distributed attacker scan detection. Also note when using TRW, we can assign different weights on scans inside only one white hole and scans within multiple different white holes because the later case is more likely a malicious scanner.

The white hole operates by preventing an importance-scanning malware application from analyzing the group distribution statistics of the legitimate network in which it is co-located. In the critical initial sampling stage of an importance-scanning malware, the malware initiator sends sampling scans to the Internet (to achieve live-host distribution), or waits for a certain number of initial infected hosts to report their vulnerable-host distribution information [1]. In the first case, response from white hole spaces will be considered as live hosts. In the case of [1], white holes will use RolePlayers to mimic infected hosts and report to the attacker, thus white hole addresses will also be considered as live vulnerable hosts. In both instances, the white holes significantly disrupts the ability of the malware to accurately assess the live address distribution in the white-hole-protected network.

We are also interested in using incoming white hole scans to detect the malware initiator [2] (malware scan detector), potentially to help filter scans to legitimate addresses within the protect network segment. For the propagation strategy of live-host distribution, one approach is to employ Bloom filters to capture common source scanning addresses to the white hole space. For the propagation strategy of vulnerable-host distribution in [1], in which the attacker waits for existing victims to report information, we can detect the attacker by observing numerous outgoing connections to a common target address in a destination-address Bloom filter. Once a malware initiator is detected, the redirector can use this information to drop scans to legitimate addresses within the protected network. We can also envision sharing bloom filters among white hole spaces, similar to that of Worminator [28].

Furthermore, we can use a LaBrea [19] like tarpit technique in white holes to TCP-based malware. These tarpit programs can answer scan attempts in such a way that the malware instance at the other end gets ”stuck” in this connection, sometimes for a very long time [19]. In Section III-B, we demonstrate that white holes will attract importance scanning malware to enter LaBrea-like network segments earlier in its infection phase, and throughout the epidemic with higher probability. We find that this tarpit defense strategy is extremely effective when combined with a white hole to attract the importance-scanning malware.

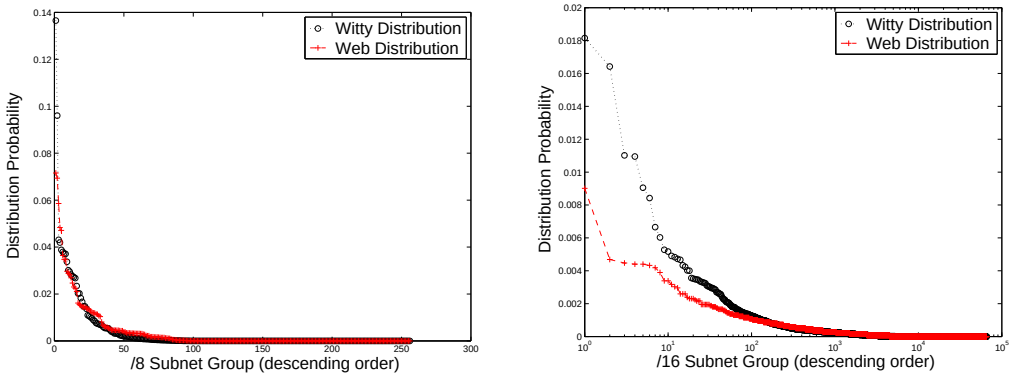
III. Mislead and Defeat Importance-Scanning Malware Propagation

Before our analysis, we list the notation used in the paper in Table I. Note there is a slight difference between

[2]We do not necessarily assume the detector works in the second (spreading) stage, although that will definitely improve our performance to defeat the malware. It is also worth noting that in later section we also analyze the situation when there is *no* malware scan detector available.

TABLE I
NOTATION USED IN THE PAPER

N	total number of vulnerable hosts on Internet
N_i	number of vulnerable hosts in group i
m	total number of groups on Internet
$I(t)$	number of infected hosts at time t
$I_i(t)$	number of infected hosts at time t in group i
Ω	total number of addresses in the scanning space
Ω_i	number of addresses in group i
s	scanning rate
α	infection rate
β	correct estimation probability of real vulnerable hosts
$p_g(i)$	percent of the live vulnerable hosts in group i
$p_g^*(i)$	probability of a scan hitting group i
U	total number of addresses used by all white holes
U_i	number of addresses covered by the white hole in group i
$K_i(t)$	average number of scans at time t in group i
$K(t)$	total number of scans at time t
$e_i(t)$	average number of newly infected hosts at time t in group i
$e(t)$	total number of newly infected hosts at time t



(a) In /8 subnet group (b) In /16 subnet group (X-axis in log scale)

Fig. 2. (Vulnerable) Host distribution in subnet group on Internet

malware propagation strategies utilizing live-host distribution and vulnerable-host distribution [1]. This will not make a fundamental difference in our general analysis framework and results. Specifically, the following analysis is based on [1] (malware that uses vulnerable-host distribution).

Using AAWP (Analytical Active Worm Propagation) model [15], we can model the propagation of a malware infection as

$$I(t+1) = I(t) + (N - I(t))[1 - (1 - \frac{1}{\Omega})^{sI(t)}]$$

, where $I(t)$ is the number of infected hosts at time t, N is the total number of vulnerable hosts on the Internet, Ω is the total number of addresses in the scanning space, s is the malware scanning rate.

When a malware infection begins to spread, $I(t) << N$ and $sI(t) << \Omega$. Thus the AAWP model can be approximated as

$$I(t+1) = I(t) + N\frac{sI(t)}{\Omega} = (1+\alpha)I(t),$$

where $\alpha = \frac{sN}{\Omega}$ is the infection rate [12], and which represents the average number of infected vulnerable hosts per unit time by a single malware victim during the early stage of malware propagation. By using address distribution information, the malware can increase its infection rate success. For example, [12] mentioned that a BGP and Class-A routing worm can speed up this infection rate by 3.5 and 2.2 times compared to a regular worm that scans the whole IPv4 space uniformly.

A. Infection Rate and Misleading Effect Analysis

The Internet is partitioned into m groups. As shown in [30], the infection rate of an importance-scanning malware is

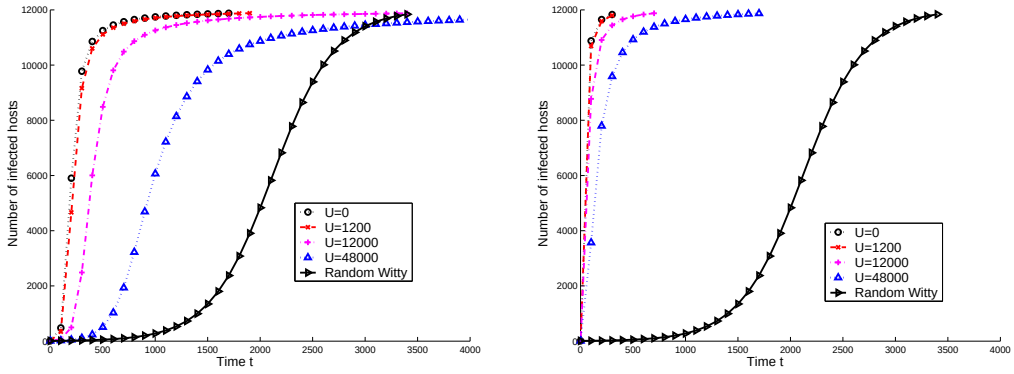
$$\alpha = sN\sum_{i=1}^{m}\frac{p_g(i)p_g^*(i)}{\Omega_i},$$

where $p_g^*(i)$ is the probability that a scan will hit group i, $p_g(i)$ is the percentage of live vulnerable hosts in group i.

If the malware can exploit the vulnerable host distribution information and scan Internet according to this probability distribution, i.e., $p_g^*(i) = p_g(i)$, we should have

$$\alpha = \frac{sN}{\Omega} \times \Omega\sum_{i=1}^{m}\frac{(p_g(i))^2}{\Omega_i},$$

where $\Omega = 2^{32}$, i.e., the full Internet address space. Therefore, importance-scanning malware can increase the infection



(a) Group size /8, misleading U, $\beta = 1$ (b) Group size /16, misleading U, $\beta = 1$

(c) Group size /8, misleading both N and U, $\beta = 0.1$ (d) Group size /16, misleading both N and U, $\beta = 0.1$

Fig. 3. Effect of white holes to slow down importance-scanning malware using witty-vulnerable-distribution. For simplicity, only *one* white holes is deployed.

rate with the factor of $\Omega \sum_{i=1}^{m} \frac{(p_g(i))^2}{\Omega_i}$, compared to random-scanning malware (where $\alpha = \frac{sN}{\Omega}$).

Let H_i denote the event that the ith group deploys a white hole that covers U_i white hole addresses.

$$H_i = \begin{cases} 1, & if\ the\ ith\ group\ deploys\ a\ white\ hole \\ 0, & otherwise \end{cases}$$

When a white hole is introduced, from the view of a malware, the number of vulnerable hosts increases from N to $N+U$ (remember all white hole addresses will appear live and vulnerable to the malware in its estimation at first stage, i.e., we mislead the malware by using extra U white hole space, we refer to this as *"mislead U"*). When we consider the case where detection and blocking (i.e., filtering detected malware scans to legitimate addresses in the sampling phase) is available (and many networks deploy address filter/blacklisting), we can provide much less real vulnerable information to the malware (we refer to this as *"mislead N"*, i.e., we mislead the malware by shrinking the true size of N). Thus, for the malware, the final vulnerable hosts are estimated as $N\beta + U$, where β is the correct estimation probability of real vulnerable hosts. With the help of detector and wide deployment of address blacklisting, we could keep β very small.

Thus, a malware estimates the vulnerable-host distribution as following

$$\hat{p}_g(i) = \frac{N_i\beta + U_i H_i}{N\beta + U} \quad (1)$$

When $p_g^*(i) = \hat{p}_g(i)$, and for simplicity, we assume that the white hole is deployed only in group k where $U_k >> N_k$,

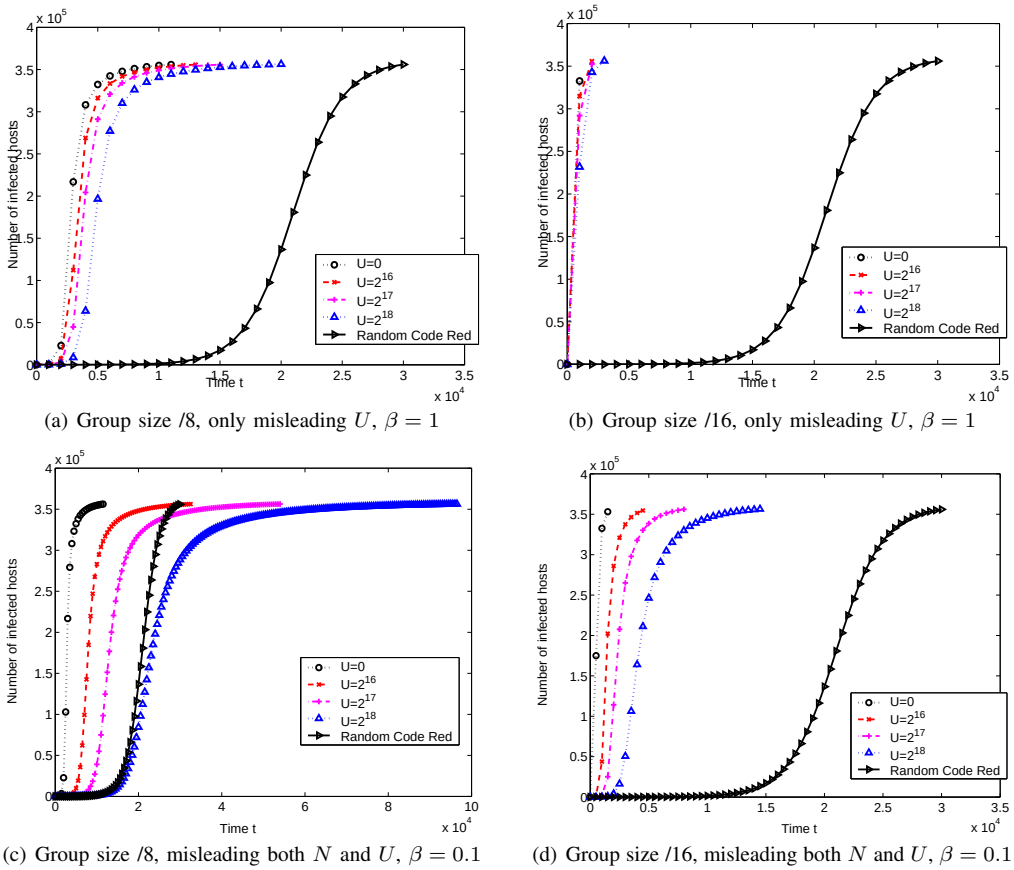
$$\begin{aligned} \alpha &= \frac{sN}{\Omega} \times r \times \Omega \sum_{i=1}^{m} \frac{(p_g(i))^2}{\Omega_i} + (1-r)sN\frac{p_g(k)}{\Omega_k} \\ &\approx \frac{sN}{\Omega} \times r \times \Omega \sum_{i=1}^{m} \frac{(p_g(i))^2}{\Omega_i}, \end{aligned}$$

where $r = \frac{N\beta}{N\beta+U}$ and we ignore the last item ($p_g(k)$ is very small as assumed). Therefore, the white hole decreases the infection rate with the factor of $\frac{N\beta+U}{N\beta}$. When $U >> N\beta$, the malware propagation is slowed down through the false information of the vulnerable-host distribution. In fact, we find that even using a relatively *small* white hole, we can still efficiently mislead and defeat and importance-scanning malware propagation.

For importance-scanning malware, infection propagation using distribution information can be modeled as following:

$$I_i(t+1) = I_i(t) + (N_i - I_i(t))[1 - (1 - \frac{1}{\Omega_i})^{sI_t p_g^*(i)}]$$

To show the analytical results in a realistic situation, we perform simulation using Matlab based on distribution information extracted from two real traces. The first data set is the witty worm from [5]. We extract the real witty victim distribution information and feed as the underlying real vulnerable distribution used by the importance scanning



(a) Group size /8, only misleading U, $\beta = 1$

(b) Group size /16, only misleading U, $\beta = 1$

(c) Group size /8, misleading both N and U, $\beta = 0.1$

(d) Group size /16, misleading both N and U, $\beta = 0.1$

Fig. 4. Effect of white holes to slow down CodeRed like importance-scanning malware using web-distribution.

malware (we refer to this distribution as *witty-vulnerable-distribution*, or simply witty-distribution). The second data set is the web server distribution from [1]. We use this information to simulate a CodeRed like importance scanning malware (we refer to this distribution as *web-distribution*). Figure 2 shows the (vulnerable) live host distribution when the addresses are aggregated into /8 and /16 groups. The shown distribution results are already sorted in descending order. It is very clear that the actual distribution of (vulnerable) live hosts is non-uniform and highly unbalanced. A small portion of groups have most of the (vulnerable) live hosts. This in another sense also verifies the intuition why using probability-based importance scanning may help malware spread faster.

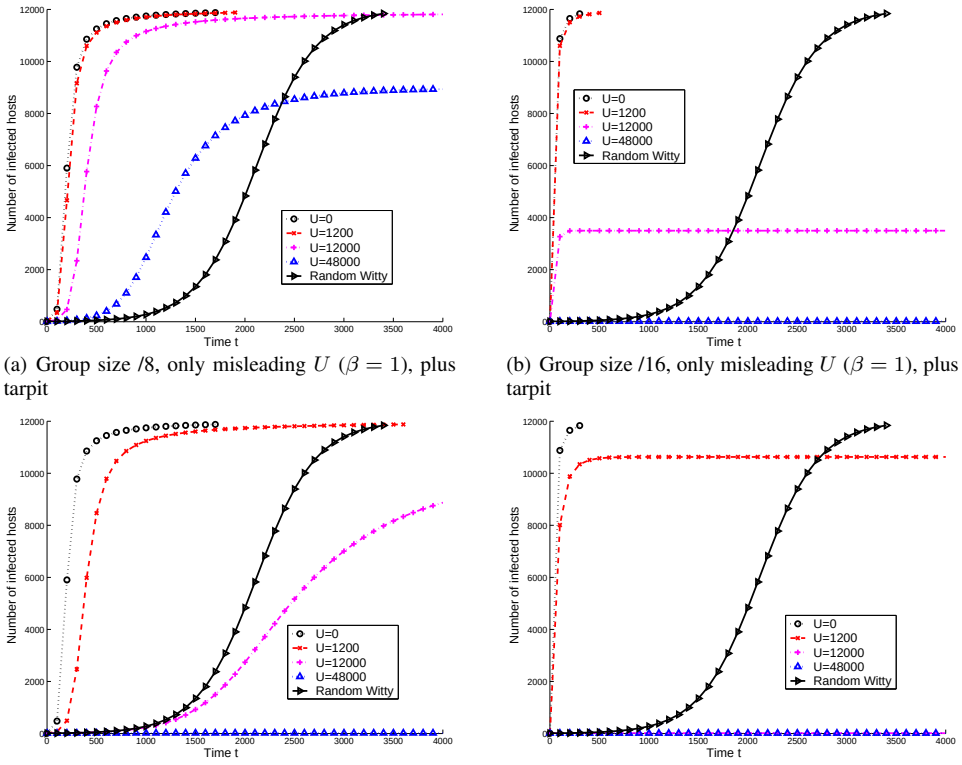
For the witty-vulnerability-distribution based worm example, we show the simulation results of misleading strategy using white holes in Figure 3. We use an initial hitlist of ten, total victim number 12,000, and scanning rate at 1,200 per unit time [5].

Figures 3(a)(b) show the results when we only mislead U (if we do not use malware scan detectors/filters). Figure 3(a) considers a group size of a /8 network, and (b) for a group size of a /16 network. We see that in both cases we slow down the importance scanning malware propagation using a variable size of white hole address space (from 1,200 to 48,000). The performance with group size /16 is worse than /8 when only using misleading U. This is because /16 distribution information is definitely more accurate than /8 distribution information. Thus, using a more detailed distribution information will make the malware spread faster. That is why although we use a white hole covering the same space in two cases, the a malware still propagates faster in the /16 scenario than in the /8 scenario.

Figure 3(c)(d) shows the cases when we mislead both U and N (i.e., we also use malware scan detectors/filters), with $\beta = 0.1$. This is much better than only misleading U. From (c) we see that white hole covering only 48,000 addresses can greatly impact the malware's infection growth rate.

For the web-distribution-based CodeRed like malware, we show the simulation results of a misleading strategy using white holes in Figure 4. We use an initial hitlist of ten, 360,000 total victims, and a scanning rate of $358/60 \approx 6$ per unit time (like CodeRed) [15]. We also use white holes covering an address space size comparable (e.g., $2^{16} - 2^{18}$) to the total number of victims.

In short, the effect is similar to the witty case. However, as we have a much greater total number of victims this time, we see that the slowing down result is a little less effective than in the witty case.



(a) Group size /8, only misleading U ($\beta = 1$), plus tarpit

(b) Group size /16, only misleading U ($\beta = 1$), plus tarpit

(c) Group size /8, misleading both N and U ($\beta = 0.1$), plus tarpit

(d) Group size /16, misleading both N and U ($\beta = 0.1$), plus tarpit

Fig. 5. Effect of white holes to defeat importance-scanning malware using witty-vulnerable-distribution. The white holes also tarpit incoming connection (LaBrea-like).

B. Using Tarpit in White Holes

We now consider the effects of incorporating a LaBrea-like [19] service into the white hole to defend against an importance-scanning malware, by sticking the connection for a long time (note this only works on TCP-based malware.) We modify the AAWP model:

$$K(t+1) = K(t)\left(1 - p_g^*(k)\frac{U}{\Omega_k}\right) + se(t)$$

$$K_i(t+1) = K(t+1)p_g^*(i)$$

$$e_i(i+1) = (N_i - I_i(t))\left[1 - (1 - \frac{1}{\Omega_i})^{K_i(t+1)}\right]$$

$$e(t+1) = \sum_{i=1}^{m} e_i(t+1)$$

$$I_i(t+1) = I_i(t) + e_i(t+1)$$

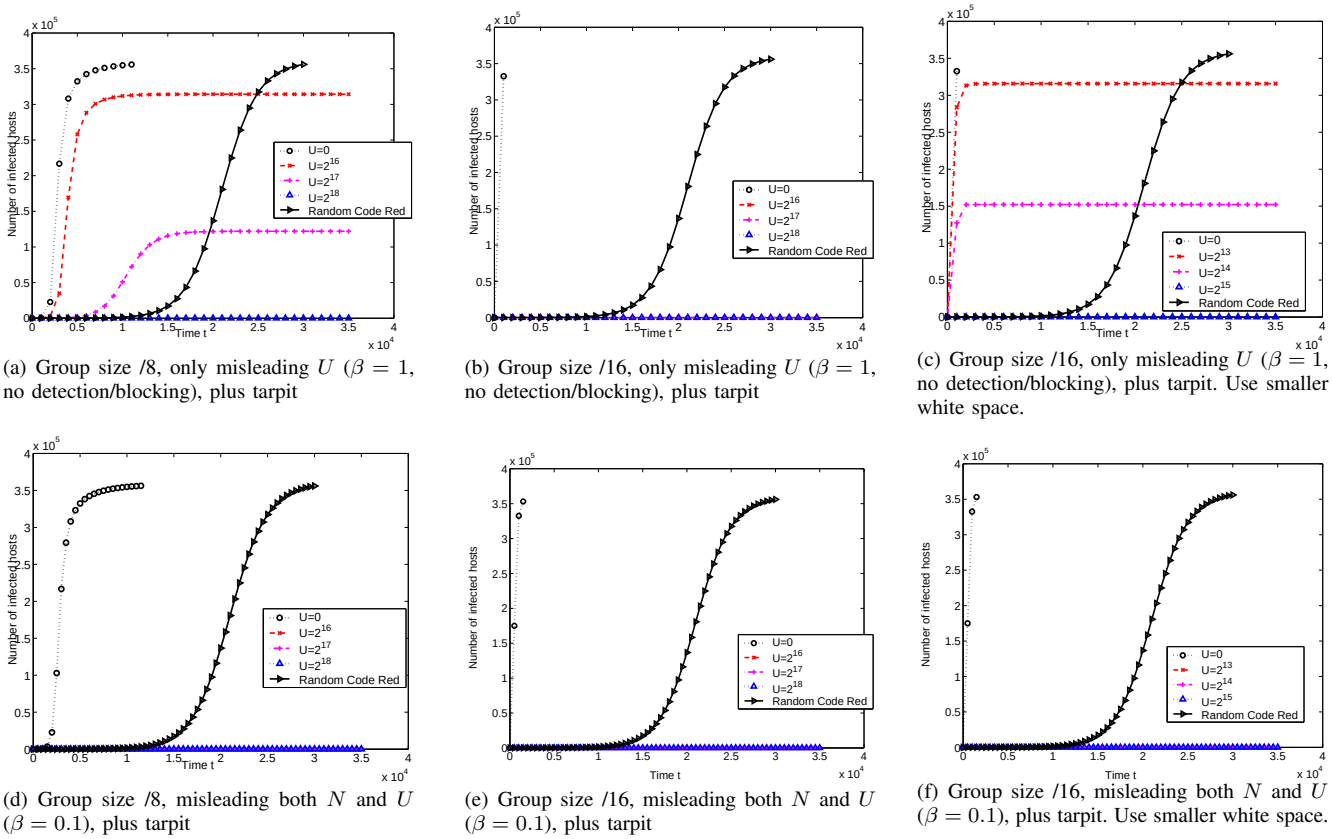
$$I(t+1) = \sum_{i=1}^{m} I_i(t+1).$$

where $K_i(t)$ and $e_i(t)$ denote the average number of scans and newly infected hosts at time t in group i (white hole is deployed in group k as before). The results with witty-vulnerable-distribution are shown in Figure 5.

From Figure 5(a)(b), we observe that the group size at /16 actually has a better performance (white holes can slow down malware propagation more) than the group size at /8 (opposite to the results from using just misleading U as shown in Figure 3). This is because the probability of the white hole being scanned is not changed when using /8 or /16, but the probability for the group with size /16 is significantly reduced from the group with /8. Thus, very likely, before the /16 group is hit, the malware is already stuck within the white holes.

Figure 6 shows the results on CodeRed like importance-scanning malware using web-distribution. We again confirm the effectiveness of white holes in slowing down or even stopping importance-scanning malware propagation. Similar to the witty case, the effectiveness of white holes in group size /16 is better than that in group size /8. Interestingly, we can use much smaller white holes (e.g., $2^{13} = 8,192$, which is much smaller compared to the total victim number $360,000$) to achieve satisfactory results (almost halting the malware propagation) as shown in Figure 6(c)(f). This means that although the higher number of total victims may help importance-scanning malware spread faster than in the case with less total victims, our white holes combining with tarpits also benefits from this situation, i.e., achieving better performance in its ability to defeat malware using relatively small white holes.

Our results suggest that combining a LaBrea-like technique is extremely effective in the context of importance-scanning malware in comparison to other malware propagation strategies. For non-importance scanning worms, Chen et al. [15]



(a) Group size /8, only misleading U ($\beta = 1$, no detection/blocking), plus tarpit

(b) Group size /16, only misleading U ($\beta = 1$, no detection/blocking), plus tarpit

(c) Group size /16, only misleading U ($\beta = 1$, no detection/blocking), plus tarpit. Use smaller white space.

(d) Group size /8, misleading both N and U ($\beta = 0.1$), plus tarpit

(e) Group size /16, misleading both N and U ($\beta = 0.1$), plus tarpit

(f) Group size /16, misleading both N and U ($\beta = 0.1$), plus tarpit. Use smaller white space.

Fig. 6. Effect of white holes to defeat CodeRed like importance-scanning malware using web-distribution. The white holes also tarpit incoming connection (LaBrea-like).

found that one needs at least 2^{18} LaBrea hosts to effectively defend against an active worm. Here we find that for the witty case with a total of 12,000 victims, a single white hole covering only 12,000 addresses (Figure 5(d)) is effective in halting the malware propagation. Even without misleading N (no detection/blocking components available), we can still use white holes covering 48,000 addresses to defeat the malware propagation efficiently (Figure 5(b)). For our CodeRed-like importance-scanning malware with a total 360,000 victims, white holes covering only $2^{13} = 8,192$ addresses (Figure 6(f)) are effective in halting the malware propagation. Even without detection/blocking components (no misleading N), we still can use white holes covering $2^{15} = 32,768$ to defeat the malware propagation efficiently (Figure 6(c)). In fact, we exploit the bias of importance-scanning results to guide the malware toward the white hole space, using the malware's own affinity to densely populated network segments against it to lure into the LaBrea countermeasure.

IV. Related Work

There is an abundance of work covering the use of unused space for malware/worm detection and defense. Most of this research involves passive monitoring techniques, such as Internet Motion Sensor [6], telescope [17], iSink [16]. Some of these systems also employ simple active response to TCP connections, but do not handle further request after TCP handshaking. Their primary purpose is to record and analyze incoming traffic.

There are also approaches that detect malware/worm outbreaks through the use of monitored traffic. Zou et al. propose a Kalman filter based detection [7] for efficient worm early warning. Wu et al. propose a victim number-based approach [7] to detecting the exponential increasing scans by worms. Gu et al. [31] propose a destination-source correlation (DSC) approach for efficient worm propagation detection in local networks.

Honeypot and honeynet techniques are used to lure attacks, and their functionality can range from simple connection acknowledgment and traffic collection, to full interaction with attackers. Several honeypot projects, such as honeynet [32], honeyd [22], honeyfarm [23], GQ [25], show great potential value for Internet malware study and defense. In some sense, a white hole could be considered as a special designed (non-traditional) active honeynet system with multiple honeypots together with other security components to fulfill a special purpose of misleading/defeating importance-scanning malware propagation, in addition to traditional general-purpose honeypot functionalities and services.

Openfire [33], which also exploits the idea of responding to all scan attempts to unused address space, is perhaps the most similar to our white holes, but with different focus and

purpose. Openfire focuses on using real decoy machines to reduce general attacks on (relative small) legitimate networks. The white hole technique is designed in the context of addressing importance-scanning malware, with the objective of misleading and ultimately defeating them when coupled with LaBrea-like countermeasures. White holes use several different response and detection techniques in operation.

V. Discussion and Limitation

There are several challenging issues with the white hole approach, many of them are our future work.

White hole dissuasion vs. attraction: Attackers may fingerprint the existence of white holes by observing that almost all IP/ports in the protected network segment are responsive to connect attempts, which can be a direct indicator of a potential probe monitoring system [34]. However, rather than this being a problem, we think it actually provides stronger motivation for adoption of the technique by a wider audience. The white hole effectively masks the legitimate network as a potential low interest address segment rather than a high interest target. We also observe a cumulative effect as more address spaces employ white holes in their networks, which further aids in disrupting malware propagation based on importance scanning. That is, U increases, and the probability of β decreases. Alternatively, white hole owners can also configure their response strategy to closely mimic real network distribution and operation with the intent of making the white hole space operate with characteristics similar to a legitimate network [35]. Thus, malware may hardly fingerprint the existence of white holes. Here the intent is to construct a network that will produce a higher than average attraction from importance-scanning malware, which can be used both the better study the malware attack strategies, and to deploy countermeasures such as a LaBrea system to impede the malware's progress. In addition, by opening originally unused IP/port space, it can efficiently distract and decrease attacks on the real IP/port space (where real hosts and services reside), a similar effect as showed in Openfire [33]. We decide to leave a deep study on dissuasion and attraction effects for future work and plan to employ game theory to study the best strategy of using/deploying white holes.

Distributed deployment strategy: When we deploy multiple white holes on the Internet, we could employ strategies to deploy the distribution according to our real vulnerable distribution. We plan to study the effect of distribution of white holes in the future (similar to the study in [11] for worm monitors).

Scalability: Existing techniques such as [23], [25] demonstrate positive progress on deploying large number of VM-based honeypot. The simple and stateless design of role player also shows positive potential. We keep most of the components simple which aids in the adoption cost, and believe scalability is not a significant issue. We hope to validate both assumptions in the future.

Attack tolerance: Malware may collect distribution information using approaches other than sampling, e.g., through address harvesting (SSH, Email, IM, etc.), other channel/out-of-band, to fingerprint live (even vulnerable) hosts. However, these approaches are much slower and harder than scanning/sampling, and they are not easy to achieve the whole picture of the Internet. Second, smart evasive malware, such as VM detection [36], honeypot-aware [37], or traffic learning, may identify whether they are within a white hole or not. Of course, future studies of the defense-attack interplay are needed in this arms race. However, in the *sampling* phase, the *primary* target of the importance scanning attack is to be stealthy to avoid detection. Honeypot-aware techniques [37] will involve more anomaly clues and yield higher risk of being detected.

LaBrea Resistance: Malware may eventually adapt to detect and escape tarpit mechanisms. That is, instead of achieving sustained sticking TCP malware, we should assume we can only tarpit for a certain time. We plan to do simulations in the future to find out the effect of different tarpit capabilities. We should keep in mind that besides LaBrea-like tarpit, we have several other alternate defense choices one could employ, such as address blacklisting, automatic signature generation (a taxonomy on defense techniques is in [38]). Finally, there is a debate on the legacy of using LaBrea-like tarpit technique [19], which is a non-technical issue out of our scope.

We should acknowledge that our proposed white hole strategy is a first step toward addressing emerging network-aware malware propagation strategies, and is a non-trivial component to design and deploy, depending on the depth of features one would want to incorporate. However, we also note that the key features envisioned in white holes represent an integration of existing techniques. Furthermore, the launching of a successful importance-scanning malware is also a non-trivial activity.

Finally, it is worth noting that although we show white holes are very effective in misleading and defeating importance-scanning malware propagation, they can also protect networks from regular scanning malware (similar to the effect in [33]), as well as serve traditional honeynet functionalities.

VI. Conclusion

In this paper we propose the design of white holes as a method to respond to a new generation of malware propagation strategies that seek to learn the address distribution statistics of the networks they are attacking. We propose the use of white holes to produce anomalous densities that are characteristic of naive honeynets that will be ignored by malware, in the spirit of hiding trees within a forest. We can also use the detection capabilities within the white hole to dynamically protect co-located legitimate addresses.

We also suggest that the density analysis of importance-scanning malware can be used against them, and propose the incorporation of LaBrea-like mechanisms into a white hole that tries to mimic dense legitimate networks. We observe that such an approach can rapidly trap the importance-scanning malware to a far greater degree than other propagation strategies. Our current assessment of this approach motivates us to

continue our study of more strategies to actively mislead and defeat future network-aware malware.

ACKNOWLEDGMENT

We are thankful to Linda Briesemeister, Chuanyi Ji and Vinod Yegneswaran for their helpful discussions and comments on an early version of this work. This material is based upon work supported through the U.S. Army Research Office (ARO) under the Cyber-TA Research Grant No.W911NF-06-1-0316 and Grant W911NF0610042, and by the National Science Foundation under Grants CCR-0133629 and CNS-0627477. Any opinions, findings, conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of U.S. ARO or the National Science Foundation.

REFERENCES

[1] Z. Chen and C. Ji, "A self-learning worm using importance scanning," in *ACM CCS Workshop on Rapid Malcode (WORM'05)*, 2005.

[2] S. Staniford, V. Paxson, and N. Weaver, "How to 0wn the internet in your spare time," in *11th USENIX Security Symposium (Security'02)*, 2002.

[3] N. Weaver, V. Paxson, S. Staniford, and R. Cunningham, "A taxonomy of computer worms," in *First ACM Workshop on Rapid Malcode (WORM'03)*, 2003.

[4] C. C. Zou, W. Gong, and D. Towsley, "Code red worm propagation modeling and analysis," in *9th ACM Conference on Computer and Communication Security (CCS'02)*, 2002.

[5] C. Shannon and D. Moore, "The spread of the witty worm," *IEEE Security and Privacy Magazine*, 2004.

[6] M. Bailey, E. Cooke, F. Jahanian, J. Nazario, and D. Watson, "The internet motion sensor: A distributed blackhole monitoring system," in *Network and Distributed System Security Symposium (NDSS'05)*, 2005.

[7] C. Zou, L. Gao, W. Gong, and D. Towsley, "Monitoring and early warning of internet worms," in *ACM Conference on Computer and Communications Security (CCS'03)*, 2003.

[8] J. Wu, S. Vanagala, L. Gao, and K. Kwiat, "An effective architecture and algorithm for detecting worms with various scan techniques," in *ISOC Network and Distributed System Security Symposium (NDSS'04)*, 2004.

[9] Y. Pryadkin, R. Lindell, J. Bannister, and R. Govindan, "An empirical evaluation of ip address space occupancy," USC/ISI, Tech. Rep. ISI-TR-598, 2004.

[10] "Distributed intrusion detection system (dshield)," http://www.dshield.org.

[11] M. A. Rajab, F. Monrose, and A. Terzis, "On the effectiveness of distributed worm monitoring," in *14th Usenix Security Symposium (Security'05)*, 2005.

[12] C. C. Zou, D. Towsley, W. Gong, and S. Cai, "Routing worm: A fast, selective attack worm based on ip address information," in *19th ACM/IEEE/SCS Workshop on Principles of Advanced and Distributed Simulation (PADS'05)*, 2005.

[13] M. Rajab, F. Monrose, and A. Terzis, "Fast and evasive attacks: Highlighting the challenges ahead," in *9th International Symposium on Recent Advances in Intrusion Detection (RAID'04)*, 2006.

[14] S. Staniford, D. Moore, V. Paxson, and N. W. Proc, "The top speed of flash worms," in *ACM CCS Workshop on Rapid Malcode (WORM'04)*, 2004.

[15] Z. Chen, L. Gao, and K. Kwiat, "Modeling the spread of active worms," in *IEEE INFOCOM'03*, 2003.

[16] V. Yegneswaran, P. Barford, and D. Plonka, "On the design and utility of internet sinks for network abuse monitoring," in *Symposium on Recent Advances in Intrusion Detection (RAID'04)*, 2004.

[17] D. Moore, "Network telescopes: Observing small or distant security events," in *11th USENIX Security Symposium (Security'02)*, 2002.

[18] Z. Chen and C. Ji, "Measuring network-aware worm spreading ability," in *IEEE INFOCOM'07*, 2007.

[19] "Labrea tarpit project," http://labrea.sourceforge.net/.

[20] U. Shankar and V. Paxson, "Active mapping: Resisting nids evasion without altering traffic," in *IEEE Symposium on Security and Privacy (Oakland'03)*, 2003.

[21] E. Cooke, M. Bailey, F. Jahanian, and R. Mortier, "The dark oracle: Perspective-aware unused and unreachable address discovery," in *3rd Symposium on Networked Systems Design and Implementation (NSDI'06)*, 2006.

[22] N. Provos, "A virtual honeypot framework," in *13th USENIX Security Symposium (Security'04)*, 2004.

[23] M. Vrable, J. Ma, J.Chen, D. Moore, E. Vandekieft, A. Snoeren, G. Voelker, and S. Savage, "Scalability, fidelity and containment in the potemkin virtual honeyfarm," in *ACM SOSP'05*, 2005.

[24] W. Cui, V. Paxson, N. Weaver, and R. H. Katz, "Protocol-independent adaptive replay of application dialog," in *13th Annual Network and Distributed System Security Symposium (NDSS'06)*, 2006.

[25] W. Cui, V. Paxson, and N. Weaver, "Gq: Realizing a system to catch worms in a quarter million places," ICSI, Tech. Rep. TR-06-004, 2006.

[26] J. Jung, V. Paxson, A. W. Berger, and H. Balakrishnan, "Fast portscan detection using sequential hypothesis testing," in *IEEE Symposium on Security and Privacy (Oakland'04)*, 2004.

[27] N. Weaver, S. Staniford, and V. Paxson, "Very fast containment of scanning worms," in *13th USENIX Security Symposium (Security'04)*, 2004.

[28] M. Locasto, J. Parekh, A. Keromytis, and S. Stolfo, "Towards collaborative security and p2p intrusion detection," in *2005 IEEE Workshop on Information Assurance and Security*, 2005.

[29] V. Yegneswaran, P. Barford, and S. Jha, "Global intrusion detection in the domino overlay system," in *Network and Distributed Security Symposium (NDSS'04)*, 2004.

[30] Z. Chen and C. Ji, "Optimal worm-scanning method using vulnerable-host distributions," *International Journal of Security and Networks (IJSN), special issue on "Computer and Network Security."*, 2006.

[31] G. Gu, M. Sharif, X. Qin, D. Dagon, W. Lee, and G. Riley, "Worm detection, early warning and response based on local victim information," in *20th Annual Computer Security Applications Conference (ACSAC'04)*, 2004.

[32] H. project, *Know your enemy: Learning about Security Threats*. Pearson Education, 2004.

[33] K. Borders, L. Falk, and A. Prakash, "Openfire: Opening networks to reduce network attacks on legitimate services," University of Michigan, Tech. Rep. CSE-TR-517-06, 2006.

[34] J. Bethencourt, J. Franklin, and M. Vernon, "Mapping internet sensors with probe response attacks," in *14th USENIX Security Symposium (Security'05)*, 2005.

[35] S. Sinha, M. Bailey, and F. Jahanian, "Shedding light on the configuration of dark addresses," in *ISOC Network and Distributed System Security Symposium (NDSS'07)*, 2007.

[36] T. Holz and F. Raynal, "Detecting honeypots and other suspicious environments," in *Sixth Annual IEEE Systems, Man and Cybernetics (SMC) Information Assurance Workshop (IAW'05)*, 2005.

[37] C. C. Zou and R. Cunningham, "Honeypot-aware advanced botnet construction and maintenance," in *International Conference on Dependable Systems and Networks (DSN'06)*, 2006.

[38] D. Brumley, L.-H. Liu, P. Poosankam, and D. Song, "Design space and analysis of worm defense strategies," in *2006 ACM Symposium on Information, Computer, and Communication Security (ASIACCS'06)*, 2006.