CyberTrap: Detecting and Quarantining Scanning Worms in Enterprise Networks

Xuxian Jiang
Department of Computer Sciences
Purdue University
West Lafayette, IN 47907
jiangx@cs.purdue.edu

Dongyan Xu
Department of Computer Sciences
Purdue University
West Lafayette, IN 47907
dxu@cs.purdue.edu

Abstract

The safety and reliability of current Internet and various enterprise networks have been constantly challenged by the increased frequency and virulence of worm outbreaks. Unfortunately, the situation is getting worse by the following observations: (1) security practices have discovered more worm-friendly *vulnerabilities than before; (2) research results show that better-engineered worms like* Warhol *worms and* Flash *worms could spread across the Internet within just* 15 *minutes or even* 30 *seconds.*

To address these challenges, this paper proposes CyberTrap, a systematic approach to mitigate worm propagation. Due to worms' scanning nature, every IP address in current IPv4 (2^{32}) space will be intended by live worms with certain probability. CyberTrap takes a defensive use of unused or darknet IP space: firstly, CyberTrap accurately identifies worm instances by trapping them with the darknet space; then CyberTrap actively takes counter-measures, like firewalling or blackholing, to quarantine those worms after collecting infection facts. This paper presents formal analysis of CyberTrap and examines its effectiveness and responsiveness in protecting enterprise networks. Both analytical and simulation results show that deployment of CyberTrap with a $/10$ internal *or* external *darknet space in one enterprise could effectively limit infectious worm percentage within that enterprise to less than* 3%.

1 Introduction

Disruptive worm spreading continues to pose a serious threat to the safety of current Internet and various enterprise networks since the infamous *Morris* worms[26] of early 1988. Unfortunately, recent worms have occurred more frequently than before: Code Red worms [4] in 2001, SQLSlammer worms [6] in 2002, MSBlaster worms [5] in 2003, and Witty Worms [9], and Sasser worms [8] in 2004. Partially due to increased complexities in system or application softwares for rich functionalities and extensible features, lots of software bugs are unintentionally introduced. CERT keeps track of reported vulnerabilities[7] and the result is shown in figure 1. It is upset to note that it apparently exhibits the undesirable rapid growth of software defects. Even worse, current worms could be further improved and better-engineered so that hyper-spreading worms like *Warhol* and *Flash* worms could spread across the Internet within just 15 minutes [28] or even 30 seconds [27]. The wide availability of vulnerabilities and potential faster worms demand instant response and effective containment of virulent worms.

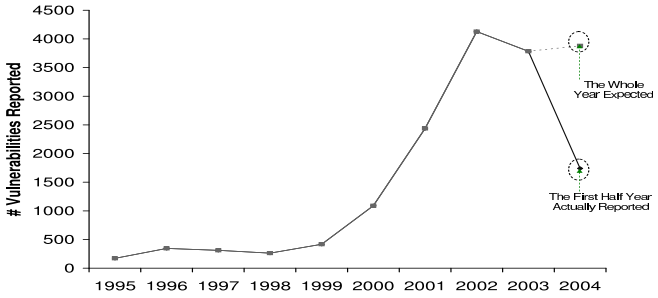


Figure 1. Number of Vulnerabilities Reported[7]

Addressing scanning worms which attempt to locate a vulnerable host before actually initiating the infection, various automated defenses have been proposed to detect the anomaly of infected hosts [35, 12, 10, 24, 14] and, accordingly, confine [39, 34, 23, 15] or slow-down [11, 31, 29] the worm propagation. This paper proposes a novel and complementing approach called CyberTrap. CyberTrap is a systematic worm-curtailing scheme and can be differentiated from other schemes in following ways:

- In order to accurately detect worm infections, CyberTrap takes advantage of possibly scattered darknet spaces and analyzes the traffic to/from those darknet.

A darknet is a portion of routeable IP space in which no active services or servers reside. Due to the nature of scanning worms, a darknet is extremely helpful to accurately locate infecting worms.

- Instead of passively monitoring any infection attempt to darknets, CyberTrap dynamically instantiates vulnerable services to capture live worms. The vulnerable services are sandboxed within a highly configurable virtual machine environment and actual infection activities are recorded. The purpose of collecting infection facts as evidence is two-fold: (1) it justifies counter-measures initiated later to quarantine worm nodes; (2) it collects live worm copies which is used to extract worm signatures for various IDS systems.

- With consideration of current peering architecture of Internet and the fact of each peering AS or enterprise is only authoritative within its own domain, CyberTrap is deployed by the authority of each enterprise domain and takes effective counter-measures to temporarily quarantine internal worms and block traffic from external worms. Different deployment of CyberTraps in different domains could cooperate with each other to maximize their effectiveness.

The rest of paper is organized as follows: Section 2 describes the overall architecture of CyberTrap and emphasizes the uniquenesses in defensive use of darknet space. The following section analyzes CyberTrap in different deployment scenarios and demonstrates its effectiveness and responsiveness by deriving numerical and simulation results. In Section 4, we further discuss operational requirements of CyberTrap and study potential attacks and further improvement. Finally, Section 5 examines related work and Section 6 concludes this paper.

2 CyberTrap Approach

Figure 2 shows the operational view of CyberTrap. Suppose some worm is currently propagating across the Internet and enterprise network A ($128.10.0.0/16$) and B ($129.10.0.0/16$) have deployed CyberTraps. For illustration purpose, Figure 2 only shows one worm interaction with CyberTrap in enterprise network A.

CyberTrap takes advantages of two types of darknet spaces: *internal* and *external*. Internal darknet addresses have been officially allocated to the deploying enterprise while *external* darknet addresses are Internet-wide and have not officially assigned to any entity. It should be noted that the Internet-wide darknet space may only be used unofficially and any enterprise should not publicly claim that address block. In Figure 2, the CyberTrap in enterprise A has an internal darknet space ($128.10.254.0/24$) and an external darknet space ($11.0.0.0/8$[1]). Similarly, the CyberTrap in enterprise B has another internal darknet space ($129.10.254.0/24$) and the same external darknet space ($11.0.0.0/8$). As shown in Figure 2, CyberTrap works as follows:

- *Observing Infection Attempt* Node $H1$ in enterprise A sends infection attempts to potential victims and every IP is likely to receive the attempt with certain probability. As shown in step 1, the worm $H1$ is attempting to infect a node with IP $11.11.11.1$. The border router $R1$ of enterprise A is able to observe the traffic since the destination IP does not belong to enterprise A.

- *Redirecting Infection Attempt* Based on configured external darknet space range, the border router $R1$ realizes that the traffic is suspicious since it is heading to an unused external IP. $R1$ redirects (step 2) the traffic to the CyberTrap center.

- *Triggering Worm Infection* CyberTrap further examines the traffic and realizes it is possibly a worm. A virtual machine with corresponding service(s) may be dynamically instantiated so that the infection is triggered (step 3) and all ensuing traffic is recorded.

- *Quarantining Infecting Worm* Once the worm is triggered to expose its behavior, Cybertrap could accurately identify the existence of worms. After node $H1$ is identified , the closest router ($R1$ in this case) to $H1$ would be instructed (step 4) to insert a filtering rule, i.e., FW1 in Cisco access list command [33], to drop any worm traffic generated by $H1$ and essentially quarantine (step 5) $H1$. However, if CyberTrap in enterprise A detects an infection from a node, say $H3$, of enterprise B, CyberTrap in A could either (1) notify CyberTrap in B with collected evidence so that CyberTrap in B could take corresponding counter-measures based on that evidence; or (2) blacklist the worm traffic from $H3$ in $R1$ so that it can protect internal vulnerable hosts in enterprise A from being infected by $H3$.

CyberTrap is unique in its *playground*, i.e., darknet, and is able to achieve nearly-zero false positive and low false negatives due to the exploitation of darknet space and the provocation of worm behaviors.

2.1 Trapping Worms Using Darknets

Worms replicate themselves without human interactions by remotely exploiting *known* vulnerabilities in operating

[1]The address block $11.0.0.0/8$ is chosen just for illustration purpose and we assume that this address block has not offi cially assigned to any entity in this paper.

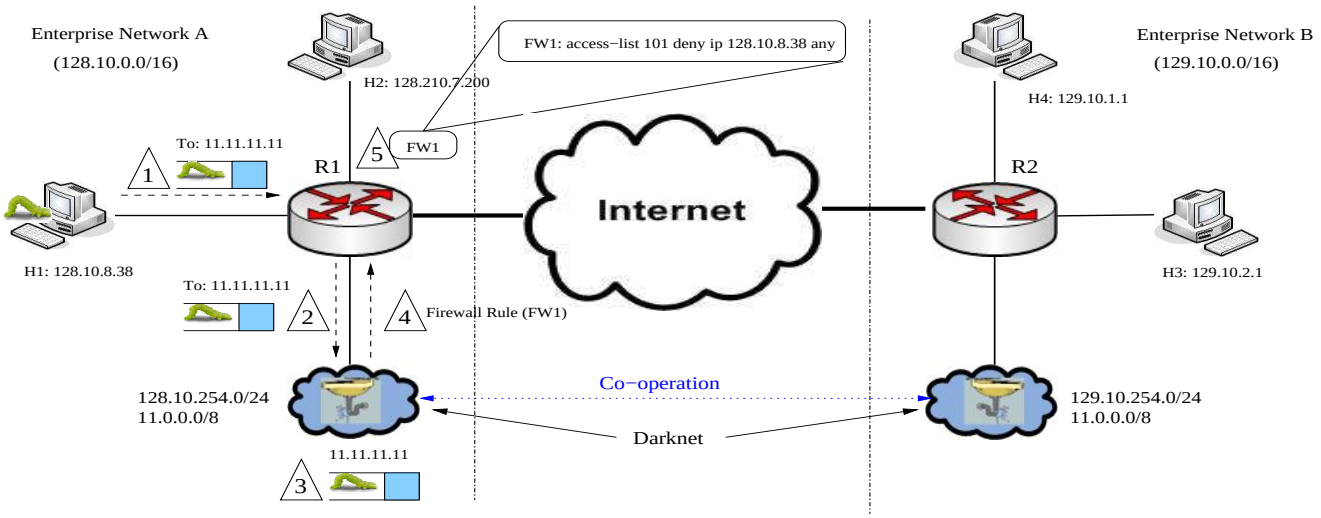


Figure 2. Operational View of CyberTrap

systems or application services. If we break down the actions of these worms [4] [5] [6], the following common behaviors or stages will be exposed: *Target Selection*, *Exploitation*, and *Replication* [19].

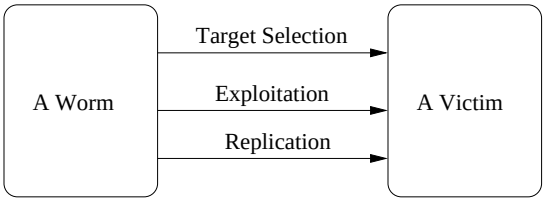


Figure 3. Staged View of Worm Infection

During the stage of target selection, a worm source picks up a target according to a certain selection logic. In the case of MSBlaster worms, with a probability of 60%, the target is chosen randomly. With a probability of 40%, hosts within the same class B network as itself will be selected. Code-Red II worm is another virulent worm which opens a number of threads[2] to probe hosts: with the probability of $1/2$, it will attempt hosts in the same class A network as the worm node; with the probability of $3/8$, it will probe targets in the same class B network as itself; with the probability of $1/8$, it will scan targets randomly[28]. A simple ICMP *echo request* or TCP *syn* packet has been observed to probe a node before actually initiating the exploitation.

A large darknet space like a class A network, i.e., a $/8$ network, could provide enough opportunity to observe worms, particularly randomly scanning worms. Assume a large number of vulnerable hosts like 10 million[3] for a randomly scanning worm and denote the probability of worms hitting the darknet space and contacting a victim as Δ and Θ, respectively. We have $\Delta/\Theta = 1.68$, which means that it is more likely for the darknet space to observe the worm first before other victims are contacted. However, due to the possibly local subnet preference for worm spreading, it is desirable to have a scattered collection of small darknets even though it may cause additional complexities in deploying and managing these darknets.

2.2 Triggering Worms Using VM-based Sandboxing

As pointed out before, worms exploits certain vulnerabilities to propagate themselves and those remotely-exploitable vulnerabilities are exhibited within particular services provided by victims. The requirement for successfully exploiting these vulnerabilities remotely and self-propagating nature of worms suggest the existence of certain characteristics in worm traffic like the same destination port number [6, 4, 5]. Though it is able to detect incoming probings by passively monitoring darknet spaces, interaction with worms is still necessary to trigger them to expose or release their payloads (Exploitation and Replication in Figure 3).

In order to expose clearly worms' behavior and justify later reaction measures like blacklisting and filtering, CyberTrap triggers live worms heading darknet spaces within a safe sandbox environment. In additions to exposing common information like destination port, the execution of worms could also unveil almost identical worm payloads[4].

[2]The number of threads is dependent on whether the Chinese language is installed in the system or not. If Chinese is the language installed, the Code-Red II worm will open 600 threads and start infections. Otherwise, 300 threads will be created.

[3]It is estimated that there is $360,000$ hosts infected by Code-Red worms.

[4]Polymorphic worms will have multiple forms deliberately hiding the

These striking similarities can be easily leveraged to identify worms' existence.

However, CyberTrap needs to instantiate appropriate vulnerable services quickly simply from collected probings and elicit live worms safely to prevent unintentional consequences or damages. Recent advances in virtual machine technique make it possible to quickly instantiate a whole system image within seconds and confine potential damages caused by live worms. However, the identification of suitable vulnerable service needs complete knowledge of existed vulnerabilities and careful classification of probings. We have a prototype of such system called BAIT-TRAP [17] which is able to compose and deploy a VM-based honeypot within seconds.

2.3 Quarantining Worms Through Blacklisting and Filtering

Address blacklisting and packet filtering are two major approaches to quarantine worm propagation. Address blacklisting excludes traffic from identified worm sources, while packet filtering could drop traffic according to specified rules. The rule can be a traffic flow specification or a typical payload content, which is identified as a particular worm signature. Strictly speaking, address blacklisting is a special form of packet filtering. The access control entry (FW1) in Figure 2 is an example of address blacklisting.

CyberTrap is designed to support both methods to mitigate spreading worms and its ultimate goal is to realize complete automation for worm quarantine:

- Firstly, traffic communicating with administrated darknet spaces are automatically classified according to intended services;
- Secondly, those worm traffic related to one service type is grouped and leveraged to automatically extract worm signatures[5];
- Thirdly, those worm signatures are automatically uploaded to reconfigure firewalling or routing devices to drop relevant worm traffic.

Recent research efforts like Autograph [21] and EarlyBird [25] are exploring automatic ways to extract worm signatures. This paper examines the approach of address blacklisting. However, it can be easily extended to accommodate signature-based content filtering.

In the following sections, we study the formal analysis of CyberTrap and examine its effectiveness and responsiveness.

worm payloads, but they still exhibit common information like same destination port number.

[5]Autograph[21] suggested the most recurring content block could be assumed as the worm signature.

3 Modeling CyberTrap

In this section, we first introduce the notations used in our analysis. Then we derive CyberTrap models based on various deployment scenarios and present their analytical and numerical solutions.

3.1 Notations

Consider a simple Internet architecture which is composed by m peering enterprise networks, $E_i, i \in \{1..m\}$. Denote the number of infectious nodes, the number of vulnerable nodes, and the number of quarantined nodes at time t within each enterprise network E_i are $I_{E_i}(t)$, $V_{E_i}(t)$, and $R_{E_i}(t)$ respectively. For convenience, we represents the total number of nodes involved in a worm outbreak as N ($N = \sum_{i=1}^{m}(I_{E_i}(t) + V_{E_i}(t) + R_{E_i}(t)) = I(t) + V(t) + V(t)$). The notations used throughout this paper are collected in table 1.

Suppose the infection rate of a certain worm is a constant α and consider the overall worm propagation, the classic epidemic worm propagation model [16] with a finite population is defined by :

$$dI(t) = \alpha \times \frac{V(t)}{N} \times I(t) \times dt. \quad (1)$$

$\alpha \times \frac{V(t)}{N} \times dt$ represents the number of new worm nodes contributed by a single worm source within dt period and $di(t)$ is the number of new worm nodes during the time period $[t, t + dt]$ with current worm population $I(t)$.

Eq (1) is also known as the *logistic equation* [32] and has the following solution:

$$I(t) = N - \frac{N}{1 + e^{\alpha(t-T)}} \quad (2)$$

where T is some constant dependent on the initial worm population.

For simplicity, we firstly derive CyberTrap model based on its deployment on one enterprise network, then extend it to multiple either uncooperative or cooperative enterprise networks.

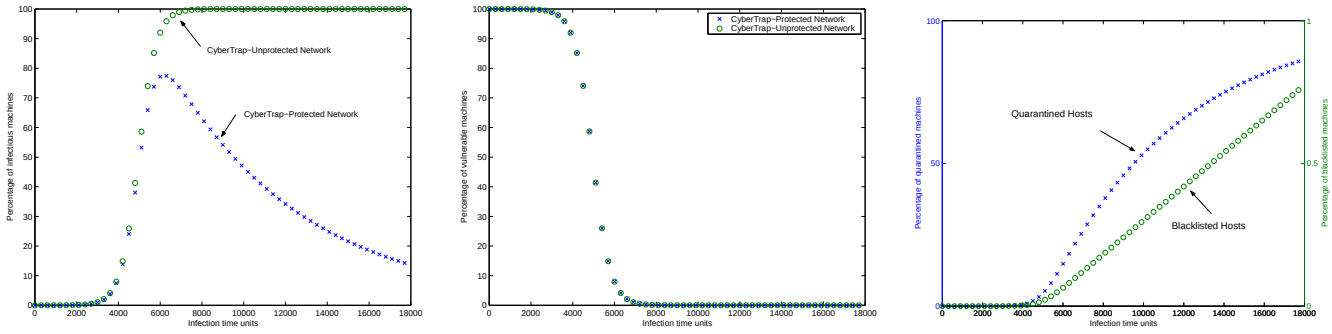
3.2 Single Deployment

Due to administrative restrictions and autonomous management requirement, the CyberTrap, once deployed within an administrative domain, needs to differentiate the source of incoming infection: if an infection is detected from its own network E_j, the worm could be quarantined directly by CyberTrap. However, if it is from other domains, that worm source can only be blacklisted by E_j network.

With the average scanning rate s, the number of scan attempts during the time period $[t, t + dt]$ from any worm

Table 1. Notation used in the paper

Symbol	*Description*
E_i	Enterprise Network E_i ($i \in \{1..m\}$)
$V_{E_i}(t)$	the number of vulnerable machines at time i within the Enterprise Network E_i during the spread of worm
$V(t)$	total number of vulnerable machines at time i during the spread of worm $V(t) = \sum_{i=1}^{m} V_{E_i}(t)$
$I_{E_i}(t)$	the number of infectious machines at time i within the Enterprise Network E_i during the spread of worm
$I(t)$	the number of infectious machines at time i during the spread of worm $I(t) = \sum_{i=1}^{m} I_{E_i}(t)$
$R_{E_i}(t)$	the number of machines which were infected but later quarantined within the Enterprise Network E_i before time t
$R(t)$	the number of machines which were infected but later quarantined before time t $R(t) = \sum_{i=1}^{m} R_{E_i}(t)$
$B_{E_j}(t)$	the size of blacklist accumulated by a CyberTrap which is deployed in network Enterprise Network E_j at time t
N	the total number of machines involved in a specific worm outbreak: N=V(t)+I(t)+R(t)
$\alpha/\alpha(t)$	the infection rate of a (self-replicating) worm at time t
D_{E_i}	the size of total darknet spaces in Enterprise Network E_i
D_{shared}	the size of Internet darknet spaces anycasted to any Enterprise Network E_i
$\beta/\beta(t)$	the hitting rate of a worm node on a CyberTrap at time t
s	the average number of machines scanned by an infected machine per unit time



(a) Percentage of Infectious Nodes

(b) Percentage of (Remaining) Vulnerable Nodes

(c) Percentage of Quarantined/Blacklisted Nodes

Figure 4. The Effectiveness of CyberTrap Against Random-Scanning Worms

source is $s \times dt$, and therefore there are $s \times dt \times I(t)$ scans in total for all $I(t)$ worm sources.

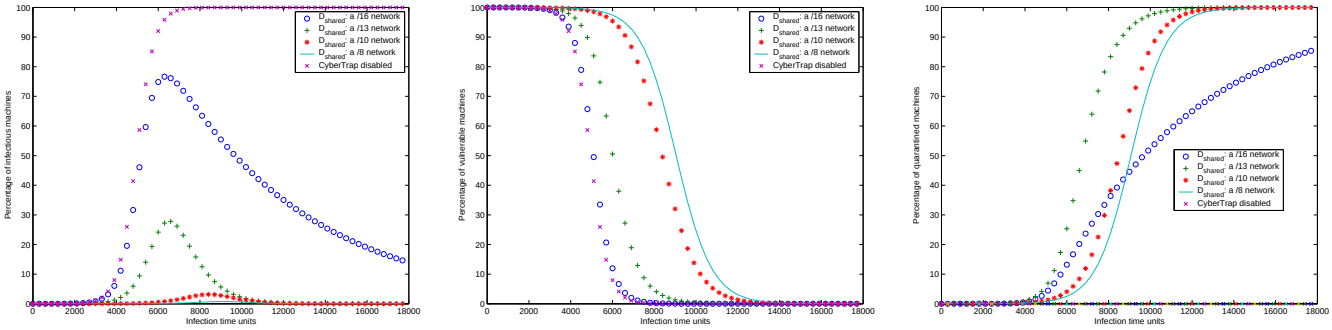
If we consider random-scanning worms which choose any Internet host with the same probability $1/2^{32}$, then the probability of a machine being scanned by any of current worm nodes is $\alpha(t) = 1 - (1 - \frac{1}{2^{32}})^{sI(t)dt} \approx C_0 I(t) dt$ [6], where the constant $C_0 = \frac{s}{2^{32}}$. With the total number of current scan attempts, the expected number of vulnerable machines in enterprise network E_i that will be subverted as infectious nodes during $[t, t+dt]$ is $\alpha(t) \times V_{E_i}(t)$. In other words,

$$\frac{dV_{E_i}(t)}{dt} = -C_0 I(t) V_{E_i}(t) \quad \forall i \in \{1..m\} \qquad (3)$$

[6]The approximation is achieved by Taylor expansion based on the fact that $sI(t)dt$ is much smaller than 2^{32}.

The minus sign shows the decreasing number of vulnerable nodes and thus the increasing number of infectious nodes due to current infection attempts.

Suppose the only CyberTrap is deployed within the Enterprise Network E_j with its own darknet space D_{E_j} and a shared Internet darknet space D_{shared}. The probability of a live worm outside E_j hitting the CyberTrap and thus blacklisted from E_j is $\gamma = 1 - (1 - \frac{D_{E_j}}{2^{32}})^{sdt} \approx C_{E_j,1} dt$, where the constant $C_{E_j,1} = \frac{sD_{E_j}}{2^{32}} = C_0 D_{E_j}$. Based on the same reasoning, the probability of a live worm inside E_j hitting the CyberTrap during time period $[t, t+dt]$ is $\beta = 1 - (1 - \frac{D_{E_j}+D_{shared}}{2^{32}})^{sdt} \approx C_{E_j,2} dt$, where the constant $C_{E_j,2} = \frac{s(D_{E_j}+D_{shared})}{2^{32}} = C_0(D_{E_j} + D_{shared}) = C_{E_j,1} + C_{D_{shared}}$ if we define $C_{D_{shared}} = C_0 D_{shared}$.



(a) Percentage of Infectious Nodes

(b) Percentage of (Remaining) Vulnerable Nodes

(c) Percentage of Quarantined Nodes

Figure 5. The Impact of Varying Size of Darknet Spaces in CyberTrap

Therefore, we can calculate the blacklist overtime as

$$\begin{aligned} dB_{E_j}(t) &= \gamma(I(t) - I_{E_j}(t) - B_{E_j}(t)) \\ &= C_{E_j,1}(I(t) - I_{E_j}(t) - B_{E_j}(t))dt, \end{aligned} \quad (4)$$

and refine the Eq (3) because of the impact of blacklisting as follows:

$$\frac{dV_{E_i}(t)}{dt} = \begin{cases} -C_0 I(t) V_{E_i}(t) & \text{if } i \neq j \\ -C_0(I(t) - B_{E_j}(t))V_{E_i}(t) & \text{if } i = j \end{cases} \quad (5)$$

Accordingly, the number of worm nodes that are quarantined during the time period $[t, t+dt]$ is:

$$dR_{E_i}(t) = \begin{cases} 0 & \text{if } i \neq j \\ \beta \times I_{E_i}(t) = C_{E_j,2} I_{E_i}(t)dt & \text{if } i = j \end{cases} \quad (6)$$

Assuming a static number of involved hosts within each domain i, we have

$$\frac{dI_{E_i}(t)}{dt} + \frac{dV_{E_i}(t)}{dt} + \frac{dR_{E_i}(t)}{dt} = 0, \quad \forall i \in \{1..m\}. \quad (7)$$

Eq (4, 5, 6, and 7) represent the CyberTrap model when there is only one deployment of CyberTrap.

In order to show the effectiveness of CyberTrap, we provide an example input in table 2 and show corresponding numerical solutions in figure (7). The X-axis is in *infection time units*: each time unit is the duration of one successful worm infection session (usually several seconds to tens of seconds). Figure 4(a), 4(b), and 4(c) show the percentage of infectious nodes, vulnerable nodes, and quarantined/blacklisted nodes (enabled by CyberTrap), respectively. The percentage is calculated based on the total number of involved nodes within its own domain only. Figure 4(a) shows that the active spreading worms can be effectively mitigated by deployed CyberTrap. With only a class B network ($D_{shared} + D_{E_1}$) as the darknet space, CyberTrap could limit the maximum percentage of infecting (random-scanning) worms to 80%. With larger darknet space like a class A network, CyberTrap could perform better (shown in Figure 5(a)). Figure 4(b) shows two almost identical vulnerable nodes curves even though one network has deployed the CyberTrap. This is due to the *reactive* nature of blacklist. The result is consistent with [23] and suggests another effective mechanism, i.e., *content filtering*. Figure 4(c) further details the accumulation of either quarantined or blacklisted hosts. The rapid growth of quarantined curve illustrates the power of CyberTrap, while the slow growth of blacklisted hosts reveals the needs for larger darknet space for CyberTrap purpose.

Parameter	*Value*	*Description*
N	10^6	Total vulnerable nodes
s	10	Scanning rate of inspected worm
m	2	Two peering networks (E_1 and E_2)
E_j	E_1	CyberTrap-protected network E_1
D_{shared}	$2^{16}-1$	a $/16$ network as external darknet
$I_{E_1}(0)$	0	Initially worm nodes in E_1
$V_{E_1}(0)$	$N/256$	Initial vulnerable nodes in E_1
$R_{E_1}(0)$	0	Initial quarantined nodes in E_1
$B_{E_1}(0)$	0	Initial blacklist size of CyberTrap
D_{E_1}	2^8-1	a $/24$ network as internal darknet
$I_{E_2}(0)$	10	Initial number of worm nodes in E_2
$V_{E_2}(0)$	$\frac{255*N}{256}$	Initially vulnerable nodes in E_2

Table 2. Parameters used to demonstrate the effectiveness of CyberTrap

The size of darknet space, particularly D_{shared}, plays an important role for CyberTrap. In order to show the impact of D_{shared}, we increase $V_{E_1}(0)$ in table 2 as $N/2$ and further derive the numerical solution with varying D_{shared}

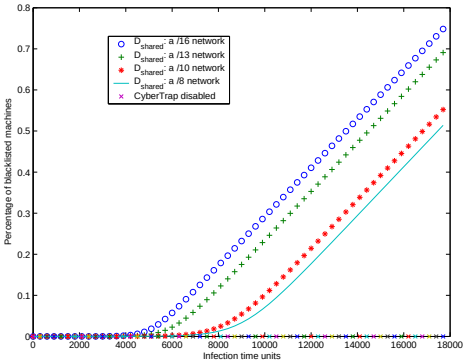


Figure 6. The Blacklist Size w/ Varying Darknet Spaces in CyberTrap

size from a $/16$ network to a $/8$ network. Figure 5(a), 5(b), and 5(c) show the percentage of infectious nodes, vulnerable nodes, and quarantined nodes, respectively. As shown in Figure 5(a), when D_{shared} reaches a $/10$ network, the maximum percentage of infecting nodes is decreased to as low as 3%. It is important to notice that even there is only one deployment, CyberTrap could also effectively slow-down Internet-wide worm propagation as shown in Figure 5(b) and 5(c). The *right-shifting* among curves with larger darknet space demonstrates the impact of quarantining internal worm nodes. The quarantine of internal worms reduces the likelihood of other nodes being infected and thus indirectly minimize the risks of other internal vulnerable nodes. The growing size of blacklist of CyberTrap (shown in Figure 6) also contributes to the Internet-wide slow-down of worm propagation.

3.3 Multiple Deployment

In this subsection, we extend previous model and deploy CyberTraps on m_k enterprise networks (E_{c_1}, ... $E_{c_{m_k}}$). Without loss of generality, we denote them as the first m_k enterprise networks, i.e., $c_k = k, k \in \{1..m_k\}$. We further differentiate two scenarios based on whether there exists cooperation among deployed CyberTraps or not.

If there is no cooperation among them, the Eq (4, 5, and 6) could be simply extended as follows:

$$\frac{dB_{E_i}(t)}{dt} = \begin{cases} C_{E_i,1}(I(t) - I_{E_i}(t) - B_{E_i}(t)) & \text{if } i \le m_k \\ 0 & \text{if } m_k < i \le m \end{cases} \quad (8)$$

$$\frac{dV_{E_i}(t)}{dt} = \begin{cases} -C_0(I(t) - B_{E_i}(t))V_{E_i}(t) & \text{if } i \le m_k \\ -C_0 I(t) V_{E_i}(t) & \text{if } m_k < i \le m \end{cases} \quad (9)$$

$$\frac{dR_{E_i}(t)}{dt} = \begin{cases} C_{E_i,2} I_{E_i}(t) & \text{if } i \le m_k \\ 0 & \text{if } m_k < i \le m \end{cases} \quad (10)$$

Combined with Eq (7), these equations (8, 9, and 10) represent uncooperative CyberTrap model with m_k deployment.

However, if different CyberTraps are cooperative in that each CyberTrap will notify other responsible CyberTraps and share blacklists once it detects some worm sources in other domains, the Eq (8, 9, and 10) can be further refined as follows:

$$\frac{dB_{E_i}(t)}{dt} = \begin{cases} C'_{E_i,1}(I'_{E_i}(t) - B_{E_i}(t)) & \text{if } i \le m_k \\ 0 & \text{if } m_k < i \le m \end{cases} \quad (11)$$

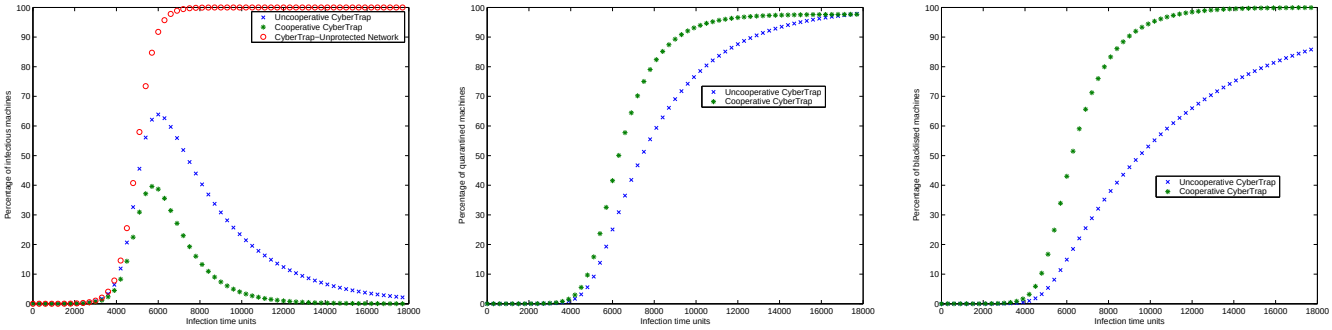
$$\frac{dV_{E_i}(t)}{dt} = \begin{cases} -C_0(I(t) - B_{E_i}(t))V_{E_i}(t) & \text{if } i \le m_k \\ -C_0 I(t) V_{E_i}(t) & \text{if } m_k < i \le m \end{cases} \quad (12)$$

$$\frac{dR_{E_i}(t)}{dt} = \begin{cases} C'_{E_i,2} I_{E_i}(t) & \text{if } i \le m_k \\ 0 & \text{if } m_k < i \le m \end{cases} \quad (13)$$

where $C'_{E_i,1} = \sum_{1 \le j \le m_k} C_{E_j,1}$, $C'_{E_i,2} = \sum_{1 \le j \le m_k} C_{E_j,1} + C_{D_{shared}}$, and $I'_{E_i}(t) = \sum_{m_k < j \le m} I_{E_j}(t)$.

In order to compare their effectiveness, we show one example deployment of CyberTrap with $m_k = 4$ and a class B network as internal darknet size and derive the numerical solutions in Figure 7(a), 7(b) and 7(c). Other parameters are the same as in table 2. As expected, cooperative CyberTraps performs better than isolated CyberTraps. This is because of the impact of federation of darknet size from different enterprises. However, even without cooperation among CyberTraps, multiple deployment still achieves better results than single deployment, which can be shown by comparing the $/16$ curve in Figure 5(a) and the uncooperative curve in Figure 7(a). We also extended the uniform-scan worm simulator originally developed by Zou [37] and the simulation results matching the numerical results very well. Particularly, a $/10$ external darknet space in one enterprise could effectively limit infectious worm percentage within that enterprise to less than 3%.

However, if the cooperation relationship is not established securely and safely, it might be vulnerable and abused to intentionally quarantine legitimate users. To enable a secure cooperation, it is necessary to authenticate the identity of cooperating CyberTraps and verify the validity of infection evidence. In following section, we further examine operational requirements, potential attacks, and further improvement related to CyberTrap.



(a) Percentage of Infectious Nodes (b) Percentage of Quarantined Nodes (c) Percentage of Blacklisted Nodes

Figure 7. Un-cooperative CyberTraps v.s. Cooperative CyberTraps

4 Discussion

As shown in section 3.2, the responsiveness and effectiveness of CyberTrap relies on the *availability* and *obscurity* of darknet space. Fortunately, CyberTrap darknet space size requirement (e.g., a $/13$ network) seems reasonable and affordable. For example, CAIDA [4] has used a $/8$ network at UCSD and two $/16$ networks at Lawrence Berkeley Laboratory (LBL) to collect real data measuring the spread of the Code Red v2 worm. Four class B networks (a $/14$ network) have also been used as Internet sinks [36] monitoring how the network is abused. CyberTrap better utilizes these darknet spaces and reuses the same external darknet space in every deployed enterprise network, which results in (1) better network efficiency by reducing unnecessary Inter-AS traffic; (2) improved scalability of CyberTrap by essentially anycasting the external darknet space to each protected enterprise network; and (3) additional feasibility in effective quarantine of worm nodes from the source.

However, the requirement for obscurity is a controversial one and is also one of the sources for potential attacks. Even though similar technique like honeypot [3, 36, 14, 18] has proved effective in practice in detecting known or even unknown attacks, security by obscurity is still an undesirable property. Similarly, those darknet space used by CyberTrap could be disclosed after a sufficiently long time. There are several practical mitigation schemes: (1) *Roaming* CyberTrap can be proposed so that CyberTrap does not rely on fixed darknet network space; (2) *Scattered* CyberTrap selects distributed darknet space which could reduce certain disclosure risks. Furthermore, a *randomization* scheme could be adopted so that the darknet is chosen randomly to minimize the risks. The randomization requirement imposes the ability of on-demand instantiation of vulnerable services. We have successfully developed a prototype called BAIT-TRAP[17] which has been deployed to take advantage of scattered darknet space and dynamically start a virtual machine with required vulnerable services.

Besides the attacks locating CyberTrap space, there is another attack called *impersonation* attack in which one worm could initiate an infection with spoofed source address. When such infection is detected by CyberTrap, an ignorant counter-measure may insert a new firewall rule to quarantine the spoofed but legitimate node. Such abusing attempts need to be detected and avoided. CyberTrap triggers the attempt with a virtual machine and collect necessary evidence before active quarantine takes place. Source-address checking and, similarly, Unicast Reverse Path Forwarding, could be further enabled to detect and prevent such spoofing attacks.

It should be noted that address blacklisting in CyberTrap only partially quarantine external worm-infected nodes and permanent blacklist could prevent them from later legitimate access. A more graceful approach is to associate with the blacklist rule a configurable time-out value. Dynamic firewall tools like [1] have been available for this purpose.

The concept of CyberTrap presented in this paper is *reactive*. The counterpart, a *proactive* CyberTrap, can also be deployed when a vulnerability is identified and the exploiting worms have not yet emerged. Such proactive CyberTrap can be safely activated in each network domain, and it will actively probe and detect vulnerable machines within its own domain. Once a vulnerable machine is found, necessary counter-measures like shield [30] could be deployed to prevent it from being exploited in the future.

5 Related Work

Modeling, detecting, and quarantining worms have drawn significant attention due to observed outrages of various worms [4, 5, 9, 8]. In the following, we examine related work in these areas:

Worm Modeling Accurate models could give insights into mitigating worm spreadings by examining various fac-

tors which influence their spread. Kephart and White *et al.* [20] proposed a classic epidemiological model to measure computer virus prevalence. Zou *et al.* [38] analyzed the propagation of the Code Red worm and presented *two-factor model* by taking into account network congestions and human counter-measures for worm propagation. Chen *et al.* [13] further considered parameters such as the worm scan rate, the vulnerability patching rate, and the victim death rate and proposed a concise discrete-time worm model, i.e., *AAWP* model. However, they did not consider each individual peering AS in current Internet and have not analyzed defense mechanisms in great depth.

Early Detection Timely detection of worms at early stage is critical in mitigating malicious spreadings. Virulent worms could cause certain traffic characteristics like abnormalities in overall traffic and similarities within worm traffic. These traffic characteristics could be leveraged for detecting the existence of worms. EarlyBird[25] examines *heavy hitter* and *many flows* in Internet traffic to infer the existence of worms. Based on highly repetitive content in worm traffic, EarlyBird further extracts worm signatures automatically. However, polymorphic or metamorphic worms impose a significant challenge by obfuscating worm payloads. Packet Matching [12] detects worm probing traffic by matching destination port numbers between incoming and outgoing connections and blocks those traffic once identified accordingly. Different from Packet Matching, CyberTrap takes advantage of darknet space to detect the existence of worm and thus is able to achieve nearly-zero false-positive (correctly identify a worm node once detected) and very low false-negative (false to detect the existence of worm nodes).

As mentioned before, darknet has advantages over normal networks in its ability collecting highly concentrated malicious traffic. With the same observation, Network Telescope[22], Internet Motion Sensor[2], and iSink[36] explore one or a set of dedicated darknet spaces for inferring certain remote network events, sensing Internet motions, and understanding network abuse. However, these approaches (1) are either passively monitoring these background radiation traffic or interacting with them in a limited fashion; and (2) did not further propose counter-measures to mitigate worm propagation. Instead, CyberTrap enables full-interaction with dynamically instantiated virtual machines and takes a further step in attempting to reactively quarantine detected worm nodes. Also with deployment within each peering enterprise networks, CyberTrap has the authoritative to block worm nodes or filter relevant traffic at the source.

Dynamic Quarantine Accurate worm modeling and early detection need to be followed by dynamic quarantine mechanisms in order to successfully curtail worm outrages. Williamson *et al.*[29] proposed the idea of host-based rate limiting by restricting the number of new outgoing connections. Chen *et al.*[11] designed a temporal rate-limit algorithm and a spatial rate-limit algorithm to make the speed of worm propagation configurable by the parameters of their defense system, i.e., *DAW*. Zou *et al.* [39] suggested to quarantine a host whenever its behavior looks suspicious by blocking traffic on its anomaly port. Then the quarantine is released after a short time, even if the host has not been inspected by security staffs yet. Weaver [31] suggested to break the network into many small *cells* and limited a worm's spread by isolating it in the cell. Wong *et al.* [34] examined the placement of rate-limiting filter and found that (1) backbone routers could be effective in limiting randomly-scanning worms and (2) a reasonable rate limits for an enterprise network would severely restrict the spread of a worm with negligible impact on almost all legitimate traffic. More generally, Moore *et al.* [23] examined the design space for worm containment systems and studied the efficacy of address blacklisting and content filtering. CyberTrap complements these approaches and further takes feasibility of counter-measures into consideration: CyberTrap actively quarantines nodes within its authoritative domain while blacklisting those nodes infecting from outside. Additionally, CyberTrap further enables the cooperation among different domains which could further slow-down worm spreadings.

6 Conclusion

Increased frequency and virulence of worm outbreaks significantly challenge the safety and reliability of any enterprise network and current shared Internet infrastructure. This paper proposes a systematic CyberTrap approach to detect and quarantine worm spreadings. CyberTrap leverages available darknet space for worm capture, utilizes virtual machines for triggering infection, and actively quarantines active worms by traffic filtering. The effectiveness and responsiveness of CyberTrap have been evaluated and demonstrated with analysis and simulation results.

References

[1] Dynamic Firewall Tools - dynfw. *http://www.gentoo.org/proj/en/dynfw.xml*.

[2] Internet Motion Sensor. *http://ims.eecs.umich.edu/*.

[3] The Honeynet Project. *http://www.honeynet.org*.

[4] Code Red Worms. *CAIDA Analysis of Code-Red Worms http://www.caida.org/analysis/security/code-red/*, 2001.

[5] MSBlaster Worms. *CERT Advisory CA-2003-20 W32/Blaster Worms http://www.cert.org/advisories/CA-2003-20.html*, Aug. 2003.

[6] SQL/Slammer Worms. *CERT Advisory CA-2003-04 MS-SQL Server Worms http://www.cert.org/advisories/CA-2003-04.html*, Jan. 2003.

[7] CERT/CC Statistics. *CERT Coordination Centre, http://www.cert.org/stats/cert_stats.html*, 2004.

[8] Sasser Worms. *http://www.microsoft.com/security/incident/sasser.asp*, May 2004.

[9] Witty Worms. *http://securityresponse.symantec.com/avcenter/venc/data/w32.witty.worm.html*, Mar. 2004.

[10] S. Chen and S. Ranka. An Internet-Worm Early Warning System. *Proceedings of the IEEE Globecom 2004 - Security and Network Management, Dallas Texas, USA*, Nov. 2004.

[11] S. Chen and Y. Tang. Slowing Down Internet Worms. *Proceedings of the 24th International Conference on Distributed Computing Systems (ICDCS'04), Tokyo, Japan*, Mar. 2004.

[12] X. Chen and J. Heidemann. Detecting Early Worm Propagation through Packet Matching. *Technical Report ISI-TR-2004-585, USC/Information Sciences Institute*, Feb. 2004.

[13] Z. Chen, L. Gao, and K. Kwiat. Modeling the Spread of Active Worms. *INFOCOM 2003, San Francisco, CA*, Mar. 2003.

[14] D. Dagon, X. Qin, G. Gu, W. Lee, J. Grizzard, J. Levine, and H. Owen. HoneyStat: Local Worm Detection Using Honeypots. *Proceedings of the 7th International Symposium on Recent Advances in Intrusion Detection (RAID 2004), Sophia Antipolis, French Riviera, France*, Sept. 2004.

[15] R. Dantu, J. Cangussu, and A. Yelimeli. Dynamic Control of Worm Propagation. *Proceedings of the IEEE International Conference on Information Technology: Coding and Computing (ITCC'04) Volume 1*, Apr. 2004.

[16] H. W. Hethcote. The Mathematics of Infectious Diseases. *SIAM Review, vol. 42, no. 4, pp. 599-653*, 2000.

[17] X. Jiang and D. Xu. BAIT-TRAP: a Catering Honeypot Framework. Aug. 2004.

[18] X. Jiang and D. Xu. Collapsar: A VM-Based Architecture for Network Attack Detention Center. *Proceedings of the USENIX 13th Security Symposium, San Diego, USA*, Aug. 2004.

[19] X. Jiang and D. Xu. Worm Meets Beehive. *Department of Computer Sciences Technical Report CSD TR 04-027, Purdue University*, May 2004.

[20] J. O. Kephart and S. R. White. Measuring and Modeling Computer Virus Prevalence. *Proc. of the 1993 IEEE Computer Society Symposium on Research in Security and Privacy, 2-15*, May 1993.

[21] H. A. Kim and B. Karp. Autograph: Toward Automated, Distributed Worm Signature Detection. *Proceedings of the 13th Usenix Security Symposium (Security 2004), San Diego, CA*, Aug. 2004.

[22] D. Moore. Network Telescopes: Observing Small or Distant Security Events. *Proc. of the 11th USENIX Security Symposium (Security '02), San Francisco, CA*, Aug. 2002.

[23] D. Moore, C. Shannon, G. M. Voelker, and S. Savage. Internet Quarantine: Requirements for Containing Self-Propagating Code. *Proceedings of the IEEE Infocom Conference, San Francisco, CA*, Apr. 2003.

[24] S. E. Schechter, J. Jung, and A. W. Berger. Fast Detection of Scanning Worm Infections. *Proceedings of the 7th International Symposium on Recent Advances in Intrusion Detection (RAID), Sophia Antipolis, French Riviera, France*, Sept. 2004.

[25] S. Singh, C. Estan, G. Varghese, and S. Savage. Automated Worm Fingerprinting. *Proceedings of the ACM/USENIX Symposium on Operating System Design and Implementation, San Francisco, CA*, Dec. 2004.

[26] E. Spafford. The Internet Worm Program: an Analysis. *Purdue CS Technical Report TR-CSD-823*, 1988.

[27] S. Staniford, G. Grim, and R. Jonkman. Flash Worms: Thirty Seconds to Infect the Internet. *http://www.silicondefense.com/flash*, Aug. 2001.

[28] S. Staniford, V. Paxson, and N. Weaver. How to 0wn the Internet in Your Spare Time. *Proceedings of the USENIX 11th Security Symposium, San Francisco, USA*, Aug. 2002.

[29] J. Twycross and M. M. Williamson. Implementing and Testing a Virus Throttle. *Proceedings of the USENIX 12th Security Symposium, Washington DC, USA*, Aug. 2003.

[30] H. J. Wang, C. Guo, D. R. Simon, and A. Zugenmaier. Shield: Vulnerability-Driven Network Filters for Preventing Known Vulnerability Exploits. *SIGCOMM 2004*, Sept. 2004.

[31] N. Weaver, S. Staniford, and V. Paxson. Very Fast Containment of Scanning Worms. *Proceedings of the USENIX 13th Security Symposium, San Diego, USA*, Aug. 2004.

[32] E. W. Weisstein. Logistic Equation. *http://mathworld.wolfram.com/LogisticEquation.html*.

[33] P. J. Welcher and G. Moerschel. Cisco PIX Firewall Basics. *http://www.netcraftsmen.net/welcher/papers/pix01.html*.

[34] C. Wong, C. Wang, D. Song, S. Bielski, and G. R. Ganger. Dynamic Quarantine of Internet Worms. *Proceedings of the International Conference on Dependable Systems and Networks (DSN-2004). Palazzo dei Congressi, Florence, Italy*, June 2004.

[35] J. Wu, S. Vangala, L. Gao, and K. Kwiat. An Effective Architecture and Algorithm for Detecting Worms with Various Scan Techniques. *Proceedings of the ISOC Network and Distributed System Security Symposium*, Feb. 2004.

[36] V. Yegneswaran, P. Barford, and D. Plonka. On the Design and Use of Internet Sinks for Network Abuse Monitoring. *Proc. of 7th International Symposium on Recent Advances in Intrusion Detection*, Sept. 2004.

[37] C. C. Zou. Internet Worm Propagation Simulator. *http://tennis.ecs.umass.edu/czou/research/wormSimulation.html*.

[38] C. C. Zou, W. Gong, and D. Towsley. Code Red Worm Propagation Modeling and Analysis. *Proc. of the 9th ACM Conference on Computer and Communication Security (CCS'02), Washington DC*, Nov. 2002.

[39] C. C. Zou, W. Gong, and D. Towsley. Worm Propagation Modeling and Analysis under Dynamic Quarantine. *Proceedings of the ACM CCS Workshop on Rapid Malcode (WORM'03), Washington DC, USA*, Oct. 2003.