2004

Monitoring and Early Detection for Internet Worms

Cliff C. Zou
University of Massachusetts - Amherst

Weibo Gong
University of Massachusetts - Amherst

Don Towsley
University of Massachusetts - Amherst

Lixin Gao
University of Massachusetts - Amherst

Monitoring and Early Detection for Internet Worms

Cliff C. Zou, Weibo Gong, Don Towsley, Lixin Gao
University of Massachusetts at Amherst
{czou, gong, lgao}@ecs.umass.edu, towsley@cs.umass.edu

***Abstract*— After several Internet-scale worm incidents in recent years, it is clear that a simple self-propagating worm can quickly spread across the Internet and cause severe damage to our society. Facing this great security threat, we must build an early detection system to detect the presence of a worm as quickly as possible in order to give people enough time for counteractions. In this paper, we first present an Internet worm monitoring system. Then based on the idea of "detecting the trend, not the burst" of monitored illegitimate traffic, we present a non-threshold based "trend detection" methodology to detect a worm at its early stage by using Kalman filter estimation. In addition, for uniform scan worms such as Code Red and Slammer, we can effectively predict the overall vulnerable population size, and estimate accurately how many computers are really infected in the global Internet based on the biased monitored data. For monitoring of non-uniform scan worms such as Blaster, we show that the address space covered by a monitoring system should be as distributed as possible.**

I. Introduction

Since the Morris worm in 1988 [21], the security threat posed by worms has steadily increased, especially in the last several years. In 2001, *Code Red* and *Nimda* infected hundreds of thousands of computers [17][4], causing millions of dollars loss to our society [27]. The SQL *Slammer* worm appeared on January 25th, 2003, and infected more than 90% of vulnerable computers in the Internet within 10 minutes [19]. On August 11th, 2003, *Blaster* hit us again and infected around 330,000 to one million computers within several days [26].

Currently, some organizations and security companies, such as the CERT, CAIDA, and SANS Institute [3][5][22], are monitoring the Internet and paying close attention to any abnormal traffic. When they observe abnormal network activities, their security experts will immediately analyze these incidents. However, until now no nation-scale malware monitoring and defense center exists. Given the fast spreading nature of Internet worms and their severe damage to our society, it is necessary to setup a nation-scale worm monitoring and early warning system.

In order to detect an unknown (zero-day) worm, a straightforward way is to use various threshold-based anomaly detection methods. We can directly use some well-studied methods established in the anomaly intrusion detection area. However, many threshold-based anomaly detections have the trouble in dealing with their high false alarm rate. In this paper, we do not try to propose another threshold-based anomaly detection method. Instead, we present a *non-threshold* based detection methodology, "*trend detection*", by using the principle "detecting monitored traffic *trend*, not *burst*" [30].

Traditional threshold-based anomaly detection methods try to detect a worm by detecting either the long-term or the short-term *burst* of monitored traffic. However, the monitored data contains noisy background traffic that is caused by many other factors besides the worm we want to detect, such as some old worms' scans or hackers' port scans. Thus traditional threshold-based detections usually will generate excessive false alarms. In the case of worm detection, we find that we can take advantage of the difference between a worm's propagation and a hacker's intrusion attack: a worm code exhibits simple attack behaviors and its propagation usually follows some dynamic models because of its large-scale infection; on the other hand, a hacker's intrusion attack, which is more complicated, usually targets one or a set of specific computers and does not follow any well-defined dynamic model in most cases.

Therefore, our "trend detection" system attempts to detect the dynamic *trend* of monitored traffic

based on the fact that at the early stage a worm propagates exponentially with a *constant, positive* exponential rate — the "trend" we try to detect is the exponential growth trend of monitored traffic.

Based on worm propagation dynamic models, we detect the propagation of a worm in its early stage by using *Kalman filter* estimation algorithms. The Kalman filter is activated when the monitoring system encounters a surge of illegitimate scan activities. If the infection rate estimated by the Kalman filter, which is also the exponential growth rate of a worm's propagation at its early stage, *stabilizes* and *oscillates* slightly around a *constant positive* value, we claim that the illegitimate scan activities are mainly caused by a worm, even if the estimated worm infection rate is still not well converged. If the monitored traffic is caused by non-worm noise, the traffic will not have the exponential growth trend, then the estimated value of infection rate would oscillate around zero. In other words, the Kalman filter is used to detect the presence of a worm by detecting the *trend*, not the *burst*, of the observed illegitimate traffic. In this way, the unpredictable, noisy illegitimate traffic we observe everyday will not cause too many false alarms to our detection system — such background noise will cause great trouble to traditional threshold-based detection methods.

In addition, we present a formula to predict a worm's vulnerable population size. We also present a formula to correct the bias in the number of infected hosts observed by a monitoring system— this bias has been mentioned in [6] and [20], but neither of them has presented methods to correct it. Furthermore, we point out that in designing a worm monitoring system, the address space covered by a monitoring system should be as distributed as possible in order to monitor and detect non-uniform scan worms, especially a sequential scan worm such as Blaster.

The rest of this paper is organized as follows. Section II surveys related work. In Section III, we introduces worm propagation models used in this paper. Section IV describes briefly the monitoring system for early detection of worms. Then we discuss data collection and provide the bias correction formula for monitored biased data in Section V. In Section VI, we present Kalman filters for early worm detection, and the formula to predict the vulnerable population size. We conduct extensive simulation experiments and show the major results in Section VII. In Section VIII, we discuss some possible future works. In the end, Section IX concludes this paper.

II. Related Work

In recent years, people have paid attention to the necessity of monitoring the Internet for malicious activities. Moore presented the concept of "network telescope", in analogy to light telescope, by using a small fraction of IP space to observe security incidents in the global Internet [20]. Yegneswaran *et al.* pointed out that there was no obvious addressing biases when using the "network telescope" monitoring methodology [28]. "Honeynet" is a network of honeypots to gather comprehensive information of attacks [11]. Symantec Corp. has an "enterprise early warning solution", which collects IDS and firewall attack data from the security systems of thousands of partners to keep track of the latest attack incidents [25]. The SANS Institute set up the "Internet Storm Center" in November 2000, which could gather the log data from participants' intrusion detection sensors distributed around the world [15]. It has quickly expanded to gather more than three million intrusion detection log entries every day. Berk *et al.* proposed a monitoring system by collecting ICMP "Destination Unreachable" messages generated by routers for packets to non-existent IP addresses [2]. Based on such a monitoring system, they presented a threshold-based detection system called TRAFFEN.

The monitoring system we present in this paper can be incorporated into the current monitoring systems such as the SANS "Internet Storm Center". Our contribution in this context is to point out the infrastructure specifically for worm monitoring, and what data should be collected for early detection of worms. We also emphasize the functionality of egress monitors, which has been overlooked in previous research. Worm monitors can be ingress or egress filters on routers, which cover more IP space and gather more comprehensive information than the log data collected from intrusion detection sensors or firewalls for current monitoring systems.

In the area of virus and worm modelling, Kephart, White and Chess of IBM performed a series of studies from 1991 to 1993 on viral infection based

on epidemiology models [12][13][14]. Staniford *et al.* [23] used the classical epidemic model to model the spread of Code Red right after the Code Red incident on July 19th, 2001. Their model matches well the increasing part of the observation data of Code Red. Zou *et al.* [29] presented a "two-factor" worm model that considered both the effect of human countermeasures and the effect of the congestion caused by extensive worm scan traffic. Chen *et al.* [6] presented a discrete-time version worm model that considered the patching and cleaning effect during a worm's propagation.

For a fast spreading worm such as Slammer, it is necessary to have automatic response and mitigation mechanisms. Moore *et al.* [18] discussed the effect of Internet quarantine for containing worm propagation. Zou *et al.* [33] presented a feedback dynamic quarantine system for automatic mitigation by borrowing two principles used in the epidemic disease control in the real world: "preemptive quarantine" and "feedback adjustment". However, both papers did not discuss how to detect a worm in its early stage. Staniford [24] presented worm quarantine for enterprise networks by using "CounterMalice" devices to separate an enterprise network into many isolated subnetworks. However, the device is used to detect a worm when some computers inside an enterprise network are infected, at which time the worm may have already infected most vulnerable computers in the Internet.

We assume that the IP infrastructure is the current IPv4. If IPv6 replaces IPv4, the 2^{128} IP space of the IPv6 would make it futile for a worm to propagate through blindly IP scans [31]. However, we believe IPv6 will not replace IPv4 in the near future, and worms will continue to use various random scan techniques to spread in the Internet.

III. Worm Propagation Model

A promising approach for modelling and evaluating the behavior of malware is the use of *fluid models*. Fluid models are appropriate for a system that consists of a large number of vulnerable hosts. The simple epidemic model assumes that each host resides in one of two states: susceptible or infected. The model further assumes that, once infected by a virus or a worm, a host remains in the infectious state forever. Thus any host has only one possible state transition: susceptible $\rightarrow$ infected [7]. The simple epidemic model for a finite population is

$$\frac{dI_t}{dt} = \beta I_t[N - I_t] \tag{1}$$

where I_t is the number of infected hosts at time t; N is the size of population; and β is called the *pairwise rate of infection* in epidemic studies [7]. At $t = 0$, I_0 hosts are infected while the remaining $N - I_0$ hosts are susceptible.

This model captures the basic mechanism of a worm's propagation, especially for the initial stage of a worm's propagation when the effect of human counteractions and network congestion is ignorable [29].

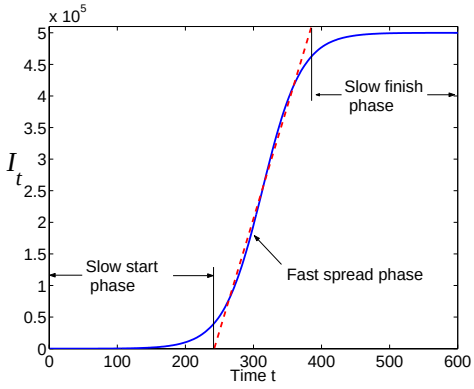


Fig. 1. Worm propagation model

For the epidemic model (1), Fig. 1 shows the dynamics of I_t as time goes on for one set of parameters. We can roughly partition a worm's propagation into three phases: the slow start phase, the fast spread phase, and the slow finish phase. During the *slow start phase*, since $I_t \ll N$, the number of infected hosts increases exponentially (model (1) becomes $dI_t/dt \approx \beta N I_t$). After many hosts are infected and then participate in infecting others, the worm enters the *fast spread phase* where vulnerable hosts are infected at a fast, near linear speed. When most of vulnerable computers have been infected, the worm enters the *slow finish phase* because the few leftover vulnerable computers are difficult for the worm to search out. Our task is to detect the presence of a worm in its slow start phase as early as possible.

At the early stage of a worm's propagation, $N - I_t \approx N$. Since we want to detect a worm at its slow start phase, we can accurately model a worm's propagation at this stage by using the exponential growth model:

$$\frac{dI_t}{dt} = \beta N I_t \tag{2}$$

TABLE I

NOTATIONS IN THIS PAPER

Notation	Definition
N	Number of hosts under consideration
Δ	The length of monitoring interval (time unit in discrete-time model)
I_t	Number of infected hosts at time $t\Delta$
β	Pairwise rate of infection
α	Infection rate per infected host, $\alpha = \beta N$
C_t	Cumulative number of infected hosts monitored by time $t\Delta$
Z_t	Monitored worm scan rate at time $t\Delta$
η	Average scan rate per infected host
p	Probability a worm scan is monitored
R	Variance of observation error of C_t
ν_t, ν_t', ν_t''	Observation noise in worm models
τ_t	Weight in Kalman filter formula
y_t	Measurement data in Kalman filter
w_t	White noise in measurement y_t at time $t\Delta$
δ	Constant in equation $y_t = \delta I_t + w_t$
MWC	Abbr. of "Malware Warning Center"
$\hat{\alpha}$	Estimated value of α
A^τ	Transpose of a matrix A
$N(\mu, \sigma^2)$	Normal distribution with mean μ and variance σ^2

which has the solution

$$I_t = I_0 e^{\beta N t} \tag{3}$$

In this paper, we use discrete-time model for worm modelling and early detection. Time is divided into intervals of length Δ, where Δ is the discrete time unit. To simplify the notations, we use "t" as the discrete time index from now on. For example, I_t means the number of infected hosts at time $t\Delta$. The discrete-time version of the simple epidemic model (1) can be written as [7]:

$$I_t = (1 + \alpha\Delta) I_{t-1} - \beta\Delta I_{t-1}^2 \tag{4}$$

where

$$\alpha = \beta N \tag{5}$$

We call α as *infection rate* because it is the average number of vulnerable hosts that can be infected per unit time by one infected host during the early stage of a worm's propagation.

For the exponential worm model (2), we derive an autoregressive (AR) discrete-time model similar to (4):

$$I_t = (1 + \alpha\Delta) I_{t-1} \tag{6}$$

which is called *AR exponential model* in this paper. We can also derive another discrete-time model by taking logarithm on both sides of the solution (3):

$$ln I_t = t\Delta\alpha + ln I_0 \tag{7}$$

which is called *transformed linear model* in this paper.

Before we go on to discuss how to use the worm models to detect and predict worm propagation, we first present the monitoring system design in the next Section IV and discuss data collection issues in Section V.

IV. MONITORING SYSTEM

In this section, we propose the architecture of a worm monitoring system. The monitoring system aims to provide comprehensive observation data on a worm's activities for the early detection of the worm. The monitoring system consists of a *Malware Warning Center* (MWC) and distributed monitors as shown in Fig. 2.

A. Monitoring System Architecture

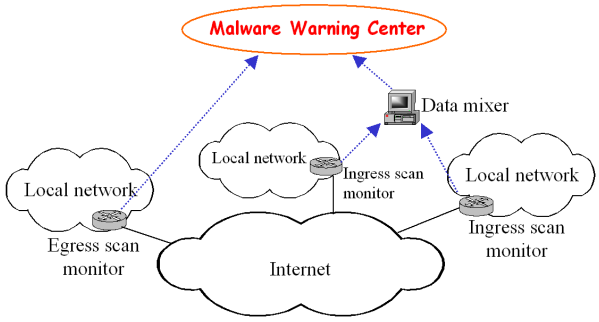


Fig. 2. A generic worm monitoring system

There are two kinds of monitors: ingress scan monitors and egress scan monitors. *Ingress scan monitors* are located on gateways or border routers of local networks. They can be the ingress filters on border routers of the local networks, or separated passive network monitors. The goal of an ingress scan monitor is to monitor scan traffic coming into a local network by logging incoming traffic to unused local IP addresses. For management reason, Local network administrators know how addresses inside their networks are allocated; it is relatively easy for them to set up the ingress scan monitor on routers in their local networks. For example, during the Code Red incident on July 19th, 2002, a /8 network at UCSD and two /16 networks at Lawrence Berkeley Laboratory were used to collect Code Red scan

traffic. All port 80 TCP SYN packets coming in to nonexistent IP addresses in these networks were considered to be Code Red scans [17].

An *egress scan monitor* is located at the egress point of a local network. It can be set up as a part of the egress filter on the routers of a local network. The goal of an egress scan monitor is to monitor the outgoing traffic from a network to infer a potential worm's scan behavior. Ingress scan monitors listen to the global traffic in the Internet; they are the sensors of the global worm incidents (or called "network telescope" in [20]). However, it is difficult to determine the behavior of each individual worm from the data collected by ingress scan monitors since such monitors cannot capture most of the scans sent out by an infected host. On the other hand, if a computer inside a local network is infected, the egress scan monitor on this network's routers can observe most of the scans sent out by the compromised computer. The closer the egress scan monitor is to an infected computer, the more accurate information could be referred about the worm's scan behavior.

For worm early warning at real-time, distributed monitors are required to send observation data to the MWC continuously without significant delay, even when the worm scan traffic has caused congestion to the Internet. For this reason, a tree-like hierarchy of *data mixers* can be set up between monitors and MWC: MWC is the root; the leaves of the tree are monitors. The monitors nearby a data mixer send observed data to the data mixer. After fusing the data together, the data mixer passes the data to a higher level data mixer or directly to MWC. An example of data fusion is the removal of repetitive IP addresses from the list of infected hosts. However, the tree structure of data mixers creates single points of failure, thus there is a trade-off in designing this hierarchical structure.

B. Location for Distributed Monitors

Ingress scan monitors on a local network may need to be put on several routers instead of only on the border router — the border router may not know the usage of all IP addresses of this local network. In addition, since worms might choose different destination addresses by using different preferences, we need to use distributed address spaces with different sizes and characteristics to ensure proper coverage. Later on, we show that for monitoring non-uniform scan worms such as Blaster, the IP space covered by a monitoring system should be as distributed as possible.

For egress scan monitors, worms on different infected computers will exhibit different behaviors. For example, Slammer's scan rate is constrained by an infected computer's bandwidth [19]. Therefore, we need to set up distributed egress filters to record the scan behaviors of many infected hosts at different locations and in different network environments. In this way, the monitoring system could obtain a comprehensive view of the behaviors of a worm.

V. Data Collection and Bias Correction

After setting up a monitoring system, we need to determine what kind of data should be collected. The main task for an egress scan monitor is to determine the behaviors of a worm, such as the worm's average scan rate and scan distribution. Denote η as the average worm scan rate, which is the *average* number of scans sent out by an infected host in a unit time. Thus in a monitoring interval Δ, an infected host sends out on average $\eta\Delta$ scans.

The ingress scan monitors record two types of data: the number of scans they receive during the t-th monitoring interval, $t = 1, 2, \cdots$ and the IP addresses of infected hosts that have sent scans to the monitors by time $t\Delta$.

If all monitors send observation data to MWC once in every monitoring interval, then MWC obtains the following observation data at each discrete time epoch t, $t = 1, 2, \cdots$:

(1). A worm's scan distribution, e.g., uniform scan or scan with address preference,

(2). A worm's average scan rate η,

(3). The number of scans monitored in a monitoring interval from time $(t-1)\Delta$ to $t\Delta$, denoted by Z_t,

(4). The cumulative number of infected hosts observed by time $t\Delta$, denoted by C_t.

In this paper, we primarily focus on worms that uniformly scan the Internet. Let p denote the probability that a worm scan is monitored by a monitoring system. If ingress scan monitors cover m IP addresses, then a worm scan has the probability $p = m/2^{32}$ to hit the monitoring system. We assume that in the discrete-time model all changes happen right before the discrete time epoch t, then we have

$$E[Z_t] = \eta \Delta p I_{t-1} \tag{8}$$

A. Correction of Biased Observation C_t

For a uniform scan worm, each worm scan has a small probability p of being observed by a monitoring system, thus an infected host will send out many scans before one of them is observed by ingress scan monitors, which follows a *Bernoulli trial* with a small success probability p. Therefore, the number of infected hosts monitored by time $t\Delta$, C_t, is not proportional to I_t. This bias has been mentioned in [6] and [20], but neither of them have presented methods to correct the bias. In the following, we present an effective way to obtain an accurate estimate for the number of infected hosts I_t based on C_t and η.

In real world, different infected hosts of a worm have different scan rates. To derive the bias correction formula, let us first assume that all infected hosts have the same scan rate η (we will show the effect of removing this assumption in the following simulation). In a monitoring interval Δ, a worm sends out $\eta\Delta$ scans on average, thus the monitoring system has the probability $1-(1-p)^{\eta\Delta}$ to detect at least one scan from an infected host in a monitoring interval.

At time $(t-1)\Delta$, the monitoring system has observed C_{t-1} infected hosts among the overall infected ones I_{t-1}. During the next monitoring interval from $(t-1)\Delta$ to $t\Delta$, every host of those not yet observed ones, $I_{t-1} - C_{t-1}$, has the probability $1-(1-p)^{\eta\Delta}$ to be observed. Suppose in the discrete-time model, all changes happen right before the discrete time epoch t, then the average number of infected hosts monitored by time $t\Delta$ conditioned on C_{t-1} is

$$E[C_t|C_{t-1}] = C_{t-1} + (I_{t-1} - C_{t-1})[1-(1-p)^{\eta\Delta}] \tag{9}$$

Removing the conditioning on C_{t-1} yields

$$E[C_t] = E[C_{t-1}] + (I_{t-1} - E[C_{t-1}])[1-(1-p)^{\eta\Delta}] \tag{10}$$

From it we can derive the formula for I_t as:

$$I_t = \frac{E[C_{t+1}] - (1-p)^{\eta\Delta} E[C_t]}{1-(1-p)^{\eta\Delta}} \tag{11}$$

Since $E[C_t]$ is unknown in one incident of a worm's propagation, we replace $E[C_t]$ by C_t and derive the estimate of I_t as

$$\hat{I}_t = \frac{C_{t+1} - (1-p)^{\eta\Delta} C_t}{1-(1-p)^{\eta\Delta}} \tag{12}$$

Now we analyze how the statistical observation error of C_t affects the estimated value of I_t. Without considering non-worm noise, suppose the observation data C_t is

$$C_t = E[C_t] + w_t \tag{13}$$

where the statistical observation error w_t is a white noise with variance R. Substituting (13) into (12) yields

$$\hat{I}_t = I_t + \mu_t \tag{14}$$

where the error μ_t is

$$\mu_t = \frac{w_{t+1} - (1-p)^{\eta\Delta} w_t}{1-(1-p)^{\eta\Delta}} \tag{15}$$

Since $E[\mu_t] = 0$, the estimated value $\hat{I}_t$ is unbiased (under the assumption that all infected hosts have the same scan rate η). The variance of the error of $\hat{I}_t$ is

$$Var[\mu_t] = E[\mu_t^2] = \frac{1+(1-p)^{2\eta\Delta}}{[1-(1-p)^{\eta\Delta}]^2} R \tag{16}$$

The equation above shows that $Var[\mu_t]$ is always larger than R, which means the statistical error of observation C_t is amplified by the bias correction formula (12). If ingress scan monitors cover smaller size of IP space, p would decrease, then (16) shows that the estimate $\hat{I}_t$ would become noisier.

We simulate Code Red propagation to check the accuracy of the bias correction formula (12). In the simulation, $N = 360,000$; the monitoring interval Δ is one minute; the average worm scan rate is $\eta = 358$ per minute.. The monitoring system covers 2^{17} IP addresses (equal to two Class B networks). Because different infected hosts have different scan rates, we assume each infected host has a scan rate x that is predetermined by the normal distribution $N(\eta, \sigma^2)$ where $\sigma = 100$ in the simulation (x is bounded by $x \geq 1$. We will explain how we choose these parameters in Section VII). The simulation result is shown in Fig. 3.

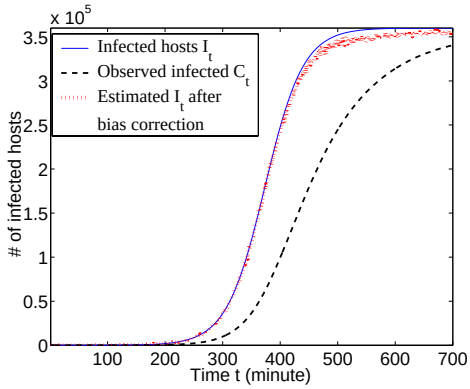


Fig. 3. Estimate $\hat{I}_t$ based on the biased observation data C_t (Monitoring 2^{17} IP space)

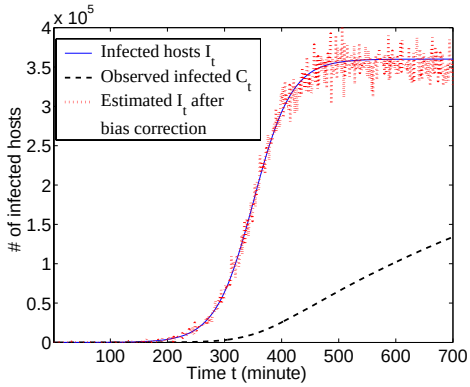


Fig. 4. Estimate $\hat{I}_t$ based on the biased observation data C_t (Monitoring 2^{14} IP space)

Fig. 3 shows that the observed number of infected hosts, C_t, deviates substantially from the real value I_t. After the bias correction by using (12), the estimated $\hat{I}_t$ matches I_t well in the simulation before the worm enters the slow finish phase ($\hat{I}_t$ deviates a little from I_t in the slow finish phase). In deriving the bias correction formula (12), we have assumed that all hosts have the same scan rate η, which is not the case in this simulation. In this simulation, some hosts have very small scan rate; these hosts will take much longer time to hit the monitoring system than others. Thus in the slow finish phase, many unobserved infected hosts are the ones with very low scan rate. Therefore, during the slow finish phase, the bias correction formula has some error due to the decreasing of the average scan rate for those unobserved infected hosts. In fact, we have run many other simulations by letting all hosts to have the same scan rate η (i.e., let $\sigma = 0$); then the $\hat{I}_t$ after bias correction always matches well with I_t without bias.

Fig. 4 shows the simulation results if the monitoring system only covers 2^{14} IP addresses. The estimate $\hat{I}_t$ after the bias correction is still accurate but noisier because of the error amplification effect described by (16).

VI. Early Detection and Estimation of Worm Virulence

In this section, we propose estimation methods based on recursive filtering algorithms (e.g., Kalman filters [1],) for stochastic dynamic systems. At MWC, we recursively estimate the parameter α based on observation data at each monitoring interval in order to detect a worm at its early stage.

Let $y_1, y_2, \cdots, y_t$, be the measurement data used by a Kalman filter estimation algorithm. Suppose the observations have one monitoring interval delay:

$$y_t = \delta I_{t-1} + w_t \tag{17}$$

where w_t is the observation error. δ is a constant ratio: if we use Z_t as y_t, then $\delta = \eta\Delta p$ as shown in (8); if we use $\hat{I}_{t-1}$ derived from C_t by the bias correction (12), then $\delta = 1$.

A. Early Detection Based on Kalman Filter Estimation

In Section III, we have presented three discrete-time worm models: the epidemic model (4), the AR exponential model (6), and the transformed linear model (7). In this section, we present three Kalman filter estimation algorithms, one for each discrete-time model.

From (17), we have

$$I_{t-1} = y_t/\delta - w_t/\delta \tag{18}$$

First, we use the simple epidemic model (4). Substituting (18) into the worm model (4) yields an equation describing the relationship between y_t and a worm's parameters α and β:

$$y_t = (1 + \alpha\Delta)y_{t-1} - \frac{\beta\Delta}{\delta}y_{t-1}^2 + \nu_t \tag{19}$$

where the noise ν_t is

$$\nu_t = w_t - (1+\alpha)w_{t-1} - \beta\Delta(w_{t-1}^2 - 2y_{t-1}w_{t-1})/\delta \tag{20}$$

A recursive least square algorithm for α and β can be cast into a standard Kalman filter format [1][16]. Let $\hat{\alpha}_t$ and $\hat{\beta}_t$ denote the estimated value

of α and β at time $t\Delta$, respectively. Define the system state as $X_t = \begin{bmatrix} 1+\Delta\alpha \\ -\beta\Delta/\delta \end{bmatrix}$. If we denote $H_t = [\ y_{t-1} \quad y_{t-1}^2\ ]$, then the system is described by

$$\begin{cases} X_t &= X_{t-1} \\ y_t &= H_t X_t + \nu_t \end{cases} \tag{21}$$

The Kalman filter in estimating X_t is

$$\begin{cases} H_t &= [\ y_{t-1} \quad y_{t-1}^2\ ] \\ K_t &= P_{t-1}H_t^\tau/(H_t P_{t-1} H_t^\tau + 1/\tau_t) \\ P_t &= (I - K_t H_t)P_{t-1} \\ \hat{X}_t &= \hat{X}_{t-1} + K_t(y_t - H_t\hat{X}_{t-1}) \end{cases} \tag{22}$$

where τ_t is the weight of the t-th error term in the Least Square (LS) estimation algorithm [16]. We can use it to adjust whether our estimation should rely more on recent monitored data (τ_t increases as t increases) or equally on all monitored data (τ_t is a constant).

ν_t in (20) is a correlated noise. The Kalman filter (22) can be extended to consider such correlated noise to derive unbiased estimates of α and β in theory (such as an *extended Kalman filter* [1]). However, an unbiased Kalman filter introduces additional parameters to estimate, thus the new filter will converge slower than the proposed filter (22). In fact, we have designed an extended Kalman filter and our experiments confirm this conjecture. In this paper the primary objective is to derive a rough estimate of α as quickly as possible for early worm detection. Therefore, it is better to use the simple Kalman filter (22) for early worm detection.

If we use Z_t as the measurement data y_t in the Kalman filter but do not know δ (e.g., if we do not have data from egress scan monitors), we still can estimate the infection rate α by letting $\delta = 1$. The Kalman filter (22) does not depend on δ in estimating α; the value of δ only affects the estimated value of β.

Now we consider the AR exponential model (6). Substituting (18) into model (6) yields:

$$y_t = (1+\Delta\alpha)y_{t-1} + \nu_t' \tag{23}$$

where the noise ν_t' is

$$\nu_t' = w_t - (1+\Delta\alpha)w_{t-1} \tag{24}$$

Equation (23) has the similar format as (19). Thus if we change X_t and H_t in the original Kalman filter (22) to $X_t = [1+\Delta\alpha]$ and $H_t = [y_{t-1}]$, we derive a new Kalman filter for early worm detection that is based on the AR exponential model (6).

For the transformed linear model (7), we can derive the formula of y_t as:

$$ln(y_t - w_t) = (t-1)\Delta\alpha + lnI_0 \tag{25}$$

In early worm detection, it's difficult or impossible for us to know when a worm starts spreading, i.e., we do not know the absolute value t. We only know a relative time $t - t_0$ where $t_0 > 0$ is the time when we activate our Kalman filter detection system — the true value of t_0 is not known. It means that in worm model we can only use variable $t - t_0$ but not t. If we let $ln(y_t - w_t) = ln(y_t) - \nu_t''$, from (25) we can derive the relationship between y_t and the worm's infection rate α as

$$ln(y_t) = (t - t_0)\Delta\alpha + K + \nu_t'' \tag{26}$$

where

$$K = (t_0 - 1)\Delta\alpha + ln\delta + lnI_0 \tag{27}$$

and the noise ν_t'' is

$$\nu_t'' = ln(y_t) - ln(y_t - w_t) \tag{28}$$

When we activate the Kalman filter in our early detection system, $y_t > 1$ and $y_t - w_t > 1$ always hold. From (28) we know that $sign(\nu_t'') = sign(w_t)$ and $|\nu_t''| < |w_t|$ because the logarithm function $y = ln(x)$ always increases slower than the function $y = x$ when x increases in the domain $x \in (1, \infty)$. In addition, from (28) we also know that

$$\frac{d|\nu_t''|}{dy_t} = -\frac{|w_t|}{y_t(y_t - w_t)} < 0 \tag{29}$$

Therefore, the noise ν_t'' in (26) decreases its magnitude when y_t increases as time goes on.

When we use the transformed linear model (7) for early worm detection, the system state vector for the Kalman filter is $X_t = \begin{bmatrix} \Delta\alpha \\ K \end{bmatrix}$. Now $H_t = [\ t - t_0 \quad 1\ ]$ and the system is described by

$$\begin{cases} X_t &= X_{t-1} \\ ln(y_t) &= H_t X_t + \nu_t'' \end{cases} \tag{30}$$

The Kalman filter in estimating X_t is

$$\begin{cases} H_t &= [\ t - t_0 \quad 1\ ] \\ K_t &= P_{t-1}H_t^\tau/(H_t P_{t-1} H_t^\tau + 1/\tau_t) \\ P_t &= (I - K_t H_t)P_{t-1} \\ \hat{X}_t &= \hat{X}_{t-1} + K_t[ln(y_t) - H_t\hat{X}_{t-1}] \end{cases} \tag{31}$$

B. Estimation of Vulnerable Population Size

For a uniform scan worm, we present an effective way to predict the population size N based on the observation data η and the estimate α from the Kalman filters above. In this way, we can know how many computers are vulnerable in the Internet when a worm is still in its slow start phase. A uniform scan worm sends out on average η scans per unit time; each scan has the probability $N/2^{32}$ to hit a host in the population under consideration. Hence, at the beginning when most hosts in the vulnerable population N are still vulnerable, a worm can infect on average $\eta N/2^{32}$ hosts per unit time (the probability of two scans sent out by a single infected host hitting the same target is negligible). From the definition of infection rate α, we have $\alpha = \eta N/2^{32}$. Therefore, the population N is

$$N = \frac{2^{32}\alpha}{\eta} \tag{32}$$

where the average worm scan rate η can be obtained directly from egress scan monitors in the monitoring system. When we use one of the Kalman filters above to estimate α, we can use (32) to estimate N along with the Kalman filter estimation. In this way, the estimation of N has similar convergence properties to the estimation of α from the Kalman filter.

C. Overview of the Steps to Detect a Worm

MWC collects and aggregates reports of worm scans from all distributed monitors once in every monitoring interval in real-time. For each TCP or UDP port, MWC has an alarm threshold for monitored illegitimate scan traffic Z_t. The observed number of scans Z_t, which contains non-worm noise, is below this threshold in most time when there is no global spreading worm. If the monitored scan traffic is over the alarm threshold for several consecutive monitoring intervals, e.g., Z_t is over the threshold for three consecutive times, the Kalman filter will be activated. Then MWC begins to record C_t and calculates the average worm scan rate η from the reports of egress scan monitors. Because C_t is a cumulative observation data that could cumulate all non-worm noise, MWC begins to record data C_t only after the Kalman filter is activated. The Kalman filter can either use C_t or Z_t to estimate all the parameters of a worm at time $t\Delta$ ($t = 1, 2, 3, \cdots$).

The recursive estimation will continue until the estimated value of α shows a trend: if the estimate $\hat{\alpha}$ stabilizes and oscillates slightly around a *positive constant* value, we have detected the presence of a worm; if the estimate $\hat{\alpha}$ oscillates around zero, we believe the surge of illegitimate monitored traffic is caused by non-worm noise.

VII. Simulation Experiments

A. Simulation Settings

We have simulated Code Red [9], SQL Slammer [19], and Blaster [10]. First, we explain how we choose the simulation parameters. In the case of Code Red, more than $359,000$ Code Red infected hosts were observed on July 19th, 2001 by CAIDA [17]. Thus in our simulation we set the Code Red vulnerable population $N = 360,000$. Staniford *et al.* [23] used a different format but the same epidemic model as (1) to model Code Red, where their model's parameter K has $K = \beta N = \alpha$ [29]. They determined that $K = 1.8$ for the time scale of one hour. Therefore, for the discrete time unit of one minute in our simulation, $\alpha = 1.8/60 = 0.03$. From (32) we can reversely derive $\eta = 2^{32}\alpha/N = 358$ per minute, i.e., Code Red sends out on average about 358 scans per minute per infected host.

Because different infected hosts have different scan rates, we assume that each infected host has a constant scan rate x, a rate that is independently predetermined by a normal distribution $N(\eta, \sigma^2)$ where $\sigma = 100$ (the scan rate x is bounded by $x \geq 1$). In our simulation, ingress scan monitors cover 2^{20} IP space. We also assume $I_0 = 10$ at the beginning.

Because of the sequential scan used by Blaster, until now people have not got an accurate estimation of how many computers were really infected by this worm when it propagated on August 11th 2003 (we will explain the reason later in our experiments). Since we want to understand how the sequential scan affects a worm's propagation and our early detection system, in this paper we assume that Blaster has the same parameters as Code Red, i.e., $N = 360,000$, $\eta = 358$ per minute, each worm's scan rate x follows normal distribution $N(358, 100^2)$ with the bound $x \geq 1$, and $I_0 = 10$ at the beginning.

For a uniform scan worm, such as Code Red and Slammer, the distribution of vulnerable hosts in the Internet will not affect the worm's propagation.

However, this distribution affects the propagation of Blaster [32] because of its sequential scan. Since we do not know the true distribution of vulnerable hosts in the Internet, in our Blaster worm simulation, we assume vulnerable hosts are uniformly distributed in the IP space defined by BGP routing prefixes, which is less than 30% of the entire IPv4 space [31][32].

In the discrete-time simulation, the monitoring interval Δ is set to be one minute for Code Red and Blaster. SQL Slammer propagates much faster and can finish infection in about 10 minutes [19]. Hence the monitoring interval should be much shorter in order to catch the dynamics of this worm. For this reason, the monitoring interval for Slammer is set to be one or several seconds.

B. Background Noise Consideration

We need to consider background non-worm noise in our simulations. Fortunately, Goldsmith [8] provided simple data of the background noise for Code Red activities monitored on a Class B network (covers 2^{16} IP addresses). He recorded TCP port 80 SYN requests from Internet hosts to any unused IP addresses inside his local network — such data are exactly the monitored data collected by ingress scan monitors in our proposed monitoring system. His monitored data showed that the background noise was small compared to Code Red traffic and the noise did not vary much. If we use normal distribution to model the background noise, then for each hour the number of noise scans follows $N(110.5, 30^2)$ and the number of source hosts that send noise follows $N(17.4, 3.3^2)$.

We try to hold the statistics of the observed background noise in our experiments: we monitor 2^{20} IP space, which is 16 times as large as what Goldsmith monitored, so the number of noise scans or noise sources should be enlarged by 16 times; we use one minute in stead of one hour as the monitoring interval, thus we should decrease the number of noise scans or noise sources by 60. In this way, in our Code Red and Blaster simulations, the noise added into the observation data at each monitoring interval follows $N(29.5, 8^2)$ for Z_t and $N(4.63, 0.893^2)$ for C_t. Of course, this kind of extension of noise is very rough, but it is the best we can do based on the data available. Currently, we are trying to obtain detailed log data on previously appeared worms from other researchers in order to have more realistic experiments.

In the simulation experiments, the alarm threshold for Z_t is set to be two times as large as the mean value of the background noise, i.e., the alarm threshold is $29.5 \times 2 = 59$. The Kalman filter we use in early detection will be activated when the monitored scan traffic Z_t is over the alarm threshold for three consecutive monitoring intervals. In this way, the Kalman filter will not be frequently activated by the surge of background noise traffic in the normal days.

C. Code Red Simulation and Early Detection

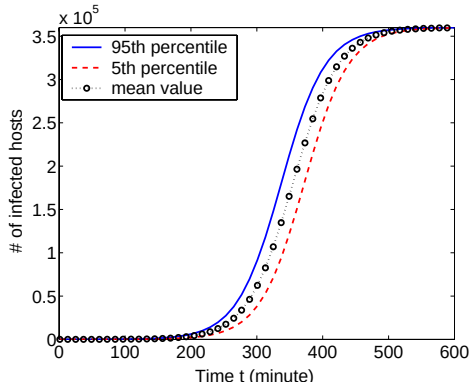
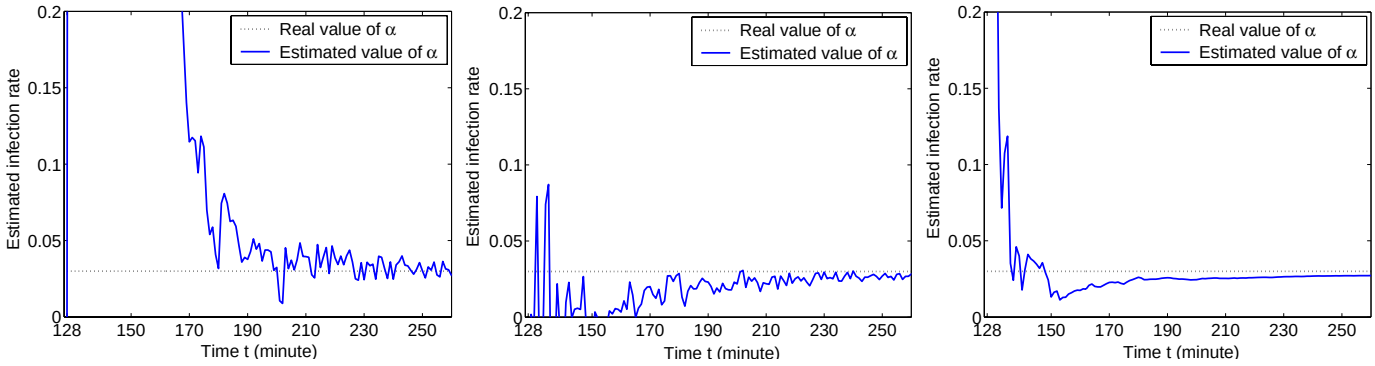


Fig. 5. Code Red propagation and its variability (100 simulation runs)

We simulate Code Red propagation for 100 simulation runs. Fig. 5 shows the number of infected hosts as a function of time for three cases: the average value, the 95th percentile, and the 5th percentile. The curve of 95th percentile means that in 95 out of our 100 simulation runs, Code Red propagates no faster than what this curve represents.

This figure shows that a worm propagates slightly differently in different sample runs. The propagation speed difference is mainly caused by a worm's spreading at the beginning when only several infected hosts scan and attempt to infect others. In fact, we have chosen $I_0 = 1$ and run Code Red propagation for another 100 simulation runs. It shows that Code Red in the $I_0 = 1$ case propagates more variously than the one shown in Fig. 5 where $I_0 = 10$.

For one Code Red propagation simulation run, Fig. 6 shows the estimation of the worm infection rate α as a function of time by using three Kalman filters based on three discrete-time models: epidemic model (4), AR exponential model (6), and transformed linear model (7), respectively. This figure



a. Based on epidemic model (4) b. Based on AR exponential model (6) c. Based on transformed linear model (7)

Fig. 6. Kalman filter estimation of Code Red infection rate α (for one simulation run)

shows the estimates by using the monitored data Z_t. We can use either the monitored data Z_t or the data C_t after bias correction (12) to estimate α for Code Red — they provide the similar estimation results [30]. Later on when we study the early detection of Blaster, because of its non-uniform scan, we cannot use the bias correction (12) for the monitored data C_t anymore and have to rely on the monitored data Z_t in our early detection. Therefore, in this paper we will only discuss early worm detection by using the monitored data Z_t.

In this simulation run, Z_t at time 126, 127, and 128 minutes are over the alarm threshold 59, thus the Kalman filter is activated at time 128 minutes. Fig. 6 shows that the Kalman filter estimation based on the transformed linear model provides much better estimation result than the other two. This is because the noise ν_t'' introduced by the transformed linear model (7) is smaller than the noise ν_t and ν_t' introduced by the other two models.

The noise ν_t, ν_t' and ν_t'' introduced by these three models are shown in (20), (24), and (28), respectively. We can see that the magnitude of the noise ν_t'' (28) introduced by the transformed linear model decreases as time goes on as shown in (29). On the other hand, the magnitude of the noise ν_t' (24) introduced by the AR exponential model does not change; the magnitude of the noise ν_t (20) introduced by the epidemic model even increases as time goes on (because y_{t-1} in (20) increases as time goes on).

Because of the decreasing noise ν_t'' in the transformed linear model, we select $\tau_t = t^2$ in the Kalman filter (31) of the transformed linear model in order to put more weight on the newest less noisy observation data. On the other hand, we select $\tau_t \equiv 1$ for both the Kalman filters of the epidemic model and the AR exponential model.

In the Code Red simulation run shown in Fig. 6, the worm infects 0.5% of vulnerable computers in the Internet at time 174 minutes. If we use the transformed linear model in our early detection, Fig. 6(c) shows that the estimate $\hat{\alpha}$ has already stabilized at a positive, constant value by that time. Therefore, we can detect the presence of Code Red when it has only infected 0.5% of all vulnerable population in the Internet. We have done such early detection by using Kalman filters for many simulation runs and have achieved the similar early detection performance. In our previous paper [30], we have shown that the early detection system can achieve the similar detection performance — detect a worm when it infects a similar small percentage of vulnerable population — no matter whether this worm propagates faster or slower in different simulation runs.

Note that the estimate $\hat{\alpha}$ shown in Fig. 6, especially the estimate in Fig. 6(c), is smaller than its true value. This bias is not caused by Kalman filters or worm models, but caused by the non-negative background noise in the monitored data Z_t. Fig. 7 shows the infected hosts I_t compared with the monitored data Z_t during the slow start phase of Code Red propagation in the logarithm format (after scaling Z_t to have the similar value scale as I_t). This figure clearly shows that Code Red propagates exponentially at the beginning — the exponential growth rate is in fact the worm infection rate α estimated by our Kalman filters. From this figure, we see that the exponential growth rate of Z_t (the

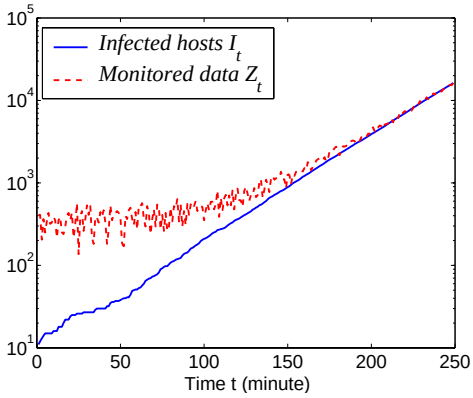


Fig. 7. Comparison of the infected hosts I_t with the monitored data Z_t at the beginning of Code Red propagation (log-format on Y-axis)

data we use in Kalman filter estimation) is slightly smaller than the real exponential growth rate of I_t. Therefore, the estimate in Fig. 6(c) is smaller than its true value α. This estimation bias, however, does not matter much because our primary objective is to detect the presence of a worm, not to get an accurate estimation of the worm's infection rate.

We predict the vulnerable population size N from (32) at each discrete time when we update the estimate of α from Kalman filters. Fig. 8 shows the estimated value of N as a function of time based on the Kalman filter of transformed linear model (31) and the Kalman filter of epidemic model (22), respectively. The estimated $\hat{N}$ from transformed linear model is smaller than the real value N because of the bias of $\hat{\alpha}$ shown in Fig. 6(c). In a real implementation, we should combine both estimation curves shown in Fig. 8 to predict the vulnerable population size N.

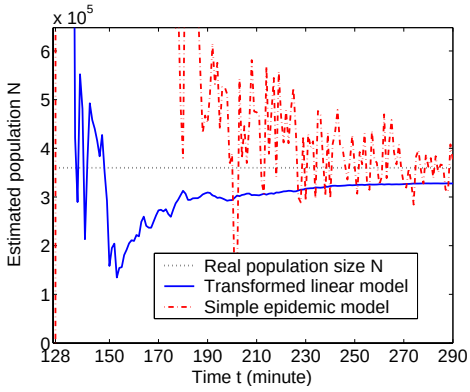


Fig. 8. Estimate of the vulnerable population size N for Code Red

Because Slammer propagates in the same way as Code Red by uniformly scanning the Internet, its propagation and its early detection are very similar to what Code Red has [30]. Therefore, we do not repeatedly show the early detection of Slammer in this paper.

D. Blaster Simulation and Early Detection

Each Blaster infected host scans the entire IP space sequentially from a selected starting point. To select this starting IP address, each worm copy has a 40% probability to choose the first address of its "Class C"-size subnet (x.x.x.0), and a 60% probability to choose a completely random IP address [10].

Because of Blaster's sequential scan mechanism, the distribution of vulnerable hosts in the IPv4 space affects the worm's propagation behavior. Since we do not know the real distribution of vulnerable hosts in the Internet, we use BGP routing tables and assume that vulnerable hosts are uniformly distributed in the BGP routable space, which is less than 30% of the entire IPv4 space [31].

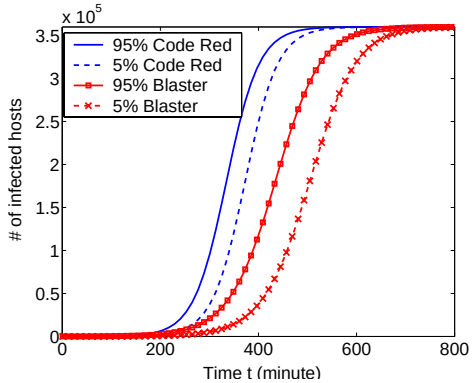
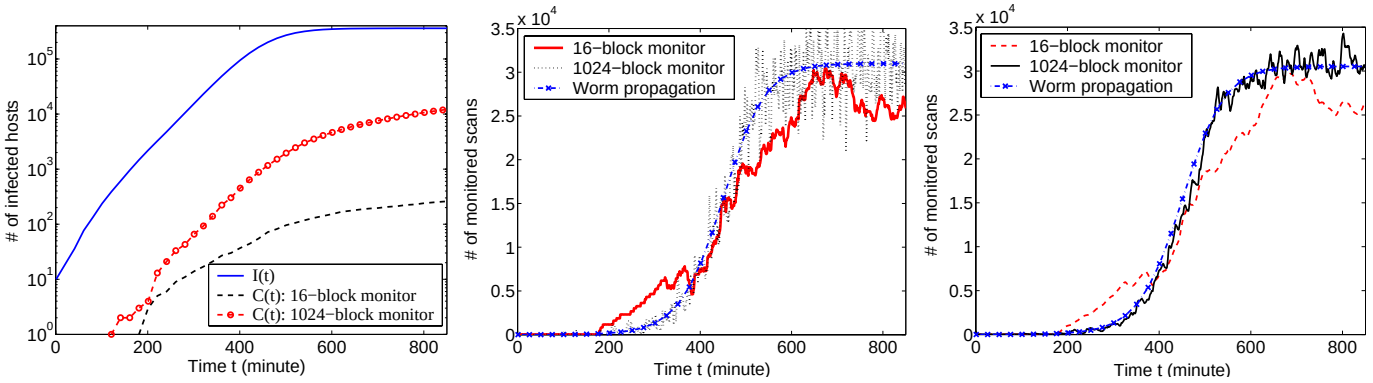


Fig. 9. Worm propagation comparison between Code Red and Blaster (100 simulation runs)

Since we select the same parameters for both Code Red and Blaster simulations, we can compare them to study how the sequential scan affects a worm's propagation. Again, we run Blaster propagation for 100 simulation runs. Fig. 9 shows the 95th percentile and 5th percentile of Blaster's propagation compared with the previous Code Red simulations shown in Fig. 5. Fig. 9 shows that Blaster still propagates according to the simple epidemic model (1), and thus the worm can be modelled by the three discrete-time models presented in this paper. This figure also shows that Blaster propagates slower than Code Red. Zou *et al.* [32] point out that this is because Blaster selects its starting scanning point



a. Worm propagation and monitored infected hosts C_t (Y-axis is in logarithm format)

b. Monitored data Z_t

c. Monitored data Z_t after using a low-pass filter

Fig. 10. Blaster propagation and its monitored data

with a local preference, not because of its sequential scan mechanism.

For monitoring of Blaster, because of the sequential scan, we cannot let the monitoring system to only cover one big block IP address space — such a monitoring system can only observe a very small fraction of infected hosts in the Internet. For example, if a sequential scan worm has the same fast scan rate $\eta = 4000$ per second as Slammer [19], one infected host will take $2^{32}/\eta = 12.4$ *days* to finish scanning the entire IPv4 space. Therefore, most hosts infected by Blaster will take days before their scans hit the big block IP address space monitored in a monitoring system.

For this reason, a good worm monitoring system should cover as distributed as possible IP address space in the Internet. In this paper, we simulate two monitoring systems. Both monitoring systems cover the same 2^{20} IP addresses (the same as the monitoring system in previous Code Red study), but they consist of different number of monitored IP blocks: one monitors 16 Class B networks; the other monitors 1024 equal-size blocks of IP space of size 2^{10}. All monitored address blocks in a monitoring system are evenly distributed in the entire IPv4 space.

Fig. 10 shows one simulation run of Blaster. Fig. 10(a) shows the number of infected hosts $I(t)$ in the Internet as a function of time t. It also shows the cumulative number of observed infected hosts, $C(t)$, from both monitoring systems. Because observed $C(t)$ is very small compared with $I(t)$, we plot this figure by taking logarithm on Y-axis.

Fig. 10(a) shows that we can observe less than 0.1% of infected hosts in the Internet from the 16-block monitoring system during the worm's propagation period. Even if we use the 1024-block monitoring system, we can only observe less than 4% of infected hosts in the Internet during the worm's propagation period. This is the reason why until now researchers have not derived an accurate estimate of how many computers were really infected by Blaster during the first several days of its break out.

Fig. 10(b) shows the monitored data $Z(t)$, the number of worm scans observed within each minute. Compared to the 16-block monitoring system, The 1024-block monitoring system gives noisier observation $Z(t)$. This is because as time goes on, an infected host will enter or leave one of the monitored IP blocks — it happens more frequently in the 1024-block monitoring system than in the 16-block monitoring system.

Although noisier than the data from the 16-block monitoring system, the monitored data from the 1024-block monitoring system represent more accurately the propagation of a sequential scan worm. From the monitored data sets, we want to know the worm propagation pattern in the global Internet, i.e., the curve of I_t shown in Fig. 10(b). Such growth pattern of I_t is a low frequency signal compared with the high frequency noise presented in the observed data Z_t. Therefore, we can use a low-pass filter to filter out high frequency noise from Z_t without changing the worm's propagation pattern. Fig. 10(c) shows the observation data Z_t

after filtered by a first-order low-pass filter [1]. This figure clearly shows that the monitored data from the 1024-block monitoring system can represent well the worm's propagation pattern in the entire Internet.

Based on the filtered monitored data Z_t from the 1024-block monitoring system as shown in Fig. 10(c), we run the Kalman filter estimation by using the transformed linear model. The estimated $\hat{\alpha}$ is shown in Fig. 11 as a function of time [2]. In this simulation run, Blaster infects 1.3% of vulnerable population at time 240 minutes, by which time the estimate $\hat{\alpha}$ has already stabilized and oscillated slightly around a positive, constant value. Hence our early detection system can detect Blaster before it infects 1.3% of vulnerable population in the Internet.

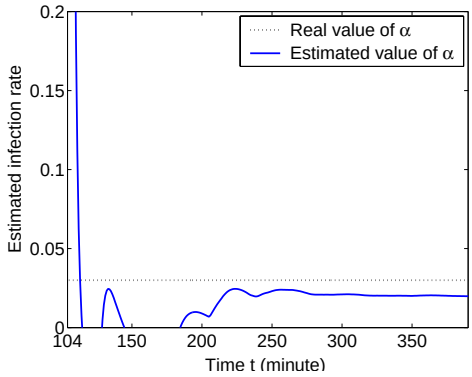


Fig. 11. Kalman filter estimation of worm infection rate α for Blaster (based on the transformed linear model and filtered data Z_t from a 1024-block monitoring system)

Worm propagation in other Blaster simulation runs give similar results as what shown in Fig. 10 and Fig. 11. On occasion the 16-block monitoring system provides as good observation as the 1024-block monitoring system. However, the 1024-block monitoring system always provides stable and good observations while the 16-block monitoring system provides poor observations in many instances.

VIII. Discussions and Future Works

We have used the simple epidemic model (1) and the exponential model (2) for the estimation and prediction. While these models give good results so far, we need to develop more detailed models to reflect a future worm's dynamics. For example, if a worm spreads through a topology, or spreads by exploiting multiple vulnerabilities, or is a meta-server worm, then its propagation may not follow the models used in this paper.

The monitoring interval Δ is an important parameter in the system design. For a slow spreading worm, it could be set to be long, but for a fast spreading worm such as Slammer, the time interval should be quite small in order to catch up with the worm's dynamics. How can we select the appropriated Δ before we know a worm's presence and its speed? We need to do further research on designing a recursive estimation algorithm that uses adaptive sampling rate. Currently, one way we think of is to tag the time stamp with each observed scans. Then at MWC, several estimators run in parallel with different monitoring intervals — from the tagged time stamp the correct C_t or Z_t for every estimators can easily be restored.

It could be useful to develop distributed estimation algorithms so as to reduce the latency and traffic for the report to a central server. We may also want to use a continuous version of the Kalman filter. This approach would reduce the significance of the monitoring interval selection and would work nicely with the distributed estimation setting.

The worm detection method presented here assumes that only worm scans can cause exponentially increased traffic to monitors, while other background scan noise cannot. We believe this is a reasonable assumption. If we want to further improve the detection accuracy, however, we can add some other rule sets in the detection system. For example, in order to distinguish a worm attack from a DDoS attack, we can exploit the differences between them: DDoS attack has one or several targets while a worm's propagation has no specific target.

The infrastructure of the monitoring system in this paper is already built up in the real world, such as the SANS's "Internet Storm Center" [15] or Symantec's enterprise early warning network [25]. However, there are still significant practical issues in setting up such a monitoring system, especially the security and privacy issues in data sharing.

[1]Denote by $\hat{Z}_t$ as the Z_t after filtering. The low-pass filter is $\hat{Z}_t = aZ_t + (1-a)\hat{Z}_{t-1}$. We use $a = 0.1$ in Fig. 10(c).

[2]Because the filtered data Z_t in Blaster scenario are noisier than the Z_t in Code Red scenario, here we choose $\tau_t = t$ in the Kalman filter (31) instead of the $\tau_t = t^2$ in the previous Code Red early detection.

IX. Conclusions

We propose a monitoring and early detection system for Internet worms to provide an accurate triggering signal for mitigation mechanisms in the early stage of a future worm. Such a system is needed in view of the propagation scale and the speed of the past worms. Although we have been lucky that the previous worms have not been very malicious, the same can not be said for the future worms. Based on the idea "detecting the *trend*, not the *burst*" of monitored illegitimate scan traffic, we present a non-threshold based "trend detection" methodology to detect the presence of a worm in its early stage by using Kalman filter and worm propagation models. Our analysis and simulation studies indicate that such a system is feasible, and the "trend detection" methodology poses many interesting research issues. We hope this paper would generate interests of discussion and participation in this topic and eventually lead to an effective monitoring and early detection system.

X. Acknowledgements

This work is supported in part by ARO contract DAAD19-01-1-0610; by DARPA under Contract DOD F30602-00-0554; by NSF under grant EIA-0080119, ANI9980552, ANI-0208116, and by Air Force Research Lab.

References

[1] B. D. O. Anderson and J. Moore. *Optimal Filtering*. Prentice Hall, 1979.

[2] V. H. Berk, R.S. Gray, and G. Bakos. Using sensor networks and data fusion for early detection of active worms. In *Proc. of the SPIE AeroSense*, 2003.

[3] Cooperative Association for Internet Data Analysis. http://www.caida.org

[4] CAIDA. Dynamic Graphs of the Nimda worm. http://www.caida.org/dynamic/analysis/security/nimda/

[5] CERT Coordination Center. http://www.cert.org

[6] Z. Chen, L. Gao, and K. Kwiat. Modeling the Spread of Active Worms, In *IEEE INFOCOM*, 2003.

[7] D. J. Daley and J. Gani. *Epidemic Modelling: An Introduction*. Cambridge University Press, 1999.

[8] Dave Goldsmith. Possible CodeRed Connection Attempts. *Incidients maillist*. http://lists.jammed.com/incidents/2001/07/0149.html

[9] eEye Digital Security. .ida "Code Red" Worm. 2001. http://www.eeye.com/html/Research/Advisories/AL20010717.html

[10] eEye Digital Security. Blaster Worm Analysis. http://www.eeye.com/html/Research/Advisories/AL20030811.html

[11] Honeynet Project. Know Your Enemy: Honeynets. http://project.honeynet.org/papers/honeynet/

[12] J. O. Kephart and S. R. White. Directed-graph Epidemiological Models of Computer Viruses. In *Proc. of IEEE Symposimum on Security and Privacy*, pages 343-359, 1991.

[13] J. O. Kephart, D. M. Chess, and S. R. White. Computers and Epidemiology. In *IEEE Spectrum*, 1993.

[14] J. O. Kephart and S. R. White. Measuring and Modeling Computer Virus Prevalence. In *Proc. of IEEE Symposimum on Security and Privacy*, 1993.

[15] Internet Storm Center. http://isc.incidents.org/

[16] L. Ljung, T. Söderström. *Theory and Practice of Recursive Identification*. MIT Press, 1983.

[17] D. Moore, C. Shannon, and J. Brown. Code-Red: a case study on the spread and victims of an Internet Worm. In *Proc. ACM/USENIX Internet Measurement Workshop*, France, November, 2002.

[18] D. Moore, C. Shannon, G. M. Voelker, and S. Savage. Internet Quarantine: Requirements for Containing Self-Propagating Code. In *IEEE INFOCOM*, 2003.

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

[20] D. Moore. Network Telescopes: Observing Small or Distant Security Events. In *USENIX Security*, 2002.

[21] D. Seeley. A tour of the worm. In *Proc. of the Winter Usenix Conference*, San Diego, CA, 1989.

[22] SANS Institute. http://www.sans.org

[23] S. Staniford, V. Paxson, and N. Weaver. How to Own the Internet in Your Spare Time. In *11th Usenix Security Symposium*, San Francisco, August, 2002.

[24] S. Staniford. Containment of Scanning Worms in Enterprise Networks. To appear in *Journal of Computer Security*.

[25] Symantec Early Warning Solutions. Symantec Corp. http://enterprisesecurity.symantec.com/SecurityServices/content.cfm?ArticleID=1522

[26] Symantec: Blaster victims top 330,000 machines. http://www.nwfusion.com/news/2003/0814blastsym.html

[27] USA Today News. The cost of Code Red: $1.2 billion. http://www.usatoday.com/tech/news/2001-08-01-code-red-costs.htm

[28] V. Yegneswaran, P. Barford, and J. Ullrich. Internet Intrusions: Global Characteristics and Prevalence. In *ACM SIGMETRICS*, June, 2003.

[29] C. C. Zou, W. Gong, and D. Towsley. Code Red Worm Propagation Modeling and Analysis. In *9th ACM Symposium on Computer and Communication Security*, Nov. 17-21, Washington DC, 2002.

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

[31] C. C. Zou, D. Towsley, W. Gong, and S. Cai. Routing Worm: A Fast, Selective Attack Worm based on IP Address Information. Univ. Massachusetts Amherst Technical Report: TR-03-CSE-06, November, 2003.

[32] C. C. Zou, D. Towsley, and W. Gong. On the Performance of Internet Worm Scanning Strategies. Univ. Massachusetts Amherst Technical Report: TR-03-CSE-07, November, 2003.

[33] C. C. Zou, W. Gong, and D. Towsley. Worm Propagation Modeling and Analysis under Dynamic Quarantine Defense. In *ACM CCS Workshop on Rapid Malcode (WORM'03)*, Oct. 27, Washington DC, 2003.