Monitoring and Early Warning for Internet Worms

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

ABSTRACT

After the Code Red incident in 2001 and the SQL Slammer in January 2003, it is clear that a simple self-propagating worm can quickly spread across the Internet, infects most vulnerable computers before people can take effective countermeasures. The fast spreading nature of worms calls for a worm monitoring and early warning system. In this paper, we propose effective algorithms for early detection of the presence of a worm and the corresponding monitoring system. Based on epidemic model and observation data from the monitoring system, by using the idea of "detecting the trend, not the rate" of monitored illegitimated scan traffic, we propose to use a *Kalman filter* to detect a worm's propagation at its early stage in real-time. In addition, we can effectively predict the overall vulnerable population size, and correct the bias in the observed number of infected hosts. Our simulation experiments for Code Red and SQL Slammer show that with observation data from a small fraction of IP addresses, we can detect the presence of a worm when it infects only 1% to 2% of the vulnerable computers on the Internet.

Categories and Subject Descriptors

K.6.5 [**Management of computing and information systems**]: Security and Protection—*Invasive software*

General Terms

Security, Algorithms

Keywords

Monitoring, Early detection, Worm propagation

1. 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, the Code Red and Nimda infected hundreds of thousands of computers [17, 22], causing millions of dollars loss to our society [8]. After a relatively quiet period, the SQL *Slammer* appeared on January 25th, 2003, and quickly spread throughout the Internet [19]. Because of its very fast scan rate, Slammer infected more than 90% of vulnerable computers on the Internet within 10 minutes [19]. In addition, the large amount of scan packets sent out by Slammer caused a global-scale denial of service attack to the Internet. Many networks across Asia, Europe, and America were effectively shut down for several hours [6].

Currently, some organizations and security companies, such as the CERT, CAIDA, and SANS Institute [3, 4, 23], 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, no nation-scale malware monitoring and defense center exists. Given the fast spreading nature of Internet worms and their heavy damage to our society, it seems appropriate 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 to detect the presence of a worm. We can directly use some well-studied methods established in the anomaly intrusion detection area. However, many threshold-based anomaly detections have the trouble to deal with their high false alarm rate. In the case of worm detection, we find that there is a major difference between a worm's propagation and a hacker's intrusion attack: the propagation of a worm code exhibits simple attack behaviors and usually follows some dynamic models because it is usually a global large-scale propagation; 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, we do not use any threshold-based anomaly detection methods in this paper. Instead, we fully exploit a worm's simple behavior based on well-studied epidemic models. We present a *Kalman filter* to detect the propagation of a worm in its early stage based on observed illegitimated scan traffic, which includes both real worm scans and background noise. The Kalman filter will not only make use of the correlation of the history trace of observation data (not just a burst of traffic at one time), but also the dynamic *trend* of the propagation of a worm — at the beginning of a worm's spreading when there are little human counteractions or network congestions, a worm propagates almost exponentially with a *constant*, *positive* infection rate.

CCS'03, October 27–30, 2003, Washington, DC, USA.

The Kalman filter is activated when the monitoring system encounters a surge of illegitimated scan activities. If the worm infection rate estimated by the Kalman filter *stabilizes* and *oscillates* a little bit around a *constant positive* value, we can claim that the illegitimated scan activities are mainly caused by a worm, even if the estimated value of the worm's infection rate is still not well converged. If the illegitimated scan 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 without a fixed central point, or it 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 *rate*, of the observed illegitimated scan traffic. In this way, the unpredictable, noisy illegitimated scan traffic we observe everyday will not cause many false alarms to our detection system — such background noise will cause great trouble to traditional threshold-based detection methods.

Our algorithms can also provide the estimated value of a worm's scan rate and its vulnerable population size. With such forecast information, people can take appropriate actions to deal with the worm. In addition, we present a formula to correct the bias in the number of infected hosts observed by monitors— this bias has been mentioned in [5] and [20], but neither of them has presented methods to correct the bias.

1.1 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 on the global Internet [20]. Yegneswaran *et al.* pointed out that there was no obvious addressing biases when using the "network telescope" monitoring methodology [26]. "Honeynet" is a network of honeypots trying to gather comprehensive information of attacks [12]. Symantec Corp. has an "enterprise early warning solution", which can collect IDS and firewall attack data from the security systems of thousands of partners to keep track of the latest attack techniques [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 [16]. It has quickly expanded to gather more than $3,000,000$ 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 also presented a threshold-based detection system called TRAFFEN.

The monitoring system we present 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 worm early detection. We also emphasize the functionality of egress monitors, which has been ignored in previous research. Worm monitors can be ingress or egress filters on routers, which can cover more IP space and gather more comprehensive information than the log data from intrusion detection sensors or firewalls.

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 [13, 14, 15]. Staniford *et al.* used the classical simple epidemic model to model the spread of Code Red right after the Code Red incident on July 19th, 2001 [24]. Their model matched well the increasing part of the observation data. Zou *et al.* presented a "two-factor" worm model that considered both the effect of human countermeasures and the effect of the congestion caused by worm scan traffic [27]. Chen *et al.* presented a discrete-time version worm model that considered the patching and cleaning effect during a worm's propagation [5].

For a very fast spreading worm such as Slammer, it is necessary to have automatic response and mitigation mechanisms. Moore *et al.* discussed the effect of Internet quarantine for containing worm propagation [18]. However, they did not present how to detect a worm in its early stage. The *CounterMalice* devices from Silicon Defense company can separate an enterprise network into cells, automatically block a worm's traffic when detecting the worm. In this way, an infected host inside a cell will not be able to infect computers in other cells of the enterprise network [9]. However, the white paper did not explain how the CounterMalice devices detect a worm at its early stage.

1.2 Discussions

In this paper, we mainly focus on worms that uniformly scan the Internet. The most widespread Internet worms, including both Code Red and Slammer, belong to this category (although the Slammer has a bad-coded random number generator, the generator has a good random initial seed. Thus "it is likely that all Internet addresses would be probed equally" [19] by Slammer). Uniform scan is the simplest and yet an efficient way for a worm to propagate when the worm has no prior knowledge of where vulnerable computers reside.

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 random IP scans. However, we believe IPv6 will not replace IPv4 in the near future, and worms will continue to use the random scan technique to spread on the Internet.

2. 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 involved in a malware attack. The simple epidemic model assumes that each host resides in one of two states: susceptible or infectious. 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 $\rightarrow$ infectious [10]. 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 [10]. At $t = 0$, I_0 hosts are infectious while the remaining $N - I_0$ hosts are susceptible.

This model could capture the mechanism of a uniform scan worm [24], especially for the initial part of a worm's

propagation when the effect of human counteractions and congestion is ignorable [27]. In this paper, we only study the initial part of worm spreading for the purpose of *early* detection. Therefore, it is suitable for us to use the simple epidemic model (1) instead of other complex models, such as the two-factor worm model in [27].

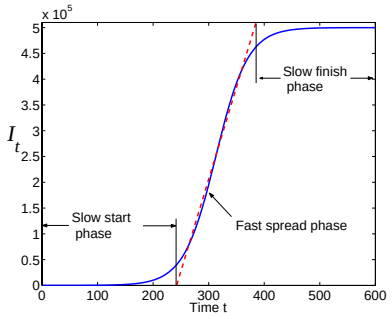


Figure 1: Worm propagation model

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.

Table 1: 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	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
y_t	Measurement data in Kalman filter
w_t	White noise in observation at time $t\Delta$
δ	Constant in equation $y_t = \delta I_t + w_t$
R	Variance of observation error
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

In this paper, we use discrete-time model for worm modelling. 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 the real time $t\Delta$. The discrete-time version of the simple epidemic model (1) can be written as [10]:

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

where $\alpha = \beta N$. We call α the *infection rate*, the average number of vulnerable hosts that can be infected per unit time by one infected host during the early stage of worm propagation.

Before we go on to discuss how to use the worm model to detect and predict worm propagation, we will first present the monitoring system design in the next Section 3 and discuss data collection issues in Section 4.

3. 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 Figure 2.

3.1 Monitoring System Architecture

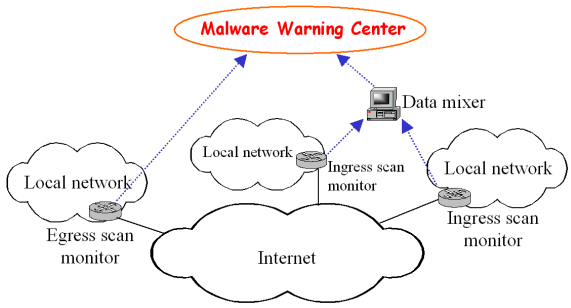


Figure 2: A generic worm monitoring system

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 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].

Berk *et al.* presented a worm monitoring system by collecting the ICMP "Destination Unreachable" messages generated by routers for packets to unused IP addresses [2]. In fact, such ICMP data are essentially the same data as the data collected by the ingress scan monitors here: when encountering packets to unused IP addresses, the routers of

local networks can either send ICMP messages to the monitoring system of [2], or send such information to the MWC of the monitoring system in this paper.

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 on 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 ingress scan 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 complete data could be obtained 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 the MWC: the 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 redundant 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.

3.2 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, e.g., non-uniform scanning, we need to use multiple address blocks with different sizes and characteristics to ensure proper coverage.

For egress scan monitors, worms on different infected computers will exhibit different behaviors. For example, the Slammer's scan rate is constrained by an infected computer's bandwidth [19]. Therefore, we need to set up multiple 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.

4. DATA COLLECTION AND BIAS CORRECTION

After setting up the 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 per monitoring interval Δ.

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 the time $t\Delta$.

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

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

(2). The worm's average scan rate η,

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

(4). The 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 the monitoring system. If the ingress scan monitors cover m IP addresses, then a worm scan has the probability $p = m/2^{32}$ to hit the monitors. We assume that in the discrete-time model all changes happen right before the discrete time epoch t, then we have

$$E[Z_t] = \eta p I_{t-1} \tag{3}$$

4.1 Correction of Biased Observation C_t

Each worm scan has a small probability p of being observed by the monitoring system, thus an infected host will send out many scans before one of them is observed by the ingress scan monitors (like a *Bernoulli trial* with a small success probability). 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 [5] and [20], but never been corrected. 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 the 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 η scans on average, thus the monitoring system has the probability $1-(1-p)^\eta$ to detect at least one scan from an infected host in a monitoring interval.

At the 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 $1-(1-p)^\eta$ 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

$$E[C_t|C_{t-1}] = C_{t-1} + (I_{t-1} - C_{t-1})[1-(1-p)^\eta]. \tag{4}$$

Removing the conditioning on C_{t-1} yields

$$E[C_t] = E[C_{t-1}] + (I_{t-1} - E[C_{t-1}])[1-(1-p)^\eta]. \tag{5}$$

Then we can derive the formula for I_t as:

$$I_t = \frac{E[C_{t+1}] - (1-p)^\eta E[C_t]}{1-(1-p)^\eta}. \tag{6}$$

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 as

$$\hat{I}_t = \frac{C_{t+1} - (1-p)^{\eta} C_t}{1-(1-p)^{\eta}}. \quad (7)$$

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

$$C_t = E[C_t] + w_t \quad (8)$$

where the statistical observation error w_t is a white noise with variance R. Substituting (8) into (7) yields

$$\hat{I}_t = I_t + \mu_t, \quad (9)$$

where the error μ_t is

$$\mu_t = \frac{w_{t+1} - (1-p)^{\eta} w_t}{1-(1-p)^{\eta}}. \quad (10)$$

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

$$Var[\mu_t] = E[\mu_t^2] = \frac{1+(1-p)^{2\eta}}{[1-(1-p)^{\eta}]^2} R \quad (11)$$

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. In addition, if the ingress scan monitors cover smaller size of IP space, p would decrease, then (11) shows that the estimate $\hat{I}_t$ would be noisier. For this reason, if we want to get an accurate estimate $\hat{I}_t$ through bias correction, the monitoring system must cover enough IP space.

We simulate Code Red propagation to check the validity of the bias correction formula (7). In the simulation, $N = 360,000$. The monitoring system covers 2^{17} IP addresses (equal to two Class B networks). The monitoring interval Δ is set to be one minute; the average worm scan rate is $\eta = 358$ per minute. Because different infected hosts have different scan rates, we assume each 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 6). The simulation result is shown in Fig. 3.

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 (7), the estimated $\hat{I}_t$ matches I_t well in the simulation before the worm enters slow finish phase ($\hat{I}_t$ deviates a little from I_t in the slow finish phase). Equation (10) shows that the estimated value should be unbiased because in deriving the bias correction formula we have assumed that all hosts have the same scan rate η, which is not the case in this simulation. In our simulation, some hosts have very small scan rate; these hosts will take much longer time to hit the monitors than other hosts. Thus in the slow finish phase, many unobserved infected hosts are hosts with very low scan rate. Therefore, the bias correction formula has some error due to the decreasing of the average scan rate for those unobserved infected hosts. In fact, we run another

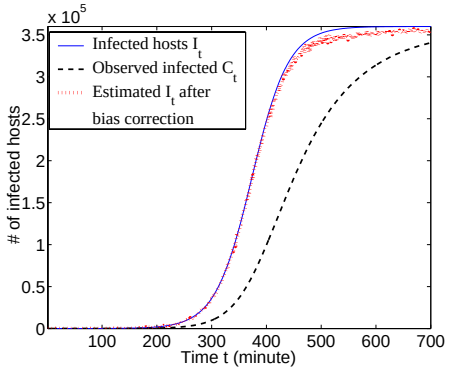


Figure 3: Estimate I_t based on the biased observation data C_t (Monitoring 2^{17} IP space)

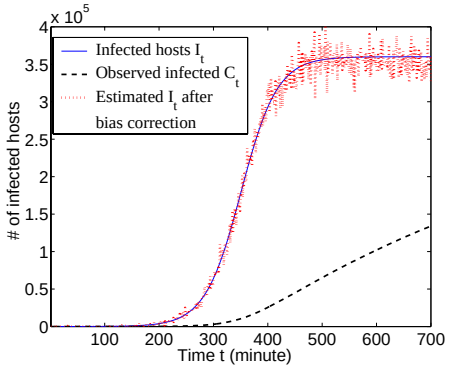


Figure 4: Estimate I_t based on the biased observation data C_t (Monitoring 2^{14} IP space)

simulation by letting all hosts having the same scan rate η (i.e., let $\sigma = 0$); then the $\hat{I}_t$ after bias correction matches well with I_t for the whole period of a worm's propagation.

Fig. 4 shows the simulation results if the monitors only cover 2^{14} IP addresses. The estimate $\hat{I}_t$ after the bias correction is very noisy because of the error amplification effect described by (11).

5. EARLY WARNING 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 the MWC, we recursively estimate the parameters β, N, and α based on observation data at each monitoring interval (these three parameters have the relationship $\alpha = \beta N$). In the following, we will first provide a Kalman filter type algorithm to estimate parameters α and β.

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

$$y_t = \delta I_{t-1} + w_t \quad (12)$$

where w_t is the observation error. δ is a constant ratio: if we use Z_t as y_t, then $\delta = \eta p$ as shown in (3); if we use $\hat{I}_{t-1}$ derived from C_t by the bias correction (7), then $\delta = 1$.

5.1 Estimation Based on Kalman Filter

From (12), we have

$$I_{t-1} = y_t/\delta - w_t/\delta. \tag{13}$$

Substituting (13) into the worm model (2) yields an equation describing the relationship between y_t and the worm's parameters:

$$y_t = (1+\alpha)y_{t-1} - \frac{\beta}{\delta}y_{t-1}^2 + \nu_t, \tag{14}$$

where the noise ν_t is

$$\nu_t = w_t - (1+\alpha)w_{t-1} - \beta(w_{t-1}^2 - 2y_{t-1}w_{t-1})/\delta. \tag{15}$$

A recursive least square algorithm for α and β can be cast into a standard Kalman filter format [1]. 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 vector as $X_t = \begin{bmatrix} 1+\alpha \\ -\beta/\delta \end{bmatrix}$. If we denote $H_t = [\ y_{t-1} \quad y_{t-1}^2\ ]$, then the system is described by

$$\begin{cases} X_t &= X_{t-1} \\ y_t &= H_t X_t + \nu_t \end{cases} \tag{16}$$

The Kalman filter to estimate the system state X_t is

$$\begin{cases} H_t &= [\ y_{t-1} \quad y_{t-1}^2\ ] \\ K_t &= P_{t-1}H_t^\tau/(H_tP_{t-1}H_t^\tau + R_\nu) \\ P_t &= (I - K_tH_t)P_{t-1} \\ \hat{X}_t &= \hat{X}_{t-1} + K_t(y_t - H_t\hat{X}_{t-1}) \end{cases} \tag{17}$$

where R_ν is the variance of noise ν_t and can be set to 1. From the experiments, we find that the value of R_ν is not important.

ν_t in (15) is a correlated noise. The Kalman filter (17) can be extended to consider such correlated noise to derive unbiased estimates of α and β in theory. However, if we use the unbiased filter, we will have more parameters to estimate, then the new filter will converge slower than the proposed filter (17). Our experiments also confirm this conjecture. In this paper the primary objective is to derive the rough estimate of α as quickly as possible. Therefore, it is better to use the simple Kalman filter (17) for worm early detection.

If we use Z_t as the measurement 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 (17) does not depend on δ in estimating α; the value of δ only affects the estimated value of β.

5.2 Estimation of the Vulnerable Population

The parameter β in model (2) is on the order of $1/N$. Thus in the Kalman filter above, the two elements in the state X_t differ in the order of N. For this reason, the Kalman filter performs poorly in estimating the small value β. Consequently, the estimation of N by using $\hat{N} = \hat{\alpha}/\hat{\beta}$ is not good.

We present an effective way to estimate the population N based on η and the estimate α from the Kalman filter above. A uniformly scanning worm sends out on average η scans per monitoring interval; 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 population are vulnerable, a worm can infect, on average, $\eta N/2^{32}$ hosts per monitoring interval (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

$$N = \frac{2^{32}\alpha}{\eta} \tag{18}$$

where the average worm scan rate η can be obtained directly from egress scan monitors in the monitoring system. We can use this equation to estimate N along with the Kalman filter in estimating α. In this way, the estimation of N can have similar convergence properties to that of the estimation of α from the Kalman filter.

5.3 Overview of the Steps to Detect a Worm

The MWC collects and aggregates reports of worm scans from all distributed monitors in real-time. For each TCP or UDP port, the MWC has an alarm threshold for monitored illegitimated scan traffic Z_t. The observed number of scans Z_t, which contains non-worm noises, is below this threshold at most time when there is no global spreading worm. If the 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 the MWC will begin to record C_t and calculate 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, the 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 the worm at the time $t\Delta$ ($t = 1, 2, 3, \cdots$). For accuracy, the MWC can also use two filters based on C_t and Z_t respectively in order to cross check to verify the results.

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 $\hat{\alpha}$ does not stabilize in a long time, or it stabilizes and oscillates around zero, we believe the surge of illegitimated traffic is caused by non-worm noise.

6. SIMULATION EXPERIMENTS

6.1 Simulation Settings

We simulate both Code Red on July 19th, 2001 [7] and the SQL Slammer on January 25th, 2003 [19]. 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.* [24] used a different format but the same epidemic model as (1) to model Code Red, where their model's parameter K has $K = \beta N = \alpha$ [27]. 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 (18) we can reversely derive $\eta = 2^{32}\alpha/N = 358$ per minute, i.e., Code Red sent out on average about $358/60 = 5.965$ scans per second.

Because different infected hosts have different scan rates, we assume that each 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, the ingress scan monitors cover 2^{20} IP space. We also assume $I_0 = 10$ at the beginning.

SQL Slammer propagated in the same way as Code Red by randomly generating target IP addresses to scan [19]. According to [19], a Slammer infected host sent out on average 4,000 scans per second at the beginning. The authors in [19] also observed that 75,000 hosts were infected in the first 30 minutes. Thus in the case of Slammer simulation, we set $\eta = 4000$ and $N = 100,000$ (since many infected hosts did not scan the monitors in the first 30 minutes due to the congestions caused by Slammer). Because the scan rate of Slammer was bandwidth limited and varied a lot among different computers, in our simulation the scan rate x of a host is predetermined by the normal distribution $N(4000, 2000^2)$ (x is bounded by $x \geq 1$).

In the discrete-time simulation, the monitoring interval Δ is set to be one minute for Code Red, and one second for the very fast SQL Slammer worm.

6.2 Background Noise Consideration

We need to consider background non-worm noise in our simulations. Fortunately, Goldsmith provided simple data of the background noise for Code Red activities monitored on a Class B network (covers 2^{16} IP addresses) [11]. Goldsmith recorded TCP port 80 SYN requests from Internet hosts to any unused IP addresses inside his local network — such method is the same as the ingress scan monitors in our proposed monitoring system. The observation 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 noise sources 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 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 the detailed log data on Code Red and Slammer from others in order to have more realistic experiments.

For the SQL Slammer, we do not have any observed data on its background noise. Thus we simply assume that it has the same background noise as the Code Red.

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. The Kalman filter will be activated when the illegitimated 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 noise traffic.

6.3 Code Red Simulation and Early Warning

For Code Red, Fig. 5(a) shows I_t in *one* simulation run as a function of time; Fig. 5(b) shows the estimate of infection rate $\hat{\alpha}$ as time goes on by using observation 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 128 minutes. Fig. 5(c) shows the estimate $\hat{\alpha}$ by using C_t after bias correction (7). Both estimates converge to the real value of α, but the estimate by using C_t is smoother. Our objective of the estimation is to find whether the estimate stabilizes and oscillates slightly around a positive constant value. Therefore in the following, we will always use C_t after bias correction to estimate α with the Kalman filter if not mentioned explicitly.

We estimate the vulnerable population size N by Equation (18) at each discrete time. Fig. 6 shows the estimated value of N as time goes on. For comparison, we also use the estimated parameter $\hat{\beta}$ derived from the Kalman filter (17) to directly calculate $\hat{N}$ by $\hat{N} = \hat{\alpha}/\hat{\beta}$. The result is also shown in Fig. 6. This figure shows that Equation (18) can provide a more accurate estimate of N than the direct estimation from the Kalman filter.

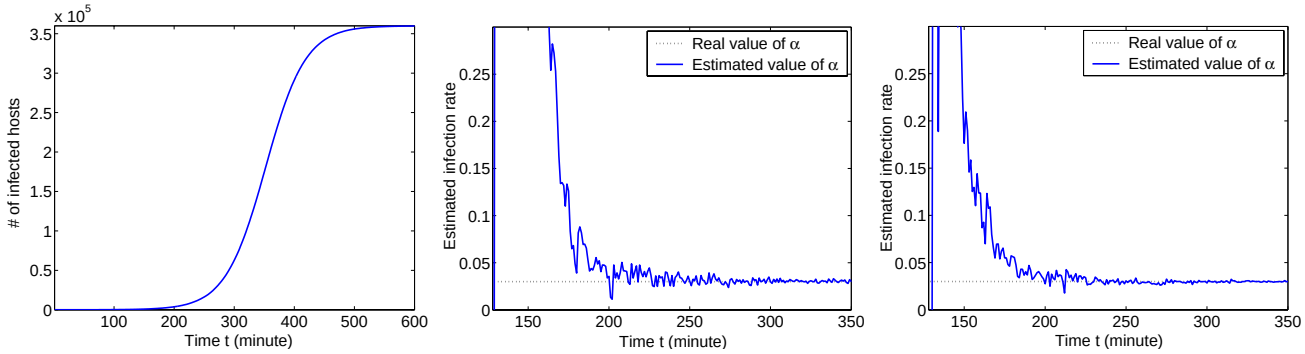
In this simulation run, Code Red infects 1% of the vulnerable population at 199 minutes, and 2% of population at 223 minutes. Fig. 5(c) shows that during the time when Code Red infects 1% to 2% population, the estimate of the worm's infection rate α has already stabilized to show the constant positive property (though the estimated value is still very rough). Therefore, the MWC can detect the presence of the worm when it infects about 1% to 2% of vulnerable population (the estimation of N is rougher but it is less important than the worm infection rate in order to detect a worm).

6.3.1 *Variability in Worm Propagation*

Worm propagation is in fact a stochastic process. In order to check if the Kalman filter detection algorithm works well under most cases, we use the same parameter settings in the previous simulation to run the Code Red simulation for 100 times. Fig. 7 shows the upper and lower bounds and the average value of the number of infected hosts in these 100 simulation runs.

For each of these 100 simulation runs, we use the Kalman filter to estimate the worm's infection rate α. Among these 100 runs, the worm infects 2% of the vulnerable population at the time between 200 to 258 minutes. During the estimation process of each simulation run, the estimated value of α will gradually decrease its oscillation and converge to its true value (as shown in Fig. 5(c)). For each simulation run, we obtain the maximum and minimum values of estimated α after the worm infects 2% population — the oscillation of the estimated α will not exceed this boundary after the worm infects 2% population. This boundary tells us how well we can obtain a stabilized estimation. The oscillation bounds are shown in Fig. 8 for these 100 simulation runs.

In order to check if the Kalman filter has worse performance when the worm propagates faster, in Fig. 8 we have sorted the 100 simulation runs according to the time when the worm infects 2% population. In other words, the worm in simulation i in Fig. 8 infects 2% of vulnerable population earlier than the worm in simulation j if $i < j$. This figure shows that the performance of the Kalman filter is not affected by the variability of the spreading speed of the worm. One reason is that the Kalman filter is activated earlier when the worm spreads faster. Another reason is that when the worm spreads faster, the signal-to-noise ratio of the observed data will become higher, thus the estimation will converge faster.



a. Code Red propagation b. use observed data Z_t c. use C_t after bias correction

Figure 5: Code Red simulation and Kalman filter estimation of infection rate (for one simulation run)

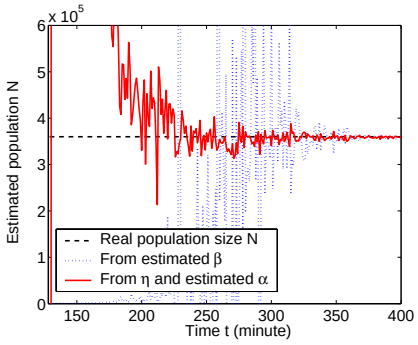


Figure 6: Estimate of population N for Code Red

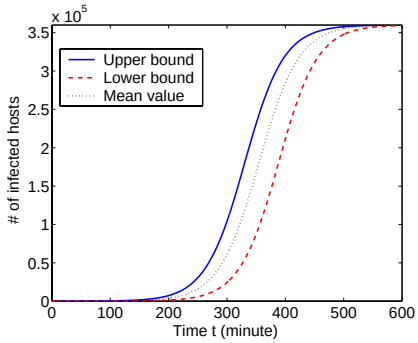


Figure 7: Variability in Code Red simulation (for 100 simulation runs)

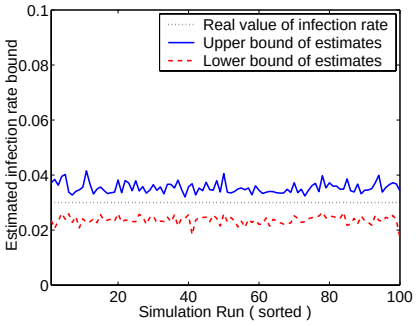


Figure 8: Bound of $\hat{\alpha}$ after worm infects 1% population (for 100 simulation runs)

6.3.2 *Early Warning for a "Hit-list" Worm*

"Hit-list" concept was first presented in [24]. The worm has a built-in list that contains thousands IP addresses of potentially vulnerable machines. The worm will first propagate by only scanning computers on this list. After infecting most hosts on the list, the worm uses random scans to try to infect other vulnerable hosts on the Internet.

At the hit-list scanning phase, the worm does not send probes to nonexistent IP addresses. Therefore, we assume that the monitoring system could collect worm scans only after the worm finishes the hit-list scanning phase. In our simulation, we assume that at time 0 the worm has already infected all hosts on the hit-list, i.e., it has a large number of initially infected hosts. Because of the worm's high scan traffic when it begins to randomly scan, the Kalman filter will be activated immediately at the beginning.

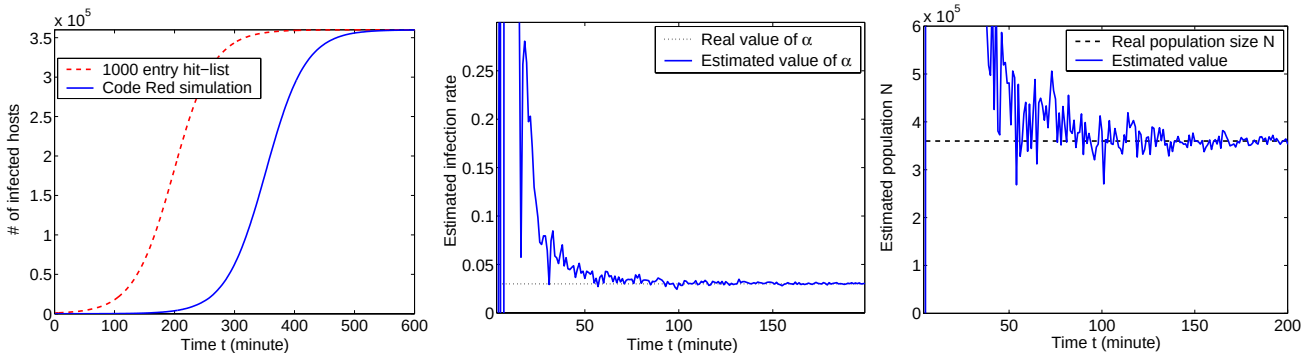
Fig. 9(a) shows the propagation of a hit-list worm that has a 1000-entry hit-list compared with previous Code Red propagation (they have the same simulation settings except the number of initially infected hosts). From Fig. 9(a) we observe that the hit-list does not affect the worm propagation pattern. Fig. 9(b)(c) show the estimation of the worm infection rate α and the vulnerable population N, respectively. Compared to Fig. 5(c), Fig. 9(b) shows that the Kalman filter estimation of the infection rate converges to the true value faster than the estimation for a worm without hit-list. It is because the worm with hit-list sends out larger amount of scans; the signal-to-noise ratio of the observation data in this experiment is higher than what used in the original Code Red experiment.

In this simulation run, the hit-list worm infects 1% and 2% of vulnerable population at time 45 and 69 minutes, respectively. Fig. 9(b) shows that the estimate has already stabilized with only small oscillation by 69 minutes. Therefore, the WMC can still detect the presence of the hit-list worm when it infects only 1% to 2% of vulnerable population.

6.4 Slammer Simulation and Early Warning

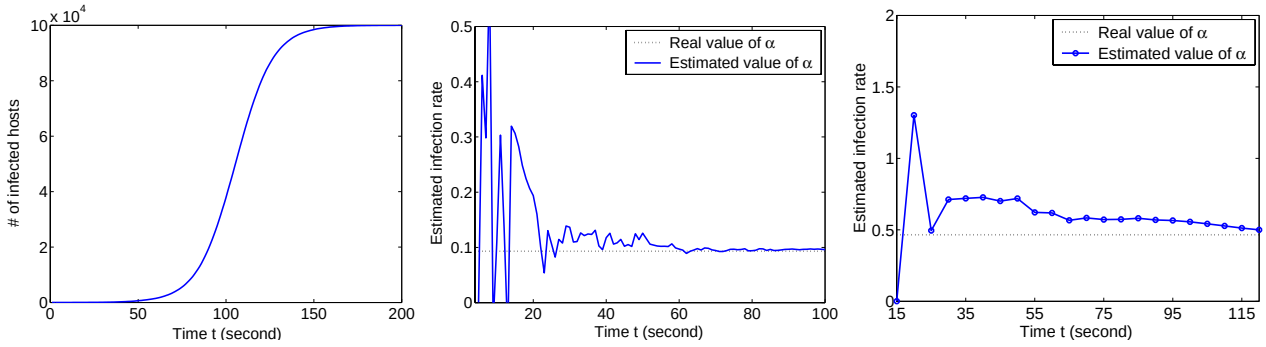
For SQL Slammer, Fig. 10 shows the results for one simulation run. Fig. 10(a) shows that without congestion and human counteractions, the SQL Slammer can infect most of the vulnerable hosts within 3 minutes. In reality, the Slammer quickly reduced its propagation speed because its huge scan traffic caused congestion or even broke down some networks. Therefore, it took about 10 minutes to infect 90% of vulnerable computers [19] instead of the 3 minutes here. However, at the beginning when there were few congestions and human counteractions, the Slammer still propagated according to the epidemic model used here (see Fig. 3 in [19]).

From Fig. 10(a), we know that the worm infects 1% vulnerable population at time 45 seconds. Fig. 10(b) shows the estimated value of the infection rate α as time goes on. It shows that when the worm infects 1% population, the estimated value of α is already stabilized with small oscillation for a while (though the oscillation central point is higher



a. Hit-list worm propagation b. Kalman filter estimation of α c. Estimation of population size N

Figure 9: Hit-list worm simulation and Kalman filter estimation (for one simulation run)



a. SQL Slammer propagation b. Kalman filter estimation of α (time unit: 1 second) c. Kalman filter estimation of α (time unit: 5 seconds)

Figure 10: SQL Slammer simulation and Kalman filter estimation (for one simulation run)

than the true value at this moment). Thus the MWC can detect the worm when it infects about 1% of vulnerable population.

Fig. 10(b) shows that at the beginning of estimation, the Kalman filter overestimates the value of α. This is because the discrete-time model (2) is a first-order discretization of the continuous model (1). The discretization introduces error in the infection rate α when the worm keeps exponentially increasing at the beginning.

Fig. 10(c) shows the simulation results when we use 5 seconds as the monitoring interval Δ. Because now the discrete time unit is larger, the discretization error shows up clearly in this figure — the estimated value $\hat{\alpha}$ is higher than its true value.

7. DISCUSSIONS AND FUTURE WORKS

We have used the epidemic model for the estimation and prediction. While this gives 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 the dynamics will not always follow the epidemic model.

For non-uniformly scanning worm, such as the Code Red II, we might have different observations by setting up the monitors on different places. The non-uniform scanning behavior of a worm may also affect the bias correction (7). For a future unknown worm, through analysis of the worm scan distribution by using the data from egress scan monitors, the MWC can determine if the worm is uniformly scanning the Internet or not. If it is not, the MWC can use data Z_t, or directly use data C_t without bias correction, to detect and predict the worm.

The monitoring interval Δ is an important parameter in the system design. For slow spreading worm, it could be set to be long, but for fast spreading worm such as the Slammer, the time interval should be quite small in order to catch up with the worm's speed. How can we select the appropriated Δ before we know the 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 the 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 exponential increasing scan 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" [16] or Symantec's enterprise early warning network [25]. However, there are still significant practical issues in setting up such monitoring system, especially the security and privacy issues in data sharing.

For a fast spreading worm such as the SQL Slammer, human's manual actions will not catch up with its speed even if the early warning is provided at the beginning of the worm's spreading. Automatic mitigation is the only way to defend against such kind of worm attack. How to decrease false alarm rate, detect a worm earlier, and collect observation data in time are the key factors in incorporating the early warning system with automatic mitigation.

8. CONCLUSIONS

We propose a monitoring and early warning system for Internet worms to provide an accurate triggering signal for mitigation mechanisms in the early stage of a future worm. Such system is needed in view of the propagation scale and 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 rate" of monitored illegitimated scan traffic, we present a Kalman filter to detect the presence of a worm in its early stage. The 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 warning system.

9. 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.

10. 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] CERT Coordination Center. http://www.cert.org

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

[6] CNN News. Computer worm grounds flights, blocks ATMs. http://europe.cnn.com/2003/TECH/internet/01/25/internet.attack/

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

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

[9] CounterMalice: military-grade worm containment. http://www.silicondefense.com/products/countermalice/

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

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

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

[13] 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.

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

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

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

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

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

[24] 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.

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

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

[27] C.C. Zou, W. Gong, and D. Towsley. Code Red Worm Propagation Modeling and Analysis. In *9th ACM Symposium on Computer and Communication Security*, pages 138-147, Washington DC, 2002.