

Rapid Detection of Worms Using ICMP-T3 Analysis

Robert S. Gray and Vincent H. Berk

Institute for Security Technology Studies, Thayer School of Engineering
Dartmouth College, Hanover, NH 03755
FirstName.LastName@Dartmouth.edu

ABSTRACT

Identification of an active Internet worm is a manual process where security analysts must observe and analyze unusual activity on multiple firewalls, intrusion-detection systems or hosts. A worm might not be positively identified until it already has spread to most of the Internet, eliminating many defensive options. In previous work, we developed an automated system that can identify active worms seconds or minutes after they first begin to spread, a necessary precursor to halting the spread of the worm rather than simply cleaning up afterward. The system collects ICMP Destination Unreachable messages from instrumented network routers, identifies those patterns of unreachable messages that indicate malicious scanning activity, and then searches for patterns of scanning activity that indicate a propagating worm. In this paper, we compare the performance of two different detection strategies, our previous threshold approach and a new line-fit approach, for different worm-propagation techniques, noise environments, and system parameters. These techniques work for worms that generate at least some of their target addresses through a random process, a feature of most recent worms. Although both being powerful methods for fast worm identification, the new line-fit approach proves to be significantly more noise resistant.

1. INTRODUCTION

In previous work,^{1,2} we developed an automated system that can identify active worms seconds or minutes after they first begin to spread. This system, which combines our DIB:S scan-identification system and TRACKing And Fusion Engine (TRAFEN),¹ can detect worms that generate at least some of their target addresses through a random process.

The DIB:S system collects ICMP Destination Unreachable messages forwarded from participating Internet routers, identifies individual hosts (IP addresses) that show one of several types of scanning behavior, and notifies the TRACKing and Fusion ENGINE (or TRAFEN) of these hosts. TRAFEN is a Process Query System (PQS), a system designed to support model-based queries against arbitrary sensor feeds. In our case, the sensor feed are the DIB:S alerts, and the model-based queries look for increases in scanning activity indicative of an epidemic. The models can base their decision on the absolute number of hosts showing the same scanning behavior within a defined time window (threshold approach) but can obtain better results by looking at the rate of increase, matching the observed activity with either a straight-line or exponential prediction (best line-fit approach).

The main tradeoffs are the risk of false positives, the computational complexity of the detection algorithm, and the speed with which the detection can take place (i.e., the number of machines infected at detection time). Although we showed in earlier writing that a primary way of reducing false positives is to acquire more DIB:S router feeds, our previous threshold-based approach is resistant only to low noise levels. Although the noise observed on our deployed routers is low enough to successfully use the threshold approach in a real detection application, we seek a more noise-tolerant approach.

In this paper, we compare our previous threshold-based approach with a new line-fit approach. Both approaches are implemented as detection models provided as input to TRAFEN. The basis for the threshold model is the time between the arrival of two DIB:S alerts with exactly the same properties (e.g., both alerts are for scans against TCP port 80). The smaller the time difference becomes, the larger the likelihood that the alerts are evidence of a propagating worm. If the likelihood passes a threshold, we assume a worm alert. The second method is a line-fit approach where we group the DIB:S alerts into time buckets, each bucket containing the count of alerts falling within the associated time period, and then compare the slope of this curve with a threshold

slope. If the slope passes this threshold, and if the curve satisfies a set of self-similarity properties, we assume a worm epidemic. In this paper, we test both of these TRAFEN detection models for different worm-propagation techniques and rates, varying noise conditions, and a wide range of system parameters. The primary focus of the experiments is to evaluate if the new approach detects worms with equal accuracy, while avoiding false positives, under more intense noise conditions.

In Section 2, we give a brief overview of the workings of DIB:S and the ICMP Destination Unreachable messages, and briefly describe how TRAFEN works. In Section 3, we present our simulation framework. In Section 4, we examine the performance of the two detection models under varying conditions. Finally, in Sections 5 and 6, we review related work, and consider promising avenues of future work.

2. BACKGROUND

The Dartmouth ICMP Bcc: System (DIB:S) uses ICMP Destination Unreachable (Type 3 – ICMP-T3) messages to identify IP addresses that show unusual scanning behavior. Routers generate ICMP-T3 messages when they are unable to deliver a packet. The cause of the delivery failure can vary. For example, the destination IP address might not be assigned, or the destination network might be unused. The router returns the ICMP-T3 message to the sender, and includes in the message the header of the packet that could not be delivered, allowing the sender to determine which packet failed delivery. Since the embedded portion of the original packet includes the the IP header plus *at least* eight bytes of the payload (which typically would be the beginning of protocol headers), the ICMP-T3 message contains the sender's IP address, the unreachable destination's IP address, and the protocol. In addition, for TCP and UDP packets, the message includes the source and destination ports.

Routers participating in the DIB:S system send copies of the ICMP-T3 messages that *they themselves* generate to the DIB:S central collector, optionally dropping the original ICMP-T3 message so as not to reveal network information to the sender. The DIB:S collector aggregates these messages and processes the embedded headers of the original, failed packets, sorting the headers by source and destination IP address. Messages age out after Δt seconds, usually configured on the order of 300 to 1800 seconds. If one port/protocol combination appears N times within Δt seconds for a specific IP address, an alert is generated, where N usually is configured between 4 and 20. The alert takes the following form: *this IP address contacted (or was contacted by) at least N other IP addresses using protocol P on port p in the last Δt seconds*. After an alert is generated, DIB:S will not generate another alert for the same port/protocol combination until at least Δt seconds have elapsed, preventing a single heavily scanning host from producing a continuous series of alerts. DIB:S can generate several other types of alerts as well,³ although we do not use these other types for worm detection.

During an active worm epidemic, infected hosts will scan IP addresses to determine if they are vulnerable to an exploit. If some or all of these IP addresses are generated randomly, as in most recent worms, chances are high that some of the addresses are not populated and/or are in unassigned address space. Routers invariably will generate a large volume of ICMP-T3 messages for return to the infected hosts*, and the number of messages generated per unit time will increase in proportion to the current number of infected hosts. Since some of the routers will be participating DIB:S routers, DIB:S, through central collection and aggregation of the triggered ICMP messages, will quickly identify infected IP addresses and report these addresses to TRAFEN.

TRAFEN is an an implementation of a Process Query System. Process Query Systems try to find evidence of stochastic processes in observation streams. In our case, the observations are the DIB:S alerts, and the detection accuracy of the combined DIB:S/TRAFEN system is dependent on the quality of the stochastic processes for which TRAFEN searches. These stochastic processes are referred to as "Process Queries" or "Models". TRAFEN takes as input one or more observation (or alert) streams and process queries. The observations can be in an arbitrary format, as long as the process queries are able to understand them. TRAFEN will accept high-level model descriptions (such as a specification of Hidden Markov Model) as process queries. As the TRAFEN services evolve, however, we currently implement the worm-detection models as Java classes conforming to TRAFEN's modeling interface.

*Routers limit the generation of ICMP messages, usually to three or four per second. We account for this limit in our simulations.

The central abstraction of TRAFEN is Multiple Hypothesis Tracking.⁴ TRAFEN maintains multiple hypotheses, each representing a different view of the “outside world”. Within each hypothesis, TRAFEN forms tracks of observations, and uses the process queries to score the tracks, each query calculating the likelihood that the track is evidence of its stochastic process. Each hypothesis is assigned an overall likelihood based on the likelihoods of the tracks. When a new observation arrives, and TRAFEN is given multiple process queries, it either will run multiple instances of Reid’s algorithm or will associate a vector of likelihoods, one for each model, with each track. The choice depends on the application domain and is at the user’s discretion.

When a new observation arrives, TRAFEN must decide to which tracks it should be added. Each observation may appear only once in each hypothesis, and thus TRAFEN generates a set of new hypotheses for every current hypothesis, adding the observation to each track in turn and using the process models to calculate the resulting likelihoods. TRAFEN also constructs a new hypothesis in which the observation forms a singleton track. This process explodes the number of hypothesis exponentially, and therefore TRAFEN prunes the hypothesis set, keeping those hypotheses with higher overall likelihoods. More details about PQS can be found in our PQS overview paper.⁵ For our purposes, TRAFEN is a data-collection and fusion system in which new worm-detection models can be implemented quickly and reliably.

3. SIMULATION

Initial evaluation of DIB:S/TRAFEN or any worm-detection system requires realistic simulation of propagating worms and their induced network effects, since attackers release real worms on the Internet too infrequently to provide meaningful coverage of the possible parameter space. ISTS researchers previously developed an add-on package to SSFNet[†], that can efficiently simulate an Internet worm event that lasts days or longer, achieving its efficiency by separating the simulation task into macro (Internet autonomous systems) and micro (subnetworks) levels.² A lighter-weight simulation can be used for worms that infect machines within hours or minutes, however, and we developed such a simulation for this project. In contrast to the SSFNet-based simulation, this second simulation does not simulate specific network devices or subnetworks, but instead just the ICMP T-3 traffic induced by a propagating worm.

The simulation allows a worm to propagate through up to the entire IPv4 address space, and selects the desired number of instrumented Class B networks (2^{16} addresses) within that address space. Each Class B network is assumed to have a single instrumented router that covers the *complete* network. An instrumented router, subject to the ICMP rate limit, will generate an ICMP T-3 message whenever a worm instance outside of the associated Class B network attempts to contact an unreachable address within the Class B network. The router will *not* generate an ICMP T-3 message if a worm instance *inside* the associated Class B network attempts to contact an unreachable address. The simulation distributes vulnerable machines throughout the selected address space, possibly including the instrumented Class B networks, marks a certain percentage of the remaining addresses as externally unreachable machines, and starts the worm on an arbitrary, externally reachable address (separate from the previously identified vulnerable machines). As the worm scans for vulnerable hosts, the simulation generates appropriate ICMP T-3 messages. Judicious use of hash tables make the process of checking whether an address is vulnerable, unreachable, and/or behind an instrumented router quite fast.

The simulation assumes that the worm is spreading on an arbitrary port, which we will refer to as the *worm port*, and can generate three types of Poisson noise. First, a pool of source addresses can attempt to contact non-worm ports of unreachable addresses at exponentially-distributed intervals, where where contact attempt involves a single probe packet. Second, a pool of source address can attempt to contact the worm port of unreachable addresses at at exponentially-distributed intervals, where each contact attempt again involves a single probe packet. Finally, a pool of source addresses can attempt to *scan* the worm port of unreachable addresses at exponentially-distributed intervals, where each scan consists of multiple probes to the worm port on different unreachable addresses within a short period of time. The number of probes per scan follows a normal distribution, while the time between individual probes in a scan follows an exponential distribution.

[†]<http://www.ssfnet.org>

The simulation has several parameters that are of specific interest. First, the *Exclude List* specifies the portion of the IPv4 address space in which there will be no vulnerable machines or instrumented routers. Further, every simulated worm ignores addresses that are on the exclude list. In our simulations, we exclude all multicast addresses and all Class A networks (2^{24} addresses) that the Internet Assigned Numbers Authority (IANA) still marked as *Reserved* as of January 15, 2004. [‡]

The *ICMP Rate Limit* is the number of ICMP messages that each router can generate per second. The configured rate limit for most real routers is 3 or 4, and we use 4 in our simulations. The *Unreachable Percentage* is the percentage of addresses that are externally unreachable. In most of our simulations, we use a value of 50%, a conservative choice given that we observed only 10% of Internet addresses to be reachable in a 2001 experiment.¹ By choosing the significantly higher value of 50%, we ensure that we will not generate more simulated ICMP T-3 messages than we could expect on the real Internet.

The *Vulnerability Distribution* is either *global* or *local*. In a global distribution, vulnerable hosts either are distributed uniformly throughout the entire non-excluded address space, while in the local case, vulnerable hosts can appear only within a specific number of Class B networks. In the latter case, we use a strictly uniform distribution, simply dividing the vulnerable hosts as evenly as possible among the desired number of Class B networks. The *Vulnerable Size* is the number of vulnerable hosts. The worm's *Scan Strategy* is either *overlap* or *divide-and-conquer*. In the overlap case, each copy of the worm scans the entire non-excluded portion of the address space, while in the divide-and-conquer case, each worm copy divides its current scan space evenly with each new worm copy that it installs on a vulnerable host. These two types of scan strategies represent extremes for our detection system, since among worms that use random target generation, overlap worms will cause the instrumented routers to see the most *unique* scanning source addresses, while divide and conquer worms will cause the instrumented routers to see the least.

The *Flash Size* are the number of vulnerable machines and the number of those machines in the worm's flash list, where the flash list is a set of previously identified vulnerable machines that the worm can scan and infect immediately before switching to random probing. The *Scan Rate* and *Local Scan Probability* are the time between each worm's probes and the probability that the worm will generate and probe an address within its local Class B network (rather than within the overall address space). Our simulation does not consider the fact that the scan rate might differ depending on whether the target machine is unreachable versus reachable or local versus external, nor does it consider the time needed to transmit the worm to a target machine. These factors generally do not affect the number of ICMP messages induced by a particular worm. Similarly, the simulator does not consider bandwidth effects, since we seek to detect worms in their early stages, well before available network bandwidth becomes a constraint on their propagation.

Finally, a set of *Noise Parameters* governs the behavior of the noise modules. There are three mean inter-arrival times, one for each of background, port and scan noise, and two source-pool sizes, one for each of port and scan noise. We eliminate the source-pool size for the background noise, since the identity of background-noise sources is not used in any of our detection algorithms. Background noise is assumed to come from any arbitrary external address. Finally, for scan noise only, we specify the average number of probes per scan, setting the standard deviation of the Normal distribution to the square root of the mean, and also specify the average delay between probes.

4. PERFORMANCE

Detection performance intuitively will depend on the number of ICMP messages observed during worm propagation, as well as on the number of distinct source addresses that induced those messages. The former determines the amount of raw data available to the DIB:S/TRAFEN system, while the latter determines the maximum number of alerts that DIB:S can produce within its history window. In this section, we first examine the number of ICMP messages and DIB:S alerts induced by different types of worms, and then consider detection performance both with and without noise. Our results generalize to many different data sources besides ICMP messages, since DIB:S/TRAFEN only needs to know when one machine has tried to contact an unreachable machine. Such

[‡]<http://www.iana.org/ipaddress/ip-addresses.htm>

information can come from many other network devices beside routers, such as firewalls, intrusion-detection systems, or custom ingress/egress filters. Zou et al., for example, uses ingress/egress filters for their worm-detection system.⁶

Unless otherwise specified, we will use the following values for the simulation runs described in this section: *Unreachable Percentage*: 50%, *ICMP Rate Limit*: 4, *Flash Size*: 0, *Vulnerable Size*: 200,000, *Vulnerable Hosts per Class B (when in local mode)*: 200, *Scan Rate*: 205 milliseconds between scans, *Instrumented Class B networks*: 16, and *Local Address Probability (when in local mode)*: 10%. The vulnerable size of 200,000 and the scan rate of 205 milliseconds are representative of recent worms, while 16 instrumented Class B networks produced effective detection results in our earlier work.¹ The values of 200 vulnerable hosts per Class B and a local address probability of 10% were chosen simply as representative values. In many cases, we will vary one of the parameters while holding the rest fixed at the indicated values. For example, Figure 1a uses the parameters above, but varies the number of instrumented networks from four to twenty. For each set of parameters, we generally will consider both overlap and divide and conquer worms for both global and local vulnerability distributions. When the vulnerability distribution is local, the worm always uses the local address probability to skew its address selection toward local addresses. If the worm does *not* skew its address selection, global and local vulnerability distributions induce the same propagation behavior, and so we do not include non-local, but non-skewed, as a separate case.

4.1. ICMPs and Alerts Per Worm

Figure 1 shows, as a function of the number of instrumented Class B networks (a) and the scan rate of the worm (b), the average number of ICMP messages generated per worm over the first one thousand infected machines. Figure 1 shows that the number of induced ICMP messages is significantly lower in the local-preference case, simply because the worm is able to infect the one thousand machines while sending fewer probes through instrumented routers. The number of ICMP messages increases as a function of the number of instrumented Class B networks, since more instrumented routers will see more probe attempts. Finally, the average number of ICMP messages is constant across different scan rates, except for the fastest scan rates where the ICMP rate limit prevents the generation of all possible unreachable messages. This rate-limit effect is particularly pronounced for divide and conquer worms, since probing a smaller portion of the address space allows a worm copy to send more probes through a particular instrumented router in the same amount of time.

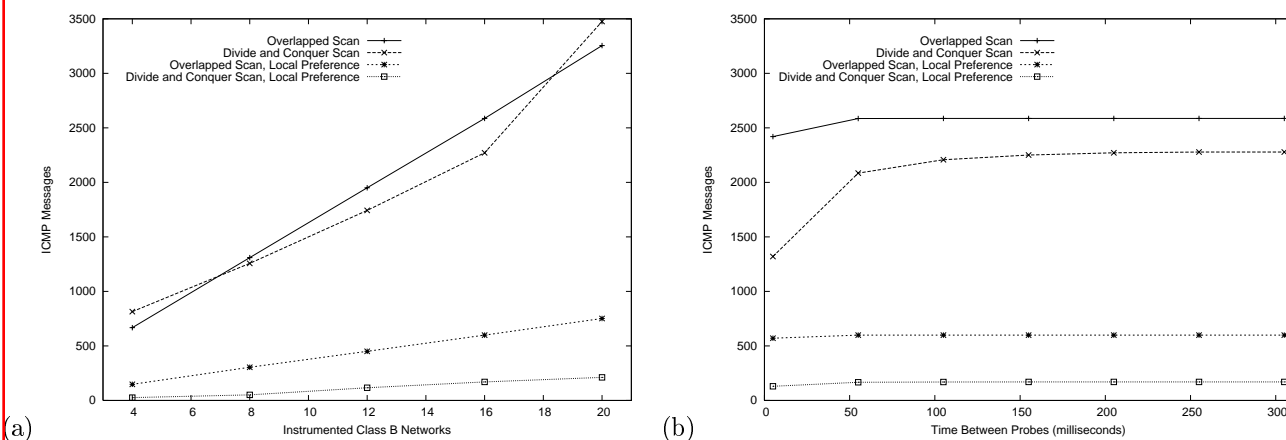


Figure 1. The number of ICMP messages that a worm induces as a function of (a) the number of instrumented Class B networks and (b) the worm's scan rate. Each graph stops at the point where one thousand hosts are infected, and each data point is the average of twenty worm runs.

To put the scan rates from Figure 1b in perspective, the Code Red v2 worm infected 360,000 machines on July 19, 2001, and each instance of Code Red used one hundred concurrent scanning threads, each of which suffered a twenty-one second timeout when attempting to contact an unreachable machine.⁷ Although this penalty did

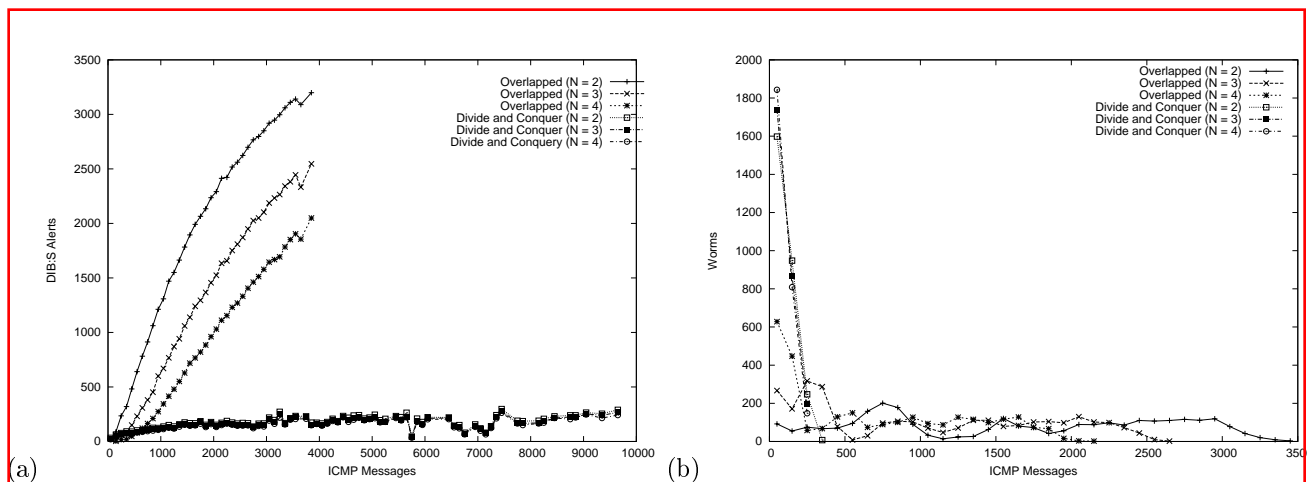


Figure 2. (a) shows the relationship between the number of ICMP messages that DIB:S observes (x-axis) versus the average number of alerts that DIB:S produces (y-axis), while (b) shows the average number of worms (y-axis) that have a particular alert count (x-axis). In both (a) and (b), each point is at the center of a 100-count bucket, and shows the average number of alerts or worms for the corresponding bucket.

not occur for reachable machines, we still can calculate the each instance of Code Red completed approximately 300 scans per minute or 5 per seconds, corresponding to an average time between scans of 200 milliseconds. For comparison, each instance of the Sapphire/Slammer worm was reported to scan nearly 4000 machines per second at its peak,⁸ corresponding to an average time between scans of 0.25 milliseconds. Extremely fast worms such as Sapphire/Slammer, currently still rare, are left to a future paper, although we did see detection successes with the threshold model even at these speeds in our earlier work.¹

The number of ICMP messages is only half the story, since even a single machine could produce the observed volume of ICMP messages simply by scanning unreachable addresses as quickly as possible. Instead, we must see a growing number of scanning machines over time. DIB:S aggregates the ICMP messages into alerts, generating an alert if there are more than N ICMP messages from the same source, *and* allowing only one alert per unique source machine per Δt seconds. Figure 2a shows the mean number of alerts generated for a given number of ICMP messages. The plot includes six thousand worm runs, including the runs represented in Figure 1. For this graph, we used a Δt value of 1800 seconds and N values of 2, 3 and 4, and grouped the worm runs according to whether the worm used an overlapped scan or a divide-and-conquer scan. Three effects are clear. First, the number of alerts decreases as N increases, simply because DIB:S must see more probes for each unique source address before issuing an alert. Second, divide and conquer worms generate dramatically fewer alerts than overlap worms, and generate a nearly constant number of alerts no matter how many ICMP messages are available. The reason is that any given Class B network is scanned by only a few copies of a divide and conquer worm, since only a few copies of the worm will have all or part of that particular Class B network in their scan range, and many of those only for a brief period of time. Finally, it is satisfying to note that the maximum standard deviation for the overlap data points is 11% of the mean, and for the divide-and-conquer data points 52% of the mean. At least for overlap worms, the number of ICMP messages and the value of N are primary factors influencing the number of alerts.

Figure 2b is a different view of the same data, and shows the average number of worms for which DIB:S generated certain numbers of alerts. For the divide and conquer strategy, every worm produced two hundred alerts or fewer during the first one thousand infections, while for the overlap strategy, many worms produced alert counts into the thousands. Again, this is because any particular instrumented router sees traffic from only a few instances of a divide and conquer worm. In addition, although it is not indicated in the graph, the overlap worms that produce fewer alerts generally are the ones that have a preference for generating local Class B addresses, an effect that can be seen in the ICMP graphs in Figure 1.

4.2. Detection without Noise

We consider two different detection models. The first detection model is our original threshold model.¹ This model assigns a probability to a sequence (or track in TRAFEN terminology) of DIB:S alerts based on the time between each alert and the destination ports. Specifically, the first alert for a particular service port is assigned likelihood 0.1, since a single alert cannot indicate a worm. For each subsequent alert, the model calculates the match likelihood between the new alert and the sequence so far. The details were discussed in our previous work,¹ but essentially the match likelihood is set between 0.675 and 0.925, scaled linearly according to where the time between alerts falls in the range 300 to 10 seconds. The likelihood of the new track is the average of the old likelihood and the match likelihood, and the model will indicate a propagating worm when the track likelihood rises above 0.9. Although this formulation appears odd at first glance, it essentially is just a recasting of a simple threshold count into the probability and hypothesis-tracking environment of TRAFEN. A series of six scans, each of which occurs within 10 seconds or less of each other, will cause the track probability to rise above 0.9. The general range of 10 to 300 seconds, however, is tuned for Code Red or faster worms (e.g., 200 milliseconds or less between probes).

The second model is the new line-fit model. Here we bucket the alerts by time using several different bucket sizes, each bucket containing a count of the number of alerts occurring in that bucket. We calculate the slope of the best-fit line for the alert counts over a sliding window of m buckets. In addition, we divide each set of m buckets into four equal-sizes groups (meaning that m must be a multiple of 4), and separately calculate the slope for each group. Then, if all of the following conditions are met for *any* of the bucket sizes, the model will indicate a worm: (1) the slope over the m buckets is greater than some positive slope threshold s ; (2) the slope of each of the four sub-groups is greater than zero; and (3) the slope of each sub-group is greater than the slope of the previous sub-group.

The last two conditions are designed to reflect the fact that a worm induces an increasing number of alerts over time. Slopes less than zero, or a decrease in slope from one bucket to the next, is suggestive of noise, rather than a worm. For our detection work, we specifically picked bucket sizes that were a constant multiple of our primary worm scan rates. Experiments indicated that the correct multiple was one thousand, and we picked bucket sizes of 5,000, 55,000, 105,000, ..., and 305,000 milliseconds. The reason why the multiple 1,000 works significantly better than 100, for example, relates to the expected number of alerts that fall into each bucket. If the bucket size is too small a multiple of the scan rate, the alerts will not form an increasing line as they are divided into buckets. If the bucket size is too large, it will take too long to detect fast moving worms. Experimentation further indicated the 0.2 was the appropriate value for the slope threshold s and that 32 was the appropriate value for the window size m , although detection accuracy (and false-positive rates) were nearly identical for a range of values around these points (particularly for slopes up to 0.3 window sizes down to 24).

With these two models in hand, we now turn our attention to the detection results. Figure 3a shows detection performance, as a function of instrumented Class B networks, for the same worm runs whose ICMP counts were shown in Figure 1a. We use a DIB:S parameters $\Delta t = 1800$ seconds and $N = 2$, a value that provides us the maximum alerts with which to work. In addition, since our explicit goal is to detect worms as early as possible within their lifetimes, this graph, as well as the other graphs, indicate the probability of detecting the worm within the first one thousand infected machines (corresponding to an infection percentage of 0.5% for our primary vulnerability population of 200,000 hosts). In many cases, the system will detect the worm later in its lifetime if the simulations are allowed to continue past one thousand infections, an issue that will be revisited in a future paper.

As one might expect intuitively from the ICMP and alert counts presented above, the line-fit model does not detect any divide-and-conquer worms, no matter how many instrumented routers. In addition, the threshold model, while doing better, has trouble detecting divide-and-conquer, local-preference (DC+LP) worms when the number of instrumented routers is small. When the number of instrumented routers is small, a DC+LP worm cannot produce enough alerts for even the threshold model to work. Finally, the line-fit model detects fewer overlap, local-preference (O+LP) worms as the number of instrumented routers decreases. With fewer routers,

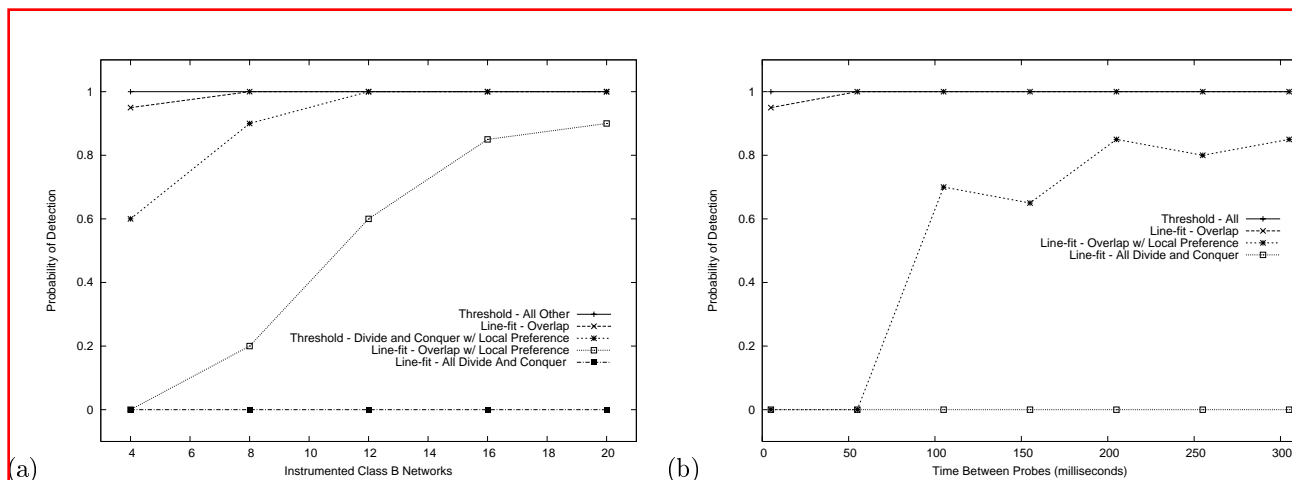


Figure 3. The probability of detecting a worm as a function of (a) the number of instrumented Class B networks and (b) the scan rate.

the O+LP worms do not induce enough ICMP messages within the first thousand infected machines.[§]

Figure 3b shows detection performance, as a function of scan rate, for the same worm runs whose ICMP counts were shown in Figure 1b. Here the line-fit model again detects none of the divide-and-conquer (DC) worms, detects every overlap, global-preference worm (O+GP), and detects fewer O+LP worms as the worms scan faster. This latter effect happens even for scan rates at which the ICMP rate limit does not play a role, indicating that there are not enough alerts to produce a smooth line (of slope 0.2 or better) for our bucket sizes. The threshold model, on the other hand, detects every worm since it simply requires a small number of DIB:S alerts within a sufficiently short period of time, a condition that is easily satisfied with sixteen instrumented routers.

Figure 4 considers the effect of the worm scan rate in more detail. Figure 4a shows the detection performance of the line-fit model for a wider range of worm scan rates than Figure 3a. Although the bucket sizes were not picked specifically for these worm rates, the results generally are the same as before. O+GP worms are detected, O+LP worms are detected more accurately as the worms scan more slowly, and DC worms are not detected at all. Note that detection of O+GP worms begins to fail for the slowest scan rates, farthest away from our selected bucket sizes. Figure 4b shows the O+GP detection performance of the line-fit model for three specific bucket sizes, 55,000 through 155,000. As can be seen, each bucket generally works perfectly within a particular range of scan rates, but then displays worse performance as the scan rate gets farther away from the bucket size. Intuitively this makes sense. For large bucket sizes, a fast-scanning worm will not fill enough buckets to satisfy our slope test, at least within the first one thousand infections. Conversely, for small bucket sizes, a slow-scanning worm will induce a more random distribution in the distributed alert counts, again causing our slope test to fail. Overall, this results suggest that no single bucket size is sufficient.

For the line-fit model, other interesting effects arise depending on which parameter we choose to vary. Figure 5a shows line-fit performance for up to 32 instrumented Class B networks. As the number of networks increases, detection performance for O+LP worms increases steadily, while all DC worms remain undetected, except for a single DC worm detected with 32 instrumented networks. Both effects arise due to the capture of more ICMP messages as more instrumented networks become available. Figure 5b shows line-fit performance as a function of the number of vulnerable hosts. Since we always divide the vulnerable hosts among 1,000 Class B networks in the local-preference case, detection performance for O+DP worms drops as the number of hosts increases, simply because each worm instance can infect more hosts without probing outside of its local Class B network. Although we do not show the graph, a similar effect can be achieved by varying the local address

[§]Local-preference, as well as divide-and-conquer worms are detected significantly faster when each Class-B network contains one or more DIB:S routers internally as well, a promising avenue for future investigation.

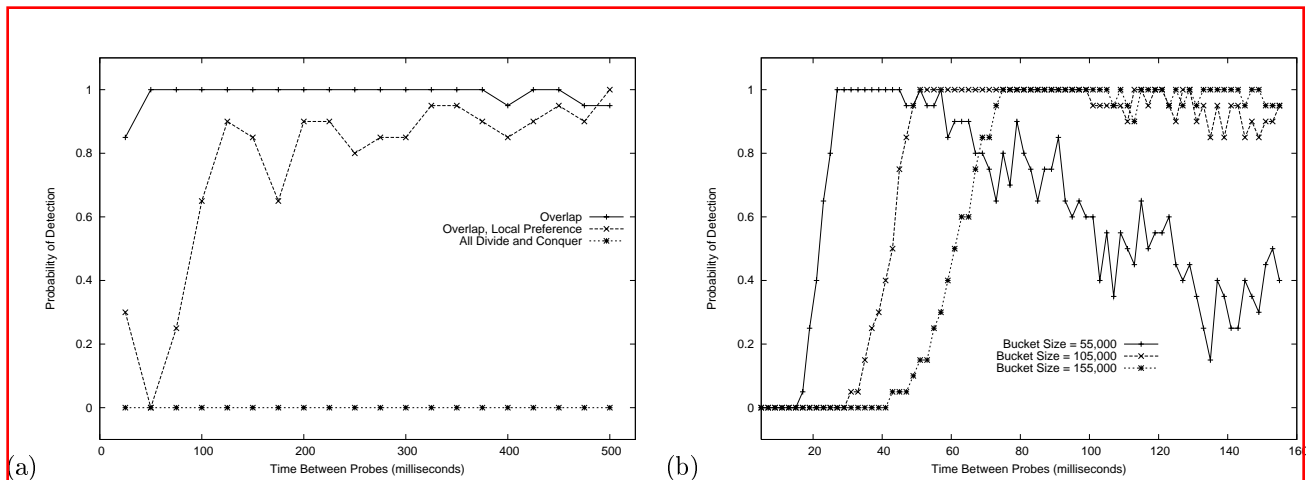


Figure 4. (a) shows the detection performance of the line-fit model for a wider range of scan rates, while (b) shows the detection performance of the line-fit model for four specific bucket sizes. (b) includes only O+GP worms, the case for which DIB:S generates the most alerts.

probability. Figure 5c shows line-fit performance as a function of unreachable percentage, i.e., the fraction of hosts for which an instrumented router can generate an ICMP message. Detection performance improves as the unreachable percentage increases, since the worm induces more ICMP messages. Not shown is worm detection as a function of flash-list size. No worms are detected at all for flash sizes above three-hundred hosts, however, since the worms infect too many hosts too quickly to induce the needed number of ICMP messages.

Overall, as shown in Figure 5d, the detection performance depends heavily on the number of alerts available to the detection models. Although better models for worm behavior will improve detection, it also will be critical to explore ways in which DIB:S can collect more raw data and produce more alerts.

4.3. Detection with Noise

The real world is noisy, and the discussion is incomplete without an examination of how noise affects our detection performance. First, Figure 6 shows the false detections per hour produced by port and scan noise rate over a given noise-generation rate. For the scan noise, we assumed an infinite source pool and set the average number of probes per scan to 10, so that almost every attempted scan would produce a DIB:S alert. For the port noise, we assumed a source pool of 5,010, which is the midpoint of a size range that appears to produce the most DIB:S alerts. Source-pool sizes that are too small generate too few alerts no matter how long the simulation runs, while source-pool sizes that are too big do not generate enough alerts within the first one thousand infected machines. As the figure indicates, the line-fit model reacts significantly better to noise than the threshold model. At the same time, however, it is interesting to note that the threshold model works well in our deployed worm-detection application, where the noise rate is significantly lower than 0.01. The threshold approach can play a role either as a detection mechanism in a low-noise environment or as a trigger for more intensive computations.

To put the noise-generation rates in perspective, Zou reports that Goldsmith observed an average of 29.5 noise probes per hour, from an average of 17.4 source addresses, on one Class B network near the time of Code Red's release.⁹ Since we instrument at the level of Class B networks, the first number becomes 0.008 noise events per instrumented network per second, just at the left side of the graphs in Figure 6. For the best-fit model, 0.008 noise events per router per second corresponds to less than 1 false detection every two days. In addition, only 17.4 source addresses per hour would eliminate the effect of both noise types, since there could never be enough DIB:S alerts produced to cause a false detection. Overall, additional real-world studies are needed to quantify the noise levels associated with particular destination ports or network segments. In addition, the acceptable false-detection rate depends significantly on the use of the alerts. Throttling bandwidth temporarily, for example, is significantly different than blocking all access to a service (and requiring human intervention to restore access).

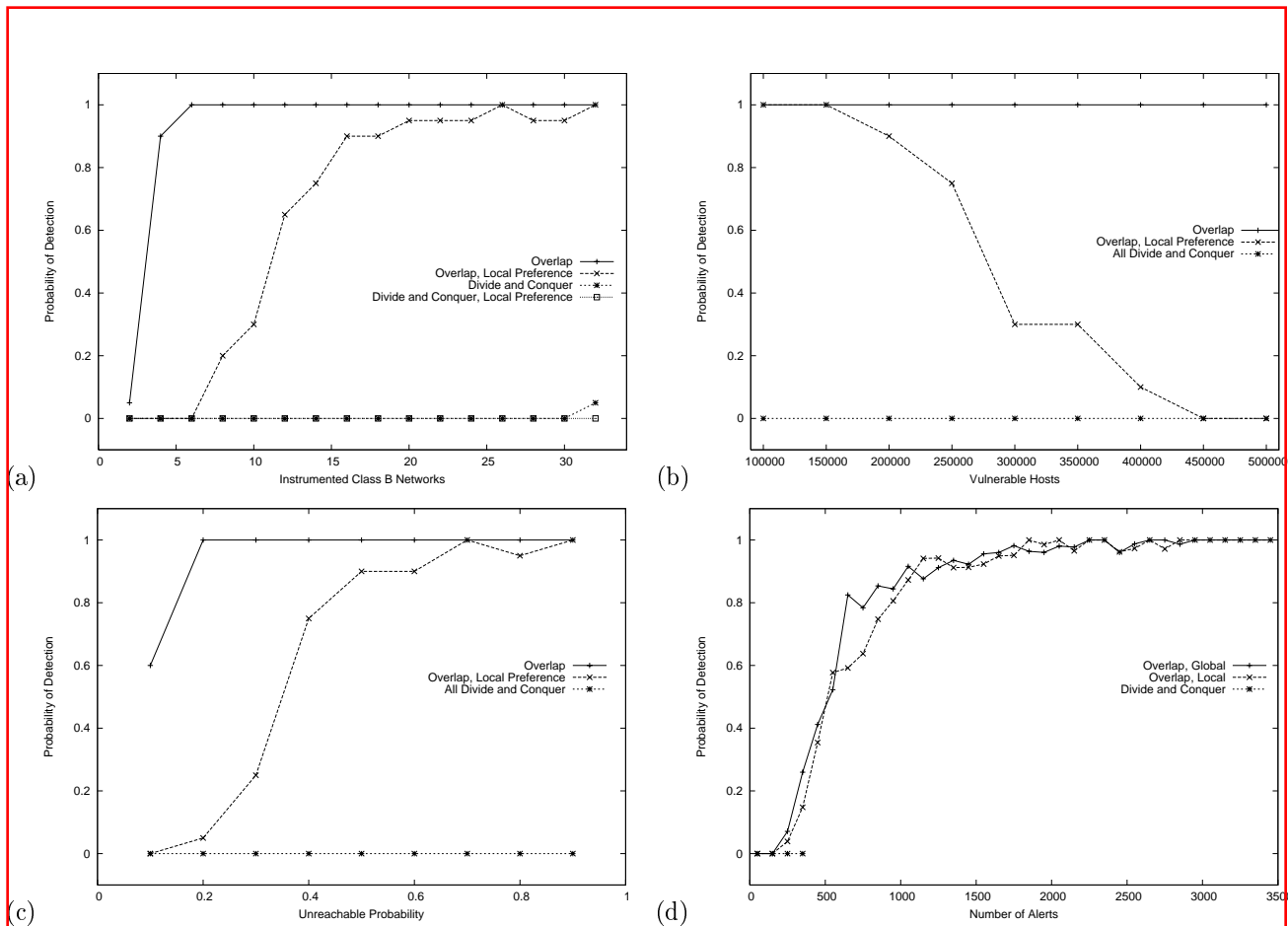


Figure 5. Detection performance for (a) a larger number of instrumented Class B networks, (b) different numbers of vulnerable hosts, (c) different fractions of unreachable hosts, and as (d) a function of the number of DIB:S alerts available to the detection models.

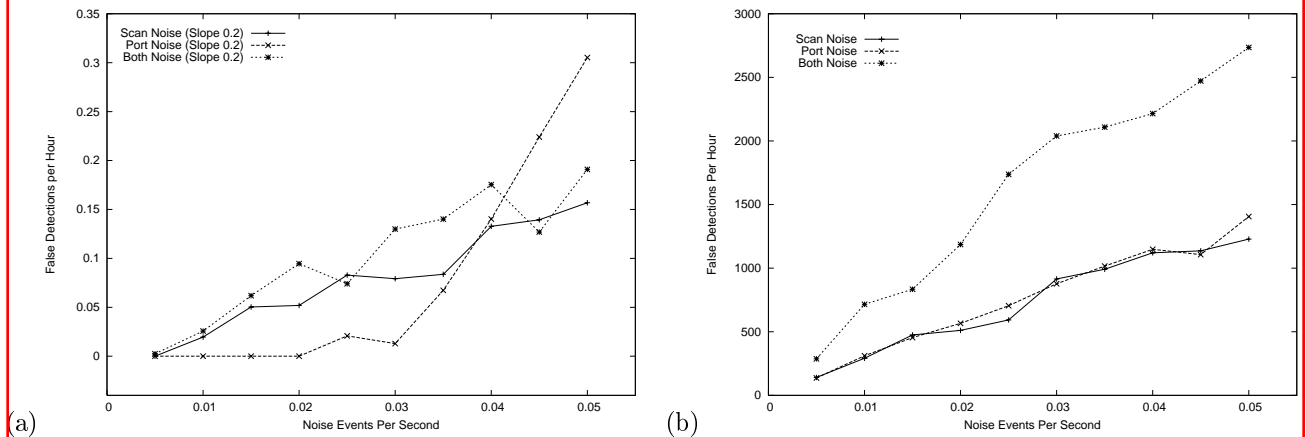


Figure 6. False detections per hour for (a) the line-fit model and (b) the threshold model, in each case for scan noise, port noise, and combined noise. We multiply the noise events per second by the number of instrumented routers (16 in this case) to get the overall noise generation rate, and each data point was derived over 542 hours of simulated noise runs.

With the false-detection rate in mind, we now can consider the effect of the noise when a worm actually is present, a situation in which the results unfortunately are not as rosy. Figure 7 shows detection performance of the line-fit model when both scan and port noise is present. The two graphs correspond to two different noise-generation rates, 0.08 (the left end of the noise curves) and 0.03 (just over the midpoint). The effect of the noise on detection is significant. First, the detection of O+LP worms becomes significantly worse, apparently as the noise patterns reduce the “straightness” of the worm’s alert curve. Conversely, the detection of DC worms becomes better, apparently as the noise pattern fills in gaps in the alert curve. Of course, such an effect is not one on which we can rely for detection. Finally, as one would expect, both of the previous effects become significantly worse as the noise generation rate changes from 0.08 to 0.03. We do not show the effects of background noise, but the background noise rate can become quite high (e.g., 8 probes per router per second) while still letting through enough ICMP messages to detect O+GP worms. Of course, the effects of background noise can be balanced by allowing the instrumented routers to generate more ICMP messages per second.

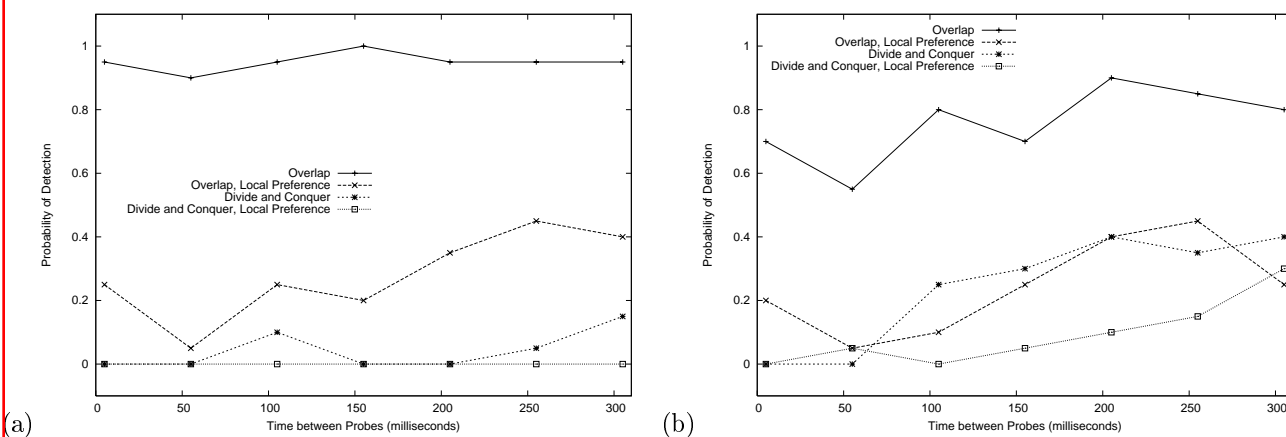


Figure 7. The effect of noise on detection performance for noise-generation rates of (a) 0.08 and (b) 0.03.

Overall, the line-fit model is much more noise resistant than the threshold approach, and remains resistant even at relatively high noise levels. The noise resistance does decrease steadily, however, as the noise rate increases. The effect is particularly pronounced in our case, since we are trying to detect the worms within the first thousand machines. For this early detection, noise-related ICMP messages easily can exceed the worm-related ICMP messages. Setting a more liberal detection target, and using correspondingly higher detection thresholds, would provide significantly better noise resistance at the expense of later detection.

5. RELATED WORK

Zou, Gao, Gong, and Towsley have developed an approach to worm detection based on ingress and egress filters placed at key network points.⁹ Although a more involved way of gathering data, it essentially provides similar information as our participating DIB:S routers. Their system could easily be integrated into TRAFEN, providing additional data to DIB:S. For analysis, they use the Kalman filter, which has the advantage of being resilient to missed observations. The Kalman filter is currently part of TRAFEN’s tracking algorithm library, and begs to be experimented with in the DIB:S data context. Additionally Zou, Towley, and Gong present a mathematical analysis of the various worm propagation strategies, a crucial component for modeling and detecting worms.¹⁰ Alternatively, Qin et al. propose a correlation approach that can be used in place of a Kalman filter for detecting worm activity on a local network.¹¹ Their approach matches incoming and outgoing traffic activity to detect likely infections.

Yagneswaran, Barford, and Ullrich study the characteristics of worm versus non-worm traffic patterns.¹² Most of their techniques are similar to methods present in TRAFEN, although others do not apply well to our DIB:S data stream. Yagneswaran also suggests monitoring of unused address space, something our DIB:S system

does inherently with the Network Unreachable messages, which are a subclass of the ICMP-T3 message. Baras, Cardenas, and Ramezani discuss detection of self-propagating code based on traffic flow patterns.¹³ Of particular interest is their analysis of network topology, a factor that we do not take into account. Finally, Moore, Voelker, and Savage also use ICMP-T3 messages, although not for worm detection.¹⁴ Their goal is to detect “backscatter” from denial-of-service attacks. Although not directly related to worm detection, their work illustrates the use of Internet error messages for anomaly detection.

6. CONCLUSION AND FUTURE WORK

The line-fit model, combined with the ICMP alert data from the DIB:S system, displays attractive detection and noise-resistance characteristics for scanning worms. The threshold model, although dramatically more vulnerable to noise than the line-fit model, remains useful for low-noise environments or as a trigger for computationally intensive models. More remains to be done, however. In addition to exploring detection behavior for significantly larger numbers of instrumented routers, as well as past the one-thousand machine boundary, there are several promising areas of future work.

First, this paper presents experimental results, deriving most parameter values from simulation runs. It is likely, however, that we can mathematically derive optimal parameters starting from the standard epidemiology equations¹⁵ or from epidemiology equations tailored to specific worm types.¹⁰ If such an end-to-end mathematical formulation is possible, it will greatly simplify the task of setting system parameters so as to detect the desired categories of worms. Such a formulation also could influence future modeling directions, since the line-fit model, although logical, also was derived through experimentation.

Second, DIB:S, TRAFEN, and our two detection models can detect only those worms that use a random process to generate at least some of their target addresses. A contagion worm, for example, would be invisible to the detection system since it would never attempt to contact an unreachable address, instead relying on the normal contact between communicating machines to identify its targets.¹⁶ Although our detection models still will be useful, additional models and significantly different data sources will be required. Similarly, the current line-fit model has trouble detecting extremely fast worms and worms that prefer to probe local addresses, and do not detect divide-and-conquer worms. In addition, slow worms will be lost among the noise. Preliminary experimentation has shown that having one or more participating routers *within* each class-B network yields a significant improvement over the router setup presented in this paper. Router placement is therefore an important direction of future work. Additionally, TRAFEN can support multiple models at once. A model tailored to detect noise could run simultaneously with a worm-detection model, with the combined results fed into a higher-level detection algorithm that considers the relative accuracy of the models.

TRAFEN is capable of handling many more streams of data than just DIB:S. As of this writing we are integrating a multitude of other sensors into our system, as well as expanding TRAFEN’s model database. This expansion will allow the query-driven detection of other types of network anomalies, and will allow the use of other types of data in the worm-detection models. Finally, the current systems works with relatively little raw data, which means that an attacker, who understood the detection models, would need to generate relatively little falsified data to cause a false detection. Protecting a worm-detection system from malicious data injection is a critical, but unexplored, area of future work.

Overall, TRAFEN provides the framework in which worm-detection and other models can be developed quickly, and our current worm-detection models show promise. Together they are an effective starting point for building a detection system for broad categories of worms.

Acknowledgments

Many thanks to George Bakos and George Cybenko for many productive discussions on ICMP, Process Query Systems, and worm detection; to the entire TRAFEN development team without whom our work would be impossible; and to the Department of Homeland Security for their generous financial support. Supported under Award No. 2000-DT-CX-K001 from the Office for Domestic Preparedness, U.S. Department of Homeland Security. Points of view in this document are those of the author(s) and do not necessarily represent the official position of the U.S. Department of Homeland Security.

REFERENCES

1. V. H. Berk, R. S. Gray, and G. Bakos, "Using sensor networks and data fusion for early detection of active worms," in *Proceedings of the SPIE Aerosense conference, Orlando Florida*, April 2003.
2. M. Liljenstam, D. M. Nicol, V. H. Berk, and R. S. Gray, "Simulating realistic network worm traffic for worm warning system design and testing," in *ACM Workshop on Rapid Malcode, Washington DC*, October 2003.
3. V. Berk, G. Bakos, and R. Morris, "Designing a framework for active worm detection on global networks," in *Proceedings of the IEEE International Workshop on Information Assurance, Darmstadt Germany*, March 2003.
4. D. B. Reid, "An algorithm for tracking multiple targets," *IEEE Transactions on Automatic Control* **AC-24**, pp. 843–854, December 1979.
5. V. Berk, W. Chung, V. Crespi, G. Cybenko, R. Gray, D. Hernando, G. Jiang, H. Li, and Y. Sheng, "Process query systems for surveillance and awareness," in *Proceedings of the Systemics, Cybernetics and Informatics (SCI2003) conference, Orlando Florida*, July 2003.
6. 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 2003)*, October 2003.
7. D. Moore, C. Shannon, and J. Brown, "Code Red: A case study on the spread and victims of an Internet worm," in *Proceedings of the Second Internet Measurement Workshop (IMW 2002)*, November 2002.
8. D. Moore, V. Paxson, S. Savage, C. Shannon, S. Staniford, and N. Weaver, "The spread of the Sapphire/Slammer worm," technical report, CAIDA, 2003.
9. 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 2003)*, October 2003.
10. C. C. Zou, D. Towsley, and W. Gong, "On the performance of internet worm scanning strategies," tech. rep., University of Massachusetts, November 2003.
11. X. Qin, D. Dagon, G. Gu, W. Lee, M. Warfield, and P. Allor, "Worm detection using local networks." Submitted to USENIX 2004 and made available on several security mailing lists, 2004.
12. V. Yagneswaran, P. Barford, and J. Ullrich, "Internet intrusions: Global characteristics and prevalence," in *Proceedings of the International Conference on Measurements and Modeling of Computer Systems (SIGMETRICS 2003)*, San Diego California, June 2003.
13. J. Baras, A. Cardenas, and V. Ramezani, "On-line detection of distributed attacks from space-time network flow patterns," in *23rd Army Science Conference*, December 2002.
14. D. Moore, G. M. Voelker, and S. Savage, "Inferring Internet Denial-of-Service activity," in *Proceedings of the 10th USENIX Security Symposium (USENIX'01)*, Washington DC, August 2001.
15. D. Daley and J. Gani, *Epidemic Modeling*, Cambridge University Press, 1999.
16. S. Staniford, V. Paxson, and N. Weaver, "How to Own the Internet in your spare time," in *Proceedings of the 11th USENIX Security Symposium (Security '02)*, August 2002.