

Hitlist Worm Detection using Source IP Address History

Jeffrey Chan, Christopher Leckie, Tao Peng

NICTA VRL, Department of Computer Science and Software Engineering

The University of Melbourne, Australia

{jkcchan, caleckie, tpeng}@csse.unimelb.edu.au

Abstract—Internet worms are a growing menace due to their increasing sophistication and speed of propagation. In this paper, we present a new worm detection scheme, *History-based IP Worm Detection*. It uses the difference in the distribution of source addresses between regular users and scanning hosts to distinguish between worm probes and normal accesses. This property is used to implement a weighted source address counting scheme, and a change point detection technique is used to detect surges in the rate of source addresses seen. In contrast to many existing techniques for worm detection, our approach is able to detect worms that only scan active addresses, while having linear time complexity.

I. INTRODUCTION

The Internet's easy and wide access has allowed malicious programs, like worms, to propagate in unprecedented ways. The traditional response to new worms has been to manually generate the defining signatures, and distribute these signatures for use in traffic filters. However, worms like Slammer [9], which infected most vulnerable systems in a matter of minutes, indicate this method is becoming infeasible. Motivated by these concerns, the intrusion detection community has investigated various proposals to automate worm detection [17] [13] [16].

Some detection schemes [13] involve automatically extracting frequent traffic patterns and considering these as worm signatures. These schemes can be effective against a wide range of worms. However, they suffer from relatively high false positive rates and high processing requirements.

Other schemes utilise unused IPv4 addresses, and rely on the observation that legitimate traffic rarely make connection attempts to these addresses [17] [16]. Hosts that make many connection attempts to these unused addresses are flagged as infected hosts. Generally, these schemes are effective against worms that use random probing to discover new victims. However, these schemes are ineffective against worms that do not probe indiscriminately, such as "hitlist" worms [14], which are seeded with a database of active target hosts.

There are also schemes that combine evidence from multiple distributed sub-networks to increase the address space been monitored, which can decrease the detection time and false positive rates [6]. However, these schemes require a significant number of heterogeneous networks to buy into the scheme, which is not realistic at this point of time.

Hence, there is a need for a detection algorithm that i) only monitors traffic on the network where it is deployed; ii) can

detect hitlist worms quickly; iii) have low false positive rates; and iv) have low computational requirements.

In this paper, we present a worm detection scheme called *History-based IP Worm Detection* (HIWD). It does not rely on scans to unused addresses for detection, so it can detect hitlist worms, and has linear time requirements in the worst case. HIWD is also designed for deployment at edge routers and local network firewalls, which avoids the buy in problem of distributed detection. It uses inbound source and outbound destination address counting to detect worms, as well as using the following observations to lower the false positive rate:

- 1) The group of regular users of a network form a relatively stable set of source addresses. In contrast, during a worm outbreak, the source addresses of connections requests are widely distributed [18] and unlikely to have been seen at the network previously.
- 2) Source addresses that share a common network prefix are usually topologically near. If one of these addresses represents a regular user, it is likely the other addresses represent legitimate users.

From our results, we find HIWD is able to quickly detect worms that only probe active addresses. When tested on a small network, our scheme detected these hitlist worms when only 6% of vulnerable hosts were infected, and with no false positives. To the best of our knowledge, our scheme is the first to be able to detect hitlist worms quickly and efficiently when protecting a local network. There has been work in randomising the address space of the network being protected to defeat hitlist worms [1], but this aims to reduce the effectiveness of hitlist worms rather than detect them.

II. BACKGROUND AND RELATED WORK

In this section, we present an introduction to a worm life-cycle proposal, several worm scanning methods, as well as a survey of other worm detection schemes.

A. Worm Lifecycle

In [3], a four stage worm life-cycle was presented. The first stage, *target selection*, represents an infected host scanning for new victims. In the next stage, *exploitation*, the worm compromises the selected victim. Then in the third stage, *infection*, the worm copies itself onto the compromised host. In the last stage, *propagation*, the newly infected victim scans for new victims to infect. This stage is the same as the target

selection stage, except it is from the perspective of the newly infected host.

B. Scanning Methods

1) *Random Scanning*: Many worms in the wild use this scanning strategy. They choose a new host to scan in a random, uniform way.

2) *Routable Scanning*: Propagation can be accelerated by only probing routable addresses. Zou et al [19] found 28.6% of the IPv4 address space has been allocated and routable. Hence, a routable scanning worm can still scan all potential victims, but reduce the scanning space by 71.4%.

3) *HitList Scanning*: All worms suffer from a long initial startup phase. One proposed solution [14] is for the worm author to collect an initial hitlist of targets, and use it to generate probing targets. When the hitlist is exhausted, the worm can revert back to random scanning. If the hitlist only contained existing hosts, hitlist worms can overcome detection techniques that rely on detecting probes to unused addresses. An example of a hitlist worm is the Witty worm [12].

C. Detection Schemes

Detection schemes can be classified into three broad areas: unused address, signature, and infection pattern based schemes. Schemes based on using unused addresses involve monitoring scans to unused addresses and inactive services. Signature based schemes automatically generate worm signatures, and infection pattern based schemes target one or more aspects of the worm lifecycle.

Generally, all schemes that rely on monitoring probes to unused addresses [17] [16] have good performance against random scanning worms. However, their biggest weakness is their inability to detect worms that do not scan indiscriminately, like the hitlist worms.

Examples of signature-based schemes are in [10] [15] [13]. In [13], the authors explored using Rabin fingerprints to automatically generate common signatures of inbound packets.

Finally, there are the infection pattern based schemes, in which we consider our scheme to belong. Chen and Heide-mann [5] proposed a scheme, named DEWP, that matched incoming and outgoing destination ports that have one or more connection requests. Their idea is based on the observation that worms consistently probe for new victims on the same set of destination ports. After matching ports, they searched for large increases in the number of unique destination addresses on those ports. We will show later that DEWP has a high false positive rate.

In summary, none of the proposed schemes a) has low false positive rates, b) is reasonably efficient and c) is able to detect worms that do not scan indiscriminately. Signature based schemes have property c), but not a) or b). Schemes based on unused addresses have property a) and b), but not c). Other proposed infection pattern based schemes lack one or two of the properties. As we shall show, our scheme possesses all three properties.

III. HISTORY-BASED IP WORM DETECTION

In this section, we define the scope of our problem, as well as outlining the worm features our scheme uses for detection. Then we describe our History-based IP Worm Detection (HIWD) scheme.

Let us define the scope of our problem. We define the *protected network* as the network protected by the detection scheme, and the *rest of the Internet* as the other parts of the Internet. Our worm detection scheme monitors the traffic passing between the protected network and the rest of the Internet. Given access to the headers of inbound and outbound packet traffic, the detection scheme should output an alert when it detects the infection of one or more protected hosts.

Next, we describe the four detection features that HIWD is based on.

- 1) Regular users of the network from a stable group of addresses, whereas infected, scanning hosts are more likely to be widely distributed and never before seen at the network.
- 2) When a network is under worm attack, the number of unique inbound source addresses seen per time interval rises above average levels [18].
- 3) For small to moderately sized networks that HIWD is designed for, a host that has been infected will scan for new victims on the same port it was infected from. Hence, it is likely that the number of unique outbound destination addresses seen per time interval (for that port) rises above average levels [3].

Due to the lack of space, we do not elaborate feature 2 and 3 further. Interested readers should refer to the provided references. For feature 1, a worm, by its nature, will spread as quickly and widely as possible. It has been observed by Yegneswaran et al [18] that worm victims are more or less uniformly distributed across the Internet. In contrast, Peng et al [11] observed that many of the users of a network were stable and regular, and Jung et al [8] observed that a new user to the network is likely to come from the same network as a regular user, while a worm victim is much less likely to come from a previously seen network. Combining these three observations, we propose that the source address (and the subnet) of a worm probe is more likely to have never connected to the protected network, while most source addresses (and subnets) of legitimate users would have previously connected to the protected network. In addition, a host with a previously seen address is more likely to be uninfected than a host with just a previously seen subnet.

To confirm our conjecture, we examined 24 hour traces collected from the Computer Science Department network from May 22nd to July 12th 2004. The examination involved computing how many addresses and subnets over a four day period (7th June to 10th June) appeared in the previous two weeks (23rd May to 6th June). These results are presented in Table I. As the table indicates, 65-75% of addresses and 74-76% of subnets were seen previously. Although there is still a significant number of new addresses, we shall show that our

Date	Src IP %	Src Subnet %
07-June-04	65.6%	74.2%
08-June-04	69.4%	73.6%
09-June-04	71.6%	74.1%
10-June-04	75.4%	76.0%

TABLE I: Percentage of source addresses and subnets of prefix width 24 previously seen from 23-May-04 to 06-June-04. Melbourne University Computer Science traces.

detection scheme is still effective with this level of regular users. Note that Peng et al [11] obtained better results when conducting the same experiment on traces from the University of Auckland WAND project and a class C Australian ISP, where they observed, respectively, 88-90% and 76-81% of the test addresses were previously seen. We denote the list of source addresses and subnets previously seen as the source address database (SAD) and the source subnet database (SSD) respectively.

A. History-based IP Worm Detection

As mentioned, HIWD utilises the four detection features (1)–(4). It consists of two steps, which is based on the target selection and propagation stages of the worm lifecycle. A vulnerable host in the protected network will first have to be scanned, infected, then in turn scan for new victims on the same set of ports it was infected from. The first stage, *History-based Detector*, monitors inbound traffic for increases in the number of unique source addresses seen in a time interval, to detect initial scanning activity of a worm attack. To further increase our confidence of an attack, a second stage (*Scan Detector*) based on detection feature 3, is introduced. If an attack is occurring, and some subset of the hosts in the protected network are infected, then it is likely those newly infected hosts will scan outside of the network on the same set of ports they were infected from. Based on this observation, the History-based Detector reports a list of suspicious ports to the Scan Detector. For each reported port, the Scan Detector monitors for increases in the number of unique outbound destination addresses. For those ports that are also reported as suspicious, it is likely there is a worm propagating using those ports.

To decide which deviation in the monitored address counts are suspicious, both detectors use a change point detection technique called CUSUM [2]. CUSUM can detect abrupt changes, as well as slower but sustained increases in the monitored addresses, which naive static thresholding cannot achieve without introducing more false positives. By using the CUSUM technique, HIWD can detect slower worms at similar false positive rates. Figure 1 provides an overview of HIWD.

1) *History-based Detector*: The History-based Detector monitors inbound traffic and looks for abrupt changes in the rate of unique (weighted) source addresses seen for each destination port. The weighting is based on whether the address has been seen before. Let A be the source address of an incoming connection, and i) S_1 be the weight of A being in both the SAD and SSD ii) S_2 the weight of A being in

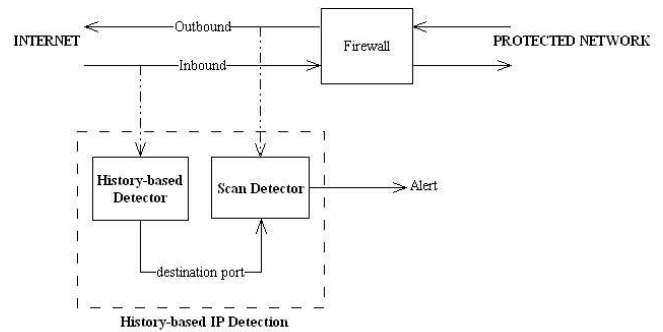


Fig. 1: Architecture of History-based IP Worm Detection

just the SSD, and iii) S_3 be the weight of being in neither. We then define $0 < S_1 < S_2 < S_3$. Addresses that have been seen before (weight S_1) are likely to be regular users. Addresses that share prefixes with seen addresses (weight S_2) are more likely to be legitimate new users than infected hosts (e.g. address of a fellow postgraduate from the same university). Addresses that have not been seen or share prefixes with seen addresses (weight S_3) are most likely to be worms, hence penalised most heavily. This weighting assists with reducing false positive rates and detection time, as a few scans/accesses from worms will trigger the detector, while it takes more accesses for regular users to falsely trigger the detector.

B. Change Point Detection

In this section, we present a summary of the change point detection theory, especially how it was adapted for worm detection. For a more complete presentation, refer to [2]. In change point detection, the aim is to monitor random sequences $\{X_n\}$ and detect a significant deviation from the mean α , called a change point. For our detection scheme, X_n is the total (weighted) address count per detection time interval.

One change point detection technique is the Cumulative Sum (CUSUM) algorithm [2]. We adapted the method presented in [11] for worm detection (for full details, refer to [4]). The basic idea is to accumulate values of the detection feature X_n that are significantly higher than the mean value, α . The algorithm requires a negative mean, which is not the case with α . Therefore, we first shift $\{X_n\}$ by a value β , so that it has a negative shifted mean $\omega = \alpha - \beta$. Let y_n be the accumulated values of $X_n - \beta$ that are > 0 . y_n can be defined as $y_n = (y_{n-1} + X_n - \beta)^+$, $y_0 = 0$, where $x^+ = x$ if $x > 0$, or 0 otherwise. The change point is then detected when the accumulated values are above a threshold T (i.e. $y_n > T$). Let τ_T be the earliest time, after the change point m , that the accumulated value hit T . Then the only parameters required for the algorithm are the shifted mean ω , the detection delay $\tau_T - m$, and the measured α .

IV. EVALUATION RESULTS AND DISCUSSION

In this section, we describe the evaluation of our worm detection scheme, including the worm simulation and the testing procedure. We then test the effectiveness of HIWD

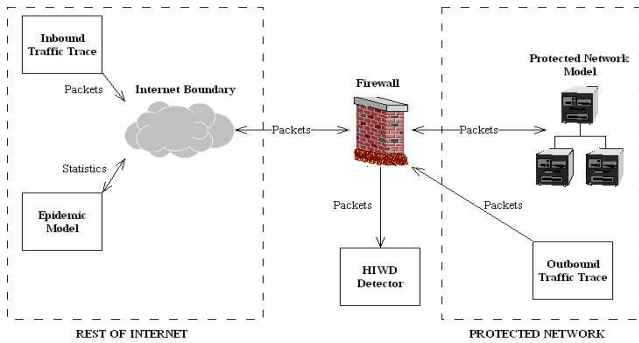


Fig. 2: Simulator schematic

in detecting worms that do not indiscriminately scan, and compare this with an earlier approach, DEWP [5]. We also provide a worst case time analysis of HIWD.

A. Hybrid Model Simulation and Testing Procedure

Our approach is to simulate worm traffic between the Internet and the protected network, as well as integrating realistic background traffic from real-life packet traces. It is based on the hybrid worm propagation model of Chen et al [5]. Figure 2 shows the general schematic of the simulator.

In our hybrid model, we separate the hosts on the protected network and the hosts in the rest of the Internet into two stratified populations. To account for the probing traffic between the two stratified populations for worm propagation, we modified the basic stratified SIR epidemic model [7].

We measure the performance of our detection scheme in terms of infection percentage (the percentage of the network infected when the first alert is raised), and the number of false positives. To examine infection percentages for the inbound History-based Detector, the background trace was switched off, as the background traffic can trigger the detectors earlier than the true detection time. For testing the false positive rate, only the background traffic was used.

The unsanitised background traffic used was the Melbourne University Computer Science traces. We chose the trace collected on June 2nd 2004 as the background traffic for testing, because it was part of the traces used to calculate the mean (May 23rd to June 6th), and this trace was not used to build the Source Address and Source Subnet databases (June 7th to June 30th). We also only chose to replay connections to destination port 80 (HTTP), as this port had the highest background traffic and most likely to cause false positives.

B. Detection of Routable and Hitlist Worms

In this section, we examine the detection performance of HIWD for the random, routable and hitlist worms. The shifted mean parameter ω of both inbound and outbound detectors were set to -2.75 and -2.0 respectively. At these values, HIWD did not produce any false positives. We will investigate the relationship between the false positive rate and the shifted mean parameter shortly.

As there are no false positives, we concentrate on the infection percentage results for the three different worm probing

methods (Figures 3a and 3b). The results indicate that the detector is effective in detecting routing and hitlist worms, which is the focus of this work. The infection percentages were higher for random scanning worms. We hypothesise this was due to the larger address space scanned by random scanning worms and the relatively low scan rates used for testing (Slammer reportedly had a scan rate of 500 scans/s [9]), so that the effective number of infected source addresses seen at the edge of the protected network is below the average. Although the infection percentages are relatively high for the random scanning worm, HIWD can be complimented by detection schemes that rely on scans to unused addresses [17], which are effective against random scanning worms, but not hitlist worms.

C. Comparison with DEWP

Recall DEWP [5] is another localised detection scheme that monitors for surges in the number of unique addresses seen and combines inbound and outbound evidence. As a comparison, we implemented the DEWP scheme, and followed Chen et al's suggestions for parameter values - detection interval of 1s, measurement interval of 8s, $\alpha = 0.125$.

We examine the relationship between the sensitivity parameter δ , false positives and infection percentage. DEWP generated an alert as soon as there was one infection in the protected network, for $\delta \leq 30$ scans/s. This was for random, routable and hitlist scanning worms. The false positive rate was less promising. Figure 3c shows the relationship between δ and false positives at a worm scan rate of 20 scans/s. The relationship is reaching an asymptotic lower bound of 2000 false positives. That is extremely high, and would infer the scheme cannot distinguish between worm and normal traffic.

D. False Positives and Infection percentage

In this section, we examine the false positive rate and the infection rate observed as we vary the shifted mean parameter ω of the CUSUM algorithm in the input and output detectors. We also examine the effect of using both the input and output detectors together, and discuss the effect of the source address databases on false positives.

First, we test the false positive rate of the History-based Detector, by turning the Scan Detector off. When the Scan Detector is off and the inbound mean is low, the number of false alerts is high (see Figure 4a). However, when the History-based Detector is combined with the Scan Detector (Figure 4b) and the outbound mean is set to 2.0, all false positives are eliminated. So for low number of false positives, low mean values are required. Now consider the effect of varying the inbound mean on infection percentages, when the outbound detector is off. As Figure 4c shows, as the inbound mean increases, the infection percentage also increases, which indicates it is not possible to have both low false positive rates and infection percentages when only one detector is used. To achieve this, both detectors of HIWD are needed.

In addition, we tested the effectiveness of our scheme without using the source address databases. Due to the lack

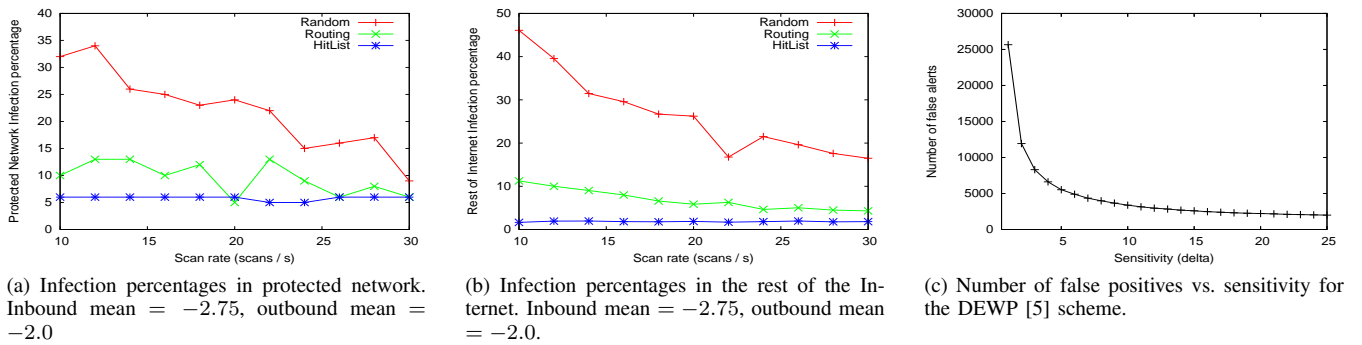


Fig. 3: Infection percentages for HIWD (Inbound: detection time = 3, outbound: detection time = 2), and false positive rates for DEWP.

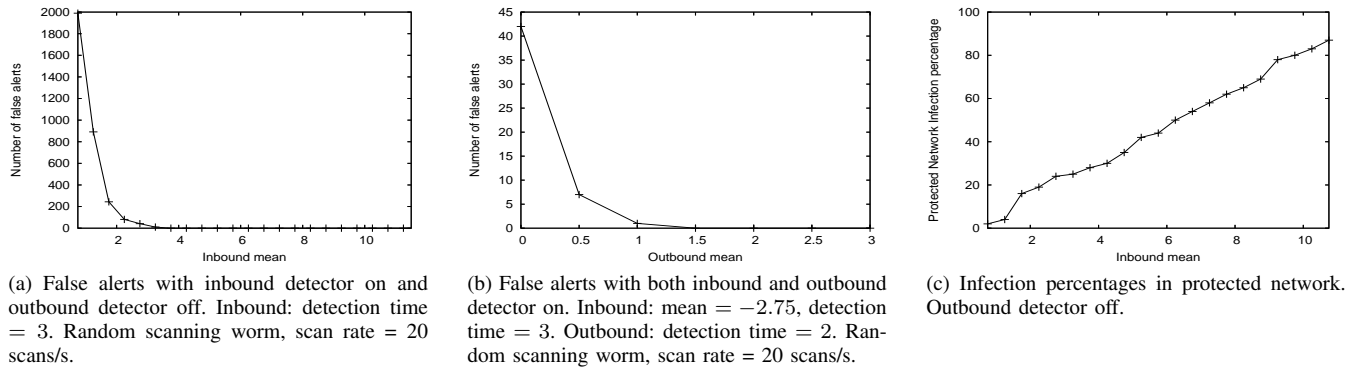


Fig. 4: Effect of varying the shift mean parameter ω on the false positives rates and the infection percentages.

of space, we do not present the results, but will state that the infection rate of all three types of worms increased by at least 10% when the databases were not used (for the same false positive rates).

E. Complexity Analysis

In this section, we analyse the worst case time complexity of HIWD. The running time is dominated by the searching time of the address databases. For the databases, we used hashing with ordered buckets and binary search to implement them. Let N_{SAD} and N_{SSD} denote the number of addresses and subnets in the databases, respectively. Then, the worst case time complexity of HIWD is $O(\log_2 N_{SAD}) + O(\log_2 N_{SSD})$. As it can be seen, even when using simple data structures, our scheme has low worst case computational requirements.

V. CONCLUSION

In this paper, we have presented a new worm detection scheme, *History-based IP Worm Detection*. History-based IP Worm Detection is based on the idea that the source address of worm scans are distributed widely and unlikely to have been previously seen at the network before, while legitimate users of a network form a stable group of seen addresses. Combined with a simple outbound scan detector, we have shown this detection feature allows our scheme to detect routing and hitlist worms quickly, while having low false positive rates. We have also presented a complexity analysis,

showing our scheme to have low processing requirements. The History-based IP Worm Detection scheme provides a robust approach to detecting both existing and new generations of worm attacks.

REFERENCES

- [1] S. Antonatos, P. Akritidis, E. Markatos, and K. Anagnostakis. Defending against hitlist worms using network address space randomization. In *WORM*, 2005.
- [2] M. Basseville and I. V. Nikiforov. *Detection of abrupt changes: theory and application*. Prentice-Hall, Inc., 1993.
- [3] F. Buchholz, T. Daniels, J. Early, R. Gopalakrishna, R. Gorman, B. Kuperman, S. Nystrom, A. Schroll, and A. Smith. Digging for worms, fishing for answers. In *Proceedings of the 18th Annual Computer Security Applications Conference*, 2002.
- [4] J. Chan, C. Leckie, and T. Peng. Detection and simulation of internet worms. <http://www.cs.mu.oz.au/~jkcchan/research/HIWD.pdf>.
- [5] X. Chen and J. Heidemann. Detecting early worm propagation through packet matching. Technical Report TR-2004-585, ISI, USC, 2004.
- [6] B. Chun, J. Lee, and H. Weatherspoon. Netbait: a distributed worm detection service. Technical Report TR-03-033, Intel Research Berkeley, 2003.
- [7] D.J. Daley and J.M. Gani. *Epidemic modelling : an introduction*. Cambridge University Press, New York, 1999.
- [8] J. Jung, B. Krishnamurthy, and M. Rabinovich. Flash crowds and denial of service attacks: Characterization and implications for cdns and web sites. In *WWW*, 2002.
- [9] D. Moore, V. Parson, S. Savage, C. Shannon, Stuart Staniford, and Nicholas Weaver. Inside the slammer worm. *IEEE S&P*, (1), 2003.
- [10] J. Newsome, B. Karp, and D. Song. Polygraph: Automatically generating signatures for polymorphic worms. *IEEE S&P*, 2005.
- [11] T. Peng, C. Leckie, and K. Ramamohanarao. Proactively detecting DDoS attack using source IP address monitoring. In *Networking*, 2004.

- [12] C. Shannon and D. Moore. The spread of the witty worm, 2004. <http://www.caida.org/analysis/security/witty>.
- [13] S. Singh, C. Estan, G. Varghese, and S. Savage. The earlybird system for real-time detection of unknown worms. Technical Report CS2003-0761, UC, San Diego, 2003.
- [14] S. Staniford, V. Paxson, and N. Weaver. How to own the internet in your spare time. In *11th USENIX Security Symposium*, 2002.
- [15] Y. Tang and S. Chen. Defending against internet worms: A signature-based approach. In *INFOCOM*, 2005.
- [16] N. Weaver, S. Staniford, and V. Paxson. Very fast containment of scanning worms. In *13th USENIX Security Symposium*, 2004.
- [17] J. Wu, S. Vangala, L. Gao, and K. Kwiat. An effective architecture and algorithm for detecting worms with various scan techniques. In *NDSS Symposium*, 2004.
- [18] V. Yegneswaran, P. Barford, and J. Ullrich. Internet intrusions: global characteristics and prevalence. In *ACM SIGMETRICS*, 2003.
- [19] C. C. Zou, D. Towsley, W. Gong, and S. Cai. Routing worms: A fast, selective attack worm based on IP address information. Technical Report TR-03-CSE-06, U. Massachusetts, Amherst, 2003.