Data Reduction for the Scalable Automated Analysis of Distributed Darknet Traffic

Michael Bailey
University of Michigan
mibailey@eecs.umich.edu

Evan Cooke
University of Michigan
emcooke@umich.edu

Farnam Jahanian
University of Michigan
Arbor Networks, Inc.
farnam@umich.edu

Niels Provos
Google, Inc.
provos@google.com

Karl Rosaen
University of Michigan
krosaen@umich.edu

David Watson
University of Michigan
dwatson@eecs.umich.edu

Abstract

Threats to the privacy of users and to the availability of Internet infrastructure are evolving at a tremendous rate. To characterize these emerging threats, researchers must effectively balance monitoring the large number of hosts needed to quickly build confidence in new attacks, while still preserving the detail required to differentiate these attacks. One class of techniques that attempts to achieve this balance involves hybrid systems that combine the scalable monitoring of unused address blocks (or darknets) with forensic honeypots (or honeyfarms). In this paper we examine the properties of individual and distributed darknets to determine the effectiveness of building scalable hybrid systems. We show that individual darknets are dominated by a small number of sources repeating the same actions. This enables source-based techniques to be effective at reducing the number of connections to be evaluated by over 90%. We demonstrate that the dominance of locally targeted attack behavior and the limited life of random scanning hosts result in few of these sources being repeated across darknets. To achieve reductions beyond source-based approaches, we look to source-distribution based methods and expand them to include notions of local and global behavior. We show that this approach is effective at reducing the number of events by deploying it in 30 production networks during early 2005. Each of the identified events during this period represented a major globally-scoped attack including the WINS vulnerability scanning, Veritas Backup Agent vulnerability scanning, and the MySQL Worm.

1 Introduction

Networks are increasingly subjected to threats that affect the reliability of critical infrastructure. These include Distributed Denial of Service attacks, such as the SCO DDoS attacks [2], and scanning worms, such as CodeRed [33] and Blaster [3]. The impact of these threats is profound, causing disruptions of real world infrastructure [26] and costing individual institutions hundreds of thousands of dollars to clean up [11]. To address the concerns raised by these threats, researchers have proposed a variety of global early warning systems whose goal is to detect and characterize these threats.

Unfortunately, the properties of these threats make them particularly difficult to address. First and foremost, they are globally scoped, respecting no geographic or topological boundaries. For example, at its peak, the Nimda worm created 5 billion infection attempts per day, which included significant numbers from Korea, China, Germany, Taiwan, and the US [33]. In addition, they can be exceptionally virulent and can propagate to the entire population of susceptible hosts in a matter of minutes. This was the case during the Slammer worm, in which the majority of the vulnerable population (75K+ susceptible hosts) was infected in less than 30 minutes [21]. This virulence is extremely taxing on network resources and creates side effects that pose problems even for those that are outside of the vulnerable population, such as the routing instability associated with the Slammer worm [17]. To make matters worse, these threats have the potential to be zero-day threats, exploiting vulnerabilities for which no signature or patch is available. For example victims of the Witty worm were compromised via their firewall software the day after a vulnerability in that software was publicized [31].

In order to address these properties, threat detection and classification systems are needed to provide detailed forensic information on new threats in a timely manner. As many threats propagate by scanning the IPv4 address space, researchers have turned to monitoring many addresses at the same time in order to quickly detect these threats [33, 32]. By monitoring large numbers of addresses, these systems can notably increase the probability of quickly detecting a new threat as it attempts to infect other hosts on the Internet [22]. However, as threats become increasingly complex, interacting with the infected hosts to ellict the important threat features, such as exploit, rootkits, or behavior,

may require increasingly complex host emulation. This, coupled with the possibility of zero-day threats that may provide little or no warning for creating these emulated behaviors, may leave wide addresses monitoring systems unable to identify the important threat characteristics. In contrast, honeypot systems provide detailed insight into new threats by monitoring behavior in a controlled environment [5, 34]. By deploying honeypot systems with monitoring software, one can automatically generate detailed forensic profiles of malicious behavior [16]. Unfortunately, this detailed analysis comes at the expense of scalability, and hence time to detection.

An interesting potential approach to this problem is to forward requests destined to darknets back to an automated bank of honeypots [15, 38, 30]. While this architecture provides the promise of quickly generating detailed forensic information, there are still serious problems that need to be addressed. In particular, there is still the very important question of what requests to forward to the honeypots. For example, a darknet consisting of an unused /8 address block (roughly 16 million routable IP addresses) observes almost 4 Mbits/sec of traffic, much of which is TCP connection requests. While seemingly a small amount of traffic, each of these connection requests may require their own virtual machine. This load can cause scalability issues for both servicing the large number of requests and storing or evaluating the large quantity of data produced [38].

In this paper, we investigate the problem of filtering darknet traffic in order to identify connections worthy of further investigation. In particular, we analyze data from a large, distributed system of darknet monitors. We characterize the traffic seen by these monitors to understand the scalability bounds of a hybrid monitoring system that consists of distributed darknet monitors and a centralized collection of honeypots (or honeyfarm). In addition, we use these characterizations to guide the design of an algorithm that is effective at reducing large traffic rates into a small number of manageable events for the honeyfarm to process.

The main contributions of this work are:

- **Measurement and analysis of a large, distributed dark address monitor.** The measurements and characterizations presented in this paper are from a multi-year deployment of over 60 darknets in 30 organizations including academic institutions, corporations, and Internet service providers. This deployment represents a tremendous amount of diverse address space including over 17 million routeable addresses with blocks in over 20% of all routed /8 networks.

- **Identification of several key threat characteristics that bound the scalability of a hybrid system.** The scalability of a hybrid system depends on limiting the number of connections that need to be sent to the honeyfarm for analysis. By examining the behavior of threats at a large number of darknets we note two important characteristics:

  - A small fraction of the total source IPs observed at a single darknet are responsible for the overwhelming majority of the packets.
  - Most behavior consists of sources, and to some extent target services, that are not observable across darknets.

  From these characterizations we show that source-based filtering is an effective method of reduction for individual darknets, but fails to provide additional benefits when multiple darknets are combined together.

- **Creation and deployment evaluation of an effective algorithm for scaling hybrid architectures.** We create an algorithm that is both very effective in reducing the large amount of traffic seen by darknets to a small handful of events and is easily within the capabilities of the most modest honeyfarms. A broad production deployment of this algorithm over a three month period in 2005 provided analysis of five major global events, including the MySQL Worm and the scanning associated with the WINS vulnerability, as well as the Veritas Backup vulnerabilities.

The remainder of this paper is structured as follows: We begin by reviewing the related work in section 2. In section 3 we introduce our hybrid architecture and some of the challenges in building any similar system. We then examine the behavior of threats at individual dark address blocks in section 4. We observe these threats across darknets in section 5, and based on the insights from these measurements, we construct a filtering algorithm that we describe in section 6. In section 7 we show how this algorithm is effective at reducing large traffic rates to a small handful of events through a broad production deployment. We then finish with our conclusions in section 8.

2 Related Work

Historic approaches to the detection and characterization of network-based security threats fall into two categories; monitoring production networks with live hosts and monitoring unused address space (or darknets). In monitoring used networks, systems may choose to watch traffic directly [20] or watch abstractions of the data, such as flow records [13]. Security devices also provide an important source of these abstractions, and alerts from host-based anti-virus software [6], intrusion detection systems [10], and firewalls have been used as an effective means of addressing certain security threats. In contrast to monitoring live hosts, darknet monitoring consists of sensors that

monitor blocks of unused address space. Because there are no legitimate hosts in a darknet, any observed traffic destined to such darknet must be the result of misconfiguration, backscatter from spoofed source addresses, or scanning from worms and other network probing. Methods for watching individual addresses with sacrificial hosts are often called honeypots [5, 34]. Techniques for monitoring much wider address blocks have a variety of names including network telescopes [22], blackholes [33], and darknets [8]. It should be noted that a limitation of this approach is that it relies on observing the target selection behavior of threats. As a result, threats that do not scan for new hosts, or threats that are specifically tailored to avoid unused blocks are not observed. Nevertheless, these techniques have been used with great success in observing denial of service activity [23], worms and scanning [32], as well as other malicious behavior.

In isolation, these techniques fail to completely address the scalability and behavioral fidelity requirements needed to monitor these threats. The scope of existing host-based techniques, such as host-based honeypots, anti-virus software, and host-based intrusion detection, is too small to capture global information such as the size of the infected population, or provide warning early in the growth phases. On the other hand, globally-scoped network sensors, such as network telescopes, do not interact sufficiently with the worm. As such, they lack enough information to characterize the vulnerability exploited and its effect on local machines. To be effective at assuring the availability of Internet resources, it is necessary to combine information from disparate network resources, each with differing levels of abstraction, into one unified view of a threat.

Acknowledging this need for both host and network views, two new approaches of combining these resources have evolved, aggregating fine-grained sensor measurements from a large numbers of sensors, and hybrid systems that use darknet monitors to concentrate connections to a centralized honeyfarm. Projects that aggregate data fall into two categories, those based on aggregating firewall logs or Intrusion Detection System (IDS) alerts across multiple enterprises [37, 36, 41], and those based on constructing and aggregating data from large numbers of honeypots [35, 25]. Hybrid systems [15, 38, 30] vary in how they perform the physical connection funneling, where and in what way they choose to filter the data, and in the diversity and amount of address space monitored. Of particular relevance is the recent work on the Potemkin Virtual Honeyfarm [39] in which the authors discuss a hybrid architecture with emphasis on a novel set of techniques for creating scalable per connection virtual machines. Their scalability gains are achieved by multiplexing across idleness in the network and by exploiting redundancies in the per-host state of the virtual machines. The gains reported vary widely based on work load (from a few hundred to a million destination IPs per physical host) but under realistic workloads a single physical host can support 50k-100k destinations IPs. This work is complimentary to ours in that we focus on limiting the number of connections seen by the honeyfarm while the Potemkin authors focus primarily on servicing these connections as efficiently as possible.

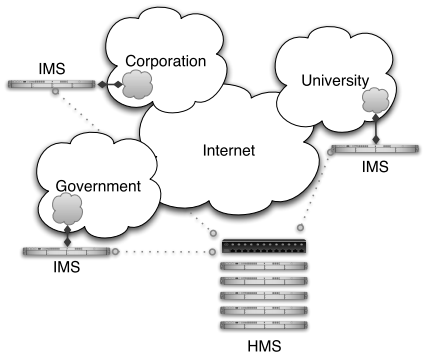

Figure 1: A Hybrid architecture with the distributed Internet Motion Sensor (IMS) with the centralized Host Motion Sensor (HMS).

3 A Hybrid Architecture for Monitoring Internet Security Threats

To provide both behavioral fidelity and global, broad coverage, we have proposed a hybrid architecture that is highly scalable but still delivers very accurate detection. This multi-resolution, hybrid architecture (shown in Figure 1) consists of two components: a collection of distributed darknet sensors (the Internet Motion Sensor or IMS), and a collection of host sensors (the Host Motion Sensor or HMS). In this architecture the IMS is used to monitor a broad, diverse set of darknets. When new activity is detected by the IMS, the connection is proxied back to the HMS for further in-depth analysis. The connection is relayed to virtual machine images running the application appropriate to the connection request. Thus, new and important threats are handed off and actually executed so the resulting activity can be monitored for new worm behavior.

By watching darknets, the traffic seen by the Internet Motion Sensor is pre-filtered to eliminate both the false positives in identifying malicious traffic and the scaling issues of other monitoring approaches. To analyze this traffic, the IMS sensors have both active and passive components. The active component responds to TCP SYN packets with a SYN-ACK packet to elicit the first data payload on all TCP streams. When a packet is received by a darknet sensor, the passive component computes the hash of the payload. If the hash doesn't match any previously observed signatures, then the payload is stored and the signature is added to the signature database. The Internet Motion Sen-

sor architecture was introduced in 2005 [2], and it has been used to track new threats [3, 1] and to evaluate the importance of distributed darknet monitoring [7].

The Host Motion Sensor (HMS) is designed to provide additional forensic analysis in response to the changing threat landscape. It consists of three components: a virtual machine management module, a network detection module, and a host resource module. The virtual machine management module runs the target operating systems on top of a virtual machine. This module determines when to start a clean OS image, when to save an infected host image to disk, and it manages the working set of applications and operating systems. The network module, like the outbound connection profiling HoneyStat system [9], is responsible for looking for infected host behavior. It profiles the originating traffic from the honeypot and alerts on any outbound connection attempt not part of its normal behavior. Publicly available intrusion detection software [28] that matches traffic against known malicious signatures is used as an oracle for classifying any newly detected threats as "known" or "unknown". The final module is the host resource profiler. This system uses BackTracker [16] to build file and process dependency trees to both detect violations of policy and provide detailed forensic information on the files and processes created by an possible infection. Again, existing techniques are used to help identify new activities, in this case host anti-virus software [6].

There are several research problems associated with this or any hybrid approach that must be solved in order to ensure successful operation. These include:

- **Filtering Interactions.** Large darknets see staggering amounts of traffic that can not simply be redirected to a honeyfarm. In order to achieve scale, the amount of data presented to the honeyfarm must be significantly reduced.

- **Emulating Real Host Behavior.** As threats become increasingly sophisticated, detection systems must become correspondingly complex to elicit the proper response and to avoid fingerprinting. Understanding this arms race and the associated tradeoffs is key to any successful deployment.

- **Automating Forensics.** At the speed new threats are capable of spreading, the operational impact of human-scaled forensic analysis is minimal. Automated techniques are needed for generating actionable forensic information about a threat, including behavioral signatures describing activity at the network and/or host levels.

- **Managing Virtual Machines.** Even with advanced filtering mechanisms, the virtual machines are expected to handle a large number of requests and must be managed efficiently (as is discussed in the Potemkin Virtual Honeyfarm [39]). In addition, the effectiveness of the entire system is dependent on its ability to accurately represent the vulnerable population of interest.

In this section we discussed both the IMS and HMS components of our hybrid monitoring system as well as several of the research problems associated with any such hybrid system. In the next section, we focus on one of these research problems, reducing and filtering the interactions between the darknets and the honeyfarm.

4 Hybrid Scalability at Individual Darknets

For hybrid systems to be effective, they must make intelligent decisions about what darknet traffic to send to the honeyfarm. An idealized mechanism achieves scale by reducing redundant information and only forwarding one instance of each unique threat to the honeyfarm. Unfortunately, the mechanisms for determining what to handoff must make these decisions with the imperfect information available at a darknet monitor. In order to minimize overhead, a darknet monitor typically collects packets passively. As such, it has available elements from the network packet including the standard 5-tuple (source IP addresses, source port, destination IP address, destination port, and protocol), as well as any payload data seen for UDP, ICMP, or TCP backscatter packets. As mentioned previously, the IMS darknet sensor also collects the first payload packet for TCP connections through a scalable responder [2]. While other methods have been proposed for eliciting additional application information (for example, by building application responders via Honeyd [25]), in this paper we fix the data available for determining what to handoff and leave the issue of exploring more sophisticated information sources for future work.

In the following section, we explore the characteristics of these six elements (the five tuple and initial payload data) in order to determine how they affect the scalability of a hybrid system at individual darknets. We begin by examining the properties of individual darknets and in particular the behavior of source IP addresses. We provide these characterizations by looking at data from 14 darknet monitors ranging in size from a /25 monitor to a /17 monitor over a period of 10 days between August 18, 2004 and August 28, 2004. We then use these characterization to examine the effectiveness of proposed filtering techniques in reducing the connection which need to be evaluated by the honeyfarm.

4.1 Characterizing Individual Blocks

We begin by evaluating the source IP addresses seen at each darknet as a mechanism for determining bounds on the number of unique connections to be evaluated. As with

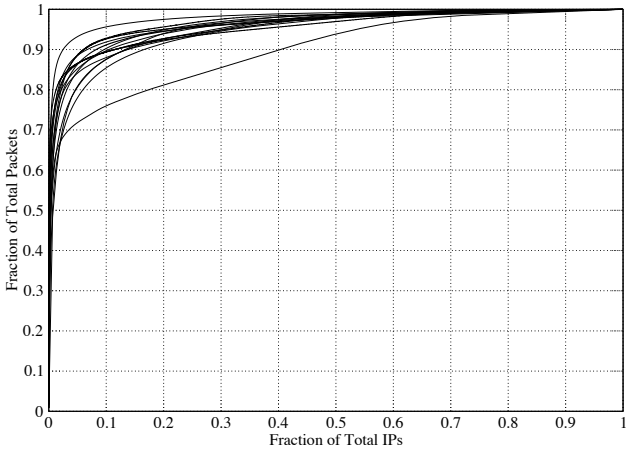

Figure 2: The contribution of individual IP to the total number of packets as seen at 14 darknets. Over 90% of the packets are from 10% of the source IP addresses.

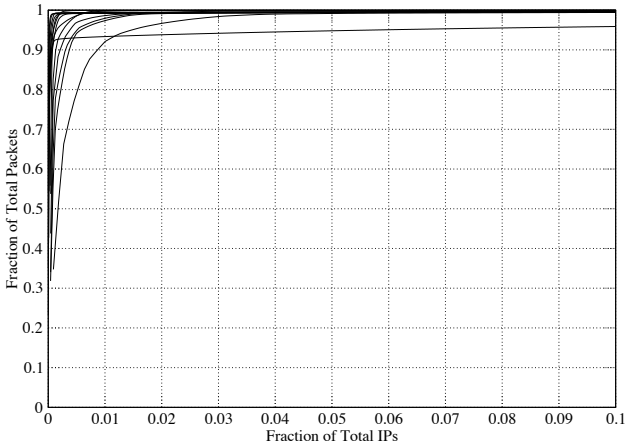

Figure 3: The contribution of a port to the total number of packets as seen at 14 darknets. Over 90% of the packets target .5% of the destination ports.

all the imperfect methods available at a darknet, source IP addresses have limitations in their ability to represent the number of unique attack sources. First, IP addresses do not represent individuals, as multiple users may use the same computer. Second, the mapping from IP address to computer is not static, so a computer may be represented multiple times with different IP addresses [32]. As in other studies [32, 24], we attempt to minimize these effects by performing analysis across small time frames of less than a day, or more often, less than an hour. However, the actual impact of dynamic addresses is not studied here.

The number of source IP addresses seen at each individual darknet varied greatly, with an inter-block mean of 75,530 sources and a variance of 92,843 sources over the 10 days. The minimum number observed in a single day was 1,345 sources with a maximum of 352,254 sources. Some of the wide variability in sources seen can be attributed to the effect of monitored darknet size. In our sample we had a mean block size of 5,385 IPs (roughly a /22), with the smallest darknet being a /25 and the largest a /17. However, even when normalizing to the smallest block size of /25, we have an inter-block mean of 40,540 sources and a variance of 30,381.

To understand how these source IP addresses behave, we examined the number of packets sent by each source IP address over the 10-day observation period at each of the 14 darknets. Figure 2 shows a surprising distribution: over 90% of the total packets observed at each darknet were sent from less than 10% of the source IP addresses seen at that darknet.

The other property that limits the number of unique connections to be evealuated is the number and types of services contacted. To explore this space, we examined the unique destination ports contacted over our 10-day observation period. As with sources, ports are imperfect measures. In particular, destination port analysis suffers from the inability to differentiate activities to the same port and to represent combinations of port activities (as is the case in multi-vector attacks) into a single action. Nevertheless these, serve as a coarse-grained approximation sufficient for exploring the size of the destination service space.

In our analysis we again see a great deal of variability based on darknet size, with a mean number of contacted ports of 17,780 and a variance of 20,397. The minimum number of ports seen was 1,097 at a /25 and the maximum was 59,960 at a /17. With maximums approaching the total number of destination ports allowed, we conjecture that many of the ports observed are simply due to ports scans. Nevertheless, unless the scanning is uniform in some way (e.g., sequential) it would be difficult for the darknet monitors to treat these packets differently. To understand the role these diverse destination ports play in darknet traffic, we investigated the distribution of these destination ports and their effect on the number of packets. Again we see a very striking result: over 90% of the packets are from .5% of the destination ports.

Despite the focused distributions, the cross product of the total unique source IP addresses and total destination ports is actually quite large. In order for us to efficiently scale, the source IP addresses must repeat similar actions. We therefore look at the behavior of the top 10% source IP addresses in terms of the number of destination ports they contact as well as the number of unique payloads they send. Figure 4 shows the number of ports contacted by 10% of the IP addresses at each of 14 darknets over a period of 10 days. At each darknet, over 55% of these source IP addresses contacted a single destination port, 90% contacted less than six, and 99% of the source IP addresses contacted less than 10. A very small fraction of these very active source IP addresses contacted considerably more ports. As

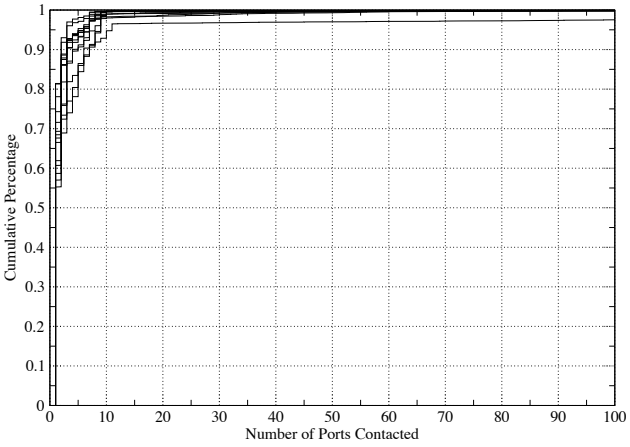

Figure 4: For the top 10 percent of IPs seen at each of the 14 darknets, the cumulative distribution function of the number of ports contacted.

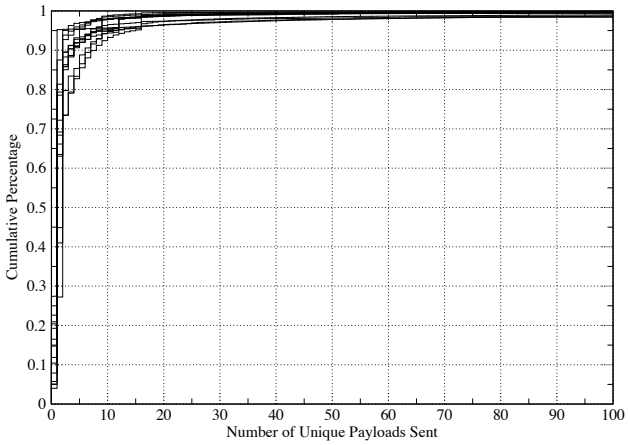

Figure 5: For the top 10 percent of IPs seen at each of the 14 darknets, the cumulative distribution function of the number of unique payloads sent.

expected, the fanout in the payloads sent is slightly larger, with 30% sending only one payload, 70% sending two or less, and 10 or less payloads seen by only 92%. In this analysis only the first payload of an action is considered. While a better differentiator of threats than ports, it may still under-represent the total number of events, as many services (e.g., Windows RPC) require multiple identical initiation packets. Nevertheless, both methods show that a significant fraction of the behaviors are the same for a source IP address, with the vast majority of the attacks involving multiple destination ports (multi-vector) and consisting of less than 10 contacted destination ports.

In this section we explored the source IP address behavior as seen by 14 individual darknets. We showed that a small fraction of the total source IP addresses are responsible for the vast majority of packets and that these sources

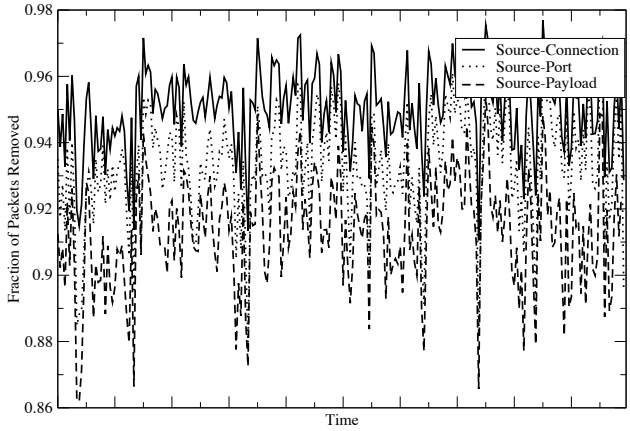

Figure 6: The reduction of Source-Connection, Source-Port, and Source-Payload filtering. Average effectiveness per hour for 14 darknets over a 10-day period with N=1.

are contacting the same, small handful of destination ports repeatedly. In the next section, we examine the effect of these results on several source-based filtering techniques and evaluate the practical value of applying these techniques to reduce packets seen at individual dark address blocks into a smaller number of unique events.

4.2 Source-Based Filtering

Recently a small body of work has emerged on filtering of the darknet traffic as a means to scale to large address blocks. This work has been published in the context of the iSink [38] project as well as the Internet Motion Sensor [2]. In the iSink work [38] the authors discuss two forms of filtering, random subnet selection and a sample and hold method. In subsequent work [24], the authors introduce four types of source-based filtering: source-connection, source-port, source-payload, and source-destination. The tradeoffs are discussed for each, including the effect of multi-stage (multiple connections to the same port) and multi-vector (multiple connections to different ports) based attacks on their accuracy. However, only source-connection is evaluated, and only at two darknets for a two-hour trace. The IMS authors have also discussed [2] preliminary results in using the contents of the first payload packets to reduce disk utilization and differentiate traffic.

The effect of three of the source-based methods is shown over a 10-day period at 14 darknets in Figure 6. In source-connection filtering, N connections from a single source are recorded, and all subsequent traffic from that source is ignored. Source-connection filtering serves as a baseline for determining the maximum reduction in any of the source-based approaches, as each contains the source as a component and is therefore limited to the number of unique

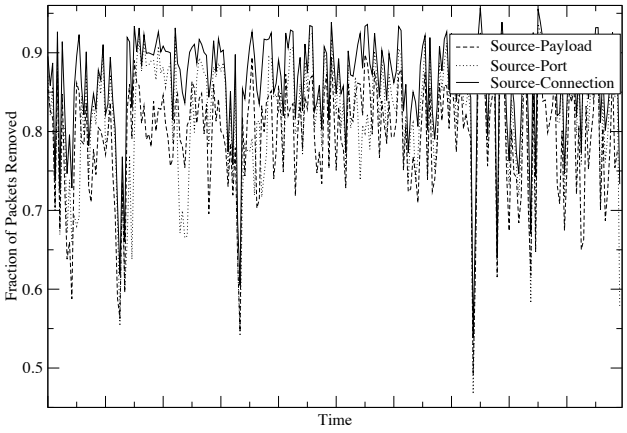

Figure 7: The minimum reduction of Source-Connection, Source-Port, and Source-Payload filtering. Minimum effectiveness per hour for 14 darknets over a 10-day period with N=1.

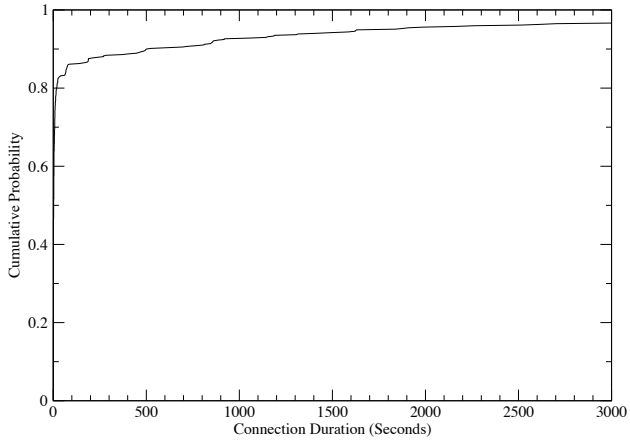

Figure 8: The cumulative distribution function of connection length from a Windows 2000 honeypot over a three-day period

sources in a period. In source-port filtering, N connections are maintained for every source and destination port pair. This method [24] eliminates the undercounting of events of source-connection filtering in the case of multi-vector activities, but multi-stage activities remain a problem, as connections to the same port may not be visible with a small number of N (e.g., N=1 will only record the first connection to a port). Finally, we have source-payload filtering in which the first N payloads sent by a source are used to define all future behavior by that source. We find that the least differentiating view of an event is seen with the source-connection filter. Because it counts any traffic from the same source as the same, it is undercounting the number of real events and therefore has the greatest reduction, a mean of 95%, across blocks. The more restrictive source-port filtering results in a mean reduction of 93% of the packets. Finally, the most differentiating view of events, source-payload, showed a mean reduction of 91%. On the whole, source-based techniques appear effective at reducing packets to a smaller number of events.

In comparing these results with the source-destination results reported previously [24] we see less effectiveness than the 99% reduction reported for values of $N = 1$, with even the least restrictive methods. While there are several possible explanations for this discrepancy, such as the difference in darknet block size (the blocks considered in [24] are /16s) we also report a great deal of variance, not only between darknets, but in a darknets over time. The intra-hour standard deviation for source-connection reduction was 2.7%, with source-port and source-payload at 3.1% and 3.9% respectively. Perhaps of more interest is the minimum value during each bin, as this is the run-time value that a hybrid filtering system would have to contend with. We show the minimum values in Figure 7. Here the minimum values drop to as low as 53.8% for source-connection filtering, and 46.7% and 49.1% for source-port and source-payload. In practice, the reductions observed may be less when applied to a runtime system that is making decisions about reduction and handoff based on the current observation period.

4.3 Effects on Hybrid Scalability

In order to understand the effect of source-based filtering on reducing the number of connections processed by a simple hybrid system, we considered the behavior at a single darknet. We applied source-payload filtering to the largest darknet over our 10-day observation window, a /17 darknet, and counted the unique source-payload events in one second bins. This analysis was designed to measure the total number of events per second a hybrid system consisting of a single darknet would be expected to handle. With averages less than 100, the system can expect to see bursts of several times that, with a single burst above 500 events per second being reported in the 10-day period.

Recall that the purpose of the hybrid system was to identify new threats and receive detailed analysis of those threats. As such, care must be taken to separate the cause of a specific event on the honeypots from its effect (for example, If a honeypot is processing several simultaneous connections, how can do we know which one successfully caused the infection?) In the worst case, we may need to separate every event and send it to a separate virtual host for processing. In this case, we now become bound not simply by the event per second rate but also by the duration of the event. To evaluate the cost of event duration, we examined the lengths of connections to an individual honeypot host, as shown in Figure 8. While roughly 77% of the connections were of zero length (a connection request and

no more), the remaining distribution is very heavy tailed, with an average connection length of 400 seconds. In combination, these results indicate that a honeyfarm for a single /17 darknet block would need to handle from 40,000 to 200,000 simultaneous connections.

In the previous sections we explored the attack characteristics, as observed by individual darknets including the number of unique source addresses and destination ports. We examined the distribution of packets among these source IP addresses and destination ports and found that a supprisingly small number of source IP addresses and an even smaller number of destination ports dominated each darknet. We showed that these distributions make source-based filtering methods avery ppealing in reducing the traffic at individual blocks, but that there was a great deal of variance in the effectiveness of these methods over time. Nevertheless, we believe that these methods can be very helpful in reducing the number of packets at an individual darknet to a much smaller handful of connections.

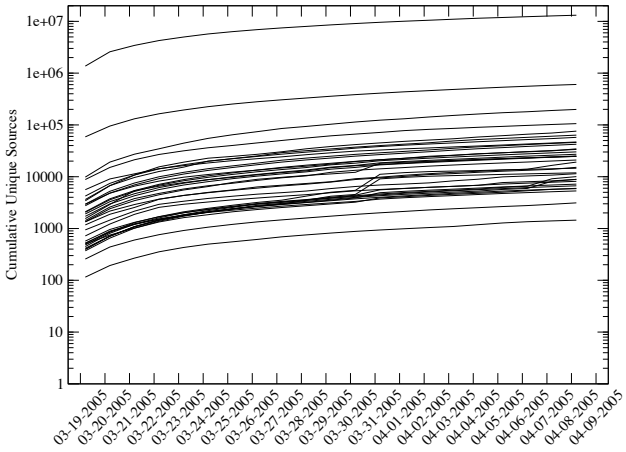

Figure 9: The number of cumulative unique sources per day, as viewed by 41 darknets from March 28th, 2005 to April 19th, 2005. Each line is a single darknet varying in size from a /25 to a /8

5 Hybrid Scalability in Distributed Dark Address Blocks

For a hybrid system to be effective, our goals of detecting new threats and providing detailed analysis of these threats must be performed quickly when a new threat emerges. While we showed in the previous section that source-based filters could produce obtainable numbers of connections for handoff, the size of these darknets (e.g., /17) may be to small to provide quick detection of scanning worms. To achieve even further detection time reductions, a darknet monitoring system can choose to monitor a larger darknet, or combine multiple, distributed darknets. In practice, however, there is a limit on the size of a darknet (e.g., /8) and few of these large darknets exist. Moving beyond that size when such a large darknet is not available requires additional darknets to be aggregated together. This distributed darknet monitoring approach also has several additional benefits, including added resilience to fingerprinting, and insight into difference between darknets. In this section, we examine the properties of source IP addresses and destination ports across darknets to determine the effects of these properties on a hybrid system consisting of a distributed collection of darknet monitors.

5.1 Characterizing Distributed Darknets

We begin by looking at the number of source IP addresses at each of the 41 darknets during a 21-day period, from March 19th through April 9th, 2005. In Figure 9, we examine the cumulative unique source IP addresses seen per day. We see that blocks receive traffic from a few hundred (the /25 darknet) to a few million (the /8 darknet) unique source IP addresses per day. There is some overlap with previous days, however, the forward difference still involves hundreds of thousands of hosts every day for the /8 darknet. This order of magnitude difference in the number of source IP addresses between the /8 and the /17 monitor discussed in the previous section adds a considerably larger number of events to evaluate for the larger darknet.

In order to see how the addition of darknets (each with their own number of unique source IP addresses over time) affects the aggregate number of sources in the hybrid monitor, we computed the overlap in unique source addresses between all darknets. Table 1 shows the average percentage of daily source IP address overlap (and standard deviation) in several of the medium to large darknets from the IMS system over a period of a month. A significant number of the source IP addresses seen at these darknets are not globally visible. The largest of the monitored darknets, the /8 darknet, consists mainly of source IP addresses not seen at any of the other darknets, as seen in the D/8 row. However, a much larger fraction of the source IP addresses at the other darknets do appear in the /8 darknet, as seen in the D/8 column. While this implies that many of the source IP address seen at a local darknet are captured by the /8, a significant faction of them are not; from 88% at one of the /22 darknets to 12% at one of the /18 darknets. In addition, the majority of the source IP addresses seen at the /8 are not seen anywhere else. This does not bode well for the scaling of a hybrid system, as the addition of each new darknet will be add a significant number of new source IP addresses and hence new connections to be evaluated.

Next we considered the space of the destination ports. For 31 darknets, we examined the top 10 destination ports, based on the number of packets, and compared these lists across darknets. Figure 10 shows the number of darknets that had a particular destination port in their top 10 list.

	A/18	B/16	C/16	D/23	D/8	E/22	E/23	F/17	G/17	H/17	H/18	H/22	I/20	I/21
A/18	100(0)	25(2)	58(5)	4(0)	78(5)	2(0)	4(0)	55(5)	28(2)	36(3)	28(3)	3(1)	16(2)	12(1)
B/16	23(3)	100(0)	38(5)	3(0)	54(8)	1(0)	3(0)	36(5)	20(3)	25(3)	18(2)	2(0)	10(1)	8(1)
C/16	23(2)	17(1)	100(0)	3(0)	78(6)	0(0)	2(0)	45(4)	20(2)	28(3)	20(1)	1(0)	9(0)	7(0)
D/23	10(0)	10(1)	20(1)	100(0)	30(1)	0(0)	1(0)	20(1)	10(0)	15(0)	11(0)	1(0)	5(0)	4(0)
D/8	2(0)	1(0)	5(0)	0(0)	100(0)	0(0)	0(0)	5(0)	1(0)	3(0)	2(0)	0(0)	0(0)	0(0)
E/22	10(2)	8(1)	13(1)	1(0)	12(1)	100(0)	3(0)	11(1)	9(1)	7(1)	6(0)	1(0)	7(0)	5(0)
E/23	25(4)	20(3)	33(5)	3(0)	34(5)	5(1)	100(0)	30(5)	21(3)	20(3)	16(2)	3(0)	16(2)	13(2)
F/17	23(1)	17(0)	48(1)	3(0)	82(1)	1(0)	2(0)	100(0)	22(0)	29(1)	21(0)	1(0)	9(0)	7(0)
G/18	20(1)	16(1)	36(1)	3(0)	51(2)	1(0)	2(0)	38(1)	100(0)	24(1)	16(1)	2(0)	9(0)	7(0)
H/17	16(0)	12(0)	31(1)	2(0)	53(1)	0(0)	1(0)	31(1)	14(0)	100(0)	27(2)	1(0)	8(0)	6(0)
H/18	20(1)	15(0)	37(2)	3(0)	56(2)	1(0)	2(0)	37(2)	16(0)	45(2)	100(0)	2(0)	11(0)	8(0)
H/22	7(2)	5(0)	9(1)	1(0)	16(3)	0(0)	1(0)	9(0)	6(0)	8(0)	6(1)	100(0)	3(0)	2(0)
I/20	11(1)	8(0)	16(1)	1(0)	16(1)	1(0)	1(0)	16(1)	8(0)	12(1)	10(0)	0(0)	100(0)	14(2)
I/21	13(1)	10(1)	20(1)	1(0)	19(1)	1(0)	2(0)	19(1)	11(1)	16(1)	12(1)	1(0)	21(4)	100(0)

Table 1: The average (and stddev) **percentage** overlap in source IP addresses between (row, column) medium to large darknets over a month period.

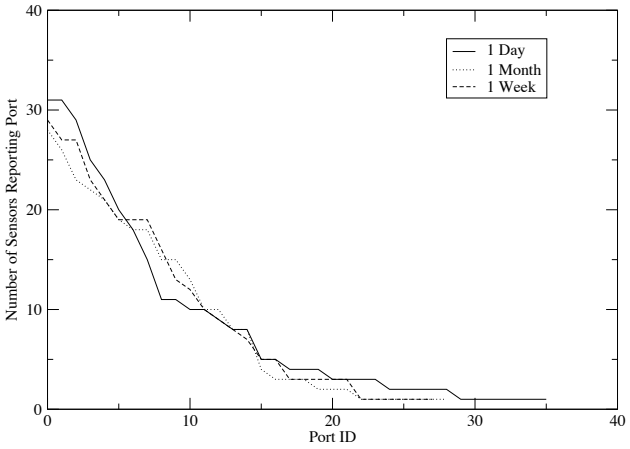

Figure 10: The number of darknets (of 31) reporting a port in the top 10 ports over a day, week, and month time frame.

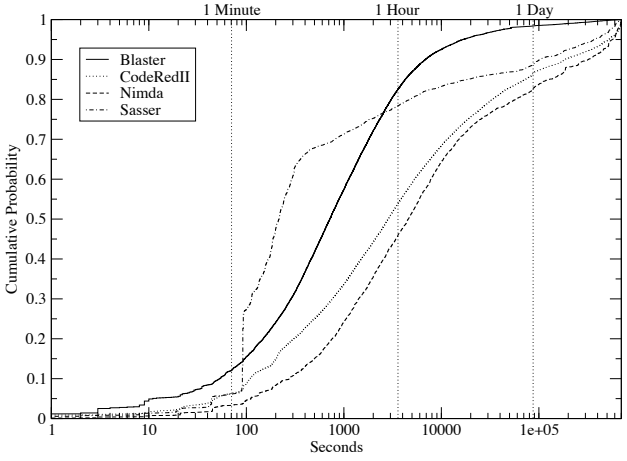

Figure 11: The duration of source IP address observations at the /8 darknet over a one week period for 4 known worms.

The analysis is performed for the top 10 destination ports over a day, top 10 destination ports over a week, and top 10 destination ports over a month. This figure shows that there are over 30 destination ports that appear on at least one darknet's top 10 list. A small handful of these destination ports appear across most darknets (1433, 445, 135, 139), but most of the destination ports appear at less then 10 of the darknets. As with the result seen with source IP addresses, the lack of consistency between darknets implies a broader number of connections to be evaluated, because of the broader number of non-overlapping destination services being contacted.

5.2 Understanding the Overlapping Size

The lack of overlap between various darknets in terms of source IP addresses, as well as destination ports, stems from four properties of the monitoring technique and the traffic.

The impact of *monitored block size, scanning rate*, and *observation time* on the probability of identifying a random scanning event. The effect of monitoring block size on the detection time of remote events is a well-studied phenomena [22]. Table 1 does show larger blocks with overlapping source IP addresses. However, the largest of these, although highly variable, only sees an average of 50% overlap.

The lifetime of the events. One possible explanation of this lack of overlap is that the events being observed are too short lived. To examine this, we looked at the behavior of four familiar worms whose (random and non-random) scanning behaviors are well known. Figure 11 shows this result. We recorded the observation lifetime of the source IP addresses across our /8 darknet. It should be noted that this method can only approximate the actual lifetime as the /8 darknet may only observe part of the worm's propagation behavior. With this caveat in mind, we note a significant faction of the source IP addresses seen at this darknet had a lifetime of less than a day. However, more than 50% had lifetimes of over an hour, which is sufficient at most scanning rates (greater than 10 connections per second) to be observed at the bigger darknets.

Targeted attacks. While the above explanations explain some of the lack of overlap, a significant number of source IPs are still not overlapping. It is our conjecture that these are representative of targeted behaviors and attacks. While this conjecture is difficult to validate without monitoring at the sources of these attacks, we can look at the distributions of destination ports for some insight as to whether the same services are being contacted across darknets. Recall that Figure 10 showed the number of darknets that report a port in their top 10 over increasing time frames of a day, week, and month. The results show that, although there are a few ports which are globally prevalent, a large number of the ports are seen over these time frames at very few blocks.

Environmental factors. In [7] we showed that a variety of factors, including filtering policy, propagation strategy, darknet visibility, and resource constraints, affect the mix of the global traffic a darknet should see. These properties provide subtle influences beyond the impact of the targeted behavior discussed above.

5.3 Effects on Hybrid Scalability

One of the drawbacks of any of the source-based techniques discussed in the previous section is their reliance on the source IP address as part of the filter. For a distributed technique based on these methods to be effective, source IP addresses must be prevalent across darknets. Unfortunately, our analysis shows this not to be the case. In order to quantify the effectiveness of source-based methods across darknets we consider the same 14 darknets as in the previous section. We choose source-port filtering and look at the reduction in unique (source, port) pairs across darknets over the same 10-day period. While this method was effective at reducing the packet space by 95%, there is only a modest 20% reduction when these methods are applied across darknets.

In this section we examined the properties of distributed darknets to understand the scaling properties of a hybrid system. We evaluated the properties of source IP addresses and destination ports across darknets and observed a significant number of non-overlapping destination ports and source IP addresses. While some of these differences are the result of well-known size and rate effects of darknet monitoring, other factors, such as the short on-time of random scanning hosts and the targeted nature of attacks observed at these darknets help to explain the additional differences. The impact of these non-intersecting events is that each individual new darknet added to a hybrid system is likely to significantly increase the total number of connections that need to be evaluated. If a hybrid system is to scale in this type of environment, a new definition of events and new methods of filtering are required.

6 Aggressive Distributed Filtering

In the previous sections, we discussed source-based filtering at individual darknets and the application of those filtering techniques to distributed darknets for the purpose of building a scalable hybrid system. We noted that source-based techniques were much less effective across darknets for two main reasons: source IP addresses are not typically visible across darknets in time frames that are useful and many attacks may be targeted at individual darknets only.

In this section we examine techniques for filtering that explicitly account for these observations. In particular, we examine the number of unique source IP addresses per destination port, and we examine the number of darknets reporting an event. We combine these variables with the technique of threshold-based alerting to provide a filter that passes traffic on global increases in the number of source IP addresses contacting a destination port.

Alerting of traffic changes by observing traffic to live hosts is a well studied area. Existing methods have alerted once traffic has reached a static threshold [27] or for thresholds that were adapted dynamically [14]. Further attempts to model traffic behavior that can be used for detecting changes include signal analysis [29], probabilistic approaches [18], and statistical approaches [12]. In this section we extend the existing techniques to look at adaptive threshold-based alerting for traffic to unused address space. Similar in vein to other source address distribution alerting mechanisms [40], we watch the distribution of unique source IP addresses to a specific port. Our work differs from this in several key ways: we modify the algorithm to explicitly allow for varying notions of current and historic behavior, we modify it to watch the distributions across multiple darknets, and we incorporate the notion of global versus local behavior.

Our current event identification scheme uses an adaptive threshold mechanism to detect new events. Every hour, each darknet is queried for the number of unique source IP addresses that have contacted it with destination port x (where x ranges over a list of destination ports we are monitoring). The event identification algorithm looks at the collected data and works as follows for each destination port x. For each hour, add up the number of unique source IP addresses contacting destination port x at each darknet. Scan over this data one hour at a time, comparing the average (per hour) over the event window (last event window hours) to the average over the history window (last event window $\times$ history factor) hours. If the ratio of event window average to history average is greater than the event threshold, an event is generated. These events are then filtered based on whether they are global or local, via the coverage threshold. The coverage threshold defines the number of darknets that would have generated an event individually for a threat. Events will not be generated more than once

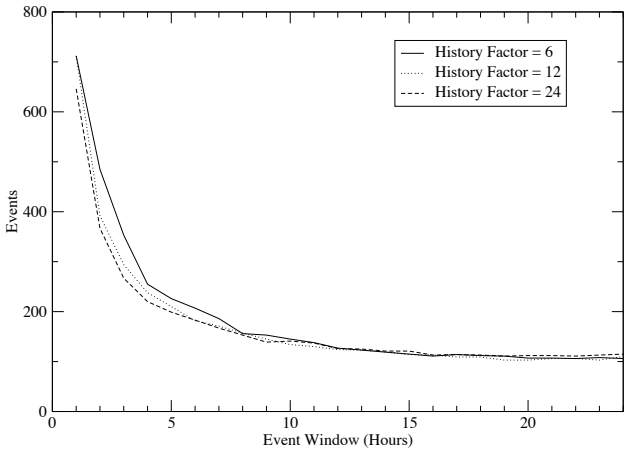

Figure 12: The effect of event window size on the number of events generated.

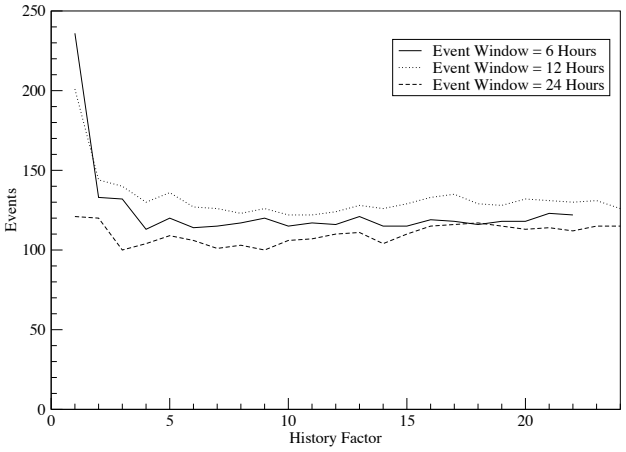

Figure 13: The effect of history factor on the number of events generated.

in a single event window. To protect against false positives on destination ports that have little or no traffic destined to them, whenever the sum of unique source IP addresses from all blocks to a specific destination port is less than one, the data point for that hour is set to 0.6. The event generation algorithm then is parameterized by four variables: the event window, the history window, the event threshold, and the coverage.

7 Evaluation and Deployment Results

In this section, we investigate the parameter space of the event identification algorithm and then evaluate it by comparing both the security events we identify and those identified by the community.

7.1 Parameterization

As discussed in the previous section, there are four basic parameters that impact the generation of a new event. In this section, we explore the tradeoffs associated with each of these parameters using data collected at 23 of the 60 IMS blocks from January 1st through April 30th, 2005.

The first parameter we explore is that of the event window. Recall that the event window defines the interval over which the current value is computed via a weighted moving average. Figure 12 shows the effect of the event window size in hours on the number of events generated for fixed window size and event threshold. The curve shows that the number of events vary from over 700 to a nerarly steady value of 100, with a significant reduction by a value of five hours. One possible explanation for this reduction is that many of the events are in fact short bursts, and longer event window sizes reduce the impact of single hour bursts.

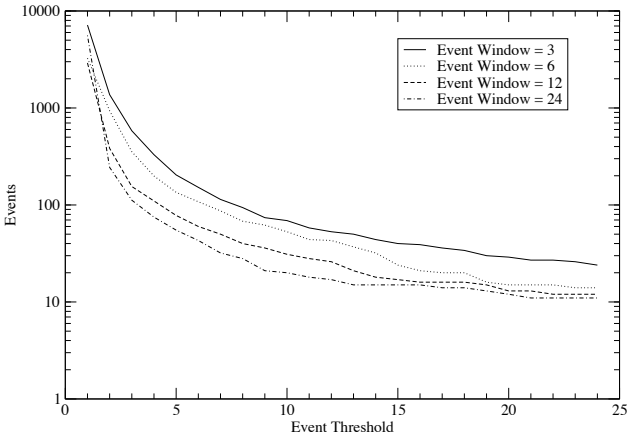

Figure 14: The effect of event threshold on the number of events generated.

History factor defines the period of time over which normal behavior is calculated via a weighted moving average. The effect of various history factor values on the number of events is explored in Figure 13. A history factor of two appears to reduce the number of events greatly. It is also interesting to note the crossover in event window size, with an event window of 12 hours creating *more* events than a smaller window of 6 hours. We find no significant difference in protocol or source distribution between these two sets. We are continuing to investigate two additional hypotheses: that sequential scanning activities need more time to reach additional darknets, and that there are frequency components of darknet traffic that are different than that of other Internet traffic.

Figure 14 shows the effect of the event threshold on the number of events. The event threshold indicates the degree to which the current value is different than the historic value. This parameter shows the largest impact on events

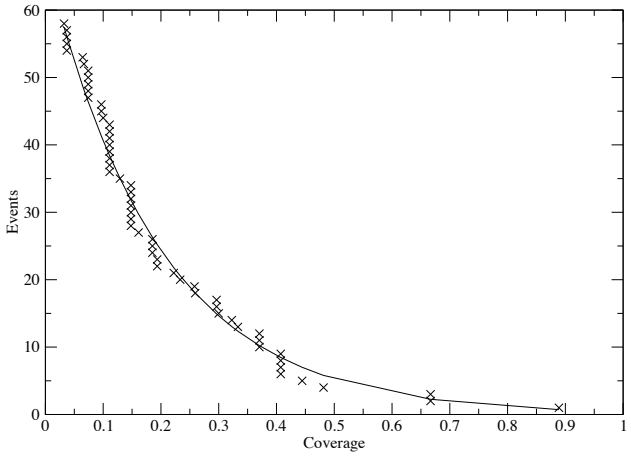

Figure 15: The effect of coverage on the number of alerts generated.

of any of the parameters, with a parameter of one generating an event for nearly every event window (event when the ratio of current to past is one). Drastic reductions in the number of events are seen by event thresholds of three to four.

Coverage represents the percentage of the total darknets monitored that would have individually reported an increase via our algorithm. Figure 15 shows the effect of various coverage values as a filter on the number of events. The majority of events generated by the algorithm are seen at a very small handful of darknets. The majority of reduction in the number events occurs when only considering events that are seen across more than 1/3 of the darknets.

7.2 Production Validation

To help validate our event generation algorithm, we compared the events identified by our system with those identified by the broader security community. It is important to highlight that our system has been in production use over the past few months and has identified several critical new threats. It is used by major network operators and governments around the world to quickly help identify new security events on the Internet.

Over an observation period of four months, the IMS system identified 13 unique events. Table 2 shows these events grouped by port. Included in these events are the TCP/42 WINS scanning, the TCP/6101 Veritas Backup Exec agent scanning, and the MySQL worm/bot on TCP/3306. These events demonstrate the quick onset, short duration, and global prevalence of many attacks. Figure 16 shows the normalized number of unique source addresses detected at the 23 IMS sensors for these events over time. As the operational test was underway before the completed parametization evaluation, the alerting algorithm used the following

Description	Port	Date	Multiple	Coverage
WINS	tcp42	01/13/05 17:31	5.36	0.4815
	tcp42	01/14/05 05:31	61.85	0.8889
	tcp42	01/14/05 17:31	9.77	0.6667
Squid and	tcp3128	02/05/05 11:31	7.73	0.4074
Alt-HTTP	tcp3128	02/05/05 23:31	18.19	0.4074
SYN Scan	tcp8080	02/05/05 10:51	7.53	0.4074
	tcp8080	02/05/05 22:51	20.95	0.3704
MYSQL	tcp3306	01/26/05 09:31	43.89	0.3704
	tcp3306	01/26/05 21:31	8.2	0.4444
	tcp3306	01/27/05 09:31	5.7	0.4074
Syn Scan	tcp5000	01/08/05 14:31	3.42	0.6667
Veritas	tcp6101	02/23/05 21:32	3.54	0.3704
	tcp6101	02/24/05 09:32	3.81	0.3333

Table 2: The interesting features identified by the algorithm since January of 2005. The "multiple" column specifies how many times larger the current window is compared to the history window. Coverage reports the percentage of sensors which would have alerted independently.

non-optimal parameters: alert window of 12 hours, history window of six days, and an alert threshold of three.

Beginning in December 2004, the IMS system observed a significant increase in activity on TCP port 42. This increased activity was notable because Microsoft had recently announced a new security vulnerability with the Windows Internet Name Service (WINS) server component of its Windows Server operating systems. The activity also followed a vulnerability report from Immunity Security on November 24, 2004 describing a remotely exploitable overflow in the WINS server component of Microsoft Windows. Payloads captured on TCP port 42 during the event include sequences that are byte-for-byte identical to exploit code found in the wild following the vulnerability announcement. Although the attack was very broadly scoped, it involved a small number of hosts and only lasted a day. Because the IMS was broadly deployed, it was able to quickly identify this small-scale event in time to enable additional monitoring capabilities.

Another quick attack occurred on January 11, 2005, when IMS observed a substantial increase in TCP port 6101 scanning. Approximately 600 sources were identified as aggressively probing many of the distributed IMS sensors. TCP port 6101 is used by the Veritas Backup Exec agent, a network service that allows for remote system backup. On December 16, 2004, iDefense announced a buffer overflow enabling remote system-level access. Then, on January 11, exploit code was published for the vulnerability and on the same day, IMS observed a large increase in activity on TCP/6101. Payloads captured from the attack include sequences that are byte-for-byte identical to the exploit code. Once again we see a very quick onset, suggesting the need for an automated system able to rapidly react to new threat activity.

Finally, in late January of 2005 IMS detected a worm/bot targeted at the MySQL database server [19]. The worm/bot propagated by randomly scanning for the Windows ver-

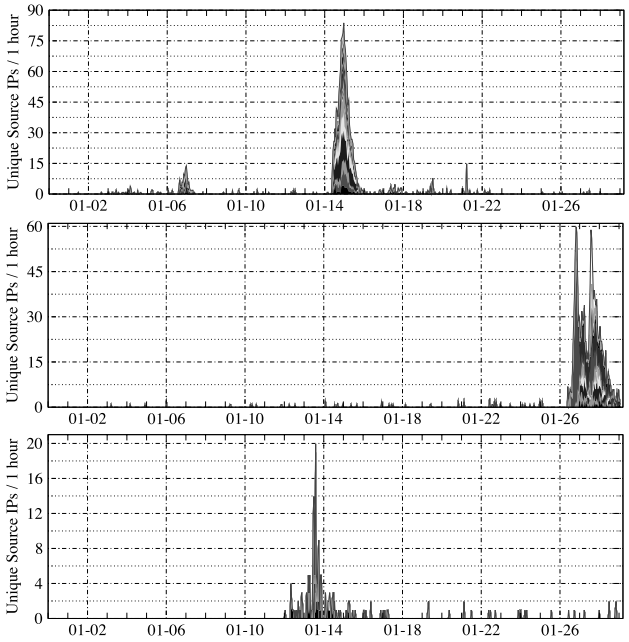

Figure 16: The unique source IP addresses over time across 23 darknets to various TCP destination ports. Widely publicized events are shown (from top to bottom TCP/42-WINS, TCP/3306-MYSQL, TCP/6101-VERITAS).

sion of MySQL on TCP port 3306. The threat was unique because it required interaction with a hidden IRC server before a new instance would start to propagate. The IRC channel was discovered by operators who shut it down, effectively eradicating the worm after a period of a few days. This event shows the importance of having more detailed information on a threat. Without observing the interaction of the worm/bot with the IRC server, there would be no way to discover the communication channel and thus find the simple way to stop the worm.

These three events help substantiate the detection approach and illustrate the value of automated distributed forensics. To further confirm these results and those listed in Table 2, we searched for correlations between the port, payloads, and time period of the events with reports from other security data sources. With the exception of the Altweb and Squid SYN scans, each of these events has been actively discussed in various security forums and corroborated by other monitoring projects [37, 4]. There is also evidence that the event identification algorithm did not miss any important events. Analysis of the NANOG, ISN, BUGTRAQ, FULL-DISCLOSURE, as well as the operator logs from ISC and CERT, do not show any major detectable events that were missed by the IMS system. In summary, there is strong evidence that the system was able to identify all the major events during the 4 month period. More time is clearly needed to fully validate the system, but the current data suggests the approach has excellent filtering qualities.

8 Conclusion

Global threats are changing at an amazing rate, requiring new innovative mechanisms for tracking and characterizing them. One approach to this problem involves hybrids of darknets with honeyfarms. There are numerous hurdles to this approach, but in this paper we investigated the most important of these, scaling interactions between the darknets and the honeyfarms. We examined several of the key characteristics of traffic at individual darknets that affect the scalability of a hybrid system. In particular, we showed that darknets are dominated by a small handful of source IP addresses contacting the same destination ports repeatedly. These characteristics allow source-based approaches to be quite effective at reducing the number of packets a block sees to a much smaller number of connections. We then showed, however, that this does not hold across darknets; neither the source IP addresses, nor to a lesser extent the destination ports, appear frequently across darknets. While some of this behavior can be explained by existing work on darknet size, scanning rate, and time to detection, we showed that much of it is the result of the targeted nature of the attacks observed as well as the short on-time of many random scanning hosts. This lack of overlap implies that every additional new block added to a distributed darknet monitor will likely bring with it its own sensor-specific events. For a large collection of darknets to be effective, a new view of events and new methods of filtering are required. We then constructed a filtering mechanism that takes these two factors into account by including distributions of source IP addresses, rather than the source IP addresses specifically, and the number of darknets observing an activity. We showed that this algorithm is effective by examining several recent events, such as activity associated with new WINS vulnerability, Veritas vulnerabilities, and the recent MY-SQL worm.

Acknowledgments

This work was supported by the Department of Homeland Security (DHS) under contract number NBCHC040146, the Advanced Research and Development Activity (ARDA) under contract number NBCHC030104, and by corporate gifts from Intel Corporation and Cisco Corporation. The authors would like to thank all of the IMS participants for their help and suggestions. We would also like to thank Joseph Rock, Jenn Burchill, Jose Nazario, Dug Song, Robert Stone, and G. Robert Malan of Arbor Networks and Larry Blunk, Bert Rossi, and Manish Karir at Merit Network for their assistance and support.

References

[1] Michael Bailey, Evan Cooke, Tim Battles, and Danny McPherson. Tracking global threats with the Internet Motion Sensor. 32nd Meeting of the North American Network Operators Group, October 2004.

[2] Michael Bailey, Evan Cooke, Farnam Jahanian, Jose Nazario, and David Watson. The Internet Motion Sensor: A distributed blackhole monitoring system. In *Proceedings of Network and Distributed System Security Symposium (NDSS '05)*, San Diego, CA, February 2005.

[3] Michael Bailey, Evan Cooke, David Watson, Faranam Jahanian, and Jose Nazario. The Blaster Worm: Then and Now. *IEEE Security & Privacy*, 3(4):26–31, 2005.

[4] Lawrence Baldwin. My Net Watchman - Network Intrusion Detection and Reporting. `http://www.mynetwatchman.com/`, 2005.

[5] Bill Cheswick. An evening with Berferd in which a cracker is lured, endured, and studied. In *Proceedings of the Winter 1992 USENIX Conference: January 20 — January 24, 1992, San Francisco, California*, pages 163–174, Berkeley, CA, USA, Winter 1992.

[6] Fred Cohen. *A Short Course on Computer Viruses*. John Wiley & Sons, 2nd edition, April 1994.

[7] Evan Cooke, Michael Bailey, Z. Morley Mao, David Watson, and Farnam Jahanian. Toward understanding distributed blackhole placement. In *Proceedings of the 2004 ACM Workshop on Rapid Malcode (WORM-04)*, New York, October 29 2004. ACM Press.

[8] Team CYMRU. The darknet project. `http://www.cymru.com/Darknet/index.html`, June 2004.

[9] David Dagon, Xinzhou Qin, Guofei Gu, Julian Grizzard, John Levine, Wenke Lee, and Henry Owen. Honeystat: Local worm detection using honeypots. In *Recent Advances in Intrusion Detection, 7th International Symposium, (RAID 2004)*, Lecture Notes in Computer Science, Sophia-Antipolis, French Riviera, France, October 2004. Springer.

[10] Stephanie Forrest, Steven A. Hofmeyr, Anil Somayaji, and Thomas A. Longstaff. A sense of self for Unix processes. In *Proceedings of the IEEE Symposium on Research in Security and Privacy*, pages 120–128, Oakland, CA, May 1996.

[11] Andrea L. Foster. Colleges Brace for the Next Worm. *The Chronicle of Higher Education*, 50(28), 2004.

[12] Joseph L. Hellerstein, Fan Zhang, and Shahabuddin Shahabuddin. A statistical approach to predictive detection. *Computer Networks (Amsterdam, Netherlands: 1999)*, 35(1):77–95, January 2001.

[13] Cisco Systems Inc. Netflow services and applications. `http://www.cisco.com/warp/public/cc/pd/iosw/ioft/neflct/tech/napps_wp.htm`, 2002.

[14] Chuanyi Ji and Marina Thottan. Adaptive thresholding for proactive network problem detection. In *Third IEEE International Workshop on Systems Management*, pages 108–116, Newport, Rhode Island, April 1998.

[15] Xuxian Jiang and Dongyan Xu. Collapsar: A VM-based architecture for network attack detention center. In *Proceedings of the 13th USENIX Security Symposium*, San Diego, CA, USA, August 2004.

[16] Samuel T. King and Peter M. Chen. Backtracking intrusions. In *Proceedings of the 19th ACM Symposium on Operating Systems Principles (SOSP'03)*, pages 223–236, Bolton Landing, NY, USA, October 2003. ACM.

[17] Lad, Zhao, Zhang, Massey, and Zhang. Analysis of BGP update surge during slammer worm attack. In *International Workshop on Distributed Computing: Mobile and Wireless Computing, LNCS*, volume 5, 2003.

[18] C. Leckie and R. Kotagiri. A probabilistic approach to detecting network scans. In *Proceedings of the Eighth IEEE Network Operations and Management Symposium (NOMS 2002)*, pages 359–372, Florence, Italy, April 2002.

[19] Robert Lemos. MySQL worm halted. `http://ecoustics-cnet.com.com/MySQL+worm+halted/2100-7349_3-5555242.html`, January 2005.

[20] G. Robert Malan and Farnam Jahanian. An extensible probe architecture for network protocol performance measurement. In *Proceedings of ACM SIGCOMM*, pages 177–185, Vancouver, British Columbia, September 1998.

[21] David Moore, Vern Paxson, Stefan Savage, Colleen Shannon, Stuart Staniford, and Nicholas Weaver. Inside the Slammer worm. *IEEE Security & Privacy*, 1(4):33–39, 2003.

[22] David Moore, Colleen Shannon, Geoffrey M. Voelker, and Stefan Savage. Network telescopes. Technical Report CS2004-0795, UC San Diego, July 2004.

[23] David Moore, Geoffrey M. Voelker, and Stefan Savage. Inferring Internet denial-of-service activity. In *Proceedings of the Tenth USENIX Security Symposium*, pages 9–22, Washington, D.C., August 2001.

[24] Ruoming Pang, Vinod Yegneswaran, Paul Barford, Vern Paxson, and Larry Peterson. Characteristics of Internet background radiation. In *Proceedings of the 4th ACM SIGCOMM conference on Internet measurement*, pages 27–40. ACM Press, 2004.

[25] Niels Provos. A Virtual Honeypot Framework. In *Proceedings of the 13th USENIX Security Symposium*, pages 1–14, San Diego, CA, USA, August 2004.

[26] Reuters. Bank of America ATMs disrupted by virus. `http://archive.infoworld.com/togo/1.html`, January 2003.

[27] S. Robertson, E. Siegel, M. Miller, and Stolfo Stolfo. Surveillance detection in high bandwidth environments. In *Proceedings of the Third DARPA Information Survivability Conference and Exposition (DISCEX 2003)*. IEEE Computer Society Press, April 2003.

[28] Martin Roesch. Snort — lightweight intrusion detection for networks. In USENIX, editor, *Proceedings of the Thirteenth Systems Administration Conference (LISA XIII): November 7—12, 1999, Seattle, WA, USA*, Berkeley, CA, USA, 1999. USENIX.

[29] Amos Ron, David Plonka, Jeffery Kline, and Paul Barford. A signal analysis of network traffic anomalies. *Internet Measurement Workshop 2002*, August 09 2002.

[30] Stefan Savage, Geoffrey Voelker, George Varghese, Vern Paxson, and Nicholas Weaver. Center for Internet epidemiology and defenses. NSF CyberTrust Center Proposal, 2004.

[31] Colleen Shannon and David Moore. The spread of the Witty worm. *IEEE Security & Privacy*, 2(4):46–50, July/August 2004.

[32] Colleen Shannon, David Moore, and Jeffery Brown. Code-red: a case study on the spread and victims of an Internet worm. In *Proceedings of the Internet Measurement Workshop (IMW)*, December 2002.

[33] Dug Song, Rob Malan, and Robert Stone. A snapshot of global Internet worm activity. FIRST Conference on Computer Security Incident Handling and Response, June 2002.

[34] Lance Spitzner. *Honeypots: Tracking Hackers*. Addison-Wesley, 2002.

[35] Lance Spitzner et al. The honeynet project. `http://project.honeynet.org/`, June 2004.

[36] Symantec Corporation. DeepSight Analyzer. `http://analyzer.securityfocus.com/`, 2005.

[37] Johannes Ullrich. DSHIELD. `http://www.dshield.org`, 2000.

[38] Paul Barford Vinod Yegneswaran and Dave Plonka. On the design and use of Internet sinks for network abuse monitoring. In *Recent Advances in Intrusion Detection—Proceedings of the 7th International Symposium (RAID 2004)*, Sophia Antipolis, French Riviera, France, October 2004.

[39] Michael Vrable, Justin Ma, Jay Chen, David Moore, Erik Vandekieft, Alex C. Snoeren, Geoffrey M. Voelker, and Stefan Savage. Scalability, fidelity and containment in the potemkin virtual honeyfarm. In *To appear in Proceedings of the 20th ACM Symposium on Operating System Principles (SOSP)*, Brighton, UK, October 2005.

[40] J. Wu, S. Vangala, L. Gao, and K. Kwiat. An effective architecture and algorithm for detecting worms with various scan techniques. In *Proceedings of the Symposium on Network and Distributed Systems Security (NDSS 2004)*, San Diego, CA, February 2004. Internet Society.

[41] Vinod Yegneswaran, Paul Barford, and Somesh Jha. Global intrusion detection in the DOMINO overlay system. In *Proceedings of Network and Distributed System Security Symposium (NDSS '04)*, San Diego, CA, February 2004.