One Size Does Not Fit All: 10 Years of Applying Context-Aware Security

Sushant Sinha, Michael Bailey, Farnam Jahanian
Computer Science and Engineering
University of Michigan
Ann Arbor, MI 48109.
{sushan, mibailey, farnam}@umich.edu

Abstract

Defenders of today's critical cyber-infrastructure (e.g., the Internet) are equipped with a wide array of security techniques including network-based intrusion detection systems (IDS), host-based anti-virus systems (AV), and decoy or reconnaissance systems such as host-based honeypots or network-based telescopes. While effective at detecting and mitigating some of the threats posed to critical infrastructure, the ubiquitous nature of malicious activity (e.g., phishing, spam, DDoS) on the Internet indicates that the current deployments of these tools do not fully live up to their promise. Over the past 10 years our research group has investigated ways of detecting and stopping cyber-attacks by using the ***context*** *available in the network, host, and the environment. In this paper, we explain what exactly we mean by context, why it is difficult to measure, and what one can do with context when it is available. We illustrate these points by examining several studies in which context was used to enable or enhance new security techniques. We conclude with some ideas about the future of context-aware security.*

1 Introduction

Internet has significantly simplified public communication and collaboration. As a result, individuals now use it for a variety of activities that including shopping, banking, reading blogs, and social networking. Many of these activities require end users to divulge personal information, which is then automatically stored and processed by remote software services. With the increasingly ubiquitous nature of the Internet in individual's lives, the vulnerabilities in software have become an easy medium for information theft and abuse. There is no doubt that threats in the form of worms, viruses, botnets, spyware, spam, and denial of service [2, 5, 27, 13] are rampant on today's Internet.

To counter these threats, a number of systems have been developed to protect host and network resources. One approach that has gained significant popularity in recent years is the use of network-based security systems. These systems are deployed on the network for threat monitoring, detection, and mitigation. They are easy to deploy and require little modifications to the end hosts. These include anomaly detection systems, intrusion detection and prevention systems (e.g., Snort [19]), honeypot systems (e.g., honeyd [18]), and spam detection systems (e.g., SpamAssassin [12]).

While network-based security systems themselves have improved considerably over time, their deployment model in most networks continue to take a *"one-size fits all"* approach. These systems are typically viewed as generic solutions and they do leverage the contextual information available in the networks to customize their deployment. Unfortunately, this information may be critical to the performance and accuracy of these systems as the networks they are deployed in differ significantly with each other in terms of policy, the topological layout, the vulnerability landscape, the exploits observed, the traffic characteristics, etc.

Many network-based security systems acknowledge the need to adapt to the network. However, such adaptation is often decided in an ad hoc fashion or left to be manually configured. For example, honeypot systems like honeyd [18] come with a default configuration file for the operating system and vulnerability configuration of the honeypots. While such systems can be manually configured by a network administrator, the scale of configuration and the diversity among different networks make it very challenging. For example, configuring honeynet in a network may require one to come up with operating system and application configuration for thousands of hosts. In addition, the diversity among networks make it difficult for people to share their configurations and mitigate this effort.

Over the last several years, our group's thesis has been that the automatic adaptation to the network context will significantly improve the performance and accuracy of network-based security systems. The general idea of adding

Operating	Networks			
System	A/16	B/16	C#1/17	D#1/19
Windows	44	25	76	77
Cisco IOS	14	7	-	-
Apple	9	36	-	-
Linux	9	7	15	6
HP printer	3	13	-	-
Solaris	9	7	1	2
*BSD	1	-	8	15

Operating Systems

TCP	Networks			
Port	A/16	B/16	C#1/17	D#1/19
139	42	17	-	-
22	41	53	30	25
135	39	10	42	69
23	27	34	4	5
445	27	11	-	-
80	21	26	93	96
25	12	10	70	83
21	8	24	77	79
427	4	26	-	-
497	3	28	-	-
110	1	-	39	17

TCP ports

Table 1. Comparing the vulnerable population in four networks, by operating systems and TCP ports(from [21]). Different networks have different vulnerability profiles.

context to computing so that they can better serve human need has received significant attention from the academic community. As explicitly adding context for each user is too cumbersome, most of these efforts have tried to leverage user group activities to automatically infer the context. In the mobile computing community, context in the form of location, role and time has been used to automatically adapt a mobile device [16]. For example, a cell phone can vibrate instead of ringing if it knows that a person is in a meeting. This automatic adaptation reduces human computer interaction and makes it easier to manage a mobile device. Search engine technologies have leveraged user click through to determine the context for a query and improve search ranking [11]. These techniques do not require each user to provide explicit feedback but automatically determine the context for a query by exploiting the large amount of user data.

In this paper, we explore different types of context information that are not currently used by current security solutions and show how such information can be used to improve the performance and accuracy of these systems. We begin in section 2 by explaining what we mean by context, why the measurement of context is difficult, and what can be accomplished with context if it can be captured. In section 3, we highlight several examples of our previous work which quantify the effects of context, utilize different forms of context to improve performance and accuracy, and aggregate context to solve difficult security problems. We conclude in section 4 with a discussion of interesting future directions for context-aware security.

2 Context

Context can mean a variety of things to computer scientists, from one's physical location, to one's current desktop environment, the task being performed, etc. We begin, therefore, by providing a definition of what we mean by context and more specifically, context-aware security. We show that the properties important to context-aware security are, in fact, non-trivial to measure. However, we argue that, if they can be measured, context can be used in a variety of ways that improve performance and accuracy of existing and new security applications.

2.1 What is security context?

Short of unplugging our computers from the network and locking them in a room, there is no absolute security. At its most fundamental level, then, security is a risk analysis activity in which practitioners decide what they wish to protect from whom and at what cost. The key to understanding these tradeoffs are three properties, which we define to make up a network's security context:

- **Vulnerability profile:** The vulnerability profile represents the space of all possible targets and ideally all methods of unauthorized access to those services. In the traditional sense, this is a mapping between the device (i.e., machine), operating system, applications, and the list of known vulnerabilities for each. More broadly, this encompasses unknown vulnerabilities in server software and the social engineering path for access acquisition in client software.

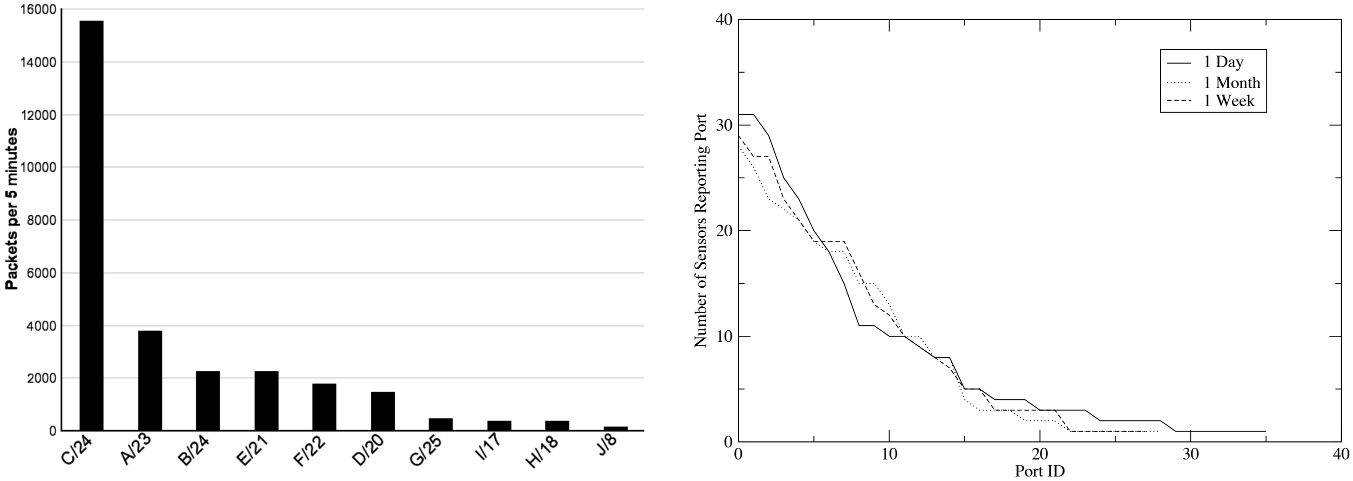

Table 2. (Left) Packet rate as seen by each sensor normalized by /24 (from [7].) (Right) The number of darknets (of 31) reporting a port in the top 10 ports over a day, week, and month time frame (from [4].). Different networks have different attack surfaces.

- **Attack surface:** The attack surface represents the unique threats posed by attackers to the defenders of a particular network. In a traditional sense, it is a measure of the remote network exploits (either attempted or successful) directed at a particular network. In a broader sense, it encompasses the notion of who the attackers are, what resources they are interested in, and the current techniques for acquiring those resources. For example, while a network might run a large number of (potentially vulnerable) printer services, attackers may avoid these services due to their uniqueness (and hence difficulty in exploiting), as well as the limited value in compromising them. Of course, other attackers may feel just the opposite about having access to printed documents–the context matters.

- **Usage model:** While the attack surface helps prioritize the potential targets specified in the vulnerability surface by defining what the attackers are interested in and the current tools used to achieve them, the usage model helps defenders prioritize the importance of the services on the network. This prioritization can be as simple as defining what services are most used on a network, but may layer in notions of data importance, disclosure liability, opportunity costs on failure in availability, etc.

2.2 Why is context hard?

In the previous section, we defined security context to include a network's attack surface, vulnerability profile, and usage model. Unfortunately, obtaining this context for most networks is not a trivial exercise. In this section, we examine the two main hurdles for context measurement: the diversity among networks and the dynamic nature of context.

2.2.1 One size does not fit all

As network and security practitioners, it should come as no surprise that different networks exhibit different traits or characteristics. What we have found during our research, however, is that these differences are surprisingly large, pervasive, and have significant impacts on all aspects of the security in an organization. For example, consider the issue of an organization's vulnerability profile. Table 1 compares the vulnerable population in these of four networks networks in two ways: by the operating system and the TCP ports. Of the four production networks (A/16, B/16, C/17 and D/19), two of these networks (A/16 and B/16) are academic networks and two (C/17 and D/19) are web server farms. The second largest operating system in network A/16 is surprisingly Cisco IOS, which is found in the wireless access points and routers in the academic campus. On the other hand, Apple Mac OS is the dominant operating system in network B/16. As expected, the web-server farms were dominated by Windows servers. While SSH seems to be predominant service found in A/16 and B/16, HTTP, FTP and SMTP seems to be the dominant services in the web server farms. Therefore, the vulnerability profile may differ significantly depending on the network.

Hospital			Library Regional Network			Government			Small College		
APPL.	IN	OUT	APPL.	IN	OUT	APPL.	IN	OUT	APP.	IN	OUT
RTSP	96.25	1.91	HTTP	17,590	2,200	HTTP	1,390	231.43	HTTP	58,420	13,710
DOMAIN	1.69	1.85	HTTPS	651.6	116.39	SSH	11.85	195.87	FLASH	4,080	84.75
HTTPS	3.49	.001	FLASH	706.02	13.13	HTTPS	108.49	98.11	HTTPS	1,280	1,400
SMTP	1.99	1.32	TCP/81	16.41	287.16	ESP	98.97	32.64	XBOX	1,010	1,610
LOTUS NOTES	1.73	.158	SMTP	100.83	166.51	SMTP	55.72	73.00	UNIDATA-LDM	947.91	950.88

Table 3. The network application usage (Kbps) at 4 different networks. Different networks have different usage models.

Today's attacks are global and everyone see the "same stuff" right? Unfortunately, the threat landscape also differs significantly for different networks. Cooke *et al.* [7] monitored unused address spaces (darknets) in different networks. Since unused addresses do not have any legitimate services, the traffic directed to these addresses are suspicious. Figure 2 shows the packet rates observed by different sensors and normalized by /24 address range. It shows that some networks receive significantly more suspicious traffic than others. In Bailey *et al.* [4], we examined, for 31 darknets, the top 10 destination ports, based on the number of packets, and compared these lists across darknets. Figure 2 shows the number of darknets that had a particular destination port in their top 10 list. 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. Not only are there more or less attacks based on where you are, those attacks are targeting different services as well.

The traffic characteristics of a network may also be significantly different than others. For example, the list of IP addresses that legitimately access services on a given network may be different from other networks. Similarly, HTTP may be the most prominent protocol in web server farms and SMTP may be the most prominent protocol in a mail service provider network. Consider the data in Table 3. While some applications (e.g., web) are global popular, many are not (e.g., Lotus Notes, XBox). Note also the stark differences in the magnitude of traffic as well as the different behavior as either servers (e.g., high in bound traffic) or clients (i.e., high outbound traffic) of a particular service.

2.2.2 Dynamic Nature of Context

Another interesting observation of our work is that these unique individual contexts are highly dynamic in nature. Attack surfaces, usage models, and even vulnerability profiles change rapidly as new attacks are released, flash crowds are formed, or new application emerge. As an example, consider Table 4 which shows the top five TCP ports and the number of packets observed over a day for five months on the a /24 darknet in the B/16 network. We find that new services were targeted heavily each month. The TCP ports 6000 and 1080 were the unusual ones targeted in April, the TCP port 5000 was targeted in May, the TCP ports 22 and 5900 were targeted in June, and TCP port 4444 in July. The highly variable nature of this threat landscape makes chasing exploits difficult for the defenders, who must adjust their vision of the attack surface to today's or this week's most popular attacks.

2.3 What do we do with context?

In the previous section, we argued that our elements of context-aware security (attack surface, vulnerability profile, and usage model) are in fact difficult to measure due to their dynamic nature and the large degree of diversity in each across networks. This argued strongly for automated techniques for capturing and updating the unique security context for individual organizations, but such mechanisms are only useful if the security context can be applied to do something useful. Perhaps the most straight forward application is in the area of configuration. Most modern security devices (e.g., IDS, Firewalls, anti-virus software) have a bewildering array of configuration options, from the mundane display and alerting features, to update frequency, sensitivity parameters, and detection techniques. Setting these options correctly and optimizing them for the individual security context of a network is one such interesting application.

Perhaps slightly less obvious is the application of these methods to the placement of security devices within a network. Many network centric security devices rely on monitoring or shaping traffic as seen on a particular network or connection. As modern networks become increasingly flattened and porous, the placement of these devices becomes less and less obvious (i.e., the death of the DMZ and the walled garden) and the need for informed placement increasingly important. While configuration and placement are interesting applications of context, they still treat an individual security device as a black box.

One of the most interesting applications of context is to modify the black box itself to suit the context of the network. For example, by understanding the unique way in which users make use of network resources, we can reorder rule evaluation to save space, time, and increase accuracy. We can suggest that a security detection device currently operating at the host might best be used in the cloud, while a network policy device, might be be used on the host. Configuration, placement, modifying key assumptions are just some of the applications of context we have explored, and many other such applications certainly exist. In the next section, we examine some examples of our work in this area.

3 Examples

The emergent definition of context and its application areas in the previous section were the direct result of 10 years of research in security and distributed systems in our research group. In this section, we highlight some of the more interesting classes of projects including: quantifying the scope of context, applying different forms of context, and aggregating multiple contexts to solve difficult problems.

3.1 Quantifying Context

As network-based threats become increasingly prominent, characterizing, monitoring, and tracking these threats is critical to the smooth running of individual organizations and to the Internet as a whole. To increase their view of these threats, researchers and network operators started instrumenting unused address space. Because there are no legitimate hosts in an unused address block, traffic must be the result of misconfiguration, backscatter from spoofed source addresses, or scanning from worms and other network probing. The most common application of this technique is the global announcement and routing of unused space to a collection infrastructure that records incoming packets [14]. Using these techniques, researchers have successfully characterized and classified the traffic observed at unused blocks [15].

In this work [7, 3] we demonstrated that achieving a representative sample may not be as simple as monitoring a few unused address blocks. To better understand how observed traffic is affected by sensor placement, we used data from the Internet Motion Sensor [8]. The Internet Motion Sensor (IMS) is a distributed collection of blackhole sensors. These sensors are deployed in networks belonging to service providers, large enterprises, and academic institutions representing a diverse sample of the IPv4 address space. We presented empirical evidence that different address blocks observe significantly different traffic volumes and patterns(e.g., Table 2(left)). This evidence was then combined with additional experimentation to build a list of sensor properties providing plausible explanations for these differences. Using these properties, we concluded with recommendations for understanding the implications of sensor placement.

3.2 Applying Different Forms of Context

Applying allocation policy This vast pool of unallocated, unrouted, and unassigned addresses sitting idle across the Internet can be used to provide intelligence on malicious and misconfigured Internet activity [17]. There are a range of techniques for monitoring contiguous ranges of unused addresses, including honeypots [1, 25, 26], virtual honeypots [4, 10, 28], emulators [18, 29], simple responders [3], and passive packet capture [9, 20]. We refer to these techniques together as *honeynet* monitoring.

Existing honeynet monitoring systems only cover a very small percentage of the available unused address space. Two fundamental problems limit monitoring more addresses. First, address allocation information is distributed across many devices, applications, and administrative domains. For example, address registries like ARIN can provide information on what addresses are assigned to an organization, but not on what addresses are routed or reachable. The second challenge is that address allocations can change quickly. For example, wireless devices can enter and leave a network, and instability in routing information can impact address reachability. The result is that honeynet monitoring systems today monitor only easily obtainable, contiguous blocks of addresses.

This work [6] presented an architecture that automated the process of discovering these non-productive addresses by participating directly with allocation, routing, and policy systems. The goal was to pervasively discover unused and unreachable ("dark") addresses inside a network so that traffic sent to those addresses can be forwarded to honeynet monitoring systems. To demonstrate our approach, we constructed the *Dark Oracle*, a system designed to discover unused and unreachable addresses within a network. The system integrated external routing data like BGP, internal routing data like OSPF, and host configuration data like DHCP server logs to construct a locally accurate map of dark addresses. The Dark Oracle automated address discovery, significantly simplifying the process of finding dark addresses. It also provided unique local visibility into internal threats and targeted attacks. We deployed a pervasive honeynet detector that uses the addresses from the Dark Oracle and showed how unused addresses from a DHCP server reveal almost 80,000 unique source addresses compared to

04/19/2006	05/19/2006	06/19/2006	07/19/2006	08/19/2006
6000	445	22	135	1433
445	139	5900	80	1080
1433	5000	3128	4444	445
1080	1433	8080	445	5900
135	80	80	1433	1521

Table 4. The top 5 TCP ports observed in a /24 sensor in network B/16, over a period of 5 months (from [21]). The attack surface changes quickly over time.

4,000 found by a traditional /24 monitor. Because we were also able to monitor outgoing addresses, we discover almost 2,000 locally infected or misconfigured hosts in an academic network. These experiments demonstrated the effectiveness of the Dark Oracle in discovering highly distributed local and global dark addresses, thereby enabling quick detection of targeted and internal attacks.

Applying work load Intrusion detection and prevention systems take a set of signatures and detect intrusions by matching them with network traffic. Existing approaches to signature evaluation apply statically-defined optimizations that do not take into account the network in which the IDS or IPS is deployed or the characteristics of the signature database. In this work [24] we argued that for higher performance, IDS and IPS systems should adapt according to the workload, which includes the set of input signatures and the network traffic characteristics.

We developed an adaptive algorithm that systematically profiles attack signatures and network traffic to generate a high performance and memory-efficient packet inspection strategy. We implemented our idea by building two distinct components over Snort: a profiler that analyzes the input rules and the observed network traffic to produce a packet inspection strategy, and an evaluation engine that pre-processes rules according to the strategy and evaluates incoming packets to determine the set of applicable signatures. We have conducted an extensive evaluation of our workload-aware Snort implementation on a collection of publicly available datasets and on live traffic from a border router at a large university network. Our evaluation shows that the workload-aware implementation outperforms Snort in the number of packets processed per second by a factor of up to 1.6x for all Snort rules and 2.7x for web-based rules with reduction in memory requirements. Similar comparison with Bro shows that the workload-aware implementation outperforms Bro by more than six times in most cases.

Applying vulnerability profiles A Honeynet is a collection of sacrificial hosts explicitly deployed to be scanned, compromised, and used in attacks. Honeynets have recently become popular to detect and characterize threats such as worms, botnets, and malware. Unfortunately, existing approaches to deploying honeynets largely ignore the problem of configuring operating systems and applications on individual hosts, leaving the user to configure them in a manual and often ad hoc fashion. In this work [21], we demonstrate that such ad hoc configurations are inadequate: they misrepresent the security landscape of the networks they are trying to protect and are relatively easy for attackers to discover. Therefore, a honeynet configuration should take the deployment context i.e., the network in which it is deployed to provide visibility into attacks and resistance to fingerprinting.

We showed that manually building honeynet configurations for each network is hard, as each network has its own unique threat and vulnerability spaces, and the potential number of hosts to configure in the honeynet is quite large. We argued that honeynets with individually consistent hosts and proportional representation of the network will achieve the two desired goals of visibility into network attacks and resistance to discovery. We developed an automated technique based on profiling the network and random sampling to generate these honeynet configurations. Through experimental evaluation and deployment of configurations generated by our technique, we demonstrated significantly more visibility and higher resistance to discovery than current methods.

3.3 Aggregating context

Aggregating to solve global problems In order to address globally scoped Internet threats, threat detection and classification systems are needed to provide detailed forensic information on new threats in a timely manner. In this work bailey:2005:filter, we investigated the problem of filtering darknet traffic in order to identify connections worthy of further investigation. In particular, we analyzed data from a large, distributed system of darknet monitors. We characterized 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). We found that a small fraction of the total source IPs observed at a single darknet are responsible for the overwhelming majority of the packets and that 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. Therefore we created 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.

Aggregating to solve a local problem Blacklists have become popular among the operational community to filter or block the explosive growth of unwanted traffic on the Internet. Blacklists generated from firewall logs are used to block compromised hosts and blacklists generated from spamtraps are used to block spam. While these techniques have gained prominence, little is known about their effectiveness or potential drawbacks.

We performed a preliminary study [22] on the effectiveness of reputation-based blacklists–namely those that are used for spam detection. We examined the effectiveness, in terms of false positives and negatives, of four blacklists, namely NJABL, SORBS, SpamHaus and SpamCop and investigated into the sources of the reported inaccuracy. We found that the blacklists studied in our network exhibited a large false negative rate. NJABL had a false negative rate of 98%, SORBS had 65%, SpamCop had 35% and SpamHaus had roughly 36%. The false positive rate of all blacklists were low except that of SORBS, which had an overall false positive rate of 10%. The false positive of SORBS came mostly from blacklisting six Google mail servers that sent significant amount of ham to our network. However, since very little is known about the approaches taken by these services to generate their blacklists, and only the results of the generation are available (not the raw data), no one has explored in depth the reasons for these failures.

To solve this problem, we proposed [23] a new context-aware approach to blacklist generation. By making use of local usage and reachability information as well as the global information provided by blacklists, we showed that we can provide a significant improvement over existing approaches. In particular, this context aware paradigm enabled two specific techniques: ratio-based blacklisting and speculative aggregation. In the ratio-based blacklisting approach, the traffic on the live network is compared to the traffic on the spamtraps to determine if it is safe to blacklist an IP address. We called this approach the ratio-based approach as the ratio of email messages on the live network to the email messages on the spamtrap is used as a measure to blacklist an IP address. In the second approach, speculative aggregation, we used local reachability information as well as application history to predict where new spam messages will come while limiting the chance that these predicted hosts or networks are of use to the local network. A deployment of context aware blacklists for over a month in a large academic network demonstrated significant improvement in blacklist accuracy.

4 Discussion

In this paper, we have taken a retrospective look at some of our research group's output and characterized this work in terms of *context-aware security*. We defined context to be the unique attack surface, vulnerability profile, and usage models that underly the unique risk tradeoffs embodied by each organization. Context-aware security, then, is the application of this context to improve the accuracy, performance (in time or space), or reliability of security software devices, algorithms, or techniques. In its most basic form, our thesis has been that ”one size does not fit all”. Security is not something that can be applied uniformly as if encryption, intrusion detection, honeypots, etc. are security in and of themselves. They must be applied differently in each unique context if they are to be effective.

While the notion of context-aware security strikes at the core of nearly all modern security problems, we do believe that several areas of context-aware security provide particularly interesting and relevant areas for future research. First, while the previous research examples included both diverse sources of contextual information and the notion of aggregation, they stopped short of a *unified framework for local context*. By explicitly representing the various aspects of context (e.g., attack surface) and the multiple representations of that data (e.g., snort, honeypots, av logs, etc.) we hope to automate the configuration, placement, and operation of various security devices in the network. Second, we believe that *context-aware risk assessment* offers perhaps the epitome of context aware application domains. While previous theoretic work on assessing the security of networks has focused on vulnerability profiles (e.g., attack graphs), we believe that a unifying context framework can help develop concrete metrics for assessing (and mitigating) risk within one's network. Finally, we believe *context-aware security interfaces* are necessary, not only to manage the increasingly ubiquitous and mobile computing environments of today, but to manage the complexity inherent in managing one's context as they move through numerous such environments.

5 Acknowledgements

We would like to thank all of the students in our research group and especially Evan Cooke for their efforts in the original research that was highlighted here. This work was supported in part by the U.S. Department of Homeland Security Science & Technology Directorate under Contracts No. NBCHC060090, NBCHC080037, and AFRL cooperative agreement FA8750-08-2-0147.

References

[1] K. G. Anagnostakis, S. Sidiroglou, P. Akritidis, K. Xinidis, E. Markatos, and A. D. Keromytis. Detecting targeted attacks using shadow honeypots. In *Proceedings of the 14th USENIX Security Symposium*, Baltimore, MD, August 2005.

[2] Arbor Networks. Worldwide infrastructure security report, Sept. 2006.

[3] M. Bailey, E. Cooke, F. Jahanian, J. Nazario, and D. 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.

[4] M. Bailey, E. Cooke, F. Jahanian, N. Provos, K. Rosaen, and D. Watson. Data Reduction for the Scalable Automated Analysis of Distributed Darknet Traffic. *Proceedings of the USENIX/ACM Internet Measurement Conference*, Oct. 2005.

[5] J. Canavan. The evolution of malicious irc bots. In *Virus Bulletin Conference*, pages 104–114, 2005.

[6] E. Cooke, M. Bailey, F. Jahanian, and R. Mortier. The dark oracle: Perspective-aware unused and unreachable address discovery. In *Proceedings of the 3rd USENIX Symposium on Networked Systems Design and Implementation (NSDI '06)*, May 2006.

[7] E. Cooke, M. Bailey, Z. M. Mao, D. Watson, and F. Jahanian. Toward understanding distributed blackhole placement. In *Proceedings of the 2004 ACM Workshop on Rapid Malcode (WORM-04)*, New York, Oct 2004. ACM Press.

[8] E. Cooke, M. Bailey, D. Watson, F. Jahanian D. McPherson, and Z. M. Mao. The Internet Motion Sensor Project. `http://ims.eecs.umich.edu`, June 2004.

[9] T. Cymru. The darknet project. `http://www.cymru.com/Darknet/index.html`, June 2004.

[10] X. Jiang and D. 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.

[11] T. Joachims. Optimizing search engines using clickthrough data. In *KDD '02: Proceedings of the eighth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 133–142, New York, NY, USA, 2002. ACM.

[12] J. Mason. Filtering Spam with SpamAssassin. SAGE-IE meeting presentation, 2002.

[13] Microsoft. Microsoft security intelligence report: January-june 2006. `http://www.microsoft.com/technet/security/default.mspx`, October 2006.

[14] D. Moore. Network telescopes: Observing small or distant security events. In *11th USENIX Security Symposium, Invited talk*, San Francisco, CA, Aug. 5–9 2002. Unpublished.

[15] D. Moore, V. Paxson, S. Savage, C. Shannon, S. Staniford, and N. Weaver. Inside the Slammer worm. *IEEE Security & Privacy*, 1(4):33–39, 2003.

[16] G. Mostefaoui, J. Pasquier-Rocha, and P. Brezillon. Context-aware computing: a guide for the pervasive computing community. *Pervasive Services, 2004. ICPS 2004. Proceedings. The IEEE/ACS International Conference on*, pages 39–48, 19-23 July 2004.

[17] R. Pang, V. Yegneswaran, P. Barford, V. Paxson, and L. Peterson. Characteristics of Internet background radiation. In *Proceedings of the 4th ACM SIGCOMM conference on Internet measurement*, pages 27–40. ACM Press, 2004.

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

[19] M. Roesch. Snort: Lightweight intrusion detection for networks. In *Proceedings of Usenix Lisa Conference*, November, 2001.

[20] C. Shannon, D. Moore, and J. Brown. Code-Red: a case study on the spread and victims of an Internet worm. In *Proceedings of the Internet Measurement Workshop (IMW)*, Dec. 2002.

[21] S. Sinha, M. Bailey, and F. Jahanian. Shedding light on the configuration of dark addresses. In *Proceedings of Network and Distributed System Security Symposium (NDSS '07)*, February 2007.

[22] S. Sinha, M. Bailey, and F. Jahanian. Shades Of Grey: On the effectiveness of reputation based blacklists. In *International Conference on Malicious and Unwanted Software (Malware 2008)*, October 2008.

[23] S. Sinha, M. Bailey, and F. Jahanian. On the generation of internet blacklists. In submission, 2009.

[24] S. Sinha, F. Jahanian, and J. M. Patel. Wind: Workload-aware intrusion detection. In *Symposium on Recent Advances in Intrusion Detection (RAID'06)*, Hamburg, Germany, September 2006.

[25] L. Spitzner. *Honeypots: Tracking Hackers*. Addison-Wesley, 2002.

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

[27] Symantec. Symantec Internet threat report: Trends for July '05 - December '05. `http://www.symantec.com/enterprise/threatreport/index.jsp`, March, 2006.

[28] M. Vrable, J. Ma, J. Chen, D. Moore, E. Vandekieft, A. C. Snoeren, G. M. Voelker, and S. Savage. Scalability, fidelity and containment in the Potemkin virtual honeyfarm. In *Proceedings of the 20th ACM Symposium on Operating System Principles (SOSP)*, Brighton, UK, Oct. 2005.

[29] V. Yegneswaran, P. Barford, and D. 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, Oct. 2004.