The Use of Honeynets to Increase Computer Network Security and User Awareness

Sven Krasser, Julian B. Grizzard, Henry L. Owen
Georgia Institute of Technology
School of Electrical and Computer Engineering
Atlanta, GA 30332-0250
{sven, grizzard, owen}@ece.gatech.edu

John G. Levine
United States Military Academy
West Point, NY 10996
john.levine@usma.edu

Abstract

In this paper, we address how honeynets, networks of computers intended to be compromised, can be used to increase network security in a large organizational environment. We outline the current threats Internet security is facing at present and show how honeynets can be used to learn about those threats for the future. We investigate issues researchers have to take into account before deploying or while running a honeynet. Moreover, we describe how we tied honeynet research into computer security classes at Georgia Tech to successfully train students and spark interest in computer security.

1 Introduction

The overwhelming success and the rapid growth of the Internet has made networked computer systems a ubiquitous resource. Such computer systems are vital for most companies as production machines, where they may be used to process confidential and critical data for the company. Moreover, most private households own computers, which are connected to the Internet.

Over the course of the last several years, the functionality and usefulness of computer systems have increased vastly, which leads to an increasing complexity of these systems. This complexity leads to vulnerabilities introduced by flaws in the program code or by misconfiguration. An attacker can use such vulnerabilities to gain remote access, granting partial or full control over these computer systems. The attacker can initiate the attack from any arbitrarily system connected to the Internet that is already under the control of the attacker.

Computer worms are a recent phenomenon that also takes advantage of computer vulnerabilities. Worms infect computer systems and automatically search for and spread to other vulnerable systems. Worms frequently install backdoors on infected systems.

Such compromised computer systems pose a serious threat both to the Internet and to the user of a compromised system. As an example of the former, an attacker can use a vast amount of computer systems compromised by a worm to launch a distributed denial of service (DDoS) attack against a victim, e.g. a website. The victim is bombarded with a huge amount of data packets using up its network or computing resources—effectively taking down the machine. Such attacks can also be launched against critical Internet infrastructure, e.g. the domain name system (DNS) root servers. These servers have a crucial role for resolving human readable domain names to network addresses, which are interpretable by Internet routers. Moreover, it has been reported that spammers start to use compromised machines to send out their unwanted advertisement emails. By using compromised machines, they are able to both conceal their identity and evade attempts to blacklist known sources of spam email.

The user of a compromised system also faces various threats. An attacker can monitor keystrokes to learn passwords and credit card numbers, use a compromised system to traffic contraband such as pirated software or credit card numbers, distribute pornography, steal sensitive data, use private data like emails for social engineering, use the machine as a launch pad for further compromises so that the user of the compromised machine appears to be the attacker, or even use an attached microphone to eavesdrop on the user of the compromised machine.

These threats are compelling arguments to actively promote security on the Internet and among its users. This involves both securing systems on the Internet and educating

users about possible threats. Many home Internet users argue that they do not have sensitive information on their home computers and use this a vindication for their sloppily secured systems. Especially with respect to the computing and network resources they make available to attackers, this attitude can be compared to leaving a loaded gun on the doorstep and justifying it with the low value the gun has. Hence, a crucial part of securing the Internet lies in raising security awareness among its users.

Another important element of security is understanding the attackers. To learn more about their techniques, tactics, intentions, and motivations, researchers have deployed honeynets, which we will focus on in this paper. The basic idea is to give attackers vulnerable systems to attack. These systems are monitored closely, and the behavior of the attackers is studied.

2 Honeynets

2.1 Overview

A honeynet is a network of honeypots. A honeypot is an information system resource whose value lies in unauthorized or illicit use of that resource, i.e. a honeypot is a resource that is intended to be compromised. As we will see, a honeynet can provide the system administrator with intelligence about vulnerabilities and compromises within the network.

Any type of system can be placed within the honeynet. Standard production systems can be used on the honeynet, in order to give the hacker the look and feel of a real system. Moreover, virtual systems can be used to emulate or simulate a number of computer systems inside one physical system, e.g. utilizing software like VMware or honeyd. We will address this later on more thoroughly.

As previously noted, compromised systems pose a threat to the Internet. Since honeypots will be compromised, it is crucial to protect other systems from being attacked by them. Therefore, a honeynet is placed behind an entity called a honeywall. The honeywall separates the honeynet and the Internet such that all inbound and outbound data traffic has to flow through it. The honeywall limits the amount of malicious traffic that can leave the honeynet so that an attacker is kept from attacking other machines on the Internet using honeynet resources. This property of a honeynet setup is called *data*

control. Furthermore, the honeywall logs all traffic from and to the honeypots. This property is known as *data capture*.

2.2 Data Control

The principle of data control is concerned with protecting non-honeynet systems that an attacker might target from a compromised honeypot. For example, a honeywall can limit the number of outgoing connections allowed per hour. Another approach is to limit bandwidth usage to provide less network resources to the attacker that he or she can exploit. Furthermore, malicious data packets that specifically target vulnerabilities on other systems can be modified on-the-fly by the honeywall to make them benign.

2.3 Data Capture

Data capture relates to the actual goal of deploying a honeynet. The goal is to gather information on attackers and their tactics. The honeywall can monitor all data traffic entering, leaving, or inside the honeynet. This data can help to analyze the steps an attacker took to compromise a honeypot and how he or she uses the honeypot after it was compromised. However, once an attacker has control over a honeypot, he or she can install encryption software to encrypt all communications between the attacker's machine and the honeypot. The honeywall is not able to decrypt this data.

To address this problem, a logger known as Sebek has been developed [1]. Sebek is a software tool running on each honeypot. It runs as part of a machine's operating system and is able to intercept data after the attacker's encryption software decrypted it. This data is then sent to a Sebek server that collects all data for later analysis.

2.4 History of the Georgia Tech Honeynet

The Georgia Tech honeynet was established during the summer of 2002. We decided to start small, using only one honeypot. We very closely monitored the honeypot, even unplugging its network connection at night and on the weekends. The honeynet was set up using open source software and surplus equipment that was far from state of the art. Current state of the art equipment is not necessary since the machines running on the honeynet, the honeypots, have no production value. The amount of traffic going to and from the honeynet should be minimum since these systems are not running any

production software. Although there are commercial versions of software and products available to establish a honeynet, we chose to establish the Georgia Tech honeynet using open source software. We felt that open source software provided us with the greatest flexibility while keeping the costs low.

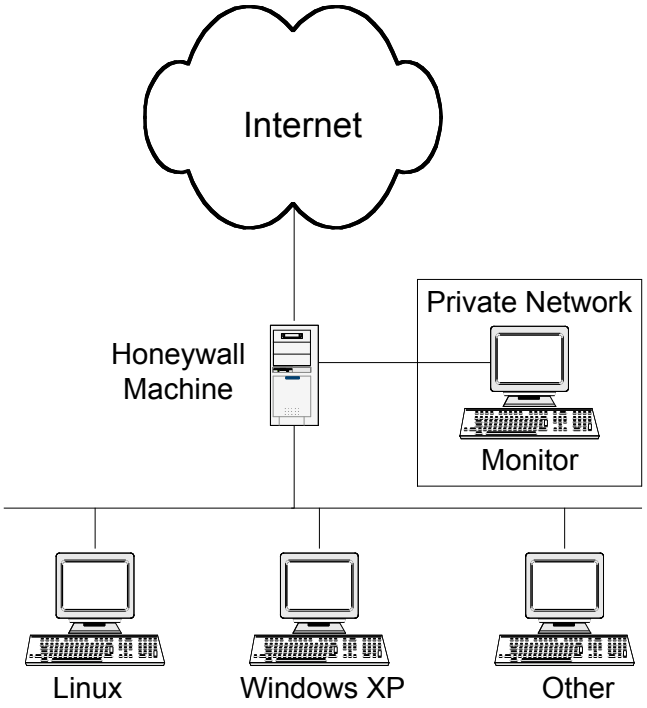

Figure 1: Typical Honeynet Setup

There are many ways to set up a honeynet. A typical configuration is shown in Figure 1. As the most important part, a computer, known as the honeywall, is placed in front of the vulnerable honeypots and is used to limit outgoing attack traffic from the honeypots. The honeywall acts as a gateway to the Internet for the honeypots and has the ability to limit malicious traffic. Our initial setup was similar to the architecture shown in Figure 1 and can be considered a Generation I honeynet. We installed a Redhat Linux 7.3 system and used the rc.firewall script developed by the Honeynet Alliance to set up the firewall and establish data control for our honeynet. This script is available at [2]. The purpose of this script is to provide data control. For example, the script can limit the number of outgoing connections to prevent a denial of service (DoS) attack from the honeypots.

In the summer of 2003, we transitioned to a Generation II honeynet. A Generation II honeynet differs from a Generation I honeynet in that the honeywall acts as a bridge instead of a router [3]. The main difference between a router and a bridge is the layer on which each of them is processing data in the network architecture. A router is an Internet device that is also visible to Internet users. It has an Internet address and if it receives data packets, it determines based on the destination Internet address in the packet header where to forward the packet. Moreover, routers process and change information in the packet header. A bridge works at a lower level inside a local area network like a honeynet. It does not have an Internet address and is unaware of Internet packet headers. It simply relays data frames between its network interfaces.

The honeywall of a Generation II honeynet uses bridging technology to hide away the data control facilities. It differs from a normal bridge in that it inspects data frames it relays and is aware of the Internet data contained in those frames. In contrast to a normal bridge, a Generation II honeywall may change fields in an Internet packet to render it benign in case it has been recognized as a malicious packet crafted by an attacker to compromise a non-honeypot system. It may also drop packets to limit the amount of network resources a honeypot consumes for outgoing traffic to inhibit denial of service attacks using honeypots.

In the summer of 2004, we transitioned to using the honeywall CD developed by the Honeynet Project available at [4]. The honeywall CD allows for a quick setup of a Generation II honeynet. With the honeywall CD, the operating system that runs the honeywall is booted off the CD. Only configuration files and log files are stored on the hard drive.

In addition to the ease of installation and configuration, there are three key advantages to using the honeywall CD as opposed to building a custom honeywall. First, any changes to the architecture can be easily upgraded simply by downloading the latest CD. Second, there is added security to using a CD as opposed to custom hard drive installations. In addition to the care taken in securing the honeywall CD operating system, the CD is read-only, meaning system binaries cannot be compromised. Finally, the Honeynet Project's current direction is to begin to analyze data from honeynets distributed across the world. The honeywall CD makes it easier to maintain the same conventions, such as file formats

and locations, as other organizations that are running honeynets. This makes it easier to share the data gathered with other researchers.

2.5 Overview of Honeynet Traffic

Since honeypots do not offer any useful services to Internet users and the Internet addresses of the honeypots are not publicly known, most traffic on the honeynet is suspicious. However, not all traffic is malicious. The traffic we see on the Georgia Tech honeynet falls into four categories:

- Benign network scans by the Georgia Tech Office of Information Technology,
- Traffic generated by honeypots due to normal network operations,
- Worms scanning for or infecting vulnerable machines on the honeynet, and
- Human attackers trying to gain access to or using honeypots.

The first category is benign scanning traffic. The Office of Information Technology routinely scans the Georgia Tech address range for vulnerable systems. Their scans occasionally also sweep the honeynet address range, which is part of the Georgia Tech address range. These scans originate from dedicated scanner systems inside the Office of Information Technology and can hence be easily distinguished from other traffic.

Honeypots also generate certain normal traffic. For example, the file sharing service of systems running Microsoft Windows sends out data traffic to find peer systems on the local network, which is within the honeynet in our case. Other configured services on the honeynet may generate similar traffic as part of their normal operations. Moreover, when honeypots try to resolve machine names like www.example.com, they have to communicate with a domain name system (DNS) server. Before honeypots communicate to other honeypots based on their Internet address, they have to resolve the Internet address to a local area network address using the address resolution protocol (ARP). Before honeypots communicate with the Internet, they also have to use ARP to learn the local area network address of the router connecting the honeynet to the Internet.

Another category of traffic is worm traffic. Worm traffic is hostile traffic. This sort of traffic includes worms scanning machines on the honeynet, worms infecting machines on the honeynet, infected honeypots scanning other machines for vulnerabilities, and infected honeypots attempting to infect other machines. The purpose of the honeywall is

to either block or mitigate the latter kind of traffic so that no machines on the Internet are compromised by honeypots.

The last category is traffic generated by human attackers. This includes random scans across the entire Internet address range of which our honeynet is a part. Similar to worms, this kind of traffic can include attempts to gain access to a honeypot or outbound traffic after the attacker has gained access. The outbound traffic has to be strictly policed so that an attacker cannot use the honeypot as a launch pad to compromise other machines on the Internet.

2.6 Issues to Consider when Deploying a Honeynet

2.6.1 Legality

One concern is that a honeynet may be considered a form of wiretapping of privileged communications. Richard Salgado investigates such legal issues in [5] (focusing on U.S. law). This section is a brief overview of some of his findings. However, readers who consider deploying a honeynet should consider talking to the legal department in their own organization before engaging the setup of a honeynet.

The Wiretap Act prohibits eavesdropping on electronic communication. However, honeynets might qualify under the Provider Protection Exception, which grants service providers to monitor their networks to secure them. Since a honeynet is deployed to be compromised, there is still some uncertainty in how far the Provider Protection Exception can be applied. Furthermore, the Computer Trespasser Exception, enacted as part of the PATRIOT Act, may be applied especially with respect to honeynets in facilities owned by the government such as public universities. This exception allows the government to monitor the communications of attackers.

If attackers compromise honeypots, they may try to use those systems to traffic contraband or other illegal material. Moreover, an attacker may attempt to use the honeynet in other sorts of crimes. The honeynet has to be closely monitored to be aware of such activity. Certain types of crime are required to be reported once they are detected, and it is a felony not to do so.

An attacker may try to compromise other systems on the Internet and use a honeypot as a launch pad for an attack. If those other systems are harmed, there may be liabilities

involved. Thus, it is crucial to closely watch what is happening on honeypots—to be able to pull the plug in case others could be harmed.

2.6.2 Circumventing Data Control or Data Capture

For the reasons stated above, data control and data capture are important cornerstones for a honeynet setup. Therefore, the honeynet should be built in a way that these two functionalities cannot be circumvented.

One common way to achieve data control is to limit the number of outgoing connections or the bandwidth available to a honeypot. However, an attacker could still use those limited resources to compromise another system on the Internet. A way to mitigate this risk is to let the honeywall modify known malicious outgoing traffic. The drawback is that this approach only tackles malicious traffic that is known to be malicious *a priori*.

There are multiple threats to data capturing. First, encryption can be used to hide the content of traffic to the capturing facilities on the honeywall. Moreover, capturing tools running on honeypots can be disabled by an attacker, and known limitations of these tools can be exploited [6].

2.6.3 Detectability

To be able to monitor skilled attackers in their attempt to break into systems, it should be hard for those attackers to be able to detect that they are compromising a honeypot rather than an actual production or home machine. This can be hard to achieve due to the data control and data capture characteristics.

An attacker may be able to detect the presence of data control facilities on a honeynet. For example, the attacker can try to initiate a number of connections beyond a connection limit by the honeynet. If this fails, this is an indication that something is odd about the machine the attacker compromised. Moreover, if the honeywall alters certain malicious data packets to prevent other machines on the Internet from being compromised, the attacker can deduce from failed attacks that a honeywall is present in the network. If the attacker sends those malicious packets back to a machine under the attacker's control, he or she can actually see whether the packet has been altered.

Data capture facilities may also be detectable by an attacker. For example, the Sebek software described above, which is used to capture data from honeypots, can be detected, or certain indications can suggest that the software runs on a honeypot [6].

2.7 Other Approaches

In this article, we focus on honeynets that use actual physical computer systems with standard operating systems as honeypots. A different idea is to use a single computer and emulate a complete honeynet with multiple virtual honeypots on it. This can be done by software like VMWare. VMWare can emulate a number of virtual computers on a single machine. The main advantage is that such a virtual honeynet is very inexpensive to set up. The downside is that it is not always hard for attackers to determine that they are in a virtual environment, which makes them more suspicious. This again is a detectability problem. Additionally, not all types of computing equipment can be set up in a virtual environment or may operate in exactly the same way as in a physical setup.

While the approaches described so far are highly interactive with the attacker—meaning that the attacker interacts with an actual computer system—there is another approach called low-interaction honeypots. Such honeypots can be implemented with software like Honeyd [7] (freely available at [8]), Specter [9], or BOF [10]. Low-interaction honeypots emulate services rather than complete systems. Hence, attackers cannot interact with actual operating systems. They interact with emulated counterfeit services like a fake web or mail server. However, the data gathered by low-interaction honeypots can give valuable information about which machines try to connect to the honeypot (and hence might be compromised machines used by attackers or infected by worms) or whether an attacker attempts to exploit known vulnerabilities in certain services. Additionally, low-interaction honeypots can serve as a decoy to attract spammers on the lookout for mail servers to relay their spam.

3 The Use of Honeynets

3.1 Detecting Compromised Systems

The traditional way of detecting compromised systems is the use of an intrusion detection system (IDS). IDSs use different techniques to detect intrusions. Anomaly-

based IDSs attempt to detect intruders by observing unusual use of system resources or network traffic. Since "unusual use" is a rather fuzzy definition, anomaly-based IDSs tend to produce a number of false-positive (legitimate use is incorrectly classified as an intrusion) and false-negative (illegitimate use is incorrectly classified as normal use) results. Signature-based IDSs look out for known kinds of attacks. Inherent to this kind of IDS is the disadvantage that it can only detect intrusions based on the signatures it has stored.

Honeynets are a new twist in intrusion detection. Since both attackers and worms use systems they compromise as a starting point to hijack other systems, their scanning for vulnerable systems in the vicinity will sooner or later swipe the honeynet. For this reason it is important to integrate the honeynet addresses well into the organization's address space. As we have outlined before, most traffic on a honeynet is suspicious. Therefore, this scanning traffic can be easily recognized in the honeynet logs and analyzed. In contrast to this, the vast amount of traffic on a production network IDSs have to sift through renders it complicated to separate legitimate from malicious traffic in a swift yet accurate fashion.

The Georgia Tech honeynet has been proven to be a valuable asset to our network in that sense. We were able to discover multiple compromised systems that slipped through IDS detection. The honeynet cannot substitute traditional IDS approaches. It has a limited visibility of the overall network and relies on attackers to scan it. However, the honeynet can serve as an important supplement to intrusion detection.

3.2 Learning about the Attackers

Honeynets allow researches to turn the table on attackers. Not only can researchers learn about the way attackers compromise and harness the resources of systems, but also attackers now get involved in detecting clandestine software running in the background. Attackers generally install backdoors onto systems they compromise so that they can regain access even if the security vulnerability they used to enter the system is later patched. Moreover, such software often also conceals the attacker's presence and activities on the system from the user. This kind of software (so-called *rootkits*) is typically hidden in parts of a system's operating system. Sebek is a similar kind of tool,

but it is used for a benevolent purpose. Attackers are now starting to explore new ways to detect this kind of software. Furthermore, as attackers become aware of the existence of honeynets, they might reconsider attacking a machine because they cannot determine from the outside whether the machine is actually worth compromising or is a honeypot machine set up to track attackers down.

Honeynets can also serve as a resource for compromised systems to do forensics on. The results of a detailed analysis can show what types of changes attackers make to a system and how they try to conceal their activities. Moreover, an attacker may target a previously unknown vulnerability to gain access to a honeypot, so that the research community can learn about it and provide fixes. Honeynets can also be used to collect rootkits that attackers install as backdoors [11], and analysis results can be used to improve rootkit detection tools.

3.3 Raising Security Awareness

Many people are not aware of the security risks their computer system faces. Further, they jeopardize their personal or company data. In fact, many people do not even notice that their system has been compromised. An attacker has an interest in concealing his or her activities to be able to keep access to a compromised system. Today's operating systems are insecure when they come freshly out-of-the-box and need to be patched. This is mainly due to the pace that security vulnerabilities are discovered. If an unprotected system is connected to the Internet simply to download the needed security fixes, it might get comprised in that short period of time—possibly unnoticed by the user of the system.

Honeynets can serve to make such threats visible. By its nature, a honeynet is closely monitored so that researchers can see what is going on under the hood. It can make people aware that a system running a standard out-of-the-box operating system and just connected to the Internet with an Internet address advertised nowhere will get scanned and eventually compromised after a short period of time [12]. People get tricked by the assumption that just because a system is not known to anybody else on the Internet it will not be found soon.

3.4 Education

We have incorporated the university honeynet into one of our undergraduate network security classes. We use the honeynet to provide realistic compromises for the students to work with. The students analyze data from a honeynet and perform forensics on the data to determine what happened. In addition to using the honeynet data directly in the classroom, we have used the honeynet capabilities for training students for participation in security exercises. In the winter of 2003, we participated in the Capture the Flag exercise hosted by the University of California at Santa Barbara [13]. The event was a multi-university event with teams across the country competing. The goal of the event was to capture the opposing teams' "flags" by gaining access to their exercise computer systems. Our students applied their honeynet forensics skills to analyze the traffic and determine how to capture the opposing teams' flags. This approach was highly successful.

The honeynet has sparked an interest beyond the classroom. A number of undergraduate students have gotten more involved in our research aspects of the honeynet. These students have gotten hands on experience with analyzing honeynet logs, learned forensics analysis of compromised honeypots, and gained insight into how research is conducted. These students have participated in being responsible for the daily analysis and reports we generate from the honeynet data. Additionally, the students have written tools, which automate many of the tasks and result in more efficient analysis of the honeynet data. One student has written a graphical analysis tool that allows graphical visualization of the status of the honeynet. Other tools are presently under development by students.

We have also exposed first year and second year students to the honeynet through the Intel Mentor program. With the Intel Mentor program, freshmen and sophomores are matched with a graduate student to work on a research project for a semester. At the end of the semester, the students make a poster and share their experiences with the rest of the students in the program. We were able to increase security awareness of the Intel scholars by involving them in our honeynet projects. We believe that this increased awareness of security through the honeynet activities will motivate some of these students to specialize in network security.

We have found the educational impact of the honeynet to be very high and well worth the investment in time required to implement a honeynet. The motivational impact of seeing your own computer probed, attacked, and compromised is unmatched by any other educational approach we have taken. Seeing a new attack, not knowing what it is, and then either analyzing it ourselves or later seeing community evaluation and analysis of that new attack is highly exciting and motivational to many potential network security students.

4 Conclusions

We have described how honeynets can be used to increase security awareness and network security in a University environment. We discussed the technologies and concepts upon which they are based, what issues are involved in deploying honeynet, and some of our experiences using a honeynet.

During the past two years, we successfully ran a honeynet at the Georgia Institute of Technology. This honeynet has proven to be of great value to network administrators, researchers, students, and researchers. We detected over a thousand allegedly compromised machines on campus, helping administrators to keep the network secure and users aware of their compromised systems.

The honeynet helped to introduce students rapidly to security research as a motivating training environment. It showed students the dangers and threats standard computers system—like their home computers—are facing when connecting to the Internet. Additionally, the honeynet aided both student and researchers at Georgia Tech in studying computer forensics. We found it to be highly motivating for students to not only learn how systems are compromised and how to protect systems but to see actual break-ins in the wild and convey hands-on experience in our labs. As a result, we believe we are helping to adequately prepare our students to protect our society, which more and more depends on networked systems.

References

[1] The Honeynet Project, "Know Your Enemy: Sebek," available online: http://honeynet.org/papers/sebek.pdf.

[2] The Honeynet Project, "Tools for Honeynets," available online: http://honeynet.org/tools/.

[3] The Honeynet Project, "Know Your Enemy: GenII Honeynets," available online: http://honeynet.org/papers/gen2/.

[4] The Honeynet Project, "Honeywall CDROM," available online: http://honeynet.org/tools/cdrom/.

[5] Richard Saldago, "Legal Issues," *Know Your Enemy*, The Honeynet Project, Addison-Wesley, 2004, pp. 225-252.

[6] Maximillian Dornseif, Thorsten Holz, and Christian N. Klein, "NoSEBrEaK—Attacking Honetnets," in *Proc. of the IEEE Workshop on Information Assurance and Security*, pp. 123-129, West Point, New York, 2004.

[7] Lance Spitzner, "Honeyd," *Honeypots—Tracking Hackers*, Addison Wesley, 2002, pp. 141-166.

[8] "Honeyd—Network Rhapsody for You," available online: http://www.citi.umich.edu/u/provos/honeyd/.

[9] Lance Spitzner, "Specter," *Honeypots—Tracking Hackers*, Addison Wesley, 2002, pp. 109-139.

[10] Lance Spitzner, "BackOfficer Friendly," *Honeypots—Tracking Hackers*, Addison Wesley, 2002, pp. 87-108.

[11] John G. Levine, Julian B. Grizzard, Henry L. Owen, "Application of a Methodology to Characterize Rootkits Retrieved from Honeynet," in *Proc. of the IEEE Workshop on Information Assurance and Security*, pp. 15-21, West Point, New York, 2004.

[12] The Honeynet Project, "Know Your Enemy: Statistics," available online: http://honeynet.org/papers/stats/.

[13] "Capture the Flag," available online: http://www.cs.ucsb.edu/~vigna/CTF/.