CHAPTER EIGHTEEN

Security Administration Tools and Practices

ESER KANDOGAN AND EBEN M. HABER

TODAY, HUNDREDS OF MILLIONS OF USERS DEPEND ON RELIABLE ACCESS TO COMPUTING AND INFORMATION services for business, educational, and personal activities. The growth of the Internet puts a world of information and services at our fingertips, yet also opens computers to attack from anywhere around the globe. The same networks that permit a tourist to read email from an airport in Singapore also permit a student in Romania to release a computer virus that disables computers and the businesses that depend on them. In addition, as the complexity of computer systems increases, new vulnerabilities are discovered each day. There is a worldwide community of people, usually referred to as hackers or crackers, who work to discover and exploit such vulnerabilities to attack and gain control of systems, sharing their techniques through various underground channels. Computers across the Internet have been subject to worms, denial-of-service attacks, password-sniffing, and other malicious activity, leading to significant inconvenience and loss of productivity for legitimate users. On the other side, vendors and computer system administrators race to discover vulnerabilities and to create, release, and apply patches before those vulnerabilities are exploited. On the front lines of this battle are *security administrators*, the people responsible for continually monitoring both their own systems and the ever-evolving security landscape in order to detect new attacks and prevent known attacks. Their work is crucial because, simply put, we cannot afford critical data and systems to get into the hands of hackers.

In this chapter, we provide an overview of the tools and work practices of security administration based on our ethnographic field studies at various computing centers. We profile two representative security administrators and detail five case studies to illustrate security work and the challenges faced by security administrators. Based on these findings, we outline some of the opportunities that lie ahead to improve security administration tools.

Introduction

Security administration takes place in an ever-changing landscape of new systems, new vulnerabilities, and new tools. As threats to computer security evolve, so too do the practices and tools of security administration. On one level, computers are being used in larger numbers and broader applications, forcing security administrators to deal with increasingly large volumes of information and placing correspondingly more demands on their tools. Many existing tools place much of the cognitive load for analysis onto the user, an unacceptable situation given the trends of ever more computers and network traffic to monitor. Administrators would clearly benefit from advances in analytics, automation, and visualization tools. On another level, computer systems are increasingly connected, providing access for an ever-wider variety of client systems including laptops, cell phones, PDAs, etc. The diversity of computing devices complicates security management and planning significantly. Increasingly complex software architectures create more opportunities for vulnerabilities to arise. With more components integrated and interacting in various ways in these architectures, there is a growing potential for unanticipated vulnerabilities. Sometimes, security breaches take advantage of multiple vulnerabilities in systems, making patterns of attack hard to predict. As a result, security administrators need to know how various devices and systems work and interact to analyze developing situations. In short, all of these changes to the information technology landscape make the job of the security administrator increasingly difficult.

So, how do security administrators secure our computing systems, defend them against attacks, limit damage proactively, and recover from attacks rapidly? In this chapter, we describe results from our ongoing field studies of system administration at various computing centers across the United States. In these studies, we examined the work practices, tools, organization, and environments of security, database, web, storage, and operating system administrators. So far, we have conducted 10 such field studies, where we observed approximately 25 administrators over a total 40 days.[1,2] We collected about

1 R. Barrett, E. Kandogan, P. P. Maglio, E. Haber, L. Takayama, and M. Prabaker, "Field Studies of Computer System Administrators: Analysis of System Management Tools and Practices," *Proceedings of the ACM Conference on Computer Supported Cooperative Work (CSCW'04)* (Chicago, Nov. 6–10, 2004), ACM Press, 2004, 388–395.

2 P. P. Maglio, E. Kandogan, and E. Haber, "Distributed Cognition and Joint Activity in Collaborative Problem Solving," *Proceedings of the Twenty-fifth Annual Conference of the Cognitive Science Society (COGSCI'03)* (Boston, July 31–Aug. 2, 2003), 758–763.

250 hours of video, which we analyzed to varying degrees of detail. In these studies, our approach has been ethnography, which involved entering the system administrators' environments and observing their practices, tools, and interactions for extended periods of time. Our analysis is based on Grounded Theory,[3,4] in which we do not use ethnography to validate a previously formulated hypothesis, but instead draw all our conclusions from what we observed.

ETHNOGRAPHY

Ethnography, defined literally as writing about people, is a technique commonly used in anthropology where one immerses one's self in a culture for extended periods of time to better understand the people and their practices. Ethnography traditionally has been practiced for understanding foreign cultures, but more recently it has been applied and has been highly effective in understanding the work practices of computer users in context, informing the design of computer systems to better suit their needs.[1,2,3]

Ethnography is almost always used to generate qualitative rather than quantitative data. Ethnographic techniques include observation, often as a full participant in daily activities, direct interviews, and collection of artifacts. Observers take notes and frequently record events using cameras, video, and audiotape. An ethnographer typically engages with a small number of subjects to study their daily lives in depth to get an understanding of the particular circumstances that drive behavior as opposed to drawing statistically significant conclusions about the whole culture. Ethnographic accounts aim to provide a rich description of events with as much detail as possible, not only expressing what happened but also interpreting the meaning and significance of events.

1 E. Hutchins, *Cognition in the Wild* (Cambridge, MA: MIT Press, 1995).
2 P. Luff, J. Hindmarsh, and C. Heath, *Workplace Studies: Recovering Work Practice and Information System Design* (Cambridge, MA: Cambridge University Press, 1999).
3 J. E. Orr, *Talking About Machines: An Ethnography of a Modern Job* (Ithaca, NY: Cornell University Press, 1996).

In this chapter we focus only on our findings in the area of security administration. We start with an overview of the attacks that security administrators work to prevent, and the tools that they use toward this end. We then give an overview of the current practices of security administration by profiling two representative security administrators, and detail five case studies to illustrate security work and the challenges faced by security administrators. We also discuss how current tools support or fail security administrators'

3 Barney G. Glaser and Anselm L. Strauss, *The Discovery of Grounded Theory: Strategies for Qualitative Research* (Chicago: Aldine, 1967).

4 Barney G. Glaser, *Basics of Grounded Theory Analysis: Emergence vs. Forcing* (Mill Valley, CA: Sociology Press, 1992).

practices. Lastly, based on these findings, we outline some of the opportunities that lie ahead to improve security administration tools.

Attacks, Detection, and Prevention

The security administrators we observed face two broad categories of attack: autonomous software such as worms and viruses that spread from machine to machine, and human-directed intrusions in which an attacker tries to compromise machines manually or by using semi-automated tools. While most virus attacks appear to have the goal of disrupting computer operations, most human-directed attacks aim to gain control of machines. These privileges are then used to attack other machines, steal data or computational resources, and occasionally destroy data. Some attackers appear to do little damage, instead concentrating on obtaining access to as many systems as possible to gain recognition from their peers. When discovered and blocked by security administrators, however, attackers sometimes strike back and try to damage the administrators' data and machines.

We saw security administrators using a variety of tools for detecting and preventing attacks. These tools include:

Global intrusion detection tools
: These monitor network traffic to analyze and report suspicious patterns—for example, Bro.[5]

Scanning tools
: These probe machines remotely for known vulnerabilities in their installed software—for example, Nessus.[6]

File/host integrity tools
: These run locally to check for compromised states; such tools include:

- Virus detection and repair tools—for example, Symantec AntiVirus.[7]
- Change management tools that track and compare system configuration information, including file organization, and alerting administrators when unauthorized changes occur—for example, Tripwire.[8]
- Rootkit hunters (a *rootkit* is a prepackaged set of programs and/or files used to exploit a vulnerability and gain control of a machine), etc.

Communication tools
: These are used to coordinate work and share information between administrators, such as email, phone, instant messaging, and chat rooms.

5 Bro Intrusion Detection System, *bro-ids.org*.

6 Nessus Open Source Vulnerability Scanner Project, *www.nessus.org*.

7 Symantec AntiVirus™, *www.symantec.com*.

8 Tripwire Change Auditing Solutions, *www.tripwire.com*.

Samples of code
: Such code exploits vulnerabilities and runs in a secure setting (e.g., VMWare) to better understand attacks.

Honeypots
: These are tools that emulate information system resources to attract attacks and capture attack data—for example, Sebek.[9]

Public information sources
: These contain data about vulnerabilities and attacks, including mailing lists and web sites such as FIRST (Forum of Incident Response and Security Teams[10]), bugtraq,[11] unisog,[12] CERT (Computer Emergency Readiness Team[13,14]), and SANS (SysAdmin, Audit, Network, Security[15]).

Administrators spend considerable time on prevention, researching new vulnerabilities and finding vulnerable machines. When faced with an automated attack, work becomes much more hectic as compromised machines are detected and isolated from the network to prevent the attack from spreading. Human-directed attacks are sometimes stopped in the same way, but occasionally security personnel will allow an attacker to continue in a controlled manner in order to trace the attack back to its source. In the next section, we describe security administrators and their work in more detail based on our studies.

Security Administrators

Over the last two years, we have conducted a number of ethnographic field studies looking at database management, web hosting, operating system administration, and security administration. We observed activities ranging from database backup and recovery to configuration of geographic load balancers, from patch management to computer security forensics. Most system administrator activities are either directly related to security or have security implications.

In this section, we profile two typical security administrators, Joe and Aaron. In the following section, we describe five security administration case studies from our observations. Finally, based upon our observations and interviews, we discuss the unique aspects of security administration work, and how well this work is supported by available tools.

9 Sebek Open Source Honeypot, *www.honeynet.org/tools/sebek*.

10 Forum of Incident Response and Security Teams (FIRST), *www.first.org*.

11 SecurityFocus.com, bugtraq, *www.securityfocus.com/archive/1*.

12 University Security Operations Group, unisog, *www.dshield.org/mailman/listinfo/unisog*.

13 Computer Emergency Readiness Team (CERT), *www.cert.org*.

14 J. H. Allen, *The CERT Guide to System and Network Security Practices* (Addison Wesley, 2001).

15 The SysAdmin, Audit, Network, Security (SANS) Institute, *www.sans.org*.

Profile of a Security Manager—Joe

Joe is a senior security engineer at a computing center in a large public university. He has been in this position for over four years. Previously, he worked as a system administrator at the same center and in various software development positions at different companies. The computing facilities at the center provide services for about 300 employees and external collaborators throughout the United States. Joe manages a team of three other security administrators. Together their responsibilities include setting up, configuring, and monitoring intrusion detection systems and responding to alerts reported. They work to proactively protect systems, detect attacks, and perform forensic analysis on compromised systems. As a manager, he is also responsible for defining policies and dealing with policy issues, and interacting with other concerned groups within the university as well as at other institutions. His motto is "Know Thy Network." Joe proudly claims that he learned most of the necessary skills primarily on the job. As shown in Figure 18-1, Joe's office has a bookcase full of various kinds of puzzles and games, and he considers this hobby as a nice metaphor for his day job.

FIGURE 18-1. *Puzzles in Joe's office*

Joe typically starts the day at 9:00 a.m. One of the first things he does is to start clients to read email and a MOO (An Object Oriented MUD—Multiple User Dialog). MOO is essentially a persistent messaging system composed of a set of virtual rooms representing dedicated spaces for admins to communicate on pre-agreed topics. Joe likes the MOO system, as it provides him a way to catch up on things quickly in the morning using its history function that shows past messages. Joe frequently checks both the MOO and email throughout the day. While the MOO allows him to feel the pulse of administration activity at the university, email provides updates from automated monitoring tools as well as security news from the outside world. Joe also participates in regular meetings and phone conferences with his counterparts at other organizations to share information

concerning recent and emerging threats. When he hears about new vulnerabilities through these channels, Joe often researches them further using a variety of web sites, some created by the security community, others by hackers.

Joe and others recently finished a five-month project to revamp the center's security policies. Because they work at a center within a larger university, they need to abide by the university's policies, yet they also need more restrictive policies in areas such as file sharing, wireless networks, connecting personal computers to the university network, external collaborator use, and so on. One of the biggest problems is to make the policies usable by making the policy document short enough so that all employees can read, remember, and follow it. Joe thinks a lot of education still needs to be done, as policy issues still do come up frequently. Users often read the policy document only once, when required to at the beginning of their employment, so continuing education and discussion are needed.

Joe typically leaves work around 5:00 p.m., but he quickly logs back in again from home so that he can be in touch with his colleagues. At 9:00 p.m., email reports from the center's change management and analysis tool are sent, and he usually likes to look at these before going to bed so that he can sleep with some peace of mind. Joe has been officially on call 24x7 for the last eight years, although he has received few off-hour calls in the last few years. He really enjoys the challenges of his work, but at times it can be a little too demanding on his personal life.

Profile of a Security Engineer—Aaron

Aaron is a security engineer who reports to Joe. He graduated with a master's degree in Computer Science last year, and has been working in this position ever since. He shares an office with Tom, another security administrator, and their office is full of computers and displays, as shown in Figure 18-2.

Aaron's schedule is similar to Joe's, but offset an hour or two later. The security group members deliberately stagger their hours to ensure better coverage at both ends of the workday. In fact, one of their team members works remotely from several time zones away, further increasing coverage. The rhythm of Aaron's day is centered on email, which he checks frequently (5 to 10 times per hour), looking for email notifications of security alerts. Through experience, he knows that certain alerts may be ignored on-sight, and others cause him to perform an immediate investigation such as looking through monitoring logs, checking network ports, searching for vulnerability/exploit data on the Web, and consulting with his co-workers via the MOO and face-to-face. He also checks the MOO regularly, but less frequently than email; he uses the MOO primarily for getting advice from more experienced admins.

When Aaron is not investigating a particular alert, he has assigned projects, such as scanning all the machines on the network for a newly discovered vulnerability. Any free time is occupied reading security mailing lists, though Aaron receives more mail from these lists than he has time to read. When he hears about new vulnerabilities through

FIGURE 18-2. *Aaron's workspace*

alerts or mailing lists, he usually performs web searches to learn more, and he occasionally downloads sample code for the attack to better understand how it works and which machines are vulnerable. Aaron has fewer meetings than Joe, and during meetings he always continues to monitor the network and research new potential attacks from his laptop. At the end of the day, Aaron heads home and, like his manager, connects in again after dinner to continue his tasks through the evening.

Aaron is the most junior member of the security team, but his educational background is strong. He spends much of his time educating himself to improve his knowledge of security. He is very motivated and really enjoys his work, and he seems to like the role of being a "good guy" helping fend off the "bad guys."

Security Administration: Cases from the Field

In this section, we describe the specifics of a number of actual security cases encountered by the security administrators profiled in this chapter. These cases fall into two categories: security checkup and attack analysis.

Security Checkup

One of the primary responsibilities of security administrators is to check the health of the systems in their charge. System "checkup" activities are performed routinely to ensure that systems are secure with the latest fixes, that only permitted applications are running with the proper approved releases, and that the system is generally free of viruses and

worms. While intrusion detection systems are constantly on alert, notifying the security administrators of any emerging suspicious patterns, administrators also proactively scan their systems through various remote security tools to reduce the risk of an attack by eliminating any known vulnerabilities. This, however, is a very intensive task with high cognitive demands on the admin.

While scanning tools produce a comprehensive list of possible vulnerabilities, it is up to the administrator to examine this list, assess the risks involved, judge whether these risks are applicable in their situation, and take the appropriate actions. To illustrate the complexity of the "checkup" tasks we'll take a look, in the following subsections, at cases where we observed Aaron handling a virus incident, dealing with alerts sent by an intrusion detection system, and going through a report generated by one such security scanner tool.

Case 1: MyDoom

Having learned through various security news groups and web sites of the emergence of a new variant of the MyDoom virus, Aaron did a network scan of systems in his subnet. Much to his dismay, he discovered that one of the systems had suspicious and increasing network traffic activity involving port 1034 (used by MyDoom for network communications). Immediately acting upon these findings, he notified the system's owner of the situation and had the system taken off the network. Now, he wanted to examine the situation closely, assess the damage, verify whether this system had actually been involved in a virus attack, and get the system cleaned before bringing it back online.

First, he wanted to understand whether other systems were affected by this incident. To do that he needed to analyze logs from the network monitoring tools. Unfortunately, network monitoring generates several gigabytes of data per day, making it impossible for a human to read and understand the logs directly. Instead, Aaron created an ad hoc command-line analysis tool using a series of commands to get the list of all IP addresses associated with port 1034 activity, as shown here:

```
./bin/ra -xcz args.out -port 1034 | awk `{print $7}' - | awk -F.  `{print $1, $2, $3,
$5}' | sort -u
```

He then copied the list into a file called *mydoom.o* to process it further by associating each IP address with its hostname using the following commands:

```
for a in `cat mydoom.o'; do echo $a; host $a | awk `{print $5}' -; done
```

These results suggested that four other systems were potentially infected by the virus. So Aaron decided to collect more information on the MyDoom virus to understand how the virus works. A Google search on the subject quickly got him the information he needed and confirmed his suspicions that 1034 is the MyDoom port. Based on the information he found, he explained to us that this particular version of the virus was also causing problems on some search engines because it queried them for valid email addresses within a particular domain (and thus kept them extremely busy). Talking to his office

mate, he added that this made the virus very easy to spot, as it leaves a clear signature in the web logs, which could be scanned to search for matches to the URL templates used by the virus in querying search engines. He then told Joe about the new machines involved in the attack so that they too would be taken off the network.

Within hours, users began to note that their systems had been disconnected, and one of them sent him an email about the situation. He spoke to that user on the telephone and gave instructions on how to clean up the machine so that it could be put back on the network.

Case 2: intrusion alert—false alarm

Throughout the day, security administrators receive email alerts from intrusion detection systems. Typically, these alerts take the highest priority in their work schedule, because such alerts might indicate an ongoing attack. While detection is for the most part automated, classification of events as normal behavior or an attack requires human judgment. On normal days, most of the alerts are either false alarms or harmless, but it is up to the administrator to make the call. This requires an intimate knowledge of the environment—that is, system workload patterns, network packet traffic, application use, hardware and software architectures, and so on. This case study examines the use of human judgment by Aaron. During our observations, we observed Aaron stop a reporting task he was working on after receiving one such alert, shown here:

```
Following alerts in tcpred.  These seem to be coming from the known compromised
systems.  Please take time to investigate.
External IP (Once compromised IP's)
123.123.10.10 #once.compromised.host.edu
> Jul 27 15:10:14 123.123.10.10 0.1kb > 210.210.10.10/http 711kb 9,9m %12345
```

The alert specified that network traffic was detected involving the IP address of a once-compromised system. While the system had been repaired, extra attention is given to such systems in case the fix was insufficient and the machine is still compromised. Specifically, the HTTP log indicated that a file of 711KB in size was transferred from the formerly compromised machine to another internal machine. Noting that the alert was referring to the HTTP log with the ID 12345, Aaron did a search on the HTTP log for this particular ID using:

```
grep %12345 http.log
```

which returned:

```
109090.04476 %12345 start 123.213.10.10 > 210.210.10.10
%12345 GET download/perftool-tar.gz
```

The HTTP log provided more information for Aaron to investigate. Specifically, it revealed what file was downloaded (*perftool-tar.gz*). This gave him sufficient leads to pursue it further. He first found the hostname of the machine using the host command and used this information to query the owner of this particular machine (using the center's online administrative tools). He then did a Google search for this particular user and found a

number of documents related to parallel programming on his web page. At this point, he was reasonably sure this could be a legitimate file; he commented:

> "If this is a person doing parallel programming, it should be all right."

Just to make sure, he pointed his web browser to the web server on this particular machine to find out what it was being used for. Realizing that the machine was used as a web server for a research group doing performance analysis of parallel programming systems, he commented:

> "I think this guy basically wrote this thing. It should be legitimate."

Relieved, he further commented that this case was fairly straightforward because they knew what application was downloaded and who downloaded it. However, he added:

> "But then there are times when somebody would download a tar file, source code, or something. Then we need to see if it is an exploit or something."

Case 3: real-time network monitoring

Most sites perform continuous network monitoring to watch for traffic that could indicate an attack. This monitoring can be surprisingly thorough, as demonstrated in this case. While we observed a meeting discussing hacker tools, the security administrators started discussing a package called *ettercap*. Being unfamiliar with this tool, one of the observers began searching the Web for information about *ettercap* from his laptop over the wireless network. A few minutes later, Aaron informed us that Fred, a security administrator working remotely, had detected this traffic and asked about it on the MOO:

```
Fred: any idea who was looking for ettercap?  dhcp logs say <observer's machine name>
is a netbios name.  nothing in email logs (like pop from that IP address).
Fred: seemed more like research.
Fred: smtp port is open on that host, but it doesn't respond as smtp.  That could be a
hacker defender port.
Aaron: we were showing how <hacker> downloaded ettercap.  One of the visitors started
searching for it.
Fred: ah, ok.  thanks.
```

In the space of only a few minutes, the security administrator had detected web searches for the dangerous *ettercap* package, identified the name of the machine in question, checked the logs for other activity by that machine, and probed the ports on the machine. Fred could see that it was probably someone doing research, but checked the MOO to verify that it was legitimate.

Case 4: security scan

Security administrators routinely probe their systems using various remote security scanner tools to identify potential vulnerabilities. When the security admins discover potential problems, they work with other administrators such as network administrators and operation system administrators to patch systems, turn off unused services, etc., to eliminate these risks. Typically these scans produce reports that are quite detailed, giving

the administrators information on the potential vulnerability, an assessment of the risk factor, possible solutions, and further references to the issue for examination, as shown in Figure 18-3.

Vulnerability	ssh (22/tcp)	You are running a version of OpenSSH older than OpenSSH 3.2.1 A buffer overflow exists in the daemon if AFS is enabled on your system, or if the options KerberosTgtPassing or AFSTokenPassing are enabled. Even in this scenario, the vulnerability may be avoided by enabling UsePrivilegeSeparation. Versions prior to 2.9.9 are vulnerable to a remote root exploit. Versions prior to 3.2.1 are vulnerable to a local root exploit. Solution : Upgrade to the latest version of OpenSSH Risk factor : High CVE : CAN-2002-0575 BID : 4560 ID : 10954

FIGURE 18-3. *Output from a scan report*

During our observations, Aaron spent quite a bit of time going through one such report. Essentially what he did was to look at the list of potential vulnerabilities to understand the risks of each one and how they work, to test the exploits on safe, isolated systems, and, when a serious problem was found, to notify the responsible admins to patch the system in question.

One of the vulnerabilities reported had to do with NIPrint (a print service) that could allow an attacker to remotely overflow an internal buffer and thereby allow arbitrary code to execute. Unfortunately, at the time, this high-risk vulnerability had no solution, and the admin was referred to the vendor site for the latest information. Aaron tried the vendor web site to see if there were new fixes for this problem, but no new information was available. Then he did a Google search on "NIPrint exploits" and quickly found sites that not only had more information, but also provided source code for the exploit. Aaron examined the code in depth and decided that he should try it out. So, he copied the source code, compiled it, and ran it on the system in question. The exploit, however, returned an error code, indicating that print services were not available to the host. This led him to check to see if the port was open through a utility that determines what hosts are available on the network, what services (application name and version) these hosts are offering, what operating systems (and versions) they are running, and so on. Based on the information, he concluded that:

> "This is also a false positive. You see NIPrint has a vulnerability on Windows machines. From the utility I just ran, I see that the port is open but it is a BSD machine, which tells me that it is not vulnerable. Most probably that is why it has not been compromised so far."

A number of other vulnerabilities were suggested by the report as potential risks. Aaron went through most of them, collecting information through the references provided by the report, seeking further information on various search engines and security web sites, and occasionally testing the exploits himself. In a number of other cases, he found applications and services (such as FTP and web servers) running old versions with significant vulnerabilities, prohibited software (such as peer-to-peer file sharing services, game servers), and open vulnerable services (such as SMTP) left unused). He typically

collected all this information and reported it to the responsible admins, requesting that they fix them as soon as possible. At the end of the day, he told us that of the 15 machines he had scanned so far, only one machine was appropriately secure and had a clean report.

Attack Analysis

While attacks are not uncommon, the level of sophistication in these attacks varies widely. Most incidents are fairly easy to handle, as attackers leave some kind of footprints in the compromised systems that lead to their identification. When confronted, attackers typically stop their malicious activities. Only rarely are hackers so persistent in their attacks that they cause damage (financial or otherwise), and seldom do they manage to avoid identification for a long time. In such cases, security administrators typically contact the appropriate authorities to attempt prosecution of the perpetrators. In the following section, we look at one such case in detail.

Case 5: persistent hackers

During the four months before our visit, Joe had been defending against an ongoing attack on his university and several other universities. This incident was consuming close to 100% of his time and was not yet resolved at the time of our observation. According to Joe, these attackers (or perhaps even just a single person) had been persistently breaking into research centers and universities. Whenever an attack was discovered and the vulnerability patched, the attackers would find another way in. As a result, Joe had been in regular contact with 10 or so senior security admins at various institutions through phone meetings and email to share information and coordinate the response.

Attacks of this scale require not only good coordination among the involved parties, but also sophisticated information organization, processing, and interaction skills and tools. Joe was particularly skilled in using various information processing commands to analyze the voluminous log files containing hints of the attackers' activities. He was also following good information management strategies, as he and his colleagues organized information on exploits, incidents, people (including hackers), sites, etc., in various file directory structures. Naturally, coordinating information of this scale requires sophisticated information interaction tools, which was evident in Joe's environment, a virtual windowing system of nine or more virtual desktops spread across two large physical screens.

One of the invaluable pieces of information Joe and his colleagues collect in such incidents is attacker session logs. Sometimes, vulnerabilities in the attack tools permit security administrators to capture a detailed log of the attacker's activity. Examination of these sessions reveals a wealth of information about the tools and techniques attackers use, as well as information such as source hostnames, that could potentially lead to the identification of the hackers.

Joe and his colleagues had worked closely to focus on figuring out the access path of the compromised machines and the various techniques that hackers use in their attacks. Joe was particularly meticulous in these efforts, and kept a master file of attacker actions; there, he would copy interesting segments of the sessions, comment them with findings, and jot down references to other information. The closer the security administrators got to the originating machine, the closer they were to identifying the hackers and potentially prosecuting them. However, this case had been particularly challenging, as the attackers frequently changed their access path. In addition, not all of the compromised sites were helpful. There were various reasons for this lack of cooperation. This was the case particularly for large ISPs, which do not have the time to deal with such cases. Sometimes, sites would provide one of these session logs, which provide them new leads. Other times, sites would not deal with the attack beyond simply shutting down compromised machines. This, however, delayed tracking activities, as the admins would then need to wait for the attackers to come in through some other compromised machine.

In one session log that they had just received, Joe was able to find some of the toolsets that attackers were using. A directory listing in the log revealed the name of one of the tools (e.g., *abcd.tgz*), and Joe did a web search on the tool. In this case, he was not particularly lucky, however, as the only information he could find was another admin's report of a similar situation. The other admin did find that the tool was an exploit for a Domain Name Service (DNS) server vulnerability.

On occasion, Joe has been able to find the full source code of an exploit on underground attacker sites, including all the source files, README files, and Makefiles, as he did for one of the other exploits used in this ongoing incident. That exploit, he found out, allowed a user to stealthily access a root shell on the machine via HTTP requests. Thus, all attack traffic was on port 80, which made it difficult to spot because all web traffic uses the same port. This added further complication to an already complicated case involving multiple exploits. In this particular situation, three different vulnerabilities were exploited including, a web server backdoor, Secure Socket Layer (SSL) buffer overflow, and a DNS exploit.

At another point of the session log, Joe was able to find an unzipped listing of the tool files and, much to his surprise, a view of one of the source code files for the vulnerability. Apparently, during the session, the hacker, for some reason, opened an editor to view one of the exploit files. Joe explained later that this happens when hackers actually copy and paste exploit source code as opposed to downloading it from a web site to prevent leaving any traces in web server logs. In any case, this was a break, so Joe quickly copied the source code onto his machine. First he spent quite a bit of time to understand what the exploit was doing. Later, he compiled it and ran it on another system. Normally when security admins try out downloaded exploits and rootkits, they do so in a quarantined environment. In any case, this particular executable did not yield much new information, but Joe stated that a lot of their work is for educational purposes to understand what the attackers are trying to do.

At one point, one of Joe's colleagues, Tom, told him that in another part of the session, he saw a process listing, and a particular process caught his eye (`ssh abc@111.111.10.10`). This was a secure connection to a particular site, and if that site's administrators were cooperative, the logs could give Joe more information. Tom also saw a DNS session where the hacker changed his server from one site to another site. In the process of doing so, Tom was able to capture the username and password used by the attacker, probably for a compromised account at this particular site. This, they decided, was another lead, but they would need to think further about how they could best use it. They could potentially prepare traps for the hacker through mirroring of the IP packets, but they considered that this could be tricky, as the attacker might notice it. They also joked about changing his password.

This had been another long day. The security administrators had found some new leads, particularly information on new tools that the hackers added to their portfolio. Unfortunately, this was not good news: it simply meant that the attackers were getting better and better, and Joe and his colleagues just wanted to put a stop to their efforts. Joe's organization had not been the prime target of attacks for some time, but he felt obligated to work on the project. The various administrators were getting closer to where the hackers were, so Joe took it upon himself to track the attackers and help out those other sites that were being compromised.

The Need for Security Administration Tools

Security administrators have a variety of tools at their disposal. In the cases we examined, we saw the use of intrusion detection systems based on network monitoring, remote and local scanning tools, public information sources, data analysis tools, etc. While these semi-automated tools help the administrators significantly overall, it is clear that security as a whole would be very difficult to automate fully—much of the analysis requires intelligence and judgment to determine whether a certain system behavior is the result of legitimate activity. For example, in case 2, Aaron investigated the research activities of the machine owner in question to determine whether a particular file download was reasonable. In case 5, Joe needed to download and examine source code, read online reports of similar problems, and coordinate activities across multiple institutions. Simply put, there isn't much room for brute force automation. Yet obviously tools can help administrators do their jobs more effectively. Advanced analytical and visualization techniques could help admins manage large amounts of information. In our observations, we saw little or no use of data mining technologies to analyze patterns of activity. Automatic classification could help the administrator to focus on questionable activities rather than the obvious false alarms. We also saw little or no use of visualization techniques put into practical use. Real-time and post-incident visualization of activities could improve the ability of security administrators to develop situational awareness.[16,17]

16 C. Brodley, P. Chan, R. Lippman, and B. Yurcik, ACM Workshop on Visualization and Data Mining for Computer Security (Washington, DC, Oct. 29, 2004).

The case studies also show that an important aspect of security administration work is the integration of data from various parts of the system to construct and understand the real story. This work may involve relating data up and down and across various components in the system. However, this can be particularly challenging, for several reasons:

- There is simply too much to look at.
- In addition to the vast quantities of data in log files, there is no single standard data format describing the various events produced from all the monitoring and scanning tools.
- In distributed systems, out-of-sync system clocks make it difficult to correlate events with timestamps.

In summary, the various security tools are not well integrated. When one tool would produce a certain piece of information, we observed admins using manual tools to derive other information—for example, looking up machine names by network address and vice versa, or looking up machine owner names using online directories. With little integration, the security administrators typically take charge of integrating and correlating data using various ad hoc tools and commands to process, combine, and make sense of the data. When processing information, security admins frequently create scratch documents to hold data as a stage for further processing, as part of a report for a colleague, or for future reference. While the security administrators we observed were fairly proficient in the tools they used, many opportunities remain for further improvement, particularly in the area of workspace/activity management—specifically in activity reuse in information processing. In our observations, we clearly observed many patterns of activity where administrators examined logs to correlate events, yet each new incident required them to perform similar information processing repeatedly. Instead of skipping back and forth repeatedly between different tools and manually integrating information using command-line tools and temporary files, security admins could benefit from integration that automatically (or manually via a user-defined dataflow) processed the information, encapsulating activities and best practices in an executable form.

Time is a crucial factor for security administrators. Their work style is event driven: typically security admins stop their routine tasks whenever they receive an intrusion alert. Time is also an enemy, as security admins are very well aware of the fact that, given enough time, many systems are compromisable. Thus, security administrators need to be more current in their field than do other system operators. New vulnerabilities and attacks are discovered daily. At high-profile computing centers, attacks can come very quickly after vulnerabilities are discovered. Proactive work pays off later on. Security admins must constantly watch for new vulnerabilities and proactively scan their systems for possible exploits. Security administrators also need to understand how various

17 J. R. Goodall, W. G. Lutters, and A. Komlodi, "I Know My Network: Expertise in Intrusion Detection," *Proceedings of the ACM Conference on Computer-Supported Cooperative Work (CSCW'04)*, (Chicago, Nov. 6–10, 2004), (ACM Press, 2004), 342–345.

systems, architectures, and tools work, which components connect to each other and over which port, and which files are created and where, to distinguish legitimate activity from subtle attacks. Here again, tools are not very effective, particularly because each environment is different—thus, Joe's motto: "Know Thy Network!" Experience takes years to build and brute force tools, while useful, fall short of meeting administrators' needs.

Security administration requires collaboration between people at many levels. Co-workers frequently check with each other face-to-face or via chat rooms to verify the validity of activities. Across organizations, they share their experiences, problems, scripts, and tools, and they collaborate to track down the most serious attacks. The "we are all in this together" mentality is widespread and most appreciated. Better tools for facilitating this collaboration may also be needed. Currently such communication is done through phone, email, scratch notes containing contact information, etc. Clearly, security administrators are in search of better means to collaborate and are exploring interesting approaches such as the use of MOOs. MOOs are a particularly interesting approach, as messages could be produced and consumed by programs such as bots. While we have not yet seen much exploration of this possibility, there are interesting opportunities where human and software information processing agents can collaboratively and cooperatively manage security. Bots can form the first line of defense, handling the obvious cases and notifying security administrators in questionable and critical cases.

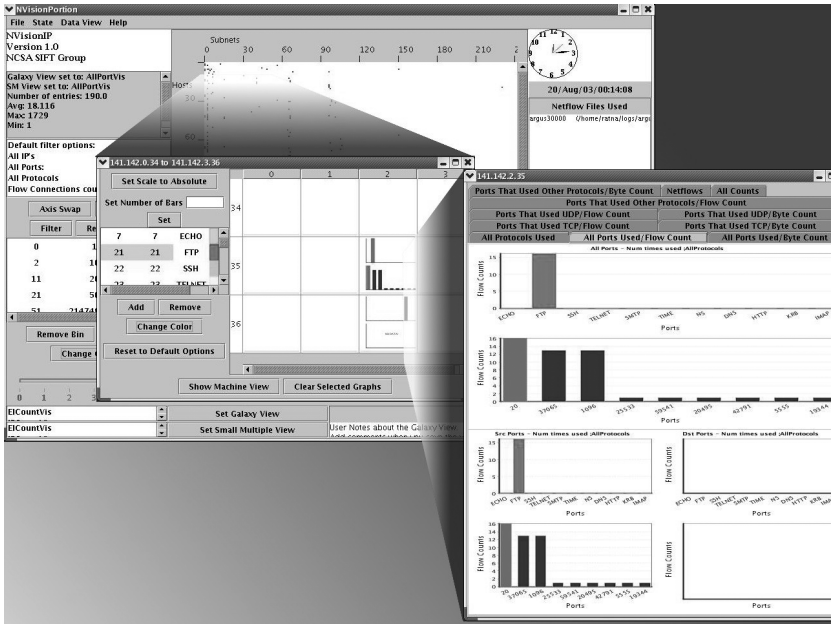

FIGURE 18-4. *NVisionIP interface (courtesy of NCSA)*

Security administration is distinct from other types of system administration, in several ways. First, while most system administration groups own some kind of computing

VISUALIZATION FOR SECURITY

Security work is likely to remain highly human intensive, yet the work is becoming increasingly challenging. High-volume, multidimensional, heterogeneous, and distributed data streams need to be analyzed both in real time and historically. In security practice, visualization tools are currently underutilized. Visualization for security is challenging, as current techniques try to match the needs of security administrators to gain situational awareness, correlate and classify security events, and improve their effectiveness by reducing noise in the data. Visualization coupled with data mining is likely to help security administrators make sense of network flow dynamics, vulnerabilities, intrusion detection alarms, virus propagation, logs, and attacks.

One approach is to provide multiple coordinated visualizations, as in the NVisionIP[1] interface shown in Figure 18-4, where network traffic is visualized at multiple levels from a single machine view to the overall network view, to improve the situational awareness of the security administrator.

[1] K. Lakkaraju, W. Yurcik, and A. J. Lee, "NVisionIP: Netflow Visualizations of System State for Security Situational Awareness," *Proceedings of the ACM Workshop on Visualization and Data Mining for Computer Security*, 2004.

resources (such as database administrators who own the database management software, the network team that owns the networking equipment, switches, firewall, etc.), the security administration team doesn't own the computing hardware or software for which they are responsible. This can lead to problems when other administrators don't share the same urgency in deploying fixes. Second, other administrators are mainly fighting against bugs and poorly working code; with security administrators, on the other hand, there are literally people out to get them. This problem is severe enough that one of the administrators we observed mentioned that he had asked to be left out of any public listing of security engineers, because attackers put extra effort into compromising machines and data belonging to security people.

As computer systems continue to grow in number and complexity, and as network traffic continues to increase, security work will only get harder unless better tools are developed. Based on our observations and discussion, we believe that the most important directions for tool development include:

- Standardization of event formats to permit easier integration between tools
- Tools to integrate and correlate events from multiple distributed systems, either automatically or manually via user-defined data flow
- Application of data mining and other analytics in activity classification, analysis, and noise reduction
- Automatic event stream processing
- Effective workspace, activity, and information management tools

- Improved collaboration and information-sharing tools
- Scalable, customizable, programmable visualizations

Conclusion

Security work seems very much like war, espionage, or intense gaming. The security admins know things the attackers don't, and vice versa, and each is trying to use his knowledge to the other's disadvantage, while keeping the knowledge secret. It seems very much like watching a game of cat-and-mouse as the security admins take advantage of vulnerabilities in the attackers' tools to observe their activity, deliberately allowing machines to remain compromised in order to trace the attack. Meanwhile, the attackers keep discovering new vulnerabilities to make further attacks.

Security administrators work on the front lines defending against people who are trying to compromise the computer systems that support much of our modern society. At times, security admins might appear paranoid, but there really *are* people out to get them. In this chapter, we introduced the work practices, tools, and needs of this important group as revealed through our field observations and interviews. Security work involves research into emerging threats, situational awareness of system status, integration and processing of data from multiple sources, and, most importantly, human judgment as to whether a particular pattern of activity is legitimate. As computer systems continue to increase in number and complexity, and as network traffic continues to increase, this work will only get harder unless better tools are developed.

Acknowledgments

We are grateful to our field study subjects, who remain nameless to preserve their anonymity. They cheerfully answered our questions and weren't bothered by our crowding their offices with extra people and video equipment, all while continuing to perform a crucial job.

About the Authors

Eser Kandogan is a research staff member at IBM Almaden Research Center. He holds a Ph.D. degree from the University of Maryland, College Park, where he studied computer science with a specialization in human-computer interaction. His current interests include human interaction with complex systems, policy-based system management, ethnographic studies of system administrators, information visualization, and end-user programming.

Eben Haber works on human-computer interaction at IBM Almaden Research Center. He holds a Ph.D. from the University of Wisconsin-Madison, where he worked on improving user interfaces for database systems. His interests include databases, user interfaces, and the visualization of structured information. He has worked in industry on data mining and

visualization, and user interface design, and is currently studying human interaction with complex systems in the USER group at IBM Almaden.