Research Directions for Network Intrusion Recovery

[Position Paper]*

Michael E. Locasto
Institute for Security Technology Studies
Dartmouth College

Matthew Burnside
Department of Computer Science
Columbia University

Darrell Bethea
Department of Computer Science
UNC Chapel Hill

ABSTRACT

One of the most significant unsolved problems for network managers and system administrators is how to repair a network infrastructure after discovering evidence of an extensive compromise. The technical issues are compounded by a breathtaking variety of human factors. We highlight lessons learned from three real, significant, and recent intrusion incidents. We do so as a way to expose the difficulties — technical, sociological, and psychological — inherent in intrusion recovery. Most network users take a "secure" network infrastructure for granted. Real events show that this level of faith is unwarranted, as is the belief that intrusions are or can be completely repaired, especially in the absence of research on network recovery mechanisms that explicitly take the needs of support staff into account.

"Damage control is much easier when the actual damage is known. If a system administrator doesn't have a log, he or she should reload his compromised system from the release tapes or CD-ROM." — Firewalls and Internet Security: Repelling the Wily Hacker [2].

General Terms

Security, Measurement

Keywords

intrusion recovery, forensics, auditing

1. INTRODUCTION

Although many people enjoy the benefits of access to information and communication through networked systems, most take the security and reliability of these infrastructures for granted. Users may incorrectly assume that IT staff can fully repair the damage or harm an attacker causes (think of copied intellectual property, spent computer cycles, damaged reputations). Even security researchers may summarily dismiss the task as a simple, if somewhat lengthy, system administration job, and thus unworthy of investigation. It is our opinion that the problem of coordinating the repair and restoration of network infrastructures is an unaddressed problem that embeds a number of unanswered research questions involving the intersection of human factors and technical challenges.

1.1 Dual Nature of the Problem

Compromises of networked systems are difficult to analyze and respond to for a number of reasons. Because of the diversity of the problem and the lack of research into methods that deal with both technical and human factors, network intrusion recovery is more of an art than a science. The state of the art often involves manually reinstalling machines from read-only media, as the traditional text on firewalls [2] reminds us in the quote above. Even when this process *is* automated, it still resets systems to some initial state, thus deleting valuable data that may not have been backed up, or information that would be of some use in a forensic investigation. At this point, we must resist the temptation to treat the problem as solved by turning to some technical solution (*e.g.,* automated network-based OS installations, "ghosting" software, or recent research on an automated process for working backward from the attack to undo the damage caused [5, 3]). As we show, both technical and human factors introduce obstacles that simply executing a software application cannot overcome.

Even with the assumption that we can reliably detect an intrusion, there are many technical issues related to repairing a wide variety of hosts, nodes, objects, and artifacts. First, even with deep auditing information, it can be difficult to describe the extent of an intrusion within the context of a single system. Second, determining the extent of the damage throughout the network requires replicating or extracting those conditions to widen the scope of the detection process. Once the process of detecting an attack and determining its scope have been accomplished,[1] then the process of recovery presents an often overwhelming series of choices and possibilities. Planning and implementing a recovery may involve choices amongst a variety of changes

*An expanded form of this work is currently under submission to RAID 2008.

Symposium on Usable Privacy and Security (SOUPS): Workshop on Usable IT Security Management 2008, July 23–25, 2008, Pittsburgh, PA USA

[1]As we can see from the incidents described later in the paper, this process is not strictly linear — and is further complicated by human factors. In this case, thinking of detection as "accomplished" rather than "ongoing" is misleading.

to systems, hardware, applications, and network topology. Individual systems require forensics and may need to be isolated, removed, updated, or reconfigured. Software applications may need to be reconfigured or have patches applied, which raises the twin issues of which applications to fix in what order and what patches to generate or obtain (and what order to apply them). The network topology may need to change: new routers, switches, or other equipment may need to be introduced or existing equipment reconfigured. Firewall rules may need to be introduced or modified. Existing IDS sensors may require retuning. During this entire process, the team must test and verify each step.

We see recovery as a complicated, fluid process. Response teams often labor under a compressed time frame to fix as large a part of the intrusion in as short a time as possible. The forensics team experiences pressure to finish quickly to reduce service downtime. The recovery team's training and skill level, along with the vagaries of interpersonal relationships, can constrain what types of actions are realistic. Promotion, demotion, hiring, or termination decisions can affect someone's willingness to engage in extensive recovery actions. Attacks rarely occur at a convenient time for the IT staff; if the incident occurs near social events or holidays, the time pressure can increase greatly.

Although some technical fixes may be "obvious", both internal (to the team) and external (*i.e.,* the team's customers and employers) vested interests in maintaining the network status quo can prevent the implementation of these fixes. The team must be familiar with the preferences, attitudes, and biases of the user or customer population in order to "sell" the repair to them. Finally, the reputations of the team, individuals, customers and users, and institution requires careful consideration.

This paper offers evidence that illustrates what might otherwise be an overlooked point by information security researchers: intrusion recovery is not a simple systems administration task. Intrusion recovery, while a large technical challenge, is further complicated by human–level issues. We believe the community should focus on creating mechanisms that deal with recovery as a system composed of both humans and computer systems.

1.2 Background and Related Work

Complete technical solutions to the problem of recovering from *realistic* intrusions in the research literature are sparse, although both classic [11, 10] and more recent [9] examples of post-mortem intrusion analysis do exist. Spafford's analysis [10] of the Morris Worm and Cheswick's annotated log of the Berferd case [2]) can be seen as catalysts for changing the way computer scientists and network researchers thought about trust and security on the fledgling Internet. The analysis of these incidents helped spur the adoption of stronger authentication mechanisms, the use of firewalls to implement host communication policies, and research on basic auditing tools and intrusion sensors. More recently, Singer's article [9] recounts how even a well-designed infrastructure managed by an experienced, professional network security team can be compromised. This latter analysis helps illustrate just how difficult and time-consuming it can be to completely remove an attacker from a system. In this case, the attacker repeatedly found new avenues into the infrastructure, just when the admins thought they had adjusted their security posture appropriately.

Defending a network involves assessing risk and allocating resources to match the perceived threats and costs [1]. Knowing that the network is at a high risk of a compromise, however, does not directly inform the procedures that should be in place for repair. Instead, it may inform strategies for reducing or managing risk, and little research exists on systems for managing the disaster workflow recovery once a network *is* compromised. Payne *et al.* [7] provide a good overview of the research in this area, including beliefs about uncertain events, decisions made under risk and uncertainty, and frameworks for decision behavior.

Finally, as we saw in our attempts to collect information for this case study, the human memory is is notoriously unreliable. The reliability of eyewitness [12] testimony has been extensively studied; Wells and Olson [12] point out that the only scientific body of literature on eyewitness reliability exists in the psychology space. In the computer security field, and in the context of rebuilding complex network infrastructures and carrying out a number of both repetitive and complex tasks over a long period of time, human memory is relied upon far too much. Our case study shows that it is possible for initial planning goals, suggestions, or objections to be misunderstood, warped, or forgotten — leaving potentially large gaps in the actual level of security achieved after repairs complete.

2. METHODOLOGY

In researching this paper, we interviewed all parties involved in recoveries from three recent attacks on a medium-scale network, including administrative staff and management. We performed an email archive search, and confirmed many of the details of the attack through analysis of disk images from a number of the compromised machines.

We emphasize that we do not aim to single out or lay blame with individuals. Each interview subject gave us permission to interview him or her and to report on the process. Our goal is to present the facts of the situation, disposition of the network, and decisions made by the staff in as clear a light as possible, in order to motivate research and development of tools that ease the burden on IT staff during the process of network intrusion recovery.

One of the most significant challenges when responding to an intrusion is performing forensic analysis to determine the exact impact of the attack. In the case of the compromises we discuss here, the nature of the attacks was such that no individual performed a single coherent analysis. Rather, the analysis was performed piecemeal by the various members of the IT staff. As we discuss later, this fractured view presents the IT staff with an immediate problem when attempting to form a coherent response, and it presents a challenge to us as researchers. In some cases, parties we interviewed reported radically different timelines and analysis, even though the interviews took place less than a month after the attacks of December 2007 and within the scope of the March 2008 attacks. Where possible, conflicting statements were reconciled through mechanical methods — email or file modification dates — but there always remains some ambiguity.

3. INTRUSION INCIDENTS

Accounts discussing the trapping and tracking of attackers in improvised honeypots form part of the classic network security literature [11, 2]. Just as these accounts relate the

first examples of a honeypot and computer forensics, the improvisation required in these early responses forecast exactly the plight of network administrators today: when faced with a real attacker, decisions must be made quickly and accurately, and the decisions may conflict with the desires of other stakeholders. At the times of these early incidents, almost no tools existed to help trace hacker activity. Tools were improvised from the ground up, and their descendants and offshoots have become part of a standard set of tools. Now, network intrusion recovery faces an even larger challenge: create a suite of tools that take into account not only the engineering challenges of repairing a network, but also the human issues surrounding this process.

The network on which we focus our attention consists of approximately 1000 Windows, Linux, and Solaris workstations, as well as a number of infrastructure servers providing DNS, DHCP, and HTTP, and several general purpose clusters of SSH servers. Approximately 150 of the workstations run Red Hat Enterprise Linux (RHEL) AS v4. These machines are periodically updated from two source machines using `rdist`. Windows machines authenticate users via Active Directory; the Linux and Solaris machines authenticate users through NIS.

Over the time period covered in this case study, the network was administered by an IT staff of from three to five people, with a single manager. There is a high turnover, and staff members come from widely disparate backgrounds; some are students with little to no experience, while some are highly knowledgeable and very experienced. The network is complex for its size and has a number of systems, including the accounting system, which remain unchanged from the late 1990s. New staff, even if highly experienced, often take months to gain a complete understanding of the intricacies of the network. Furthermore, the network supports a research environment, with a strong tradition of open access to all information. This tradition supplies a political force that has precluded the use of any form of firewall in front of the end-user systems or the servers.

3.1 March 2007 Attack

In March of 2007, an attacker attempted to use a kernel exploit to gain root privileges on several of the RHEL workstations. For each machine on which the attack failed at least once, the IT team were able to use system logs to determine the origin of the attacker and the compromised user accounts he was using to access the machine.

The failed attacks were not all the same, however; the attacker was revising his methods, and there was no way to determine if he had succeeded. The staff checked the logs of other susceptible machines (those harboring the same vulnerability, but showing no indication of failed attacks). While staff could uncover no indication that the attacker had connected to the machines, it is possible that he altered the logs after gaining root access.

3.2 March 2007 Response

It is possible that the attacker never succeeded. Regardless, the safest response in this situation, recognized by all members of the IT staff, would have been to reinstall all vulnerable machines with a patched version of the operating system. There were, however, external constraints that prevented this approach. The attacks occurred in the middle of the semester and involved many machines heavily used by classes. Thus, the staff needed to carry out a solution as quickly as possible to avoid disruption to the Department's academic mission.

Most of the systems were nearly identical, with the exception of the servers, including the `rdist` masters. Reinstalling the `rdist` masters from scratch would have been time consuming and error-prone, as the `rdist` distribution architecture in use was archaic and proprietary, and those most familiar with it were no longer employed.

Furthermore, reinstalling the workstations using the `rdist` new-install process would have taken far too much time, as each install generally took about a half day, and due to network bottlenecks (much of the install was network-based), no more than four or five machines could reasonably be installed at any given moment.

The IT staff's primary insight was that there were two classes of vulnerable machines: servers and workstations. The attack required a user-level shell account on the target machine in order to work, and the attacker had compromised at least one or two student accounts (as indicated from the logs of the failed attacks). Student accounts, however, do not have access to the servers, so the likelihood of an infection on those machines was less than on any given workstation, as long as the staff assumed that the attacker had not compromised any administrator accounts. The staff brought down each server and ran several rootkit checkers. They also manually inspected the logs for indications of an attack. Seeing none, they patched the servers and brought them back online.

The staff then performed a standard (half-day) new install on a single workstation via the master server. While this new, clean workstation was installing, the staff used the time to analyze workstations of many different configurations to determine the minimal set of configuration files that would differ per machine. Once the first workstation was finished installing, the team went to each remaining workstation, booted to a LiveCD, and inspected the configuration files which were to be left untouched to verify that they contained nothing malicious.

The staff members then downloaded a script from the local intranet and ran it. This script erased most world-writable locations on the machine (`/tmp`, parts of `/var`, etc.). It then synchronized the remainder of the local filesystem (with the exception of the wiped partitions and the workstation-unique configuration files) directly from a known-clean workstation. Staff then re-configured and re-installed the bootloader, and the workstations were back online.

Once the single clean workstation had been cloned, it was possible to use the newly cloned machines themselves as `rdist` masters for other machines. For example, choosing masters on the same local switch allowed for a dramatic decrease in the time needed for the entire recovery. Note the level of detail and manual effort involved in starting and evolving the repair and recovery process, which includes a heuristic learned through direct experience with reinstalling machines in a localized fashion.

3.3 December 2007 Attack

Early in 2007, four new machines arrived at the department, intended to be used for high-performance graphics. Each machine was equipped with a high-end NVIDIA graphics card. No official Linux drivers for these graphics cards existed, so staff used unofficial drivers. In the first week of

December 2007, all four machines stopped working. The IT staff installed updated (and now official) graphics drivers. The next day, all four machines crashed.

The staff pushes out updates to all RHEL machines through its two `rdist` servers, `starsky` and `hutch`. `starsky` is the primary master `rdist` server, and `hutch` is a secondary. The infrastructure accomplishes upgrades with a two stage process. In the first stage, the active RHEL installation on `starsky` is upgraded, manually imaged, and copied to `hutch`. A `cron` job on each of these machines pushes the upgraded image out to half of the 150 machines[2]. The staff installed the updated NVIDIA driver on `starsky` to prevent subsequent `rdist`s from overwriting it on the graphics machines.

In addition to handling the NVIDIA issue, the staff also upgraded the kernel on `starsky` from version `2.6.9-55.0.9.EL` to `2.6.9-55.0.12.EL`. At 4 AM, the cron job delivered the upgrade to all 150 machines. On 10 December 2007, the staff discovered that both `starsky` and `hutch` had crashed. The staff attributed the failure to the recent upgrade, and investigating it was added to the end of a long task list for one of the staff members. Both machines crashed again on several subsequent nights.

The issue was explored on 13 December 2007, and the recent patches were rolled back on `starsky`. That night, both machines crashed again. This was a strong indication that the patches were not the problem, so an attempt was made to re-upgrade `starsky`. The upgrade failed when, during kernel compilation, the `mkdir` command returned an error. On the morning of 17 December 2007, exploration of this error determined that `mkdir` failed when creating directory names consisting only of numerals. IT staff began to suspect a rootkit. Booting to a LiveCD confirmed that suspicion: several files, including `mount`, had been replaced.

The staff's hypothesis is that the rootkit conflicted with the kernel module of the NVIDIA driver. If the attack took place in the first week of December, the rootkit would have been pushed to the graphics machines, a conflict ensued, and the machines crashed. Installing the driver on `starsky` caused that machine to crash too. The near-simultaneous kernel update obscured the real issue.

3.4 December 2007 Response

Discussion and planning for the response took place in a hallway at around 1pm on 17 December 2007. The planning group was assembled informally and consisted of the IT manager, three IT staff, and two authors of this paper, who happened to be nearby and were drawn into the meeting.

Initial discussion surrounded disagreements on the scale of the attack and the nature of the exposure. There was a brief argument over whether the `rdist` servers could be re-imaged and a clean install pushed out to all machines. This idea was discarded because it was recognized that all 150 machines would have to be reformatted from scratch. There were a number of questions that were raised immediately. What, if any, changes should we make to the system architecture? If we make changes, what machines (and order) should we roll those changes out to? Who will be involved? Staffing shortages imply that any changes beyond the simplest would take weeks or months to put in place. How will changes affect end users? Finals week is in progress, so taking large clusters of machines offline is undesirable.

Discussion immediately centered around whether the staff should migrate from Red Hat Enterprise Linux to another operating system.[3] OpenBSD was proposed and discarded, primarily due to the IT staff's unfamiliarity with the operating system. One staff member argued for Ubuntu. The IT staff has high turnover, so there was no RHEL expert currently employed, and there were no individuals present who were capable of competently comparing RHEL and Ubuntu. Lacking any qualitative (let alone quantitative) comparisons, no strong opposing voices emerged, and the Ubuntu motion carried. Discussion moved on to the user directory and authentication system. One member of the IT staff had a pre-built LDAP server in place, so movement to LDAP was quickly agreed upon.

The agreement of those in the meeting was that a new network, independent of the existing network, had to be created, and each account had to be re-created with fresh authentication credentials (passwords, SSH keys) in the new network. Since it was finals week, most machines were under heavy use. An underused 8-machine cluster was proposed as a testbed for the deployment, and the group agreed that that cluster should become the testbed for the Ubuntu rollout.

Since it was possible that the attack had been an insider attack (perhaps aimed at gleaning final exam information), the highest priority was to build clean Ubuntu images for the faculty. Thus, the faculty and finals remained the first critical concerns. The December 17 meeting then broke up, and the IT staff began work.

3.5 March 2008 attack

The March 2008 attack was detected by a member of the IT staff who noticed a new account named `mysqld` with root privileges on an important web server. Examining the contents of the home directory of this account showed several interesting files.

1. `.bash_history` containing what is probably a partial record of the attacker's behavior.
2. `ali.txt` containing the results of an NMAP scan for port 5555 (`freeciv`) across a /16 network.
3. `bot.pl` An IRC-based bot engine.
4. `dos.pl` A simple denial-of-service engine.
5. `xpl.c` Source code for the recently-revealed vmsplice Linux local privilege escalation exploit.

The `mysqld` account appeared in the `lastlog` history, along with the attacker's source IP address. Searching for that address in the Apache web server logs indicated that the attacker had repeatedly requested several files in a directory containing a common PHP web application, which was several revisions out of date, with remote exploits in the wild. The attacker added a copy of the `nsTView` remote web administration tool to the web app directory, leaving it set up with the default password.

The Apache logs also indicated that the attacker had downloaded a file he had created called `secret.txt`, containing the username and password for the web application's MySQL

[2]The unfortunate consequence of this architecture is that a compromise on `starsky` would be pushed out automatically to the entire network.

[3]We note that were was no *a priori* reason to blame RHEL for the intrusion, and we question whether this was an appropriate first topic for the response team to examine.

database, and the IP address for the remote host on which the database was running. Unfortunately, logging was disabled on the MySQL database, so investigations are limited in that direction. It is unknown whether the attacker ever connected to that database, or used one of several MySQL privilege escalation attacks to examine any of the several other databases running on that server.

3.6 March 2008 response

The response to the attack began by removing `mysqld` from `/etc/passwd` in order to disable it. The MySQL server daemon was shut down shortly thereafter. The owner of the vulnerable web application was then contacted and it too was shut down. These responses were performed quickly – within two hours of the attack first being detected – and then the response turned to a policy discussion. What architecture and policy changes need to take place to prevent such attacks in the future?

As of the time of this writing, no policy decisions have been made.

4. DISCUSSION

We next highlight some of the key decisions, discuss why they were not based purely on technical considerations, and suggest research directions. Note that our purpose is not to pass judgment on a particular decision by labeling it "good" or "bad": the central goals of our analysis are to observe how non-technical factors influenced decisions and to highlight what kinds of technical systems might be constructed to help manage that influence.

4.1 Observations

The first minutes after an intrusion discovery, in which there is no complete information about the attacker's entry point(s), history of actions, short and long-term intent, or current level of activity, hold the potential for panic, an overwhelming amount of data to analyze, and a paralyzed thought process. The hallway discussion on 17 December involved multiple people, ideas, and proposals. The system administrator involved with our case study achieved a certain level of success at repairing the network only because he was able to quickly assemble a key team of people and rapidly sift through the different proposals that the team members articulated.

Lesson 1: In the first minutes after discovery of an extensive intrusion, rapidly setting a diagnosis and recovery agenda requires clear, informed decision making.

Decision making at this point should be aided by automated processes that help manage the signal-to-noise ratio; in studies on decision–making, the manner in which information is organized often appears more important than simply getting increased amounts of information [7]. It is clear that the IT staff did not have an *a priori* idea of what procedures should be enacted to combat or rectify the intrusion or to process and prioritize information about the incident. While the lack of a disaster recovery plan is a major operational shortcoming, disaster recovery plans alone are not a panacea. The plan, like any proactive defense method, may be incomplete, outdated, or unlikely to work given the current personnel. For example, the IT manager faced a critical personnel shortage due to events unrelated to the intrusion: half the staff was leaving for new jobs in a matter of days.

Lesson 2: Designing and maintaining a disaster recovery plan can aid recovery efforts, but the plan must be continuously — not periodically — updated.

A disaster recovery plan must constantly evolve. Each new attack, vulnerability, or patch affects the recovery details. Similarly, employee turnover, improved employee skill set, and application deployment require modifications to the plan. The question of how often to update the disaster recovery plan is a risk analysis and assessment task that must balance the needs of the staff to accomplish everyday system administration tasks against spending an inordinate amount of time planning for disasters that might never occur. The open research question here is how personnel changes, catalogs of personnel skill, and lists of resources, sensors, countermeasures, toolsets, and inventory can drive an *automated, real–time* update of the disaster recovery plan.

Staff may be torn between a number of actions, including continuing diagnosis and forensic efforts, fixing the immediate problem or small–scale symptoms of an attack (turn off a particular service, unplug a particular machine, remove a log-in entry from `/etc/passwd`), and fixing the larger–scale symptoms or root causes of an intrusion. Somewhat ironically, the technology that the staff used to plan out recovery activities included a whiteboard and a marker. The whiteboard was recently and inadvertently erased. The marker remains at large.

People often prefer more natural interfaces like whiteboards than a ticketing system accessed via a GUI. Usability research on display–centered group activities has found that displays are important in the planning stages of the activity, but grow progressively less useful as the plan is enacted [4]. The core research question centers on how to create a system that continuously learns about the environment it is deployed in, automatically creates scenarios, and makes recommendations for recovery actions. The system should also be introspective in that it reasons about the quality of the actions taken based on its recommendations.

Lesson 3: Human memory and recall is far from perfect; multiple points of view supply sometimes conflicting details of attacks and do not assist efforts in forensics, auditing, or planning for the next attack.

During our interviews, we observed that details of the attacks and the responses often differed wildly between individuals. Individuals often disagreed on dates — one person confused an attack from March 2007 with one from May 2006 and provided a mixture of details from both. In other cases, individuals presented radically different reports on which actions were taken. Two members of the IT staff disagreed on the date and method of detection of the December 2007 attack, while another viewed it as a continuation of the March 2007 attack. Without a coherent view of the state of the network, it is difficult for staff to make informed decisions to guide the attack response.

Even though researchers have proposed work on attack trees [6, 8], relatively little attention has been paid to analyzing the process of a response. Automatically increasing the rate and types of events logged after an intrusion is discovered and the recovery process is started can assist efforts to revise a disaster recovery plan. More logging can make sure that key decisions are clearly recorded and not subject to human recollection of events occurring during a stressful time of rapid change and high rates of information.

Lesson 4: Decisions about appropriate technology shifts are driven by informal personal inclinations.

Making changes to a complex and corrupted infrastructure requires (besides a quality analysis of the intrusion) a good understanding of the benefits offered by selecting one technology over another. For example, when the staff discussed whether to change computing platforms from RHEL to Ubuntu, the decision was made without any point–by–point comparison of the *security* benefits of either system. Although a question was raised about whether Ubuntu incorporated SELinux by default, as RHEL does, it was neglected (a symptom of the need for Lesson 3). The staff expressed comfort with Ubuntu's package management software[4] and indicated that one staff member had already prototyped an Ubuntu system that would support stronger authentication.

In this case study, the IT staff did not perform an examination of the release notes of the latest versions of the operating systems under consideration. While the circumstances and the time pressure demanded a quick decision, it would be best if the IT staff were not placed in such a bind to begin with. Providing systems that automated these types of comparisons and parameterizing them with the details of the intrusion or incident can assist staff efforts to make rational, informed, and *technical* decisions rather than intuitive ones.

Lesson 5: Staff do not have the luxury of complete forensics.

After detecting an incident or intrusion, it is difficult to immediately identify and execute the appropriate next steps; a staff is effectively in the middle of diagnosis. Undertaking an effective forensics process is challenging. There is a tension between short-term operational demands to keep services running and long-term demands from the ISP to keep a network clean. Disks and machines have to be kept in use; we don't have the luxury of taking them offline for extensive cleanroom analysis. But while operational demands are important, the forensic analysis they preclude might reveal information which ultimately proves more critical.

Lesson 6: Incidents are not always detected by IDS systems

All three attacks were discovered manually through symptoms and side–effects of each attacker's activities rather than traditional intrusion sensors like Snort or a commercial antivirus product. This situation suggests that alert and educated IT staff and users are critical to uncovering stealthy attacks. We note that the sample size of incidents is small and purposefully focused on extensive intrusions (rather than well known worm infection attempts). This lesson should be taken as a call to focus on creating anomaly sensors that span multiple levels of a system.

5. CONCLUSION

Currently, repairing a network infrastructure after a serious intrusion is largely a manual process. Furthermore, the psychological and sociological aspects of the problem are grossly understudied. Systems involve people, and their security decisions and risk assessments are often based on reasons that are not purely technical. The purpose of this case study is not to question whether the IT staff could have done a better job, or if the organization should have had a more robust network to begin with.

Instead, the lessons we should learn are that real security problems — those whose scope is sometimes too large to comprehend and deal with in any single research publication, are brushed aside as either too large to be interesting, or too close to human and organizational problems to be strictly "systems" security issues. With this case study, we hope to show that interesting possibilities for systems security research exist. Fundamentally, we think that human decisions should be assisted with automated methods that help filter and classify the available information. The problem of network intrusion recovery is a particularly thorny exercise in researching, designing, and creating usable security mechanisms.

[4]While good package management software can greatly ease the job of system administration, it should not be the primary or only factor in a security–related decision.

6. REFERENCES

[1] L. Carin, G. Cybenko, and J. Hughes. Quantitative Evaluation of Risk for Investment Efficient Strategies in Cybersecurity: The QuERIES Methodology. *To appear in IEEE Computer*, 2008.

[2] W. R. Cheswick and S. M. Bellovin. *Firewalls and Internet Security: Repelling the Wily Hacker*. Addison-Wesley, 1994.

[3] G. W. Dunlap, S. King, S. Cinar, M. A. Basrai, and P. M. Chen. ReVirt: Enabling Intrusion Analysis Through Virtual-Machine Logging and Replay. In *Proceedings of the 2002 Symposium on Operating Systems Design and Implementation (OSDI)*, February 2002.

[4] E. M. Huang, E. Mynatt, and J. P. Trimble. Displays in the Wild: Understanding the Dynamics and Evolution of a Display Ecology. In *Proceedings of the* 4^{th} *International Conference on Pervasive Computing*, May 2006.

[5] S. T. King and P. M. Chen. Backtracking Intrusions. In 19^{th} *ACM Symposium on Operating Systems Principles (SOSP)*, October 2003.

[6] X. Ou, W. F. Boyer, and M. A. McQueen. A Scalable Approach to Attack Graph Generation. In *Proceedings of the* 13^{th} *ACM Conference on Computer and Communications Security (CCS)*, October 2006.

[7] J. W. Payne, J. R. Bettman, and E. J. Johnson. Behavioral Decision Research: A Constructive Processing Perspective. *Annual Review of Psychology*, 43:88–131, 1992.

[8] O. Sheyner, J. Haines, S. Jha, R. Lippmann, and J. Wing. Automated Generation and Analysis of Attack Graphs. In *Proceedings of the IEEE Symposium on Security and Privacy*, May 2002.

[9] A. Singer. Tempting Fate. *USENIX login;*, 30(1):27–30, February 2005.

[10] E. H. Spafford. The Internet Worm: Crisis and Aftermath. *Communications of the ACM*, 32(6):678–687, June 1989.

[11] C. Stoll. Stalking the Wily Hacker. *Communications of the ACM*, 31(5):484, May 1988.

[12] G. L. Wells and E. A. Olson. Eyewitness Testimony. *Annual Review of Psychology*, 54:277–295, 2003.