Making a Game of Network Security

Marc Dougherty – Northeastern University

ABSTRACT

This paper describes our experiences in the design and implementation of a network for security training exercises, and one such exercise. The network described herein is flexible, and could be used for a wide variety of training exercises. Organizations of all sizes can use this type of exercise to train new personnel or keep existing staff at their best. In addition to the training benefits, these exercises can also be highly entertaining. The designs described here are intended to assist others in the construction of similar networks, and additional training scenarios.

Introduction

Companies large and small invest a great deal of money in the training of security personnel and system administrators. However, there is little formal training available for individuals without a company willing to pick up the check. This leaves the majority of sysadmins to learn the old fashioned way: by making mistakes. This is especially true of aspiring sysadmins at the college level, where system administration and security are discussed as a side note, if at all. In light of this, we set out to create a safe environment where all types of sysadmins could practice their skills, without putting personal or business data at risk.

Inspired by the ''Capture the Flag'' competition held annually at Defcon [1], we sought to create a similar environment, with particular emphasis on security and teamwork. Security has become a higher priority for many companies, security-related spending has increased drastically, and as a result, many sysadmins have been ''promoted'' to specialized security roles, without any additional training. Teamwork is also a common weakness for sysadmins. Many sysadmins work individually, and few have much experience working with others. Team training exercises like those proposed here can help overcome these common weaknesses and be entertaining at the same time.

Because of the sysadmin predilection for experiential learning, informal training exercises like those described here can be enormously beneficial, both for the individual sysadmin and his or her employer.

The environment described here is the result of hard work from a group of students at Northeastern University, working jointly with the Systems Group at the College of Computer and Information Science.

The network we have created involves two smaller networks, connected by an OpenBSD machine which performs the majority of the routing, and provides limited access to the Internet from both networks via HTTP, HTTPS and SSH. Internet access is vital as it allows access to the most current patches, tools, and technologies for both defense and offense.

The first internal network, called the defender network, is populated with the machines that contestants administer. We provide contestants with a machine on this network, which they must defend. Contestant teams may choose their preferred OS from a set of alternatives provided by the contest admins. Members of the administrative team often sabotage default security features of these installations, making the defender's job more interesting.

The second network is a new addition to the contest, inspired by many requests of participants to bring along their own machines for use in the contest. The attacker network was created to allow participants to plug into the contest, while minimizing their risk of being attacked. A strict set of ACLs prevent incoming connections to machines on this network and prevent hosts on this network from communicating with each other.

The keystone of the network is the routing policy on the OpenBSD gateway machine. This routing policy is written entirely in pf, the OpenBSD packet filter. No incoming connections are allowed to the attacker network, but responses to existing connections are allowed. All traffic from the attacker network to the defender network is passed through unmodified, because much of that traffic will be crafted packets and exploits which should not be tampered with. Other duties of this gateway machine include serving DHCP to the attacker network and providing DNS, NAT, and Internet access for both internal networks. This machine is also responsible for the Nessus server which is used by the scoring system.

Since this scenario was designed to be as realistic as possible, there is a minimal rule set. The only form of attack that is disallowed outright is denial of service attacks. However, the rules allow the contest admins to deem the conduct of a team ''unsportsmanlike,'' and punish them accordingly. Punishments can range from a score penalty to expulsion from the contest.

The total score of each team is the sum of offensive and defensive scores. The offensive score is very difficult to automate, and has typically been handled manually. When a participant compromises a service, they are required to leave some type proof, which the contest admins will then verify. Defensive scoring is

based on the availability of the team's services. Periodically, the scoring system tests the functionality of each service on the contestant's host using several custom Nessus plugins written for this purpose. The state of these services is then recorded in a database. From the database, the information can take on any number of forms for presentation. Previous methods have included a text file, a web page, and an XML document processed by a Macromedia Flash application which displays the information in a more exciting way.

This contest has evolved to its current condition over the last several years, and continues to do so. Work is being done to move all routing policy into a Cisco Catalyst switch, which frees up the OpenBSD gateway machine to act as a web and ftp proxy for outbound connections. Also under development is a fully automated scoring system which merges the offensive and defensive aspects of scoring. The new system uses flag files to indicate when a service has been compromised. This scoring system requires significantly less manual intervention, and simplifies the lives of the contest administrators.

Because we have been unable to find a suitable metric by which to measure system administration skill, we cannot prove that the participants in our contest have become better admins. However, many participants have reported learning a great deal from the contest. In addition, many former participants have become members of the contest's administrative team, which indicates that the contest inspires participants to strengthen their teamwork and security skills.

Network Layout

Designing a network for use in security training involves finding a delicate balance in communication policy, to assure that participants are given enough freedom to use various attack techniques, without granting them the ability to attack machines outside of the scope of the training environment. The environment we have created consists of a defender network segment, and an attacker network segment. These networks are connected to each other, and to the Internet, by a gateway machine that performs routing and packet filtering.

Access to the Internet is provided to allow participants to research protocols and programs with which they are not familiar. The defender network is populated with machines created by the contest administrators. The attacker network must be able to attack the machines on the defender network, while remaining protected from the attacks of others. Both networks must be prevented from launching attacks on other machines outside of the training environment. The establishment of proper routing policies is critical to the success of this network. If the routing policies are not restrictive enough, this network could be used to launch attacks to the outside. On the other hand, if the policies are too restrictive, they may interfere with the intended use of the network by blocking some types of anomalous traffic.

Internet Access

The simplest of the network policies involved is the routing for the Internet connection. Both the attacker and defender networks are permitted access to the Internet via HTTP(S) and SSH. All other traffic bound for the Internet is dropped. In the past, we have allowed teams to participate remotely by forwarding external ports on the gateway to hosts on the defender network. Each machine on the defender network is accessible by Remote Desktop and Secure Shell. In the most recent incarnation of the contest, we encouraged participants to physically attend, eliminating the need to provide remote access.

Defender Network

The defender network is home to the machines that the contestants must defend. In order to give participants the opportunity to experiment with sniffing tools, the defender network has typically been hubbed rather than switched. This network uses private, non-routable IP addresses in the 10.10.0.0/24 range [2], which are provided by the gateway. To decrease the potential for abuse of this network, Internet access is limited to HTTP(S) and SSH. To minimize the vulnerability of machines on the attacker networks, defender machines may not initiate connections to the attacker network.

Attacker Network

The attacker network was a new addition in the most recent incarnation of the network and its design posed an interesting challenge. The goal of this network was to allow participants to bring along their own machines for use the attacking portion of the exercise, while minimizing the risk associated with plugging into such a hostile network.

The attacker network uses private non-routable IP addresses in the 10.20.0.0/24 range, which are provided by the DHCP server on the gateway. Packets from this network destined for machines on the defender network are passed through with no modification since many of these packets are likely to be exploits or other types of malicious traffic. The attacker network is shielded from incoming connections from the defender network by the defender network's filtering, but attackers are still susceptible to attacks from fellow attackers. The ideal way to isolate attackers from each other is to create a separate network for each attacker. Since we lacked the hardware resources to do so, the solution we have implemented relies on ACLs to control the network traffic of attackers.

Using a Cisco Catalyst 3550, attacker communications can be limited with VLAN-layer ACLs. The attacker network ACLs prohibit machines on the attacker network from communicating with any other host on the attacker network, except for the gateway

machine's attacker network interface. This approach is significantly simpler than segregating each attacker on their own VLAN, and is more scalable. Even with these countermeasures in place, all participants are forewarned that they are plugging into a hostile network at their own risk.

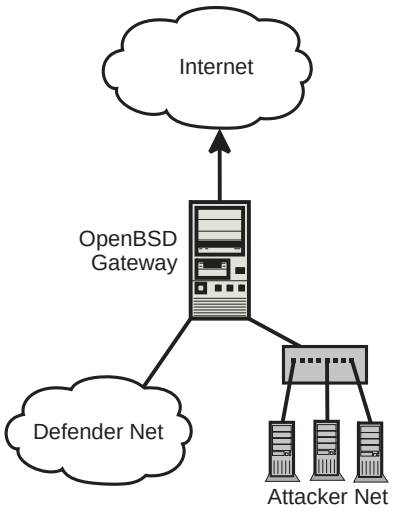

Figure 1: All routing, policy (and proxy) functions.

Gateway and Routing

Because the routing and communication policy between networks is so important to the success of the network, the policy was implemented using the method with which the administrators of the network were most familiar. Currently, the routing policy is written using OpenBSD's packet filter, pf [3], but there is work in progress to port this configuration to Cisco IOS.

The packet filtering rules were created using a set of class-based queues designed such that administrative tools like Secure Shell and Remote Desktop are allotted 20% of the available bandwidth, and given a higher priority than other traffic on the network. This was done to minimize the negative effects of network saturation on any segment of the network.

We also learned – the hard way – that the default size of the OpenBSD state table is far too small to run a contest like this. Shortly after the contest began, the state table was filled to capacity and stalled network communications. We have since increased the state table to store 20,000 states, and optimized the state timeouts to be more aggressive.

The hardware used for the gateway was a 500 MHz Pentium II with 256 MB of RAM and three network cards. Although this may not seem like a lot of computing power, the machine performed well under the strain of the contest. The primary duty of this machine is the routing between networks and performing NAT for both the attacker and defender networks. In addition, this machine also provides DNS resolution for the attacker and defender networks using DNSMasq [4] and provides DHCP leases to machines on the attacker network. In the future, this machine may also serve as a web and ftp proxy for the attacker and defender network.

This network began as a simple test network and has evolved into a secure environment for network security testing. The potentially harmful traffic is isolated to this network, while still allowing participants access to the Internet. This network provides a safe environment in which sysadmins of all skill levels may experiment without endangering any important data. This environment can be used to stage a wide variety of testing scenarios, training sessions, and security challenges.

The Game

One possible use for the above network is a game that we have created, based primarily on an annual Defcon tradition now known as Root-Fu [5, 6], formerly known as Capture the Flag. Each team of participants is assigned a host on the defender network, which they must secure against the attacks of the other teams. Teams are awarded points for each successful test of their services, and each successful compromise of an opponent's service. Teams must balance offensive and defensive tactics to gain as many points as they can over the duration of the contest. However, there are some restrictions on the tactics that may be used in this contest.

Rule Guidelines

Laying down rules for a security challenge is more difficult than it seems, and enforcing rules is even more difficult. Strict rules may prevent participants from making use of some useful attack strategies. Rules that are too lenient could result in teams using unfair methods to give themselves an advantage. The current rules do not allow for ARP spoofing or Denial of Service attacks, because these techniques have been used to disrupt the contest in previous years. The rules allow the contest administrators the ultimate authority in determining if a team's behavior constitutes ''unsportsmanlike conduct,'' and penalizing them accordingly, either with scoring penalties or expulsion from the contest. To spot potential violations, contest administrators keep a close eye on traffic in the defender network. Contestants are also provided with a private, direct means of communication with the contest admins, in the event that they suspect their opponents of violating the rules. Fortunately, rule violations have been rare since most contestants understand the spirit in which the game is run.

The newest addition to the contest rules is the concept of machine ownership. If a team has gained administrative access on another machine on the defender network, the team may request ownership of the machine. This is done by filling out a ''Change of

Ownership'' report in the contest web tools, discussed later. The contest administrators then update the ownership of that machine in the database, and the team begins to earn defensive points for any services offered by that machine.

Getting Started

The first step in getting a competition like this running is recruiting participants. Each year, several members of the administrative team are tasked with creating advertising for the contest. Past advertising campaigns have consisted of posters and word of mouth, but more recently we have invested in a free-standing sawhorse sign featuring a skull and crossbones. Once participants are interested, they will need to register.

Team registration is done using a web-based tool that stores registration information in a database, which is used later for scoring. The registration form includes all the basics like name and contact information, in addition to an Operating System preference, which allows teams to choose a preference from a list of available OS choices. This list includes at least one Unix variant, and one Windows variant. The registration system should be made available no less than one month before the scheduled beginning of the contest. This insures that interested individuals have enough time to find other interested parties to join with in the creation of a team. In order to give the administrators enough time to complete the initial setup, registration should close approximately one week before the contest begins. Once the number of registered teams is known, the real work can begin.

Initial Contest Setup

The most time-consuming portion of running this contest is the building of machines destined for the defender network. In addition to the drudgery of OS installation, members of the administrative team often sabotage the default security of the OS by adding administrative users with easily guessed passwords, or disabling common security features. In order to grant the teams access to their machines, the administrators of the contest must maintain a list of the passwords set for the Administrator/root account, so that the passwords can be distributed to the contestant team. Since this is the least interesting part of running the contest, there are efforts under way to automate this part of the procedure.

In early contests, the participants focused primarily on defense, which made for a relatively unexciting contest. In order to avoid this, a new class of machines also inhabits the Defender network for this competition. A variety of Non-Player Computers, or *NPC*s, were set up, and deliberately left vulnerable, to give participants something at which to point their attacking skills. These NPCs are built from spare hardware found in storage rooms, and frequently include devices like printers, or similar networked devices, just to keep the competition interesting. Often, the task of setting up NPCs is delegated to a newer student, without much experience in system administration. The student can then observe his NPC to see how well the configuration fares against the attacking skills of the contestants and learn from any mistakes. In this way, even students involved in the administration of the contest can gain experience.

Since the use of an Attacker network requires that the contestants be in physical proximity to the rest of the contest network, the contest now needs some physical space in which it can be held. This also requires that the contest be mobile enough to relocate for the duration of the contest. Fortunately, the CtF administrative team was able to obtain a spare 19'' rack, in which they mounted a keyboard, mouse, monitor, and 16-port KVM switch. The bottom half of the rack was used to house the contestant machines, and the OpenBSD gateway machine described earlier, which also handles some of the duties of scoring.

Contest Timetable

On the first day of the competition, a fair amount of time must be devoted to administrative details and the rest is reserved for teams to work on securing the machine they have been assigned. The first administrative task is distribute information packets to each team. The administrators must verify the identity of the team leaders and the team leaders are then entrusted with the information packet. This information packet contains an official copy of the contest rules and the password for the administrative account on their assigned machine on the defender network. Also in the packet of information are instructions for connecting to the contest network from elsewhere on the Internet. With the creation of an attacker network, the remaining members of the administrative team are making sure that the machines contestants have brought with them are properly connected to the attacker network.

For the first day, participants are not allowed to attack their opponents. A member of the administrative team watches the network carefully, using Snort [7] as an Intrusion Detection System to catch any teams who violate this policy.

On the second day of the contest, the gloves come off, and the participants may begin to attack each other. For the remainder of the contest, the machines on the defender network will be the target of countless port scans and exploits. The second day of the contest also marks the beginning of the scoring period, which continues until the end of the contest.

Scoring Mechanisms

The scoring mechanism used in this contest is divided into two types: defensive and offensive. The defensive portion of the score is derived from the availability of the services offered by the team's assigned machine, as tested by an automated system. Offensive scoring has proven very difficult to automate,

and is currently handled through a web-based reporting system discussed below.

In order for an automated service checking system to be effective, it must accurately simulate user interactions with the service. If the service verification is not thorough enough, a team could replace a complex service with a simplified ''stub'' service. Fortunately, the Nessus remote security scanner [8] provides a flexible plugin architecture suitable for the needs of this contest. Nessus plugins can be written in either C, or the Nessus Attack Scripting Language (NASL) [9]. All contest-related service tests are written in NASL, and report success or failure using two of the built in Nessus plugin return states. The assignment of status is arbitrary, but must be consistent with the Nessus output parser.

Defensive scoring is divided into ''rounds'' that last ten minutes. During each round, a Nessus client connects to the Nessus daemon running on the gateway and requests a test of the defender network using only the contest-specific tests. After the server performs the checks, the data is fed back to the client as XML. The Nessus output is then parsed using XPath expressions and the result of each test is recorded in the scoring database. Storing the test results in this manner not only allows multiple tests for a single service, but if a test proves to be unreliable on the contest network, it can be disqualified without losing any additional scoring data. SQL queries can then be used to determine which services on each machine passed all tests, and should be awarded points. Point values for each service are also stored in the database, so that a team's points could be calculated using a single SQL query.

Of course, the score of the game is not nearly as exciting unless it is displayed for all to see, so some of the more graphically gifted members of the administrative team put together a scoreboard using Macromedia's Flash [10]. The scoreboard retrieves a simple XML file containing various scoring-related quantities, and displays them in interesting ways. For example, the percentage of running services is represented as a gauge, and the overall team score is displayed simply as a number. There are many other quantities that can be measured in this setting and these quantities are presented only as examples. The scoreboard is a valuable addition to the contest, as it allows contestants and spectators alike to get a clear picture of the contest standings.

For the remainder of the contest, the only task for the administrative team is to respond to feedback from participants. Teams may provide this feedback to the administrative team using a set of web tools that were created to help the contest run more smoothly.

Web-Based Administrative Tools

The CtF web tools were created to facilitate communication between the contest admins, and the contestants. As the contest became more complex, so have the tools. The tools were originally written using AxKit [11], but work is underway to remove that dependency. The registration tool is not protected by authentication, but all the remaining tools require a team username and password unless otherwise noted.

The most fundamental of the web tools used in the administration of the contest is the registration form. The registration form is filled out by the teams before the contest, and includes information such as team name, OS of choice, and the name and contact information for each member of the team. This information is recorded in the scoring database for later use by scoring software and web tools.

When a team compromises another machine on the defender network, they must fill out an offensive report. The offensive report requires that a team specify the IP address of the machine they have compromised, and provide the administrators some measure of proof that they have gained elevated access on the machine. This can be done in a number of ways, including modifying web content or binding a shell to a specific port. Any proof of compromise must be verifiable without local access to the machine, since the contest administrators do not maintain any level of access to contestant machines. The administrators then use a correspondence web tool to reply to the teams, indicating whether their proof was sufficient or not.

In the event that a team's assigned machine becomes unusable, the team may use the web tools to request that their machine be re-installed. Re-installation is done at the convenience of the contest administrators, but once rebuilt, the machine will come up unpatched and mis-configured on a hostile network. Because of this, re-install requests are rare, and may be removed from the contest in the future.

If a team gains complete control of another machine on the defender network, they may request ownership of the machine, in order to gain more defensive points for services offered by that machine. The web tool requires that the team provide the special flag, stored on a CDROM, and readable only by the administrative user. When the contest administrators receive a change of ownership request, the ownership of the machine is updated in the scoring database, and the flag on CDROM is changed to prevent the previous owners of the machine from re-claiming it. The replacement scoring system in development handles ownership at a service level, rather than a host level, so this system will be obsolete.

Teams are also encouraged to turn each other in for violations of contest rules, since the contest administrators cannot hope to catch all violations themselves. If a team believes that they have found a rule violation, they fill out a web form detailing whom they believe to be violating the rules. The contest administrators will then investigate and respond to the team who reported the violation.

If a team wishes to convey a message to the contest administrators that does not fit into one of the categories above, a general comment tool is also provided.

The administrative side of the web tool suite presents the user with a list of reports that have not yet been handled. Resolving a report emails the contest administrator's comments to the captain of the team who filed the report. These tools allow the contest administrators to efficiently handle feedback from contestants.

This particular game has evolved to its current state over the last few years and is only an example of the many possible uses for this network. Several similar games are in development, including a ''King of the Hill'' type game where contestants would attempt to gain root access on a machine, and attempt to maintain that access as long as possible. Both the network setup, and the game have evolved a great deal to reach their current status, but there is much more work still being done to improve them.

Future Improvements

Over the years, this contest has improved as members of the administrative team have found ways to automate and simplify various aspects. In the spirit of continuous improvement, there is still a great deal of work in progress to make the network easier for administrators to set up and the game more enjoyable for the participants.

Network Improvements

With the creation of the attacker network, the administrators found that the network was transferring significantly more data than it had before, due to downloading and web browsing by the participants. In the past, contestants connected to the contest from their homes, so the contest was not responsible for providing normal access to the Internet. However, with the advent of the attacker network, the contest network must provide enough bandwidth for teams to research service protocols, and investigate possible attack vectors.

In order to achieve this, the routing policies described initially will be enforced by the Cisco Catalyst 3550, which performs many routing functions in hardware. Relocating this functionality will increase the throughput of the contest network, and free up the gateway for other tasks.

One task that the gateway's resources could be used for is a proxy server. Currently, the routing configuration of the gateway does not have the ability to restrict traffic to the HTTP and HTTPS protocols, merely to their respective ports. Using an application-level proxy would reduce the potential for misuse of the network by ensuring that only valid requests pass through.

Game Improvements

The most time consuming task involved in building the contest is the bulk creation of machines for use on the defender network. Although identical machines can easily be constructed using disk imaging software, a network full of identical machines is not very interesting. Research is currently under way to investigate the potential use of other configuration management systems, such as Fully Automated Installation [12], or Radmind [13] to create basic installations that are similar, but not identical. The use of such systems would drastically reduce the amount of work required for contest setup, and may allow the contest to be run more frequently.

Automation of the defensive scoring for Capture the Flag saved the administrative team quite a bit of time and effort, while automating the offensive scoring could save even more. The automated scoring system in development combines the defensive and offensive scoring into a single system based on dynamic service flags.

Each service must make the flag information available to all clients, and the flag must be located in a pre-determined location for each service. For example, the web server on host 10.10.0.3 would make its flag available at http://10.10.0.3/flag.txt. Other services must make their flags available in a similar manner.

At the start of the contest, each team is issued a special initialization flag. When a team takes control of a service, they replace the existing flag with their own initialization flag. In the next round, the scoring system will recognize that the ownership of that service has changed.

Each round, the scoring system connects to a scoring daemon on the target contestant machine, and retrieves the flag for each service from the filesystem. The scoring system then connects to the service itself and performs a series of validation tests, which includes retrieving the flag for the service. If the service flag from the filesystem does not match the flag obtained through the service, the contestant has attempted to trick the scoring system, and should be penalized. If the two flags match, they are compared to the expected value stored in the scoring system's database. If the flags match, the service is still under the control of the same team, and should receive points. If the flag retrieved by the scoring system matches the initialization key of another team, the scoring system updates the ownership of that service to reflect the compromise, but no points are awarded. This was done to deter teams from simply replacing their flags with initial flags each round.

If the flag does not match the expected value or any team's initialization key, the flag has been tampered with, and the contest admins should be alerted. As of this writing, the above system is under active development, although not yet completed.

Several contest additions have been proposed to add additional realism. The best way to add a realistic angle to the contest is to present each contestant team

with content for each service they run. Teams would be required to serve this content in order to earn points. Service tests could then be used to verify that the provided content was still in place. The number of services could also be narrowed, allowing contestants to focus their research efforts on a smaller number of protocols and allowing administrators to write more complex functionality tests for those services. Bandwidth usage and performance are important factors as well, and may be incorporated into the contest in the future.

The bandwidth used by each contestant machine could be tracked, and the teams could be ''charged'' some number of points, based on their bandwidth usage in each scoring round. The tracking of bandwidth usage could easily be performed by the switch, but this information must still be fed back into the scoring system. A similar ''charge'' could be applied for slow response to scoring checks. For example, if the response time for a particular host is significantly longer than the response of the other hosts, that host should be penalized for their poor performance. Implementing these penalties has not yet been attempted.

Although this contest has evolved for several years, there are still many more performance enhancements and features that are in development. Suggestions and feature requests are welcome.

Availability

The code and configuration files used in the contest will be made available at http://www.nerdcircus.org/ctftools/.

Conclusions

There are many reasons for an organization to run a competition like this, whether for educating new sysadmins, or just to keep current sysadmins sharp. Security training exercises like the one described here can be a great benefit to any organization. Such exercises can be used to raise awareness of common security problems and their solutions. In an academic environment, these exercises can be used to give interested students a safe environment in which they may explore many aspects of security, without endangering the security of others.

Corporate environments stand to gain just as much. Using the techniques described here would allow organizations to better evaluate the technical proficiency of the individuals being considered for security-related positions. In addition to entertainment, these training exercises could be used to keep existing members of security teams at their best.

The network described here can be built using hardware that many organizations may already have laying around in storage rooms.

Although these contests are a valuable learning experience, and have numerous other uses, the main reason that we have continued running them is simple: fun.

References

[1] Defcon, http://www.defcon.org.

[2] Rekhter, Y., B. Moskowitz, D. Karrenberg, G. J. de, and E. Lear, ''Address Allocation for Private Internets,'' *RFC 1918*, Internet Engineering Task Force, February 1996.

[3] *OpenBSD*, http://www.openbsd.org .

[4] Simon Kelley, *Dnsmasq*, http://thekelleys.org.uk/dnsmasq/doc.html .

[5] The Ghetto Hackers, *Root Fu!*, http://ghettohackers.net/rootfu/ .

[6] divide and dd, ''Root-Fu; Rise of the Ninjas,'' In *Toorcon 5*, http://www.toorcon.org/slides/rootfu-toorcon.ppt , 2003.

[7] Roesch, Marty, *Snort*, http://snort.org .

[8] Deraison, Renaud, *Nessus*, http://nessus.org .

[9] Deraison, Renaud, *NASL 2 Reference Manual*, http://nessus.org/doc/nasl2_reference.pdf , February 2002.

[10] Macromedia, Inc., http://www.macromedia.com .

[11] Apache Software Foundation, *AxKit*, http://axkit.org .

[12] Lange, Thomas, ''FAI – Fully Automated Installation,'' *Eighth International Linux-Kongress Proceedings*, November 2001.

[13] Craig, Wesley D. and Patrick M. McNeal, ''Radmind: The Integration of Filesystem Integrity Checking With Filesystem Management,'' *LISA '03 – Large Installation System Administration Conference*, October 2003.