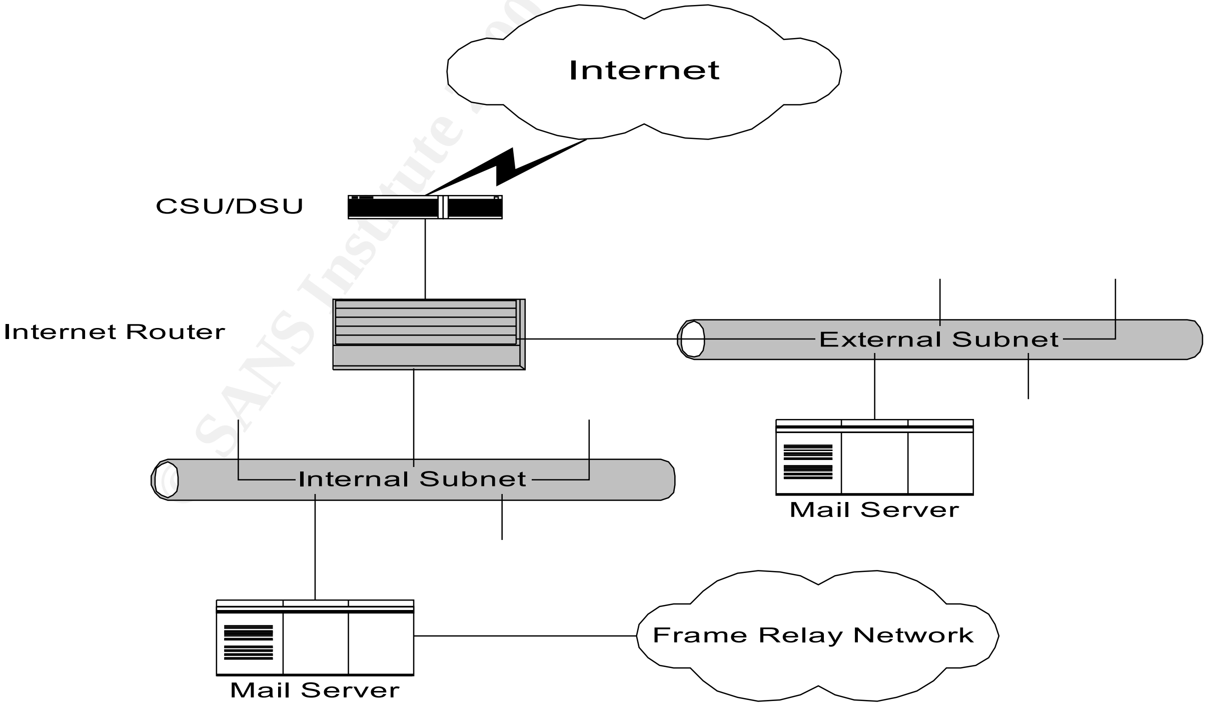
I. Executive Summary

This incident occurred several years ago, when I was a System Administrator at a large multinational corporation. I was responsible for about twenty Unix servers. At the time, the machines were roughly equally divided between DEC AlphaServers running Digital Unix and Suns running either Solaris 2.5.1 or Solaris 2.6. Most of the machines were dedicated to a single function. About 6 of them were DNS servers, 2 external and 4 internal. Another 3 were used as mail servers. A pair of them were news servers. And the rest served a variety of functions – Oracle dB server, 2 web servers, etc. At the time our “firewall” was actually a router access list. We had a large number of frame relay customers and one of the services we performed for the frame customers was to receive e-mail for them through our Internet gateway and then relay it across the frame network to their mail servers. To accomplish this we had a pair of mail servers, one on each side of our “firewall”. All of the MX records pointed to the server outside of the router. I had modified the sendmail.cf file so that the external server relayed all of its mail to the server inside the router. The router itself was configured so that only mail originating from the machine outside of the firewall was passed to the inside machine. Everything else was blocked. Both of these machines were DEC Alphas
.



Early one morning I discovered that one of the AlphaServers had been hijacked and was being used to send spam to thousands of AOL accounts. Because the outside server relayed all of its mail to the inside machine, thousands of spam messages were being received into our corporate mail gateway and sent back out to AOL.

Fortunately, neither server had not been broken into. The spammer had found an easy way to exploit a vulnerability in sendmail. The hijacked machine also ran DNS for all of the domains for which it relayed mail. But only for its own internal use. This was done to speed up the delivery of mail.

Because of the volume of mail we handled, mail logs were only kept for seven days, so it was impossible to say how many times the server had been hijacked prior to this incident. Upon inspection of the logs, however, there was evidence of two other security events. Neither of them could match the scale of the incident that was happening that morning and had gone unnoticed.

Thanks to the cooperation of the ISP from which the spam originated, I was able to stop the flow of the mail and regain control of the server within a couple of hours of the discovery. By taking the server offline, I also was able to purge several thousand messages before they could be sent. Ultimately, the sendmail program had to be upgraded to the latest version to prevent a reoccurrence of the problem.

II. Preparation

Surprisingly, despite the size of the corporation, there was very little in the way of a corporate security policy. At one time most of our corporate security had originated out of a business unit that specialized in defense work. This business unit had been sold about two years prior to this incident, however, and the burden for security had fallen upon each remaining business unit. Several other business units were in the process of being sold and very little remained in the way of a corporate policy.

At the site where I worked, there was no security officer, no written policy, other than a warning from HR concerning porn, and no incident response team. Security had pretty much become the responsibility of the various system administrators. The Unix servers that I administered were physically secure and each had the following warning banner:

```
*******************************************************************************************

                  This system is NOT FOR PUBLIC USE.
                Unauthorized access or use is subject to
             DISCIPLINE, CRIMINAL AND/OR CIVIL SANCTIONS.
                   ALL USERS CONSENT TO MONITORING.

*******************************************************************************************
```

Although there were several other knowledgeable Unix administrators with whom I worked, I had sole responsibility for the servers I administered. The closest thing we had to a person in charge of overall security was the router engineer who handled the Internet connection and most of the frame relay connections.

Due to understaffing, there was also very little monitoring of the logs. This was a contributing factor to the incident being discussed. In short, we were grossly unprepared. Instead of being proactive, we merely reacted to situations as they arose. Other than checking several queues and logs 2 or 3 times a day, the only preparation I faithfully performed was to back up all of my systems as needed. And, although the backup tapes did not figure in the resolution of this incident, they saved me many times.

III. Identification

It was my normal routine to check the mail queue upon arriving at work in the morning. This usually occurred between 6:15 and 6:30 AM. On the morning in question, I noticed an unusually large number of messages in the mail queue (over 3500). My first thought was that the sendmail program had died for some reason and I issued a 'ps –ef |grep sendmail' command to verify that sendmail was still running. After satisfying myself that it was, I typed tail –f /var/adm/syslog.dated/14-Oct-97:50/mail.log and was surprised to find that, not only was sendmail running, but my machine was relaying dozens of messages a minute, most of which were addressed to AOL accounts. I quickly determined that this had been going on for over 5 hours.

Upon checking the log further, I discovered that, although the domain remained the same, after every 5 or 6 messages, the sender's ID was changing. This was important because it prevented me from easily blocking mail. My next step was to change directories to /var/spool/mqueue and inspect both the body and the header of several messages. The body was identical in all of the messages that I checked. Typing 'ls –l' quickly confirmed that all of the messages in question were identical in size.

At this point it was obvious that my mail relay server had been hijacked and was being used to spam AOL accounts. And to make things worse, there was little I could do about it at the moment because we were also legitimately relaying mail to 35 or 40 domains belonging to our frame relay customers. And, since the spam mail was coming into my system from a large well known tier 1 ISP, I could not filter out the all of the mail being sent from that ISP at the router. Remember, the sender's ID was changing every 5 or 6 messages and I was receiving thousands of them. There seemed to be little I could do at the moment other than keep logs and ride it out. There was no incident handle team to notify and no one to whom I could assign the incident. It was my system and I would have to be the incident handler. Other than informing my Supervisor that an incident was taking place, there was no escalation.

IV. Containment

My highest priority at this point was stopping the inbound flow of spam mail without impacting legitimate email. Obviously, this was easier said than done. I decided that the one thing I could do was notify the ISP. Upon checking their web site, I found both an email address and a telephone number for reporting system abuses. When I called the number, identified myself, and explained the problem, I was told to send proof in the way of logs to the abuse email address. I quickly copied several hours of the mail log into a separate file and attached it to a brief explanation of my problem and sent it to the ISP.

Within 30 minutes, I received a phone call from the analyst with whom I had spoken earlier and was informed that they had determined the true identity of the offending account. Luckily for me, the account belonged to one of their customers and they had terminated it. A quick check of my logs confirmed that I was no longer receiving messages from the account in question.

I had stopped the flow of spam mail but I still had the immediate problem of what to do with the almost 4000 messages still queued for AOL. Initially, I didn't see any easy way of purging the unwanted mail from the queue and was tempted to simply send the messages and let the mail queue clear that way. I quickly decided not to. Instead of letting the mail queue clean itself out, I killed sendmail and performed a level 0 dump of the entire mail server. This caused an interruption in the flow of inbound mail but I knew that the server outside of the firewall would queue the messages until the internal server came back on line. Outbound mail was not effected, except for replies to messages, because there were several shorter paths we used to send email out of the corporation.

I also still had the long range problem of identifying and fixing the vulnerability that had permitted the mail server to be hijacked.

V. Eradication

Next I enlisted the help of a fellow system administrator who was very good at writing shell scripts.

Since the message ID was contained in the data that was returned by the mailq command, we were able to pipe the output of mailq to a file, parse out the message IDs, and pipe them to a new file.

The new file was then used as input to a small script that looped through and deleted the appropriate files from /var/spool/mailq. Once all of the spam files were deleted, I restarted sendmail and began to search for a permanent solution to the problem. The outage had lasted for less than an hour. Because it was still early in the morning, no one had seemed to notice that the server was down.

Luckily, this was all I had to do in the way of eradication. No malicious agent had been planted on my server. Instead, I had been hijacked and used as a tool for delivering spam. Embarrassing to be sure, but otherwise my system had not been harmed.

Before long I also knew the cause of the incident. I was running a very old version of sendmail (sendmail 5.65) and was acting as an open relay. I had no way of preventing future mail server hijackings. There simply wasn't a way to configure this version of sendmail to block unwanted relaying. I would have to upgrade.

VI. Recovery

Level 0 dumps of root and var were done each time the OS was patched or upgraded. The system was used strictly as a mail gateway and relay, and had just 2 accounts, root and wrb, an administrative account I had set up for myself. Since the system had not been broken into, I decided that it was pointless to restore it from tapes. I did change the password on both accounts just to be safe. My immediate problem was to prevent the server from being hijacked again.

I had been able to discover rather quickly that the problem was due to an old version of sendmail that shipped as part of Digital Unix. (Actually OSF/1 when the machine was purchased.) I also found out that it was not possible to configure the version of sendmail I was using, sendmail 5.65, to block unwanted relays to third party domains.

Upon going to www.sendmail.org, I was shocked to discover how many versions of sendmail I was behind. It was not possible to take the mail relay out of service for the amount of time I would need to install sendmail 8.9.1. So I developed an alternate plan.

I discovered that the Sun Ultra5 workstations running Solaris 2.6 that we used as DNS servers had sendmail 8.8.8 installed and that this version of sendmail, while not the latest and greatest, could be configured so that it relayed only to selected domains. Although sendmail was not currently running on any of the servers in question, it would be easy to activate and configure it on one of them. After making tape backups, I took one of the internal DNS servers out of service and, using sys-unconfig, gave it the name and IP address of the mail relay. This was only temporary. As soon as I rebuilt the DEC server, I intended to put the Sun back in service as a DNS server.

I had worked with Solaris sendmail before so it was fairly simple to configure the Sun to mimic the DEC. I replaced the DEC with the Ultra5 early in the afternoon and monitored it closely. Once I was convinced that mail relaying was working normally, I download the latest version of sendmail from www.sendmail.org began the job of upgrading the DEC.

Sendmail 8.9.1 by default blocked all relaying to third party domains. As part of my configuration, I included a file that listed the domains for which I would relay mail. Everything else would be blocked. Here is a log entry showing a blocked relay:

```
Aug 10 01:49:06 ftp sendmail[19842]: BAA19842: ruleset=check_rcpt, arg1=<smith.
j.a@nert.xxx.yyy.com>, relay=dialup-209.244.234.228.Weehawken1.Level3.net [209.2
44.234.228], reject=550 <smith.j.a@nert.xxx.yyy.com>... Relaying denied
```

After sendmail 8.9.1 was compiled and configured on the machine that had been used to relay the spam email, I performed another level 0 dump of the entire server and temporarily stopped sendmail on the external mail server to let the mail queue on the internal machine empty. After the last message was sent, I downed the Sun and brought the DEC back up in its place. I then restarted sendmail on the external mail server and observed the flow of mail through the inside machine until I was satisfied that mail was flowing unobstructed to our frame relay customers. I also performed some tests of my own by trying to relay mail to third party domains through the restored server. In all cases, the server performed as desired.

VII. Follow-up

The primary lesson I learned from this incident was to keep all applications current. In retrospect, I was lucky that the server had only been hijacked and not broken into.

There were many vulnerabilities associated with the version of sendmail that was running on the hijacked server. During the next couple of weeks I used the same procedure to upgrade sendmail on all of our mail servers. I also took the time to upgrade BIND and to upgrade all of the Unix machines to the most current version and patch level of Digital Unix or Solaris. As each machine was upgraded and patched, I did a level 0 dump of all file systems. I began to search for vulnerabilities and patches on a weekly basis.

Shortly after this incident, another system administrator was hired. Between the two of us we were able to more effectively develop a security plan. In effect, we became the incident response team.

None of the mail servers were ever hijacked again. My routine check of the logs found many messages where relaying had be denied so there is no doubt in my mind that we would have been hijacked repeated had sendmail not been upgraded.

Since I was the only person involved in this incident, there was no meeting or report, other than a brief verbal explanation to my manager about what had happened and how I had fixed it.

VIII. For at least one operating system involved in the incident, show the process used to assess and contain, including screen shots and operating system commands. In this section you should describe your jump kit, or all the tools that you used.

As described previously, only one server, running Digital Unix 3.2c was involved in the incident. No special tools were used to assess and contain the incident. Initially, I used simple Unix commands such as tail, more, and grep to view the logs. The contents of the mail queue was obtained by issuing the mqueue command. Once the ISP stopped the flow of spam to my server, I was again able use the 'sendmail stop' command to kill the sendmail daemon while I deleted the unwanted messages from the mail directory. All backups were done using dump. Nothing was restored from tape so that command was not needed.

IX. For at least one operating system involved in the incident, describe in detail the process used to back up the system. This write-up should include descriptions of the hardware, commands, and any problems that you ran into.

Although I also made a complete backup of the DNS server (Solaris 2.6) that temporarily was used as the mail relay, the only operating system that was truly involved in the incident was Digital Unix. As I mentioned before, the machine was a DEC AlphaServer 1000 running Digital Unix 3.2c. Attached to the server was a Digital tape drive that used DDS2 tapes.

A full (level 0) dump was taken from the AlphaServer twice during the incident. The initial dump was completed just after the machine was taken off line. The second dump was done after sendmail was upgraded but before the machine was placed back into service. Root (/) and var (/var) were the only file systems on the machine and each was dumped to its own tape. The command syntax used to create the tapes was:

```
dump –0u –f  /dev/rmt0h /var
```

This command states that a level 0 dump is to be made of /var to tape device /dev/rmt0h and that /etc/dumpdates should be updated. A similar command was used to back up the root filesystem.

Doing a level 0 dump was my standard method of making backups so I was very familiar with the command and did not experience any problems. Neither tape was used to restore the system.

X. Describe in detail the chain of custody procedures used, any affirmations, and a listing of all evidence.

The chain of custody in this incident was rather simple. Except for the portion of the log file that I sent to the ISP, custody remained with me at all times. I personally performed all of the system backups and, upon completion, placed all of the tapes in the same locked cabinet as usual. I had the only key to the cabinet.

The only real evidence was the log files and the mail remaining in the queue when I completed the initial level 0 dump. My supervisor had spoken to the ISP to confirm that they had terminated the spammer's account and decided that no further action would be taken. The evidence remained in the tape storage locker for about six months. Eventually, I reused the tapes and the data was overwritten as part of a normal backup. In effect, this officially ended this incident forever.

William Bergman