Forensic Analysis Without an IDS: A Detailed Account of Blind Incident Response

Eric “Loki” Hines
Email: loki@fatelabs.com
Fri Jan 4 19:02:36 EST 2002

Overview

This paper documents the compromise and forensic analysis steps taken to ascertain the motives, attack, and tools used by a Blackhat in the compromise of a military web server.

This paper hopes to introduce the Internet community to a new breed of sophisticated hacker, a neoteric menace that is far beyond her Skript Kiddie foe, bringing to light the strong line between the Blackhat and Skript Kiddie. This paper documents the systematic compromise of over 20 computers and how the attack was so well laid out, planned, and executed that it went unnoticed for 3 months.

Fate Labs had the opportunity to work with the NIPC and FBI regarding the further investigation into identifying the alleged hacker. So this document will also provide details on how the hacker's ISP was tracked down, identified, and eventually contacted.

The Knock at the Door

It was the morning of December 18th that we received notice that the machine was possibly compromised as a bot appropriately labeled ^domain^ had joined a private EFNET channel matching ^domain^.hostname.domain.com to our organization. To ensure that the email we received was not in fact a hoax I decided to do an nmap portscan of the box remembering to grab a list of open high ports with the following syntax.

```
[root@pa-lnx01 /]# nmap -sS -O -vv www.domain.com -p1-65535

Starting nmap V. 2.54BETA22 ( www.insecure.org/nmap/ )
Host www.domain.com (192.168.0.1) appears to be up ... good.
Initiating SYN Stealth Scan against www.domain.com (192.168.0.1)
Adding TCP port 1198 (state open).
Adding TCP port 1028 (state open).
Adding TCP port 51000 (state open).
Adding TCP port 21 (state open).
Adding TCP port 50000 (state open).
Adding TCP port 54000 (state open).
Adding TCP port 139 (state open).
Adding TCP port 80 (state open).
Adding TCP port 443 (state open).
Adding TCP port 135 (state open).
Adding TCP port 1034 (state open).
Adding TCP port 1031 (state open).
The SYN Stealth Scan took 1177 seconds to scan 65535 ports.
For OSScan assuming that port 21 is open and port 1 is closed and neither are fi
rewalled
```

```
Interesting ports on www.domain.com (192.168.0.1):

(The 65523 ports scanned but not shown below are in state: closed)
Port       State       Service
21/tcp     open        ftp
80/tcp     open        http
135/tcp    open        loc-srv
139/tcp    open        netbios-ssn
443/tcp    open        https
1028/tcp   open        unknown
1031/tcp   open        iad2
1034/tcp   open        unknown
1198/tcp   open        unknown
50000/tcp  open        unknown
51000/tcp  open        unknown
54000/tcp  open        unknown

Remote operating system guess: Windows NT4 / Win95 / Win98
OS Fingerprint:
TSeq(Class=TD%gcd=1%SI=5%IPID=BI%TS=U)
T1(Resp=Y%DF=Y%W=2017%ACK=S++%Flags=AS%Ops=M)
T2(Resp=Y%DF=N%W=0%ACK=S%Flags=AR%Ops=)
T3(Resp=Y%DF=Y%W=2017%ACK=S++%Flags=AS%Ops=M)
T4(Resp=Y%DF=N%W=0%ACK=O%Flags=R%Ops=)
T5(Resp=Y%DF=N%W=0%ACK=S++%Flags=AR%Ops=)
T6(Resp=Y%DF=N%W=0%ACK=O%Flags=R%Ops=)
T7(Resp=Y%DF=N%W=0%ACK=S++%Flags=AR%Ops=)
PU(Resp=N)

TCP Sequence Prediction: Class=trivial time dependency
                         Difficulty=5 (Trivial joke)
TCP ISN Seq. Numbers: 446F2D98 446F2DA2 446F2DA8 446F2DB1 446F2DC4 446F2DD5
IPID Sequence Generation: Broken little-endian incremental

Nmap run completed -- 1 IP address (1 host up) scanned in 1178 seconds
```

Forensic Analysis Procedures

After I ran my nmap scan the ports 50000, 51000, and 54000 looked too suspicious to leave alone. So, to get an idea of exactly what might be bound to those ports the only resource I had was to telnet to each port that looked suspicious and see if I could get any telnet banners. Amazingly enough, this is what I got.

```
[root@pa-lnx01 root]# telnet www.domain.com 50000
Trying 192.168.0.1...
Connected to www.domain.com.
Escape character is '^]'.


^domain^  (Eggdrop v1.6.8 (C) 1997 Robey Pointer (C) 2002 Eggheads)

Please enter your nickname.
Sorry, that nickname format is invalid.
Connection closed by foreign host.


[root@pa-lnx01 root]# telnet www.domain.com 54000
Trying 194.235.171.133...
Connected to www.domain.com (192.168.0.1).
Escape character is '^]'.
220 Serv-U FTP-Server v2.5k for WinSock ready
```

At this point the only real course of action was to try and ascertain how the hacker might have got in, whether or not it was a w0rm or in fact it was individually picked out. The first step was to identify the web server software running on the remote machine.

We will simply telnet to the machine and issue the GET HTTP /1.1 command and see what our response is.

```
pa-obsd01# telnet www.domain.com 80
Trying 136.142.42.14...
Connected to www.domain.com.
Escape character is '^]'.

GET HTTP/1.1

HTTP/1.1 400 Bad Request
Server: Microsoft-IIS/5.0
Date: Mon, 21 Jan 2002 19:34:52 GMT
Content-Type: text/html
Content-Length: 87
```

Now that we have identified the machine as being an IIS/5 machine, let's try the first chapter of the bible to compromising Windows WWW servers.

Chapter 1: Trivial Compromises of Windows Machines with Unicode:

At this point I went ahead and just utilized a script I own that contains over 250 different Unicode strings.

```
pa-obsd01# ./iis1
[root@pa-lnx01 exploits]# ./iis1

 Hackweiser global domination y2k+1
 This is 100% USDA Non-Approved Beef...
  Those Bastards... 263 checks. -Hackah Jak
 ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~

 File Name: string-scan30b.pl
 Version: 3.0b, Date: Friday May 18th 2001
  *Look at source for more info.

 Few Side Notes:
 URL can be changed at anytime by typing URL.
 The Webserver can be re-SCANed at anytime by typing SCAN.
 Program can be QUIT at anytime by typing QUIT.
 Also an easy way to backdoor is to type: copy c:\winnt\system32\cmd.exe cmd1.exe
  Then goto www.server.com/vulndir/cmd1.exe?/c+dir
 HELP prints this ...
 ENJOY !

Host : www.domain.com
Port : 80

Command :dir
HTTP://www.domain.com/msadc/root.exe?/c+dir
OUTPUT FROM www.domain.com

Server: Microsoft-IIS/4.0
Date: Mon, 21 Jan 2002 19:52:27 GMT
Content-Type: application/octet-stream
Volume in drive C has no label.
Volume Serial Number is 9637-F8AB

 Directory of C:\Program Files\Common Files\system\msadc

08/06/99  12:19p       <DIR>          .
08/06/99  12:19p       <DIR>          ..
10/02/97  09:28a              19,388 readme.txt
01/06/02  06:13a                   0 TFTP1143
01/06/02  06:24a                   0 TFTP842
01/06/02  07:01a                   0 TFTP1000
```

Well, it should be a bit obvious to us at this point that the attacker broke into the machine with a trivial Unicode attack. From the TFTP logs found in that directory as well as cmd.exe renamed and moved to one of the webroot subdirs, it should also be apparent that the individual also used TFTP to send/receive files back and forth from the machine. We will find those files a bit later in this document.

Now that we have ascertained exactly how the machine was compromised, we will now move forward with going on-site to the machine to recover both an fport listing and possible IIS logs for Unicode GET requests, presupposing that the Hacker did not delete the log directory in Windows.

First thing is first… A simple nmap scan will not tell us exactly what is bound to what ports. Well, it will.. sort of, but we want more information than just telneting to the individual ports. It's not merely enough to know just the open ports on a system, but the programs bound to each port. This is accomplished by using the tool fport.exe by Foundstone (http://www.foundstone.com)

Below is an output of the programs that were specifically bound to individual open ports on the machine. This is a great way to find where on the system Trojans have been hidden. See below.

```
FPort v1.33 - TCP/IP Process to Port Mapper
Copyright 2000 by Foundstone, Inc.
http://www.foundstone.com

Pid   Process          Port  Proto Path
2     System       -> 21    TCP
2     System       -> 80    TCP
2     System       -> 135   TCP
71    RpcSs        -> 135   TCP   C:\WINNT\system32\RpcSs.exe
2     System       -> 139   TCP
2     System       -> 443   TCP
71    RpcSs        -> 1025  TCP   C:\WINNT\system32\RpcSs.exe
2     System       -> 1025  TCP
71    RpcSs        -> 1026  TCP   C:\WINNT\system32\RpcSs.exe
2     System       -> 1026  TCP
2     System       -> 1027  TCP
56    msdtc        -> 1027  TCP   C:\WINNT\System32\msdtc.exe
2     System       -> 1028  TCP
56    msdtc        -> 1028  TCP   C:\WINNT\System32\msdtc.exe
2     System       -> 1029  TCP
56    msdtc        -> 1029  TCP   C:\WINNT\System32\msdtc.exe
2     System       -> 1030  TCP
130   MSTask       -> 1030  TCP   C:\WINNT\system32\MSTask.exe
130   MSTask       -> 1031  TCP   C:\WINNT\system32\MSTask.exe
2     System       -> 1031  TCP
130   MSTask       -> 1032  TCP   C:\WINNT\system32\MSTask.exe
2     System       -> 1032  TCP
2     System       -> 1033  TCP
2     System       -> 1034  TCP
2     System       -> 1184  TCP
2     System       -> 1197  TCP
183   alertsvc     -> 1197  TCP   C:\PROGRA~1\Navnt\alertsvc.exe
2     System       -> 1198  TCP
183   alertsvc     -> 1198  TCP   C:\PROGRA~1\Navnt\alertsvc.exe
2     System       -> 1211  TCP
183   alertsvc     -> 1211  TCP   C:\PROGRA~1\Navnt\alertsvc.exe
2     System       -> 3013  TCP
148   spsvc        -> 3013  TCP   C:\WINNT\system32\spsvc.exe
2     System       -> 4118  TCP
148   spsvc        -> 4118  TCP   C:\WINNT\system32\spsvc.exe
2     System       -> 50000 TCP
148   spsvc        -> 50000 TCP   C:\WINNT\system32\spsvc.exe
2     System       -> 51000 TCP
148   spsvc        -> 51000 TCP   C:\WINNT\system32\spsvc.exe
```

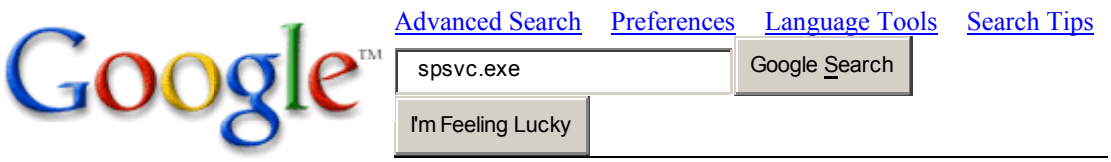
```
2    System       ->  54000 TCP
152  spsvc        ->  54000 TCP   C:\WINNT\system32\inetsrv\iisadmpwd\spsvc.exe

71   RpcSs        ->  135   UDP   C:\WINNT\system32\RpcSs.exe
2    System       ->  135   UDP
2    System       ->  137   UDP
2    System       ->  138   UDP
2    System       ->  1035  UDP
148  spsvc        ->  1035  UDP   C:\WINNT\system32\spsvc.exe
```

The output above at first glance would not show anything that might look suspicious if quickly scanning the filenames for "malicious" looking names. (Trojan.exe, cmd.exe, command.com, nc.exe, etc). This is where things get really exciting.

The immediate thing I noticed was the multiple instances of spsvc.exe. This sounds like something that might ship with Windows but wasn't too quick to believe that it would bind itself to so many high number ports. First thing was first, checking to see if there was any native help for the command, if it was in fact a built-in Windows binary.

```
C:\WinNT\System32>spsvc.exe /?
'spsvc.exe' is not recognized as an internal or external command,
operable program or batch file.
```

Something I like to say a lot is that "the answer to life's questions is on google.com." ☺ I visit google.com on probably every single incident response case I investigate. What I did was I wanted to see if there was ANYTHING in the Google.com databases on this filename. I did a simple search for spsvc.exe hoping to find at least ONE hit matching that file if it was an actual program or something that came with Windows.



Did you mean: ***spsc***.exe

Your search - **spsvc.exe** - did not match any documents.

Suggestions:

- Make sure all words are spelled correctly.
- Try different keywords.
- Try more general keywords.

At this point we've pretty much realized that it is most likely not a program that ships with Windows, nor is it a third party application. I guess we've already found this out by telneting to the host and seeing those Eggdrop telnet banners, but if you haven't tried that yet, are sitting at localhost and just want to see what Trojans might be listening on particular ports that don't answer to telnet requests, than this was a great exercise for you.

All to often the details provided in the TASK MGR in Windows just doesn't suffice for wanting to know exactly what those programs are, what the executables are named, or if you want something more than just active ports provided by 'netstat –an'. This is where your fport.exe file comes into play.

Taking a closer look at the fport listing, you quickly realize that the spsvc.exe file is conveniently concealed in C:\WINNT\system32\inetsrv\iisadmpwd\spsvc.exe and C:\WINNT\system32

We can pretty much simply download this spsvc.exe file to our Linux machine and run a 'strings' on the binary to find any human readable text located inside the binary file. This will help us to quickly identify key identifiers as to what the file is or does. Again, this information was provided by telneting to these ports, but this exercise is to merely point out different avenues of incident response capabilities.

```
ade:g:G:hi:lLno:p:rs:tuvw:z
wrong
Cmd line:
port numbers can be individual or ranges: m-n [inclusive]
        -u              UDP mode
        -v              verbose [use twice to be more verbose]
        -w secs         timeout for connects and final net reads
        -z              zero-I/O mode [used for scanning]
        -t              answer TELNET negotiation
        -g gateway      source-routing hop point[s], up to 8
        -G num          source-routing pointer: 4, 8, 12, ...
        -h              this cruft
        -i secs         delay interval for lines sent, ports scanned
        -l              listen mode, for inbound connects
        -L              listen harder, re-listen on socket close
        -n              numeric-only IP addresses, no DNS
        -o file         hex dump of traffic
        -p port         local port number
        -r              randomize local and remote ports
        -s addr         local source address
        -e prog         inbound program to exec [dangerous!!]
        -d              detach from console, stealth mode
[v1.10 NT]
connect to somewhere:   nc [-options] hostname port[s] [ports] ...
listen for inbound:     nc -l -p port [options] [hostname] [port]
options:
GetNumberOfConsoleInputEvents
CreateFileA
SetEndOfFile
LCMapStringA
LCMapStringW
own -l -p 99 -t -e cmd.exe
```

This looks to be the modified Netcat by eEye Digital Security (ncx99.exe)
This is a hacked netcat-based trojan used to exploit the eEye NT4+IIS4 URL remote buffer overflow (for use on port 99).

Jumping over to the above directories I quickly noticed several files that well, pretty much should not have been there. It was at this point that the home directory of where the hacker was storing his files started to come into light.

Located in C:\WINNT\System32 was not only the spsvc.exe file but several files containing the keyword *bot*. Remember, although the hacker can easily rename the Win-Eggdrop to a new filename, it'd be pretty ridiculous to try and attempt to decompile than recompile the Eggie to call a different configuration file. Therefore we know that the hacker must keep the default Eggdrop config filename the same or the Eggdrop will not be able to run as it wouldn't be able to find its own configuration file.

Familiar with running Eggdrop bots in the past, I knew that the Eggdrop calls its username file, *my.user*. This file was found in the same directory along with low and behold, eggdrop.conf

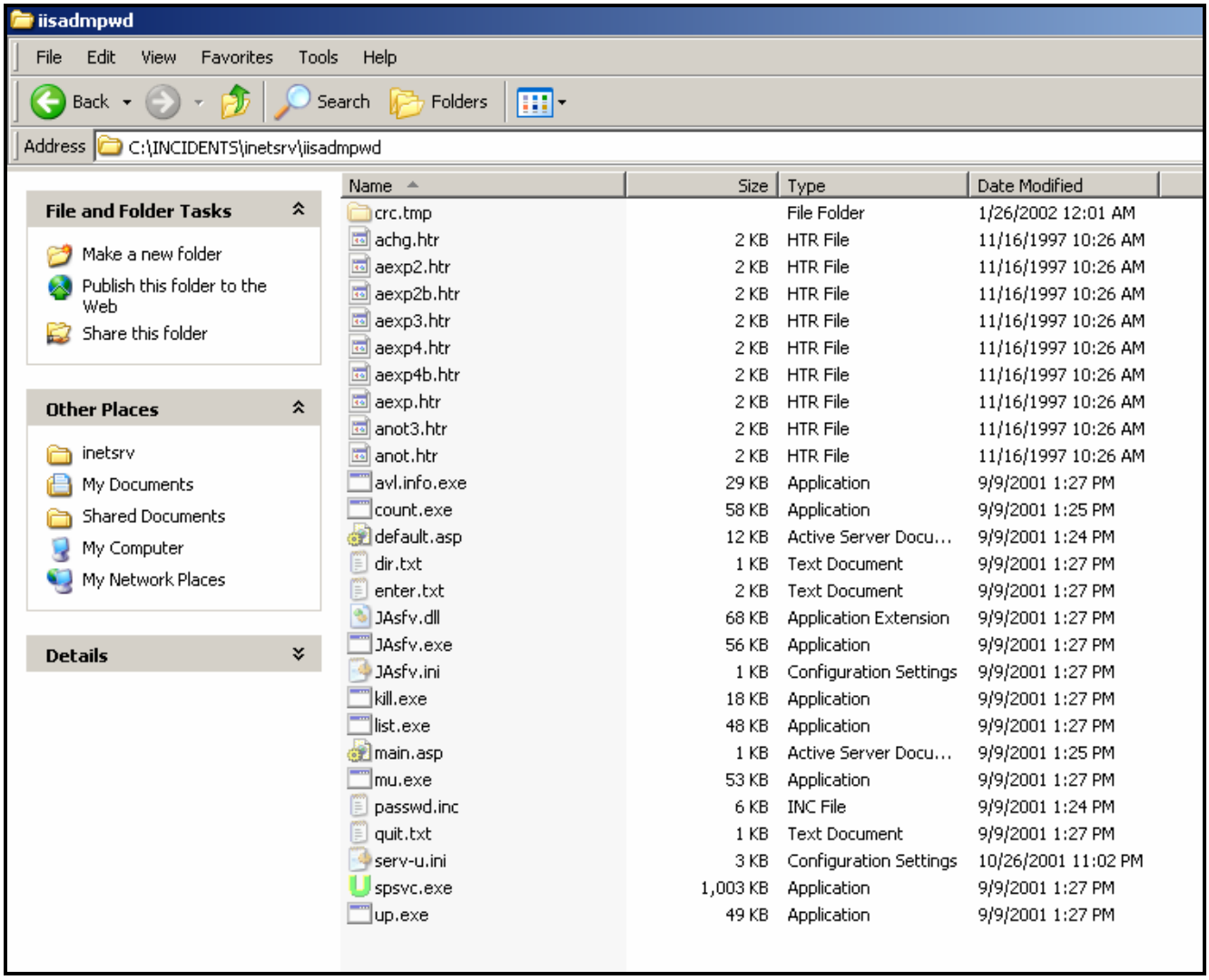
One thing for those researchers out there who like tracking their culprits down on IRC, if you see a configuration file for an Eggdrop installed on your machine, open it up and look for the following line so you can /whois the nickname on an IRC server such as EFNet. Also, you never know… The hacker might even be stupid enough to put in his real email address and handle.

```
# who's running this bot?
set admin "lumepume <email: not@net.no>"
```

Also, check out *my.user* you might find some interesting things in this file ;). The file is used by the Eggdrop to record the userID and host of every machine/user that authenticates with the Eggdrop.

```
lumepuma    - fhjmnoptx
--HOSTS *!raped@*.arcor-ip.net
--HOSTS *ped@*.dip.t-dialin.net
--HOSTS *!raped@*.dip.t-dialin.net
--HOSTS *!lum3@*.waterbong.co.uk
--HOSTS *!BuG@*.dip.t-dialin.net
--HOSTS *!lum3@62.146.208.191
--HOSTS *!BuG@*.daf.kun.nl
--HOSTS *!avl@*.strath.ac.uk
--HOSTS *!*inistra@*.t-dialin.net
--HOSTS *!raped@*.dip0.t-ipconnect.de
--XTRA auth 0
--XTRA created 997684181
--XTRA permident *!*@*
--XTRA authnick lum3
--XTRA authhost pD4B9ED10.dip.t-dialin.net
--LASTON 1009997650 @^hub^
--PASS +zsAy6/sq9uX/
```

Located somewhere in this file might also be a connect from your hacker's real dialin. You never know how lazy your kid might be. I went ahead and checked both *dialin* domains and found that one fo them was actually a dialup ISP located in Germany.



If you can see closely enough you will notice that this directory is in fact full of several files that definitely should not be on this computer, legitimately. Also, look at the spsvc.exe binary and the ICON associated to it. This is the infamous green-U of the SERV-U FTP Daemon. This is an immediate indicator that the machine had SERV-U installed on it to transfer files to/from the machine.

There are several other files there that I want all of you to pay particularly close attention to.

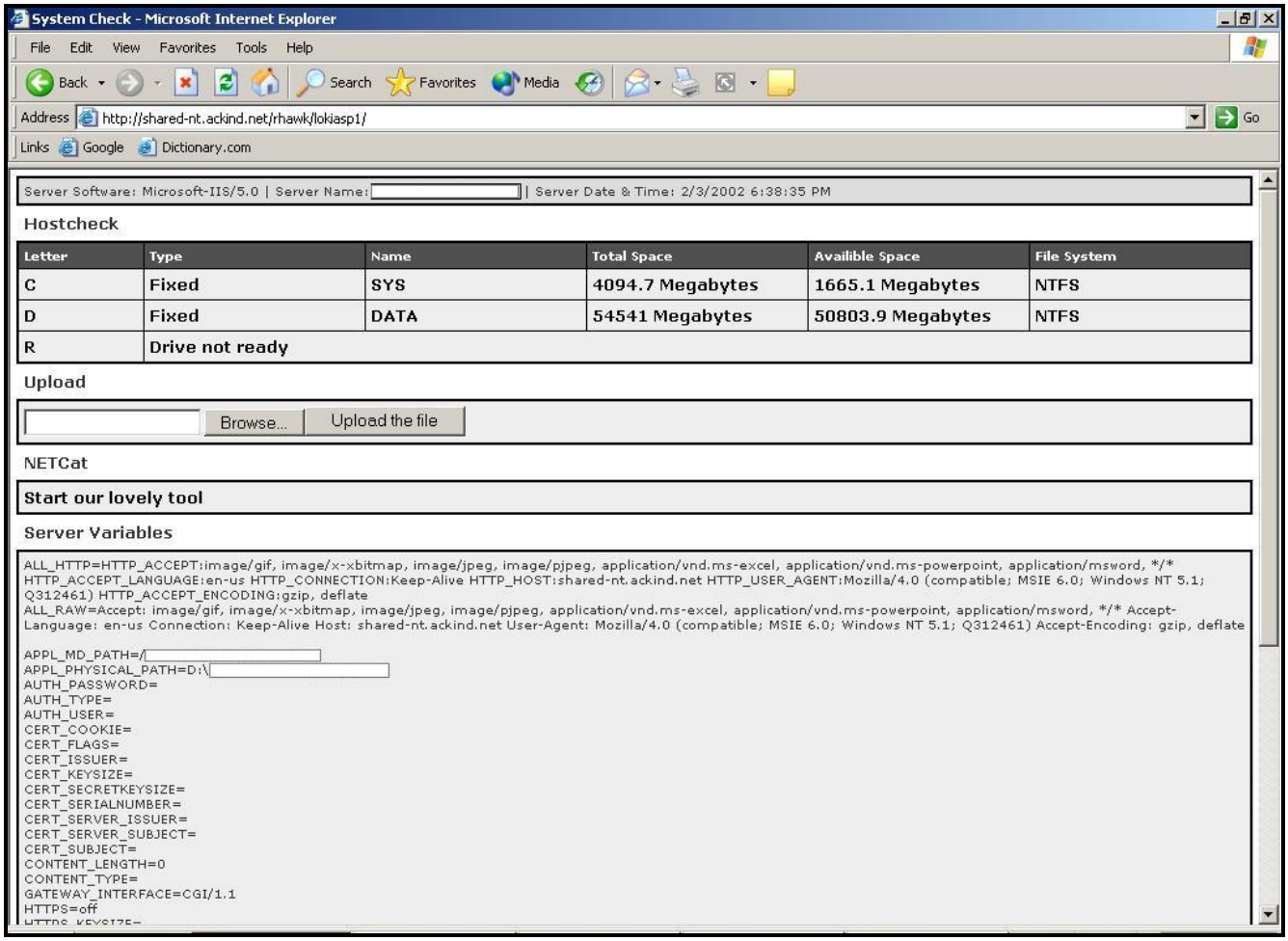
1. kill.exe
2. list.exe
3. main.asp
4. default.asp
5. passwd.inc

These (5) five files are definitely bad news.

1. kill.exe allows you to kill a running PID from the command prompt (perfect if you don't have a GUI to CNTRL-ALT-DEL forcekill running processes from within Windows)
2. list.exe allows the user to list all processes in a Windows shell. This is obviously needed to utilize the kill.exe program. Hackers will typically use this to find out what processes are running on a machine (IIS/etc) kill them as needed should they require specific use of an active port. Who knows really, I guess you could pretty much have a great deal of reasons for using these 2 utilities.
3. main.asp and default.asp both raised flags with me. Remember, we still don't know that many details on the compromise or all the tools that were used.

This is where much of this write-up will hopefully start to get interesting or possibly show you some new ways in which hackers are automating or making their lives much easier in the compromise of a remote machine. If you remember back earlier in this document, you will remember we found a bunch of tftp log files in the IIS directory. TFTP, or Trivial FTP is a simple way of sending and receiving files to or from the machine. What all of you need to remember is that when you have a Unicode shell to the machine through port 80, it isn't a fully Interactive shell. Meaning, if you type, lets say ftp.. you won't get the full response back.. actually it will just lock your connection to the machine and will force you to start another Unicode attack against the host. So in order to get around this, a hacker will want to bind cmd.exe (nt/2k/xp) or command.com (win95/98/me) to a port to connect to. Telneting to the port once cmd.exe is bound to it will obviously do one thing, drop you into a dos shell on the box.

Lets go ahead and take a closer look at default.asp and main.asp. Both are web pages residing outside of the main wwwroot directory in a hidden directory created by the hacker. We'll go ahead and assume it was the hacker who uploaded these asp scripts to the machine. Check out this lovely asp script below. I will be the first to admit, I've never seen anything this beautiful in my many years of incident response.



At this point we will go ahead and take a look at the IIS logs we retrieved from the C:\WINNT\System32\LogFiles directory. We'll want to check for cmd.exe access as we will assume the machine was compromised with the IIS/Unicode attack.

Let's take a look at a few of those logs now.

```
13:33:18 192.168.0.5 GET /c/winnt/system32/cmd.exe 404
13:33:18 192.168.0.5 GET /d/winnt/system32/cmd.exe 404
13:33:18 192.168.0.5 GET
/_mem_bin/..%5c../..%5c../..%5c../winnt/system32/cmd.exe 404
```

Above you will see how IIS stores logs of all GET strings to the aforementioned System32 subdirectory. In this directory we basically just searched the folder for files containing cmd.exe. This was our first approach at actually identifying the hacker and where he came in from. Typically I find in forensic analysis incidents the hacker was too lazy to go through a web anonymizer and ran Unicode strings directly from his Internet Explorer or Netscape browser from his computer. This gave us the real IP address of the attacker who than came in through a netcat bound port on 99. Why the hacker did not just run a Unicode scanner from a hacked shell is beyond me, laziness maybe?

After identifying the Eggdrop bot as being connected to an EFNet IRC server I than compared the IP address of the IIS logs to the alleged admin of the botnet I found in the IRC channel. Both addresses matched perfectly. Lets go ahead and meet our culprit.

[lum3 has address spl33n@busfac32.busfac.calpoly.edu]
Jan 04 00:12:50 <lum3> i did a few pieces of
Jan 04 00:13:01 <Loki--> you are a good coder
Jan 04 00:13:34 <lum3> not realy
Jan 04 00:13:38 <lum3> i learn it atm
Jan 04 00:14:28 <Loki--> lum, i saw a crt.32 directory, what did you create that for?
Jan 04 00:14:58 <lum3> thats for crc checks of files
Jan 04 00:15:01 <lum3> if i need it

<lum3> http://www.tellabs.dk/unicode.pdf
<lum3> get thatone
<lum3> cause i don´t want any maschine i could get
<lum3> i could have ~15000 eggdrops in here
<lum3> but for what?
<lum3> i take thoose with nice hostnames hehe
<lum3> protect them for unicode
<lum3> install my eggdrop and serv-u and thats all
<Loki--> what were you using serv-u for?
<lum3> then they loose ~10-15megs of ram and ~150mb traffic a month
<lum3> for uploading files to the eggdrop
<lum3> updating the eggdrop
<lum3> nothing more

<lum3> uploading files means i add tcls and such stuff
<lum3> i never looked for
<lum3> seen some sensitie things on *.dfn.de
<lum3> but thats too hot then
<lum3> so i leave that servers
<Loki--> too hot?
<Loki--> :)
<Loki--> lum3= hot?
<lum3> sek
<lum3> Subnet mask campus Domain Name
<lum3>
<lum3> 129.171.0-31.0 255.255.224.0 RSMAS miami.edu
<lum3> 129.171.32-63.0 255.255.224.0 Gables miami.edu
<lum3> 129.171.64-95.0 255.255.224.0 Medical miami.edu
<lum3> 129.171.96-127.0 255.255.224.0 RSMAS miami.edu
<lum3> 129.171.128-159.0 255.255.224.0 Medical miami.edu
<lum3> 129.171.160-191.0 255.255.224.0 Gables miami.edu
<lum3> 129.171.192-223.0 255.255.224.0 Gables miami.edu
<lum3>
<lum3> 192.239.208.0 255.255.255.0 RSMAS(UMIA) miami.edu
<lum3> 192.111.123.0 255.255.255.0 RSMAS(AOML) aoml.erl.gov
<lum3> 199.242.231.0 255.255.255.0 RSMAS(Fisheries) sefsc.noaa.gov
<lum3> 199.242.232.0 255.255.255.0 RSMAS(Fisheries) sefsc.noaa.gov
<lum3> 199.242.233.0 255.255.255.0 RSMAS(Fisheries) sefsc.noaa.gov
<lum3> 199.4.250.0 255.255.255.0 RSMAS(South Pole) spole.gov
<lum3> 199.4.251.0 255.255.255.0 RSMAS(South Pole) spole.gov
<lum3> 204.89.132.0 255.255.255.0 RSMAS(South Pole) spole.gov
<lum3> 204.89.133.0 255.255.255.0 RSMAS(South Pole) spole.gov
<lum3> 204.145.215.0 255.255.255.0 RSMAS(South Pole-Palmer) nsf.gov
<lum3> 204.145.157.0 255.255.255.0 RSMAS(vBNS ONLY) NOTE:
<lum3> BELLSOUTH.NET SHOULD NOT ADVERTISE.
<lum3> 192.88.124.0 255.255.255.0 Gables(ECE) ece.miami.edu
<lum3> 192.70.171.0 255.255.255.0 Gables(MATH) cs.miami.edu
<lum3> 192.31.89.0 255.255.255.0 Gables(MATH) cs.miami.edu
<lum3>
<lum3> 204.68.64.0 255.255.224.0 Medical(UMMS) ummedical.edu
<lum3> NOTE: Multi-homed to UUNET and NASA Science Internet with BGP-4. AS Number for all networks
<lum3> above is 4511.
<lum3> Prepared by Buddhi Abeysekera, Sr. Network Engineer
<lum3> University of Miami, Coral Gables, FL
<lum3> Date: June 16, 1999
<lum3> that was my first server
<lum3> the pc of the network admin at miami.edu
<lum3> rofl
<lum3> thats the only thing i ever dloaded
<lum3> never found anything interessting aterwards
<lum3> only netbus, bo and other torjans on many nt4 servers

<lum3> too many kiddies online

The following table represents all known information about machines compromised by this Blackhat. All systems identified as being compromised were all University machines and several small companies.

1. Opening up the Windows NT Task Manager showed two tasks running spsvc.exe which was suspicious as it is not an NT service.
2. Opening up the NT Services Control Panel showed two apparently identical 'Printer Spooler' services running which also seemed strange.
3. The original \winnt\system32\spoolsvc.exe was apparently replaced on 9/8/01.
4. fport showed the TCP port 21000 (a Serv-U FTP server) belonged to a service executed out of file \winnt\system32\inetsrv\iisadmpwd\spsvc.exe
5. Besides a number of related files (serv-u-ini, dir.txt, enter.txt, quit.txt, kill.exe, list.exe, mu.exe, up.exe) in the directory \winnt\system32\inetsrv\iisadmpwd\ there was a subdirectory (crc.tmp)
6. fport showed TCP ports 50000 and 51000 (an Eggdrop IRC "bot) belonged to a service executed out of file \winnt\system32\spsvc.exe -- which had several related (or modified) files in \winnt\system32:

   cygwin1.dll - Cygnus Windows Unix library
   cygz.dll - Cygnus ??? library
   eggdrop.conf - config file
   my.user - other IRC bots
   my.user~bak - backup version of above
   op.chan - Channel operator commands
   resolv.conf - DNS resolver file
   tcl83.dll - TCL interpreter DLL
   tclpip83.dll - TCL interpreter DLL
   up_stats.file - timestamps and IRC server name (irc.colorado.edu in our case)

   as well as the related subdirectories:

   filesys - empty
   help - Eggdrop IRC bot help files
   modules - DLL files for the bot (incl. Blowfish crypto)
   netbots - TCL scripts to program (add functionality to) the bot
   scripts - More of the above.
   text - motd (IRC bot message of the day file)
   tmp - empty

7. Registry entries for spool16 and spool32 (the Printer Spooler subsystem service in the Services control panel) had been modified with new values (Firestarter) to start these fake printer spooler services).

Conclusion
Everything from hidden iisadmpwd directories to Trojans renamed to what might look to be common system files, you will see the tactics of Blackhats and even Skript Kiddies progress further and further as security engineers gain more and more intel about their enemy. As security technology advances, as Intrusion Detection Systems gain more and more artificial intelligence, the tools of hackers will also shift in this often large labyrinth of incident response and forensic analysis.

We saw this in a recent presentation at Defcon this last year where payload encrypters were presented to slide past Intrusion Detection Systems. How can an IDS inspect the payload of a packet when the payload is encrypted to prevent content matching for say, Unicode strings sent to web servers?

Just like the incident covered in this paper where the attacker actually put his Eggdrops and Servu FTP daemon in the Windows Service Mgr set to startup on boot, covertly naming it Printer Spool services, we too as Security Engineers need to develop with them. Start taking a closer looks at files that you think were installed by IIS. Start wondering why certain directories were given an ATTRIB +H to hide it from your dir listings. Ask yourself "Why there are 3 Printer Spool services running in my services control panel?" What you see here is an example of how Crackers are becoming more and more sophisticated and clean when they compromise systems. Gone are the days where cmd.exe is renamed to root.exe or exploits are left in / or C:\

Appendix
Fate Research Labs
http://www.fatelabs.com

Security Focus/Bugtraq Advisory on Unicode
http://www.securityfocus.com/bid/1806

Honeynet Project
http://project.honeynet.org