SANS/GIAC Certifications:
Track 3 – Intrusion Detection In-Depth

Practical Assignment
Version 3.3

Submitted May 28th, 2003

By Andrew J. Patrick
In pursuit of GCIA Certification
Abstract:

This paper is submitted for the GCIA Version 3.3 Practical.

It consists of three parts:

1) Customized IDS Signature Creation: Why you aren’t using your IDS to its full potential - One of the more under-utilized capabilities of most modern Intrusion Detection Systems (IDS) is the ability of the organization to “roll their own” customized network traffic patterns for the IDS to sniff for. This may not sound like much of an innovation to some of the more expert members of the Intrusion Detection Community, but after close examination of how IDS is implemented in several organizations, I have become convinced that this may be the single biggest “bang-for-buck” improvement available. By following the few simple and cost-effective steps I will outline in this paper, any reasonably competent IDS analyst should be able to greatly improve the value of intrusion intelligence that their IDS provides to their organization.

2) Three “10 step standard format” SANS/GIAC Intrusion Analysis detects:
   
   Detect #1 – Front Page Extensions & Anonymous FTP
   
   Detect #2 – A Visit from a Curious Friend
   
   Detect # 3 – “It’s Code Red! No, It’s Nimda! No, it’s, it’s….???”

3) “Analyze This!” assignment. the University is confronted with three major issues with regard to its Intrusion Detection efforts: 1) There is far too much “noise” being generated by the IDS system. The majority of the alerts being generated appear to due to benign traffic, mis-configured hosts or IDS rules, and some customized alerts that do not appear to serve any useful function. This has the deleterious effect of swamping both the IDS sensors and the Analyst’s bandwidth. 2) The use of peer-to-peer networks such as Gnutella, Morpheus, and Kazaa is obviously quite widespread at the University. Rampant uncontrolled use of P2P networks is becoming much more controversial and potentially costly, as the Recording Industry Association of America has recently begun mounting major legal challenged to educational institutions which turn a blind eye to the resulting inevitable theft of copyrighted material. This uncontrolled file-sharing also constitutes a serious abuse of the University’s Internet capacity, which is quite likely to be interfering with much more legitimate bandwidth demands. 3) There are indications that several University systems may have been compromised by worms and/or external attackers. These should be examined immediately for any signs of potential compromise by the University’s InfoSec personnel.
Assignment 1:
Customized IDS Signature Creation:
Why you aren’t using your IDS to its full potential

One of the more under-utilized capabilities of most modern Intrusion Detection Systems (IDS) is the ability of the organization to “roll their own” customized network traffic patterns for the IDS to sniff for. This may not sound like much of an innovation to some of the more expert members of the Intrusion Detection Community, but after close examination of how IDS is implemented in several organizations, I have become convinced that this may be the single biggest “bang-for-buck” improvement available. By following the few simple and cost-effective steps I will outline in this paper, any reasonably competent IDS analyst should be able to greatly improve the value of intrusion intelligence that their IDS provides to their organization.

For motivation, I would first like to consider the reasons we would want to customize our own IDS in the first place. I believe that the single most significant obstacle to the proper use of IDS technology in Corporate America is the tendency to treat it (and, indeed, many forms of information technology) as a “black box” – i.e., plug it in, turn it on and wait for it to do its thing. In general, this is a poor way to implement technology, and it is an extremely poor way to implement IDS technologies. IDS excels at identifying unique threat patterns in massive amounts of data whizzing by its sensors, so why not train your IDS to look for those unique patterns that pose the greatest threat to your particular organization? (This is not something a “black box” will ever be capable of doing. The bad news is that we have to do a little bit of work here)

For example, suppose there is an evil competitor or disgruntled former employee you suspect may be seeking to do you harm. Also suppose that you are aware of some unique pattern that an attack from this evil entity would possess (say, a source IP address or the employee’s former userid). Why not program this info into your IDS system as a customized alert? Isn’t this exactly the sort of function your IDS is supposed to serve?

Some critics may argue that this would be an undue invasion of privacy, or that by doing this, we may generate a lot of bogus alerts. To the first point, I argue that this is conceptually no different from looking at the signature for Code Red, Nimda or any of the numerous other nasties that we know to be bona fide threats to the information security of our organizations. The second criticism is a much better point, as IDS are very prone to false positives, and that is precisely why I am emphasizing that the signatures in question need to be UNIQUE to the threat involved. As Intrusion Analysts, we’ve become quite familiar with the experience of wading through large numbers of false positive alerts. However, my experience is that there are in fact unique signatures that can be utilized (and sometimes even self-generated!) that can alert to real threats whilst keeping false positives to a bare minimum.

Let’s examine some situations where customized IDS alerts prove useful.

Example 1 – Major new threat emerges, IDS vendor does not yet have signature available:
In this case, the organization becomes aware of the new threat via the news media, or perhaps a distribution list such as BugTraq. The IDS Analyst checks for an updated signature from the vendor, but it has not yet been released (it is not unusual for several days to pass before the major commercial IDS vendors release signatures for major new threats, and even longer for lesser threats).

Not to fear. All our intrepid IDS analyst requires is some unique set of characteristics describing the new threat and she can easily fabricate her own customized signature. The recently discovered SendMail vulnerability is an excellent example. As of this writing, many commercial IDS vendors have not yet released a detect signature for this particular vulnerability, even though the existence of published exploit code has been reported in the mass media (MSNBC, Cnet, etc).

This sounds very worrisome, but the fact that there is published exploit code is actually a boon to the creative IDS analyst who is willing to create her own alerts! The exploit code should give us the exact pattern we need to build our custom signature.

A very recent (and quite amusing example) of how quickly the Intrusion Detection community can respond to the emergence of new information was seen on April 1st, 2003 with the release of “RFC 3514: The Security Flag in the IPv4 Header” by S. Bellovin of AT&T Labs Research. (ftp://ftp.rfc-editor.org/in-notes/rfc3514.txt). While the entire RFC is clearly intended as a tongue-in-cheek April Fool’s prank, the immediate responses from various IDS practitioners were stunning in their ingenuity (and humor). My personal favorite was the following customized Snort signature, posted by Edward Southcote-Want southcotewant@yahoo.co.uk, within a few hours of the time the original “RFC” was made public on April 1st.

```
alert tcp $EXTERNAL_NET 80 -> $ANY_NET
(msg:"Here PigE, PigE; Here PigE, PigE"; flags: E+; content:"Evil";
reference:arachnids,03; sid:666; classtype:bad-unknown; rev:3;)
```

Example 2 – Detecting Self-Generated Coded Signals from Home Grown Applications:

IDS are a relatively new technology in the corporate world, so it is not all that surprising that application development has largely ignored their existence. But, what if our organization’s applications and automated scripts were designed to be “IDS aware”?

Quite recently, an Application Security Analyst colleague brought an interesting question to me. It appeared that a proposed new application would likely have very weak passwords, and he had a (very valid) concern that it would be highly vulnerable to brute-force attacks. He asked me to verify that if this application was designed to send out a very unique data string upon a certain number of bad passwords attempts, could we then program our IDS to detect that same string and alert us when a potential brute-force attack is in progress. By making the data string as unique as possible, we could reduce the likelihood of false positive to be virtually nil. My response was something along the lines of: “Sure, that’s simple!”
In order to simulate this process, we generated a random string of sixteen hex characters, which we then incorporated into one of our Real Secure sensors. The beauty of this concept is that, provided we have the ability to program our applications to generate a specific pattern, we can tune our IDS systems to be on the alert for that exact same string. We can (and should) make this pattern something that would be virtually impossible to occur solely by random chance.

We assumed that the application would be sending this custom string as part of a URL data-stream to a web-server on one of our IDS monitored subnets. The actual destination is really not important; all that really matters here is that the IDS sensor see the ASCII string on the wire.

Note that this is not a theoretical discussion by any means, we have actually implemented this on a Real Secure sensor responsible for monitoring activity on the internal network segment in my organization. It has yet to produce a single false positive, but has correctly fired on the sixteen character string every time we have tested it.

A customized Snort rule incorporating this same technique looks like this:

```
alert tcp $DMZ_NET 80 -> $DMZ_SENSOR
  (msg "TAT_App: Possible Brute Force Attack!";
  content: "22f867a5d4d488417b47a0bbbe80863bdec569e";
  classtype: custom; rev:1;)
```

Here, “TAT” is the name of the internal application, which lives on a web-server on the DMZ_NET subnet. The application is programmed to automatically send a packet containing the hex string "22f867a5d4d488417b47a0bbbe80863bdec569e" to the existing IDS sensor located on the same DMZ, upon the detection of possible brute force attacks. The IDS will take it from there, sending out the appropriate alerts and logging the attack to the database.

Through this process, we demonstrate that, by generating a totally random string of sufficient length, it is possible to design “IDS aware” alerts into home-grown applications and scripts that are effective warning mechanisms and have extremely low probability of generating false positives.

**Example 3 – Detecting Potential Attacks from Organization-Specific Threats:**
Each organization faces its own unique set of information security risks. This could take the form of a disgruntled former employee, a major competitor set on corporate espionage, or even an irate customer trying to deface a website. By molding our customized IDS alerts to match the profile of the threats affecting our organization, we can much more accurately detect the relevant events of interest.
In the case of the disgruntled former employee, we are likely to know a bit about what an attempted logon attempt from this individual would look like. We certainly know her userid (hopefully now disabled!), and may have other relevant information in our logs, such as home IP address(es), and home computer name(s). Since we have this information, why not use it? It is trivial to program these few facts into a set of customized alerts in our IDS, yet it could potentially provide us with extremely valuable warning that an attack was in progress. Additionally, the IDS logs of any trespass by this individual could prove to be bona fide evidence in a subsequent criminal prosecution.

For example, consider the following customized Snort rule:

```
alert tcp $ANY_NET -> $WIN_DOM_CONTROLLER
    (msg "Disgruntled Ex-Employee Logon Attempt!");
    content: "joe_angry [Windows user-id of employee]";
    classtype: custom; rev:1;)
```

**Summary**

Customized IDS signatures represent an extremely powerful detection enhancement, allowing the IDS analyst to tailor their systems to the unique needs of their organization. The astute IDS analyst should become adept at writing and implementing customized rules in order to maximize their organization’s return on investment in their intrusions detection systems.

Customized signatures can detect brand new or emerging threats for which the IDS vendor has not yet released a signature. They can be used to detect organization-specific threats, such as access attempts generated by a disgruntled former employee or a competitor. They can even be used to enhance the security of internal applications if such applications can be designed and developed in an “IDS-aware” manner.

Clearly, the ability to customize IDS signatures brings many benefits to the table. The wise IDS analyst will cultivate their signature customization skill set to take advantage of these benefits.

**References for Assignment #1:**


Assignment # 2 –
Analysis of Three IDS Detects:

Detect #1 –
Front Page Extensions & Anonymous FTP:

07:41:39.834488 IP adsl-211-78-179-19.TYON.sparqnet.net.64874 > 46.5.180.133.21: P 557163300:557163316 (DF) 0x0000 450 0038 dafa 737b 0000 5553 4552 2061 6e6f P...s(.USER.ano 0x0030 6e79 edf 5753 0dda nymous...

08:47:48.554488 IP 207.178.214.186.3157 > 46.5.180.133.21: P 3660070479:3660070495 (DF) 0x0000 450 0038 f5e0 4000 6e06 93ff cfb2 d6a8 E..8..@.n..=.... 0x0010 2e05 b485 0f47 0015 4133 e7de 6114 d2ca ........G..A3..a... 0x0020 5018 4071 f80d 0000 5553 4552 2061 6e6f P.@q......USER.ano 0x0030 6e79 edf 5753 0dda nymous...

09:08:11.584488 IP 207.178.214.168.4010 > 46.5.180.133.21: P 2039960623:2039960639 (DF) 0x0000 450 0038 f5e0 4000 6e06 93ff cfb2 d6a8 E..8..@.n..=.... 0x0010 2e05 b485 0f47 0015 4133 e7de 6114 d2ca ........G..A3..a... 0x0020 5018 4071 f80d 0000 5553 4552 2061 6e6f P.@q......USER.ano 0x0030 6e79 edf 5753 0dda nymous...

09:08:12.674488 IP 207.178.214.220.3911 > 46.5.180.133.21: P 993837810:993837826 (DF) 0x0000 450 0038 f5e0 4000 6e06 93ff cfb2 d6a8 E..8..@.n..=.... 0x0010 2e05 b485 0f47 0015 4133 e7de 6114 d2ca ........G..A3..a... 0x0020 5018 4071 f80d 0000 5553 4552 2061 6e6f P.@q......USER.ano 0x0030 6e79 edf 5753 0dda nymous...

09:08:13.284488 IP 207.178.214.223.3593 > 46.5.180.133.21: P 1617729588:1617729604 (DF) 0x0000 450 0038 f5e0 4000 6e06 93ff cfb2 d6a8 E..8..@.n..=.... 0x0010 2e05 b485 0f47 0015 4133 e7de 6114 d2ca ........G..A3..a... 0x0020 5018 4071 f80d 0000 5553 4552 2061 6e6f P.@q......USER.ano 0x0030 6e79 edf 5753 0dda nymous...

09:08:16.914488 IP 207.178.214.228.3801 > 46.5.180.133.21: P 1617729588:1617729604 (DF) 0x0000 450 0038 f5e0 4000 6e06 93ff cfb2 d6a8 E..8..@.n..=.... 0x0010 2e05 b485 0f47 0015 4133 e7de 6114 d2ca ........G..A3..a... 0x0020 5018 4071 f80d 0000 5553 4552 2061 6e6f P.@q......USER.ano 0x0030 6e79 edf 5753 0dda nymous...

09:08:18.454488 IP 207.178.214.229.2801 > 46.5.180.133.21: P 1442445886:1442445902 (DF) 0x0000 450 0038 67fa 4000 6e06 21af cfb2 d6a8 E..8..@.n..=.... 0x0010 2e05 b485 0f47 0015 5559 fa3e 6196 73af ..........U..>a.s... 0x0020 5018 2019 4a7f 0000 5553 4552 2061 6e6f P...T"..USER.ano 0x0030 6e79 edf 5753 0dda nymous...

09:08:26.904488 IP 207.178.214.266.1052 > 46.5.180.133.21: P 3249564905:3249564921 (DF) 0x0000 450 0038 67fa 4000 6e06 21af cfb2 d6a8 E..8..@.n..=.... 0x0010 2e05 b485 0f47 0015 5559 fa3e 6196 73af ..........U..>a.s... 0x0020 5018 2019 4a7f 0000 5553 4552 2061 6e6f P...T"..USER.ano 0x0030 6e79 edf 5753 0dda nymous...

© SANS Institute 2003, As part of GIAC practical repository. Author retains full rights.
0x0160  6167  6d61  3a20  6e6f  2d63  6163  6865  0d0a  agma:.no-cache..
0x0170  4361  6368  652d  436f  6e74  726f  6c3a  206d  Cache-Control:m..
0x0180  6178  2d74  6d65  3d30  0d0a  0d0a  ax-stale=0...
0x0190  7468  6f64  3d73  6572  7665  7273  thod=server+vers
0x01a0  6966  6e25  3361  3425  3265  3025  3265  3225  ion%3a4%2e0%2e2%
0x01b0  3265  3236  3131  0d0a  0a  2e2611...

18:44:56.054888 IP 203.241.151.50.16446 > 46.5.180.133.80: P
3203231433:3203231695(262) ack 276032168 win 65535 <nop,nop,timestamp 4462412
7429266>
0x0000  4500  013a  87df  0000  3006  c236  cbf1  9732  E........0..6...2
0x0010  2e05  b485  3d93  0050  beed  6ec9  a487  4000  ..........P.n...
0x0020  0818  ffff  388e  0000  0101  080a  0044  174c  ....8..........D.L
0x0030  071f  5c92  4745  5420  2f5f  7674  695f  696e  .q\._~server/
0x0040  662e  6874  6d6c  2048  545e  542f  312e  30d  f.html.HTTP/1.0.
0x0050  0a4a  0a44  6174  653a  2053  6178  2d73  thod=server+vers
0x0060  3265  3236  3131  0a0d  0a                  2e2611...

18:44:57.184488 IP 203.241.151.50.15763 > 46.5.180.133.80: P
3511070194:3511070583(389) ack 2769305419 win 65535 <nop,nop,timestamp 4462415
7429379>
0x0000  4500  013a  abc8  0000  3006  9dce  cbf1  9732  E........0.......2
0x0010  2e05  b485  403e  0050  da4e  adf2  a510  3f4b  .............D.O
0x0020  0818  ffff  01f1  0000  0101  080a  0044  174c  .............D.O
0x0030  071f  5d03  504f  5354  202f  5f76  7469  5f62  in\shtml.exe/_vt
0x0040  695f  726f  646c  2048  545e  542f  312e  30d  GMT,.MIME-Version
0x0050  652e  7361  6c65  3d30  0d0a  0d0a  6d65  4361  :.no-cache..Cach
0x0060  6368  652d  436f  6e74  726f  6c3a  206d  Cache-Control:.max-s
0x0070  6178  2d73 6561 676d 2030 0d0a 0d0a  tale=0....
07:13:19.714488 IP 81.28.32.103.133 > 46.5.180.133.80: P 4121934171:4121934427(256) ack 2943193079 win 8760 (DF)
0x0000  4500 0128 0e2a 4000 e6c6 b19e 511c 2067 E..(*@.1...g
0x0010  2e05 b485 0533 0050 f5af b95b af6d 9f6f ...3.P...[m....
0x0020  5018 2238 1937 0000 4745 5420 25f6 7674 P."*.GET./_vt
0x0030  695f 6269 6e2f 6773 7320 7672 2064 6c6c i_bin/cwssvr.dll
0x0040  3f55 4c3d 3126 4143 543d 3426 4255 494c ?UL=1&ACT=4&BUTL
0x0050  443d 3236 2653 5452 4d55 3d34 3d34 D=2614&STRMVER=4
0x0060  2643 4150 5245 533d 3020 4858 5540 2f31 &CAPREQ=0.0.HTTP/1
0x0070  3e31 0d0a 4163 6356 7074 3a20 4e44 0d46 .1..Accept:/*;.
0x0080  0a41 6363 6570 742d 456e 636f 6e65 6374 .Accept-Encoding
0x0090  2e68 6572 6f6c 3a20 6e6f 2d63 616e 204b .Host:.www.XXXX
0x00a0  695f 6269 6e2f 6f77 7373 7672 2e64 6c6c .Keep-Alive:.Cac
0x00b0  6865 2d4d 6f64 696e 67 7072 3a20 4e44 .he-Control:--no-c
0x00c0  6865 6d6d 0d0a 0d0a .ache....

11:34:47.744488 IP hnlhil-arl-4-65-052-124.hnlhil.dsl-verizon.net.2114 > 46.5.180.133.80: P 4824952:4825217(265) ack 2314464474 win 8064 (DF)
0x0000  4500 0131 5c4a 4000 e6c6 9e3b 0441 347c E..[@\1...A4;
0x0010  0a05 b485 0842 0050 0049 9f78 89f3 5c3a ......B.P.I.x....
0x0020  0e2a 05d9 0000 4745 5420 25f6 7674 P."*.GET./_vt
0x0030  695f 696e 662f 6773 7320 7672 2064 6c6c i_bin/cwssvr.dll
0x0040  3f55 4c3d 3126 4143 543d 3426 4255 494c ?UL=1&ACT=4&BUTL
0x0050  443d 3236 2653 5452 4d55 3d34 3d34 D=2614&STRMVER=4
0x0060  2643 4150 5245 533d 3020 4858 5540 2f31 &CAPREQ=0.0.HTTP/1
0x0070  3e31 0d0a 4163 6356 7074 3a20 4e44 0d46 .1..Accept:/*;.
0x0080  0a41 6363 6570 742d 456e 636f 6e65 6374 .Accept-Encoding
0x0090  2e68 6572 6f6c 3a20 6e6f 2d63 616e 204b .Host:.www.XXXX
0x00a0  695f 6269 6e2f 6f77 7373 7672 2e64 6c6c .Keep-Alive:.Cac
0x00b0  6865 2d4d 6f64 696e 67 7072 3a20 4e44 .he-Control:--no-c
0x00c0  6865 6d6d 0d0a 0d0a .ache....

© SANS Institute 2003, As part of GIAC practical repository. Author retains full rights.
1. Source of Trace -
I've purposely chosen the raw data file “2002.6.13”, available from the Incidents.Org website at http://www.incidents.org/logs/Raw/, due to the fact that there has been relatively little analysis...
performed on it to date. The output above consists of relevant packets for my analysis which I have output using tcpdump hex and ascii formatting (-X switch enabled).

2. **Detect was generated by** –
The Snort IDS sensors on this network are detecting traffic to and from the Class B 46.5.xxx.yyy IP address space (these are not the actual IP addresses, as they have been sanitized in these log files). Note that ALL the packets in this raw file triggered a Snort alert of some kind, or else they would not have been logged.

The Snort rules that generated these alerts are:

```snort
alert tcp $EXTERNAL_NET any -> $HTTP_SERVERS $HTTP_PORTS
  (msg:"WEB-IIS _vti_inf access"; flags:A+; uricontent:"_vti_inf.html"; nocase; classtype:web-application-activity; sid:990; rev:5;)
```

and:

```snort
alert tcp $EXTERNAL_NET any -> $HTTP_SERVERS $HTTP_PORTS
  (msg:"WEB-FRONTPAGE _vti_rpc access"; flags:A+; uricontent:"/_vti_rpc"; nocase; reference:bugtraq.2144; classtype:web-application-activity; sid:937; rev:6;)
```

I have chosen to focus solely on traffic directed to the IIS/FTP server located at IP address 46.5.180.133, and that falls into one of two categories: they are either Anonymous FTP logons to the server, or they are part of a Front Page publishing attempt directed at this same server.

3. **Probability the source address was spoofed** –

It is extremely unlikely that the source IP addresses were spoofed. We see numerous tcp connections occurring in the raw data file, including several anonymous FTP logons and connections to the Front Page Extensions that are running on this web server. For a complete tcp “3 way connection” to occur, IP spoofing is almost certainly out of the question, unless we are dealing with a highly skilled hacker capable of pulling off some sort of Man in the Middle attack.

It is extremely likely, however, that the attacker(s) may be using open anonymous proxy servers and/or previously compromised hosts to mask the IP address from which they are basing their operations.

4. **Description of attack** –
The IIS web server located at IP address 46.5.180.133 appears to be allowing both Front Page publishing access as well as Anonymous FTP. This can be inferred from the numerous packets of both types directed toward it. These are targeted toward this server; we do not see any other such traffic directed at any other IP address being monitored by this IDS system. Unfortunately, since we have access only to the packets that actually triggered Snort alerts, a rigorous analysis of the entire tcp stream in question is not possible.
Captain John M. Melvin has written an excellent and comprehensive analysis of the Front Page-specific detects in this data file. Please refer to his analysis at http://cert.uni-stuttgart.de/archive/intrusions/2002/09/msg00265.html for a detailed description of the Front Page vulnerabilities present on this server.

We observe numerous anonymous FTP logons to the server from 11 distinct IP addresses. Virtually all of these logons originated from DSL connections with major ISPs. Interestingly, we see these anonymous FTP connections originating in several different nations on three continents: Germany, Japan, Korea, and the US.

The Front Page connections to the server originate from 5 distinct IP addresses, 2 of which have recent complaints lodged against them at www.Dshield.org:

IP Address: 218.17.234.225
HostName: 218.17.234.225
DSHield Profile: Country:
Contact E-mail:
Total Records against IP: 204
Number of targets: 76
Date Range: 2003-02-15 to 2003-02-17

Fightback: not sent
Whois: inetnum: 218.13.0.0 - 218.18.255.255
netname: CHINANET-GD
country: CN
descr: CHINANET Guangdong province network
       Data Communication Division
       China Telecom
admin_c: CH93-AP
tech_c: WM12-AP
remarks: MAINT-CHINANET
changed: hostmaster@ns.chinanet.cn.net 20010528
status: ALLOCATED PORTABLE
source: APNIC
notify:
mnt_lower: MAINT-CHINANET-GD
rev_srv:
start: 3658285056
end: 3658678271
diff: 393215
person: WU MIAN
address: NO.1,RO.DONGYUANHENG,YUEXIUNAN,GUANGZHOU
country: CN
phone: +086-20-83877223
fax_no: +86-20-83877223
e_mail: ipadm@gddc.com.cn
nic_hdl: WM12-AP
mnt_by: MAINT-CHINANET-GD
changed: ipadm@gddc.com.cn 20010820
source: APNIC

From this, we see that this individual is using an IP address belonging to China Telecom, located in Guangdong province. We also see that this IP address has over 200 prior complaint records logged against it in Dshield. These prior complaints are based on prior suspicious traffic from this user’s current IP to 76 different hosts. While there is a good possibility that the ISP
uses DHCP to assign IP addresses, and that our user may have obtained a “Dshield tainted” IP address completely by chance, this is still evidence of malfeasance sufficient to raise our suspicions regarding traffic from this IP address.

5. Attack mechanism –
There is significant evidence that the IIS service in question has been set in a very permissive and lenient manner. Captain John Melvin has performed an exhaustive examination of the possible attack vectors against these exact same Front Page Extensions in his excellent analysis (available at http://cert.uni-stuttgart.de/archive/intrusions/2002/09/msg00265.html).

However, I draw a very different conclusion from Captain Melvin regarding the intent of the Front Page traffic. He concluded that the repeated accesses to the Front Page extensions were valid traffic from authorized users making changes to the web server’s content. I propose that a more malicious intent is behind this traffic.

My conclusion is that this web server is being used as a Warez repository (i.e., a storage location hackers use to store and exchange illegally obtained software). In support of this conclusion, I offer the following observations:

1) The fact we see numerous anonymous FTP connections from DSL lines all over the world are consistent with this being a Warez server. If it truly is a Warez server, its IP address is likely to be disseminated via hacker websites and IRC chat rooms. This would explain why we are seeing so many anonymous connections from so many disparate geographic locations.

2) The Front Page authoring connections to the server could be the mechanism by which new files are transferred to the server. Remember that two of the source IP addresses have prior complaints against them at Dshield.org. This is consistent with the theory that these IP addresses belong to nefarious individuals who might well be trafficking in warez.

3) Warezers have been widely known to utilize the Front Page Extensions as a convenient mechanism for managing hordes of pirated software. This is due to a combination of ease-of-use and subterfuge. Publishing files to an IIS web server with Front Page Extensions is extremely easy, while the convoluted directory structure created by Front Page makes a great place to hide files away from prying SysAdmin eyes. I have personally witnessed this firsthand on compromised IIS servers that I have had the pleasure of rehabilitating. A Google search of “_vti” and “warez” yields 41 hits, many of which are actual warez sites with subdirectories named _vti.

6. Correlations –
Captain John Melvin’s analysis of the Front Page configuration issues on this same server: http://cert.uni-stuttgart.de/archive/intrusions/2002/09/msg00265.html).

Dshield.Org analysis of IP Address:
http://www.dshield.org/ipinfo.php
Corroborating evidence of this type of Warezer behavior can be found in the following FTP log at http://www.incidents.org/archives/intrusions/msg02584.html:

WAREZ Obfuscation on World-Writable FTP [LOGS]

Date: Mon, 7 Jan 2002 16:44:07 -0600
From: "James Crossman" <jcrossman@xxxxxxxxxxxxx>
Subject: WAREZ Obfuscation on World-Writable FTP [LOGS]

My partner is busy fighting these WAREZ and wants to be left alone while he works on the files, so let me share with you a couple of log extracts I pieced together... The tricks they tried include renaming files and directories after they were made and one renamed a directory to a .DOC extension and locked up windows explorer every time....

Log #1 - Uncopyable (or deletable) Nirvana


7. Evidence of Active Targeting –
There is ample evidence of active targeting here. The Front Page accesses are directed solely at the IP address of this web server; we do not see any attempts to access any Front Page components on any other IP address in the data file. The same is true of the anonymous FTP connections - these are also directed solely at the web server at IP 46.5.180.133. We do not witness any other anonymous FTP connection attempts directed at any other IP address in the 2002.6.13 raw data file.

I conclude that some of the Front Page accesses we see in the raw data file are potentially publications of new Warez files to the compromised server, probably by the individual(s) who
originally performed the compromise. The anonymous FTP accesses are downloads of these Warez files by the end user hackers, who probably learned of its IP address through a hacker web site or chat room.

8. **Severity** –

Severity = (Criticality + Lethality) – (System Countermeasures + Network Countermeasures)

Criticality: importance of the targeted system. The IIS web server in question is open to web traffic from all over the world, and is likely housed in a DMZ. It is therefore likely that no highly sensitive data is stored on this server. I will assign a rating of 3 here.

Lethality: How severe would the damage be if the attack succeeds. Having your server used as a Warez repository is obviously a major nuisance, and there are certainly liability issues that could arise from this scenario. However, there is no evidence that there has been any significant damage to the server itself (indeed, it appears to be serving its warez in a very robust fashion, nor does it seem likely that a total system compromise is likely). I will assign a 3, primarily due to the potential liability issues that could arise from this scenario, and also due to the potential theft of bandwidth and server resources.

System Countermeasures: We have no way of knowing which service patches and hot-fixes have been applied to this server, but it does appear that the configuration of the Front Page extensions and anonymous FTP access are quite lenient. Anonymous FTP should only be allowed if it is absolutely essential for some business function. I will assign a 2 in this category.

Network Countermeasures: Port 80 is wide open at the firewall by necessity (this is a publicly-accessible web server, after all). All IP addresses appear to be allowed access, regardless of geographic location. An IDS is obviously in place, or we would not be analyzing this detect in the first place. I will assign a rating of 2 here.

**Severity** = (3 + 3) – (2 + 2) = 2

9. **Defensive Recommendation** –

The SysAdmin for this server should **immediately** examine it for any evidence that it is being used as a Warez repository. A file-system search on all local logical drives for the most recently modified files should answer this question posthaste. If the server really is being used as a Warez repository, this needs to be addressed ASAP by deleting all the illegal software and tightening up the Front Page exposures that allowed this situation to emerge in the first place.

I would recommend completely disabling both the Front Page Extensions and the Anonymous FTP access, unless they are absolutely needed. If they are needed, it is essential to ensure that they are fully patched with all the latest service packs and hot-fixes. Accounts using the Front Page extensions should be subjected to rigorous password auditing.

As with any IIS web server, the SysAdmin would be wise to run the IIS Lockdown Tool and install the URLScan ISAPI filter as documented by Microsoft at these URLs:

IIS Lockdown Tool:
10. Multiple Choice Question –

What is this user up to?

a) downloading a Nirvana song from a commercial music service.
b) creating a hidden storage repository for illegally-obtained copyrighted materials.
c) sharing his own Nirvana music collection on the Gnutella network.
d) playing “Smells Like Teen Spirit” on his MP3 player.

Correct answer –

B – each “MKD” command is making a subdirectory on the FTP server. The user is attempting to hide his or her stash of copyrighted MP3s in an obscure location deep within a subdirectory structure that looks ostensibly like a normal Front Page “_vti” subdirectory.

Note:

Detect #1 was submitted to the “incidents.org” mailing list on March 28th, and is publicly accessible here in the mailing list archives:
http://cert.uni-stuttgart.de/archive/intrusions/2003/03/msg00378.html

I received one public response from Andrew Rucker Jones, it is also available here:
http://cert.uni-stuttgart.de/archive/intrusions/2003/03/msg00385.html
Detect #2 –
A Visit from a Curious Friend:

[**] [1:895:5] WEB-CGI redirect access [**]
[Classification: Attempted Information Leak] [Priority: 2]
03/28-03:15:28.145975 66.77.73.84:2019 -> xxx.yyy.zzz.142:80
TCP TTL:50 TOS:0x0 ID:51084 IpLen:20 DgmLen:285 DF
***AP*** Seq: 0x35157923 Ack: 0x54B502E Win: 0xFFFF TcpLen: 20
[Xref => cve CVE-2000-0382][Xref => bugtraq 1179]

[**] [1:895:5] WEB-CGI redirect access [**]
[Classification: Attempted Information Leak] [Priority: 2]
03/28-03:16:58.730474 66.77.73.84:3254 -> xxx.yyy.zzz.142:80
TCP TTL:50 TOS:0x0 ID:5525 IpLen:20 DgmLen:285 DF
***AP*** Seq: 0x51A0AA09 Ack: 0x54B507E Win: 0xFFFF TcpLen: 20
[Xref => cve CVE-2000-0382][Xref => bugtraq 1179]

[**] [1:895:5] WEB-CGI redirect access [**]
[Classification: Attempted Information Leak] [Priority: 2]
03/28-03:18:31.737326 66.77.73.84:4373 -> xxx.yyy.zzz.142:80
TCP TTL:50 TOS:0x0 ID:21836 IpLen:20 DgmLen:285 DF
***AP*** Seq: 0x85E69E2D Ack: 0x54B50BE Win: 0xFFFF TcpLen: 20
[Xref => cve CVE-2000-0382][Xref => bugtraq 1179]

[**] [1:895:5] WEB-CGI redirect access [**]
[Classification: Attempted Information Leak] [Priority: 2]
03/28-03:19:32.95718 66.77.73.84:1274 -> xxx.yyy.zzz.142:80
TCP TTL:50 TOS:0x0 ID:43066 IpLen:20 DgmLen:285 DF
***AP*** Seq: 0xC5308CCF Ack: 0x54B5117 Win: 0xFFFF TcpLen: 20
[Xref => cve CVE-2000-0382][Xref => bugtraq 1179]

[**] [1:895:5] WEB-CGI redirect access [**]
[Classification: Attempted Information Leak] [Priority: 2]
03/28-03:20:01.852392 66.77.73.84:4580 -> xxx.yyy.zzz.142:80
TCP TTL:50 TOS:0x0 ID:23578 IpLen:20 DgmLen:285 DF
***AP*** Seq: 0x365314FD Ack: 0x54B512F Win: 0xFFFF TcpLen: 20
[Xref => cve CVE-2000-0382][Xref => bugtraq 1179]

[**] [1:895:5] WEB-CGI redirect access [**]
[Classification: Attempted Information Leak] [Priority: 2]
03/28-03:20:31.972895 66.77.73.84:2166 -> xxx.yyy.zzz.142:80
TCP TTL:50 TOS:0x0 ID:49342 IpLen:20 DgmLen:285 DF
***AP*** Seq: 0x3D0E47B2 Ack: 0x54B5184 Win: 0xFFFF TcpLen: 20
[Xref => cve CVE-2000-0382][Xref => bugtraq 1179]
1. Source of Trace -  
For the past month, I have been capturing the first 300 bytes of every packet passing through the Internet-facing subnet on my home network. This is being accomplished with a Linux server running tcpdump in promiscuous mode on eth0. Periodically, I run the captured traffic through a default Snort configuration using the default rule set.

2. Detect was generated by –  
The network detect is from a Snort intrusion detection system with a standard 1.9.1 rule set.

The Snort rule that generated the alert was:

```
alert tcp $EXTERNAL_NET any -> $HTTP_SERVERS 80 (msg:"WEB-CGI redirect access";flags: A+; uricontent:"/redirect"; nocase;reference:bugtraq,1179; reference:cve,CVE-2000-0382; classtype:attempted-recon; sid:895; rev:2;)
```

3. Probability the source address was spoofed –  
It is extremely unlikely that the source IP address was spoofed. This alert is triggered by an automated reconnaissance probe against a web server that is co-located on my network. In order to get worthwhile information back, the “attacker” must complete the 3-way tcp handshake and supply a valid IP address under her control.
I will present more information regarding the attackers motivations in the following discussion.

4. Description of “attack” –

These CGI Redirect packets are being sent to a co-located web server on my network that hosts a very popular hobbyist bulletin board and chat room. There is substantial evidence that this is a benign reconnaissance probe being conducted by an automated “web-bot” that has been programmed to search out and catalog web content.

A lookup of the source IP address in Dshield yields up the following:

IP Address: 66.77.73.84
HostName: cr045r01-2.sac2.fastsearch.net
DShield Profile: Country: US
Contact E-mail: ip-admin@qis.qwest.net
Total Records against IP: 36
Number of targets: 2
Date Range: 2003-01-19 to 2003-02-02

<table>
<thead>
<tr>
<th>Port</th>
<th>Attacks</th>
<th>Start</th>
<th>End</th>
</tr>
</thead>
<tbody>
<tr>
<td>80</td>
<td>155</td>
<td>2003-02-27</td>
<td>2003-03-26</td>
</tr>
</tbody>
</table>

Fightback: not sent
Whois:
CustName: Fast Search, Inc.
Address: 93 Worcester Street, 4th Floor
City: Wellesley
StateProv: MA
PostalCode: 02481
Country: US
RegDate: 2002-01-10
Updated: 2002-01-10

NetRange: 66.77.73.0 - 66.77.73.255
CIDR: 66.77.73.0/24
NetName: QWEST-MCC-FASTSRCH3
NetHandle: NET-66-77-73-0-1
Parent: NET-66-77-0-0-1
NetType: Reassigned
Comment: 
RegDate: 2002-01-10
Updated: 2002-01-10

Clearly, ours is not the only server on the WWW that the folks at Fast Search, Inc. have been checking out. A quick check on their website at http://www.fastsearch.net/products/websearch/index.asp yields some very relevant information concerning one of their products:

The FAST Web Search index powers portals that reach the majority of European searches and over a quarter of all US searches--over 100 million users worldwide. Unlike other search engines, every query we receive is submitted across the entire catalog and results are returned in less than half a second. In addition to our industry-leading relevancy algorithms, we are committed to providing our users with the largest and freshest catalog on the Net.
5. “Attack” mechanism –

First, we see the FastSearch Web Bot making a normal TCP connection to the web server on port 80, with completion of the 3-way handshake (Syn, Syn Ack, Ack are bolded):

```
21:40:57.256310 IP 66.77.73.84.1138 > xxx.yyy.zzz.142.80: S 2236556219:2236556219(0) win 65535 <mss 1380,nop,wscale 1,nop,nop,timestap 2438146351 0> (DF) 0x0000 4500 000c 4606 4000 3206 bb34 424d 4954 E..<E@.2..BMIT 0x0010 d151 ea8e 0472 0050 854f 23bb 0000 0000 .Q...r.P.O#.... 0x0020 a002 ffff 8e63 0000 0204 0564 0103 0301 ...........d.. 0x0030 0101 080a 9401 e223 0000 0000 ..........d....
```

This Web Bot does not waste any time!! It already knows what files it wants, and it only wants them if they have not changed since February 28 (probably the last time it paid a visit to this web-site). The current visit occurred in the wee hours March 28th, exactly one month later. I doubt this is a coincidence. Evidently, FastSearch’s web bots update their content on a monthly basis.

```
21:40:57.256590 IP xxx.yyy.zzz.142.80 > 66.77.73.84.1138: S 85585584:85585584(0) ack 2236556220 win 8280 <mss 1460> (DF) 0x0000 4500 002c 6b8c 4000 8006 47be d151 ea8e E..,k.@.. 0x0010 424d 4954 0050 0472 0519 eeb0 854f 23bc BMIT.P.r.....O#. 0x0020 6012 2058 8ea5 0000 0204 05b4 0000 `..X..........
```

21:40:57.359989 IP 66.77.73.84.1138 > xxx.yyy.zzz.142.80: . ack 1 win 65535 (DF) 0x0000 4500 0028 464e 4000 3206 ba2e 424d 4954 E..(FN@.2...BMIT 0x0010 d151 ea8e 0472 0050 854f 23bc 0519 eeb1 .Q...r.P.O#.... 0x0020 0x0030 5010 ffff c6ba 0000 0204 0564 0103 P..........d..

This Web Bot does not waste any time!! It already knows what files it wants, and it only wants them if they have not changed since February 28 (probably the last time it paid a visit to this web-site). The current visit occurred in the wee hours March 28th, exactly one month later. I doubt this is a coincidence. Evidently, FastSearch’s web bots update their content on a monthly basis.
Here the web bot is double-checking for instructions in a “robots” META file. These files are used by savvy webmasters to express their preferences for content inclusion in the web bots’ index. In general, this is a well-behaved web-bot, following the standard conventions and Netiquette expected of an automated web content update agent.

The original purpose of the “WEB-CGI redirect access” Snort rule was to warn of potential attempts to exploit a known vulnerability in older versions of Allaire’s ClusterCats load balancer and ColdFusion products. Detailed information regarding this vulnerability is available here: http://online.securityfocus.com/bid/1179/info/

Note that this rule will fire an any http request containing the text string “/redirect”. Given that the webserver in question uses http redirects by necessity, and does not run any of the vulnerable Allaire products, this particular Snort rule will likely yield nothing but false positives, at least with regard to this particular web server.
6. Correlations –
Description of the FAST “Web Search” product:
http://www.fastsearch.net/products/websearch/index.asp

Advisories regarding the original vulnerabilities in Allaire ColdFusion and ClusterCats:
http://online.securityfocus.com/bid/1179/info/

David Leadston’s Practical Detect posting to the Incidents.Org mailing list regarding a
ClusterCats detect:
http://cert.uni-stuttgart.de/archive/intrusions/2002/10/msg00049.html

7. Evidence of Active Targeting –
There is ample evidence of active targeting here. Out of 14 routable IP addresses on my
network, this “attack” was only directed at ONLY ONE address, namely xxx.yyy.zzz.241. This
is especially clear when one consider that there are other active web servers on the network that
were not targeted at all. The “attacker” clearly has selected this particular IP address with some
very specific goal in mind. That goal is to update their catalog of information regarding the web
content of this server.

8. Severity –
Severity = (Criticality + Lethality) – (System Countermeasures + Network Countermeasures)

Criticality: Importance of the targeted system. This is a co-located web-server on the Internet
facing-DMZ of my home network. While it does not contain any highly sensitive data, it is the
home base of a very popular hobbyist bulletin-board and chat room frequented by avid home-
brewers. These hobbyists would suffer significant disappointment if the server were negatively
impacted. I will assign a 2 to Criticality.

Lethality: How severe is the damage if the attack succeeds? In this particular case, a successful
“attack” is actually something rather positive: It results in up-to-date information regarding the
hobbyist web-site and its services being made available to a broader audience. I will assign a 1
here.

System Countermeasures: The web-server’s System Administrators are extremely security
conscious, and apply any patches relevant to major new vulnerabilities immediately upon
notification. However, sometimes they are too busy to immediately address vulnerabilities
receiving lower levels of notoriety. I assign a 4 for System Countermeasures.

Network Countermeasures: Port 80 is wide open at the firewall by necessity (this is a publicly-
accessible web-server, after all). All IP addresses are allowed access, regardless of geographic
location. An IDS is obviously in place, or we would not be analyzing this detect in the first
place. I will assign a rating of 2 here.
Severity = (2 + 1) – (4 + 2) = -3

9. Defensive Recommendation –
Existing system & network countermeasures are adequate. Given that there are no vulnerable Allaire products at this site, this Snort rule can be safely disabled without ill effect.

However, SysAdmins should continue to monitor for new service packs and patches relevant to the server.

10. Multiple Choice Question –
What is the most likely explanation for the trace below?

```
21:40:59.408512 IP xxx.yyy.zzz.142.80 > 66.77.73.84.1218: 1218: P 1:605(604) ack 170 win 8111 (DF)
0x0000 4500 0284 7a8c 4000 8006 3666 d151 ea8e E...z.@...6f.Q..
0x0010 5018 1faf cdff 0000 4854 5450 2f31 2e31 BMIT.P....@...
0x0020 2034 3034 204f 0a53 6164 3e3c 7469 746c 653e 4572 726f 7220 Error.
0x0030 3430 343c 2f74 6974 6c65 3e0d 0a0d 3c4d 4554 2048 5454 502d 4551 5549 s".content="noindex">...
```

a) a hacker is searching this web server for “warez”, without success.
b) Code Red has tried and failed to infect this web server.
c) An automated web content indexing agent is doing its job.
d) A vulnerability scanner is being run against this web server.

Correct answer:
c) An automated web content indexing agent is doing its job.

(the “meta.name=’robots’.content=’noindex’” text is a very strong indication this is an automated web-indexing agent at work, cataloging web content).
Detect #3 –
“It’s Code Red! No, It’s Nimda! No, it’s, it’s....???”

```
[Classification: Web Application Attack] [Priority: 1]
TCP TTL:113 TOS:0x0 ID:47201 IpLen:20 DgmLen:106 DF
***AP*** Seq: 0x171A0707 Ack: 0xBC7D52 Win: 0x2238 TcpLen: 20
[Xref => url www.cert.org/advisories/CA-2001-19.html]

[**] [1:1002:5] WEB-IIS cmd.exe access [**]
[Classification: Web Application Attack] [Priority: 1]
TCP TTL:113 TOS:0x0 ID:52577 IpLen:20 DgmLen:150 DF
***AP*** Seq: 0x819EE1D9 Ack: 0xBC7D67 Win: 0x2238 TcpLen: 20

[Classification: Web Application Attack] [Priority: 1]
TCP TTL:113 TOS:0x0 ID:6242 IpLen:20 DgmLen:106 DF
***AP*** Seq: 0xBC7D93 Win: 0x2238 TcpLen: 20
[Xref => url www.cert.org/advisories/CA-2001-19.html]

[**] [1:1002:5] WEB-IIS cmd.exe access [**]
[Classification: Web Application Attack] [Priority: 1]
TCP TTL:113 TOS:0x0 ID:16482 IpLen:20 DgmLen:156 DF
***AP*** Seq: 0xDC5A2460 Ack: 0xBC7DBA Win: 0x2238 TcpLen: 20
[Xref => cve CVE-2000-0884]
```

```
[**] [1:1945:1] WEB-IIS unicode directory traversal attempt [**]
[Classification: Web Application Attack] [Priority: 1]
TCP TTL:113 TOS:0x0 ID:21602 IpLen:20 DgmLen:144 DF
***AP*** Seq: 0x265DDBF6 Ack: 0xBC7DB4 Win: 0x2238 TcpLen: 20
[Xref => cve CVE-2000-0884]
```

```
[**] [1:1002:5] WEB-IIS cmd.exe access [**]
[Classification: Web Application Attack] [Priority: 1]
TCP TTL:113 TOS:0x0 ID:23138 IpLen:20 DgmLen:156 DF
***AP*** Seq: 0xAC25C5CD Ack: 0xBC7DC9 Win: 0x2238 TcpLen: 20
```

```
[**] [1:1002:5] WEB-IIS cmd.exe access [**]
[Classification: Web Application Attack] [Priority: 1]
TCP TTL:113 TOS:0x0 ID:23138 IpLen:20 DgmLen:156 DF
***AP*** Seq: 0xAC25C5CD Ack: 0xBC7DC9 Win: 0x2238 TcpLen: 20
```
TCP TTL:113 TOS:0x0 ID:28002 IpLen:20 DgmLen:145 DF
***AP*** Seq: 0xA3DAFDC4 Ack: 0xBC7DDA Win: 0x2238 TcpLen: 20

[**] [1:1945:1] WEB-IIS unicode directory traversal attempt [**]
[Classification: Web Application Attack] [Priority: 1]
03/29-06:27:32.076022 202.136.189.51:3419 -> xxx.yyy.zzz.142:80
TCP TTL:113 TOS:0x0 ID:28514 IpLen:20 DgmLen:144 DF
***AP*** Seq: 0x2A236276 Ack: 0xBC7DEE Win: 0x2238 TcpLen: 20
[Xref => cve CVE-2000-0884]

[**] [1:1002:5] WEB-IIS cmd.exe access [**]
[Classification: Web Application Attack] [Priority: 1]
TCP TTL:113 TOS:0x0 ID:36450 IpLen:20 DgmLen:145 DF
***AP*** Seq: 0xCB00FFFA Ack: 0xBC7DEE Win: 0x2238 TcpLen: 20

[**] [1:1002:5] WEB-IIS cmd.exe access [**]
[Classification: Web Application Attack] [Priority: 1]
TCP TTL:113 TOS:0x0 ID:41570 IpLen:20 DgmLen:145 DF
***AP*** Seq: 0x2D1B1B17 Ack: 0xBC7DEE Win: 0x2238 TcpLen: 20

[**] [1:1002:5] WEB-IIS cmd.exe access [**]
[Classification: Web Application Attack] [Priority: 1]
TCP TTL:113 TOS:0x0 ID:36450 IpLen:20 DgmLen:145 DF
***AP*** Seq: 0xCB00FFFA Ack: 0xBC7DEE Win: 0x2238 TcpLen: 20

[**] [1:1002:5] WEB-IIS cmd.exe access [**]
[Classification: Web Application Attack] [Priority: 1]
03/29-06:27:34.140866 202.136.189.51:3431 -> xxx.yyy.zzz.142:80
TCP TTL:113 TOS:0x0 ID:44642 IpLen:20 DgmLen:139 DF
***AP*** Seq: 0x59C34B71 Ack: 0xBC7DEE Win: 0x2238 TcpLen: 20

[**] [1:1945:1] WEB-IIS unicode directory traversal attempt [**]
[Classification: Web Application Attack] [Priority: 1]
03/29-06:27:34.140866 202.136.189.51:3431 -> xxx.yyy.zzz.141:80
TCP TTL:113 TOS:0x0 ID:49506 IpLen:20 DgmLen:139 DF
***AP*** Seq: 0x380E5209 Ack: 0xBC7DEE Win: 0x2238 TcpLen: 20
[Xref => cve CVE-2000-0884]

This pattern of alerts continues repeatedly at this very rapid pace until 6:40am — a total attack window of almost 13 solid minutes, hundreds of alerts were generated during this timeframe!

1. Source of Trace
For the past month I have been capturing the first 300 bytes of every packet passing through the Internet-facing subnet on my home network. This is being accomplished with a Linux server in my DMZ running tcpdump in promiscuous mode on ‘eth0’. Periodically, I run all the captured traffic through a Snort configurations using the default rule set.

2. Detect was generated by –
The network detect is from a Snort intrusion detection system with a standard default 1.9.1 rule set.

The Snort rules that generated these alerts were:
"WEB-IIS CodeRed v2 root.exe access" alert:

"WEB-IIS cmd.exe access" alert:
alert tcp $EXTERNAL_NET any -> $HTTP_SERVERS $HTTP_PORTS (msg:"WEB-IIS cmd.exe access"; flow:to_server,established; content:"cmd.exe"; nocase; classtype:web-application-attack; sid:1002; rev:5;)

"WEB-IIS unicode directory traversal attempt" alert:
alert tcp $EXTERNAL_NET any -> $HTTP_SERVERS $HTTP_PORTS (msg:"WEB-IIS unicode directory traversal attempt"; flow:to_server,established; content:"/..%c0%af../"; nocase; classtype:web-application-attack; reference:cve,CVE-2000-0884; sid:981; rev:6;)

3. Probability the source address was spoofed –
It is very unlikely that the source IP address was spoofed. These types of web application exploits are explicitly directed at obtaining a root shell on a vulnerable IIS web-server. In order to do this, the attacker would need to somehow establish a successful three-way tcp handshake with the web-server in question, and would need access to the system that originated the connection.

However, there is an excellent probability that the attacking system is a previously root-compromised system under the attacker’s direct control that he or she is using as an “attack host” to direct any resulting scrutiny elsewhere.

4. Description of the attack –
Here is what we know about the attacking IP address (from www.Dshield.Org):

IP Address: 202.136.189.51
HostName: mail.glorymhm.com.sg
Total Records against IP: 1372
Number of targets: 2
Ports Attacked (up to 10):
<table>
<thead>
<tr>
<th>Port</th>
<th>Attacks</th>
<th>Start</th>
<th>End</th>
</tr>
</thead>
<tbody>
<tr>
<td>80</td>
<td>2058</td>
<td>2003-03-25</td>
<td>2003-03-25</td>
</tr>
</tbody>
</table>

netname: NTTS
country: SG
descr: NTT Singapore Pte.Ltd.
admin_c: OO2-AP
tech_c: OO2-AP
remarks: NTT Singapore plans to start internet data center service and internet access service on April. 1st.
mnt_by: APNIC-HM
changed: hostmaster@apnic.net 20010222
status: ALLOCATED PORTABLE
source: APNIC
mnt_lower: MAINT-SG-NTTS
start: 3397951488
diff: 3397959679
end: 8191
Clearly, this host has been used to mount significant suspicious activities against web-servers before! Further investigation is definitely warranted.

At first glance this network traffic pattern could easily be mistaken as a routine automated probe of my network by a web-server infected with the Nimda worm. The attempted exploits shown in the trace above are the exact same ones used by Nimda in its efforts to compromise any IIS web-servers it may find in its random searches of routable Internet IP space.

However, upon closer examination, this traffic is not consistent with Nimda (at least not any incarnation of Nimda that I have seen to date). Nimda is not this persistent. It will generally hit a target server with several unicode and cmd.exe attempts in extremely short order (under one minute). It will either infect the server within this time or not, but either way it will be moving on to its next target much more quickly than we see here. This makes sense; it could certainly not have become one of the more successful and notorious worms in Internet history if it spent multiple minutes trying to infect servers that are not vulnerable.

But if it’s not Nimda, what is it? Probably some kind of automated attack script or maybe even a commercial vulnerability scanning tool, but we need to perform more analysis to say anything conclusive.

5. Attack Mechanism –
Further investigation is definitely warranted, so let’s have a close look at some to the tcpdump capture files from this timeframe:

```
tcpdump -r capturefile -X -n ‘host 202.136.189.51’ > textfile
```

Looking through the resulting textfile, we can easily how the attack began:

```
06:26:14.003387 IP 202.136.189.51 > xxx.yyy.zzz.128: icmp 17: echo request seq 2897 0x0000 4500 0025 225b 0000 7101 e3ee ca88 bd33 E..%"[...q....3 0x0010 d151 ea80 0800 297d 0100 0b51 6b69 6c6c .Q..)}...Qhell 0x0020 6f20 3f3f 3f00 0000 0000 0000 .o.???...........
06:26:14.003960 IP xxx.yyy.zzz.133 > 202.136.189.51: icmp 17: echo reply seq 2897 0x0000 4500 0025 225b 0000 7101 e3ee ca88 bd33 E..%"[...q....3 0x0010 d151 ea80 0800 297d 0100 0b51 6b69 6c6c .Q..)}...Qhell 0x0020 6f20 3f3f 3f00 0000 0000 0000 .o.???...........
06:26:14.005830 IP xxx.yyy.zzz.139 > 202.136.189.51: icmp 17: echo reply seq 2897 0x0000 4500 0025 225b 0000 7101 e3ee ca88 bd33 E..%"[...q....3 0x0010 d151 ea80 0800 297d 0100 0b51 6b69 6c6c .Q....1}...Qhell 0x0020 6f20 3f3f 3f00 0000 0000 0000 .o.???...........
```

© SANS Institute 2003, As part of GIAC practical repository. Author retains full rights.
First, the attacking host sends an icmp echo request packet to the target IP address in a fairly typical (and not always effective) attempt to identify whether a host exists at this address at all. But it gets two replies, from two different IP addresses. Why is this? In this case, the .128 IP address is actually a host network broadcast address, as my /28 subnet of routable IP space begins at .128 and ends at .144, with the endpoint addresses being reserved for host net and broadcast purposes. This automated tool appears to be oblivious to this fact, however, and actually ignores the source IP addresses in the replies that are sent back. It then proceeds to send two SYN connection attempt to port 80 on the original IP it probed. Neither of these is successful. Luckily, neither of my chatty hosts that feels the need to respond to broadcast pings has anything running on tcp port 80.

That’s it for traffic directed at .128; now the process repeats against other IP addresses:

Uh OH!! Syn, Syn-Ack, ACK! The 3 way handshake is now complete, a tcp connection is made, and my web-server at .130 may be in for some big trouble! Let’s see what happens next:
A simple HTTP 1.0 HEAD request to the root directory of the web server. My stalwart webserver at .130 replies:

```
0x0000  4500 00bc 5843 4000 8006 5e68 d151 ea82 E...XC@...^h.Q..
0x0010  ca88 bd33 0050 0c6e 160a aa2f e875 059d ...3.P.n...u.
0x0020  5018 220f e87f 0000 4854 5450 2f31 2e31 P.".....HTTP/1.1
0x0030  2034 3011 2041 6363 6573 7320 4465 6e69 .401.Access.Deni
0x0040  6564 0d0a 486f73743a 2032 3039 2e38 312e 3233 342e 31 1.234.130...
0x0050  300d 0a48 6f73743a 2032 3039 2e38 312e 3233 342e 31 1.234.130...
0x0060  300d 0a0d 4650 00bc 5843 4000 7106 aee3 ca88 bd33 E.\@.\.
0x0070  d151 ea82 0c6e 0050 e875 059d 160a aac3 .Q...n.P.u.
0x0080  5011 21a4 8f00 0000 0204 05b4 0a0d P.!...........
0x0090  4500 0028 195c 4000 7106 acdf ca88 bd33 E.\(@.).Q..3
0x00a0  d151 ea82 0c6e 0050 e875 059d 160a aac4 .Q...n.P.u.
0x00b0  5010 21a4 8f00 0000 0204 05b4 0a0d P.!...........
```

Hey, you need some NTLM logon credentials to get anything outta me, d00d!! Have a nice day!” Good job, .130! .130 has told the attacker what he can do with his automated attack script. Our attacker is not yet done with .130 though:

```
06:26:25.371984 IP 202.136.189.51.3184 > xxx.yyy.zzz.130.57: S 4193413308:4193413308(0) win 8192 <mss 1460> (DF)
0x0000  4500 002c 195c 4000 7106 ade3 ca88 bd33 E.\.@.\.
0x0010  d151 ea82 0c6e 0050 e875 059d 160a aac4 .Q...n.P.u.
0x0020  5011 21a4 8f00 0000 0204 05b4 0a0d P.!...........
```

© SANS Institute 2003, Author retains full rights. As part of GIAC practical repository.
No dice here.; we don’t speak no tcp port 57 round these parts.  From the timing of the packets, I believe the repetitions are due to tcp retries at the stack level.  I had to look up the significance of tcp port 57.  It is a Mail Transfer Protocol (MTP) described in RFC 772 and 780.

Undaunted, our attacker (or his automated minion) moves on.  Since we are quite familiar with the sequence 1) ICMP echo request, 2) SYN connect to port 80, 3) HTTP 1.0 HEAD request, 4) probe tcp ports 57 and 21, I will not show these packets in the subsequent analysis.  Our automated attacker continues upward with this now very familiar pattern sequentially upward through the IP address space of my network.  Nothing interesting happens until we get to IP .141, port 80:

Uh oh, – an “HTTP 302 – Object moved message”.  This is actually an open, publicly accessible web-server. How will our automated attack buddy react?

Like this:

```
```

He's also looking for FTP servers.  Again, no dice on this host.
Not nice!! The minute the attack tool smells a web server that might be vulnerable, it starts throwing several well-known IIS exploits at the server, not just once, but in a systematic effort to test every possible permutation of subdirectory locations that might possibly used to get at cmd.exe! (There are literally hundreds of attempts to exploit cmd.exe through the MDAC, Unicode, and Double-Decode vulnerabilities in these logs. I will spare the gentle reader all of the gory details).
Luckily, in this particular case the web-server in question was up the latest patch levels, and was not vulnerable to this exploit. Had the web-servers in question not been patched, they would certainly have been completely compromised by this automated attack.

But web-servers are not the only type of system this automated tool is seeking to exploit. We’ve seen that it also probes tcp ports 21 and 57. Can we learn anything further about what its intentions might be toward any FTP or MTP servers that it might encounter in its automated travels?

```
tcpdump -r capturefile -X -n host 202.136.1 89.51 and port 21 > ftpfile.txt
```

yields the following sobering exchange:

```
06:27:52.088223 IP xxx.yyy.zzz.139.21 > 202.136.189.51.3523: P 1:55(54) ack 1 win 5840 (DF)
0x0000  4500 005e Oce1 4000 4006 eaf1 d151 ea8b  E..^..@.@....Q..
0x0010  ca88 bd33 0015 0dc3 a92c bb11 3743 08cb ...3........7C..
0x0020  5018 16d0 df3c 0000 3232 3020 6761 6c61 P...<.220.gala
driel.FTP.server
0x0030  2028 5665 7273 6e6f 7269 656c 2028 5665 7273 6f72 6420 6173 2070 6173 776f72 642e 0d0a .(Version.wu-mail.address.as...password... 
0x0050  2079 6f75 7220 636f 6d70 6c65 7465 2065 -your.complete.e
0x0060  2d6d 6169 6c20 6164 6472 6573 7320 6173 -mail.address.as
0x0070  2070 6173 776f72 642e 0d0a .password...
0x0000  4500 005e Oce1 4000 4006 eaf1 d151 ea8b  E..^..@.@....Q..
0x0010  ca88 bd33 0015 0dc3 a92c bb11 3743 08cb ...3........7C..
0x0020  5018 16d0 df3c 0000 3232 3020 6761 6c61 P...<.220.gala
driel.FTP.server
0x0030  2028 5665 7273 6e6f 7269 656c 2028 5665 7273 6f72 6420 6173 2070 6173 776f72 642e 0d0a .(Version.wu-mail.address.as...password... 
0x0050  2079 6f75 7220 636f 6d70 6c65 7465 2065 -your.complete.e
0x0060  2d6d 6169 6c20 6164 6472 6573 7320 6173 -mail.address.as
0x0070  2070 6173 776f72 642e 0d0a .password...
0x0000  4500 005e Oce1 4000 4006 eaf1 d151 ea8b  E..^..@.@....Q..
0x0010  ca88 bd33 0015 0dc3 a92c bb11 3743 08cb ...3........7C..
0x0020  5018 16d0 df3c 0000 3232 3020 6761 6c61 P...<.220.gala
driel.FTP.server
0x0030  2028 5665 7273 6e6f 7269 656c 2028 5665 7273 6f72 6420 6173 2070 6173 776f72 642e 0d0a .(Version.wu-mail.address.as...password... 
0x0050  2079 6f75 7220 636f 6d70 6c65 7465 2065 -your.complete.e
0x0060  2d6d 6169 6c20 6164 6472 6573 7320 6173 -mail.address.as
0x0070  2070 6173 776f72 642e 0d0a .password...
```

Whoa! The attack tool has logged onto my DMZ Linux server via anonymous FTP!! Not good! Let’s see what it did: (commands sent by the tool are in bold in the ascii dump on the right)

```
06:27:52.088223 IP xxx.yyy.zzz.139.21 > 202.136.189.51.3523: P 1:55(54) ack 1 win 5840
0x0000  4500 005e Oce1 4000 4006 eaf1 d151 ea8b  E..^..@.@....Q..
0x0010  ca88 bd33 0015 0dc3 a92c bb11 3743 08cb ...3........7C..
0x0020  5018 16d0 df3c 0000 3232 3020 6761 6c61 P...<.220.gala
driel.FTP.server
0x0030  2028 5665 7273 6e6f 7269 656c 2028 5665 7273 6f72 6420 6173 2070 6173 776f72 642e 0d0a .(Version.wu-mail.address.as...password... 
0x0050  2079 6f75 7220 636f 6d70 6c65 7465 2065 -your.complete.e
0x0060  2d6d 6169 6c20 6164 6472 6573 7320 6173 -mail.address.as
0x0070  2070 6173 776f72 642e 0d0a .password...
```

06:27:52.099651 IP xxx.yyy.zzz.139.21 > 202.136.189.51.3523: P 1:55(54) ack 1 win 5840
0x0000  4500 005e Oce1 4000 4006 eaf1 d151 ea8b  E..^..@.@....Q..
0x0010  ca88 bd33 0015 0dc3 a92c bb11 3743 08cb ...3........7C..
0x0020  5018 16d0 df3c 0000 3232 3020 6761 6c61 P...<.220.gala
driel.FTP.server
0x0030  2028 5665 7273 6e6f 7269 656c 2028 5665 7273 6f72 6420 6173 2070 6173 776f72 642e 0d0a .(Version.wu-mail.address.as...password... 
0x0050  2079 6f75 7220 636f 6d70 6c65 7465 2065 -your.complete.e
0x0060  2d6d 6169 6c20 6164 6472 6573 7320 6173 -mail.address.as
0x0070  2070 6173 776f72 642e 0d0a .password...
```

Whoa! The attack tool has logged onto my DMZ Linux server via anonymous FTP!! Not good! Let’s see what it did: (commands sent by the tool are in bold in the ascii dump on the right)

```
06:27:52.099651 IP xxx.yyy.zzz.139.21 > 202.136.189.51.3523: P 1:55(54) ack 1 win 5840
0x0000  4500 005e Oce1 4000 4006 eaf1 d151 ea8b  E..^..@.@....Q..
0x0010  ca88 bd33 0015 0dc3 a92c bb11 3743 08cb ...3........7C..
0x0020  5018 16d0 df3c 0000 3232 3020 6761 6c61 P...<.220.gala
driel.FTP.server
0x0030  2028 5665 7273 6n...
06:27:53.035570 IP xxx.yyy.zzz.139.21 > 201.136.189.51.3523: P 171:191(20) ack 43 win 5840 (DF)
0x0000 4500 003c 0ce5 4000 4006 ea3d d151 ea8b E.<..<@@..Q..
0x0010 ca88 bd33 0015 0dc3 a92c bbbf 3743 08f5 ...3.........7C..
0x0020 5018 16d0 0f73 0000 3230 30 5479 7065 P....s.200.Type
0x0030 2073 6574 2074 6f20 492e 0d0a .set.to...
0x0000 4500 0030 3f65 4000 7106 86c9 ca88 bd33 E..0e@q....3
0x0010 d151 ea8b 0dc3 0015 3743 08f5 a92c bbbf .Q......7C..
0x0020 5018 217a c4aa 0000 5354 5255 2046 0d0a P.!......STRU.F
06:27:53.335461 IP xxx.yyy.zzz.139.21 > 201.136.189.51.3523: P 191:207(16) ack 51 win 5840 (DF)
0x0000 4500 0038 0ce6 4000 4006 ea3f d151 ea8b E..8..@.@..Q..
0x0010 ca88 bd33 0015 0dc3 a92c bbbf 3743 08fd ...3.....,..7C..
0x0020 5018 16d0 e156 0000 3230 30 5479 7065 P....V..200.MODE
0x0030 2073 6574 2074 6f2e 0d0a .set.to.Init...
That’s some pretty scary traffic to see directed at your FTP server! The tool was likely trying to upload some sort of attack tool to the various subdirectory of my Linux server, including /bin!! Luckily, the server’s permissions are set to deny uploads via FTP, or things could have turned out very much for the worse.

Correlations -
On February 4th (less than 2 months ago, as I write this), an individual named “Hoof Hearted” capbligh2001@hotmail.com reported observing the following traffic directed at his or her FTP logs in the discussion forum at:
http://cert.uni-stuttgart.de/archive/incidents/2003/02/msg00027.html:

[1] Tue 04Feb03 10:16:03 - Starting FTP Server... (Version 2.5f (32-bit))
[6] Tue 04Feb03 10:21:20 - (000001) 220 Serv-FTP-Server v2.5f for WinSock ready...
[6] Tue 04Feb03 10:21:21 - (000001) 331 User name okay, please send complete E-mail address as password.
[2] Tue 04Feb03 10:21:21 - (000001) PASS ano@ano.com
[5] Tue 04Feb03 10:21:21 - (000001) ANONYMOUS logged in, password: ANO@ANO.COM
[5] Tue 04Feb03 10:21:22 - (000001) 200 Type set to I.
Note that commands and the order in which they are executed are identical to those recorded by my tcdump capture file. I believe that this FTP log was created by the same attack tool that produced the trace routes on my network.

Also, note that both of the attacking hosts (mine and the one “Hoof Hearted” observed) are registered IP addresses of mail servers located in southeast Asia (Singapore in my case, Hong Kong in Hoof Hearted’s).

Hoof-hearted did receive one reply - (located at http://www.fiberstarr.com/pipermail/incidents_fiberstarr.com/2003-February/000381.html) to his query about this traffic, stating that it was likely just normal authorized FTP access. In light of the activities I have documented here, I find this assertion very difficult to believe. The chances of these exact commands being repeated in exactly the same sequence are negligible. Also, note the timestamps in these logs; these commands are executed with just a few tenths of a second separating them. This is decidedly an automated tool, and not a human being manually typing these commands. In addition, we have seen ample evidence of this automated tool’s true motives from the above analysis of the myriad attempted web exploits it generates.

Evidence of Active Targeting -

There is absolutely no evidence of active targeting here. If anything, there is overwhelming evidence that the attacker merely enters a range of IP addresses that he would like to investigate and/or compromise, and the tool just blindly does his or her bidding. Every IP address in my network was probed, regardless of whether there was a live host there or not, and regardless of operating system, function, or publicly available information.

Severity -

Severity = (Criticality + Lethality) – (System Countermeasures + Network Countermeasures)

Criticality: importance of the targeted system. These servers on my DMZ subnet provide email, file storage, and limited web services to me and a community of hobbyists. While no highly sensitive data is stored on any of them, it would be a significant inconvenience if they were to go down. I will assign a rating of 3 here.
Lethality: how severe is the damage if the attack succeeds? Had this automated attack tool been successful, it could have resulted in remote compromise of several servers on my DMZ. Also, these servers would likely then be utilized as attack hosts to launch malicious attacks against other networks, the damage from which I could conceivably be held liable for in a court of law. Therefore, I will assign a worst-case rating of 5 here.

System Countermeasures: Patches and hot-fixes are in very good shape, and all but one unnecessary service was turned off. Unfortunately, it was that one service (FTP, with anonymous access allowed) than very nearly enabled the attack tool to anonymously upload malware to my Linux server. I will assign a 2 here.

Network Countermeasures: Port 80 is wide open at the firewall by necessity (these are publicly-accessible web servers, after all). All IP addresses appear to be allowed access, regardless of geographic location. An IDS is obviously in place, or we would not be analyzing this detect in the first place. I will assign a rating of 2 here.

Severity = (3 + 5) – (2 + 2) = 4

Defensive Recommendations -
With regards to the anonymous FTP server, mea culpa!! I do not generally run FTP anywhere on my network at all, and had “temporarily” enabled it just a few hours prior to this attack, to facilitate the transfer of the large packet sniffer capture files between systems. Regardless, the anonymous feature should have been disabled, and I should have set the FTP parameters to allow only one simultaneous logon, and then stopped the server the instant I was done transferring the files. This is yet another example of how easily and quickly an automated tool or worm can potentially compromise a vulnerable Internet-facing system! Luckily in this case, other system countermeasures prevented the attack from succeeding, but this was definitely a close call.

On the web-servers, system countermeasures were adequate to thwart this attack, but this certainly reinforces the absolute necessity for keeping up on all the latest service packs and patches!

On the network side, as a direct result of this attack, I have decided to disallow all incoming ICMP echo requests, even those directed to my DMZ subnet. Had my DMZ hosts not responded to the initial ping requests, the automated tool would have totally skipped my IP address entirely. In that case, my hosts, packet sniffer and Snort IDS would not have even noticed anything amiss. Of course, it could well be argued that it is probably a good idea to know when someone is attacking you, whether they have any chance of success or not. If I want to know about these sorts of attacks in the future, I will need to deploy some IDS technology outside the DMZ’s external firewall.

I am also considering implementing a block list on my border router that will drop all traffic from well-documented hostile IP addresses and subnets.

Multiple Choice Question –
What is the most plausible explanation for this log file?

a) this is normal anonymous FTP usage by an authorized end-user.
b) this is an attempt to compromise an FTP server via the notorious buffer overflow in older versions of Wu-FTP.
c) this is an automated script running against an anonymous FTP server.
d) The use of the string "ANO@ANO.COM" proves this is Warez activity, as Warezers frequently use that string as their anonymous FTP password.

Correct answer: C – If you look at the extremely short time elapsed between the commands being sent, it should be clear that these commands could only have been generated by an automated tool of some kind.
Assignment 3: Analyze This!

Executive Summary –

After some deliberation, I’ve come to the conclusion that the University is confronted with three major issues with regard to its Intrusion Detection efforts:

1) There is far too much “noise” being generated by the IDS system. The majority of the alerts being generated appear to due to benign traffic, mis-configured hosts or IDS rules, and some customized alerts that do not appear to serve any useful function. This has the deleterious effect of swamping both the IDS sensors and the Analyst’s bandwidth.

2) The use of peer-to-peer networks such as Gnutella, Morpheus, and Kazaa is obviously quite widespread at the University. Rampant uncontrolled use of P2P networks is becoming much more controversial and potentially costly, as the Recording Industry Association of America has recently begun mounting major legal challenged to educational institutions which turn a blind eye to the resulting inevitable theft of copyrighted material. This uncontrolled file-sharing also constitutes a serious abuse of the University’s Internet capacity, which is quite likely to be interfering with much more legitimate bandwidth demands.

3) There are indications that several University systems may have been compromised by worms and/or external attackers. These should be examined immediately for any signs of potential compromise by the University’s InfoSec personnel.

Recommendation:
I recommend that the University embark upon the following lock-down strategy, post-haste:

1) Fine tune the IDS to ignore traffic that is generally benign, e.g. internal SMB/Netbios Wildcards, Unicode traffic directed at well known and legitimate webservers in Southeast Asia, etc.

2) Block all outbound unsolicited HTTP traffic on the standard web server ports (tcp 80, 433, 8080, etc). This will stop the University from infecting any external systems, and thereby hopefully limit any potential legal liabilities. This will not prevent the University’s legitimate webservers from doing their job, as any legitimate traffic from them will be a response to already established incoming requests.

3) Clarify the University’s policy forbidding the trafficking in copyrighted materials via P2P protocols on the University’s data networks. Set a start date for the policy, this will be the date that the main P2P protocols are blocked at the border routers and/or firewalls.

4) Seek out all internally compromised systems and rebuild them (see section entitled “Possibly Compromised Hosts”).

The immediate execution of these 4 steps will greatly enhance the security of the University’s data networks. Also, a great deal of legal liability can potentially be avoided, especially in light of the RIAA’s recent aggressive legal posture regarding P2P file-sharing of copyrighted material.
Scope of the Investigation –

The University provided me with the following data files:

Snort Alert files from 3/27/03 through 3/31/03, in “fast alert” format and gzipped:
alert.030327.gz
alert.030328.gz
alert.030329.gz
alert.030330.gz
alert.030331.gz

Snort Scan files from the same time period:
scans.030327.gz
scans.030328.gz
scans.030329.gz
scans.030330.gz
scans.030331.gz

Snort “OOS” Out Of Spec Reports from the same time period:
OOS_Report_2003_03_27_23034
OOS_Report_2003_03_28_5271
OOS_Report_2003_03_29_20502
OOS_Report_2003_03_30_15595
OOS_Report_2003_03_31_15057

Unfortunately, no other data was available, so I did not have the opportunity to corroborate my finding with full packet capture files, application or firewall logs, etc. The University should endeavor to corroborate these findings with the relevant system log files at the earliest opportunity.

“Top Talkers” -
In this section, I focus on the hosts (both internal and external) that generated the largest number of Snort alerts. These are the so called “Top Talkers” of this particular data set. By this, I mean they are the most frequently occurring source IP addresses in the population of Snort alerts.

<table>
<thead>
<tr>
<th>Internal Source IP Address</th>
<th># of Alerts</th>
<th>External Source IP Address</th>
<th># of Alerts</th>
<th>DNS Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>MY.NET.105.204</td>
<td>6561</td>
<td>68.49.35.0</td>
<td>16151</td>
<td>bgp01534079bgs.gambrl01.md.comcast.net</td>
</tr>
<tr>
<td>MY.NET.201.58</td>
<td>4408</td>
<td>212.179.48.177</td>
<td>5808</td>
<td>fw.cloverleafcomm.com</td>
</tr>
<tr>
<td>MY.NET.240.78</td>
<td>3425</td>
<td>63.148.150.226</td>
<td>5589</td>
<td>fomp.com</td>
</tr>
<tr>
<td>MY.NET.222.174</td>
<td>2298</td>
<td>194.87.6.230</td>
<td>5405</td>
<td>230.6.87.194.dynamic.dol.ru</td>
</tr>
<tr>
<td>MY.NET.242.250</td>
<td>2085</td>
<td>66.95.149.154</td>
<td>4510</td>
<td>66-95-149-154.generic.nas-inter.net</td>
</tr>
<tr>
<td>MY.NET.152.157</td>
<td>1629</td>
<td>128.8.10.18</td>
<td>3610</td>
<td>grapevine.wam.umd.edu</td>
</tr>
<tr>
<td>MY.NET.75.144</td>
<td>1038</td>
<td>212.179.43.225</td>
<td>3333</td>
<td>bzq-179-43-225.cust.bezeqint.net</td>
</tr>
</tbody>
</table>
The vast majority of the traffic generated by the “top talkers” can be classified into just a few distinct categories. By far the most common of these is Peer to Peer (P2P) file sharing activity, using such programs as Kazaa, Morpheus, eDonkey, etc. While not overtly malicious, this traffic should be of significant concern to The University. The RIAA has recently stepped up legal pursuits institutions that allow unauthorized exchange of copyrighted materials via P2P networks. These P2P software programs are also well-known vehicles for the dissemination of viruses and trojan horse programs.

Another major category of traffic observed from the “top talkers” is the frequent use of streaming media programs such as MS Media Player, Real Player, QuickTime, etc. While much of this traffic may be legitimate University traffic (video learning, online teleconferencing, etc), there is a strong probability that students are using such programs to stream on-line radio and video across the Internet. This is primarily a bandwidth utilization issue, and it is up to The University to clearly define its Acceptable Use Policy regarding such activities.

A third major category of traffic frequently seen passing between the “top talkers” is Internet Relay Chat traffic on tcp ports 6666-6667. Again, while this is not necessarily malicious in and of itself, there is significant evidence that several hosts inside The University are being used as “XDCC Bots” (see section on Compromised Hosts below), and may well be illegally serving up copyrighted materials, and making their presence known via IRC communications channels.

**Most Commonly Occurring Alerts**

The following are the most frequently occurring alerts generated by the University’s IDS system. I have listed only the top 25 most common here. It should not be inferred that the most common alerts are the ones that deserve the most attention; that is decidedly not the case. I present this table only to give a general sense of the overall status and operation of the University’s IDS.

**Table 1 –**

<table>
<thead>
<tr>
<th>Frequency</th>
<th>Snort Alert Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>406011</td>
<td>SMB Name Wildcard</td>
</tr>
<tr>
<td>65882</td>
<td>TCP SRC and DST outside network</td>
</tr>
<tr>
<td>45660</td>
<td>CS WEBSERVER - external web traffic</td>
</tr>
<tr>
<td>30109</td>
<td>Watchlist 000220 IL-ISDNNET-990517</td>
</tr>
<tr>
<td>22221</td>
<td>High port 65535 udp - possible Red Worm - traffic</td>
</tr>
<tr>
<td>20342</td>
<td>MY.NET.30.3 activity</td>
</tr>
<tr>
<td>15861</td>
<td>spp_http_decode: IIS Unicode attack detected</td>
</tr>
<tr>
<td>14801</td>
<td>External RFC call</td>
</tr>
<tr>
<td>11978</td>
<td>Russia Dynamo - SANS Flash 28-jul-00</td>
</tr>
<tr>
<td>8128</td>
<td>SUNRPC highport access!</td>
</tr>
<tr>
<td>6057</td>
<td>MY.NET.30.4 activity</td>
</tr>
</tbody>
</table>
“SMB Name Wildcard” -
The SMB Name Wildcard alert is caused by the normal behavior of Windows networking clients and server doing their routine WINS name resolutions that they do so frequently. Numerous GCIA analysts (most recently and notably Todd Beardsley & Les Gordon) have noted that these alerts are generally benign if caused by internal traffic, which the vast majority of these are.

Provided that all port 135-139 traffic is adequately blocked at all perimeter devices (both tcp and udp!), this alert should be deactivated to save wear and tear on the IDS and its analysts. Alternatively, in the very unusual event that some Netbios traffic is allowed to cross the network perimeter, this rule can be modified to fire only upon Netbios Wildcard traffic that crosses the network perimeter boundary.

Customized Watch Lists -
Several of these are alert signatures have been pre-defined in the IDS to watch for traffic to or from certain hosts. These are:

<table>
<thead>
<tr>
<th>Watchlist</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>CS WEBSERVER - external web traffic</td>
<td></td>
</tr>
<tr>
<td>CS WEBSERVER - external ftp traffic</td>
<td></td>
</tr>
<tr>
<td>Watchlist 000220 IL-ISDNNET-990517</td>
<td></td>
</tr>
<tr>
<td>Watchlist 000222 NET-NCFC</td>
<td></td>
</tr>
<tr>
<td>MY.NET.30.3 activity</td>
<td></td>
</tr>
<tr>
<td>MY.NET.30.4 activity</td>
<td></td>
</tr>
<tr>
<td>Russia Dynamo - SANS Flash 28-jul-00</td>
<td></td>
</tr>
</tbody>
</table>

These hosts are generating a tremendous number of alerts (by definition, any traffic directed to or from these watchlists will generate an alert”. Also, the majority of the Much of the traffic being generated by these hosts is Peer to Peer file sharing and streaming media traffic. It appears that these hosts are generating very large quantities of IDS alerts, with very little real malicious activity behind them.

I would recommend dispensing with the watchlists, and focus on identifying attacks via the effective use of an up-to-date Snort database. If there are certain subnets or regions the University does not need to exchange traffic with, a more effective strategy may be to block all traffic to our from those subnets altogether at the firewall or border router.

Possibly Compromised Hosts –
Even from the limited data available in files provided, there is significant evidence that several hosts on the University’s internal network have been compromised by worms and/or external
attackers. We see a significant number of attack attempts originating on the internal network and directed at hosts on the Internet. Also, there are significant indications that some of these same systems are being utilized as attack hosts to perform scanning against other external networks.

I will now examine the hosts showing the most significant evidence of prior compromise:

**“XDCC Bots” –**
- 130.85.151.107
- 130.85.237.6
- 130.85.80.209
- 130.85.253.42
- 130.85.195.5
- 130.85.150.179
- 130.85.203.234
- 130.85.87.87
- 130.85.234.54
- 130.85.227.202
- 130.85.122.106
- 130.85.86.33

All of these hosts were responsible for the generation of at least one “IRC evil - running XDCC” alert. An XDCC “bot” is an automated process running on a compromised system that is generally used to make pirated movies, MP3s and other copyrighted material available for download via Internet Relay Chat (IRC channels). Recently, American universities have become a prime target for these “bots,” because they tend to have extremely fast Internet connections, coupled with relatively lenient network perimeter defenses. This has prompted the Recording Industry Association of America to go on a major legal offensive against some of the leading offenders. This could potentially become an area of significant legal liability for the University.

These alerts are very unlikely to be false positives, as the alert is keyed of a very specific text string in the payload of the packet. Also, manual verification of the destination IP addresses has confirmed them to be IRC servers well-known for the hosting of XDCC bot activity (see “Significant External Hosts” section for details).

**Possible Adore “Red Worm” & use as a scanning/attack host -**
- 130.85.240.78

High port 65535 tcp – possible Red Worm – traffic
TFTP – External TCP connection to internal tftp server

This host has been the source of an extremely large number of Snort “Tiny Fragment” alerts. These types of alerts are frequently caused maliciously crafted tcp fragments fabricated for the express purpose of evading firewalls and IDS systems. This host should be examined for signs of possible compromise, it is possible that it has been taken over and is being used as an “attack host”.

© SANS Institute 2003, As part of GIAC practical repository. Author retains full rights.
**Significant External Hosts:**
Several external hosts warrant more detailed scrutiny. These include the master controllers of compromised systems on the University Network, the IRC servers that control the XDCC “bots,” and a few other external hosts which consume a relatively large fraction of the University’s network bandwidth.

**Evil IRC Server #1 at 65.57.64.224**
This IRC server is responsible for a great deal of abuse. It is the control center for several of the XDCC bot parasites that are infesting the University’s network. Why Dshield has not yet seen fit to send a “fightback” notice to Level3 Communications is beyond me! Here is the detailed result of a query on this IP address at [www.Dshield.org](http://www.Dshield.org). Note there are over 72,000 prior attacks logged against 21 unique targets!

IP Address: 65.57.64.224
HostName: irc-4.aniverse.com
DShield Profile: Country:
Contact E-mail:
Total Records against IP: 72707
Number of targets: 21
Date Range: 2003-01-11 to 2003-03-29

Ports Attacked:

<table>
<thead>
<tr>
<th>Port</th>
<th>Attacks</th>
<th>Start</th>
<th>End</th>
</tr>
</thead>
<tbody>
<tr>
<td>23</td>
<td></td>
<td>2003-03-05</td>
<td>2003-03-31</td>
</tr>
<tr>
<td>1080</td>
<td></td>
<td>2003-03-05</td>
<td>2003-03-31</td>
</tr>
<tr>
<td>6588</td>
<td></td>
<td>2003-03-05</td>
<td>2003-03-31</td>
</tr>
<tr>
<td>8080</td>
<td></td>
<td>2003-03-05</td>
<td>2003-03-31</td>
</tr>
<tr>
<td>81</td>
<td></td>
<td>2003-03-05</td>
<td>2003-03-30</td>
</tr>
<tr>
<td>8000</td>
<td></td>
<td>2003-03-05</td>
<td>2003-03-30</td>
</tr>
<tr>
<td>8001</td>
<td></td>
<td>2003-03-05</td>
<td>2003-03-30</td>
</tr>
<tr>
<td>8081</td>
<td></td>
<td>2003-03-05</td>
<td>2003-03-30</td>
</tr>
<tr>
<td>3128</td>
<td></td>
<td>2003-03-05</td>
<td>2003-03-31</td>
</tr>
<tr>
<td>8520</td>
<td></td>
<td>2003-03-05</td>
<td>2003-03-30</td>
</tr>
</tbody>
</table>

OrgName: Level 3 Communications, Inc.
OrgID: LVLT
Address: 1025 Eldorado Blvd.
City: Broomfield
StateProv: CO
PostalCode: 80021
Country: US
NetRange: 65.56.0.0 - 65.59.255.255
CIDR: 65.56.0.0/14
NetName: LC-ORG-ARIN-BLK2
NetHandle: NET-65-56-0-0-1
Parent: NET-65-0-0-0-0
NetType: Direct Allocation
NameServer: NS1.LEVEL3.NET
NameServer: NS2.LEVEL3.NET
RegDate: 2001-09-21
Updated: 2002-08-08
TechPhone: +1-877-453-8353
TechEmail: ipaddressing@level3.com
OrgAbuseHandle: APL8-ARIN
OrgAbuseName: Abuse POC LVLT
OrgAbusePhone: +1-877-453-8353
OrgAbuseEmail: abuse@level3.com

**Evil IRC Server #2 at 205.188.149.12** –  
This IRC Server at AOL.com is also controlling some of the XDCC bot activity at the University:  
IP Address: 205.188.149.12  
HostName: undernet.irc.aol.com  
DShield Profile: Country: US  
Contact E-mail: TOSUsenet@aol.com  
Total Records against IP: 1197  
Number of targets: 319  
Date Range: 2003-01-12 to 2003-03-27  
Ports Attacked (up to 10):  
<table>
<thead>
<tr>
<th>Port</th>
<th>Attacks</th>
<th>Start</th>
<th>End</th>
</tr>
</thead>
<tbody>
<tr>
<td>2793</td>
<td>388</td>
<td>2003-03-21</td>
<td>2003-03-28</td>
</tr>
<tr>
<td>3476</td>
<td>116</td>
<td>2003-03-20</td>
<td>2003-03-22</td>
</tr>
<tr>
<td>113</td>
<td>83</td>
<td>2003-03-07</td>
<td>2003-04-02</td>
</tr>
<tr>
<td>1693</td>
<td>75</td>
<td>2003-03-24</td>
<td>2003-03-31</td>
</tr>
<tr>
<td>4266</td>
<td>61</td>
<td>2003-03-18</td>
<td>2003-03-19</td>
</tr>
<tr>
<td>1352</td>
<td>27</td>
<td>2003-03-19</td>
<td>2003-03-19</td>
</tr>
<tr>
<td>1722</td>
<td>27</td>
<td>2003-03-30</td>
<td>2003-03-31</td>
</tr>
<tr>
<td>2433</td>
<td>22</td>
<td>2003-03-09</td>
<td>2003-03-09</td>
</tr>
<tr>
<td>1281</td>
<td>18</td>
<td>2003-03-28</td>
<td>2003-03-29</td>
</tr>
<tr>
<td>2839</td>
<td>16</td>
<td>2003-03-17</td>
<td>2003-03-18</td>
</tr>
</tbody>
</table>

Fightback: not sent  
OrgName: America Online, Inc  
OrgID: AMERIC-59  
Address: 22080 Pacific Blvd  
City: Sterling  
StateProv: VA  
PostalCode: 20166  
Country: US  
NetRange: 205.188.0.0 - 205.188.255.255  
CIDR: 205.188.0.0/16  
NetName: AOL-DTC  
NetHandle: NET-205-188-0-0-1  
Parent: NET-205-0-0-0-0  
NetType: Direct Assignment  
NameServer: DNS-01.NS.AOL.COM  
NameServer: DNS-02.NS.AOL.COM  
RegDate: 1998-04-18  
Updated: 1998-04-27  
TechHandle: AOL-NOC-ARIN  
TechName: America Online, Inc.  
TechPhone: +1-703-265-4670  
TechEmail: domains@aol.net

**Evil IRC Server #3 at 193.163.220.3** –  
Yet another known “evil” IRC server controlling some of the XDCC bot traffic, this one is in Denmark:  
IP Address: 193.163.220.3  
HostName: irc.inet.tele.dk  
DShield Profile: Country:  
Contact E-mail:  
Total Records against IP: 21  
Number of targets: 5
Date Range: 2003-01-16 to 2003-02-26
Ports Attacked (up to 10):
<table>
<thead>
<tr>
<th>Port</th>
<th>Attacks</th>
<th>Start</th>
<th>End</th>
</tr>
</thead>
<tbody>
<tr>
<td>113</td>
<td>40</td>
<td>2003-03-11</td>
<td>2003-03-31</td>
</tr>
<tr>
<td>1433</td>
<td>7</td>
<td>2003-03-17</td>
<td>2003-03-21</td>
</tr>
</tbody>
</table>
Fightback: not sent

inetnum: 193.163.220.0 - 193.163.220.7
netname: JAN-CHRILLESEN-NET
descr: IRC services
country: DK
admin-c: JC301-RIPE
tech-c: JC301-RIPE
status: ASSIGNED PI
mnt-by: TDINET-MNT
remarks: For abuse and security issues contact
remarks: abuse@eris.dk
remarks: -------------------------------------
changed: jan@chrillesen.dk 20030204
source: RIPE
route: 193.163.220.0/24
descr: Jan Chrillesen
origin: AS3292
mnt-by: AS3292-MNT
changed: auto-ripe@ip.tele.dk 20000629
changed: auto-ripe@ip.tele.dk 20010311
changed: auto-ripe@ip.tele.dk 20020730
address: Hougaardsvej 48, 2.th
address: DK-8220 Brabrand
address: Denmark
phone: +45 87 47 00 01
fax-no: +45 87 47 00 02
nic-hdl: JC301-RIPE
changed: hostmaster@DK.net 19970212

**FastSearch Web Bot Server at 66.77.73.144**

This is the exact same Web Bot activity that I analyzed in Assignment 2, section 2 of this document. Here, it is doing a content update of the University’s Internet-facing web server at IP address 130.85.100.165.

IP Address: 66.77.73.144
HostName: cr005r01.sac2.fastsearch.net
DSHield Profile: Country: US
Contact E-mail: ip-admin@gis.qwest.net
Total Records against IP: 23
Number of targets: 3
Date Range: 2003-01-14 to 2003-02-03
Ports Attacked (up to 10):
<table>
<thead>
<tr>
<th>Port</th>
<th>Attacks</th>
<th>Start</th>
<th>End</th>
</tr>
</thead>
<tbody>
<tr>
<td>80</td>
<td>35</td>
<td>2003-03-10</td>
<td>2003-03-26</td>
</tr>
</tbody>
</table>
CustName: Fast Search, Inc.
Address: 93 Worcester Street, 4th Floor
City: Wellesley
StateProv: MA
PostalCode: 02481
Country: US
RegDate: 2002-01-10
Updated: 2002-01-10
NetRange: 66.77.73.0 - 66.77.73.255
CIDR: 66.77.73.0/24
Notable False Positives:
130.85.106.92
130.85.106.108
130.85.104.117
130.85.152.45

I’ve grouped these four hosts together because they all show a very similar set of symptoms. All of them are sending out large amounts of “IIS Unicode attack” traffic. Virtually all of this traffic is directed at one specific host (210.219.197.11) in South Korea. Here is the IP registration information for this host’s network. There are no prior complaints against this address in Dshield:

<table>
<thead>
<tr>
<th>IP Address</th>
<th>210.219.197.0-210.219.197.127</th>
</tr>
</thead>
<tbody>
<tr>
<td>Connect ISP Name</td>
<td>ELIMNET</td>
</tr>
<tr>
<td>Connect Date</td>
<td>19990510</td>
</tr>
<tr>
<td>Registration Date</td>
<td>200001004</td>
</tr>
<tr>
<td>Network Name</td>
<td>DONGA-LACADEMY</td>
</tr>
</tbody>
</table>

[ Admin Contact Information]
Name: HONGSOON CHOI
State: SEOUL
Address: #601 SEOJIN PLAZA, 403-1 SANG1 DONG, WONMI GU, PUCHEN SI
Zip Code: 420-031
Phone: +82-32-666-3301
E-Mail: domain@elim.net

The “IIS Unicode” attack is frequently associated with the Nimda worm, which relies heavily on the Unicode decoding flaw in the IIS web-server (as well as several other IIS vulnerabilities) to do its dirty work. However, after some investigation, I do NOT believe this host to be infected with the Nimda worm. With Nimda, we would expect to see other signatures being triggered, such as the “Remote CMD” execution and the “Double-Decode” alerts. While it is possible that these alerts have been suppressed in the IDS system, another factor that leads me to conclude that this is not a Nimda-infected host has to do with the target selection. Nimda will typically target several different hosts in a somewhat random manner (somewhat, in that it prefers to target subnets that are relatively “close” to the infected host in IP space). This host is not doing this at all. Rather, it appears to be limiting its attacks to a very small number of target hosts, which it is very rigorously probing repeatedly with Unicode attacks.

Numerous GCIA candidates have speculated upon the underlying cause of these Unicode alerts. Carlin Carpenter also noticed the preponderance of traffic involved sites in SE Asia, and speculated that it might have something to do with the international hacking incidents (prompted by the US Spyplane incident) which was occurring at the time. However, we now see that this Unicode traffic appears to be a permanent fixture in the University’s portfolio of IDS alerts.

A more fitting explanation is presented by Les Gordon in his December 31st, 2002 practical, where he notes that these alerts are triggered whenever Snort encounters a Unicode encoded form of the period, comma, or forward/backslash characters. Unfortunately, these encodings are
rather common in legitimate ssl-encrypted traffic, as well in normal http communications with any web-sites that use extended character sets (including Korean).

Indeed, I had already independently reached the exact same conclusion prior to reading Mr Gordon’s excellent and very recently released practical. As I indicated in the first submission of this paper, I believe, quite simply, that the vast majority of these Unicode alerts are being caused by Korean-speaking students and/or faculty legitimately surfing Korean-language web-servers. Snort’s unicode attack signature may well be misinterpreting the Unicode encoding of the Korean language characters as Unicode attacks. This would certainly explain why so many of these alerts are consistently directed at a very small handful of web servers in South Korea. Indeed, of the nearly “4,00 IIS-Unicode Attack” alerts generated by Snort in the 3-31-03 alerts file, 855 were generated by four client systems communicating with exactly ONE web server owned by ELIMNET in South Korea (210.219.197.11). No other alerts were detected by any of these systems, and the web-server comes up clean on Dshield.Org. The lack of any other suspicious traffic from any of these systems corroborates the theory that these this is valid web-surfing traffic setting off false-positive Unicode alerts due to the prevalence of unicode encoding in the extended character sets used to represent Korean language characters.

A link graph depicting the alert frequency of the Unicode alerts for these top four hosts should help make the argument more convincing:
If it does indeed turn out these alerts are being generated by legitimate web surfing, the University should attempt to tighten up the firing syntax for the “Unicode Attack” Snort rule. Perhaps unicode traffic directed toward well-known bona fide Korean language web servers could be excluded from scrutiny. Alternatively, the unicode detect could be dispensed with altogether, as there is ample correlation indicating its propensity to false positives. Most unicode-based attacks possess other distinguishing signatures we can “netgrep” for, as long as these rules are in place, maybe the Unicode rule is simply not worth its false positive baggage! A rule that constantly fires on legitimate (and frequent!) traffic is worse than no rule at all, as it
invariably has the perverse effect of instilling complacency on the part of the University’s InfoSec staff who monitor the IDS.

**Scanning Activity Overview:**

Nearly 3.2 million scans were detected by the University’s IDS during the 5 day period from 3/27/03 through 3/31/03. The following table shows the daily breakdown of the scans that were detected. Note the overwhelming tendency for the scans to occur during the weekend hours (3/29/03 was a Saturday).

**Daily Distribution of Scans Reported by the University’s IDS**  
(3/27/03 – 3/31/03)

<table>
<thead>
<tr>
<th></th>
<th>Number of Scans Detected</th>
<th>Type of Scan</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1128025</td>
<td>3/29/03</td>
</tr>
<tr>
<td>2</td>
<td>1252924</td>
<td>3/30/03</td>
</tr>
<tr>
<td>3</td>
<td>5991</td>
<td>3/27/03</td>
</tr>
<tr>
<td>4</td>
<td>1606</td>
<td>3/28/03</td>
</tr>
<tr>
<td>5</td>
<td>896</td>
<td>3/31/03</td>
</tr>
</tbody>
</table>

The vast majority of these scans were TCP SYN and UDP scans. In general, the bulk of these scans are likely to be generated by automated port-scanning tools such as nmap.

**Scan Types Reported by the University’s IDS**  
(3/27/03 – 3/31/03)

<table>
<thead>
<tr>
<th></th>
<th>Number of Scans Detected</th>
<th>Type of Scan</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2014622</td>
<td>SYN scan</td>
</tr>
<tr>
<td>2</td>
<td>1252924</td>
<td>UDP scan</td>
</tr>
<tr>
<td>3</td>
<td>5991</td>
<td>FIN scan</td>
</tr>
<tr>
<td>4</td>
<td>1606</td>
<td>NULL scan</td>
</tr>
<tr>
<td>5</td>
<td>896</td>
<td>NOACK scan</td>
</tr>
<tr>
<td>6</td>
<td>586</td>
<td>VECNA scan</td>
</tr>
<tr>
<td>7</td>
<td>328</td>
<td>INVALIDACK scan</td>
</tr>
<tr>
<td>8</td>
<td>217</td>
<td>UNKNOWN scan</td>
</tr>
<tr>
<td>9</td>
<td>45</td>
<td>XMAS scan</td>
</tr>
<tr>
<td>10</td>
<td>29</td>
<td>FULLXMAS scan</td>
</tr>
</tbody>
</table>

The following table lists the ten most active “scanning” source IP addresses, much as our “Top Talkers” listed the hosts that generated the most Snort alerts. Note that eight of the top ten scanning source addresses are local to the University’s internal network. While port-scanning is not inherently illegal, it is frequently a leading indicator that the host’s owner may be enumerating potential targets for future nefarious activities. Therefore, these hosts should be scrutinized on an ongoing basis for signs of compromise and/or malicious hacking by the users who control them.
Top Ten “Scanners”  
(3/27/03 – 3/31/03)

<table>
<thead>
<tr>
<th>Number of Scans Detected</th>
<th>IP Address of Scanning Host</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>226065 130.85.97.43</td>
</tr>
<tr>
<td>2</td>
<td>118942 130.85.1.3</td>
</tr>
<tr>
<td>3</td>
<td>116768 130.85.195.155</td>
</tr>
<tr>
<td>4</td>
<td>59656 130.85.217.150</td>
</tr>
<tr>
<td>5</td>
<td>59488 130.85.235.250</td>
</tr>
<tr>
<td>6</td>
<td>47786 130.85.97.53</td>
</tr>
<tr>
<td>7</td>
<td>42993 211.58.53.253</td>
</tr>
<tr>
<td>8</td>
<td>36080 217.215.65.6</td>
</tr>
<tr>
<td>9</td>
<td>33889 130.85.228.30</td>
</tr>
<tr>
<td>10</td>
<td>32734 130.85.217.50</td>
</tr>
</tbody>
</table>

Summary –

The University is confronted with three major issues with regard to its Intrusion Detection efforts:

1) There is far too much “noise” being generated by the IDS system. The majority of the alerts being generated appear to due to benign traffic, mis-configured hosts or IDS rules, as well as some customized alerts that do not appear to serve any useful function. This has the deleterious effect of swamping both the IDS sensors and the Analyst’s bandwidth.

2) The use of peer-to-peer networks such as Gnutella, Morpheus, and Kazaa is obviously quite widespread at the University. Rampant uncontrolled use of P2P networks is becoming much more controversial and potentially costly, as the Recording Industry Association of America has recently begun mounting major legal challenged to educational institutions which turn a blind eye to the resulting inevitable theft of copyrighted material. This uncontrolled file-sharing also constitutes a serious abuse of the University’s Internet capacity, which is quite likely to be interfering with much more legitimate bandwidth demands.

3) There are indications that several University systems may have been compromised by worms and/or external attackers. These should be examined immediately for any signs of potential compromise by the University’s InfoSec personnel.

Defensive Recommendations:
I recommend that the University embark upon the following lock-down strategy, post-haste:

5) Fine tune the IDS to ignore traffic that is generally benign, e.g. internal SMB/Netbios Wildcards, Unicode traffic directed at well known and legitimate webservers in Southeast Asia, etc.
6) Block all outbound **unsolicited** HTTP traffic on the standard web server ports (tcp 80, 433, 8080, etc). This will stop the University from infecting any external systems, and thereby hopefully limit any potential legal liabilities. This will **not** prevent the University’s legitimate webservers from doing their job, as any legitimate traffic from them will be a response to already established incoming requests.

7) Clarify the University’s policy forbidding the trafficking in copyrighted materials via P2P protocols on the University’s data networks. Set a start date for the policy, this will be the date that the main P2P protocols are blocked at the border routers and/or firewalls.

8) Seek out all internally compromised systems and rebuild them (see section entitled “Possibly Compromised Hosts”).

The immediate execution of these 4 steps will greatly enhance the security of the University’s data networks. Also, a great deal of legal liability can potentially be avoided, especially in light of the RIAA’s recent aggressive legal posture regarding P2P file-sharing of copyrighted material.

**Methodology used for Analyze This!**

At first I attempted to run the SnortSnarf Perl on the entire dataset. Unfortunately, as was the case with several worthy analysts before me, I encountered difficulties with the use of non-numeric data to designate the home network Class B address “MY.NET”, as well as a general lack of sufficient RAM and CPU resources to get the job down in a timely fashion. A tip for those who follow: having massive amounts of RAM is evidently the key, otherwise your system is like end up thrashing for days.

I was, however, successful in processing a couple of individual days worth of the Alerts data with SnortSnarf, and found the tool to be incredibly useful in terms of the holistic view it provided into the relationships amongst the various hosts. I used these smaller SnortSnarf datasets to extensively track down specific relationships between various source and destination hosts, as well as to give a general “reality check” to the results I obtained via other means (see below).

As a fallback, I borrowed some tricks and scripts from the excellent previous practicals by Thomas Beardsley and Lorraine Weaver. Lorraine clued me in to a very simple and efficient way to concatenate all the data files together:

```
cat alert.1 > alerts
cat alert.2 >> alerts
cat alert.3 >> alerts
cat alert.4 >> alerts
cat alert.5 >> alerts
```

Which I translated into command-line Windows as:

```
Copy alert.1+alert.2+alert.3+alert.4+alert.5 alerts.all
```

This process was repeated with the Scan and the OOS files.
Once all files are concatenated (a surprisingly quick and painless process, using this method), I was able to use Thomas Beardsley excellent Perl scripts: csv.pl and summarize.pl to process and summarize the data files.

In general, my approach from this point on was identify the Top Talkers in terms of alerts and scans, then look them up in the individual day’s SnortSnarf data I had processed. Once you have the alert or host of interest identified, SnortSnarf provides extremely convenient hyperlinks to relevant port information, source and destination IP addresses, Dshield lookups, etc.

References for Sections 2 & 3:


Coochey, Giles. “WEB-IIS cmd.exe access.” Snort Signatures Database. URL: http://www.snort.org/snort-db/sid.html?id=1002


FAST Seach, Inc. Description of Automated Web Indexing Bot. URL: http://www.fastsearch.net/products/websearch/index.asp


Internet Assigned Numbers Authority [IANA]. “Port Numbers.” URL: http://www.iana.org/assignments/port-numbers


Silicon Defense, Inc. “SnortSnarf” Perl Scripts. URL:


## Upcoming Training

<table>
<thead>
<tr>
<th>Event</th>
<th>Location</th>
<th>Dates</th>
<th>Format</th>
</tr>
</thead>
<tbody>
<tr>
<td>SANS Blue Team and Purple Team 2020</td>
<td>United Arab Emirates</td>
<td>Jul 13, 2020 - Jul 18, 2020</td>
<td>CyberCon</td>
</tr>
<tr>
<td>SANS Rocky Mountain Summer 2020</td>
<td>CO</td>
<td>Jul 20, 2020 - Jul 25, 2020</td>
<td>CyberCon</td>
</tr>
<tr>
<td>Live Online - SEC503: Intrusion Detection In-Depth</td>
<td>United Arab Emirates</td>
<td>Jul 29, 2020 - Sep 04, 2020</td>
<td>vLive</td>
</tr>
<tr>
<td>Instructor-Led Training</td>
<td>Aug 3 ET</td>
<td>Aug 03, 2020 - Aug 08, 2020</td>
<td>CyberCon</td>
</tr>
<tr>
<td>SANS vLive - SEC503: Intrusion Detection In-Depth</td>
<td>SEC503 - 202008,</td>
<td>Aug 10, 2020 - Sep 16, 2020</td>
<td>vLive</td>
</tr>
<tr>
<td>SANS Essentials Live Online 2020</td>
<td>Australia</td>
<td>Aug 17, 2020 - Aug 22, 2020</td>
<td>CyberCon</td>
</tr>
<tr>
<td>SANS Baltimore Fall 2020</td>
<td>Baltimore, MD</td>
<td>Sep 08, 2020 - Sep 13, 2020</td>
<td>Live Event</td>
</tr>
<tr>
<td>SANS Baltimore Fall 2020 - Live Online</td>
<td>Baltimore, MD</td>
<td>Sep 08, 2020 - Sep 13, 2020</td>
<td>CyberCon</td>
</tr>
<tr>
<td>SANS Network Security 2020 - Live Online</td>
<td>Las Vegas, NV</td>
<td>Sep 20, 2020 - Sep 25, 2020</td>
<td>CyberCon</td>
</tr>
<tr>
<td>SANS London October 2020 - Live Online</td>
<td>London, United Kingdom</td>
<td>Oct 12, 2020 - Oct 17, 2020</td>
<td>CyberCon</td>
</tr>
<tr>
<td>SANS October Singapore 2020</td>
<td>Singapore, Singapore</td>
<td>Oct 12, 2020 - Oct 24, 2020</td>
<td>CyberCon</td>
</tr>
<tr>
<td>SANS October Singapore 2020 - Live Online</td>
<td>Singapore, Singapore</td>
<td>Oct 12, 2020 - Oct 24, 2020</td>
<td>CyberCon</td>
</tr>
<tr>
<td>SANS Dallas Fall 2020</td>
<td>Dallas, TX</td>
<td>Oct 19, 2020 - Oct 24, 2020</td>
<td>Live Event</td>
</tr>
<tr>
<td>SANS Dallas Fall 2020 - Live Online</td>
<td>Dallas, TX</td>
<td>Oct 19, 2020 - Oct 24, 2020</td>
<td>CyberCon</td>
</tr>
<tr>
<td>SANS London November 2020</td>
<td>London, United Kingdom</td>
<td>Nov 02, 2020 - Nov 07, 2020</td>
<td>Live Event</td>
</tr>
<tr>
<td>SANS London November 2020 - Live Online</td>
<td>London, United Kingdom</td>
<td>Nov 02, 2020 - Nov 07, 2020</td>
<td>CyberCon</td>
</tr>
<tr>
<td>SANS San Diego Fall 2020</td>
<td>San Diego, CA</td>
<td>Nov 16, 2020 - Nov 21, 2020</td>
<td>Live Event</td>
</tr>
<tr>
<td>SANS San Diego Fall 2020 - Live Online</td>
<td>San Diego, CA</td>
<td>Nov 16, 2020 - Nov 21, 2020</td>
<td>CyberCon</td>
</tr>
<tr>
<td>SANS Cyber Defense Initiative 2020 - Live Online</td>
<td>Washington, DC</td>
<td>Dec 12, 2020 - Dec 19, 2020</td>
<td>CyberCon</td>
</tr>
<tr>
<td>SANS OnDemand</td>
<td>Online</td>
<td>Anytime</td>
<td>Self Paced</td>
</tr>
<tr>
<td>SANS SelfStudy</td>
<td>Books &amp; MP3s Only</td>
<td>Anytime</td>
<td>Self Paced</td>
</tr>
</tbody>
</table>