GIAC
CERTIFICATIONS

Global Information Assurance Certification Paper

Copyright SANS Institute
Author Retains Full Rights

This paper is taken from the GIAC directory of certified professionals. Reposting is not permited without express written permission.

Interested in learning more?

Check out the list of upcoming events offering
"Network Monitoring and Threat Detection In-Depth (Security 503)"
at http://www.giac.org/registration/gcia

*** Northcutt, ESL grading, Javier really helped out with the color coding. The TCPdump filters were also a really nice touch. 89 *

SANS PRACTICAL EXAM

For GIAC - Intrusion Detection Analyst

Author **:** Javier Romero
Date **:** April 24, 2000

© SANS Institute 2000 - 2002, Author retains full rights.

© SANS Institute 2000 - 2002 As part of GIAC practical repository. Author retains full rights.

HOW TO START THIS ANALYSIS

Then of SANS2000, I redesigned the security architecture of the network, that offer Internet services to the world: DNS, Hosting and Housing (WWW, FTP, IRC, etc), Media Services, Mail (imap, smtp, pop3).

Steps:

1. A project to build real security architecture was designed. With 2 phases, the first to develop the infrastructure, policies, architecture, flow of communications, security teams, and so on, with dates and people, and consultants to start final tests. The second to structure the sequence to maintain operative our security in the system (network and components).

2. Our Cisco IDS was stopped, due to false positives. And was started when a new tuning.

3. The capacity of our central switch (Cisco Catalyst) was evaluated, to employ 2 ports with port monitor option or SPAN PORT.

4. Key points to put sniffers and sensors were established. The log from the Firewall PIX was active. Each sensor ran under Solaris (UltraSPARC10 400Mhz) and Linux (Presario DL380 – Pentium III) respectively.

5. To be reviewed each log day by day.

6. Basic Filters for sensors were prepared.

7. False positives were eliminated, by checking and correcting the architecture and the firewall configuration.

8. Anomalous transactions or requests were researched with more details.

According to this, the traces or incidents showed here were found in the PIX´s log (Syslog server), then was analyzed with more details in the TCPDUMP program.

© SANS Institute 2000 - 2002, Author retains full rights.

CONTEXT:

We are a Internet Service Provider, and we use Microsoft servers to build out Data Center where you can find multiple Internet services. The end-nodes (POP to CPE) is managing for us, having the capacity to configure each end-router (CPE), to make filters, to assign IP address, and so on. Our DNS servers storage several zones from Internet (customers in special).

© SANS Institute 2000 - 2002 As part of GIAC practical repository. Author retains full rights.

© SANS Institute 2000 - 2002, Author retains full rights.

© SANS Institute 2000 - 2002 As part of GIAC practical repository. Author retains full rights.

INCIDENT 01

Severity: (3.Criticality + 4.Lethality) – (3.System + 2.Net Countermeasures)=2
Technique: TCP options and timestamp wasting all resources of the service.
Intent: Denial of Service Attack against Microsoft Media Server (Unicast Server).

MS-Streaming Media Server out of service
This server is outside from the protected network. This server doesn't have systems of logging from network level.

Here we can see connections to the port 1755. Where all connection seem inoffensive. But the streaming media sessions in this time were interrupted.

```
First Handshake
01:57:12.889571 P 10.10.10.1.1082 > x.x.x.A.1755: S 3118948512:3118948512(0) win 16060 <mss
1460,sackOK,timestamp 325246 0,nop,wscale 0> (DF)
                        4500 003c 03e6 4000 4006 c4b1 0a0a 0a01
                        xxxx xxxx 043a 06db b9e7 60a0 0000 0000
                        a002 3ebc 7b06 0000 0204 05b4 0402 080a
                        0004 f67e 0000 0000 0103 0300
01:57:12.889706 P x.x.x.A.1755 > 10.10.10.1.1082: S 977106048:977106048(0) ack 3118948513 win
17520 <mss 1460,nop,wscale 0,nop,nop,timestamp 0 0,nop,nop,sackOK> (DF)
                        4500 0040 053f 4000 8006 8354 xxxx xxxx
                        0a0a 0a01 06db 043a 3a3d 7480 b9e7 60a1
                        b012 4470 ab00 0000 0204 05b4 0103 0300
                        0101 080a 0000 0000 0000 0000 0101 0402
01:57:12.889897 P 10.10.10.1.1082 > x.x.x.A.1755: . 1:1(0) ack 1 win 16060 <nop,nop,timestamp
325246 0> (DF)
                        4500 0034 03e7 4000 4006 c4b8 0a0a 0a01
                        xxxx xxxx 043a 06db b9e7 60a1 3a3d 7481
                        8010 3ebc fafc 0000 0101 080a 0004 f67e
                        0000 0000
First data (only will show 130 bytes)
01:57:12.894896 P 10.10.10.1.1082 > x.x.x.A.1755: P 1:177(176) ack 1 win 16060
<nop,nop,timestamp 325247 0> (DF)
                        4500 00e4 03e8 4000 4006 c407 0a0a 0a01
                        xxxx xxxx 043a 06db b9e7 60a1 3a3d 7481
                        8018 3ebc 4f3a 0000 0101 080a 0004 f67f
                        0000 0000 0100 0000 cefa 0bb0 a000 0000
01:57:12.963644 P x.x.x.A.1755 > 10.10.10.1.1082: P 1:257(256) ack 177 win 17344
<nop,nop,timestamp 4869838 325247> (DF)
                        4500 0134 0540 4000 8006 825f xxxx xxxx
                        c80e f105 06db 043a 3a3d 7481 b9e7 6151
                        8018 43c0 a753 0000 0101 080a 004a 4ece
                        0004 f67f 0100 0000 cefa 0bb0 f000 0000
01:57:12.964035 P 10.10.10.1.1082 > x.x.x.A.1755: P 177:713(536) ack 257 win 15804
<nop,nop,timestamp 325254 4869838> (DF)
                        4500 024c 03e9 4000 4006 c29e 0a0a 0a01
                        xxxx xxxx 043a 06db b9e7 6151 3a3d 7581
                        8018 3dbc 2110 0000 0101 080a 0004 f686
                        004a 4ece 0100 0000 cefa 0bb0 2000 0000
01:57:12.964342 P x.x.x.A.1755 > 10.10.10.1.1082: P 257:769(512) ack 713 win 16808
<nop,nop,timestamp 4869838 325254> (DF)
                        4500 0234 0541 4000 8006 815e xxxx xxxx
                        0a0a 0a01 06db 043a 3a3d 7581 b9e7 6369
                        8018 41a8 5c0b 0000 0101 080a 004a 4ece
                        0004 f686 0100 0000 cefa 0bb0 f001 0000
01:57:12.964433 P x.x.x.A.1755 > 10.10.10.1.1082: P 769:1281(512) ack 713 win 16808
<nop,nop,timestamp 4869838 325254> (DF)
                        4500 0234 0542 4000 8006 815d xxxx xxxx
                        c80e f105 06db 043a 3a3d 7781 b9e7 6369
                        8018 41a8 414e 0000 0101 080a 004a 4ece
                        0004 f686 0100 0000 cefa 0bb0 f001 0000
01:57:12.964614 P x.x.x.A.1755 > 10.10.10.1.1082: P 1281:2337(1056) ack 713 win 16808
<nop,nop,timestamp 4869838 325254> (DF)
                        4500 0454 0543 4000 8006 7f3c xxxx xxxx
                        0a0a 0a01 06db 043a 3a3d 7981 b9e7 6369
                        8018 41a8 abe8 0000 0101 080a 004a 4ece
                        0004 f686 0100 0000 cefa 0bb0 1004 0000
```

© SANS Institute 2000 - 2002 As part of GIAC practical repository. Author retains full rights.

```
01:57:12.964831 P x.x.x.A.1755 > 10.10.10.1.1082: P 2337:2417(80) ack 713 win 16808
<nop,nop,timestamp 4869838 325254> (DF)
                        4500 0084 0544 4000 8006 830b xxxx xxxx
                        0a0a 0a01 06db 043a 3a3d 7da1 b9e7 6369
                        8018 41a8 40b1 0000 0101 080a 004a 4ece
                        0004 f686 0100 0000 cefa 0bb0 4000 0000
01:57:12.965541 P x.x.x.A.1755 > 10.10.10.1.1082: P 2417:2569(152) ack 713 win 16808
<nop,nop,timestamp 4869838 325254> (DF)
                        4500 00cc 0545 4000 8006 82c2 xxxx xxxx
                        0a0a 0a01 06db 043a 3a3d 7df1 b9e7 6369
                        8018 41a8 403f 0000 0101 080a 004a 4ece
                        0004 f686 0100 0000 cefa 0bb0 8800 0000
01:57:12.965695 P x.x.x.A.1755 > 10.10.10.1.1082: P 2569:2625(56) ack 713 win 16808
<nop,nop,timestamp 4869838 325254> (DF)
                        4500 006c 0546 4000 8006 8321 xxxx xxxx
                        0a0a 0a01 06db 043a 3a3d 7e89 b9e7 6369
                        8018 41a8 7447 0000 0101 080a 004a 4ece
01:57:12.965853 P x.x.x.A.1755 > 10.10.10.1.1082: P 2625:2673(48) ack 713 win 16808
<nop,nop,timestamp 4869838 325254> (DF)
                        4500 0064 0547 4000 8006 8328 xxxx xxxx
                        0a0a 0a01 06db 043a 3a3d 7ec1 b9e7 6369
                        8018 41a8 cb17 0000 0101 080a 004a 4ece
                        0004 f686 0100 0000 cefa 0bb0 2000 0000
01:57:12.971943 P 10.10.10.1.1082 > x.x.x.A.1755: . 713:713(0) ack 2673 win 13388
<nop,nop,timestamp 325255 4869838> (DF)
                        4500 0034 03ea 4000 4006 c4b5 c80e f105
                        xxxx xxxx 043a 06db b9e7 6369 3a3d 7ef1
                        8010 344c a913 0000 0101 080a 0004 f687
01:57:24.102639 P x.x.x.A.1755 > 10.10.10.1.1082: R 977108721:977108721(0) win 0 (DF)
                        4500 0028 0549 4000 8006 8362 xxxx xxxx
                        0a0a 0a01 06db 043a 3a3d 7ef1 b9e7 6369
                        5004 0000 5c27 0000 0770 7561 7169
```

Analyzing

1. This transaction hides the real intention. In general we can say that the TCP Options are using in several transactions with this particular service for example. Initially is a little difficult determine why the service is stopped. Because the session is alive and continuous transactions is occurred, but the video casting is down.

2. Seeing in the 12th octet of TCP header, you can see that TCP OPTIONS is active. Due to vulnerability in the MS Streaming Media Server, the option called TIMESTAMP permit that attacker consumes the resources from the system. The signature is underlined in red.

3. We can see continuous arrive from packets with timestamp option. By reviewing the Microsoft web page, we can find a patch to stop this attack.

4. This attack will pass standard a IDS, because is a valid transaction.

5. The most important thing in this detection is, the handshaking needed for complete the attack, If doesn't exist this, the DoS attack will never occur.

6. And finally is your security team against Denial-of-Service attack prepare a correct filter for this kind of DoS attack, your team will be able to catch the intruder, if the intruder are sending another kind of DoS attack, for example:

   tcpdump –w –i eth1 /home/username/msms-dos-filter.in
   ´(tcp and ((tcp[12] & 0xf0 >=112) and (tcp[13] & 0x1a !=0)) and port 1755)´

   Where:
   (tcp[12] & 0xf0 >=112) Is used for packets with TCP options greater than 2 bytes 112 dec. = 70 hex (7 is the number of bytes in TCP header)

© SANS Institute 2000 - 2002, Author retains full rights.

(tcp[13] & 0x1a !=0)

Is used for packets with SYN, PSH, ACK.

7. Remember that DoS are difficult to catch, but you can reduce the time to capture the intruder with a good filter.

INCIDENT 02

Severity: (1.Criticality + 1.Lethality) – (3.System + 4.Net Countermeasures)=-5
Technique: Stealthy Scanning
Intent: The attacker intent to be update with our changes, in our main zone.

```
0:02:46 <162>  Apr 19 2000   %PIX-2-106007   Deny inbound UDP from 192.168.76.122/7962 to 10.14.241.39/53 due to DNS Query
0:02:47 <162>  Apr 19 2000   %PIX-2-106007   Deny inbound UDP from 192.168.76.122/7962 to 10.14.241.39/53 due to DNS Query
0:02:49 <162>  Apr 19 2000   %PIX-2-106007   Deny inbound UDP from 192.168.76.122/7962 to 10.14.241.39/53 due to DNS Query
0:17:46 <162>  Apr 19 2000   %PIX-2-106007   Deny inbound UDP from 192.168.76.122/7966 to 10.14.241.39/53 due to DNS Query
0:17:47 <162>  Apr 19 2000   %PIX-2-106007   Deny inbound UDP from 192.168.76.122/7966 to 10.14.241.39/53 due to DNS Query
0:17:49 <162>  Apr 19 2000   %PIX-2-106007   Deny inbound UDP from 192.168.76.122/7966 to 10.14.241.39/53 due to DNS Query
0:32:45 <162>  Apr 19 2000   %PIX-2-106007   Deny inbound UDP from 192.168.76.122/7970 to 10.14.241.39/53 due to DNS Query
0:32:47 <162>  Apr 19 2000   %PIX-2-106007   Deny inbound UDP from 192.168.76.122/7970 to 10.14.241.39/53 due to DNS Query
0:32:48 <162>  Apr 19 2000   %PIX-2-106007   Deny inbound UDP from 192.168.76.122/7970 to 10.14.241.39/53 due to DNS Query
0:47:45 <162>  Apr 19 2000   %PIX-2-106007   Deny inbound UDP from 192.168.76.122/7974 to 10.14.241.39/53 due to DNS Query
0:47:47 <162>  Apr 19 2000   %PIX-2-106007   Deny inbound UDP from 192.168.76.122/7974 to 10.14.241.39/53 due to DNS Query
0:47:48 <162>  Apr 19 2000   %PIX-2-106007   Deny inbound UDP from 192.168.76.122/7974 to 10.14.241.39/53 due to DNS Query
```

1. This scan was discover in the SYSLOG server (PIX firewall). Here we can see request to a server that not listens in this port. But the most important thing here is the frequency in each request (15 min) in all days, and all weeks
   For these case the 5n0r7.c tool can help you:
   http://packetstorm.securify.com/sniffers/snort/5n0r7.c

2. The sequence encourages us, to capture all packets forward to this server. And we can see that:

```
root@ANALYZER /root]# tcpdump -x -r /home/jromero/INCIDENT00-192.168.76.122-53
13:01:20.911728 P 192.168.76.122.619 > 10.14.241.39.domain: 7374+ A? HELP.mydomain.com. (50)
13:05:20.912729 P 192.168.76.122.631 > 10.14.241.36.domain: 7452+ A? HELP.mydomain.com. (50)
```

3. In the next analysis, we can see that scanning is not only for no-DNS servers, is for real DNS servers too:

```
10:44:17.908400 P 192.168.76.122.725 > 10.14.241.36.domain: 120+ A? HELP.mydomain.com. (50)
10:44:17.908803 P 10.14.241.36.domain > 192.168.76.122.725: 120 NXDomain* 0/1/0 (111)
```

   When the request goes to a real DNS server, it gets response with error.

4. We can see now, the real objective. That is: Know when the mydomain.com zone has changed. This can get, because the DNS response with the serial number when exist a error.

5. The intruder waits for a response with error, because he knows that HELP.mydomain.com don´t exist. But he waits the serial number. We can reproduce the packet with a Packet Generator:

```
DNS: ----- Internet Domain Name Service header -----
      DNS:
      DNS: ID = 7374
      DNS: Flags = 85
      DNS: 1... .... = Response
      DNS: .... .1.. = Authoritative answer
      DNS: .000 0...   = Query
      DNS: .... ..0. = Not truncated
      DNS: Flags = 8X
      DNS: ..0. .... = Data NOT verified
```

© SANS Institute 2000 - 2002 As part of GIAC practical repository. Author retains full rights.

```
DNS: 1... .... = Recursion available
DNS: Response code = Name error (3)
DNS: ...0 .... = Unicast packet
DNS: Question count = 1, Answer count = 0
DNS: Authority count = 1, Additional record count = 0
DNS:
DNS: ZONE Section
DNS:     Name = HELP.mydomain.com
DNS:     Type = Host address (A,1)
DNS:     Class = Internet (IN,1)
DNS:
DNS: Authority section:
DNS:     Name = mydomain.com
DNS:     Type = Start of zone authority (SOA,6)
DNS:     Class = Internet (IN,1)
DNS:     Time-to-live = 3600 (seconds)
DNS:     Length = 43
DNS:     Server that is original data source = ns3.mydomain.com
DNS:     Responsible mailbox = dnsmaster.mail.mydomain.com
DNS:     Serial number = 22754 (This change when the zone is modified)
DNS:     Refresh = 3600
DNS:     Retry = 600
DNS:     Expire = 86400
DNS:     Minimum time-to-live = 3600 (seconds)
DNS:
```

6. The intruder did this complex process, to break out a automatic IDS like (Cisco IDS). Cisco IDS alerts you when somebody is asking about the complete information of zones in your DNS server. Another good technical from intruder was the sequence. But he make a mistake when try to ask to another host.

7. To catch this kind of stealthy scan with more accuracy, you can use a filter:

   tcpdump –w /home/yourdomainscan ´udp[11]=0x83´

   Where: 83 is indicative that the request had a mistake.

© SANS Institute 2000 - 2002 As part of GIAC practical repository. Author retains full rights.

INCIDENT 03

Severity: (5.Criticality + 5.Lethality) – (4.System + 4.Net Countermeasures)=2
Technique: Trojan for remote polling + Microsoft basic services
Intent: This kind of problems may occur in an ISP, and if you don't see with care, this kind of false positives can change easily to the prelude of a real attack when the origin is a monitor and is manage by another front via trojan. (A trojan can be use to polling the time when you server is ready to listen a port)

File Attached
Review attach file jromero01.txt

Briefing
(host Nro. 1)

```
<162>Apr 10 2000 20:32:22: %PIX-2-106006: Deny inbound UDP from 216.244.156.251/137 to 10.14.241.38/137
<162>Apr 10 2000 20:32:24: %PIX-2-106006: Deny inbound UDP from 216.244.156.251/137 to 10.14.241.38/137
<162>Apr 10 2000 20:32:25: %PIX-2-106006: Deny inbound UDP from 216.244.156.251/137 to 10.14.241.38/137
<162>Apr 10 2000 20:34:22: %PIX-2-106006: Deny inbound UDP from 216.244.156.251/137 to 10.14.241.36/137
<162>Apr 10 2000 20:34:24: %PIX-2-106006: Deny inbound UDP from 216.244.156.251/137 to 10.14.241.36/137
<162>Apr 10 2000 20:34:25: %PIX-2-106006: Deny inbound UDP from 216.244.156.251/137 to 10.14.241.36/137
<162>Apr 10 2000 20:36:22: %PIX-2-106006: Deny inbound UDP from 216.244.156.251/137 to 10.14.241.36/137
<162>Apr 10 2000 20:36:24: %PIX-2-106006: Deny inbound UDP from 216.244.156.251/137 to 10.14.241.36/137
<162>Apr 10 2000 20:36:25: %PIX-2-106006: Deny inbound UDP from 216.244.156.251/137 to 10.14.241.36/137
<162>Apr 10 2000 20:38:22: %PIX-2-106006: Deny inbound UDP from 216.244.156.251/137 to 10.14.241.36/137
<162>Apr 10 2000 20:38:24: %PIX-2-106006: Deny inbound UDP from 216.244.156.251/137 to 10.14.241.36/137
```

(host Nro. 2)

```
13:12:58.836086 P 216.244.146.38.netbios-ns > 10.14.241.36.netbios-ns:NBT UDP PACKET(137):
QUERY; REQUEST; UNICAST
                        4500 004e 438f 0000 7c11 d6c1 d8f4 9226
                        0a0e f124 0089 0089 003a e9c1 83ee 0100
                        0001 0000 0000 0000 2046 4445 5045 4746
                        4546 4845 4246 4345 4643 4143 4143 4143
                        4143 4143 4143 4142 4d00 0020 0001
13:14:59.835315 P 216.244.146.38.netbios-ns > 10.14.241.38.netbios-ns:NBT UDP PACKET(137):
QUERY; REQUEST; UNICAST
                        4500 004e 448f 0000 7c11 d5bf d8f4 9226
                        0a0e f126 0089 0089 003a e9c3 83ea 0100
                        0001 0000 0000 0000 2046 4445 5045 4746
                        4546 4845 4246 4345 4643 4143 4143 4143
                        4143 4143 4143 4142 4d00 0020 0001
13:16:00.335195 P 216.244.146.38.netbios-ns > 10.14.241.36.netbios-ns:NBT UDP PACKET(137):
QUERY; REQUEST; UNICAST
                        4500 004e 458f 0000 7c11 d4c1 d8f4 9226
                        0a0e f124 0089 0089 003a e9c1 83ee 0100
                        0001 0000 0000 0000 2046 4445 5045 4746
                        4546 4845 4246 4345 4643 4143 4143 4143
                        4143 4143 4143 4142 4d00 0020 0001
```

These host belong to customer networks from our Internet backbone.
Here we have got to hosts 216.244.156.251/200.14.241.38 are requesting each 2 min. (aprox.) by days, by week. These packets are direct to the DNS servers (Windows NT), primary and secondary.
Note: In the past, these servers maintained web pages (easy to know what kind of OS) over IIS.

© SANS Institute 2000 - 2002 As part of GIAC practical repository. Author retains full rights.

Analyzing

1. Here, we can see something like a sequence.

   Behavior with Sequence (freq. 2 min)

   <162>Apr 10 2000 20:32:22: %PIX-2-106006: Deny inbound UDP from 216.244.156.251/137 to 10.14.241.38/137
   <162>Apr 10 2000 20:32:24: %PIX-2-106006: Deny inbound UDP from 216.244.156.251/137 to 10.14.241.38/137
   <162>Apr 10 2000 20:32:25: %PIX-2-106006: Deny inbound UDP from 216.244.156.251/137 to 10.14.241.38/137
   <162>Apr 10 2000 20:34:22: %PIX-2-106006: Deny inbound UDP from 216.244.156.251/137 to 10.14.241.36/137
   <162>Apr 10 2000 20:34:24: %PIX-2-106006: Deny inbound UDP from 216.244.156.251/137 to 10.14.241.36/137
   <162>Apr 10 2000 20:34:25: %PIX-2-106006: Deny inbound UDP from 216.244.156.251/137 to 10.14.241.36/137
   <162>Apr 10 2000 20:36:22: %PIX-2-106006: Deny inbound UDP from 216.244.156.251/137 to 10.14.241.36/137
   <162>Apr 10 2000 20:36:24: %PIX-2-106006: Deny inbound UDP from 216.244.156.251/137 to 10.14.241.36/137
   <162>Apr 10 2000 20:36:25: %PIX-2-106006: Deny inbound UDP from 216.244.156.251/137 to 10.14.241.36/137

   By reviewing the lines, is observed that each request arrived in intervals of 2 min.

2. The analyst can think that this is a event to fill the log file. Really theses events filled our logs. The reached size was 10 times more than normal size, from 250 KB to 2500 KB. For that we can think that the objective is rise the sizes of the log file.

3. Due to both host belong to 2 different networks (LANs in our WAN). We can think that the real sources are spoofing the IP. For that reason we called to the administrators of these LANs, to indicate the correct configuration for DNS and not WINS. And this was the real problem.

4. A good analysis was dumped the data that was request in the netbios packet, which was a normal request to the inadequate host.

5. In the future a ISP can suffer this kind of event. Where, the log will fill fully, the attacker distract to the analyst.

6. But the most important will be that a attacker will use to another host to monitor a ISP (while the administrator in the ISP guess that is a problem with a configuration in the customer network). The administrator will be receiving packets from a passive monitor system installed (like a trojan) in a customer network.

What do you can if you are in a real attack?

1. As analyst, we can redirect the traffic to this port by forwarding to another host (in the main router) in the external network (in the network where the Firewall's outside interface is), with the objective to know what is the real intention of the intruder when the port is open.
2. In the exhaustive revision on the packet you will find that the hosts query, or refresh information, and what will happen when the host (false scanner) find the ports open.
3. You will need active a analyzer/sensor for this server when you have ready the "redirection command" in your main router.
4. The most important thing here will be that you will find a trojan in the source host, that are being manage from another real intruder.
5. Before you will be able to active the log system in the router POP (point of presence) always if the stealth scanning is from your backbone (like Internet Service Provider). And there you will catch the real intruder that manage the trojan.

© SANS Institute 2000 - 2002, Author retains full rights.

© SANS Institute 2000 - 2002 As part of GIAC practical repository. Author retains full rights.

INCIDENT 04 "significant researched analysis" – posted in GIAC – "Proxy Note"

http://www.sans.org/y2k/proxy.htm

Severity: (4.Criticality + 1.Lethality) – (4.System + 4.Net Countermeasures)=-3
Technique: Popular-Scan**, the source sent requests to real specific hosts, the attacker never failed to open session to non-existent host.**
Intent: The real proposal from the source is only catch our attention. Bellow, I explain more about this..

File attached
Review attach file jromero02.xls

Briefing:

3:35:41 <162>Apr 17 2000 %PIX-2-106001 Inbound TCP connection denied from 207.77.174.66/4607 to 10.14.241.200/1080 flags SYN
3:35:41 <162>Apr 17 2000 %PIX-2-106001 Inbound TCP connection denied from 207.77.174.66/4608 to 10.14.241.134/1080 flags SYN
3:35:41 <162>Apr 17 2000 %PIX-2-106001 Inbound TCP connection denied from 207.77.174.66/4609 to 10.14.241.135/1080 flags SYN
3:35:41 <162>Apr 17 2000 %PIX-2-106001 Inbound TCP connection denied from 207.77.174.66/4610 to 10.14.241.142/1080 flags SYN
3:36:34 <162>Apr 17 2000 %PIX-2-106001 Inbound TCP connection denied from 207.77.174.66/1143 to 10.14.241.200/3128 flags SYN
3:36:34 <162>Apr 17 2000 %PIX-2-106001 Inbound TCP connection denied from 207.77.174.66/1144 to 10.14.241.134/3128 flags SYN
3:36:34 <162>Apr 17 2000 %PIX-2-106001 Inbound TCP connection denied from 207.77.174.66/1145 to 10.14.241.135/3128 flags SYN
3:36:34 <162>Apr 17 2000 %PIX-2-106001 Inbound TCP connection denied from 207.77.174.66/1146 to 10.14.241.142/3128 flags SYN
3:37:10 <162>Apr 17 2000 %PIX-2-106001 Inbound TCP connection denied from 207.77.174.66/1447 to 10.14.241.134/8080 flags SYN
3:37:10 <162>Apr 17 2000 %PIX-2-106001 Inbound TCP connection denied from 207.77.174.66/1448 to 10.14.241.135/8080 flags SYN
3:37:10 <162>Apr 17 2000 %PIX-2-106001 Inbound TCP connection denied from 207.77.174.66/1449 to 10.14.241.142/8080 flags SYN
3:37:10 <162>Apr 17 2000 %PIX-2-106001 Inbound TCP connection denied from 207.77.174.66/1450 to 10.14.241.143/8080 flags SYN

Analyzing

This is a real detection in our network, please don't confuse this detection like a lie.
But, I will submit it like a "significant researched analysis", if you prefer.

1. I detected a scan for around 10 minutes, from the UUNet network. (Since 3:27 AM to 3:48 AM)
2. The possible source was Contractor's Internet Services (NETBLK-CLINTON207) CLINTON207 207.77.168.0 - 207.77.175.0.
3. For this reason, we have two appreciations around this incident:
   a. This can be a method to know what kind of response will make the intrusion detection analyst when this event is occurred.
      I say that for the following response from this host:

```
C:\WINDOWS>ping -i 128 207.77.174.66
Pinging 207.77.174.66 with 32 bytes of data:
Reply from 207.77.168.201: TTL expired in transit.
Reply from 207.77.168.201: TTL expired in transit.
Reply from 207.77.168.201: TTL expired in transit.
Reply from 207.77.168.201: TTL expired in transit.

Ping statistics for 207.77.174.66:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
    Minimum = 0ms, Maximum =  0ms, Average =  0ms

C:\WINDOWS>ping -i 255 207.77.174.66
Pinging 207.77.174.66 with 32 bytes of data:
Reply from 207.77.168.26: TTL expired in transit.
Request timed out.
Reply from 207.77.168.26: TTL expired in transit.
Reply from 207.77.168.26: TTL expired in transit.

Ping statistics for 207.77.174.66:
    Packets: Sent = 4, Received = 3, Lost = 1 (25% loss),
```

© SANS Institute 2000 - 2002 As part of GIAC practical repository. Author retains full rights.

Approximate round trip times in milli-seconds:
Minimum = 0ms, Maximum = 0ms, Average = 0ms

b. Spoof the source with addresses from another networks, to make that victim take the laws to solution the problem.
I can't confirm that, because my firewall doesn't response request, and my server never responded these requests. If some of them response, I will be able to check if the source was the real source. (If it was not, the host will be responding with a port unreachable message)

4. This scan sent 27 packets in only 1 second.
5. This scan went to the correct address in use. The knowledge of the intruder about this network was very good.
6. I recommended make a new filter to catch it. In special areas like a ISP, capture all data in the networks could be a big problem, the principal logs are very useful, in this case, we can see the logs from the firewall, that was good.

© SANS Institute 2000 - 2002, Author retains full rights.

© SANS Institute 2000 - 2002 As part of GIAC practical repository. Author retains full rights.

INCIDENT 05

Severity: (5.Criticality + 5.Lethality) – (4.System + 4.Net Countermeasures)=2
Technique: Spoof packets through the main router
Intent: This attacker is trying to start a DoS, without a previous scan. This DoS will be never success, due to the restriction in the firewall. But in this occasion, these spoffing packets can help you to determine if you have tries of DoS Attack.

<162>	Apr 17 2000	8:55:51 %PIX-2-106006	Deny inbound UDP from 1.1.1.0/137 to 10.14.241.41/137
<162>	Apr 17 2000	8:55:53 %PIX-2-106006	Deny inbound UDP from 1.1.1.0/137 to 10.14.241.41/137
<162>	Apr 17 2000	8:58:36 %PIX-2-106006	Deny inbound UDP from 172.21.16.4/137 to 10.14.241.41/137
<162>	Apr 17 2000	8:58:37 %PIX-2-106006	Deny inbound UDP from 172.21.16.4/137 to 10.14.241.41/137
<162>	Apr 17 2000	8:58:39 %PIX-2-106006	Deny inbound UDP from 172.21.16.4/137 to 10.14.241.41/137
<162>	Apr 17 2000	9:08:35 %PIX-2-106006	Deny inbound UDP from 172.18.9.2/137 to 10.14.241.41/137
<162>	Apr 17 2000	9:08:37 %PIX-2-106006	Deny inbound UDP from 172.18.9.2/137 to 10.14.241.41/137
<162>	Apr 17 2000	9:08:38 %PIX-2-106006	Deny inbound UDP from 172.18.9.2/137 to 10.14.241.41/137
<162>	Apr 17 2000	9:13:30 %PIX-2-106006	Deny inbound UDP from 172.30.0.42/137 to 10.14.241.41/137
<162>	Apr 17 2000	9:13:32 %PIX-2-106006	Deny inbound UDP from 172.30.0.42/137 to 10.14.241.41/137
<162>	Apr 17 2000	9:13:33 %PIX-2-106006	Deny inbound UDP from 172.30.0.42/137 to 10.14.241.41/137
<162>	Apr 17 2000	9:28:30 %PIX-2-106006	Deny inbound UDP from 192.168.1.2/137 to 10.14.241.41/137
<162>	Apr 17 2000	9:30:33 %PIX-2-106006	Deny inbound UDP from 172.21.16.4/137 to 10.14.241.41/137
<162>	Apr 17 2000	9:30:35 %PIX-2-106006	Deny inbound UDP from 172.21.16.4/137 to 10.14.241.41/137
<162>	Apr 17 2000	9:30:36 %PIX-2-106006	Deny inbound UDP from 172.21.16.4/137 to 10.14.241.41/137
<162>	Apr 17 2000	9:32:13 %PIX-2-106006	Deny inbound UDP from 192.168.1.2/137 to 10.14.241.41/137
<162>	Apr 17 2000	9:32:16 %PIX-2-106006	Deny inbound UDP from 192.168.1.2/137 to 10.14.241.41/137
<162>	Apr 17 2000	9:54:41 %PIX-2-106006	Deny inbound UDP from 172.16.20.5/137 to 10.14.241.41/137

Analyzing

1. This is a transaction with spoofing sources.
2. The objective from the hacker side is a DoS Attack using port 137.
3. A IDS can detect this packets with RFC 1918 address, but it can't determine if the number of packets is a DoS attack. Only a manual searching will be able to catch the real intent.
4. The main router has permitted to pass through traffic with source IP address from private networks, but the PIX firewall never left pass this packets.
5. In this case, these addresses permit to our analysis "trace the real intent". Due to we had another continuos transactions in the same time period.
6. The most important in these kind of incidents is the capability to get additional information, which offers this kind of packets.

© SANS Institute 2000 - 2002, Author retains full rights.

INCIDENT 06

Severity: (4.Criticality + 5.Lethality) – (4.System + 4.Net Countermeasures)=1
Technique: Automatic scanner from any kind of Internet worm
Intent: An automated program to infect another systems.

```
<162> Apr 17 2000  9:07:26 %PIX-2-106006  Deny inbound UDP from 100.134.1.2/137 to 10.14.241.41/137
<162> Apr 17 2000  9:21:18 %PIX-2-106006  Deny inbound UDP from 11.160.125.201/137 to 10.14.241.36/137
<162> Apr 17 2000 12:11:22 %PIX-2-106006  Deny inbound UDP from 12.128.105.50/137 to 10.14.241.41/137
<162> Apr 17 2000 14:30:26 %PIX-2-106006  Deny inbound UDP from 128.1.1.2/137 to 10.14.241.41/137
<162> Apr 17 2000 12:17:32 %PIX-2-106006  Deny inbound UDP from 128.10.81.200/137 to 10.14.241.42/137
<162> Apr 17 2000  9:17:01 %PIX-2-106006  Deny inbound UDP from 130.110.1.232/137 to 10.14.241.41/137
<162> Apr 17 2000 11:44:35 %PIX-2-106006  Deny inbound UDP from 131.107.3.70/137 to 10.14.241.42/137
<162> Apr 17 2000  7:24:49 %PIX-2-106006  Deny inbound UDP from 200.32.74.57/137 to 10.14.241.41/137
```

Here we have a scan from multiple networks trying to open a session in udp port 137 with several servers in our data center.
(I have cut a briefing for this incident. This packet arrived to the network only twice for week)

Analyzing

1. By searching these addresses we can find the following:
   Netblock: 96.0.0.0 - 126.255.255.255 (ARIN)
   Netblock: 11.0.0.0 - 11.255.255.255 (Defense Intelligence Agency)
   Netblock: 12.0.0.0 - 12.255.255.255 (AT&T ITS)
   Netnumber: 128.1.0.0 (BBN Communications – BBN-TESTNET)
   Netnumber: 128.10.0.0 (Purdue University)
   Netnumber: 130.110.0.0 (Combustion Engineering)
   Netnumber: 131.107.0.0 (Microsoft Corporation)
   (Netblock: 200.128.0.0 - 200.255.255.0) (Netname: BRAZIL-BLK2)

2. About the frequency from these packets, I prefer to show only a sample, something like this:

```
<162> Apr 17 2000  7:06:02 %PIX-2-106006  Deny inbound UDP from 207.234.35.61/137 to 10.14.241.38/137
<162> Apr 17 2000  7:24:46 %PIX-2-106006  Deny inbound UDP from 200.32.74.50/137 to 10.14.241.41/137
<162> Apr 17 2000  7:24:46 %PIX-2-106006  Deny inbound UDP from 200.32.74.53/608 to 10.14.241.41/137
<162> Apr 17 2000  7:24:46 %PIX-2-106006  Deny inbound UDP from 200.32.74.57/137 to 10.14.241.41/137
<162> Apr 17 2000  7:24:48 %PIX-2-106006  Deny inbound UDP from 200.32.74.50/137 to 10.14.241.41/137
<162> Apr 17 2000  7:24:48 %PIX-2-106006  Deny inbound UDP from 200.32.74.53/608 to 10.14.241.41/137
<162> Apr 17 2000  7:24:48 %PIX-2-106006  Deny inbound UDP from 200.32.74.57/137 to 10.14.241.41/137
<162> Apr 17 2000  7:24:49 %PIX-2-106006  Deny inbound UDP from 200.32.74.50/137 to 10.14.241.41/137
<162> Apr 17 2000  7:24:49 %PIX-2-106006  Deny inbound UDP from 200.32.74.53/608 to 10.14.241.41/137
<162> Apr 17 2000  7:24:49 %PIX-2-106006  Deny inbound UDP from 200.32.74.57/137 to 10.14.241.41/137
<162> Apr 17 2000  7:29:35 %PIX-2-106006  Deny inbound UDP from 207.234.35.61/137 to 10.14.241.36/137
```

   Maybe a DoS attack, but the tries from each source are only a few ones, is very low to change a DoS attack. Finally, this is a port scan.

3. Finally this incident show us a comprised networks with any kind of auto-amplifier virus, We can think that is a scanning to initiate a session with our Microsoft servers.
4. But the first IP address for example, belong to addresses reserved by the IANA, and only appears 3 time bye day (remember that this incident occur 2 or 3 days for week). And this behavior shows us a new signature. When the first request is the prelude for a long series of request to 137 port.

© SANS Institute 2000 - 2002 As part of GIAC practical repository. Author retains full rights.

INCIDENT 07

Severity: (4.Criticality + 1.Lethality) – (4.System + 4.Net Countermeasures)=-3
No exist inner routers.
Technique: Reconnaissance of Allocation in Backbone via traceroute (33434).
Intent: The intruder is sensing the changes in the internal network. This incident can be repeating in the rest of the network. But we can't capture more details in the rest of routers.

TRACES WITH 33434

<162> Apr 21 2000 9:06:09 %PIX-2-106006 Deny inbound UDP from 216.33.87.8/2713 to 10.14.241.36/33434
<162> Apr 21 2000 9:06:10 %PIX-2-106006 Deny inbound UDP from 216.33.87.8/2714 to 10.14.241.36/33434
<162> Apr 21 2000 9:06:11 %PIX-2-106006 Deny inbound UDP from 216.33.87.8/2715 to 10.14.241.36/33434
<162> Apr 21 2000 9:06:15 %PIX-2-106006 Deny inbound UDP from 216.33.87.9/2812 to 10.14.241.36/33434
<162> Apr 21 2000 9:06:16 %PIX-2-106006 Deny inbound UDP from 216.33.87.9/2813 to 10.14.241.36/33434
<162> Apr 21 2000 9:06:17 %PIX-2-106006 Deny inbound UDP from 216.33.87.9/2814 to 10.14.241.36/33434
<162> Apr 21 2000 9:06:19 %PIX-2-106006 Deny inbound UDP from 216.33.87.9/2815 to 10.14.241.36/33434

Analyzing

1. This intruder is monitoring the change in our topology. For example, to know if our data center has a router or a firewall.
2. This kind of traces can indicate that the intruder is trying to know what kind of gateway our DNS have in the perimeter.
3. The probability that the intruder is running others systems to check our topology is high.
4. With a post-analysis the intruder can know the allocation that our data center in the backbone, to be use in futures DoS attack, where the bandwidth is a high factor.
5. I recommend active filters to alert to the IDS central (analyzer) when we have a high request from other host on that port.
   The principal condition can be that the requests are generating each day for week, in logical intervals of time.
6. When you have a firewall you will be able to filter this traces like our case.

© SANS Institute 2000 - 2002 As part of GIAC practical repository. Author retains full rights.

INCIDENT 08

Severity: (5.Criticality + 5.Lethality) – (4.System + 4.Net Countermeasures)=2
Our firewall drops this kind of packets.
We don't have internal routers.
In that moment we didn't capture packets with IP options.

Technique: Firewall reconnaissance with IP options packets.

Intent: Due to we couldn't capture the total of the frame (never was occur again), we can say, that the intent was know more about the firewall, because the firewall PIX drop the packets when they use IP options.

<162> Apr 17 2000 6:53:41 %PIX-2-106012 Deny IP from 177.22.2.27 to 200.14.241.36, IP options 0x80310a24
<162> Apr 17 2000 6:53:49 %PIX-2-106012 Deny IP from 177.22.2.27 to 200.14.241.36, IP options 0x80350ce4
<162> Apr 17 2000 6:53:59 %PIX-2-106012 Deny IP from 177.22.2.27 to 200.14.241.36, IP options 0x80298174
<162> Apr 17 2000 6:54:15 %PIX-2-106012 Deny IP from 177.22.2.27 to 200.14.241.38, IP options 0x8033d4b4
<162> Apr 17 2000 6:54:20 %PIX-2-106012 Deny IP from 177.22.2.27 to 200.14.241.38, IP options 0x802cb414
<162> Apr 17 2000 6:54:31 %PIX-2-106012 Deny IP from 177.22.2.27 to 200.14.241.38, IP options 0x8030f554

Analyzing

1. By searching the meaning of 0x80 in the documentation about IP Options, we didn't find any information about it.
2. This is a example when the false positives are using to know more about our security devices.
3. Due to a bug (CSCdm 66259) in a PIX Firewall, we got wrong values for the IP Options.
4. In conclusion without more information, this detection is impossible to determine what's going on. We will need to review this kind of traffic with sensor w/TCPDUMP.

© SANS Institute 2000 - 2002 Author retains full rights.

INCIDENT 09

Severity: (5.Criticality + 5.Lethality) – (4.System + 4.Net Countermeasures)=2
Our system use NetBios name, but only to internal communication. The Windows hosts in the DMZ aren't connected to our internal system.

Technique: NetBios Name service (udp port 137)

Intent: Know more about the domains in the internal networks (NT network). This kind of attack can be reported from another systems, I note this because the source didn't send another request to our network.

```
<162>  Apr 17 2000     0:50:38 %PIX-2-106006  Deny inbound UDP from 194.130.67.128/137 to 10.14.241.36/137
<162>  Apr 17 2000     0:50:38 %PIX-2-106006  Deny inbound UDP from 194.130.67.128/137 to 10.14.241.38/137
<162>  Apr 17 2000     0:50:38 %PIX-2-106006  Deny inbound UDP from 194.130.67.254/137 to 10.14.241.36/137
<162>  Apr 17 2000     0:50:38 %PIX-2-106006  Deny inbound UDP from 194.130.67.254/137 to 10.14.241.38/137
<162>  Apr 17 2000     0:50:39 %PIX-2-106006  Deny inbound UDP from 194.130.67.128/137 to 10.14.241.36/137
<162>  Apr 17 2000     0:50:39 %PIX-2-106006  Deny inbound UDP from 194.130.67.128/137 to 10.14.241.38/137
<162>  Apr 17 2000     0:50:39 %PIX-2-106006  Deny inbound UDP from 194.130.67.254/137 to 10.14.241.36/137
<162>  Apr 17 2000     0:50:39 %PIX-2-106006  Deny inbound UDP from 194.130.67.254/137 to 10.14.241.38/137
<162>  Apr 17 2000      0:50:4 %PIX-2-106006  Deny inbound UDP from 194.130.67.128/137 to 10.14.241.36/137
```

Analyzing

1. This trace was show in the SYSLOG server, and then of rigorous analysis was determine like a scan from the network 194.130.67.128 (Hill & Knowlton (UK) Ltd). 4 request each minute x hour x day x week.
2. Then the source was dumped from the FULL-LOG in the sensor running TCPDUMP, and all packets showed the same pattern:

```
10:55:14.356596 P 194.130.67.128.netbios-ns > 10.14.241.36.netbios-ns:NBT UDP PACKET(137):
QUERY; REQUEST; UNICAST
                        4500 004e 4f58 0000 7011 3c11 c282 4380
                        0a0e f124 0089 0089 003a 55aa ab4e 0000
                        0001 0000 0000 0000 2043 4b41 4141 4141
                        4141 4141 4141 4141 4141 4141 4141 4141
                        4141 4141 4141 4141 4100 0021 0001
10:55:14.371180 P 194.130.67.254.netbios-ns > 10.14.241.36.netbios-ns:NBT UDP PACKET(137):
QUERY; REQUEST; UNICAST
                        4500 004e 5058 0000 7011 3a93 c282 43fe
                        0a0e f124 0089 0089 003a 552a ab50 0000
                        0001 0000 0000 0000 2043 4b41 4141 4141
                        4141 4141 4141 4141 4141 4141 4141 4141
                        4141 4141 4141 4141 4100 0021 0001
```

3. To know what command sent the source, we used a NetBios decoder (NAI's sniffer), and saw the real intent from the source. Through a correct command via NetBios request, you can get the database from a WINS server about all domains name. The command is as follow (above in red)

```
WINS: ----- WINS Name Service header -----
      WINS:
      WINS: ID = 58436
      WINS: Flags = 00
      WINS: 0... .... = Command
      WINS: .000 0...   = Query
      WINS: .... ..0. = Not truncated
      WINS: .... ...0 = No recursion desired
      WINS: Flags = 1X
      WINS: ...1 .... = Non Verified data is acceptable
      WINS: Question count = 1, Answer count = 0
      WINS: Authority count = 0, Additional record count = 0
      WINS:
      WINS: Question section:
      WINS:     Name = *<00000000000000000000000000><00> <Workstation/Redirector>
      WINS:     Type = NetBIOS node status (WINS) (NetBIOS node,33)
```

© SANS Institute 2000 - 2002 As part of GIAC practical repository. Author retains full rights.

```
      WINS:     Class = Internet (IN,1)
      WINS:
```

4. The result from this command could be.

```
WINS: ----- WINS Name Service header -----
      WINS:
      WINS: ID = 58436
      WINS: Flags = 84
      WINS: 1... .... = Response
      WINS: .... .1.. = Authoritative answer
      WINS: .000 0...   = Query
      WINS: .... ..0. = Not truncated
      WINS: Flags = 0X
      WINS: ..0. .... = Data NOT verified
      WINS: 0... .... = Recursion not available
      WINS: Response code = OK (0)
      WINS: ...0 .... = Unicast packet
      WINS: Question count = 0, Answer count = 1
      WINS: Authority count = 0, Additional record count = 0
      WINS:
      WINS: Answer section:
      WINS:     Name = *<00000000000000000000000000000000><00> <Workstation/Redirector>
      WINS:     Type = NetBIOS node status (WINS) (NetBIOS node,33)
      WINS:     Class = Internet (IN,1)
      WINS:     Time-to-live = 0 (seconds)
      WINS:     Length = 317
      WINS: Number of names = 15
      WINS: NetBIOS name = NETWORK001      <00>
      WINS: Name flags = 44
      WINS:  0... .... = Unique NetBIOS name
      WINS:  .10. ....   = M-type node
      WINS:  .... .1..   = Active name
      WINS:  .... ..0. = Not permanent name
      WINS: NetBIOS name = NETWORK002
      WINS: Name flags = 44
      WINS:  0... .... = Unique NetBIOS name
      WINS:  .10. ....   = M-type node
      WINS:  .... .1..   = Active name
      WINS:  .... ..0. = Not permanent name
      WINS: NetBIOS name = NETWORK003      <00>
      WINS: Name flags = C4
      WINS:  1... .... = Group NetBIOS name
      WINS:  .10. ....   = M-type node
      WINS:  .... .1..   = Active name
      WINS:  .... ..0. = Not permanent name
        .
        .
        .
WINS: NetBIOS name = <0102>__MSBROWSE__<0201>
      WINS: Name flags = C4
      WINS:  1... .... = Group NetBIOS name
      WINS:  .10. ....   = M-type node
      WINS:  .... .1..   = Active name
      WINS:  .... ..0. = Not permanent name
      WINS:
      WINS: ------- NODE STATISTICS -------
      WINS:
      WINS: Unit ID = 00105A1C6682
      WINS: Jumpers = 00
      WINS: Test result = 00
      WINS: Version number = 0.0
      WINS: Statistics period = 0
      WINS: CRC errors = 0
      WINS: Alignment errors = 0
      WINS: Collisions = 0
      WINS: Send aborts = 0
      WINS: Good sends = 0
      WINS: Good receives = 0
      WINS: Retransmits = 0      WINS: No resource conditions = 0
      WINS: Free command blocks = 0
      WINS: Total command blocks = 0
```

© SANS Institute 2000 - 2002, Author retains full rights.

```
WINS: Maximum total command blocks = 0
WINS: Pending sessions = 0
WINS: Maximum pending sessions = 0
WINS: Total sessions possible = 0
WINS: Session data packet size = 0
WINS:
```

INCIDENT 10 "significant researched analysis" – posted in GIAC "April 22, 2000"

http://www.sans.org/y2k/042200.htm

Severity: **---To be qualify for each one.**
Technique: **---Multiple scanning with a probable "remote tool"---**

Analyzing:

1. Only if you couldn't confirm with the network administrators from those sources, you probably has a new signature. I guess that this posted has been review and checking with respective source. And was impossible for you to check this source.
2. If it is so, you can be posting a really new behavior. Finally I guess that somebody is building a huge database of ISPs, and networks in the Internet.
3. Somebody teach me, always think in the worst incident. But, I think that this is another kind of incident. I think that you are experiencing a new MACRO scan, we will probably suffer the same scan in the future. Maybe somebody is building a huge database with information about all sites. Information, like response or not response to the scan, ports to listen, operative hosts, etc.
4. Is important create a good language to posted this kind of incidents in order to get the real intent, in a quick time.

Posted in GIAC

(I am willing to say, I have no clue, and I sure have no clue on this one. Anyone else see this or know the answer?)
Greetings,
On Tuesday night 04/19/00 from 2000/04/19 9:21:25 PM GMT -0400 to 2000/04/19 10:42:12 PM GMT -0400 I detected numerous scans of port 37015 and 47015. These scans appear to have originated from several sources, including:
216.161.215.139
216.63.97.39
194.229.103.215
216.103.51.189
128.211.218.115
24.1.97.66
131.215.103.89
208.61.0.233
209.51.167.221
195.197.41.184
141.64.111.90
63.10.148.174
12.77.153.134
63.17.105.92
24.48.150.25
208.14.222.162
24.94.251.83
216.232.96.245
24.66.196.134
24.92.134.35
There were approximately 100 scans in the course of about an hour. They typically are in groups of three or six. Here are some examples:
2000/04/19 9:21:25 PM GMT -0400: Linksys LNE100TX ..[0004]
[No matching rule] Blocking incoming UDP: src=216.161.215.139, dst=24.24.60.246,

© SANS Institute 2000 - 2002 As part of GIAC practical repository. Author retains full rights.

sport=3232, dport=37015.
2000/04/19 10:15:25 PM GMT -0400: Linksys LNE100TX ..[0004][No matching rule] Blocking incoming UDP: src=208.14.222.162, dst=24.24.60.246, sport=4358, dport=47015.
What is port 37015 used for?

© SANS Institute 2000 - 2002, Author retains full rights.