Interested in learning
more about security?

SANS Institute
InfoSec Reading Room

This paper is from the SANS Institute Reading Room site. Reposting is not permitted without express written permission.

Simple Traffic Analysis With Ethereal

This paper describes how to use the Ethereal Display Filter to examine a capture log file. The data analyzed was recorded by port and the amount of packet traffic received. The attack patterns that emerged from the data analysis generally correspond with well published vulnerabilities from expected open ports on a server. Attackers also seem to have a variety of ways to get a server and/or firewall to acknowledge traffic and verify a potential target.

Copyright SANS Institute
Author Retains Full Rights

AD

Simple Traffic Analysis With Ethereal

GIAC Security Essentials
Certification (GSEC)
Practical Assignment
Version 1.4c

Option 1 - Research on Topics
in Information Security

Submitted by: <Neil Orlando>
Location: <SANS@Home>
March 16, 2005.

Paper Abstract: The use of Ethereal Display filters
to examine a capture log and see patterns in
attempted attacks against the server.

© SANS Institute 2000 - 2005 Author retains full rights.

Table of Contents

Abstract/Summary 1
Introduction 1
The Setup 1
A Review and Use of Filters 2
Results of Viewing the Captured Data with Display Filters 4
Conclusion 12
References 14

List of Figures

Figure 1. Filter Tool Bar 2
Figure 2. TCP 135 NT RPC 5
Figure 3. TCP 445 Microsoft SMB 6
Figure 4. UDP 137 NetBIOS Name Server 10
Figure 5. Protocol Hierarchy Summary Graph 11

© SANS Institute 2000 - 2005, Author retains full rights.

Abstract/Summary

This paper describes how to use the Ethereal Display Filter to examine a capture log file. The data analyzed was recorded by port and the amount of packet traffic received. The attack patterns that emerged from the data analysis generally correspond with well published vulnerabilities from expected open ports on a server. Attackers also seem to have a variety of ways to get a server and/or firewall to acknowledge traffic and verify a potential target.

Introduction

The goal of this project was to review traffic to a specific IP address by examining a capture log file via Ethereal, identifying the ports an intruder tried to use to connect, and finding any backdoor programs that use those ports.

The Setup

An additional IP, 66.92.xx.xx, was aliased to a network interface card (NIC) of a functioning web server to make it multi-homed. Tcpdump was run to capture the traffic to the IP address before the traffic hit the firewall and was dropped. This mechanism protected the existing web server that existed on another IP address. Nothing was running on IP 66.92.xx.xx during the test. In short, there were no reachable services.

The server was connected directly to the internet, and there were no routers, firewalls, or other packet filters in the way. In addition, no capture filters were used and the ISP did not block ports, so every packet captured was an actual packet that somebody tried sending to that IP address.

A total of 4297 packets were captured during an eight-day period:
Feb. 9, 2005, 21:52 hrs through Feb 18, 2005, 10:23 hrs
The machine from which these log files were captured ran Red Hat 9, kernel version 2.4.20-8.

Analysis was done on a Windows XP Professional box with an installation of Wincap 3.0 and Ethereal version 0.10.5.

© SANS Institute 2000 - 2005 Author retains full rights.

A Review and Use of Filters

Ethereal uses filtering to help sort and find the data. Ethereal can use capture filters and display filters. The capture filters are used when logging data to a file for later analysis. "The capture filter syntax follows the same syntax that Tcpdump uses from the libpcap library. This is used on the command line or in the capture filter dialog box to capture certain types of traffic."[1] This paper does not review capture filters, since the data examined for the test was already captured. The test used display filters to review the captured data and match traffic with specific protocols. Please note that display filters have a different command format from capture filters[2].

The following steps were used to load the existing capture file in order to view it.[3]
File – Open – change drives and directory to log capture file – Open

Display filters were accessed in two ways for this test:
1) Menu Bar - click Analyze –> Display Filters. This brings up the Ethereal Display Filter dialog box
2) via the filter tool bar found near the top of Ethereal screen (note: older versions of Ethereal had the filter tool bar on the bottom of the screen).

Figure 1. Filter Tool Bar

- Click on button listed as Filter on the left hand side to bring up the Ethereal Display Filter dialog box; or,
- use the text box if you know the exact filter you want to use.

  The left middle text box provides an area to enter or edit display filter strings… A syntax check of your filter string is done while you are typing. The background will turn red if you enter an incomplete or invalid string, and will become green when you enter a valid string. You can click on the pull down arrow to select a previously-entered filter string from a list. The entries in the pull down list will remain available even after a program restart.[4]

---

[1] Orebaugh, Angela. Ethereal Packet Sniffing. Syngress, 2004. Pg. 48.
[2] More information about capture filters can be found:
Sharpe, Richard. http://www.ethereal.com/docs/user-guide-sp/#ChUseCaptureMenuSection; and,
Orebaugh, Angela. Ethereal Packet Sniffing. Syngress, 2004. Pg. 209.
[3] Ethereal can read capture files from a variety of different tools. See the online documentation for more information on file formats: Sharpe, Richard. http://www.ethereal.com/docs/user-guide-sp/#ChUseFilterToolbarSection.

© SANS Institute 2000 - 2005 Author retains full rights.

A new filter must be created when the display filter box is opened if this is the first instance of running display filters. The following example illustrates how to create a filter to see all HTTP traffic to port 80.

Example. Filter traffic to port 80.

- Click New.
- Name the filter: "tcp destination port 80".
- Remove the word "new" from the filter string field.
- Click the "+ Expression" button to get a list of protocols that can be displayed.
- Scroll down to "TCP" (Transmission Control Protocol) in the field name and click the left arrow to get sub listings of TCP.
- Select tcp.dstport – Destination Port.

Note: There are various types of TCP port field names that can be used. This variety can also be seen with the other protocols, e.g., Internet Protocol (IP), Internet Control Message Protocol (ICMP), User Datagram Protocol (UDP), and Address Resolution Protocol (ARP).

- Click == in Relation field.
- Type "80" into the field under "Value"
- Click "OK"
- Click "Save"
- Click "Apply" to apply the new filter to the data.
- Click "Close" to close the dialog box.

All the data with the TCP destination port of 80 will show.

In order to use a variation of the new filter to view FTP traffic on TCP destination port 21, click inside the filter tool bar text box and change 80 to 21, and then click Apply.

To remove a display filter and view the data unfiltered in Ethereal, in the filter tool bar click Clear[5].

[4] Sharpe, Richard. http://www.ethereal.com/docs/user-guide-sp/#ChUseFilterToolbarSection

[5] For more information on using Display filters see:
Sharpe, Richard. http://www.ethereal.com/docs/user-guide-sp/#ChUseFilterToolbarSection; and, Orebaugh, Angela. Ethereal Packet Sniffing. Syngress, 2004. Pg. 172

© SANS Institute 2000 - 2005, Author retains full rights.

Results of Viewing the Captured Data with Display Filters

For purposes of this test, the destination is the server from which the log files were examined with IP address of 66.92.xx.xx unless otherwise noted.

The port filter used for TCP traffic in this test was tcp.port == xy in order to obtain traffic from either the source or destination port. The same was done with UDP traffic udp.port == xy.

These are the results of an eight day traffic capture, Feb. 9, 2005, 21:52 hrs through Feb 18, 2005, 10:23 hrs.

Results
4297 packets were captured during this test. 76.03 % of the data was from TCP; 21.6% was from UDP; 1.56% was from ICMP, and 0.81% was from ARP.

Only the two highest traffic protocols of TCP & UDP are listed in this paper.
In TCP, the two largest packet captures each have a graph; and in addition, the largest UDP packet capture also has a graph.

Category 1, TCP Port[6] traffic:
tcp.port == 21 Protocol FTP as destination port with one exception as the source port, 17 packets total. Possible Ramen Worm[7].

tcp.port == 22 SSH destination port, 32 packets, source ports varied, but nearly all were high five digit numbers.

tcp.port == 23 Telnet no traffic.
tcp.port == 25 SMTP traffic as destination port, 22 packets, from five source IP's, source ports varied, but nearly all were high five digit numbers.

tcp.port == 42 Name Server as destination port, six packets from port 80 HTTP, one frame from port 3389 MS Term Serv, the remaining frame from port 6000 Xwindows, total of eight packets of traffic.

tcp.port == 53 DNS traffic as destination port, 4 packets.
tcp.port == 79 Finger no traffic.
tcp.port == 80 HTTP destination port, 21 packets, one IP tried eight times from source port 4293.

tcp.port == 110 POP3 no traffic.

---

[6] Scrambray, Joel. Hacking Exposed: Network Security Secrets & Solutions,2nd Edition. Berkeley, 2001. Pages 658 -660.
[7] Orebaugh, Angela. Ethereal Packet Sniffing. Syngress, 2004. Pg. 371.

© SANS Institute 2000 - 2005 Author retains full rights.

tcp.port == 111 Sun RPC traffic destination port, 3 consecutive packets.

tcp.port == 135 NT RPC or DCE endpoint resolution destination port, 583 packets, from multiple IP's, with several blocks of same source IP, but different source ports. This port shows up under its own Information heading of "epmap" in Ethereal.

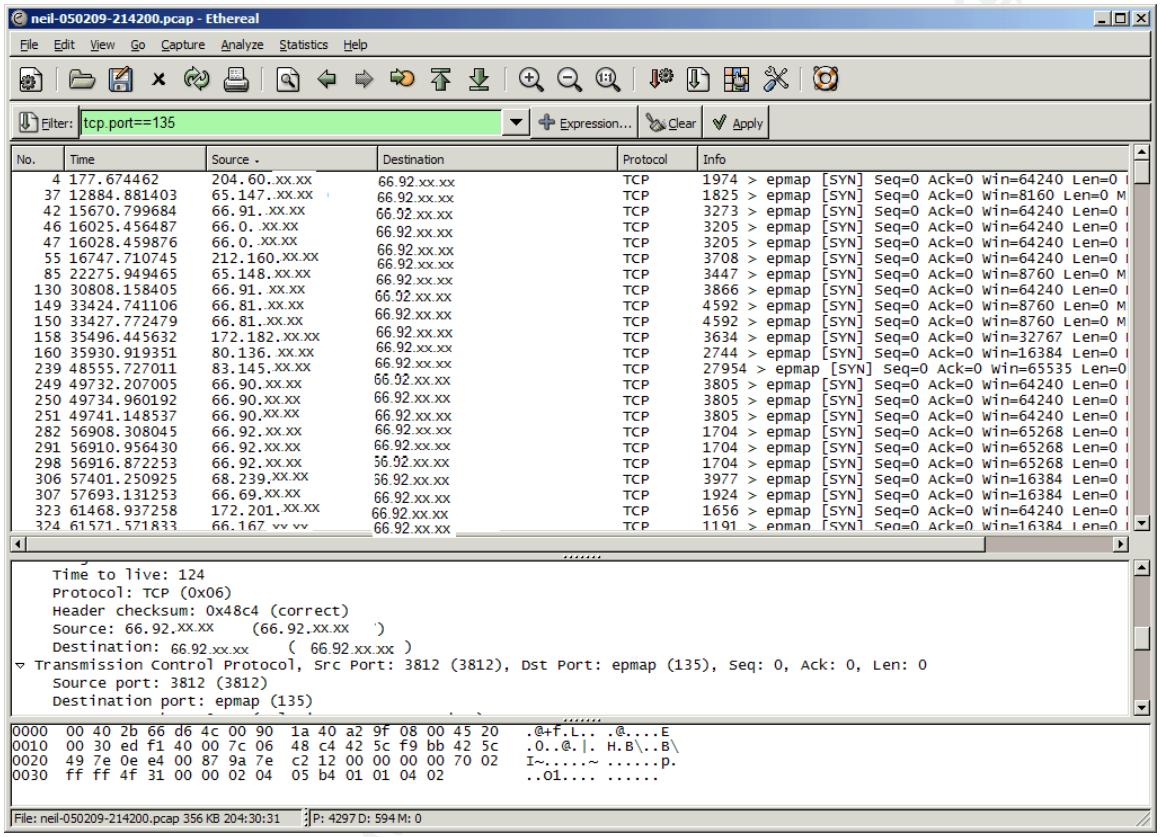

Figure 2. TCP 135 NT RPC

tcp.port == 137 NetBIOS[8] Name Service port, no traffic.

tcp.port == 139 NETBIOS Session Service destination port, 239 packets, many different source IP's, some same source IP's for two or three scans.
This shows up under its own information heading of "netbios-ssn" in Ethereal.

tcp.port == 143 Imap no traffic.
tcp.port == 162 SNMP-Trap two packets.
tcp.port == 328 Common Name unassigned[9], seven packets as destination port with a source port of 80 HTTP, 3 different groupings of source IP's.

tcp.port == 389 ldap two packets.
tcp.port == 443 HTTPS, SSL, 4 packets.

[8] IANA, http://www.iana.org/assignments/port-numbers
[9] IANA, http://www.iana.org/assignments/port-numbers

tcp.port == 445 Microsoft SMB destination port, 1524 packets, varied source IP's. 114 Packets with one source IP that tried a large variety of ports. Another source IP scanned for 103 packets, and another for 91 packets.

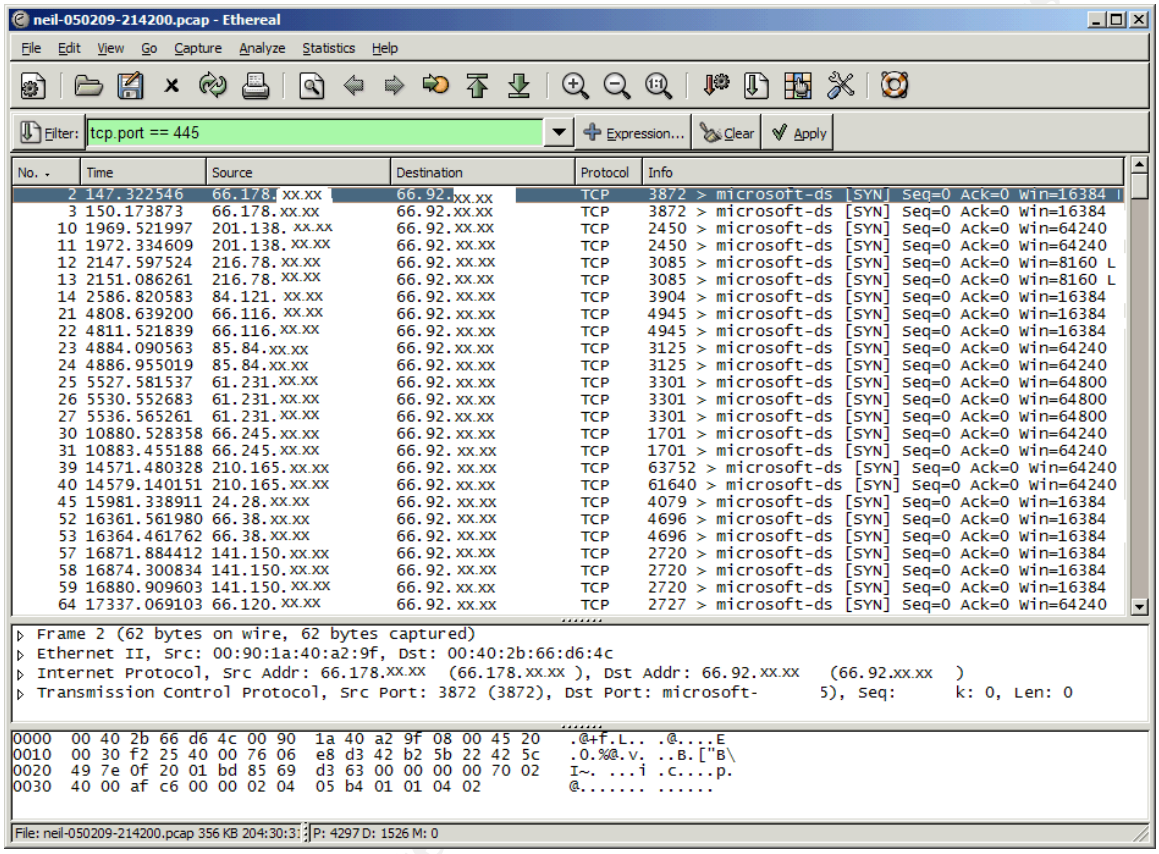

Figure 3. TCP 445 Microsoft SMB

445 Microsoft SMB destination port shows up under its own Information heading of "Microsoft-ds" in Ethereal.

tcp.port == 515 printer destination port, 3 consecutive packets, possible Ramen[10].
tcp.port == 554 Real Time Stream Control Protocol.
Destination port, 23 packets, from 12 different source IP's.
tcp.port == 943 unassigned destination port, 17 packets, 5 incremental source IP's, all source ports are port 80 HTTP.

tcp.port == 1025 network blackjack destination port, 21 packets, 8 source IP's.
tcp.port == 1080 Socks dozen packets.
tcp.port == 1313 bmc-patroldb Source port, destination port 445 MS SMB, 1 frame.

[10] Orebaugh, Angela. Ethereal Packet Sniffing. Syngress, 2004. Pg. 371.

tcp.port == 1243 SubSeven port, destination or source port, 4 packets, from two different IP's. Two packets with source port of 1243 went to destination port of 445 MS NetBIOS. Other two packets had source port of 4132 NUTS Daemon is default and destination port of 1243.

tcp.port == 1433 MS SQL destination port, 96 packets, varied IP's, one block of traffic had same source IP for 9 packets, using 5 different source ports.
This shows up under its own Information heading of "ms-sql-s" in Ethereal.

tcp.port == 1434 Microsoft SQL Monitor no traffic.
tcp.port == 2301 Compaq-web three packets source port, destination port 445 MS SMB.
tcp.port == 3300 bmc-patrol-agent two frame source port, destination port 445 MS SMB.
tcp.port == 2745 Bagel back door[11], destination port, nine packets.

tcp.port == 3127 My Doom back door[12], 12 packets, from five IP addresses, two had source ports of 3127, and the remaining had destination ports of 3127.
The two packets with source port of 3127 had a destination port of 445 MS SMB destination.

tcp.port == 3198 My Doom back door[13], no traffic.
tcp.port == 3306 MySQL destination port 37 packets, 21 source IP's, SQL worm. MySQL UDF Worm[14].

tcp.port == 3351 ssql source port, five packets, two groupings, consecutive packets, destination port 445.

tcp.port == 3389 MS Term Serv, both source and destination port.
Destination port, nine packets, 5 source IP's.
Source port, six packets, 3 source IP's.

tcp.port == 4001 Cisco-Mgmt source port, two consecutive fames, destination port 445 MS SMB.
tcp.port == 4045 NFS-Lockd two consecutive packets source port, destination port 135 NT RPC.

tcp.port == 4899 Radmin remote access port[15], destination port, 109 packets, 12 packets from same IP address.

[11] Nazario, Jose. http://ims.eecs.umich.edu/worm_report/
[12] Nazario, Jose. http://ims.eecs.umich.edu/worm_report/
[13] Nazario, Jose. http://ims.eecs.umich.edu/worm_report/
[14] US-Cert, http://www.us-cert.gov/current/current_activity.html#MySQLUDF
[15] Scheidell, Michael. http://www.securityfocus.com/archive/1/290099/2002-09-01/2002-09-07/0

© SANS Institute 2000 - 2005, Author retains full rights.

tcp.port == 5631 PCAnywhere no traffic.
tcp.port == 5800 VNC no traffic.
tcp.port == 5900 RealVNC no traffic.

tcp.port == 6000 XWindows, source port for 12 packets of traffic; six had destination port of 1433 MS SQL; two had destination of port 4899 Radmin, one a destination port of 42 Name Server; one a destination port of 3389 MS Term Serv; one a destination port of 6129.

tcp.port == 6667 IRC clients, common Trojan port, 7 packets, all from same IP address. Variety of Exploits:

W32.gaobot.cii
Backdoor.lateda.b
Protoride.b
Backdoor.Alcani
W32.spybot.dnb
W32.Randex.ATS
W32.korgo.a
SubSeven[16]

tcp.port == 6711 - 6713 SubSeven[17] ports, no traffic.
tcp.port == 6776 SubSeven port[18], no traffic.
tcp.port == 6777 Bagle.A port[19], no traffic.

tcp.port == 7000 Exploit port variety of attacks. Traffic 53 packets source port and 29 packets destination port.

Exploit Translation Server, Kazimas,
Remote Grab, SubSeven,
SubSeven 2.1 Gold[20]

tcp.port == 8000 Web applications, destination port, 5 packets.
tcp.port == 8080 Web applications, 7 packets (5 from same IP address), and destination port.

tcp.port == 8420 Unassigned[21] port as a common name, destination port for all 13 packets, 8 different source IP's (sometimes the same source IP will try the same attack several times).

tcp.port == 12345 NetBus[22] No traffic.

---

[16] http://www.doshelp.com/Ports/6667.htm
[17] SANS Institute. Track 1 - Internet Security Technologies Jan 2004. Pg 166.
[18] SANS Institute. Track 1 - Internet Security Technologies Jan 2004. Pg 166.
[19] Nazario, Jose. http://ims.eecs.umich.edu/worm_report/
[20] http://www.blackcode.com/trojans/ports.php?port=7000
[21] IANA, http://www.iana.org/assignments/port-numbers
[22] SANS Institute. Track 1 - Internet Security Technologies Jan 2004. Pg. 162.

© SANS Institute 2000 - 2005 Author retains full rights.

tcp.port == 12346 NetBus[23] No traffic.
tcp.port == 27374 default SubSeven[24] & Ramen[25] port no traffic.
tcp.port == 31337 Back Orifice[26] no traffic.
tcp.port == 32771 rcp-solaris no traffic.
tcp.port == 43188 reachout no traffic.

tcp.port == 50736 Unknown[27] common port, destination port for all 17 packets (10 different source IP's, one IP tried same attack six consecutive times, some IP's repeated non-consecutive traffic. Eleven packets from port 80 HTTP, three packets from port 7000, one from port 7777, one from port 21 (FTP), one from port 4000.

tcp.port == 65301 PCAnywhere – def no traffic
tcp.port == 65535 or 0, Sons of Jackal[28] no traffic

Category 2, UDP port traffic
udp.port == 53 DNS[29] no traffic
udp.port == 69 TFTP[30] no traffic
udp.port == 135 DCE endpoint resolution no traffic

udp.port == 137 NetBIOS Name Server[31] destination port, 577 packets, largest block of same source IP is 20 packets, the packets were only consecutive a few at a time. It appears this Source IP retried its attack every thousand packets or so.

The scan has the characteristics of older port scanning worm network.vbs[32] and its derivatives.

[23] SANS Institute. Track 1 - Internet Security Technologies Jan 2004. Pg. 162.
[24] SANS Institute. Track 1 - Internet Security Technologies Jan 2004. Pg. 163
[25] Orebaugh, Angela. Ethereal Packet Sniffing. Syngress, 2004. P. 371
[26] SANS Institute. Track 1 - Internet Security Technologies Jan 2004. Pg. 167.
[27] IANA, http://www.iana.org/assignments/port-numbers
[28] SANS Institute. Track 1 - Internet Security Technologies Jan 2004. Pg. 173.
[29] IANA, http://www.iana.org/assignments/port-numbers
[30] IANA, http://www.iana.org/assignments/port-numbers
[31] IANA, http://www.iana.org/assignments/port-numbers
[32] Alexander, Bryce. http://www.sans.org/resources/idfaq/port_137.php

© SANS Institute 2000 - 2005, Author retains full rights.

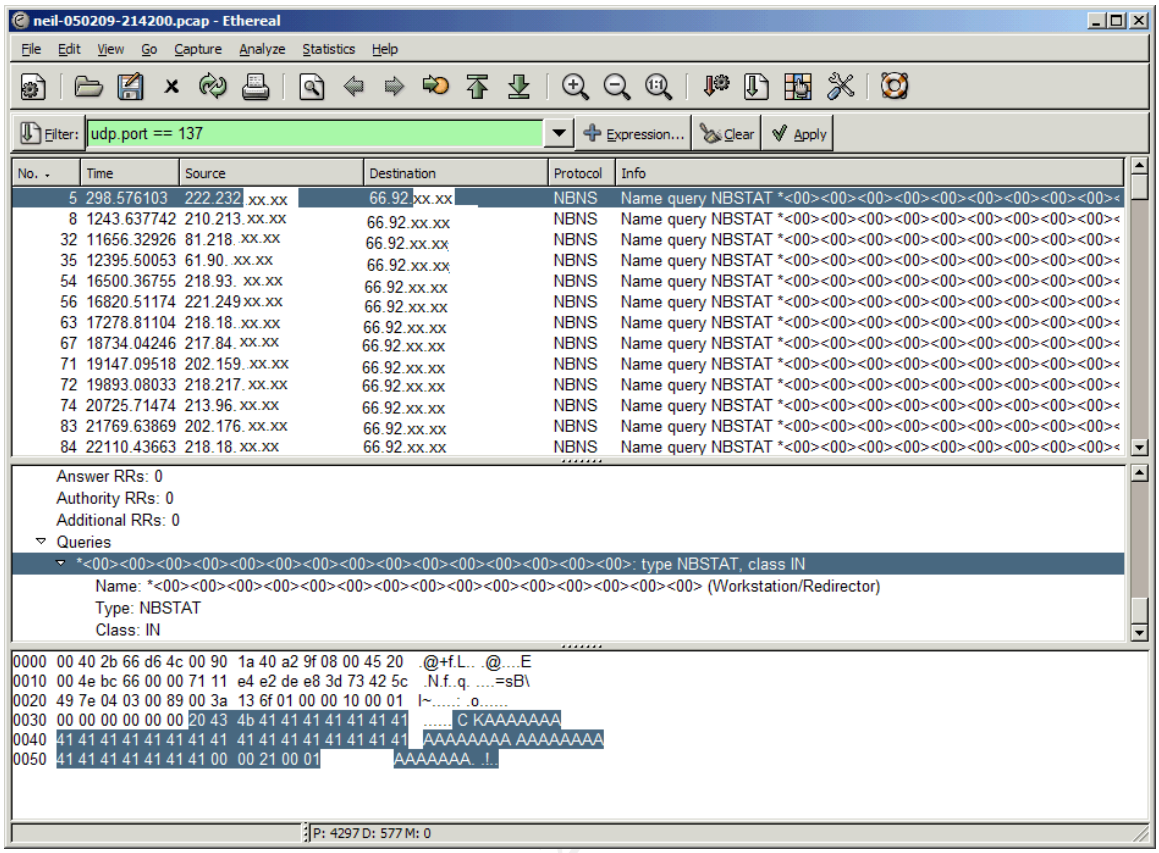

Figure 4. UDP 137 NetBIOS Name Server

The Ethereal Summary window Information column per packet contains:
Name query NBSTAT
*<00><00><00><00><00><00><00><00><00><00><00><00><00><00><00><00>

The below, is from the Ethereal Data View Window of packet 5; however, the characteristic "CKAAAAAAA" repeats itself in each packet.

```
0000   00 40 2b 66 d6 4c 00 90 1a 40 a2 9f 08 00 45 20  .@+f.L...@....E
0010   00 4e e5 f5 00 00 76 11 49 75 53 2b 36 0f 42 5c  .N....v.IuS+6.B\
0020   49 7e 4c c3 00 89 00 3a 5d ff 00 d1 00 10 00 01  I~L.....:].......
0030   00 00 00 00 00 00 20 43 4b 41 41 41 41 41 41 41  ..... CKAAAAAAA
0040   41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41
AAAAAAAAAAAAAAAA
0050   41 41 41 41 41 41 41 00 00 21 00 01              AAAAAAA..!..
```

Verified NetBIOS traffic is only to destination port by using "udp.dstport" in Ethereal display filter as well. This traffic shows up under its own Protocol heading of NBNS in Ethereal.

udp.port == 139 NetBIOS Session Service[33] no traffic.

[33] IANA, http://www.iana.org/assignments/port-numbers

udp.port == 445 Microsoft SMB[34] no traffic.
udp.port == 1433 Microsoft-SQL-Server[35] no traffic.
udp.port == 1434 Microsoft SQL Monitor[36], destination port, 69 packets.

udp.port == 31337 Backorifice[37] no traffic.
udp.port == 27374 default SubSeven[38] no traffic.

A summary of all the protocols found by Ethereal by the menu bar, Statistics - Protocol Hierarchy graph:

Ethereal: Protocol Hierarchy Statistics

Protocol	% Packets	Packets	Bytes	Mbit/s	End Packets	End Bytes	End Mbit/s
Frame	100.00%	4297	512690	0.000	0	0	0.000
Ethernet	100.00%	4297	512690	0.000	0	0	0.000
Internet Protocol	99.19%	4262	510590	0.000	0	0	0.000
Internet Control Message Protocol	1.56%	67	5590	0.000	67	5590	0.000
Transmission Control Protocol	76.03%	3267	203442	0.000	3267	203442	0.000
User Datagram Protocol	21.60%	928	301558	0.000	1	922	0.000
NetBIOS Name Service	13.43%	577	53084	0.000	577	53084	0.000
Data	8.12%	349	246630	0.000	349	246630	0.000
RX Protocol	0.02%	1	922	0.000	1	922	0.000
Address Resolution Protocol	0.81%	35	2100	0.000	35	2100	0.000

OK

Figure 5. Protocol Hierarchy Summary Graph

[34] IANA, http://www.iana.org/assignments/port-numbers
[35] IANA, http://www.iana.org/assignments/port-numbers
[36] IANA, http://www.iana.org/assignments/port-numbers
[37] SANS Institute. Track 1 - Internet Security Technologies Jan 2004. Pg. 167.
[38] SANS Institute. Track 1 - Internet Security Technologies Jan 2004. Pg. 166.

Conclusion

Traffic usually repeated itself from the same IP and same source port two to three consecutive times against a specific destination port. For example, source IP 218.30.21.xxx, TCP source port http, destination port 328 showed up two times in a row twice: frames 1177 & 1178; and 1182 & 1183. One possible explanation of this may be that the attacker hopes to protect against dropped or timed-out traffic per connection. For example: the attacker attempts a connection, SYN, several times using the same method and hopes that one attempt should be successful with an SYN/ACK.

Many of the attacks were against recent well published vulnerabilities. These were mostly Microsoft with some Linux as well. However, any system is vulnerable to attack.

In Ethereal display filter, the query tcp.flags.syn==1&&tcp.flags.ack==1[39] showed three or more attempted connections against open ports:
7000
50736
8420
328
943

One interpretation of this attack style is:

> The attackers seem to have tricks to get through firewalls, including sending various types of malformed packets. A firewall may be configured not to let new connections in, so an attacker will set the "ACK" flag to try
> making any such firewalls along the way think that the packet is part of an open TCP connection. The packets erroneously show up in your analysis as belonging to open connections for this same reason[40].

However, in this case, the IP did not send out a single packet. "One cannot have a working TCP connection without data flowing in both directions.[41]"

"The other thing that some naively-configured firewalls do is accept traffic from well-known ports. For example, your firewall may accept traffic from port 3389 so that you can successfully connect to somebody else's Microsoft Terminal Server[42]." This was partially seen in the TCP traffic captured and noted earlier going to port 3389. "Because the attackers have

[39] Orebaugh, Angela. Ethereal Packet Sniffing. Syngress, 2004. Pg. 355.
[40] Madden, Patrick. E-mail. March 2005.
[41] Madden, Patrick. E-mail. March 2005.
[42] Madden, Patrick. E-mail. March 2005.

© SANS Institute 2000 - 2005 Author retains full rights.

complete control over source port, they try using source port numbers that a naively-configured firewall will allow through, thinking that it's a connection the user opened.[43]" The main way to counter the above threat would be to place the server behind a firewall on a private network (10.x.x.x for example), and to have the trusted users or machines connect to the servers via a gatekeeper service, such as a VPN (Virtual Private Network) or a TCP connection tunneled through SSH.

I also expected to see incremental increasing source IP addresses due to IP spoofing. This was not the case.

One of the disappointments of Ethereal was that I could not find a simple way to get an automated count of the exact number of packets used by a specific protocol and port, e.g., TCP port 7000. I am not familiar with Tcpdump or WinDump, and was seeking a way to get a text file from Ethereal that could be imported into an application like MS Excel and be manipulated. It took some time to play with all text exporting options to find what I was seeking. I include the following to save a person the same effort in the future.

While reading Display file in Ethereal

- click File
- Export as "Plain Text file",
- uncheck Packet Details,
- type in path and name of file,
- click OK.

When open this text file in Excel,

- choose Delimited,
- with a delimiter of a space.

[43] Madden, Patrick. E-mail. March 2005.

© SANS Institute 2000 - 2005, Author retains full rights.

References

1. Alexander, Bryce. "Intrusion Detection FAQ: Port 137 Scan." SANS. May 10, 2000. URL: http://www.sans.org/resources/idfaq/port_137.php

2. "Show Trojan By Port Number." Blackcode Trojan Library. March 12, 2005. URL: http://www.blackcode.com/trojans/ports.php

3. "Known Trojans/Worms for Port: 6667." Doshelp.com Known Trojans/Worms for Port: 6667. Feb 2005. URL: http://www.doshelp.com/Ports/6667.htm

4. Hazeleger, Dick. "Packet Sniffing: A 'Crash course'." 2003. URL: http://security.hazeleger.net/

5. IANA, "Port Numbers." February 23, 2005. URL: http://www.iana.org/assignments/port-numbers

6. Nazario, Jose, and Evan Cooke. "Increased Bagle and MyDoom Backdoor Scanning Activity." Arbor Networks. March 26, 2004. URL: http://ims.eecs.umich.edu/worm_report/

7. Madden, Patrick. "Re: Preliminary results very rough." E-mail reply to Neil Orlando from computer security consultant giving feed back on findings after data analysis by author for this practical paper. March 7, 2005. 5:38pm.

8. OPENXTRA Limited. "Ethereal - Setting a Display Filter." URL: http://www.openxtra.co.uk/support/howto/ethereal-display-filter.php

9. OPENXTRA Limited. "Getting Started with Ethereal." May 2003. URL: http://www.openxtra.co.uk/support/documentation/ethereal-getting-started-guide.pdf

10. Orebaugh, Angela, Greg Morris, Ed Warnicke, and Gilbert Ramirez. Ethereal Packet Sniffing. Rockland: Syngress, 2004.

11. SANS Institute. Track 1 - Internet Security Technologies, Volume 1.3. SANS Press , Jan 2004.

12. Scheidell, Michael. "SECNAP Security Alert: Radmin Default install options vulnerability." SecurityFocus. September 2, 2002. URL: http://www.securityfocus.com/archive/1/290099/2002-09-01/2002-09-

© SANS Institute 2000 - 2005 Author retains full rights.

07/0.

13. Scambray, Joel, Stuart McClure, George Kurtz. Hacking Exposed: Network Security Secrets & Solutions, 2nd Edition. Berkeley: Osbourne/McGraw-Hill, 2001.

14. Sharpe, Richard, Ed Warnicke, Ulf Lamping. "Ethereal User's Guide." 2004. URL: http://www.ethereal.com/docs/user-guide/

15. US-CERT, "US-Cert Current Activity." March 11, 2005 15:05:14 EST. URL: http://www.us-cert.gov/current/current_activity.html

© SANS Institute 2000 - 2005, Author retains full rights.

Last Updated: May 25th, 2013

Upcoming SANS Training

Click Here for a full list of all Upcoming SANS Events by Location

SANS Malaysia @ MCMC 2013	Cyberjaya, MY	Jun 03, 2013 - Jun 08, 2013	Live Event
SANS Pen Test Berlin 2013	Berlin, DE	Jun 03, 2013 - Jun 08, 2013	Live Event
Industial Control Systems Security Training - Houston	Houston, TXUS	Jun 10, 2013 - Jun 15, 2013	Live Event
Security Impact of IPv6 Summit 2013	Washington, DCUS	Jun 14, 2013 - Jun 16, 2013	Live Event
SANSFIRE 2013	Washington, DCUS	Jun 14, 2013 - Jun 22, 2013	Live Event
SANS Canberra 2013	Canberra, AU	Jul 01, 2013 - Jul 13, 2013	Live Event
Digital Forensics & Incident Response Summit 2013	Austin, TXUS	Jul 09, 2013 - Jul 16, 2013	Live Event
SANS London Summer 2013	London, GB	Jul 09, 2013 - Jul 16, 2013	Live Event
SANS Rocky Mountain 2013	Denver, COUS	Jul 14, 2013 - Jul 20, 2013	Live Event
SANS Mumbai 2013	Mumbai, IN	Jul 22, 2013 - Jul 27, 2013	Live Event
SEC528 SANS Training Program for the CompTIA® New Advanced Security Practitioner (CASP) Certification	Washington, DCUS	Jul 22, 2013 - Jul 26, 2013	Live Event
SANS San Francisco 2013	San Francisco, CAUS	Jul 29, 2013 - Aug 03, 2013	Live Event
SANS SEC 560: Network Penetration Testing @ Bangalore 2013	Bangalore, IN	Aug 05, 2013 - Aug 10, 2013	Live Event
SANS Boston 2013	Boston, MAUS	Aug 05, 2013 - Aug 10, 2013	Live Event
Critical Security Controls Summit	Washington, DCUS	Aug 12, 2013 - Aug 18, 2013	Live Event
Industrial Control Systems Security Training - DC	Washington, DCUS	Aug 12, 2013 - Aug 16, 2013	Live Event
SANS Thailand 2013	Bangkok, TH	Aug 19, 2013 - Aug 31, 2013	Live Event
SANS Virginia Beach 2013	Virginia Beach, VAUS	Aug 19, 2013 - Aug 30, 2013	Live Event
Mobile Device Security Summit 2013	OnlineCAUS	May 30, 2013 - Jun 06, 2013	Live Event
SANS OnDemand	Books & MP3s OnlyUS	Anytime	Self Paced