DECEMBER 1, 2024

ATTACK ANALYSIS

LAB 04

ANTHONY CAMPBELL (STUDENT)
Professor McCulley
University of Texas at San Antonio IS 3523

Contents

Introduction

My task is to investigate a potential breach using the forensic artifacts provided. The analysis involves examining a packet capture (PCAP) file, an application log, and a security log to identify the attack vector, scope of the compromise, and any malicious activities. Utilizing tools like Wireshark and Network Miner, along with detailed log analysis, I aim to reconstruct the attack timeline, determine the root cause, and provide actionable recommendations to mitigate future risks. The findings from this investigation will directly support the organization's incident response efforts.

Forensic Artifact Details

Packet Capture File

File Name: Maybe Hacked Box—Capture.pcapng

Size: 1.9 MB

Date and Time of Capture: 2017-07-08 14:09:32 | Elapsed: 00:04:12

Description: Contains network traffic captured during the suspected breach period.

Application Log

File Name: Maybe Hacked Box—Application Log.evt

Size: 18.7 KB

Description: Logs generated by the affected applications, potentially revealing unauthorized access or suspicious behaviors.

Security Log

File Name: Maybe Hacked Box—Security Log.evt

Size: 39.1 KB

Description: Logs detailing security events, including user authentications, permissions changes, and alerts.

Packet Capture Findings

NetworkMiner

The network traffic analysis between 192.168.134.129 (Linux) and 192.168.134.132 (Windows) reveals suspicious activity suggesting a potential breach. 192.168.134.129 sent 1,573 packets (1,712,909 bytes) and received 585 packets (156,369 bytes), while 192.168.134.132 sent 591 packets (156,945 bytes) and received 1,582 packets (1,714,025 bytes). The traffic analysis shows multiple connections on TCP ports 445, 1000, 135, and 139 between the two devices.

Ports 445, 135, and 139 are reserved for services such as Microsoft-DS, LOC-SRV, and NetBIOS-SS, respectively. However, port 1000 is not reserved and can be exploited by backdoor trojans, as can port 139. The presence of open port 29922 on 192.168.134.129 is particularly concerning, as it is not reserved and is likely being used for malicious purposes. Given these open ports and the unusual traffic patterns, including potential encrypted communication to evade detection, it is apparent that a breach has occurred. (Speed Guide, n.d.) (see Appendix 1: General NetworkMiner)

The sessions between the device gives use some timestamps to investigate further:

24-hour format	12-hour format
19:09:40 UTC to 19:13:39 UTC	7:09:40 PM UTC to 7:13:39 PM UTC
19:09:40 UTC to 19:09:40 UTC	7:09:40 PM UTC to 7:09:40 PM UTC
19:13:22 UTC to 19:13:22 UTC	7:13:22 PM UTC to 7:13:22 PM UTC
19:13:26 UTC to 19:13:26 UTC	7:13:26 PM UTC to 7:13:26 PM UTC
19:10:49 UTC to 19:12:19 UTC	7:10:49 PM UTC to 7:12:19 PM UTC
19:13:22 UTC to 19:13:22 UTC	7:13:22 PM UTC to 7:13:22 PM UTC

Additionally, when we analyze the “Parameters” tab we can see some odd SMB (Server Message Block) interactions. SMB typically uses ports 139 and 445 for communication. However, we've observed multiple instances of SMB traffic on port 41254, which is not a standard port for this protocol. This anomaly, where SMB communication occurs between a Linux host and a Windows machine using port 41254 instead of the standard 445, could indicate potential network misconfigurations or suspicious activity. It's essential to monitor such instances closely, as they may be an indication of attempts to evade detection or bypass network security controls. (Techa, 2024) (see Appendix 2: NetworkMiner Parameter information)

Wireshark

With our malicious connection timestamps, we can filter some of our packets out with a filter. We must account for the fact that the timestamps are initially in UTC, while in Wireshark it presents the timestamps in current time zone time (CDT currently). With our filter we cut our packets to peruse down to 79.1%. This will help with timeliness, filter: *frame.time >= "2017-07-08 14:09:40" && frame.time <= "2017-07-08 14:13:22"*

Perusing in the first 38 packets alone, numbered 26 to 63, further confirms the suspicions we started to find in NetworkMiner. We have SMB Session Setup attempts, which include multiple failed logon responses. Notably, packet 35 shows a "STATUS_MORE_PROCESSING_REQUIRED" error after an NTLMSSP_NEGOTIATE request, followed by a "STATUS_LOGON_FAILURE" in packet 37, indicating an authentication issue. Additionally, there are several "NT Create AndX Response" errors, such as in packet 43, where the response indicates "STATUS_ACCESS_DENIED." These failed authentication and access attempts suggest potential unauthorized access attempts or misconfigurations. The presence of repeated errors, including "Provider rejection" and "Fault: nca_s_fault_ndr" in subsequent packets, provides further evidence that these interactions may involve malicious activity or network misconfigurations. The SMB issues continue in some ways. These failures are abnormal and warrant further investigation. (Kelley, 2023) (see Appendix 3: Wireshark Packets 26-63 (SMB Suspicions))

The next analysis block of packets I chose are frames 106-135, about 35 packets. The packet sequence begins with an attempt to establish SMB communication between the two devices (192.168.134.129 and 192.168.134.132), involving several bind requests and responses, but with various issues, such as provider rejections. These exchanges indicate that the devices are negotiating context and protocol versions for the SMB session. The devices seem to struggle initially with ensuring proper context acceptance and completing the bind process. Despite this, they continue with the SMB protocol, with multiple requests for data reading and writing, albeit encountering various responses and packet exchanges.

After these SMB attempts, the devices eventually seem to successfully establish a working SMB session. Following this resolution, they proceed to use TCP for data transfer, with a sequence of TCP packets indicating that the session has moved from SMB-specific operations to standard TCP-based communication. The devices appear to have successfully overcome the SMB communication hurdles and transitioned to TCP for the subsequent data transfer, implying that the SMB connection was established first before switching to TCP for handling larger payloads. The data transfer proceeds in a continuous flow of ACK packets, confirming the established connection and successful data exchange between the two devices. (see Appendix 4: Wireshark Packets 106-135 (SMB, TCP Begins)

TCP connections, Potentially Noteworthy incidents inbetween

From frame 118 to 1648 we have a stream of TCP connections. Here are some notable frames that stand out, with a small summary:

Window Full / ZeroWindow packets (packets 254, 642): These indicate that the receiver's buffer is full, and the sender should stop sending data temporarily.

Window Update packets (packets 256, 274, 595): These packets signal an update in the available window size, indicating that the receiver's buffer has more space available. In response to the Window Full / ZeroWindow it seems the bad actor(s) manipulate the window size, increasing the amount of data and facilitating larger-scale exfiltration of stolen data without alerting.

TCP Flow Control (packets 254, 642, 645): These packets help control the flow of data to avoid congestion and buffer overflow on either side of the connection. Manipulating flow control can create opportunities for data exfiltration, with pauses and adjustments used to avoid detecting all while siphoning small amounts of data overtime.

The TCO connections during the timestamps in the graph above are certainly when some data exfiltration occurs, with the packets above providing further evidence. (Pal, 2022)

Further Packet Notes

The captured network traffic shows multiple signs of potential malicious activity, including ARP spoofing, suspicious DNS queries, and frequent LLMNR and multicast traffic. These behaviors suggest attempts at network reconnaissance, such as identifying live hosts or exploiting local name resolution protocols for man-in-the-middle attacks. Additionally, the presence of unusual TCP communication, including frequent data exchanges and FIN flags, points to further data exfiltration. ICMP and UDP traffic also raise concerns, indicating potential DoS attempts or probing for vulnerable devices. Overall, these patterns are consistent with a targeted attack aimed at disrupting the network or gaining unauthorized access. (Adams, 2023) (Imperva, n.d.) (Gorman, 2023)

Conclusion

The forensic evidence strongly suggests that the devices involved, 192.168.134.129 and 192.168.134.132, have been compromised and there are likely other compromised devices on the network. Suspicious traffic patterns, unusual port activity, failed SMB authentication attempts, and potential encrypted communications point to a breach. Manipulation of TCP flow control and signs of data exfiltration suggest that bad actors may have been siphoning information without detection. Given the timeframe of 2017, I would even gamble to say from my research that the EternalBlue exploit may have been used as a vector for the attack, which was a prominent threat starting in that period, and even today a number of devices remain unpatched. The combination of these indicators confirms a significant security breach and data has been stolen.

Additional Note

I know I did not include evidence from the security and application event logs. I had perused them but I spent too much time trying and testing other things ‘for fun’ (like RITA, zeet, and Hayabusa that didn’t work for these caps for whatever reason). I may continue to write on my paper and submit ‘late’ but I feel confident in my findings with NetworkMiner and Wireshark alone. Thank you.

Bibliography

Adams, H. (2023, September 25). *LLMNR Poisoning and How to Prevent It in Active Directory*. Retrieved from TCM Security: https://tcm-sec.com/llmnr-poisoning-and-how-to-prevent-it/

Gorman, B. (2023, September 22). *EternalBlue Exploit: What Is It and Is It Still a Threat?* . Retrieved from AVG: https://www.avg.com/en/signal/eternal-blue#

Imperva. (n.d., n.d. n.d.). *DNS SPoofing*. Retrieved from Imperva: https://www.imperva.com/learn/application-security/dns-spoofing/

Kelley, D. (2023, April 19). *How to defend against TCP port 445 and other SMB exploits*. Retrieved from TechTarget: https://www.techtarget.com/searchsecurity/answer/Detecting-and-defending-against-TCP-port-445-attacks

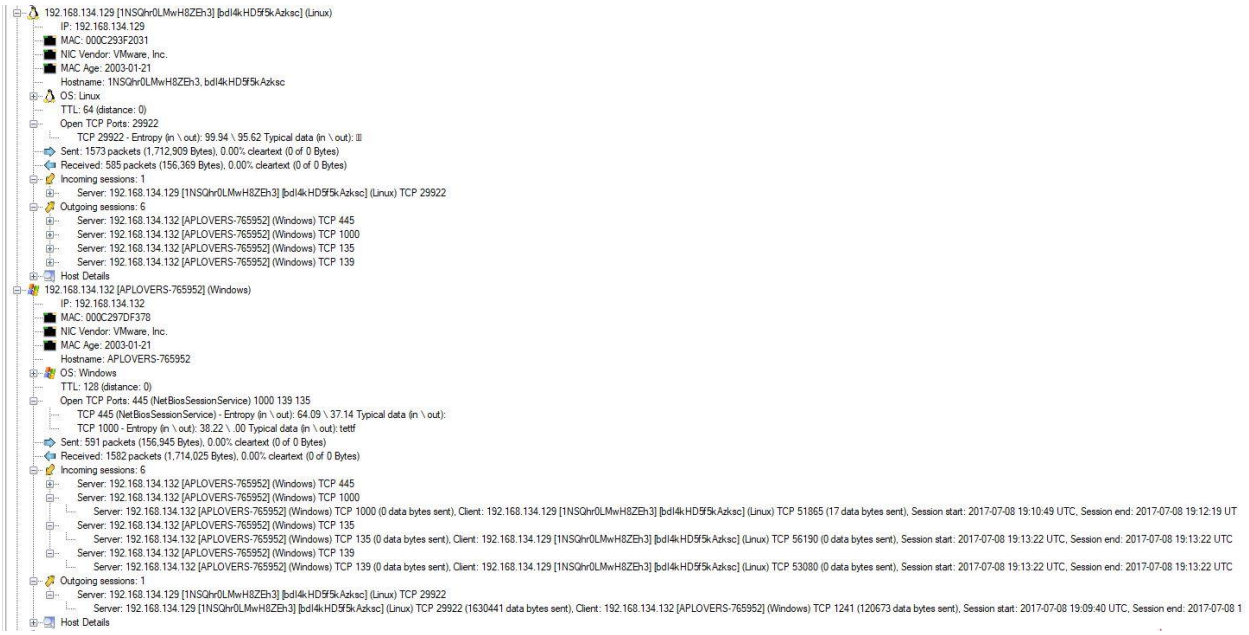
Pal, D. (2022, March 31). *How to: Detect and prevent common data exfiltration attacks*. Retrieved from APNIC: https://blog.apnic.net/2022/03/31/how-to-detect-and-prevent-common-data-exfiltration-attacks/

Speed Guide. (n.d., n.d. n.d.). *Ports Database*. Retrieved November 28, 2024, from https://www.speedguide.net/ports.php

Techa, M. (2024, September 27). *What are SMB Ports, Port 139 and Port 445?* Retrieved from netwrix: https://blog.netwrix.com/smb-port

Appendices

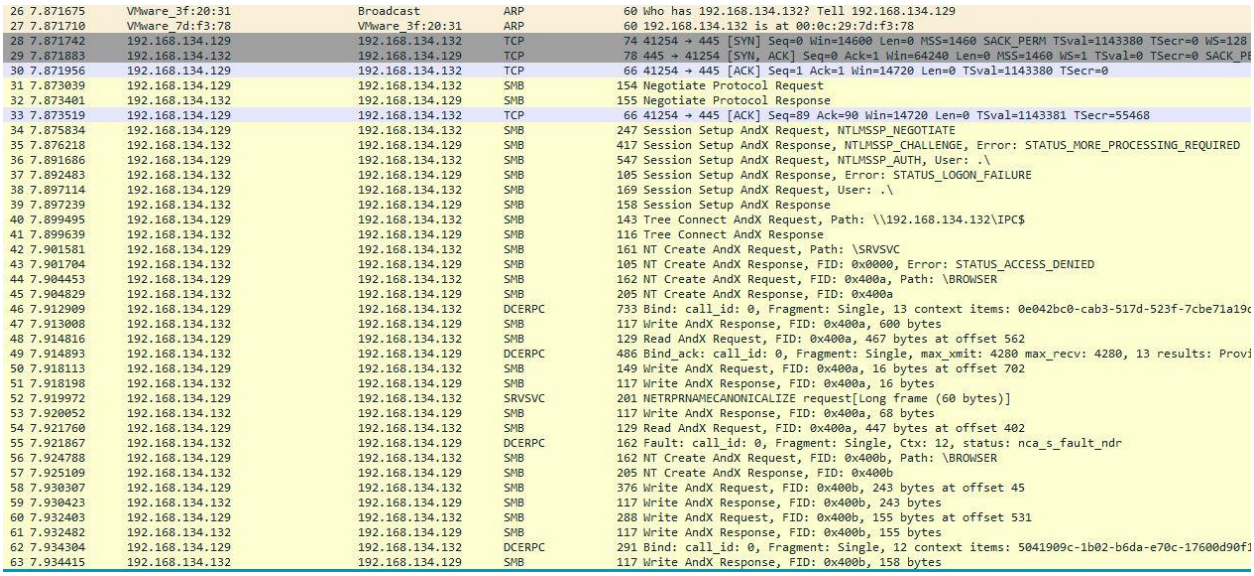
Appendix 1: General NetworkMiner host/session information



Appendix 2: NetworkMiner Parameter information

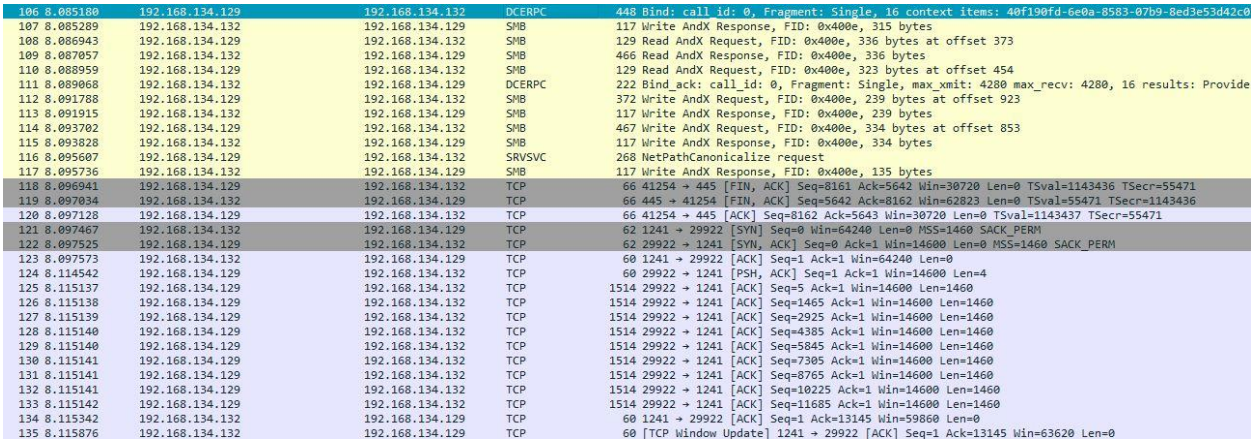
Parameter name	Parameter value	Frame number	Source host	Source port	Destination host	Destination port	Timestamp
SMB Native LAN Manager	Windows 2000 5.0	34	192.168.134.129 [1NSQhr0LMwH8ZEh3] (Linux)	TCP 41254	192.168.134.132 (Windows)	TCP 445	2017-07-08
SMB Native OS	Windows 2000 2195	34	192.168.134.129 [1NSQhr0LMwH8ZEh3] (Linux)	TCP 41254	192.168.134.132 (Windows)	TCP 445	2017-07-08
SMB Native LAN Manager	Windows 2000 LAN Manager	35	192.168.134.132 (Windows)	TCP 445	192.168.134.129 [1NSQhr0LMwH8ZEh3] (Linux)	TCP 41254	2017-07-08
SMB Native OS	Windows 5.1	35	192.168.134.132 (Windows)	TCP 445	192.168.134.129 [1NSQhr0LMwH8ZEh3] (Linux)	TCP 41254	2017-07-08
SMB Native LAN Manager	Windows 2000 5.0	36	192.168.134.129 [1NSQhr0LMwH8ZEh3] (Linux)	TCP 41254	192.168.134.132 (Windows)	TCP 445	2017-07-08
SMB Native OS	Windows 2000 2195	36	192.168.134.129 [1NSQhr0LMwH8ZEh3] (Linux)	TCP 41254	192.168.134.132 (Windows)	TCP 445	2017-07-08
SMB Native LAN Manager	Windows 2000 5.0	38	192.168.134.129 [1NSQhr0LMwH8ZEh3] (Linux)	TCP 41254	192.168.134.132 (Windows)	TCP 445	2017-07-08
SMB Native OS	Windows 2000 2195	38	192.168.134.129 [1NSQhr0LMwH8ZEh3] (Linux)	TCP 41254	192.168.134.132 (Windows)	TCP 445	2017-07-08
SMB Primary Domain	.	38	192.168.134.129 [1NSQhr0LMwH8ZEh3] (Linux)	TCP 41254	192.168.134.132 (Windows)	TCP 445	2017-07-08
SMB Native LAN Manager	Windows 2000 LAN Manager	39	192.168.134.132 (Windows)	TCP 445	192.168.134.129 [1NSQhr0LMwH8ZEh3] (Linux)	TCP 41254	2017-07-08
SMB Native OS	Windows 5.1	39	192.168.134.132 (Windows)	TCP 445	192.168.134.129 [1NSQhr0LMwH8ZEh3] (Linux)	TCP 41254	2017-07-08
SMB Tree Connect AndX Request 23793	\\192.168.134.132\IPC$	40	192.168.134.129 [1NSQhr0LMwH8ZEh3] (Linux)	TCP 41254	192.168.134.132 (Windows)	TCP 445	2017-07-08
SMB NT Create AndX Request 23793	\SRVSVC	42	192.168.134.129 [1NSQhr0LMwH8ZEh3] (Linux)	TCP 41254	192.168.134.132 (Windows)	TCP 445	2017-07-08
SMB NT Create AndX Request 23793	\BROWSER	44	192.168.134.129 [1NSQhr0LMwH8ZEh3] (Linux)	TCP 41254	192.168.134.132 (Windows)	TCP 445	2017-07-08
SMB NT Create AndX Request 23793	\BROWSER	56	192.168.134.129 [1NSQhr0LMwH8ZEh3] [bdI4kHD5f5kAz...	TCP 41254	192.168.134.132 [APLOVERS-765952] (Windows)	TCP 445	2017-07-08
SMB NT Create AndX Request 23793	\SPOOLSS	70	192.168.134.129 [1NSQhr0LMwH8ZEh3] [bdI4kHD5f5kAz...	TCP 41254	192.168.134.132 [APLOVERS-765952] (Windows)	TCP 445	2017-07-08
SMB NT Create AndX Request 23793	\SPOOLSS	88	192.168.134.129 [1NSQhr0LMwH8ZEh3] [bdI4kHD5f5kAz...	TCP 41254	192.168.134.132 [APLOVERS-765952] (Windows)	TCP 445	2017-07-08
SMB NT Create AndX Request 23793	\BROWSER	102	192.168.134.129 [1NSQhr0LMwH8ZEh3] [bdI4kHD5f5kAz...	TCP 41254	192.168.134.132 [APLOVERS-765952] (Windows)	TCP 445	2017-07-08
NetBIOS Query	APLOVERS-765952<20>	1734	192.168.134.132 [APLOVERS-765952] (Windows)	UDP 137	192.168.134.2	UDP 137	2017-07-08
APLOVERS-765952<20>	192.168.134.132	1734	192.168.134.132 [APLOVERS-765952] (Windows)	UDP 137	192.168.134.2	UDP 137	2017-07-08
NetBIOS Query	APLOVERS-765952<20>	1735	192.168.134.132 [APLOVERS-765952] (Windows)	UDP 137	192.168.134.2	UDP 137	2017-07-08
APLOVERS-765952<20>	192.168.134.132	1735	192.168.134.132 [APLOVERS-765952] (Windows)	UDP 137	192.168.134.2	UDP 137	2017-07-08
NetBIOS Query	APLOVERS-765952<20>	1741	192.168.134.132 [APLOVERS-765952] (Windows)	UDP 137	192.168.134.2	UDP 137	2017-07-08
APLOVERS-765952<20>	192.168.134.132	1741	192.168.134.132 [APLOVERS-765952] (Windows)	UDP 137	192.168.134.2	UDP 137	2017-07-08

Appendix 3: Wireshark Packets 26-63 (SMB Suspicions)



26	7.871675	VMware_3f:20:31	Broadcast	ARP	60 Who has 192.168.134.132? Tell 192.168.134.129
27	7.871710	VMware_7d:f3:78	VMware_3f:20:31	ARP	60 192.168.134.132 is at 00:0c:29:7d:f3:78
28	7.871742	192.168.134.129	192.168.134.132	TCP	74 41254 → 445 [SYN] Seq=0 Win=14600 Len=0 MSS=1460 SACK_PERM TSval=1143380 TSecr=0 WS=128
29	7.871883	192.168.134.132	192.168.134.129	TCP	78 445 → 41254 [SYN, ACK] Seq=0 Ack=1 Win=64240 Len=0 MSS=1460 WS=1 TSval=0 TSecr=0 SACK_PE
30	7.871956	192.168.134.129	192.168.134.132	TCP	66 41254 → 445 [ACK] Seq=1 Ack=1 Win=14720 Len=0 TSval=1143380 TSecr=0
31	7.873039	192.168.134.129	192.168.134.132	SMB	154 Negotiate Protocol Request
32	7.873401	192.168.134.132	192.168.134.129	SMB	155 Negotiate Protocol Response
33	7.873519	192.168.134.129	192.168.134.132	TCP	66 41254 → 445 [ACK] Seq=89 Ack=90 Win=14720 Len=0 TSval=1143381 TSecr=55468
34	7.875834	192.168.134.129	192.168.134.132	SMB	247 Session Setup AndX Request, NTLMSSP_NEGOTIATE
35	7.876218	192.168.134.132	192.168.134.129	SMB	417 Session Setup AndX Response, NTLMSSP_CHALLENGE, Error: STATUS_MORE_PROCESSING_REQUIRED
36	7.891686	192.168.134.129	192.168.134.132	SMB	547 Session Setup AndX Request, NTLMSSP_AUTH, User: .\
37	7.892483	192.168.134.132	192.168.134.129	SMB	105 Session Setup AndX Response, Error: STATUS_LOGON_FAILURE
38	7.897114	192.168.134.129	192.168.134.132	SMB	169 Session Setup AndX Request, User: .\
39	7.897239	192.168.134.132	192.168.134.129	SMB	158 Session Setup AndX Response
40	7.899495	192.168.134.129	192.168.134.132	SMB	143 Tree Connect AndX Request, Path: \\192.168.134.132\IPC$
41	7.899639	192.168.134.132	192.168.134.129	SMB	116 Tree Connect AndX Response
42	7.901581	192.168.134.129	192.168.134.132	SMB	161 NT Create AndX Request, Path: \SRVSVC
43	7.901704	192.168.134.132	192.168.134.129	SMB	105 NT Create AndX Response, FID: 0x0000, Error: STATUS_ACCESS_DENIED
44	7.904453	192.168.134.129	192.168.134.132	SMB	162 NT Create AndX Request, FID: 0x400a, Path: \BROWSER
45	7.904829	192.168.134.132	192.168.134.129	SMB	205 NT Create AndX Response, FID: 0x400a
46	7.912909	192.168.134.129	192.168.134.132	DCERPC	733 Bind: call_id: 0, Fragment: Single, 13 context items: 0e042bc0-cab3-517d-523f-7cbe71a19c
47	7.913008	192.168.134.132	192.168.134.129	SMB	117 Write AndX Response, FID: 0x400a, 600 bytes
48	7.914816	192.168.134.129	192.168.134.132	SMB	129 Read AndX Request, FID: 0x400a, 467 bytes at offset 562
49	7.914893	192.168.134.132	192.168.134.129	DCERPC	486 Bind_ack: call_id: 0, Fragment: Single, max_xmit: 4280 max_recv: 4280, 13 results: Provi
50	7.918113	192.168.134.129	192.168.134.132	SMB	149 Write AndX Request, FID: 0x400a, 16 bytes at offset 702
51	7.918198	192.168.134.132	192.168.134.129	SMB	117 Write AndX Response, FID: 0x400a, 16 bytes
52	7.919972	192.168.134.129	192.168.134.132	SRVSVC	201 NETRPRNAMECANONICALIZE request[Long frame (60 bytes)]
53	7.920052	192.168.134.132	192.168.134.129	SMB	117 Write AndX Response, FID: 0x400a, 68 bytes
54	7.921760	192.168.134.129	192.168.134.132	SMB	129 Read AndX Request, FID: 0x400a, 447 bytes at offset 402
55	7.921867	192.168.134.132	192.168.134.129	DCERPC	162 Fault: call_id: 0, Fragment: Single, Ctx: 12, status: nca_s_fault_ndr
56	7.924788	192.168.134.129	192.168.134.132	SMB	162 NT Create AndX Request, FID: 0x400b, Path: \BROWSER
57	7.925109	192.168.134.132	192.168.134.129	SMB	205 NT Create AndX Response, FID: 0x400b
58	7.930307	192.168.134.129	192.168.134.132	SMB	376 Write AndX Request, FID: 0x400b, 243 bytes at offset 45
59	7.930423	192.168.134.132	192.168.134.129	SMB	117 Write AndX Response, FID: 0x400b, 243 bytes
60	7.932403	192.168.134.129	192.168.134.132	SMB	288 Write AndX Request, FID: 0x400b, 155 bytes at offset 531
61	7.932482	192.168.134.132	192.168.134.129	SMB	117 Write AndX Response, FID: 0x400b, 155 bytes
62	7.934304	192.168.134.129	192.168.134.132	DCERPC	291 Bind: call_id: 0, Fragment: Single, 12 context items: 5041909c-1b02-b6da-e70c-17600d90f1
63	7.934415	192.168.134.132	192.168.134.129	SMB	117 Write AndX Response, FID: 0x400b, 158 bytes

Appendix 4: Wireshark Packets 106-135 (SMB, TCP Begins)



106	8.085180	192.168.134.129	192.168.134.132	DCERPC	448 Bind: call_id: 0, Fragment: Single, 16 context items: 40f190fd-6e0a-8583-07b9-8ed3e53d42c0
107	8.085289	192.168.134.132	192.168.134.129	SMB	117 Write AndX Response, FID: 0x400e, 315 bytes
108	8.086943	192.168.134.129	192.168.134.132	SMB	129 Read AndX Request, FID: 0x400e, 336 bytes at offset 373
109	8.087057	192.168.134.132	192.168.134.129	SMB	466 Read AndX Response, FID: 0x400e, 336 bytes
110	8.088959	192.168.134.129	192.168.134.132	SMB	129 Read AndX Request, FID: 0x400e, 323 bytes at offset 454
111	8.089068	192.168.134.132	192.168.134.129	DCERPC	222 Bind_ack: call_id: 0, Fragment: Single, max_xmit: 4280 max_recv: 4280, 16 results: Provide
112	8.091788	192.168.134.129	192.168.134.132	SMB	372 Write AndX Request, FID: 0x400e, 239 bytes at offset 923
113	8.091915	192.168.134.132	192.168.134.129	SMB	117 Write AndX Response, FID: 0x400e, 239 bytes
114	8.093702	192.168.134.129	192.168.134.132	SMB	467 Write AndX Request, FID: 0x400e, 334 bytes at offset 853
115	8.093828	192.168.134.132	192.168.134.129	SMB	117 Write AndX Response, FID: 0x400e, 334 bytes
116	8.095607	192.168.134.129	192.168.134.132	SRVSVC	268 NetPathCanonicalize request
117	8.095736	192.168.134.132	192.168.134.129	SMB	117 Write AndX Response, FID: 0x400e, 135 bytes
118	8.096941	192.168.134.129	192.168.134.132	TCP	66 41254 → 445 [FIN, ACK] Seq=8161 Ack=5642 Win=30720 Len=0 TSval=1143436 TSecr=55471
119	8.097034	192.168.134.132	192.168.134.129	TCP	66 445 → 41254 [FIN, ACK] Seq=5642 Ack=8162 Win=62823 Len=0 TSval=55471 TSecr=1143436
120	8.097128	192.168.134.129	192.168.134.132	TCP	66 41254 → 445 [ACK] Seq=8162 Ack=5643 Win=30720 Len=0 TSval=1143437 TSecr=55471
121	8.097467	192.168.134.132	192.168.134.129	TCP	62 1241 → 29922 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM
122	8.097525	192.168.134.129	192.168.134.132	TCP	62 29922 → 1241 [SYN, ACK] Seq=0 Ack=1 Win=14600 Len=0 MSS=1460 SACK_PERM
123	8.097573	192.168.134.132	192.168.134.129	TCP	60 1241 → 29922 [ACK] Seq=1 Ack=1 Win=64240 Len=0
124	8.114542	192.168.134.129	192.168.134.132	TCP	60 29922 → 1241 [PSH, ACK] Seq=1 Ack=1 Win=14600 Len=4
125	8.115137	192.168.134.129	192.168.134.132	TCP	1514 29922 → 1241 [ACK] Seq=5 Ack=1 Win=14600 Len=1460
126	8.115138	192.168.134.129	192.168.134.132	TCP	1514 29922 → 1241 [ACK] Seq=1465 Ack=1 Win=14600 Len=1460
127	8.115139	192.168.134.129	192.168.134.132	TCP	1514 29922 → 1241 [ACK] Seq=2925 Ack=1 Win=14600 Len=1460
128	8.115140	192.168.134.129	192.168.134.132	TCP	1514 29922 → 1241 [ACK] Seq=4385 Ack=1 Win=14600 Len=1460
129	8.115140	192.168.134.129	192.168.134.132	TCP	1514 29922 → 1241 [ACK] Seq=5845 Ack=1 Win=14600 Len=1460
130	8.115141	192.168.134.129	192.168.134.132	TCP	1514 29922 → 1241 [ACK] Seq=7305 Ack=1 Win=14600 Len=1460
131	8.115141	192.168.134.129	192.168.134.132	TCP	1514 29922 → 1241 [ACK] Seq=8765 Ack=1 Win=14600 Len=1460
132	8.115141	192.168.134.129	192.168.134.132	TCP	1514 29922 → 1241 [ACK] Seq=10225 Ack=1 Win=14600 Len=1460
133	8.115142	192.168.134.129	192.168.134.132	TCP	1514 29922 → 1241 [ACK] Seq=11685 Ack=1 Win=14600 Len=1460
134	8.115342	192.168.134.132	192.168.134.129	TCP	60 1241 → 29922 [ACK] Seq=1 Ack=13145 Win=59860 Len=0
135	8.115876	192.168.134.132	192.168.134.129	TCP	60 [TCP Window Update] 1241 → 29922 [ACK] Seq=1 Ack=13145 Win=63620 Len=0