

# Monitoring and Incident Detection

Welcome back to another project. This time, I'm going to perform monitoring and incident detection using Wireshark. I will use the website "<https://www.malware-traffic-analysis.net/%E2%80%9D>" to obtain complex captures and perform monitoring.



We click on "Traffic Analysis Exercises", which will take us to another part of the page where it shows a variety of captures with different themes. In my case, I will choose the first one as recommended by the creator, since being the most recent, we can be a bit more up-to-date with the daily challenges in this topic.

## TRAFFIC ANALYSIS EXERCISES

### NOTES:

- If you're new to these exercises, start from the most recent and work your way back.
- Do **not** start with the oldest ones first!
- Malware and malware traffic is constantly evolving, so the further back you go, the less these exercises reflect our current threat landscape.
- Also, I grew better at creating these, so the earliest ones are not as good for training.

### EXERCISE LINKS:

- **2024-11-26 -- Traffic analysis exercise: Nemotodes**

We can observe that the creator of the page provides us with two zip files containing alerts and the ".pcap" along with instructions for the password of the zips. Additionally, there is a scenario to perform the corresponding tasks upon receiving a malware alert.

## 2024-11-26 - TRAFFIC ANALYSIS EXERCISE: NEMOTODES

### ASSOCIATED FILES:

- Zip archive of the pcap: [2024-11-26-traffic-analysis-exercise.pcap.zip](#) 19.7 MB (19,664,067 bytes)
- Zip archive of the alerts: [2024-11-26-traffic-analysis-exercise-alerts.zip](#) 297.5 kB (297,496 bytes)

### NOTES:

- Zip files are password-protected. Of note, this site has a new password scheme. For the password, see the "about" page of this website.

## BACKGROUND

You work as a analyst at a Security Operation Center (SOC) for a medical research facility specializing in nemotodes. Alerts on traffic in your network indicate someone has been infected. You don't know which is more disgusting, the nemotodes or the malware.

My recommendation is to always keep files organized and named. In my case, I created a folder with the files and captures I will use in this project. Once I have the material I will use, I will proceed to prepare my environment to begin the analysis. We will have Wireshark installed on our computer.

## Analysis Plan

### 1. Initial Analysis:

- Open the PCAP file in Wireshark to identify initial patterns.
- Filter by common protocols such as HTTP, HTTPS, DNS and SMB
- Look for traffic spikes or abnormal patterns.

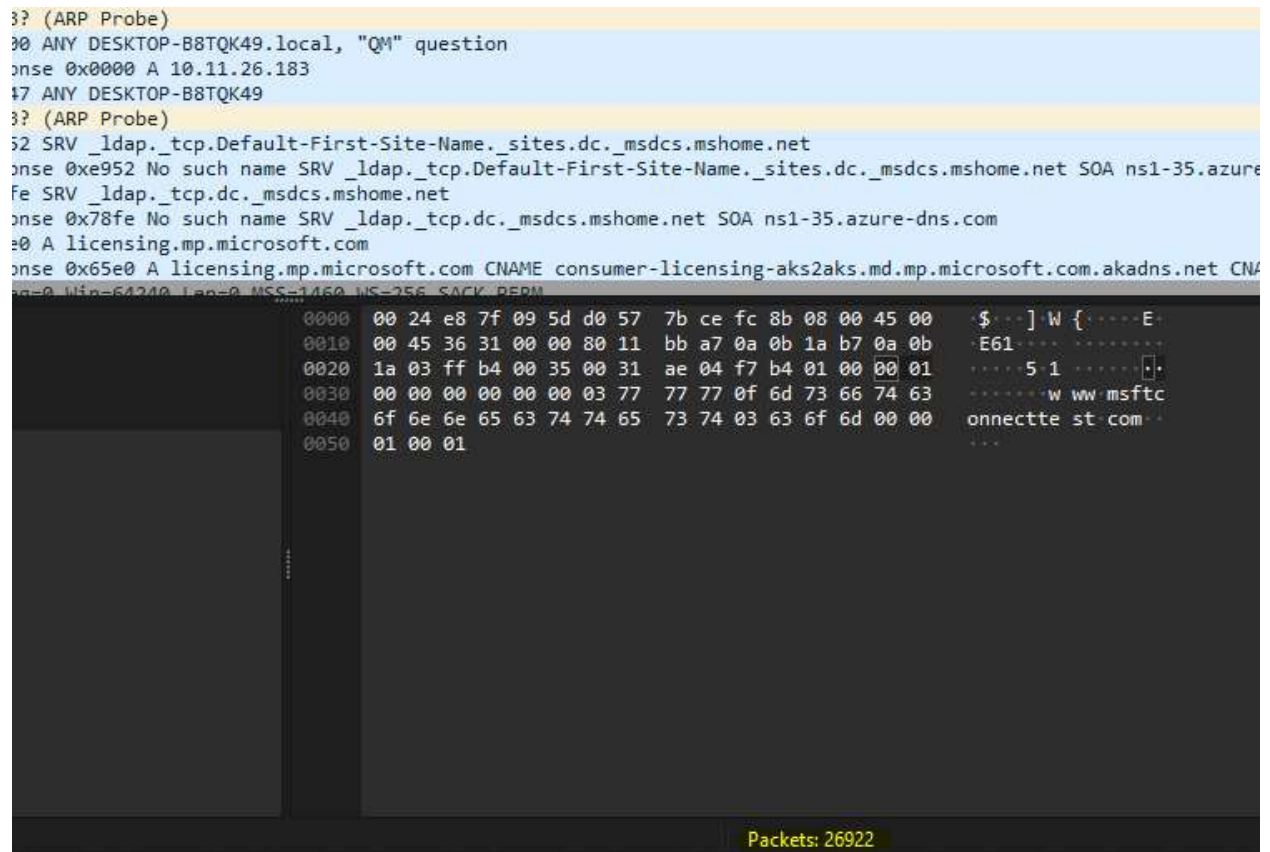
## Initial Analysis

We open the “.pcap” file in Wireshark.

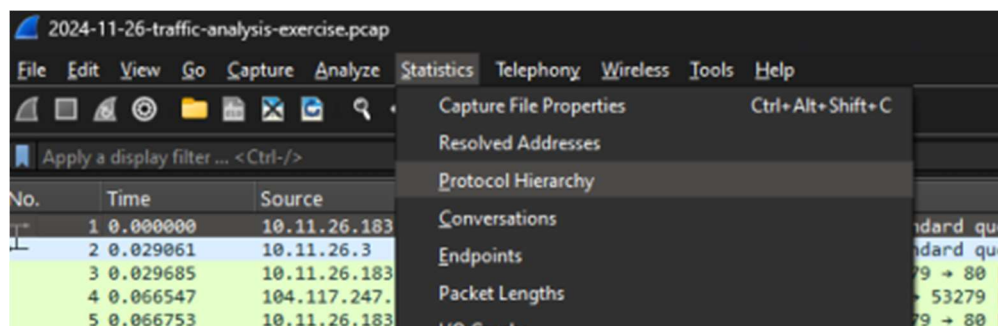
No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000	10.11.26.183	10.11.26.3	DNS	83	Standard query 0x7704 A www.microsoftconnect.com
2	0.029001	10.11.26.3	10.11.26.183	DNS	227	Standard query response 0x7704 A www.microsoftconnect.com CNAME nci.ge.trafficmanager.net CNAME www.mftncsl.com.edgesuite.net CNAME a1901.g2.akamai.net A 1...
3	0.029005	10.11.26.183	10.11.26.183	TCP	60	53279 → 80 [WIN] Seq=64240 Len=0 RST=1408 W=256 SACK_PERM
4	0.060547	10.11.26.183	10.11.26.183	TCP	60	80 → 53279 [WIN, ACK] Seq=64240 Len=0 RST=1408 W=256 SACK_PERM
5	0.060751	10.11.26.183	10.11.26.183	TCP	60	53279 → 80 [ACK] Seq=64240 Win=13872 Len=0
6	0.060876	10.11.26.183	10.11.26.183	HTTP	145	GET /connecttest.txt HTTP/1.1
7	0.114405	10.11.26.183	10.11.26.183	TCP	60	80 → 53279 [ACK] Seq=64240 Win=13872 Len=0
8	0.114702	10.11.26.183	10.11.26.183	HTTP	241	HTTP/1.1 200 OK (text/plain)
9	0.114726	10.11.26.183	10.11.26.183	TCP	60	80 → 53279 [FIN, ACK] Seq=64240 Win=13872 Len=0
10	0.114919	10.11.26.183	10.11.26.183	TCP	60	53279 → 80 [ACK] Seq=64240 Win=13872 Len=0
11	0.114948	10.11.26.183	10.11.26.183	TCP	60	53279 → 80 [FIN, ACK] Seq=64240 Win=13872 Len=0
12	0.115037	10.11.26.183	10.11.26.183	TCP	60	80 → 53279 [ACK] Seq=64240 Win=13872 Len=0
13	0.460408	10.11.26.183	10.11.26.3	DNS	118	Standard query 0x334c SWV _ldap._tcp.Default-First-Site-Name._sites.nemotoads.health
14	0.460624	10.11.26.183	10.11.26.3	DNS	118	Standard query 0x334c SWV _ldap._tcp.Default-First-Site-Name._sites.nemotoads.health
15	0.460802	10.11.26.183	10.11.26.3	DNS	81	Standard query 0x334c A upad.nemotoads.health
16	0.460807	10.11.26.183	10.11.26.3	DNS	81	Standard query 0x334c A upad.nemotoads.health
17	0.460925	10.11.26.3	10.11.26.183	DNS	153	Standard query response 0x334c SWV _ldap._tcp.Default-First-Site-Name._sites.nemotoads.health SWV 0 100 500 nemotoads-dc.nemotoads.health A 10.11.26.3
18	0.460909	10.11.26.3	10.11.26.183	DNS	153	Standard query response 0x334c SWV _ldap._tcp.Default-First-Site-Name._sites.nemotoads.health SWV 0 100 500 nemotoads-dc.nemotoads.health A 10.11.26.3
19	0.461801	10.11.26.3	10.11.26.183	DNS	157	Standard query response 0x334c No such name A upad.nemotoads.health SOA nemotoads-dc.nemotoads.health
20	0.461801	10.11.26.3	10.11.26.183	DNS	157	Standard query response 0x334c No such name A upad.nemotoads.health SOA nemotoads-dc.nemotoads.health
21	0.461135	10.11.26.183	10.11.26.3	DNS	75	Standard query 0x4801 A upad.msphone.net
22	0.461135	10.11.26.183	10.11.26.3	DNS	75	Standard query 0x4801 A upad.msphone.net
23	0.461138	10.11.26.3	10.11.26.183	DNS	161	Standard query response 0x4801 No such name A upad.msphone.net SOA nsl-35.azure-dns.com
24	0.461138	10.11.26.183	10.11.26.3	LDAP	209	searchRequest(92) "dc=upad" baseObject
25	0.461137	10.11.26.3	10.11.26.183	DNS	161	Standard query response 0x4801 No such name A upad.msphone.net SOA nsl-35.azure-dns.com
26	0.461441	10.11.26.183	10.11.26.3	LDAP	209	searchRequest(93) "dc=upad" baseObject
27	0.461573	10.11.26.3	10.11.26.183	LDAP	231	searchResultEntry(93) "dc=upad" searchResultEntry(93) success [2 results]
28	0.461517	10.11.26.3	10.11.26.183	LDAP	231	searchResultEntry(92) "dc=upad" searchResultEntry(92) success [2 results]
29	0.570646	10.11.26.183	10.11.26.3	TCP	60	53280 → 80 [WIN] Seq=64240 Len=0 RST=1408 W=256 SACK_PERM
30	0.570716	10.11.26.3	10.11.26.183	TCP	60	80 → 53280 [WIN, ACK] Seq=64240 Len=0 RST=1408 W=256 SACK_PERM
31	0.570800	10.11.26.3	10.11.26.183	TCP	60	80 → 53280 [ACK] Seq=64240 Win=13872 Len=0
32	0.570800	10.11.26.3	10.11.26.183	TCP	60	80 → 53280 [ACK] Seq=64240 Win=13872 Len=0
33	0.571099	10.11.26.183	10.11.26.3	TCP	60	53280 → 80 [ACK] Seq=64240 Win=13872 Len=0
34	0.571139	10.11.26.183	10.11.26.3	TCP	60	53281 → 80 [ACK] Seq=64240 Win=13872 Len=0
35	0.571140	10.11.26.183	10.11.26.3	TCP	60	53281 → 80 [ACK] Seq=64240 Win=13872 Len=0
36	0.571206	10.11.26.183	10.11.26.3	TCP	60	53281 → 80 [ACK] Seq=64240 Win=13872 Len=0
37	0.571206	10.11.26.183	10.11.26.3	TCP	60	53281 → 80 [ACK] Seq=64240 Win=13872 Len=0
38	0.571206	10.11.26.183	10.11.26.3	TCP	60	53281 → 80 [ACK] Seq=64240 Win=13872 Len=0
39	0.572062	10.11.26.3	10.11.26.183	TCP	1514	309 → 53280 [ACK] Seq=64240 Win=13872 Len=0 [TCP PMTU reassembled in 40]
40	0.572076	10.11.26.3	10.11.26.183	LDAP	1393	searchRequest(112) "dc=upad" baseObject
41	0.572705	10.11.26.183	10.11.26.3	TCP	60	53280 → 80 [ACK] Seq=64240 Win=13872 Len=0
42	0.572725	10.11.26.183	10.11.26.3	TCP	1514	309 → 53280 [ACK] Seq=64240 Win=13872 Len=0 [TCP PMTU reassembled in 43]
43	0.572735	10.11.26.183	10.11.26.3	LDAP	715	bindRequest(14) "dc=upad" baseObject
44	0.572808	10.11.26.3	10.11.26.183	TCP	60	80 → 53280 [ACK] Seq=64240 Win=13872 Len=0
45	0.572807	10.11.26.3	10.11.26.183	LDAP	209	searchRequest(14) success
46	0.572836	10.11.26.183	10.11.26.3	LDAP	97	5461 655-APP Integrity: upadInResponse(1)
47	0.572836	10.11.26.183	10.11.26.3	TCP	60	53280 → 80 [ACK] Seq=64240 Win=13872 Len=0
48	0.572836	10.11.26.183	10.11.26.3	TCP	60	53280 → 80 [ACK] Seq=64240 Win=13872 Len=0
49	0.572836	10.11.26.183	10.11.26.3	TCP	60	53280 → 80 [ACK] Seq=64240 Win=13872 Len=0
50	0.572836	10.11.26.183	10.11.26.3	TCP	60	53280 → 80 [ACK] Seq=64240 Win=13872 Len=0
51	0.572836	10.11.26.183	10.11.26.3	TCP	60	53280 → 80 [ACK] Seq=64240 Win=13872 Len=0
52	0.572836	10.11.26.183	10.11.26.3	TCP	60	53280 → 80 [ACK] Seq=64240 Win=13872 Len=0
53	0.572836	10.11.26.183	10.11.26.3	TCP	60	53280 → 80 [ACK] Seq=64240 Win=13872 Len=0
54	0.572836	10.11.26.183	10.11.26.3	TCP	60	53280 → 80 [ACK] Seq=64240 Win=13872 Len=0
55	0.572836	10.11.26.183	10.11.26.3	TCP	60	53280 → 80 [ACK] Seq=64240 Win=13872 Len=0
56	0.572836	10.11.26.183	10.11.26.3	TCP	60	53280 → 80 [ACK] Seq=64240 Win=13872 Len=0
57	0.572836	10.11.26.183	10.11.26.3	TCP	60	53280 → 80 [ACK] Seq=64240 Win=13872 Len=0
58	0.572836	10.11.26.183	10.11.26.3	TCP	60	53280 → 80 [ACK] Seq=64240 Win=13872 Len=0
59	0.572836	10.11.26.183	10.11.26.3	TCP	60	53280 → 80 [ACK] Seq=64240 Win=13872 Len=0
60	0.572836	10.11.26.183	10.11.26.3	TCP	60	53280 → 80 [ACK] Seq=64240 Win=13872 Len=0
61	0.572836	10.11.26.183	10.11.26.3	TCP	60	53280 → 80 [ACK] Seq=64240 Win=13872 Len=0
62	0.572836	10.11.26.183	10.11.26.3	TCP	60	53280 → 80 [ACK] Seq=64240 Win=13872 Len=0
63	0.572836	10.11.26.183	10.11.26.3	TCP	60	53280 → 80 [ACK] Seq=64240 Win=13872 Len=0
64	0.572836	10.11.26.183	10.11.26.3	TCP	60	53280 → 80 [ACK] Seq=64240 Win=13872 Len=0
65	0.572836	10.11.26.183	10.11.26.3	TCP	60	53280 → 80 [ACK] Seq=64240 Win=13872 Len=0
66	0.572836	10.11.26.183	10.11.26.3	TCP	60	53280 → 80 [ACK] Seq=64240 Win=13872 Len=0
67	0.572836	10.11.26.183	10.11.26.3	TCP	60	53280 → 80 [ACK] Seq=64240 Win=13872 Len=0
68	0.572836	10.11.26.183	10.11.26.3	TCP	60	53280 → 80 [ACK] Seq=64240 Win=13872 Len=0
69	0.572836	10.11.26.183	10.11.26.3	TCP	60	53280 → 80 [ACK] Seq=64240 Win=13872 Len=0
70	0.572836	10.11.26.183	10.11.26.3	TCP	60	53280 → 80 [ACK] Seq=64240 Win=13872 Len=0
71	0.572836	10.11.26.183	10.11.26.3	TCP	60	53280 → 80 [ACK] Seq=64240 Win=13872 Len=0
72	0.572836	10.11.26.183	10.11.26.3	TCP	60	53280 → 80 [ACK] Seq=64240 Win=13872 Len=0
73	0.572836	10.11.26.183	10.11.26.3	TCP	60	53280 → 80 [ACK] Seq=64240 Win=13872 Len=0
74	0.572836	10.11.26.183	10.11.26.3	TCP	60	53280 → 80 [ACK] Seq=64240 Win=13872 Len=0
75	0.572836	10.11.26.183	10.11.26.3	TCP	60	53280 → 80 [ACK] Seq=64240 Win=13872 Len=0
76	0.572836	10.11.26.183	10.11.26.3	TCP	60	53280 → 80 [ACK] Seq=64240 Win=13872 Len=0
77	0.572836	10.11.26.183	10.11.26.3	TCP	60	53280 → 80 [ACK] Seq=64240 Win=13872 Len=0
78	0.572836	10.11.26.183	10.11.26.3	TCP	60	53280 → 80 [ACK] Seq=64240 Win=13872 Len=0
79	0.572836	10.11.26.183	10.11.26.3	TCP	60	53280 → 80 [ACK] Seq=64240 Win=13872 Len=0
80	0.572836	10.11.26.183	10.11.26.3	TCP	60	53280 → 80 [ACK] Seq=64240 Win=13872 Len=0
81	0.572836	10.11.26.183	10.11.26.3	TCP	60	53280 → 80 [ACK] Seq=64240 Win=13872 Len=0
82	0.572836	10.11.26.183	10.11.26.3	TCP	60	53280 → 80 [ACK] Seq=64240 Win=13872 Len=0
83	0.572836	10.11.26.183	10.11.26.3	TCP	60	53280 → 80 [ACK] Seq=64240 Win=13872 Len=0
84	0.572836	10.11.26.183	10.11.26.3	TCP	60	53280 → 80 [ACK] Seq=64240 Win=13872 Len=0
85	0.572836	10.11.26.183	10.11.26.3	TCP	60	53280 → 80 [ACK] Seq=64240 Win=13872 Len=0
86	0.572836	10.11.26.183	10.11.26.3	TCP	60	53280 → 80 [ACK] Seq=64240 Win=13872 Len=0
87	0.572836	10.11.26.183	10.11.26.3	TCP	60	53280 → 80 [ACK] Seq=64240 Win=13872 Len=0
88	0.572836	10.11.26.183	10.11.26.3	TCP	60	53280 → 80 [ACK] Seq=64240 Win=13872 Len=0
89	0.572836	10.11.26.183	10.11.26.3	TCP	60	53280 → 80 [ACK] Seq=64240 Win=13872 Len=0
90	0.572836	10.11.26.183	10.11.26.3	TCP	60	53280 → 80 [ACK] Seq=64240 Win=13872 Len=0
91	0.572836	10.11.26.183	10.11.26.3	TCP	60	53280 → 80 [ACK] Seq=64240 Win=13872 Len=0
92	0.572836	10.11.26.183	10.11.26.3	TCP	60	53280 → 80 [ACK] Seq=64240 Win=13872 Len=0
93	0.572836	10.11.26.183	10.11.26.3	TCP	60	53280 → 80 [ACK] Seq=64240 Win=13872 Len=0
94	0.572836	10.11.26.183	10.11.26.3	TCP	60	53280 → 80 [ACK] Seq=64240 Win=13872 Len=0
95	0.572836	10.11.26.183	10.11.26.3	TCP	60	53280 → 80 [ACK] Seq=64240 Win=13872 Len=0
96	0.572836	10.11.26.183	10.11.26.3	TCP	60	53280 → 80 [ACK] Seq=64240 Win=13872 Len=0
97	0.572836	10.11.26.183	10.11.26.3	TCP	60	53280 → 80 [ACK] Seq=64240 Win=13872 Len=0
98	0.572836	10.11.26.183	10.11.26.3	TCP	60	53280 → 80 [ACK] Seq=64240 Win=13872 Len=0
99	0.572836	10.11.26.183	10.11.26.3	TCP	60	53280 → 80 [ACK] Seq=64240 Win=13872 Len=0
100	0.572836	10.11.26.183	10.11.26.3	TCP	60	53280 → 80 [ACK] Seq=64240 Win=13872 Len=0

There are the basics we will observe when opening the “.pcap” file in Wireshark. We have a lot of data that we will analyze:

1. Number of Packets: in the bottom bar of Wireshark, we will see the number of packets, which will give us an idea of the traffic size. In total, we have 26,922 packets.



2. Main Protocols: Go to the menu Statistics > Protocol Hierarchy, this shows a breakdown of the present protocols.



Protocol	Percent Packets	Packets	Percent Bytes	Bytes	Bits/s	End Packets	End Bytes	End Bits/s	POUs
Frame	100.0	26922	100.0	20851149	51 k	0	0	0	26922
Ethernet	100.0	26922	2.1	443576	1088	0	0	0	26922
Internet Protocol Version 4	98.6	26544	2.5	530880	1302	0	0	0	26544
User Datagram Protocol	2.5	680	0.0	5440	13	0	0	0	680
Simple Service Discovery Protocol	0.0	10	0.0	1370	3	10	1370	3	10
QUIC IETF	0.9	253	0.7	143743	352	253	143608	352	253
Network Time Protocol	0.0	8	0.0	960	2	8	960	2	8
NetBIOS Name Service	0.1	20	0.0	1312	3	20	1312	3	20
NetBIOS Datagram Service	0.1	17	0.0	1394	3	0	0	0	17
SMB (Server Message Block Protocol)	0.1	17	0.0	1843	4	0	0	0	17
SMB MailSlot Protocol	0.1	17	0.0	425	1	0	0	0	17
Microsoft Windows Browser Protocol	0.1	17	0.0	381	0	17	381	0	17
Multicast Domain Name System	0.0	7	0.0	296	0	7	296	0	7
Link-local Multicast Name Resolution	0.0	2	0.0	66	0	2	66	0	2
Domain Name System	1.3	343	0.1	28764	70	343	28764	70	343
Connectionless Lightweight Directory Access Protocol	0.1	20	0.0	4201	10	20	4201	10	20
Transmission Control Protocol	96.1	25864	2.5	522596	1282	20285	411016	1008	25864
Transport Layer Security	17.8	4780	90.4	18847010	46 k	4780	17719038	43 k	4904
NetBIOS Session Service	1.4	388	0.5	94251	231	10	388	0	388
SMB2 (Server Message Block Protocol version 2)	1.1	292	0.4	81099	199	256	66295	162	314
SMB (Server Message Block Protocol)	0.3	86	0.1	11260	27	66	9440	23	86
SMB Pipe Protocol	0.1	20	0.0	290	0	0	0	0	20
Microsoft Windows Lanman Remote API Protocol	0.1	20	0.0	340	0	20	340	0	20
Lightweight Directory Access Protocol	0.4	113	0.4	78458	192	113	59019	144	120
Kerberos	0.1	16	0.1	23415	57	16	23415	57	16
Hypertext Transfer Protocol	0.3	74	0.1	15973	39	8	1892	4	74
Online Certificate Status Protocol	0.0	1	0.0	504	1	1	504	1	1
Line-based text data	0.0	3	0.0	42	0	3	42	0	3
HTML Form URL Encoded	0.2	62	0.0	3050	7	62	3050	7	62
Distributed Computing Environment / Remote Procedure Call (DCE/RPC)	0.5	140	0.2	35903	88	34	13855	34	140
SAMR (pidl)	0.1	30	0.0	2240	5	30	2240	5	30
Microsoft Network Logon	0.0	2	0.0	1872	4	2	1872	4	2
Local Security Authority	0.0	4	0.0	560	1	4	560	1	4
DCE/RPC Endpoint Mapper	0.1	20	0.0	3976	9	20	3976	9	20
Active Directory Replication	0.2	50	0.0	6272	15	50	6272	15	50
Data	0.4	104	0.2	46661	114	104	46661	114	104
Address Resolution Protocol	1.4	378	0.1	10584	25	378	10584	25	378

We identify the most common protocols and note which ones could be relevant. In this case, I will focus on the HTTP, HTTPS, DNS, and SMB protocols. We can also filter by common protocols.

- HTTP: in the filter bar, we type “http”. We can look for request like GET, POST, or any other HTTP traffic.



.3

- HTTPS: in the filter bar, we type “tls” to see HTTPS traffic. Although it is encrypted, you can search for domains through the SNI (Server Name Indication) field in the TLS Client Hello packets. Right-click on a “tls” packet and select Follow > TLS Stream to analyze the TLS flow.



- DNS: Repeat the same process by applying the “dns” filter. We can examine DNS queries and their responses. Look for long, strange, or failed domains that may indicate malicious traffic.



- SMB: We type “smb” or “smb2” in the filter bar, we can examine whether there are any attempts to access shared resources that could indicate lateral movements or exfiltration attempts.

## Identify anomalies (DNS)

We will use “dns” in the filter bar to search for anomalies. What are we looking for?

- Domains with strange or long names.
- DNS queries that have no response (which could be indicators of command and control attempts).

### 1. Example: WPAD (Web Proxy Auto-Discovery Protocol) Lookups.

No.	Time	Source	Destination	Protocol	Length	Info
187	5.794700	10.11.26.183	10.11.26.3	DNS	75	Standard query 0xe884 A wpad.mshome.net
15	0.460652	10.11.26.183	10.11.26.3	DNS	81	Standard query 0xa909 A wpad.nemotoads.health

Queries to the “wpad” domain indicate that the system is looking for automatic proxy configurations. This can be expected behavior in corporate networks, but it could also be exploited by attackers who set up a malicious WPAD server.

What can we do to detect anomalies?

- Verify if the “wpad” domains are legitimate in your network.
- If there are unexpected response or external domains in the DNS responses related to “wpad”, it could be a malicious redirection attempt.

### 2. Example: DNS Responses with Multiple CNAME Records.

No.	Time	Source	Destination	Protocol	Src Port	Dest Port	Length	Info
2	0.029061	10.11.26.3	10.11.26.183	DNS	53	65460	227	Standard query response 0xf7b4 A www.msftconnecttest.com CNAME ncsi-geo.trafficmanager.net CNAME www.msftnccs

The domain “[www.msftconnecttest.com](http://www.msftconnecttest.com)” is a legitimate resource used by Windows to verify internet connectivity. However, a long chain of CNAMEs could be suspicious if it comes from unknown or unrelated domains.

What can we do to detect anomalies?

- Identify if all domains in the CNAME chain are legitimate.
- If there are unrecognized domains, perform additional analysis to check if they are associated with malicious activity.



### 3. Example: Repeated queries to invalid domains.

Time	Source	Destination	Protocol	Src Port	Dest Port	Length	Info
113 3.248564	10.11.26.3	10.11.26.183	DNS	53	50534	177	Standard query response 0xae3d No such name SRV _ldap._tcp.dc._msdcs.mshome.net

Queries to invalid domains, such as “mshome.net”, can be normal in test environments or misconfigurations. However, if the domains are unrelated to your network, it could indicate unwanted activity.

What can we do to detect anomalies?

- Identify if the domains are related to your network.
- If not, investigate which device is making these queries and if it is legitimate.

### 4. Example: Unknown domain with multiple queries.

Time	Source	Destination	Protocol	Src Port	Dest Port	Length	Info
13 0.460460	10.11.26.183	10.11.26.3	DNS	55542	53	118	Standard query 0x3b4c SRV _ldap._tcp.Default-First-Site-Name._sites.nemotoads.h

The domain “nemotoads.health” doesn’t seem to be common or well-known. If it’s not part of the infrastructure, it could be a possible misconfiguration or a malicious attempt to access.

What can we do to detect anomalies?

- Validate if this domain belongs to the organization.
- If it’s not legitimate, track the device that is generating these queries.

### 5. Example: Common negative response (No such name).

Time	Source	Destination	Protocol	Src Port	Dest Port	Length	Info
56 2.196834	10.11.26.3	10.11.26.183	DNS	53	59980	208	Standard query response 0xe952 No such name SRV _ldap._tcp.Default-First-Site-N

If you see many “No such name” responses, it could be a symptom of misconfigurations or infected devices trying to communicate with malicious domains.

What can we do to detect anomalies?

- Review the device generating these queries.
- Ensure that the devices are configured correctly.

## Additional tools to go deeper

- Domain checking: We can use services like “VirusTotal” to check if the domains or IPs querying have reports of malicious activity.
- Wireshark filters:
  1. Use the filter “dns.flags.rcode != 0” to focus on DNS errors.
  2. Filter by specific domains: “dns.qry.name == wpad.nemotoads.health”.

## Example:

1374 35.856917 10.11.26.3 10.11.26.183 DNS 53 52957 96 Standard query response 0xa31d A modandcrackedapk.com A 193.42.38.139



SUMMARY

DETECTION

DETAILS

RELATIONS

COMMUNITY

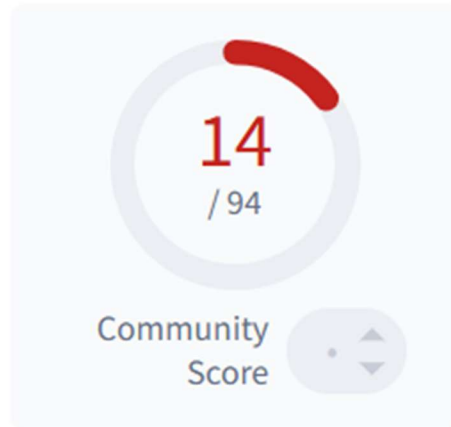
Security vendors' analysis ⓘ

Do you want to automate checks?

alphaMountain.ai	⚠ Phishing
BitDefender	⚠ Malware
CRDF	⚠ Malicious
CyRadar	⚠ Malicious
ESET	⚠ Phishing
G-Data	⚠ Malware
Lionic	⚠ Malware
Lumu	⚠ Malware
MalwareURL	⚠ Malware
Seclookup	⚠ Malicious
SOCRadar	⚠ Malicious
Sophos	⚠ Phishing
VIPRE	⚠ Phishing
Webroot	⚠ Malicious
Gridinsoft	ⓘ Suspicious
Abusix	✅ Clean
Acronis	✅ Clean
ADMINUSLabs	✅ Clean



**14/94 security vendors flagged this domain as malicious**



modandcrackedapk.com

**Creation date**

1 year ago

**Last analysis date**

2 days ago



## Identify Anomalies (HTTP and HTTPS)

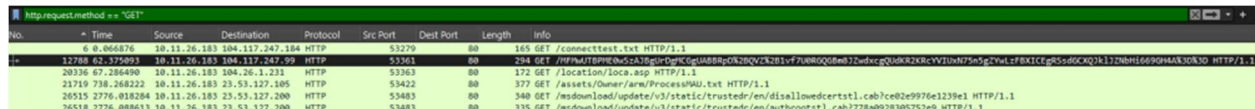
For HTTP traffic, we start by using the filter “http”, or we can be more specific by using the following filters:

- “http.request.method == “GET” ” (For GET request).
- “http.request.method == “Post” ” (For POST request).

For HTTPS traffic, we use the filter “ssl” or “tls.handshake.type == 1” (To identify TLS packets with SNI).

### 1. Example:

We can observe the result of the filter “http.request.method == “GET” ”



The image shows a Wireshark packet capture window with the filter "http.request.method == 'GET'". The packet list shows several HTTP GET requests. The first packet is a GET request to /connecttest.txt. The second packet is a GET request to /location/loc.asp. The third packet is a GET request to /assets/Owner/arm/ProcessPAU.txt. The fourth packet is a GET request to /msdownload/update/v3/static/trusted/en/disallowedcertstl.cab. The fifth packet is a GET request to /msdownload/update/v3/static/trusted/en/authrootstl.cab.

No.	Time	Source	Destination	Protocol	Src Port	Dest Port	Length	Info
0	0.066576	10.11.26.183	104.117.247.184	HTTP	53279	80	165	GET /connecttest.txt HTTP/1.1
12184	0.2872092	10.11.26.183	10.11.26.127	HTTP	53281	80	214	GET /location/loc.asp HTTP/1.1
20336	67.286490	10.11.26.183	104.26.1.231	HTTP	53283	80	172	GET /assets/Owner/arm/ProcessPAU.txt HTTP/1.1
21719	738.268222	10.11.26.183	23.53.127.105	HTTP	53422	80	377	GET /msdownload/update/v3/static/trusted/en/disallowedcertstl.cab HTTP/1.1
26515	2776.018284	10.11.26.183	23.53.127.200	HTTP	53483	80	340	GET /msdownload/update/v3/static/trusted/en/authrootstl.cab HTTP/1.1
26518	2776.888613	10.11.26.183	23.53.127.200	HTTP	53483	80	335	GET /msdownload/update/v3/static/trusted/en/authrootstl.cab HTTP/1.1

This caught my attention, I double-clicked it to inspect the details. Such as the host (domain) and packet information.

```

▶ Frame 12788: 294 bytes on wire (2352 bits), 294 bytes captured (2352 bits)
▶ Ethernet II, Src: Intel_ce:fc:8b (d0:57:7b:ce:fc:8b), Dst: Cisco_b8:29:5e (00:17:e0:b8:29:5e)
▼ Internet Protocol Version 4, Src: 10.11.26.183, Dst: 104.117.247.99
  0100 .... = Version: 4
  .... 0101 = Header Length: 20 bytes (5)
  ▼ Differentiated Services Field: 0x00 (DSCP: CS0, ECN: Not-ECT)
    0000 00.. = Differentiated Services Codepoint: Default (0)
    .... ..00 = Explicit Congestion Notification: Not ECN-Capable Transport (0)
  Total Length: 280
  Identification: 0x3495 (13461)
  ▶ 010. .... = Flags: 0x2, Don't fragment
  ...0 0000 0000 0000 = Fragment Offset: 0
  Time to Live: 128
  Protocol: TCP (6)
  Header Checksum: 0x40b0 [validation disabled]
  [Header checksum status: Unverified]
  Source Address: 10.11.26.183
  Destination Address: 104.117.247.99
  [Stream index: 35]
▼ Transmission Control Protocol, Src Port: 53361, Dst Port: 80, Seq: 1, Ack: 1, Len: 240
  Source Port: 53361
  Destination Port: 80
  [Stream index: 76]
  [Conversation completeness: Complete, WITH_DATA (47)]
  [TCP Segment Len: 240]
  Sequence Number: 1 (relative sequence number)
  Sequence Number (raw): 3793564077
  [Next Sequence Number: 241 (relative sequence number)]
  Acknowledgment Number: 1 (relative ack number)
  Acknowledgment number (raw): 3347828949
  0101 .... = Header Length: 20 bytes (5)
  ▶ Flags: 0x018 (PSH, ACK)
  Window: 512
  [Calculated window size: 131072]
  [Window size scaling factor: 256]
  Checksum: 0x04ad [unverified]
  [Checksum Status: Unverified]
  Urgent Pointer: 0
  [Timestamps]
  [SEQ/ACK analysis]
  TCP payload (240 bytes)
▼ Hypertext Transfer Protocol
  GET /MFMwUTBPME0wSzAJBgUrDgMCGGUABBRpD%2BQVZ%2B1vf7U0RGQG8m8JZwxcgQUdKR2KRcYVIUxN75n5gZYwLzFBXICEgRSsdGCXQJklJZNbHi669GH4A%3D%3D HTTP/1.1\r\n
    Request Method: GET
    Request URI: /MFMwUTBPME0wSzAJBgUrDgMCGGUABBRpD%2BQVZ%2B1vf7U0RGQG8m8JZwxcgQUdKR2KRcYVIUxN75n5gZYwLzFBXICEgRSsdGCXQJklJZNbHi669GH4A%3D%3D
    Request Version: HTTP/1.1
    Connection: Keep-Alive\r\n
    Accept: */*\r\n
    User-Agent: Microsoft-CryptoAPI/10.0\r\n
    Host: r10.o.lencr.org\r\n
  \r\n
  [Response in frame: 12790]
  [Full request URI: http://r10.o.lencr.org/MFMwUTBPME0wSzAJBgUrDgMCGGUABBRpD%2BQVZ%2B1vf7U0RGQG8m8JZwxcgQUdKR2KRcYVIUxN75n5gZYwLzFBXICEgRSsdGCXQJklJZNbHi669GH4A%3D%3D]

```

1. As highlighted information we have the host (domain):
  - The domain of interest is r10.o.lencr.org. This should be investigate to verify if it is legitimate or suspicious.
2. Packet information:
  - HTTP Method: GET
  - URL:

/MFMwUTBPME0wSzAJBgUrDgMCGGUABBRpD%2BQVZ%2B1vf7U0RGQG8m8JZwxcgQUdKR2KRcYVIUxN75n5gZYwLzFBXICEgRSsdGCXQJklJZNbHi669GH4A%3D%3D

Complex and long, but it is important to analyze if it points to any malicious or interesting resource.

- User-Agent: Microsoft-CryptoAPI/10.0.
- Dest IP: 104.117.247.99, corresponds to the domain but more details about the IP can be verified.
- Connection: Keep-Alive.

The steps to analyze them in depth are not so different from what we did the DNS:

1. We verify the domain in VirusTotal:

0

/ 94

Community Score

At least 10 detected files communicating with this domain

Reanalyze

Similar

More

r10.o.lencr.org

lencr.org

top 10K

Registrar

CloudFlare, Inc.

Creation Date

4 years ago

Last Analysis Date

17 minutes ago

DETECTION

DETAILS

RELATIONS

COMMUNITY 11

Popularity ranks

Rank	Position	Ingestion Time
Cisco Umbrella	2892	2025-01-20 14:38:07 UTC

Last DNS records

Record type	TTL	Value
A	20	23.212.62.92
A	20	23.212.62.91
AAAA	20	2600:1406:3a00:6::173e:2ed7
AAAA	20	2600:1406:3a00:6::173e:2ecb
CNAME	300	o.lencr.edgesuite.net
CNAME	8450	a1887.dscq.akamai.net
CNAME	8449	a1887.dscq.akamai.net

Last HTTPS Certificate

JARM Fingerprint

27d27d27d29d27d21c42d42d00000996c218236a1fd203fd29824aa76026c

Last HTTPS Certificate

Data:

Version: V3

Serial Number: b0efa6998487092a5d64ec0e7a56ef2

Thumbprint: 2839af637d02e8f71723a0eee8c92f9c6417680a

Signature Algorithm:

Issuer: C=US , O=DigiCert Inc , CN=DigiCert TLS RSA SHA256 2020 CA1

Validity

Not Before: 2024-04-18 00:00:00

Not After: 2025-04-19 23:59:59

Subject: C=US , ST=Massachusetts , L=Cambridge , O=Akamai Technologies, Inc. , CN=a248.e.akamai.net

Subject Public Key Info:

Public Key Algorithm: EC

2. We verify the destination IP:

0

/ 94

Community Score

6 detected files communicating with this IP address

Reanalyze

Similar

More

104.117.247.99 (104.117.192.0/18)

US

Last Analysis Date

2 months ago

DETECTION

DETAILS

RELATIONS

COMMUNITY 1

Basic Properties

Network

104.117.192.0/18

Autonomous System Number

20940

Autonomous System Label

Akamai International B.V.

Regional Internet Registry

ARIN

Country

US

Continent

NA

Last HTTPS Certificate

JARM Fingerprint

27d27d27d29d27d21c42d42d00000996c218236a1fd203fd29824aa76026c

Last HTTPS Certificate

Data:

Version: V3

Serial Number: b0efa6998487092a5d64ec0e7a56ef2

Thumbprint: 2839af637d02e8f71723a0eee0c92f9c6417680a

Signature Algorithm:

Issuer: C=US , O=DigiCert Inc , CN=DigiCert TLS RSA SHA256 2020 CA1

Validity

Not Before: 2024-04-18 00:00:00

Not After: 2025-04-19 23:59:59

Subject: C=US , ST=Massachusetts , L=Cambridge , O=Akamai Technologies, Inc. , CN=a248.e.akamai.net

Subject Public Key Info:

Public Key Algorithm : EC

OID: secp256r1

pub:

30:59:30:13:06:07:2a:06:48:ce:3d:02:01:06:08:

2a:06:48:ce:3d:03:01:07:03:42:00:04:05:e0:b1:

45:35:fa:50:0e:96:8a:e5:a8:52:f9:a3:c4:23:96:

8c:b6:56:d0:30:84:d3:31:0c:a9:3a:94:13:69:df:

c1:58:ec:a0:d0:c7:96:20:fc:77:4b:55:05:f7:82:

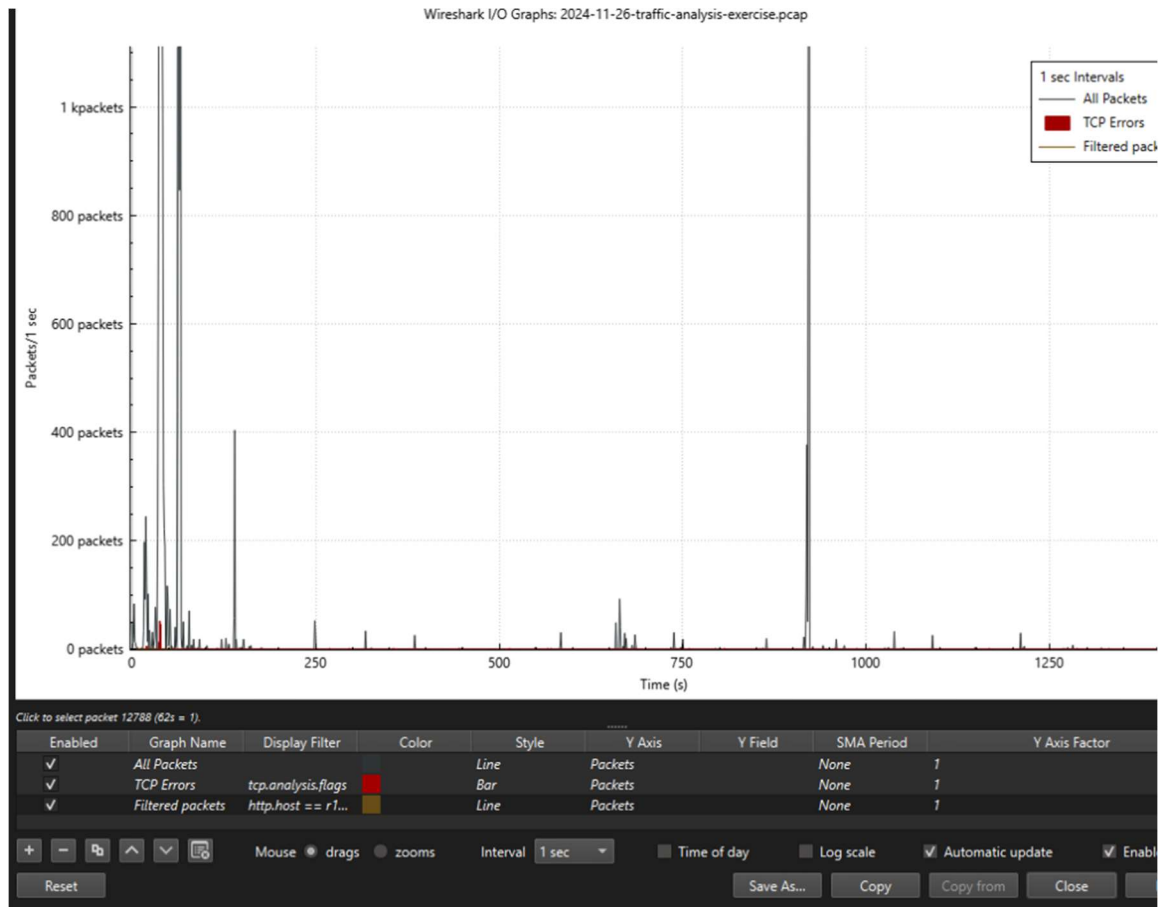
### 3. Analyze the traffic context:

- In Wireshark, we filter all requests from the host “http.host == “r10.o.lencr.org””, to be able to see all HTTP requests to this domain.

No.	Time	Source	Destination	Protocol	Src Port	Dest Port	Length	Info
12788	2024-11-25 23:58:48.833291	10.11.26.183	104.117.247.99	HTTP	53561	80	294	GET /PFPhU78PHEbUa3A78g0rDgKc0gUa888p0N2BQVZ2B1vF7U88Q08d72w8cgQ04K820RcYV1Uw075n5gYwLz7BXC7gS5d8CQ3k37ZNBH1669QHAB3

We can observe that we only have one request.

- Identify temporal patterns: We can review the “Time” column to see if the requests are made at intervals or unexpected times, such as outside of business hours.
- If the requests are frequent and do not correlate with normal activities, it could indicate suspicious behavior.
- We can use tools like traffic flow graphs to visualize the volume of requests to this domain compared to others,



Identify relevant patterns.

- Traffic spikes:** By observing the initial spikes in the graph, such as in the first 500 seconds. This indicates a significant increase in the number of packets per second. These spikes could represent unusual events, such as massive data transfers or attacks.
- Sustained or sporadic traffic:** After the spikes, there appears to be more constant traffic with some sporadic activity.
- TCP errors:** The red line shows events related to TCP errors. If you see an increase in this line, it could indicate retransmissions or failed connections.

## Example 2:

Now we observe using the filter “http.request.method == “POST” ”.

http.request.method == "POST"									
No.	Time	Source	Destination	Protocol	Src Port	Dest Port	Length	* Info	
20340	2024-11-25 23:50:45.849438	10.11.26.183	194.180.191.64	HTTP	53362	443	274	POST http://194.180.191.64/fakeurl.htm HTTP/1.1	(application/x-www-form-urlencoded)
26809	2024-11-26 00:40:28.001338	10.11.26.183	194.180.191.64	HTTP	53360	443	274	POST http://194.180.191.64/fakeurl.htm HTTP/1.1	(application/x-www-form-urlencoded)
20572	2024-11-25 23:51:46.921682	10.11.26.183	194.180.191.64	HTTP	53362	443	288	POST http://194.180.191.64/fakeurl.htm HTTP/1.1	(application/x-www-form-urlencoded)
21145	2024-11-25 23:52:47.103354	10.11.26.183	194.180.191.64	HTTP	53362	443	288	POST http://194.180.191.64/fakeurl.htm HTTP/1.1	(application/x-www-form-urlencoded)
21153	2024-11-25 23:53:47.260093	10.11.26.183	194.180.191.64	HTTP	53362	443	288	POST http://194.180.191.64/fakeurl.htm HTTP/1.1	(application/x-www-form-urlencoded)
21246	2024-11-25 23:54:47.427092	10.11.26.183	194.180.191.64	HTTP	53362	443	288	POST http://194.180.191.64/fakeurl.htm HTTP/1.1	(application/x-www-form-urlencoded)
21295	2024-11-25 23:55:47.485672	10.11.26.183	194.180.191.64	HTTP	53362	443	288	POST http://194.180.191.64/fakeurl.htm HTTP/1.1	(application/x-www-form-urlencoded)
21337	2024-11-25 23:56:47.546358	10.11.26.183	194.180.191.64	HTTP	53362	443	288	POST http://194.180.191.64/fakeurl.htm HTTP/1.1	(application/x-www-form-urlencoded)
21352	2024-11-25 23:57:47.608604	10.11.26.183	194.180.191.64	HTTP	53362	443	288	POST http://194.180.191.64/fakeurl.htm HTTP/1.1	(application/x-www-form-urlencoded)
21364	2024-11-25 23:58:47.763032	10.11.26.183	194.180.191.64	HTTP	53362	443	288	POST http://194.180.191.64/fakeurl.htm HTTP/1.1	(application/x-www-form-urlencoded)
21415	2024-11-25 23:59:47.818149	10.11.26.183	194.180.191.64	HTTP	53362	443	288	POST http://194.180.191.64/fakeurl.htm HTTP/1.1	(application/x-www-form-urlencoded)
21608	2024-11-26 00:00:48.022865	10.11.26.183	194.180.191.64	HTTP	53362	443	288	POST http://194.180.191.64/fakeurl.htm HTTP/1.1	(application/x-www-form-urlencoded)
21708	2024-11-26 00:01:48.318189	10.11.26.183	194.180.191.64	HTTP	53362	443	288	POST http://194.180.191.64/fakeurl.htm HTTP/1.1	(application/x-www-form-urlencoded)
21782	2024-11-26 00:02:48.487475	10.11.26.183	194.180.191.64	HTTP	53362	443	288	POST http://194.180.191.64/fakeurl.htm HTTP/1.1	(application/x-www-form-urlencoded)
21792	2024-11-26 00:03:48.558212	10.11.26.183	194.180.191.64	HTTP	53362	443	288	POST http://194.180.191.64/fakeurl.htm HTTP/1.1	(application/x-www-form-urlencoded)
21823	2024-11-26 00:04:48.715402	10.11.26.183	194.180.191.64	HTTP	53362	443	288	POST http://194.180.191.64/fakeurl.htm HTTP/1.1	(application/x-www-form-urlencoded)
25880	2024-11-26 00:05:48.896946	10.11.26.183	194.180.191.64	HTTP	53362	443	288	POST http://194.180.191.64/fakeurl.htm HTTP/1.1	(application/x-www-form-urlencoded)
25901	2024-11-26 00:06:48.946318	10.11.26.183	194.180.191.64	HTTP	53362	443	288	POST http://194.180.191.64/fakeurl.htm HTTP/1.1	(application/x-www-form-urlencoded)
25937	2024-11-26 00:07:49.001919	10.11.26.183	194.180.191.64	HTTP	53362	443	288	POST http://194.180.191.64/fakeurl.htm HTTP/1.1	(application/x-www-form-urlencoded)
25968	2024-11-26 00:08:49.062950	10.11.26.183	194.180.191.64	HTTP	53362	443	288	POST http://194.180.191.64/fakeurl.htm HTTP/1.1	(application/x-www-form-urlencoded)
25985	2024-11-26 00:09:49.216290	10.11.26.183	194.180.191.64	HTTP	53362	443	288	POST http://194.180.191.64/fakeurl.htm HTTP/1.1	(application/x-www-form-urlencoded)
26022	2024-11-26 00:10:49.369834	10.11.26.183	194.180.191.64	HTTP	53362	443	288	POST http://194.180.191.64/fakeurl.htm HTTP/1.1	(application/x-www-form-urlencoded)
26051	2024-11-26 00:11:49.523158	10.11.26.183	194.180.191.64	HTTP	53362	443	288	POST http://194.180.191.64/fakeurl.htm HTTP/1.1	(application/x-www-form-urlencoded)
26055	2024-11-26 00:12:49.680246	10.11.26.183	194.180.191.64	HTTP	53362	443	288	POST http://194.180.191.64/fakeurl.htm HTTP/1.1	(application/x-www-form-urlencoded)
26066	2024-11-26 00:13:49.737747	10.11.26.183	194.180.191.64	HTTP	53362	443	288	POST http://194.180.191.64/fakeurl.htm HTTP/1.1	(application/x-www-form-urlencoded)
26074	2024-11-26 00:14:49.880222	10.11.26.183	194.180.191.64	HTTP	53362	443	288	POST http://194.180.191.64/fakeurl.htm HTTP/1.1	(application/x-www-form-urlencoded)
26087	2024-11-26 00:15:49.936096	10.11.26.183	194.180.191.64	HTTP	53362	443	288	POST http://194.180.191.64/fakeurl.htm HTTP/1.1	(application/x-www-form-urlencoded)
26097	2024-11-26 00:16:50.090804	10.11.26.183	194.180.191.64	HTTP	53362	443	288	POST http://194.180.191.64/fakeurl.htm HTTP/1.1	(application/x-www-form-urlencoded)
26107	2024-11-26 00:17:50.148715	10.11.26.183	194.180.191.64	HTTP	53362	443	288	POST http://194.180.191.64/fakeurl.htm HTTP/1.1	(application/x-www-form-urlencoded)
26110	2024-11-26 00:18:50.207754	10.11.26.183	194.180.191.64	HTTP	53362	443	288	POST http://194.180.191.64/fakeurl.htm HTTP/1.1	(application/x-www-form-urlencoded)

We can repeat the same procedure for this analysis. In this case, there isn't much to do since it's a capture made as part of a practice.

However, I will mention the patterns to observe for HTTP POST:

- **Source IP:** 10.11.26.183 (internal host).
- **Destination IP:** 194.180.191.64 (external host).
- **Destination URL:** <http://194.180.191.64/fakeurl.htm>.
- **Content-Type:** application/x-www-form-urlencoded.
- **Port:** 443 (HTTPS).

For HTTPS traffic, we search in the filter “ssl” or “tls.handshake.type == 1” to identify TLS packets with SNI.

tls.handshake.type == 1									
No.	Time	Source	Destination	Protocol	Src Port	Dest Port	Length	* Info	
1058	2024-11-25 23:50:06.840908	10.11.26.183	13.107.246.57	TLSv1.3	53317	443	376	Client Hello (SNI=inputuggestions.msdxcdn.microsoft.com)	
20649	2024-11-25 23:51:58.304948	10.11.26.183	204.79.197.203	TLSv1.3	53377	443	379	Client Hello (SNI=www.msn.com)	
20772	2024-11-25 23:51:58.792127	10.11.26.183	204.79.197.203	TLSv1.3	53379	443	379	Client Hello (SNI=api.msn.com)	
20822	2024-11-25 23:51:59.021655	10.11.26.183	204.79.197.203	TLSv1.3	53380	443	380	Client Hello (SNI=srtb.msn.com)	
21427	2024-11-26 00:00:37.746118	10.11.26.183	23.204.171.61	TLSv1.2	53413	443	383	Client Hello (SNI=storecatalogrevocation.storequality.microsoft.com)	
1828	2024-11-25 23:50:14.915243	10.11.26.183	142.251.186.1	TLSv1.3	53333	443	386	Client Hello (SNI=www.google.com)	
2739	2024-11-25 23:50:15.797852	10.11.26.183	204.79.197.239	TLSv1.2	53338	443	386	Client Hello (SNI=edge.microsoft.com)	
1490	2024-11-25 23:50:14.488757	10.11.26.183	193.42.38.139	TLSv1.3	53327	443	388	Client Hello (SNI=modandcrackedapk.com)	
2983	2024-11-25 23:50:15.827162	10.11.26.183	193.42.38.139	TLSv1.3	53337	443	388	Client Hello (SNI=modandcrackedapk.com)	
2153	2024-11-25 23:50:15.396979	10.11.26.183	142.250.115.95	TLSv1.3	53335	443	391	Client Hello (SNI=maps.googleapis.com)	
965	2024-11-25 23:50:01.137006	10.11.26.183	13.107.5.93	TLSv1.3	53313	443	401	Change Cipher Spec, Client Hello (SNI=default.exp-tas.com)	
1028	2024-11-25 23:50:02.917874	10.11.26.183	13.107.5.93	TLSv1.3	53315	443	401	Change Cipher Spec, Client Hello (SNI=default.exp-tas.com)	
1062	2024-11-25 23:50:06.889668	10.11.26.183	13.107.246.57	TLSv1.3	53317	443	415	Change Cipher Spec, Client Hello (SNI=inputuggestions.msdxcdn.microsoft.com)	
20623	2024-11-25 23:51:57.846042	10.11.26.183	204.79.197.203	TLSv1.3	53376	443	415	Client Hello (SNI=windows.msn.com)	
1188	2024-11-25 23:50:11.399775	10.11.26.183	13.107.21.239	TLSv1.2	53321	443	418	Client Hello (SNI=edge.microsoft.com)	
12134	2024-11-25 23:50:23.379240	10.11.26.183	204.79.197.239	TLSv1.2	53354	443	418	Client Hello (SNI=edge.microsoft.com)	
3810	2024-11-25 23:50:16.574611	10.11.26.183	142.250.113.94	TLSv1.3	53344	443	419	Client Hello (SNI=www.gstatic.com)	
25984	2024-11-26 00:09:49.178852	10.11.26.183	20.189.173.16	TLSv1.3	53445	443	419	Change Cipher Spec, Client Hello (SNI=mobile.events.data.microsoft.com)	
647	2024-11-25 23:49:57.726676	10.11.26.183	204.79.197.203	TLSv1.3	53308	443	420	Change Cipher Spec, Client Hello (SNI=www.msn.com)	
1407	2024-11-25 23:50:14.403860	10.11.26.183	193.42.38.139	TLSv1.3	53326	443	420	Client Hello (SNI=modandcrackedapk.com)	
1772	2024-11-25 23:50:14.849040	10.11.26.183	142.250.138.94	TLSv1.3	53329	443	421	Client Hello (SNI=fonts.gstatic.com)	
1779	2024-11-25 23:50:14.856007	10.11.26.183	142.250.138.94	TLSv1.3	53330	443	421	Client Hello (SNI=fonts.gstatic.com)	
1366	2024-11-25 23:50:14.310565	10.11.26.183	52.8.34.0	TLSv1.3	53325	443	423	Client Hello (SNI=confirmsubscription.com)	
2510	2024-11-25 23:50:15.726675	10.11.26.183	142.250.115.95	TLSv1.3	53336	443	423	Client Hello (SNI=maps.googleapis.com)	
452	2024-11-25 23:49:57.357302	10.11.26.183	204.79.197.203	TLSv1.3	53300	443	424	Change Cipher Spec, Client Hello (SNI=windows.msn.com)	
623	2024-11-25 23:49:57.615857	10.11.26.183	40.126.29.9	TLSv1.3	53317	443	434	Change Cipher Spec, Client Hello (SNI=login.microsoftonline.com)	
21832	2024-11-26 00:04:54.231822	10.11.26.183	20.189.173.26	TLSv1.3	53427	443	443	Client Hello (SNI=mobile.events.data.microsoft.com)	
26454	2024-11-26 00:34:54.111943	10.11.26.183	20.189.173.26	TLSv1.3	53481	443	443	Client Hello (SNI=mobile.events.data.microsoft.com)	
1206	2024-11-25 23:50:11.512916	10.11.26.183	213.246.109.5	TLSv1.3	53322	443	448	Client Hello (SNI=classicgrand.com)	
3613	2024-11-25 23:50:16.274450	10.11.26.183	18.160.156.61	TLSv1.3	53340	443	450	Client Hello (SNI=js.createsend.com)	
3795	2024-11-25 23:50:16.533605	10.11.26.183	13.107.21.239	TLSv1.2	53342	443	450	Client Hello (SNI=edge.microsoft.com)	
3611	2024-11-25 23:50:16.274241	10.11.26.183	18.160.156.103	TLSv1.3	53341	443	451	Client Hello (SNI=css.createsend.com)	



We can observe at first glance that there are common SNIs in the domains:

- inputsuggestions.msdxcdn.microsoft.com
- [www.msn.com](http://www.msn.com)
- edge.microsoft.com
- storecatalogrevocation.storequality.microsoft.com
- mobile.events.data.microsoft.com

These domains are legitimate and associated with Microsoft services such as Edge, MSN, and Windows updates.

The traffic appears to be related to automatic updates and basic functions of the operating system.

#### Google and related services:

- [www.google.com](http://www.google.com)
- maps.googleapis.com
- fonts.gstatic.com

These are also legitimate domains used for maps, fonts, and web navigation services. No suspicious activity.

#### Uncommon or potentially suspicious domains:

- modandcrackedapk.com

This domain is related to modified or pirated software, which represents a potential security risk.

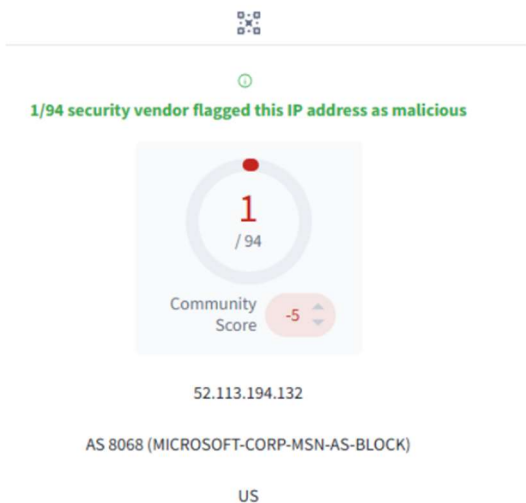
- confirmsubscription.com

Although not necessarily malicious, it could be associated with email or notification systems. It would be best to authenticate this domain.

### Example 1:

Time	Source	Destination	Protocol	Src Port	Dest Port	Length	Info
20779 2024-11-25 23:51:58.861647	10.11.26.183	204.79.197.203	TLSv1.3	53379	443	621	Change Cipher Spec, Client Hello
20842 2024-11-25 23:51:59.092213	10.11.26.183	204.79.197.203	TLSv1.3	53380	443	622	Change Cipher Spec, Client Hello
20627 2024-11-25 23:51:57.911883	10.11.26.183	204.79.197.203	TLSv1.3	53376	443	657	Change Cipher Spec, Client Hello
12376 2024-11-25 23:50:26.773909	10.11.26.183	213.246.109.5	TLSv1.3	53356	443	751	Client Hello (SNI=classicgrand.co
20501 2024-11-25 23:51:02.988273	10.11.26.183	52.113.194.132	TLSv1.3	53370	443	781	Client Hello (SNI=ecs.office.com)
21305 2024-11-25 23:56:03.469275	10.11.26.183	52.113.194.132	TLSv1.3	53400	443	781	Client Hello (SNI=ecs.office.com)

In this example, we have a domain with the SNI “ecs.office.com”. We verify the destination IP on the VirusTotal.com community and get the following report.



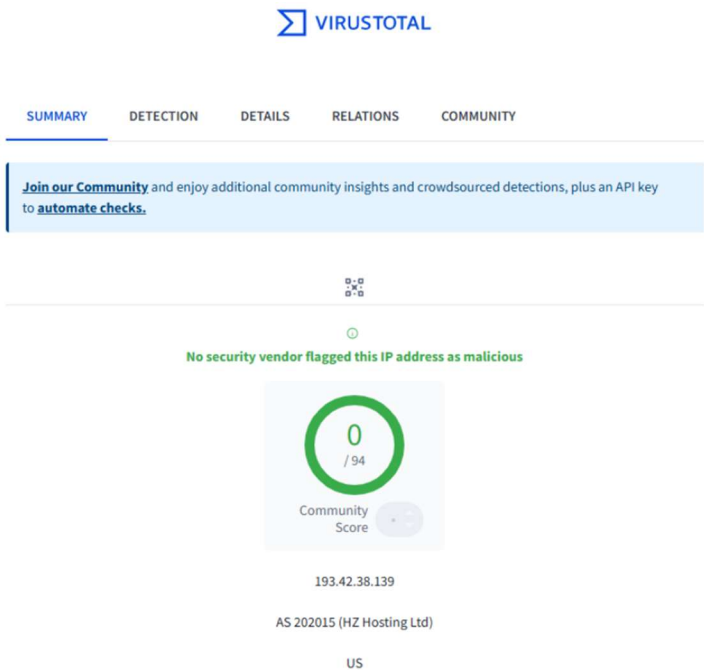
We can observe the presence of malware.

Example 2:

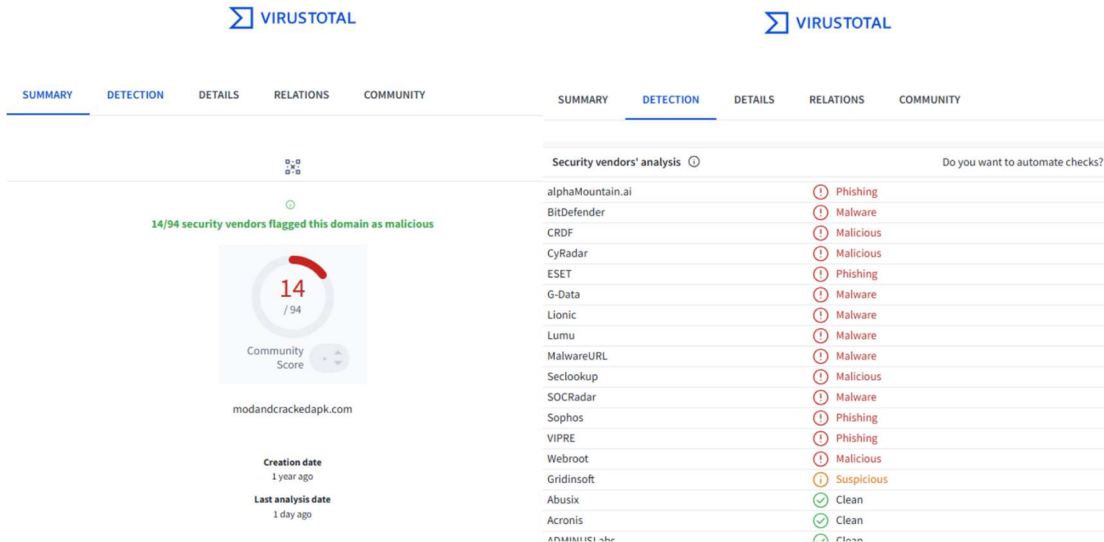
Now let’s proceed with one of the domains I mentioned earlier.

Time	Source	Destination	Protocol	Src Port	Dest Port	Length	Info
2739	2024-11-25 23:50:15.797852	10.11.26.183 204.79.197.239	TLSv1.2	53338	443	386	Client Hello (SNI=edge.microsoft.com)
1490	2024-11-25 23:50:14.488757	10.11.26.183 193.42.38.139	TLSv1.3	53327	443	388	Client Hello (SNI=modandcrackedapk.com)
2983	2024-11-25 23:50:15.827162	10.11.26.183 193.42.38.139	TLSv1.3	53337	443	388	Client Hello (SNI=modandcrackedapk.com)

When investigating the destination IP, I found nothing relevant or malicious.



However, with the domain, we do have a report indicating malicious activity.



On the other hand, we will mention the protocols and versions, and their differences:

- **TLShv1.2 and TLShv1.3:**

Most connections use TLShv1.3, which is positive due to its enhanced security features. Some connections still use TLShv1.2 (edge.microsoft.com, among others), which could be a concern if these versions are not updated.

## Filter SMB traffic:

SMB (port 445) is a protocol used for sharing files, printers, and other resources on a network. In Wireshark, we can filter SMB traffic using this filter:

“smb || tcp.port == 445”.

smb    tcp.port == 445									
Time	Source	Destination	Protocol	Src Port	Dest Port	Length	Info		
253	2024-11-25 23:49:55.580323	10.11.26.183 10.11.26.3	TCP	53290	445	60	53290 → 445 [ACK] Seq=1 Ack=1 Win=1049600 Len=0		
318	2024-11-25 23:49:55.587361	10.11.26.3 10.11.26.183	TCP	445	53290	60	445 → 53290 [ACK] Seq=629 Ack=4120 Win=1049600 Len=0		
323	2024-11-25 23:49:55.589349	10.11.26.183 10.11.26.3	TCP	53290	445	60	53290 → 445 [ACK] Seq=4524 Ack=1398 Win=1048320 Len=0		
407	2024-11-25 23:49:55.641799	10.11.26.183 10.11.26.3	TCP	53290	445	60	53290 → 445 [ACK] Seq=8085 Ack=5502 Win=1048576 Len=0		
411	2024-11-25 23:49:55.829127	10.11.26.183 10.11.26.3	TCP	53298	445	60	53298 → 445 [ACK] Seq=1 Ack=1 Win=1049600 Len=0		
429	2024-11-25 23:49:55.834655	10.11.26.3 10.11.26.183	TCP	445	53298	60	445 → 53298 [ACK] Seq=377 Ack=3201 Win=1049600 Len=0		
435	2024-11-25 23:49:55.892942	10.11.26.183 10.11.26.3	TCP	53298	445	60	53298 → 445 [ACK] Seq=4124 Ack=990 Win=1048576 Len=0		
664	2024-11-25 23:49:57.797117	10.11.26.3 10.11.26.183	TCP	445	53298	60	445 → 53298 [ACK] Seq=990 Ack=4632 Win=1048064 Len=0		
667	2024-11-25 23:49:57.797595	10.11.26.183 10.11.26.3	TCP	53298	445	60	53298 → 445 [ACK] Seq=4632 Ack=1310 Win=1048320 Len=0		
680	2024-11-25 23:49:57.846657	10.11.26.183 10.11.26.3	TCP	53298	445	60	53298 → 445 [ACK] Seq=4920 Ack=1514 Win=1049600 Len=0		
747	2024-11-25 23:49:58.223157	10.11.26.183 10.11.26.3	TCP	53298	445	60	53298 → 445 [ACK] Seq=5192 Ack=1886 Win=1049344 Len=0		
793	2024-11-25 23:49:58.786683	10.11.26.183 10.11.26.3	TCP	53298	445	60	53298 → 445 [ACK] Seq=6280 Ack=3374 Win=1049344 Len=0		
1051	2024-11-25 23:50:06.280979	10.11.26.183 10.11.26.3	TCP	53290	445	60	53290 → 445 [RST, ACK] Seq=8229 Ack=5646 Win=0 Len=0		
12586	2024-11-25 23:50:28.739506	10.11.26.183 10.11.26.3	TCP	53298	445	60	[TCP Keep-Alive] 53298 → 445 [ACK] Seq=6279 Ack=3374 Win=1049344 Len=1		
12600	2024-11-25 23:50:29.581792	10.11.26.183 10.11.26.3	TCP	53298	445	60	53298 → 445 [ACK] Seq=6824 Ack=4118 Win=1048576 Len=0		
12700	2024-11-25 23:50:35.124842	10.11.26.183 10.11.26.3	TCP	53298	445	60	53298 → 445 [ACK] Seq=7152 Ack=4194 Win=1048320 Len=0		
12703	2024-11-25 23:50:37.893080	10.11.26.183 10.11.26.3	TCP	53357	445	60	53357 → 445 [ACK] Seq=1 Ack=1 Win=1049600 Len=0		
12711	2024-11-25 23:50:37.895940	10.11.26.3 10.11.26.183	TCP	445	53357	60	445 → 53357 [ACK] Seq=629 Ack=3400 Win=1049600 Len=0		
12719	2024-11-25 23:50:37.899489	10.11.26.183 10.11.26.3	TCP	53357	445	60	53357 → 445 [ACK] Seq=4644 Ack=1460 Win=1048064 Len=0		
20369	2024-11-25 23:50:48.629416	10.11.26.183 10.11.26.3	TCP	53357	445	60	53357 → 445 [RST, ACK] Seq=4788 Ack=1604 Win=0 Len=0		
20515	2024-11-25 23:51:05.083596	10.11.26.183 10.11.26.3	TCP	53298	445	60	[TCP Keep-Alive] 53298 → 445 [ACK] Seq=7151 Ack=4194 Win=1048320 Len=1		
20549	2024-11-25 23:51:35.087792	10.11.26.183 10.11.26.3	TCP	53298	445	60	[TCP Keep-Alive] 53298 → 445 [ACK] Seq=7151 Ack=4194 Win=1048320 Len=1		
21138	2024-11-25 23:52:35.129385	10.11.26.3 10.11.26.183	TCP	445	53298	60	[TCP Keep-Alive] 445 → 53298 [ACK] Seq=4193 Ack=7152 Win=1048576 Len=1		
21240	2024-11-25 23:54:35.134499	10.11.26.3 10.11.26.183	TCP	445	53298	60	[TCP Keep-Alive] 445 → 53298 [ACK] Seq=4193 Ack=7152 Win=1048576 Len=1		
21288	2024-11-25 23:55:05.428760	10.11.26.183 10.11.26.3	TCP	53298	445	60	53298 → 445 [ACK] Seq=7276 Ack=4462 Win=1049600 Len=0		
21291	2024-11-25 23:55:35.480780	10.11.26.183 10.11.26.3	TCP	53298	445	60	[TCP Keep-Alive] 53298 → 445 [ACK] Seq=7275 Ack=4462 Win=1049600 Len=1		
21329	2024-11-25 23:56:05.480344	10.11.26.183 10.11.26.3	TCP	53298	445	60	[TCP Keep-Alive] 53298 → 445 [ACK] Seq=7275 Ack=4462 Win=1049600 Len=1		
21343	2024-11-25 23:57:05.431749	10.11.26.3 10.11.26.183	TCP	445	53298	60	[TCP Keep-Alive] 445 → 53298 [ACK] Seq=4461 Ack=7276 Win=1048576 Len=1		
21368	2024-11-25 23:59:05.436402	10.11.26.3 10.11.26.183	TCP	445	53298	60	[TCP Keep-Alive] 445 → 53298 [ACK] Seq=4461 Ack=7276 Win=1048576 Len=1		
21705	2024-11-26 00:01:05.451841	10.11.26.3 10.11.26.183	TCP	445	53298	60	[TCP Keep-Alive] 445 → 53298 [ACK] Seq=4461 Ack=7276 Win=1048576 Len=1		
21788	2024-11-26 00:03:05.462117	10.11.26.3 10.11.26.183	TCP	445	53298	60	[TCP Keep-Alive] 445 → 53298 [ACK] Seq=4461 Ack=7276 Win=1048576 Len=1		
25835	2024-11-26 00:05:05.466735	10.11.26.183 10.11.26.3	TCP	445	53298	60	[TCP Keep-Alive] 445 → 53298 [ACK] Seq=4461 Ack=7276 Win=1048576 Len=1		
25839	2024-11-26 00:05:05.766934	10.11.26.183 10.11.26.3	TCP	53298	445	60	53298 → 445 [ACK] Seq=7400 Ack=4730 Win=1049344 Len=0		
25851	2024-11-26 00:05:05.735858	10.11.26.183 10.11.26.3	TCP	53298	445	60	[TCP Keep-Alive] 53298 → 445 [ACK] Seq=7399 Ack=4730 Win=1049344 Len=1		
25855	2024-11-26 00:05:37.880909	10.11.26.183 10.11.26.3	TCP	53442	445	60	53442 → 445 [ACK] Seq=1 Ack=1 Win=131328 Len=0		
25863	2024-11-26 00:05:37.886288	10.11.26.3 10.11.26.183	TCP	445	53442	60	445 → 53442 [ACK] Seq=629 Ack=3400 Win=1049600 Len=0		
25871	2024-11-26 00:05:37.890902	10.11.26.183 10.11.26.3	TCP	53442	445	60	53442 → 445 [ACK] Seq=4644 Ack=1460 Win=129792 Len=0		
25879	2024-11-26 00:05:48.610626	10.11.26.183 10.11.26.3	TCP	53442	445	60	53442 → 445 [RST, ACK] Seq=4788 Ack=1604 Win=0 Len=0		
25884	2024-11-26 00:06:05.746948	10.11.26.183 10.11.26.3	TCP	53298	445	60	[TCP Keep-Alive] 53298 → 445 [ACK] Seq=7399 Ack=4730 Win=1049344 Len=1		
25930	2024-11-26 00:07:05.770702	10.11.26.3 10.11.26.183	TCP	445	53298	60	[TCP Keep-Alive] 445 → 53298 [ACK] Seq=4729 Ack=7400 Win=1048320 Len=1		

To understand the structure of SMB traffic, I'll explain a bit about what we can observe in the capture.

- **Source and Destination:** This indicates that the machine is sending and receiving the packets.
- **Protocol:** We can observe a normal result based on the filter we applied, because SMB packets operate over TCP.
- **Ports:** In the Src Port and Dest Port section, it shows the ports used.
- **Info:** Details about the packets (ACK, RST, Keep-Alive).

## 1. Identify normal patterns:

In typical SMB traffic, you should see:

- **Established connection:** A TCP handshake (SYN, SYN-ACK, ACK) followed by SMB transfers.
- **Keep-Alive:** Packets to keep the connection active.
- **SMB Data:** Packets with specific operations (Tree Connect, Read, Write).

In the capture, we can observe:

- **ACK:** Packets confirming reception.
- **RST, ACK:** Packets that terminate the connection unexpectedly.

## 2. Identify anomalies:

- Based on the capture, here are potential anomalies:
- **RST (Reset) packets:** The packets that appear in red (RST, ACK) indicate that one of the machines is closing the connection abruptly.
- **Investigation:** This could be a legitimate interruption or a failed connection attempt. We can inspect previous packets to identify if there were any errors.
- **Keep-Alive repetitions:**

Many connections show TCP Keep-Alive packets. This could be normal, but if there is an excess, it might indicate a communication issue.

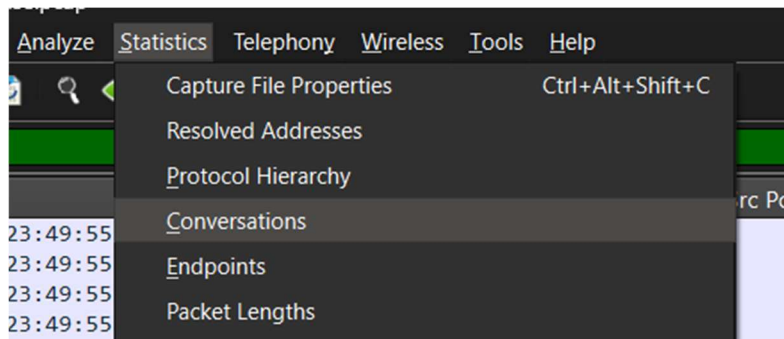
- **Investigation:** We can check if these connections are consuming resources unusually or if they come from a suspicious source.

### 3. Unusual source ports:

- Although destination port 445 is expected, check the source ports (53290, 53357, etc.) to identify if there are any strange patterns.
- **Investigation:** We can verify if the source ports change rapidly (this could indicate a port scan).

### 4. Tools for deeper analysis:

Wireshark has several tools we've mentioned earlier, such as viewing traffic statistics. In this case, we'll select one of the packets marked in red and go to the **Statistics > Conversations** tab.



And a tab like this will open.

Wireshark - Conversations - 2024-11-26-traffic-analysis-exercise.pcap

Conversation Settings

Name resolution

Absolute start time

Limit to display filter

Copy

Follow Stream...

Graph...

Protocol

Bluetooth

Bluetooth

Bluetooth

Bluetooth

Bluetooth

Bluetooth

Bluetooth

Bluetooth

Bluetooth

Bluetooth

Bluetooth

Bluetooth

Bluetooth

Bluetooth

Bluetooth

Bluetooth

Bluetooth

Bluetooth

Bluetooth

Bluetooth

Bluetooth

Bluetooth

Bluetooth

Bluetooth

Bluetooth

Bluetooth

Bluetooth

Bluetooth

Bluetooth

Bluetooth

Bluetooth

Bluetooth

Bluetooth

Bluetooth

Bluetooth

Bluetooth

Bluetooth

Bluetooth

Bluetooth

Bluetooth

Bluetooth

Bluetooth

Bluetooth

Bluetooth

Bluetooth

Bluetooth

Bluetooth

Bluetooth

Bluetooth

Bluetooth

Bluetooth

Bluetooth

Bluetooth

Bluetooth

Bluetooth

Bluetooth

Bluetooth

Bluetooth

Bluetooth

Bluetooth

Bluetooth

Bluetooth

Bluetooth

Bluetooth

Bluetooth

Bluetooth

Bluetooth

Bluetooth

Bluetooth

Bluetooth

Bluetooth

Bluetooth

Bluetooth

Bluetooth

Bluetooth

Bluetooth

Bluetooth

Bluetooth

Bluetooth

Bluetooth

Bluetooth

Bluetooth

Bluetooth

Bluetooth

Bluetooth

Bluetooth

Bluetooth

Bluetooth

Bluetooth

Bluetooth

Bluetooth

Bluetooth

Bluetooth

Bluetooth

Bluetooth

Bluetooth

Bluetooth

Bluetooth

Bluetooth

Bluetooth

Bluetooth

Bluetooth

Bluetooth

Bluetooth

Bluetooth

Bluetooth

Bluetooth

Bluetooth

Bluetooth

Bluetooth

Bluetooth

Bluetooth

Bluetooth

Bluetooth

Bluetooth

Bluetooth

Bluetooth

Bluetooth

Bluetooth

Bluetooth

Bluetooth

Bluetooth

Bluetooth

Bluetooth

Bluetooth

Bluetooth

Bluetooth

Bluetooth

Bluetooth

Bluetooth

Bluetooth

Bluetooth

Bluetooth

Bluetooth

Bluetooth

Bluetooth

Bluetooth

Bluetooth

Bluetooth

Bluetooth

Bluetooth

Bluetooth

Bluetooth

Bluetooth

Bluetooth

Bluetooth

Bluetooth

Bluetooth

Bluetooth

Bluetooth

Bluetooth

Bluetooth

Bluetooth

Bluetooth

Bluetooth

Bluetooth

Bluetooth

Bluetooth

Bluetooth

Bluetooth

Bluetooth

Bluetooth

Bluetooth

Bluetooth

Bluetooth

Bluetooth

Bluetooth

Bluetooth

Bluetooth

Bluetooth

Bluetooth

Bluetooth

Bluetooth

Bluetooth

Bluetooth

Bluetooth

Bluetooth

Bluetooth

Bluetooth

Bluetooth

Bluetooth

Bluetooth

Bluetooth

Bluetooth

Bluetooth

Bluetooth

Bluetooth

Bluetooth

Bluetooth

Bluetooth

Bluetooth

Bluetooth

Bluetooth

Bluetooth

Bluetooth

Bluetooth

Bluetooth

Bluetooth

Bluetooth

Bluetooth

Bluetooth

Bluetooth

Bluetooth

Bluetooth

Bluetooth

Bluetooth

Bluetooth

Bluetooth

Bluetooth

Bluetooth

Bluetooth

Bluetooth

Bluetooth

Bluetooth

Bluetooth

Bluetooth

Bluetooth

Bluetooth

Bluetooth

Bluetooth

Bluetooth

Bluetooth

Bluetooth

Bluetooth

Bluetooth

Bluetooth

Bluetooth

Bluetooth

Bluetooth

Bluetooth

Bluetooth

Bluetooth

Bluetooth

Bluetooth

Bluetooth

Bluetooth

Bluetooth

Bluetooth

Bluetooth

Bluetooth

Bluetooth

Bluetooth

Bluetooth

Bluetooth

Bluetooth

Bluetooth

Bluetooth

Bluetooth

Bluetooth

Bluetooth

Bluetooth

Bluetooth

Bluetooth

Bluetooth

Bluetooth

Bluetooth

Bluetooth

Bluetooth

Bluetooth

Bluetooth

Bluetooth

Bluetooth

Bluetooth

Bluetooth

Bluetooth

Bluetooth

Bluetooth

Bluetooth

Bluetooth

Bluetooth

Bluetooth

Bluetooth

Bluetooth

Bluetooth

Bluetooth

Bluetooth

Bluetooth

Bluetooth

Bluetooth

Bluetooth

Bluetooth

<

What you observe in **Statistics > Conversations** in Wireshark shows multiple connections between the same source IP address (10.11.26.183) and the destination IP (10.11.26.3) on port 445 (SMB - Server Message Block). This repetitive traffic could indicate:

1. **Legitimate SMB activity:** If these devices are on a corporate or test network, there could be normal SMB traffic, such as authentications or file transfers.
2. **Brute force attacks or port scans:** If the traffic repeats excessively in short intervals, someone might be trying to brute-force credentials on SMB.
3. **Malware propagation:** Some malware, such as WannaCry or EternalBlue, exploits vulnerabilities in SMB and generates unusual traffic on port 445.

## What to do?

- **Filter by IP and review the packets:** Use the filter `ip.addr == 10.11.26.183 && tcp.port == 445` to see details of the traffic.
- **Check connection patterns:** If the connections have failed authentication attempts, it could be a brute force attack.
- **Review packet timings and sizes:** Legitimate SMB traffic usually transfers files, while malicious traffic may send small packets with suspicious commands.

I filtered the suspicious IP address to analyze further and came across the following result:

Ethernet - 6	IPv4 - 60	IPv6	TCP - 142	UDP - 134										
Address A →	Port A	Address B	Port B	Packets	Bytes	Stream ID	Total Packets	Percent Filtered	Packets A → B	Bytes A → B	Packets B → A	Bytes B → A	Rel Start	Duration
10.11.26.183	53282	4.149.227.78	443	35	15 kB	3	35	100.00%	17	4 kB	18	11 kB	2.738780	0.4859
10.11.26.183	53283	4.149.227.78	443	34	15 kB	4	34	100.00%	17	4 kB	17	11 kB	3.139470	0.4624
10.11.26.183	53311	4.150.155.223	443	21	10 kB	31	21	100.00%	10	2 kB	11	8 kB	21.815305	0.4295
10.11.26.183	53312	4.150.155.223	443	22	11 kB	32	22	100.00%	11	2 kB	11	9 kB	22.865588	0.4086
10.11.26.183	53280	10.11.26.3	389	15	6 kB	1	15	100.00%	8	3 kB	7	3 kB	0.570664	0.0060
10.11.26.183	53281	10.11.26.3	389	6	372 bytes	2	6	100.00%	3	186 bytes	3	186 bytes	0.570716	0.0008
10.11.26.183	53286	10.11.26.3	88	8	898 bytes	6	8	100.00%	4	467 bytes	4	431 bytes	16.964457	0.0018
10.11.26.183	53287	10.11.26.3	88	10	3 kB	7	10	100.00%	5	607 bytes	5	2 kB	16.970538	0.0026
10.11.26.183	53288	10.11.26.3	88	13	4 kB	8	13	100.00%	6	2 kB	7	2 kB	16.973351	0.0034
10.11.26.183	53289	10.11.26.3	135	14	2 kB	9	14	100.00%	8	964 bytes	6	1 kB	17.121276	13.4971
10.11.26.183	53290	10.11.26.3	445	63	17 kB	10	63	100.00%	34	10 kB	29	7 kB	17.122009	10.7089
10.11.26.183	53291	10.11.26.3	49671	22	6 kB	11	22	100.00%	12	4 kB	10	2 kB	17.122876	33.4894
10.11.26.183	53292	10.11.26.3	88	12	4 kB	12	12	100.00%	6	2 kB	6	2 kB	17.124356	0.0026
10.11.26.183	53293	10.11.26.3	88	12	4 kB	13	12	100.00%	6	2 kB	6	2 kB	17.124569	0.0022
10.11.26.183	53294	10.11.26.3	88	12	4 kB	14	12	100.00%	6	2 kB	6	2 kB	17.127182	0.0014
10.11.26.183	53295	10.11.26.3	49693	12	3 kB	15	12	100.00%	7	2 kB	5	1 kB	17.132732	13.4857
10.11.26.183	53296	10.11.26.3	389	19	8 kB	16	19	100.00%	10	4 kB	9	4 kB	17.135846	0.0212
10.11.26.183	53297	10.11.26.3	389	15	4 kB	17	15	100.00%	8	3 kB	7	929 bytes	17.144210	0.0104
10.11.26.183	53298	10.11.26.3	445	153	22 kB	18	153	100.00%	83	13 kB	70	10 kB	17.370698	3190.9889
10.11.26.183	53299	10.11.26.3	88	12	4 kB	19	12	100.00%	6	2 kB	6	2 kB	17.372243	0.0038
10.11.26.183	53303	10.11.26.3	135	16	3 kB	23	16	100.00%	10	1 kB	6	2 kB	18.965094	14.2702
10.11.26.183	53304	10.11.26.3	49671	36	10 kB	24	36	100.00%	20	6 kB	16	4 kB	18.967733	14.2676
10.11.26.183	53305	10.11.26.3	389	21	8 kB	25	21	100.00%	11	4 kB	10	4 kB	18.979421	0.0315
10.11.26.183	53306	10.11.26.3	389	18	7 kB	26	18	100.00%	9	3 kB	9	4 kB	19.006320	0.0045
10.11.26.183	53318	10.11.26.3	139	24	4 kB	37	24	100.00%	13	2 kB	11	1 kB	31.958521	10.7013
10.11.26.183	53319	10.11.26.3	389	17	4 kB	38	17	100.00%	9	3 kB	8	2 kB	32.552332	0.0131
10.11.26.183	53320	10.11.26.3	389	18	7 kB	39	18	100.00%	9	3 kB	9	4 kB	32.559300	0.0060
10.11.26.183	53357	10.11.26.3	445	24	8 kB	72	24	100.00%	14	6 kB	10	2 kB	59.434541	10.7367
10.11.26.183	53359	10.11.26.3	139	24	4 kB	74	24	100.00%	13	2 kB	11	2	61.995516	13.5519
10.11.26.183	53366	10.11.26.3	135	10	2 kB	80	10	100.00%	6	844 bytes	4	872 bytes	78.068289	0.0132

Key observation:

- 1. Connections to multiple HTTPS servers (port 443):**
  - Connections to 4.149.227.78 and 4.150.155.223 through port 443 (HTTPS) with data flow of around 15 kB in packets. This be legitimate web traffic or malicious activity if connecting to multiple servers in a short time.



**2. LDAP/Active Directory connections (port 389):**

- There are connection attempts to the server 10.11.26.3 in port 389 (LDAP). LDAP is used to authentication and directory management. A high number of connections could indicate user enumeration attempts or stolen credentials.

**3. Traffic on SMB and NetBIOS ports (445, 139, 135):**

- Port 445 (SMB) with 153 packets and 22kB of data transferred.
- Port 139 (NetBIOS) and 135 (RPC) with multiple connections.
- This could indicate attempts to explore or access Windows shared resources.

**4. Multiple connections on port 88 (Kerberos):**

- Multiple connection attempts to port 88, which is used Active Directory authentication. A high number of connections here could indicate authentication attacks, such as Pass-the-Ticket or Kerberoasting.

**5. Long durations in some sessions:**

- Some connections last more than 3,000 seconds, suggesting persistent sessions or prolonged activity on the system. This could be legitimate data transfers or attempts at data exfiltration.