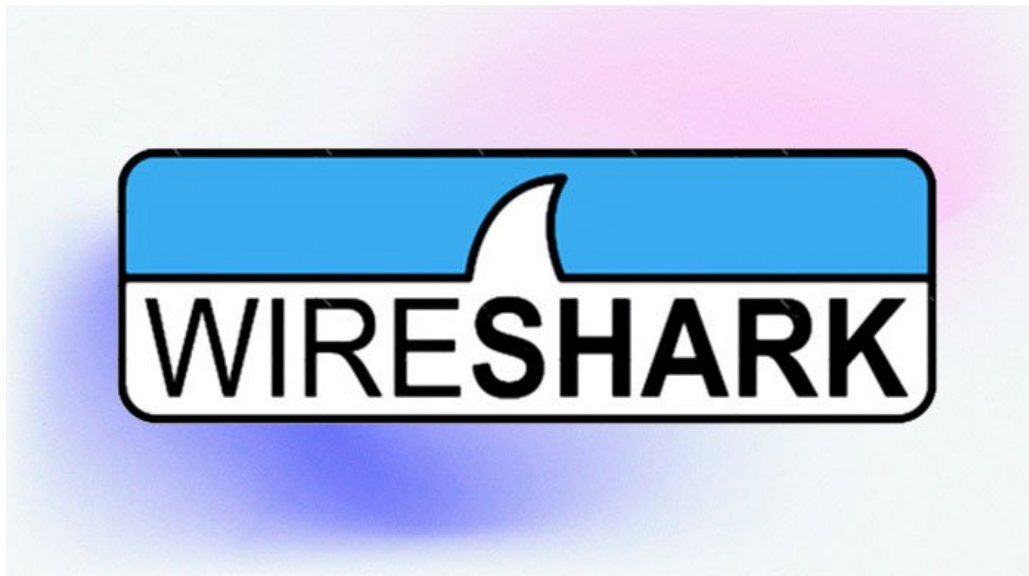


Capture and Analyze Network Traffic Using Wireshark.



- **Installation Wireshark and use Wireshark to find DNS Traffics.**
- **Starting Capture on Your Active Network Interface**

Sample : <https://www.kali.org/> (for Htpps)

No.	Time	Source	Destination	Protocol	Length	Info
7178	28.173720	192.168.31.1	192.168.31.137	TCP	54	53 → 38962 [ACK] Seq=1 Ack=3 Win=64256 Len=0
7179	28.173720	192.168.31.1	192.168.31.137	TCP	54	53 → 64440 [ACK] Seq=1 Ack=3 Win=64256 Len=0
7180	28.173835	192.168.31.137	192.168.31.1	TCP	54	[TCP ZeroWindow] 38962 → 53 [ACK] Seq=34 Ack=66 Win=0 Len=0
7181	28.173949	192.168.31.137	192.168.31.1	TCP	54	64440 → 53 [ACK] Seq=41 Ack=69 Win=65280 Len=0
7182	28.174072	192.168.31.137	192.168.31.1	TCP	54	64440 → 53 [FIN, ACK] Seq=41 Ack=69 Win=65280 Len=0
7183	28.174242	192.168.31.137	192.168.31.1	TCP	54	26936 → 53 [FIN, ACK] Seq=41 Ack=57 Win=65280 Len=0
7184	28.174441	192.168.31.1	192.168.31.137	DNS	203	Standard query response 0x1f82 HTTPS www.kali.org HTTPS
7185	28.174631	192.168.31.137	192.168.31.1	TCP	54	27308 → 53 [FIN, ACK] Seq=33 Ack=150 Win=65280 Len=0
7186	28.174920	192.168.31.137	51.8.207.171	TCP	54	14021 → 443 [FIN, ACK] Seq=3165 Ack=5899 Win=64768 Len=0
7187	28.174976	192.168.31.137	51.8.207.171	TCP	54	14021 → 443 [RST, ACK] Seq=3166 Ack=5899 Win=0 Len=0
7188	28.175281	2409:4091:10:c209:e...	2606:4700:8d90:747e...	TCP	86	35341 → 443 [SYN] Seq=0 Win=65535 Len=0 MSS=1440 WS=256 SACK_PERM
7189	28.175564	2409:4091:10:c209:e...	2606:4700:8d90:747e...	TCP	86	18282 → 443 [SYN] Seq=0 Win=65535 Len=0 MSS=1440 WS=256 SACK_PERM
7190	28.175846	192.168.31.1	192.168.31.137	TCP	54	53 → 27308 [ACK] Seq=1 Ack=3 Win=64256 Len=0
7191	28.175846	192.168.31.1	192.168.31.137	DNS	151	Standard query response 0x3247 HTTPS fonts.googleapis.com SOA ns1.google.com
7192	28.175869	192.168.31.137	192.168.31.1	TCP	54	[TCP Dup ACK 7185#1] 27308 → 53 [ACK] Seq=34 Ack=150 Win=65280 Len=0
7193	28.175930	192.168.31.1	192.168.31.137	TCP	54	53 → 26036 [ACK] Seq=1 Ack=3 Win=64256 Len=0
7194	28.175930	192.168.31.1	192.168.31.137	TCP	54	53 → 30688 [ACK] Seq=1 Ack=3 Win=64256 Len=0
7195	28.175930	192.168.31.1	192.168.31.137	TCP	54	53 → 38962 [RST] Seq=66 Win=0 Len=0
7196	28.175930	192.168.31.1	192.168.31.137	TCP	54	53 → 31768 [ACK] Seq=1 Ack=3 Win=64256 Len=0

<pre> Frame 7184: Packet, 203 bytes on wire (1624 bits), 203 bytes captured (1624 bits) Ethernet II, Src: ServercomPri_2a:5b:2b (28:a9:15:2a:5b:2b), Dst: ce:0e:6e:72:84:3c (ce:0e:6e:72:84:3c) Internet Protocol Version 4, Src: 192.168.31.1, Dst: 192.168.31.137 Transmission Control Protocol, Src Port: 53, Dst Port: 27308, Seq: 1, Ack: 33, Len: 149 Source Port: 53 Destination Port: 27308 [Stream index: 18] [Stream Packet Number: 7] [Conversation completeness: Complete, WITH_DATA (31)] [TCP Segment Len: 149] Sequence Number: 1 (relative sequence number) Sequence Number (raw): 949772418 [Next Sequence Number: 150 (relative sequence number)] Acknowledgment Number: 33 (relative ack number) Acknowledgment number (raw): 945625778 0101 = Header Length: 20 bytes (5) Flags: 0x018 (PSH, ACK) Window: 1800 [Calculated window size: 64256] [Window size scaling factor: 64] Checksum: 0x3247 (correct) </pre>	<pre> 0000 ce 0e 6e 72 84 3c 28 a9 15 2a 5b 2b 08 00 45 00 ...n<([+- E 0010 00 bd 5a 85 40 00 40 06 1f db c0 a8 1f 01 c0 a8 ...Z @ @ 0020 1f 89 00 35 6a ac 38 9c 60 82 38 5d 1a b2 50 18 ...5j 8 . 8] . P 0030 03 ec 22 a0 00 00 93 1f 82 81 80 00 01 00 01 ... 0040 00 00 00 00 03 77 77 77 04 6b 61 6c 69 03 6f 72 ...www.kali.or 0050 67 00 00 41 00 01 c0 0c 00 41 00 01 00 00 01 ...g . A A 0060 00 69 00 01 00 00 01 00 03 02 68 32 00 05 00 47 ...i h2 . . G 0070 00 45 fe 0d 00 41 47 09 20 00 20 84 26 a5 7f f1 ...E . AG . . & . . 0080 01 7c 7d da 72 68 31 62 f4 28 67 82 55 4d 55 de ...l) rHbB . (g UNJ 0090 88 d4 81 12 a3 52 ac be 50 f4 07 00 04 00 01 00 R P 00a0 01 00 12 63 6c 6f 75 64 66 6c 61 72 65 2d 65 63 ...cloud flare-ec 00b0 68 2e 63 6f 6d 00 00 06 00 10 26 06 47 00 8d ...h.com & G . 00c0 90 74 7e 0a d9 00 01 cf c8 06 36 ...t~ F6 </pre>
--	--

Using wireshark and found this type result :

Here DNS Traffic and HTTPS traffic reports

And the second time we found the HTTP traffic report :
(Random)

Time	Source	Destination	Protocol	Length	Info
1 0.000000	192.168.31.231	192.168.31.255	UDP	84	55611 → 5775 Len=42
2 0.101240	192.168.31.137	224.0.0.22	IGMPv3	54	Membership Report / Join group 224.0.0.252 for any sources
3 0.101446	fe80::f1b7:49ca:9ff::ff02::16	ff02::16	ICMPv6	110	Multicast Listener Report Message v2
4 0.409102	fe80::9c4c:d6ff:fe1::ff02::16	ff02::16	ICMPv6	130	Multicast Listener Report Message v2
5 0.410606	fe80::7408:d6ff:fed::ff02::16	ff02::16	ICMPv6	130	Multicast Listener Report Message v2
6 0.412114	fe80::fce2:2aff:fed::ff02::16	ff02::16	ICMPv6	130	Multicast Listener Report Message v2
7 1.024479	fe80::2372:e848:bbb::ff02::16	ff02::16	ICMPv6	130	Multicast Listener Report Message v2
8 1.424482	192.168.31.137	239.255.255.250	SSDP	179	M-SEARCH * HTTP/1.1
9 2.048985	192.168.31.231	192.168.31.255	UDP	84	55611 → 5775 Len=42
10 3.888593	192.168.31.137	192.168.31.1	DNS	83	Standard query 0x47b A ctldl.windowsupdate.com
11 3.889139	192.168.31.137	192.168.31.1	DNS	83	Standard query 0xc264 AAAA ctldl.windowsupdate.com
12 3.910099	192.168.31.1	192.168.31.137	DNS	235	Standard query response 0x47b A ctldl.windowsupdate.com CNAME ctldl.windowsupdate.com.delivery.microsoft.com CNAME wu-b-net.trafficm
13 3.912056	192.168.31.1	192.168.31.137	DNS	247	Standard query response 0xc264 AAAA ctldl.windowsupdate.com CNAME ctldl.windowsupdate.com.delivery.microsoft.com CNAME wu-b-net.trafficm
14 3.914324	2409:4091:10:c209:2::2a04:4e42:9::684	192.168.31.137	TCP	86	54673 → 80 [SYN] Seq=0 Win=65535 Len=0 MSS=1440 WS=256 SACK_PERM
15 4.096498	192.168.31.231	192.168.31.255	UDP	84	55611 → 5775 Len=42
16 4.224536	192.168.31.137	146.75.122.172	TCP	66	54674 → 80 [SYN] Seq=0 Win=65535 Len=0 MSS=1460 WS=256 SACK_PERM
17 4.234392	2a04:4e42:9::684	2409:4091:10:c209:2::2a04:4e42:9::684	TCP	86	80 → 54673 [SYN, ACK] Seq=0 Ack=1 Win=65535 Len=0 MSS=1440 SACK_PERM WS=512
18 4.433052	192.168.31.137	239.255.255.250	SSDP	179	M-SEARCH * HTTP/1.1
19 4.444192	146.75.122.172	192.168.31.137	TCP	66	80 → 54674 [SYN, ACK] Seq=0 Ack=1 Win=65535 Len=0 MSS=1460 SACK_PERM WS=512

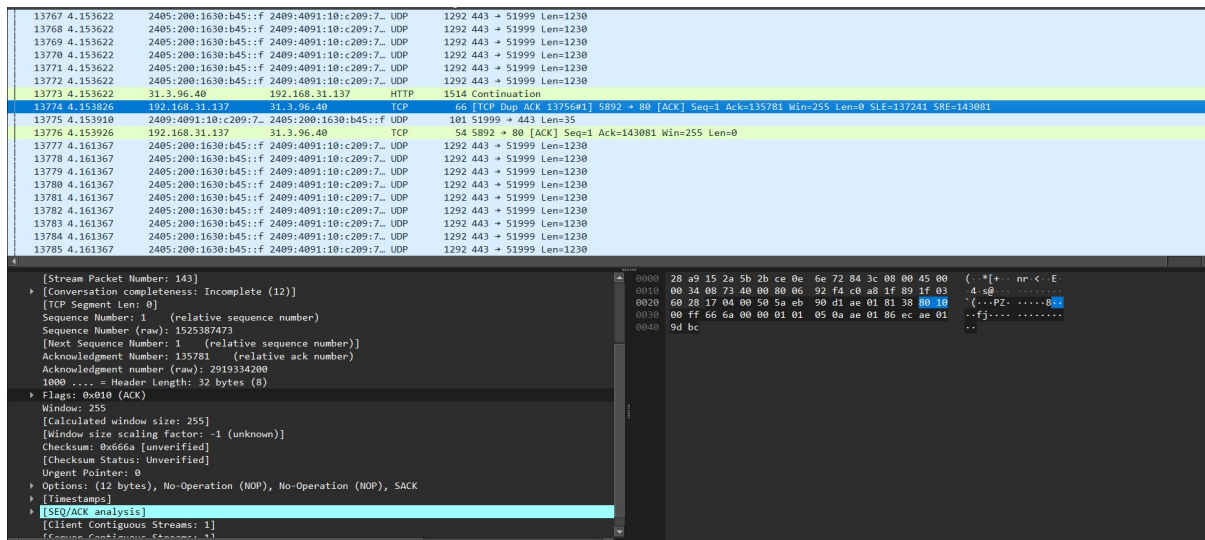
Frame 8: Packet, 179 bytes on wire (1432 bits), 179 bytes captured (1432 bits) on Ethernet II, Src: ce:8b:6e:72:84:3e (ce:8b:6e:72:84:3e), Dst: IP:mcast_7f:ff:fa (01:00:5e:7f:ff:fa)	0000 01 00 5e 7f ff fa ce 0e 6e 72 84 3e 08 00 45 00nr ce E
Internet Protocol Version 4, Src: 192.168.31.137, Dst: 239.255.255.250	0010 ff fa c2 03 07 06 00 91 73 12 4d 2d 53 45 41 52s M-SEAR
User Datagram Protocol, Src Port: 49667, Dst Port: 1900	0030 43 48 20 2a 20 48 54 54 50 2f 31 2e 31 0d 0a 48	CH * HTTP/1.1. H
Simple Service Discovery Protocol	0040 6f 73 74 3a 20 32 33 39 2e 32 35 35 2e 32 35 35	ost: 239.255.255
M-SEARCH * HTTP/1.1	0050 2e 32 35 30 3a 31 39 30 0d 0a 53 54 3a 20 75	.250:190 0 st: u
Host: 239.255.255.250:1900	0060 72 6e 3a 73 63 68 65 6d 61 73 2d 75 70 6e 70 2d	rn:schem as:upnp-
URI: urn:schemas-upnp-org:device:InternetGatewayDevice:1	0070 6f 72 67 3a 64 65 76 69 63 65 3a 49 6e 74 65 72	org:devi ce:Inter
Man: ssdp:discover	0080 6e 65 74 47 61 74 65 77 61 79 44 65 76 69 63 65	netGatew ayDevice
MX: 3	0090 3a 31 0d 0a 4d 61 6e 3a 20 22 73 64 70 3a 64	:1 Man: "ssdp:dis
	00a0 69 73 63 6f 76 65 72 2d 0d 0a 4d 58 3a 20 33 0d	icover" MX: 3
	00b0 0a 0d 0a

13755 4.152784	31.3.96.40	192.168.31.137	TCP	1514	[TCP Retransmission] 80 → 5892 [ACK] Seq=134321 Ack=1 Win=262 Len=1460
13756 4.152857	192.168.31.137	31.3.96.40	TCP	66	5892 → 80 [ACK] Seq=1 Ack=135781 Win=255 Len=0 SLE=137241 SRE=141621
13757 4.153028	2409:4091:10:c209:7::2a05:200:1630:b45::f	2405:200:1630:b45::f	UDP	102	51999 → 443 Len=36
13758 4.153622	2405:200:1630:b45::f	2409:4091:10:c209:7::2a05:200:1630:b45::f	UDP	1292	443 → 51999 Len=1230
13759 4.153622	2405:200:1630:b45::f	2409:4091:10:c209:7::2a05:200:1630:b45::f	UDP	1292	443 → 51999 Len=1230
13760 4.153622	31.3.96.40	192.168.31.137	HTTP	1514	Continuation
13761 4.153622	2405:200:1630:b45::f	2409:4091:10:c209:7::2a05:200:1630:b45::f	UDP	1292	443 → 51999 Len=1230
13762 4.153622	31.3.96.40	192.168.31.137	HTTP	1514	[TCP Out-Of-Order] Continuation
13763 4.153622	2405:200:1630:b45::f	2409:4091:10:c209:7::2a05:200:1630:b45::f	UDP	1292	443 → 51999 Len=1230
13764 4.153622	2405:200:1630:b45::f	2409:4091:10:c209:7::2a05:200:1630:b45::f	UDP	1292	443 → 51999 Len=1230
13765 4.153622	2405:200:1630:b45::f	2409:4091:10:c209:7::2a05:200:1630:b45::f	UDP	1292	443 → 51999 Len=1230
13766 4.153622	2405:200:1630:b45::f	2409:4091:10:c209:7::2a05:200:1630:b45::f	UDP	1292	443 → 51999 Len=1230
13767 4.153622	2405:200:1630:b45::f	2409:4091:10:c209:7::2a05:200:1630:b45::f	UDP	1292	443 → 51999 Len=1230
13768 4.153622	2405:200:1630:b45::f	2409:4091:10:c209:7::2a05:200:1630:b45::f	UDP	1292	443 → 51999 Len=1230
13769 4.153622	2405:200:1630:b45::f	2409:4091:10:c209:7::2a05:200:1630:b45::f	UDP	1292	443 → 51999 Len=1230
13770 4.153622	2405:200:1630:b45::f	2409:4091:10:c209:7::2a05:200:1630:b45::f	UDP	1292	443 → 51999 Len=1230
13771 4.153622	2405:200:1630:b45::f	2409:4091:10:c209:7::2a05:200:1630:b45::f	UDP	1292	443 → 51999 Len=1230
13772 4.153622	2405:200:1630:b45::f	2409:4091:10:c209:7::2a05:200:1630:b45::f	UDP	1292	443 → 51999 Len=1230
13773 4.153622	31.3.96.40	192.168.31.137	HTTP	1514	Continuation

[Stream Packet Number: 141]	0020 1f 89 00 50 17 04 ae 01 81 38 5a eb 90 d1 50 10	...P...0Z...P
[Conversation completeness: Incomplete (12)]	0030 01 06 ba 98 00 00 32 0f d3 db 15 03 9c 1c ee 072.....
[TCP Segment Len: 1460]	0040 71 e5 ba f5 ec 0d 22 30 d0 92 49 c7 53 9e bf 43	q... "0...S...C
Sequence Number (raw): 2915334200	0050 f9 50 04 d9 c9 3c 0d ca 38 c7 6f ad 3b 79 54 5e	/P...<...8...yT
[Next Sequence Number: 137241 (relative sequence number)]	0060 a5 07 71 d4 f7 ab df 20 d4 9c 8c 7e 1f a7 f9 eb	q... ..s... ..
Acknowledgment Number: 1 (relative ack number)	0070 51 b4 99 1b 09 29 eb bb 9c d0 04 fb 82 e5 bf 84	Q... ..).....
Acknowledgment number (raw): 1525387473	0080 71 9e 7d 0f e3 fe 7d a9 15 98 2e 15 95 b1 c0 e7
0101 = Header Length: 20 bytes (5)	0090 ff 00 af 9f ff 00 55 43 24 81 d7 0c 7a 9e bd 87UC \$...z...
Flags: 0x010 (ACK)	00a0 5f f3 d7 b5 22 c9 83 95 3b 73 d8 74 ce 39 fe 5fs...t...9...
[calculated window size: 262]	00b0 e7 34 01 23 39 c0 2d f2 e7 a0 ee 4a 7f c6 91 a5	...-#9... ..N... ..
[Window size scaling factor: -1 (unknown)]	00c0 20 2e d6 1d 3e 60 38 f4 a8 8c 98 20 2e 19 87 0a	...-5-8... ..
Checksum: 0xbe98 [unverified]	00d0 7b ff 00 9f ff 00 55 2b b0 89 c0 e1 70 3b 1a 94U... ..sps...
[Checksum Status: Unverified]	00e0 01 2f 99 20 e0 9c e0 73 c1 3f e7 7c fe 74 0d	/... ..s...P...
Urgent Pointer: 0	00f0 db f9 18 e3 8d bd 8f f9 ed 51 80 54 a0 c0 f5 04Q... ..T... ..
[Timestamps]	0100 7f 3e 3d f1 4c 79 0a a9 3c 02 c7 18 00 fc dd 68	>= Ly... ..<.....h
[SEQ/ACK analysis]	0110 02 7d ec cc 72 fd 3a ae ff 00 f3 d2 9a 08 d8 ce<..... ..
[Client Contiguous Streams: 1]	0120 ee 49 e9 03 1f 95 48 ef b1 46 31 f4 c7 f1 7f 91<..... ..
[Server Contiguous Streams: 1]	0130 48 64 da b8 0d 81 db 9e 83 e9 f5 a0 09 51 db 1d	Hd..... ..<..... ..
TCP segment flags: ACK	0140 3e 52 38 40 3a 8a 6b 49 86 2c 0f 3c 30 0d 81 fe	>R8g: kT... ..<0... ..
	0150 78 a4 5c 96 6d c7 68 cf 14 bb d4 b0 18 dd 82 31	x \... ..h... ..<1
	0160 b4 63 ef 3d 68 01 c0 90 e0 82 39 c9 e7 d6 94 aa	c... ..h... ..<9... ..
	0170 68 2a 74 60 b7 e7 f9 ef c1 09 79 00 8d a7 9c 2f	*2... ..<... ..Q... ..y... ..

Here we find two types of HTTP report randomly.

Here We found the TCP traffic report:



Here In Wireshark, ACK stands for acknowledgment and signifies that a TCP packet was received successfully. When the ACK flag is set, it tells the receiving host to process the acknowledgment number in the packet, which confirms the receipt of data from the other side of the connection. This flag is fundamental to TCP's reliability, ensuring that data is delivered correctly and coordinating the flow of information.

- **Confirmation of receipt:**

An ACK packet is sent in response to a data packet to confirm that it was received by the destination.

- **Coordination of data flow:**

It is a key part of the TCP handshake, which establishes and maintains a reliable connection by coordinating the flow of data between two hosts.

- **Used with other flags:**

You will often see ACK used with other flags, such as SYN, ACK (used to establish a connection) or FIN, ACK (used to gracefully close a connection).

- **Reliability:**

Without the ACK flag, there would be no reliable way for a host to confirm that the data it sent was successfully received by the other party.

Here is the result of wireshark packet traffic like DNS, HTTP, TCP packet Traffic sniffing.

Thankyou.