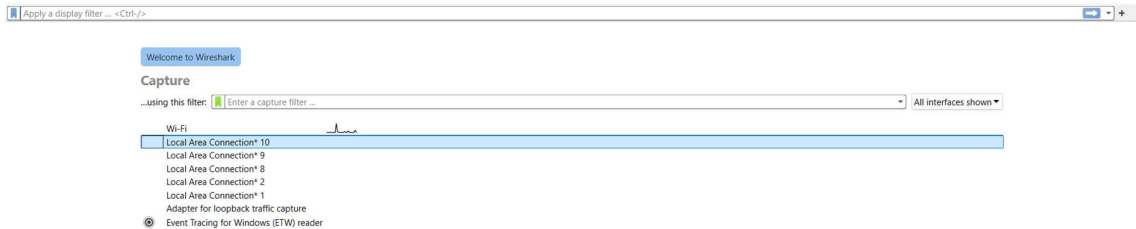


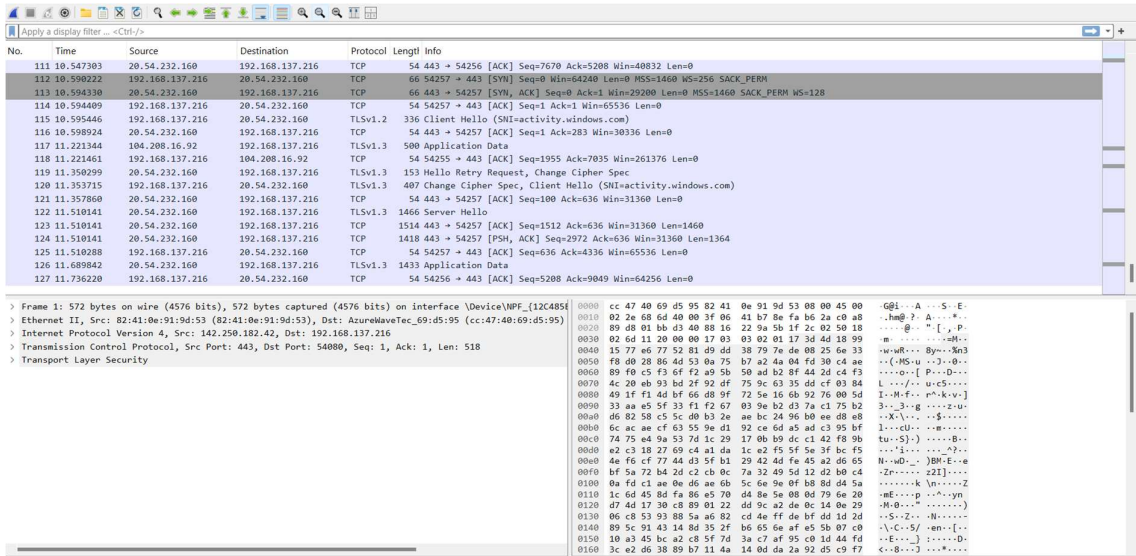
Practical 5

Aim:

Experiments on Packet capture tool: Wireshark



Capturing Packets:

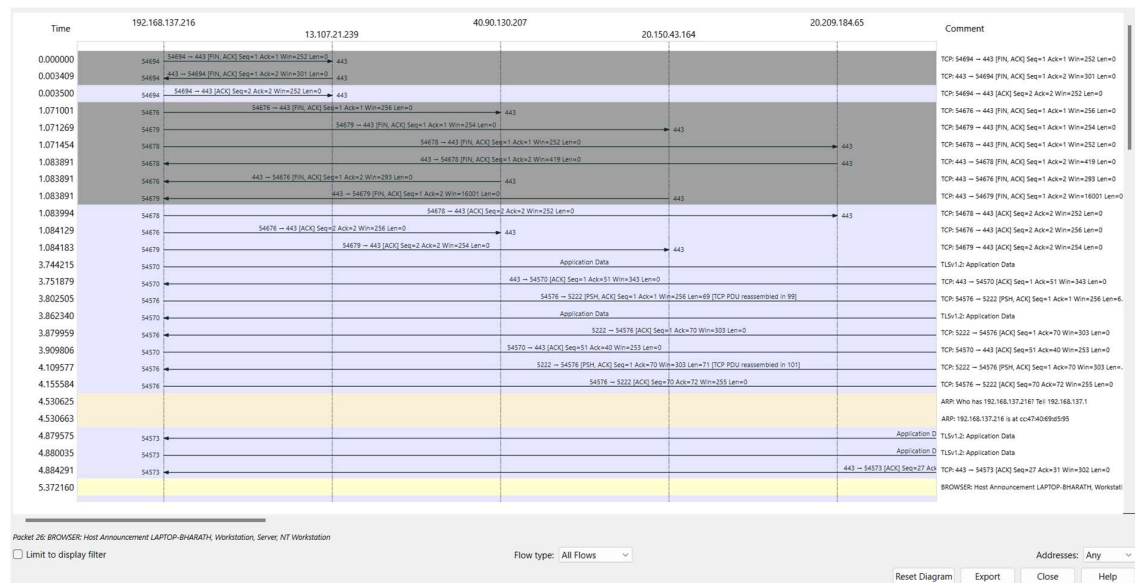


Name	Filter
Bad TCP	tcp.analysis.flags && /tcp.analysis.window_update && /tcp.analysis.keep_alive && /tcp.analysis.keep_alive_ack
HSRP State Change	hsrp.state != 8 && hsrp.state != 16
Spanning Tree Topology Change	stp.type == 0x00
OSPF State Change	ospf.msg != 1
ICMP errors	icmp.type in {3,5,11} icmp.v6.type in {1,4}
ARP	arp
ICMP	icmp icmpv6
TCP RST	tcp.flags.reset eq 1
SCPT ABORT	scpt.chunk_type eq ABORT
IPv4 TTL low or unexpected	((ip.dst != 224.0.0.0/4 && (ip.ttl < 5 && !ipim) ospf eigrp bgp tcp.port == 179) (ip.dst == 224.0.0.0/24 && ip.dst != 224.0.0.251 && ip.ttl != 1 && !vrrp carp eigrp rip glibp))
IPv6 hop limit low or unexpected	(ip6.dst != ff02::8 && ip6.hlim < 5 && ! ospf bgp tcp.port == 179) (ip6.dst == ff02::8 && ip6.hlim not in {1, 64, 255})
Checksum Errors	eth[1].checksum != 0x00 !checksum.status == "Bad" tcp.checksum.status == "Bad" udp.checksum.status == "Bad" sctp.checksum.status == "Bad" ospf.checksum.status == "Bad" carp.checksum.status == "Bad" eigrp.checksum.status == "Bad" vrrp vrrp6 rip rip6 glibp glibp6
SNMP	snmp nms nms netbox
HTTP	http tcp.port == 80 http2
DCERPC	dcerpc
Routing	hsrp eigrp ospf bgp cdp vrrp carp gwp igmp ismp
TCP SYN/FIN	tcp.flags & 0x02 tcp.flags.fin == 1
TCP	tcp
UDP	udp
netconfig	en[10] & 1
System Event	systemd_journal sysdig

Filter Name	Filter Expression
Ethernet address 00:00:5e:00:53:00	eth.addr == 00:00:5e:00:53:00
Ethernet type 0x0806 (ARP)	eth.type == 0x0806
Ethernet broadcast	eth.addr == #####
No ARP	not arp
IPv4 only	ip
IPv4 address 192.0.2.1	ip.addr == 192.0.2.1
IPv4 address isn't 192.0.2.1	ip.addr != 192.0.2.1
IPv6 only	ipv6
IPv6 address 2001:db8::1	ipv6.addr == 2001:db8::1
TCP only	tcp
UDP only	udp
Non-DNS port	!(udp.port == 53 tcp.port == 53)
TCP or UDP port is 80 (HTTP)	tcp.port == 80 udp.port == 80
HTTP	http
No ARP and no DNS	not arp and not dns
Non-HTTP and non-SMTP to/from 192.0.2.1	ip.addr == 192.0.2.1 and tcp.port not in (80, 25)

Search TCP packets in search bar.

Save the packets.



2. Create a Filter to display only ARP packets and inspect the packets.

Procedure

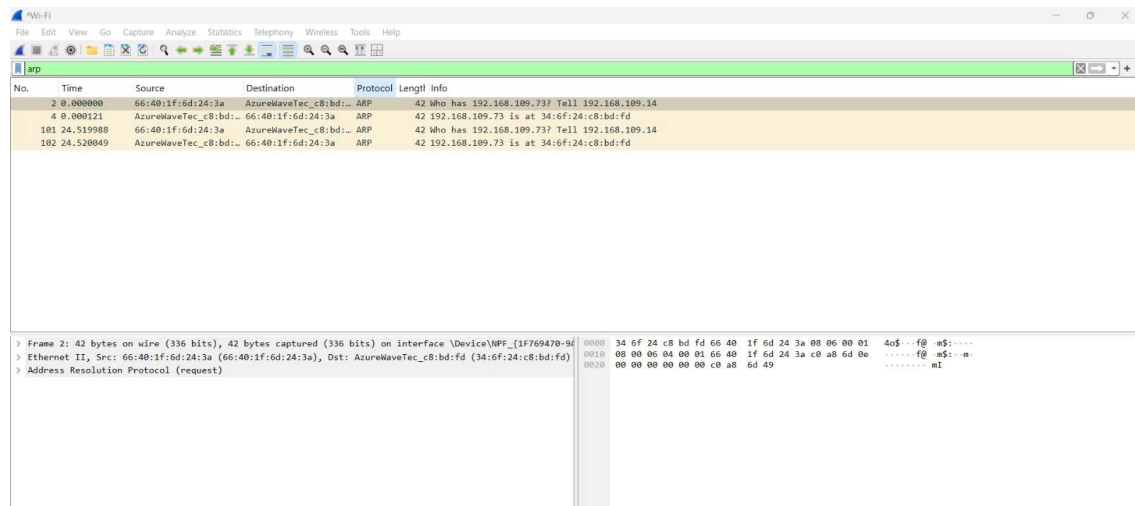
Go to capture → option

Select stop capture automatically after 100 packets.

Then click Start capture.

Search ARP packets in search bar.

Save the packets.



3. Create a Filter to display only DNS packets and provide the flow graph.

Procedure

Go to capture → option

Select stop capture automatically after 100 packets.

Then click Start capture.

Search DNS packets in search bar.

To see flow graph click Statistics → Flow graph.

Save the packets.

dns

No.	Time	Source	Destination	Protocol	Length	Info
94	6.680975	192.168.137.216	192.168.137.1	DNS	74	Standard query 0xc5ee A assets.msn.com
95	6.715291	192.168.137.216	192.168.137.1	DNS	74	Standard query 0xc5ee A assets.msn.com
96	6.746597	192.168.137.1	192.168.137.216	DNS	294	Standard query response 0xc5ee A assets.msn.com CNAME assets.msn.com.edgekey.net CNAME e28578.d.akamaiedge.net A 23.211.105.219 A 23...
97	6.746597	192.168.137.1	192.168.137.216	DNS	294	Standard query response 0xc5ee A assets.msn.com CNAME assets.msn.com.edgekey.net CNAME e28578.d.akamaiedge.net A 23.211.105.219 A 23...

> Frame 94: 74 bytes on wire (592 bits), 74 bytes captured (592 bits) on interface \Device\NPF_{12C485EE-...}

> Ethernet II, Src: AzureWaveTec, 69:d5:95 (cc:47:40:69:d5:95), Dst: 82:41:0e:91:9d:53 (82:41:0e:91:9d:53)

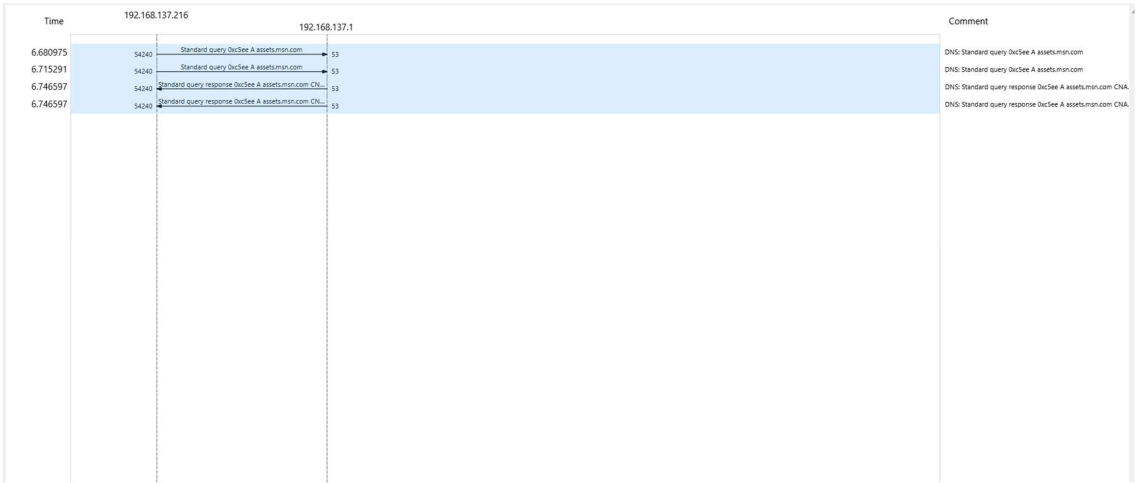
> Internet Protocol Version 4, Src: 192.168.137.216, Dst: 192.168.137.1

> User Datagram Protocol, Src Port: 54240, Dst Port: 53

> Domain Name System (query)

0000 82 41 0e 91 9d 53 cc 47 40 69 d5 95 00 00 45 00 A..S-G@!...E.
0010 00 3c a0 56 00 00 00 11 06 30 c9 a8 89 d8 c9 a8 <V...@.....
0020 89 01 d3 e0 00 35 00 28 d2 38 c5 ee 01 00 00 01S(.8.....
0030 00 00 00 00 00 00 06 61 73 73 65 74 73 03 6d 73a ssets.ms
0040 6e 03 63 6f 6d 00 00 01 00 01 n.com...<

Flowgraph:



Save the packets

The image shows a Wireshark packet capture of a DHCP transaction. The packet list on the left shows two packets: a DHCP Request (573) and a DHCP ACK (574). The packet details for the selected DHCP ACK (574) are shown on the right. The details pane shows the following structure:

- Ethernet II, Src: AzureWaveTec_c8:bd:fd (34:6f:24:c8:bd:fd), Dst: 66:40:1f:6d:24:3a (66:40:1f:6d:24:3a)
- Internet Protocol Version 4, Src: 192.168.109.73, Dst: 192.168.109.14
- User Datagram Protocol, Src Port: 68, Dst Port: 67
- Dynamic Host Configuration Protocol (Request)

The packet bytes pane on the right shows the raw data of the packet, with a hex dump and ASCII representation. The hex dump shows the following data:

```

0000  66 40 1f 6d 24 3a 34 6f 24 c8 bd fd 08 00 45 00  f6 m5:40 $---- E-
0010  01 4e 55 70 00 00 00 11 08 06 c0 a8 6d 49 c0 a8  Hdp ---- mI-
0020  6d 0e 00 44 00 43 01 3a 01 cc 01 01 06 00 53 b1  m D C :---- S-
0030  54 47 00 00 00 c0 a8 6d 49 00 00 00 00 00 00  T----- m-----
0040  00 00 00 00 00 14 6f 24 c8 bd fd 00 00 00 00  ---- 4e $-----
0050  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  <-----
0060  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  <-----
0070  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  <-----
0080  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  <-----
0090  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  <-----
00a0  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  <-----
00b0  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  <-----
00c0  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  <-----
00d0  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  <-----
00e0  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  <-----
00f0  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  <-----

```