

---

***COMP 4108 – COMPUTER SYSTEMS SECURITY***

*Experience 3 – WireShark - Network Auditing For Anomaly  
Detection and Network Security*

---



Student Name: Ben Cendana  
Student #: 100811212  
Date: April 10 2018

## Table of Contents

1.0: Introduction .....	3
2.0: Performing A Capture .....	3
3.0: Capture Filtering .....	5
3.1: Filtering By IP .....	5
3.2: Filtering By Protocol and Port Number.....	6
3.3: Combined Filtering.....	7
4.0: Wireshark Statistics For Anomaly Detection .....	8
5.0: Wireshark As A Network Security Mechanism .....	10

## 1.0: Introduction

An underappreciated yet effective method in preventing the spread of viruses through the network is anomaly detection. But in order to implement effective anomaly detection we must be able to perform some kind of network audit that collects and intercepts all incoming network traffic. Once this data is collected we can then determine what the network's behaviour is.

The purpose of this experience is to become better familiar with a network packet analyser called Wireshark and understand how to use it to its full potential, once we fully understand how to use the software we can apply it to intrusion detection.

By the end of this experience we should have mastered all of the main Wireshark features.

## 2.0: Performing A Capture

The very first step is to setup the capture interface for the environment (Figure 1), since the primary internet connection is wifi we will use Wi-Fi as the interface type and make sure *Use promiscuous mode* is selected to allow all packets to be captured.

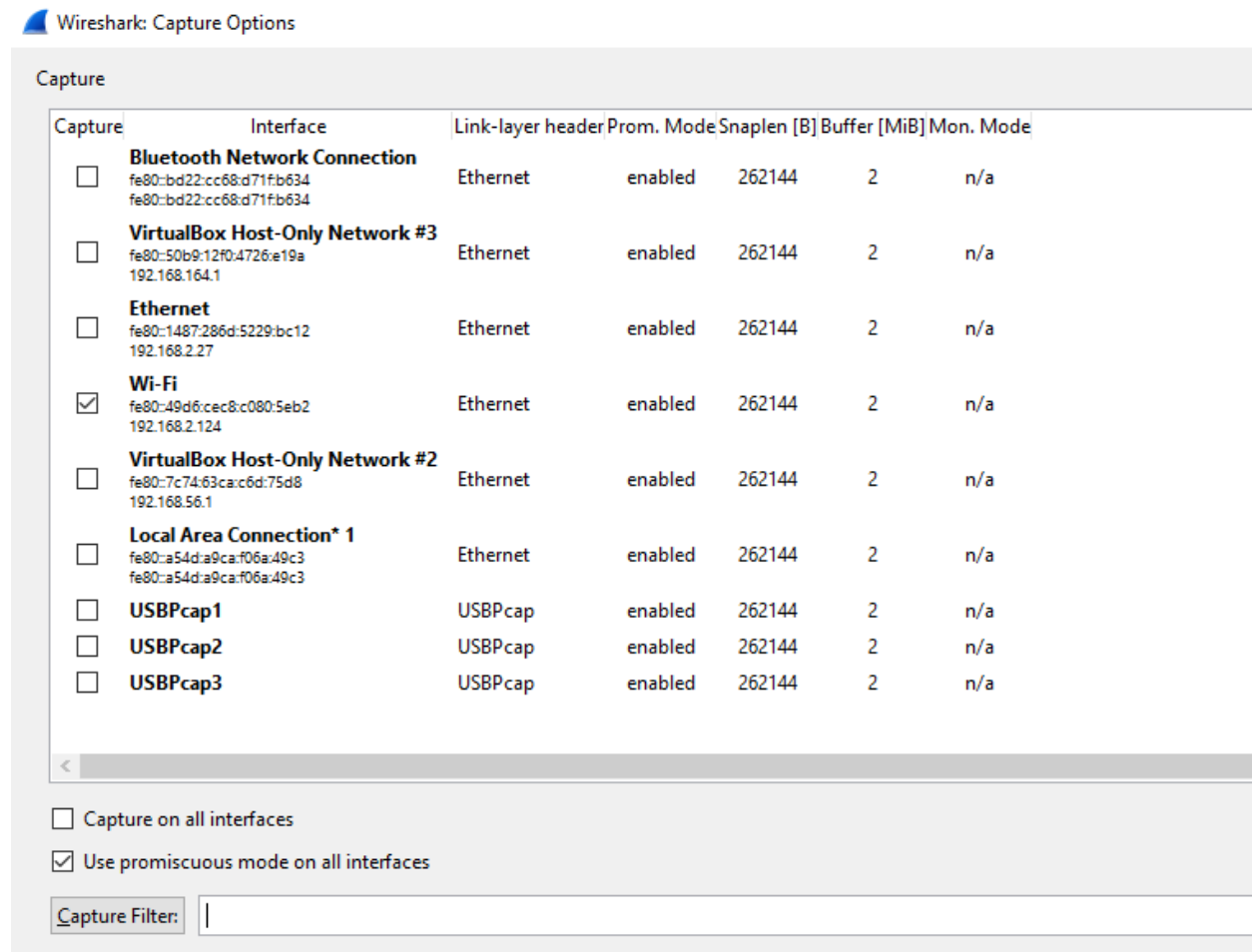


Figure 1: Setting Up The Capture Interface

Wi-Fi [Wireshark 2.4.0 (v2.4.0-0-g9be0fa500d)]

File Edit View Go Capture Analyze Statistics Telephony Tools Internals Help

Filter: Expression... Clear Apply Save

No.	Time	Source	Destination	Protocol	Length	Info
48	16.004815	131.253.33.254	192.168.2.124	TCP	60	
49	18.249424	192.168.2.124	134.117.10.200	TCP	66	
50	18.278644	134.117.10.200	192.168.2.124	TCP	66	
51	18.278731	192.168.2.124	134.117.10.200	TCP	54	
52	18.279321	192.168.2.124	134.117.10.200	TCP	1514	
53	18.279330	192.168.2.124	134.117.10.200	HTTP	656	GET /_ajax_htmlview?action=77&__className=bookhtr
54	18.306931	134.117.10.200	192.168.2.124	TCP	60	
55	18.336484	134.117.10.200	192.168.2.124	TCP	60	
56	18.767618	SenaoNet_58:e3:8b	Broadcast	EAPOL	70	
57	18.837707	134.117.10.200	192.168.2.124	TCP	1132	
58	18.881425	192.168.2.124	134.117.10.200	TCP	54	
59	18.928255	134.117.10.200	192.168.2.124	TCP	1514	
60	18.929313	134.117.10.200	192.168.2.124	TCP	639	
61	18.929314	134.117.10.200	192.168.2.124	TCP	64	
62	18.929314	134.117.10.200	192.168.2.124	HTTP	60	
63	18.929557	192.168.2.124	134.117.10.200	TCP	54	
64	18.929732	192.168.2.124	134.117.10.200	TCP	54	
65	18.974034	134.117.10.200	192.168.2.124	TCP	60	
66	20.918028	SenaoNet_12:11:b1	Broadcast	EAPOL	70	
67	21.634977	192.168.2.82	239.255.255.250	SSDP	167	
68	21.737364	192.168.2.82	239.255.255.250	SSDP	167	
69	21.839775	192.168.2.82	239.255.255.250	SSDP	167	
70	23.170991	192.168.2.71	224.0.0.251	MDNS	159	query 0x0000 PTR _homekit._tcp.local, "QU" question PTR _airpla
71	23.171629	fe80::14a9:1d0d:ff02::fb		MDNS	179	query 0x0000 PTR _homekit._tcp.local, "QU" question PTR _airpla
72	24.093666	192.168.2.71	224.0.0.251	MDNS	159	query 0x0000 PTR _homekit._tcp.local, "QM" question PTR _airpla
73	24.093824	fe80::14a9:1d0d:ff02::fb		MDNS	179	query 0x0000 PTR _homekit._tcp.local, "QM" question PTR _airpla
74	24.399761	SenaoNet_58:e3:7f	Broadcast	EAPOL	70	

Frame 1: 42 bytes on wire (336 bits), 42 bytes captured (336 bits) on interface 0

Ethernet II, Src: IntelCor\_a0:8e:53 (68:17:29:a0:8e:53), Dst: Fortinet\_45:07:3e (08:5b:0e:45:07:3e)

Address Resolution Protocol (request)

Figure 2: A successful Capture

As we can see in Figure 2 we now have a successful capture of all the incoming packets into the network.

Furthermore the capture provides us with some packet information: these are time of the packet, source IP, destination IP, protocol, port, length and info.

We can get some hint of what the packet is doing based on the protocol information, If the protocol for the packet says TCP we know that the packet is just setting up the network communication and if the packet is entering through port 8080 we know it must be a server application.

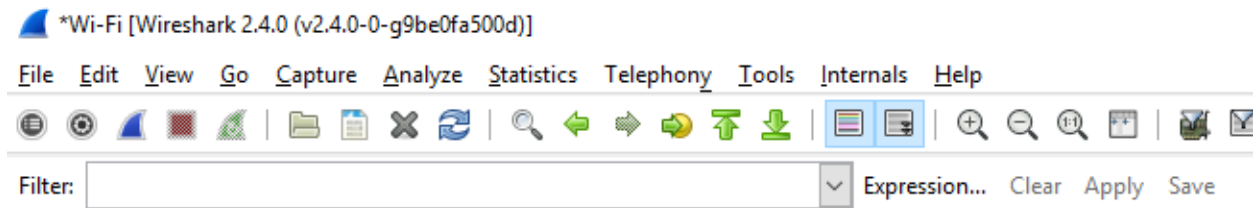
Putting together what we now know based on the port being used and what protocol is being used we know that it's a server application trying to establish communication.

## 3.0: Capture Filtering

As we can see in Figure 2 we are left with many packets, most of which we don't need. To focus on the packets we do need we will use filters to filter out the packets we only want.

### 3.1: Filtering By IP

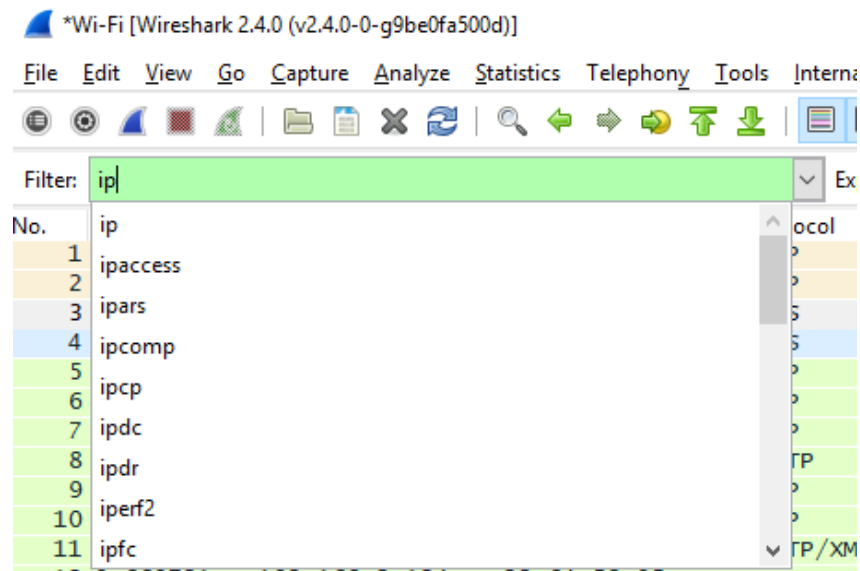
Just below the main menu we can see there exists a text field called Filter(Figure 3), in this text field we will type in the filter commands we want.



**Figure 3:** Wireshark Filter

Since we are interested in filtering the packets based on a particular source and destination IP, we will apply a filter that filters out the packets based on this criteria.

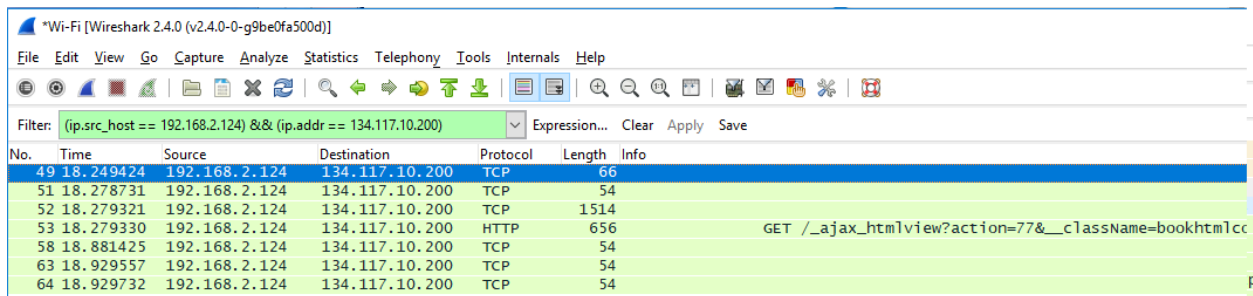
We start by typing *ip*, if we don't exactly remember the syntax Wireshark features an intellisense that will guide us (Figure 4) by providing a list of available filters.



**Figure 4:** The Wireshark Filter Intellisense

Furthermore If the syntax is valid it will remain green, otherwise it will changed to red to indicate a syntax error.

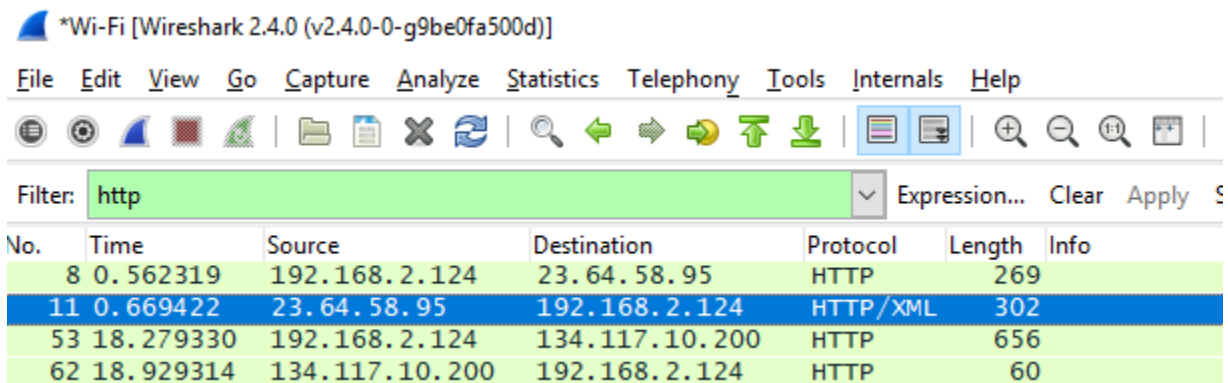
We type in the command *(ip.src.host == 192.168.2.124) && (ip.addr == 134.117.10.200)* (Figure 5) to filter packets coming from the source IP 192.168.2.124 to the destination IP 134.117.10.200.



**Figure 5:** The Wireshark Filter Intellisense

As we can see we are now left with only the packets from the source IP address 192.168.2.124 and destination IP 134.117.10.200 (Figure 5).

### 3.2: Filtering By Protocol and Port Number



**Figure 6:** Filtering By Protocol

We can also filter by the protocol type by typing in the protocol type (Figure 6) and by the port number (Figure 7).

\*Wi-Fi [Wireshark 2.4.0 (v2.4.0-0-g9be0fa500d)]

File Edit View Go Capture Analyze Statistics Telephony Tools Internals Help

Filter: tcp.port == 443 Expression... Clear A

No.	Time	Source	Destination	Protocol	Length	Info
18	2.555063	172.217.1.2	192.168.2.124	TLSv1.2	117	
33	8.338232	204.79.197.200	192.168.2.124	TCP	60	
37	10.234769	204.79.197.200	192.168.2.124	TCP	60	
48	16.004815	131.253.33.254	192.168.2.124	TCP	60	

**Figure 7:** Filtering By Port Number

### 3.3: Combined Filtering

\*Wi-Fi [Wireshark 2.4.0 (v2.4.0-0-g9be0fa500d)]

File Edit View Go Capture Analyze Statistics Telephony Tools Internals Help

Filter: (ip.addr == 192.168.2.124) && (tcp.port == 443) Expression... Clear App

No.	Time	Source	Destination	Protocol	Length	Info
18	2.555063	172.217.1.2	192.168.2.124	TLSv1.2	117	
33	8.338232	204.79.197.200	192.168.2.124	TCP	60	
37	10.234769	204.79.197.200	192.168.2.124	TCP	60	
48	16.004815	131.253.33.254	192.168.2.124	TCP	60	

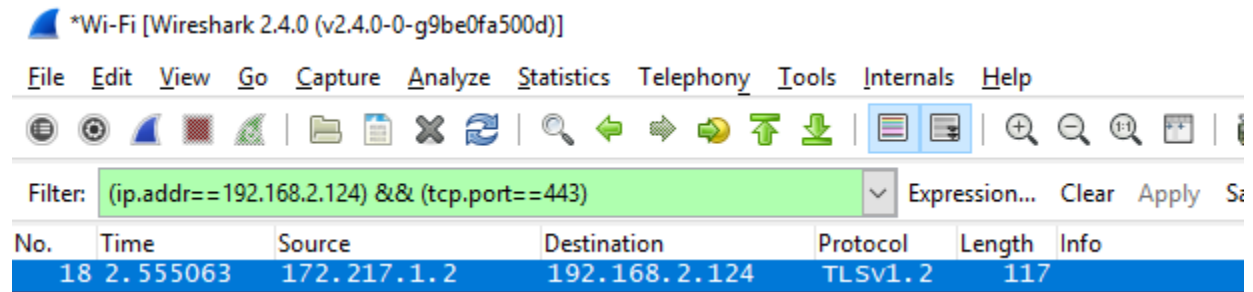
**Figure 8:** Filtering By Port Number And IP Address

We can also do a combined filter where we can search for all IP address using a particular port (Figure 8)

## 4.0: Wireshark Statistics For Anomaly Detection

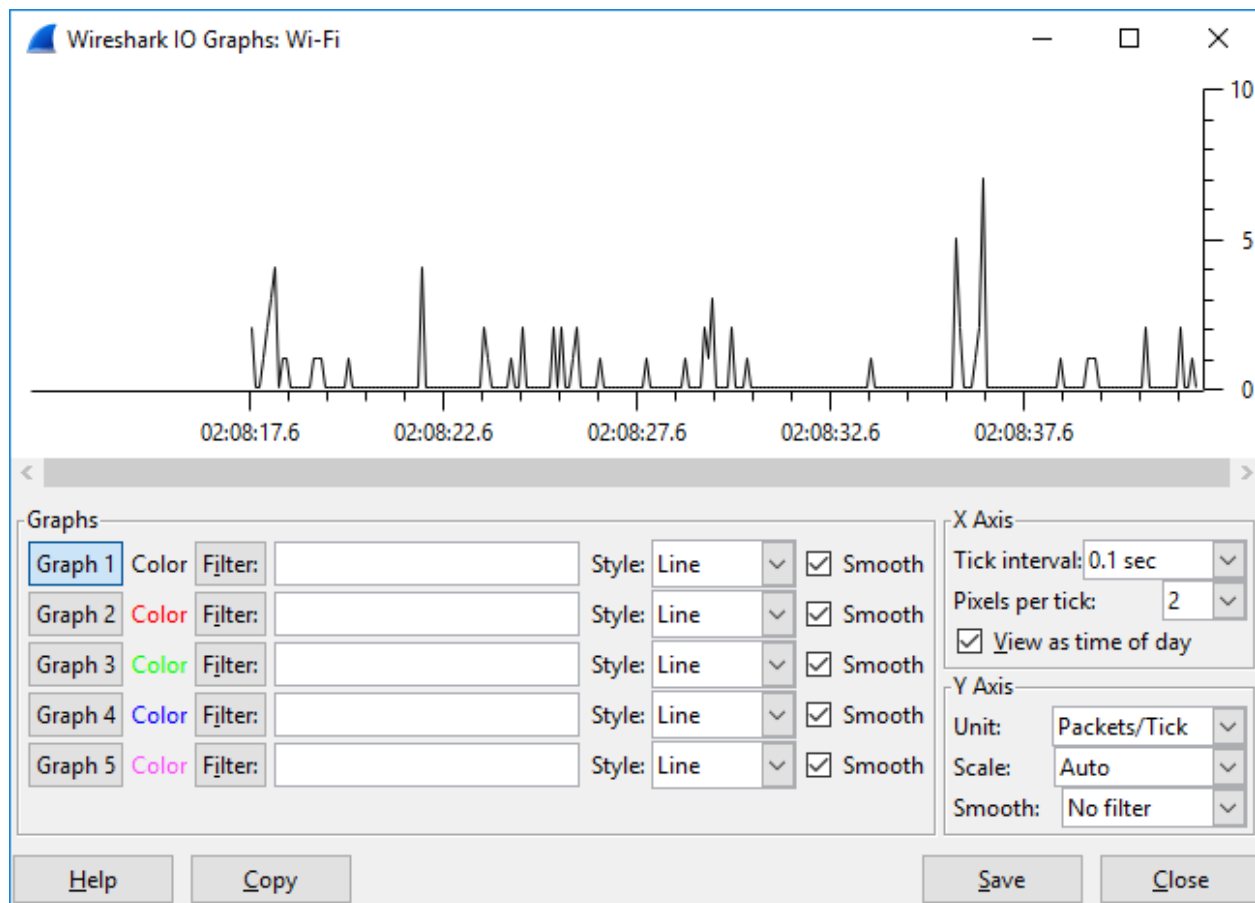
Anomaly detection works by having some form of statistic to determine the network's behaviour. By clicking on the packet and selecting *statistics* Wireshark provides a statistical audit for that packet.

We can then use this information to profile the packet's behaviour and look for anomalies on an ongoing basis.



**Figure 9:** The Packet With A Statistic Applied To





**Figure 10:** Displaying The Statistic Information For The Particular Packet

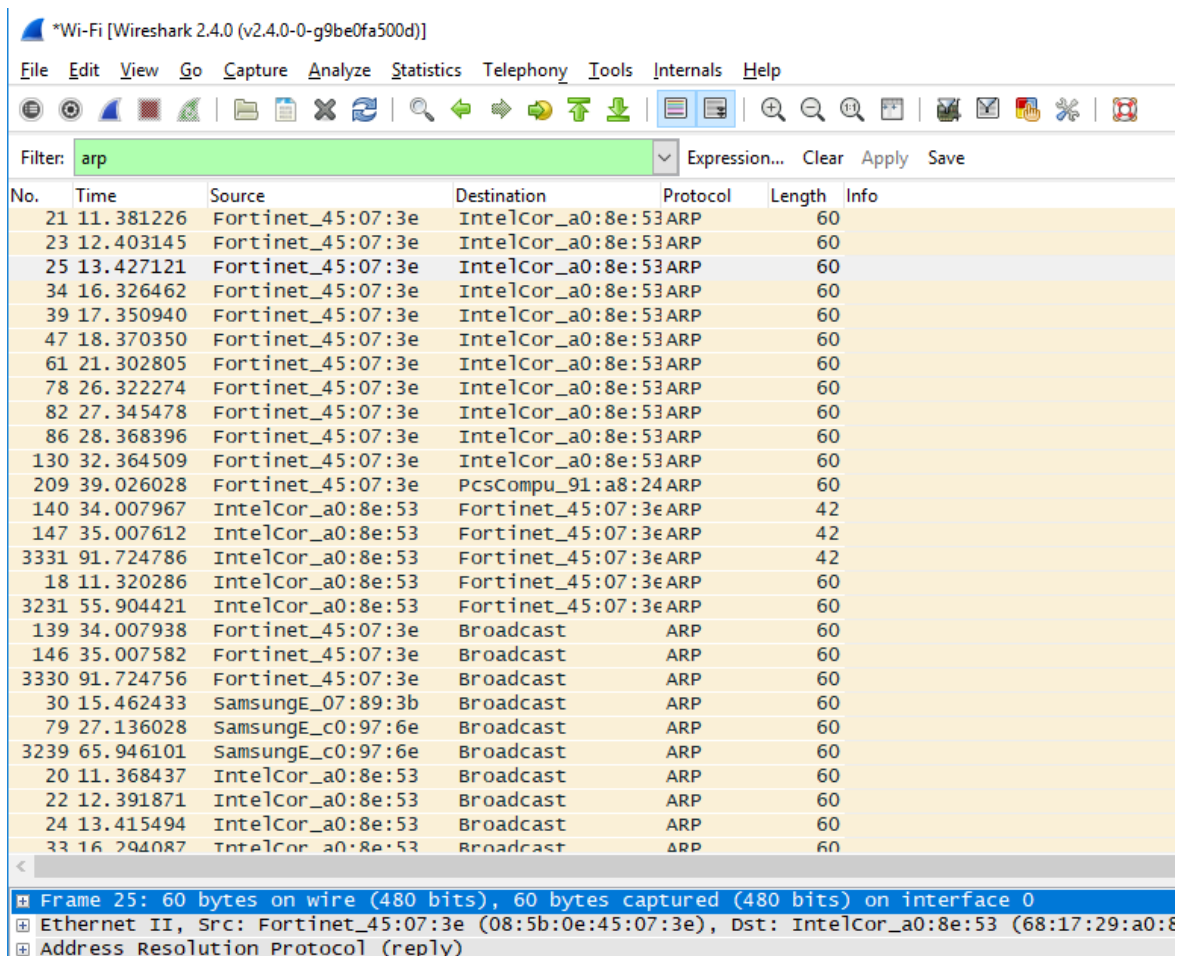
As can be seen in Figure 11 the x axis displays the time indicated by 0.1 sec and the number of packets, based on this statistic we can spot a pattern where a packet burst occurs every 0.5 seconds. If we observe a change in this behaviour then this that might be an indication of an intrusion.

(Next Page)

## 5.0: Wireshark As A Network Security Mechanism

Since we now understand how to capture packets and analyze them we can now understand how to use Wireshark for viruses and malware detection.

Lets assume that we monitor our network with Wireshark everyday and we suddenly find a new protocol called *arp* included with a packet, we deem this with some suspicion and decide to investigate.



\*Wi-Fi [Wireshark 2.4.0 (v2.4.0-0-g9be0fa500d)]

File Edit View Go Capture Analyze Statistics Telephony Tools Internals Help

Filter: **arp** Expression... Clear Apply Save

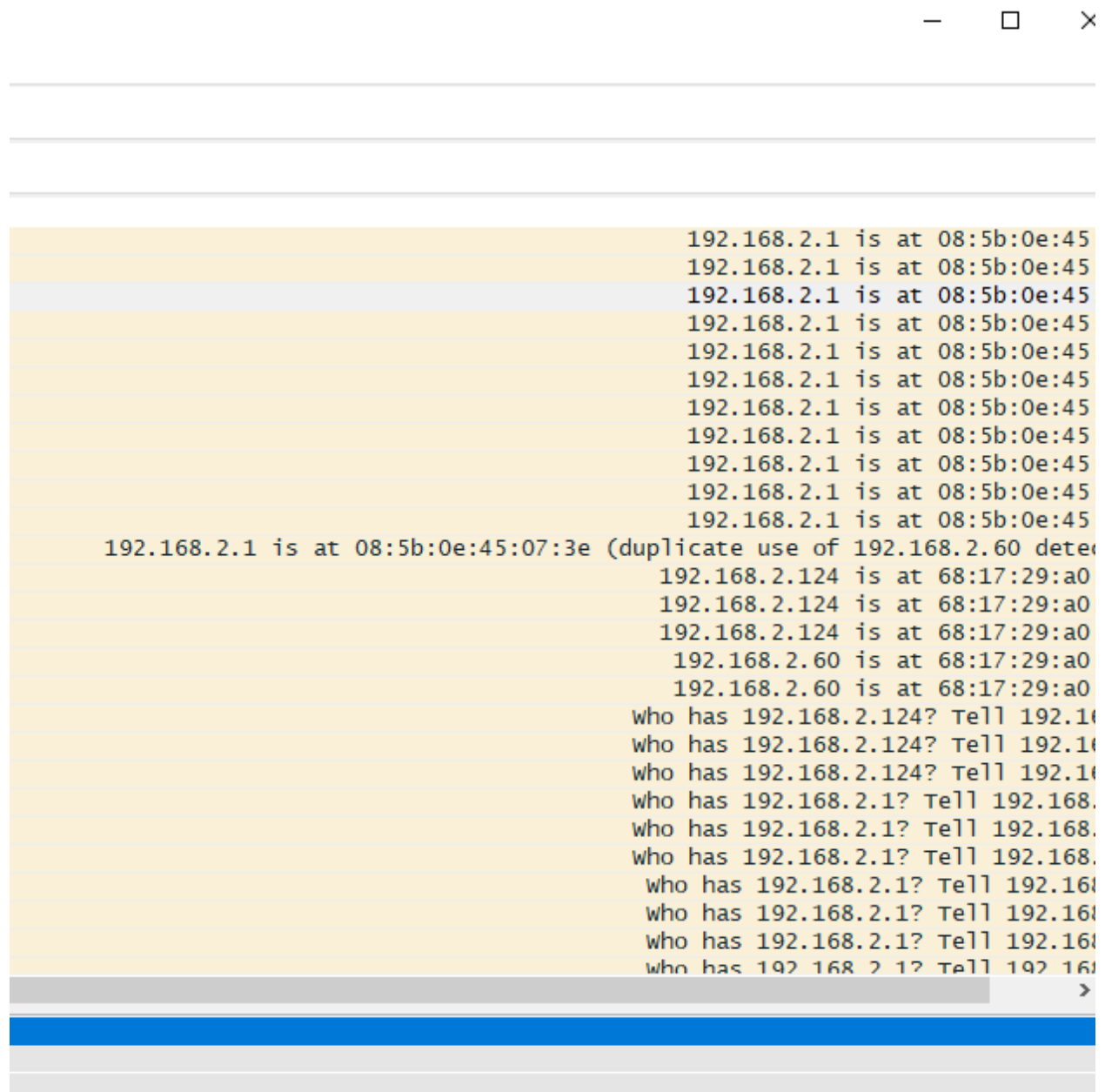
No.	Time	Source	Destination	Protocol	Length	Info
21	11.381226	Fortinet_45:07:3e	IntelCor_a0:8e:53	ARP	60	
23	12.403145	Fortinet_45:07:3e	IntelCor_a0:8e:53	ARP	60	
25	13.427121	Fortinet_45:07:3e	IntelCor_a0:8e:53	ARP	60	
34	16.326462	Fortinet_45:07:3e	IntelCor_a0:8e:53	ARP	60	
39	17.350940	Fortinet_45:07:3e	IntelCor_a0:8e:53	ARP	60	
47	18.370350	Fortinet_45:07:3e	IntelCor_a0:8e:53	ARP	60	
61	21.302805	Fortinet_45:07:3e	IntelCor_a0:8e:53	ARP	60	
78	26.322274	Fortinet_45:07:3e	IntelCor_a0:8e:53	ARP	60	
82	27.345478	Fortinet_45:07:3e	IntelCor_a0:8e:53	ARP	60	
86	28.368396	Fortinet_45:07:3e	IntelCor_a0:8e:53	ARP	60	
130	32.364509	Fortinet_45:07:3e	IntelCor_a0:8e:53	ARP	60	
209	39.026028	Fortinet_45:07:3e	PcsCompu_91:a8:24	ARP	60	
140	34.007967	IntelCor_a0:8e:53	Fortinet_45:07:3e	ARP	42	
147	35.007612	IntelCor_a0:8e:53	Fortinet_45:07:3e	ARP	42	
3331	91.724786	IntelCor_a0:8e:53	Fortinet_45:07:3e	ARP	42	
18	11.320286	IntelCor_a0:8e:53	Fortinet_45:07:3e	ARP	60	
3231	55.904421	IntelCor_a0:8e:53	Fortinet_45:07:3e	ARP	60	
139	34.007938	Fortinet_45:07:3e	Broadcast	ARP	60	
146	35.007582	Fortinet_45:07:3e	Broadcast	ARP	60	
3330	91.724756	Fortinet_45:07:3e	Broadcast	ARP	60	
30	15.462433	SamsungE_07:89:3b	Broadcast	ARP	60	
79	27.136028	SamsungE_c0:97:6e	Broadcast	ARP	60	
3239	65.946101	SamsungE_c0:97:6e	Broadcast	ARP	60	
20	11.368437	IntelCor_a0:8e:53	Broadcast	ARP	60	
22	12.391871	IntelCor_a0:8e:53	Broadcast	ARP	60	
24	13.415494	IntelCor_a0:8e:53	Broadcast	ARP	60	
23	16.294087	IntelCor_a0:8e:53	Broadcast	ARP	60	

Frame 25: 60 bytes on wire (480 bits), 60 bytes captured (480 bits) on interface 0

Ethernet II, Src: Fortinet\_45:07:3e (08:5b:0e:45:07:3e), Dst: IntelCor\_a0:8e:53 (68:17:29:a0:8e:53)

Address Resolution Protocol (reply)

Figure 11: ARP Protocol Captured



**Figure 12:** ARP Protocol Captured Information

After looking at the packet information we find a whole list of IPs being scanned and the source of the IP scan was originating from a source outside of our network (Figure 11).

We then realize the ARP protocol was being used by someone outside of our network to sniff the network to determine all the computer connected to the network .

As this example as demonstrated Wireshark was effectively used as a security tool to observe unusually network patterns and avoid a potential attack.

