

Everlyn Leon

## Abstract

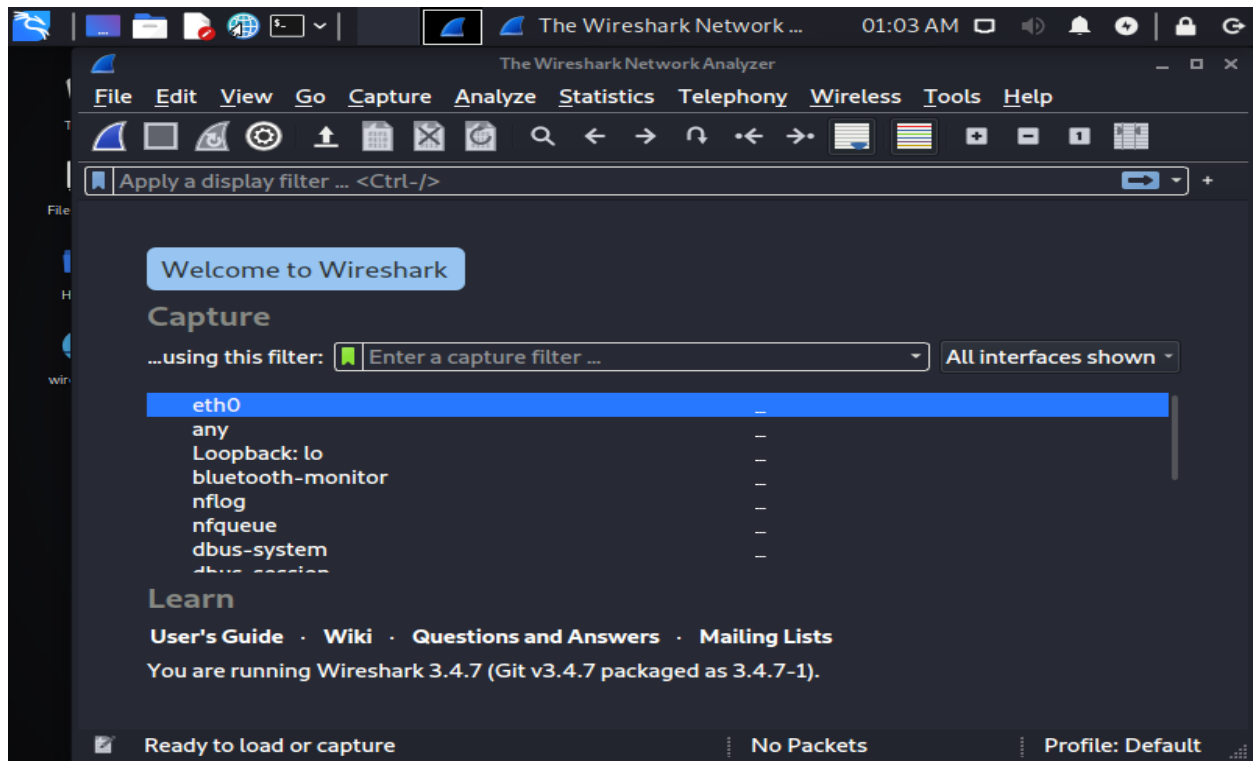
Using Wireshark on a Kali Linux VM to intercept network traffic. Wireshark will capture network packets from the traffic being sent/received from the host computer and present the data captured in a readable format.

## Introduction

Wireshark on a Kali Linux VM will be used to demonstrate how Wireshark captures and displays packets, along with useful tools Wireshark provides to make protocol analysis much easier. Communication between an Ubuntu VM and a Kali Linux VM using netcat will be demonstrated as an example to how Wireshark will analyze the network packets being sent from these two machines.

## Summary of Results

In Kali Linux, Wireshark will be used to observe the network packet exchanges. Here selecting 'any' will allow Wireshark to listen on all available interfaces.



Wireshark provides a variety of tools to make network packets easy to analyze, such as a display filter, the list of captured packets, the details to any selected packet, and the ASCII and hexadecimal contained within the packets. The display filter helps with having to narrow down specific types of packets, for example typing 'HTTP' for finding packets received from websites, or looking up a specific ip with a 'ip.addr == [ipaddress]' command.

The image shows the Wireshark network protocol analyzer interface. At the top, the status bar indicates 'Capturing from eth0' and the time is 11:43 PM. The menu bar includes File, Edit, View, Go, Capture, Analyze, Statistics, Telephony, Wireless, Tools, and Help. Below the menu is a toolbar with various icons for packet capture and analysis. A display filter bar shows 'Apply a display filter ... <Ctrl-/>' and a button to 'Enlarge the main window text'. The packet list pane shows a table of captured packets:

No.	Time	Source	Destination	Protocol	Length	Info
6174	3443.1650571...	fe80::a00:27ff:fe1e...	ff02::16	ICMPv6	130	Mul
6175	3443.3154309...	10.0.0.11	10.0.0.255	UDP	86	576
6176	3445.3404718...	fe80::461c:12ff:fef...	ff02::1	ICMPv6	174	Rou
6177	3445.3529425...	fe80::a00:27ff:fe1e...	ff02::16	ICMPv6	130	Mul
6178	3445.6608194...	fe80::a00:27ff:fe1e...	ff02::16	ICMPv6	130	Mul

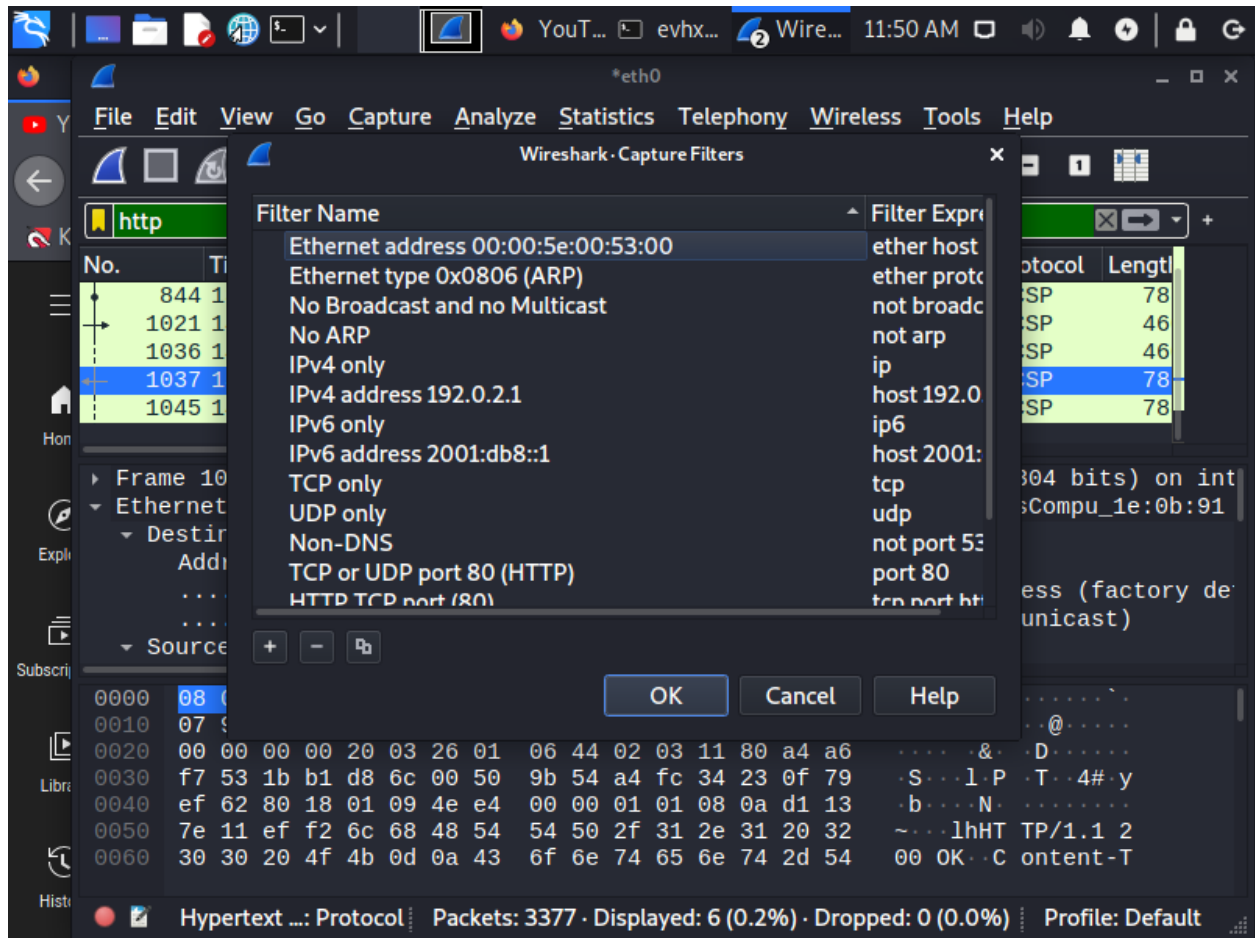
The details pane for the first packet (Frame 1) shows the following information:

- Frame 1: 330 bytes on wire (2640 bits), 330 bytes captured (2640 bits) on interface
- Ethernet II, Src: PcsCompu\_1e:0b:91 (08:00:27:1e:0b:91), Dst: Broadcast (ff:ff:ff:ff:ff:ff)
- Internet Protocol Version 4, Src: 0.0.0.0, Dst: 255.255.255.255
- User Datagram Protocol, Src Port: 68, Dst Port: 67
- Dynamic Host Configuration Protocol (Request)

The packet bytes pane shows the raw data in hexadecimal and ASCII. The first few bytes are ff ff ff ff ff ff 08 00 27 1e 0b 91 08 00 45 c0, which correspond to the Ethernet II header. The ASCII column shows the corresponding characters: < @ @ 8 . . . . . E .

At the bottom, the status bar shows 'eth0: <live capture in progress>' and 'Packets: 6178 · Displayed: 6178 (100.0%) · Profile: Default'.

Network packets can also be filtered by going into the 'Capture Filters' command in the menu.

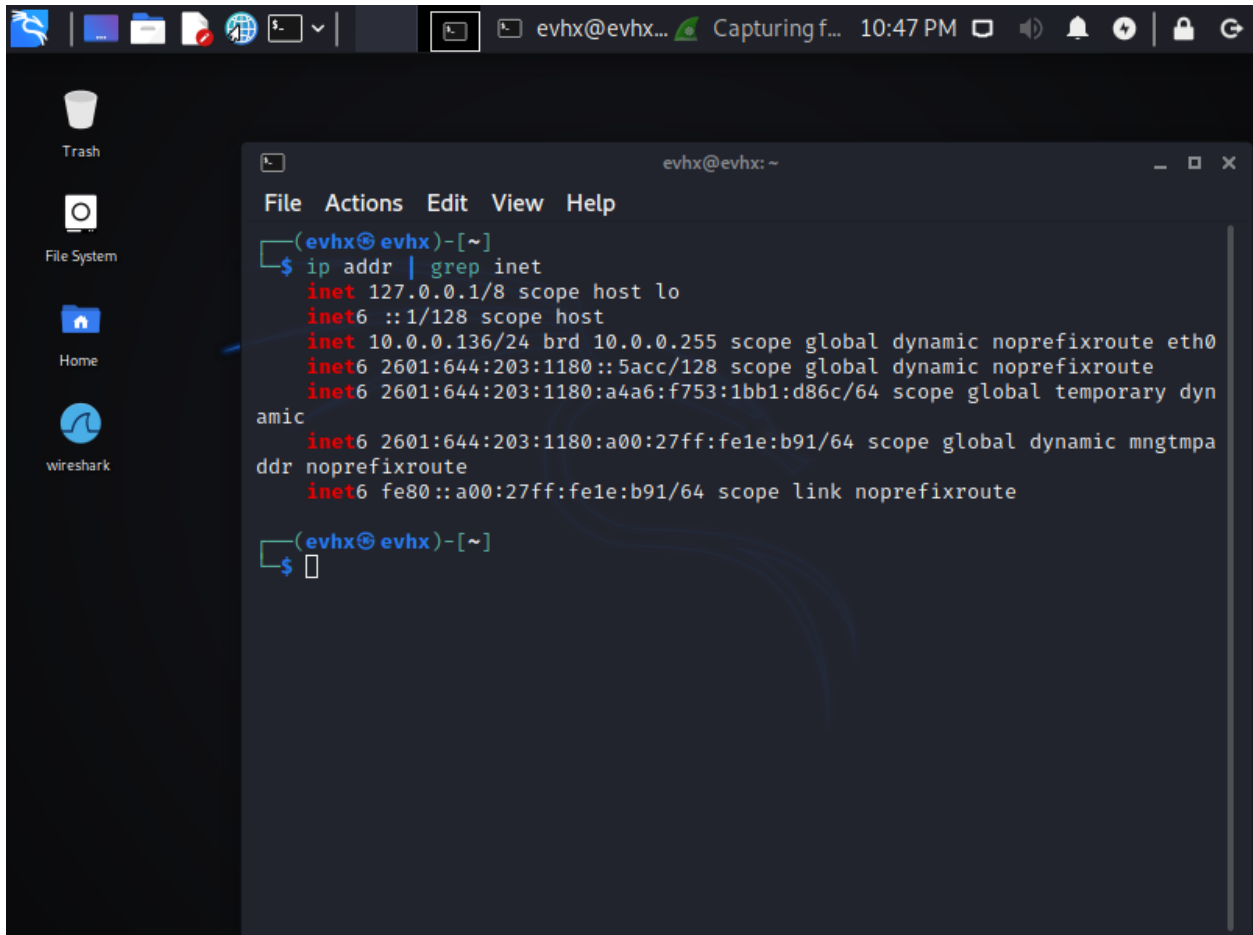


## Capturing Packets from Communication Between a Ubuntu VM and a Kali Linux VM

In both Kali Linux and Ubuntu, open the terminal and type in 'ip addr | grep inet' to retrieve the IP address of the machines.

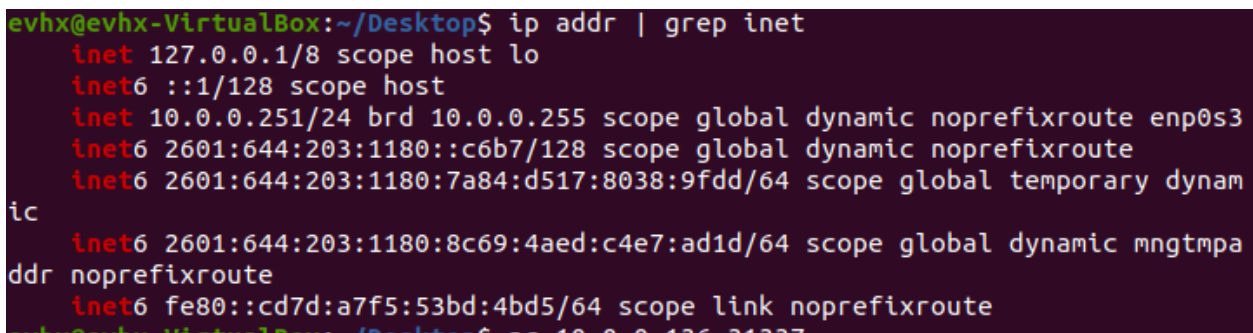
Kali Linux IP: 10.0.0.136

Ubuntu IP: 10.0.0.251



The screenshot shows a Kali Linux desktop environment. A terminal window is open, displaying the command `ip addr | grep inet` and its output. The output lists several network interfaces and their IP addresses, including the primary interface `eth0` with IP `10.0.0.136`. The desktop background is dark, and the terminal window has a dark theme.

```
(evhx@evhx)-[~]  
$ ip addr | grep inet  
inet 127.0.0.1/8 scope host lo  
inet6 ::1/128 scope host  
inet 10.0.0.136/24 brd 10.0.0.255 scope global dynamic noprefixroute eth0  
inet6 2601:644:203:1180::5acc/128 scope global dynamic noprefixroute  
inet6 2601:644:203:1180:a4a6:f753:1bb1:d86c/64 scope global temporary dyn  
amic  
inet6 2601:644:203:1180:a00:27ff:fe1e:b91/64 scope global dynamic mngtmpa  
ddr noprefixroute  
inet6 fe80::a00:27ff:fe1e:b91/64 scope link noprefixroute  
  
(evhx@evhx)-[~]  
$
```



The screenshot shows a terminal window with the command `ip addr | grep inet` and its output. The output lists several network interfaces and their IP addresses, including the primary interface `enp0s3` with IP `10.0.0.251`. The terminal has a dark background with red and green text for the command and output respectively.

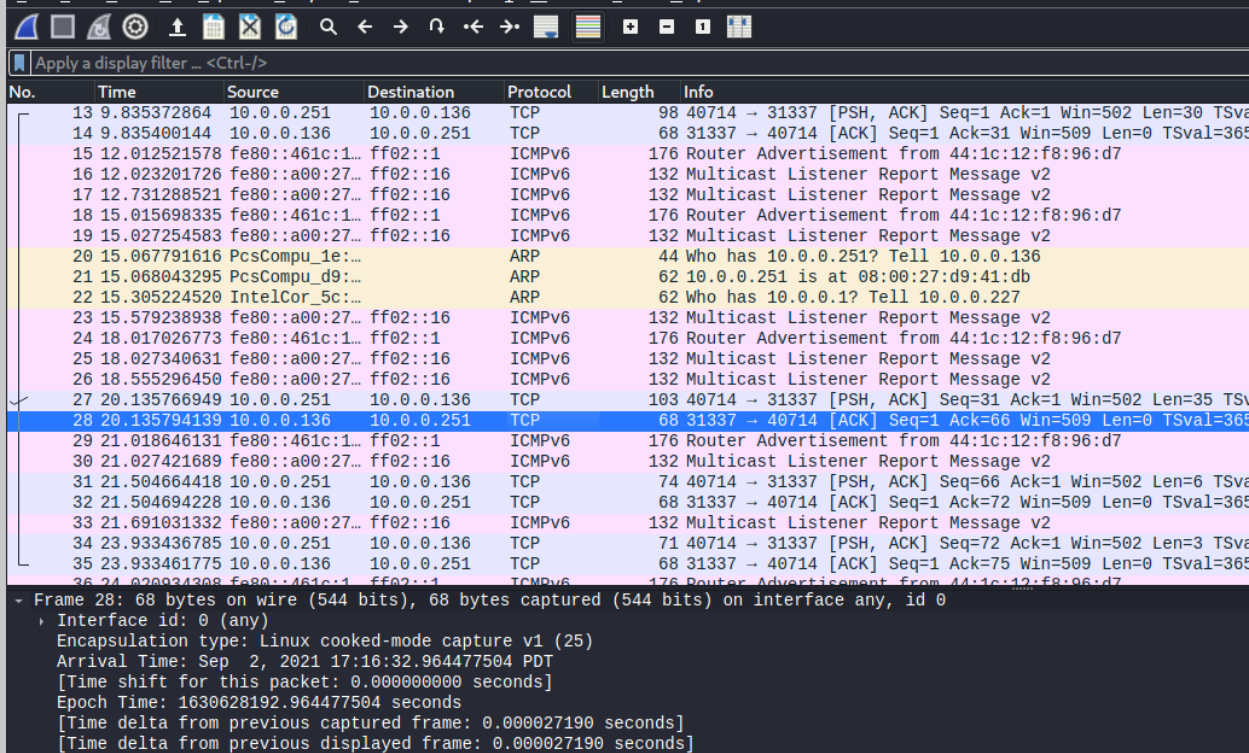
```
evhx@evhx-VirtualBox:~/Desktop$ ip addr | grep inet  
inet 127.0.0.1/8 scope host lo  
inet6 ::1/128 scope host  
inet 10.0.0.251/24 brd 10.0.0.255 scope global dynamic noprefixroute enp0s3  
inet6 2601:644:203:1180::c6b7/128 scope global dynamic noprefixroute  
inet6 2601:644:203:1180:7a84:d517:8038:9fdd/64 scope global temporary dynam  
ic  
inet6 2601:644:203:1180:8c69:4aed:c4e7:ad1d/64 scope global dynamic mngtmpa  
ddr noprefixroute  
inet6 fe80::cd7d:a7f5:53bd:4bd5/64 scope link noprefixroute  
evhx@evhx-VirtualBox:~/Desktop$
```

Here the command 'nc' will be used, which stands for netcat--a utility that can be used to read and write network connections in relation to TCP, UDP and UNIX sockets. Here a socket will be created in Kali Linux using the command 'nc -l -p [socket number]', and then in Ubuntu we will use the command 'nc [IP address of machine being contacted with] [socket number]' to connect to the Kali Linux machine, and communicate the displayed message.

```
evhx@evhx: ~  
File Actions Edit View Help  
(evhx@evhx)-[~]  
$ man nc  
  
(evhx@evhx)-[~]  
$ ip addr | grep inet  
inet 127.0.0.1/8 scope host lo  
inet6 ::1/128 scope host  
inet 10.0.0.136/24 brd 10.0.0.255 scope global dynamic noprefixroute eth0  
inet6 2601:644:203:1180::5acc/128 scope global dynamic noprefixroute  
inet6 2601:644:203:1180:6ca5:67ea:3a21:aa0c/64 scope global temporary dynamic  
inet6 2601:644:203:1180:a00:27ff:fe1e:b91/64 scope global dynamic mngtmpaddr noprefixroute  
inet6 fe80::a00:27ff:fe1e:b91/64 scope link noprefixroute  
  
(evhx@evhx)-[~]  
$ nc -l -p 31337  
hello meow  
how are you feeling mr.linux  
I am talking to you from mr.Ubuntu  
lolol  
:D  
█
```

```
evhx@evhx-VirtualBox: ~/Desktop  
evhx@evhx-VirtualBox:~/Desktop$ nc 10.0.2.136 51047  
evhx@evhx-VirtualBox:~/Desktop$ nc 10.0.2.136 51047  
this is a tesevhx@evhx-VirtualBox:~/Desktop$ this is a test meow  
Command 'this' not found, did you mean:  
  command 'thin' from deb thin (1.7.2-1build1)  
Try: sudo apt install <deb name>  
  
evhx@evhx-VirtualBox:~/Desktop$ meow  
meow: command not found  
evhx@evhx-VirtualBox:~/Desktop$ ip addr | grep inet  
inet 127.0.0.1/8 scope host lo  
inet6 ::1/128 scope host  
inet 10.0.0.251/24 brd 10.0.0.255 scope global dynamic noprefixroute enp0s3  
inet6 2601:644:203:1180::c6b7/128 scope global dynamic noprefixroute  
inet6 2601:644:203:1180:7a84:d517:8038:9fdd/64 scope global temporary dynam  
inet6 2601:644:203:1180:8c69:4aed:c4e7:ad1d/64 scope global dynamic mngtmpa  
ddr noprefixroute  
inet6 fe80::cd7d:a7f5:53bd:4bd5/64 scope link noprefixroute  
evhx@evhx-VirtualBox:~/Desktop$ nc 10.0.0.136 31337  
hello meow  
how are you feeling mr.linux  
I am talking to you from mr.Ubuntu  
lolol  
:D  
█
```

In Kali Linux, using Wireshark we can see the IP address from the Ubuntu machine as a TCP protocol.



No.	Time	Source	Destination	Protocol	Length	Info
13	9.835372864	10.0.0.251	10.0.0.136	TCP	98	40714 → 31337 [PSH, ACK] Seq=1 Ack=1 Win=502 Len=30 TSval=365
14	9.835400144	10.0.0.136	10.0.0.251	TCP	68	31337 → 40714 [ACK] Seq=1 Ack=31 Win=509 Len=0 TSval=365
15	12.012521578	fe80::461c:1...	ff02::1	ICMPv6	176	Router Advertisement from 44:1c:12:f8:96:d7
16	12.023201726	fe80::a00:27...	ff02::16	ICMPv6	132	Multicast Listener Report Message v2
17	12.731288521	fe80::a00:27...	ff02::16	ICMPv6	132	Multicast Listener Report Message v2
18	15.015698335	fe80::461c:1...	ff02::1	ICMPv6	176	Router Advertisement from 44:1c:12:f8:96:d7
19	15.027254583	fe80::a00:27...	ff02::16	ICMPv6	132	Multicast Listener Report Message v2
20	15.067791616	PcsCompu_1e:...		ARP	44	Who has 10.0.0.251? Tell 10.0.0.136
21	15.068043295	PcsCompu_d9:...		ARP	62	10.0.0.251 is at 08:00:27:d9:41:db
22	15.305224520	IntelCor_5c:...		ARP	62	Who has 10.0.0.1? Tell 10.0.0.227
23	15.579238938	fe80::a00:27...	ff02::16	ICMPv6	132	Multicast Listener Report Message v2
24	18.017026773	fe80::461c:1...	ff02::1	ICMPv6	176	Router Advertisement from 44:1c:12:f8:96:d7
25	18.027340631	fe80::a00:27...	ff02::16	ICMPv6	132	Multicast Listener Report Message v2
26	18.555296450	fe80::a00:27...	ff02::16	ICMPv6	132	Multicast Listener Report Message v2
✓ 27	20.135766949	10.0.0.251	10.0.0.136	TCP	103	40714 → 31337 [PSH, ACK] Seq=31 Ack=1 Win=502 Len=35 TSval=365
28	20.135794139	10.0.0.136	10.0.0.251	TCP	68	31337 → 40714 [ACK] Seq=1 Ack=66 Win=509 Len=0 TSval=365
29	21.018646131	fe80::461c:1...	ff02::1	ICMPv6	176	Router Advertisement from 44:1c:12:f8:96:d7
30	21.027421689	fe80::a00:27...	ff02::16	ICMPv6	132	Multicast Listener Report Message v2
31	21.504664418	10.0.0.251	10.0.0.136	TCP	74	40714 → 31337 [PSH, ACK] Seq=66 Ack=1 Win=502 Len=6 TSval=365
32	21.504694228	10.0.0.136	10.0.0.251	TCP	68	31337 → 40714 [ACK] Seq=1 Ack=72 Win=509 Len=0 TSval=365
33	21.691031332	fe80::a00:27...	ff02::16	ICMPv6	132	Multicast Listener Report Message v2
34	23.933436785	10.0.0.251	10.0.0.136	TCP	71	40714 → 31337 [PSH, ACK] Seq=72 Ack=1 Win=502 Len=3 TSval=365
35	23.933461775	10.0.0.136	10.0.0.251	TCP	68	31337 → 40714 [ACK] Seq=1 Ack=75 Win=509 Len=0 TSval=365
36	24.020934308	fe80::461c:1...	ff02::1	ICMPv6	176	Router Advertisement from 44:1c:12:f8:96:d7

Frame 28: 68 bytes on wire (544 bits), 68 bytes captured (544 bits) on interface any, id 0  
Interface id: 0 (any)  
Encapsulation type: Linux cooked-mode capture v1 (25)  
Arrival Time: Sep 2, 2021 17:16:32.964477504 PDT  
[Time shift for this packet: 0.000000000 seconds]  
Epoch Time: 1630628192.964477504 seconds  
[Time delta from previous captured frame: 0.000027190 seconds]  
[Time delta from previous displayed frame: 0.000027190 seconds]

To view all of the packets being sent/received from the Ubuntu VM, type in 'ip.addr==10.0.0.136' or 'ip.addr==10.0.0.251' into the filter to view the packets being communicated between the two machines.

ip.addr==10.0.0.136							
No.	Time	Source	Destination	Protocol	Length	Info	
13	9.835372864	10.0.0.251	10.0.0.136	TCP	98	40714 → 31337	[PSH, ACK] Seq=1 Ack=1 Win=502 Len=30 TSval=3345
14	9.835400144	10.0.0.136	10.0.0.251	TCP	68	31337 → 40714	[ACK] Seq=1 Ack=31 Win=509 Len=0 TSval=365219638
27	20.135766949	10.0.0.251	10.0.0.136	TCP	103	40714 → 31337	[PSH, ACK] Seq=31 Ack=1 Win=502 Len=35 TSval=334
28	20.135794139	10.0.0.136	10.0.0.251	TCP	68	31337 → 40714	[ACK] Seq=1 Ack=66 Win=509 Len=0 TSval=365220668
31	21.504664418	10.0.0.251	10.0.0.136	TCP	74	40714 → 31337	[PSH, ACK] Seq=66 Ack=1 Win=502 Len=6 TSval=3345
32	21.504694228	10.0.0.136	10.0.0.251	TCP	68	31337 → 40714	[ACK] Seq=1 Ack=72 Win=509 Len=0 TSval=365220805
34	23.933436785	10.0.0.251	10.0.0.136	TCP	71	40714 → 31337	[PSH, ACK] Seq=72 Ack=1 Win=502 Len=3 TSval=3345
35	23.933461775	10.0.0.136	10.0.0.251	TCP	68	31337 → 40714	[ACK] Seq=1 Ack=75 Win=509 Len=0 TSval=365221048

- Frame 28: 68 bytes on wire (544 bits), 68 bytes captured (544 bits) on interface any, id 0

Interface id: 0 (any)

Encapsulation type: Linux cooked-mode capture v1 (25)

Arrival Time: Sep 2, 2021 17:16:32.964477504 PDT

[Time shift for this packet: 0.000000000 seconds]

Epoch Time: 1630628192.964477504 seconds

[Time delta from previous captured frame: 0.000027190 seconds]

[Time delta from previous displayed frame: 0.000027190 seconds]

[Time since reference or first frame: 20.135794139 seconds]

Frame Number: 28

Frame Length: 68 bytes (544 bits)

Capture Length: 68 bytes (544 bits)

[Frame is marked: False]

[Frame is ignored: False]

[Protocols in frame: sll:ethertype:ip:tcp]

[Coloring Rule Name: TCP]

[Coloring Rule String: tcp]

Linux cooked capture v1

Internet Protocol Version 4, Src: 10.0.0.136, Dst: 10.0.0.251



Wireshark is able to thoroughly inspect packet data and display it in an easy-to-read format. Data such as:

No. : Order of the packet when captured.

Time: How long it took for the packet to be captured.

Source: Address of the system sent from the packet.

Destination: Address of the system receiving the packet.

Protocol: Type of packet.

Length: Length of the packet in bytes.

Info: More information about the packet.

Packet Bytes: Data displayed in hexadecimal or ASCII (message communicated between the Linux and Ubuntu machine is located here)

The image shows a Wireshark packet capture analysis. At the top, a filter is set to `ip.addr==10.0.0.136`. Below this is a table of captured packets. Packet 27 is selected, showing its details and raw data.

No.	Time	Source	Destination	Protocol	Length	Info
13	9.835372864	10.0.0.251	10.0.0.136	TCP	98	40714 → 31337 [U]
14	9.835400144	10.0.0.136	10.0.0.251	TCP	68	31337 → 40714 [U]
27	20.135766949	10.0.0.251	10.0.0.136	TCP	103	40714 → 31337 [U]
28	20.135794139	10.0.0.136	10.0.0.251	TCP	68	31337 → 40714 [U]
31	21.504664418	10.0.0.251	10.0.0.136	TCP	74	40714 → 31337 [U]
32	21.504694228	10.0.0.136	10.0.0.251	TCP	68	31337 → 40714 [U]
34	23.933436785	10.0.0.251	10.0.0.136	TCP	71	40714 → 31337 [U]
35	23.933461775	10.0.0.136	10.0.0.251	TCP	68	31337 → 40714 [U]

Sequence Number (raw): 366414646  
[Next Sequence Number: 66 (relative sequence number)]  
Acknowledgment Number: 1 (relative ack number)  
Acknowledgment number (raw): 1199180  
1000 .... = Header Length: 32 bytes (8)  
+ Flags: 0x018 (PSH, ACK)  
Window: 502  
[Calculated window size: 502]  
[Window size scaling factor: -1 (unknown)]  
Checksum: 0xef80 [unverified]  
[Checksum Status: Unverified]  
Urgent Pointer: 0  
+ Options: (12 bytes), No-Operation (NOP), No-Operation (NOP), Timestamps  
+ [SEQ/ACK analysis]  
+ [Timestamps]  
TCP payload (35 bytes)  
- Data (35 bytes)  
Data: 4920616d2074616c6b696e6720746f752067726f6d722e5562756e74750a  
[Length: 35]  
0000 00 00 00 01 00 06 08 00 27 d9 41 db 00 00 08 00 ..... ! A .....  
0010 45 00 00 57 82 d3 40 00 40 06 a2 4b 0a 00 00 fb E..W...@..@..K....  
0020 0a 00 00 88 9f 0a 7a 69 15 d7 0b 36 00 12 4c 4c .....zi...6..LL  
0030 80 18 01 f6 ef 80 00 00 01 01 08 0a c7 6b f9 56 .....k.V  
0040 d9 b0 18 21 49 20 61 6d 20 74 61 6c 6b 69 6e 67 ...!I am talking  
0050 20 74 6f 20 79 6f 75 20 66 72 6f 6d 20 6d 72 2e to you from mr.  
0060 55 62 75 6e 74 75 0a Ubuntu.



**Conclusion**

Wireshark is able to capture network packets and displays the content of the packets in an easy-to-read format, which is especially useful when troubleshooting issues like malicious activity, dropped packets, and in general allowing you to thoroughly analyze every packet in detail. Wireshark is best used when the user has knowledge of how a network operates, in order to understand the details provided by Wireshark.