

H M Mythreya
PES2UG20CS130
CNS-Lab Week-2

Task 2.1A: Sniffing Packets

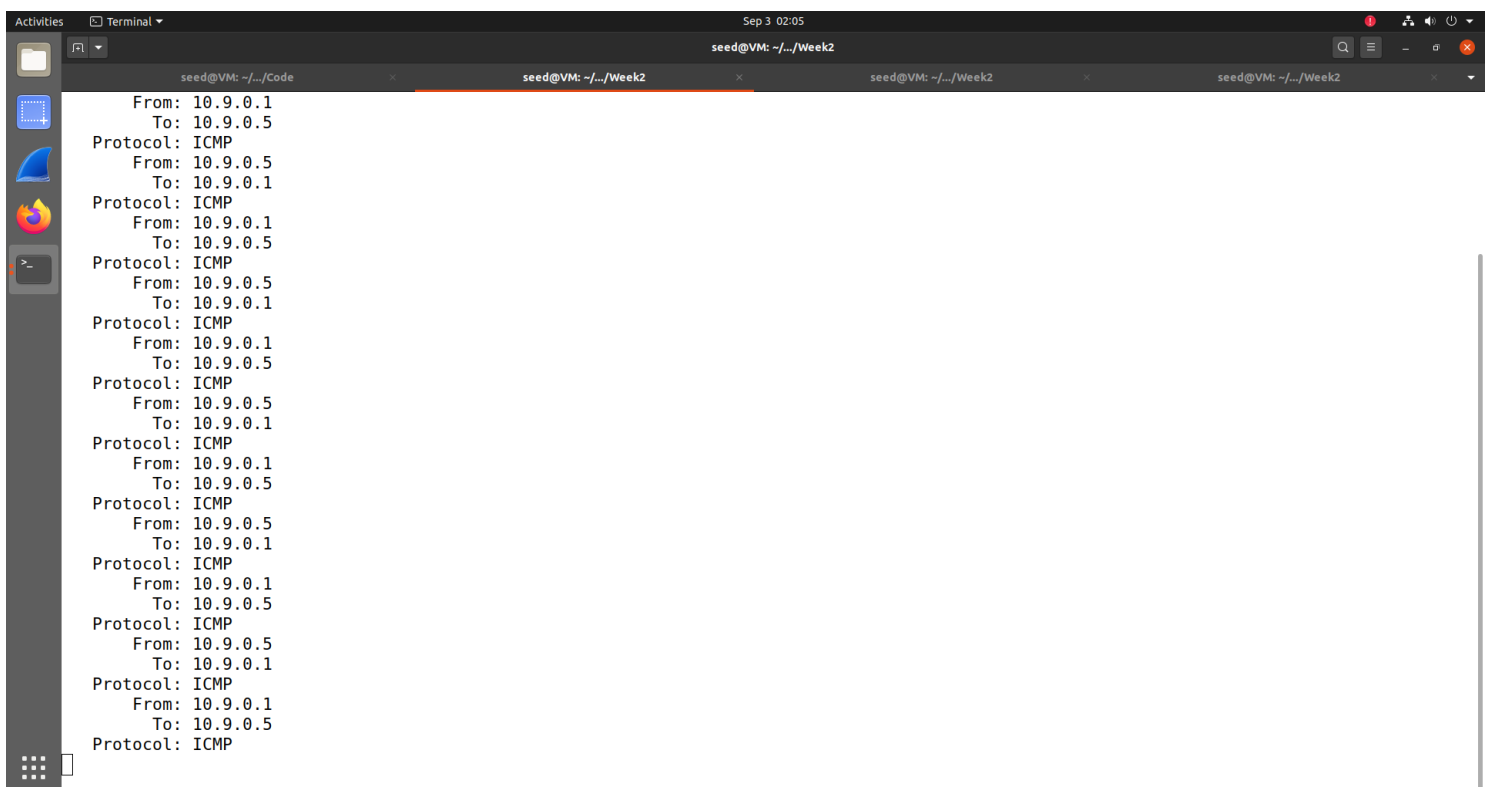
It is important to turn on promiscuous mode so that the packets not addressed to the attacker's machine is also received and can be sniffed.

```
handle = pcap_open_live("br-ea9efb963438", BUFSIZ, 1, 1000, errbuf);
```

^

1 = promiscuous is on

Output

A screenshot of a terminal window titled 'Terminal' with a date and time of 'Sep 3 02:05'. The terminal shows a series of network traffic capture results. Each entry consists of three lines: 'From: 10.9.0.1', 'To: 10.9.0.5', and 'Protocol: ICMP'. These entries are repeated multiple times, alternating between the two IP addresses in the 'From' and 'To' fields. The terminal window has a dark background and a light-colored text. The left sidebar of the terminal shows icons for file manager, web browser, and terminal. The top of the terminal window has a title bar with 'Activities' and 'Terminal' labels.

Important library calls:

Question1)

pcap_open_live() → Used to obtain a packet capture handle in order to look at/sniff at packets in the network

pcap_compile() → Used to compile a string that contains the filter into a filter program

Question2) In root mode, the code doesn't work since the function being called needs root access to work.

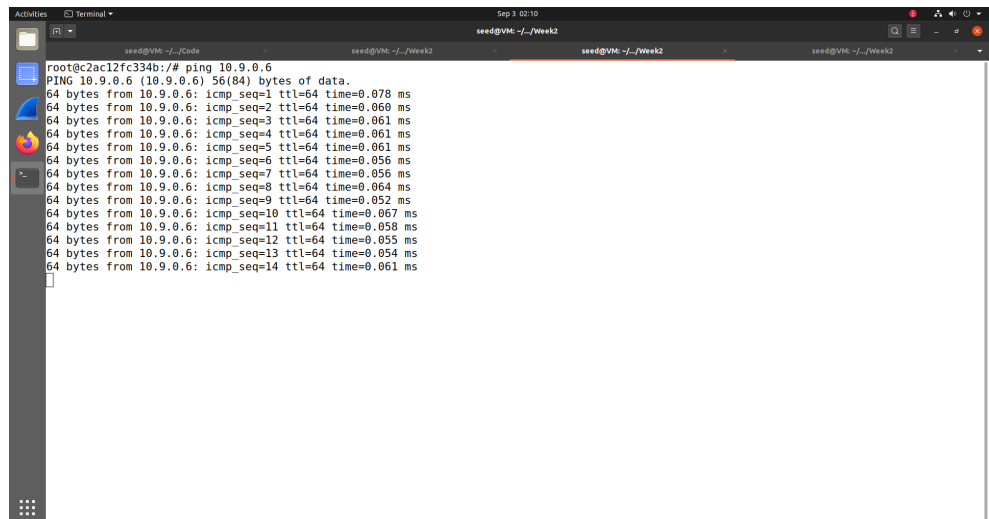
The screenshot shows a Kali Linux desktop environment. On the left is the Dash sidebar with icons for Activities, Home, Recent Documents, and Applications. The top panel displays the date and time as "Sep 3 02:08". A terminal window titled "Terminal" is open, showing a multi-tabbed interface. The active tab is labeled "seed@VM: ~/../Week2". The terminal output consists of several lines of ICMP traffic details:

```
From: 10.9.0.1  
To: 10.9.0.5  
Protocol: ICMP  
From: 10.9.0.5  
To: 10.9.0.1  
Protocol: ICMP  
From: 10.9.0.1  
To: 10.9.0.5  
Protocol: ICMP  
From: 10.9.0.5  
To: 10.9.0.1  
Protocol: ICMP  
From: 10.9.0.1  
To: 10.9.0.5  
Protocol: ICMP  
From: 10.9.0.5  
To: 10.9.0.1  
Protocol: ICMP  
From: 10.9.0.1  
To: 10.9.0.5  
Protocol: ICMP  
From: 10.9.0.5  
To: 10.9.0.1  
Protocol: ICMP  
From: 10.9.0.1  
To: 10.9.0.5  
Protocol: ICMP
```

The prompt at the bottom of the terminal is "root@VM:/volumes#". Below the last line of traffic, there are three lines of input/output:

```
^C  
root@VM:/volumes# su seed  
seed@VM:/volumes$ ./sniff  
Segmentation fault (core dumped)  
seed@VM:/volumes$ su root  
Password:  
root@VM:/volumes#
```

Without Promiscuous mode:



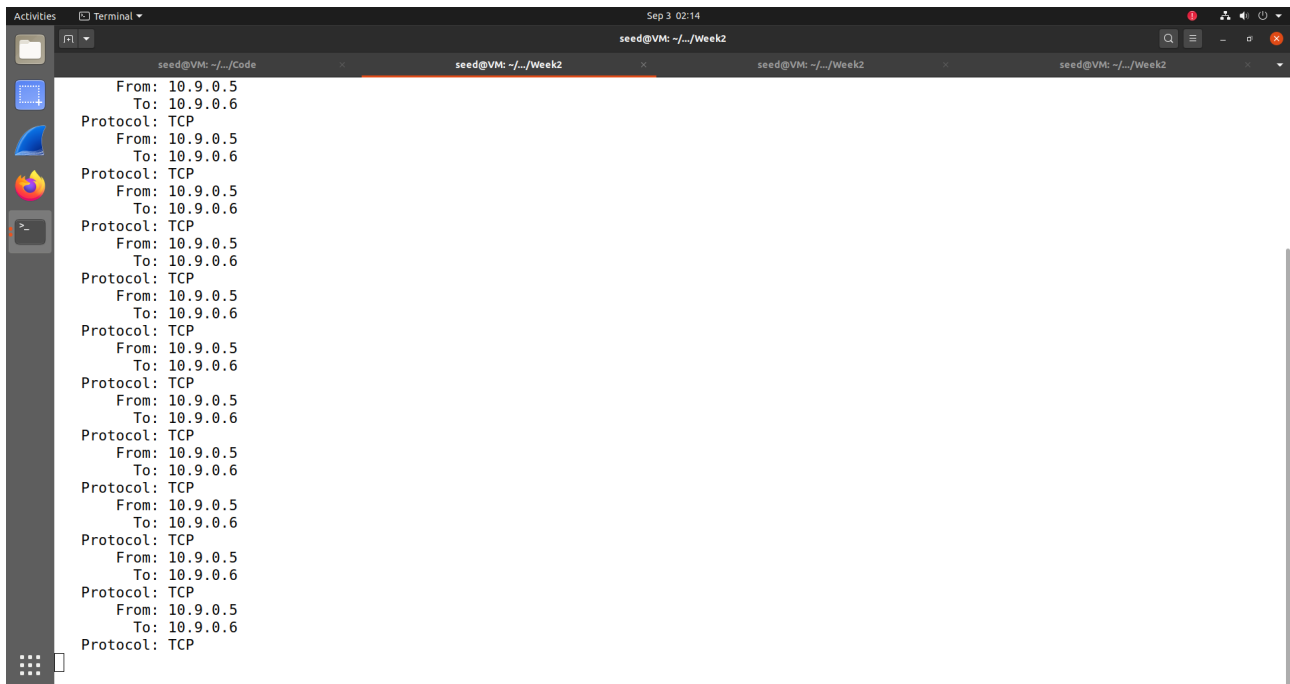
The screenshot shows a Linux desktop environment with a terminal window open. The terminal title bar indicates the session is titled "seed@VM: ~/.../Week2". Inside the terminal, there are several lines of network traffic logs:

```
From: 10.9.0.6  
To: 10.9.0.5  
Protocol: ICMP  
From: 10.9.0.5  
To: 10.9.0.6  
Protocol: ICMP  
From: 10.9.0.6  
To: 10.9.0.5  
Protocol: ICMP  
From: 10.9.0.5  
To: 10.9.0.6  
Protocol: ICMP  
From: 10.9.0.6  
To: 10.9.0.5  
Protocol: ICMP  
From: 10.9.0.5  
To: 10.9.0.6  
Protocol: ICMP  
From: 10.9.0.6  
To: 10.9.0.5  
Protocol: ICMP  
From: 10.9.0.5  
To: 10.9.0.6  
Protocol: ICMP  
From: 10.9.0.6  
To: 10.9.0.5  
Protocol: ICMP
```

The left sidebar of the desktop contains icons for various applications, including a file manager, a web browser, and a terminal. The top status bar shows system information like time and battery level.

Filter 2) TCP, destination port range 10-100.

```
char filter_exp[] = "proto TCP and dst portrange 10-100";
```

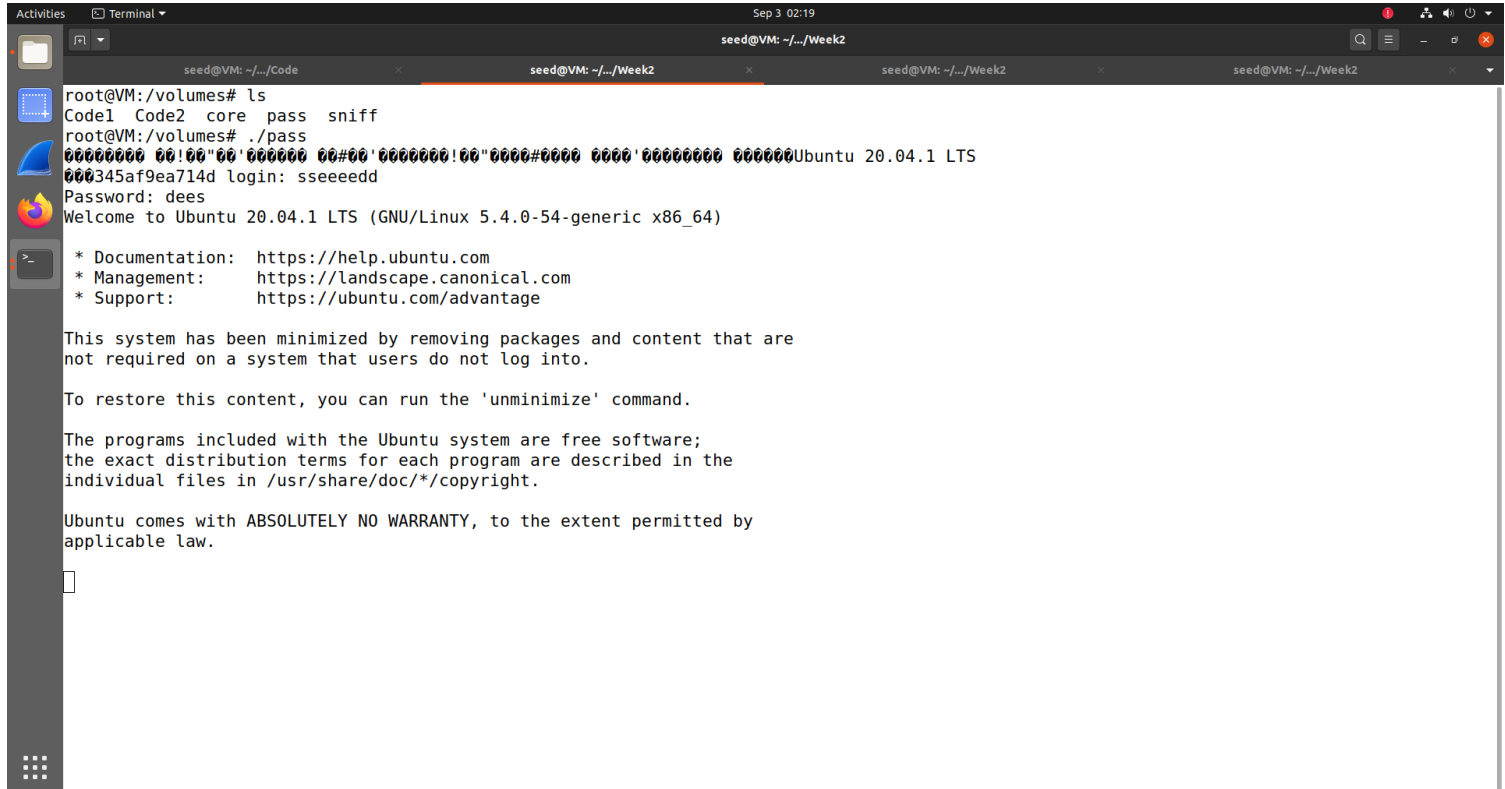


The screenshot shows a terminal window with a dark background and a light-colored text area. The terminal displays a series of network traffic logs. Each log entry consists of four lines: 'From: 10.9.0.5', 'To: 10.9.0.6', 'Protocol: TCP', and 'Protocol: TCP'. The logs are repeated multiple times, indicating a continuous stream of traffic. The terminal window has a title bar that reads 'Terminal' and a date/time indicator 'Sep 3 02:14'. The terminal is divided into several panes, each showing the same log data. The left sidebar of the terminal window contains icons for various applications, including a file manager, a web browser, and a terminal. The overall layout is clean and professional, typical of a network analysis tool or a terminal emulator.

Task 2.1 C: Sniff Passwords on Telnet

```
char filter_exp[] = "port 23";
```

Output



The screenshot shows a terminal window with the following content:

```
root@VM:/volumes# ls
Code1 Code2 core pass sniff
root@VM:/volumes# ./pass
00000000 00!00"00'000000 00#00'000000!00"0000#0000 0000'00000000 000000Ubuntu 20.04.1 LTS
000345af9ea714d login: sseeedd
Password: dees
Welcome to Ubuntu 20.04.1 LTS (GNU/Linux 5.4.0-54-generic x86_64)

* Documentation:  https://help.ubuntu.com
* Management:    https://landscape.canonical.com
* Support:        https://ubuntu.com/advantage

This system has been minimized by removing packages and content that are
not required on a system that users do not log into.

To restore this content, you can run the 'unminimize' command.

The programs included with the Ubuntu system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

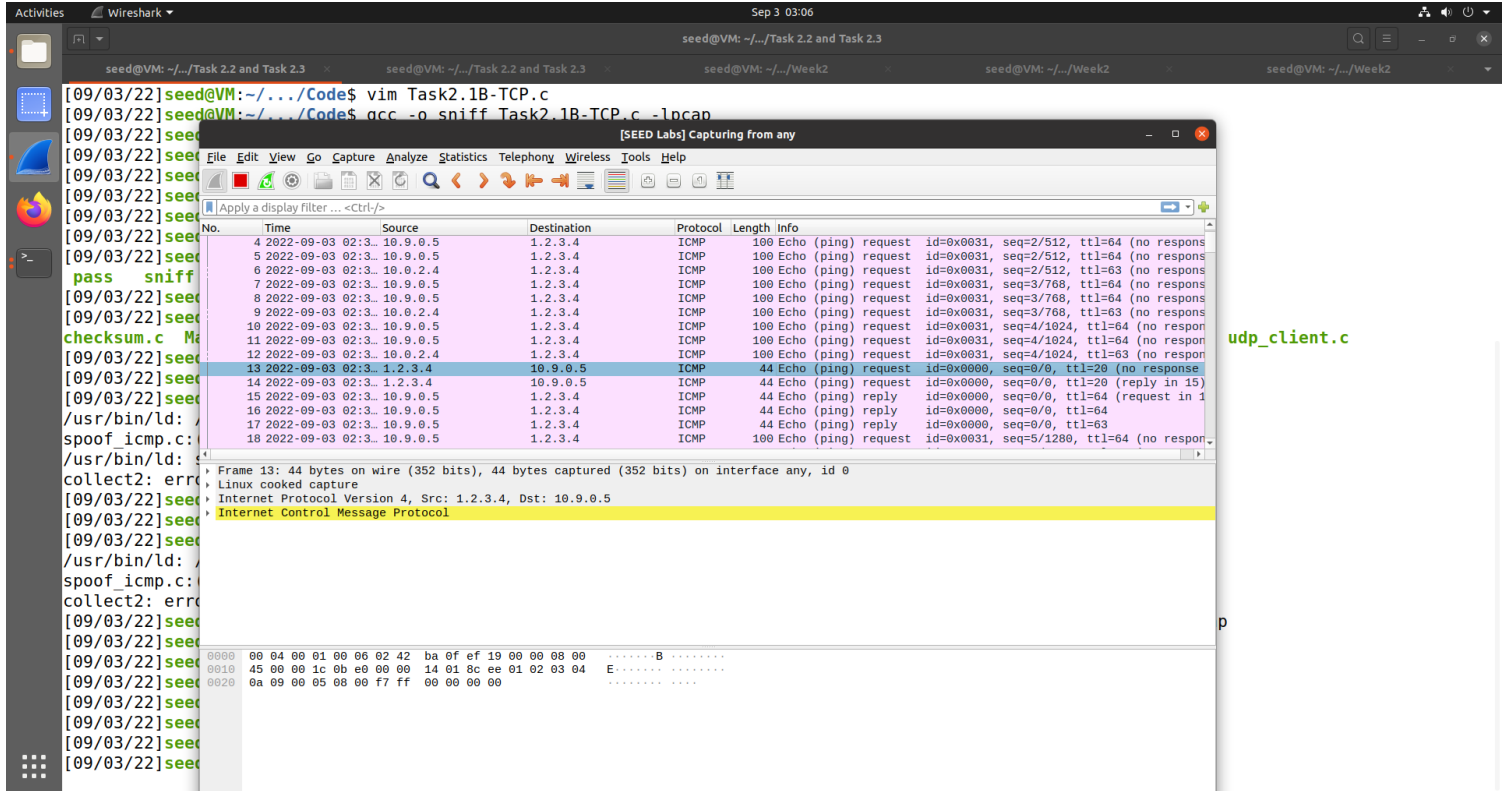
Ubuntu comes with ABSOLUTELY NO WARRANTY, to the extent permitted by
applicable law.
```

As seen above, the password is visible: dees

Task 2.2B: Spoofing Packets

Spoofing is the process of pretending/assuming someone else's ID to trick the victim into accepting communication with the attacker. The attacker's actual ID is hidden.

Output



Question 4: Yes you have to calculate the checksum.

Question 5: Raw sockets require root privileges because creating a raw socket require kernel commands and requires direct interaction with the network card.

Task 2.3 Sniff and then Spoof

The image shows a Linux terminal window with a dark theme. The top bar indicates the date and time as 'Sep 3 14:28'. The terminal is running on a virtual machine named 'seed@VM: ~/../Week2'. The output shows network statistics for two interfaces: 'veth45bdb6b' and 'vethfc4afd'. Both interfaces are configured with flags=4163<UP,BROADCAST,RUNNING,MULTICAST> and mtu 1500. The statistics for 'veth45bdb6b' show 159 RX packets (12.4 KB) and 262 TX packets (21.3 KB). The statistics for 'vethfc4afd' show 20639 RX packets (1.9 MB) and 886 TX packets (50.3 KB). Below the statistics, the terminal shows the execution of a script './sniff_spoof_work' which sends ICMP echo requests from 10.9.0.5 to 1.2.3.4. The output of the script shows four successful ICMP echo requests.

```
seed@VM: ~/../Week2
veth45bdb6b: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet6 fe80::9c38:8fff:fee4:d67f prefixlen 64 scopeid 0x20<link>
    ether 9e:38:8f:e4:d6:7f txqueuelen 0 (Ethernet)
    RX packets 159 bytes 12445 (12.4 KB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 262 bytes 21311 (21.3 KB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

vethfc4afd: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet6 fe80::a3:49ff:fe72:fc9d prefixlen 64 scopeid 0x20<link>
    ether 02:a3:49:72:fc:9d txqueuelen 0 (Ethernet)
    RX packets 20639 bytes 1985015 (1.9 MB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 886 bytes 50321 (50.3 KB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

root@VM:/volumes# ./sniff_spoof_work
From: 10.9.0.5
To: 1.2.3.4
Protocol: ICMP
From: 10.9.0.5
To: 10.9.0.5
Protocol: ICMP
From: 10.9.0.5
To: 1.2.3.4
Protocol: ICMP
From: 10.9.0.5
To: 10.9.0.5
Protocol: ICMP
From: 10.9.0.5
To: 1.2.3.4
Protocol: ICMP
```

The screenshot displays a Kali Linux desktop environment. In the foreground, a terminal window is open, showing the execution of a ping command from the root user on a machine named root@2ac127c334b. The command is `ping 1.2.3.4`, and the output shows successful replies from 1.2.3.4 with varying times and TTL values.

In the background, a Wireshark packet capture window is open, showing a capture on the `eth0` interface. The capture shows a series of ICMP Echo (ping) requests and replies between 192.168.1.100 and 1.2.3.4. The packet list pane shows the first few packets, and the packet details pane shows the structure of an ICMP Echo request.

The terminal output is as follows:

```
root@2ac127c334b:~# ping 1.2.3.4
PING 1.2.3.4 (1.2.3.4) 56(84) bytes of data.
64 bytes from 1.2.3.4: icmp_seq=1 ttl=20 time=350 ms
64 bytes from 1.2.3.4: icmp_seq=2 ttl=20 time=373 ms
64 bytes from 1.2.3.4: icmp_seq=3 ttl=20 time=391 ms
64 bytes from 1.2.3.4: icmp_seq=4 ttl=20 time=392 ms
64 bytes from 1.2.3.4: icmp_seq=5 ttl=20 time=412 ms
```

The Wireshark packet capture shows the following details for the first packet (Frame 1):

- Frame 1: 100 bytes on wire (800 bits), 100 bytes captured (800 bits) on interface any, Id 0
- Ethernet II, Src: Linux cooked capture, Dst: 192.168.1.100
- Internet Protocol Version 4, Src: 192.168.1.100, Dst: 1.2.3.4
- Internet Control Message Protocol

The packet details pane shows the structure of an ICMP Echo request:

- Type: Echo (ping) request
- Code: 0
- Checksum: 0x0000
- Identifier: 0x0000
- Sequence Number: 1

The packet bytes pane shows the raw data of the packet, including the Ethernet II header, IP header, and ICMP Echo request data.

As seen above, the Host A receives a response even though 1.2.3.4 doesn't exist. By sniffing and spoofing, the attacker has claimed the identity of 1.2.3.4 and established a communication with Host A.