

# TCPDump 工具使用实验

## 一【实验目标】

- 理解 TCPDump 的含义。
- 掌握 BackTrack 中网络嗅探的方法。

## 二【实验环境】

- Ubuntu 操作系统
- Windows 10 操作系统

## 三【实验原理】

网络嗅探是指一个能够监视网络数据的软件程序或硬件设备。它可以通过复制数据的方法测试网络连通状况，不会修改数据。使用网络嗅探器，可以了解网络都有些什么信息。网络嗅探器既可以帮助网络工程师解决网络问题，但它同时也可以实现具有恶意的目标。如果网络数据未经加密，而且计算机之间的连接是使用集线器，那网络通信信息，例如用户名和密码、邮件内容等信息，都将很容易被捕获。幸运的是，如果组网使用的是交换机，那么问题会复杂一些，但仍可以捕获信息。

Tcpdump 提供了源代码，公开了接口，具备很强的可扩展性，是网络维护中常用的分析工具。Linux 自带 Tcpdump 工具，用户需要 root 权限将网卡设置为混杂模式才可以捕获网络数据包。

Tcpdump 可以将网络中传送的数据包的“头”完全截获下来提供分析。它支持针对网络层、协议、主机、网络或端口的过滤。作为互联网上经典的系统管理员必备工具，Tcpdump 以其强大的功能，灵活的截取策略，成为每个高级的系统管理员分析网络，排查问题等所必备的工具。

## 四【实验步骤】

实验具体操作步骤如下：

1. 进入 Ubuntu 系统，输入密码“123456”，如图 1 所示。

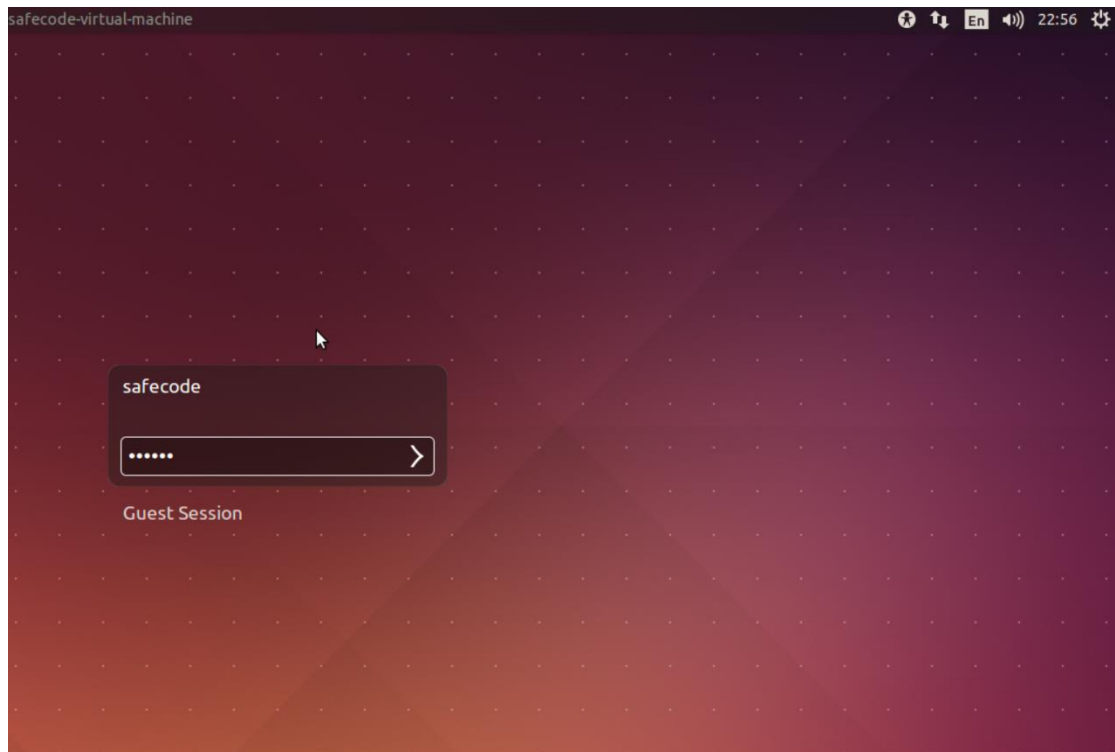


图 1

2. 点击“搜索”按钮，输入“xterm”进入控制台。如图 2 所示。

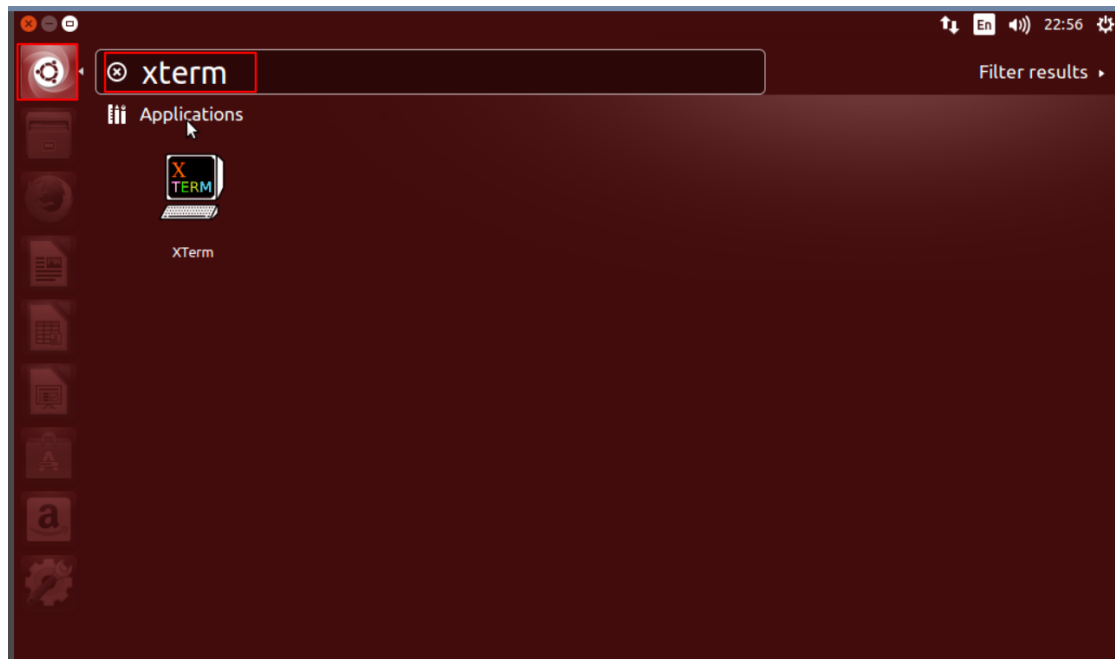


图 2

3. 输入【tcpdump -h】命令，可以查看 tcpdump 的使用方法，如图 3 所示。

```
safecode@safecode-virtual-machine: ~
safecode@safecode-virtual-machine:~$ tcpdump -h
tcpdump version 4.5.1
libpcap version 1.5.3
Usage: tcpdump [-aAbCdEfHhIJKlLnMQpqRStuUvxxX] [-B size] [-c count]
               [-C file_size] [-E algo:secret] [-F file] [-G seconds]
               [-i interface] [-j tstamptype] [-M secret]
               [-P inoutlinout]
               [-r file] [-s snaplen] [-T type] [-V file] [-w file]
               [-W filecount] [-y datalinktype] [-z command]
               [-Z user] [expression]
safecode@safecode-virtual-machine:~$
```

图 3

4. 在命令行中输入【man tcpdump】同样也可以查看 tcpdump 的使用方法，输入【q】停止查看帮助文档，如图 4 所示。

```
System Manager's Manual
TCPDUMP(8) TCPDUMP(8)

NAME
    tcpdump - dump traffic on a network

SYNOPSIS
    tcpdump [-AbCdEfHhIJKlLnMQpqRStuUvxxX] [-B buffer_size] [-c count]
            [-C file_size] [-G rotate_seconds] [-F file]
            [-i interface] [-j tstamp_type] [-M module] [-M secret]
            [-P inoutlinout]
            [-r file] [-V file] [-s snaplen] [-T type] [-w file]
            [-W filecount]
            [-E spi@ipaddr algo:secret,...]
            [-y datalinktype] [-z postrotate-command] [-Z user]
            [expression]

DESCRIPTION
    Tcpdump prints out a description of the contents of packets on a network interface that match the boolean expression. It can also be run with the -w flag, which causes it to save the packet data to a file for later analysis, and/or with the -r flag, which causes it to read from a saved packet file rather than to read packets from a network interface (please note tcpdump is protected via an enforcing apparmor(7) profile in Ubuntu which limits the files tcpdump may access). It can also be run with the -V flag, which causes it to read a list of saved packet files. In all cases, only packets that match expression will be processed by tcpdump.

    Tcpdump will, if not run with the -c flag, continue capturing packets until it is interrupted by a SIGINT signal (generated, for example, by typing your interrupt character, typically control-C) or a SIGTERM signal (typically generated with the kill(1) command); if run with the -c flag, it will capture packets until it is interrupted by a SIGINT or SIGTERM signal or the specified number of packets have been processed.

    When tcpdump finishes capturing packets, it will report counts of:
    Manual page tcpdump(8) line 1 (press h for help or q to quit)
```

图 4

5. 输入命令【sudo -i】。输入密码“123456”，进入 root 权限，如图 5 所示。

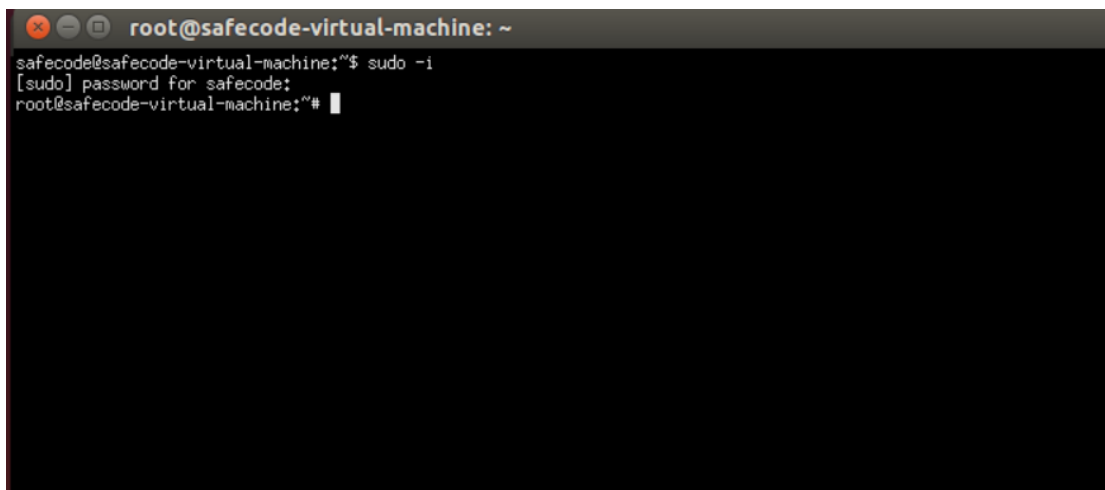


图 5

6. 在一般情况下，只输入【tcpdump】将监听第一个有效网卡中经过的所有数据包，输入【ctrl + c】可停止监听。如图 6 所示。

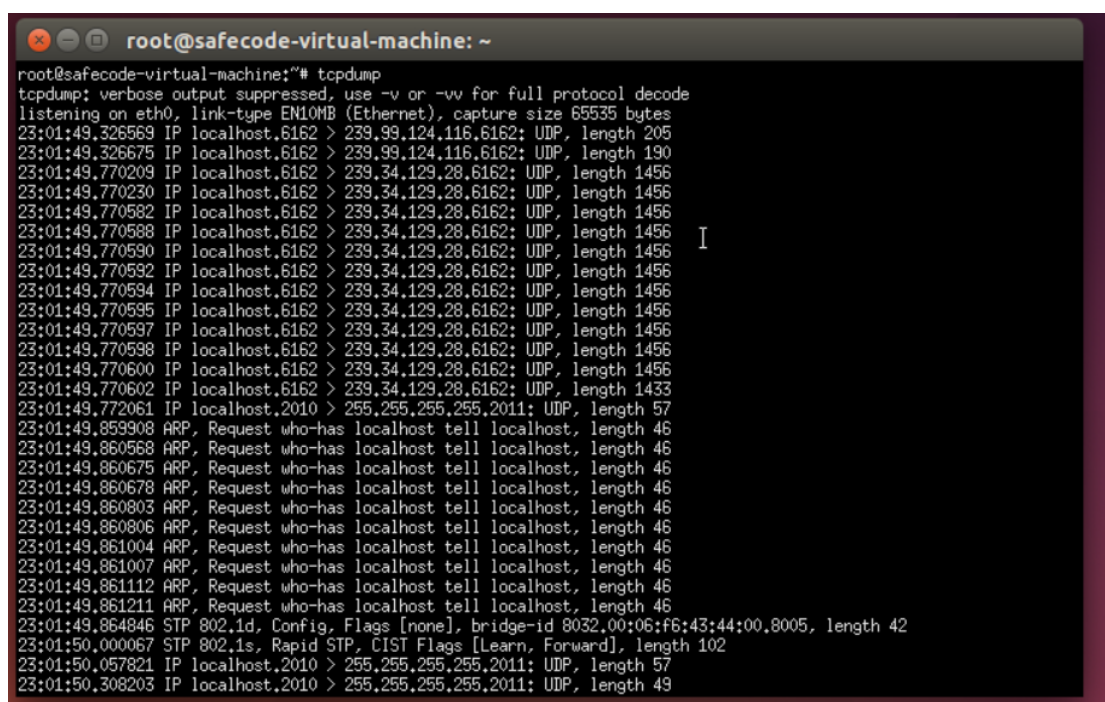


图 6

7. 打开一个新的命令行窗口，在界面中输入【ping 127.0.0.1】，进行本地协议栈自检。如图 7 所示。

```
root@safecode-virtual-machine: ~
safecode@safecode-virtual-machine:~$ sudo -i
[sudo] password for safecode:
root@safecode-virtual-machine:~# ping 127.0.0.1
PING 127.0.0.1 (127.0.0.1) 56(84) bytes of data:
64 bytes from 127.0.0.1: icmp_seq=1 ttl=64 time=0.020 ms
64 bytes from 127.0.0.1: icmp_seq=2 ttl=64 time=0.024 ms
64 bytes from 127.0.0.1: icmp_seq=3 ttl=64 time=0.024 ms
64 bytes from 127.0.0.1: icmp_seq=4 ttl=64 time=0.030 ms
64 bytes from 127.0.0.1: icmp_seq=5 ttl=64 time=0.032 ms
64 bytes from 127.0.0.1: icmp_seq=6 ttl=64 time=0.029 ms
64 bytes from 127.0.0.1: icmp_seq=7 ttl=64 time=0.032 ms
64 bytes from 127.0.0.1: icmp_seq=8 ttl=64 time=0.034 ms
64 bytes from 127.0.0.1: icmp_seq=9 ttl=64 time=0.033 ms
64 bytes from 127.0.0.1: icmp_seq=10 ttl=64 time=0.036 ms
64 bytes from 127.0.0.1: icmp_seq=11 ttl=64 time=0.031 ms
64 bytes from 127.0.0.1: icmp_seq=12 ttl=64 time=0.024 ms
64 bytes from 127.0.0.1: icmp_seq=13 ttl=64 time=0.031 ms
```

图 7

8. 在命令行下输入【**tcpdump -i lo -n**】，（“-n”参数的作用是将主机名换成对应的 IP 地址），继续在另一个窗口中输入【**ping 127.0.0.1**】，查看抓到的数据包并与上一张截图进行对比，可以发现之前的【localhost】换成了【127.0.0.1】。如图 8 所示。

```
0 packets dropped by kernel
root@safecode-virtual-machine:~# tcpdump -i lo
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on lo, link-type EN10MB (Ethernet), capture size 65535 bytes
23:07:51.744220 IP localhost > localhost: ICMP echo request, id 32066, seq 66, length 64
23:07:51.744235 IP localhost > localhost: ICMP echo reply, id 32066, seq 66, length 64
23:07:52.744196 IP localhost > localhost: ICMP echo request, id 32066, seq 67, length 64
23:07:52.744205 IP localhost > localhost: ICMP echo reply, id 32066, seq 67, length 64
23:07:53.744179 IP localhost > localhost: ICMP echo request, id 32066, seq 68, length 64
23:07:53.744189 IP localhost > localhost: ICMP echo reply, id 32066, seq 68, length 64
23:07:54.744188 IP localhost > localhost: ICMP echo request, id 32066, seq 69, length 64
23:07:54.744199 IP localhost > localhost: ICMP echo reply, id 32066, seq 69, length 64
23:07:55.744197 IP localhost > localhost: ICMP echo request, id 32066, seq 70, length 64
23:07:55.744207 IP localhost > localhost: ICMP echo reply, id 32066, seq 70, length 64
23:07:56.744177 IP localhost > localhost: ICMP echo request, id 32066, seq 71, length 64
23:07:56.744186 IP localhost > localhost: ICMP echo reply, id 32066, seq 71, length 64
23:07:57.744202 IP localhost > localhost: ICMP echo request, id 32066, seq 72, length 64
23:07:57.744215 IP localhost > localhost: ICMP echo reply, id 32066, seq 72, length 64
23:07:58.744195 IP localhost > localhost: ICMP echo request, id 32066, seq 73, length 64
23:07:58.744204 IP localhost > localhost: ICMP echo reply, id 32066, seq 73, length 64
```

图 8

9. 在命令行窗口中输入【**tcpdump -i lo -n -c 4**】（“-c”参数可以指定抓包数量，该命令表示只抓取 4 个数据包），在另一个窗口中输入【**ping 127.0.0.1**】，观察显示的抓包情况，可以发现过滤器接收了 8 个数据包（视具体情况而定），但是只捕获了 4 个数据包。如图 9 所示。

```
0 packets dropped by kernel
root@safecode-virtual-machine:~# tcpdump -i lo -n -c 4
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on lo, link-type EN10MB (Ethernet), capture size 65535 bytes
23:09:03.744170 IP 127.0.0.1 > 127.0.0.1: ICMP echo request, id 32066, seq 138, length 64
23:09:03.744178 IP 127.0.0.1 > 127.0.0.1: ICMP echo reply, id 32066, seq 138, length 64
23:09:04.744179 IP 127.0.0.1 > 127.0.0.1: ICMP echo request, id 32066, seq 139, length 64
23:09:04.744193 IP 127.0.0.1 > 127.0.0.1: ICMP echo reply, id 32066, seq 139, length 64
4 packets captured
8 packets received by filter
0 packets dropped by kernel
root@safecode-virtual-machine:~#
```

图 9

10. 在命名行下输入【**tcpdump -D**】查看有效的网口，“-D”参数可以查看主机

有效的网口，在缺省情况下监听第一个网口，如图 10 所示。

```
0 packets dropped by kernel
root@safecode-virtual-machine:~# tcpdump -i
1.eth0
2.any (Pseudo-device that captures on all interfaces)
3.lo
root@safecode-virtual-machine:~#
```

图 10

11. 打开 PC2Windows 10 操作机，密码为“Admin123456”。如图 11 所示。

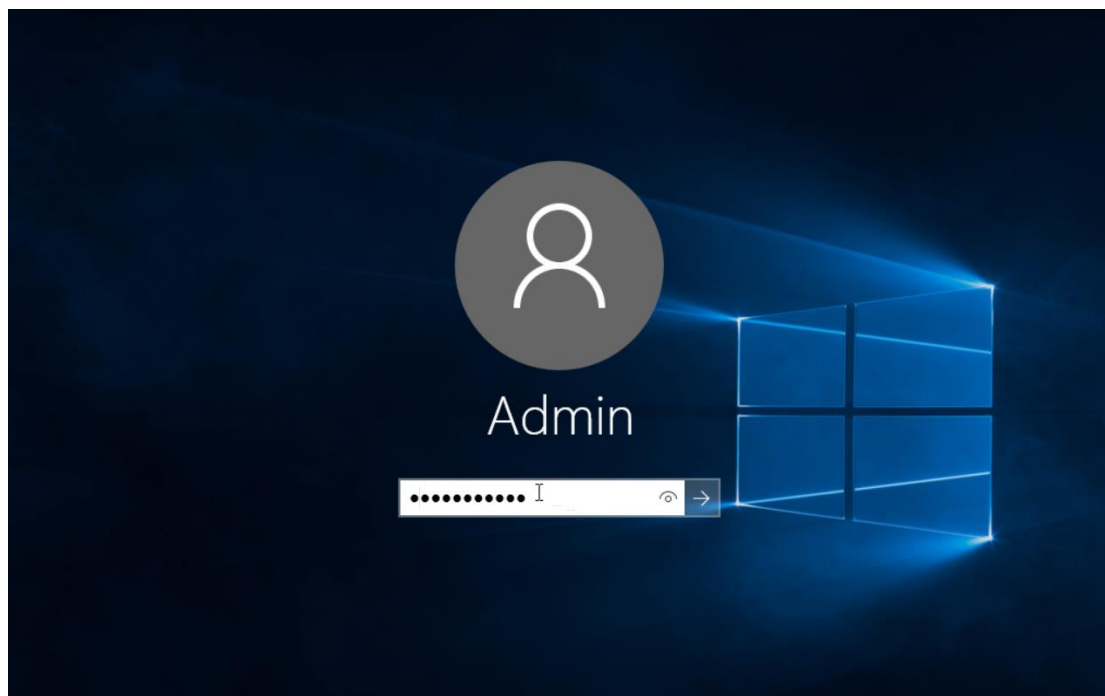


图 11

12. 查看 PC2 的 IP 地址，地址为“192.168.48.198”。如图 12 所示。

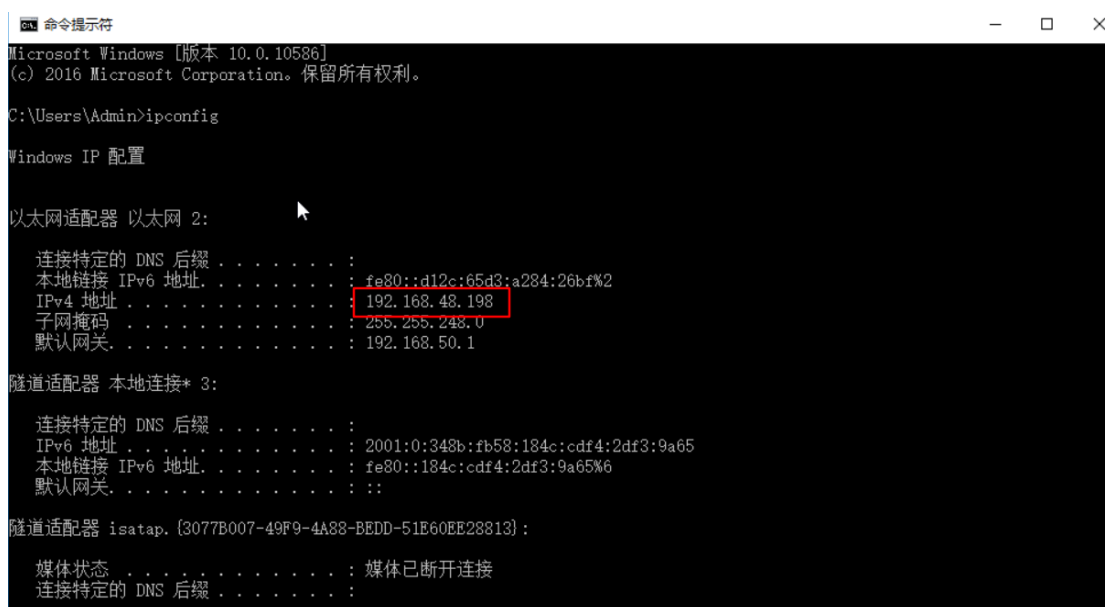


图 12

13. 回到 PC1，打开一个新的命令行界面，输入【ping 192.168.48.198】，可以看到连通的信息。若不能联通可以关闭 Windows 系统防火墙，如图 13 所示。

```
root@safecode-virtual-machine: ~
root@safecode-virtual-machine:~# ping 192.168.48.198
PING 192.168.48.198 (192.168.48.198) 56(84) bytes of data:
64 bytes from 192.168.48.198: icmp_seq=1 ttl=128 time=0.678 ms
64 bytes from 192.168.48.198: icmp_seq=2 ttl=128 time=0.289 ms
64 bytes from 192.168.48.198: icmp_seq=3 ttl=128 time=2.24 ms
64 bytes from 192.168.48.198: icmp_seq=4 ttl=128 time=1.44 ms
```

图 13

14. 【host】关键词可以指定监听的主机名或 IP 地址，在第一个打开的命令行界面输入【tcpdump host 192.168.48.198】，可以看到捕获源 IP 或目的 IP 为“192.168.48.198”的数据包。如图 14 所示。

```
root@safecode-virtual-machine:~# tcpdump host 192.168.48.198
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on eth0, link-type EN10MB (Ethernet), capture size 65535 bytes
23:19:34.280199 IP 192.168.51.222 > 192.168.48.198: ICMP echo request, id 9765, seq 1, length 64
23:19:34.281630 IP 192.168.48.198 > 192.168.51.222: ICMP echo reply, id 9765, seq 1, length 64
23:19:35.281748 IP 192.168.51.222 > 192.168.48.198: ICMP echo request, id 9765, seq 2, length 64
23:19:35.282577 IP 192.168.48.198 > 192.168.51.222: ICMP echo reply, id 9765, seq 2, length 64
23:19:36.280749 IP 192.168.51.222 > 192.168.48.198: ICMP echo request, id 9765, seq 3, length 64
23:19:36.282582 IP 192.168.48.198 > 192.168.51.222: ICMP echo reply, id 9765, seq 3, length 64
23:19:37.282736 IP 192.168.51.222 > 192.168.48.198: ICMP echo request, id 9765, seq 4, length 64
23:19:37.284300 IP 192.168.48.198 > 192.168.51.222: ICMP echo reply, id 9765, seq 4, length 64
23:19:38.284436 IP 192.168.51.222 > 192.168.48.198: ICMP echo request, id 9765, seq 5, length 64
23:19:38.284667 IP 192.168.48.198 > 192.168.51.222: ICMP echo reply, id 9765, seq 5, length 64
23:19:38.903332 ARP, Request who-has 192.168.51.222 (fe:fc:fe:b0:f9:99 (oui Unknown)) tell 192.168.48.198, length 28
23:19:38.903413 ARP, Reply 192.168.51.222 is-at fe:fc:fe:b0:f9:99 (oui Unknown), length 28
23:19:39.284214 IP 192.168.51.222 > 192.168.48.198: ICMP echo request, id 9765, seq 6, length 64
23:19:39.285684 IP 192.168.48.198 > 192.168.51.222: ICMP echo reply, id 9765, seq 6, length 64
```

图 14

## 五【实验思考】

- 思考还可使用 TCPDump 工具嗅探到哪些网络信息？