Task 1.1 A

With root privilege:

For attacker side, we run the codes directly in VM. For client side, we will enter one of the containers of the client, and ping 10.9.0.1 which is the VM’s IP address.

Result:

文本

描述已自动生成

Without root privilege:

This will generate error since socker require superuser priority to invoke.

Result:

文本

描述已自动生成

Task 1.1 B

1. Capture only the ICMP packetfilter=’icmp’
2. Capture any TCP packet that comes from a particular IP and with a destination port number 23

filter=’tcp and dst port 23 and ip host *ip*’

1. Capture packets comes from or to go to a particular subnet

Filter=’net xxx.xxx.xxx.xxx/xx’

Task 1.2

Change the ip address in sample code to 10.0.0.5 which is one of the client container

图形用户界面

中度可信度描述已自动生成

Task 1.3

In the sample code, only send is provided which is not capable of checking the reply message, therefore, we use sr1 instead to check the reply packet. Notice that after checking, I found some routers between may not reply to my ICMP packet thus timeout is needed to prevent blocking.

文本

描述已自动生成

Task 1.4

The basic idea is very simple, which is simply sniff the sending packet inside the subnet and then manipulate the receiving package and send back again.

However, there are some treaty details. First, of course, you need to reverse the order of src and dst. Besides, you also need to reset ttl since sometimes ttl may be too small to travel to target. Then you need to set the type in ICMP which is no reply.

Codes:

文本

描述已自动生成

(basically copy from video lecture)

Testing result:

Required experiments:

文本

描述已自动生成

(1)

图形用户界面, 文本

描述已自动生成

(2)

图形用户界面, 文本

描述已自动生成

In the above experiments, we successfully forged the reply in (1) and (2) except (3). The reason is that from ip route get (shown below), we can easily find that the user program would need to go through 10.9.0.1 to search for ip in (1) and (2) (which is the ip address of VM and function as gateway). For 10.9.0.99 which lies in the same LAN, the user container would first of course, sent out the ARP and get no response. Since no one would actually has this address, so the host A would not get any the target IP then it will never send out an ICMP packet.

文本

描述已自动生成

Ping in host outside the subnet

电脑屏幕的照片上有字

低可信度描述已自动生成

Task 2.1 A

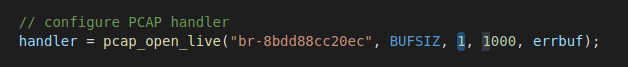
1. Step 1. pcap\_open\_live: configure a pcap handler

Step 2. pcap\_compile: compile the BPF filter from human-readable codes to machine codes

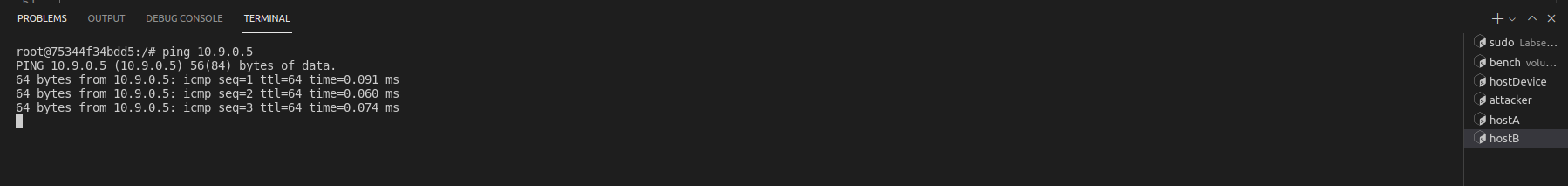
Step 3. pcap\_setfilter: set filter in OS

Step 4. pcap\_loop: start capturing

1. Pcap may need to access low-level device like NIC, which requires root privilege for security and safe issue. Besides, pcap may need to open raw socket which also requires root privilege.
   1. Turn on the promiscuous:



Host B ping Host A:



Attacker:



Before running:

文本

描述已自动生成

During running:



* 1. Turn off promiscuous:



Host B ping Host A:



Attacker:



Before running:



During running:



After turning on the promiscuous mode, the attacker could sniff all the packets passing from its NIC even though it is not the target destination.

Notice that this works only for one host ping another in an intranet, since if a host in the subnet want to communicate with the devices outside the subnet it must go through the attacker (cause it is the gateway of the subnet, i.e. 10.9.0.1).

Task 2.1 B:

Will do later.

Task 2.1 C:

In this case, we could simply print out the data payload in TCP packet. The length of the TCP header could be calculated by data\_offset.

We can modify the code like this:

文本

描述已自动生成

The followings are part of the running result. We can see that the messages caught after the “password” is clearly the password “d”, “e”, “e”, “s”.

文本

描述已自动生成

文本

描述已自动生成

Task 2.2 A:

In this task, for the sake of simplicity, a UDP packet was spoofed and sent from attacker (10.9.0.1) to host A(10.9.0.5).

Different from using python, in C, you need to create a data buffer to construct a packet (in this report, from IP layer) and set all necessary fields. To make things easier, we can actually apply type casting to cast the packet struct on the allocated data buffer and do the operations.

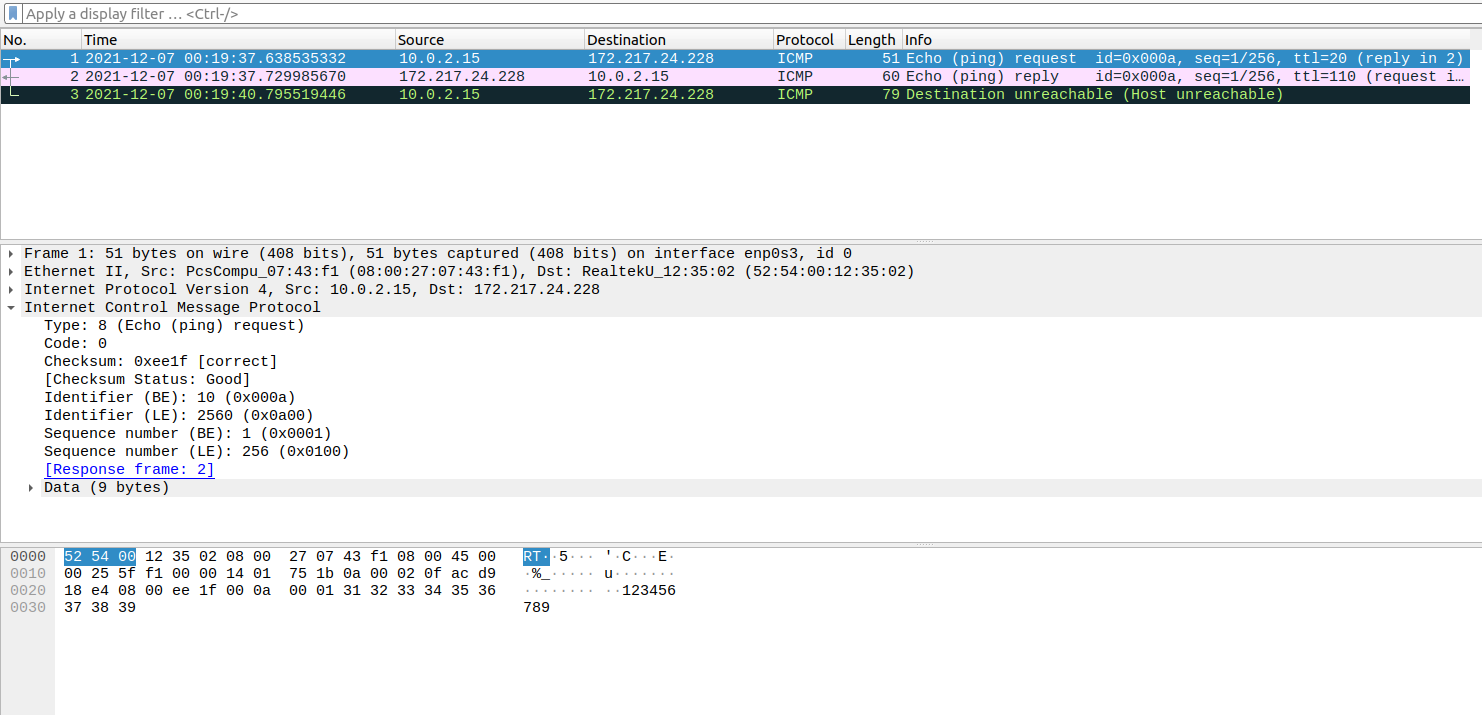
Testing result:

图形用户界面, 文本, 应用程序

描述已自动生成

Task 2.2 B:

To spoof an ICMP, things will be a little bit complicated, especially for the checksum.



Notice that the first blue line is sent by attacker and the middle purple one is sent by server while the last black one is sent by host A (which we fake the source ip).

Question 4:

No, if wring packet size is set, the socket could not send out the packet.

文本

描述已自动生成

Question 5:

No, we could set the checksum field in IP to be 0 then OS would directly calculate the checksum for you.

Question 6:

Creating a raw socket requires root privilege since int may involve some operations with NIC.

If not using root, the sento( ) function would simply crash.

图形用户界面, 文本, 应用程序

描述已自动生成

Task 2.3:

We can simply combine the sniffer and spoofer then we can finish this task.

Running result (attacker sniff and spoof, host A run ping program):

图形用户界面

中度可信度描述已自动生成

Notice that there are multiple duplicated packets since the program would not only spoof the packets from host A but also from target which is the reply to host A.

文本

中度可信度描述已自动生成

Notice that the replying time is generally quite slow, around 2000ms. I guess this is may be attributed to the performance of sendto( ).