Network Security Basics

N1.1. What is the network prefix for each of the following hosts: (1) `10.1.2.200/24` and (2) `10.1.2.200/26`?

N1.2. In Linux, the network interface names used to be `et0`, `eth1`, ..., but now, they become `enp0s3`, `enp1s4`, ..., what is the reason behind this change? (Note: the answer is not in the book, but you can find the reason from the Internet).

N1.3. We write the following program to send a packet. (1) How is the source port number of the packet decided? (2) How is the source IP of the packet decided?

```
data = b"Hello, Server\n"
udp = socket.socket(socket.AF_INET, socket.SOCK_DGRAM)
udp.sendto(data, ("10.9.0.5", 9090))
```

N1.4. When we construct a packet, we need to set the address at each layer. What are the addresses called at he following layers: (1) transport layer, (2) network layer, and (3) data link layer?

N1.5. When we run "`nc -lnu 9090`", what does the options "`lnu`" mean?

N1.6. What does the `bind` do in the following code snippet? What is the meaning of `0.0.0.0`? If we are only interested in the packets coming from the localhost, what should be do?

```
udp = socket.socket(socket.AF_INET, socket.SOCK_DGRAM)
udp.bind(("0.0.0.0", 9090))
```

N1.7. In a server program, we bind the program to a port number, so when a packet comes to that port, the OS knows which program is the recipient of the packet. However, for the client program, we typically do not do the binding. When a reply from the server comes, how does the OS know that this client program is the recipient?

N1.8. Why does the sniffing use the superuser privilege?

N1.9. (1) Write down the `tcpdump` command to print out all the packets to UDP destination port 53 on the interface `xyz`. (2) Write down the `tcpdump` command to print out all the packets from port 1000 to port 53 on the interface `xyz`.

N1.10. Please explain the purpose of each of the arguments in the following Scapy code snippet.

```
pkt = sniff(iface='eth0', filter='icmp', prn=print_pkt)
```

N1.11. The following result of the "`ip address`" command shows all the network interfaces on a host. Please write a Scapy program to sniff the packets transmitted over the `192.168.5.0/24` network. You are only interested in the UDP packets to 8.8.8.8's port 53.

```
$ ip -br address
lo         UNKNOWN   127.0.0.1/8
enp0s3     UP        10.0.5.5/24
enp1s8     UP        192.168.5.0/24
```

N1.12. Please write a code snippet to spoof an ICMP echo request packet that has the following properties: (1) TTL is set to 100, (2) the src IP address is set to `1.2.3.4`.

N1.13. In Scapy, we can use `ls(IP)` to list all the fields of the IP class. See the following. When we spoof an IP packet, we simply use `ip = IP()` without setting any argument, what will be the values of the `ttl` and `src` IP fields of the packet? How about if we do set the `dst` field: `ip = IP(dst="1.1.1.1")`?

```
>>> from scapy.all import *
>>> ls(IP)
len        : ShortField                              = (None)
id         : ShortField                              = (1)
ttl        : ByteField                               = (64)
proto      : ByteEnumField                           = (0)
chksum     : XShortField                             = (None)
src        : SourceIPField                           = (None)
...
```

N1.14. In the following code snippet, we construct an Ethernet frame. (1) What type of object do we get from `pkt.payload.payload`? (2) What will happen if we do this `pkt[UDP]`? (3) How do we get the actual payload string `hello`?

```
pkt = Ether()/IP()/ICMP()/"hello"
```