



Bangladesh University of Engineering and Technology

Computer Science and Engineering Department

ICMP Ping Spoofing

Computer Security Sessional 406

Made by:
Faria Huq
Student ID: 1505052

Contents

1	Introduction	2
2	Attack Design and Implementation	2
2.1	Requirements	2
2.2	Language and Framework Details	2
2.3	Attack Strategy	2
2.4	Attack Demonstration	3
2.4.1	Attack on Android Platform	3
2.4.2	Attack on Linux OS	5
2.5	Was the attack successful and why	9
2.6	Challenges	9
3	Preventive Measure	10
3.1	Future Work	10
4	Conclusion	10

1 Introduction

ICMP ping spoofing is the creation of ICMP packets with a false source address, for the purpose of impersonating another computing system.

ICMP ping spoofing attacks can be used to bypass IP address-based authentication. This process can be very difficult and is primarily used when trust relationships are in place between machines on a network and internal systems. Trust relationships use IP addresses (rather than user logins) to verify machines' identities when attempting to access systems. This enables malicious parties to use spoofing attacks to impersonate machines with access permissions and bypass trust-based network security measures.

2 Attack Design and Implementation

Our main idea is to generate ICMP packets with the spoofed IP stored as the source IP and the server IP as the destination IP.

2.1 Requirements

The device must be running on/ able to simulate Linux OS.

- **Net-tools** for running *ifconfig* to get IP address
- **tcpdump** for running *tcpdump* to monitor ICMP traffic

2.2 Language and Framework Details

We built our attack tool in **C language**. Here we listed related framework and header files used -

Name	Framework/header file	Task
unistd.h	C Header File	Defines miscellaneous symbolic constants and types
sys/socket.h	C Header File	Internet protocol family for socket generation and communication
netinet/ip.h	C Header File	Internet address family that defined IP address, port number etc.
arpa/inet.h	C Header File	Definitions for internet operations
tcpdump	Unix Command	Traces Icmp packet transaction
QT	Framework	Front End Design

Table 1: List of related sources used

We verified our attack both on **termux- a light weight Ubuntu OS made for Android** and Desktop connected on the same LAN.

2.3 Attack Strategy

We designed our attack tool like the following -

- We provide our system with the Spoofed source IP, destination IP , message to be sent and its' count from the frontend
- We create a packet with the spoofed IP as the source
- We open a socket and send the IP packet
- We verify by using tcpdump to track packet transition

The gist of opening a socket and sending a packet

```
int sock = socket(AF_INET, SOCK_RAW, IPPROTO_RAW);

setsockopt(sock, IPPROTO_IP, IP_HDRINCL,
           &enable, sizeof(enable));

dest_info.sin_family = AF_INET;
dest_info.sin_addr = ip->iph_destip;

sendto(sock, ip, ntohs(ip->iph_len), 0,
       (struct sockaddr *)&dest_info, sizeof(dest_info));
close(sock);
```

2.4 Attack Demonstration

We will show our demonstration both on android phone and ubuntu setup.

2.4.1 Attack on Android Platform

- We install **Termux** on our android Devices and get their IP configuration.

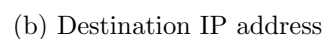


Figure 1: Initial Setup

- We feed the IP addresses from the attacker side and use tcpdump to verify out attack tool is working.

```

oalshi@oalshi: ~/Documents/Sniffing_Spoofing/C_spoof
File Edit View Search Terminal Help
RX packets 0 bytes 0 (0.0 B)
RX errors 0 dropped 0 overruns 0 frame 0
TX packets 0 bytes 0 (0.0 B)
TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

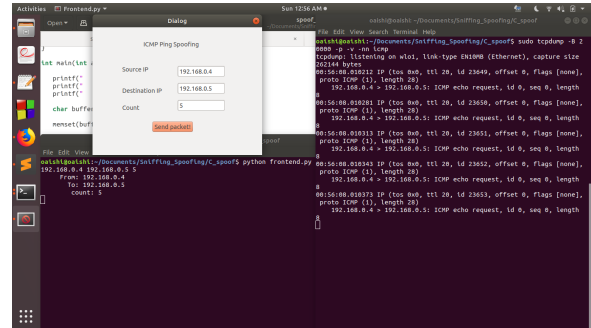
lo: flags=73<UP,LOOPBACK,RUNNING> mtu 65536
inet 127.0.0.1 netmask 255.0.0.0
inet6 ::1 prefixlen 128 scopeid 0x10<host>
loop txqueuelen 1000 (Local Loopback)
RX packets 22495 bytes 2011453 (2.0 MB)
RX errors 0 dropped 0 overruns 0 frame 0
TX packets 22495 bytes 2011453 (2.0 MB)
TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

wlo1: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
inet 192.168.0.0 netmask 255.255.255.0 broadcast 192.168.0.255
inet6 fe80::5be8:7e0:57e5:fac3 prefixlen 64 scopeid 0x20<link>
ether e0:94:07:9d:b0ed txqueuelen 1000 (Ethernet)
RX packets 408235 bytes 37609639 (37.6 MB)
RX errors 0 dropped 0 overruns 0 frame 0
TX packets 240746 bytes 42269133 (42.2 MB)
TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

oalshi@oalshi:~/Documents/Sniffing_Spoofing/C_spoof$

```

(a) Attacker IP address



(b) Inputting data and verifying using tcpdump

Figure 2: correctness verification

2.4.2 Attack on Linux OS

- We run ifconfig and get the IP address.

```

Desktop
subangkar@HP: ~/Desktop/ping-spoofing
RX packets 0 bytes 0 (0.0 B)
RX errors 0 dropped 0 overruns 0 frame 0
TX packets 0 bytes 0 (0.0 B)
TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING> mtu 65536
inet 127.0.0.1 netmask 255.0.0.0
inet6 ::1 prefixlen 128 scopeid 0x10<host>
loop txqueuelen 1000 (Local Loopback)
RX packets 1472 bytes 143458 (143.4 KB)
RX errors 0 dropped 0 overruns 0 frame 0
TX packets 1472 bytes 143458 (143.4 KB)
TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

+ wlp2s0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
inet 172.20.57.31 netmask 255.255.248.0 broadcast 172.20.63.255
inet6 fe80::1e18:fb83:2a00:6fb5 prefixlen 64 scopeid 0x20<link>
ether 30:e3:7a:56:16:e3 txqueuelen 1000 (Ethernet)
RX packets 48996 bytes 21104631 (21.1 MB)
RX errors 0 dropped 0 overruns 0 frame 0
TX packets 17030 bytes 5753757 (5.7 MB)
TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

subangkar@HP:~/Desktop/ping-spoofing$

```

Figure 3: Spoofed IP address

```
shashata@shashata-X456UB: ~/Documents/Spoofing_Project

RX packets 0 bytes 0 (0.0 B)
RX errors 0 dropped 0 overruns 0 frame 0
TX packets 324 bytes 0 (0.0 B)
TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

vmnet8: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
inet 192.168.202.1 netmask 255.255.255.0 broadcast 192.168.202.255
inet6 fe80::250:56ff:fec0:8 prefixlen 64 scopeid 0x20<link>
ether 00:50:56:c0:00:08 txqueuelen 1000 (Ethernet)
RX packets 0 bytes 0 (0.0 B)
RX errors 0 dropped 0 overruns 0 frame 0
TX packets 324 bytes 0 (0.0 B)
TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

wlp3s0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
inet 172.20.57.28 netmask 255.255.248.0 broadcast 172.20.63.255
inet6 fe80::f575:6023:8e71:64eb prefixlen 64 scopeid 0x20<link>
ether 80:a5:89:33:3a:89 txqueuelen 1000 (Ethernet)
RX packets 256897 bytes 259727225 (259.7 MB)
RX errors 0 dropped 0 overruns 0 frame 0
TX packets 112373 bytes 19837624 (19.8 MB)
TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

shashata@shashata-X456UB:~/Documents/Spoofing_Project$
```

Figure 4: Destination IP address

```
oaishi@oaishi: ~/Desktop/sniffer

File Edit View Search Terminal Help

RX packets 0 bytes 0 (0.0 B)
RX errors 0 dropped 0 overruns 0 frame 0
TX packets 0 bytes 0 (0.0 B)
TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING> mtu 65536
inet 127.0.0.1 netmask 255.0.0.0
inet6 ::1 prefixlen 128 scopeid 0x10<host>
loop txqueuelen 1000 (Local Loopback)
RX packets 34736 bytes 3005211 (3.0 MB)
RX errors 0 dropped 0 overruns 0 frame 0
TX packets 34736 bytes 3005211 (3.0 MB)
TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

wlo1: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
inet 172.20.57.34 netmask 255.255.248.0 broadcast 172.20.63.255
inet6 fe80::ca8:5a0a:f9dd:691 prefixlen 64 scopeid 0x20<link>
ether e0:94:67:9d:b0:ed txqueuelen 1000 (Ethernet)
RX packets 564441 bytes 489353082 (489.3 MB)
RX errors 0 dropped 0 overruns 0 frame 0
TX packets 330784 bytes 57155303 (57.1 MB)
TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

oaishi@oaishi:~/Desktop/sniffer$
```

Figure 5: Attacked IP address

- We run tcpdump on the three devices for keeping track if our attack is working.

(a) Victim window

(b) Server window

(c) Attacker window

Figure 6: Initial Condition

- We feed the IP address in our front end.

(a) Front End Design

(b) Data input

Figure 7: Front End Configuration

- As seen from the tcpdump output, the attack has been successfully deployed.


```
oaishi@oaishi: ~/Documents/karmaker
File Edit View Search Terminal Help
oaishi@oaishi:~/Documents/karmaker$ sudo tcpdump -B 20000 -p -v -nn icmp
tcpdump: listening on wlo1, link-type EN10MB (Ethernet), capture size 262144 bytes
11:29:16.032890 IP (tos 0x0, ttl 20, id 34994, offset 0, flags [none], proto ICMP (1), length 28)
    172.20.57.31 > 172.20.57.28: ICMP echo request, id 0, seq 0, length 8
11:29:16.032934 IP (tos 0x0, ttl 20, id 34995, offset 0, flags [none], proto ICMP (1), length 28)
    172.20.57.31 > 172.20.57.28: ICMP echo request, id 0, seq 0, length 8
11:29:16.032951 IP (tos 0x0, ttl 20, id 34996, offset 0, flags [none], proto ICMP (1), length 28)
    172.20.57.31 > 172.20.57.28: ICMP echo request, id 0, seq 0, length 8
11:29:16.032966 IP (tos 0x0, ttl 20, id 34997, offset 0, flags [none], proto ICMP (1), length 28)
    172.20.57.31 > 172.20.57.28: ICMP echo request, id 0, seq 0, length 8
11:29:16.032984 IP (tos 0x0, ttl 20, id 34998, offset 0, flags [none], proto ICMP (1), length 28)
    172.20.57.31 > 172.20.57.28: ICMP echo request, id 0, seq 0, length 8
```

Figure 8: ICMP Request successfully send from IP "172.20.57.34", but sent as IP "172.20.57.31"

```
shashata@shashata-X456UB: ~/Documents/Spoofing_Project
tcpdump: listening on wlp3s0, link-type EN10MB (Ethernet), capture size 262144 bytes
11:29:16.071390 IP (tos 0x0, ttl 20, id 34994, offset 0, flags [none], proto ICMP (1), length 28)
    172.20.57.31 > 172.20.57.28: ICMP echo request, id 0, seq 0, length 8
11:29:16.071476 IP (tos 0x0, ttl 64, id 5268, offset 0, flags [none], proto ICMP (1), length 28)
    172.20.57.28 > 172.20.57.31: ICMP echo reply, id 0, seq 0, length 8
11:29:16.071547 IP (tos 0x0, ttl 20, id 34995, offset 0, flags [none], proto ICMP (1), length 28)
    172.20.57.31 > 172.20.57.28: ICMP echo request, id 0, seq 0, length 8
11:29:16.071566 IP (tos 0x0, ttl 64, id 5269, offset 0, flags [none], proto ICMP (1), length 28)
    172.20.57.28 > 172.20.57.31: ICMP echo reply, id 0, seq 0, length 8
11:29:16.075733 IP (tos 0x0, ttl 20, id 34996, offset 0, flags [none], proto ICMP (1), length 28)
    172.20.57.31 > 172.20.57.28: ICMP echo request, id 0, seq 0, length 8
11:29:16.075813 IP (tos 0x0, ttl 64, id 5270, offset 0, flags [none], proto ICMP (1), length 28)
    172.20.57.28 > 172.20.57.31: ICMP echo reply, id 0, seq 0, length 8
11:29:16.076511 IP (tos 0x0, ttl 20, id 34997, offset 0, flags [none], proto ICMP (1), length 28)
    172.20.57.31 > 172.20.57.28: ICMP echo request, id 0, seq 0, length 8
11:29:16.076564 IP (tos 0x0, ttl 64, id 5271, offset 0, flags [none], proto ICMP
```

Figure 9: ICMP Request coming as spoofed IP "172.27.52.31"

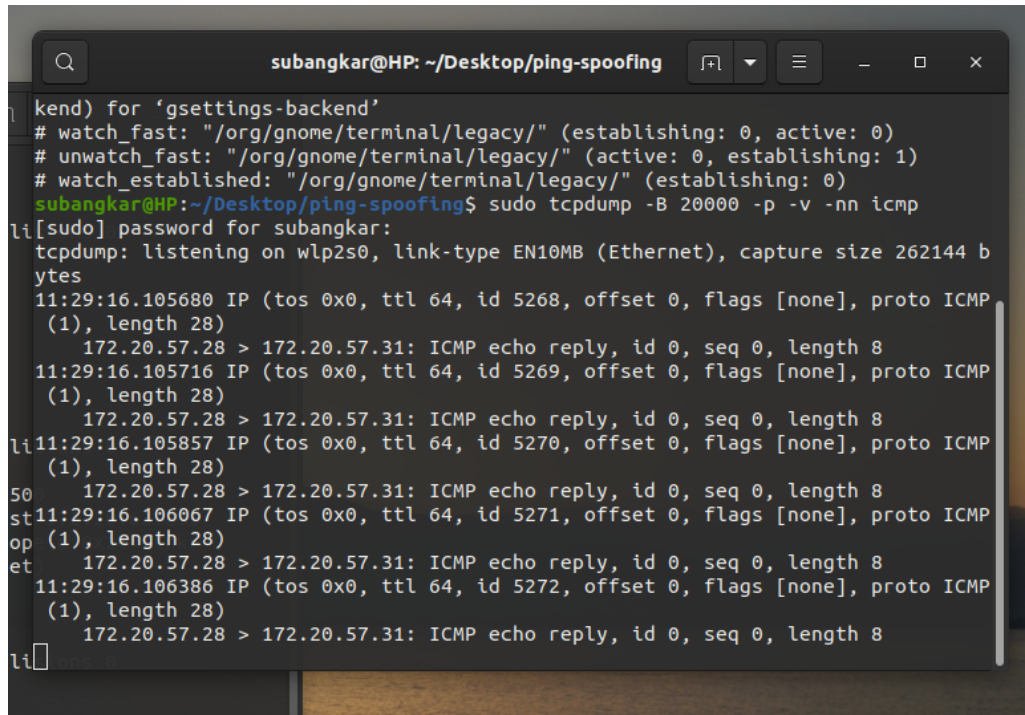


Figure 10: ICMP Reply successfully came at IP "127.27.52.31"

2.5 Was the attack successful and why

As seen from the screenshots, our designed attack tool worked both for android and ubuntu setup. The reason behind this is- We implemented the raw socket ourselves which allows us to set both the source and destination address in the ICMP header. Hence, we were successful to send a packet from the spoofed source IP to destination IP address.

2.6 Challenges

The main requirements of a successful attack are -

- At least three devices are needed
- There must be well enough Internet connection
- All of them must be connected to the same LAN

The main challenge was to ensure all these requirements. We faced issues running the tcpdump command from Termux. Moreover, we could not find any suitable packet analyzer for unrooted devices. Hence, we proved using tcpdump on our attacker PC that our tool works successfully on Smartphone too.

To ensure there is internet connection and all of the three devices were connected on the same LAN- was a challenge too.

3 Preventive Measure

This is practically impossible to design any preventive measure against ICMP ping spoofing when the devices are over LAN. But when the devices are connected over different LAN, the routers in between can take some preventive measures. As seen in our scenerio, we could successfully exploit our attack in the same LAN, but when the devices are connected

3.1 Future Work

- Making an integrated tool
- Designing preventive measure

4 Conclusion