CS305 Computer Network

Lab1_Report

SID: 11811806

Name: 熊卓晨

Describe

A2.1

Find Narcissistic Number

- filename: narcissistic_number.py
- requirement: implement a function to find all the narcissistic numbers in a range

A2.2

Use Wireshark to capture packets and answer the questions with your screenshots: 1. launch a http session between your host and “www.example.com” 1-1. what 's the filter used for the HTTP session between your host and “www.example.com”? 1-2. Find a HTTP response packet in this http session. what's the decimal and hexdecimal representation of the src ip addr, src port, dst ip addr and dst port of this packet. 2. launch a http session between your host and “www.baidu.com” 2-1. answer the question 1-2 based on the new http session between your host and “www.baidu.com” 2-2. list the items which value is same in the answers of both question 1-2 and 2-1.

A2.3

Using ICMPv4 to trace route between your computer(source) and “www.163.com”(destionation). Using a proper capture filter/display filter to capture/display this session. Answer the following questions with words and screenshots on both the execution result of command and capture result of wireshark: 1. How many 'time-to-live exceed' and 'echo reply' response messages are received ? What’s the source IP address of the 1st received 'time-to-live exceed' message, What’s the source IP address of the 1st received ‘echo reply' message? 2. Calculate the RTT (round-trip time) between your host and www.163.com based on the packets captured. Are they same with RTT from command execution result? 3. Add the value of hops(between source and destination) and TTL value of ICMPv4 messages received by source(which send ICMPv4 echo request). Is it the initial value of TTL from ICMPv4 message send by source or the ICMPv4 message send by destination? how to prove this conclusion

Assignment2.1

The file have been uploaded,the code is as follow

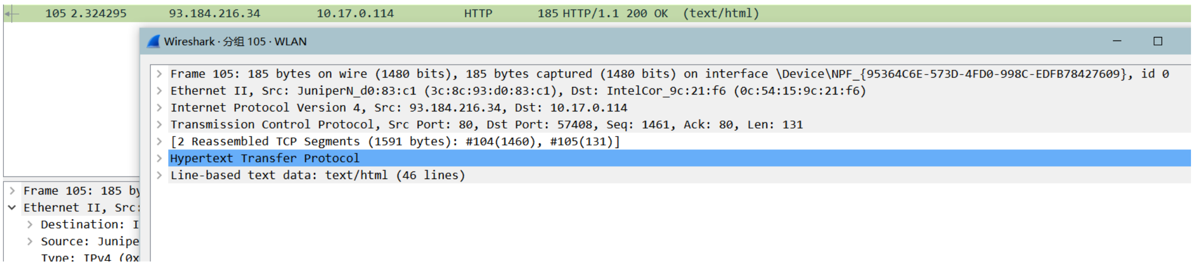
```
import math
def find_narcissistic_number(start:int,end:int)->list:
    ans_list = []
    for x in range(start,end+1):
        nums_list = []
        nums = 0
        tmp = x
        while tmp != 0:
            # print(str(nums) + "," +str(tmp))
            nums_list.append(tmp%10)
            tmp //= 10
            nums += 1
        sum = 0
        for i in range(0,nums):

            sum += int(pow(nums_list[i],nums))
        if sum == x:
            ans_list.append(x)
    return ans_list

if __name__ == '__main__':
    print(str(find_narcissistic_number(1,1000000)))
```

Assignment2.2

1-1: Use "**http and ip.addr==93.184.216.34**" in filter

1-2:

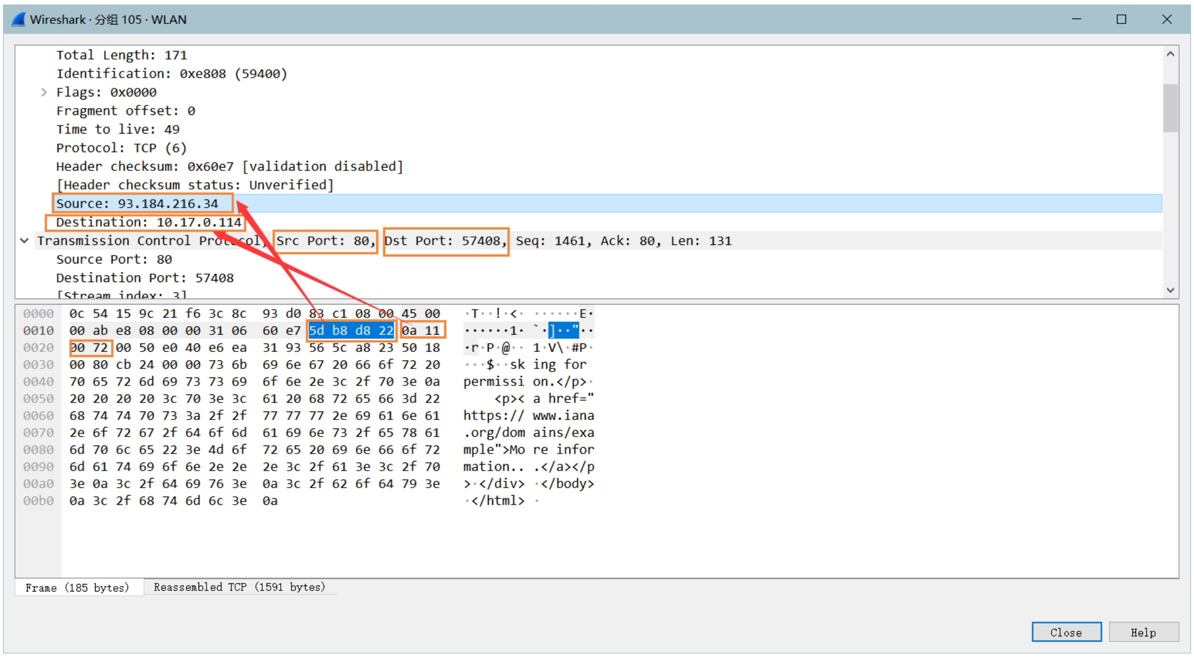
This is the HTTP response packet in this htp session.



```
105 2.324295    93.184.216.34    10.17.0.114    HTTP    185 HTTP/1.1 200 OK  (text/html)

> Frame 105: 185 bytes on wire (1480 bits), 185 bytes captured (1480 bits) on interface \Device\NPF_{95364C6E-573D-4FD0-998C-EDFB78427609}, id 0
> Ethernet II, Src: JuniperN_d0:83:c1 (3c:8c:93:d0:83:c1), Dst: IntelCor_9c:21:f6 (0c:54:15:9c:21:f6)
> Internet Protocol Version 4, Src: 93.184.216.34, Dst: 10.17.0.114
> Transmission Control Protocol, Src Port: 80, Dst Port: 57408, Seq: 1461, Ack: 80, Len: 131
> [2 Reassembled TCP Segments (1591 bytes): #104(1460), #105(131)]
> Hypertext Transfer Protocol
> Line-based text data: text/html (46 lines)

> Frame 105: 185 by
v Ethernet II, Src:
  > Destination: I
  > Source: Junipe
    Type: IPv4 (0x
```

Fig.1

Here are the details:



```
    Total Length: 171
    Identification: 0xe808 (59400)
  > Flags: 0x0000
    Fragment offset: 0
    Time to live: 49
    Protocol: TCP (6)
    Header checksum: 0x60e7 [validation disabled]
    [Header checksum status: Unverified]
    Source: 93.184.216.34
    Destination: 10.17.0.114
v Transmission Control Protocol, Src Port: 80, Dst Port: 57408, Seq: 1461, Ack: 80, Len: 131
    Source Port: 80
    Destination Port: 57408
    [Stream index: 3]

0000  0c 54 15 9c 21 f6 3c 8c  93 d0 83 c1 08 00 45 00   ·T··!·<· ······E·
0010  00 ab e8 08 00 00 31 06  60 e7 5d b8 d8 22 0a 11   ······1· `·]··"··
0020  00 72 00 50 e0 40 e6 ea  31 93 56 5c a8 23 50 18   ·r·P·@·· 1·V\·#P·
0030  00 80 cb 24 00 00 73 6b  69 6e 67 20 66 6f 72 20   ···$··sk ing for 
0040  70 65 72 6d 69 73 73 69  6f 6e 2e 3c 2f 70 3e 0a   permissi on.</p>·
0050  20 20 20 20 3c 70 3e 3c  61 20 68 72 65 66 3d 22       <p>< a href="
0060  68 74 74 70 73 3a 2f 2f  77 77 77 2e 69 61 6e 61   https:// www.iana
0070  2e 6f 72 67 2f 64 6f 6d  61 69 6e 73 2f 65 78 61   .org/dom ains/exa
0080  6d 70 6c 65 22 3e 4d 6f  72 65 20 69 6e 66 6f 72   mple">Mo re infor
0090  6d 61 74 69 6f 6e 2e 2e  2e 3c 2f 61 3e 3c 2f 70   mation.. .</a></p
00a0  3e 0a 3c 2f 64 69 76 3e  0a 3c 2f 62 6f 64 79 3e   >·</div> ·</body>
00b0  0a 3c 2f 68 74 6d 6c 3e  0a                        ·</html> ·

Frame (185 bytes)   Reassembled TCP (1591 bytes)
```

Fig.2

Decimal

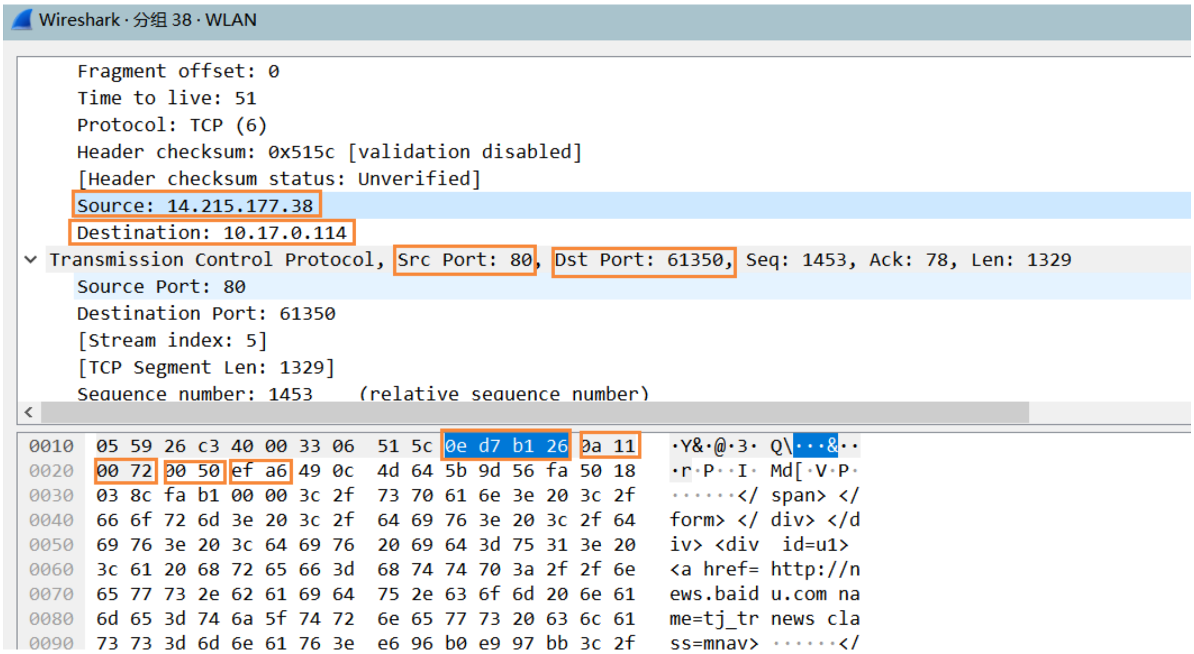
- src ip addr: 93.184.216.34
- src port: 80

- dst ip addr: 10.17.0.114
- dst port:57408

Hexdecimal

- src ip addr: 5d.b8.d0.22
- src port: 00 50
- dst ip addr: 0a.11.00.72
- dst port: e0 40

2-1



Wireshark · 分组 38 · WLAN

```
    Fragment offset: 0
    Time to live: 51
    Protocol: TCP (6)
    Header checksum: 0x515c [validation disabled]
    [Header checksum status: Unverified]
    Source: 14.215.177.38
    Destination: 10.17.0.114
v Transmission Control Protocol, Src Port: 80, Dst Port: 61350, Seq: 1453, Ack: 78, Len: 1329
    Source Port: 80
    Destination Port: 61350
    [Stream index: 5]
    [TCP Segment Len: 1329]
    Sequence number: 1453    (relative sequence number)
```

```
0010  05 59 26 c3 40 00 33 06  51 5c 0e d7 b1 26 0a 11   ·Y&·@·3· Q\···&··
0020  00 72 00 50 ef a6 49 0c  4d 64 5b 9d 56 fa 50 18   ·r·P··I· Md[·V·P·
0030  03 8c fa b1 00 00 3c 2f  73 70 61 6e 3e 20 3c 2f   ······</ span> </
0040  66 6f 72 6d 3e 20 3c 2f  64 69 76 3e 20 3c 2f 64   form> </ div> </d
0050  69 76 3e 20 3c 64 69 76  20 69 64 3d 75 31 3e 20   iv> <div  id=u1>
0060  3c 61 20 68 72 65 66 3d  68 74 74 70 3a 2f 2f 6e   <a href= http://n
0070  65 77 73 2e 62 61 69 64  75 2e 63 6f 6d 20 6e 61   ews.baid u.com na
0080  6d 65 3d 74 6a 5f 74 72  6e 65 77 73 20 63 6c 61   me=tj_tr news cla
0090  73 73 3d 6d 6e 61 76 3e  e6 96 b0 e9 97 bb 3c 2f   ss=mnav> ······</
```

Fig.3

Decimal

- src ip addr: 14.215.177.38
- src port: 80
- dst ip addr: 10.17.0.114
- dst port:57408

Hexdecimal

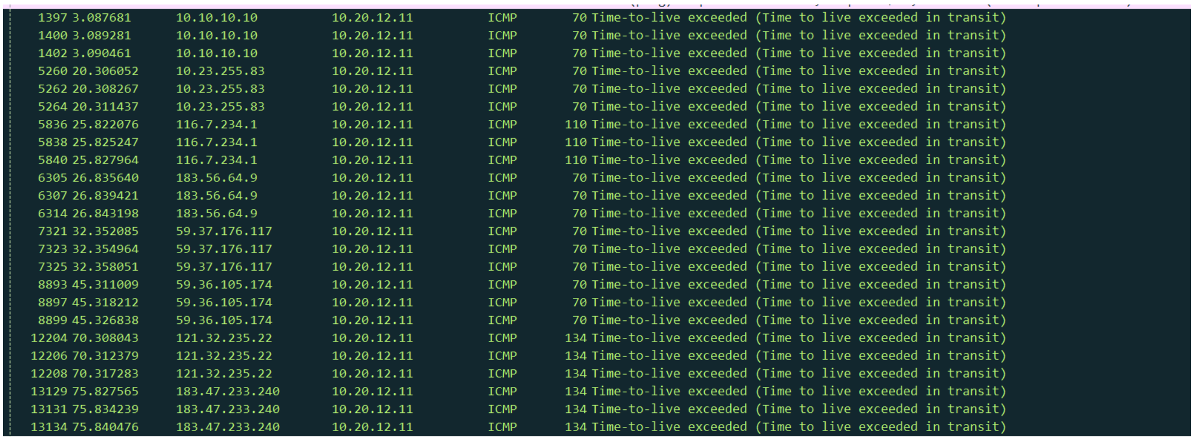
- src ip addr: 0e.d7.b1.26
- src port: 00 50
- dst ip addr: 0a.11.00.72
- dst port: ef a6

2-2

src port and **dst ip addr**

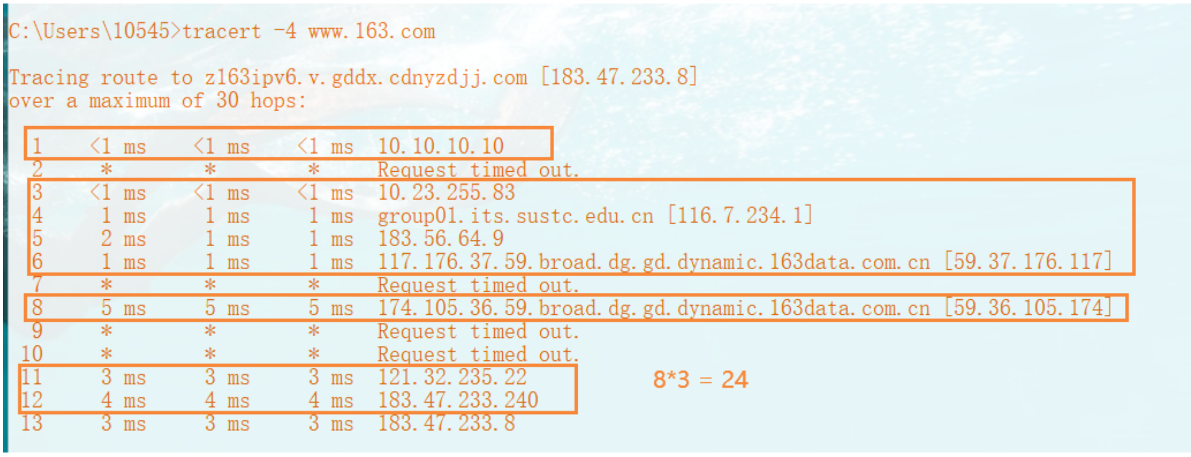
Assignment2.3

3-1



```
 1397 3.087681    10.10.10.10      10.20.12.11    ICMP     70 Time-to-live exceeded (Time to live exceeded in transit)
 1400 3.089281    10.10.10.10      10.20.12.11    ICMP     70 Time-to-live exceeded (Time to live exceeded in transit)
 1402 3.090461    10.10.10.10      10.20.12.11    ICMP     70 Time-to-live exceeded (Time to live exceeded in transit)
 5260 20.306052   10.23.255.83     10.20.12.11    ICMP     70 Time-to-live exceeded (Time to live exceeded in transit)
 5262 20.308267   10.23.255.83     10.20.12.11    ICMP     70 Time-to-live exceeded (Time to live exceeded in transit)
 5264 20.311437   10.23.255.83     10.20.12.11    ICMP     70 Time-to-live exceeded (Time to live exceeded in transit)
 5836 25.822076   116.7.234.1      10.20.12.11    ICMP    110 Time-to-live exceeded (Time to live exceeded in transit)
 5838 25.825247   116.7.234.1      10.20.12.11    ICMP    110 Time-to-live exceeded (Time to live exceeded in transit)
 5840 25.827964   116.7.234.1      10.20.12.11    ICMP    110 Time-to-live exceeded (Time to live exceeded in transit)
 6305 26.835640   183.56.64.9      10.20.12.11    ICMP     70 Time-to-live exceeded (Time to live exceeded in transit)
 6307 26.839421   183.56.64.9      10.20.12.11    ICMP     70 Time-to-live exceeded (Time to live exceeded in transit)
 6314 26.843198   183.56.64.9      10.20.12.11    ICMP     70 Time-to-live exceeded (Time to live exceeded in transit)
 7321 32.352085   59.37.176.117    10.20.12.11    ICMP     70 Time-to-live exceeded (Time to live exceeded in transit)
 7323 32.354964   59.37.176.117    10.20.12.11    ICMP     70 Time-to-live exceeded (Time to live exceeded in transit)
 7325 32.358051   59.37.176.117    10.20.12.11    ICMP     70 Time-to-live exceeded (Time to live exceeded in transit)
 8893 45.311009   59.36.105.174    10.20.12.11    ICMP     70 Time-to-live exceeded (Time to live exceeded in transit)
 8897 45.318212   59.36.105.174    10.20.12.11    ICMP     70 Time-to-live exceeded (Time to live exceeded in transit)
 8899 45.326838   59.36.105.174    10.20.12.11    ICMP     70 Time-to-live exceeded (Time to live exceeded in transit)
12204 70.308043   121.32.235.22    10.20.12.11    ICMP    134 Time-to-live exceeded (Time to live exceeded in transit)
12206 70.312379   121.32.235.22    10.20.12.11    ICMP    134 Time-to-live exceeded (Time to live exceeded in transit)
12208 70.317283   121.32.235.22    10.20.12.11    ICMP    134 Time-to-live exceeded (Time to live exceeded in transit)
13129 75.827565   183.47.233.240   10.20.12.11    ICMP    134 Time-to-live exceeded (Time to live exceeded in transit)
13131 75.834239   183.47.233.240   10.20.12.11    ICMP    134 Time-to-live exceeded (Time to live exceeded in transit)
13134 75.840476   183.47.233.240   10.20.12.11    ICMP    134 Time-to-live exceeded (Time to live exceeded in transit)
```

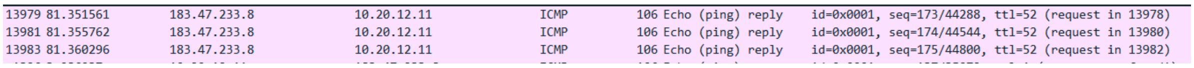
Fig.1



```
C:\Users\10545>tracert -4 www.163.com

Tracing route to z163ipv6.v.gddx.cdnyzdjj.com [183.47.233.8]
over a maximum of 30 hops:

  1    <1 ms    <1 ms    <1 ms  10.10.10.10
  2     *        *        *     Request timed out.
  3    <1 ms    <1 ms    <1 ms  10.23.255.83
  4     1 ms     1 ms     1 ms  group01.its.sustc.edu.cn [116.7.234.1]
  5     2 ms     1 ms     1 ms  183.56.64.9
  6     1 ms     1 ms     1 ms  117.176.37.59.broad.dg.gd.dynamic.163data.com.cn [59.37.176.117]
  7     *        *        *     Request timed out.
  8     5 ms     5 ms     5 ms  174.105.36.59.broad.dg.gd.dynamic.163data.com.cn [59.36.105.174]
  9     *        *        *     Request timed out.
 10     *        *        *     Request timed out.
 11     3 ms     3 ms     3 ms  121.32.235.22
 12     4 ms     4 ms     4 ms  183.47.233.240
 13     3 ms     3 ms     3 ms  183.47.233.8
```

Fig.2

There are 24 TTL exceed response messages are received.



```
13979 81.351561   183.47.233.8   10.20.12.11   ICMP   106 Echo (ping) reply   id=0x0001, seq=173/44288, ttl=52 (request in 13978)
13981 81.355762   183.47.233.8   10.20.12.11   ICMP   106 Echo (ping) reply   id=0x0001, seq=174/44544, ttl=52 (request in 13980)
13983 81.360296   183.47.233.8   10.20.12.11   ICMP   106 Echo (ping) reply   id=0x0001, seq=175/44800, ttl=52 (request in 13982)
```

Fig.3

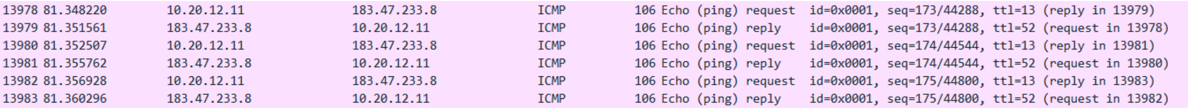
There are 3 echo reply reponse messages.

The source IP address of the 1st received 'time-to-live exceed' message is **10.10.10.10**

The source IP address of the 1st received ‘echo reply' message is **183.47.233.8**

3-2

In **Command Line,** According to Fig.2 we have the RTT is **3ms.**



```
13978 81.348220   10.20.12.11    183.47.233.8   ICMP   106 Echo (ping) request  id=0x0001, seq=173/44288, ttl=13 (reply in 13979)
13979 81.351561   183.47.233.8   10.20.12.11    ICMP   106 Echo (ping) reply    id=0x0001, seq=173/44288, ttl=52 (request in 13978)
13980 81.352507   10.20.12.11    183.47.233.8   ICMP   106 Echo (ping) request  id=0x0001, seq=174/44544, ttl=13 (reply in 13981)
13981 81.355762   183.47.233.8   10.20.12.11    ICMP   106 Echo (ping) reply    id=0x0001, seq=174/44544, ttl=52 (request in 13980)
13982 81.356928   10.20.12.11    183.47.233.8   ICMP   106 Echo (ping) request  id=0x0001, seq=175/44800, ttl=13 (reply in 13983)
13983 81.360296   183.47.233.8   10.20.12.11    ICMP   106 Echo (ping) reply    id=0x0001, seq=175/44800, ttl=52 (request in 13982)
```

Fig.4

In **wireshark,** RTT = 81.351561-81.348220 = 0.003341 s $\approx$ **3ms**.

3-3

According to Fig.2, the value of hops is 13. TTL value of ICMPv4 message received by source is 52 and the sum is **52+13=65$\neq$ 13**

According to ICMP principle, during a tracert, source (taking my computer as an example) will send several packets to the near server/router, whose TTL is increasing by 1 for each hop. So when the destination server send a echo reply ICMP packet, each middle server/router will reduce 1 TTL. That means the initial TTL equals to the received packet's TTL plus hops.

Summary

In this Lab, we lean abouthow to use python and use wireshark to capture **http** and **ICMP** packets.