Detection of DOS Attack Using Wireshark

Ronak C Desai – 16382dr

Overview:-

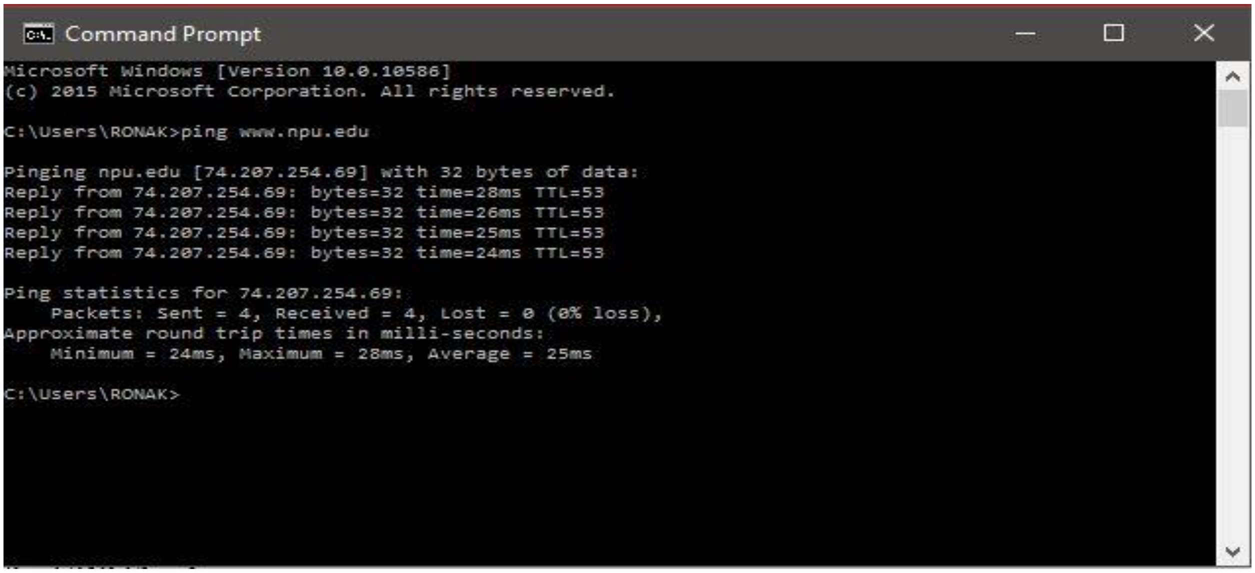
Denial of service attack commonly knowns as DOS attack is an attack in which the attacker indents to bring down a service or make resources unavailable for other users by flooding the network/server of the host or the service provider with bogus requests. Here is an attempt to detect the attack using Wireshark- a network protocol analyzer.

Body:-

There are two type of DOS attacks, attack from one attacker and attack from multiple attacker. Also there are different types of request attack.

1. Ping Flood (ICMP Flood)
2. UDP Flood
3. SYN Flood
4. Nuke
5. Peer-To-Peer attack
6. Reflected Attack etc.

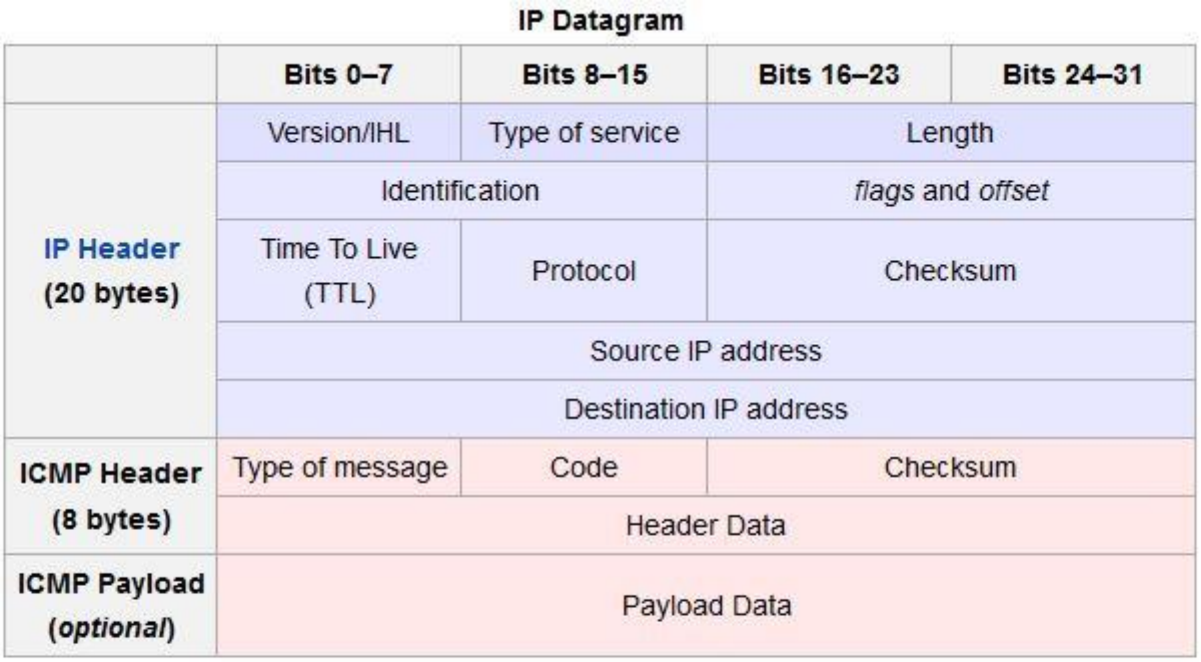
I am going to focus on Ping attack/ICMP flood/ Smurf attack. Ping command is usually used for the host discovery and IP level connectivity. Figure below show the normal working of ping command in command line under Windows operating system.



```
Microsoft Windows [Version 10.0.10586]
(c) 2015 Microsoft Corporation. All rights reserved.

C:\Users\RONAK>ping www.npu.edu

Pinging npu.edu [74.207.254.69] with 32 bytes of data:
Reply from 74.207.254.69: bytes=32 time=28ms TTL=53
Reply from 74.207.254.69: bytes=32 time=26ms TTL=53
Reply from 74.207.254.69: bytes=32 time=25ms TTL=53
Reply from 74.207.254.69: bytes=32 time=24ms TTL=53

Ping statistics for 74.207.254.69:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
    Minimum = 24ms, Maximum = 28ms, Average = 25ms

C:\Users\RONAK>
```

Working of PING and how it can cause security consideration:-

The ping command when fired send ICMP (Internet Control Message Protocol) packets to the targeted host. The common composition of ICMP packet is shown below



IP Datagram

	Bits 0–7	Bits 8–15	Bits 16–23	Bits 24–31
IP Header (20 bytes)	Version/IHL	Type of service	Length	
	Identification		*flags* and *offset*	
	Time To Live (TTL)	Protocol	Checksum	
	Source IP address			
	Destination IP address			
ICMP Header (8 bytes)	Type of message	Code	Checksum	
	Header Data			
ICMP Payload (*optional*)	Payload Data			

Now how to take down a service using this command below is the mostly script to bring down a service of the targeted host.

ping –n 'value' –I 'value' www.npu.edu

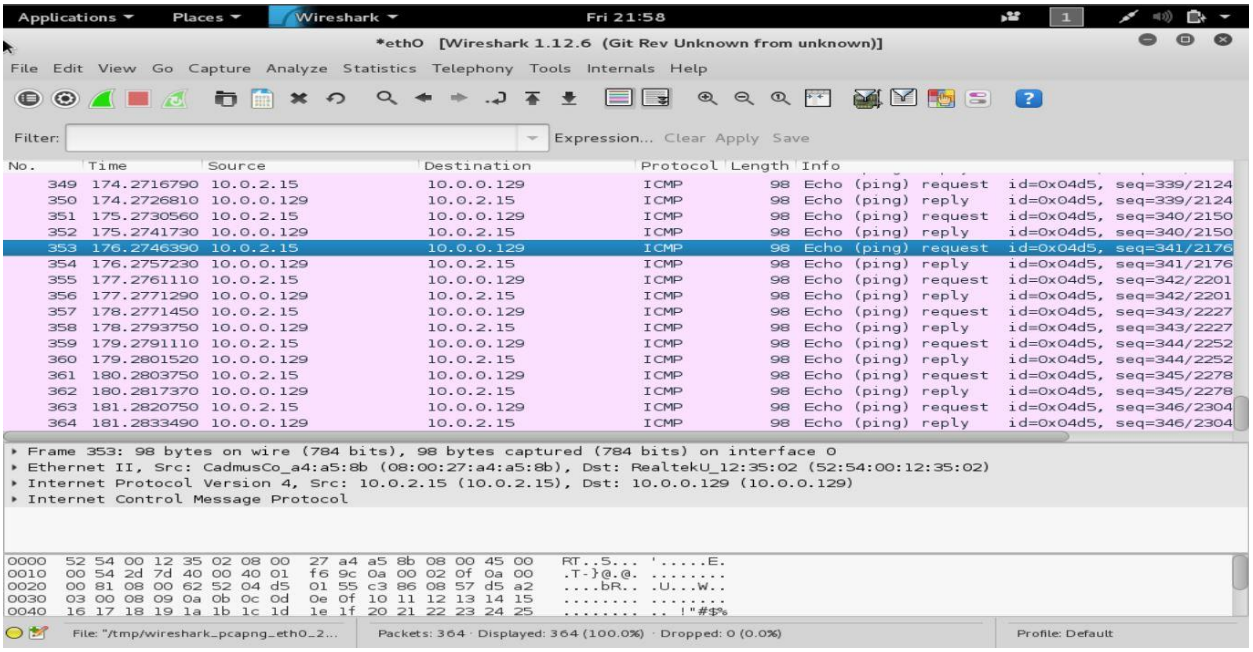
Here the attacker will use the maximum value of I and N.

Now as the targeted host begins to get the request from the attacker, RFC 1122 mandated the host to reply to all the request which are made by the user. This creates a security loophole in the system.

Now the attacker uses the maximum value of I and N and starts sending packets to targeted system and the system in return start giving replies to the echo request with an echo response.

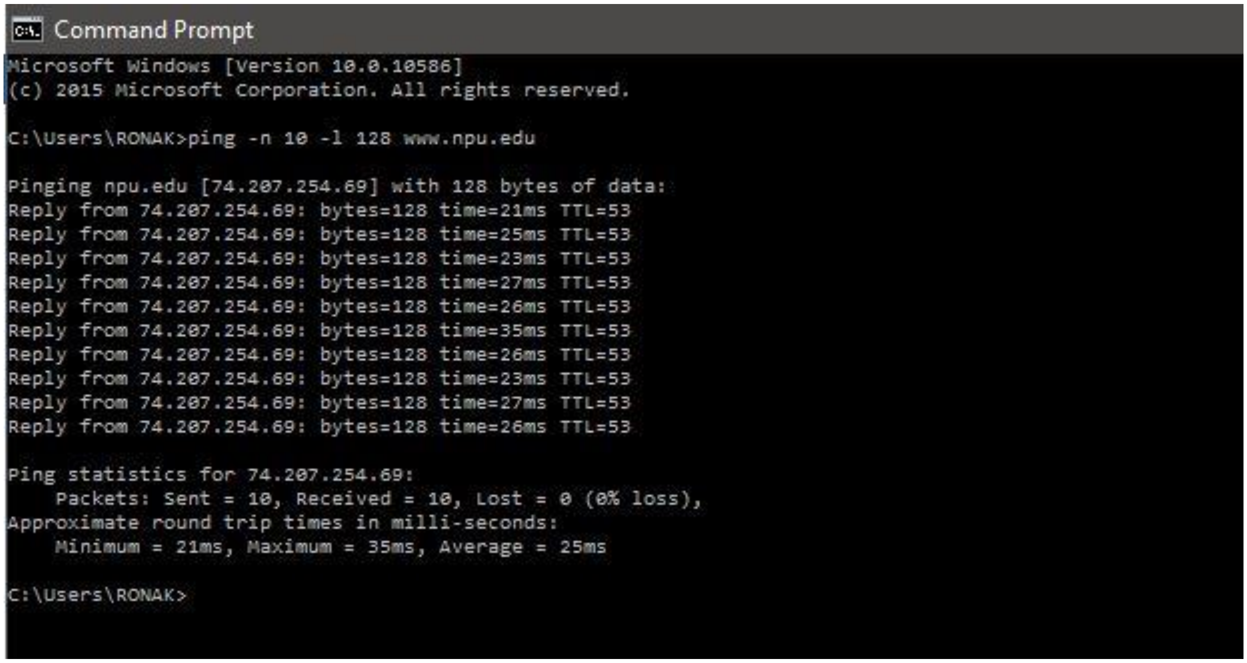
Now to defend the network and resources the targeted host- Wireshark at your rescue, needs to detect the attack first and then prevent it. To detect the packets Wireshark which is a network protocol analyzer it will capture all the packets incoming as well as outgoing of the network.

The incoming packets will be ICMP echo request packets and the outgoing packets will be echo response (for the targeted host) which is shown in the figure below.


Applications
Places
Wireshark
Fri 21:58
1
*eth0 [Wireshark 1.12.6 (Git Rev Unknown from unknown)]
File Edit View Go Capture Analyze Statistics Telephony Tools Internals Help
Filter:
Expression... Clear Apply Save

```
No.   Time         Source        Destination   Protocol Length Info
  349 174.2716790  10.0.2.15     10.0.0.129    ICMP         98 Echo (ping) request  id=0x04d5, seq=339/2124
  350 174.2726810  10.0.0.129    10.0.2.15     ICMP         98 Echo (ping) reply    id=0x04d5, seq=339/2124
  351 175.2730560  10.0.2.15     10.0.0.129    ICMP         98 Echo (ping) request  id=0x04d5, seq=340/2150
  352 175.2741730  10.0.0.129    10.0.2.15     ICMP         98 Echo (ping) reply    id=0x04d5, seq=340/2150
  353 176.2746390  10.0.2.15     10.0.0.129    ICMP         98 Echo (ping) request  id=0x04d5, seq=341/2176
  354 176.2757230  10.0.0.129    10.0.2.15     ICMP         98 Echo (ping) reply    id=0x04d5, seq=341/2176
  355 177.2761110  10.0.2.15     10.0.0.129    ICMP         98 Echo (ping) request  id=0x04d5, seq=342/2201
  356 177.2771290  10.0.0.129    10.0.2.15     ICMP         98 Echo (ping) reply    id=0x04d5, seq=342/2201
  357 178.2771450  10.0.2.15     10.0.0.129    ICMP         98 Echo (ping) request  id=0x04d5, seq=343/2227
  358 178.2793750  10.0.0.129    10.0.2.15     ICMP         98 Echo (ping) reply    id=0x04d5, seq=343/2227
  359 179.2791110  10.0.2.15     10.0.0.129    ICMP         98 Echo (ping) request  id=0x04d5, seq=344/2252
  360 179.2801520  10.0.0.129    10.0.2.15     ICMP         98 Echo (ping) reply    id=0x04d5, seq=344/2252
  361 180.2803750  10.0.2.15     10.0.0.129    ICMP         98 Echo (ping) request  id=0x04d5, seq=345/2278
  362 180.2817370  10.0.0.129    10.0.2.15     ICMP         98 Echo (ping) reply    id=0x04d5, seq=345/2278
  363 181.2820750  10.0.2.15     10.0.0.129    ICMP         98 Echo (ping) request  id=0x04d5, seq=346/2304
  364 181.2833490  10.0.0.129    10.0.2.15     ICMP         98 Echo (ping) reply    id=0x04d5, seq=346/2304

▸ Frame 353: 98 bytes on wire (784 bits), 98 bytes captured (784 bits) on interface 0
▸ Ethernet II, Src: CadmusCo_a4:a5:8b (08:00:27:a4:a5:8b), Dst: RealtekU_12:35:02 (52:54:00:12:35:02)
▸ Internet Protocol Version 4, Src: 10.0.2.15 (10.0.2.15), Dst: 10.0.0.129 (10.0.0.129)
▸ Internet Control Message Protocol

0000  52 54 00 12 35 02 08 00  27 a4 a5 8b 08 00 45 00   RT..5... '.....E.
0010  00 54 2d 7d 40 00 40 01  f6 9c 0a 00 02 0f 0a 00   .T-}@.@. ........
0020  00 81 08 00 62 52 04 d5  01 55 c3 86 08 57 d5 a2   ....bR.. .U...W..
0030  03 00 08 09 0a 0b 0c 0d  0e 0f 10 11 12 13 14 15   ........ ........
0040  16 17 18 19 1a 1b 1c 1d  1e 1f 20 21 22 23 24 25   ........ .. !"#$%

File: "/tmp/wireshark_pcapng_eth0_2...   Packets: 364 · Displayed: 364 (100.0%) · Dropped: 0 (0.0%)   Profile: Default
```

Below is the figure in which I’ve pinged www.npu.edu with 100 packets of size 128 (these are not the maximum values) to show the modified ping command and an attacker can use the same command to ping flood a network/resources with some huge values.

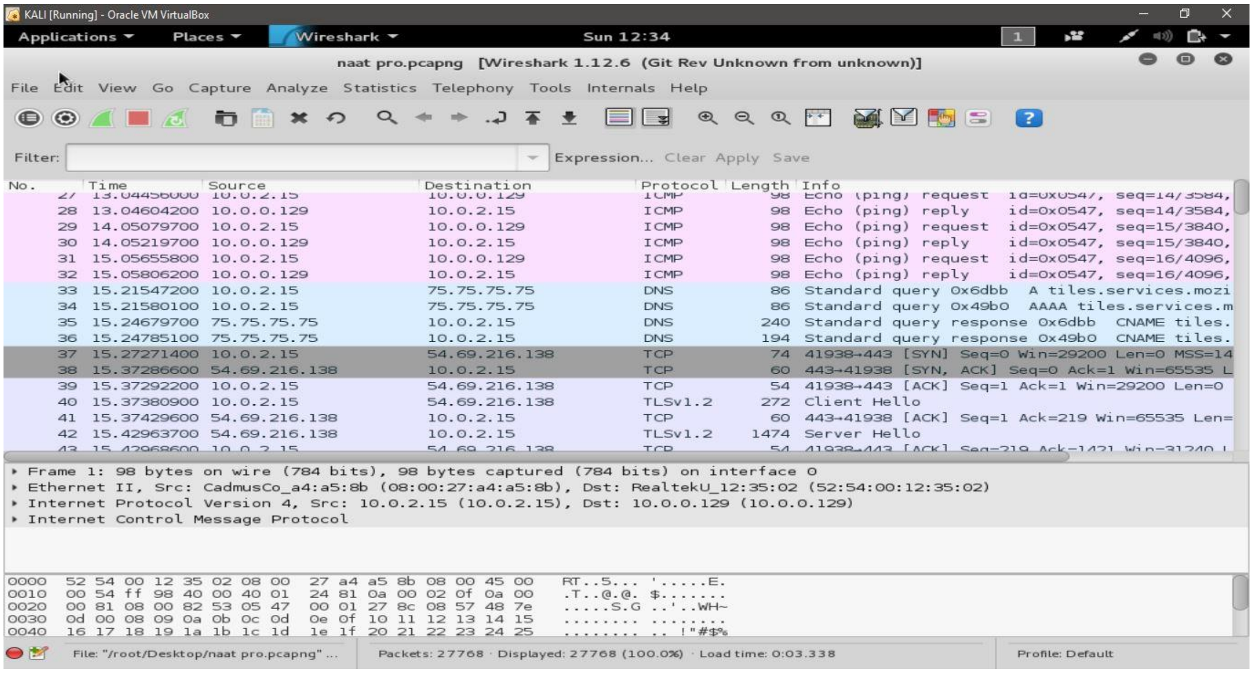


```
Microsoft Windows [Version 10.0.10586]
(c) 2015 Microsoft Corporation. All rights reserved.

C:\Users\RONAK>ping -n 10 -l 128 www.npu.edu

Pinging npu.edu [74.207.254.69] with 128 bytes of data:
Reply from 74.207.254.69: bytes=128 time=21ms TTL=53
Reply from 74.207.254.69: bytes=128 time=25ms TTL=53
Reply from 74.207.254.69: bytes=128 time=23ms TTL=53
Reply from 74.207.254.69: bytes=128 time=27ms TTL=53
Reply from 74.207.254.69: bytes=128 time=26ms TTL=53
Reply from 74.207.254.69: bytes=128 time=35ms TTL=53
Reply from 74.207.254.69: bytes=128 time=26ms TTL=53
Reply from 74.207.254.69: bytes=128 time=23ms TTL=53
Reply from 74.207.254.69: bytes=128 time=27ms TTL=53
Reply from 74.207.254.69: bytes=128 time=26ms TTL=53

Ping statistics for 74.207.254.69:
    Packets: Sent = 10, Received = 10, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
    Minimum = 21ms, Maximum = 35ms, Average = 25ms

C:\Users\RONAK>
```

The default packet size is 32 bytes but with altering the parameters I’ve made the packet size 128 bytes and number of packets from 4 to 10.

Now how to detect such bogus packets incoming into the network when these packet are amid the normal traffic.

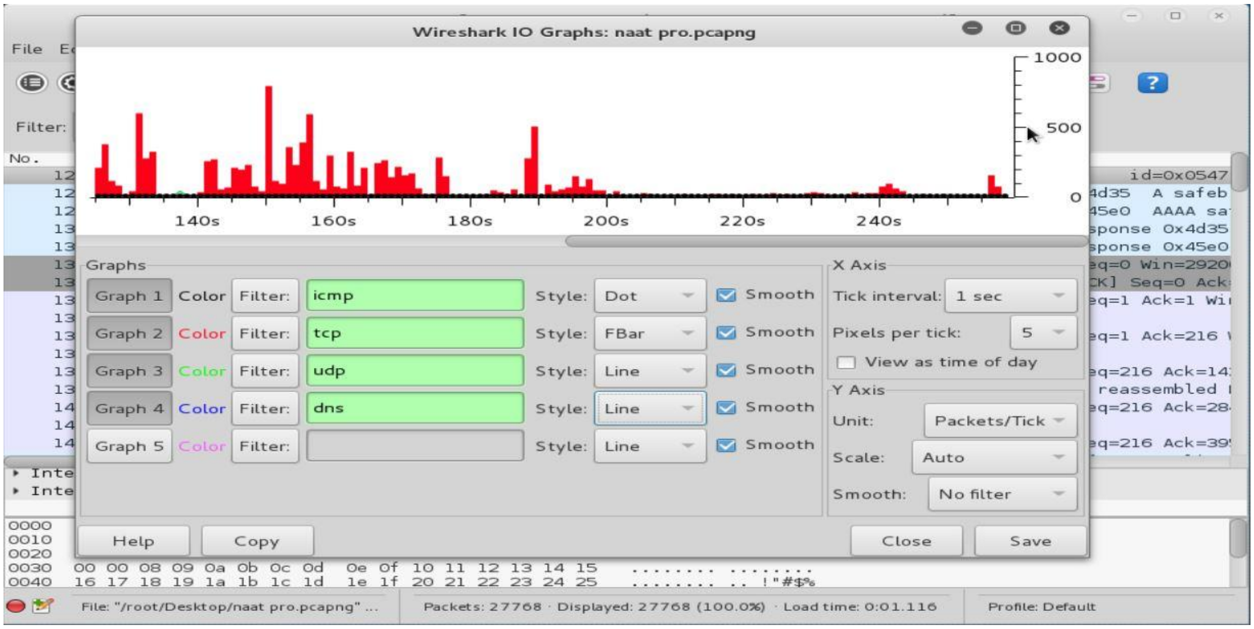
ICMP packets amid the normal traffic would look something like these, as shown in the figure.


KALI [Running] - Oracle VM VirtualBox
Applications
Places
Wireshark
Sun 12:34
naat pro.pcapng [Wireshark 1.12.6 (Git Rev Unknown from unknown)]
File Edit View Go Capture Analyze Statistics Telephony Tools Internals Help
Filter:
Expression... Clear Apply Save

```
No.   Time         Source          Destination     Protocol Length Info
  27 13.04456000 10.0.2.15        10.0.0.129        ICMP        98 Echo (ping) request  id=0x0547, seq=14/3584,
  28 13.04604200 10.0.0.129       10.0.2.15         ICMP        98 Echo (ping) reply    id=0x0547, seq=14/3584,
  29 14.05079700 10.0.2.15        10.0.0.129        ICMP        98 Echo (ping) request  id=0x0547, seq=15/3840,
  30 14.05219700 10.0.0.129       10.0.2.15         ICMP        98 Echo (ping) reply    id=0x0547, seq=15/3840,
  31 15.05655800 10.0.2.15        10.0.0.129        ICMP        98 Echo (ping) request  id=0x0547, seq=16/4096,
  32 15.05806200 10.0.0.129       10.0.2.15         ICMP        98 Echo (ping) reply    id=0x0547, seq=16/4096,
  33 15.21547200 10.0.2.15        75.75.75.75       DNS         86 Standard query 0x6dbb  A tiles.services.mozi
  34 15.21580100 10.0.2.15        75.75.75.75       DNS         86 Standard query 0x49b0  AAAA tiles.services.m
  35 15.24679700 75.75.75.75      10.0.2.15         DNS        240 Standard query response 0x6dbb  CNAME tiles.
  36 15.24785100 75.75.75.75      10.0.2.15         DNS        194 Standard query response 0x49b0  CNAME tiles.
  37 15.27271400 10.0.2.15        54.69.216.138     TCP         74 41938→443 [SYN] Seq=0 Win=29200 Len=0 MSS=14
  38 15.37286600 54.69.216.138    10.0.2.15         TCP         60 443→41938 [SYN, ACK] Seq=0 Ack=1 Win=65535 L
  39 15.37292200 10.0.2.15        54.69.216.138     TCP         54 41938→443 [ACK] Seq=1 Ack=1 Win=29200 Len=0
  40 15.37380900 10.0.2.15        54.69.216.138     TLSv1.2    272 Client Hello
  41 15.37429600 54.69.216.138    10.0.2.15         TCP         60 443→41938 [ACK] Seq=1 Ack=219 Win=65535 Len=
  42 15.42963700 54.69.216.138    10.0.2.15         TLSv1.2   1474 Server Hello
  43 15.42968600 10.0.2.15        54.69.216.138     TCP         54 41938→443 [ACK] Seq=219 Ack=1421 Win=31240 L

▸ Frame 1: 98 bytes on wire (784 bits), 98 bytes captured (784 bits) on interface 0
▸ Ethernet II, Src: CadmusCo_a4:a5:8b (08:00:27:a4:a5:8b), Dst: RealtekU_12:35:02 (52:54:00:12:35:02)
▸ Internet Protocol Version 4, Src: 10.0.2.15 (10.0.2.15), Dst: 10.0.0.129 (10.0.0.129)
▸ Internet Control Message Protocol

0000  52 54 00 12 35 02 08 00  27 a4 a5 8b 08 00 45 00   RT..5... '.....E.
0010  00 54 ff 98 40 00 40 01  24 81 0a 00 02 0f 0a 00   .T..@.@. $.......
0020  00 81 08 00 82 53 05 47  00 01 27 8c 08 57 48 7e   .....S.G ..'..WH~
0030  0d 00 08 09 0a 0b 0c 0d  0e 0f 10 11 12 13 14 15   ........ ........
0040  16 17 18 19 1a 1b 1c 1d  1e 1f 20 21 22 23 24 25   ........ .. !"#$%

File: "/root/Desktop/naat pro.pcapng" ...   Packets: 27768 · Displayed: 27768 (100.0%) · Load time: 0:03.338   Profile: Default
```

These ICMP packet would be most reoccurring incoming and outgoing packets.

Figure below shows the IO Graph which shows the continues ping request and replies.



Graphs

Graph 1	Color	Filter: icmp	Style: Dot	☑ Smooth
Graph 2	Color	Filter: tcp	Style: FBar	☑ Smooth
Graph 3	Color	Filter: udp	Style: Line	☑ Smooth
Graph 4	Color	Filter: dns	Style: Line	☑ Smooth
Graph 5	Color	Filter:	Style: Line	☑ Smooth

X Axis

Tick interval: 1 sec

Pixels per tick: 5

□ View as time of day

Y Axis

Unit: Packets/Tick

Scale: Auto

Smooth: No filter

```
0030  00 00 08 09 0a 0b 0c 0d  0e 0f 10 11 12 13 14 15   ........ ........
0040  16 17 18 19 1a 1b 1c 1d  1e 1f 20 21 22 23 24 25   ........ .. !"#$%

File: "/root/Desktop/naat pro.pcapng" ...   Packets: 27768 · Displayed: 27768 (100.0%) · Load time: 0:01.116   Profile: Default
```

Key points for catching the bogus ICMP packets

- Frequent occurrence of ICMP packets.
- Change of default size (32 bytes) of the packets.
- Request and reply to and from the same IP address.
- Continuous activity in IO Graph of Wireshark.