

Andre Hei Wang Law

4017 5600

COEN 366 – FL-X

Mininet Assignment 1 + Wireshark Assignment 2

1. Mininet Assignment 1

Question 1: Give a brief explanation about each topology mentioned above. (In your own words)

Single: A single topology is a basic topology with N hosts connected to one switch.

Reversed: This topology is similar to the single topology, since all N hosts are connected to only one switch. However, the difference lies in the reverse order of connections between the N hosts and the switches.

Linear: A linear topology have N switches and N hosts connected in a linear fashion forming a sequence with each switch connected to the one in the previous line.

Tree: A tree topology, as its name suggest, is a multilevel hierarchical tree where there is a starting node that branches off with each new generation child resulting in a new level, this a N level topology. In addition, this topology has two hosts per switch (two child per node).

Question 2: Using the CLI, generate the above topologies with the following nodes:

Single: 10 Hosts

```
vboxuser@ubu22to:~/Desktop$ sudo mn --topo single,10
*** Creating network
*** Adding controller
*** Adding hosts:
h1 h2 h3 h4 h5 h6 h7 h8 h9 h10
*** Adding switches:
s1
*** Adding links:
(h1, s1) (h2, s1) (h3, s1) (h4, s1) (h5, s1) (h6, s1) (h7, s1) (h8, s1) (h9, s1) (h10, s1)
*** Configuring hosts
h1 h2 h3 h4 h5 h6 h7 h8 h9 h10
*** Starting controller
c0
*** Starting 1 switches
s1 ...
*** Starting CLI:
mininet> nodes
available nodes are:
c0 h1 h10 h2 h3 h4 h5 h6 h7 h8 h9 s1
mininet> pingall
*** Ping: testing ping reachability
h1 -> h2 h3 h4 h5 h6 h7 h8 h9 h10
h2 -> h1 h3 h4 h5 h6 h7 h8 h9 h10
h3 -> h1 h2 h4 h5 h6 h7 h8 h9 h10
h4 -> h1 h2 h3 h5 h6 h7 h8 h9 h10
h5 -> h1 h2 h3 h4 h6 h7 h8 h9 h10
h6 -> h1 h2 h3 h4 h5 h7 h8 h9 h10
h7 -> h1 h2 h3 h4 h5 h6 h8 h9 h10
h8 -> h1 h2 h3 h4 h5 h6 h7 h9 h10
h9 -> h1 h2 h3 h4 h5 h6 h7 h8 h10
h10 -> h1 h2 h3 h4 h5 h6 h7 h8 h9
*** Results: 0% dropped (90/90 received)
mininet> exit
*** Stopping 1 controllers
c0
*** Stopping 10 links
.....
*** Stopping 1 switches
s1
*** Stopping 10 hosts
h1 h2 h3 h4 h5 h6 h7 h8 h9 h10
*** Done
completed in 16.720 seconds
vboxuser@ubu22to:~/Desktop$
```

Reversed: 10 Hosts

```
vboxuser@ubu22to:~/Desktop$ sudo mn --topo reversed,10
*** Creating network
*** Adding controller
*** Adding hosts:
h1 h2 h3 h4 h5 h6 h7 h8 h9 h10
*** Adding switches:
s1
*** Adding links:
(h1, s1) (h2, s1) (h3, s1) (h4, s1) (h5, s1) (h6, s1) (h7, s1) (h8, s1) (h9, s1) (h10, s1)
*** Configuring hosts
h1 h2 h3 h4 h5 h6 h7 h8 h9 h10
*** Starting controller
c0
*** Starting 1 switches
s1 ...
*** Starting CLI:
mininet> nodes
available nodes are:
c0 h1 h10 h2 h3 h4 h5 h6 h7 h8 h9 s1
mininet> pingall
*** Ping: testing ping reachability
h1 -> h2 h3 h4 h5 h6 h7 h8 h9 h10
h2 -> h1 h3 h4 h5 h6 h7 h8 h9 h10
h3 -> h1 h2 h4 h5 h6 h7 h8 h9 h10
h4 -> h1 h2 h3 h5 h6 h7 h8 h9 h10
h5 -> h1 h2 h3 h4 h6 h7 h8 h9 h10
h6 -> h1 h2 h3 h4 h5 h7 h8 h9 h10
h7 -> h1 h2 h3 h4 h5 h6 h8 h9 h10
h8 -> h1 h2 h3 h4 h5 h6 h7 h9 h10
h9 -> h1 h2 h3 h4 h5 h6 h7 h8 h10
h10 -> h1 h2 h3 h4 h5 h6 h7 h8 h9
*** Results: 0% dropped (90/90 received)
mininet> exit
*** Stopping 1 controllers
c0
*** Stopping 10 links
.....
*** Stopping 1 switches
s1
*** Stopping 10 hosts
h1 h2 h3 h4 h5 h6 h7 h8 h9 h10
*** Done
completed in 15.446 seconds
vboxuser@ubu22to:~/Desktop$
```

Linear: 10 Switches

```
vboxuser@ubu22to:~/Desktop$ sudo mn --topo linear,10
*** Creating network
*** Adding controller
*** Adding hosts:
h1 h2 h3 h4 h5 h6 h7 h8 h9 h10
*** Adding switches:
s1 s2 s3 s4 s5 s6 s7 s8 s9 s10
*** Adding links:
(h1, s1) (h2, s2) (h3, s3) (h4, s4) (h5, s5) (h6, s6) (h7, s7) (h8, s8) (h9, s9) (h10,
s10) (s2, s1) (s3, s2) (s4, s3) (s5, s4) (s6, s5) (s7, s6) (s8, s7) (s9, s8) (s10, s9)
*** Configuring hosts
h1 h2 h3 h4 h5 h6 h7 h8 h9 h10
*** Starting controller
c0
*** Starting 10 switches
s1 s2 s3 s4 s5 s6 s7 s8 s9 s10 ...
*** Starting CLI:
mininet> nodes
available nodes are:
c0 h1 h10 h2 h3 h4 h5 h6 h7 h8 h9 s1 s10 s2 s3 s4 s5 s6 s7 s8 s9
mininet> pingall
*** Ping: testing ping reachability
h1 -> h2 h3 h4 h5 h6 h7 h8 h9 h10
h2 -> h1 h3 h4 h5 h6 h7 h8 h9 h10
h3 -> h1 h2 h4 h5 h6 h7 h8 h9 h10
h4 -> h1 h2 h3 h5 h6 h7 h8 h9 h10
h5 -> h1 h2 h3 h4 h6 h7 h8 h9 h10
h6 -> h1 h2 h3 h4 h5 h7 h8 h9 h10
h7 -> h1 h2 h3 h4 h5 h6 h8 h9 h10
h8 -> h1 h2 h3 h4 h5 h6 h7 h9 h10
h9 -> h1 h2 h3 h4 h5 h6 h7 h8 h10
h10 -> h1 h2 h3 h4 h5 h6 h7 h8 h9
*** Results: 0% dropped (90/90 received)
mininet> exit
*** Stopping 1 controllers
c0
*** Stopping 19 links
.....
*** Stopping 10 switches
s1 s2 s3 s4 s5 s6 s7 s8 s9 s10
*** Stopping 10 hosts
h1 h2 h3 h4 h5 h6 h7 h8 h9 h10
*** Done
completed in 12.087 seconds
vboxuser@ubu22to:~/Desktop$
```

Tree: 3 Switches and 4 Hosts

```
vboxuser@ubu22to:~/Desktop$ sudo mn --topo tree,depth=2,fanout=2
*** Creating network
*** Adding controller
*** Adding hosts:
h1 h2 h3 h4
*** Adding switches:
s1 s2 s3
*** Adding links:
(s1, s2) (s1, s3) (s2, h1) (s2, h2) (s3, h3) (s3, h4)
*** Configuring hosts
h1 h2 h3 h4
*** Starting controller
c0
*** Starting 3 switches
s1 s2 s3 ...
*** Starting CLI:
mininet> nodes
available nodes are:
c0 h1 h2 h3 h4 s1 s2 s3
mininet> pingall
*** Ping: testing ping reachability
h1 -> h2 h3 h4
h2 -> h1 h3 h4
h3 -> h1 h2 h4
h4 -> h1 h2 h3
*** Results: 0% dropped (12/12 received)
mininet> exit
*** Stopping 1 controllers
c0
*** Stopping 6 links
.....
*** Stopping 3 switches
s1 s2 s3
*** Stopping 4 hosts
h1 h2 h3 h4
*** Done
completed in 8.968 seconds
vboxuser@ubu22to:~/Desktop$
```

Question 3: Write a python script that generates the following topology

1. In every topology there should be controller. Make sure to add a controller c0 in your code.

```
9         # add controller
10        c0 = net.addController('c0')
```

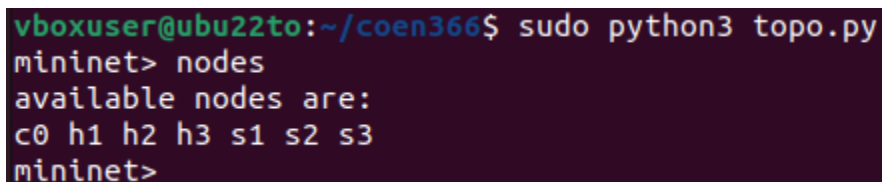
2. You must use the Mid-Level APIs to create the given topology.

```
6         # mininet network object
7        net = Mininet(controller=Controller, switch=OVSKernelSwitch)
8
9        # add controller
10       c0 = net.addController('c0')
11
12       # add the 3 hosts
13       h1 = net.addHost('h1')
14       h2 = net.addHost('h2')
15       h3 = net.addHost('h3')
16
17       # add the 3 switches
18       s1 = net.addSwitch('s1')
19       s2 = net.addSwitch('s2')
20       s3 = net.addSwitch('s3')
21
22       # links logic:
23       # s1-s2-s3 all connected together in triangle connection
24       # s1-h1, s2-h2, s3-h3 connections
25       net.addLink(s1, s2)
26       net.addLink(s2, s3)
27       net.addLink(s3, s1)
28       net.addLink(h1, s1)
29       net.addLink(h2, s2)
30       net.addLink(h3, s3)
```

3. Use the CLI object to open the CLI of Mininet

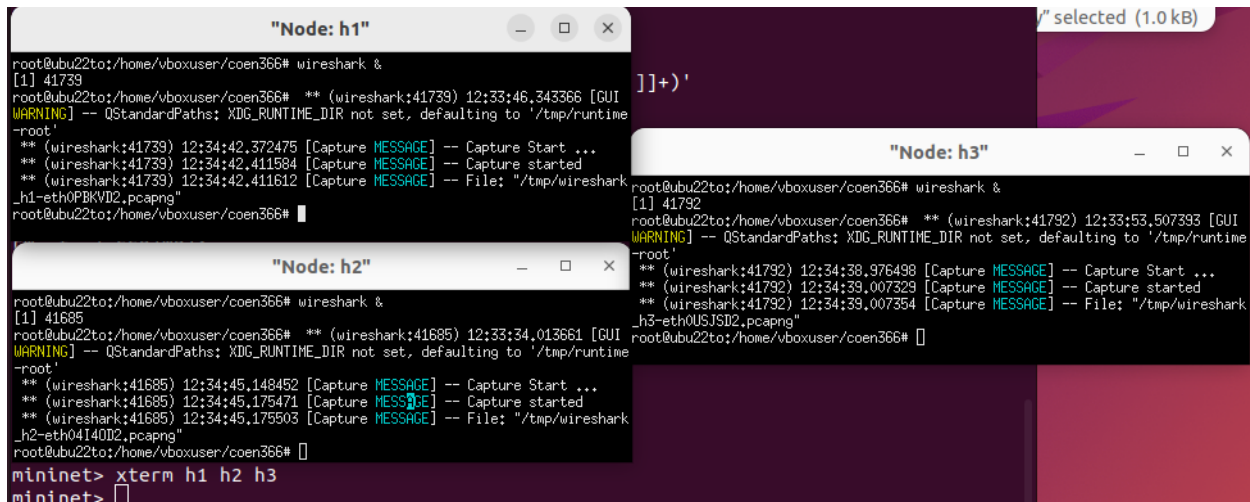
```
42         # open cli mininet
43        CLI(net)
```

4. Run the nodes and show the result. (Provide screenshot)



```
vboxuser@ubu22to:~/coen366$ sudo python3 topo.py
mininet> nodes
available nodes are:
c0 h1 h2 h3 s1 s2 s3
mininet>
```

5. Run Wireshark on h1, h2 and on h3.

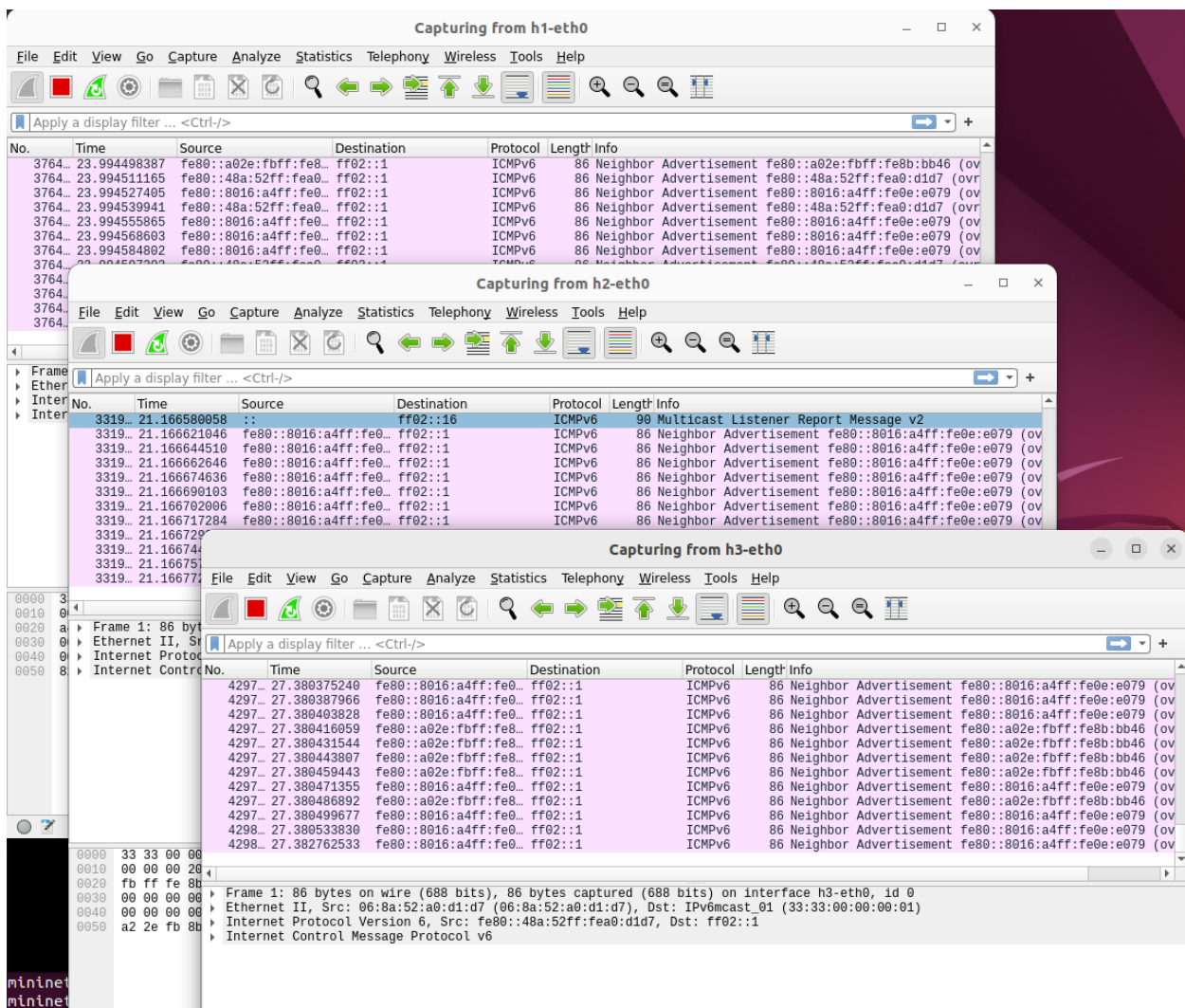


```
root@ubu22to:/home/vboxuser/coen366# wireshark &
[1] 41739
root@ubu22to:/home/vboxuser/coen366# ** (wireshark:41739) 12:33:46,343366 [GUI
WARNING] -- QStandardPaths: XDG_RUNTIME_DIR not set, defaulting to '/tmp/runtime
-root'
** (wireshark:41739) 12:34:42,372475 [Capture MESSAGE] -- Capture Start ...
** (wireshark:41739) 12:34:42,411584 [Capture MESSAGE] -- Capture started
** (wireshark:41739) 12:34:42,411612 [Capture MESSAGE] -- File: "/tmp/wireshark
_h1-eth0PBKVD2.pcapng"
root@ubu22to:/home/vboxuser/coen366#

root@ubu22to:/home/vboxuser/coen366# wireshark &
[1] 41685
root@ubu22to:/home/vboxuser/coen366# ** (wireshark:41685) 12:33:34,013661 [GUI
WARNING] -- QStandardPaths: XDG_RUNTIME_DIR not set, defaulting to '/tmp/runtime
-root'
** (wireshark:41685) 12:34:45,148452 [Capture MESSAGE] -- Capture Start ...
** (wireshark:41685) 12:34:45,175471 [Capture MESSAGE] -- Capture started
** (wireshark:41685) 12:34:45,175503 [Capture MESSAGE] -- File: "/tmp/wireshark
_h2-eth04I40D2.pcapng"
root@ubu22to:/home/vboxuser/coen366#

root@ubu22to:/home/vboxuser/coen366# wireshark &
[1] 41792
root@ubu22to:/home/vboxuser/coen366# ** (wireshark:41792) 12:33:53,507393 [GUI
WARNING] -- QStandardPaths: XDG_RUNTIME_DIR not set, defaulting to '/tmp/runtime
-root'
** (wireshark:41792) 12:34:38,976498 [Capture MESSAGE] -- Capture Start ...
** (wireshark:41792) 12:34:39,007329 [Capture MESSAGE] -- Capture started
** (wireshark:41792) 12:34:39,007354 [Capture MESSAGE] -- File: "/tmp/wireshark
_h3-eth0USJSD2.pcapng"
root@ubu22to:/home/vboxuser/coen366#

mininet> xterm h1 h2 h3
mininet>
```



Capturing from h1-eth0

No.	Time	Source	Destination	Protocol	Length	Info
3764.	23.994498387	fe80::a02e:fbff:fe0...	ff02::1	ICMPv6	86	Neighbor Advertisement fe80::a02e:fbff:fe0b:bb46 (ov
3764.	23.994511165	fe80::48a:52ff:fea0...	ff02::1	ICMPv6	86	Neighbor Advertisement fe80::48a:52ff:fea0:d1d7 (ovr
3764.	23.994527405	fe80::8016:a4ff:fe0...	ff02::1	ICMPv6	86	Neighbor Advertisement fe80::8016:a4ff:fe0e:e079 (ov
3764.	23.994539941	fe80::48a:52ff:fea0...	ff02::1	ICMPv6	86	Neighbor Advertisement fe80::48a:52ff:fea0:d1d7 (ovr
3764.	23.994555865	fe80::8016:a4ff:fe0...	ff02::1	ICMPv6	86	Neighbor Advertisement fe80::8016:a4ff:fe0e:e079 (ov
3764.	23.994568693	fe80::8016:a4ff:fe0...	ff02::1	ICMPv6	86	Neighbor Advertisement fe80::8016:a4ff:fe0e:e079 (ov
3764.	23.994584892	fe80::8016:a4ff:fe0...	ff02::1	ICMPv6	86	Neighbor Advertisement fe80::8016:a4ff:fe0e:e079 (ov

Capturing from h2-eth0

No.	Time	Source	Destination	Protocol	Length	Info
3319.	21.166580058	::	ff02::16	ICMPv6	90	Multicast Listener Report Message v2
3319.	21.166621046	fe80::8016:a4ff:fe0...	ff02::1	ICMPv6	86	Neighbor Advertisement fe80::8016:a4ff:fe0e:e079 (ov
3319.	21.166644510	fe80::8016:a4ff:fe0...	ff02::1	ICMPv6	86	Neighbor Advertisement fe80::8016:a4ff:fe0e:e079 (ov
3319.	21.166662646	fe80::8016:a4ff:fe0...	ff02::1	ICMPv6	86	Neighbor Advertisement fe80::8016:a4ff:fe0e:e079 (ov
3319.	21.166674636	fe80::8016:a4ff:fe0...	ff02::1	ICMPv6	86	Neighbor Advertisement fe80::8016:a4ff:fe0e:e079 (ov
3319.	21.166690103	fe80::8016:a4ff:fe0...	ff02::1	ICMPv6	86	Neighbor Advertisement fe80::8016:a4ff:fe0e:e079 (ov
3319.	21.166702906	fe80::8016:a4ff:fe0...	ff02::1	ICMPv6	86	Neighbor Advertisement fe80::8016:a4ff:fe0e:e079 (ov
3319.	21.166717284	fe80::8016:a4ff:fe0...	ff02::1	ICMPv6	86	Neighbor Advertisement fe80::8016:a4ff:fe0e:e079 (ov

Capturing from h3-eth0

No.	Time	Source	Destination	Protocol	Length	Info
4297...	27.380375240	fe80::8016:a4ff:fe0...	ff02::1	ICMPv6	86	Neighbor Advertisement fe80::8016:a4ff:fe0e:e079 (ov
4297...	27.380387966	fe80::8016:a4ff:fe0...	ff02::1	ICMPv6	86	Neighbor Advertisement fe80::8016:a4ff:fe0e:e079 (ov
4297...	27.380403828	fe80::8016:a4ff:fe0...	ff02::1	ICMPv6	86	Neighbor Advertisement fe80::8016:a4ff:fe0e:e079 (ov
4297...	27.380416059	fe80::a02e:fbff:fe8...	ff02::1	ICMPv6	86	Neighbor Advertisement fe80::a02e:fbff:fe8b:bb46 (ov
4297...	27.380431544	fe80::a02e:fbff:fe8...	ff02::1	ICMPv6	86	Neighbor Advertisement fe80::a02e:fbff:fe8b:bb46 (ov
4297...	27.380443807	fe80::a02e:fbff:fe8...	ff02::1	ICMPv6	86	Neighbor Advertisement fe80::a02e:fbff:fe8b:bb46 (ov
4297...	27.380459443	fe80::a02e:fbff:fe8...	ff02::1	ICMPv6	86	Neighbor Advertisement fe80::a02e:fbff:fe8b:bb46 (ov
4297...	27.380471355	fe80::8016:a4ff:fe0...	ff02::1	ICMPv6	86	Neighbor Advertisement fe80::8016:a4ff:fe0e:e079 (ov
4297...	27.380486892	fe80::a02e:fbff:fe8...	ff02::1	ICMPv6	86	Neighbor Advertisement fe80::a02e:fbff:fe8b:bb46 (ov
4297...	27.380499677	fe80::8016:a4ff:fe0...	ff02::1	ICMPv6	86	Neighbor Advertisement fe80::8016:a4ff:fe0e:e079 (ov
4298...	27.380533830	fe80::8016:a4ff:fe0...	ff02::1	ICMPv6	86	Neighbor Advertisement fe80::8016:a4ff:fe0e:e079 (ov
4298...	27.382762533	fe80::8016:a4ff:fe0...	ff02::1	ICMPv6	86	Neighbor Advertisement fe80::8016:a4ff:fe0e:e079 (ov

Frame 1: 86 bytes on wire (688 bits), 86 bytes captured (688 bits) on interface h3-eth0, id 0
Ethernet II, Src: 06:8a:52:a0:d1:d7 (06:8a:52:a0:d1:d7), Dst: IPv6mcast 01 (33:33:00:00:00:01)
Internet Protocol Version 6, Src: fe80::48a:52ff:fea0:d1d7, Dst: ff02::1
Internet Control Message Protocol v6

6. Ping h1 from h3 (one packet), how long did it take to ping? Ping h3 from h1. How long does it take? Is there a difference? Capture the incoming packet in Wireshark and provide a screenshot of it. These will help the TAs validate that your submission is working correctly on your machine if for some reason your submitted code does not work in the grading environment.

I was not able to have a valid ping connection between h1, h2 and h3. I was only able to capture the ping of itself such as h1 to h1, h2 to h2 and h3 to h3 in which their time is 0.054ms, 0.029ms and 0.028ms respectively.

"Node: h1"	"Node: h2"	"Node: h3"
<pre>root@ubu22to:/home/vboxuser/coen366# ping -c 1 10.0.0.1 PING 10.0.0.1 (10.0.0.1) 56(84) bytes of data: 64 bytes from 10.0.0.1: icmp_seq=1 ttl=64 time=0.054 ms --- 10.0.0.1 ping statistics --- 1 packets transmitted, 1 received, 0% packet loss, time 0ms rtt min/avg/max/mdev = 0.054/0.054/0.054/0.000 ms</pre>	<pre>root@ubu22to:/home/vboxuser/coen366# ping -c 1 10.0.0.2 PING 10.0.0.2 (10.0.0.2) 56(84) bytes of data: 64 bytes from 10.0.0.2: icmp_seq=1 ttl=64 time=0.029 ms --- 10.0.0.2 ping statistics --- 1 packets transmitted, 1 received, 0% packet loss, time 0ms rtt min/avg/max/mdev = 0.029/0.029/0.029/0.000 ms</pre>	<pre>root@ubu22to:/home/vboxuser/coen366# ping -c 1 10.0.0.3 PING 10.0.0.3 (10.0.0.3) 56(84) bytes of data: 64 bytes from 10.0.0.3: icmp_seq=1 ttl=64 time=0.028 ms --- 10.0.0.3 ping statistics --- 1 packets transmitted, 1 received, 0% packet loss, time 0ms rtt min/avg/max/mdev = 0.028/0.028/0.028/0.000 ms</pre>

7. Repeat Step 6 for h1-h2 and h2-h3.

"Node: h1"	"Node: h2"	"Node: h3"
<pre>root@ubu22to:/home/vboxuser/coen366# ping -c 1 10.0.0.2 PING 10.0.0.2 (10.0.0.2) 56(84) bytes of data: From 10.0.0.1 icmp_seq=1 Destination Host Unreachable --- 10.0.0.2 ping statistics --- 1 packets transmitted, 0 received, +1 errors, 100% packet loss, time 0ms root@ubu22to:/home/vboxuser/coen366# ping -c 1 10.0.0.3 PING 10.0.0.3 (10.0.0.3) 56(84) bytes of data: From 10.0.0.1 icmp_seq=1 Destination Host Unreachable --- 10.0.0.3 ping statistics --- 1 packets transmitted, 0 received, +1 errors, 100% packet loss, time 0ms root@ubu22to:/home/vboxuser/coen366#</pre>	<pre>root@ubu22to:/home/vboxuser/coen366# ping -c 1 10.0.0.1 PING 10.0.0.1 (10.0.0.1) 56(84) bytes of data: From 10.0.0.2 icmp_seq=1 Destination Host Unreachable --- 10.0.0.1 ping statistics --- 1 packets transmitted, 0 received, +1 errors, 100% packet loss, time 0ms root@ubu22to:/home/vboxuser/coen366# ping -c 1 10.0.0.3 PING 10.0.0.3 (10.0.0.3) 56(84) bytes of data: From 10.0.0.2 icmp_seq=1 Destination Host Unreachable --- 10.0.0.3 ping statistics --- 1 packets transmitted, 0 received, +1 errors, 100% packet loss, time 0ms root@ubu22to:/home/vboxuser/coen366#</pre>	<pre>root@ubu22to:/home/vboxuser/coen366# ping -c 1 10.0.0.1 PING 10.0.0.1 (10.0.0.1) 56(84) bytes of data: From 10.0.0.3 icmp_seq=1 Destination Host Unreachable --- 10.0.0.1 ping statistics --- 1 packets transmitted, 0 received, +1 errors, 100% packet loss, time 0ms root@ubu22to:/home/vboxuser/coen366# ping -c 1 10.0.0.2 PING 10.0.0.2 (10.0.0.2) 56(84) bytes of data: From 10.0.0.3 icmp_seq=1 Destination Host Unreachable --- 10.0.0.2 ping statistics --- 1 packets transmitted, 0 received, +1 errors, 100% packet loss, time 0ms root@ubu22to:/home/vboxuser/coen366#</pre>

8. Perform a Pingall and copy the output, verifying that all hosts are pingable.

```
mininet> pingall
*** Ping: testing ping reachability
h1 -> X X
h2 -> X X
h3 -> X X
*** Results: 100% dropped (0/6 received)
```


From running “pingall” command, I know the issues lies in the code where it has a network connectivity problem. The pings are failing due to the host unable to reach each and other. I have tried setting IP addresses to each host such as 10.0.0.1 for h1, but it still won’t work.

9. Submit your .py file along with your assignment document.

The “topo.py” code is copy and paste below as well as attached to the final submission document.

“topo.py” Code

```
from mininet.net import Mininet
from mininet.node import Controller, OVSKernelSwitch
from mininet.cli import CLI

def topo():
    # mininet network object
    net = Mininet(controller=Controller, switch=OVSKernelSwitch)

    # add controller, hosts and switches
    c0 = net.addController('c0')
    h1 = net.addHost('h1')
    h2 = net.addHost('h2')
    h3 = net.addHost('h3')
    s1 = net.addSwitch('s1')
    s2 = net.addSwitch('s2')
    s3 = net.addSwitch('s3')

    # links logic:
    # s1-s2-s3 all connected together in triangle connection
    # s1-h1, s2-h2, s3-h3 connections
    net.addLink(s1, s2)
    net.addLink(s1, s2)
    net.addLink(s2, s3)
    net.addLink(h1, s1)
    net.addLink(h2, s2)
    net.addLink(h3, s3)

    # begin network
    net.build()
    c0.start()
    s1.start([c0])
    s2.start([c0])
    s3.start([c0])
    h1.cmd('ifconfig h1-eth0 10.0.0.1/24')
    h2.cmd('ifconfig h2-eth0 10.0.0.2/24')
    h3.cmd('ifconfig h3-eth0 10.0.0.3/24')

    # open cli mininet
    CLI(net)

    # stop network
    net.stop()

if __name__ == '__main__':
    topo()
```

2. Wireshark Assignment 2

1. Run nslookup to obtain the IP address of a Web server in Asia. What is the IP address of that server?

```
C:\Users\andre>nslookup im.qq.com
Server: h268a
Address: 192.168.1.1

Non-authoritative answer:
Name: im.qq.com
Addresses: 240e:ff:f100:1009::10f
           203.205.254.62

C:\Users\andre>
```

I chose <https://im.qq.com/> as the website of choice which has an IP address of 203.205.254.62.

2. Run nslookup to determine the authoritative DNS servers for a university in Europe.

```
C:\Users\andre>nslookup -type=NS www.ox.ac.uk
Server: h268a
Address: 192.168.1.1

ox.ac.uk
primary name server = raptor.dns.ox.ac.uk
responsible mail addr = hostmaster.ox.ac.uk
serial = 2023110268
refresh = 3600 (1 hour)
retry = 1800 (30 mins)
expire = 1209600 (14 days)
default TTL = 900 (15 mins)

C:\Users\andre>
```

3. Run nslookup so that one of the DNS servers obtained in Question 2 is queried for the mail servers for Yahoo! mail. What is its IP address?

```

C:\Users\andre>nslookup www.ox.ac.uk mail.yahoo.com
DNS request timed out.
    timeout was 2 seconds.
Server:  UnKnown
Address:  69.147.92.11

DNS request timed out.
    timeout was 2 seconds.
DNS request timed out.
    timeout was 2 seconds.
DNS request timed out.
    timeout was 2 seconds.
DNS request timed out.
    timeout was 2 seconds.
*** Request to UnKnown timed-out

C:\Users\andre>

```

The IP address is 69.147.92.11.

4. Locate the DNS query and response messages. Are then sent over UDP or TCP?

25507	37.318...	192.168.1.24	192.168.1.1	DNS	72 Standard query 0x05c2 A www.ietf.org
25508	37.318...	192.168.1.24	192.168.1.1	DNS	72 Standard query 0xb218 HTTPS www.ietf.org
25509	37.331...	192.168.1.24	192.168.1.1	DNS	72 Standard query 0xee74 A www.ietf.org
25510	37.338...	192.168.1.1	192.168.1.24	DNS	104 Standard query response 0x05c2 A www.ietf.org A 104.16.45.99 A 104.16.4
25511	37.339...	192.168.1.1	192.168.1.24	DNS	145 Standard query response 0xb218 HTTPS www.ietf.org HTTPS
25512	37.341...	23.223.17.207	192.168.1.24	TCP	60 443 → 51815 [ACK] Seq=1 Ack=2 Win=501 Len=0
25513	37.343...	192.168.1.24	192.168.1.1	DNS	72 Standard query 0x7841 A www.ietf.org
25514	37.344...	192.168.1.24	192.168.1.1	DNS	72 Standard query 0xdd24 HTTPS www.ietf.org

> Frame 25507: 72 bytes on wire (576 bits), 72 bytes captured (576	0000	d0 60 8c 03 d4 cc f0 2f 74 cc 70 1e 08 00 45 00/ t.p...E.
> Ethernet II, Src: ASUSTekC_cc:70:1e (f0:2f:74:cc:70:1e), Dst: zt	0010	00 3a cf 3a 00 00 80 11 e8 0e c0 a8 01 18 c0 a8	...:.....
> Internet Protocol Version 4, Src: 192.168.1.24, Dst: 192.168.1.1	0020	01 01 c5 df 00 35 00 26 8e 1c 05 c2 01 00 00 01	...:5-&.....
> User Datagram Protocol, Src Port: 50655, Dst Port: 53	0030	00 00 00 00 00 00 03 77 77 77 04 69 65 74 66 03W ww.ietf.
Source Port: 50655	0040	6f 72 67 00 00 01 00 01	org:....
Destination Port: 53			
Length: 38			
Checksum: 0x8e1c [unverified]			
[Checksum Status: Unverified]			
[Stream index: 18]			
> [Timestamps]			
UDP payload (30 bytes)			
> Domain Name System (query)			

They are sent over User Datagram Protocol (UDP).

5. What is the destination port for the DNS query message? What is the source port of DNS response message?

```

25507 37.318... 192.168.1.24 192.168.1.1 DNS 72 Standard query 0x05c2 A www.ietf.org
25508 37.318... 192.168.1.24 192.168.1.1 DNS 72 Standard query 0xb218 HTTPS www.ietf.org
25509 37.331... 192.168.1.24 192.168.1.1 DNS 72 Standard query 0xee74 A www.ietf.org
25510 37.338... 192.168.1.1 192.168.1.24 DNS 104 Standard query response 0x05c2 A www.ietf.org A 104.16.45.99 A 104.16.4
25511 37.339... 192.168.1.1 192.168.1.24 DNS 145 Standard query response 0xb218 HTTPS www.ietf.org HTTPS
25512 37.341... 23.223.17.207 192.168.1.24 TCP 60 443 → 51815 [ACK] Seq=1 Ack=2 Win=501 Len=0
25513 37.343... 192.168.1.24 192.168.1.1 DNS 72 Standard query 0x7841 A www.ietf.org
25514 37.344... 192.168.1.24 192.168.1.1 DNS 72 Standard query 0xdd24 HTTPS www.ietf.org

> Frame 25507: 72 bytes on wire (576 bits), 72 bytes captured (576) on interface 0
> Ethernet II, Src: ASUSTekC_cc:70:1e (f0:2f:74:cc:70:1e), Dst: zt
> Internet Protocol Version 4, Src: 192.168.1.24, Dst: 192.168.1.1
> User Datagram Protocol, Src Port: 50655, Dst Port: 53
  Source Port: 50655
  Destination Port: 53
  Length: 30
  Checksum: 0x8e1c [unverified]
  [Checksum Status: Unverified]
  [Stream index: 18]
  > [Timestamps]
  UDP payload (30 bytes)
  Domain Name System (query)
    0000 d0 60 8c 03 d4 cc f0 2f 74 cc 70 1e 08 00 45 00 / t.p...E
    0010 00 3a cf 3a 00 00 80 11 e8 0e c0 a8 01 18 c0 a8 : : : :
    0020 01 01 c5 df 00 35 00 26 8e 1c 05 c2 01 00 00 01 : : : : 5 &
    0030 00 00 00 00 00 00 03 77 77 77 04 69 65 74 66 03 : : : : w ww.ietf
    0040 6f 72 67 00 00 01 00 01 : : : : org : : : :

```

The destination port of the query message and the source port of the response message are 53.

6. To what IP address is the DNS query message sent? Use ipconfig to determine the IP address of your local DNS server. Are these two IP addresses the same?

```

> Internet Protocol Version 4, Src: 192.168.1.24, Dst: 192.168.1.1

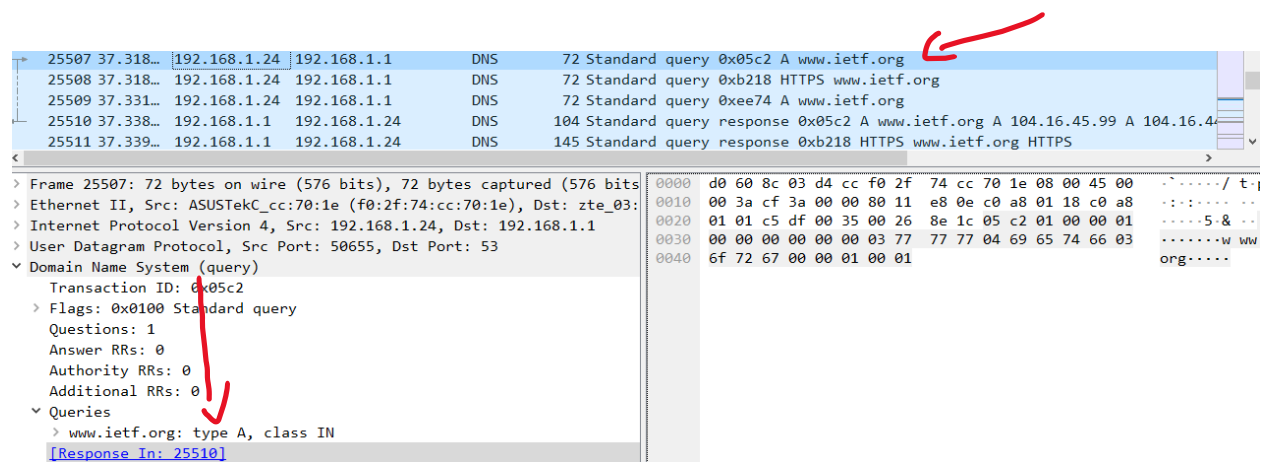
Ethernet adapter Ethernet:

    Connection-specific DNS Suffix . : 
    Description . . . . . : Realtek PCIe GBE Family Controller
    Physical Address. . . . . : F0-2F-74-CC-70-1E
    DHCP Enabled. . . . . : Yes
    Autoconfiguration Enabled . . . . : Yes
    IPv4 Address. . . . . : 192.168.1.24(Preferred)
    Subnet Mask . . . . . : 255.255.255.0
    Lease Obtained. . . . . : Wednesday, November 1, 2023 7:52:02 PM
    Lease Expires . . . . . : Friday, November 3, 2023 9:31:50 AM
    Default Gateway . . . . . : 192.168.1.1
    DHCP Server . . . . . : 192.168.1.1
    DNS Servers . . . . . : 192.168.1.1
    NetBIOS over Tcpip. . . . . : Enabled

```

Yes, both IP addresses are 192.168.1.1.

7. Examine the DNS query message. What “Type” of DNS query is it? Does the query message contain any “answers”?

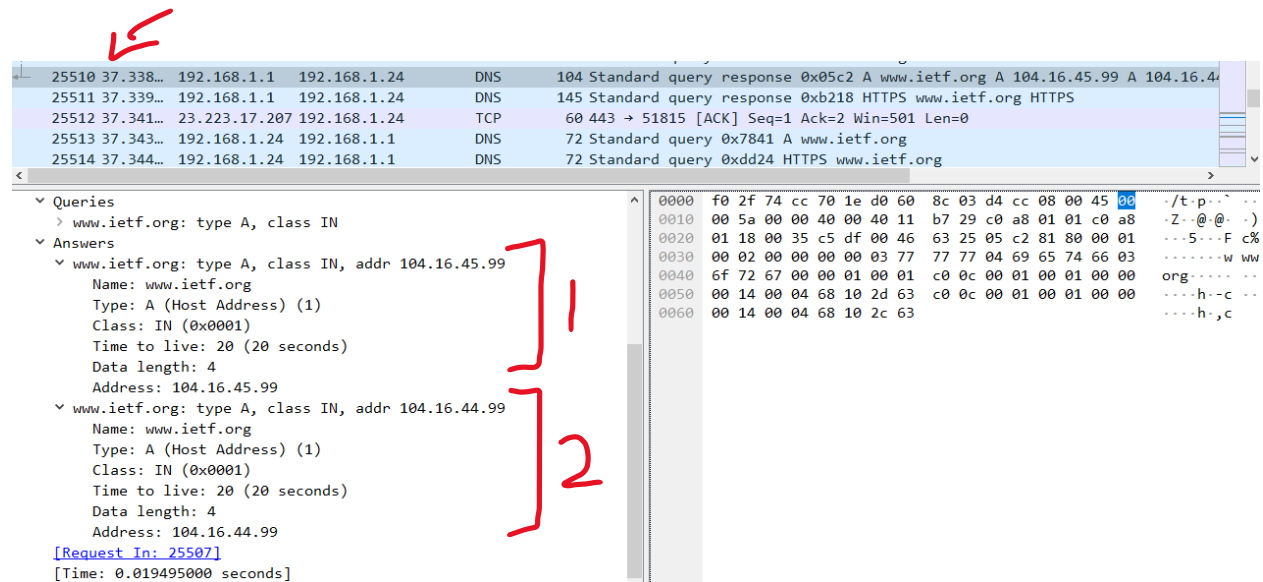


Packet 25507: 72 bytes on wire (576 bits), 72 bytes captured (576 bits) on interface 0. The packet is a DNS Standard query (Type A) for www.ietf.org. The query is sent from 192.168.1.24 to 192.168.1.1. The packet details show the following structure:

- Frame 25507: 72 bytes on wire (576 bits), 72 bytes captured (576 bits)
- Ethernet II, Src: ASUSTekC_cc:70:1e (f0:2f:74:cc:70:1e), Dst: zte_03:00:00:00:00:00
- Internet Protocol Version 4, Src: 192.168.1.24, Dst: 192.168.1.1
- User Datagram Protocol, Src Port: 50655, Dst Port: 53
- Domain Name System (query)
 - Transaction ID: 0x05c2
 - Flags: 0x0100 Standard query
 - Questions: 1
 - Answer RRs: 0
 - Authority RRs: 0
 - Additional RRs: 0
 - Queries
 - www.ietf.org: type A, class IN

It is of type A, however it doesn't contain any answers.

8. Examine the DNS response message. How many “answers” are provided? What do each of these answers contain?



Packet 25510: 104 bytes on wire (832 bits), 104 bytes captured (832 bits) on interface 0. The packet is a DNS Standard query response (Type A) for www.ietf.org. The response is sent from 192.168.1.1 to 192.168.1.24. The packet details show the following structure:

- Frame 25510: 104 bytes on wire (832 bits), 104 bytes captured (832 bits)
- Ethernet II, Src: zte_03:00:00:00:00:00, Dst: ASUSTekC_cc:70:1e (f0:2f:74:cc:70:1e)
- Internet Protocol Version 4, Src: 192.168.1.1, Dst: 192.168.1.24
- User Datagram Protocol, Src Port: 53, Dst Port: 50655
- Domain Name System (query response)
 - Transaction ID: 0x05c2
 - Flags: 0x0000 Standard query response
 - Questions: 1
 - Answer RRs: 2
 - Authority RRs: 0
 - Additional RRs: 0
 - Queries
 - www.ietf.org: type A, class IN
 - Answers
 - www.ietf.org: type A, class IN, addr 104.16.45.99
 - Name: www.ietf.org
 - Type: A (Host Address) (1)
 - Class: IN (0x0001)
 - Time to live: 20 (20 seconds)
 - Data length: 4
 - Address: 104.16.45.99
 - www.ietf.org: type A, class IN, addr 104.16.44.99
 - Name: www.ietf.org
 - Type: A (Host Address) (1)
 - Class: IN (0x0001)
 - Time to live: 20 (20 seconds)
 - Data length: 4
 - Address: 104.16.44.99

The response No.25510 has two answers with the difference being their IP addresses being 104.16.45.99 and 104.16.44.99 respectively. They contain name, class, time, data len and address.

9. Consider the subsequent TCP SYN packet sent by your host. Does the destination IP address of the SYN packet correspond to any of the IP addresses provided in the DNS response message?

The destination IP address of the SYN packet corresponds to the first IP address provided in the DNS response message being 104.16.45.99.

10. This web page contains images. Before retrieving each image, does your host issue new DNS queries?

No.	Time	Source	Destination	Protocol	Length	Info
871	2.387452	192.168.1.24	208.111.130.0	HTTP	486	GET /filestreamingservice/files/b6784df7-c744-43b7-ad1e-24d3b6b3457d?P1=1699
906	2.405063	208.111.130.0	192.168.1.24	HTTP	1204	HTTP/1.1 206 Partial Content (application/x-chrome-extension)
2711	5.429811	192.168.1.24	208.111.130.0	HTTP	487	GET /filestreamingservice/files/b6784df7-c744-43b7-ad1e-24d3b6b3457d?P1=1699
2789	5.467005	208.111.130.0	192.168.1.24	HTTP	1200	HTTP/1.1 206 Partial Content (application/x-chrome-extension)
4568	8.477860	192.168.1.24	208.111.130.0	HTTP	488	GET /filestreamingservice/files/b6784df7-c744-43b7-ad1e-24d3b6b3457d?P1=1699
4720	8.505711	208.111.130.0	192.168.1.24	HTTP	1500	HTTP/1.1 206 Partial Content (application/x-chrome-extension)
6407	11.546...	192.168.1.24	208.111.130.0	HTTP	488	GET /filestreamingservice/files/b6784df7-c744-43b7-ad1e-24d3b6b3457d?P1=1699
7591	13.587...	192.168.1.24	208.111.130.0	HTTP	489	GET /filestreamingservice/files/b6784df7-c744-43b7-ad1e-24d3b6b3457d?P1=1699
10016	15.622...	192.168.1.24	208.111.130.0	HTTP	490	GET /filestreamingservice/files/b6784df7-c744-43b7-ad1e-24d3b6b3457d?P1=1699
10205	15.658...	208.111.130.0	192.168.1.24	HTTP	581	HTTP/1.1 206 Partial Content (application/x-chrome-extension)
11142	16.970...	192.168.1.24	208.111.130.0	HTTP	411	HEAD /filestreamingservice/files/f22c863f-657e-4ae8-b972-495f0d906687?P1=1699
11143	16.984...	208.111.130.0	192.168.1.24	HTTP	906	HTTP/1.1 200 OK
11147	17.029...	192.168.1.24	208.111.130.0	HTTP	462	GET /filestreamingservice/files/f22c863f-657e-4ae8-b972-495f0d906687?P1=1699
11154	17.043...	208.111.130.0	192.168.1.24	HTTP	495	HTTP/1.1 200 OK (application/x-chrome-extension)
12920	21.079...	192.168.1.24	208.111.130.0	HTTP	407	HEAD /filestreamingservice/files/dd65cd6e-da04-4d01-9089-b864f080118b?P1=1699
12921	21.093...	208.111.130.0	192.168.1.24	HTTP	909	HTTP/1.1 200 OK
12922	21.124...	192.168.1.24	208.111.130.0	HTTP	458	GET /filestreamingservice/files/dd65cd6e-da04-4d01-9089-b864f080118b?P1=1699
20335	29.366...	192.168.1.24	208.111.130.0	HTTP	411	HEAD /filestreamingservice/files/93a7a55e-7f70-46ef-90f1-1523d4352c55?P1=1699
20336	29.383...	208.111.130.0	192.168.1.24	HTTP	908	HTTP/1.1 200 OK
20337	29.404...	192.168.1.24	208.111.130.0	HTTP	462	GET /filestreamingservice/files/93a7a55e-7f70-46ef-90f1-1523d4352c55?P1=1699
20480	29.433...	208.111.130.0	192.168.1.24	HTTP	1008	HTTP/1.1 200 OK (application/x-chrome-extension)
22027	33.472...	192.168.1.24	208.111.130.0	HTTP	409	HEAD /filestreamingservice/files/81f53c6c-e17f-45cb-b87d-615f2cb641f9?P1=1699
22028	33.489...	208.111.130.0	192.168.1.24	HTTP	907	HTTP/1.1 200 OK
22031	33.512...	192.168.1.24	208.111.130.0	HTTP	460	GET /filestreamingservice/files/81f53c6c-e17f-45cb-b87d-615f2cb641f9?P1=1699
22087	33.556...	208.111.130.0	192.168.1.24	HTTP	1510	HTTP/1.1 200 OK (application/x-chrome-extension)

> Frame 22031: 460 bytes on wire (3680 bits), 460 bytes captured (3680	0000	d0 60 8c 03 d4 cc f0 2f 74 cc 70 1e 08 00 45 00/ t
> Ethernet II, Src: ASUSTekC_cc:70:1e (f0:2f:74:cc:70:1e), Dst: zte_03:	0010	01 be 27 cd 40 00 80 06 bd 3c c0 a8 01 18 d0 6f	..'.@... 8
> Internet Protocol Version 4, Src: 192.168.1.24, Dst: 208.111.130.0	0020	82 00 ca 62 00 50 2d f3 25 c0 f8 cc 46 81 50 18	...b.P... 8
> Transmission Control Protocol, Src Port: 51810, Dst Port: 80, Seq: 52	0030	18 17 7d fc 00 00 47 45 54 20 2f 66 69 6c 65 73	..}...GE 1
> Hypertext Transfer Protocol	0040	74 72 65 61 6d 69 6e 67 73 65 72 76 69 63 65 2f	treaming s
	0050	66 69 6c 65 73 2f 38 31 66 35 33 63 36 63 2d 65	files/81 f
	0060	31 37 66 2d 34 35 63 62 2d 62 38 37 64 2d 36 31	17f-45cb -

No, all images are loaded directly from <http://www.ieft.org>. This can be observed due to the fact that no HTTP GET image are found when filtering for “http”.

11. What is the destination port for the DNS query message? What is the source port of DNS response message?

The image shows two Wireshark packet capture screenshots. The top screenshot displays a list of packets where packet 1279 is selected. The details pane shows the 'Domain Name System (query)' section with 'Transaction ID: 0xb8bf', 'Flags: 0x0100 Standard query', and 'Questions: 1'. The 'Queries' section is expanded, showing '[Response In: 1282]'. The bottom screenshot shows packet 1282 selected. The details pane shows the 'Domain Name System (response)' section with 'Transaction ID: 0xb8bf', 'Flags: 0x8180 Standard query response, No error', and 'Answer RRs: 3'. The 'Answers' section is expanded, showing '[Request In: 1279]'. Both screenshots have red circles around the 'Src Port: 59653, Dst Port: 53' field in the 'User Datagram Protocol' section of the selected packets.

The destination port of query message (No. 1279) is 53, while the source port of the response message (No. 1282) is also 53.

12. To what IP address is the DNS query message sent? Is this the IP address of your default local DNS server?

The image shows a Wireshark packet capture snippet. The 'Internet Protocol Version 4' section is expanded, showing 'Src: 192.168.1.24, Dst: 192.168.1.1'. The 'User Datagram Protocol' section is also expanded, showing 'Src Port: 59653, Dst Port: 53'. The IP address '192.168.1.1' is circled in red.


```

Ethernet adapter Ethernet:

    Connection-specific DNS Suffix  . : 
    Description . . . . . : Realtek PCIe GBE Family Controller
    Physical Address. . . . . : F0-2F-74-CC-70-1E
    DHCP Enabled. . . . . : Yes
    Autoconfiguration Enabled . . . . : Yes
    IPv4 Address. . . . . : 192.168.1.24(Preferred)
    Subnet Mask . . . . . : 255.255.255.0
    Lease Obtained. . . . . : Wednesday, November 1, 2023 7:52:02 PM
    Lease Expires . . . . . : Friday, November 3, 2023 9:31:50 AM
    Default Gateway . . . . . : 192.168.1.1
    DHCP Server . . . . . : 192.168.1.1
    DNS Servers . . . . . : 192.168.1.1
    NetBIOS over Tcpip. . . . . : Enabled

```

The IP address sent is to my default local DNS server of 192.168.1.1.

13. Examine the DNS query message. What “Type” of DNS query is it? Does the query message contain any “answers”?

The image shows a Wireshark packet capture of a DNS query. The packet list at the top shows three packets: a standard query (1279), a standard query (1280), and a standard query (1281). The selected packet (1279) is a standard query from 192.168.1.24 to 192.168.1.1. The packet details pane shows the query structure: Transaction ID: 0xb8bf, Flags: 0x0100 Standard query, Questions: 1, Answer RRs: 0, Authority RRs: 0, Additional RRs: 0. The query is for www.mit.edu: type A, class IN. The packet bytes pane shows the raw data of the query.

It is of type A, however it doesn't contain any answers.

14. Examine the DNS response message. How many “answers” are provided? What do each of these answers contain?

```
1282 2.08949 192.168.1.1 192.168.1.24 DNS 163 Standard query response 0xb8bf A www.mit.edu CNAME www.mit.edu.edgekey.net
1283 2.108949 192.168.1.1 192.168.1.24 DNS 225 Standard query response 0x4ad2 HTTPS www.mit.edu CNAME www.mit.edu.edgekey.net

Class: IN (0x0001)
Answers
  www.mit.edu: type CNAME, class IN, cname www.mit.edu.edgekey.net
    Name: www.mit.edu
    Type: CNAME (Canonical NAME for an alias) (5)
    Class: IN (0x0001)
    Time to live: 1800 (30 minutes)
    Data length: 25
    CNAME: www.mit.edu.edgekey.net
  www.mit.edu.edgekey.net: type CNAME, class IN, cname e9566.dscb.akamaiedge.net
    Name: www.mit.edu.edgekey.net
    Type: CNAME (Canonical NAME for an alias) (5)
    Class: IN (0x0001)
    Time to live: 60 (1 minute)
    Data length: 27
    CNAME: e9566.dscb.akamaiedge.net
  e9566.dscb.akamaiedge.net: type A, class IN, addr 184.31.129.206
    Name: e9566.dscb.akamaiedge.net
    Type: A (Host Address) (1)
    Class: IN (0x0001)
    Time to live: 20 (20 seconds)
    Data length: 4
    Address: 184.31.129.206
[Request In: 1279]
[Time: 0.026266000 seconds]
```

We have three answers. The first two are type CNAME while one is of type A. The first two contain name, class, time, data len and cname. The last one contains name, class, time, data len and address.

15. Provide a screenshot.

```
1279 2.082683 192.168.1.24 192.168.1.1 DNS 71 Standard query 0xb8bf A www.mit.edu
1280 2.082793 192.168.1.24 192.168.1.1 DNS 71 Standard query 0x4ad2 HTTPS www.mit.edu
1281 2.096347 192.168.1.24 192.168.1.1 DNS 71 Standard query 0x401e A www.mit.edu
1282 2.108949 192.168.1.1 192.168.1.24 DNS 163 Standard query response 0xb8bf A www.mit.edu CNAME www.mit.edu.edgekey.net
1283 2.108949 192.168.1.1 192.168.1.24 DNS 225 Standard query response 0x4ad2 HTTPS www.mit.edu CNAME www.mit.edu.edgekey.net
1284 2.109304 192.168.1.24 184.31.129.206 TCP 66 52306 -> 80 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK_PERM
1285 2.109611 192.168.1.24 184.31.129.206 TCP 66 52307 -> 80 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK_PERM
1286 2.118416 192.168.1.1 192.168.1.24 DNS 163 Standard query response 0x401e A www.mit.edu CNAME www.mit.edu.edgekey.net

> Queries
Answers
  www.mit.edu: type CNAME, class IN, cname www.mit.edu.edgekey.net
    Name: www.mit.edu
    Type: CNAME (Canonical NAME for an alias) (5)
    Class: IN (0x0001)
    Time to live: 1800 (30 minutes)
    Data length: 25
    CNAME: www.mit.edu.edgekey.net
  www.mit.edu.edgekey.net: type CNAME, class IN, cname e9566.dscb.akamaiedge.net
    Name: www.mit.edu.edgekey.net
    Type: CNAME (Canonical NAME for an alias) (5)
    Class: IN (0x0001)
    Time to live: 60 (1 minute)
    Data length: 27
    CNAME: e9566.dscb.akamaiedge.net
  e9566.dscb.akamaiedge.net: type A, class IN, addr 184.31.129.206
    Name: e9566.dscb.akamaiedge.net
    Type: A (Host Address) (1)
    Class: IN (0x0001)
    Time to live: 20 (20 seconds)
    Data length: 4
    Address: 184.31.129.206
[Request In: 1279]
[Time: 0.026266000 seconds]
```

16. To what IP address is the DNS query message sent? Is this the IP address of your default local DNS server?

```
C:\Users\andre>nslookup -type=NS mit.edu
Server: h268a
Address: 192.168.1.1

Non-authoritative answer:
mit.edu nameserver = eur5.akam.net
mit.edu nameserver = asia2.akam.net
mit.edu nameserver = ns1-37.akam.net
mit.edu nameserver = ns1-173.akam.net
mit.edu nameserver = asia1.akam.net
mit.edu nameserver = use5.akam.net
mit.edu nameserver = use2.akam.net
mit.edu nameserver = usw2.akam.net

eur5.akam.net    internet address = 23.74.25.64
use2.akam.net    internet address = 96.7.49.64
use5.akam.net    internet address = 2.16.40.64
asia1.akam.net   internet address = 95.100.175.64
asia2.akam.net   internet address = 95.101.36.64
use5.akam.net    AAAA IPv6 address = 2600:1403:a::40
```

The image shows a Wireshark packet capture of a DNS query. The packet list on the left shows a DNS Standard query from 192.168.1.24 to 192.168.1.1. The packet details on the right show the query for 'mit.edu' with type NS and class IN. The packet bytes on the right show the raw data of the query.

No.	Time	Source	Destination	Protocol	Length	Info
23	1.743085	192.168.1.24	192.168.1.1	DNS	67	Standard query 0x0002 NS mit.edu
24	1.770691	192.168.1.1	192.168.1.24	DNS	342	Standard query response 0x0002 NS mit.edu NS eur5.akam.net NS asia2.akam.net
26	2.788957	192.168.1.24	184.150.157.58	TCP	66	52598 → 443 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK_PERM
27	2.819107	184.150.157.58	192.168.1.24	TCP	66	443 → 52598 [SYN, ACK] Seq=0 Ack=1 Win=65535 Len=0 MSS=1460 SACK_PERM W
28	2.810135	192.168.1.24	184.150.157.58	TCP	64	52598 → 443 [ACK] Seq=1 Ack=1 Win=131328 Len=0

Frame 23: 67 bytes on wire (536 bits), 67 bytes captured (536 bits) on interface \Device\NPF{...}

Ethernet II, Src: ASUSTekC_cc:70:1e (f0:2f:74:cc:70:1e), Dst: zte_03:d4:cc (d0:60:8c:03:d4:cc)

Internet Protocol Version 4, Src: 192.168.1.24, Dst: 192.168.1.1

User Datagram Protocol, Src Port: 49920, Dst Port: 53

Domain Name System (query)

Transaction ID: 0x0002

Flags: 0x0100 Standard query

Questions: 1

Answer RRs: 0

Authority RRs: 0

Additional RRs: 0

Queries

[Response In: 24]

The DNS query message was sent to my default local DNS server with IP address 192.168.1.1.

17. Examine the DNS query message. What “Type” of DNS query is it? Does the query message contain any “answers”?

The image shows the details of the DNS query message. The query type is NS and the class is IN. The query is for 'mit.edu'.

Queries

mit.edu type NS, class IN

Name: mit.edu

[Name Length: 7]

[Label Count: 2]

Type: NS (authoritative Name Server) (2)

Class: IN (0x0001)

[Response In: 24]

It is of type NS and it has no answers.

18. Examine the DNS response message. What MIT nameservers does the response message provide? Does this response message also provide the IP addresses of the MIT namesers?

The screenshot displays a network packet capture analysis of a DNS query and response. The top section shows the packet details, including the source and destination IP addresses, the protocol (DNS), and the query type (Standard query response). The main body of the packet is divided into sections: Authority RRs, Additional RRs, Queries, and Answers. The Answers section contains several records, including NS records for 'mit.edu' and A records for 'eur5.akam.net', 'use2.akam.net', 'use5.akam.net', 'asia1.akam.net', and 'asia2.akam.net'. Handwritten red annotations highlight the 'NS' and 'IP' sections. The bottom section shows the time taken for the request (0.027606000 seconds).

No.	Time	Source	Destination	Protocol	Length	Info
24	1.770691	192.168.1.1	192.168.1.24	DNS	342	Standard query response 0x0002 NS mit.edu NS eur5.akam.net NS asia2.akam.net
26	2.788957	192.168.1.24	184.150.157.58	TCP	66	52598 → 443 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK_PERM

Authority RRs: 0
Additional RRs: 6

Queries

Answers

- mit.edu: type NS, class IN, ns eur5.akam.net
- mit.edu: type NS, class IN, ns asia2.akam.net
- mit.edu: type NS, class IN, ns ns1-37.akam.net
- mit.edu: type NS, class IN, ns ns1-173.akam.net
- mit.edu: type NS, class IN, ns asia1.akam.net
- mit.edu: type NS, class IN, ns use5.akam.net
- mit.edu: type NS, class IN, ns use2.akam.net
- mit.edu: type NS, class IN, ns usw2.akam.net

Additional records

- eur5.akam.net: type A, class IN, addr 23.74.25.64
- use2.akam.net: type A, class IN, addr 96.7.49.64
- use5.akam.net: type A, class IN, addr 2.16.40.64
- asia1.akam.net: type A, class IN, addr 95.100.175.64
- asia2.akam.net: type A, class IN, addr 95.101.36.64
- use5.akam.net: type AAAA, class IN, addr 2600:1403:a::40

[Request In: 23]
[Time: 0.027606000 seconds]

Handwritten red annotations: A red arrow points to the top of the packet list. A red bracket labeled 'NS' highlights the NS records in the Answers section. A red bracket labeled 'IP' highlights the A records in the Additional records section.

The name servers are: eur5.akam.net asia2.akam.net ns1-37.akam.net ns1-173.akam.net asia1.akam.net use5.akam.net use2.akam.net usw2.akam.net. Their corresponding IP addresses are on “Additional records” sections.

19. Provide a screenshot.

No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000	162.159.136...	192.168.1.24	TLV1.2	122	Application Data
2	0.052550	192.168.1.24	162.159.136.234	TCP	54	52284 → 443 [ACK] Seq=1 Ack=69 Win=513 Len=0
5	1.092540	192.168.1.24	13.68.233.9	TCP	1494	52587 → 443 [ACK] Seq=1 Ack=1 Win=516 Len=1440 [TCP segment of a reasse
6	1.092540	192.168.1.24	13.68.233.9	TLV1.2	220	Application Data
7	1.092562	192.168.1.24	13.68.233.9	TCP	1494	52587 → 443 [ACK] Seq=1607 Ack=1 Win=516 Len=1440 [TCP segment of a reas
8	1.092562	192.168.1.24	13.68.233.9	TLV1.2	117	Application Data
9	1.133996	13.68.233.9	192.168.1.24	TCP	60	443 → 52587 [ACK] Seq=1 Ack=3047 Win=16385 Len=0
10	1.136210	13.68.233.9	192.168.1.24	TCP	60	443 → 52587 [ACK] Seq=1 Ack=3110 Win=16384 Len=0
11	1.139579	13.68.233.9	192.168.1.24	TCP	1514	443 → 52587 [ACK] Seq=1 Ack=3110 Win=16384 Len=1460 [TCP segment of a re
12	1.140198	13.68.233.9	192.168.1.24	TLV1.2	645	Application Data
13	1.140210	192.168.1.24	13.68.233.9	TCP	54	52587 → 443 [ACK] Seq=3110 Ack=2052 Win=517 Len=0
14	1.142772	192.168.1.24	13.68.233.9	TCP	1494	52587 → 443 [ACK] Seq=3110 Ack=2052 Win=517 Len=1440 [TCP segment of a
15	1.142772	192.168.1.24	13.68.233.9	TLV1.2	207	Application Data
16	1.171717	13.68.233.9	192.168.1.24	TCP	60	443 → 52587 [ACK] Seq=2052 Ack=4703 Win=16385 Len=0
17	1.238902	13.68.233.9	192.168.1.24	TLV1.2	428	Application Data
18	1.289239	192.168.1.24	13.68.233.9	TCP	54	52587 → 443 [ACK] Seq=4703 Ack=2426 Win=516 Len=0
21	1.740141	192.168.1.24	192.168.1.1	DNS	84	Standard query 0x0001 PTR 1.1.168.192.in-addr.arpa
22	1.741054	192.168.1.1	192.168.1.24	DNS	141	Standard query response 0x0001 PTR 1.1.168.192.in-addr.arpa PTR h268a P
23	1.743085	192.168.1.24	192.168.1.1	DNS	67	Standard query 0x0002 NS mit.edu
24	1.770691	192.168.1.1	192.168.1.24	DNS	342	Standard query response 0x0002 NS mit.edu NS eur5.akam.net NS asia2.akar
26	2.788957	192.168.1.24	184.150.157.58	TCP	66	52598 → 443 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK_PERM
27	2.819107	184.150.157...	192.168.1.24	TCP	66	443 → 52598 [SYN, ACK] Seq=1 Win=65535 Len=0 MSS=1460 SACK_PERM W
28	2.819135	192.168.1.24	184.150.157.58	TCP	54	52598 → 443 [ACK] Seq=1 Ack=1 Win=13132 Len=0

> Frame 23: 67 bytes on wire (536 bits), 67 bytes captured (536 bits) on interface \Device\NPF...

> Ethernet II, Src: ASUSTekC_cc:70:1e (f0:2f:74:cc:70:1e), Dst: zte_03:d4:cc (d0:60:8c:03:d4:cc)

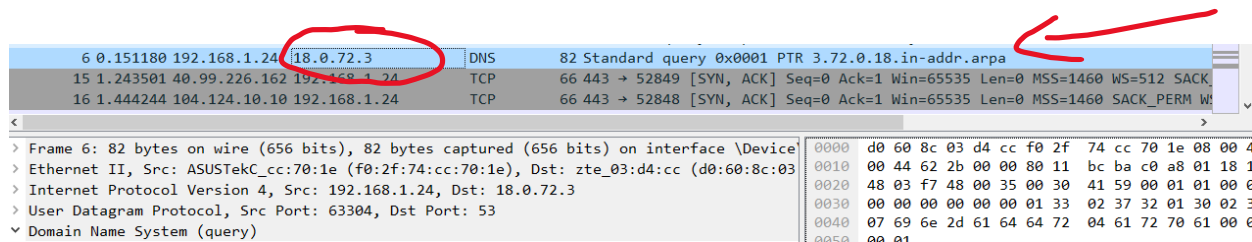
> Internet Protocol Version 4, Src: 192.168.1.24, Dst: 192.168.1.1

> User Datagram Protocol, Src Port: 49920, Dst Port: 53

> Domain Name System (query)

0000	d0 60 8c 03 d4 cc 70 1e	08 00 00 00 00 00 00 00
0010	00 35 d2 06 00 00 00 11	e5 47 00 a8 01 18 c
0020	01 01 c3 00 00 35 00 21	e0 4d 00 02 01 00 0
0030	00 00 00 00 00 00 03 6d	69 74 03 65 64 75 0
0040	02 00 01	

20. To what IP address is the DNS query message sent? Is this the IP address of your default local DNS server? If not, what does the IP address correspond to?

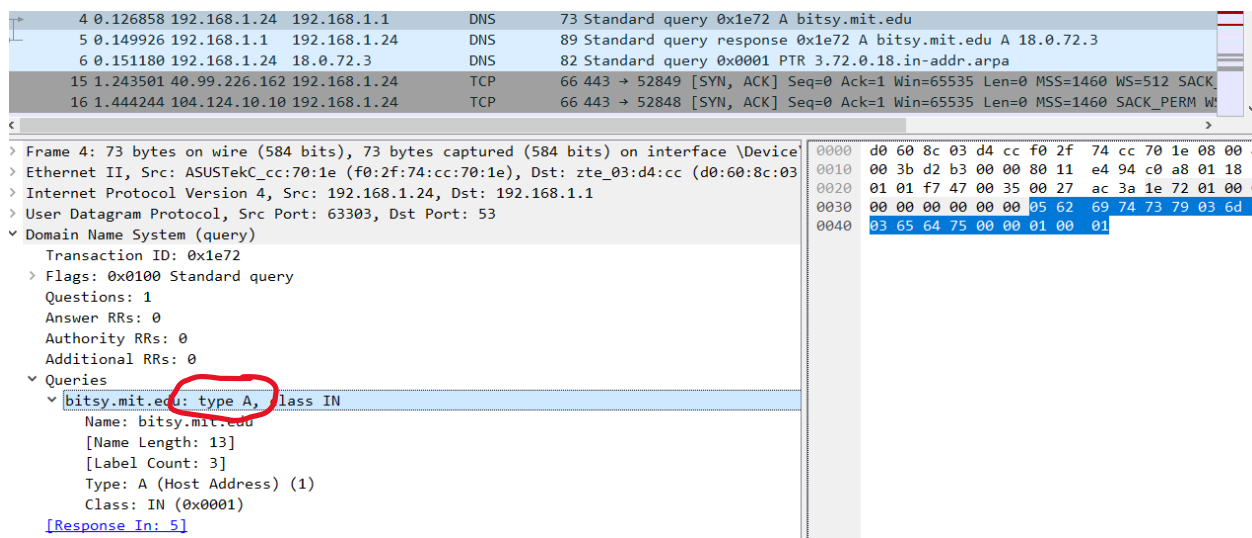


6	0.151180	192.168.1.24	18.0.72.3	DNS	82	Standard query 0x0001 PTR 3.72.0.18.in-addr.arpa
15	1.243501	40.99.226.162	192.168.1.24	TCP	66	443 → 52849 [SYN, ACK] Seq=0 Ack=1 Win=65535 Len=0 MSS=1460 WS=512 SACK
16	1.444244	104.124.10.10	192.168.1.24	TCP	66	443 → 52848 [SYN, ACK] Seq=0 Ack=1 Win=65535 Len=0 MSS=1460 SACK_PERM W

> Frame 6: 82 bytes on wire (656 bits), 82 bytes captured (656 bits) on interface \Device
 > Ethernet II, Src: ASUSTekC_cc:70:1e (f0:2f:74:cc:70:1e), Dst: zte_03:d4:cc (d0:60:8c:03
 > Internet Protocol Version 4, Src: 192.168.1.24, Dst: 18.0.72.3
 > User Datagram Protocol, Src Port: 63304, Dst Port: 53
 > Domain Name System (query)

It was sent to 18.0.72.3 which is not my default DNS server. The IP corresponds to bitsy.mit.edu.

21. Examine the DNS query message. What “Type” of DNS query is it? Does the query message contain any “answers”?



4	0.126858	192.168.1.24	192.168.1.1	DNS	73	Standard query 0x1e72 A bitsy.mit.edu
5	0.149926	192.168.1.1	192.168.1.24	DNS	89	Standard query response 0x1e72 A bitsy.mit.edu A 18.0.72.3
6	0.151180	192.168.1.24	18.0.72.3	DNS	82	Standard query 0x0001 PTR 3.72.0.18.in-addr.arpa
15	1.243501	40.99.226.162	192.168.1.24	TCP	66	443 → 52849 [SYN, ACK] Seq=0 Ack=1 Win=65535 Len=0 MSS=1460 WS=512 SACK
16	1.444244	104.124.10.10	192.168.1.24	TCP	66	443 → 52848 [SYN, ACK] Seq=0 Ack=1 Win=65535 Len=0 MSS=1460 SACK_PERM W

> Frame 4: 73 bytes on wire (584 bits), 73 bytes captured (584 bits) on interface \Device
 > Ethernet II, Src: ASUSTekC_cc:70:1e (f0:2f:74:cc:70:1e), Dst: zte_03:d4:cc (d0:60:8c:03
 > Internet Protocol Version 4, Src: 192.168.1.24, Dst: 192.168.1.1
 > User Datagram Protocol, Src Port: 63303, Dst Port: 53
 > Domain Name System (query)
 Transaction ID: 0x1e72
 > Flags: 0x0100 Standard query
 Questions: 1
 Answer RRs: 0
 Authority RRs: 0
 Additional RRs: 0
 > Queries
 bitsy.mit.edu: type A, class IN
 Name: bitsy.mit.edu
 [Name Length: 13]
 [Label Count: 3]
 Type: A (Host Address) (1)
 Class: IN (0x0001)
 [Response In: 5]

It is of type A and it contains no answers.

22. Examine the DNS response message. How many “answers” are provided? What does each of these answers contain?

Packet 5: 89 bytes on wire (712 bits), 89 bytes captured (712 bits) on interface \Device\NPF...

Ethernet II, Src: zte_03:d4:cc (d0:60:8c:03:d4:cc), Dst: ASUSTekC_cc:70:1e (f0:2f:74:cc:70:1e)

Internet Protocol Version 4, Src: 192.168.1.1, Dst: 192.168.1.24

User Datagram Protocol, Src Port: 53, Dst Port: 63303

Domain Name System (response)

Transaction ID: 0x1e72

Flags: 0x8180 Standard query response, No error

Questions: 1

Answer RRs: 1

Authority RRs: 0

Additional RRs: 0

Queries

Answers

- bitsy.mit.edu: type A, class IN, addr 18.0.72.3
 - Name: bitsy.mit.edu
 - Type: A (Host Address) (1)
 - Class: IN (0x0001)
 - Time to live: 1800 (30 minutes)
 - Data length: 4
 - Address: 18.0.72.3

[Request In: 4]

[Time: 0.023068000 seconds]

The DNS response message has one answer provided. It contains the name, type, class, time to live, data length and address.

23. Provide a screenshot.

Filter: ip.addr == 192.168.1.24

Packet 4: 73 bytes on wire (584 bits), 73 bytes captured (584 bits) on interface \Device\NPF...

Ethernet II, Src: ASUSTekC_cc:70:1e (f0:2f:74:cc:70:1e), Dst: zte_03:d4:cc (d0:60:8c:03:d4:cc)

Internet Protocol Version 4, Src: 192.168.1.24, Dst: 192.168.1.1

User Datagram Protocol, Src Port: 63303, Dst Port: 53

Domain Name System (query)

Transaction ID: 0x1e72

Flags: 0x0100 Standard query

Questions: 1

Answer RRs: 0

Authority RRs: 0

Additional RRs: 0

Queries

- bitsy.mit.edu: type A, class IN, addr 18.0.72.3

[Response In: 5]

3. Concepts Learned from this Lab

With the Mininet assignment 1, students were able to work on the virtual machine Ubuntu and tests four topologies. They are the single, reversed, linear and tree topology. By working on coding a python topology code, we were also able to understand more about hosts, switches and controller which all ties together into a network. With the Wireshark assignment 2, students learned about the domain name system (DNS). By capturing packages on Wireshark, we could read queries and responses, sources and destinations IP addresses, DNS message types, DNS answers, etc. Overall, these two provided students with hands-on experience in topology and network traffic analysis of DNS.