CS211 Lab09
fa2022
streller/dutta

PROBLEM STATEMENT:

More wireshark packet dumps

CODE :

As usual, start X-windows and start up the wireshark program and the terminal program. This week we will be looking at the DNS and ICMP packet structures.

First DNS packets:

DNS Query Message Format

Introduction

This section will deal with the analysis of the DNS packets. This will allow us to see the way DNS messages are formatted and the options and variables they contain. To understand a protocol, you must understand the information the protocol carries from one host to another.

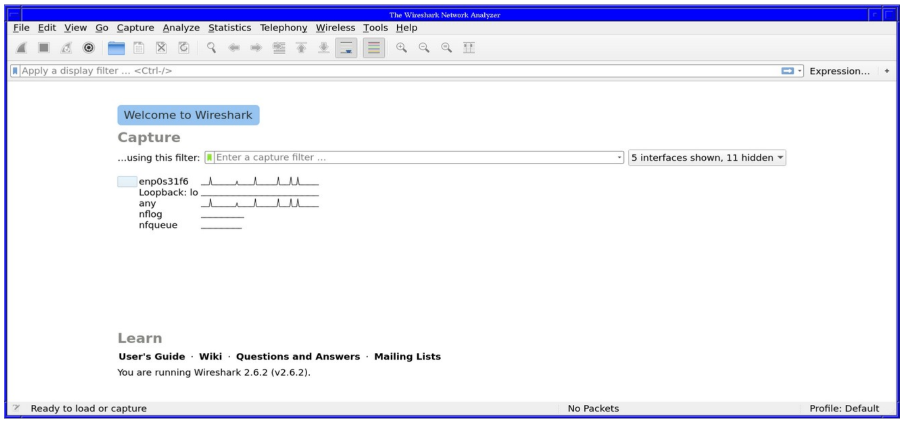
Because the DNS message format can vary, depending on the query and the response, I've broken this analysis into two parts. Part 1 analyzes the DNS format of a query, in other words, it shows how the packet looks when we ask a DNS server to resolve a domain. Part 2 analyzes the DNS format of an answer, where the DNS server is responding to our query.

DNS Analysis - Host Query

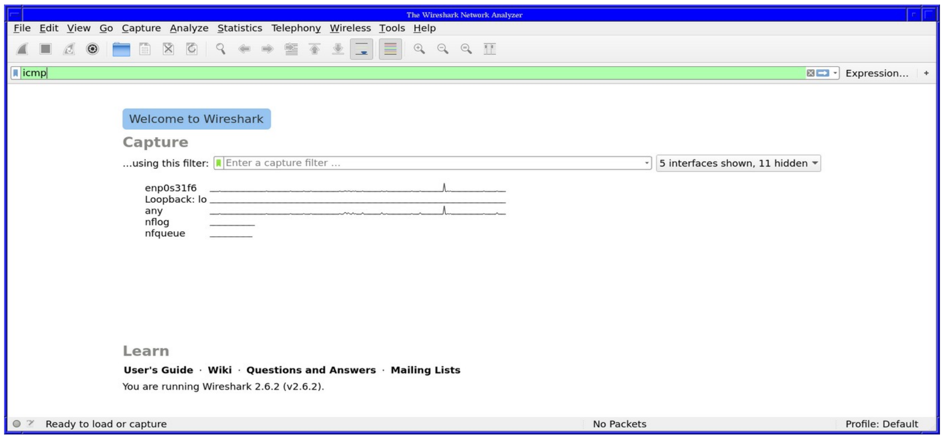
A DNS query is generated when the client needs to resolve a domain name into an IP Address. This could be the result of entering “ping www.usatoday.com” on the command line, then a DNS query is generated in order to successfully communicate with the host or server it needs.

Now, I've also included a live example (using wireshark), so you can compare theory with practice for a better understanding. After this we will have a look at the meaning of each field in the packet, so let's check out what a packet containing a DNS query would look like on our network:

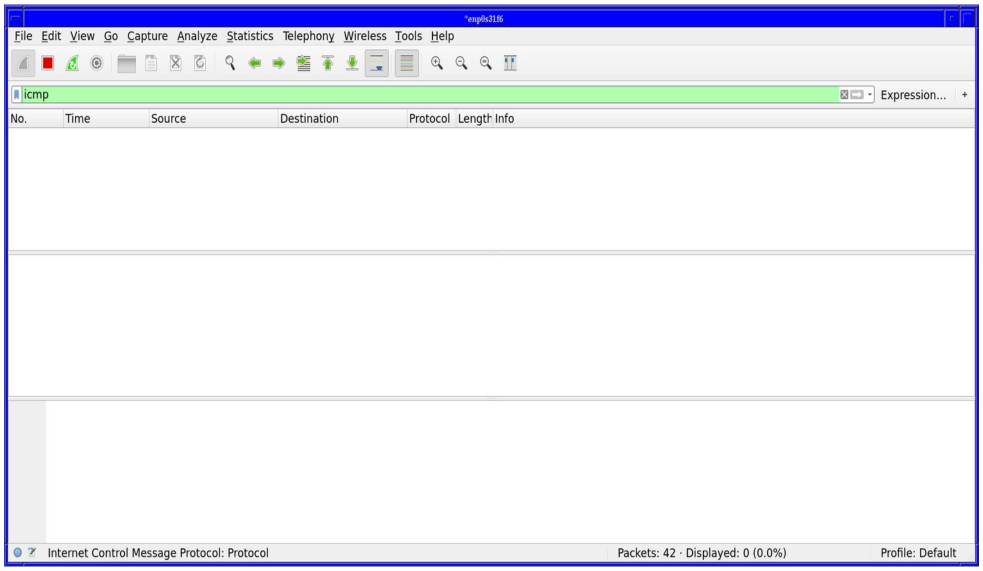
Start wireshark from the terminal (don’t forget to do sudo wireshark)
make sure the capture is your nic card; see below image


The Wireshark Network Analyzer
File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help
Apply a display filter ... <Ctrl-/>
Expression...
Welcome to Wireshark
Capture
...using this filter:
Enter a capture filter ...
5 interfaces shown, 11 hidden
enp0s31f6
Loopback: lo
any
nflog
nfqueue
Learn
User's Guide · Wiki · Questions and Answers · Mailing Lists
You are running Wireshark 2.6.2 (v2.6.2).
Ready to load or capture
No Packets
Profile: Default

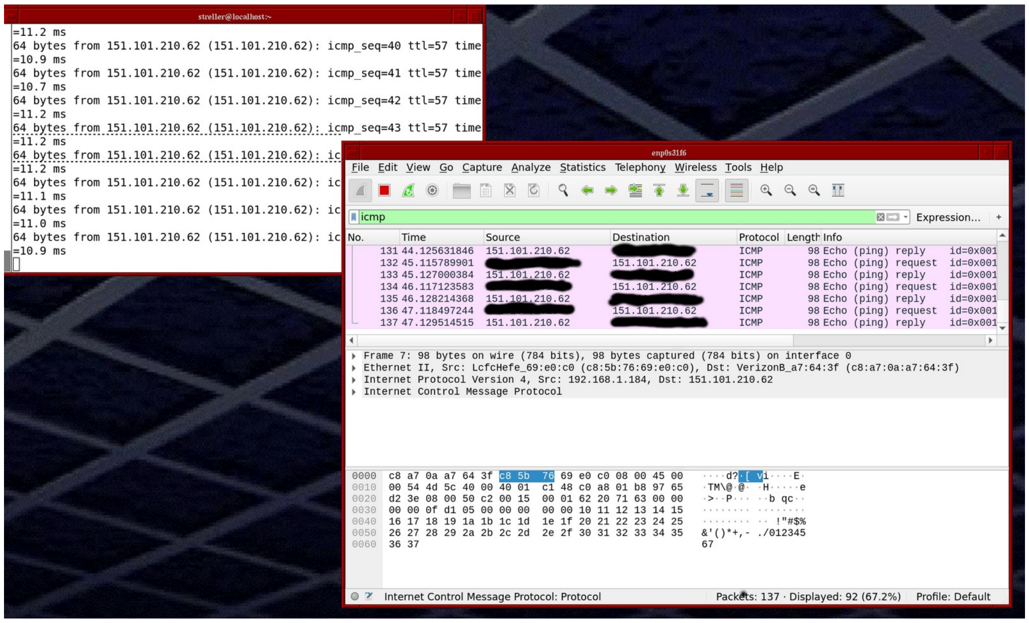
Then in the filer display window enter icmp


The Wireshark Network Analyzer
File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help
icmp
Expression…
Welcome to Wireshark
Capture
…using this filter:
Enter a capture filter …
5 interfaces shown, 11 hidden
enp0s31f6
Loopback: lo
any
nflog
nfqueue
Learn
User's Guide · Wiki · Questions and Answers · Mailing Lists
You are running Wireshark 2.6.2 (v2.6.2).

Ready to load or capture   No Packets   Profile: Default


*enp0s31f6
File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help
icmp
Expression…
No. Time Source Destination Protocol Length Info
Internet Control Message Protocol: Protocol   Packets: 42 · Displayed: 0 (0.0%)   Profile: Default

Then, from another terminal enter: ping   www.usatoday.com
Wireshark should immediately start capturing the packets
(if it doesn't, go to the Capture menu , and start the capture)


streller@localhost:~
enp0s31f6
File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help
icmp
Expression...

```
=11.2 ms
64 bytes from 151.101.210.62 (151.101.210.62): icmp_seq=40 ttl=57 time
=10.9 ms
64 bytes from 151.101.210.62 (151.101.210.62): icmp_seq=41 ttl=57 time
=10.7 ms
64 bytes from 151.101.210.62 (151.101.210.62): icmp_seq=42 ttl=57 time
=11.2 ms
64 bytes from 151.101.210.62 (151.101.210.62): icmp_seq=43 ttl=57 time
=11.2 ms
64 bytes from 151.101.210.62 (151.101.210.62): ic
=11.2 ms
64 bytes from 151.101.210.62 (151.101.210.62): ic
=11.1 ms
64 bytes from 151.101.210.62 (151.101.210.62): ic
=11.0 ms
64 bytes from 151.101.210.62 (151.101.210.62): ic
=10.9 ms
```

```
No.   Time          Source           Destination      Protocol Length Info
  131 44.125631846  151.101.210.62   [redacted]       ICMP         98 Echo (ping) reply    id=0x001
  132 45.115789901  [redacted]       151.101.210.62   ICMP         98 Echo (ping) request  id=0x001
  133 45.127000384  151.101.210.62   [redacted]       ICMP         98 Echo (ping) reply    id=0x001
  134 46.117123583  [redacted]       151.101.210.62   ICMP         98 Echo (ping) request  id=0x001
  135 46.128214368  151.101.210.62   [redacted]       ICMP         98 Echo (ping) reply    id=0x001
  136 47.118497244  [redacted]       151.101.210.62   ICMP         98 Echo (ping) request  id=0x001
  137 47.129514515  151.101.210.62   [redacted]       ICMP         98 Echo (ping) reply    id=0x001
```

```
▸ Frame 7: 98 bytes on wire (784 bits), 98 bytes captured (784 bits) on interface 0
▸ Ethernet II, Src: LcfcHefe_69:e0:c0 (c8:5b:76:69:e0:c0), Dst: VerizonB_a7:64:3f (c8:a7:0a:a7:64:3f)
▸ Internet Protocol Version 4, Src: 192.168.1.184, Dst: 151.101.210.62
▸ Internet Control Message Protocol
```

```
0000  c8 a7 0a a7 64 3f c8 5b  76 69 e0 c0 08 00 45 00   ····d?·[ vi····E·
0010  00 54 4d 5c 40 00 40 01  c1 48 c0 a8 01 b8 97 65   ·TM\@·@·  ·H····e
0020  d2 3e 08 00 50 c2 00 15  00 01 62 20 71 63 00 00   ·>··P···  ··b qc··
0030  00 00 0f d1 05 00 00 00  00 00 10 11 12 13 14 15   ········  ········
0040  16 17 18 19 1a 1b 1c 1d  1e 1f 20 21 22 23 24 25   ········  ·· !"#$%
0050  26 27 28 29 2a 2b 2c 2d  2e 2f 30 31 32 33 34 35   &'()*+,-  ./012345
0060  36 37                                              67
```

```
Internet Control Message Protocol: Protocol     Packets: 137 · Displayed: 92 (67.2%)     Profile: Default
```

Standard DNS Query A.   www.usatoday.com

Frame 3 (76 bytes on wire, 76 bytes captured)
Arrival Time: Oct 21, 2017 12:53:02.496287000
Time delta from previous packet: 0.000310000 seconds
Time relative to first packet: 0.000656000 seconds
Frame Number: 3
Packet Length: 76 bytes
Capture Length: 76 bytes
Ethernet II, Src: 00:04:75:ae:bd:ca, Dst: 00:10:4b:36:be:d3
Destination: 00:10:4b:36:be:d3 (3Com_36:be:d3)
Source: 00:04:75:ae:bd:ca (3_Com_ae:bd:ca)
Type: IP (0x0800)
Internet Protocol, Src Addr: 192.168.236.22 (192.168.236.22), Dst Addr: 128.205.1.2 (128.205.1.2)
Version: 4
Header length: 20 bytes
Differentiated Services Field: 0x00 (DSCP 0x00: Default; ECN: 0x00)
0000 00.. = Differentiated Services Codepoint: Default (0x00)
2 of 13
.... ..0. = ECN-Capable Transport (ECT): 0
....0 = ECN-CE: 0
Total Length: 62
Identification: 0xfaf8
Flags: 0x04
.1.. = Don't fragment: Set
..0. = More fragments: Not set
Fragment offset: 0
Time to live: 64
Protocol: UDP (0x11)
Header checksum: 0x1128 (correct)
Source: 192.168.236.22 (192.168.236.22)
Destination: 128.205.1.2 (128.205.1.2)
User Datagram Protocol, Src Port: 32770 (32770), Dst Port: domain (53)
Source port: 32770 (32770)
Destination port: domain (53)

```
Length: 42
Checksum: 0x502f (correct)
Domain Name System (query)
Transaction ID: 0xe698
Flags: 0x0100 (Standard query)
0... .... .... .... = Response: Message is a query
.000 0... .... .... = Opcode: Standard query (0)
.... ..0. .... .... = Truncated: Message is not truncated
.... ...1 .... .... = Recursion desired: Do query recursively
.... .... ...0 .... = Non-authenticated data OK: Non-authenticated data is unacceptable
Questions: 1
Answer RRs: 0
Authority RRs: 0
Additional RRs: 0
Queries
www.usatoday.com: type A, class inet
Name: www.usatoday.com
Type: Host address
Class: inet
```

This is the captured packet we are going to deal with.

To generate this packet, I typed "ping www.usatoday.com" from terminal prompt.

The command generated this packet, which was put on my network with the destination being a name server at UB ip #128.205.1.2(Dst Addr). Notice the Port Destination which is set to 53, the port on which DNS works and the protocol used for the DNS Query, which is UDP.

DNS Analysis - Host Query

Ethernet II is the most common type of frame found on LANs, in fact it probably is the only type you will find on 95% of all networks if you're only running TCP/IP and Windows or Unix-like machines. This particular one contains a DNS section, which could be either a Query or Response. We are assuming a Query, so it can fit nicely in our example.

We are going to take the DNS Section above and analyze its contents.

From this whole packet, the DNS Query Section is the part we're interested in (analyzed shortly), the rest is more or less overhead and information to let the server know a bit more information about our query.

```
Domain Name System (query)
Transaction ID: 0xe698
Flags: 0x0100 (Standard query)
```

```
0... .... .... .... = Response: Message is a query
.000 0... .... .... = Opcode: Standard query (0)
.... ..0. .... .... = Truncated: Message is not truncated
.... ...1 .... .... = Recursion desired: Do query recursively
.... .... ...0 .... = Non-authenticated data OK: Non-authenticated data is unacceptable
Questions: 1
Answer RRs: 0
Authority RRs: 0
Additional RRs: 0
Queries
www.usatoday.com: type A, class inet
Name: www.usatoday.com
Type: Host address
Class: inet
```

All fields in the DNS Query section except the DNS Name field (in bold), have set lengths. The DNS Name field has no set length because it varies depending on the domain name length.

The Flags/Parameters Field

The Parameter Field (labeled Flags) is one of the most important fields in DNS because it is responsible for letting the server or client know a lot of important information about the DNS packet. For example, it contains information as to whether the DNS packet is a query or response and, in the case of a query, if it should be a recursive or non-recursive type.

Let's have a closer look at the flags and explain the meaning of each one.

Flag/Parameter Fields

Bit No. Meaning
1 0 = query
1 = response

2-5 0000 = standard query
0100 = inverse (in-addr.arpa)
0010 & 0001 not used

6 0 = non-authoritative DNS answer
1 = authoritative DNS answer

7 0 = message not truncated
1 = message truncated

8 0 = non-recursive query
1 = recursive query

9 0 = recursion not available
1 = recursion available

10&12 reserved

11 0 = answer/authority portion was not authenticated by the server
1 = answer/authority was authenticated by the server

4 of 13
13-16 0000 = no error
0100 = format error in query
0010 = server failure
0001 = name does not exist

When you read the DNS response message format page, you will find a similar packet captured which is a response to the above query and the rest of the bits used are analyzed.

And that just about does it for the DNS Query message format page. Next up is the DNS Response message format page.

2nd DNS packet: (reply)

```
Frame 4 (206 bytes on wire, 206 bytes captured)
Arrival Time: Oct 21, 2017 12:53:02.507708000
Time delta from previous packet: 0.011421000 seconds
Time relative to first packet: 0.012077000 seconds
Frame Number: 4
Packet Length: 206 bytes
Capture Length: 206 bytes
Ethernet II, Src: 00:10:4b:36:be:d3, Dst: 00:04:75:ae:bd:ca
Destination: 00:04:75:ae:bd:ca (3_Com_ae:bd:ca)
Source: 00:10:4b:36:be:d3 (3Com_36:be:d3)
Type: IP (0x0800)
Internet Protocol, Src Addr: 128.205.1.2 (128.205.1.2), Dst Addr: 192.168.236.22
```

```
(192.168.236.22)
Version: 4
Header length: 20 bytes
Differentiated Services Field: 0x00 (DSCP 0x00: Default; ECN: 0x00)
0000 00.. = Differentiated Services Codepoint: Default (0x00)
.... ..0. = ECN-Capable Transport (ECT): 0
.... ...0 = ECN-CE: 0
Total Length: 192
Identification: 0xae1f
Flags: 0x04
.1.. = Don't fragment: Set
..0. = More fragments: Not set
Fragment offset: 0
Time to live: 247
Protocol: UDP (0x11)
Header checksum: 0xa67e (correct)
Source: 128.205.1.2 (128.205.1.2)
Destination: 192.168.236.22 (192.168.236.22)
User Datagram Protocol, Src Port: domain (53), Dst Port: 32770 (32770)
Source port: domain (53)
Destination port: 32770 (32770)
Length: 172
Checksum: 0xf81d (correct)
Domain Name System (response)
Transaction ID: 0xe698
Flags: 0x8180 (Standard query response, No error)
1... .... .... .... = Response: Message is a response
.000 0... .... .... = Opcode: Standard query (0)
.... .0.. .... .... = Authoritative: Server is not an authority for domain
.... ..0. .... .... = Truncated: Message is not truncated
.... ...1 .... .... = Recursion desired: Do query recursively
.... .... 1... .... = Recursion available: Server can do recursive queries
.... .... ..0. .... = Answer authenticated: Answer/authority portion was not authenticated by the
server
.... .... .... 0000 = Reply code: No error (0)
Questions: 1
Answer RRs: 1
```

```
Authority RRs: 3
Additional RRs: 3
Queries
www.usatoday.com: type A, class inet
Name: www.usatoday.com
Type: Host address
Class: inet
Answers
www.usatoday.com: type A, class inet, addr 66.54.32.240
Name: www.usatoday.com
Type: Host address
Class: inet
Time to live: 13 seconds
Data length: 4
Addr: 66.54.32.240
Authoritative nameservers
usatoday.com: type NS, class inet, ns ns01moc.usatoday.com
Name: usatoday.com
Type: Authoritative name server
Class: inet
Time to live: 4 minutes, 30 seconds
Data length: 10
Name server: ns01moc.usatoday.com
usatoday.com: type NS, class inet, ns ns02moc.usatoday.com
Name: usatoday.com
Type: Authoritative name server
Class: inet
Time to live: 4 minutes, 30 seconds
Data length: 10
Name server: ns02moc.usatoday.com
```

```
usatoday.com: type NS, class inet, ns ns02str.usatoday.com
Name: usatoday.com
Type: Authoritative name server
Class: inet
Time to live: 4 minutes, 30 seconds
Data length: 10
Name server: ns02str.usatoday.com
Additional records
ns01moc.usatoday.com: type A, class inet, addr 159.54.34.3
Name: ns01moc.usatoday.com
Type: Host address
Class: inet
Time to live: 4 minutes, 30 seconds
Data length: 4
Addr: 159.54.34.3
ns02moc.usatoday.com: type A, class inet, addr 159.54.34.5
Name: ns02moc.usatoday.com
Type: Host address
Class: inet
Time to live: 4 minutes, 30 seconds
Data length: 4
Addr: 159.54.34.5
ns02str.usatoday.com: type A, class inet, addr 209.97.62.22
Name: ns02str.usatoday.com
Type: Host address
Class: inet
Time to live: 4 minutes, 30 seconds
Data length: 4
Addr: 209.97.62.22
```

DNS Response Message Format

The previous packet dealt with the DNS Query message formats. We analyzed them in great detail and showed how various options are selected by the host using the Flags/Parameters field. 6 of 13
We will now analyze the response we get from the generated query. This response, in the case of a recursive query, came directly from the DNS server to which we sent the query and, in the case of a non-recursive query, will come from the last DNS server the client contacts in order to get the required information.

DNS Analysis - Server Response

Something worth paying attention to is the time this query took to come back to my Linux workstation. The time taken, from the moment the packet was sent from the Linux file server, until it received the answer, was only 0.011421000 seconds!
During this short period of time the packet traveled from Buffalo, NY to UB Amherst, reached the DNS server, which sent its queries to other DNS servers until it found the answer and then generated a DNS response that was sent back to Buffalo where the K236Stu?? network is !
There are a lot of factors that contribute to this fairly fast response. The transport protocol UDP, which does not require any 3-way handshake, the load of the DNS server to which I sent the query, the load of DNS servers it then had to ask, the speed at which all these servers and myself are connected to the Internet and the general load between the routers that my packet had to travel in order to get to its various destinations!
As you can clearly see, there is a lot happening for just one DNS query and response. Try to consider what happens when you have 20,000,000 DNS queries happening at once on the Internet and you have a good idea on how well this protocol and the underlying technology have been designed !
By comparing the two packets(i.e. query and response), you can see that there are fields in the DNS Response packet that didn't exist in the Query.
The DNS Section in a response packet is considerably larger and more complex than that of a query. For this reason we are going to analyze it in parts rather than all together. The query had only one section that required in-depth analysis whereas the response has three since the first one is the original query sent.

Domain Name System (response)

```
Transaction ID: 0xe698
Flags: 0x8180 (Standard query response, No error)
1... .... .... .... = Response: Message is a response
.000 0... .... .... = Opcode: Standard query (0)
.... .0.. .... .... = Authoritative: Server is not an authority for domain
.... ..0. .... .... = Truncated: Message is not truncated
.... ...1 .... .... = Recursion desired: Do query recursively
.... .... 1... .... = Recursion available: Server can do recursive queries
.... .... ..0. .... = Answer authenticated: Answer/authority portion was not authenticated by the
server
.... .... .... 0000 = Reply code: No error (0)
Questions: 1
Answer RRs: 1
Authority RRs: 3
Additional RRs: 3
Queries
www.usatoday.com: type A, class inet
Name: www.usatoday.com
Type: Host address
Class: inet
Answers
www.usatoday.com: type A, class inet, addr 66.54.32.240
Name: www.usatoday.com
Type: Host address
Class: inet
Time to live: 13 seconds
Data length: 4
Addr: 66.54.32.240
Authoritative nameservers
usatoday.com: type NS, class inet, ns ns01moc.usatoday.com
Name: usatoday.com
Type: Authoritative name server
Class: inet
Time to live: 4 minutes, 30 seconds
7 of 13
Data length: 10
Name server: ns01moc.usatoday.com
usatoday.com: type NS, class inet, ns ns02moc.usatoday.com
Name: usatoday.com
Type: Authoritative name server
Class: inet
Time to live: 4 minutes, 30 seconds
Data length: 10
Name server: ns02moc.usatoday.com
usatoday.com: type NS, class inet, ns ns02str.usatoday.com
Name: usatoday.com
Type: Authoritative name server
Class: inet
Time to live: 4 minutes, 30 seconds
Data length: 10
Name server: ns02str.usatoday.com
Additional records
ns01moc.usatoday.com: type A, class inet, addr 159.54.34.3
Name: ns01moc.usatoday.com
Type: Host address
Class: inet
Time to live: 4 minutes, 30 seconds
Data length: 4
Addr: 159.54.34.3
ns02moc.usatoday.com: type A, class inet, addr 159.54.34.5
Name: ns02moc.usatoday.com
Type: Host address
Class: inet
Time to live: 4 minutes, 30 seconds
Data length: 4
Addr: 159.54.34.5
ns02str.usatoday.com: type A, class inet, addr 209.97.62.22
Name: ns02str.usatoday.com
```

Type: Host address
Class: inet
Time to live: 4 minutes, 30 seconds
Data length: 4
Addr: 209.97.62.22

What we need to need understand is that each one of these three sections have identical fields(i.e. Answers, Authoritative nameservers and Additional records).
Even though the information they contain might seem a bit different, the fields are exactly the same.
For example, looking at line 1, part 1 in the Answers Section (in bold), you get a summary of what's to follow: www.usatoday.com: type A, class inet, addr 66.54.32.240

The Type Field

The Type field determines the type or part of information we require about a domain. To give you the simplest example, when we have a Type=Host Address , we are given the IP Address of the domain or host (look at Answers section above), whereas a Type= Authoritative name server means we are given the Authoritative Name Servers that are responsible for the domain (look at Authoritative Name Servers section above) in essence the ip address of www.usatoday.com is 66.54.32.240.

Now lets look ICMP.

ICMP stands for the Internet Control Message Protocol, and it was designed to send control messages between routers and hosts. For example, an ICMP packet may be sent when a router is experiencing congestion or when a destination host is unavailable.
An ICMP packet has a slightly different structure than we've seen before. An ICMP header follows the IP header in an IP packet, but it is not considered to be a Layer 4 header like TCP or UDP. Instead, ICMP is considered to be an integral part of IP; in fact, every vendor's implementation of IP is required to include ICMP.
Here is a picture of the fields an ICMP header adds to an IP packet:
8 of 13

8	16	32 bits
Type	Code	Checksum
Identifier		Sequence number
Data		

You'll note that an ICMP header is composed of six fields. Interestingly, the Data field does not contain the actual ICMP "message." Instead, the Type and the Code fields contain numeric values, and each numeric value represents a specific ICMP message. Every ICMP packet must have a Type value, but only some ICMP types have an associated non-zero Code value.

RFC 1700 contains the possible values for each ICMP type and code; I've summarized these into the following table:

Type	**Name**	**Code(s)**
0	Echo reply	0 – none
1	Unassigned	
2	Unassigned	
3	Destination unreachable	0 - Net unreachable
		1 - Host unreachable
		2 - Protocol unreachable
		3 - Port unreachable
		4 - Fragmentation needed and DF bit set
		5 - Source route failed
		6 - Destination network unknown
		7 - Destination host unknown
		8 - Source host isolated
		9 - Communication with destination network is administratively prohibited
		10 - Communication with destination host is administratively prohibited
		11 - Destination network unreachable for TOS
		12 - Destination host unreachable for TOS
4	Source quench	0 – none
5	Redirect	0 - Redirect datagram for the network
		1 - Redirect datagram for the host
		2 - Redirect datagram for the TOS and network
		3 - Redirect datagram for the TOS and host
6	Alternate host address	0 - Alternate address for host
7	Unassigned	
8	Echo	0 – None
9	Router advertisement	0 – None
10	Router selection	0 – None
11	Time Exceeded	0 – Time to live exceeded in transit
		1 - Fragment reassembly time exceeded
12	Parameter problem	0 - Pointer indicates the error
		1 - Missing a required option
		2 - Bad length
13	Timestamp	0 – None

14	Timestamp reply	0 – None
15	Information request	0 – None
16	Information reply	0 – None

9 of 13

17	Address mask request	0 – None
18	Address mask reply	0 – None
19	Reserved (for security)	
20-29	Reserved (for robustness experiment)	
30	Traceroute	
31	Datagram conversion error	
32	Mobile host redirect	
33	IPv6 where-are-you	
34	IPv6 I-am-here	
35	Mobile registration request	
36	Mobile registration reply	
37-255	Reserved	

You'll note that the ICMP types that do have associated codes use the Code field to further explain the message value in the Type field. For example, ICMP Type 3 represents "destination unreachable." There can be many reasons why a destination is unreachable; accordingly, every ICMP Type 3 packet will also use one of the codes to explain why the destination was unreachable. In our dump file, packets 5-6 contained ICMP information. These packets were created right after DNS had determined the destination ip address. Let's take a look at packets 5 and 6:

```
Frame 5 (98 bytes on wire, 98 bytes captured)
Arrival Time: Oct 21, 2017 12:53:02.508277000
Time delta from previous packet: 0.000569000 seconds
Time relative to first packet: 0.012646000 seconds
Frame Number: 5
Packet Length: 98 bytes
Capture Length: 98 bytes
Ethernet II, Src: 00:04:75:ae:bd:ca, Dst: 00:10:4b:36:be:d3
Destination: 00:10:4b:36:be:d3 (3Com_36:be:d3)
Source: 00:04:75:ae:bd:ca (3_Com_ae:bd:ca)
Type: IP (0x0800)
Internet Protocol, Src Addr: 192.168.236.22 (192.168.236.22), Dst Addr: 66.54.32.240 (66.54.32.240)
Version: 4
Header length: 20 bytes
Differentiated Services Field: 0x00 (DSCP 0x00: Default; ECN: 0x00)
0000 00.. = Differentiated Services Codepoint: Default (0x00)
.... ..0. = ECN-Capable Transport (ECT): 0
.... ...0 = ECN-CE: 0
Total Length: 84
Identification: 0x0000
Flags: 0x04
.1.. = Don't fragment: Set
..0. = More fragments: Not set
Fragment offset: 0
Time to live: 64
Protocol: ICMP (0x01)
Header checksum: 0x2ac4 (correct)
Source: 192.168.236.22 (192.168.236.22)
Destination: 66.54.32.240 (66.54.32.240)
Internet Control Message Protocol
Type: 8 (Echo (ping) request)
```

```
Code: 0
Checksum: 0xc587 (correct)
Identifier: 0xd60e
Sequence number: 00:01
Data (56 bytes)

0000 6e 64 95 3f 66 c1 07 00 08 09 0a 0b 0c 0d 0e 0f nd.?f...........
0010 10 11 12 13 14 15 16 17 18 19 1a 1b 1c 1d 1e 1f ................
```

```
0020 20 21 22 23 24 25 26 27 28 29 2a 2b 2c 2d 2e 2f !"#$%&'()*+,-./
0030 30 31 32 33 34 35 36 37 01234567


Frame 6 (98 bytes on wire, 98 bytes captured)
Arrival Time: Oct 21, 2017 12:53:02.524332000
Time delta from previous packet: 0.016055000 seconds
Time relative to first packet: 0.028701000 seconds
Frame Number: 6
Packet Length: 98 bytes
Capture Length: 98 bytes
Ethernet II, Src: 00:10:4b:36:be:d3, Dst: 00:04:75:ae:bd:ca
Destination: 00:04:75:ae:bd:ca (3_Com_ae:bd:ca)
Source: 00:10:4b:36:be:d3 (3Com_36:be:d3)
Type: IP (0x0800)
Internet Protocol, Src Addr: 66.54.32.240 (66.54.32.240), Dst Addr: 192.168.236.22 (192.168.236.22)
Version: 4
Header length: 20 bytes
Differentiated Services Field: 0x00 (DSCP 0x00: Default; ECN: 0x00)
0000 00.. = Differentiated Services Codepoint: Default (0x00)
.... ..0. = ECN-Capable Transport (ECT): 0
.... ...0 = ECN-CE: 0
Total Length: 84
Identification: 0x4669
Flags: 0x00
.0.. = Don't fragment: Not set
..0. = More fragments: Not set
Fragment offset: 0
Time to live: 237
Protocol: ICMP (0x01)
Header checksum: 0x775a (correct)
Source: 66.54.32.240 (66.54.32.240)
Destination: 192.168.236.22 (192.168.236.22)
Internet Control Message Protocol
Type: 0 (Echo (ping) reply)
Code: 0
Checksum: 0xcd87 (correct)
Identifier: 0xd60e
Sequence number: 00:01
Data (56 bytes)

0000 6e 64 95 3f 66 c1 07 00 08 09 0a 0b 0c 0d 0e 0f nd.?f...........
0010 10 11 12 13 14 15 16 17 18 19 1a 1b 1c 1d 1e 1f ................
0020 20 21 22 23 24 25 26 27 28 29 2a 2b 2c 2d 2e 2f !"#$%&'()*+,-./
0030 30 31 32 33 34 35 36 37 01234567
```

Notice that these are normal IP packets with the expected IP header fields. Immediately following the IP header is the ICMP header which is followed by some strange-looking data. You can see that Packet No. 5 was an echo-request and Packet No. 6 was an echo-reply. If we look up these names in the chart, we'll see that Packet 5 contains an ICMP Type 8 Code 0 message, and Packet 6 contains an ICMP Type 0 Code 0 message.

Notice that Wireshark interprets all of the ICMP fields, including the Type and Code numbers. It also indicates the name of the utility that issued these ICMP packets, "ping" packets were sent out to verify connectivity between my workstation and the site www.usatoady.com. The first ping packet contained the echo-request and it was followed by the desired echo-reply.

To summarize, whenever you run the ping utility, you will send out ICMP Type 8 Code 0 packets. Each packet will have the same identifier, but every packet's sequence number will be increased by 1 as your machine continues to ping. If you have connectivity to the other host, you should receive back ICMP Type 0 Code 0 packets with the same identifier. If you don't receive all the packets back in sequence, you don't have a very reliable connection. 11 of 13

You've probably used the ping utility yourself to test the connection between two hosts running TCP/IP; you may have not known that ping uses ICMP.

The traceroute utility is another utility that uses ICMP messages, but its usage is different from that of the ping utility. When you type traceroute hostname, three UDP packets are sent out with a TTL (time to live) value of 1. These three packets will arrive at the router closest to you which will decrease the TTL by one, meaning the TTL will now be 0. When routers notice a TTL of 0, they respond by sending an ICMP packet of Type 11 Code 0, or "time exceeded" as "time to live exceeded in transit." The traceroute utility will make note of the IP address of the router that sent back the three ICMP packets, calculate the time it took to receive each of the packets, then send out three more UDP packets, this time with a TTL of 2.
Because these packets have a TTL of 2, ICMP packets should be returned by the router that is two hops away from you. Once these packets are received and noted, traceroute sends out three more packets with a TTL of 3. The traceroute utility will continue this pattern until you either reach your final destination or you've gone through the default maximum of 30 routers.

The results will be sent to your screen like so:

```
/usr/sbin/traceroute -I  www.freebsd.org
```

(the switch -I is a capital I (eye))

```
traceroute to freefall.freebsd.org (216.136.204.21), 30 hops max, 40 byte packets
1 10.69.4.1 (10.69.4.1) 33.137 ms 110.654 ms 52.307 ms
2 d226-12-1.home.cgocable.net (24.226.12.1) 15.413 ms 36.285 ms 12.538 ms
3 cgowave-0-158.cgocable.net (24.226.0.158) 13.857 ms 14.130 ms 16.433 ms
4 cgowave-busy-core.cgocable.net (24.226.1.1) 15.304 ms 15.470 ms 14.940 ms
5 cgowave-0-202.cgocable.net (24.226.0.202) 16.681 ms 14.324 ms 16.357 ms
6 10.0.185.33 (10.0.185.33) 16.066 ms 15.919 ms 17.318 ms
7 c1-pos8-0.bflony1.home.net (24.7.74.29) 18.234 ms 18.063 ms 19.266 ms
8 c1-pos1-0.hrfrct1.home.net (24.7.65.253) 27.590 ms 25.213 ms 48.447 ms
9 c1-pos3-0.nycmny1.home.net (24.7.69.2) 32.722 ms 29.405 ms 29.724 ms
10 ibr02-p1-0.jrcy01.exodus.net (24.7.70.122) 31.728 ms 48.891 ms 29.017 ms
11 bbr02-g4-0.jrcy01.exodus.net (216.32.223.114) 37.117 ms 37.070 ms 62.180 ms
12 bbr01-p2-0.okbr01.exodus.net (216.32.132.109) 59.707 ms 40.090 ms 39.422 ms
13 bbr02-p3-0.sttl01.exodus.net (216.32.132.89) 142.048 ms 101.184 ms 86.259 ms
14 bbr01-g5-0.sttl01.exodus.net (216.32.29.19) 83.362 ms 83.433 ms 83.103 ms
15 bbr01-p1-0.tkwl01.exodus.net (209.185.9.66) 85.309 ms 123.174 ms 83.753 ms
16 bbr01-p4-0.sntc05.exodus.net (216.32.173.229) 88.995 ms 90.207 ms 88.723 ms
17 dcr01-g2-0.sntc05.exodus.net (64.56.192.3) 109.213 ms 90.418 ms 90.458 ms
18 g2-1.bas1-m.sc5.yahoo.com (64.56.207.146) 170.210 ms 164.354 ms 281.053 ms
19 freefall.freebsd.org (216.136.204.21) 91.146 ms 88.509 ms 91.049 ms
```

Note that the traceroute utility numbered each hop, gave the name and IP address of the associated router, and recorded the time it took to receive an ICMP response to each of the three UDP packets that were sent to each router.

The ping and traceroute utilities are the most common utilities used by users that involve the ICMP protocol. However, there is another ICMP type that you should be aware of as it can affect network performance if there are routers between you and your final destination.
What would happen if two hosts were not on the same LAN and their packets had to pass through a network that could only accept frames with a maximum transmission unit (MTU) size of 576 bytes? Because the two end hosts may have already agreed upon a segment size of 1,460 bytes, they would be creating their IP packets accordingly. When these IP packets arrive at the router, which is cabled to the network with the smaller MTU, it will have to re-package every packet into smaller segments that will fit into the smaller size frames of that network. The destination host will then have to reassemble all of the fragmented packets back into the

original agreed-upon sized segment. This creates more work and definitely slows things down. To help prevent this, TCP uses something called Path-MTU Discovery. TCP will send out IP packets using the agreed MSS(maximum segment size), but will set the DF (don't fragment) bit to 1. If this 12 of 13
packet is received by a router that needs to fragment the packet so that it will fit over a network that uses smaller-sized frames, the router will respond with an ICMP Type 3 Code 4 packet which translates to "destination unreachable as fragmentation needed" and "DF bit set." When the host receives this ICMP packet, it knows that it needs to start sending smaller packets.
The last ICMP type I'd like to cover is Source Quench, or ICMP Type 4 Code 0. This message is sent whenever a router is being overwhelmed by packets. It basically tells the host to slow down the rate it is sending packets so it can have a chance to deal with the packets it has already received. This is an important message -- if the host does not slow down its transmission rate, the router will run out of buffer space to store packets and will have to start throwing packets away. Every packet that is thrown away will have to be re-transmitted which will make the original situation worse.
The ICMP types we've covered do have implications when you start creating packet filter rules on your Linux system. I'd like to summarize the ICMP types and codes that we'll need to be mindful of:

ICMP Type	Code	Used By
0	0	Ping
3	4	Path-MTU Discovery
4	0	Source Quench
8	0	Ping
11	0	traceroute

Fill in the table by capturing packets using wireshark and pinging a host(external) in the terminal windows,
first type the command to ping a host in the terminal window, do not press the enter key yet, then start the packet capturing by activating the wireshark window clicking Capture, Start and OK, then in the terminal window press enter, after a few packet responses arrive click the stop capture button.

Examine the captured packets and fill in the table on the submission page.

<u>Deliverables:</u>

For the report for this lab, complete and turn in the following page.

DUE DATE 11:00am 22 November 2022

CS211
Lab9

Name: ___________________________

Complete the tables below from you data above

DNS Tables:

QUERY Packet		Response Packet
# of bytes captured		# of bytes captured
MAC address of source		MAC address of last router
MAC address of 1st router		MAC address of workstation
Source ip		Source ip
Source Port		Source Port
Destination Port		Destination Port
Name Server ip		Answer ip
Checksum value		Checksum value
Query for host		Authoritative Name Server (list 1 only)ASCII name and ip
		Time Since query packet

ICMP TABLE (for ping)

Echo Request		Echo Reply
# of bytes captured		# of bytes captured
MAC address of source		MAC address of source
MAC address of 1st router		MAC address of 1st router
Source ip		Source ip
Destination ip		Destination ip
Protocol		Protocol
Type		Type
Code		Code
Sequence Number		Sequence Number
Checksum value		Checksum value
		Time Since request packet