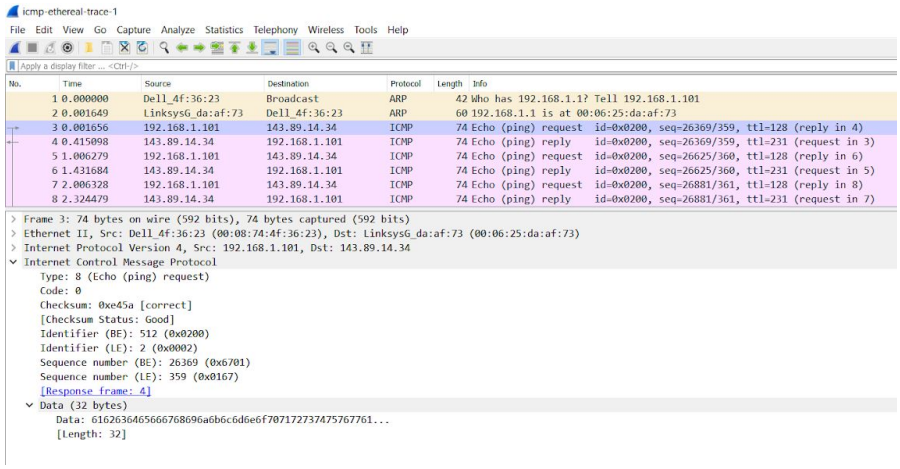
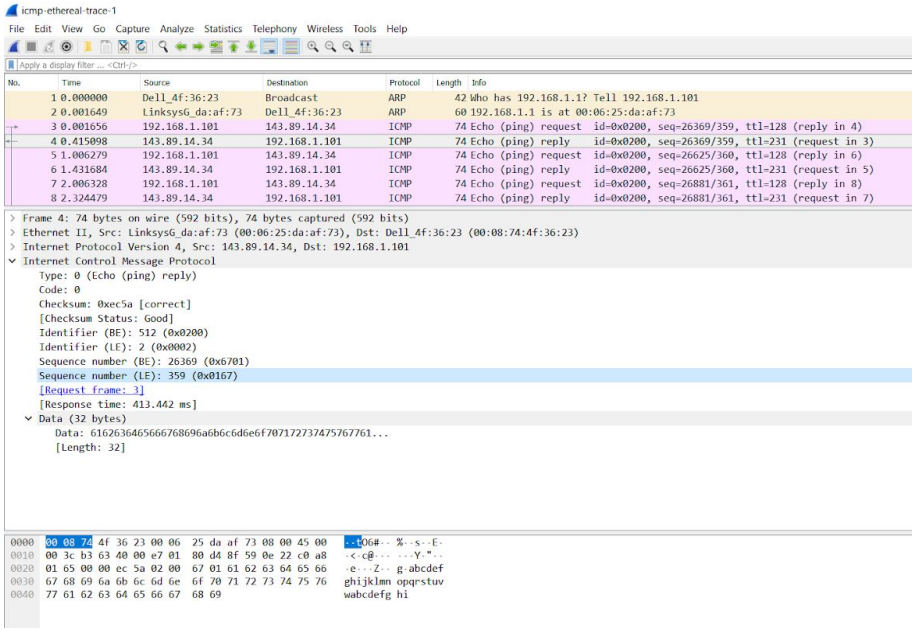
60-367
Lab 3
Ryan Lebeau, Calvin Moras
07/11/18

Part 1: ICMP

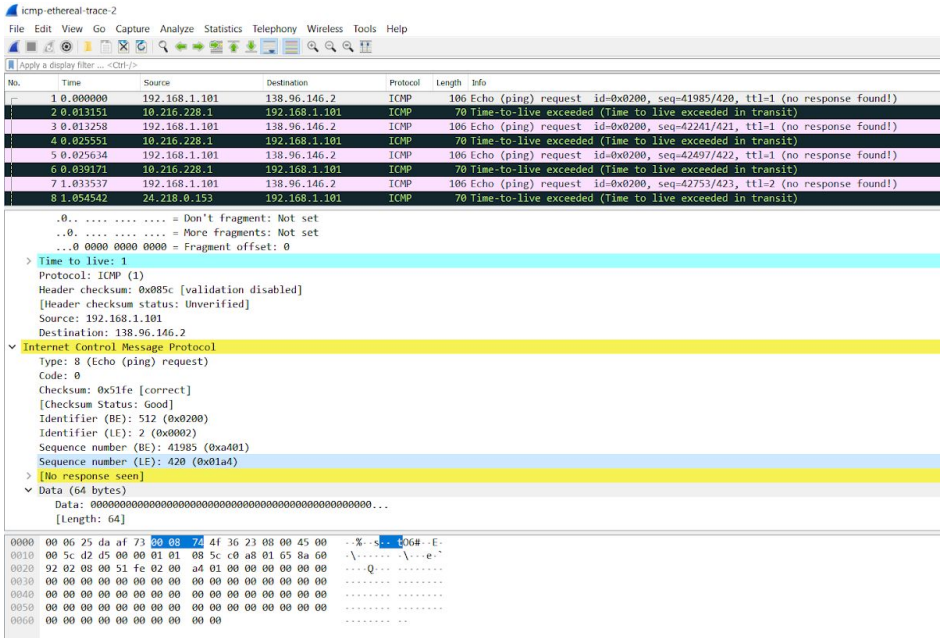
1. Host IP address is 192.168.1.101 and the IP address of the destination host is 143.89.14.34
2. There are no source and destination port numbers in an ICMP packet since it is designed to communicate between network layers which only require a Type and Code combination.
3. ICMP type numbers 8 and code number 0, there are also identifier (both BE and LE), sequence numbers (BE/LE), checksum field, and data fields. The checksum bytes are 2, the identifier bytes are 2, and sequence number bytes are 2.



4. The ICMP type number 0 and code number 0, there are also identifier (both BE and LE) and sequence numbers (BE/LE) , checksum field, and data fields. The checksum bytes are 2, the identifier bytes are 2, and sequence number bytes are 2.

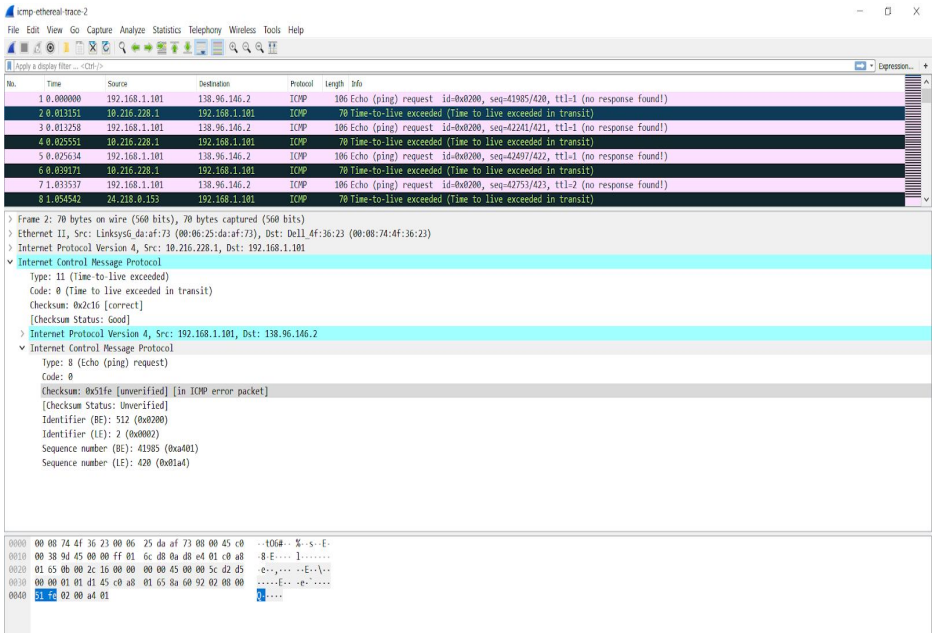


5. The IP address is 192.168.1.101 and the destination source IP is 138.96.146.2
6. If the ICMP sent UDP packets instead the protocol number would not be 01 it would be 0x11.

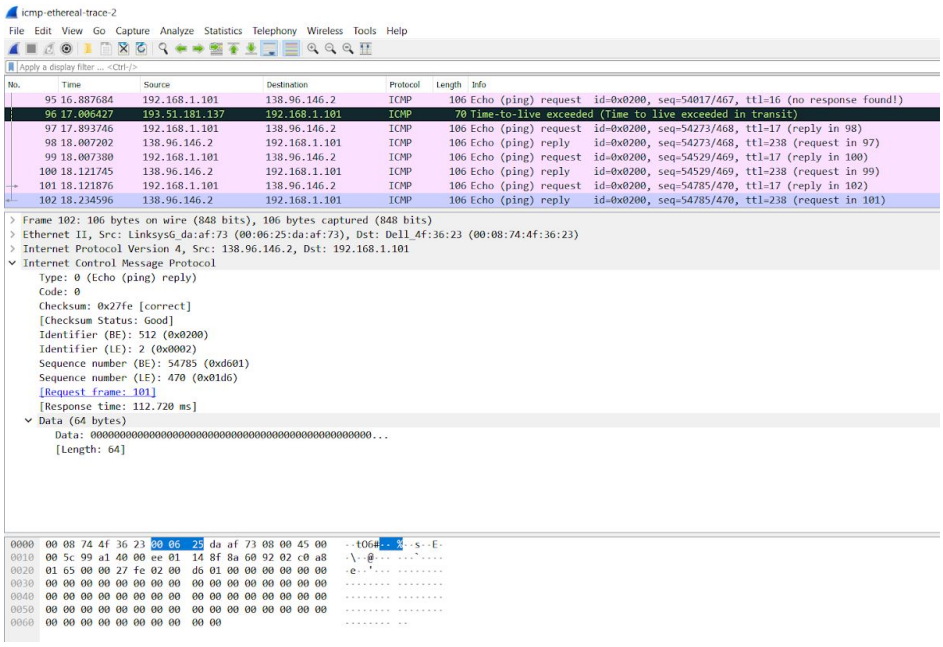


7. The ICMP echo packet is the same as the IP ping request.

8. The ICMP error packet contains the type, code, checksum, identifier, and sequence number of the original ICMP echo packet.

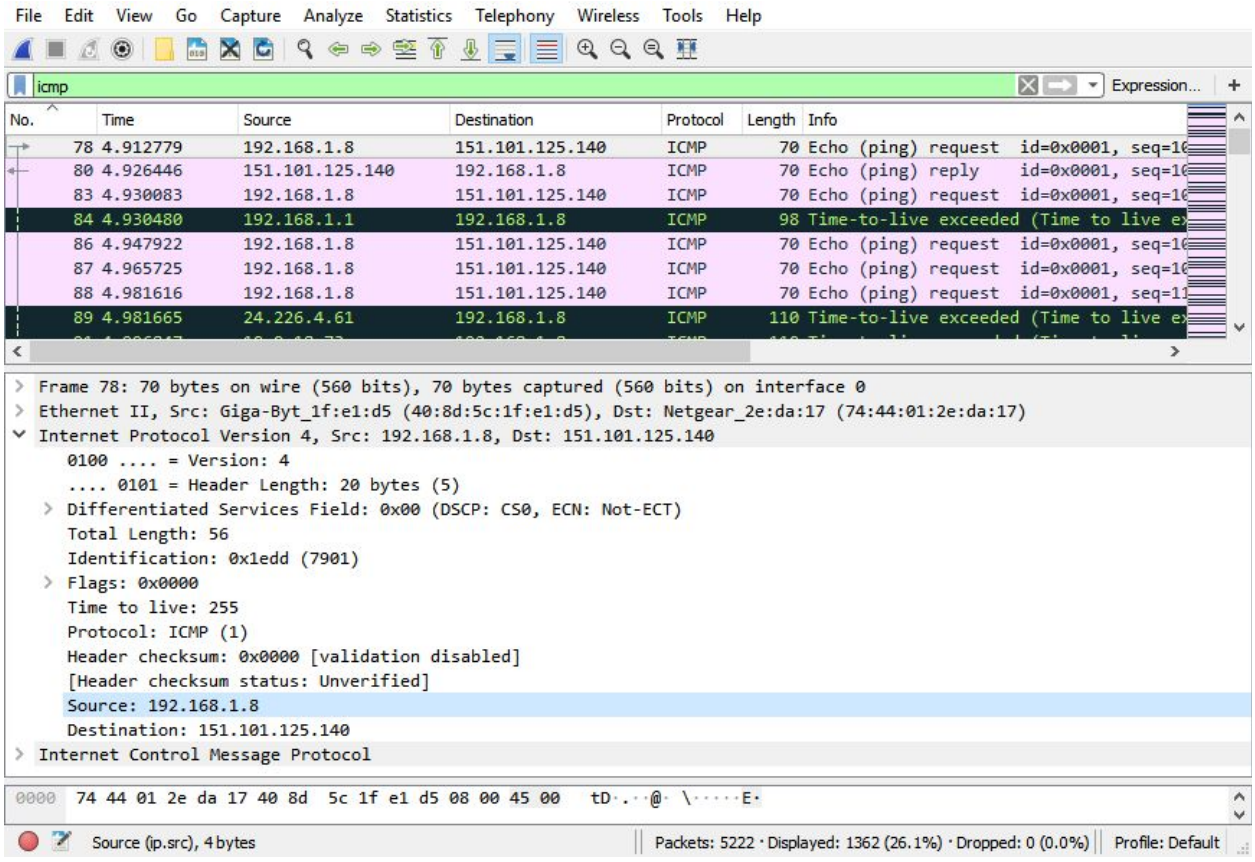


9. The last three ICMP packets are reply messages to the request packet instead of error packets, they have type number 0.



10. There is a significant longer link between step 11 and 12. The link is from New York to Aubervilliers, France.

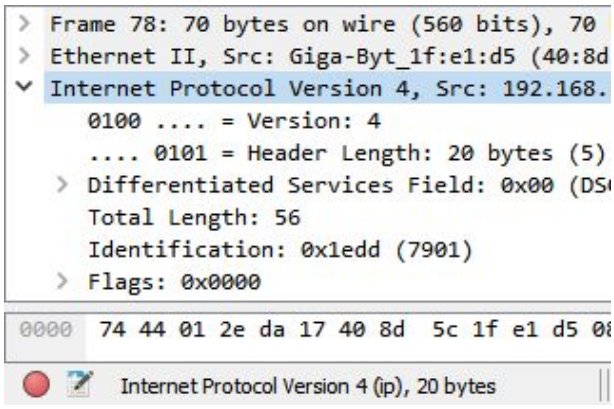
Part 2: IP

1. 

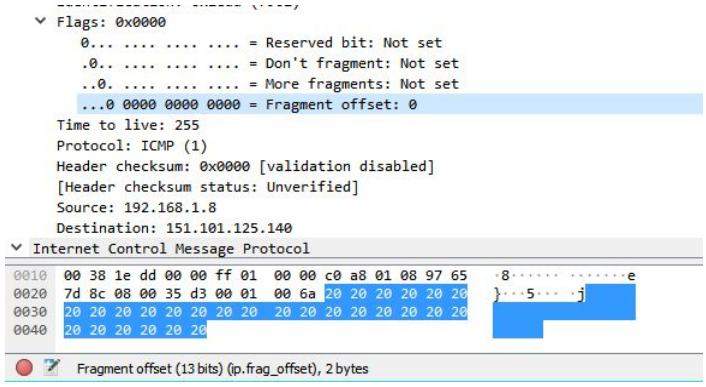
   The IP address of my computer is 192.168.1.8.

2. The value in the Protocol field of the IP Header is ICMP (1).
3. The IP header itself contains 20 bytes, the payload of the IP datagram is 36 bytes. This is because the total length field tells us the overall length is 56 bytes, so 56-20=36. This

can be verified by checking the length of the ICMP field in wireshark as well.

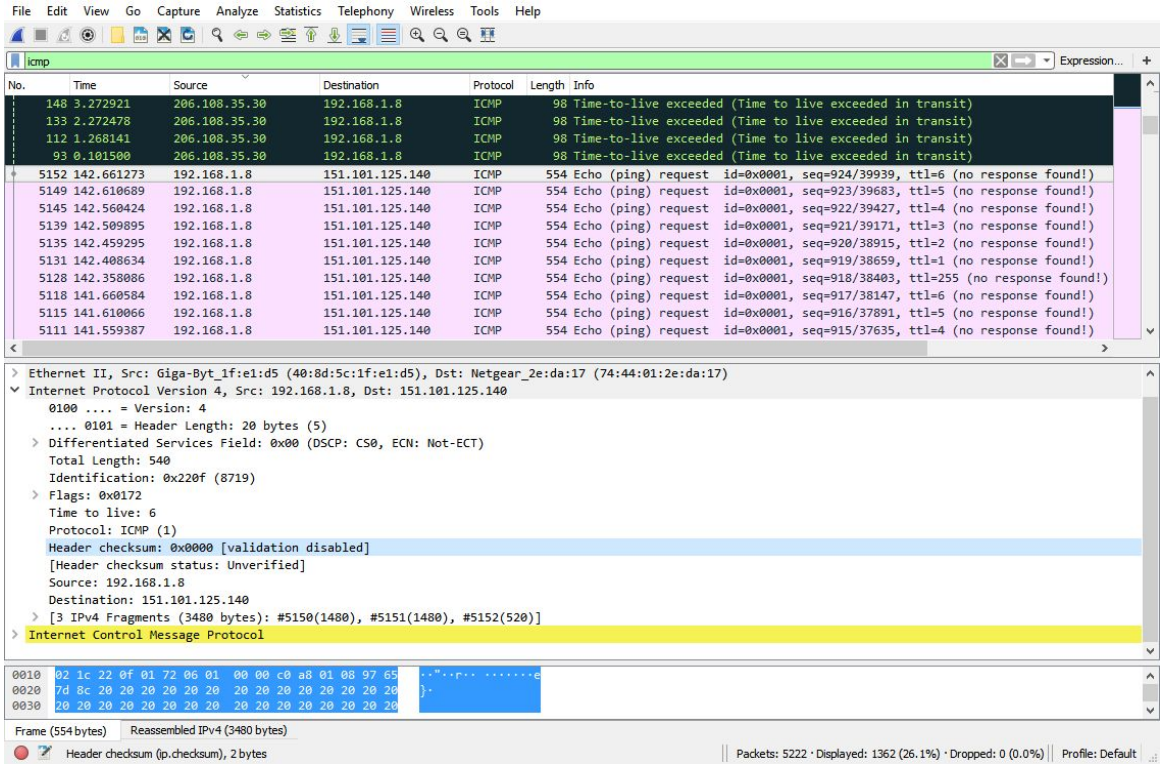


4. This IP datagram has not been fragmented. This can be seen by the fragment offset flag having a value of 0:



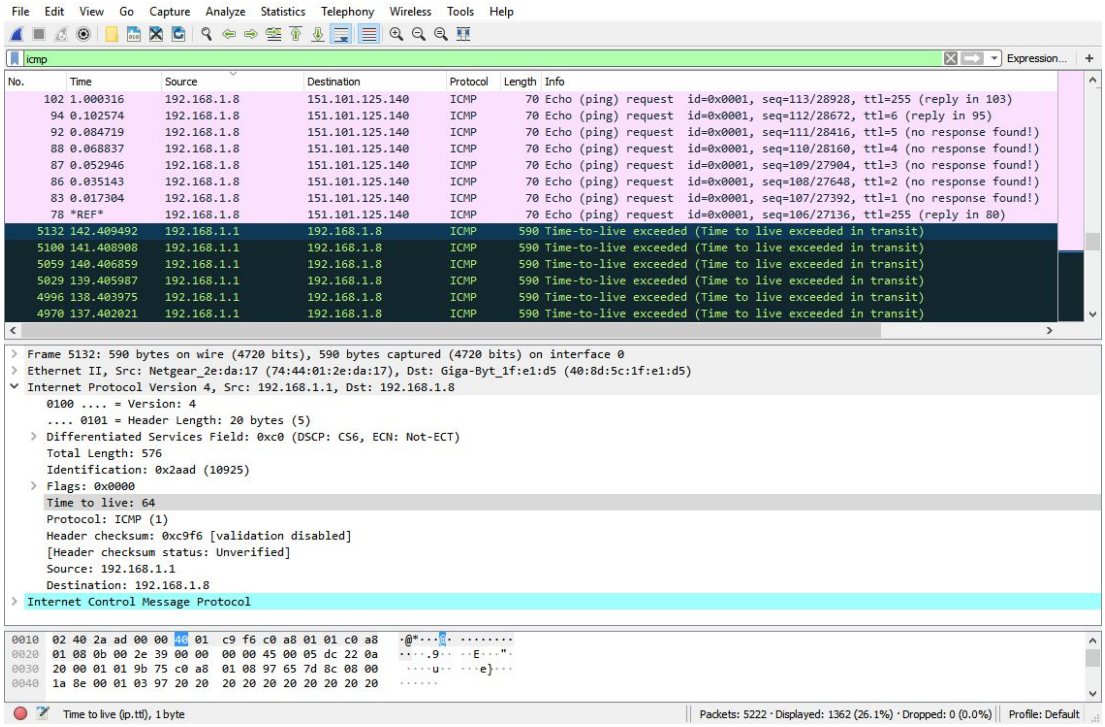
5. The only two values two change from one datagram to the next for me, were the Identification and Time to Live values. However it is worth noting that my installation of wireshark had checksum validation disabled by default, and had it been enabled at the

time of capture, the checksum of each header should have differed.



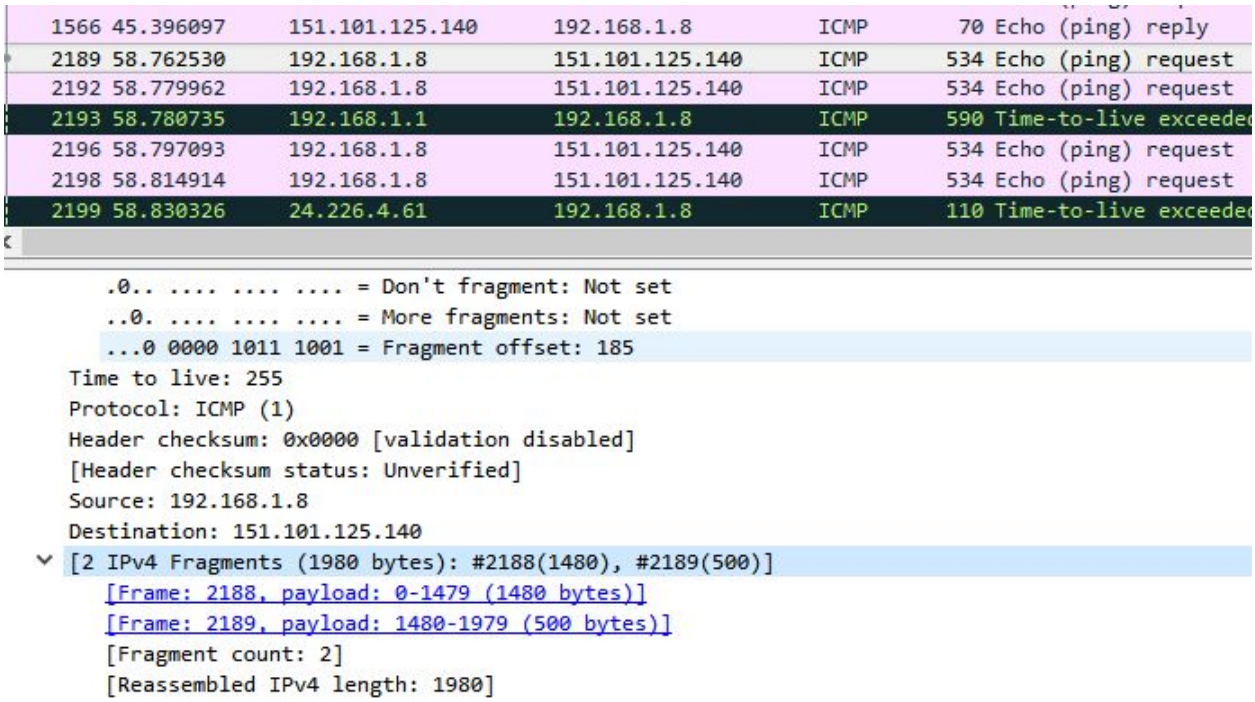
6. Source, Destination, Protocol / version, upper layer protocol, Differentiated Services Field (including sub-entries), and Header Length all remain constant. The source and destination must remain constant so long as the same two devices are communicating, both types of protocols must remain constant since IPV4 is being used and all packets are ICMP packets, which means they will all also use the same services. Identification must change so as to uniquely identify each packet, time to live changes due to the traceroute's incrementation of each packet, and once again although it did not on mine, due to the nature of checksums, the header checksum would change each time as well, as each packet's checksum is a product of the information its header contains.
7. Each identification number is a sequential hex value from the previous packet's identification number (these values are technically ascending each packet, we are just viewing in descending order).

8. The Identification field's value is 0x2aad (10925 in decimal). The TTL value is 64.



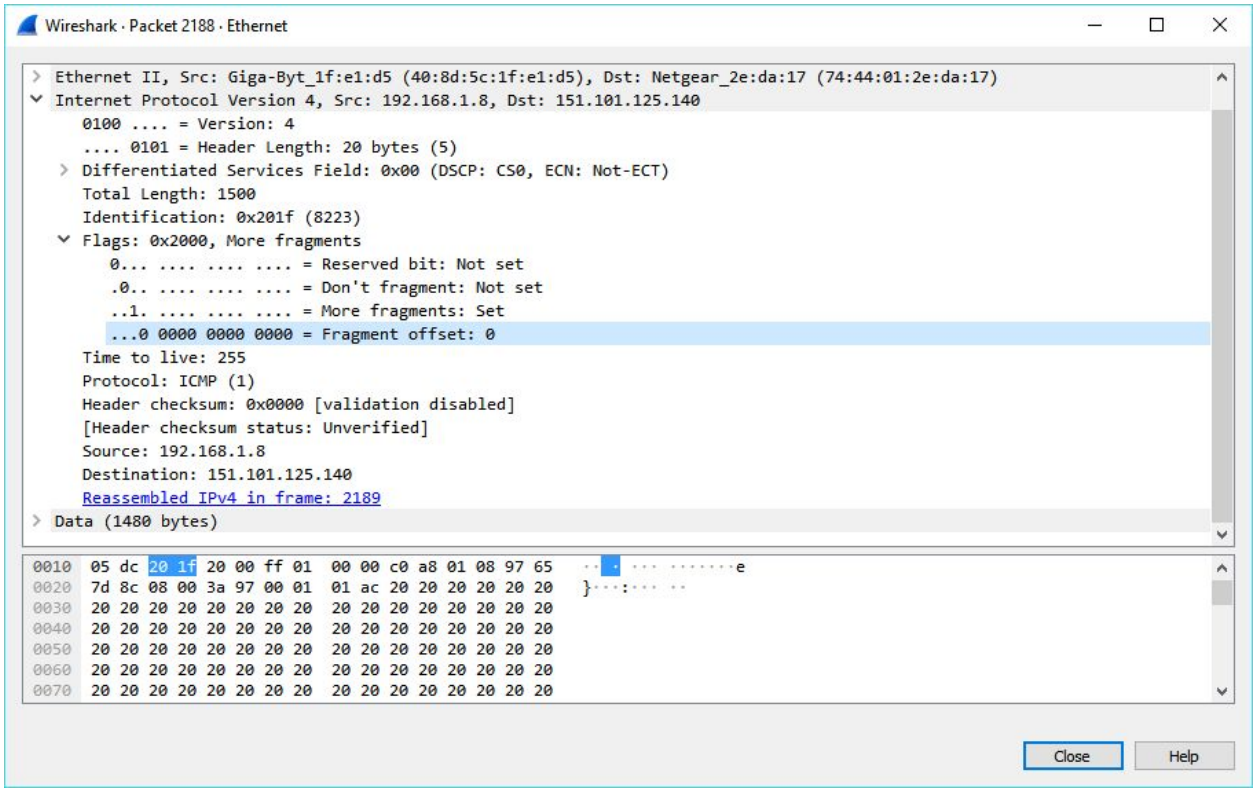
9. The identification value should not remain unchanged as it is meant to uniquely identify each packet. The TTL remains unchanged because the TTL for the first hop router always remains the same, and since the TTL was exceeded in transit, the request was sent again to the same first hop router.

10. As indicated by both the fragment flag and the wireshark populated “IP Fragments” field, this message has been fragmented.



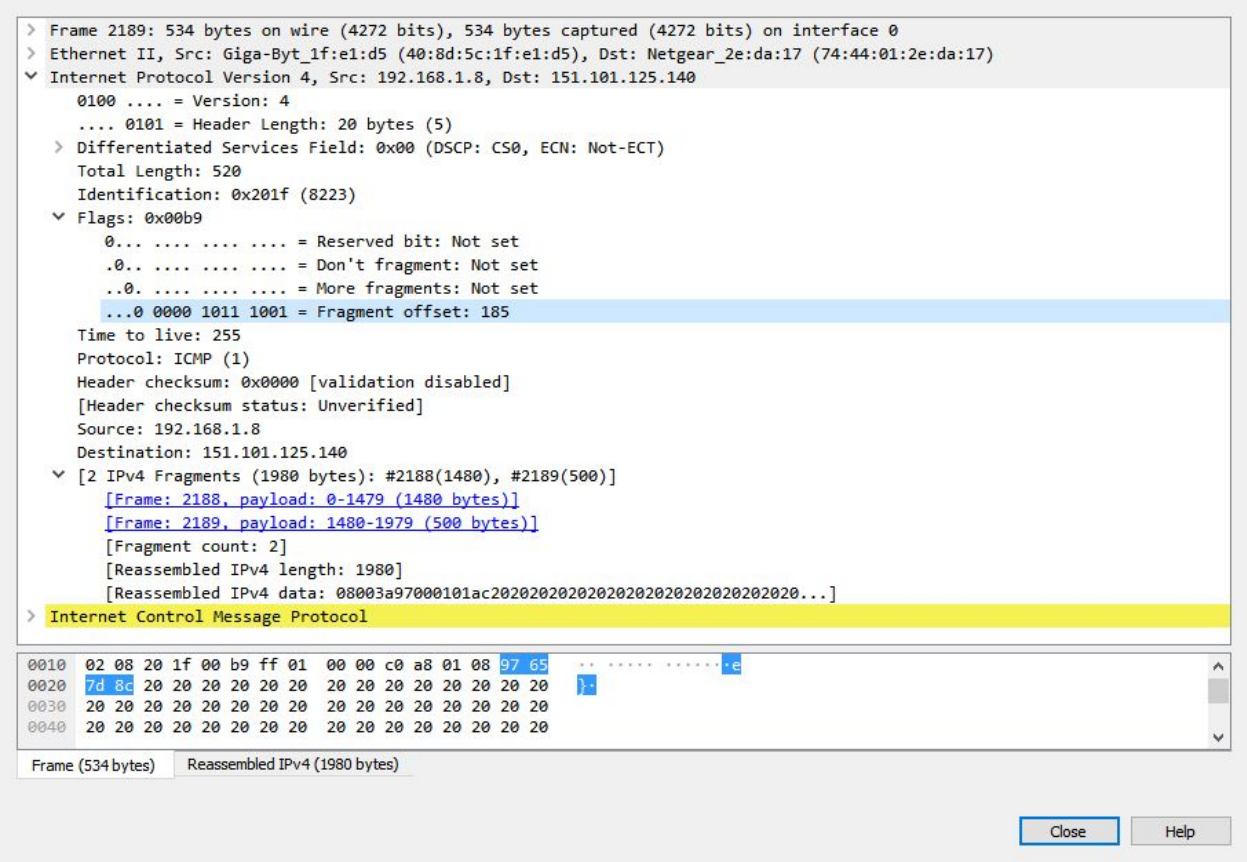
11. Although the fragment offset is set to 0, we can see the datagram has been fragmented because the more fragments flag is set. The fragment offset of 0 also tells us that this is the first fragment in the message. This IP datagram has a total length of 1500 bytes. The remaining 500 bytes are contained in the latter fragment, adding to a total of 2000 bytes.

This can also be seen in the frame sizes in the image above.



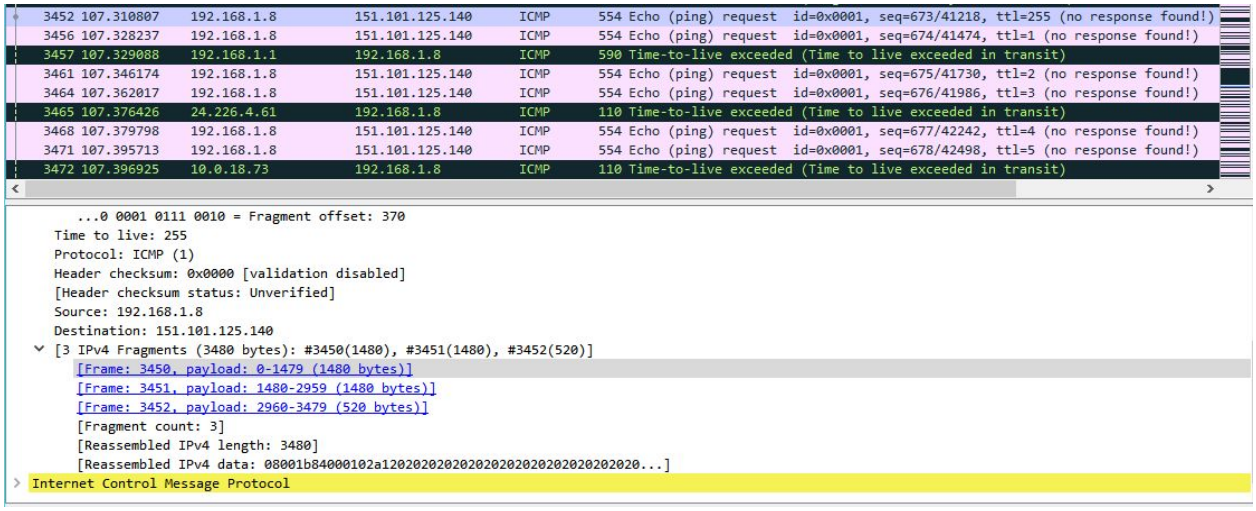
12. The second fragment has a non-zero fragment offset of 185. This tells us it is a fragment and not the first one. There are no more fragments after this one because the “More

fragments” flag is not set.



13. The only fields to change between the two fragments are the total length and any appropriate fragment flags such as the “more fragments” flag or the fragment offset, and once again, checksum would change in this case as well, since information in the IP header differs again.

14. When the packet size was set to 3500 bytes, 3 fragments were created from the original datagram.



15. Once again, between the three fragments, the different fragment flags change (offset, more), the total length differs between either the cap of 1500 bytes, or the total remaining number of bytes in the last fragment (in my case 540, the extra 40 bytes of course, coming from the 20 bytes in each previous IP header in each fragment). Checksum would also change each time due to the changes in each IP header, though once again, not in mine as this was not enabled at the time of the trace.