Step 3.) Packet retrieved from “`curl https://alexagruso.dev`”:

Version & Length	DSF	Total Length	ID	Flags	TTL	Protocol	···
2 Bytes	1 Byte	1 Byte	2 Bytes	2 Bytes	1 Byte	1 Byte	

···	Checksum	Source	Destination
	2 Bytes	4 Bytes	4 Bytes

1.) Based on the packet information, my IP address is `153.33.157.22` while the IP address of `alexagruso.dev` is `64.23.211.94`.

2.) The header length field is explicitly set to 20, while the total length field is set to 52, thus we can assume that the total length encommpasses both the headers and the payload.

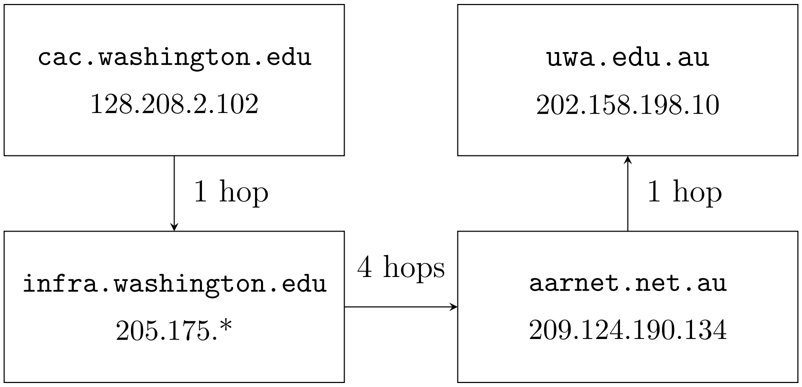
3.) While the identifcation differs for each packet, there are common prefixes, notably `xe**` and `75**` in my case. This is the case in both directions.

4.) Each packet with my IP address as it’s source has a TTL of 52, thus that must be the initial TTL. It is not the maximum possible value, as the response packets have a TTL of 64.

5.) The flag header of the packet has a value of `0400`, and inspecting in wireshark I found that the “don’t fragment” bit is set to 1, thus the packet is not fragmented.

6.) In my case, the length of the IP header was 20 bytes, but in the data this is encoded as a 5, thus the length is given in words (assuming 4 byte words) rather than bytes.

Step 4.)



Step 5.) 1.) Choosing a packet from the remote server and splitting it into 2 byte words, we get `4502`, `0120`, `7537`, `4000`, `3406`, `d88c`, `4017`, `d35e`, `0abf`, and `d9dd`.

2.) Summing these words, we get `3fffc`.

3.) Taking the 1's complement sum, we add `3` + `fffc` to get `10000` then invert the bits, resulting in `ffff`.

4.) Because the result is `ffff`, we can conclude that the checksum is correct.