PCAP Boston Puzzle 2 Solution by Tanner Kinkead

Per standard, the ICMP echo reply packet contains the same data that was initially received on the ICMP echo request packet.  Therefore, the payload of the ICMP echo reply packet, packet #11, shows the payload ICMP echo request packet as it was reassembled by the receiver (192.168.1.1).

In the attached spreadsheet, I extracted the ICMP headers and payload from the ICMP echo reply packet (packet #11) and placed it on line 13.  I then extracted the ten ICMP fragments that were sent by sender 192.168.11.13 and placed each packet on its own line (packets are labeled P1 through P10).  Each packet was placed at the offset defined in the Fragmentation Offset given in each packet's IP header.  The intent was to align all packets based on their offsets, in order to understand how the receiver must have processed those packets.

The ICMP header is eight bytes, and the ICMP echo reply had a payload of 72 bytes.  Therefore, the total size of the reassembled ICMP echo request packet will be 80 bytes--the 8 byte header and the 72 byte payload.

In the attached spreadsheet, the bytes highlighted in green are the bytes used in the reassembled ICMP echo request packet, while the bytes highlighted in red are not used (either discarded or overwritten).

Bytes 0-15 of the reassembled echo request packet are taken from Packet #1.  The first 8 bytes of this packet are header data.

Bytes 16-31 of the reassembled echo request packet are taken from Packet #3.  Packets #2 and #3 are overlapping fragments, as they are both set with an offset of 16 and a size of 16 bytes.  Based on the data provided in the ICMP echo reply packet, the receiver clearly overwrites the data taken from Packet #2 with the data that was received last, from Packet #3.

Bytes 32-47 of the reassembled echo request packet are taken from Packet #4.  While bytes 16-31 demonstrated that the receiver will overwrite the first packet received with the data from a later overlapping packet, in this case, Packet #5 is discarded due to a bad header checksum.

Bytes 48-55 of the reassembled echo request packet are taken from Packet #6.  There is no other overlapping packet in this byte range.

Bytes 56-63 of the reassembled echo request packet are also taken from Packet #6.  In this case, Packet #6 and Packet #7 both overlap at this byte range.  While bytes 16-31 demonstrated that the receiver will overwrite the first packet received with the data from a later overlapping packet, in this case, Packet #7 is not viewed as a fragment of the reassembled echo request packet because it has a different IP ID than all the other packets.

Bytes 64-71 of the reassembled echo request packet are taken from Packet #9.  While Packet #8 also overlaps this byte range, Packet #8 is likely discarded due to the packet having a TTL of 0.  Even if the packet is not discarded by the receiver, we have demonstrated in bytes 16-31 that the receiver will overwrite the first packet received with a later received packet if those packets overlap, so the receiver would likely use the data in Packet #9 even if Packet #8 were not discarded.

Bytes 72-79 of the reassembled echo request packet are taken from Packet #10.  Packet #8 also overlaps this byte range.  However, as was just discussed, Packet #8 is likely discarded due to a TTL of 0, and we would expect Packet #10 to be used by the receiver anyway, based on its previous treatment of overlapping packets.