EECS 325/425: Computer Networks
Project #4
Trace File Format

The format of this packet trace is as follows:

Each packet in the trace is prefaced by twelve bytes of meta information about the packet, as follows:

- **Bytes 1–4:** The number of seconds since Unix epoch (midnight GMT on January 1 1970). This value must be combined with the fractional value from the previous field to form the timestamp. This value is an unsigned 4-byte number encoded in *network byte order*. See *ntohl()*.
- **Bytes 5–8:** The fractional part of the timestamp. This is a number of microseconds (so ranging from 0–999,999) that must be combined with the next field to form the timestamp. This value is an unsigned 4-byte number encoded in *network byte order*. See *ntohl()*.
- **Bytes 9–10:** The number of bytes of the packet included in the trace file (the "captured length" or `caplen`). This value *does not* include the 12 bytes of meta information included for the given packet (i.e., it is *only* the packet portion). This will dictate the degree to which the packet can be processed. This value is an unsigned 2-byte number represented in *network byte order*. See *ntohs()*.
- **Bytes 11-12:** These bytes are not used for this project and while they are present, must be ignored.

The above information will dictate how much can be understood about the included packet. If the `caplen` is at least 14 bytes the Ethernet header is included in the trace file, as follows:

- **Bytes 13–26:** The Ethernet header of the packet: 6 bytes for the destination MAC address, 6 bytes for the source MAC address and 2 bytes for the type, in this order.
  Note: The Ethernet *preamble* is not included in the trace file. Likewise, the Ethernet trailer that includes the CRC is not included in the trace file.
  Note: The type field is encoded in *network byte order* in the trace file. See *ntohs()*.

If the `caplen` is at least 34 bytes and the Ethernet "type" field indicates an IP packet (0x0800) you will also be able to process the fixed IP header, as follows:

- **Bytes 27–46**: fixed IP header
  Note: The 16 and 32 bit fields in the header are unsigned values represented in network byte order and you'll need to use *ntohs()* and *ntohl()* to find the proper values.

- **Bytes 47–**:
  These bytes depend on the IP header. They could be IP options. They could be transport layer headers. You will need to figure this out based on the IP header length field and the value of `caplen`.

Following the IP header will be the transport header. Note: the `caplen` must be large enough to include the transport header or the transport header cannot be processed.

Note: The 2- and 4-byte quantities in the UDP and TCP headers are unsigned numbers encoded in network byte order and therefore will need processed with *ntohs()* and *ntohl()* before using these values.

Note: You will need to use the `caplen` to guide you to the next packet in the trace. That is, after `caplen` bytes you will find the twelve bytes of meta information for the next packet.

Assumptions

All packets included in the trace files will be Ethernet packets. That is, bytes that immediately follow the meta information for a packet can safely be interpreted as an Ethernet header.

Only full packet headers are included in the trace. In other words, no partial headers will be present. Consider a `caplen` of 20 bytes. This would indicate that the entire Ethernet header was present (14 bytes), but only 6 bytes of the IP header were included. Since an IP header must be at least 20 bytes this would only be part of the IP header. The trace files have been made to ensure this case will not happen.

Additional Hints

- You can find a structure for the Ethernet header in /usr/include/net/ethernet.h as `struct ether_header`.
- You can find a structure for the IP header in /usr/include/netinet/ip.h as `struct iphdr`.
- You can find a structure for the UDP header in /usr/include/netinet/udp.h as `struct udphdr`.
- You can find a structure for the TCP header in /usr/include/netinet/tcp.h as `struct tcphdr`.
- You can find constants for IP protocol numbers in /usr/include/netinet/in.h.

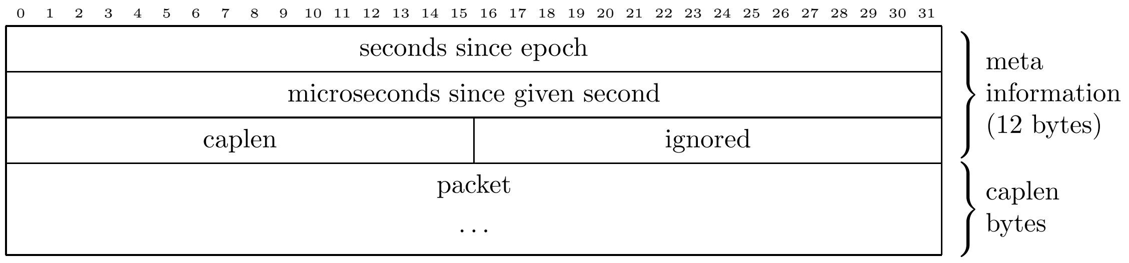

Figure 1: This is the general format of a single packet in the trace file. The first 12 bytes contain meta information (a timestamp and caplen). Following this are the first *caplen* bytes of the packet.

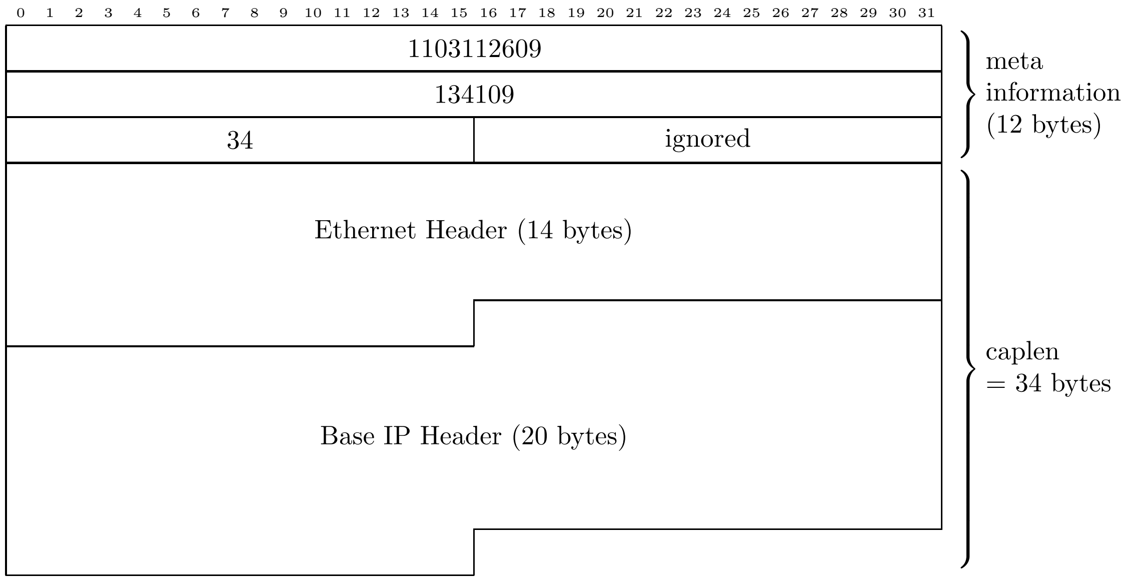

Figure 2: In this example, the meta information indicates the packet was captured 1103112609.134109 seconds after Unix epoch. Further, 34 bytes of the packet are available in the trace file. Finally, given the caplen (34 bytes), the Ethernet header (14 bytes) and base IP header (20 bytes) follow the meta information in the file.

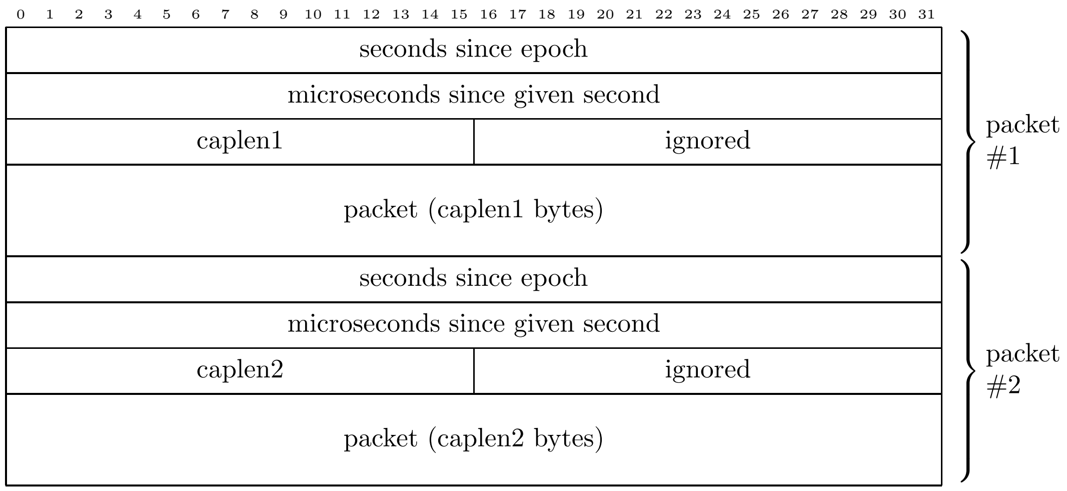

Figure 3: This example shows two packets in the packet trace. The first includes 12 bytes of meta information and then “caplen1” bytes of the first packet. After those caplen1 bytes, another 12 bytes of meta information for the second packet appears. The packet trace contains “caplen2” bytes of the second packet, which follows the second set of meta information.

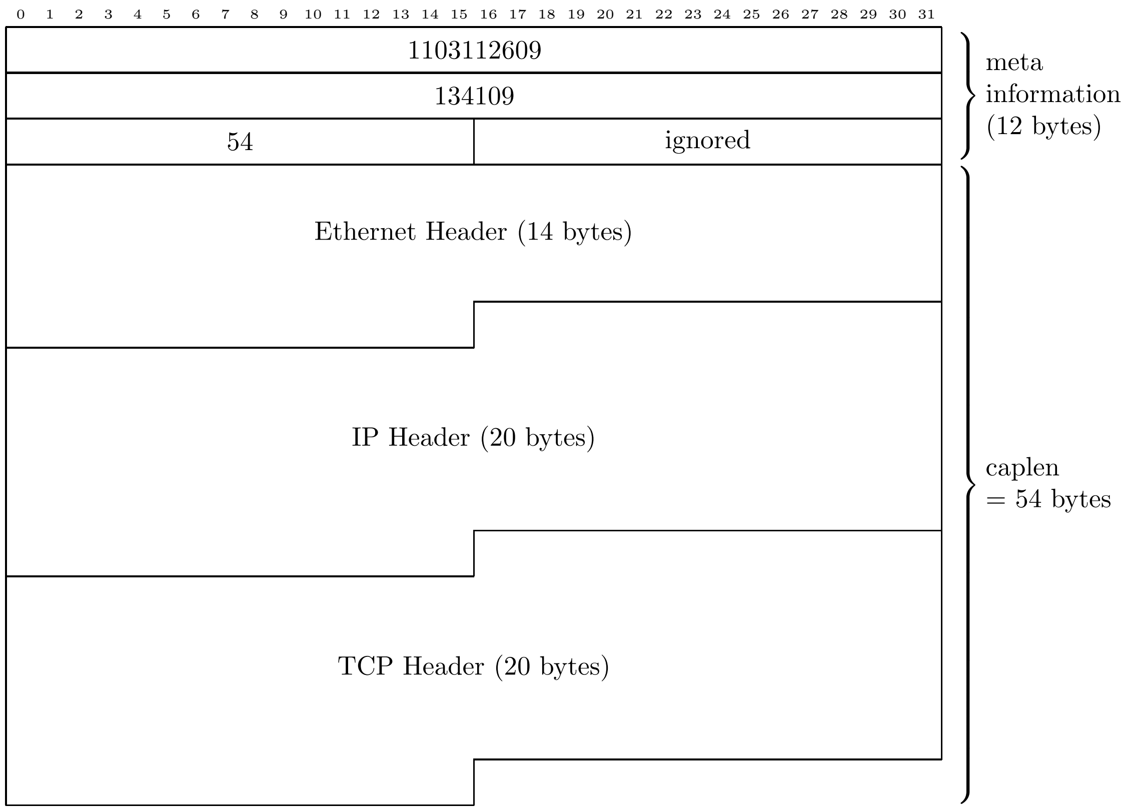

Figure 4: In this example, the meta information indicates that the packet was captured 1103112609.134109 seconds after Unix epoch. In addition, 54 bytes of the packet are available in the trace file. Further, given the caplen (54 bytes), the Ethernet header (14 bytes), the 20 byte IP header and the 20 byte TCP header follow the meta information in the file. (Note: This assumes that the IP and TCP headers do not contain options.)