(1) Tcpdump command:

sudo tcpdump -i en0 -n host 34.193.77.105 and port 1080 -w http_1080.pcap
sudo tcpdump -i en0 -n host 34.193.77.105 and port 1081 -w tcp_1081.pcap
sudo tcpdump -i en0 -n host 34.193.77.105 and port 1082 -w tcp_1082.pcap

Port:
http://www.sbunetsyslabs.com:1080
https://www.sbunetsyslabs.com:1081
https://www.sbunetsyslabs.com:1082

(2) Reassemble each unique HTTP Request/Response for http_1080.pcap

By st = chr(buf[66]) + chr(buf[67]) + chr(buf[68]) + chr(buf[69]), the tag of the packet is obtained. If the tag is 'HTTP', set it to 'Response'. If the tag is 'GET' or 'POST', set it to 'Request'. Then reassemble these packets and add a unique <source, dest, seq, ack> TCP tuple.

Packet type | < source, dest, seq, ack>
Request | < 53201 , 1080 , 2647119237 , 1085291901>
Response | < 1080 , 53201 , 1085291901 , 2647119612>
Request | < 53203 , 1080 , 2788615538 , 3302933901>
Request | < 53205 , 1080 , 3619712864 , 2426890533>
Request | < 53204 , 1080 , 2903719127 , 1519419686>
Response | < 1080 , 53203 , 3302933901 , 2788615951>
Response | < 1080 , 53205 , 2426890533 , 3619713205>
Response | < 1080 , 53204 , 1519419686 , 2903719541>
Request | < 53206 , 1080 , 2514788452 , 2206938078>
Request | < 53207 , 1080 , 1718781776 , 244523138>
Response | < 1080 , 53206 , 2206938078 , 2514788866>
Response | < 1080 , 53207 , 244523138 , 1718782194>
Request | < 53208 , 1080 , 108355115 , 227600883>
Response | < 1080 , 53208 , 227600883 , 108355539>
Request | < 53209 , 1080 , 202161362 , 1188820181>
Request | < 53211 , 1080 , 1778884972 , 1655787414>
Request | < 53210 , 1080 , 3695288357 , 1707992248>
Request | < 53212 , 1080 , 2114612987 , 1696915322>
Request | < 53213 , 1080 , 241761684 , 1267824720>
Request | < 53214 , 1080 , 4053044216 , 2463070101>
Response | < 1080 , 53211 , 1655787414 , 1778885389>
Response | < 1080 , 53209 , 1188820181 , 202161784>
Response | < 1080 , 53210 , 1707992248 , 3695288776>
Response | < 1080 , 53212 , 1696915322 , 2114613406>

Response | < 1080 , 53213 , 1267824720 , 241762101>
Response | < 1080 , 53214 , 2463070101 , 4053044634>
Response | < 1080 , 53209 , 1188820181 , 202161784>
Request | < 53215 , 1080 , 3842899204 , 993292112>
Request | < 53217 , 1080 , 241668507 , 4008666740>
Request | < 53216 , 1080 , 4059706464 , 1845061346>
Request | < 53218 , 1080 , 1527944225 , 1704606806>
Response | < 1080 , 53217 , 4008666740 , 241668921>
Response | < 1080 , 53216 , 1845061346 , 4059706881>
Response | < 1080 , 53215 , 993292112 , 3842899623>
Response | < 1080 , 53218 , 1704606806 , 1527944639>
Request | < 53219 , 1080 , 829692922 , 2605665756>
Response | < 1080 , 53219 , 2605665756 , 829693263>

(3) Identify which HTTP protocol is being used for each PCAP file

To judge the PCAP file, first, if the FIN flag of the data packet appears in the first half of the file, it means that Non Persistent HTTP is used, and HTTP 1.0 is used. Then if the number of TCP flows is greater than 1, it means Parallelization HTTP, HTTP 1.1 is used. If the number of TCP flows is 1, it means that it is Pipeling HTTP, and HTTP 2.0 is used.

HTTP protocol of the PCAP file: http_1080.pcap = HTTP 1.0
HTTP protocol of the PCAP file: tcp_1081.pcap = HTTP 1.1
HTTP protocol of the PCAP file: tcp_1082.pcap = HTTP 2.0

(4) Finally, after you've labeled the PCAPs with their appropriate versions of HTTP, answer the following:

1.Which version of the protocol did the site load the fastest under? The Slowest?

The fastest among the protocols is HTTP 1.0, because there are too few activities in the web page and the number of objects is small. Since TTP1.1 and HTTP 2.0 are more suitable for batch processing objects, HTTP 1.0 is faster than HTTP 1.1 and HTTP 2.0.

The slowest protocol in the protocol is HTTP 1.1. The reason is that too many flows are opened and closed, resulting in many more packets than HTTP2.0, so HTTP 1.1 is slower than HTTP2.0.

The site load time: HTTP 1.0 = 2.6580018997192383 Seconds
The site load time: HTTP 1.1 = 6.0530009269714355 Seconds
The site load time: HTTP 2.0 = 5.621369123458862 Seconds

2.Which sent the most number of packets and raw bytes? Which protocol sent the least?

The most packets are HTTP1.0, because too many flow opening and closing will generate more packets. The least packets is HTTP2.0, because it has the least flow. The number of raw bytes is not much different, because the number and size of objects are the same, and the data packets of SYN and FIN are very small, so the raw bytes transmitted are almost the same.

HTTP 1.0
Number of packets: 1972
Number of raw bytes: 2279463 Bytes
HTTP 1.1
Number of packets: 1937
Number of raw bytes: 2304206 Bytes
HTTP 2.0
Number of packets: 1705
Number of raw bytes: 2292555 Bytes