Understanding Software Dynamics Ch 6 Lab Report

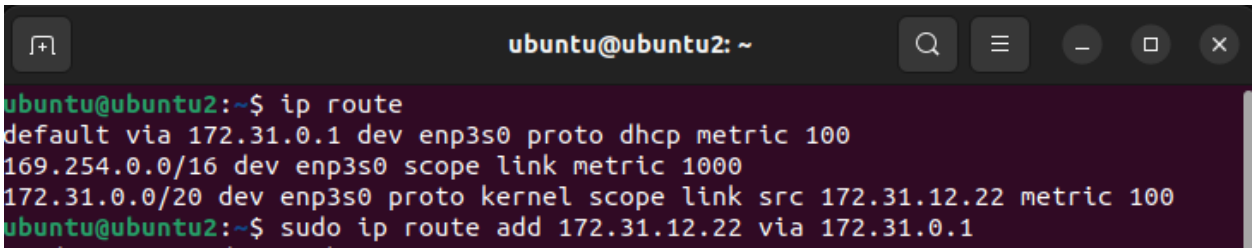
I tried many times and cannot postprocess. Then I looked at the code and found that the book's commands given on p.106 are incorrect. The correct command:
./dumplogfile4 "ping" client4_20250929_092429_ubuntu2_392753.log >client4_20250929_092429_ubuntu2_392753.json
./makeself show_rpc.html client4_20250929_092429_ubuntu2_392753.json client4_20250929_092429_ubuntu2_392753.html
The above command needed to be run in /postproc.

---

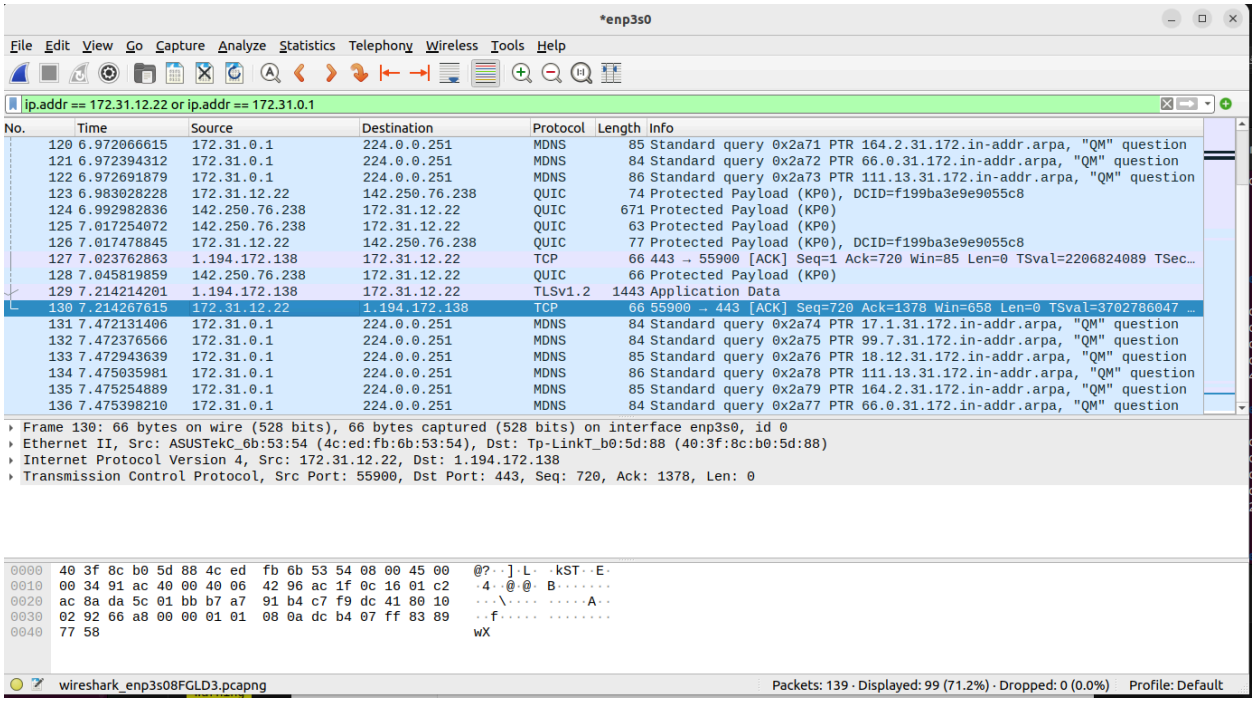
useful command: ip addr show

I run server and client in the same machine using
./client4_20250929_092429_ubuntu2_392753.html
And find out that the estimated network time is extremely small (1us), which is 1000x less than expected. It is because of the OS's internal routing optimization.

Next, I have to configure to bypass OS internal routing optimization (which keeps traffic local), so I modify the routing table.
modify_routing_table.png



Then I verified there is tcp handshake between a client-server test by Wireshark.
wireshark_tcp_handshake.png(frame 127, 129, 130)
(simple client-server test:
$ python3 -m http.server 8000 --bind 172.31.12.22
$ curl http://172.31.12.22:8000)



./client4 172.31.12.22 12345 ping
./client4 172.31.12.22 12345 -k 10 write -key "kkkkk" + -value "kkkkk" + 1000000
./client4 172.31.12.22 12345 -k 10 read -key "kkkkk" +
./client4 172.31.12.22 12345 quit

Important: 50ms is the standard magnitude for request-response latency. (RTT)

ubuntu@ubuntu2: ~/Documents/github/KUtrace-experiments

```
ubuntu@ubuntu2:~/Documents/github/KUtrace-experiments$ ./client4 172.31.12.22 12
345 ping
at client, server IP = ac1f0c16:3039

Histogram of floor log 2 buckets of usec response times
1 2+ 4+ us            1+ 2+ 4+ msec         1+ 2+ 4+ sec           1K+ 2k+ secs
|                     |                     |                      |
0 0 0 0 0 0 0 0 0 1   0 0 0 0 0 0 0 0 0 0   0 0 0 0 0 0 0 0 0 0   0 0
1 RPCs,   0.5 msec, 0.000 TxMB, 0.000 RxMB total
1945.5 RPC/s (0.514 msec/RPC),   0.2 TxMB/s,   0.2 RxMB/s

client4_20250930_144727_ubuntu2_523896.log written
0.514ms  ubuntu@ubuntu2:~/Documents/github/KUtrace-experiments$ ./client4 172.31
.12.22 12345 -k 10 write -key "kkkkk" + -value "kkkkk" + 1000000
at client, server IP = ac1f0c16:3039
1.993ms  1.917ms  1.711ms  1.221ms  1.256ms  1.207ms  1.323ms  1.250ms  1.235ms
 1.388ms

Histogram of floor log 2 buckets of usec response times
1 2+ 4+ us            1+ 2+ 4+ msec         1+ 2+ 4+ sec           1K+ 2k+ secs
|                     |                     |                      |
0 0 0 0 0 0 0 0 0 0   10 0 0 0 0 0 0 0 0 0   0 0 0 0 0 0 0 0 0 0   0 0
10 RPCs,  14.5 msec, 10.001 TxMB, 0.001 RxMB total
689.6 RPC/s (1.450 msec/RPC), 689.7 TxMB/s,   0.1 RxMB/s

client4_20250930_144756_ubuntu2_523938.log written
ubuntu@ubuntu2:~/Documents/github/KUtrace-experiments$ ./client4 172.31.12.22 12
345 -k 10 read -key "kkkkk" +
at client, server IP = ac1f0c16:3039
1.085ms  0.571ms  0.298ms  0.293ms  0.242ms  0.247ms  0.272ms  0.261ms  0.260ms
 0.260ms

Histogram of floor log 2 buckets of usec response times
1 2+ 4+ us            1+ 2+ 4+ msec         1+ 2+ 4+ sec           1K+ 2k+ secs
|                     |                     |                      |
0 0 0 0 0 0 0 2 6 1   1 0 0 0 0 0 0 0 0 0   0 0 0 0 0 0 0 0 0 0   0 0
10 RPCs,   3.8 msec, 0.001 TxMB, 10.001 RxMB total
2639.2 RPC/s (0.379 msec/RPC),   0.3 TxMB/s, 2639.5 RxMB/s

client4_20250930_144804_ubuntu2_523948.log written
ubuntu@ubuntu2:~/Documents/github/KUtrace-experiments$ ./client4 172.31.12.22 12
345 quit
at client, server IP = ac1f0c16:3039

Histogram of floor log 2 buckets of usec response times
1 2+ 4+ us            1+ 2+ 4+ msec         1+ 2+ 4+ sec           1K+ 2k+ secs
|                     |                     |                      |
0 0 0 0 0 0 0 0 1 0   0 0 0 0 0 0 0 0 0 0   0 0 0 0 0 0 0 0 0 0   0 0
1 RPCs,   0.4 msec, 0.000 TxMB, 0.000 RxMB total
2469.1 RPC/s (0.405 msec/RPC),   0.2 TxMB/s,   0.2 RxMB/s

client4_20250930_144810_ubuntu2_523960.log written
```

6.1) client4_20250930_144727_ubuntu2_523896.html
estimate: 0.01ms pings are usually small ICMP packets. And the server only needs to validate the checksum.
reality: 0.5ms

6.2) client4_20250930_144756_ubuntu2_523938.html
estimate: 0.05ms for 1MB if written in RAM, because it involves key parsing, memory allocation, and data copy
Reality: 1.5ms each on average

6.3) client4_20250930_144804_ubuntu2_523948.html
estimate: 0.03ms because reads are usually faster than writes, because no memory allocation
Reality: 1ms for the first read, and 0.3ms on average for later reads

And all the response message transmission above just adds RTT/2 (from the server back to the client after processing).

Reason for difference:

- Software Overhead: Parsing, serialization, or protocol handling (e.g., TCP, HTTP stack overhead like kernel interrupts and socket handling) adds ~0.1-0.5 ms.
- System Load: complex memory management (e.g., allocation delay/fragmentation when searching for available memory, garbage collection to find free space) adds ~0.5-1 ms for 1 MB ops.
- Initialization: First read's 1 ms likely includes setup (e.g., cache miss, session init), while subsequent reads benefit from caching.

It is not likely to be because of disk IO, which would be 5-10ms.