

一、运行成功截图

```
• ubuntu@ubuntu2310:~/Code/minnow$ cmake --build build --target check_webget
Test project /home/ubuntu/Code/minnow/build
  Start 1: compile with bug-checkers
1/2 Test #1: compile with bug-checkers ..... Passed    85.29 sec
  Start 2: t_webget
2/2 Test #2: t_webget ..... Passed    1.74 sec

100% tests passed, 0 tests failed out of 2

Total Test time (real) = 87.05 sec
Built target check_webget
```

二、实验过程

Code > minnow > data.txt

38322 [1734769620.834310] 64 bytes from 146.75.115.10: icmp_seq=38495 ttl=128 time=303 ms

38323 [1734769621.094269] 64 bytes from 146.75.115.10: icmp_seq=38496 ttl=128 time=303 ms

38324 [1734769621.227611] 64 bytes from 146.75.115.10: icmp_seq=38497 ttl=128 time=234 ms

38325 [1734769621.429866] 64 bytes from 146.75.115.10: icmp_seq=38498 ttl=128 time=236 ms

38326 [1734769621.626255] 64 bytes from 146.75.115.10: icmp_seq=38499 ttl=128 time=231 ms

38327 [1734769621.827137] 64 bytes from 146.75.115.10: icmp_seq=38500 ttl=128 time=230 ms

38328 [1734769622.100235] 64 bytes from 146.75.115.10: icmp_seq=38501 ttl=128 time=302 ms

38329 [1734769622.230305] 64 bytes from 146.75.115.10: icmp_seq=38502 ttl=128 time=231 ms

38330 [1734769622.437294] 64 bytes from 146.75.115.10: icmp_seq=38503 ttl=128 time=237 ms

38331 [1734769622.702411] 64 bytes from 146.75.115.10: icmp_seq=38504 ttl=128 time=301 ms

38332 [1734769622.896583] 64 bytes from 146.75.115.10: icmp_seq=38505 ttl=128 time=293 ms

38333 [1734769623.090899] 64 bytes from 146.75.115.10: icmp_seq=38506 ttl=128 time=287 ms

38334 [1734769623.294541] 64 bytes from 146.75.115.10: icmp_seq=38507 ttl=128 time=289 ms

38335 [1734769623.402150] 64 bytes from 146.75.115.10: icmp_seq=38508 ttl=128 time=285 ms

问题 6 输出 终端 端口

调试控制台 终端

筛选器(例如 text、lexclud...

>

[1734769653.402802] 64 bytes from 146.75.115.10: icmp_seq=38657 ttl=128 time=231 ms

[1734769653.607493] 64 bytes from 146.75.115.10: icmp_seq=38658 ttl=128 time=235 ms

[1734769653.803576] 64 bytes from 146.75.115.10: icmp_seq=38659 ttl=128 time=230 ms

[1734769654.006267] 64 bytes from 146.75.115.10: icmp_seq=38660 ttl=128 time=231 ms

[1734769654.206093] 64 bytes from 146.75.115.10: icmp_seq=38661 ttl=128 time=230 ms

[1734769654.486539] 64 bytes from 146.75.115.10: icmp_seq=38662 ttl=128 time=308 ms

[1734769654.789583] 64 bytes from 146.75.115.10: icmp_seq=38663 ttl=128 time=411 ms

[1734769654.811030] 64 bytes from 146.75.115.10: icmp_seq=38664 ttl=128 time=231 ms

[1734769655.093419] 64 bytes from 146.75.115.10: icmp_seq=38665 ttl=128 time=312 ms

[1734769655.218011] 64 bytes from 146.75.115.10: icmp_seq=38666 ttl=128 time=237 ms

[1734769655.486268] 64 bytes from 146.75.115.10: icmp_seq=38667 ttl=128 time=304 ms

[1734769655.612021] 64 bytes from 146.75.115.10: icmp_seq=38668 ttl=128 time=229 ms

[1734769655.888918] 64 bytes from 146.75.115.10: icmp_seq=38669 ttl=128 time=305 ms

[1734769656.087755] 64 bytes from 146.75.115.10: icmp_seq=38670 ttl=128 time=303 ms

[1734769656.216293] 64 bytes from 146.75.115.10: icmp_seq=38671 ttl=128 time=231 ms

[1734769656.498157] 64 bytes from 146.75.115.10: icmp_seq=38672 ttl=128 time=311 ms

[1734769656.687540] 64 bytes from 146.75.115.10: icmp_seq=38673 ttl=128 time=300 ms

[1734769656.885595] 64 bytes from 146.75.115.10: icmp_seq=38674 ttl=128 time=297 ms

[1734769657.085542] 64 bytes from 146.75.115.10: icmp_seq=38675 ttl=128 time=295 ms

[1734769657.221920] 64 bytes from 146.75.115.10: icmp_seq=38676 ttl=128 time=230 ms

[1734769657.487366] 64 bytes from 146.75.115.10: icmp_seq=38677 ttl=128 time=294 ms

[1734769657.626695] 64 bytes from 146.75.115.10: icmp_seq=38678 ttl=128 time=233 ms

二、数据分析

1. deliver rate

99.54%

2. longest consecutive string of successful pings

3231

3. longest burst of losses

9

4. How independent or correlated is the event of “packet loss” over time?

- Given that echo request #N received a reply, what is the probability that echo request #(N+1) was also successfully replied-to?

99.69%

- Given that echo request #N did not receive a reply, what is the probability that echo request #(N+1) was successfully replied-to?

90.15%

- How do these figures (the conditional delivery rates) compare with the overall “unconditional” packet delivery rate in the first question? How independent or bursty were the losses?

If previous request received a reply, it has higher probability to receive a successfully reply than the previous request didn't receive a reply.

Since the probability of success is lower if the previous failed, so the losses may be cumulative.

5. minimum RTT

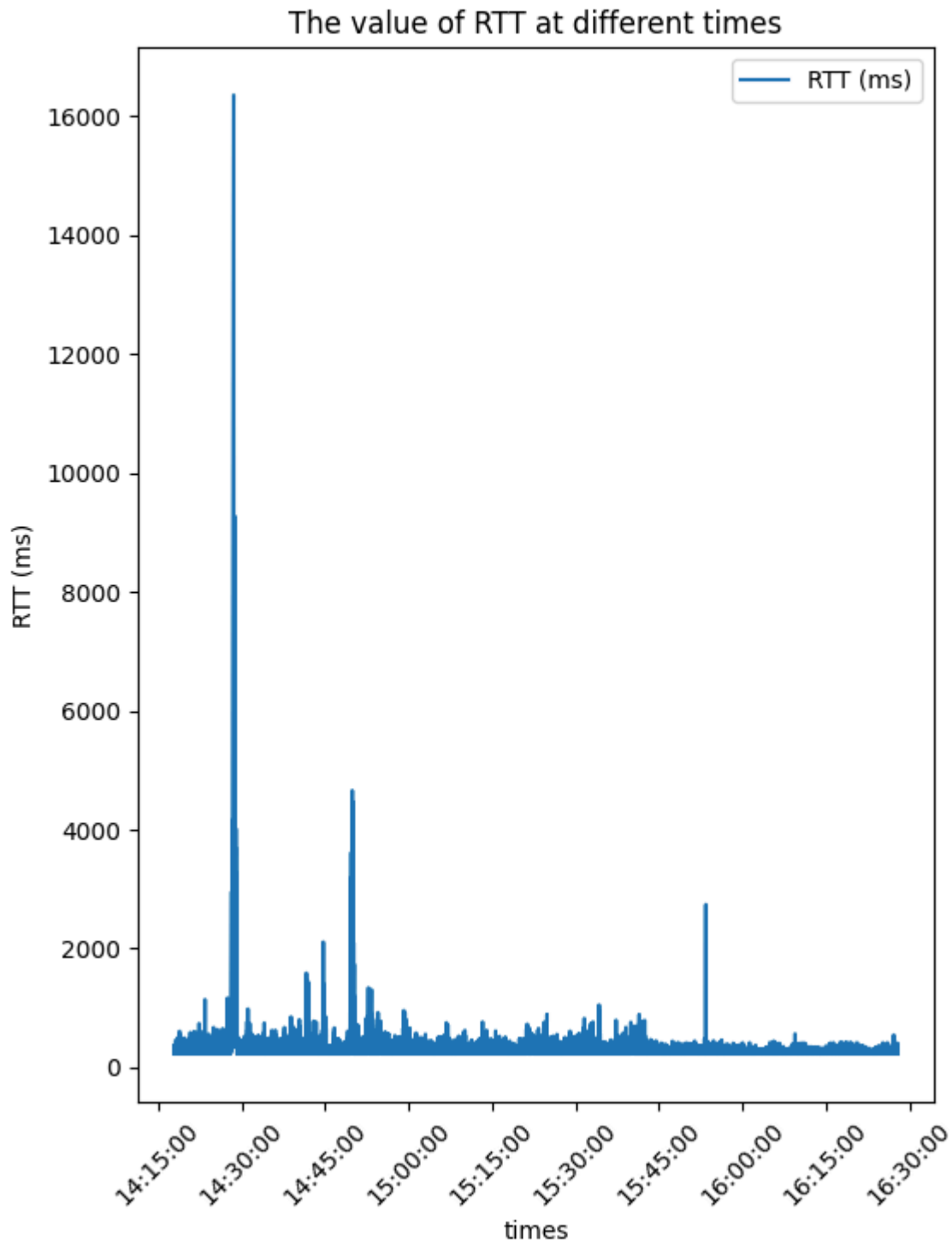
229

6. maximum RTT

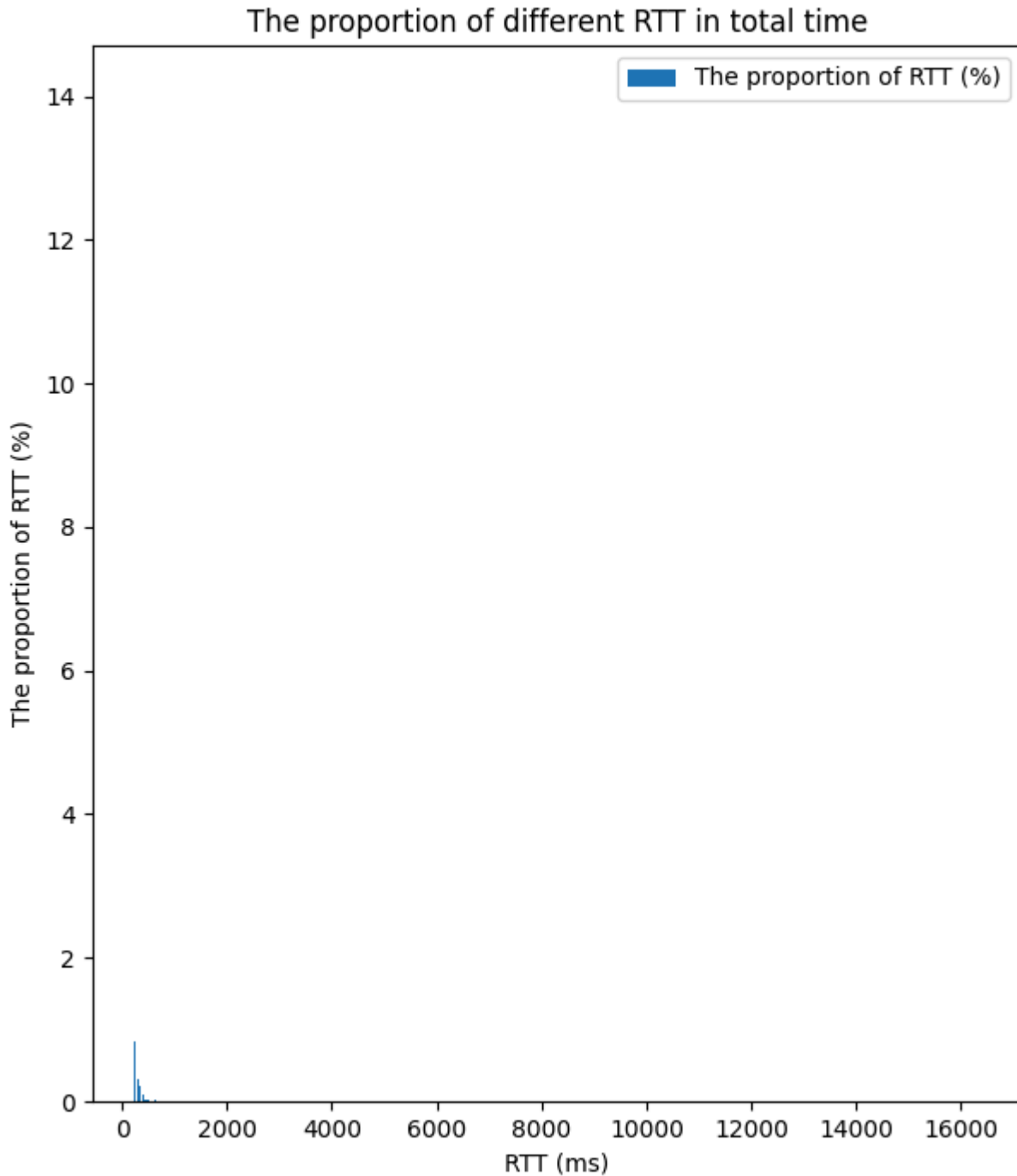
16355

```
Delivery Rate: 99.54%  
Longest Consecutive Successful Pings: 3231  
Longest Burst of Losses: 9  
Min RTT: 229.0 ms  
Max RTT: 16355.0 ms  
Success After Success: 38342/38461 (99.69%)  
Loss After Success: 119/132 (90.15%)
```

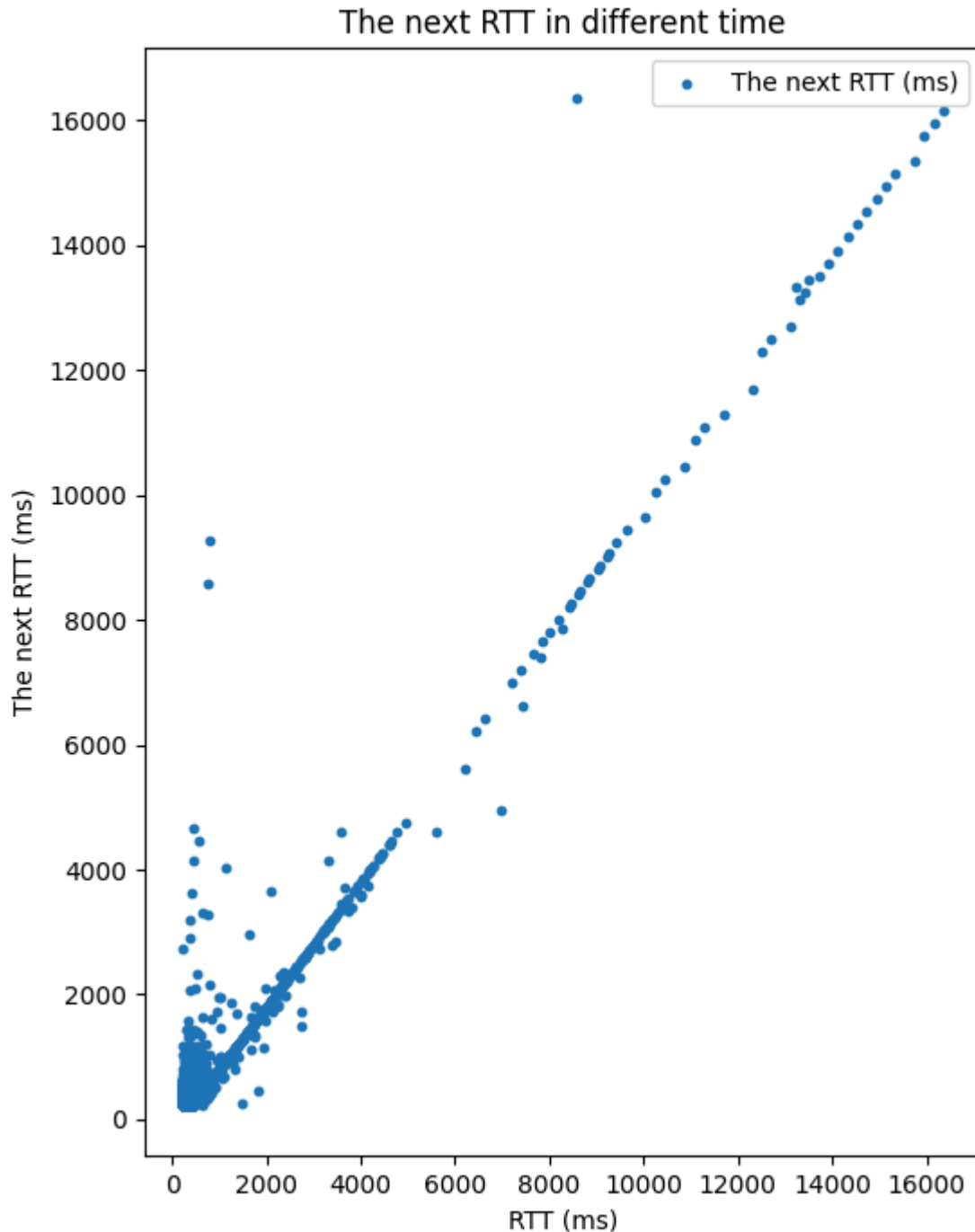
7. graph of the RTT as a function of time



8. histogram or Cumulative Distribution Function of the distribution of RTTs



9. correlation between “RTT of ping #N” and “RTT of ping #N+1”



10. Conclusions

From this data, we can see that packet loss is not independent, and when the last request fails, the next request is more likely to lose packets.

At the same time, RTT is basically normal distribution, but in this experiment, because there are individual maximum RTT, the chart does not look very obvious. This may be because during the experiment, the network was not particularly stable for a while, which is also a very common phenomenon in real life.

The relationship between the last RTT and the next RTT also reflects the correlation of RTT. The icon shows that it can be approximated as a straight line, that is, the higher the value of the last RTT, the higher the value of the next RTT