Q1How does the latency change as the number of clients change? Does it change for different types of requests?

Q2How does the latency of the lookup requests compare to trade? Since trade requests involve all these microservices, while lookup requests only involve two microservices, does it impact the observed latency?

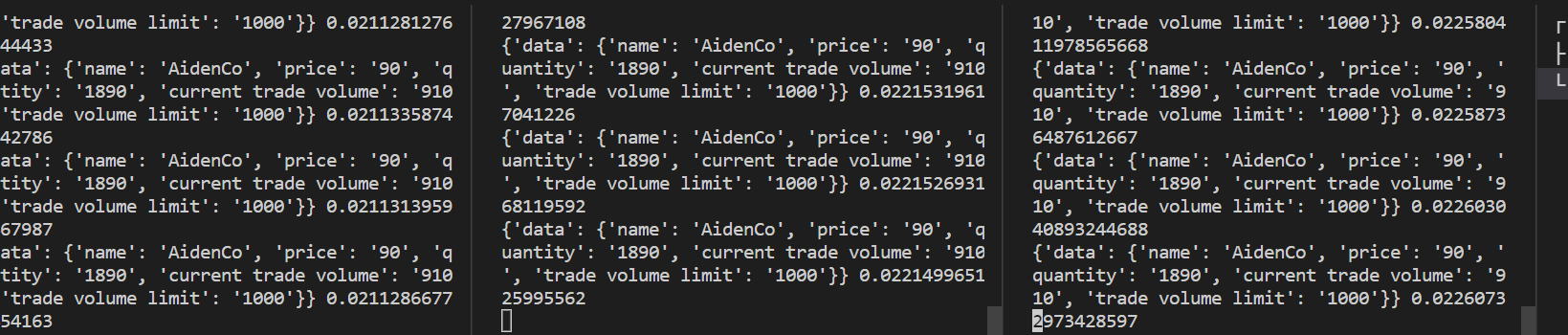
**We can see that the respond time increase linearly as the number of clients goes up, and since trade request needs to go through one more server, it takes longer time. As is depicted below**

Here shows the response time(in seconds) of sending lookup requests to the frontend server, using one,three five,seven clients(each client keeps sending request in a while loop)

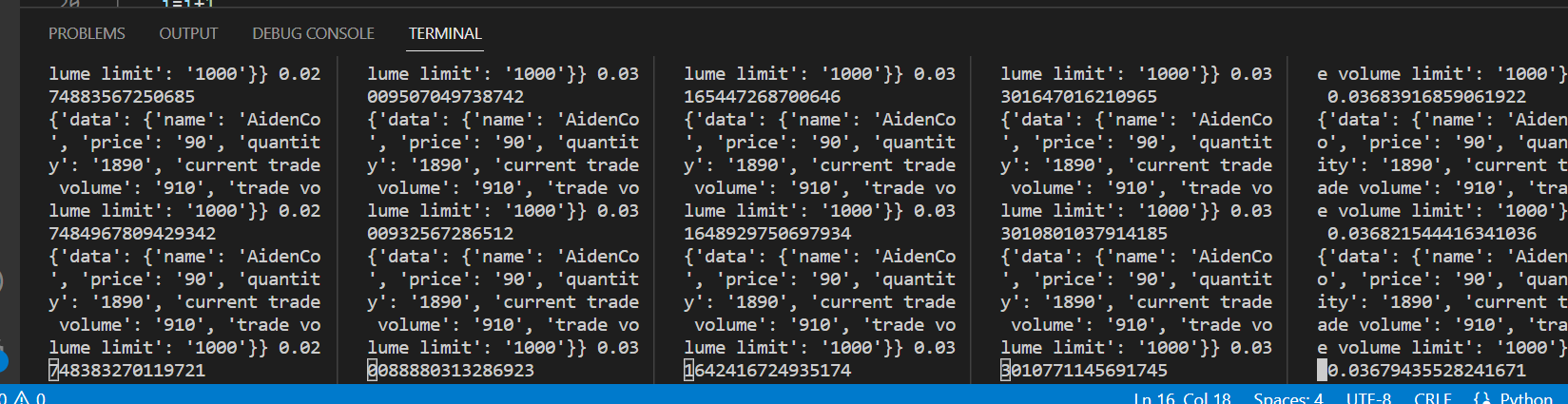
One client



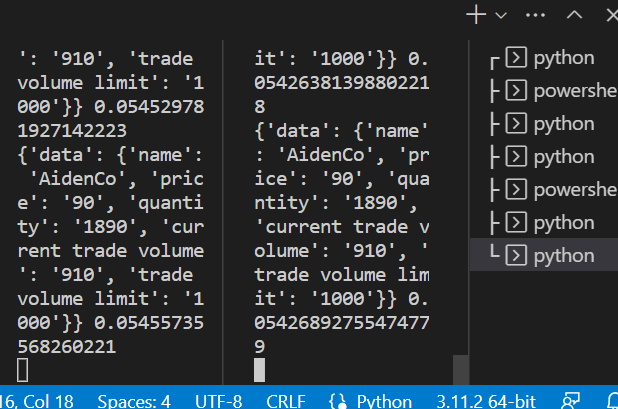
three clients



Five clients



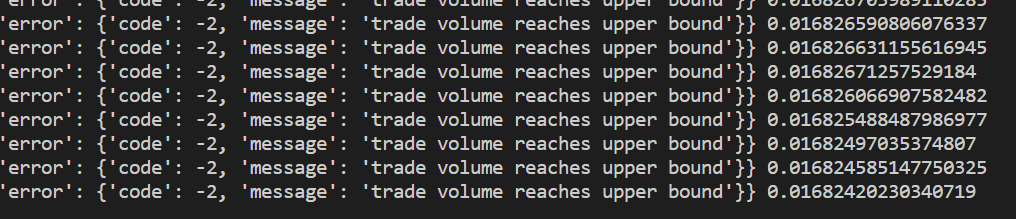
seven clients



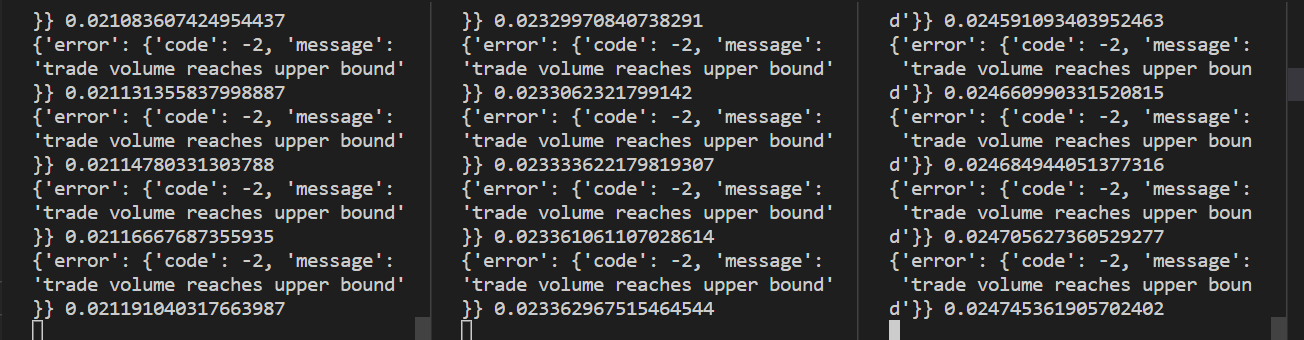
Here shows the response time(in seconds) of sending trade requests to the frontend server, using one, five,ten clients(each client keeps sending request in a while loop).

The result will keep showing the “trade volume reaches upper bound”, that’s because our continuous trade operation. However, that don’t affect our test because the request still go through all three server’s processing and being send back.

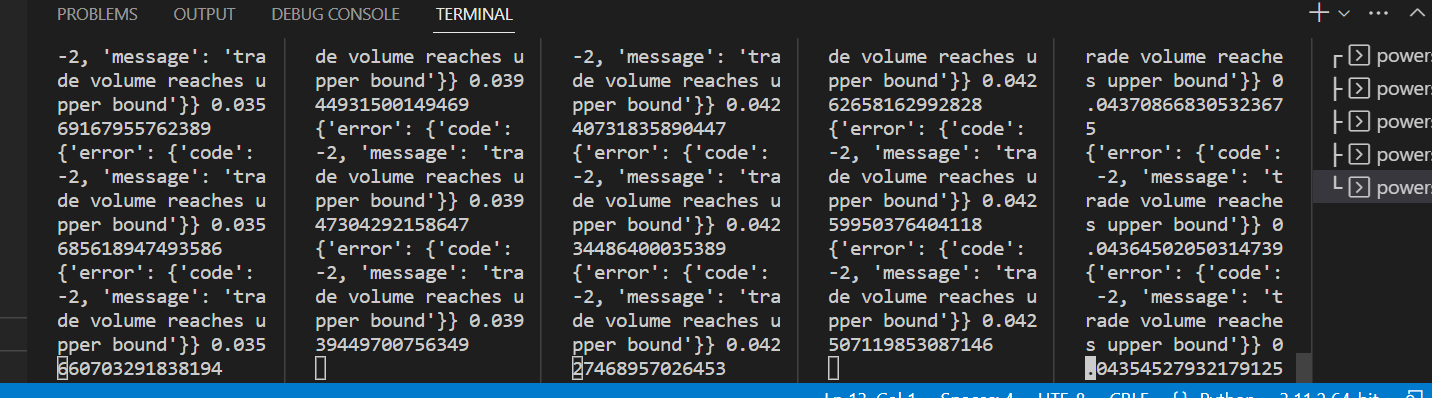
One client



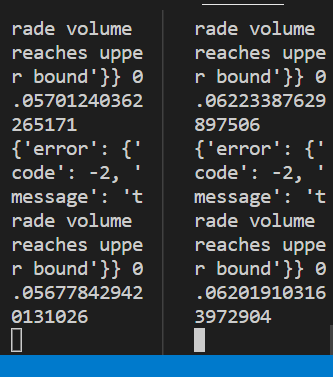
Three clients



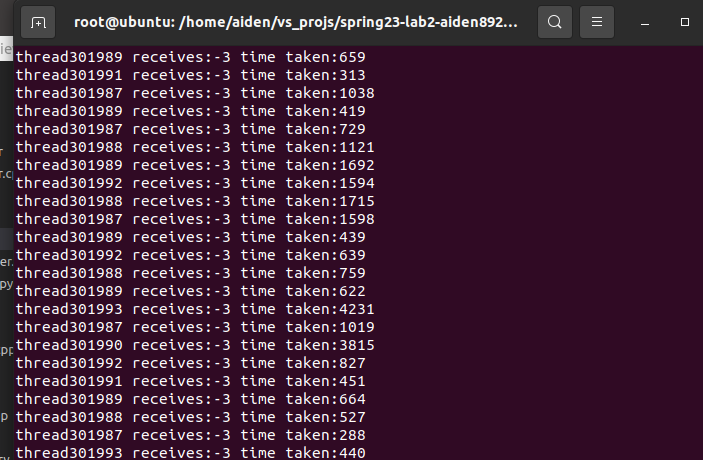
five clients



Seven clients

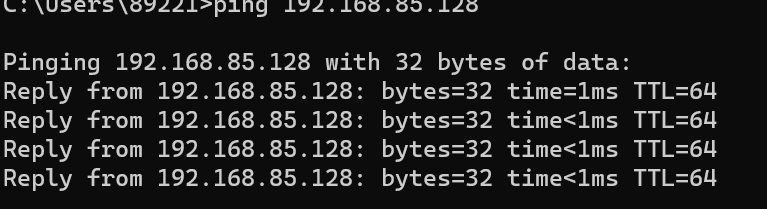


Then we want to know which part becomes the bottleneck of the system, the following picture shows the latency(in microsecond) we get when we have 7 clients send trade requests directly to the order server, we do this because we suspect the python-implemented frontend is the cause, the catalog server and order server are all implemented using c++.



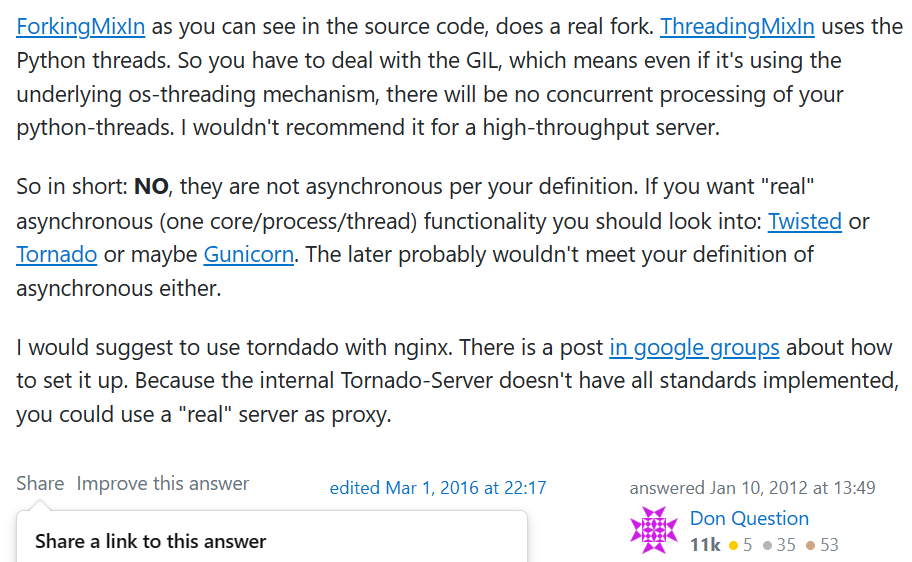
Here the delay are from hundreds of microsecond to a few milisecond,this including processing and transmiting delay in two servers. and we maesured the network delay between client and frontend server using ping, which shows it is less than 1 second.

Thus we can say, among the 50 milisecond delay we witnessed in the above test, the majority of them are incured by the python frontend.



Why is python-implemented frontend server so slow? it is supposed to be multithreaded as well.

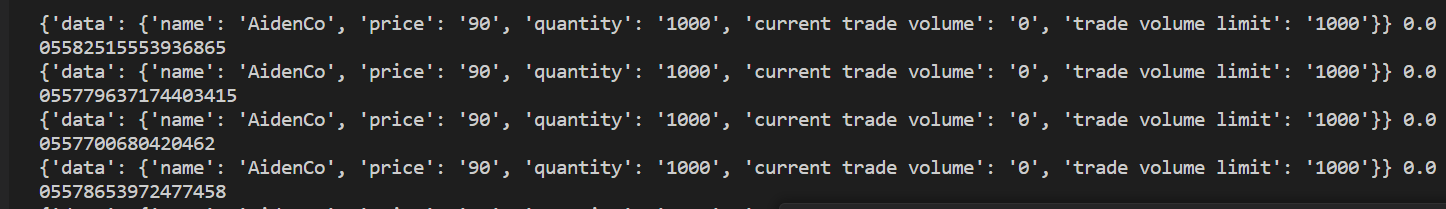
Under further research, I found the reason from stackoverflow



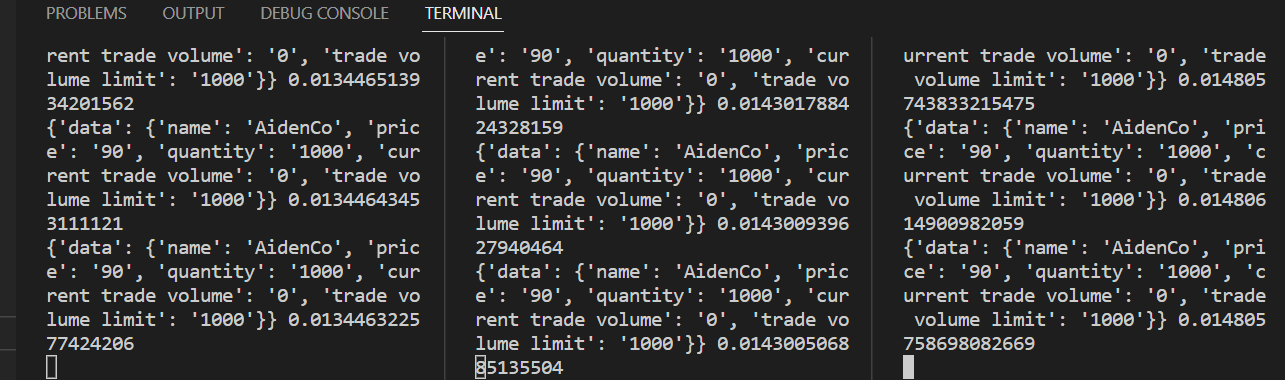
Does the latency of the application change with and without Docker containers? Did virtualization add any overheads?

Again we show the response time(in seconds) of sending lookup requests to the frontend server, using one,three five,seven clients(each client keeps sending request in a while loop), but each server are now run inside a container

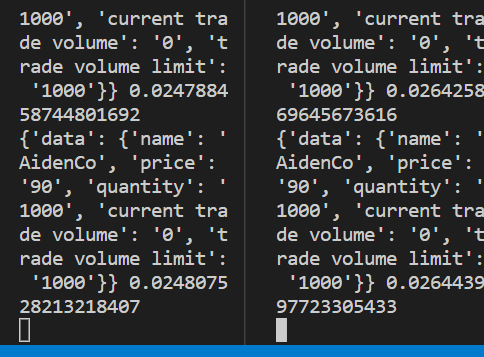
One client



Three clients



Five clients



**We see the response time didn’t increase at all, even decrease for quite a amount. The reason I suspect is that In non-docker setting, I run all servers in the same machine, and have the clients send requests from a different machine, in this case, three server process had to content resources with many other process which also running on the machine. While on the docker setting, each server has dedicated resources for them. Resulting in better performance.**