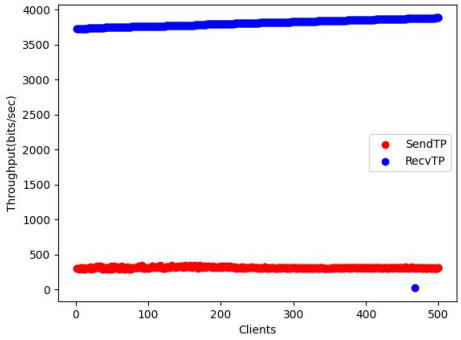
Since, tcpDump is tun on the server machine, metrics calculated will indicate reliable results in case of server. Hence, throughput is split into two categories, sending throughput and receiving throughput.

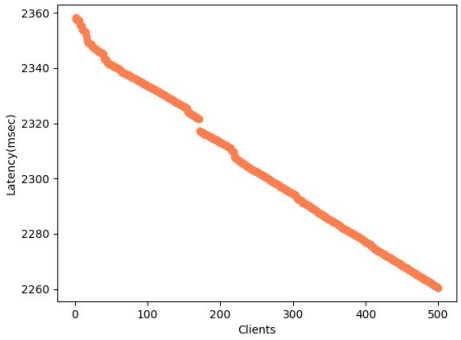
Specifications:

- CPU-> 3 cores
- RAM-> 4 GB

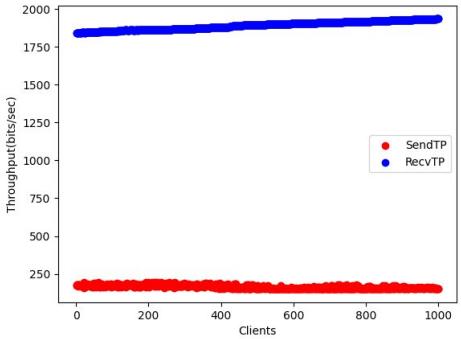
FORK

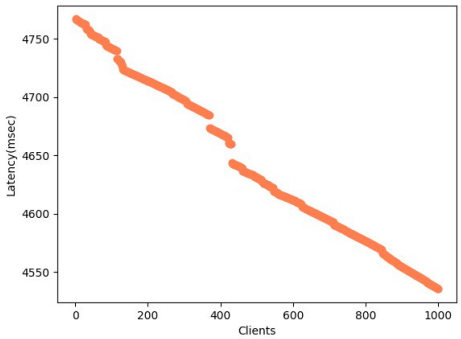
- **500**



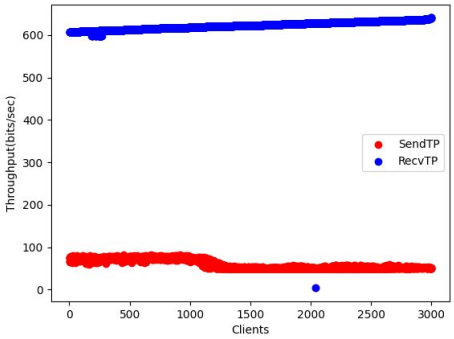


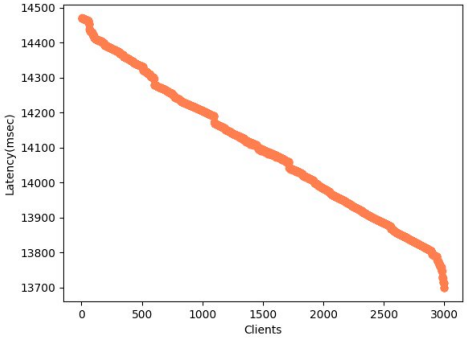
- **1000**





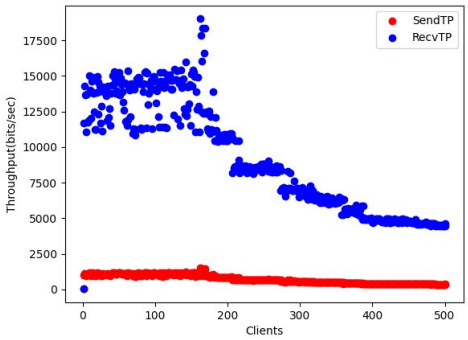
- **3000**

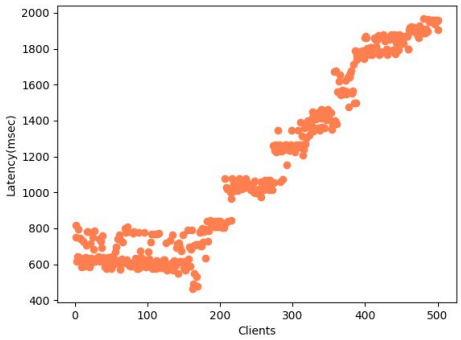




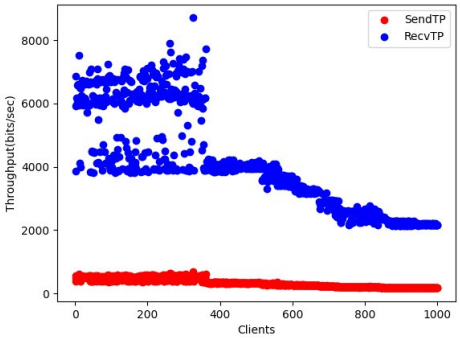
MULTITHREAD

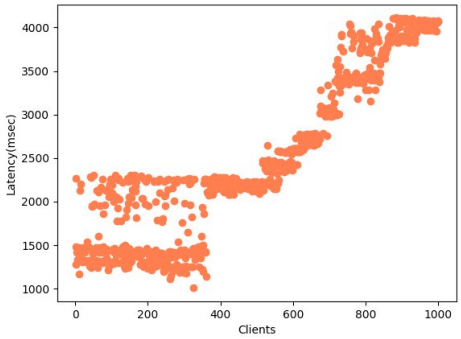
- **500**



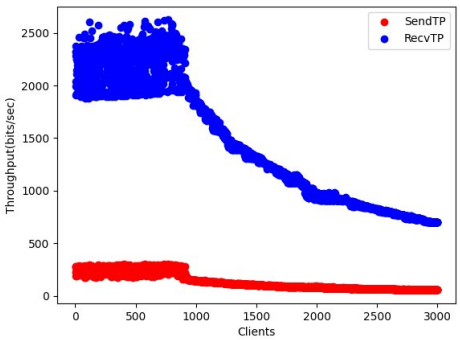


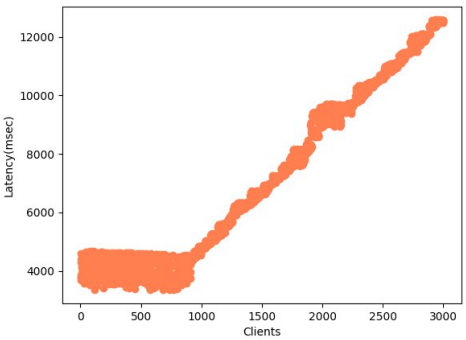
- **1000**





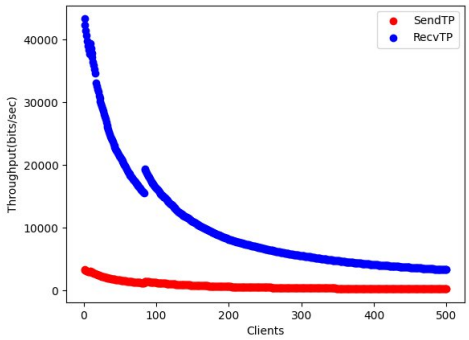
- **3000**

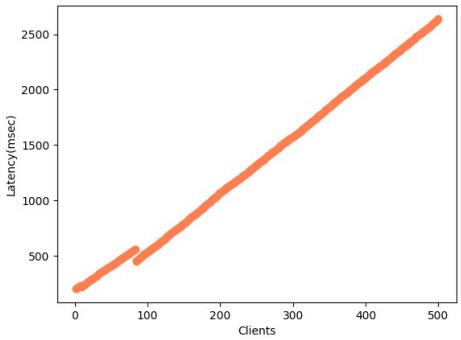




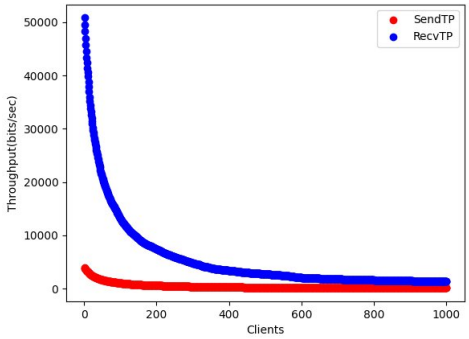
SELECT

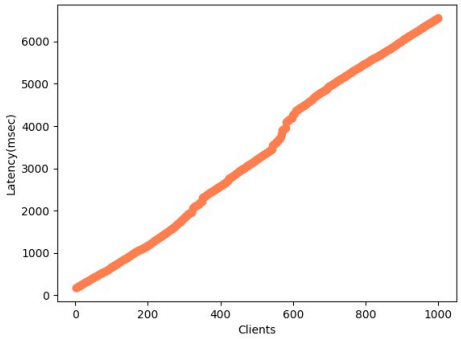
- **500**





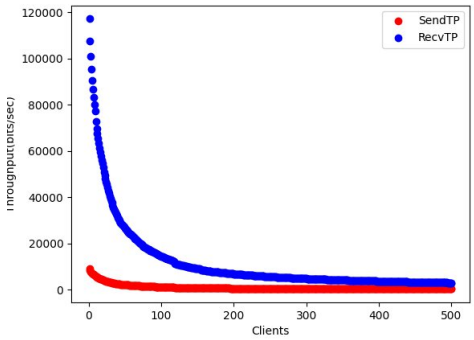
- **1000**

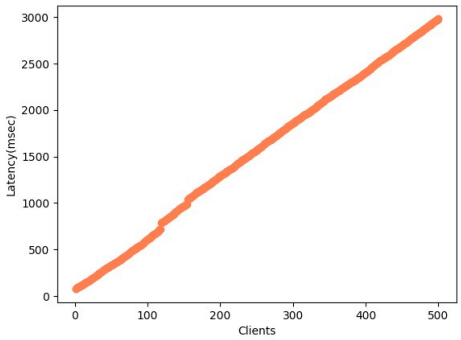




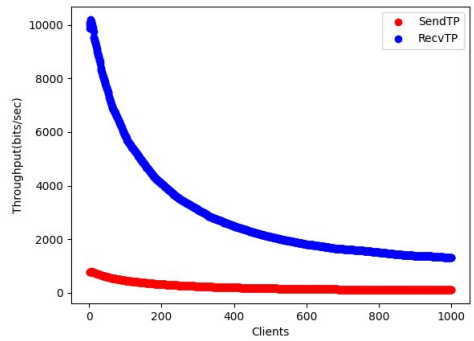
POLL

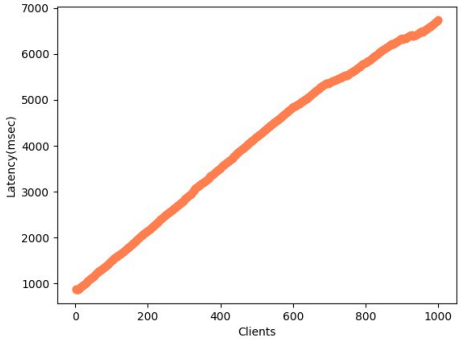
- **500**



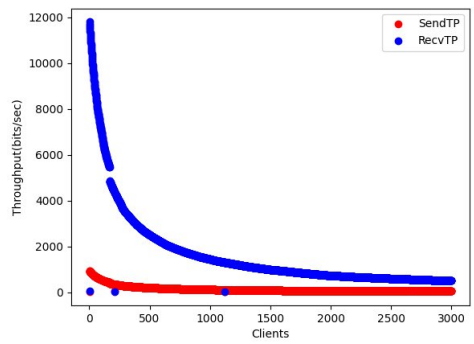


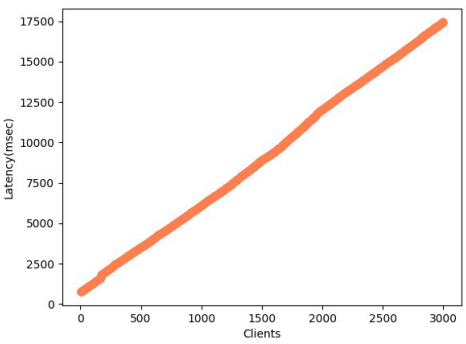
- **1000**





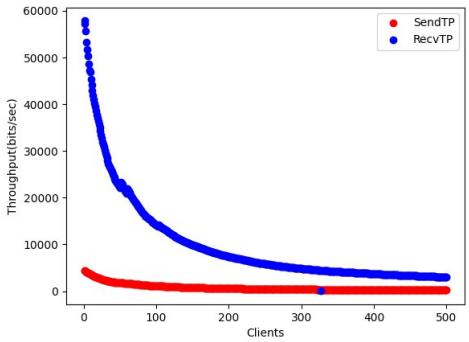
- **3000**

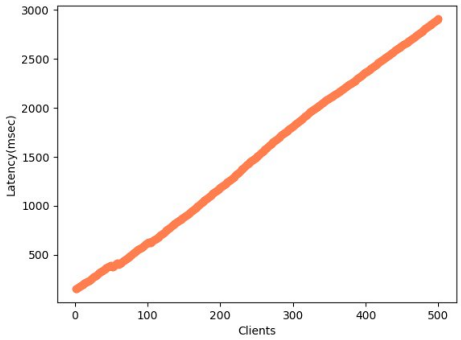




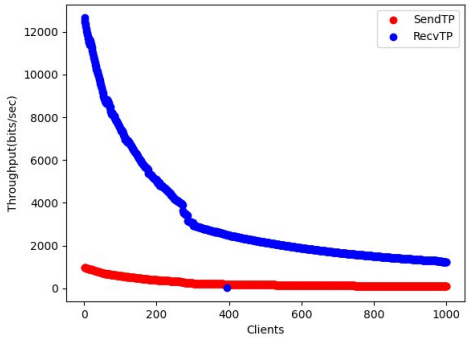
EPOLL

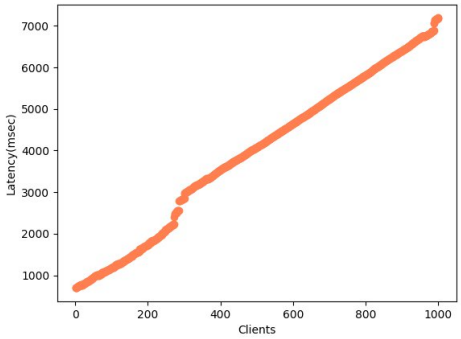
- **500**



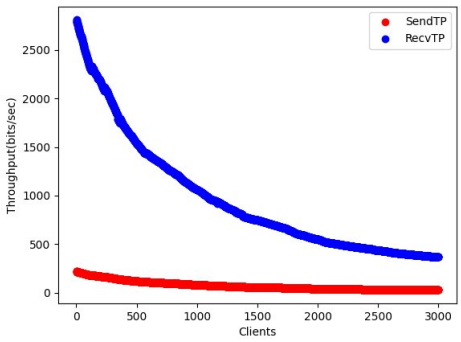


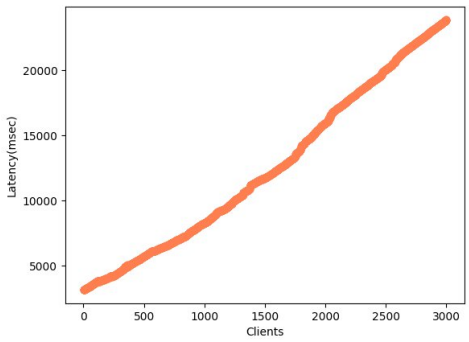
- **1000**





- **3000**





	Clients	CPU(%)	Memory(%)
Fork	500	57	18.4
	1000	61	52.6
	3000	63	~100
Multi-thread	500	57	3.4
	1000	62	21.4
	3000	66	45.1
Select	500	48	0.1
	1000	54	0.1
Poll	500	38	0.1
	1000	46	0.1
	3000	53	0.1
Epoll	500	32	0.1
	1000	50	0.1
	3000	52	0.1

Parallelism within concurrent processes is archived either by spawning processes or pooling threads, hence, memory usage is expected to shoot-up in case of fork and multi-threading. This increase is more prominent in case of fork since process is more memory costly than threads. In other cases, memory footprint is small because of a single running process. With increase in parallelism offered, throughput gets sacrificed.

CPU demand is expected to increase as number of clients increase.

Usually, as the program runs, the CPU utilization decreases.