AQM-LLM

This document is intended for IoT lab researchers. For details, please refer to the AQM-LLM documentation.

This document is used to quickly build the experimental structure, not to quickly complete all experiments.

Preparation

There are 3 VMs in total:

VM_ BBR2: Contains BBR2 congestion control algorithm

VM_ BBR3: Contains BB3 congestion control algorithm

VM_L4S: Contains everything about the L4S architecture, but only for the VMs for Client and Server (**TCP Prague congestion control algorithm and ECN enabled**). Routers need to manually install the L4S architecture and enable **DualPI2 AQM**.

The VM backup is only accessible to the Deakin account, and I don't have permission to change it to anyone else.

VM_BBR2 OneDrive link: VM bbr2

VM_BBR3 OneDrive link: VM bbr3

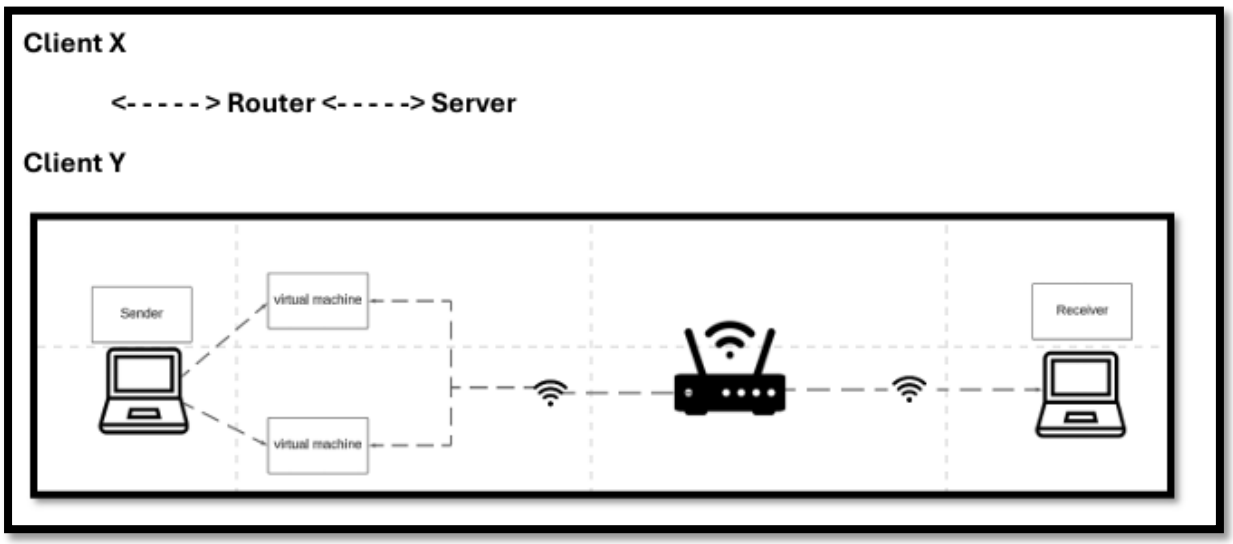
VM_L4S OneDrive link: Ubuntu 64 prague

The su password and login password for the VM are both **SIT7232024**

Process

Hardware: 2 Windows computers (PC1 and PC2), 1 Ubuntu computer or Linux-supported router.

Software: VMware workstation Pro (17th generation is a personal free version, I use 16th generation, any differences need to be solved by yourself.)



Roles: Client, Router, Server.

Client X is a client that supports L4S architecture, and Client Y is a client that does not support L4S architecture.

Client X uses VM_L4S, and Client Y uses VM_BBR, VM_BBR2, or VM_BBR3 according to your experimental requirements.

Router requires you to install L4S architecture on MikroTik router or Ubuntu system computer.

Server uses VM_L4S.

Open VMware workstation Pro on PC1 and PC2 and start VM_L4S on both VMware.

Enter the following command in the VMs terminal

Server side：

iperf3 -s -p 3000

iperf3 -s -p 3001

Client X side：

iperf3 -c [ip address] -p 3000 -C prague -tinf

Client Y side：

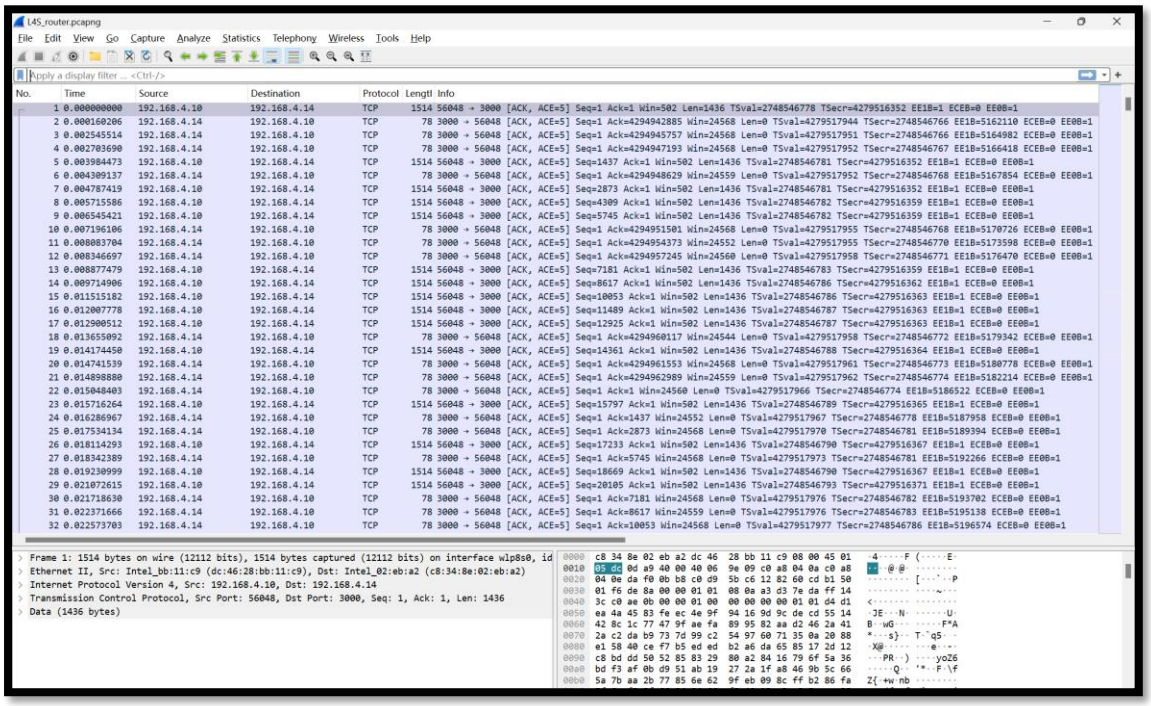
iperf3 -c [ip address] -p 3001 -C cubic -tinf

[ip address] is the IP address of your Server side.

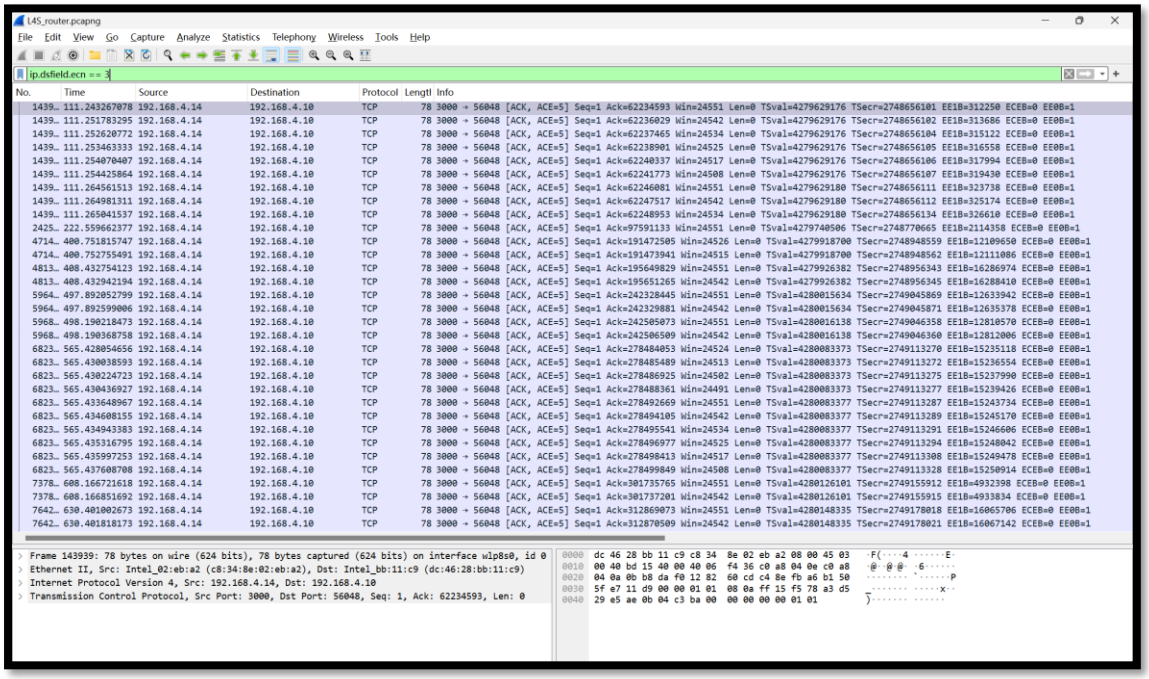
Launch Wireshark on Router or Server side, to check L4S is enable or not.

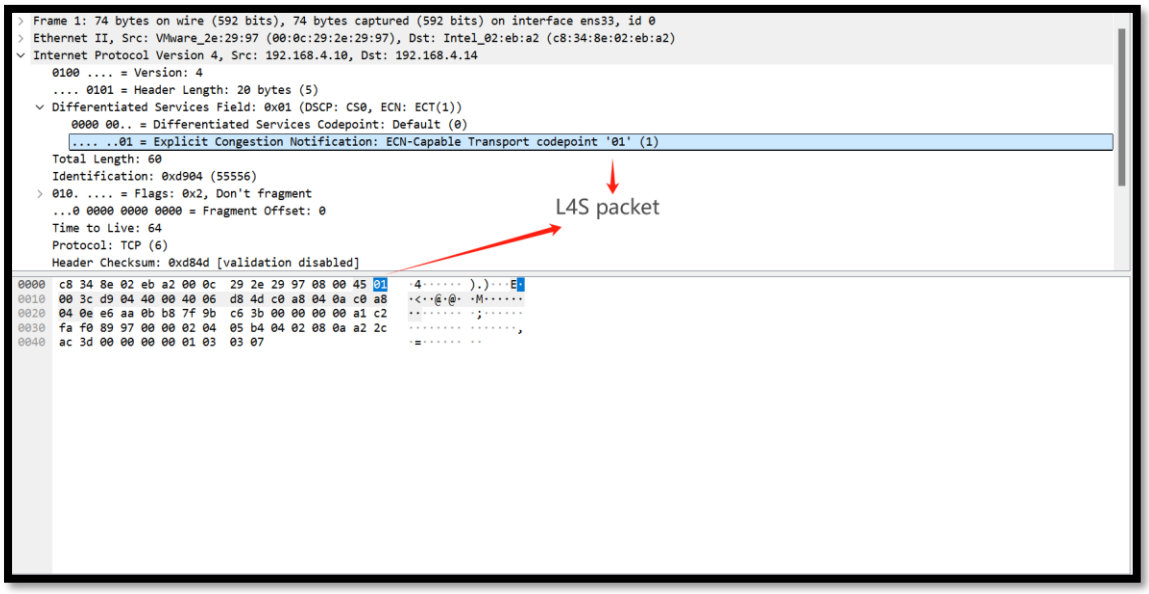
After iperf3 connects properly, please keep Clients connected to Server. Then start Wireshark on Server side or Router side, after opening Wireshark enter

ip.dsfield.ecn == 3



As long as you can see the result after typing it, it means that ECN=3 marked packets appeared, it also means that DualPI2 is marking the congested packets with ECN. please let iperf3 connect for a longer period of time, in my previous tests congestion appeared at a ratio of about 1:6000 or 20,000th packets before the first congestion marking appeared.





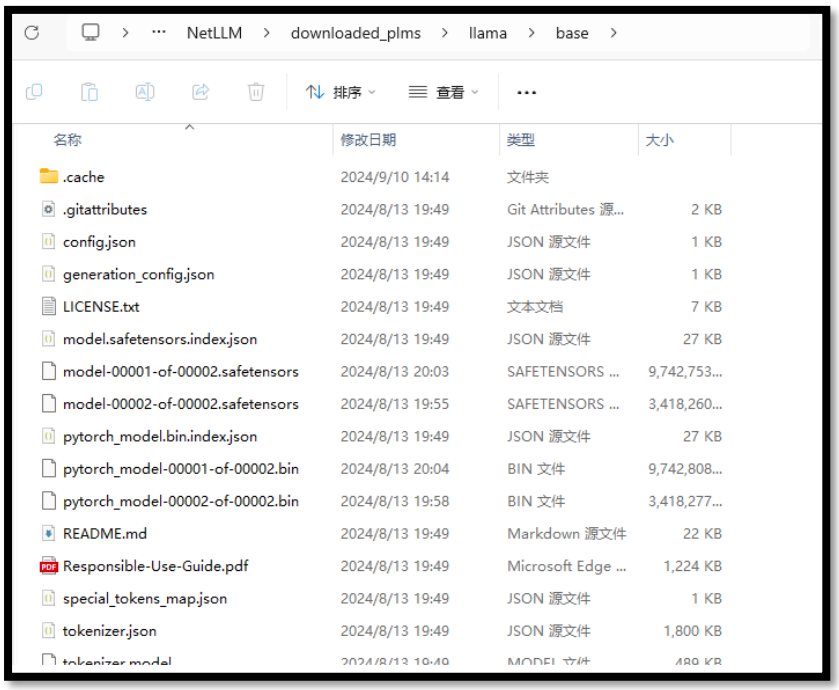
Checkpoint: If you can see packets with ECN=3, it means that the L4S architecture has been enabled. You have completed the setup part of the experiment. The next step is to collect L4S data, convert the collected data into PKL experience pool files that LLM can understand, and finally train in RunPod.

Next, please follow the AQM-LLM Documentation to download the files required for the LLM model and NetLLM architecture. (If you are in a hurry, please seek help from Deol in the IoT lab. The workstation computer theoretically has the complete NetLLM + LLM files.)

After you download the Llama2-7b-hf and NetLLM files, the structure of the entire trained model should look like this:

Open the NetLLM folder:

Adaptive_bitrate_steaming is all the files needed to train LLM and downloaded_plms is the location where the LLM (Llama2-7b-hf) model is stored. The directory where the LLM is saved is:



The process of renting RunPod is not explained here. If you have a local GPU that can train NetLLM, you don’t need to rent RunPod. It is recommended to read the process of AQM-LLM Documentation before quick operation.

Use the scp command to upload the NetLLM folder to RunPod. Be sure to upload it to cd /workspace, otherwise all storage contents will be erased when you close RunPod. cd to the adaptive_bitrate_streaming folder and use the following command:

python -m pip install --upgrade pip && pip install openprompt==1.0.1 && pip install numpy==1.24.4 && pip install peft==0.6.2 && pip install transformers==4.34.1 && pip install --upgrade huggingface_hub && pip install scikit-learn && pip install munch

python run_plm.py --adapt --grad-accum-steps 32 --plm-type llama --plm-size base --rank 128 --device cuda:0 --lr 0.0001 --warmup-steps 2000 --num-epochs 80 --eval-per-epoch 2

This process is trained entirely using the official NetLLM file, just to familiarize you with how the entire NetLLM and LLM work. You don't have to run 80 iterations.

Next, you need to modify the PKL file yourself, modify other codes to remove the restrictions on the model's PKL content, and then train. The specific process AQM-LLM Documentation has explained it.