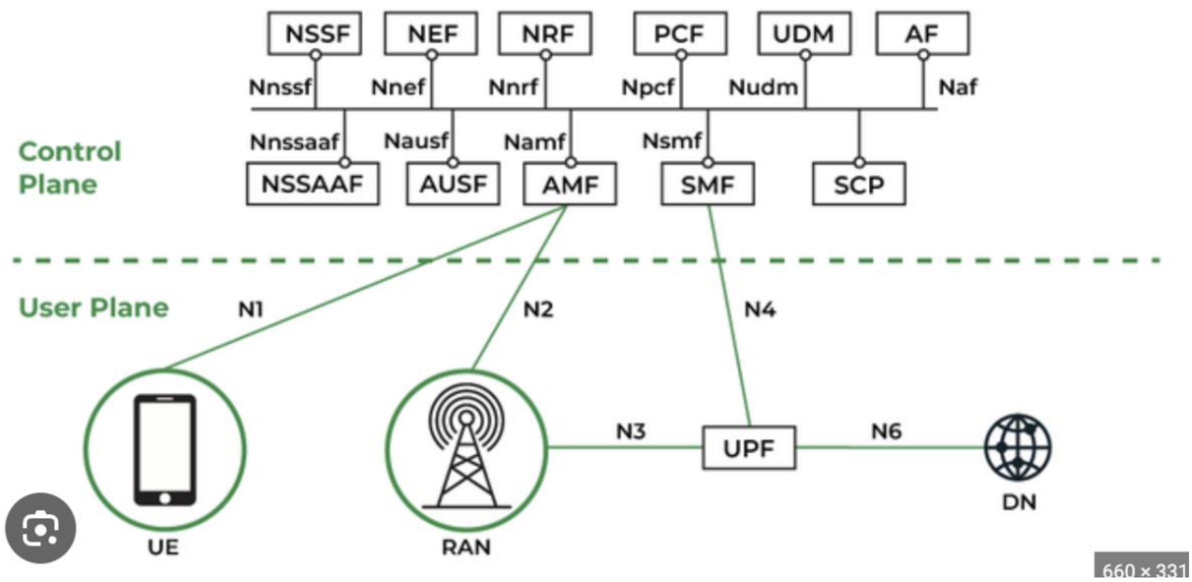
Sharanya Shashikumar
Documentation on 5G SA Network Setup
Andro Computational Solutions

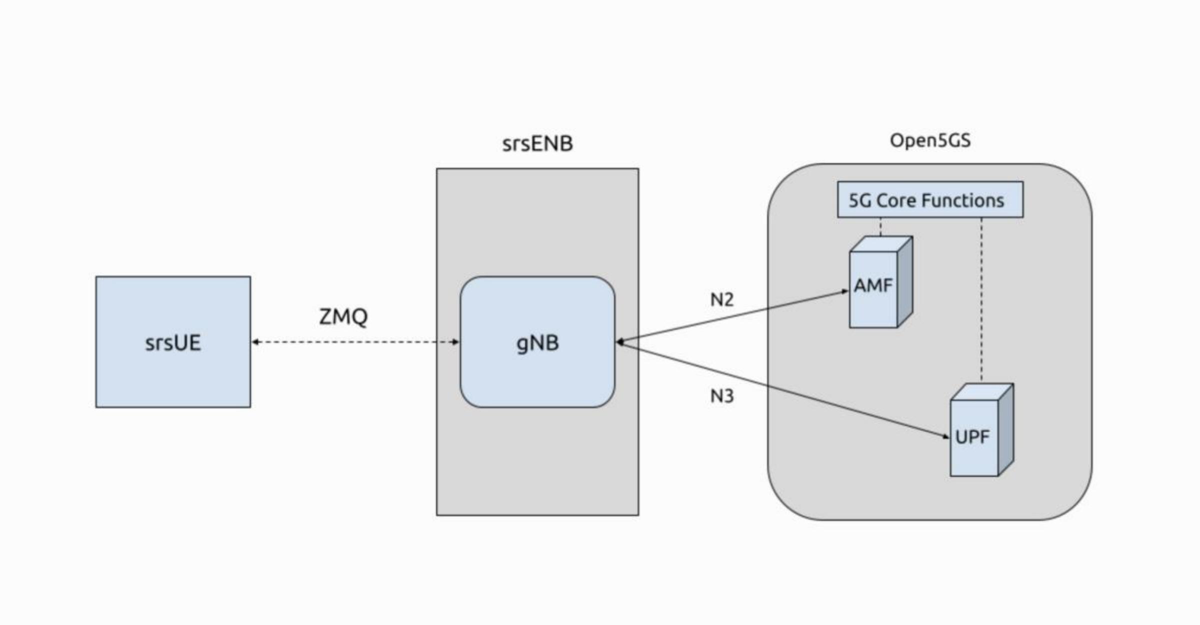
Introduction:
A 5G Standalone (SA) network is the latest generation of mobile communication infrastructure that operates independently, without relying on existing 4G networks. It offers very fast data transfer speeds, ultra-low latency, and the ability to connect a massive number of devices, making it well-suited for a wide range of applications, from augmented reality and IoT to critical services like remote surgery and autonomous vehicles. With features like network slicing, beamforming, and edge computing, 5G SA has the potential to revolutionize communication, enabling innovative solutions across various industries and promising a more versatile and reliable wireless future.

Technologies and OS:
- Ubuntu 20.04.4
- srsRAN 22.04
- Open 5gsCore
- ZeroMQ Mode of operation

Network Architecture of 5G



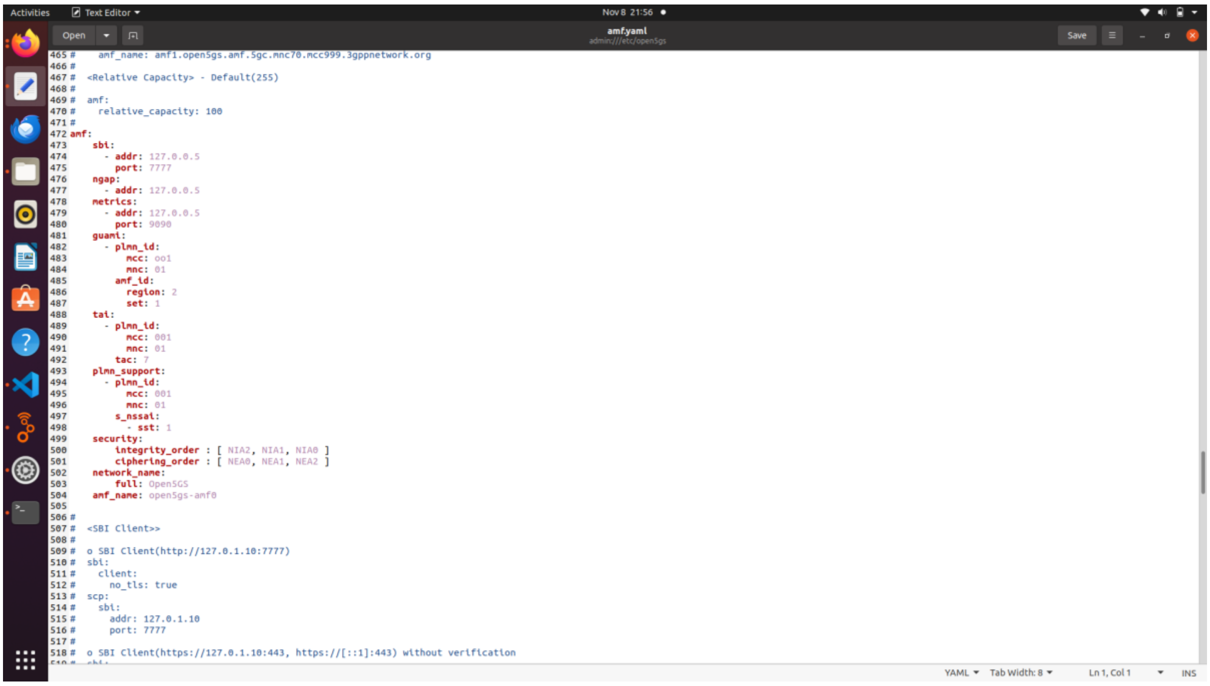
Flow Diagram used to establish E2E 5G Network.



Necessary Installation steps

1. To support 5G SA, it's important to have the latest version of srsRAN. I upgraded the srsRAN version to 22.04 to support 5G SA.
2. Installed and Configured MongoDB: MongoDB is a NoSQL database required by Open5GS for storing subscriber data and network-related information.
3. Install Open5GS with a Package Manager: Open5GS is the core network component for 5G SA networks.
4. Install WebUI for Open5GS: The WebUI for Open5GS provides a graphical interface for managing the core network.

Configuration changes made to amf.yaml file:

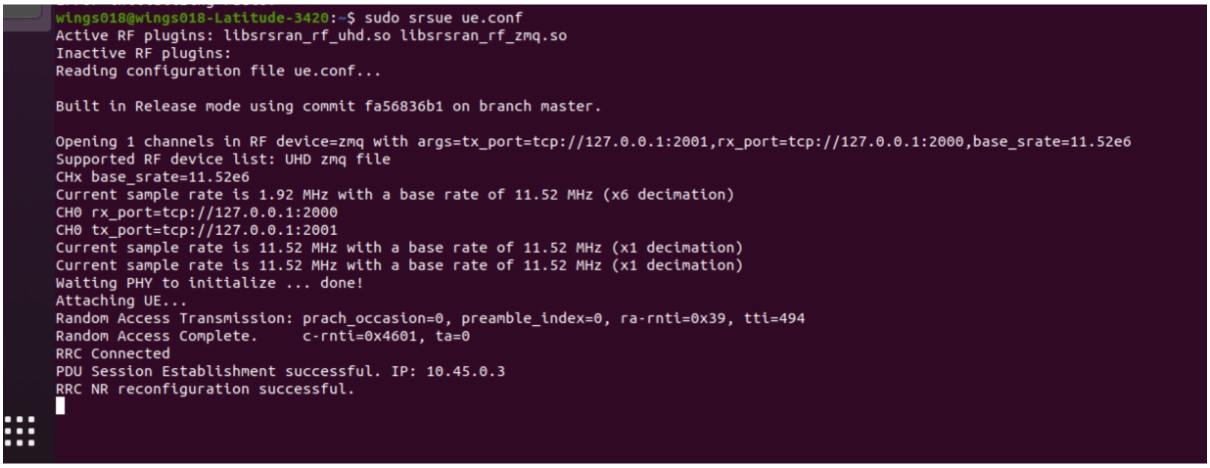


Network establishment and E2E Tests

Command used to run gNB:

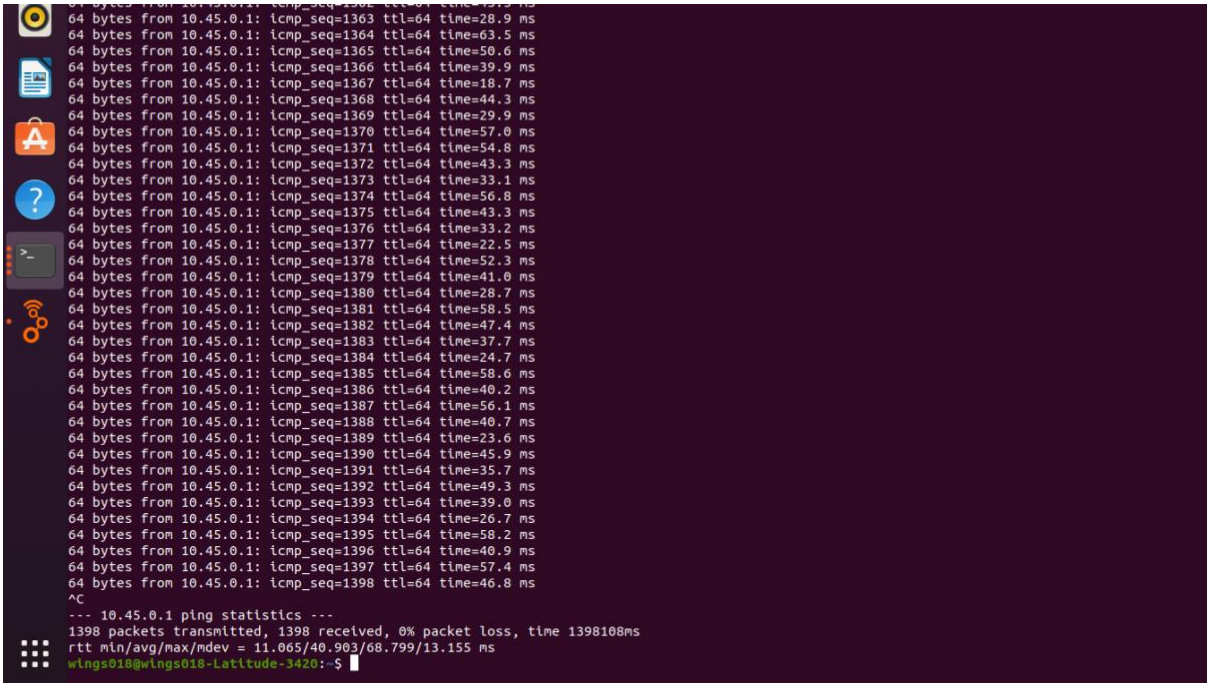
gNB is successfully connecting to the AMF.

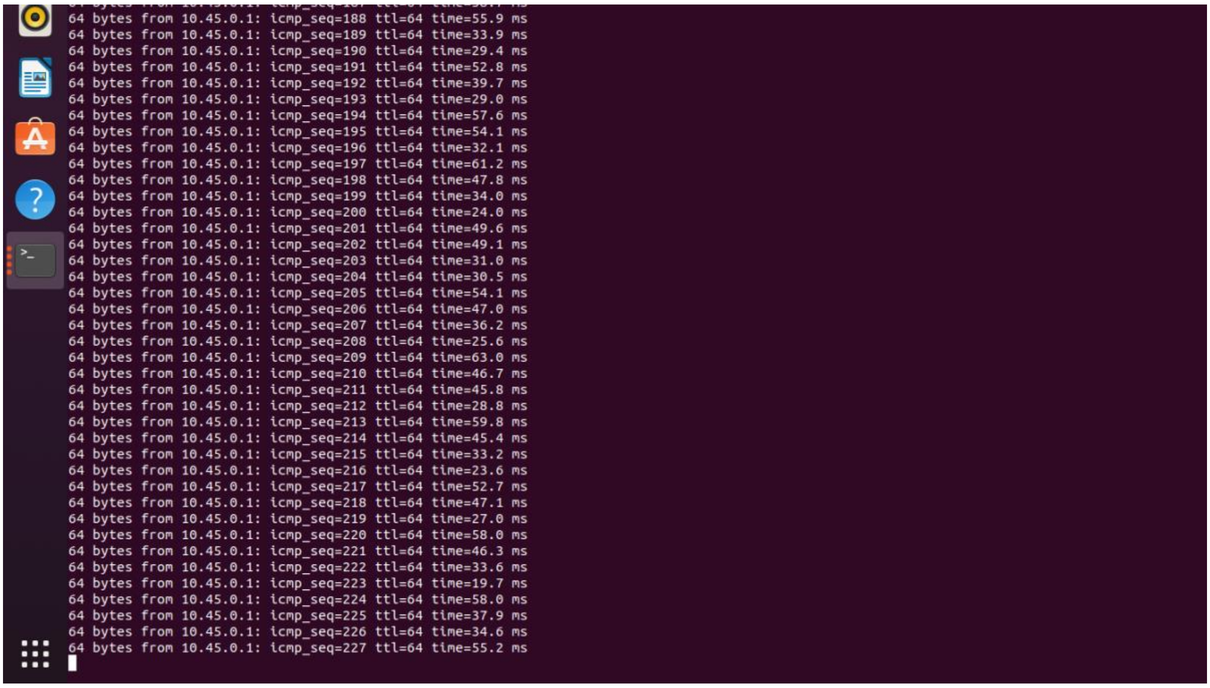
Command used to run UE:



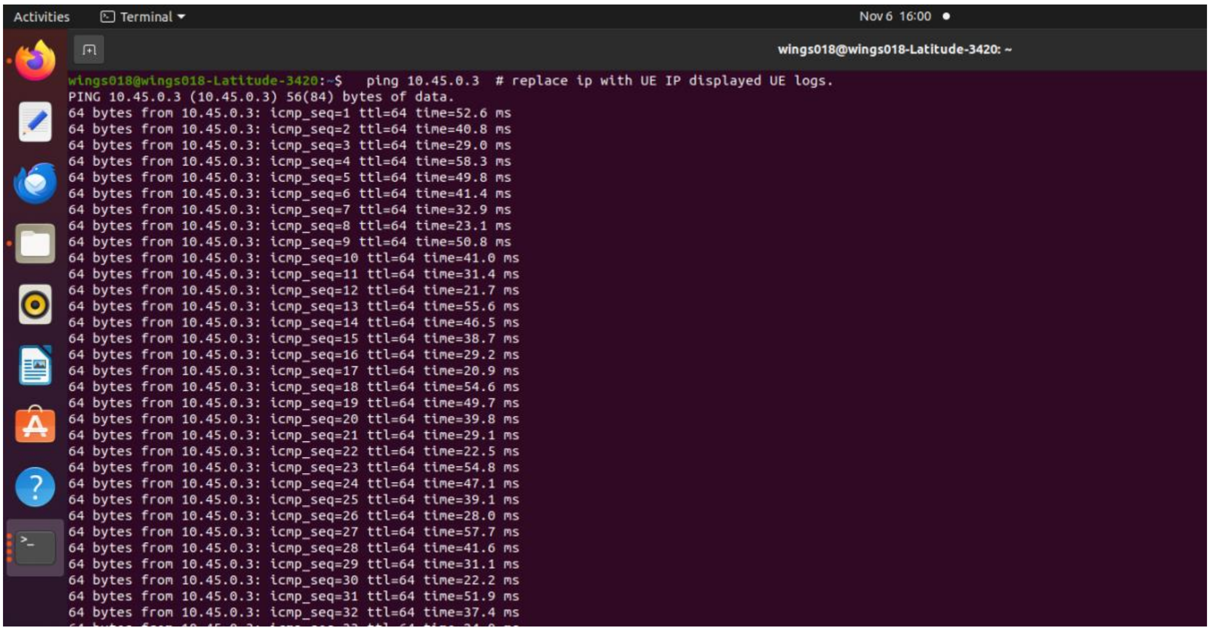
RRC connection and PDC session Establishment successful.

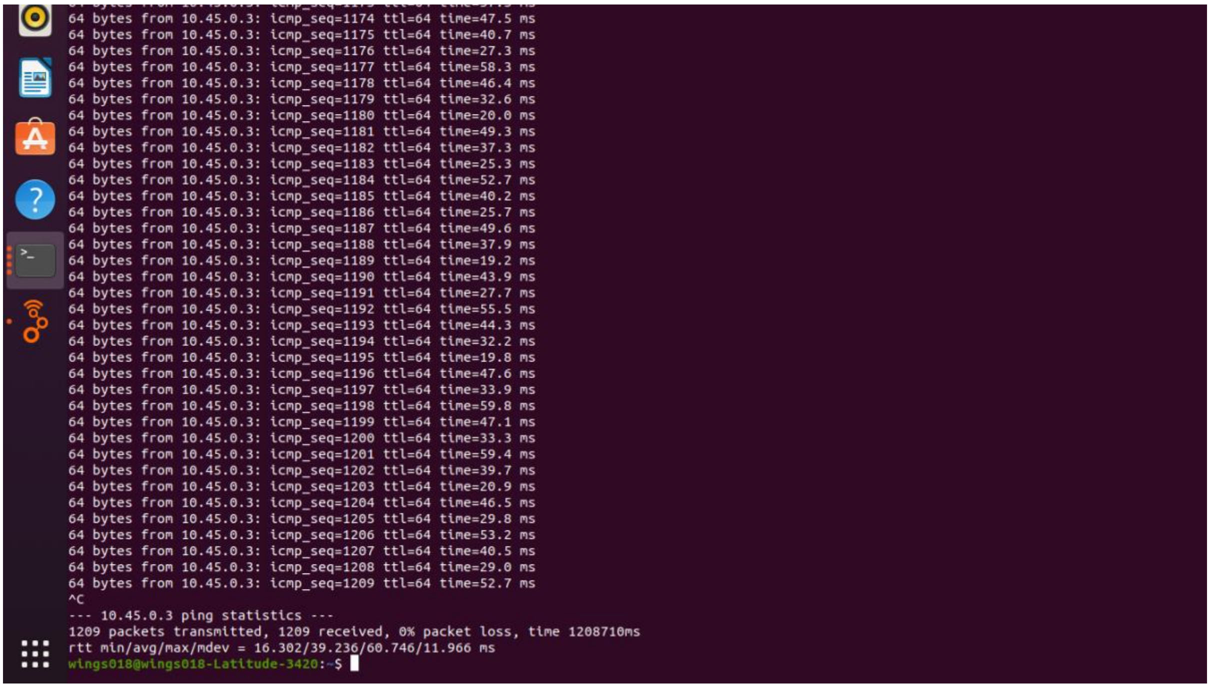
Uplink Ping


64 bytes from 10.45.0.1: icmp_seq=1363 ttl=64 time=28.9 ms
64 bytes from 10.45.0.1: icmp_seq=1364 ttl=64 time=63.5 ms
64 bytes from 10.45.0.1: icmp_seq=1365 ttl=64 time=50.6 ms
64 bytes from 10.45.0.1: icmp_seq=1366 ttl=64 time=39.9 ms
64 bytes from 10.45.0.1: icmp_seq=1367 ttl=64 time=18.7 ms
64 bytes from 10.45.0.1: icmp_seq=1368 ttl=64 time=44.3 ms
64 bytes from 10.45.0.1: icmp_seq=1369 ttl=64 time=29.9 ms
64 bytes from 10.45.0.1: icmp_seq=1370 ttl=64 time=57.0 ms
64 bytes from 10.45.0.1: icmp_seq=1371 ttl=64 time=54.8 ms
64 bytes from 10.45.0.1: icmp_seq=1372 ttl=64 time=43.3 ms
64 bytes from 10.45.0.1: icmp_seq=1373 ttl=64 time=33.1 ms
64 bytes from 10.45.0.1: icmp_seq=1374 ttl=64 time=56.8 ms
64 bytes from 10.45.0.1: icmp_seq=1375 ttl=64 time=43.3 ms
64 bytes from 10.45.0.1: icmp_seq=1376 ttl=64 time=33.2 ms
64 bytes from 10.45.0.1: icmp_seq=1377 ttl=64 time=22.5 ms
64 bytes from 10.45.0.1: icmp_seq=1378 ttl=64 time=52.3 ms
64 bytes from 10.45.0.1: icmp_seq=1379 ttl=64 time=41.0 ms
64 bytes from 10.45.0.1: icmp_seq=1380 ttl=64 time=28.7 ms
64 bytes from 10.45.0.1: icmp_seq=1381 ttl=64 time=58.5 ms
64 bytes from 10.45.0.1: icmp_seq=1382 ttl=64 time=47.4 ms
64 bytes from 10.45.0.1: icmp_seq=1383 ttl=64 time=37.7 ms
64 bytes from 10.45.0.1: icmp_seq=1384 ttl=64 time=24.7 ms
64 bytes from 10.45.0.1: icmp_seq=1385 ttl=64 time=58.6 ms
64 bytes from 10.45.0.1: icmp_seq=1386 ttl=64 time=40.2 ms
64 bytes from 10.45.0.1: icmp_seq=1387 ttl=64 time=56.1 ms
64 bytes from 10.45.0.1: icmp_seq=1388 ttl=64 time=40.7 ms
64 bytes from 10.45.0.1: icmp_seq=1389 ttl=64 time=23.6 ms
64 bytes from 10.45.0.1: icmp_seq=1390 ttl=64 time=45.9 ms
64 bytes from 10.45.0.1: icmp_seq=1391 ttl=64 time=35.7 ms
64 bytes from 10.45.0.1: icmp_seq=1392 ttl=64 time=49.3 ms
64 bytes from 10.45.0.1: icmp_seq=1393 ttl=64 time=39.0 ms
64 bytes from 10.45.0.1: icmp_seq=1394 ttl=64 time=26.7 ms
64 bytes from 10.45.0.1: icmp_seq=1395 ttl=64 time=58.2 ms
64 bytes from 10.45.0.1: icmp_seq=1396 ttl=64 time=40.9 ms
64 bytes from 10.45.0.1: icmp_seq=1397 ttl=64 time=57.4 ms
64 bytes from 10.45.0.1: icmp_seq=1398 ttl=64 time=46.8 ms
^C
--- 10.45.0.1 ping statistics ---
1398 packets transmitted, 1398 received, 0% packet loss, time 1398108ms
rtt min/avg/max/mdev = 11.065/40.903/68.799/13.155 ms
wings018@wings018-Latitude-3420:~$


64 bytes from 10.45.0.1: icmp_seq=188 ttl=64 time=55.9 ms
64 bytes from 10.45.0.1: icmp_seq=189 ttl=64 time=33.9 ms
64 bytes from 10.45.0.1: icmp_seq=190 ttl=64 time=29.4 ms
64 bytes from 10.45.0.1: icmp_seq=191 ttl=64 time=52.8 ms
64 bytes from 10.45.0.1: icmp_seq=192 ttl=64 time=39.7 ms
64 bytes from 10.45.0.1: icmp_seq=193 ttl=64 time=29.0 ms
64 bytes from 10.45.0.1: icmp_seq=194 ttl=64 time=57.6 ms
64 bytes from 10.45.0.1: icmp_seq=195 ttl=64 time=54.1 ms
64 bytes from 10.45.0.1: icmp_seq=196 ttl=64 time=32.1 ms
64 bytes from 10.45.0.1: icmp_seq=197 ttl=64 time=61.2 ms
64 bytes from 10.45.0.1: icmp_seq=198 ttl=64 time=47.8 ms
64 bytes from 10.45.0.1: icmp_seq=199 ttl=64 time=34.0 ms
64 bytes from 10.45.0.1: icmp_seq=200 ttl=64 time=24.0 ms
64 bytes from 10.45.0.1: icmp_seq=201 ttl=64 time=49.6 ms
64 bytes from 10.45.0.1: icmp_seq=202 ttl=64 time=49.1 ms
64 bytes from 10.45.0.1: icmp_seq=203 ttl=64 time=31.0 ms
64 bytes from 10.45.0.1: icmp_seq=204 ttl=64 time=30.5 ms
64 bytes from 10.45.0.1: icmp_seq=205 ttl=64 time=54.1 ms
64 bytes from 10.45.0.1: icmp_seq=206 ttl=64 time=47.0 ms
64 bytes from 10.45.0.1: icmp_seq=207 ttl=64 time=36.2 ms
64 bytes from 10.45.0.1: icmp_seq=208 ttl=64 time=25.6 ms
64 bytes from 10.45.0.1: icmp_seq=209 ttl=64 time=63.0 ms
64 bytes from 10.45.0.1: icmp_seq=210 ttl=64 time=46.7 ms
64 bytes from 10.45.0.1: icmp_seq=211 ttl=64 time=45.8 ms
64 bytes from 10.45.0.1: icmp_seq=212 ttl=64 time=28.8 ms
64 bytes from 10.45.0.1: icmp_seq=213 ttl=64 time=59.8 ms
64 bytes from 10.45.0.1: icmp_seq=214 ttl=64 time=45.4 ms
64 bytes from 10.45.0.1: icmp_seq=215 ttl=64 time=33.2 ms
64 bytes from 10.45.0.1: icmp_seq=216 ttl=64 time=23.6 ms
64 bytes from 10.45.0.1: icmp_seq=217 ttl=64 time=52.7 ms
64 bytes from 10.45.0.1: icmp_seq=218 ttl=64 time=47.1 ms
64 bytes from 10.45.0.1: icmp_seq=219 ttl=64 time=27.0 ms
64 bytes from 10.45.0.1: icmp_seq=220 ttl=64 time=58.0 ms
64 bytes from 10.45.0.1: icmp_seq=221 ttl=64 time=46.3 ms
64 bytes from 10.45.0.1: icmp_seq=222 ttl=64 time=33.6 ms
64 bytes from 10.45.0.1: icmp_seq=223 ttl=64 time=19.7 ms
64 bytes from 10.45.0.1: icmp_seq=224 ttl=64 time=58.0 ms
64 bytes from 10.45.0.1: icmp_seq=225 ttl=64 time=37.9 ms
64 bytes from 10.45.0.1: icmp_seq=226 ttl=64 time=34.6 ms
64 bytes from 10.45.0.1: icmp_seq=227 ttl=64 time=55.2 ms

Downlink Ping





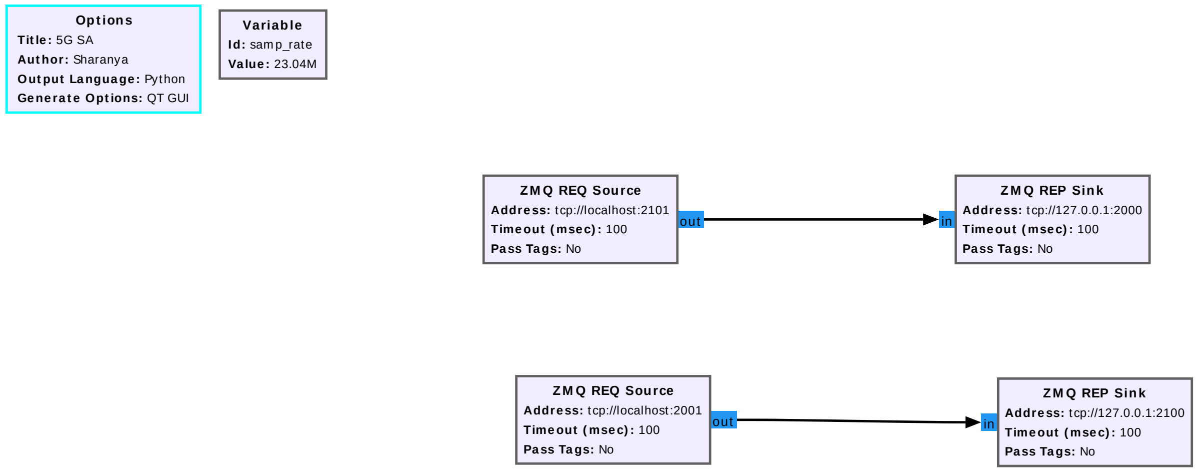
Troubleshooting

Initial Setup Challenge: Initially, I encountered challenges while setting up the 5G network, particularly when trying to run Open5GS, which serves as the core component for 5G Standalone (5G SA) networks.

AMF Configuration Issue: The primary issue I faced was related to the AMF configuration file. When I checked the logs I understood that the AMF was not getting connected to the open5gs core. This is because the amf.yaml configuration file was not properly configured, specifically the mnc , mcc and the tac. This was crucial for the network setup.

Research: In order to address this issue, I looked for online resources, and documentations and also the video links shared by Anu when I emailed her about the issue to understand the AMF configuration requirements and how to resolve the problem.
Solution: During my research, I came across a website that provided valuable insights and resources for 5G network setup. This website not only offered the correct amf.yaml configuration file but also detailed the specific changes required to make the network fully operational.

GNUradio companion block diagram.



Python script that was generated when I ran the particular .grc file.

When I run the 5G Set up that is start the gNB and the UE, followed by executing the grc file, the TCP ports used by the 5G network are not syncing with the grc. I identified this issue when I got the error "Address already in use" in the grc window while trying to execute.
In order to re confirm if this is the actual issue, I terminated the 5G network and ran the grc, this time the blocks executed but since there was no signal exchanged between the gNB and UE, running the generated python script fetched nothing. At the same time, I tried setting up the 5G network while the grc was still running. Since the ports are not syncing, the gNB was unable to connect to the AMF and hence the 5G network did not get established. I've explored various solutions, but unfortunately, none of them have resolved the issue. Whether I ran the 5G network separately or tried to establish it while the GNU Radio Companion (GRC) was running, the problem of unsynchronized TCP ports persisted. This prevented the gNB from connecting to the AMF and, consequently, the 5G network could not be established. Despite my best efforts, I was unable to overcome the 'Address already in use' error, leaving this port synchronization challenge unresolved.

Since I was unable to get the required .dat file containing the baseband samples from UE, I continued working on the code part by taking a complex signal as input and computed the STFT and PSD for that input signal.

Below is a short explanation on the logic used for the code.

1. Import necessary libraries for FFT (Fast Fourier Transform), threading, vector manipulation, and plotting.
2. Create mutexes for thread synchronization.
3. computeSTFT Function by Iterating over the input signal in frames with a hop size of hopSize. For each frame, converting it to a complex vector and perform FFT using FFTW library. Later storing the resulting spectrum in the stftBuffer.

4. computePSD Function by extracting a frame from the input signal and perform FFT to obtain the spectrum. Calculating the Power Spectrum Density by taking the squared norm of the spectrum and updating the global psdBuffer with the computed PSD.
5. Use Gnuplot to plot the STFT data in a 3D surface plot.
6. Use Gnuplot to plot the PSD data in a 2D line plot.
7. Main Function: Initialize the input signal with a cosine waveform.
   Create two threads, one for displaying the STFT and the other for displaying the PSD.