

Universidad
Nacional
de Loja

Universidad Nacional de Loja

Facultad de la Energía, las Industrias y los Recursos Naturales no Renovables

Carrera de Ingeniería en Telecomunicaciones

Diseño e implementación de un prototipo de arquitectura 5G-NR para servicios de voz y datos basado en SDR (Software Defined Radio) y software Open Source

Trabajo de Integración Curricular,
previo a la obtención del título de
Ingeniero en Telecomunicaciones.

AUTOR:

Juan Pablo Rivas Mora

DIRECTOR:

Ing. Juan Gabriel Ochoa Aldeán, Mg. Sc

Loja – Ecuador

2024

Certificación

Loja, 9 de enero de 2024

Ing. Juan Gabriel Ochoa Aldeán, Mg. Sc

DIRECTOR DEL TRABAJO DE INTEGRACIÓN CURRICULAR

CERTIFICO:

Que he revisado y orientado todo el proceso de elaboración del Trabajo de Integración Curricular denominado: **Diseño e implementación de un prototipo de arquitectura 5G NR para servicios de voz y datos basado en SDR (Software Defined Radio) y software Open Source**, previo a la obtención del título de **Ingeniero en Telecomunicaciones**, de la autoría del estudiante **Juan Pablo Rivas Mora**, con **cédula de identidad Nro. 1104242332**, una vez que el trabajo cumple con todos los requisitos exigidos por la Universidad Nacional de Loja para el efecto, autorizo la presentación del mismo para su respectiva sustentación y defensa.

Ing. Juan Gabriel Ochoa Aldeán, Mg. Sc

DIRECTOR DEL TRABAJO DE INTEGRACIÓN CURRICULAR

Autoría

Yo, **Juan Pablo Rivas Mora**, declaro ser el autor del presente Trabajo de Integración Curricular y eximo expresamente a la Universidad Nacional de Loja y a sus representantes jurídicos de posibles reclamos y acciones legales, por el contenido del mismo. Adicionalmente acepto y autorizo a la Universidad Nacional de Loja la publicación de mi Trabajo de Integración Curricular en el Repositorio Digital Institucional – Biblioteca Virtual.

Firma:

Cédula de identidad: 1104242332

Fecha: 19 de febrero de 2024

Correo electrónico: juan.rivas@unl.edu.ec

Teléfono: 0986663810

Carta de autorización por parte del autor, para la consulta, reproducción parcial o total y/o publicación electrónica del texto completo del Trabajo de Integración Curricular.

Yo, **Juan Pablo Rivas Mora**, declaro ser autor del Trabajo de Integración Curricular denominado: **Diseño e implementación de un prototipo de arquitectura 5G-NR para servicios de voz y datos basado en SDR (Software Defined Radio) y software Open Source**, como requisito para optar el título de **Ingeniero en Telecomunicaciones**, autorizo al sistema Bibliotecario de la Universidad Nacional de Loja para que, con fines académicos, muestre la producción intelectual de la Universidad, a través de la visibilidad de su contenido en el Repositorio Institucional.

Los usuarios pueden consultar el contenido de este trabajo en el Repositorio Institucional, en las redes de información del país y del exterior con las cuales tenga convenio la Universidad. La Universidad Nacional de Loja, no se responsabiliza por el plagio o copia del Trabajo de Integración Curricular que realice un tercero.

Para constancia de esta autorización, suscribo, en la ciudad de Loja, a los diecinueve días del mes de febrero del dos mil veinticuatro.

Firma:

Autor: Juan Pablo Rivas Mora

Cédula de identidad: 1104242332

Dirección: Av. José María Vivar Castro

Correo electrónico: juan.rivas@unl.edu.ec

Teléfono: 0986663810

DATOS COMPLEMENTARIOS:

Director Trabajo de Integración Curricular: Ing. Juan Gabriel Ochoa Aldeán. Mg. Sc

Dedicatoria

Deseo dedicar este Trabajo de Integración Curricular a toda mi familia, en especial a mis apreciados padres y queridos hermanos. Su apoyo incondicional y constante motivación han sido los pilares fundamentales sobre los cuales he edificado este logro. Su presencia y estímulo han iluminado mi camino a lo largo de mi travesía universitaria, impulsándome en cada paso que he dado. Sin su presencia inspiradora, no habría llegado tan lejos. Desde lo más profundo de mi corazón, les agradezco por ser la fuerza propulsora que ha impulsado mis sueños hacia la realidad.

Juan Pablo Rivas Mora

Agradecimiento

Expreso mi profundo agradecimiento a mis padres por su constante presencia y dedicación, fundamentales durante esta etapa crucial de mi vida. Su incansable esfuerzo para proporcionarme las herramientas necesarias y su apoyo incondicional a lo largo de mis años universitarios son invaluable y merecen mi más sincera gratitud.

También deseo reconocer al director de mi Trabajo de Integración Curricular, Ingeniero Juan Ochoa, cuya orientación y respaldo fueron esenciales en los momentos difíciles que enfrenté durante este proyecto de investigación. Sin su guía experta y aliento constante, no habría podido culminar este trabajo de manera exitosa.

Además, agradezco al Ingeniero Manuel Quiñónez por facilitar el acceso al laboratorio y equipos de la Universidad Técnica Particular de Loja, necesarios para llevar a cabo nuestras investigaciones de manera efectiva.

Mi gratitud hacia cada uno de ustedes es inmensa. Su contribución y respaldo fueron cruciales en la realización de este proyecto académico que ahora concluye exitosamente.

Juan Pablo Rivas Mora

Índice de contenidos

Portada	i
Certificación	ii
Autoría	iii
Carta de autorización	iv
Dedicatoria	v
Agradecimiento	vi
Índice de contenidos	vii
Índice de tablas:.....	x
Índice de figuras:.....	xi
Índice de anexos:.....	xiii
1. Título	1
2. Resumen	2
Abstract.....	3
3. Introducción	4
4. Marco teórico	7
4.1 5G NR (New Radio).....	7
4.1.1 Estandarización.....	7
4.1.2 Características 5G NR.....	8
4.1.3 Escenarios de uso de IMT-2020.....	9
4.2 Arquitectura 5G NR.....	11
4.2.1 5G NSA (Non-StandAlone).....	12
4.2.2 5G SA (StandAlone).....	13
4.2.3 5G Core Network.....	13
4.2.4 Radio-Access Network.....	15
4.3 Radio definida por software (SDR).....	16
4.3.1 Esquema genérico de la SDR.....	17

4.3.2	Arquitectura de la SDR	18
4.3.3	Dispositivos SDR	19
4.4	Implementaciones Open Source para redes 5G	21
4.4.1	Paquetes de software 5G RAN Open Source	21
4.4.2	Paquetes de software 5GC Open Source	23
5.	Metodología.....	25
5.1	Fases del proyecto	25
5.1.1	Fase 1: Investigación y recolección de información sobre 5G NR y SDR	25
5.1.2	Fase 2: Selección de hardware y software a emplear.....	26
5.1.3	Fase 3: Diseño e implementación del prototipo	26
5.1.4	Fase 4: Ejecución de pruebas del prototipo.....	27
5.2	Requerimientos de hardware y software.....	27
5.2.1	Hardware	27
5.2.2	Software	34
5.3	Diseño de la arquitectura 5G NR.....	37
5.3.1	Diseño propuesto.....	37
5.4	Implementación	41
5.4.1	Configuración de la SIM	42
5.4.2	Instalación del controlador de hardware BladeRF	43
5.4.3	Instalación del controlador de hardware USRP B210.....	44
5.4.4	Configuración de la red	45
5.4.5	Registro de suscriptores	51
5.4.6	Configuración del UE.....	52
5.4.7	Conexión a la red.....	55
5.4.8	Mejora del rendimiento de srsENB	60
5.5	Metodología de las pruebas	60

6.	Resultados	62
6.1	Test de datos	64
6.1.1	Resultados teóricos.....	64
6.1.2	Resultados experimentales	65
6.1.3	Acceso a Internet.....	68
6.1.4	Análisis de tráfico.....	70
6.2	Test de VoIP	75
6.3	Test de potencia: RX	80
6.4	Análisis de frecuencia y ancho de banda.....	82
6.5	Análisis de Costos	83
7.	Discusión	85
8.	Conclusiones	87
9.	Recomendaciones	89
10.	Bibliografía	91
11.	Anexos	96

Índice de tablas:

Tabla 1. Capacidades de los SDR y su integración con el software RAN	20
Tabla 2. Comparación de paquetes de software RAN 5G.....	22
Tabla 3. Comparación de paquetes de software 5G Core	24
Tabla 4. Características del computador portátil.....	28
Tabla 5. Características del SDR BladeRF 2.0 micro xA9	29
Tabla 6. Características del USRP B210.....	30
Tabla 7. Características de Antena Dipolo Banda Ancha	30
Tabla 8. Características Xiaomi Poco M3 Pro 5G	31
Tabla 9. Características de CardMan 3121 USB CCID interface	32
Tabla 10. Parámetros configurados en el gNB.....	61
Tabla 11. Resultados teóricos.....	65
Tabla 12. Resultados experimentales de la implementación.....	65
Tabla 13. Usuarios registrados al servidor SIP de Asterisk	76
Tabla 14. MOS (Mean Opinion Score)	79
Tabla 15. Resultados de RSRP, SINR, RSRQ en función de la distancia del gNB.....	81
Tabla 16. Estimación de costos del prototipo.....	84

Índice de figuras:

Figura 1. Escenarios de uso de IMT-2020	11
Figura 2. Bloques generales del sistema 5G	12
Figura 3. Arquitectura NSA	12
Figura 4. Arquitectura SA	13
Figura 5. Arquitectura de red central de alto nivel (descripción basada en servicios).....	14
Figura 6. Arquitectura NG-RAN.....	16
Figura 7. Arquitectura genérica de la SDR	17
Figura 8. Arquitectura ideal de SDR.....	19
Figura 9. Fases del proyecto.....	25
Figura 10. Lenovo ThinkPad L15 Gen 2	28
Figura 11. BladeRF 2.0 micro xA9.....	29
Figura 12. USRP B210.....	29
Figura 13. TECHTOO Antena 3G 4G de banda ancha.....	30
Figura 14. Xiaomi Poco M3 Pro 5G	31
Figura 15. sysmoISIM-SJA2.....	32
Figura 16. CardMan 3121 USB CCID interface.....	33
Figura 17. USRP GPS-Disciplined Oscillator Kit	34
Figura 18. Antena GPS activa de 3 voltios	34
Figura 19. Suite de softwares de srsRAN	35
Figura 20. Diagrama de componentes y funciones de red de Open5GS.....	36
Figura 21. Diagrama general de los componentes de hardware y software para los escenarios de pruebas.....	38
Figura 22. Arquitectura de los componentes del núcleo de 5G empleado.....	38
Figura 23. Despliegue de las comunicaciones en el equipo del gNB	39
Figura 24. Arquitectura de la red 5G SA propuesta.....	40
Figura 25. Descripción general de la red.....	41
Figura 26: Registro de suscriptores en Open5GS (WebUI).....	51
Figura 27. Ajustes de la tarjeta SIM.....	53
Figura 28. Configuración de APN en UE comercial.....	54
Figura 29. Tipo de red preferido: NR Only.....	55
Figura 30. Conexión del UE a la red Open5GS	58
Figura 31. Entorno empleado para el despliegue de 5G SA con USRP B210.....	62

Figura 32. Entorno empleado para el despliegue de 5G SA con BladeRF 2.0 micro Ax9	63
Figura 33. Entorno empleado para el despliegue de 5G SA con USRP B210 y GPSDO.....	64
Figura 34. Uplink teórico y Uplink experimental	66
Figura 35. Downlink teórico y Downlink experimental	67
Figura 36. Throughput alcanzado utilizando un MCS de 28	67
Figura 37. Throughput alcanzado utilizando un MCS de 5	68
Figura 38. Navegación por la web	69
Figura 39. Reproducción de videos.....	69
Figura 40. Descarga de aplicaciones	70
Figura 41. Gráfica de flujo del tráfico intercambiado entre el gNB y el AMF.....	71
Figura 42. Global RAN Node ID y RAN Node Name.....	71
Figura 43. AMF Name y PLMN Support List	72
Figura 44. User Location Information.....	72
Figura 45. Security protected 5GS NAS message.....	73
Figura 46. Dirección IP del UPF asociado al contexto y tipo de protocolo empleado en la conexión PDN	74
Figura 47. Mensajes HTTP para comprobar si hay salida a internet	74
Figura 48. Cuerpo de la petición GTP<HTTP> para la comprobación de salida a internet ...	75
Figura 49. Paquetes intercambiados entre el UE e internet empleando el protocolo GTP	75
Figura 50. Llamada VoIP recibida desde el cliente SIP Xiaomi Poco M3 Pro 5G.....	77
Figura 51. Estadísticas de red entre user1 (ext101) y user2 (ext102)	78
Figura 52. Llamada VoIP recibida desde el cliente SIP Lenovo Thinkpad L15 Gen2.....	78
Figura 53. Resultados estimados del MOS	79
Figura 54. Potencias promedio recibida a 0 metros de distancia de la gNB.....	80
Figura 55. Potencias promedio recibida a 8 metros de distancia de la gNB.....	81
Figura 56. Banda de frecuencia Downlink del gNB	82

Índice de anexos:

Anexo 1. Archivo de configuración enb.conf	96
Anexo 2. Archivo de configuración rr.conf	103
Anexo 3. Archivo de configuración amf.yaml.....	106
Anexo 4. Archivo de configuración upf.yaml.....	117
Anexo 5. Preparación del entorno	121
Anexo 6. Muestras de la medición de Thougput	123
Anexo 7. Muestras de la medición de potencia.....	124
Anexo 8. Certificado de Traducción del Resumen	125

1. Título

Diseño e implementación de un prototipo de arquitectura 5G NR para servicios de voz y datos basado en SDR (Software Defined Radio) y software Open Source

2. Resumen

Este proyecto de titulación se concentra en el diseño e implementación de un prototipo de arquitectura 5G NR mediante el uso de un dispositivo de SDR (Radio Definida por Software) y software de código abierto. El objetivo principal es llevar a cabo un estudio detallado y realizar pruebas en un entorno realista para evaluar la viabilidad y operatividad de una red privada 5G SA bajo esta configuración, abarcando servicios de voz sobre IP y datos. Además, se busca crear un entorno completamente funcional para propósitos educativos, el cual permita analizar este escenario como una situación auténtica, evitando la necesidad de simulaciones en cualquier parte de la arquitectura.

Para lograr este propósito, se realizó un minucioso análisis de las características y la arquitectura subyacente de las redes 5G. Asimismo, se llevó a cabo una comparación exhaustiva entre diversas opciones de hardware y software que permiten la implementación de una red privada 5G SA. Finalmente, se optó por desplegar el prototipo utilizando dos reconocidas plataformas de software de código abierto: srsRAN y Open5GS, para la implementación de RAN (Radio Access Network) y 5GC (5G Core) respectivamente. Se emplearon radios físicas en la ejecución de las transmisiones inalámbricas. Específicamente, se utilizaron los dispositivos SDR USRP B210 y Blade RF 2.0 con el fin de realizar una comparación de su rendimiento.

Tras la implementación de la red 5G, se llevaron a cabo pruebas exhaustivas en un entorno de laboratorio. Estas evaluaciones incluyeron la medición de la calidad del servicio de voz mediante la Puntuación de Opinión Media (MOS), así como el rendimiento de los enlaces descendentes y ascendentes utilizando un dispositivo de usuario convencional conectado a la red privada. Además, se realizaron pruebas de potencia para evaluar el comportamiento en diversas condiciones.

En resumen, este proyecto tiene como finalidad presentar un enfoque hacia las redes privadas basadas en SDR y software de código abierto. Esta solución se presenta como una opción económica, flexible y escalable. Por medio de la implementación y evaluación de un prototipo en un entorno realista, se pretende demostrar la viabilidad de estas tecnologías en el ámbito de las redes 5G y su potencial para aplicaciones prácticas y educativas.

Palabras claves: 5G, Open5GS, SDR (Radio Definida por Software), srsRAN, BladeRF, USRP.

Abstract

This degree project focuses on the design and implementation of a prototype 5G NR architecture through the use of SDR (Software Defined Radio) device and open source software. The main objective is to carry out a detailed study and perform tests in a realistic environment to assess the viability and operability of a private 5G SA network under this configuration, this covers voice over IP and data services. In addition, it seeks to create a fully functional environment for educational purposes, which allows analyzing this scenario as an authentic situation, it avoids the need for simulations in any part of the architecture.

To achieve this purpose, we realized a thorough analysis of the characteristics and underlying architecture of 5G networks. Likewise, we realized an exhaustive comparison between various hardware and software options that allow the implementation of a private 5G SA network. Finally, It was choosed to deploy the prototype using two recognized open source software platforms: srsRAN and Open5GS, for the implementation of RAN (Radio Access Network) and 5GC (5G Core) respectively. Physical radios were used in the execution of wireless transmissions. Specifically, the USRP B210 and Blade RF 2.0 SDR devices were used to benchmark their performance.

Following the implementation of the 5G network, extensive testing was carried out in a laboratory environment. These evaluations included measuring voice quality of service using the Mean Opinion Score (MOS), as well as downlink and uplink performance using a conventional user device connected to the private network. In addition, power tests were performed to evaluate behavior under various conditions.

In summary, this project aims to present an approach towards private networks based on SDR and open source software. This solution is presented as an economical, flexible and scalable option. Through the implementation and evaluation of a prototype in a realistic environment, the aim is to demonstrate the viability of these technologies in the field of 5G networks and their potential for practical and educational applications.

Keywords: 5G, Open5GS, SDR (Software Defined Radio), srsRAN, BladeRF, USRP.

3. Introducción

La telefonía móvil se ha convertido en un elemento esencial en la vida diaria de millones de personas desde la invención del teléfono móvil en 1973. A lo largo de las generaciones, las redes móviles han experimentado una rápida evolución, desde la 1G que permitía solo llamadas de voz analógicas, hasta la 4G que mejoró la velocidad y capacidad de las redes (Balapuwaduge and Li, 2018). Sin embargo, este progreso tecnológico y la creciente demanda plantean desafíos y oportunidades para las redes móviles.

En este contexto, surge la quinta generación (5G) de redes móviles, que promete revolucionar las telecomunicaciones al ofrecer una experiencia móvil sin precedentes. El 5G destaca por velocidades hasta 100 veces superiores a las de la 4G, baja latencia, alta confiabilidad y una densidad de conexiones notablemente alta. Además, posibilita la interconexión de una amplia gama de dispositivos, desde smartphones hasta objetos inteligentes y vehículos autónomos.

Uno de los desafíos en la implementación de nuevas tecnologías móviles es el costo de instalación, debido a la necesidad de actualizaciones de hardware y software que sean compatibles con las soluciones existentes. Este problema se agrava en áreas remotas con poca población, donde la inversión en infraestructura puede no ser rentable para los operadores debido a la baja densidad de población. Las redes tradicionales generalmente son propietarias, lo que dificulta la integración de soluciones de diferentes marcas. Esto limita la capacidad de los operadores móviles y proveedores de servicios para expandir sus redes y ofrecer servicios avanzados en estas áreas. Sin embargo, la alianza internacional O-RAN (Open RAN) surge como una solución a este problema. Centrada en la arquitectura de red, O-RAN tiene como objetivo permitir la interoperabilidad entre diferentes dispositivos de hardware y software. Esto ofrece escalabilidad en la implementación de la red 5G y futuras generaciones de comunicaciones móviles (Polese et al., 2023).

En paralelo a los desafíos de la implementación de nuevas tecnologías móviles, la evolución hacia componentes multipropósito, como servidores y dispositivos específicos, está reemplazando a los circuitos analógicos estáticos. Las radios definidas por software (SDR) son un ejemplo de esta evolución, ofreciendo versatilidad y adaptabilidad a diversas aplicaciones y requisitos de comunicación. Según Ibanez et al. (2022), “una de las tecnologías que se han beneficiado del SDR son las comunicaciones móviles, siendo las tecnologías LTE y 5G NR las que están actualmente disponibles, bajo soluciones Open Source en SDR para pruebas de laboratorio”.

Además, Sadiku & Akujuobi (2004) destacan que “el objetivo principal de la SDR es sustituir el mayor número posible de componentes analógicos y dispositivos VLSI digitales cableados del transceptor por dispositivos programables”. Esto aporta una flexibilidad y adaptabilidad impresionantes a las redes móviles modernas.

Por otro lado, se observa un problema en la escasez de prototipos o proyectos que estén relacionados con las redes móviles y que utilicen soluciones como las SDR. Esta situación representa un obstáculo para los estudiantes de Telecomunicaciones, ya que dificulta la oportunidad de ver en la práctica los conceptos teóricos aprendidos en las aulas y de validarlos a través de resultados. Esta falta de recursos prácticos pone de manifiesto la necesidad de desarrollar más proyectos que integren tecnologías como SDR en el campo de las redes móviles.

En este contexto, el objetivo central de este trabajo es diseñar y desarrollar un prototipo de red 5G NR utilizando radios definidas por software como plataforma experimental. Para lograr este objetivo, se utilizarán las SDR para realizar las funciones convencionales de una radio, como la modulación, demodulación, filtrado y codificación, mediante software en lugar de hardware específico. Además, se aprovecharán diversas soluciones de software Open Source y hardware SDR para implementar tanto la arquitectura de la red como las funciones de red.

Esta iniciativa no solo aprovecha el estado actual de los avances tecnológicos para validar la posibilidad de futuras implementaciones de redes 5G portátiles, flexibles y de bajo costo, sino que también establece un sólido fundamento científico y experimental. Este enfoque resulta beneficioso para estudiantes e investigadores en el campo de las telecomunicaciones, ya que proporciona una base práctica para la comprensión y aplicación de los conceptos teóricos.

Siguiendo este marco de referencia, el objetivo general se centra en diseñar e implementar un prototipo de arquitectura 5G NR para servicios de voz y datos basado en SDR (Software Defined Radio) y software Open Source. El prototipo comprenderá un sistema transceptor basado en dispositivos SDR comerciales y software Open Source srsRAN y Open5GS.

En consonancia con la investigación previa, los objetivos específicos de este proyecto son:

- Realizar un estudio de los diferentes equipos SDR (Software Defined Radio) y software Open Source existentes que implementen soluciones para el despliegue de una red 5G NR.
- Utilizar softwares Open Source para configurar los elementos clave que intervienen en una arquitectura 5G NR, para garantizar un correcto despliegue y funcionamiento en un entorno de laboratorio.

- Realizar pruebas de los servicios de voz y datos utilizando equipos de usuario (UE) comerciales conectados a la red 5G NR, para evaluar el desempeño de la red a través de diferentes métodos.

4. Marco teórico

4.1 5G NR (New Radio)

La tecnología 5G marca un hito importante en la evolución de las redes móviles, ampliando su alcance para crear una red integrada en diversos contextos de uso. (Dahlman et al., 2020). En específico, la tecnología de acceso de radio (RAT) conocida como 5G NR (New Radio) ha sido desarrollada por 3GPP para las redes móviles 5G, buscando establecer un estándar global para la interfaz de radio y mejorar la flexibilidad, escalabilidad y eficiencia en términos de consumo energético y aprovechamiento del espectro (Galal and O'Halloran, 2020).

La 5G-NR tiene como objetivo respaldar diversas situaciones y escenarios de uso, desde comunicación máquina a máquina a gran escala (mMTC) hasta transmisión ultrafiable de baja latencia (URLLC) y mejora de la comunicación móvil (eMBB) (Lien et al., 2017). La NR se basa en una arquitectura flexible y escalable, permitiendo la adaptación de parámetros y funciones en las capas física (PHY) y de control de acceso al medio (MAC) según las necesidades del canal. Se espera que esto se logre mediante un mayor rendimiento energético de la red y la explotación de espectro en frecuencias elevadas, utilizando componentes tecnológicos como numerología adaptable, estructura de trama de baja latencia, tecnología MIMO, interoperabilidad entre bandas de alta y baja frecuencia y transmisiones ultraligeras (Parkvall et al., 2017).

4.1.1 Estandarización

La Unión Internacional de Telecomunicaciones (UIT) definió el 5G en 2015 a través de los requisitos IMT-2020 (ITU, 2022). Para cumplir con estos criterios, el proyecto Third Generation Partnership Project (3GPP) inició en 2016 la especificación técnica del 5G NR (New Radio) y la arquitectura de red 5G NG (Next Generation) (3GPP, 2019).

La tecnología 5G se especificó por primera vez en la Release 15 del 3GPP, que se estructuró en tres fases. La primera fase, aprobada en diciembre de 2017, definió una versión no autónoma (NSA) de 5G NR, que se apoya tanto en interfaces aire LTE como NR y reutiliza la red central LTE. La segunda fase, especificada en junio de 2018, definió una versión autónoma (SA) de 5G NR, que incluye además una red central compatible con 5G con capacidades completas de plano de usuario y de control. La tercera fase, completada a finales de 2018, permitió que el núcleo 5G interactuara por igual con la red de acceso (RAN) LTE y NR (Fuentes et al., 2020).

La Release 16 comenzó a principios de 2018 y finalizó a finales de 2019, con el propósito de introducir mejoras y nuevas funcionalidades para una variedad de casos de uso y

escenarios de implementación de 5G (Ivezic, 2020). En cambio, la Release 17, publicada por el 3GPP en diciembre de 2020 y prevista para completarse en el primer trimestre de 2022, introduce nuevas prestaciones para respaldar casos de uso y despliegues diversos, incluyendo equipos de capacidad reducida, redes no terrestres, bandas de frecuencia más allá de 52 GHz y servicios de multicast y broadcast (Rahman et al., 2022).

La Release 18 del 3GPP marca el inicio de 5G Advanced, una evolución significativa del sistema 5G. Esta versión incorporará una mayor inteligencia en las redes inalámbricas a través de técnicas basadas en inteligencia artificial y aprendizaje automático a distintos niveles de la red. Se prevé que la Release 18 se complete a finales de 2023 (Rahman et al., 2022).

En resumen, la evolución del 5G ha sido guiada por una serie de releases progresivas del 3GPP, cada una añadiendo mejoras y nuevas capacidades para satisfacer una amplia gama de necesidades de comunicación y despliegue. Estas releases han permitido el desarrollo de una infraestructura flexible y adaptable que puede soportar los diversos casos de uso y escenarios de implementación que se esperan en la era del 5G.

4.1.2 Características 5G NR

Con el propósito de cumplir los objetivos específicos de cada caso de uso, 5G NR ofrece soluciones de comunicación adaptadas a una variedad de escenarios y necesidades. Por ejemplo, posibilita transmisiones de video en alta definición para el caso de mejora de la banda ancha móvil (eMBB), comunicaciones de baja latencia para vehículos de control remoto en las comunicaciones ultrarreliable de baja latencia (URLLC), y comunicaciones de tipo máquina con bajo consumo de energía y velocidad de datos en las comunicaciones masivas de tipo máquina (mMTC). Estas capacidades se basan en algunos de los elementos cruciales mencionados previamente y que serán detallados a continuación.

- **Network slicing:** El network slicing es una técnica que permite crear redes virtuales personalizadas sobre una misma red física 5G. Esta técnica facilita el manejo de los distintos casos de uso que tienen diferentes requisitos de rendimiento, como la velocidad, la capacidad o la latencia (ETSI, 2018). Así, cada dispositivo puede tener un segmento de la red optimizado para su aplicación específica. El network slicing es una arquitectura clave para lograr la escalabilidad y la flexibilidad de la nueva radio 5G.
- **Bandas milimétricas:** Las bandas milimétricas, también conocidas como Extra-High Frequency (EHF), son fundamentales para el desarrollo del 5G, ya que permiten la transmisión de datos a alta velocidad y baja latencia. Incluyendo frecuencias por debajo y por encima de los 30 GHz, como las bandas de 28 GHz y 26 GHz, estas bandas

proporcionan un amplio ancho de banda necesario para aplicaciones como eMBB. Además, su longitud de onda corta facilita la implementación de MIMO masivo con numerosas antenas sin ocupar demasiado espacio en torres o equipos de usuario (Nordrum & Clark, 2017a; Fonte et al., 2018; Hong et al., 2021).

- **MIMO:** El MIMO masivo, esencial en el contexto del 5G, permite la transmisión simultánea de múltiples flujos de datos a través de numerosas antenas, siendo complementado por el Multi-User-MIMO (MU-MIMO). Este último asigna recursos espaciales según la ubicación y canal de cada usuario, logrando una alta capacidad y velocidad en el 5G. El MU-MIMO también favorece el uso de arreglos de antenas adaptables y amplios (Dreifuerst et al., 2023).
- **Beamforming:** El Beamforming, una avanzada estrategia de procesamiento de señales, revoluciona la conectividad al dirigir precisamente el haz de la estación base hacia los dispositivos de usuario (UE), optimizando la emisión y recepción de señales. Integrada con el MIMO masivo, que aprovecha múltiples elementos de antena para lograr la simultánea transmisión y recepción de flujos de datos, ambas técnicas posibilitan enlaces más enfocados y eficaces entre estaciones base y usuarios. (Nordrum and Clark, 2017b).
- **Small Cells:** Las small cells, estaciones base celulares compactas, mejoran la conectividad en áreas específicas al instalarse en diversos sitios y conectarse a una macrocelda de mayor alcance, superando distancias y obstáculos (Kostopoulos et al., 2018). Estas celdas pequeñas optimizan el espectro mediante celdas compactas con reutilización de frecuencias, elevando la velocidad y capacidad de datos para usuarios, reduciendo interferencias y mejorando la calidad de señal al acercarse la fuente de transmisión (Vahid et al., 2015).

4.1.3 Escenarios de uso de IMT-2020

La tecnología 5G surge para abordar nuevos casos de uso solicitados por usuarios e industrias, y la UIT-R (Unión Internacional de Telecomunicaciones - Radiocomunicaciones) establece tres escenarios de aplicación integrados en la recomendación IMT Vision (ITU-R, 2015). Desarrollados en colaboración con la industria de comunicaciones móviles, organizaciones regionales y operadores, estos escenarios son parte del proceso IMT-2020 supervisado por el grupo WP5D de la UIT-R.

- **Banda ancha móvil mejorada (eMBB):** Sigue siendo un escenario de máxima relevancia debido a la creciente demanda y la aparición de nuevos casos de uso en el ámbito de la banda ancha móvil. La UIT-R ha establecido nuevos requisitos para mejorar la banda ancha móvil, que abarca diversos casos de uso con desafíos distintivos, como la necesidad de altas velocidades, capacidad para manejar una densidad elevada de usuarios en puntos de acceso, garantizar la movilidad fluida y proporcionar una experiencia ininterrumpida en áreas extensas (Dahlman et al., 2020). En términos generales, se reconoce que el escenario de eMBB se orienta hacia la comunicación centrada en el ser humano, enfocándose en satisfacer las cambiantes necesidades de los usuarios finales en un mundo cada vez más conectado.
- **Comunicación masiva de tipo máquina (mMTC):** El término "mMTC" se refiere a la conectividad inalámbrica diseñada para manejar decenas de miles de millones de dispositivos de tipo máquina, habilitando aplicaciones de la Internet de las Cosas (IoT) en entornos como ciudades inteligentes, agricultura avanzada e implementaciones de la industria 4.0. Según Bockelmann et al., (2016), el principal desafío de mMTC reside en establecer una conectividad escalable y eficiente para un gran número de dispositivos que emiten paquetes extremadamente cortos. Las soluciones para mMTC deben ofrecer una cobertura extensa, incluso con capacidad de penetración en espacios interiores, todo ello manteniendo costos moderados y minimizando el consumo energético.
- **Comunicación ultrarrápida y de baja latencia (URLLC):** Busca abarcar tanto la comunicación centrada en el ser humano como la orientada a máquinas, también conocida como Comunicación Crítica de Tipo Máquina (C-MTC). Este ámbito se distingue por involucrar casos de uso con requisitos rigurosos de latencia, confiabilidad y alta disponibilidad (Dahlman et al., 2020). Ejemplos de aplicaciones críticas incluyen el control remoto de robots y maquinaria, gestión de generación de energía, juegos en línea y telemedicina. Para habilitar estas aplicaciones, los investigadores, como She et al., (2023), destacan los considerables esfuerzos dedicados para alcanzar niveles de latencia y confiabilidad necesarios (aproximadamente 1 ms de latencia y hasta un 99,99999% de confiabilidad) en el contexto de 5G NR (New Radio), permitiendo respuestas instantáneas y continuas.

En la **Figura 1**, se presenta una representación visual de los tres escenarios de uso principales de 5G, acompañados por ejemplos ilustrativos de casos de uso. Esta categorización simplificada tiene como objetivo facilitar la definición de los requisitos para la especificación de la tecnología. Sin embargo, es crucial destacar que numerosos casos de uso no se ajustan

perfectamente a una sola categoría. Por ejemplo, pueden surgir servicios que demanden alta confiabilidad sin ser extremadamente sensibles a la latencia. De manera similar, es posible que haya casos de uso que requieran dispositivos de bajo costo sin que la duración de la batería sea una prioridad.

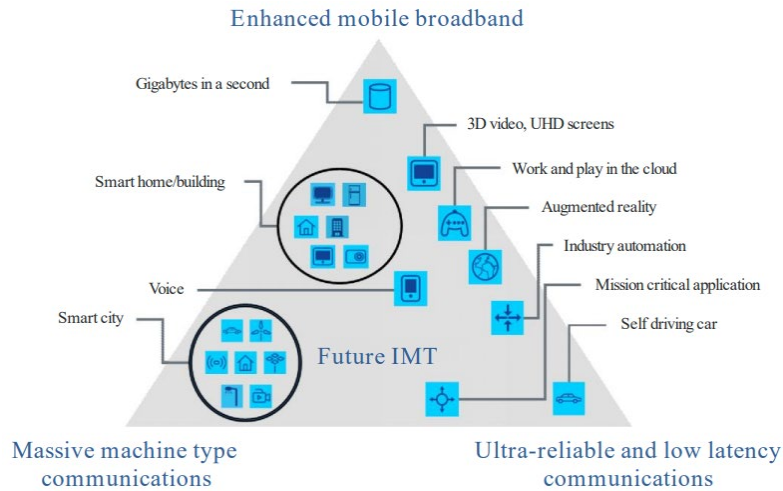


Figura 1. Escenarios de uso de IMT-2020
Fuente: Obtenido de (ITU-R, 2015)

4.2 Arquitectura 5G NR

La arquitectura del sistema 5G NR abarca elementos y funciones fundamentales para la tecnología 5G, desplegándose en dos modalidades: 5G NSA (Non-StandAlone) y 5G SA (StandAlone), diferenciándose por su nivel de interdependencia con el sistema 4G. En el primer caso, la red 5G se apoya parcialmente en la red 4G para ciertas funciones de control y gestión, mientras que en el segundo, la red 5G opera de manera independiente de la red 4G, contando con su propio núcleo de red.

El sistema 5G, conocido como 5GS (5G System), se compone de tres elementos clave según las definiciones del 3GPP (2018): el equipo de usuario (UE), la red de acceso radio (5G-RAN) y la red central (5GC). El equipo de usuario representa dispositivos conectados al sistema 5G, como teléfonos móviles o sensores, mientras que la red de acceso radio facilita la comunicación inalámbrica entre el equipo de usuario y la red central. A su vez, la red central administra funciones de control, gestión y enrutamiento del tráfico en el sistema 5G.

La **Figura 2** proporciona una representación gráfica de estos tres componentes esenciales que conforman el sistema 5G.

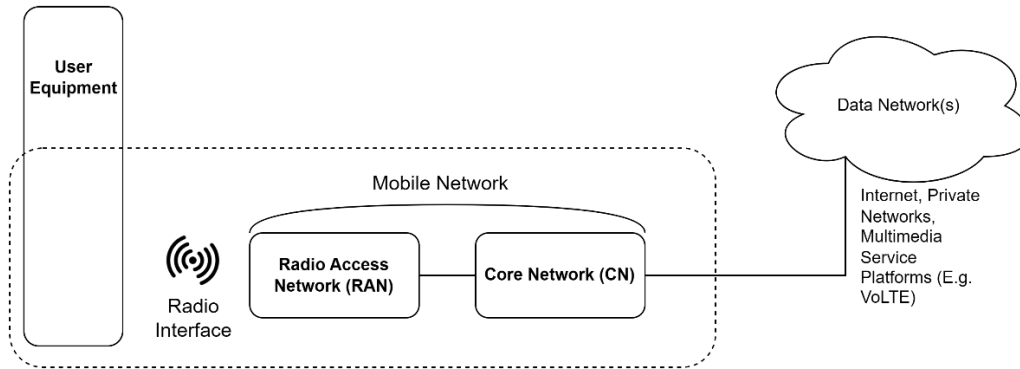


Figura 2. Bloques generales del sistema 5G
Fuente: Elaboración propia

4.2.1 5G NSA (Non-StandAlone)

La configuración 5G Non-Standalone (NSA) establece la conexión entre la Red de Acceso de Radio (RAN) 5G NR y la infraestructura 4G Evolved Packet Core (EPC), permitiendo a los operadores optimizar el rendimiento de la RAN y migrar gradualmente hacia el 5G sin la necesidad de un sistema completo. Esta arquitectura aprovecha la infraestructura existente en 4G, siendo común la combinación de 5G NR con EPC, aunque también es posible emplear la RAN de 4G LTE junto con la red 5G Core, especialmente en soluciones de código abierto (Coimbra, 2022).

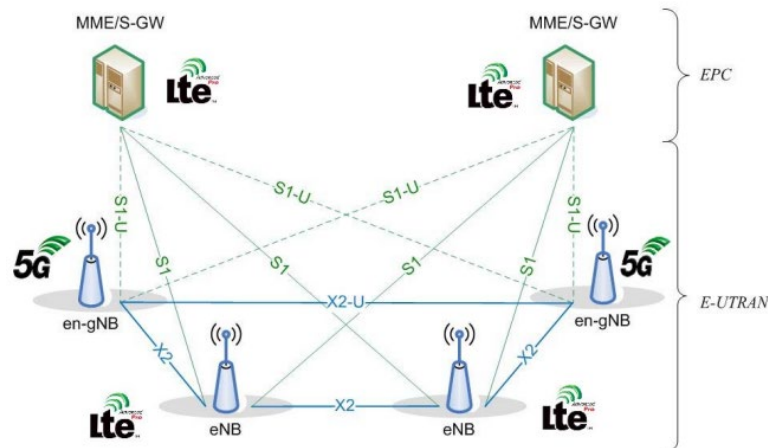


Figura 3. Arquitectura NSA
Fuente: Obtenido de (ETSI, 2019)

La **Figura 3** representa cómo la estación base NR de 5G se conecta con la estación base LTE de 4G a través de la interfaz X2, introducida en la release 15 para enlazar estaciones base LTE.

En el contexto de NSA, las estaciones base gNB gestionan el tráfico en el plano de usuario, compartiendo esa responsabilidad con las estaciones base eNB a través de la interfaz

X2, mientras que las estaciones base eNB controlan el tráfico de señalización entre el núcleo de red LTE y el usuario (Rischke et al., 2021).

4.2.2 5G SA (StandAlone)

La arquitectura 5G Standalone (SA) se fundamenta exclusivamente en componentes y funciones de red 5G, prescindiendo de la tecnología de radio LTE, la red 4G Core y generaciones anteriores. Se caracteriza por una red básica virtualizada basada en servicios 5G, empleando la computación en la nube para proporcionar diversos servicios a través de Internet. En este enfoque, el ecosistema 5G se fragmenta en componentes más pequeños e independientes (Coimbra, 2022).

La arquitectura SA representa la implementación completa de 5G, operando sin la intervención de una red 4G. La **Figura 4** ilustra cómo las estaciones base NR (nodo lógico "gNB") se conectan entre sí mediante la interfaz Xn, y la red de acceso (denominada "NG-RAN para la arquitectura SA") se vincula con la red 5GC a través de la interfaz NG.

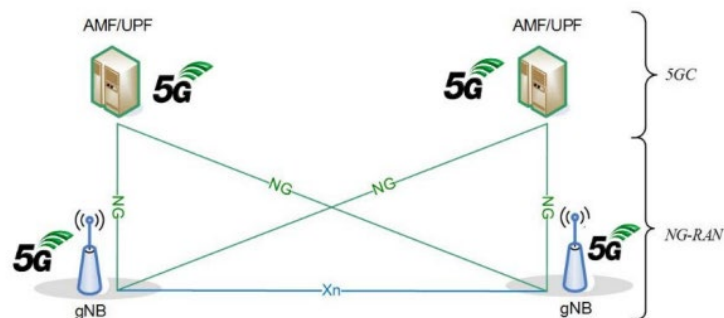


Figura 4. Arquitectura SA
Fuente: Obtenido de (ETSI, 2019)

En el escenario 5G SA, una única RAN compuesta por gNB se conecta al núcleo de red 5G. Estas estaciones base gestionan todo el tráfico de la red, tanto de datos como de señalización (Simmons, 2023).

4.2.3 5G Core Network

El núcleo de la red 5G se edifica sobre el EPC, introduciendo tres mejoras clave respecto a su predecesor: una arquitectura orientada a servicios, adaptabilidad a la fragmentación de la red y una clara separación entre el plano de control y el plano de usuario. La esencia del núcleo 5G se centra en una arquitectura basada en servicios, donde la especificación se enfoca en los servicios y funcionalidades proporcionados por la red central, más que en los nodos en sí. Esta perspectiva es coherente, dado que la red central tiende a estar altamente virtualizada en la actualidad, con la funcionalidad del núcleo ejecutándose en hardware informático de naturaleza

genérica (Dahlman et al., 2020). La **Figura 5** ofrece una representación de nivel superior del núcleo 5G, fundamentada en la idea de servicios y funcionalidades como punto focal central.

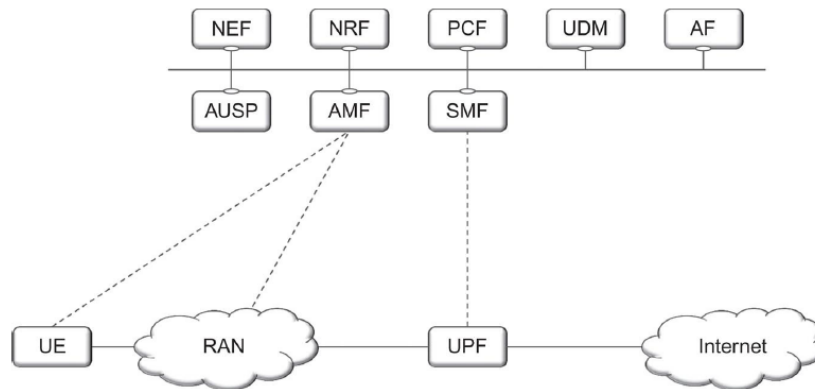


Figura 5. Arquitectura de red central de alto nivel (descripción basada en servicios)

Fuente: Obtenido de (Dahlman et al., 2020)

La función del plano de usuario abarca:

- **UPF (User Plane Function):** Es una gateway entre la RAN y redes externas como Internet. Sus responsabilidades incluyen el enrutamiento y reenvío de paquetes, la inspección de paquetes, la gestión de la calidad de servicio y el filtrado de paquetes, así como las mediciones de tráfico. También sirve de punto de anclaje para la movilidad (inter-RAT) cuando es necesario.

Las funciones del plano de control constan de varias partes:

- **SMF (Session Management Function):** se encarga, entre otras cosas, de la asignación de direcciones IP al dispositivo (también conocido como equipo de usuario, UE), el control de la aplicación de políticas y las funciones generales de gestión de sesiones.
- **AMF (Access and Mobility Management Function):** se encarga de la señalización de control entre la red central y el dispositivo, la seguridad de los datos de usuario, la movilidad en estado de reposo y la autenticación.
- **PCF (Policy Control Function):** se trata de la función de red que aplica las políticas de tarificación que los usuarios tienen contratados en la red. Es usada por medio de otras funciones a través del plano de control. La información sobre las reglas de tarificación es almacenada en el UDM.
- **UDM (Unified Data Management):** se trata de una función que se encarga de almacenar los datos de las suscripciones de los usuarios, las políticas de tarificación, y datos sobre la exposición del resto de funciones de red a otras redes, entre otros. El UDM permite que otras funciones de red guarden y consulten los datos que poseen.

- **NEF (Network Exposure Function):** su función se basa en interconectar de manera segura el núcleo de la red con otras redes. Esta función de red recibe datos desde otras redes y los guarda en el UDR. Una vez guardada la información, el NEF puede reenviarla a otras redes o bien funciones de red del propio núcleo.
- **NRF (Network Repository Function):** Se trata de un repositorio utilizado para que el resto de las funciones de red puedan ser alcanzables por otras funciones de red. Por esta razón, es importante que cuando se cree una nueva instancia de una función de red, ésta se registre en este repositorio.
- **AUSF (Authentication Server Function):** Maneja la autenticación mutua de las peticiones de acceso a la red. Una vez hechas las comprobaciones necesarias, el AUSF informa al UDM del resultado de la autenticación. Estas funciones se manejaban anteriormente en el AuC y el HSS del núcleo LTE.
- **AF (Application Function):** Se trata de una función de red cuyo cometido es conectar los servicios de otras redes al núcleo. Esta función es usada para conectar redes IMS al núcleo 5G, de manera que se posibilite el voNR en la red.

Cabe señalar que las funciones básicas de la red pueden implementarse de muchas maneras. Por ejemplo, todas las funciones pueden implementarse en un único nodo físico, distribuirse en varios nodos o ejecutarse en una plataforma en la nube.

4.2.4 Radio-Access Network

La red de acceso radio (RAN) puede estar conformada por dos tipos de nodos que se conectan a la red central 5G:

- **gNB (gNodeB):** Este nodo brinda servicios a dispositivos NR utilizando los protocolos NR de plano de usuario y de plano de control;
- **ng-eNB (Next Generation evolved NodeB):** Este nodo proporciona servicios a dispositivos LTE (Long Term Evolution), utilizando los protocolos LTE de plano de usuario y de plano de control.

Cuando una RAN está compuesta por ng-eNB para el acceso de radio LTE y gNB para el acceso de radio NR, se la conoce como NG-RAN (ETSI, 2020).

El gNB (o ng-eNB) es responsable de todas las funciones relacionadas con la radio en una o varias celdas, por ejemplo, la gestión de recursos radioeléctricos, el control de admisión, el establecimiento de conexiones, el encaminamiento de datos del plano de usuario al UPF y de

información del plano de control al AMF, y la gestión de flujos QoS (Quality of Service) (Dahlman et al., 2020).

En la **Figura 6**, se ilustra la conexión entre los nodos gNB y ng-eNB mediante la interfaz Xn. Ambos nodos, gNB y ng-eNB, establecen conexiones con la infraestructura 5G Core Network (5GC) mediante las interfaces NG, conectándose al AMF a través de la interfaz NG-C y al UPF mediante la interfaz NG-U.

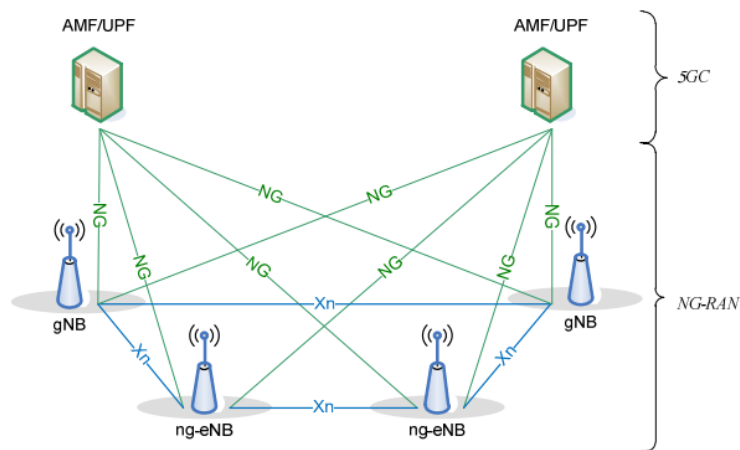


Figura 6. Arquitectura NG-RAN
Fuente: Obtenido de (ETSI, 2020)

La interfaz Xn, responsable de vincular los nodos gNB, cumple funciones clave como el soporte a la movilidad activa y conectividad dual. Además, posibilita funciones de gestión de recursos radioeléctricos (RRM) multicelda, permitiendo una movilidad fluida entre células vecinas mediante reenvío de paquetes (Dahlman et al., 2020).

4.3 Radio definida por software (SDR)

El concepto de "Radio Definida por Software" o SDR (Software Defined Radio) ha evolucionado con múltiples definiciones a lo largo del tiempo, llevando al Foro de Innovación Inalámbrica a colaborar con el grupo P1900.1 del IEEE para establecer una definición clara. Según este foro, la SDR se define como "Radio en las que algunas o todas las funciones de la capa física se definen por software" (Wireless Innovation Forum, 2017) , lo que implica la implementación de funciones de radio tradicionales mediante software en lugar de hardware.

El desarrollo de las SDR ha estado vinculado a tecnologías clave como los convertidores AD (Analógico-Digital) y DA (Digital-Analógico), procesadores de señales digitales (DSPs), procesadores de propósito general (GPPs) y matrices de puertas programables (FPGAs). Este progreso fue impulsado por la necesidad de soluciones de comunicación inalámbrica más

flexibles, inicialmente en el sector militar, respaldado por inversiones gubernamentales de Estados Unidos y Europa (Ulversoy, 2010).

Hoy en día, las SDR generan gran interés tanto en la comunidad científica como en operadores móviles, quienes adoptan esta tecnología emergente para avanzar en las comunicaciones inalámbricas. Dos conceptos destacan en este contexto: la flexibilidad y la versatilidad del software, que se traducen en una eficiente gestión de recursos y en un enfoque adaptable para el desarrollo continuo de las comunicaciones inalámbricas.

4.3.1 Esquema genérico de la SDR

Durante los últimos años, la SDR ha experimentado un rápido avance, tanto en su concepto como en sus arquitecturas. Sin embargo, su desarrollo se ha basado en un esquema fundamental que consta de tres bloques funcionales: la sección de RF (Radiofrecuencia), la sección de IF (Frecuencia Intermedia) y la sección de banda base (Wipro Technologies, 2002). La **Figura 7** muestra el esquema genérico de la SDR.

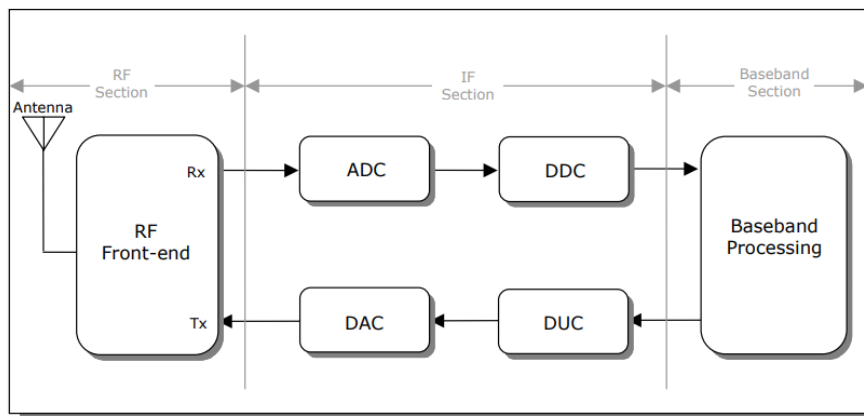


Figura 7. Arquitectura genérica de la SDR
Fuente: Obtenido de (Wipro Technologies, 2002)

Cada uno de estos bloques tiene una función específica dentro del sistema SDR. A continuación se detallan los elementos que intervienen en cada una de las secciones previamente mencionadas (Wipro Technologies, 2002):

- **La sección RF (RF front-end):** es la encargada de transmitir o recibir la señal de radio frecuencia (RF) a través de la antena, acoplándola y realizando una conversión descendente a una señal IF (en recepción). En transmisión se realiza una conversión ascendente para convertir la señal IF a RF, seguida de una etapa de amplificación.
- **La sección IF:** se encarga de pasar la señal IF a banda base y llevar a cabo la digitalización en recepción o la conversión digital-analógica en transmisión. Incluye bloques como ADC (Analog-to-Digital Converter) y DAC (Digital-to-Analog

Converter) para la conversión analógico-digital y digital-analógico, respectivamente. Además, incorpora bloques como DDC (Digital Down Converter) para la conversión digital descendente y DUC (Digital Up Converter) para la conversión digital ascendente, encargados de funciones de modulación y demodulación.

- **Sección de banda base:** realiza diversas operaciones como configuración de la conexión, ecualización, salto de frecuencia (frequency hopping), recuperación de sincronismo (timing recovery), entre otras, completando así el proceso integral de la SDR.

4.3.2 Arquitectura de la SDR

Para comprender el funcionamiento de los SDR, examinamos la arquitectura general de estos sistemas y sus requisitos de procesamiento. La arquitectura se compone de dos partes esenciales: un subsistema analógico y un subsistema digital, como muestra la **Figura 8**. El subsistema analógico abarca la alimentación de la antena, filtros RF, amplificadores y generadores de frecuencias de referencia, adaptando la señal de radiofrecuencia al rango de operación del subsistema digital.

En contraste, el subsistema digital se adapta al hardware y contiene los componentes vinculados a los equipos terminales y los equipos de transmisión e interconexión, como sistemas operativos e interfaces de programación de aplicaciones, entre otros (Burns, 2003). Su función central es realizar el procesamiento digital de la señal mediante software, implementando funciones como modulación, demodulación, filtrado y codificación.

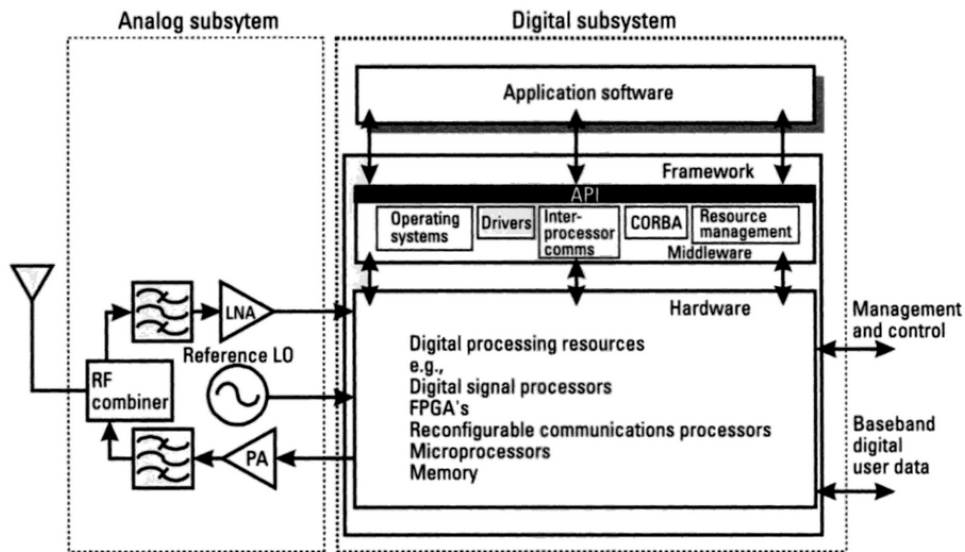


Figura 8. Arquitectura ideal de SDR
Fuente: Obtenido de (Burns, 2003)

4.3.3 Dispositivos SDR

Además de los programas de código abierto que se mencionan en la sección 4.4, que mayormente son aplicables en hardware elemental, se recurre a menudo a equipos SDR para desarrollar funciones vinculadas con la capa física. Estas plataformas brindan a los investigadores la capacidad de implementar y probar redes de extremo a extremo, incluso en ausencia del hardware convencional utilizado por los proveedores de servicios. A continuación, se describen algunas opciones de equipos SDR empleados con este propósito:

- **USRP B210:** Este dispositivo de radio por software opera con dos canales, abarcando un rango de frecuencias desde 70 MHz hasta 6 GHz, y un ancho de banda de aproximadamente 56 MHz. Es compatible con USB 3.0, GNU Radio y OpenBTS, además de soportar MIMO 2X2. Su precio ronda los 2.310 USD (Ettus Research, 2023a).
- **USRP X310:** Otro dispositivo de radio por software con dos canales, capaz de cubrir desde frecuencias continuas hasta 6 GHz, con un ancho de banda de 120 MHz. Dispone de interfaces PCIe, 10Gigabit Ethernet, Express Card y Gigabit Ethernet. Compatible con GNU Radio, C++, Python, Amarisoft LTE 100, OpenBTS, entre otros, y soporta MIMO 4X4. Tiene un precio aproximado de 10.290 USD (Ettus Research, 2023b).
- **USRP N310:** Este dispositivo de radio definida por software opera con cuatro canales de transmisión y cuatro de recepción, abarcando frecuencias desde 10 MHz hasta 6 GHz, con un ancho de banda de más de 100 MHz. Cuenta con interfaces Gigabit

Ethernet, 10Gigabit Ethernet y dos SFP+, soporta alta densidad de arrays MIMO y es compatible con GNU Radio. Su precio se estima en alrededor de 18.970 USD (Ettus Research, 2023c).

- **LimeSDR:** Un dispositivo de radio por software con uno o dos canales que abarcan desde 100 kHz hasta 3,8 GHz, y un ancho de banda de 61 MHz. Soporta desde 2G hasta 4G, tiene conectividad USB 3.0 y es compatible con GNU Radio, Pothos, SoapySDR y UHD. MIMO 2X2 también es posible. Su precio es aproximado de 3.533 USD (Lime Microsystems, 2020).
- **Blade RF:** Este dispositivo de radio definida por software opera con dos canales, cubriendo frecuencias desde 300 MHz hasta 3,8 GHz, con un ancho de banda de 28 MHz por cada canal. Ofrece una interfaz USB 3.0, soporta SISO y es compatible con GNU Radio. Su costo es de alrededor de 520 USD (Nuand, 2023a).
- **BladeRF 2.0:** La versión de segunda generación del Blade RF, también con dos canales, abarcando frecuencias desde 47 MHz hasta 6 GHz, con un ancho de banda de 56 MHz por canal. Posee una interfaz USB 3.0, soporta MIMO 2X2 y es compatible con GNU Radio. Su precio es de aproximadamente 860 USD (Nuand, 2023b).

La **Tabla 1** presenta un resumen de las capacidades de cada equipo SDR y los paquetes de software RAN compatibles.

Tabla 1. Capacidades de los SDR y su integración con el software RAN

	Rango de frecuencias	Ancho de banda (MHz)	Canales	MIMO	Software RAN
USRP B210	[70 MHz; 6 GHz]	56	2 TX/RX	2x2	OAI, srsRAN
USRP X310	[10 MHz; 6 GHz]	120	2 TX/RX	4x4	OAI, srsRAN
USRP N310	[10 MHz; 6 GHz]	100	4 TX-4 RX	Alta densidad	OAI, srsRAN
LimeSDR	[100 kHz; 3.8 GHz]	61.44	1 / 2 TX/RX	2X2	OAI, srsRAN
BadeRF	[300 MHz; 3.8 GHz]	28	2 TX/RX	SISO	OAI, srsRAN
BladeRF 2.0	[47 MHz; 6 GHz]	56	2 TX/RX	2X2	OAI, srsRAN

Fuente: Elaboración propia

En términos generales, hay varios tipos de SDR que se adaptan a distintas necesidades de implementación. Por ejemplo, las opciones de Universal Software Radio Peripheral (USRP), como el USRP B210 y el USRP N310, suelen desempeñar roles específicos, como estaciones base en tejados o pequeñas torres, respectivamente. En contraste, SDR como el LimeSDR y el bladeRF a menudo son utilizados en células pequeñas debido a su menor potencia en comparación con las soluciones USRP.

En el marco de este proyecto, se ha optado por la integración de los dispositivos SDR Ettus B210 y bladeRF 2.0. Estas elecciones se sustentan en su completa compatibilidad con las herramientas seleccionadas para la implementación de la red, así como en su disponibilidad. Es relevante destacar que el dispositivo bladeRF 2.0 se encuentra accesible en el laboratorio de Antenas y Telecomunicaciones de la Facultad de Energía, Industrias y Recursos Naturales no Renovables de la Universidad Nacional de Loja, mientras que el SDR Ettus B210 fue cedido por la Universidad Técnica Particular de Loja. Esta elección de equipos nos proporciona la oportunidad de realizar una comparativa de sus rendimientos en el despliegue de la red 5G.

4.4 Implementaciones Open Source para redes 5G

Las redes 5G representan la vanguardia de las comunicaciones móviles, ofreciendo velocidades de transmisión elevadas, baja latencia y una notable confiabilidad, sustentadas en una arquitectura flexible y escalable (Rinaldi et al., 2021). Para lograr esta versatilidad, la implementación de SDR (Software Defined Radio) y el uso de software de código abierto son cruciales, permitiendo la ejecución de funciones de red a través de software en lugar de depender de hardware especializado. Esta metodología no solo reduce costos y acelera el desarrollo e innovación de las redes 5G, sino que también simplifica la interoperabilidad y adaptación a diferentes contextos.

Dentro de esta sección, se presentan algunas de las implementaciones de código abierto más destacadas en las redes 5G, abarcando tanto el núcleo de red (5GC) como el acceso de radio (RAN).

4.4.1 Paquetes de software 5G RAN Open Source

Además del núcleo de red (5GC), otro componente esencial en la implementación de redes 5G es la red de acceso de radio (RAN). Para acelerar el desarrollo e innovación de la RAN, han surgido diversos paquetes de software de código abierto específicamente diseñados para esta función. Estos paquetes proporcionan soluciones versátiles que pueden desplegarse tanto en plataformas de computación en la nube como en el borde de la red. Alineados con el estándar O-RAN, estos conjuntos de software para RAN establecen una arquitectura abierta e interoperable, facilitando la integración fluida con diversos proveedores y componentes. Según O-RAN (2023), el objetivo principal de esta iniciativa es "lograr una solución que unifique y acelere la evolución y el despliegue en la RAN". En esta sección, se destacan algunos de los paquetes de software para RAN de código abierto más relevantes en el contexto de las redes 5G.

- **OAI:** implementa una red de acceso radioeléctrico 4G LTE y 5G, que incluye UE y gNB. Proporciona una de las mejores redes de acceso radioeléctrico mediante software RAN de código abierto. Debido a las numerosas restricciones asociadas a la radio definida por software (SDR) y a las evaluaciones en tiempo real, OAI también ofrece un emulador de RAN 4G y 5G (OpenAirInterface, 2023).
- **UERANSIM:** es un proyecto C++ de código abierto que emula un UE y un gNB. Debido a su simplicidad y facilidad de uso, se utiliza a menudo para probar paquetes de software que sólo proporcionan la red Core 5G, ya que permite emular la RAN de forma transparente a la red Core (Güngör, 2022).
- **Free5GRAN:** es una pila 5G RAN de código abierto en C/C++. Aunque se encuentra en desarrollo activo, presenta un enfoque centrado en una integración perfecta con SDR. Debido a que se encuentra en una fase temprana de desarrollo, no es tan popular como los demás (Free5GRAN, 2021).
- **srsRAN:** se ha centrado en desarrollos teniendo en cuenta 4G. Actualmente están trabajando en la pila completa 5G SA para gNB. Este proyecto también se conoce como srsLTE por su biblioteca de componentes SDR gratuitos y de código abierto para 4G LTE; srsRAN se considera la evolución hacia 5G NR (srsRAN, 2023a).

La comparación entre los cinco proyectos se presenta en la **Tabla 2**.

Tabla 2. Comparación de paquetes de software RAN 5G

	OAI-RAN	UERANSIM	Free5GRAN	srsRAN	OAI-Emulator
Lenguaje de programación	C, C++	Varios	Go	C	C, C++
Código abierto	Sí	Sí	Sí	Sí	Sí
Compatibilidad SDR	Sí	No	Sí	Sí	No
Soporte de plataformas	Linux	Linux	Linux	Linux	Linux
Documentación	Muy buena	Buena	Buena	Muy buena	Muy buena
Aplicación(es)	4G, 5G	5G	5G	4G, 5G	4G, 5G

Fuente: Elaboración propia

Los estudios realizados por Geng et al. (2017) sobre "Análisis de rendimiento y comparación de sistemas SDR basados en GPP" evaluaron la implementación de una red 4G-LTE con OpenAirInterface (OAI) y srsLTE (actualmente conocido como srsRAN). Concluyeron que srsLTE presenta un rendimiento superior en términos de estabilidad en la PHY DL. Además, los módulos PHY de srsLTE superan en velocidad a los de OAI en el tiempo de ejecución de la CPU, y srsLTE demuestra un consumo de memoria más eficiente que OAI.

En un estudio adicional realizado por Nieto (2022) sobre "Arquitecturas LTE-5G mediante SDR y OpenAirInterface", se determinó que para lograr un rendimiento óptimo con OAI es necesario un ajuste significativo de los parámetros de control, en comparación con la alternativa de srsRAN. Este último ofrece un rendimiento y estabilidad superiores sin requerir ajustes costosos.

Basándonos en la comparativa de diferentes softwares en este contexto, se ha decidido implementar una red 5G utilizando srsRAN para el Radio Access Network (RAN), considerando sus destacadas características en cuanto a requisitos computacionales, tiempo de ejecución y compatibilidad con dispositivos como bladeRF 2.0 y USRP B210.

4.4.2 Paquetes de software 5GC Open Source

La implementación efectiva de una red 5G requiere un núcleo de red (5GC) capaz de respaldar las funciones y servicios establecidos por el estándar 3GPP. Sin embargo, el desarrollo e implementación de un 5GC pueden ser costosos y complicados. Como alternativa, el uso de conjuntos de software de código abierto se presenta como una solución adaptable y flexible a las necesidades de operadores e investigadores (Dzogovic et al., 2018). En esta sección, se destacan proyectos de Core Open Source relevantes para configurar y desplegar un 5GC mediante software, ya sea en la nube o en el borde de la red.

- **Open Air Interface (OAI):** es un proyecto de código abierto, desarrollado actualmente por OpenAirInterface Software Alliance (OSA), capaz de implementar una red 5G plenamente funcional. Aunque es bastante reciente en lo que se refiere a la red de acceso, proporciona todos los componentes necesarios para el desarrollo de la red core (OpenAirInterface, 2023).
- **Open5GCore:** es una implementación práctica de la red 3GPP 5G Core desarrollada por el Instituto Fraunhofer, capaz de implantar una red 5G autónoma. Aunque permite una innovación 5G rápida y específica y un banco de pruebas fiable, el software está sujeto a licencia (Open5GCore, 2023).
- **Free5GC:** es un proyecto de código abierto para redes 5G Core. El objetivo de este proyecto es implementar la red 5G Core. Sin embargo, no tiene una red de acceso de radio, a diferencia de OAI, confiando en UERANSIM - un simulador UE/RAN para fines de prueba (Free5GC, 2023).
- **Open5GS:** es una implementación de código abierto en lenguaje C de las redes 5G Core y 4G Core. Similar a Free5GC, utiliza el simulador UERANSIM para la RAN. También

tiene una aplicación WebUI, similar a WebConsole, implementada en Node.JS y React (Open5GS, 2023a).

- **Magma:** es una plataforma de código abierto desarrollada por Facebook y respaldada por muchas otras entidades, como la Fundación Linux y la OAI. Su principal objetivo es garantizar funciones 5G abiertas y nativas de la nube, a fin de garantizar la interoperabilidad entre proveedores de software, hardware y RAN en todo el ecosistema (Magma, 2023).

Tabla 3. Comparación de paquetes de software 5G Core

	OAI	Open5GCore	Free5GC	Open5GS	Magma
Lenguaje de programación	C, C++	Varios	Go	C	Varios
Código abierto	Sí	No	Sí	Sí	Sí
Interfaz	CLI	CLI, WebUI	CLI, WebUI	CLI, WebUI	CLI
Soporte de plataformas	Linux	Linux	Linux	Linux, MacOS	Linux, MacOS
Documentación	Muy buena	Muy buena	Buena	Muy buena	Buena
Aplicación(es)	4G, 5G	5G	5G	4G, 5G	4G, 5G

Fuente: Elaboración propia

En el presente proyecto de titulación, se emplea el software srsRAN, que proporciona funcionalidades de UE, eNodeB/gNodeB y núcleo de la red 4G LTE. Sin embargo, para incorporar la funcionalidad del 5GC, es necesario utilizar un software de terceros compatible con srsRAN.

Estas soluciones son comúnmente utilizadas en entornos de prueba de redes 5G para verificar las funcionalidades y características de sus componentes principales. En la **Tabla 3** se proporciona una comparación entre los cinco paquetes de software 5G para el núcleo de red.

5. Metodología

5.1 Fases del proyecto

Este Trabajo de Integración Curricular se basó en métodos y procedimientos precisos que permitieron llevar a cabo la investigación y cumplir con los objetivos específicos del proyecto. Para ello, se utilizó una metodología estructurada que aseguró un desarrollo eficiente y ordenado. Las etapas de desarrollo fueron ajustadas de acuerdo al diagrama de flujo representado en la **Figura 9**, lo que aseguró el cumplimiento de los objetivos del proyecto. A continuación, se describe en detalle el procedimiento de cada fase que condujo a la consecución de los objetivos iniciales.

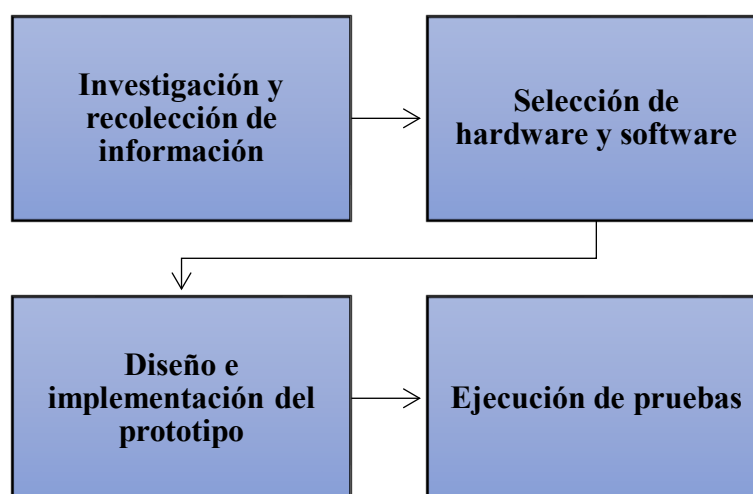


Figura 9. Fases del proyecto
Fuente: Elaboración propia

5.1.1 Fase 1: Investigación y recolección de información sobre 5G NR y SDR

En la primera fase, se llevó a cabo una exhaustiva investigación y recolección de información acerca de la tecnología 5G NR. Este proceso abarcó aspectos como su arquitectura, funciones de red, protocolos e interfaces. Simultáneamente, se procedió a realizar un análisis detallado de la tecnología SDR, comprendiendo su arquitectura y funcionamiento. Durante esta investigación, se exploraron las diversas opciones disponibles en el mercado, identificando tendencias relacionadas con equipos SDR y software de código abierto destinados a redes 5G NR.

Este enfoque integral permitió adquirir una comprensión completa de las opciones de hardware y software disponibles. Se logró identificar tanto desafíos como oportunidades inherentes a la arquitectura 5G NR, así como el empleo de SDR y software de código abierto para abordar estas necesidades específicas.

5.1.2 Fase 2: Selección de hardware y software a emplear

En la segunda fase de la investigación, se amplió el alcance de la indagación anterior, que se centró en explorar las diversas opciones disponibles en el mercado. Posteriormente, se llevó a cabo una comparación exhaustiva de las soluciones identificadas, constituyendo un paso crucial previo a la selección del hardware y software más idóneos para el desarrollo del proyecto.

En lo que respecta a la selección del hardware, se enfocó principalmente en la elección de los equipos SDR a utilizar. La sección [4.3.3](#) detalla las diferentes alternativas y presenta su comparación. La decisión de optar por los equipos Blade RF 2.0 y USRP B210 se basó en sus características, compatibilidad con el software seleccionado y disponibilidad. Además, se eligieron como puntos de comparación para evaluar el rendimiento en diversos escenarios.

En cuanto a la selección del software, el enfoque se orientó hacia la implementación de la RAN y 5GC de la red. La sección [4.4.1](#) y [4.4.2](#) proporciona una descripción detallada de las diversas alternativas y su comparación. La elección de estos softwares se fundamentó en sus características y compatibilidad mutua, respaldada por revisiones de investigaciones que compararon implementaciones en redes 4G y 5G basadas en dichos softwares. En resumen, se seleccionó el software srsRAN para la implementación de la RAN, mientras que para el 5GC se optó por Open5GS, debido a su compatibilidad mutua y destacándose frente a otras alternativas como OpenAirInterface en términos de rendimiento y eficiencia.

Adicionalmente, en la sección [5.2](#) se definen todos los requerimientos de hardware y software necesarios para el despliegue de la red 5G.

5.1.3 Fase 3: Diseño e implementación del prototipo

En esta fase del proyecto, se llevó a cabo la elaboración de un diseño detallado para la arquitectura de la red 5G NR, basado en los requerimientos previamente definidos para el hardware y software óptimos. Se estableció una arquitectura de red 5G SA que cumpliera con los diversos requisitos establecidos, identificando cada elemento participante en la red y delineando sus interfaces de comunicación correspondientes. Las especificaciones de diseño y el diseño propuesto se encuentran detallados en la sección [5.3](#).

Simultáneamente, en esta misma etapa del proyecto, se abordó la implementación práctica de la red 5G SA. Con este propósito, se llevó a cabo la instalación y configuración de la infraestructura necesaria de la red (RAN y 5GC), centrándose en una solución basada en SDR y software de código abierto. Se siguió meticulosamente las directrices y recomendaciones

proporcionadas por los proveedores de software, asegurando una instalación y configuración precisa de los elementos de la red.

Los detalles específicos de la implementación se describen en la sección [5.4](#), que abarca los siguientes puntos:

- Configuración de la SIM
- Instalación del controlador de hardware BladeRF
- Instalación del controlador de hardware USRP B210
- Configuración de la red
- Registro de suscriptores
- Configuración del UE
- Conexión a la red
- Mejora del rendimiento de srsENB

5.1.4 Fase 4: Ejecución de pruebas del prototipo

En la cuarta fase del proyecto, se centró en la realización de pruebas de los servicios de voz y datos en la red 5G SA, una vez instalada y configurada. Estas evaluaciones se llevaron a cabo utilizando un dispositivo de usuario (UE) comercial, lo que permitió analizar el rendimiento y la calidad de la red en diversas condiciones.

La metodología detallada de las pruebas se encuentra descrita en la sección [5.5](#). Estas pruebas abarcaron diversos aspectos, incluyendo pruebas de datos, pruebas de VoIP y pruebas de potencia: RX. Fueron fundamentales para determinar el grado de funcionalidad del prototipo de red. Además, los resultados obtenidos en esta fase se utilizaron para realizar ajustes y mejoras adicionales en la configuración de la red.

5.2 Requerimientos de hardware y software

En esta sección, se describirán los requerimientos de hardware y software considerados en el diseño e implementación del prototipo de la arquitectura 5G SA.

5.2.1 Hardware

En cuanto al hardware, a continuación se presenta la descripción detallada de la computadora utilizada para instalar y operar los diversos componentes de la red 5G. Además, se detallan los dispositivos SDR y las antenas empleadas para la transmisión y recepción de señales NR, así como las tarjetas SIM/USIM programadas con claves conocidas y el dispositivo de usuario utilizado.

Computador portátil

Para llevar a cabo el escenario real, se utilizó un portátil Lenovo ThinkPad L15 Gen 2 (ver **Figura 10**). Este dispositivo sirvió como plataforma para alojar tanto el software del 5GC como sus componentes y el gNB. Desde esta máquina, también se llevaron a cabo las operaciones de lectura y programación de las tarjetas SIM de los dispositivos de usuario. Es importante señalar que la ejecución de los programas diseñados para la RAN y el 5GC del prototipo de red 5G requería un ordenador con sistema operativo Linux como base, y en este proyecto se seleccionó la distribución Ubuntu 22.04, que ya ha sido probada con los softwares a utilizar.



Figura 10. Lenovo ThinkPad L15 Gen 2

Fuente: Obtenido de www.lenovo.com

Adicionalmente, fue esencial que el ordenador tuviera al menos un puerto USB 3.0, ya que los dispositivos SDR necesitaban esta versión para funcionar correctamente. También se establecieron requisitos mínimos de hardware, que incluían 8 GB de memoria RAM, 8 núcleos, una frecuencia de CPU de 2.8 GHz y un procesador Intel Core i7. Los atributos principales de este equipo se detallan en la **Tabla 4**.

Tabla 4. Características del computador portátil

Marca	Lenovo
Modelo	ThinkPad L15 Gen 2
Procesador	Intel® Core™ i7-1165G7 de 11ª generación
Velocidad del procesador	2,80 GHz hasta 4,70 GHz
Almacenamiento	500 GB SSD
Memoria	16 GB DDR4
Sistema operativo	Ubuntu 22.04 LTS
Número de procesadores	8

Fuente: Elaboración propia

SDRs

La implementación del gNB se llevó a cabo utilizando la tecnología SDR. Específicamente, se eligieron el dispositivo BladeRF 2.0 micro xA9 de Nuand (ver **Figura 11**) y el USRP B210 de Ettus (ver **Figura 12**) para realizar una comparación de rendimiento. Esta elección se basó en un análisis exhaustivo de diversos equipos SDR que ofrecían soluciones para el despliegue de una red 5G NR, considerando sus características y compatibilidad con los softwares seleccionados. A continuación, en la **Tabla 5** y **Tabla 6**, se detallan las principales características técnicas de estos dispositivos:

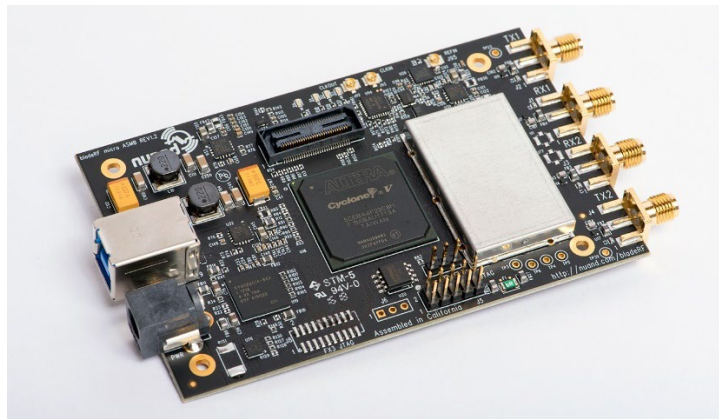


Figura 11. BladeRF 2.0 micro xA9
Fuente: Obtenido de www.nuand.com

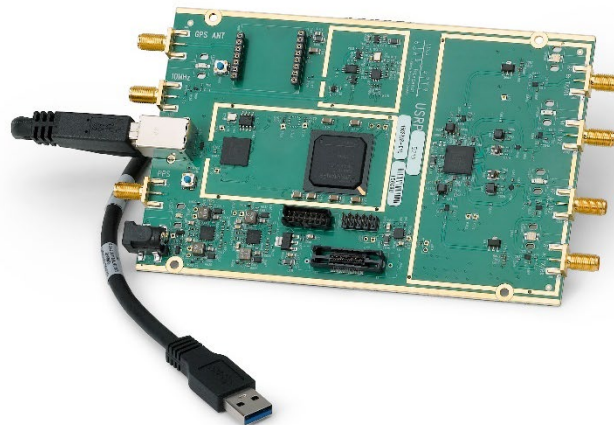


Figura 12. USRP B210
Fuente: Obtenido de www.ettus.com

Tabla 5. Características del SDR BladeRF 2.0 micro xA9

Rango de frecuencias	[47 MHz; 6 GHz]
Ancho de banda (MHz)	56
Canales	2 TX/RX
MIMO	2X2
Conexiones	USB 3.0
Frecuencia de muestreo	61,44 MHz
Driver	BladeRF
Software RAN	OAI, srsRAN

Fuente: Elaboración propia

Tabla 6. Características del USRP B210

Rango de frecuencias	[70 MHz; 6 GHz]
Ancho de banda (MHz)	56
Canales	2 TX/RX
MIMO	2X2
Conexiones	USB 3.0
Frecuencia de muestreo	61,44 MHz
Driver	UHD
Software RAN	OAI, srsRAN

Fuente: Elaboración propia

Antena Dipolo Banda Ancha

Las antenas utilizadas en conjunto con los SDR fueron del fabricante TECHTOO (ver **Figura 13**). Se trata de antenas de banda ancha de 5 dBi Omni Direccional con conector macho SMA, diseñadas específicamente para su compatibilidad con los dispositivos SDR. Las características principales de estas antenas se detallan en la **Tabla 7**.



Figura 13. TECHTOO Antena 3G 4G de banda ancha

Fuente: Obtenido de www.amazon.com

Tabla 7. Características de Antena Dipolo Banda Ancha

Marca	TECHTOO
Impedancia	50 Ohm
Alcance máximo	150 pies
Rango de frecuencia (MHz)	698-960/1710-2170/2500-2700
Ganancia (dBi)	5dBi
Diámetro de la antena	14mm

Fuente: Elaboración propia

Equipo de usuario comercial

Se empleó el Xiaomi Poco M3 Pro 5G (ver **Figura 14**) como dispositivo de usuario comercial para llevar a cabo las pruebas y análisis en la implementación de la red 5G. Las

características detalladas de este dispositivo se encuentran en la **Tabla 8**. Es relevante destacar que el software utilizado solo admitía un espaciado entre subportadoras de 15 kHz, lo que limitaba la selección de dispositivos de usuario a aquellos que pudieran operar en bandas compatibles con FDD (por ejemplo, banda 3), como es el caso de este dispositivo. Además, era necesario que el dispositivo admitiera el modo 5G SA, ya que esta arquitectura era la focal del Trabajo de Integración Curricular.



Figura 14. Xiaomi Poco M3 Pro 5G
Fuente: Obtenido de www.mi.com

Tabla 8. Características Xiaomi Poco M3 Pro 5G

	2G	GSM 850 / 900 / 1800 / 1900 (SIM 1 & SIM 2)
	3G	HSDPA 850 / 900 / 1700(AWS) / 1900 / 2100
	4G	LTE B1 / B2 / B3 / B4 / B5 / B7 / B8 / B12 / B17 / B20 / B28 / B32 / B38 / B40 / B41 / B66
	5G	SA/NSA B1 / B3 / B7 / B8 / B20 / B28 / B38 / B40 / B41 / B66 / B77 / B78
Red	GPRS	Si
	EDGE	Si
	SIM	nano-SIM dual
	Puerto infrarrojo	Si

Fuente: Elaboración propia

Tarjetas USIM

Para establecer la conexión y llevar a cabo la autenticación en una red 5G, se hizo necesario contar con un componente esencial: la tarjeta USIM/SIM. En este proyecto, se optó por utilizar tarjetas programables de la marca SYSMOCOM, en concreto, el modelo sysmoISIM-SJA2 (ver **Figura 15**), que representa la evolución del sysmoUSIM-SJS1 y cuenta con soporte para tecnología 5G.

Según Sysmocom (2020) las principales mejoras incluyen:

- Aplicación ISIM para IMS/VoLTE (además de aplicaciones SIM + USIM)

- applet ARA-M preinstalado para usar con Android Carrier Privileges
- Solo un producto que cubra el factor de forma 2FF, 3FF y 4FF
- Todos los archivos 5G especificados por 3GPP en la tarjeta

La nueva tarjeta, incluye todas las características principales de su predecesor, tales como:

- Modificación del contenido de cualquier archivo en la aplicación SIM, ISIM, USIM (usando el pin AMD)
- Modificación de material clave, algoritmo de autenticación seleccionado y parámetros relacionados (usando el pin AMD)
- Instalación de applets Java a través de GlobalPlatform SCP02 o TS 03.48 OTA
- Gestión remota de archivos a través de TS 03.48 OTA



Figura 15. sysmoISIM-SJA2
Fuente: Obtenido de [sysmocom Webshop](http://sysmocom.com)

Lector de tarjetas SIM/USIM

La programación de las tarjetas sysmoISIM-SJA2 con parámetros específicos se llevó a cabo utilizando un lector de tarjetas desarrollado por la compañía Omnikey. Se empleó el modelo CardMan 3121 USB CCID interface (ver **Figura 16**) en conjunto con el software pysim. Las características principales de este lector de tarjetas se detallan en la **Tabla 9**.

Tabla 9. Características de CardMan 3121 USB CCID interface

Tipos de tarjeta	Tarjetas inteligentes de 5 V, 3 V y 1.8 V, ISO 7816 Clase A, B y C
Estándares	ISO 7816 y EMV2 2000, nivel 1
Protocolos	T=0, T=1, 2 hilos: SLE 4432/42 (S=10), 3 hilos: SLE 4418/28 (S=9), I2C (S=8)
USB	Puerto USB 2.0 CCID (también conforme a USB 1.1)
Sistema operativo	Windows (32bits, 64bits) Linux Mac OS X Android

Fuente: Elaboración propia



Figura 16. CardMan 3121 USB CCID interface
Fuente: Obtenido de [sysmocom Webshop](https://www.sysmocom.com)

USRP GPS-Disciplined Oscillator Kit

Dadas las limitaciones de los relojes internos de los equipos SDR y las recomendaciones del software de usar un reloj de referencia externa de 10 MHz o un oscilador disciplinado por GPSDO, se optó por un reloj de referencia externo para el dispositivo SDR USRP B210. Esta medida se tomó para reducir los problemas de sincronización entre los dispositivos de usuario (UE) y la estación base 5G. Se utilizó específicamente el Kit GPSDO para USRP N200/N210, mostrado en la **Figura 17**. Este kit incluye un cable de alimentación, un cable serie, tres cables de RF y dos tornillos para la instalación en el chasis de la serie USRP N200.

Antena GPS activa de 3 voltios

Además del USRP GPS-Disciplined Oscillator Kit, también se necesitó una antena GPS activa de 3V (ver **Figura 18**) diseñada para su uso con el kit. Un GPSDO, cuando está sincronizado con la constelación GPS, puede proporcionar una sincronización horaria con una precisión de hasta 50 ns respecto al estándar global de GPS, lo que resulta en una alta precisión del reloj.

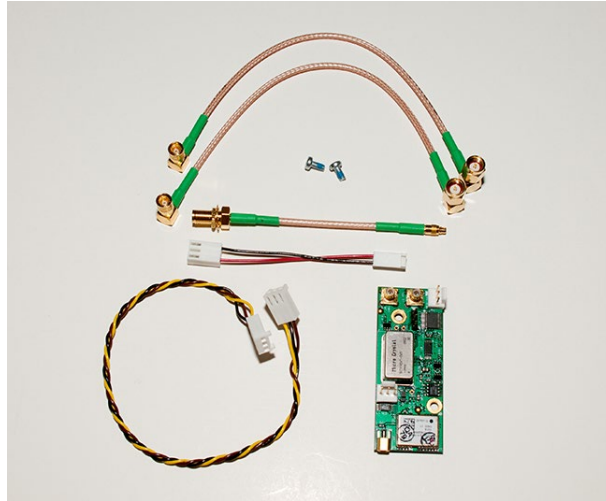


Figura 17. USRP GPS-Disciplined Oscillator Kit
Fuente: Obtenido de www.ettus.com



Figura 18. Antena GPS activa de 3 voltios
Fuente: Obtenido de www.ettus.com

5.2.2 Software

A continuación, se presentan los softwares utilizados para ejecutar la RAN y el 5GC en la red 5G. Estas aplicaciones fueron seleccionadas después de realizar un análisis comparativo de diversas alternativas de software de código abierto para la RAN y el 5GC, como se detalla en las secciones [4.4.1](#) y [4.4.2](#). La elección se fundamentó en su eficiencia y compatibilidad con los SDR disponibles para la implementación del prototipo de la red 5G.

srsRAN

srsRAN es una suite de radio de software para 4G y 5G, gratuita y de código abierto, desarrollada por la empresa Software Radio System (SRS). Se trata de un conjunto de software de código abierto que implementa las funciones del UE, eNodeB/gNodeB y núcleo de la red 4G (srsRAN, 2022a). Incluye tres softwares diferentes para ejecutar cada una de estas funciones:

- **srsUE:** una aplicación de UE con pila completa para 4G y 5G NSA/SA

- **srsENB**: una aplicación de eNodeB/gNodeB con pila completa para 4G y 5G NSA/SA
- **srsEPC**: una implementación ligera de EPC 4G con MME, HSS y S/P-GW

Todo el software srsRAN es compatible con el sistema operativo Linux y se ejecuta en hardware estándar de cómputo y radio. La **Figura 19** ilustra las tres funciones proporcionadas por el software srsRAN. No obstante, en este Trabajo de Integración Curricular, nos enfocamos únicamente en srsENB, al que se le ajustó la configuración para que operara como gNodeB en la red desplegada.

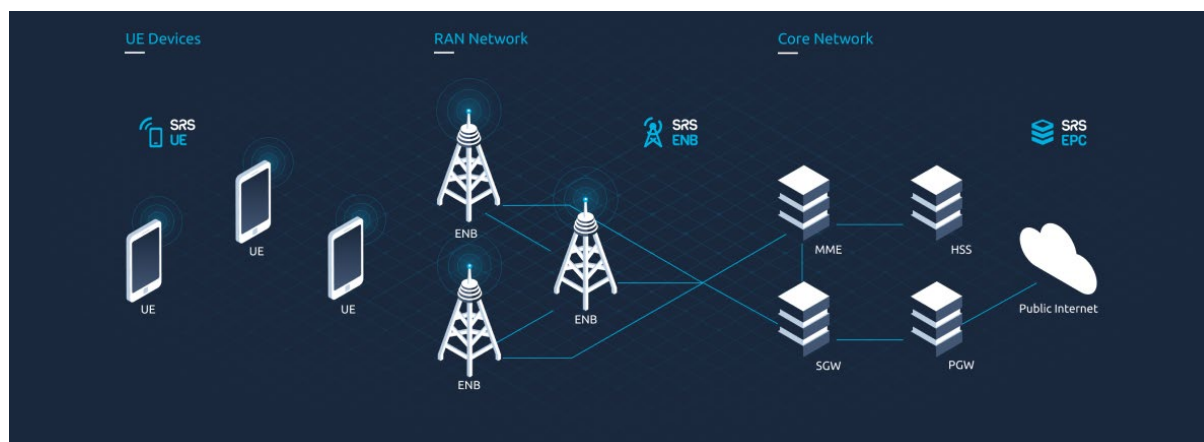


Figura 19. Suite de softwares de srsRAN

Fuente: Obtenido de (srsRAN, 2022b)

srsENB es un software que implementa una estación base LTE eNodeB. Funciona como una aplicación en un sistema operativo Linux estándar y se conecta a cualquier red central LTE (EPC) para crear una celda LTE local. Para la transmisión y recepción de señales de radio, srsENB necesita hardware SDR como el USRP de Ettus Research, BladeRF, Lime SDR, etc (srsRAN, 2022c). Cabe destacar que srsENB también soporta 5G NR. Para establecer una red SA 5G de extremo a extremo, se requiere usar el software srsUE, srsENB y un núcleo 5G de terceros, ya que srsRAN no dispone de software para el 5GC.

Open5GS

Open5GS es una implementación de código abierto en lenguaje C que contiene una serie de componentes de software y funciones de red que implementan las funciones principales 4G/5G NSA y 5G SA. Según Open5GS (2023b), el 5GC incluye las siguientes funciones de red:

- **NRF** - NF Repository Function
- **SCP** - Service Communication Proxy
- **AMF** - Access and Mobility Management Function
- **SMF** - Session Management Function

- **UPF** - User Plane Function
- **AUSF** - Authentication Server Function
- **UDM** - Unified Data Management
- **UDR** - Unified Data Repository
- **PCF** - Policy and Charging Function
- **NSSF** - Network Slice Selection Function
- **BSF** - Binding Support Function

La **Figura 20** muestra un diagrama de los elementos del plano de control y de usuario, así como sus interfaces de comunicación, para las configuraciones de 4G/5G NSA y 5G SA.

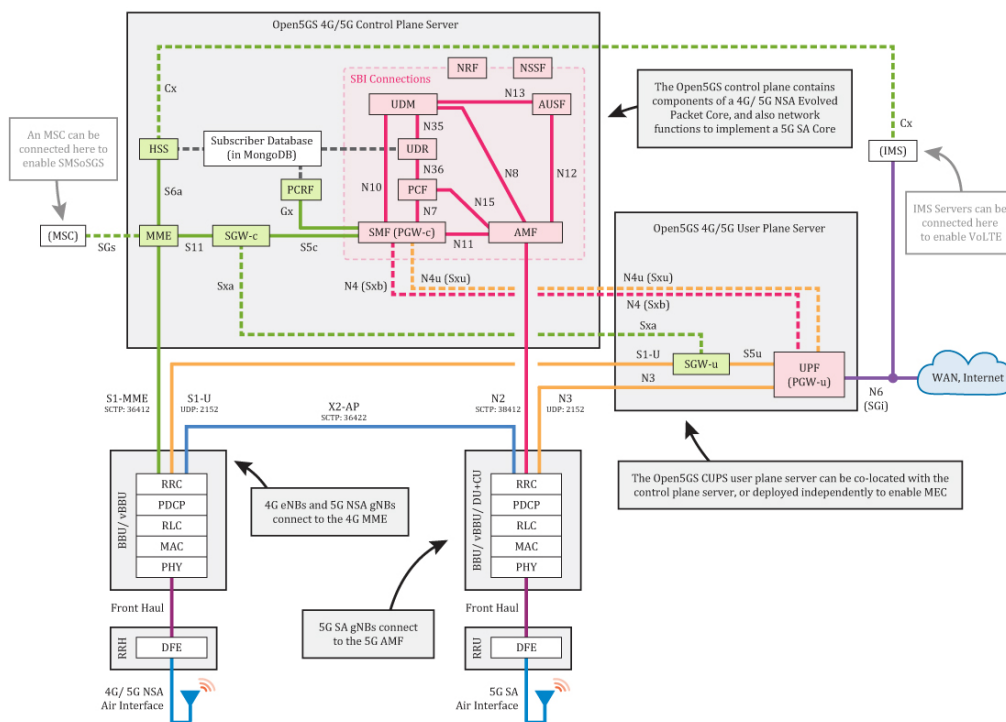


Figura 20. Diagrama de componentes y funciones de red de Open5GS
Fuente: Obtenido de (Open5GS, 2023b)

En resumen, para implementar la red 5G SA, se emplearon dos softwares específicos: srsRAN para la Radio Access Network y Open5GS para el 5G Core. srsENB, actuando como una estación base 5G gNodeB completamente basada en software, funcionó como una aplicación en un sistema operativo Linux estándar. Este componente se conectó a la red central NR (5GC) y estableció una celda NR local. En términos de transmisión y recepción de señales de radio, srsENB se apoyó en el hardware SDR USRP B210 y BladeRF 2.0.

5.3 Diseño de la arquitectura 5G NR

En esta sección del proyecto, nos enfocamos en el diseño de la arquitectura de la red 5G SA mediante el uso de tecnología SDR y software de código abierto. El objetivo primordial es crear una red que sea flexible, escalable y capaz de ajustarse a las necesidades particulares de los usuarios y los servicios de voz y datos. A lo largo de este apartado, se describirá detalladamente la interconexión de diversos componentes de la red, incluyendo las estaciones base, funciones de red, interfaces, entre otros.

5.3.1 Diseño propuesto

A continuación, se detallan los componentes y configuraciones utilizados en el despliegue de la red 5G SA.

La implementación de la RAN se llevó a cabo mediante el software de código abierto srsRAN, que proporciona una solución integral para la interfaz radio de la red 5G NR. Este software abarca las implementaciones de elementos clave de la RAN, como el gNodeB (gNB), y los canales físicos correspondientes. Para ejecutar srsRAN, se seleccionaron dispositivos SDR compatibles, específicamente los bladeRF 2.0 micro xA9 de Nuand y USRP B210 de Ettus, debido a su alta flexibilidad y capacidad de procesamiento. Estos dispositivos SDR actuaron como emisores de la señal NR en la banda n3.

Para permitir que el gNB reconociera el dispositivo SDR conectado, se empleó el controlador RF bladeRF para el bladeRF 2.0 micro xA9 y el driver UHD para el USRP B210. Estos controladores facilitaron la comunicación entre el gNodeB y el dispositivo SDR, conectado al ordenador a través de un puerto USB 3.0.

En cuanto al 5GC, se implementó utilizando el software de código abierto Open5GS, que proporciona una solución completa para las funciones y servicios de la red 5G. Este software abarca la implementación de varios componentes del 5GC, como el AMF (access and mobility management function), AUSF (authentication server function), UDM (unified data management), UDR (unified data repository), PCF (policy control function), SMF (session management function), UPF (user plane function), y NRF (network repository function).

Para optimizar el rendimiento, tanto el gNodeB como el 5GC se ejecutaron en un sistema que operaba con un kernel de baja latencia, así como un controlador de escalado de CPU para garantizar un rendimiento óptimo o modo de rendimiento constante. Además, con el objetivo de facilitar el despliegue, el núcleo de la red se implementó en el mismo equipo que la RAN.

La **Figura 21** ilustra un diagrama general de los componentes de hardware y software para los escenarios de pruebas.

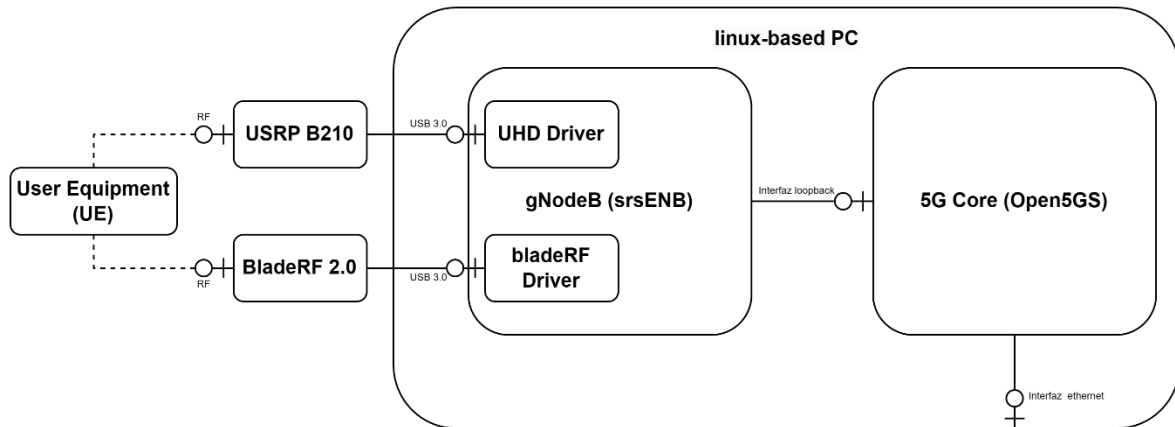


Figura 21. Diagrama general de los componentes de hardware y software para los escenarios de pruebas

Fuente: Elaboración propia

La configuración de las interfaces y los puertos utilizados para establecer la comunicación entre el gNB y las funciones de red del 5GC se realizó mediante diversas interfaces de loopback presentes en el equipo donde se desplegó la red 5G SA, todas con direcciones IP dentro del prefijo 127.0.0.0/8.

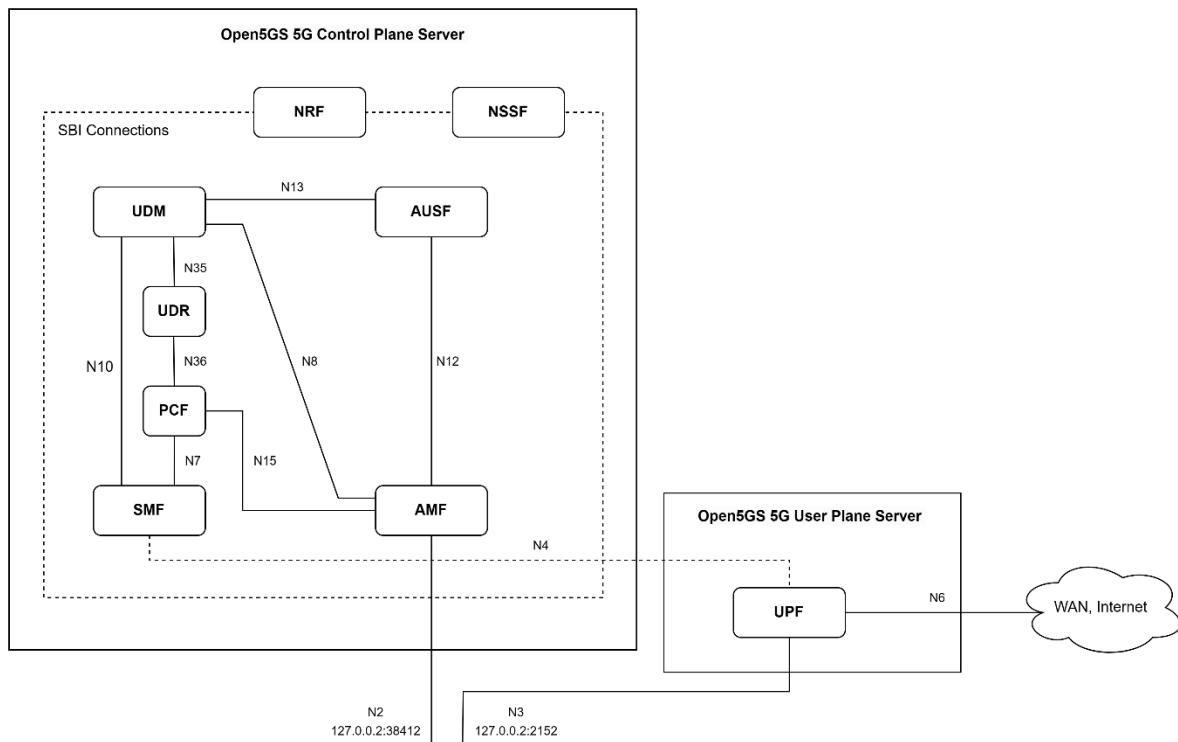


Figura 22. Arquitectura de los componentes del núcleo de 5G empleado

Fuente: Elaboración propia

Cada función de red tenía asignada una dirección IP y una serie de puertos para ofrecer sus servicios a otras funciones o al gNB. En la **Figura 22** se representan estas conexiones mediante la nomenclatura Nx, donde x es un número que identifica la interfaz. Por ejemplo, la interfaz N2 conecta el gNB con el AMF.

En este diseño, el gNB se conectó al puerto 38412 de la dirección IP 127.0.0.2 del AMF. Este puerto es utilizado por el AMF para el manejo de la gestión de las conexiones y la movilidad de los usuarios a través de la interfaz N2. Además, el AMF utiliza otros puertos con la misma dirección IP para comunicarse con otros componentes de la red, como el UDM a través de la interfaz N8. En el caso específico de la interfaz N8 mostrada en la **Figura 22**, se estableció una conexión entre la dirección IP 127.0.0.2 y el puerto 777 del AMF y la dirección IP 127.0.0.12 y el puerto 777 del UDM. De esta manera, el AMF pudo acceder a los servicios del UDM a través de dicho puerto.

Para simplificar la implementación, todas las funciones restantes de la red también fueron configuradas con direcciones IP en el mismo segmento de red loopback (127.0.0.0/8). Durante la implementación, se les asignaron las siguientes direcciones IP a cada función de red:

- UDM: 127.0.0.12
- AUSF: 127.0.0.11
- UDR: 127.0.0.20
- PCF: 127.0.0.13
- SMF: 127.0.0.4
- NRF: 127.0.0.10
- NSSF: 127.0.0.14
- AMF: 127.0.0.5
- UPF: 127.0.0.7

En la **Figura 23** se muestra un diagrama que representa las direcciones IP asignadas a cada función de red del 5GC.

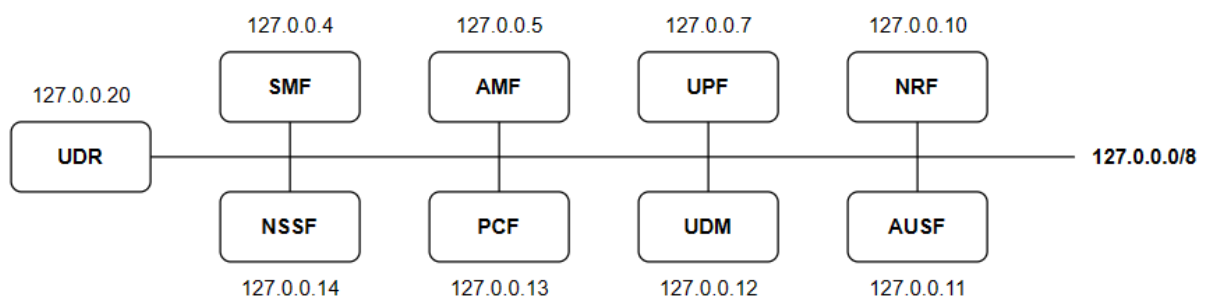


Figura 23. Despliegue de las comunicaciones en el equipo del gNB

Fuente: Elaboración propia

Tomando en cuenta las capas incluidas en srsENB, así como las funciones de red del 5GC mencionadas anteriormente, se pudo representar la arquitectura de la red 5G SA desplegada para el análisis en la **Figura 24**. Mediante esta propuesta de diseño, se aseguró la exitosa implementación de una red 5G completamente operativa. Esto se tradujo en la exitosa transmisión y recepción de señales NR a través de la interfaz aérea, así como en la realización de un proceso de registro (attach) por parte de un dispositivo de usuario comercial.

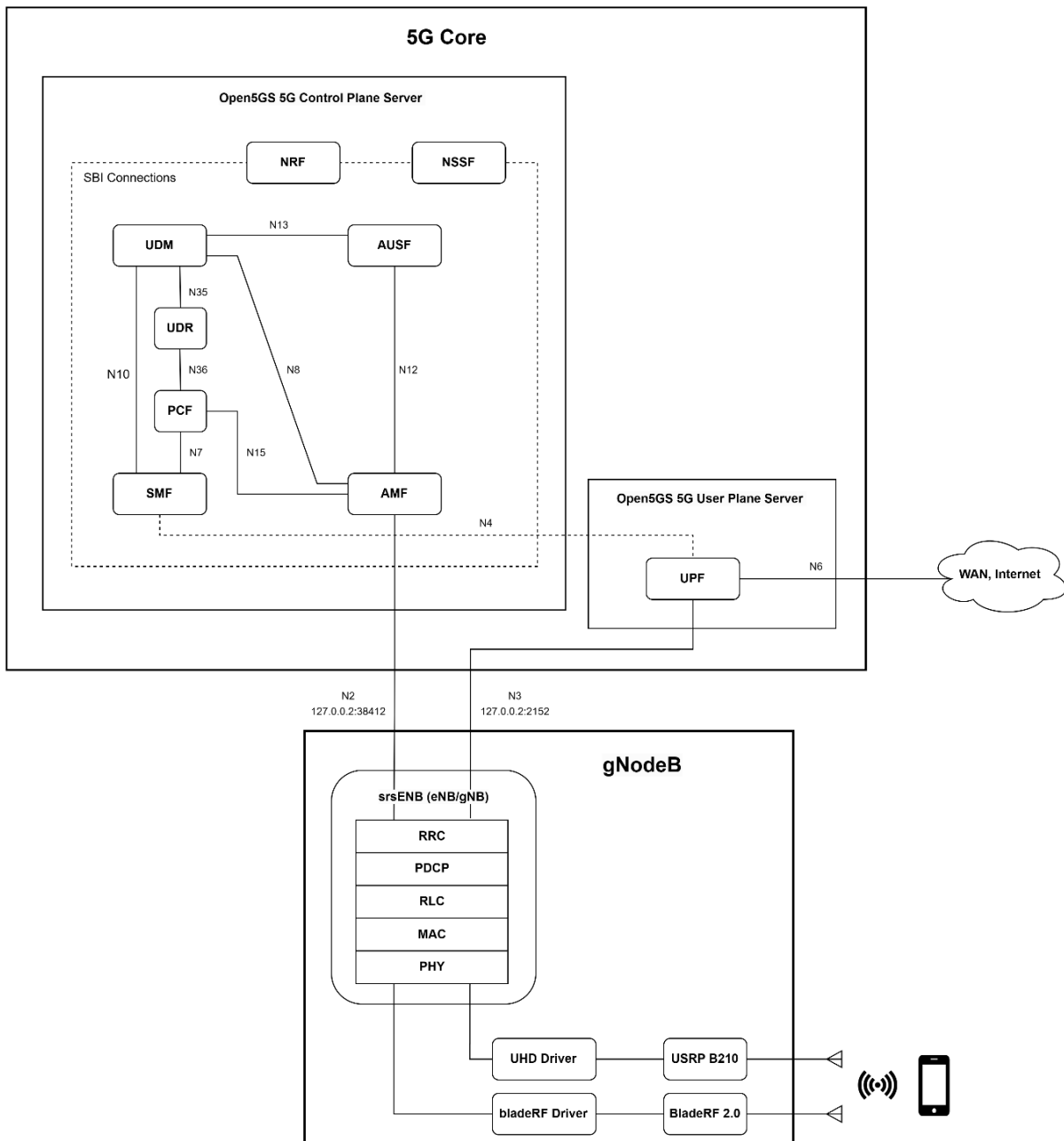


Figura 24. Arquitectura de la red 5G SA propuesta
Fuente: Elaboración propia

Desde una perspectiva más amplia, podemos observar esta arquitectura representada en la **Figura 25**.

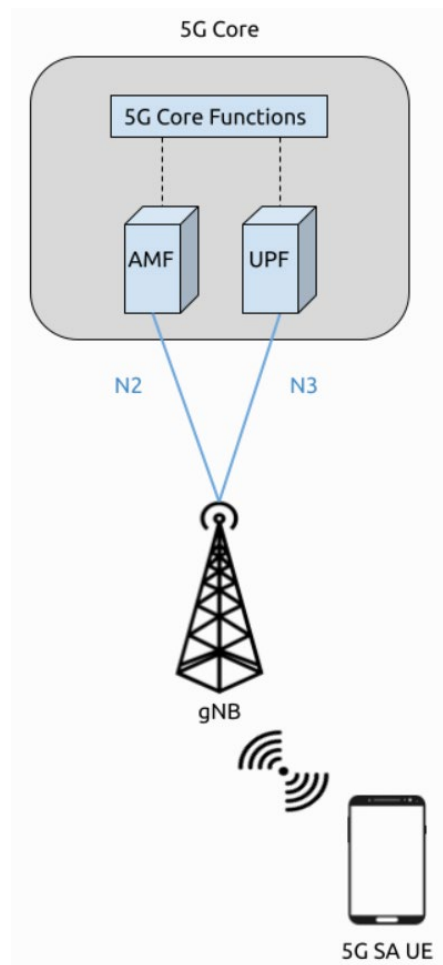


Figura 25. Descripción general de la red

Fuente: Obtenido de (srsRAN, 2023b)

5.4 Implementación

En este capítulo de implementación, se detallan los pasos que se llevaron a cabo para lograr la implementación exitosa de la arquitectura 5G SA. Se proporcionan explicaciones que abarcan desde la configuración de las tarjetas SIM/USIM hasta la instalación de los controladores para los SDRs y los programas srsRAN y Open5GS. También se incluye la configuración de los componentes de red necesarios, así como la preparación del Punto de Acceso (APN) en el dispositivo móvil para un registro adecuado en la red y para el acceso a Internet.

Con el propósito de facilitar la comprensión de esta sección, se han creado apéndices que contienen los archivos de configuración más relevantes. Estos apéndices se encuentran al final del documento y son referenciados a lo largo de las secciones subsiguientes.

5.4.1 Configuración de la SIM

Para llevar a cabo el registro exitoso del dispositivo de usuario en la red 5G, resultó crucial ajustar ciertos parámetros. Esto involucró la modificación del MMC, MNC e IMSI de la tarjeta SIM. Es importante resaltar que, para llevar a cabo la programación de este tipo de tarjetas, es necesario disponer de las claves ADM. En este caso, estas claves fueron proporcionadas por el fabricante una vez que las tarjetas fueron adquiridas.

Los pasos que se siguieron para realizar estas modificaciones fueron los siguientes:

1. Para comenzar, se instalaron las dependencias requeridas mediante el siguiente comando:

```
sudo apt-get install pcscd pcsc-tools libccid libpcsclite-dev python3-pyocard git
```

2. Luego, se procedió a conectar el lector de tarjetas SIM a la computadora y a insertar la tarjeta SIM programable en dicho lector. Para confirmar el estado de la conexión, se ingresó el siguiente comando:

```
pcsc_scan
```

Si el lector de tarjetas SIM era reconocido de manera exitosa, la terminal debería mostrarel mensaje "Card inserted".

3. Para llevar a cabo la reprogramación de las tarjetas, fue necesario descargar el código de PySIM junto con las dependencias de instalación:

```
git clone https://github.com/osmocom/pysim
cd pysim
sudo apt-get install --no-install-recommends \
  pcscd libpcsclite-dev \
  python3 \
  python3-setuptools \
  python3-pyocard \
  python3-pip
pip3 install -r requirements.txt
```

4. A continuación, se ejecutaron los siguientes comandos desde el directorio de pysim. En primer lugar, se verificó la configuración actual de la tarjeta SIM:

```
./pySim-read.py -p0
```

5. Posteriormente, se reconfiguró la tarjeta SIM al proporcionarle los siguientes parámetros:

```
./pySim-prog.py -p 0 -n Open5GS -a <ADM-KEY> -s <ICCID> -i <IMSI> -x <MCC> -y <MNC> -k <KI> -o <OPC>
```

6. Fue necesario establecer al menos el PLMN en 00101, y en caso deseado, reconfigurar otros parámetros de la tarjeta SIM. Para llevar a cabo esta configuración, se empleó el siguiente comando:

```
./pySim-prog.py -p 0 -n Open5GS -a 89043232 -s 898821100000722200 -i
001010000043201 -x 001 -y 01 -k 69A11D9D8F7FBFD7AAFA6DA98B6CF98E -o
0901F2ABDC631CC50FD16C02FC65FCD2
```

7. Fue necesario realizar ajustes en los campos relacionados con 5G en la tarjeta SIM, específicamente configurar o deshabilitar la ocultación del Identificador de Usuario del Suscriptor (SUPI por sus siglas en inglés) o la Cadena de Identidad de Usuario de Conexión Segura (SUCI por sus siglas en inglés). La desactivación de la ocultación de SUPI se llevó a cabo mediante los siguientes comandos, siendo necesario sustituir <ADM-KEY> con la clave ADM correspondiente a la tarjeta SIM en cuestión.

```
pySIM-shell (MF)> select MF
pySIM-shell (MF)> select ADF.USIM
pySIM-shell (MF/ADF.USIM)> select EF.UST
pySIM-shell (MF)> verify_adm <ADM-KEY>
pySIM-shell (MF/ADF.USIM/EF.UST)> ust_service_deactivate 124
pySIM-shell (MF/ADF.USIM/EF.UST)> ust_service_deactivate 125
```

8. Una vez concluidos estos procedimientos, los servicios UST 124 y 125 quedaron desactivados. Finalmente, se pudo confirmar la configuración realizada en la tarjeta SIM mediante el siguiente comando:

```
./pySim-read.py -p0
```

5.4.2 Instalación del controlador de hardware BladeRF

Para emplear el dispositivo SDR en la computadora con Ubuntu 22.04 LTS, conectado a través de USB 3.0, se requería la instalación de los controladores adecuados. A continuación, se detallan los pasos que se siguieron para instalar el controlador bladeRF correspondiente al BladeRF 2.0:

1. Se procedió a instalar las dependencias necesarias:

```
sudo apt-get install libusb-1.0-0-dev libusb-1.0-0 build-essential cmake
libncurses5-dev libtecla1 libtecla-dev pkg-config git wget
```

2. Se realizó la descarga, compilación e instalación del controlador bladeRF:

```
git clone https://github.com/Nuand/bladeRF.git ./bladeRF
cd ./bladeRF
cd host/
mkdir build
cd build
cmake ../
make -j4
```



```
sudo make install
```

3. Se generaron los enlaces necesarios hacia las bibliotecas recién instaladas:

```
sudo ldconfig
```

4. Luego, se procedió a la descarga del firmware correspondiente a la tarjeta bladeRF utilizando el siguiente comando:

```
sudo bladeRF-install-firmware
```

5. Se verificó si el controlador instalado en la computadora detectaba el dispositivo conectado de manera adecuada, mediante el siguiente comando:

```
bladeRF-cli -p
```

5.4.3 Instalación del controlador de hardware USRP B210

De manera similar al procedimiento con el software anterior, se siguieron los pasos detallados en el manual de instalación de Open5GS. A continuación, se detallan los pasos que se llevaron a cabo:

1. Para comenzar, se procedió a la instalación de MongoDB como el primer paso para gestionar la base de datos de suscriptores. En este sentido, se importó la clave pública empleada por el sistema de gestión de paquetes.

```
sudo apt update
sudo apt install gnupg
curl -fsSL https://pgp.mongodb.com/server-6.0.asc | sudo gpg -o
/usr/share/keyrings/mongodb-server-6.0.gpg --dearmor
```

2. Se creó el archivo de lista `/etc/apt/sources.list.d/mongodb-org-6.0.list` correspondiente a la versión de Ubuntu utilizada. En el caso de Ubuntu 22.04 (Jammy):

```
echo "deb [ arch=amd64,arm64 signed-by=/usr/share/keyrings/mongodb-server-6.0.gpg]
https://repo.mongodb.org/apt/ubuntu jammy/mongodb-org/6.0 multiverse" | sudo tee
/etc/apt/sources.list.d/mongodb-org-6.0.list
```

3. Se procedió a instalar los paquetes de MongoDB.

```
sudo apt update
sudo apt install -y mongodb-org
sudo systemctl start mongod
sudo systemctl enable mongod
```

4. Para instalar el software Open5GS, se añadió el repositorio correspondiente y se procedió a la instalación del software a través del gestor de paquetes apt-get. Esto se logró mediante los siguientes comandos:

```
sudo apt update
```

```
sudo apt install software-properties-common
sudo add-apt-repository ppa:open5gs/latest
sudo apt update
sudo apt install open5gs
```

5. Finalmente, se procedió con la instalación de la interfaz web del sistema para permitir la edición sencilla e intuitiva de la base de datos de usuarios. Esta instalación se realizó ejecutando los siguientes comandos:

```
sudo apt update
sudo apt install curl
curl -fsSL https://deb.nodesource.com/setup_18.x | sudo -E bash -
sudo apt install nodejs
curl -fsSL https://open5gs.org/open5gs/assets/webui/install | sudo -E bash -
```

5.4.4 Configuración de la red

gNodeB (srsENB)

srsRAN incorpora varios archivos de configuración para los programas srsENB, srsUE y srsEPC, como “*enb.conf*”, “*epc.conf*”, “*mbms.conf*”, “*rb.conf*”, “*rr.conf*”, “*sib.conf*”, “*ue.conf*” y “*user_db.csv*”. Estos archivos se ubican en la siguiente ruta: “*~/config/srsran*”.

Conforme se mencionó previamente, en esta implementación nos centraremos únicamente en el software srsENB. En consecuencia, nos limitaremos al archivo de configuración principal denominado “*enb.conf*” y al “*rr.conf*” para ajustar los recursos radio.

enb.conf

El archivo de configuración “*enb.conf*” contenía la configuración principal del programa. Este archivo estaba escrito en formato INI, un tipo de archivo de configuración en el que cada parámetro se definía como un conjunto de sección-propiedad-valor. Cada sección del archivo de configuración contenía varias propiedades, cada una con un valor correspondiente configurado. Cada parámetro de configuración venía acompañado de un comentario en el archivo por defecto, lo que permitía entender fácilmente su función. A continuación, se describen los parámetros principales que necesitaban modificarse para adaptar la red según nuestras necesidades.

- **enb.enb_id**: Corresponde al identificador en hexadecimal del eNodeB/gNodeB. Este valor es relevante en casos de handover, donde cada eNodeB/gNodeB debe tener un identificador distinto.
- **enb.mcc** y **enb.mnc**: Estos parámetros permiten configurar el Mobile Country Code (MCC) y Mobile Network Code (MNC) del eNodeB. La combinación de ambos define

el operador que brinda el servicio. Para alinearse con los IMSI de las SIM, utilizamos 001-01 como MCC y MNC.

- **enb.mme_addr**: Representa la dirección IP de la interfaz S1-C en el servicio MME del núcleo. Sin embargo, en implementaciones 5G, esta dirección se usa para las interfaces N2 y N3 en los servicios AMF y UPF del 5GC.
- **enb.gtp_bind_addr**: Es la IP de la interfaz de datos de usuario S1-U o la interfaz GTP en el eNodeB.
- **enb.s1c_bind_addr**: Corresponde a la IP de la interfaz de control S1-C o la interfaz S1-MME en el eNodeB.
- **enb.n_prb**: Indica el número de Bloques de Recursos Físicos (PRB, Physical Resource Blocks), proporcional al ancho de banda, siendo un PRB equivalente a 0,2MHz. Los valores permitidos para 4G son 6=1.4MHz, 15=3MHz, 25=5MHz, 50=10MHz, 75=15MHz y 100=20MHz. En implementaciones 5G, se limita a un máximo de 10MHz.
- **enb.tm**: Define el modo de transmisión (SISO, MIMO) del eNodeB. Hay cuatro opciones: una única antena, transmitir diversidad, CDD (Cyclic Delay Diversity) o multiplexado espacial de lazo cerrado.
- **enb.nof_ports**: Este valor indica el número de puertos de transmisión, con 1 por defecto y 2 para los modos de transmisión 2, 3 y 4.
- **rf.tx_gain**: Representa la ganancia de transmisión, en dBm. Define la potencia con la que el dispositivo SDR transmitirá.
- **rf.rx_gain**: Corresponde a la ganancia en la recepción. Define la amplificación de la señal recibida antes del procesamiento. Si se omite, el AGC (Control Automático de Ganancia) se habilita, ajustándose en tiempo real según la potencia recibida.
- **rf.device_name**: Se utilizó este parámetro para indicar la familia de controladores a emplear. Las opciones compatibles incluían "auto" (para usar el primer controlador encontrado), "UHD", "bladeRF", "soapy", "zmq" o "Sidekiq".
- **scheduler.nr_pdsch_mcs**: Parámetro que indica el esquema de modulación y codificación (MCS) utilizado para el canal físico de enlace descendente compartido (PDSCH) en una red 5G NR.
- **scheduler.nr_pusch_mcs**: Parámetro que indica el esquema de modulación y codificación (MCS) utilizado para el canal físico de enlace ascendente compartido (PUSCH) en una red 5G NR.

En este contexto, a continuación se presentan los parámetros del archivo de configuración “*enb.conf*” que incluyen los aspectos más importantes y necesarios para implementar la red 5G SA propuesta.

```
[enb]
enb_id = 0x19B
mcc = 001
mnc = 01
mme_addr = 127.0.0.2
gtp_bind_addr = 127.0.1.1
s1c_bind_addr = 127.0.1.1
s1c_bind_port = 0
n_prb = 50

[rf]
tx_gain = 80
rx_gain = 40

device_name = UHD

[scheduler]
nr_pdsch_mcs=28
nr_pusch_mcs=28
```

Es relevante subrayar que los valores de ganancia de transmisión (TX) y recepción (RX) requirieron ajustes cuando las unidades de usuario (UE) enfrentaban dificultades para visualizar la red o establecer una conexión. La intensidad de la señal de radiofrecuencia (RF) estaba sujeta a influencias de diversas condiciones físicas, variables en cada caso de uso y configuración.

Adicionalmente, el software, de manera predeterminada, emplea un valor máximo de MCS=28 para la tasa máxima. Sin embargo, esta elección podía resultar excesiva en función de las condiciones de RF. En estos casos, se hizo necesario reducir el MCS.

Por consiguiente, es crucial reconocer que la configuración mencionada anteriormente no garantizaba un rendimiento óptimo para todos los usuarios debido a las variaciones en las condiciones de RF. Se debe ajustar la configuración según sea necesario para asegurar un desempeño adecuado. El archivo de configuración utilizado para la implementación de la red se encuentra disponible en el **Anexo 1**.

rr.conf

El archivo “*rr.conf*” fue empleado para la configuración de los recursos radio del sistema, permitiendo ajustar diversos parámetros de nivel bajo en concordancia con los estándares LTE y NR. En otras palabras, los parámetros configurables en este archivo coinciden con aquellos presentes en las estaciones eNodeB/gNodeB utilizadas por los operadores. Este archivo también habilita la configuración de varios sectores de celdas, el handover y la opción de habilitar 5G NSA o 5G SA (desactivado por defecto). Para este último propósito, era

necesario realizar ajustes en el archivo *"rr.conf"* para agregar una celda NR a la lista de celdas. Además, las celdas LTE predeterminadas debían ser comentadas o eliminadas por completo de la lista para evitar su consideración en el sistema. La configuración de la celda NR se llevaba a cabo al final del archivo *"rr.conf"* de la siguiente manera:

```
nr_cell_list =
(
{
rf_port = 0;
cell_id = 1;
root_seq_idx = 1;
tac = 7;
pci = 500;
dl_arfcn = 368500;
coreset0_idx = 6;
band = 3;
}
);
```

En la configuración, se optó por comentar la lista de celdas LTE, aunque también era factible eliminarla o dejarla vacía. El archivo de configuración completo utilizado se encuentra disponible en el **Anexo 2**.

Por otro lado, es importante señalar que los archivos *"rb.conf"* y *"sib.conf"* se mantuvieron en sus configuraciones predeterminadas en este Trabajo de Integración Curricular y, por lo tanto, no fueron objeto de análisis detallado.

5G Core (Open5GS)

Una vez completada la instalación, varios servicios se activaron automáticamente en la máquina local, ejecutándose en segundo plano. Estos servicios incluyen: *"open5gs-mmed"*, *"open5gs-sgwcd"*, *"open5gs-smfd"*, *"open5gs-amfd"*, *"open5gs-sgwud"*, *"open5gs-upfd"*, *"open5gs-hssd"*, *"open5gs-pcrfd"*, *"open5gs-nrfd"*, *"open5gs-ausfd"*, *"open5gs-udmd"*, *"open5gs-pcfd"*, *"open5gs-nssf"*, *"open5gs-bsfd"*, *"open5gs-udrd"* y *"open5gs-webui"*. Esta colección de servicios abarca tanto servicios de 4G como de 5G SA. En este Trabajo de Integración Curricular, se utilizaron exclusivamente aquellos relacionados con el núcleo 5GC, que son los siguientes:

- NRF - NF Repository Function: Implementado por el servicio *"open5gs-nrfd"*.
- AMF - Access and Mobility Management Function: Implementado por el servicio *"open5gs-amfd"*.
- SMF - Session Management Function: Implementado por el servicio *"open5gs-smfd"*.
- UPF - User Plane Function: Implementado por el servicio *"open5gs-upfd"*.

- AUSF - Authentication Server Function: Implementado por el servicio “*open5gs-ausfd*”.
- UDM - Unified Data Management: Implementado por el servicio “*open5gs-udmd*”.
- UDR - Unified Data Repository: Implementado por el servicio “*open5gs-udrd*”.
- PCF - Policy and Charging Function: Implementado por el servicio “*open5gs-pcfd*”.
- NSSF - Network Slice Selection Function: Implementado por el servicio “*open5gs-nssfd*”.
- BSF - Binding Support Function: Implementado por el servicio “*open5gs-bsfd*”.

Es esencial considerar que cada uno de estos servicios puede ejecutarse de forma independiente en diferentes máquinas, estableciendo comunicación mediante interfaces de red. Sin embargo, en nuestro escenario, se ejecutan en la misma máquina y utilizan las direcciones IP de loopback del sistema para la comunicación.

Cada servicio tiene su propio archivo de configuración, donde se pueden ajustar sus parámetros. Por defecto, todos los servicios están configurados para operar en una única computadora, con interfaces de red configuradas para utilizar direcciones IP de loopback.

Los archivos de configuración mencionados se encuentran en el directorio “*/etc/open5gs/*”. Las modificaciones clave se llevaron a cabo en los archivos de configuración “*amf.yaml*” y “*upf.yaml*”, e incluyen:

- Modificación del TAC en la configuración de AMF a 7, según lo definido en el archivo de configuración “*rr.conf*”.
- Asegurarse de que las direcciones NGAP y GTPU sean correctas en los archivos de configuración de AMF y UPF, tal como se establecieron en el archivo de configuración “*enb.conf*”.
- Actualización de los valores de PLMN de acuerdo a lo definido en el archivo de configuración “*enb.conf*”.

amf.yaml

Para ajustar la dirección IP NGAP (127.0.0.2), el ID PLMN (00101) y el TAC (7), se procedió a editar el archivo “*/etc/open5gs/amf.yaml*”. Las configuraciones en este archivo se realizaron de la siguiente manera:

```

amf:
  sbi:
    - addr: 127.0.0.5
      port: 7777
  ngap:
    - addr: 127.0.0.2

```

```

metrics:
- addr: 127.0.0.5
port: 9090
guami:
- plmn_id:
mcc: 001
mnc: 01
amf_id:
region: 2
set: 1
tai:
- plmn_id:
mcc: 001
mnc: 01
tac: 7
plmn_support:
- plmn_id:
mcc: 001
mnc: 01
s_nssai:
- sst: 1
security:
integrity_order : [ NIA2, NIA1, NIA0 ]
ciphering_order : [ NEA0, NEA1, NEA2 ]
network_name:
full: Open5GS
amf_name: open5gs-amf0

```

El archivo de configuración completo que se empleó se lo puede encontrar en el **Anexo**

3.

upf.yaml

Para establecer la dirección GTP-U (127.0.0.2) en el archivo de configuración UPF, se realizó la siguiente modificación en el archivo “*/etc/open5gs/upf.yaml*”:

```

upf:
pfcpc:
- addr: 127.0.0.7
gtpu:
- addr: 127.0.0.2
subnet:
- addr: 10.45.0.1/16
- addr: 2001:db8:cafe::1/48

```

El archivo de configuración completo que se empleó se lo puede encontrar en el **Anexo**

4.

Tras configurar los archivos según los pasos anteriores, resulta crucial reiniciar los servicios AMF y UPF. Esta acción debe realizarse cada vez que se efectúe alguna modificación en los archivos de configuración correspondientes, garantizando así que dichos cambios surtan efecto. Para llevar a cabo esta operación, ejecutamos el siguiente comando:

```

sudo systemctl restart open5gs-amfd
sudo systemctl restart open5gs-upfd

```

5.4.5 Registro de suscriptores

El último paso involucró registrar a los suscriptores a través de la interfaz web de Open5GS (WebUI). Fue esencial verificar que los valores en cada campo coincidieran con los datos asociados a la tarjeta SIM en uso.

Una configuración precisa del APN (Nombre de Punto de Acceso) resultaba fundamental. Un error en esta configuración podría impedir que el Usuario de Equipo (UE) estableciera una conexión a internet.

El proceso para registrar un UE en el núcleo consistió en acceder a la interfaz web de Open5GS (WebUI) a través de `https://localhost:3000`, utilizando las credenciales de usuario "admin" y la contraseña "1234".

Dentro de la interfaz, se seleccionó la opción "Subscriber" y se hizo clic en el icono "+" para añadir los datos de acuerdo a la tarjeta SIM, como se ilustra en la **Figura 26**.

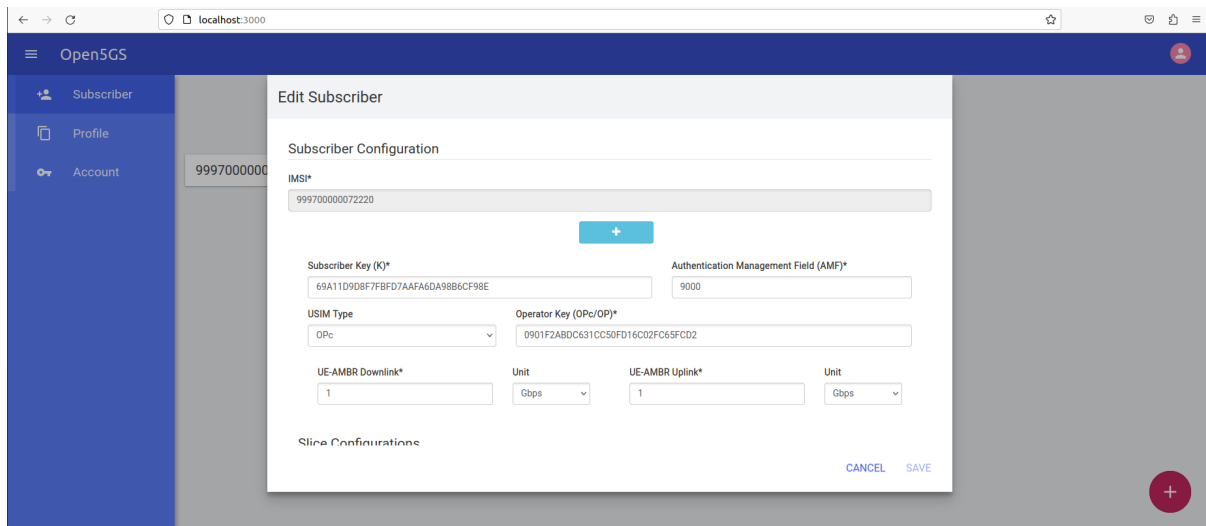


Figura 26: Registro de suscriptores en Open5GS (WebUI)

Fuente: Elaboración propia

Es importante completar también el AMF y el APN que está sirviendo al terminal. Hay que tener en cuenta que, en el caso de la tarjeta SIM modelo symsoISIM-SJA2, el valor del AMF será 9000.

Para este entorno de pruebas, se utilizará el siguiente APN:

- Nombre: internet
- APN: internet
- Tipo: IPv4

Con el objetivo de establecer una conexión entre PGWU/UPF y la WAN (Internet), se activó el reenvío de IPv4 e IPv6 en el sistema y se implementaron dos reglas NAT mediante iptables.

Para activar el reenvío, se ejecutó el siguiente comando:

```
sudo sysctl -w net.ipv4.ip_forward=1
sudo sysctl -w net.ipv6.conf.all.forwarding=1
```

Y para agregar la regla NAT, se utilizó el siguiente comando:

```
sudo iptables -t nat -A POSTROUTING -s 10.45.0.0/16 ! -o ogstun -j MASQUERADE
sudo ip6tables -t nat -A POSTROUTING -s 2001:db8:cafe::/48 ! -o ogstun -j MASQUERADE
```

5.4.6 Configuración del UE

Para facilitar la conexión del dispositivo del usuario a la red, se llevaron a cabo los siguientes pasos:

- Habilitar la tarjeta SIM.
- Activar el modo 5G autónomo (SA).
- Desactivar VoLTE y/o VoNR.
- Configurar el Punto de Acceso (APN).
- Forzar el tipo de red preferido exclusivamente a NR.

Habilitar SIM, 5G SA

El primer paso en la configuración del UE consistió en confirmar la habilitación tanto de la tarjeta SIM como del modo 5G SA. La SIM se insertó en la ranura SIM 1, sin otras tarjetas SIM presentes.

Las configuraciones implementadas en el dispositivo móvil se pueden observar en las **Figura 27**. En resumen, se verificó la correcta activación de la tarjeta SIM y la habilitación de los datos móviles, así como la activación del modo 5G SA. Cabe destacar que, de manera predeterminada, los dispositivos Xiaomi tienen esta opción desactivada. Para habilitar la opción SA oculta, es necesario marcar al "##726633##", aunque es importante señalar que este proceso puede variar según el modelo del dispositivo.

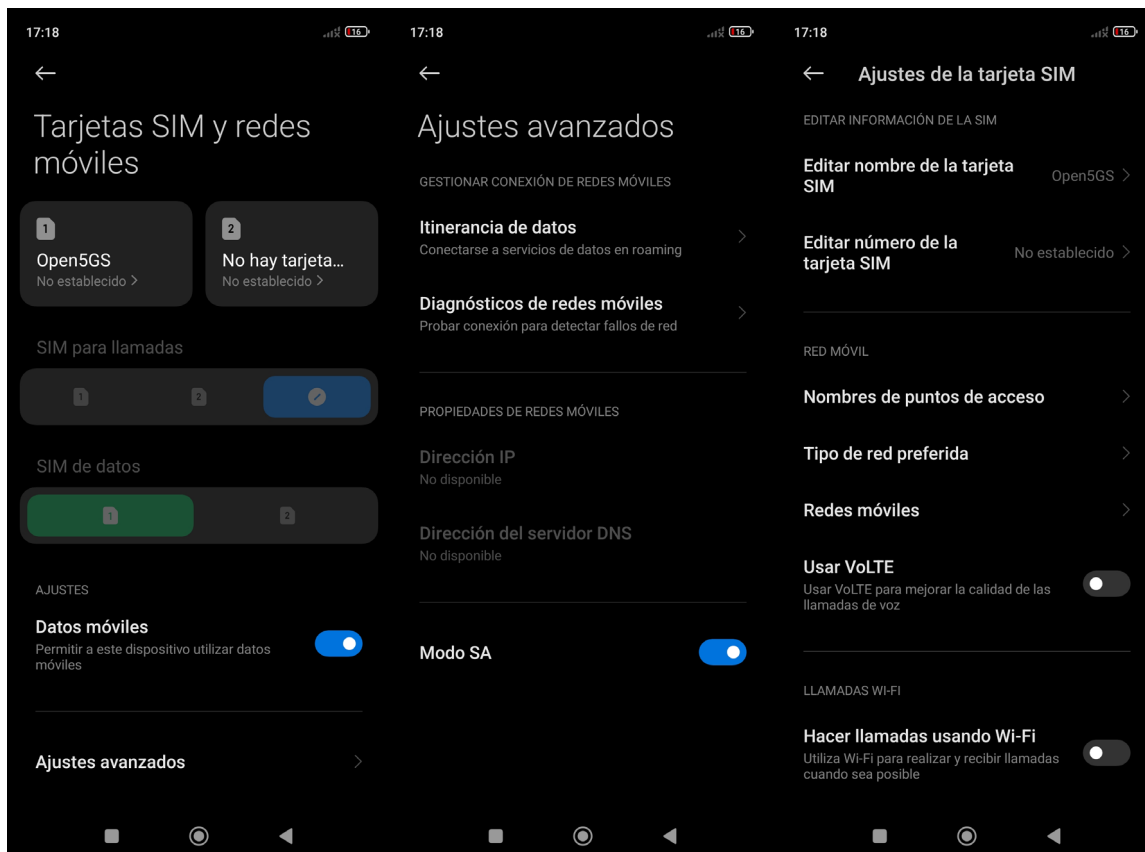


Figura 27. Ajustes de la tarjeta SIM
Fuente: Elaboración propia

Configuración del APN

Se añadió un APN al dispositivo del usuario para facilitar la conexión a Internet. Este APN coincidió con el definido en la entrada del suscriptor en el núcleo mediante la interfaz web de Open5Gs. Es crucial tener en cuenta los siguientes puntos: los Protocolos del APN y el Protocolo de roaming APN deben configurarse en IPv4, conforme al registro de suscriptor de Open5GS. La configuración de IPv6 o IPv4/6 puede ocasionar problemas de conexión a Internet. Se recomienda dejar todas las demás opciones en sus valores predeterminados.

Este procedimiento se llevó a cabo a través de la configuración de red del equipo de usuario (UE), siguiendo el ejemplo proporcionado en la **Figura 28**. Por lo general, se accede a esta configuración mediante la siguiente ruta (o una similar):

Ajustes > Tarjetas SIM y redes móviles > SIM > Nombres de puntos de acceso

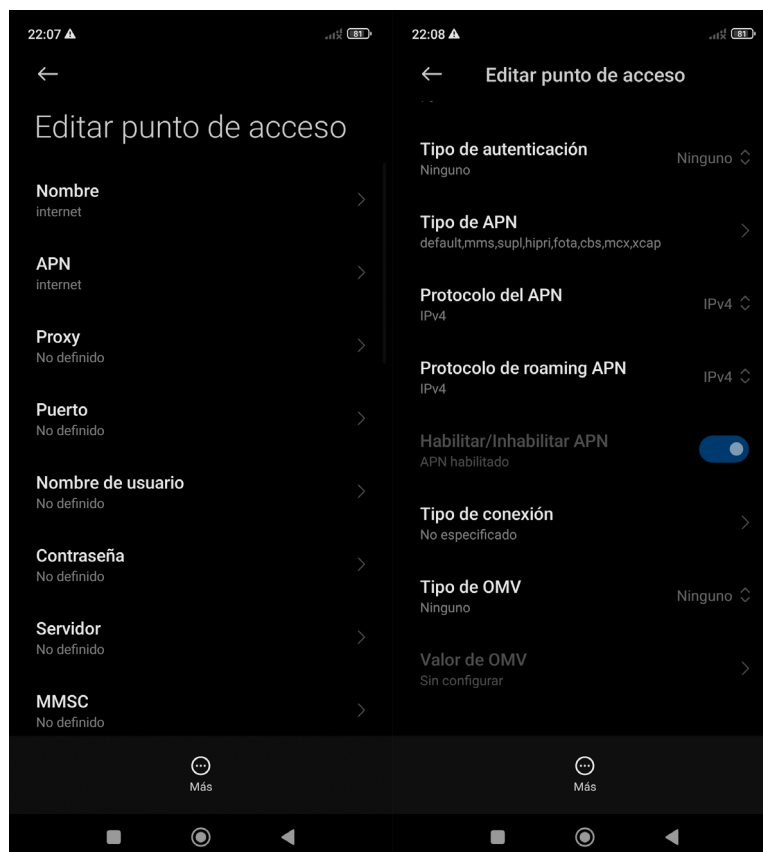


Figura 28. Configuración de APN en UE comercial
Fuente: Elaboración propia

Forzar el tipo de red preferido a exclusivamente NR

Se empleó la aplicación "5G Switch – Force 5G Only" para forzar al dispositivo a detectar exclusivamente redes 5G NR, una funcionalidad eficaz en dispositivos no rooteados. Como parte de esta configuración, se integró esta herramienta para garantizar que el dispositivo pudiera identificar y conectarse a la red. Aunque no era un requisito esencial para la conexión del teléfono, simplificó la conexión automática y constante a la red.

Al ejecutar la aplicación, era posible seleccionar la opción "Establecer tipo de red preferido: NR Only" del menú, como se muestra en la **Figura 29**. Alternativamente, se podía acceder al mismo menú sin necesidad de instalar esta aplicación marcando al "###4636###".

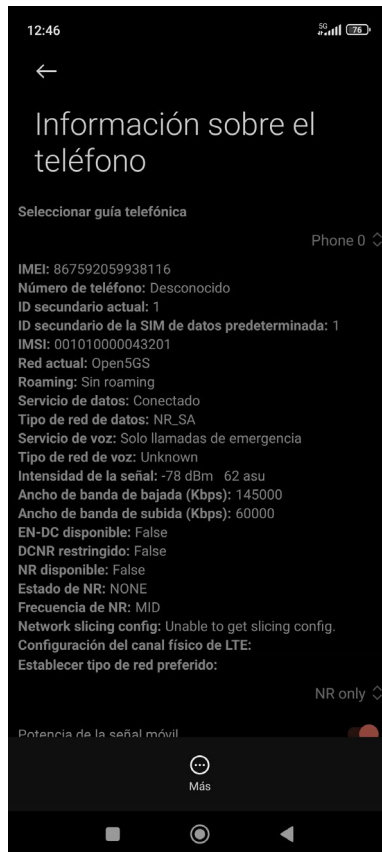


Figura 29. Tipo de red preferido: NR Only
Fuente: Elaboración propia

5.4.7 Conexión a la red

Para conectar el UE a la red, se requería seguir los siguientes pasos una vez que tanto el teléfono como la red hubieran sido configurados:

- Ejecutar el gNB y verificar su correcta conexión al núcleo.
- Buscar la red desde la UE.
- Seleccionar y establecer conexión con la red.
- Verificar el archivo adjunto.
- Iniciar la transmisión de datos.

Verificar que el núcleo se está ejecutando correctamente

Inicialmente, era recomendable verificar la correcta ejecución de Open5GS. Esto se lograba mediante el uso del siguiente comando:

```
ps aux | grep open5gs
```

En caso de que el núcleo se hubiera ejecutado de manera exitosa, la salida esperada consistía en lo siguiente:

```

open5gs 1601 0.0 0.0 141680 15872 ? Ssl 10:36 0:00 /usr/bin/open5gs-bsfd -c
/etc/open5gs/bsf.yaml
open5gs 1606 0.0 0.1 134452 24840 ? Ssl 10:36 0:01 /usr/bin/open5gs-nrfd -c
/etc/open5gs/nrf.yaml
open5gs 1613 0.0 0.2 147068 41720 ? Ssl 10:36 0:02 /usr/bin/open5gs-scpd -c
/etc/open5gs/scp.yaml
open5gs 2663 0.0 0.1 2801740 16788 ? Ssl 10:36 0:02 /usr/bin/open5gs-hssd -c
/etc/open5gs/hss.yaml
open5gs 2675 0.0 0.1 2800268 16568 ? Ssl 10:36 0:02 /usr/bin/open5gs-pcrfd -c
/etc/open5gs/pcrf.yaml
open5gs 2676 0.0 0.1 185572 21584 ? Ssl 10:36 0:00 /usr/bin/open5gs-pcfd -c
/etc/open5gs/pcf.yaml
open5gs 2690 0.0 0.1 169668 20768 ? Ssl 10:36 0:00 /usr/bin/open5gs-udrd -c
/etc/open5gs/udr.yaml
open5gs 3065 0.0 0.1 155984 20136 ? Ssl 10:36 0:00 /usr/bin/open5gs-amfd -c
/etc/open5gs/amf.yaml
open5gs 3067 0.0 0.0 136052 15960 ? Ssl 10:36 0:00 /usr/bin/open5gs-ausfd -c
/etc/open5gs/ausf.yaml
open5gs 3071 0.0 0.0 2778684 14404 ? Ssl 10:36 0:02 /usr/bin/open5gs-mmed -c
/etc/open5gs/mme.yaml
open5gs 3074 0.0 0.0 134300 15416 ? Ssl 10:36 0:00 /usr/bin/open5gs-nssf -c
/etc/open5gs/nssf.yaml
open5gs 3079 0.0 0.1 260852 19656 ? Ssl 10:36 0:00 /usr/bin/open5gs-sgwcd -c
/etc/open5gs/sgwc.yaml
open5gs 3081 0.0 0.1 249660 17840 ? Ssl 10:36 0:00 /usr/bin/open5gs-sgwud -c
/etc/open5gs/sgwu.yaml
open5gs 3084 0.0 0.2 3127048 44456 ? Ssl 10:36 0:02 /usr/bin/open5gs-smfd -c
/etc/open5gs/smf.yaml
open5gs 3091 0.0 0.1 136072 17136 ? Ssl 10:36 0:00 /usr/bin/open5gs-udmd -c
/etc/open5gs/udm.yaml
open5gs 3099 0.0 0.1 274176 24588 ? Ssl 10:36 0:00 /usr/bin/open5gs-upfd -c
/etc/open5gs/upf.yaml

```

En total debería haber 16 procesos en ejecución. Una vez que el núcleo estaba en ejecución, resultaba beneficioso examinar los registros del AMF durante la realización de las pruebas. Esto proporciona claridad acerca de los momentos en que el gNB se conectaba y cuándo el UE establecía una conexión exitosa con la red.

Para visualizar esta información, se empleó el siguiente comando:

```
tail -f /var/log/open5gs/amf.log
```

Debería ver una salida similar a la siguiente:

```

04/03 10:36:52.012: [sctp] INFO: AMF initialize...done (./src/amf/app.c:33)
04/03 10:36:52.049: [sbi] INFO: [aea4db10-d1fa-41ed-916b-e56218b693e5] (NRF-notify) NF
registered (./lib/sbi/nrf-handler.c:632)
04/03 10:36:52.049: [sbi] INFO: [aea4db10-d1fa-41ed-916b-e56218b693e5] (NRF-notify) NF
Profile updated (./lib/sbi/nrf-handler.c:642)
04/03 10:36:52.049: [sbi] WARNING: [aea4db10-d1fa-41ed-916b-e56218b693e5] (NRF-notify) NF
has already been added (./lib/sbi/nrf-handler.c:636)
04/03 10:36:52.049: [sbi] INFO: [aea4db10-d1fa-41ed-916b-e56218b693e5] (NRF-notify) NF
Profile updated (./lib/sbi/nrf-handler.c:642)
04/03 10:36:52.049: [sbi] WARNING: NF EndPoint updated [127.0.0.12:80]
(./lib/sbi/context.c:1618)
04/03 10:36:52.049: [sbi] WARNING: NF EndPoint updated [127.0.0.12:7777]
(./lib/sbi/context.c:1527)
04/03 10:36:52.238: [app] INFO: SIGHUP received (./src/main.c:58)
04/03 10:36:52.350: [sbi] INFO: [aea6bae8-d1fa-41ed-904f-f78f7a58f5f3] (NRF-notify) NF
registered (./lib/sbi/nrf-handler.c:632)

```

```
04/03 10:36:52.350: [sbi] INFO: [aea6bae8-d1fa-41ed-904f-f78f7a58f5f3] (NRF-notify) NF Profile updated (../lib/sbi/nrf-handler.c:642)
```

Ejecutar el gNB

Para ejecutar el gNB, se necesitó utilizar el siguiente comando, en el cual se especificó el archivo de configuración *"enb.conf"*:

```
sudo srsenb enb.conf
```

El comando anterior asume que el archivo de configuración se ubica en el mismo directorio, por lo que se aconsejaba ejecutarlo en *"~/config/srsran/"*.

Una vez que la conexión del gNB con el núcleo se estableció con éxito, la información resultante se mostró en la consola:

```
--- Software Radio Systems LTE eNodeB ---
Reading configuration file enb.conf...

Opening 1 channels in RF device=default with args=default
Supported RF device list: UHD bladeRF zmq file
Trying to open RF device 'UHD'
NG connection successful
[INFO] [UHD] linux; GNU C++ version 9.4.0; Boost_107100; UHD_4.2.0.HEAD-0-g197cdc4f
[INFO] [LOGGING] Fastpath logging disabled at runtime.
Opening USRP channels=1, args: type=b200, master_clock_rate=23.04e6
[INFO] [UHD RF] RF UHD Generic instance constructed
[INFO] [B200] Detected Device: B200mini
[INFO] [B200] Operating over USB 3.
[INFO] [B200] Initialize CODEC control...
[INFO] [B200] Initialize Radio control...
[INFO] [B200] Performing register loopback test...
[INFO] [B200] Register loopback test passed
[INFO] [B200] Asking for clock rate 23.040000 MHz...
[INFO] [B200] Actually got clock rate 23.040000 MHz.
RF device 'UHD' successfully opened

==== eNodeB started ====
Type <t> to view trace
Setting frequency: DL=1842.5 Mhz, DL_SSB=1843.25 Mhz (SSB-ARFCN=368650), UL=1747.5 MHz for
cc_idx=0 nof_prb=52
```

El mensaje "NG connection successful" confirmó la exitosa conexión de srsENB con el núcleo 5GC. Para garantizar un mayor nivel de certeza, se revisó lo siguiente en el registro de AMF:

```
04/03 13:25:13.469: [amf] INFO: gNB-N2 accepted[127.0.0.1]:47633 in ng-path module
(../src/amf/ngap-sctp.c:113)
04/03 13:25:13.469: [amf] INFO: gNB-N2 accepted[127.0.0.1] in master_sm module
(../src/amf/amf-sm.c:706)
04/03 13:25:13.469: [amf] INFO: [Added] Number of gNBs is now 1 (../src/amf/context.c:1034)
```

Conexión del UE a la red 5G

La búsqueda de redes se inicia desde el dispositivo del usuario, accediendo a la opción correspondiente a través de la siguiente ruta de menú (u otra similar):

Ajustes > Tarjetas SIM y redes móviles > SIM1 > Redes móviles

En este momento, el dispositivo inicia la búsqueda de las redes móviles disponibles y se conecta automáticamente a la preferida. También se brinda la opción de seleccionar manualmente la red deseada una vez que se han listado todas las disponibles.

Si el dispositivo ha logrado recibir de manera exitosa los SIBs, especialmente el SIB1, y ha detectado la red, esta última se mostrará en la lista de operadores disponibles. En dicha lista, se identificará con los nombres "Open5GS 5G" o "00101 5G". Sin embargo, en caso de que la PLMN se haya configurado de manera diferente, la red podría aparecer bajo el nombre "[PLMN] 5G".

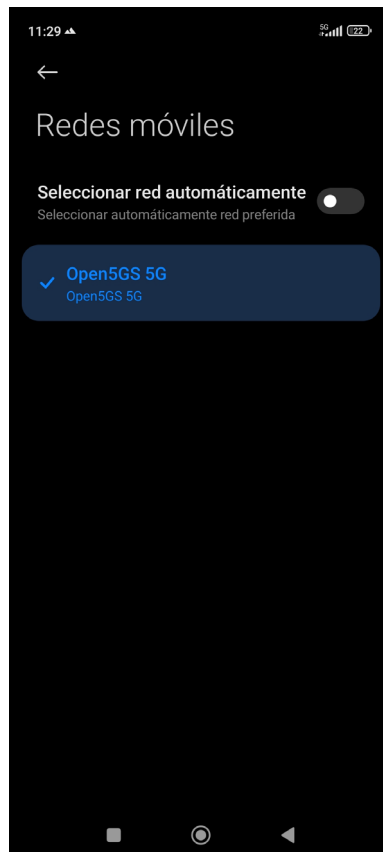


Figura 30. Conexión del UE a la red Open5GS
Fuente: Elaboración propia

Como se observa en la **Figura 30**, la única red 5G disponible es Open5GS, y el dispositivo del usuario se conectó automáticamente a ella.

Con el propósito de confirmar que la conexión se había establecido sin problemas, se monitorizaron tanto los registros del AMF como la información que se mostraba en la consola del gNB.

Si el UE se había conectado adecuadamente a la red, se esperaba observar una actualización en la salida de la consola del srsENB. Esta actualización se manifestaría de la siguiente manera:

```
==== eNodeB started ====
Type <t> to view trace
Setting frequency: DL=1842.5 Mhz, DL_SSB=1843.25 Mhz (SSB-ARFCN=368650), UL=1747.5 MHz for
cc_idx=0 nof_prb=52

RACH: slot=3051, cc=0, preamble=6, offset=10, temp_crnti=0x4601
User 0x46 connected
```

La confirmación del proceso de registro se evidenciaba cuando la consola presentaba el mensaje "User 0x46 connected".

Por otro lado, el registro de AMF debía exhibir una estructura similar a la siguiente:

```
04/27 13:16:31.746: [amf] INFO: InitialUEMessage (./src/amf/ngap-handler.c:361)
04/27 13:16:31.746: [amf] INFO: [Added] Number of gNB-UEs is now 1
(./src/amf/context.c:2036)
04/27 13:16:31.746: [amf] INFO: RAN_UE_NGAP_ID[0] AMF_UE_NGAP_ID[78] TAC[7] CellID[0x0]
(./src/amf/ngap-handler.c:497)
04/27 13:16:31.746: [amf] INFO: [suci-0-001-01-0-0-0-0000068960] Known UE by 5G-
S_TMSI[AMF_ID:0x20040,M_TMSI:0xdd00ff1a] (./src/amf/context.c:1402)
04/27 13:16:31.746: [gmm] INFO: Registration request (./src/amf/gmm-sm.c:134)
04/27 13:16:31.746: [gmm] INFO: [suci-0-001-01-0-0-0-0000068960] 5G-
S_GUTI[AMF_ID:0x20040,M_TMSI:0xdd00ff1a] (./src/amf/gmm-handler.c:169)
04/27 13:16:31.913: [gmm] INFO: [imsi-001010000068960] Registration complete
(./src/amf/gmm-sm.c:1063)
04/27 13:16:31.913: [amf] INFO: [imsi-001010000068960] Configuration update command
(./src/amf/nas-path.c:389)
04/27 13:16:31.913: [gmm] INFO: UTC [2023-04-27T13:16:31] Timezone[0]/DST[0]
(./src/amf/gmm-build.c:502)
04/27 13:16:31.913: [gmm] INFO: LOCAL [2023-04-27T13:16:31] Timezone[0]/DST[0]
(./src/amf/gmm-build.c:507)
04/27 13:16:32.105: [gmm] INFO: UE SUPI[imsi-001010000068960] DNN[srsapn] S_NSSAI[SST:1
SD:0xfffffff] (./src/amf/gmm-handler.c:1042)
```

Trazas del srsENB

La siguiente salida de consola de ejemplo muestra el seguimiento srsENB de un UE que envía y recibe datos a través de la red:

```
-----DL-----|-----UL-----
rat pci rnti cqi ri mcs brate ok nok (%) | pusch pucch phr mcs brate ok nok (%) bsr
nr 0 4601 15 0 25 1.2M 40 0 0% | 15.6 12.0 0 8 81k 10 0 0% 0.0
nr 0 4601 12 0 25 25M 837 0 0% | 15.4 16.6 0 8 548k 68 0 0% 0.0
nr 0 4601 11 0 25 27M 879 0 0% | 15.4 16.6 0 8 202k 25 0 0% 0.0
nr 0 4601 9 0 25 27M 900 0 0% | 15.4 16.5 0 8 202k 25 0 0% 0.0
nr 0 4601 10 0 25 25M 827 0 0% | 15.5 16.4 0 8 194k 24 0 0% 0.0
nr 0 4601 10 0 25 26M 851 0 0% | 15.5 16.4 0 8 202k 25 0 0% 0.0
nr 0 4601 10 0 25 27M 879 0 0% | 15.3 16.3 0 8 202k 25 0 0% 0.0
nr 0 4601 11 0 25 27M 892 0 0% | 15.3 16.3 0 8 202k 25 0 0% 0.0
nr 0 4601 12 0 25 27M 900 0 0% | 15.4 16.2 0 8 202k 25 0 0% 0.0
```


nr 0 4601 10 0 25 27M 900 0 0%		15.4 16.3 0 8 202k 25 0 0% 0.0
nr 0 4601 11 0 25 25M 811 0 0%		15.5 16.2 0 8 202k 25 0 0% 0.0

5.4.8 Mejora del rendimiento de srsENB

Con el fin de avanzar en el desarrollo y optimización de la implementación, es crucial destacar las consideraciones clave que se tuvieron en cuenta al diseñar e implementar la red 5G SA.

En primer lugar, el rendimiento máximo de srsENB puede verse limitado por diversas razones, como la capacidad de la computadora, la configuración de la red, las características del hardware de radiofrecuencia (RF) y las condiciones físicas de la red.

Con el objetivo de alcanzar el máximo rendimiento, se eligió una computadora equipada con un procesador i7 de 12ª generación y sistema operativo Ubuntu 22.04. Es importante señalar que incluso equipos con especificaciones más modestas pueden ejecutar srsENB de manera exitosa, aunque el rendimiento máximo alcanzable podría ser más limitado.

Adicionalmente, se configuró el regulador de la CPU de la computadora en modo de alto rendimiento con el fin de aprovechar al máximo la potencia de cálculo y el rendimiento. Asimismo, se instaló el kernel de baja latencia de Linux para modificar el flujo de datos y peticiones del sistema operativo. Este ajuste permite que el flujo se realice sin interrupciones y con menor retraso entre la entrada y la salida de la señal, contribuyendo así a la disminución de la latencia.

Dado que se trató de una computadora portátil, fue fundamental mantenerla conectada a una fuente de alimentación durante la ejecución de srsENB para evitar la disminución del rendimiento debido a la adaptación de frecuencia de la CPU en el sistema.

Los pasos detallados para la configuración del kernel de baja latencia y el regulador de la CPU se encuentran en el **Anexo 5**. La adecuada preparación del equipo según las indicaciones de este anexo desempeñó un papel fundamental en la optimización del rendimiento general de la implementación.

5.5 Metodología de las pruebas

Para evaluar el rendimiento, hemos empleado diversas aplicaciones disponibles en la tienda de Google Play. Por ejemplo, utilizamos Speedtest de Ookla, una herramienta que nos permitió cuantificar tanto la velocidad de carga como la de descarga proporcionada por la red. También implementamos la aplicación Zoiper, un softphone que posibilita la realización de llamadas de voz sobre IP (VoIP) a través del protocolo SIP (Session Initiation Protocol). Para

cumplir con esta tarea, fue esencial contar con el software Asterisk, una solución de código abierto que facilita la creación de un servidor de comunicación VoIP basado en SIP. Además, aprovechamos Network Cell Info Lite para llevar a cabo mediciones y diagnósticos de las redes móviles.

En el transcurso de los experimentos, se estableció un flujo de datos único (SISO, por sus iniciales en inglés). También definimos el ancho de banda mediante la especificación del número máximo de bloques de recursos (RB, por sus iniciales en inglés) permitidos. Esta característica fue controlada mediante el uso del software srsRAN, con un límite máximo de 52 RB. Siguiendo la ecuación (1), el ancho de banda (BW) muestra una variación lineal en relación con la cantidad de RB y guarda una correspondencia directa con el espaciado de subportadoras (SCS, por sus iniciales en inglés).

Dadas las limitaciones presentes en las implementaciones de srsRAN, se optó por un valor de 15 kHz para el SCS en la red 5G. En consecuencia, el ancho de banda máximo utilizado se ajustó a 10 MHz.

$$BW = N_{RB} \cdot 12 \cdot SCS \quad (1)$$

Los detalles clave de configuración en la estación base gNB durante la etapa de despliegue y las pruebas de la red se encuentran resumidos en la **Tabla 10**.

Tabla 10. Parámetros configurados en el gNB

Parámetro	Valor
Modo de duplexación	FDD
Ancho de banda	10 MHz
Modo de transmisión	TM1
SCS	15 kHz
No. RBs	52
Flujos	SISO
Banda	3
Frecuencia Downlink	1842.500 MHz
Frecuencia Uplink	1747.500 MHz
Ganancia TX	50 dB
Ganancia RX	40 dB
MCC	001
MNC	01
TAC	7

Fuente: Elaboración propia

6. Resultados

En esta sección, se detallan las pruebas llevadas a cabo en el prototipo de red 5G SA privada implementada, junto con los resultados obtenidos en una variedad de escenarios. También se presentan las distintas herramientas de software utilizadas para analizar la red. Estos experimentos se realizaron en dos laboratorios: el Laboratorio de Antenas y Telecomunicaciones de la Facultad de Energía, Industrias y Recursos Naturales no Renovables, y el Laboratorio de Telecomunicaciones de la Universidad Técnica Particular de Loja. Estos entornos proporcionaron la infraestructura de hardware necesaria para llevar a cabo las pruebas de manera adecuada.

Con el propósito de realizar comparaciones precisas de rendimiento entre diferentes dispositivos de hardware, se eligieron dos escenarios distintos para implementar la red: uno utilizando el dispositivo SDR BladeRF 2.0 micro Ax9 y otro con el USRP B210.

En el entorno práctico, el equipo empleado constaba de un sistema operativo Ubuntu 22.04 LTS con una memoria RAM de 16 GB y un disco duro de 500 GiB. Este sistema estaba equipado con un procesador Intel Core i7 funcionando a 4 GHz en modo performance.

La **Figura 31** presenta el primer entorno utilizado para implementar la configuración 5G SA utilizando el dispositivo USRP B210, mientras que la **Figura 32** representa el segundo entorno empleado para implementar la configuración 5G SA utilizando el dispositivo BladeRF 2.0 micro Ax9.

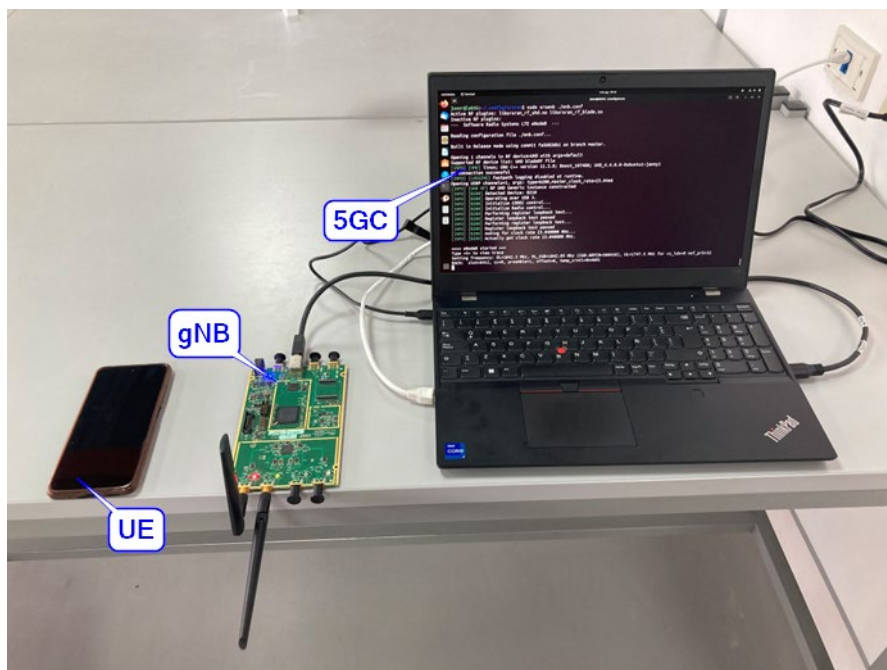


Figura 31. Entorno empleado para el despliegue de 5G SA con USRP B210
Fuente: Elaboración propia



Figura 32. Entorno empleado para el despliegue de 5G SA con BladeRF 2.0 micro Ax9
Fuente: Elaboración propia

Es importante destacar que una vez que la red fue implementada, se identificaron ciertas inestabilidades en su funcionamiento. Uno de los principales desafíos fue que los Dispositivos de Usuario (UE) tenían dificultades para mantener la conectividad. Esto se debió a que los relojes internos integrados en los dispositivos de Radio Definida por Software (SDR) no siempre ofrecen la precisión requerida para operar dentro de los márgenes de tolerancia establecidos por los dispositivos móviles. Para abordar este problema, se utilizó el kit GPSDO para USRP N200/N210, el cual se encontraba disponible en el Laboratorio de Telecomunicaciones de la UTPL. Al combinar este kit con el USRP B210, se logró estabilizar la red de manera efectiva.

Por otro lado, la implementación con el BladeRF 2.0 permitió la conexión exitosa del dispositivo de usuario. Sin embargo, no fue posible mejorar la estabilidad de la red debido a la falta de relojes externos disponibles en el Laboratorio de Antenas y Telecomunicaciones de la UNL.

En consecuencia, en la **Figura 33** se presenta el entorno final en el cual se desplegó la red privada 5G SA. Sobre esta configuración se llevaron a cabo las correspondientes pruebas, cuyos resultados se presentan en las secciones subsiguientes. Estos resultados proporcionan una comprensión más profunda de la eficacia y desempeño de la red implementada.

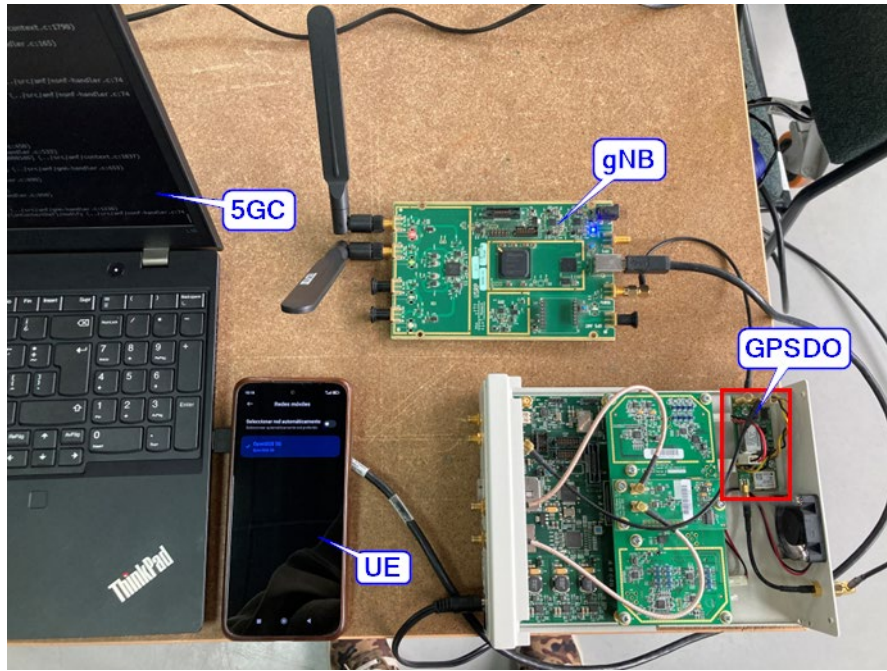


Figura 33. Entorno empleado para el despliegue de 5G SA con USRP B210 y GPSDO
Fuente: Elaboración propia

6.1 Test de datos

6.1.1 Resultados teóricos

Según el documento 3GPP TS 38.306, la ecuación aproximada para calcular el máximo Throughput en 5G es la siguiente:

$$\text{data rate (in Mbps)} = 10^{-6} \cdot \sum_{j=1}^J \left(v_{Layers}^{(j)} \cdot Q_m^{(j)} \cdot f^{(j)} \cdot R_{max} \cdot \frac{N_{PRB}^{BW(j),u}}{T_s^u} \cdot (1 - OH(j)) \right) \quad (2)$$

Donde:

- J es el número de portadoras agregadas en una banda o combinación de bandas.
- $v_{Layers}^{(j)}$ es el número máximo de capas para la j -ésima portadora.
- $Q_m^{(j)}$ es el orden máximo de modulación, que puede ser 2 para QPSK, 4 para 16-QAM, 6 para 64-QAM o 8 para 256-QAM.
- $f^{(j)}$ es el factor de escala, que puede tomar valores de 1/0.8/0.75/0.4.
- R_{max} es un factor que representa la máxima eficiencia de codificación que se puede lograr en una red 5G.
- $N_{PRB}^{BW(j),u}$ es el número máximo de bloques de recursos asignados en el ancho de banda $BW(j)$ con la numerología u .

- T_s^u es la duración promedio de un símbolo OFDM en una subtrama para la numerología u .
- $OH(j)$ es el factor de sobrecarga, que varía según el rango de frecuencia y el sentido del enlace.

Para realizar los cálculos teóricos, se adaptaron los valores de $OH(j)$, estableciendo $OH(j) = 0,14$ para la dirección de enlace descendente (DL) y $OH(j) = 0,08$ para la dirección de enlace ascendente (UL), respectivamente. Además, se aplicaron diferentes esquemas de modulación y codificación para el tráfico UL y DL, definidos a través del esquema MCS (Modulation and Coding Scheme). Como resultado de estas consideraciones, los cálculos de rendimiento, basados en la ecuación (2), han sido desglosados y se presentan en la **Tabla 11**.

Tabla 11. Resultados teóricos

Resultados teóricos						
Streams	SISO					
No. RBs	52					
MCS	5	10	15	20	25	28
Uplink (Mbps)	5.94	10.67	19.33	26.70	38.71	44.64
Downlink (Mbps)	5.56	9.98	18.08	24.96	36.19	41.73

Fuente: Elaboración propia

6.1.2 Resultados experimentales

Con base en las pruebas realizadas en el prototipo, presentamos los resultados obtenidos en términos de rendimiento en el entorno real, los cuales se detallan en la **Tabla 12**. Para la representación de cada valor alcanzado, se tomaron un total de 10 muestras para cada escenario de MCS, y posteriormente se realizó un promedio en cada caso.

Tabla 12. Resultados experimentales de la implementación

Resultados experimentales						
Streams	SISO					
No. RBs	52					
MCS	5	10	15	20	25	28
Uplink (Mbps)	4.6	8.8	10.4	13.1	20.4	25.3
Downlink (Mbps)	4.4	7.5	10.9	15.8	22.8	27.4

Fuente: Elaboración propia

La conectividad de la red lograda a través de esta implementación se ha desarrollado conforme a las expectativas establecidas. Sin embargo, la restricción impuesta al SCS, limitándolo a tan solo 15 kHz, incide en la capacidad de acceso a los datos por parte de los dispositivos conectados. En términos de desempeño, los resultados han resultado estar por debajo de los resultados teóricos, tanto en el tráfico de enlace ascendente (UL) como en el de enlace descendente (DL). Una de las razones subyacentes a este desempeño inferior es que el

factor de sobrecarga asociado al protocolo de señalización en la red 5G resultó ser mayor de lo inicialmente estimado para el cálculo de rendimiento, con valores de $OH(j) = 0,14$ para DL y $OH(j) = 0,08$ para UL. Una representación gráfica de la comparativa del Throughput teórico y experimental se ilustra en la **Figura 34** y **Figura 35**.

Uplink Teórico (Mbps) y Uplink Experimental (Mbps)

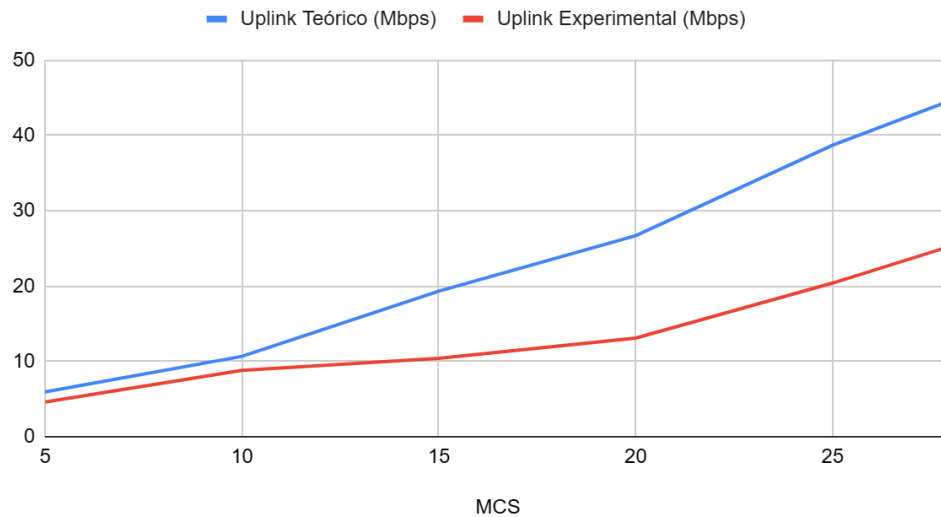


Figura 34. Uplink teórico y Uplink experimental

Fuente: Elaboración propia

En la **Figura 36**, se muestra una de las muestras utilizando un esquema de modulación y codificación (MCS) de 28.

Por otro lado, en la **Figura 37**, se muestra una de las muestras utilizando un esquema de modulación y codificación (MCS) de 5. En el **Anexo 6** se presentan varias muestras adicionales de la medición del Throughput que se tomaron durante las pruebas.

Downlink Teórico (Mbps) y Downlink Experimental (Mbps)

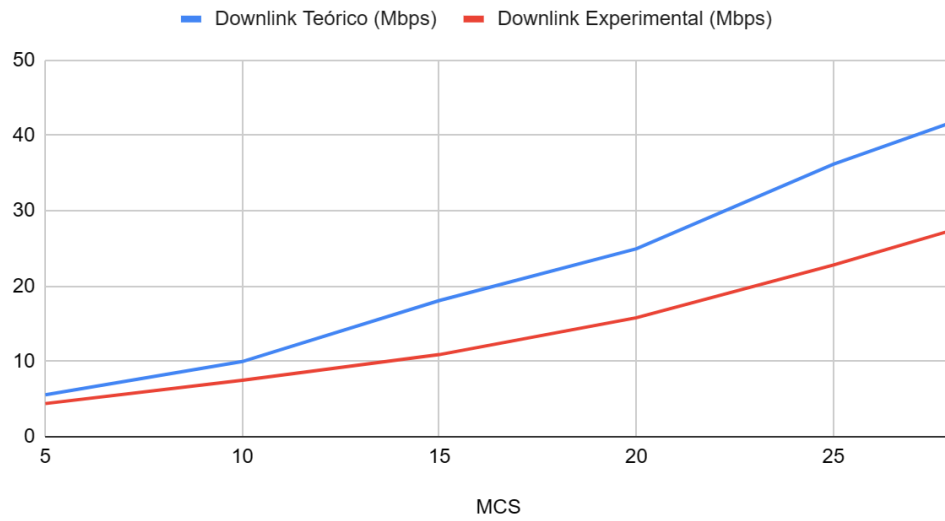


Figura 35. Downlink teórico y Downlink experimental
Fuente: Elaboración propia

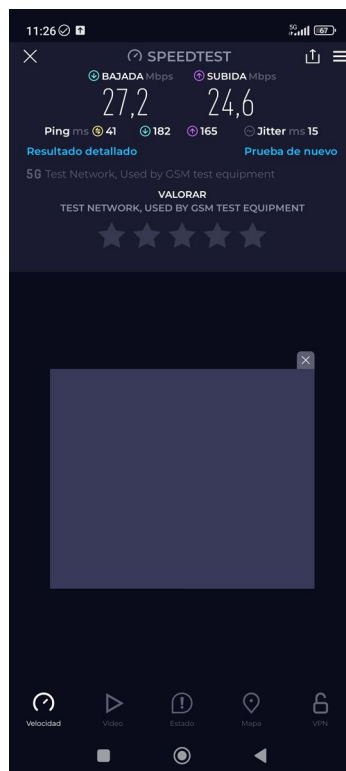


Figura 36. Throughput alcanzado utilizando un MCS de 28
Fuente: Elaboración propia

6.1.3 Acceso a Internet

La evaluación del acceso a Internet abarcó la navegación web, con acceso a diversas páginas, entre las que se incluye el sitio de la Universidad Nacional de Loja, como se aprecia en la **Figura 38**.

Adicionalmente, se llevaron a cabo pruebas en la reproducción de contenidos multimedia, como videos provenientes de plataformas como YouTube. En este contexto, se logró alcanzar la resolución máxima admitida por el dispositivo (1440p), como se ilustra en la **Figura 39**.

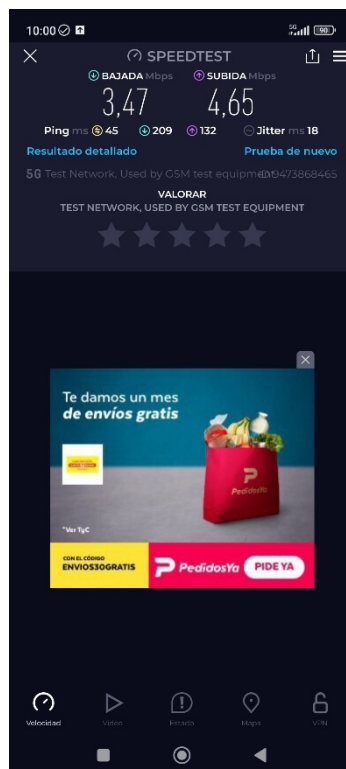


Figura 37. Throughput alcanzado utilizando un MCS de 5
Fuente: Elaboración propia

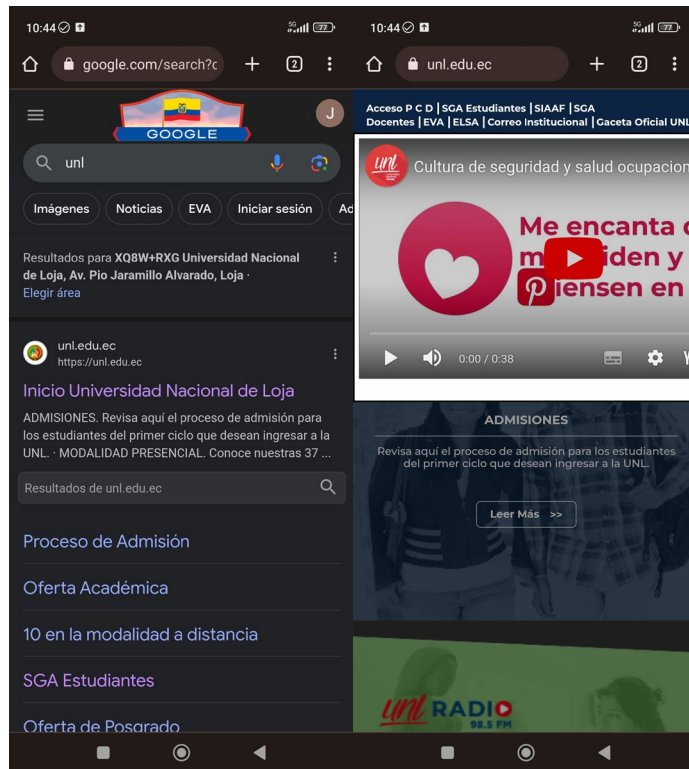


Figura 38. Navegación por la web
Fuente: Elaboración propia

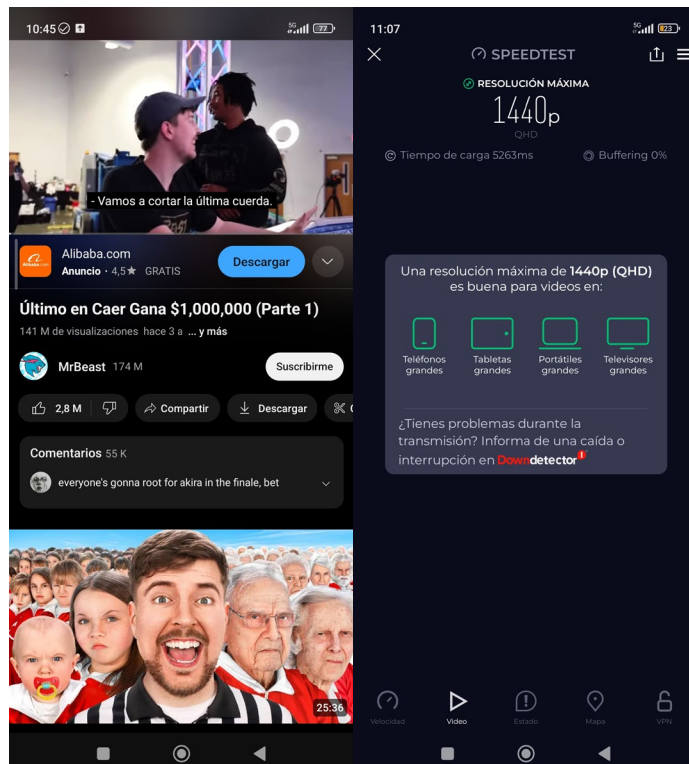


Figura 39. Reproducción de videos
Fuente: Elaboración propia

Asimismo, se llevaron a cabo descargas de aplicaciones utilizando los datos proporcionados por la red, como se puede observar en la **Figura 40**.

En resumen, estas pruebas se llevaron a cabo de manera exitosa, proporcionando un rendimiento óptimo al usuario durante la ejecución de estas actividades a través de la red, sin contratiempos. Esta mejora en la experiencia se atribuye a la velocidad que la red brinda al dispositivo, la cual es adecuada para el aprovechamiento fluido de dichos servicios.

6.1.4 Análisis de tráfico

En esta sección, se emprende un análisis de los mensajes de tráfico intercambiados en el prototipo de red, con el propósito de evaluar su operatividad y detectar posibles fallos inherentes al despliegue.

La **Figura 41** presenta un gráfico de flujo que representa el tráfico intercambiado entre el gNB y el AMF.

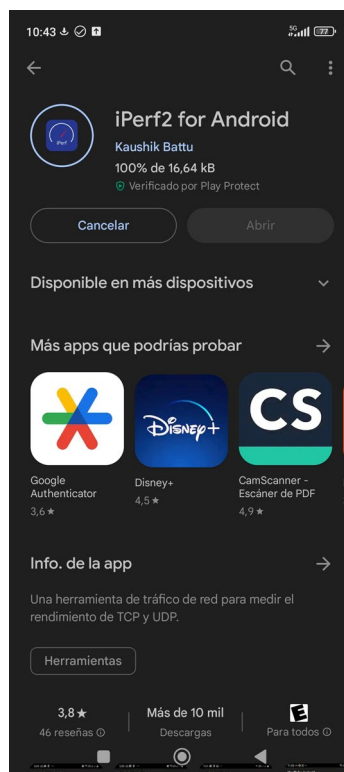


Figura 40. Descarga de aplicaciones
Fuente: Elaboración propia

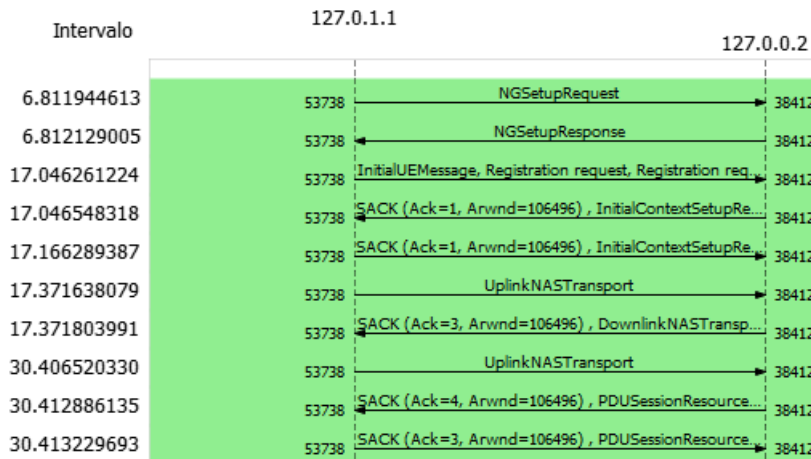


Figura 41. Grafica de flujo del tráfico intercambiado entre el gNB y el AMF
Fuente: Elaboración propia

Para establecer una conexión NG entre el gNB y el AMF en la red 5G SA (Standalone), se envía en primer lugar el mensaje de señalización “NGSetupRequest”. Este mensaje se transmite mediante el protocolo NGAP a través de la interfaz N2, su contenido se lo puede observar en la **Figura 42**, en donde se contempla el identificador único del gNB, formado por el PLMN ID y el gNB ID, así como el RAN Node Name, que corresponde al nombre del software srsENB.

```

  v NGSetupRequest
  v protocolIEs: 4 items
  v Item 0: id-GlobalRANNodeID
  v ProtocolIE-Field
  id: id-GlobalRANNodeID (27)
  criticality: reject (0)
  v value
  v GlobalRANNodeID: globalGNB-ID (0)
  v globalGNB-ID
  v pLMNIdentity: 00f110
  Mobile Country Code (MCC): Unknown (1)
  Mobile Network Code (MNC): Unknown (01)
  v gNB-ID: gNB-ID (0)
  gNB-ID: 00066c [bit length 22, 2 LSB pad bits, 0000 0000 0000 0110 0110 11.. decimal value 411]
  v Item 1: id-RANNodeName
  v ProtocolIE-Field
  id: id-RANNodeName (82)
  criticality: ignore (1)
  v value
  RANNodeName: srsenb01

```

Figura 42. Global RAN Node ID y RAN Node Name
Fuente: Elaboración propia

El AMF responde al mensaje “NGSetupRequest” con un mensaje “NGSetupResponse” para confirmar la conexión entre el gNB y el AMF. En la **Figura 43** se muestran los campos del AMF Name, que corresponde al nombre del software Open5GS, y la PLMN Support List, que indica los PLMN que el AMF puede soportar. En este caso, solo hay un PLMN en la lista, que es el 01001.

```

  ▾ NGSetupResponse
    ▾ protocolIEs: 4 items
      ▾ Item 0: id-AMFName
        ▾ ProtocolIE-Field
          id: id-AMFName (1)
          criticality: reject (0)
          ▾ value
            AMFName: open5gs-amf0
        > Item 1: id-ServedGUAMList
        > Item 2: id-RelativeAMFCapacity
      ▾ Item 3: id-PLMNSupportList
        ▾ ProtocolIE-Field
          id: id-PLMNSupportList (80)
          criticality: reject (0)
          ▾ value
            ▾ PLMNSupportList: 1 item
              ▾ Item 0
                ▾ PLMNSupportItem
                  ▾ pLMNIdentity: 00f110
                    Mobile Country Code (MCC): Unknown (1)
                    Mobile Network Code (MNC): Unknown (01)

```

Figura 43. AMF Name y PLMN Support List
Fuente: Elaboración propia

El gNB envía el mensaje de señalización “InitialUEMessage” al AMF para iniciar el procedimiento de registro del UE en la red 5G SA. Este mensaje se transmite desde el gNB al AMF después de recibir el mensaje “RRCSetupComplete” del UE, que contiene una solicitud de registro. El propósito de este mensaje es informar al AMF de la intención del UE de registrarse en la red 5G y transmitir la solicitud de registro, que contiene información sobre el UE.

Como se muestra en la **Figura 44**, el campo “userLocationInformationNR” tiene los siguientes subcampos:

- **nR-CGI:** el identificador de la celda NR, que consta de un PLMN ID y un NR Cell Identity.
- **tAI:** el identificador de área de seguimiento, que consta de un PLMN ID y un TAC.

```

  ▾ Item 2: id-UserLocationInformation
    ▾ ProtocolIE-Field
      id: id-UserLocationInformation (121)
      criticality: reject (0)
      ▾ value
        ▾ UserLocationInformation: userLocationInformationNR (1)
          ▾ userLocationInformationNR
            ▾ nR-CGI
              ▾ pLMNIdentity: 00f110
                Mobile Country Code (MCC): Unknown (1)
                Mobile Network Code (MNC): Unknown (01)
                nRCellIdentity: 0x000066c001
            ▾ tAI
              ▾ pLMNIdentity: 00f110
                Mobile Country Code (MCC): Unknown (1)
                Mobile Network Code (MNC): Unknown (01)
                tAC: 7 (0x000007)

```

Figura 44. User Location Information
Fuente: Elaboración propia

El AMF envía el mensaje “InitialContextSetupRequest” al gNB para establecer el contexto UE en el gNB. Este contexto incluye la información necesaria para gestionar el acceso y la movilidad del UE en la red, así como para configurar los recursos radio necesarios para la comunicación.

A continuación, el gNB envía el mensaje “InitialContextSetupResponse” al AMF para confirmar el establecimiento del contexto UE en el gNB.

El gNB envía un mensaje “UplinkNASTransport” para transportar un paquete de datos NAS desde el UE al AMF. El propósito de este mensaje es permitir al UE enviar mensajes NAS al AMF para realizar diversas funciones, como la autenticación, la actualización de ubicación, la gestión de sesión, la gestión de movilidad, la configuración de seguridad y el control de políticas.

El AMF recibe el mensaje “UplinkNASTransport” y lo procesa para realizar las funciones correspondientes al mensaje NAS recibido. Después, el AMF responde con un mensaje “DownlinkNASTransport” para transportar un paquete de datos NAS desde el AMF al UE.

El subcampo “Security protected 5GS NAS message” del campo “NAS-PDU” se muestra en la **Figura 45**. Este subcampo contiene un mensaje NAS 5GS protegido por seguridad, que puede estar cifrado y/o protegido por integridad. Es decir, todos los mensajes NAS han sido cifrados desde el proceso de establecimiento de una conexión RRC.

```

  v Item 2: id-NAS-PDU
    v ProtocolIE-Field
      id: id-NAS-PDU (38)
      criticality: reject (0)
      v value
        v NAS-PDU: 7e02f37d47954a7e0043
          v Non-Access-Stratum 5GS (NAS)PDU
            v Security protected NAS 5GS message
              Extended protocol discriminator: 5G mobility management messages (126)
              0000 .... = Spare Half Octet: 0
              .... 0010 = Security header type: Integrity protected and ciphered (2)
              Message authentication code: 0xf37d4795
              Sequence number: 74
              Encrypted data

```

Figura 45. Security protected 5GS NAS message

Fuente: Elaboración propia

El AMF envía un mensaje “PDUSessionResourceSetupRequest” al gNB para solicitar que establezca los recursos de sesión PDU para el UE. El propósito de este mensaje es proporcionar al gNB los parámetros necesarios para configurar los recursos de radio y los portadores de servicio para el UE.

El mensaje también incluye la información de túnel GTP entre el UPF y el gNB para el transporte de datos del usuario en el campo “PDUSessionResourceSetupListSReq”. Esta información contiene la dirección IP del UPF que está sirviendo al contexto creado, así como el tipo de protocolo empleado en la conexión, tal y como se aprecia en la **Figura 46**.

```

  Item 1: id-UL-NGU-UP-TNLInformation
  ProtocolIE-Field
  id: id-UL-NGU-UP-TNLInformation (139)
  criticality: reject (0)
  value
  UPTransportLayerInformation: gTPTunnel (0)
  gTPTunnel
  transportLayerAddress: 7f000002 [bit length 32, 0111 1111 0000 0000 0000 0000 0000 0010 decimal value 2130706434]
  transportLayerAddress (IPv4): 127.0.0.2
  gTP-TEID: 00004f36
Item 2: id-PDUSessionType
ProtocolIE-Field
id: id-PDUSessionType (134)
criticality: reject (0)
value
PDUSessionType: ipv4 (0)

```

Figura 46. Dirección IP del UPF asociado al contexto y tipo de protocolo empleado en la conexión PDN

Fuente: Elaboración propia

El gNB envía un mensaje “PDUSessionResourceSetupResponse” al AMF para confirmar el establecimiento o la modificación de los recursos del plano de usuario para una sesión PDU entre el UE y la UPF. El propósito de este mensaje es informar al AMF del resultado del establecimiento o la modificación de los recursos solicitados por el AMF en el mensaje “PDUSessionResourceSetupRequest”.

Análisis de mensajes HTTP con la finalidad de comprobación de salida a internet

En el análisis de la captura, sobresalen los mensajes HTTP dirigidos al host “connectivitycheck.gstatic.com”, los cuales son enviados desde el dispositivo terminal hacia Internet, tal y como se muestra en la **Figura 47**. Estos mensajes cumplen la función de verificar la disponibilidad de conexión a la red.

No.	Time	Source	Destination	Protocol	Length	Info
9201	51.948623262	10.45.0.21	142.250.189.131	GTP <HTTP>	331	GET /generate_204 HTTP/1.1
9214	51.948665678	172.17.182.8	142.250.189.131	HTTP	295	GET /generate_204 HTTP/1.1

Figura 47. Mensajes HTTP para comprobar si hay salida a internet

Fuente: Elaboración propia

Estos mensajes han sido capturados en la interfaz N3, que establece la conexión entre la red de acceso y el UPF, presentado en este despliegue a través de una interfaz loopback. Conforme se aprecia en la **Figura 48**, la interfaz loopback con dirección 127.0.1.1 inicia el proceso enviando el primer mensaje de consulta al UPF, cuya dirección es 127.0.0.2.

Luego, el UPF encamina el paquete hacia la interfaz N6 para su salida a internet. Este proceso da lugar al segundo paquete, el cual ya no está encapsulado en el túnel GTP, dado que ha alcanzado el UPF.

Para lograr que el paquete salga por la interfaz N6, previamente se ha llevado a cabo un proceso de traducción de direcciones (NAT), que implicó el cambio de la dirección de origen 10.45.0.21, asociada al UE. En lugar de esta dirección, se empleó la nueva dirección de origen 172.17.182.8, correspondiente a la interfaz de salida a internet del equipo en el que se ejecuta el núcleo.

```
> Frame 9201: 331 bytes on wire (2648 bits), 331 bytes captured (2648 bits) on interface any, id 0
> Linux cooked capture v1
> Internet Protocol Version 4, Src: 127.0.1.1, Dst: 127.0.0.2
> User Datagram Protocol, Src Port: 2152, Dst Port: 2152
> GPRS Tunneling Protocol
> Internet Protocol Version 4, Src: 10.45.0.21, Dst: 142.250.189.131
> Transmission Control Protocol, Src Port: 57890, Dst Port: 80, Seq: 1, Ack: 1, Len: 227
< Hypertext Transfer Protocol
  > GET /generate_204 HTTP/1.1\r\n
    Connection: close\r\n
    User-Agent: Mozilla/5.0 (X11; Linux x86_64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/60.0.3112.32 Safari/537.36\r\n
    Host: connectivitycheck.gstatic.com\r\n
    Accept-Encoding: gzip\r\n
    \r\n
    [Full request URI: http://connectivitycheck.gstatic.com/generate_204]
    [HTTP request 1/1]
    [Response in frame: 9379]
```

Figura 48. Cuerpo de la petición GTP<HTTP> para la comprobación de salida a internet
Fuente: Elaboración propia

A continuación, en la **Figura 49**, se presentan diversos mensajes correspondientes al tráfico que transita entre el UE e Internet, los cuales se encuentran encapsulados bajo el protocolo GTP.

No.	Time	Source	Destination	Protocol	Length	Info
9201	51.948623262	10.45.0.21	142.250.189.131	GTP <HTTP>	331	GET /generate_204 HTTP/1.1
17990	64.148659046	10.45.0.21	45.161.33.194	GTP <TCP>	145	46708 → 8080 [PSH, ACK] Seq=75 Ack=78 Win=81920 Len=41 TSval=1583053552 TSecr=3891807572
18195	64.228695199	10.45.0.21	45.161.33.194	GTP <TCP>	147	46712 → 8080 [PSH, ACK] Seq=107 Ack=125 Win=81920 Len=43 TSval=1583053602 TSecr=3891807609
68919	69.318306934	10.45.0.21	45.161.33.194	GTP <TCP>	200	46712 → 8080 [PSH, ACK] Seq=4648 Ack=4584 Win=81920 Len=96 TSval=1583058650 TSecr=3891812658
82216	70.678214727	10.45.0.21	45.161.33.194	GTP <TCP>	160	46712 → 8080 [PSH, ACK] Seq=5983 Ack=5817 Win=81920 Len=56 TSval=1583059991 TSecr=3891814009
1441.	79.558002992	10.45.0.21	45.161.33.194	GTP <TCP>	1452	46720 → 8080 [ACK] Seq=123 Ack=69 Win=81920 Len=1348 TSval=1583068949 TSecr=3891822970
1441.	79.558007800	10.45.0.21	45.161.33.194	GTP <TCP>	1452	46720 → 8080 [ACK] Seq=1471 Ack=69 Win=81920 Len=1348 TSval=1583068949 TSecr=3891822970
1441.	79.559846281	10.45.0.21	45.161.33.194	GTP <TCP>	154	46730 → 8080 [PSH, ACK] Seq=59 Ack=70 Win=81920 Len=50 TSval=1583068949 TSecr=3891822969
1441.	79.560741415	10.45.0.21	45.161.33.194	GTP <TCP>	1452	46726 → 8080 [ACK] Seq=10897 Ack=70 Win=81920 Len=1348 TSval=1583068949 TSecr=3891822969
1442.	79.562003200	10.45.0.21	45.161.33.194	GTP <TCP>	1452	46722 → 8080 [ACK] Seq=1461 Ack=69 Win=81920 Len=1348 TSval=1583068956 TSecr=3891822977
1443.	79.628701076	10.45.0.21	45.161.33.194	GTP <TCP>	1452	46726 → 8080 [ACK] Seq=12245 Ack=70 Win=81920 Len=1348 TSval=1583068989 TSecr=3891823009
1443.	79.628715926	10.45.0.21	45.161.33.194	GTP <TCP>	1452	46726 → 8080 [ACK] Seq=13593 Ack=70 Win=81920 Len=1348 TSval=1583068989 TSecr=3891823009
1443.	79.637848863	10.45.0.21	45.161.33.194	GTP <TCP>	1452	46726 → 8080 [ACK] Seq=18985 Ack=70 Win=81920 Len=1348 TSval=1583068989 TSecr=3891823009
1443.	79.637855351	10.45.0.21	45.161.33.194	GTP <TCP>	146	46728 → 8080 [PSH, ACK] Seq=128 Ack=140 Win=81920 Len=42 TSval=1583068990 TSecr=3891823009
1444.	79.643701786	10.45.0.21	45.161.33.194	GTP <TCP>	1452	46726 → 8080 [ACK] Seq=31117 Ack=70 Win=81920 Len=1348 TSval=1583068990 TSecr=3891823019
1444.	79.647839657	10.45.0.21	45.161.33.194	GTP <TCP>	1452	46720 → 8080 [ACK] Seq=33823 Ack=69 Win=81920 Len=1348 TSval=1583069001 TSecr=3891823023
1444.	79.647858290	10.45.0.21	45.161.33.194	GTP <TCP>	1452	46720 → 8080 [ACK] Seq=37867 Ack=69 Win=81920 Len=1348 TSval=1583069001 TSecr=3891823023
1444.	79.650779640	10.45.0.21	45.161.33.194	GTP <TCP>	1452	46722 → 8080 [ACK] Seq=17637 Ack=69 Win=81920 Len=1348 TSval=1583069002 TSecr=3891823023
1445.	79.654786622	10.45.0.21	45.161.33.194	GTP <TCP>	1452	46724 → 8080 [ACK] Seq=27086 Ack=69 Win=81920 Len=1348 TSval=1583069007 TSecr=3891823028
1445.	79.655857680	10.45.0.21	45.161.33.194	GTP <TCP>	1452	46724 → 8080 [ACK] Seq=33826 Ack=69 Win=81920 Len=1348 TSval=1583069007 TSecr=3891823028
1445.	79.657710855	10.45.0.21	45.161.33.194	GTP <TCP>	1452	46722 → 8080 [ACK] Seq=27073 Ack=69 Win=81920 Len=1348 TSval=1583069008 TSecr=3891823029
1445.	79.659758265	10.45.0.21	45.161.33.194	GTP <TCP>	1452	46722 → 8080 [ACK] Seq=35161 Ack=69 Win=81920 Len=1348 TSval=1583069008 TSecr=3891823029
1445.	79.659767510	10.45.0.21	45.161.33.194	GTP <TCP>	1452	46722 → 8080 [ACK] Seq=36509 Ack=69 Win=81920 Len=1348 TSval=1583069008 TSecr=3891823029
1446.	79.671741487	10.45.0.21	45.161.33.194	GTP <TCP>	1452	46726 → 8080 [ACK] Seq=54033 Ack=70 Win=81920 Len=1348 TSval=1583069076 TSecr=3891823097
1446.	79.673929069	10.45.0.21	45.161.33.194	GTP <TCP>	1452	46726 → 8080 [ACK] Seq=63469 Ack=70 Win=81920 Len=1348 TSval=1583069077 TSecr=3891823098
1446.	79.674804581	10.45.0.21	45.161.33.194	GTP <TCP>	1452	46726 → 8080 [ACK] Seq=67513 Ack=70 Win=81920 Len=1348 TSval=1583069081 TSecr=3891823102

Figura 49. Paquetes intercambiados entre el UE e internet empleando el protocolo GTP
Fuente: Elaboración propia

6.2 Test de VoIP

Para llevar a cabo esta prueba, se estableció una llamada de voz entre dos clientes SIP utilizando un servidor SIP como Asterisk (instalado en la laptop) y la aplicación Zoiper instalada en los clientes SIP. La configuración implicó la modificación de dos archivos fundamentales en Asterisk, que son:

- **Sip.conf:** Este archivo posibilita la inclusión de cuentas/usuarios SIP que están asociados con las extensiones (101, 102).
- **Extensions.conf:** En este archivo se define el dialplan o plan de numeración que la centralita seguirá para cada contexto y, por lo tanto, para cada usuario. En otras palabras, este archivo define el comportamiento que adoptará el servidor en una llamada.

La **Tabla 13** presenta los clientes SIP que se utilizaron para establecer la comunicación VoIP a través de la red, junto con sus extensiones y nombres de usuario correspondientes.

Tabla 13. Usuarios registrados al servidor SIP de Asterisk

Usuario	Extensión	Dispositivo
User1	101	Xiaomi Poco M3 Pro 5G
User2	102	Lenovo Thinkpad L15 gen 2

Fuente: Elaboración propia

Asterisk posee soporte de extensión con IP, esto permite que los usuarios que este dentro de una red pueda comunicarse a través del servidor. En este caso la función de red SMF es la que crea la red 10.45.0.0/16 y asigna direcciones IP a los usuarios. La dirección IP a través de la cual los usuarios podrán registrarse al servidor SIP de Asterisk es 10.45.0.1, dirección IP que el SMF asigna a la laptop.

Una vez registrados, procedimos a utilizar el siguiente comando para iniciar la interfaz de línea de comandos (CLI) de Asterisk, estableciendo un nivel de verbosidad elevado (vvv). Esto conlleva la visualización de mensajes de depuración y estado más detallados en la consola.

```
asterisk -rvvv
```

Luego, empleamos el siguiente comando para verificar la información relativa a los dispositivos SIP registrados en Asterisk, que abarca su nombre, dirección IP, estado y calidad de la señal.

```
sip show peers
```

Finalmente, con el siguiente comando, exhibimos información concerniente a los usuarios SIP definidos en el archivo sip.conf de Asterisk, incluyendo su nombre, contexto y clave secreta.

```
sip show users
```

La **Figura 50** ilustra la llamada realizada desde el cliente SIP Xiaomi Poco M3 Pro 5G (ext101) hacia el cliente SIP Lenovo Thinkpad L15 Gen2 (ext102). Adicionalmente, se presentan las estadísticas de la red durante el transcurso de la llamada en la **Figura 51**. En estas

estadísticas, se observa que el jitter y la pérdida de paquetes se mantuvieron en un nivel de 0%, lo que contribuyó a evitar distorsiones, silencios o ruidos en la llamada.

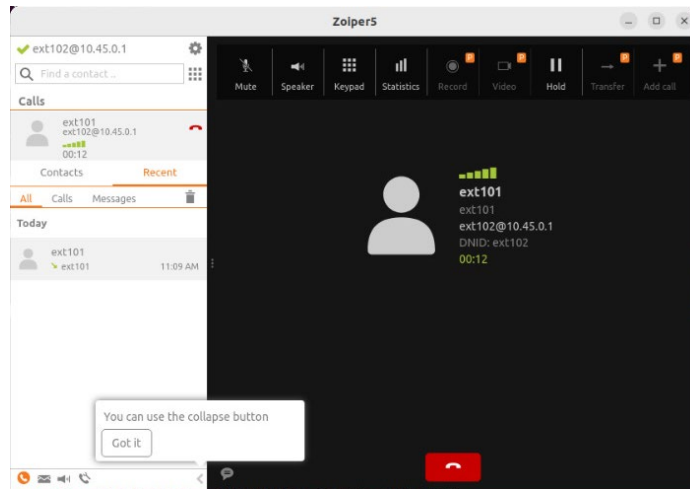


Figura 50. Llamada VoIP recibida desde el cliente SIP Xiaomi Poco M3 Pro 5G
Fuente: Elaboración propia

Además, en la **Figura 52** se presenta la llamada en sentido inverso, es decir, desde el cliente SIP Lenovo Thinkpad L15 Gen2 hacia el cliente SIP Xiaomi Poco M3 Pro 5G (ext101).

También se llevó a cabo una prueba de MOS (Mean Opinion Score) para evaluar la calidad percibida de la llamada VoIP. El MOS se traduce en una escala numérica que abarca del 1 al 5, donde 1 denota una calidad pésima y 5 representa una calidad excelente, como se detalla en la **Tabla 14**.

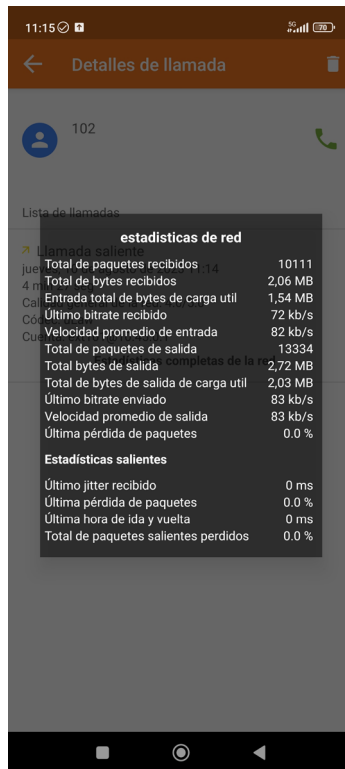


Figura 51. Estadísticas de red entre user1 (ext101) y user2 (ext102)
Fuente: Elaboración propia

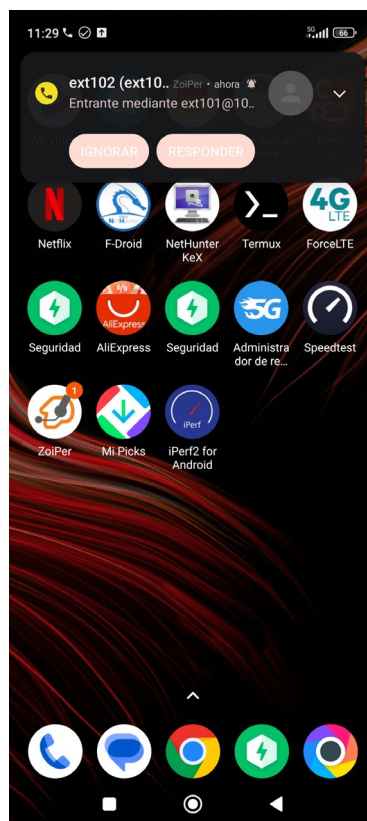


Figura 52. Llamada VoIP recibida desde el cliente SIP Lenovo Thinkpad L15 Gen2
Fuente: Elaboración propia

Tabla 14. MOS (Mean Opinion Score)

MOS	Calidad	Deficiencia
5	Excelente	Imperceptible
4	Buena	Perceptible pero no molesto
3	Media	A simple vista molesto
2	Pobre	Molesto
1	Mala	Muy molesto

Fuente: Elaboración propia

Para este caso, se optó por emplear el método objetivo, el cual se fundamenta en algoritmos matemáticos que estiman el MOS a partir de parámetros técnicos de la transmisión, tales como el ancho de banda, la latencia, el jitter y la pérdida de paquetes. En este contexto, se utilizaron los resultados proporcionados por el software Zoiper basados en estos parámetros técnicos.

Los resultados de la **Figura 53** indican un MOS de 5.0, reflejando una calidad excelente en la red durante las llamadas VoIP entre los clientes SIP. Este puntaje destaca la experiencia satisfactoria y fluida de los usuarios, subrayando así la eficacia y el rendimiento óptimo de la infraestructura de la red.

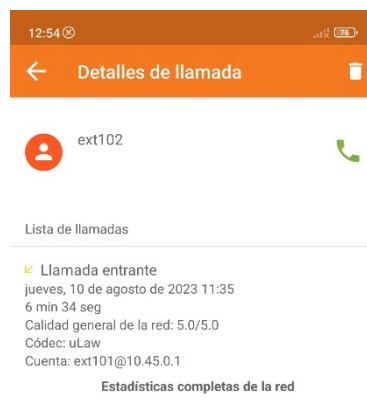


Figura 53. Resultados estimados del MOS
Fuente: Elaboración propia

6.3 Test de potencia: RX

Adicionalmente se realizaron pruebas de medición de la potencia promedio de las señales piloto recibidas desde la estación base. Para realizar estas pruebas se empleó la aplicación Network Cell Info Lite (NCIL) en el UE conectado a la red. Esta aplicación consta de herramientas de medición y diagnóstico de redes móviles y wifi, lo cual permite observar la banda NR utilizada, la Potencia de Señal Recibida de Referencia (Reference Signal Received Power), la Relación Señal a Interferencia más Ruido (Signal to Interference plus Noise Ratio) y la Calidad de Señal Recibida de Referencia (Reference Signal Received Quality).

Los resultados de las mediciones desde el dispositivo de usuario a distancias de 0 metros y 8 metros de la gNB se presentan en la **Figura 54**. A corta distancia de la gNB, se obtuvo un RSRP de -67 dBm, indicando una señal robusta. Además, un SINR de 22 dB refleja una excelente calidad de señal, mientras que un RSRQ de -5 dB sugiere una calidad sobresaliente.



Figura 54. Potencias promedio recibida a 0 metros de distancia de la gNB

Fuente: Elaboración propia

En contraste, al colocar el dispositivo a 8 metros de la gNB, los resultados experimentan cambios significativos, tal y como se observa en la **Figura 55**. En este escenario, el RSRP disminuye a -116 dBm, indicando una señal muy débil. Además, un SINR de 9 dB apunta a una calidad de señal pobre, y un RSRQ de -10 dB refleja una señal deficiente. Adicionalmente, la

Tabla 15 resume los resultados de RSRP, SINR y RSRQ en relación con la distancia desde la gNB.

En el **Anexo 7** se presentan varias muestras adicionales de la medición de la potencia RX que se tomaron durante las pruebas.



Figura 55. Potencias promedio recibida a 8 metros de distancia de la gNB
Fuente: Elaboración propia

Tabla 15. Resultados de RSRP, SINR, RSRQ en función de la distancia del gNB

Distancia (m)	RSRP (dBm)	SINR (dB)	RSRQ (dB)
0	-66	24.0	-5
1	-73	22.0	-5
2	-94	18.0	-5
3	-107	12.0	-7
4	-109	12.0	-8
5	-112	10.0	-8
6	-112	9.0	-10
7	-113	9.0	-12
8	-115	7.0	-12

Fuente: Elaboración propia

Estos hallazgos subrayan la importancia de la proximidad al nodo base en la calidad de la señal y el rendimiento general en una red 5G. Según estos resultados, la red pudo manejar efectivamente servicios de datos y VoIP en la mayoría de las distancias medidas. Sin embargo, se observó una disminución en la calidad del servicio a distancias más largas (por ejemplo, 8 metros y más) debido a la reducción de la RSRP y la SINR.

6.4 Análisis de frecuencia y ancho de banda

Con el fin de llevar a cabo un análisis de la frecuencia y el ancho de banda, se empleó el analizador de espectro de la Universidad Técnica Particular de Loja. Este instrumento evalúa y escanea las frecuencias en el rango desde 9 kHz hasta 7.5 GHz. Se utilizó el analizador de espectro CXA Signal Analyzer N9000A de la marca Keysight, en conjunto con el software Keysight Spectrum Analyzer.

La **Figura 56** representa la banda de frecuencia Downlink utilizada por el gNB a una distancia de 1 metro del UE. Al examinar la figura, se evidencia que la red opera en la frecuencia de 1842.500 MHz (1.8425 GHz), previamente configurada. Además, se destaca un ancho de banda de 10 MHz, el máximo permitido por srsRAN.

En la misma gráfica, se muestra la potencia recibida por las antenas del esquema mencionado, con un valor de -81 dBm. Su potencia es notablemente superior a la de los equipos de usuario, gracias a una mejor ganancia en la antena.

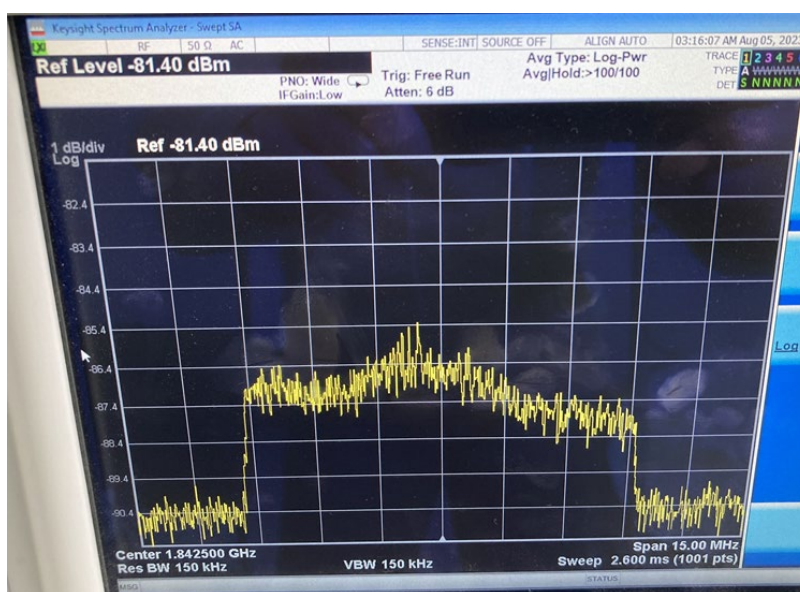


Figura 56. Banda de frecuencia Downlink del gNB
Fuente: Elaboración propia

Este análisis de frecuencia y ancho de banda ofrece una visión detallada de la configuración de la red en términos de frecuencia y capacidad de transmisión. Este enfoque se alinea con las definiciones establecidas a través del sistema y se revela como esencial para asegurar un rendimiento óptimo en la comunicación y conectividad de los dispositivos de usuarios.

6.5 Análisis de Costos

Este capítulo proporciona una estimación de los costos asociados con el desarrollo del proyecto. El análisis incluye los recursos de hardware y software utilizados para implementar el prototipo de red 5G SA. A continuación, se presenta un desglose del presupuesto:

- **Lenovo ThinkPad L15 Gen 2:** Este equipo, valorado en aproximadamente \$1.459, fue proporcionado por la Dirección de Tecnologías de Información de la Universidad Nacional de Loja para el desarrollo del proyecto.
- **BladeRF 2.0 micro xA9:** Este dispositivo, con un costo de alrededor de \$880, ya estaba disponible en el laboratorio de Antenas y Telecomunicaciones de la Universidad Nacional de Loja.
- **USRP B210:** Este equipo, valorado en aproximadamente \$2.310, fue proporcionado por el laboratorio de Telecomunicaciones de la Universidad Técnica Particular de Loja para el desarrollo del proyecto.
- **Antena de banda ancha:** Esta antena, utilizada para la emisión por radiofrecuencia con frecuencias de 1710 MHz - 2170 MHz, tiene un costo de \$10 y ya estaba disponible en los laboratorios de la UNL y UTPL.
- **Dispositivo terminal Xiaomi Poco M3 Pro 5G:** Este dispositivo, valorado en \$120, fue prestado para la realización de este proyecto.
- **Tarjetas SIM sysmoISIM-SJA2 compatibles con 5G:** Estas tarjetas tienen un costo de \$88,07 ya que se venden en paquetes de 10 unidades.
- **CardMan 3121 USB CCID interface:** Esta interfaz, utilizada para mejorar la lectura de las tarjetas SIM, tiene un costo de \$38,85.
- **USRP GPS-Disciplined Oscillator Kit:** Este kit, valorado en \$1.635, ya estaba disponible en el laboratorio de Telecomunicaciones de la UTPL.
- **Antena GPS activa de 3 voltios:** Esta antena, valorada en \$115, ya estaba disponible en el laboratorio de Telecomunicaciones de la UTPL.
- **srsRAN, Open5GS, Ubuntu 22.04 LTS, Wireshark, Pysim, Zoiper:** Estos son proyectos de código abierto gratuitos y, por lo tanto, no tienen costo asociado.

Todos estos elementos se encuentran descritos en detalle en la **Tabla 16**.

Tabla 16. Estimación de costos del prototipo

Concepto	Descripción	Coste Unitario	Unidad de Medida	Cantidad	Total
Hardware	Lenovo ThinkPad L15 Gen 2	\$1.459	N/A	1	\$ 1.459
	BladeRF 2.0 micro xA9	\$880	N/A	1	\$ 880
	USRP B210	\$2.310	N/A	1	\$ 2.310
	Antena de banda ancha	\$10	N/A	1	\$ 10
	Xiaomi Poco M3 Pro 5G	\$130	N/A	1	\$ 130
	sysmoISIM-SJA2	\$88,07	N/A	1	\$ 88,07
	CardMan 3121 USB CCID interface	\$38,85	N/A	1	\$ 38,85
	USRP GPS-Disciplined Oscillator Kit	\$1.635	N/A	1	\$ 1.635
	Antena GPS activa de 3 voltios	\$115	N/A	1	\$ 115
Software	srsRAN	\$0	N/A	1	\$0
	Open5GS	\$0	N/A	1	\$0
	Ubuntu 22.04 LTS	\$0	N/A	1	\$0
	Wireshark	\$0	N/A	1	\$0
	pysim	\$0	N/A	1	\$0
	Zoiper	\$0	N/A	1	\$0
TOTAL: \$6.665,92					

Fuente: Elaboración propia

7. Discusión

Durante la implementación del prototipo de red privada 5G SA, se requirió ajustar los parámetros del software para lograr una mayor estabilidad en el rendimiento. Se encontró que al realizar modificaciones en los parámetros de transmisión (TX) y recepción (RX), se pudieron resolver los problemas relacionados con la incapacidad de los Dispositivos de Usuario (UE) para conectarse de manera efectiva a la red. Esta situación se debió a que la intensidad de la señal de radiofrecuencia (RF) se ve influenciada por varias condiciones físicas intrínsecas en cada caso de uso y configuración específica. En consecuencia, es importante destacar que la configuración adoptada en este proyecto de tesis podría no ser óptima para todos los usuarios, potencialmente requiriendo ajustes adaptativos según las circunstancias particulares. En este escenario particular, se observó que la reducción de la ganancia en las etapas de transmisión y recepción (TX y RX) permitió a los dispositivos lograr un proceso de "attach" exitoso.

Otro parámetro crítico durante el despliegue de la red fue el esquema de Modulación y Codificación (MCS). Aunque el software srsRAN tiene por defecto la configuración con el valor máximo de MCS 28 tanto para la dirección descendente (DL) como ascendente (UL), utilizando un esquema de modulación 64QAM para alcanzar la máxima tasa de transferencia de datos, se reconoció que este valor podría resultar excesivamente elevado en función de las condiciones de la señal de radiofrecuencia (RF). Tal elección podría afectar la capacidad del usuario de mantener una conexión estable. En este contexto, se encontró que la elección de un valor de MCS más bajo contribuyó significativamente a la estabilidad general de la red.

En las primeras etapas de las pruebas, se enfrentaron dificultades en relación con la habilidad del dispositivo móvil para mantener una conexión sostenida. A pesar de que se habían definido múltiples canales de acceso aleatorio (RACH) para permitir al dispositivo solicitar recursos a la red o iniciar una comunicación, no se lograba establecer un "attach" con la red. La causa subyacente radicaba en la falta de precisión del reloj en la interfaz de Radio Definida por Software (SDR), lo que resultaba en una falta de sincronización entre los Dispositivos de Usuario (UE) y la Estación Base 5G (gNB). Para superar esta problemática, se determinó que la implementación de una referencia externa de 10 MHz sería la solución idónea para este inconveniente, logrando así resolver la falta de sincronización. Sin embargo, es importante mencionar que debido a la limitada disponibilidad de relojes de referencia externos en la Universidad Técnica Particular de Loja, esta solución únicamente pudo ser implementada en el dispositivo USRP B210.

Inicialmente, se optó por desplegar la red 5G SA sobre una máquina virtual. No obstante, se identificó que debido a la virtualización de los recursos de hardware del host

principal, no se podía alcanzar el rendimiento máximo posible. Por esta razón, se tomó la decisión de migrar el despliegue de la red a una PC con sistema operativo Ubuntu, donde se configuró el regulador del CPU en modo de rendimiento. Esto permitió garantizar la utilización máxima de capacidad de cómputo y rendimiento del sistema.

Dado que, hasta el momento, el software srsRAN no dispone de compatibilidad con Voice over New Radio (VoNR), se tomó la decisión de implementar una centralita IP que posibilitara la realización de llamadas de voz mediante el protocolo de Voz sobre Protocolo de Internet (VoIP). Esta estrategia no solo permitió evaluar el rendimiento de la red con la inclusión de este servicio adicional, sino que también contribuyó a la consecución de los objetivos primordiales definidos en este proyecto de titulación.

8. Conclusiones

Tras la realización de las distintas fases del presente proyecto, se considera que se han alcanzado los objetivos de manera plenamente satisfactoria, obteniendo las siguientes conclusiones:

Se ha logrado cumplir con éxito el objetivo de concebir e implementar un prototipo de arquitectura 5G-NR enfocado en la provisión de servicios de voz y datos. Para llevar a cabo este logro, se emplearon dispositivos de Radio Definida por Software (SDR) como el USRP B210 y el BladeRF 2.0 para la transmisión aérea. Además, se utilizó una combinación de software de código abierto, como srsRAN y Open 5GS, para implementar tanto la Red de Acceso por Radio (RAN) como el Núcleo de la Red 5G (5GC), permitiendo así la creación y establecimiento de una red privada 5G autónoma (5G SA) completamente funcional de extremo a extremo.

Hemos llevado a cabo un profundo estudio de las diversas opciones de equipos de Radio Definida por Software (SDR) disponibles y soluciones de software de código abierto que permiten implementar soluciones para el despliegue de redes 5G NR. Durante este proceso, hemos explorado y analizado sus características, capacidades y limitaciones. Este estudio nos ha proporcionado una base sólida para la selección de los componentes adecuados en la implementación de nuestra arquitectura 5G NR.

Este enfoque integrado de hardware SDR y software de código abierto posibilitó no solo la demostración de la viabilidad de la red 5G SA, sino también la validación de la interoperabilidad entre los componentes de la RAN y el 5GC. El uso de hardware SDR facilitó la transmisión y recepción eficiente de señales 5G sobre el aire, mientras que las soluciones de software garantizaron la gestión y el control efectivo de la red.

Las pruebas de los servicios de voz y datos, empleando equipos de usuario (UE) comerciales, han sido fundamentales para evaluar el desempeño de la red 5G NR. Hemos obtenido valiosos datos sobre la capacidad de la red para soportar la carga de tráfico, mantener una conectividad estable y brindar servicios de alta calidad. Estas pruebas han validado la robustez y eficacia de nuestra arquitectura implementada, para este tipo de servicios, proporcionando información esencial para su futura optimización y expansión.

Se ha constatado que el software de código abierto disponible para despliegues de redes 5G en la actualidad presenta limitaciones significativas en comparación con las redes comerciales. Esta situación se debe a que el software todavía está en fase de desarrollo y está en proceso de incorporar las nuevas características conforme al estándar 3GPP en evolución.

Se ha logrado facilitar la creación de un entorno completamente utilizable para propósitos educativos. En el futuro, los estudiantes tendrán la capacidad de analizar este escenario como una situación realista, eliminando la necesidad de simular cualquier parte de la arquitectura, un proceso que ha sido común hasta la fecha. Esto permitirá a los estudiantes adentrarse en situaciones prácticas y reales, fomentando un aprendizaje más profundo y enriquecedor.

9. Recomendaciones

Algunos dispositivos de Usuario (UE) pueden no detectar las redes configuradas con valores de MCC y MNC de prueba. Por ejemplo, los valores MCC/MNC comúnmente empleados, como 999/70, 901/70 o 001/01, pueden no ser funcionales, especialmente en dispositivos iPhone equipados con chipsets de banda base de Intel. En lugar de ello, se aconseja considerar la posibilidad de configurar el MCC de la red con el valor específico de su país (por ejemplo, 740 para Ecuador). Esto podría incrementar la probabilidad de que los dispositivos detecten correctamente la red y mejoren la conectividad.

Las condiciones de radiofrecuencia (RF) pueden ser influenciadas por el tipo de antena en uso. Es recomendable ubicar las antenas en un ángulo de 90° entre sí para reducir la posibilidad de diafonía. En caso de ser viable, se sugiere emplear un analizador de espectro u otro equipo similar para evaluar la calidad de las señales transmitidas por el hardware de RF. Si las señales exhiben debilidad o presentan distorsiones, existe la posibilidad de que los UE no logren recibirlas de manera exitosa, lo que a su vez podría afectar el proceso de conexión.

Se recomienda emplear un reloj externo para contrarrestar posibles errores originados por imprecisiones en el reloj interno del módulo de RF en uso. La cuestión proviene de los estrictos límites temporales establecidos por los dispositivos 5G. Los relojes internos incorporados en los dispositivos SDR no siempre presentan la precisión necesaria para operar dentro de los márgenes permitidos por los teléfonos móviles. La implementación de un reloj externo, el cual ostenta una precisión considerablemente mayor que los relojes internos de los SDR, reduce considerablemente la posibilidad de que un UE no sea capaz de detectar o conectarse a la red debido a desafíos de sincronización. Esta medida proporciona una mayor garantía de que los problemas de sincronización no afecten adversamente la conectividad del UE.

Para garantizar el rendimiento óptimo, se sugiere emplear una computadora con un procesador i7 de 9ª generación o superior. Aunque máquinas con especificaciones más modestas también pueden ejecutar srsENB con éxito, su capacidad para alcanzar un rendimiento máximo será más limitada. Además, es crucial configurar el regulador de la CPU de la PC en el modo de rendimiento, de manera que se permita aprovechar al máximo la potencia de cómputo y el rendimiento disponibles.

Además, es esencial verificar que los controladores de los dispositivos SDR estén debidamente actualizados y que la conexión se esté realizando mediante USB 3.0, dado que este último factor también influirá en el logro del rendimiento máximo.

Los requisitos de capacidad computacional de la aplicación srsENB están intrínsecamente ligados al ancho de banda del operador NR implementado. Por ejemplo, lograr el rendimiento óptimo en una portadora de 25 PRB demandará una CPU más potente en comparación con el rendimiento óptimo necesario para una portadora de 50 PRB. En caso de que la capacidad de su equipo no sea suficiente para respaldar la ejecución de srsENB con una configuración de red específica, es posible que experimente notificaciones de paquetes con retrasos y/o desbordamiento desde el módulo de SDR.

Una de las restricciones actuales de srsRAN radica en su compatibilidad exclusiva con un espaciado de 15 kHz entre subportadoras. Esto implica que es crucial tener en cuenta que solamente se pueden emplear dispositivos capaces de operar en bandas que respalden la Duplexión por División de Frecuencia (FDD), como la banda 3, por ejemplo. Numerosos UE comerciales demandan un SCS de 30 kHz para bandas de TDD, una característica que actualmente no se encuentra soportada en el software. Esta limitación resalta la importancia de considerar la compatibilidad de banda y la distribución de subportadoras al seleccionar dispositivos y bandas específicas para el despliegue de la red 5G SA.

10. Bibliografia

- 3GPP, 2019. Release 15 [WWW Document]. URL <https://www.3gpp.org/specifications-technologies/releases/release-15> (accessed 5.1.23).
- 3GPP, 2018. 3GPP TS 23.501.
- Balapuwaduge, I.A., Li, F.Y., 2018. Cellular networks: an evolution from 1g to 4g. *Cent. Integr. Emerg. Manag.*
- Bockelmann, C., Pratas, N., Nikopour, H., Au, K., Svensson, T., Stefanovic, C., Popovski, P., Dekorsy, A., 2016. Massive machine-type communications in 5g: Physical and MAC-layer solutions. *IEEE Commun. Mag.* 54, 59–65. <https://doi.org/10.1109/MCOM.2016.7565189>
- Burns, P., 2003. *Software defined radio for 3G*. Artech House.
- Coimbra, D., 2022. *Control and Positioning of a 5G Radio Access Node Deployed in a Mobile Robotic Platform*. Universidade do porto.
- Dahlman, E., Parkvall, S., Johan, S., 2020. *5G NR: The Next Generation Wireless Access Technology*. Academic Press.
- Dreifuerst, R.M., Member, S., Heath Jr Fellow, R.W., 2023. *Massive MIMO in 5G: How Beamforming, Codebooks, and Feedback Enable Larger Arrays*.
- Dzogovic, B., Do, V.T., Feng, B., Van Do, T., 2018. Building virtualized 5G networks using open source software. *ISCAIE 2018 - 2018 IEEE Symp. Comput. Appl. Ind. Electron.* 360–366. <https://doi.org/10.1109/ISCAIE.2018.8405499>
- ETSI, 2020. 3GPP TS 38.300 version 16.2.0 Release 16.
- ETSI, 2019. 3GPP TR 21.915 version 15.0.0 Release 15.
- ETSI, 2018. 3GPP TS 23.501 version 15.3.0 Release 15.
- Ettus Research, 2023a. USRP B210 [WWW Document]. URL <https://www.ettus.com/all-products/ub210-kit/> (accessed 8.16.23).
- Ettus Research, 2023b. USRP X310 [WWW Document]. URL <https://www.ettus.com/all-products/x310-kit/> (accessed 8.16.23).
- Ettus Research, 2023c. USRP N310 [WWW Document]. URL <https://www.ettus.com/all-products/usrp-n310/> (accessed 8.16.23).
- Fonte, A., Plutino, F., Moquillon, L., Razafimandimby, S., Pruvost, S., 2018. 5G 26 GHz and 28 GHz Bands SiGe:C Receiver with Very High-Linearity and 56 dB Dynamic Range. *EuMIC 2018 - 2018 13th Eur. Microw. Integr. Circuits Conf.* 57–60. <https://doi.org/10.23919/EUMIC.2018.8539921>
- Free5GC, 2023. What is free5GC? [WWW Document]. URL <https://www.free5gc.org/>

(accessed 5.8.23).

- Free5GRAN, 2021. free5GRAN [WWW Document]. URL <https://free5g.github.io/free5GRAN-documentation/index.html> (accessed 5.8.23).
- Fuentes, M., Carcel, J.L., Dietrich, C., Yu, L., Garro, E., Pauli, V., Lazarakis, F.I., Grondalen, O., Bulakci, O., Yu, J., Mohr, W., Gomez-Barquero, D., 2020. 5G New Radio Evaluation against IMT-2020 Key Performance Indicators. *IEEE Access* 8, 110880–110896. <https://doi.org/10.1109/ACCESS.2020.3001641>
- Galal, H., O'Halloran, D., 2020. The Impact of 5G: Creating New Value across Industries and Society. World Econ. Forum Whitepaper.
- Geng, Z., Wei, X., Liu, H., Xu, R., Zheng, K., 2017. Performance analysis and comparison of GPP-based SDR systems. *Proc. 2017 7th IEEE Int. Symp. Microwave, Antenna, Propagation, EMC Technol. MAPE 2017* 2018-January, 124–129. <https://doi.org/10.1109/MAPE.2017.8250816>
- Güngör, A., 2022. Open source 5G UE and RAN (gNodeB) implementation. [WWW Document]. URL <https://github.com/aligungr/UERANSIM> (accessed 5.8.23).
- Hong, W., Jiang, Z.H., Yu, C., Hou, D., Wang, H., Guo, C., Hu, Y., Kuai, L., Yu, Y., Jiang, Z., Chen, Z., Chen, J., Yu, Z., Zhai, J., Zhang, N., Tian, L., Wu, F., Yang, G., Hao, Z.-C., Zhou, J.Y., 2021. The Role of Millimeter-Wave Technologies in 5G/6G Wireless Communications. *IEEE J. Microwaves* 1, 101–122. <https://doi.org/10.1109/JMW.2020.3035541>
- Ibanez, A., Sanchez, J., Gomez-Barquero, D., Mika, J., Babel, S., Kuehnhammer, K., 2022. 5G Broadcast SDR Open Source Platforms. *IEEE Int. Symp. Broadband Multimed. Syst. Broadcast. BMSB 2022-June*. <https://doi.org/10.1109/BMSB55706.2022.9828570>
- ITU-R, 2015. IMT Vision – Framework and overall objectives of the future development of IMT for 2020 and beyond. *Recomm. ITU-R M.2083-0*.
- ITU, 2022. 5G - Fifth generation of mobile technologies [WWW Document]. URL <https://www.itu.int/en/mediacentre/backgrounders/Pages/5G-fifth-generation-of-mobile-technologies.aspx> (accessed 5.1.23).
- Ivezic, M., 2020. Introduction to 3GPP and 3GPP 5G Releases 15, 16 and 17 [WWW Document]. URL <https://5g.security/5g-business-policy/5g-3gpp-releases-15-16-17/> (accessed 5.1.23).
- Kostopoulos, A., Chochliouros, I.P., Munaretto, D., Meani, C., Keuker, C., Garriga, E.T., Hidalgo, J.F., Cid, M.C., Khalife, H., Liberal, F., 2018. Use cases for 5g networks using small cells. *IFIP Adv. Inf. Commun. Technol.* 520, 39–49. <https://doi.org/10.1007/978-3->

- Lien, S.Y., Hung, S.C., Deng, D.J., Wang, Y.J., 2017. Efficient ultra-reliable and low latency communications and massive machine-Type communications in 5G new radio. 2017 IEEE Glob. Commun. Conf. GLOBECOM 2017 - Proc. 2018-January, 1–7. <https://doi.org/10.1109/GLOCOM.2017.8254211>
- Lime Microsystems, 2020. LimeSDR [WWW Document]. URL <https://limemicro.com/products/boards/limesdr/> (accessed 8.18.23).
- Magma, 2023. A modern mobile core network solution [WWW Document]. URL <https://magmacore.org/> (accessed 5.8.23).
- Nieto, Í., 2022. Arquitecturas LTE-5G mediante SDR y OpenAirInterface. Universidad de Valladolid, Valladolid.
- Nordrum, A., Clark, K., 2017a. 5G Bytes: Millimeter Waves Explained [WWW Document]. URL <https://spectrum.ieee.org/5g-bytes-millimeter-waves-explained#toggle-gdpr> (accessed 5.1.23).
- Nordrum, A., Clark, K., 2017b. 5G Bytes: Beamforming Explained [WWW Document]. IEEE Spectr. URL <https://spectrum.ieee.org/5g-bytes-beamforming-explained> (accessed 5.3.23).
- Nuand, 2023a. bladeRF [WWW Document]. URL <https://www.nuand.com/bladerf-1/> (accessed 8.18.23).
- Nuand, 2023b. bladeRF 2.0 micro xA9 [WWW Document]. URL <https://www.nuand.com/product/bladerf-xa9/> (accessed 8.18.23).
- O-RAN, 2023. Open Software for the RAN [WWW Document]. URL <https://www.o-ran.org/software> (accessed 8.20.23).
- Open5GCore, 2023. Open5GCore [WWW Document]. URL <https://www.open5gcore.org/> (accessed 5.8.23).
- Open5GS, 2023a. Open Source implementation for 5G Core and EPC, i.e. the core network of LTE/NR network (Release-17) [WWW Document]. URL <https://open5gs.org/> (accessed 5.8.23).
- Open5GS, 2023b. Quickstart [WWW Document]. URL <https://open5gs.org/open5gs/docs/guide/01-quickstart/> (accessed 2.1.23).
- OpenAirInterface, 2023. 5G CORE NETWORK [WWW Document]. URL <https://openairinterface.org/oai-5g-core-network-project/> (accessed 5.8.23).
- Parkvall, S., Dahlman, E., Furuskar, A., Frenne, M., 2017. NR: The new 5G radio access technology. IEEE Commun. Stand. Mag. 1, 24–30.

<https://doi.org/10.1109/MCOMSTD.2017.1700042>

- Polese, M., Bonati, L., D’Oro, S., Basagni, S., Melodia, T., 2023. Understanding O-RAN: Architecture, Interfaces, Algorithms, Security, and Research Challenges. *IEEE Commun. Surv. Tutorials* 25, 1376–1411. <https://doi.org/10.1109/COMST.2023.3239220>
- Rahman, I., Razavi, S.M., Liberg, O., Hoymann, C., Wiemann, H., Tidestav, C., Schliwa-Bertling, P., Persson, P., Gerstenberger, D., 2022. 5G Evolution Toward 5G Advanced: An overview of 3GPP releases 17 and 18. *Ericsson Technol. Rev.* 2021, 2–12. <https://doi.org/10.23919/ETR.2021.9904665>
- Rinaldi, F., Raschellà, A., Pizzi, S., 2021. 5G NR system design: a concise survey of key features and capabilities. *Wirel. Networks* 27, 5173–5188. <https://doi.org/10.1007/S11276-021-02811-Y/FIGURES/11>
- Rischke, J., Sossalla, P., Itting, S., Fitzek, F.H.P., Reisslein, M., 2021. 5G Campus Networks: A First Measurement Study. *IEEE Access* 9, 121786–121803. <https://doi.org/10.1109/ACCESS.2021.3108423>
- Sadiku, M.N., Akujuobi, C.M., 2004. Software-defined radio: a brief overview. *Ieee Potentials* 23, 14–15.
- She, C., Pan, C., Duong, T.Q., Quek, T.Q.S., Schober, R., Simsek, M., Zhu, P., 2023. xURLLC in 6G: Next Generation Ultra-Reliable and Low-Latency Communications [WWW Document]. *IEEE Commun. Soc. URL* <https://www.comsoc.org/publications/journals/ieee-jsac/cfp/xurllic-6g-next-generation-ultra-reliable-and-low-latency> (accessed 5.4.23).
- Simmons, A., 2023. 5G Standalone (SA): What is it? and How Does it Work? [WWW Document]. URL <https://dgtlinfra.com/5g-standalone-sa/> (accessed 8.15.23).
- srsRAN, 2023a. srsRAN Project - Open Source RAN [WWW Document]. URL <https://www.srslte.com/> (accessed 5.8.23).
- srsRAN, 2023b. 5G SA COTS UE [WWW Document]. URL https://docs.srsran.com/projects/4g/en/latest/app_notes/source/5g_sa_COTS/source/index.html (accessed 5.28.23).
- srsRAN, 2022a. srsRAN 22.04.1 Documentation [WWW Document]. URL https://docs.srsran.com/projects/4g/en/hovertxref_test/index.html (accessed 8.22.23).
- srsRAN, 2022b. srsRAN 4G 22.10 Documentation [WWW Document]. URL <https://docs.srsran.com/projects/4g/en/next/index.html> (accessed 8.22.23).
- srsRAN, 2022c. Introduction — srsRAN 22.04.1 documentation [WWW Document]. URL https://docs.srsran.com/projects/4g/en/hovertxref_test/usermanuals/source/srsenb/source/

1_enb_intro.html (accessed 8.22.23).

Sysmocom, 2020. New Product: sysmoISIM-SJA2 [WWW Document]. URL <https://www.sysmocom.de/news/sysmoISIM-SJA2/index.html> (accessed 5.9.23).

Ulversoy, T., 2010. Software defined radio: Challenges and opportunities. IEEE Commun. Surv. Tutorials 12, 531–550. <https://doi.org/10.1109/SURV.2010.032910.00019>

Vahid, S., Tafazolli, R., Filo, M., 2015. Small Cells for 5G Mobile Networks. Fundam. 5G Mob. Networks 63–104. <https://doi.org/10.1002/9781118867464.CH3>

Wipro Technologies, 2002. Software-Defined Radio White Paper.

Wireless Innovation Forum, 2017. Introduction to Software Defined Radio [WWW Document]. URL https://www.wirelessinnovation.org/Introduction_to_SDR (accessed 5.7.23).

11. Anexos

Anexo 1. Archivo de configuración enb.conf

```
#####
# srsENB configuration file
#####

#####
# eNB configuration
#
# enb_id: 20-bit eNB identifier.
# mcc: Mobile Country Code
# mnc: Mobile Network Code
# mme_addr: IP address of MME for S1 connection
# gtp_bind_addr: Local IP address to bind for GTP connection
# gtp_advertise_addr: IP address of eNB to advertise for DL GTP-U Traffic
# s1c_bind_addr: Local IP address to bind for S1AP connection
# s1c_bind_port: Source port for S1AP connection (0 means any)
# n_prb: Number of Physical Resource Blocks (6,15,25,50,75,100)
# tm: Transmission mode 1-4 (TM1 default)
# nof_ports: Number of Tx ports (1 port default, set to 2 for TM2/3/4)
#
#####
[enb]
enb_id = 0x19B
mcc = 001
mnc = 01
mme_addr = 127.0.0.2
gtp_bind_addr = 127.0.1.1
s1c_bind_addr = 127.0.1.1
s1c_bind_port = 0
n_prb = 50
#tm = 4
#nof_ports = 2

#####
# eNB configuration files
#
# sib_config: SIB1, SIB2 and SIB3 configuration file
# note: When enabling MBMS, use the sib.conf.mbsfn configuration file which includes SIB13
# rr_config: Radio Resources configuration file
# rb_config: SRB/DRB configuration file
#####
[enb_files]
sib_config = sib.conf
rr_config = rr.conf
rb_config = rb.conf

#####
# RF configuration
#
# dl_earfcn: EARFCN code for DL (only valid if a single cell is configured in rr.conf)
# tx_gain: Transmit gain (dB).
# rx_gain: Optional receive gain (dB). If disabled, AGC if enabled
#
# Optional parameters:
# dl_freq: Override DL frequency corresponding to dl_earfcn
# ul_freq: Override UL frequency corresponding to dl_earfcn (must be set if dl_freq is set)
# device_name: Device driver family
# Supported options: "auto" (uses first driver found), "UHD", "bladeRF", "soapy", "zmq" or
"Sidekiq"
# device_args: Arguments for the device driver. Options are "auto" or any string.
# Default for UHD: "recv_frame_size=9232,send_frame_size=9232"
# Default for bladeRF: ""
```

```

# time_adv_nsamples: Transmission time advance (in number of samples) to compensate for RF
delay
# from antenna to timestamp insertion.
# Default "auto". B210 USRP: 100 samples, bladeRF: 27
#####
[rf]
#dl_earfcn = 3350
tx_gain = 50
rx_gain = 40

device_name = auto

# For best performance in 2x2 MIMO and >= 15 MHz use the following device_args settings:
# USRP B210: num_recv_frames=64,num_send_frames=64
# And for 75 PRBs, also append ",master_clock_rate=15.36e6" to the device args

# For best performance when BW<5 MHz (25 PRB), use the following device_args settings:
# USRP B210: send_frame_size=512,recv_frame_size=512

#device_args = auto
#time_adv_nsamples = auto

# Example for ZMQ-based operation with TCP transport for I/Q samples
#device_name = zmq
#device_args =
fail_on_disconnect=true,tx_port=tcp://*:2000,rx_port=tcp://localhost:2001,id=enb,base_srate
=23.04e6

#####
# Packet capture configuration
#
# MAC-layer packets are captured to a file in the compact format which can
# be decoded by Wireshark. For decoding, use the UDP dissector and the UDP
# heuristic dissection. Edit the preferences (Edit > Preferences >
# Protocols > DLT_USER) for DLT_USER to add an entry for DLT=149 with
# Protocol=udp. Further, enable the heuristic dissection in UDP under:
# Analyze > Enabled Protocols > MAC-LTE > mac_lte_udp and MAC-NR > mac_nr_udp
# For more information see: https://wiki.wireshark.org/MAC-LTE
# Configuring this Wireshark preferences is needed for decoding the MAC PCAP
# files as well as for the live network capture option.
#
# Please note that this setting will by default only capture MAC
# frames on dedicated channels, and not SIB. You have to build with
# WRITE_SIB_PCAP enabled in srsenb/src/stack/mac/mac.cc if you want
# SIB to be part of the MAC pcap file.
#
# S1AP Packets are captured to a file in the compact format which can
# be decoded by the Wireshark s1ap dissector with DLT 150.
# To use the dissector, edit the preferences for DLT_USER to
# add an entry with DLT=150, Payload Protocol=s1ap.
#
# enable: Enable MAC layer packet captures (true/false)
# filename: File path to use for LTE MAC packet captures
# nr_filename: File path to use for NR MAC packet captures
# s1ap_enable: Enable or disable the PCAP.
# s1ap_filename: File name where to save the PCAP.
#
# mac_net_enable: Enable MAC layer packet captures sent over the network (true/false
default: false)
# bind_ip: Bind IP address for MAC network trace (default: "0.0.0.0")
# bind_port: Bind port for MAC network trace (default: 5687)
# client_ip: Client IP address for MAC network trace (default: "127.0.0.1")
# client_port Client IP address for MAC network trace (default: 5847)
#####
[pcap]
#enable = false

```

```

#filename = /tmp/enb_mac.pcap
#nr_filename = /tmp/enb_mac_nr.pcap
#slap_enable = false
#slap_filename = /tmp/enb_slap.pcap

#mac_net_enable = false
#bind_ip = 0.0.0.0
#bind_port = 5687
#client_ip = 127.0.0.1
#client_port = 5847

#####
# Log configuration
#
# Log levels can be set for individual layers. "all_level" sets log
# level for all layers unless otherwise configured.
# Format: e.g. phy_level = info
#
# In the same way, packet hex dumps can be limited for each level.
# "all_hex_limit" sets the hex limit for all layers unless otherwise
# configured.
# Format: e.g. phy_hex_limit = 32
#
# Logging layers: rf, phy, phy_lib, mac, rlc, pdcp, rrc, gtpu, slap, stack, all
# Logging levels: debug, info, warning, error, none
#
# filename: File path to use for log output. Can be set to stdout
# to print logs to standard output
# file_max_size: Maximum file size (in kilobytes). When passed, multiple files are created.
# If set to negative, a single log file will be created.
#####
[log]
all_level = warning
all_hex_limit = 32
filename = /tmp/enb.log
file_max_size = -1

[gui]
enable = false

#####
# Scheduler configuration options
#
# sched_policy: User MAC scheduling policy (E.g. time_rr, time_pf)
# min_aggr_level: Optional minimum aggregation level index (l=log2(L) can be 0, 1, 2 or 3)
# max_aggr_level: Optional maximum aggregation level index (l=log2(L) can be 0, 1, 2 or 3)
# adaptive_aggr_level: Boolean flag to enable/disable adaptive aggregation level based on
target BLER
# pdsch_mcs: Optional fixed PDSCH MCS (ignores reported CQIs if specified)
# pdsch_max_mcs: Optional PDSCH MCS limit
# pusch_mcs: Optional fixed PUSCH MCS (ignores reported CQIs if specified)
# pusch_max_mcs: Optional PUSCH MCS limit
# min_nof_ctrl_symbols: Minimum number of control symbols
# max_nof_ctrl_symbols: Maximum number of control symbols
# pucch_multiplex_enable: Allow PUCCH HARQ to collide with PUSCH and other PUCCH
# pucch_harq_max_rb: Maximum number of RB to be used for PUCCH on the edges of the grid.
# If defined and greater than 0, the scheduler will avoid DL PDCCH allocations if
# PUCCH HARQ falls outside this region
# target_bler: Target BLER (in decimal) to achieve via adaptive link
# max_delta_dl_cqi: Maximum shift in CQI for adaptive DL link
# max_delta_ul_snr: Maximum shift in UL SNR for adaptive UL link
# adaptive_dl_mcs_step_size: Step size or learning rate used in adaptive DL MCS link
# adaptive_ul_mcs_step_size: Step size or learning rate used in adaptive UL MCS link
# min_tpc_tti_interval: Minimum TTI interval between TPCs different than 1
# ul_snr_avg_alpha: Exponential Average alpha coefficient used in estimation of UL SNR
# init_ul_snr_value: Initial UL SNR value used for computing MCS in the first UL grant

```

```

# init_dl_cqi: DL CQI value used before any CQI report is available to the eNB
# max_sib_coderate: Upper bound on SIB and RAR grants coderate
# pdcch_cqi_offset: CQI offset in derivation of PDCCH aggregation level
# nr_pdsch_mcs: Optional fixed NR PDSCH MCS (ignores reported CQIs if specified)
# nr_pusch_mcs: Optional fixed NR PUSCH MCS (ignores reported CQIs if specified)
#
#####
[scheduler]
#policy = time_pf
#policy_args = 2
#min_aggr_level = 0
#max_aggr_level = 3
#adaptive_aggr_level = false
#pdsch_mcs = -1
#pdsch_max_mcs = -1
#pusch_mcs = -1
#pusch_max_mcs = 16
#min_nof_ctrl_symbols = 1
#max_nof_ctrl_symbols = 3
#pucch_multiplex_enable = false
#pucch_harq_max_rb = 0
#target_bler = 0.05
#max_delta_dl_cqi = 5
#max_delta_ul_snr = 5
#adaptive_dl_mcs_step_size = 0.001
#adaptive_ul_mcs_step_size = 0.001
#min_tpc_tti_interval = 1
#ul_snr_avg_alpha=0.05
#init_ul_snr_value=5
#init_dl_cqi=5
#max_sib_coderate=0.3
#pdcch_cqi_offset=0
nr_pdsch_mcs=28
nr_pusch_mcs=28

#####
# eMBMS configuration options
#
# enable: Enable MBMS transmission in the eNB
# m1u_multiaddr: Multicast address the M1-U socket will register to
# m1u_if_addr: Address of the interface the M1-U interface will listen to for multicast
packets
# mcs: Modulation and Coding scheme for MBMS traffic
#
#####
[embms]
#enable = false
#m1u_multiaddr = 239.255.0.1
#m1u_if_addr = 127.0.1.201
#mcs = 20

#####
# Channel emulator options:
# enable: Enable/disable internal Downlink/Uplink channel emulator
#
# -- AWGN Generator
# awgn.enable: Enable/disable AWGN generator
# awgn.snr: Target SNR in dB
#
# -- Fading emulator
# fading.enable: Enable/disable fading simulator
# fading.model: Fading model + maximum doppler (E.g. none, epa5, eva70, etu300, etc)
#

```



```

# -- Delay Emulator delay(t) = delay_min + (delay_max - delay_min) * (1 +
sin(2pi*t/period)) / 2
# Maximum speed [m/s]: (delay_max - delay_min) * pi * 300 / period
# delay.enable: Enable/disable delay simulator
# delay.period_s: Delay period in seconds
# delay.init_time_s: Delay initial time in seconds
# delay.maximum_us: Maximum delay in microseconds
# delay.minumum_us: Minimum delay in microseconds
#
# -- Radio-Link Failure (RLF) Emulator
# rlf.enable: Enable/disable RLF simulator
# rlf.t_on_ms: Time for On state of the channel (ms)
# rlf.t_off_ms: Time for Off state of the channel (ms)
#
# -- High Speed Train Doppler model simulator
# hst.enable: Enable/disable HST simulator
# hst.period_s: HST simulation period in seconds
# hst.fd_hz: Doppler frequency in Hz
# hst.init_time_s: Initial time in seconds
#####
[channel.dl]
#enable = false

[channel.dl.awgn]
#enable = false
#snr = 30

[channel.dl.fading]
#enable = false
#model = none

[channel.dl.delay]
#enable = false
#period_s = 3600
#init_time_s = 0
#maximum_us = 100
#minimum_us = 10

[channel.dl.rlf]
#enable = false
#t_on_ms = 10000
#t_off_ms = 2000

[channel.dl.hst]
#enable = false
#period_s = 7.2
#fd_hz = 750.0
#init_time_s = 0.0

[channel.ul]
#enable = false

[channel.ul.awgn]
#enable = false
#n0 = -30

[channel.ul.fading]
#enable = false
#model = none

[channel.ul.delay]
#enable = false
#period_s = 3600
#init_time_s = 0
#maximum_us = 100
#minimum_us = 10

```

```

[channel.ul.rlf]
#enable = false
#t_on_ms = 10000
#t_off_ms = 2000

[channel.ul.hst]
#enable = false
#period_s = 7.2
#fd_hz = -750.0
#init_time_s = 0.0

#####
# CFR configuration options
#
# The CFR module provides crest factor reduction for the transmitted signal.
#
# enable: Enable or disable the CFR. Default: disabled
#
# mode: manual: CFR threshold is set by cfr_manual_thres (default).
# auto_ema: CFR threshold is adaptive based on the signal PAPR. Power avg. with Exponential
Moving Average.
# The time constant of the averaging can be tweaked with the ema_alpha parameter.
# auto_cma: CFR threshold is adaptive based on the signal PAPR. Power avg. with Cumulative
Moving Average.
# Use with care, as CMA's increasingly slow response may be unsuitable for most use cases.
#
# strength: Ratio between amplitude-limited vs unprocessed signal (0 to 1). Default: 1
# manual_thres: Fixed manual clipping threshold for CFR manual mode. Default: 0.5
# auto_target_papr: Signal PAPR target (in dB) in CFR auto modes. output PAPR can be higher
due to peak smoothing. Default: 8
# ema_alpha: Alpha coefficient for the power average in auto_ema mode. Default: 1/7
#
#####
[cfr]
#enable = false
#mode = manual
#manual_thres = 0.5
#strength = 1
#auto_target_papr = 8
#ema_alpha = 0.0143

#####
# Expert configuration options
#
# pusch_max_its: Maximum number of turbo decoder iterations (default: 4)
# nr_pusch_max_its: Maximum number of LDPC iterations for NR (Default 10)
# pusch_8bit_decoder: Use 8-bit for LLR representation and turbo decoder trellis
computation (experimental)
# nof_phy_threads: Selects the number of PHY threads (maximum: 4, minimum: 1, default: 3)
# metrics_period_secs: Sets the period at which metrics are requested from the eNB
# metrics_csv_enable: Write eNB metrics to CSV file.
# metrics_csv_filename: File path to use for CSV metrics
# report_json_enable: Write eNB report to JSON file (default: disabled)
# report_json_filename: Report JSON filename (default: /tmp/enb_report.json)
# report_json_asn1_oct: Prints ASN1 messages encoded as an octet string instead of plain
text in the JSON report file
# alarms_log_enable: Enable Alarms logging (default: disabled)
# alarms_filename: Alarms logging filename (default: /tmp/alarms.log)
# tracing_enable: Write source code tracing information to a file
# tracing_filename: File path to use for tracing information
# tracing_buffcapacity: Maximum capacity in bytes the tracing framework can store
# stdout_ts_enable: Prints once per second the timestamp into stdout
# tx_amplitude: Transmit amplitude factor (set 0-1 to reduce PAPR)
# rrc_inactivity_timer Inactivity timeout used to remove UE context from RRC (in
milliseconds)

```

```

# max_mac_dl_kos: Maximum number of consecutive KOs in DL before triggering the UE's
release (default: 100)
# max_mac_ul_kos: Maximum number of consecutive KOs in UL before triggering the UE's
release (default: 100)
# max_prach_offset_us: Maximum allowed RACH offset (in us)
# nof_prealloc_ues: Number of UE memory resources to preallocate during eNB initialization
for faster UE creation (default: 8)
# rlf_release_timer_ms: Time taken by eNB to release UE context after it detects an RLF
# eea_pref_list: Ordered preference list for the selection of encryption algorithm (EEA)
(default: EEA0, EEA2, EEA1)
# eia_pref_list: Ordered preference list for the selection of integrity algorithm (EIA)
(default: EIA2, EIA1, EIA0)
# gtpu_tunnel_timeout: Time that GTPU takes to release indirect forwarding tunnel since the
last received GTPU PDU (0 for no timer)
# ts1_reloc_prep_timeout: S1AP TS 36.413 TS1RelocPrep Expiry Timeout value in milliseconds
# ts1_reloc_overall_timeout: S1AP TS 36.413 TS1RelocOverall Expiry Timeout value in
milliseconds
# rlf_release_timer_ms: Time taken by eNB to release UE context after it detects a RLF
# rlf_min_ul_snr_estim: SNR threshold in dB below which the enb is notified with RLF ko
# s1_setup_max_retries: Maximum amount of retries to setup the S1AP connection. If this
value is exceeded, an alarm is written to the log. -1 means infinity.
# s1_connect_timer: Connection Retry Timer for S1 connection (seconds)
# rx_gain_offset: RX Gain offset to add to rx_gain to calibrate RSRP readings
#####
[expert]
#pusch_max_its = 8 # These are half iterations
#nr_pusch_max_its = 10
#pusch_8bit_decoder = false
#nof_phy_threads = 3
#metrics_period_secs = 1
#metrics_csv_enable = false
#metrics_csv_filename = /tmp/enb_metrics.csv
#report_json_enable = true
#report_json_filename = /tmp/enb_report.json
#report_json_asn1_oct = false
#alarms_log_enable = true
#alarms_filename = /tmp/enb_alarms.log
#tracing_enable = true
#tracing_filename = /tmp/enb_tracing.log
#tracing_buffcapacity = 1000000
#stdout_ts_enable = false
#tx_amplitude = 0.6
#rrc_inactivity_timer = 30000
#max_mac_dl_kos = 100
#max_mac_ul_kos = 100
#max_prach_offset_us = 30
#nof_prealloc_ues = 8
#rlf_release_timer_ms = 4000
#lcid_padding = 3
#eea_pref_list = EEA0, EEA2, EEA1
#eia_pref_list = EIA2, EIA1, EIA0
#gtpu_tunnel_timeout = 0
#extended_cp = false
#ts1_reloc_prep_timeout = 10000
#ts1_reloc_overall_timeout = 10000
#rlf_release_timer_ms = 4000
#rlf_min_ul_snr_estim = -2
#s1_setup_max_retries = -1
#s1_connect_timer = 10
#rx_gain_offset = 62
#mac_prach_bi = 0

```

Anexo 2. Archivo de configuración rr.conf

```
mac_cnfg =
{
  phr_cnfg =
  {
    dl_pathloss_change = "dB3"; // Valid: 1, 3, 6 or INFINITY
    periodic_phr_timer = 50;
    prohibit_phr_timer = 0;
  };
  ulsch_cnfg =
  {
    max_harq_tx = 4;
    periodic_bsr_timer = 20; // in ms
    retx_bsr_timer = 320; // in ms
  };

  time_alignment_timer = -1; // -1 is infinity
};

phy_cnfg =
{
  phich_cnfg =
  {
    duration = "Normal";
    resources = "1/6";
  };

  pusch_cnfg_ded =
  {
    beta_offset_ack_idx = 6;
    beta_offset_ri_idx = 6;
    beta_offset_cqi_idx = 6;
  };

  // PUCCH-SR resources are scheduled on time-frequency domain first, then multiplexed in the
  same resource.
  sched_request_cnfg =
  {
    dsr_trans_max = 64;
    period = 20; // in ms
    //subframe = [1, 11]; // Optional vector of subframe indices allowed for SR transmissions
    (default uses all)
    nof_prb = 1; // number of PRBs on each extreme used for SR (total prb is twice this
    number)
  };
  cqi_report_cnfg =
  {
    mode = "periodic";
    simultaneousAckCQI = true;
    period = 40; // in ms
    //subframe = [0, 10, 20, 30]; // Optional vector of subframe indices every period where
    CQI resources will be allocated (default uses all)
    m_ri = 8; // RI period in CQI period
    //subband_k = 1; // If enabled and > 0, configures sub-band CQI reporting and defines K
    (see 36.213 7.2.2). If disabled, configures wideband CQI
  };
};

cell_list =
(
  //--- COMMENT OUT LTE CELLS ---//
  /*
  {
    // rf_port = 0;
    cell_id = 0x01;
  }
  */
);
```

```

tac = 0x0007;
pci = 1;
// root_seq_idx = 204;
dl_earfcn = 3350;
//ul_earfcn = 21400;
ho_active = false;
//meas_gap_period = 0; // 0 (inactive), 40 or 80
//meas_gap_offset_subframe = [6, 12, 18, 24, 30];
// target_pusch_sinr = -1;
// target_pucch_sinr = -1;
// enable_phr_handling = false;
// min_phr_thres = 0;
// allowed_meas_bw = 6;
// t304 = 2000; // in msec. possible values: 50, 100, 150, 200, 500, 1000, 2000

// CA cells
scell_list = (
// {cell_id = 0x02; cross_carrier_scheduling = false; scheduling_cell_id = 0x02;
ul_allowed = true}
)

// Cells available for handover
meas_cell_list =
(
{
eci = 0x19C02;
dl_earfcn = 2850;
pci = 2;
//direct_forward_path_available = false;
//allowed_meas_bw = 6;
//cell_individual_offset = 0;
}
);

// Select measurement report configuration (all reports are combined with all measurement
objects)
meas_report_desc =
(
{
eventA = 3
a3_offset = 6;
hysteresis = 0;
time_to_trigger = 480;
trigger_quant = "RSRP";
max_report_cells = 1;
report_interv = 120;
report_amount = 1;
}
);
meas_quant_desc = {
// averaging filter coefficient
rsrq_config = 4;
rsrp_config = 4;
};
}
*/
// Add here more cells
);

nr_cell_list =
(
{
rf_port = 0;
cell_id = 1;
root_seq_idx = 1;
tac = 7;

```

```
pci = 500;  
dl_arfcn = 368500;  
coreset0_idx = 6;  
band = 3;  
}  
);
```

Anexo 3. Archivo de configuración amf.yaml

```
#
# o Set OGS_LOG_INFO to all domain level
# - If `level` is omitted, the default level is OGS_LOG_INFO)
# - If `domain` is omitted, the all domain level is set from 'level'
# (Default values are used, so no configuration is required)
#
# o Set OGS_LOG_ERROR to all domain level
# - `level` can be set with none, fatal, error, warn, info, debug, trace
# logger:
# level: error
#
# o Set OGS_LOG_DEBUG to mme/emm domain level
# logger:
# level: debug
# domain: mme,emm
#
# o Set OGS_LOG_TRACE to all domain level
# logger:
# level: trace
# domain: core,sbi,ausf,event,tlv,mem,sock
#
logger:
  file: /var/log/open5gs/amf.log

#
# o TLS enable/disable
# sbi:
# server|client:
# no_tls: false|true
# - false: (Default) Use TLS
# - true: TLS disabled
#
# o Verification enable/disable
# sbi:
# server|client:
# no_verify: false|true
# - false: (Default) Verify the PEER
# - true: Skip the verification step
#
# o Server-side does not use TLS
# sbi:
# server:
# no_tls: true
#
# o Client-side skips the verification step
# sbi:
# client:
# no_verify: true
# key: /etc/open5gs/tls/amf.key
# cert: /etc/open5gs/tls/amf.crt
#
# o Use the specified certificate while verifying the client
# sbi:
# server
# cacert: /etc/open5gs/tls/ca.crt
#
# o Use the specified certificate while verifying the server
# sbi:
# client
# cacert: /etc/open5gs/tls/ca.crt
#
sbi:
  server:
  no_tls: true
```

```

cacert: /etc/open5gs/tls/ca.crt
key: /etc/open5gs/tls/amf.key
cert: /etc/open5gs/tls/amf.crt
client:
no_tls: true
cacert: /etc/open5gs/tls/ca.crt
key: /etc/open5gs/tls/amf.key
cert: /etc/open5gs/tls/amf.crt

#
# <SBI Server>
#
# o SBI Server(http://<all address available>:80)
# sbi:
# server:
# no_tls: true
# amf:
# sbi:
#
# o SBI Server(http://<any address>:7777)
# sbi:
# server:
# no_tls: true
# amf:
# sbi:
# - addr:
# - 0.0.0.0
# - ::0
# port: 7777
#
# o SBI Server(https://<all address available>:443)
# sbi:
# server:
# key: /etc/open5gs/tls/amf.key
# cert: /etc/open5gs/tls/amf.crt
# amf:
# sbi:
#
# o SBI Server(https://127.0.0.5:443, https://[::1]:443) without verification
# sbi:
# server:
# no_verify: true
# key: /etc/open5gs/tls/amf.key
# cert: /etc/open5gs/tls/amf.crt
# amf:
# sbi:
# - addr: 127.0.0.5
# - addr: ::1
#
# o SBI Server(https://amf.open5gs.org:443)
# Use the specified certificate while verifying the client
#
# sbi:
# server:
# cacert: /etc/open5gs/tls/ca.crt
# key: /etc/open5gs/tls/amf.key
# cert: /etc/open5gs/tls/amf.crt
# amf:
# sbi:
# - name: amf.open5gs.org
#
# o SBI Server(http://127.0.0.5:7777)
# sbi:
# server:
# no_tls: true
# amf:

```



```

# sbi:
# - addr: 127.0.0.5
# port: 7777
#
# o SBI Server(http://<eth0 IP address>:80)
# sbi:
# server:
# no_tls: true
# amf:
# sbi:
# - dev: eth0
#
# o Provide custom SBI address to be advertised to NRF
# sbi:
# server:
# no_tls: true
# amf:
# sbi:
# - dev: eth0
# advertise: open5gs-amf.svc.local
#
# o Another example of advertising on NRF
# sbi:
# server:
# no_tls: true
# amf:
# sbi:
# - addr: localhost
# advertise:
# - 127.0.0.99
# - ::1
#
# o SBI Option (Default)
# - tcp_nodelay : true
# - so_linger.l_onoff : false
#
# sbi:
# server:
# no_tls: true
# amf:
# sbi:
# addr: 127.0.0.5
# option:
# tcp_nodelay: false
# so_linger:
# l_onoff: true
# l_linger: 10
#
# <NF Service>
#
# o NF Service Name(Default : all NF services available)
# amf:
# service_name:
#
# o NF Service Name(Only some NF services are available)
# amf:
# service_name:
# - namf-comm
#
# <NF Discovery Query Parameter>
#
# o (Default) If you do not set Query Parameter as shown below,
#
# sbi:
# server:
# no_tls: true

```

```

# amf:
# sbi:
# - addr: 127.0.0.5
# port: 7777
#
# - 'service-names' is included.
#
# o Service-Names are not included
# sbi:
# server:
# no_tls: true
# amf:
# sbi:
# - addr: 127.0.0.5
# port: 7777
# discovery:
# option:
# no_service_names: false
#
# o To remove 'service-names' from URI query parameters in NS Discovery
# no_service_names: true
#
# * For Indirect Communication with Delegated Discovery,
# 'service-names' is always included in the URI query parameter.
# * That is, 'no_service_names' has no effect.
#
# <For Indirect Communication with Delegated Discovery>
#
# o (Default) If you do not set Delegated Discovery as shown below,
#
# sbi:
# server:
# no_tls: true
# amf:
# sbi:
# - addr: 127.0.0.5
# port: 7777
#
# - Use SCP if SCP available. Otherwise NRF is used.
# => App fails if both NRF and SCP are unavailable.
#
# sbi:
# server:
# no_tls: true
# amf:
# sbi:
# - addr: 127.0.0.5
# port: 7777
# discovery:
# delegated: auto
#
# o To use SCP always => App fails if no SCP available.
# delegated: yes
#
# o Don't use SCP server => App fails if no NRF available.
# delegated: no
#
# <NGAP Server>>
#
# o NGAP Server(all address available)
# amf:
# ngap:
#
# o NGAP Server(0.0.0.0:38412)
# amf:
# ngap:

```

```

# addr: 0.0.0.0
#
# o NGAP Server(127.0.0.5:38412, [::1]:38412)
# amf:
# ngap:
# - addr: 127.0.0.5
# - addr: ::1
#
# o NGAP Server(different port)
# amf:
# ngap:
# - addr: 127.0.0.5
# port: 38413
#
# o NGAP Server(address available in `eth0` interface)
# amf:
# ngap:
# dev: eth0
#
# o NGAP Option (Default)
# - sctp_nodelay : true
# - so_linger.l_onoff : false
#
# amf:
# ngap:
# addr: 127.0.0.5
# option:
# stcp_nodelay: false
# so_linger:
# l_onoff: true
# l_linger: 10
#
# o NGAP SCTP Option (Default)
# - spp_hbinterval : 5000 (5secs)
# - spp_sackdelay : 200 (200ms)
# - srto_initial : 3000 (3secs)
# - srto_min : 1000 (1sec)
# - srto_max : 5000 (5secs)
# - sinit_num_ostreams : 30
# - sinit_max_instreams : 65535
# - sinit_max_attempts : 4
# - sinit_max_init_timeo : 8000(8secs)
#
# amf:
# ngap:
# addr: 127.0.0.5
# option:
# sctp:
# spp_hbinterval : 5000
# spp_sackdelay : 200
# srto_initial : 3000
# srto_min : 1000
# srto_max : 5000
# sinit_num_ostreams : 30
# sinit_max_instreams : 65535
# sinit_max_attempts : 4
# sinit_max_init_timeo : 8000
#
# <Metrics Server>
#
# o Metrics Server(http://<any address>:9090)
# amf:
# metrics:
# - addr: 0.0.0.0
# port: 9090
#
#

```

```

# <GUAMI>
#
# o Multiple GUAMI
# amf:
# guami:
# - plmn_id:
# mcc: 999
# mnc: 70
# amf_id:
# region: 2
# set: 1
# pointer: 4
# - plmn_id:
# mcc: 001
# mnc: 01
# amf_id:
# region: 5
# set: 2
#
# <TAI>
#
# o Multiple TAI
#
# When multiple TAIs are configured as shown below,
# the Served TAI is determined by comparing UserLocationInformation
# of UplinkNASTransport sent from gNB.
#
# For example, if the gNB sends TAC with 30 to the AMF,
# the fourth TAI (TAC: 20, 28, 29-32, 36-38, 40-42, 50, 60, 70, 70)
# is determined as the Served TAI. The result is transmitted to the gNB
# as a Tracking Area identity List in Registration Accept.
#
# amf:
# tai:
# - plmn_id:
# mcc: 001
# mnc: 01
# tac: [1, 3, 5]
# tai:
# - plmn_id:
# mcc: 002
# mnc: 02
# tac: [6-10, 15-18]
# tai:
# - plmn_id:
# mcc: 003
# mnc: 03
# tac: 20
# - plmn_id:
# mcc: 004
# mnc: 04
# tac: 21
# tai:
# - plmn_id:
# mcc: 005
# mnc: 05
# tac: [22, 28]
# - plmn_id:
# mcc: 006
# mnc: 06
# tac: [30-32, 34, 36-38, 40-42, 44, 46, 48]
# - plmn_id:
# mcc: 007
# mnc: 07
# tac: 50
# - plmn_id:

```

```

# mcc: 008
# mnc: 08
# tac: 60
# - plmn_id:
# mcc: 009
# mnc: 09
# tac: [70, 80]
#
# <PLMN Support>
#
# o Multiple PLMN Support
# amf:
# plmn_support:
# - plmn_id:
# mcc: 999
# mnc: 70
# s_nssai:
# - sst: 1
# sd: 010000
# - plmn_id:
# mcc: 999
# mnc: 70
# s_nssai:
# - sst: 1
#
#
# <Access Control>
#
# If access_control is not specified, then all networks are allowed
# If access_control is defined,
# no other networks are allowed other than matching plmn_id.
#
# default_reject_cause may be used to overwrite the default error cause #11
# for non matching plmn_id
#
# for matching plmn_id with reject_cause defined,
# the AMF rejects access with the reject_cause error cause
#
# for matching plmn_id without reject_cause defined,
# the AMF accepts the PLMN traffic
#
# o The example below only accepts 002/02 and 999/70 PLMNs.
# 001/01 is rejected with cause 15,
# and the rest of the PLMNs are rejected with default cause 13.
#
# amf:
# access_control:
# - default_reject_cause: 13
# - plmn_id:
# reject_cause: 15
# mcc: 001
# mnc: 01
# - plmn_id:
# mcc: 002
# mnc: 02
# - plmn_id:
# mcc: 999
# mnc: 70
#
#
# <Network Name>
#
# amf:
# network_name:
# full: Open5GS
# short: Next

```

```

#
# <AMF Name>
#
# amf:
# amf_name: amf1.open5gs.amf.5gc.mnc70.mcc999.3gppnetwork.org
#
# <Relative Capacity> - Default(255)
#
# amf:
# relative_capacity: 100
#
amf:
  sbi:
    - addr: 127.0.0.5
    port: 7777
  ngap:
    - addr: 127.0.0.2
  metrics:
    - addr: 127.0.0.5
    port: 9090
  guami:
    - plmn_id:
      mcc: 001
      mnc: 01
      amf_id:
        region: 2
        set: 1
      tai:
        - plmn_id:
          mcc: 001
          mnc: 01
          tac: 7
        plmn_support:
          - plmn_id:
            mcc: 001
            mnc: 01
          s_nssai:
            - sst: 1
          security:
            integrity_order : [ NIA2, NIA1, NIA0 ]
            ciphering_order : [ NEA0, NEA1, NEA2 ]
          network_name:
            full: Open5GS
            amf_name: open5gs-amf0
#
# <SBI Client>>
#
# o SBI Client(http://127.0.1.10:7777)
# sbi:
# client:
# no_tls: true
# scp:
# sbi:
# addr: 127.0.1.10
# port: 7777
#
# o SBI Client(https://127.0.1.10:443, https://[::1]:443) without verification
# sbi:
# client:
# no_verify: true
# key: /etc/open5gs/tls/amf.key
# cert: /etc/open5gs/tls/amf.crt
# scp:
# sbi:
# - addr: 127.0.1.10

```

```

# - addr: ::1
#
# o SBI Client(https://scp.open5gs.org:443)
# Use the specified certificate while verifying the server
#
# sbi:
# client:
# cacert: /etc/open5gs/tls/ca.crt
# key: /etc/open5gs/tls/amf.key
# cert: /etc/open5gs/tls/amf.crt
# scp:
# sbi:
# - name: scp.open5gs.org
#
# o SBI Client(http://[fd69:f21d:873c:fb::1]:80)
# If prefer_ipv4 is true, http://127.0.1.10:80 is selected.
#
# sbi:
# client:
# no_tls: true
# scp:
# sbi:
# addr:
# - 127.0.1.10
# - fd69:f21d:873c:fb::1
#
# o SBI Option (Default)
# - tcp_nodelay : true
# - so_linger.l_onoff : false
#
# sbi:
# client:
# no_tls: true
# scp:
# sbi:
# addr: 127.0.1.10
# option:
# tcp_nodelay: false
# so_linger:
# l_onoff: true
# l_linger: 10
#
#
# scp:
# sbi:
# - addr: 127.0.1.10
# port: 7777
#
# <SBI Client>
#
# o SBI Client(http://127.0.0.10:7777)
# sbi:
# client:
# no_tls: true
# nrf:
# sbi:
# addr: 127.0.0.10
# port: 7777
#
# o SBI Client(https://127.0.0.10:443, https://[::1]:443) without verification
# sbi:
# client:
# no_verify: true
# key: /etc/open5gs/tls/amf.key
# cert: /etc/open5gs/tls/amf.crt

```

```

# nrf:
# sbi:
# - addr: 127.0.0.10
# - addr: ::1
#
# o SBI Client(https://nrf.open5gs.org:443)
# Use the specified certificate while verifying the server
#
# sbi:
# client:
# cacert: /etc/open5gs/tls/ca.crt
# key: /etc/open5gs/tls/amf.key
# cert: /etc/open5gs/tls/amf.crt
# nrf:
# sbi:
# - name: nrf.open5gs.org
#
# o SBI Client(http://[fd69:f21d:873c:fa::1]:80)
# If prefer_ipv4 is true, http://127.0.0.10:80 is selected.
#
# sbi:
# addr:
# - 127.0.0.10
# - fd69:f21d:873c:fa::1
#
# o SBI Option (Default)
# - tcp_nodelay : true
# - so_linger.l_onoff : false
#
# sbi:
# client:
# no_tls: true
# nrf:
# sbi:
# addr: 127.0.0.10
# option:
# tcp_nodelay: false
# so_linger:
# l_onoff: true
# l_linger: 10
#
#nrf:
# sbi:
# - addr:
# - 127.0.0.10
# - ::1
# port: 7777

#
# o Disable use of IPv4 addresses (only IPv6)
# parameter:
# no_ipv4: true
#
# o Disable use of IPv6 addresses (only IPv4)
# parameter:
# no_ipv6: true
#
# o Prefer IPv4 instead of IPv6 for establishing new GTP connections.
# parameter:
# prefer_ipv4: true
#
parameter:

#
# o Maximum Number of UE
# max:

```



```

# ue: 1024
#
# o Maximum Number of Peer(S1AP/NGAP, DIAMETER, GTP, PFCP or SBI)
# max:
# peer: 64
#
max:

#
# usrsctp:
# udp_port : 9899
#
usrsctp:

#
# o NF Instance Heartbeat (Default : 0)
# NFs will not send heart-beat timer in NFProfile
# NRF will send heart-beat timer in NFProfile
# (Default values are used, so no configuration is required)
#
# o NF Instance Heartbeat (20 seconds)
# NFs will send heart-beat timer (20 seconds) in NFProfile
# NRF can change heart-beat timer in NFProfile
#
# time:
# nf_instance:
# heartbeat: 20
#
# o Message Wait Duration (Default : 10,000 ms = 10 seconds)
# (Default values are used, so no configuration is required)
#
# o Message Wait Duration (3000 ms)
# time:
# message:
# duration: 3000
#
# o Handover Wait Duration (Default : 300 ms)
# Time to wait for AMF to send UEContextReleaseCommand
# to the source gNB after receiving HandoverNotify
# (Default values are used, so no configuration is required)
#
# o Handover Wait Duration (500ms)
# time:
# handover:
# duration: 500
#
# o Timers of 5GS mobility/session management
# time:
# t3502:
# value: 720 # 12 minutes * 60 = 720 seconds
# t3512:
# value: 3240 # 54 minutes * 60 = 3240 seconds
#
time:
t3512:
value: 540 # 9 mintues * 60 = 540 seconds

```

Anexo 4. Archivo de configuración upf.yaml

```
#
# o Set OGS_LOG_INFO to all domain level
# - If `level` is omitted, the default level is OGS_LOG_INFO)
# - If `domain` is omitted, the all domain level is set from 'level'
# (Default values are used, so no configuration is required)
#
# o Set OGS_LOG_ERROR to all domain level
# - `level` can be set with none, fatal, error, warn, info, debug, trace
# logger:
# level: error
#
# o Set OGS_LOG_DEBUG to mme/emm domain level
# logger:
# level: debug
# domain: mme,emm
#
# o Set OGS_LOG_TRACE to all domain level
# logger:
# level: trace
# domain: core,sbi,ausf,event,tlv,mem,sock
#
logger:
  file: /var/log/open5gs/upf.log

#
# <PCFP Server>
#
# o PCFP Server(127.0.0.7:8805, ::1:8805)
# upf:
# pfcf:
# - addr: 127.0.0.7
# - addr: ::1
#
# o PCFP-U Server(127.0.0.1:2152, [::1]:2152)
# upf:
# pfcf:
# name: localhost
#
# o PCFP Option (Default)
# - so_bindtodevice : NULL
#
# upf:
# pfcf:
# addr: 127.0.0.7
# option:
# so_bindtodevice: vrf-blue
#
# o Provide custom PCFP address to be advertised to SMF in PCFP association
# request/respond
# upf:
# pfcf:
# - addr: 0.0.0.0
# advertise: open5gs-smf.svc.local
#
# <GTP-U Server>>
#
# o GTP-U Server(127.0.0.7:2152, [::1]:2152)
# upf:
# gtpu:
# - addr: 127.0.0.7
# - addr: ::1
#
# o GTP-U Server(127.0.0.1:2152, [::1]:2152)
# upf:
```

```

# gtpu:
# name: localhost
#
# o User Plane IP Resource information
# upf:
# gtpu:
# - addr:
# - 127.0.0.7
# - ::1
# teid_range_indication: 4
# teid_range: 10
# network_instance: internet
# source_interface: 0
# - addr: 127.0.10.4
# teid_range_indication: 4
# teid_range: 5
# network_instance: ims
# source_interface: 1
#
# o Provide custom UPF GTP-U address to be advertised inside NGAP messages
# upf:
# gtpu:
# - addr: 10.4.128.21
# advertise: 172.24.15.30
#
# upf:
# gtpu:
# - addr: 10.4.128.21
# advertise:
# - 127.0.0.1
# - ::1
#
# upf:
# gtpu:
# - addr: 10.4.128.21
# advertise: upf1.5gc.mnc001.mcc001.3gppnetwork.org
#
# upf:
# gtpu:
# - dev: ens3
# advertise: upf1.5gc.mnc001.mcc001.3gppnetwork.org
#
# o GTP-U Option (Default)
# - so_bindtodevice : NULL
#
# upf:
# gtpu:
# addr: 127.0.0.7
# option:
# so_bindtodevice: vrf-blue
#
# <Subnet for UE network>
#
# Note that you need to setup your UE network using TUN device.
# (ogstun, ogstun2, ogstunX, ..)
#
# o IPv4 Pool
# $ sudo ip addr add 10.45.0.1/16 dev ogstun
#
# upf:
# subnet:
# addr: 10.45.0.1/16
#
# o IPv4/IPv6 Pool
# $ sudo ip addr add 10.45.0.1/16 dev ogstun
# $ sudo ip addr add 2001:db8:cafe::1/48 dev ogstun

```

```

#
# upf:
# subnet:
# - addr: 10.45.0.1/16
# - addr: 2001:db8:cafe::1/48
#
#
# o Specific DNN/APN(e.g 'ims') uses 10.46.0.1/16, 2001:db8:babe::1/48
# All other APNs use 10.45.0.1/16, 2001:db8:cafe::1/48
# $ sudo ip addr add 10.45.0.1/16 dev ogstun
# $ sudo ip addr add 10.46.0.1/16 dev ogstun
# $ sudo ip addr add 2001:db8:cafe::1/48 dev ogstun
# $ sudo ip addr add 2001:db8:babe::1/48 dev ogstun
#
# ; If the UE has unknown DNN/APN(not internet/ims), SMF/UPF will crash.
#
# upf:
# subnet:
# - addr: 10.45.0.1/16
# dnn: internet
# - addr: 2001:db8:cafe::1/48
# dnn: internet
# - addr: 10.46.0.1/16
# dnn: ims
# - addr: 2001:db8:babe::1/48
# dnn: ims
#
# o Specific DNN/APN with the FALLBACK SUBNET(10.47.0.1/16)
# ; Note that put the FALLBACK SUBNET last to avoid SMF/UPF crash.
#
# upf:
# subnet:
# - addr: 10.45.0.1/16
# dnn: internet
# - addr: 10.46.0.1/16
# dnn: ims
# - addr: 10.50.0.1/16 ## FALLBACK SUBNET
#
# o Multiple Devices (default: ogstun)
# $ sudo ip addr add 10.45.0.1/16 dev ogstun
# $ sudo ip addr add 2001:db8:cafe::1/48 dev ogstun2
# $ sudo ip addr add 10.46.0.1/16 dev ogstun3
# $ sudo ip addr add 2001:db8:babe::1/48 dev ogstun3
#
# upf:
# subnet:
# - addr: 10.45.0.1/16
# dnn: internet
# - addr: 2001:db8:cafe::1/48
# dnn: internet
# dev: ogstun2
# - addr: 10.46.0.1/16
# dnn: ims
# dev: ogstun3
# - addr: 2001:db8:babe::1/48
# dnn: ims
# dev: ogstun3
#
# <Metrics Server>
#
# o Metrics Server(http://<any address>:9090)
# upf:
# metrics:
# - addr: 0.0.0.0
# port: 9090
#

```

```

upf:
  pfcf:
    - addr: 127.0.0.7
  gtpu:
    - addr: 127.0.0.2
  subnet:
    - addr: 10.45.0.1/16
    - addr: 2001:db8:cafe::1/48
  metrics:
    - addr: 127.0.0.7
  port: 9090

#
# <PCFP Client>
#
# o PCFP Client(127.0.0.4:8805)
# smf:
# pfcf:
# addr: 127.0.0.4
#
smf:

#
# o Number of output streams per SCTP associations.
# parameter:
# sctp_streams: 30
#
# o Disable use of IPv4 addresses (only IPv6)
# parameter:
# no_ipv4: true
#
# o Disable use of IPv6 addresses (only IPv4)
# parameter:
# no_ipv6: true
#
# o Prefer IPv4 instead of IPv6 for establishing new GTP connections.
# parameter:
# prefer_ipv4: true
#
parameter:

#
# o Maximum Number of UE
# max:
# ue: 1024
#
# o Maximum Number of Peer(S1AP/NGAP, DIAMETER, GTP, PCFP or SBI)
# max:
# peer: 64
#
max:

#
# o Message Wait Duration (Default : 10,000 ms = 10 seconds)
# (Default values are used, so no configuration is required)
#
# o Message Wait Duration (3000 ms)
# time:
# message:
# duration: 3000
time:

```

Anexo 5. Preparación del entorno

Una vez que el sistema operativo esté en funcionamiento, será necesario llevar a cabo la ejecución del comando que se presenta a continuación:

```
sudo nano /etc/default/grub
```

Una vez que el archivo haya sido abierto, se procederá a eliminar el contenido de la variable `GRUB_CMDLINE_LINUX_DEFAULT`, con el propósito de reemplazarlo posteriormente por la cadena “quiet intel_pstate=disable processor.max_cstate=1 intel_idle.max_cstate=0 idle=poll”.

Para garantizar la implementación de los cambios, se ejecutará el siguiente comando: *sudo update-grub*. Una vez concluida la ejecución de este comando, se verificará la correcta finalización del proceso a través del mensaje “hecho”.

A continuación, será necesario efectuar la modificación en el archivo `blacklist.conf` mediante el uso del siguiente comando:

```
sudo nano /etc/modprobe.d/blacklist.conf
```

Una vez el archivo haya sido abierto, será necesario agregar la línea “blacklist intel_powerclamp” al final del mismo. En caso de que el archivo “blacklist.conf” no exista, será creado automáticamente sin inconvenientes.

Luego de esta acción, procederemos a instalar `i7z` mediante la ejecución del siguiente comando:

```
sudo apt-get install i7z
```

A continuación, procederemos a instalar `cpufrequtils` utilizando el siguiente comando:

```
sudo apt-get install -y cpufrequtils
```

Una vez que la instalación esté completa, procederemos a realizar modificaciones en el archivo `cpufrequtils` a través del siguiente comando:

```
sudo nano /etc/default/cpufrequtils
```

Una vez que el archivo esté abierto, será necesario agregar la siguiente línea: `GOVERNOR=“performance”`.

A continuación, procederemos a reemplazar el texto `GOVERNOR=“ondemand”` por `GOVERNOR=“performance”` en el archivo “`/etc/init.d/cpufrequtils`”. Posteriormente, deberán ejecutarse los siguientes comandos para que los cambios se apliquen en el sistema:

```
sudo systemctl disable ondemand
sudo /etc/init.d/cpufrequtils restart
```

Luego, será posible observar la configuración mediante la ejecución de `cpufreq-info` en la terminal.

Finalmente, en esta etapa ha sido necesario instalar un kernel de baja latencia. Para lograrlo, se ha empleado el siguiente comando:

```
sudo apt-get install linux-image-lowlatency linux-headers-lowlatency
```

Luego de esto, será necesario proceder con la instalación del controlador utilizando el siguiente comando:

```
sudo apt-get install cpufreqd
```

Para asegurar el funcionamiento adecuado de los componentes del sistema que serán utilizados a partir de este punto, es esencial iniciar Ubuntu a través del kernel de baja latencia. Con este fin, se procederá a abrir el archivo de configuración del gestor de arranque (`grub`) mediante el siguiente comando:

```
sudo nano /etc/default/grub
```

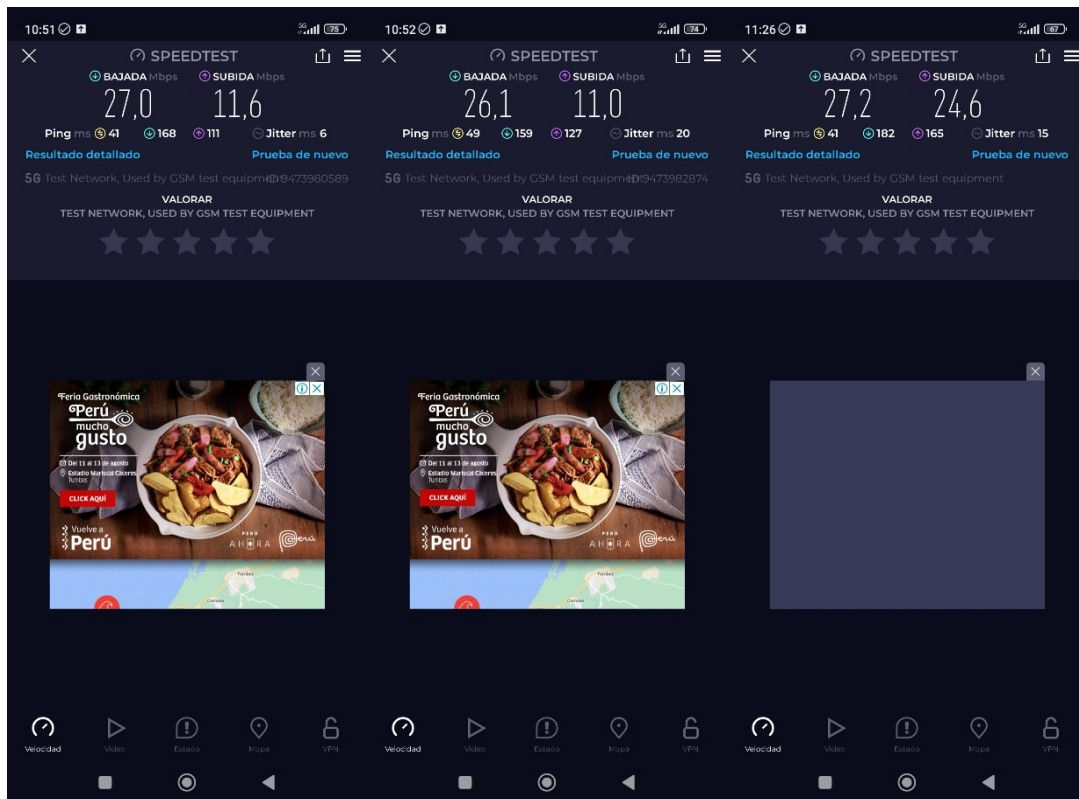
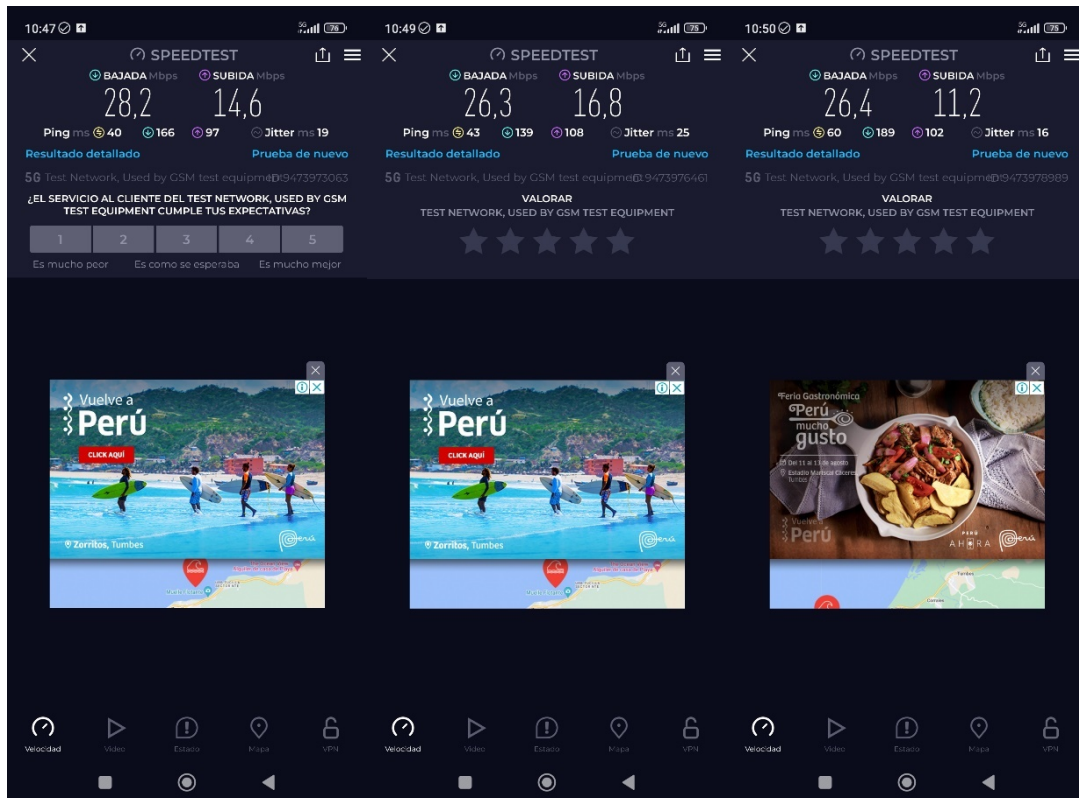
A continuación, se procederá a configurar las líneas `GRUB_TIMEOUT_STYLE` como “menú” y `GRUB_TIMEOUT` en 10.

Una vez realizados estos ajustes, se guardarán las modificaciones efectuadas en el archivo y se ejecutará el siguiente comando:

```
sudo update-grub
```

Después de haber actualizado la configuración del GRUB, procederemos a reiniciar el sistema. Al momento de visualizar el selector de arranque GRUB, será esencial acceder a la opción "Opciones avanzadas para Ubuntu". A continuación, se seleccionará el kernel de baja latencia más reciente dentro de ese menú.

Anexo 6. Muestras de la medición de Thougput



Anexo 7. Muestras de la medición de potencia



Anexo 8. Certificado de Traducción del Resumen

Loja, 08 de febrero de 2024

Lic. Mónica Guicela Jiménez Abad

DOCENTE DE LA UNIDAD EDUCATIVA LUIS TUFÍÑO

CERTIFICO:

Que el documento aquí compuesto es fiel traducción del idioma español al idioma inglés del resumen del Trabajo de Integración Curricular: **“Diseño e implementación de un prototipo de arquitectura 5G NR para servicios de voz y datos basado en SDR (Software Defined Radio) y software Open Source”**, autoría de **Juan Pablo Rivas Mora** con **CI: 1104242332** de la carrera de Ingeniería en Telecomunicaciones, de la Universidad Nacional de Loja.

Lo certifica en honor a la verdad y autorizo al interesado hacer uso del presente en lo que a sus intereses convenga.

Atentamente,



MONICA GUICELA JIMENEZ ABAD

DOCENTE DE LA UNIDAD EDUCATIVA LUIS TUFÍÑO

REGISTRO SENECYT N°: 1008-2016-1695984