UNIVERSIDADE FEDERAL DE SANTA CATARINA
CENTRO TECNOLÓGICO, DE CIÊNCIAS EXATAS E EDUCAÇÃO
DEPARTAMENTO DE ENG. DE CONTROLE, AUTOMAÇÃO E COMPUTAÇÃO
CURSO DE GRADUAÇÃO EM ENGENHARIA DE CONTROLE E AUTOMAÇÃO

Douglas Liao

# Implementação de QoS em rede 5G através do Network Slicing

Blumenau
2023

Douglas Liao

# Implementação de QoS em rede 5G através do Network Slicing

Trabalho de Conclusão de Curso de Graduação em Engenharia de Controle e Automação do Centro Tecnológico, de Ciências Exatas e Educação da Universidade Federal de Santa Catarina como requisito para a obtenção do título de Engenheiro de Controle e Automação.
Orientador: Prof. Dr. Adão Boava.
Coorientador: Me. Christian Mailer

Blumenau
2023

Ficha de identificação da obra elaborada pelo autor,
através do Programa de Geração Automática da Biblioteca Universitária da UFSC.

Liao, Douglas
Implementação de QoS em rede 5G através do Network Slicing / Douglas Liao ; orientador, Adão Boava, coorientador, Christian Mailer, 2023.
75 p.

Trabalho de Conclusão de Curso (graduação) - Universidade Federal de Santa Catarina, Campus Blumenau, Graduação em Engenharia de Controle e Automação, Blumenau, 2023.

Inclui referências.

1. Engenharia de Controle e Automação. 2. 5G. 3. QoS. 4. Network Slicing . 5. Latência. I. Boava, Adão. II. Mailer, Christian. III. Universidade Federal de Santa Catarina. Graduação em Engenharia de Controle e Automação. IV. Título.

Douglas Liao

**Implementação de QoS em rede 5G através do Network Slicing**

Este Trabalho de Conclusão de Curso foi julgado adequado para obtenção do Título de "Engenheiro de Controle e Automação" e aprovado em sua forma final pelo Curso de Graduação em Engenharia de Controle e Automação.

Blumenau, 21 de Junho de 2023.

**Banca Examinadora:**


Documento assinado digitalmente
Adao Boava
Data: 12/07/2023 13:32:37-0300
CPF: ***.640.969-**
Verifique as assinaturas em https://v.ufsc.br

Prof. Dr. Adão Boava
Universidade Federal de Santa Catarina


Documento assinado digitalmente
Ciro Andre Pitz
Data: 12/07/2023 06:44:10-0300
CPF: ***.728.089-**
Verifique as assinaturas em https://v.ufsc.br

Prof. Dr. Ciro André Pitz
Universidade Federal de Santa Catarina


Documento assinado digitalmente
Mauri Ferrandin
Data: 12/07/2023 08:34:05-0300
CPF: ***.580.869-**
Verifique as assinaturas em https://v.ufsc.br

Prof. Dr. Mauri Ferrandin
Universidade Federal de Santa Catarina

Este trabalho é dedicado a minha família e ao meu futuro.

## AGRADECIMENTOS

Agradeço a minha família que sempre me apoiaram e me deram suporte para buscar um caminho que eu escolhesse. Também agradeço aos meus amigos que eu encontrei ao longo dessa caminhada, me deram suporte e compartilharam os seus conhecimentos. Por fim, agradeço a todos que me ajudaram nesse último trabalho.

Resiliência é continuar numa constante transformação diante de todas as pressões presentes. É o sentido *master* da ressignificação. (Nilton PEDREIRA, 2016)

## RESUMO

O 5G é a geração de redes móveis que promete revolucionar as comunicações sem fio. Entre os principais pontos dessa tecnologia, destaca-se a Qualidade de Serviço (QoS), que permite oferecer diferentes níveis de desempenho para atender às demandas de uma variedade de aplicativos e serviços. Neste trabalho, é realizado uma análise aprofundada do QoS em redes 5G por meio do *network slicing*. O componente principal de uma rede 5G é o seu CORE, responsável por gerenciar a conexão dos dispositivos de usuário e controlar o fluxo de informações. Para a implementação do nosso estudo, é aplicado uma plataforma CORE baseada em contêineres e disponibilizada em uma máquina virtual. É utilizado o pacote Aether-in-a-box para executar as funções necessárias na rede 5G. Configuramos as informações dos dispositivos de usuário no CORE por meio de uma interface web, adaptando algoritmos de simulação de acesso presentes no pacote. Além disso, é ajustado a interface web para permitir a aplicação do *network slicing* em parâmetros de QoS configurados pelo usuário. É realizado uma série de testes para avaliar a capacidade de gerenciamento dos contêineres, bem como o registro dos dispositivos de usuário na rede 5G. Ao total, é registrado 30 dispositivos para avaliar o *throughput* e o *network slicing*. Os resultados demonstraram que o *throughput* se manteve dentro dos parâmetros e quanto a latência também se manteve dentro dos parâmetros definidos pela QoS, evidenciando a eficácia da separação das fatias de rede com alta, média e baixa prioridade.



**Palavras-chave**: 5G. QoS. Network Slicing. Latência. Aether-in-a-box.

**ABSTRACT**

The 5G is the generation of mobile networks that promises to revolutionize wireless communications. Among the key points of this technology, Quality of Service (QoS) stands out, allowing different levels of performance to meet the demands of a variety of applications and services. In this work, we propose an in-depth analysis of QoS through network slicing. The main component of a 5G network is its CORE, responsible for managing user device connections and controlling the flow of information. For the implementation of our study, we applied a container-based CORE platform provided in a virtual machine. We used the Aether-in-a-box package to perform the necessary functions in the 5G network. We configured user device information in the CORE through a web interface, adapting simulation access algorithms present in the package. Additionally, we adjusted the web interface to allow the application of network slicing on user-configured QoS parameters. We conducted a series of tests to evaluate the container management capability, as well as the registration of user devices in the 5G network. In total, we registered 30 devices to evaluate throughput and network slicing. The results demonstrated that the throughput remained within parameters, and the latency also remained within the parameters defined by QoS, highlighting the effectiveness of separating network slices with high, medium, and low priority.

**Keywords**: 5G. QoS. Network Slicing. Latency. Aether-in-a-box

## LISTA DE FIGURAS

## LISTA DE QUADROS

## LISTA DE TABELAS

## LISTA DE ABREVIATURAS E SIGLAS

| | |
|---|---|
| 1G | *1st Generation* |
| 2G | *2st Generation* |
| 3G | *3st Generation* |
| 3GPP | *3rd Generation Partnership Program* |
| 4G | *4st Generation* |
| 5G | *5st Generation* |
| AMF | *Access and Mobility Management Function* |
| AMPS | *Advanced Mobile Phone System* |
| API | *Application Programming Interface* |
| ARIB | *Association of Radio Industries and Companies* |
| ARP | *Address Resolution Protocol* |
| ARPANET | *Advanced Research Projects Agency Network* |
| ATIS | *Alliance for Telecommunications Industry Solutions* |
| AUSF | *Authentication Server Function* |
| BTS | *Base Transceiver Station* |
| CCSA | *China Communications Standards Association* |
| CDMA | *Code Division Multiple Access* |
| CNCF | *Cloud Native Computing Foundation* |
| CRN | *Cognitive Radio Network* |
| DL | *Downlink* |
| eMBB | *Enhanced Mobile Broad-Band* |
| EPS | *Evolved Packet System* |
| ETSI | *European Telecommunications Standards Institute* |
| FDMA | *Frequency Division Multiple Access* |
| GBR | *Guaranteed Bit Rate* |
| GSM | *Global System for Mobile Communications* |
| Hz | *Hertz* |
| IoT | *Internet of Things* |
| ISO | *International Standards Organization* |
| LTE | *Long Term Evolution* |
| MIMO | *(Multiple-Input Multiple-Output* |
| mMTC | *Massive Machine Type Communication* |
| NF | *Network Function* |
| NFV | *Network Function Virtualization* |
| NMT | *Nordic Mobile Telecommunications* |
| NRF | *Network Repository Function* |
| NSSF | *Network Slice Selection Function* |

| | |
|---|---|
| OFDM | *Orthogonal Frequency Division Multiplexing* |
| OFDMA | *Orthogonal Frequency Division Multiplexing* |
| ONF | *Open Networking Foundation* |
| OSI | *Open Systems Interconnection* |
| PCF | *Policy Control Function* |
| PDB | *Packet Delay and Budget* |
| PDU | *Protocol Data Unit* |
| QFI | *QoS Flow Identifier* |
| QoS | *Quality of Service* |
| RAN | *Radio Access Network* |
| RFSP | *Radio Frequency Selection Priority* |
| ROC | *Runtime Operational Control* |
| RRH | *Remote Radio Head* |
| SC-FDE | *Single-Carrier Frequency Domain Equalization* |
| SCTP | *Stream Control Transmission Protocol* |
| SCTPLB | *Stream Control Transmission Protocol Load Balancer* |
| SDN | *Software Defined Network* |
| SD-WAN | *Software-Defined Wide Area Network* |
| SMF | *Session Management Function* |
| SMS | *Short Message Service* |
| S-NSSAI | *Single Network Slice Selection Assistance Information* |
| SUCI | *Subscription Concealed Identifier* |
| SUPI | *Subscription Permanent Identifier* |
| TACS | *Total Access Communication System* |
| TCP | *Transmission Control Protocol* |
| TDMA | *Time Division Multiple Access* |
| TSDSI | *Telecommunications Standards Development Society, India* |
| TSG | *Technical Specification Group* |
| TTA | *Telecommunications Technology Association* |
| TTC | *Toronto Transit Commission* |
| UDM | *Unified Data Management* |
| UDR | *Unified Data Repository* |
| UE | *User Equipments* |
| UL | *Uplink* |
| UP | *User Plane* |
| UPF | *User Plane Function* |
| URLLC | *Ultra Reliable Low Latency Communication* |
| VM | *Virtual Machine* |
| WCDMA | *Wide Band Code Divison Multiple Access* |

**SUMÁRIO**

# 1 INTRODUÇÃO

## 1.1 PROBLEMATIZAÇÃO

Dentre os diversos desafios enfrentados pela implantação do 5G (*5st Generation*), um dos aspectos é a implementação eficiente da QoS (*Quality of Service*) em um cenário que envolve redes privadas, o conceito de *slicing* e diferentes meios de serviço definidos pelo 3GPP (*3rd Generation Partnership Program*), como eMBB (*Enhanced Mobile Broad-Band*), mMTC (*Massive Machine Type Communication*) e URLLC (*Ultra Reliable Low Latency Communication*) (3GPP, 2023).

Uma das principais problemáticas é como garantir a QoS em redes 5G que adotam o conceito de *slicing*. O *slicing* permite a criação de várias fatias de rede independentes, cada uma adaptada a um conjunto específico de requisitos de aplicativos e a usuários. No entanto, quando se trata de redes privadas, nas quais múltiplos usuários compartilham os recursos de uma fatia específica, a alocação adequada de recursos e a garantia de QoS para cada usuário individualmente podem se tornar um desafio significativo. Além disso, ao considerar uma topologia de rede que inclua 30 usuários, surgem problemas de qualidade de serviço.

Os 3 serviços definidos pelo 3GPP - eMBB, mMTC e URLLC - apresentam requisitos distintos. O eMBB visa fornecer serviços de alta largura de banda para conectividade sem fio, o mMTC tem como objetivo prover comunicação confiável para bilhões de sensores e dispositivos de monitoramento, enquanto o URLLC busca oferecer Comunicação ultra-confiável e de baixa latência para requisitos críticos (NETWORKS, 2023).

Essas características relacionadas a 5G, *slicing*, redes privadas, 3GPP e os diferentes meios de serviço eMBB, mMTC e URLLC, destacam a importância de explorar estratégias eficientes de QoS para garantir um desempenho ideal e uma experiência de usuário aprimorada em ambientes 5G cada vez mais complexos.

Assim, ao longo desse trabalho, será utilizada a topologia com 30 dispositivos conectados ao uma rede de acesso ao rádio como base para as implementações e aplicações desenvolvidas.

## 1.2 OBJETIVO GERAL

O objetivo principal deste trabalho é estudar e analisar o QoS por meio do conceito de *Network Slicing* por meio da rede 5G.

## 1.3 OBJETIVOS ESPECÍFICOS

Os objetivos específicos desse trabalho são listados como:

- Compreender de maneira profunda as redes 5G, incluindo seus componentes, sistemas e serviços;
- Realizar a configuração, projeto e simulação de uma topologia de rede 5G, abrangendo equipamentos de usuário e a rede de acesso ao rádio.;
- Analisar as técnicas de QoS aplicadas no contexto do sistema 5G;
- Realizar o monitoramento dos recursos do sistema 5G, a fim de avaliar seu desempenho e eficiência;
- Aplicar o conceito de *Network Slicing* nas redes 5G
- Trabalhar com um novo simulador para redes 5G, o Aether-in-a-box

## 1.4 ESTRUTURA DO DOCUMENTO

Este documento está estruturado em cinco capítulos. No capítulo 2, serão abordados os conceitos essenciais para compreender o 5G, bem como as principais referências teóricas utilizadas na metodologia, incluindo máquinas virtuais e a descrição do software. No capítulo 3, será detalhada a metodologia adotada, descrevendo passo a passo como foram executadas as etapas e destacando o processo de coleta de dados. Por fim, o quarto capítulo apresentará os resultados obtidos com base na aplicação do conceito de *network slicing*.

# 2 REVISÃO DE LITERATURA

Neste capítulo, serão apresentados os conceitos principais das tecnologias abordadas neste trabalho.

Inicialmente, serão discutidos os fundamentos básicos de redes e telecomunicações, seguidos pela história e evolução das gerações de redes móveis. Será dada uma atenção ao 5G de redes móveis e à arquitetura utilizada nessa tecnologia. Além disso, serão abordados tópicos importantes, como *network slicing*, QoS no contexto da arquitetura 5G.

Também será explorado o conceito de SD-WAN (*Software-Defined Wide Area Network*) e suas aplicações relevantes. Serão apresentadas ferramentas utilizadas ao longo do trabalho para simulação e análise das redes.

Essa reformulação permite uma organização mais clara e direta das informações, destacando os principais tópicos abordados no capítulo

## 2.1 CAMADAS DE PROTOCOLO

O modelo OSI (*Open Systems Interconnection*) é um modelo de referência baseado em uma proposta da ISO (*International Standards Organization*) para padronização dos protocolos utilizados em redes. Ele consiste em sete camadas e foi projetado seguindo os seguintes princípios de (TANENBAUM, 2002):

- Criação de uma camada sempre que uma nova abstração é necessária.
- Cada camada possui uma função bem definida.
- Seleção das funções de cada camada com o objetivo de estabelecer protocolos internacionalmente padronizados.
- Os limites entre as camadas devem ser escolhidos de forma a minimizar o fluxo de informações através das interfaces.
- O número de camadas deve ser grande o suficiente para evitar a necessidade de agrupar funções distintas na mesma camada por necessidade, mas também deve ser pequeno o suficiente para evitar que a arquitetura se torne complicada.

O modelo OSI foi desenvolvido para permitir a conexão de sistemas abertos, ou seja, sistemas que podem se comunicar com outros sistemas.

## 2.2 3GPP

O Projeto de Parceria da 3ª Geração (3GPP) é uma colaboração entre sete organizações de desenvolvimento de padrões de telecomunicações em todo o mundo. Essas organizações incluem a ARIB (*Association of Radio Industries and Companies*) no Japão, a ATIS (*Alliance for Telecommunications Industry Solutions*) na América do Norte,

a CCSA (*China Communications Standards Association*), o ETSI (*European Telecommunications Standards Institute*), a TSDSI (*Telecommunications Standards Development Society, India*), a TTA (*Telecommunications Technology Association*) da Coreia do Sul e a TTC (*Toronto Transit Commission*) no Canadá.

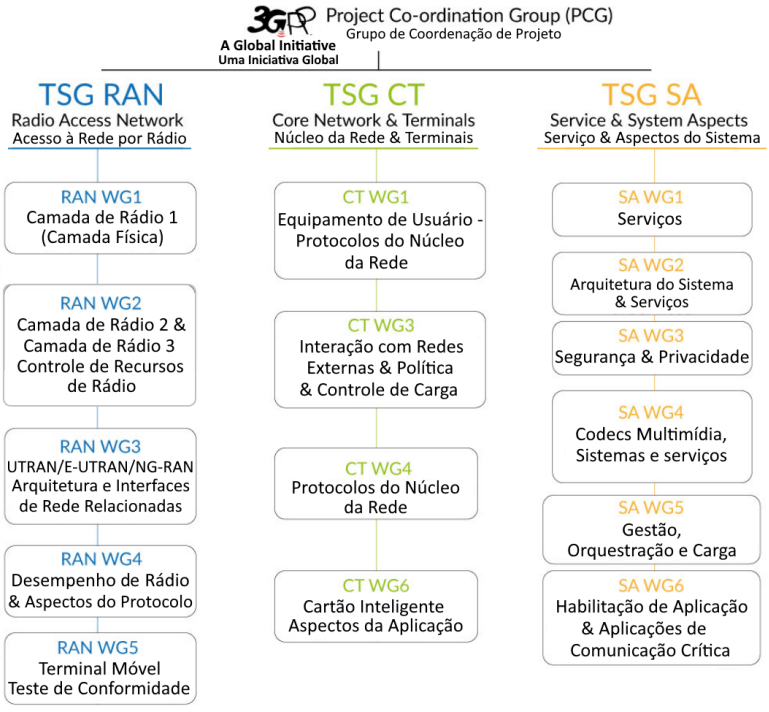
Essas organizações, conhecidas como "parceiros organizacionais", fornecem um ambiente estável para a produção de relatórios e especificações que definem as tecnologias 3GPP, que por sua vez é responsável por definir as especificações para um sistema móvel completo, abrangendo terminais, redes de acesso de rádio, redes principais e partes da rede de serviços.

As especificações do 3GPP são estruturadas em diferentes versões e as discussões geralmente se referem às funcionalidades de uma versão específica. É importante destacar que todas as novas versões são compatíveis com as versões anteriores, garantindo a continuidade e interoperabilidade dos sistemas.

O 3GPP fornece os resultados do seu trabalho aos órgãos de padronização em cada região do mundo, que podem adotar essas especificações como normas ou recomendações formais em suas respectivas regiões. Isso permite a adoção consistente das tecnologias 3GPP em todo o mundo.

A figura 1 abaixo ilustra os três grupos de especificações técnicas TSG (*Technical Specification Group*) do 3GPP, que desempenham um papel no desenvolvimento e definição das tecnologias 3GPP.

Figura 1 – Os três TSGs do 3GPP




Fonte: (KRAUS, 2021)

## 2.3 PROTOCOLO TCP/IP

O ARPANET ( *Advanced Research Projects Agency Network*), uma rede de pesquisa financiada pelo Departamento de Defesa dos Estados Unidos, foi um marco importante na história das redes de computadores. Inicialmente, conectava universidades e instituições governamentais por meio de linhas telefônicas alugadas. Com o tempo, a adição de redes via satélite e rádio apresentou desafios de interoperabilidade com os protocolos existentes, levando à necessidade de uma nova arquitetura de referência (TANENBAUM, 2002).

Essa arquitetura resultou no Modelo de Referência TCP/IP, nomeado após seus principais protocolos. Criado por *Vinton Cerf* e *Robert Kahn*, o TCP/IP foi descrito em 1974 e posteriormente refinado e padronizado pela comunidade da *internet*. O modelo foi projetado para permitir a conexão contínua de múltiplas redes, oferecendo uma base sólida para o funcionamento da *internet* como a conhecemos hoje (TANENBAUM, 2002).

Além de promover a conectividade entre redes, o TCP/IP também foi desenvolvido com foco na resiliência. O Departamento de Defesa tinha a preocupação de que a rede pudesse sobreviver a possíveis ataques e falhas de *hardware* sem interromper as comunicações em andamento. Dessa forma, mesmo que parte da infraestrutura fosse danificada, as conexões permaneceriam intactas, desde que as máquinas de origem e destino estivessem funcionando.

Uma característica importante do TCP/IP é sua flexibilidade para atender a diferentes requisitos de aplicativos. Desde a transferência de arquivos até a transmissão de voz em tempo real, o modelo foi projetado para acomodar uma ampla variedade de necessidades, tornando-se a base para a comunicação eficiente na *internet*.

## 2.4 HISTÓRIA DAS REDES

O sistema de telefonia móvel de primeira geração 1G foi introduzido na década de 80, com foco na comunicação por voz. Utilizando modulação analógica e a técnica FDMA (Acesso Múltiplo por Divisão de Frequência), permitia que vários usuários acessassem diferentes frequências. Foram desenvolvidos diversos sistemas celulares nessa época, como o NMT, TACSe AMPS.

Um aspecto importante do 1G era o formato hexagonal das células, que influenciava o planejamento das frequências do sistema. Esse formato foi escolhido devido à transmissão omnidirecional, com as Estações Base (BTS) posicionadas no centro da célula, formando uma área de cobertura semelhante a um círculo. No entanto, o formato circular apresentava sobreposição entre células ou áreas sem cobertura quando um grupo delas era implementado. Alguns polígonos regulares, como o hexágono, triângulo e quadrado, não tinham essa desvantagem. Portanto, o formato hexagonal foi escolhido por se assemelhar mais a um círculo. Além disso, foram adotadas técnicas de reutilização de frequência para expandir a área de cobertura do sistema. Isso permitia que as frequências usadas

em uma BTS fossem simultaneamente utilizadas em outra BTS, desde que estivessem suficientemente distantes para evitar ou minimizar interferências. O sistema 1G já incluía recursos de *roaming* e *handoff*, que possibilitavam a comunicação móvel fora do serviço local e a transição entre células, respectivamente.

O emprego de tecnologia digital em sistemas celulares permitiu um avanço considerável em termos de desempenho e capacidade da rede, o que culminou no desenvolvimento do sistema de telefonia móvel de segunda geração (2G). Essa evolução permitiu aumentar a qualidade das transmissões de voz, a segurança e a capacidade do sistema. Além disso, no 2G, foi implementado o serviço de SMS (*Short Message Service*). Em redes de segunda geração, o controle de acesso aos recursos da rede é realizado tanto no tempo quanto na frequência, por meio das técnicas de TDMA (*Time Division Multiple Access*) e divisão na frequência FDMA (*Frequency Division Multiple Access*), respectivamente. Entre os padrões 2G, destacam-se o padrão europeu GSM (*Global System for Mobile Communications*) e os padrões americanos IS-54, IS-136 e IS-95. O último deles, também conhecido como CDMA-One, utiliza a técnica CDMA (*Code Division Multiple Access*), que aplica códigos pseudo-aleatórios para separar as informações de diversos usuários regendo o acesso ao meio. Adicionalmente, desenvolveu-se um mecanismo de controle de potência, a fim de mitigar as interferências entre usuários.

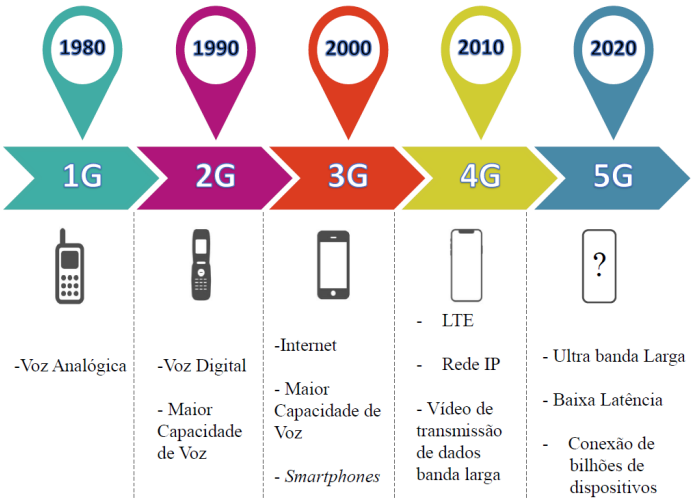
O surgimento do sistema de telefonia móvel de terceira geração (3G) contribuiu significativamente para a popularização da *internet* e dos serviços baseados em pacotes. Nas redes de terceira geração, o múltiplo acesso aos recursos da rede é realizado por meio da técnica WCDMA (*Wide Band Code Divison Multiple Access*). A diferença crucial do WCDMA em relação ao CDMA está na utilização de múltiplas portadoras de banda larga. Para tornar o mercado do 3G global, ao contrário do que aconteceu no 2G, onde cada continente utilizava um padrão celular diferente, criou-se a organização padronizadora 3GPP. Essa organização fornece as especificações de sistemas de comunicações móveis, em forma de *Releases*, atuando inclusive nas gerações de comunicações móveis sucessoras. O uso de novas técnicas de transmissão possibilitou a evolução dos sistemas celulares, visando o aumento de capacidade e da vazão de dados. Entre essas técnicas, cita-se o uso de MIMO (*(Multiple-Input Multiple-Output*), OFDM (*Orthogonal Frequency Division Multiplexing*) e agregação de portadoras, as quais são empregadas em sistemas de telefonia móvel de quarta geração (4G).

O 4G surgiu para suprir a crescente demanda por elevadas taxas de transmissões e alta capacidade. O padrão de comunicações móveis 4G ficou conhecido como LTE (*Long Term Evolution*). Nas redes de quarta geração, utiliza-se no DL (*Downlink*) a técnica OFDM, na qual divide-se a largura de banda total do sinal em múltiplas sub-bandas, espaçadas de 15 kHz, acrescentando robustez contra desvanecimentos seletivos causados por múltiplos percursos. O múltiplo acesso ao meio é realizado por meio da combinação das técnicas OFDMA (*Orthogonal Frequency Division Multiplexing*) e TDMA. No UL (*Uplink*),

utiliza-se a forma de onda SC-FDE (*Single-Carrier Frequency Domain Equalization*) e TDMA/FDMA para controlar o acesso ao meio. (PEREIRA, 2020)

A Figura 2 apresenta um resumo das evoluções das redes de comunicações móveis com suas principais características. Ao longo da evolução das redes móveis, houve um aumento da complexidade, uma vez que as novas técnicas de comunicação tiveram que atender a novos pré-requisitos desafiadores. As redes móveis de quinta geração possibilitam novas funcionalidades, por meio da melhoria de técnicas utilizadas nas gerações anteriores, bem como novas técnicas e abordagens aplicadas a novos cenários de uso da rede móvel.

Figura 2 – Evolução dos sistemas de comunicações móveis




Fonte: (PEREIRA, 2020)

## 2.5 5G E SEUS OBJETIVOS

As redes 5G representam uma evolução em relação às redes LTE, proporcionando maiores taxas de transmissão de dados e menor latência fim-a-fim. Essas características são altamente desejadas devido ao crescente número de dispositivos multimídia conectados, e essa tendência continuará a se expandir nos próximos anos. A Tabela 1 apresenta alguns dos principais requisitos dessa nova tecnologia, que buscam enfrentar esses desafios de forma eficaz.

No entanto, o 5G não é apenas uma progressão natural das gerações anteriores; ele se diferencia delas em diversos aspectos. Um dos mais significativos é a mudança de paradigma, em que as operadoras não apenas direcionam seus esforços aos usuários finais, mas também passam a ter as indústrias como seus principais clientes. Essa transformação não se limita apenas à esfera tecnológica, mas também afeta o modelo de negócios (SILVA, 2022).

Tabela 1 – Requisitos das redes 5G.

| **Métrica** | **Requisito** |
|---|---|
| Pico da taxa de dados | Até 20 Gb/s DL, até 10 Gb/s UL |
| Taxas de transmissão médias observadas | Até 100Mb/s DL, até 50 Mb/s UL |
| Eficiência espectral | Até 30 bits/s Hz DL, até 15 bits/s Hz DL |
| Densidade de conexão | Até 1 milhão de equipamentos/$km^2$ |
| Vida de bateria dos equipamentos | Mais que 10 anos |
| Mobilidade | Até 500 km/h |
| Latência de dados do usuário | 1ms para casos de uso na indústria, 4ms para MBB |
| Confiabilidade | Pelo menos 99,999% |

Fonte: (SILVA, 2022)

Outra distinção notável entre o 4G e o 5G é a criação de novas categorias de serviços que visam atender aos diferentes cenários, aplicações e casos de uso. As três principais são as seguintes (SILVA, 2022):

- eMBB (*Enhanced Mobile Broad-Band*): direcionada aos centros metropolitanos densamente povoados, essa categoria visa fornecer velocidades de DL de até 1 Gbit/s.
- mMTC (*Massive Machine Type Communication*): habilita a comunicação entre máquinas e aplicações de IoT (*Internet of Things*), atendendo às demandas de uma nova geração de dispositivos sem fio, sem comprometer outras classes de serviço.
- URLLC (*Ultra Reliable Low Latency Communication*): projetada para atender às necessidades de comunicação crítica, priorizando a velocidade de resposta com uma latência fim-a-fim de 1 ms ou menos.

Em suma, o 5G representa um avanço significativo no campo das comunicações sem fio, trazendo benefícios como velocidades mais altas, menor latência e capacidade de atender a uma variedade de necessidades e casos de uso.

## 2.6 ARQUITETURA 5G

A arquitetura da rede 5G é composta por diferentes propostas, cada uma com seu próprio conjunto de características (ANDRÉ QUEIROZ DÉBORA PINA, 2016). A seguir, são descritas as cinco principais camadas da arquitetura 5G:

- Arquitetura de duas camadas: As arquiteturas de duas camadas na rede 5G consistem em uma macro célula na camada superior, que abrange células menores, como femto, pico, micro e macro, na camada inferior. A macro célula é responsável por supervisionar e controlar as células menores.

- Arquitetura baseada em redes de rádio cognitivo: a utilização da CRN (*Cognitive Radio Network*) permite uma arquitetura de múltiplas camadas, como mencionado anteriormente, reduzindo a interferência entre as células e minimizando o consumo de energia.
- Arquitetura de comunicação D2D: tecnologia de comunicação "dispositivo para dispositivo"ou D2D (*device-to-device*) permite que dispositivos móveis próximos se comuniquem diretamente, sem a necessidade de envolver uma estação base.
- Arquitetura baseada na nuvem: A arquitetura atual da rede de acesso por rádio (RAN) é composta por estações que incluem antenas, unidades de rádio (RRH (*Remote Radio Head*)) e unidades de processamento de banda (BBU), que podem estar localizadas a uma distância de até 40 km dos RRH.
- Arquitetura de energia eficiente: A arquitetura 5G também enfoca a eficiência energética, visando reduzir o consumo de energia dos dispositivos e infraestrutura de rede. Isso é especialmente importante considerando o crescente número de dispositivos conectados e a necessidade de uma rede sustentável (ANDRÉ QUEIROZ DÉBORA PINA, 2016).

### 2.6.1 Core

A arquitetura central das redes 5G deve ser capaz de suportar a nova abordagem baseada em serviços, permitindo a construção de redes modulares e fornecendo uma experiência consistente para usuários de redes 3GPP e não 3GPP. Além disso, deve ser adaptável a tecnologias em nuvem, computação de borda e comunicação máquina-máquina, possibilitando a utilização de carros autônomos e linhas de montagem robotizadas, por exemplo.

No 5GC, são utilizadas interfaces baseadas em serviços, em vez de interfaces ponto-a-ponto, para interconectar as funções de rede. Cada interação entre as funções de rede envolve uma função atuando como consumidora de serviço e outra como produtora. Isso representa uma grande diferença em relação às gerações anteriores e suas arquiteturas tradicionais, onde as interações eram diferentes.

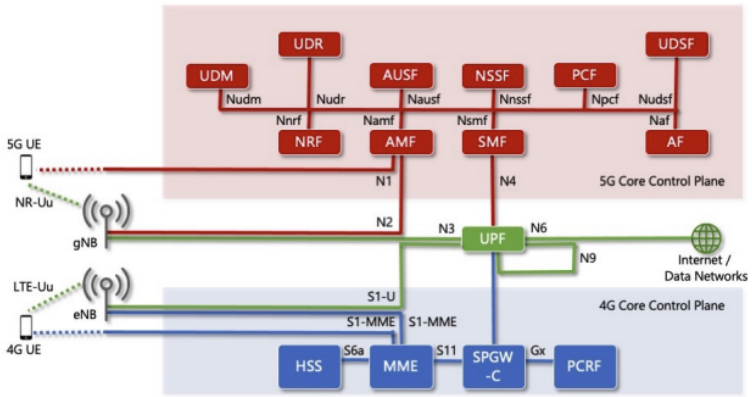
No *Aether-in-a-box*, é utilizado o *SD-Core*, que por sua vez é um núcleo móvel desagregado 4G/5G otimizado para implantação em nuvem pública em conjunto com nuvens distribuídas de borda, sendo ideal para redes 5G de operadoras e empresas privadas. Ele expõe interfaces padrão do 3GPP, permitindo o uso do *SD-Core* como um núcleo móvel convencional. Também está disponível pré-integrado com um adaptador (parte do subsistema *Aether ROC (Runtime Operational Control)*) para aqueles que o implementam como uma solução de núcleo móvel como serviço. O *SD-Core* é um componente integral do Aether, a plataforma de borda conectada 5G da ONF (*Open Networking Foundation*) para conectividade móvel privada e serviços de nuvem de borda. Pode ser implantado

rapidamente pré-integrado com o Aether, como um núcleo móvel 5G/4G autônomo ou como componentes de controle e plano de dados (UPF) integrados em soluções customizadas. Essa ampla versatilidade torna a plataforma *SD-Core* de código aberto ideal para a maior variedade de casos de uso e cenários de implantação (FOUNDATION, 2020).

As funções de rede consumidoras podem localizar e contatar funções produtoras de serviços por meio do *Service Discovery*. O registro dessas interações é realizado pela Função do Repositório de Rede (NRF), que é responsável por manter um registro das funções produtoras e consumidoras de serviços. Além disso, as funções de rede podem interagir por meio de inscrição e notificação de serviços, permitindo que as funções produtoras enviem notificações aos consumidores quando critérios pré-estabelecidos forem atingidos.

Embora existam várias funções de rede no núcleo 5G, as funções utilizadas durante o processo de simulação, além de outras relevantes para este trabalho, serão descritas a seguir. A Figura a seguir mostra as funções de rede essenciais consideradas.

Figura 3 – Funções essenciais na rede 5G




Fonte: (FOUNDATION, 2020)

### *2.6.1.1 AMF*

A AMF (*Access and Mobility Management Function*) é responsável por interagir com a rede de acesso e os dispositivos UE (*User Equipments*) através das interfaces N1 e N2. Além disso, estabelece conexões de sinalização criptografadas para registro, autenticação e movimentação dos UE entre células de rádio. Além disso, a AMF gerencia a transmissão de mensagens de sinalização, SMS e serviços de localização entre os dispositivos e outras Funções de Rede.

Também possui funcionalidades de segurança e autorização, garantindo a integridade das comunicações. Por fim, permite também alcançar e ativar equipamentos que estão em modo ocupado (STEFAN ROMMER PETER HEDMAN, 2019).

#### *2.6.1.2 SMF*

A funcionalidade de Gerenciamento de Sessão do sistema 5G é responsável pelo estabelecimento e gerenciamento da conectividade do dispositivo UE com as redes de dados. O SMF (*Session Management Function*) é a função de controle que gerencia as sessões dos usuários, incluindo o estabelecimento, modificação e encerramento das sessões, e também pode alocar endereços IP para as sessões de dados. O SMF se comunica indiretamente com o UE por meio da AMF, que transmite as mensagens relacionadas à sessão entre os dispositivos e o SMF.

Comparado ao EPS (*Evolved Packet System*), o gerenciamento de sessão possui maior flexibilidade, como novas opções para tipos de protocolos de usuário final, diferentes abordagens para a continuidade de serviço e sessão, além de uma arquitetura flexível de Plano do Usuário.

O SMF interage com outras Funções de Rede por meio de interfaces baseadas em serviços e também seleciona e controla as diferentes funções de rede UPF por meio da interface de rede N4.

O SMF interage com a Função de Rede PCF para obter políticas que são usadas para configurar a UPF para a sessão de PDU, incluindo a configuração do direcionamento de tráfego na UPF para sessões individuais.

O SMF também coleta dados de cobrança e controla a funcionalidade de cobrança na UPF. O SMF oferece suporte tanto à cobrança offline quanto à cobrança online (STEFAN ROMMER PETER HEDMAN, 2019).

#### *2.6.1.3 UPF*

O UPF processa e encaminha os dados do usuário. Sua funcionalidade é controlada pelo SMF. Ele se conecta a redes IP externas e atua como um ponto de ancoragem para os dispositivos UE em direção a redes externas, ocultando a mobilidade. Isso significa que um endereço IP de uma sessão de PDU de UE específica pode ser roteado para o UPF que está servindo esse UE e sessão.

O UPF realiza diversos tipos de processamento dos dados encaminhados. Ele gera registros de dados de cobrança e relatórios de uso de tráfego. Ele pode realizar "inspeção de pacotes", analisando o conteúdo dos pacotes de dados do usuário para uso como entrada em decisões de política ou como base para relatórios de tráfego.

Também executa várias políticas de rede ou de usuário, como controle de tráfego, redirecionamento de tráfego ou aplicação de limitações de taxa de dados diferentes.

Quando um dispositivo está em estado ocioso e não está imediatamente acessível pela rede, qualquer tráfego enviado para esse dispositivo é armazenado em *buffer* pelo UPF, que aciona uma página da rede para forçar o dispositivo a retornar ao estado conectado e receber seus dados.

O UPF do núcleo 5G pode ser implantado em série, por exemplo, um UPF distribuído em direção à borda da rede e um UPF localizado em um local de rede mais central. Regras de rede podem ser usadas para controlar o encaminhamento de tráfego do UPF distribuído. A classificação dos pacotes de dados provenientes do UE (pacotes de UL) pode ser aplicada para determinar se os dados devem ser enviados para uma rede IP local e distribuída ou se os pacotes devem ser encaminhados para o UPF centralizado.

O UPF também pode aplicar marcação de Qualidade de Serviço (QoS) nos pacotes em direção à rede de rádio ou a redes externas. Isso pode ser usado pela rede de transporte para lidar com cada pacote com a prioridade correta em caso de congestionamento na rede (STEFAN ROMMER PETER HEDMAN, 2019).

#### *2.6.1.4 AUSF*

Ele é responsável por lidar com a autenticação na rede doméstica, com base nas informações recebidas do UE e informações obtidas do UDM (gerenciamento de dados do assinante). Ele fornece parâmetros de segurança para proteger as informações de encaminhamento de *roaming* e também fornece parâmetros de segurança para proteger as informações no procedimento de atualização do UE (STEFAN ROMMER PETER HEDMAN, 2019).

#### *2.6.1.5 NRF*

O NRF é um repositório que armazena os perfis das funções de rede disponíveis na rede. Sua função é permitir que os consumidores de serviço, como as NF, descubram e selecionem as NF e serviços de NF adequados, sem a necessidade de configurações prévias.

Quando uma nova instância de uma função de rede é implantada ou alterada, o NRF é atualizado com as informações de perfil correspondentes. Essas informações podem ser atualizadas pela própria função de rede ou por outras entidades em seu nome. Além disso, o NRF possui um mecanismo de manutenção ativa que garante a integridade do repositório, removendo os perfis de funções de rede ausentes ou inativas.

Os perfis das NF armazenados no NRF contêm informações como tipo de NF, endereço, capacidade e serviços de NF suportados, incluindo os endereços de cada instância de serviço de NF. Essas informações são fornecidas aos consumidores de serviços de NF durante o processo de descoberta, permitindo que escolham a interface de serviço da NF adequada.

Além disso, o perfil do NRF inclui informações de autorização e os perfis são disponibilizados apenas aos consumidores que possuem permissão para descobrir funções de rede ou serviços específicos (STEFAN ROMMER PETER HEDMAN, 2019).

#### *2.6.1.6 NSSF*

O NSSF (*Network Slice Selection Function*) tem como função dar suporte a seleção de fatias de rede com base na combinação de valores S-NSSAI (*Single Network Slice Selection Assistance Information*), que são definidos na rede e solicitados por cada dispositivo autorizado durante o processo de inscrição (STEFAN ROMMER PETER HEDMAN, 2019).

#### *2.6.1.7 PCF*

O PCF (*Policy Control Function*) fornece controle de políticas para diversas funcionalidades relacionadas à gestão de sessões, seleção de acesso e mobilidade, seleção de acesso do UE e seleção de sessões PDU, além de suportar a negociação de transferências futuras de dados em segundo plano. O PCF interage com funções de aplicação e o SMF para fornecer controle de QoS autorizado, controle de cobrança para fluxos de dados de serviço, controle de políticas relacionadas a sessões PDU e relatórios de eventos para as sessões PDU.

O PCF também interage com o AMF para controle de políticas de acesso e mobilidade, incluindo o gerenciamento de restrições de área de serviço e o gerenciamento de RFSP (Radio Frequency Selection Priority, um parâmetro usado pelo NG-RAN para diferenciar o tratamento de diferentes UE).

Além disso, o PCF fornece informações de política para o UE (via AMF), incluindo políticas de descoberta e seleção para redes não 3GPP, política de seleção de modo de continuidade de sessão, política de seleção de fatia de rede, política de seleção de nome de rede de dados e muito mais (STEFAN ROMMER PETER HEDMAN, 2019).

#### *2.6.1.8 UDM*

O UDM (Unified Data Management) é uma interface para os dados de assinatura do usuário armazenados no UDR (*Unified Data Repository*). O UDM utiliza os dados de assinatura armazenados no UDR para executar lógicas de aplicação, como autorização de acesso, gerenciamento de registro e alcançabilidade para eventos terminais, como SMS.

Quando um UE se conecta ao sistema, o UDM autoriza o acesso e realiza várias verificações de recursos suportados, bloqueios e restrições devido, por exemplo, ao *roaming*. O UDM gera as credenciais de autenticação que o AUSF usa para autenticar o UE. Ele também gerencia a privacidade da identidade permanente e pode ser usado por outras entidades para resolver a identidade permanente oculta (SUCI) para a identidade permanente real (SUPI).

Diferentes instâncias do UDM podem ser usadas para o mesmo usuário em transações diferentes. O UDM também rastreia qual instância do AMF está atendendo a um

UE específico e também os SMF(s) que estão atendendo às suas sessões PDU (STEFAN ROMMER PETER HEDMAN, 2019).

#### *2.6.1.9 UDR*

O UDR é um banco de dados que armazena diversos tipos de dados, incluindo informações de assinatura e políticas de rede e usuário. Ele oferece serviços de armazenamento e acesso a esses dados para outras funções de rede, como UDM, PCF e NEF (STEFAN ROMMER PETER HEDMAN, 2019).

#### *2.6.1.10 SCTPLB*

SCTPLB (*Stream Control Transmission Protocol Load Balancer*), é uma solução de balanceamento de carga projetada para lidar com o protocolo SCTP (*Stream Control Transmission Protocol*). O SCTP (*Stream Control Transmission Protocol*) é um protocolo de transporte orientado a conexão que oferece recursos avançados em relação ao TCP (Transmission Control Protocol), como suporte a múltiplas *streams* e controle de congestionamento.

O SCTPLB é uma implementação de balanceamento de carga específica para o protocolo SCTP, que distribui o tráfego entre múltiplos servidores ou destinos finais. Ele atua como um intermediário entre os clientes SCTP e os servidores SCTP, garantindo uma distribuição equilibrada das conexões e cargas de trabalho.

Uma das principais finalidades do SCTPLB é melhorar a escalabilidade e a disponibilidade dos serviços SCTP (*Stream Control Transmission Protocol*). Ele pode ser usado em cenários onde há um grande número de conexões SCTP e a carga precisa ser distribuída de forma eficiente para garantir o desempenho e a capacidade de resposta dos servidores.

O SCTPLB funciona interceptando as conexões SCTP dos clientes e redirecionando-as para os servidores de destino. Ele realiza a distribuição de carga com base em diferentes algoritmos, como *round-robin* (cada servidor recebe uma conexão sequencialmente), balanceamento de carga ponderado (atribui pesos diferentes a servidores com capacidades diferentes) ou algoritmos baseados em métricas de desempenho, como latência ou carga atual dos servidores.

Além do balanceamento de carga, o SCTPLB também pode oferecer recursos adicionais, como monitoramento de servidores, detecção de falhas e *failover*. Ele pode verificar periodicamente o estado dos servidores e remover ou adicionar servidores à configuração do balanceador de carga com base em seu status. Isso ajuda a garantir que apenas servidores saudáveis recebam o tráfego SCTP e melhora a resiliência do sistema (STEFAN ROMMER PETER HEDMAN, 2019).

## 2.7 QUALITY OF SERVICE - QOS

O modelo da (QoS) do 5G é baseado em fluxos de QoS, que possuem identificadores únicos. Existem dois tipos de fluxos: com GBR (Taxa de Bits Garantida) e sem GBR. Esses fluxos são utilizados para diferenciar o tratamento de tráfego de usuário. No nível do usuário, os pacotes são filtrados e mapeados para os fluxos de QoS. No nível de acesso, os fluxos de QoS são mapeados para as Portadoras de Rádio de Dados (DRBs). Cada fluxo de QoS possui um perfil com parâmetros e características que podem ser padronizadas ou configuradas dinamicamente.

### 2.7.1 Parâmetros da QoS

Cada fluxo de QoS possui um identificador único chamado Identificador de Fluxo de QoS (QFI). Existem dois tipos de fluxos: Fluxos de QoS com Taxa de Bits Garantida (GBR) e Fluxos de QoS não GBR. O Fluxo deQoS é a menor granularidade de diferenciação de QoS na Sessão PDU. O tráfego da Plataforma do Usuário (UP) com o mesmo QFI recebe o mesmo tratamento de encaminhamento. Cada fluxo de QoS possui um perfil de QoS que inclui parâmetros de QoS e características deles. Os parâmetros aplicáveis dependem do tipo de fluxo GBR ou não GBR. As características de QoS são padronizadas ou configuradas dinamicamente (DEVOPEDIA, 2021). Os parâmetros da QoS e especificação estão listados do anexo A.

Alguns Detalhes importantes para ressaltar:

- Identificador de QoS 5G (5QI): Um identificador para características de QoS que influenciam pesos de agendamento, limites de admissão, limites de gerenciamento de fila, configuração do protocolo de camada de enlace, etc.
- Prioridade de Alocação e Retenção (ARP): Informações sobre nível de prioridade, capacidade de substituição (pode substituir recursos atribuídos a outros fluxos de QoS) e vulnerabilidade à substituição (pode ser substituído por outros fluxos de QoS).
- Taxa de Fluxo Garantida (GFBR): Medida ao longo da Janela de Tempo Médio. Recomendado como a taxa de bits mais baixa na qual o serviço sobreviverá.
- Taxa de Fluxo Máxima (MFBR): Limita a taxa de bits ao máximo esperado por este fluxo de QoS.
- Taxa Máxima de Bits Agregada (AMBR): Session-AMBR é por sessão PDU em todos os seus fluxos de QoS. UE-AMBR é para cada UE.
- Controle de Notificação de QoS (QNC): Configura a NG-RAN para notificar o se o GFBR não puder ser atendido. Útil se a aplicação puder se adaptar às condições em mudança. Se perfis de QoS alternativos forem configurados, a NG-

RAN indica se um deles corresponde às métricas de desempenho atualmente atendidas.

- Taxa Máxima de Perda de Pacotes: No (3GP, 2020), isso é limitado a mídia de voz.
- *Packet Error Loss Rate* (PELR): A Taxa de Perda de Erros de Pacote (PELR) é uma métrica que estabelece um limite superior para a taxa de perda de pacotes que ocorrem devido a erros no nível da camada de enlace de uma rede. Ele representa a proporção de Unidades de Dados de Serviço (SDUs) processadas pelo remetente de um protocolo de camada de enlace, mas que não são entregues com sucesso pelo receptor correspondente à camada superior (3GPP, 2011)
- *Packet Delay and Budget* (PDB): Define um limite superior para o tempo que um pacote pode ser atrasado entre o UE (Equipamento do Usuário) e o ponto de terminação N6 na UPF. O PDB se aplica ao pacote DL recebido pela UPF na interface N6 e ao pacote UL enviado pelo UE. Para um determinado 5QI, o valor do PDB é o mesmo no UL e DL (3GPP, 2022).

### 2.7.2 Slicing

O *slicing* de rede, também conhecido como fatiamento de rede, é uma tecnologia avançada integrada no padrão 5G que permite a divisão de uma rede física em fatias lógicas separadas. Cada fatia proporciona isolamento e controle da qualidade de serviço (QoS) individualizado para atender às necessidades específicas de diferentes tipos de aplicação.

Por exemplo, aplicações que requerem baixa latência, como comunicações ultra-confiáveis e de baixa latência (URLLC), podem ser alocadas em uma fatia separada das aplicações que exigem maior largura de banda, como comunicações em banda larga aprimorada (eMBB). Isso permite que as operadoras de rede gerenciem de forma mais eficiente os requisitos e o desempenho de cada classe de uso.

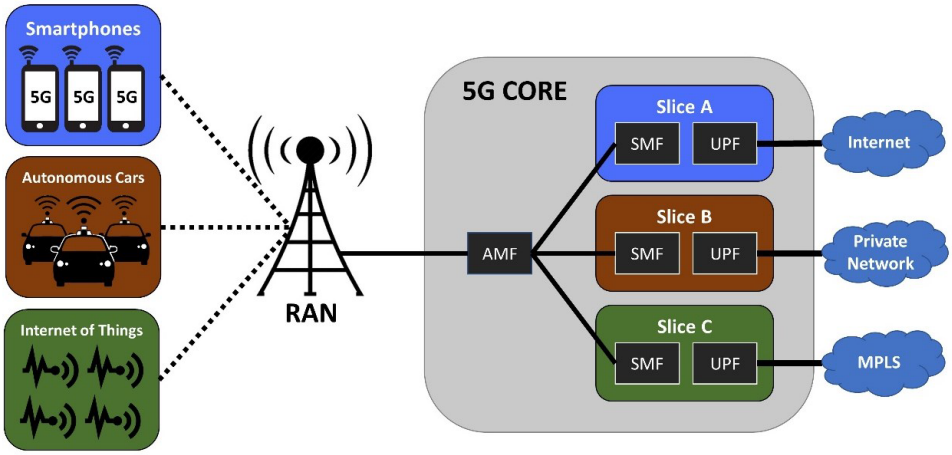
Além dos benefícios de desempenho e controle, as fatias de rede também proporciona maior segurança. Como estão isoladas logicamente, qualquer violação de segurança em uma fatia não afetará as outras, garantindo maior proteção para as aplicações.

Na Figura 4, é apresentado um exemplo prático de utilização de fatiamento de rede em um cenário com dispositivos representando cada uma das três classes de uso do 5G (URLLC, eMBB e mMTC). No exemplo, a fatia B é ajustada para ter menor latência e maior confiabilidade, recebendo prioridade sobre os recursos das outras fatias.

Dessa forma, o *slicing* de rede no 5G oferece maior flexibilidade, desempenho aprimorado, gerenciamento otimizado de recursos e maior segurança para atender às diversas demandas das aplicações na era da conectividade avançada.

Para identificar cada fatia, atribui-se a elas um valor chamado de S-NSSAI que, por sua vez, é dividido em SST (Slice/Service Type), que indica o tipo do serviço ou

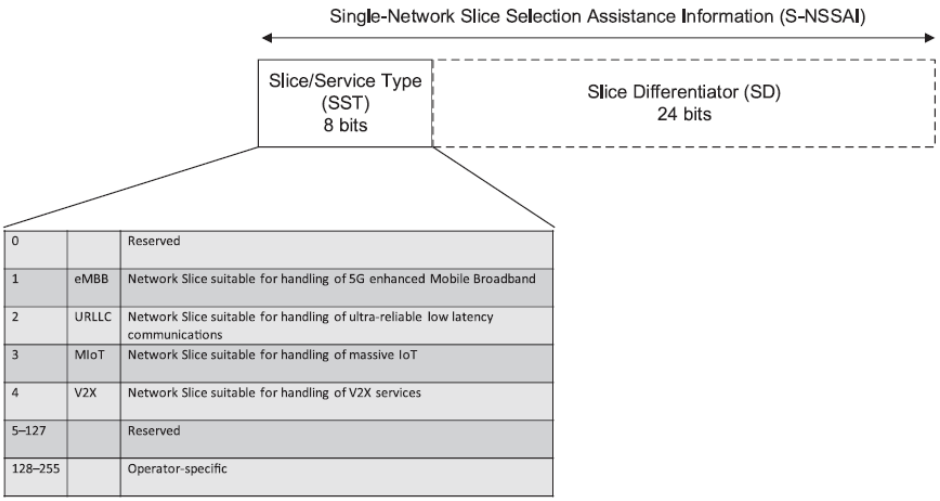
Figura 4 – Diferentes tipos de dispositivos utilizando o fatiamento de rede.




Fonte: (MAILER, 2020)

fatia, e SD (Slice Differentiator), parâmetro opcional que indica as diferentes fatias de um mesmo tipo (mesmo SST). A figura 5 mostra a regra de atribuição do S-NSSAI às fatias (MAILER, 2020).

Figura 5 – Formato do S-NSSAI.




| | | |
|---|---|---|
| 0 | | Reserved |
| 1 | eMBB | Network Slice suitable for handling of 5G enhanced Mobile Broadband |
| 2 | URLLC | Network Slice suitable for handling of ultra-reliable low latency communications |
| 3 | MIoT | Network Slice suitable for handling of massive IoT |
| 4 | V2X | Network Slice suitable for handling of V2X services |
| 5–127 | | Reserved |
| 128–255 | | Operator-specific |

Fonte: (MAILER, 2020)

Neste trabalho é utilizado o network slicing devido ao Aether in a box não suportar testes sem network slicing.

## 2.8 VIRTUALIZAÇÃO

Tradicionalmente, as redes móveis possuíam elementos funcionais do núcleo distribuídos em hardware dedicado. No entanto, a virtualização trouxe uma transformação significativa ao migrar essas aplicações para ambientes virtualizados, inicialmente por meio de máquinas virtuais (VM) e, posteriormente, com o uso de *containers*.

Essa transição para recursos virtualizados proporcionou maior flexibilidade na implementação das aplicações, uma vez que não estão mais limitadas pelos recursos e capacidades de um único equipamento físico. Essa abordagem oferece inúmeras vantagens para projetos de redes e telecomunicações, permitindo escalabilidade tanto em grandes implantações quanto em implementações mais modestas.

No contexto do 5G, há um foco em alcançar maior nível de abstração, simplificando a gestão e a operacionalização da rede. Além disso, a rápida evolução de novos modelos de negócios demanda a implementação ágil de novos serviços, impulsionando as operadoras a adotarem redes programáveis e definidas por *software*. Nesse sentido, as técnicas de virtualização desempenham um papel crucial ao atender essas necessidades em constante evolução (PUJOLLE, 2020).

Este trabalho explora técnicas de virtualização com o objetivo de alcançar o objetivo de simular a rede 5G e aplicar o *network slicing*, impulsionando a inovação e a transformação no contexto do 5G.

### 2.8.1 Máquinas Virtuais

A virtualização de sistema é uma tecnologia que permite a execução simultânea de várias instâncias de um sistema operacional em um *hypervisor*. Existem dois tipos de *hypervisors*: tipo 1 e tipo 2.

No caso do *hypervisor* tipo 1, ele é executado diretamente na plataforma de hardware e permite que os sistemas operacionais convidados sejam executados sem modificações, utilizando os *drivers* do próprio *hypervisor*. Exemplos de *hypervisor* tipo 1 incluem *VMware vSphere*, *VMware ESX*, *Microsoft Hyper-V* e *KVM*. Já o *hypervisor* tipo 2 é executado em um sistema operacional nativo e utiliza um emulador para permitir que os sistemas operacionais convidados sejam executados. Nesse caso, os sistemas operacionais convidados não estão cientes de que estão sendo virtualizados e não requerem modificações. Exemplos de *hypervisors* tipo 2 incluem *Microsoft Virtual PC*, *Oracle VM VirtualBox*, *VMware Player* e *QEMU*.

Essas tecnologias de máquinas virtuais surgiram como uma forma de otimizar o uso de recursos computacionais, permitindo a execução de múltiplos sistemas operacionais em um único *hardware*. Elas proporcionam isolamento, flexibilidade e escalabilidade, sendo amplamente utilizadas em ambientes de servidores, data centers e nuvem.

## 2.9 *CONTAINERS*

Os *containers* são um modelo de virtualização que oferece isolamento entre as aplicações, enquanto compartilha o *kernel* do sistema operacional. No setor de telecomunicações, os *containers* têm sido especialmente úteis em cenários que exigem baixa latência, resiliência e portabilidade, como em ambientes de computação de borda. Eles são ideais para a implementação de serviços de curta duração, em que há a necessidade de rápida implementação e agilidade. Além disso, em áreas como aprendizado de máquina e inteligência artificial, os *containers* ajudam a dividir problemas em tarefas menores, facilitando a automação. Existem várias soluções de *containers* disponíveis, e alguns exemplos são Linux-*Vserver*, *chroot*, *BSD Jail*, *OpenVZ* e *Docker*. O *Docker*, em particular, é uma das soluções mais populares e amplamente adotadas na indústria.

Essas tecnologias de *containers* proporcionam uma abordagem eficiente para empacotar, distribuir e executar aplicações, tornando o ambiente de execução mais leve e eficiente em comparação com a virtualização tradicional de máquinas virtuais.

Neste trabalho foi utilizado na metodologia a orquestração de *conteiners* em *kubernetes* no software do Aether-in-a-box.

## 2.10 REDES DEFINIDAS POR SOFTWARE E VIRTUALIZAÇÃO DE FUNÇÕES DE REDE

A arquitetura de Rede Definida por Software (SDN) e a Virtualização de Funções de Rede (NFV) são abordagens inovadoras que têm revolucionado o campo das redes de comunicação.

O SDN é uma arquitetura que visa superar as limitações do *hardware* por meio da abstração das funções de rede em níveis mais baixos. Em vez disso, as funções de rede são executadas de forma programática e baseada em *software*, com um plano de controle centralizado e interfaces de programação de aplicativos (API). No contexto do 5G, o SDN pode ser utilizado como uma estrutura para otimizar a transmissão de dados, melhorar a largura de banda e reduzir a latência. Ele permite o roteamento e re-roteamento de dados em tempo real, minimizando interrupções e contribuindo para o desenvolvimento de serviços de alta disponibilidade, especialmente importantes para aplicações críticas em tempo real (PENTTINEN, 2019).

Por outro lado, a NFV é uma abordagem que busca virtualizar as funções de rede tradicionalmente executadas em hardware dedicado. Com a NFV, é possível executar essas funções como softwares virtualizados, compartilhando recursos de hardware. Isso proporciona maior flexibilidade e escalabilidade na implementação de serviços de rede, além de reduzir os custos de infraestrutura. No contexto do 5G, a NFV desempenha um papel fundamental ao permitir a rápida implantação e adaptação de serviços para atender às demandas em constante evolução.

Em conjunto, o SDN e a NFV estão impulsionando a transformação das redes de comunicação, permitindo maior agilidade, eficiência e flexibilidade no provisionamento e gerenciamento de serviços de rede

## 2.11 KUBERNETES

O *Kubernetes* foi originalmente desenvolvido no *Google*, a partir de sua plataforma interna de gerenciamento de contêineres, chamada *Borg*. As lições aprendidas com a execução do *Borg* em produção tiveram um grande impacto no *Kubernetes*. O *Google* costumava executar 2 bilhões de contêineres por semana e, para gerenciar um ambiente tão massivo, eles usavam construções declarativas que definem a consistência eventual de um sistema e permitem que o orquestrador a alcance por conta própria. Esse design é repetível, facilmente atualizável e segue o conceito de 'configurar uma vez e depois esquecer' ao executar sistemas modernos.

O *Kubernetes* é o primeiro projeto do CNCF (*Cloud Native Computing Foundation*) a se formar e agora é uma plataforma de orquestração de contêineres de código aberto amplamente adotada pela indústria. Sua principal função é rastrear a utilização de contêineres, dimensionar com base na utilização, verificar continuamente a saúde, balancear a carga e realizar automaticamente a recuperação de falhas reiniciando os contêineres, se necessário. O *Kubernetes* pode ser executado em nuvens públicas ou utilizado como um serviço gerenciado pelos principais provedores de nuvem (KUBERNETES, 2023).

## 2.12 FERRAMENTAS DE SIMULAÇÃO

A seguir, serão apresentadas as principais ferramentas utilizadas na etapa de simulação deste trabalho.

### 2.12.1 Virtual Box

O *Oracle VM VirtualBox* é uma solução de virtualização multiplataforma de código aberto amplamente reconhecida e utilizada. Ele permite que desenvolvedores e usuários executem vários sistemas operacionais simultaneamente em um único computador, acelerando o processo de entrega de soluções de software.

O *VirtualBox* possui uma interface gráfica intuitiva e funcional, facilitando a configuração, clonagem, restauração e inicialização de máquinas virtuais (VMs). Essa interface amigável contribui para uma experiência de uso mais conveniente e eficiente.

Uma das grandes vantagens do *VirtualBox* é a sua licença de código aberto, do tipo GPL-2.0 (General Public License v2.0). Isso significa que o *VirtualBox* é livre para uso e distribuição, sem a necessidade de pagamento de taxas de licenciamento.

No contexto geral, o *VirtualBox* é uma ferramenta extremamente útil para simulações e testes de diversos cenários. Ele permite criar ambientes virtuais isolados, onde é

possível testar diferentes configurações de sistemas operacionais, aplicativos e redes. Isso é especialmente valioso para desenvolvedores de software, profissionais de TI e entusiastas que desejam experimentar e explorar novas tecnologias, sem comprometer o ambiente de produção.

Além disso, o *VirtualBox* também é amplamente utilizado para fins educacionais, fornecendo aos estudantes e pesquisadores uma plataforma segura e flexível para aprender e realizar experimentos em diferentes sistemas operacionais e configurações de rede (VIRTUALBOX, 2023).

### 2.12.2 Aether-in-a-box

O *Aether-in-a-Box* é um projeto de pesquisa e desenvolvimento no contexto do 5G que visa criar uma plataforma de testes e experimentação para a tecnologia 5G e suas aplicações. Esse projeto é financiado e apoiado por várias empresas líderes da indústria de telecomunicações, incluindo fornecedores de equipamentos, operadoras de rede e instituições acadêmicas.

O objetivo do *Aether-in-a-Box* é fornecer uma solução completa e de ponta a ponta para testes e validação de tecnologias 5G, incluindo equipamentos de rede, dispositivos móveis e aplicações. Ele busca criar um ambiente controlado e flexível, onde pesquisadores e desenvolvedores possam explorar os recursos do 5G, avaliar o desempenho de novos serviços e experimentar novas ideias em redes privadas.

A pesquisa realizada no âmbito do projeto *Aether-in-a-Box* abrange diversas áreas relacionadas ao 5G, como redes de acesso sem fio, virtualização de redes, computação em nuvem, *internet* das coisas (IoT), segurança e muito mais. Os pesquisadores envolvidos têm a oportunidade de explorar os desafios e as oportunidades apresentadas pelo 5G, contribuindo para o avanço dessa tecnologia e suas aplicações.

O *Aether-in-a-Box* surgiu como uma resposta à crescente demanda por plataformas de testes 5G mais acessíveis e flexíveis e também a rede privada é um conceito de grande impacto. A ideia por trás do projeto é permitir que empresas e instituições de pesquisa tenham acesso a um ambiente de teste realista e controlado, sem a necessidade de grandes investimentos em infraestrutura.

Ao disponibilizar uma plataforma aberta e modular, o *Aether-in-a-Box* incentiva a colaboração e o compartilhamento de conhecimento entre diferentes atores da indústria de telecomunicações. Isso contribui para impulsionar a inovação e acelerar o desenvolvimento de novas aplicações e serviços baseados no 5G.

No decorrer do projeto, várias empresas estão envolvidas no financiamento e suporte, incluindo fornecedores de equipamentos de rede, fabricantes de dispositivos móveis, operadoras de telecomunicações e instituições de pesquisa. Essa colaboração entre diferentes *stakeholders* é fundamental para impulsionar o desenvolvimento do 5G e garantir sua adoção bem-sucedida.

No geral, o projeto *Aether-in-a-Box* representa um esforço conjunto para impulsionar a pesquisa pela ONF, desenvolvimento e adoção do 5G, fornecendo uma plataforma de testes aberta e flexível. Com o envolvimento de várias empresas e instituições, ele está contribuindo para a evolução contínua do 5G e a criação de um ecossistema inovador para as futuras redes de telecomunicações (FOUNDATION, 2020).

### 2.12.3 Ueransim

O *UERANSIM* é um simulador de código aberto que foi desenvolvido para testar e estudar o sistema 5G, abrangendo o UE (Equipamento do Usuário) e a gNB (Estação Base da Próxima Geração). Ele suporta a versão 15 em diante da especificação 3GPP e possui a licença pública geral GNU v3.0 (GPL-3.0 - GNU General Public License v3.0).

Uma das principais vantagens do *UERANSIM* é que ele oferece a primeira e única implementação de código aberto do UE e da gNB 5G no mundo. Isso significa que o simulador está pronto para uso e possui funcionalidades completas. Vale ressaltar que algumas partes desse software estão com patente pendente, demonstrando o caráter inovador e exclusivo da ferramenta (GÜNGÖR, 2022).

# 3 DESCRIÇÃO E CONFIGURAÇÃO DO SISTEMA DE SIMULAÇÃO

## 3.1 AMBIENTE DE SIMULAÇÃO

A metodologia para a construção do ambiente de simulação, que tem como objetivo analisar a QoS do tráfego do core da rede 5G, é descrita neste capítulo. Iniciamos com uma visão geral do ambiente de simulação, seguido das configurações nos simuladores Aether-in-a-box e UERANSIM. Por fim, detalhamos as funções responsáveis pelo tráfego e classificação do QoS.

## 3.2 CRIAÇÃO DAS VMS

Para criar o ambiente de simulação, foram configuradas duas máquinas virtuais (VMs) utilizando o *software* VirtualBox, executando em um sistema hospedeiro Ubuntu 20.04.06. As configurações das VMs são apresentadas na tabela 2:

Tabela 2 – Configurações das VMs e Computador hospedeiro.

| | **Aplicação** | **SO** | **Endereço IP** | **Núcleos** | **Memória RAM** | **Armazenamento em disco** |
|---|---|---|---|---|---|---|
| VM2 | Aether-in-a-box | Ubuntu Server 18.04 | 192.168.56.120 | 8 un | 16384 MB | 100 GB |
| VM1 | UERANSIM | Ubuntu Server 18.04 | 192.168.56.110 | 2 un | 4096 MB | 25 GB |
| Host | Computador Hospedeiro | Ubuntu 20.04.06 | 192.168.18.1 | 12 un | 32.768 MB | 512 GB |

Fonte: O Autor.

Na VM2, está instalado o simulador Aether-in-a-box, que instancia os contêineres contendo as funções de rede virtualizadas do core, além de aplicativos adicionais, como o ROC (*Runtime Operational Control*), e o banco de dados MongoDB. A VM1 hospeda o UERANSIM, no qual são criados 30 dispositivos de usuário (UEs) que são conectados a uma central de rede. A partir da VM1, o tráfego é enviado para a rede por meio desses dispositivos de usuário.

A criação e inicialização das VMs foram realizadas seguindo os seguintes passos. Primeiramente, o programa VirtualBox foi instalado no computador hospedeiro, e em seguida, a imagem do Ubuntu Server foi baixada. Posteriormente, foram criadas duas VMs no software de virtualização, conforme descrito na tabela 2. Cada VM foi configurada com dois adaptadores de rede: o primeiro adaptador configurado como NAT e o segundo como *host-only*. A imagem do Ubuntu Server foi então inserida nas VMs, e o processo de instalação foi concluído.

Para ambas as VMs, foram utilizados o Código Fonte 3.1 *scripts* para editar o arquivo *sudoers*, responsável por definir os usuários e grupos com privilégios de adminis-

trador. (BRASIL, 2019)

Código Fonte 3.1 – Script para definir usuário com direito a administrador

```
sudo visudo
.
.
aether ALL=(ALL) NOPASSWD:ALL
```

O mesmo foi utilizado na VM1, alternado somente o nome de usuário de aether para ueransim.

No próximo passo, foram atribuídos endereços IP estáticos à interface de rede do Adaptador *Host-Only* (enp0s8), e a configuração de DHCP foi desabilitada nessa mesma interface, conforme exemplificado no Código Fonte 3.2 para a VM2.

Código Fonte 3.2 – Comandos para tornar estática os endereços IP

```
$ sudo vim /etc/netplan/00-installer-config.yaml
.
.
.
network:
    ethernets:
    enp0s3:
        dhcp4: true
    enp0s8:
        dhcp4: false
        addresses: [192.168.56.120/24]
    version: 2
```

Após a definição do endereço IP estático na VM, os comandos *netplan try* e *netplan apply* foram utilizados para verificar a validade do endereço IP e aplicar as configurações, como demonstrado no Código Fonte 3.3. Além disso, um processo de reinicialização foi executado para confirmar as alterações de configuração.

Código Fonte 3.3 – Comandos para tornar estática os endereços IP

```
$ sudo netplan try
...
Press Enter
...
$ sudo netplan apply
$ sudo shutdown -r now
```

Por fim, com os endereços IP atribuídos à VM1 e à VM2, é possível estabelecer uma conexão SSH (*Secure Shell*) por meio de um programa de terminal. Para essa finalidade,

foi utilizada a extensão *Remote - SSH* (VSCODE, 2023) da IDE Visual Studio Code, permitindo a execução de comandos e a adição de *scripts* a partir do sistema anfitrião. Isso possibilita uma interação direta e controle das VMs a partir do computador hospedeiro.

## 3.3 CONFIGURAÇÃO DO AETHER-IN-A-BOX

Primeiramente, acessando a VM2 via SSH, é clonado o repositório do Aether-in-a-box localizado em (FOUNDATION, 2021b), e em seguida ajustados o trecho localizado no arquivo sd-core-5g-values.yaml para expor a porta IP do serviço AMF conforme o Código Fonte 3.4

Código Fonte 3.4 – Ajuste no script para expor a porta IP com a porta do serviço Amf

```
amf:
    ngapp:
        externalIp: 192.168.56.120
        port: 38412
```

Vale ressaltar que também foi feito um ajuste no mesmo arquivo no parâmetro "simapp", que é um componente de gerenciamento de cartões SIM do SD-Core como mostra o Código Fonte 3.5. Esse componente é responsável por comunicar os cartões SIM recém-provisionados para o ROC.

Código Fonte 3.5 – Ajuste no script para possibiliar o gerenciamento de cartões SIM ao ROC

```
simapp.yaml:
  configuration:
    sub-proxy-endpt:
      addr: subscriber-proxy.aether-roc.svc.cluster.local
      port: 5000
```

Após a alteração no *script*, podemos executar o seguinte comando do Código Fonte 3.13 para inicializar a plataforma de rede central SD-CORE (FOUNDATION, 2021a).

Código Fonte 3.6 – Comandos iniciar o Aether-in-a-box

```
$ ENABLE_GNBSIM=false DATA_IFACE=enp0s8 CHARTS=release-2.1
   make 5g-core
```

Para adicionar mais UPFs à rede central, foi feito um ajuste no Makefile do projeto (Código Fonte 3.7). Esse ajuste permite que, sempre que o comando for executado, um novo UPF seja implantado em um novo *namespace*, que é usado para isolar e agrupar recursos dentro de um *cluster*. O namespace segue o formato "upf-n", onde "n"é um valor crescente que começa em "1". Em outras palavras, na primeira execução do alvo "5g-upf",

um novo UPF é implantado no *namespace* "upf-1". Na próxima execução, outro UPF é implantado no *namespace* "upf-2"e assim por diante.

Além disso, o alvo "5g-upf"também lida com a configuração de rede para o UPF. Ele atribui endereços IP às interfaces *SD-Core* e *Access* do UPF de forma crescente. O primeiro UPF adicional recebe o "IP_ID = 4", o segundo UPF adicional recebe o "IP_ID = 5"e assim por diante.(FOUNDATION, 2020)

Código Fonte 3.7 – Ajuste no script para adicionar mais UPFs

```
1  @if [[ $(UPF_NUMBER) -lt 1 ]]; then \
2        echo "Deploy '5g-core' before adding additional UPF(s)"
           ; \
3          exit 1; \
4  fi
5  $(eval IP_ID=$(shell echo ((3+$(UPF_NUMBER)))))
6  $(eval IP_UE_ID=$(shell echo $$((250-$(UPF_NUMBER)))))
7  NODE_IP=${NODE_IP} DATA_IFACE=${DATA_IFACE} RAN_SUBNET=${
     RAN_SUBNET} IP_ID=${IP_ID} IP_UE_ID=${IP_UE_ID} envsubst <
     $(5G_UPF_VALUES) | \
8  helm upgrade --create-namespace --install --wait $(
     HELM_GLOBAL_ARGS) \
9          --namespace upf-${UPF_NUMBER} \
10         --values - \
11         bess-upf \
12         $(UPF_CHART)
13 @if [[ "${ENABLE_GNBSIM}" == "true" ]]; then \
14     kubectl -n default exec -ti router -- ip route add 172.$(
         IP_UE_ID).0.0/16
15     via 192.168.250.$(IP_ID); \
16 else \
17     sudo ip route add 172.$(IP_UE_ID).0.0/16 via 192.168.250.
         $(IP_ID) dev core; \
18 fi
19 $(eval UPF_NUMBER=$(shell echo $$(($(UPF_NUMBER)+1))))
20 @echo $(UPF_NUMBER) > ${UPF_COUNT}
```

Após realizar os ajustes necessários, podemos utilizar o seguinte comando no Código Fonte 3.8 para adicionar mais UPFs. Foram executadas apenas mais duas vezes, uma vez que uma UPF já foi adicionada por padrão pelo Código Fonte. Essas UPFs serão utilizadas posteriormente para a realização de testes com *slices*.

Código Fonte 3.8 – Comandos adicionar UPF ao ambiente

```
$ ENABLE_GNBSIM=false DATA_IFACE=enp0s8 CHARTS=release-2.1
   make 5g-upf
```

No Código Fonte 3.9 podemos observar um arquivo que contém *shell scripts* que contêm algumas regras de configuração de *firewall*. A primeira configuração cria uma condição para aceitar o encaminhamento de pacotes vindo nessa interface, a segunda configuração permite a ativação de roteamento e a terceira cria uma condição para pacotes que estão prestes a sair do core pela interface enp0s3 para IPs dinâmicos.

Código Fonte 3.9 – Arquivo contendo regras de firewall para o roteamento de pacotes no core

```
#!/bin/bash
iptables --policy FORWARD ACCEPT
sysctl -w net.ipv4.ip_forward=1
iptables -t nat -A POSTROUTING -o enp0s3 -j MASQUERADE
```

Para habilitar o serviço de ROC, foi utilizado o comando do Código Fonte 3.10.

Código Fonte 3.10 – Comandos habilitar o roc

```
$ ENABLE_GNBSIM=false CHARTS=release-2.1 make roc-5g-models
```

Assim finalizamos a instalação do sistema e podemos iniciar o ajuste no ROC para adicionar mais *slices* e ajuste no QoS para testes.

### 3.3.1 Runtime Operational Control - ROC

Após a instalação do ROC utilzando o Código Fonte 3.10, podemos acessar a interface pelo endereço IP 192.168.56.120:31194 do navegador, e assim encontramos a primeira interface 7, que é o *Dashboard*.

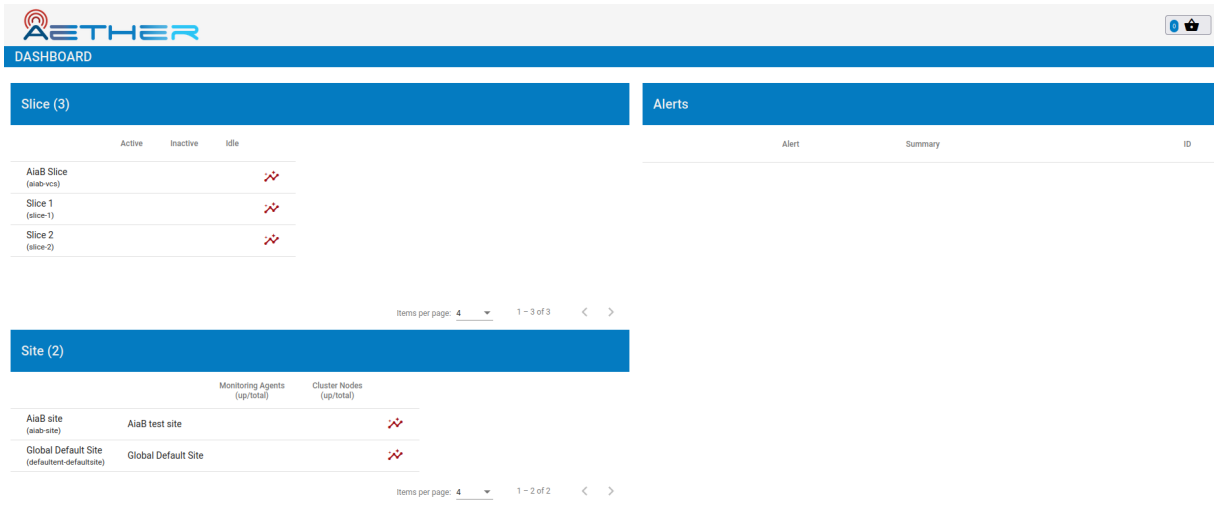
O ROC possui uma interface intuitiva que pode ser configurado seguindo o documento em (FOUNDATION, 2020).

## 3.4 CONFIGURAÇÃO DO UERANSIM

O simulador UERANSIM, assim como o Aether-in-a-box na VM2, teve seu repositório disponível em (GÜNGÖR, 2022) clonado na VM1. Foram realizadas as configurações iniciais necessárias para o seu funcionamento. O UERANSIM possui arquivos de configuração que precisaram ser ajustados para integração com as aplicações simuladas do SD-Core na VM1. Nesse sentido, foram realizadas modificações em apenas 2 arquivos, enquanto outros 29 arquivos foram adicionados especificamente para este trabalho, visando uma configuração adequada e personalizada.

Durante a configuração inicial, foram realizados ajustes no arquivo da antena 5G simulada, onde os endereços linkIp, ngapIp e gtpIp foram configurados para apontar

Figura 6 – Interface gráfica do ROC




Fonte: Aether ROC

para o endereço da própria VM. Além disso, foi feito o ajuste do endereço do AMF para estabelecer a comunicação com a VM2, conforme configurado no Código Fonte 3.2. Também foram adicionadas três novas fatias de serviço (Código Fonte 3.11), juntamente com seus respectivos Modelos de Fatiamento de Serviço (SST - *Service Slice Template*)

Código Fonte 3.11 – Arquivo para a configuração da antena 5G simulada

```
1  mcc: '208'
2  mnc: '93'
3
4  nci: '0x000000010'
5  idLength: 32
6  tac: 1
7
8  linkIp: 192.168.56.110
9  ngapIp: 192.168.56.110
10 gtpIp: 192.168.56.110
11
12 amfConfigs:
13   - address: 192.168.56.120
14     port: 38412
15
16 slices:
17   - sst: 0x1
18     sd: 0x010203
```

```
19    - sst: 0x2
20      sd: 0x010203
21    - sst: 0x3
22      sd: 0x010203
23
24  ignoreStreamIds: true
```

O segundo arquivo editado corresponde ao arquivo de configurações do equipamento do usuário, conforme mostrado no Código Fonte 3.11. No campo que representa a antena que estabelecerá a conexão com o UE, foi inserido o valor do endereço IP fixo da VM2. Além disso, os valores das *sessions* e *configured-nssai* foram ajustados de acordo com a respectiva fatia que seria testada posteriormente.

Os 29 arquivos adicionados foram copiados a partir do Código Fonte 3.12 e tiveram seus valores de supi ajustados para a inscrição no ROC. Também foram ajustados os valores de *slice* e *configured-nssai* para realizar testes de fatiamento de rede.

Código Fonte 3.12 – Arquivo para a configuração do equipamento do usuário

```
1  supi: 'imsi-208930100007487'
2  mcc: '208'
3  mnc: '93'
4
5  key: '5122250214c33e723a5dd523fc145fc0'
6  op: '981d464c7c52eb6e5036234984ad0bcf'
7  opType: 'OPC'
8  amf: '8000'
9  imei: '356938035643803'
10 imeiSv: '4370816125816151'
11
12 gnbSearchList:
13   - 192.168.56.110
14
15 uacAic:
16   mps: false
17   mcs: false
18
19 uacAcc:
20   normalclass: 0
21   class11: false
22   class12: false
23   class13: false
```

```
24    class14: false
25    class15: false
26
27  sessions:
28    - type: 'IPv4'
29      apn: 'internet'
30      slice:
31        sst: 0x1
32        sd: 0x010203
33
34  # Configured NSSAI for this UE by HPLMN
35  configured-nssai:
36    - sst: 0x1
37      sd: 0x010203
38
39  # Default Configured NSSAI for this UE
40  default-nssai:
41    - sst: 1
42      sd: 1
43
44  # Supported integrity algorithms by this UE
45  integrity:
46    IA1: true
47    IA2: true
48    IA3: true
49
50  # Supported encryption algorithms by this UE
51  ciphering:
52    EA1: true
53    EA2: true
54    EA3: true
55
56  # Integrity protection maximum data rate for user plane
57  integrityMaxRate:
58    uplink: 'full'
59    downlink: 'full'
```

Após configurar a antena simulada e os 30 equipamentos de usuário, é possível inicializá-los utilizando os comandos fornecidos no Código Fonte 3.13 em terminais separados. Esses comandos devem ser executados após iniciar o SD-Core simulado na VM2.

Durante a execução, serão criados 30 túneis para o tráfego de dados dos usuários, denominados de uesimtun0 a uesimtun30. Cada um desses túneis estabelece uma interface N1 que conecta o UE à AMF. Além disso, há uma interface N2 que conecta a antena à AMF e uma interface N3 que liga a RAN à UPFB no lado do core. Essas interfaces providenciam comunicação entre os diferentes elementos da rede, permitindo a transferência de dados e o gerenciamento adequado das conexões.

Código Fonte 3.13 – Comandos para tornar estática os endereços IP

```
# Inicializar a antena 5G
$ sudo build/nr-gnb -c config/custom-gnb.yaml
.
.
# Inicializar UE
$ sudo build/nr-ue -c config/custom-ue-0.yaml
.
.
.
$ sudo build/nr-ue -c config/custom-ue-29.yaml
```

Por fim, utilizando os comando do Código Fonte 3.14 as rotas para o acesso para enviar tráfego pela interface enp0s8 da VM2, essas rotas são as UPFs adicionadas pelo comando Código Fonte 3.7 a partir da VM2.

Código Fonte 3.14 – Comandos para adicionar rotas para o acesso do core

```
sudo ip route add 192.168.252.3 via 192.168.56.120
sudo ip route add 192.168.252.4 via 192.168.56.120
sudo ip route add 192.168.252.5 via 192.168.56.120
```

## 3.5 ANÁLISES COM PING E NETPERF

Após a conclusão da criação dos túneis para o envio de tráfego, utilizou-se a ferramenta Ping para verificar a latência da rede entre as duas máquinas virtuais. O comando foi executado na interface de rede gerada pelo Código Fonte 3.15, que percorre o túnel e tem como trajeto a rede do core, chegando até a rede de internet.

Outra ferramenta utilizada foi o Netperf, uma referência para medição de diversos aspectos de desempenho de redes. O foco principal do Netperf é a transferência de dados em massa e o desempenho de solicitação/resposta, utilizando os protocolos TCP ou UDP e a interface de *sockets Berkeley* (PAGE, 2023). Neste trabalho, o comando foi utilizado para medir a largura de banda do túnel formado pelo Código Fonte 3.15 até a interface de rede da VM2.

Os comandos de Ping em cada uma das 30 interfaces criadas pelo Código Fonte A e em seguida utilizado o comando Netperf nas mesmas interfaces criadas. Primeiramente, o comando Ping foi executado nas 30 interfaces para analisar a conectividade e a latência da rede. O comando Netperf foi executado nas mesmas interfaces para medir a largura de banda e o desempenho de solicitação/resposta por 10 segundos. Os resultados obtidos em cada etapa foram posteriormente importados para um arquivo de texto, permitindo uma análise mais detalhada.

## 3.6 ASSOCIAÇÃO DE IP DOMAIN NO ROC

Para associar o IP Domain gerada ao ROC, executamos o comando do Código Fonte 3.15 que captura a rota gerada ao core:

Código Fonte 3.15 – Comando para localizar o endereço subnet do core

```
1 $ route -n | grep "Iface\|core"
2 Destination    Gateway          Genmask      Flags . . . Iface
3 172.248.0.0    192.168.250.5    255.255.0.0 UG    . . . core
4 172.249.0.0    192.168.250.4    255.255.0.0 UG    . . . core
5 172.250.0.0    192.168.250.3    255.255.0.0 UG    . . . core
```

A regra acima corresponde a pacotes destinados aos dispositivos do usuário (UEs) na subnet 172.250.0.0/16, 172.249.0.0/16, 172.248.0.0/16. Assim o próximo trajeto para esses pacotes é o endereço IP do núcleo da UPF (FOUNDATION, 2020). Com esses dados, foi preenchido os campos do ROC na seção de IP *Domain* com os valores dos destinatários coletados no Código Fonte 4.1.

## 3.7 CONFIGURAÇÃO DA QOS

Para a criação de prioridades no Aether-in-a-box, é selecionado pela interface de usuário os seguintes campos:

- *Packet Error Loss Rate* (PELR)
- *Packet Delay and Budget* (PDB)
- *Allocation and Retendion Priority* (ARP)
- QoS *class Indentifier* (QCI)

Os valores de PELR variam de 0 a 10, e os valores de PDB e ARP variam de 0 a 1000 e 1 a 15 respectivamente, e por fim, o valor da QoS varia entre 1 a 32, os valores foram tabelados no A

Para os testes executados foram selecionados os valores a pela tabela 3, os valores são padronizados e definidos pelo software Aether-in-a-box.

Tabela 3 – Configurações de *traffic class*.

| **PELR** | **PDB** | **ARP** | **QCI** | **Prioridade** |
|---|---|---|---|---|
| 0 | 0 | 1 | 1 | Alta |
| 0 | 20 | 5 | 9 | Média |
| 0 | 500 | 9 | 18 | Baixa |

Fonte: O Autor.

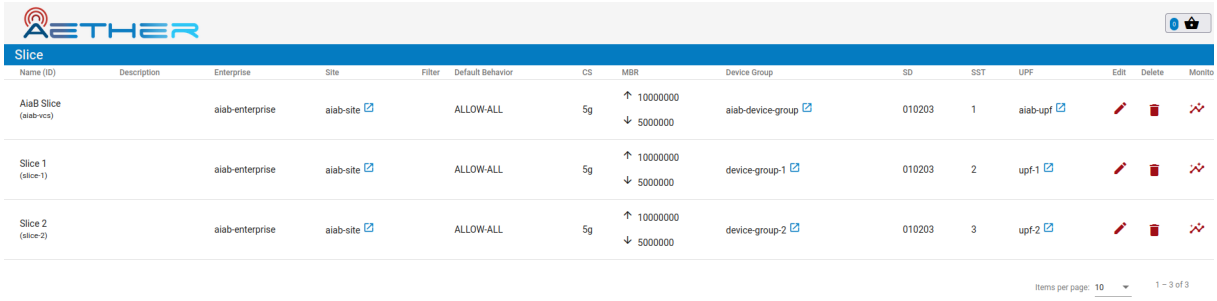
## 3.8 ASSOCIAÇÃO DE DISPOSITIVO AO GRUPO DE DISPOSITIVOS E AO SLICE

Após o registro do UE no SD-Core hospedado na VM2 por meio do ROC GUI, é realizada a associação dos dispositivos criados ao *device groups*. Nessa página, é possível selecionar o site, IP *Domain*, *Uplink*, os dispositivos que farão parte do *slice*, e *traffic class*, que corresponde à classe de serviço de acordo com a Tabela 3.

## 3.9 ASSOCIAÇÃO DE GRUPOS DE DISPOSITIVOS E UPF AOS *SLICES*

Para a criação de *slices* e associar os grupos de dispositivos e também UPF aos *slices*, foi criado na interface conforme a figura 7

Figura 7 – Configuração de slice no ROC




Fonte: Aether ROC

Podendo preencher os campos de grupo de dispositivos, UPF, SST e SD. Vale ressaltar que para os testes, os *slices* 2 e 3 não foram utilizados nos testes de 1 ao 3, e o *slice* 3 foi utilizado nos testes 7 a 9 como mostra a tabela 4 na seção dos resultados.

# 4 RESULTADOS

Neste capítulo, serão apresentados os resultados obtidos por meio de imagens, gráficos e quadros, com o objetivo de fornecer detalhes esclarecedores sobre o *slices* utilizados na simulação de tráfego do 5G.

## 4.1 PADRÃO DOS TESTES

Os padrões dos testes executados seguem a Tabela 4. Cada teste realizado foi configurado uma prioridade para cada *slice*. O critério para definir a prioridade do grupo foi arbitrária e o número de interfaces de cada grupo foi dividido homogeneamente para análise.

Tabela 4 – Padrão dos testes realizados.

| **Teste** | **Grupo** | **Interfaces uesimtun** | **Prioridade** |
|---|---|---|---|
| 1 | Grupo 1 | 1 ao 30 | Alta |
| 2 | Grupo 1 | 1 ao 30 | Média |
| 3 | Grupo 1 | 1 ao 30 | Baixa |
| 4 | Grupo 1<br>Grupo 2 | 1 ao 15<br>16 ao 30 | Alta<br>Média |
| 5 | Grupo 1<br>Grupo 2 | 1 ao 15<br>16 ao 30 | Alta<br>Baixa |
| 6 | Grupo 1<br>Grupo 2 | 1 ao 15<br>16 ao 30 | Alta<br>Alta |
| 7 | Grupo 1<br>Grupo 2<br>Grupo 3 | 1 ao 10<br>11 ao 20<br>21 ao 30 | Alta<br>Média<br>Baixa |
| 8 | Grupo 1<br>Grupo 2<br>Grupo 3 | 1 ao 10<br>11 ao 20<br>21 ao 30 | Alta<br>Média<br>Média |
| 9 | Grupo 1<br>Grupo 2<br>Grupo 3 | 1 ao 10<br>11 ao 20<br>21 ao 30 | Alta<br>Alta<br>Alta |

Fonte: O Autor.

## 4.2 DADOS OBTIDOS UTILIZANDO NETPERF

Nesta seção, são apresentados os dados obtidos utilizando o Netperf para medir a latência, o *jitter*, a perda de pacotes e o *throughput* com o protocolo UDP. No código fonte 4.1, podemos observar o tráfego de pacotes que passam pela interface entre o cliente e o servidor.

Na VM1, é utilizado o comando netperf, de acordo com o código fornecido. Esse comando tem como parâmetros o endereço IP do servidor, o endereço IP da interface pela

qual os pacotes serão transmitidos, a quantidade de pacotes enviados, o tipo de protocolo, o tamanho dos pacotes e, por fim, o intervalo de tempo entre os envios. Vale ressaltar que os testes foram executados simultaneamente em todos os 30 dispositivos e os resultados individuais localizam se no apêndice D.

Nos resultados com 1 *slice*, o *throughput* médio foi de 60,603 Mbps para a prioridade alta, 51,822 Mbps para a prioridade média e 49,776 Mbps para a prioridade baixa. Os valores utilizados para o cálculo do *throughput* médio estão no apêndice A.

Para 2 *slices*, o *throughput* médio foi de 68,576 Mbps para as *slices* com alta e baixa prioridade. Para a combinação de alta e média prioridade, o *throughput* médio foi de 65,270 Mbps. Por fim, para duas *slices* configuradas com alta prioridade, o *throughput* médio foi de 60,428 Mbps, valor inferior aos dois anteriores.

No caso de 3 *slices*, o *throughput* médio para a configuração com *slices* de alta, média e baixa prioridade foi igual para a configuração com alta prioridade e duas médias prioridades, o valor médio do *throughput* foi de 75,303 Mbps. Por fim, para a configuração com 3 *slices* de alta prioridade, o valor médio do *throughput* foi de 69,097 Mbps.

Também foi calculado o valor médio de *throughput* por slice como mostra a Tabela 5, podemos avaliar que a configuração de 1 slice o *throughput* é inferior quanto a configuração de mais slices, e conforme aumenta o número de slices, pode ver um *throughput* maior. Nota-se que a slice com priorioridade alta tem uma vazão maior na configuração de duas slices, e conforme aumenta a prioridade, o *throughput* diminui. E na configuração de 3 slices, podemos avaliar que na configuração de alta média e baixa, o slice com prioridade alta teve um *throughput* alto comparado com os outros, e conforme diminuímos as prioridades, o valor não é uniforme como esperado.

Nota-se que a configuração com prioridade alta e baixa em duas slices, a interface uesimtun1 possui um valor elevado comparado com as outras interfaces e o mesmo ocorre na configuração de 3 slices, que na qual a interface uesimtun7 possui o mesmo comportamento, pode-se afirmar que é uma limitação do computador hospedeiro.

Podemos afirmar que o aumento de slices e os ajustes na QoS afetam a taxa de transferência. Ao analisar os valores da Tabela 5, observamos um aumento gradual na taxa de transferência à medida que o número de slices é aumentado, com exceção da configuração de 3 slices com prioridade alta e duas médias, visto que a transição não foi positiva.

Tabela 5 – Valor médio de *throughput*(Mbps) de cada teste realizado.

| **Slice(s)** | **Prioridade** | **Valor médio por slice** |
|---|---|---|
| 1 | Alta | 60,603 |
| | Média | 51,822 |
| | Baixa | 49,776 |
| 2 | Alta e Baixa | 75,775 e 61,37 |
| | Alta e Média | 64,74 e 65,80 |
| | 2 Altas | 58,87 e 57,99 |
| 3 | Alta, Média e Baixa | 90,05 69,55 e 66,29 |
| | 3 Altas | 62,91 63,34 e 61,17 |
| | Alta e duas Médias | 67,75 69,55 e 66,30 |

Fonte: O Autor.

## 4.3 GRÁFICO DOS DADOS PING OBTIDOS

Com o objetivo de ilustrar e comparar os resultados obtidos em um ambiente mais próximo a realidade, foram gerados gráficos mostrando as latências obtidas por meio do protocolo ICMP do cliente até a rede de internet (8.8.8.8). O comando utilizado é mostrado no código fonte 4.1:

Código Fonte 4.1 – Comando executado para testes de latência

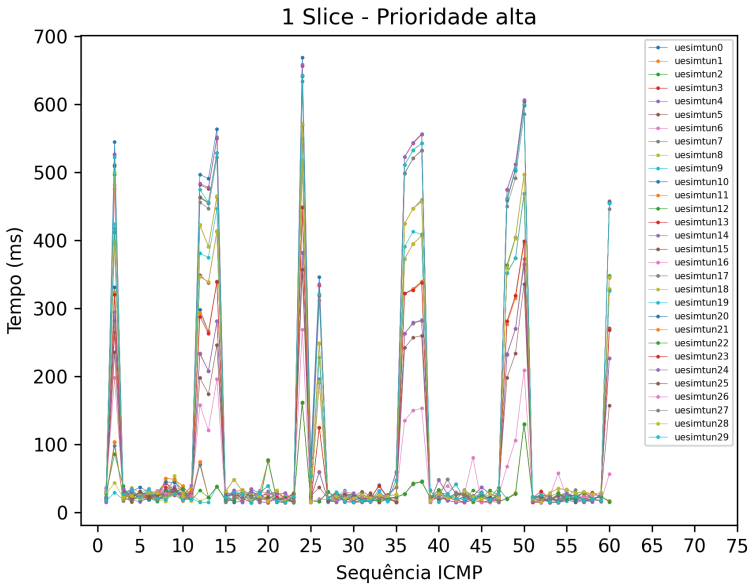
```
1  PING 8.8.8.8 (8.8.8.8) from 172.250.237.143 uesimtun0: 56(84)
       bytes of data.
2  64 bytes from 8.8.8.8: icmp_seq=1 ttl=61 time=15.1 ms
3  64 bytes from 8.8.8.8: icmp_seq=2 ttl=61 time=530 ms
4  .
5  .
6  .
7  64 bytes from 8.8.8.8: icmp_seq=58 ttl=61 time=20.4 ms
8  64 bytes from 8.8.8.8: icmp_seq=59 ttl=61 time=25.4 ms
9  64 bytes from 8.8.8.8: icmp_seq=60 ttl=61 time=19.6 ms
10
11 --- 8.8.8.8 ping statistics ---
12 60 packets transmitted, 60 received, 0% packet loss, time
       59066ms
13 rtt min/avg/max/mdev = 15.191/194.511/984.863/317.386 ms
```

Os dados foram obtidos durante a leitura de 60 segundos, que capturou valores a cada 1 segundo, os valores obtidos foram separados e tratados no Apêndice B. Ressalta que os 30 dispositivos foram executados simultaneamente utilizando o código anexado em C

### 4.3.1 1 Slice

Na configuração de 1 slice com as prioridades em alta, média e baixa são ilustradas nas figuras 8, 9 e 10, que na qual é possível observar que os 30 dispositivos apresentam latências e curvas acentuadas.

Figura 8 – 1 Slice e configuração do QoS em alta prioridade



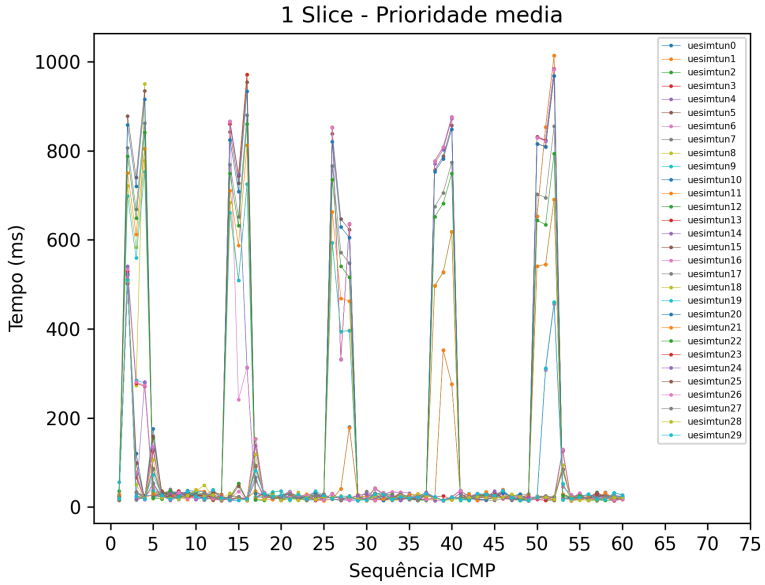

Fonte: O autor.

Observa-se que, nas sequências ICMP da Figura 8 de 0 a 5 e de 23 a 25, há algumas ocorrências de alta latência em algumas interfaces, resultando em um consumo elevado de largura de banda. Isso é evidenciado por uma sequência em forma de Sawtooth, indicando uma saturação da banda em algumas interfaces.

Na configuração com prioridade média, é possível observar na Figura 9 um comportamento semelhante ao da slice em alta prioridade. No início do gráfico, na sequência de 0 a 5, algumas interfaces apresentam um desempenho superior em termos de latência em relação às outras. No entanto, nos pontos seguintes, podemos observar uma sequência semelhante à da prioridade alta. Isso ocorre porque todas as interfaces estão agrupadas em uma única slice, sem uma divisão igual de recursos, como demonstrado nos resultados do netperf. A última configuração com 1 slice é de prioridade baixa, nesse caso podemos notar na Figura 10 que comparado aos dois primeiros casos, há um início lento em todas as interfaces, isso pode ser analisado nos pontos de 0 a 2, em que o tempo de resposta varia de 500 ms a 600 ms, e o restante dos resultados é notado o comportamento igual aos testes de prioridade alta e média.

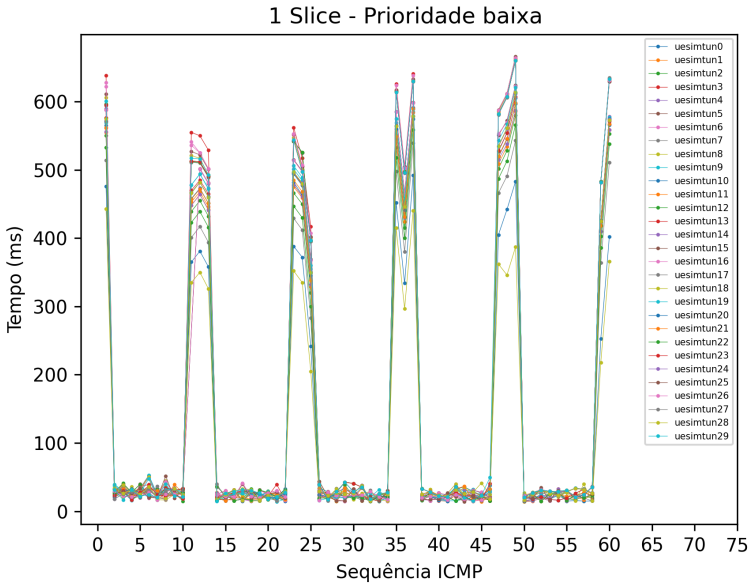
Ressalta-se o tempo de resposta de prioridade baixa, o valor médio da latência entre

Figura 9 – 1 Slice e configuração do QoS em média prioridade




Fonte: O autor.

Figura 10 – 1 Slice e configuração do QoS em baixa prioridade



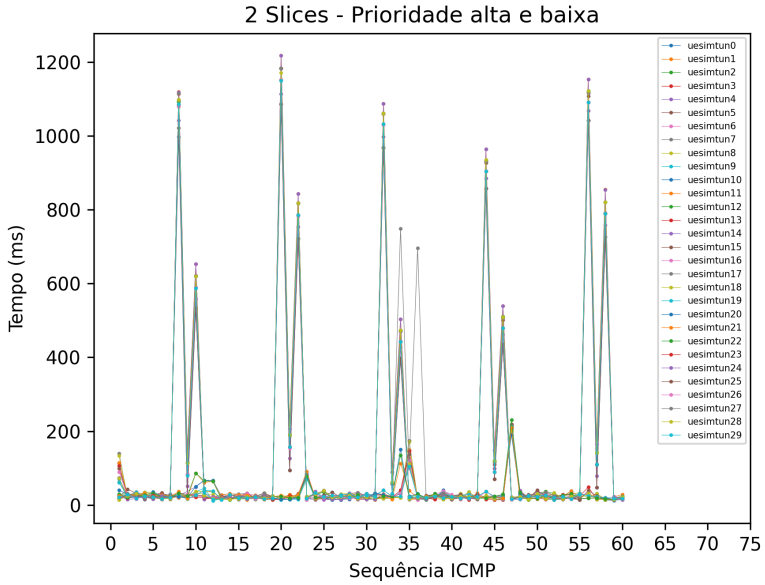

Fonte: O autor.

os 30 dispositivos resultaram no valor de 114,256s, e o valor médio na prioridade alta e média é de 137,963s e 190,511s respectivamente, afirmando assim uma melhor estabilidade na prioridade baixa.

### 4.3.2 2 Slices

Na configuração de duas slices, podemos avaliar uma diferença comparada com a configuração de 1 slice pela Figura 11

Figura 11 – 2 Slices e configuração do QoS em alta e baixa prioridade



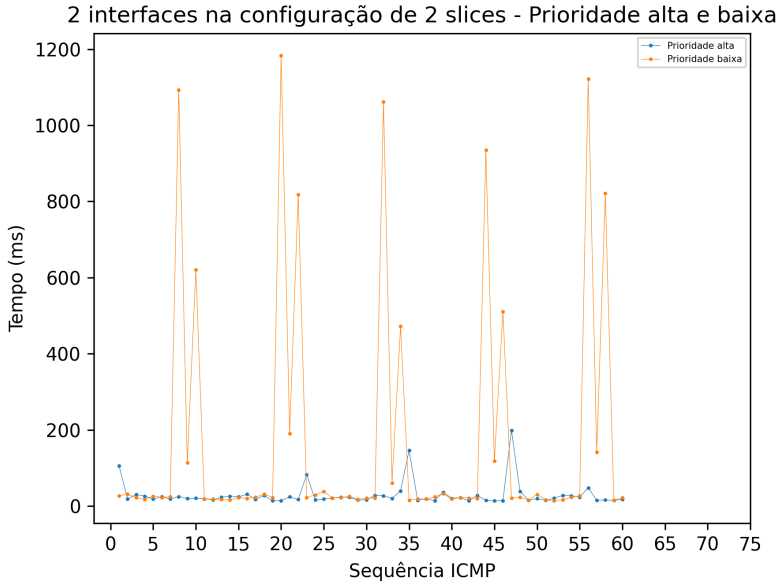

Fonte: O autor.

Na Figura 12 foi selecionado duas interfaces, uma de cada slice para a melhor análise, nota-se que a prioridade baixa possui uma latência superior a da prioridade alta, que por sua vez, garante o seu isolamento comparado aos resultados de uma slice.

Para as slices configuradas com QoS de prioridade média e alta, considerando apenas duas interfaces para uma análise mais detalhada, podemos observar na Figura 13 que a slice com prioridade alta apresenta uma latência inferior em comparação com a slice de prioridade média. Isso evidencia a eficácia da divisão de slices com diferentes níveis de prioridade, onde a slice de maior prioridade é capaz de obter uma latência menor em relação à slice de prioridade média.

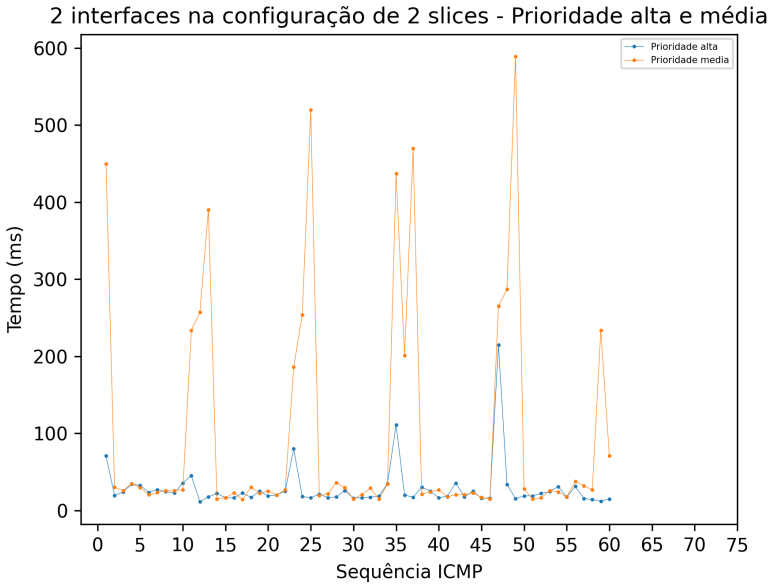
Na próxima configuração, foram realizados testes com duas slices configuradas com alta prioridade, selecionando apenas duas interfaces para uma melhor comparação. O resultado evidenciou que ambas as slices possuem prioridades iguais, resultando em uma latência alta. Isso indica que nenhuma das slices tem uma vantagem em termos de prioridade sobre a outra, resultando em um comportamento de latência semelhante e elevada para ambas, conforme ilustrado na Figura 14.

Figura 12 – 2 interfaces selecionados em prioridade alta e baixa




Fonte: O autor.

Figura 13 – 2 interfaces selecionados em prioridade alta e média

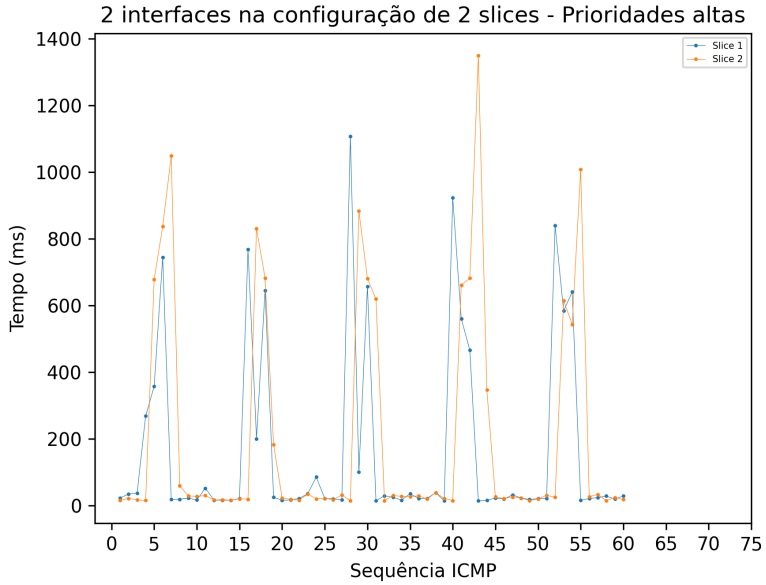



Fonte: O autor.

### 4.3.3 3 Slices

Na configuração de 3 slices, foram realizados testes com três slices configuradas com prioridades alta, média e baixa. Na Figura 15, é possível observar os 30 dispositivos

Figura 14 – 2 interfaces selecionados em prioridades altas



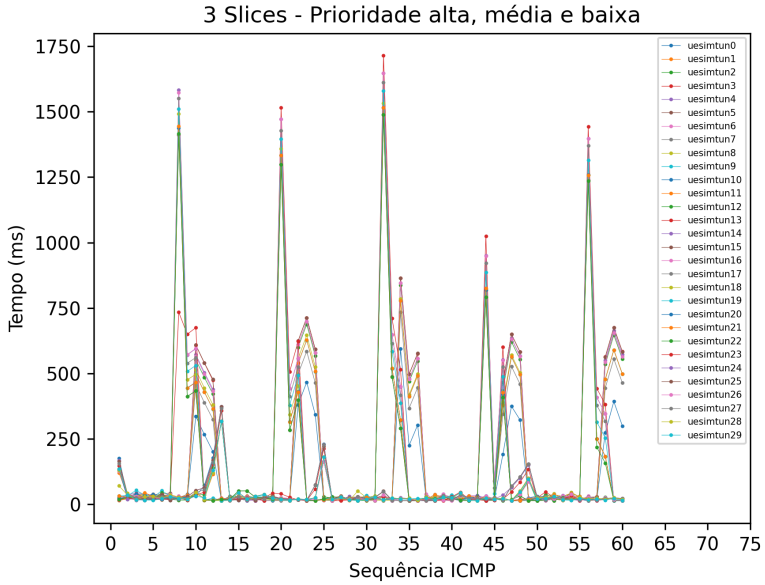

Fonte: O autor.

iniciando simultaneamente, revelando uma diferença significativa entre as interfaces. Ao selecionar uma interface de cada slice, conforme mostrado na Figura 16, fica evidente que a interface com prioridade alta apresenta uma latência baixa em comparação com as outras interfaces. Além disso, a interface com prioridade média exibe uma latência inferior em relação à interface com baixa prioridade.

Para as slices configuradas com prioridade alta e duas com prioridade média, selecionamos duas interfaces para uma análise mais detalhada como mostra a 17. Podemos observar um conflito entre as duas interfaces de prioridade média, o que confirma a proposta de que, quando configuradas com mesma prioridade, nenhuma tem prioridade sobre a outra. Além disso, a interface com prioridade alta mantém uma latência baixa, evidenciando a prioridade em relação às outras interfaces. Também é possível observar a curva das sequências ICMP de 10 a 15 em relação à saturação da banda, onde é percebido um impacto mínimo na interface de alta prioridade. Nota-se que o conflito de recursos pelas duas slices em médias prioridades é inferior que a baixa prioridade discutido na Figura 16.

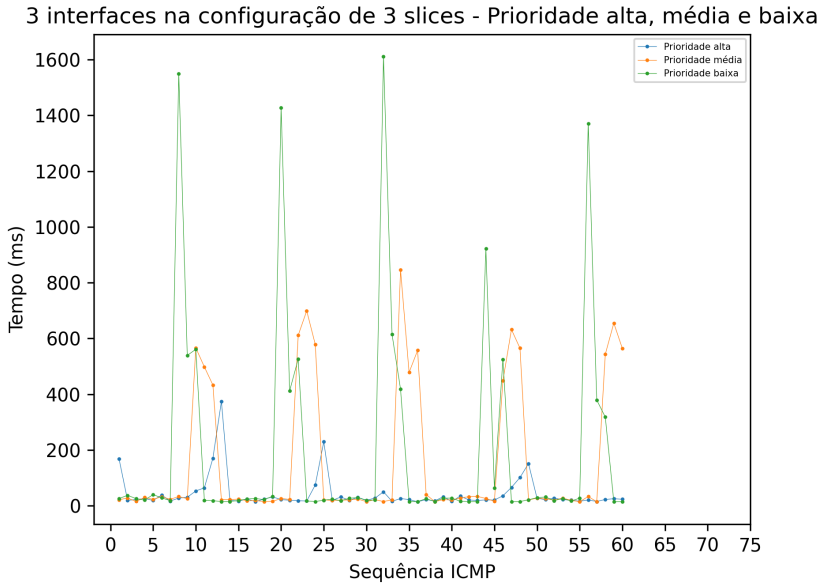
Por fim, a última configuração é com 3 prioridades altas, resultando em um comportamento semelhante ao teste realizado com duas slices em alta prioridade. Podemos analisar que os dispositivos em slice 1 obteve uma prioridade superior quanto as outras duas mas o seu valor médio de latência é superior comparado com outros testes de slices. Além disso, observa-se uma diminuição significativa no valor da latência, levando em consideração a distribuição do número de dispositivos.

Figura 15 – 3 Slices e configuração do QoS em alta, média e baixa prioridade




Fonte: O autor.

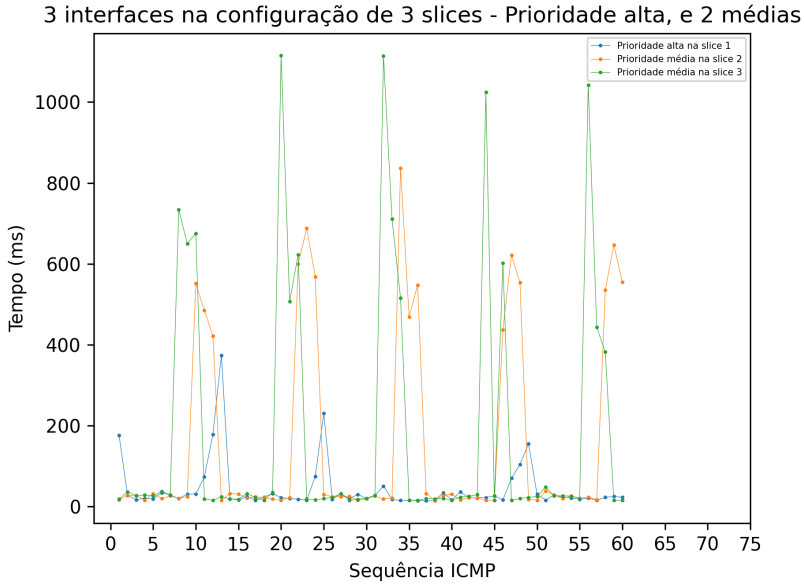
Figura 16 – 3 interfaces selecionados em prioridade alta, média e baixa




Fonte: O autor.

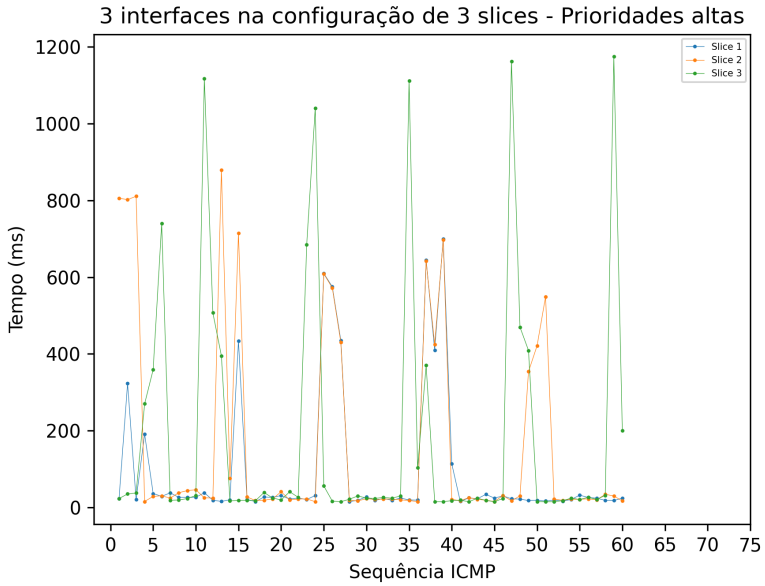
Os valores médios do comando ping podem ser analisados na Tabela 6. Para 1 slice, observa-se um valor alto, como esperado, uma vez que os 30 dispositivos estão em apenas uma slice. No caso da duas slice, os valores das slices de prioridade baixa e média

Figura 17 – 3 Interfaces selecionados em prioridade alta e duas médias




Fonte: O autor.

Figura 18 – 3 Slices e configuração do QoS em 3 prioridades altas




Fonte: O autor.

são impactados devido à sua prioridade inferior em relação à slice de prioridade alta. Na configuração de 2 slices com prioridade alta, o valor é esperado, já que as slices não têm prioridade uma sobre a outra, resultando em um valor alto. Em relação aos valores com 3

slices, eles são condizentes com a prioridade de QoS com exceção da configuração em 3 prioridades altas, pode-se afirmar que uma slice apresentou um valor média do latência abaixo em comparação com as outras.

Tabela 6 – Valor médio de *ping* em ms de cada teste realizado.

| **Slice(s)** | **Prioridade** | **Valor médio** |
|---|---|---|
| 1 | Alta | 137,963 |
| | Média | 190,511 |
| | Baixa | 114,256 |
| 2 | Alta e Baixa | 28,349 e 255,793 |
| | Alta e Média | 28,023 e 108,682 |
| | 2 Altas | 221,350 e 232,491 |
| 3 | Alta, Média e Baixa | 21,140 134,492 e 209,746 |
| | 3 Altas | 71,224 177,667 e 145,254 |
| | Alta e duas Médias | 24,221 198,555 e 188,536 |

Fonte: O Autor.

## 4.4 CONSUMO DE RECURSOS DA MÁQUINA VIRTUAL

O consumo da memória RAM e CPU pode ser monitorado pelo terminal da máquina virtual pelos comandos do código fonte 4.2, o consumo foi medido na da VM2. Nos testes, o consumo inicial de CPU foi de 20% e o consumo de memória RAM foi de 10GB. Ao configurar mais UPFs e slices, o consumo aumentou para 68% de CPU e 11,9Gb de memória RAM.

Código Fonte 4.2 – Comando executado para análise de recursos de CPU e RAM

```
1 $ cat /proc/meminfo
2 $ cat /proc/cpuinfo
```

Ao iniciar os testes com dispositivos e utilizar os comandos de netperf e ping, o uso de CPU e RAM apresentou um aumento gradual. O uso de CPU atingiu uma média de 70% e o consumo de memória RAM foi de 13,5 GB.

Ao conectar a segunda slice e realizar os testes nos dispositivos, o uso médio de CPU aumentou para 72% e o consumo médio de memória RAM foi de 14 GB. Por fim, ao adicionar a terceira e última slice, o uso médio de CPU foi de 76% e o consumo médio de memória RAM foi de 14,1 GB, podendo afirmar o valor médio elevado em throughput nos testes efetuados em 2 e 3 slices.

É importante ressaltar que a documentação do (FOUNDATION, 2020) estabelece requisitos mínimos a serem atendidos ao simular o sistema.

# 5 CONCLUSÃO

Este trabalho resultou em um estudo relevante sobre a QoS em *network slicing* no contexto do 5G, especialmente em casos que envolvem 30 dispositivos enviando informações simultaneamente pela rede, passando pelo core. Foi possível observar como os sistemas 5G operam e se comportam em ambientes simulados, compreendendo o impacto dessa tecnologia nas soluções de rede em um futuro próximo. O simulador Aether-in-a-box, implementado em contêineres, mostrou-se altamente versátil e adaptável, com uma interface gráfica intuitiva e de configuração simples. Foi possível adicionar dispositivos, configurar parâmetros de QoS, separar dispositivos em *slices* e adicionar ao UPF, mas ao decorrer do experimento, o software teve instabilidades devido a atualizações do sistema. O simulador UERANSIM estabeleceu a conexão dos equipamentos simulados e da antena 5G de forma intuitiva e prática.

O Netperf provou ser uma ferramenta versátil para medir o *throughput* e a perda de pacotes. Nos testes de duas *slices* com configuração de prioridade alta e baixa, houve um valor inesperado na interface uesimtun1, com um *throughput* acima da média. Na configuração de 3 *slices* com prioridade alta e duas médias, pode haver uma limitação no hardware do hospedeiro. O comando ping, por sua vez, foi executado de forma simples e intuitiva, e sua resposta foi tratada para gerar um gráfico de tempo baseado na sequência ICMP.

Os resultados obtidos, embora não possam ser generalizados para todas as aplicações devido à sua complexidade, não foram conforme o esperado na configuração de 1 slice com prioridade baixa, onde a latência média foi inferior nesse teste. Em comparação com a configuração de duas *slices* ou três *slices*, onde a prioridade baixa está presente, a latência foi menor do que o esperado, e na configuração de 3 *slices* em prioridade alta, avaliou-se que uma slice obteve uma latência inferior comparado com as outras *slices* na mesma prioridade,

Em relação ao testes do *throughput* tem diferença positiva quando aumenta o número de *slices*, mas os resultados não foram uniformes conforme esperado, na configuração de 3 *slices* com prioridade alta e duas médias, a prioridade alta não obteve um valor de *throughput* superior quanto aos *slices* de prioridade média. Destaca-se este trabalho como material de estudo para aqueles interessados em analisar a latência entre as *slices* de rede 5G e avaliar o QoS.

Com relação aos objetivos propostos, este trabalho demonstrou que os resultados do *network slicing* obteve uma baixa latência e um valor de *throughput* relacionado a prioridade . Foi possível compreender de maneira mais detalhada o funcionamento e a aplicação das redes 5G, além de entender os seus benefícios.

## 5.1 PESQUISAS FUTURAS

Para pesquisas futuras, sugere-se implementar o 5G em ambiente real, com foco em simulação de *network slicing* com dispositivos físicos. Além disso, recomenda-se utilizar um sistema com requisitos mínimos para atender os requisitos do sistema para que possa utilizar um número maior de dispositivos.

Também recomenda-se utilizar outro simulador para aplicar a pesquisa, visto que o Aether-in-a-box apresentou instabilidades ao decorrer do experimento.

# REFERÊNCIAS

3GP. **Release 16**. 2020. Disponível em: https://www.3gpp.org/specifications-technologies/releases/release-16. Acesso em: 6 jun. 2023.

3GPP. **5G QoS characteristics**. 2022. Disponível em: https://www.tech-invite.com/3m23/toc/tinv-3gpp-23-501%5C_za.html. Acesso em: 6 jun. 2023.

3GPP. **Digital cellular telecommunications system**. 2011. Disponível em: https://www.etsi.org/deliver/etsi%5C_ts/123200%5C_123299/123203/09.09.00%5C_60/ts%5C_123203v090900p.pdf. Acesso em: 6 jun. 2023.

3GPP. **Standardized 5QI to QoS characteristics mapping**. 2016. Disponível em: https://www.tech-invite.com/3m23/toc/tinv-3gpp-23-501%5C_zb.html%5C#e-5-7-4. Acesso em: 6 jun. 2023.

3GPP. **TSGs#100, June 2023**. 2023. Disponível em: https://www.3gpp.org/release-15. Acesso em: 6 jun. 2023.

ANDRÉ QUEIROZ DÉBORA PINA, Eduardo Mello. **Arquiteturas para o futuro**. 2016. Disponível em: https://www.gta.ufrj.br/ensino/eel879/trabalhos%5C_vf%5C_2015%5C_2/5G/arquiteturas.html. Acesso em: 6 jun. 2023.

BRASIL, Linux Force. **Comandos Linux – Comando visudo**. 2019. Disponível em: https://www.linuxforce.com.br/comandos-linux/comandos-linux-comando-visudo/. Acesso em: 6 jun. 2023.

DEVOPEDIA. **5G Quality of Service**. 2021. Disponível em: https://devopedia.org/5g-quality-of-service. Acesso em: 6 jun. 2023.

FOUNDATION, Open Networking. **Aether-in-a-Box for Developers**. 2020. Disponível em: https://docs.aetherproject.org/master/developer/aiabhw5g.html. Acesso em: 6 jun. 2023.

FOUNDATION, Open Networking. **SD-Core as a Cloud Managed Service**. 2021. Disponível em:

https://docs.sd-core.opennetworking.org/sdcore-1.1/overview/architecture.html. Acesso em: 6 jun. 2023.

FOUNDATION, Open Networking. **UERANSIM Deployment Guide**. 2021. Disponível em: https://docs.sd-core.opennetworking.org/master/deployment/deploymentueransim.html. Acesso em: 6 jun. 2023.

GÜNGÖR, ALİ. **UERANSIM**. 2022. Disponível em: https://github.com/aligungr/UERANSIM. Acesso em: 6 jun. 2023.

KRAUS, Dener. **Computação de borda para indústria utilizando a rede 5G**. 2021. Universidade Federal de Santa Catarina.

KUBERNETES. **Kubernetes**. 2023. Disponível em: https://kubernetes.io/pt-br/docs/home/. Acesso em: 6 jun. 2023.

MAILER, Christian. **Plataforma de CORE 5G em nuvem para disponibilização de funções de rede como serviço**. 2020. Universidade Federal de Santa Catarina.

NETWORKS, Juniper. **O que é 5G**. 2023. Disponível em: https://www.juniper.net/br/pt/research-topics/what-is-5g.html. Acesso em: 6 jun. 2023.

PAGE, Linux man. **netperf**. 2023. Disponível em: https://linux.die.net/man/1/netperf. Acesso em: 6 jun. 2023.

PENTTINEN, Jyrki T. J. **5G Explained: Security and Deployment of Advanced Mobile Communications**. [*S.l.*]: John Wiley & Sons Ltd, 2019. ISBN 9781119275688; 1119275687.

PEREIRA, Luiz Augusto Melo. **Sistemas Fiber-wireless 5G NR**. 2020. o Instituto Nacional de Telecomunicações.

PUJOLLE, Guy. **Software Networks Virtualization, SDN, 5G, and Security**. 2nd ed. [*S.l.*]: John Wiley & Sons, Incorporated, 2020. ISBN 9781119694724; 1119694728; 9781786304582; 1786304589.

SILVA, Gabriel Henrique Davanço. **Classificação de tráfego por classes de serviço do núcleo 5G**. 2022. Universidade Federal de Santa Catarina.

STEFAN ROMMER PETER HEDMAN, Magnus Olsson. **5G Core Networks Powering Digitalization**. 1. ed. [*S.l.*]: Academic Press, 2019. ISBN 0081030096; 9780081030097.

TANENBAUM, Andrew S. **Computer networks**. 4. ed. [*S.l.*]: Prentice Hall, 2002. ISBN 0130661023; 9780130661029.

VIRTUALBOX. **virtualbox**. 2023. Disponível em: https://www.virtualbox.org/. Acesso em: 6 jun. 2023.

VSCODE. **Remote SSH**. 2023. Disponível em: https://code.visualstudio.com/docs/remote/ssh. Acesso em: 6 jun. 2023.

# APÊNDICE A – Código em python para calcular a média de throughput

Código Fonte A.1 – Código para calcular a média de throughput

```
1  import glob
2
3  def extrair_valores(arquivo):
4      with open(arquivo, 'r') as file:
5          linhas = file.readlines()
6
7      throughput = None
8
9      for linha in linhas[::-1]:
10         if linha.strip():
11             valores = linha.split()
12             try:
13                 throughput = float(valores[-1])
14                 break
15             except ValueError:
16                 continue
17
18     return throughput
19
20 arquivos = glob.glob('*.txt')
21
22 throughputs = []
23
24 for arquivo in arquivos:
25     throughput = extrair_valores(arquivo)
26     if throughput is not None:
27         throughputs.append(throughput)
28
29 media_throughput = sum(throughputs) / len(throughputs)
30 with open("resultado.txt", "w") as file:
31     file.write(resultado)
```

# APÊNDICE B – Código em python para a geração do gráfico do comando ping

Código Fonte B.1 – Código em python para a geração do gráfico do comando ping

```
1  import glob
2  import re
3  import matplotlib.pyplot as plt
4  import numpy as np
5
6  icmp_seq_valores = []
7  time_valores = []
8
9  arquivos = glob.glob('*.txt')
10
11 cores = plt.cm.get_cmap('tab10').colors
12
13 legenda_arquivos = [re.search(r'uesimtun(\d+)', arquivo).
       group(0) for arquivo in arquivos]
14
15 for i, arquivo in enumerate(arquivos):
16     with open(arquivo, 'r') as file:
17         linhas = file.readlines()
18
19     for linha in linhas:
20         match = re.search(r'icmp_seq=(\d+)', linha)
21         if match:
22             icmp_seq_valores.append(int(match.group(1)))
23
24         match = re.search(r'time=(\d+\.?\d*)', linha)
25         if match:
26             time_valores.append(float(match.group(1)))
27
28     plt.plot(icmp_seq_valores, time_valores, marker='.',
          linewidth=0.4, color=cores[i % len(cores)])
29
30     icmp_seq_valores.clear()
31     time_valores.clear()
32
33 plt.xlabel('Sequencia ICMP')
```

```
34 plt.ylabel('Tempo (ms)')
35 plt.title('Fatias - Prioridade ')
36
37 plt.legend(legenda_arquivos, fontsize='small',  loc='upper
      right')
38
39 plt.xticks(ticks=(range(0, 70, 1)))
40
41 plt.savefig('.png', dpi=300)
```

# APÊNDICE C – Arquivo contendo script em bash para realizar o comando de ping e nerperf para 30 dispositivos

Código Fonte C.1 – Arquivo contendo script em bash para realizar o comando de ping e nerperf para 30 dispositivos

```
#!/bin/bash

num_interfaces=30

for ((i = 0; i < num_interfaces; i++)); do
    interface="uesimtun$i"
    ping_output_file="ping_results_$interface.txt"
    netperf_output_file="netperf_results_$interface.txt"
    interfaces[$i]=$interface
    ping_output_files[$i]=$ping_output_file
    netperf_output_files[$i]=$netperf_output_file
done

function run_ping() {
    interface=$1
    output_file=$2
    sudo ping 8.8.8.8 -I "$interface" -c 60 > "$output_file"
}

function run_netperf() {
    interface=$1
    output_file=$2
    sudo netperf -H 192.168.56.120 -l 10 -B "$interface" > "$
       output_file"
}

for i in "${!interfaces[@]}"; do
    interface="${interfaces[$i]}"
    ping_output_file="${ping_output_files[$i]}"
    netperf_output_file="${netperf_output_files[$i]}"

    run_ping "$interface" "$ping_output_file" &
    run_netperf "$interface" "$netperf_output_file" &
done
```

```
34
35 wait
36
37 for file in "${ping_output_files[@]}" "${netperf_output_files
      [@]}"; do
38     echo "$file"
39 done
```

# APÊNDICE D – Tabelas contendo resultados netperf realizados

Tabela 7 – Resultados de Throughput($10^6$bits/sec) com configuração de 1 Slice.

| **Interface** | **Proridade Alta** | **Proridade Média** | **Proridade Baixa** |
|---|---|---|---|
| uesimtun0 | 71,78 | 52,03 | 50,64 |
| uesimtun1 | 65,34 | 69,86 | 48,30 |
| uesimtun2 | 60,83 | 54,44 | 47,93 |
| uesimtun3 | 67,93 | 55,40 | 52,10 |
| uesimtun4 | 56,72 | 57,89 | 51,74 |
| uesimtun5 | 60,41 | 44,10 | 49,56 |
| uesimtun6 | 59,09 | 51,31 | 48,12 |
| uesimtun7 | 64,48 | 45,34 | 48,67 |
| uesimtun8 | 60,30 | 50,48 | 44,91 |
| uesimtun9 | 61,41 | 46,08 | 52,44 |
| uesimtun10 | 52,13 | 54,17 | 55,23 |
| uesimtun11 | 57,87 | 50,03 | 54,10 |
| uesimtun12 | 60,72 | 56,97 | 50,92 |
| uesimtun13 | 62,17 | 50,14 | 48,93 |
| uesimtun14 | 53,39 | 51,75 | 49,97 |
| uesimtun15 | 56,50 | 43,14 | 53,69 |
| uesimtun16 | 63,36 | 50,70 | 46,84 |
| uesimtun17 | 56,18 | 52,27 | 53,92 |
| uesimtun18 | 60,88 | 52,39 | 53,71 |
| uesimtun19 | 60,93 | 45,42 | 50,13 |
| uesimtun20 | 55,26 | 55,06 | 45,41 |
| uesimtun21 | 60,07 | 51,22 | 46,26 |
| uesimtun22 | 64,12 | 49,70 | 40,95 |
| uesimtun23 | 56,71 | 54,63 | 51,15 |
| uesimtun24 | 62,09 | 51,61 | 49,59 |
| uesimtun25 | 62,62 | 47,08 | 47,24 |
| uesimtun26 | 66,65 | 54,68 | 47,55 |
| uesimtun27 | 61,90 | 53,56 | 52,43 |
| uesimtun28 | 60,34 | 49,89 | 46,51 |
| uesimtun29 | 55,91 | 53,32 | 54,33 |
| Média | 60,603 | 51,822 | 49,776 |

Fonte: O Autor.

Tabela 8 – Resultados de Throughput($10^6$bits/sec) com configuração de 2 Slices com prioridade alta, média e baixa.

| **Interface** | **Alta e baixa** | **Alta e Média** | **Alta e Alta** |
|---|---|---|---|
| uesimtun0 | 56,52 | 65,37 | 58,01 |
| uesimtun1 | 293,31 | 63,36 | 61,69 |
| uesimtun2 | 69,49 | 68,94 | 67,39 |
| uesimtun3 | 57,19 | 58,79 | 54,81 |
| uesimtun4 | 63,90 | 64,36 | 57,03 |
| uesimtun5 | 65,37 | 59,37 | 58,40 |
| uesimtun6 | 56,59 | 63,50 | 59,57 |
| uesimtun7 | 55,60 | 61,44 | 54,04 |
| uesimtun8 | 61,87 | 72,79 | 65,89 |
| uesimtun9 | 52,56 | 63,06 | 58,41 |
| uesimtun10 | 60,05 | 68,53 | 58,56 |
| uesimtun11 | 63,31 | 68,97 | 54,48 |
| uesimtun12 | 59,08 | 66,43 | 57,59 |
| uesimtun13 | 63,41 | 62,99 | 60,38 |
| uesimtun14 | 58,37 | 63,21 | 56,75 |
| uesimtun15 | 63,35 | 71,09 | 66,78 |
| uesimtun16 | 65,62 | 73,18 | 76,29 |
| uesimtun17 | 56,89 | 65,46 | 60,19 |
| uesimtun18 | 64,98 | 62,11 | 58,31 |
| uesimtun19 | 63,77 | 61,45 | 66,81 |
| uesimtun20 | 54,46 | 60,57 | 62,39 |
| uesimtun21 | 63,71 | 66,77 | 60,00 |
| uesimtun22 | 64,19 | 66,70 | 55,14 |
| uesimtun23 | 63,31 | 67,67 | 60,78 |
| uesimtun24 | 56,25 | 65,56 | 58,10 |
| uesimtun25 | 64,15 | 58,82 | 52,51 |
| uesimtun26 | 57,62 | 70,35 | 54,10 |
| uesimtun27 | 58,60 | 61,85 | 70,43 |
| uesimtun28 | 52,76 | 64,53 | 64,24 |
| uesimtun29 | 71,02 | 70,89 | 63,76 |
| Média | 68,576 | 65,270 | 60,428 |

Fonte: O Autor.

Tabela 9 – Resultados de Throughput($10^6$bits/sec) com configuração de 3 Slices com prioridade alta, média e baixa.

| **Interface** | **Alta, Média e Baixa** | **3 Altas** | **Alta e duas Médias** |
|---|---|---|---|
| uesimtun0 | 61,07 | 70,28 | 61,07 |
| uesimtun1 | 64,40 | 67,36 | 64,40 |
| uesimtun2 | 74,30 | 71,86 | 74,30 |
| uesimtun3 | 68,86 | 64,42 | 68,86 |
| uesimtun4 | 69,67 | 71,92 | 69,67 |
| uesimtun5 | 68,05 | 73,55 | 68,05 |
| uesimtun6 | 61,30 | 73,84 | 61,30 |
| uesimtun7 | 294,54 | 74,70 | 71,54 |
| uesimtun8 | 71,31 | 58,61 | 71,31 |
| uesimtun9 | 67,03 | 69,94 | 67,03 |
| uesimtun10 | 68,73 | 68,31 | 68,73 |
| uesimtun11 | 61,76 | 78,38 | 61,76 |
| uesimtun12 | 82,56 | 69,35 | 82,56 |
| uesimtun13 | 73,27 | 60,51 | 73,27 |
| uesimtun14 | 62,16 | 71,43 | 62,16 |
| uesimtun15 | 71,68 | 69,38 | 71,68 |
| uesimtun16 | 73,21 | 72,71 | 73,21 |
| uesimtun17 | 81,21 | 62,88 | 81,21 |
| uesimtun18 | 56,12 | 73,89 | 56,12 |
| uesimtun19 | 64,88 | 67,12 | 64,88 |
| uesimtun20 | 73,90 | 68,64 | 73,90 |
| uesimtun21 | 62,19 | 70,71 | 62,19 |
| uesimtun22 | 65,73 | 73,41 | 65,73 |
| uesimtun23 | 58,53 | 60,89 | 58,53 |
| uesimtun24 | 62,32 | 65,98 | 62,32 |
| uesimtun25 | 74,30 | 68,59 | 74,30 |
| uesimtun26 | 58,57 | 73,56 | 58,57 |
| uesimtun27 | 70,80 | 67,50 | 70,80 |
| uesimtun28 | 70,16 | 64,93 | 70,16 |
| uesimtun29 | 66,49 | 68,25 | 66,49 |
| Média | 75,303 | 61,17 | 67,86 |

Fonte: O Autor.

# ANEXO A – Mapeamento padronizado de 5QI para características de QoS

Quadro 1 – Mapeamento padronizado de 5QI para características de QoS com recurso GBR.

| **Aether** | **Valor 5QI** | **Nível de prioridade** | **Limite de atraso de pacote** | **Taxa de erro de pacote** | **Volume máximo de dados** | **Janela de média** | **Exemplo** |
|---|---|---|---|---|---|---|---|
| 1 | 1 | 20 | 100 ms | $10^{-2}$ | N/A | N/A | Voz |
| 2 | 2 | 40 | 150 ms | $10^{-3}$ | N/A | N/A | Live Streaming |
| 3 | 3 | 30 | 50 ms | $10^{-3}$ | N/A | N/A | Jogo em tempo real |
| 4 | 4 | 50 | 300 ms | $10^{-6}$ | N/A | N/A | Vídeo Não Conversacional (Streaming em Buffer) |
| 5 | 65 | 7 | 75 ms | $10^{-2}$ | N/A | N/A | Voz em Tempo Real Crítica para a Missão Crítica (Push to Talk) |
| 6 | 66 | 20 | 100 ms | $10^{-2}$ | N/A | N/A | Voz em Tempo Real Não Crítica para a Missão (por exemplo, Push to Talk - PTT Não Crítico para a Missão) |
| 7 | 67 | 15 | 100 ms | $10^{-3}$ | N/A | N/A | *Mission Critical Video user plane* |
| 8 | 75 | 25 | 50 ms | $10^{-2}$ | N/A | N/A | Mensagens V2X |
| 9 | 71 | 56 | 150 ms | $10^{-6}$ | N/A | N/A | *Uplink Streaming* |
| 10 | 72 | 56 | 300 ms | $10^{-4}$ | N/A | N/A | *Uplink Streaming* |
| 11 | 73 | 56 | 300 ms | $10^{-8}$ | N/A | N/A | *Uplink Streaming* |
| 12 | 74 | 56 | 500 ms | $10^{-8}$ | N/A | N/A | *Uplink Streaming* |
| 13 | 76 | 56 | 500 ms | $10^{-4}$ | N/A | N/A | *Uplink Streaming* |

Fonte: Adaptado de (3GPP, 2016).

Quadro 2 – Mapeamento padronizado de 5QI para características de QoS com recurso Non-GBR.

| **Aether** | **Valor 5QI** | **Nível de prioridade** | **Limite de atraso de pacote** | **Taxa de erro de pacote** | **Volume máximo de dados** | **Janela de média** | **Exemplo** |
|---|---|---|---|---|---|---|---|
| 14 | 5 | 10 | 100 ms | $10^{-6}$ | N/A | N/A | Sinalização IMS |
| 15 | 6 | 60 | 300 ms | $10^{-6}$ | N/A | N/A | Vídeo (transmissão ao vivo) |
| 16 | 7 | 70 | 100 ms | $10^{-6}$ | N/A | N/A | Voz, vídeo (transmissão ao vivo) jogos interativos |
| 17 | 8 | 80 | 300 ms | $10^{-6}$ | N/A | N/A | Vídeo (transmissão ao vivo) baseado em TCP (por exemplo, www, e-mail, chat, ftp, compartilhamento de arquivos p2p, vídeo progressivo) |
| 18 | 9 | 90 | 300 ms | $10^{-6}$ | N/A | N/A | Vídeo (transmissão ao vivo) baseado em TCP (por exemplo, www, e-mail, chat, ftp, compartilhamento de arquivos p2p, vídeo progressivo) |
| 19 | 10 | 90 | 1100 ms | $10^{-6}$ | N/A | N/A | Vídeo (transmissão ao vivo) |
| 20 | 69 | 5 | 60 ms | $10^{-6}$ | N/A | N/A | Sinalização sensível ao atraso de missão crítica |
| 21 | 70 | 55 | 200 ms | $10^{-6}$ | N/A | N/A | Dados de Missão Crítica |
| 22 | 79 | 65 | 50 ms | $10^{-2}$ | N/A | N/A | Mensagens V2X |
| 23 | 80 | 68 | 10 ms | $10^{-6}$ | N/A | N/A | Aplicações eMBB de baixa latência; realidade aumentada |

Fonte: Adaptado de (3GPP, 2016).

Quadro 3 – Mapeamento padronizado de 5QI para características de QoS com recurso Delay Crital GBR.

| **Aether** | **Valor 5QI** | **Nível de prioridade** | **Limite de atraso de pacote** | **Taxa de erro de pacote** | **Volume máximo de dados** | **Janela de média** | **Exemplo** |
|---|---|---|---|---|---|---|---|
| 24 | 82 | 19 | 10 ms | $10^{-4}$ | 255 bytes | 2000 ms | Automação Discreta |
| 25 | 83 | 22 | 10 ms | $10^{-4}$ | 1354 bytes | 2000 ms | Automação Discreta; Mensagens Vehicle-to-Everything (V2X) |
| 26 | 84 | 24 | 30 ms | $10^{-5}$ | 1354 bytes | 2000 ms | Sistemas de transporte inteligentes |
| 27 | 85 | 21 | 5 ms | $10^{-5}$ | 255 bytes | 2000 ms | Distribuição de energia – alta voltagem; mensagens V2X |
| 28 | 86 | 18 | 5 ms | $10^{-4}$ | 1354 bytes | 2000 ms | Mensagens V2X |
| 29 | 87 | 25 | 5 ms | $10^{-3}$ | 500 bytes | 2000 ms | Serviço interativo – dados de rastreamento de movimento |
| 30 | 88 | 25 | 10 ms | $10^{-3}$ | 1125 bytes | 2000 ms | Serviço interativo – dados de rastreamento de movimento |
| 31 | 89 | 25 | 15 ms | $10^{-4}$ | 17000 bytes | 2000 ms | Conteúdo visual para renderização em nuvem/borda |
| 32 | 90 | 25 | 20 ms | $10^{-4}$ | 63000 bytes | 2000 ms | Conteúdo visual para renderização em nuvem/borda |

Fonte: Adaptado de (3GPP, 2016).