FH AACHEN
UNIVERSITY OF APPLIED SCIENCES

# Fachhochschule Aachen, Campus Jülich

Faculty 09 – Medizintechnik und Technomathematik
Angewandte Mathematik und Informatik

## Seminar Thesis

# Analysis and Design of an Environment for Systematic 5G Network Performance Testing

*Author:*
Leonard Esser

*Matriculation Number:*
3277392

Aachen
December 15, 2022

# Confidentiality Clause

This thesis contains confidential data of Ericsson GmbH. This work may only be made available to the first and second reviewers and authorized members of the board of examiners. Any publication and duplication of this thesis - even in part - is prohibited. An inspection of this work by third parties requires the expressed permission of the author and Ericsson GmbH.

Prof. Dr. Philipp Rohde

Dr. Thorsten Lohmar

Aachen, December 2022

ERICSSON

# Statutory Declaration

I hereby assure that this thesis with the title

**"Analysis and Design of an Environment for Systematic 5G Network Performance Testing"**

has been produced and written on my own and has not been part of any other study or examination performance. No other means, sources or resources have been used, other than the listed ones. Every part of my work that contains quotations is marked accordingly and can be found in the bibliography.
I vow to keep a copy of this thesis for five years and to hand it out on request by the examination office of the faculty "Medizintechnik und Technomathematik".

Herzogenrath, 15.12.2022

Location, Date

Leonard Esser
Digitally signed by Leonard Esser
Date: 2022.12.15 15:30:52 +01'00'

Leonard Esser

This thesis was supervised by

Prof. Dr. Philipp Rohde

Dr. Thorsten Lohmar

Aachen, December 2022

## Abstract

The fifth generation of mobile networks is rapidly gaining popularity and usage across most sections of businesses. With the number of use cases rising, the necessity of a way to control and reliably measure data flow is getting progressively more significant.

This thesis aims to address this need by describing and evaluating the construction and design of an environment for testing multiple User Equipments (UE). The idea is to build a program that can later easily be used for testing and analyzing Key-Performance-Indicators (KPI) in a given network. This thesis will provide an overview of what system has to be used for such an application and which structures and tools offer the best solutions. With the tool developed in this thesis, proceeding measurements of KPIs, measured in a 5G System (5GS), for example, will be quick and easy to set up and modify.

The tool will make it possible to start all UEs that are used for measuring the KPIs at once, which is necessary for a valuable testing experience. Additionally, the tool allows for different amounts of data sent by each UE.

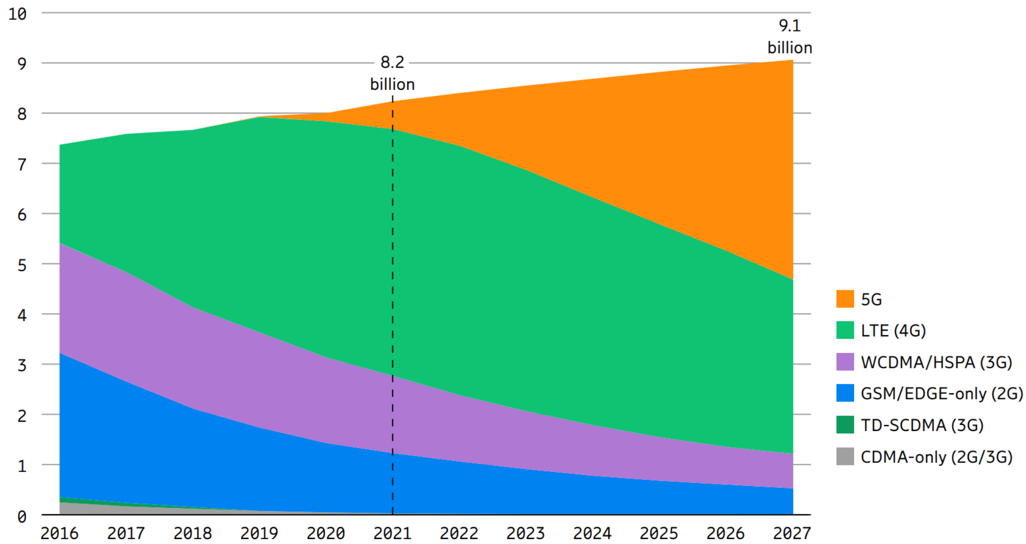
Overall, this thesis will provide a comprehensive overview of the construction and design of an environment for testing multiple UEs in a network. It will also be a valuable tool for any businesses and organizations looking to optimize and improve their data flow in a network system.

# Contents

# 1 Introduction

In one quarter of 2022 alone, the number of 5G subscriptions grew from 550 to around 620 million, and by the end of 2027, the 5G technology is expected to reach more than four billion subscribers. [1]

Figure 1.1: The usage of 5G in comparison to older standards over time [1]




There are many differences between the former 4G and LTE technology and the new modern 5G standard, such as speed and the number of possible connections that can be made to one radio station. However, one of the essential improvements is how 5G Systems (5GS) allow for exposure to enterprise customers. This means that a company can make its products and services available to external parties via application programming interfaces (API). The companies accessing these APIs are known as enterprise customers. Furthermore, the usage of 5GS by vertical industries as a private network is an added feature the 5G technology provides.

This means that the organization using the 5G technology creates a network exclusively for its own use, allowing for many benefits like increased security and faster speeds. [2][3]

With 5GSs gaining more and more popularity, testing already implemented networks is a task that rapidly gains importance. The application discussed in this thesis will help to provide the tools and functionality necessary to configure test setups for these kinds of networks.

Because of the functionality of the tool and for more accessible reliability in further reading, the application written for this thesis will be called Distributed Measurement Control System (DMCS) for further reference.

This thesis starts with a detailed description of Quality of Service (QoS) and how to measure it reliably, as well as a more thorough look into 5GS. It proceeds by describing the concept of work for DMCS and some potential test cases. Chapter 4 continues by providing more information about the implementation and challenges faced when developing the code. Finally, chapter 5 summarizes the findings and suggests ideas for further research.

# 2 Background

## 2.1 Literature Review

This section will provide an overview of existing solutions for measuring and monitoring network characteristics and how the program developed for this thesis differs from existing ones. There are several core differences between existing solutions for measuring these network characteristics within QoS. The DMCS will use existing network testing solutions to send back results as specific parameters. The program will stand at the beginning of the testing process as it configures and sets up the devices, which will then be used for testing. The programs that run on these devices are programs like Iperf3, which can determine and measure several QoS indicators. However, DMCS does not aim to test a system for specific Key-Performance-Indicator (KPI)s but instead configures devices where programs like Iperf3 can run without needing to set up each device individually. Ultimately, this enables the assessment of the impact on traffic to determine whether changes, such as prioritizing one traffic flow over the other, resulting in improved outcomes.

Currently, the way to test multiple devices would be to manually connect to each one individually and go into the terminal to set them up as either a client or a server. This method takes longer and requires a keyboard and monitor ready for each device. The reduction of this time and the required hardware is what DMCS aims for.

## 2.2 Quality of Service and Existing Solutions

Quality of Service is a set of technologies that allows for measuring and managing network traffic or, as in this example, 5G System reliably. These technologies also allow the detection of changing network conditions and prioritizing or holding back specific network traffic. With this type of control, it is possible to prioritize the incoming traffic of a latency-sensitive application, such as live streaming immense amounts of data over 5G. The main focus is the balance between cost and effectiveness. [4]. There are a number of different QoS solutions that can be used to ensure that network traffic is delivered in a timely and reliable manner. Some examples of QoS solutions that are commonly used in 3GPP networks include [5]:

- **Admission Control**: This is a mechanism used to control the amount of traffic allowed to enter the network based on the current network load and the available resources. Admission control aims to ensure that the network has adequate resources to handle incoming traffic without overburdening the network and causing performance degradation. Admission control is typically implemented by establishing thresholds for various network resources, such as bandwidth, processing power, and storage capacity. When a new traffic flow attempts to enter the network, the admission control system assesses the availability of network resources to accommodate the flow. The traffic flow is admitted into the network if sufficient resources are available. Otherwise, the traffic flow may be rejected or given a lower priority than other traffic flows. [6]

- **Traffic Shaping**: This is a method used to regulate the flow of traffic in a network to prevent congestion and ensure that the network operates efficiently. The goal of traffic shaping is to ensure that the network has sufficient capacity to handle the incoming traffic without overloading the network and causing performance degradation. Traffic shaping is typically implemented by applying specific rules or policies to the traffic entering the network. For example, traffic shaping might involve prioritizing certain types of traffic over others so that more essential or time-sensitive traffic is delivered ahead of less important traffic. It might also involve limiting the amount of traffic allowed to enter the network at any given time to prevent congestion.

There are several ways in which traffic shaping can be implemented, depending on the specific requirements and goals of the network. For example, traffic shaping might be implemented at the network edge by applying rules and policies to incoming traffic before it enters the network. Alternatively, traffic shaping might be implemented at intermediate points within the network to manage traffic flow between different network segments. [7][8]

- **QoS Classification and Marking**: This describes a method used in computer networks to identify different types of network traffic and assign a corresponding priority level to each type of traffic based on factors such as priority, type of service, and source and destination. This allows network administrators to ensure that urgent traffic is given priority over less critical traffic, such as web browsing. It is used to prevent congestion in the network and ensure that all users have a consistent and reliable experience. [9]

## 2.3 Measuring Quality of Service Parameters

There are quite a few different parameters that are suited for QoS measuring[10], including

- Packet loss
- Network throughput
- Bandwidth
- Latency
- Retransmissions
- Jitter

### 2.3.1 The Meaning of Quality of Service Parameters

*PacketLoss* is a measure of the percentage of packets not successfully delivered over a network. Packets are small units of data that are transmitted over a network, and packet loss can have several causes, including network congestion, interference, or hardware failures. High packet loss can be noticed very quickly as it slows down the performance of all network applications. With increasing packet loss, the difficulty of diagnosing and troubleshooting network problems is also increased as packet loss can be caused by various factors, and thus be difficult to pinpoint the root cause of the problem. [11]

*Network throughput* is a measure of the amount of data that can be transferred over a network in a given amount of time. It is typically expressed in bits per second or bytes per second and is used to assess the performance and efficiency of a network connection. Several factors, including the speed of the network connection, the amount of traffic on the network, and the type and size of the transferred data influence throughput. A high throughput indicates that a network connection can quickly transfer a large amount of data, whereas a low throughput indicates that the connection is slow or congested. Measuring throughput can be useful for identifying bottlenecks or problems with a network and assessing the connection's overall quality.

*Bandwidth* is a measure of the maximum amount of data that can be transferred over a network in a given time. It is usually measured in bits per second or bytes per second and is used to assess the capacity of a network connection. Bandwidth is determined by the speed of the network connection and the type of technology used. Measuring bandwidth can be useful for determining the maximum capacity of a network connection and comparing the capabilities of different network technologies to choose the best technology for a given application.
In a nutshell, the bandwidth is just the maximum network throughout and can be described as throughput $\leq$ bandwidth. [12]

*Latency* is a measure of the time it takes for a packet to travel from the sender to the receiver over a network and is typically measured in milliseconds. Many factors, including the distance between the sender and receiver, the speed of the network connection, and the amount of traffic on the network, influence latency.
High latency can negatively impact a network's performance as it can cause delays in data transmission and affect the responsiveness of applications or services that rely on the network. For example, a video conferencing system's high latency can cause audio or video delays, making the conversation difficult or impossible to follow. In general, low latency is desirable as it indicates that packets can be transmitted quickly and efficiently over the network. [13]

*Retransmissions* are packets transmitted over a network repeatedly due to errors or losses. They are typically used to ensure the reliability of a network transmission as they can help to recover lost or corrupted packets and ensure that all of the data is delivered successfully. Retransmissions can negatively impact a network's performance as they consume additional bandwidth and can increase the connection's latency. In general, a low rate of retransmissions is desirable as it indicates that the network is able to deliver packets successfully. [14]

Finally, *jitter* describes the divagation from expected or actual periodicity and is usually undesired. Jitter is generally measured in milliseconds by calculating the average time difference between each packet sequence. Several factors, including network congestion, interference, or the distance between the sender and receiver, can cause jitter.

High jitter can have a negative impact on the performance of a network as it can cause delays and fluctuations in the transmission of data. This can affect the quality and reliability of the transmission and can cause problems for applications or services that rely on the network. For example, high jitter in a Voice-over IP (VoIP) system can cause disruptions or interruptions in the audio, which can make the conversation difficult to understand. [15]

### 2.3.2 Measurements of KPIs done by Iperf3

In chapter 4 regarding the implementation of DMCS, a tool called Iperf3 is frequently mentioned. It is used to measure network performance. Iperf3 works by running on two devices at the same time, one working as the server and one as the client. The client then sends data to the server, and the server measures how much data it receives and how quickly it does so. This measurement is then reported back to the client. Additionally, the amount of data sent can be adjusted, as well as the length of the test and the type of data sent. Types of data that are usually sent using Iperf3 include Transmission Control Protocol (TCP) and User Datagram Protocol (UDP) traffic. [16]

The primary distinction between the TCP and the UDP lies in their approaches to acknowledgments and retransmissions. TCP employs a flow control and error-checking mechanism, which confers a high degree of reliability. In contrast, UDP transmits data more rapidly and efficiently than TCP, but at the expense of reliability. [17]

## 2.4 5G Overview

A 5GS can be distinguished between an Access Network and a Core Network. [18] The Access Network connects a UE with the Core Network. There are a few critical differences between 5GS and the Evolved Packet System (EPS), which is used in 4G LTE systems.

- With the new 5GS there are new possibilities to separate the User Plane (UP) from the Control Plane (CP) functions. The old 3GPP could already do that in theory, but because it still had to work with the older versions and be backward compatible, it could not be extended to its full potential. [19]
- Additionally, a 5GS is also capable of providing more speed with LTE having a maximum of 1.000 Mega-Bits per second in download. In contrast, a 5G System can reach up to 10.000 MBits per second. However, the maximum number of UEs that

can be connected to one 5G radio station simultaneously is much more important for denser populated areas. This number rose from 100.000 connectable UEs per square kilometer to one million UEs. [20]

- A 5GS also has a much more overall modular structure. Running on servers or public clouds 5GS allows for easier scaling. Because of this decentralized structure, there is also less need to buy telecommunications hardware. [21]
- The most important difference for network performance testing environments, however, is how 5G mobile systems allow for network slicing and prioritization. This allows network operators to create multiple virtual networks on a single physical network infrastructure. This enables the possibility to provide different levels of QoS to different customers or applications and tailor the network to the specific needs of each use case. On a large scale, this could allow a 5G network to prioritize critical, important applications, such as communication needed for public safety, like police or ambulance communication, over less time-sensitive traffic, such as streaming video or social media browsing. Additionally, network slicing allows the network to allocate resources dynamically according to changing network conditions and user demands. Network slicing relies on several key technologies, such as Network Function Virtualization (NFV), to achieve this. NFV allows operators to virtualize network functions and move them from dedicated hardware to software-based platforms. This makes it possible to create multiple virtual network functions (VNFs) that can be deployed and managed independently, enabling operators to create virtual networks with different characteristics and capabilities. [22]

# 3 Concept

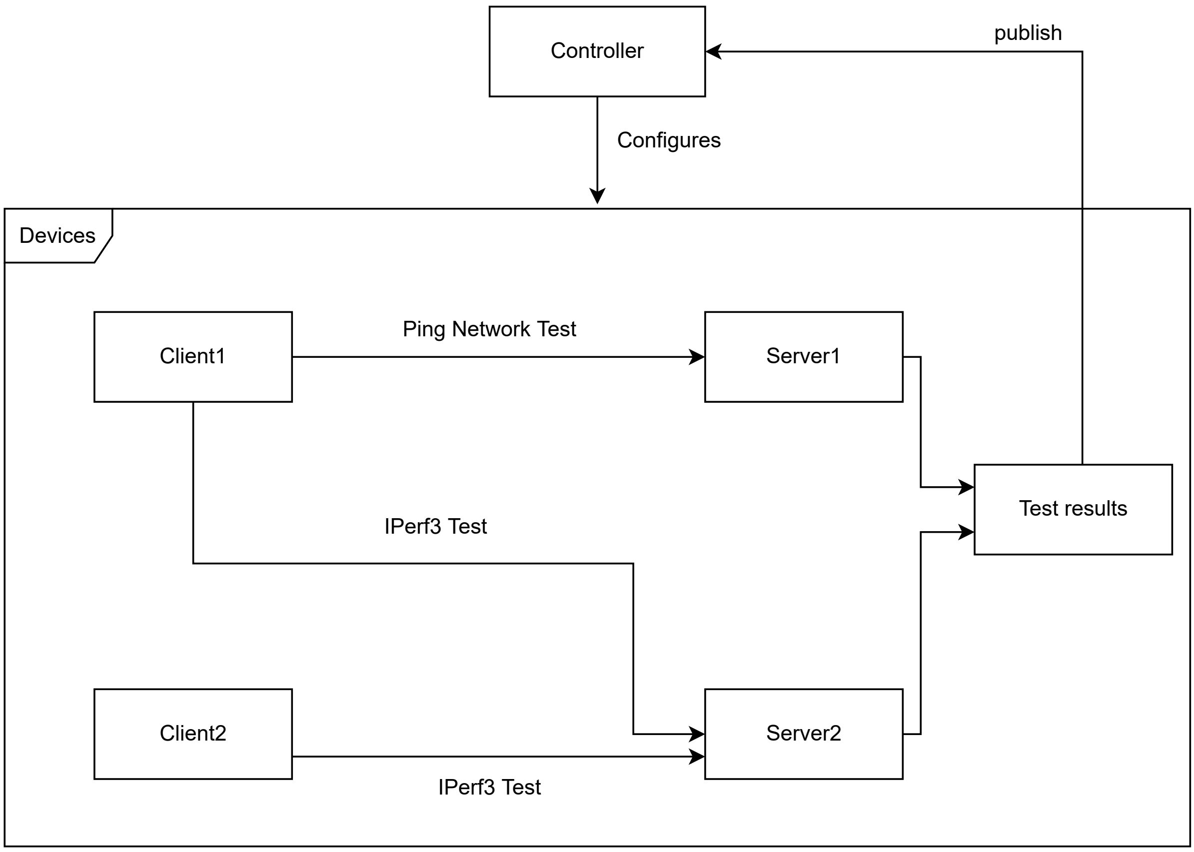
## 3.1 Introduction

This thesis aims to design and test an environment that can be used for systematic performance testing of a given network, precisely controlling multiple servers and clients and allowing the configuration and execution of different test scenarios. DMCS provides a way for a remote control device to connect over a 5GS to any number of supporting devices, configuring them to work as either servers or clients for various kinds of network testing. In the beginning, the program's user decides what network tests should be made and how many devices should be used for these tests. Once configured, the DMCS will automatically run network tests on the specified number of devices using the previously specified instructions. After these are finished, the results will be sent back to the user's remote device. What exact parameters will be tested and sent back will be described later in this thesis.

## 3.2 Conceptual Framework

This section will provide an overview of the key concepts and ideas that underlie DMCS. More detailed information will be given in chapter 4.

Figure 3.1: The setup considered within this thesis




The visualization above shows a simplified version of a possible test setup. The controller remotely configures the two clients and two servers to do either a network test using ping or Iperf3. In this example, client one would connect to both servers and do both tests on one server each. Client two would only connect to server two and execute an Iperf3 test. After the tests are finished, the servers gather the test results and publish them for the controller to display.

The user input that is expected by the program is the number of devices that should be set up as any specific server or client, like Iperf3 in the example above. As will be explained later in chapter 4, DMCS uses a publisher-subscriber-based way of communication. To avoid further confusion or lack of clarity, the roles of the devices in the system will be explained together with a name that will be used for the upcoming parts of the thesis. One device acts as remote access to all other devices. Additionally, it should be able to assign specific configurations to specific devices. This device will be referred to as the controller. The devices handled by the controller will be called either servers or clients, depending on their role. As seen later in this thesis, calling them subscribers or publishers could lead to confusion as each can take on either role.

## 3.3 KPI Thresholds for Maintaining Network Efficiency

The application developed in this thesis exists as an environment to later measure, analyze and evaluate the parameters mentioned above. Knowing what values can be expected and what is commonly accepted as the minimum standard for network or 5GS is therefore essential. Starting with packet loss, the general levels of quality are the following: "0-1% of packet loss is good, 1%-2.5% is acceptable, 2.5%-5% is poor, 5%-12% is very poor and greater than 12% is bad. Our observations show that above 4-6% packet loss video conferencing becomes irritating, and non native language speakers become unable to communicate. Above 10-12% packet loss there is an unacceptable level of back to back loss of packets and extremely long timeouts, connections start to get broken, and video conferencing is unusable." [23]

The bit rate is harder to evaluate because it depends on what you want to measure. For streaming videos, for example, you need a much higher bit rate, than for sending small files and documents from one UE to another.

The throughput also depends on the parameters measured but can be expressed by a ratio of the maximum possible and the actual throughput. For example, if a 50 Mbits/s throughput

is achieved in a 100 Mbits/s ethernet connection, the efficiency is 50%.
The transmission delay is predetermined by the physical layer and the amount of data that can be transferred in this manner. Since the measurements will be conducted in a 5G surrounding, the actual delay should be negligible. This is because the transmission procedure is hugely optimized in the fifteenth version of 3rd Generation Partnership Program (3GPP) by, among a lot of other things, introducing configured grant transmission. [24]
A common aim for availability for most organizations and users is the so-called ”five nines,” referring to 99.999% availability. This standard is set to prevent potential revenue losses due to network unavailability.
Jitter values should not surpass 30 milliseconds because performance issues will be highly noticeable to a user.

## 3.4 Publisher Subscriber Pattern

To command multiple UEs or servers at the same time, to give instructions on what data to send and when to do so, you need to connect to them simultaneously. The best approach to reach this kind of behavior is an observer pattern.

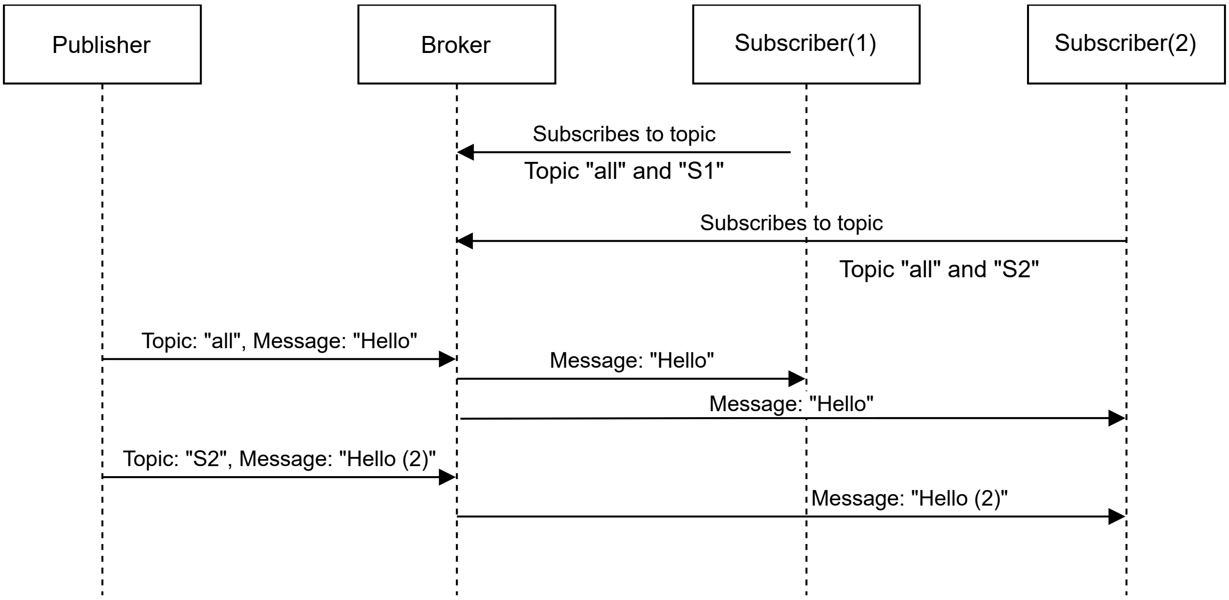



Figure 3.2: The structure of an Message Queueing Telemetry Transport (MQTT) subscription pattern

As visible above, in a system like this, there is usually one publisher, one broker, and any number of subscribers. Every subscriber is subscribed to one or more topics, which are handled as strings. The publisher then publishes a message containing a topic string and a message string. This message is sent to the broker, which is a distributed server. The broker checks how many (if any) subscribers are subscribed to the current topic of the message. In the case of just one subscriber who needs to receive the message, the broker forwards the message. If no subscriber is interested in the current topic, the message sent by the publisher is discarded.

If more than one subscriber listens to the topic specified in the current message, however, the broker multiplies the message and sends it forward to every subscriber simultaneously.

## 3.5 Test Scenarios

To test if the DMCS implementation behaves as expected, some test cases could be implemented, which are listed in the Table below.

| **Test case** | **Description** |
|---|---|
| Connectivity and configuration | Configures two devices, for one to work as an Iperf3 server and one as an Iperf3 client. Then sends simple messages, in the form of strings, by the publisher to the topic everyone is subscribed to. Checks whether these messages arrived as expected and were not changed in any way. |
| Iperf3 and ping functionality | Creates a similar setup to before, where one device acts as the server while the other acts as a client. This time set them up as either ping or Iperf3 specific clients and servers. Check if any errors occur. |
| Publish results | Creates a setup with one server and one client, and run one of the tests. Checks if the results of these tests are published to the controller device as expected. |
| Multiple tests | Creates a device setup, where multiple ping or Iperf3 tests are run simultaneously. Checks if the tool can handle the load and display the results correctly. |

# 4 Implementation

## 4.1 Structure of the publisher-subscriber pattern

In this thesis, the code written for the DMCS is made in python. A built-in python library that makes such a structure possible is the Message Queueing Telemetry Transport (MQTT) library. The conceptual design of such a pattern was already shown in figure 3.2

## 4.2 Python implementation

The problem with this way of handling many clients or UEs at once is that the message sent by the publisher containing a particular topic can not be changed individually for each subscriber. There are a few ways to handle this problem. For this program, however, it was also a requirement that all UEs or servers are set up by just one command to reduce the work that has to be done by hand, like connecting a display and a keyboard to every single server and booting up a file. Instead, we use a config file, which can be started on all servers remotely and will create a file with unique properties for each server or UE.

To do so, each subscriber needs a personal unique ID that is persistent, even after rebooting the program. A simple solution is to use the Media Access Control (MAC) address of the device, as it is unique and stays the same even after rebooting. MAC addresses are unique identifiers assigned to a *network interface controller* to target specific devices within a network segment. The MAC addresses of all the devices are known beforehand and are therefore stored in an array. To set up the clients, first, the user is asked to choose the number of clients to set up for Iperf3 and ping measurements individually. Then a simple check is made to ensure the number of devices chosen for one specific task is possible as no more than eight devices can be configured at a time, and Iperf3 needs at least two devices for one to work as the server and one as the client.

Additionally, each device's MAC address is also stored in a hash map, with the value being the kind of tool the corresponding address is configured to. This is done so the correct code can be sent again in case of a failed connection or another error on the client's site. The hash map has a structure like this:

```
hash_map[MAC_addresses[i]] = cmd_for_iperf_client
```

, where i is the ith device set up in this loop and *cmd_for_iperf_client* is the code that gets sent to the device, that will act as the Iperf3 client. It tells the client to run an  test and what IP to ping.

If the program throws an error on the client's site, it will be caught and send a message as a publisher to the topic "error", to which the remote controller that configures the devices is subscribed itself. The remote controller checks for these error messages, and if it receives one, it resends the code using the hash map described above. The structure looks like this:

```
def on_message(client, userdata, msg):
# Look for errors and resend the code to the client
if msg.topic == "Error":
    id_of_sender, error = msg.payload.decode("utf-8").split(";")
    print("Error: " + error + " for id " + id_of_sender)
    client.publish("Code", id_of_sender + "," + hash_map[id_of_sender])
```

As mentioned above, the id_of_sender is the MAC in this case. As the goal was to have as few parameters required to be known as possible, the IP address of any device needed for ping and Iperf3 measurements had to be sent to the controller as well. This was done by extending the "on_connection" function of the MQTT library to publish a message containing the IP address and the corresponding MAC address to the controller. The extended function looks like this:

```python
def on_connect(client, userdata, flags, rc):
    if rc == 0:
        print("connected OK Returned code=", rc)
    else:
        print("Bad connection Returned code=", rc)
        # publish as an error to the contoller
        client.publish("error", s_id + ";Client" + str(s_id) + " could not connect to broker")
    # find the ip address of the current device and publish it
    ip_address = socket.gethostbyname(socket.gethostname())
    client.publish("ip_config", s_id + ";" + ip_address)
```

This IP address is then again stored by the controller in a hash-map that maps the MAC address as a key to the IP address of one device, acting either as a server or a client. The implementation is done the following way:

```python
if msg.topic == "ip_config":
    id_of_sender, ip = msg.payload.decode("utf-8").split(";")
    Mac_ip_map[id_of_sender] = ip
    print("Received ip " + ip + " for id " + id_of_sender)
```

Saving results in order to send them back to the controller so they can be analyzed and evaluated can be done in a number of ways. As an example, a straightforward way to do it would be to save all the results that are usually printed to the console by programs like Iperf3 and just send the entire file to the controller. In this thesis, however, to save storage and as a result of the tests not being at large scale, the sending back was done by stripping the results to the information needed and sending it back as a list containing three rows. The code to do that is shown here:

```python
output = subprocess.check_output(msg_content, shell=True).decode("utf-8")
    lines = output.split("\n")
    results = lines[-6:-3] # filters results for the significant three lines only
    payload = json.dumps(results) # Serializes list into string
```

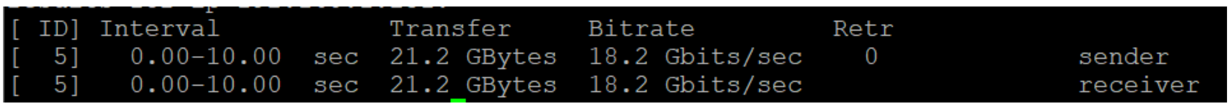
```python
    client.publish("iperf", s_id + ";" + payload)
```

The list *results* can not be published directly because strings can only be sent using the publish method. To avoid using other types of sending data, the list is, therefore, first serialized using *json.dumps()*. As mentioned above, the code will be published to the topic *iperf*, and the device's ID will be sent along with it so that the controller can easily distinguish between all the results it receives. To be able to work with this data as a list again, the controller then needs to deserialize that data again using JSON like this:

```python
results = json.loads(message.payload)
```

An example of how the results of such a test could look is shown below.

Figure 4.1: Sample test results of an Iperf3 test




The row labeled *Retr* in the illustration above represents retransmissions.

# 5 Conclusion and outlook

This thesis provided an overview of why an application for remote configuration for network performance testing is essential with networks, especially 5GS, gaining increasingly in popularity and density of use. It described the background of all the necessary KPIs that need to be evaluated and can be tested using such a setup. Also, it focused on the explanation and understanding of Iperf3 testing and how its features are used in the application.

Currently, the most efficient way to create an environment the way this application sets it up is to connect to each device separately, either remotely or locally. This Distributed Measurement Control System (DMCS) helps rapidly speed up that process and allows for reliable and fast testing environments.

With the working tool, not only existing and working networks can be tested for the parameters of Quality of Service, but the amount of data sent through a single UE can be changed as well, which allows further analysis in a network.

Several topics can be discussed in further research, like the actual measurement of KPIs and determining if they are above the mentioned thresholds, addressed in section 3.3. Additionally, more options for sending traffic and measuring traffic flow could be implemented, like using GStreamer, which allows for the compression of video files to transmit real video data instead of the fabricated data sent by tools like Iperf3.

# Glossary

**bandwidth** is a measure of the maximum amount of data that can be transferred over a network in a given amount of time.

**controller** is the device that remotely accesses other devices and configures them to be either clients or servers.

**Iperf3** is a newer version of iperf, which allows for network performance measurements via UDP or TCP.

**jitter** is a measure of the variation in the delay of packet arrivals over a network.

**latency** is a measure of the time it takes for a packet to travel from the sender to the receiver over a network.

**network interface controller** is a hardware component that connects a computer to a computer network.

**packet loss** is the percentage of packets that are not successfully delivered over a network.

**retransmissions** are packets that are transmitted over a network again due to errors or losses.

**throughput** is a measure of the amount of data that can be transferred over a network in a given amount of time.

# Acronyms

**3GPP** 3rd Generation Partnership Program.

**5GS** 5G System.

**API** application programming interface.

**CP** Control Plane.

**DMCS** Distributed Measurement Control System.

**EPS** Evolved Packet System.

**KPI** Key-Performance-Indicator.

**MAC** Media Access Control.

**MQTT** Message Queueing Telemetry Transport.

**NFV** Network Function Virtualization.

**PCAP** Packet capture.

**QoS** Quality of Service.

**RAN** Radio Access Network.

**RTP** Real-Time Transport Protocol.

**TCP** Transmission Control Protocol.

**UDP** User Datagram Protocol.

**UE** User Equipment.

**UP** User Plane.

**VoIP** Voice-over IP.

# A Bibliography


[1] Ericsson. "Mobile subscriptions outlook". In: (June 2022). URL: https://www.ericsson.com/en/reports-and-papers/mobility-report/dataforecasts/mobile-subscriptions-outlook. (accessed: 28.11.2022).

[2] Menglan Jiang, Massimo Condoluci, and Toktam Mahmoodi. "Network slicing management & prioritization in 5G mobile systems". In: *European wireless 2016; 22th european wireless conference*. VDE. 2016, pp. 1–6.

[3] *Enable innovation with open network exposure*. 2022. URL: https://www.ericsson.com/en/service-orchestration/network-exposure. (visited 11.12.2022).

[4] Archiveddocs. *Quality of Service (QoS) Overview*. Aug. 2016. URL: https://learn.microsoft.com/en-us/previous-versions/windows/it-pro/windows-server-2012-R2-and-2012/hh831679(v=ws.11)?redirectedfrom=MSDN. (accessed: 01.12.2022).

[5] Marko Luoma. "QoS and queuing disciplines, traffic shaping and admission control." In: *Seminar at Laboratory of Telecommunications Technology*. 1999.

[6] E. Mykoniati et al. "Admission control for providing QoS in DiffServ IP networks: the TEQUILA approach". In: *IEEE Communications Magazine* 41.1 (2003), pp. 38–44. DOI: 10.1109/MCOM.2003.1166652.

[7] Massimiliano Marcon et al. "The local and global effects of traffic shaping in the internet". In: *2011 Third International Conference on Communication Systems and Networks (COMSNETS 2011)*. 2011, pp. 1–10. DOI: 10.1109/COMSNETS.2011.5716420.

[8] *Traffic Shaping Definition*. Oct. 2022. URL: https://avinetworks.com/glossary/traffic-shaping/. (accessed: 13.12.2022).

[9] Jan. 2022. URL: https://ipcisco.com/lesson/classification-and-marking/. (accessed:13.12.2022).

[10] Jose Javier Garcia Aranda et al. *The Quality for Service (Q4S) Protocol*. RFC 8802. July 2020. DOI: 10.17487/RFC8802. URL: https://www.rfc-editor.org/info/rfc8802. (visited: 08.12.2022).

[11] James F. Kurose and Keith W. Ross. *Computer Networking: A Top-Down Approach*. 5th. Boston, MA: Boston : Addison-Wesley, 2010.

[12] Guowang Miao et al. *Fundamentals of Mobile Data Networks*. Cambridge University Press, 2016. DOI: 10.1017/CBO9781316534298. (accessed: 08.12.2022).

[13] IR Team. *Network Latency - Common Causes and Best Solutions — IR*. 2022. URL: https://www.ir.com/guides/what-is-network-latency. (accessed: 11.12.2022).

[14] *Packet Retransmissions*. June 2019. URL: https://www.haivision.com/resources/streaming-video-definitions/packet-retransmission/. (accessed:11.12.2022).

[15] Wikipedia Contributors. *Jitter*. Oct. 2022. URL: https://en.wikipedia.org/wiki/Jitter. (accessed: 11.12.2022).

[16] Vivien GUEANT. *iPerf - iPerf3 and iPerf2 user documentation*. 2013. URL: https://iperf.fr/iperf-doc.php. (visited on 12.12.2022).

[17] Laura Klusiaté. Apr. 2021. URL: https://nordvpn.com/de/blog/tcp-vs-udp/. (accessed: 14.12.2022).

[18] *What Is 5G Network Architecture?* 2021. URL: https://www.digi.com/blog/post/5g-network-architecture. (accessed: 13.12.2022).

[19] *Control and User Plane Separation (CUPS) for Next Generation EPC*. Apr. 2019. URL: https://www.rebaca.com/insights-blogs/abot-test-orchestration-solution/control-and-user-plane-separation-for-next-generation-epc. (accessed: 30.11.2022).

[20] *Samsung's 6G White Paper Lays Out the Company's Vision for the Next Generation of Communications Technology*. 2020. URL: https://news.samsung.com/global/samsungs-6g-white-paper-lays-out-the-companys-vision-for-the-next-generation-of-communications-technology. (accessed: 11.12.2022).

[21] Mahin K. Atiq et al. "When IEEE 802.11 and 5G Meet Time-Sensitive Networking". In: *IEEE Open Journal of the Industrial Electronics Society* 3.21525853 (), pp. 14–36. DOI: 10.1109/OJIES.2021.3135524.

[22] Bo Han et al. *Network Functions Virtualization: Challenges and Opportunities for Innovations*. URL: https://www-users.cselabs.umn.edu/classes/Fall-2017/csci8211/Papers/NFV%5C%20Challenges-n-Opportunities.pdf. (accessed: 11.12.2022).

[23] ICTP Science Dissemination Unit. *ICTP-SDU: about PingER*. 2019. URL: https://web.archive.org/web/20131010010244/http://sdu.ictp.it/pinger/pinger.html. (archived from the original on 27.11.2022. Retrieved 16.05.2013.)

[24] Jing Xu et al. "Chapter 15 - 5G Ultra-reliable and low-latency communication: PHY layer". In: *5G NR and Enhancements*. Ed. by Jia Shen et al. Elsevier, 2022, pp. 687–739. ISBN: 978-0-323-91060-6. DOI: https://doi.org/10.1016/B978-0-323-91060-6.00015-5. URL: https://www.sciencedirect.com/science/article/pii/B9780323910606000155. (accessed: 05.12.2022).