Master's thesis

NTNU
Norwegian University of Science and Technology
Faculty of Information Technology and Electrical Engineering
Department of Information Security and Communication Technology

Mathias Kjølleberg Førland

SDN - a crucial security component towards 5G

Master's thesis in Kommunikasjonsteknologi
Supervisor: Katina Kralevska
June 2019

NTNU
Norwegian University of
Science and Technology

Mathias Kjølleberg Førland

SDN - a crucial security component towards 5G

Master's thesis in Kommunikasjonsteknologi
Supervisor: Katina Kralevska
June 2019

Norwegian University of Science and Technology
Faculty of Information Technology and Electrical Engineering
Department of Information Security and Communication Technology

NTNU
Norwegian University of
Science and Technology

Title: SDN - a crucial security component towards 5G

Student: Mathias Kjølleberg Førland

Problem description:

The interest and threats of mobile network security are growing in our increasingly digitized world. Major values, critical infrastructures, sensitive information, and a huge number of devices are increasingly connected to mobile communication systems. This attracts malicious stakeholders to exploit the vulnerabilities of the mobile communication systems for their own gain.

An attack that has noticeably increased frequency in recent years and is, therefore, an increasing threat is the Denial-of-Service (DoS) attack. It can exhaust network resources and make services unavailable. It is also hard to prevent as it is challenging to distinguish between increased traffic and malicious flooding.

Next generation of mobile communication systems 5G is also exposed to the same threat. The complex network architecture and the introduction of novel technologies in 5G have paved paths for the introduction of new variants of the DoS attack to the mobile communications system.

In this thesis, I will focus on one of the technologies to be used in 5G, Software Defined Networking (SDN) and explore if some of its attributes can be used as a security mechanism in the network. My goal is to create a proof of concept (POC) demonstration of SDN that detects and prevents DoS attacks through simulation.

I used last semester to research 5G and the technology's vulnerabilities, as well as potential solutions. For my master project, I will implement an SDN controller in the simulation program OMNet ++. The controller uses its overview of the network and its centralized control to detect DoS variants called "SYN flooding attacks" in order to prevent those attacks by dropping the packets. The simulated network environment will be a simplified version of a small 5G network.

Responsible professor: Danilo Gligoroski, IIK

Supervisor: Katina Kralevska, IIK

Abstract

The fifth generation (5G) of mobile communications systems is expected to be commercially launched in 2020 and anticipated as a disruptive technology that enables among other things critical infrastructure use cases like autonomous vehicles and smart cities. Software-Defined Networking (SDN) is one of the key technologies of 5G as it introduces the concept of separation of data and control to the mobile communication system, adding centralized control, flexibility and simplicity to the network architecture. Some crucial security issues have been identified for 5G, including the increased danger of Distributed Denial-of-Service (DDoS) attacks as the number of connected devices is expected to grow 10 to 100 times with the new mobile networking technology. A proposed solution is to utilize SDNs overview of the network, programmability and centralized control for early detection and swift mitigation of network attacks.

In this thesis work, we present an investigation of the security capabilities of SDN in a close-to 5G scenario, using the OMNeT++ Discrete Event Simulator and novel integration of the extension libraries, INET, simuLTE, and OpenFlowOMNeTSuite. An SDN DDoS Defense controller application is implemented with a connection rate-based detection method and tested in a close-to 5G environment against a distributed SYN flood attack. The performance of the controller application is evaluated by a sensitivity and specificity analysis and validated through numerous experiments where parameters such as attack rate, detection threshold, flow entry timeout, and the number of malicious and benign nodes are varied.

The results obtained show that the SDN DDoS Defense controller application is can and mitigate SYN flood attacks effectively and provide a Proof of Concept for SDN as a security component. The impact of different network characteristics and controller application configurations are discussed, as well as experiment limitations and the potential of SDN as a security component in 5G. Finally, suggestions for future work are given.

Sammendrag

Den femte generasjonen (5G) av mobilkommunikasjonssystemer forventes å bli kommersielt lansert i 2020 og forutses som en banebrytende teknologi som blant annet muliggjør brukstilfeller for kritisk infrastruktur som autonome kjøretøy og smarte byer. Software-Defined Networking (SDN) vil være en av nøkkelteknologiene i 5G, ettersom det introduserer separasjon av data og kontroll i mobilkommunikasjonssystemet og bidrar til sentralisert kontroll, fleksibilitet og simplifisering av nettverksarkitekturen. Noen kritiske sikkerhetsproblemer er identifisert for 5G, som inkluderer den økte faren for distribuert tjenestenektangrep da antallet tilkoblede enheter forventes å vokse 10 til 100 ganger med den nye mobilnetteteknologien. En foreslått løsning er å bruke SDN sin oversikt over nettverket, programmerbarhet og sentraliserte kontroll til tidlig deteksjon og rask mitigering av nettverksangrep.

I denne masteroppgaven presenteres en praktisk analyse av sikkerhetsegenskapene til SDN i et nært 5G-scenario ved hjelp av OMNeT ++ Discrete Event Simulator og en ny integrering av utvidelsesbibliotekene INET, simuLTE og OpenFlowOMNeTSuite. En SDN kontrollerapplikasjon som forsvarer mot tjenestenektangrep er implementert med en deteksjonsmetode basert på tilkoblingshastighet. Applikasjonen er testet i et nært 5G-miljø mot et distribuert SYN-flomangrep. Kontrollerapplikasjonen sin prestasjon er evaluert av en følsomhets- og spesifisitetsanalyse, og validert gjennom mange eksperimenter der parametere som angrepshastighet, deteksjonsgrense, levetiden til flyttabeller og antall ondsinnede og godsinnede noder i nettverket varierer.

Resultatene som er oppnådd viser at kontrollerapplikasjonen er i stand til å oppdage og mitigere SYN-flomangrepene effektivt, og gir derfor en konseptbeviselse for SDN som en sikkerhetskomponent. Effekten forskjellige nettverksmiljøer og konfigurasjonen av kontrollerapplikasjon har på applikasjon sin detekteringsevne diskuteres, samt eksperimentbegrensninger og potensialet til SDN som en sikkerhetskomponent i 5G. Til slutt er forslag til fremtidig arbeid gitt.

Acknowledgments

This work would not be possible without help from the following people, and I want to thank them for their generosity and support personally. Katina Kralevska for her continuous supervision, constant availability, vast knowledge and essential advice that she has shared with me throughout the work. Danilo Gligoroski for his excellent guidance and for sharing his expertise by taking part in discussions and valuable feedback. I am thankful and indebted to Michele Garau for generously assisting me with the integration of simulation frameworks and his continuing support and insightful help. Nicholas Gray at the University of Würzburg for his expertise and help with OpenFlowOMNeTSuite, as well as the OMNeT++ community for sharing their knowledge and experience. I also thank my family and friends for their never-ending support, encouragement and attention.

Contents

List of Figures

List of Tables

List of Acronyms

1G First Generation.

3GPP 3rd Generation Partnership Project.

5G-PPP Fifth Generation Infrastructure Public Private Partnership.

ACK acknowledgement.

APN Access Point Name.

ARP Address Resolution Protocol.

DaaS Detection-as-a-Service.

DDoS Distributed Denial-of-Service.

DNS Domain Name System.

DoS Denial-of-Service.

EC Edge Controller.

eMBB Enhanced Mobile Broadband.

eNB eNode-B.

EPC Evolved Packet Core.

E-UTRAN Evolved-UMTS Terrestrial Radio Access Network.

FN False Negative.

FP False Positive.

GC Global Controller.

GTP GPRS Tunneling Protocol.

HSS Home Subscriber Server.

ICMP Internet Control Message Protocol.

IDS Intrusion Detection System.

IGMP Internet Group Management Protocol.

IMSI International Mobile Subscriber Identity.

IoRL Internet of Radio Light.

IoT Internet of Things.

IP Internet Protocol.

IPS Intrusion Prevention System.

ITU International Telecommuncation Union.

KPI Key Performance Indicator.

LTE-A Long Term Evolution Advanced.

MAC Media Access Control.

MEC Mobile Edge Computing.

MIMO Multiple Input Multiple Output.

MME Mobility Management Entity.

mMTC Massive Machine Type Communication.

MTU Maximum Transmission Unit.

MULTOPS Multi-Level Tree for Online Packet Statistics.

NAS Non-access Stratum.

NED Network Description File.

NFV Network Function Virtualization.

NMT Nordic Mobile Telephone.

NTP Network Time Protocol.

ONF Open Networking Foundation.

OVS Open vSwitch.

PCDP Packet Data Convergence Protocol.

PDN-GW Packet Data Network Gateway.

PDU Protocol Data Unit.

POC Proof of Concept.

pps data packets per second.

QAM Quadrature Amplitude Modulation.

QoS Quality of Service.

RAN Radio Access Network.

RAT Radio Access Technology.

RLC Radio Link Control.

RNC Radio Network Controller.

ROHC Robust Header Compression.

RRC Radio Resource Control.

SDN Software-Defined Networking.

S-GW Serving Gateway.

SMS Short Message Service.

SOM self-organizing map.

SPTN Software-Defined Packet Transport Network.

SYN synchronize.

TA Tracking Area.

TCB Transmission Control Block.

TCP Transmission Control Protocol.

TN True Negative.

TNR True Negative Rate.

TP True Positive.

TPR True Positive Rate.

UDP User Datagram Protocol.

UE User Equipment.

UMTS Universal Mobile Telecommunications System.

URLLC Ultra Reliable Low Latency Communication.

VNF Virtualized Network Function.

Chapter 1
Introduction

1.1 Motivation

New digital devices and gadgets are introduced every year to improve our everyday life, both for private and corporate use. The uprising of Internet of Things (IoT), need for streaming capabilities for UE, autonomous cars and smart cities are examples of use cases that rely on advanced mobile communications systems. As these kinds of technologies evolve and the number of connected devices increases, the requirements for reliable connectivity, mobility, low latency and broadband availability increases as well. This calls for the mobile networking technology to leap into the new era of the fifth generation (5G) of mobile communications systems. A commercial 5G network is expected to be available by 2020 [AKL+18]. The Fifth Generation Infrastructure Public Private Partnership (5G-PPP) have defined six Key Performance Indicators (KPIs) for 5G [5gK]:

- The mobile data volume per geographical area is increased by a factor of 1000.
- The number of connected devices grow 10 to 100 times.
- Energy consumption is decreased by a factor of 10.
- End-to-end latency below 1ms.
- 10 to 100 times higher typical data rate.
- Ubiquitous access to 5G, even sectors with low population density.

To achieve these expectations, 5G will rely on Software-Defined Networking (SDN), Network Function Virtualization (NFV) and Cloud Computing concepts to provide flexibility in network operations and efficient mobile network management. These technology concepts contribute with simplicity to the network architecture, but also have inherent security challenges [AKL+18].

The introduction of novel technologies to the mobile communication system comes with a new set of security challenges, in addition to inheriting many of the old ones. Recent research has revealed a great deal of issues that need to be solved. The authors in [AKL+18] identified eight key security challenges in 5G. One of them discusses the lack of mandated security in the 5G network, as service-driven constraints on the security architecture lead to the optional use of security measures. Two other challenges are about the threat of Denial of service (DoS) attacks, where both the infrastructure and end-user device are vulnerable. The complex network architecture makes the infrastructure vulnerable as several elements of network control are visible and there exist unencrypted control channels. Additionally, end-user devices are vulnerable because there are no security measures for operating systems, applications, and configuration data on them.

Such security issues are important to address as they may jeopardize the entire network if exploited, both in terms of functionality and trust between users and service providers. Some of the identified use cases of 5G, such as smart cities and autonomous cars, relate to the critical infrastructure of nations. If such systems where to be exploited it would cause large amounts of damage both societally, financially and individually. Therefore these security issues must be solved if 5G is to be the successful and disruptive technology as it is expected.

In a security-context, SDN has some useful attributes. It has a global view of the network, centralized control and provides programmability of the network elements. This makes it possible for SDN to monitor and gather network statistics through its southbound API and detect anomalies or malicious patterns in network traffic. It can use its global view of the network to identify the malicious patterns as, for example, a DoS attack and respond to the event by updating flow tables to filter out the traffic or redirect it to an Intrusion Prevention System (IPS) using its centralized control. Early threat detection at any network location and quick response at run-time can become an vital security application in the 5G network.

1.2 Contribution

This thesis investigates the security capabilities of SDN in a close-to 5G environment by implementing an SDN DDoS Defense controller application in a simulation environment that will detect and mitigate malicious flows. The simulation environment is created in OMNeT++ Discrete Event Simulator with a novel integration of two OMNeT++ extension libraries, namely simuLTE and OpenFlowOMNeTSuite. The close-to 5G environment created by the combination of two packages enable vast and diverse testing of 5G-like topologies, as well as performance analysis of SDN security application with various detection and mitigation methods. An experiment has been conducted in the simulation environment that is designed to test the application's

performance against a distributed SYN flood attack performed by compromised UEs. The experiment is repeated with varying parameters of number of attack nodes, number of benign nodes, detection threshold, flow entry timeout and attack rate. The results are quantified and validated using an evaluation method. The end of the thesis combines theory and experimental results to provide a Proof of Concept.

1.3 Outline

The rest of the thesis is structured as followed:

Chapter 2 provides detailed background information about mobile communication technologies that are relevant to the thesis, as well as elaborates on the threats and countermeasures that relate to the mobile communication systems.

Chapter 3 elaborates on the security attributes of SDN and state-of-the-art research related to the use of SDN in a security context.

Chapter 4 describes and explains the OMNeT++ Discrete Event Simulator and extension frameworks used to perform experiments in the thesis.

Chapter 5 gives details on the topology and architecture of the experiment, the implemented SDN application, evaluation metrics as well as presents the results and analysis.

Chapter 6 discusses the results presented in chapter 5 and the limitations of the experiments.

Chapter 7 concludes the thesis and suggests further work.

Chapter 2
Background

2.1 From 4G to 5G

Since the introduction of the first commercial cellular services by Nordic Mobile Telephone (NMT) in 1981, known as First Generation (1G) systems, the wireless mobile telecommunication technology has been under continuously research and development [MK15]. The extensive scrutiny has led to a new generation of mobile technology approximately every ten years, as shown in Figure 2.1. Each generation has introduced new features and higher capabilities than its predecessor, while still providing backward compatibility to a certain extent.

Long Term Evolution (LTE) is the 3rd Generation Partnership Project (3GPP) release 8 and was developed to replace 3G, but did not meet the full 4G requirements set by the International Telecommuncation Union (ITU). This led to the continued development for 4G which was achieved by the 3GPP release 10 called Long Term Evolution Advanced (LTE-A) [lte]. 5G is expected to replace 4G/LTE-A over time, while still inherit some of its components.

2.1.1 4G/LTE-Advanced Architecture

The LTE-A architecture builds upon the Universal Mobile Telecommunications System (UMTS) architecture, both having in common the clear separation of the radio network and core network in their design. In LTE-A the core network is referred to as the Evolved Packet Core (EPC) and the Radio Access Network (RAN) is referred to as the Evolved-UMTS Terrestrial Radio Access Network (E-UTRAN). The LTE-A architecture improves latency and reduces cost compared to UMTS by reducing the number of components in the architecture. An overview of the LTE-A Network with all its nodes is given in Figure 2.2. In the following subsections, the purpose and interactions of each element are described.

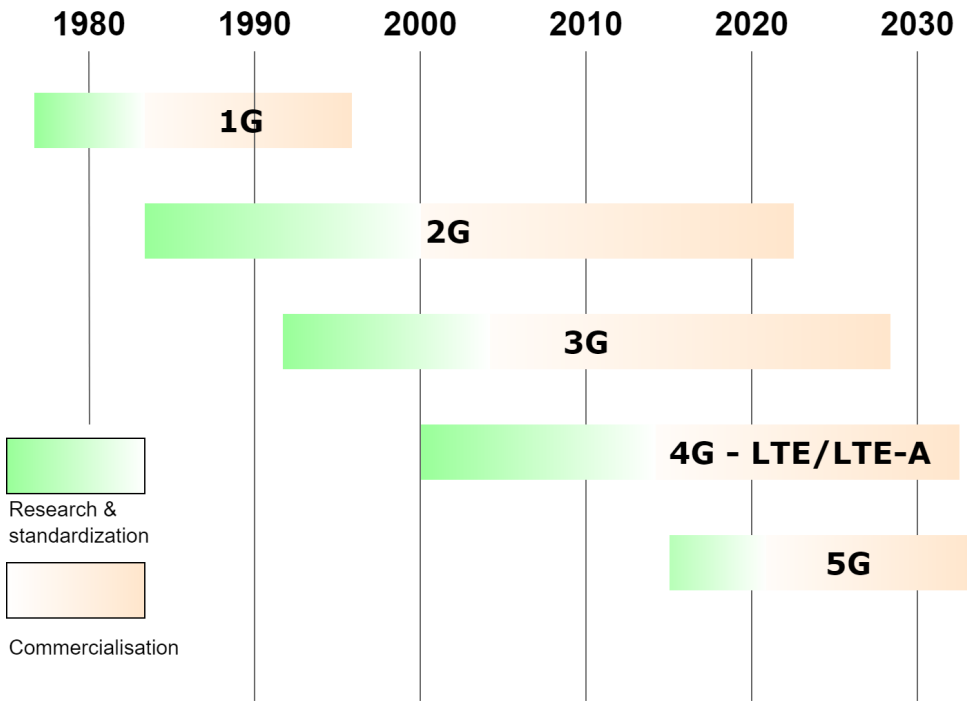


Figure 2.1: The evolution of mobile telecommunication technology [evo].

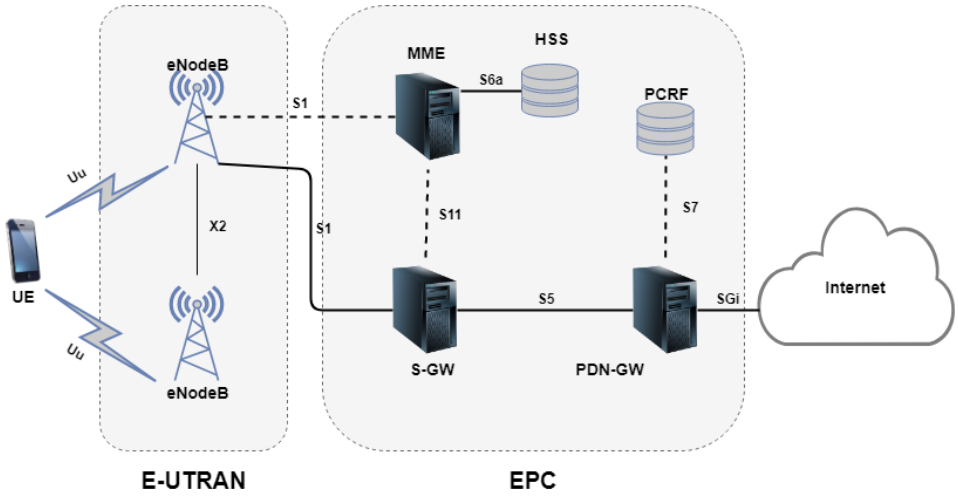


Figure 2.2: LTE-A Architecture [RBB+16]

User Equipment (UE)

As described in [Sau14], the UEs represent all mobile devices that connect to the LTE-A according to the LTE specifications. For a mobile device to be UMTS compatible it needs to support the 64-Quadrature Amplitude Modulation (QAM) in the downlink direction, and at least the 16-QAM in the uplink direction. These modulation schemes facilitate the use of Multiple Input Multiple Output (MIMO) transmissions, which also is a requirement in the downlink direction. MIMO allows the base station to transmit numerous data streams on the same carrier frequency with several antennas to be received by several antennas at the mobile device. The modulation schemes make possible for the receiver to differentiate the different signals transmitted even if they reach it through different paths. The standard number of antennas used in LTE-A networks are two transmitters and two receivers in each device (2x2 MIMO), which allows two data streams to be transmitted simultaneously in the same direction. Most UEs can achieve peak datarates of 100 to 150 Mbit/s using a 20MHz carrier, with the average speed naturally being lower.

eNodeB

The eNode-B (eNB) is an evolved version of the base station Node-B used in UMTS, thereby the name, where 'e' stands for 'evolved'. According to [Sau14], the primary function of the component is to handle the air interface, called the LTE Uu. The component is comprised of three central elements, namely radio antennas, radio modules, and digital modules. Radio antennas enable to communicate by transmitting and receiving signals. The radio modules modulate transmitted signals so the message can be transmitted correctly over the air interface, and to demodulate the received messages to extract the original information that was sent over the air interface. The digital modules are used as an interface to the core network and are called the S1 interface.

The eNB also takes on new responsibilities compared to its predecessor. Functionality, such as mobility management, user management, interference management, air interface resource scheduling, ensuring Quality of Service (QoS), and load balancing, that the Radio Network Controller (RNC) in UMTS was responsible for has now been added to the eNB.

The theoretical peak data rates over the air interface that can be achieved by the eNB depend on the spectrum usage of a cell, but with optimal conditions, 2 x 2 MIMO and 20 MHz carrier it can achieve up to 150 Mbits/s. Bandwidth allocations from 1.25 MHz to 20 MHz are allowed in LTE-A. Real achievable peak data rates are much lower than the theoretical peak data rates as they are affected by variables such as interference, transmission power, and distance between the UE and the eNB.

Mobility Management Entity (MME)

The Mobility Management Entity (MME) is the network node that is responsible for all signalling from base stations and users to the core network, and from the core network to base stations and users. This signalling is called Non-access Stratum (NAS) as it is not apart of the air interface. In [Sau14], they define six main tasks that comprise the MME's functionality.

Authentication: The MME is apart of the authentication process of new subscribers attempting to connect to an LTE-A network. The eNB contacts the MME when a new subscriber arrives and is used in the communication between the MME and the subscriber. After the MME receives authentication information from the subscriber, the MME contacts the Home Subscriber Server (HSS) to validate the given authentication information. If the subscriber is authenticated by the HSS, the MME transmits encryption keys to be used to cipher the following data transfer to the eNB. The HSS will be described in more detail in Section 2.1.1.

Establishment of bearers: The MME assists the eNB to establish Internet Protocol (IP) tunnels to the Internet gateway and is responsible for choosing which gateway router should be used.

NAS Mobility Management: When a UE is inactive for an amount of time in the LTE-A network it goes into a mode where it is free to roam within a Tracking Area (TA) without updating the network of its whereabouts as its air interface connection and resource are released. This decreases the signalling overhead in the network as well as decreases the UE's power consumption. If a packet arrives in the network that is destined for a UE in this mode, it needs to be notified. Therefore the MME will page all eNBs within the TA and make the UE reestablish its air interface connection.

Handover support: The MME supports the handover of users between two eNBs if the X2 interface is not available by working as an intermediate for their message exchange.

Interworking with other radio networks: It is the MME's responsibility to function as the primary facilitator when a UE travels out-of-range of the LTE-A network and should be transferred to another radio network.

Short Message Service (SMS) and voice Support: It is used to make LTE-A supports SMS and voice calls.

Home Subscriber Server (HSS)

As described in [Sau14], the HSS functions is a centralized subscriber database in the home network that uses the IP-protocol DIAMETER to communicate with other

network nodes. Each subscriber record in the database contains the subscribers' International Mobile Subscriber Identity (IMSI), authentication information, the subscriber's telephone number, allowed circuit-switched services (e.g. SMS, voice calls) packet-switched services (e.g. Access Point Names (APNs)), IMS-specific information, and IDs of essential network nodes responsible for the subscriber.

Serving Gateway (S-GW)

The main functionality of the Serving Gateway (S-GW) is to interconnect the radio network and the Packet Data Network Gateway (PDN-GW) by managing the tunnels between them. The PDN-GW is described in greater detail in Section 2.1.1.

Packet Data Network Gateway (PDN-GW)

The main functionality of the PDN-GW is to serve as the gateway to the Internet from the EPC, assign IP addresses to the UEs and serve as a mobility anchor for international roaming by having a GPRS Tunneling Protocol (GTP) tunnel created between the PDN-GW in the home network to the S-GW in the visited network.

2.1.2 5G Architecture

Figure 2.3 illustrates a proposed SDN-based 5G architecture by the authors of [SLLL14], which is used as basis for the network topology in the experiment described later in Section 5.1.1. The architecture consists of a layered Cloud Controller for improved latency and load balancing, where the Edge Controllers (ECs) are responsible for events within a single RAN domain, while the Global Controller (GC) takes care of events generated across domains, such as handover between the WiFi and the LTE network. 5G integrates all existing Radio Access Technologys (RATs) and adds new RATs like mmWave.

Furthermore, Figure 2.3 demonstrates the key enabling technologies - SDN, NFV, and Mobile Edge Computing (MEC). SDN is showcased in the architecture with the separation of the control plane and data plane, where the OF-vSwitch represents the data plane and the Cloud Controller represents the control plane. Cloud Computing is expressed with controllers implemented in the cloud. NFV is represented by the use of off-the-shelf commodity hardware where its functionality is implemented as software in a virtualized environment. The three key enabling technologies are explained in more detail in Section 2.2 below.

2.2 5G Key Technologies

In order to facilitate different use cases on the same infrastructure, efficient use of the resources is essential. Key technologies that enable that are SDN, NFV, and MEC.

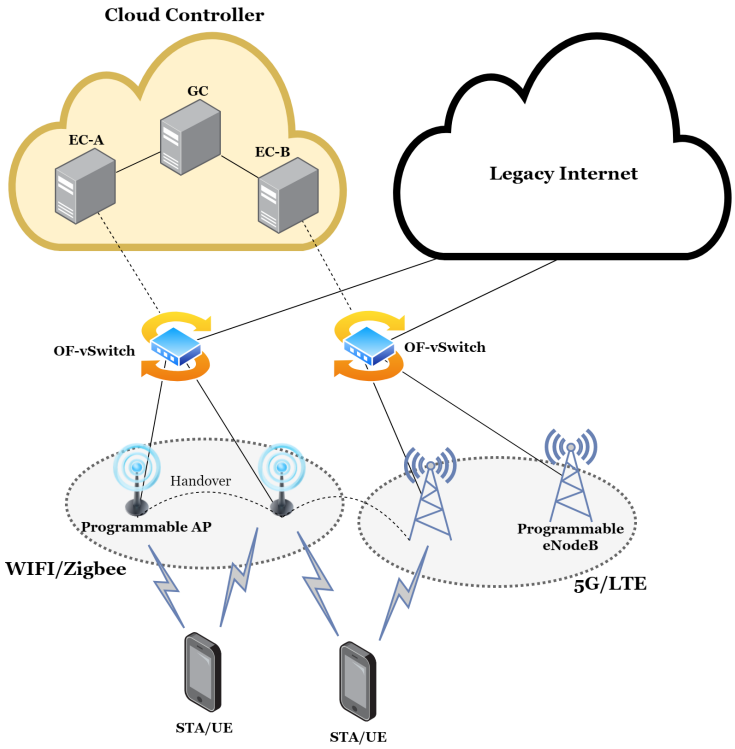


Figure 2.3: SDN-based 5G oriented network architecture [SLLL14]

2.2.1 Software-Defined Networking (SDN)

Today's Internet architecture has problems with the scalability of routing and support for mobility, as well as traffic engineering capabilities for domains both large and small. These problems are caused mainly by the lack of flexibility in the current architecture [Sta15]. To fix the issues, the Internet needs a flexible architecture which enables swift and straightforward configuration of the network equipment.

SDN is a network architecture concept that introduces the separation of the control plane and the data plane in communication networks, as presented in Figure 2.4. In traditional networks, servers or routers are responsible for making routing decisions and handle fast packet forwarding. Routing decisions and everything that comes with it is often referred to the control plane, while packet forwarding is referred to the data plane. Unlike the traditional network model where it is troublesome and error-prone to reconfigure network components after their initial configuration, SDN provides configuration flexibility through logically centralized control and programmable interfaces. The control plane may consist of one or

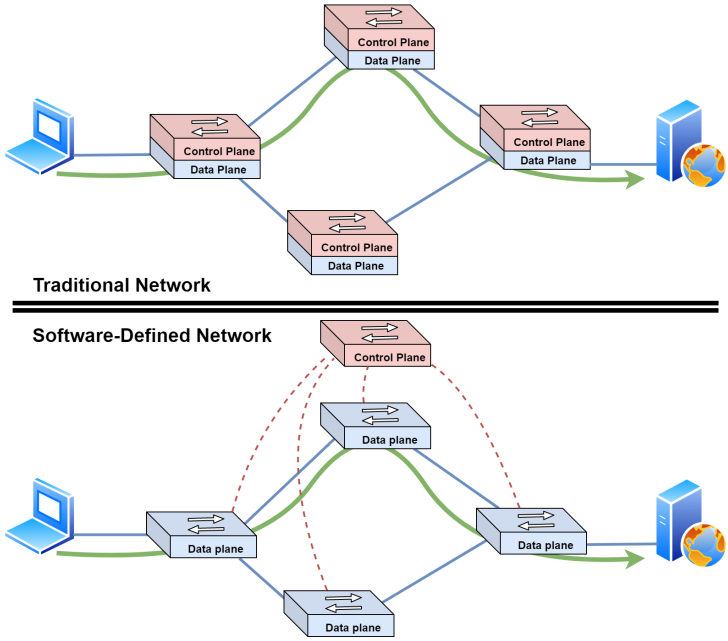


Figure 2.4: Routing and forwarding in Traditional Networks compared to Software Defined Network, illustrating the separation of the control plane and data plane [sdna].

more SDN controllers which are responsible for the decision-making logic of routing the networks traffic flow. The controller communicates with an upper application layer through a northbound interface protocol and with the underlying network infrastructure through a southbound interface, as depicted in Figure 2.5. The upper application layer supplies the logic of the SDN controller, and its decisions are enforced by installing forwarding rules on the switches in the network through the southbound interface. The switches represent the data plane as they are responsible for the packet forwarding in the network. For 5G, SDN will be important because it simplifies the network architecture, eases network management and provides improved configuration flexibility.

OpenFlow

There are several different protocol for the southbound interface for SDN, but the most prominent has become OpenFlow. OpenFlow is an open standard that allows the execution of experimental protocols in networks. It is developed and maintained by the Open Networking Foundation (ONF) and functions as a version of SDN. A controller is used to separate the control plane from the data plane in a switch. The controller manages routing decisions by installing rules called 'flow tables' in the

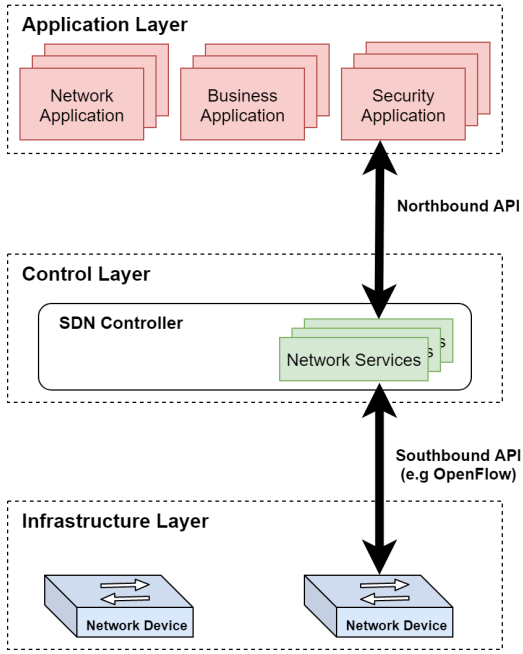


Figure 2.5: The three-layer architecture of SDN [LMK16].

switch. This solution presents high flexibility in routing of network traffic and makes it possible to easily alter the nature of a network device without disturbing the rest of the network traffic.

OpenFlow has become one of the most suitable options to solve the constraints and troubles of the present Internet stack. Commercial routers, switch and other network equipment have now included it as a feature by providing a standardized hook the researches can use to perform tests without needing the knowledge of vendor-specific behaviour of within network equipment.

Connection Setup: The controller and switch need to find each other to establish a connection. The switch can be pre-configured with the required IP address of the controller, making it automatically trying to connect, or the controller can detect the switch in the network and start connection setup from there. The connection setup begins with the switch initiating a secure TCP connection. After the connection is completed, they exchange hello messages to agree on which OpenFlow version to use. They both transmit information of the highest version each entity support, where the lowest of them is chosen. Then, the controller uses the Feature-Request message to inquire about the switch's capabilities, which is answered by the switch with Feature-Reply messages to complete the connection setup. The controller can

now use this TCP connection to manage the switch using the OpenFlow protocol.

Packet Processing: In ordinary Ethernet switches, a newly received Ethernet packet is handled by looking up the destination Media Access Control (MAC) address in its MAC table. The MAC table contains the mapping between MAC addresses and output switch ports. The Ethernet packet is then sent through the port that maps to the destination MAC address. If the source MAC address of the received Ethernet packet is not in the MAC table, it is now added and mapped to the ingress port.

This behaviour differs from how OpenFlow switches operate. An OpenFlow switch uses one or more flow tables to handle incoming packets, as illustrated by Figure 2.6. A flow table is comprised of flow entries where each entry contains match fields, counter and instructions. Entries are distinguished by their match fields which are a combination of a switch ingress port and different optional packet header fields, shown in Table 2.1 [PLH+12]. Matching can be done on fields from the link (L2), network (L3) and transport (L4) layer. A flow entry field can be wildcarded and use the value ANY, which makes it match all packets no matter the value of the packet header. Upon the reception of a packet, the switch extracts metadata from the packet called the key and tries to match it with the match fields in different flow entries, traversing through all flow tables until a match is found.

The packet type of the received packet determines the considered packet match fields in the flow table checks. If the packet header fields of the received packet have an equal value to the packet header fields used in a flow entry, there is a match. In the occurrence of a match, the counter of the matched flow entry must be updated, and the associated instruction must be executed.

If no match is found in any of the flow tables the default action for the switch is to create a Packet-In message which contains either the entire mismatched packet or parts of it. If only part of the packet is included in the Packet-In message, the switch will store a copy of the mismatched packet in a buffer and include the buffer ID in the Packet-In message. The Packet-In message is then sent to the controller.

Controller Behavior: The controller uses a secure channel to connect and control the switch by administering the flow table entries. When the controller receives a Packet-In message from the switch, the controller inspects the header fields of the packet to determine if a new flow entry is required and what actions should be applied. In the instance of creating a new flow entry, the controller generates a Flow-Mod message containing the new or changed flow entries and send the message to the switch, which installs the flow entry. Independently of creating a new flow entry or not, the controller finalizes the handling of the Packet-In by sending a Packet-Out message which informs the switch either which port the packet should be forwarded to or to drop the packet.

Table 2.1: Match fields included in OpenFlow 1.3

Layer	Field Name	Description
L1	in_port	Ingress port
L1	in_phy_port	Physical Ingress Port
L2	eth_dst	Destination MAC Address
L2	eth_src	Source MAC Address
L2	eth_type	Ethernet Type
L2	vlan_vid	VLAN ID
L2	vlan_pcp	VLAN Priority Code Point
L3	ip_dscp	IP Differentiated Services Code Point
L3	ip_ecn	IP Explicit Congestion Notification
L3	ip_proto	IP Protocol
L3	ipv4_src	Source IPv4 Address
L3	ipv4_dst	Destination IPv4 Address
L3	arp_op	ARP Opcode
L3	arp_spa	ARP Sender Protocol Address
L3	arp_tpa	ARP Target Protocol Address
L3	arp_sha	ARP Sender Hardware Address
L3	arp_tha	ARP Target Hardware Address
L3	ipv6_src	Source IPv6 Address
L3	ipv6_dst	Destination IPv6 Address
L3	ipv6_flabel	IPv6 Flow Label
L3	ipv6_nd_target	IPv6 Neighbour Discovery Target Address
L3	ipv6_nd_sll	IPv6 Neighbour Discovery Source Link-Layer
L3	ipv6_nd_tll	IPv6 Neighbour Discovery Target Link-Layer
L3	mpls_label	MPLS Label
L3	mpls_tc	MPLS Traffic Control
L3	mpls_bos	MPLS Bottom of Stack
L4	tcp_src	TCP Source Port
L4	tcp_dst	TCP Destination Port
L4	udp_src	UDP Source Port
L4	udp_dst	UDP Destination Port
L4	sctp_src	SCTP Source Port
L4	sctp_dst	SCTP Destination Port
L4	icmpv4_type	ICMPv4 Type
L4	icmpv4_code	ICMPv4 Code
L4	icmpv6_type	ICMPv6 Type
L4	icmpv6_code	ICMPv6 Code

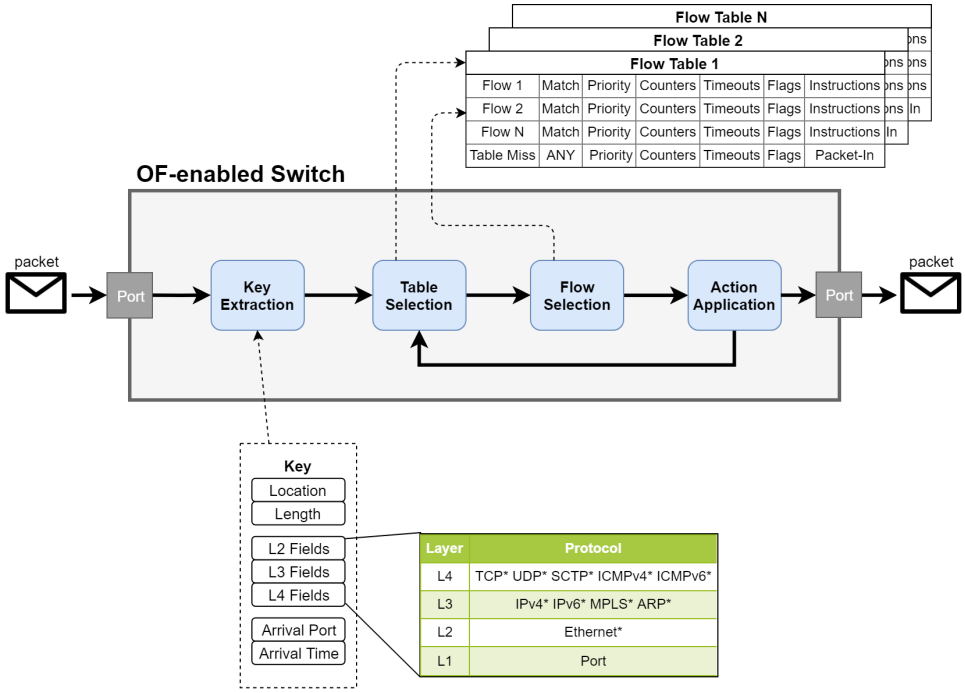


Figure 2.6: The Packet Processing of an OpenFlow-enabled Switch [OFP].

This interface between the switch and the controller provided by the OpenFlow protocol allow alteration of switch behaviour without directly programming it.

Deployment and use today

Before the launch of 5G, SDN is already widely used in the Internet architecture today. It was first deployed in 2010 by Nicira to control Open vSwitch (OVS) from Onix [KCG+10]. Google's B4 deployment in 2012 may be the most well-known use of SDN and OpenFlow, where it is was used to achieve near 100% utilization on many B4 links and an average of 70% utilization on all links over long periods [JKM+13]. China Mobile also has a large deployment in its Software-Defined Packet Transport Network (SPTN) which supports OpenFlow [sdnb].

2.2.2 Network Function Virtualization

NFV makes it possible to run most entities in a network architecture on off-the-shelf commodity hardware. It runs network functions as software in a virtualized environment, called Virtualized Network Functions (VNFs), thus making the need

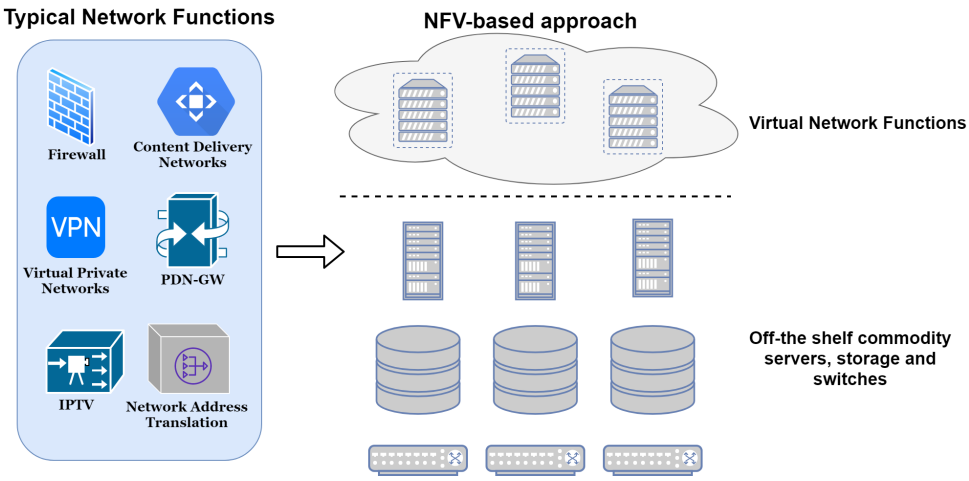


Figure 2.7: From dedicated network function hardware to VNFs running on off-the-shelf commodity hardware [HGJL15]

for nearly all specialized network hardware appliances obsolete. Figure 2.7 illustrates the transition from dedicated hardware appliances to network functions running in a virtualized environment. The advantages of VNFs is that they are cheaply deployable compared to installing and setting up physical hardware. They also provide flexibility, as a VNF can consist of several virtual machines that can be quickly started up and terminated according to what network function is required at a specific time or in a particular network architecture, which again improves the network's scalability. Examples of network functions that can be virtualized are load balancers, Intrusion Detection System (IDS), and firewalls.

2.2.3 Mobile Edge Computing (MEC)

MEC moves the concepts of Cloud Computing to the edge of the network to enable applications running in RAN proximity and closer to mobile users, as illustrated in Figure 2.8. The applications can be hosted in a virtualized platform at the edge to provide ultra-low latency and high-bandwidth service environment, which allows the development of applications requiring these capabilities. Examples of technologies requiring such capabilities are augmented reality, connected cars, and intelligent video acceleration [HPS+15].

All of these three leading technologies combine well and complement each other. The applications defined for NFV work with the SDN framework, and SDN also benefits from the use of these applications. NFV and MEC both use a virtualized

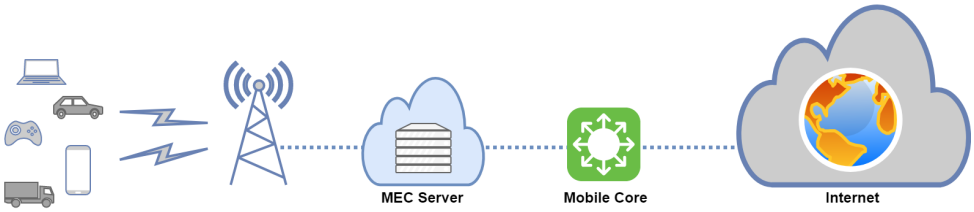


Figure 2.8: Example of MEC server placement in the network architecture [HPS+15]

approach to achieve their purpose, making it possible to host both VNFs and MEC applications on the same platform, with the same underlying infrastructure [HPS+15]. Because NFV makes it possible to deploy all network functions on virtual servers, the RAN can act as the edge of the cloud while the core functions can be regarded as the core cloud, while SDN can configure the connection between virtual servers in the two cloud-types [ea17].

2.2.4 Network Slicing

The three technologies explained above are the main enablers of a key concept in 5G called network slicing. The network slicing concept is slicing a physical network into several isolated virtual networks, each with its own set of flexible network capabilities and characteristics customized for a use case [ZLC+17]. The resources of the physical network are allocated dynamically to the different slices, depending on priority and demand of each slice [GK19]. Each slice is self-contained in terms of traffic flow and operations, and it can have its network resources, engineering mechanism and network provision. The purpose of the network slices is to provide support for adaptable creation of various use cases [ea17].

One of the main drivers for 5G is to accommodate various use cases with different requirements on the same infrastructure. A large number of use cases have been defined so far, where they vary according to the visions and needs from different project, organizations and vertical industrial sectors. The defined use cases are grouped in to three general application scenarios, namely Enhanced Mobile Broadband (eMBB), Massive Machine Type Communication (mMTC) and Ultra Reliable Low Latency Communication (URLLC) [LAA+18, p.32]. Figure 2.9 illustrates the three application scenarios being supported on a single physical architecture using network slicing.

eMBB is a scenario which aims to improve the current mobile broadband services by enhancing data rates, traffic volume, coverage, and seamless mobility. Some

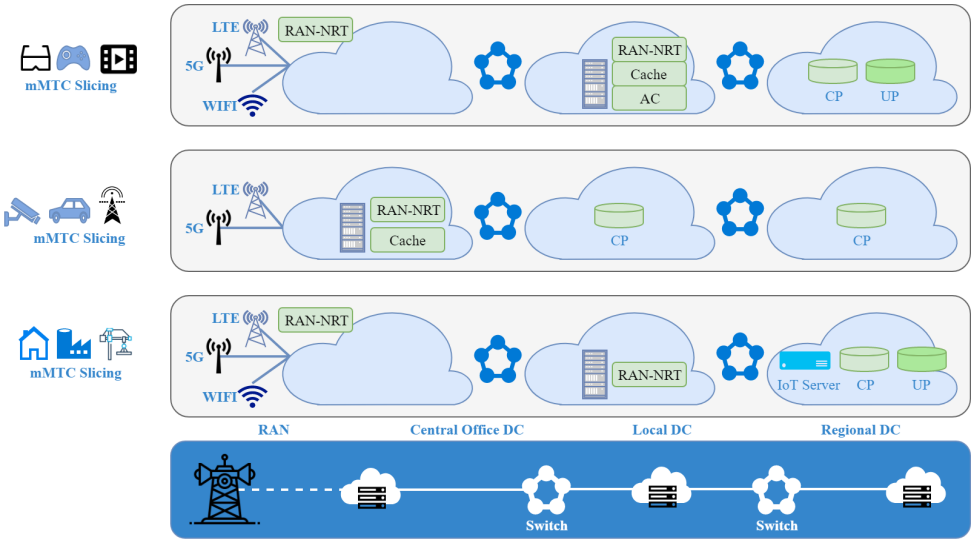


Figure 2.9: A single physical architecture supporting Enhanced Mobile Broadband (eMBB), Massive Machine Type Communication (mMTC), and Ultra Reliable Low Latency Communication (URLLC) using network slicing, where each use case supports different services such as RAN-Non-Real-Time (NRT), Cache, Composable Control Function (CP), Programmable Data Forwarding (UP), and Admission Control (AC) in various data centers (DC) [HPS+15].

examples of defined use cases are augmented reality, virtual reality, and ultra-high definition video.

The mMTC application scenario refers to the transmission of small amounts of data between a large number of connected devices in a high-density network, typically IoT sensors or utility meters. Use cases that fit this scenario are, for example, smart city, smart meters, and video surveillance.

URLLC is a scenario which focuses on mission-critical communications, which are latency sensitive, require high reliability and have strict security demands. Some examples of use cases that fall under this category are autonomous driving cars, smart grid, industrial automation and control, and real-time medical services.

2.3 5G threats

The 5G mobile communication system is expected to have a more critical role and a higher impact on society than any previous network generation. Therefore the motivation to threaten and attack will also be stronger and have a more extensive range.

The motivation for attacking URLLC applications used in critical infrastructures can be due to economic gain, political motives, cyber warfare, espionage, and more. Threat agents that can have such reasons and are capable of a successful attack can be internal, in the form of malicious insiders, or external with adversaries such as organized crime cartels, nation-state intelligence services, and cybercriminals. Any attacks on critical infrastructures through 5G is expected to require a high level of skills and resources to execute. Therefore it is most likely that the most frequent attackers will come from or be sponsored by politically motivated agents such as nation-states [LAA+18, p.66-69].

With the introduction of new technologies to the mobile communication network comes a new set of security challenges. Recent research has revealed many issues that need to be solved. The authors of [All16] have identified seven key security challenges in 5G, but in this thesis, we will focus on two of them.

1. The resources of the network infrastructure will be vital for its function, as the number of network connections is expected to grow tremendously. Growth in network connections increases the possibility of vulnerabilities such as the visible network control elements, and unencrypted control channels to be exploited as a part Distributed Denial-of-Service (DDoS) attacks.

   Such attacks can target the signalling plane, user plane, management plane, support systems, radio resources, and logical and physical resources within a virtualized infrastructure to directly deplete the resources of the network infrastructure which supports 5G users and devices.

2. Denial-of-Service (DoS) attacks may also be directed at specific users or devices by attacking physical resources such as battery, memory, disk, CPU, radio, actuators and sensors or logical resources such as operating systems, applications, configuration data, and patching support system.

These security issues are important to address as they may jeopardize the entire network if exploited, both in terms of functionality and trust between users and service providers. Some of the identified use cases of 5G, such as smart cities and autonomous cars, relate to the critical infrastructure of nations. If such systems were to be exploited, it would cause massive amounts of damage both societally, financially and individually. Therefore these security issues must be solved if 5G is to be the successful and disruptive technology as it is expected.

2.3.1 Denial-of-Service (DoS), Distributed DoS (DDoS), and Botnets

A DoS attack can be described as an attack that aims to prevent a targeted network resource from answering and serving requests from legitimate users.

A DDoS attack uses several compromised systems in a coordinated fashion to decrease or hinder the availability of network service. The collection of compromised systems, called bots or zombies, used in the attack are often referred to as a botnet and are usually distributed globally.

As described in [TZJT13], a botnet commonly consists of bots, handlers, and one or more botnet masters which centrally manage the entire network and are controlled by the attacker. The handlers are mechanisms that allow the masters to communicate with the bots. Communication is necessary for the master to control them by sending commands. Zombies are network devices that the handlers have corrupted and use to perform the actual attack. A device gets typically compromised and added to the botnet through malicious software (malware) such as worms, Trojan horses or backdoors without the system owners knowledge. The zombies typically stay dormant for an extended period until the botnet has grown big enough to serve its purpose and the botnet master remotely orders the attack command.

The resources of the botnet allow DDoS attacks to be significantly more extensive and disruptive than a normal DoS attack. With a botnet dispersed worldwide, the botnet master and original attacker can hide behind the botnet by using spoofed IP addresses for the zombies, making it much more complicated for defence mechanisms to detect the attack and identify not only the architect but also the zombies.

2.3.2 Flooding attacks

There are many versions and ways to perform a DDoS attack, but in this thesis, the focus will be on flooding attacks, especially SYN flood attack as it is a part of the practical section of this thesis. [BSS17] defines a flooding attack as a DDoS attack that attempts to exhaust either the bandwidth or the resources of a host by overloading it with an enormous amount of network traffic. The different types of DDoS flooding attacks can be grouped into the following three main categories:

- Protocol exploitation
- Reflection-based
- Reflection and amplification-based

The subsequent sections describe the categories along with some examples.

Protocol exploitation

Protocol exploitation attacks take advantage of a feature or implementation vulnerability of some protocol used by the victim.

SYN FLOOD ATTACK: The aim of a SYN Flood attack is to exploit the vulnerability in the second stage of TCP's three-way handshake process to make a server unable to perform its function.

The TCP three-way handshake is the process that is used in TCP protocol to negotiate and create a connection between two network entities. Figure 2.10 illustrates the steps of the three-way handshake protocol. It consists of three message exchanges. First, the client (Host A) initiates the connection by sending a synchronize (SYN) packet to the server (Host B). The packet contains a random-generated sequence number M which the first packet of the clients' transmission will have.

Second, Host B receives the SYN packet and creates a half-open TCP connection by initiating a Transmission Control Block (TCB) to uniquely identify the connection and thereby bind resources on the server to be used when the connection is established. For TCP devices to be able to manage multiple connections, each connection needs to be uniquely identified to keep them separated. In TCP each connection is defined by the IP address and port number of the two connected devices. The unique data structures named TCBs are used to store this information at both devices. TCBs also include connection information such as pointers to incoming and outgoing data buffers, current window size, and counters for numbers of bytes received, number of bytes acknowledged, etc.

Then Host B replies to Host A by sending a SYN-acknowledgement (ACK) packet. The SYN-ACK packet contains the destination host's own randomly generated sequence number and an acknowledgement number for the reception of the first SYN packet, which is Host A's sequence number incremented by one. Finally, Host A receives the SYN-ACK from Host B and replies by sending an ACK message consisting of an ACK number that equals Host B's sequence number incremented by one.

The vulnerability of the TCP protocol is that the receiver of a TCP SYN request will create a half-open connection and reserve resources for the connection until it receives an ACK message or a timer runs out. Figure 2.11 illustrates how an attacker exploits the three-way handshake protocol in a SYN flood attack. The attacker uses the reservation of resources by flooding the server with SYN requests to make the server bind all its resources and make it unavailable to connect with legitimate users.

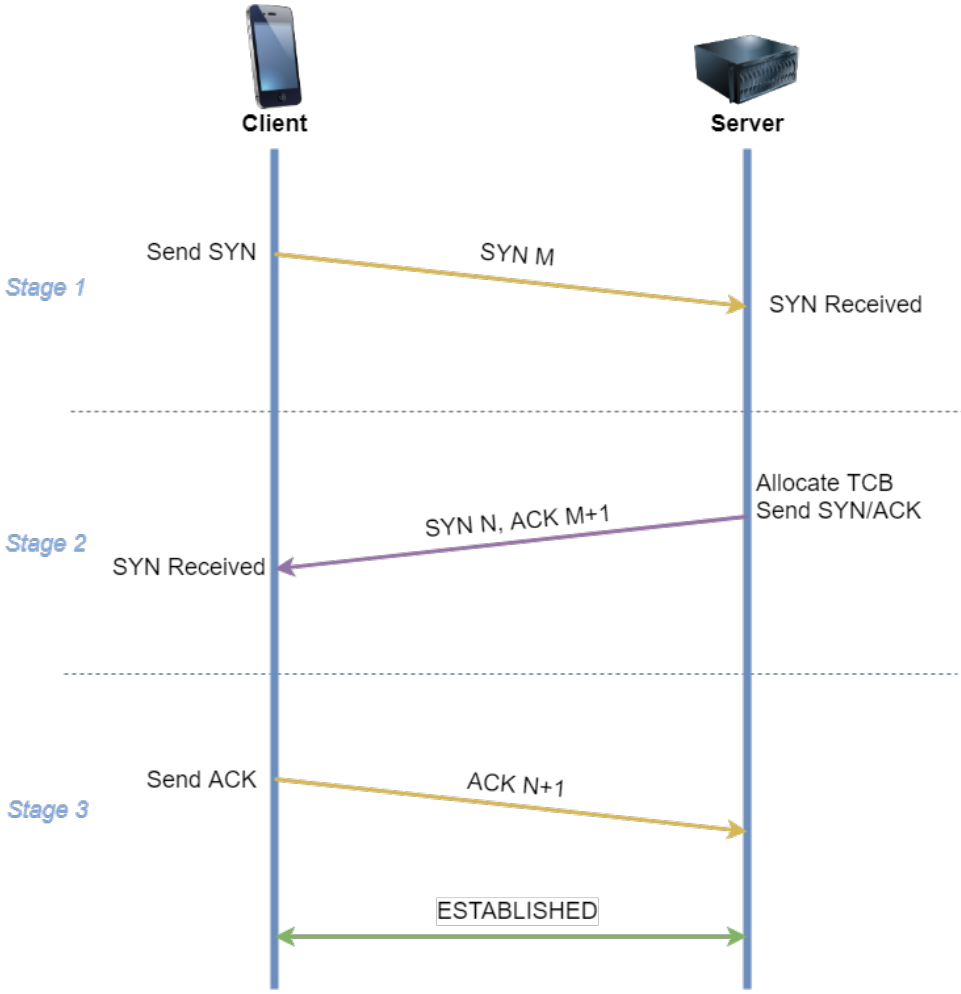


Figure 2.10: Normal TCP 3-way handshake

Even if the server restarts or frees up all its resources, the intensity and length of the SYN flooding attack may cause all the resources to instantly by starved again.

The attack can be made more efficient by exploiting the time the server waits for an ACK respond from the connecting host. The attacker can choose to not reply to all of the SYN-ACK packets from the server or choose to send the SYN request with a spoofed source IP address, making the server send the SYN-ACK packet to an IP address which will not reply as it did not send the original SYN request.

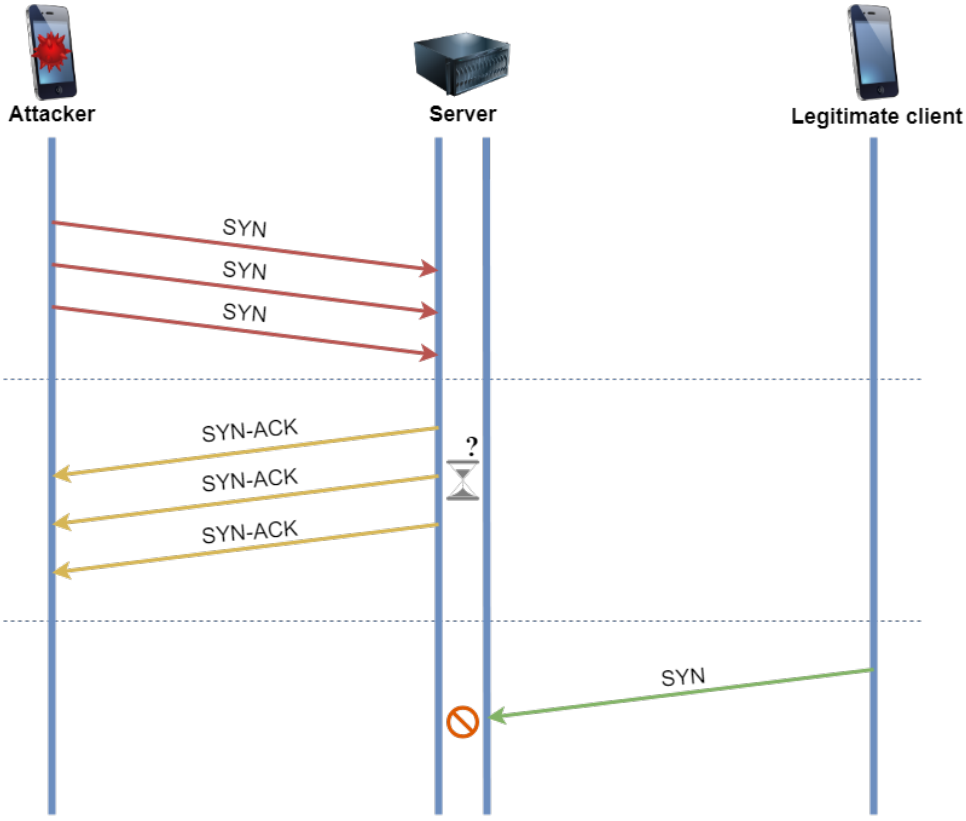


Figure 2.11: Simplified SYN Flood attack

UDP FRAGMENTATION ATTACK: Every network device has a size limit for received packets called the Maximum Transmission Unit (MTU). Any packets that are larger than the MTU has to be fragmented into smaller packets before transmission, and the receiver has to re-assemble the fragmented packets to get the original packet. According to [KPS03], a User Datagram Protocol (UDP) Fragmentation attack exploits the reassembly mechanic by sending forged UDP packets larger than the MTU to the victim. The size of the packets is usually from 1500 bytes and larger, causing them to be fragmented. Because these packets are forged, they cannot be re-assembled, casing the victim to consume resources in the attempt. The advantage of this attack is that the attacker can waste more of the victim's resources with fewer packets.

Reflection-based

The main principle behind reflection-based flooding attacks is to exploit the challenge-response mechanics used in authentication systems to generate illegitimate response traffic towards a specific target. In the text below, there are two examples of attacks that utilize this principle to perform a DDoS flood attack.

SMURF ATTACK: As described in [TZJT13], a Smurf attack is based on forging the victim's IP address and use it as the source IP address in a tremendous amount of Internet Control Message Protocol (ICMP) packets that get broadcasted in a network using an IP Broadcast address, making the receivers of the packet believe that the victim host is the sender. All the nodes in the network reply to the ICMP packet by sending ICMP response packet to the victim, causing massive network traffic that overloads the victim as well as exhausting the network's resources.

FRAGGLE ATTACK: A Fraggle attack uses the same attack principle as the Smurf attack by generating UDP traffic instead of ICMP traffic.

Reflection and amplification-based

Reflection and amplification-based flooding attacks are attacks that use reflection based attack methods that generate larger response traffic than the initial request traffic to amplify the attack.

DNS AMPLIFICATION FLOODING ATTACK: The Domain Name System (DNS) is an essential Internet component that provides the mapping of domain names to IP addresses. A DNS query is of disproportional size to the DNS query response, caused by the DNS query response often including the original DNS query. A DNS amplification flooding attack takes advantage of this by sending DNS queries with the victim's spoofed IP address to generate massive DNS query response traffic that exhausts the victim's resources. The attacker can amplify the attack by using different DNS extensions in the query such as the EDNS0 DNS protocol extension, which allows for large messages to be sent. An attacker typically uses a botnet to generate more query requests as well as hiding the attacker identity. According to [PLR07], query traffic can be amplified with a factor of 73 using different amplification methods.

NTP AMPLIFICATION FLOODING ATTACK: The Network Time Protocol (NTP) amplification flooding attack has a similar attack methodology as the DNS amplification flooding attack, except it uses NTP servers as reflectors instead of DNS servers.

2.4 Countermeasures to SYN Flooding Attack

As this master thesis focuses on detection of SYN flood attack, it only covers the existing countermeasures of the attack. According to [HDK02], all SYN flood attack detecting techniques can be categorized into two broad categories.

- End-Host Countermeasures
- Network-Based Countermeasures

The categorization is based on where the detection method is installed and which component is responsible for the detection.

2.4.1 End-Host Countermeasures

Server-based, or end-host based, SYN attack countermeasures are performed in the end-host, either by tweaking some parameters or using specialized software or data structures to mitigate the effects of a SYN flood attack [Edd06].

SYN Cache

A SYN cache is used to prevent a full connection state to be allocated for a TCB at the very moment a SYN packet is received. It postpones the full state allocation until the three-way handshake is completed by storing a subset of the TCB data in hash buckets within a global hash table structure, instead of a traditional per-socket linear chain of incomplete queued connections. Each bucket has a limited amount of space, and there is a limited number of entries allowed in the table to provide an upper boundary for the amount of memory the hash table is allowed to occupy. If the TCP connection is completed, the data can be moved to allocate the full TCB [Edd06].

SYN Cookies

SYN cookie is similar to the SYN cache as both countermeasures want to postpone the state allocation for a TCB until the TCP connection is fully completed. SYN cookies do this by not allocating any connection state when the initial SYN packet is received, but instead compress the state information and add it to the bits of the sequence number used in the SYN-ACK reply packet. If an ACK segment is received to complete the connection, it contains the sequence number incremented by one, which can be used to regenerate the state information and instantiate the full TCB safely. The drawback of this approach is that the Sequence Number field is only 32-bit and does not allow all the TCB data to be included, which forces some TCP

options to be excluded. Since the end-host requires state to retransmit SYN-ACKs, this is not possible using SYN cookies [Edd06].

Increasing TCP Backlog

Configurations can be done to the end-host to mitigate the effects of SYN flood attacks. As the end goal of a SYN flood attack is to exhaust the resources of the target by overflowing its TCP backlog, a natural solution to the problem is to increase the backlog size. Increasing the backlog size is not a very effective countermeasure as it is probable that the attacker may be able to counter this countermeasure by increasing the size of the attack [Edd06].

Reducing the SYN-RECEIVED Timer

Another configuration that can be done is to reduce the SYN-RECEIVED Timer, and thereby decreasing the time before the allocated resources of an uncompleted connection attempt are released. Like the previously mentioned countermeasure, this is not very effective either, and may have a reversed effect against very aggressive attacks as more legitimate hosts that are amid ACK retransmission attempts may be dropped [Edd06].

2.4.2 Network-Based Countermeasures

Firewalls

The firewall is an important security measure to protect a server before the establishment of new connections. It filters all packets headed towards the server. The processing of all packets is a drawback as it adds an extra delay for all incoming communication.

SynDefender is a firewall-based countermeasure software developed by Check Point [Synb] as apart of the Firewall-1 product. It intercepts all SYN packets on behalf of the host it is protecting and relays the packets after the three-way handshake is completed.

Syn Proxying is another version of a firewall-based countermeasure that Junos OS uses to defend against SYN Flood attack [Syna]. Its function is very similar to the SynDefender, but the Syn proxying is only active once the number of SYN packets per second from a similar ingress interface and with a similar destination address reaches a certain threshold. The proxy intercepts all SYN packets on behalf of the host and can either store the incomplete connection attempts or drop them.

MULTOPS

Multi-Level Tree for Online Packet Statistics (MULTOPS) is a detection scheme proposed by Gil and Poletto in 2001 [GP] that uses packet monitoring to identify potential attacks. The scheme is based on the assumption that the uplink and downlink packet rates should be reasonably similar and therefore monitors the packet rate in both directions to detect disproportional differences. It uses a dynamic tree structure to store the data rate statistics, which itself can be vulnerable to targeted memory exhausting attack according to the authors of [PLR07]. Once a bandwidth attack is detected, mitigation measures like source ingress filtering can be applied. Attackers can easily countermeasure MULTOPS by using randomly spoofed IP addresses, which will interfere with the data rate statistics of legitimate IP addresses and make them unreliable.

Chapter 3
SDN-based security applications and Related Works

3.1 Security attributes of SDN

As Software-Defined Networking (SDN) is expected to be an integral part of the 5G architecture, it is anticipated that the technology can become a crucial contributor in the defence against Distributed Denial-of-Service (DDoS) attacks. SDN brings unique features like a global view of the network, centralized control and provides programmability of the network elements to the Internet architecture that can prove powerful in a security context.

The separation of the control and data plane and logically centralized controller give SDN both an overview of network flows that can be used to gather network statistics as well as the ability to scrutinize data packets. Such features can be used for real-time traffic analysis and attack detection. The centralized control and the flexibility provided by the programmable interface also enable SDN quick, broad and adaptable responses to mitigate detected malicious events in an ever-changing and rapidly evolving threat landscape.

Despite the security attributes of SDN, it is still vulnerable to network attacks such as DDoS attacks [YGY17]. The crucial role of the controller in a SDN-based network makes it a potential target, as forged packets that does not match any flow tables can be sent by attackers with high frequency and volume to deplete the controllers resources, causing increased delay in the network or even network unavailability [YGD16]. The security challenges of SDN needs to be dealt with if it is to be used as a security component. In this thesis, the security problems of SDN are out of scope and therefore assumed solved.

In this section established SDN-based DDoS attack detection and mitigation techniques are explained.

3.1.1 SDN-based DDoS Detection Techniques

Entropy-based

The paper [BSS17] defines entropy as a way of measuring the randomness of an attribute in a specified time window. It indicates the likelihood that an event occurs in relation to the total number of events and has proven to be a useful tool for assessing randomness in dataset analysis. Intrusion Detection System (IDS) typically apply entropy-based techniques to detect anomalies by computing the entropy value of characteristics like source and destination IP address, packet rate or packet size.

The advantage of entropy-based detection is that it does not add any network traffic or need specialized hardware when used. It has also proven a lower rate of false positives caused by its higher sensitivity and ability to detect fine-grained patterns.

The disadvantage of entropy-based detection is its limitation to solely assess one attribute when analyzing datasets and present the results with only a single value. This causes some critical contextual information to be lost in the process. As entropy indicates randomness, detection techniques based on it can only detect anomalies that interfere randomness.

Machine learning

Machine learning-based detection methods evaluate a large amount of network and traffic characteristics to detect anomalies. It uses a collected dataset of real network traffic to train the detection algorithm to identify malicious patterns. Therefore the quality of the dataset used for training is heavily linked to the accuracy of these techniques. Well known machine learning concepts like fuzzy logic, Bayesian networks, self-organizing map (SOM), and artificial neural networks are commonly used in these types of detection methods [BSS17].

Traffic Pattern Analysis

Detection techniques based on traffic pattern analysis is based on the assumption that malicious or infected host of a botnet has similar behaviour which differentiates from the behaviour of benign hosts. The assumption is based on that a botnet is typically controlled by a single botmaster which issues the same command for the entire botnet to execute an attack. Similar packets per second or bytes per second are traffic patterns that can indicate a potential threat. Characteristics like common destination, similar connection time and related platform features can be used to identify malicious hosts [BSS17].

Connection rate-based

Connection rate-based detection techniques can be separated into two main types, namely connection success ratio and the number of connections established [BSS17].

The connection success ratio is based on the assumption that a benign host should have a higher connections success ratio than a malicious host. If a host's success ratio gets below a certain threshold within a time window, it is marked as malicious.

The number of connections established is based on the assumption that a malicious host attempts to establish more connections in a short period than a benign host. It is also presumed that an uninfected host has a lower connection attempt rate and more likely reattempt to connect to a recently connected server. If a host has a higher connection attempt rate than a certain threshold, it is marked as malicious.

3.1.2 SYN-based DDoS Mitigation Methods

When a DDoS attack is detected by one of the detection techniques mentioned in the section above, the attributes of SDN allow different techniques to be used to mitigate the attack. Standard mitigation techniques like dropping packets, blocking ports, or redirection of traffic can be easily performed using SDN. SDN can enforce these techniques by sending Flow-Mod messages to its switches and install new or edit flow entries to include the mitigation techniques as the action.

Dropping packets can be implemented by having packets that match the characteristics of the detected attacker to be dropped. Examples of match fields that can be used are source IP address and destination IP address. Packets can also be dropped by having the default action of unmatched packets to be drop. All traffic from an attacking port can be blocked by adding a flow entry with the only match entry being an ingress port number and the action to be drop. Traffic can be redirected by forwarding legitimate traffic to a new IP address.

Because mitigation techniques like packet drop and port block disallow traffic completely, they risk discarding legitimate traffic as well. Traffic redirection causes the targeted server's connections to be terminated and added processing delay on the network traffic.

SDN allows the use of some alternative mitigation strategies. Network reconfiguration can be used to control the bandwidth of different switch interfaces to prevent bandwidth exhaustion. The network topology can be changed to eliminate a detected attack path and create new paths for legitimate traffic by altering each switch's flow table.

3.2 Related Works

The paper [MKK17] suggests an architecture using intrusion detection systems (IDSs) together with SDN to make an anomaly detection system for mobile network operators. The suggested architecture includes two switches, a SDN controller, a SDN application, several Detection-as-a-Service (DaaS) nodes, and a clustering algorithm. The architecture aims to stop user-generated malicious traffic from going further than the gateway. They propose copying every received packet by a switch and send the copy through a clustering algorithm which determines which DaaS node should analyze the packet based on load balancing and packet characteristics. The DaaS node analyzes the packet and notify the SDN application if a malicious packet is detected. The SDN application, which runs on top of the SDN controller, informs the SDN controller to block the traffic, which is enforced by the SDN controller installing appropriate flow entries in the switch. This paper does not specify the detection algorithm used by the DaaS nodes or give any specifics on which DDoS attacks it aims to detect and prevent. The study does not perform any testing to evaluate and validate the efficiency of the suggested architecture.

The authors of [CGM+18] have investigated the use of a security framework for the Internet of Radio Light (IoRL) system that is integrated and based on SDN. The proposed framework is designed to detect and mitigate different versions of TCP SYN Scanning attack. In the system, the SDN controller tracks the state of each TCP connection that gets attempted through the switch by receiving a copy of every incoming packet by the switch. The TCP connection state information is used to count the number of unfulfilled connection attempts by a host. If the number of unfulfilled connection attempts during a predefined window of time exceeds a preset threshold, the source IP address of the attempts gets banned for a configurable amount of time by the controller adding a flow entry that drops all packets from the source IP in the switch. The authors have used a test-bed to evaluate the effectiveness of their proposed security framework with great results.

In the paper [TSD16] the authors present an OMNet++/INET extension that enables some performance and security evaluation of SDN networks. The extension includes a framework to create security attacks, support to SDN-based monitoring system by introducing two new OpenFlow messages named OFPT_STATS_REQUEST and OFPT_STATS_REPLY that enable the SDN controller to collect flow statistics from the switch to use in anomaly detection. They present a simulation of a simple DoS attack scenario that includes a SDN controller, a SDN switch, four client hosts and three server hosts that run UDP applications. The controller uses entropy-based techniques combined with a threshold for the transmission rate of a host to detect anomalies. The controller mitigates detected attacks by installing a drop action for the malicious host as a flow table in the switch. The extension is currently under

work, and the source code is made available.

In the letter [YGY17] a scheduling method for SDN controller is suggested to mitigate DDoS attacks targeted at the SDN controller. The presented scheduling method is called “MultiSlot” and is based on a logical multi-queue architecture which uses time slicing as allocation strategy. The multi-queue scheduling method helps the controller to distinguish normal functioning OpenFlow switches from switches that are under DDoS attack, and thereby prioritize serving the legitimate traffic and preventing the SDN controller from getting overloaded by false flow requests.

The paper [YGD16] presents a detection method for DDoS attacks in an SDN Network based on fuzzy synthetic evaluation. The detection method uses entropy to quantify every factor used in the fuzzy comprehensive decision. The goal of the method is to detect attacks targeted at the SDN controller or switch. They perform multiple simulations in Mininet using a POX controller to test the detection method against various DDoS attacks with different attack intensity. The authors also evaluate their method up against other suggested DDoS detection algorithms by conducting a comparable experiment.

As presented in the related works above, there has been a lot of theoretical works related to SDN-based network security defence systems and architectures and practical works with SDN as a security service. To the extent of our knowledge, there has not been any practical work with the combination of SDN as a security component against DDoS in a 5G network environment. The contribution of this thesis to the state-of-the-art is a practical evaluation of an implemented SDN security application in a close-to 5G scenario.

Chapter 4
Methodology, OMNeT++, and extension frameworks

4.1 Methodology

In this section, the research methodology used in the master thesis is presented. The section defines the chosen methodology, give the reason for using it and describes how it has been utilized when performing the master thesis work.

To carry out the master thesis in a sound scientific manner, it was necessary to equip ourselves with some methods to control the thesis work and mindset iteratively through the work period. The methods should ensure that we work continuously towards the same goal, even when unforeseen challenges appear. The method should be a tool which enables systematic problem solving as well as verify and ensure the correctness of obtained results before taking on a new task.

To identify these methods, elements of the Design Science methodology has been adopted. Design science focuses on the design and evaluation of artifacts in a context by employing a build-and-evaluate loop that suits software engineering research [Wie14]. An artifact can be anything built or designed by a human, while the context is anything that interacts or affects the artifact. The main goal of any research related to computer science is to solve or mitigate a problem. First, the problem needs to be identified, then a possible solution needs to be suggested. It is not necessary yet to know if the suggested solution is the optimal one. The iterative research process provided by guidelines from the design science methodology enables us to test a suggested solution in a build-and-evaluate loop, where each loop iteration will either move us towards or away from a solution to the problem. With either outcome, knowledge is acquired about the problem and potential solution.

Figure 4.1 depicts the build-and-evaluate loop that is used in the thesis. The iterative process typically starts with an idea of a potential solution to a problem. An artifact is developed and tested in a context. The performance of the solution is measured, and the generated data is analyzed to acquire knowledge about the problem

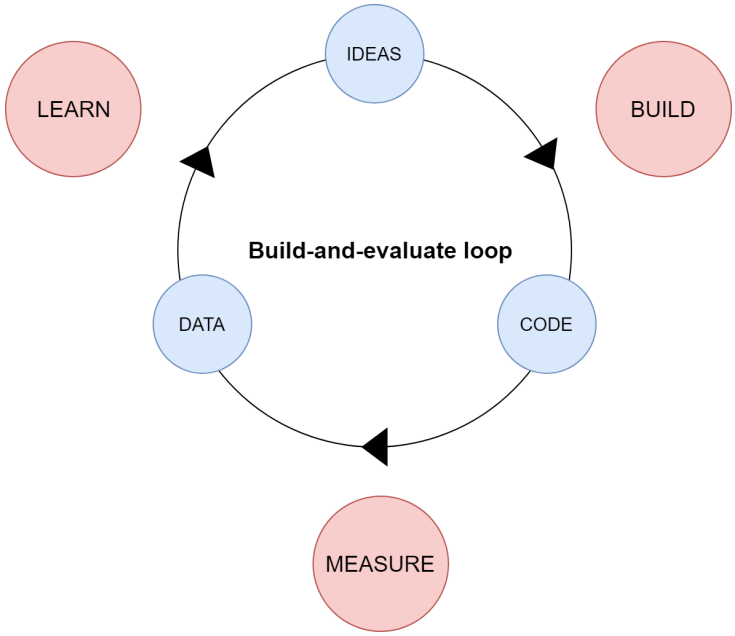


Figure 4.1: Build-and-evaluate loop, adapted by [And18].

and the suggested solution. New knowledge can be gathered through literature review if necessary, to created new ideas and improve the suggested solution. The loop is continued until a satisfactory solution to the problem is developed. This process was used throughout the experiments.

4.1.1 Literature review

A vital part of the thesis has been to acquire new knowledge by searching and analyzing information sources like articles, books, various documents, web pages, and other literature. A literature review has been used as a method to evaluate and select sources for this thesis. To locate relevant literature, search engines and online libraries such as orio.no, Google Search and Google Scholar were used, as well as inquiring with the thesis supervisor and Google Groups. The criteria for choosing sources were their relevance, credibility and preferably state-of-the-art. The literature should increase knowledge and contribute to the theory in the thesis. When evaluating the credibility of the literature, the publisher, author, and publication date are considered. The number of citations, their references and peer-reviews are also useful to include in the evaluation.

4.1.2 Simulation

Simulation is chosen as the primary method for evaluating our research questions because it is the closest available method for observing the artifact in a real life environment with real users, hardware and traffic. Observing the real world is too expensive and too slow, and as 5G is not yet deployed or standardized, it is also not possible. By using simulation, we get a flexible environment which can test a large number of parallel simulations with different parameters in a matter of seconds or minutes. We explored the option of using actual deployed test beds, but they are not available. The artifacts of my thesis and their context are explained below.

When deciding on which simulation tool to use, three potential options were considered during the pre-project of the thesis. SoftFire, which is a federated experimental testbed [sof], OpenAirInterface [opeb], which is an open-source hardware and software wireless technology platform, and OMNeT++ [omnb], which is an extensible and modular discrete event simulator. After emailing with the project administrators of SoftFire, it was discovered that the testbed and the entire project were only temporary and ended in Spring 2018. Between the two remaining options, OMNeT++ was chosen over OpenAirInterface mainly because of its graphical run-time environment, its extensive documentation, the flexibility and re-usability of models and components, its ability to record and present results and statistics, and finally the possibility of parallel distributed simulation. The disadvantage of OpenAirInterface was its lack of interface, less documentation than OMNeT++ and less flexibility than OMNeT++'s modular implementation. OMNeT++ and its extension libraries are presented in more detail in the following sections.

4.2 OMNeT++

OMNeT++ is an extensible, modular component-based C++ simulation library and framework. The framework is not itself a simulator but allows the creation of tools to be used for simulation and provides the environment to run them. All OMNeT++ simulations are built up of modules interacting with each other. The modules can be connected through gates (or ports) and interact using messages sent through the gates. Connected modules are called compound modules. There is no limit to the hierarchy of compound modules, and even the network model itself is a compound module. The lowest in the module hierarchy is referred to as simple modules [omnc]. Figure 4.2 illustrates how simple modules, compound modules, and network interrelate.

A simple module is built by C++ code and organized into two main files, namely the C++ file and the Network Description File (NED). The C++ file provides the behaviour of the module in C++ code. The NED file mainly provide default values

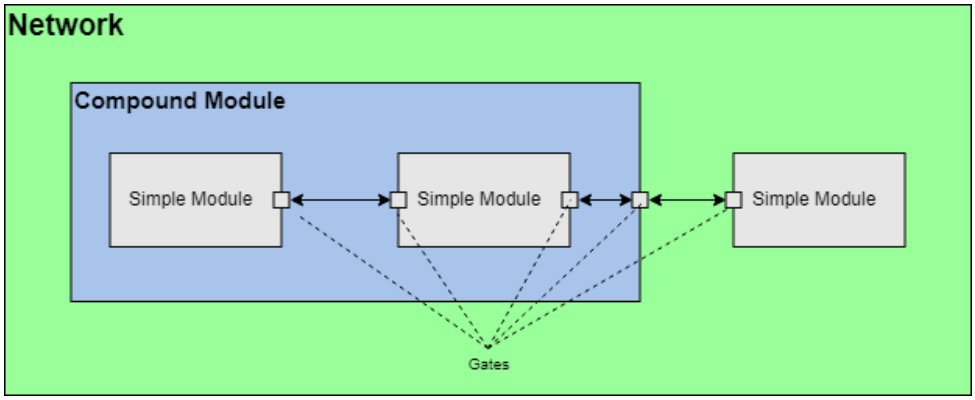


Figure 4.2: A concept model of an OMNet++ network comprised of a compound module and a simple module

for parameters declared in the corresponding C++ file and declare gates which can be used for input and output communication with the model. It can also be used to configure the module's look in the graphical interface. All NED files are written in OMNeT++'s own created NED language.

The NED language has been created to make it easy for developers to create and configure their network topologies. It allows connecting reusable modules to create compound modules or models. The ability to create and combine simple modules into compound modules, and again combine compound modules to more complex compound modules, is powerful and facilitate developers to create OMNeT++ extension packages/projects which can be reused by other users to create their simulations.

The configuration file omnetpp.ini is often created before simulating in order specify other parameter values than the default values set in the NED file. Parameters can be used to determine module behaviour and configuration during simulation in addition to parameterizing the topology.

When running a simulation, OMNeT++ has some options. A graphical interface called Tkenv displays the topology during the simulation and animate events such as message passing. The user has options to slow down, speed up, or stop the simulation while it is running. The graphical interface shows an overview of all messages passed in the network as well as modules output. This can also be shown for a specific module, which can be useful for debugging and verification purposes. The cost of running simulations in the graphical interface is longer execution time and the limitation of one simulation at a time. The command-line interface Cmdenv allows

for parallel and batch simulations and performs better than the graphical interface concerning execution time.

The results of OMNeT++ simulations are recorded as output vectors and output scalars through built-in support. Output vectors are data that have a connected time value, such as e.g end-to-end delay or queuing times, and are recorded in series. Output scalars are single values the summarize a collection of recorded results during the simulation. It is calculated after the simulation is completed and all results have been recorded. Examples of output scalars are mean, sum, average, min, max, and count.

To record results, the user can either use the signal mechanism or record directly from the C++ code. The signal mechanism works by the developer declaring a signal and implementing it to be emitted in the source code of a module. By including that the signal should be recorded in the configuration file, OMNeT++ automatically create a listener for the particular signal which records all emitted signals and performs the specified calculations to produce the wanted result.

To provide OMNeT with the capabilities of simulating SDN in a close-to 5G environment, the third party extensions OpenflowOMNetSuite [GZGTG16] has been chosen as a simulation framework for an SDN-controller and SimuLTE [VSN14] as a modular system-level simulator for LTE-A Networks. The existence of these two third-party extensions was an essential factor when deciding to use OMNeT++ as network simulator in this thesis.

4.3 INET

INET can be considered as the standard protocol model library for OMNeT++, as it provides the most common Internet protocols as re-usable modules, and additionally, several simple applications and entities [omnd]. The full TCP/IP stack, support for mobility, and wireless technologies are implemented [ine]. Both SimuLTE and OpenFlowOMNeTSuite use modules from the INET library as a foundation of their models.

4.4 SimuLTE

SimuLTE is an LTE simulator and developed as an extension for the OMNeT++ simulation framework. It has implemented the data plane of the LTE-A RAN and EPC with scheduling capabilities, a realistic physical layer, and a full protocol stack according to LTE-A standards [VSN14].

To simulate the LTE-A RAN, the developers has implemented modules for the UE and eNB. As seen in Figure 4.3, the UE compound module consists of 6 compound modules, namely NIC, networkLayer, TCP, UDP, as well tcpApp[numTcpApps] and udpApp[NumUdpApps], where multiple instances of applications to be run on top of TCP and UDP. All modules except the NIC module are inherited from the INET library.

The NIC module simulate a Network Interface Card and implements the LTE-A stack. The LTE-A stack consists of, from top to bottom, Packet Data Convergence Protocol (PCDP)-Radio Resource Control (RRC), Radio Link Control (RLC), MAC and physical layer. The PCDP-RRC receives packets from the network layer of the UE and first performs Robust Header Compression (ROHC), as well as some connection identifier management. The packets are encapsulated as a PCDP Protocol Data Unit (PDU) and forwarded downstream to the RLC. If a packet is received upstream from the RLC, the PCDP-RRC module is responsible for decapsulation and decompression of header, and finally forwarding the resulting PCDP PDU upstream to the network layer.

The NIC module is used in both UE and eNB, but the UE version includes a Feedback Generator module to create channel feedback which is managed by the physical layer.

The networkLayer modules simulates the network layer and supports IPv4, Address Resolution Protocol (ARP), error handling, ICMP, and Internet Group Management Protocol (IGMP). In the UE it connects the LTE-A stack with the TCP/UDP layer.

The TCP/UDP applications represent the end of an connection, where the TCP and UDP modules, which implement the respective transport layer protocol, connects the application to the LTE-A stack.

4.5 OpenFlow OMNeT Suite

The OpenFlow extension of OMNeT was developed to cover the lack of options for performance evaluation of OpenFlow architecture. Especially the lack of testing OpenFlow deployment on a larger scale, where testbeds typically are limited to smaller topologies. By using simulations tools, a controllers scalability can be thoroughly evaluated as they are not limited to the same degree by hardware or distance. Simulation tools also give the advantage of quickly adapting any specification changes that are done to the tested technology, as well as the option to validate components and functions before distribution.

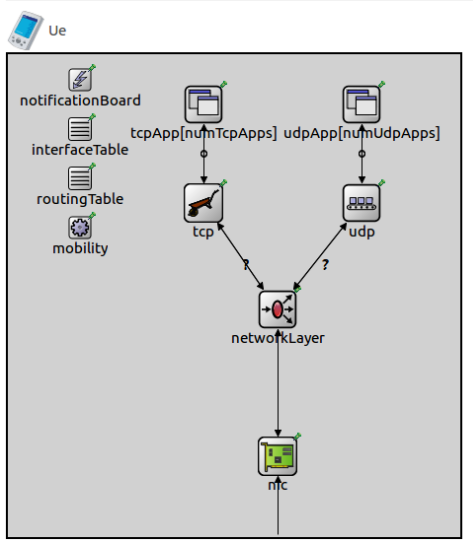


Figure 4.3: A screenshot of the UE module in OMNeT++'s graphical interface.

4.5.1 Model Overview

According to the developers of OpenFlowOMNeTSuite [KJ13], they have used version 1.3 of the OpenFlow switch specification as it is base when modelling the OpenFlow protocol model. In the implementation of OpenFlow, the developers use elements from the INET framework as its foundation. The OpenFlow OMNeT Suite includes an implementation of the OpenFlow switch and the OpenFlow controller as nodes, the most critical messages used for communication between the two nodes, and openflow.h, which is a header file that is used to model the protocol and its messages. Other notable inclusions in the framework is a spanning tree module and a controller placement module which is implemented as utility modules, as well as some controller applications.

4.5.2 OpenFlow Messages

The framework has implemented the essential messages of the OpenFlow protocol and more [GZGTG16]. To model the messages as close to reality as possible they all include the OpenFlow message header definition and corresponding C++ structs from the OFP_Header, which they are subclassed from. The suite supports so far Hello, Feature-Request and Feature-Reply, Packet-In, Packet-Out, Flow-Mod, and Port-Mod. See Table 4.1 for a description of each message.

Table 4.1: The OpenFlow messages supported by OpenFlow OMNeT Suite

OpenFlow Message	Description
Hello	Used for version negotiation, and can be sent by either controller or switch
Features-Request	Used in the secure channel setup between the controller and the switch, where it is sent by the controller to the switch to get the capabilities of the switch.
Features-Reply	Used in the secure channel setup between the controller and the switch, where it is sent by the switch to the controller to give the controller its capability information
Packet-In	Sent by the switch to the controller when a received packet does not match any flow table rules in switch
Packet-Out	Sent to the switch by the controller to change the state of an OpenFlow port
Flow-Mod	Sent to the switch by the controller when the controller wants to change, delete or add a flow entry in the flow tables in the switch
Port-Mod	Sent to the switch by the controller to change the state of an OpenFlow port

4.5.3 OpenFlow Switch

The OpenFlow enabled switch is implemented as a compound module where the modules can be grouped into three, as illustrated in Figure 4.4.

The Control Plane consists of three modules that handle all communication with the control plane, meaning the SDN controller. The networkLayer module and the TCP module are from the INET framework. The Switch Logic group consists of one module called OF_Switch, which represent the functionality of the OpenFlow enabled switch. This module keeps the flow tables, timeouts, thresholds, creates and handles OpenFlow messages, and is responsible for connection setup with the controller. The Data Plane Interface consists of one interface modules which forward packets between the OF_Switch module and the rest of the network.

4.5.4 SDN Controller

The SDN Controller is implemented as a compound module where the modules can be grouped into three, as illustrated in Figure 4.5.

The Control Plane Interface group functions similarly to its equivalent in the OpenFlow Switch module. The Controller Logic group consists of the OF_Controller

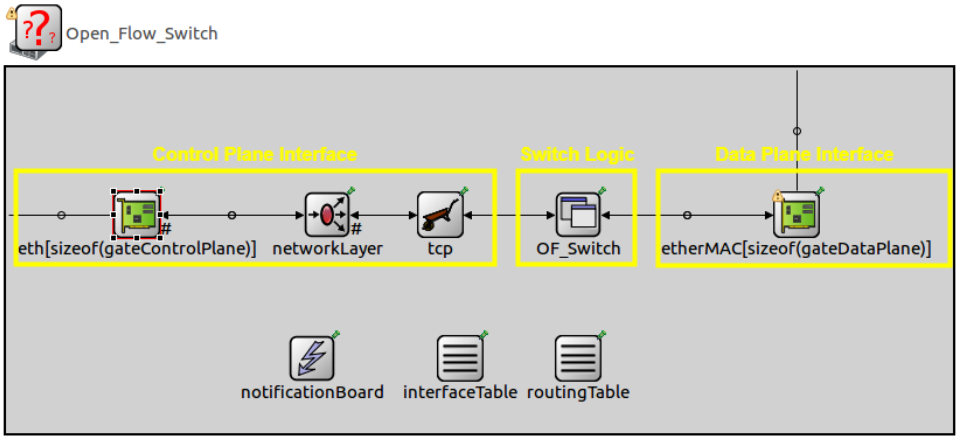


Figure 4.4: A screenshot of the OpenFlow Switch module in OMNeT++'s graphical interface.

module, which represents the main logic of the SDN controller. It is responsible for connection setup and maintenance, in addition to facilitating and communicating with modules in the Controller Application Slots group. The Controller Application Slots group consist of the controllerApps module and the tcpControllerApps module. These two modules allow the simulator to use several controller applications to run on top of the OF_Controller to modify the behaviour of the controller by, e.g. changing routing logic, handling of Packet-In messages, and match fields and actions to be included in Flow-Mod messages.

4.5.5 Controller Applications and Utility Modules

The developers of OpenFlowOMNeTSuite has included some controller applications and utility modules for users to test out their suite. Some noticeable controller applications are the Hub module and the LearningSwitch module.

The Hub module gives the SDN controller one of its most straightforward behaviours, where the OpenFlow enabled switch is instructed to flood all received packets through every port except the ingress port.

The Learning switch module is an advancement of the Hub module's behaviour. It starts as the hub by flooding packets with a destination MAC address it has not seen before, but maps the source MAC address of the incoming packet with the ingress port to create flow table entries and directly forward packets instead of flooding.

OpenFlowOMNeTSuite also includes controller applications like ARPResponder,

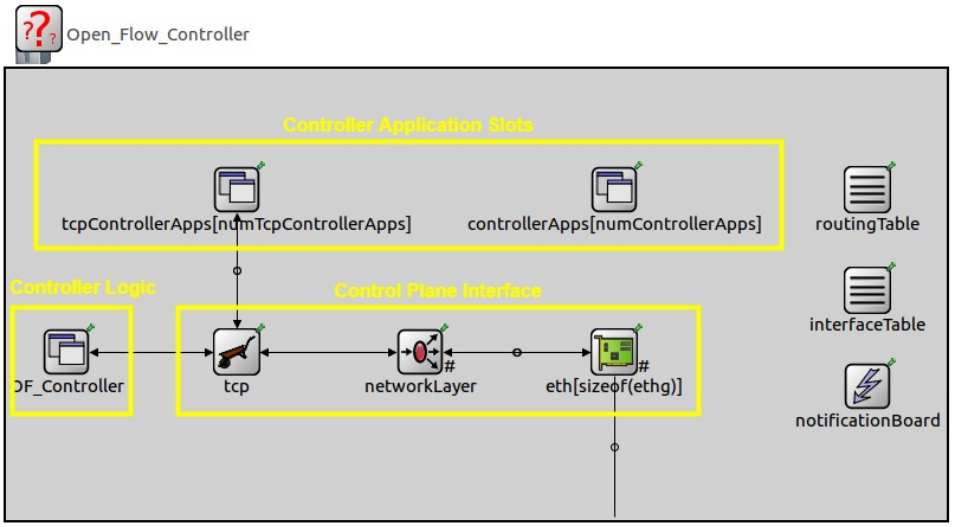


Figure 4.5: A screenshot of the SDN Controller module in OMNeT++'s graphical interface.

LLDPAgent, LLDP Forwarding, host applications like Ping App Random, TCP Traffic Generator, and TCP Traffic Sink, and distributed SDN controllers like HyperFlow and Kandoo, but these are out of scope for this thesis and will therefore not be mentioned further.

For utility, the developers have implemented StaticSpanningTree module for supporting network topologies containing loops and OpenFlowGraphAnalyzer to collect graph statistics from OpenFlow enabled simulation networks.

4.6 Integration of simuLTE and OpenFlowOMNeTSuite

The simuLTE framework and OpenFlowOMNeTSuite have been developed without the thought of interoperation between the two. When developing a new extension framework to OMNeT++ and use INET modules in the framework, the version of both OMNeT++ and INET is a factor. As both simuLTE and OpenFlowOMNeTSuite use INET modules in their framework, it was important to use download versions of each framework that used a compatible version of INET and OMNeT++ for them to work together.

OpenFlowOMNeTSuite has one working and published version which is developed using OMNeT++ 4.6 and INET 2.5. As they only have one working build, it was important that SimuLTE had compatible versions.

SimuLTE has one completed version v1.0.0 which is compatible with OMNeT++ 5.0 and INET 3.4.0. It also has an earlier version called v0.9.1 which is published and compatible with OMNeT++ 4.6 and INET 2.3.

The working build was installed in Oracle VM Virtualbox 5.2.20 [vir] using Ubuntu 16.04 with OMNeT++ 4.6, INET 2.5, simuLTE v0.9.1 and OpenFlowOMNeTSuite. Virtualbox was installed on a Dell OptiPlex 7060 with processor Intel(R) Core(TM) i7-8700 and 32.0 GB installed RAM running Windows 10 Pro.

When installing OMNeT++ 4.6, the installation guide at [omna] was followed for Linux and Ubuntu systems using the graphical installation. When installing INET 2.5, the manual installation of the installation guide at [ine] was followed. When installing simuLTE the installation guide at [sim] was followed. For OpenFlowOMNeTSuite, there exists no installation guide. Therefore it was installed using the same steps as described in the simuLTE installation guide. At the Github page of OpenFlowOMNeTSuite [opea] there were instructions on replacing three INET files with modified INET files made by the developers of OpenFlowOMNeTSuite, which also was followed.

The combination of simuLTE and OpenFlowOMNeTSuite packages in OMNeT++ has to the extent of our knowledge never been done before, and therefore, this platform development represents a contribution to the state-of-art. By integrating the two extension frameworks, we were able to create a simulation environment that facilitates testing of close-to 5G scenarios. The modularity of OMNeT++ enables the creation of various 5G-like topologies which can be used to experiment with SDN controller applications in a single or multiple controller architecture. OMNeT++ provides tools for collecting statistics during simulation, which can be utilized to evaluate the performance of, e.g. detection algorithms and mitigation methods.

Chapter 5
SDN-based DDoS Defense Experiment and Analysis

This chapter presents an SDN-based DDoS Defense experiment that tests a self-developed SDN controller application. The controller application is designed to detect and mitigate SYN flood attacks in a close-to 5G environment. The motivation for the experiment is to verify the theoretical assumptions of SDN's security capabilities by providing a Proof of Concept (POC).

5.1 Experiment

The experiment is performed by simulation using the OMNeT++ Discrete Event Simulator. The following sections describes the topology, hardware, simulation setup, attack scenario and SDN controller application behavior.

5.1.1 Topology

The topology used in the experiment is depicted in Figure 5.1 and is inspired by an SDN-based 5G oriented network architecture proposed in [Zol15], which is depicted by Figure 2.1.2. The topology consists of 8 node types in total. The ue, ueMal, eNodeB, and pgw nodes represent the LTE-A part of the network and the open_flow_switch and the open_flow_controller nodes represent the SDN part of the network. Together these two parts simulate a simple, close-to 5G network topology. Originally, the topology was designed to have the open_flow_switch directly connected to the eNodeB, but the simulation framework would not allow it. More on this in Section 6.2. Therefore the pgw and router were introduced to interconnected the two parts of the network. The server represents a generic Internet server that answers any requests from connecting hosts. All traffic flows go between the server and the hosts connected to the eNodeB. The reason for placing the open_flow_switch as the only connected node to the server is to make all packet flows be processed by the open_flow_switch and the open_flow_controller.

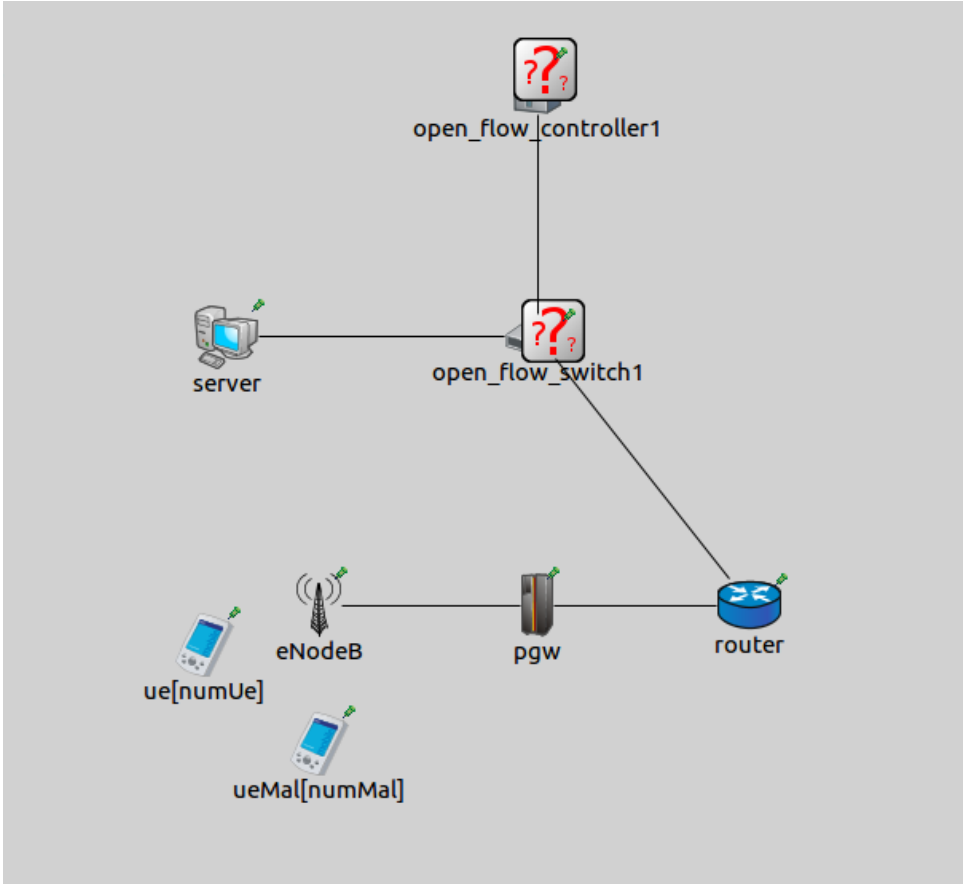


Figure 5.1: Topology of the SDN-based DDoS Defense experiment. The ue and ueMal communicate with the server, while the open_flow_switch forwards all traffic and communicates only with the open_flow_controller.

5.1.2 Hardware

For specification on the hardware used during simulation, see Section 4.6.

5.1.3 Simulation setup

The general parameters used in the experiment are specified in Table 5.1. The simulation runtime was set to 50 seconds to limit the real runtime of the experiment and still be able to test a fully active SYN flood attack against the controller application. The maximum number of UEs connected was based on an approximate calculation done by the author of [Sau], who concluded that the number of simultaneously connected and active users per eNB could be from 60 to 100 devices. In the simulation, it assumed that the majority of the connected devices to the eNB is corrupted, therefore 20 benign UEs and 40 malicious UEs was used. Additionally, the number was limited by the simulation environment as it would crash if more than 80 UE were active simultaneously. The packet rate of both the benign UEs and malicious UEs was not set to necessarily mimic real-life values, but to have a proportionate relationship where the malicious packet rate would be drastically higher than the benign packet rate. The default value set in OpenFlowOMNeTSuite is used for the SYN-RECEIVED timer and SYN-ACK Retransmissions, which is set according to [SKK+97]. If an ACK is received before the timer runs out, the connection is completed. If not, the connection attempt is aborted and the reserved resources for the connection are released. The default value set in OpenFlowOMNeTSuite for the idle timeout of flow entries is used. The detection threshold is set according to the used packet rate generated by the UEs. Depending on the generated events and included nodes, the actual simulation time of a conducted experiment range from 15 to 60 minutes.

Table 5.2 and 5.3 provide some of the parameters used for respectively the UEs and eNB.

5.1.4 Generating legitimate traffic

The experiment includes 20 UE nodes to represent legitimate users that generate legitimate traffic to create a more realistic scenario. Each UE node creates a new TCP connection at a time chosen uniformly random within every ten-second window, i.e. 1-10s, 11-20s, 21-30s, 31-40s, and 41-50s with the server and transmits data with a randomly chosen byte size in a uniform distribution between 100 bytes to 2000 bytes. The data transmission start time is also chosen randomly in uniform distribution.

Table 5.1: The basis parameters used during simulation

Parameter	Value
Simulation runtime	50s
Number of UEs	20
Packet rate pr UE	0.1 data packets per second (pps)
Number of malicious UEs	40
Packet rate pr malicious UE	3.5 pps
SYN-RECEIVED Timer	75s
SYN-ACK Retransmissions	Loop: 3s - 6s - 12s
Flow entry timeout	10s
Detection threshold	0.5 pps
TCP Algorithm	TCP Reno

Table 5.2: The parameters used for both benign and malicious UEs during simulation

Parameter	Value
Mobility type	Stationary
Transmission direction	Omni-directional
Transmission Power	26dBm
Delayed Acks Enabled	false
SACK Support	false
Multiple MIMO	true
Queue size	1MiB
Max Bytes per time-to-live	1KiB

Table 5.3: The parameters used for the eNB during simulation

Parameter	Value
Resource block allocation	Distributed
Scheduling Discipline	MAXCI
Transmission direction	Omni-directional
Transmission Power	26dBm
Queue size	2MiB
Max Bytes per time-to-live	3000KiB
TCP App	TCPSessionApp

5.1.5 Attack scenario

The TCP SYN Flood attack is performed by 40 ueMal nodes which represent the attackers. The attack nodes have identical behaviour to imitate bots in a botnet. The attack is designed as a DDoS attack targeted at the server. The ueMal nodes transmits TCP SYN messages to the server without replying to the TCP SYN-ACK response in order to create half-open connection and exhaust the server's resources. Each ueMal node transmits a "wave" of 10 SYN messages every 3 seconds for a total duration of 29 seconds. The first wave is transmitted after two seconds in simulation time. This equates to 10 waves in a flood attack, where a wave lasts for 1 second.

5.1.6 SDN Controller application

To evaluate the security capabilities of SDN, an SDN DDoS Defence controller application has been developed for this experiment. The application determines the behaviour of the open_flow_controller by specifying the routing logic, how to handle Packet_In messages from the open_flow_switch, and how to construct the Flow_Mod messages and Packet_Out messages. It is implemented as an altered version of the OpenFlow Controller application "LearningSwitch" which is included in the OpenFlowOMNeTSuite package. The application functions in two stages, with a detection phase and a mitigation phase. In the detection phase, the detection method is used, and a mitigation method is used in the mitigation phase.

Detection method

The detection method is a simple version of a connection rate-based detection method, where detection is based on the assumption that malicious hosts have a higher connection attempt rate than benign hosts. To utilize the shared characteristics of the attacker hosts the open_flow_controller needs to keep an overview of the connection attempt rate of all active hosts in the network.

Instead of the controller keeping a table with the state of all SYN attempts for each IP address in the network, it separates normal packets from TCP SYN packets. Flow entries for normal flows are installed with match fields for Ethernet type, incoming port, source MAC address, destination MAC address and a TCP SYN flag, while special flow entries are made for any packet that has the TCP SYN flag raised. These SYN flow entries include the additional match fields of the ACK flag, IP source address and IP destination address. The ACK flag is included to separate SYN messages from SYN-ACK messages. The IP source and destination address are included to identify the hosts that send an abnormally high number of TCP SYN messages to a specific target.

In each flow entry for SYN packets the controller includes a threshold value that the switch is instructed to compare with its counter value for each SYN flow entry. The threshold value is a number set dependent on the flow entry idle timeout and hard timeout. All flows have an idle timeout of 10 seconds as the default value and do not utilize the hard timeout as it was not implemented in the used version OpenFlow OMNeT Suite. The idle timeout is reset each time a new packet matches a flow entries as long as the timeout has not expired. That means that the threshold potentially can be surpassed even with a low packet rate if the idle timeout is reset right before it expires every time. This low packet rate is calculated using Equation (5.1) and referred to as the minimum packet rate threshold from now on. The duration the minimum packet rate has to continue to surpass the threshold is the set threshold value times the flow entry idle time. The maximum packet rate threshold is calculated by Equation (5.2).

$$MinPpsThreshold = \frac{thresholdValue}{thresholdvalue * idleTimeout} = \frac{1}{idleTimeout} \tag{5.1}$$

$$MaxPpsThreshold = \frac{thresholdValue}{idleTimeout} \tag{5.2}$$

The detection method does not distinguish between completed connection attempt and failed connection attempts mainly to simplify the solution, but also because of the assumption that the malicious connection attempt rate is much higher than the legitimate connection attempt rate, and therefore the completed connection attempts can be viewed as negligible in this experiment.

Mitigation method

Once the threshold of a flow entry is surpassed, the open_flow_switch is instructed to transmit a Packet-In that contains the triggering packet and unique value for the field packet_in_reason to the open_flow_controller. The controller application replies to the special Packet-In with a Flow-Mod message instructing the switch to change the action of the respective flow entry to "drop", as well as a Packet-Out message instructing the switch to drop the triggering packet. Any SYN packet that matches this flow entry is dropped once it enters the switch, mitigating the exhausting attack on the server.

This method can be bypassed easily by an attacker that uses spoofed IP addresses, as the detection method relies heavily on packet IP address field, but this is beyond the scope of the experiment. The experiment is designed to provide a Proof of Concept (POC), knowing that the application only functions against a specific attack.

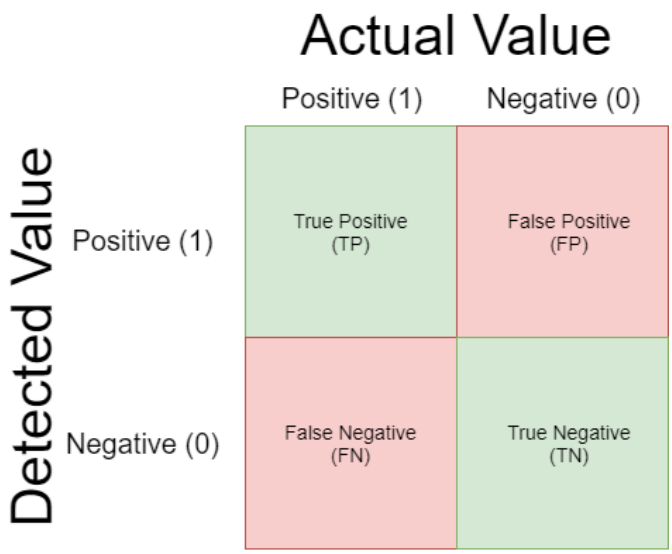


Figure 5.2: Contingency table or confusion matrix

The POC implies that more intricate and improved detection and mitigation methods can be implemented in SDN, using other logic, analysis, match fields and actions than what this method applies.

5.2 Evaluation metric for detection algorithm

The application behaves as a binary classifier by attempting to correctly classify all packets into two groups, either malicious or benign. By evaluating how successfully it performs the classification it can give validity to the POC. To evaluate the performance of the implemented SDN controller application, it is necessary to look at quantifiable data and statistics. A well-known and proven method of evaluating binary classifiers is sensitivity and specificity analysis. Sensitivity, or True Positive Rate (TPR), is a measure of how good the application is to detect malicious packets by providing the probability of detection, and is calculated by Equation (5.3). Specificity, or True Negative Rate (TNR), is the measure of how good the application is to identify legitimate traffic. Equation (5.4) calculates the probability of not dropping legitimate traffic. To calculate the sensitivity and specificity, it is first needed to build a 2x2 contingency table, consisting of True Positives (TPs), True Negatives (TNs), False Positives (FPs), and False Negatives (FNs) as depicted in Figure 5.2.

The OMNeT++ framework allows the creation and transmission of signals during a simulation for statistics purposes, which are stored and can be viewed after. Four

signals have been implemented in the code to register respectively TPs, TNs, FPs, and FNs. TPs are defined as any packet that has been correctly registered as malicious by the switch and therefore dropped. TNs are defined as any packet that has been correctly registered as benign by the switch, and therefore forwarded through a port. FPs are defined as any packet that has been incorrectly registered as malicious by the switch, and are therefore dropped instead of forwarded. FNs are defined as any packet that is incorrectly registered as benign by the switch, and are therefore forwarded instead of dropped.

$$TPR = \frac{TP}{TP + FN} \tag{5.3}$$

$$TNR = \frac{TN}{TN + FP} \tag{5.4}$$

5.3 Results and Analysis

The initial results were produced using the parameters showed in Table 5.1.3, and generated the confusion matrix displayed in Figure 5.3 with the Sensitivity or TPR calculated using Equation (5.3) and Specificity or TNR calculated using Equation (5.4). A Sensitivity of 96% is a promising initial result and indicates a good performance in this environment, where it is suspected that the undetected 4% represents the first malicious packets of the attack that go undetected until the detection packet rate threshold is reached. A specificity performance of 100% can be explained by the relatively low packet rates of the benign UEs, which averages 0.1 pps and consequently do not surpass the maximum packet rate detection threshold of 0.5 pps or the minimum packet rate threshold as the simulation at 50 seconds, before the threshold can potentially be surpassed.

The performance of the implemented detection method may depend on the parameters used to create the simulation environment. To further explore how the SDN DDoS Defense application performs against SYN flood attacks and understands the initial results, five additional experiments have been conducted using the parameters in Table 5.1 as the basis and varying values of the following parameters individually;

- Number of malicious UEs
- Number of benign UEs
- Flow entry timeout
- Malicious packet rate

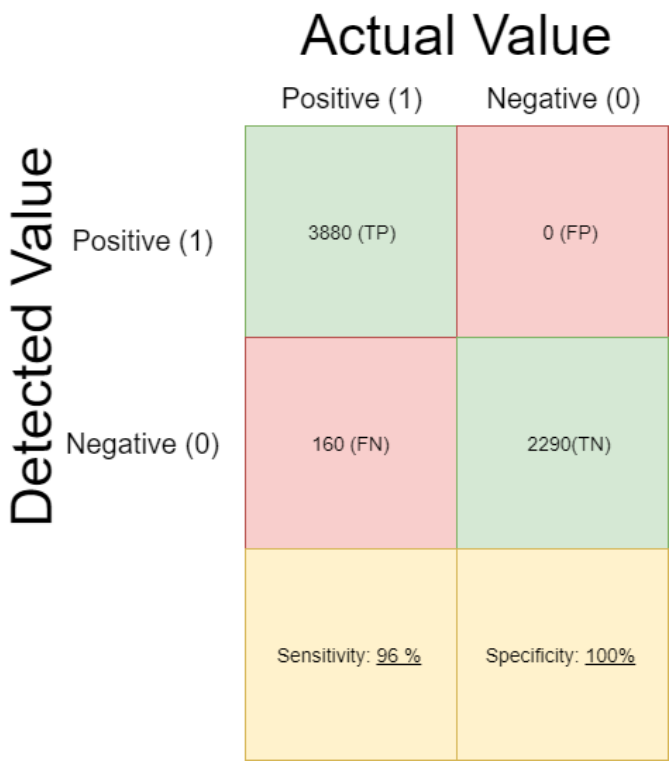


Figure 5.3: Confusion matrix of initial experiment

– Detection threshold

The results of these experiments are presented in the following sections.

5.3.1 Performance when varying number of malicious nodes

In this experiment, the initial parameters presented in Table 5.1 used during simulation, except for the number of malicious UEs. The experiment is repeated 8 times with the number of attack nodes parameter starting at zero and increasing by 10 nodes for each repetition until it reaches 70 attack nodes in the network. The reason for testing the minimum parameter's value of 0 is to evaluate the specificity performance of the detection application in a network with only legitimate traffic. The maximum parameter value of 70 malicious nodes is chosen to evaluate how the application performed in an environment with an overwhelming amount of generated malicious traffic compared to legitimate traffic. The increased value of 10 malicious nodes for each repetition allows the evaluation of how the number of malicious

impacts the performance indicators as it grows. The results of the experiments are presented in Figure 5.4.

The results show that the performance indicators sensitivity and specificity remain the same for any number of malicious UEs using the parameters in Table 5.1. The specificity value is at ∼96%, and the specificity value is at 100%. The OpenFlow enabled switch creates a flow table entry for each malicious UE which can affect the performance of the switch as real-life switches do not have unlimited memory for flow table entries. In the simulation environment, the maximum number of malicious nodes used in these experiments does not surpass the limit of flow entries the switch can have, and therefore the sensitivity and specificity are not affected by it.

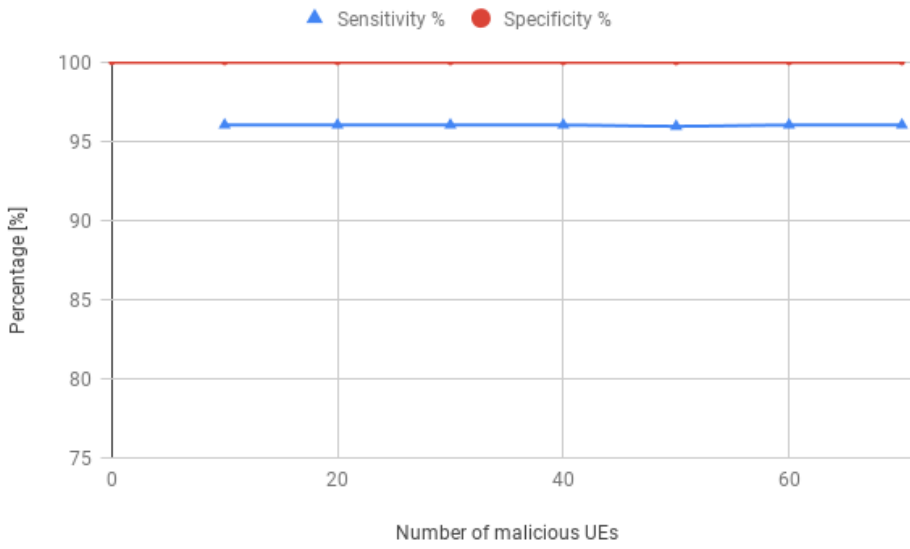


Figure 5.4: The performance of the SDN application in regards to sensitivity and specificity when varying the number of malicious UE's used in the simulation

5.3.2 Performance when varying number of benign nodes

To evaluate the impact of the relationship between legitimate traffic and malicious traffic on the detection application, an experiment is repeated eight times where the number of benign UE nodes increase with 10 nodes per repetition in the range of 0 to 70 nodes. Because an eNB can have a maximum of 100 active UEs simultaneously connected, the number of malicious nodes is 20 in all of the experiment repetitions. The results presented in Figure 5.5 indicate a continuous decline in the detection

application's sensitivity as the number of benign UEs increases. Notice that the controller achieves a specificity value even with 0 benign UEs in the experiment to generate legitimate traffic. This is caused by the legitimate SYN-ACK messages generated by the server as a response to the malicious SYN messages received from the attack nodes.

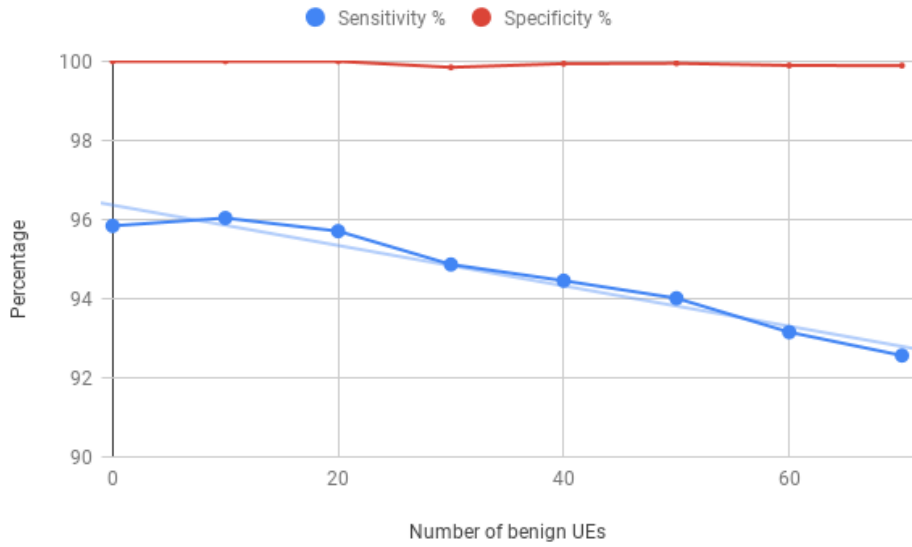


Figure 5.5: The performance of the SDN application in regards to sensitivity and specificity when varying the number of benign UEs used in the simulation

5.3.3 Performance when varying flow entry timeout

The SDN DDoS Defense application relies on flow entries to mitigate attack traffic from nodes marked as malicious and therefore the impact of idle timeout of flow entries should be evaluated as the timeout dictates how long a flow entry remains active. The results of ten simulations where with the flow entry idle timeout set to 1, 2, 4, 6, 8, 10, 20, 30, 40, and 50 seconds respectively is presented as a graph in Figure 5.6.

When the flow entry timeout is set to 2 seconds or lower, the results show a sensitivity of ∼64%, while the sensitivity value is ∼96% for timeout values higher than 4 seconds. The results can be explained by the frequency of packet transmissions by the malicious nodes being set to every 2 seconds in the simulation. The 2 seconds waiting time before a new wave of SYN packets allows the flow entry timeout to

run out and causes it to be deleted from the flow tables of the OpenFlow enabled switch, causing the malicious traffic to reach the server again. As each malicious node transmits ten SYN packets in each wave, the first four malicious packets of the wave are judged as legitimate by the switch as the detection threshold is not surpassed yet. The fifth packet surpasses the detection threshold and causes itself and the remaining five packets to be marked as malicious and be dropped at the switch.

The SDN DDoS Defense application performance in regards to specificity decreases from ∼100% at flow entry timeout values of 10 seconds and lower, to ∼98% at timeout values of 20 seconds and higher.

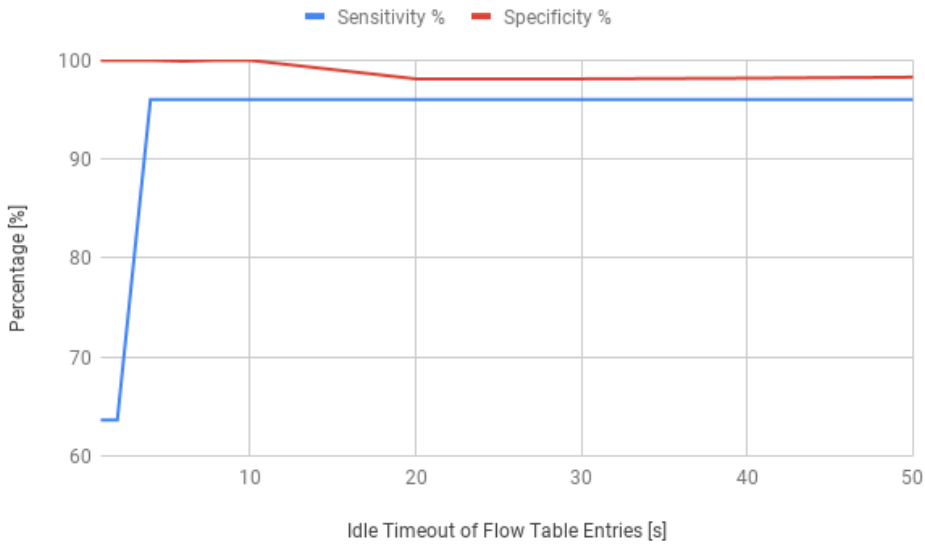


Figure 5.6: The performance of the SDN application in regards to sensitivity and specificity when varying the idle timeout of flow entries

5.3.4 Performance when varying detection threshold

The impact of the detection threshold value is evaluated by conducting an experiment that is repeated seven times with the maximum packet rate threshold starting at 0.1 pps, then from 0.5 pps to 3 pps, where the parameter is increased by 0.5 pps for each repetition.

The results presented as a chart in Figure 5.7 show a linear decrease of sensitivity as the threshold is increased. Naturally, the number of false negatives increases as

the threshold is lowered, because more packets are required to surpass the threshold, allowing a larger number of malicious packets reaching the server before the attack node is detected and its traffic is dropped at the OpenFlow enabled switch. With a maximum packet rate threshold of 0.1 pps the specificity is at ∼94%, but stays at 100% from 0.5 pps and higher. A low threshold value makes the SDN controller mistakenly mark legitimate nodes with a spike SYN message rate higher than the threshold value as malicious.

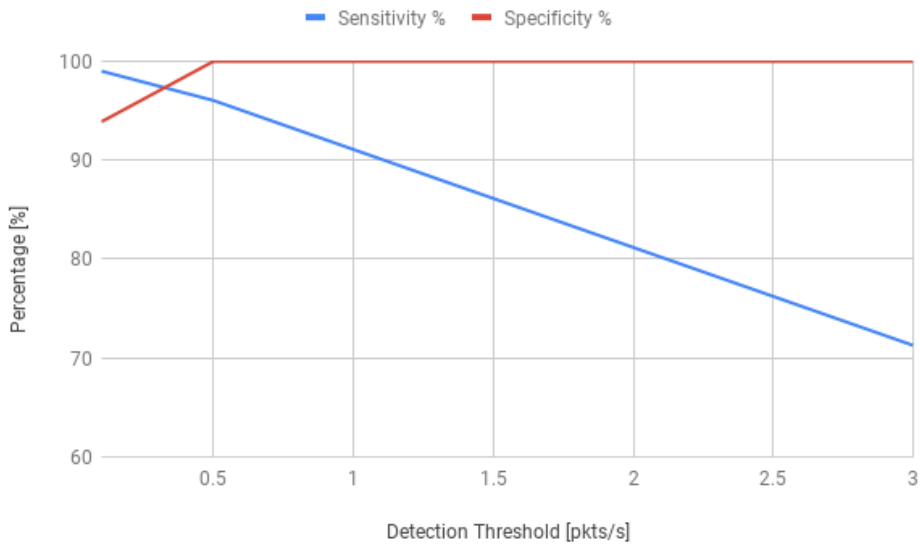


Figure 5.7: The performance of the SDN application in regards to sensitivity and specificity when varying the packet rate threshold for detection

5.3.5 Performance when varying malicious packet rate

The last two experiments to be conducted is to evaluate the effect of the malicious packet rate on the detection method. Both experiments are repeated four times with the attack rate per wave ranging from 5 packets to 20 packets, with an increase of 5 packets per repetition. The first experiment has the detection threshold value set to 5 packets, and the second experiment has the detection threshold value set to 30 packets.

The results of the first experiment, presented in Figure 5.8, shows an increase in the detection methods specificity as the packet rate per wave increases. A detection threshold value of 5 packets gives a maximum packet rate threshold of 0.5 pps,

making the first four SYN packets of the attack to be registered as false negatives, while the following packets are registered as true positives. When the attack rate per wave is increased, the number of true positives increases while the number of false negatives remains the same. The sensitivity equation displayed in Equation (5.3) shows that if the true positives increases, the the sensitivity increases as well. Note that sensitivity does not reach 100% as long as the maximum packet rate threshold is over 1 pps.

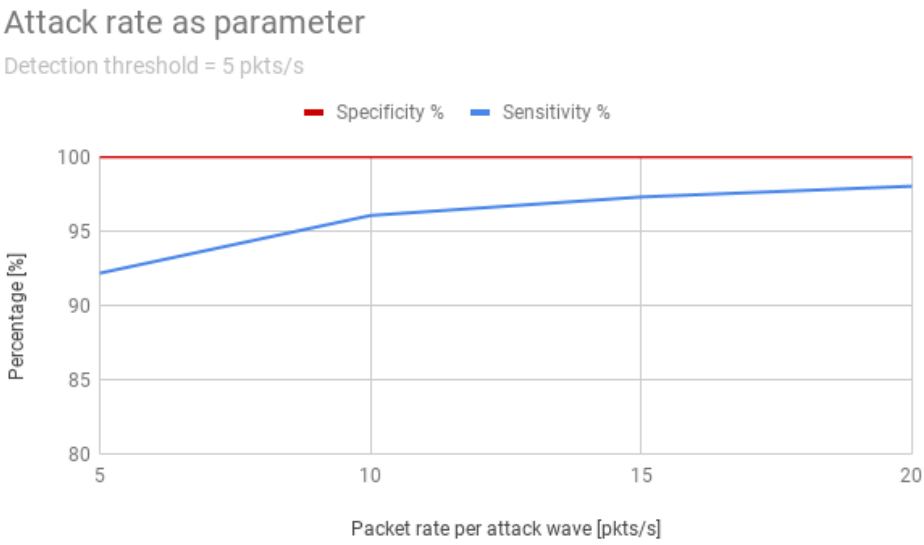


Figure 5.8: The performance of the SDN application in regards to sensitivity and specificity with a packet rate threshold of 5pkts/sec for detection and varying the malicious packet rate

The results of the second experiment, presented in Figure 5.9, reaffirm the results of the first experiment by showing a similar growth in sensitivity as the packet rate per attack wave increases.

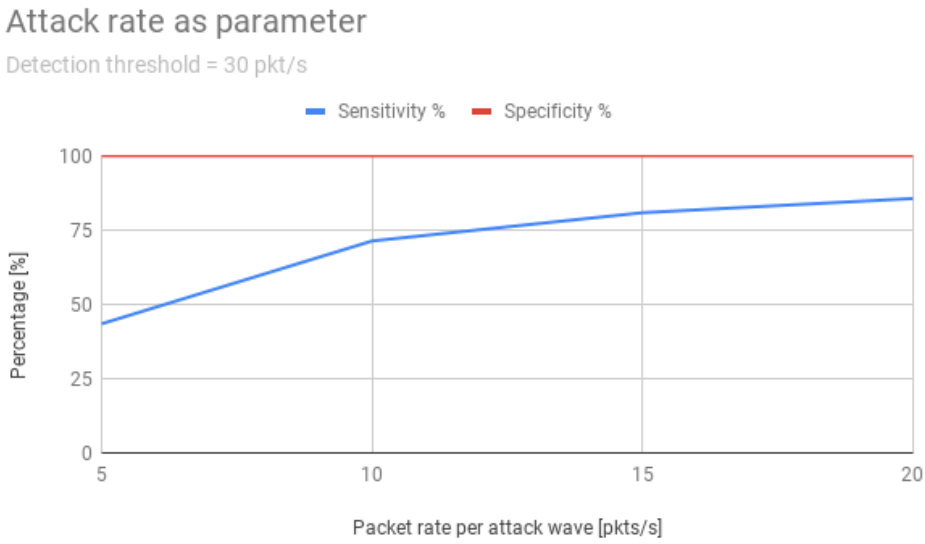


Figure 5.9: The performance of the SDN application in regards to sensitivity and specificity with a packet rate threshold of 30pkts/sec for detection and varying the malicious packet rate

Chapter 6
Discussion

In this thesis, an SDN DDoS Defense controller application is implemented and tested in a simulation environment against an SYN flooding attack. A sensitivity and specificity analysis evaluate the performance of the application, and validated by repeated experiments with different values for parameters such as the number of malicious nodes, the number of benign nodes, flow entry timeout, malicious packet rate, and detection threshold. The results demonstrate the feasibility of SDN as a crucial security component in a close-to 5G environment, as the application performs well and achieves similar sensitivity and specificity throughout the experiments, thereby providing reliable results. This chapter discusses the obtained results and the limitations of the experiments, as well as the aspects of using SDN as a security component in 5G.

6.1 Experiment Results

6.1.1 Number of malicious nodes

The number of malicious nodes does not impact the performance of the application, for reasons explained in Section 5.3.1, which implies that the controller application could scale well in terms of the number of attack nodes, given that the OpenFlow-enabled switch has sufficient memory for flow tables.

6.1.2 Number of benign nodes

The sensitivity performance of the application seems to suffer when the number of benign nodes and by extension, network traffic increases while the malicious traffic remains the same. The reason for the decrease in performance can be increased transmission delay in the network caused by the increased network traffic, which causes a greater delay between malicious SYN packets and thereby a lower attack rate. As the results of Section 5.3.5 shows, the application performs worse when the attack rate is low.

6.1.3 Flow entry timeout

The choice of flow entry timeout is of great importance for the performance of the detection application as a low timeout can be exploited by attackers who use a large number of attack nodes with the time between attack transmissions being longer than the flow entry timeout. A high timeout value causes nodes marked as malicious to be blocked for a longer duration, which is unfortunate for benign nodes wrongly marked as malicious.

6.1.4 Detection threshold

The choice of the detection threshold naturally has an impact on the performance of the SDN DDoS Defense application and causes a trade-off between sensitivity and specificity. A low detection threshold increases the sensitivity as a lower packet rate is required to surpass the detection threshold, causing more flows to be marked as malicious, which increases the number of false positives and therefore effects the specificity negatively. On the other hand, a high detection threshold has the opposite effect where a higher packet rate is required to surpass the detection threshold, causing less false positives and higher specificity, but more false negatives and subsequently a lower sensitivity.

6.1.5 Malicious packet rate

The results indicate that the detection method is not as effective against low attack rates, which attackers may be able to exploit by using a low attack rate per attack node, but this demands a higher number of attack nodes to generate enough traffic to exhaust the targets' resources.

6.2 Limitations

As mention in Section 5.1.1, the simuLTE and OpenFlowOMNeTSuite modules was not compatible in OMNeT++ and could not be connected directly. This caused an alteration of the planned simulation architecture. The OpenFlow-enabled switch was connected directly to the eNB in the original network design, similar to the SDN-based 5G oriented network architecture proposed by the authors of [Zol15] and depicted in Figure 2.3. As this was not possible, a traditional network router from the INET library was introduced to function as an intermediate between the switch and eNB module. However, an eNB cannot talk directly to a traditional network router. Therefore the PDN-GW module from simuLTE was included to represent both the S-GW and PDN-GW. The forced change alters the network architecture from being a 5G architecture to a close-to 5G architecture. SDN is located more towards the core network, making the DDoS attack propagate further in the network and cause more harm, as the detection and mitigation happen later than it would in the original

network design. The router especially represents a weakness in the architecture in the case of malicious attacks, since it is not controlled or programmable by the SDN controller.

6.3 SDN as a security component in 5G

Given the limitations of the conducted experiments, it is not possible to conclude that SDN would successfully function as an effective security component in 5G deployments, based on the results of the experiments alone. The results do however show potential for security application in SDN, as the SDN DDoS Defense controller application implemented in this thesis is far from perfect, but still achieve high sensitivity and specificity performance, thereby providing a Proof of Concept for the use of SDN as a security component. As SDN is to be a key technology in 5G, utilizing its security capabilities could become crucial for the success of the new mobile communications system. Security applications with more suitable and efficient detection and mitigation methods for 5G can be implemented for higher performance and to defend against more DoS attack types than a distributed SYN flood attack.

Chapter 7
Conclusion and Further Work

7.1 Conclusion

In this thesis work, we presented a background study of the architecture and key technologies of 4G/LTE-A and 5G mobile telecommunication network. The security threats of 5G and existing countermeasures are explained, with a focus on DDoS attacks. The security attributes of one key technology of 5G, SDN, and the possibility of utilizing the attributes to detect and mitigate DDoS attacks are discussed. An experimental analysis of SDN as a security component in a close-to 5G environment is given.

A novel integration of simulation frameworks simuLTE and OpenFlowOMNeT-Suite in OMNeT++ has been done to implement and test a SDN DDoS Defense controller application against SYN flooding attacks in a close-to 5G environment. The control application takes advantage of the attributes of OpenFlow's flow tables in its connection rate-based detection method and mitigation method. In particular, it utilizes the flow entry hit counters to detect malicious flows and mitigate attacks by altering the action of flow entries to drop through Flow-Mod messages. Simulation experiments with different parameters have been conducted to evaluate and validate the performance of the security application using sensitivity and specificity analysis.

The conducted experiments achieved consistent results for experiments with varied network environment parameters such as malicious packet rate, number of malicious nodes, and number of benign nodes, given a sensible configuration of the SDN DDoS Defense controller application. The performance of the application was not affected by the number of attack nodes, which indicates that the implemented security application could scale to the size of real-life attacks. The results obtained demonstrated that the SDN application effectively detects SYN flood attacks and thereby provide a Proof of Concept for SDN to be used as a security component.

7.2 Further Work

This section presents suggestions to further work that can be done in continuation of this work.

In the thesis work, a simple detection and mitigation were implemented as a controller application. Network attack methods are constantly evolving and utilizing the platform development that has been done in this thesis work to further develop and evaluate SDN security application in a close-to 5G environment would be a good idea.

In the present work, the performance is evaluated in a simplified, close-to 5G environment. Further platform development to bring the simulation environment closer to a 5G architecture should be done, as well as expanding the existing environment with more SDN controllers to test SDN collaborative attack mitigation schemes where the controllers notify each other of detected attacks through controller-to-controller (C-to-C) communication to mitigate attacks across networks.

References

[5gK] 5GPPP key performance indicators. https://5g-ppp.eu/kpis/. Accessed: 2019-06-02.

[AKL+18] Ijaz Ahmad, Tanesh Kumar, Madhusanka Liyanage, Jude Okwuibe, Mika Ylianttila, and Andrei Gurtov. Overview of 5g security challenges and solutions. *IEEE Communications Standards Magazine*, 2(1):36–43, 2018.

[All16] NGMN Alliance. 5g security recommendations package. 2016.

[And18] Thomas Andersen. Energy-efficient adaptive sensing in low power wide area networks. 2018.

[BSS17] Narmeen Bawany, Jawwad Shamsi, and Khaled Salah. Ddos attack detection and mitigation using sdn: Methods, practices, and solutions. *Arabian Journal for Science and Engineering*, 42, 02 2017.

[CGM+18] Krzysztof Cabaj, Marcin Gregorczyk, Wojciech Mazurczyk, Piotr Nowakowski, and Piotr Żórawski. Sdn-based mitigation of scanning attacks for the 5g internet of radio light system. In *Proceedings of the 13th International Conference on Availability, Reliability and Security*, ARES 2018, pages 49:1–49:10, New York, NY, USA, 2018. ACM.

[ea17] PAN Chengkang et al. End to end network slicing. *Wireless World Research Forum OUTLOOK*, 2017.

[Edd06] Wesley M Eddy. Defenses against tcp syn flooding attacks. *The Internet Protocol Journal*, 9(4):2–16, 2006.

[evo] 1g, 2g, 3g, 4g, 5g. https://its-wiki.no/images/c/c8/From_1G_to_5G_Simon.pdf. Accessed: 2019-06-02.

[GK19] D. Gligoroski and K. Kralevska. Expanded combinatorial designs as tool to model network slicing in 5g. *IEEE Access*, 7:54879–54887, 2019.

[GP] Thomer M Gil and Massimiliano Poletto. Multops: A data-structure for bandwidth attack detection.

[GZGTG16] Nicholas Gray, Thomas Zinner, Steffen Gebert, and Phuoc Tran-Gia. Simulation framework for distributed sdn-controller architectures in omnet++. In *International Conference on Mobile Networks and Management*, pages 3–18. Springer, 2016.

[HDK02] Haining Wang, Danlu Zhang, and Kang G. Shin. Detecting syn flooding attacks. In *Proceedings.Twenty-First Annual Joint Conference of the IEEE Computer and Communications Societies*, volume 3, pages 1530–1539, June 2002.

[HGJL15] Bo Han, Vijay Gopalakrishnan, Lusheng Ji, and Seungjoon Lee. Network function virtualization: Challenges and opportunities for innovations. *IEEE Communications Magazine*, 53(2):90–97, 2015.

[HPS+15] Yun Chao Hu, Milan Patel, Dario Sabella, Nurit Sprecher, and Valerie Young. Mobile edge computing—a key technology towards 5g. *ETSI white paper*, 11(11):1–16, 2015.

[ine] INET Framework for OMNeT++ manual. https://doc.omnetpp.org/inet/api-3.4.0/inet-manual-draft.pdf. Accessed: 2019-05-26.

[JKM+13] Sushant Jain, Alok Kumar, Subhasree Mandal, Joon Ong, Leon Poutievski, Arjun Singh, Subbaiah Venkata, Jim Wanderer, Junlan Zhou, Min Zhu, et al. B4: Experience with a globally-deployed software defined wan. In *ACM SIGCOMM Computer Communication Review*, volume 43, pages 3–14. ACM, 2013.

[KCG+10] Teemu Koponen, Martin Casado, Natasha Gude, Jeremy Stribling, Leon Poutievski, Min Zhu, Rajiv Ramanathan, Yuichiro Iwata, Hiroaki Inoue, Takayuki Hama, et al. Onix: A distributed control platform for large-scale production networks. In *OSDI*, volume 10, pages 1–6, 2010.

[KJ13] Dominik Klein and Michael Jarschel. An openflow extension for the omnet++ inet framework. In *Proceedings of the 6th International ICST Conference on Simulation Tools and Techniques*, SimuTools '13, pages 322–329, ICST, Brussels, Belgium, Belgium, 2013. ICST (Institute for Computer Sciences, Social-Informatics and Telecommunications Engineering).

[KPS03] Charlie Kaufman, Radia Perlman, and Bill Sommerfeld. Dos protection for udp-based protocols. In *Proceedings of the 10th ACM conference on Computer and communications security*, pages 2–7. ACM, 2003.

[LAA+18] Madhusanka Liyanage, Ijaz Ahmad, Ahmed Abro, Andrei Gurtov, and Mika Ylianttila. *A Comprehensive Guide to 5G Security*. 04 2018.

[LMK16] Wenjuan Li, Weizhi Meng, and Lam For Kwok. A survey on openflow-based software defined networks: Security challenges and countermeasures. *Journal of Network and Computer Applications*, 68:126 – 139, 2016.

[lte] Lte-advanced. https://www.3gpp.org/technologies/keywords-acronyms/97-lte-advanced. Accessed: 2019-05-09.

[MK15] M Meraj and S Kumar. Evolution of mobile wireless technology from 0g to 5g. *International Journal of Computer Science and Information Technologies*, 6(3):2545–2551, 2015.

[MKK17] Mehrnoosh Monshizadeh, Vikramajeet Khatri, and Raimo Kantola. Detection as a service: an sdn application. In *2017 19th International Conference on Advanced Communication Technology (ICACT)*, pages 285–290. IEEE, 2017.

[OFP] OpenFlow Overview data plane - packet lifecycle. http://flowgrammable.org/sdn/openflow/#tab_switch. Accessed: 2019-06-04.

[omna] OMNeT++ installation guide version 5.4.1. https://doc.omnetpp.org/omnetpp/InstallGuide.pdf. Accessed: 2019-05-29.

[omnb] OMNeTpp home page. https://omnetpp.org/. Accessed: 2018-11-03.

[omnc] OMNeTpp omnet simulation manual. https://doc.omnetpp.org/omnetpp/manual. Accessed: 2019-05-20.

[omnd] OMNeTpp simulation models page. https://omnetpp.org/models. Accessed: 2018-11-09.

[opea] GitHub openflowomnetsuite. https://github.com/lsinfo3/OpenFlowOMNeTSuite/tree/master/openflow. Accessed: 2019-05-26.

[opeb] OpenAirInterface home page. http://www.openairinterface.org/. Accessed: 2018-11-10.

[PLH+12] Ben Pfaff, Bob Lantz, Bea Heller, et al. Openflow switch specification, version 1.3. 0. *Open Networking Foundation*, pages 39–46, 2012.

[PLR07] Tao Peng, Christopher Leckie, and Kotagiri Ramamohanarao. Survey of network-based defense mechanisms countering the dos and ddos problems. *ACM Computing Surveys (CSUR)*, 39(1):3, 2007.

[RBB+16] P. Rost, A. Banchs, I. Berberana, M. Breitbach, M. Doll, H. Droste, C. Mannweiler, M. A. Puente, K. Samdanis, and B. Sayadi. Mobile network architecture evolution toward 5g. *IEEE Communications Magazine*, 54(5):84–91, May 2016.

[Sau] Martin Sauter. [wirlessmoves]lte and the number of simultaneously connected users. https://blog.wirelessmoves.com/2016/02/lte-and-the-number-of-simultaneously-connected-users.html. Accessed: 2019-05-03.

[Sau14] Martin Sauter. *From GSM to LTE-advanced: an introduction to mobile networks and mobile broadband*. John Wiley & Sons, 2014.

[sdna] The basics you need to know about sdn. https://hamilton-barnes.co.uk/sdn. Accessed: 2019-06-03.

[sdnb] "mpls-tp openflow protocol extensions for sptn" becomes a formal onf standard by unanimous approval. https://www.telecomtv.com/content/tracker/mpls-tp-openflow-protocol-extensions-for-sptn-becomes-a-formal-onf-standard-by-unanimous-app Accessed: 2019-05-09.

[sim] SimuLTE install guide. http://simulte.com/install.html. Accessed: 2019-05-16.

[SKK+97] C. L. Schuba, I. V. Krsul, M. G. Kuhn, E. H. Spafford, A. Sundaram, and D. Zamboni. Analysis of a denial of service attack on tcp. In *Proceedings. 1997 IEEE Symposium on Security and Privacy (Cat. No.97CB36097)*, pages 208–223, May 1997.

[SLLL14] Guolin Sun, Feng Liu, Junyu Lai, and Guisong Liu. Software defined wireless network architecture for the next generation mobile communication: Proposal and initial prototype. *Journal of Communications*, 9, 12 2014.

[sof] Softfire software description. https://www.softfire.eu/. Accessed: 2018-11-18.

[Sta15] William Stallings. *Background and Motivation of Software-Defined Networks (SDN)*. Addison-Wesley Professional, 2015.

[Syna] Network dos attacks. https://www.juniper.net/documentation/en_US/junos/topics/topic-map/security-network-dos-attack.html. Accessed: 2019-05-09.

[Synb] Syndefender. https://www.checkpoint.com/smb/help/utm1/8.2/6061.html. Accessed: 2019-05-09.

[TSD16] Marco Tiloca, Alexandra Stagkopoulou, and Gianluca Dini. Performance and security evaluation of SDN networks in omnet++/inet. *CoRR*, abs/1609.04554, 2016.

[TZJT13] Saman Taghavi Zargar, James Joshi, and David Tipper. A survey of defense mechanisms against distributed denial of service (ddos) flooding attacks. *IEEE Communications Surveys &amp Tutorials*, 15:2046 – 2069, 11 2013.

[vir] VirtualBox welcome to virtualbox.org. https://www.virtualbox.org/. Accessed: 2019-05-26.

[VSN14] Antonio Virdis, Giovanni Stea, and Giovanni Nardini. Simulte-a modular system-level simulator for lte/lte-a networks based on omnet++. In *Simulation and Modeling Methodologies, Technologies and Applications (SIMULTECH), 2014 International Conference on*, pages 59–70. IEEE, 2014.

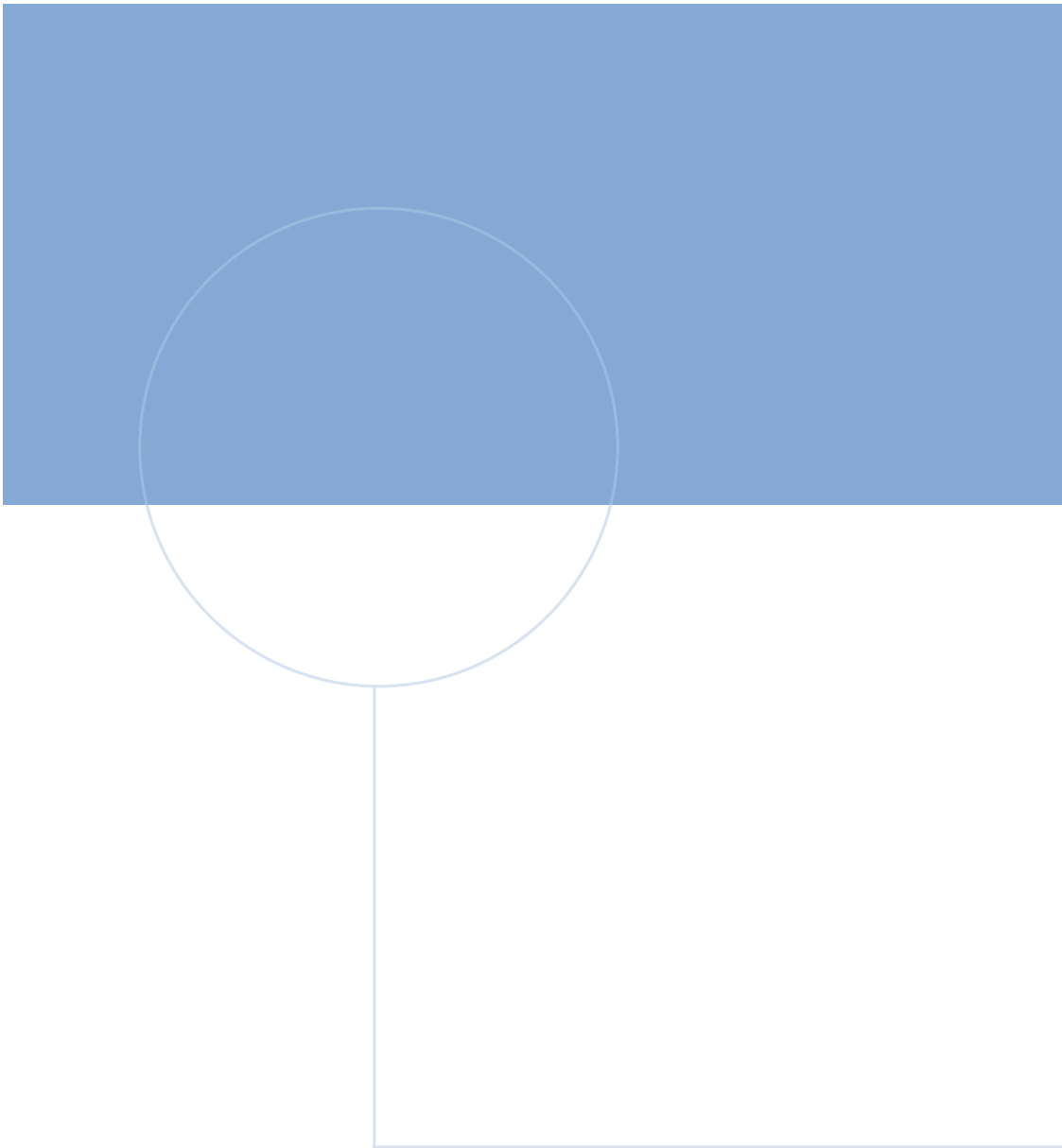
[Wie14] Roel J Wieringa. *Design science methodology for information systems and software engineering*. Springer, 2014.

[YGD16] Qiao Yan, Qingxiang Gong, and Fang-an Deng. Detection of ddos attacks against wireless sdn controllers based on the fuzzy synthetic evaluation decision-making model. *Adhoc & Sensor Wireless Networks*, 33, 2016.

[YGY17] Q. Yan, Q. Gong, and F. R. Yu. Effective software-defined networking controller scheduling method to mitigate ddos attacks. *Electronics Letters*, 53(7):469–471, 2017.

[ZLC+17] Haijun Zhang, Na Liu, Xiaoli Chu, Keping Long, A Aghvami, and Victor Leung. Network slicing based 5g and future mobile networks: Mobility, resource management, and challenges. *IEEE Communications Magazine*, 55, 01 2017.

[Zol15] Maede Zolanvari. Sdn for 5g. 2015.



NTNU
Norwegian University of
Science and Technology