UNIVERSITÄT ZU LÜBECK
INSTITUT FÜR MULTIMEDIALE
UND INTERAKTIVE SYSTEME

Titel der Arbeit

Masterarbeit/Bachelorarbeit

im Rahmen des Studiengangs
Medieninformatik
der Universität zu Lübeck

vorgelegt von:

Vorname Nachname

ausgegeben und betreut von:

Titel Name Erstgutachter

mit Unterstützung von:

Titel Name Betreuer

[Optional:] Die Arbeit ist im Rahmen einer Tätigkeit bei der Firma XYZ entstanden.

Lübeck, Tag. Monat. Jahr

Abstract

Der Abstract einer Abschlussarbeit sollte eine kurze Zusammenfassung enthalten, damit der Leser nach einigen Sätzen einen Eindruck davon bekommt, welches Thema bearbeitet wurde. Ein Abstract ist dabei kein "Teaser" sondern eher eine "Executive Summary".
Dieses Dokument dient als Vorlage und gleichzeitig als kleine Anleitung, um eine Abschlussarbeit mit LaTeX zu erstellen. Um das Template für die eigene Abschlussarbeit zu verwenden, kann einfach der vorhandene Text gelöscht und eigener Text hinzugefügt werden. Das Dokument enthält keine ausführliche Erklärung für das Arbeiten mit LaTeX, da es hierzu eine Vielzahl von Tutorials im Internet gibt. Stattdessen enthält es einige Tipps und Richtlinien. Der Quellcode ist ausführlich dokumentiert, damit es einfach ist das Template für die eigene Arbeit anzupassen.

Contents

1 Introduction

1.1 Motivation

1.2 Related Work

1.3 Research Questions

2 Background

2.1 Positioning in the distributed Test Support System

2.2 TCP/IP Reference Model

Protocols are the basis for communication between instances in a network. They specify rules that must be followed by all communication partners [10]. Reference models arrange protocols hierarchically in layers. Each layer solves a specific part of the communication task and uses the services of the layer below while providing certain services to the layer above [11].

Figure 2.1 illustrates the relationship between service and protocol. A service refers to a set of operations that a layer provides to the layer above it, and it defines the interface between the two layers [10].

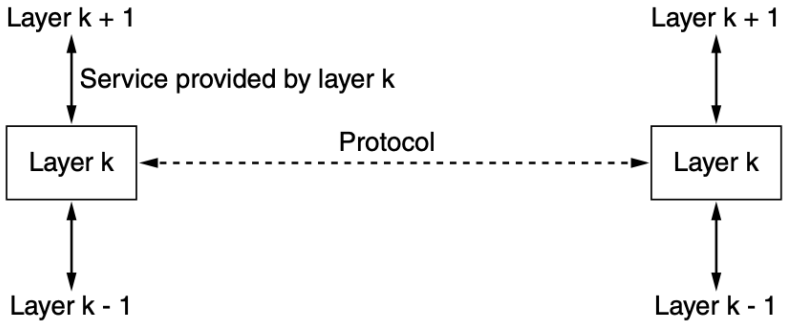

Figure 2.1: Relationship between service and protocol. Source: [10].

A protocol is a set of rules that define the format of messages exchanged within a layer [10]. These rules define the implementation of the service offered by the layer The transparency principle applies, meaning that the implementing protocol is transparent to the service user and can be changed as long as the service offered remains unchanged [11].

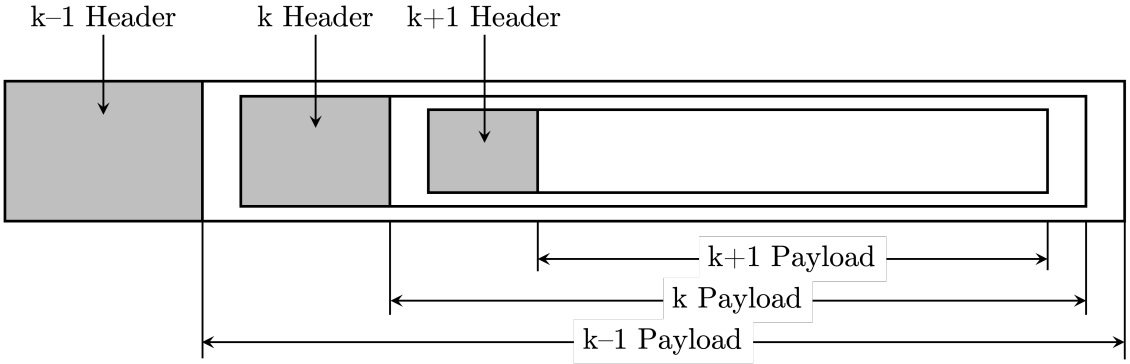

Figure 2.2: Encapsulation Principle. Adapted from: [10].

Protocols define the format of control information required by layer k to provide the service. This information is attached as a header or trailer to the data of layer k + 1, known as the payload, and is removed by the receiving instance. This principle is known as the 'Encapsulation Principle' and is illustrated in Figure 2.2 [10].

2.2.1 Introduction of the Reference Model

The following section presents a explanation of the TCP/IP reference model. Throughout this section, we will refer to the hybrid reference model proposed by Andrew S. Tanenbaum in [10]. Figure 2.3 shows this hybrid reference model. The physical layer is at the bottom, and the application layer is at the top. The tasks of each layer are briefly described here. For additional information, please refer to [10].

- The **Physical Layer** serves as the interface between a network node and the transmission medium, responsible for transmitting a bit stream. This involves line coding, which converts binary data into a signal. Additionally, the physical layer encompasses the transmission medium and the connection to this medium [10, 11].

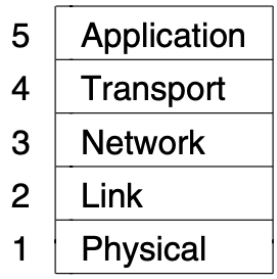

Figure 2.3: Hybrid TCP/IP Reference Model. Source: [10].

- The **Link Layer** facilitates reliable transmission of a sequence of bits (called a frame) between adjacent network nodes. This encompasses frame synchronization, which involves detecting frame boundaries in the bit stream, error protection, flow control, channel access control, and addressing [11].
- The **Network Layer** provides end-to-end communication between two network nodes. This includes addressing and routing [10, 11].
- The **Transport Layer** provides the transfer of a data stream of any length between two application processes. This involves collecting outgoing messages from all application processes and distributing incoming messages to them [11].
- The **Application Layer** serves as the interface to the application. It is responsible for implementing protocols for network use, such as file transfer or network management [11].

2.2.2 Protocols of the Reference Model

Figure 2.4 shows a selection of important protocols of the TCP/IP reference model including their assignment to the respective layer. The illustration also shows the dependency of the protocols on each other.

In this section, the characteristics of the protocols TCP, UDP, IP and Ethernet (IEEE 802.3), which are relevant for this work, are explained in detail according to [10]. Further information about the protocols of the TCP/IP reference model can be found in [10].

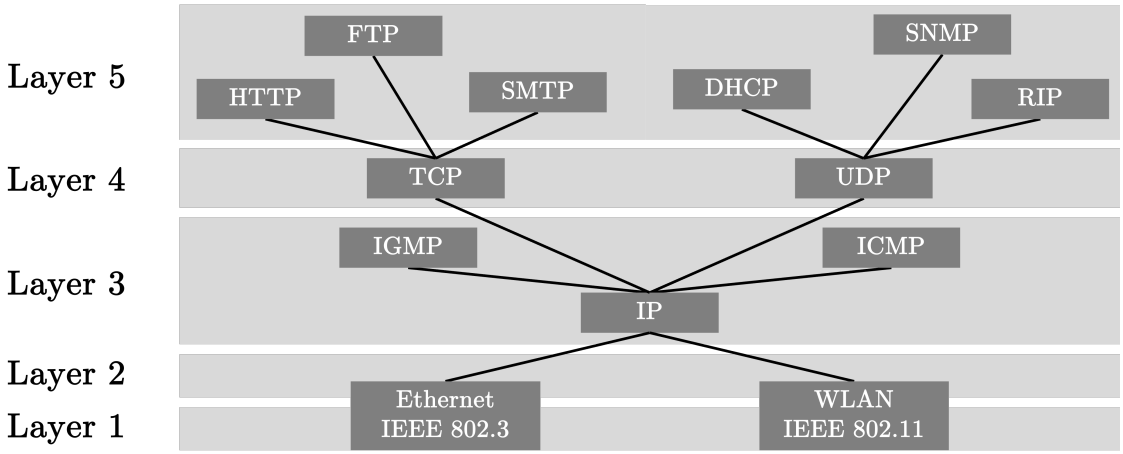

Figure 2.4: Selection of important protocols of the hybrid TCP/IP Reference Model. Adapted from: [11].

2.2.2.1 Ethernet (IEEE 802.3)

Ethernet, as defined by IEEE standard 802.3, specifies both hardware and software for wired data networks. This means that Ethernet includes both the physical layer and the link layer of the presented hybrid TCP/IP reference model.

2.2.2.1.1 Ethernet Physical Layer

The Ethernet physical layer consists of a number of standards that define different media types associated with different transmission rates and cable lengths.

Ethernet defines physical layer standards with transmission rates ranging from 10 Mbit/s to 1.6 Tbit/s, which is currently under development as the 802.3dj standard [4]. Both fiber and copper are used as transmission media. In the following, the 802.3an standard will be briefly discussed, since it is the one that will be used most in this thesis.

The 802.3an standard was published in 2006 and defines data transmission with a transmission rate of 10 Gbit/s over twisted-pair cables [2], also referred to as 10 GbE. Twisted-pair cables are copper cables in which pairs of copper wires are twisted together to reduce electromagnetic interference. Twisted-pair cables are divided into

categories based on various characteristics, such as shielding or twist strength [1]. For 802.3an, a maximum cable length of 100 meters is specified in conjunction with Cat7 cables. 802.3an specifies the RJ45 connector as the plug connector.

According to 802.3an, the PAM16 line coding is used for Ethernet at 10 Gbps. It uses the principle of pulse amplitude modulation, which is described in detail in [3]. PAM16 allows the transmission of data by varying the amplitude of a signal in 16 different stages. Each stage represents four bits of information.

In addition to 802.3an, the Ethernet physical layer according to 802.3ae was also used in this work. This also defines the physical layer with a transmission rate of 10 Gbit/s. However, fiber optic cables are used in conjunction with transceiver modules called SPF+ SR [2].

2.2.2.1.2 Ethernet Link Layer

At the link layer, Ethernet defines frame formatting, addressing, error detection, and access control. This is also called the Medium Access Control (MAC) sublayer.

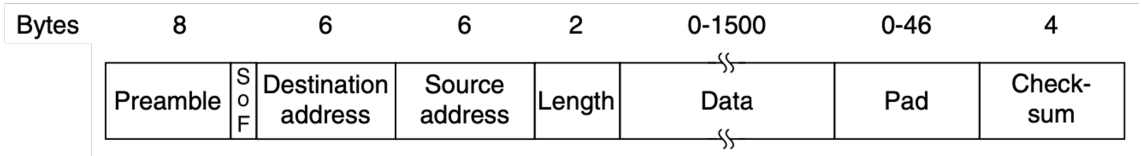

Figure 2.5: Structure of the Ethernet frame. Source: [10].

Figure 2.5 shows the IEE 802.3 frame format. The Ethernet header consists of the fields '*Destination address*', '*Source address*' and '*Length*' and therefore has a size of 14 bytes.

Each frame begins with a *preamble*. This has a length of 8 bytes and contains the bit sequence 10101010. An exception is the last byte, which contains the bit sequence 10101011 and is referred to as the *Start of Frame* (SoF). The preamble is used for synchronization between the sender and receiver. The last byte of the preamble marks the start of a frame [10].

This is followed by the *destination* and *source address*. This is the MAC address,

which is uniquely assigned globally to a network interface [11]. This consists of a manufacturer code with a length of 3 bytes, followed by the serial number of the network interface, which also has a length of 3 bytes. The MAC address enables the Ethernet protocol to uniquely identify a station in the local network.

The *Length* field specifies the length of the next data field. In IEEE 802.3 Ethernet, this has a maximum length, called the Maximum Transfer Unit (MTU), of 1500 bytes. However, there are Ethernet implementations that use a larger MTU than specified in the original standard. These are known as jumbo frames [12].

In addition to a maximum length, the Ethernet standard also specifies a minimum length. An entire Ethernet frame must therefore have a minimum length of 64 bytes from the destination address to the checksum. To ensure that this can be achieved even with a small data field, padding information is added. The specification of the minimum length is related to the access control used.

The Ethernet frame ends with a 4-byte *checksum* that is used for the Cyclic Redundancy Check (CRC) based on polynomial divisions, as explained in [10]. This checksum serves to detect errors during transmission.

Ethernet originally used a shared transmission medium, allowing multiple communication participants to use it simultaneously. To control access, the MAC sublayer employs the CSMA/CD algorithm, ensuring that only one device transmits data at a time. Each device listens to the medium (carrier sense) before sending data to determine whether it is free. It also performs collision detection to determine whether two devices have started sending at the same time. In such a case, the devices stop the transmission and retry it after a random waiting time to avoid the collision [10].

The 802.3an specification for 10 Gigabit Ethernet is exclusively for point-to-point full-duplex connections, which eliminates the need for access control such as CSMA/CD. As a result, it is no longer included in the specification [8].

In order to connect multiple network devices with point-to-point connections, Ethernet switches are used. They have multiple ports and forward packets based on the MAC address. Ethernet switches operate on layer 2 of the reference model.

To summarise, with Ethernet there is no guarantee that data will be transmitted

reliably and without loss. Although Ethernet uses CRC for error detection, faulty frames are generally discarded. Additionally, Ethernet does not provide flow control or overload detection, which must be performed by a higher layer.

2.2.2.2 IP

The Internet Protocol (IP) is a central protocol in the TCP/IP reference model. Its tasks include connecting different networks, addressing network participants, and fragmenting packets [11]. IP is a connectionless protocol that operates on the 'best effort' principle, meaning it does not guarantee delivery.

There are two versions of the Internet Protocol: IPv4 and IPv6. As this work uses the IPv4 protocol, it is presented in more detail below.

2.2.2.2.1 IP Header

The IPv4 datagram is divided into a header and a payload. The header typically spans 20 bytes, but may also include an optional variable-length section. The header is shown in Figure 2.6.

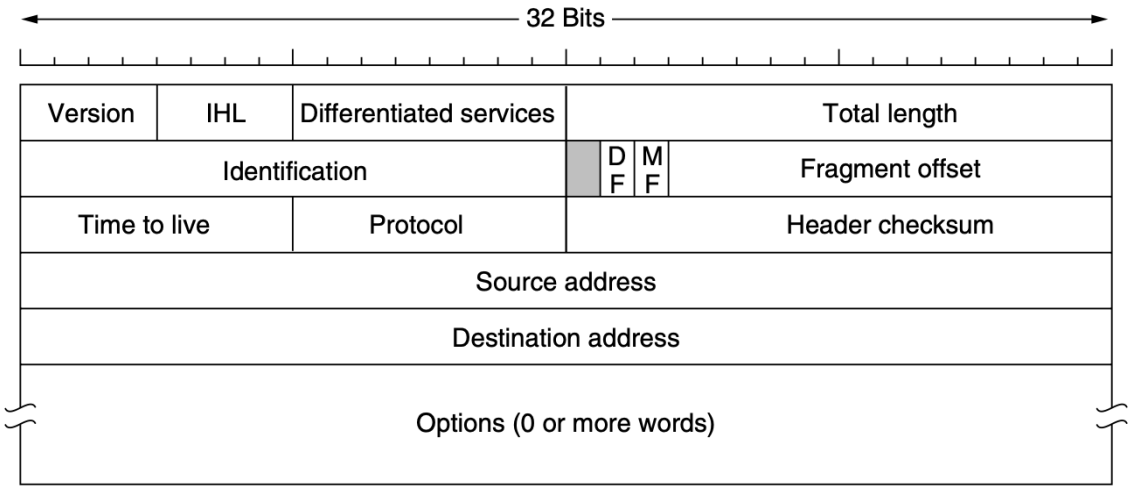

Figure 2.6: Structure of the IP Header. Source: [10].

The first field in the header is the 4-bit *Version* field. This indicates the IP version

used. For IPv4, the value is always 4.

The *IHL* (Internet Header Length) field specifies the number of 32-bit words in the header. This is necessary because the header can contain options and therefore has a variable length. The minimum value of the field is 5 if there are no options.

The *Differenciated Services* field specifies the service class of a packet, allowing for prioritisation of certain data traffic using Quality of Service (QoS). For a more detailed description of Quality of Service, please refer to section 2.5.1.

The *Total Length* field indicates the total length of the datagram, including the header. Due to the field size of 16 bits, the maximum length is 65535 bytes. However, a packet's length is also limited by the Layer 2 MTU [11], resulting in datagrams being split into multiple packets, known as fragmentation.

The *Identification* field is assigned a number by the sender, which is shared by all fragments of a datagram.

A flag field with a length of 3 bits follows, with the first bit being unused. The second section includes the 'Don't Fragment' (*DF*) flag, which indicates that intermediate stations should not fragment this packet. The third section contains the 'More Fragments' (*MF*) flag, which indicates whether additional fragments follow. This flag is set for all fragments except the last one of a datagram.

The *Fragment Offset* field specifies the position of a fragment in the entire datagram.

The *Time to live* (TTL) field specifies the maximum lifetime of a packet. The TTL value is measured in seconds and can be set to a maximum of 255 seconds. This is done to prevent packets from endlessly circulating in the network.

The *Protocol* field identifies the Layer 4 protocol used for the service. This allows the network layer to forward the packet to the corresponding protocol of the transport layer. The numbering of the protocols is standardized throughout the Internet.

The *Header checksum* field contains the checksum of the fields in the IP header. The IP datagram's user data is not verified for efficiency reasons [5]. The checksum is calculated by taking the 1's complement of the sum of all 16-bit half-words in the header. It is assumed that the checksum is zero at the start of the calculation for

the purpose of this algorithm.

The two 32-bit fields *Source Address* and *Destination Address* contain the Internet Protocol address, called the IP address. Section 2.2.2.2.2 provides further details on this topic.

The *Options* field can be used to add additional information to the IP protocol. For example, there are options to mark the route of a packet.

2.2.2.2.2 IP addresses and routing

This section provides a brief description of the structure and important properties of IP addresses. The network examined in this thesis is an isolated local network that is not connected to other networks. As a result, the network layer does not perform any routing based on IP addresses. For further information on routing, please refer to [10].

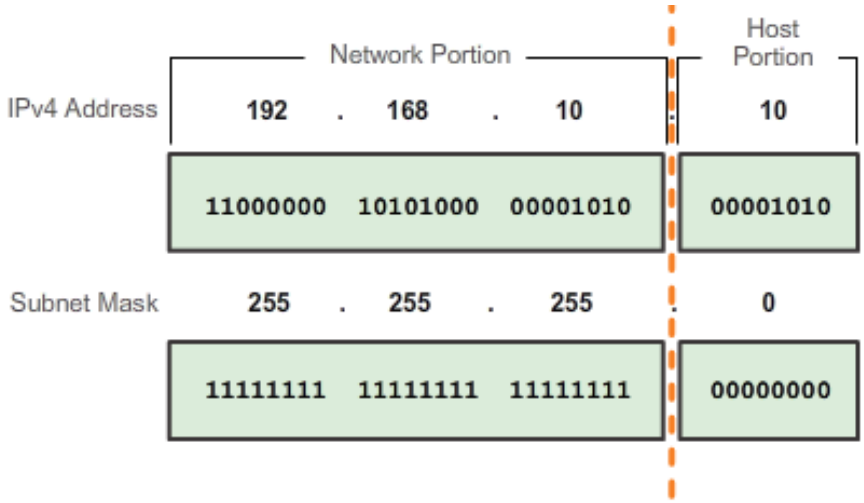

Figure 2.7: Structure of the IP address and subnet mask. Source: [9].

Every participant on the Internet has a unique address, known as an IP address. This has a total length of 32 bits and a hierarchical structure that divides the IP address into a network portion and a host portion. The division between the two parts is variable and is defined by a so-called subnet mask, which is illustrated in Figure 2.7. The bits of the network portion of the IP address are marked with ones.

- The **network portion** identifies a specific network, such as a local Ethernet network, and is the same for all participants in this network.
- The **host portion** identifies a specific device within this network.

Routing, which is another important task of the network layer, is based on IP addresses. The packet can be directed to its destination using the network portion of the IP address. The path to the destination is determined by specific routing algorithms. As mentioned earlier, the thesis only considers an isolated local network, so further discussion on routing will be omitted.

2.2.2.2.3 Address Resolution Protocol

The Address Resolution Protocol, abbreviated to ARP, is an auxiliary protocol of the network layer. Its task is to map the IP addresses to a MAC address and vice versa, as the sending and receiving of data in the underlying link layer is based on these MAC addresses [11].

2.2.2.2.4 Fragmentation and Defragmentation

As explained in 2.2.2.1.2, the link layer defines a maximum data size known as MTU. Since IPv4 datagrams have a maximum size of 65535 bytes, they must be divided into smaller packets, or fragments, each with its own IP header.

The IP header (refer to Figure 2.6) contains information necessary for the target system to assemble fragmented packets, a process known as defragmentation. This includes the ID that assigns all fragmented packets to a datagram, as well as the fragment offset that specifies their position within the datagram. The 'More Fragment' flag indicates whether additional fragments will follow.

Fragmentation has the advantage of allowing IPv4 datagrams larger than the MTU to be sent, but the disadvantage is that the loss of a single fragment results in the loss of the entire datagram. Additionally, fragmentation can cause packet reordering [7].

2.2.2.3 TCP and UDP

TPC and UDP are transport layer protocols. As a service, they provide the transmission of a data stream of any length between two application processes. The services of the network layer are used for this purpose.

2.2.2.3.1 TCP

TCP provides **reliable** transmission of a byte stream in a **connection-oriented** manner. A virtual connection is established between the two instances before transmission, which is terminated after transmission.

TCP also implements flow control to ensure reliable data transfer between sender and receiver without losses and to prevent overloading at the receiver. TCP provides congestion control to prevent network overload and ensures reliable transmission using Positive Acknowledgement with Re-Transmission (PAR) algorithm [6].

TCP is known for its secure data transmission. However, it requires a significant amount of control information to implement its functions. The Transmission Control Protocol (TCP) header is 20 bytes in size. In addition to an application identifier (port number), it contains flow control and congestion control information. This overhead can negatively impact transmission speed. Additionally, the data loss from the underlying layers combined with the flow control used by TCP leads to delays and reduced throughput, which can have a significant impact on the performance of the application.

2.2.2.3.2 UDP

In contrast to TCP, UDP is an **unreliable** and **connectionless** protocol. The protocol sends packets, called datagrams or segments, individually. UDP lacks mechanisms for detecting the loss of individual datagrams, and the correct sequence of these is not guaranteed.

Figure 2.8 displays the UDP header, which has a size of 8 bytes. It is considerably smaller than the TPC header, which has a size of 20 bytes.

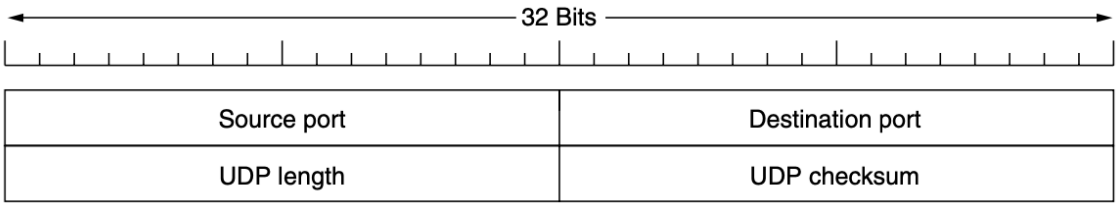

Figure 2.8: Structure of the UDP Header. Source: [10].

The header includes the fields *Source port* and *Destination port* to identify the endpoints in the respective instance. When a packet arrives, the payload is passed to the application using the appropriate port number via the UDP protocol.

The *UDP length* field indicates the length of the segment, including the header. The maximum length of data that can be transmitted via UDP is limited to 65,515 bytes due to the underlying Internet Protocol.

The last field of the header is a 16-bit *UDP checksum*. This checksum is formed via the so-called IP pseudoheader, which contains the source and destination IP address, the protocol number from the IP header, and the *UDP length* field of the UDP header.

Compared to TCP, UDP can achieve higher data transmission speeds due to its lower protocol overhead, ase the UDP header is only 8 bytes in size. Furthermore, UDP does not require an acknowledgement of the transport or other mechanisms used by TCP to provide a reliable connection. This makes it very efficient and reduces processing overhead.

2.3 Linux-Kernel

2.4 UDP communication with a Linux Operating System

2.4.1 Socket Types

2.4.1.1 UDP Sockets

2.4.1.2 Raw Sockets

2.4.1.3 Packet Sockets

2.4.2 Sending and Receiving a Packet

2.5 Tuning Options

2.5.1 Quality of Service

3 Method

3.1 Setup

3.1.1 Hardware Setup

3.1.2 Software Setup

3.2 Architecture

3.2.1 Setup with Ethernet Switch

3.2.2 Setup with Star Topology

3.3 TestSuite

3.3.1 Description of Software Design

3.3.2 Generation and Measurement of Target Communication

3.3.3 Recorded and Analyzed Data

3.4 Generation of additional System Load

3.4.1 stress-ng

3.4.2 iPerf

4 Tutorial zu dieser LaTeX-Vorlage

Dieses Kapitel ist spezifisch für die LaTeXVorlage und sollte natürlich in der finalen Abgabe nicht enthalten sein. Dieser Teil der Arbeit kann bei Bedarf durch einen Kommentar einfach ausgeblendet werden (siehe thesis.tex Zeile 125).

4.1 Verwendung dieser Vorlage

Dieses Template ist für die Verwendung mit pdflatex gedacht. Am einfachsten ist es die Vorlage in Overleaf [?] zu öffnen. Overleaf ist eine Onlineanwendung zum Arbeiten mit LaTeX, was den Vorteil hat, dass nichts lokal installiert werden muss und mit jedem Betriebssystem gearbeitet werden kann, das über einen Browser verfügt. Außerdem ist es möglich mit mehreren Personen gemeinsam an einem Projekt zu arbeiten. Die Vorlage kann auch mit einer lokalen Installation verwendet werden. Die Website von Overleaf bietet zudem einige gute Tutorials zum Arbeiten mit LaTeX: `https://www.overleaf.com/learn`.

Fehler und Warnungen, die beim Compilieren erzeugt werden, sollten direkt behoben werden, da es später schwierig sein kann den eigentlichen Auslöser einer Fehlermeldung zu finden. Manchmal sieht das Dokument trotz Fehlermeldung oder Warnung korrekt aus, der Fehler macht sich dann aber später bemerkbar. Die Meldungen sind leider oft nicht sehr aussagekräftig, weshalb es am einfachsten ist direkt nach dem Auftreten eines Fehlers den Teil der Arbeit anzuschauen, der als letztes geändert wurde.

4.2 Projektstruktur

Der Hauptteil einer Thesis besteht üblicherweise aus mehreren Kapiteln, die verschiedene Aspekte der Arbeit beleuchten. Es ist ratsam für jedes Kapitel ein eigene Tex-Datei anzulegen, damit der Quellcode übersichtlich bleibt. Durch einen numerischen Präfix (z.B.: `20_relatedwork.tex` siehe Abb. 4.1) werden die Quelldateien in der richtigen Reihenfolge in Overleaf angezeigt. Wir verwenden 20 anstatt 2, damit wir nachträglich auch noch 21, 22, etc. einfügen können.

4.2.1 Unterkapitel

Unterkapitel sollten ein abgeschlossenes Thema behandeln. Einzelne Unterkapitel in einem Kapitel sind zu vermeiden, also z. B. in Kapitel 2 das Unterkapitel 2.1, aber kein weiteres Unterkapitel. In diesem Fall ist es besser entweder den Inhalt von 2.1 direkt in Kapitel 2 zu schreiben, oder falls 2 und 2.1 thematisch zu weit voneinander entfernt sind, aus Unterkapitel 2.1 ein eigenes Kapitel 3 zu machen. Dieses Unterkapitel ist ein negativ Beispiel dafür.

4.3 Grafiken

In der Informatik sind die häufigsten Grafiken entweder Diagramme oder Plots. Beide Arten von Grafiken lassen sich gut als Vektorgrafiken erstellen und einbinden. Der Vorteil von Vektor- gegenüber Pixelgrafiken ist, dass beliebig weit in eine Grafik hereingezoomt werden kann, ohne dass sie unscharf wird. Zudem benötigen Vektorgrafiken meistens weniger Speicherplatz.

4.3.1 Vektor- vs Pixelgrafiken

Für die meisten Abbildungen sollte man Vektorgrafiken verwenden, diese können verlustfrei skaliert werden und bieten somit die meisten Freiheiten. Wenn man Grafiken in R erstellt, können die Plots als pdf oder svg-Dateien abgespeichert werden

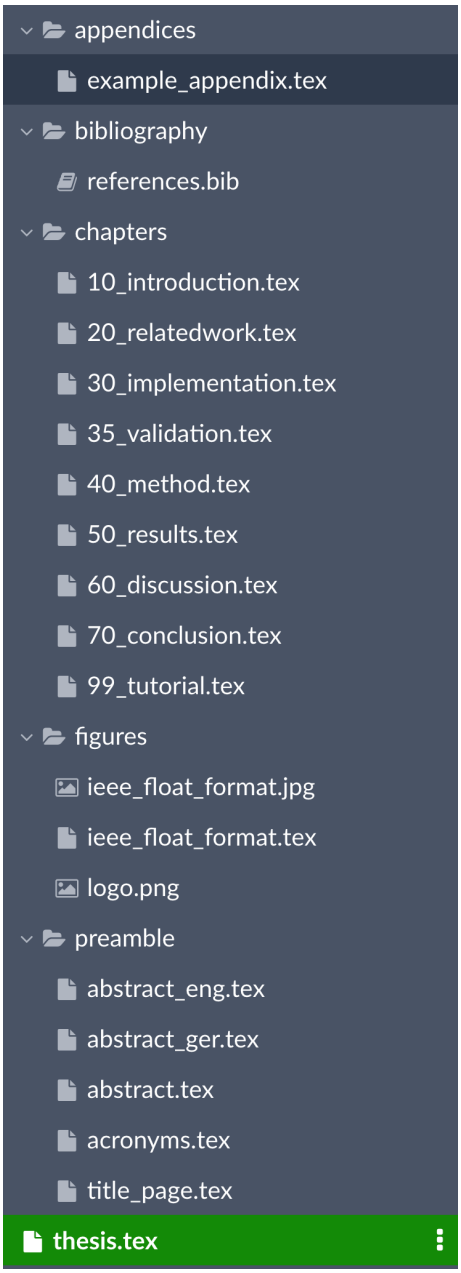

Figure 4.1: Ordnerstruktur eines LaTeX-Projektes

und liegen dann ebenfalls als Vektorgrafik vor. Ein weiteres beliebtes Programm zum Erstellen von Vektorgrafiken ist Inkscape [?]. Zudem bieten viele Programme die Möglichkeit eine Grafik z. B. als PDF zu exportieren, was in LaTeX als Vektorgrafik eingebunden werden kann. Adobe Indesign wird auch häufig zum Erstellen von Vektorgrafiken verwendet.

4.3.1.0.1 Profi-Tipp TikZ.

Grafiken können direkt in LaTeX mit dem TikZ Paket [?] erstellt werden. Die Verwendung ist etwas gewöhnungsbedürftig, da Grafiken mit Code beschrieben werden, bietet aber viele Freiheiten. Außerdem werden die so erstellten Grafiken direkt in LaTeX gerendert und verwenden die selbe Schriftart wie im Text und eine konsistente Schriftgröße im gesamten Dokument.

Pixelgrafiken lassen sich nicht immer vermeiden, z. B. wenn eine Foto in die Arbeit eingebunden werden soll. In diesem Fall sollte darauf geachtet werden, dass die Grafik über eine ausreichende Auflösung verfügt. Eine Auflösung von 300 dpi ist ein guter Richtwert, um beim Drucken ein gutes Ergebnis zu erhalten.

Abbildungen 4.2 und 4.3 zeigen beide den Aufbau des IEEE Floating Point Formats. Abbidlung 4.3 ist eine Pixelgrafik, während Abbildung 4.2 mit TikZ erstellt wurde. Der Unterschied wird beim hereinzoomen deutlich.

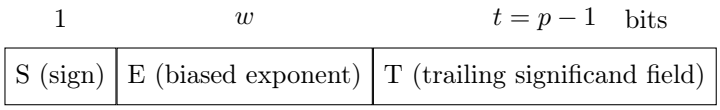

Figure 4.2: Aufbau des IEEE Floating Point Formats als Vektorgrafik mit TikZ erzeugt.

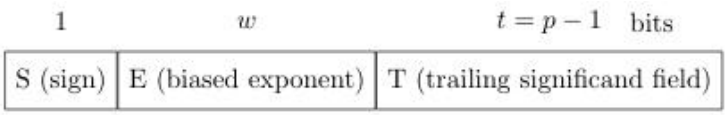

Figure 4.3: Aufbau des IEEE Floating Point Formats als Pixelgrafik.

4.4 Tabellen

Tabellen können in LaTeX direkt erstellt werden. Tabelle 4.1 zeigt ein Beispiel dafür. Einfache Tabellen lassen sich schnell erstellen, bei komplizierteren Tabellen ist es manchmal einfacher zusätzliche Pakete zu verwenden. Mit dem Paket `multirow` können z. B. einfacher Tabellen erstellt werden, bei denen einzelne Zeilen oder Spalten zusammengefasst sind.

Unter `https://www.tablesgenerator.com/` findet man ein hilfreiches online Werkzeug zum erstellen von Tabellen. Wichtig ist es hier den "Default table style" zum "Booktabs table style umzustellen".

Parameter	binary16	binary32	binary64	binary128
k, storage width in bits	16	32	64	128
w, exponent field width in bits	5	8	11	15
t, significand field width in bits	10	23	52	112
emax, maximum exponent e	15	127	1023	16383
bias, $E - e$	15	127	1023	16383

Table 4.1: IEEE 754-2019 Floating Point Formate als Beispiel für das Einbinden einer Tabelle.

4.5 Quellcode

Um Quellcode in die Arbeit einzubinden, können in LaTeX Listings verwendet werden. Es gibt für populäre Sprachen vorgefertigte Umgebungen, welche die Syntax farblich hervorheben. Quellcode sollte eingebunden werden, wenn eine konkrete Implementierung in einer Sprache erläutert wird. Für die Erklärung eines Algorithmus ist es oft übersichtlicher ein Schaubild oder Pseudocode zu verwenden. Es sollten nur kurze Codeabschnitte eingebunden werden, die für den Leser einfach nachvollziehbar sind und nur den für die Erklärung relevanten Code enthalten. Längere Codeabschnitte können im Anhang stehen. Der komplette Code, der für die Arbeit geschrieben wurde, sollte in einem Repository (Gitlab) abgelegt werden.

Listing 4.1 zeigt ein Beispiel für ein Codelisting in der Programmiersprache C.

Algorithmus 4.1 zeigt einen Routing Algorithmus als Pseudocode. Der Code wurde mit dem Paket algorithm2e [**?**] erstellt.

```c
#include <stdio.h>
// comments are highlighted in green
void main() {
    // keywords of the language are highlighted in blue
    for (int i = 0; i <= 42; ++i) {
        printf("%d\n", i);
    }
    // strings are highlighted in red
    printf("Hello World!");
}
```

Listing 4.1: Beispiel für ein Codelisting in der Sprache C.

if *destination in west direction* **then**
  go West;
else if *destination in same column* **then**
  if *destination in north direction* **then**
    go North;
  else
    go South;
  end
else if *destination in north east direction* **then**
  go North or go East
else if *destination in south east direction* **then**
  go South or go East
else if *destination in same row and in east direction* **then**
  go East;
else if *at destination* **then**
  done;

Algorithmus 4.1: West First-Routing Algorithm.

4.6 Literatur

Ein Literaturverzeichnis sollte mit dem apa Paket für BibLatex erstellt werden. Dazu wird für jede Quelle ein Eintrag in der Datei `references.bib` angelegt. An der

passenden Stelle im Text können diese Einträge mit dem `\cite{}` Befehl zitiert werden. Für jede Quelle die zitiert wird, legt LaTeX im Literaturverzeichnis einen Eintrag an.

Beschreibungen der Quellen im Bibtex-Format müssen meistens nicht selbst erstellt werden, sondern können direkt bei vielen Verlagen und Bibliotheken direkt generiert werden. Google bietet mit dem "Scholar-Button" ein Chromium Plugin, mit dem schnell bibtex-Einträge generiert werden können. Bei Google-Scholar generierten Bibtex-Einträgen muss auch eine manuelle Endkontrolle stattfinden, um zu prüfen, ob die Daten in Google korrekt gespeichert waren (z.B. fehlende Autoren, falsche Jahreszahl, etc.).

Hier[1] gibt es ein hilfreiches Cheat-Sheet.

Hinweis: Keine dieser Referenzen müssen Sie in Ihrer Arbeit zitieren!

4.7 Abkürzungen

Für jede verwendete Abkürzung kann ein Eintrag in der Datei `acronyms.tex` angelegt werden. Wenn diese Abkürzung im Text zum ersten Mal auftaucht, sollte der Begriff ausgeschrieben werden mit der Abkürzung in Klammern dahinter. Bei weiteren Vorkommen im Text kann dann die eigentliche Abkürzung verwendet werden. In LaTeX gibt es dafür spezielle Befehle. Beispiel für ausgeschriebene Abkürzung (siehe Quelltext des Dokuments für die entsprechenden Befehle).

Erste Verwendung mit Erläuterung: Network Interface Controller (NIC).
Beispiel für das Verwenden der Abkürzung: NIC.

Die verwendeten Abkürzungen werden automatisch im Abkürzungsverzeichnis aufgelistet.

[1] `https://tug.ctan.org/info/biblatex-cheatsheet/biblatex-cheatsheet.pdf`

´

5 Bibliography

[1] ISO/IEC 11801-1:2017: Information technology - Generic cabling for customer premises, Part 1: General requirements. International Standard, 2017.

[2] Elektronik-Kompendium. 10-Gigabit-Ethernet / 10GE / IEEE 802.3ae / IEEE 802.3an, 2023. Accessed on 27.11.2023. URL: https://www.elektronik-kompendium.de/sites/net/1107311.htm.

[3] Elprocus. Pulse Amplitude Modulation, 2023. Accessed on 27.11.2023. URL: https://www.elprocus.com/pulse-amplitude-modulation/.

[4] IEEE P802.3dj Task Force. IEEE 802.3 Ethernet WG Opening Plenary, 2023. Accessed on 27.11.2023. URL: https://grouper.ieee.org/groups/802/3/minutes/mar23/2303_3dj_open_report.pdf.

[5] Heiko Holtkamp. TCP/IP im Detail: Internet-Schicht, 2001. Accessed on 21.11.2023. URL: http://www.cfd.tu-berlin.de/Lehre/EDV2/tcpip/kap_2_3.html.

[6] Heiko Holtkamp. TCP/IP im Detail: Transportschicht, 2001. Accessed on 21.11.2023. URL: http://www.cfd.tu-berlin.de/Lehre/EDV2/tcpip/kap_2_4.html.

[7] Steven Iveson. IP Fragmentation in Detail, 2019. Accessed on 29.11.2023. URL: https://packetpushers.net/ip-fragmentation-in-detail/.

[8] Stefan Luber and Andreas Donner. Definition: Was ist 10GbE?, 2018. Accessed on 27.11.2023. URL: https://www.ip-insider.de/was-ist-10gbe-a-680925/.

[9] Shubham Negi. DCSE252 — Course Notes — Unit 1–5, 2023. Accessed on 29.11.2023. URL: https://medium.com/@shubham64negi/cn-unit-1-5-9b60cbf94230.

[10] Andrew S. Tanenbaum and David J. Wetherall. *Computer Networks*. Prentice Hall Press, USA, 5th edition, 2010.

[11] Inge Weigel. Computer Networks. Lecture Material in the Course "Computer Networks", 2021. Technische Hochschule Ingolstadt.

[12] Robert Winter, Rich Hernandez, Gaurav Chawla, et al. Ethernet jumbo frames. Technical report, Ethernet Alliance, Beaverton, OR, November 2009. URL: http://www.ethernetalliance.org/wp-content/uploads/2011/10/EA-Ethernet-Jumbo-Frames-v0-1.pdf.

A Appendix

Der Anhang kann Teile der Arbeit enthalten, die im Hauptteil zu weit führen würden, aber trotzdem für manche Leser interessant sein könnten. Das können z. B. die Ergebnisse weiterer Messungen sein, die im Hauptteil nicht betrachtet werden aber trotzdem durchgeführt wurden. Es ist ebenfalls möglich längere Codeabschnitte anzuhängen. Jedoch sollte der Anhang kein Ersatz für ein Repository sein und nicht einfach den gesamten Code enthalten.

B List of Figures

C List of Tables

D Listings