**DIPLOMARBEIT**

**Virtueller Server**

**Höhere Technische Bundeslehr- und Versuchsanstalt Anichstraße**

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**Abteilung**

**Elektronik und technische Informatik**

|  |  |  |
| --- | --- | --- |
| Ausgeführt im Schuljahr 2022/23 von: |  | Betreuer/Betreuerin: |
| Simon Angerer 5AHEL  Patrick Edelmann 5AHEL |  | Dipl.-Ing. Kevin Schiechtl |

Innsbruck, am 31.03.2023

Abgabevermerk: Betreuer/in: Dipl.-Ing. Kevin Schiechtl

Datum: 31.03.2023

## Kurzfassung /Abstract

Die vorliegende Diplomarbeit beschäftigt sich mit einem virtuellen Server, der mithilfe eines Python Client Programmes kommuniziert. Viele Firmen bieten keine geeigneten virtuellen Server für den allgemeinen Gebrauch für junge Menschen an. Entweder der Preis oder die unübersichtliche Konfigurationsvielfalt stellt ein Problem dar. Aus diesen Gründen, soll diese Arbeit Firmen, Privatpersonen ein kostengünstiges und einfaches Bild von virtuellen Servern zeigen. Das geplante Ergebnis ist nachhaltig und auch verwertbar. Die Dokumentation beinhaltet unter anderem Prototypen, die einzelnen Konfigurationsschritte, den Source Code für das Client Programm, Analyse der Ergebnisse und das fertige Produkt. Das geplante Endergebnis beinhaltet einen funktionsfähigen virtuellen Server, der mit einem Python Client Programm kommunizieren kann. Ziel der Diplomarbeit ist die Entwicklung eines virtuellen Servers, welcher Wissen im Bereich Software- und Netzwerktechnik erfordert. Die Kommunikation zu einem Client-Laptop soll nach Projektfertigstellung möglich sein. Umgesetzt werden Kenntnisse der Softwareprogrammierung, Kommunikationssysteme und -netze sowie Projektmanagement und Organisation.

This thesis is about a virtual server that communicates using a Python client program. Many companies do not offer suitable virtual servers for typical use by young people. Either the price or the confusing variety of configurations is a problem. For these reasons, this work is intended to show companies, private persons an inexpensive and simple picture of virtual servers. The planned result is sustainable and usable. The documentation includes prototypes, the individual configuration steps, the source code for the client program, analysis of the results and the finished product. The planned result includes a functional virtual server that can communicate with a Python client program. The aim of the thesis is the development of a virtual server, which requires knowledge in the field of software and network technology. Communication with a client laptop should be possible after completion of the project. Knowledge of software programming, communication systems and networks as well as project management and organization are implemented.

## Vorwort/Danksagung

Wir bedanken uns hiermit recht herzlich bei allen, die uns während der Erarbeitung dieser Diplomarbeit unterstützt haben.

Zuerst gebührt unser Dank Herrn Kevin Schiechtl, Dipl.-Ing. (FH), der unsere Diplomarbeit betreut und begutachtet hat. Seine hilfreichen Anregungen sowie seine konstruktive Kritik, haben uns stark bei der Erstellung dieser Arbeit unterstützt. Wir schätzen Ihre Geduld und Ihr Engagement, besonders in den schwierigen Phasen des Projekts, sehr. Ihre regelmäßigen Rückmeldungen, Ihr wertvolles Feedback haben uns geholfen, unsere Arbeit zu verbessern und unsere Ideen zu entwickeln.

Weiters möchten wir uns ebenso bei unserem privaten Umfeld bedanken. Hier gilt unser Dank besonders unseren Eltern, die uns durch ihre Unterstützung unsere schulische Ausbildung merklich erleichtern. Besonders auch mit ihrer emotionalen Unterstützung schafften wir es durch jede Tiefphase in unserer schulischen Laufbahn.

Das Projektteam:

*Simon Angerer*

*Patrick Edelmann*

Innsbruck, 10.03.2023

## Projektergebnis

Es wurden alle Aufgaben laut Meilensteine durchgeführt und dokumentiert. Die Ergebnisse eines virtuellen Ubuntu Linux Servers ermöglicht es Benutzern, einen Server mit einer Vielzahl von Funktionen einzurichten. Der Server unterstützt DHCP, DNS, Firewall, Mail Server, Admin Benutzer und normale Benutzer. Der Server ist auch mit einem Python Client Programm kompatibel, das es Benutzern ermöglicht, eine Verbindung zu dem Server herzustellen und mit ihm zu kommunizieren. Der Server bietet Benutzern die Möglichkeit, ihre Netzwerkressourcen zu schützen. Dazu gehört die Verwendung der Firewall, die den Netzwerkverkehr kontrolliert und unerwünschte Verkehr blockiert. Der Server kann auch DHCP und DNS konfigurieren, um Netzwerkverbindungen zu verwalten. Darüber hinaus bietet der Server einen Mail Server, der Benutzern ermöglicht, E-Mails zu senden und zu empfangen. Der Server kann auch verschiedene Benutzerprofile erstellen, damit Benutzer mit unterschiedlichen Rechten auf den Server zugreifen können. Der Server kann zudem verschiedene Admin Benutzerprofile erstellen, um den Zugriff auf den Server zu kontrollieren und zu verwalten. Der Server ist auch mit einem Python Client Programm kompatibel, das Benutzern ermöglicht, eine Verbindung zum Server aufzubauen und mit ihm zu interagieren.

**Inhaltsverzeichnis**

[Kurzfassung /Abstract i](#_Toc129305446)

[Vorwort/Danksagung i](#_Toc129305447)

[Projektergebnis i](#_Toc129305448)

[1 Einleitung 1](#_Toc129305449)

[2 Vertiefende Aufgabenstellung 1](#_Toc129305450)

[2.1 Patrick Edelmann 1](#_Toc129305451)

[2.2 Simon Angerer 1](#_Toc129305452)

[3 Dokumentation der Arbeit 2](#_Toc129305453)

[3.1 Grundkonzept 2](#_Toc129305454)

[3.2 Theoretischer Teil 3](#_Toc129305455)

[3.2.1 DHCP 3](#_Toc129305456)

[3.2.2 Admin-Benutzer 7](#_Toc129305457)

[3.2.3 Mail-Server 9](#_Toc129305458)

[3.2.4 Python Client 12](#_Toc129305459)

[3.2.5 NCAT 13](#_Toc129305460)

[3.2.6 Geschichte der virtuellen Server 15](#_Toc129305461)

[3.2.7 RFC 17](#_Toc129305462)

[3.2.8 DNS 19](#_Toc129305463)

[3.2.9 Normaler Benutzer 22](#_Toc129305464)

[3.2.10 Active Directory 23](#_Toc129305465)

[3.2.11 Firewall 24](#_Toc129305466)

[3.3 Praktischer Teil 26](#_Toc129305467)

[3.3.1 Praktische Umsetzung 26](#_Toc129305468)

[3.3.2 Server 27](#_Toc129305469)

[3.3.3 DHCP-Server 33](#_Toc129305470)

[3.3.4 Admin Benutzer 35](#_Toc129305471)

[3.3.5 Mail-Server 37](#_Toc129305472)

[3.3.6 Python Client Programm 41](#_Toc129305473)

[3.3.7 DNS-Server 45](#_Toc129305474)

[3.3.8 Normaler Benutzer 46](#_Toc129305475)

[3.3.9 Active Directory 48](#_Toc129305476)

[3.3.10 Firewall 51](#_Toc129305477)

[4 Erklärung der Eigenständigkeit der Arbeit 53](#_Toc129305478)

[5 Abbildungsverzeichnis 54](#_Toc129305479)

[6 Tabellenverzeichnis 55](#_Toc129305480)

[7 Literaturverzeichnis 55](#_Toc129305481)

[8 Abkürzungs- und Symbolverzeichnis 57](#_Toc129305482)

[9 Anhang 58](#_Toc129305483)

[9.1 Schlussfolgerung / Projekterfahrung 58](#_Toc129305484)

[10 Projektterminplanung 59](#_Toc129305485)

[10.1 Simon Angerer 59](#_Toc129305486)

[10.2 Patrick Edelmann 59](#_Toc129305487)

[11 Arbeitsnachweis Diplomarbeit 60](#_Toc129305488)

1. Einleitung

Viele Firmen bieten keine geeigneten virtuellen Server für den allgemeinen Gebrauch für junge Menschen an. Entweder der Preis oder die unübersichtliche Konfigurationsvielfalt stellt ein Problem dar. Aus diesen Gründen, soll diese Arbeit Firmen, Privatpersonen ein kostengünstiges und einfaches Bild von virtuellen Servern zeigen. Ziel der Diplomarbeit ist die Entwicklung eines virtuellen Servers, welcher Wissen im Bereich Software- und Netzwerktechnik erfordert. Die Kommunikation zu einem Client-Laptop soll nach Projektfertigstellung möglich sein. Umgesetzt werden die Kenntnisse der Softwareprogrammierung, Kommunikationssysteme und -netze sowie Projektmanagement und Organisation. Die Aufgabe der Diplomarbeit lautet einen virtuellen Server zu konfigurieren. Der Server sollte dann aber mit Features geupdatet werden wie mit einem DNS-Server, DHCP-Server, Mail-Server, Admin und normalen Benutzer, Firewall. Die einzelnen Erweiterungen werden auch in der Theorie behandelt. Zum Schluss sollte noch ein Python Programm realisiert werden, dass mit dem Server kommunizieren kann bzw. soll.

1. Vertiefende Aufgabenstellung

## Patrick Edelmann

Befasst sich mit theoretischen Grundlagen von DHCP, Admin-Rechten, Mail-Server und Python. Er setzt den virtuellen Server auf, befasst sich mit der DHCP-Funktion und der Konfiguration. Er legt einen Benutzer, der Admin Rechte hat an, befasst sich mit Grundkonzept und praktischer Umsetzung. Ebenfalls befasst er sich mit einem Mail-Server und mit einem Python Client-Programm.

## Simon Angerer

Befasst sich mit theoretischen Grundlagen von DNS, Active Directory, Firewall und Benutzer mit gewissen Rechten, der Geschichte der virtuellen Server, mit der DNS-Konfiguration am Server. Er befasst sich auch mit dem Anlegen eines Benutzers, der nur Lese-oder Schreibrechte hat, ebenfalls am Server, auch mit Active Directory und der Installation der Firewall.

# Dokumentation der Arbeit

## Grundkonzept

Der Titel der Diplomarbeit lautet Virtueller Server. Die Arbeit soll aus einem funktionierenden Server und einem Python Client Programm, was später mit dem Server kommunizieren kann, bestehen. Der Server wird auch mit DHCP, DNS, AD, diverse Benutzer (Admin bzw. eingeschränkte Rechte) und einem Mail-Server (POP3 bzw. die neueste Variante IMAP) ausgestattet. Die Ausgangssituation dieser Arbeit ist, dass in vielen Unternehmen kein geeigneter virtueller Server für den allgemeinen Gebrauch angeboten wird. Es spricht viel dafür: virtuelle Server sind schnell verfügbar, Zentralisierte Verwaltung und Kontrolle, Schonung von Ressourcen, Kosteneffizienz, Flexibilität und Desaster Recovery. Allerdings hat diese Art von Technologie auch ihre Nachteile. Die da wären, Leistungseinbrüche, Abhängigkeit vom Hauptsystem und Sicherheitsproblematik. Ziel der Diplomarbeit ist die Entwicklung eines virtuellen Servers, welcher Wissen in Bereich Software- und Netzwerktechnik erfordert. Die Kommunikation zu einem Python-Client Programm soll nach Projektfertigstellung möglich sein. Umgesetzt werden Kenntnisse der Softwareprogrammierung, Kommunikationssysteme und -netze sowie Projektmanagement und Organisation. Die Zielgruppe was angesprochen werden mit diesem Projekt, sind vor allem Technik – Unternehmen aber auch junge Leute, die sich mit diesen Themen auseinandersetzten. Geplante Maßnahmen ist die Konfiguration eines Linux basierten virtuellen Servers, der auf der virtuellen Maschine Virtual Box läuft. Weiters geplant sind die Erweiterung DHCP, DNS, AD, IMAP, Amin- bzw. beschränkte Recht. Wenn das einwandfrei läuft, dann wird ein Python Client-Programm geschrieben, was mit dem Server kommunizieren soll. Die Diplomarbeit wird von zwei HTL-Schüler durchgeführt. Die Schüler heißen Simon Angerer und Patrick Edelmann. Eine Kooperation mit einer Firma wurde versucht, jedoch ohne Erfolg. Die ganze Dokumentation der Diplomarbeit wurde in einer sachlichen Perspektive durchgeführt, dies bedeutet keine ich oder wir Sätze.

## Theoretischer Teil

### DHCP

#### Geschichte

DHCP ist ein Netzwerkprotokoll, das von Netzwerkadministratoren verwendet wird, um den Computern in einem Netzwerk automatisch IP-Adressen zuzuweisen und zu verwalten. DHCP wurde ursprünglich von der IETF im Jahr 1993 entwickelt, um die Verwaltung von IP-Adressen in lokalen Netzwerken zu vereinfachen. Es wurde als Ergänzung zu dem älteren Protokoll BOOTP entwickelt, das nicht über ein dynamisches Adressierungssystem verfügte. Im Laufe der Jahre wurde DHCP immer weiterentwickelt, um den Netzwerkadministratoren eine einfachere und schnellere Möglichkeit zu bieten, Netzwerkressourcen zu verwalten. Es wurde zudem zu einem der am meisten verwendeten Protokolle im Internet, da es die Verwaltung von IP-Adressen vereinfacht und Netzwerkadministratoren dabei hilft, ein sicheres Netzwerk aufzubauen. Heutzutage ist DHCP ein essenzieller Bestandteil moderner Netzwerke, da es die Verwaltung von IP-Adressen vereinfacht und Netzwerkadministratoren dabei hilft, das Netzwerk effizient zu verwalten. Es ist auch ein wichtiger Bestandteil des Internets und ermöglicht es Computern, sich an verschiedenen Netzwerken anzumelden und ihre IP-Adressen zu bekommen.

#### Funktion

DHCP ist ein Netzwerkprotokoll, das von einem Server verwendet wird, um anderen Computern, Geräten und Diensten automatisch IP-Adressen und andere Netzwerkkonfigurationen zur Verfügung zu stellen. Es ermöglicht es den Netzwerkgeräten, sich automatisch anzumelden und sofort Netzwerkdienste zu nutzen, ohne dass manuell Einstellungen vorgenommen werden müssen. DHCP ist eine einfache, aber sehr leistungsfähige Möglichkeit, dynamisch Netzwerkkonfigurationen für eine Vielzahl von Geräten und Diensten bereitzustellen.

#### Aufbau

* **op:** Ist, ob man eine Anfrage oder eine Antwort handelt
* **htype:** Netz typ (z.B. 1 Ethernet, 6=IEEE 802 Netzwerk)

Abbildung 1 Zeichnung eines DHCP-Paketes

* **hlen:** Länge einer Netzadresse in Bytes
* **hops:** Anzahl der DHCP-Relay -Agents auf dem Datenpfad
* **xid:** ID von der Verbindung Server/Client
* **secs:** Zeit nach dem Start des Clients in sec
* **flags**: ob Client noch gültige IP-Adresse hat
* **ciaddr**: Client-IP-Adresse
* **yiaddr**: eigene IP-Adresse
* **siaddr**: Server IP-Adresse
* **giaddr**: Relay-Agent IP-Adresse
* **chaddr**: Client MAC-Adresse
* **sname**: Name des DHCP-Servers (optional)
* **file**: Name einer Datei zB. System Kernel (optional)
* **options**: DHCP-Parameter und Options (RFC21329)

#### Zuordnung

Es gibt drei verschiedene DHCP-Zuordnungsmodelle, die da wären, statisch, automatisch und dynamisch Zuordnung.

##### Statische Zuordnung

In diesem Modell werden die IP-Adressen bestimmten MAC-Adressen zugeordnet. Diese Adressen werden der MAC-Adresse auf unbestimmter Zeit zugeteilt. Nachteil dieser Methode ist, dass keine zusätzlichen Clients in das Netz mit eingebunden werden, da die Adressen fest vergeben sind. Diese Zuordnung wird nur dann vorgenommen, wenn der Client Server Dienste ausführen möchte und unter dieser festen IP-Adresse erreichbar sein soll.

##### Automatische Zuordnung

Dort werden IP-Adressen gewissen Bereichen (range) vergeben. Die IP-Adressen werden an die MAC-Adresse von neuen Clients zugewiesen. Diese Zuweisung wird in einer Tabelle festgeschrieben. Diese Zuordnungen sind permanent und werden nicht gelöscht. Vorteil kann sein, dass Hosts immer derselben IP-Adresse zugewiesen werden, die zugewiesene IP-Adresse keinen anderen Host. Nachteil wäre zum Beispiel neue Clients, die sich verbinden möchten, keine IP-Adressen mehr bekommen, wenn der Adressbereich belegt ist, obwohl alte IP-Adressen nicht mehr aktiv benutzt werden.

##### Dynamische Zuordnung

Die Dynamische Zuordnung ist fast dieselbe wie die automatische. Sie unterscheidet sich nur in einem Punkt. Und zwar wie lang eine IP-Adresse an einem Client vergeben werden soll. Wenn diese Abläuft muss der Client sich beim DHCP melden, um eine Verlängerung zu bekommen. Wenn sich der Client nicht meldet, wird diese Adresse wieder frei und ein anderer Client kann sich verbinden. Diese Leihdauer wird in der Netzwerktechnik als Lease-Time bezeichnet.

#### Ablauf

* DHCP**Discover:** Client schickt Broadcast Anfrage um IP-Adressenzuweisung
* DHCP**OFFER:** der DHCP-Server antwortet mit entsprechenden Werten auf das **Discover**
* DHCP**REQUEST**: Der Client fordert eine der angebotenen IP-Adressen, weiters verlängert er auch die Lease-Time vom antwortenden Server
* DHCP**ACK:** Bestätigung durch Server und Übermittlung Konfigurationsparameter die durch Client angefragt worden sind (DHCP**INFORM**)
* DHCP**NAK**: Ablehnung der **REQUEST** des Servers
* DHCP**DECLINE**: Ablehnung durch Client da IP schon verwendet wird
* DHCP**RELASE**: Der Client gibt Konfiguration frei, dass es für andere Clients bereitsteht
* DHCP**INFORM**: Anfrage vom Client für weiter Konfigurationsparameter, weil Client eine Statische IP besitzt.

#### Sicherheit

DHCP kann leicht gehackt werden, weil Clients jeden DHCP-Server akzeptieren. Man kann einen DHCP-Server sehr leicht aktivieren, beispielsweise durch einen WLAN-Router im Auslieferungszustand. Der antwortet möglicherweise schneller als der DHCP-Server und verteilt dadurch ungültige Konfigurationen. Ein Hacker kann eine DHCP Starvation Attack durchführen. Das bedeutet er lässt alle Adressen am Server reservieren, dadurch die Antworten auf weitere Anfragen verhindert und anschließend er selbst als DHCP-Server auftretet. Er kann nun eine sogenannte rogue DHCP Spoofing betreiben, das bedeutet er leitet auf andere DNS-Server um, die auf Endgeräte verweisen, die dann die Kommunikation manipulieren. Ein Angreifer könnte auch ein Denial-of-Service-Angriffe starten. Er müsste nur jedem einzelnen Client ein Subnetz zuweisen, kein Gateway oder auf allen Anfragen die gleiche IP-Adresse verwenden. Er kann auch den Man-in-the-Middle-Angriff durchführen dazu muss der Angreifer falsche Gateway und DNS-Adressen vergeben und ein fremder Router einzuschleusen, der den Datenverkehr mitschneidet. Diese Probleme kann man umgehen mit dem sogenannten Peg DHCP Protokoll.

#### Peg

Peg wurde am 1.April 1998 Internet Engineering Task Force veröffentlicht. Doch Peg wurde anfangs als Aprilscherz angepriesen doch diese Methode fand überraschend Erfolg in der Praxis. Jede IP die zu vergeben ist wird auf eine Wäscheklammer geschrieben, die zur Abholung bereitgehalten wird. Die gültigen Parameter der Netzmaske und die IP-Adresse des Routers werden auf Papier geschrieben und ausgehängt und Kopien werden verteilt. Jeder einzelne Parameter werden auf die einzelnen Geräte separat eingestellt und die Wäscheklammer dann am jeweiligen Netzwerkkabel befestigt. Im die IPs vorzuhalten werden die IP-Adressen als Wäscheleinen gesehen und die Subnetze als Kleiderbügel. Die Definition umfasst, wie die Wäscheklammern und Papier an Brieftauben befestigt werden, um das IPoAC anzuwenden.

### Admin-Benutzer

#### Geschichte

Die Geschichte des Linux-Benutzers geht bis in die 1980er Jahre zurück, als das Betriebssystem Unix erstmals veröffentlicht wurde. Unix wurde von Programmierern entwickelt, die das System zur Steuerung von Computern verwendeten. 1991 veröffentlichte der Programmierer Linus Torvalds den Quellcode für ein neues Betriebssystem namens Linux, das auf dem Unix-Code basierte. Die Anfänge des Linux-Benutzers begannen, als die ersten Entwickler begannen, auf Linux basierende Betriebssysteme zu erstellen. Diese Systeme ermöglichten es Benutzern, ihre Programme und Dateien direkt auf Linux-Computern auszuführen. In den frühen 2000er Jahren wurde Linux immer beliebter, als es auf vielen Desktop-Computern und Notebooks verfügbar wurde. Heute verwenden viele Menschen Linux als ihr primäres Betriebssystem, und es ist auf einer Vielzahl von Geräten verfügbar, von Smartphones bis hin zu Supercomputern. Linux hat sich im Laufe der Jahre enorm weiterentwickelt und ist heute eine sehr leistungsstarke Plattform, die es Benutzern ermöglicht, ihre Ideen zu verwirklichen und neue Anwendungen zu entwickeln. Linux-Benutzer genießen die Freiheit, auf einfache Weise zahlreiche verschiedene Softwarepakete zu installieren und zu konfigurieren, und auf diese Weise ihr System individuell anzupassen. Linux ist eine stabile und sichere Plattform, die es dem Benutzer ermöglicht, eine Vielzahl verschiedener Aufgaben auszuführen.

#### Funktion

Sudo (Superuser do) ist ein Befehl in Linux, der es einem Benutzer ermöglicht, Programme mit Administratorrechten auszuführen. Dies bedeutet, dass der Benutzer bestimmte Programme ausführen kann, die normalerweise nur für Benutzer mit Administratorebene verfügbar sind. Der Befehl wird in der Regel verwendet, um Pakete zu installieren, Systemeinstellungen zu ändern oder Systemdateien zu bearbeiten.

#### VAX-11/750

Der VAX-11/750 ist ein 32-Bit-Computer, der von DEC im Jahr 1977 entwickelt wurde. Er wurde unter anderem für die Verwendung in Forschung und Entwicklung, Wissenschaft und Hochschulbereichen entwickelt. Der VAX-11/750 verwendete die VAX-Architektur und lieferte eine Leistung von 0,75 MIPS bei einer Taktfrequenz von 10 MHz. Der Computer verfügte über 16KB EPROM, 64 KB RAM und ein 8-Bit-Ein-/Ausgabesystem. Der VAX-11/750 war der erste Computer von DEC mit einem 32-Bit-Prozessor und ist als ein Meilenstein in der Geschichte der Computertechnologie anerkannt.

#### BSD

Ist eine Softwarevertrieb des Betriebssystem Unix. Entstanden ist diese Software in dem Jahr 1977 an der „University of California of Berkeley“.

#### AT&Ts

AT&T ist ein US-amerikanisches Telekommunikationsunternehmen, das weltweit tätig ist. Es ist eines der größten Anbieter von Mobilfunkdiensten und bietet auch Festnetz-, Internet- und Fernsehdienste an. AT&T ist einer der größten Anbieter von Telekommunikationsdiensten in den USA und bietet Netzwerke in mehr als 225 Ländern an. Das Unternehmen hat seinen Hauptsitz in Dallas, Texas.

#### System V

System V ist ein Betriebssystem, das ursprünglich von AT&T entwickelt wurde und für die Unix-basierten Computer verwendet wird. Es wurde 1985 veröffentlicht und war das fünfte Unix-basierte Betriebssystem, das von AT&T entwickelt wurde. Es hatte eine Reihe von Verbesserungen gegenüber früheren Versionen, einschließlich eines verbesserten Dateisystems, besserer Netzwerkunterstützung und einer größeren Skalierbarkeit. Es wurde in den folgenden Jahren mit mehreren Erweiterungen verbessert, bevor es schließlich durch die SVR4 ersetzt wurde. Heute wird es immer noch in vielen Unternehmens- und Industrieumgebungen verwendet.

#### GNU

GNU ist eine freie Software, die von der FSF entwickelt wurde. Es ist ein Betriebssystem, das die Grundlage für viele andere Betriebssysteme, darunter Linux, bildet. Der Name GNU ist ein Akronym für "GNU ist nicht Unix". GNU ist ein gemeinfreies Projekt, das die Freiheit für jeden Benutzer schützt. Mit GNU können Benutzer Programme schreiben, ändern und kopieren, ohne dass sie Angst haben müssen, dass sie dafür verklagt werden. Es ist ein sehr nützliches Werkzeug, das den Benutzern viele Freiheiten und Vorteile bietet.

### Mail-Server

#### Geschichte

Die Geschichte der E-Mail-Server geht zurück in die frühen Tage des Internets in den späten 1960er Jahren, als die ersten E-Mail-Systeme entwickelt wurden. Zu dieser Zeit dienten E-Mail-Server als zentrale Verteiler für Nachrichten, die zwischen Nutzern über das Netzwerk übertragen wurden. Im Laufe der Jahre wurden immer mehr E-Mail-Systeme entwickelt und verbessert, was schließlich zu der modernen E-Mail-Architektur führte, die heutzutage verwendet wird. Heute sind E-Mail-Server im Grunde genommen spezialisierte Computer, die speziell für die Verarbeitung und Weiterleitung von E-Mail-Nachrichten konfiguriert sind. Sie arbeiten mit anderen Servern zusammen, um E-Mails zu senden und zu empfangen, und verfügen über eine Reihe von Diensten wie Autorisierung, Authentifizierung und Verschlüsselung. Dank der Verbreitung des Internets sind E-Mail-Server zu einem der am meisten genutzten Dienste im Internet geworden. Sie sind ein unverzichtbarer Bestandteil der modernen Kommunikation und ermöglichen es Nutzern, E-Mails zu versenden, zu empfangen und zu speichern. Dank der fortschrittlichen Technologien, die heutzutage verwendet werden, sind E-Mail-Server sicherer als je zuvor und bieten ein hohes Maß an Zuverlässigkeit und Leistung.

#### Funktion

Der E-Mail-Server ist ein Computerprogramm, dass E-Mails verarbeitet und übermittelt. Es ist für E-Mail-Konten und -Dienste verantwortlich und arbeitet als intermediär zwischen dem E-Mail-Client und dem Internet. Der E-Mail-Server ist für die Authentifizierung des E-Mail-Kontos und die Überprüfung der E-Mail-Adressen zuständig. Er verarbeitet die ausgehenden und eingehenden E-Mails, speichert sie und leitet sie an die entsprechenden Konten weiter. Er ist die zentrale Komponente eines E-Mail-Systems und kann als Teil eines Netzwerks oder als eigenständiger Server betrieben werden. Der Server kann auch IMAP- und POP3-Dienste anbieten, die es den Benutzern ermöglichen, ihre E-Mails zu empfangen und zu verwalten. Der Server kann auch SMTP-Dienste anbieten, die es Benutzern ermöglichen, E-Mails an andere Computer zu senden. Der Server kann auch mit anderen Servern kommunizieren, um E-Mails zwischen Benutzern zu versenden. Je nachdem, welche Art von E-Mail-System der Benutzer hat, kann der Server auch einen Spam-Filter, eine Kontaktliste oder andere Funktionen anbieten.

#### Protokoll Arten

Es gibt drei verschiedene E-Mail-Protokoll Arten, die da wären, POP3, IMAP und SMTP. Die folgenden Arten werden anschließen erklärt.

##### POP3

POP3 ist ein Internetprotokoll zum Abrufen von E-Mails aus einem E-Mail-Konto. Es wurde im Jahr 1984 von J.V.D.Boren entwickelt und ist eine Weiterentwicklung des POP2-Protokolls. POP3 ermöglicht es Benutzern, E-Mails von einem Server herunterzuladen und auf ihrem Computer zu speichern. Wenn ein Benutzer die E-Mails auf einem Server abruft, werden die E-Mails automatisch aus dem Server gelöscht. POP3 ist besonders nützlich, wenn man E-Mails zwischen mehreren Computern oder Geräten synchronisieren möchte. POP3 wird häufig von E-Mail-Diensten wie Google Mail und Microsoft Outlook verwendet. Es wird auch von vielen anderen E-Mail-Clients unterstützt. Viele E-Mail-Dienste verwenden eine Kombination aus POP3 und IMAP, um Benutzern die Möglichkeit zu geben, E-Mails auf mehreren Geräten abzurufen. POP3 hat sich im Laufe der Jahre zu einem zuverlässigen und zuverlässigen E-Mail-Protokoll entwickelt und ist heute ein unverzichtbares Werkzeug für E-Mail-Benutzer.

##### IMAP

IMAP ist ein Standardprotokoll, dass es Benutzern ermöglicht, E-Mails von einem E-Mail-Server zu lesen und zu verwalten. Es wurde im Jahr 1986 von Mark Crispin als Reaktion auf die damalige E-Mail-Technologie POP entwickelt. IMAP stellt eine einfache Möglichkeit zur Verwaltung von E-Mails auf dem Server bereit, was es Benutzern ermöglicht, E-Mails zu speichern, zu löschen und zu verschieben. Es ermöglicht auch das Lesen und Antworten auf E-Mails, ohne sie auf den Computer herunterladen zu müssen. IMAP ist eine bessere Lösung als POP, da es mehr Flexibilität und Kontrolle über E-Mails bietet. Heutzutage wird IMAP von vielen E-Mail-Providern wie Gmail, Yahoo und Outlook unterstützt. Viele Benutzer bevorzugen IMAP, da es Benutzern mehr Kontrolle und Flexibilität über ihre E-Mails bietet. Zudem können Benutzer auf E-Mails von mehreren Geräten aus zugreifen, da IMAP-E-Mails auf dem Server gespeichert werden.

##### SMTP

SMTP ist ein Internetprotokoll, das es Computern ermöglicht, E-Mails zu senden und zu empfangen. Es ist eines der ältesten und am weitesten verbreiteten Internetprotokolle und wird heute noch für viele E-Mail-Anwendungen verwendet. Es wurde ursprünglich 1982 von zwei Forschern an der Universität von Kalifornien in Berkeley entwickelt. Diese ersten Versionen des Protokolls stellten eine einfache Verbindung zwischen zwei Computern her, um E-Mails zu senden und zu empfangen. Im Laufe der Jahre wurde SMTP immer weiter verbessert und angepasst, um die heutige Version zu schaffen, die von vielen E-Mail-Anbietern und E-Mail-Clients unterstützt wird. SMTP-Dienste werden normalerweise von E-Mail-Providern bereitgestellt, die Benutzern ermöglichen, E-Mails zu senden und zu empfangen. SMTP-Dienste sorgen dafür, dass E-Mails über das Internet durchlaufen und an die richtige Adresse gesendet werden. Es gibt auch einige E-Mail-Clients, die SMTP-Funktionen haben, wodurch Benutzer E-Mails direkt von ihrem Computer aus senden und empfangen können. SMTP ist eines der am häufigsten verwendeten Internetprotokolle und ist ein wesentlicher Bestandteil des E-Mail-Verkehrs. Es ist einfach zu bedienen und ermöglicht es Benutzern, E-Mails schnell und sicher zu senden und zu empfangen.

### Python Client

#### Geschichte

Python wurde Ende der 1980er Jahre von Guido van Rossum entwickelt. Van Rossum wollte eine Programmiersprache schaffen, die leicht zu lesen und zu schreiben ist und gleichzeitig mächtig und flexibel ist. Er nannte seine Sprache zuerst "ABC" und benannte sie schließlich in Python um, nach dem britischen Komödienduo Monty Python. Python wurde zuerst als Scripting-Sprache entwickelt, aber es hat sich mit der Zeit zu einer breiten Palette von Anwendungen entwickelt. Es wird häufig für Webentwicklung, Datenanalyse, maschinelles Lernen und andere Aufgaben verwendet. Python ist heute eine der am häufigsten verwendeten Programmiersprachen der Welt, und viele Softwareprojekte basieren auf Python. Es wird in vielen Unternehmen, Universitäten und Schulen als Lehr- und Forschungssprache verwendet.

#### Funktion

Python ist eine hoch entwickelte, objektorientierte Programmiersprache, die für die Verarbeitung von Daten und das Erstellen von Anwendungen verwendet wird. Es ist eine der am weitesten verbreiteten Programmiersprachen und wird in Bereichen wie Datenanalyse, künstliche Intelligenz, Webentwicklung und vielem mehr eingesetzt. Python bietet eine Vielzahl von Funktionen, die es Programmierern ermöglichen, schnell und effizient komplexe Aufgaben zu lösen. Einige der wichtigsten Funktionen von Python sind:

Einfache Syntax: Python-Code ist leicht zu lesen und zu verstehen, was es Programmierern ermöglicht, schneller zu programmieren und komplexe Aufgaben zu lösen.

Objektorientierung: Python-Code kann als objektorientierte Programmierung aufgebaut werden, was es Programmierern ermöglicht, Software effizienter zu entwickeln.

Bibliotheken: Python enthält eine Vielzahl von Bibliotheken, die es Programmierern ermöglichen, auf bereits vorhandene Ressourcen zuzugreifen und komplexe Aufgaben zu lösen.

#### Python vs C/C++

Python ist eine sehr einfache Programmiersprache, die eine schnelle Entwicklung ermöglicht. Das bedeutet, dass man schneller ein funktionsfähiges Programm erstellen kann. C und C++ sind sehr komplexe Sprachen, die eine längere Entwicklungszeit erfordern. Python ist auch sehr portabel, was bedeutet, dass ein Programm, das in Python geschrieben wurde, auf verschiedenen Betriebssystemen und Plattformen problemlos ausgeführt werden kann.

### NCAT

#### Geschichte

Die Geschichte des Ncat beginnt mit dem Schreiben des ersten Codes im Jahr 2001 von der Firma Nmap Project. Der erste öffentliche Release erfolgte im Jahr 2002. Seitdem wurde Ncat kontinuierlich weiterentwickelt und verbessert, um die Sicherheits- und Bedienungsfunktionen zu verbessern. Ncat ist eine Kombination aus netcat, einem Unix-basierten Netzwerk-Swiss-Army-Knife, und dem Datenübertragungsprotokoll SSL/TLS. Mit Ncat können Benutzer Daten über ein Netzwerk senden und empfangen, entfernte Server verwalten und verschiedene Netzwerkverbindungen herstellen. Ncat ist auch für die Verwendung von zentralen Sockets, Remote-Execution und proxying geeignet. Ncat ist ein einzigartiges Werkzeug, da es eine Reihe von Funktionen bietet, die in der Netzwerk- und Systemadministration nützlich sind.

#### Funktion

Ncat ist ein Netzwerk-Tool, mit dem Netzwerkkonnektivität und Kommunikation über IPv4 und IPv6 hergestellt werden kann. Es kann als ein Ersatz für das Telnet-Programm verwendet werden, aber mit vielen zusätzlichen Funktionen. Mit Ncat können Netzwerkverbindungen verschiedener Arten hergestellt werden, darunter Server, Client, UDP und TCP. Es kann auch als ein Port-Scanner, Proxy-Server, Datei-Übertragungs-Tool und als ein Netzwerk-Monitoring-Programm verwendet werden. Ncat kann auch als ein Sicherheits-Tool verwendet werden, um Netzwerk-Verbindungen zu verschlüsseln und Daten vor unerwünschten Zugriffen zu schützen. Es kann auch verwendet werden, um Netzwerk-Protokolle zu testen und zu überprüfen.

### Geschichte der virtuellen Server

Die Anfänge der Server, die schließlich zur Entwicklung virtueller Server führten, waren 1989 in Cern. Der Informatiker Tim Berners-Lee hatte das Ziel, den Informationsaustausch weltweit für jedermann zugänglich zu gestalten. Am 25. Dezember 1990 wurde der weltweit erste Webserver im Namen des US-Amerikanischen Computer- und Serverherstellers NeXT1 veröffentlicht. Dieser Server war mit einer 2 Gigabyte-Festplatte, einer 256-MHz-CPU und einem Graustufenmonitor ausgestattet. 1993 gab es schließlich den nächsten Durchbruch mit den sogenannten Rack-Servern2. Diese bestanden aus mehreren übereinander gestapelten Einbauplätzen. Die Erfindung führte schließlich dazu, dass mehrere Server auf engstem Raum gestapelt werden konnten. Das einzige noch nicht gelöste Problem war die Wärmebildung. Die 2001 auf den Markt gebrachten Blade-Server der Firma RLX-Technologies3 konnten dieses Problem lösen. Zum einen beseitigten sie mehrere Einschränkungen des Rack-Server-Frameworks und zum anderen nutzten Blade Server weniger Komponenten. Dies führte dazu, dass nun weniger Strom verbraucht wurde. Die Teile der beiden Server wurden nun zusammengefügt. Das Blade Gehäuse, dass nun Funktionen wie Kühlung bereitstellen konnte und die Rack-Server, die dann im Blade Gehäuse montiert wurden. Einer von vielen Vorteilen dieser Zusammensetzung war, dass nun die Anzahl der Server in kleinen Räumen merklich gesteigert werden konnte.

2005 stand die effiziente Nutzung der schon bestehenden Serverarten im Vordergrund. Eine Idee dazu waren Server Cluster4. Dadurch konnte, beispielsweise wenn ein Server ausfällt, seine Workload in den nächstgelegenen Server im gleichen Cluster geladen werden. Im gleichen Zeitraum wurde das Remote-Management entwickelt. Das auch Lights-Out-Management (LOM) genannte Tool verschaffte den Vorteil, die Serververwaltung durchzuführen, um ohne physisch auf den Rechner zugreifen zu müssen, die Serververwaltung stattfinden konnte. Diese neue Vorgehensweise ermöglichte eine Prozessoptimierung in der Technik. Ebenfalls wurden nun weniger IT-Techniker/Administratoren benötigt.

2013 entwickelte HPE-Labs den weltweit ersten software-designed Server – Moonshot. Moonshot Server liefen auf energiesparenden Mikroprozessoren, welche ursprünglich für Handys gebaut wurden. 89% weniger Energie, 80% weniger Platz, 77% billiger als bis dahin erhältliche Server. Diese Server wurden dann schlussendlich auch für das Cloud-Computing auf Hochleistung verwendet.

Cloud Computing - Teil der Virtualisierung. Virtuelle Server haben die gleichen Fähigkeiten wie ein „echter“, also Hardware basierter Server. Virtualisierungssoftwares, auch Hypervisor genannt, unterteilen den physischen Server in mehrere kleinen, virtuellen Server.

Nach Installation der Virtualisierungssoftware können je nach Einstellung der virtuellen Maschine Ressourcen und Speicherplatz vom darunter liegenden System abgezogen werden. Vorteile Virtueller Server sind zum Beispiel die einfache Verwaltung, die leichte Skalierbarkeit und die Kostensenkung.

### RFC

Request for Comments ist ein numerisches Dokument, welches von IETF verwaltet wird. Dieses Dokument beinhaltet Protokolle, Konzepte, Methoden und Programme des Internets welche hier beschrieben werden. Das Format des Dokumentes ist ASCII. Das bedeutet, dass es sich hier um reinen Text handelt. Jedes Protokoll, Konzept, jede Methode und jedes Programm sind mit einem eindeutigen Titel und einer Nummer identifizierbar.

Weiters folgen einige nähere Beschreibungen zu RFC-Protokollen um die Beschreibung des DNS vereinfacht zu erklären:

#### RFC 799

Dieses Protokoll beschreibt, wie ein Host eindeutig über eine oder mehrere 32-bit Internet Adressen identifiziert werden kann. Noch eine satz dazu

#### RFC 819

Wurde im August 1928 definiert, dass ein Name aus mehreren Teilen bestehen soll. Das soll heißen, von links nach rechts gelesen wird zuerst der spezifischste Teil des Namens angegeben, und anschließend der allgemeine Teil.

#### RFC 882/883

Paul Mockapetris beschrieb 1982 als Erster das Domaine Name System. RFC 882 und RFC 883 wurden bereits von den zwei neueren RFCs 1034 und 1035 abgelöst. Ebenfalls wurden dazu noch viele weitere Standards hinzugefügt.

#### RFC 920

DNS unterstützt die für zum Beispiel SMTP erforderliche, eindeutige Zuteilung von Netz- und Hostadressen zu Rechnernamen.

Zum besseren Verständnis wird folgendes Beispiel aufgeführt. Aus sicherheitstechnischen Gründen wendet sich der fremde Host an DNS. Das Domain Name System überprüft dann, ob der IP-Adresse des rufenden Rechners ein Domain Name zugeordnet werden kann. Falls dem rufenden Rechner kein passender Domain Name zugeordnet werden kann, wird der Verbindungsaufbau unterbrochen.

### DNS

#### Vorgängermodelle

##### Host.txt

Die früher verwendete host.txt (RFC 226) Datei war damals für wenige hunderte oder tausende Rechner ausreichend. Diese Datei wurde jeden Tag einmal aktualisiert und damit auf den neuesten Stand gebracht. Heute findet man im Ordner C:\Windows\System32\Drivers\etc. immer noch (auch auf Windows10) diese host.txt Datei. Die Datei enthält unter Windows10 keine Einträge, da diese nichtmehr auf der host.txt, sondern am DNS abgespeichert werden.

#### WINS

Ein Bild, das Text enthält.

Automatisch generierte BeschreibungEine heute nichtmehr gebräuchliche Art des DNS ist der WINS. Der Windows Internet Naming Service ist ein veralteter, nur mehr auf älteren Windows Systemen zu findender Dienst. Im Unterschied zum heute herkömmlichem DNS wandelt WINS keine hierarchisch angeordneten Namen in IP-Adressen um. Hier werden NetBIOS Namen umgewandelt. NetBIOS Namen haben keine Struktur. Es wird beispielsweise nicht Angerer.server.at, sondern Angerer eingegeben. Windows arbeitet hier mit dem PDC.

Abbildung 2 Screenshot vom privaten Computer host.txt; Dateipfad C:\Windows\System32\Drivers\etc.

#### DNS

Der Gebrauch der host.txt Datei war damals für die wenigen Tausend Rechner ausreichend. Laut ITU werden im Jahr 2022 voraussichtlich 410 Millionen Computer verkauft werden. Das ergibt rund 13 Stück pro Sekunde. Auf diese Menge an Rechnern kann das Prinzip der host.txt Datei auf Grund der enormen Menge nicht mehr angewendet werden. Hier kommt das Domain Name System zum Einsatz. Ein einziger DNS-Server ist hier nicht ausreichend. Durch den hohen Datentransfer auf einen einzigen Server würde der theoretisch auf dem Application Layer verankerte Service direkt überlasten. Daher das Konzept der Zonen. Einige dieser TLDs sind .com, .at, .org, .net, oder .ch. TLDs bezeichnen den letzten Abschnitt einer Domain, dieser ist die hierarchisch am höchsten gelegene im Namensraum. Jede Zone hat mehrere eigene DNS-Server zur Verfügung. Die österreichische Verwaltung ist im Internet unter <https://www.nic.at> zu finden.

Jede im Internet erreichbare Zone muss zwingend einen DNS-Server betreiben. Hierarchisch an höchster Stelle liegen insgesamt 13 Root-DNS-Server. Diese kennen alle darunter liegenden, jeweils die für eine spezifische Zone zuständigen DNS-Server.

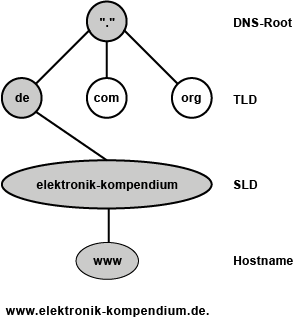


Abbildung 3 Aufbau DNS Server; https://www.elektronik-kompendium.de/sites/net/bilder/09011414.gif

#### Beispiel

Ausgegangen wird, dass der bediente PC angerer.bsp.com heißt. Dieser will auf den Rechner www.server.com zugreifen. Da der PC die IP-Adresse von www.server.com benötigt, schickt er eine Anfrage an den DNS-Server dns.bsp.com. Da die beiden Maschinen in der gleichen Zone liegen und hierarchisch gleichrangig sind, wird dns.bsp.com die Anfrage sofort beantworten können.

Nun will der gleiche PC angerer.bsp.com aber auf eine in einer höheren Zone liegenden Website edelmann.bsp.at zugreifen. Der Server dns.bsp.com hat die IP-Adresse von edelmann.bsp.at nicht gespeichert. Er gibt also die Anfrage an den DNS-Server dns.bsp.at weiter. Wenn die IP-Adresse für edelmann.bsp.at nun gespeichert ist, wird diese über dns.bsp.com an das Ziel weitergeleitet.

### Normaler Benutzer

Das Verwenden eines normalen Benutzers erhöht die Sicherheit im System selbst. Linux löst das Problem der sofortigen Administrator-Rechte beim erstmaligen Einschalten (Windows: erster Benutzer= Administrator), mit dem Sudo Befehl. Das bedeutet nicht, dass der Admin-Status nicht mehr verwendet werden kann, sondern dass man Admin-Rechte nurmehr durch das Eingeben des dann gefragten sudo-Passwortes erlangen kann. Somit bleibt der Benutzer immer ein „normaler Benutzer“ und erlangt root-Rechte nur für von ihm bestimmte Befehle. Siehe 3.2.2 Admin-Benutzer.

Durch den fast dauerhaften „normalen“ Status ohne Administrator-Rechte ist es fast unmöglich, dass sensible Daten oder für den Server unersetzbare Dateien versehentlich gelöscht werden, da hierzu Adminrechte nötig sind. Die standardmäßige Usereinstellung ist insofern sicher, da ein potenzieller Hacker, anders als wenn dieser den Administrator hacken würde, auf beispielsweise die root-Schell nicht so einfach zugreifen kann.

### Active Directory

### Firewall

#### Theoretische Grundlagen

Die grundlegende Aufgabe einer Firewall ist der Schutz des Rechners von außen. Einfach erklärt werden in der Firewall bestimmte Regeln und Filter im Vorhinein definiert und angewandt. So können schädliche Daten frühzeitig entdeckt und abgeblockt werden. Durch den Gebrauch von Firewalls werden Firmen oder auch Privatnutzer vor Viren geschützt. Dadurch kommt es auch zu keinem Datenverlust. Ebenfalls schützt die Firewall vor unbefugten Zugang auf sensible Daten.

In der IT-Geschichte haben sich bis heute fünf wesentliche Firewall-Typen herauskristallisiert, welche in den folgenden Kapiteln genauer beschrieben werden.

##### Paket Filter

Die Software für diesen Filter findet Anwendung bei wichtigen Punkten im System, wie dem Router oder dem Switch. Der Paket-Filter prüft dauerhaft den Datenverkehr der Pakete, die eingehen und ausgehen. Die Kontrolle basiert auf zuvor festgelegten Sicherheitsrichtlinien. Dazu gehören Kriterien wie die Portnummer, IP-Adresse oder Pakettype. Dazu werden Richtung, Ziel- und Quelladresse analysiert. Falls das Gerät Pakete erkennt, die nicht den gewollten Regeln entsprechen, werden sie nicht ins interne Netz weitergeleitet.

Zu erwähnen ist jedoch noch, dass diese Art der Firewall nicht die Sicherste ist. Zum einen können bei der Einrichtung der Sicherheitsregeln viele Fehler passieren und zum anderen schützt er nicht vor Absenderfälschungen, daher wird diese als die schwächste der möglichen Firewall Varianten bezeichnet.

##### Circuit Level Gateway

Die TCP-Dateien im gesamten System werden überwacht. Dadurch ist eine relativ schnelle Identifikation bösartiger Dateien möglich. Es wird festgestellt, ob die geöffnete Sitzung sicher ist. Es wird zwar ermöglicht, dass spezifische Ports oder IP-Adressen gezielt gesperrt werden können, aber die Bestimmung des Paketinhaltes ist nicht möglich.

##### Stateful Inspection Firewall

Stateful Inspection Firewalls analysieren jedes Paket und untersuchen gleichzeitig, ob dieses Paket Teil einer autorisierten TCP-Sitzung ist. In jeder Session werden Port, Quell-/ und Zieladresse überwacht. Nicht autorisierte Pakete werden sofort abgewiesen. Diese Variante ist zwar sicherer, ist aber auch mit einer höheren Belastung der Netzwerkleistung verbunden.

##### Proxy-Firewall

Dieser Sicherheitsmechanismus arbeitet auf dem Application-Layer. Er überwacht, anders als der Paket-Filter, nicht nur spezifisch programmierten Pakets, sondern analysiert den Traffic auf dem Application-Layer. Die Proxy-Firewall agiert zwischen eigenem und fremdem Netzwerk. Anfragen aus dem Internet werden abgefangen, gefiltert und analysiert und anschließend stellvertretend weitergeleitet oder eben abgeblockt. Jedes Protokoll, wie HTTPS, SMTP oder DNS bekommt einen eigenen Filter, der bestimmte Datenkommunikation verbieten und herausfiltern kann. Diese Art der Firewall ist sehr sicher, da das eigene und das fremde Netzwerk quasi nie miteinander direkt verbunden ist.

##### Next-Generation Firewall

Die Next-Generation Firewall, kurz NGFW, beschreibt die dritte Generation der Firewall-Technologie. Um gegen moderne Viren und Bedrohungen zu schützen sind Firewalls, die bestimmte Ports filtern, oft nicht mehr ausreichend. Das ist darauf zurückzuführen, dass Viren oft über Port 80, bzw. Port 443 in das private Netz eindringen. Die Next-Generation Firewall untersucht nicht nur Protokoll und Port, sondern erkennt ungewöhnlichen Datenfluss und filtert abnormale Dateien weg. Next-Generation Firewalls besitzen unter anderem auch Antiviren-, Anti-Spam Filter und ähnliche Funktionen. Durch die zusätzlichen Features werden Extra-Security Gateways, Proxys oder ähnliche Softwaresysteme überflüssig.

## Praktischer Teil

### Praktische Umsetzung

Dieses Kapitel beschreibt die praktische Umsetzung. Zu Beginn wird die VM auf der Software Virtual Box installiert. Anschließend ist auf der VM der Linux Server zu installieren. Danach werden die einzelnen Installations- und die Konfigurationsschritte erklärt [siehe 3.3.2]. Der vollständige, installierte und konfigurierte Server beinhaltet keine grafische Oberfläche, sondern eine command-line. Die Erweiterungen wie zum Beispiel DHCP, DNS, AD, IMAP und Benutzer werden mittels command Befehlen hinzugefügt. Mehr dazu findet man in den jeweiligen Konfigurationsabschnitten. Wenn der Server vollständig läuft und alle Features laufen, geht es an die Softwareprogrammierung. Zuerst sollte der Code UDP basierend geschrieben werden. Durch Recherche und Rücksprache mit dem Partner, wurde sich für TCP entschieden. Aus dem folgenden Grund, da TCP verbindungsorientiert und zuverlässig ist. Verbindungsorientiert heißt so viel, wie dass zuerst eine Verbindung zum Server hergestellt wird. Dann erst werden weitere Anfragen durchgeführt. Es werden keine Daten in falscher Reihenfolge oder Datenpakete verloren gehen wie bei UDP. Das TCP-Protokoll fordert automatisch das Verlorene oder fehlerbehaftete Paket neu an. Um die Arbeit mit dem Diplomarbeitspartner zu vereinfachen, wird eine GitHub Projekt angelegt, wo man verfolgen kann, wer wann am Programmcode gearbeitet hat bzw. was verbessert worden ist. Es werden auch die ganzen einzelnen Dokumentationen hochgeladen. Der Link zum GitHub Projekt befindet sich beim Unterpunkt Quellen. Der Programmcode wird in dem Programm PyCharm Community Edition 2021 2.2 implementiert.

### Server

#### Ubuntu Server Installation

Um den virtuellen Server installieren zu können muss man die Software Oracle VM VirtualBox installieren [VirtualBox herunterladen | heise Download](https://www.heise.de/download/product/virtualbox-40385/download). Wenn das erledigt ist, benötigt man die Installation des Ubuntu Server [Get Ubuntu Server | Download | Ubuntu](https://ubuntu.com/download/server). Dieser Linux Ubuntu ist für Server speziell ausgelegt. Diese Software besitzt keine grafische Oberfläche, wenn man eine Oberfläche benötigt, kann man sich auch für das normale Ubuntu entscheiden. Möchte man die Iso Datei nicht auf eine CD brennen, kann man auch die OpenSource Software Daemon Tools installieren.

#### Installation VirtualBox

Die Exe Datei ausführen und den Speicherort auswählen. Wurde das alles erledigt, ist auf den Button fertig stellen zu klicken und die Software sollte dann so aussehen:

Ein Bild, das Text enthält.

Automatisch generierte Beschreibung

Abbildung 4 Virtual Box Oberfläche

#### Name und Betriebssystem

In diesem Fenster kann man den Namen der neu angelegten virtuellen Maschine festlegen. Der komplette Server wird EdelAng-Server HTL benannt. Dann den Speicherort und das Betriebssystem auswählen mit der gewünschten Version. Mit weiter bestätigen.

Ein Bild, das Text enthält.

Automatisch generierte Beschreibung

Abbildung 5 Name und Betriebssystem für VM auswählen

#### Speichergröße

In dem Neuen geöffneten Fenster kann man den Speicher festlegen. Es wird sich für die empfohlene Größe entschieden 1024MB.

Ein Bild, das Text enthält.

Automatisch generierte Beschreibung

Abbildung 6 Speichergröße von der VM einstellen

#### Festplatte auswählen

Wenn man auf Weiter klickt, kommt ein Fenster, was abfragt, ob man eine neue virtuelle Festplatte, keine Festplatte oder eine vorhandene Festplatte nutzen will. Es wird eine neue Festplatte installiert, da keine vorhanden war.

#### Grundkonfiguration

Wenn man auf Erzeugen klickt, kommt das Fenster Dateityp der Festplatte. Es gibt drei Punkte zur Auswahl VDI, VHD und VMDK. Es wird der 1. Dateityp VDI ausgewählt, da dieser die meisten Features hat. Das wird auch von den Lehrpersonen empfohlen. Dann auf Weiter klicken, ein neues Fenster öffnet sich, und zwar wie die Speicherung erfolgen soll. Entweder kann man dynamisch alloziert oder eine feste Größe auswählen. Dynamisch alloziert nimmt nur so viel Speicherplatz weg wie viel es wirklich benötigt. Die feste Größe nimmt jedoch den Platz von Anfang an weg. Hier wird für das dynamische allozierte ausgewählt. Wieder auf Weiter klicken und das Fenster Dateiname und Größe der Festplatte erscheint. Der Dateiname wird zu EdelAng-Server HTL verändert und die Größe wird auf 10GB eingestellt. Auf den Button Erzeugen klicken und es erscheint die Zusammenfassung der virtuellen Maschine. Jetzt wird die virtuelle Maschine normal gestartet. Es erscheint ein Fenster, wo man die Iso Datei auswählen muss. Dann auf Starten klicken und die Sprache auswählen und mit Enter bestätigen. Dann das Tastaturlayout auswählen und auf Erledigt mit den Pfeil Tasten, anschließend auf Enter drücken. Anschließend öffnet sich ein Fenster was für ein Typ installiert werden soll. Einmal Ubuntu Server, und Ubuntu Server minimized. Das Kreuz wird bei den Normalen gesetzt und dann wieder auf Erledigt gehen und Enter anklicken. Es verlangt nach einer Netzwerkverbindung, damit der PC mit anderen Maschinen kommunizieren kann. Es wird sich für IPv4 entschieden auf Automatisch DHCP gestellt und dann wieder auf Erledigt klicken. Jetzt öffnet sich die Proxy Konfiguration. Dort wird alles unverändert übernommen. Als nächster Konfigurationsschritt wird eine sogenannte Ubuntu-Archiv-Mirror Adresse abgefragt. Dort bleibt wie beim Proxy Server alles unverändert. Weiter geht es mit der Partitionierung. Wir wählen den Punkt Eine ganze Festplatte aus und in LVM-Gruppen aus. Warum LVM? Bei LVM kann ein Dateisystem nachträglich vergrößert werden und es ist ein häufiges Partitionierungsschema im Linux Bereich. Zum Schluss ist eine Zusammenfassung der ganzen Speicherplatzkonfiguration zu sehen.

Ein Bild, das Text enthält.

Automatisch generierte Beschreibung

Abbildung 7 Speicherplatzkonfigurieren am Server

#### Profileinrichtung

Es ist ein Profil mit den Namen der Diplomarbeitsverfasser zu erstellen. Es wird nach einem Namen des Servers gefragt. Das ist fast der gleiche Name wie bei der virtuellen Maschine. Außer dass unterstriche verwendet werden und alles klein geschrieben wird. Der Benutzername ist aus den ersten paar Buchstaben der Nachnamen der Verfasser **Edel**mann und **Ang**erer erstellt worden. Das Passwort wird vom Verfasser gewählt und bleibt geheim.

Ein Bild, das Text enthält.

Automatisch generierte Beschreibung

Abbildung 8 Profil des Server Benutzers anlegen

#### SSH

Man kann bei der Installierung auswählen, ob man ein SSH haben will oder nicht. Es ist auszuwählen, um ein sicheres Kommunizieren zwischen zwei Computer zu garantieren. Bei SSH-Identität wird Nein angekreuzt.

#### Server-Snaps

Jetzt werden beliebte Snaps angezeigt, was in Serverumgebung verwendet, werden können. Nach Recherche und gewissen Diskussionen werden keine Snaps ausgewählt.

#### Installation und Neu starten

Hat man bei den Server Snaps auf Erledigt geklickt so wird die Grundkonfiguration installiert. Dies könnten einige Minuten dauern. Wenn das geschafft ist, kommt ganz unten ein Button Jetzt neustarten.

#### Server Installation fertiggestellt

Der Server wird fertiggestellt und neu gestartet und man kann sich mit dem Benutzernamen und dem Password einloggen. Nunn kann mit dem erweiterten Konfigurieren begonnen werden, DHCP, DNS usw.

Ein Bild, das Text enthält.

Automatisch generierte Beschreibung

Abbildung 9 Server Oberfläche nach dem Neustart

### DHCP-Server

#### Installation des DHCP-Pakets

Wie gehabt startet man die Software Virtuell Box und darauf den drauf laufenden Server. Danach mit den Anmeldedaten anmelden. Im Terminal des Linux Servers das Kommando „sudo apt install isc-dhcp-server“. Der Befehl sudo ist ein Befehl der den Prozess mit den Rechten eines weiteren Benutzers zu Starten. In dem Fall ist es der Benutzer edelang. Wenn man diesen Befehl ausführt, muss man zuerst das Passwort des Benutzers eingeben, um fortzuführen. Apt ist ein Packetmanagment System. Es ist dafür da, um Pakete einfach und schnell zu finden und zu installieren oder deinstallieren und auch das System zu updaten. Der Befehl „install“ ist eigentlich klar, um dieses Datenpaket was man benötigt, hier auf diesem virtuellen System zu installieren. Der letzte Teil „isc-dhcp-server“ wird benötigt, um alle DHCP-Funktionen zu installieren die es im Bereich DHCP gibt. So wird es gemacht:

Ein Bild, das Text enthält.

Automatisch generierte Beschreibung

Abbildung 10 Installation der DHCP-Pakete

#### Konfiguration

DHCP soll in unserem Fall die IP-Adressen Zuweisung zufällig sein. Um das zu gewähren kann man mit Befehl „nano /etc/default/isc-dhcp-server“ erreichen. Um diesen Befehl ausführen zu können muss man zuerst ein Tool herunterladen, was dies ermöglicht. Mit dem Befehl wie angezeigt „sudo apt install embree-tools“. Dann den Befehl “nano /etc/default/isc-dhcp-server“ eingeben. Mit diesem Befehl kommst du in den Nano-Editor, um die Netzwerkinterface zu definieren. Als nächstes muss man unter „sudo nano /etc/dhcp/dhcpd.conf“ die DHCP lease definieren. Unter dem Begriff DHCP Lease ist zu verstehen, weil lange die IP-Adresse dem Client zur Verfügung steht. Diese DHCP-Lease ist der wichtigste Begriff im DHCP-Protokoll. Im Command Fenster sieht das wie folgt an. In dem Editor „authoritative“ nicht auskommentieren, weil das dem DHCP-Server mitteilt, des er für die LAN-Verbindungen zuständig ist. Der Editor sollte so verändert werden das man die „Subnet“ die „option routers“ und „option“ domain-name-servers umstellen. Dann alles wieder mit STRG O abspeichern.

Ein Bild, das Text enthält.

Automatisch generierte Beschreibung

Abbildung 11 DHCP-Lease definieren

#### Funktionstest

Nach der ganzen Konfiguration möchte man natürlich wissen, ob der DHCP-Server aktiv ist und arbeitet. Mit dem Befehl „ip r“ sieht man gelistet die default router, welche als DHCP-Server arbeiten. Man sieht am Screenshot welche IP-Adressen via DHCP vergeben worden sind. Wie zum Beispiel die Adressen: 10.0.2.2, 10.0.2.15, 195.58.161.123 und 212.186.211.21.

Ein Bild, das Text enthält.

Automatisch generierte Beschreibung

Abbildung 12 Funktionstest des DHCP-Servers

### Admin Benutzer

#### Konfiguration

Zuerst am Ubuntu Server als Root-Benutzer anmelden. Als nächsten Schritt erstellt man einen neuen Sudo-Benutzer. Mit dem Befehl „useradd“ erstellt man unter Linux einen neuen Benutzer. Das -m bedeutet, dass ein Home-Verzeichnis erstellt, wird für den neu angelegten Benutzer, sofern dieser noch nicht existiert. -s gibt die Shell für den Login an für den Benutzer. -c ist ein Kommentar, was in der Kontodatei des Benutzers gespeichert wird. Jetzt sollte man als Security Technischen Problemen, ein sicheres Passwort erstellen. Dies funktioniert wie folgt. Mit dem Befehl „passwd“ kann man ein Passwort für den Admin-Benutzer anlegen. Um administrative Aufgaben zu erledigen so muss man den admin zuerst aktivieren mit dem Befehl „sudo“ und mit dem Befehl „usermod“. Mit -a wird der Admin-Benutzer an eine zusätzliche Gruppe dazugeschaltet. -G gibt wiederum die Gruppe an wo er hinzugefügt werden soll.

Ein Bild, das Text enthält.

Automatisch generierte Beschreibung

Abbildung 13 Passwort für Admin anlegen

#### Funktionstest

Um den Admin Benutzer zu testen, muss man mit dem Befehl „su – admin“ den Benutzer wechseln. Danach wird das Passwort für Admin abgefragt und wenn das eingegeben ist, kann man mit dem Neuen Benutzer loslegen. Um zu überprüfen ob der Admin-Benutzer auch Admin-Rechte hat, kann man beispielsweise einen Verzeichnisbaum zu erstellen. Natürlich wieder das Passwort eintippen und der Verzeichnisbaum wird erstellt.

Ein Bild, das Text enthält.

Automatisch generierte Beschreibung

Abbildung 14 Wechsel zum Admin-Benutzer

### Mail-Server

#### Postfix installieren

Um den E-Mail-Server zu konfigurieren, muss man zuerst den Postfix installieren. Doch vorher sollte man mit „sudo apt-get“ update die Pakete aktualisieren. Nachdem eingeben von „sudo apt-get install postfix“ werden die ganzen Pakete für Postfix heruntergeladen.

#### Postfix konfigurieren

Wenn alles installiert ist, muss man den Postfix konfigurieren. Mit dem Befehl „sudo dpkg-reconfigure postfix“. Dann kommt ein Fenster mit Postfix Konfigurationen. Hier wählen wir als erstes den Typen der Mail Konfiguration aus. In diesem Fall wählt man „Internet-Site“ aus. Danach erscheint wieder ein Fenster was uns fragt was man für ein Mail Namen haben will. Danach kommt ein Wurzel- und Postmaster-Mail-Empfänger das auf „root“ setzen. Synchrone Update in der Mailwarteschlange auf Nein. Lokale Netzwerke ist die 127.0.0.0/8. Procemail für die lokale Zustellung verwenden auf Nein stellen. Die Postfachgröße auf 0Bytes setzen. Lokales Adresserweiterungszeichen + hinschreiben. Zu verwendende Internet-Protokolle ist alle anzukreuzen. Nachdem alle Postfix konfigurationen beendet sind, könnte man die Postfix Einstellungen mit dem Befehl „sudo postconf -e ` [new setting] ´ ändern. Um die E-Mail-Verbindungen vor Fremdeinwirkung zu schützen, wird ein selbstsigniertes SSL-Zertifikat erstellt.

Ein Bild, das Text enthält.

Automatisch generierte Beschreibung

Abbildung 15 Name des Mail-Servers

#### SSL-Zertifikat erstellen

Mit den beiden Befehlen “ sudo openssl req -x509 -nodes -newkey rsa:2048 -keyout mailserver.key -out mailserver.crt -nodes -days 365 “, “ sudo openssl req -new -x509 -extensions v3\_ca -keyout cakey.pem -out cacert.pem -days 3650“ wir nichts anders als zwei Dateien erstellt. Einmal die mailserver.key und mailserver.crt. Wenn man diese beiden Befehle eingibt kommen jeweils Fragen, die man beantworten kann, aber wenn man sie nicht beantworten will einfach mit Enter weiter, dann wird diese Frage ausgelassen. Mit dem Befehl „sudo mkdir /etc/postfix/ssl“ wird eine Zertifikatsdatei erstellt und mit den Befehlen „sudo mv mailserver.key /etc/postfix/ssl“, „sudo mv mailserver.crt /etc/postfix/ssl“, „sudo mv cakey.pem /etc/postfix/ssl“, „sudo mv cacert.pem /etc/postfix/ssl“, werden diese Zertifikatsdateien in den Ordner verschoben.

#### SMTP-AUTH. Einrichten

Für die Sicherung des Servers wird, dass Protokoll SMTP-AUTH. Ausgewählt. Mit den folgenden Befehlen kann der Mail-Server gesichert werden. Als nächsten Befehl für die Einrichtung lautet „sudo postconf -e ‚myhostname = patrickedelmann.com‘. Danach wir eine Datei erstellt „sudo nano /etc/postfix/sasl/smtpd.conf“ und diese wird bearbeitet und es wird reingeschrieben „pwcheck\_method: saslauthd, mech\_list: plain login“ Nach dieser Postfix erweiterung muss der Postfix-Daemon gestartet werden und das schafft man mit dem Befehl „sudo systemctl restart postfix“.

Ein Bild, das Text enthält.

Automatisch generierte Beschreibung

Abbildung 16 Postfix Erweiterung und Postfix Daemon starten

#### Installation von SASL

SASL wird von Postfix verwendet, um Authentifizierungen mittels SMTP AUTH durchzulaufen. Da SMTP AUTH installiert wurden, wird jetzt SASL mit dem Befehl „sudo apt-get install libsasl2-2 sasl2-bin libsasl2-modules. Nach dem Installieren wird die Datei „sudo nano /etc/dafault/sasluths“ bearbeitet. Und zwar in der Zeile #Sholud saslauthd run automatically on startup? und das START= stellt man auf Ja. Dnach wir die Datei gespeichert und beendet. Mit dem Befehl „sudo dpkg-statoverride --force --update --add root sasl 755 /var/spool/postfix/var/run/saslauthd” um die dpkg Satus zu aktualiesiern. Für die Datei wird ein Symlink erstellt „sudo ln -s /etc/default/saslauthd /etc/saslauthd“ und dann wird die SASL-Daemon gestartet mit „sudo /etc/init. d/saslauthd start“.

#### Funktionstest mit Telnet

Um postfix zu überprüfen, wird zuerst eine Verbindung zum Server erstellt und ein E-Mail-Protokoll durchgeführt. Zuerst wird Telnet installiert mit „sudo apt-get install telnet“. Wenn Telnet installiert ist, wird mit dem Befehl „telnet localhost 25“ eine Verbindung via SMTP-Port zum Server hergestellt.

Ein Bild, das Text enthält.

Automatisch generierte Beschreibung

Abbildung 17 Output der Verbindung des Mail-Servers

Bei der Ausgabe sieht man, dass der Postfix läuft und eine Postfix Verbindung mit patrickedelmann.com hat. Mit dem Befehl „ehlo localhost“ sieht man folgende Ausgabe:

Ein Bild, das Text enthält.

Automatisch generierte Beschreibung

Abbildung 18 Ausgabe des ehlo localhost

Die Gelbs markierten Ausgaben, sieht man, dass der SMTP AUTH funktioniert. Jetzt kann man den Server sagen, von wem die E-Mail gesendet wurde „mail from: [some-person@some-other-server.com](mailto:some-person@some-other-server.com)“. Dann wird dem Server mitgeteilt an wen die E-Mail versendet werden soll „rcpt to: [root@patrickedelmann.com](mailto:root@patrickedelmann.com)“. Dann mit „data“ kann ein Nachrichtentext versendet werden. Dann die Nachricht eingeben „Patrick Edelmann Schüler der 5AHEL an der HTL-Anichstraße, der Diplomarbeits Meilenstein-E-Mail-Server konfigurieren hat funktioniert. Mein Partner bei der Diplomarbeit ist Simon Angerer ebenfalls Schüler der HTL-Anichstraße. Lg. Edelmann Patrick 5AHEL“. Mit „quit“ kann die Sitzung geschlossen werden. Wenn man alles richtig eingegeben hat, sollte diese Nachricht mit dem Befehl „ll /root/Maildir/new“ angezeigt werden.

Ein Bild, das Text enthält.

Automatisch generierte Beschreibung

Abbildung 19 Ausgabe der E-Mail die versendet wurde

### Python Client Programm

#### Konzept

Ziel: Unser Ziel ist es, ein Python-Programm zu entwickeln, mit dem sich ein virtueller Linux Ubuntu-Server kommunizieren lässt.

Voraussetzungen: Ein Linux Ubuntu-Server soll vorhanden sein. Der Server muss über eine Netzwerkverbindung mit dem Client (dem Computer, auf dem das Python-Programm läuft) verbunden sein. Eine SSH-Verbindung zwischen dem Client und dem Server ist erforderlich. Python-Programmiersprache muss installiert sein.

Funktionalität: Der Benutzer muss in der Lage sein, über das Python-Programm Befehle an den Server zu senden und mit diesem zu kommunizieren. Das Programm sollte in der Lage sein, Dateien auf dem Server zu lesen, zu schreiben und zu bearbeiten. Die Kommunikation zwischen dem Client und dem Server muss über SSH erfolgen. Das Programm sollte auch in der Lage sein, verschiedene Arten von Daten zu empfangen und zu senden.

Testen: Die Funktionalität des Programms muss auf dem Diplomarbeits-Server getestet werden. Der Server muss die oben genannten Voraussetzungen erfüllen. Eine Reihe von Testszenarien muss erstellt werden, um das Programm zu testen. Die Ergebnisse der Tests müssen überprüft und analysiert werden, um sicherzustellen, dass das Programm wie erwartet funktioniert.

#### Funktion

Dieser Programmcode erstellt eine Verbindung zu einem Server mit der IP-Adresse "127.0.0.1" und dem Port 23456. Es wird eine Nachricht über "Hallo hier ist Patrick Edelmann von der HTL-Anichstraße 5AHEL" versendet und die Antwort des Servers über die angegebenen 1024 Bytes empfangen. Die Nachricht und die Antwort werden in cp850 codiert und dekodiert. Schlussendlich wird die Verbindung geschlossen.

#### Programmcode als Text

|  |
| --- |
| #Autor: Edelmann Patrick  #Datum der Finalisierung: 29.12.2022  #Klasse:5AHEL  # Dieses Programm erstellt einen Socket und versucht, eine Verbindung zu einem Server mit der IP-Adresse 127.0.0.1 und dem Port 23456 herzustellen.  #Wenn die Verbindung erfolgreich ist, sendet es eine Nachricht mit dem Namen des Absenders an den Server, der dann eine Antwort zurückgibt.  #Anschließend wird die Verbindung geschlossen.  import socket # importiere das socket modul  host = "127.0.0.1"  port = 23456  s=socket.create\_connection(("localhost",port))  s.send("Hallo hier ist Patrick Edelmann von der HTL-Anichstraße 5AHEL".encode("cp850")) # schreiben der Daten die an Server geschickt werden sollen  bytes=s.recv(1024) # Daten die empangen werden  print("Response from server:",bytes.decode("cp850")) #empfangene Daten dekodieren und ausgeben  s.close() #Verbindung wird geschlossen wenn die Übertragung dieser Nachricht beendet ist |

#### Ausgabe

Es wurde eine ncat.exe Datei verwendet. Die Datei per Drag and Drop auf den Virtuellen Server hineinscheiben. Danach hinter dem Link -v-l Port Nummer hinschreiben und auf Enter drücken. Jetzt hört der Server auf diesen Port, bis das Programm gestartet wird. Mehr dazu bei der Funktion des Codes.

Ein Bild, das Text enthält.

Automatisch generierte Beschreibung

Abbildung 20 Server Ausgabe von dem Programmcode

### DNS-Server

Nach Inbetriebnahme des Servers wird zuerst der Befehl “sudo apt install bind9” ausgeführt. “Sudo” führt einen Befehl mit Root-Rechten aus. Vor jeder Ausführung eines Befehls mit Root-Rechten wird das Passwort für den jeweiligen Admin-Benutzer abgefragt. Der Zusatz “apt” (Advanced Package Tool) ist für die vereinfachte und schnelle Installation von Packages verantwortlich. Es basiert auf Debian und Ubuntu basierenden Distributionen und ist das Dienstprogramm, welches für die Paketverwaltung zuständig ist. Dieser Befehl kann in der Befehlszeile mit “apt [comand] [package]” ohne weiteres verwendet werden. Da doch die meisten Befehle auf der Systemebene passieren, ist der vorher genannte Befehl “sudo” wichtig. Insbesondere “apt install” installiert in diesem Fall bind9. Bind ist ein Open-Source-Programmpaket, verwendet für die Namensauflösung im DNS. BIND9 kann verschieden verwendet werden. Als Caching Nameserver oder auch als Primary, oder Secondary Server, um nur ein paar der Möglichkeiten zu nennen.

Nach der Installation dieses Pakets wird mit “sudo apt install dnsutils” ein hilfreiches Tool installiert, welches gern für Troubleshooting und Testen von DNS Problemen verwendet wird. “Dnsutils” ist üblicherweise schon vorinstalliert. Bei diesem Server war das nicht der Fall, daher wird die erstmalige Installation mit dem genannten Befehl durchgeführt.

**🡪REST ZU DNS KOMMT NOCH**

### Normaler Benutzer

Nach Login in die Virtuelle Maschine und den Benutzer edelang, kann mit dem Befehl “sudo adduser [username]” ein neuer Benutzer angelegt werden. Hier zu beachten ist, dass wieder der Befehl mit “sudo” ausgeführt werden muss. Der Befehl lautet mit meinen Angaben “sudo adduser angerer”. Da keine Option wie “–system” oder “–uid ID” miteingegeben wird, bzw. nicht erwünscht ist, wird automatisch ein normaler Benutzer angelegt. Nach der Eingabe des Passworts für edelang wird der Benutzer initialisiert. Abb.: 21 zeigt, dass ein neuer User mit “group angerer” erstellt wird. Ebenfalls wird das Homeverzeichnis /home/angerer erstellt.

Danach wird nach einem neuen Passwort für den User angerer gefragt. Nach Eingabe persönlicher Daten wird nach Korrektheit der Daten gefragt, dies ist mit y zu bestätigen.

Ein Bild, das Text enthält.

Automatisch generierte Beschreibung

Abbildung 21 normaler Benutzer; Initialisierung

#### Funktionstest

Kontrolliert wird das erfolgreiche Ergebnis mittels Logins mit dem Benutzer angerer. Dazu wird bei der Benutzerabfrage nicht mehr mit edelang eingestiegen, sondern mit angerer. Nach Eingabe des Passworts ist zu erkennen, dass nun der normale Benutzer voll funktionsfähig ist.

Ein Bild, das Text enthält.

Automatisch generierte Beschreibung

Abbildung 22 normaler Benutzer; Login

Erkennbar durch den Erfolgreichen Login und „angerer@edelang:~$“.

### Active Directory

Vor etwaigen Änderungen am Server wird der Befehl “sudo apt update” ausgeführt, um diesen auf den neuesten Stand zu bringen. Wie immer wird auch hier der “sudo“ Befehl immer vorangeschrieben, um als Admin zu agieren. Anschließend mit “sudo hostnamectl set-hostname myubuntu.example.com” den Serverhostname hinzufügen. Die Richtigkeit des konfigurierten DNS kann mit cat /etc/resolv.conf nachgeprüft werden.

Ein Bild, das Text enthält.

Automatisch generierte Beschreibung

Abbildung 23 Hostname Konfirmation

Nun müssen mit dem Befehl “sudo apt -y install realmd libnss-sss libpam-sss sssd sssd-tools adcli samba-common-bin oddjob oddjob-mkhomedir packagekit” die weiters benötigten Packages heruntergeladen werden. Ohne diese ist das Aktivieren des Active Directory nicht möglich.

REALM-NAME UND DOMAIN NAME SIND AD1.ANGERER.COM!!(?)

Die Instruktion “sudo realm discover” gibt die soweit komplette domain Konfiguration, und benötigte Packages welche benötigt werden an. Hinter diesen Befehl muss der zu analysierende Hostname gesetzt werden, also “ad1.angerer.com”. Die Abbildung zeigt unter anderem, dass unter den required packages die zuvor heruntergeladenen Packages auch zu finden sind. Um nicht nur nach einer spezifischen realm suchen zu müssen kann auch der Befehl “realm list” ausgeführt werden. Dieser zeigt alle realm Details an.

Ein Bild, das Text enthält.

Automatisch generierte Beschreibung

Mit “sudo pam-auth-update" werden nun die aktivierbaren PAM-Profile angezeigt. “Activate mkhomedir” wird mit einem Stern markiert, also auch enabled. Mit <OK> die Anwendung schließen. Wann immer eine Änderung am SSSD vorenommen wird, muss ein Neustart durchgeführt werden. Da das durch den vorherigen Befehl der Fall ist, wird mit “sudo systemctl restart sssd” das SSSD neugestartet.

|  |
| --- |
| Systemctl status sssd àad user info sollte sehen |

Ein Bild, das Text enthält.

Automatisch generierte Beschreibung

Jetzt kann Zugriff auf den Server mittels Permission oder Deny vergeben werden. Hierbei ist der Befehl “sudo realm permit [user1] [user2]…” auszuführen. Für den gegenteiligen Befehl muss lediglich das “permit” mit einem “deny“ ausgewechselt werden. User1 und User2 symbolisieren hierbei rein fiktive, mögliche User. Für angelegte groups kann der Befehl “sudo realm permit -g [user1]” verwendet werden. Durch dieses Kommando wird das File sssd.conf abgeändert. Im genannten Fall wird mit dem Befehl “sudo realm permit –all” allen Benutzern der gleiche Status vergeben.

Ein Bild, das Text enthält.

Automatisch generierte Beschreibung

Abbildung 24 Statusvergabe AD

Jetzt haben alle Benutzer Zugriff auf den Server, aber noch keine Rechte, mit “sudo” zu agieren. Dazu wird ein Permission-File mit “sudo vim /etc/sudoers.d/domain\_admins” angelegt. In dieses werden alle Benutzer, denen Zugriff auf sudo nicht verweigert sein soll, eingetragen. Das geschieht mit user1@ad1.angerer.com ALL=(ALL) ALL. So müssen alle User einzeln in das File eingetragen werden.

ENDE VON LINK STEP 8 TEST SSH ACCESS PROBIEREN

### Firewall

Moderne Linux Kernel werden standardmäßig mit dem so genannten Paket-Filtering ausgestattet. Dieser entscheidet, welcher Traffic an den Server weitergeleitet wird, und welcher abgeblockt wird.

Das Standard Firewall System für Ubuntu ist ufw (Uncomplicated Firewall). Diese Firewall wird mit “sudo ufw enable” eingeschalten, bzw. mit “sudo ufw disable” ausgeschalten. Port 22 (SSH) wird mit “sudo ufw allow 22” konfiguriert. Um die Liste dieser erlaubten Ports zu nummerieren, kann der Befehl auch als “sudo ufw insert 1 allow 80” (am Beispiel Port 80) geschrieben werden.

Gleich wie eine Regel hinzugefügt wird, könnte sie durch den Befehl “sudo ufw deny 22” geschlossen, oder mit “sudo ufw delete deny 22” gelöscht werden. Das wurde für die in der Diplomarbeit angeführten VM nicht gemacht.

Der Befehl “sudo ufw –dry-run allow http” ermöglicht es die Auswirkung des Kommandos “sudo ufw allow http” trocken (also theoretisch) durchzuspielen.

In weiterer Folge werden drei verschiedene Varianten in Form von Abbildungen dargestellt, den Firewall Status anzeigen zu lassen.

1. sudo ufw status:

Ein Bild, das Text enthält.

Automatisch generierte Beschreibung

Abbildung 25 Firewall; sudo ufw status

* Es werden die wichtigsten Daten zum Status der Firewall angezeigt.

1. sudo ufw status verbose:

Ein Bild, das Text enthält.

Automatisch generierte Beschreibung

Abbildung 26 Firewall; sudo ufw status

* Detaillierte Auflistung der Firewall-Daten.

1. sudo ufw status numbered:

Ein Bild, das Text enthält.

Automatisch generierte Beschreibung

Abbildung 27 Firewall; sudo ufw status numbered

🡪Zeigt die Liste gleich wie Punkt a) an, mit dem Unterschied, dass die Tabelle aufgelistet wird.

# Erklärung der Eigenständigkeit der Arbeit

EIDESSTATTLICHE ERKLÄRUNG

Ich erkläre an Eides statt, dass ich die vorliegende Arbeit selbständig und ohne fremde Hilfe verfasst, andere als die angegebenen Quellen und Hilfsmittel nicht benutzt und die den benutzten Quellen wörtlich und inhaltlich entnommenen Stellen als solche erkenntlich gemacht habe. Meine Arbeit darf öffentlich zugänglich gemacht werden, wenn kein Sperrvermerk vorliegt.

|  |  |  |
| --- | --- | --- |
| Ort, Datum |  | Simon Angerer |

|  |  |  |
| --- | --- | --- |
| Ort, Datum |  | Patrick Edelmann |

1. Abbildungsverzeichnis

[Abbildung 1 Zeichnung eines DHCP-Paketes 4](file:///C:\Users\pedel\OneDrive\Dokumente\HTL\5AHEL\Diplomarbeit\Diplomarbeit.docx#_Toc129326465)

[Abbildung 2 Screenshot vom privaten Computer host.txt; Dateipfad C:\Windows\System32\Drivers\etc. 18](file:///C:\Users\pedel\OneDrive\Dokumente\HTL\5AHEL\Diplomarbeit\Diplomarbeit.docx#_Toc129326466)

[Abbildung 3 Aufbau DNS Server; https://www.elektronik-kompendium.de/sites/net/bilder/09011414.gif 19](#_Toc129326467)

[Abbildung 4 Virtual Box Oberfläche 26](#_Toc129326468)

[Abbildung 5 Name und Betriebssystem für VM auswählen 27](#_Toc129326469)

[Abbildung 6 Speichergröße von der VM einstellen 27](#_Toc129326470)

[Abbildung 7 Speicherplatzkonfigurieren am Server 29](#_Toc129326471)

[Abbildung 8 Profil des Server Benutzers anlegen 30](#_Toc129326472)

[Abbildung 9 Server Oberfläche nach dem Neustart 31](#_Toc129326473)

[Abbildung 10 Installation der DHCP-Pakete 32](#_Toc129326474)

[Abbildung 11 DHCP-Lease definieren 33](#_Toc129326475)

[Abbildung 12 Funktionstest des DHCP-Servers 34](#_Toc129326476)

[Abbildung 13 Passwort für Admin anlegen 35](#_Toc129326477)

[Abbildung 14 Wechsel zum Admin-Benutzer 35](#_Toc129326478)

[Abbildung 15 Name des Mail-Servers 37](#_Toc129326479)

[Abbildung 16 Postfix Erweiterung und Postfix Daemon starten 38](#_Toc129326480)

[Abbildung 17 Output der Verbindung des Mail-Servers 39](#_Toc129326481)

[Abbildung 18 Ausgabe des ehlo localhost 39](#_Toc129326482)

[Abbildung 19 Ausgabe der E-Mail die versendet wurde 39](#_Toc129326483)

[Abbildung 20 Server Ausgabe von dem Programmcode 42](#_Toc129326484)

[Abbildung 21 normaler Benutzer; Initialisierung 44](#_Toc129326485)

[Abbildung 22 normaler Benutzer; Login 45](#_Toc129326486)

[Abbildung 23 Hostname Konfirmation 46](#_Toc129326487)

[Abbildung 24 Statusvergabe AD 48](#_Toc129326488)

[Abbildung 25 Firewall; sudo ufw status 49](#_Toc129326489)

[Abbildung 26 Firewall; sudo ufw status 50](#_Toc129326490)

[Abbildung 27 Firewall; sudo ufw status numbered 50](#_Toc129326491)

[Abbildung 28 Gantt-Diagramm Edelmann Patrick 57](file:///C:\Users\pedel\OneDrive\Dokumente\HTL\5AHEL\Diplomarbeit\Diplomarbeit.docx#_Toc129326492)

1. Tabellenverzeichnis

[Tabelle 1 Arbeitsnachweis der Diplomarbeit; Angerer Simon 61](#_Toc129305518)

[Tabelle 2 Arbeitsnachweis der Diplomarbeit; Edelmann Patrick 63](#_Toc129305519)

1. Literaturverzeichnis

1. **Edelmann, Patrick.** GitHub. [Online] [Zitat vom: 18. März 2023.] https://github.com/Patrick2345564/Diplomarbeit-.git.

2. **Torsten, Franz.** wiki.ubuntuuser. [Online] [Zitat vom: 19. Oktober 2022.] https://wiki.ubuntuusers.de/APT/.

3. —. wiki.ubuntuuser. [Online] [Zitat vom: 29. November 2022.] https://wiki.ubuntuusers.de/ISC-DHCPD/.

4. —. wiki.ubuntuuser. [Online] [Zitat vom: 15. November 2022.] https://wiki.ubuntuusers.de/snap/.

5. —. wiki.ubuntuuser. [Online] [Zitat vom: 25. Oktober 2022.] https://forum.ubuntuusers.de/topic/fehler-in-der-isc-dhcp-server-conf/#post-7081483.

6. **Triebel, Jonas.** Computerwoche. [Online] [Zitat vom: 1. November 2022.] https://www.tecchannel.de/a/vier-tricks-gegen-linux-netzwerkprobleme,2075503.

7. **Vog, Monika.** foerdermittel-wissenswert. [Online] [Zitat vom: 8. September 2022.] https://foerdermittel-wissenswert.de/so-schreibst-du-ein-konzept/.

8. **Wienströer, Stefan.** a-coding-project. [Online] [Zitat vom: 14. Dezember 2022.] https://www.a-coding-project.de/ratgeber/virtualisierung/ubuntu-server-installation-auf-virtualbox.

9. **Mrugalski, Tomek.** internet system consortium. [Online] [Zitat vom: 11. Dezember 2022.] https://www.isc.org/dhcphistory/.

10. **Brehm, Till.** How to Forge. [Online] [Zitat vom: 23. Jänner 2023.] https://www.howtoforge.de/anleitung/wie-man-einen-dhcp-server-unter-ubuntu-20-04-installiert-und-konfiguriert/.

11. **Kadelke, Markus.** Ionos. [Online] [Zitat vom: 15. Oktober 2022.] https://www.ionos.at/digitalguide/e-mail/e-mail-technik/postfix-mail-server-mit-dovecot-und-squirrelmail-auf-ubuntu/.

12. **Dancuk, Milicia.** phoenixnap. [Online] [Zitat vom: 16. September 2022.] https://phoenixnap.com/kb/useradd-vs-adduser.

13. **Hatem, Naguib.** Baracuda. [Online] [Zitat vom: 24. November 2022.] https://de.barracuda.com/support/glossary/next-generation-firewall.

14. **Franz, Torsten.** Ubuntu. [Online] [Zitat vom: 14. Dezember 2022.] https://manpages.ubuntu.com/manpages/xenial/man8/adduser.8.html?\_ga=2.56716458.1281046089.1678049659-1113399391.1672302534.

15. **Teshome, Lirenso.** wachemo-elearning.net. [Online] [Zitat vom: 14. September 2022.] https://wachemo-elearning.net/courses/information-assurance-and-security/lessons/chapter-four-review-of-shared-key-cryptography-and-hash-functions/topic/firewalls/.

16. **Franz, Torsten.** ubuntu.com. [Online] [Zitat vom: 29. Oktober 2022.] https://manpages.ubuntu.com/manpages/xenial/man8/adduser.8.html?\_ga=2.56716458.1281046089.1678049659-1113399391.1672302534.

1. Abkürzungs- und Symbolverzeichnis

AD *Active Directory*

admin *Administrator*

Apt *advanced packaging tool*

AT&T *American Telephone and Telegraph Company*

BOOTP *Bootstrap Protocol*

BSD Berkley Software Distribution

CD *Compact Disc*

DEC *Digital Equipment Corporation*

DHCP Dynamic Host Configuration Protocol

DNS *Domian Name System*

erkläre *gundi*

Exe *Executable*

FSF *Free Software Foundation*

GB *Gigabyte*

IEEE *Institute of Electrical and Electronics Engineers*

IETF *Internet Engineering Task Force*

IMAP *Internet Message Access Protocol*

IPoAC *Internet Protocol over Avian Carriers*

IPv4 *Internet Protocol Version 4*

IPv6 *Internet Protocol Version 6*

ITU *Internationale Fernmeldeunion*

LVM Logical Volume Manager

MAC *Media Access Control Adresse*

MB *Megabyte*

Ncat *Network Concatenation*

Nmap *Network Mapper*

PDC *Primary Domain Controller*

POP3 *Post Office Protocol 3*

Quellen *buch*

RFC *Request for Commends*

SMTP *Simple Mail Transfer Protocol*

SSH *Secure Shell*

SSL *Secure Sockets Layer*

STRG *Steuerung*

Sudo *super user do*

SVR4 *Sytsem V Release 4*

TCP *Transmission Control Protocol*

TLD *Top Level Domain*

TLS *Transport Layer Security*

UDP *User Datagram Protocol*

Unix *Uniplexed Information and Computing Service*

VAX *Virtual Address Xtension*

VDI *Virtual Disk Image*

VHD *Virtual Hard Disk*

VM *Virtual Machine*

VMDK *Virtual Machine Disk*

1. Anhang

## Schlussfolgerung / Projekterfahrung

Während unserer Diplomarbeit hatten wir die Möglichkeit, an einem spannenden Projekt zu arbeiten, bei dem ein virtueller Ubuntu-Server installiert und konfiguriert wurde. Dabei wurden verschiedene Modifikationen wie DHCP, DNS und Firewalls hinzugefügt, um eine sichere und zuverlässige Serverumgebung zu schaffen.

Ein weiterer Teil des Projekts war es, ein Client-Programm in Python zu entwickeln, das mit dem Server kommunizieren konnte. Hierbei war es besonders wichtig, die Kommunikation zwischen dem Client und dem Server sicherzustellen und die Datenübertragung zu verschlüsseln.

Während der Entwicklung mussten wir verschiedene Herausforderungen meistern, wie beispielsweise die Konfiguration der Firewall-Regeln, sowie die Installation und Konfiguration der DNS-Server.

Insgesamt war die Arbeit an diesem Projekt eine äußerst wertvolle Erfahrung, die uns nicht nur fortgeschrittene Kenntnisse in der Konfiguration von Servern und Netzwerken vermittelt hat, sondern auch unsere Fähigkeiten in der Programmierung und im Projektmanagement verbessert hat.

Durch die erfolgreiche Umsetzung des Projekts konnten wir unserer Diplomarbeit erfolgreich abschließen und haben wertvolle praktische Erfahrungen gesammelt, die uns bei unserer zukünftigen beruflichen Laufbahn von Nutzen sein werden.

1. Projektterminplanung

## Simon Angerer

## Patrick Edelmann

Abbildung 28 Gantt-Diagramm Edelmann Patrick

1. Arbeitsnachweis Diplomarbeit

Tabelle 1 Arbeitsnachweis der Diplomarbeit; Angerer Simon

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Angerer Simon** | | | | |
| **Datum** | **Uhrzeit** | **Stunden** | **Beschreibung** | **Betreuer** |
| 25.02.2022 | 09:30-11:00 | 01:30 | Themenfindung; Diplomarbeitsdatenbank Info, 1. Besprechung | Schiechtl Kevin, Dipl.-Ing. (FH) |
| 15.04.2022 | 12:30-15:00 | 02:30 | Überlegung Projekt | Schiechtl Kevin, Dipl.-Ing. (FH) |
| 19.05.2022 | 11:40-13:20 | 01:40 | Besprechung über Zielvorstellung und Ergebnis | Schiechtl Kevin, Dipl.-Ing. (FH) |
| 20.05.2022 | 15:00-17:00 | 02:00 | Brainstorming Zielvorstellung | Schiechtl Kevin, Dipl.-Ing. (FH) |
| 01.06.2022 | 14:10-15:20 | 01:10 | Überlegung Projektablauf und Zeitplan | Schiechtl Kevin, Dipl.-Ing. (FH) |
| 13.09.2022 | 16:00-18:45 | 02:45 | Überlegung eigener Zielvorstellung und Einteilung | Schiechtl Kevin, Dipl.-Ing. (FH) |
| 14.09.2022 | 13:30-15:20 | 01:50 | Notitzerstellung, Bild eigenen Arbeitsaufwandes machen | Schiechtl Kevin, Dipl.-Ing. (FH) |
| 16.09.2022 | 12:00-12:45  13:20-14:00 | 01:25 | Recherche alte Diplomarbeiten Bibliothek | Schiechtl Kevin, Dipl.-Ing. (FH) |
| 17.09.2022 | 07:40-10:20 | 01:20 | Info Diplomarbeiten auf www.diplomarbeiten-bbs.at | Schiechtl Kevin, Dipl.-Ing. (FH) |
| 19.09.2022 | 12:35-15:40 | 02:55 | DNS erste Vorschläge und Überlegungen | Schiechtl Kevin, Dipl.-Ing. (FH) |
| 23.09.2022 | 15:50-19:00 | 03:10 | DNS-Informationen sammeln, YouTube/Internet, etc. | Schiechtl Kevin, Dipl.-Ing. (FH) |
| 02.10.2022 | 08:00-012:45 | 04:45 | Recherche DNS-Theoretische Grundlagen | Schiechtl Kevin, Dipl.-Ing. (FH) |
| 05.10.2022 | 12:30-17:30 | 05:00 | Recherche und Doku DNS-Theoretische Grundlagen angefangen | Schiechtl Kevin, Dipl.-Ing. (FH) |
| 10.10.2022 | 09:05-10:30 | 01:25 | Praktischer Teil - DNS Konfiguration 1. Fehlversuch | Schiechtl Kevin, Dipl.-Ing. (FH) |
| 17.10.2022 | 12:00-15:30 | 03:30 | Praktischer Teil - DNS Konfiguration 2. Versuch | Schiechtl Kevin, Dipl.-Ing. (FH) |
| 19.10.2022 | 10:15-11:00 | 00:45 | Admin Nutzer/ normaler Nutzer Unterschiede Notizen | Schiechtl Kevin, Dipl.-Ing. (FH) |
| 22.10.2022 | 13:55-15:00 | 01:05 | Vorbereitung normalen Benutzer anzulegen/ Informationen sammeln | Schiechtl Kevin, Dipl.-Ing. (FH) |
| 09.11.2022 | 11:00-12:30 | 01:30 | Praktischer Teil - Anlegen normaler Benutzer | Schiechtl Kevin, Dipl.-Ing. (FH) |
| 14.11.2022 | 17:00-22:30 | 05:30 | Active Directory über Windows - Recherche | Schiechtl Kevin, Dipl.-Ing. (FH) |
| 17.11.2022 | 14:00-17:15 | 02:45 | Recherche Samba | Schiechtl Kevin, Dipl.-Ing. (FH) |
| 20.11.2022 | 13:50-16:20 | 02:30 | Notizen/Themensuche - Samba/AD | Schiechtl Kevin, Dipl.-Ing. (FH) |
| 21.11.2022 | 10:00-12:30 | 02:30 | Praktischer Teil - Samba; Fehlversuch | Schiechtl Kevin, Dipl.-Ing. (FH) |
| 27.11.2022 | 08:00-12:00 | 04:00 | Recherche Firewall | Schiechtl Kevin, Dipl.-Ing. (FH) |
| 01.12.2022 | 09:00-16:45 | 07:45 | Stadtbibliothek, Information technischer Literatur | Schiechtl Kevin, Dipl.-Ing. (FH) |
| 04.12.2022 | 16:00-20:45 | 04:45 | Recherche und Doku Firewall Theoretische Grundlagen Anfang | Schiechtl Kevin, Dipl.-Ing. (FH) |
| 07.12.2022 | 12:15-14:00 | 01:45 | Praktischer Teil - Firewall | Schiechtl Kevin, Dipl.-Ing. (FH) |
| 10.12.2022 | 10:15-11:00 | 00:45 | Fehlersuche Samba/ Recherche Active Directory andere Vorgehensweise | Schiechtl Kevin, Dipl.-Ing. (FH) |
| 12.12.2022 | 13:20-17:45 | 04:25 | Doku Active Directory Theoretische Grundlagen Anfang | Schiechtl Kevin, Dipl.-Ing. (FH) |
| 23.12.2022 | 09:00-09:40 | 00:40 | Praktischer Teil Active Directory Vorbereitung | Schiechtl Kevin, Dipl.-Ing. (FH) |
| 29.12.2022 | 08:30-13:15 | 04:45 | Recherche und Praktischer Teil Active Directory | Schiechtl Kevin, Dipl.-Ing. (FH) |
| 02.01.2023 | 09:30-14:40 | 05:10 | Doku praktischer Teil: Firewall, normaler Benutzer | Schiechtl Kevin, Dipl.-Ing. (FH) |
| 03.01.2023 | 17:00-19:45 | 02:45 | Doku praktischer Teil: Firewall, normaler Benutzer | Schiechtl Kevin, Dipl.-Ing. (FH) |
| 05.01.2023 | 13:20-17:45 | 04:25 | Doku praktischer Teil Active Directory Ende | Schiechtl Kevin, Dipl.-Ing. (FH) |
| 05.01.2023 | 09:00-15:45 | 06:45 | Doku Theoretische Hintergründe DNS | Schiechtl Kevin, Dipl.-Ing. (FH) |
| 06.01.2023 | 08:45-12:15 | 03:30 | Doku theoretischer Hintergründe Active Directory | Schiechtl Kevin, Dipl.-Ing. (FH) |
| 08.01.2023 | 07:45-11:30 | 03:45 | Doku theoretischer Hintergründe Firewall | Schiechtl Kevin, Dipl.-Ing. (FH) |
| 15.01.2023 | 14:00-15:20 | 01:20 | Doku theoretischer Hintergründe normaler Benutzer | Schiechtl Kevin, Dipl.-Ing. (FH) |
| 20.01.2023 | 09:15-10:00 | 00:45:00 | Doku praktischer teil DNS Teilweise | Schiechtl Kevin, Dipl.-Ing. (FH) |
| 23.01.2023 | 18:30-21:20 | 02:50 | Verbesserung bereits verfasster Dokumentation | Schiechtl Kevin, Dipl.-Ing. (FH) |
| 31.01.2023 | 16:30.20:30 | 04:00 | Kurzbeschreibung und Abstract geschrieben und verbessert | Schiechtl Kevin, Dipl.-Ing. (FH) |
| 01.02.2023 | 15:00-19:30 | 04:30 | Diplomarbeitsdatenbank ausgefüllt mit Abstract und Einleitung dokumentiert | Schiechtl Kevin, Dipl.-Ing. (FH) |
| 02.02.2023 | 18:00-21:30 | 03:30 | Projektergebnisse dokumentiert und Abstract nochmal verbessert | Schiechtl Kevin, Dipl.-Ing. (FH) |
| 07.02.2023 | 17:00-19:30 | 02:30 | Vertiefende Aufgabenstellung geschrieben | Schiechtl Kevin, Dipl.-Ing. (FH) |
| 08.02.2023 | 14:00-18:00 | 04:00 | Doku verfeinert mit Abstract verbesserung, Zusammenfassung etc. | Schiechtl Kevin, Dipl.-Ing. (FH) |
| 09.02.2023 | 18:00-21:00 | 03:00 | Recherche Software für Gantt Diagramm | Schiechtl Kevin, Dipl.-Ing. (FH) |
| 13.02.2023 | 16:00-20:00 | 04:00 | Gantt Diagramm Erstellung, nicht funktioniert | Schiechtl Kevin, Dipl.-Ing. (FH) |
| 14.02.2023 | 08:00-13:00 | 05:00 | Fehlersuche abgeschlossen und Diagramm erstellt | Schiechtl Kevin, Dipl.-Ing. (FH) |
| 31.03.2023 | 08:00-11:00 | 03:30 | Verbesserungen gemacht und Abgabe der Diplomarbeit | Schiechtl Kevin, Dipl.-Ing. (FH) |
| 09.03.2023 | 16:30-18:30 | 02:30 | Projekterfahrung verfasst | Schiechtl Kevin, Dipl.-Ing. (FH) |
| 10.03.2023 | 09:30-11:00 | 01:30 | Danksagung verfasst | Schiechtl Kevin, Dipl.-Ing. (FH) |
|  |  | **SUMME:**  **150.84** |  |  |
|  |

Tabelle 2 Arbeitsnachweis der Diplomarbeit; Edelmann Patrick

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Edelmann Patrick** | | | | |
| **Datum** | **Uhrzeit** | **Stunden** | **Beschreibung** | **Betreuer** |
| 12.09.2022 | 10:50-13:40 | 02:50min | Recherche der einzelnen Themen und Finalisierung der Datenbank | Schiechtl Kevin, Dipl.-Ing. (FH) |
| 13.09.2022 | 10:30-13:30 | 03:00min | Recherche der Virtuellen Maschine und Server Konfiguration (YouTube Video bzw. Brainstorming durch Internetrecherche) | Schiechtl Kevin, Dipl.-Ing. (FH) |
| 14.09.2022 | 9:45-14:45 | 05:00min | Virtuelle Maschine erstellt und darauf Linux Server installiert | Schiechtl Kevin, Dipl.-Ing. (FH) |
| 17.09.2022 | 13:00-15:00 | 02:00min | Linux Server fertig konfiguriert und mit der Doku angefangen | Schiechtl Kevin, Dipl.-Ing. (FH) |
| 18.09.2022 | 16:30-19:00 | 02:30min | Dokumentation des Servers fertiggestellt | Schiechtl Kevin, Dipl.-Ing. (FH) |
| 25.09.2022 | 10:45-13:00 | 02:15min | Konzept für die Diplomarbeit erstellt | Schiechtl Kevin, Dipl.-Ing. (FH) |
| 25.09.2022 | 17:00-19:30 | 2:30min | Konzept überarbeitet | Schiechtl Kevin, Dipl.-Ing. (FH) |
| 26.09.2022 | 15:00-17:30 | 2:30min | Praktische Umsetzung | Schiechtl Kevin, Dipl.-Ing. (FH) |
| 17.10.2022 | 12:30-15:00 | 2:30min | DHCP YouTube Videos und Artikel im Internet angesehen zur Konfiguration | Schiechtl Kevin, Dipl.-Ing. (FH) |
| 19.10.2022 | 14:00-18:00 | 04:00min | DHCP-Konfiguration im Virtuell Box gemacht | Schiechtl Kevin, Dipl.-Ing. (FH) |
| 20.10.2022 | 20:00-22:00 | 02:00min | DHCP-Konfiguration Doku begonnen | Schiechtl Kevin, Dipl.-Ing. (FH) |
| 21.10.2022 | 15:00-16:30 | 01:30min | DHCP Dokumentation Konfiguration fertiggestellt | Schiechtl Kevin, Dipl.-Ing. (FH) |
| 22.10.2022 | 09:50-13:00 | 03:10min | DHCP-Geschichte, Funktion Zuordnung, DHCP-Paket, DHCP-Kommunikation und Sicherheit dokumentiert und recherchiert | Schiechtl Kevin, Dipl.-Ing. (FH) |
| 10.11.2022 | 13:20-16:00 | 02:40min | Datenbank englischer Titel eingetragen, Recherche Admin Benutzer anlegen und Funktion usw. | Schiechtl Kevin, Dipl.-Ing. (FH) |
| 12.11.2022 | 09:00-13:00 | 04:00min | DHCP-Paket Händischen Aufbau gezeichnet, Sudo Admin Benutzer konfiguriert und Doku angefangen | Schiechtl Kevin, Dipl.-Ing. (FH) |
| 13.11.2022 | 11:30-15:00 | 03:30min | Doku Sudo Admin Benutzer fertiggestellt, Sudo Benutzer (Funktion, Geschichte usw.) recherchiert im Internet und YouTube Videos angesehen | Schiechtl Kevin, Dipl.-Ing. (FH) |
| 14.11.2022 | 14:00-16:30 | 02:30min | Sudo Benutzer (Funktion, Geschichte usw.) Doku begonnen und fertiggestellt | Schiechtl Kevin, Dipl.-Ing. (FH) |
| 26.11.2022 | 10:00-14:00 | 04:00min | Ganzen Konfiguration Word Datei zusammengefügt, recherchiert über E-Mail-Server (Protokolle, Konfiguration usw.) und mit der Konfiguration-E-Mail-Server begonnen | Schiechtl Kevin, Dipl.-Ing. (FH) |
| 27.11.2022 | 09:00-14:30 | 05:30min | Konfiguration beendet und Dokumentation dieser Konfiguration angefangen | Schiechtl Kevin, Dipl.-Ing. (FH) |
| 28.11.2022 | 14:00-18:30 | 04:30min | Doku der E-Mail-Konfiguration beendet, recherche über das Python Client Programm (Funktion, was ist zu beachten usw.) | Schiechtl Kevin, Dipl.-Ing. (FH) |
| 11.12.2022 | 14:00-16:30 | 02:30min | Doku E-Mail-Geschichte Funktion usw. begonnen und beendet | Schiechtl Kevin, Dipl.-Ing. (FH) |
| 12.12.2022 | 15:00-18:00 | 03:00min | Doku von Server konfiguration, E-Mail-Server, DHCP-Server, Admin Benutzer abgeändert Teil 1 | Schiechtl Kevin, Dipl.-Ing. (FH) |
| 14.12.2022 | 13:00-16:30 | 03:30min | Doku von Server konfiguration, E-Mail-Server, DHCP-Server, Admin Benutzer abgeändert Teil 2 | Schiechtl Kevin, Dipl.-Ing. (FH) |
| 17.12.2022 | 09:00-13:30 | 04:30min | Doku von dem Dhcp-Server, Admin Benutzer abgeändert bzw. neu gestaltet | Schiechtl Kevin, Dipl.-Ing. (FH) |
| 23.12.2022 | 13:00-17:30 | 04:30min | Doku von der Theorie des E-Mail-Servers angefangen und beendet | Schiechtl Kevin, Dipl.-Ing. (FH) |
| 24.12.2022 | 08:00-11:45 | 03:45min | Begonnen mit dem Programmcode Konzept was der Client alles machen sollt bzw. kann und Programmcode angefangen zu realisieren | Schiechtl Kevin, Dipl.-Ing. (FH) |
| 25.12.2022 | 15:30-18:30 | 03:00min | Programmcode fertig realisiert und fehler Suche in Zeile 5 | Schiechtl Kevin, Dipl.-Ing. (FH) |
| 26.12.2022 | 16:00-20:00 | 04:00min | Programmcode verbessert bzw. überarbeitet und fehler in Zeile 5 ausgebessert, aber weiteren fehler in Zeile 8 | Schiechtl Kevin, Dipl.-Ing. (FH) |
| 27.12.2022 | 08:00-12:00 | 04:00min | Fehler Suche in Zeile 8 und fehler ausgebessert | Schiechtl Kevin, Dipl.-Ing. (FH) |
| 28.12.2022 | 18:00-21:30 | 03:30min | Programmcode umgeschrieben, sodass das Client Programm nachrichten schickt und Antwort vom Server anzeigen kann | Schiechtl Kevin, Dipl.-Ing. (FH) |
| 29.12.2022 | 12:30-17:00 | 04:30min | Doku für das Programm begonnen und fertiggestellt und kleine verbesserungen am Code und Code auskommentiert | Schiechtl Kevin, Dipl.-Ing. (FH) |
| 30.12.2022 | 08:00-12:30 | 04:30min | Praktische Umsetzungen Programmcode dokumentiert und fertiggestellt und theorieteil angefangen mit der recherche | Schiechtl Kevin, Dipl.-Ing. (FH) |
| 31.12.2022 | 12:30-15:30 | 03:00min | Theorieteil bzw. Geschichte Funktion usw. Dokumentation angefangen | Schiechtl Kevin, Dipl.-Ing. (FH) |
| 01.01.2023 | 08:00-11:30 | 03:30min | Theorieteil bzw. Geschichte Funktion usw. Dokumentation fertiggestellt | Schiechtl Kevin, Dipl.-Ing. (FH) |
| 02.01.2023 | 12:00-15:00 | 03:00min | Praktische Teile zusammengefasst in ein Word Dokument Abbildungsnummern richtig nummeriert und Schriftarten usw. angepasst | Schiechtl Kevin, Dipl.-Ing. (FH) |
| 03.01.2023 | 11:00-14:00 | 03:00min | Theorie Teil zusammengefasst in ein Word Dokument und Schriftart usw. angepasst | Schiechtl Kevin, Dipl.-Ing. (FH) |
| 28.01.2023 | 11:00-14:00 | 03:00min | Dilomarbeitsdatenbank überarbeitet und Konzept noch überarbeitet | Schiechtl Kevin, Dipl.-Ing. (FH) |
| 29.01.2023 | 10:30-15:30 | 05:00min | Diplomarbeit theoretischer Teil zusammengefügt | Schiechtl Kevin, Dipl.-Ing. (FH) |
| 30.01.2023 | 14:00-18:30 | 04:30min | Diplomarbeit praktischer Teil zusammengefügt | Schiechtl Kevin, Dipl.-Ing. (FH) |
| 31.01.2023 | 16:30.20:30 | 04:00min | Kurzbeschreibung und Abstract geschrieben und verbessert | Schiechtl Kevin, Dipl.-Ing. (FH) |
| 01.02.2023 | 15:00-19:30 | 04:30min | Diplomarbeitsdatenbank ausgefüllt mit Abstract und Einleitung dokumentiert | Schiechtl Kevin, Dipl.-Ing. (FH) |
| 02.02.2023 | 18:00-21:30 | 03:30min | Projektergebnisse dokumentiert und Abstract nochmal verbessert | Schiechtl Kevin, Dipl.-Ing. (FH) |
| 07.02.2023 | 17:00-19:30 | 02:30min | Vertiefende Aufgabenstellung geschrieben | Schiechtl Kevin, Dipl.-Ing. (FH) |
| 08.02.2023 | 14:00-18:00 | 04:00min | Diplom Doku überarbeitet (Zusammenfassung/Abstract, Aufgabenstellung, Seitenzahl, Bildunterschriften usw.) | Schiechtl Kevin, Dipl.-Ing. (FH) |
| 09.02.2023 | 18:00-21:00 | 03:00min | Recherche für die richtige Software für ein Gantt Diagramm | Schiechtl Kevin, Dipl.-Ing. (FH) |
| 11.02.2023 | 09:00-13:40 | 04:40min | Gantt Diagramm versucht zu erstellen (Hat nicht funktioniert), Abstract und Zusammenfassung geändert bzw. verbessert | Schiechtl Kevin, Dipl.-Ing. (FH) |
| 12.02.2023 | 10:00-15:00 | 05:00min | Abkürzungsverzeichnis erstellt und Literaturverzeichnis | Schiechtl Kevin, Dipl.-Ing. (FH) |
| 13.02.2023 | 16:00-20:00 | 04:00min | Gantt Diagramm Erstellung bis Datum 29.01.2022 mehr hat nicht funktioniert und Fehlersuche | Schiechtl Kevin, Dipl.-Ing. (FH) |
| 14.02.2023 | 08:00-13:00 | 05:00min | Fehlersuche abgeschlossen und Diagramm erstellt | Schiechtl Kevin, Dipl.-Ing. (FH) |
| 31.03.2023 | 08:00-11:00 | 03:30min | Verbesserungen gemacht und Abgabe der Diplomarbeit | Schiechtl Kevin, Dipl.-Ing. (FH) |
| 09.03.2023 | 16:30-18:30 | 02:30min | Projekterfahrung verfasst | Schiechtl Kevin, Dipl.-Ing. (FH) |
| 10.03.2023 | 09:30-11:00 | 01:30min | Danksagung verfasst | Schiechtl Kevin, Dipl.-Ing. (FH) |
|  |  | **SUMME:**  **180min** |  |  |