

Fachinformatiker für Anwendungsentwicklung  
Dokumentation zur betrieblichen Projektarbeit

# Automatisierte Accounterstellung via AMQP-Messaging-System

mit Konsolidierung der Datenquellen  
(Übergabeprotokoll und Angebotssystem)

Auszubildender: Andreas Biller

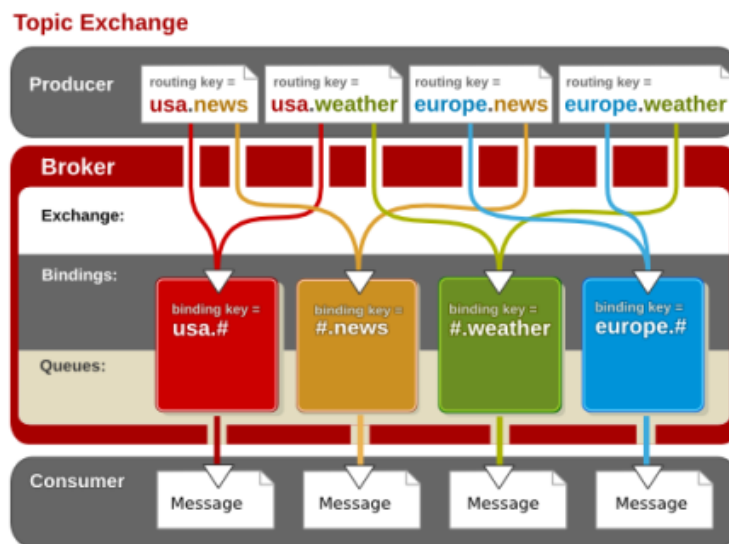


Abbildung 1: Interprozesskommunikation mit Messaging-Queues

**Abgabetermin:** Berlin, den 24.05.2018



Doctena Germany GmbH  
Urbanstr. 116, 10967 Berlin

Dieses Werk einschließlich seiner Teile ist **urheberrechtlich geschützt**. Jede Verwertung außerhalb der engen Grenzen des Urheberrechtsgesetzes ist ohne Zustimmung des Autors unzulässig und strafbar. Das gilt insbesondere für Vervielfältigungen, Übersetzungen, Mikroverfilmungen sowie die Einspeicherung und Verarbeitung in elektronischen Systemen.

## Inhaltsverzeichnis

<b>Abbildungsverzeichnis</b>	<b>III</b>
<b>Tabellenverzeichnis</b>	<b>IV</b>
<b>Listings</b>	<b>V</b>
<b>Abkürzungsverzeichnis</b>	<b>VI</b>
<b>1 Einleitung</b>	<b>1</b>
1.1 Projektumfeld . . . . .	1
1.2 Projektziel . . . . .	1
1.3 Projektbegründung . . . . .	2
1.4 Projektschnittstellen . . . . .	2
1.5 Projektabgrenzung . . . . .	2
<b>2 Projektplanung</b>	<b>3</b>
2.1 Projektphasen . . . . .	3
2.2 Zeitplanung . . . . .	3
2.3 Ressourcenplanung . . . . .	3
2.4 Entwicklungsprozess . . . . .	4
<b>3 Analysephase</b>	<b>4</b>
3.1 Ist-Analyse . . . . .	4
3.2 Wirtschaftlichkeitsanalyse . . . . .	5
3.2.1 „Make or Buy“-Entscheidung . . . . .	5
3.2.2 Projektkosten . . . . .	5
3.2.3 Amortisationsdauer . . . . .	6
3.3 Nutzwertanalyse . . . . .	6
3.4 Qualitätsanforderungen . . . . .	6
3.5 Lastenheft . . . . .	6
3.6 Zwischenstand . . . . .	7
<b>4 Entwurfsphase</b>	<b>7</b>
4.1 Zielplattform . . . . .	8
4.2 Architekturdesign . . . . .	8
4.3 Entwurf der Benutzungsoberfläche . . . . .	8
4.4 Datenmodell . . . . .	8
4.5 Geschäftslogik . . . . .	8
4.6 Maßnahmen zur Qualitätssicherung . . . . .	8
4.7 Pflichtenheft . . . . .	9
4.8 Zwischenstand . . . . .	9

<b>5</b>	<b>Implementierungsphase</b>	<b>10</b>
5.1	Implementierung der Datenstruktur . . . . .	10
5.2	Implementierung der Benutzeroberfläche . . . . .	10
5.3	Implementierung der Geschäftslogik . . . . .	10
5.4	Zwischenstand . . . . .	10
<b>6</b>	<b>Qualitätssicherung und Abnahme</b>	<b>11</b>
6.1	Testing . . . . .	11
6.2	Abnahme . . . . .	11
6.3	Zwischenstand . . . . .	11
<b>7</b>	<b>Einführung</b>	<b>11</b>
7.1	Geplante Einführung . . . . .	11
<b>8</b>	<b>Dokumentation</b>	<b>11</b>
8.1	Benutzerdokumentation . . . . .	12
8.2	Zwischenstand . . . . .	12
<b>9</b>	<b>Fazit</b>	<b>12</b>
9.1	Soll-/Ist-Vergleich . . . . .	12
9.2	Lessons Learned . . . . .	13
9.3	Ausblick . . . . .	13
	<b>Literaturverzeichnis</b>	<b>14</b>
	<b>Eidesstattliche Erklärung</b>	<b>16</b>
<b>A</b>	<b>Anhang</b>	<b>i</b>
A.1	Schritt-für-Schritt Anleitung . . . . .	i
A.2	Lastenheft . . . . .	vi
A.3	Pflichtenheft . . . . .	vii
A.4	Netzpläne . . . . .	viii
A.5	Kompetenzportfolios . . . . .	ix
<b>B</b>	<b>Testdokumentation</b>	<b>xii</b>
B.1	Aufbau der Testumgebung . . . . .	xii
B.1.1	Implementierung der Virtuellen Maschinen . . . . .	xii
B.1.2	Implementierung des virtuellen Netzwerkes . . . . .	xiii
B.1.3	Implementierung des Domain Name Server (DNS)-Servers . . . . .	xiii
B.1.4	Testen der Firewall . . . . .	xiii
B.1.5	Testprotokolle . . . . .	xiii
B.2	Firewall-Skripte . . . . .	xviii
B.2.1	firewall.sh (auf dem Outside-Router) . . . . .	xviii
B.2.2	firewall.sh (auf dem Inside-Router) . . . . .	xxvi

## Abbildungsverzeichnis

1	Interprozesskommunikation mit Messaging-Queues . . . . .	1
2	Netzplan der Demilitarisierte Zone (DMZ) in Raum 3.1.01 (Arbeitsgruppe 9) . . . . .	viii
3	Netzplan der erweiterten DMZ in unserer virtuellen Testumgebung . . . . .	viii

## Tabellenverzeichnis

1	Zeitplanung . . . . .	3
2	Projektkosten . . . . .	6
3	Zwischenstand nach der Analysephase . . . . .	7
4	Zwischenstand nach der Entwurfsphase . . . . .	9
5	Zwischenstand nach der Implementierungsphase . . . . .	10
6	Zwischenstand nach der Abnahmephase . . . . .	11
7	Zwischenstand nach der Dokumentation . . . . .	12
8	Soll-/Ist-Vergleich . . . . .	12
9	Hardwaredetails des Testsystems . . . . .	xii
10	Aus - Aus . . . . .	xiv
11	Aus - An . . . . .	xv
12	An - Aus . . . . .	xvi
13	An - An . . . . .	xvii

Listings

Listings/outside/firewall.sh . . . . .	xviii
Listings/inside/firewall.sh . . . . .	xxvi

## Abkürzungsverzeichnis

<b>AMQP</b>	Advanced Message Queuing Protocol
<b>CAS</b>	Column Access Strobe
<b>CL</b>	Column Access Strobe ( <a href="#">CAS</a> ) latency
<b>DDR4</b>	Double Data Rate 4th-Generation
<b>DIMM</b>	Dual In-line Memory Module
<b>DMZ</b>	Demilitarisierte Zone
<b>DNS</b>	Domain Name Server
<b>GB</b>	Giga Byte
<b>GHz</b>	Giga Hertz
<b>ID</b>	Identification
<b>IP</b>	Internet Protokoll
<b>LAN</b>	Local Area Network
<b>MB</b>	Mega Byte
<b>NTP</b>	Network Time Protocol
<b>RAM</b>	Random Access Memory
<b>SSD</b>	Solid State Drive
<b>VLAN</b>	Virtual Local Area Network ( <a href="#">LAN</a> )
<b>VM</b>	Virtual Machine

## 1 Einleitung

*Pläne sind nichts, Planung ist alles.*

Dwight D. Eisenhower, ehemaliger US-Präsident

*Kein Plan überlebt die erste Feindberührung.*

Helmuth von Moltke, preußischer Generalfeldmarschall

### 1.1 Projektumfeld

Das Projekt wird als Teil der Abschlussprüfung im Rahmen meiner Ausbildung zum Fachinformatiker für Anwendungsentwicklung bei der Doctena Germany GmbH umgesetzt. Doctena ist ein internationales Unternehmen mit Hauptsitz in Luxemburg und Niederlassungen in 5 weiteren europäischen Ländern. Doctena bietet Patienten eine internationale Plattform zur Online-Terminbuchung an. "2013 (...) wurde die Plattform Doctena in Luxemburg ins Leben gerufen. (...) Aufgrund des anhaltenden Erfolges und des schnellen Wachstums des Projekts dehnte das Unternehmen seine Aktivitäten auf Belgien, die Niederlande, [Österreich,] die Schweiz und Deutschland aus. Seit 2016 hat sich Doctena mit sechs Wettbewerbern zusammengeschlossen: DocBook (BE), Doxter (DE), Terminland (DE), Sanmax (BE), Mednanny (AT) und Bookmydoc (LU). Doctena ist heute die führende medizinische Buchungsplattform in Europa."<sup>1</sup> Doctena beschäftigt momentan um die 80 Mitarbeiter, ca. 30 davon hier in Berlin. Hauptprodukt neben der Terminbuchungs-Plattform für Patienten ist die cloudbasierte Terminverwaltungs-Lösung Doctena Pro für Ärzte und Praxen. "Doctena hat das Ziel, den Zugriff von Patienten auf verfügbare Termine von Ärzten und Praktikern zu vereinfachen. Patienten können mit Hilfe der Onlineplattform oder der Handy-App verfügbare Termine sehen und buchen. (...) Die Lösung ist mit vielen medizinischen Buchungssoftwares kompatibel und kann deshalb leicht in die Struktur von Ärzten und ihren Praxen integriert werden."<sup>2</sup> So können Ärzte praxisintern ihre Verfügbarkeiten managen und gleichzeitig freie Termine über die Plattform oder über die eigene Internetseite anbieten. Eigentlicher Kunde des Projektes sind die firmeninternen Abteilungen Verkauf und Onboarding bei der Doctena Germany GmbH. Die Abnahme erfolgt hierbei durch den CIO der Doctena Germany GmbH, André Rauschenbach.

### 1.2 Projektziel

Ziel des Projektes ist die Erweiterung des zur Angebots- und Vertragserstellung benutzten Angebots-systemes. Den Onboarding-Managern soll damit ermöglicht werden, nach Annahme des Angebotes durch den Kunden, in unserem System die automatische Accounterstellung auf dem luxemburger System Doctena Pro über den bestehenden Advanced Message Queuing Protocol (AMQP) Message-Bus auszulösen. Zusätzlich soll die Maske des Angebotsformulares um Eingabefelder für das in Folge zu

---

<sup>1</sup>Über Doctena - Unsere Firma - Eine Erfolgsgeschichte, [HTTPS://WWW.DOCTENA.COM](https://www.doctena.com) [2018b]

<sup>2</sup>Über Doctena - Der Doctena Update Catalog, [HTTPS://WWW.DOCTENA.COM](https://www.doctena.com) [2018a]



## 1 Einleitung

---

erstellende Übergabeprotokoll erweitert und so eine Konsolidierung der bisher für den Onboarding-Prozess verwendeten Datenquellen erreicht werden.

### 1.3 Projektbegründung

Mit diesem Projekt soll die bisherige manuelle Erstellung der neuen Kunden-Accounts durch die Onboarding-Manager automatisiert werden, was zu einer Zeit- und somit auch Kostenreduzierung im Onboarding-Prozess führt. Gleichzeitig sollen mögliche Fehlerquellen bei der im bisherigen Prozess hierzu verwendeten doppelten Datenhaltung eliminiert werden. Hauptmotivation hinter dem Projekt ist somit die Prozessoptimierung bei der bisherigen Accounterstellung. Neben dem reinen Arbeitsaufwand sollen hiermit auch Fehler reduziert werden, die durch die vielen repetitive Aufgaben beim Onboarding und anspruchslöse Copy-und-Paste-Tätigkeiten leicht entstehen können.

### 1.4 Projektschnittstellen

Das Angebotssystem im Backend des deutschen Systems Doctena Standard wurde mit Ruby on Rails erstellt. Es besteht aus einer in Rails üblichen MVC-Struktur. Somit erfolgt die interne Kommunikation zwischen der in der Datenhaltungsschicht angebundenen nicht-relationalen Datenbank MongoDB und den Modell-Klassen über einen in Rails als Bibliothek eingebundenen Data-Connector, MongoDB Driver. Die browserbasierte Eingabemaske des Views in der Schicht der Benutzeroberfläche interagiert intern über die im Controller vorhandene Geschäftslogik mit in Rails üblichen CRUD-Operationen bzw. Routes mit dem Modell und somit der Datenstruktur. Die externe Kommunikation zwischen den internen Modellen in Rails und den Objekten im Zielsystem Doctena Pro, mit denen der neue Benutzeraccount nach Vertragserstellung angelegt werden soll, erfolgt über einen AMQP Message-Bus. Dieser wird bereits zur Synchronisation der Verfügbarkeiten der Ärzte aus den verschiedenen Backend-Bereichen der angeschlossenen Systeme mit dem CPP von Doctena benutzt. Die Datenstruktur, der AMQP-Exchange-Type, ist JSON. Das Projekt wurde von Doctenas internationalem CIO, Alain Fountain, in Absprache mit dem CIO von Doctena Germany, André Rauschenbach, genehmigt. Doctena stellt somit als Kunde im Rahmen der Projektarbeit und Ausbildung alle zur Umsetzung benötigten Mittel zur Verfügung. Benutzer der Anwendung sind die Mitarbeiter von Doctena Germany in den Abteilungen Verkauf und Onboarding. Diesen soll nach Abnahme durch den Auftraggeber, vertreten durch den CIO von Doctena Germany, das fertige Produkt präsentiert werden. Zusätzlich soll für die Benutzer eine Benutzerdokumentation für die Accounterstellung im Firmeninternen Wiki erstellt werden.

### 1.5 Projektabgrenzung

Das Aktivieren von Features über den Bus ist Seitens Luxemburg noch nicht möglich. Deswegen wurde diese Funktionalität aus dem Projekt ausgeklammert. Sie wird erst zu einem späteren Zeitpunkt

umgesetzt. Da das Einrichten der Testumgebungen für zwei komplette Systeme sehr aufwendig ist und den Projektrahmen übersteigt, wurde auf die Korrektheit der Messages auf dem Bus getestet.

## 2 Projektplanung

### 2.1 Projektphasen

Das Projekt soll im Zeitraum zwischen der schriftlichen Abschlussprüfung und dem Abgabetermin des Projekts während den täglichen Arbeitszeiten von 10:00 bis 19:00 Uhr umgesetzt werden. Die Arbeitszeit kann während diesem Zeitraum neben der von der Erledigung dringender oder anderweitig notwendiger Aufgaben beanspruchten Zeit frei für das Projekt genutzt werden. Entsprechend dem in Kapitel 2.5 beschriebenen Entwicklungsprozess wurde der Projektablauf in die entsprechenden Phasen unterteilt. Deren nähere Planung bzw. Durchführung kann den entsprechenden Kapiteln dieser Dokumentation entnommen werden.

### 2.2 Zeitplanung

Für die Umsetzung des Projektes stehen Seitens der Anforderungen der IHK 70 Stunden zur Verfügung. Diese wurden zur Antragstellung auf die einzelnen Phasen verteilt. Die grobe Zeitplanung der Hauptphasen kann der Tabelle 1 Zeitplanung auf dieser Seite entnommen werden. Eine ausführlichere Zeitplanung findet sich im Anhang unter (TODO)

Projektphase	Geplante Zeit
Analyse	6 h
Entwurf	11 h
Implementierung	39 h
Abnahme und Deployment	5 h
Dokumentation	9 h
<b>Gesamt</b>	<b>70 h</b>

Tabelle 1: Zeitplanung

### 2.3 Ressourcenplanung

TODO: Liste Ressourcen, verlinken

Die benötigten Mittel zur Durchführung des Projektes werden vom Auftraggeber Doctena zur Verfügung gestellt. Eine detaillierte Auflistung der zur Durchführung benötigten Ressourcen findet sich im Anhang unter (TODO). Die Benutzung dieser Ressourcen wird später mit einem pauschalen Wert für die in Kapitel 3 angestellten Berechnungen berücksichtigt.

## 2.4 Entwicklungsprozess

Der Projektablauf wurde vom Wasserfallmodell ausgehend in die folgenden Projektphasen unterteilt:

**Definition und Projektantrag, Planung, Analyse, Entwurf, Implementierung, Qualitätssicherung und Abnahme, Einführung, Dokumentation**

Diese werden in sequentieller Reihenfolge mit zwischenzeitlichen Projektbesprechungen mit den beteiligten Stellen zum aktuellen Projektstand bis zum Abgabetermin abgearbeitet. Die Entwicklung während der Implementierungsphase wird nach TDD-Prinzipien durchgeführt, wodurch auch die Tests zur Sicherstellung der Einhaltung der vereinbarten Anforderungen aus dem Pflichtenheft teilweise während dieser Phase erstellt werden. Da das Angebotssystem an unsere CI-Pipeline angebunden ist, kann eine erfolgreiche Abnahme und anschließende Einführung erst nach einem bestehen aller Tests der Qualitätssicherung erfolgen. Artefakte für die Dokumentation werden, wo möglich, bereits während der gesamten Durchführung gesammelt.

## 3 Analysephase

Eine erste Analyse wurde bereits während der Projektdefinition zur Antragstellung durchgeführt. Darauf aufbauend folgt in diesem Kapitel eine genauere Analyse der Situation. Die hierbei gewonnenen Erkenntnisse werden dann als Anforderungen im Lastenheft genauer definiert.

### 3.1 Ist-Analyse

Unsere Verkäufer erstellen täglich Angebote an Ärzte aus ganz Deutschland. Die Daten, die zur Erstellung dieser Angebote im Formular des Angebotssystems eingegeben werden, können bei Doctena Standard bereits zum automatischen Erstellen eines Accounts für den zukünftigen Benutzer verwendet werden. Um einen neuen Account bei Doctena Pro anzulegen werden die gleichen Daten als Grundlage benutzt. Die im Angebotssystem bereits in digitaler Form vorliegenden Daten zu Ärzten und Praxis werden im momentanen Onboarding-Prozess bei Doctena Pro von einem Onboarding-Manager per Copy-und-Paste aus dem Angebot in ein Übergabeprotokoll im von Doctena verwendeten CRM, Close.io, kopiert. Hier kommen einige zusätzliche Informationen, wie Termine für Schulung, Feinabstimmung und Datenimport hinzu. Dann werden in einem weiteren manuellen Schritt die Accounts im Backend von Doctena Pro vom Onboarding-Manager angeklickt und die Daten zu Ärzten und Praxis wieder aus dem Übergabeprotokoll per Copy-und-Paste in die Eingabemaske des Backends transferiert. Dieser Prozess ist repetitiv, nimmt unnötig Zeit in Anspruch, erzeugt in seinem Verlauf redundante Daten und ist somit fehleranfällig. Gleichzeitig existiert zum Datenaustausch der verschiedenen internationalen Systeme ein AMQP-Message-Queue-Bus, über den die Verfügbarkeiten der Ärzte mit dem CPP synchronisiert werden. Um den Onboarding-Prozess zu optimieren, sollen zum einen die Daten des Übergabeprotokolls direkt im Angebotsformular eingegeben werden können, zum anderen sollen die Daten zu Praxis und den Ärzten dazu verwendet werden, die Objekterstellung in

Doctena Pro über den Bus auszulösen. Hierzu müssen zusätzliche Felder im Formular der Benutzungsoberfläche und im verwendeten Datenmodell hinzugefügt werden. In der Geschäftslogik im Controller müssen die Daten dann zum auf dem Message-Queue-Bus verwendeten AMQP-Exchange-Type konvertiert werden. Es soll ein Button zum Auslösen des Versandes über den Bus erstellt werden, welcher nur für Onboarding-Manager sichtbar ist. Bei einem Vertragsabschluss für Doctena Pro soll automatisch eine E-Mail an den verantwortlichen Onboarding-Manager versendet werden, damit dieser die Daten des Angebots im Formular überprüfen und ggf. korrigieren kann und dann die automatische Account-Erstellung auslösen kann.

## 3.2 Wirtschaftlichkeitsanalyse

### 3.2.1 „Make or Buy“-Entscheidung

Die „Make or Buy“-Entscheidung ist in diesem Fall leicht getroffen. Zum einen existiert bereits eine bestehende [AMQP](#)-Busverbindung zum Datentransfer zwischen dem Ruby on Rails Backend von Doctena Standard, in welchem das Angebotssystem eingebettet ist, und dem Java Zielsystem Doctena Pro in Luxemburg. Zum anderen kann über kommerziell erhältliche Lösungen zur Angebotserstellung die automatische Account-Erstellung nicht realisiert, und so auch der bisherige manuelle Onboarding-Prozess nicht optimiert werden.

### 3.2.2 Projektkosten

TODO: Berechnung anzeigen, Kostentabelle erweitern, Endwert eintragen

Die realen Kosten für die Durchführung des Projekts setzen sich sowohl aus Personal-, als auch aus Ressourcenkosten zusammen. Gerechnet wird hier bei den Personalkosten lediglich mit dem fiktiven Gehalt eines Auszubildenden im dritten Lehrjahr von ungefähr 884 € Brutto bei 20 Arbeitstagen im Monat. Der Arbeitgeberanteil zur Sozialversicherung bemisst sich auf monatlich 229,53 €. Für das Gehalt aller übrigen am Projekt beteiligten Mitarbeiter wird ein pauschaler Stundensatz von 30 € angenommen.

(TODO)

Es ergibt sich also ein Stundenlohn von 6,96 €. Die Durchführungszeit des Projekts beträgt 70 Stunden. Die Nutzung von Ressourcen<sup>3</sup> wird hier mit einem pauschalen Stundensatz von 10 € € berücksichtigt. Eine Aufstellung der Kosten befindet sich in Tabelle 2. Die Kosten belaufen sich auf insgesamt (TODO).

---

<sup>3</sup>Räumlichkeiten, Arbeitsplatzrechner, Lizenzen, etc.

Vorgang	Zeit	Kosten pro Stunde	Kosten
Entwicklungskosten	42 h	$4,62 \text{ €} \cdot 2 = 9,24 \text{ €}$	388,08 €
			<b>388,08 €</b>

Tabelle 2: Projektkosten

### 3.2.3 Amortisationsdauer

TODO: Amortisationsdauer berechnen, Berechnung anzeigen, Text verfassen

(TODO)

(TODO)

## 3.3 Nutzwertanalyse

Den größten Nutzen des Projektes stellt die Optimierung des bisherigen Arbeitsprozesses beim Onboarding neuer Kunden dar. Dadurch wird Zeit eingespart und so freie Kapazitäten für andere Tätigkeitsfelder im Onboarding-Prozess geschaffen. Durch die Reduzierung der sich wiederholenden manuellen Tätigkeiten und der dadurch möglichen Fehler erhöht sich die Kundenzufriedenheit, wodurch zusätzlich die Anzahl der Kundenanfragen an die Support-Abteilung reduziert wird.

## 3.4 Qualitätsanforderungen

Da das Angebotssystem als Webanwendung sowohl von den Mitarbeitern von Doctena Germany in den Abteilungen Verkauf und Onboarding, wie auch von den Kunden zum Abschließen von Verträgen online genutzt wird, wird bei den Qualitätsanforderungen besonderes Augenmerk auf die Abgrenzung der Funktionalität bei den verschiedenen Benutzerrollen gelegt. Die Funktionen zur Account-Erstellung sollen nur Onboarding-Managern zur Verfügung stehen, die Eingabefelder für system-interne Daten nur den Angestellten von Doctena, während die Kunden in der Lage sein sollen, ihre personenbezogenen Daten ggf. ändern zu können.

## 3.5 Lastenheft

Im folgenden werden die wichtigsten Anforderungen an das Projekt aus dem Lastenheft dargestellt.

### Anforderungen an die Benutzungsoberfläche:

**LB10:** Die Daten des Übergabeprotokolls sollen im Angebotsformular integriert werden.

**LB20:** Es soll ein Button zur Account-Erstellung hinzugefügt werden.

**LB30:** benötigte zusätzliche Felder zur Account-Erstellung benötigten Daten hinzugefügt werden.

### Funktionelle Anforderungen:

**LF10:** Der Button zur Account-Erstellung soll nur von Onboarding-Managern genutzt werden können. Die Formularfelder zur Account-Erstellung und des Übergabeprotokolls sollen nur für Verkauf und Onboarding zur Verfügung stehen.

**LF20:** Bei einem Vertragsabschluss durch einen Kunden soll die Onboarding-Abteilung eine Benachrichtigung per E-Mail bekommen.

**LF30:** Es sollen die für einen Account auf Doctena Pro benötigten Objekte erzeugt werden.

**LF40:** Die Objekte sollen über den AMQP Message-Queue-Bus an Doctena Pro übermittelt werden.

**LF50:** Wenn mehr als ein Arzt in der Praxis vorhanden ist, soll ein zusätzlicher Admin-User für die Praxis in Doctena Pro angelegt werden.

### Sonstige Anforderungen:

**LS10:** Die Funktionalität soll durch Tests der Anforderungen gewährleistet werden, damit das Projekt im Produktivsystem genutzt werden kann.

**LS20:** Die Benutzer sollen im Umgang mit den neuen Funktionen geschult werden.

## 3.6 Zwischenstand

Tabelle 3 zeigt den Zwischenstand nach der Analysephase.

Vorgang	Geplant	Tatsächlich	Differenz
1. Analyse des Ist-Zustands	3 h	3 h	
2. Zeit- und Ressourcenplanung	2 h	3 h	+1 h
3. Wirtschaftlichkeit und „Make or buy“-Entscheidung	2 h	1 h	-1 h
4. Qualitätsanforderungen und Lastenheft	3 h	3 h	
5. Beginn Dokumentation und Kompetenzportfolio	2 h	3 h	+1 h
<b>Analysephase</b>	<b>12 h</b>	<b>13 h</b>	<b>+1 h</b>

Tabelle 3: Zwischenstand nach der Analysephase

## 4 Entwurfsphase

Da das Projekt die Erweiterung des Funktionsumfanges eines bereits bestehenden Systems darstellt, sind viele Entscheidungen zu den verwendeten Technologien bereits im Vorfeld getroffen. Im Folgenden wird deshalb detaillierter beschrieben, wie die geplanten Erweiterungen im existierenden Angebotssystem realisiert werden sollen.

## 4.1 Zielplattform

TODO: browser, html, css, javascript, ruby, rails, mongodb, aws, heroku, cloud

(TODO)

## 4.2 Architekturdesign

TODO: rails, mvc

(TODO)

## 4.3 Entwurf der Benutzungsoberfläche

TODO: für wen, bootstrap, screenshot, neue felder, mockup

(TODO)

## 4.4 Datenmodell

TODO: erm contract, zustandsdiagramm contract

(TODO)

## 4.5 Geschäftslogik

TODO: mapping standard > pro, klassendiagramme standard und pro

(TODO)

## 4.6 Maßnahmen zur Qualitätssicherung

TODO: minitest, tdd, ci

(TODO)

## 4.7 Pflichtenheft

Im folgenden die wichtigsten vereinbarten Pflichten des Projektes bezüglich Qualitätssicherung, um die Funktionalität und Erweiterbarkeit während der aktiven Nutzung im Produktivsystem gewährleisten zu können.

### Pflichten der Benutzungsoberfläche:

**PB10:** Die Daten des Übergabeprotokolls müssen persistiert, geladen und erneut persistiert werden.

**PB20:** Der Button zur Account-Erstellung muss den Datenversand über den AMQP Message-Queue-Bus auslösen.

**PB30:** Die Daten zur Account-Erstellung müssen persistiert, geladen und erneut persistiert werden.

### Funktionelle Pflichten:

**PF10:** Für Benutzer mit der Rolle Manager muss der Button zur Account-Erstellung sowie die Formularfelder zur Account-Erstellung und das Übergabeprotokoll sichtbar sein. Für Benutzer mit der Rolle Verkäufer müssen die Formularfelder zur Account-Erstellung und das Übergabeprotokoll sichtbar sein. Für einen Benutzer ohne Rolle dürfen weder der Button zur Account-Erstellung noch die Formularfelder zur Account-Erstellung oder das Übergabeprotokoll vorhanden sein.

**PF20:** Bei einem Vertragsabschluss muss eine E-Mail an die Adresse [onboarding@doctena.com](mailto:onboarding@doctena.com) gesendet werden.

**PF40:** Die Daten aus dem Vertrag müssen zur Account-Erstellung in das auf dem AMQP Message-Queue-Bus verwendete Datenformat übersetzt werden.

**PF50:** Bei mehr als einem Arzt pro Praxis muss ein zusätzlicher User als Admin-User ohne Arzt und Kalender erstellt und dieses zusätzliche Objekt über den Bus gesendet werden.

## 4.8 Zwischenstand

Tabelle 4 zeigt den Zwischenstand nach der Entwurfsphase.

Vorgang	Geplant	Tatsächlich	Differenz
1. Erstellen des Netzwerkplanes	1 h	1 h	
2. Qualitätssicherung	1 h	2 h	+1 h
3. Erstellen des Pflichtenhefts	1 h	4 h	+3 h
4. Dokumentation erweitern	2 h	1 h	-1 h
<b>Entwurfsphase</b>	<b>5 h</b>	<b>8 h</b>	<b>+3 h</b>

Tabelle 4: Zwischenstand nach der Entwurfsphase



## 5 Implementierungsphase

### 5.1 Implementierung der Datenstruktur

TODO: mongodb, connector, model, listing model

(TODO)

### 5.2 Implementierung der Benutzeroberfläche

TODO: Text, Screenshot verlinken, Listing view/partial

(TODO) Screenshots der Anwendung in der Entwicklungsphase mit Dummy-Daten befinden sich im Anhang (TODO)

### 5.3 Implementierung der Geschäftslogik

TODO: Text, Listing controller

(TODO)

### 5.4 Zwischenstand

Tabelle 5 zeigt den Zwischenstand nach der Implementierungsphase.

Vorgang	Geplant	Tatsächlich	Differenz
1. Einrichten der VMs	1 h	1 h	
2. Konfiguration der Router	4 h	3 h	-1 h
3. Einrichtung von NAT und Portforwarding	6 h	4 h	-2 h
4. Einrichtung physische Hosts	1 h	1 h	
5. Einrichten des Webserver	3 h	1 h	-2 h
6. Erstellen der Webserver-Startseite	1 h	2 h	+1 h
7. Konfiguration der Firewall	30 h	28 h	-2 h
8. Qualitätssicherung	3 h	1 h	-2 h
9. Dokumentation erweitern	12 h	30 h	+18 h
<b>Implementierungsphase</b>	<b>61 h</b>	<b>71 h</b>	<b>+10 h</b>

Tabelle 5: Zwischenstand nach der Implementierungsphase

## 6 Qualitätssicherung und Abnahme

### 6.1 Testing

TODO: Text, Screenshots Testresultate verlinken

(TODO)

### 6.2 Abnahme

TODO: Text

(TODO)

### 6.3 Zwischenstand

Tabelle 6 zeigt den Zwischenstand nach der Abnahmephase.

Vorgang	Geplant	Tatsächlich	Differenz
1. Abnahmetest der Fachabteilung	1 h	0 h	-1 h

Tabelle 6: Zwischenstand nach der Abnahmephase

## 7 Einführung

### 7.1 Geplante Einführung

TODO: Text, geplante Schulung, Wiki, kein Zwischenstand

(TODO)

## 8 Dokumentation

Die Dokumentation wurde in  $\text{\LaTeX}$  erstellt. Zu Ihrer Erstellung wurden zusätzlich folgende Webseiten zu Hilfe gezogen: [TEXSTUDIO.SOURCEFORGE.NET](https://www.texstudio.com/) [2017], [WWW.LATEX TUTORIAL.COM](https://www.latex-tutorial.com/) [2015]

TODO: Text

(TODO)

## 8.1 Benutzerdokumentation

TODO: Screenshot Wiki, verlinken

Die Benutzerdokumentation wurde zur Unterstützung der Mitarbeiter beim neuen Onboarding-Prozess im unternehmenseigenen Wiki erstellt und soll den Benutzern jederzeit einen einfachen Überblick über die Bedeutung der neuen Eingabefelder und den neuen Prozess im allgemeinen bieten. Sie wird zusammen mit der neuen Funktionalität während des Termines der geplanten Benutzerschulung vorgestellt. Ein Screenshot des Wiki-Eintrages befindet sich im Anhang [A.1: Schritt-für-Schritt Anleitung](#) auf Seite [i](#).

## 8.2 Zwischenstand

Tabelle 7 zeigt den Zwischenstand nach der Dokumentation.

Vorgang	Geplant	Tatsächlich	Differenz
1. Erstellen des Kompetenzportfolios	3 h	3 h	
2. Erstellen der Projektdokumentation	12 h	40 h	+28 h
3. Erstellen der Testdokumentation	12 h	20 h	+6 h
<b>Dokumentation</b>	<b>27 h</b>	<b>63 h</b>	<b>+34 h</b>

Tabelle 7: Zwischenstand nach der Dokumentation

## 9 Fazit

### 9.1 Soll-/Ist-Vergleich

TODO: Zeitplanung Vergleich anpassen, Text, Link

(TODO)

Phase	Geplant	Tatsächlich	Differenz
Analysephase	12 h	13 h	+1 h
Entwurfsphase	5 h	8 h	+3 h
Implementierungsphase	61 h	71 h	+10 h
Abnahmetest der Fachabteilung	1 h	0 h	-1 h
Erstellen der Dokumentation	27 h	63 h	+34 h
Pufferzeit	20 h	0 h	-2 h
<b>Gesamt</b>	<b>126 h</b>	<b>155 h</b>	<b>+45 h</b>

Tabelle 8: Soll-/Ist-Vergleich

## 9.2 Lessons Learned

TODO: Text

(TODO)

## 9.3 Ausblick

TODO: Text

(TODO)

## Literaturverzeichnis

### **blog.workinghardinit.work 2017**

BLOG.WORKINGHARDINIT.WORK, Install Windows Server 2016 Network D. HowTo: *Didier Van Hoyer, Installing Intel I211, I217V, I218V and I219V drivers on Windows Server 2016 with EUFI boot.* 2017. – <https://blog.workinghardinit.work/2017/06/19/installing-intel-i211-i217v-i218v-i219v-drivers-windows-server-2016-eufi-boot/>, Aufgerufen 2017-06-27

### **https://www.doctena.com 2018a**

HTTPS://WWW.DOCTENA.COM: *Über Doctena - Der Doctena Update Catalog, DOCTENA.* 2018. – <https://www.doctena.com/doctena-update-catalog-deutsch/>, Aufgerufen 2018-05-20

### **https://www.doctena.com 2018b**

HTTPS://WWW.DOCTENA.COM: *Über Doctena - Unsere Firma - Eine Erfolgsgeschichte, DOCTENA.* 2018. – <https://www.doctena.com/uber-doctena/unsere-firma-doctena/>, Aufgerufen 2018-05-20

### **texstudio.sourceforge.net 2017**

TEXSTUDIO.SOURCEFORGE.NET, User M. Manual: *TexStudio Manual, User Manual.* 2017. – [http://texstudio.sourceforge.net/manual/current/usermanual\\_en.html](http://texstudio.sourceforge.net/manual/current/usermanual_en.html), Aufgerufen 2017-06-07

### **www.latex tutorial.com 2015**

TUTORIAL.COM, Bibtex www.latex Tutorial: *Bibliography in LaTeX with Bibtex/Biblatex.* 2015. – <https://www.latex-tutorial.com/tutorials/beginners/latex-bibtex/>, Aufgerufen 2017-06-30

### **www.debian.org 2017a**

WWW.DEBIAN.ORG, Chapter 1. Manual: *Debian Manual, Chapter 12: Programming.* 2017. – <https://www.debian.org/doc/manuals/debian-reference/ch12.en.html>, Aufgerufen 2017-06-10

### **www.debian.org 2017b**

WWW.DEBIAN.ORG, Chapter 5. Manual: *Debian Manual, Chapter 5: Network Setup.* 2017. – <https://www.debian.org/doc/manuals/debian-reference/ch05.en.html>, Aufgerufen 2017-06-10

### **www.debian.org 2015**

WWW.DEBIAN.ORG, Securing D. HowTo: *Debian Manual, Detailed user guide for securing and hardening of the default Debian installation.* 2015. – <http://www.debian.org/doc/manuals/securing-debian-howto/>, Aufgerufen 2017-06-12

### **www.frozentux.net 2006**

WWW.FROZENTUX.NET, Iptables Tutorial: *Oskar Andreasson, Iptables Tutorial 1.2.1.* 2006. –

<https://www.frozentux.net/iptables-tutorial/chunkyhtml/index.html>, Aufgerufen 2017-06-14

**www.netfilter.org 2002a**

WWW.NETFILTER.ORG, NAT HowTo: *Rusty Russell, Linux 2.4 NAT HOWTO*. 2002. – <http://www.netfilter.org/documentation/HOWTO/NAT-HOWTO.html>, Aufgerufen 2017-06-14

**www.netfilter.org 2002b**

WWW.NETFILTER.ORG, Packet F. HowTo: *Rusty Russell, Linux 2.4 Packet Filtering HOWTO*. 2002. – <http://www.netfilter.org/documentation/HOWTO/packet-filtering-HOWTO-7.html>, Aufgerufen 2017-06-14

## Eidesstattliche Erklärung

Ich, Andreas Biller, versichere hiermit, dass ich meine **Dokumentation zur betrieblichen Projektarbeit** mit dem Thema

*Automatisierte Accounterstellung via AMQP-Messaging-System mit Konsolidierung der Datenquellen (Übergabeprotokoll und Angebotssystem)*

selbständig verfasst und keine anderen als die angegebenen Quellen und Hilfsmittel benutzt habe, wobei ich alle wörtlichen und sinngemäßen Zitate als solche gekennzeichnet habe. Die Arbeit wurde bisher keiner anderen Fachkraft vorgelegt und auch nicht veröffentlicht.

Berlin, den 24.05.2018

---

ANDREAS BILLER

## A Anhang

### A.1 Schritt-für-Schritt Anleitung

FA54

P / LZ

Herr Henze

Gruppe 9

Andreas Biller, Rico Krüger

Thema: Aufbau einer DMZ

#### 1. Aufsetzen der virtuellen Maschinen

Auf zwei Clients je eine virtuelle Maschine mit Linux-OS (Debian) aufsetzen (mit VM-Ware Player). Falls VM bereits vorhanden, diese in eigenen Benutzer-Ordner kopieren. Sonst über Linux mit VM-Ware Player installieren.

Rolle	Name	Passwort
Benutzer	user	oszimt
Administrator	root	osz

#### 2. Änderung des Modus der Netzwerkschnittstellen

Wir öffnen VM-Ware Player und starten Linux. Dann versetzen wir in den Einstellungen die Netzwerkschnittstellen in den **Bridge-Modus**.

#### 3. Erstellung Netzwerkplan

Wir erstellen einen Netzplan und vergeben die benötigten IP-Adressen.

#### 4. Konfiguration Schnittstellen und NAT der Linux-VMs als Router

Die Schnittstellen werden auf beiden Debian-Systemen in der Datei „*/etc/network/interfaces*“ konfiguriert.

##### 4.1. Konfiguration Inside-Router

```
# The loopback network interface
auto lo
iface lo inet loopback

# The primary network interface
allow-hotplug eth0
iface eth0 inet static
address 10.0.9.1
netmask 255.255.255.0

# The second interface
allow-hotplug eth1
iface eth1 inet static
address 172.16.9.2
netmask 255.255.255.0
gateway 172.16.9.1
```

##### 4.2. Konfiguration Outside-Router

```
# The loopback network interface
auto lo
iface lo inet loopback

# The primary network interface
allow-hotplug eth0
iface eth0 inet static
address 172.16.9.1
netmask 255.255.255.0

# second interface
allow-hotplug eth1
iface eth1 inet static
address 192.168.200.109
netmask 255.255.255.0
gateway 192.168.200.1

### static routing ###
post-up route add -net 10.0.9.0 netmask 255.255.255.0 gw 172.16.9.2
pre-down route del -net 10.0.9.0 netmask 255.255.255.0 gw 172.16.9.2

### NAT and Port-Forwarding ###
```



FA54

P / LZ

Herr Henze

Gruppe 9

Andreas Biller, Rico Krüger

Thema: Aufbau einer DMZ

```
post-up iptables -A FORWARD -o eth1 -s 172.16.9.0/24 -m conntrack --ctstate NEW -j ACCEPT
post-up iptables -A FORWARD -o eth1 -s 10.0.9.0/24 -m conntrack --ctstate NEW -j ACCEPT
post-up iptables -t nat -A POSTROUTING -o eth1 -j MASQUERADE

post-up iptables -A PREROUTING -t nat -i eth1 -p tcp --dport 80 -j DNAT --to-destination 172.16.9.3:80
post-up iptables -A FORWARD -p tcp -d 172.16.9.3 --dport 80 -j ACCEPT
post-up iptables -A POSTROUTING -t nat -s 172.16.9.3 -o eth1 -j MASQUERADE
```

## 5. Aktivierung IP-Forwarding

Temporäre Aktivierung:

Ausführen des Befehls: `echo „1“ > /proc/sys/net/ipv4/ip_forward`

Permanente Aktivierung:

In der Datei „`/etc/sysctl.conf`“ den Wert von „`#net.ipv4.ip_forward`“ auf **1** setzen und die Auskommentierung aufheben: `net.ipv4.ip_forward=1`

## 6. Neustarten der Schnittstellen zum Übernehmen der Konfiguration

Dafür werden folgende Befehle nacheinander ausgeführt:

```
ifdown eth0
ifdown eth1
ifup eth0
ifup eth1
```

## 7. Konfiguration der physikalischen Netzwerk-Schnittstellen der Windows-Clients

Die physikalischen Schnittstellen der Hosts von den beiden Linux-VMs werden über „Systemsteuerung“ -> „Netzwerk- und Freigabecenter“ -> „Adaptoreinstellungen ändern“ -> „Ethernet-Adapter“ -> „Eigenschaften“ -> „Internetprotokoll, Version 4 (TCP/IPv4)“ -> „Eigenschaften“ geändert.

### 7.1. Konfiguration Host Inside-Router



## 7.2. Konfiguration Host Outside-Router



## 8. Deaktivierung der Windows-Firewall

Firewall auf den Windows-Clients deaktivieren.

## 9. Bereitstellung des Webservers

Auf dem physischen Host des Outside-Routers wird ein einfacher Webserver auf Port 80 gestartet.  
**Index.htm** in das Root-Verzeichnis des Webservers kopieren / aktualisieren.

## 10. Testen der Konfigurationen

- Zugriff auf das Internet vom Client aus dem Inside-Netz testen.
- Zugriff auf das Internet vom Client aus dem Outside-Netz testen
- Zugriff auf den Webserver aus dem Inside- und Labornetz (192.168.200.0/24) testen.

## 11. Einrichten der Firewall

### Outside-Router:

Wir erstellen mit `mkdir /root/bin` den Ordner, wechseln dorthin und erstellen `touch firewall.sh` im Ordner **/root/bin/** als root folgendes **firewall.sh** Script und machen dieses mit `chmod 700 firewall.sh` ausführbar:

```
#!/bin/sh
case "$1" in
stop)
echo
echo "Stopping Firewall..."
echo
iptables -F
iptables -P INPUT ACCEPT
```

FA54

P / LZ

Herr Henze

Gruppe 9

Andreas Biller, Rico Krüger

Thema: Aufbau einer DMZ

```
iptables -P FORWARD ACCEPT
iptables -P OUTPUT ACCEPT
;;
start)
echo
echo "Starting Firewall..."
echo
iptables -A OUTPUT -p icmp --icmp-type 8 -m state --state
NEW,ESTABLISHED,RELATED -j ACCEPT
iptables -A INPUT -p icmp --icmp-type 0 -m state --state
ESTABLISHED,RELATED -j ACCEPT
iptables -P INPUT DROP
iptables -P FORWARD DROP
iptables -P OUTPUT DROP
;;
*)
echo "Es wurde kein oder ein falscher Parameter übergeben"
echo "start: Zum Starten der Firewall."
echo "stop: Zum Beenden der Firewall."
esac
iptables -L
```

Dann fügen wir den Ordner **/root/bin** zur PATH-Variablen hinzu, um das Script von überall ausführbar zu machen:

```
PATH=$PATH:/root/bin
```

#### Inside-Router:

Wir erstellen mit `mkdir /root/bin` den Ordner, wechseln dorthin und erstellen `touch firewall.sh` im Ordner **/root/bin/** als root folgendes **firewall.sh** Script und machen dieses mit `chmod 700 firewall.sh` ausführbar:

```
#!/bin/bash
if [ -z "$1" ]; then
echo ""
echo "enter \"start\" or \"stop\" as an argument to start or stop the
firewall"
echo "enter \"show\" as an argument to display the current configuration"
echo ""
exit 1
else
if [ "$1" = "start" ]; then
echo ""
echo "starting firewall..."
echo ""
# set default policy to drop everything
iptables -P INPUT DROP
iptables -P FORWARD DROP
iptables -P OUTPUT DROP
# flush all filter table rules
iptables -F
# flush all user defined filter table rules
# iptables -X
# allow outgoing ping request
iptables -A OUTPUT -p icmp --icmp-type 8 -m state --state
NEW,ESTABLISHED,RELATED -j ACCEPT
```

FA54

P / LZ

Herr Henze

Gruppe 9

Andreas Biller, Rico Krüger

Thema: Aufbau einer DMZ

```
iptables -A INPUT -p icmp --icmp-type 0 -m state --state
ESTABLISHED,RELATED -j ACCEPT
# allow incoming ping request
iptables -A INPUT -p icmp --icmp-type 8 -m state --state
NEW,ESTABLISHED,RELATED -j ACCEPT
iptables -A OUTPUT -p icmp --icmp-type 0 -m state --state
ESTABLISHED,RELATED -j ACCEPT
elif [ "$1" = "stop" ]; then
    echo ""
    echo "stopping firewall..."
    echo ""
    # allow everything
    iptables -P INPUT ACCEPT
    iptables -P FORWARD ACCEPT
    iptables -P OUTPUT ACCEPT
    # flush all filter table rules
    iptables -F
elif [ "$1" = "show" ]; then
    echo ""
    echo "showing iptables:"
    echo ""
    iptables -L
else
    echo ""
    echo "unrecognized argument: $1"
    echo "exiting script..."
    echo "enter \"start\" or \"stop\" as argument to start or stop the
firewall"
    echo ""
    exit 1
fi
# show iptables
iptables -L
echo ""
echo "Good job! All done."
echo ""
exit 0
fi
```

Dann fügen wir den Ordner **/root/bin** zur PATH-Variablen hinzu, um das Script von überall ausführbar zu machen:

```
PATH=$PATH:/root/bin
```

**TODO:** allow `ssh` for using `puTTY` and `xming` through **firewall.sh**, DNS mit NAMESERVER `ip-dns-labornetz` (inside und outside) in die `/etc/resolv.conf`

## A.2 Lastenheft

Es folgt unser Lastenheft mit Fokus auf den Anforderungen:

Die Umsetzung muss folgende Anforderungen erfüllen:

### 1. DMZ

- 1.1. Die DMZ soll aus zwei virtuellen, zu Routern konfigurierten Linux-Distributionen bestehen, welche die Netze INSIDE, OUTSIDE und das DMZ-Netz miteinander verbinden.
- 1.2. Die Router sollen entsprechend des Netzplanes eingerichtet und konfiguriert werden.
- 1.3. Die DMZ soll Zugriffe auf den Webserver erlauben, aber Zugriffe auf das INSIDE-Netz verhindern. Hierzu soll auf dem Outside-Router NAT, Portforwarding und eine Firewall laufen.
- 1.4. Die Router sollen nur vom Client-Rechner her fernadministrierbar sein.

### 2. Client-Rechner

- 2.1. Der Client-Rechner im INSIDE-Netz nutzt das Betriebssystem Windows.
- 2.2. Der Webserver soll eine Webseite mit dem aktuellen Stand der Gruppe anzeigen.

### 3. Webserver

- 3.1. Der Webserver nutzt das Betriebssystem Windows. Er wird über das Tool Mini-Webserver vom Auftraggeber bereitgestellt.
- 3.2. Der Webserver im DMZ-Netz muss vom OUTSIDE-Netz über Port 80 erreichbar sein. Hierzu soll auf dem Outside-Router NAT und Port-Forwarding eingerichtet werden.
- 3.3. Der Webserver soll eine Webseite mit dem aktuellen Stand der Gruppe anzeigen.

### 4. Firewall

- 4.1. Die Firewall soll den Webserver in der DMZ über Port 80 erreichbar sein lassen.
- 4.2. Die Firewall soll SSH nur vom Admin-PC zulassen.
- 4.3. Die Firewall soll ICMP zulassen.
- 4.4. Die Firewall soll DNS zulassen.
- 4.5. Die Firewall soll RDP zulassen.
- 4.6. Die Firewall soll per Script an- und ausschaltbar sein. Hierzu muss an diversen Stellen per Script die Linux-Systemkonfiguration verändert werden

### 5. Sonstige Anforderungen

- 5.1. Das Projekt soll unter Berücksichtigung der von der IHK ausgegebenen Richtlinien für eine Projektdokumentation dokumentiert werden.
- 5.2. Es soll ein logischer Netzplan in Papierform erstellt und der Dokumentation angefügt werden.

- 5.3. Pro Person soll ein ausführliches Kompetenzportfolio erstellt werden, welches einen kritischen Überblick über unsere individuellen Kompetenzstände vor, während und nach dem Projekt liefert. Diese sollen der Dokumentation angehängt werden.
- 5.4. Die Funktionalität der Firewall soll getestet und die Ergebnisse in zwei Testprotokollen festgehalten werden. Diese sind der Dokumentation anzuhängen.

### A.3 Pflichtenheft

Unser aus den Anforderungen des Lastenheftes erstelltes Pflichtenheft:

#### 1. Musskriterien

- 1.1. Das DMZ-Netz erhält die Netzmaske 172.16.9.0/24
- 1.2. Das intere Netz erhält die Netzmaske 10.0.9.0/24
- 1.3. Die öffentliche Schnittstelle des Outside-Router erhält die IP 192.168.200.109
- 1.4. Der Outside-Router erhält als Standard-Gateway die IP 192.168.200.1
- 1.5. Der Outside-Router erhält eine statische Route für das interne und DMZ-Netz
- 1.6. Der Inside-Router erhält als Standard-Gateway das Interface des Outside-Routers, welches in die DMZ zeigt
- 1.7. Der Webserver ist über die öffentliche IP des Outside-Routers über HTTP/S von außen erreichbar
- 1.8. Der Webserver ist über die lokale IP 172.16.9.3 über HTTP/S aus dem internen Netzwerk erreichbar
- 1.9. Die Router und Windows-Clients bekommen als DNS-Server die IPs 192.168.95.40 und 192.168.95.41
- 1.10. Die Router und Windows-Clients bekommen als NTP-Server die IP 192.168.200.1
- 1.11. Die Firewall verhindert unrechtmäßigen Datentransfer zwischen den Netzen und auf den Routern
- 1.12. Der Admin-PC mit der IP 10.0.9.2 ist berechtigt mittels SSH auf die Router zuzugreifen

#### 2. Kannkriterien

- 2.1. Die Firewall lässt sich mit den Optionen `startünd` `stopän`- bzw. ausschalten
- 2.2. Die Firewall-Scripts der Router befinden sich im Verzeichnis `/root/bin`
- 2.3. Die Veränderung der Firewall-Konfiguration befindet sich jeweils im Verzeichnis `/var/log/-firewall`
- 2.4. Der Admin-PC mit der IP 10.0.9.2 ist berechtigt mittels RDP auf den Webserver zuzugreifen

## A.4 Netzpläne

Der Netzplan unserer **DMZ** in der Projektumgebung im Labor 3.1.01:

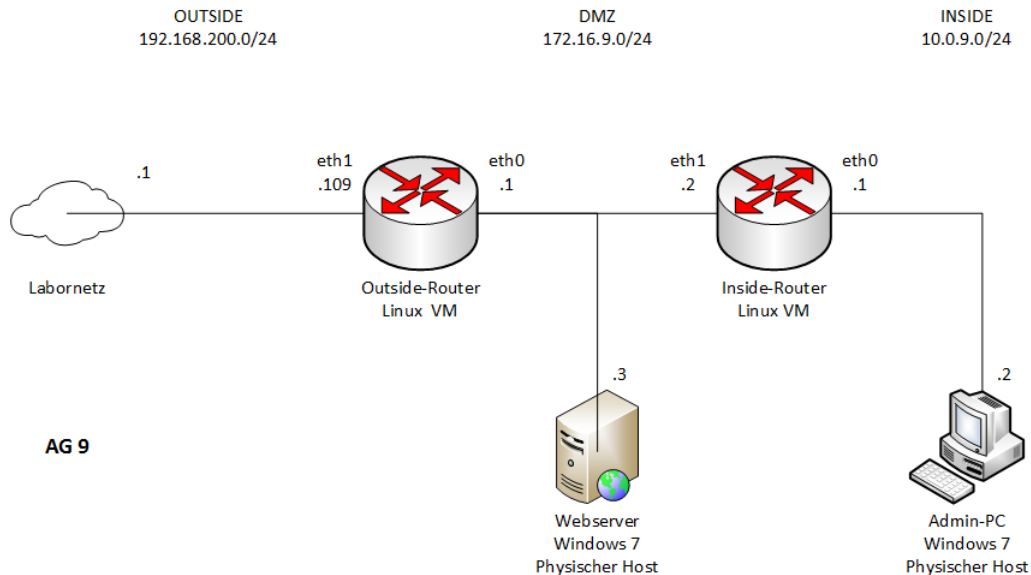


Abbildung 2: Netzplan der **DMZ** in Raum 3.1.01 (Arbeitsgruppe 9)

Der Netzplan unserer **DMZ** in der virtualisierten Testumgebung:

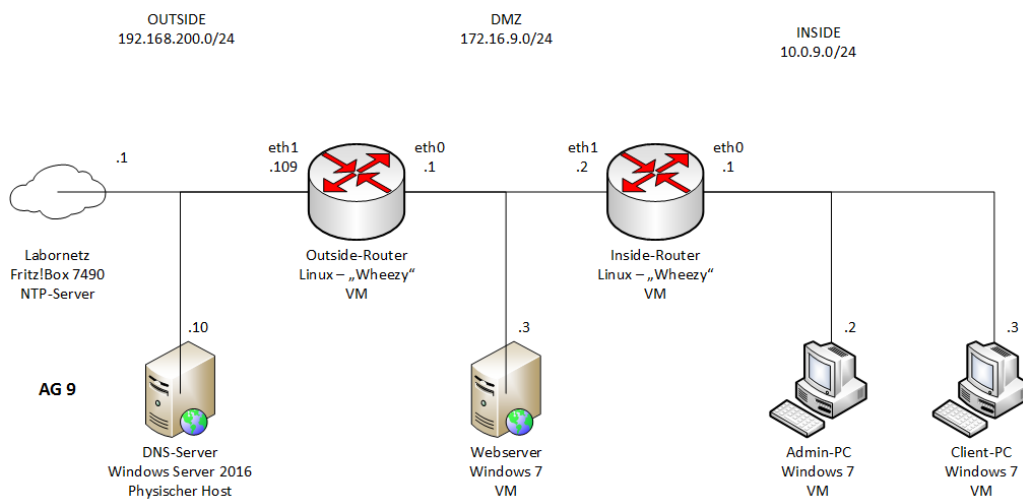


Abbildung 3: Netzplan der erweiterten **DMZ** in unserer virtuellen Testumgebung

## A.5 Kompetenzportfolios

Andreas Biller, Rico Krüger

Gruppe 9

Herr Henze

P / LZ

FA54

### Kompetenzportfolios (Thema: Aufbau einer DMZ)

#### Kompetenzportfolio - Andreas Biller:

Themen / Inhalte	Kenntnisse & Fertigkeiten			Ausblick / Fazit
	Vor dem Projekt	Während dem Projekt	Nach dem Projekt	
<b>Linux (Basics):</b> Command Line Interface, Navigation im Dateisystem, Benutzer und Gruppen, Man Pages, Berechtigungen, usw.	Im Betrieb wird mit Linux gearbeitet und ich bin u. a. für Installation und den technischen Support zuständig. Daher arbeite ich täglich auf der Kommandozeile im Linux Dateisystem, lege Benutzer an und vergebe und ändere Berechtigungen. Sich wiederholende Tätigkeiten werden mit Ansible-Playbooks oder Shell-Skripten erledigt, um Fehler zu vermeiden. In Man-Pages schaue ich regelmäßig, um benötigte Parameter für seltener benutzte Befehle herauszusuchen. Die Rechner im Betriebsnetz lassen sich remote über ssh administrieren. Mit grep, cut, pipes, xargs, wget und anderen Befehlen werden hier z. B. aus csv-Dateien urls aus Spalten ausgelesen, mit anderen Kriterien verglichen oder korrigiert. So können durch einfache Bash-Befehlsketten unter Linux sehr kreative Ergebnisse erzielt werden, für die andere schon mal Excel und dessen Sortierfunktionen benutzen müssen, um Daten aus csv-Dateien auswerten zu können.	Da ich im Betrieb jeden Tag auf der Linux-Kommandozeile arbeite gibt hier nur bedingt neues für mich zu lernen. Die Grundlagen werden eher durch Wiederholung und Anwendung aufgefrischt, wenn in einem Lab Befehle geübt werden, die im Arbeitsalltag nicht so häufig Verwendung finden. So habe ich z. B. dank der guten Erklärung des netacad-Labs die verschiedenen Eigenschaften bei gesetzten Sticky-Bits besser verstanden. Auch die Suche nach Dateien mit find fand ich erfrischend einfach, da ich im Betrieb selten nach Dateien an sich, sondern eher nach Text in Dateien mit Befehlen wie grep und Regulären Ausdrücken suche. Gerade das Arbeiten in der Bash schätze ich zunehmend dank dem schnellen Wiederholen von Befehlen durch die „Pfeil hoch“- bzw. „Pfeil runter“-Tasten oder dem interaktiven, rekursiven Suchen in der shell-History mit Strg+R.	Ich kenne mich dank vieler Aufgaben im Betrieb bereits gut im Linux-Dateisystem aus, deswegen wusste ich bereits, wo z. B. ausführbare Dateien liegen (bin), wie man die Path-Variable für neu installierte Befehle erweitert, wo Konfigurations- (etc) oder Log-Dateien (var) abgelegt werden. Ich kann dank den netacad-Labs und Chaptern vieles wiederholen und Skripte, Exit-Codes, verschiedene Outputvarianten mit Umgebungsvariablen, Pipes und Befehlen zu komplexen Anwendungen verknüpfen und weiß wo und wie ich Hilfe zu den benutzten Programmen finde.	Ich werde weiterhin auf Linux arbeiten und auch zuhause nutze ich in meiner Freizeit mindestens einen Rechner, auf dem Linux installiert ist. Grundwissen wird also immer wieder angewendet werden und sich festigen und weiterentwickeln, bis ich hoffentlich nicht mehr bei so vielen der seltener benutzten Befehlen in die Anleitung schauen muss.
<b>Linux (Advanced):</b> Shell-Skripte & Befehle, Umgebungsvariablen, Editoren (vi, nano, etc.), Konfigurationsdateien, usw.	Andere Aufgaben im Betrieb werden durch Skripte (.sh oder .rb) zu wiederholbaren Prozessen, CSV-Listen mit Befehlen wie grep, cut, xargs und dem Pipe-Operator ausgewertet. Auch Webseiten für Kunden erstelle ich dort über Commandline-Tools indem Markdown-Dateien mit jekyll und liquid templates zu html umgewandelt wird. Umgebungsvariablen benutzen wir in der Entwicklung und Produktion um die verwendeten Passwörter aus unserem Programmcode herauszuhalten. Zum Editieren benutze ich in der bash meist nano, ansonsten benutze ich der zusätzlichen Funktionalität wegen lieber einen graphischen Editor wie gedit oder sublime.	Hier war der Nutzen schon größer, da ich im Betrieb aufgrund der für unsere Plattform genutzten Programmiersprache meine Skripte hier meist in ruby schreibe und bash-Skripte nur dann verwende, wenn ich unbedingt einige der normalen Linux-Befehle für etwas Spezielles benötige. Aber auch hier ist vieles Wiederholung, gehört doch die Arbeit im CLI mit Umgebungsvariablen, Skripten, allgemeinen Befehlen wie git oder eher ruby-spezifischen wie rvm, bundle, rails, rubocop, etc. zu meinen täglichen Aufgaben. Zur Konfiguration editiere ich auch schon mal Dateien wie sudoers.d um Benutzern sudo-Rechte zu entziehen. Das Arbeiten in der shell fällt mir in der Arbeit dank um eigene Aliase erweiterter bash.rc einfacher als in einem unmodifizierten Debian wie dem genutzten	Ich habe viele Dinge wiederholt, die im Betrieb auch schon relevant waren, einige Sachen besser verstanden und gerade beim Skripten in der Shell Dinge angewendet, die im Betrieb wegen der Präferenz ruby zu benutzen häufig nicht in Shell-Skripten umgesetzt werden. Obwohl es mich sehr gereizt hätte, besser mit vi umgehen zu lernen, hat mir auch dieser kurze Versuch damit editieren zu wollen gezeigt, warum ich hier wenn möglich immer noch zu nano oder anderen, weniger komplexen Editoren greife.	Auch hoffe ich irgendwann die Zeit aufzubringen mich länger und intensiver mit vi oder vim zu beschäftigen. Da unser Betrieb stetig wächst und ich einen Linux-Rechner nach dem anderen konfiguriere, werde ich auch hier weitere Fertigkeiten entwickeln und mit dem gelernten Wissen Skripte schreiben und in Übung bleiben.
<b>Netzwerk:</b> Planung & Darstellung, Dienste, Routingtabellen, Statische Routen,	Netzwerke kenne ich hauptsächlich aus dem privaten Bereich, z. B. von der Konfiguration des Routers bzw. Netzwerkdruckers zuhause oder von kleinen Netzwerken über Hubs auf LAN-Parties. Mit ping habe ich bereits das eine oder andere mal die Funktionalität von LAN- und Internetverbindungen getestet, auch wenn ich nicht wußte, was dort alles genau passiert.	Dank der vorherigen Konfiguration von Netzwerken im ITS-Unterricht mit Packet-Tracer in der cisco-Syntax, auf Windows-Rechnern in deren Dialekt und nun dem Übersetzen bzw. Wiederholen der bereits bekannten Befehle nach Linux lerne ich hier gefühlt am meisten, da ich bei Befehlen wie ip route sehe, wie ähnlich sich die unterschiedlichen Systeme sind und dass sich meistens nur einige Schlüsselwörter oder die Schreibweise der Parameter	Ich habe ein besseres Verständnis davon erhalten, wie die Kommunikation in einem bzw. die Konfiguration eines Netzwerkes an sich funktioniert, egal auf welchem System man diese vornimmt. Dennoch fehlt mir noch einiges an Erfahrung, da ich im Fall von Problemen immer noch lange Suchen muß, bis ich das Problem eingrenzen kann.	Auch im Bereich Netzwerk kann ich das eine oder andere Gelernte aus diesem Projekt im Betrieb bestimmt noch praktisch umsetzen, da durch unseren momentanen Wachstum auch das verwendete Netzwerk immer wieder erweitert werden muss.



### Gruppe 9

**Herr Henze**

P / LZ

FA54

### Kompetenzportfolios ( Thema: Aufbau einer DMZ )

Testen (ping), usw.		unterscheiden, die zugrundeliegenden Prinzipien jedoch gleich sind.		
<b>NAT:</b> Webserver aufsetzen, Schnittstellen konfigurieren, Konfigurationen speichern, usw.	Ich habe schon eigene Webserver (apache2) konfiguriert und betrieben, um Webseiten lokal zu testen und mein altes Portfolio selbst zu hosten, bzw. Portforwarding im Router eingerichtet um online mit Freunden spielen zu können. Den Router sowie die Schnittstellen habe ich bisher über graphische Oberflächen (unter Windows) eingerichtet.	Da ich bereits eigene Erfahrungen mit dem Aufsetzen und Konfigurieren von apache2 sowohl unter Windows wie unter Linux habe, ist das miniwebserver Tool keine Quelle neuer Lernerfahrungen (wenn auch einfach zu benutzen). NAT und Portforwarding sind dank dem gezielten Suchen nach Informationen im Internet und dem in ITS Erlernten schnell in die benötigten Konfigurationsdateien (/proc/sys/net/ipv4/ip_forward) geschrieben und über den auch für die Firewall benutzten Befehl iptables in der FORWARD-Chain mit dem Parameter MASQUERADE eingerichtet.	Auch im Betrieb arbeite ich unter anderem mit einem lokalen apache2 Webserver, um zu bearbeitende Webseiten in html oder php über localhost anzeigen zu lassen. Auch jekyll beim Erstellen von Webseiten sowie das „ruby on rails“-Framework bietet neben einer CLI- auch eine über den Browser geservte Entwicklungsumgebung, in der einiges konfiguriert werden muss, bis alles läuft wie es soll. Das erlangte Wissen aus dem Projekt wird hier in der einen oder anderen Form sicher immer wieder auftauchen.	NAT muss ich betrieblich nicht wirklich einrichten, allerdings kann ich erlerntes Wissen bestimmt auf die eine oder andere Art bei der Arbeit an bzw. der Konfiguration von unserer bei Heroku gehosteten Webapp weiter verwenden.
<b>Firewall:</b> Skript erstellen bzw. anpassen, iptables verstehen & benutzen (Tables, Chains, Rules), usw.	Im Rahmen mehrerer Weiterbildungsmaßnahmen des Jobcenters durfte ich bereits vor der Arbeit im heutigen Betrieb als Aushilfe im Server- und IT-Bereich an der Rixdorfer Grundschule für einige Zeit Grundlagen in der PC- und Linux-Administration erlernen, unter anderem auch die prinzipielle Arbeitsweise und das Erstellen von Filterregeln für die einzelnen Chains bei einer Firewall, allerdings in einer graphischen Umgebung (webmin). Und leider musste ich diese Fähigkeiten dann nie wirklich praktisch anwenden und hatte das meiste davon bis heute wieder vergessen.	Da wir mit den Linux-Grundlagen und dem Erstellen des Netzwerkes die erste Hälfte des Projektes beschäftigt waren und auch in ITS erst zum zweiten Halbjahr etwas zu Access-Control-Lists gelernt haben, hatte ich nur dank dem beim NAT verwendeten Befehl iptables etwas zur Firewall gelernt, ohne dies jedoch zu wissen.	Nach dem Projekt habe ich die Bedeutung der einzelnen Chains für den Routing-Prozess verstanden und weiß wieder wie die iptable Regeln zusammen mit den Standard-Policies als Firewall verwendet werden können und wie ich diese mit Hilfe unseres Scriptes in der post-up bzw. pre-down Sequenz der interfaces speichern und so sowohl bei einem Neustart durch Befehle wie ifup/ifdown sowie beim Booten des Systems aktivieren bzw. auch wieder deaktivieren kann.	Das Wissen um die Funktionsweise von Firewall-Regeln und grundlegende IT-Sicherheit sind wichtig, aber die Übung im Erstellen einer Dokumentation nach den IHK-Richtlinien für das bevorstehende Prüfungsprojekt sind dagegen nahezu unbezahlbar und werden mir von all dem Gelernten mit Sicherheit am ehesten noch unschätzbare Dienste leisten.

### Kompetenzportfolio - Rico Krüger:

Themen / Inhalte	Kenntnisse & Fertigkeiten			Ausblick / Fazit
	Vor dem Projekt	Während dem Projekt	Nach dem Projekt	
<b>Linux</b> (Shell: Navigation, Befehle, Scripte, Editor)	Auf der Arbeit oder zu Hause arbeite ich selten in der Konsole oder mit Linux. Über die Verzeichnisstruktur und wichtige Dateien weiß ich kaum etwas. Die meisten Befehle und deren Optionen sind mir nicht geläufig. Ich habe in Linux noch kein Script geschrieben. Ich nutzte bisher stets einen grafischen Editor.	Ich nutze anfangs viel Google um nach Befehlen oder Dateien zu suchen. Die Befehle werden geläufiger und ich versuche mich an die man pages für Kommandos zu gewöhnen. Um Dateien schnell zu finden ist <code>find / -name [name]</code> sehr hilfreich. Anfangs wechsle ich noch in den Ordner um Dateien zu öffnen. Das ist nicht nötig. Alle Dateien lassen sich von überall her ansprechen. Das ist bei grafischen Oberflächen nicht gegeben. Ein Script wird stets mit <code>#!/[Path][Shell]</code> eingeleitet. Um ein Script auszuführen muss man die Berechtigung mit <code>chmod +x [Path][Script]</code> ändern. Zum Editieren wechsle ich zwischen nano und vi, welcher mich an das less Kommando erinnert. Aufgerufen wird ein Script über <code>./[Path][Script]</code> . Kommandos, die man aufrufen werden nicht gespeichert, solange die Änderung	Ich fühle mich in der Shell wesentlich wohler und finde mich in der Verzeichnisstruktur zurecht. Das Arbeiten in der Konsole bringt viele Vorteile mit sich. Getätigte Befehle lassen schnell wiederholen und man muss sich nicht lange durch irgendwelche Fenster und Verzeichnisse navigieren. Viele Programme haben <code>.dotfiles</code> , mithilfe man diese konfigurieren kann. Einige Befehle haben sich eingeprägt. Um mir Optionen anzeigen zu lassen benutze ich <code>--help</code> und für genauere Informationen man. Nichtsdestotrotz greife ich noch, vor allem bei mir noch unbekannten Befehlen, auf Google zurück. Bevor man ein Script mit mehreren Befehlen schreibt, kann man die einzelnen Befehle erstmal problemlos in der Shell testen und hier auch nach Hilfe suchen. Alle Befehle haben Standardkanäle für Ausgabe(0), Eingabe(1) und Fehler(2).	Ich würde gerne weiter mit Linux arbeiten um meine Fähigkeiten im Scripting und Nutzen des Shell zu verbessern und so auch meine Produktivität zu steigern. Zudem ist Linux „sauberer“ als Windows, wo schon eine schier unendliche Anzahl an Diensten, Programmen und Bibliotheken vorinstalliert ist und man von vornherein jeglichen Überblick verloren hat, welche Dienste und Programme schon vorinstalliert sind und was diese eigentlich schon alles definieren

Andreas Biller, Rico Krüger

Gruppe 9

Herr Henze

P / LZ

FA54

Kompetenzportfolios ( Thema: Aufbau einer DMZ )

		nicht in eine Datei geschrieben werden. Die Interfaces konfiguriere ich in /etc/interfaces. Hier kann man auch DSN, statische Routen und NAT konfigurieren und speichern. Diese Datei wird beim Booten von Linux geladen.	Dieses Verhalten lässt sich mittels [x]> ändern. Ich bevorzuge vi nano, da ich so auch gleich die Navigation mit less verinnerliche und ich beim Editieren die <i>home row</i> nicht verlassen muss. Die DNS-Server trägt man in der /etc/resolv.conf ein. Um das mittels eines Scripts zu lösen kann man >> benutzen. Für Routen und NAT erstellt man am besten ein Script.	und ausführen. Mir gefällt die Logik von vi und würde gerne die diversen Eingaben aus dem Effekt beherrschen. So könnte ich wesentlich schneller arbeiten ohne auch nur die Tastatur verlassen zu müssen.
<b>Netzwerk</b>  (NAT, Portforwarding, Statische Routen)	Ein Netzwerk zu konfigurieren haben ich bisher nur unter Windows mithilfe einer GUI gemacht. Wofür Statische Routen, NAT usw. gebraucht werden, wusste ich zwar jedoch kannte ich nicht den genauen Inhalt und wie ich diese unter Linux konfiguriere.	Das wichtigste ist erstmal, dass ich lerne dass ich ohne weiteres ein Linux-System als Router konfigurieren kann. Um IP-Forwarding zu aktivieren nutze ich echo "1" > /proc/sys/net/ipv4/ip_forward, für die statische Route ip route add -net [Netz] netmask [Netzmaske] gw [gw]. Um NAT zu konfigurieren für die Netzwerke nutze ich iptables -A FORWARD -o eth1 -s 172.16.9.0/24 --ctstate NEW -j ACCEPT. NAT konfiguriere ich mittels iptables -t nat -A POSTROUTING -o eth1 -j MASQUERADE. Danach muss ich ggf. die Interfaces über ifup / ifdown neustarten. Speicher ich diese Befehle in interfaces kann ich mit post-up / pre-down dafür sorgen, dass diese Konfiguration automatisch beim Booten geladen wird	Ich habe diese Befehle in mein Firewall.sh Script geschrieben und bin jetzt in der Lage, diese Befehle automatisiert ausführen zu lassen. Mithilfe von NAT werden interne Adressen in eine öffentliche Adresse des Routers gewandelt. Der Router speichert diese Zuordnung in eine Tabelle. Dort werden die Anfragen mit Hilfe von Portnummern gespeichert um diese wieder dem Client und dem zugehörigen Dienst zuordnen zu können.	Ich habe mein Wissen definitiv gesteigert und werde es auch noch weiter. Ich finde es hat Spaß gemacht und hilft mir auch auf der Arbeit. Ich werde mir mal privat ein kleines Netzwerk erstellen und dafür ein raspberry pi nutzen.
<b>Firewall</b>	Wofür eine Firewall gut ist, war mir schon bewusst, jedoch nicht nach welchem Prinzip sie arbeitet. Auch fand jegliche Konfiguration stets über eine GUI statt.	Ich lerne, dass solche Befehle in einer Liste abgearbeitet werden. Diese werden mit iptables -A angehängt. Dabei unterscheidet der Router in der INSIDE, OUTSIDE und FORWARD-chain. Man kann hier die Ports, das Protokoll sowie die Quell- und Zieladressen definieren. Dabei spielt es immer eine Rolle aus welcher Richtung die Anfrage kommt und an wen sie gerichtet ist. Diese Regeln werden von oben nach unten abgearbeitet.	Ich habe ein Firewall.sh Script. In diesen sich jetzt die NAT-Regeln und ein paar weitere Berechtigungen für DNS, HTTP/S und SSH. Ich denke, ich habe das Prinzip der Abarbeitung der ACLs verstanden und kann die verschiedenen chains auseinanderhalten.	Mein Überblick hat sich erhöht und ich kann das Prinzip gut nachvollziehen. Inwieweit ich diese Thematik jedoch noch vertiefen kann ich zum jetzigen Zeitpunkt noch nicht sagen, könnte mir aber vorstellen mal für zu Hause mir eigene Firewall aufzusetzen.
<b>VMs</b>	Ich habe vorher schon mit VMware und HyperV gearbeitet.	Ich habe VMWare Player genommen um Linux auszuführen.	Ich habe über Virtualisierung nichts Neues gelernt.	Ich werde wohl mehr mit HyperV, aufgrund der kostenlosen Lizenz arbeiten. So kann ich auch ein komplettes virtuelles Netzwerk errichten.

## B Testdokumentation

Bauteil	Spezifikation
Prozessor	Intel® Core™ i7-6700K Prozessor 8 MB Cache, 4.20 GHz
RAM	2x16GB DDR4-2400 DIMM CL15 Dual
Speicher	1x250GB SSD + 1x500GB SSD
Betriebssystem	Windows Server 2016 Datacenter

tab:SysteminformationSysteminformation.tex

Tabelle 9: Hardwaredetails des Testsystems

### B.1 Aufbau der Testumgebung

Die zur Umsetzung dieses Kapitels benötigten Informationen entstammen unter anderem den hilfreichen Artikeln folgender Webseiten: [BLOG.WORKINGHARDINIT.WORK](#) [2017]

#### B.1.1 Implementierung der Virtuellen Maschinen

Im Server-Manager fügen wir über *Verwalten > Rollen und Features hinzufügen* den Hyper-V-Manager hinzu indem wir dem Assistenten folgen. Dieser gestattet es virtuelle Maschinen und Netzwerke zu installieren. Als nächstes wird eine neue virtuelle Linux (Debian 7.1) Maschine (Generation 1) aus einem Image erstellt. Dies geschieht mit Hilfe eines Assistenten. Sie bekommt einen virtuellen Prozessor und 1 GB Arbeitsspeicher. Des weiteren wird bei der Installation eine 5 GB große Festplatte für die Maschine erstellt und ihr zugewiesen. Als virtuellen Switch weisen wir ihr vorläufig den Netzwerkadapter des Hosts zu. Somit besitzt unsere Linux-Virtual Machine (VM) Internet. Um sie zu installieren, startet man nun die Maschine und verbindet sich zu ihr. Danach folgt man wie gewohnt den Installationsschritten wie bei einer physischen Maschine. Danach installieren wir den Network Time Protocol (NTP)-Service. Ist die Grundkonfiguration fertig, wird die Maschine ausgeschaltet. Die Installation der Windows 7 VM erfolgt analog zu die der Linux VM. Wir vergeben jedoch 4 GB RAM und erstellen eine mindestens 30 GB große virtuelle Festplatte. Nach der Installation wird die Firewall wie in *Implementierungsphase* eingerichtet. Zusätzlich werden noch nützliche Software wie putty oder winscp heruntergeladen. Nach der Grundkonfiguration der beiden VMs können diese nun dupliziert werden. Dazu muss man die virtuelle Maschine erst exportieren, um sie danach wieder zu importieren. Beim Import sollte man darauf achten, dass man *eine neue eindeutige Identification (ID)* erstellt. Nachdem starten der importierten Maschine wird als erstes der Hostname geändert, um sie von der Originalen zu unterscheiden und um DNS-Konflikte zu vermeiden.

### B.1.2 Implementierung des virtuellen Netzwerkes

Virtuelle Netzwerke werden über das Hinzufügen virtueller Switche an den Netzwerkadaptern der virtuellen Maschine erstellt. Auf diesen lassen sich auch Virtual LAN (VLAN)s einrichten. Die Installation eines solchen Switch wird ebenfalls vom Hyper-V-Manager mit einem Assistenten bereitgestellt. Für Testzwecke werden 2 *private* Switche erstellt, da diese die direkte Kommunikation mit dem Host unterbinden und somit nicht die Router umgangen werden. Diese erhalten den Namen DMZ- bzw. LAN-Switch. Ein *öffentlicher* Switch ist bereits vorhanden. Mit diesem ist der physische Netzwerkadappter des Hosts verbunden. Diese werden dann den VMs entsprechend des Netzplänes zugeordnet. Für die Linux-VMs, die als Router fungieren, muss evtl. noch ein zweiter Netzwerkadapter hinzugefügt werden. Nun können die Router und Clients (Siehe Bild und Implementierung) konfiguriert werden.

### B.1.3 Implementierung des DNS-Servers

Der DNS-Server wird ebenfalls über den Server-Manager (unter *Rollen und Features hinzufügen*) installiert. Diesen kann man nun über den DNS-Manager verwalten. Es genügt eine *Forward-Lookup*-Zone zu erstellen. Als Zonennamen wählen wir *fritz.box* da bereits das Standard-Gateway darauf verweist. Dies ist die Domäne bzw. das DNS-Suffix. Dieses Suffix wird auf den Windows-VMs in den Internet Protokoll (IP)v4-Einstellungen des Netzwerkadapters nachgetragen. Auf den Linux-VMs tragen wir dies zusätzlich in die */etc/resolv.conf* vor unserem DNS-Server ein. **siehe resolv.conf oder selber schreiben** Über den DNS-Manager werden im Anschluss noch in der Zone *fritz.box* unsere VMs (A-Record) mit Namen und IP-Adressen eingetragen. Siehe DNSManager.png.

### B.1.4 Testen der Firewall

Nachdem das Firewall-Script auf die Router kopiert und die DNS-Server angepasst wurden, kann mit den Tests begonnen und die Firewall ggf. angepasst werden. Dazu speichern wir den Verlauf der erstellten Regeln als Log-Ausgabe in */var/log/firewall/firewallConfig* ab. Die Ergebnisse unserer Tests finden sich als Übersicht in den folgenden Tabellen der Testprotokolle:

### B.1.5 Testprotokolle

B Testdokumentation

Service	Command	Source-IP	Destination-IP	Soll	Ist
ICMP	ping	10.0.9.2	10.0.9.1	ja	ja
ICMP	ping	10.0.9.3	10.0.9.1	ja	ja
ICMP	ping	10.0.9.2	172.16.9.1	ja	ja
ICMP	ping	10.0.9.3	172.16.9.1	ja	ja
ICMP	ping	172.16.9.3	172.16.9.1	ja	ja
ICMP	ping	172.16.9.3	172.16.9.2	ja	ja
ICMP	ping	10.0.9.3	192.168.200.10	ja	ja
ICMP	ping	192.168.200.10	10.0.9.3	ja	ja
ICMP	ping	192.168.200.10	172.16.9.3	ja	ja
ICMP	ping	192.168.200.10	192.168.200.109	ja	ja
ICMP	ping	10.0.9.3	8.8.8.8	ja	ja
HTTP	http://172.16.9.3	192.168.200.10	192.168.200.109	ja	ja
HTTP	http://172.16.9.3	192.168.200.10	172.16.9.3	ja	ja
HTTP	http://172.16.9.3	10.0.9.3	192.168.200.109	ja	ja
HTTP	http://172.16.9.3	10.0.9.3	172.16.9.3	ja	ja
NTP	w32tm /stripchart /computer:192.168.200.1	10.0.9.2	192.168.200.1	ja	ja
NTP	w32tm /stripchart /computer:192.168.200.1	172.16.9.3	192.168.200.1	ja	ja
NTP	ntpq -p	192.168.200.109	192.168.200.1	ja	ja
RDP	mstsc.exe	10.0.9.2	172.16.9.3	ja	ja
RDP	mstsc.exe	10.0.9.3	172.16.9.3	ja	ja
SSH	putty	10.0.9.2	10.0.9.1	ja	ja
SSH	putty	10.0.9.2	172.16.9.1	ja	ja
SSH	putty	10.0.9.3	10.0.9.1	ja	ja
SSH	putty	10.0.9.3	172.16.9.1	ja	ja
SSH	putty	172.16.9.3	172.16.9.1	ja	ja
SSH	putty	172.16.9.3	172.16.9.2	ja	ja
DNS	nslookup 172.16.9.1	10.0.9.3	192.168.200.10	ja	ja
DNS	nslookup Inside-Router	172.16.9.3	192.168.200.10	ja	ja
DNS	nslookup 10.0.9.1	172.16.9.2	192.168.200.10	ja	ja
DNS	nslookup Client-PC	192.168.200.109	192.168.200.10	ja	ja

Tabelle 10: Aus - Aus

Service	Command	Source-IP	Destination-IP	Soll	Ist
ICMP	ping	10.0.9.2	10.0.9.1	ja	ja
ICMP	ping	10.0.9.3	10.0.9.1	nein	nein
ICMP	ping	10.0.9.2	172.16.9.1	ja	ja
ICMP	ping	10.0.9.3	172.16.9.1	nein	nein
ICMP	ping	172.16.9.3	172.16.9.1	ja	ja
ICMP	ping	172.16.9.3	172.16.9.2	nein	nein
ICMP	ping	10.0.9.3	192.168.200.10	ja	ja
ICMP	ping	192.168.200.10	10.0.9.3	nein	nein
ICMP	ping	192.168.200.10	172.16.9.3	ja	ja
ICMP	ping	192.168.200.10	192.168.200.109	ja	ja
ICMP	ping	10.0.9.3	8.8.8.8	ja	ja
HTTP	http://172.16.9.3	192.168.200.10	192.168.200.109	ja	ja
HTTP	http://172.16.9.3	192.168.200.10	172.16.9.3	ja	ja
HTTP	http://172.16.9.3	10.0.9.3	192.168.200.109	nein	nein
HTTP	http://172.16.9.3	10.0.9.3	172.16.9.3	ja	ja
NTP	w32tm /stripchart /computer:192.168.200.1	10.0.9.2	192.168.200.1	ja	ja
NTP	w32tm /stripchart /computer:192.168.200.1	172.16.9.3	192.168.200.1	ja	ja
NTP	ntpq -p	192.168.200.109	192.168.200.1	ja	ja
RDP	mstsc.exe	10.0.9.2	172.16.9.3	ja	ja
RDP	mstsc.exe	10.0.9.3	172.16.9.3	nein	nein
SSH	putty	10.0.9.2	10.0.9.1	ja	ja
SSH	putty	10.0.9.2	172.16.9.1	ja	ja
SSH	putty	10.0.9.3	10.0.9.1	nein	nein
SSH	putty	10.0.9.3	172.16.9.1	ja	ja
SSH	putty	172.16.9.3	172.16.9.1	ja	ja
SSH	putty	172.16.9.3	172.16.9.2	nein	nein
DNS	nslookup 172.16.9.1	10.0.9.3	192.168.200.10	ja	ja
DNS	nslookup Inside-Router	172.16.9.3	192.168.200.10	ja	ja
DNS	nslookup 10.0.9.1	172.16.9.2	192.168.200.10	ja	ja
DNS	nslookup Client-PC	192.168.200.109	192.168.200.10	ja	ja

Tabelle 11: Aus - An

Service	Command	Source-IP	Destination-IP	Soll	Ist
ICMP	ping	10.0.9.2	10.0.9.1	ja	ja
ICMP	ping	10.0.9.3	10.0.9.1	ja	ja
ICMP	ping	10.0.9.2	172.16.9.1	ja	ja
ICMP	ping	10.0.9.3	172.16.9.1	nein	nein
ICMP	ping	172.16.9.3	172.16.9.1	nein	nein
ICMP	ping	172.16.9.3	172.16.9.2	ja	ja
ICMP	ping	10.0.9.3	192.168.200.10	ja	ja
ICMP	ping	192.168.200.10	10.0.9.3	nein	nein
ICMP	ping	192.168.200.10	172.16.9.3	nein	nein
ICMP	ping	192.168.200.10	192.168.200.109	nein	nein
ICMP	ping	10.0.9.3	8.8.8.8	ja	ja
HTTP	http://172.16.9.3	192.168.200.10	192.168.200.109	ja	ja
HTTP	http://172.16.9.3	192.168.200.10	172.16.9.3	ja	ja
HTTP	http://172.16.9.3	10.0.9.3	192.168.200.109	nein	nein
HTTP	http://172.16.9.3	10.0.9.3	172.16.9.3	ja	ja
NTP	w32tm /stripchart /computer:192.168.200.1	10.0.9.2	192.168.200.1	ja	ja
NTP	w32tm /stripchart /computer:192.168.200.1	172.16.9.3	192.168.200.1	ja	ja
NTP	ntpq -p	192.168.200.109	192.168.200.1	ja	ja
RDP	mstsc.exe	10.0.9.2	172.16.9.3	ja	ja
RDP	mstsc.exe	10.0.9.3	172.16.9.3	nein	nein
SSH	putty	10.0.9.2	10.0.9.1	ja	ja
SSH	putty	10.0.9.2	172.16.9.1	ja	ja
SSH	putty	10.0.9.3	10.0.9.1	ja	ja
SSH	putty	10.0.9.3	172.16.9.1	ja	ja
SSH	putty	172.16.9.3	172.16.9.1	nein	nein
SSH	putty	172.16.9.3	172.16.9.2	ja	ja
DNS	nslookup 172.16.9.1	10.0.9.3	192.168.200.10	ja	ja
DNS	nslookup Inside-Router	172.16.9.3	192.168.200.10	ja	ja
DNS	nslookup 10.0.9.1	172.16.9.2	192.168.200.10	ja	ja
DNS	nslookup Client-PC	192.168.200.109	192.168.200.10	ja	ja

Tabelle 12: An - Aus

Service	Command	Source-IP	Destination-IP	Soll	Ist
ICMP	ping	10.0.9.2	10.0.9.1	ja	ja
ICMP	ping	10.0.9.3	10.0.9.1	ja	ja
ICMP	ping	10.0.9.2	172.16.9.1	ja	ja
ICMP	ping	10.0.9.3	172.16.9.1	nein	nein
ICMP	ping	172.16.9.3	172.16.9.1	nein	nein
ICMP	ping	172.16.9.3	172.16.9.2	ja	ja
ICMP	ping	10.0.9.3	192.168.200.10	ja	ja
ICMP	ping	192.168.200.10	10.0.9.3	nein	nein
ICMP	ping	192.168.200.10	172.16.9.3	nein	nein
ICMP	ping	192.168.200.10	192.168.200.109	nein	nein
ICMP	ping	10.0.9.3	8.8.8.8	ja	ja
HTTP	http://172.16.9.3	192.168.200.10	192.168.200.109	ja	ja
HTTP	http://172.16.9.3	192.168.200.10	172.16.9.3	ja	ja
HTTP	http://172.16.9.3	10.0.9.3	192.168.200.109	nein	nein
HTTP	http://172.16.9.3	10.0.9.3	172.16.9.3	ja	ja
NTP	w32tm /stripchart /computer:192.168.200.1	10.0.9.2	192.168.200.1	ja	ja
NTP	w32tm /stripchart /computer:192.168.200.1	172.16.9.3	192.168.200.1	ja	ja
NTP	ntpq -p	192.168.200.109	192.168.200.1	ja	ja
RDP	mstsc.exe	10.0.9.2	172.16.9.3	ja	ja
RDP	mstsc.exe	10.0.9.3	172.16.9.3	nein	nein
SSH	putty	10.0.9.2	10.0.9.1	ja	ja
SSH	putty	10.0.9.2	172.16.9.1	ja	ja
SSH	putty	10.0.9.3	10.0.9.1	ja	ja
SSH	putty	10.0.9.3	172.16.9.1	ja	ja
SSH	putty	172.16.9.3	172.16.9.1	nein	nein
SSH	putty	172.16.9.3	172.16.9.2	ja	ja
DNS	nslookup 172.16.9.1	10.0.9.3	192.168.200.10	ja	ja
DNS	nslookup Inside-Router	172.16.9.3	192.168.200.10	ja	ja
DNS	nslookup 10.0.9.1	172.16.9.2	192.168.200.10	ja	ja
DNS	nslookup Client-PC	192.168.200.109	192.168.200.10	ja	ja

Tabelle 13: An - An



## B.2 Firewall-Skripte

### B.2.1 firewall.sh (auf dem Outside-Router)

```
1 #!/bin/bash
2 # Bourne- Again Shell#
3
4 # =====
5 # === Aufgabenstellung =====
6 # =====
7
8 # 1. Datei in " firewall .sh" umbenennen
9 # 2. Datei ausfuehrbar machen: Auf der Kommandozeile das Skript starten mit: ./firewall.sh ENTER
10 # 3. Wenn Fehlermeldung (das Skript laeuft gar nicht) Konvertierung mit "dos2unix Dateiname"
11 #-----Aufgabenstellung
12
13 # 1.Passen Sie dieses Firewall-Skript an die folgende Aufgabenstellung an.
14 # 2.Ihre unter Linux laufenden Rechner (Router/Firewalls) sollen mindestens folgendermassen konfiguriert sein:
15 # a) Jeder Rechner (Webserver, Host, zwei Linux-Router) Ihrer Arbeitsgruppe muss die eigene Zeit mit einem
16 #    Zeitserver
17 #    synchronisieren koennen. Nehmen Sie auf jeden Fall den schulinternen Zeitserver (Standardgateway:192.168.200.1)
18 #    , da die
19 #    externen evt. nicht erreichbar sind.
20 # b) Ihr Webserver soll von ueberall (eigenes LAN und fremde Netzwerke) nur auf Port 80 erreichbar sein.
21 # c) Ping (echo-request) soll fuer alle Rechner des eigenen Netzes (Intern) erlaubt sein und auch echo-reply
22 #    Antworten
23 #    aus dem Internet erhalten. (z.B. ping 141.1.1.1, ping 8.8.8.8)
24 # d) Die Wartung der Linux Router mittels 'ssh' soll nur von einem ausgezeichneten Rechner Ihres eigenen LANs
25 #    erlaubt sein.
26 # Die Linux-Router sind vor allen anderen Zugriffen zu schuetzen!!
27 # e) Der/die Rechner des eigenen LANs sollen per "http" in das Internet (google, gmx etc.) kommen koennen.
28 # f) Die "Default Policy" der Firewalls muss auf "DROP" stehen. (Alles was nicht explizit erlaubt ist , ist verboten
29 #    !!)
30 # g) Darueber hinaus lassen Sie sich in Ihrer Kreativitaet nicht einschraenken.
31 #
32 # 3.Tipp: Sie sollten sich ein zweites, kurzes Skript schreiben, das die Firewall komplett oeffnet und alle Regeln
33 #    loescht, um
34 #    jederzeit testen zu koennen, ob Ihr Netzwerk noch steht.
35 #
36 #-----Ende--Aufgabenstellung
37
38 # =====
39 # === Part 1: Variablen =====
40 # =====
41 echo " -- Variablen werden gesetzt"
42
43 # Pfad zu iptables
44 IPTABLES=/sbin/iptables
```

```
39
40 # Macht Linux-Maschine zu einem Router
41 echo "1" > /proc/sys/net/ipv4/ip_forward
42
43 # Interfaces
44 iINT=eth0
45 iEXT=eth1
46
47 # Definition DNS
48 DNS=("192.168.95.40/32 192.168.95.41/32")
49
50 # Timeserver: hier Standardgateway
51 TimeSrv=192.168.200.1
52
53 # Der Rechner, auf dem die Firewall (Inside) laufen soll , hier die VMWare
54 LinuxInside_in=10.0.9.1
55 LinuxInside_dmz=172.16.9.2
56
57 # Der Rechner, auf dem die Firewall (Outside) laufen soll , hier die VMWare
58 LinuxOutside_out=192.168.200.109
59 LinuxOutside_dmz=172.16.9.1
60
61 # Rechner fuer Fernwartung z.B. mit ssh, hier der Windowswirt (XP, Win7 o.ae.)
62 AdminPC=10.0.9.2
63
64 # Webserver
65 Webserver=172.16.9.3
66
67 # Das DMZ-Netz
68 DMZ=172.16.9.0/24
69
70 # Das LAN-Netz
71 LAN=10.0.9.0/24
72
73 # Protokolle
74 protocols=("tcp" "udp")
75
76 # DNS Ports
77 dnsPorts=("53" "853")
78
79 # HTTP/S Port
80 webPorts=("80" "443")
81
82 # ntp Port
83 ntpPort=123
84
85 # rdp Port
86 rdpPort=3389
87
88 # Pfad zur aktuellen Firewall Konfiguration
89 lopPath="/var/log/firewall/firewallconfig "
```

```
90
91 #-----Ende---Variablen setzen
92     -----
93
94 # =====
95 # =====
96 # === Starten / Stoppen / Hilfe =====
97 # =====
98 # =====
99 case "$1" in
100
101
102 # =====
103 # =====
104 # === Firewall stoppen =====
105 # =====
106 # =====
107 stop)
108
109 # =====
110 # === Part 2: Default Policy setzen =====
111 # =====
112
113 # ***** Alles erlauben und alle Regeln loeschen
114 echo " -- do: Policy and flush"
115 # Default policy setzen (Alles erlauben)
116 $IPTABLES -P INPUT ACCEPT
117 $IPTABLES -P FORWARD ACCEPT # Bei 2 Interfaces (Router)
118 $IPTABLES -P OUTPUT ACCEPT
119
120 # Loesche alle Filterregeln
121 $IPTABLES -F # flush aller chains (Tabelle filter )
122 $IPTABLES -t nat -F # flush aller chains (Tabelle nat)
123 $IPTABLES -X # delete all userdefined chains (Tabelle filter )
124
125 # ***** ENDE ***** NAT und Port-Forwarding *****
126 echo " -- done: Policy and flush"
127
128
129 # =====
130 # === Part 3: NAT und Port-Forwarding implementieren ===
131 # =====
132
133 # ***** NAT und Port-Forwarding aktivieren
134 echo " -- do: NAT und Port-Forwarding"
135 # Hier die Zeilen schreiben, die
136 # a) NAT auf dem Outside-Router implementiert und
137 $IPTABLES -A FORWARD -o $iEXT -s $DMZ -m conntrack --ctstate NEW -j ACCEPT
138 $IPTABLES -A FORWARD -o $iEXT -s $LAN -m conntrack --ctstate NEW -j ACCEPT
139 $IPTABLES -t nat -A POSTROUTING -o $iEXT -j MASQUERADE
```

```
140
141 # b) das Port-Forwarding von ausserhalb zu dem Webserver aktivieren
142 $IPTABLES -A PREROUTING -t nat -i $iEXT -p tcp --dport 80 -j DNAT --to-destination $Webserver:80
143 $IPTABLES -A FORWARD -p TCP -d $Webserver --dport 80 -j ACCEPT
144 $IPTABLES -A POSTROUTING -t nat -s $Webserver -o $iEXT -j MASQUERADE
145
146 # ***** ENDE ***** NAT und Port-Forwarding aktivieren
147 echo " - done: NAT und Port-Forwarding"
148
149
150 # =====
151 # === Ausgabe =====
152 # =====
153
154 # ***** Konfiguration in DATEI umleiten
155 echo " - do: Schreibe Konfiguration in $loPath"
156 echo -e "\n\n===== " >> $loPath
157 date >> $loPath
158 echo "Firewall gestoppt" >> $loPath
159 echo -e "=====\n" >> $loPath
160 $IPTABLES -L -v -n >> $loPath
161
162 # ***** ENDE ***** Konfiguration in DATEI umleiten
163 echo " - done: Schreibe Konfiguration in $loPath"
164
165 # =====
166 #*****ENDE***** Firewall stoppen *****
167 # =====
168 ;;
169
170
171 # =====
172 # =====
173 # === Firewall starten =====
174 # =====
175 # =====
176 start )
177
178 # =====
179 # === Part 2: Default Policy setzen =====
180 # =====
181
182 # ***** Alles verbieten und alle Regeln loeschen
183 echo " - do: Policy and flush"
184
185 # Default Policy: Alles verbieten
186 $IPTABLES -P INPUT DROP
187 $IPTABLES -P FORWARD DROP # Bei 2 Interfaces (Router)
188 $IPTABLES -P OUTPUT DROP
189
190 # Loesche alte Filterregeln
```

```
191 # chain (engl. Kette, Folge, Befehlsfolge)
192 $IPTABLES -F # flush aller chains (Tabelle filter )
193 $IPTABLES -t nat -F # flush aller chains (Tabelle nat)
194 $IPTABLES -X # delete all userdefined chains (Tabelle filter )
195
196 # ***** ENDE ***** Alles verbieten und alle Regeln loeschen
197 echo " - done: Policy and flush"
198
199
200 # =====
201 # === Part 3: NAT und Port-Forwarding implementieren ===
202 # =====
203
204 # ***** Loopback erlauben *****
205 echo " - do: NAT und Port-Forwarding"
206 # Hier die Zeilen schreiben, die
207 # a) NAT auf dem Outside-Router implementiert und
208 $IPTABLES -A FORWARD -o $iEXT -s $DMZ -m conntrack --ctstate NEW -j ACCEPT
209 $IPTABLES -A FORWARD -o $iEXT -s $LAN -m conntrack --ctstate NEW -j ACCEPT
210 $IPTABLES -t nat -A POSTROUTING -o $iEXT -j MASQUERADE
211
212 # b) das Port-Forwarding von ausserhalb zu dem Webserver aktivieren
213 $IPTABLES -A PREROUTING -t nat -i $iEXT -p tcp --dport 80 -j DNAT --to-destination $Webserver:80
214 $IPTABLES -A FORWARD -p TCP -d $Webserver --dport 80 -j ACCEPT
215 $IPTABLES -A POSTROUTING -t nat -s $Webserver -o $iEXT -j MASQUERADE
216
217 # *****ENDE ***** Alles verbieten und alle Regeln loeschen
218 echo " - done: NAT und Port-Forwarding"
219
220
221 # =====
222 # === Part 4: Aufgabenstellung umsetzen =====
223 # =====
224
225 # ***** Loopback erlauben *****
226 echo " - do: Loopback erlauben"
227 $IPTABLES -A INPUT -i lo -j ACCEPT
228 $IPTABLES -A OUTPUT -o lo -j ACCEPT
229
230 #***** ENDE ***** Loopback erlauben *****
231 echo " - done: Loopback erlauben"
232
233
234 # ***** ssh-Zugriff vom AdminPC auf Router sicherstellen
235 echo " - do: SSH-Zugang fuer AdminPC"
236 # fuer den Outside-Router
237 $IPTABLES -A INPUT -p TCP -s $AdminPC --dport ssh -j ACCEPT
238 $IPTABLES -A OUTPUT -p TCP -d $AdminPC --sport ssh -j ACCEPT
239
240 # fuer den Inside-Router
241 $IPTABLES -A FORWARD -p TCP -s $AdminPC -d $LinuxInside_dmz --dport ssh -j ACCEPT
```

B Testdokumentation

```
242 $IPTABLES -A FORWARD -p TCP -s $LinuxInside_dmz -d $AdminPC --sport ssh -j ACCEPT
243 $IPTABLES -A FORWARD -p TCP -s $AdminPC -d $LinuxInside_in --dport ssh -j ACCEPT
244 $IPTABLES -A FORWARD -p TCP -s $LinuxInside_in -d $AdminPC --sport ssh -j ACCEPT
245
246 # ***** ENDE ***** ssh-Zugriff vom AdminPC auf Router sicherstellen
247 echo " - done: SSH-Zugang fuer AdminPC"
248
249
250 # ***** Verbindung zu einem Zeitserver erlauben
251 echo " - do: NTP Ports oeffnen"
252 # fuer diesen (den Outside-) Router erlauben
253 $IPTABLES -A INPUT -p udp -s $TimeSrv --sport $ntpPort -j ACCEPT
254 $IPTABLES -A OUTPUT -p udp -d $TimeSrv --dport $ntpPort -j ACCEPT
255
256 # fuer DMZ-Netz erlauben
257 $IPTABLES -A FORWARD -p udp -s $DMZ -d $TimeSrv --dport $ntpPort -j ACCEPT
258 $IPTABLES -A FORWARD -p udp -d $DMZ -s $TimeSrv --sport $ntpPort -j ACCEPT
259
260 # fuer LAN-Netz erlauben
261 $IPTABLES -A FORWARD -p udp -s $LAN -d $TimeSrv --dport $ntpPort -j ACCEPT
262 $IPTABLES -A FORWARD -p udp -d $LAN -s $TimeSrv --sport $ntpPort -j ACCEPT
263
264 # ***** ENDE ***** Konfiguration fuer Zeitsynchronisation
265 echo " - done: NTP Ports oeffnen"
266
267
268 echo " - do: Zugang fuer Webserver"
269 # ***** Verbindung zum Webserver zulassen *
270 $IPTABLES -A FORWARD -p TCP -s $Webserver --sport 80 -j ACCEPT
271 $IPTABLES -A FORWARD -p TCP -d $Webserver --dport 80 -j ACCEPT
272
273 # ***** ENDE ***** Konfiguration fuer Zugriff auf Webserver
274 echo " - done: Zugang fuer Webserver"
275
276
277 # ***** ICMP Erlauben *****
278 echo " - do: Ping erlauben"
279 # ICMP-ECHO Request und ICMP-ECHO Reply fuer den Outside-Router durch AdminPC erlauben
280 $IPTABLES -A INPUT -p icmp --icmp-type 8 -s $AdminPC -j ACCEPT
281 $IPTABLES -A OUTPUT -p icmp --icmp-type 0 -d $AdminPC -j ACCEPT
282
283 # ICMP-ECHO Request und ICMP-ECHO Reply fuer das DMZ-Netz erlauben
284 $IPTABLES -A FORWARD -p icmp --icmp-type 8 -s $DMZ -j ACCEPT
285 $IPTABLES -A FORWARD -p icmp --icmp-type 0 -d $DMZ -j ACCEPT
286
287 # ICMP-ECHO Request und ICMP-ECHO Reply fuer das LAN-Netz
288 $IPTABLES -A FORWARD -p icmp --icmp-type 8 -s $LAN -j ACCEPT
289 $IPTABLES -A FORWARD -p icmp --icmp-type 0 -d $LAN -j ACCEPT
290
291 # ***** ENDE ***** Konfiguration ICMP *****
292 echo " - done: Ping erlauben"
```

```
293
294
295 # ***** Konfiguration DNS HTTP HTTPS *****
296 echo " -- do: DNS erlauben"
297 ## DNS durchlassen fuer DMZ und LAN
298 for port in ${dnsPorts[@]}
299 do
300     for protocol in ${protocols[@]}
301     do
302         $IPTABLES -A FORWARD -p "$protocol" -s $DMZ --dport "$port" -j ACCEPT
303         $IPTABLES -A FORWARD -p "$protocol" -d $DMZ --sport "$port" -j ACCEPT
304         $IPTABLES -A FORWARD -p "$protocol" -s $LAN --dport "$port" -j ACCEPT
305         $IPTABLES -A FORWARD -p "$protocol" -d $LAN --sport "$port" -j ACCEPT
306     done
307 done
308
309 ## Bestimmte DNS-Server fuer Router
310 for port in ${dnsPorts[@]}
311 do
312     for protocol in ${protocols[@]}
313     do
314         for dnsSrv in ${DNS[@]}
315         do
316             $IPTABLES -A INPUT -p "$protocol" -s "$dnsSrv" -d $LinuxOutside_out --sport "$port" -j ACCEPT
317             $IPTABLES -A OUTPUT -p "$protocol" -s $LinuxOutside_out -d "$dnsSrv" --dport "$port" -j ACCEPT
318         done
319     done
320 done
321
322 # ***** ENDE ***** Konfiguration DNS *****
323 echo " -- done: DNS erlauben"
324
325
326 # ***** Konfiguration HTTP HTTPS *****
327 echo " -- do: HTTP/S erlauben"
328 ## HTTP/S fuer LAN und DMZ erlauben
329 for port in ${webPorts[@]}
330 do
331     $IPTABLES -A FORWARD -p TCP -s $DMZ --dport "$port" -j ACCEPT
332     $IPTABLES -A FORWARD -p TCP -d $DMZ --sport "$port" -j ACCEPT
333     $IPTABLES -A FORWARD -p TCP -s $LAN --dport "$port" -j ACCEPT
334     $IPTABLES -A FORWARD -p TCP -d $LAN --sport "$port" -j ACCEPT
335 done
336
337 # ***** ENDE ***** Konfiguration DNS HTTP/S *****
338 echo " -- done: HTTP/S erlauben"
339
340
341 # ***** Konfiguration RDP *****
342 echo " -- do: RDP erlauben"
343 # RDP Zugang fuer DMZ-Server
```

B Testdokumentation

```

344 for protocol in ${protocols[@]}
345 do
346     $IPTABLES -A FORWARD -p "$protocol" -s $Webserver -d $AdminPC --sport $rdpPort -j ACCEPT
347     $IPTABLES -A FORWARD -p "$protocol" -s $AdminPC -d $Webserver --dport $rdpPort -j ACCEPT
348 done
349
350 # ***** ENDE ***** Konfiguration RDP *****
351 echo " - done: RDP erlauben"
352
353
354 # =====
355 # === Ausgabe ===
356 # =====
357
358 # ***** Konfiguration in DATEI umleiten ***
359 echo " - do: Schreibe Konfiguration in $loPath"
360 echo -e "\n\n===== " >> $loPath
361 date >> $loPath
362 echo "Firewall gestartet" >> $loPath
363 echo -e "===== \n" >> $loPath
364 $IPTABLES -L -v -n >> $loPath
365
366 # ***** ENDE ***** Konfiguration in DATEI umleiten
367 echo " - done: Schreibe Konfiguration in $loPath"
368
369 # =====
370 # ***** ENDE ***** Firewall starten *****
371 # =====
372 ;;
373
374
375 # =====
376 # === Firewall Parameter anzeigen ===
377 # =====
378 *)
379
380 # ***** Anzeige Fehlermeldung und Hilfe
381 echo "Falscher oder kein Parameter uebergeben!"
382 echo "stop - Stoppt die Firewall."
383 echo "start - Startet die Firewall."
384
385 # ***** ENDE ***** Anzeige Fehlermeldung und Hilfe
386
387 # =====
388 # ***** ENDE ***** Eingabeoptionen anzeigen *****
389 # =====
390 ;;
391
392
393 esac

```



## B.2.2 firewall.sh (auf dem Inside-Router)

```
1 #!/bin/bash
2 # Bourne- Again Shell#
3
4 # =====
5 # === Bemerkung =====
6 # =====
7
8 # Sekundaere Firewall
9 # ...verhindert unbefugten Zugriff vom lokalem Netz auf lokales Interface des Routers.
10 # Wenn die Firewall gestoppt ist, wird auch NAT gestoppt. Dies sorgt dafuer,
11 # dass auch alle internen Anfragen an das DMZ-Netz ueber den Outside-Router laufen.
12 # Wenn die Firewall startet laeuft der Verkehr zwischen LAN und DMZ nur ueber den Inside-Router.
13 # Da NAT die IP des Admin-PCs uebersetzt, greifen die Zugriffsberechtigungen
14 # (ICMP, SSH) auf dem Outside-Router nicht. Daher wird er vom NAT ausgeschlossen.
15 #
16 #-----Ende--Bemerkung-----
17
18
19 # =====
20 # === Part 1: Variablen =====
21 # =====
22 echo " - Variablen werden gesetzt"
23
24 # Pfad zu iptables
25 IPTABLES=/sbin/iptables
26
27 # Macht Linux-Maschine zu einem Router
28 echo "1" > /proc/sys/net/ipv4/ip_forward
29
30 # Interfaces
31 iINT=eth0
32 iEXT=eth1
33
34 # Definition DNS
35 DNS=("192.168.95.40/32 192.168.95.41/32")
36
37 # Timeserver: hier Standardgateway
38 TimeSrv=192.168.200.1
39
40 # Der Rechner, auf dem die Firewall (Inside) laufen soll , hier die VMWare
41 LinuxInside_in=10.0.9.1
42 LinuxInside_dmz=172.16.9.2
43
44 # Der Rechner, auf dem die Firewall (Outside) laufen soll , hier die VMWare
45 LinuxOutside_out=192.168.200.109
46 LinuxOutside_dmz=172.16.9.1
47
48 # Rechner fuer Fernwartung z.B. mit ssh, hier der Windowswirt (XP, Win7 o.ae.)
49 AdminPC=10.0.9.2
```

```
50
51 # Webserver
52 Webserver=172.16.9.3
53
54 # Das DMZ-Netz
55 DMZ=172.16.9.0/24
56
57 # Das LAN-Netz
58 LAN=10.0.9.0/24
59
60 # Protokolle
61 protocols=("tcp" "udp")
62
63 # DNS Ports
64 dnsPorts=("53" "853")
65
66 # HTTP/S Port
67 webPorts=("80" "443")
68
69 # ntp Port
70 ntpPort=123
71
72 # rdp Port
73 rdpPort=3389
74
75 # Pfad zur aktuellen Firewall Konfiguration
76 lopPath="/var/log/firewall/ firewallconfig "
77
78 # ***** ENDE ***** Variablen setzen *****
79
80
81 # =====
82 # =====
83 # === Starten / Stoppen / Hilfe ===
84 # =====
85 # =====
86 case "$1" in
87
88 # =====
89 # =====
90 # === Firewall stoppen ===
91 # =====
92 # =====
93 stop)
94
95 # =====
96 # === Part 2: Default Policy setzen ===
97 # =====
98
99 # ***** Alles erlauben und alle Regeln loeschen
100 echo " -- do: Policy and flush "
```

B Testdokumentation

```

101 # Default policy setzen (Alles erlauben)
102 $IPTABLES -P INPUT ACCEPT
103 $IPTABLES -P FORWARD ACCEPT
104 $IPTABLES -P OUTPUT ACCEPT
105
106 # Loesche alle Filterregeln
107 $IPTABLES -F # flush aller chains (Tabelle filter )
108 $IPTABLES -t nat -F # flush aller chains (Tabelle nat)
109 $IPTABLES -X # delete all userdefined chains (Tabelle filter )
110
111 # ***** ENDE ***** NAT und Port-Forwarding
112 echo " - done: Policy and flush"
113
114
115 # =====
116 # === Ausgabe =====
117 # =====
118
119 # ***** Konfiguration in Datei umleiten
120 echo " - do: Schreibe Konfiguration in $loPath"
121 echo -e "\n\n===== " >> $loPath
122 date >> $loPath
123 echo "Firewall gestoppt" >> $loPath
124 echo -e "===== \n" >> $loPath
125 $IPTABLES -L -v -n >> $loPath
126
127 # ***** Konfiguration in Datei umleiten
128 echo " - done: Schreibe Konfiguration in $loPath"
129
130 # =====
131 # ***** Firewall stoppen *****
132 # =====
133 ;;
134
135
136 # =====
137 # =====
138 # === Firewall starten =====
139 # =====
140 # =====
141 start )
142
143 # =====
144 # === Default Policy setzen und NAT =====
145 # =====
146
147 # ***** Alles verbieten und alle Regeln loeschen
148 echo " - do: Policy and flush"
149
150 # Default Policy: Alles verbieten
151 $IPTABLES -P INPUT DROP

```

B Testdokumentation

```
152 $IPTABLES -P FORWARD DROP
153 $IPTABLES -P OUTPUT DROP
154
155 # Loesche alte Filterregeln
156 # chain (engl. Kette, Folge, Befehlsfolge)
157 $IPTABLES -F # flush aller chains (Tabelle filter )
158 $IPTABLES -t nat -F # flush aller chains (Tabelle nat)
159 $IPTABLES -X # delete all userdefined chains (Tabelle filter )
160
161 # ***** ENDE ***** Alles verbieten und alle Regeln loeschen
162 echo " - done: Policy and flush"
163
164
165 # ***** NAT aktivieren *****
166 echo " - do: NAT"
167 # NAT auf dem Inside-Router implementieren, Admin-PC ausschliessen
168 $IPTABLES -A FORWARD -o $iEXT -s $LAN -m conntrack --ctstate NEW -j ACCEPT
169 $IPTABLES -t nat -A POSTROUTING -m iprange --src-range 10.0.9.3-10.0.9.254 -o $iEXT -j
    MASQUERADE
170
171 # ***** ENDE ***** NAT aktivieren *****
172 echo " - done: NAT"
173
174
175 # =====
176 # === LO, NTP, ICMP, SSH, DNS, HTTPS, RDP =====
177 # =====
178
179 # ***** Loopback erlauben *****
180 echo " - do: Loopback erlauben"
181 $IPTABLES -A INPUT -i lo -j ACCEPT
182 $IPTABLES -A OUTPUT -o lo -j ACCEPT
183
184 # ***** ENDE ***** Loopback erlauben *****
185 echo " - done: Loopback erlauben"
186
187
188 # ***** Verbindung zu einem Zeitserver erlauben
189 echo " - do: NTP Ports oeffnen"
190 # fuer diesen (den Inside-) Router erlauben
191 $IPTABLES -A INPUT -p udp -s $TimeSrv --sport $ntpPort -j ACCEPT
192 $IPTABLES -A OUTPUT -p udp -d $TimeSrv --dport $ntpPort -j ACCEPT
193
194 # fuer LAN-Netz erlauben
195 $IPTABLES -A FORWARD -p udp -s $LAN -d $TimeSrv --dport $ntpPort -j ACCEPT
196 $IPTABLES -A FORWARD -p udp -d $LAN -s $TimeSrv --sport $ntpPort -j ACCEPT
197
198 # ***** ENDE ***** Konfiguration fuer Zeitsynchronisation
199 echo " - done: NTP Ports oeffnen"
200
201
```

```
202 # ***** ssh-Zugriff vom AdminPC auf Router sicherstellen
203 echo " -- do: SSH-Zugang fuer AdminPC"
204 # fuer den Inside-Router
205 $IPTABLES -A INPUT -p TCP -s $AdminPC --dport ssh -j ACCEPT
206 $IPTABLES -A OUTPUT -p TCP -d $AdminPC --sport ssh -j ACCEPT
207
208 # fuer den Outside-Router
209 $IPTABLES -A FORWARD -s $AdminPC -d $LinuxOutside_dmz -p TCP --dport ssh -j ACCEPT
210 $IPTABLES -A FORWARD -s $LinuxOutside_dmz -d $AdminPC -p TCP --sport ssh -j ACCEPT
211 $IPTABLES -A FORWARD -s $AdminPC -d $LinuxOutside_out -p TCP --dport ssh -j ACCEPT
212 $IPTABLES -A FORWARD -s $LinuxOutside_out -d $AdminPC -p TCP --sport ssh -j ACCEPT
213
214 # ***** ENDE ***** ssh-Zugriff vom AdminPC auf Router sicherstellen
215 echo " -- done: SSH-Zugang fuer AdminPC"
216
217
218 # ***** ICMP Erlauben *****
219 echo " -- do: Ping erlauben"
220 # ICMP-ECHO Request und ICMP-ECHO Reply fuer den Inside-Router durch AdminPC erlauben
221 $IPTABLES -A INPUT -p icmp --icmp-type 8 -s $AdminPC -j ACCEPT
222 $IPTABLES -A OUTPUT -p icmp --icmp-type 0 -d $AdminPC -j ACCEPT
223
224 # ICMP-ECHO Request und ICMP-ECHO Reply fuer das DMZ-Netz erlauben
225 $IPTABLES -A FORWARD -p icmp --icmp-type 8 -s $DMZ -j ACCEPT
226 $IPTABLES -A FORWARD -p icmp --icmp-type 0 -d $DMZ -j ACCEPT
227
228 # ICMP-ECHO Request und ICMP-ECHO Reply fuer das LAN-Netz
229 $IPTABLES -A FORWARD -p icmp --icmp-type 8 -s $LAN -j ACCEPT
230 $IPTABLES -A FORWARD -p icmp --icmp-type 0 -d $LAN -j ACCEPT
231
232 # ***** ENDE ***** Konfiguration ICMP *****
233 echo " -- done: Ping erlauben"
234
235
236 echo " -- do: Zugang fuer Webserver"
237 # ***** Verbindung zum Webserver zulassen
238 $IPTABLES -A FORWARD -p TCP -s $Webserver --sport 80 -j ACCEPT
239 $IPTABLES -A FORWARD -p TCP -d $Webserver --dport 80 -j ACCEPT
240
241 # ***** ENDE ***** Konfiguration fuer Zugriff auf Webserver
242 echo " -- done: Zugang fuer Webserver"
243
244
245 # ***** Konfiguration DNS HTTP HTTPS *****
246 echo " -- do: DNS erlauben"
247 ## DNS durchlassen fuer LAN
248 for port in ${dnsPorts[@]}
249 do
250     for protocol in ${protocols[@]}
251     do
252         $IPTABLES -A FORWARD -p "$protocol" -s $LAN --dport "$port" -j ACCEPT
```

B Testdokumentation

```

253     $IPTABLES -A FORWARD -p "$protocol" -d $LAN --sport "$sport" -j ACCEPT
254 done
255 done
256
257 ## Bestimmte DNS-Server fuer Router
258 for port in ${dnsPorts[@]}
259 do
260     for protocol in ${protocols[@]}
261     do
262         for dnsSrv in ${DNS[@]}
263         do
264             $IPTABLES -A INPUT -p "$protocol" -s "$dnsSrv" -d $LinuxInside_dmz --sport "$sport" -j ACCEPT
265             $IPTABLES -A OUTPUT -p "$protocol" -s $LinuxInside_dmz -d "$dnsSrv" --dport "$sport" -j ACCEPT
266         done
267     done
268 done
269
270 # *****ENDE***** Konfiguration DNS *****
271 echo " - done: DNS erlauben"
272
273
274 # ***** Konfiguration HTTP HTTPS *****
275 echo " - do: HTTP/S erlauben"
276 ## HTTP/S fuer LAN und DMZ erlauben
277 for port in ${webPorts[@]}
278 do
279     $IPTABLES -A FORWARD -p TCP -s $LAN --dport "$port" -j ACCEPT
280     $IPTABLES -A FORWARD -p TCP -d $LAN --sport "$port" -j ACCEPT
281 done
282
283 # ***** ENDE ***** Konfiguration DNS HTTP/S *****
284 echo " - done: HTTP/S erlauben"
285
286
287 # ***** Konfiguration RDP *****
288 echo " - do: RDP erlauben"
289 # RDP Zugang fuer DMZ-Server
290 for protocol in ${protocols[@]}
291 do
292     $IPTABLES -A FORWARD -p "$protocol" -s $AdminPC -d $Webserver --dport $rdpPort -j ACCEPT
293     $IPTABLES -A FORWARD -p "$protocol" -s $Webserver -d $AdminPC --sport $rdpPort -j ACCEPT
294 done
295
296 # ***** ENDE ***** Konfiguration RDP *****
297 echo " - done: RDP erlauben"
298
299
300 # =====
301 # === Ausgabe =====
302 # =====
303

```

B Testdokumentation

```
304 # ***** Konfiguration in DATEI umleiten
305 echo " - do: Schreibe Konfiguration in $loPath"
306 echo -e "\n\n===== " >> $loPath
307 date >> $loPath
308 echo "Firewall gestartet " >> $loPath
309 echo -e "===== \n" >> $loPath
310 $IPTABLES -L -v -n >> $loPath
311
312 # ***** ENDE ***** Konfiguration in DATEI umleiten
313 echo " - done: Schreibe Konfiguration in $loPath"
314
315 # =====
316 # ***** ENDE ***** Firewall starten *****
317 # =====
318 ;;
319
320
321 # =====
322 # === Eingabeoptionen anzeigen =====
323 # =====
324 *)
325
326 # ***** Anzeige Fehlermeldung und Hilfe ***
327 echo "Falscher oder kein Parameter uebergeben!"
328 echo "stop - Stoppt die Firewall."
329 echo "start - Startet die Firewall."
330
331 # ***** ENDE ***** Anzeige Fehlermeldung und Hilfe
332
333 # =====
334 # ***** ENDE ***** Eingabeoptionen anzeigen *****
335 # =====
336 ;;
337
338
339 esac
```