UNIVERSITY OF APPLIED SCIENCES
FHDW
FACHHOCHSCHULE FÜR DIE WIRTSCHAFT
HANNOVER

Installation und Einrichtung eines Firewallsystems unter openSUSE mit iptables für das Projekt Schul-IT (Projekt ∑=0) an der FHDW Hannover

Betreuer: Prof. Dr. Günther Hellberg
Datum 06.07.2009
Version 0.1
Autoren: Patrick Scheel
Oliver Enns

Inhaltsverzeichnis

1. Motivation

Basis für dieses Thema bildet das Gesamtprojekt „Schul-IT“. Das Projekt wurde ins Leben gerufen, um eine einfache und hocheffektive IT-Infrastruktur für Schulen aller Art - wie Grundschule, weiterführende Schule etc. - zu schaffen. Es soll mit dem Projekt „Schul-IT“ ein offener quasi Standard für Schulnetzwerke geschaffen werden, um nicht seitens einer Schule auf proprietäre oder kommerzielle Lösungen zurückgreifen zu müssen, was letztendlich auch die Gefahr birgt, von einem Hersteller oder Dienstleister abhängig zu sein. Die Grundidee des Projekts ist, dass mit kostengünstigen bzw. kostenfreien Ansätzen gearbeitet wird, sodass die finanziellen Mittel der Steuerzahler genau da deren Verwendung finden, wofür sie nämlich auch vorgesehen sind: und zwar in der Schulbildung.
Das Ziel des Gesamtprojekts ist die Entwicklung eines einheitlichen Konzepts, dass anschließend an jeder Schule eingesetzt werden kann. Aufgrund des von vornherein homogenen Projektentwurfs, sollte die interne Struktur einer Schule dabei keine Rolle mehr spielen. Aus der Sicht des Gesamtprojekts wird ein hoher Wert auf Modularität, Transparenz, Flexibilität, Sicherheit, Erweiterbarkeit und Redundanz gelegt.
Primäres Ziel des Projektes soll es also sein für Schüler, Lehrer und Administratoren eine transparente IT-Infrastruktur bereitzustellen, die einfache Verwaltung garantieren und den Lernerfolg der Schüler steigern. Hierbei soll das Konzept mit Hilfe der Schüler und Lehrkräfte erarbeitet und laufend verbessert werden.
Dieses Projekt ist somit ein Teilprojekt und beschäftigt sich mit der Installation von Firewallsystemen, die den Schutz des Schulnetzwerkes sicherstellen sollen.

2. Projektziel

Das Ziel des Teilprojektes „Firewall“ ist die Installation und Einrichtung von zwei Firewallsystemen, welche das Schulnetzwerk schützen sollen.
Für das Einrichten der Firewallsysteme müssen einige Vorbereitungen getroffen werden. Hierzu gehört das Festlegen der IP-Adressierung, ebenso wie die Festlegung der Ports für Dienste, die von den Serversystemen in der DMZ (Demilitarisierte Zone) bereitgestellt werden.
Die Firewallsysteme werden in zwei unterschiedlichen Netzwerksegmenten eingesetzt. Aufgrund ihrer Anordnung im Schulnetzwerk, wird die Firewall, die das Schulnetzwerk vom Internet trennt als Firewall „Außen“ bezeichnet. Die im internen Netzwerk eingesetzte Firewall „Innen“ separiert das Clientnetzwerk vom Netzwerksegment für Server (DMZ).
Ein weiterer Bestandteil, um die Kommunikation zwischen den einzelnen Netzwerksegmenten und dem Internet zu ermöglichen, ist das Festlegen und Konfigurieren der Routingregeln für die Firewallsysteme.
Eine der größten Herausforderungen für das Projekt Firewall ist es ein geeignetes Regelkonzept für beide Firewallsysteme zu finden. Dabei soll auf die Kommunikationspfade der Netzwerkkomponenten wie Clients und Server und den angeforderten Diensten (wie z.B. den http-Verkehr) Rücksicht genommen werden. Dies bedeutet, dass freigegebener Netzwerkverkehr die Firewalls passieren darf, während unerwünschter Datenverkehr und Angriffe auf das Schulnetzwerks an den Firewalls abgewiesen werden.

Nachdem ein geeignetes Regelkonzept gefunden wurde, bedarf es einer Integration dieses Konzeptes auf die Firewallsysteme in Form von Firewallregeln. Diese werden für jede Firewall separat definiert und auf den entsprechenden Systemen installiert.
Die folgende Tabelle führt die für das Regelkonzept notwendigen Punkte auf:

Ziel	Beschreibung	FW-A	FW-I
Regeldefinition für die DMZ-Dienste	Aus dem Schul-Netz sollen die DMZ-Dienste erreichbar sein.		X
Webserveranfragen	Es soll von außen der interne Webserver erreichbar sein. Die Anfragen von außen müssten an den entsprechenden Server weitergeleitet werden.	X	
Mailverkehr	Es soll von außen der interne Mailserver erreichbar sein. Zu einem um Mails von anderen Servern zu empfangen oder das Abholen der Mails durch die Clients zu ermöglichen. Die Anfragen von außen müssten an den entsprechenden Server weitergeleitet werden. Ebenso sollte auch aus dem Schul-Netz der Mailverkehr möglich sein.	X	X
Logging	Für bestimmte Pakete soll eine Protokollierung vorgenommen werden. Diese Regeln sind jeweils auf beiden Systemen eingerichtet.	X	X
DHCP-Anfragen	Da der DHCP-Server im DMZ untergebracht ist, soll ermöglicht werden, dass die Clients aus dem Schul-Netz IP-Adressen anfordern bzw. beziehen können.		X
Standard-Regel	Die Firewall hat eine Standard-Regel, die dann greift, wenn keine passende Regel gefunden werden kann. Folgende Standard-Regel für die Firewall ist festgelegt: „Es ist alles verboten, was nicht explizit zugelassen ist!“	X	X
HTTP-Verkehr über Proxy leiten	Der Internet-Verkehr ist nur über den Proxy möglich.	X	X
Bekannte Angriffsmuster verhindern	Es sollen Regeln für bekannte Angriffsmuster (IP-Spoofing) implementiert werden.	X	X
TCP-Flag-Check	Pakete mit ungültiger TCP-Flag Kombination sollen erkannt und geblockt werden.	X	X
Fernadministration ermöglichen	Für Administrationszwecke soll eine Regel eingerichtet werden, die ermöglicht, dass über das SSH-Protokoll eine Verbindung zu dem Wirtsystem aufgebaut werden kann.	X	X
Diagnoseprotokoll (ICMP)	Für Diagnosezwecke soll das Protokoll ICMP erlaubt werden.	X	X

Tabelle 1: Regelziel-Tabelle

Damit die korrekte Funktionsweise der Firewalls und der definierten Firewall-Regeln gewährleistet wird, muss sichergestellt werden, dass diese Regeln nach einem Neustart des Systems wieder wirksam werden. Hier wird das Ziel verfolgt, mit Hilfe geeigneter Start-und Stopskripte, die Firewall-Regeln auf den Systemen aktivieren und ggf. auch wieder deaktivieren zu können.
Alle in diesem Kapitel beschriebenen „Teilziele“ sollen dann zusammengeführt werden, um das Konzept zum Schutz eines Client-Netzwerkes zu realisieren. Dieses Projekt beschäftigt sich

vorerst mit einem Client-Netzwerk, welches später als Basis für die weiteren Client-Netzwerke dienen soll. Das projektbezogene Client-Netz hat den Adressbereich 192.168.10.0/24.

3. Projekt-Betriebsmittel

Für die Realisierung der oben genannten Projektziele müssen geeignete Betriebsmittel zur Verfügung stehen. Die Firewallsysteme, sowie alle weiteren Server die Netzwerkdienste zur Verfügung stellen, werden auf virtuellen Maschinen installiert.
Da bei diesem Projekt besonderen Wert auf die Minimierung der Kosten gelegt wird, werden Software-Produkte eingesetzt, die keine Lizenzgebühren erfordern.
Die Betriebsinfrastruktur der virtuellen Maschinen, stellt daher der kostenfreie VMWare Server in der Version 1.0.8 des Virtualisierungsspezialisten VMware [1] zur Verfügung.
Das Basisbetriebssystem für die Firewallsysteme ist openSuse 11.0 [2]. Dadurch das nie mehr als 4GB Arbeitsspeicher pro System benötigt werden, wird die 32Bit Version von openSuse installiert. Eine Installationsanleitung des Betriebssystems openSuse 11.0 auf eine virtuelle Maschine liegt dem Anhang bei.
Die Grundlage der Firewall bildet netfilter/iptables [3] in der Version 1.4.0. Iptables wird in Kapitel 8 detaillierter beschrieben.

4. Entwurf Netzwerkarchitektur

In diesem Kapitel wird der Entwurf der Netzwerkarchitektur beschrieben. Dabei wird ebenso auf die Netzwerkumgebung eingegangen, als auch auf die im Netzwerk zur Verfügung gestellten Dienste.

4.1. Netzwerkumgebungen

Der Einsatzort für die die Firewallsysteme „Innen“ und „Außen“ sind Schulnetzwerke.
Die Firewallsysteme werden eingesetzt, um

1.) die DMZ bzw. das Servernetzwerk vom Internet (Firewall „Außen“) und
2.) die Klassennetzwerke von der demilitarisierten Zone zu separieren (Firewall „Innen“)

Hierbei werden im Wesentlichen drei Netzbereiche unterschieden. Der erste Bereich ist ein Netz zwischen dem eigentlichen DSL-Device und der Firewall. Das DSL-Device stellt den Internetzugang bereit. Über ein privates Netzwerk ist das DSL-Device mit der Firewall verbunden. Der zweite Bereich stellt die demilitarisierte Zone (DMZ) dar. In dieser Zone werden Serversysteme installiert, die verschiedene Dienste anbieten wie z.B. Proxy, Webserver, Emailserver usw.
In der dritten Zone befinden sich die Klassennetzwerke. Diese erhalten unterschiedliche Adressbereiche und sind von der DMZ durch eine weitere Firewall getrennt. Nachfolgende Grafik veranschaulicht noch einmal die Netzwerkarchitektur:

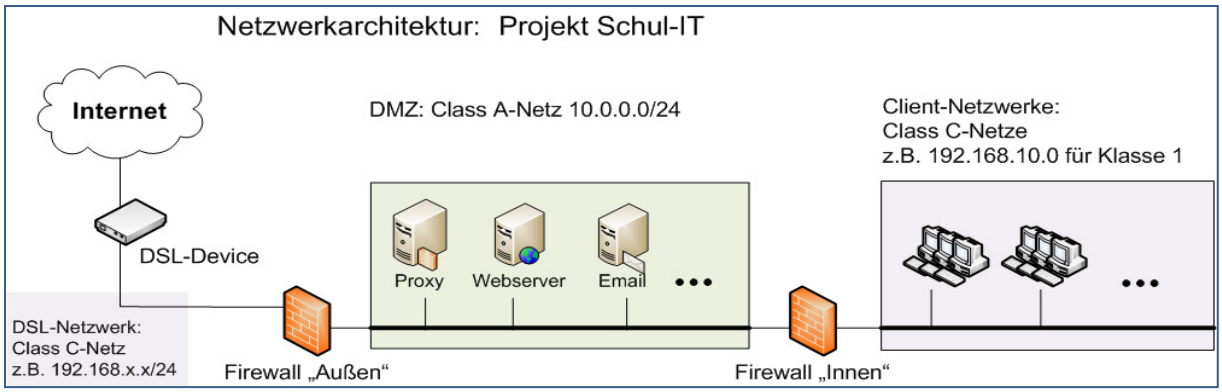


Abbildung 1: Projekt-Netzwerkarchitektur

Der Aufbau des Netzwerkbereiches vor der Firewall „Außen“ kann variieren. Diese Konfiguration muss nach lokalen Gegebenheiten eingerichtet werden.
Das Servernetzwerk beschreibt ein Class-A-Netzwerk mit einem verfügbaren IP-Adressbereich von 10.0.0.1 – 10.0.0.254. Dabei gilt folgende Aufteilung und Festlegung der Adressbereiche:

- Der Adressbereich von 10.0.0.1 – 10.0.0.99 ist für zusätzliche Netzwerkkomponenten wie Switche, Router usw. reserviert.
- Der Adressbereich von 10.0.0.100 – 10.0.0.254 ist für die Server und Dienste reserviert.

Das Test-Clientnetzwerk umfasst 254 IP-Adressen aus einem Class-C-Netz mit dem IP-Adressbereich 192.168.10.0/24, wie bereits im Kapitel *Projektziel* erwähnt wurde.

4.2. Dienste

Für die Erstellung der Firewall- und Routingregeln ist es notwendig, dass die IP-Adressen und Portnummern für die Server und Dienste schon im Entwurf der Netzwerkarchitektur festgelegt werden. Die nachfolgende Tabelle zeigt die Dienste und deren zugehörige Ports und Übertragungsprotokolle, welche innerhalb des Schulnetzes benötigt werden.

Service	Port	Protokoll	
		TCP	UDP
DNS	53	X	X
DHCP (BOOTP)	67/68	X	X
SMB	445	X	-
LDAP	389	X	-
SMTP	25	X	-
IMAP	143	X	-
POP3	110	X	-
http	80	X	-
HTTPS	443	X	-
FTP	20/21	X	-
OPSI	4447	X	X
NOMACHINE	5000	X	-
SSH	22	X	-
SQUID/Proxy	3128	X	-

Legende	
X	Protokoll wird verwendet
-	Protokoll wird nicht verwendet

Tabelle 2: Dienste in der DMZ

In der nachfolgenden Tabelle werden die IP-Adressen für die Serversysteme und Firewalls festgelegt:

IP-Adresse	Name	Port / Dienst
10.0.0.100	Cluster	22 / SSH
10.0.0.10	HA-Links	-
10.0.0.11	HA-Rechts	-
10.0.0.1 192.168.x.x	Firewall „Außen“	-
192.168.10.1 10.0.0.2	Firewall „Innen“	-
10.0.0.101	Email-Server	25 / SMTP 110 / POP3 143 / IMAP
10.0.0.102	OPSI-Server	4441 / OPSI 67/ 68 / DHCP 53 / DNS
10.0.0.103	DNS/DHCP-Server	67/ 68 / DHCP 53 / DNS
10.0.0.104	File-Server	445 / SMB
10.0.0.105	LDAP-Server	389 / LDAP
10.0.0.106	Proxy-Server	3128 / HTTP (Squid)
10.0.0.107	Web-Server	80 / HTTP

Tabelle 3: IP-Adressen der Server in der DMZ

In der Regel sind die Serversysteme mit jeweils einer Netzwerkkarte ausgestattet. Eine Ausnahme machen hier jedoch die Firewallsysteme. Diese verfügen über zwei Schnittstellen. Die Einrichtung der Netzwerkkarten für die Firewallsysteme wird im Kapitel „*Konfiguration der Firewallsysteme*“. Die festgelegten IP-Adressen und Portnummern werden in der nachfolgenden Grafik noch einmal im Gesamtkontext des Projektes „Schul-IT“ dargestellt. Auf der Grafik lässt sich das bereits definierte Clustersystem erkennen. HA-Links und HA-Rechts stellen die Clusterknoten dar. Diese sind vom Aufbau redundant. D.h. fällt einer der Clusterknoten aus, übernimmt der andere Knoten die Aufgaben. Das Gesamtsystem ist unter der Cluster-IP-Adresse 10.0.0.100 zu erreichen. Innerhalb eines Clusterknotens befindet sich die virtuelle Infrastruktur, unter der die Serversysteme und Firewalls zu finden sind.

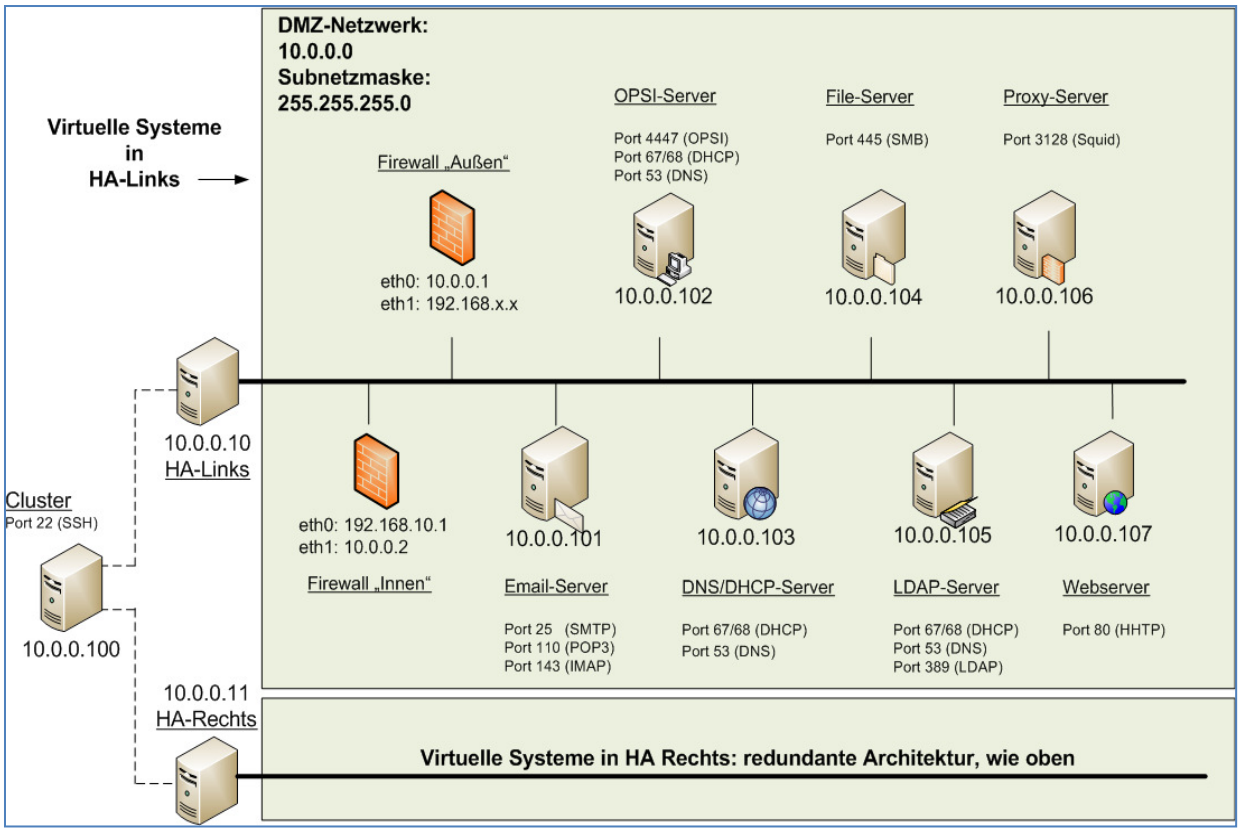


Abbildung 2: Struktur des Clustersystems

5. Systemumgebung der virtuellen Maschinen

In diesem Kapitel wird beschrieben, welche Einstellungen auf Seiten der virtuellen Betriebsumgebung gemacht werden müssen. Hierzu zählen die Hardware-Ressourcen die bereitgestellt werden müssen, ebenso wie die Einstellungen der Netzwerkkarten. Wie bereits erwähnt, werden die Firewallsysteme auf virtuellen Maschinen installiert.
Die hier beschriebenen Einstellungen müssen somit in der Vmware Server Console vorgenommen werden. Die Ausstattung einer solchen Maschine wurde folgendermaßen festgelegt:

Ausstattung	
Prozessor:	1
CD-Laufwerk:	1
Netzwerkschnittstellen:	2
Festplatten:	HD1: 30 GB Betriebssystem HD2: 50 GB Daten
Arbeitsspeicher:	512 MB

Tabelle 4: Ausstattung der virtuellen Maschinen

Jedes der Firewallsysteme bekommt in der Grundinstallation zwei Netzwerkkarten zugeordnet. Diese werden benötigt, um den Datenverkehr zwischen den Netzwerksegmenten zu

gewährleisten. Es wurden für die Netzwerkschnittstellen der Firewallsysteme folgende Einstellungen festgelegt:

Firewall Außen	
Schnittstelle	Netzwerkmode:
eth0	Vmnet1
eth1	Bridged

Tabelle 5: Netzwerkkonfiguration - "Firewall Außen"

Firewall Innen	
Schnittstelle	Netzwerkmode:
eth0	Bridged
eth1	Vmnet1

Tabelle 6: Netzwerkkonfiguration - "Firewall Innen"

6. Konfiguration der Firewallsysteme

In diesem Kapitel werden die Einstellungen für Netzwerkkarten, Hostnamen und Routing-Einträge festgelegt.

6.1. Netzwerkkarten

Um die Firewallsysteme netzwerkfähig zu machen müssen Einstellungen, wie IP-Adresse, Subnetzmaske und Standard-Gateways an den Netzwerkkarten vorgenommen werden. Dabei richtet sich die Konfiguration der Netzwerkkarten für die Firewallsysteme nach dem Entwurf der Netzwerkarchitektur. Wie schon aus Kapitel 4.2 bekannt verfügen die Firewallsysteme je über zwei Netzwerkschnittstellen. Die Konfiguration der Schnittstellen soll nun weiter präzisiert werden:

Firewall Außen	
Schnittstelle	IP-Adresse/Subnetzmaske:
eth0	10.0.0.1 / 24
eth1	192.168.x.x / 24

Tabelle 7: IP-Adressierung - "Firewall Außen"

Firewall Innen	
Schnittstelle	IP-Adresse/Subnetzmaske:
eth0	192.168.10.1 / 24
eth1	10.0.0.2 / 24

Tabelle 8: IP-Adressierung - "Firewall Innen"

Durch die Auswahl der IP-Adressen wird sichergestellt, dass die Firewalls in je zwei unterschiedlichen Netzwerksegmenten erreichbar sind.
Im Anhang „*Konfiguration-Netzwerkeinstellungen"* befindet sich eine detaillierte Beschreibung, wie die IP-Adressen und die Subnetzmaske auf die Netzwerkschnittstellen gebunden werden.

6.2. Hostname

Um die einzelnen Server und Client-Systeme im Netzwerk erkennen zu können, ist es möglich jedem System einen eindeutigen Systemnamen zuzuordnen.
Durch die Angabe einer sogenannten Domain bekommt das System zusätzlich eine logische Zuordnung zu einem Namensraum. Die Kombination aus Hostname und Domain identifiziert das System im Netzwerk.
Für die Firewallsysteme im Projekt „Schul-IT" wurden folgende Hostnamen und Namensräume ausgewählt:

- Firewall „Innen": fw-intern
- Firewall „Außen": fw-extern

Beiden Systeme wurde der Namensraum „school" zugeordnet. Damit die Firewallsysteme DNS-Namen (Domain Name Service) auflösen können, ist es notwendig den DNS-Server des Netzwerkes einzutragen. Anforderungen an eine Auflösung von Systemnamen werden dann an diesen Server weitergeleitet.
Die Beschreibung zur Einrichtung von Hostname und DNS-Server liegt in dem Anhang „*Konfiguration-Netzwerkeinstellungen"* bei.

6.3. Routing

Allein die Einrichtung der Netzwerkkarte und des Hostnamens genügen nicht, um die Firewall einsatzfähig zu machen. Da die Firewalls Datenpakete innerhalb der Netzwerke übertragen und weiterleiten müssen, werden sogenannte Routen (Wegfindungen) benötigt. Dabei bestimmt das Routing den gesamten Weg eines Nachrichtenstroms (der Datenpakete) durch die verschiedenen Netzwerke. Die Firewall führt also entsprechende Routinglisten, die es ermöglichen, dass die Datenpakete zu ihrem Ziel gelangen.
Die Firewallsysteme benötigen also Einträge in den Routinglisten, um Datenpakete zwischen Netzwerken zu transportieren. Damit dies im Schulnetzwerk erfolgreich durchgeführt werden kann, müssen nun nachfolgende Einträge zu den Routingtabellen hinzugefügt werden.
Zusätzlich wird der Eintrag für das Standard-Gateway benötigt. Das Default- oder Standard-Gateway ist ein System in einem Computernetzwerk, an das die Clients ihre Pakete senden, wenn die Ziel-IP-Adresse außerhalb des eigenen Netzwerks ist. Das Gateway verfügt entweder über die

Information wie das Zielnetzwerk zu erreichen ist, oder es leitet die Pakete an sein Default-Gateway weiter.

Die Beschreibung, wie dies mit Hilfe von Suse YAST realisiert werden kann liegt dem Anhang bei.

Routingtabelle Firewall Außen			
Ziel	Gateway	Subnetzmaske	Netzwerkschnittstelle
192.168.10.0	10.0.0.2	255.255.255.0	eth0

Tabelle 9: Routingtabelle - "Firewall Außen"

Standardgateway Firewall Außen	
Adresse	Netzwerkschnittstelle
192.168.x.x	eth1

Tabelle 10: Standardgateway - "Firewall Außen"

Der genannte Routingeintrag stellt sicher, dass die externe Firewall das Clientnetzwerk kennt. Pakete die auf der Firewall eintreffen und an das Clientnetzwerk adressiert sind, werden über das Gateway mit der IP-Adresse 10.0.0.2 (hier die Firewall „Innen“) und die Netzwerkschnittstelle eth0 weitergeleitet. Der Eintrag für das Standard-Gateway richtet sich nach der vorhanden Netzwerkarchitektur. Wird ein weiterer Router verwendet, so muss in das Feld für das Standard-Gateway die IP-Adresse des Routers eingetragen werden. Wird die Netzwerkschnittstelle eth1 der Firewall „Außen“ als DSL-Einwahl (PPPOE – Point-to-Point-Protokoll over Ethernet) verwendet, so wird das Standard-Gateway vom Internet Service Provider automatisch bereitgestellt und muss nicht eingetragen werden. In unserem Entwicklungsnetzwerk und auch später im Schulnetzwerk jedoch, ist ein zusätzliches DSL-Device als letztes Glied vorhanden. Somit muss die Firewall-Außen die unzustellbaren Pakete an das DSL-Device weiterleiten. Für das Netz der DSL-Router wurde das Netz 192.168.x.x/24 festgelegt.
Die Routingeinträge für die innere Firewall stellen sich wie folgt dar:

Routingtabelle Firewall Innen			
Ziel	Gateway	Subnetzmaske	Netzwerkschnittstelle
10.0.0.0	0.0.0.0	255.255.255.0	eth1
192.168.10.0	0.0.0.0	255.255.255.0	eth0

Tabelle 11: Routingtabelle - "Firewall Innen"

Standardgateway Firewall Innen	
Adresse	Netzwerkschnittstelle
10.0.0.1	eth1

Tabelle 12: Standardgateway - "Firewall Innen"

Mit diesen Routingeinträgen kennt die Firewall „Innen“ zwei Netzwerke. Zum Einen kennt die Firewall das Clientnetzwerk. Dies ist über die Netzwerkschnittstelle eth0 erreichbar. Der Eintrag 0.0.0.0 bezeichnet dabei eigenen Host im betreffenden Netzwerk. Zum Anderen kennt die Firewall auch das DMZ- bzw. Servernetzwerk. Dies ist über die Netzwerkschnittstelle eth1 erreichbar. Als Standard-Gateway ist die Firewall „Außen“ (10.0.0.1) eingetragen. Das heißt, dass alle Anfragen, die nicht von der internen Firewall selbst beantwortet werden können, werden an die 10.0.0.1 weitergeleitet. Falls sich als letztes Gerät nicht die „Firewall-Außen“ selbst, sondern ein anderes DSL-Device befindet, müssen in diesem ebenfalls die Routen angepasst werden.

7. Entwurf des Regelkonzepts

Im Projektziel wurde bereits grob auf die Regeln eingegangen, die umgesetzt werden sollen. Nun soll ein Konzept erstellt werden, welches als Vorlage für die Realisierung der Regeln dienen soll. Es wird jeweils für jede Firewall ein Konzept beschrieben.

7.1. Regel-Syntax

Vorab wurde jedoch eine Regel-Syntax festgelegt, die bei der Erstellung des Konzept und der anschließenden Implementierung behilflich sein soll.

Client-System -> Von-Netz -> Nach-Netz -> Dienst -> Regel

Die Syntax wird wie folgt interpretiert:

- **Client-System**
  Ist ein PC im Client-Netzwerk

- **Von-Netz**
  Das Netz, aus dem die Verbindung aufgebaut wird.

- **Nach-Netz**
  Das Netz, zu dem eine Verbindung aufgebaut wird.

- **Dienst**
  Der Dienst, bei dem es sich bei der Verbindung handelt.

- **Regel**
  Die Regel, die eintritt, falls die Bedingung auf die Verbindung zutrifft.

7.2. Firewall – Innen

Die Firewall-Innen hat den primären Schutz, den ankommenden Verkehr aus dem DMZ-Netz in das interne Netz zu kontrollieren bzw. zu filtern. Aus dem Client-Netz sollte nur der Verkehr auf die festgelegten Dienste zugelassen sein. Dies sind vorwiegend die DMZ-Dienste selbst und der Internetverkehr. Alles andere ist nicht zulässig.

Der Sachverhalt ist in der unteren Grafik dargestellt.

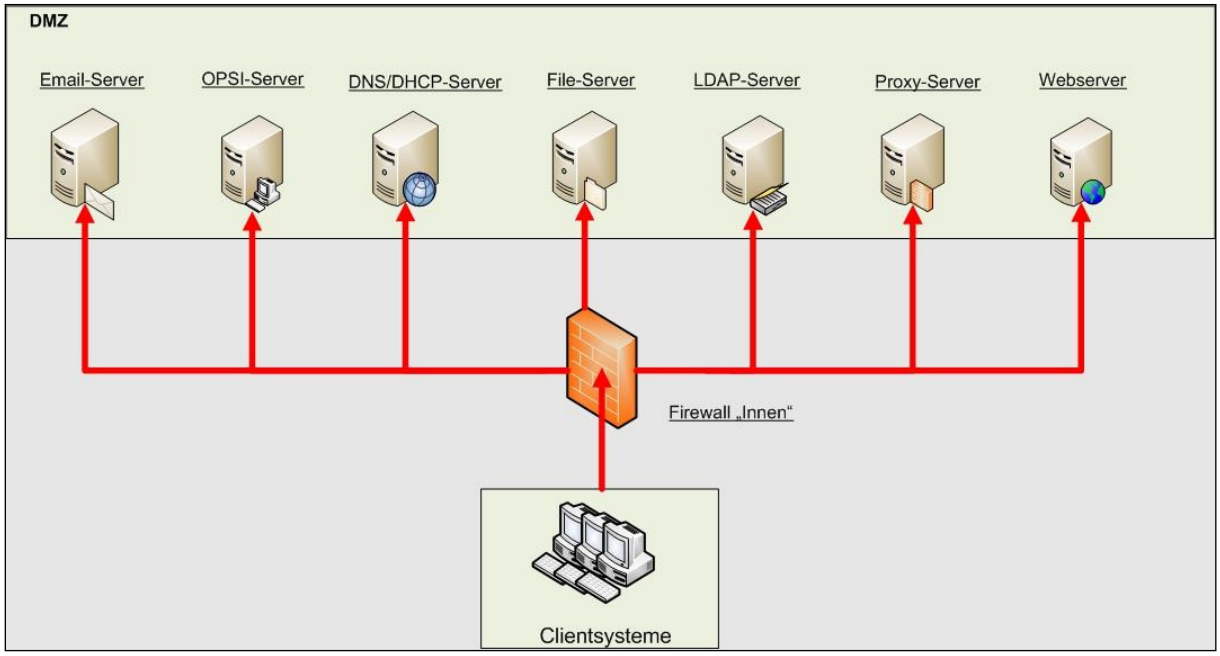


Abbildung 3: Regelkonzept - "Firewall-Innen"

Die Grafik stellt jeweils den initiierenden Verkehr dar. Damit ist der Aufbau einer Verbindung gemeint. Wird der Aufbau von der Firewall akzeptiert, so werden auch die nachfolgenden Pakete – dank der **Stateful-Inspection** Funktionalität – zugelassen. Ein Aufbau von außen nach innen wird nicht zugelassen.

Eine Tabelle soll nun die folgenden Regeln festhalten:

Von Netz	Nach Netz	Dienst	Regel
Client	DMZ	Email	Accept
Client	DMZ	OPSI	Accept
Client	DMZ	DNS/DHCP	Accept
Client	DMZ	File (Samba)	Accept
Client	DMZ	Proxy	Accept
Client	DMZ	HTTP	Accept
Client	Internet	HTTP	Accept
DMZ	Client	SSH	Accept
Client	DMZ	SSH	Accept
-	-	ICMP	Accept

Tabelle 13: Regeldefinition - "Firewall-Innen"

Alles was nicht in der Tabelle definiert ist, soll ausdrücklich verboten werden.

7.3. Firewall – Außen

Die Firewall-Außen ist gegenüber der Firewall-Innen einer größeren Bedrohung aus dem externen Netzwerk ausgesetzt. Deswegen bedarf dieses System einer höheren Sorgfalt hinsichtlich

der Konfiguration. Von außen darf nur bedingt eine Verbindung mit dem inneren Netz aufgebaut werden. Es müssen zuvor Dienste festgelegt werden, die von außen erreichbar sein sollen.
Eine Grafik soll den Sachverhalt darstellen:

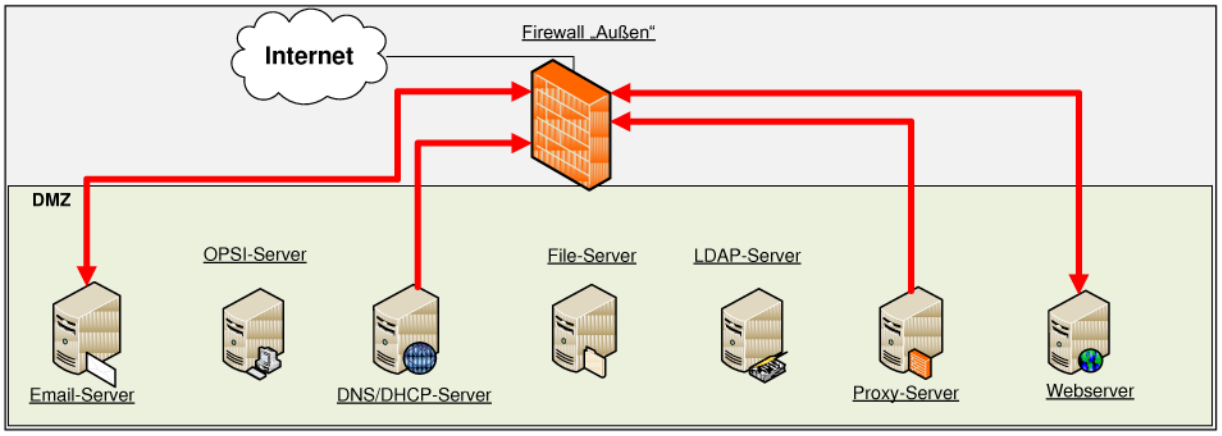


Abbildung 4: Regelkonzept - "Firewall-Außen"

Zusätzlich zu den gezeigten Diensten, wird auch eine zwecks Fernadministration eine Verbindung von außen nach innen benötigt. Hierfür könnte der SSH-Dienst genutzt werden. Im internen Netzwerk müsste ein Administrations-PC installiert werden, der das alleinige Recht besitzt sich über SSH mit den Firewall-Systemen zu verbinden. Eine Anfrage von außen auf den SSH-Port würde dann auf den jeweiligen Administrationsrechner weitergeleitet werden. Weiterhin wird für Diagnosezwecke wird das ICMP-Protokoll aus jedem Netz und dem Internet zugelassen.

Die Regel-Tabelle sehe dementsprechend folgendermaßen aus:

Von Netz	Nach Netz	Dienst	Regel
DMZ	Internet	Email	Accept
Internet	DMZ	Email	Accept
DMZ	Internet	DNS/DHCP	Accept
DMZ	Internet	Proxy	Accept
Internet	DMZ	HTTP	Accept
DMZ	Internet	HTTP	Accept
Internet	DMZ	SSH	Accept
Client	DMZ	SSH	Accept
-	-	ICMP	Accept

Tabelle 14: Regeldefinition - "Firewall-Innen"

Alles was nicht in der Tabelle definiert ist, soll ausdrücklich verboten werden.

8. Realisierung des Regelkonzepts mit iptables

Im vorigen Kapitel wurde die abstrakte Sicht der Regelstruktur behandelt. In diesem Kapitel wird nun auf die Realisierung des Regelkonzepts eingegangen. Dies geschieht konkret mit den zur Verfügung stehenden Mitteln, in unserem Falle iptables. Beim Einsatz eines Linux-Betriebssystems bleibt als Alternative auch nicht viel übrig. Denn iptables ist seit der

Kernelversion 2.4 ein Bestandteil des Kernels. Dies bedeutet, iptables ist fest in den Kernel integriert.
Die Paketfilter-Funktionalität wird in Linux bereits ab der Version 1.0 verwendet. Der erste Paketfilter und somit der Urvater von iptables war *ipfw.* Folgend mit der Kernelversion 2.0 erschien *ipfwadm* und mit der Version 2.2 darauf aufbauend *ipchains*. Letztendlich ist iptables der de facto Standard bei den Paketfiltern. Aufgrund der Beliebtheit von iptables, nutzen sogar viele kommerzielle oder kostenfreie Produkte iptables als Basis für die Paketfilterung.
Folgende wichtige Neuerungen sind in iptables eingeflossen:

- Stateful Inspection-Funktionalität
- Einführung der Tabellenstruktur
- Tabellen können jetzt zusätzlich auch mehrere selbstdefinierte Kette enthalten
- Die Source-NAT, Destination-NAT und Masquerade-Funktionalität wurde hinzugefügt
- Filterung auch nach Ethernet MAC-Adressen möglich
- Die Abweisung bei REJECT kann jetzt genauer definiert werden
- Limitierung von Paketen möglich, um sich zum Beispiel gegen DoS-Attacken zu wehren
- Die Detailtiefe kann beim Logging nun durch Level festgelegt werden

Doch bevor auch nur die erste Regel des Konzepts umgesetzt werden kann, ist das Wissen zur Funktionsweise und Aufbau von iptables von elementarer Bedeutung.
In den nachfolgenden Abschnitten wird somit auf die genannten Punkte eingegangen, um das Grundwissen über iptables zu vermitteln.

8.1. Aufbau

Das Grundkonzept von iptables besteht aus Tabellen, Ketten und Regeln. Es folgt eine detaillierte Beschreibung dieser drei Punkte und wie diese zu einander in Beziehung stehen.

8.1.1. Tabellen

Die Basis von iptables bilden die Tabellen. Das bedeutet, dass das ein Paket während seiner Verarbeitung im Kernel immer einer bestimmten Tabelle zugeordnet werden kann. Eine detaillierte Beschreibung des Ablaufs erfolgt im Abschnitt Funktionsweise.

In iptables sind vier Grund-Tabellen festdefiniert:

- **conntrack/raw**
  Diese Tabelle wurde eingeführt, um Pakete, bevor sie vom Connection Tracking erfasst werden, bereits behandeln zu können. Die Raw-Tabelle bietet den ersten Zugriff auf die ankommenden (PREROUTING) und die lokal erzeugten (OUTPUT) Pakete. Der Hauptzweck dieser Tabelle ist, dass Ausnahmen von der Paketfilterung definiert werden können. Damit könnten bestimmte Pakete, bei denen eine Verbindungsüberwachung sinnlos oder überflüssig ist, von dieser ausgenommen werden. Die eigentliche Filterung sollte in dieser Tabelle jedoch nicht vorgenommen werden.

- **mangle**
  Diese Tabelle enthält alle fünf Basis-Ketten. Diese Kette ist nicht dafür vorgesehen, zur Filterung oder zur Weiterleitung der Pakete verwendet zu werden. Primär ist diese Kette zur Manipulation des IP-Headers gedacht. Es können somit zum Beispiel die Attribute

TOS oder TTL verändert werden. Da diese Tabelle hauptsächlich für Sonderzwecke eingesetzt wird, wird diese im Laufe des Projekts auch nicht verwendet.

- **nat**
  Die nat-Tabelle sollte hauptsächlich zur Manipulation von Paket-Headern, bezüglich der IP-Adresse und/oder des Ports, verwendet werden. Das wären zum Bespiel Adress-Umsetzungen oder Paket-Weiterleitungen. Die Filterung der Pakete ist von den Entwicklern von iptables in dieser Tabelle **nicht** vorgesehen. Diese Tabelle besteht aus den Ketten PREROUTING, OUTPUT und POSTROUTING.

- **filter**
  Diese Tabelle ist ausschließlich zu der eigentlichen Filterung der Pakete vorgesehen. Das bedeutet dann auch, dass in dieser Tabelle die meisten Regeln vorzufinden sein werden. Die Tabelle besteht aus den drei Ketten FORWARD, INPUT und OUTPUT. In iptables gilt diese Tabelle als Standard-Tabelle.

Dies bedeutet, dass beim Einsatz von iptables auf ein Paket, sich das Paket immer in einem dieser vier Tabellen aufhält. Es können keine weiteren Tabellen hinzugefügt oder die bestehenden gelöscht werden. Eine Tabelle selbst kann man als einen Container betrachten, da eine Tabelle eine Menge von Ketten beinhaltet.

8.1.2. Ketten

Die Ketten sind den Tabellen sehr ähnlich. Auch hier gibt es festdefinierte Ketten:

- PREROUTING
  Das ist die erste Kette in iptables, die von dem Paket durchlaufen wird.

- FORWARD
  Diese Kette tritt in Kraft, wenn das ankommende Paket nicht an die jeweilige Netzwerkschnittstelle gerichtet ist, sondern weitergeleitet werden muss (routing/forward).

- INPUT
  Hier kommen nur Pakete an, die an die Netzwerkschnittstelle gerichtet sind und nicht weitergeleitet werden. Dies bedeutet, dass es sich entweder um ein Antwortpaket handelt, dass zu einem lokalen Prozess gehört oder es wird versucht, eine neue Verbindung mit der Netzwerkschnittstelle aufzubauen.

- OUTPUT
  Diese Kette wird durchlaufen, wenn die Pakete von einem lokalen Prozess initiiert wurden. Dabei kann es sich um eine neuen Verbindungsaufbau oder um ein Antwortpaket handeln.

- POSTROUTING
  Dies ist die letzte Kette in iptables. Diese Kette wird wiederrum jeweils von lokal-stammenden sowie auch von weitergeleiteten Paketen durchlaufen.

Der Ort eines Pakets kann jetzt noch genauer bestimmt werden. Denn jetzt ist zusätzlich zu der Tabelle auch die Information gegeben, in welcher Kette sich das Paket befindet. Doch im

Gegensatz zu den Tabellen, können weitere Ketten erstellt und diese auch wieder gelöscht werden. Eine Kette hat ebenfalls einen Container-Charakter, da sie selbst eine Menge an Regeln enthält.

8.1.3. Regeln

Die Regeln letztendlich dienen dem eigentlichen Filtern des Pakets. Es können in allen Tabellen und Ketten Regeln erstellt werden. Eine Regel wird immer an die nächste angefügt, sodass eine Schlange an Regeln in einer Kette entstehen kann.
Folgend eine grafische Darstellung des beschriebenen Aufbaus:

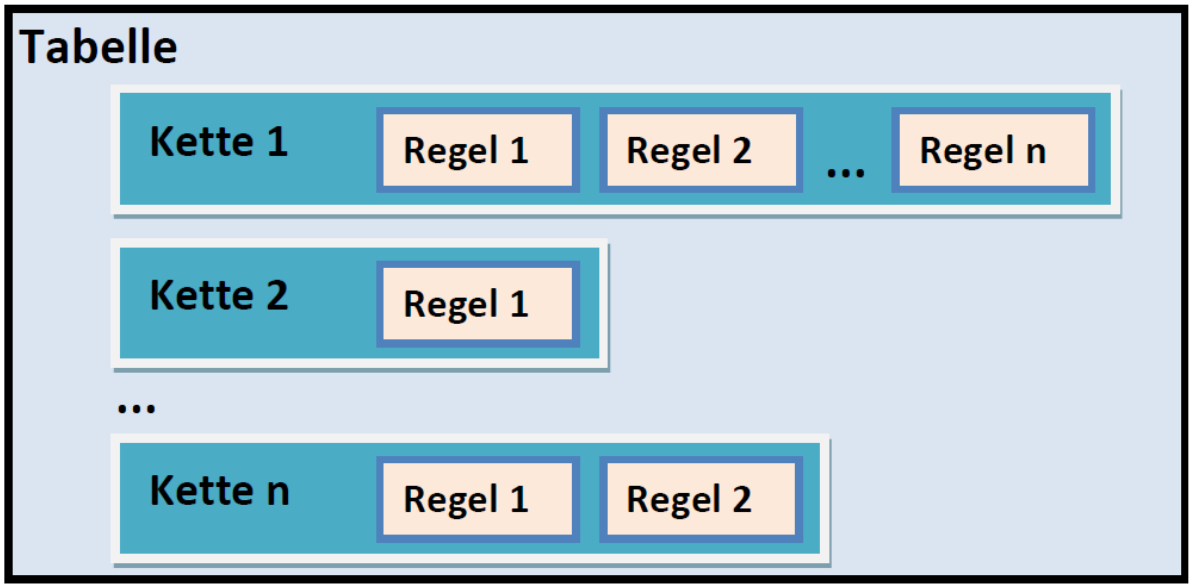


Abbildung 5: Tabellen, Ketten und Regeln in iptables

Jetzt soll noch mal auf den Zusammenhang zwischen den Tabellen und Ketten eingegangen werden. Die folgende Tabelle zeigt die verfügbaren beziehungsweise möglichen Ketten in den jeweiligen vier Tabellen an.

Tabelle	Kette
conntrack	PREROUTING
	OUTPUT
mangle	PREROUTING
	FORWARD
	INPUT
	OUTPUT
	POSTROUTING
nat	PREROUTING
	OUTPUT
	POSTROUTING
filter	FORWARD
	INPUT
	OUTPUT

Tabelle 1: Tabellen- und Kettenübersicht in iptables

ACHTUNG! Es handelt sich hierbei nicht um Kettengleichheit. So ist zum Beispiel die PREROUTING-Kette in der mangle-Tabelle eine andere als in der nat-Tabelle. Dies bedeutet, dass die Regeln in der Kette mangle/PREROUTING nicht in der Kette nat/PREROUTING sichtbar sind oder für diese gelten können.

8.2. Funktionsweise

Nach dem Aufbau von iptables, folgt die Funktionsweise dessen. Es soll das Funktionsprinzip beschrieben werden, dass iptables im Grunde so simpel aber auch zugleich sehr effektiv ausmacht.

8.2.1. Paketfluss

Das Wissen über den Aufbau von iptables im vorigen Kapitel wird nun eingesetzt, um den Weg des Pakets durch den Kernel darzustellen.
Beim Ankommen des Paktes an eine Netzwerkschnittstelle, legt dieses einen bestimmten Weg zurück. Um welche Strecke es sich dabei handelt, hängt ganz von den Eigenschaften des Pakets ab. Es durchwandert dabei die bereits erwähnten Tabellen und Ketten.

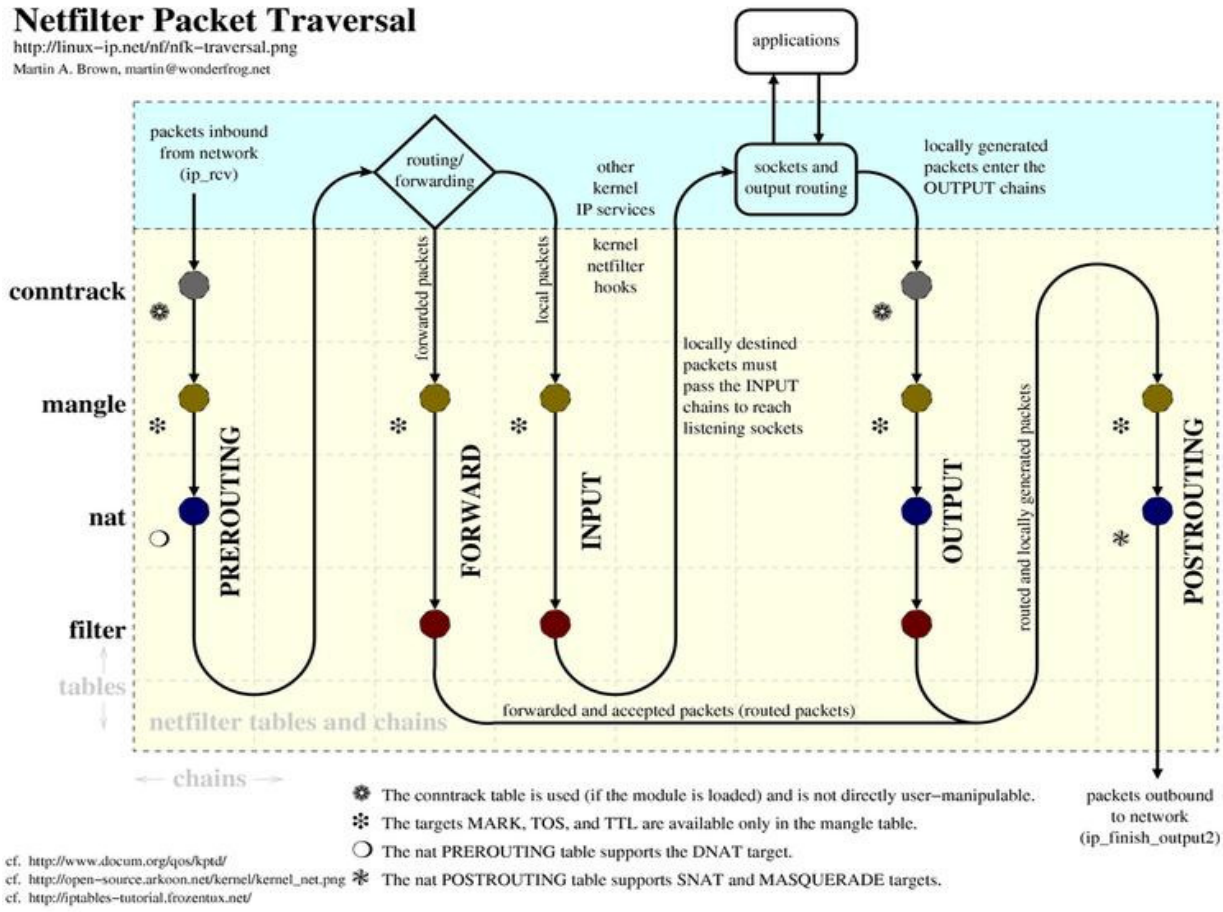


Abbildung 6: Paketfluss in iptables

Das Schaubild ist als eine Matrix aufgezeichnet:

- Y-Achse: Tabellen
- X-Achse: Ketten
- Aufenthaltsort = Punkt(X, Y)

Ein Punkt kennzeichnet dabei den Aufenthaltsort – bestehend aus Tabelle (Zeile) und Kette (Spalte) – des Pakets. So befindet sich das Paket beim Eintritt sofort in dem Punkt (PREROUTING, connttrack). Wird nun der Weg des Pakets weiterverfolgt, kommt es an einer Bedingung an, wo entschieden wird, welchen Weg das Paket weiter gehen soll. Die Entscheidung hängt dabei von zwei Fällen ab:

1. Das Paket ist für das lokale System selbst bestimmt ist
2. Das Paket ist an einen anderen Netzwerkteilnehmer gerichtet ist

Im ersten Fall, würde das Paket durch die INPUT-Kette laufen und letztendlich an einen Prozess übergeben werden. Im anderen Fall durchläuft das Paket die FORWARD-Kette. Hierbei würde auch die OUTPUT-Kette übersprungen werden, womit die letzte Kette POSTROUTING wäre.
Zum Schluss kann natürlich auch der Fall eintreten, dass Paket von einem lokalen Prozess erzeugt wird, was einem initiierendem Verbindungsaufbau entspräche. Dann aber würde das Paket bei der OUTPUT-Kette starten und als letzte ebenfalls die POSTROUTING-Kette passieren.

8.2.2. Paketauswertung

Die Auswertung der ankommenden Pakete erfolgt nach einem einfachen Prinzip:

1. Es wird die erste Regel in der Kette ausgeführt, in der sich das Paket gerade befindet. Ist keine Regel definiert, so greift die festgelegte Standard-Regel (Default Policy) in der Kette:
   - Paket verwerfen (DROP)
   - Paket akzeptieren (ACCEPT)

2. Die Regelbedingung wird mit dem Paket verglichen. Das Paket wird sozusagen einem Vergleich mit den in der Regel festgelegten Kriterien unterzogen:
   - WENN (Regelbedingung = Paketeigenschaft) dann ZIEL
   - ANSONSTEN nimm nächste Regel in der jeweiligen Kette

   Die Abarbeitung der Regeln erfolgt nach dem Top-Down Prinzip. Falls die erste Regel auf das Paket nicht zutrifft, wird gegen die nächste Regel geprüft.

3. Trifft keine der definierten Regeln in der Kette zu, greift wiederrum die Standard-Regel. Stimmt ein Paket mit der Regelbedingung überein, so wird das dafür definierte Ziel verwendet und die Abarbeitung des Pakets wäre damit beendet.

Der beschriebene Vorgang ist folgend als Flussdiagramm dargestellt:

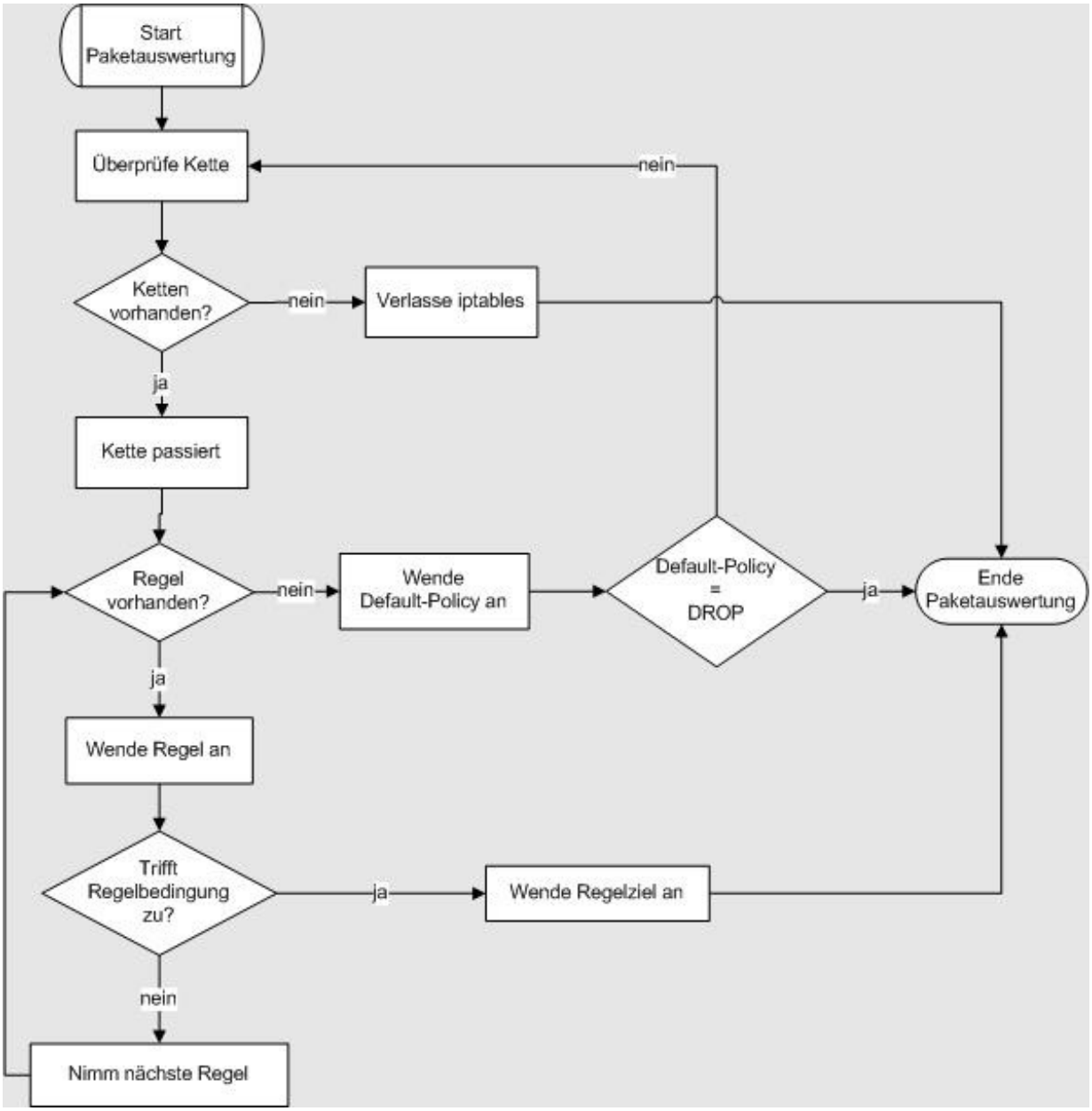


Abbildung 7: Paketauswertung

Nachfolgend werden nur die wesentlichen Ziele und deren Ketten, in denen diese Ziele möglich sind, einmal genauer beschrieben. Es existiert jedoch ein größerer Satz an Zielen in iptables, siehe hierfür [1].

Ziel	Beschreibung	Kette
ACCEPT	Das Paket wird akzeptiert.	FORWARD
		INPUT
		OUTPUT
DNAT	Mit diesem Ziel besteht die Möglichkeit, die IP-und TCP-Header zu verändern. Es kann ein anderes Ziel angegeben, womit das Paket dann an das neue Ziel gesendet wird.	Nat/PREROUTING
		Nat/OUTPUT
DROP	Das Paket wird verworfen. Es erfolgt keine Benachrichtigung an den Sender.	Alle

LOG	Das Paket wird vom Kernel protokolliert. Nach Ausführung dieses Ziels, geht es jedoch mit der nächsten Regel weiter. Dieses Ziel führt nicht zum Ende der Auswertung in iptables.	Alle
MASQUERADE	Mit diesem Ziel, wird die IP-Adresse des ausgehenden Pakets durch die lokale Adresse ersetzt (verschleiert). Dies ist dann sinnvoll, wenn ein Internetzugang von mehreren Teilnehmern genutzt werden soll und ohne dabei nach außen Informationen über das interne Netzwerk preiszugeben. So ist nach außen nur die IP-Adresse der Netzwerkschnittstelle sichtbar und die Quelladressen bleiben verborgen.	Nat/POSTROUTING
REDIRECT	Diese Ziel gleicht DNAT, mit dem einzigen Unterschied, dass das Paket an den lokalen Prozess selbst weitergeleitet wird.	Nat/PREROUTING
		Nat/OUTPUT
REJECT	Das Paket wird verworfen. Darüber erfolgt anschließend erfolgt eine Benachrichtigung an den Sender.	FORWARD
		INPUT
		OUTPUT
SNAT	Mit diesem Ziel werden, wie bei DNAT, die IP und TCP-Header des Pakets verändert. Es gleicht dem Funktionsprinzip von MASQUERADE, doch dieses Ziel sollte gewählt werden, falls eine statische IP-Adresse vorhanden ist. Dies kann zum Beispiel die Adresse des Internet-Providers sein. MASQUERADE wird hingegen verwendet, wenn die IP-Adresse dynamisch vergeben wird.	Nat/POSTROUTING

Tabelle 15: iptables-Zieltabelle

8.3. Syntax

Iptables bietet eine sehr umfangreiche Dokumentation bezüglich der Befehle und deren Verwendung. Die komplette Syntax hier nochmals aufzuführen würde den Rahmen dieser Arbeit bei weitem sprengen. Somit sei auch direkt auf die man-Pages von iptables verwiesen[1]. Diese stehen auch online zur Verfügung [2].

9. Firewall-Skripte

Erstmals hier kommt iptables zum Einsatz. Dabei kann die Erstellung der Regeln auf zwei verschiedenen Arten erfolgen. Entweder direkt auf der Kommandozeile durch die Eingabe der spezifischen Befehle. Der andere Weg wäre, das Festhalten der Regeln in einer entsprechenden Datei. Dies hat den Vorteil, dass die Regeln nach einem Neustart nicht neu erstellt werden müssen. Denn iptables verliert seine Einstellungen nach einem Neustart. Weiterhin ist die Übersicht über die Regeln gegeben und das Ändern oder Umsortieren der Regeln in einer Datei ist weitaus komfortabler als über die Kommandozeile. Der größte Vorteil, der durch eine Datei erreicht wird ist, dass die Portierung auf andere (Firewall-)Systeme ermöglicht wird. Und da das Projektnetzwerk zufälligerweise zwei Firewalls besitzt, können die Regeln der einen Firewall für die andere Firewall zum Teil übernommen werden.

[1] Auf der Kommandozeile: „man iptables“ ausführen

9.1. Regel-Skript

Das Regel-Skript kann mit jedem beliebigen Editor erstellt werden. So erfolgte auch die Implementierung der Regeln auf den Firewall-Systemen mit dem Editor *joe* [3]. Die Basis für die beiden Skripte bildete jeweils das Konzept aus Kapitel 7.
Die beiden Skripte sind dem Anhang *„Regel-Skript – Firewall-Innen“* und *„Regel-Skript – Firewall-Außen“* beigelegt.

9.2. Start-und Stopskript

Nach einem Neustart der Firewallsysteme sind alle angelegten Firewall-Regeln verloren. Dies liegt daran, dass die Firewall-Regeln nur temporär abgelegt werden. Aus diesem Grund wird ein Konzept benötigt, dass die Firewall-Regeln nach einem Neustart wieder erfolgreich aktiviert. Weiterhin sollte es nicht schaden, wenn dem Administrator ermöglicht wird, mit Hilfe eines Befehls die Firewall an der Serverkonsole manuell zu starten bzw. zu stoppen.
Dieses Kapitel beschreibt nun den Aufbau und die Installation eines Start- bzw. Stopskriptes um die Firewallsysteme zu steuern.

Insgesamt werden für beide Firewallsysteme sechs Dateien benötigt.

Firewall „Außen“:
- fw_extern.up
- fw_extern.down
- firewall

Firewall „Innen“:
- fw_intern.up
- fw_intern.down
- firewall

Die Skripte mit den Namen „fw-xxx.up“ und „fw-xxx.down“ enthalten die Einstellungen, welche die Firewall für das erfolgreiche Anlegen der Regeln benötigt. Diese Skripte sind bereits in dem vorhergehenden Kapitel erläutert worden. Die Datei „firewall“ ist nun das eigentliche Start- bzw. Stopskript.
Ein Ladevorgang der Firewallregeln stellt sich grob so dar:
- Sobald Runlevel 3 erreicht wurde wird das Skript „firewall“ mit dem Befehl „rcfirewall start“ geladen.
- Das Skript überprüft, ob die Dateien fw_xxx.up und fw_xxx.down verfügbar sind
- Es erfolgt ein Include des Skriptes fw_xxx.up, dass die Firewall-Regeln in Iptables einträgt

Wenn das System herunterfährt, so wird das Skript mit dem Befehl „rcfirewall stop“ ausgeführt. Der Unterschied zum oberen Ablauf ist, dass nun das Skript fw_xxx.down inkludiert wird und die Firewallregeln entfernt.
Im nachfolgenden Abschnitt wird nun erklärt, wie das Start- und Stopskript erstellt und für den Autostartmechanismus vorbereitet wird.

9.2.1. Vorbereitungen

Um die nachfolgenden Schritte durchzuführen, empfiehlt es sich als Benutzer „root“ am Linuxsystem anzumelden. Die Beschreibung erfolgt hier für ein Firewallsystem. Alle Schritte werden auf dem zweiten System analog durchgeführt.

Zunächst wird ein Ordner erstellt, der die Skript-Dateien enthalten soll. In der Konsole sind dazu folgende Befehle einzugeben:

```
fwproto:~# cd /
fwproto:~# mkdir fw
```

Es müssen die beiden Skripte fw-xxx.up und fw-xxx.down in dieses Verzeichnis kopiert werden. Der nächste Schritt ist das Anlegen des Start- bzw. Stopskriptes. Dazu wird in das Verzeichnis /etc/init.d gewechselt.

```
fwproto:~# cd /etc/init.d
```

Es folgt das Anlegen der Datei „firewall“ in diesem Ordner.

```
fwproto:~# touch firewall
```

Diese Datei wird wieder mit einem Texteditor geöffnet:

```
fwproto:~# joe firewall
```

Die Datei „firewall“ enthält den folgenden Code:

Code	Erläuterung
`#!/bin/sh`	Festlegen mit welchem Interpreter das Skript ausgeführt werden soll

`### BEGIN INIT INFO`
`# Provides: firewall`
`# Required-Start: $syslog $remote_fs`
`# Should-Start: $time ypbind`
`# Required-Stop: $syslog $remote_fs`
`# Should-Stop: $time ypbind smtp`
`# Default-Start: 3 5`
`# Default-Stop: 0 1 2 6`
`# Short-Description: Enabling firewall settings on iptables`
`# Description: Enabling firewall settings on iptables`
`### END INIT INFO`
	In dem Abschnitt INIT INFO werden Informationen für Yast wie z.B. Beschreibung und Systemparameter hinterlegt, die zum Starten benötigt werden. Hier wird u.a. festgelegt in welchem Runlevel das Skript beim Rechnerneustart starten soll.

`FW_BIN=/fw/fw_xxx.up` `FW_SHUTDOWN=/fw/fw_xxx.down`	Pfad der Skripte in Variablen

`test -x $FW_BIN || { echo „Start skript not available in /fw/ (fw_extern.up)“;`
`if [„$1“=“stop“]; then exit 0;`
`else exit 5; fi; }`
 	
`test -x $FW_SHUTDOWN || { echo „Stop skript not available in /fw/ (fw_extern.down)“;`
`if [„$1“=“stop“]; then exit 0;`
`else exit 5; fi; }`
	Check, ob Skripte auch verfügbar sind

`. /etc/rc.status`	Shell Funktion von Extern laden /etc/rc.status
`rc_reset`	Den Status des Service zurücksetzen

`case „$1“ in`
`start)`
`echo „Setting FW-Rules“`
`$FW_BIN`
`rc_status -v ;;`
`stop)`
`echo -n „Shutdown firewall“`
`$FW_SHUTDOWN`
`rc_status -v ;;`
`*)`
`echo „Usage: $0 {start|stop}“`
`exit 1 ;;`
`esac`
`rc_exit`
	Abfrage welcher Parameter hinter dem Befehl “rcfirewall” angegeben wurde
	Bei Parameter “start”, lade das Skript zum Setzen der Firewallregeln
	
	Bei Parameter “stop”, lade das Skript zum Löschen der Firewallregeln
	
	Ausgabe, wenn kein Parameter angegeben wurde
	
	Skript abschließen

Tabelle 16: Startskript

Nachdem das Start- bzw. Stopskript erfolgreich angelegt wurde, muss nun ein Verlinkung in das Verzeichnis /sbin/rcfirewall erstellt werden. Diese dient dazu, dass der Skriptaufruf „rcfirewall“ global im System verfügbar ist:

```
fwproto:~# ln –s /etc/init.d/firewall  /sbin/rcfirewall
```

Damit die Skripte beim Neustart bzw. Shutdown in die entsprechenden Runlevel des Systems geladen werden können, müssen diese mit dem Programm *insserv* registriert werden:

```
fwproto:~# insserv /etc/init.d/firewall
```

Zum Starten und Beenden der Firewall können nun folgende Befehle vom Administrator aufgerufen werden:

```
fwproto:~# rcfirewall start      zum Start der Firewall
fwproto:~# rcfirewall stop       zum Beenden der Firewall
```

Auch nach einem Neustart des Systems wird das Startskript und somit auch das Skript zum Setzen der Firewall-Regeln ausgeführt.

10. Fazit

Nach dem Abschluss dieses Projekts, existieren zwei Firewall-Systeme – *Firewall-Außen* und *Firewall-Innen* – jeweils basierend auf dem Betriebssystem openSUSE 11.0. Die Implementierung des Firewall-Konzepts selbst erfolgte für beide Systeme mit iptables 1.4.0.
Um für das gesamte System einen größtmöglichen Schutz in dem DMZ- und Client-Netz zu bieten, wurde die folgende Philosophie verfolgt: „*Es wird alles geblockt, was nicht explizit zugelassen ist!*“ Die Umsetzung dessen erfolgte in allen Ketten der *filter* –Tabelle[2]. Dies bedeutet konkret, dass alle Pakete, für die keine Regeln zutreffen, am Ende verworfen werden.
Um zwischen den verschiedenen Netzen eine Kommunikation gewährleisten zu können, mussten entsprechende Routing-Einträge in den Firewalls vorgenommen werden.
Wie bereits in dem Kapitel „*Projektziel*“ erwähnt, sollten DHCP-Anfragen aus dem Client-Netz möglich sein. Dies wurde nun ebenfalls realisiert, so dass einem Client auch eine IP-Adresse zugewiesen werden kann. Um eine Namensauflösung aus dem Schul-Netz sowie auch dem DMZ-Netz bereit zu stellen, wurde die entsprechende Regel in beiden Systemen eingerichtet. Zusätzlich hat auch ein Client aus dem Schul-Netz Zugriff auf alle Dienste in der DMZ. Das bedeutet, dass ein Client eine Verbindung zur DMZ aufbauen kann, aber der Verbindungsaufbau aus der DMZ ins Client-Netz ist verboten. Was jedoch nicht bedeutet, dass die Antwort-Pakete aus der DMZ geblockt werden. Hierfür wird die Stateful Inspection Funktionalität in iptables eingesetzt. Das Stateful Inspection findet auch bei allen weiteren Paketen statt, wo auf einen expliziten Verbindungsaufbau geantwortet wird.
Den Clients ist es nun auch erlaubt am Internetgeschehen teilzuhaben. Der http-Verkehr jedoch wird über den DMZ-Proxy zwecks Authentifizierung und Filterung weitergeleitet. Es ist nun auch möglich den Webserver in der DMZ von außen zu erreichen. Die http-Anfragen werden durch

[2] Siehe Kapitel *„Realisierung des Regelkonzepts mit iptables“*

die äußere Firewall an den internen http-Dienst weitergereicht. Ebenso auch für den Mailverkehr. Bei diesbezüglichen Anfragen von außen erfolgt ebenfalls eine Weiterleitung an den Mailserver in der DMZ. Die Clients können nun Mail aus dem internen Netz verschicken und das Abholen der Mails ist sowohl aus dem internen wie auch dem externen (Internet) Netz möglich.
Für die Administration der Firewallsysteme wurde ein Zugang über das SSH-Protokoll realisiert. Dafür wurden in den Firewall-Systemen die entsprechenden Ports freigeschaltet. Für Diagnosezwecke der Systeme wird das Protokoll ICMP aus allen Netzen akzeptiert.
Zum Schutz der Systeme gegen Angriffe wird jedes Paket einem TCP-Flag-Check unterzogen. Somit werden Pakete abgewiesen, die eine ungültige TCP-Flag-Struktur aufweisen. Dieses Konzept findet auf beiden Systemen statt. Als weitere Sicherheitsfunktion wurde der Schutz gegen das sogenannte IP-Spoofing eingerichtet. Diese Funktion kommt ebenfalls auf beiden Systeme zum Tragen. Sobald einer der eben genannten Angriffe durchgeführt werden, erfolgt diesbezüglich eine Protokollierung. Hierfür sind die Logging-Regeln auf den Systemen zuständig.
Für die Regeln selbst existieren eigene Skripte. Die Skripte wurden auf beiden Firewalls in den Autostart eingetragen, so dass nach dem Start des Systems die Filterfunktionen aktiviert sind. Durch spezielle Kommandobefehle ist es nun möglich, die Firewall zu starten oder zu beenden. Weiterhin bieten die Skripte den Vorteil, dass zukünftige Änderungen oder Erweiterungen effektiver durchgeführt werden können. Zusätzlich ist eine Dokumentation der Regeln in den Skripten möglich, wodurch eine bessere Übersicht über die Regeln gegeben ist.

11. Problemfälle

Im Laufe des Projekts sind gewisse Problemfälle aufgekommen. Auf diese Probleme wird in diesem Kapitel näher eingegangen und soweit möglich, auch ein möglicher Lösungsansatz diskutiert.
Ein Hauptproblem beschäftigt sich mit der Skalierung des Client-Netzes. Während des Projekts fiel die Konzentration nur auf ein Netzwerk, dass durch die Firewall-Innen geschützt werden sollte. Dieses eine Netz wurde jedoch nur auf eine Klasse abgebildet. Eine Schule hat aber die Eigenschaft, mehr als einer Klasse zu besitzen. Folgt man nun der Idee, jeder Klasse ein eigenes Netz bereitzustellen, kommt man wohlmöglich, aus Ressourcen-Gründen, zu einem Konflikt. Folgende Punkte müssten bei der Vielzahl an Klassen beachtet werden:

- **Anbindung der Netze an die „Firewall-Innen“**
  Die innere Firewall benötigt zu jedem Netz eine logische Verbindung. Dies kann über verschiedene Lösungsansätze realisiert werden:

  1. **Die Firewall benötigt gleichviele Netzwerkschnittstellen, wie Netze vorhanden sind.**
     Dieser Lösungsansatz stößt ab einer bestimmten Netzanzahl an physikalische Grenze. Zum Einem bieten ein Rechner aus der physikalischen Sicht nur einen begrenzte Anzahl an Erweiterungsmöglichkeiten und zum Anderen ist auch bei der virtuellen Maschine die Zahl der Netzwerkschnittstellen auf unter zehn festgelegt.

  2. **Virtuelle IP-Adressen**
     Der Firewall-Innen könnten die entsprechenden virtuellen IP-Adressen der verschiedenen Netze zugewiesen werden. So hätte die Firewall die gleiche Anzahl an IP-Adressen wie Anzahl Netze existieren.

3. **Ein Netz für Alle**
   Um statt auf das Klasse C Netz zurückzugreifen, könnte ein Klasse B Netz verwendet werden. Damit besteht die Möglichkeit bis zu 65352 IP-Adressen zu vergeben, womit das Problem mit mehreren Netzwerkschnittstellen gelöst wäre. Doch nun besteht nicht mehr der Grundgedanke, den man bei vielen Klasse C Netzen hatte, und zwar Filtermechanismen auf Klassenebene durchzuführen. So dass für die 5. Klasse andere Richtlinien gelten als für die 10. Klasse.

4. **Entsprechende Koppler verwenden**
   Bei Verwendung von Netzwerkkopplern, wie zum Beispiel dem Switch, können diese die Funktionalität der Netztrennung übernehmen. Viele dieser Geräte haben als Feature die VLAN-Funktion integriert. Womit Physikalische Netze getrennt werden könnten. Doch diese Implementierung ist sehr kostspielig, da mehrere Geräte angeschafft werden müssten und somit gegen die Projektphilosophie[3] widersprochen wird.

5. **Subnetting**
   Beim Einsatz von Subnetting könnte zum Beispiel das Klasse B Netz in mehrere kleine Teilnetze aufgeteilt werden. Dies hat zur Folge, dass es mehrere IP-Adress-Verwaltungsbereiche bzw. Segmente gibt. Dies vereinfacht die Verwaltung. Diese Lösung erfordert allerdings eine genaue Netzwerk-Planung der zuständigen Administratoren. Mit Subnetting lässt sich auch der Traffic durch Broadcasts minimieren, da diese nun nur innerhalb der „kleinen" Netzwerksegmente bleiben. Nachteil dieser Lösung ist, dass ggf. Router oder Switche zwischen den Netzwerken eingesetzt werden müssen, um eine Datenübertragung in andere Netze zu gewährleisten.

Das Problem verschiedener Netze zieht jedoch auch andere Punkte nach sich. Denn abhängig von der gewählten Netzwerkstruktur, müsste auch ein Konzept zur Verteilung der IP-Adressen entworfen werden. Denn bei mehreren Netzen, bedarf es an einem DHCP-Server, der mit diesen Netzen umgehen kann. So müssten auch dementsprechend viele DHCP-Relay-Agents installiert werden.

- **Speicherplatz für Log-Daten**
  Zur Speicherung der im Laufe der Zeit entstandenen Log-Dateien, muss auch ausreichend Speicherplatz bevor der Betriebssystem-Installation berücksichtigt werden. Da virtuelle Systeme als Basis dienen, könnte eine eigenes virtuelles Laufwerk hierfür vorgesehen werden. Das Laufwerk würde dann als eine zusätzliche Partition in der jeweiligen Firewall eingehängt werden.

- **Flaschenhals-Problem**
  Beim Einsatz von virtuellen Maschinen, werden die internen Netzwerkkarten simuliert. Was letztendlich bedeutet, das am Ende der Weg der Pakete durch die physikalische Schnittstelle erfolgt. Bei mehreren virtuellen Maschinen bzw. den jeweiligen Diensten bedeutet dies, dass diese alle nur eine Netzwerkschnittstelle nutzen. Abhängig von den Clients im Schulnetz, könnte die Gefahr eines Flaschenhalsproblems entstehen. Dies müsste beobachtet und getestet werden. Technisch könnte dem Probleme vorgebeugt

[3] Zielsumme = 0 !

werden, indem gleich darauf geachtet wird, dass die physikalische Netzwerkschnittstelle die schnellste Übertragungsgeschwindigkeit unterstützt.

- Sicherheitsrisiko bei virtuellen Maschinen
  Die Firewall „Außen“ ist der größten Bedrohung durch Angriffe und nicht-autorisierten Zugriffen aus dem Internet ausgesetzt. Dies bringt einen entscheidenden Nachteil mit sich: Gelingt es einem Angreifer die Kontrolle über das Firewallsystem zu erhalten, so befindet sich dieser auch auf dem Cluster und erhält somit auch die volle Kontrolle über die anderen Serversysteme.
  Dieses Szenario bietet genug Anlass darüber nachzudenken, ob es nicht vielleicht sinnvoller wäre, die Firewall „Außen“ als eigenständiges, unabhängiges und physikalisches (also keine virtuelle Maschine) System zu installieren. Sollte es nun einem Angreifer tatsächlich gelingen die Firewall zu kompromittieren, so erhält er nicht sofort den Zugriff auf das Clustersystem, dass evtl. durch weitere Schutzmaßnahmen, wie z.B. komplexe Passwörter geschützt ist.

12. Ausblick

Viele der genannten Anforderungen aus den Projektzielen konnten im Verlauf der Projektarbeit realisiert werden. Dennoch gibt es einige Punkte, die nicht ausreichend bearbeitet werden konnten, da der Zeitrahmen eine weitere, intensivere Bearbeitung dieser Themen nicht zuließ. In diesem Kapitel soll nun ein Ausblick gegeben werden, auf die Aufgaben die noch anstehen werden bzw. noch einmal einer Überarbeitung bedürfen.
Zunächst sind die noch fehlenden Firewall-Regeln zu nennen. Für die erste Teststellung der Firewallsysteme wurde bereits eine Reihe von Firewall-Regeln definiert. In einem großen Netzwerk sind allerdings sehr viele Dienste und Services verfügbar, so dass weitere Regeln erstellt und implementiert werden müssten. Die restlichen Dienste, die noch berücksichtigt werden müssten, wären:

- OPSI: Für das Desktopmanagementsystem müssen Regeln definiert werden, die die Kommunikation des OPSI Servers mit den Clients zulässt. Hierbei werden auch die zusätzlichen Netzwerkdienste DHCP und DNS benötigt.

- LDAP: Damit Benutzerkonten und Passwörter im Schul-Netz verwaltet werden können und somit eine Authentifizierungsmöglichkeit für Desktopsysteme, Mail und Proxy zur Verfügung stehen, muss der Datenverkehr, der über das LDAP-Protokoll stattfindet über die Firewallsysteme erlaubt werden. Auch hier müssen noch Regeln definiert werden.
- SMB: Um einen Datenaustausch und Druckerfreigaben zwischen Client und Serversystemen zu ermöglichen, müssen Regeln für das SMB-Protokoll implementiert werden. Dieser Dienst wird in der Regel von einem File-Server angeboten.
- Andere: Werden in dem jeweiligen Schulnetzwerk noch weitere Services und Dienste angeboten, so müssen auch für diese Datenübertragungen Firewall-Regeln gegebenenfalls auf beiden Systemen festgelegt werden.

Während der Projektarbeit, wurde primär auf der Konsole des jeweiligen Betriebssystems gearbeitet. Um einen etwas „komfortableren“ Weg nutzen zu können, bedarf es einer Integration einer grafischen Benutzungsoberfläche (GUI) für das Firewallsystem. Angebracht wäre hier, dem

Administrator über einen externen Client (Verwaltungsprogramm) oder einer webbasierten Lösung (Verwaltung über den Browser) die Möglichkeit zu geben, Einstellungen an der Firewall vornehmen zu lassen. Dies könnte einen Vorteil in der Übersichtlichkeit und somit zur Vermeidung von Fehlern führen. Desweiteren könnten die Logging-Protokolle auf eine übersichtliche Art und Weise dargestellt werden, um die Lesbarkeit der Informationen zu erhöhen.
Bezüglich der Protokollierung, die vorwiegend während kritischen Situationen stattfindet, ist es essenziell wichtig einen Benachrichtigungsmechanismus bei Events, wie z.B. Angriffen, zu besitzen. Denn allein die Protokollierung reicht in den meisten Fällen nicht aus, weil nur auf die Ausgabe dieser Informationen keine Reaktion des Administrators erfolgen kann, wenn dieser von dem Geschehen nichts mitbekommt. Dabei sollte die Möglichkeit bestehen verantwortliche Administratoren z.B. via Email, SMS oder auf anderen Kommunikationswegen im Falle eines Events zu informieren. Die Erstellung dieses Konzepts gehört ebenfalls zu den offenen Aufgaben.
Da in diesem Projekt ein sehr großer Wert auf die Modularisierung und Flexibilität der Systeme gelegt wurde, sollte auch auf eine Modularisierung der Skripte geachtet werden. Das heißt, dass die Skripte zum Einen so erstellt werden sollten, dass sie ohne große Probleme auf ein anderes System übertragen werden können. Zum Anderen könnte man aber auch die einzelnen Firewall-Regeln für die unterschiedlichen Services und Dienste so modular halten, dass sie je nach Bedarf in die Firewall-Regel-Skripte integriert werden können. Für jeden Dienst können somit auch individuell Regeln hinzugefügt oder gelöscht werden. Die Regeln könnten nach einem Baukasten-Prinzip zusammengebaut werden. Somit wäre die Redundanz, die die Firewallsysteme bei einigen Regeln besitzen, vermieden werden, weil die Systeme die gleichen Modul-Skripte nutzen würden.
Ein weiterer Punkt für den Ausbau der Firewallsysteme besteht in der Integration schon bestehender Regeln für den Schutz vor bekannten Attacken. In dieser Projektarbeit wurden bereits zwei Sicherheitssysteme (IP-Spoofing und TCP-Flag-Check) integriert. Dennoch gibt es weitere bereits definierte Sicherheitseinstellungen für iptables, deren Integration noch Sinn machen würde. Auch diese Schutzmaßnahmen könnten soweit modularisiert werden, dass einfach nur ein „include“ in das Regelwerk nötig ausreichen würde.
In dem Verlauf der Projektarbeit wurde, wie bereits bekannt, die Firewall-Regeln für ein Client-Netzwerk definiert. Eine weitere Herausforderung für das Projekt Firewall wird es nun noch sein, das Konzept für weitere Client-Netzwerke auszuarbeiten. Dabei muss enorm auf die Skalierbarkeit der Systeme geachtet werden, um jedwede Anzahl von Client-Netzwerken abdecken und schützen zu können. Das Hinzufügen weiterer Netzwerke würde auch eine Änderung bei den jetzigen Regelskripten nach sich ziehen. An dieser Stelle fehlt ebenfalls ein Konzept, dass ausgearbeitet werden müsste.

13. Glossar

Begriff	Definition
Client	Ein Client ist ein Anwendungsprogramm, das Kontakt zu einem anderen Anwendungsprogramm (Server) aufnimmt, um dessen angebotene Services und Dienste in Anspruch nimmt.
Default Policy (Standard-Policy)	Trifft keine Firewall-Filterregel auf ein Datenpaket zu, wird mit dem Paket nach der Default-Policy verfahren. Die Default-Policy befindet sich am Ende einer Regelkette und trifft zu, wenn keine vorhergehende Regel angewendet wurde. Die Standard-Einstellung sollte das Verwerfen (DROP) der Pakete sein.
DHCP (Dynamic Host Configuration Protocol)	Protokoll zu automatischen, zeitlich begrenzten Vergabe von IP-Adressen in einem IP-Netzwerk, basierend auf dem Protokoll BOOTP.
DMZ (Demilitarisierte Zone)	Eine demilitarisierte bezeichnet ein Netzwerk, das durch Sicherheitsmaßnahmen geschützt ist, aber dennoch den (kontrollierten) Zugriff auf Server und Dienste erlaubt.
DNAT (Destination Network Address Translation)	Das als DNAT bezeichnete Verfahren wird oft genutzt, um bestimmte Verbindungen aus dem Internet an einen Server im eigentlich geschützten lokalen Netzwerk weiterzuleiten.
DNS (Domain Name System)	Protokoll zum Auflösen von Host- und Domainnamen.
DSL-Device	Das Gerät, das einen Zugang zum Internet via Digital Subscriber Line ermöglicht (digitaler Breitband-Internetzugang).
FTP (File Transfer Protocol)	Übertragungsprotokoll zum ungesicherten Datenaustausch von Dateien über ein Netzwerk .
Gast	Gast bezeichnet das virtuelle Betriebssystem das unter dem Wirtssystem Betrieben wird.
HA (High Availability)	Hochverfügbarkeit
Host	Rechner in einem TCP/IP-Netzwerk.
http (Hyper Text Transfer Protocol)	http ist ein Protokoll zur Übertragung von Daten über ein Netzwerk. Es wird hauptsächlich eingesetzt, um Webseiten aus dem World Wide Web (WWW) in einen Webbrowser zu laden.
ICMP (Internet Control Message Protocol)	Protokoll zum Austausch von Informations- und Fehlermeldungen über das Internet-Protokoll (IP).
IMAP (Internet Message Access Protocol)	Anwendungsprotokoll, dass den Zugriff auf die Verwaltung von Emails gestattet.
IP-Forward	Einstellmöglichkeit in Linux-Betriebssystem, um IP-Datenverkehr über den Kernel zuzulassen.
IP-Spoofing	Angriffsart auf Netzwerke, indem der Angreifer versucht mit Hilfe von bekannten internen oder externen IP-Adressen durch die Firewall zu kommen.
LDAP (Lightweight Directory Access Protocol)	Anwendungsprotokoll, dass die Abfrage und Modifikation von Verzeichnisdienst Informationen ermöglicht.

NX NOMACHINE	Software zur Remote-Administration von Computersystemen. Mit NX kann man den Bildschirminhalt eines entfernten Computers auf einen lokalen Rechner (auch betriebssystem-übergreifend) übertragen und damit arbeiten, als säße man direkt davor.
OPSI (Open PC Server Integration)	Desktop Management System für Windows Workstations, basierend auf Linux.
POP3 (Post Office Protocol Version 3)	Übertragungsprotokoll, welches ein Email-Client benutzt um Emails von einem Email-Server abzuholen.
PROXY	Ein Programm, das stellvertretend für interne Clients mit externen Servern kommuniziert. proxy = Stellvertreter
Server	Computer der in einem Netzwerk, Services und Dienste zur Verfügung stellt.
SMB (Server Message Block)	SMB ist ein Kommunikationsprotokoll für Datei-, Druck- und andere Serverdienste in Computernetzwerken.
SMTP (Simple Mail Transport Protocol)	Übertragungsprotokoll, das dazu dient Emails in einem Computernetzwerk zu übertragen.
SNAT	SNAT wird für die Adressenübersetzung zwischen zwei privaten Netzen genutzt. Der zwischen beiden Netzen liegende Router hat in seiner Routing-Tabelle die Netzadressen beider Netze abgelegt.
SPI (Stateful Packet Inspection)	Dynamische Filtertechnik eines Firewallsystems, bei dem jedes Datenpaket einer bestimmten aktiven Session zugeordnet wird.
SQUID	Squid ist ein freier Proxyserver.
SSH (Secure Shell)	SSH ermöglicht eine verschlüsselte und sichere Netzwerkverbindung mit einem entfernten Computer herzustellen.
TCP (Transmission Control Protocol)	TCP baut eine bestätigte Ende-zu-Ende-Verbindung auf und bietet eine Segment-Synchronisation.
UDP (User Datagram Protocol)	UDP ist ein verbindungsloses, unbestätigendes Netzwerkprotokoll.
Wirt	In einer virtuellen Umgebung der Server, der die virtuelle Betriebsumgebung zur Verfügung stellt.
YAST (Yet Another Setup Tool)	(Teilweise) grafisches Konfigurations- und Management-Tool unter Suse-Linux-Betriebssystemen.

14. Abbild-und Tabellenverzeichnis

15. Literaturverzeichnis

[1]. **Andreasson, Oskar.** Iptables Tutorial 1.2.2. [Online] 2006 . http://iptables-tutorial.frozentux.net/iptables-tutorial.html#TARGETS.
[2]. **Russell, Rusty.** linuxmanpages. [Online] Name.net LLC, 09 2002. [Zitat vom: 27. 06 2009.] http://www.linuxmanpages.com/man8/iptables.8.php.
[3]. Joe's Own Editor. [Online] [Zitat vom: 27. 06 2009.] http://joe-editor.sourceforge.net/.
[4]. VMware Cloud Computing with Virtualization, Green IT, Virtual Machine & Servers. [Online] VMware Inc., 2009. [Zitat vom: 28. 06 2009.] http://www.vmware.com.
[5]. Software.openSUSE.org. [Online] Novell, 2009. [Zitat vom: 28. 06 2009.] http://software.opensuse.org/old/11.0.
[6]. netfilter/iptables project homepage - The netfilter.org project. [Online] netfilter, 2008. [Zitat vom: 28. 06 2009.] http://www.netfilter.org/.

16. Anhang

16.1. Installation des Betriebssystems für die Firewallsysteme

16.2. Konfiguration – Netzwerkeinstellungen

16.3. Konfiguration – Hostname und DNS

16.4. Konfiguration – Routing

16.5. Start-Stopskript – „Firewall-Innen“

16.6. Start-Stopskript – „Firewall-Außen“

16.7. Regel-Skript – „Firewall-Innen“

16.8. Regel-Skript – „Firewall-Außen“

16.1. Installation des Betriebssystems für die Firewallsysteme

Nach dem Booten der openSUSE Installation von der Installation sollte zunächst die Sprache eingestellt werden.
Hier wird „Deutsch“ und „Deutsches Tastaturlayout“ ausgewählt. Nach dem Bestätigen der Lizenzvereinbarungen wird ein Systemcheck gemacht und man erhält die Möglichkeit den Installationsmodus auszuwählen. Folgende Einstellungen sind vorzunehmen:

Installationsmodus:	Neuinstallation
Benutze automatsiche Installation:	Ja

Im nächsten Schritt werden die Regionseinstellungen festgelegt:

Region:	Europa
Zeitzone:	Deutschland

Es wird empfohlen die aktuelle Uhrzeit einzustellen.

Grafische Benutzungsoberfläche:

Die grafische Benutzungsoberfläche ist KDE 3.5. Für einen reinen Serverbetrieb kann auch auf die grafische Benutzungsoberfläche verzichetet werden. Dann werden folgende Einstellung festgelegt:

Partitionierung:

Es folgt die Partitionierung der Festplatte. Hierbei wird folgende Partitionierung empfohlen:

Die Festplatte wird aufgeteilt in 3 Partitionen:

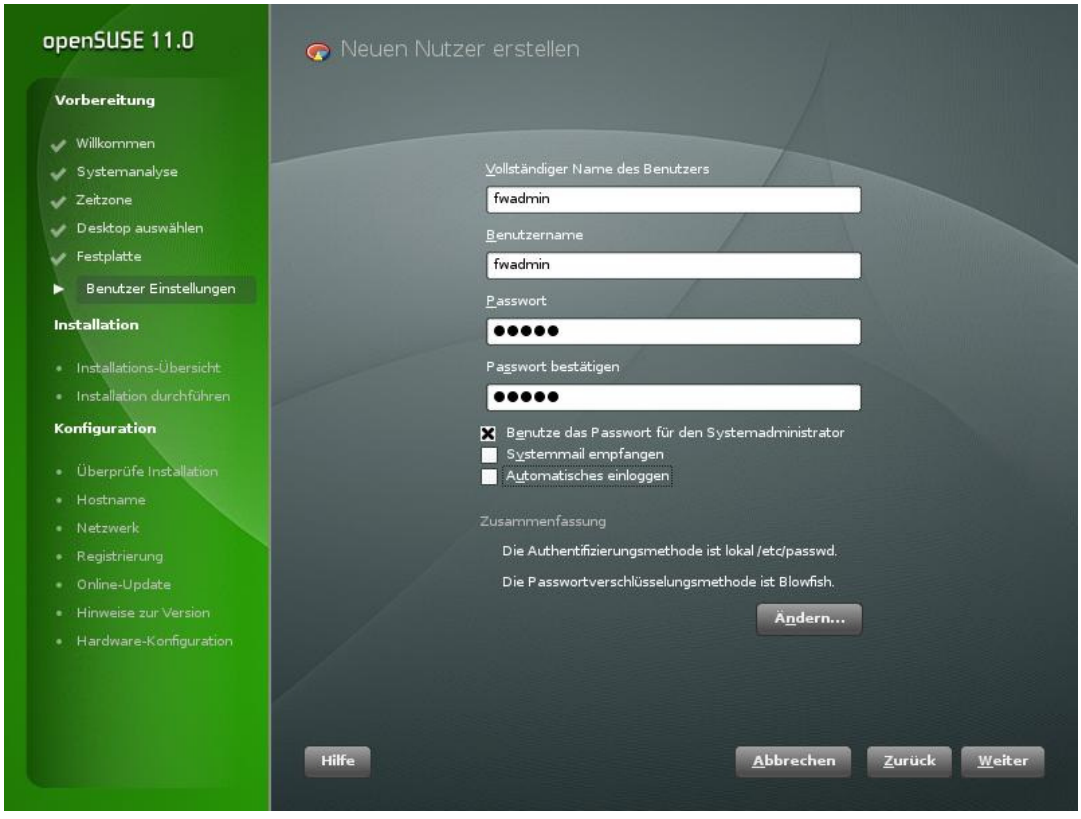
Partitionierung (sda)			
Gerät	Größe	Filesystem Type	Einhängepunkt
/dev/sda1	1.0 GB	EXT3	/boot
/dev/sda2	2.0 GB	Swap	Swap
/dev/sda3	27.0 GB	EXT3	/

Wenn eine weitere Datenfestplatte hinzugefügt wurde werden zusätzlich folgende Partitionen eingerichtet:

Partitionierung (sdb)			
Gerät	Größe	Filesystem Type	Einhängepunkt
/dev/sdb1	50.0 GB	EXT3	/data

Benutzer:

Es folgt die Einrichtung eines Benutzers. Hier werden folgende Einstellungen empfohlen:

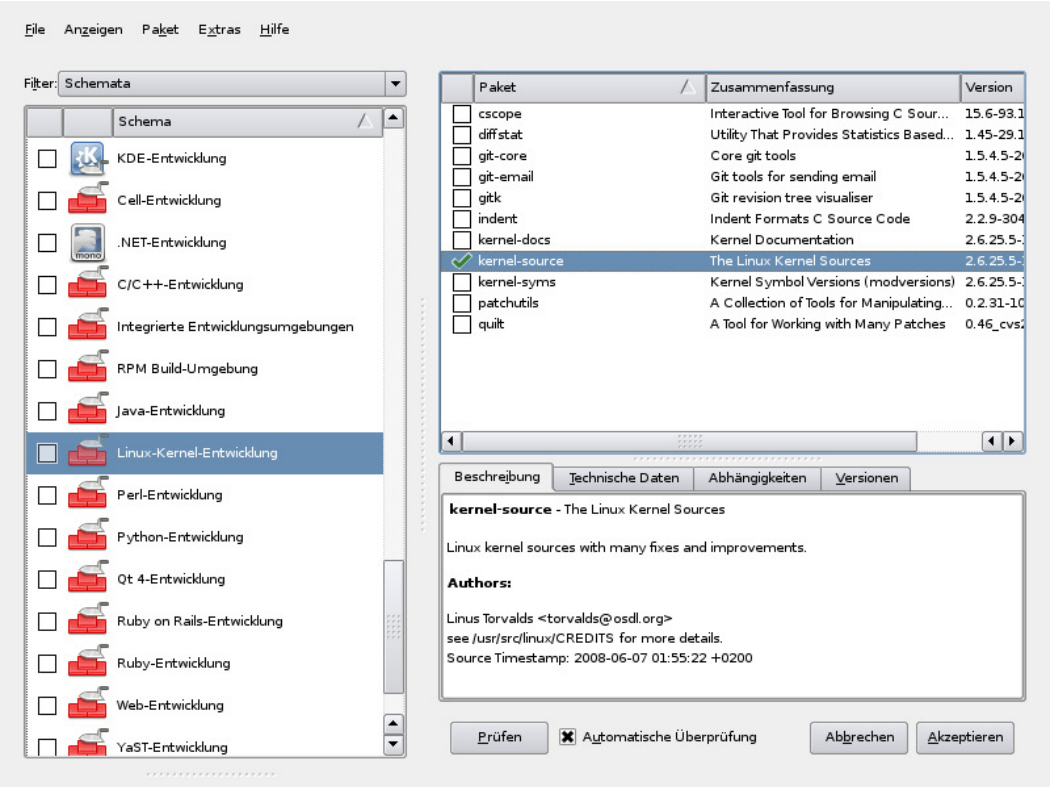


Als Passwort wird ein sicheres Passwort mit mindestens 8 Zeichen und hoher Komplexität[1] empfohlen.

Zusammenfassung:

Es folgt eine Zusammenfassung der bereits getätigten Einstellungen. Nun muss die Softwareauswahl angepasst werden. Dazu klickt man auf „Software-Auswahl“ und erhält ein Fenster indem man die zu installierende Software auswählen kann.
Wichtig: Um Konflikte zwischen SuseFirewall und iptables zu vermeiden wird empfohlen, alle Pakete für die SuseFirewall zu deaktivieren.
Neben den bereits aktivierten Softwarepaketen, müssen folgende Pakete zusätzlich aktiviert bzw. installiert werden:

- Konsolenwerkzeuge
- Dateiserver
- Netzwerkverwaltung
- Grundlegende Entwicklungsumgebung
- Linux-Kernel-Entwicklung
- Iptables
- Iptraf



Nachdem alle Software-Produkte ausgewählt wurden, kann mit „Akzeptieren“ die Installation gestartet werden. Das System wird nun installiert.
Nach erfolgreicher Installation sollte unbedingt ein Online-Update der Softwarekomponenten des Betriebssystems durchgeführt werden.

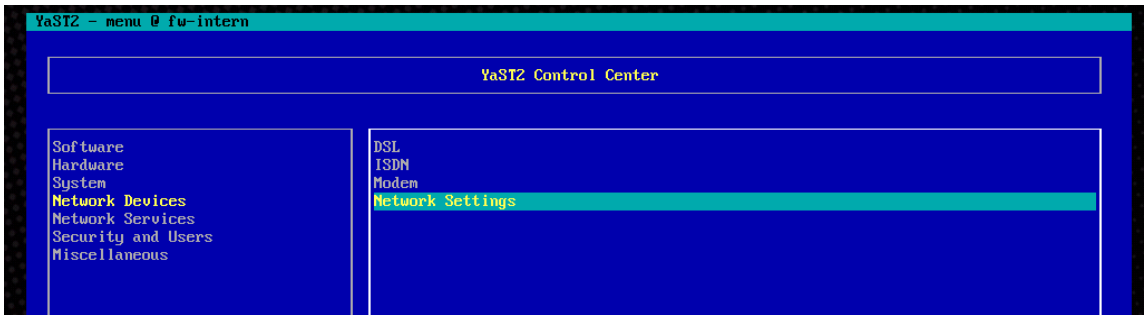
[1] Groß- und Kleinbuchstaben, Zahlen, Sonderzeichen

16.2. Konfiguration – Netzwerkeinstellungen

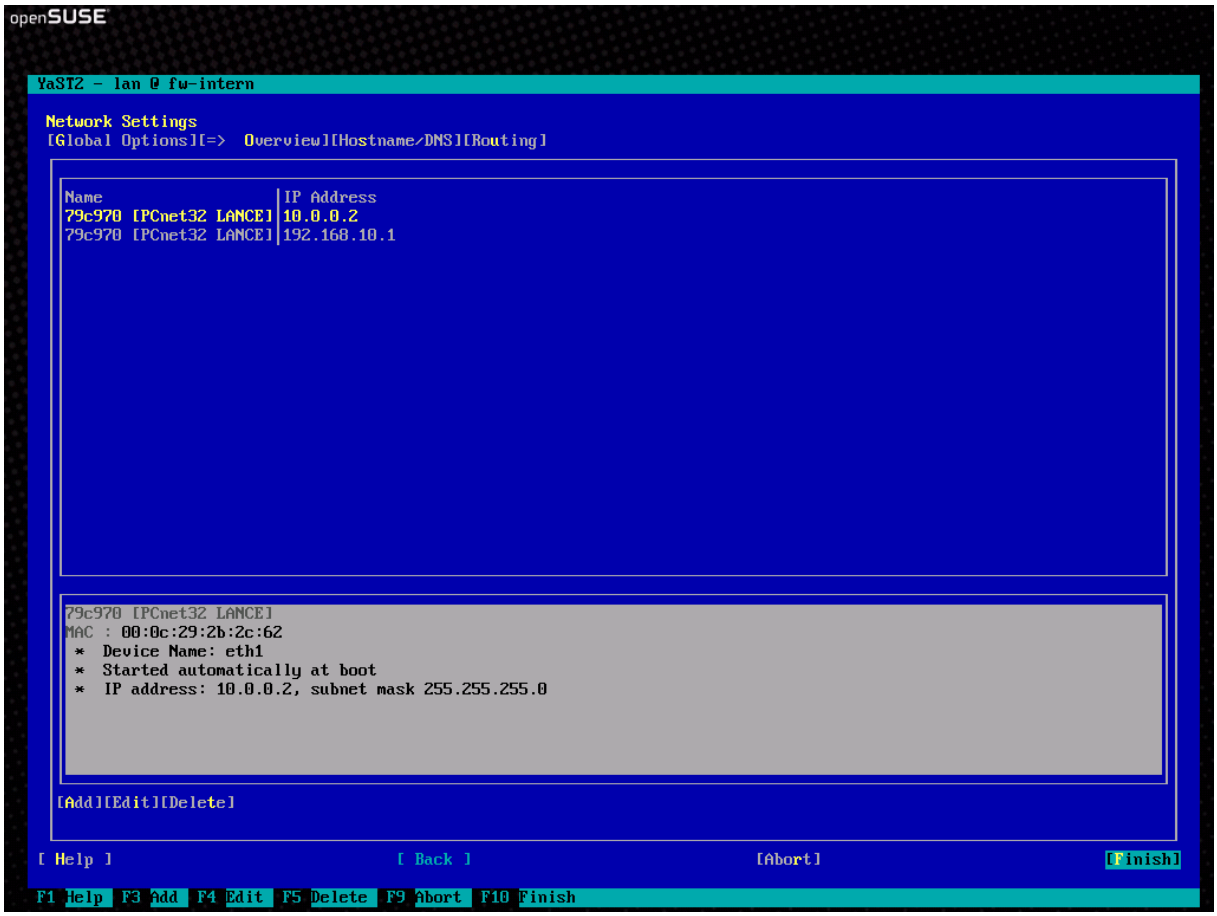
Um die Firewallsysteme netzwerkfähig zu machen, müssen Einstellungen, wie IP-Adresse, Subnetzmaske, Standard-Gateways und Routing-Einträgen an den Netzwerkkarten vorgenommen werden.
Mit Hilfe des Konfigurationstools YAST werden diese Einstellungen getätigt. Dazu wird in der Konsole folgender Befehl aufgerufen, um YAST zu starten:

```
fwproto:~# yast
```

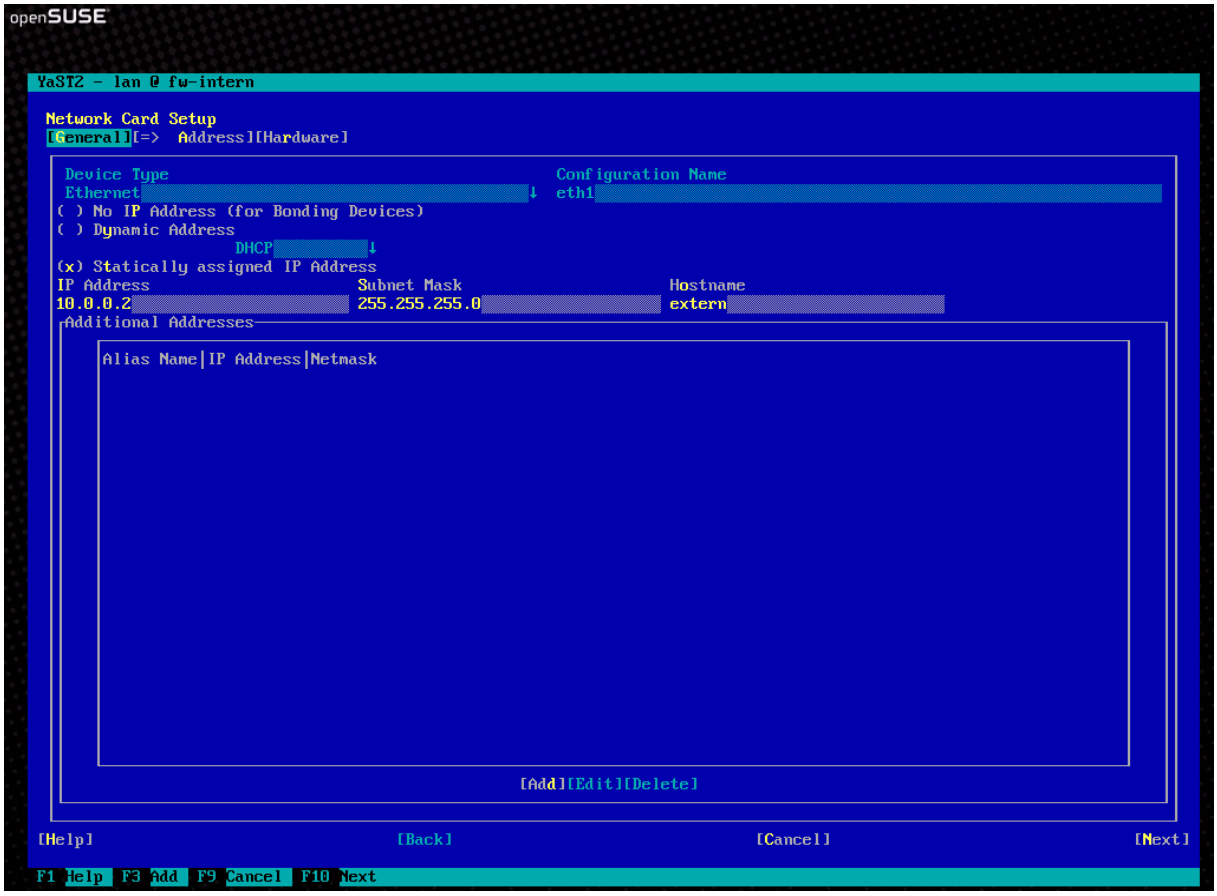
Mit den Pfeiltasten wird nach Network Devices > Network Settings navigiert:



Nachdem die ENTER-Taste gedrückt wurde erhält man das folgende Menü:



Hier können nun die Einstellungen für die Netzwerkkarten vorgenommen werden. Dazu wählt man die entsprechende Karte aus und drückt die Taste F4. Es folgt ein weiteres Konfigurationsfenster in dem folgende Einstellungen getätigt werden:



Für jede Netzwerkschnittstelle in den Firewalls müssen diese Konfiguration vorgenommen werden. Die richtigen Einstellungen sind dem Kapitel xxx zu entnehmen.
Nachdem die Einstellungen vorgenommen wurden, muss zunächst das Netzwerksubsystem neu gestartet werden. Dazu führt man auf der Konsole folgenden Befehl aus:

```
fwproto:~# rcnetwork restart
```

oder man startet das gesamte System neu:

```
fwproto:~# reboot
```

Mit dem Befehl:

```
fwproto:~# ifconfig
```

lässt sich überprüfen, ob die Einstellungen für die Netzwerkkarten ordnungsgemäß übernommen wurden.
Es sollte folgende Ausgabe für die Firewall „Extern“ erscheinen:

```
fw-extern:~ # ifconfig
eth0      Link encap:Ethernet  HWaddr 00:0C:29:9D:22:5A
          inet addr:10.0.0.1  Bcast:10.0.0.255  Mask:255.255.255.0
          inet6 addr: fe80::20c:29ff:fe9d:225a/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:0 errors:0 dropped:0 overruns:0 frame:0
          TX packets:16 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:0 (0.0 b)  TX bytes:2477 (2.4 Kb)
          Interrupt:18 Base address:0x2024

eth1      Link encap:Ethernet  HWaddr 00:0C:29:9D:22:64
          inet addr:192.168.2.156  Bcast:192.168.2.255  Mask:255.255.255.0
          inet6 addr: fe80::20c:29ff:fe9d:2264/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:3805 errors:0 dropped:0 overruns:0 frame:0
          TX packets:46 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:683157 (667.1 Kb)  TX bytes:7191 (7.0 Kb)
          Interrupt:19 Base address:0x20a4

lo        Link encap:Local Loopback
          inet addr:127.0.0.1  Mask:255.0.0.0
          inet6 addr: ::1/128 Scope:Host
          UP LOOPBACK RUNNING  MTU:16436  Metric:1
          RX packets:0 errors:0 dropped:0 overruns:0 frame:0
          TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:0 (0.0 b)  TX bytes:0 (0.0 b)
```

Es sollte folgende Ausgabe für die Firewall „Intern“ erscheinen:

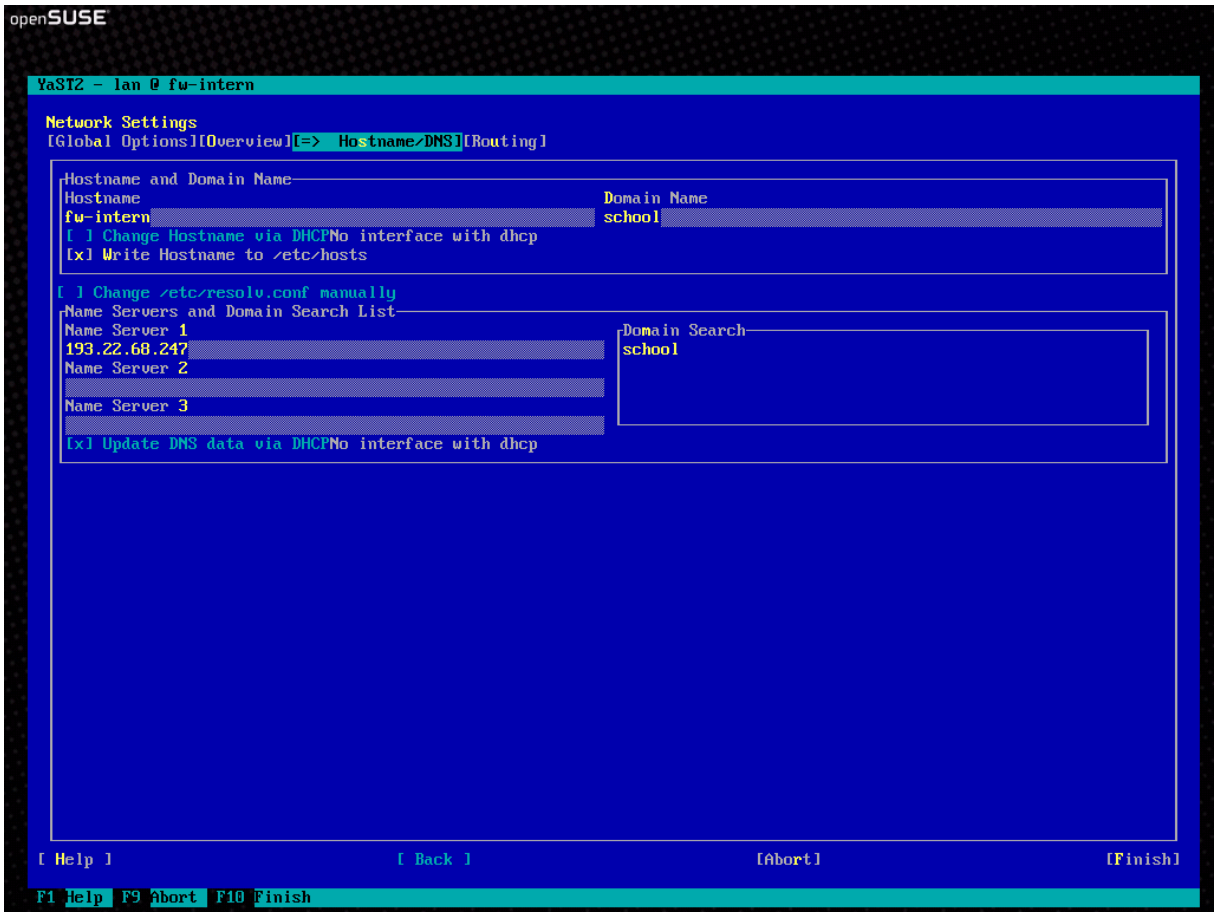
```
fw-intern:~ # ifconfig
eth0      Link encap:Ethernet  HWaddr 00:0C:29:2B:2C:58
          inet addr:192.168.10.1  Bcast:192.168.10.255  Mask:255.255.255.0
          inet6 addr: fe80::20c:29ff:fe2b:2c58/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:0 errors:0 dropped:0 overruns:0 frame:0
          TX packets:14 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:0 (0.0 b)  TX bytes:1593 (1.5 Kb)
          Interrupt:18 Base address:0x2024

eth1      Link encap:Ethernet  HWaddr 00:0C:29:2B:2C:62
          inet addr:10.0.0.2  Bcast:10.0.0.255  Mask:255.255.255.0
          inet6 addr: fe80::20c:29ff:fe2b:2c62/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:10 errors:0 dropped:0 overruns:0 frame:0
          TX packets:14 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:2015 (1.9 Kb)  TX bytes:1577 (1.5 Kb)
          Interrupt:19 Base address:0x20a4

lo        Link encap:Local Loopback
          inet addr:127.0.0.1  Mask:255.0.0.0
          inet6 addr: ::1/128 Scope:Host
          UP LOOPBACK RUNNING  MTU:16436  Metric:1
          RX packets:0 errors:0 dropped:0 overruns:0 frame:0
          TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:0 (0.0 b)  TX bytes:0 (0.0 b)
```

16.3. Konfiguration – Hostname und DNS

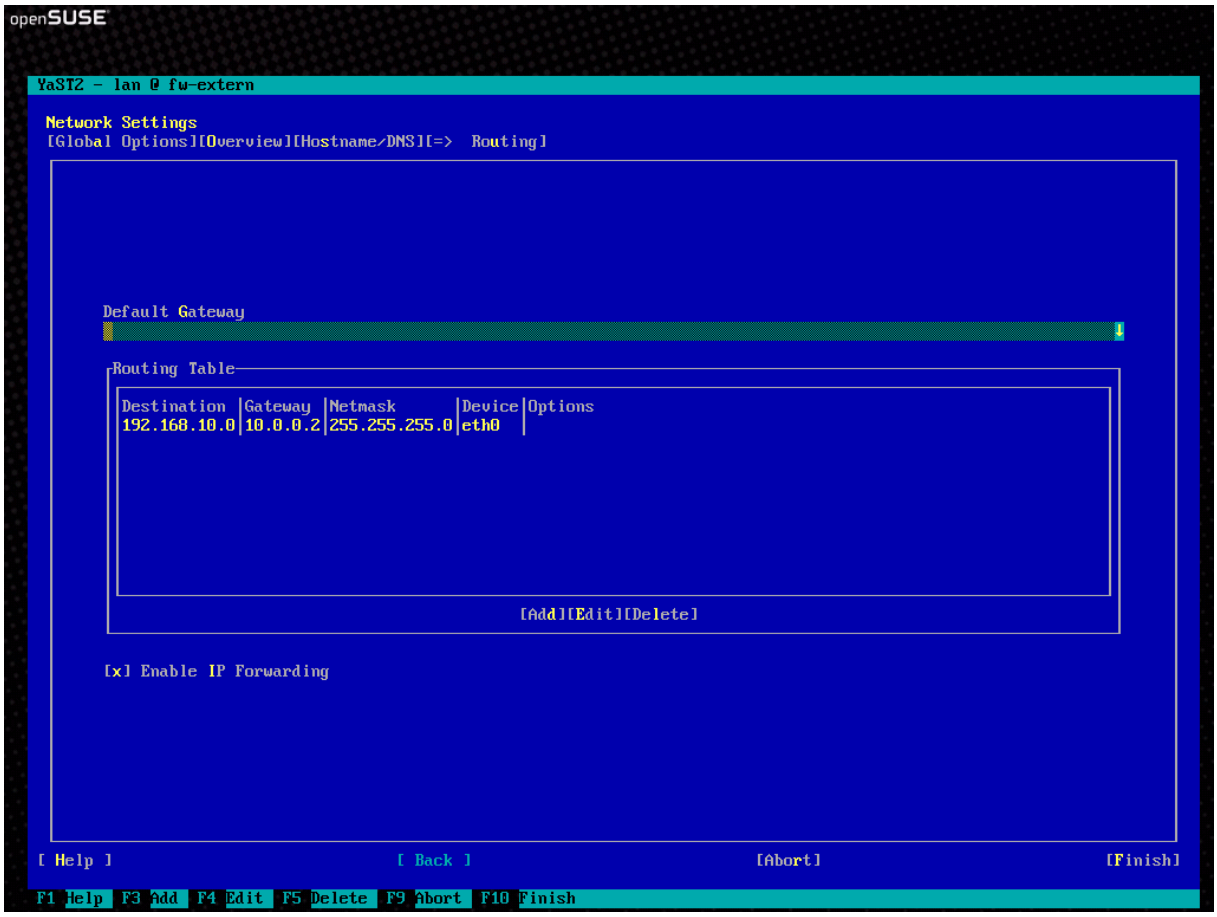
Um die einzelnen Server und Client-Systeme im Netzwerk identifizieren zu können, ist es möglich jedem System einen Namen zuzuordnen. Diese Einstellung kann unter YAST > Network Devices > Network Settings in der Registerkarte Hostname/DNS vorgenommen werden. Weiterhin ist es möglich dem System eine logische Zuordnung zu einer sogenannten Domain (Namensraum) zu geben. Die Kombination aus Hostname und Domain identifiziert den Rechner im Netzwerk.
Damit die Firewallsysteme DNS-Namen auflösen können ist es notwendig den DNS-Server des Netzwerkes einzutragen. Für die Firewall „Intern“ könnte solch ein Eintrag folgendermaßen aussehen:



Der Name- bzw. DNS-Server muss an die Betriebsumgebung angepasst werden.

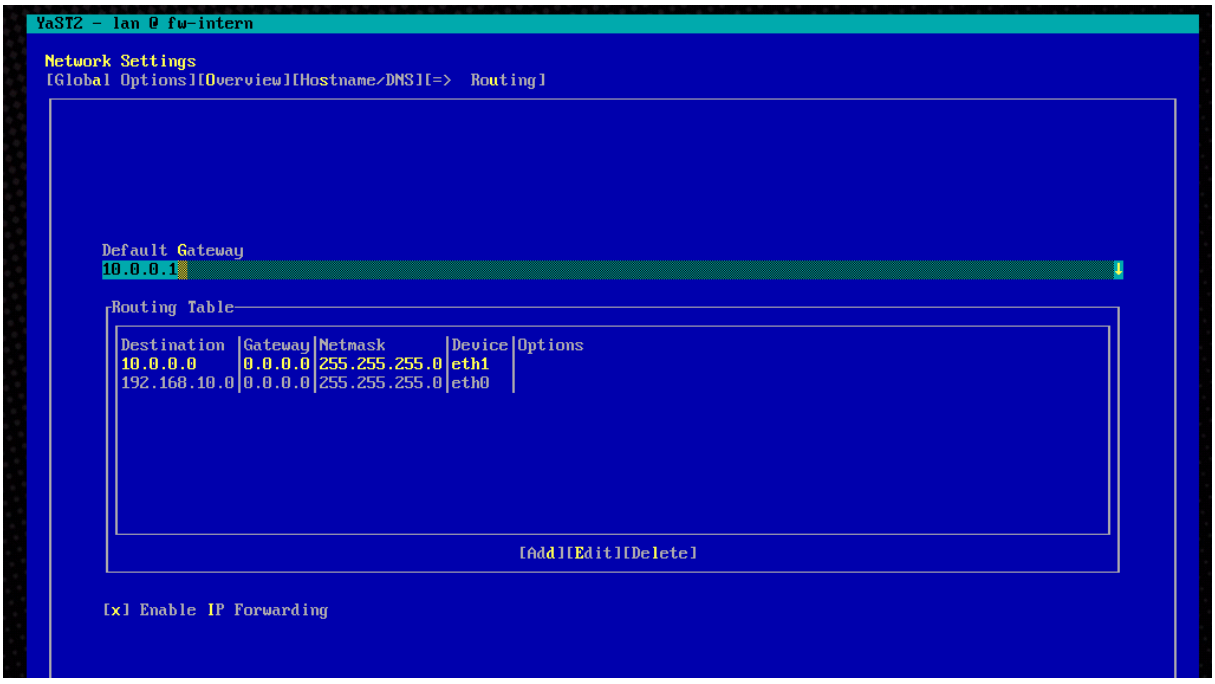
16.4. Konfiguration – Routing

Allein die Einrichtung der Netzwerkkarte genügt nicht, um die Firewall einsatzfähig zu machen. Da die Firewalls Datenpakete innerhalb der Netzwerke übertragen und weiterleiten müssen, werden sogenannte Routen (Wegfindungen) benötigt. Dabei bestimmt das Routing den gesamten Weg eines Nachrichtenstroms (der Datenpakete) durch das Netzwerk. Die Firewall führt also entsprechende Routinglisten, die es ermöglichen, dass die Datenpakete zu ihrem Ziel gelangen.
Routinglisten können mit Hilfe von YAST eingerichtet werden. Dies erfolgt unter Network Devices > Network Settings in der Registerkarte Routing:
Mit „Add“ können Routingeinträge hinzugefügt werden (hier Firewall „Extern“):



Der Default-Gateway wird je nach Netzwerkarchitektur eingetragen (siehe Kapitel xxx).

Die Routingeinträge für die Firewall „Intern“ sehen folgendermaßen aus:



Als Standard-Gateway wird hier die IP-Adresse der Firewall „Extern“ eingetragen. Das Default-Gateway entspricht der Firewall „Außen“.
Auch nach dieser Konfiguration ist ein Neustart der Netzwerkdienste unbedingt notwendig!
Dazu führt man auf der Konsole folgenden Befehl aus:

```
fwproto:~# rcnetwork restart
```

oder man startet das gesamte System neu:

```
fwproto:~# reboot
```

16.5. Start-Stopskript – „Firewall-Innen“

```
#!/bin/sh
### BEGIN INIT INFO
# Provides:             firewall intern
# Required-Start:               $syslog $remote_fs
# Should-Start:         $time ypbind
# Required-Stop:                $syslog $remote_fs
# Should-Stop:          $time ypbind smtp
# Default-Start:                3 5
# Default-Stop:         0 1 2 6
# Short-Description:    Enabling firewall settings
                        on iptables
# Description:          Enabling firewall settings
                        on iptables
### END INIT INFO
FW_BIN=/fw/fw_intern.up
FW_SHUTDOWN=/fw/fw_intern.down



test -x $FW_BIN || { echo "Start skript not available in /fw/ (fw_intern.up)";
                                if ["$1"="stop"]; then exit 0;
                                else exit 5; fi; }

test -x $FW_SHUTDOWN || { echo "Stop skript not available in /fw/ (fw_intern.down)";
                                if ["$1"="stop"]; then exit 0;
                                else exit 5; fi; }



. /etc/rc.status
rc_reset
case "$1" in

              start)
            echo "Setting FW-Rules intern"
```

```
            $FW_BIN
            rc_status –v ;;
          stop)
            echo -n "Shutdown firewall intern"
            $FW_SHUTDOWN
            rc_status –v ;;
          *)
            echo "Usage: $0 {start|stop}"
            exit 1 ;;
esac
rc_exit
```

16.6. Start-Stopskript – „Firewall-Außen“

```
#!/bin/sh
### BEGIN INIT INFO
# Provides:             firewall extern
# Required-Start:               $syslog $remote_fs
# Should-Start:         $time ypbind
# Required-Stop:                $syslog $remote_fs
# Should-Stop:          $time ypbind smtp
# Default-Start:                3 5
# Default-Stop:         0 1 2 6
# Short-Description:    Enabling firewall settings
                        on iptables
# Description:          Enabling firewall settings
                        on iptables
### END INIT INFO
FW_BIN=/fw/fw_extern.up
FW_SHUTDOWN=/fw/fw_extern.down




test -x $FW_BIN || { echo "Start skript not available in /fw/ (fw_extern.up)";
                                if ["$1"="stop"]; then exit 0;
                                else exit 5; fi; }

test -x $FW_SHUTDOWN || { echo "Stop skript not available in /fw/ (fw_extern.down)";
                                if ["$1"="stop"]; then exit 0;
                                else exit 5; fi; }




. /etc/rc.status
rc_reset
case "$1" in

            start)
```

```
            echo "Setting FW-Rules extern"
            $FW_BIN
            rc_status –v ;;
          stop)
            echo -n "Shutdown firewall extern"
            $FW_SHUTDOWN
            rc_status –v ;;
          *)
            echo "Usage: $0 {start|stop}"
            exit 1 ;;
esac
rc_exit
```

```
################################################################################
#   file:       fw_intern.down
#
#   date:       2009-05-17
#
#   author:     Oliver Enns - OE
#
#   brief:      Skript zum "stoppen" der Firewall. Nach Ausführung des Skripts, findet keine Filterung der Pakete mehr statt.
#               Es wird alles akzeptiert. Alle Benuterketten und Regeln werden gelöscht.
#               Dieses Skript wird auf der Firewall-Innen ausgeführt.
#
#   history:    2009-05-17      - OE        - erstellt
#
################################################################################

#!/bin/sh

###################################### begin ###################################################
#   comment:    Einstellungen für die Tabelle 'filter'
#
#   history:    2009-05-17      - OE        - #01       - erstellt

#   brief:      Alle Regeln in der Tabelle 'filter' löschen
#
#   last:       2009-05-17      - OE        - #01       - erstellt
iptables -t nat -F

#   brief:      Alle Benutzerketten in der Tabelle 'filter' löschen
#
#   last:       2009-05-17      - OE        - #01       - erstellt
iptables -X

#   brief:      Default-Policy für Kette INPUT in Tabelle 'filter' setzen
#
#   last:       2009-05-17      - OE        - #01       - erstellt
iptables -P INPUT ACCEPT

#   brief:      Default-Policy für Kette FORWARD itnTabelle 'filter' setzen
#
```

```
#   last:       2009-05-17      - OE        - #01       - erstellt
iptables -P FORWARD ACCEPT

#   brief:      Default-Policy für Kette OUTPUT in Tabelle 'filter' setzen
#
#   last:       2009-05-17      - OE        - #01       - erstellt
iptables -P OUTPUT ACCEPT
######################################## end ##############################################

######################################## begin #############################################
#   comment:    Einstellungen für die Tabelle 'nat'
#
#   history:    2009-05-17      - OE        - #01       - erstellt

#   brief:      All e Regeln in der Tabelle 'nat' löschen
#
#   last:       2009-05-17      - OE        - #01       - erstellt
iptables -t nat -F

#   brief:      Alle Benutzerketten in der Tabelle 'nat' löschen
#
#   last:       2009-05-17      - OE        - #01       - erstellt
iptables -t nat -X

#   brief:      Default-Policy für Kette PREROUTING in Tabelle 'nat' setzen
#
#   last:       2009-05-17      - OE        - #01       - erstellt
iptables -t nat -P PREROUTING ACCEPT

#   brief:      Default-Policy für Kette OUTPUT in Tabelle 'nat' setzen
#
#   last:       2009-05-17      - OE        - #01       - erstellt
iptables -t nat -P OUTPUT ACCEPT

#   brief:      Default-Policy für Kette POSTROUTING in Tabelle 'nat' setzen
#
#   last:       2009-05-17      - OE        - #01       - erstellt
iptables -t nat -P POSTROUTING ACCEPT
######################################## end ##############################################

######################################## begin #############################################
```

```
#   comment:    Einstellungen für die Tabelle 'mangle'
#
#   history:    2009-05-17      - OE        - #01       - erstellt

#   brief:      All e Regeln in der Tabelle 'mangle' löschen
#
#   last:       2009-05-17      - OE        - #01       - erstellt
iptables -t mangle -F

#   brief:      Alle Benutzerketten in der Tabelle 'mangle' löschen
#
#   last:       2009-05-17      - OE        - #01       - erstellt
iptables -t mangle -X

#   brief:      Default-Policy für Kette PREROUTING in Tabelle 'mangle' setzen
#
#   last:       2009-05-17      - OE        - #01       - erstellt
iptables -t mangle -P PREROUTING ACCEPT

#   brief:      Default-Policy für Kette INPUT in Tabelle 'mangle' setzen
#
#   last:       2009-05-17      - OE        - #01       - erstellt
iptables -t mangle -P INPUT ACCEPT

#   brief:      Default-Policy für Kette FORWARD in Tabelle 'mangle' setzen
#
#   last:       2009-05-17      - OE        - #01       - erstellt
iptables -t mangle -P FORWARD ACCEPT

#   brief:      Default-Policy für Kette OUTPUT in Tabelle 'mangle' setzen
#
#   last:       2009-05-17      - OE        - #01       - erstellt
iptables -t mangle -P OUTPUT ACCEPT

#   brief:      Default-Policy für Kette POSTROUTING in Tabelle 'mangle' setzen
#
#   last:       2009-05-17      - OE        - #01       - erstellt
iptables -t mangle -P POSTROUTING ACCEPT
######################################## end ##############################################

#################################################################################
```

```
#   file:       fw_intern.up
#
#   date:       2009-05-22
#
#   author:     Oliver Enns - OE
#
#   brief:      Skript zur Erstelleung der Filter-Regeln und Benutzerketten. Dient dem Schutz des Client-Netzes.
#               Dieses Skript wird auf der Firewall-Innen ausgeführt.
#
#   history:    2009-05-22      - OE        - erstellt
#
################################################################################

#!/bin/sh


#   brief:      Vor dem Start der Firewall wird IP-Forward deaktiviert, da die Firewall am Anfang alles durchlässt.
#
#   last:       2009-05-22      - OE        - #01       - erstellt
echo 0 > /proc/sys/net/ipv4/ip_forward

###################################### begin ############################################
#   comment:    Es werden die in der DMZ-Netz verfügbaren Dienste und deren IP-Adresse und Port festgelegt.
#               Die Netzwerkkarten-Einstellung der Firewall werden festgelt.
#
#   history:    2009-05-22      -OE     -#01        -erstellt

#   brief:      DHCP-Server in DMZ.
#
#   last:       2009-05-22      - OE        - #01       - erstellt
DHCPSERVER='10.0.0.103'

#   brief:      DHCP-Server-Port
#
#   last:       2009-05-22      - OE        - #01       - erstellt
DHCPSERVER_PORT='67'

#   brief:      DHCP-Client-Port
#
#   last:       2009-05-22      - OE        - #01       - erstellt
DHCPCLIENT_PORT='68'
```

```
#   brief:      DNS-Server in DMZ.
#
#   last:       2009-05-22      - OE        - #01       - erstellt
DNSSERVER='10.0.0.103'

#   brief:      DNS-Server-Port.
#
#   last:       2009-05-22      - OE        - #01       - erstellt
DNSSERVER_PORT='53'

#   brief:      Proxy-Server in DMZ.
#
#   last:       2009-05-22      - OE        - #01       - erstellt
PROXY='10.0.0.106'

#   brief:      Proxy-Server in DMZ.
#
#   last:       2009-05-22      - OE        - #01       - erstellt
PROXY_PORT='3128'

#   brief:      Mail-Server in DMZ
#
#   last:       2009-05-22      - OE        - #01       - erstellt
MAILSERVER='10.0.0.101'

#   brief:      Mail-Server-Port für den Versand der Mails.
#
#   last:       2009-05-22      - OE        - #01       - erstellt
MAILSERVER_PORT_SMTP='25'

#   brief:      Mail-Server-Port zum Abholen der Mails
#
#   last:       2009-05-22      - OE        - #01       - erstellt
MAILSERVER_PORT_POP='110'

#   brief:      Mail-Server-Port zum Abholen der Mails
#
#   last:       2009-05-22      - OE        - #01       - erstellt
MAILSERVER_PORT_IMAP='143'
```

```
#   brief:      WebServer in der DMZ
#
#   last:       2009-05-22      - OE        - #01       - erstellt
WEBSERVER="10.0.0.107"

#   brief:      Webserver-Port in DMZ
#
#   last:       2009-05-22      - OE        - #01       - erstellt
WEBSERVER_PORT='80'

#   brief:      OPSI-Server in DMZ
#
#   last:       2009-05-22      - OE        - #01       - erstellt
OPSISERVER="10.0.0.102/32"

#   brief:      OPSI-Server-Port
#
#   last:       2009-05-22      - OE        - #01       - erstellt
OPSISERVER_PORT='11111 '

#   brief:      Samba-Server in DMZ
#
#   last:       2009-05-22      - OE        - #01       - erstellt
SAMBASERVER="10.0.0.104/32"

#   brief:      Samba-Server-Port
#
#   last:       2009-05-22      - OE        - #01       - erstellt
#SAMBASERVER_PORT=

#   brief:      LDAP-Server in DMZ
#
#   last:       2009-05-22      - OE        - #01       - erstellt
LDAPSERVER="10.0.0.104"

#   brief:      LDAP-Server-Port
#
#   last:       2009-05-22      - OE        - #01       - erstellt
LDAPSERVER_PORT='389'

#   brief:      Externe Netzwerkschnittstelle der Firewall.
```

```
#
#   last:       2009-05-22      - OE        - #01       - erstellt
IF_EXT_IP='10.0.0.2'

#   brief:      Name  der  externen Netzwerkschnittstellen Firewall
#
#   last:       2009-05-22      - OE        - #01       - erstellt
IF_EXT='eth1'

#   brief:      Interne Netzwerkschnittstelle der Firewall.
#
#   last:       2009-05-22      - OE        - #01       - erstellt
IF_INT_IP='192.168.10.1'

#   brief:      Name  der  internen Netzwerkschnittstellen Firewall
#
#   last:       2009-05-22      - OE        - #01       - erstellt
IF_INT='eth0'

#   brief:      Adressbereich    des externen Netzes (DMZ).
#
#   last:       2009-05-22      - OE        - #01       - erstellt
EXT_NET='10.0.0.0/24'

#   brief:      Adressbereich des internen Netzes (Client).
#
#   last:       2009-05-22      - OE        - #01       - erstellt
INT_NET='192.168.10.0/24'

#   brief:      Definition für beliebiges Netz
#
#   last:       2009-05-22      - OE        - #01       - erstellt
ALL_NET='0.0.0.0'

#   brief:      Der PC, von dem ein SSH Zugriff auf dieses System gewährt wird.
#
#   last:       2009-05-22      - OE        - #01       - erstellt
ADMIN_PC='192.168.10.9/24'

#   brief:      Die MAC_Adresse des Admin-PCs
#
```

```
#   last:       2009-05-22      - OE        - #01       - erstellt
ADMIN_PC_MAC=''
######################################## end ################################################

#   brief:      Das Skript zum Zurücksetzen aller Regeln und Ketteb wird ausgeführt.
#
#   last:       2009-05-22      - OE        - #01       - erstellt
/fw/fw_intern.down

######################################## begin ###############################################
#   comment:    Es werden die Default-Policies für die Tabelle 'filter' gesetzt.
#
#   history:    2009-05-22      - OE        - #01       - erstellt

#   brief:      Es wird in der Kette INPUT alles geblockt.
#
#   last:       2009-05-22      - OE        - #01       - erstellt
iptables -P INPUT DROP

#   brief:      Es wird in der FORWARD Kette  alles geblockt.
#
#   last:       2009-05-22      - OE        - #01       - erstellt
iptables -P FORWARD DROP

#   brief:      Es wird in der Kette OUTPUT alles geblockt.
#
#   last:       2009-05-22      - OE        - #01       - erstellt
iptables -P OUTPUT DROP
######################################## end ################################################

#   brief:      IP-Forward kann nun wieder aktiviert werden, um eine Weiterleitung der Pakete zu ermöglichen.
#
#   last:       2009-05-22      - OE        - #01       - erstellt
echo 1 > /proc/sys/net/ipv4/ip_forward

######################################## begin ###############################################
#   comment:        Erstellung von Benutzerketten in der Tabelle 'filter'.
#
#   history:    2009-05-22      - OE        - #01       - erstellt

#   brief:      Hier werden Beutzerketten in der Tabelle 'filter' erstellt.
```

```
#
#   last:       2009-05-22      - OE        - #01       - erstellt

#   brief:      Kette, für die ICMP-Pakte
#
#   last:       2009-05-22      -OE     -#01        -erstellt
iptables -N icmp

#   brief:      Kette, für die DNS-Eingangs-Pakte
#
#   last:           009-05-23       -OE     -#01        -erstellt
iptables -N dns_in

#   brief:      Kette, für die DNS-Ausgangs-Pakte
#
#   last:       2009-05-23      -OE     -#01        -erstellt
iptables -N dns_out

#   brief:      Kette, für die SMTP-Eingangs-Pakte
#
#   last:       2009-05-23      -OE     -#01        -erstellt
iptables -N mail_smtp_in

#   brief:      Kette, für die SMTP-Ausgangs-Pakte
#
#   last:       2009-05-23      -OE     -#01        -erstellt
iptables -N mail_smtp_out

#   brief:      Kette, für die POP-Eingangs-Pakte
#
#   last:       2009-05-23      -OE     -#01        -erstellt
iptables -N mail_pop_in

#   brief:      Kette, für die POP-Ausgangs-Pakte
#
#   last:       2009-05-23      -OE     -#01        -erstellt
iptables -N mail_pop_out

#   brief:      Kette, für die IMAP-Eingangs-Pakte
#
#   last:       2009-05-23      -OE     -#01        -erstellt
```

```
iptables -N mail_imap_in

#   brief:      Kette, für die IMAP-Ausgangs-Pakte
#
#   last:       2009-05-23      -OE     -#01        -erstellt
iptables -N mail_imap_out

#   brief:      Kette, zum RFC-Konformen abweisen der Pakete, falls keine Regel in einer Kette zutreffen sollte
#
#   last:       2009-05-23      -OE     -#01        -erstellt
#Reject-Rule
iptables -N reject_packets

#   brief:      Kette, zum Überprüfen von ungültigen TCP-Flag Kombinationen
#
#   last:       2009-05-23      -OE     -#01        -erstellt
iptables -N tcp-flags_check
######################################## end ##############################################

######################################## begin #############################################
#   comment:    Deklaration von Benutzerketten in der Tabelle 'filter'.
#
#   history:    2009-05-26      - OE        - #02   - Mailregeln hinzugefügt

#   brief:      Es werden alle ICMP-Pakete akzeptiert.
#
#   last:       2009-05-24      -OE     -#01        -erstellt
iptables -A icmp -p icmp -j ACCEPT

#   brief:      Es werden die DNS-Pakete vom und zu dem DNS-Server akzeptiert.
#
#   last:       2009-05-24      -OE     -#01        -erstellt
iptables -A dns_in -p udp -s $DNSSERVER --sport $DNSSERVER_PORT -j ACCEPT
iptables -A dns_out -p udp -d $DNSSERVER --dport $DNSSERVER_PORT -j ACCEPT

#   brief:      Überprüft das TCP-Paket auf gültige TCP-Flag Kombination.
#
#   last:       2009-05-24      -OE     -#01        -erstellt
iptables -A tcp-flags_check -p tcp --tcp-flags ALL ALL -j reject_packets
iptables -A tcp-flags_check -p tcp --tcp-flags ALL NONE -j reject_packets
iptables -A tcp-flags_check -p tcp --tcp-flags ALL FIN,PSH,URG -j reject_packets
```

```
iptables -A tcp-flags_check -p tcp --tcp-flags ALL SYN,FIN,PSH,URG -j reject_packets
iptables -A tcp-flags_check -p tcp --tcp-flags ALL SYN,RST,ACK,FIN,URG -j reject_packets
iptables -A tcp-flags_check -p tcp --tcp-flags SYN,FIN SYN,FIN -j reject_packets
iptables -A tcp-flags_check -p tcp --tcp-flags SYN,RST SYN,RST -j reject_packets
iptables -A tcp-flags_check -p tcp --tcp-flags FIN,RST FIN,RST -j reject_packets
iptables -A tcp-flags_check -p tcp --tcp-flags ACK,FIN FIN -j reject_packets
iptables -A tcp-flags_check -p tcp --tcp-flags ACK,URG URG -j reject_packets
iptables -A tcp-flags_check -p tcp --tcp-flags ACK,PSH PSH -j reject_packets
iptables -A tcp-flags_check -p tcp --tcp-flags ACK,PSH PSH -j reject_packets

#   brief:      Dient der Abweisung eines Pakets. Es erfolgt eine Benachrichtigung an den Absender. Vor dem Abweisen,
#               wird das Paket protkolliert.
#
#   last:       2009-05-24      -OE     -#01        -erstellt
iptables -A reject_packets -j LOG
iptables -A reject_packets -p tcp -j REJECT --reject-with tcp-reset
iptables -A reject_packets -p ALL -j REJECT --reject-with icmp-port-unreachable

#   brief:      Es werden die Mail-Pakete vom und zu dem Mail-Server akzeptiert.
#
#   last:       2009-05-26      -OE     -#01        -erstellt
iptables -A mail_smtp_in -p tcp --sport $MAILSERVER_PORT_SMTP -s $MAILSERVER -m state --state ESTABLISHED -j ACCEPT
iptables -A mail_smtp_out -p tcp --dport $MAILSERVER_PORT_SMTP -d $MAILSERVER -m state --state NEW,ESTABLISHED -j ACCEPT
iptables -A mail_pop_in -p tcp --sport $MAILSERVER_PORT_POP -s $MAILSERVER -m state --state ESTABLISHED -j ACCEPT
iptables -A mail_pop_out -p tcp --dport $MAILSERVER_PORT_POP -d $MAILSERVER -m state --state NEW,ESTABLISHED -j ACCEPT
iptables -A mail_imap_in -p tcp --sport $MAILSERVER_PORT_IMAP -s $MAILSERVER -m state --state ESTABLISHED -j ACCEPT
iptables -A mail_imap_out -p tcp --dport $MAILSERVER_PORT_IMAP -d $MAILSERVER -m state --state NEW,ESTABLISHED -j ACCEPT
##################################### end ##############################################

##################################### begin ############################################
#   comment:    Regeldefinition in der Tabelle 'filter', Kette INPUT
#
#   history:    2009-05-22      - OE        - #01   - erstellt

#   brief:      SSH-Pakete werden reingelassen.
#
#   last:       2009-05-24      -OE     -#01        -erstellt
iptables -A INPUT -i $IF_EXT -p tcp --dport 22 -m state --state NEW,ESTABLISHED -j ACCEPT

#   brief:      Spoofing verhinden
#
```

```
#   last:       2009-05-24      -OE     -#01        -erstellt
iptables -A INPUT -s $INT_NET -j DROP

#   brief:      ICMP-Pakete beim Eintreffen an das entsprechende Ziel weiterleiten.
#
#   last:       2009-05-24      -OE     -#01        -erstellt
iptables -A INPUT -j icmp

#   brief:      DNS-Pakete beim Eintreffen an das entsprechende Ziel weiterleiten.
#
#   last:       2009-05-24      -OE     -#01        -erstellt
iptables -A INPUT -i $IF_EXT -j dns_in

#   brief:      DHCP-Pakete beim Eintreffen an das entsprechende Ziel weiterleiten.
#
#   last:       2009-05-24      -OE     -#01        -erstellt
iptables -A INPUT -i $IF_INT -p udp -s $ALL_NET --sport $DHCPCLIENT_PORT -j ACCEPT

#   brief:      Letzte Regel in der Kette INPUT. Pakete werden protokolliert und abgewiesen.
#
#   last:       2009-05-24      -OE     -#01        -erstellt
iptables -A INPUT -j reject_packets
######################################## end ##############################################

######################################## begin #############################################
#   comment:    Regeldefinition in der Tabelle 'filter', Kette FORWARD
#
#   history:    2009-05-26      - OE        - #01       - erstellt

#   brief:      ICMP-Pakete beim Eintreffen an das entsprechende Ziel weiterleiten.
#
#   last:       2009-05-26      -OE     -#01        -erstellt
iptables -A FORWARD -j icmp

#   brief:      HTTP-Pakete an den Proxy weiterleiten
#
#   last:       2009-05-26      -OE     -#01        -erstellt
iptables -A FORWARD -i $IF_INT -o $IF_EXT -s $INT_NET -d $PROXY -p tcp -m state --state NEW,ESTABLISHED -j ACCEPT
iptables -A FORWARD -i $IF_EXT -o $IF_INT -d $INT_NET -s $PROXY -p tcp -m state --state ESTABLISHED -j ACCEPT

#   brief:      DNS- IN und OUT-Pakete bearbeiten
```

```
#
#   last:       2009-05-26      -OE     -#01        -erstellt
iptables -A FORWARD -i $IF_EXT -o $IF_INT -j dns_in
iptables -A FORWARD -i $IF_INT -o $IF_EXT -j dns_out


#   brief:      EMail- IN und OUT-Pakete weiterleiten
#
#   last:       2009-05-26      -OE     -#01        -erstellt
iptables -A FORWARD -i $IF_EXT -o $IF_INT -j mail_smtp_in
iptables -A FORWARD -i $IF_INT -o $IF_EXT -j mail_smtp_out
iptables -A FORWARD -i $IF_EXT -o $IF_INT -j mail_pop_in
iptables -A FORWARD -i $IF_INT -o $IF_EXT -j mail_pop_out
iptables -A FORWARD -i $IF_EXT -o $IF_INT -j mail_imap_in
iptables -A FORWARD -i $IF_INT -o $IF_EXT -j mail_imap_out

#   brief:      Letzte Regel in der Kette FORWARD. Pakete werden protokolliert und abgewiesen.
#
#   last:       2009-05-26      -OE     -#01        -erstellt
iptables -A FORWARD -j reject_packets
######################################## end ##############################################

######################################## begin ############################################
#   comment:        Regeldefinition in der Tabelle 'filter',  Kette OUTPUT
#
#   history:    2009-05-26      - OE        - #01       - erstellt

#   brief:      ICMP-Pakete werden rausgelassen
#
#   last:       2009-05-26      -OE     -#01        -erstellt
iptables -A OUTPUT -j icmp

#   brief:      SSH-Pakete rasuslassen
#
#   last:       2009-05-24      -OE     -#01        -erstellt
iptables -A OUTPUT -o $IF_EXT -p tcp --sport 22 -m state --state ESTABLISHED -j ACCEPT

#   brief:      DNS-Pakete rasuslassen
#
#   last:       2009-05-24      -OE     -#01        -erstellt
iptables -A OUTPUT -o $IF_EXT -j dns_out
```

```
#   brief:      DHCP-Pakete rasuslassen
#
#   last:       2009-05-24      -OE     -#01        -erstellt
iptables -A OUTPUT -o $IF_EXT -p udp -d $DHCPSERVER --dport 67 -j ACCEPT

#   brief:      Letzte Regel in der Kette OUTPUT. Pakete werden protokolliert und abgewiesen.
#
#   last:       2009-05-26      -OE     -#01        -erstellt
iptables -A OUTPUT -j reject_packets

######################################## end #############################################

######################################## begin ###########################################
#   comment:        Regeldefinition in der Tabelle 'nat', Kette PREROUTING
#
#   history:    2009-05-27      - OE        - #01       - erstellt

#   brief:      HTTP-Pakete an den Proxy weiterleiten
#
#   last:       2009-05-27      -OE     -#01        -erstellt
iptables -t nat -A PREROUTING -i $IF_INT -s $INT_NET -p tcp --dport 80 -m state --state NEW -j DNAT --to $PROXY:$PROXY_PORT

#   brief:      Mail-Pakete an den Mailserver weiterleiten
#
#   last:       2009-05-27      -OE     -#01        -erstellt
iptables -t nat -A PREROUTING -i $IF_INT -s $INT_NET -p tcp --dport $MAILSERVER_PORT_SMTP -m state --state NEW -j DNAT --to
$MAILSERVER:$MAILSERVER_PORT_SMTP
iptables -t nat -A PREROUTING -i $IF_INT -s $INT_NET -p tcp --dport $MAILSERVER_PORT_POP -m state --state NEW -j DNAT --to
$MAILSERVER:$MAILSERVER_PORT_POP
iptables -t nat -A PREROUTING -i $IF_INT -s $INT_NET -p tcp --dport $MAILSERVER_PORT_IMAP -m state --state NEW -j DNAT --to
$MAILSERVER:$MAILSERVER_PORT_IMAP
######################################## end #############################################
```

```
################################################################################
#   file:       fw_extern.down
#
#   date:       2009-05-17
#
#   author:     Oliver Enns - OE
#
#   brief:      Skript zum "stoppen" der Firewall. Nach Ausführung des Skripts, findet
#               keine Filterung der Pakete mehr statt. Es wird alles
#               akzeptiert. Alle Benuterketten und Regeln werden gelöscht.
#               Dieses Skript wird auf der Firewall-Außen ausgeführt.
#
#   history:    2009-05-17      - OE        - erstellt
#
################################################################################

#!/bin/sh

######################################## begin ################################################
#   comment:    Einstellungen für die Tabelle 'filter'
#
#   history:    2009-05-17      - OE        - #01       - erstellt

#   brief:      Alle Regeln in der Tabelle 'filter' löschen
#
#   last:       2009-05-17      - OE        - #01       - erstellt
iptables -t nat -F

#   brief:      Alle Benutzerketten in der Tabelle 'filter' löschen
#
#   last:       2009-05-17      - OE        - #01       - erstellt
iptables -X

#   brief:      Default-Policy für Kette INPUT in Tabelle 'filter' setzen
#
#   last:       2009-05-17      - OE        - #01       - erstellt
iptables -P INPUT ACCEPT

#   brief:      Default-Policy für Kette FORWARD itnTabelle 'filter' setzen
```

```
#
#   last:       2009-05-17      - OE        - #01       - erstellt
iptables -P FORWARD ACCEPT

#   brief:      Default-Policy für Kette OUTPUT in Tabelle 'filter' setzen
#
#   last:       2009-05-17      - OE        - #01       - erstellt
iptables -P OUTPUT ACCEPT
###################################### end ##############################################

###################################### begin ############################################
#   comment:    Einstellungen für die Tabelle 'nat'
#
#   history:    2009-05-17      - OE        - #01       - erstellt

#   brief:      Alle Regeln in der Tabelle 'nat' löschen
#
#   last:       2009-05-17      - OE        - #01       - erstellt
iptables -t nat -F

#   brief:      Alle Benutzerketten in der Tabelle 'nat' löschen
#
#   last:       2009-05-17      - OE        - #01       - erstellt
iptables -t nat -X

#   brief:      Default-Policy für Kette PREROUTING in Tabelle 'nat' setzen
#
#   last:       2009-05-17      - OE        - #01       - erstellt
iptables -t nat -P PREROUTING ACCEPT

#   brief:      Default-Policy für Kette OUTPUT in Tabelle 'nat' setzen
#
#   last:       2009-05-17      - OE        - #01       - erstellt
iptables -t nat -P OUTPUT ACCEPT

#   brief:      Default-Policy für Kette POSTROUTING in Tabelle 'nat' setzen
#
#   last:       2009-05-17      - OE        - #01       - erstellt
iptables -t nat -P POSTROUTING ACCEPT
###################################### end ##############################################
```

```
######################################## begin ###############################################
#   comment:    Einstellungen für die Tabelle 'mangle'
#
#   history:    2009-05-17      - OE        - #01       - erstellt

#   brief:      Alle Regeln in der Tabelle 'mangle' löschen
#
#   last:       2009-05-17      - OE        - #01       - erstellt
iptables -t mangle -F

#   brief:      Alle Benutzerketten in der Tabelle 'mangle' löschen
#
#   last:       2009-05-17      - OE        - #01       - erstellt
iptables -t mangle -X

#   brief:      Default-Policy für Kette PREROUTING in Tabelle 'mangle' setzen
#
#   last:       2009-05-17      - OE        - #01       - erstellt
iptables -t mangle -P PREROUTING ACCEPT

#   brief:      Default-Policy für Kette INPUT in Tabelle 'mangle' setzen
#
#   last:       2009-05-17      - OE        - #01       - erstellt
iptables -t mangle -P INPUT ACCEPT

#   brief:      Default-Policy für Kette FORWARD in Tabelle 'mangle' setzen
#
#   last:       2009-05-17      - OE        - #01       - erstellt
iptables -t mangle -P FORWARD ACCEPT

#   brief:      Default-Policy für Kette OUTPUT in Tabelle 'mangle' setzen
#
#   last:       2009-05-17      - OE        - #01       - erstellt
iptables -t mangle -P OUTPUT ACCEPT

#   brief:      Default-Policy für Kette POSTROUTING in Tabelle 'mangle' setzen
#
#   last:       2009-05-17      - OE        - #01       - erstellt
iptables -t mangle -P POSTROUTING ACCEPT
######################################## end #################################################
```

```
#################################################################################################
#   file:       fw_extern.up
#
#   date:       2009-05-30
#
#   author:     Oliver Enns - OE
#
#   brief:      Skript zur Erstelleung der Filter-Regeln und Benutzerketten. Dient dem Schutz des DMZ-Netzes.
#               Dieses Skript wird auf der Firewall-Außen ausgeführt.
#
#   history:    2009-05-30  - OE        - erstellt
#
#################################################################################################

#!/bin/sh

#   brief:      Vor dem Start der Firewall wird IP-Forward deaktiviert, da die Firewall am Anfang alles durchlässt.
#
#   last:       2009-05-30      - OE        - #01       - erstellt
echo 0 > /proc/sys/net/ipv4/ip_forward


######################################## begin ###############################################
#   comment:    Es werden die in der DMZ-Netz verfügbaren Dienste und deren IP-Adresse und Port festgelegt.
#               Die Netzwerkkarten-Einstellung der Firewall werden festgelt.
#
#   history:    2009-05-30      -OE         -#01        -erstellt

#   brief:      DHCP-Server in DMZ.
#
#   last:       2009-05-30      - OE        - #01       - erstellt
DHCPSERVER='10.0.0.103'

#   brief:      DHCP-Server-Port
#
#   last:       2009-05-30      - OE        - #01       - erstellt
DHCPSERVER_PORT='67'

#   brief:      DHCP-Client-Port
#
#   last:       2009-05-30      - OE        - #01       - erstellt
```

```
DHCPCLIENT_PORT='68'

#   brief:      DNS-Server in DMZ.
#
#   last:       2009-05-30      - OE        - #01       - erstellt
DNSSERVER='10.0.0.103'

#   brief:      DNS-Server-Port.
#
#   last:       2009-05-30      - OE        - #01       - erstellt
DNSSERVER_PORT='53'

#   brief:      Proxy-Server in DMZ.
#
#   last:       2009-05-30      - OE        - #01       - erstellt
PROXY='10.0.0.106'

#   brief:      Proxy-Server in DMZ.
#
#   last:       2009-05-30      - OE        - #01       - erstellt
PROXY_PORT='3128'

#   brief:      Mail-Server in DMZ
#
#   last:       2009-05-30      - OE        - #01       - erstellt
MAILSERVER='10.0.0.101'

#   brief:      Mail-Server-Port für den Versand der Mails.
#
#   last:       2009-05-30      - OE        - #01       - erstellt
MAILSERVER_PORT_SMTP='25'

#   brief:      Mail-Server-Port zum Abholen der Mails
#
#   last:       2009-05-30      - OE        - #01       - erstellt
MAILSERVER_PORT_POP='110'

#   brief:      Mail-Server-Port zum Abholen der Mails
#
#   last:       2009-05-30      - OE        - #01       - erstellt
MAILSERVER_PORT_IMAP='143'
```

```
#   brief:      WebServer in der DMZ
#
#   last:       2009-05-30      - OE        - #01       - erstellt
WEBSERVER="10.0.0.107"

#   brief:      Webserver-Port in DMZ
#
#   last:       2009-05-30      - OE        - #01       - erstellt
WEBSERVER_PORT='80'

#   brief:      OPSI-Server in DMZ
#
#   last:       2009-05-30      - OE        - #01       - erstellt
OPSISERVER="10.0.0.102/32"

#   brief:      OPSI-Server-Port
#
#   last:       2009-05-30      - OE        - #01       - erstellt
OPSISERVER_PORT='11111 '

#   brief:      Samba-Server in DMZ
#
#   last:       2009-05-30      - OE        - #01       - erstellt
SAMBASERVER="10.0.0.104/32"

#   brief:      Samba-Server-Port
#
#   last:       2009-05-30      - OE        - #01       - erstellt
#SAMBASERVER_PORT=

#   brief:      LDAP-Server in DMZ
#
#   last:       2009-05-30      - OE        - #01       - erstellt
LDAPSERVER="10.0.0.104"

#   brief:      LDAP-Server-Port
#
#   last:       2009-05-30      - OE        - #01       - erstellt
LDAPSERVER_PORT='389'
```

```
#   brief:      Externe Netzwerkschnittstelle der Firewall.
#
#   last:       2009-05-30      - OE        - #01       - erstellt
IF_EXT_IP='10.0.0.2'

#   brief:      Name  der  externen Netzwerkschnittstellen Firewall
#
#   last:       2009-05-30      - OE        - #01       - erstellt
IF_EXT='eth1'

#   brief:      Interne Netzwerkschnittstelle der Firewall.
#
#   last:       2009-05-30      - OE        - #01       - erstellt
IF_INT_IP='192.168.10.1'

#   brief:      Name  der  internen Netzwerkschnittstellen Firewall
#
#   last:       2009-05-30      - OE        - #01       - erstellt
IF_INT='eth0'

#   brief:      Adressbereich des externen Netzes (Internet).
#
#   last:       2009-05-30      - OE        - #01       - erstellt
EXT_NET="192.168.1.0/24"

#   brief:      Adressbereich des internen Netzes (Client).
#
#   last:       2009-05-30      - OE        - #01       - erstellt
INT_NET='192.168.10.0/24'

#   brief:      Adressbereich der DMZ
#
#   last:       2009-05-30      - OE        - #01       - erstellt
DMZ_NET="10.0.0.0/24"

#   brief:      Der PC, von dem ein SSH Zugriff auf dieses System gewährt wird.

#   last:       2009-05-30      - OE        - #01       - erstellt
ADMIN_PC='192.168.10.9/24'

#   brief:      Die MAC_Adresse des Admin-PCs
```

```
#
#   last:       2009-05-30      - OE        - #01       - erstellt
ADMIN_PC_MAC=''

#   brief:      Definition für beliebiges Netz
#
#   last:       2009-05-30      - OE        - #01       - erstellt
ALL_NET='0.0.0.0'
######################################## end ##############################################

#   brief:      Das Skript zum Zurücksetzen aller Regeln und Ketteb wird ausgeführt.
#
#   last:       2009-05-30      - OE        - #01       - erstellt
/fw/fw_extern.down

######################################## begin #############################################
#   comment:    Es werden die Default-Policies für die Tabelle 'filter' gesetzt.
#
#   history:    2009-05-30      - OE        - #01       - erstellt

#   brief:      Es wird in der Kette INPUT alles geblockt.
#
#   last:       2009-05-30      - OE        - #01       - erstellt
iptables -P INPUT DROP

#   brief:      Es wird in der FORWARD Kette  alles geblockt.
#
#   last:       2009-05-30      - OE        - #01       - erstellt
iptables -P FORWARD DROP

#   brief:      Es wird in der Kette OUTPUT alles geblockt.
#
#   last:       2009-05-30      - OE        - #01       - erstellt
iptables -P OUTPUT DROP
######################################## end ##############################################

#   brief:      IP-Forward kann nun wieder aktiviert werden, um eine Weiterleitung der Pakete zu ermöglichen.
#
#   last:       2009-05-30      - OE        - #01       - erstellt
echo 1 > /proc/sys/net/ipv4/ip_forward
```

```
######################################## begin ###############################################
#   comment:    Erstellung von Benutzerketten in der Tabelle 'filter'.
#
#   history:    2009-06-02      - OE        - #01       - erstellt

#   brief:      Hier werden Beutzerketten in der Tabelle 'filter' erstellt.
#
#   last:       2009-06-02      - OE        - #01       - erstellt

#   brief:      Kette, für die ICMP-Pakte
#
#   last:       2009-06-02      -OE     -#01        -erstellt
iptables -N icmp

#   brief:      Kette, für die SSH-Eingangspakete
#
#   last:       2009-06-02      -OE     -#01        -erstellt
iptables -N ssh_in

#   brief:      Kette, für die SSH-Ausgangspakete
#
#   last:       2009-06-02      -OE     -#01        -erstellt
iptables -N ssh_out

#   brief:      Kette, für die DNS-Eingangs-Pakte
#
#   last:       2009-06-02      -OE     -#01        -erstellt
iptables -N dns_in

#   brief:      Kette, für die DNS-Ausgangs-Pakte
#
#   last:       2009-06-02      -OE     -#01        -erstellt
iptables -N dns_out

#   brief:      Kette, für die SMTP-Eingangs-Pakte
#
#   last:       2009-06-02      -OE     -#01        -erstellt
iptables -N mail_smtp_in

#   brief:      Kette, für die SMTP-Ausgangs-Pakte
#
```

```
#   last:       2009-06-02      -OE     -#01        -erstellt
iptables -N mail_smtp_out

#   brief:      Kette, für die POP-Eingangs-Pakte
#
#   last:       2009-06-02      -OE     -#01        -erstellt
iptables -N mail_pop_in

#   brief:      Kette, für die POP-Ausgangs-Pakte
#
#   last:       2009-06-02      -OE     -#01        -erstellt
iptables -N mail_pop_out

#   brief:      Kette, für die IMAP-Eingangs-Pakte
#
#   last:       2009-06-02      -OE     -#01        -erstellt
iptables -N mail_imap_in

#   brief:      Kette, für die IMAP-Ausgangs-Pakte
#
#   last:       2009-06-02      -OE     -#01        -erstellt
iptables -N mail_imap_out

#   brief:      Kette, zum RFC-Konformen abweisen der Pakete, falls keine Regel in einer Kette zutreffen sollte
#
#   last:       2009-06-02      -OE     -#01        -erstellt
#Reject-Rule
iptables -N reject_packets

#   brief:      Kette, zum Überprüfen von ungültigen TCP-Flag Kombinationen
#
#   last:       2009-06-02      -OE     -#01        -erstellt
iptables -N tcp-flags_check
######################################## end ##############################################

######################################## begin #############################################
#   comment:    Deklaration von Benutzerketten in der Tabelle 'filter'.
#
#   history:    2009-06-02      - OE        - #02   - Mailregeln hinzugefügt

#   brief:      Es werden alle ICMP-Pakete akzeptiert.
```

```
#
#   last:       2009-06-02      -OE     -#01        -erstellt
iptables -A icmp -p icmp -j ACCEPT

#   brief:      SSH-Pakete werden akzeptiert.
#
#   last:       2009-06-02      -OE     -#01        -erstellt
iptables -A ssh_in -p tcp --dport 22 -m state --state NEW,ESTABLISHED -j ACCEPT
iptables -A ssh_out -p tcp --sport 22 -m state --state ESTABLISHED -j ACCEPT

#   brief:      Es werden die DNS-Pakete vom und zu dem DNS-Server akzeptiert.
#
#   last:       2009-06-02      -OE     -#01        -erstellt
iptables -A dns_in -p udp --sport $DNSSERVER_PORT -j ACCEPT
iptables -A dns_out -p udp --dport $DNSSERVER_PORT -j ACCEPT

#   brief:      Überprüft das TCP-Paket auf gültige TCP-Flag Kombination.
#
#   last:       2009-06-02      -OE     -#01        -erstellt
iptables -A tcp-flags_check -p tcp --tcp-flags ALL ALL -j reject_packets
iptables -A tcp-flags_check -p tcp --tcp-flags ALL NONE -j reject_packets
iptables -A tcp-flags_check -p tcp --tcp-flags ALL FIN,PSH,URG -j reject_packets
iptables -A tcp-flags_check -p tcp --tcp-flags ALL SYN,FIN,PSH,URG -j reject_packets
iptables -A tcp-flags_check -p tcp --tcp-flags ALL SYN,RST,ACK,FIN,URG -j reject_packets
iptables -A tcp-flags_check -p tcp --tcp-flags SYN,FIN SYN,FIN -j reject_packets
iptables -A tcp-flags_check -p tcp --tcp-flags SYN,RST SYN,RST -j reject_packets
iptables -A tcp-flags_check -p tcp --tcp-flags FIN,RST FIN,RST -j reject_packets
iptables -A tcp-flags_check -p tcp --tcp-flags ACK,FIN FIN -j reject_packets
iptables -A tcp-flags_check -p tcp --tcp-flags ACK,URG URG -j reject_packets
iptables -A tcp-flags_check -p tcp --tcp-flags ACK,PSH PSH -j reject_packets
iptables -A tcp-flags_check -p tcp --tcp-flags ACK,PSH PSH -j reject_packets

#   brief:      Dient der Abweisung eines Pakets. Es erfolgt eine Benachrichtigung an den Absender.
#               Vor dem Abweisen, wird das Paket protkolliert.
#
#   last:       2009-06-02      -OE     -#01        -erstellt
iptables -A reject_packets -j LOG
iptables -A reject_packets -p tcp -j REJECT --reject-with tcp-reset
iptables -A reject_packets -p ALL -j REJECT --reject-with icmp-port-unreachable

#   brief:      Es werden die Mail-Pakete vom und zu dem Mail-Server akzeptiert.
```

```
#
#   last:       2009-06-02      -OE     -#01        -erstellt
iptables -A mail_smtp_in -p $MAILSERVER_PROT -d $MAILSERVER --dport $MAILSERVER_PORT_SMTP -m state --state ESTABLISHED,RELATED -j
ACCEPT
iptables -A mail_smtp_out -p $MAILSERVER_PROT -s $MAILSERVER --sport $MAILSERVER_PORT_SMTP -m state --state NEW,ESTABLISHED,RELATED
-j ACCEPT
iptables -A mail_pop_in -p $MAILSERVER_PROT -d $MAILSERVER --dport $MAILSERVER_PORT_POP -m state --state ESTABLISHED,RELATED -j ACCEPT
iptables -A mail_pop_out -p $MAILSERVER_PROT -s $MAILSERVER --sport $MAILSERVER_PORT_POP -m state --state NEW,ESTABLISHED,RELATED -j
ACCEPT
iptables -A mail_imap_in -p $MAILSERVER_PROT -d $MAILSERVER --dport $MAILSERVER_PORT_IMAP -m state --state ESTABLISHED,RELATED -j
ACCEPT
iptables -A mail_imap_out -p $MAILSERVER_PROT -s $MAILSERVER --sport $MAILSERVER_PORT_IMAP -m state --state NEW,ESTABLISHED,RELATED
-j ACCEPT
###################################### end ##############################################



###################################### begin ############################################
#   comment:    Regeldefinition in der Tabelle 'filter', Kette INPUT
#
#   history:    2009-06-03      - OE        - #01   - erstellt

#   brief:      Spoofing verhinden
#
#   last:       2009-06-03      -OE     -#01        -erstellt
iptables -A INPUT -s $DMZ_NET -j DROP

#   brief:      SSH-Pakete werden reingelassen.
#
#   last:       2009-06-03      -OE     -#01        -erstellt
iptables -A INPUT -i $IF_EXT -p tcp --dport 22 -m state --state NEW,ESTABLISHED -j ACCEPT

#   brief:      ICMP-Pakete beim Eintreffen an das entsprechende Ziel weiterleiten.
#
#   last:       2009-06-03      -OE     -#01        -erstellt
iptables -A INPUT -j icmp

#   brief:      SSH-Pakete beim Eintreffen an das entsprechende Ziel weiterleiten.
#
#   last:       2009-06-03      -OE     -#01        -erstellt
iptables -A INPUT -j ssh_in
```

```
#   brief:      Zugehörige HTTP-Pakete szulassen
#
#   last:       2009-06-03      -OE     -#01        -erstellt
iptables -A INPUT -i $IF_EXT -p tcp --sport 80 -m state --state ESTABLISHED -j ACCEPT

#DNS-Pakete zulassen
#   brief:      DNS-Pakete beim Eintreffen an das entsprechende Ziel weiterleiten.
#
#   last:       2009-06-03      -OE     -#01        -erstellt
iptables -A INPUT -i $IF_INT -s $DNSSERVER -j dns_in

#   brief:      DHCP-Pakete beim Eintreffen an das entsprechende Ziel weiterleiten.
#
#   last:       2009-06-03      -OE     -#01        -erstellt
iptables -A INPUT -i $IF_INT -p udp -s $ALL_NET --sport 68 -j ACCEPT

#   brief:      Letzte Regel in der Kette INPUT. Pakete werden protokolliert und abgewiesen.
#
#   last:       2009-06-03      -OE     -#01        -erstellt
iptables -A INPUT -j reject_packets
######################################## end ##############################################

######################################## begin #############################################
#   comment:    Regeldefinition in der Tabelle 'filter', Kette FORWARD
#
#   history:    2009-06-04      - OE        - #01   - erstellt

#   brief:      DNS- IN und OUT-Pakete weiterleiten
#
#   last:       2009-06-04      -OE     -#01        -erstellt
iptables -A FORWARD -j dns_in
iptables -A FORWARD -j dns_out

#   brief:      Pakete vom und zu Proxy weiterleiten
#
#   last:       2009-06-04      -OE     -#01        -erstellt
iptables -A FORWARD -i $IF_INT -o $IF_EXT -p tcp -s $PROXY -j ACCEPT
iptables -A FORWARD -o $IF_INT -p tcp -d $PROXY -j ACCEPT

#   brief:      Routing-Pakete aus dem DMZ-Netz weiterleiten
```

```
#
#   last:       2009-06-04      -OE     -#01        -erstellt
iptables -A FORWARD -i $IF_INT -o $IF_INT -s $DMZ_NET -j ACCEPT

#   brief:      ICMP-Pakete  weiterleiten
#
#   last:       2009-06-04      -OE     -#01        -erstellt
iptables -A FORWARD -j icmp

#   brief:      EMail- IN und OUT-Pakete weiterleiten
#
#   last:       2009-06-04      -OE     -#01        -erstellt
iptables -A FORWARD -i $IF_EXT -o $IF_INT -j mail_smtp_in
iptables -A FORWARD -i $IF_INT -o $IF_EXT -j mail_smtp_out
iptables -A FORWARD -i $IF_EXT -o $IF_INT -j mail_pop_in
iptables -A FORWARD -i $IF_INT -o $IF_EXT -j mail_pop_out
iptables -A FORWARD -i $IF_EXT -o $IF_INT -j mail_imap_in
iptables -A FORWARD -i $IF_INT -o $IF_EXT -j mail_imap_out

#   brief:      Letzte Regel in der Kette FORWARD. Pakete werden protokolliert und abgewiesen.
#
#   last:       2009-06-04      -OE     -#01        -erstellt
iptables -A FORWARD -j reject_packets
######################################## end ##############################################

######################################## begin #############################################
#   comment:    Regeldefinition in der Tabelle 'filter', Kette OUTPUT
#
#   history:    2009-06-03      - OE        - #01   - erstellt

#   brief:      ICMP-Pakete werden rausgelassen
#
#   last:       2009-06-03      -OE     -#01        -erstellt
iptables -A OUTPUT -j icmp

#   brief:      SSH-Pakete werden rausgelassen
#
#   last:       2009-06-03      -OE     -#01        -erstellt
iptables -A OUTPUT -j ssh_out

#   brief:      HTTP-Pakete werden rausgelassen
```

```
#
#   last:       2009-06-03      -OE     -#01        -erstellt
iptables -A OUTPUT -o $IF_EXT -p tcp --dport 80 -m state --state NEW,ESTABLISHED -j ACCEPT

#   brief:      DNS-Pakete werden rausgelassen
#
#   last:       2009-06-03      -OE     -#01        -erstellt
iptables -A OUTPUT -d $DNSSERVER -j dns_out

#   brief:      Letzte Regel in der Kette OUTPUT. Pakete werden protokolliert und abgewiesen.
#
#   last:       2009-06-03      -OE     -#01        -erstellt
iptables -A OUTPUT -j reject_packets
###################################### end ##############################################

###################################### begin ############################################
#   comment:    Regeldefinition in der Tabelle 'nat', Kette PREROUTING
#
#   history:    2009-06-07      - OE        - #01   - erstellt

#   brief:      HTTP-Anfragen aus dem Internet, sollen an den internen Webserver umgeleitet werden
#
#   last:       2009-06-07      -OE     -#01        -erstellt

iptables -t nat -A PREROUTING -i $IF_EXT -p tcp --dport $WEBSERVER_PORT -m state --state NEW,ESTABLISHED -j DNAT --to
$WEBSERVER:$WEBSERVER_PORT

#   brief:      HTTP-Pakete aus dem Client-und DMZ-Netz sollen zum internen Proxy umgeleitet werden.
#
#   last:       2009-06-07      -OE     -#01        -erstellt
iptables -t nat -A PREROUTING -i $IF_INT ! -s $PROXY -p tcp --dport 80 -m state --state NEW -j DNAT --to $PROXY:$PROXY_PORT

#   brief:      SMTP-Pakete aus dem Internet sollen zum internen Mailserver umgeleitet werden.
#
#   last:       2009-06-07      -OE     -#01        -erstellt
iptables -t nat -A PREROUTING -i $IF_EXT -p tcp --dport $MAILSERVER_PORT_SMTP -m state --state NEW -j DNAT --to
$MAILSERVER:$MAILSERVER_PORT_SMTP

#   brief:      POP-Pakete aus dem Internet sollen zum internen Mailserver umgeleitet werden.
#
#   last:       2009-06-07      -OE     -#01        -erstellt
```

```
iptables -t nat -A PREROUTING -i $IF_EXT -p tcp --dport $MAILSERVER_PORT_POP -m state --state NEW -j DNAT --to
$MAILSERVER:$MAILSERVER_PORT_POP

#   brief:      IMAP-Pakete aus dem Internet sollen zum internen Mailserver umgeleitet werden.
#
#   last:       2009-06-07      -OE     -#01        -erstellt
iptables -t nat -A PREROUTING -i $IF_EXT -p tcp --dport $MAILSERVER_PORT_IMAP -m state --state NEW -j DNAT --to
$MAILSERVER:$MAILSERVER_PORT_IMAP
######################################## end ##############################################

######################################## begin #############################################
#   comment:    Regeldefinition in der Tabelle 'nat', Kette POSTROUTING
#
#   history:    2009-06-07      - OE        - #01   - erstellt

#   brief:      Masquerading für ins Internet ausgehende Pakete aktivieren.
#
#   last:       2009-06-07      -OE     -#01        -erstellt
iptables -t nat -A POSTROUTING -o $IF_EXT -j MASQUERADE
######################################## end ##############################################
```