Kapitel 3

IP 3 - Firewall-Versuch

Im Wandel der Zeit werden immer mehr private Netze mit dem Internet verbunden. Der Trend sich im Internet zu präsentieren, Informationen anzubieten sowie diverse im Internet vorhandene Dienste zu nutzen, scheint ungebrochen. Der Zusammenschluß verschiedener Netze, insbesondere der Anschluß an das Internet, birgt jedoch erhebliche Sicherheitsrisiken. Zum Schutz der privaten Netze vor dem „bösen Internet“ werden Firewalls eingesetzt.

3.1 Theorie

3.1.1 Einführung

Der Firewall-Versuch kann als Fortsetzung des vorangegangenen Versuchstages angesehen werden. Die Einführung und die Theorie-Fragen geben einen Einblick in die Themen Firewall und Sicherheit. Im Praxisteil werden Sie sich - am Beispiel von Netfilter/iptables - mit der Konfiguration eines **Paket-Filters** vertraut machen.

Diejenigen Studenten, die `iptables` oder seinen Vorgänger `ipchains` schon verwendet haben, werden die Versuche ebenso schnell lösen können, wie das beireits bei der Netzkonfiguration der Fall war. Alle anderen sollten sich **vor** dem Versuchstag mit der Architektur von `iptables` vertraut machen (z.B. mittels der `iptables` manpage). Denn nur bei ausreichender Vorbereitung sind die Aufgaben in moderater Zeit zu lösen.

3.1.1.1 Einleitung

Der Zusammenschluß verschiedener Netze, insbesondere der Anschluß an das Internet, ist mit Sicherheitsrisiken verbunden. Zum Schutze privater oder behördeninterner Netzes werden Firewalls eingesetzt. In einem Gebäude dient eine Brandmauer (Firewall) dazu, das Übergreifen eines Feuers auf andere Gebäudeteile zu verhindern. Eine Internet-Firewall verfolgt im Prinzip ein ähnliches Ziel: Sie soll verhindern, dass die Gefahren des Internets

auf das interne Netz übergreifen. Aber auch **innerhalb** eines Firmennetzes können Firewalls nützliche Dienst leisten. Hier werden Sie häufig eingesetzt, um einzelne Abteilungen voneinander abzugrenzen.

Im Szenario des Praktikums ist eine Firewall ein Gerät, welches die einzige physische Verbindung zwischen einem privaten Netz und einem öffentlichen Netz darstellt und gegenüber „dahergelaufenen“ IP-Paketen als eine als „mürrischer Türsteher“ fungiert. Die Firewall ist - natürlich - sowohl mit dem privaten Netz als auch mit dem öffentliche Netz verbunden. Hierbei ist von entscheidender Wichtigkeit, dass keine vergessenen „Hintereingänge“ existieren, die das private und das öffentlich Netze verbinden und hierbei an der Firewall vorbeiführen. [gren96]

3.1.1.2 Sicherheitspolitik

Die Installation einer Firewall sollte in Verbindung mit einer **Sicherheitspolitik** stehen. Die Sicherheitspolitik ist vorab zu definieren, um die Rahmenbedingungen der Sicherheit im Netz festzulegen. Es muß darauf geachtet werden, dass die Benutzer in den Rahmen der Sicherheitspolitik eingebunden werden. Dienste die von der Sicherheitspolitik ausgeschlossen werden, sollten auf eine dem Benutzer verständliche Art und Weise begründet werden. Auch ist es wichtig, dass der Benutzer die Sicherheitsproblematik erkennt und somit den Einsatz von Sicherheitsmaßnahmen akzeptiert. Im Idealfall gilt es eine **Benutzerordnung** zu erstellen. Diese sollte allerdings nicht aus Ver- und Geboten sondern - wie bereits erwähnt - aus **Informationen für die Benutzung** bestehen. Für eine Sicherheitspolitik sind mindestens folgende Fragen zu beantworten (siehe auch [fuhr98]):

- Welcher Schutzbedarf ist nötig?
- Wie sieht die Struktur des vorhandenen Netzes aus?
- Wie sieht das Kommunikationsprofil aus?
- Welche Informationen soll die Firewall verdecken?
- Wer ist Administrator der Firewall?
- Wer ist für die Protokollauswertung verantwortlich?
- Auf welche Weise sollen die Endbenutzer vor Schadsoftware geschützt werden?
- Wer ist für die Erstellung von Backups verantwortlich?

3.1.1.3 Grundsätzliche Hinweise

Alle Rechner im Intranet müssen, um aus dem zu schützenden Netz auf das Internet zuzugreifen, die Firewall als **Gateway** benutzen. Ist diese Voraussetzung nicht gegeben, kann,

je nachdem welche Dienste von diesem Rechner angeboten werden, die Sicherheit des gesamten Intranets ausgehebelt werden. Man sollte deshalb alle ISDN-Geräte und Modems aus den Rechnern soweit als möglich entfernen.

Auch wenn eine Firewall einsetzt wird, sollte man trotzdem die Sicherheit der lokalen Rechner nicht aus dem Auge verlieren. Je mehr Hürden ein Angreifer zu überwinden hat, desto eher wird er vor Erreichen seines Ziels aufgeben oder bemerkt [died98]. Auch die Möglichkeit, dass ein Angreifer von innen kommt, ist nicht außer acht zu lassen.

3.1.1.4 Vorgehensweise

Als erstes werden alle nicht benötigten Netz- und Systemdienste deaktiviert. Auf einem Firewallrechner sollten keine zusätzlichen Dienste angeboten werden, da jeder zusätzliche Dienst, wiederum Sicherheitslücken aufweisen kann. Handelt es sich um eine Softwarelösung, sollte der Rechner als Minimalsystem ausgelegt sein (z.B. ohne graphische Oberfälche).

Bei der Konfiguration der Firewall selbst gibt es zwei grundsätzlich unterschiedliche Vorgehensweisen. Zum einen die **optimistische** Vorgehensweise. Hierbei werden zunächst alle Dienst freigeschaltet und anschließend wird versucht, diejenigen Dienste, die eine Sicherheitsrisiko darstellen, abzuschalten.

Diesem optimitischen Ansatz steht der sog. **pessimistische** gegenüber; i.e. alles, was nicht ausdrücklich erlaubt ist, ist verboten. Zunächst werden als sämtliche Dienste abgeschaltet. Anschließend werden nur solche Ports und IP Addressen freigeschaltet, die für die anzubietenden Dienste zwingend erforderlich sind. Letzteres Vorgehen hat den Vorteil, dass auch viele bis dato noch unbekannte Sicherheitsrisiken ausgeschlossen werden und stellt mithin - zumindest soweit es Firewalls betrifft - den einzig richtigen Ansatz dar.

3.1.2 Firewall-Typen

Generell unterscheidet man zwei verschiedene Typen von Firewalls. Zum einen die sog. **Paket-Filter** und zum anderen die **Application Level Firewalls** (auch Proxies genannt). Im Rahmen dieses Versuchstages werden Sie nur Paket-Filter kennenlernen. Typischerweise werden die beiden Firewalltypen allerdings kombiniert verwendet. Falls Sie zu dem schönen Themengebiet der Firewalls weitergehendes Interesse aufbauen sollten, so steht Ihnen mit dem IT-Sicherheitspraktikum eine hervorragende Möglichkeit offen, Ihre einschlägigen Kenntnisse zu vertiefen.

3.1.2.1 Paket Filter

Ein Paket-Filter Firewall arbeitet typischerweise auf den OSI Schichten 3 und 4. Die - überaus klar durchkonzipierte - Paket-Filter Firewall 'Netfilter/iptables' bietet Ihnen überdies Zugriff auf Informationen der MAC-Teilschicht der OSI Schicht 2 (MAC-Adressen). Die

Paket Filter Firewall überprüft alle ankommenden und ausgehenden Protocol Data Units (PDUs), auf bestimmte Informationen, welche dem jeweiligen Protokollheader entnommen werden. Der Firewalladministrator hat die Möglichkeit, hierfür spezielle Regeln zu definieren. Die PDUs werden dann gegenüber diesen Regeln ausgewertet. Eine Regel kann das Passieren ('accept') oder Zurückgeweisen der Pakete durch die Firewall bewirken. Existiert eine accept-Regel, wird das Paket weitergeroutet. Bei zurückweisenden Regeln unterscheidet man zwischen 'reject' und 'deny'. Im 'reject' Fall wird eine Meldung an den Benutzer zurückgegeben (not reachable oder permission denied), im 'deny' Fall wird keine solche Rückmeldung erzeugt. Um einen Dienst freizuschalten, müssen zwei Regeln pro Interface eingetragen werden. Das liegt daran, dass die meisten Protokolle bidirektional sind. Man definiert eine Regel, die den Datenstrom von der Quelle zum Ziel erlaubt und eine weitere Regel, die den Antwortdatenstrom passieren läßt. Die Abbildungen 3.1, 3.2, 3.3 und 3.4 zeigen die Protokollheader der im Internetumfeld wichtigsten Protokolle [fuhr98, Seite 137/138]. Die Optionen, nach denen gefiltert werden kann, wurden grau unterlegt. Übrigens ist auch das Interface, über welches ein IP-Paket eingeht, eine wertvolle Information für den Paket-Filter. Diese Information ist nämlich absolut fälschungssicher.

3.1.2.2 Beispielkonfiguration eines statischen Paketfilters

Ein Regelsatz eines statischen Paketfilters, der Telnetsitzungen von dem zu schützenden Netz aus zu Rechnern im Internet erlaubt, könnte z.B. folgendermaßen aussehen: Regel A erlaubt, dass aus dem zu schützenden Netz (intern) eine Verbindung auf Port 23 (Telnet) in Richtung des Zielrechners (out) hergestellt werden darf. Regel B erlaubt die Kommunikation vom Server zurück zum Client. Regel C resultiert aus dem pessimistischen Ansatz und verbietet jeden anderen Dienst.

Rule	Direction	Source	Destination	Protocol	Source Port	Dest.Port	Flags	Action
A	out	intern	any	TCP	>1023	23	any	accept
B	in	any	intern	TCP	23	>1023	ACK	accept
C	any	any	any	any	any	any	any	deny

Tabelle 3.1: Filtertabelle für nach außen gerichtetes Telenet

3.1.2.3 Dynamische Paket-Filter (stateful inspection)

Gewöhnliche Paket-Filter Firewalls (statische Paket-Filter) bieten gerade für TCP/IP eine gute Möglichkeit der Filterung. Schwieriger wird es, möchte man z.B. das **verbindungslose** Protokoll UDP filtern. Durch das Fehlen von Verbindungsstatusinformationen, ist für die Firewall am Header einer ankommenden UDP- Protocol Data Unit nicht erkennbar, ob es sich um ein Datagramm einer bereits bestehende Verbindung oder um eine Verbindungsanforderung handelt. **Dynamische Paket-Filter** sind eine Weiterentwicklung

herkömmlicher Paket-Filter, die genau dieses Problem lösen. Der dynamische Filter führt über die augenblicklichen Kommunikationsvorgänge eigenständig Buch (in einer Verbindungsstatustabelle) und ist überdies in der Lage, seine Filterregeln kurzzeitig zu ändern, um somit bestimmte **erwartete** Pakete (und nur diese) die Firewall passieren zu lassen. Für diese Art der Paketfilterung haben sich die beiden Ausdrücke **Stateful Inspection** und **Connection Tracking** eingebürgert.

3.1.2.4 Filtermöglichkeiten IP

Der Header einer IP-PDU (Abbildung 3.1) bietet eine Fülle von Filteroptionen.

- Man kann nach **Quell- und Zieladressen** filtern.
- In das Feld **Protokoll der Transportschicht** wird der Protokolltyp der übergeordneten Schicht eingetragen (dieses Feld stellt übrigens eine gewisse Durchtrennung der OSI-Schichtarchitektur dar). Der Paketfilter hat somit die Möglichkeit nach Transportprotokollen zu filtern. Die genaue Semantik des protocol-Feldes findet sich in den RFCs 790 und 1010, wobei die Festlegungen im wesentlichen mit dem Inhalt der Datei `/etc/protocols` übereinstimmen.
- Die **Flags** geben unter anderem Auskunft über die Fragmentierung.

3.1.2.5 Filtermöglichkeiten ICMP

Bei ICMP Paketen (Abbildung 3.2) enthält das Typenfeld die entscheidende Information. Als Typ kann beispielsweise „Echo Request“ oder „Echo Reply“ eingetragen sein (siehe auch Tabelle 3.2). Der Paketfilter `iptables` ermöglich aber durchaus auch Filterung aufgrund des Inhaltes des Code-Feldes im ICMP-Header **(auch das Code-Feld ist grau zu unterlegen)**.

0	Echo Reply	13	Timestamp Request
3	Destination Unreachable	14	Timestamp Reply
4	Source Quench	15	Information Request
5	Redirect (change route)	16	Information Reply
8	Echo Request	17	Adress Mask Request
11	Time Exceeded	18	Adress Mask Reply
12	Parameter Problem		

Tabelle 3.2: Das ICMP Typfeld

<table>
<tr><td>4-Bit</td><td>4-Bit</td><td>8-Bit</td><td colspan="2">16-Bit</td></tr>
<tr><td>Version</td><td>Header-
Länge</td><td>Service-Type</td><td colspan="2">Länge des Datagramms</td></tr>
<tr><td colspan="3">Identifikationsnummer</td><td>Flags</td><td>Offset des Fragments</td></tr>
<tr><td colspan="2">Time to Live</td><td>Protokoll der
Transportschicht</td><td colspan="2">Prüfsumme des Headers</td></tr>
<tr><td colspan="5">Internet-Adresse des Quell-Rechners</td></tr>
<tr><td colspan="5">Internet-Adresse des Ziel-Rechners</td></tr>
<tr><td colspan="4">Optionen (z.B. Source Routing)</td><td>Füllzeichen</td></tr>
<tr><td colspan="5">Header des Transportprotokolls</td></tr>
</table>

Abbildung 3.1: Aufbau des IP-Headers

8-Bit	8-Bit	16-Bit
Typ	Code	Prüfsumme

Abbildung 3.2: Aufbau des ICMP-Headers

3.1.2.6 Filtermöglichkeiten TCP

Für die Filterung von TCP Paketen (Abbildung 3.3) kommen die Portnummern und Flags in Frage. Die Portnummer dient als Schnittstelle zur nächsthöheren Schicht. Dabei ist jedem Dienst eine Portnummer zugeordnet. Bei Unix Systemen wird die Zuordnung über die Datei `/etc/services` konfiguriert. Die Tabelle 3.3 zeigt typische Portnummern von TCP-basierten Diensten. Über die Flags läßt sich beispielsweise bestimmen, ob es sich um einen Verbindungsaufbau handelt, oder um eine bereits bestehende Verbindung. Möchte man von außen keinen Zugriff über TCP auf das interne Netz erlauben, muß man nur alle eingehenden Pakete herausfiltern, die das ACK-Flag nicht gesetzt haben.

<table>
<tr><td colspan="2">8-Bit</td><td>8-Bit</td><td>16-Bit</td></tr>
<tr><td colspan="3">Portnummer des Absenders</td><td>Portnummer des Empfängers</td></tr>
<tr><td colspan="4">Sequenznummer</td></tr>
<tr><td colspan="4">Quittungsnummer</td></tr>
<tr><td>Header-
Länge</td><td>Reser-
viert</td><td>Flags</td><td>Fenstergröße</td></tr>
<tr><td colspan="3">Prüfsumme des Headers</td><td>Urgent-Pointer</td></tr>
<tr><td colspan="3">Optionen</td><td>Füllzeichen</td></tr>
</table>

Abbildung 3.3: Aufbau des TCP-Headers

20	ftp-data	80	http
21	ftp	110	pop3
22	ssh	119	nntp
23	telnet	443	https
25	smtp	513	rlogin
53	dns (Zonentransfer)	6000+n	X11
79	finger		

Tabelle 3.3: Ports und Dienste einiger ausgewählter TCP-basierter Protokolle

3.1.2.7 Filtermöglichkeiten UDP

UDP ist ein verbindungsloses Protokoll der Schicht 4. Die einzigen Informationen, welche der Header der UDP-PDU (Abbildung 3.4) bereitstellt, sind die Portnummern. Über diese kann der angesprochene Dienst herausgefunden werden (vgl. auch Tabelle 3.4).

3.1.2.8 Application Level Firewall

Hierunter versteht man sogenannte **Proxy Dienste** (Proxy = Bevollmächtigter). Proxy Dienste kommunizieren stellvertretend für einen Client mit einem Server außerhalb des Subnetzes. Der Verbindungsaufbau geschieht in zwei Phasen. Zuerst stellt der Client eine Verbindung zum Proxy Server her. Dieser entscheidet dann, ob der angeforderte Dienst vom Client am Zielrechner genutzt werden darf. Ist das der Fall, baut der Proxy Server stellvertretend für den Client eine Verbindung zum angewählten Server auf. Aus Sicht

16 Bit	16 Bit
Portnummer des Absenders	Portnummer des Empfängers
Datagrammlänge	Prüfsumme

Abbildung 3.4: Aufbau des UDP-Headers

53	dns	161	snmp
111	Sun RPC	162	snmptrap
		517	talk

Tabelle 3.4: Ports und Dienste UDP-basierter Protokolle

des Clients ist der Proxy Server der kontaktierte Zielrechner. Aus Sicht des angewählten Servers spielt der Proxy Server die Rolle des Clients; siehe Abbildung 3.5. Proxy Server arbeiten auf Schicht 7 (Application Layer). Der Verbindungsaufbau geschieht dabei weitestgehend transparent, so daß der Benutzer davon nichts mitbekommen sollte. Es werden zwei verschiedene Funktionsweisen unterschieden:

- angepaßte Client-Software
- modifizierte Verfahren für Benutzer

Bei den Application Level Firewalls unterscheidet man den Application Level Proxy und den Circuit Level Proxy. Das Application Level Proxy spricht das Protokoll, für das es Proxy Dienste leistet. Es kann die Kommandos des Anwendungsprotokolls verstehen und interpretieren. Application Level Proxies arbeiten häufig mit modifizierten Verfahren. Der Benutzer muß in den meisten Fällen die Kommunikation nicht explizit mit dem Proxy führen, da dieser ja das Protokoll spricht und sich aus diesem die erforderlichen Informationen ergeben. Viele Proxy Server bieten außerdem zusätzliche Funktionalität, wie das Zwischenspeichern von Daten. Diese Daten müssen dann nicht immer von neuem geholt werden, sondern werden aus dem Cache des Proxies geladen. Darüber hinaus bieten Sie bessere Protokollierungsmöglichkeiten und Zugangskontrollen.

Circuit Level Proxies arbeiten auf der Sitzungsschicht. Sie sind nicht in der Lage das Protokoll der Anwendungsschicht zu interpretieren. Ein Circuit Level Proxy kontrolliert das Handshaking beim Verbindungsaufbau. Erst wenn der Proxy feststellt, daß Client und Server authorisiert sind, eine Verbindung herzustellen, werden Daten über den Proxy hinweg übertragen. Die Daten werden dann vom Circuit Level Proxy für die Sitzungsdauer nur noch kopiert und weitergeleitet. Der Vorteil von Circuit Level Proxies ist, dass sie Dienste für eine Vielzahl verschiedener Protokolle bieten bzw. an diese angepaßt werden können.

Außerdem können mehrere Anwendungsprotokolle von einem Proxyprozeß verarbeitet werden. Das hat zum einen den Vorteil, dass man das Risiko fehlerhafter Software verkleinert und zum anderen, dass sich Aufwand und Fehler bei der Konfiguration verkleinern, da diese nicht mehr für jeden Proxy separat vorzunehmen ist.

Da Circuit Level Proxies eine Verbindung nur auf Sitzungsschicht auswerten, kann nicht überprüft werden, welches Anwendungsprotokoll tatsächlich über die Verbindung läuft. Ist eine Sitzung erst einmal initiiert, kann prinzipiell jedes Anwendungsprotokoll gefahren werden.

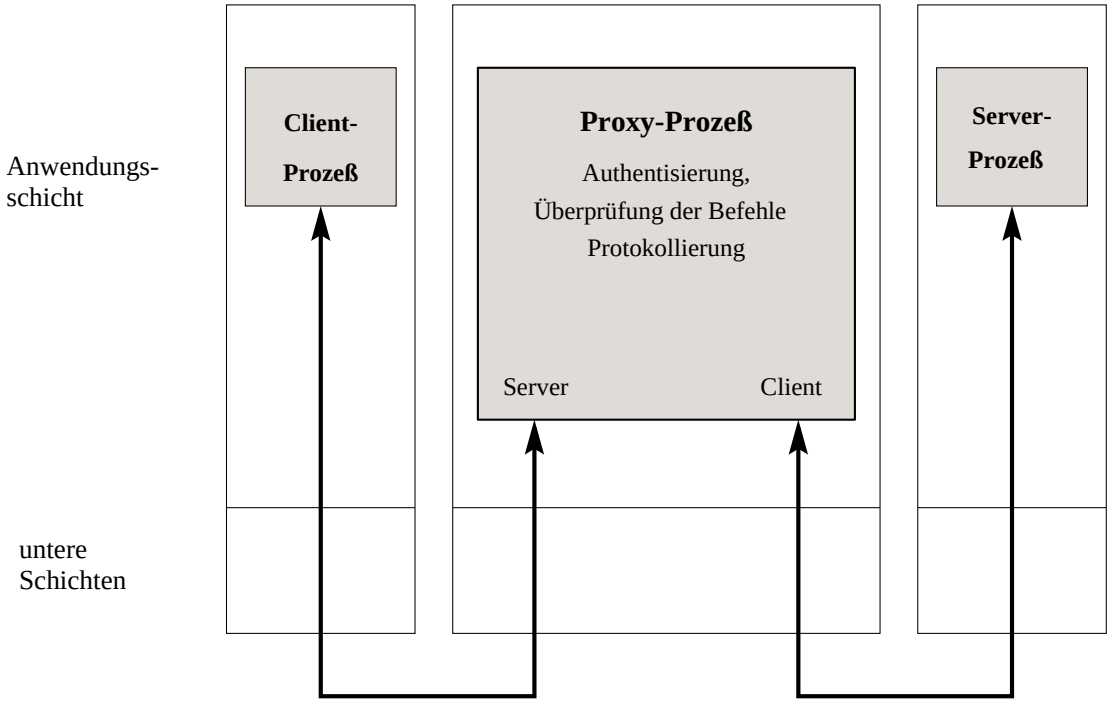

Abbildung 3.5: Funktionsweise eines Proxies

Circuit Level Proxies gibt es sowohl mit modifizierten Verfahren (beispielsweise könnte man beim Aufruf eine Portnummer angeben, über die der Zielhost identifiziert werden kann) als auch mit angepaßten Clients.

Üblicherweise werden Circuit Level Proxies nicht stand-alone, sondern gebündelt mit Application Level Proxies angeboten. Nicht jedes Protokoll eignet sich gleichermaßen für einen Proxy Einsatz. Store and Forward Protokolle eignen sich beispielsweise gut, da die Daten kurzfristig zwischengespeichert werden. Andere Protokolle eignen sich weniger z.B. talk (Verbindungsaufbau über UDP, Daten werden über TCP übertragen (siehe [chap96]) oder RPC basierte Dienste (verwenden oftmals keine festen Portnummern).

3.1.3 Architekturen

Firewalls lassen sich in beinahe beliebiger Kombination aus den beiden Funktionen Paket-Filter und Application Gateway zusammenstellen. Als Grundsatz für jede Architektur gilt, dass die Position der Firewall immer soweit außen wie möglich liegen sollte.

Die einfachste Möglichkeit eines Firewallaufbaus besteht aus einem Packet Filter. Dieser wird zwischen das Internet und das zu schützende Netz geschaltet. Diese Lösung reicht für kleine Netze mit einem begrenzten Dienstangebot aus, das zudem nur vom zu schützenden Netz aus genutzt wird. Packet Filter können auch sinnvoll im Intranet eingesetzt werden, indem Sie einzelne Teilnetze voneinander trennen.

3.1.3.1 Dual-Homed-Application-Gateway

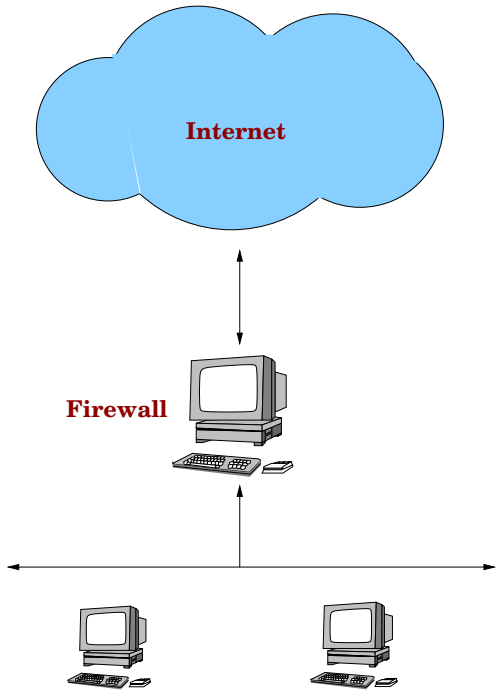

Abbildung 3.6: Dual Homed Host Architecture

Eine Erhöhung der Sicherheit gegenüber einfachen Paketfiltern bietet der Einsatz von Application Level Proxies, die den Netzverkehr zwischen Internet und dem zu schützenden Netz kontrollieren (siehe Abbildung 3.6). Ein solcher Rechner wird mit mindestens zwei Netzschnittstellen ausgestattet (mit mehr als zwei Netzschnittstellen wird ein solcher Rechner auch als Multi-Homed-Application Gateway bezeichnet. Ist ein Application Gateway als erster oder einziger Rechner aus dem Internet erreichbar, bezeichnet man ihn auch als **Bastion Host**. Eine Auftrennung in mehrere Netzschnittstellen hat den Vorteil, dass

der Netzverkehr nur innerhalb des Subnetzes sichtbar ist und jede Verbindung über den Gateway geroutet werden muß.

3.1.3.2 Screened Subnet Architecture

Bei dieser Architektur wird zwischen dem zu schützenden Netz und dem Internet zusätzlich ein weiteres Netz aufgebaut (siehe Abbildung 3.7. Dieses Subnetz wird in der Literatur häufig als Perimeter Netz, Grenznetz oder DeMilitarised Zone (DMZ) bezeichnet.

Bei Verwendung eines Subnetzes baut man eine zusätzliche Sicherungsschicht zwischen dem Internet und dem zu schützenden Netz auf. Das hat zur Folge, dass im Falle eines Einbruchs in den Bastion Host, der Eindringling nicht den gesamten Netzverkehr abhören kann, sondern nur den Netzverkehr im Grenznetz (bei moderneren Netzinfrastrukturen wie Switched Ethernet tritt dieses Problem nicht mehr auf). Soll das innere Netz erreicht werden, müssen die Daten aus dem Internet durch das Grenznetz geschickt werden.

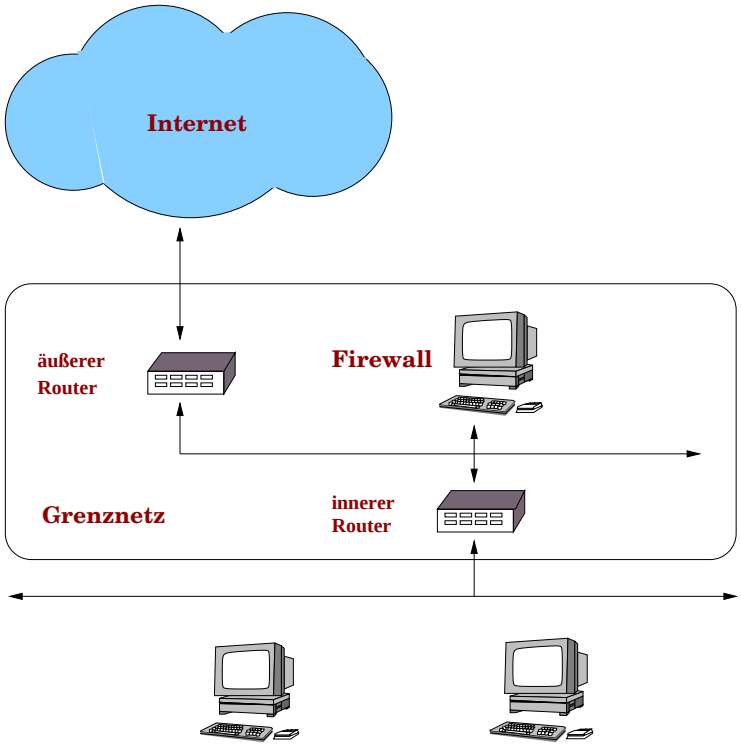

Abbildung 3.7: Screened Subnet Architecture

Der äußere Router wird häufig dazu benutzt, IP Pakete mit gefälschten Absenderadressen auszufiltern, indem er Pakete mit Quelladressen des Subnetzes ausfiltert. Außerdem werden alle Pakete ausgefiltert, die als Ziel nicht den Application Gateway adressieren.

Der innere Router bildet eine zusätzliche Barriere. Er schützt das innere Netz zum einen vor

Angriffen aus dem Internet, zum anderen gegen Angriffe aus dem Grenznetz. Außerdem sollte der innere Router sicherstellen, dass Dienste, für die ein Application Level Proxy bereit steht, nur über das Gateway benutzbar sind. Typischerweise ist der innere Router so konfiguriert, dass nur der Application Level Proxy des Grenznetzes Zugriff auf das zu schützende Netz erhält.

Das Grenznetz kann neben den Firewallkomponenten noch weitere Rechner enthalten (z.B. DNS-Server, WWW-Server, ...), die so ebenfalls einen gewissen Schutz erhalten. Weitere Hinweise zur Positionierung von Diensten im Grenznetz finden Sie in [fuhr98, Kapitel 5/6] und [chap96, Kapitel 4]. Was bei der Konfiguration eines Bastion Hosts zu beachten ist, wird detailliert in [chap96, Kapitel 5] erörtert.

3.1.4 Aufgaben zur Theorie

1. Welche Bedrohungsszenarien für ein Netz kennen Sie? (siehe [chap96, Seite 7ff]
2. Stellen Sie die Vor- und Nachteile der beiden Firewalltypen gegenüber.
3. Welche Anforderungen sollte ein Packet Filter Firewall erfüllen? Begründen Sie ihre Antwort.
4. Machen Sie sich den Verbindungsaufbau einer TCP/IP Verbindung klar. Welche Rolle spielt das ACK-Flag?
5. Erklären Sie kurz den Unterschied der beiden FTP Modi normal und passiv. Welchen Modus wird man in Verbindung mit einem Paketfilter einsetzen? Begründen Sie ihre Meinung! (siehe [chap96, Kapitel 8, Seite 252ff])
6. Welche Nachteile bringt der Einsatz von Firewall Rechnern?
7. Erstellen Sie einen Regelsatz eines Paketfilters, der **Telnet** von `pcrt1` auf `pcrnp10` erlaubt. Gehen Sie davon aus, daß das interne Netzinterface der Firewall mit `eth1` und das äußere mit `eth0` bezeichnet wird.

3.1.5 Theoriefragen zu Netfilter/iptables

1. Machen Sie sich mit den Kommandos `lsof` und `netstat` vertraut. Welche Informationen liefern insbesondere die Aufrufe `lsof -i`, `netstat -tu` und `netstat -lp`?
2. Der Paketfilter 'netfilter' ist als Kernelmodul realisiert. Konfiguriert wird dieser kernelinterne Filter mit Hilfe des Kommandos `iptables`. Verdeutlichen Sie sich kurz das Konzept von `netfilter/iptables` z.B. mittels der entsprechenden Manpage oder mit Hilfe des ausliegenden Paket Filter Tutorial.

3. Was versteht man unter einer „chain“? Erklären Sie kurz die Aufgaben der 5 Standard chains und stellen Sie den Weg eines Datenpaketes durch den Kernel (die einzelnen chains) grafisch da.

4. Was ist die Aufgabe einer „table“ und welche stellt das System zu Verfügung? Welche Zuordnung kann zwischen den standard chains und den 3 vorgegebenen tables gemacht werden? Welche table ist für unseren einfachen Paketfilter interessant?

5. Die Paket-Filter `netfilter/iptables` besitzt ausgefeilte Techniken zur Stateful Inspection. Erklären Sie kurz die Verbindungszustände `NEW`, `ESTABLISHED` und `RELATED` der Zustandsmaschine von Netfilter/iptables. Folgender Link könnte Ihnen von Nutzen sein:

   `http://iptables-tutorial.frozentux.net/iptables-tutorial.html`

6. Sobald das netfilter Modul geladen ist, bietet der Kernel einige Optionen, welche den Betrieb als Firewall erleichtern bzw. die Sicherheit des Systemes erhöhen können. Diese Optionen sind in der Datei

   `/usr/src/linux/Documentation/networking/ip-sysctl.txt`

   beschrieben und können unter `/proc/sys/net/ipv4/` bzw. unter `/proc/sys/net/ipv4/conf/*` aktiviert werden. Wählen Sie ein paar für die Firewall interessanten Parameter aus und erklären Sie kurz ihren Zweck.

3.2 Versuchsaufbau

Wie schon im letzten Versuch existiert für jede Gruppe ein beinahe identischer Versuchsaufbau (hier nochmals der Versuchsaufbau [vgl. Abbildung 2.4] des vergangenen Versuchstages). Die Rechnernamen unterscheiden sich nur am durch „X“ gekennzeichneten Stelle, die für die Gruppennummer („1“ oder „2“) steht.

Der Rechner sollte zumindest am Anfang des Versuchstages neu gestartet werden, um eine funktionierende Grundkonfiguration zu gewährleisten. Wie auch schon im letzten Versuch ist es für die Aufgaben notwendig, sich als `root` anzumelden. Für die weiteren Aufgaben soll der Versuchsaufbau wie in Abbildung 3.8 sein. Im Laufe des Versuchsnachmittages soll auf dem Rechner `pcfwX` mittels `netfilter/iptables` eine Packet-Filter-Firewall eingerichtet werden. Diese soll die Rechner `pcrtX`, `pcrtXsub1` und `pcrtXsub2` (das zu schützende Netz) vor einem ungewollten Zugriff aus dem RNP-Netz schützen, bzw. den gegenseitigen Zugriff regulieren.

Aufgrund der komplexen Gegebenheiten des RNP (Gewährleistung einer einheitlichen Basis für jede Praktikumsgruppe, Backup Strategien, Administrationsaufwand) können leider nicht alle Vorgaben hinsichtlich eines Minimalsystems verwirklicht werden. Die Versuche

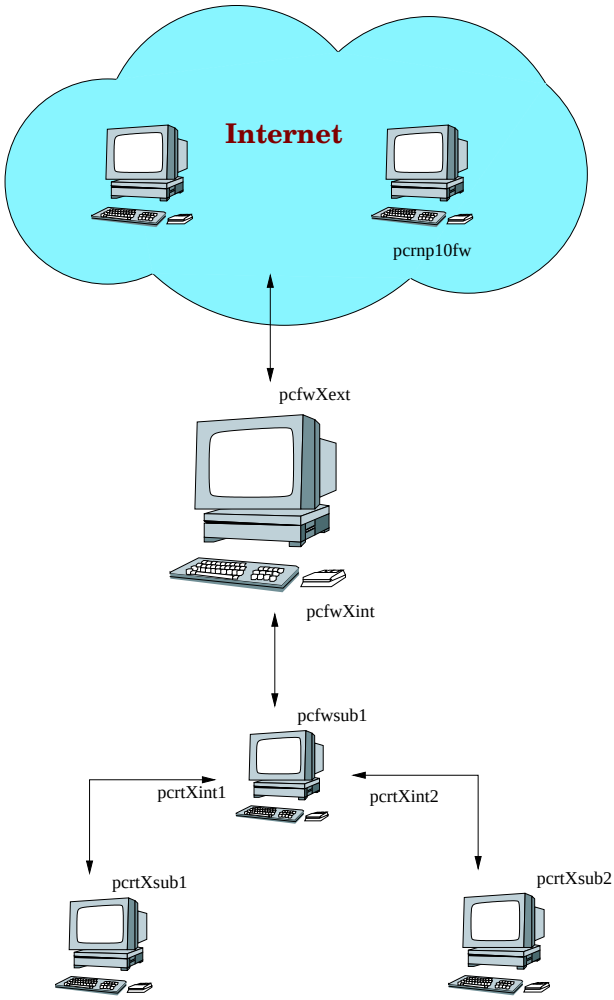

Abbildung 3.8: Versuchsaufbau

sollen deshalb nur eine Einführung in dieses Thema geben, da die Ausarbeitung und Implementierung eines vollständigen Firewall Konzepts den Rahmen dieses Praktikums sprengen würden.

3.3 Versuch I: Kontrolle der freigeschalteten Dienste

Wie schon erwähnt, kann im Praktikum leider kein Minimalsystem realisiert werden. Als erstes sollte man daher feststellen, welche Dienste laufen. Führen Sie zu diesem Zweck auf der Firewall `pcfwX` die Programmaufrufe `lsof -i`, `netstat -tu` und `netstat -lp` aus. Interpretieren Sie kurz die angezeigten Werte.

3.4 Versuch II: Statische Paketfilterung mit Netfilter

In den folgenden Aufgaben werden wir - Schritt für Schritt - einen Konfigurationsskript für unsere Paketfilterfirewall erstellen. Sie sollten dabei von dem rudimentären Skript `/tmp/firewall/firewall` ausgehen, weil es nämlich schon einige Definitionen enthält, die Ihnen nützlich sein könnten. Um vernünftig arbeiten zu können, sollten Sie dieses Skript ins Verzeichnis `/etc/init.d/` kopieren und überdies folgenden Link setzen:

```
ln -s ../../etc/init.d/firewall /usr/sbin/rcfirewall
```

Anschließend können Sie das Konfigurationsskript bequem mit dem Kommando `rcfirewall [start|stop|close]` ausführen.

1. Wir wollen von einer pessimistischen Vorgehensweise bei der Konfiguration unseres Paketfilters ausgehen. Das Rahmenskript sollte so erweitert werden, dass beim Aktivieren der Firewall (`rcconfig start`) vorerst jeglicher Datenverkehr verhindert wird und somit das interne Netz abgeschottet ist. Es bietet sich an, diese Regeln auch gleich in die hierfür vorgesehene Abteilung `close)` zu schreiben, welche genau für diesen Zweck (der Abschottung) angelegt wurde. Der Abschnitt unter `stop)` soll so erweitert werden, dass die Firewall abgeschaltet wird. In diesem Fall soll die Firewall wieder normal routen und alle Datenpakete akzeptieren.

2. Für die Verwendung von auf dem Rechner lokal installierten Diensten, wie z.B. DNS, NIS oder HTTP, steht normalerweise ein loopback Interface zu Verfügung (Auf einem reinen Paketfilter hätten dieses natürlich nichts verloren). Ein loopback Interface kann aber auch für Testzwecke (z.B. des Protokollstacks oder der obigen Dienste) nötig sein. Das firewall Skript soll nun so erweitert werden, dass bei aktivierter Firewall ein uneingeschränkter Zugriff auf das loopback Interface möglich ist.

3. Bevor in den folgenden Abschnitten der Paketfilter immer weiter geöffnet wird, um bestimmte Dienste anzubieten, sollte noch eine Einschränkung vorgenommen werden. Weil unser zu schützendes Netz einen festen IP Bereich besitzt, sollten alle Datenpakete, welche am externen Interface (`pcfwXext`) des Paketfilters einlaufen, darauf überprüft werden, ob ihre angegebene Quelladresse **außerhalb** des IP Bereichs des internen Netzes liegt. Einlaufende Pakete an `pcfw1ext`, deren Quell-IP-Adresse behauptet von einem internen Rechner zu stammen („gespoofte Adresse“), zeugen mit ziemlicher Sicherheit von einem Angriffsversuch (oder von einem falsch konfigurierten Rechner).

   Um die Realisation der Aufgabe einfach zu halten, demonstrieren wir das Antispoofing nicht am externen sondern am **internen** Interface (`pcfwXint`), indem wird umgekehrt fordern, dass die Quelladressen aller dort einlaufenden Pakete tatsächlich aus dem internen Netz stammen. Zum „internen Netz“ rechnen wir die Subnetze

192.168.215.80 und .96 (Gruppe 1) bzw. 192.168.215.144 und .160 (Gruppe 2). Wir stellen dabei uns auf den Standpunkt, dass die Subnetze 192.168.215.112 bzw. .176 **nicht** zu unserem „internen Netz“ gehören und der Host `pcrtXsub2` „feindlich“ ist und in unserem Netz eigentlich nichts zu suchen hat (mit ihm können wir dann bequem IP-Pakete mit „gespoofter“ Quelladresse senden).

Schreiben Sie also statische Firewallregeln, welche diese gefälschten Adressen protokollieren und anschließend verwerfen. Sorgen Sie dafür, dass die entsprechenden Meldungen unter `/var/log/messages` eindeutig auffindbar sind und überprüfen Sie dies, indem Sie sich diese Datei anzeigen lassen und nach entsprechenden Meldungen suchen. Verwenden Sie hierbei die Möglichkeit eigene (sub-) chains zu erstellen, um ihr Konfigurationsskript übersichtlich zu halten. Testen Sie die Antispoofing-Konfiguration ihrer Firewall. **Zur Überprüfung der Konfiguration steht auf dem Rechner das tool zu Verfügung. Beim Aufrufen dieses Programmes wird eine Befehlssyntax ausgegeben.**

4. Für administrative Zwecke ist das ICMP Protokoll eine große Hilfe. Schreiben Sie einen Regelsatz, welcher die häufigsten Dienste (`ping`, `traceroute`) in jeglicher Richtung erlaubt (auch wenn dies sicherheitstechnisch bedenklich ist, weil `ping` an Broadcastadressen zum Spionieren und für DoS-Attacken benutzt werden kann). Wenn genügend Zeit vorhanden ist, steht es ihnen natürlich frei, auch einen feingranulareren Filter zu entwerfen. Beachten Sie auch die Kernelparameter, welche Sie am Ende des Theorieteils untersucht haben. Vergessen sie nicht, ihre Regeln zu testen.

   Hinweis: Teilweise können sich hier die Funktionen der Kernelparamter mit denen der Regeln überschneiden. Dies ist bewusst so gewählt. Zum einen könnten Situationen auftreten in denen die Parameter nicht zu Verfügung stehen, zum anderen ist es schwierig die Korrektheit der Funktionsweise der Kernelparameter zu überprüfen.

Hinweis:

```
#!/bin/sh

# Rahmen des Konfigurationsskripts für Netfilter/iptables

IPTABLES="/usr/sbin/iptables"
DEVEXTERN="eth0"
DEVINTERN="eth1"

# Einschränkung auf pcfw1int (Gruppe 1)
NETZINTERN_A="192.168.215.80/28"
```

```
NETZINTERN_B="192.168.215.96/28
LO="127.0.0.1"

case "$1" in
  start)
    echo "Firewall wird aktiviert"

    exit 1
    ;;
  stop)
    echo "Firewall wird deaktiviert"
    # Rechner arbeitet als normaler Router

    exit1
    ;;
  close)
    echo "Firewall abschotten"
    # Rechner routet nicht und blockt jeden Datenverkehr

    exit1
    ;;
  *)
    echo usage: $0 start|stop|close"
    exit 1
esac

exit 0
```

3.5 Versuch III: Dynamische Paketfilterung mit Netfilter

Netfilter/`iptables` bietet mit dem Modul `ip_state` die Möglichkeit des „connection tracking“ (stateful inspection). Hiermit werden die Zustände aller Kommunikationsverbindungen, welche durch die Firewall aktiv bestehen, mitverfolgt und es kann entsprechend darauf reagiert werden. Dies geschieht in der Datei `/proc/net/ip_conntrack` und funktioniert auch mit zustandslosen Verbindungen wie bei UDP oder ICMP. Verifizieren Sie dies, indem Sie sich diese Datei während eines laufenden Pings anzeigen lassen. Falls keine Verbindung angezeigt wurde, führen Sie einen Ping an eine nicht existierende Adresse durch, um eine noch nicht beendete Verbindung zu erhalten. Wiederholen Sie dies später, wenn weitere Verbindungen möglich sind.

1. Wie schon erwähnt, sollten auf einem reinen Paketfilter eigentlich keine Dienste laufen. Evtl. ist es aber manchmal nötig, die Firewall aus der Ferne zu administrieren. Richten Sie zu diesem Zweck eine Regel ein, die es erlaubt, aus dem internen Netz mittels SSH auf den Paketfilter zuzugreifen. Machen Sie dabei so viele Einschränkungen wie möglich und achten sie darauf, dass weder vom externen Netz, noch vom Firewall Rechner selber SSH Verbindungen in das interne Netz zulässig sind.

2. Nun wird es Zeit, dass wir den Zugriff aus dem inneren Netz auf das äußere erlauben. Schreiben Sie eine einfache Regel, die alle TCP/UDP Verbindungen aus dem internen Netz und die zugehörigen Rückverbindungen ins Netz zulässt. Beachten Sie den korrekten Aufbau einer neuen TCP Verbindung (siehe Theorieaufgabe). Sorgen Sie dafür, dass nur solche Pakete eine neue TCP Verbindung aufbauen dürfen und alle „nicht korrekten“ TCP Verbindungsversuche verworfen werden.

3. Die vorhergegangene Regel geht natürlich davon aus, dass sich die Benutzer einigermaßen vernünftig verhalten und keine unerlaubte Software ausführen. Hier ist zum einen bösartiger Code zu nennen, der sowohl von internen Rechnern Angriffe auf andere Netze ausführen kann, als auch Verbindungen zu fremden Rechnern öffnet und interne Daten verschicken kann. Zum anderen sind hier aber auch file sharing Programme zu nennen, deren Verwendung wahrscheinlich untersagt wurde.

   Schränken Sie deswegen jetzt den Zugang zum äußeren Netz auf WWW (http, https) und EMail (pop3, smtp) Dienste ein. Falls gewünscht können natürlich auch noch weitere Dienste wie in `/etc/services` beschrieben verwendet werden. Vergessen Sie nicht die Namensauflösung zu erlauben, da ohne sie ein normales Arbeiten kaum möglich ist.

.

3.6 Versuch IV: Firewall Builder (FWBUILDER)

FIREWALL BUILDER ist ein objetktorientiertes Frontend für die Erstellung von Firewall Skripts, welches unter anderem auch `iptables` unterstützt. Objektorientiert heißt in diesem Fall, dass alle Elemente, welche später in den Filterregeln Verwendung finden, durch Objekte repräsentiert werden. Diese Objekte müssen zunächst definiert werden.

1. Starten Sie den `fwbuilder` und machen Sie sich zunächst mit der Bedienung der GUI vertraut. Die GUI sollte eigentlich intuitiv benutzbar sein. Es steht aber auch ein Tutorial in gedruckter Form zu Verfügung, welches eine kurze Einführung bietet. Das wichtigste Objekt ist natürlich die Firewall selbst. Das Firewall-Objekt sollte als erstes erzeugt und so weit wie möglich konfiguriert werden (die Reiter **Sysinfo** und **compile/install** können so gelassen werden wie sie sind). Aus welchem Grund

bietet das Firewall Objekt zwei verschiedene Stellen, an denen Regeln erstellt werden können? Welche Auswirkungen hat die Einstellung **Assume firewall object is part of any** in den FIREWALL Optionen des Firewall Objekts?

2. Bevor nun überhaupt irgendwelche Regeln verwaltet werden können, müssen erst einmal alle benötigten Objekte erzeugt werden. Dies kann entweder von Hand gemacht werden, oder man benutzt das Tool **Discover Objects**. Probieren sie hier alle Möglichkeiten der Erkennung aus. Für die Erkennung mittels SNMP verwenden sie den community string „rnp“ oder „public“. Um Gruppen zu füllen, werden Objekte aus dem linken Objektbaum einfach in das offene Gruppen Fenster gezogen.

3. Erstellen Sie nun ein paar einfache Regeln und verwenden Sie auch den eingebauten **policy building druid** unter RULES/HELP ME BUILD FIREWALL POLICY. Wie sind diese Hilfsmittel zu bewerten? Es sollte mindestens ein **time** Object erstellt und in einer Regel verwendet werden. Dieses dient dazu, Filterregeln nur für einen bestimmten Zeitraum zu aktivieren.

   Bevor mittels RULES/COMPILE von `fwbuilder` ein `iptables`-Skript erzeugt werden kann, ist es ratsam, das Firewall Objekt zu sichern (z.B. in `/tmp/firewall/`). Das erzeugte Skript sollte nun genauer untersucht und die Unterschiede zwischen den Regeln im `fwbuilder` und dem dazu erzeugten Skript diskutiert werden. Was fällt ihnen auf?

4. Erstellen Sie mit dem `fwbuilder` (soweit möglich) nun noch einmal alle Regeln, die Sie oben per Hand erstellt haben. Benutzen Sie auch hier alle Möglichkeiten, welche ihnen `fwbuilder` bietet um einen übersichtlichen Aufbau zu gewährleisten. Auch hier sollte wieder eine Gegenüberstellung zwischen compilierten Regeln und dem von Ihnen per Hand erstellten Skript erfolgen. Sind die zwei Skripten vom Funktionsumfang äquivalent? Diskutieren Sie die Vor- und Nachteile einer GUI.

5. Um diesen Vergleich auch später noch nachvollziehen zu können ist es notwendig, auch die Regeln der GUI in die Ausarbeitung aufzunehmen. Tun Sie dies entweder von Hand in Form von Tabellen (wie in den Theorieaufgaben) oder durch Screenshots z.B. mittels `xv`.