

Berner Fachhochschule
Hochschule für Technik und Informatik HTI

Technische Dokumentation

Version 10.01.2005

Diplomarbeit I00 (2004)

MuSeGa



Mobile User Secure Gateway

Experte:	Andreas Dürsteler (Swisscom)
Betreuer:	Hansjürg Wenger (HTI) Gerhard Hassenstein (HTI)
Diplomand:	Lukas Reusser (Swisscom)



Abstract

In diesem Dokument werden alle technischen Aspekte von MuSeGa beleuchtet und die verschiedenen Zusammenhänge aufgezeigt.

Versionskontrolle

Version	Datum	Kommentar	Genehmigt
0.1	07.01.2005	Erster Draft	
0.2	08.01.2005	Erste vollständig erscheinende Version	
0.3	10.01.2005	Alles fertig!	Lu



Inhaltsverzeichnis

- 1 Allgemeines..... 7
 - 1.1 Sinn und Zweck..... 7
 - 1.2 Leserkreis..... 7
 - 1.3 Umfang..... 7
- 2 Hauptkomponenten..... 8
 - 2.1 Login Daemon (musega_logind.pl)..... 8
 - 2.1.1 Funktionsweise..... 8
 - 2.2 Statistik Daemon (musega_statsd.pl)..... 9
 - 2.2.1 Funktionsweise..... 9
 - 2.2.2 Performance..... 9
 - 2.3 RRD Daemon (musega_rrdd.pl)..... 9
 - 2.3.1 Funktionsweise..... 10
 - 2.4 Musega Control (musegactrl.pl)..... 11
 - 2.4.1 Funktionsweise..... 11
 - 2.5 SWITCHmobile ACL Gen (switchmobile_acl_gen.pl)..... 12
 - 2.5.1 Funktionsweise..... 12
 - 2.6 Admin-GUI (admin.pl)..... 12
 - 2.6.1 Funktionsweise..... 12
 - 2.7 Benutzer Anmeldemaske (login.pl)..... 13
 - 2.7.1 Funktionsweise..... 13
 - 2.8 PPTP Rechte-Checkscript (musega_checkrights)..... 14
 - 2.8.1 Funktionsweise..... 14
- 3 Firewall Konzept..... 15
 - 3.1 Objekttypen..... 15
 - 3.1.1 Statische Objekten..... 16
 - 3.1.2 Dynamische Objekte..... 16
 - 3.2 Regeltypen..... 16
 - 3.2.1 Statische Regeln..... 16
 - 3.2.2 Dynamische Regeln..... 16
- 4 Wichtige Abläufe..... 17
 - 4.1 Login..... 17
 - 4.2 Logout..... 18
 - 4.3 Regeln Kompilieren..... 18
- 5 Datenbank..... 19
 - 5.1 ERD..... 19
 - 5.2 Tabellen..... 20
 - 5.2.1 conf_dhcp_range..... 20
 - Verwendung..... 20
 - Schema..... 20
 - 20
 - 5.2.2 conf_dhcp_static..... 21
 - Verwendung..... 21
 - Schema..... 21
 - 5.2.3 conf_interfaces..... 21

- Verwendung..... 21
- Schema..... 21
- 5.2.4 conf_iptables..... 21
 - Verwendung..... 21
 - Schema..... 21
- 5.2.5 conf_radius..... 22
 - Verwendung..... 22
 - Schema..... 22
- 5.2.6 dhcp_leases..... 22
 - Verwendung..... 22
 - Schema..... 22
- 5.2.7 dubiously_clients..... 23
 - Verwendung..... 23
 - Schema..... 23
- 5.2.8 fw_active_redirect..... 23
 - Verwendung..... 23
 - Schema..... 23
- 5.2.9 fw_compiled_mangle_rule..... 23
 - Verwendung..... 23
 - Schema..... 23
- 5.2.10 fw_compiled_nat_rule..... 24
 - Verwendung..... 24
 - Schema..... 24
- 5.2.11 fw_compiled_rule..... 24
 - Verwendung..... 24
 - Schema..... 24
- 5.2.12 fw_dynamic_active_rule..... 25
 - Verwendung..... 25
 - Schema..... 25
- 5.2.13 fw_mangle..... 25
 - Verwendung..... 25
 - Schema..... 25
- 5.2.14 fw_nat_rule..... 25
 - Verwendung..... 25
 - Schema..... 25
- 5.2.15 fw_object_addressranges..... 26
 - Verwendung..... 26
 - Schema..... 26
- 5.2.16 fw_object_groups..... 26
 - Verwendung..... 26
 - Schema..... 26
- 5.2.17 fw_object_hosts..... 26
 - Verwendung..... 26
 - Schema..... 27
- 5.2.18 fw_object_networks..... 27
 - Verwendung..... 27
 - Schema..... 27
- 5.2.19 fw_rule..... 27



Verwendung.....	27
Schema.....	27
5.2.20 fw_service_groups.....	28
Verwendung.....	28
Schema.....	28
5.2.21 fw_service_icmp.....	28
Verwendung.....	28
Schema.....	28
5.2.22 fw_service_ip.....	28
Verwendung.....	28
Schema.....	28
5.2.23 fw_service_tcp.....	29
Verwendung.....	29
Schema.....	29
5.2.24 fw_service_udp.....	29
Verwendung.....	29
Schema.....	29
5.2.25 fw_time_timerange.....	30
Verwendung.....	30
Schema.....	30
5.2.26 fw_traffic_queue.....	30
Verwendung.....	30
Schema.....	30
5.2.27 group.....	31
Verwendung.....	31
Schema.....	31
5.2.28 mac_blacklist.....	31
Verwendung.....	31
Schema.....	31
5.2.29 mac_user_history.....	31
Verwendung.....	31
Schema.....	31
5.2.30 switchmobiel_acs.....	32
Verwendung.....	32
Schema.....	32
5.2.31 traffic_stats_services.....	32
Verwendung.....	32
Schema.....	32
5.2.32 traffic_throughput.....	32
Verwendung.....	32
Schema.....	33
5.2.33 user.....	33
Verwendung.....	33
Schema.....	33
6 MuSeGa Systeminformationen.....	33
6.1 Hardware Informationen.....	33
6.2 Betriebssystem Version.....	34
6.3 Kernel Version.....	34



6.4	Installierte Software.....	34
6.5	Selbst kompilierte Software.....	38
7	Glossar.....	38



1 Allgemeines

1.1 Sinn und Zweck

Der Sinn dieses Dokumentes ist es, die technische Funktionsweise von MuSeGa aufzuzeigen, ohne das man sich den Sourcecode anschauen muss. Alle wichtigen Abläufe werden hier beschrieben und erläutert.

1.2 Leserkreis

Dieses Dokument richtet sich an Entwickler, die an MuSeGa weiter arbeiten wollen und an den Betreiber eines Mobile User Secure Gateways, der mehr über die technischen Zusammenhänge erfahren will.

1.3 Umfang

In diesem Dokument werden nur technische Zusammenhänge und Abläufe aufgezeigt. Die Installation hingegen wird mit dem Installationshandbuch, die Konfiguration mit dem Betriebshandbuch abgedeckt.



2 Hauptkomponenten

2.1 Login Daemon (musega_logind.pl)

Der Login Daemon (im weiteren nur noch als logind bezeichnet) ist für das ein- und ausloggen der Benutzer zuständig. Seine interne Struktur verteilt sich auf drei Hauptaufgaben:

- Er überprüft die Datei `/var/lib/dhcp/dhcp.leases` auf Änderungen und reagiert falls nötig darauf.
- Er überprüft ob die Datei `/var/lib/musega/rules2update` existiert, und wendet die darin enthaltenen iptables Regeln an
- Er überprüft anhand der DHCP Leases, ob er einen Benutzer ausloggen muss.

Das Ganze folgt nun noch etwas detaillierter:

2.1.1 Funktionsweise

Wenn logind startet, überprüft er erst einmal ob er die Leases Datei lesen kann. Kann er dies nicht, wird das Programm sofort beendet

Anschliessend kommt er in eine endlose Schleife, wo er seine Arbeit verrichtet. Am Ende der Schleife wartet er eine Sekunde, bevor er seine Arbeit wieder von Vorne beginnt.

Anhand eines Zeitstempels prüft er nun, ob die Leases Datei verändert wurde.

Wurde sie verändert, liest er sie aus und vergleicht sie mit den bereits vorhandenen Einträgen in der MySQL Datenbank. Handelt es sich um einen neuen Eintrag, wird er in der Datenbank gespeichert und der Client auf die Loginseite weitergeleitet. Das Weiterleiten basiert auf iptables Regeln, welche den http Verkehr der vom Client kommt zum MuSeGa umleiten. Handelt es sich um einen bereits existierenden Eintrag (Renew), wird nur der Lease Zeitstempel in der Datenbank ergänzt. Ist der betreffende Client nicht eingeloggt, wird er auch in diesem Fall auf die Loginseite weitergeleitet.

Als nächstes wird überprüft ob die Datei rules2update existiert. Ist dies der Fall, werden die darin enthaltenen Firewallregeln aktiviert. Das hat folgenden Grund. Das Login wird ja mit der Datei login.pl gemacht, welche vom Benutzer aufgerufen wird. Dieses Script läuft mit den Rechten des Apache Webserver. Nach erfolgreicher Authentisierung weiss nun dieses Script, welche iptables Regeln für den eingeloggten Benutzer aktiviert werden müssen. Es kann aber diese Regeln nicht selber aktivieren, da es nicht mit Root-Rechten läuft. Nun hat aber der Apache Webserver Schreibrechte im Verzeichnis `/var/lib/musega`. Das Script schreibt dort nun die Regeln im Anfügemodus in die Datei rules2update, wo sie vom Login-Daemon aktiviert werden.

Den letzten Schritt der nun folgt, führt der logind nur jeden zehnten Durchgang aus. Also ca. alle 10 Sekunden wird überprüft ob ein Client seinen DHCP Lease nicht erneuert hat. Ist ein Lease endgültig abgelaufen, wird der zugehörige Benutzer ausgeloggt.



Hier ist das Ende der Schleife erreicht und der Vorgang beginnt von neuem.

2.2 Statistik Daemon (musega_statsd.pl)

Der Statistik Daemon (im weiteren nur noch als statsd bezeichnet) ist für das Sammeln von Informationen zuständig, aus denen später Statistiken erstellt werden können. Seine gesammelten Daten verwendet er nicht selbst, sondern er gibt sie an den RRD Daemon weiter.

2.2.1 Funktionsweise

Der statsd öffnet das Programm tcpdump und verwendet dessen Output als Input. Er extrahiert alle gebrauchten Informationen aus dem Output und legt diese in Variablen ab. Nun wird anhand der Adresse des Subnetzes geprüft, ob es sich um eingehenden oder ausgehenden Verkehr handelt. Anschliessend wird der Verkehr in einer mehrdimensionalen Hashtabelle abgespeichert. Hier ein kleiner Auszug wie die Hashtabelle organisiert ist.

```
hashmatrix{matrix}{sent}{$source}{$destination}{$protocol}{$dport}{packets}
hashmatrix{matrix}{sent}{$source}{$destination}{$protocol}{$dport}{bytes}
hashmatrix{matrix}{received}{$source}{$destination}{$protocol}{$dport}{packets}
hashmatrix{matrix}{received}{$source}{$destination}{$protocol}{$dport}{bytes}
```

Die in dieser Hashtabelle gespeicherten Informationen sind immens. Man weiss wie viel Pakete und Bytes von jeder Source nach jeder Destination geflossen sind. Zudem weiss man noch über welches Protokoll und über welchen Port.

Alle 60 Sekunden werden diese Informationen über eine Prozess übergreifende Variable dem RRD Daemon übermittelt. Die Hashtabelle wird wieder zurückgesetzt und die neuen Daten werden gesammelt.

2.2.2 Performance

Der statsd ist derjenige Prozess auf dem Gateway, der mit Abstand die meiste CPU Zeit in Anspruch nimmt. Auf den ersten Blick ist vielleicht nicht ganz klar wieso das so ist, aber wenn man seine Funktion etwas genauer anschaut, wird alles klar:

Er schaut sich jedes eingehende Paket an, und speichert Teile dessen Eigenschaften in einer Hashtabelle ab. Nichts sehr spektakuläres. Wenn man jetzt aber bedenkt das bei einem 100MBit/s FTP-Download um die 9'000 Pakete pro Sekunde durch den Gateway wandern, kriegt man langsam eine Ahnung davon wie viel das zu tun gibt bei jedem Paket noch schnell den Header auszuwerten und diese Daten abzuspeichern. Auf einem Gigabit Ethernet Interface kann es unter Last locker auf 30'000 – 40'000 Pakete/s kommen.

2.3 RRD Daemon (musega_rrdd.pl)

Wenn der statsd zu stark ausgelastet ist, kann er nicht mehr alle eingehenden Pakete auswerten. Aus diesem Grund habe ich das Erstellen von Statistiken in einen zweiten Prozess ausgelagert.



Der RRD Daemon (im folgenden nur noch rrrd genannt) nimmt also die gesammelten Daten vom statsd entgegen und erstellt daraus die Statistiken.

RRD kommt von Round Robin Database, eine spezielle Datenstruktur zur Ablage von Statistiken. Weitere Informationen dazu gibts auf der RRD Homepage:
<http://people.ee.ethz.ch/~oetiker/webtools/rrdtool/>

Ursprünglich war rrrd nur dazu gedacht, um eben diese RRD Dateien zu erstellen. Mittlerweile macht er noch mehr, der Name ist aber geblieben.

2.3.1 Funktionsweise

In einer Endlosschleife überprüft rrrd jede Sekunde, ob sich ein Wert in der Prozess übergreifenden Variable befindet. Ist dies der Fall, kopiert er den Wert in eine lokale Variable und setzt die original Variable wieder zurück. Nun beginnt die eigentliche Arbeit von rrrd:

Er erstellt für jeden aktiven Benutzer oder IP-Adressen im WLAN eine RRD Statistik von dessen Bytes/s und dessen Pakets/s die er/sie gesendet und empfangen hat. Weiter wird der durchschnittliche Durchsatz dieser Zeitperiode errechnet und in die Datenbank gespeichert.

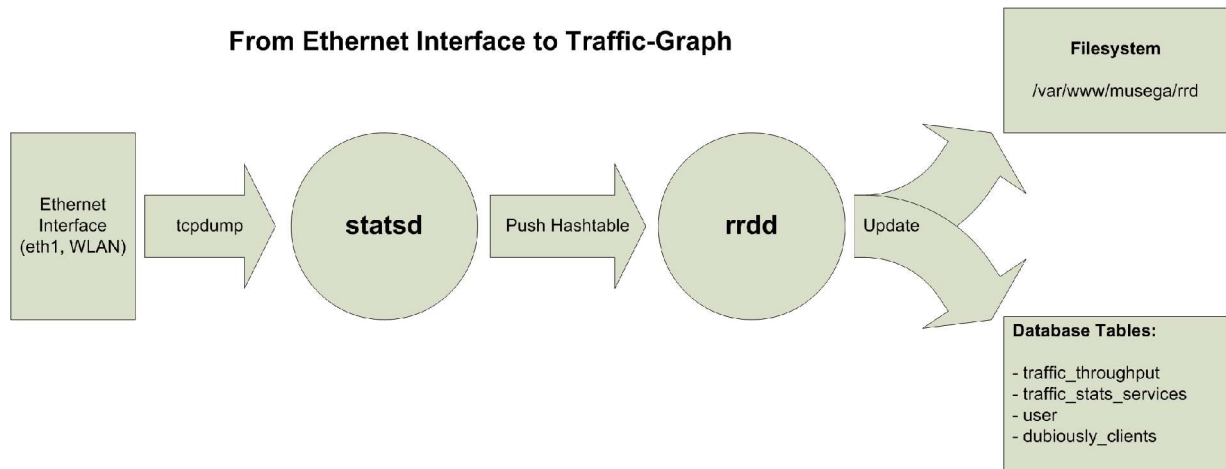
Genau das Selbe wie für die Benutzer wird auf für jeden einzelnen Service (Protokoll + Destinationport) gemacht. So kriegt man eine Übersicht welche Services am meisten gebraucht werden.

Alle erstellten Statistiken bieten eine Übersicht der letzten zwei Jahre.

In einem letzten Schritt wird noch überprüft, zu wie vielen Destinationen ein Benutzer eine Verbindung aufgebaut hat. Übersteigt diese Anzahl einen gewissen Schwellwert (Default 60 pro Minute), wird ein Eintrag in die Tabelle „Curiously_Clients“ gemacht und der betreffende Benutzer ausgeloggt.

Nun ist der Daemon fertig und wartet auf neuen Input.

Folgende Grafik stellt den ganzen Statistik Prozess grafisch dar:



(Abbildung 001, Vom Netzwerk zur Statistik)

Wie aus der Grafik zu entnehmen ist, schreibt der rrd gewisse Werte in die Datenbank sowie die RRD-Daten in die RRD-Dateien.

Datenbank Tabellen:

- traffic_throughput: Hier wird der Datendurchsatz der letzten Zeitperiode gespeichert: Input und Output, sowohl Bytes/s als auch Pakets/s
- traffic_stats_services: Hier wird die totale Anzahl Bytes und Pakete für jeden einzelnen Service (Protokoll + Destination-Port), sowohl Input und Output gespeichert. Zusätzlich wird noch für jeden Service der Durchsatz (Bytes/s, Pakete/s) aus der letzten Zeitperiode gespeichert. Mit dieser Tabelle ist es also möglich herauszufinden, welcher Service zur Zeit gerade am stärksten genutzt wird.
- user: Hier wird die totale Anzahl gesendeter und empfangener Pakete sowie die Anzahl Bytes gespeichert.
- dubiously_clients: Falls ein Client negativ auffällt, wird ein Eintrag in dieser Tabelle gemacht.

2.4 Musega Control (musegactrl.pl)

Um alle Daemons gleichzeitig starten und stoppen zu können, habe ich musegactrl.pl geschrieben. Musegactrl.pl unterstützt die Parameter: start, stop, status und reload.

2.4.1 Funktionsweise

Start:

Zuerst wird der Status jedes Benutzers auf offline gesetzt.
Dann wird die Tabelle der aktiven DHCP Leases geleert.
Anschliessend wird auch die Tabelle der aktiven, dynamischen Regeln geleert.
Zuletzt wird auch noch die Tabelle der aktiven Redirects geleert

Nun wird das Firewall-Script aktiviert: `/etc/musega/iptables.fw`



Danach werden in folgender Reihenfolge alle Daemons gestartet:
logind, statsd, rrrd

Beim Starten wird für jeden Daemon anhand einer Pid-Datei überprüft, ob der Prozess bereits läuft. Läuft eine alte Instanz, wird dieser zuerst beendet. Anschliessend wird der neue Prozess gestartet

Stop:

Beim Stoppen werden einfach alle laufenden Daemons anhand der Pid-Dateien beendet.

Status:

Anhand der Pid-Dateien wird überprüft ob die jeweiligen Prozesse laufen und dann wird deren Status ausgegeben.

Reload:

Reload lädt das Firewall-Script neu und anschliessend alle vorher aktiven, dynamischen Regeln. Diese Regeln findet es in der Tabelle „fw_dynamic_active_rule“.

2.5 SWITCHmobile ACL Gen (switchmobile_acl_gen.pl)

Dieses Script überprüft auf der Switch Homepage, ob neue ACLs verfügbar sind und lädt diese gegebenenfalls herunter. Anschliessend generiert es daraus iptables Regeln, fügt sie in die Konfiguration ein und macht ein musegactrl reload.

2.5.1 Funktionsweise

Zuerst wird die ACL Versionsnummer auf der Switch Homepage mit der lokal gespeicherten Versionsnummer verglichen. Gibt es bei Switch eine neuere Version, wird das Update ausgelöst, ansonsten schreibt der Prozess die aktuelle Uhrzeit in die Datenbank und beendet sich. Gibt es aber nun neue ACLs, so werden diese herunter geladen und daraus iptables Regeln generiert. Diese Regeln werden dann in die „switchmobile_acl“ Sektion in der Tabelle „conf_iptables“ kopiert. Nun wird eine neue Version der Datei /etc/musega/iptables.fw generiert und ins Dateisystem geschrieben. Schlussendlich wird noch ein musegactrl reload ausgeführt und die ACLs sind wieder auf dem neusten Stand.

2.6 Admin-GUI (admin.pl)

Admin.pl ist die Datei, die sich hinter dem Administrations-Interface versteckt. Wie sie das Admin-GUI bedienen, entnehmen sie bitte aus dem Betriebshandbuch.

2.6.1 Funktionsweise

Das Script admin.pl arbeitet in mehreren Schritten:

1. Die Query-Parameter werden entgegen genommen und in globalen Variablen abgelegt
2. Query-Parameter die als Array daher kommen brauchen noch eine spezielle Behandlung; dies wird jetzt gemacht.



3. Der HTML Header wird geschrieben
4. Eine Datenbank Verbindung wird hergestellt
5. Anhand der Query-Parameter werden diverse Datenbank Updates durchgeführt
6. Das Hauptmenü oben auf der Seite wird geschrieben
7. Je nach gewähltem Menü (Query-Parameter) wird der entsprechende Inhalt ausgegeben
8. Die Datenbank Verbindung wird geschlossen
9. Die Fussnote wird geschrieben

Je nachdem in welchem Menü man sich befindet und was man gerade macht, werden andere Funktionen aufgerufen. Es würde zu weit führen alle Funktionen zu erläutern. Wenn sie genau wissen wollen wann was passiert, schauen sie sich den Sourcecode von admin.pl an. Darin ist sehr schnell ersichtlich welche Funktionen in welchem Menü ausgeführt werden.

Einige wichtige Abläufe werden weiter unten unter „Wichtige Abläufe“ beschrieben.

2.7 Benutzer Anmeldemaske (login.pl)

Hinter der Anmeldemaske für die Benutzer verbirgt sich die Datei login.pl. Die Aufgabe dieses Scriptes ist es, den Benutzernamen sowie das Passwort des Benutzers entgegen zu nehmen und diese Werte zu überprüfen.

2.7.1 Funktionsweise

Zuerst werden die Query-Parameter entgegen genommen und in globalen Variablen abgelegt. Anschliessend wird anhand der IP-Adresse, die MAC-Adresse des Clients ermittelt

Falls sich die MAC-Adresse des Clients in der Blacklist befindet, kriegt der Benutzer bereits jetzt eine Fehlermeldung.

Nun wird anhand der IP- und MAC-Adresse überprüft, ob der entsprechende Client bereits eingeloggt ist.

Ist dies der Fall, wird als nächstes überprüft, ob der Benutzer auf den Logout Knopf gedrückt hat. Ist dies so wird der Benutzer ausgeloggt, andern Falls werden die Statusinformationen des Benutzers ausgegeben.

Ist aber der Client noch nicht eingeloggt, wird nun dessen Benutzernamen und dessen Passwort überprüft.

Sind diese beiden Werte nicht definiert, wird die Login-Maske ausgegeben und das Script ist am Ende angekommen.

Sind Benutzername und Passwort definiert, werden sie überprüft. Dies geschieht entweder via RADIUS oder über die lokale Benutzerdatenbank. Je nachdem was der Benutzer ausgewählt hat.

Wurde RADIUS gewählt, bekommt das Script die Gruppenzugehörigkeit vom RADIUS Server mit Hilfe des Class Attributes. Andernfalls wird die Gruppenzugehörigkeit aus der lokalen Benutzerdatenbank entnommen.

War die Authentisierung des Benutzers erfolgreich, wird mit dem Login fortgefahren, andernfalls erhält er eine Fehlermeldung.

Falls für den Benutzer nun eine Nachricht wartet, wird diese ausgegeben. Ansonsten wird der Client auf die ursprünglich eingegebene URL weitergeleitet.



2.8 PPTP Rechte-Checkscript (musega_checkrights)

Das Script `musega_checkrights` befindet sich im Verzeichnis `/etc/ppp/ip-up.d/`. Alle Scripts in diesem Verzeichnis werden gestartet, sobald jemand eine pptp Verbindung erstellt. In unserem Fall dient es dazu, zu überprüfen ob jemand berechtigt ist sich via pptp am Gateway anzumelden. Dazu geht es wie folgt vor:

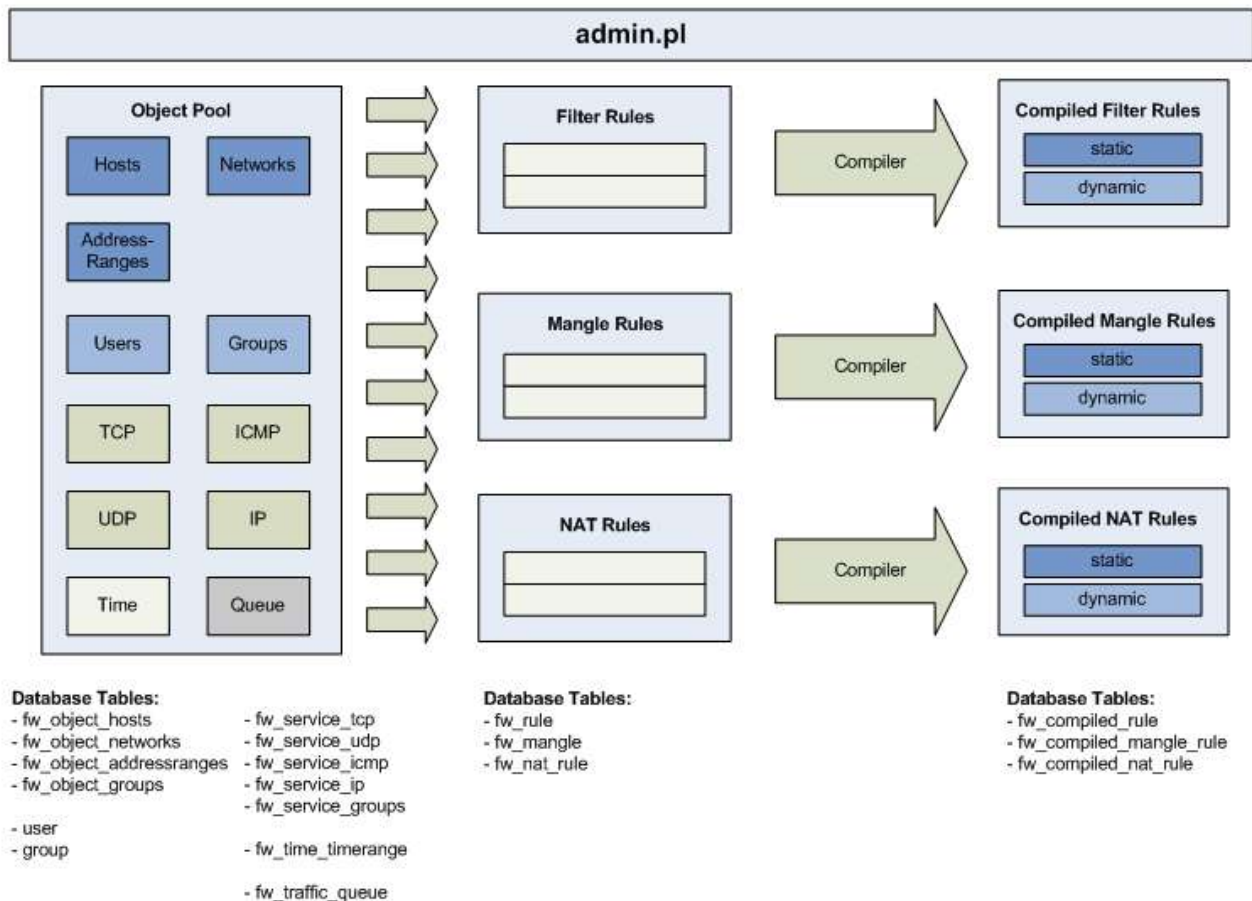
2.8.1 Funktionsweise

Laut Definition sind nur Benutzer berechtigt via pptp einzuloggen, bei denen das RADIUS Class Attribut auf „_pptp“ oder „pptp;“ endet. Dank dem `radattr` Plugin vom ppp Daemon werden nun beim Verbindungsaufbau alle Attribute die der RADIUS-Server zurück gibt in der Datei `/var/run/radattr.INTERFACE` gespeichert. Das Script liest also diese Datei und sucht nach dem Class Attribut. Nun prüft es ob die Endung mit der Definition übereinstimmt. Ist alles OK beendet sich das Script. Andernfalls wird mit Hilfe der Pid-Datei der zu dieser Verbindung gehörende Prozess gestoppt und somit wird die Verbindung aus Sicht des Benutzers sofort abgebrochen.



3 Firewall Konzept

Wie bereits erwähnt, arbeite ich bei den Firewall-Regeln mit Objekten. Folgende Grafik zeigt einen sehr schönen Überblick über alle Objekte, wo das sie gebraucht werden, sowie in welchen Datenbank Tabellen sie abgespeichert werden:



(Abbildung 002, Firewall Regeln erstellen, Übersicht)

Alle Objekte im Object Pool können also benutzt werden um die verschiedenen Regeltypen zu erstellen. Es sind auch Gruppen von Objekten möglich. Eine fertige Regel hat im Prinzip nichts anderes als eine Liste von Verweisen auf verschiedene Objekte gespeichert.

Kompiliert man nun die fertigen Regeln, wird der jeweilige Verweis auf ein Objekt mit dessen Daten ersetzt.

3.1 Objekttypen

Es gibt zwei Typen von Objekten, nämlich die statischen sowie die dynamischen.



3.1.1 Statische Objekten

Zu den statischen Objekten gehören alle diejenigen, welche einmal definiert werden und dann meistens für immer so bleiben. Das sind die folgenden: hosts, networks, addressranges, tcp, udp, icmp, ip sowie time und queue.

3.1.2 Dynamische Objekte

Zu den dynamischen Objekten zählen wir user und group. Da ein Benutzer ziemlich sicher jedes mal wenn er sich anmeldet eine andere IP-Adresse erhält und er vielleicht auch einmal eine andere MAC-Adresse hat, kann man dem User Objekt keine fixen Adressen zuteilen. Anstelle der Adressen wird ein Platzhalter eingefügt. Dieser Platzhalter wird später wenn sich ein Benutzer anmeldet mit seiner aktuellen IP- und MAC Adresse ersetzt.

3.2 Regeltypen

Es gibt zwei Typen von Regeln. Die statischen sowie die dynamischen. Dies kommt natürlich von den dynamischen Objekten her.

3.2.1 Statische Regeln

Eine statische Regel besteht also nur aus statischen Objekten. Sie wird einmal kompiliert und ändert sich dann lange nicht mehr. Wenn sie kompiliert ist wird sie in die iptables Konfigurationsdatei geschrieben. Sie wird erst aktiv wenn das Skript neu gestartet wird (musegactrl start oder reload).

3.2.2 Dynamische Regeln

Eine dynamische Regel besteht aus dynamischen Objekten. Wenn sie kompiliert ist, wird sie nicht in die iptables Konfigurationsdatei geschrieben, sondern in der Tabelle der kompilierten Regeln belassen. Meldet sich ein Benutzer an MuSeGa an, wird nun nach den Regeln für diesen Benutzer gesucht. Wird sie gefunden, werden die Platzhalter in der Regel mit der aktuellen IP- und MAC-Adresse des Clients ersetzt und die Regel aktiviert. Dynamische, kompilierte Regeln sind also sofort aktiv, wenn sich der Benutzer neu anmeldet.



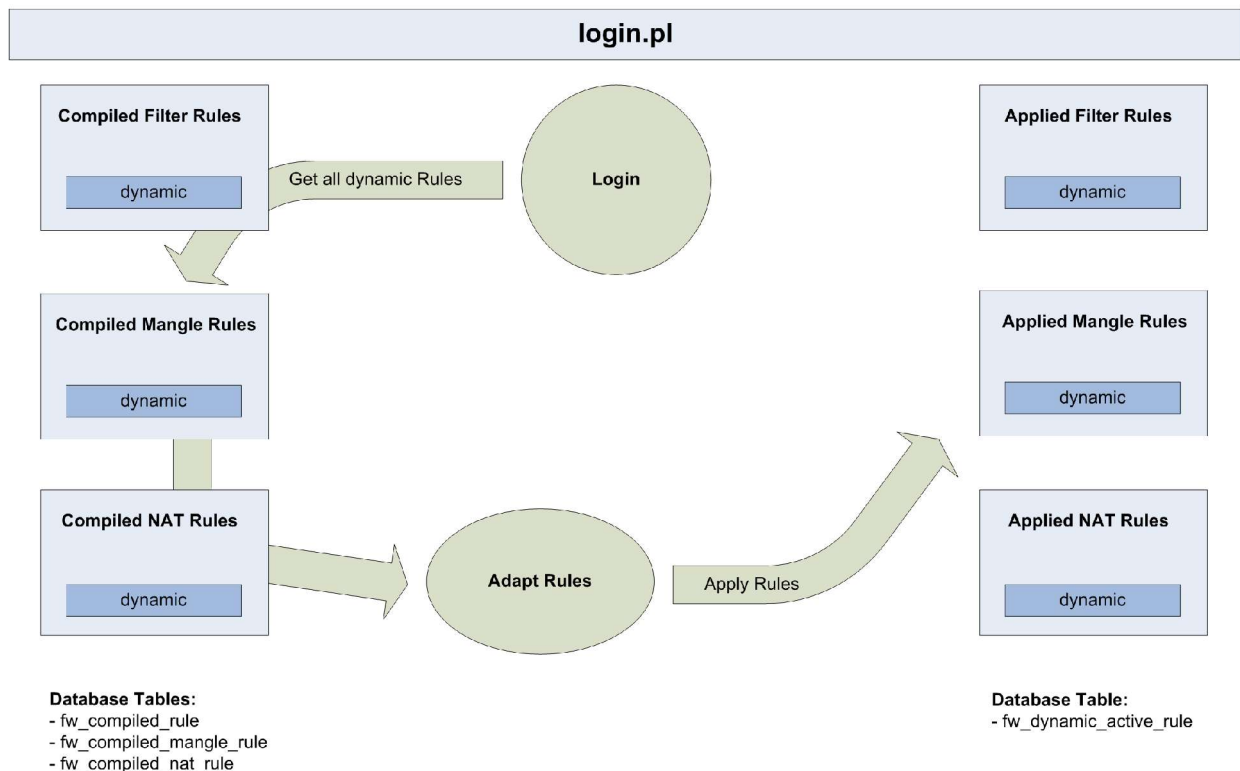
4 Wichtige Abläufe

4.1 Login

Einer der wichtigsten Abläufe bildet sicher das Login. Im folgenden werde ich nun erläutern wie der ganze Prozess von dem Zeitpunkt an wo ein Client eingeschaltet wird bis er erfolgreich im Internet surfen kann, vor sich geht.

- Ein Benutzer startet seinen Notebook und bootet sein Betriebssystem
- Sein Rechner schickt einen DHCP Request.
- Der DHCP Daemon auf MuSeGa beantwortet den Request und teilt eine IP-Adresse zu. Diesen Lease schreibt der Daemon in die Leases Datei.
- Der logind bemerkt, das es eine Änderung in der Leases Datei gegeben hat und er überprüft sofort was zu tun ist.
- Er bemerkt das es sich um einen neuen Client handelt und er wendet die nötigen iptables Regeln an um den Client zur Loginmaske umzuleiten.
- In der Zwischenzeit ist der Benutzer nun soweit um einen Webbrowser zu starten
- Der Webbrowser des Benutzers wird auf die Loginseite von MuSeGa umgeleitet, die ursprünglich eingegebene URL wird zwischengespeichert.
- Der Benutzer gibt seinen Benutzernamen und sein Passwort ein.
- Nun wird vom Gateway einen Authentication Request an den RADIUS-Server geschickt, mit den angaben des Benutzers.
- Die Antwort vom RADIUS-Server erhält nun ein Class Attribut mit der Gruppenzuteilung des Benutzers.
- Nun wird in den drei Tabellen der kompilierten Regeln, nach Regeln für diesen Benutzer oder dessen Gruppe gesucht.
- Die Platzhalter der gefundenen Regeln werden dann mit IP- und MAC Adresse des Clients ersetzt, in die Tabelle „fw_dynamic_active_rule“ eingetragen und in die Datei rules2update geschrieben. Dies ist so gelöst weil der Login Prozess keine Rechte hat um iptables Regeln direkt anzuwenden.
- Der logind bemerkt nun, dass neue Regeln zum Anwenden eingetroffen sind und er wendet diese Regeln am System an. Die Regeln sind nun sofort aktiv.
- Der Client wird nun mittels Redirect auf die ursprünglich eingegebene URL weiter geleitet.
- Der Benutzer kann nun alles machen was seine ihm zugeteilten Regeln zulassen

Folgende Grafik veranschaulicht wie das mit den Firewall-Regeln während dem Login vor sich geht:



(Abbildung 003, Firewallregeln während dem Login)

4.2 Logout

Das Logout geht wie folgt vor sich:

Meldet sich ein Benutzer ab, oder wird er vom System oder vom Administrator ausgeloggt, wird anhand seiner IP- und MAC Adresse nach den dynamischen, aktiven Regeln für diesen Benutzer gesucht. Dafür wird die Tabelle `fw_dynamic_active_rule` verwendet. Bei allen gefundenen Regeln wird im Regeltext nun das `--insert` mit einem `--delete` ersetzt, und die modifizierten Regeln erneut angewendet. Die Regeln sind nun so vom System gelöscht. Anschliessend werden diese Regeln auch aus dieser Tabelle entfernt. Nun wird noch die Benutzerdatenbank aktualisiert. Dort wird nämlich das Feld `online` auf 'no' gesetzt. Schlussendlich werden wieder iptables Regeln gesetzt, die den Client wieder auf die Loginseite umleiten.

4.3 Regeln Kompilieren

Der Regel-Compiler ist ein Herzstück von MuSeGa. Wenn man im Admin-GUI auf `Compile_Rule` klickt, passiert ungefähr folgendes:

Zuerst werden alle in den Regeln verwendeten Objekte vorbereitet, das bedeutet dass nun alle Objekte kompiliert werden. Die kompilierten Objekte befinden sich nun in lokalen Hashtabellen:



```
my %hosts           = fwCompileHosts($dbh);
my %networks        = fwCompileNetworks($dbh);
my %addressranges   = fwCompileAddressRanges($dbh);
my %ogroups         = dbGetObjectGroupHash($dbh);
my %sgroups         = dbGetServiceGroupHash($dbh);
my %tcp             = fwCompileTCP($dbh);
my %udp             = fwCompileUDP($dbh);
my %icmp            = fwCompileICMP($dbh);
my %ip              = fwCompileIP($dbh);
my %timerange       = fwCompileTimeRange($dbh);
my %interfaces      = dbGetInterfaceAddresses($dbh);
```

(Codeauszug aus der Funktion fwCompileAllRules())

Nun werden alle alten Regeln aus der Tabelle der kompilierten Regeln gelöscht.

Anschliessend wird über die Liste der zu kompilierenden Regeln iteriert.

Zuerst wird jeder Verweis auf ein Objekt mit dessen Inhalt ersetzt und in eine Liste eingefügt. Für jedes Objektfeld in der Regel wird also eine Liste erstellt. In diesem Fall sind das also eine Liste für source, destination, service und timerange.

Sind alle Listen abgefüllt, wird für jede Liste eine Schleife durchlaufen:

```
foreach my $source (@sources) {
    foreach my $destination (@destinations) {
        foreach my $service (@services) {
            foreach my $timerange (@timeranges) {
```

(Codeauszug aus der Funktion fwCompileAllRules())

Nun werden noch diverse Entscheidungen getroffen. Zum Beispiel wird überprüft ob es sich um eine dynamische oder eine statische Regel handelt.

Schlussendlich erhält man eine syntaktische korrekte iptables Regel.

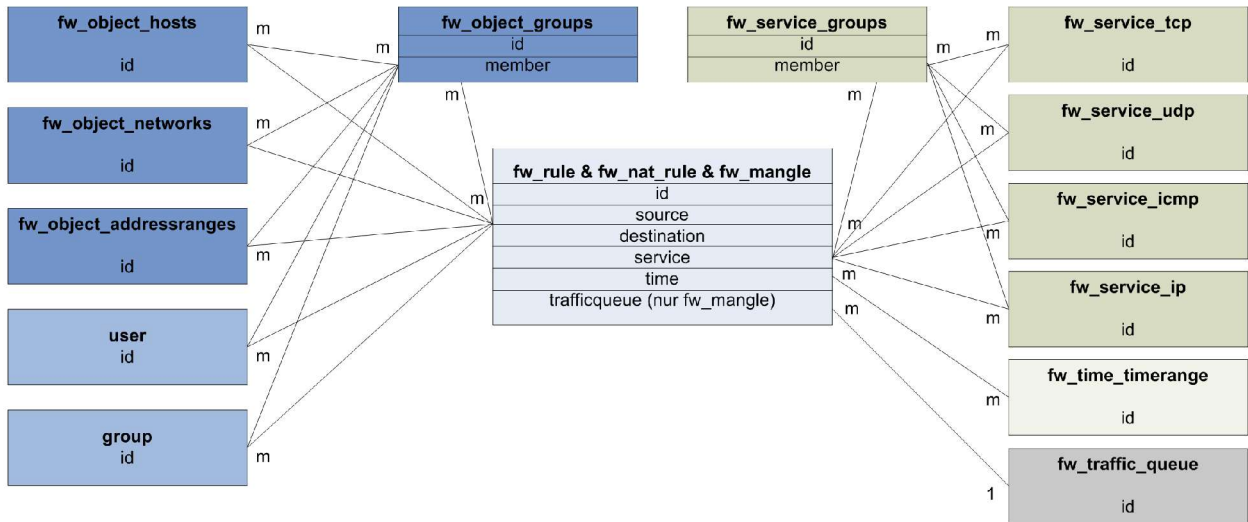
Diese Regeln werden dann in die Tabelle der kompilierten Regeln eingefügt.

5 Datenbank

5.1 ERD

In der folgenden Grafik nicht aufgeführte Tabellen werden nur zur Speicherung von Daten verwendet und haben keine weiteren Beziehungen zu anderen Tabellen.

(Abbildung 004, ERD)



5.2 Tabellen

5.2.1 conf_dhcp_range

Verwendung

Diese Tabelle enthält die Konfiguration der DHCP Ranges

Schema

```

-----

--
-- Table structure for table `conf_dhcpd_range`
--
CREATE TABLE `conf_dhcpd_range` (
  `id` int(10) unsigned NOT NULL auto_increment,
  `subnet` varchar(45) NOT NULL default '',
  `netmask` varchar(15) NOT NULL default '',
  `range_start` varchar(45) NOT NULL default '',
  `range_end` varchar(45) NOT NULL default '',
  `o_domain_name_server` varchar(80) NOT NULL default '',
  `o_domain_name` varchar(30) NOT NULL default '',
  `o_routers` varchar(80) NOT NULL default '',
  `o_subnet_mask` varchar(15) NOT NULL default '',
  `o_broadcast_address` varchar(45) NOT NULL default '',
  `default_lease_time` int(10) unsigned NOT NULL default '0',
  `max_lease_time` int(10) unsigned NOT NULL default '0',
  `comment` varchar(100) NOT NULL default '',
  PRIMARY KEY (`id`)
) TYPE=MyISAM AUTO_INCREMENT=8 ;
    
```

5.2.2 conf_dhcp_static

Verwendung

Diese Tabelle enthält die Konfiguration der statischen DHCP Einträge



Schema

```

-----
--
-- Table structure for table `conf_dhcpd_static`
--
CREATE TABLE `conf_dhcpd_static` (
  `id` int(10) unsigned NOT NULL auto_increment,
  `hostname` varchar(50) NOT NULL default '',
  `ethernet_address` varchar(17) NOT NULL default '',
  `fixed_address` varchar(45) NOT NULL default '',
  `comment` varchar(100) NOT NULL default '',
  PRIMARY KEY (`id`)
) TYPE=MyISAM AUTO_INCREMENT=6 ;

```

5.2.3 conf_interfaces

Verwendung

Diese Tabelle enthält die Interface Konfiguration.

Schema

```

-----
--
-- Table structure for table `conf_interfaces`
--
CREATE TABLE `conf_interfaces` (
  `id` int(10) unsigned NOT NULL auto_increment,
  `name` varchar(20) NOT NULL default '',
  `type` varchar(8) NOT NULL default '',
  `on_startup` char(3) NOT NULL default '',
  `dhcp` varchar(6) NOT NULL default '',
  `address` varchar(45) NOT NULL default '',
  `netmask` varchar(15) NOT NULL default '',
  `network` varchar(45) NOT NULL default '',
  `broadcast` varchar(45) NOT NULL default '',
  `gateway` varchar(45) NOT NULL default '',
  `vlan_raw_device` varchar(20) NOT NULL default '',
  `ip_version` varchar(5) NOT NULL default '',
  `comment` varchar(100) NOT NULL default '',
  PRIMARY KEY (`id`)
) TYPE=MyISAM AUTO_INCREMENT=7 ;

```

5.2.4 conf_iptables

Verwendung

Diese Tabelle enthält alle fertigen Konfigurationen im Zusammenhang mit der Firewall Konfiguration. Aus dieser Tabelle wird schlussendlich die Datei /etc/musega/firewall.fw generiert.

Schema

```

-----
--
-- Table structure for table `conf_iptables`
--
CREATE TABLE `conf_iptables` (
  `id` int(10) unsigned NOT NULL auto_increment,
  `created` datetime NOT NULL default '0000-00-00 00:00:00',
  `shell` varchar(30) NOT NULL default '',

```



```

`comment` text NOT NULL,
`path` text NOT NULL,
`functions` text NOT NULL,
`constants` text NOT NULL,
`interfaces` varchar(200) NOT NULL default '',
`tests` text NOT NULL,
`flags` text NOT NULL,
`default_policy` text NOT NULL,
`clear_chains` text NOT NULL,
`interface_labels` text NOT NULL,
`interface_addresses` text NOT NULL,
`modules` text NOT NULL,
`log` text NOT NULL,
`pre_rules` longtext NOT NULL,
`pre_userdef` longtext NOT NULL,
`switchmobile_acl` longtext NOT NULL,
`nat_rules` longtext NOT NULL,
`filter_rules` longtext NOT NULL,
`trafficcontrol` longtext NOT NULL,
`mangle_chains` longtext NOT NULL,
`mangle_rules` longtext NOT NULL,
`ip_forward` varchar(50) NOT NULL default '',
PRIMARY KEY (`id`)
) TYPE=MyISAM AUTO_INCREMENT=2 ;

```

5.2.5 conf_radius

Verwendung

Diese Tabelle enthält die RADIUS Konfiguration.

Schema

```

-----
--
-- Table structure for table `conf_radius`
--
CREATE TABLE `conf_radius` (
  `id` int(11) NOT NULL auto_increment,
  `server` varchar(45) NOT NULL default '',
  `secret` varchar(50) NOT NULL default '',
  `timeout` int(11) NOT NULL default '0',
  PRIMARY KEY (`id`)
) TYPE=MyISAM AUTO_INCREMENT=2 ;

```

5.2.6 dhcp_leases

Verwendung

Diese Tabelle enthält alle aktiven DHCP Leases.

Schema

```

-----
--
-- Table structure for table `dhcp_leases`
--
CREATE TABLE `dhcp_leases` (
  `id` int(11) NOT NULL auto_increment,
  `ip_address` varchar(45) NOT NULL default '',
  `mac` varchar(17) NOT NULL default '',
  `starts` datetime NOT NULL default '0000-00-00 00:00:00',
  `ends` datetime NOT NULL default '0000-00-00 00:00:00',
  `hostname` varchar(63) NOT NULL default '',
  PRIMARY KEY (`id`)
)

```



```
) TYPE=MyISAM AUTO_INCREMENT=2 ;
```

5.2.7 dubiously_clients

Verwendung

Diese Tabelle enthält alle negativ aufgefallenen Clients.

Schema

```
-----  
--  
-- Table structure for table `dubiously_clients`  
--  
CREATE TABLE `dubiously_clients` (  
  `id` int(10) unsigned NOT NULL auto_increment,  
  `ip_address` varchar(45) NOT NULL default '',  
  `username` varchar(30) NOT NULL default '',  
  `destinations` mediumtext NOT NULL,  
  `time` datetime NOT NULL default '0000-00-00 00:00:00',  
  PRIMARY KEY (`id`)  
) TYPE=MyISAM AUTO_INCREMENT=22 ;
```

5.2.8 fw_active_redirect

Verwendung

Diese Tabelle enthält alle aktiven Weiterleitungen der Clients zur Loginmaske.

Schema

```
-----  
--  
-- Table structure for table `fw_active_redirect`  
--  
CREATE TABLE `fw_active_redirect` (  
  `id` int(10) unsigned NOT NULL auto_increment,  
  `ip_address` char(45) NOT NULL default '',  
  `mac_address` char(17) NOT NULL default '',  
  `update` datetime NOT NULL default '0000-00-00 00:00:00',  
  PRIMARY KEY (`id`)  
) TYPE=MyISAM AUTO_INCREMENT=5 ;
```

5.2.9 fw_compiled_mangle_rule

Verwendung

Diese Tabelle enthält alle kompilierten Mangle Regeln die konfiguriert wurden.

Schema

```
-----  
--  
-- Table structure for table `fw_compiled_mangle_rule`  
--  
CREATE TABLE `fw_compiled_mangle_rule` (  
  `id` int(10) unsigned NOT NULL auto_increment,  
  `number` int(10) unsigned NOT NULL default '0',  
  `rule` text NOT NULL,  
  `description` varchar(200) NOT NULL default '',  
  `user` varchar(30) NOT NULL default '',
```



```
`group` varchar(30) NOT NULL default '',
`dynamic` char(3) NOT NULL default '',
PRIMARY KEY (`id`)
) TYPE=MyISAM AUTO_INCREMENT=1 ;
```

5.2.10 fw_compiled_nat_rule

Verwendung

Diese Tabelle enthält alle kompilierten NAT Regeln die konfiguriert wurden.

Schema

```
-----
--
-- Table structure for table `fw_compiled_nat_rule`
--
CREATE TABLE `fw_compiled_nat_rule` (
  `id` int(10) unsigned NOT NULL auto_increment,
  `number` int(10) unsigned NOT NULL default '0',
  `rule` text NOT NULL,
  `description` varchar(200) NOT NULL default '',
  `user` varchar(30) NOT NULL default '',
  `group` varchar(30) NOT NULL default '',
  `dynamic` tinyint(1) unsigned NOT NULL default '0',
  PRIMARY KEY (`id`)
) TYPE=MyISAM AUTO_INCREMENT=7 ;
```

5.2.11 fw_compiled_rule

Verwendung

Diese Tabelle enthält alle kompilierten Firewall Regeln die konfiguriert wurden.

Schema

```
-----
--
-- Table structure for table `fw_compiled_rule`
--
CREATE TABLE `fw_compiled_rule` (
  `id` int(10) unsigned NOT NULL auto_increment,
  `number` int(10) unsigned NOT NULL default '0',
  `intnumber` int(10) unsigned NOT NULL default '0',
  `rule` text NOT NULL,
  `description` varchar(200) NOT NULL default '',
  `user` varchar(30) NOT NULL default '',
  `group` varchar(30) NOT NULL default '',
  `dynamic` char(3) NOT NULL default '',
  PRIMARY KEY (`id`)
) TYPE=MyISAM AUTO_INCREMENT=38 ;
```

5.2.12 fw_dynamic_active_rule

Verwendung

Diese Tabelle enthält alle dynamischen Regeln, welche durch einen einloggenden Benutzer aktiviert wurden.

Schema

```
-----
```



```
--  
-- Table structure for table `fw_dynamic_active_rule`  
--  
CREATE TABLE `fw_dynamic_active_rule` (  
  `id` int(11) unsigned NOT NULL auto_increment,  
  `mac` varchar(17) NOT NULL default '',  
  `ip_address` varchar(45) NOT NULL default '',  
  `rule` text NOT NULL,  
  PRIMARY KEY (`id`)  
) TYPE=MyISAM AUTO_INCREMENT=10 ;
```

5.2.13 fw_mangle

Verwendung

Diese Tabelle enthält alle konfigurierten Mangle Regeln.

Schema

```
-- -----  
--  
-- Table structure for table `fw_mangle`  
--  
CREATE TABLE `fw_mangle` (  
  `id` int(10) unsigned NOT NULL auto_increment,  
  `number` smallint(5) unsigned NOT NULL default '0',  
  `source` text NOT NULL,  
  `destination` text NOT NULL,  
  `service` text NOT NULL,  
  `trafficqueue` varchar(20) NOT NULL default '',  
  `time` text NOT NULL,  
  `description` varchar(200) NOT NULL default '',  
  `enabled` tinyint(1) unsigned NOT NULL default '1',  
  PRIMARY KEY (`id`)  
) TYPE=MyISAM AUTO_INCREMENT=4 ;
```

5.2.14 fw_nat_rule

Verwendung

Diese Tabelle enthält alle konfigurierten NAT Regeln.

Schema

```
-- -----  
--  
-- Table structure for table `fw_nat_rule`  
--  
CREATE TABLE `fw_nat_rule` (  
  `id` int(10) unsigned NOT NULL auto_increment,  
  `number` smallint(5) unsigned NOT NULL default '0',  
  `original_source` text NOT NULL,  
  `original_destination` text NOT NULL,  
  `original_service` text NOT NULL,  
  `translated_source` text NOT NULL,  
  `translated_destination` text NOT NULL,  
  `translated_service` text NOT NULL,  
  `description` varchar(200) NOT NULL default '',  
  `enabled` tinyint(1) unsigned NOT NULL default '1',  
  PRIMARY KEY (`id`)  
) TYPE=MyISAM AUTO_INCREMENT=29 ;
```



5.2.15 fw_object_addressranges

Verwendung

Diese Tabelle enthält alle Objekte des Typs Addressrange.

Schema

```
-----  
--  
-- Table structure for table `fw_object_addressranges`  
--  
  
CREATE TABLE `fw_object_addressranges` (  
  `id` int(10) unsigned NOT NULL auto_increment,  
  `name` varchar(50) NOT NULL default '',  
  `start_address` varchar(45) NOT NULL default '',  
  `stop_address` varchar(45) NOT NULL default '',  
  `description` varchar(200) NOT NULL default '',  
  PRIMARY KEY (`id`)  
) TYPE=MyISAM AUTO_INCREMENT=6 ;
```

5.2.16 fw_object_groups

Verwendung

Diese Tabelle enthält alle Objekte des Typs Groups. Das sind also Gruppen von Objekten.

Schema

```
-----  
--  
-- Table structure for table `fw_object_groups`  
--  
  
CREATE TABLE `fw_object_groups` (  
  `id` int(10) unsigned NOT NULL auto_increment,  
  `name` varchar(50) NOT NULL default '',  
  `member` text NOT NULL,  
  `description` varchar(200) NOT NULL default '',  
  PRIMARY KEY (`id`)  
) TYPE=MyISAM AUTO_INCREMENT=5 ;
```

5.2.17 fw_object_hosts

Verwendung

Diese Tabelle enthält alle Objekte des Typs Host.

Schema

```
-----  
--  
-- Table structure for table `fw_object_hosts`  
--  
  
CREATE TABLE `fw_object_hosts` (  
  `id` int(10) unsigned NOT NULL auto_increment,  
  `name` varchar(50) NOT NULL default '',  
  `ip_address` varchar(45) NOT NULL default '',  
  `mac` varchar(17) NOT NULL default '',  
  `description` varchar(200) NOT NULL default '',
```



```
PRIMARY KEY (`id`)
) TYPE=MyISAM AUTO_INCREMENT=46 ;
```

5.2.18 fw_object_networks

Verwendung

Diese Tabelle enthält alle Objekte des Typs Network.

Schema

```
-----
--
--
-- Table structure for table `fw_object_networks`
--
CREATE TABLE `fw_object_networks` (
  `id` int(10) unsigned NOT NULL auto_increment,
  `name` varchar(50) NOT NULL default '',
  `address` varchar(45) NOT NULL default '',
  `netmask` varchar(15) NOT NULL default '',
  `description` varchar(200) NOT NULL default '',
  PRIMARY KEY (`id`)
) TYPE=MyISAM AUTO_INCREMENT=8 ;
```

5.2.19 fw_rule

Verwendung

Diese Tabelle enthält alle konfigurierten Filter Regeln.

Schema

```
-----
--
--
-- Table structure for table `fw_rule`
--
CREATE TABLE `fw_rule` (
  `id` int(10) unsigned NOT NULL auto_increment,
  `number` smallint(5) unsigned NOT NULL default '0',
  `source` text NOT NULL,
  `destination` text NOT NULL,
  `service` text NOT NULL,
  `action` varchar(6) NOT NULL default 'drop',
  `time` text NOT NULL,
  `log_prefix` varchar(30) NOT NULL default '',
  `log_level` varchar(30) NOT NULL default '',
  `stateful` tinyint(1) unsigned NOT NULL default '1',
  `description` varchar(200) NOT NULL default '',
  `enabled` tinyint(1) unsigned NOT NULL default '1',
  PRIMARY KEY (`id`)
) TYPE=MyISAM AUTO_INCREMENT=157 ;
```

5.2.20 fw_service_groups

Verwendung

Diese Tabelle enthält alle Objekte des Service-Gruppe. Das ist also eine Gruppe von Service Objekten.

Schema

```
-----
```



```
--  
-- Table structure for table `fw_service_groups`  
--  
  
CREATE TABLE `fw_service_groups` (  
  `id` int(10) unsigned NOT NULL auto_increment,  
  `name` varchar(50) NOT NULL default '',  
  `member` text NOT NULL,  
  `description` varchar(200) NOT NULL default '',  
  PRIMARY KEY (`id`)  
) TYPE=MyISAM AUTO_INCREMENT=3 ;
```

5.2.21 fw_service_icmp

Verwendung

Diese Tabelle enthält alle Objekte des Typs ICMP.

Schema

```
-----  
--  
-- Table structure for table `fw_service_icmp`  
--  
  
CREATE TABLE `fw_service_icmp` (  
  `id` int(10) unsigned NOT NULL auto_increment,  
  `name` varchar(50) NOT NULL default '',  
  `type` smallint(4) NOT NULL default '-1',  
  `type_desc` varchar(50) NOT NULL default '',  
  `code` smallint(4) NOT NULL default '-1',  
  `code_desc` varchar(50) NOT NULL default '',  
  `description` varchar(200) NOT NULL default '',  
  PRIMARY KEY (`id`)  
) TYPE=MyISAM AUTO_INCREMENT=7 ;
```

5.2.22 fw_service_ip

Verwendung

Diese Tabelle enthält alle Objekte des Typs IP.

Schema

```
-----  
--  
-- Table structure for table `fw_service_ip`  
--  
  
CREATE TABLE `fw_service_ip` (  
  `id` int(10) unsigned NOT NULL auto_increment,  
  `name` varchar(50) NOT NULL default '',  
  `protocol` tinyint(3) unsigned NOT NULL default '0',  
  `option` varchar(100) NOT NULL default '',  
  `description` varchar(200) NOT NULL default '',  
  PRIMARY KEY (`id`)  
) TYPE=MyISAM AUTO_INCREMENT=19 ;
```

5.2.23 fw_service_tcp

Verwendung

Diese Tabelle enthält alle Objekte des Typs TCP.



Schema

```

-----
--
-- Table structure for table `fw_service_tcp`
--
CREATE TABLE `fw_service_tcp` (
  `id` int(10) unsigned NOT NULL auto_increment,
  `name` varchar(50) NOT NULL default '',
  `source_port_start` smallint(5) unsigned NOT NULL default '1024',
  `source_port_end` smallint(5) unsigned NOT NULL default '65535',
  `destination_port_start` smallint(5) unsigned NOT NULL default '0',
  `destination_port_end` smallint(5) unsigned NOT NULL default '0',
  `examine_flag_urg` smallint(1) unsigned NOT NULL default '0',
  `examine_flag_ack` smallint(1) unsigned NOT NULL default '0',
  `examine_flag_psh` smallint(1) unsigned NOT NULL default '0',
  `examine_flag_rst` smallint(1) unsigned NOT NULL default '0',
  `examine_flag_syn` smallint(1) unsigned NOT NULL default '0',
  `examine_flag_fin` smallint(1) unsigned NOT NULL default '0',
  `flag_urg` smallint(1) unsigned NOT NULL default '0',
  `flag_ack` smallint(1) unsigned NOT NULL default '0',
  `flag_psh` smallint(1) unsigned NOT NULL default '0',
  `flag_rst` smallint(1) unsigned NOT NULL default '0',
  `flag_syn` smallint(1) unsigned NOT NULL default '0',
  `flag_fin` smallint(1) unsigned NOT NULL default '0',
  `description` varchar(200) NOT NULL default '',
  PRIMARY KEY (`id`)
) TYPE=MyISAM AUTO_INCREMENT=12 ;

```

5.2.24 fw_service_udp

Verwendung

Diese Tabelle enthält alle Objekte des Typs UDP.

Schema

```

-----
--
-- Table structure for table `fw_service_udp`
--
CREATE TABLE `fw_service_udp` (
  `id` int(10) unsigned NOT NULL auto_increment,
  `name` varchar(50) NOT NULL default '',
  `source_port_start` smallint(5) unsigned NOT NULL default '0',
  `source_port_end` smallint(5) unsigned NOT NULL default '0',
  `destination_port_start` smallint(5) unsigned NOT NULL default '0',
  `destination_port_end` smallint(5) unsigned NOT NULL default '0',
  `description` varchar(200) NOT NULL default '',
  PRIMARY KEY (`id`)
) TYPE=MyISAM AUTO_INCREMENT=7 ;

```

5.2.25 fw_time_timerange

Verwendung

Diese Tabelle enthält alle Objekte des Typs Timerange.

Schema

```

-----
--
-- Table structure for table `fw_time_timerange`
--

```



```
CREATE TABLE `fw_time_timerange` (
  `id` int(10) unsigned NOT NULL auto_increment,
  `name` varchar(50) NOT NULL default '',
  `start_minute` tinyint(2) NOT NULL default '-1',
  `start_hour` tinyint(2) NOT NULL default '-1',
  `start_day` tinyint(2) NOT NULL default '-1',
  `start_month` tinyint(2) NOT NULL default '-1',
  `start_year` smallint(4) NOT NULL default '-1',
  `stop_minute` tinyint(2) NOT NULL default '-1',
  `stop_hour` tinyint(2) NOT NULL default '-1',
  `stop_day` tinyint(2) NOT NULL default '-1',
  `stop_month` tinyint(2) NOT NULL default '-1',
  `stop_year` smallint(4) NOT NULL default '-1',
  `description` varchar(200) NOT NULL default '',
  `mon` tinyint(2) NOT NULL default '-1',
  `tue` tinyint(2) NOT NULL default '-1',
  `wed` tinyint(2) NOT NULL default '-1',
  `thu` tinyint(2) NOT NULL default '-1',
  `fri` tinyint(2) NOT NULL default '-1',
  `sat` tinyint(2) NOT NULL default '-1',
  `sun` tinyint(2) NOT NULL default '-1',
  PRIMARY KEY (`id`)
) TYPE=MyISAM AUTO_INCREMENT=4 ;
```

5.2.26 fw_traffic_queue

Verwendung

Diese Tabelle enthält alle Objekte des Typs Trafficqueue.

Schema

```
-----
--
-- Table structure for table `fw_traffic_queue`
--
CREATE TABLE `fw_traffic_queue` (
  `id` int(10) unsigned NOT NULL auto_increment,
  `name` varchar(20) NOT NULL default '',
  `comment` varchar(100) NOT NULL default '',
  `device` varchar(20) NOT NULL default '',
  `parent` varchar(20) NOT NULL default '',
  `class_id` varchar(20) NOT NULL default '',
  `queuetype` varchar(20) NOT NULL default '',
  `rate` varchar(20) NOT NULL default '',
  `ceil` varchar(20) NOT NULL default '',
  `prio` int(11) NOT NULL default '0',
  PRIMARY KEY (`id`)
) TYPE=MyISAM AUTO_INCREMENT=6 ;
```

5.2.27 group

Verwendung

Diese Tabelle enthält alle Benutzergruppen die konfiguriert wurden.

Schema

```
-----
--
-- Table structure for table `group`
--
CREATE TABLE `group` (
  `id` int(10) unsigned NOT NULL auto_increment,
```



```

`name` varchar(30) NOT NULL default '',
`class_string` varchar(30) NOT NULL default '',
`description` varchar(50) NOT NULL default '',
`unused` text NOT NULL,
PRIMARY KEY (`id`)
) TYPE=MyISAM AUTO_INCREMENT=13 ;

```

5.2.28 mac_blacklist

Verwendung

Diese Tabelle enthält alle MAC Adressen die gesperrt wurden.

Schema

```

-----
--
-- Table structure for table `mac_blacklist`
--
CREATE TABLE `mac_blacklist` (
  `id` int(10) unsigned NOT NULL auto_increment,
  `mac` varchar(17) NOT NULL default '',
  `date` datetime NOT NULL default '0000-00-00 00:00:00',
  `reason` text NOT NULL,
  `how_long` datetime NOT NULL default '0000-00-00 00:00:00',
  PRIMARY KEY (`id`)
) TYPE=MyISAM AUTO_INCREMENT=5 ;

```

5.2.29 mac_user_history

Verwendung

Diese Tabelle enthält eine Liste aller MAC Adressen die ein Benutzer je hatte wenn er sich eingeloggt hat.

Schema

```

-----
--
-- Table structure for table `mac_user_history`
--
CREATE TABLE `mac_user_history` (
  `id` bigint(20) unsigned NOT NULL auto_increment,
  `userid` bigint(20) unsigned NOT NULL default '0',
  `mac_address` varchar(17) NOT NULL default '',
  `username` varchar(30) NOT NULL default '',
  `ip_address` varchar(45) NOT NULL default '',
  `first_time` datetime NOT NULL default '0000-00-00 00:00:00',
  `last_time` datetime NOT NULL default '0000-00-00 00:00:00',
  PRIMARY KEY (`id`)
) TYPE=MyISAM AUTO_INCREMENT=10 ;

```

5.2.30 switchmobiel_acls

Verwendung

Diese Tabelle enthält die aktuellen SWITCHmobile ACLs.

Schema

```

-----

```



```
--
-- Table structure for table `switchmobile_acls`
--

CREATE TABLE `switchmobile_acls` (
  `id` int(10) unsigned NOT NULL default '0',
  `current_version` varchar(14) NOT NULL default '',
  `last_check` datetime NOT NULL default '0000-00-00 00:00:00',
  `last_update` datetime NOT NULL default '0000-00-00 00:00:00',
  `current_acls` text NOT NULL,
  PRIMARY KEY (`id`)
) TYPE=MyISAM;
```

5.2.31 traffic_stats_services

Verwendung

Diese Tabelle enthält den totalen Verkehr den ein Service (Protokoll + Destination-Port) je produziert hat. Zudem wird auch für jeden Service den Durchsatz aus der letzten Zeitperiode festgehalten.

Schema

```
-- -----
--
-- Table structure for table `traffic_stats_services`
--

CREATE TABLE `traffic_stats_services` (
  `protocol` varchar(4) NOT NULL default '',
  `port` smallint(5) unsigned NOT NULL default '0',
  `bytes_received` bigint(20) unsigned NOT NULL default '0',
  `bytes_sent` bigint(20) unsigned NOT NULL default '0',
  `packets_received` bigint(20) unsigned NOT NULL default '0',
  `packets_sent` bigint(20) unsigned NOT NULL default '0',
  `bytes_sec_received` int(10) unsigned NOT NULL default '0',
  `bytes_sec_sent` int(10) unsigned NOT NULL default '0',
  `packets_sec_received` int(10) unsigned NOT NULL default '0',
  `packets_sec_sent` int(10) unsigned NOT NULL default '0',
  `updated` datetime NOT NULL default '0000-00-00 00:00:00',
  PRIMARY KEY (`protocol`,`port`)
) TYPE=MyISAM;
```

5.2.32 traffic_throughput

Verwendung

Diese Tabelle enthält den Datendurchsatzes für das ganze System aus der letzten Zeitperiode.

Schema

```
-- -----
--
-- Table structure for table `traffic_throughput`
--

CREATE TABLE `traffic_throughput` (
  `who` varchar(45) NOT NULL default '',
  `packets_sec_received` bigint(20) unsigned NOT NULL default '0',
  `bytes_sec_received` bigint(20) unsigned NOT NULL default '0',
  `packets_sec_sent` bigint(20) unsigned NOT NULL default '0',
  `bytes_sec_sent` bigint(20) unsigned NOT NULL default '0'
) TYPE=MyISAM;
```



5.2.33 user

Verwendung

Diese Tabelle enthält alle Benutzer von MuSeGa. Ein Benutzer wird hier automatisch eingetragen wenn er sich via RADIUS anmeldet.

Schema

```

-----
--
-- Table structure for table `user`
--
CREATE TABLE `user` (
  `id` int(10) unsigned NOT NULL auto_increment,
  `username` varchar(30) NOT NULL default '',
  `group` varchar(30) NOT NULL default '',
  `mac_address` varchar(17) NOT NULL default '',
  `ip_address` varchar(45) NOT NULL default '',
  `online` char(3) NOT NULL default '',
  `last_login` datetime NOT NULL default '0000-00-00 00:00:00',
  `nb_login` int(10) unsigned NOT NULL default '0',
  `locked` char(3) NOT NULL default '',
  `total_packet_sent` bigint(20) unsigned NOT NULL default '0',
  `total_byte_sent` bigint(20) unsigned NOT NULL default '0',
  `total_packet_received` bigint(20) unsigned NOT NULL default '0',
  `total_byte_received` bigint(20) unsigned NOT NULL default '0',
  `local_auth` char(3) NOT NULL default 'no',
  `local_pw` varchar(15) NOT NULL default '',
  `nmap_result` text NOT NULL,
  `client_os` varchar(30) NOT NULL default '',
  `client_open_port` text NOT NULL,
  `description` varchar(100) NOT NULL default '',
  `message` text NOT NULL,
  PRIMARY KEY (`id`)
) TYPE=MyISAM AUTO_INCREMENT=27 ;

```

6 MuSeGa Systeminformationen

6.1 Hardware Informationen

IBM eServer xSeries 335

Processors	4 (2 Hyperthreaded)
Model	Intel(R) Xeon(TM) CPU 2.80GHz
Chip MHz	2791.95 MHz
Cache Size	512 KB
System	22216.7
Bogomips	



```

PCI Devices      0000:00:01.0 VGA compatible controller: ATI Technologies Inc Rage XL
                 0000:00:0f.1 IDE interface: ServerWorks CSB5 IDE Controller
                 0000:01:01.0 SCSI storage controller: LSI Logic / Symbios Logic 53c1030 PCI-X
                 Fusion-MPT Dual Ultra320 SCSI
                 0000:01:02.0 Ethernet controller: Intel Corp. 82557/8/9 [Ethernet Pro 100]
                 0000:02:01.0 Ethernet controller: Broadcom Corporation NetXtreme BCM5703X
                 Gigabit Ethernet
                 0000:02:02.0 Ethernet controller: Broadcom Corporation NetXtreme BCM5703X
                 Gigabit Ethernet

IDE Devices      none

SCSI Devices     LSILOGIC 1030 IM (Direct-Access)
                 IBM 25P3495a S320 1 (Processor)

Memory          1GB ECC Memory

Disks           2x 36GB 15'000rpm U320 SCSI Disks (RAID1)
    
```

6.2 Betriebssystem Version

Linux: Debian 3.1, Codename Sarge

6.3 Kernel Version

2.6.8-mppe-ipt-04 (SMP)

6.4 Installierte Software

```

musega:/# dpkg -l
Desired=Unknown/Install/Remove/Purge/Hold
| Status=Not/Installed/Config-files/Unpacked/Failed-config/Half-installed
|/ Err?=(none)/Hold/Reinst-required/X=both-problems (Status,Err: uppercase=bad)
||/ Name                    Version                Description
+++-----
-----
ii  adduser                    3.59                   Add and remove users and groups
ii  apache-common              1.3.33-2               Support files for all Apache webserver
ii  apache-ssl                 1.3.33-2               Versatile, high-performance HTTP server with SSL
support
ii  apache-utils               1.3.33-2               Utility programs for webserver
ii  apt                        0.5.27                 Advanced front-end for dpkg
ii  apt-utils                  0.5.27                 APT utility programs
ii  aptitude                   0.2.15.8-1            terminal-based apt frontend
ii  ash                         0.5.1-3                Compatibility package for the Debian Almquist Shell
ii  at                          3.1.8-11              Delayed job execution and batch processing
ii  base-config                2.53.5                 Debian base system configurator
ii  base-files                  3.1.2                  Debian base system miscellaneous files
ii  base-passwd                 3.5.9                  Debian base system master password and group files
ii  bash                        2.05b-24              The GNU Bourne Again SHell
ii  binutils                    2.15-5                 The GNU assembler, linker and binary utilities
ii  bsdmainutils                6.0.17                 collection of more utilities from FreeBSD
ii  bsdutils                    2.12-10                Basic utilities from 4.4BSD-Lite
ii  bzip2                       1.0.2-1                A high-quality block-sorting file compressor -
utilities
ii  console-common              0.7.47                 Basic infrastructure for text console configuration
ii  console-data                2002.12.04dbs-46      Keymaps, fonts, charset maps, fallback tables for
console-tools
ii  console-tools               0.2.3dbs-55           Linux console and font utilities
ii  coreutils                   5.2.1-2                The GNU core utilities
ii  cpio                        2.5-1.1                GNU cpio -- a program to manage archives of files.
ii  cpp                          3.3.5-1                The GNU C preprocessor (cpp)
ii  cpp-2.95                    2.95.4-22             The GNU C preprocessor
ii  cpp-3.3                     3.3.5-5                The GNU C preprocessor
ii  cramfsprogs                 1.1-6                  Tools for CramFs (Compressed ROM File System)
ii  cron                         3.0pl1-86              management of regular background processing
ii  dash                         0.5.1-3                The Debian Almquist Shell
ii  debconf                     1.4.30.11              Debian configuration management system
ii  debconf-english             1.4.30.11              small footprint English-only debconf
ii  debconf-utils               1.4.30.11              debconf utilities
ii  debhelper                   4.2.28                 helper programs for debian/rules
ii  debianutils                 2.8.4                  Miscellaneous utilities specific to Debian
ii  defoma                      0.11.8-0.1             Debian Font Manager -- automatic font configuration
    
```



framework		
ii devscripts	2.8.5	Scripts to make the life of a Debian Package
maintainer easier		
ii dhcp	2.0p15-19.1	DHCP server for automatic IP address assignment
ii dhcp-client	2.0p15-19.1	DHCP Client
ii dialog	1.0-20041222-1	Displays user-friendly dialog boxes from shell
scripts		
ii diff	2.8.1-7	File comparison utilities
ii dpkg	1.10.25	Package maintenance system for Debian
ii dpkg-dev	1.10.25	Package building tools for Debian
ii dselect	1.10.25	a user tool to manage Debian packages
ii e2fslibs	1.35-6	The EXT2 filesystem libraries
ii e2fsprogs	1.35-6	The EXT2 file system utilities and libraries
ii emacs20	20.7-13.1	The GNU Emacs editor.
ii emacsen-common	1.4.15	Common facilities for all emacsen.
ii ethereal-common	0.10.6-1	Network traffic analyser (common files)
ii exim	3.36-11	An MTA (Mail Transport Agent)
ii fdutils	5.4-20040228-1	Linux floppy utilities
ii file	4.12-1	Determines file type using "magic" numbers
ii fileutils	5.2.1-2	The GNU file management utilities (transitional
package)		
ii findutils	4.1.20-5	utilities for finding files--find, xargs, and locate
ii fontconfig	2.2.3-4	generic font configuration library
ii gcc	3.3.5-1	The GNU C compiler
ii gcc-2.95	2.95.4-22	The GNU C compiler
ii gcc-3.3	3.3.5-5	The GNU C compiler
ii gcc-3.3-base	3.3.5-5	The GNU Compiler Collection (base package)
ii gettext	0.14.1-6	GNU Internationalization utilities
ii gettext-base	0.14.1-6	GNU Internationalization utilities for the base
system		
ii gettext-el	0.14.1-6	Emacs po-mode for editing .po files
ii gradm2	2.0.1-3	Administration program for the grsecurity2 RBAC
based ACL system		
ii grep	2.5.1.ds1-4	GNU grep, egrep and fgrep
ii grep-dctrl	2.1.7	Grep Debian package information
ii groff-base	1.18.1.1-5	GNU troff text-formatting system (base system
components)		
ii gzip	1.3.5-9	The GNU compression utility
ii host	20000331-9	utility for querying DNS servers
ii hostname	2.13	A utility to set/show the host name or domain name
ii html2text	1.3.2a-1	An advanced HTML to text converter
ii ifupdown	0.6.4-4.8	High level tools to configure network interfaces
ii info	4.7-2.2	Standalone GNU Info documentation browser
ii initrd-tools	0.1.74	tools to create initrd image for prepackaged Linux
kernel		
ii initscripts	2.86-5	Standard scripts needed for booting and shutting
down		
ii intltool-debian	0.30+20040213	Help i18n of RFC822 compliant config files
ii ipchains	1.3.10-15	Network firewalling for Linux 2.2.x
ii iproute	20041019-1	Professional tools to control the networking in
Linux kernels		
ii iptables	1.2.11-9	Linux kernel 2.4+ iptables administration tools
ii kernel-image-2.6.8-mppe-ipt-02	luk.3	Linux kernel binary image for version 2.6.8-mppe-
ipt-02.		
ii kernel-image-2.6.8-mppe-ipt-04	luk.5	Linux kernel binary image for version 2.6.8-mppe-
ipt-04.		
ii kernel-package	8.117	A utility for building Linux kernel related Debian
packages.		
ii kernel-source-2.6.8	2.6.8-11	Linux kernel source for version 2.6.8 with Debian
patches		
ii klogd	1.4.1-16	Kernel Logging Daemon
ii less	382-1	Pager program similar to more
ii libacl1	2.2.23-1	Access control list shared library
ii libadns1	1.0-8.2	Asynchronous-capable DNS client library and
utilities		
ii libapache-mod-php4	4.3.10-2	server-side, HTML-embedded scripting language
(apache 1.3 module)		
ii libatml	2.4.1-16	shared library for ATM (Asynchronous Transfer Mode)
ii libatml-dev	2.4.1-16	Development files for compiling ATM programs
ii libattr1	2.4.16-1	Extended attribute shared library
ii libauthen-radius-perl	0.09-1	user authentication against radius
ii libblkid1	1.35-6	Block device id library
ii libbz2-1.0	1.0.2-1	A high-quality block-sorting file compressor library
- runtime		
ii libc6	2.3.2.ds1-20	GNU C Library: Shared libraries and Timezone data
ii libc6-dev	2.3.2.ds1-20	GNU C Library: Development Libraries and Header
Files		
ii libcap1	1.10-14	support for getting/setting POSIX.1e capabilities
ii libcomerr2	1.35-6	The Common Error Description library
ii libconsole	0.2.3dbs-55	Shared libraries for Linux console and font
manipulation		
ii libdb1-compat	2.1.3-7	The Berkeley database routines [glibc 2.0/2.1



compatibility]		
ii libdb2	2.7.7.0-9	The Berkeley database routines (run-time files)
ii libdb3	3.2.9-20	Berkeley v3 Database Libraries [runtime]
ii libdb3-util	3.2.9-20	Berkeley v3 Database Utilities
ii libdb4.2	4.2.52-17	Berkeley v4.2 Database Libraries [runtime]
ii libdbd-mysql-perl	2.9003-3	A Perl5 database interface to the MySQL database
ii libdbi-perl	1.45-1	The Perl5 Database Interface by Tim Bunce
ii libdevmapper1.00	1.00.19-2	The Linux Kernel Device Mapper userspace library
ii libexpat1	1.95.8-1	XML parsing C library - runtime library
ii libfontconfig1	2.2.3-4	generic font configuration library (shared library)
ii libfreetype6	2.1.7-2.3	FreeType 2 font engine, shared library files
ii libgcc1	3.4.3-6	GCC support library
ii libgcrypt11	1.2.0-4	LGPL Crypto library - runtime library
ii libgd2-noxpm	2.0.33-1.1	GD Graphics Library version 2 (without XPM support)
ii libgdbm3	1.8.3-2	GNU dbm database routines (runtime version)
ii libgdbmgl	1.7.3-28	GNU dbm database routines (runtime version)
ii libglib1.2	1.2.10-9	The GLib library of C routines
ii libglib2.0-0	2.4.8-1	The GLib library of C routines
ii libgnutls11	1.0.16-9	GNU TLS library - runtime library
ii libgpg-error0	1.0-1	library for common error values and messages in
GnuPG components		
ii libice6	4.3.0.dfsg.1-8	Inter-Client Exchange library
ii libident	0.22-2.2	simple RFC1413 client library - runtime
ii libipc-shareable-perl	0.60-4	Access IPC shared memory segments through perl.
ii libipc-sharelite-perl	0.09-3	Perl module that provides a simple interface to
shared memory		
ii libjpeg62	6b-9	The Independent JPEG Group's JPEG runtime library
ii libldap2	2.1.30-3	OpenLDAP libraries
ii liblocale-gettext-perl	1.01-17	Using libc functions for internationalization in
Perl		
ii liblockfile1	1.06	NFS-safe locking library, includes dotlockfile
program		
ii liblzo1	1.08-1.2	A real-time data compression library
ii libmagic1	4.12-1	File type determination library using "magic"
numbers		
ii libmm11	1.1.3-6.2	Shared memory library
ii libmysqlclient10	3.23.56-2	LGPL-licensed client library for MySQL databases
ii libmysqlclient12	4.0.23-1	mysql database client library
ii libncurses5	5.4-4	Shared libraries for terminal handling
ii libncurses5-dev	5.4-4	Developer's libraries and docs for ncurses
ii libncursesw5	5.4-4	Shared libraries for terminal handling (wide
character support)		
ii libnet-daemon-perl	0.38-1	Perl module for building portable Perl daemons
easily.		
ii libnet-pcap-perl	0.04-3	Pcap interface for perl
ii libnewt0	0.50.17-9.6	Not Erik's Windowing Toolkit - text mode windowing
with slang		
ii libopencdk8	0.5.5-10	Open Crypto Development Kit (OpenCDK) (runtime)
ii libpam-modules	0.76-22	Pluggable Authentication Modules for PAM
ii libpam-runtime	0.76-22	Runtime support for the PAM library
ii libpam0g	0.76-22	Pluggable Authentication Modules library
ii libpam0g-dev	0.76-22	Development files for PAM
ii libpcap0	0.6.2-2	System interface for user-level packet capture.
ii libpcap0.7	0.7.2-7	System interface for user-level packet capture
ii libpcap0.7-dev	0.7.2-7	Development library and header files for libpcap 0.7
ii libpcap0.8	0.8.3-5	System interface for user-level packet capture
ii libpcre3	4.5-1.1	Perl 5 Compatible Regular Expression Library -
runtime files		
ii libperl5.8	5.8.4-5	Shared Perl library
ii libplrpc-perl	0.2017-1	Perl extensions for writing PLRPC servers and
clients		
ii libpng12-0	1.2.8rel-1	PNG library - runtime
ii libpopt0	1.7-5	lib for parsing cmdline parameters
ii libradius1	0.3.2-8	/bin/login replacement with RADIUS. Shared lib to
used by programs.		
ii libreadline4	4.3-11	GNU readline and history libraries, run-time
libraries		
ii librrd0	1.0.49-1	Time-series data storage and display system
(runtime)		
ii librrdp-perl	1.0.49-1	Time-series data storage and display system (perl-
piped)		
ii librrds-perl	1.0.49-1	Time-series data storage and display system (perl-
shared)		
ii libsasl2	2.1.19-1.5	Authentication abstraction library
ii libsasl17	1.5.28-6.4	Authentication abstraction library
ii libsigc++1.2-5c102	1.2.5-1	Type-safe Signal Framework for C++ - runtime
ii libsm6	4.3.0.dfsg.1-8	X Window System Session Management library
ii libsp1	1.3.4-1.2.1-43	Runtime library for James Clark's SP suite
ii libss2	1.35-6	Command-line interface parsing library
ii libssl-dev	0.9.7e-2	SSL development libraries, header files and
documentation		
ii libssl0.9.6	0.9.6m-1	SSL shared libraries (old version)



ii libssl0.9.7	0.9.7e-2	SSL shared libraries
ii libstdc++2.10-glibc2.2	2.95.4-22	The GNU stdc++ library
ii libstdc++5	3.3.5-5	The GNU Standard C++ Library v3
ii libtasn1-2	0.2.10-3	Manage ASN.1 structures (runtime)
ii libuuid1	1.35-6	Universally unique id library
ii libwrap0	7.6.dbs-6	Wietse Venema's TCP wrappers library
ii libx11-6	4.3.0.dfsg.1-8	X Window System protocol client library
ii libxaw7	4.3.0.dfsg.1-8	X Athena widget set library
ii libxext6	4.3.0.dfsg.1-8	X Window System miscellaneous extension library
ii libxft1	4.3.0.dfsg.1-8	FreeType-based font drawing library for X (version 1)
ii libxi6	4.3.0.dfsg.1-8	X Window System Input extension library
ii libxmu6	4.3.0.dfsg.1-8	X Window System miscellaneous utility library
ii libxmuu1	4.3.0.dfsg.1-8	lightweight X Window System miscellaneous utility library
ii libxp6	4.3.0.dfsg.1-8	X Window System printing extension library
ii libxpm4	4.3.0.dfsg.1-8	X pixmap library
ii libxrandr2	4.3.0.dfsg.1-8	X Window System Resize, Rotate and Reflection extension library
ii libxrender1	0.8.3-7	X Rendering Extension client library
ii libxt6	4.3.0.dfsg.1-8	X Toolkit Intrinsics
ii libxtrap6	4.3.0.dfsg.1-8	X Window System protocol-trapping extension library
ii libxtst6	4.3.0.dfsg.1-8	X Window System event recording and testing extension library
ii lilo	22.6.1-4	Linux LOader - The Classic OS loader can load Linux and others
ii linux-kernel-headers	2.5.999-test7-bk-16	Linux Kernel Headers for development
ii linuxdoc-tools	0.9.21	SGML converters for the LinuxDoc DTD only.
ii linuxlogo	4.09-1	Color ANSI System Logo
ii login	4.0.3-30.7	System login tools
ii logrotate	3.7-2	Log rotation utility
ii lynx-ssl	2.8.4.1b-3.1	Text-mode WWW Browser supporting SSL
ii mailx	8.1.2-0.20040524cv5-4	A simple mail user agent
ii make	3.80-9	The GNU version of the "make" utility.
ii makedev	2.3.1-75	Creates device files in /dev
ii man-db	2.4.2-19	The on-line manual pager
ii manpages	1.70-1	Manual pages about using a GNU/Linux system
ii mawk	1.3.3-11	a pattern scanning and text processing language
ii mbr	1.1.5-2	Master Boot Record for IBM-PC compatible computers.
ii mime-support	3.28-1	MIME files 'mime.types' & 'mailcap', and support programs
rc modconf	0.2.43	Device Driver Configuration
ii module-init-tools	3.1-rel-2	tools for managing Linux kernel modules
ii modutils	2.4.26-1.2	Linux module utilities
ii mount	2.12-10	Tools for mounting and manipulating filesystems
ii mysql-client	4.0.23-1	mysql database client binaries
ii mysql-common	4.0.23-1	mysql database common files (e.g. /etc/mysql/my.cnf)
ii mysql-server	4.0.23-1	mysql database server binaries
ii ncurses-base	5.4-4	Descriptions of common terminal types
ii ncurses-bin	5.4-4	Terminal-related programs and man pages
ii net-tools	1.60-10	The NET-3 networking toolkit
ii netbase	4.19	Basic TCP/IP networking system
ii netkit-inetd	0.10-10	The Internet Superserver
ii netkit-ping	0.10-10	The ping utility from netkit
ii nload	0.6.0-2	A realtime console network usage monitor
ii ntpdate	4.2.0a-11	The ntpdate client for setting system time from NTP servers
ii nvi	1.79-21	4.4BSD re-implementation of vi
ii openssl	0.9.7e-2	Secure Socket Layer (SSL) binary and related cryptographic tools
ii passwd	4.0.3-30.7	Change and administer password and group data
ii patch	2.5.9-2	Apply a diff file to an original
ii pciutils	2.1.11-15	Linux PCI Utilities
ii perl	5.8.4-5	Larry Wall's Practical Extraction and Report Language
ii perl-base	5.8.4-5	The Pathologically Eclectic Rubbish Lister
ii perl-modules	5.8.4-5	Core Perl modules
ii perl-suid	5.8.4-5	Runs setuid Perl scripts
ii php4	4.3.10-2	server-side, HTML-embedded scripting language (meta-package)
ii php4-common	4.3.10-2	Common files for packages built from the php4 source
ii php4-mysql	4.3.10-2	MySQL module for php4
ii phpmysqladmin	2.6.1-rc1-1	A set of PHP-scripts to administrate MySQL over the WWW
ii phpsysinfo	2.3-1	PHP Based Host Information
ii po-debconf	0.8.15	Manage translated Debconf templates files with gettext
ii procs	3.2.1-2	The /proc file system utilities
ii psmisc	21.5-1	Utilities that use the proc filesystem
ii radiusclient1	0.3.2-8	/bin/login replacement which uses the RADIUS protocol for authentication.
ii rrdtool	1.0.49-1	Time-series data storage and display system



(programs)		
ii sed	4.1.2-8	The GNU sed stream editor
ii setserial	2.17-36	Controls configuration of serial ports
ii sgml-base	1.26	SGML infrastructure and SGML catalog file support
ii sgml-data	2.0.2	common SGML and XML data
ii shellutils	5.2.1-2	The GNU shell programming utilities (transitional package)
ii slang1	1.4.9dbs-8	The S-Lang programming library - runtime version
ii slangla-utf8	1.4.9dbs-8	The S-Lang programming library with utf8 support
ii sp	1.3.4-1.2.1-43	James Clark's SGML parsing tools
ii ssh	3.8.1pl-8.sarge.4	Secure rlogin/rsh/rcp replacement (OpenSSH)
ii ssl-cert	1.0-10	Simple debconf wrapper for openssl
ii strace	4.5.5-1	A system call tracer
ii sysklogd	1.4.1-16	System Logging Daemon
ii syslinux	2.11-0.1	Bootloader for Linux/i386 using MS-DOS floppies
ii sysv-rc	2.86-5	Standard boot mechanism using symlinks in /etc/rc?.d
ii sysvinit	2.86-5	System-V like init
ii tar	1.13.93-4	GNU tar
ii tasksel	2.15	Tool for selecting tasks for installation on Debian system
ii tcpd	7.6.dbs-6	Wietse Venema's TCP wrapper utilities
ii tcpdump	3.8.3-3	A powerful tool for network monitoring and data acquisition
ii telnet	0.17-26	The telnet client.
ii tethereal	0.10.6-1	Network traffic analyzer (console)
ii textutils	5.2.1-2	The GNU text file processing utilities (transitional package)
ii ttf-bitstream-vera	1.10-3	The Bitstream Vera family of free TrueType fonts
ii ucf	1.13	Update Configuration File: preserves user changes to config files.
ii util-linux	2.12-10	Miscellaneous system utilities
ii wget	1.9.1-8	retrieves files from the web
ii wwwconfig-common	0.0.42	Debian web auto configuration
ii xfree86-common	4.3.0.dfsg.1-8	X Window System (XFree86) infrastructure
ii xlibs	4.3.0.dfsg.1-8	X Window System client libraries metapackage and XKB data
ii xlibs-data	4.3.0.dfsg.1-8	X Window System client data
ii xml-core	0.09	XML infrastructure and XML catalog file support
ii zlib1g	1.2.2-3	compression library - runtime

6.5 Selbst kompilierte Software

Zu den Debianpaketen kamen noch diverse selbst kompilierte Software dazu:

- **ppp-2.4.3**: mit selbst gepatchtem RADIUS Plugin
- **pptpd-1.2.1**

Weiter wurde natürlich der **Kernel** selbst übersetzt und daraus ein Debianpaket erstellt.

Das **iptables** Paket habe ich ebenfalls selbst kompiliert und neu erstellt.

7 Glossar

Der Glossar wird aufgrund der einfacheren Erweiterung in einem separaten Dokument geführt. Er ist unter <http://musega.ch> zu finden.