6 Qmail

6.1 Qmail, die erste (Pre-Installation)

Qmail liegt (offiziell) in drei verschiedenen Formaten vor:

- dem Qmail Tar-Archiv,
- dem Qmail RPM für Red-Hat Linux,
- dem Qmail-Paket für Debain (.deb),
- einem (nicht mehr verfügbaren) Port für FreeBSD (3.x/4.0).

Grund hierfür ist die "restriktive" Lizenz-Politik von Dan J. Bernstein. Selbst der Port für Qmail, der noch unter FreeBSD 3.x/4.0 zur Verfügung stand, wird nicht mehr angeboten (kann aber trotzdem noch genutzt werden).

Bedenkt man, dass kommerzielle (aber freie) Produkte wie der Netscape-Browser, Adobe's Acrobat Reader oder gar die verschiedenen Office-Suiten problemlos in die verschiedenen Distributionen Einklang finden, ist dies beachtlich und verdeutlicht einmal mehr Dan J. Bernstein's konsequente Einstellung gegenüber einer nicht-konformen Paketierung.

Ich hatte einmal das "Vergnügung", Qmail unter SuSE LINUX 6.4 zu betreuen. Obwohl ich trotz heftiger Suche nie auf ein SuSE Qmail-Paket gestossen bin, muss es sowas geben. Einen Hinweis findet sich nur im SuSE Handbuch (dieser Version, wo Qmail nur im Zusammenhang mit meinen (etwas betagten) Artikel in der iX ("In der ersten Reihe - Email Gateway mit qmail unter Linux” 12/1997) Erwähnung findet. Dieses Paket war so verbogen, dass Dan sicherlich die Haare zu Berge gestanden hätten. Insbesondere lief Qmail unter *root*-Rechten.

Aus dem gesagten soll geschlussfolgert werden, dass es am besten ist, Qmail von der Quelle, d.h. dem sog. Tarball zu installieren. Damit ist man/frau sicher, dass

- das Produkt vollständig und gemäss Vorschrift installiert,
- das Abhängigkeiten vom Betriebssystem erfasst, und evtl. Kompiler- und Libraray-Abhängigkeiten korrekt verarbeitet und aufgelöst wurden und ggf.
- Patches, die in der Regel gegen die Qmail Source-Dateien laufen, eingebunden werden können.

Bevor jedoch zur Installation geschritten werden kann, sind noch einige Vorüberlegungen anzustellen.

6.1.1 Qmail-User

Qmail verteilt seine Aufgaben auf unterschiedliche Unix Benutzer und zwei Unix Gruppen. Das Dokument `INSTALL.ids` gibt Anweisungen, welche das sind und wie sie ins Betriebssystem einzurichten sind.

Zunächst sind die Gruppen zu erstellen. Dies sollte — wie alle weiter unten geschilderten Aktionen — mit "Bordmitteln" geschehen, d.h. ohne die Hilfe entsprechender Administrations-Tools wie **yast**, **linuxconf** oder **/stand/sysinstall**. Die Gruppen können in der Regel einfach durch einen Editor-Eintrag in die Datei `/etc/group` aufgenommen werden:

```
qmail:*:2107:
nofiles:*:2108:
```

Listing 6-1: Von Qmail benutzte Gruppen; die GID ist weitgehend frei wählbar.

Hierbei ist nur der Name von Belang; die GID kann beliebig vergeben werden. Dem Sicherheitsgedanken von Dan J. Benstein entsprechend, ist der numerische Wert recht "gross" gewählt; und sollte keineswegs unter 100 betragen.

Anschliessend sind die einzelnen Qmail-User einzurichten. Dies sollte ebenfalls durch Systemkommandos erfolgen. Im Prinzip können die entsprechenden Anweisungen in der Datei `INSTALL.ids` einfach per "drag'n'drop" kopiert und für das betreffende Betriebssystem ausgeführt werden. Die korekte Installation lässt sich durch einen Blick in die Datei `/etc/passwd` verifizieren:

```
alias:*:7790:2108::/var/qmail/alias:/nonexistant
qmaild:*:7791:2108::/var/qmail:/ nonexistant
qmaill:*:7792:2108::/var/qmail:/ nonexistant
qmailp:*:7793:2108::/var/qmail: nonexistant
qmailq:*:7794:2107::/var/qmail:/ nonexistant
qmailr:*:7795:2107::/var/qmail:/ nonexistant
qmails:*:7796:2107::/var/qmail:/ nonexistant
```

Listing 6-2: Die Qmail-User; auch hier gilt, dass die UID weitgehend selbst bestimmt werden kann, alle Acccounts werden ohne reale Login-Shell eingerichtet.

Hierbei ist folgendes zu bemerken:

- Die Qmail-User werden ohne Passwort und ohne "reale" Login-Shell eingerichtet ("`/nonexistant`" oder ggf. "`/bin/true`").
- Die per Kommando erzeugten UIDs und GIDs der Qmail-Benutzer werden

in der Regel einfach ausgehend vom letzten eingerichteten Benutzer hochgezählt. Wer Wert darauf legt, dass Qmail entweder den Vorgaben von Dan J. Bernstein, oder nach (sinnvollen) lokalen Vorgaben nach installiert wird, kommt nicht umhin diese als Parameter der Kommandos mit anzugeben. Keinesfalls sollte anschliessend die Datei `/etc/passwd` manipuliert werden. Da heutige Unix-Systeme in der Regel eine sog. Shadow-Passwort-Datei pflegen, bedeutet dies unweigerlich eine Inkonsistenz des Datenbestandes für die Benutzerdaten.

- Mit Ausnahme des Users *alias* wird ihnen das Home-Verzeichnis `/var/qmail/` zugesprochen; was aber nicht kritisch ist.
- Der User *alias* verfügt als Home-Directory über `/var/qmail/alias/`. Sollte sich dies durch eine Skript-gesteuerte Installation z.B. auf `/home/alias/` geändert haben, gibt es Schwierigkeit!

Um es nochmals zu betonen: Eine per Installations-Skript ausgeführt Einrichtung der User führt in der Regel zu unerwünschten Nebeneffekten. Teilweise werden die User dann nach den Vorgaben im Skeleton-Verzeichnis (`/etc/skel/`) erzeugt; teilweise werden Verzeichnisse für die graphischen Benutzeroberflächen Gnome oder KDE angelegt und teilweise werden die Qmail-User mit Mailboxen unter `/var/spool/mail` versorgt. Ein andermal weigert sich das Installations-Tools eine Shell names "nonexistant" einzurichten, weil diese nicht in der Datei `/etc/shells` enthalten ist. Das ist sicherlich nicht das, was wir wollen.

6.1.2 Bedeutung der Qmail-User

- *qmailr* - Verantwortlich für das Senden von E-Mails an entfernte (per SMTP, d.h. remote) Empfänger (**qmail-rspawn**).
- *qmails* - Der allgemeine "Sende"-User von qmail (**qmail-send**).
- *qmailq* - Ist verantwortlich für die Qmail-Queue, in der die zu verarbeitenden Nachrichten (E-Mails) gespeichert sind.
- *qmailp* - Dieser Qmail-User wird benötigt, die Qmail-eigene User-Datenbank zu pflegen.
- *qmaill* - Verantwortlich für das Schreiben von Log-Mitteilungen (**splogger** — nur bei **qmail-send**).
- *qmaild* - Dieser Benutzer wird Qmail-Daemon-Prozessen zugeteilt, z.B. **qmail-smtpd**.
- *alias* - Dieser Benuzter ist verantwortlich für alle E-Mails, die keinem definierten lokalen Benutzer zugewiesen sind.

Vergleicht man diese Aufstellung mit der Einordnung in die Gruppen-Datei, so

erkennt man die zugrundeliegende Aufspaltung: In der Gruppe *qmail* finden sich die "Hilfsaccounts", die zum Betrieb von Qmail notwendig sind, während die anderen User der Gruppe *nofiles* eine übergeordnete Rolle besitzen. Hier ist zum einem *qmaild* für Daemon-Prozesse zu nennen und zum anderen der User *alias*, dem die zentrale Aufgabe zufällt, lokale Mails ohne zugeordneten Systemaccount zu bearbeiten. Zu dieser Kategorie gehören unter anderem Bounce-Mails, E-Mails an den User *root* und an den Postmaster.

6.1.3 Alias-User

Während die meisten Qmail-User "Laufzeit"-User sind, die also vom Betriebssystem für Ausführen der Qmail-Programmdateien benötigt werden, fällt dem Benutzer *alias* die zentrale Rolle zu, als Default-Empfänger zu fungieren, d.h. als stellvertrender Empfänger für einen nicht existierenden Benutzer. Jede E-Mail, die nicht an einen lokalen Benutzer zugestellt werden kann, wird zwangsläufig vom User *alias* entgegengenommen und von diesem verarbeitet. Die Regeln hierzu finden sich im Verzeichnes `/var/qmail/alias/` als "dot-qmail" Dateien, worauf weiter unten ausführlich eingegangen werden soll.

Das Dokument `INSTALL.alias`, beschreibt welche Alias-User mindestens eingerichtet werden sollen:

- `MAILER-DAEMON` Der Mailer-Daemon ist ein sozusagen inhärenter Benutzer, der dann gefordert ist, wenn eine von Qmail ausgeschickte Unzustellbarkeitsquittung fehlschläget; wir haben das als "Double-Bounce" kenngelernt. Dieser Benutzer existiert praktisch automatisch im Qmail-System. Sollte hier ein "realer" Benutzer entsprechende Notifikationen erhalten, erfolgt dies über einen Eintrag in die Datei `.qmail-MAILER-DAEMON`.
- `Postmaster` Vom Grundsatz her, ist der Postmaster verantwortlich für alle Probleme, die sich bei der Übertragung und Auslieferung von E-Mails auf dem lokalen MTA ergeben. Diese Systemadministrations-Aufgabe sollte von einer realen Person versehen werden, deren E-Mail-Adresse in .qmail-postmaster hinterlegt wird. Andernfalls landen Anfragen an den Postmaster im "Papierkorb".
- `Root` Ein Konzept von Qmail ist es, an den User *root* (UID=0) keine E-Mails auszuliefern. Viele Programme unter Unix tun dies aber als Default, z.B. der Syslog-Daemon; und ohne Aufforderung. Es empfiehlt sich also dringend, hier (d.h. in `.qmail-root`) ebenfalls einen realen Benutzer zu hinterlegen. Im Gegensatz zum Postmaster ist dieser nicht für die Belange des Mailsystems (also Qmail), sondern für die des lokalen Systems zuständig.

Weitere System-Accounts wie *ftp*, *www* und *uucp* sollten ebenfalls über eine

geeignete `.qmail` Datei verfügen. Ansonsten kann eine E-Mail, die an diese "User" gerichtet ist, nicht in Empfang genommen bzw. an den gewünschten Empfänger weiter geleitet werden.

6.1.4 Dateisysteme

Bevor Qmail kompiliert und installiert werden kann, muss das Heimatverzeichnis von Qmail erzeugt werden, das in der Regel unter `/var/qmail/` liegt. Eine der am häufgisten gestellten Fragen ist, wieso Qmail ausgerechnet unter `/var/qmail/` installiert werden sollte; wo doch jede andere Software üblicherweise unter findet `/usr/local/...` ihre Heimat findet?. Warum hat Dan Bernstein `/var/` ausgewählt? Der Grund ist, dass unter `/var/` hostspezifische Dateien abgelegt werden, die — per definitionem — nicht (per Binär-Transfer) von einem Rechner A auf einen Rechner B übertragen werden können. Dies gilt im besonderen für die per Installation erzeugte Qmail-Queue. Diese ist mit den (lokalen) Inodes des Dateisystems verknüpft. Um die Integrität und Performance von Qmail zu garantieren, hat Dan Bernstein beschlossen, auch die ausführbaren Qmail-Dateien im gleichen Verzeichnisbaum zu installieren.

Es ist natürlich prinzipiell möglich, Qmail statt in `/var/` in einem beliebigen Verzeichnis zu installieren — dies sehen wir weiter unten — nur, es ist nicht angeraten.

Die nächste Frage, die sich stellt, ist, welche Dateisysteme optimal für Qmail (unter dem Mountpunkt `/var`) geeignet sind. Unter Linux stellt sich z.B. die Wahl hinsichtlich ext2fs, ext3fs und reiserfs. Während ext2fs das Standard Linux-Filesystem ist, sind sowohl ext3fs wie auch reiserfs sog. Journaled Dateisysteme (JFS). Journaled Filesysteme schreiben Änderungen an Dateien im laufenden Betrieb mit und bieten somit bei einem Systemcrash die Möglichkeit, erfolgte Änderungen nachzuvollziehen; das Dateisystem bleibt also intakt.

Abgesehen von den ausführbaren Qmail-Dateien und ein paar Konfigurationsfiles, befindet sich unter `/var/qmail/` im wesentlichen die Qmail-Queue. Qmail ist im wesentlichen eine Datenbank, in die Nachrichten (E-Mails) als Dateien eingestellt werden und die von den entsprechenden Qmail-Modulen verarbeitet werden. Dan Bernstein hat jede Anstrengung unternommen, die Datenbank einerseits so zuverlässig wie möglich; andererseits so performant wie notwendig zu machen.

Es stellt sich an dieser Stelle die Frage, was "so performant wie notwendig" bedeutet. Hierauf gibt es keine eindeutige Antwort. Ich habe schon darauf verwiesen, dass Qmail (1.03) aus dem Jahre 1998 stammt. Bedenkt man, dass Dan Bernstein sicherlich einige Zeit gebraucht hat, die grundlegenden Ideen auszuformulieren, liegt der Schluss nahe, dass die Geburtsstunde von Qmail um 1995 liegt. Heute schreiben wir das Jahr 2002. In der Zwischenzeit hat sich das Mailvorlumen im Internet um mehrere Grössenordnungen erhöht. Dies ist auf

verschiedene Umstände zurückzuführen. Die wichtigsten sind, dass einfach viel mehr Menschen heutezutage "Online" sind und per E-Mail kommunizieren. Der nächste Faktor ist, dass E-Mail nicht mehr allein zum Transport von (verbalen) Nachrichten genutzt, sondern im grossen Umfang zur Übertragung von allgemeinen Infomationen wie Audio, Video und sonstigen Dateien herangezogen wird. Eine weitere Quelle sind kommerzielle E-Mails (Spam), die umfangreich versendet werden (und in den E-Mail Briefkasten flattern). Im geleichen Zeitraum hat sich natürlich auch die Kapazität des Internets vervielfältigt; auch wurden die Computer schneller. Kurz gesagt, die Vorausetzungen, unter denen heute die ISPs ihre E-Mail Gateways betreiben, sind deutlich andere, die noch 1998 gegolten haben. Die Anforderungen an E-Mail Gateways — und Qmail wird aufgrund seiner Sicherheitsstruktur gerne hier eingesetzt — haben sich dementsprechend drastisch erhöht. Für die zentralen MTAs der E-Mail Provider gilt, dass diese nicht nur mehr ein Volumen von mehreren zehntausend E-Mails pro Tag, sondern von 100.000 und mehr E-Mails pro Stunde verarbeiten müssen.

Bei der Unterbrinung der Qmail-Queue im Dateisystem stellen sich systembedingt zwei Fragen:

- ***Sicherheit***: Wie sicher sind die Dateien, d.h. Nachrichten, die in der Qmail-Queue untergebracht sind ?
- ***Performance***: Wie schnell können die Dateien (Nachrichten) (weiter-)verarbeitet werden ?

Im Zusammenhang mit der ***Ausfallsicherheit*** der Daten in der Qmail-Queue können zwei Antworten gegeben werden.

1. ***Software-Ausfallsicherheit***
   In der Regel bedarf es keiner weiteren Anstrengungen die (Ausfall-)Sicherheit der Qmail-Queue zu erhöhen, da Dan Bernstein hier bereits exemplarisches geleistet hat. Das entscheidende Kriterium ist hier, dass das Dateisystem die Daten synchron auf die Festplatte schreibt, d.h. dass sowohl die Dateien wie auch die Modifikation der Metainformation in der `./` Datei (die das Mutterverzeichnis beschreibt) synchron eingestellt werden. Dem hat Dan Bernstein eine eigene Web-Seite gewidmet (`http://cr.yp.to/qmail/faq/reliability.html#filesystems`). Faktisch ist es so, dass das asynchrone Schreiben gerne als performance-steigernde Feature eingesetzt wird, was unter dem Linux ext2 sowieso die Standardeinstellung. Gleiches gilt für FreeBSD, wobei hier der Terminus *softupdates* benutzt wird.
   Kennzeichnend ist in beiden Fällen, dass nach dem Schreiben einer Datei zunächst kein automatisches **fsync** erfolgt (sondern eben asynchron); sondern die Dateisystem-Information zunächst im Kernel konsistent gehalten wird; was einigen Verwaltungsaufwand bedeutet, aber I/Os erspart.

2. ***Hardware-Ausfallsicherheit***
   Die Entwicklung der heutigen Festplattentechnologie vollzieht sich in der Regel unabhängig davon, ob sie mit SCSI- und IDE-Interface ausgestattet werden. Da SCSI-Festplatten aber in der Regel einen höheren Verkaufspreis aufweisen, werden die aktuellsten Modelle gerne dort zuerst angeboten. Zudem ist das SCSI-Interface – meiner Meinung nach – dem von ATAPI überlegen, weil robuster. Bei den mittlerweilen hohen Geschwindigkeiten (Taktraten) sowohl auf dem SCSI- wie auch ATAPI-Bus, spielt die Frage der Verkabelung und Abschirmung eine bedeutende Rolle. Hier darf nicht an Qualität gespart werden.

   Die gängige Methode eine Hardware-Ausfallsicherheit zu garantieren, ist der Einsatz eines Plattenarrays in RAID-5 Architektur. Bekanntermassen werden hier zusätzlich Paritäts-Informationen auf die einzelnen Platten verteilt, sodass der Verlust einer Festplatte verkraftet werden kann. Daher sollte das RAID-5 Array mindestens 3 Platten umfassen; besser sind jedoch 5. Bei einem Netto-Datenvolumen von 20 Gbyte pro Platten stehen somit bei 5 Festplatten 100 GByte brutto und 80 GByte netto zur Verfügung. Das ist mehr als genug. Das Problem was sich hierbei stellt, ist ein ziemlich triviales: Bei der gegebenen MTBF[1] von Festplatten fallen diese erst Jahre später aus. Häufig kommen diese Festplatten auch aus einer Charge. Dann ist die Wahrscheinlichkeit gross, dass sehr bald eine zweite Festplatte ausfällt und das RAID-Array ist dahin. Versucht man, zum Zeitpunkt des Aufalls neue Festplatten (in der Regel sollten es die gleichen Typen sein) zu beschaffen, ist dies mit grossen Schwierigkeiten verknüpft, da normalerweise diese "alten" Platten nicht mehr zur Verfügung stehen — besonders wenn es sich um Festplatten aus dem Consumer-Bereich handelt. Man tut also gut daran, sich bei RAID-5 zum Zeitpunkt der Einrichtung einen Vorrat von Festplatten zu beschaffen.

   Möglicherweise besitzen die SCSI-Varianten der Festplatten auch — weil für den Servermarkt vorgesehen — geringere mechanische Toleranzen und längere Lebensdauern im ununterbrochenen Einsatz. Leider gibt es meines Wissens keine unabhängige Erhebung über die Lebendauer und Ausfallwahrscheinlichkeiten von Festplatten der verschiedenen Hersteller, abhängig von Modell, Charge und Interface. Verschiedentlich wird dies in den Fachzeitschriften sowie in SysAdmin-Kreisen diskutiert; häufig fehlt aber einfach eine statistische Signifikanz der Aussagen.

Die Frage der ***Performance*** wird dann relevant, wenn (bei heutiger Hardware) mehr als 100.000 E-Mails/Tag bzw. 12.000 E-Mails simultan von Qmail verarbeitet werden sollen. Bei einem Durchsatz von 100.000 E-Mails/Stunde sind

[1] MTBF Mean Time Between Failure

schon einige Überlegungen anzustrengen, um dies auch real zu erzielen. Die Performance des Filesystems hängt von vielen Faktoren ab (in abnehmender Bedeutung):

- Wie schnell ist die Festplattte, die `/var/qmail/` beheimatet?
- Wie sind die Dateisysteme auf die Festplatten verteilt?
- Hängt die Platte für das Dateisystem `/var/qmail/` ggf. an einem eigenen Kontroller (und Festplatte)?
- Welcher Typ von Dateisystem wurde für `/var/qmail/` gewählt?
- Wie ist das Dateisystem gemountet?
- Wieviele Bytes werden per Inode adressiert, d.h. wie hoch ist die Inode-Dichte?

RAID-Systeme, die ein simultanes Schreiben und Lesen auf mehreren Festplatten gestatten, gibt es mittlerweile für ATAPI- und SCSI-Kontroller. In jedem Fall ist der getrennte Einfluss auf die Lese- und Schreibgeschwindigkeit zu untersuchen. Dies gilt auch für das Caching des Betriebssystems auf dem gemounteten Dateisystem im Zusammenhang mit dem Caching des Controllers und/oder der Festplatten, die ja in der Regel auch heute über 2 MByte Cache verfügen. Verschiedentlich wurde in der Qmail-Diskussionsliste auch der Einsatz von "Solid-State" Disks – also Speichermodul-Disks – besprochen.

Eine abschliessende Empfehlung kann aber hier nicht erfolgen; die Situation bei der Qmail-Queue ähnlich bei bei Datenbank-Anwendungen wie Oracle, die ja häufig auf Raw-Devices angesiedelt werden: Bei extremen Anforderungen und festgestellten Performance-Engpässen muss das Gesamtsystem untersucht werden.

Wir wollen kurz zusammenfassen:

- Qmail braucht nicht unbedingt auf einem Journaled Filesystem installiert werden; vielmehr sollte der Typ des Dateisystems synchron arbeiten, d.h. die Metadaten zeitgleich mit den eigentlichen Nutzdaten abspeichern.
- Bei einem hohen E-Mail Volumen empfiehlt es sich, das Verzeichnis `/var/qmail/` auf eine eigene (schnelle) Platte zu legen, ggfs. an einen separaten Kontroller (zweiter IDE-Kanal oder SCSI-Kontroller).
- Je schneller die Festplatte umso besser. Hier spielt nicht unbedingt der (sustained) Durchsatz, sondern die Sektoren-Seektime eine Rolle, wobei aktuelle Ultra160-SCSI-Festplatten mit 15.000 UPM ca. 4 ms Zugriffszeit bieten und damit Vorteile bringen.
- Die Einrichtung von `/var/qmail/` auf einem RAID-Array bringt allenfalls bei ISPs spezifische Vorteile: Redundante Systeme wie RAID-5 erhöhen die

Ausfallsicherheit; RAID-0 Systeme bringen einen Gewinn im Durchsatz.

- Die weiteren Parameter spielen nur bei sehr hohen Datenvolumen eine Rolle. Dann ist es allerdings auch notwendig, alle anderen Performance-relevanten Faktoren zu optimieren (DNS-lookup, Netzwerk-Performance, PCI-Bus, Kernel-Parameter etc.).

6.2 Installation von Qmail

Der sicherste und m.E. auch schnellste Weg Qmail zu installieren, besteht zunächst darin, den sog. Qmail-Tarballs von Dan Bernsteins Qmail-Seite über folgende URL: `http://cr.yp.to/software/qmail-1.03.tar.gz` per Download zu holen. Wer ganz sicher sein will, dass er die richtige Qmail-Version erwischt hat, findet unter dem Link `http://cr.yp.to/qmail/dist.html` die korrekte MD5-Checksum, die für Qmail 1.03 `622f65f982e380dbe86e6574f3abcb7c` beträgt.

Häufig wird diese Qmail-Version als "Vanilla-Qmail" bezeichnet und die meisten Patche (so wie auch mein SPAMCONTROL) laufen gegen diese Version. Auf der Web-Seite `http://cr.yp.to/qmail/var-qmail.html` beschreibt Dan auch die Paketierung von Qmail für beliebige Unix-Syteme, doch sind über die Qmail-Homepage z.Z. nur Versionen für Redhat und Mandrake (also als RedHat Package Manager — RPM — Format) verfügbar sowie für Debin in Form von .debs. Es kursieren aber noch einige mehr oder weniger inoffizielle und z.T. gepatchte Versionen von Qmail; von deren Einsatz kann ich aber nur abraten.

In praktisch jedem Fall sind noch Post-Installations-Konfigurationen durchzuführen, die das scheinbar einfache Einspielen Vor-konfigurierter Pakete konterkarieren. Die Installation von Qmail kann zwar bis zum lauffähigen System (plattformspezifisch) weitgehend automatisiert werden; dies beinhaltet aber in jedem Fall nur eine Basis-Konfiguration.

Die Standard-Installation von der Quelle ist — aufgrund der eigenen DJB-Libraries — sowieso unproblematisch und geschieht bei heutigen Rechnern im Handumdrehen. Bei einigen Browsern werden (je nach MIME-Type) beim Download zusätzliche Bytes angehängt, die das Auspacken des Tar-Archivs torpedierern.

Eventuell auftretende Probleme äussern sich in entsprechenden Fehlermeldungen, wobei die Meldung "`main is not int`" bei gegenwärtigen Stand des (gnu) C-Compilers ignoriert werden kann. Notwendig sind jedoch Anpassungen für den gcc 3.2 Compiler, die im nächsten Kapitel beschrieben sind.

Bevor aber der Tarball ausgepackt wird, sollte zunächst überlegt und festgehalten werden, von welchem Verzeichnis aus die Installation zu erfolgen hat. Es ist zu

bedenken, dass Installationen nicht notwendigerweise einen "temporären" Charakter haben, sondern dass ggf. die Sourcen geändert und angepasst (per patch) werden müssen. Ich plädiere dafür, als Standard `/usr/local/src/` zu nutzen; das Verzeichnis `/tmp/` ist eindeutig ungeeignet. Nach `/usr/local/src/` werden alle zu installierenden Programme kopiert und dort entpackt. In der Regel erzeugen die tar-Archive entsprechende Unterverzeichnisse.

Wir wechseln in das erzeugte Vezeichnis `./qmail-1.03/` und sehen und den Inhalt an. Die beigefügten Qmail-Dokumente sind in Grossbuchstanden kenntlich gemacht: `ls -la [A-Z]*`. Es ist empfehlenswert, diese sich zunächst anzuschauen, vor allem die Dateien INSTALL*. Alle Dokumente befinden sich übrigens nach der Installation im Verzeichnis `/var/qmail/doc/`.

Die ausführbaren Dateien kennzeichnen sich durch den Suffix ".c". Ein kurzer Blick genügt, festzustellen, dass Qmail davon jede Menge mitbringt: `ls -la *.c`. Die meisten hiervon sind "embedded" werden also von den Hauptprogrammen benötigt. Im Gegensatz zu den neueren Entwicklungen von Dan J. Bernstein, kommt Qmail noch mit einer vollständigen Sammlung von man-pages. Diese lassen sich durch ihren numerischen Suffix erkennen, also: `ls -la *.[1-9]`. Wer mit den den Konventionen der man-pages (`man man`) vertraut ist, erkennt sofort, welche informatiellen Charakter haben, und welche sich auf ausführbare Dateien beziehen.

Sofern nicht zusätzliche Patche eingespielt werden müssen und evtl. die initiale Qmail-Konfiguration anzupassen sind, geschieht die Installation von Qmail vollkommen undramatisch. Das ebenfalls im Qmail-Quell-Verzeichnis liegende Skript **find-systype.sh** ermittelt die notwendigen plattform-spezifischen Einstellungen und die Installation von Qmail vollzieht sich (vorausgesetzt das Verzeichnis `/var/qmail/` existiert und die Qmail-User sind eingerichtet) durch das Kommando:

```
# make setup check
```

Durch das make setup werden die Sourcen kompiliert und gelinkt, als auch die Binärdateien nach `/var/qmail/bin/` transferiert und einige Shell-Skripte wie z.B. `mailsubj.sh` und `qmail-qstat.sh` ihres Suffixes beraubt und nach `/var/qmail/bin/` kopiert. Im gleichen Schritt werden die man-Dateien nach `/var/qmail/man/` eingestellt und die mitgelieferten Qmail-Dokumente in `/var/qmail/doc/` untergebracht. Dummerweise beinhaltet das Makefile nicht alle *.3 Dateien als man-page.

Wir holen das nach erfolgter Qmail-Installation nach:

```
% su
# mkdir /var/qmail/man/man3
# gzip *.3
```

```
# cp *.3.gz /var/qmail/man/man3/
```

Der Aufruf `man qmail` führt aber leider noch immer nicht zum gewünschten Resultat. Die Ursache liest im Default-Suchpfad für das Kommando **man**, das vom Verzeichnis `/var/qmail/man/` nichts weiss. Dieser Pfad ist üblicherweise `/usr/share/man/`. Alternative Suchpfade werden in der Konfigurationsdatei `/etc/manpath.config` (manchmal auch `man_db.config`) gesetzt. Wir holen dies nach und fügen an geeigneten Stellen ein:

```
MANDATORY_MANPATH       /var/qmail/man
...
MANPATH_MAP     /var/qmail/bin          /var/qmail/man
```

*Listing 6-3: Ergänzung des Suchpfades für das **man**-Kommando*

Der Teile `make check` erzeugt und verifiziert hingegen die Qmail-Queue, also das Verzeichnis `/var/qmail/queue/`. Sollte es einmal notwendig sein, die Qmail-Queue neu aufzusetzen (nach einem Plattenchrash oder falls die Auslieferung von E-Mails — durch das Löschen ganzer Unterverzeichnissse — ohne Rücksicht auf Verluste gestoppt werden müssen), genügt ein Aufruf von `make check` um dieses Dateisystem wieder konsistent neu auzubauen.

Wir schauen uns an, welche Verzeichnisstruktur **make** generiert hat (`ls -al /var/qmail`):

```
% ls -la /var/qmail
drwxr-xr-x  11 root    qmail   512 Apr  3  2001 .
drwxr-xr-x  20 root    wheel   512 Sep 13  2001 ..
drwxr-sr-x   2 alias   qmail   512 Jan 30 10:19 alias
drwxr-xr-x   2 root    qmail  1536 Sep 13  2001 bin
drwxr-xr-x   2 root    qmail   512 Sep 13  2001 boot
drwxr-xr-x   2 root    qmail   512 Feb 22 10:32 control
drwxr-xr-x   3 root    qmail  1536 Feb 27 10:20 doc
drwxr-xr-x  10 root    qmail   512 Apr 20 04:17 man
drwxr-x---  11 qmailq  qmail   512 Sep 13  2001 queue
drwxr-xr-x   2 root    qmail   512 Sep 13  2001 users
```

Wir berücksichtigen auch den Zusammenhang mit den weiter oben eingeführten Qmail-User und erkennen folgende Unterverzeichnisse:

- `/var/qmail/alias/` - Home-Directory des Qmail-Alias-Users. Der Eigentümer ist hier selbstverständlich alias und das Verzeichnis ist für die

Gruppe qmail "sticky" (Mode 2755). Hierdurch wird sichergestellt, das allgemeine User hier Dateien einstellen, aber nicht löschen können.

- `/var/qmail/bin/` - Hier liegen die ausführbaren Qmail-Dateien.
- `/var/qmail/boot/` - Dan Bernstein hat hier einige (mitterlweile weitgehend obsolete) Bootskripte untergebracht.
- `/var/qmail/control/` - Das Verzeichnis für die Qmail-Konfigurationsdateien.
- `/var/qmail/man/` - Ablageort für die Qmail man-pages.
- `/var/qmail/queue/` - Stark untergliedertes Verzeichnis für die Qmail-Queue mit geschütztem Lese-Zugirff.
- `/var/qmail/users/` - Verzeichnis der speziellen Qmail-User.

6.2.1 Erweiterte Konfiguration

Dan Bernstein hat seine Software-Pakete so installiert, dass diese einerseits schnell angepasst werden können, andererseits aber dennoch seinen Sicherheitsgarantien entsprechen. Hierzu dienen im Qmail-Source-Verzeichnis die Dateien conf-*, deren Bedeutung hier vorgestellt werden soll:

Der erste Block der conf-Dateien beschreibt das Installationsverhalten von Qmail:

- `conf-qmail`: Festlegen des Home-Verzeichnisses von qmail (Default: `/var/qmail`). Dies sollte ein lokales Dateisystem sein Gänze sein.
- `conf-users`: Die Namen der Qmail-User (wie weiter oben besprochen). Diese können hier nach Wunsch angepasst werden.
- `conf-groups`: Die Namen der beiden Qmail-Gruppen *nogroup* und *nofiles*. Mitglieder der Gruppe *nofiles* sollten keine eigenen Dateien besitzen, da diese Gruppe ausschliesslich für Qmail-Prozesse genutzt wird, was daher auch den Eintrag der üblichen Unix-Gruppen *nobody* und *nogroup* ausschliesst.
- `conf-cc`: Die C-Compiler-Optionen bzw. Optimierung (Default: `cc -02`).
- `conf-ld`: Die Optionen des Linkers (Default: `cc -d`).

Performance-relevante Optionen werden in zwei weiteren `conf`-Dateien festgelegt:

- `conf-spawn`: Das "Silent Concurrency Limit", mit der beschrieben wird, wievele File-Descriptoren pro `select()`-Statement (und pro Unix-User)

maximal offen sein dürfen. Hierdurch wird für **qmail-rspawn** und **qmail-lspawn** auch gleichzeitig die obere Grenze der Anzahl der Kindprozesse festgelegt. Der Default ist auf 120 gesetzt, der in den C-Header-Dateien verankerte Maximalwert beträgt 255. Dieser Wert bestimmt auch die Obergrenze für die Konfigurationsdateien `concorrencylocal` und `concurrencyremote`.

- `conf-split`: Dieser Parameter legt fest, wieviele Unterverzeichnisse die einzelnen Bestandteile der Qmail-Queue (`/var/qmail/queue/../`) enthalten (Default 23). Damit wird die Anzahl der Verzeichnisse festgelegt, auf die E-Mails in die Qmail-Queue verteilt. Diese Zahl muss immer eine Primzahl sein.

Die Laufzeit-Optionen bestimmen, welche Defaults Qmail bei der Auslieferung lokal zuzustellender E-Mails benutzt:

- `conf-patrn`: Der Parameter "Paternalism". Vergleichbar dem Unix-Befehl **umask** werden hier (per Ausschlussverfahren) die Rechte der Home-Directories und `.qmail`-Dateien festgelegt, die diese *nicht* nicht aufweisen dürfen, damit Qmail E-Mails lokal zustellt. Der Default-Wert 002 legt also fest, dass die benutzerspezifischen Heimat-Verzeichnisse und die `.qmail`-Dateien *nicht* World-Writeable sein dürfen. Ist dies der Fall, verweigert Qmail die Auslieferung.
- `conf-break`: Durch diesen Parameter wird die Qmail-Extension beschrieben; per Default das Zeichen "-". Dieses Zeichen kann aber im laufenden Qmail-System individuell konfiguiert werden.

In der Regeln sollten die performance-relevanten Einstellungen für Qmail bei einem E-Mail Aufkommen von mehr als 100.000 E-Mails/Tag bzw. bei 12.000 simultan zu übertragenden E-Mails angepasst werden, und bilden die Grundlage des "big-todo" Patches. Im Gegensatz zu Änderungen (Patches) der Qmail-C-Sourcen, die über das **make**-Utility automatisch erkennt werden, sind Modifikationen der conf-Dateien vor dem **make** vorzunehmen. Ändert man die gewünschten Enstellungen anschliessend, führt dies — bei gleichen C-Sourcen — zu keinem Erfolg. Notwendig ist also die Änderung vor dem anschliessend Compiler-Lauf.

6.2.2 Qmail-Patche?

Qmail 1.03 wurde von Dan Bernstein 1998 freigegeben und wird seitdem unverändert angeboten. Es gibt keinen relevanten RFC, dem Qmail nicht entspricht. Insofern kann auch Vanilla-Qmail in seiner Ursprungsform bedenkenlos eingestetz werden. Prinzipiell können alle Ergänzungen — seien es Viren-Scanner oder Spam-Filter — durch eine individuelle Anpassung des Qmail-Zustellprozesses (**qmail-local**) eingebunden werden.

Seit Anfang 2003 finden sich auf der Qmail Homepage sog. "Recommended Patche". Hierunter werden aufgeführt:

Ein gcc/glibc Patch für den neuen GNU 3.x c-Compiler.

Ein Patch gegen qmail -local.c

Das sog. ipme patch oder

An eine Grenze kommen diese Modifikationen dann, wenn es darum gilt

- zusätzliche Log-Informationen aufzuzeichnen,
- Änderungen des SMTP-Dialog vorzunehmen,
- Unverträglichkeiten mit nicht RFC-konformenen Produkten zu beseitigen sowie
- umfangreiche Ergänzungen einzupflegen.

Die wichtigsten Patche sind auf der Qmail Home-Page nach Themengebiete sortiert zu finden. Nur: Die Logik lässt sich manchmal kaum nachvollziehen. Wichtige Patche wie das "POP3-before-SMTP" fallen unter die Rubrik "checkpassword" Modifikationen.

Ein weiteres Problem ergibt sich zwangsläufig bei der Implementierung mehrerer Patche gegen die gleichen C-Sourcen. Die Resultate können in diesem Fall nicht vorhergesehen werden. So gibt es z.B. Unverträglichkeiten bei meinem SPAMCONTROL-Patch mit der Qmail-LDAP-Ergänzung.

Grundsätzlich gilt daher: Es werden nur Patche eingesetzt, falls keine anderen Massnahmen greifen. Keinesfalls sollten Patche "blind" eingespielt werden. In der Regel führt auch jeder Author von Patches eine Versionskennung durch und dokumentiert seine Ergänzungen/Änderungen in einem zugehörigen README, dass auf jeden Fall vor dem Einsatz des Patches sehr genau konsultiert werden muss. Der Einsatz nicht-dokumentierte Patche ist ein Seitensprung ohne Kondom.

Ein erfahrener System-Administrator ist mit den Werkzeugen **diff** und **patch** vertraut und kann ggf. selbst auch Hand an die Änderungen legen. Qmail-Patche werden aber in der Regel dort eingesetzt, wo Qmail als SMTP-Gateway zum Internet eingesetzt wird; speziell bei Internet Service Providern (ISPs). Während der Boom-Phase des Internet Anno 2000 wurden aber häufig wenig-qualifizierte Personen zu Unix-System-Administratoren benannt, die — dem guten Ruf von Qmail folgend — ihre bestehende Sendmail- gegen eine Qmail-Installation austauschten, in der Hoffnung, dass Qmail die infrastrukturellen und organisatorischen Problem löst.

An dieser Stelle sei von meiner Seite noch einmal eine Klarstellung angebracht: Qmail, Sendmail oder auch Postfix sind *Werkzeuge*; zu ihrer Beherrschung

bedarf es *Methoden*, die jeweils individuell für das Werkzeug anzueignen sind. Dies gilt auch für Qmail-Patche. Häufig gibt es es für ein Problem mehrere Lösungswege. Z.B. im Bereich Spam-Unterdrückung bietet die Qmail-Homepage bestimmt ein Dutzend Patche an. Mit keiner der vorgestellten Lösungen lässt sich ein 100%iges Spam-Filtern ermöglichen. oSpam, SpamAssassin, SPAMCONTROL und wie sie alle heissen, sind Werkzeuge zur Spam-Unterdrückung, die jeweils eigenen Überlegungen und Methoden folgen, und die häufig für ein spezielles Problem eine Lösung darstellen. Diese können aber nicht unbedingt verallgemeinert werden und lösen im besonderen nicht unbedingt die drängende lokale Spam-Situation. Ferner verletzen diese Patche Werkzeuge (wie z.B. mein SPAMCONTROL-Patch) die Regeln des SMTP-Protokolls und können sich daher auch kontraproduktiv auswirken, speziell wenn sie in einem Umfeld eingesetzt werden, wo potentielle Kommunikationspartner dies nicht erwarten.

Der System-Administrator, der bestimmte Patches für Qmail einsetzt, ist in einer der eines Formel-1-Piloten vergleichbaren Situation: Dieser muss sein Fahrzeug kennen (das Werkzeug), aber auch in einer konkreten Situaton entscheiden, wie sein Rennwagen zu tunen (patchen) ist. Kein Rennen ist wie das andere, es gibt neue Konkurrenten (Spammer) und wenn auf der gleichen Strecke wie im Vorjahr gefahren wird, sind doch die Witterungsumstände unterschiedlich.

6.2.3 Checkpassword

Beim Einsatz des Qmail-eigenen POP3-Servers **qmail-pop3d** benötigt man ein zusätzliches Programm, dass Benutzer-Authentisierung in Form einer Standard-Passwort-Abfrage vornimmt und POP3/IMAP4-konforme Antworten zurück gibt. Dies stellt Dan Bernstein als eigenes Programmpaket Checkpassword bereit. Hierdurch befreit er den POP3-Daemon von dieser Aufgabe und ermöglicht somit gleichzeitig die Unterstützung unterschiedlicher Authentisierungsmethoden.

Das Checkpassword-Programm liegt ebenfalls als Tar-Achiv vor: `checkpassword-0_81_tar.gz`. Wir kopieren es ins Verzeichnis `/usr/local/src/` und entpacken es. Hierdurch wird das Unterverzeichnis `./checkpassword-0.81` erzeugt, in das wir wechseln. Auch hier werfen wir einen Blick in die ausgepackten Sourcen und stellen auch Dan Bernstein's Standard-Struktur fest. Als einziges ausführbares Programm wird hier **checkpassword** erzeugt. Neben einigen Dokumenten (`ls -la [A-Z]*`) können wir auch über die Datei `conf-home` festlegen, in welchem Verzeihnis **checkpassword** installiert werden soll, normalerweise ist dies `/bin/`.

Die Installation von **checkpassword** ist denkbar einfach. Entsprechend der Beschreibung `INSTALL` genügt:

```
% make
```

```
% su
# make setup check
```

und uns stehen sowohl die Binärdatei **checkpassword** als auch eine entsprechende man-page zur Verfügung. Abschliessend werfen wir einen Blick auf den Dateimode von **checkpassword**:

```
% ls -la /bin/checkpassword
-rwx------  1 root  wheel  4640 Dec  7 10:25
/bin/checkpassword
```

und erkennen, dass nur der Superuser das Recht hat, diese Datei auszuführen.

Die man-page von **checkpassword** gibt fälschlicherweise noch die Versionsnummer 0.80 aus.

6.2.4 Fastforward

Das Programmpaket Fastforward dient dazu, einen cdb-Datenbank aufzubauen, in den das Forwarding für "lokale" E-Mail-Adressen mittels des Qmail-Programms **forward** geregelt. wird. Dieses Verfahren ist äquivalent dem **aliases**-Mechanismus von Sendmail und tatsächlich sind die Eingabe-Dateien auch kompatibel. Hierdurch ist es möglich, die Forwarding-Listen (`/etc/aliases`) aus einer bestehenden Sendmail-Installation zu nutzen und ohne inhaltliche Änderungen nach Qmail zu migrieren. Fastforward ersetzt komplett das ursprünglich von Dan Bernstein entwickelte Qmsmac (`qmsmac-0.71.tgz`), das er nicht mehr anbietet.

Fastforward steht in der Version 0.51 als Tar-Archiv zur Verfügung (`http://cr.yp.to/fastforward.html`). Nach dem Auspacken des Archivs (unter `/usr/local/src/`) und Wechsel ins Verzeichnis `fastforward-0.51` geschieht die Installation in einen einzigen Schritt (als *root*):

```
# make setup check
```

Die Binaries werden im Qmail `/var/qmail/bin/` Verzeichnis abgelegt, die man-pages für **fastforward**, **newinclude**, **newaliases**, **printforward**, **printmaillist**, **setmaillist** und **setforward** bei den Qmail man-pages. Sollte Qmail an einem anderen Ort installiert sein, lässt sich dies über die Datei `conf-qmail` anpassen.

6.3 Post-Installation

Nach dem alle Elemente von Qmail zur Verfügung stehen, kommt es darauf an, Qmail zu konfigurieren und es nutzbar in ein Administrationskonzept einzubetten. Auch hier sind einige Vorüberlegungen anzustellen:

- Von welchem (lokalen) Account administriere ich Qmail?
- Wie organisiere ich den Start/Stopp von Qmail?
- Wie sollen die lokal zugestellten E-Mails abgelegt werden?
- Welches Logging-Verfahren setze ich für Qmail ein?

Wir wollen diese Fragen schrittweise klären, doch zunächst besteht ein Bedarf an initialer Konfiguration, dem wir vor dem ersten Start von Qmail entsprechen müssen.

6.3.1 Basiskonfiguration

Dan Bernstein hat es uns mit der Erst-Konfiguration von Qmail einfach gemacht, indem er die Skript **config** und **config-fast** zur Verfügung stellt. Während beim Skript **config-fast** ein (beliebig) wählbarer FQDN als Argument anzugeben ist, ermittelt im Gegensatz hierzu **config** den lokalen Hostname und desssen IP-Adresse(n) über einen DNS-Lookup:

```
# ./config

Your hostname is qmailer.fehnet.de.

Your host's fully qualified name in DNS is
qmailer.fehnet.de.

Putting qmailer.fehnet.de into control/me...

Putting fehnet.de into control/defaultdomain...

Putting fehnet.de into control/plusdomain...

Checking local IP addresses:

127.0.0.1: Adding localhost to control/locals...

192.168.192.2: Adding qmailer.fehnet.de to control/locals...

If there are any other domain names that point to you,

you will have to add them to /var/qmail/control/locals.

You don't have to worry about aliases, i.e., domains with
CNAME records.

Copying /var/qmail/control/locals to
/var/qmail/control/rcpthosts...

Now qmail will refuse to accept SMTP messages except to
those hosts.

Make sure to change rcpthosts if you add hosts to locals or
virtualdomains!
```

Beide Skripte befüllen die Konfigurationsdateien `/var/qmail/control/me` und `/var/qmail/control/rcpthosts` mit dem ermittelten bzw.

übergebenen Hostname. Ferner wird in die Dateien `./defaultdomain` und `./plusdomain` mit die Domain-Kennung eingefügt. Unterschiedlich verhalten sich beide Skripte, wenn es um die Kontrolldatei `./locals` geht. Hier fügt **config-fast** den FQDN Hostanme ein, während **config** die im DNS verfügbaren Namen *aller* lokalen IP-Adressen des Rechners einträgt.

Abschliessend ist in das Verzeichnis `/var/qmail/alias/` zu wechseln und es sind dort die wichtigen Qmail-Alias-User einzurichten:

```
# touch .qmail-postmaster
# touch .qmail-mailer-daemon
# touch .qmail-root
# chmod 644 ~alias/.qmail*
```

Diese Alias-Dateien sind zunächst leer und erfüllen ersteinmal ihren Dienst darin, dass Qmail z.B. bei E-Mail an Root keine Fehlermeldung im Log produziert. Ich empfehle aber dringend, diese Alias-Dateien mit einer konkreten E-Mail-Adresse zu befüllen: dem lokalen Qmail-Administrator. Lautet dessen lokaler Account z.B. *qmailadmin*, genügt ein Eintrag in jede dieser Alias-Dateien:

```
&qmailadmin
```

Dies veranlasst den Qmail-User *alias*, E-Mails an *root* an den (lokalen) Account *qmailadmin* weiter zu reichen; ebenso erhalten wir hiermit alle nicht-zustellbaren Bounce-E-Mails. Voraussetzung ist natürlich, dass für den Account *qmailadmin* ein lokalen Postfach eingerichtet wurde, was aber weiter unten beschrieben ist.

Im Anschluss an diese Massnahmen verfügen wir ein ein vollständig funktionsfähiges Qmail-System, dass allerdings noch "schläft" also nicht gestartet ist. Bevor wir dies tun, werfen wir noch Blick in die Konfiguration von Qmail und müssen noch einige "Aufräumarbeiten" durchführen.

6.3.2 man-pages

Neben der Dokumentation von Qmail, die Dan Bernstein im Verzeichnis `/var/qmail/doc/` hinterlegt hat, bilden die Qmail man-pages die wichtigste Quelle für das Verständnis aber vor allem dem Einsatz von Qmail. Da wir den Pfad der Qmail man-pages korrekt eingerichtet haben, stehen uns diese Informationen bequem per man-Aufruf zur Verfügung. Guter Startpunkt ist die man-page `qmail`. Weitere unbedingt zu lesende man-pages sind `qmail-local`, `qmail-send`, `qmail-inject`, `qmail-remote` und `qmail-smtpd`. Einen Gesamtüberblick über die Qmail Konfigurationsdatein wird übersichtlich in `qmail-control` vermittelt und das Logging von Qmail in `qmail-log` erläutert.

Leider hat Dan Bernstein das Konzept der man-pages bei seinen neueren

Software-Produkten nicht mehr weiter verfolgt, sondern verweist auf die HTML-Dokumente auf seinen Web-Seiten.

6.3.3 Kontrolldateien

Die im Verzeichnis `/var/qmail/control/` vorhandenen Kontrolldateien bilden die zentrale Konfigurationsbasis von Qmail. Auf diese Konfigurationseinstellungen greifen unterschiedliche Programmteile von Qmail zu, wobei die Datei `me`, d.h. die Identifikation des Qmail-MTA das absolute Minimum darstellt und von den zentralen Qmail-Programmn wie **qmail-send**, **qmai-smtpd** und **qmail-remote** ausgewertet werden. Wir folgen hier bei den Beispieladressen den Vorgaben von Dan Berstein, um die Zuordnung zu seiner Dokumentation zu erleichtern.

Die Kontrolldateien für **qmail-send** im einzelnen:

- `bouncefrom`
  Vergebener Name für Single-Bounce Nachrichten (Default: MAILER-DAEMON) im E-Mail-Header; der SMTP-Sender ist unbesetzt (MAIL FROM: <>).
- `bouncehost`
  Name des Bounce-Host. (Default: `me`) **qmail-send** sendet eine Single-Bounce Nachricht zurück unter dem Absender "`From: bouncefrom@bouncehost`".
- `concurrencylocal`
  Maximale Anzahl der lokalen Auslieferungsversuche (Default: 10, Maximum 120).
- `concurrencyremote`
  Maximale Anzahl der Auslieferungsversuche für entfernte Rechner. (Default: 20, Maximum 120).
- `doublebouncehost`
  An den hier angegebenen Hostname werden doppelt loopende (Double-Bounce) E-Mails zugestellt (Default: `me`).
- `doublebounceto`
  Adresse des *Empfängers* für doppelte Bounce-E-Mails (Default: *Postmaster*).
- `envnoathost`
  Domainbezeichnung für lokal abgeschickte E-Mails ohne `@` (Default: `me`). Ansonsten wird die vollständige Adresse gebildet als `...@envnoathost`.

- `locals`
  Liste der Domänen, für die lokal zuzustellende E-Mails (Default: `me`).

- `percenthack`
  Liste der Domänen für die der "Percent Hack" angewandt wird. Wird hierin eine Domäne aufgenommen so wird jede Adresse in der Form `user%fqdn@domain` umgeschrieben als `user@fqdn`. Der "Percent Hack" kann iterativ eingestezt werden, da auch `user` ein Prozentzeichen enthalten darf. Diese Datei wird vor `locals` ausgewertet.

- `queuelifetime`
  Anzahl der Sekunden, wie lange eine E-Mail in der Warteschlange verbleiben darf (Default: 604800 Sekunden bzw. eine Woche). Nach Verstreichen dieser Zeit versucht **qmail-send** ein letztes Mal die E-Mail zu versenden und gibt bei Nicht-Erfolg auf.

- `virtualdomains`
  Liste sog. virtueller Domänen in From von `domain:prepend` ohne zusätzliche Leerbuchstaben. Sieht **qmail-send** einen Empfänger für diese Domäne, z.B., `user@domain`, ersetzt es den Domän-Namen durch Hinzufügen der Bezeichnung für die virtuelle Domäne und betrachtet die E-Mail als lokal zuzustellen. Beispielsweise ist `nowhere.mil:joe-foo` in `virtualdomains`. Eine E-Mail für `info@nowhere.mil` wird von **qmail-send** als `joe-foo-info@nowhere.mil` umgeschrieben und lokal ausgeliefert. `virtualdomains` erlaubt die Nutzung von Wildcards:

```
.fax:uucp-fax
:alias-catchall
.nowhere.mil:joe-foo-host
```

  In `virtualdomains` aufgelistete Domänen mit "leerer" virtueller Bezeichnung werden als Ausnahmen betrachtet. Da `virtualdomains` nach `locals` ausgewertet wird, haben (evtl. doppelt vorhandene) Einträge in `locals` Vorrang.

Diese Kontrolldateien werden beim Start von **qmail-send** eingelesen und stehen unverändert während der Laufzeit von Qmail zur Verfügung. Änderungen werden erst dann aktiv, wenn entweder **qmail-send** neu gestartet wird oder ein HUP-Signal empfängt.

Im Gegensatz zum Daemon-Prozess **qmail-send**, werden **qmail-smtpd**, **qmail-inject** oder **qmail-remote** bei Bedarf gestartet. Änderungen an den zugehörigen Kontrolldateien werden somit quasi on-the-fly wirksam. Der Qmail SMTP-Server **qmai-smtpd** wird über folgende (optionalen) Kontrolldateien konfiguriert.

- `badmailfrom`
  E-Mails von Absendern, die in dieser Datei aufgelistet sind, werden nicht

akzeptiert (Negativliste).

- `databytes`
  Maximale Grösse (in Bytes) einer einlaufenden E-Mail. Wird für eine E-Mail diese Grösse überschritten, meldet **qmail-smtpd** dies als permanenten Fehler dem Sender. Defaultwert ist 0, d.h. alle E-Mails werden akzeptiert.
- `localiphost`
  Bedarfsweise kann hiermit der Name des lokalen Qmail MTAs für einlaufende E-Mails, die nicht via Domain-Name, sondern per IP-Adresse `...@[1.2.3.4]` adressiert sind, auf den hierin hinterlegten Hostname abgebildet werden. Dieses Verfahren greift vor der Datei `rcpthosts`.
- `morercpthosts`
  erweitert die `rcpthosts` Liste um dynamisch hinzugefügte Domänen in Form einer cdb, wozu das Kommando **qmail-newmrh** herangezogen werden muss.
- `rcpthosts`
  Im Kontrast zur Konfigurationsdatei `badmailfrom` werden durch Befüllen der Datei `rcpthosts` einlaufende E-Mails nur für die hierin aufgeführten Hosts akzeptiert (Positivliste). Ist diese Datei nicht existent, werden alle E-Mails akzeptiert und der Qmail-MTA ist somit ein offenes Relay.
- `smtpgreeting`
  Legt den "Begrüssungsspruch" von Qmail fest.
- `timeoutsmtpd`
  Zur Festlegung der Sekunden für **qmail-smtpd** zur Pufferung neuer E-Mails eines anderen MTAs. Default ist 1200 Sekunden.

qmail-remote, d.h. der SMTP-Client stützt sich auf folgende Kontrolldateien:

- `helohost`
  Der logische Verbindungsname zum SMTP-Server, in der Regel der Hostname des Qmail-MTA. Default: `me`.
- `smtproutes`
  Fest vorgegebener Pfad zum SMTP-Empfänger, in Ergänzung/Änderung der Ermittlung des üblichen SMTP-Pfades per DNS-MX-Lookup, wobei zusätzlich eine TCP-Portnummer mit angegeben werden kann. Beispiele:

```
inside.af.mil:firewall.af.mil:26
.af.mil:
:heaven.af.mil
```

  Im ersten Fall werden E-Mails für Empfänger in der Domain `inside.af.mil` an den Rechner `firewall.af.mil` ans Port 26

übermittelt. Im zweiten Fall werden alle E-Mails, die mit der Empfängeradresse `.af.mil` enden (aber nicht `af.mil` selbst), per MX-Lookup zugestellt; der dritte Eintrag legt fest, dass jede ausgehende E-Mail, deren Adresse nicht einer der ersten beiden Kriteiren entspricht, per Default an den Rechner `heaven.al.mil` zustellt werden.

- `timeoutconnect`
  Die Zahl der Sekunden, die **qmail-remote** wartet bis der SMTP-Server antwortet, per Default 60 Sekunden.
- `timeoutremote`
  Die maximale Zahl der Sekunden, die **qmail-remote** bis zum Empfang des nächsten SMTP-Kommandos verstreichen lässt; der Defaultwert beträgt 1200 Sekunden (20 Minuten).

qmail-inject ist der lokale E-Mail-Client von Qmail, der sich auf die folgenden Kontrolldateien stützen kann:

- `defaultdomain`
  Angabe der Defaultdomain des Rechners, in der Regel also der Domain-Teil des Rechner-FQDN, der überlicherweise in der Datei `me` festgelegt ist. Diese Domain-Kennung wird bei Angabe eines unqualifizierten Hostname hinzugefügt.
- `defaulthost`
  Angabe des Default-Hostname, typischerweise der FQDN, der in der Datei `me` steht.
- `idhost`
  Der Name, der als Message-ID in einer auslaufenden E-Mail geschrieben wird, normaler der Name, der in der Datei `me` spezifiziert ist, oder ein sonstiger (beliebiger) Name mit korrekter Domain-Bezeichnung.
- `plusdomain`
  Der hierin hinterlegte Name wird von **qmail-inject** all den Hostname auf der Kommandozeile hinzugefügt, die mit einem Plus-Zeichen "+" enden.

Dan Bernstein's *Quick Mail Queueing Protocol* Client hat die Aufgabe, E-Mails nicht in eine lokale Queue zu stellen, sondern zu einem QMQP-Server zu übertragen. **qmail-qmqpc** braucht also folgende Informationen:

- `qmqpservers`
  Liste von IP-Adressen der QMQP-Server, zu denen E-Mails ausgeliefert werden sollen.

Bei der Befüllung der Qmail Kontrolldateien ist eine gewisse Vorsicht angebracht. Der Inhalt dieser Dateien wird einerseits ohne syntaktische Prüfung eingelesen, andererseits sind auch Kommentarzeilen nur in den Dateien

`badmailfrom`, `locals`, `percenthack`, `qmqpservers`, `rcpthosts`, `smtproutes` und `virtualdomains` gestattet. Kommentarzeilen beginnen grundsätzlich mit einem "#" als erstes Zeichen in der Zeile. Leerzeichen und Tabulatoren am Ende einer Zeile werden ignoriert.

Ungünstig ist auch, leere Kontrolldateien zu erzeugen. Teilweise interpretiert Qmail dann nicht den Inhalt, sondern nimmt den Namen der Kontrolldatei. Den Inhalt der Kontrolldatei kann über das Qmail-Kommando **qmail-showctl** ausgebeben werden. Dies ist ein wichtiges Instrument, sowohl den aktuellen Status zu dokumentieren, als auch im Fehlerfalle die notwendigen Konfigurations-Informationen bereit zu stellen.

Domain-Wildcards

Bei der Eingabe von Domain-Namen folgt Dan Bernstein einer eigenen Syntax, was mit dem Parsing der eingelesenen Kontrolldateien zusammen hängt. So bezeichnet z.B. `af.mil` genau die Domain "AF.MIL", während `.af.mil` die untergeordneten Hosts und Subdomänen adressiert. Will man beispielsweise einen Qmail-MTA aufsetzen, der die E-Mails für alle in der Domaine "AF.MIL" beheimateten Rechner und Benutzer entgegen nehmen soll, so ist die Datei `rcpthosts` wie folgt aufzubauen:

```
af.mil
.af.mil
```

Erst mit letztem Statement wird sichergestellt, dass z.B. auch eine an `@inside.af.mil` adressierte E-Mail in Empfang genommen wird.

Adressen

Abschliessend ist zu berücksichtigen, dass es bei allen genannten Adressen, ausschliesslich um Adressen im SMTP-Umschlag d.h. DNS-Adressen handelt. Die in der ausgelieferten E-Mail zu findenden Angaben, müssen hierzu keine Entsprechung besitzen und können ggf. frei gewählt sein

6.3.4 Sendmail entfernen

Wer Qmail will, sollte Sendmail entfernen. Dies geschieht in mehreren Schritten, die von Dan Bernstein im Dokument `REMOVE.sendmail` festgehalten sind:

1. Der aktuell laufende sendmail-Prozess is per `ps -aux | grep sendmail` bzw. `ps -ef | grep sendmail` zu identifizieren und zu beenden. War **sendmail** aktiv, d.h. mit der Auslieferung von E-Mails beschäfigt sollte dies über das Stoppen von **sendmail** über das STOP-Signal erfolgen, damit ausstehende E-Mails noch sauber prozessiert werden können.

2. Anschliessend ist zu ermitteln, wie **sendmail** gestartet wird. Typischerweise geschieht dies bei System V Unix'en über ein rc-Runlevel Skript, typischerweise `/etc/init.d/sendmail`. Möglicherweise liegt der Start über den `rc.local` Mechanismus gestartet. In jedem Fall sind ist das

sendmail Start-Skript entweder zu komplett zu löschen, oder durch ein **chmod**-Befehl seiner Lese- und Ausführungsrechte zu berauben, z.B.:

```
# chmod -rx /etc/init.d/sendmail
```

Bei SuSE-Linux sollte der Start von **sendmail** über die Yast-Konfiguration bzw. über die Datei `rc.suse` unterbunden werden. Vergleichbares gilt für FreeBSD, wo der Start von **sendmail** in der Konfigurationsdatei `/etc/rc` eingebunden ist und per `/etc/rc.conf` getriggert wird.

3. Als nächsten suchen wir die **sendmail** Binärdatei mittels der Befehle `which sendmail` bzw. `locate sendmail` und wechseln in dieses Verzeichnis (z.B. `/usr/libexec/` oder `/usr/sbin/` ...). Es ist sinvoll, diese Datei umzubenennen, damit sie nicht mehr aufgerufen werden kann, und auch ihre Ausführungsrechte und speziell das SUID-Bit zurückzusetzen.

```
# chmod 0000 sendmail
# mv sendmail sendmail.org
```

4. Einige systemnahe Dienste, wie z.B. **cron**, haben **sendmail** als Kommando zum Absetzen von E-Mails fest verankert. Wir müssen also einen sinnvollen Ersatz anbieten. Dies geschieht mittels des Qmail **sendmail** Wrappers. Diesen setzen wir über eines symbolischen Link an ursprüngliche Stelle von **sendmail**:

```
# ln -s /var/qmail/bin/sendmail sendmail
```

5. Möglicherweise sind noch einige Abschlussarbeiten notwendig, z.B. das Entfernen einer **sendmail** PID-Datei (überlicherweise unter `/var/run/`) oder auch das Aufräumen des **sendmail** Logfiles z.B. `/var/log/mail` bzw. `/var/log/messages`.

6.3.5 VSM

Unter dem Kürzel VSM wird das typische Sendmail-Verzeichnis `/var/spool/mail/` bezeichnet. Dies ist das Verzeichnis, wo üblicherweise **sendmail** die für einen User auszuliefernde E-Mail zunächst ablegt. E-Mail-Clientes wie **mail** überführen diese E-Mails anschliessend in die lokale Mailbox, die üblicherweise `$HOME/mail` oder `$HOME/mbox` lautet.

Als Mailbox wird die Organisation der E-Mails in einer gemeinsamen Datei verstanden. Die E-Mails werden hierbei nicht als separate Dateien, sondern als konkatinierter File abgelegt — neue E-Mails werden einfach in der Datei `mail` bzw. `mbox` angehängt. Zu den Zeiten kleiner E-Mails und wenig Speicherplatz und Systemleistung vielleicht verständlich; heutzutage aber ein Unding. Zudem ist das Ablegen der ungelesenen E-Mails in einem einzigen Verzeichnis und in benutzerspezifischen Dateien vom Sicherheitsaspekt her indistutabel. Mit einem

einfachen Kommando kann jeder Benutzer im System abfragen und festhalten, wann neue E-Mails für einen beliebigen anderen Benutzer eingetroffen sind.

E-Mails sind im Sinne von Dan Berstein stets streng privat; sie werden vom System (**qmail-local**) im Heimat-Verzeichnis des Adressaten abgelegt. Die Organisation der E-Mails kann entweder dem konventionellen Mailbox (mbox) Format folgen, oder im einem Mailverzeichnis (Maildir) vorgenommen werden.

Bei der Migration von Sendmail nach Qmail sind daher folgende Schritte vorzunehmen:

1. Eventuell vorhandene E-Mails im VSM-Verzeichnis sind in die Home-Directories der Empfänger zu überführen.
2. Es sollte sichergestellt werden, dass die lokalen Mail User Agents (MUA) über das Setzen der Umgebungsvariable `$MAIL` über diese Änderung informiert werden, wobei zu berücksichtigen ist, dass erste die neueste Generation der MUAs das Maildir-Format unterstützten. Häufig findet sich die Default-Einstellung für die `$MAIL` Umgebungsvariable in der Datei `/etc/login.conf`.
3. E-Mails in den alten Mailbox-Formaten müssen ggf. in das neue Maildir-Format konvertiert werden.
4. Werden zum Einrichten von Benutzer-Accounts Skripte eingesetzt, ist sicherzustellen, dass diese im Home-Directory die initialen Mailboxen bzw. Mailverzeichnisse erzeugen; keinesfalls jedoch das benutzerspezifischen Dateien im nun obsoleten `/var/spool/mail/`-Verzeichnis generieren.

Auf die genaue Spezifikation der Qmail Mailboxen und Mailverzeichnisse und deren (friedliche) Koexistenz wird noch weiter unten eingegangen.

6.4 Qmail Start-Up

Qmail besteht initial aus drei Daemon-Diensten **qmail-start**, **qmail-smtpd** und **qmail-pop3d**. Da **qmail-smtpd** und **qmail-pop3d** aus sich heraus weder auf einem Port binden noch über einen DNS-Stubb-Resolver verfügen, sind diese Dienste zwangsläufig über einen sog. Superdaemon zu starten, sei es nun INETD, XINETD oder **tcpserver**. Während **qmail-pop3d** den POP3-Server von Qmail darstellt (der sich auch durch einen IMAP4-Server ergänzen bzw. ersetzen lässt), ist **qmail-start** von übergeordneter Bedeutung, da hierüber das gesamte Qmail-System initialisiert wird.

Hierbei wurde der zentrale Dienste **qmail-start** über ein Shell-Start-Up-Skript als Hintergrundprozesse gestartet. Obwohl von Dan Bernstein schon frühzeitig mit dem Packet UCSPI den **tcpserver** zur Verfügung gestellt wurde, wurden, fand die Einbindung von **qmail-smtpd** und **qmail-pop3d** traditionell über den

INETD statt, was auch in der Qmail Dokumentation beschrieben ist.

Speziell mit der Verfügbarkeit des **rblsmtp**-Werkzeugs als Teil des UCSPI-Paketes fand der **tcpserver** beim Aufruf von **qmail-smtpd** und **qmail-pop3d** verbreiteteren Einsatz. Somit war der Zustand geschaffen, Qmail entweder über ein "grosses" Skript, das alle Dienste beinhaltet, zu starten, oder über partikuläre Skripte. In jedem Fall existieren somit drei Daemonprozesse, **qmail-start, qmail-smtpd** und **qmail-pop3d**, die quasi unkontrolliert im Unix-System agieren.

Durch die Entwicklung der Daemontools hat Dan Bernstein einen weiteren Meilenstein geschaffen, eine einheitliche Plattform für Systemdienste zu realisieren. Dieser stellt die geeignete Basis dar, Qmail zu einer Hochverfügbarkeitsanwendung werden zu lassen. Doch wir wollen systematisch vorgehen und stellen die Alternativen einzeln vor.

6.4.1 Standard-Qmail

Bei der Standard-Installation wird nur Qmail als Ersatz für z.B. Sendmail eingesetzt. Hierbei wird (als erster Schritt) **qmail-start** über ein Skript aufgerufen und in den Hintergrund gestekt (Daemon) und **qmail-smtpd** und **qmail-pop3d** typischerweise in den Aufrauf des INETD eingebunden. Ergänzend wird beim Aufruf von **qmail-start** der Log-Output per **splogger** in den Syslog eingespeist.

```
env - PATH="/var/qmail/bin:$PATH" \
csh -cf 'qmail-start ./Mailbox splogger qmail &'
```

Listing 6-4: Einfachstes Qmail-Startup Skript.

Das Startup-Skript "säubert" über die Anweisung env - zunächst das Shell-Environment mit Ausnahme der PATH-Variable, die um die Angabe des Qmail-Pfades ergänzt wird. Anschliessend wird über den Shell-Aufruf `csh -c` das Programm **qmail-start** mit den Argumenten `./Mailbox` und **splogger** `qmail` gestartet und in den Hintergrund gestellt (&). Der csh-Parameter `-f` sorgt dafür, dass eine evtl. vorhandene `~./cshrc` Datei nicht interpretiert wird.

Dem aufgerufenen Programm **qmail-start** fallen hierbei fünf Aufgaben zu:

1. Es startet das zentrale Qmail-Programm **qmail-send**, d.h. den Qmail Sende-Daemon, der Nachrichten aus der Qmail-Queue liest.
2. Der Prozess **qmail-clean** wird initial aufgerufen, dem die Aufgabe zufällt, die Qmail-Queue aufzuräumen.
3. Der Daemon-Prozess **qmail-rspawn** wird initialisiert, der wiederum die

Auslieferung der in die Queue eingestellten Nachrichten an entfernte Empfänger triggert.

4. Der Daemon-Prozess **qmail-lspawn** wird mit der Instruktion einer sog. Default-Delivery aufgerufen. **qmail-lspawn** hat die Aufgabe, den lokalen Zustellagenten (**qmail-local**) zu triggern und ihm mitzuteilen, wohin dieser die zuzustellenden Nachrichten auszuliefern hat, falls keine benutzerpezifischen Angaben gemacht sind; in unserem Fall in die Datei `~/Mailbox`.

5. Mitteilungen und Logging der gestarteten Programme **qmail-send**, **qmail-rspawn** und **qmail-lspawn** über den File-Descriptor 2 werden an den **splogger** weitergereicht. Dessen Aufgabe ist es, die Ausgabe dem **syslogd** zur Verfügung zu stellen, wobei als Präfix der Meldungen der Parameter `qmail` eingestellt ist.

Nach dem Aufruf des Qmail-Startup-Skriptes beobachten wir folgende Prozesse im System:

```
% ps -auxww | grep qmail

qmaill     149  0.0  0.8   824   292  ?  S    08:52   0:00
splogger qmail

qmailq     152  0.0  0.6   816   220  ?  S    08:52   0:00
qmail-clean

qmailr     151  0.0  0.6   820   232  ?  S    08:52   0:00
qmail-rspawn

qmails     129  0.1  0.7   860   264  ?  S    08:52   0:02
qmail-send

root       150  0.0  0.6   820   232  ?  S    08:52   0:00
qmail-lspawn ./Mailbox
```

Wir sehen, dass entsprechend den Vorgaben der Qmail-User alle Prozesse mit dedizierter und jeweils unterschiedlicher Benutzerkennung im Unix-System laufen. Lediglich der Prozess **qmail-lspawn** findet unter der Benutzerkennung *root* statt. dies ist notwendig, da nur hierdurch der Wechsel in den aktuellen Benutzerkontext möglich ist. D.h. **qmail-lspawn** startet den Zustellprozess **qmail-local** (der Mail-Delivery Agent MDA) jeweils mit der UID/GID des Benutzters, zu dem die Nachricht zuzustellen ist.

Um die Migration einer bestehenden Sendmail-Installation auf eine Qmail-System zu vereinfachen, hat Dan Bernstein im Verzeichnis `/var/qmail/boot/` einige Startup-Skripte hinterlegt, die beispielhaft zeigen,

- wie Qmail weiterhin das Standard-Sendmail `/var/spool/mail/` Verzeichnis einschliesslich der häufig eingesetzte `.forward` Dateien nutzen kann

- wie der Zustelldienst von Qmail zusätzlich mit dem Programm **procmail**, und den hiermit gegebenen erweiterten Filtermöglichkeiten, kombiniert werden kann.

Typischerweise wird vorgeschlagen, eins dieser Skripte an die lokalen Gegebenheiten anzupassen und unter dem Namen `rc` in das Verzeichnis `/var/qmail/` zu kopieren. Dieses Skript wird als Ausgangspunkt für den Qmail-Start herangezogen und als System-Bootskript genutzt. Bei einem BSD-System würde man z.B. folgende Zeile in das Boot-Skript `/etc/rc` eintragen:

```
csh -cf '/var/qmail/rc &'
```

Run-Level-(Start/Stop-)Skripte in System-V Systemen (wie Linux) werden im Dokumente INSTALL hierbei nicht berücksichtigt.

In einem zweiten und dritten Schritt, sind die Qmail-Daemon-Prozesse **qmail-smtpd** und **qmail-pop3d** in den INETD zu integrieren. Hierzu reicht es aus, in der Konfigurationsdatei `/etc/inted.conf` folgende Zeilen unterzubringen:

```
...
smtp stream tcp nowait qmaild /var/qmail/bin/tcp-env   tcp-
env /var/qmail/bin/qmail-smtpd
....
pop3 stream tcp nowait root  /var/qmail/bin/qmail-popup
qmail-popup YOURHOST /bin/checkpassword
/var/qmail/bin/qmail-pop3d Mailbox
```

Listing 6-5: Auszug aus `inetd.conf`.

Wir wollen uns beide Zeilen etwas genauer anschauen:

- Bei Aufruf des Prozess **qmail-smtpd** wechselt **inetd** zunächst zu den Rechte den Benutzers *qmaild.* Der Parameter `smtp` sowie die Angabe `tcp` veranlasst **inetd**, den TPC-Port 25 zu binden, der in der Datei `/etc/service` den Alias-Namen `smtp` besitzt muss. Durch die Parametrierung `stream` und `nowait` wird **inetd** mitgeteilt, dass der eigentlich Service multithreaded arbeitet. Als eigentlicher Serverprozess wird nicht **qmail-smtpd,** sondern zunächst **tcp-env** aufgerufen und dieses mit dem Programm **qmail-smtpd** als Argument gestartet. **tcp-env** ermittelt die für **qmail-smtpd** notwendigen Environment-Variablen, die bereits im Abschnitt 4.5 vorgestellt wurden.
- Der Aufruf von **qmail-pop3d** vollzieht sich analog. Das Programm **qmail-popup** wird mit den effektiven Benutzerrechten von *root* unter Angabe der Parameter `qmail-popup YOURHOST checkpassword qmail-pop3d Mailbox` gestartet, wobei zusätzlich noch die Pfadangaben der Programme anzugeben sind. Hierbei liest **qmail-popup** für den angegebenen

Hostname `YOURHOST` einen POP3 Benutzernamen und mittels des Programms **checkpassword** das Benutzer-Passwort, bevor es das den eigentlichen Pop3-Server **qmail-pop3d** startet, der die Auslieferung der Nachricht in unserem Fall in eine benutzerspezifische `Mailbox` vornimmt.

Was hier so wortreich beschrieben ist, ist doch nicht mehr als eine Minimal-Konfiguration für den Qmail-Start. Diese Konfiguration ist weder besonders einfach und übersichtlich, noch vollständig, noch Portabel zwischen den verschiedenen Unix-Derivaten. Z.B. wurde Linux Red Hat in der Version 7.1 statt mit dem INETD mit dem XINETD in der Grundkonfiguration ausgestattet. Die Einbindung des Qmail SMTPD- und POP3-Servers vollzieht sich hierbei über die Konfigurationdatei `/etc/xinetd.conf` typischerweise wie folgt:

```
service smtp
{
    socket_type     = stream
    protocol        = tcp
    wait            = no
    user            = qmaild
    server          = /usr/sbin/tcp-env
    server_args     = -R /var/qmail/bin/qmail-smtpd
    log_type        = FILE /var/log/xinetd.log
    log_on_success  = HOST
    log_on_failure  = HOST RECORD
}
service pop3
{
    socket_type     = stream
    protocol        = tcp
    wait            = no
    user            = root
    server          = /var/qmail/bin/qmail-popup
    server_args     = YOURHOST /bin/checkpassword
/var/qmail/bin/qmail-pop3d Mailbox
    log_type        = FILE /var/log/xinetd.log
    log_on_success  = HOST
    log_on_failure  = HOST RECORD
}
```

*Listing 6-6: **qmail-smtpd** und **qmail-pop3d** unter Kontrolle des XINETD.*

Wir beachten, dass im Fall des "service smtp" das Programm **tcp-env** mit dem zusätzlichen Parameter `-R` aufgerufen wurde, der eine dem **tcpserver** identische Bedeutung besitzt, also den IDENT-Lookup unterdrückt (vgl. 4.7). Zudem kann die Log-Information beim XINETD viel feiner gesteuert werden. Im Beispiel oben werden diese in die Datei **/var/log/xinetd.log** eingestellt. Über den XINETD kann noch zusätzlich ein TCP-Wrapper, also der **tcpd** eingebunden werden. Zusätzlich können einige weitere Flags in der Datei `xinetd.conf` gesetzt werden, auf die aber hier nicht weiter eingegangen werden soll.

Es ist zu beachten, dass Änderungen der `inet.conf` durch ein HUP-Signal an den Prozess **inetd** wirksam werden, d.h. ein Neu-Lesen der Konfigurationsdatei erzwungen wird. Beim XINETD ist hierzu das HUP-Signal wirkungslos; an seine Stelle tritt das Signal SIGUSR1, bzw. bei BSD USR1.

6.4.2 Qmail, tcpserver und Run-Level-Skripte

Die Auftrennung der Qmail-Daemon-Prozesse in einerseits ein Startup-Skript und andererseits der Unterordnung unter den INETD/XINETD ist unter betrieblichen Aspekten wenig hilfreich. Beispielsweise kann die Situation eintreten, dass der Qmail SMTP-Server zeitweise unverzüglich abgestellt werden muss, um einer massiven Spam-Flut vorzubeugen.

Es ist zwar möglich, den **qmail-send** Daemon mit entsprechenden Signalen zu beinflussen, also zu stoppen und restarten, bei den von Superdaemons abhängigen Prozessen fehlt aber eine entsprechende Feinsteuerung. Wehe der Systemadministrator greift kann zum "letzten" Mittel den entsprechenden Dienst, z.B. **inetd** einfach zu "killen". Häufig ist dies mit einem Gang zur Rechnerkonsole verbunden (unter dem Spott der Kollegen).

Das Einrichten des Qmail SMTP- und POP3-Daemons unter Kontrolle von **tcpserver** bietet prinzipiell die Möglichkeit, ein vollständiges und komfortables Qmail-Start/Stopp-Skript bereitzustellen.

Qmail Start/Stop-Skripte gibt es wahrscheinlich so viele, wie Qmail Installationen auf den unterschiedlichen System. Im folgenden Beispiel wurde die IP-Adresse von Qmail per Befehl `hostname -i` ermittelt. Zwar ist der Befehl, nicht jedoch der Parameter `-i` auf jedem Unix-System vorhanden. Die aktuelle PID des gestarteten Qmail-Prozesses wird in einer PID-Datei unter `/var/run/` hinterlegt, was bei vielen Unix-Systemen einen Standard darstellt. Wird jedoch diese Datei versehentlich gelöscht, kann der Prozess nicht mehr beendet werden. Unter Linux exisitiert z.B. der Befehl **killproc**, mit der die PID eines Prozesses aufgrund seines Names ermittelt werden kann; dieser Befehl ist z.B. den BSD-Systemen unbekannt.

BSD

Unter FreeBSD empfiehlt sich die Installations der Pakete psmisc und pslist; wobei ersteres für das Tool qmHandle obligatorisch ist.

```
#!/bin/sh
## /etc/init.d/rc.qmail
#
PATH="/bin:/usr/bin:/usr/local/bin:/var/qmail/bin"
export PATH
HOSTNAME=‘hostname‘
#HOSTIP=‘hostname -i‘        # not supported on all systems
HOSTIP=abc.def.geh.ijk
QMAILDUID=‘id -u qmaild‘
QMAILDGID=‘id -g qmaild‘
QMAILLUID=‘id -u qmaill‘
#
case "$1" in
    start)
      echo "Initializing qmail:"
        qmail-start ./Maildir/ splogger qmail &
        echo $! > /var/run/qmail-send.pid
        echo "Done. Pid=‘cat /var/run/qmail-send.pid‘"
        echo "Starting qmail SMTP service:"
        tcpserver -R -u $QMAILDUID -g $QMAILDGID \
       -l $HOSTNAME $HOSTIP smtp qmail-smtpd &
        echo $! > /var/run/qmail-smtpd.pid
        echo "Done. Pid=‘cat /var/run/qmail-smtpd.pid‘"
        echo "Starting qmail POP3 service:"
        tcpserver -R 0 pop3 qmail-popup $HOSTNAME \
        checkpassword qmail-pop3d Maildir &
        echo $! > /var/run/qmailpop3d.pid
        echo "Done. Pid=‘cat /var/run/qmail-pop3d.pid‘"
        ;;
    stop)
        echo "Shutting down qmail SMTP service:"
        kill -TERM ‘cat /var/run/qmail-smtpd.pid‘
```

```
        echo "Done."
        echo "Shutting down qmail POP3 service:"
        kill -TERM ‘cat /var/run/qmail-pop3d.pid‘
        echo "Done."
        echo "Shutting down qmail send service:"
        kill -TERM ‘cat /var/run/qmail-send.pid‘
        echo "Done."
        ;;
    hup)
        echo "Reinitialising of qmail service:"
        kill -HUP ‘cat /var/run/qmail-send.pid‘
        echo "Done."
        ;;
    alarm)
        echo "Alarm processsing of E-Mails (qmail):"
        kill -ALRM ‘cat /var/run/qmail-send.pid‘
        echo "Done."
        ;;
    *)
        echo "Usage: $0 {start|stop|alarm|hup}"
        exit 1
esac
exit 0
```

Listing 6-7: System V Start/Stopp-Skript für Qmail.

Mit diesem Skript wird Qmail quasi als Ganzes gestartet bzw. gestoppt. Die Standard-Parameter (start/stop) des Skripts ermöglichen es, dieses als Grundlage von System V-Runlevel Initialisierung zu verwenden (vgl. 2.xxx). Sollen die Dienste **qmail-send**, **qmail-smtpd** und **qmail-pop3d** einzeln beeinflusst werden, ist das Skript in drei einzelne Skripte zu gliedern und entsprechend aufzurufen.

Ein andersgelagertes Problem ergibt sich, wenn wir die Fehlerausgabe (File-Descriptor 2) von **tcpserver** unter Angabe der Option `-v` mitschreiben wollen. Wir machen folgenden Versuch:

```
#!/bin/sh
## /etc/init.d/rc.qmail-smtpd
```

```
#
PATH="/bin:/usr/bin:/usr/local/bin:/var/qmail/bin"
export PATH
HOSTNAME=‘hostname‘
#HOSTIP=‘hostname -i‘        # not supported on all systems
HOSTIP=abc.def.geh.ijk
QMAILDUID=‘id -u qmaild‘
QMAILDGID=‘id -g qmaild‘
QMAILLUID=‘id -u qmaill‘
#
case "$1" in
    start)
        echo "Starting qmail SMTP service:"
        tcpserver -v -R -u $QMAILDUID -g $QMAILDGID \
        -l $HOSTNAME $HOSTIP smtp qmail-smtpd \
        2>&1 | splogger qmail &
        echo $! > /var/run/qmailsmtpd.pid
        echo "Done. Pid=‘cat /var/run/qmail-smtpd.pid‘"
        ;;
    stop)
        echo "Shutting down qmail SMTP service:"
        kill -TERM ‘cat /var/run/qmail-smtpd.pid‘
        echo "Done."
        ;;
     *)
        echo "Usage: $0 {start|stop}"
        exit 1
esac
exit 0
```

Listing 6-8: ***qmail-smtpd*** *Start/Stop-Skript unter Einbeziehung der Fehlerausgabe nach* ***splogger.***

Das Skript arbeitet beim Start fehlerfrei und wir verifizieren mit einer Testmail, dass die Ausgabe von **tcpserver** auch zusammen mit dem **qmail-send** Log per

splogger aufgezeichneit wird. Beim Aufruf des Stop-Parameters gibt es dann eine Überraschung: Der **tcpserver**/**qmail-smtpd** Prozess bleibt hängen!

```
qmaild 374  ....  0:00.01 tcpserver -v -R -u 82 -g 81 -l
qmailer.fehnet.de 192.168.192.2 smtp qmail-smtpd
```

Der Grund ist schnell ermittelt: Die Pid-Datei beinhaltet nicht die PID des gestarteten **tcpserver** sondern natürgemäss die PID des letzten Prozesses; **splogger** nämlich. Das saubere Beenden von "verketteten" Prozessen im Rahmen von Run-Level Skripten gestaltet sich als Herausforderung.

Im bekannten "Life with Qmail" gibt Dave Sill Hinweise, wie ein Run-Level-Skript unter Einbeziehung von Supervise aufgesetzt werden kann. Damit ist aber noch lange nicht geklärt, in welchen konkreten Run-Level das Skript per Link eingestellt wird; dies hängt an der konkreten Implementierung. Unter HP-UX ist es ausserdem gebräuchlich, in der `case`-Struktur die Anwendungshinweise vom konkreten Aufruf zu trennen.

Kurz gesagt, auch das hier vorgestellte Skript ist (a) Plattform-abhängig und (b) kaum sinnvoll erweiterbar. Punkt (a) ergibt sich aus den diskutierten Umständen, Punkt (b) ist eine Folge, dass die verschiedenen Qmail-Prozesse unterschiedliche Signale verarbeiten. Das Signal ALRM macht beispielsweise im Hinblick auf **qmail-smtpd** keinen Sinn.

6.4.3 Qmail unter Daemontools

Mit den Daemontools von Dan Bernstein steht zum ersten Mal eine für den Server-Betrieb und die Administration der Daemonprozesse Plattform-unabhängige Software bereit, die bereits ausführlich im fünften Kapitel vorgestellt wurde.

Nach Installation der Daemontools ist es erforderlich, einerseits das obligatorische `/service/` Verzeichnis anzulegen, andererseits die für Qmail notwendigen run-Skripte zu erzeugen. Der Abkehr von den traditionellen Betriebssystem-Mitteln zur Daemon-Verwaltung geht in der Regel einher mit dem Verzicht auf den **syslogd** im Hinblick auf das Einstellen der Qmail-Meldungen. Die Erfahrung vieler Unix "Power-User" zeigt, dass der **syslogd** bei starker Inanspruchnahme zu einer echten "Last" im System werden kann.

Die Alternative in Form des Daemontools Programms **multilog** stellt sich somit quasi unwillkürlich. Duch den Verzicht auf das Sticky-Bit im Daemontools `/service/.../log/` Verzeichnis und die zusätzliche Möglichkeit, die **multilog**-Logdateien durch ein ALRM-Signal zu schliessen, hat den Einsatz von **multilog** sicherlich stark vereinfacht und auch befördert.

Die Kombination Qmail/UCSPI/Daemontools ist ein probates Mittel, einen Hochverfügbarkeits-MTA aufzubauen und zu betreiben, dessen Verfügbarkeit

einerseits (intern) durch die Systemressourcen und andererseits (extern) durch das Netz und seine Komponenten (speziell die DNS-Server) begrenzt ist; sieht man einmal von Hardware-Ausfällen aus.

Obwohl teilweise eine Mixtur besteht, den Qmail-Sendeprozess per System-V Run-Level-Skript zu steuern und bei den Daemons **qmail-smtpd** und **qmail-pop3d** zum generischen Supervise-Start zu greifen, halte ich es mit der "reinen Lehre": **svc** wird — vergleichbar dem **inetd** — als zentraler Prozess gestartet; alles weitere ist nachgeordnet.

Wir gehen davon aus, dass die Daemontools entsprechend Vorgabe installiert sind, sodass es zunächst gilt, die Qmail-Skripte hierfür zu erstellen. Dan Bernstein empfiehlt, die Skripte nicht direkt unter ihrem Dienstnamen unter **/service/Dienstname/** zu erzeugen, sondern die Quelldateien beim Dienst selbst zu belassen und per symbolischen Link nach **/service/** zu überführen.

Dave Sill plädiert in seinem "Life with Qmail" und dem "qmail Handbook" hierfür ein Verzeichnis **/var/qmail/supervise/** anzulegen. Ich bin da etwas schreibfauler und verwende bei mir das Verzeichnis **/var/qmail/svc/**, wobei sich das Verzeichnis `/var/qmail/etc/` als weniger geeignet herausgestellt hat. Hierunter erzeuge ich mir die Dienst-spezifischen Verzeichnisse, also **qmail-send**, **qmail-smtpd** und **qmail-pop3d** und die zugehörigen `log`-Verzeichnisse:

```
# mkdir -p /var/qmail/svc/qmail-send/log
# mkdir -p /var/qmail/svc/qmail-smtpd/log
# mkdir -p /var/qmail/svc/qmail-pop3d/log
```

Wir beachten, dass in der Regel **qmail-smtpd** und **qmail-pop3d** keine Loginformationen schreiben. Unter Einsatz des **tcpserver** und der Option `-v` erhalten wir aber zumindest einen Überblick darüber, welcher Rechner wann eine Verbindung auf- und wieder abgebaut hat. Zudem existieren eine Reihe Patches, mit der sowohl **qmail-smtpd** als auch **qmail-pop3d** zusätzliche Informationen auf den File-Descriptor 2 leiten, die dann per **multilog** im entsprechenden Logfile landen.

Mit den uns schon bekannten Runlevel-Skripten können auch die Supervise-Skripte für Qmail weitgehend generisch aufgebaut werden. Zusätzlich nutzen wir jedoch einige Hilfsprogramme aus dem Baukasten der Daemontools, um eine etwas bessere Kontrolle der eigentlichen Daemonprozesse zu ermöglichen.

Zur kompletten Steuerung von Qmail mittels der Daemontools benötigen wird insgesamt sechs `run`-Skripte, jeweils eins für den eigentlichen Prozess und eins für das Logging. Es ist zu beachten, dass die `run`-Skripte ausführbar sein müssen; hier hilft ein `chmod +x run`. Wir gehen einfach einmal durch:

1. `/var/qmail/svc/qmail-send/run`:

```
#!/bin/sh
```

```
exec env - PATH="/var/qmail/bin:$PATH" \
qmail-start ./Maildir/
```

2. /var/qmail/svc/qmail-send/log/run:

```
#!/bin/sh
exec setuidgid qmaill multilog t /var/log/qmail-send
```

3. /var/qmail/svc/qmail-smtpd/run:

```
#!/bin/sh
QMAILDUID=‘id -u qmaild‘
QMAILDGID=‘id -g qmaild‘
HOSTNAME=‘hostname‘
MAXCONCURRENCY=‘cat /var/qmail/control/concurrencyincoming‘
exec softlimit -m 200000 \
        tcpserver -v -R -l $HOSTNAME -c $MAXCONCURRENCY \
        -u $QMAILDUID -g $QMAILDGID 0 smtp \
        /var/qmail/bin/qmail-smtpd 2>&1
```

4. /var/qmail/svc/qmail-smtpd/log/run:

```
#!/bin/sh
exec setuidgid qmaill multilog t /var/log/qmail-smtpd
```

5. /var/qmail/svc/qmail-pop3d/run:

```
#!/bin/sh
HOSTNAME=‘hostname‘
#HOSTIP=‘hostname -i‘   # not supported on all Unix systems
QMAILDUID=‘id -u qmaild‘
QMAILDGID=‘id -g qmaild‘
QMAILLUID=‘id -u qmaill‘
exec tcpserver -v -H -R -c100 0 pop3 \
       /var/qmail/bin/qmail-popup \
       $HOSTNAME /bin/checkpassword \
       /var/qmail/bin/qmail-pop3d Maildir 2>&1
```

6. /var/qmail/svc/qmail-pop3d/log/run:

```
#!/bin/sh
exec setuidgid qmaill multilog t /var/log/qmail-pop3d
```

Listing 6-9: Qmail `run`-Skripte für Supvervise.

Wir betrachten zunächst die `run`-Skripte 2, 4 und 6 für das Logging. Jeder Verbindungsauf- und Abbau wird durch die **tcpserver** Option `-v` aufgezeichnet. Hierbei sind folgende Aspekte von Bedeutung:

- Das Logging findet unter der gemeinsamen User-Kennung *qmaill* statt.
- Das Logging läuft in getrennten Verzeichnissen ab; wobei jeweils in diesem Verzeichnis eine Datei `current` beim Start des Logs erstellt wird, die von **multilog** administriert wird. Ein gemeinsames Logfile — wie unter Nutzung des **sploggers** — ist nicht möglich.
- Die Verzeichnisse müssen vor dem ersten Start mit dem Eigentümer *qmaill* erzeugt worden. Wir haben hier als "Mutterverzeichnis" `/var/log/` gewählt.

Die `run`-Skripte 1, 3 und 5 sind für die eigentlichen Daemonprozesse relevant. Dieses sind weitgehend generisch aufgesetzt, mit den folgenden Ausnahmen:

- Das `run`-Skript für **qmail-send** beinhaltet keinen Neuigkeiten; ausser, dass wir vorgeschrieben haben, lokale E-Mails in ein Maildir per Default einzustellen, was weiter unten genauer besprochen ist.
- Beim Aufsetzen des Skriptes für **qmail-smtpd** haben wir uns entschlossen, im Verzeichnis `/var/qmail/control/` die Datei `concurrencyincoming` anzulegen, mit der das Concurrency-Limit für **tcpserver** (Option `-c`) vorgegeben wird. Dies gestattet eine flexible Behandlung dieses Wertes für **tcpserver** in Übereinstimmung mit den möglichen Qmail Konfigurationsdateien `concurrencylocal` und `concurrencyremote`.

  Ferner haben wir die Speichernutzung des **qmail-smtpd** Prozesses per **softlimit** beschränkt, sodass das maximal genutzte Data- und Stack-Segment 200.000 Byte beträgt-
- Beim **qmail-pop3** Damon haben wir das Concurrencylimit im Aufruf fest begrenzt (`-c100`).

Wir beachten, dass in manchen Skripten von zusätzlichen Unix-Kommandos Gebrauch gemacht wird (**hostname**, **id**). Diese müssen natürliche zur Aufrufzeit des Skriptes im (Default-)Pfad liegen. Nachdem diese Verzeichnisse und die `run`-Skripte erzeugt und gestestet wurden (das einfach durch ihren Aufruf erfolgt) und die Dienste anschliessend wieder per **kill**-Befehl beendet wurde, sind die Skripte für Supervise verfügbar zu machen. Dies geschieht über symbolische Links:

```
# ln -s /var/qmail/svc/qmail-send /service/qmail-send
```

```
# ln -s /var/qmail/svc/qmail-smtpd /service/qmail-smtpd
# ln -s /var/qmail/svc/qmail-pop3d /service/qmail-pop3d
```


Achtung! Ist **svc** bereits aktiv wird anschliessend sofort der entsprechende Dienst gestartet. Haben wir bei der Konfiguration Fehler gemacht (z.B. weil wir bei einem Log-Verzeichnis versäumt haben, ein `chown qmaill ...` vorzunehmen) quittiert uns dies **svc** mit Einträgen in der Prozess-Ausgabe von **readproctitle**.

Die Qmail-Prozesse können nun sehr einfach gemeinsam gestartet bzw. gestoppt werden:

```
# svc -u /service/qmail*
# svc -d /service/qmail*
```

Hierbei blieben die Log-Prozessse per **multilog** aktiv. Will man diese gemeinsam beenden, genügt ein

```
# svc -d /service/qmail*/log
```

Wie wir sehen, vereinfacht uns Supervise die Administration von Unix-Daemons enorm. Wir bedenken allerdings, dass — je nachdem wie **supervise** im System gestartet wird — die per **svc** administrierten Dienste von den Run-Leveln entkoppelt sind. Soll hingegen ein Dienst in einem Run-Level nicht gestartet werden, aber **supervise** dennoch aktiv sein, muss der entsprechende **svc**-Aufruf in ein Run-Level-Skript einfliessen. Dies bedingt aber immer, dass **svc** zunächst den Dienst startet und ihn dann per Befehl wieder herunter fährt. Soll dies auf keinen Fall geschehen, genügt es — bevor **supervise** gestartet wird — im entsprechenden Service-Unterverzeichnis eine Datei `down` zu erzeugen (z.B. `touch down`).

6.4.4 Default Delivery

Die Angabe der sog. Default-Delivery beim Start von Qmail ist für den Qmail-Anfänger häufig mit vielen Fragezeichen versehen. Zudem wird in den ursprünglichen Qmail-Dokumenten von Dan Bernstein, mal von Mailbox und dann wieder von Maildir gesprochen. Ferner ist die Syntax von Mailbox und Maildir noch verschieden von den sonstigen Deklarationen in den `.qmail`-Dateien, sodass sich der Sinnzusammenhang nicht unmittelbar erschliesst.

Wir betrachten hierzu nochmals unser Beispiel beim Aufruf von **qmail-start** unter **svc**-Kontrolle:

```
#!/bin/sh
exec env - PATH="/var/qmail/bin:$PATH" \
qmail-start ./Maildir/
```

*Listing 6-10: Qmail `run`-Skrip zum Start von **qmail-start**.*

Kurz gesagt, besteht die Angabe der Default-Delivery darin, wie **qmail-local** zu Verfahren hat, wenn keine benutzerspezifische `.qmail` oder `.qmail-default` Datei vorliegt. Hierbei wird dann die Angabe genutzt, die Qmail bei seinem Start mitgeteilt bekommt. Wir berücksichtigen hier einige wesentliche Randbedingungen:

- Wird als Defaultdelivery eine Mailbox angegeben, wird in jedem Fall eine Nachricht auch dann erfolgreich zugestellt, wenn für den Empfänger weder eine **.qmail** Datei noch die Mailbox existierte. Letztere wird automatisch bei dier Zustellung erzeugt.
- Wird hingegen das Maildir-Format in der Default-Delivery vorgeschrieben, muss vor der ersten Auslieferung einer Nachricht an einen Benutzer in dessen Stamm-Verzeichniss das Maildir mit den korrekten Rechten bestehen, ansonsten wird dir Zustellung *deferred*.
- Der Benutzer kann in der `.qmail` Datei, die in seinem Home-Directory liegt, und für die er Verantwortung hat, jederzeit die Art und Weise bestimmen, mit der **qmail-local** an ihn zuzustellende Nachrichten verarbeiten und speziell auch wo und wie ablegen soll.
- Sollen die lokal abgelegten Nachrichten (E-Mails) per **qmail-pop3d** eingesehen werden können, ist zwingend das Maildir-Format vorgeschreiben.

6.5 Qmail-Architektur

6.5.1 Essenz von Qmail

Wer im Internet nach Qmail sucht, wird vielleicht schon über den Begriff "Qmail - The big picture" gestolpert sein. In den Qmail-Dokumenten liegt es Dan Bernstein fern, eine griffige und kompakte Definition der Arbeitsweise von Qmail vorzulegen. Wir haben bereits einige Aspekte von Qmail kennengelernt, die dieses Programm (und auch alle anderen Dan Bernstein Anwendung) von Standard-Unix-Programmen unterscheidet.

Doch: Was ist die Essenz von Qmail? Was unterscheidet Qmail so massgeblich von z.B. Sendmail. Da es keine "offizielle" Antwort gibt, muss diese zwangsläufig aus einer Art "Reverse-Engineering" der Sourcen erfolgen. Hierzu betrachten wir Abbildung 6-1, die in ähnlicher Form auch in der Datei INTERNALS der Qmail-Dokumentation zu finden ist.

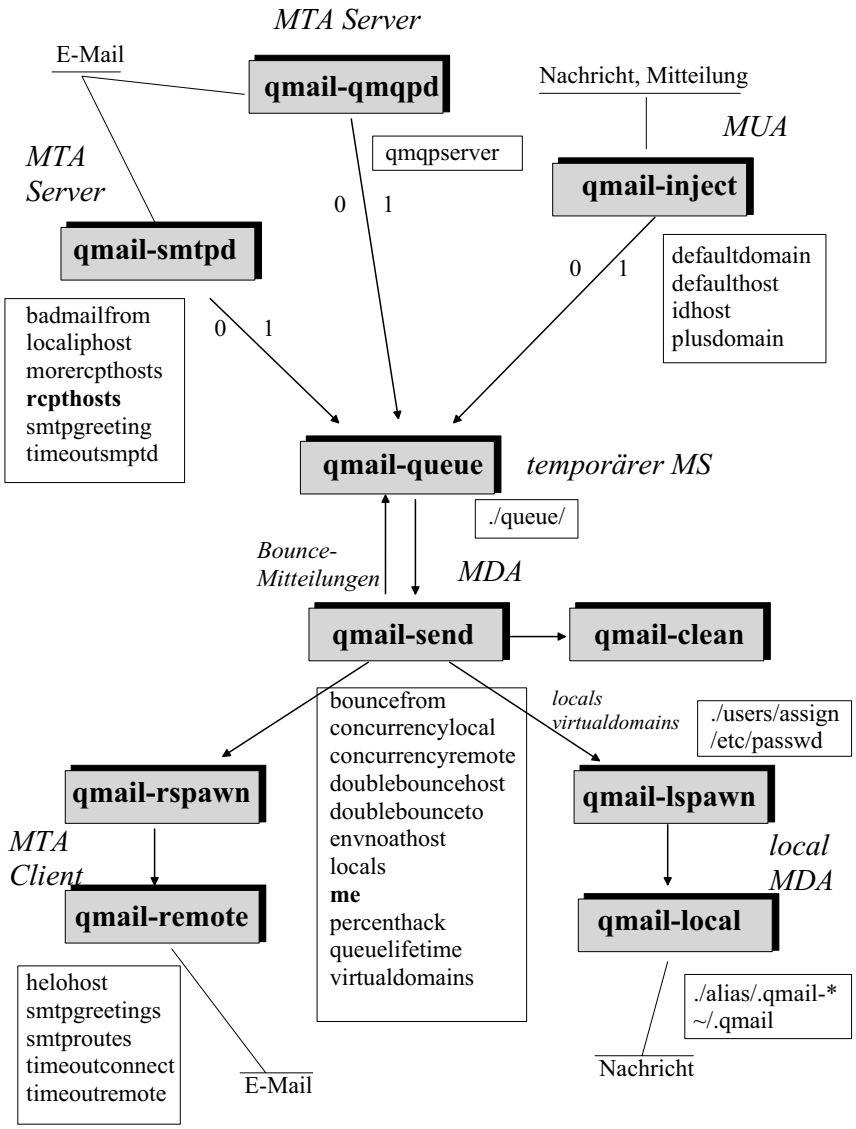

Abbildung 6-1: Arbeitsweise von Qmail und zugehörige Dateien; wir beachten den Unterschied zwischen E-Mail, Nachricht und Mitteilung. MTA = Mail Transfer Agent, MS = Message Store, MUA = Mail User Agent, MDA = Mail Delivery Agent..

Entsprechend dieser Abbildung geht der Datenfluss innerhalb von Qmail wie folgt vonstatten:

- *Einstellen von Nachrichten*:
  Jede Nachricht wird von **qmail-queue** in eine zentrale Nachrichten-Queue eingereiht. **qmail-queue** wird nur bei Bedarf aufgerufen, so z.B. von **qmail-inject**, falls Nachrichten oder Mitteilungen vom lokalen System aus zu versenden sind, von **qmail-smtpd** für via SMTP empfangene E-Mails oder von **qmail-send** für nicht-zustellbare (und zurück gekommene) E-Mails. **qmail-queue** liest die Mitteilung von File-Descriptor 0 und den Nachrichten-Header von FD 1.

- *Versenden von Nachrichten*:
  Zum Senden bzw. Ausliefern von Nachricht, stützt sich Qmail auf vier Daemon-Prozesse: **qmail-send**, **qmail-lspawn**, **qmail-rspawn** sowie **qmail-clean**. **qmail-send** kommt hierbei die entscheidende Aufgabe zu, die Auslieferung der Nachrichten aus der Queue zu triggern (scheduling) und and die untergeordneten Prozesse **qmail-local** oder **qmail-remote** zu übermitteln und anschliessend die Nachrichten mittels **qmail-clean** aus der Queue zu entfernen.

An zentraler Stelle steht bei Qmail daher die Nachrichten-Queue. Diese kann entsprechend den Vorgaben von Dan Bernstein, als sicherer, zuverlässiger und (weitgehend) schneller Datenspeicher für Nachrichten aufgefasst werden. Qmail ist daher also in erster Line quasi eine Datenbank-Anwendung für kurz- und mittelfristige Daten. Die Fristigkeit der Daten richtet sich nach folgenden Parametern:

- Wie schnell kann die Nachricht lokal ausgeliefert werden?
- Wie schnell kann die E-Mail zum Empfänger (dem verantwortliche Ziel-MTA) zugestellt werden?
- Wie gross ist die Verweildauer für Nicht-Zustellbare E-Mails?

Sieht man einmal von Problemen mit dem lokalen Empfang ab, geschieht die lokale Auslieferung lokaler Nachrichten in Sekundenbruchteilen. Die Versendung von E-Mails an entfernte Empfänger ist natürlich komplexer. Neben den Zeiten für die Feststellung des Empfänger-MTAs kommen hier Netz-Latenzzeiten und die Eigenschaften und Auslastung des entfernten SMTP-Servers hinzu; selbstverständlich auch seine "Bereitschaft" unsere E-Mail überhaupt anzunehmen. Die Verweildauer in der Queue kann jedenfalls vom Mail-Administrator vorgegeben werden; der Defaultwert beträgt hier 7 Tage, was durchaus vernünftig ist.

6.5.2 Qmail-Programme

Um Aufschluss zu erhalten, was bei der Installation von Qmail (und den anderen oben beschriebenen Werkzeugen) im Verzeichnis `/var/qmail/bin/`

"gelandet" ist, führen wir den Befehl **file** aus:

```
file /var/qmail/bin/*
```

Neben den kompilierten und gelinkten C-Programmen, hat uns Dan Bernstein einige Skripte mitgliefert: **datemail**, **elq**, **mailsubj**, **qail**, **qmail-qstat**. Sofern wir **file** als normaler User und nicht als *root* aufgerufen haben, verweigert uns **file** die Identifikation der Programme **qmail-clean**, **qmail-getpw**, **qmail-local**, **qmail-lspawn**, **qmail-newmrh**, **qmail-popup**, **qmail-pw2u**, **qmail-queue**, **qmail-remote**, **qmail-send**, **qmail-start** und **splogger**. Diese Pogramme besitzen den Owner *root* sowie den Dateimodus 700.

Einen thematischen Überblick über die installierten Qmail-Programme bieten die Tabellen 6-1 bis 6-9:

- Daemon-Programme

Programm	**Parameter**	**Beschreibung**
qmail-start	`[ defaultdelivery [ logger arg ... ] ]`	startet die untergeordneten Daemon-Prozesse **qmail-send**, **qmail-lspawn**, **qmail-rspawn** und **qmail-clean**
qmail-send		überführt Nachrichten aus der Queue zu lokalen oder entfernten Empfängern; zusätzliche Signale: TERM — ausstehende Nachrichten werden noch ausgeliefert, anschliessend Prozess "sauber" beendet, ALRM — alle Nachrichten in der Queue werden unverzüglich ausgeliefert, HUP — die Konfigurationsdateien `locals` und `virtualdomains` werden erneut gelesen und ausgewertet
qmail-smptd		SMTP Daemon-Prozess
qmail-queue		verwaltet die Qmail-Queue
qmail-clean		löscht aus der Queue übermittelte oder nicht übermittelbare Nachrichten
qmail-lspawn		verwaltet die Auslieferung

Programm	Parameter	Beschreibung
		(mittels **qmail-local**) lokal zuzustellender Nachrichten
qmail-rspawn		verwaltet die Auslieferung (mittels **qmail-remote**) von Nachrichten an entfernte Empfänger
splogger	`[ tag [ fac ] ]`	Übertragung von Log-Mitteilungen an den **syslogd**
tcp-env	`[ -rR ] [ -ttimeout ] program [ arg ... ]`	Emittlung der TCP-Umgebungsvariablen

Tabelle 6-1: Qmail Kommandos, die als Daemon-Prozesse laufen bzw. gestartet werden.

- QMQ-Programme

Programm	Parameter	Beschreibung
qmail-qmtpd		QMTP Daemon-Prozess
qmail-qmqpc		Überstellung von Nachrichten nicht in die Queue, sondern zu einem entfernten QMQP-Server
qmail-qmqpd		QMQP-Server-Prozess

Kommandos und Daemon-Prozesse des Quick Mail Protocols (QMP)

- POP3-Programme

Programm	Parameter	Beschreibung
qmail-pop3d	`maildirname`	POP3-Daemon-Prozess
qmail-popup	`hostname subprogram`	lesen eines POP3 Benutzernames und Kennworts
checkpassword	`subprogram [ args ... ]`	Überprüfung eines Kennworts mittels der System-Kommandos **getpwnam** und **crypt**

Tabelle 6-2: Qmail's POP3-Schnittstellen-Programme.

- Mail-Delivery Agents

Programm	Parameter	Beschreibung
qmail-local	`[ -nN ] user homedir local dash ext domain sender defaultdelivery`	Ausliefern (und optionales Weiterverarbeiten) lokaler Nachrichten
qmail-remote	`host sender recip [ recip ... ]`	SMTP-Client

Tabelle 6-3: Die Zustellprogramme von Qmail.

- Mail User-Agents

Programm	Parameter	Beschreibung
qmail-inject	`[ -nNaAhH ] [ -fsender ] [ recip ... ]`	list eine Mitteilung, erzeugt einen Nachrichtenkopf und platziert die Nachricht in die Queue
mailsubj	`subject recip ...`	erzeugt eine Nachricht mit vorgebenem "Subject:" und Empfänger und stellt diese in die Queue ein
sendmail	`[ -t ] [ -fsender ] [ -Fname ] [ -bp ] [ -bs ] [ arg ... ]`	Sendmail-Wrapper für **qmail-inject**
datemail	`[ -t ] [ -fsender ] [ -Fname ] [ -bp ] [ -bs ] [ arg ... ]`	Skript als Sendmail-Wrapper, wobei zusätzlich ein "Date"-Feld im E-Mail-Header eingefügt wird

Tabelle 6-4: Qmail Benutzerkommandos, um Nachrichten in die Queue zu platzieren.

- Benutzer-Programme für .qmail-Dateien

Programm	Parameter	Beschreibung
bouncesaying	`error [ program [ arg ... ] ]`	erzeugt (optional) eine Bounce-Nachricht an den Absender
condredirect	`newaddress program [ arg ... ]`	optionales Weiterleiten einer Nachricht an andere Empfänger

Programm	Parameter	Beschreibung
execept	`program [ arg ... ]`	Umkehrung des Exit-Codes eines Programms
elq		Überführen eines Mailverzeichnis ins mbox-Format und Aufruf von des MUA **elm**
forward	`address ...`	Forwarding von Nachrichten an weitere Empfänger
preline	`command`	Ergänzung einer Nachricht um die UUCP-Zeilen From_, Return-Path und Delivered-To
qail		Überführen eines Mailverzeichnis ins mbox-Format und Aufruf von des MUA **Mail**
qbiff		Nachrichten-Notifikation für den Empfänger (eingeloggt in der Benutzer-Shell)
qreceipt	`youraddress`	Nachrichten-Notifikation für den Absender

Tabelle 6-5: Satz von Hilfsprogramme, die optional innerhalb einer `.qmail`*-Datei von* **qmail-local** *ausgeführt werden.*

- Maildir/Mailbox-Programme

Programm	Parameter	Beschreibung
maildirmake	`[ -S -f folder -s mode --add name=/shared/maildir/path, --del ] maildir`	Erzeugen eines Mailverzeichnisses
maildirwatch		Ausgabe der eingelaufenen neuen Nachrichten eines

Programm	Parameter	Beschreibung
		Mailverzeichnisses auf dem Bildschirm
maildir2mbox		Überführung eines Mailverzeichnisses in das Mbox-Format

Tabelle 6-6: Programme, zur Erzeugung und Verwaltung des von Qmail bevorzugten Maildir-Formats.

- Qmail-Administrations-Programme

Programm	Parameter	Beschreibung
qmail-newmrh		Erzeugen der morercpthosts cdb aus einer Datei
qmail-qstat		Anzeige der Queue-Stati
qmail-qread		Anzeige der Nachrichten in der ausgehenden Queue
qmail-tcpto		Anzeige der SMTP-Verbindungen im TCP-Timeout
qmail-tcpok		Löschen der TCP-Timeout-Verbindungenstabelle
qmail-showctl		Ausgabe der Inhalte der Qmail Kontrolldateien

Tabelle 6-7: Die spärlichen Qmail Verwaltungswerkzeuge.

- Benutzer-Administrations-Programme

Programm	Parameter	Beschreibung
qmail-pw2u	`[ -/ohHuUC ] [ -cchar ]`	Erzeugen einer Qmail-User `assign` Datei aus der Unix `/etc/passwd`
qmail-newu		Aufbau der Qmail-User cdb aus einer Qmail-User Datei

Programme zur Erzeugung der Qmail-User

- Erzeugen und Verwalten einer Forwarding-Datenbank

Programm	Parameter	Beschreibung
setforward	`cdb tmp`	Erzeugen einer Forwarding cdb
fastforward	`[ -nNpPdD ] cdb`	Weiterleiten von Nachrichten gemäss der Forwarding cdb
printforward		Ausgabe der Forwarding cdb
newaliases		Erzeugen einer Sendmail-kompatiblen Alias-Benutzer-Datenbank

Tabelle 6-8: Hilfsprogramme zum gezielten Weiterleiten von Nachrichten.

- Mailing-Listen-Programme

Programm	Parameter	Beschreibung
setmaillist	`bin tmp`	Erzeugen einer Mailing-Listen-Binärdatei
printmaillist		Ausgabe einer Mailing-Listen-Binärdatei
newinclude	`list`	Erzeugen einer Binärdatei für Mailinglinsten aus einer `:include` Datei

Tabelle 6-9: Rudimentäre Qmail Mailing-Listen-Programme.

6.5.3 Der Daemon-Prozess: qmail-send

Abbildung 6-1 zeigt, dass **qmail-send** der zentrale Qmail-Daemonprozess ist. Er wird mittels **qmail-start** unter Angabe der Default-Delivery aufgerufen; verbleibt aber — anders als typische Daemonprozesse — immer im Vordergrund.

Aufgaben von **qmail-send** sind:

- Verwaltung der Qmail-Queue, insbesondere das "Aufräumen" der Queue mittels **qmail-clean**.
- Feststellen, ob eine Nachricht für einen lokalen (einschliesslich evtl. Virtual Domains) oder für einen entfernten Empfänger bestimmt ist.

- Scheduling der auszuliefernden Nachrichten.
- Starten der untergeordneten Prozesse **qmail-lspawn** und **qmail-rspawn**.
- Verschicken von Bounce-Mitteilungen,.
- Erzeugen eines Aktivitätslog.

Innerhalb der Qmail-Programmstruktur ist sicherlich **qmail-send** das komplexeste Programm; was Dan Bernstein auch im Source-Code an vielen Stellen mit der ironischen Bermerkung "this file ist too long" kommentiert hat. Im Grunde genommen, hat Dan Bernstein für **qmail-send** einen falschen Namen gewählt; es gibt tatsächlich nur *eine* Situation, in der **qmail-send** auch wirklich "sendet": Bei Bounce-Mitteilungen.

6.5.3.1 Eigenschaften von qmail-send

Beim Initialisieren von **qmail-send** werden zunächst die in Tabelle 6-10 nochmals aufgeführten Konfigurationsdateien gelesen. Durch Auswertung der Dateien `locals` sowie `virtualdomains` weiss **qmail-send** nun, welche Nachrichten lokal zuzustellen sind; d.h. für E-Mails mit diesen Host- bzw. Domainangaben wird **qmail-lspawn** gestartet; andernfalls **qmail-rspawn**. Sind diese Konfigurationsdateien nicht vorhanden, wird per Dault der Hostname in der Datei `me` herangezogen.

Konfigurationsdatei	**Bedeutung**
`concurrencylocal`	Anzahl der maximal simultan gestarteten **qmail-lspawn** Prozesse. Default: 10, Minmum: 0 (keine lokale Zustellung), Maximum 120.
`concurrencyremote`	Anzahl der maximal simultan gestarteten **qmail-rspawn** Prozesse. Default: 20, Minimum: 0 (keine lokale Zustellung), Maximum: 120.
`bouncefrom`	Absender-Name im E-Mail Header bei Bounce-Mitteilungen. Default: MAILER-DAEMON.
`bouncehost`**)	Absender-Host in Bounce-Mitteilungen im E-Mail-Header. Default: `me`.
`doublebouncehost`**)	Hostname, an die Double-Bounce Mitteilungen verschickt werden. Default: `me`.
`doublebounceto`	Empfängername für Double-Bounce Mitteilungen. Default: *Postmaster*.
`locals`*)	Liste der lokalen Domains (Default **me**).
`virtualdomains`*)	Liste der Virtuellen Domains und auf welchen lokalen lokalen User sie abgebildet werden.

Konfigurationsdatei	Bedeutung
`envnoathost`	Angemommener und ergänzter Domain-Name für Adressen ohne Host- bzw. Domainkennung. Default: `me`.
`percenthack`	E-Mail-Adressen im UUCP-Style (`user%fqdn@domain`) für Domains, die hier aufgelistet sind, werden in das SMTP-Format `user@fqdn` überführt.
`queuelifetime`	Lebensdauer (in Sekunden) der Nachrichten in der Queue. Default: 604800.

*Tabelle 6-10: Konfigurationsdateien für **qmail-send** die beim Start eingelesen werden. Die mit einem *) versehenen Datei werden zusätzlich nach dem Empfangs eines HUP-Signals neu ausgewertet; besitzen die mit **) markiert Dateien keinen Inhalt, nimmt **qmail-send** den Dateinamen an.*

qmail-send kommuniziert mit den Programmen **qmail-clean**, **qmail-lspawn** und **qmail-rspawn** über definierte File-Descriptoren, wie dies aus Abbildung 6-2 entnommen werden kann. Das Schreiben des Aktivitätslogs geht über den File-Descriptor 0, wobei in der Regel die Ausgabe in das Programm **multilog** erfolgt, ober aber — über den Umweg **splogger** — im Syslog erscheint.

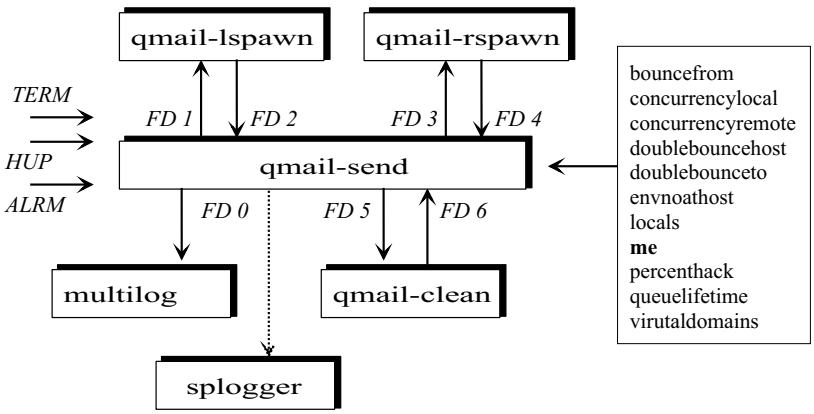

*Abbildung 6-2: Zusammenspiel von **qmail-send** mit seinen nachgeordneten Programmen unter Angabe der File-Descriptoren. Über den FD 0 wird in das Log-Programm beschrieben; entweder **multilog** oder (mit gestricheltem Pfeil dargestellt) **splogger**.*

Die Steuerung von **qmail-send** erfolgt über drei Signale:

- TERM — **qmail-send** wird beendet; zuvor versucht es aber noch die in der Queue ausstehenden Nachrichten "loszuwerden".
- HUP — **qmail-send** wird veranlasst, die Konfigurationsdateien `locals` und `virtualdomains` neu zu lesen.
- ALRM — **qmail-send** prozessiert alle in der Queue stehenden Nachrichten unkonditional erneut.

Es ist klar, dass auch das Standard-Signal KILL zum Einsatz kommen kann; **qmail-send** wird dann aber beendet, ohne die Queue in einem definierten Zustand zu hinterlassen. Dies ist leider manchmal einer Notwendigkeit, da das Ausführen des TERM-Signals zu lange dauern kann.

6.5.3.2 Nachrichten-Identifier

Unter Qmail wird eine Nachricht durch mehrere Identifier geführt, die im **qmail-send** Logfile zu finden sind:

- Message-Number (`msg`): Inode der Nachricht in der Queue; wir beachten, dass diese Message-Number nicht identisch ist mit "Message-ID" im E-Mail-Header.
- Queue-Process-Id (`qp`): Beim Einstellen einer Nachricht in die Qmail-Queue wird dieser eine Queue-Process-Id zugewiesen. Dies gilt für alle Qmail-Programme, die die Queue bedienen, d.h. neben **qmail-send** auch z.B. **qmail-inject** und **qmail-smtpd**.
- Delivery-Id (`delivery`): Zustellungsnummer für Nachrichten; diese werden einfach seit dem Start von **qmail-send** hochgezählt und für jeden Zustellungsversuch neu vergeben.
- Empfangs-Id (`uid`): Effektive Benutzerkennung (UID) des Users zu dem die Nachricht in erster Instanz zugestellt wird, z.B. der Qmail *alias* Benutzer.

Hierbei ist die Message-Number der "long term" Identifier einer Nachricht in der Queue. Bleibt eine Nachricht in der Qmail-Queue auch nach einem Neustart von **qmail-send**, verändern sich alle Kenngrössen ausser der Message-Number; sieht man einmal von der Empfangs-Id ab

Hingegen wird unter "Message-ID" ein spezielles E-Mail Header-Feld verstanden. Aufgabe dieses Feldes ist, eine E-Mail "über Raum und Zeit" eindeutig zu kennzeichnen und darüber hinaus den sendenden MTA zu benennen. **qmail-send** und **qmail-inject** bauen den Inhalt des Feldes "Message-ID:" wie folgt auf:

```
Message-ID:
<19950926044653.12345.qmail@silverton.berkeley.edu>
```

Hierbei steht der Teil `19950926044653` für das aktuelle Datum (UTC) entsprechend der Konvention `YYYYMMDDhhmmss` (Y=Year, M=Month, D=Day, h=hour, m=minute, s=second). Die Angabe der Zahl nach dem Punkt ist die PID von **qmail-send** bzw. **qmail-inject** beim Versenden. Anschliessend identifiziert sich der Qmail-MTA in der E-Mail durch den String "`qmail`". Der Teil der Message-ID hinter dem "@" ist die Information, die in der Konfigurationsdatei `idhost` hinterlegt, bzw. aus `me` entnommen wird.

Wir bedenken, dass die Message-ID vom SMTP-Sender generiert wird. Dies ist in der Regel der Mail-Client des Anwenders; im Falle von Qmail **qmail-inject** und nur im Falle von Bounces **qmail-send**.

6.5.3.3 Bounces

Kann eine E-Mail an einen (lokalen oder in der Regel entfernten) Empfänger dauerhaft nicht ausgeliefert werden, generiert der verantwortliche MTA eine sog. (Single-)Bounce-Mitteilung an den Absender der unzustellbaren E-Mail. Laut RFC 2821 bleibt hierbei im SMTP-Envelope der Absender leer; also `MAIL FROM: <>`.

Verantwortlich für die Erstellung von Bounce-Mitteilungen ist **qmail-send**. Dies geschieht typischerweise dann, wenn die E-Mail per **qmail-remote** nicht zustelllt werden kann. Qmail informiert den Absender über die fehlerhafte Zustellung unter Angabe des (vorgesehenen) Empfängers, einen Hinweis auf die mögliche Ursache sowie einer Kopie der usprünglichen Nachricht als Teil der Bounce-Nachricht. Dieses Format nennt Dan Bernstein QSBMF*: qmail-send Bounce Message Format* (`http://cr.yp.to/proto/qsbmf.txt`). Die Bounce-Nachricht hat dann immer folgenden Aufbau:

```
Date: 17 Mar 1996 03:54:40 -0000
From: MAILER-DAEMON@silverton.berkeley.edu
To: djb@silverton.berkeley.edu
Subject: failure notice


Hi. This is the qmail-send program at
silverton.berkeley.edu.
I'm afraid I wasn't able to deliver your message to
the
following addresses. This is a permanent error; I've
given up.
Sorry it didn't work out.


<god@heaven.af.mil>:
```

```
      Sorry, I couldn't find any host by that name.

      --- Below this line is a copy of the message.

      Return-Path: <djb@silverton.berkeley.edu>
      Received: (qmail 317 invoked by uid 7); 17 Mar 1996
03:54:38 -0000
      Date: 17 Mar 1996 03:54:38 -0000
      Message-ID:
<19960317035438.316.qmail@silverton.berkeley.edu>
      From: djb@silverton.berkeley.edu (D. J. Bernstein)
      To: god@heaven.af.mil
      Subject: are you there?

      Just checking.
```

*Listing 6-11: Aufbau einer **qmail-send** Single-Bonce Mitteilung.*

Manchmal ist es nicht mehr möglich, die Bounce-Mitteilung an den Absender zuzustellen, weil dieser z.B. falsch angegeben wurde. Kommt diese vom zugeordneten MTA zurück, dann generiert **qmail-send** eine Double-Bounce-Mitteilung, die als Failure-Notiz an den Postmaster verschickt wird:

```
From: MAILER-DAEMON@orion.fehnet.de
To: postmaster@orion.fehnet.de
Subject: failure notice

Hi. This is the qmail-send program at orion.fehnet.de.
I tried to deliver a bounce message to this address, but the
bounce bounced!
```

*Listing 6-12: Aufbau einer **qmail-send** Double-Bounce Mitteilung.*

Obwohl Bounce-Mitteilungen insgesamt nicht sehr häufig sind, besitzen sie dennoch zur Analyse etwaiger Zustellungsprobleme einige Bedeutung. Bei der Konstruktion des QSBMF ging Dan Bernstein von folgenden Überlegungen aus:

- Die von RFC 1892/1894 vorgesehenen Delivery Status Notifications (DSN) werden nicht eingesetzt; es soll keine Mixtur von Failure und Deferral Informationen stattfinden.

- Die Bounce-Mitteilung beginnt immer mit "`Hi. This ist the qmail-send program ...`" und benennt den (nicht erreichbaren) SMTP-Empfänger in spitzen Klammern "`<god@heaven.af.mil>:`" mit der finalen Angabe des Grundes: "`Sorry, I ...`".
- Die ursprüngliche E-Mail wird in die Bounce-Mitteilung nach der Kennung "`--- Below this line is a copy for the message.`" kopiert. Hierdurch ist der komplette E-Mail Header vollständig erhalten, die MIME-Struktur der eigentlichen Nachricht geht jedoch verloren, bzw. ist von den normalen User-Agents nicht mehr erkennbar.
- Der Absender der Bounce E-Mail kann durch die Konfigurationsdatei `/var/qmail/control/bouncefrom` angepasst werden; ansonsten lautet der (bei SMTP typische) Name MAILER-DAEMON.
- Soll in der Bounce-E-Mail nicht der in der Konfigurationsdatei `me` angegebene Hostname verwendet werden (wie z.B. im Beispiel oben "orion.fehnet.de"), sondern — wie bei der Angabe `idhost` für die Message-ID — ein generischer Absender-Hostname, kann dies über `bouncefrom` realisiert werden.
- Sollen Double-Bounce-Nachricht nicht vom lokalen MTA selbst (identifiziert über die Datei `me`) sondern von einem anderen MTA verarbeitet werden, können diese von **qmail-send** durch Angabe von Empfänger und Hostname weitergeleitet werden: Der Hostname des Empfängersystems wird über die Datei `doublebouncehost` angepasst; und statt zum *Postmaster* lässt sich die Double-Bounce E-Mail an den in `doublebounceto` hinterlegten Namen versenden.
- Als Absender der Double-Bounce Nachricht wird `MAIL FROM: <#@[]>` eingesetzt. Diese laut RFC 821 illegale Adressangabe nutzt **qmail-send**, um sicherzustellen, dass nicht auch diese Nachricht weiter kursiert.

Im Prinzip ist es natürlich möglich, dass auch eine Double-Bounce-E-Mail ihr Ziel nicht erreicht. In diesem Fall gibt **qmail-send** auf: Triple-Bounce-Nachrichten werden nicht generiert.

An dieser Stelle sei vermerkt, dass andere MTA, wie z.B. Microsoft's Exchange, bei Bounce-Mitteilungen die ursprüngliche Nachricht als MIME-Attachment im Format RFC822 multipart/mixed einschliesst. Dies hat den Vorteil, dass hierdurch die MUAs die gesendete Nachricht quasi im Original einsehen können. Der Nachteil besteht jedoch darin, dass sich hierduch die E-Mail — je nach Umsetzung des MIME-Teils der Nachricht — immer mehr aufbläht.

MTA Bounce-Death

Mir sind Fälle bekannt, wo bzw. ein Anwender von Outlook über Exchange zu seinem T-Online Account zu Hause unbedingt eine 8 Mbyte grosse Software verschicken wollte. E-Mails von dieser Grösse werden jedoch von T-Online

abgelehnt und als Bounce-Mitteilung an den Absender "postwendend" zurück geschickt. Um nichts falsch zumachen wird die Original-Nachricht als Base64 MIME-Attachment eingefügt, was diese wieder um 20% vergrössert. Die Firma hatte ebenfalls eine Regel aufgestellt, dass von ausserhalb einlaufende E-Mails nicht grösser als 5 Mbyte sein durften. So schauckelten sich die Bounce E-Mails immer mehr auf, bis das Exchange-System der Firma zusammenbrach, weil es die vielen (grossen) Bounce E-Mails nicht mehr los wurde: Death of a MTA.

An dieser Stelle möchte ich aber darauf hinweisen, dass es von Federik P. Lindberg einen Patch für **qmail-send** gibt, durch den innerhalb der Bounce E-Mail die ursprüngliche Nachricht als RFC822-MIME-Attachments generiert wird (`qmail-mime_tar.gz`).

6.5.4 Local Injection

Unter *local injection* soll die Einreihung einer lokal erstellten Nachricht in die Qmail-Queue verstanden werden. Wie aus Abbildung 6-1 hervorgeheht, nimmt diese Aufgabe das Programm **qmail-inject** wahr:

- **qmail-inject** — Das Benutzerinterface um Einstellen von Nachrichten. **qmail-inject** liest die Mitteilung von FD 0 und
  - erzeugt mittels der übergebenen Parameter, der gesetzten Envionment-Variablen und aufgrund eines evtl. bereits vorhandenen Headers den aktuellen Nachrichten-Header und
  - injeziert die somit nun vollständige und RFC822-konforme Nachricht in die Qmail-Queue.

qmail-inject ist ein "poor man's" Benutzer-Interface, da es nicht in der Lage ist, komplexe Nachrichten, z.B. mit einer MIME-Struktur zu erzeugen. Leider hat uns Dan Bernstein keinen kompletten E-Mail Client spendiert, vergleichbar dem **mail** Programm (in der Qmail Dokumentation als `binmail` bezeichnet; da es typischerweise unter `/usr/bin/mail` zu finden ist).

- Hilfsweise existiert jedoch **mailsubj**, ein Wrapper-Skript um **qmail-inject**, mit dem ein einfacher E-Mail-Versand — lediglich unter Angabe des "Subject:" und der Empfängeradresse — möglich ist. Um dessen Einsatz zu vereinfachen, sollte es als Symlink in `/usr/local/bin/` eingestellt werden. Folgendes Beispiel demonstriert seinen Nutzen:

```
# ln -s /var/qmail/bin/mailsubj /usr/local/bin/mailsubj
# mailsubj "passwd Datei" root@localhost < /etc/passwd
```

Wir beachten, dass **mailsubj** keine Möglichkeit des interaktiven Aufbaus einer Nachricht beinhaltet. Die Mitteilung wird vom Standard-Input gelesen; um den Header einschliesslich des angegebenen "Subjects:" ergänzt und an **qmail-queue**

übertragen. Mehr Komfort hat uns Dan Bernstein nicht zugestanden.

6.5.4.1 Aufruf von qmail-inject

Der Aufruf von **qmail-inject** geschieht folgendermassen:

```
qmail-inject [ -nNaAhH ] [ -fsender ] [ recip ...  ]
```

Die Beim Erzeugen des Nachrichten-Header greift **qmail-inject** auf unterschiedliche Informationen zurück:

- *Übertragungsspezifische Informationen* — vor allem der Empfänger der Nachricht — werden über Aufruf-Parameter mitgeteilt,
- *User-spezifischen Daten* werden aus der laufenden laufenden Shell ermittelt, was über lokale Environment-Variablen geschieht,
- *Host-spezifische Daten*, deren Default-Werte aus den Qmail Kontrolldateien ermittelt werden.

Ist die Nachricht erfolgreich an **qmail-queue** ausgeliefert worden, bedankt sich **qmail-inject** mit einem Exit-Code 0. Ein Exit-Code 100 weist darauf hin, dass die Nachricht syntaktisch nicht in Ordnung ist, Exit-Code 111 verweist darauf, dass **qmail-queue** (temporär, z.B aufgrund des Concurrency-Limits) nicht in der Lage ist, die Nachricht entgegen zu nehmen.

Wir betrachten zunächst die möglichen **qmail-inject** Parameter:

Optionen von qmail-inject	**Bedeutung**
`-a`	Address evaluation only E-Mail wird an alle Adressen gesendet, die als Argumente aber nicht über den Nachrichten-Header mitgeteilt werden.
`-h`	Header evaluation only E-Mail wird an alle Empfänger im Nachrichten-Header weitergeleitet (To, Cc, Bcc, Appemtly-To sowie Resent-To, Resent-Cc, Resent-Bcc); die Angabe der Empfänger auf der Kommandozeile wird ignoriert
`-A`	Address and Header evaluation Sendet die Nachricht an alle als Argumente eingespeiste Empfänger; sind keine Argumente angegeben, wird der Nachrichten-Header ausgewertet (Default).
`-H`	Header and Address evaluation Die Nachricht wird an alle im Header angegeben Empfänger als auch an alle per Argument mitgeteilten Empfänger weiter geleitet.
`-fsender`	Die Angabe von `sender` wird **qmail-queue** als Absenderangabe im

Optionen von qmail-inject	**Bedeutung**
	E-Mail Umschlag vorgeschrieben. Der "Return-Path "sowie alle weiteren Environment Variablen werden ignoriert.
`-N`	Die resultierende E-Mail wird nach **qmail-qeue** eingeleitet (Default).
`-n`	Die E-Mail wird gedruckt, statt in die **qmail-queue** eingeleitet

*Tabelle 6-11: Parameter beim Aufruf von **qmail-inject.***

6.5.4.2 Parameter und Environment-Variablen unter qmail-inject

Der Anwender hat aber immer durch das Setzen lokaler Environment-Variablen die Möglichkeit, auf die Erstellung der Nachricht Einfluss zu nehmen, was folgenden Tabelle demonstriert:

Environment-Variable(n)	**Bedeutung**
`$QMAILUSER`, `$MAILUSER`, `$USER` oder `$LOGNAME`	Der Name im "From:" Header-Feld
`$QMAILHOST` oder `$MAILHOST`	Der Hostname im "From:" Header-Feld, normalerweise der Eintrag in der Konfigurationsdatei `defaulthost`
`$QMAILNAME`, `$MAILNAME` oder `$NAME`	Der persönliche Name (Feld "Sender:" im Nachrichten Header)
`$QMAILSUSER` sowie `$QMAILSHOST`	Die Absenderangabe beim SMTP "Mail From:"
`$QMAILMFTFILE`	Liste von E-Mail-Adressen (eine pro Zeile) zum Versand

*Tabelle 6-12: Benutzerspezifische Einvironment-Variablen von **qmail-inject.***

Bei der Erzeugung von Nachrichten berücksichtigt **qmail-inject** drei Fälle:

- Die Nachricht ist neu. In diesem Fall wird ein komplett neuer und "minimalistischer" Nachrichten-Header geformt.
- Die Nachricht ist eine Antwort auf eine bereits in Empfang genommene E-Mail. Dann existiert bereits eine Nachrichten-Header und dieser muss modifiziert und ergänzt werden. Die neue Nachricht ist dann entweder ein Reply an den Absender (erkennbar an der Subject-Zeile mit vorgestelltem "Re:" oder "Antw:") oder ein Forward an neue Empfänger (Subject-Zeile mit vorgestelltem "Fwd:" oder "Wtlg.:").

- Die Nachricht enthält ein oder mehrere "Resent:" Header-Felder. Der — von aktuellen MUA nicht mehr praktizierte — Mechanismus erzeugt eine Duplizierung der Header-Felder "Sender:", "From:", Reply-To:", "Date:", "Message-ID:" und anderer beim erneuten Versenden der Nachricht in Form von "Resent-X:", wobei "X" für das ursprüngliche Feld steht. Dan Bernstein hält diesen in RFC822 beschriebenen Mechanismus für überflüssig (`http://cr.yp.to/immhf/resent.html`), und dementsprechend behandelt **qmail-inject** das Neuschreiben des Headers dieser Nachrichten sehr individuell

Bei der Weiterverarbeitung von E-Mails kann auf die Gestaltung des neuen Nachrichten-Headers durch das Setzen der `$QMAILINJECT` Environmen-Variablen Einfluss genommen werden. Dies ist sicherlich nur in Aunahmefällen notwendig

`$QMAILINJECT` Parameter	**Bedeutung**
`c`	Comment address style Normales Format der Kommentarzeile im "From:" Feld; normalerweise die Adresse.
`s`	Set Return-Path Keine Auswertung der "Return-Path"-Angabe als Rücksendeadresse. Normalerweise wird mittles des "Return-Path" die SMTP Sender-Adresse unter Umgehung aller Environment-Varablen gesetzt. Ferner wird der Header "Return-Path" gelöscht.
`f`	From: Löschen aller "From:" Angabe im Nachrichten-Header. Nomalerweise wird durch das "From" Feld in der Nachricht das Hinzufügung des "From" Feldes durch **qmail-inject** unterdrückt.
`i`	Ignore Message-ID Unterdrückung des Message-ID-Feldes; asonsten wie bei `f`.
`r`	Per-Recipient VERP Pro Empfänger wird ein Variable Envelop Return Path VERP generiert, der im "Sender" Feld zusätzlich die Angabe des Empängers beinhaltet (`Send: Sender-Recipient`).
`m`	Per-Message VERP Pro E-Mail wird ein VERP generiert, der im "Sender" Feld zusätzlich das aktuelle Datum und den PID von **qmail-inject** enthält.

Tabelle 6-13: Paramter der `$QMAILINJECT` Environment-Variablen.

6.5.4.3 qmail-inject Konfigurationsdateien

Neben den benutzerspezifischen Informationen nutzt **qmail-inject** hostspezifischen Einstellungen, in im Kontrollverzeichnis von Qmail hinterlegt werden. Diese beinhalten im wesentlichen Defaultwerte, wie der Domain- bzw. Hostteil der Absenderadresse im Nachrichtenkopf zu erstellen ist. Dan Bernstein betrachtet dies als *address qualificationen*, was in der man-page `addresses` aufgeführt ist.

Alle diese Konfigurationsdateien besitzen in Bezug auf die oben genannten Parametern und Environment-Variablen lediglich nachrangige Bedeutung.

Konfigurationsdatei	**Bedeutung**
`defaulthost`*)	Angabe des Hostname im Absender; normalerweise der Name, der in der Datei `me` steht. Der Eintrag wird für alle Adressen ohne Hostteil im Absender eingesetzt. Der angegebene Name braucht nicht dem realen Hostname zu entsprechen, sondern kann z.B. nur den Domainname enthalten: `localpart -> localpart@defaulthost`
`defaultdomain`*)	Angabe für die Domaine im Absender; normalerweise der Name, der in der Datei `me` steht. Dieser Eintrag wird zu allen spezifizierten Hostadressen ohne "." ergänzt: `localpart@localhost -> localpart@localhost.defaultdomain`
`plusdomain`*)	Name, der an jeden Hostname hinzugefügt wird, der mit einem "+" endet und ersetzt dieses Zeichen: `localpart@hostname+ -> localcart@hostname.plusdomain.`
`idhost`*)	Hostname in der Message-ID Nachrichtenzeile eingesetzt wird; auch hier gilt als Defaulteintrag `me`. Dieser Name kann frei gewählt werden; sollte aber einem existierenden Namen im DNS für die Domaine entsprechen und für jeden Rechner (MTA) eindeutig sein.

*Tabelle 6-14: **qmail-inject** Konfigurationsdateien; *) bleibt die Datei leer, wird als Wert der Name der Konfigurationsdatei angenommen.*

6.5.5 SMTP-Server

Das Qmail-Programm **qmail-smtpd** ist der SMTP-Server von Qmail. Dieser führt den SMTP-Dialog mit dem Client durch, nimmt letztlich die E-Mails entgegen und stellt sie — wenn entsprechende Kriterien erfüllt sind — in die

qmail-queue ein. Das Programm ist "8-bit clean" und erlaubt somit auch Nicht-ASCII Zeichen in der E-Mail. Es verlangt jedoch, dass E-Mails das Zeilen-Ende-Zeichen CR/LF aufweist; der Empfang von E-Mails, die diese Anforderung verletzten, wird abgelehnt und der sendende MTA erhält folgende nette Belehrung :

```
451 See http://pobox.com/~djb/docs/smtplf.html.
```

qmail-smtpd ist ESMTP-konform und unterstützt somit sowol die transparente 8-Bit-MIME-Übertragung als auch das SMTP-Pipelining. Hierzu gibt es allerdings eine Ausnahme: Übermittelt der SMTP-Client zunächst die Grösse der zu übertragenden E-Mails, so gibt **qmail-smtpd** dem sendenden MTA beim Überschreiten des in der Konfigurationsdatei `databytes` definierten Wertes keine korrekte Response zurück, sondern erst beim Empfang der E-Mail. **qmail-smtpd** traut also dieser Angabe nicht, sondern wartet erst einmal ab und wertet die tatsächlich übertragende Datenmenge aus, bis es eine entsprechende Protokollmitteilung generiert. Ist in `databytes` keine Begrenzung vorgegeben, nimmt **qmail-smtpd** E-Mails beliebiger Grösse entgegen.

Das Programm besitzt lediglich zwei "innere" Beschränkungen:

Beschränkungen

- Der Timeout der SMTP-Sitzung, der auf 1200 Sekunden (20 Minuten) ausgelegt ist, sich allerdings auch parametrieren lässt und
- die Anzahl der "SMTP-Hops", d.h. die Anzahl der SMTP-Relays über die die Nachricht geleitet wurde (Felder "Received:" bzw. "Delivered-To:" im Nachrichten-Header), die auf 100 beschränkt sind

6.5.5.1 Empfangs-Regularien

Für das Verständnis der Arbeitsweise von **qmail-smtpd** ist es wichtig zu verstehen, dass das Programm eine Unterscheidung trifft zwischen

- lokalen Sendern und
- entfernten Sendern sowie
- lokalen Empfängern,
- entfernten Empfängern.

Vorrang in der Deklaration besitzt zunächst die Feststellung eines lokalen bzw. entfernten *Senders*, was durch die gesetzte `$RELAYCLIENT` Variable bewirkt wird. Diese wird (in der Regel) von der festgestellten IP-Adresse des *sendenden SMTP-Clients* (über die Environment-Variable `$TCPREMOTEIP`) abhängig gemacht. Dieses Verhalten — die Auswertung von Informationen aufgrund von Flags (bzw. hier gesetzter Environment-Variablen) zu kopplen — soll als *Split Horizion* Verfahren bezeichnet werden. Ist diese Variable `$RELAYCLIENT` gesetzt, werden die Dateien `rcpthosts`/`morercpthosts` nicht gelesen.

Nachrangig erfolgt die Unterscheidung hinsichtlich des bzw. der *Empfänger* durch die Auswertung des "RCPT TO:" im SMTP-Envelope. Die Fallunterscheidung hinsichtlich der Empfänger wird über die Auswertung der Konfigurationsdateien `rcpthosts` und `morercpthosts` vorgenommen. Wir bemerken, dass die Angabe des "MAIL FROM:" zunächst unberücksichtigt bleibt. Somit können folgende Kombinationen auftreten:

	lokaler Sender	**entfernter Sender**
lokaler Empfänger	`$RELAYCLIENT` per Subnetz gesetzt. Domain in `rcpthosts/morercpthosts`*) eingetragen	Domain in `rcpthosts/morercpthosts` eingetragen.
entfernter Empfänger	`$RELAYCLIENT` per Subnetz gesetzt.	*Annahme der E-Mail verweigert!* Mitteilung: `553 sorry, that domain isn't in my list of allowed rcpthosts (#5.7.1)`

*Tabelle 6-15: **qmail-smtpd** Sende/Empfangsdomain-Klassifizierung; *) wird nicht ausgelesen.*

Environment-Variablen vs. Konfigurationsdateien

An diesem Beispiel erkennen wir, dass die Übergabe von Parametern an **qmail-smtpd** sowohl in Form von Environment-Variablen als auch über Konfigurationsdateien erfolgen kann. Der Einsatz von Environment-Variablen bringt erhebliche Vorteile mit sich:

- Das Setzen der Variablen kann von einer vorher erfolgten Auswertung abhängig gemacht werden (z.B. kann `$RELAYCLIENT` aufgrund einer Fallunterschiedung gesetzt werden),
- die Information wird transparent von einem Agenten-Programm (**tcpserver**) an z.B. **qmail-smtpd** übergeben werden, sodass Agenten-Programm und **qmail-smtpd** über die identischen Daten verfügen,
- bei gesetzter Envionment-Variablen kann auf das Auslesen einer Konfigurationsdatei verzichtet werden, da die relevante Information bereits vorhanden ist; wir sparen also vor allem I/O-Aufrufe.

Neben der Environment-Variablen `$RELAYCLIENT` lässt sich als Beispiel hier `$DATABYTES` aufführen. Für Rechner, die aus dem Internet Kontakt mit aus aufnehmen — also entfernte Sender — können wir z.B. den Wert hier auf 5 MB (5000000) setzten, während wir für Sender aus dem Intranet (lokale Sender) die E-Mail-Grösse unbeschränkt lassen.

6.5.5.2 Environment-Variablen

qmail-smtpd besitzt keinen DNS-Stub-Resolver und benötigt daher immer ein "TCP-Agenten". Dan Bernstein stellt diesen in der Qmail-Source in Form des Programms **tcp-env** zur Verfügung. Bei Verwendung des UCSPI-Paketes wird dieser durch **tcpserver** ersetzt. **qmail-smtpd** liest zur Laufzeit die folgenden Environment-Variablen aus:

Environment-Variable	Bedeutung
`$DATABYTES`	Maximale Grösse (in Byte) einer einlaufenden E-Mail, überschreitet eine E-Mail diese Grösse, wird der Empfang abgelehnt.
`$RELAYCLIENT`	Ist diese Variable gesetzt (ohne Wert), wird der Sender als "lokal" klassifiziert und der Empfang von E-Mails auch dann akzeptiert, wenn dessen Empfänger nicht als "lokal" klassifiziert ist; ansonsten wird dies abgelehnt.
`$TCPREMOTEIP`	vgl. Abschnitt 4.6.1
`$TCPLOCALHOST`	vgl. Abschnitt 4.6.1.
`$TCPLOCALIP`	vgl. Abschnitt 4.6.1.
`$TCPREMOTEHOST`	vgl. Abschnitt 4.6.1.

*Tabelle 6-16: Von **qmail-smtpd** ausgelesene und genutzte Environment-Variablen.*

Die `$TCP*`-Variablen werden entweder über das Programm **tcp-env** oder **tcpserver** zur Verfügung gestellt. Währenddessen sind `$DATABYTES` und insbesondere `$RELAYCLIENT` sog. Benutzer-definierte Variablen. In der Regel wird die Deklaration dieser Variablen abhängig gemacht von dem SMTP-Client bzw. dessen IP-Adresse (die über `$TCPREMOTEIP` **qmail-smtpd** mitgeteilt wird). Hierdurch arbeitet **qmail-smtpd** in einem "Split-Horizion" Verfahren, d.h. abhängig davon ob (und wie) diese Variable gesetzt ist. Besonders wichtig ist hierbei `$RELAYCLIENT`. E-Mails von SMTP-Clients für die diese Variable gesetzt ist werden als "lokale" Sender betrachtet. **qmail-smtpd** akzeptiert das Weiterleiten von E-Mails von solchen Sendern sowohl an lokale als auch an entfernte Empfänger.

Hieraus wird klar, dass die Kenntnis der IP-Adresse des Senders eine überragende Bedeutung besitzt. Im Gegensatz zu der Angabe des "MAIL FROM:" im SMTP-Envelope, des SMTP-Begrüssungsnamens (HELO/EHLO) und möglicherweise auch des DNS-FQDN (über ein DNS-Spoofing), ist die IP-Adresse immer eindeutig; sieht man einmal von der prinzipiellen Problematik der Network Address Translation (NAT) ab.

6.5.5.3 Konfigurationsdateien

qmail-smtpd stützt sich bei seiner Arbeit auf einige Konfigurationsdateien, die in Tabelle 6-17 aufgeführt sind. Wir beachten, dass Änderungen hierin "on-the-fly" wirksam sind, d.h. beim erneuten Aufruf von **qmail-smtpd**, in der Regel also die nächste einlaufende E-Mail. Neben der Datei `me`, die als Statthalter für einige untergeordnete (aber nichtsdestotrotz gelegentlich sehr wichtige) Dateien herangezogen wird, hat die Konfigurationsdatei `rcpthosts` die wichtigste Bedeutung inne:

Selektives Relay

Erst durch das Einsetzen von Host- und Domain-Einträgen in die Datei `rcpthosts` wird der Qmail-MTA zum selektiven Relay. Ist diese Datei leer oder nicht Existent, agiert Qmail als offenes Relay.

Beim Ausführen des Konfigurationsskriptes während der Standardinstallation wird diese Datei allerdings befüllt. Wir kommen auf die Frage der konkreten Ausgestaltung der `rcpthosts`-Datei noch einmal unter dem nachfolgenden Kapitel im Abschnitt Relaying zurück.

Konfigurationsdatei	**Bedeutung**
`badmailfrom`[#)]	Liste von SMTP-Sendern ("Mail From:), für die der Empfang von E-Mails abgelehnt wird. Dies kann entweder der komplette Name sein, oder lediglich der Host- bzw. Domainpart (@Domainame).
`databytes`[+)]	Grösse der Datei, wie sie auf Platte geschrieben werden wird; daher nicht unbedingt identisch mit der Anzahl der übertragenen Bytes (CR/LF Umsetzung).
`localiphosts`	Besitzt ein Rechner mehrere IP-Adressen, wird der hier eingetragene Namen als Rechnernamen eingesetzt (statt des Primärnames). **qmail-smtpd** substituiert Empfängeradressen ("Rcpt To:") mit IP-Notation [d.d.d.d] durch diesen Namen. Als Default wird der Inhalt der Datei `me` herangezogen.
`morercpthosts`[*+)]	Zusätzliche RCPT-Domains, die im CDB-Format vorliegen. Diese wird mittels des Kommandos **qmail-newmrh** aus einer `rcpthosts` vergleichbaren Datei erzeugt.
`rcpthosts`[*+)]	RCPT-Domains, deren Empfänger als "lokal" gelten. Diese Datei wird nicht ausgewertet, falls die Environment-Variable `$RELAYCLIENT` gesetzt ist, vielmehr wird der Wert dieser Variable per VERP-Mechanimus der "Rcpt To:" Adresse hinzugefügt. Die hierarchisch gebildeten Domain-Adressen können durch

Konfigurationsdatei	Bedeutung
	die Angabe des führenden Punktes "." abgekürzt werden. SMTP Empfängeradressen ohne "@" werden als "lokal" betrachtet.
`smtpgreeting`	SMTP Begrüssungsspruch, per Default einfach der Inhalt der Datei `me`.
`timeoutsmtpd`	Anzahl der Sekunden, die **qmail-smtpd** bis zur nächsten Übermittlung von Daten wartet, normalerweise 1200 Sekunden.

*Tabelle 6-17: Konfigurationsdateien von **qmail-smtpd**; #) Kommentare in Form eines "#" in erster Spalte möglich, *) Wildcards sind gestattet, +) Verarbeitung abhängig von der gesetzten Environment-Variablen.*

Wildcards

Es ist darauf hinzuweisen, dass **qmail-smtpd** (und auch ganz Qmail) keine Wildcards in Form von Unix-Shell Pattern-Matching (Regular Expressions) ermöglicht. Wildcards in dem hier beschriebenen Sinne sind an die hierarchische Struktur einer FQDN- bzw. SMTP-Adresse gebunden. Sie nutzten daher bei SMTP-Adressen die Unterscheidung zwischen Benutzerteil (d.h. links vom "@", also z.B. "erwin_hoffmann" bei "erwin_hoffmann@compuserve.com") und Domain- bzw. Hostteil. Oder aber sie berücksichtigen die hierarchische Struktur der IP-Adresse bzw. des FQDN bei einem Rechnernamen. So steht z.B. "exampledomain.com" genau für die Domain "ExampleDomain.Com", während ".exampledomain.com" als Wildcard auch für alle untergeordneten Domains aufzufassen ist. Vergleichbares gilt für IP-Adressen: Z.B. ist "127.0.0.1" genau die Loopback-Adresse, während "127.0.0." das mit Class-C Netzmaske klassifizierte Loopback-Netz ist; im Gegensatz zu "127.0.0" das das Class-B Netzwerk "127.0.0" darstellt.

6.5.5.4 SMTP-Adressen

qmail-smtpd akzeptiert sog. NULLSENDER E-Mails also solche, wo kein "MAIL FROM:" spezifiziert ist. Solche E-Mails können z.B. bei sog. Double-Bounces auftreten und gehen dann an lediglich an den Postmaster. Leider ist es so, dass NULLSENDER E-Mails an reale Benutzer häufig mit Viren verseucht sind.

qmail-smtpd substituiert E-Mails bei denen die RCPT TO: Adresse als IP-Adresse angegeben ist, umgehend durch den lokalen Hostnamen. Hierdurch wird z.B. eine E-Mail mit "RCPT TO: <erwin@[192.168.192.2]>" nach <erwin@qmailer.fehnet.de> bereits von **qmail-smtpd** umgesetzt noch bevor die Dateien `rcpthosts`/`morercpthosts` ausgewertet werden. Wir beachten, dass dies auch für die Loopback Adresse 127.0.0.1 gilt. Dieses Verhalten kann durch die Konfigurationsdatei `localiphost` gesteuert werden. IP-Adressen

sind hierbei in Klammern "[IP-Adresse]" zu setzen, ansonsten werden sie wie Domain-Names betrachtet.

Entsprechend RFC 1122 gilt die IP-Adresse 0.0.0.0 als "this host, this network". Standard **qmail-smtpd** erkennt diese spezielle Adresse nicht als lokal. Jedoch kann von Scott Gifford's "IPME" Patch Gebrauch machen (`http://www.suspectclass.com/~sgifford/qmail/qmail-1.03-moreipme-0.4.patch`). Dies ist aber sicherlich nur in sehr wenigen Fällen relevant.

E-Mails mit nicht vorhandenem Domain-Teil, z.B. "RCPT TO: <erwin>" (aber nicht "RCPT TO: <erwin@>" !) werden beim Empfang akzeptiert (und um den lokalen Domain-Name ergänzt).

Gross/Kleinschreibung von Adressen

Der Host- bzw. Domain-Teil einer FQDN- oder SMTP-Adresse ist per construktionem unabhängig davon, ob in Gross- oder Kleinbuchstaben geschrieben (case-insensitiv). Qmail — und speziell auch **qmail-smptd** — wertet aber auch den Benutzerteil der SMTP-Adresse (also der Teil links vor dem "@") unspezifisch aus und konvertiert die gesamte Adresse in "lower case". Davon betroffen ist natürlich auch der Eintrag in der Datei `badmailfrom`; hier spielt Gross- und Kleinschreibung keine Rolle. Die Weitergabe der Addressinformation an die nachfolgenden Qmail-Programme geschieht jedoch transparent, d.h. mit ursprünglicher Gross /Kleinschreibung.

6.5.5.5 Filter-Mechanismen

Den einzigen Filtermechanismus auf SMTP-Basis, den Dan **Bernstein qmail-smtpd** gegönnt hat, ist das Filtern auf Grundlage der "MAIL FROM:" Angabe im SMTP-Envelope. Sollen Absender aufgrund ihrer vollqualifizierten Adresse oder aufgrund des Host- bzw. Domain-Teils der Absenderadresse hier blockiert werden, sind diese in die Konfigurationsdatei `badmailfrom` aufzunehmen. Erkennt **qmail-smtpd** im SMTP-Dialog mit dem Client eine in **badmailfrom** vorliegende Adresse, so wird dies mit

```
553 sorry, your envelope sender is in my badmailfrom list
(#5.7.1)
```

quittiert, die SMTP-Verbindung aber nicht abgebrochen. Der sendende MTA kann somit in der bestehenden Sitzung neue Anläufe probieren. `badmailfrom` ist zwar eine nützliche Datei um spezielle Absender zu blockieren, stellt jedoch keinen hinreichen Spam-Filtermechanismus dar.

6.5.5.6 SMTP-Dialog, Protokoll-Antworten, DSNs

Bei der Konversation mit seinem SMTP/QMTP-Partner ist **qmail-smtpd** nicht allzusehr gesprächig. Es benutzt nur einen Subset der möglichen SMTP Return-Codes und der *Delivery Status Notification* (DSN).

Dies kann nur als positive Eigenschaft vermerkt werden. Denn erstens, kostet natürlich jeder überflüssige Dialog Bandbreite und Zeit. Zweitens gibt es bei dem — in der Regel problemlosen — Austausch von E-Mails sowie nur wenig zu berichten. Und drittens — und das ist wichtig — ist es sowieso angeraten nicht allzuviel zu Quatschen und einem möglichen Angreifer sowenig wie möglich an Informationen in die Hand zu geben.

Daher hat Dan Bernstein auch schon gar nicht so überflüssige SMTP-Befehle wie EXPN oder VRFY Implementiert. Der einzige Hinweise, den sich **qmail-smtpd** auf seine Natur erlaubt, ist der "HELP" Befehl, den es mit der unten stehenden Antwort quittiert. Dan Bernstein nutzt diese Eigenschaft, um gelegentlich Surveys von MTAs durchzuführen und die Nutzung von Qmail im Internet zu analysieren.

Hierbei brauchen wir jedoch keine Angst vor versteckten oder nicht dokumentierten Kommandos zu haben. Qmail ist Open Source. **qmail-smtpd** kann nicht mehr (aber eben auch nicht weniger) als in seinem Quelltext steht. Und dieser ist mit 421 Zeilen überschaubar.

qmail-smtpd **Protokollmitteilung**	**Bedeutung**
`214 qmail home page: http://pobox.com/~djb/qmail.html`	Antwort auf die Eingabe von "Help" im SMTP-Dialog.
`250-PIPELINING`	Antwort auf ESMTP vom Client.
`250 8BITMIME`	Antwort auf ESMTP vom Client.
`250 ok`	Kommando wurde akzeptiert.
`250 flushed`	E-Mail wurde in der laufenden Sitzung schon einmal empfangen.
`252 send some mail, i'll try my best`	SMTP-Client hat "DATA"-Befehl abgegeben; E-Mail wurde angenommen.
`354 go ahead`	QTP "Go Ahead" Befehl.
`421 out of memory (#4.3.0)`	Zu wenig Speicher zum Empfang der E-Mail.
`421 unable to read controls (#4.3.0)`	Konfigurationsdateien können nicht gelesen werden (`me`).
`421 unable to figure out my IP addresses (#4.3.0)`	Lokale IP-Adresse kann nicht ermittelt werden.
`451 See http://pobox.com/~djb/docs/smtplf.html.`	Mitteilung bei fehlendem CR in der E-Mail.

qmail-smtpd **Protokollmitteilung**	**Bedeutung**
`451 timeout (#4.4.2)`	Timout beim SMTP-Dialog.
`451 qqt failure (#4.3.0)`	Fehler beim Einstellen der Nachricht in die Queue.
`502 unimplemented (#5.5.1)`	SMTP-Kommando unbekannt.
`503 MAIL first (#5.5.1)`	DATA-Befehl wurde zu früh abgegeben.
`503 RCPT first (#5.5.1)`	DATA-Befehl wurde zu früh abgegeben.
`552 sorry, that message size exceeds my databytes limit (#5.3.4)`	Konfigurierte E-Mail-Grösse wurde überschritten.
`553 sorry, your envelope sender is in my badmailfrom list (#5.7.1)`	Absender ("Mail from:") steht in `badmailfrom`.
`553 sorry, that domain isn't in my list of allowed rcpthosts (#5.7.1)`	Entfernter Sender und SMTP-Empfänger steht nicht in `rcpthosts`.
`554 too many hops, this message is looping (#5.4.6)`	E-Mail lief über zu viele SMTP-Relays.
`555 syntax error (#5.5.4)`	Fehler beim Befehl.

*Tabelle 6-18: **qmail-smtpd** Mitteilungen an den SMTP-Client.*

6.5.5.7 Received-Zeilen im E-Mail-Header

RFC (2)822 fordert, dass der empfangende MTA eine Pfad-Information in der einlaufenden E-Mail aufzeichnet. Qmail entspricht diesem, indem **qmail-smtpd** und **qmail-queue** jeweils eigenständige "Received:"-Header-Zeilen hinzufügen:

```
Received: (qmail 18969 invoked from network); 26 Sep 1995
04:46:54 -0000

Received: from relay1.uu.net (HELO uunet.uu.net)
(7@192.48.96.5)

     by silverton.berkeley.edu with SMTP; 26 Sep 1995
04:46:54 -0000
```

- Die erste Zeile stammt von **qmail-queue**. Hier wird zunächst die PID von

qmail-queue ausgegeben und anschliessend eine Information, von wem die Nachricht eingestellt wurde. **qmail-queue** ist so "smart", dass es für uns die ID des Systemaccounts *qmaild* in "from network", die von *alias* in "by alias" sowie die von *qmails* in "for bounce" übersetzt. Ist dies nicht der Fall, wird die UID im Klartext mitgeteilt.

- In der zweiten Zeile liefert uns **qmail-smtpd** bedeutend mehr Informationen. Angegeben wird der lokale Name (`from $TCPLOCALHOST`), die HELO-Kennung des SMTP-Clients (falls vorhanden), dessen Absender (`$TCPREMOTEINFO@$TCPREMOTEIP`) sowie sein FQDN (`by $TCPREMOTEHOST`). Bei keiner "Received:"-Zeile darf natürlich der Datumstempel fehlen.

Die Abfolge der "Received:"-Zeilen ist i.d.R. streng chronologisch, d.h. der E-Mail-Header sollte um jede folgende "Received:"-Zeile "nach oben" ergänzt werden. Der E-Mail-Header gibt somit Auskunft über den Weg der E-Mail im SMTP-Mail-Verbund. Es gibt jedoch keine Garantie auf die Richtigkeit der hier hinterlegten Informationen. Insbesonders ist der Zeitstempel häufig falsch oder zumindest unpräzise.

6.5.5.8 Logging

Abschliessend stellt sich die Frage, wie das Logging von **qmail-smtpd** vonstatten geht. Die Antwort ist simpel: Gar nicht. **qmail-smtpd** logt kein Byte seiner Aktivität mit, sondern überlässt diese Aufgabe ausschliesslich den Programmen **tcpserver**/**tcp-env** und **qmail-send**.

qmail-smtpd Logging bei ISPs

Daher bekommt der Qmail-Sysadmin auch in der Regel nicht davon mit, wenn z.B. eine Ablehnung der E-Mail aufgrund eines Eintrags in `badmailfrom` erfolgt ist. Gerade die Nutzung von Qmail bei einem Internet Service Provider (ISP) verlangt aber in der Regel eine relativ genaue Kenntnis des SMTP-Verkehrs; speziell, wenn dieser einmal "klemmt". Zwar kann mit dem Hilfsprogramm **recordio** die gesamte SMTP-Sitzung mitgeschrieben werden, jedoch ist für die 99% der Sitzungen, die problemfrei ablaufen ein Overkill; schiesslich ist man in der Regel an dem 1% der E-Mails interessiert, die Schwierigkeiten machen. Daher kommt man an dieser Stelle nicht umhin**, qmail-smtpd** zu patchen und ein Logging zu erzwingen.

6.5.6 Local Delivery

Bislang haben dir die Fälle betrachtet, dass eine Nachricht bzw. eine E-Mail in die Qmail-Queue eingestellt wird. Wir wechseln nun den Kontext und verfolgen, was passiert, wenn eine Nachricht auszuliefern bzw. zuzustellen ist, d.h. aus der Queue herausgenommen werden muss. Wir betrachten zunächst den Qmail Local Delivery Agent: **qmail-lspawn** in Verbindung mit **qmail-local**.

Damit diese in Aktion treten können, ist relevant, ob der Empfänger einer Nachricht als lokal erkannt bzw. angenommen wurde. Wir berücksichtigen, dass Qmail zu diesem Zeitpunkt keine Kenntnis mehr besitzt, wie die Nachricht in die Queue eingestellt wurde.

Neben der Konstruktion der Qmail-Queue ist die lokale Zustellung bei Qmail sicherlich Dan Bensteins zweites Meisterstück bei Qmail — aber leider von erheblicher Komplexität begleitet — führt doch Dan Bernstein hier zum ersten mal bei einem MTA das "virtual Domains" Konzept ein. Ferner kann der Delivery-Prozess auf Anwenderseite über die bereit gestellten Mechanismen sehr fein gesteuert werden.

6.5.6.1 Die Kette: qmail-send, qmail-lspawn und qmail-local

Es ist Aufgabe von **qmail-send** festzustellen, ob eine Nachricht in der Sende-Queue für einen lokalen oder einen entfernten Empfänger bestimmt ist. **qmail-send** ruft **qmail-lspawn** unter den folgenden Umständen auf:

- Die Nachricht wurde von **qmail-inject** bzw. **qmail-smtpd** lediglich unter Angabe des Benutzerteil als Empfängeradresse, d.h. z.B. "erwin" in die Queue eingestellt. Per Default wird der lokale Domain-Name ergänzt.
- Der Domain-Teil der Empfängeradresse ist in der Konfigurationsdatei `locals` eingetragen.
- Der Domain-Teil der Empfängeradresse ist in der Konfigurationsdatei `virtualdomains` zu finden.

Einträge in `locals` und `virtualdomains` schliessen sich gegeneinander aus, wobei ein Eintrag in `locals` Vorrang geniesst. Wir betrachten exemplarisch den Aufbau der Datei `locals`:

```
localhost
qmailer.fehnet.de
fehnet.de
.fehnet.de
```

Listing 6-13: Beispiel für die Konfigurationsdatei `locals`.

Wir sehen hier zunächst die *Host-Einträge* `localhost` und `qmailer.fehnet.de` gefolgt von Domain-Einträgen `fehnet.de` sowie `.fehnet.de` und erinnern uns an die weiter oben gemachten Aussagen zur Verwendung von Wildcards. Der Eintrag `localhost` ist eigentlich überflüssig, wird aber dann manchmal gebraucht, wenn bestimmte Daemons — wie z.B. der **syslogd** oder speziell auch **fetchmail** — nicht auf den konfigurierten Hostname zurückgreifen, sondern mit localhost, d.h. auf Grundlage der IP-Adresse

127.0.0.1 arbeiten.

qmail-send ist also für die Interpretation einer Empfängeradresse zuständig, stützt sich dabei auf die in Abschnitt 6.3.3 vorgestellten Konfigurationsdateien, die allesamt beim Start von qmail eingelesen werden. Durch ein `kill -HUP qmail-send` wird das Neulesen von `locals` und `virtualdomains` erzwungen.

Anschliessend ruft **qmail-send qmail-lspawn** auf (vgl. Abbildung 6-1). Eine Nachricht kann natürlich mehrere Empfänger enthalten (vor allem bei Nutzung des "CC:" und "BCC:" Mechanismus). **qmail-lspawn** ist verantwortlich dafür, dass die Nachricht einzeln an diese Adressaten geht und versorgt anschliessend **qmail-local** mit dem Default-Delivery Verfahren. Der Aufruf von **qmail-local** erfolgt hierbei asynchron, d.h. nicht unbedingt in der Reihenfolge der Empfänger. Eine wichtige Aufgabe von **qmail-lspawn** besteht darin, zu überprüfen, ob ein Empfänger als spezieller Qmail-User eingetragen ist. Ist dies der Fall, werden die hierin hinterlegten Parameter **qmail-local** übergeben.

Das Gespann **qmail-send**, **qmail-lspawn** und **qmail-local** nimmt die Zustellung einer Nachricht nach dem folgenden Regelwerk (Abbildung 6-3) vor:

1. **qmail-send** stellt fest, ob (a) der Empfänger tatsächlich "lokal" im eigentliche Sinne vorliegt, oder ob (b) eine "Virtual Domain" einem lokalen User zugewiesen wurde.

2. **qmail-lspawn** überprüft anschliessend, ob (a) der Empfänger als spezifischer Qmail-User existiert und liest die hier übergebenen Werte aus. Sollte dies nicht der Fall sein, wird (b) per **qmail-getpw** ein User Lookup im Unix-System vorgenommen und die relevanten Informationen (wie z.B. Home-Directory) an **qmail-local** übergeben. War weder (a) noch (b) erfolgreich, wird (c) per Default der Qmail Alias User als Adressat herangezogen.

3. **qmail-local** nutzt die von **qmail-lspawn** vorgegebenen Wert zum Ausliefern der Nachricht an den lokalen Empfänger, wobei es hierbei die ermittelten effektiven User-Rechte annimmt, und die in den dot-qmail Dateien hinterlegten (benutzerspezifischen) Kommandos ausführt.

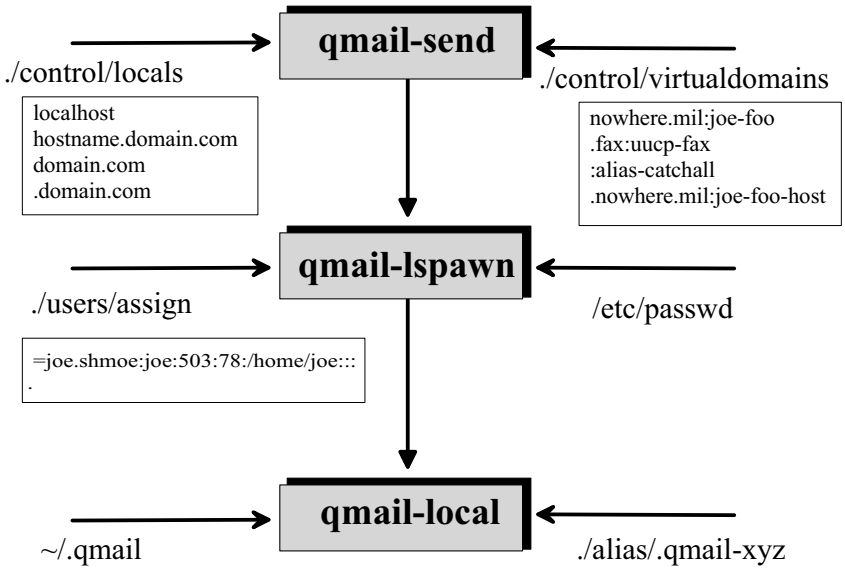

Abbildung 6-3: Hierarchie der lokalen Auslieferung von Nachrichten.Die aus Konfigurationsdateien (deren typische Syntax in einem Kasten zu sehen sind) gelesenen Einträge auf der linken Seite, haben Vorrang vor denen auf der rechten Seite. Der "." steht für das Installationsverzeichnis vom Qmail; in der Regel `/var/qmail`.

Die lokale Auslieferung geht somit in fest definierten Schritten vor, wobei — kurz gesagt — die Ermittlung des Host- bzw. Domainteils der Empfängeradresse **qmail-send** obliegt, die Auswertung des Benutzerteils (im Kontext der lokalen Informationen) durch **qmail-lspawn** vorgenommen wird. **qmail-local** ist somit nur der "Arbeitsknecht", dem die Aufgabe zufällt die Nachricht an den festgestellten Empfänger auszuliefern. Diesen *Mail Delivery Agent* (MDA) realisiert **qmail-local** jedoch auf komplexe Weise, indem es die Vorschriften auswertet, die der Benutzer in einer (lokalen, d.h. in seinem Heimatverzeichnis liegenden) dot-qmail-Datei `~/.qmail` hinterlegt hat; worauf wir weiter unten ausführlich zu sprechen kommen.

Wir schauen uns noch einmal den Aufruf von **qmail-local** an:

```
qmail-local  [  -nN  ]  user homedir local dash ext domain
sender defaultdelivery
```

Die Parameter:

- `user` - Der ermittelte Benutzername entsprechend **qmail-lspawn.**

- `homdir` - Der absolute Pfad zum Heimatverzeichnis von *user*.
- `local` - Die Angabe "local" bei der Empfängeradresse "local@domain".
- `dash` - Das Zeichen für die Qmail-Extension; normalerweise der "`-`".
- `ext` - Die Qmail-Ergänzung (Extension) im Namen nach dem "dash".
- `domain` - Die Angabe "domain" bei der Empfängeradresse "local@domain".
- `sender` - Der (festgestellte) Absender der E-Mail.
- `defaultdelivery` - Die Angabe der Defaultdelivery per **qmail-send** (vgl. Abschnitt 6.4.4)

Die Optionen:

- `-n` Statt die Nachricht zuzustellen, findet lediglich eine Ausgabe der Zustellungsparameter statt.
- `-N` Normale Zustellung.

Mailbox/Maildir Zustellung

Qmail macht eine Unterscheidung bei der Zustellung der Nachricht in eine Mailbox oder ein Mail-Verzeichnis (Maildir). Bei der Auslieferung in eine Mailbox (z.B. der Datei `mbox`) legt **qmail-local** diese Datei beim allerersten Aufruf an; ist hingegen die Zustellung in ein Mail-Verzeichnis vorgesehen, muss dieses per **maildirmake** vorher existieren! Ansonsten kann die Zustellung nicht erfolgen und im **qmail-send** Log findet sich folgender Eintrag:

```
... status: local 1/10 remote 0/20
... delivery 1198: deferral:
unable_to_chdir_to_maildir._(#4.2.1)/
```

Nach erfolgter Zustellung meldet dies **qmail-local** durch den Return-Code 0; jeder Wert darüber weist auf einen Zustellungsfehler hin, wobei der Return-Code 11 einen temporären Fehler darstellt, wenn z.B. ein Mail-Verzeichnis nicht existiert. **qmail-send** nimmt dies zur Kenntnis und versucht die Auslieferung entsprechend des Queuing-Schemas (was weiter unten erläutert wird) später erneut.

6.5.6.2 Delivered-To und Return-Path im Nachrichten-Header

qmail-local führt Buch darüber, ob eine Nachricht bereits an einen Empfänger erfolgreich ausgeliefert wurde. Hierzu ergänzt es den Nachrichten-Header um die Felder "Delivered-To" und "Return-Path". Letzteres ist der Grund, warum das **qmail-local** Kommando den Parameter `sender` aufweist. Wird eine Nachricht von **qmail-local** ausgeliefert, die ursprünglich von **qmail-smtpd** in Empfang genommen wurde, sehen die ersten drei Zeilen im Nachrichten-Header folgendermassen aus:

```
Return-Path: <MAILER-DAEMON@qmailer.fehnet.de>
Delivered-To: erwin@localhost
Received: (qmail 18969 invoked from network); 7 Jun 2002
14:05:11 -0000
Received: from localhost (127.0.0.1) by orion.fehnet.de with
SMTP; 7 Jun 2002 14:05:11 -0000
```

*Listing 6-14: Beispiel für die von **qmail-local** eingefügten "Return-Path" und `Delivered-To` Felder im Nachrichten-Header.*

Die Angabe `Return-Path` enthält also die SMTP-Absenderadresse (sofern angegeben). Diese Information ist unabhängig von den Nachrichtenzeilen `From` bzw. `Sender`, die der MUA des Absenders erzeugt. Mit der `Delivered-To` Zeile wird erreicht, dass Qmail feststellen kann, ob und an wen die E-Mail bzw. Nachricht schon einmal zugestellt wurde. Somit können sog. Mail-Forwarding-Loops erkannt und ausgeschlossen werden.

Die im obigen Beispiel sind zusätzlich zwei Zeilen `Received` aufgeführt. Diese wurden von **qmail-send** erzeugt. In unserem Beispiel wurde die Nachricht per E-Mail von **qmail-smtpd** in Empfang genommen. Daher findet sich zunächst die Angabe "invoked from network" mit der Angabe der PID sowie des Zeitpunkts. Die unterste Zeile ergänzt diese Information um Sender-MTA ("from localhost") und den Empfänger-MTA ("by orion.fehnet.de") und eines Zeitstempels. In der Praxis finden sich häufig mehrere dieser Received Zeilen, pro SMTP-Hop eine. Ob die Angaben hierin aber sinnvoll sind, entscheidet der System-Administrator beim Aufsetzen des Mail-Systems.

6.5.6.3 VERP — Variable Envelope Return Paths

Mit dem VERP-Machanismus (`http://cr.yp.to/proto/verp.txt`) hat Dan Bernstein ein Verfahren eingeführt, das sich als äusserst brauchbar für die Administration von Mailing-Listen herausgestellt hat, aber auch im Standard-Qmail umfangreichen Einsatz findet. Doch was ist VERP?

Eine "normale" E-Mail Adresse besteht bekanntlich aus einem lokalen und einem Host(Domain-)-spezifischen Teil, die durch den "Klammeraffen" @ getrennt sind, typischerweise also "benutzername@host.domain.com". Die Hierarchie des Host-Teils der E-Mail-Adresse ergibt sich aufgrund der Struktur des Domain Name Systems; der lokale Teil der Adresse ist hingegen beliebig, sofern sie den Standards von RFC 821 genügen.

Das VERP-Verfahren nun darin, den lokalen Teil einer E-Mail-Adresse um zusätzliche — benutzerspezifische Informationen — zu ergänzen und beim Empfang der E-Mail diese Information geeignet zu interpretieren. Taucht im lokalen Teil der E-Mail-Adresse ein spezielles Zeichen auf, interpretiert **qmail-**

local den lokalen Teil der Empfängeradresse als VERP-Information, was sich dann wie folgt liest:

```
local@host.domain.com -> benutzername-ext@host.domain.com
```

Hierzu muss natürlich zunächst ein Zeichen als Delimitor definiert sein, das die Qmail-Ergänzung vom eigentlichen Benutzernamen trennt. Beim Standard-Qmail ist dies der Bindestrich "`-`", was aber durch Editieren der Datei **conf-break** (vgl. Abschnitt 6.1) angepasst werden kann.

Das VERP-Verfahren bringt mehrere Konsequenzen mit sich:

1. Das Delimitor-Zeichen zum Aufteilen zwischen dem eigentlichen Benutzername(n) und der VERP-Ergänzung muss eindeutig sein; es darf nicht als Teil von `benutzername` auftauchen. Mit einem Blick auf die für den SMTP-Verkehr relevante 7-Bit-ASCII-Zeichentabelle und den Vorgaben gemäss RFC821, verbieten sich von selbst alle alphanumerischen Zeichen, sowie ein Grossteil der Sonderzeichen. Es bleiben folgende übrig: "-", "_", "=", "`,`", "`;`", "#" und "`:`". Die Gesamtlänge des benutzerspezifischen Teils der Adresse sollte 80 Zeichen nicht überschreiten

2. Mehrere Qmail-Ergänzungen sind möglich; müssen jedoch denselben Delimitor besitzen. Hierdurch wird eine (maximal dreistufige) Hierarchie von Qmail-Ergänzungen möglich: `ext -> ext2-ext3-ext4`.

3. Das Versenden von E-Mail mit Qmail-Ergänzung dient vor allem zur Steuerung der Replys. Gängige MUAs nutzen beim Erstellen einer Antwort an den Absender automatisch als Nachrichten-Feld "`From:`" oder "`Reply-To:`". Wird dieses in der ursprünglichen Nachricht mit der VERP-Angabe versehen, geht die E-Mail nicht an `benutzername@host.domain.com`, sondern an `ext-benutzername@host.domain.com`. Dies ist besonders für E-Mail-Listen interessant, wo die gleiche Nachricht mit verschiedenen Absender-Adressen an unterschiedliche Empfänger verschickt wird. Hierdurch ist es möglich, Rückläufer in Form von Bounces — also nicht zustellbare E-Mails — gezielt, d.h. Empfänger-spezifisch zu verarbeiten.

4. Beim Empfang von E-Mails mit einer Qmail-Erweiterung muss der MDA in der Lage sein, E-Mails an den gleichen `benutzername` aber mit unterschiedlichen Qmail-Ergänzungen an das Konto des eigentlichen Empfängers zu übertragen. Es obliegt dem Empfänger, die Nachrichten mit unterschiedlichen VERPs geeignet zu verarbeiten.

Der VERP-Mechanismus findet beim "normalen" Qmail speziell bei der Deklaration von virtuellen Domains sowie beim Qmail Alias-User Einsatz. Hierauf soll weiter unten im Detail eingegangen werden.

Adobe Kundenmailings

Obwohl ich nicht weiss, ob Adobe Qmail verwendet, benutzt Adobe bei seiner Kundeninformation die VERP-Methode. Ist man als Kunde mit einer E-Mail-Adresse bekannt, verschickt Adobe Marketinginformationen zu ihren Kunden mit einem "From:" Nachrichtenfeld im VERP-Format. Dies sieht dann folgendermassen aus:

```
From: Adobe Systems Incorporated  <Mailinginfo-
Kundennummer@adobe.com>
```

Antwortet der Kunde mit der Return-Adresse, kann dies Adobe sofort mit der bekannten Kundennummer und dem aktuellen Mailing korrelieren.

6.5.6.4 Environment-Variablen

Nachdem **qmail-lspawn qmail-local** aufgerufen hat, wird dieses unter der effektiven User-Id gestartet und wechselt ins Verzeichnis des Nachrichtenempfängers. **qmail-local** übermittelt aber nicht nur die eigentliche Nachricht, sondern bringt auch Informationen über den Absender und die Verarbeitungsschritte von Qmail in Form von Environment-Variablen mit:

qmail-local **Environment-Variable**	**Bedeutung**
$SENDER	Die Adresse des Absenders im SMTP-Umschlag.
$NEWSENDER	Die SMTP-Absenderadresse des Forwarding-Senders.
$RECIPIENT	Die Adresse des Empfänger im Umschlag in der Form: `local@domain.`
$USER	Der Empfänger im Unix-System..
$HOME	Das Home-Directory des Empfängers.
$HOST	Der Domain-Teil des Empfängeradresse.
$HOST2	Der Teil der Host-Adresse vor dem letzten Punkt.
$HOST3	Der Teil der Host-Adresse vor dem vorletzten Punkt.
$HOST4	Der Teil der Host-Adresse vor dem drittletzten Punkt.
$LOCAL	Der lokale (Benutzer-) Teil der SMTP-Adresse.
$EXT	Die Qmail-Extension.
$EXT2	Teil der Qmail-Extension nach dem ersten Bindestrich.
$EXT3	Teil der Qmail-Extension nach dem zweiten Bindestrich.
$EXT4	Teil der Qmail-Extension nach dem dritten Bindestrich.
$DEFAULT	Name der "Default"-Ergänzung der dot-qmail Default-Datei,

	normalerweise `.qmail-default`.
`$DTLINE` und `$RPLINE`	Der Inhalt der "Delivered-To:" und der "Return-Path:" Zeilen im Nachrichten-Header.
`$UFLINE`	UUCP `From_:` Format; das von **qmail-local** für mbox-formatierte Dateien hinzugefügt wird.

*Tabelle 6-19: Environment Variablen von **qmail-local**; vgl. man-page **qmail-command**.*

6.5.6.5 dot-qmail-Dateien

Die Feinsteuerung der Auslieferung der Nachricht im Verzeichnis des lokalen Empfängers geschieht über die sog. dot-qmail-Dateien, die sich im Heimat-Verzeichnis des Anwenders befinden. Der Einsatz der dot-qmail-Dateien wird in den man-pages `dot-qmail`, `qmail-command` und `qmail-local` beschrieben.

Die dot-qmail-Dateien stellen ein "Kommando"-Environment zur konsekutiven Abarbeitung beliebiger weiterer Kommandos, Skripte und Auslieferungsanweisungen dar und sind somit (näherungsweise) vergleichbar mit Unix Profil-Dateien. Zunächst wollen wir uns jedoch an einem Beispiel ansehen, wie typischerweise dot-qmail-Dateien bei einem beliebigen Benutzer genutzt werden:

```
% su - erwin
% ls -la .qmail*
.qmail
.qmail-return_to_sender
.qmail-default
```

In diesem Beispiel sind drei grundsätzliche Typen von dot-qmail-Dateien genannt:

- `.qmail` — Diese Datei wird herangezogen, wenn der Empfänger genau `erwin@qmailer.fehnet.de` lautet.
- `.qmail-return_to_sender` — Wird der (VERP-)Empfänger `erwin-return_to_sender@qmailer.fehnet.de` spezifiziert, wird genau diese dot-qmail-Dateien ausgelesen, bzw. ausgeführt.
- `.qmail-default` — Bei allen anderen VERP-Empfängern, wie bzw. `erwin-XYZ@qmailer.fehnet.de` wird die Datei `.qmail-default` genutzt.

Für den Anwender stellt dies eine sehr elegante Möglichkeit dar, einlaufende

Nachrichten ausschliesslich durch die Angabe des VERP-Teils in der Empfängeradresse auszuwerten. Es seien jedoch zwei Einschränkungen genannt:

1. Die dot-qmail-Datei darf nicht ausführbar sein; ist sie es dennoch, werden hierin enthaltene Kommandos und Auslieferungsbefehle nicht ausgeführt.

2. Die VERP-Ergänzungen müssen immer in Kleinbuchstaben angegeben werden. Eine E-Mail mit der Empfängeradresse `Return_to_Sender-erwin@qmailer.fehnet.de` führt dazu, dass die dot-qmail-Datei `.qmail-default` angezogen wird. Es nutzt auch nichts, die dot-qmail-Datei in `erwin-Return_to_Sender@qmailer.fehnet.de` umzubenennen.

Der dot-qmail Mechanismus von Qmail stützt sich also auf explizite und implizite Regeln zur Analyse des Empfängernamens und der zugeordneten dot-qmail Datei. Ist der (lokale) Empfängername mit mehrfachen VERP-Erweiterungen aufgebaut, gilt das oben genannt Beispiel prinzipiell bis zur dritten VERP-Erweiterung, speziell z.B. bei der (nicht aufgeführten) dot-qmail Datei `.qmail-return_to_sender-default`.

qmail-local arbeitet den Inhalt der zuständigen dot-qmail Dateien schrittweise (zeilenweise) ab. Die Steuerung des Ablaufs geschieht hierbei über den Return- bzw. Exit-Code des letzten Kommandos. Folgende Exit-Codes werden erkannt.

Exit-Code	**Bedeutung**	**Aktion**
0	Verarbeitungsschritt war erfolgreich	Ausführung der nächsten Zeile
64, 65, 70, 76, 77, 78 und 112	Verarbeitungschritt war fehlerhaft (Hard Error)	Abarbeitung bricht ab
99	Verarbeitungsschritt war erfolgreich	Verarbeitung der dot-qmail Datei wird abgebrochen
110	Verarbeitungsschritt war fehlerhaft (permanent failure);	keine weitere Zustellung; **qmail-send** schickt eine Bounce-Nachricht an den Absender
111	Verarbeitungschritt war nicht erfolgreich (Soft Error; temporary failure)	Zustellung der E-Mail ist "deferred" (verzögert) und wird weiter versucht

Tabelle 6-20: Exit-Codes und ihre Wirkungsweise bei dot-qmail Dateien.

Innerhalb der dot-qmail Datei sind Kommentarzeilen erlaubt; diese beginnen wie üblich mit einem "#" in der ersten Spalte. Die Abarbeitung der dot-qmail Kommandos erfolgt per **qmail-local** immer in dem Environment des

zugehörigen Benutzers mittels des Aufrufs "`sh -c`" Liegt die Datei `~./qmail` als z.B. im Home-Verzeichnis `/home/erwin` des zugehörigen Benutzers *erwin*, kann nur darauf zugegriffen werden, worauf der User *erwin* auch Rechte hat; dies gilt insbeonderes für das Zustellungsverzeichnis bzw. die Mailbox. Ein Skript, das hier ausgeführt wird, in der Laufzeit aber z.B. auf `/var/log/` Schreibrechte verlangt, bricht u.U. unkontrolliert ab.

Beim Einsatz von Skripten (egal ob z.B. Shell oder PERL) zeigt das dot-qmail Kommando-Environment seine besondere Stärke, da nun auf alle in Tabelle 6-19 deklarierten **qmail-local** Umgebungsvariablen zurück gegriffen werden kann. Dies ist insbesonders bei der Verwendung des Virtual-Domain Konzepts von Qmail und des VERP-Mechanismus von enormer praktischer Bedeutung.

Bei der Abarbeitung der dot-qmail Datei kennt **qmail-local** drei unterschiedliche Fälle:

Die Weitervarbeitung findet über ein Programm (z.B. ein Skript) statt, dann der (ausführbaren) Datei ein Pipe-Zeichen vorgestellt:

```
|/usr/local/bin/qmvc
|preline /usr/ucb/vacation djb
|/var/qmail/bin/forward admin@qmailer.fehnet.de
```

Mittels der zusätzlichen Programms **preline** kann der Nachrichten-Header um weitere Angaben ergänzt werden.

Die E-Mail wird an andere Adressaten weiter geleitet (forwarding), wobei die ursprünglich hinzugefügte "Return-Path:" Information wieder entfernt wird:

```
# Das Ampersandzeichen kann i.d.R. auch weggelassen werden
erwin@localhost
&me@new.job.com
```

Die Zustellung erfolgt in ein (lokales) Mail-Verzeichnis oder eine Mailbox:

```
# Statt des vollen Pfadnamens kann auch das aktuelle
Verzeichniss
# ueber einen "." angegeben werden
#/home/erwin/Maildir
./Maildir
/var/spool/mail/erwin
```

Welche von diesen Schritten **qmail-local** bei der Zustellung von Nachrichten vorgenommen hat, kann aus dem **qmail-send** Logfile entnommen werden:

```
delivery 1397: success: did_1+0+2/
```

Hierbei steht der — nachfolgend dem "did" — die Anzahl der Zustellungen in

ein Maildir bzw. eine Mailbox (1); es folgt die Zahl der Forwards (0) sowie die Angabe über die zusätzlich ausgeführten Skripte bzw. Programme (2).

Beim Forwarding macht es im übrigen einen Unterschied über das Forward über das "&" oder mittels des Qmail-Befehls **forward** erfolgt.

Bei der Festlegung der Abarbeitungsschritte sollte die oben genannte Reihenfolge eingehalten werden; also erst Verarbeitung des bzw. der Skripte (z.B. meinen Virenscanner QMVC), dann Weiterleitung an andere E-Mail Accounts und anschliessend Zustellung ins eigene Mail-Verzeichnis.

Ist es notwendig, die dot-qmail Datei im laufenden Betrieb zu editieren, so kann dies zu unerwünschten Konsequenzen führen. Es ist angeraten über ein `chmod +t .qmail` zunächst die Datei "sticky" zu machen; dann wird diese nämlich von **qmail-local** nicht ausgewertet und die Zustellung findet erst nach Entfernen des Sticky-Bits wieder statt. Eine ähnliche Konsequenz hat das Execute-Bit. Ist dies gesetzt, z.B. `chmod +x .qmail-default`, darf die dot-qmail Datei nur Forwarding-Anweisungen enhalten. Andernfalls wird die Ausführung in Gänze abgelehnt und ein temporärer Zustellungsfehler generiert.

6.5.6.6 Qmail-User

Für die meisten Qmail-Anwender stellen die Qmail-User ein besonderes Mysterium dar, das nur geringe Beachtung findet und von daher auch nur wenig genutzt wird; wobei häufig der Einsatz schon mit den Fehlern beim Aufbau der Qmail-User-Datenbank endet.

Doch was sind Qmail-User? Kurz gesagt: Qmail besitzt eine eigene Benutzerdatenbank — vergleichbar der Datei `/etc/passwd` — zur Verknüpfung beliebiger E-Mail-Adressen mit System-Benutzern und zu ihrer kontrollierten lokalen Zustellung; was als Qmail-User bezeichnet wird. Betrachten wir die Abbildung 6-4, so erkennen wir, dass die Deklaration der Qmail-User Vorrang vor System-Benutzer besitzt. Der wesentliche Vorteil ist nun, dass hierdurch einerseits die Auslieferung von E-Mails (aufgrund ihrer Empfängeradresse) von den üblichen (per `/etc/passwd`) System-Benutzern entkoppelt wird, und andererseits die Weiterverarbeitung address-spezifisch (über den VERP-Mechanismus) gesteuert werden kann.

Die Qmail-User werden im Verzeichnis `/var/qmail/users/` eingerichtet, wozu die Datei `assign` genutzt wird. Der Aufbau der Datei `assign` wird in der man-page **qmail-users** beschrieben. Im Gegensatz zur Standard Unix-Datei `/etc/passwd` liegt die Qmail Benutzerdatenbank im Binärformat vor; muss also bei jeder Änderung bzw. Ergänzung neu kompiliert werden. Mittels der Qmail-User können folgende Aufgaben elegant erledigt werden:

- 1:1 Zuweisung von E-Mail Adresse und Benutzer-Account:

```
=local:user:uid:gid:homedir:dash:ext:
```

E-Mails für den Empfänger `local` (als Teil der Recipient-Adresse `local@host`) wird an den System-Account *user* mit der UID `uid` und der GID `gid` zugestellt, wobei dieser das Home-Directory `home` besitzt. Die in diesem Verzeichnis liegende dot-qmail Datei `~/.qmail`dashext wird ausgeführt.

- n:1 Zuweisung von E-Mail-Adressen und Benutzer-Account (Wildcard-Zuweisung):

```
+loc:user:uid:gid:homedir:dash:pre:
```

E-Mails für den Empfänger `loc-ext` (als Teil der Recipient-Adresse `loc-ext@host`) wird an den System-Account *user* mit der UID `uid` und der GID `gid` zugestellt, wobei dieser das Home-Directory `home` besitzt. Die in diesem Verzeichnis liegende dot-qmail Datei `~/.qmail`dashpreext wird ausgeführt.

Der Unterschied zwischen der spezifischen und der Wildcard-Zuweisung liegt also in der Verwendung des Zeichens "=" bzw. "+". Die Syntax der Zuweisung verlangt, dass alle Felder angegeben werden (evtl. mit einem Null-String "") und dass das letzte Zeichen in der Zeile ein ":" ist.

Der Einsatz der Qmail-User gibt uns somit die einmalige Möglichkeit, das Delimitor-Zeichen (üblicherweise das "-") zur Separation der VERP-Ergänzung zu steuern. Normalerweise wird dies einmalig bei der Konfiguration von Qmail festgelegt. Dan Bernstein hat aber **qmail-local** so flexibel ausgelegt, dass das Delimitor-Zeichen beim Aufruf als Argument übergeben werden kann.

Wir betrachten folgendes Beispiel (etwas abgewandelt aus der man-page **qmail-users**):

```
+:alias:7790:2108:/var/qmail/alias:-::
+joe_:joe:507:100:/home/joe:_::
=joe:joe:507:100:/home/joe:::
.
```

Bei der Abarbeitung der Qmail-User geht **qmail-lspawn** immer vom speziellsten zum allgemeinsten Fall vor; unabhängig von der Anordnung der User in der Datei `assign` selbst, da diese kompiliert wird. Zum Aufbau einer neuen Qmail-User Datenbank dient das Programm **qmail-newu**. Fehlt in der Datei `assign` der Punkt "." in der letzten Zeile, weigert sich **qmail-newu** die Datenbank zu generieren. **qmail-newu** liest beim Aufruf die Datei `/var/qmail/users/assign` ein und erzeugt die Binärdatei `/var/qmail/users/cdb`, deren Inhalt von **qmail-lspawn** ausgelesen und deren Werte als Parameter an **qmail-local** übergeben wird.

Doch wie interpretiert **qmail-lspawn** das obige Beispiel für unseren Zielrechner

`host.domain.com`? Der speziellste Fall ist, dass E-Mail an `joe@host.domain.com` dem Account *joe* zugestellt wird (dritte Zeile: "`=joe:`"), wobei dann natürlich die Weiterverarbeitung über die dot-qmail Datei `~/.qmail` erfolgt. Mittels der zweiten Zeile ("`+joe_:`") werden E-Mails an die Adresse `joe_direct@host.domain.com` ebenfalls an *joe* ausgeliefert, wobei aber nun die Weiterverarbeitung entweder, z.B. über die dot-qmail Datei `~/.qmail_direct` oder (allgemein) über `~/.qmail_default` erfolgt. Wir beachten, dass hier der Default-Delimitor "`-`" durch das "_" ersetzt wurde. Die allgemeinste Zustellung findet über die erste Zeile statt. Jede andere E-Mail wird nun dem Qmail *alias* User zugestellt, was mittels der ersten Zeile und der Anweisung "`+:`" realisiert wird.

Vorteile bringt der Qmail-User Mechanismus nicht nur hinsichtlich seiner Flexibilität, sondern auch, wenn auf dem System viele lokale Benutzer als E-Mail Empfänger fungieren. Statt über den Qmail Befehl **qmail-getpw** die Datei `/etc/passwd` auszulesen, erfolgt der Lookup nun gegenüber der Binärdatei `/var/qmail/users/cdb` und verläuft somit wesentlich schneller und sicherer (siehe auch man-page **qmail-getpw**).

qmail-pw2u

Zur Unterstützung dieses Verfahrens hat uns Dan Bernstein das **Programm qmail-pw2u** spendiert, dass ein Standard V7-Format `/etc/passwd` Datei in eine Qmail-User `assign` Datei umwandelt. Da es in der Regel nicht erwünscht ist, alle in der Datei `/etc/passwd` eingetragenen Accounts auch via E-Mail zu erreichen, lässt sich der Aufbau der `assign`-Datei über Regeln steuern, die in verschiedenen Dateien hinterlegt werden:

- `/var/qmail/users/include` — Liste aller Benutzernamen, für die ein Assignment vorgenommen werden soll; ist ein Benutzer hier nicht enthalten, wird er ignoriert.
- `/var/qmail/users/include` — Liste aller Benutzer, die auf keinen Fall zu berücksichtigen sind; hierzu könnten z.B. die User *ftp* und *bind* zählen.
- `/var/qmail/users/mailnames` — Alias-Liste von Namen für einen Benutzer. Hierdurch kann z.B. E-Mail an *ftp*, *bind* oder *apache* an den User *admin* umgeleitet werden: `admin:ftp:bind:apache:`.
- `/var/qmail/users/subusers` — Liegt diese Datei im Format `sub:user:pre:` vor, dann wird eine E-Mail an `sub@host.domain.com` an User *user* ausgeliefert, der die Weiterverarbeitung über die dot-qmail Datei `~/.qmail-pre` steuert. Bei `sub` kann es sich um eine VERP-Adresse handeln, sodass bei einer E-Mail z.B. für `sub-ext@host.domain.com` die weitere Verarbeitung per `~/.qmail-pre-ext` erfolgt.

qmail-pw2u bietet noch einige weitere Möglichkeiten, zu deren Nutzung ich

allerdings auf die man-page **qmai-pw2u** verweisen möchte.

6.5.6.7 Virtual Domains

Bei der Konstruktion der Virtual Domains für Qmail hat Dan Bernstein nutzbringend das einsetzten können, was wir bereits als VERP kennengelernt haben; nur hat es es genau umgedreht. Doch zunächst wollen wir klären, was unter "Virtual Domains" zu verstehen ist. Hierzu betrachten wir Abbildung 6-4:

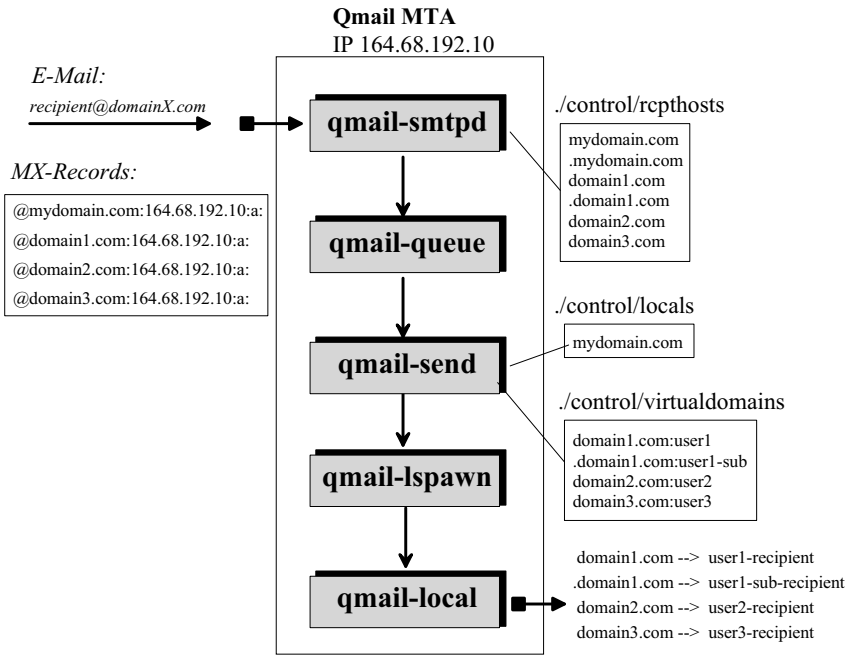

Abbildung 6-4: Ablauf der Zustellung und Verarbeitung von E-Mails für eine "virtuelle Domain". Hierbei empfängt ein MTA die E-Mails für unterschiedliche Domains. Unter Qmail werden E-Mails für diese virtuellen Domains dedizierten System-Benutzern zugestellt. Neben den virtuellen Domains `domain1.com`, `domain2.com` und `domain3.com` hostet der Qmail MTA auch noch eine eigene Domain `mydomain.com`.

Unter "Virtual Domains" oder "Virtual Domain Hosting" wird allgemein ein Verfahren beschrieben, dass E-Mails für unterschiedliche Domains zu einem einzigen MTA zugestellt werden, der diese dann weiter zu verarbeiten hat.

Zunächst sind die allgemeinen Voraussetzungen zu schaffen, d.h. DNS MX-Records für die virtuellen Domains müssen alle auf den Ziel-MTA weisen. Dies

ist in Abbildung 6-4 unter Anlehnung an das **tinydns** Datenformat gezeigt, wobei der MTA nur eine einzige IP-Adresse im DNS aufweist, was aber hier nur zur vereinfachten Darstellung dient. Faktisch können natürlich mehrere IP-Adressen pro Interface gebunden werden, oder aber der Rechner verfügt über mehrere Netzwerkkarten bzw. Internet-Zugänge (von z.B. für unterschiedliche Provider).

Als nächstes müssen natürlich alle Empfänger-Domains in der Datei rcpthosts bzw. morecpthosts eingetragen sein. Will man dafür Sorge tragen, dass auch alle untergeordneten Domains erfasst werden, sorgt für die Domain domain1.com der zusätzliche (Wildcard-) Eintrag `.domain1.com` dafür, dass auch alle Subdomains bzw. untergeordneten Hostadressen mit erfasst werden.

Der entscheidende Schritt zur Auslieferung der E-Mails für die virtuellen Domains ist die Datei `./control/virtualdomiains`, die von **qmail-send** ausgelesen und interpretiert wird (und zwar nur zur Startzeit von Qmail bzw. nach einem HUP-Signal). Hierin wird die Zustellung der E-Mails an System-Accounts festgelegt. Wir erinnern uns kurz zurück an das VERP-Verfahren: Der lokale Teil einer E-Mail Adresse wurde am Ende — nach einem Delimitor-Zeichen — um eine zusätzliche Information ergänzt (Extension). Bei der Verarbeitung von virtuellen Domains ist es genau andersherum: Der Name des per `./virtualdomains` zugewiesenen System-Accounts (vgl. Abbildung 6-3) wird der Recipient-Adresse der einlaufenden E-Mail *vorgestellt* (prepend). Hierdurch kann der VERP-Mechanismus unverändert für Empfänger in virtuellen Domains eingesetzt werden.

Anwendungsbeispiel

Wir ergänzen das Beispiel für die Datei `./virtualdomains` aus Abbildung 6-3 um einen weiteren Eintrag und gehen die Möglichkeiten systematisch durch:

```
domain1.com:user1
.domain1.com:user1-sub
domain2.com:user2
domain3.com:user3
:user4
```

Listing 6-15: Aufbau der Datei `./virtualdomains`.

E-Mails an z.B. `recipient1@domain1.com` werden dem lokalen Empfänger *user1* zugestellt. Genauso verhält es sich mit E-Mails für `recipient2@domain2.com` und `recipient3@domain3.com`, die entsprechend an *user2* und *user3* ausgeliefert werden. Doch was machen wir den dort einlaufenden E-Mails? Hier greift nun die zugewiesene dot-qmail Datei `~/.qmail` bzw. `~/.qmail-default`. Im einfachsten Fall steht hierin ein Forwarding-Befehl mit der Angabe des Zielrechners, z.B. ein Lotus Notes Gateway in der Empfänger-Domain:

```
/home/user1/.qmail:
|/var/qmail/bin/forward ${EXT}@gw.domain1.com
```

bzw.

```
/home/user2/.qmail-default:
|/var/qmail/bin/forward ${EXT}@[IP-Adresse]
```

Für die Domain `domain1.com` existiert noch eine "Wildcard"-Zuweisung für E-Mails an z.B. `recipient4@msexchange.domain1.com`. Dieser Fall wird in der zweiten Zeile der Datei `./virtualdomains` abgewickelt, wo alle E-Mails an `*@*.domain1.com` die Qmail-Extension `sub` erhalten, also

```
recipient4@msexchange.domain1.com -> user1-sub-recipient4
```

und unsere dot-qmail-Datei kann dann wie folgt aussehen:

```
/home/user1/.qmail-sub:
|/var/qmail/bin/forward ${EXT2}@msexchange.domain1.com
```

Mail-Loops

Hierbei kann es sehr leicht vorkommen, einen wunderschönen Mail-Loop zu erzeugen, wenn die Forwarding-Host-Adresse per MX-Record auf unseren eigenen MTA zurück weist. Zum Glück erkennt und unterbindet Qmail solche Mail-Loops. Erfahrene Qmail Administratoren setzen zur Kontrolle der Weiterleitung der E-Mail die Möglichkeit von **qmail-remote** ein, sog. SMTP-Routing zu deklarieren. Hierzu wird die Datei `./smtproutes` herangezogen, in der z.B. folgendes stehen könnte:

```
gw.domain1.com:[IP-Adresse_1]
msexchange.domain1.com:[IP-Adresse_2]
```

Wir "verdrahten" also den Domain- bzw. Hostname mit einer festen IP-Adresse und forwarden letztlich zu dieser Adresse, vergleichbar dem Beispiel für `domain2.com`. Allerdings entfallen dann alle Backup-Möglichkeiten für die E-Mail Zustellung entsprechend den MX-Records.

Zum Abschluss wollen wir die vierte Zeile in Listing 6-15 betrachten. Diese beschreibt ebenfalls eine "Wildcard"-Zuweisung. E-Mails für Domains, die wir eigentlich nicht mehr supporten, werden über den lokalen Account *user4* abgewickelt. Dies kann z.B. dann der Fall sein, wenn ein Kunde eine Domain bei uns gekündigt hat, der MX-Record bereits umgestellt wurde, aber manche Anwender noch zu der alten IP-Adresse E-Mails verschicken. Dann macht es Sinn, eine dot-qmail Datei wie folgt parat zu haben:

```
/home/user4/.qmail-default:
| /var/qmail/bin/bouncesaying 'Sorry, we don't support this
domain anymore'
| > /dev/null
```

Wir beachten, dass sich diese Wildcard-Zuweisung nicht mit unserer lokalen Domain `mydomain.com` "beisst", da der Eintrag in `./locals` Vorrang geniesst und dass natürlich noch der alte Eintrag für diese Domain in der Datei `./rcpthosts` bestehen muss. Will man mehr Komfort beim Einrichten und Verwalten virtueller Domains haben, greift man allerdings auf die Qmail Ergänzung Vmailmgr oder Vpopmail zurück, die im nächsten Kapitel besprochen werden.

6.5.7 Remote Delivery

qmail-remote ist der SMTP-Client von Qmail. Er wird immer dann aufgerufen, wenn die E-Mail nicht lokal zuzustellen ist. Der SMTP-Server **qmail-smtpd** und **qmail-remote** sind ungleiche Brüder. Während **qmail-smtpd** sich auf die TCP-Hilfsprogramme **tcp-env** bzw. **tcpserver** verlässt, bedient sich **qmail-remote** der Socket-Schnittstelle und nimmt einen DNS-Lookup des Peer-Partners vor. Im Gegensatz hierzu, setzt bei Qmail **qmail-remote** auf **qmail-rspawn** auf, über das es in der Regel aufgerufen wird. **qmail-remote** kennt bzw. nutzt auch keine Environment-Variablen, sondern stützt sich lediglich auf Aufrufparameter und einige wenige Kontrolldateien.

6.5.7.1 Aufruf und Arbeitsweise von qmail-remote

```
qmail-remote HOST SENDER RECIPIENT1 RECIPIENT2 ...
```

qmail-remote erwartet die E-Mail auf Filedescriptor 1 (Std-Input). Aufgabe von **qmail-rspawn** ist es, Nachrichten aus der Qmail-Queue zu entnehem, **qmail-remote** mit den notwendigen Informationen zu versorgen und bei erfolgter Übertragung die E-Mail aus der Queue zu löschen.

Per RFC 822 wird das Ende einer E-Mail wird mit einem "." und einem CR/LF in der letzten Zeile gekennzeichnet. Da normalerwise unter Unix das LF das Zeilen-Ende Zeichen ist, ergänzt **qmail-remote** dies jeweils um das CR Zeichen. Weitere Änderungen des einlaufenden Datenstroms nimmt **qmail-remote** nicht vor.

Anschliessend wird die Variable `HOST` ausgewertet, wobei hier entweder ein FQDN steht; oder aber die Angabe einer IP-Adresse in Klammern `[IP-Adresse]`. Zunächst wird überprüft, ob für `HOST` eine IP-Adresse sowie ein gültiger MX-Eintrag im DNS vorliegt

qmail-remote baut im folgenden für `HOST` eine SMTP-Sitzung auf und versendet die E-Mail mit der Angabe von `MAIL FROM: <SENDER>` an `RCPT TO: <RECIPIENT1>` an `HOST` usw.. **qmail-remote** berichtet über den Erfolg seiner Aktionen, die jeweils mit einem charakteristischen Buchstaben beginnen:

- `r` die erfolgreiche Annahme der E-Mail, mit
- `h` die permanente Ablehnung des Absenders und mit
- `s` eine temporäres Ablehnung des Absenders, sowie mit
- `K` eine erfolgreiche Zustellung bzw. mit
- `Z` ein temporärer Fehler bei der Zustellung und letztlich mit
- `D` eine permanente Fehlersituation.

Anschliessend wird im Klartext auf die Aktion bzw. Fehlersituation hingewiesen die weitgehend selbstsprechend sind. Der `Fehler "System resources temporarily unavailable"` bedeutet in der Regel, dass keine Sockets mehr erzeugt werden konnten. Findet sich der Hinweis: `"Although I'm listed as a best-preference MX or A for that host, it isn't in my control/locals file, so I don't treat it as local"` so weist dies auf eine Fehlkonfiguration des E-Mail-Systems hin, bei der zwar der MX-Record auf den eigenen Rechner verweist, die entsprechende Qmail Konfigurationsdateien `locals` oder `virtualdomains` aber nicht angepasst wurden.

qmail-remote **Fehlermeldungen**
`Out of memory. (#4.3.0)`
`System resources temporarily unavailable. (#4.3.0)`
`Sorry, I wasn't able to establish an SMTP connection. (#4.4.1)`
`Unable to read message. (#4.3.0)`
`CNAME lookup failed temporarily. (#4.4.3)`
`Sorry, I couldn't find any host by that name. (#4.1.2)`
`Unable to switch to home directory. (#4.3.0)`
`Unable to read control files. (#4.3.0)`
`SMTP cannot transfer messages with partial final lines. (#5.6.2)`
`I (qmail-remote) was invoked improperly. (#5.3.5)`
`Sorry, I couldn't find any host named REMOTEHOST (#5.1.2)`
`Sorry, I couldn't find a mail exchanger or IP address. (#5.4.4)`
`Sorry. Although I'm listed as a best-preference MX or`

```
A for that host, it isn't in my control/locals file,
so I don't treat it as local. (#5.4.6)
```

*Tabelle 6-20: Fehlermeldungen von **qmail-remote**.*

qmail-remote nutzt in der Regel die Angabe des (FQDN) Hostname in der Datei `me` um sich per HELO Befehl dem SMTP-Partner vorzustellen. Will man dies nicht — weil es sich um einen internen Namen handelt, oder weil sich z.B. alle Backup-MTAs gleich melden sollen — kann statt dessen die Konfigurationsdatei `helohost` entsprechend befüllt werden.

Wenig spannend sind auch die Konfigurationsdateien `timeoutconnect` und `timeouteremote`, die **qmail-remote** mitteilen, wie lange es auf eine Verbindung mit dem SMTP-Server warten soll, bzw. welche maximale Zeit bis zur Empfang der erwarteten SMTP-Response verstreichen darf. Die Defaultparameter in **qmail-remote** von 60 bzw. 1200 Sekunden brauchen in der Regel keine Anpassung.

Eindeutig wichtigste Konfigurationsdatei ist `smtproutes`. Hierin wird in Abweichung vom normalen DNS-MX-Lookup festgelegt, zu welchem Rechner sich **qmail-remote** direkt beim Aufruf mit dem Parameter `HOST` verbinden soll, ohne auch nur eine DNS-CNAME-Abfrage vorzunehmen. `smtproutes` besteht aus einer Liste von Zielrechnern (Host), zugeordneten Relay- bzw. Empfangs-MTAs und einer Portnummer, wobei die Angaben durch Doppelpunkte getrennt sind. Wir betrachten folgendes Beispiel:

```
:mail.myisp.com
taffline.de:mailer.taffline.de
.taffline.de:mailer.taffline.de
mydomain.com:[192.168.1.15]:26
```

Listing 6-16: Einsatz der Konfigurationsdatei `smtproutes`.

Die erste Zeile sagt aus, dass in der Regel alle auslaufenden E-Mail über den MTA meines ISPs versendet werden. Hierzu nutzen wir die "Wildcard"-Eigenschaft von `smtproutes`, indem der erste Wert vor dem Doppelpunkt leer bleibt. E-Mails an die Domains `*.taffline.de` und `taffline.de` selbst, werden hingegen ohne Umwege an den MTA `mailer.taffline.de` übermittelt, wobei Schliesslich soll jede für meine Domain ausgehende E-Mail über einen MTA mit der IP-Adresse `192.168.1.15` übermittelt werden, wobei dieser aus Sicheheitsgründen nicht das SMTP-Port 25 sondern 26 nutzt.

6.5.7.2 Single Recipient

Bei Qmail erfolgt der Aufruf von **qmail-remote** immer über **qmail-rspawn**,

wobei in der Tat **qmail-remote** auch "stand alone" genutzt werden könnte. **qmail-remote** liest die einzelnen Nachrichten aus der Queue und übergibt diese für jeden Empfänger einzeln an **qmail-remote**. Das heisst, **qmail-rspawn** sortiert die Nachrichten nicht nach dem Kriterium HOST, sondern ruft **qmail-remote** immer nur mit einem einzigen Empfänger auf, ohne Rücksicht darauf zu nehmen, ob eine Nachricht mit gleichem Body oder gar mehrere Nachrichten mit unterschiedlichem Inhalt an den gleichen HOST zu übermitteln sind.

Dieses Verhalten wird als *Single Recipient* bezeichnet und ist häufig als Kritikpunkt für Qmail hinsichtlich der Performance aufgeführt. Immerhin muss nun für jede einzelne Nachricht ein neuer TCP-Puffer allokiert werden, eine neue SMTP-Verbindung aufgebaut werden und die Nachricht (mit u.U. identischem Body) jeweils einzeln übertragen werden.

An dieser Stelle hat Dan Bernstein Qmail nicht hinsichtlich Performance sondern hinsichtlich Sicherheit optimiert. Innerhalb der Queue ist jede Nachricht durch ihre Inode-Kennung sowie — wie wir weiter unten sehen werden — ihren Status in der Bearbeitung eindeutig identifiziert. Lehnt z.B. bei einer Massenübertragung von *N* E-Mails der SMTP-Empfänger die Nachricht *n* ab, weil z.B. Datei-Quota überschritten wurden, hiesse das, dass die ersten *n-1* Nachrichten aus der Queue gelöscht werden könnten, die *n*-te Nachricht zurückzustellen ist (und evtl. in der gleichen Sitzung nochmals zu übertragen ist) und mit den Nachrichten *n+1* bis *N* fortgefahren werden kann. Dieses Szenario kann beliebig komplex gemacht werden und würde den Auslieferungsprozess (hier **qmail-remote**) mit einer enormen Buchhaltung beschweren. Den einzigen Schutz bietet das KISS-Prinzip: Keep It Simple Stupid! und die Schlussfolgerung, diese Aufgabe an die untergeordneten Instanzen zu delegieren, die deren Bewältigung sehr viel einfacher und effizienter realisiert werden kann.

6.5.8 Quick Mail Protocol

Neben den Protokollen SMTP und POP3 unterstützt Qmail auch das von Dan Bernstein erfundene Qmail Mail Protocol QMP (siehe `http://cr.yp.to/proto/qmqp.html` sowie `http://cr.yp.to/proto/qmtp.txt`). Im Gegensatz zum SMTP, das eine MTA-zu-MTA-Kommunikation beschreibt, ist das QMP eine Client/Server-Anwendung zwischen benachbarte Message-Stores (MS). Das Quick Mail Transport Protokoll (QMTP) setzt selbstverständlich auf TCP auf, wobei in der Regel der Port 628 genutzt wird (im FAQ zu Qmail ist fälschlicherweise der Port 209 angegeben).

Qmail bringt drei Komponenten des QMP zum Einsatz: den Qmail Mail Protocol Client (**qmail-qmpc**), den Quick Mail Protocol Server (**qmail-qmpd**) und den Quick Mail Protocol Transport Daemon (**qmail-qmtpd**), die beide in der Regel über **tcpserver** gestartet werden.

Das Quick Mail Protokoll wird vor allem dann eingesetzt, wenn es darum geht, E-Mails zwischen einzelnen und "vertrauenswürdigen" Rechnern (z.B. in einem Cluster) auszutauschen. Im Rahmen eines sog. "Mini-Qmail" kann via QMQP ein Rechner als MTA konfiguriert werden, der die zentrale Nachrichten-Queue beheimatet und die anderen Rechner — mittels **qmail-qmqpc** — quasi nur noch als Nachrichten-Lieferanten fungieren. Für den Austausch der Nachrichten im Cluster kann dann nämlich auf SMTP komplett verzichtet werden, was den Durchsatz wesentlich steigert.

6.5.8.1 Nachrichtenformat

Vor dem Versand mittels **qmail-qmqpc** wird die Nachricht zunächst in einen Text-String im Format "firstline\012\secondline\012...\012\lastline" umgewandelt. Hierbei ist "`\012`" der oktale Wert des Linefeeds LF (Hexadezimal `x"0a"`). Die Nachricht selbst darf hierbei diese Sequenz nicht enthalten, was Dan Bernstein als "save-8-bit-text message" bezeichnet. Dieser Text-String wird anschliessend in einen sog. "netstring" umgewandelt. Der netstring beinhaltet als erste Angabe die Länge des ursprünglichen Text-Strings und schliesst mit einem Komma ab: `[len]":"[string]","`. Aus dem Text-String "`hello world!`" wird somit `x"31 32 3a 68 65 6c 6c 6f 20 77 6f 72 6c 64 21 2c"` in hexadezimaler Notation oder einfach "`12:hello world!,`" in ASCII.

6.5.8.2 Protokollelemente

Das QMTP kennt nur wenige Protokollelemente. Der Client sendet die Nachricht, die Absenderadresse und die Empfängeradresse(n) als netstring zum QMTP Server. Der Server wartet mit seiner Antwort auf das abschliessende Byte der Übertragung, d.h. die letzte Empfängeradresse.

Die Antwort des Server besteht in einer Folge von 8bit Bytes (in der netstring Notation), wobei das erste Byte folgende Informationen beinhaltet (vgl. Abschnitt 6.5.6.1):

- K — Die Nachricht wurde zur Zustellung an die Empfänger akzeptiert.
- Z — Temporäre Ablehnung; die Übertragung soll weiter versucht werden.
- D — Permanente Ablehnung.

Die übrigen netstring-Bytes beinhalten eine mögliche Erklärung in englischer Sprache und sind in UTF-8 kodiert.

6.5.8.3 QMQPC

Das Programm **qmail-qmqpc** ist der Client des Quick Mail Protocols. Nachrichten, die via **qmail-qmqpc** übertragen werden, landen nicht in der lokalen Queue, sondern werden auf die QMP-Server übertragen. Im Qmail-

Verbund ist daher das Programm **qmail-queue** durch **qmail-qmqpc** zu ersetzen:

```
# cd /var/qmail/bin
# mv qmail-queue qmail-queue.org
# ln -s qmail-qmqpc qmail-queue
```

qmail-qmqpc muss natürlich wissen, auf welchen Rechnern der entfernte Message Store, respektive die Qmail-Queue liegt. Hierzu bedient sich **qmail-qmqpc** der Konfigurationsdatei `qmqpservers`, in denen die IP-Adresse der QMPQ-Server hinterlegt ist, die im Round-Robin-Verfahren angefragt werden, bis ein QMQP-Server sich für den Empfang der Nachrichten verantwortlich fühlt.

6.5.8.4 QMPD

Zum Empfang der E-Mails über das Quick Mail Protokoll dient das Programm **qmail-qmqpd**. Die einlaufenden Nachrichten werden einfach in die Ausgangs-Queue gestellt. Im Gegensatz zum Programm **qmail-qmtpd** besitzt es geringere Konfigurationsmöglichkeiten, indem es sich ausschliesslich auf die TCP-Environment-Variablen stützt.

6.5.8.5 QMTPD

qmail-qmptd ist der Server des Quick Mail Transfer Protocol (QMTP). Seine Aufgabe ist der vergleichbar dem des **qmail-smtpd**; nur besitzt es eine wesentlich geringere Komplexität, da es (naturgemäss) keine Rücksicht auf schlecht konfigurierte Clients nehmen muss.

qmail-qmptd verlässt sich auf die Informationen, die ihm über die **tcp-env** bzw. **tcpserver** bereit gestellt werden. Zusätzlich (und im Vergleich zu **qmail-qmqpd**) wird hierbei jedoch der Domain-Name des QMP Clients gegen den Inhalt der Dateien `rcpthosts` bzw. `morercpthosts` geprüft. Weiterhin werden die Umgebungsvariablen `$RELAYCLIENTS` und `$DATABYTES` bzw. die zugehörige Konfigurationsdatei ausgewertet.

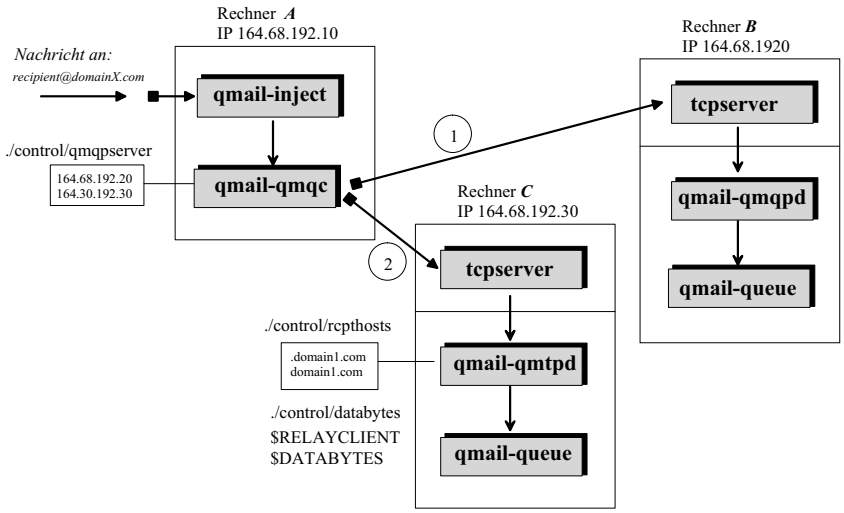

*Abbildung 6-5: Aufbau eines Quick Mail Verbundsystems. Wir beachten, dass sich der Prozess **qmail-qmtpd** zusätzlich auf die Konfigurationsdateien `rcpthosts` sowie `databytes` stützt und zudem die wichtige Environment-Variable `$RELAYCLIENT` auswertet.*

In Abbildung 6-5 ist das Zusammenspiel der Quick Mail Komponenten beispielhaft dargestellt. Der Rechner ***A*** mit der IP 168.68.192.10 fungiert als Quick Mail Client, der die beiden Server ***B*** und ***C*** bedient. Auf den beiden Server lauscht zum einen der Prozess **qmail-qmqpd** und zum anderen **qmail-qmtpd** jeweils über den **tcpserver**. Für den letzten Fall würde ein Daemontools `run`-Skript zusammen mit der notwendigen `cdb` folgenden beispielhaften Aufbau besitzen:

```
#!/bin/sh
QMAILDUID=‘id -u qmaild‘
QMAILDGID=‘id -g qmaild‘
HOSTNAME=‘hostname‘
QMQPPORT=‘628‘
exec tcpserver -v -R -l $HOSTNAME -x qmqpclients.cdb \
        -u $QMAILDUID -g $QMAILDGID 0 $QMQPPORT \
       /var/qmail/bin/qmail-qmtpd 2>&1
qmqpclients:
164.68.192.10:allow,RELAYCLIENT=""
164.68.192.20:allow,RELAYCLIENT=""
```

```
164.68.192.30:allow,RELAYCLIENT=""
:deny
```

*Listing 6-17: Ein **qmail-qmtpd** run-Skript einschliesslich der einschränkenden Konfigurationsdatei `qmqpclients`, die in natürlich in das cdb-Binärformat überführt werden muss.*

6.5.9 POP3-Server

Das Post Office Protocol (POP) Version 3 dienst zum Zugriff auf die lokalen Postfächer von entfernten Clients aus, den Remote User Agents. Hierzu zählen kmail, Outlook, Eudora, Pegasus Mail — und wie sie alle heissen mögen. Der POP3-Server von Qmail wurde massgeblich von Russ Nelson mit beinflusst, wie der man-page zu **qmail-pop3d** entnommen werden kann, und ist modular konzipiert:

- **qmail-popup** zum Aufbau der POP3-Sitzung unter Angabe des POP3-Servers (AUTHORIZATION Status),
- **checkpassword** zur Verfikation des Benutzer-Kennworts und schliesslich
- **qmail-pop3d** zum Zugriff auf die in einem Maildir organisierten E-Mails (TRANSACTION Status)..

Das Programm **checkpassword** muss separat besorgt und installiert werden (Abschnitt 6.2.3). Der Qmail POP3-Server unterstützt ausschliesslich Dan Bernstein's Maildir-Format. Typischerweise wird auch der POP3-Server unter Verwendung der Daemontools eingerichtet, was hier noch einmal in einem `run`-Skript kurz dargestellt werden soll:

```
#!/bin/sh
HOSTNAME=‘hostname‘
#HOSTIP=‘hostname -i‘   # not supported on all Unix systems
QMAILDUID=‘id -u qmaild‘
QMAILDGID=‘id -g qmaild‘
QMAILLUID=‘id -u qmaill‘
exec tcpserver -v -H -R -c100 0 pop3 \
       /var/qmail/bin/qmail-popup \
        $HOSTNAME /bin/checkpassword \
       /var/qmail/bin/qmail-pop3d Maildir 2>&1
```

Listing 6-18: Der POP3-Server unter Daemontools-Kontrolle; die Option `-c100` lässt maximal 100 gleichzeitige POP3-Sitzungen zu.

6.5.9.1 qmail-popup

Das POP3-Login-Programm **qmail-popup** kennt nur die Befehle "USER", "PASS", "APOP", "QUIT" sowie "NOOP". Zur Übertragung des Passworts kommt also das Kommando "PASS", bzw. das sicherere APOP zum Einsatz. Zunächst liest **qmail-popup** die Benutzerinformation (via **tcpserver**) von FD 0 und schreibt auf FD 1. Diese Datei-Deskriptoren bleiben zur Durchreichung der Kommunikation zwischen **tcpserver** und dem aufgerufenen Programm (**qmail-pop3d**) bestehen. Anschliessend wird in der Regel das Programm **checkpassword** gestartet und über den FD 3 die Benutzerinformation (Name und Passwort, sowie ggf. ein APOP-Zeitstempel) übertragen. Über den FD 2 kommuniziert **qmail-popup** mit dem Netzwerk, d.h. **tcpserver**. Nach spätestens 20 Minuten schliesst qmail-popup die TCP-Verbindung (auf Port 110), wenn keine Informationen übertragen werden.

6.5.9.2 qmail-pop3d

Das Programm **qmail-pop3d** ist das eigentliche "Arbeitspferd", da durch dieses die E-Mails, die im Maildir-Format vorliegen müssen, bearbeitet werden können. Die möglichen Kommandos hierzu lauten "STAT", "LIST", "UIDL", "DELE", "RETR", "LAST", "TOP", "RSET" sowie "NOOP".

Fehlermeldungen werden bei POP3 durch den String "-ERR" eingeleitet und weisen bei **qmail-pop3d** folgende Ergänzungen auf:

qmail-qop3d **Fehlermeldungen**
`out of memory`
`this user has no $HOME/Maildir`
`unable to scan $HOME/Maildir`
`syntax error`
`unimplemented`
`already deleted`
`messages are counted from 1`
`not that many messages`
`unable to open that message`
`unable to unlink all deleted messages`

*Tabelle 6-21: Mögliche **qmail-pop3d** Fehlermeldungen.*

Wie auch **qmail-popup**, schliesst **qmail-pop3d** die Verbindung, falls diese über 20 Minuten idle war. An jede Nachricht fügt **qmail-pop3d** bei der Übertragung eine leere Zeile hinzu. Zusätzlich ist zu erwähnen, dass **qmail-pop3d** keine ergänzende Log-Information zur Verfügung stellt. Lediglich die mit mittels **tcpserver -v** verfügbaren TCP/IP-Verbindungsinformationen können somit Einzug in den Log finden.

6.5.9.3 checkpassword

Dan Bernstein's **checkpassword** Programm nutzt zum Lesen der Unix-Passwörter die Systemroutinen **getpwnam** und **crypt**. Accounts, bei denen kein Kennwort eingerichtet werden, werden von **checkpassword** abgelehnt, was durch ein Return-Code von 1 kundgetan wird; bei temporären Fehlern beendet sich **checkpassword** mit 111.

Der allgemeine Aufruf von checkpassword ist:

```
checkpassword subprogram [ args ...  ]
```

Wie aus Listing 6-18 hervorgeht wird von **checkpassword** (bei erfolgreicher Authentisierung) die Environment-Variablen `$USER`, `$HOME` und `$SHELL` besetzt und anschliessend das Unterprogramm (hier: **qmail-pop3d**) aufgerufen, wobei sich **checkpassword** der Routine **execvp** bedient. Da es ausnahmslos den Deskriptor 3 zur Übertragung der Authentisierungsinformation nutzt, bleibt die normale Kommunikation zwischen POP3-Server und Netzwerk über die Datei-Deskriptoren 0 und 1 unbeeinflusst.

Alternative checkpassword Implementierungen

Es ist zu bemerken, dass die von **checkpassword** durchgeführte Authentisierung mittlerweile durch viele andere ergänzt worden ist. Hinweise hierzu finden sich auf der Qmail Home-Page unter einem eigenen Anker `http://qmail.org/top.html#checkpassword`. Die Authentisierung kann gegenüber einer Datenbank (Oracle, mySQL, Postgress etc.), über einen LDAP-Lookup, selbst gegenüber einem Windows-NT Domänen-Controller vorgenommen werden, wobei die Verschlüsselungsverfahren mannigfaltig sind.

6.6 Qmail-Queue

Die Qmail-Queue ist der zentrale "building block" von Qmail. Sie beinhaltet die eigentliche Nachrichten und eine (Meta-)Information über den Zustand bzw. den Verarbeitungsstatus der Nachricht. mittles einer separat abgelegten Zustellungsinformation (Sender- und Empfänger) in Filesystem der Queue. Dan Bernstein hat den Aufbau der Queue in seinem Dokument `INTERNALS` beschrieben, von dem hier auch reichlich Gebrauch gemacht werden soll. Zugriff auf die Queue haben nur zwei Programme

- **qmail-send** und
- **qmail-queue**

deren Zusammenarbeit nur näher beschrieben werden soll.

6.6.1 qmail-queue

Es werden unterschiedliche Nachrichten in die Queue eingestellt:

- Nachrichten von lokalen Unix-Benutzern via **qmail-inject** (oder **mailsubj**),
- Nachrichten von entfernten MTAs über **qmail-smtpd**,
- weitergeleitete Nachrichten über **qmail-local**, die Mittels des **forward** oder **fastforward** Mechanismus eingestellt wurden,
- Bounce-Nachrichten über **qmail-send** sowie
- Nachrichten, die über das Quick Mail Protokoll in Empfang genommen wurden, mittels **qmail-qmqpd** und **qmail-qmtpd**.

Zum Ablegen der Nachrichen in die Queue bedienen sich diese Programme **qmail-queue**, wobei die eigentliche Nachricht auf File-Descriptor 0 in Empfang genommen wird und die Zustellungsinformation über FD 1.

Ausgelesen wird die Queue von **qmail-send**; ein etwaig notwendiger Clean-Up der Nachrichten erfolgt über **qmail-clean**. Es ist nicht möglich, die Verarbeitung der Nachrichten über Prioritäten zu steuern.

qmail-queue terminiert mit den folgenden Return-Codes, ohne allerdings Fehlermeldungen auszugeben:

Fehler Code	**Beschreibung**
0	Succesfully queued the message.
11	Address too long.
31	Mail server permanently refuses to send the message to any recipients. (Not used by **qmail-queue**, but can be used by programs offering the same interface.)
51	Out of memory.
52	Timeout.
53	Write error; e.g., disk full.
54	Unable to read the message or envelope.
55	Unable to read a configuration file. (Not used by **qmail-queue**.)

56	Problem making a network connection from this host. (Not used by **qmail-queue**.)
61	Problem with the qmail home directory.
62	Problem with the queue directory.
63	Problem with `queue/pid`.
64	Problem with `queue/mess`.
65	Problem with `queue/intd`.
66	Problem with `queue/todo`.
71	Mail server temporarily refuses to send the message to any recipients. (Not used by **qmail-queue**.)
72	Connection to mail server timed out. (Not used by **qmail-queue**.)
73	Connection to mail server rejected. (Not used by **qmail-queue**.)
74	Connection to mail server succeeded, but communication failed. (Not used by **qmail-queue**.)
81	Internal bug; e.g., segmentation fault.
91	Envelope format error.
99	Failed to queue the message.

*Tabelle 6-22: Exit Codes von **qmail-queue**; Codes zwischen 11 und 40 bezeichnen eine permanente Fehlersituation.*

Die eigentliche Nachricht wird von **qmail-queue** werder inspiziert noch geprüft. Allerdings fügt **qmail-queue** eine `Received:` Zeile an den Anfang der Nachricht (dem Nachrichten-Header) hinzu (vgl. Abschnitt 6.5.5.7).

6.6.2 Aufbau der Queue

Das Queue-Verzeichnis von Qmail ist in mehrere Unterverzeichnisse aufgeteilt, die ihrerseits stark untergliedert sind. Wir betrachen zunächst die "High-Level"-Verzeichnisse in der Qmail-Queue:

```
% su
# la -la /var/qmail/queue
drwxr-x---  11 qmailq  qmail  512 Apr 21 11:45 .
drwxr-xr-x  12 root    qmail  512 Jun 10 17:03 ..
drwx------   2 qmails  qmail  512 Aug 29 18:05 bounce
```

```
drwx------  25 qmails  qmail  512 Sep  5  2001 info
drwx------   2 qmailq  qmail  512 Aug 29 21:09 intd
drwx------  25 qmails  qmail  512 Sep  5  2001 local
drwxr-x---   2 qmailq  qmail  512 Sep  5  2001 lock
drwxr-x---  25 qmailq  qmail  512 Sep  5  2001 mess
drwx------   2 qmailq  qmail  512 Aug 29 21:09 pid
drwx------  25 qmails  qmail  512 Sep  5  2001 remote
drwxr-x---   2 qmailq  qmail  512 Aug 29 21:09 todo
```

Die Verzeichnisse `./mess` (und ihre Unterverzeichnisse) müssen im gleichen Dateisystem wie `./pid` vorhanden sein; gleiches gilt für `./todo` und `./intd`. Ein Blick in das Verzeichnis `./info` zeigt, dass Nachrichten und Zustellungsinformationen zum schnellen Zugriff in *hash directories* organisiert sind:

```
# cd info
# ls
0       11      14      17      2       22      5       8
1       12      15      18      20      3       6       9
10      13      16      19      21      4       7
```

Die gleiche Struktur ergibt sich für die Verzeichnisse `./local`, `./mess`, `./todo` sowie `./remote`. Die Anzahl der erzeugten Unterverzeichnisse wird durch den Wert in der Datei **conf-split** (Default: 23) festgelegt und bei der Installation durch das Kommando `make setup check` erzeugt und überprüft. Innerhalb der Hash-Verzeichnisstruktur werden Dateien gleichmässig auf die Unterverzeichnisse zu verteilt. Ist eine hohe Auslastung des Mail-Servers zu erwarten (> 20.000 E-Mail/Tag), sollte der Defaultwert entsprechend erhöht werden; aber immer eine Primzahl sein.

Hingegen besitzen die Verzeichnisse `./bounce`, `./lock` und `./pid` keine Unterverzeichnisse. Eine kleine Überraschung hat das Verzeichnis `./lock` parat. Es ergibt sich folgender Inhalt:

```
# cd ..
# cd lock
# ls -la
total 3
drwxr-x---   2 qmailq  qmail   512 Sep  5  2001 .
drwxr-x---  11 qmailq  qmail   512 Apr 21 11:45 ..
-rw-------   1 qmails  qmail     0 Sep  5  2001 sendmutex
```

```
-rw-r--r--   1 qmailr  qmail  1024 Aug 29 21:17 tcpto
prw--w--w-   1 qmails  qmail     0 Aug 30 11:05 trigger
```

In diesem Verzeichnis werden folgende Statusinformationen gesammelt:

- Die Named Pipe `trigger` dient zum Handshake zwischen **qmail-queue** und **qmail-send**, wie weiter unten erläutert, die Datei
- `tcpto` enthält — befüllt von **qmail-remote** — die IP-Adressen der Empfangsrechner Rechner (MTAs) nach einem TCP Verbindungs-TimeOut, und die Datei
- `sendmutex` wird beim Start von **qmail-send** gelockt, sodass hierüber sichergestellt ist, dass immer nur ein **qmail-send** Prozess für diese Queue läuft.

Wir wollen zunächst betrachten, wie Qmail die Nachrichten in der Queue organisiert. Unter der Kontrolle von **qmail-queue** wird jede E-Mail in der Queue durch eine eindeutige Nachrichtennummer identifizierte. Diese Nachrichtennummer entspricht einer Unix Inode-Nummer. Diese ist — bezogen auf das Intervall der Erstellung und der finalen Bearbeitung der Nachricht — eindeutig und ist der Queue-Process Identifier (QP-Id) der Nachricht.

Die Queue ist auf mehrere Unterverzeichnisse verteilt. Eine Nachricht mit der Nummer 457 wird wie folgt abgelegt:

`/var/qmail/queue/mess/.../457:`
: die von **qmail-queue** in die Queue eingestellte Nachricht

`/var/qmail/queue/todo/457:`
: die von **qmail-queue** bereitgestellte Zustellungsinformation (Sender/Empfänger)

`/var/qmail/queue/intd/.../457:`
: die Zustellungsinformation von **qmail-queue** in Bearbeitung

`/var/qmail/queue/info/.../457:`
: Senderadresse der Zustellungsinformation nach der Bearbeitung und Festlegung des Bearbeitungszeitpunkts (für **qmail-send** aufbereitet)

`/var/qmail/queue/local/.../457:`
: die lokale Empfängeradresse der Zustellungsinformation nach der Bearbeitung (für **qmail-send**)

`/var/qmail/queue/remote/.../457:`
: die remote Empfängeradresse der Zustellungsinformation nach der Bearbeitung (für **qmail-send**)

`/var/qmail/queue/bounce/.../457:`
: unzustellbare Nachrichten (von **qmail-send**)

Hierbei stehen die Punkte "`...`" für die Anzahl der angelegten, numerischen Unterverzeichnisse. Im Bearbeitungszyklus kann eine E-Mail in folgenden Zuständen sein:

Zustand	../mess	../intd	../todo	../info	../local	../remote	../bounce
S1	—	—	—	—	—	—	—
S2	+	—	—	—	—	—	—
S3	+	+	—	—	—	—	—
S4	+	?	+	?	?	?	— (queued)
S5	+	—	—	+	?	?	? (pre-processed)

Tabelle 6-23: Zustandsmatrix für Nachrichten in der Qmail-Queue.

***Legende**: Ein Zustand mit "+" bedeutet, dass eine Datei existiert; falls "—" existiert keine und falls "?" kann eine Datei vorhanden sein*

Die Zustellungsinformation ist im wesentlichen identisch mit der Sender/Recipient-Information des SMTP-Umschlags (Dan Bernstein spricht in der man-page von **qmail-queue** auch von "envelope information"), d.h. wird gebildet aus dem Buchstaben "`F`", der Senderadresse, einem 0 Byte und der Liste der Empfängeradressen. wobei jede Emfängeradresse mit einem "`T`" eingeleitet und ebenfalls mit einem 0 Byte terminiert, sowie der ganze String noch mit einem 0 Byte abgeschlossen wird. Empfängeradressen bestehen immer aus dem Empfängernamen, dem Klammeraffen "@" und dem FQDN des Empfangssystems.

Qmail organisisert seine Queue getrennt nach der eigentlichen Nachricht und der Zustellungsinformation; wobei nur die letztere relevant für die spätere Weiterverarbeitung ist. Während die Nachricht im "Auffangbehälter", d.h. dem Queue-Verzeichnis `./mess` bleibt, "wandert" die Zustellungsinformation im Zuge der Verarbeitung in die verschiedenen Stati, was bei Qmail gleichbedeutend ist mit der Existenz der Zustellungsinformation in verschiedenen Verzeichnissen. Im Vergleich zur Nachricht — deren Grösse leicht einige Megabyte betragen kann — ist die Zustellungsinformation nur wenige Byte gross, sodass nur wenig Informationen kopiert werden muss, was massgeblich zur Performanz der Queue und ihrer Stabilität beträgt.

Im Gedankenmodell kannn die Queue ferner in einen einlaufenden Zweig und einen auslaufenden Zweig unterteilt werden. Zur Bearbeitung der einlaufenden Nachricht ist **qmail-queue** zuständig, während die Verarbeitungsschritte im auslaufenden Zweig als "Mail Preprocessing" bezeichnet wird und unter der Regie von **qmail-send** erfolgt.

Zum Einfügen einer E-Mail in die Queue generiert **qmail-queue** zunächst eine Datei im Verzeichnis `./pid` mit eindeutigem aber beliebigen Namen. Im Dateisystem wird diese Datei unter ihrer Inode-Nummer (z.B. 457) geführt und **qmail-queue** locked diese Datei, so dass sich die E-Mail nun im Zustand **S1** befindet. **qmail-queue** benennt die beliebige Datei unter `./pid/Datei` nun in die Inode-Nummer (457) um und verschiebt sie nach `./mess/.../475`.

Die neue Datei 457 ist nun im Zustand **S2**. In der weiteren Bearbeitung wird eine zusätzliche Datei `./intd/.../457` im Zustand **S3** erzeugt, die die Zustellungsinformation enthält. Anschliessend wird ausgehend von `./intd/457` ein Link `./todo/457` im erzeugt. Zu diesem Zeitpunkt, d.h. dem Zustand **S4,** ist die Nachricht erfolgreich gequeued und **qmail-queue** signalisert dies durch Schreiben in die Datei `./lock/trigger`, die als Named Pipe vorliegt. Sollte **qmail-send** "schlafen", wird es hierdruch "geweck" und die weitere Verarbeitung wird anschliessend von **qmail-send** vorgenommen.

6.6.3 Mail-Prepeprocessing, Zustellung und Clean-Up der Queue

Wurde eine Nachricht erfolgreich in die Queue eingefügt (Zustand **S4**), überprüft **qmail-send** zunächst den Zustand der Triggerdatei `./lock/trigger`. Im weiteren ermittelt **qmail-send** über die Datei `./todo/457`, ob die Nachricht lokal oder an ein entferntes System (remote) oder sowohl-als-auch auszuliefern ist. Dies wird anhand der Empfängeradresse festgestellt, die eventuell auch geändert werden kann.

Liegt eine Datei z.B. `./todo/457` vor, so weiss **qmail-send**, dass Nachricht 457 im Zustand **S4** ist. **qmail-send** entfernt nun `./info/457` und `./local/.../457`, bzw. `./remote/.../457`, falls diese vorhanden sind.

Die Zustellungsinformation liegen weiter unter `./todo/457` vor. Entsprechend diesen Informationen generiert **qmail-send** die Dateien `./info/457`, sowie ggf. `./local/.../457` bzw. `./remote/.../457`. Abschliessend wird auch `./intd/.../457` gelöscht. Zu diesem Zeitpunkt — immer noch im Zustand **S4** — wird `./todo/457` entfernt, was dem Zustand **S5** entspricht: Die Nachricht wurde erfolgreich bearbeitet (preprocessed).

Während die also eigntlichen Nachricht in `./mess/..../457` verbleibt, liegt die Zustellungsinformation entweder in `./local/.../457` oder `./remote/.../457` oder in beiden vor und es obliegt **qmail-send**, nun entweder **qmail-local** oder **qmail-remote** (oder beide) zum Einsatz zu bringen.

Ob eine Nachrich an eine Empfängeradresse in `./local/.../457` oder `./remote/.../457` ausgeliefert wurde stellt **qmail-send** über Flags, d.h, konkret einem Buchstaben, der an den Anfang jeder einzelnen Empfängeradresse

angeheftet wird (was wir auch schon bei QMQP gesehen haben):

- `T` — Nachricht ist neu und muss zugestellt werden,
- `D` — permanter Zustellungsfehler,
- `Z` —temporärer Zustellungsfehler; Nachricht wird zurückgestellt (*deferred*) und ihre Zustellung später erneut versucht,
- `K` — Nachricht wurde erfolgreich zugstellt.

Konnte die Nachricht letztendlich nicht ausgeliefert werden, generiert **qmail-send** die Datei `./bounce/457`; falls sie nicht schon existiert (aufgrund anderer Empfängeradressen). Bounce-Nachrichten können von **qmail-send** zu einem beliebigen Zeitpunkt— ausgehend von `./bounce/457` und `./mess457` — erzeugt werden. Anschliessend wird `./bounce/457` gelöscht.

Wurden alle Adressen unter `./local/.../457` abgearbeitet, löscht **qmail-send** auch diese Datei. Gleiches gilt für `./remote/.../457`. In der weiteren Verarbeitung werden auch evtl. Bounce-Dateien unter `./bounce/457` entfernt und anschliessend `./info/457`. Somit ist die Zustellung wieder im Zustand **S2**. Zum Abschluss erleidet die Datei ./**messs/.../457** das gleiche Schicksal; das Processing ist nun im Zustand **S1**.

Das eigentliche Löschen von Dateien delegiert **qmail-send** (als Daemon-Prozess) **qmail-clean** in periodischen Abständen; spätestens aber nach 24 Stunden. Die Qmail-Queue und ihre Abarbeitung ist also auf Sicherheit ausgerichtet. Ein Crash des Rechners oder des Dateisystems (Inkonsistenz) führt schlimmstenfalls zur doppelten Versendung einer Nachricht; niemals aber zu ihrem Verlust.

6.6.4 Qmail-Kanäle

Wir haben bereits kenngelernt, dass es sinnvoll ist die Qmail-Queue in eine einlaufende und auslaufende Queue gedanklich zu trennen. Dan Bernstein hat — wenig offensichtlich — die auslaufende Queue noch in zwei Kanäle getrennt:

- *Local channel* (`L`): Die Nachrichten in der Queue für lokale Empfänger (`./local/.../457`) sowie
- *Remote channel* (`R`): Die Nachrichten in der Queue, die für entfernte Empfänger bestimmt sind, z.B. `./remote/..../457`.

Bekanntlich "feuert" **qmail-send** für die Zustellung der Nachrichten im ersten Kanal **qmail-local** und **qmail-remote** für die Nachrichten im zweiten Kanal. Diese Trennung macht nicht nur unter diesem Gesichtspunkt Sinn; sondern auch hinsichtlich der Verwaltung der unterschiedlichen Queues: Nachrichten im lokalen Kanal werden lokal zugestellt, d.h. obliegen letztlich **qmail-local**. Für Nachrichten an entfernte Empfänger ist **qmail-remote** zuständig. Während die

Zustellung lokaler Nachrichten quasi per constructionem sichergestellt ist (sieht man einmal von Fehlerfällen z.B. hinsichtlich nicht vorhander Mailverzeichnisse oder Rechte ab), kann prinzipiell die remote Zustellung nicht garantiert werden. Einerseits fliessen hier die Stati der TCP/SMTP-Verbindung und ihre möglichen TimeOuts eins; andererseits kann der Empfangshost immer noch die Annahme temporär oder permanent mit den entsprechenden SMTP-Return-Codes verweigern.

Kurz geesagt, muss für die Verwaltung der Nachrichten im *Remote channel* eine umfangreiche Buchhaltung vorgenommen werden, die unterschiedliche Fehlersituationen und Timer berücksichtigt. Dies kann für die Nachrichten im *Local channel* praktisch komplett entfallen. Nicht zustellbare Nachrichten werden hier als *deferred* betrachtet und die Zustellung einfach immer weiter versucht.

6.6.5 Delivery Schedule

Eine wichtige Konsequenz der Arbeitsweise der Qmail-Queue ist, dass Nachrichten, die neu in die Queue gestellt werden sofort ausgeliefert werden; es gibt also kein "Turn-Around" Parameter, wie dies z.B. bei dem Aufruf von Sendmail (`sendmail -q15`) typisch ist. Kann die Nachricht nicht an den Empfänger zugestellt werden, wird die erneute Auslieferung nicht in festen Zeitintervallen vorgenommen, sondern Qmail (**qmail-send**) wartet nach einem quadratischen Schema immer länger. Hierdurch wird letztlich die Zustellung "alter" Nachrichten in der Queue weniger häufig versucht. Es wird also weniger Zeit darauf verschwendet, evtl. problematische E-Mails loszuwerden.

- Für Nachrichten im local Channel gilt:

  $T_{queue} = 100ms * N^2_{deliver}$

  $T_{delsy} = 200ms * N_{deliver} + 100ms$

- Für Nachrichten im remote Channel folgt hingegen:

  $T_{queue} = 400ms * N^2_{deliver}$

  $T_{delay} = 800ms * N_{deliver} + 400ms$

Hierbei ist T_{queue} die Zeit der Nachricht in der Queue, T_{delay}, die Zeit, die **qmail-send** bis zur erneuten Bearbeitung wartet und $N_{deliver}$ die Anzahl der Zustellungsversuche.

Peter Samules von der SAGE in der Australien hat dieses Verhalten in einer schönen Graphik dargestellt, auf die ich gerne zurückgreifen möchte:

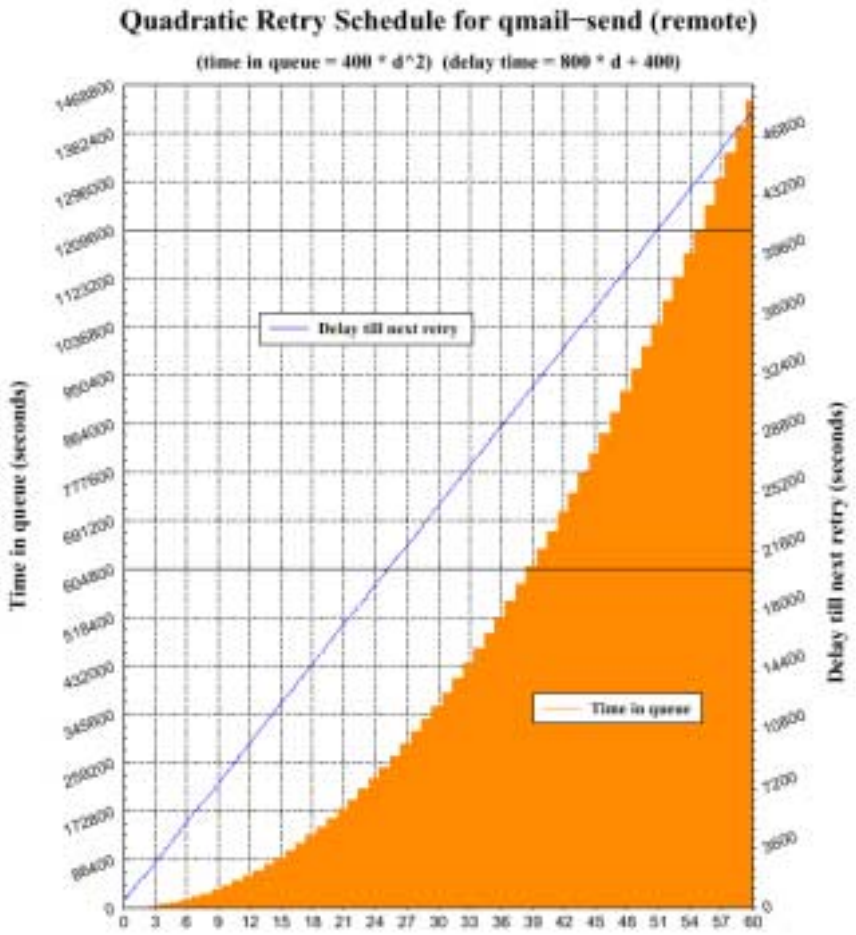

*Abbildung 6-6: Quadratisches Zustellungsverhalten von **qmail-send** für Nachrichten. Die Anzahl der Zustellungsversuche N ist auf der Abszisse dargestellt, die Verweildauer der Nachricht in der Queue T_{queue} auf der linken Ordinate, und die Verzögerung bis zur erneuten Auslieferung T_{delay} auf der rechten Ordinate.*

Über die Konfigurationsdatei `./control/queuelifetime` wird die Zeit festgesetzt, wie lange eine Nachricht maximal in der Queue bleibt, bis sie gelöcht wird. Ist kein Wert eingestellt, nimmt **qmail-send** 604800 Sekunden, d.h. eine Woche an. Ist diese Verweildauer abgelaufen, versucht **qmail-send** es ein letztes Mal, um dann (falls die Auslieferung wiederum scheitert) die Nachricht als permanent unzustellbar zu klassifizieren und zu bouncen.

6.6.6 Queue Management

Qmail macht dem Administrator das Management der Queue nicht einfach; im wesentlich — bedingt durch den komplexen Aufbau der Queue und der Stati der einzelnen Nachrichten hierin — sollte auf einen Eingriff ohne geeignet Hilfsmittel ganz verzichtet werden. Wir haben bereits erfahren, dass es keine Möglichkeit gibt, diesen inneren Mechanismus der Queue für einzelne Nachrichten in der Queue zu beinflussen. Der Zeitstempel der Nachricht in der Queue wird in der Datei `./info/.../457` festgehalten und damit auch der letzte Zustellungsversuch, der Zustellungsstatus einer Nachricht wird für den remote und local Channel über einen Kennbuchstaben (vgl. oben) vorgenommen.

Im wesentlichen geht es beim Queue Mangement um vier Aspekte:

1. Wieviele Nachrichten stehen noch in der Queue und wie ist ihr Status?
2. Kann ich das erneute, soforige Zustellen aller oder einzelner Nachrichten erzwingen?
3. Wie kann ich einzelne Nachrichten in der Queue löschen?
4. Wie kann ich eine korrumpierte Queue erkennen und ggf. wieder herstellen?

Dan Bernstein hat uns für diese Aufgaben lediglich vier Programme spendiert:

- **qmail-qread** Programm zum Aulesen der Queue und Ausgabe der Zustellungsinformationen einzelner Nachrichten in der Queue.
- **qmail-qstat** Shell-Skript, das **qmail-qread** nutzt, und eine Überblicksinformation die Nachrichten in der Queue in ihren Preprocessed-Status liefert.
- **qmail-tcpto** Programm zur Anzeige derjenigen MTAs, für die zuzustellende E-Mails im Zustand TCP Time-Out für länger als 60 Minuten stehen.
- **qmail-tcpok** Programm zum Zurücksetzten der TCP-Statusinformation für Nachrichten im remote Channel

6.6.6.1 Queue-Verwaltung mit Bordmitteln

Glücklicherweise hat Dan Bernstein uns mit einem kleinen aber wichtigen Werkzeug ausgestattet, eine sofortige, unkonditionale Auslieferung aller Nachrichten in der Queue zu erzwingen: das **qmail-send** ARLM Signal. Der Aufruf

```
# svc -a /service/qmail-send
```

führt zum Triggern der **qmail-send** Zustellung und zum sofortigen Füllen der Prozessliste mit **qmail-rspawn** und **qmail-lspawn** Aktivitäten, bis die Concurrency-Limits erschöpft sind.

Sollen alle Nachrichten in der Queue möglichst schnell und unter Qmail-Kontrolle gelöcht werden, lässt sich dies das Setzen des Eintrags in `./control/queuelifetime` z.B. auf 10 (Sekunden) und über einen anschliessenden Stopp und Start von **qmail-send** realisieren. Ist die Queue leer, wird der ursprüngliche Parameter in `./control/queuelifetime` wiederhergestellt und in Folge ist wiederum **qmail-send** zu stoppen und zu starten.

Gelegentlich hilfreich ist auch das Programm **qmail-tcpok**. War z.B. der MTA `mailer.taffline.de` für einige Zeit nicht erreichbar, und haben sich die Nachrichten für diesen Rechner in der Queue angestaut und sind in einem TCP-TimeOut gelandet, kann dies zunächst mit dem Kommando

```
# /var/qmail/bin/qmail-tcpto
```

```
194.25.134.72 timed out 2564 seconds ago; recent timeouts:
10
```

festgestellt werden. Weis der Qmail-Administrator, dass der Rechner jetzt wieder Online ist, kann er mittels

```
# /var/qmail/bin/qmail-tpcok
```

die TCP TimeOut-Tabelle zurücksetzen und **qmail-remote** somit zu einer sofortigen, erneuten Auslieferung der E-Mails bewegen.

An dieser Stelle sei die Warnung ausgesprochen, Dateien in der Qmail-Queue "per Hand" anzufassen und ggf. zu löschen. Häufig ist die Folge eine Inkonsistente Queue, die auch gelegentliche dann auftritt, wenn der Rechner im Verlauf des Qmail-Prozesses abgestürzt ist. Zwar repariert **qmail-send** die Queue wieder, jedoch sind anschliessend hässliche Meldungen im **qmail-send** Logfile auszumachen.

Eine Möglichkeit, die Konsistenz der Qmail-Queue zu überprüfen ist das **qmail-sanity** Perl-Skript von Russell Nelson (`htttp://www.qmail.org/qmail-qsanity-0.52`). Wer es riskieren kann, kann — sofern keine Verluste von Nachrichten zu befürchten sind — auch die gesamte Queue unter `/var/qmail/queue/` löschen. Durch den Wechsel ins Qmail-Installationsverzeichnis und dem Aufruf

```
# make check
```

wird das komplette (leere) Queue-Verzeichnis neu generiert. Diese Massnahme ist auch dann notwendig, wenn z.B. der `conf-split` Parameter erhöht wurde.

6.6.6.2 qmHandle

Ein meiner Meinung nach absolut wichtiges Werkzeug zur Manipulation der Nachrichten in der Queue hat Michele Betram beigesteuert **qmHandle**.

`qmHandle` (aktuell: Version 1.0.0) ist ein Perl-Skript, das zur Anzeige der

Nachrichten in der Queue aber vor allem zum Löschen einzelner Nachrichten herangezogen werden kann. Hierzu stützt es sich das Programm **qmail-qread**, "läuft" durch den Queue-Verzeichnisbaum und löscht die korresponideren Dateien. Um dies ohne Inkonsistenz der Qmail-Queue zu vollziehen, stoppt es **qmail-send** zuerst und startet es anschliessend wieder.

Das Skript kommt in einem Tar-Archiv. Nach dem Auspacken sollte man es nach `/usr/local/bin/` oder nach `/var/qmail/bin/` kopieren. **qmHandle** braucht das Wissen über die lokale Qmail-Konfiguration; d.h. wo liegt die Qmail-Queue und wie wird Qmail gestartet bzw. gestoppt? Hierzu sind folgende Passagen von **qmHandle** anzupassen.

```
#!/usr/bin/perl

# qmHandle 1.0.0

# Copyright(c) 1998 -> 2001 Michele Beltrame <mick@io.com>

# This program is distributed under the GNU GPL.

# For more information have a look at http://www.gnu.org

use strict;

use diagnostics;

#################### USER CONFIGURATION BEGIN
####################

#####

# Set this to your qmail queue directory (be sure to include
the final slash!)

my ($queue) = '/var/qmail/queue/';

#####

# If your system has got automated command to start/stop
qmail, then

# enter them here.

# ### Be sure to uncomment only ONE of each variable
declarations ###

# For instance, this is if you have DJB's daemontools

my ($stopqmail) = '/usr/local/bin/svc -d /service/qmail-
send';

my ($startqmail) = '/usr/local/bin/svc -u /service/qmail-
send';

# While this is if you have a Debian GNU/Linux with its
qmail package

#my ($stopqmail) = '/etc/init.d/qmail stop';

#my ($startqmail) = '/etc/init.d/qmail start';
```

```
# If you don't have scripts, leave $stopqmail blank (the
process will
# be hunted and killed by qmHandle):
#my ($stopqmail) = '';
# However, you still need to launch qmail in a way or the
other. So,
# if you have a standard qmail 1.03 use this:
#my ($startqmail) = "csh -cf '/var/qmail/rc &'";
# While, if you have a standard qmail < 1.03 you should use
this:
#my ($startqmail) = '/var/qmail/bin/qmail-start ./Mailbox
splogger qmail &';
#####
# Enter here the system command which returns qmail PID. The
following
# should work on most Unixes:
my ($pidcmd) = 'pidof qmail-send';
####################  USER CONFIGURATION END
####################
```

Listung 6-17: Die Konfiguration der Datei `qmHandle`; angepasst für Qmail unter Supervise-Kontrolle, Leerzeichen wurden unterdrückt.

Zur Prozessverwaltung stützt sich **qmHandle** auf das Programm **pidof**, dass unter Linux im allgemeinen vorliegt bei FreeBSD über das Paket `pmisc` installiert werden muss.

Mit **qmHandle** können folgende Aktionen vollzogen werden:

```
# qmHandle
qmHandle v1.0.0
by Michele Beltrame
Wrong parameters entered, available ones are:
  -l   : list message queues
  -L   : list local message queue
  -R   : list remote message queue
  -s   : show some statistics
  -vN  : display message number N
  -dN  : delete message number N
```

```
  -D   : delete all messages in the queue (local & remote)
Additional (optional) parameters are:
  -c   : display colored output
  -N   : list message numbers only
         (to be used either with -l, -L or -R)
You can view/delete multiple message eg -d123 -v456 -d567
```

Sollen einzelne Nachrichten in der Queue gelöscht werden, können diese zunächst über den Empfänger/Sender oder das "Subject:" ermittelt werden. Anschliessend lässt sich hierüber die Message-ID ermitteln und somit gezielt diese Nachrichten in der Queue löschen.

Da hierzu immer **qmail-send** gestartet bzw. gestoppt werden muss, ist die Manipulation der Nachrichten aber zumindest zeitaufwändig. Wir betrachten abschliessend ein Beispiel für die Ausgabe von qmHandle:

```
# qmHandle -l
....
53956 (21, L)
  Return-path: dns-return-17202-feh=fehcom.de@list.cr.yp.to
  From: =?ISO-8859-1?Q?Juli=E1n_Mu=F1oz?=
<jmunoz@telefonica.net>
  To: <dns@list.cr.yp.to>
  Subject: djbdns log analysis
  Date: Wed, 3 Apr 2002 14:10:11 +0000 (GMT)
  Size: 2681 bytes


53979 (21, L)
  Return-path: dns-return-17204-feh=fehcom.de@list.cr.yp.to
  From: Felix von Leitner <leitner@fefe.de>
  To: 'djbdns Mailing List ' <dns@list.cr.yp.to>
  Subject: Re: pdns - has somebody got any background
information?
  Date: Wed, 3 Apr 2002 16:41:20 +0200
  Size: 2952 bytes
....

Messages in local queue: 18
```

```
Messages in remote queue: 0
```

Während die meisten Ausgabezeilen von **qmHandle** selbsterklärend sind, bedarf z.B. die Zeile

```
53979 (21, L)
```

einer gesonderten Erklärung: Die erste Zahl `53979` ist die Message-Id (`msg`); diese entspricht der Inode der eingestellten Nachricht. Über die Angaben in Klammer `(21, L)` wird zunächt festgestellt, dass es sich im eine Nachricht im *Local channel* der Queue handelt (`L`), sowie in welchem Unterverzeichnis die Nachricht liegt (`21`).

6.7 Sendmail Kompatibilität

Die Frage der Sendmail Kompatibilität von Qmail stellt sich insbesonders dann, wenn es darum geht, eine bestehendes Sendmail-System nach Qmail zu migrieren. Häufig ist es das erste Ziel, die bestehende Installation — mit Ausnahme des "Austauschs" des MTA — weiter zu benutzen. Qmail liefert hierzu in erster Linie den **sendmail**-Wrapper bei. Des weiteren besteht die Möglichkeit sowohl die alten Maildateien sowie auch die im Einsatz befindlichen **sendmail** Alias-Listen zu benutzen.

6.7.1 sendmail-Wrapper

Bei den meisten Unix-Systemen ist sendmail als Standard-Mailsystem verankert. So nutzt z.B. der Cron-Daemon von Paul Vixie **sendmail**. Hierbei wird die Ausgabe der in den (Benutzer-spezifischen) `crontabs` hinterlegten Kommandos dem Eigentümer der `crontab` versandt. Um dies auch unter Qmail zu gewährleisten, verfügt Qmail zum einen über den **sendmail**-Wrapper und zum anderen über die Möglichkeit, E-Mails an *root* einem anderen Account zuzuweisen:

- Das alte **sendmail**-Programm ist zu entfernen und der **sendmail**-Wrapper an seine Stelle zu linken, wie in Abschnitt 6.3.4 beschrieben.
- Für den User *root* ist ein geeignete dot-qmail Aliasdatei zu erzeugen, also `/var/qmail/alias/.qmail-root`.

Der Qmail sendmail-Wrapper emuliert nur einige **sendmail**-Optionen (vgl. Tabelle 6-4), bei der Nutzung des **sendmail**-Befehls in Skripten (z.B. CGI oder PHP) sollte darauf geachtet werden. Dies gilt auch für die korrekte Syntax des Headers und der obligartorischen Verzichts auf den Carriage Return (CR). Qmail ist hier u.U. weniger tolerant als das alte **sendmail**.

Die dot-qmail Datei ermöglicht die Weiterleitung der E-Mail an einen anderen Benutzer oder die lokale Zustellung in eine Mailbox bzw. ein Mailverzeichnis. Beides findet in jedem Fall unter der effektiven UID des Qmail Users *alias* statt. Der *alias* User muss daher Schreibrechte in die entsprechende Datei bzw. das Verzeichnis haben; was in der Regel nicht der Fall ist. Daher sollte man sich zur Regel machen, bei der Nutzung der dot-qmail Dateien unter `/var/qmail/alias/` nur die Weiterleitungsfunktion zu nutzen. Ein Eintrag wie

```
/var/qmail/alias/.qmai-root
# Mail an root
/var/spool/mail/root
```

wird daher nicht funktionieren!

6.7.2 Auslieferung nach /var/spool/mail

Unter Sendmail werden die einlaufenden Nachrichten in einem zentralen Verzeichnis `/var/spool/mail/` abgelegt, und für jeden lokalen Empfänger in einer eigenen Datei hinterlegt, die dem Namen des Accounts entspricht:

```
/var/spool/mail/$USER
```

wobei `$user` dem lokalen Account-Namen des Empfängers entspricht. Der lokale MDA (Mail Delivery Agent) unter Sendmail ist das Programm **mail**, was in der Konfigurationsdatei von Sendmail `/etc/sendmail.cf` über das Label Mlocal bekannt gemacht wird. Die Zuordnung zwischen den SMTP-Empfangsadressen — speziell dem lokalen Teil der Adresse — und dem Benutzer-Name unter Unix, geschieht über den Alias-Mechanismus, der uns auch schon von Qmail bekannt ist.

Qmail kann sich in der Defaultdelivery (vgl. Abschnitt 6.4.4) ebenfalls des Sendmail MDAs **mail** bedienen. Dankenswerterweise hat uns Dan Bernstein bereits unter `/var/qmail/boot/` einige Konfigurationsskripte hinterlegt, die diesen Einsatz beispielhaft für BSD 4.4, System 5 sowie Unix V7 beschreiben:

```
#!/bin/sh
# Using splogger to send the log through syslog.
# Using binmail to deliver messages to /var/spool/mail/$USER
by default.
# Using BSD 4.4 binmail interface: /usr/libexec/mail.local -
r
exec env - PATH="/var/qmail/bin:$PATH" \
qmail-start \
'|preline -f /usr/libexec/mail.local -r "${SENDER:-MAILER-
```

```
DAEMON}" -d "$USER"' \
splogger qmail
```

*Listing 6-18: Bespiel für die Nutzung des **mail** MDA in der Defaultdelivery von Qmail für BSD-Systeme.*

```
#!/bin/sh
# Using splogger to send the log through syslog.
# Using binmail to deliver messages to /var/spool/mail/$USER
by default.
# Using SVR4 binmail interface: /bin/mail -r
exec env - PATH="/var/qmail/bin:$PATH" \
qmail-start \
'|preline -f /bin/mail -r "${SENDER:-MAILER-DAEMON}" -d
"$USER"' \
splogger qmail
```

*Listing 6-19: Bespiel für die Nutzung des **mail** MDA in der Defaultdelivery von Qmail unter System V (Linux).*

Wir beachten, dass in beiden Fällen das Qmail-Programm **preline** eingesetzt wird. Dies ergänzt die Nachricht durch die Header-Zeilen "Return-Path:" sowie "Delivered-To:", was ansonsten Aufgabe von **qmail-local** ist.

6.7.3 Aliases-Datei

Befand sich das Sendmail-System schon lange im Einsatz und untersützt es viele Endanwender, existiert wahrscheinlich ein umfangreiche Mapping zwischen E-Mail-Adressen und den Namen der Benutzeraccounts. Dieses Mapping wird in der `aliases` Datei verdrahtet, die typischerweise unter `/etc/aliases` zu findent ist. sendmail liest nicht die ursprüngliche `aliases` Datei ein, sondern verlangt eine Binärform hiervon als `/etc/aliases.cdb`, die über das Programm **aliases** erzeugt wird und dann on-the-fly wirksam wird.

Diese `aliases` Datei kann einschliesslich der ":inlcude" Anweisungen unverändert unter Qmail verwendet werden; nur muss sie nun vom Qmail-Programm **newaliases** übersetzt werden. Die systemweite **aliases** Datei wird stellvertretend von Qmail alias Benutzer nur die **fastforward** Anweisung in der Datei `/var/qmail/alias/.qmail-default` aktiviert:

```
# /var/qmail/aliases/.qmail-default
# Include sendmail's aliases cdb
```

```
# Don't forget to call newaliases after each change
|/var/qmail/bin/fastforward /etc/aliases.cdb
```

*Listing 6-20: Nutzung des Qmail **alias** Benutzers zur Auswertung der per **newaliases** übersetzten `aliases` Datei von Sendmail.*

6.7.4 Der .forward-Mechanismus

Ebenso kann ein komplexes Weiterleitungsregelwerk in Form der Datei `~/.forward` formliert sein. Auch diese lässt sich unter Qmail weiter verwenden. Allerding benötigen wir hierzu zusätzlich das dot-forward Programm-Paket von Dan Bernstein (`http://cr.yp.to/dot-forward.html`). Dies muss gesondert installiert werden. Die ausführbare Datei **dot-forward** installiert sich im Verzeichnis `/var/qmail/bin/` und kann dann im Startaufruf von Qmail eingebaut werden, was wie folgt geschieht:

```
#!/bin/sh
# Using splogger to send the log through syslog.
# Using dot-forward to support sendmail-style ~/.forward
files.
# Using binmail to deliver messages to /var/spool/mail/$USER
by default.
# Using BSD 4.4 binmail interface: /usr/libexec/mail.local -
r
exec env - PATH="/var/qmail/bin:$PATH" \
qmail-start '|dot-forward .forward
|preline -f /usr/libexec/mail.local -r "${SENDER:-MAILER-
DAEMON}" -d "$USER"' \
splogger qmail
```

*Listing 6-21: Bespiel für die Nutzung von **dot-forward** in der Defaultdelivery von Qmail für BSD-Systeme.*

```
#!/bin/sh
# Using splogger to send the log through syslog.
# Using dot-forward to support sendmail-style ~/.forward
files.
# Using binmail to deliver messages to /var/spool/mail/$USER
by default.
# Using SVR4 binmail interface: /bin/mail -r
exec env - PATH="/var/qmail/bin:$PATH" \
```

```
qmail-start '|dot-forward .forward

|preline -f /bin/mail -r "${SENDER:-MAILER-DAEMON}" -d
"$USER"' \

splogger qmail
```

*Listing 6-22: Bespiel für die Nutzung von **dot-forward** in der Defaultdelivery von Qmail unter System V (Linux).*

Die Unterstützung der `~./forward` Anweisung geht allerdings nicht so weit wie beim Alias-Mechanismus. Das typische Forwarding und die Weiterleiten an Programme kann genutzt werden; das Umleiten in Dateien aber nicht. Ebensowenig werden hier die ":include:" Anweisungen ausgeführt.

6.7.5 Procmail

Der Mail-Processor **procmail** lässt sich über eine dot-qmail Datei natürlich von jedem Benutzer individuell nutzen. Wenn es aber darum geht, systemweit eine Qmail-MTA mit procmail Unterstützung aufzusetzen.

Das procmail-Regelwerk wird über die Konfigurationsdatei `~./procmail` festgelegt. Zwar kann das Programm individuell über die `~./forward` Datei gestartet werden; beim systemweiten Einsatz lässt sich **procmail** aber auch als Standard-MDA verwenden, was aus folgendem Qmail Startskript hervorgeht:

```
#!/bin/sh

# Using splogger to send the log through syslog.

# Using procmail to deliver messages to
/var/spool/mail/$USER by default.

exec env - PATH="/var/qmail/bin:$PATH" \

qmail-start '|preline procmail' splogger qmail
```

*Listing 6-23: Bespiel für die Nutzung von **procmail** in der Defaultdelivery von Qmail.*