DNS & BIND
GE-PACKT

Johannes Franken

DNS & BIND GE-PACKT

Bibliografische Information Der Deutschen Bibliothek
Die Deutsche Bibliothek verzeichnet diese Publikation in der Deutschen Nationalbibliografie; detaillierte bibliografische Daten sind im Internet über http://dnb.ddb.de abrufbar.

ISBN 3-8266-1502-6
1. Auflage 2005

Printed in Germany

Lektorat: Gunnar Jehle
Sprachkorrektorat: Petra Heubach-Erdmann
Satz und Layout: G&U e.Publishing Services GmbH, Flensburg
Druck: Kösel, Krugzell

Inhaltsverzeichnis

Vorwort

Kaum ein anderes Open-Source-Softwareprojekt ist so essenziell für das einwandfreie Funktionieren des Internets wie BIND (Berkeley Internet Name Domain). Dieses Softwarepaket entstand lange, bevor man überhaupt »Open Source« als eigenständigen Begriff geprägt hatte, und ist von jeher im Quelltext verfügbar. Es feierte in seiner ursprünglichen Version inzwischen seinen 20. Geburtstag.

Mit der Version 9 liegt eine völlig neu entwickelte Variante dieses Softwarepakets vor, das in wesentlichen Punkten erweitert wurde, um den heutigen Anforderungen zu genügen. Diese Neuerungen umfassen unter anderem die Möglichkeit, Sonderzeichen in Domain-Namen zu nutzen, sowie die Auswahlmöglichkeit zwischen verschiedenen Datenbankmodulen (LDAP, BerkeleyDB, Dateisystem).

Das vorliegende Buch geht auf alle Features von BIND 9.3.0 wie beispielsweise die Unterstützung von IPv6, dynamischen Updates und inkrementellen Zonentransfers ein und beschreibt jeden DNS-Recordtyp sowie alle Konfigurationsanweisungen. Ein Grundlagenkapitel über DNS verschafft den notwendigen Überblick über die komplexe Materie. In den Folgekapiteln werden die Konfiguration und Administration der aktuellen BIND-Version konkret behandelt.

Die Verweise auf die in BIND implementierten RFCs erlauben einen noch tieferen Einstieg in die Details. Auch auf die Bedeutung des DNS für das Microsoft Active Directory wird kurz eingegangen. Die Erläuterungen über die Konfiguration von BIND bei der Nutzung des so genannten Classless Inter Domain Routings (CIDR) sind sehr hilfreich, da CIDR aufgrund der Knappheit von IPv4-Adressen in den letzten Jahren stark an Bedeutung gewonnen hat.

Zahlreiche Beispielkonfigurationen und konkrete Administrationshinweise runden das Werk ab. Damit ist es ein kompaktes und praktisches Nachschlagewerk für die tägliche Arbeit, das jeder Netzwerkadministrator jederzeit griffbereit haben sollte.

Thomas Uhl
Vorstand der Topalis AG

E Über diese Referenz

E.1 Abgrenzung

Das **Domain Name System** (DNS) ist

- ein *Verfahren* zum dezentralen Speichern und Replizieren hierarchischer Adressdaten.
- der Name der *Datenbank*, die die Zuordnung zwischen symbolischen Hostnamen und numerischen IP-Adressen im Internet enthält. Ihre Datensätze liegen nicht zentral vor, sondern sind auf hunderttausende so genannter »Nameserver« verteilt.
- der Name des *Protokolls* zur Kommunikation mit den Nameservern.

Die Anwendung des DNS ist nicht auf das Internet begrenzt: In vielen Netzen wird das DNS auch zur internen IP-Kommunikation eingesetzt.

Einige Organisationen verwenden das DNS zur Verwaltung ihrer DECNet-Adressen, Public-Keys, Usernamen, Passwörter und Telefonnummern. Da diese Aufgabe jedoch mit den neueren und spezialisierteren *Verzeichnisdiensten* (z.B. LDAP) besser lösbar ist, konzentriert sich dieses Buch auf die Verwaltung von IP-Adressen.

Die **Berkeley Internet Name Domain** (BIND) ist die *Implementierung* eines Name*servers* und eines *Resolvers*, die beide zurzeit weit verbreitet sind.

Studien von Prof. Dr. Daniel J. Bernstein (`http://cr.yp.to/surveys/dns1.html`) und Mark Lottor (`http://www.nw.com/`) zeigen, dass mehr als 75% aller Internetdomains unter BIND laufen.

E.2 Zielgruppe

Diese Referenz richtet sich an

- Systemadministratoren, die einen Nameserver installieren, optimieren oder betreiben möchten

- fortgeschrittene Anwender, die einen Blick hinter die »Kulissen« des Internets werfen möchten
- Software-Entwickler, die das DNS in ihre Anwendungen integrieren möchten

Zum Verständnis des DNS benötigen Sie Kenntnisse in TCP/IP, die Sie sich zum Beispiel durch die Lektüre folgender Bücher aneignen können:

- W. Richard Stevens: *TCP/IP*, Hüthig-Telekommunikation 2003, ISBN 3-8266-5042-5
- Douglas E. Comer: *TCP/IP*, 4. Auflage, mitp-Verlag 2003, ISBN 3-8266-0995-6

E.3 Aufbau

DNS & BIND GE-PACKT besteht aus vier Teilen:

1. Der erste Teil befasst sich mit den Konzepten und dem Einsatz des DNS.
2. Der zweite Teil beschreibt alle Konfigurationsmöglichkeiten von BIND und den Aufbau BIND-kompatibler Zonefiles. Dabei werden viele Beispiele aus der Praxis beleuchtet, die Sie leicht übernehmen können.
3. Der dritte Teil geht auf die Administration von BIND ein. Von der Installation und Updates über Security-Maßnahmen und Admintools bis zur Fehlersuche werden alle Themen behandelt, die auf Hostmaster zukommen.
4. Der Anhang erklärt die Geschichte und Aufgabe der Organisationen, die im Internet für den Betrieb des DNS zuständig sind, und enthält Listen aller Top-Level-Domains, Root-Nameserver, RFCs und Abkürzungen.

E.4 Schreibweisen

In diesem Buch werden folgende Schreibweisen verwendet:

- Die Namen von Dateien, Domains, Zonen etc. erscheinen in Blockschrift: z.B. `/etc/named.conf`
- Listen, Dateiinhalte und Scripts erscheinen auf hellgrauem Hintergrund und tragen eine Unterschrift:

```
search informatik.uni-frankfurt.de uni-frankfurt.de
```

```
nameserver 141.2.1.1
nameserver 141.2.1.3
sortlist 127.0.0.1 10.0.0.0 141.2.1.0/255.255.248.0
options timeout:5 attempts:5 no-check-names
```

Listing E.1: Inhalt der Datei /etc/resolv.conf

- In Beispielen sind Benutzereingaben in fetter und Programmausgaben (insb. der Shell-Prompt $) in dünner Blockschrift gesetzt:

```
$ tail -n +0 -f /var/log/{syslog,messages} |grep named &
$ rndc reload
[...]named: loading configuration from '/etc/bind/named.conf'
[...]named: named.conf:72: unknown option 'allow-reKursion'
[...]named: reloading configuration failed: failure
$
```

Listing E.2: Fehler in der Konfigurationsdatei festgestellt

- Die Syntax von Anweisungen wird wie in Manpages üblich angegeben:
  `nslookup [-Optionen] (Hostname|IP) [Nameserver]`
  Hierbei können Sie alle Teile weglassen, die in eckigen Klammern stehen. Von den Alternativen, die in runden Klammern stehen und mit | getrennt sind, müssen Sie eine auswählen. Bitte geben Sie die Zeichen `()[]|` nicht mit ein! Die Wörter in Schrägschrift sind Platzhalter für Ihre Eingaben.
- Die Beschreibung häufiger Fehler, die unerfahrene Hostmaster immer wieder machen, ist grau hinterlegt und mit **»Achtung:«** gekennzeichnet:

> **Achtung:** Wenn das Verweisziel nicht mit einem Punkt endet, wird der Nameserver beim Einlesen des Zonefile den Namen der Zone anhängen.

- Verweise auf weiterführende Informationen in einem anderen Abschnitt oder im Internet sind grau hinterlegt und mit **»Mehr zum Thema:«** gekennzeichnet:

Mehr zum Thema: Die NOTIFY-Nachricht wird in RFC1996 definiert.

- In langen Tabellen sind die wichtigsten Optionen mit einem Ausrufezeichen markiert:

Anweisung	Beschreibung
`lwres`	konfiguriert die Zugriffsmöglichkeiten für die Lightweight Resolver Library.
`options`	konfiguriert die allgemeinen Eigenschaften des Nameservers.
`...`	...

Tabelle E.1: Anweisungen in `named.conf`

- Abkürzungen werden nur bei ihrer ersten Verwendung ausgeschrieben: »1983 stellte Paul Mockapetris in RFC882 das Domain Name System (DNS) vor. Ziel des DNS war die Ablösung der zentral gepflegten `HOSTS.TXT`-Datei.«
  Wenn Sie über eine unbekannte Abkürzung stolpern, weil Sie vielleicht das Buch nicht von Beginn gelesen haben, können Sie ihre Bedeutung im Abkürzungsverzeichnis nachschlagen, das Sie in Anhang D finden.

E.5 Die RFCs

An vielen Stellen finden Sie Verweise auf die Request-for-comment-(RFC-) Dokumente der Internet Engineering Task Force (IETF).

Die RFCs sind Textdateien, die detaillierte Beschreibungen zu Verfahren und Protokollen im Internet enthalten und von vielen Entwicklern als verbindliche Richtlinien betrachtet werden.

Sie können alle RFCs auf `http://www.ietf.org/rfc/` betrachten und auch als ZIP-Archiv herunterladen.

Mehr zum Thema: Eine Übersicht wichtiger RFCs zum Thema DNS finden Sie in Anhang C.

E.6 Beispiele

Die meisten Beispiele in diesem Buch wurden so gewählt, dass Sie sie am eigenen Rechner nachvollziehen können. Da sie im August 2004 erstellt wurden, kann es jedoch sein, dass die entsprechenden Zonefiles sich bis zu dem Zeitpunkt, an dem Sie dieses Buch lesen, geändert haben.

Damit Sie sich nicht laufend in neue Netze eindenken müssen, stammen die meisten Beispiele aus zwei (in ihrer Art sehr ergiebigen) Domains:

informatik.uni-frankfurt.de:

Diese Domain ist seit 1990 in Betrieb und hinreichend komplex: Sie enthält elf öffentlich zugängliche Nameserver, 15 Subdomains (zwölf Professuren und drei zur Verwaltung) und für jeden der 20 Gebäudeabschnitte ein eigenes, öffentliches IP-Netz.

Die Workstations und die zentralen Server befinden sich in der Subdomain `rbi`, was für »Rechnerbetriebsgruppe Informatik« steht. Darüber hinaus gibt es einige Aliase zwischen den Subdomains, die den Studierenden und Angestellten das Leben erleichtern.

ibm.com:

Die Firma IBM legt offensichtlich viel Wert auf die Verfügbarkeit ihrer Dienste: Die Nameserver und Mailserver befinden sich auf mehreren Kontinenten, und allein die Webserver, die die Startseite enthalten, verteilen sich auf sechs verschiedene Netze.

E.7 Feedback und Errata

Ich danke Ihnen für alle Kommentare, Ergänzungen und Verbesserungsvorschläge, die Sie bitte per E-Mail an `dnsbind@jfranken.de` senden.

Alle gemeldeten Fehler und deren Korrektur finden Sie auf

`http://www.jfranken.de/homepages/johannes/books/dnsbind.de.html`.

E.8 Danksagungen

Mein Dank geht an (alphabetisch sortiert)

- die fachlichen Korrektoren dieser Ausgabe:
  - Beate Franken, meine Frau, die sich seit 1996 mit Linux und dem Internet beschäftigt

- Dipl.-Inform. med. Tobias Häcker, der sich seit über zehn Jahren mit Internetprotokollen herumschlägt und seinem Namen als Security-Experte Ehre zu machen versucht
- Dipl.-Inform. Philipp Hahn, der an der Universität Oldenburg auf dem Gebiet Betriebssysteme und Netzwerke forscht und seit vielen Jahren mehrere (Name-)Server administriert
- Dipl.-Ing. Walter Kohl, bei dem ich schon 1989 viel über die Entwicklung von Netzwerk- und Datenbankanwendungen lernte
- Dipl.-Inform. Thomas Köhler, just another geek (JAG) mit großer Erfahrung im Unix- und Netzwerkbereich
- Dipl.-Phys. Michael Lestinsky, Wissenschaftler am Heidelberger Max-Planck-Institut für Kernphysik, der seit 1996 Unix-Software entwickelt und in der Unix-User-Group Rhein-Neckar aktiv ist
- Dipl.-Inform. Stefan Watermann, der die Entwicklung des Deutschen Internets seit vielen Jahren kennt und für internationale ISPs und Telcos arbeitete

- und die Helfer
  - Gunnar Jehle vom mitp-Verlag, der die Idee zu diesem Buch hatte
  - Dipl.-Inform. Gerhard Leuck, Hostmaster der Domain `informatik.uni-frankfurt.de`, aus der die meisten Beispiele stammen
  - Dipl-Ing. Heiko Nonnenmacher, der sich ausgezeichnet mit Windows 2000 und dem Active Directory Service auskennt
  - Tom Podchul, Network- und Security-Consultant, der mir viel über die Vorgänge im RIPE-NCC und der ICANN erzählen konnte
  - Horst Rother, ein lebendiger Weltatlas, der mir bei der Übersetzung der ccTLDs half
  - Dipl.-Inform. med. Thomas Uhl, Open-Source-Influencer der ersten Stunde, Technologie-Visionär und Vorstand der Topalis AG, der das Vorwort zu dieser Ausgabe schrieb
  - Robert Zakon, Autor der RFC2235, der mir die Nutzung seiner Internet-Timeline erlaubte

Ohne eure Hilfe wäre dieses Buch ein schlechteres.

Eschborn, im November 2004 *Johannes Franken*

Teil I

Konzepte und Einsatz von DNS und BIND

1 Das Domain Name System

1.1 Motivation

Rechner im Internet werden über ihre *IP-Adressen* angesprochen. Eine IP-Adresse ist eine natürliche Zahl zwischen 0 und 2^{32} (IPv4) oder 2^{128} (IPv6).

Anstatt die Adresse als Dezimalzahl zu notieren, verwendet man normalerweise die »Dotted Quad«-Notation, die durch Konvertieren der Adresse in das Binärsystem, Aufteilen in 8-Bit-Gruppen und Zurückkonvertieren der Gruppen in das Dezimalsystem entsteht:

`123457890` (Dezimalzahl)
= `0100.1001 1001.0110 0000.0010 1101.0010` (Binäre Darstellung)
entspricht `73.150.2.210` (Dotted Quad)

Anhand dieser Adresse werden die Datenpakete im Internet weitergeleitet, jedoch kann sich der menschliche Anwender diese Schreibweise schlecht merken. Deshalb soll der numerischen Adresse ein symbolischer Name zugeordnet werden, der das Kommunizieren der Adresse einfacher macht.

Daneben bietet dieser zusätzliche indirekte Ansatz den Vorteil, dass sich die Zuordnung von Namen zu Dienst bzw. Host für den Anwender unsichtbar (»transparent«) ändern lässt: Selbst wenn der Host mit dem symbolischen Namen `www.example.com` physisch umzieht und deshalb eine andere IP-Adresse bekommt, ist er weiterhin unter `www.example.com` erreichbar. Dabei können aber kurzfristig Fehler auftreten, bis sich die Änderung der Adresszuordnung herumgesprochen hat.

1.2 Entstehung

Nach der Erfindung des globalen Hostnamens durch Peggy Karp im Jahr 1971 (RFC226) mussten die Administratoren die Adressen aller Rechner, die am Advanced Research Projects Agency Network (ARPANET, dem Vorgänger des heutigen Internets) teilnehmen sollten, per E-Mail beim Network Informa-

tion Center am Stanford Research Institute (SRI-NIC) anmelden. Dort wurden sie von Hand in die zentrale HOSTS.TXT-Datei aufgenommen.

```
HOST : 10.0.0.73 : SRI-NIC,NIC : FOONLY-F3 : TENEX :
       NCP/TELNET,NCP/FTP, TCP/TELNET, TCP/FTP :
HOST : 10.2.0.52 : USC-ISIF,ISIF : DEC-1090T : TOPS20 :
       TCP/TELNET,TCP/SMTP,TCP/FTP,TCP/FINGER,UDP/TFTP :
```

Listing 1.1: HOSTS.TXT-Einträge aus dem Jahr 1983

Die neuen Rechner waren jedoch erst über ihre Namen erreichbar, nachdem *alle* anderen Rechner die nächste Ausgabe der HOSTS.TXT-Datei per FTP vom SRI-NIC bezogen hatten. Einige Systemadministratoren patchten die Datei nach jedem Download, um ihre inoffiziellen Aliase und geheimen Hosts hinzuzufügen.

Mit der Zahl der Rechner stiegen

- der Aufwand zur Pflege der HOSTS.TXT-Datei im SRI-NIC,
- die Belastung des Netzes durch die Downloads der Datei und
- die Wahrscheinlichkeit, dass ein Hostname doppelt vergeben wurde.

Die »Department of Defense Advanced Research Projects Agency« (DARPA) bat um Vorschläge zur Lösung der Probleme.

Im September 1981 schlug Dr. David Mills (Comsat Laboratories) in RFC799 die Einführung mehrerer »Domains« vor, um den Namensraum der Hostnamen zu erweitern.

Im August 1982 stellten Dr. Jon Postel von der University of Southern California (USC) und Zaw-Sing Su vom SRI in RFC819 das Konzept eines »Domain Name Service« vor: Alle Einträge werden in hierarchischen Organisationseinheiten (Domains) angeordnet. Innerhalb jeder Domain führt ein Administrator eine Liste seiner Rechner und betreibt einen *Nameserver*, der Anfragen nach Hostnamen oder IP-Adressen entweder selbst beantwortet oder an andere Nameserver weiterleitet.

Dr. Paul Mockapetris, der zu diesem Zeitpunkt ebenfalls an der USC tätig war, setzte dieses Konzept um: Er entwickelte den ersten Nameserver (»Jeeves«)

und veröffentlichte im November 1983 die eingesetzten Protokolle in RFC882 und RFC883 unter dem Titel »Domain Name System«.

Im Oktober 1984 veröffentlichten Jon Postel und Joyce Reynolds die Pläne für die neuen Top-Level-Domains (com, net, org, edu, mil, arpa) und die Umstellung aller Software auf DNS (RFC920, RFC921).

Im November 1987 legte Mockapetris mit RFC1034 und RFC1035 eine überarbeitete Spezifikation seines Domain Name System vor, die bis heute weltweit gültig ist, jedoch mehrmals erweitert wurde. Er beschreibt darin

- die Bestandteile des DNS (siehe Abschnitt 1.3) und
- die Ziele, an denen er sich beim Entwurf des DNS orientierte:
  - Das DNS bietet einen Namensraum, in dem jeder Eintrag eindeutig adressierbar ist. Das Format der Einträge und das Protokoll zur Abfrage sind exakt festgelegt.
  - Die hohe Zahl der Einträge und deren Aktualisierungen erfordern das Aufteilen der Datenbank auf verschiedene Nameserver.
  - Zwischenspeicher (Caches) verringern die Netzlast. Der Client bestimmt, ob er die Daten vom hauptverantwortlichen Nameserver oder lieber aus einem schnelleren, aber evtl. veralteten Zwischenspeicher abfragt.
  - Das Konzept der Klassen und Typen ermöglicht die Einführung neuer Protokolle und Eintragsarten, ohne dass man dazu die bestehenden Daten ändern müsste.
  - Die Integrität der Daten kann durch Signaturen sichergestellt werden, die lediglich für eine Transaktion gültig sind.
  - DNS ist auf allen Plattformen (vom PC bis zum Großrechner) einsetzbar.

Mehr zum Thema: Die Geschichte des Internets wird in Anhang A und die Entstehung von BIND in Kapitel 2 beschrieben.

1.3 Bestandteile

1.3.1 Zonen

Alle Einträge werden in *Zonen* eingeteilt, die jeweils einen organisatorischen Verantwortungsbereich umfassen. Lokale Systemadministratoren tragen die Rechner ihres Netzes in *Zonefiles* (Mockapetris nennt sie »Masterfiles«) ein, die auf einem ihrer Server liegen.

Mehr zum Thema: Details zu Zonen werden in Abschnitt 1.5 und das Erstellen von Zonefiles in Kapitel 7 beschrieben.

1.3.2 Nameserver

Eine spezielle »Nameserver«-Software beantwortet die Anfragen aus dem Netz, indem sie auf die vorhandenen Zonefiles zurückgreift oder andere Nameserver befragt. Deren Antworten werden im Cache für weitere, gleichlautende Anfragen gespeichert.

Es gibt verschiedene Arten (Betriebsmodi) von Nameservern:

- *Autoritative Nameserver* sind für eine oder mehrere Zonen hauptverantwortlich und geben definitive Auskunft zu den Inhalten der Zonefiles.
- *Recursive-Only Nameserver* leiten nur Anfragen an andere, zuständige Nameserver weiter und übermitteln die Antworten an die *Resolver* (siehe Abschnitt 1.3.3).
- *Caching Nameserver* unterscheiden sich von Recursive-Only-Nameservern nur darin, dass sie alle Ergebnisse einige Zeit zwischenspeichern, was die Bearbeitungszeit und die Netzlast bei späteren Abfragen bereits bekannter Einträge reduziert.

BIND beherrscht alle Betriebsmodi und kann diese sogar gleichzeitig einsetzen, was von Security-Experten als schwere Sicherheitslücke betrachtet wird.

Mehr zum Thema: Die Konfiguration und Administration des Nameservers »BIND« für jeden dieser Betriebsmodi wird in Teil B und C ausführlich beschrieben.

1.3.3 Resolver

Aus Sicht des Anwenders ist die *Resolver*-Funktion des Betriebssystems für die Namensauflösung zuständig.

Es gibt zwei Arten von Resolvern:

- *Stub-Resolver* senden rekursive Abfragen: Sie erwarten, dass der Nameserver ihnen die Antwort liefert.
- *(Full-)Resolver* senden iterative Abfragen: Sie folgen eigenständig den von Nameservern zurückgegebenen Verweisen auf andere Nameserver, bis sie eine definitive Antwort erhalten.

Die meisten heutigen Resolver sind *Stub-Resolver.*

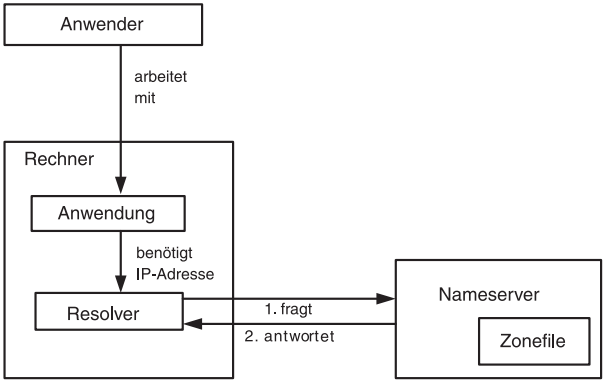

Abbildung 1.1: Resolver, Nameserver und Zonefiles (vereinfachte Darstellung)

Mehr zum Thema: Der Unterschied zwischen rekursiven und iterativen Abfragen besteht in dem RD-Flag, das im nachfolgenden Abschnitt 1.3.4 beschrieben wird. Die Auswirkung dieses Flags wird in Abschnitt 1.8 erklärt. Details zur Konfiguration verschiedener Resolver finden Sie in Kapitel 3.

1.3.4 Protokoll

Das DNS-Protokoll sieht vor, dass Nameserver und Resolver sich Nachrichten im *DNS-Nachrichtenformat* über UDP oder TCP schicken:

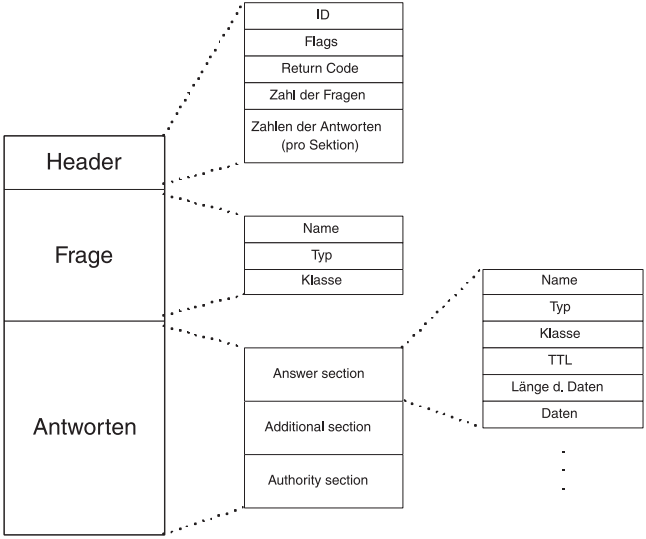

Abbildung 1.2: Das DNS-Nachrichtenformat

Eine DNS-Nachricht kann neben dem Header mehrere Fragen und verschiedene Arten von Antworten enthalten.

Flags

Der Header enthält neben der ID, dem Return-Code und verschiedenen Zahlen folgende Flags:

Flag	Beschreibung
QR	Das »Query-Response«-Flag gibt an, ob es sich bei der Nachricht um eine Frage (0) oder Antwort (1) handelt.
opcode	Eine Zahl zwischen 0 und 15, die die Art der Frage angibt: 0: Normale Abfrage (RFC1035) 1: Inverse Query (RFC1035) (obsolet seit RFC3425) 2: Server status Request (RFC1035) 4: Notify (Erweiterung aus RFC1996) 5: Dynamic Update (Erweiterung aus RFC2136)
AA	Der Nameserver kann das »Authoritative Answer«-Flag setzen: 0: Die Antwort stammt aus dem Cache. 1: Die Antwort stammt aus einem lokalen Zonefile.
TC	Der Nameserver muss das »Truncated«-Flag setzen, wenn die Nachricht zu lang für ein UDP-Datagramm war und daher nach 512 Byte abgeschnitten wurde.
RD	Der Client kann das »Recursion Desired«-Flag setzen: 0: Der Client gibt sich mit Antworten zufrieden, die nur Verweise auf andere Nameserver enthalten (Iterative Abfrage). 1: Der Client interessiert sich nur für das Endergebnis seiner Abfrage (Rekursive Abfrage). Ein Nameserver muss dieses Bit allerdings nicht beachten.
RA	Der Nameserver kann das »Recursion Available«-Flag setzen, um dem Client zu signalisieren, dass er rekursive Abfragen beantworten würde.
Z	Das »Zero«-Flag hat noch keine offizielle Bedeutung und sollte daher auch nicht gesetzt werden.
AD	Der Nameserver kann das »Authentic Data«-Flag setzen, um dem Client zu signalisieren, dass er die in der Answer- und Authority-Section enthaltenen Records mit DNSSEC geprüft hat und für echt befindet. Dies ist eine Erweiterung aus RFC2535.
CD	Der Client kann das »Checking Disabled«-Flag setzen, um einem Nameserver, der normalerweise DNSSEC einsetzt, zu signalisieren, dass er sich auch mit ungeprüften Antworten zufrieden gibt. Dies ist eine Erweiterung aus RFC2535.

Tabelle 1.1: Flags einer DNS-Nachricht

Antworten

Die Antwort setzt sich aus mehreren Teilen zusammen:

- Die *Answer*-Section kann eine oder mehrere Antworten auf die Frage enthalten,
- Die *Additional*-Section kann Transaktionssignaturen und zusätzliche Records, die weitere Abfragen vorwegnehmen (z.B. die von MX-, CNAME- und NS-Records referenzierten A-Records) enthalten.
- Die *Authority*-Section kann die Namen der zuständigen Nameserver enthalten.

Mehr zum Thema: Details zum Protokoll und zum Nachrichtenformat können Sie der RFC1035 und dem Buch *TCP/IP* von W. Richard Stevens (siehe Einleitung) entnehmen. Beispiele zur Namensauflösung finden Sie in Abschnitt 1.8.

1.3.5 Hostmaster

Die Aufgabe der *Hostmaster* ist es, die Zonefiles zu pflegen und sich um die Konfiguration der Nameserver und Resolver ihrer Zone zu kümmern:

- Die Konfiguration eines Nameservers erfordert die Angabe aller Zonen,
  - für die der Nameservernselbst hauptverantwortlich ist und
  - die der Nameserver von anderen Nameservern beziehen und zwischenspeichern soll.
- Zur Konfiguration des Resolvers zählt die Angabe aller Nameserver, die der Nameserver ansprechen soll.

Mehr zum Thema: Die Aufgaben eines Hostmasters werden in RFC1033 beschrieben.

1.4 Namensraum

1.4.1 Domains

Domains sind *Behälter* für Hostnamen und andere Einträge. Sie werden von lokalen Administratoren (den »Hostmastern«) gepflegt.

Da Domains neben Hostnamen auch Sub-Domains enthalten können, entsteht eine hierarchische Struktur, die man als *Baum* darstellen kann:

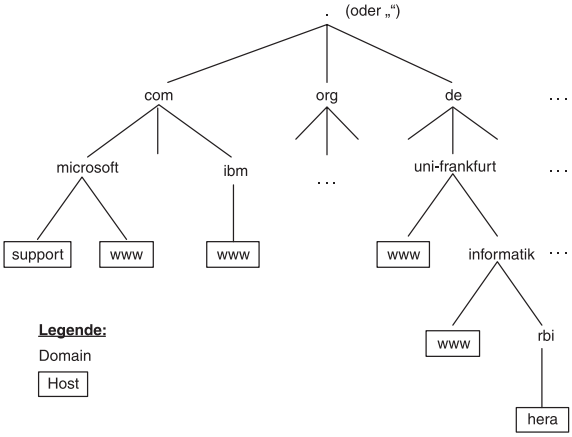

Abbildung 1.3: Domains und Hosts in einer Baumstruktur

Zum Beispiel enthält die `com`-Domain die Domains `microsoft` und `ibm`.

Der Wurzelknoten steht an oberster Stelle, wird als »Root« bezeichnet und oft als Punkt dargestellt.

1.4.2 Labels und Namen

Jeder Knoten der Baumstruktur

- ist mit einem *Label* beschriftet. Ein Label ist laut RFC2181 eine binäre Zeichenkette, die bis zu 63 beliebige 8-Bit-Zeichen (inkl. ASCII-0 und allen

Sonderzeichen) enthalten kann. Allerdings müssen sich alle Knoten, die denselben Elternknoten haben, selber durch ein eindeutiges Label voneinander unterscheiden.
- hat einen *Namen*, der sich aus den Labels auf dem Weg zum Wurzelknoten ergibt (siehe Abschnitt *Absolute Hostnamen*).
- enthält eine *Menge* (»Recordset«) von Einträgen (»Records«), die jeweils unterschiedliche Informationen über den Knoten enthalten können. Ein Beispiel finden Sie in Abschnitt 1.6.4.

Der Wurzelknoten hat den leeren String (das ASCII-Zeichen 0) als Namen.

Mehr zum Thema: RFC1178 enthält Tipps zum Erstellen *guter* Hostnamen.

Nameserver unterscheiden zurzeit noch nicht zwischen Groß- und Kleinbuchstaben, weshalb die folgenden Hostnamen (und weitere Varianten) identisch sind:

```
WWW.IBM.COM
WwW.iBM.Com
www.ibm.com
```

Listing 1.2: Identische Hostnamen

Da die Groß-/Klein-Schreibung bei einer Erweiterung des Alphabets (z.B. für chinesische Schriftzeichen) wesentlich werden kann und die Hostnamen in URLs traditionell kleingeschrieben werden, sollten Sie schon heute auf Großbuchstaben in Namen verzichten.

Beschränkungen für Host- und Domainnamen

Für die Namen von Hosts und Domains gelten jedoch zusätzliche Einschränkungen, die sich aus den Resolver-Implementierungen verschiedener Betriebssysteme ergeben. Diese Namen dürfen nur

- Buchstaben (A–Z, a–z),
- Ziffern (0–9) und
- Minuszeichen

enthalten und das erste und letzte Zeichen darf kein Minuszeichen sein. Die ursprüngliche Regelung, dass das erste Zeichen auch keine Ziffer sein darf, wurde mit RFC1123 aufgehoben.

Achtung: Host- und Domainnamen dürfen insbesondere keine Unterstriche (_), Umlaute und Sonderzeichen enthalten. Während einige Nameserver kein Problem mit den ungültigen Namen haben, ignorieren andere sie oder sperren sogar die gesamte Domain. Unter BIND können Sie das Verhalten mit der `check-names{}`-Option festlegen.

Es gibt verschiedene Versuche, diese Beschränkungen aufzuheben:

- Der bei Windows 2000 und Windows 2003 enthaltene DNS-Server ermöglicht Ihnen, Labels im UTF-8-Format zu kodieren.

  Achtung: Die UTF-8-Namen kann man nicht mit anderen Betriebssystemen ansprechen und selbst unter Windows kann man sie nicht zuverlässig über das Internet abfragen.

- Moderne Webbrowser beherrschen den »Internationalized Domain Names for Applications«(IDNA)-Algorithmus [RFC3490], der mit Hilfe der Algorithmen Stringprep [RFC3454], Nameprep [RFC3491] und Punycode [RFC3492] Umlaute im Host-Teil von URLs in gültige Host- oder Domainnamen umwandelt. Zum Beispiel würde ein Browser die URL `http://www.müller.de/` auf dem Webserver `www.xn--mller-kva.de` suchen. Dies stellt jedoch weder für das DNS noch den Resolver eine Erweiterung des Alphabets dar.

  Mehr zum Thema: IDNA wird auf `http://www.denic.de/de/domains/idns/index.html` beschrieben.

Absolute Hostnamen

Der *Fully Qualified Domain Name* (FQDN) oder *absolute Hostname* eines Hosts entsteht, indem Sie alle Labels vom Blatt (unten) bis zur Wurzel (oben) aufzählen und an Stelle aller Verbindungslinien (einschließlich der zur Wurzel) jeweils einen Punkt als Trennzeichen einsetzen. Korrekterweise enden daher alle FQDNs mit einem Punkt.

Der Baum in Abbildung 1.3 enthält also folgende FQDNs:

```
support.microsoft.com.
www.microsoft.com.
www.ibm.com.
www.uni-frankfurt.de.
www.informatik.uni-frankfurt.de.
hera.rbi.informatik.uni-frankfurt.de.
```

Listing 1.3: FQDNs

Ein FQDN darf einschließlich aller Punkte maximal 255 Zeichen lang sein.

Achtung: FQDNs enden immer auf einen Punkt.

Relative Hostnamen

Ein Host- oder Domainname wird *relativ* zu einer Domain, wenn man ein oder mehrere Labels am Ende weglässt. Man bezeichnet ihn dann auch als *Unqualified Domain Name* (UDN) oder *Relative Distinguished Name* (RDN).

Achtung: RDNs enden niemals auf einen Punkt.

Beispiel: Bei entsprechender Konfiguration des Resolvers kann man an Rechnern der Domain `informatik.uni-frankfurt.de` die relativen Hostnamen `hera` und `hera.rbi` auflösen.

Mehr zum Thema: Die `search`-Anweisung in `/etc/resolv.conf` wird in Abschnitt 3.1.1 beschrieben.

1.4.3 Top-Level-Domains

Man bezeichnet

- den Wurzelknoten ».« als *Root*-Domain
- die Domains der zweiten Ebene (z.B. `com`.) als *Top-Level-Domains* (TLD)
- unterhalb der dritten Ebene als *Subdomains* oder *Third-* oder *Fourth-Level*-Domains

Die Domains des in Abbildung 1.3 gezeigten Baumes kann man daher folgendermaßen einteilen:

Hostname	ggf. weitere Subdomains	(2nd-Level-) Domain	Top-Level-Domain
`support`		`microsoft`	`com`
`www`		`microsoft`	`com`
`www`		`ibm`	`com`
`www`		`uni-frankfurt`	`de`
`www`	`informatik`	`uni-frankfurt`	`de`
`hera`	`rbi.informatik`	`uni-frankfurt`	`de`

Tabelle 1.2: FQDNs, TLDs, Domains und Subdomains

Es gibt zurzeit 261 TLDs.

- Davon sind 246 TLDs für die Nationen vorgesehen (z.B. `de`.). Man bezeichnet sie als »country-code-Top-Level-Domains« (ccTLD). Die Country-Codes sind zwei Buchstaben lang und wurden 1981 in ISO-3166 festgelegt.
- Davon sind 14 TLDs für allgemeine Zwecke vorgesehen (z.B. `com`.). Man bezeichnet sie als »generic Top-Level-Domains« (gTLD).
- Davon wird eine TLD (`arpa`.) für die Verwaltung der Infrastruktur (IP-Adressen, Telefonnummern und Netzdienste) eingesetzt. Sie enthält verschiedene Zeiger auf Einträge in anderen TLDs.

Achtung: `arpa` steht nicht (mehr) für »Advanced Research Projects Agency«, sondern für »Address Routing Parameter Area«.

Der Eintrag neuer Top-Level-Domains erfolgt durch die Internet Corporation for Assigned Names and Numbers (ICANN) auf den *Root-Nameservern*. Informationen zum Betrieb der Root-Nameserver stehen in RFC2870.

Der Eintrag neuer Second-Level-Domains erfolgt durch die *Registry* der Top-Level-Domain auf den Nameservern der Top-Level-Domain.

Die größeren Registries haben die Verteilung der Second-Level-Domains an *Registrare* ausgelagert. Zum Beispiel hat der DeNIC e.V. die Anmeldung und Verwaltung der `de`.-Domains an seine Mitglieder (z.B. die DeNIC-direct GmbH, T-Online, Schlund, Joker) übertragen.

Mehr zum Thema: Den Hintergrund zur ICANN und dem Registry-System finden Sie in Anhang A. Listen aller TLDs und gTLD-Registries finden Sie in Anhang B.

1.4.4 Die in-addr.arpa-Domain

Mit dem in Abbildung 1.3 gezeigten Baum können Nameserver sehr effizient die IP-Adresse zu einem gegebenen Hostnamen ermitteln. Doch wie kann

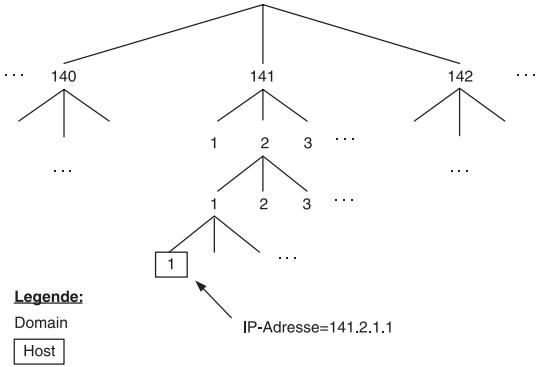

Abbildung 1.4: Eine IP-Adresse in der Baumstruktur

man den Hostnamen zu einer gegebenen IP-Adresse ermitteln? Damit die Nameserver hierzu nicht den gesamten Baum durchsuchen müssen, nimmt man jeden Host in einen zweiten Baum auf, der entlang der IP-Adressen angeordnet ist (siehe Abbildung 1.4).

Dieser Baum ist unter der Domain `in-addr.arpa` in das DNS eingehängt (siehe Abbildung 1.5).

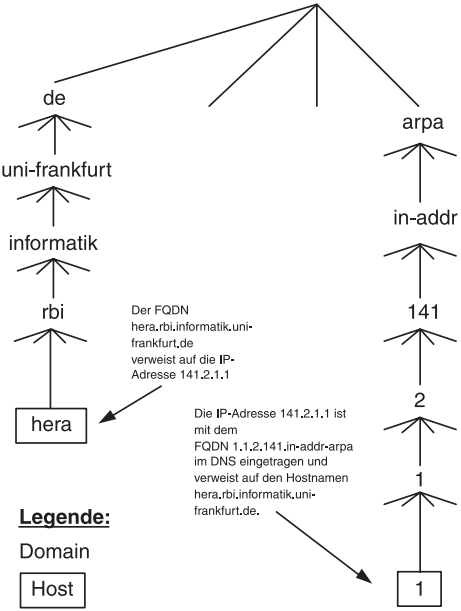

Abbildung 1.5: Anbindung der `in-addr.arpa`-Domain

Der FQDN zu der IP-Adresse `141.2.1.1` lautet konsequenterweise `1.1.2.141.in-addr.arpa.`.

Achtung: Bitte beachten Sie, dass die Bytes der IP-Adresse im FQDN in umgekehrter Reihenfolge erscheinen.

Neben den Zonefiles für `de.`, `uni-frankfurt.de.`, `informatik.uni-frankfurt.de.` und `rbi.informatik.uni-frankfurt.de.` müssen auch die Zonefiles für die Subnetze `141.0.0.0/8` (Zone `141.in-addr.arpa.`), `141.2.0.0/16` (Zone `2.141.in-addr.arpa.`) usw. verwaltet werden. Diese liegen normalerweise auf den für diese Netze zuständigen Nameservern.

Problematisch ist allerdings die Verwendung von Classless Inter-Domain Routing (CIDR), da die Grenzen zwischen Netzen dann nicht mehr anhand der 8-Bit-Gruppierung festgemacht werden. Abhilfe schafft die in RFC2317 beschriebene *Classless in-addr.arpa delegation*.

Mehr zum Thema: Beispiele zu Classless Delegationen finden Sie in Abschnitt 9.1.

1.4.5 Die ip6.arpa-Domain

Gemäß RFC3596 werden IPv6-Adressen unter `ip6.arpa` (früher: `ip6.int` laut RFC1886) eingetragen, indem man jedes Nibble (Hexadezimalziffer) als Domain behandelt.

Der FQDN für die IPv6-Adresse `4321:0:1:2:3:4:567:89ab` lautet demnach `b.a.9.8.7.6.5.0.4.0.0.0.3.0.0.0.2.0.0.0.1.0.0.0.0.0.0.0.1.2.3.4.ip6.arpa.` .

RFC2874 beschreibt die *Bitstring*-Methode, mit der man diese Adresse unter Verwendung von A6- und DNAME-Records noch flexibler als `\[x4321000000010002000300040056789ab/128].ip6.arpa.` oder `\[x000300040056789ab/64].\[b10/16].\[x432100000001/48].ip6.arpa.` delegieren kann. Diese Methode konnte sich jedoch nicht durchsetzen, weshalb die IETF sie im August 2002 per RFC3363 mitsamt des A6-Typs auf »experimentell« zurückstufte.

Achtung: Die BIND-Versionen 9.0 bis 9.2.3 beherrschen Bitstring-Labels; zur Version 9.3 hat das Internet Systems Consortium (ISC) den Bitstring-Support wieder aus dem BIND-Code entfernt.

1.4.6 Die e164.arpa-Domain

RFC2916 und RFC3245 erklären das E.164-number-(ENUM-)Verfahren, mit dem man aus einer Telefonnummer einen Domainnamen bildet: aus `+46-8-9761234` wird `4.3.2.1.6.7.9.8.6.4.e164.arpa.`.

Wenn Sie z.B. an einer solchen Stelle im DNS einen NAPTR-Record auf Ihre Homepage und Ihre E-Mail-Adresse verweisen lassen, könnten moderne Telefone den Angerufenen Ihre Homepage schon während des Klingelns anzeigen.

Mehr zum Thema: siehe `http://www.denic.de/de/enum/index.html`

1.4.7 Die uri.arpa- und urn.arpa-Domains

RFC3401 bis RFC3405 stellen das Dynamic Delegation Discovery System (DDNS) vor: Uniform Resource Identifier (URI) beschreiben die verschiedenen Positionen beliebiger Netzdienste (z.B. die Spiegel einer Webseite). Sie werden mit Hilfe von SRV- und Naming-Authority-Pointer-(NAPTR-)Einträgen im DNS unterhalb der Domains `urn.arpa.` und `uri.arpa.` angemeldet.

Mehr zum Thema: Der Unterschied zwischen URL, URN und URI wird in Abschnitt 7.5.26 (bei den NAPTR-Records) erklärt. Dort finden Sie auch Beispiele.

1.5 Zonen und Delegationen

Wie in Abschnitt 1.3.2 beschrieben, ist der gesamte DNS-Baum in Zonen unterteilt, die jeweils eine Domain und evtl. einige ihrer Subdomains umfassen.

Zu jeder Zone gibt es mindestens einen Nameserver (besser aber zwei oder mehr), die das Zonefile einlesen.

Ein Zonefile kann neben allen Einträgen der Zone zusätzlich Verweise zu anderen Zonen enthalten. Die Grenzen der Zonen ergeben sich durch *Delegation* von Subdomains an einen oder mehrere Nameserver.

> **Mehr zum Thema:** Beispiele zur Delegation finden Sie in Abschnitt 7.5.27 (bei den NS-Records).

Beispiele:

- Die Top-Level-Domain `de.` entstand dadurch, dass die Hostmaster der Root-Zone (»`.`«) Einträge in das Zonefile der Root-Zone schrieben, die die Subdomain `de.` an die Nameserver des DeNIC delegieren.

  > **Mehr zum Thema:** Details zu den Nameservern und Hostmastern der Root-Zone finden Sie in Anhang A und das Zonefile der Root-Zone auf `ftp://ftp.internic.net/domain/root.zone.gz`.

- Die Domain `uni-frankfurt.de.` entstand, indem die Hostmaster von DeNIC im Zonefile der `de.`-Zone ihre Subdomain `uni-frankfurt` an die Nameserver der Universität delegierten.
- Die Domain `informatik.uni-frankfurt.de.` ergibt sich durch Delegation der Subdomain `informatik` aus der Zone `uni-frankfurt.de.`.

Zonen können Einträge ihrer Subdomains (beliebiger Tiefe) enthalten.

Beispiele:

- In Abbildung 1.6 liegt der Host `www.cad` in der Zone `informatik`.
- Die Domain `uit.int` hat keine eigene Zone, weshalb der Eintrag `www.uit.int` direkt in der Zone `int.` liegt.

Zur besseren Übersichtlichkeit richten die meisten Hostmaster jedoch für jede Subdomain eine separate Zone ein.

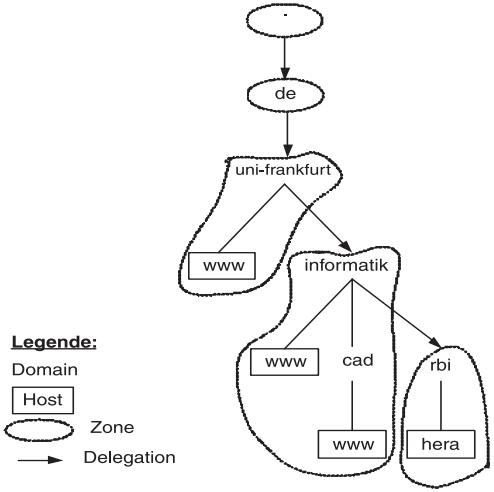

Abbildung 1.6: Zonen

1.6 Resource-Records

Jeder Eintrag im DNS-Baum besteht aus einem oder mehreren *Resource-Records* (RRs oder einfach nur »Records«), die jeweils eine Eigenschaft beschreiben. Bei Hosts kann dies z.B. eine IP-Adresse sein, bei Domains die zuständigen Nameserver. Man bezeichnet die Menge aller Records eines Eintrags als *Recordset* (RRset).

Mehr zum Thema: Ein Beispiel finden Sie in Abbildung 1.7, Abschnitt 1.6.4.

1.6.1 Felder

Jeder Resource-Record besteht aus fünf Feldern:

Feld	Beschreibung
Name	Die Position des Eintrags in der Baumstruktur, z.B. ein FQDN
TTL	(»Time to live«): gibt an, wie lange Resolver und Nameserver diesen Eintrag maximal zwischenspeichern dürfen. Die Einheit ist Sekunden. Ein Wert von 0 verhindert, dass der Eintrag zwischengespeichert wird.
Klasse	Klasse des Netzes, siehe nachfolgenden Abschnitt 1.6.2.
Typ	Art des Eintrags, siehe nachfolgenden Abschnitt 1.6.3.
Daten	Der Wert (z.B. bei Typ A die IP-Adresse). Das Format hängt von Typ und Klasse ab und wird in Kapitel 7 für jeden einzelnen Typ beschrieben.

Tabelle 1.3: Felder eines RR

1.6.2 Klassen

Die Klasse eines Records bezeichnet die Art des Netzes, in dem er enthalten ist. Dieses Feld erlaubt die Erweiterung des DNS für Zwecke außerhalb des Internets.

Jede Klasse »kennt« eine eigene Menge von Record-Typen und Adressen.

Aktuell sind folgende Klassen in Gebrauch:

Kürzel	Beschreibung
`IN`	Alle Netze, die das Internetprotokoll (IP) verwenden
`CH` oder `CHAOS`	Das CHAOSnet, ein am MIT eingesetztes LAN-Protokoll aus den 1970ern oder die Klasse, unter der BIND seine Versionsnummer, den Hostnamen, die System-ID und die Namen der BIND-Entwickler anbietet.
`HS`	Hesiod, ein Verzeichnisdienst (ähnlich NIS) aus dem Athena-Projekt des MIT

Tabelle 1.4: Record-Klassen

Zu Beginn des DNS gab es vorübergehend die Klasse `CS` für das CSNET, das kurz darauf jedoch im Internet aufging.

1.6.3 Typen

Das Typ-Feld gibt an, ob der Record einen Host oder eine Domain beschreibt und um welche Eigenschaft es sich handelt.

> **Mehr zum Thema:** Kapitel 7 enthält detaillierte Beschreibungen aller Record-Typen, die mit BIND möglich sind.

Die wichtigsten Record-Typen zur Beschreibung eines Hosts:

Typ	Verwendung
A	speichert die IPv4-Adresse(n) eines Hosts.
AAAA	speichert die IPv6-Adresse(n) eines Hosts.
CNAME	referenziert einen bereits vorhandenen Host, so dass dieser *zusätzlich* auch unter dem Namen dieses Eintrags bekannt ist (Alias).
PTR	verweist ebenfalls auf einen A- oder AAAA-Record, ist jedoch für Reverse queries (Fragen nach dem Host- oder Netznamen zu einer IP-Adresse) innerhalb der Domain `in-addr.arpa.` vorgesehen (siehe Abschnitt 1.4.4).

Tabelle 1.5: Record-Typen für Hosts

Die wichtigsten Record-Typen zur Beschreibung einer Domain:

Typ	Verwendung
MX	nennt einen Mail-Exchanger (A- oder AAAA-Record!) der Domain.
NS	nennt den Nameserver (A- oder AAAA-Record!) einer Zone. Dabei handelt es sich entweder um einen Authority- oder einen Delegation-Eintrag (siehe Abschnitt 1.5).
SOA	steht für »Start of Authority« und nennt den primären Nameserver, die E-Mail-Adresse seines Hostmasters sowie verschiedene Timeouts für die Kommunikation mit den sekundären Nameservern (siehe Abschnitt 1.9).

Tabelle 1.6: Record-Typen für Domains

Das DNS-Protokoll unterstützt die gezielte Abfrage nach bestimmten Typen von Informationen zu den Knoten. Neben den in Kapitel 7 aufgeführten Record-Typen gibt es noch neun weitere »Metatypen«, die selbst nicht als Record-Typ zum Eintragen verwendet, wohl aber bei Abfragen genutzt werden können:

Typ	Reaktion
ANY oder *	Abfrage aller Einträge des angegebenen Namens, also des gesamten RRsets.
AXFR	Der Nameserver sendet alle Einträge der Zone. Man nennt diese Abfrage »Zonetransfer«. Sekundäre Nameserver verwenden Zonetransfers, um ihre Zonefiles von den primären Nameservern zu beziehen.
IXFR	Abfrage aller Änderungen seit der übergebenen Versionsnummer. Man nennt diese Abfrage »inkrementellen Zonetransfer«. Moderne, sekundäre Nameserver verwenden inkrementelle Zonetransfers, um ihre Zonefiles auf den Stand des primären Nameservers zu bringen.
MAILA	Abfrage aller Records der Typen MD und MF (ist in BIND nicht mehr implementiert).
MAILB	Abfrage aller Records der Typen MB, MR, MG und MINFO (ist in BIND nicht mehr implementiert).
NULL	kann für eigene Testzwecke eingesetzt werden (siehe RFC1035).
OPT	wird zur Konfiguration der EDNS0-Optionen eingesetzt. EDNS0 ist eine Erweiterung des DNS-Protokolls und in RFC2671 beschrieben.
TKEY	wird zur Übertragung von TSIG-Schlüsseln verwendet (siehe RFC2930 und Abschnitt 12.2.1).
TSIG	wird zur Authentifizierung zwischen Nameservern verwendet (siehe RFC2845 und Kapitel 12.2.1).

Tabelle 1.7: Metatypen

1.6.4 Beispiel

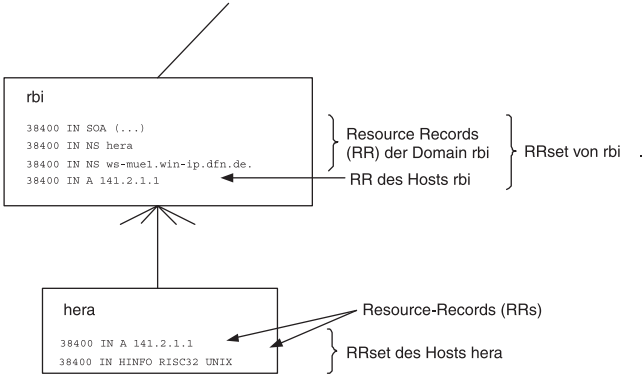

Abbildung 1.7: Records und Recordsets

Dieses Beispiel enthält eine Besonderheit. Sehen Sie in dem Recordset von `rbi` den Record `38400 IN A 141.2.1.1`? Er bewirkt, dass `rbi.informatik.uni-frankfurt.de` gleichzeitig Name einer Domain und eines Hosts ist, den man per Mail, ftp, www etc. erreichen kann.

1.7 Glue-Records

Wer eine Subdomain an einen Nameserver delegiert, der sich selbst in der Subdomain befindet, muss »Glue-Records« (Kopien der A-Records dieses Nameservers) in sein Zonefile aufnehmen.

Beispiel: Das Zonefile der Zone `informatik.uni-frankfurt.de.` enthält neben dem Record, der die Subdomain `rbi` an den Nameserver `hermes.rbi` delegiert, einen A-Record für `hermes.rbi`.

1.8 Iterative und rekursive Namensauflösung

Namensauflösung bezeichnet den Vorgang, bei dem ein Resolver die IP-Adresse eines Hostnamens ermittelt. Hierbei verbindet sich der Resolver mit einem der Nameserver, die im Betriebssystem konfiguriert sind.

> **Mehr zum Thema:** Details zur Konfiguration der Resolver verschiedener Betriebssysteme finden Sie in Kapitel 3.

Obwohl dieser Nameserver häufig beim Provider steht, wird er in diesem Buch als *eigener* Nameserver bezeichnet.

Die nächsten Schritte hängen von der Konfiguration des Nameservers ab:

- Der eigene Nameserver beantwortet die Anfrage *autoritativ*, wenn er selbst für die Zone zuständig ist, in der sich der angefragte Host- oder Domainname befindet.
- Der eigene Nameserver beantwortet die Anfrage *aus seinem Cache*, wenn er den gesuchten Record in der Vergangenheit schon einmal abgefragt hatte und seitdem die Lebensdauer »TTL« nicht überschritten wurde. Einige Nameserver-Implementierungen (z.B. BIND) halten ihren Cache im Arbeitsspeicher und verlieren ihn spätestens beim Beenden, während andere (z.B. Microsoft DNS) ihn in einem speziellen »Zonefile« speichern.
- Der eigene Nameserver kann sich *iterativ* auf die Suche begeben, indem er andere Nameserver fragt, die laut seinem Cache oder der Auskunft der Root-Nameservers eher zuständig sind (siehe Schritt 3, 5 und 7 in Abbildung 1.8). Er folgt den Verweisen, bis er auf einen Nameserver trifft, der die Anfrage autoritativ oder rekursiv beantwortet (siehe Schritt 8 in Abbildung 1.8).
- Der eigene Nameserver könnte die Abfrage auch *rekursiv* an einen anderen Nameserver (*Forwarder*) weiterleiten, der z.B. beim Provider steht. Die meisten Nameserver im Internet ignorieren jedoch das Recursion-Desired-Flag, um Überlastung zu vermeiden.

Im nachfolgenden Beispiel ist der eigene Nameserver so konfiguriert, dass er rekursiv die Anfrage des Resolvers auflöst. Er selber arbeitet dazu aber itera-

tiv und fragt sich selber von der Root-Zone aus angefangen bis zum autoritativen Nameserver der Zone `informatik.uni-frankfurt.de.` der Reihe nach durch.

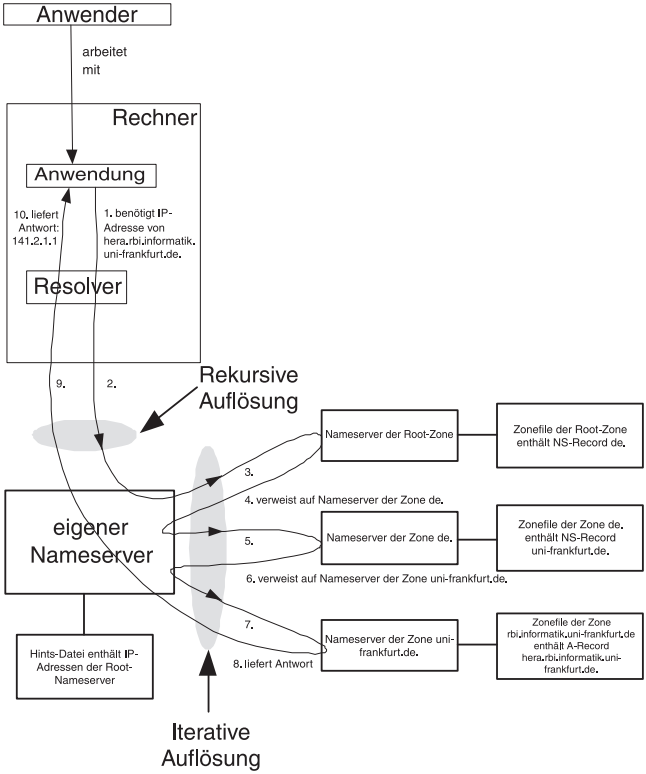

Abbildung 1.8: Rekursive und iterative Namensauflösung

1.9 Sekundäre Nameserver

Zum Schutz vor Ausfällen statten viele Hostmaster ihre Zone mit *mehreren* Nameservern aus, die gleichermaßen für die Zone zuständig (»autoritativ«) sind und von Resolvern und anderen Nameservern ebenfalls angesprochen werden. Einige TLDs nehmen nur Domains auf, die mindestens zwei Nameserver in unterschiedlichen Netzen aufweisen können.

Wenn eine Zone *n* Nameserver hat, verteilen Resolver und andere Nameserver die ersten *n* Abfragen zu dieser Zone gleichmäßig auf die vorhandenen Nameserver und sprechen anschließend nur noch den Nameserver an, der ihnen am schnellsten geantwortet hatte.

Wenn Änderungen innerhalb der Zone erforderlich sind (z.B. Hinzufügen eines neuen Eintrags), führt der Hostmaster folgende Schritte durch:

1. Er bearbeitet das Zonefile z.B. mit einem Texteditor oder einer grafischen Benutzungsoberfläche.
2. Er erhöht die Versionsnummer der Zone im SOA-Record und speichert das Zonefile.
3. Er teilt dem Nameserver (z.B. mit dem Befehl `rndc reload`) mit, dass er seine Zonefiles neu einlesen soll.

Damit die Hostmaster diese Schritte nicht auf jedem einzelnen Nameserver ausführen müssen, kennzeichnen sie die Zone auf einem der Nameserver als *primär* und auf allen weiteren als *sekundär*.

Achtung: Man bezeichnet primäre Zonen auch als »Master«- und sekundäre als »Slave«- Zonen.

Achtung: Da ein (physischer) Nameserver gleichzeitig primäre und sekundäre Zonen betreiben kann, darf man die Begriffe »primärer Nameserver« und »sekundärer Nameserver« nur in Bezug auf eine gegebene Zone verwenden.

Die sekundären Nameserver der Zone beziehen das Zonefile dann automatisch per *Zonetransfer* vom primären Nameserver der Zone. Wann und wie oft eine Zone neu übertragen werden wird, wird im SOA-Record festgelegt.

Mehr zum Thema: Details zur Konfiguration sekundärer Zonen finden Sie in Kapitel 7.5.38 (bei den SOA-Records) und in Abschnitt 6.3.11 (bei der `zone{}`-Anweisung).

1.9.1 Refresh-Queries

Die sekundären Nameserver einer Zone fragen den primären Nameserver dieser Zone in regelmäßigem Intervall (z.B. »alle drei Stunden«) nach dem SOA-Record dieser Zone. Wenn die darin enthaltene Versionsnummer seit der letzten Abfrage erhöht wurde, bringt der sekundäre Nameserver seine Zone mit einem Zonetransfer auf den aktuellen Stand. Das Intervall ergibt sich aus dem *Refresh Timeout*, das ebenfalls im SOA-Record der Zone enthalten ist.

Wenn ein sekundärer Nameserver den primären Nameserver seiner Zone nicht erreichen kann, ändert er das Intervall auf das im SOA-Record angegebene *Retry Timeout*. Dieses beträgt oft ein Drittel oder die Hälfte des Refresh-Intervalls.

Wenn der Kontakt trotz Retry-Intervall nicht innerhalb des Zeitraums zustande kommt, der im *Expire Timeout* des SOA-Records angegeben ist (z.B. zwei Wochen), gibt der Nameserver seine Zuständigkeit für diese Zone auf und verweist bei allen zukünftigen Anfragen auf die Root-Nameserver.

Mehr zum Thema: Die verschiedenen Timeouts werden in Abschnitt 7.5.38 (beim SOA-Record-Typ) und Zonetransfers in Abschnitt 1.9.3 beschrieben.

1.9.2 NOTIFY-Nachrichten

Lange Zeit war es schwierig, den richtigen Wert für die Refresh Timeouts zu finden:

- Zu große Werte verzögerten die Verbreitung von Aktualisierungen der Zone.
- Zu kleine Werte verursachten viel Last durch unnötige Anfragen.

Daher stellte Paul Vixie mit der BIND-Version 8 die *NOTIFY-Nachricht* vor:

Wenn der primäre Nameserver einer Zone startet oder feststellt, dass die Versionsnummer im Zonefile erhöht wurde, sendet er den SOA-Record per NOTIFY-Nachricht (opcode=4, qtype=SOA) an alle sekundären Nameserver der Zone.

Die sekundären Nameserver vergleichen den SOA-Record des primären Nameservers (aus der Notify-Nachricht) mit dem ihrer lokalen Kopie der Zone. Wenn die Versionsnummer auf dem primären Nameserver höher ist, aktualisieren sie ihr Zonefile per Zonetransfer.

Mehr zum Thema: Die NOTIFY-Nachricht wird in RFC1996 definiert. Abschnitt 6.3.7 beschreibt die `notify`- und `also-notify`-Optionen, mit denen Sie BIND die Adressen sekundärer Nameserver mitteilen.

1.9.3 Vollständige und inkrementelle Zonetransfers

Es gibt zwei verschiedene Arten von Zonetransfers:

- Der *vollständige* Zonetransfer ist die Antwort eines Nameservers auf eine AXFR-Abfrage. Dabei sendet er alle Records einer Zone in einem TCP-Datenstrom.
- Der *inkrementelle* Zonetransfer ist die Antwort eines Nameservers auf eine IXFR-Abfrage. Dabei sendet er alle Änderungen, die eine Zone seit der übergebenen Versionsnummer erfahren hat, in einem UDP-Datagramm oder einem TCP-Stream.
  Bei großen Zonen spart das Zeit und Netzbandbreite. Denken Sie z.B. an das Zonefile der `com`.-Domain, das mit seinen 30 Millionen Domain-Ein-

trägen (Stand: Juli 2004), die vielleicht durchschnittlich jeweils 100 Bytes belegen, mehrere Gigabytes umfasst.

Freilich kann ein Nameserver auf eine IXFR-Abfrage nur dann mit einem inkrementellen Zonetransfer antworten, wenn er die *Vorgeschichte* der Zone kennt. BIND führt hierzu für jede entsprechend konfigurierte Zone eine »Journal«-Datei, in der Dynamic Updates protokolliert werden, um bei inkrementellen Zonetransfers nur die relevanten Änderungen übertragen zu können. Wenn die Journal-Datei jedoch fehlt (z.B. wenn in der BIND-Konfigurationsdatei die Optionen `ixfr-from-differences`, `allow-update` oder `update-policy` fehlen), antwortet der Nameserver mit einem vollständigen Zonetransfer (AXFR).

Mehr zum Thema: Vollständige Zonetransfers werden in RFC1035, inkrementelle Zonetransfers in RFC1995 und Dynamic Updates im anschließenden Abschnitt 1.10 beschrieben.

1.10 Dynamic Updates

Einige Anwendungen (z.B. DHCP-Server, Windows-2000-Rechner und einige DNS-Admintools) möchten Records einer Zone verändern, löschen oder hinzufügen. Daher stellte Paul Vixie mit der BIND-Version 8 die *UPDATE-Nachricht* vor. Ab BIND 8.1 sind Dynamic Updates sogar die Voraussetzung für inkrementelle Zonetransfers, die bei großen Zonen erforderlich sind.

Wenn ein Nameserver von einem autorisierten Absender eine UPDATE-Nachricht für eine Zone erhält, die in seinem Zuständigkeitsbereich liegt, wird er diese wie folgt verarbeiten:

- Wenn er der *primäre* Nameserver für diese Zone ist,
  - führt er die in der UPDATE-Nachricht beschriebenen Änderungen (Löschungen und Neueinträge) aus,
  - aktualisiert das Zonefile und die Journal-Datei der Zone und
  - sendet ggf. NOTIFY-Nachrichten an die sekundären Nameserver.

- Wenn er ein *sekundärer* Nameserver für diese Zone ist,
  - leitet er die UPDATE-Nachricht an den primären Nameserver weiter, der im SOA-Record angegeben ist.

Achtung: Dynamic Updates sind ein Sicherheitsrisiko! Mit BIND können Sie den Zugriff auf einzelne Absender-IP-Adressen begrenzen und mit einem TSIG-Schlüssel authentifizieren.

Sie können Dynamic Updates mit dem Programm `nsupdate` erzeugen, das der BIND-Distribution beiliegt.

Mehr zum Thema:
Definition von Dynamic Updates: siehe RFC2136
`nsupdate`: siehe Abschnitt 11.3
Inkrementelle Zonetransfers: siehe Abschnitt 1.9.3

1.11 Dynamic DNS

»Dynamic DNS« (DynDNS) ist ein kostenloser Dienst der Firma Dynamic Network Services Inc. Er ermöglicht Rechnern, die IP-Adresse ihrer A-Records über http auf den Nameservern von Dynamic Networks selbst zu ändern. Dies ist sinnvoll, wenn Rechner bei jeder Einwahl ins Internet eine neue IP-Adresse erhalten, aber permanent unter demselben symbolischen Hostnamen erreichbar sein sollen.

Mehr zum Thema: siehe `http://www.dyndns.org/`

2 Nameserver im Überblick

2.1 BIND

2.1.1 Entstehung

Kaum hatte Dr. Paul Mockapetris 1983 den Nameserver »Jeeves« für DEC-1090-Systeme unter dem Betriebssystem TOPS-20 vorgestellt, begannen schon Studenten der Berkeley-Universität in Kalifornien, einen eigenen Nameserver für ihre VAX-Systeme unter BSD-Unix zu entwickeln: 1984 entstand der »Berkeley Internet Name Domain«(BIND)-Server. Mit der Unterstützung einiger Freiwilliger folgten viele Verbesserungen und Erweiterungen bis zur Version 4.8.3.

Von 1985 bis 1987 setzte ein Team um Kevin Dunlap – ein an der Universität tätiger Mitarbeiter der Firma DEC – die Arbeiten fort. Dunlap veröffentlichte die Releases 4.9 und 4.9.1 im Namen von DEC.

Mit Version 4.9.2 übernahm Paul Vixie – zunächst ebenfalls angestellt bei DEC, später selbstständig – die Projektleitung.

Um die Interessen und Spendengelder der großen Softwarefirmen (insb. UUNET und Network Solutions Inc.) in die Entwicklung einzubeziehen, gründete Vixie 1994 das Internet Software Consortium (ISC). Alle BIND-Versionen ab Version 4.9.3, die am 31. Dezember 1995 erschien, enthalten daher einen Copyright-Hinweis des ISC.

Die BIND-Versionen 5, 6 und 7 wurden übersprungen.

Im Mai 1997 stellte das ISC die BIND-Version 8.1 vor, die viele neue Fähigkeiten aufwies und wesentlich zuverlässiger funktionierte.

Schon bald war BIND so verbreitet, dass kein Weg mehr an den neuen Anforderungen (Security, Performance, Verfügbarkeit, IPv6, Views, DNSSEC) vorbeiführte. Inzwischen war jedoch der Sourcecode derart durchwachsen, dass

man ihn dazu fast vollständig neu schreiben musste. Das ISC gab den Auftrag an David Conrad, der im Vorstand von APNIC und ARIN saß. Mit der Unterstützung von Paul Vixie und dem Geld des ISC gründete Conrad die Firma Nominum Inc. und stellte Dr. Paul Mockapetris als Entwickler ein.

Im Februar 2000 veröffentlichte Nominum die erste Beta-Version von BIND9.

Im Jahr 2001 endete der Vertrag zwischen ISC und Nominum. Die folgenden Versionen (zumindest bis 9.3) wurden im Wesentlichen von dem australischen ISC-Mitarbeiter Mark Andrews erstellt.

Im Jahr 2004 nannte sich das ISC in »Internet Systems Consortium« um.

2.1.2 Versionsunterschiede

BIND ist aufgrund seiner gewachsenen Komplexität in Verruf geraten, fehleranfällig und unsicher zu sein. Deshalb sollen Sie regelmäßig nach Sicherheitsupdates schauen und diese auch einspielen.

Einige Anbieter liefern Rechner oder Betriebssystem-CDs mit veralteter BIND-Software und verweigern den Support, sobald der Kunde darauf »Fremdsoftware« einspielt. Insbesondere kleinere Unternehmen, die keine Betriebssystem-Spezialisten haben, müssen sich dann mit einer älteren Version aus dem Lieferumfang des Anbieters begnügen, mit allen damit verbundenen Sicherheitsproblemen. Die Sicherheitsupdates sollten von den Herstellern bereitgestellt werden, doch leider stellen nicht alle Hersteller alle Updates sofort zur Verfügung. Zum Beispiel hat das FreeBSD-Projekt im September 2004 damit begonnen, zum nächsten Release (5.3) auf die BIND-Version 9 umzustellen.

Achtung: Sie sollten aber Vorsicht walten lassen, bevor Sie von einer Version auf eine andere Major-Version updaten, da sich teilweise die Syntax der Konfiguration geändert hat und sogar Funktionen entfernt wurden (z.B. A6, Bitstring-Label). Testen Sie eine Neuinstallation möglichst zuvor im Parallelbetrieb zu Ihrer alten Version, bevor Sie komplett ohne funktionierendes DNS dastehen.

Mehr zum Thema: Anleitungen zum Update von den Versionen 4 oder 8 auf die Version 9 finden Sie in Abschnitt 10.4.

Wenn Sie Ihre BIND-Version selbst auswählen dürfen, können Sie sich an Abbildung 2.1 orientieren.

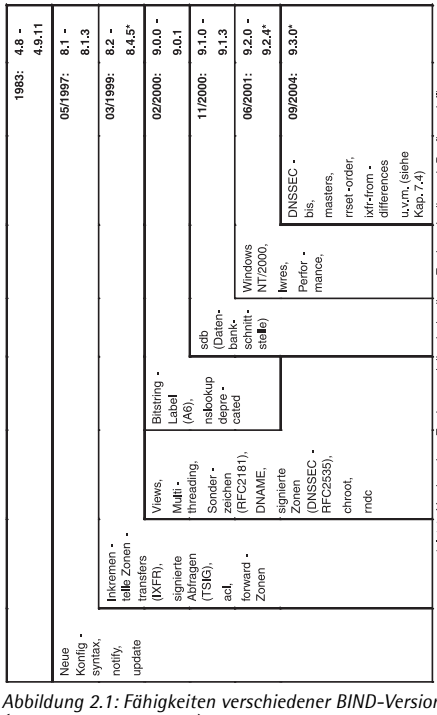

Abbildung 2.1: Fähigkeiten verschiedener BIND-Versionen (Stand: September 2004)

Bisher sind die Sicherheitslücken bekannt, die in Abb. 2.2 gezeigt werden:

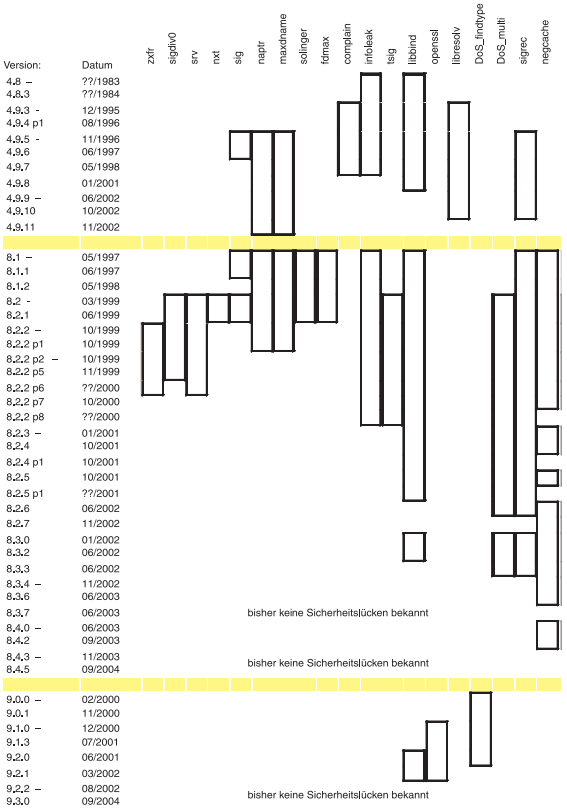

Abbildung 2.2: Sicherheitslücken der BIND-Versionen (Stand: September 2004)

Mehr zum Thema: Details zu den Sicherheitslücken finden Sie auf `http://www.isc.org/sw/bind/bind-security.php` .

Achtung: Zum Schutz vor Ausfällen sollten Sie Software, die Sie im Internet oder für unternehmenskritische Anwendungen einsetzen, grundsätzlich auf dem aktuellen Stand halten.

2.2 Alternativen zu BIND

Da die BIND-Versionen 4 und 8 in Bezug auf Geschwindigkeit und Verfügbarkeit viele Wünsche offen ließen, entstanden viele Alternativen zu BIND.

Die nachfolgenden Tabellen zeigt eine Auswahl alternativer Nameserver.

2.2.1 Open-Source-Projekte

Projekt	Beschreibung
djbdns/ tinydns	ist eine auf Sicherheit und Effizienz ausgelegte Implementierung eines DNS-Servers. Der Autor und berühmte Informatikprofessor Dr. Daniel Bernstein bietet auf seiner Webseite demjenigen 500 $ an, der ihm als Erster eine Sicherheitslücke in djbdns selbst nachweisen kann. **Weitere Infos:** `http://cr.yp.to/djbdns.html`
dnsmasq	ist ein einfacher DNS- und DHCP-Server für Linux, BSD und Mac OS/X, der die `/etc/hosts`-Datei als Zonefile einliest und darüber hinausgehende Abfragen an die in `/etc/resolv.conf` angegebenen Nameserver weiterleitet. **Weitere Infos:** `http://thekelleys.org.uk/dnsmasq/doc.html`
ldapdns	greift bei alle Anfragen live auf einen LDAP-Server zu. **Weitere Infos:** `http://sourceforge.net/projects/ldapdns/`
MaraDNS	ist ein einfacher Caching- und autoritativer Nameserver, der den Anspruch erhebt, besonders sicher zu sein. **Weitere Infos:** `http://www.maradns.org/`

Tabelle 2.1: Open-Source-Alternativen zu BIND

Projekt	Beschreibung
MyDNS	liest die Zonen direkt aus einer MySQL- oder PostgreSQL-Datenbank. **Weitere Infos:** `http://mydns.bboy.net/`
NSD	steht für »Name Server Daemon« und ist ein auf Performance ausgelegter Authoritative-Only-Nameserver für Unix-Systeme. **Weitere Infos:** `http://nlnetlabs.nl/nsd/`
pdnsd	ist ein »Proxy«-Nameserver für Linux, der alle Antworten im Filesystem speichert und daher speziell für den Offlinebetrieb von Notebooks und Dialin-Systemen geeignet ist. **Weitere Infos:** `http://www.phys.uu.nl/~rombouts/pdnsd/`
Posadis	arbeitet ebenfalls mit Zonefiles und ist für Linux, BSD und Windows erhältlich. **Weitere Infos:** `http://posadis.sourceforge.net/`
PowerDNS	ist ein auf Performance ausgelegter Authoritative-Only-Nameserver für Windows 2000/XP, Linux, FreeBSD und Mac OS/X, der bei Abfragen neben Zonefiles auch direkt auf verschiedene relationale Datenbanksysteme (Oracle, DB/2, MySQL, PostgreSQL, ODBC) oder ein LDAP-Verzeichnis zugreifen kann. **Weitere Infos:** `http://www.powerdns.com/`

Tabelle 2.1: Open-Source-Alternativen zu BIND (Fortsetzung)

2.2.2 Kommerzielle Produkte

Produkt	Beschreibung
Incognito DNS Commander	besteht aus einem performanten, hochverfügbaren Nameserver für Windows 2000/XP, Solaris oder Linux sowie einem Windows- und einem webbasierten Admintool. **Preis:** 495,- $ pro Nameserver zzgl. Wartungspauschale **Weitere Infos:** `http://www.incognito.com/products/DNSCommander/overview.asp`

Tabelle 2.2: Kommerzielle Alternativen zu BIND

Produkt	Beschreibung
Infoblox DNSone	ist eine Applicance (19-Zoll, 1HE), die ein angepasstes Linux, BIND, einen DHCP-Server und ein webbasiertes Managementtool enthält. **Preis:** bis 13.000,- € pro Gerät **Weitere Infos:** `http://www.infoblox.com/products/dnsone_overview.cfm` (Technik), `http://www.tlk.de` (Vertrieb)
Lucent VitalQIP	ist ein System zur Verwaltung großer Netze in einer Oracle- oder Sybase-Datenbank. Es enthält ein grafisches Admintool, mit dem Sie die Config-, Lease- und Zonefiles Ihrer DHCP- und DNS-Server aktualisieren können. Die mitgelieferten DNS-Server stammen von BIND9 ab und laufen unter Windows, Linux und Solaris. **Preis:** auf Anfrage **Weitere Infos:** `http://www.lucent.com/` (Technik), `http://www.n3k.de/` (Vertrieb)
Microsoft DNS-Server	ist der in Windows 2000/2003 enthaltene DNS-Server. Seine Konfiguration und die Zonefiles können mit einem grafischen Admintool bearbeitet werden. **Preis:** ab 250,- € **Weitere Infos:** `http://www.microsoft.com/`
Nominum Foundation ANS/CNS	Der »Authoritative Nameserver« (ANS) und der »Caching Nameserver« (CNS) sind auf hohe Performance ausgelegte Nameserver für Linux, FreeBSD und Solaris. Die Firma Nominum, bei der Paul Mockapetris inzwischen tätig ist, war zwar an der Entwicklung von BIND9 beteiligt, versichert jedoch, dass ihre Foundation-Produkte auf einer vollständig neuen Code-Basis entwickelt wurden. **Preis:** auf Anfrage **Weitere Infos:** `http://www.nominum.com/`
Simple DNS plus	besteht aus einem modernen Name- und DHCP-Server für Windows sowie mehreren grafischen und webbasierten Admintools. **Preis:** 90,- $ **Weitere Infos:** `http://www.jhsoft.com/`

Tabelle 2.2: Kommerzielle Alternativen zu BIND (Fortsetzung)

3 Resolver-Konfiguration

Viele Betriebssysteme enthalten Funktionen, die die Aufgabe des Resolvers erfüllen, also zwischen Hostnamen und IP-Adressen umwandeln. Entwickler von Netzwerkanwendungen können darauf zugreifen und müssen sich nicht mit den Details des DNS beschäftigen.

Einige Resolver-Implementierungen können sogar iterative Antworten verarbeiten, Ergebnisse zwischenspeichern oder andere Quellen (z.B. die Hosts-Datei, Berkeley-Datenbanken oder NIS) in die Suche einbeziehen.

3.1 Die GLIBC (Linux & Co.)

3.1.1 /etc/resolv.conf

Diese Textdatei enthält

- die IP-Adressen der Nameserver
- Domainvorschläge für relative Hostnamen
- evtl. Finetuning-Parameter

Jedes Programm liest sie bei der ersten Namensauflösung einmal ein.

Daher müssen Sie einige Programme (insb. Webbrowser und Dämonen) nach jeder Änderung der Datei `/etc/resolv.conf` neu starten.

Achtung: Der Neustart ist insbesondere auf *wandernden* Rechnern (z.B. Notebooks und Dialup-Systemen) erforderlich, die regelmäßig neue Nameserverkonfigurationen über DHCP oder PPP erhalten.

Beispiel:

```
# Diese Nameserver verwenden:
nameserver 141.2.1.1
nameserver 141.2.1.3
```

```
# Hostnamen ohne abschließenden Punkt sind
# relativ zu den folgenden Domains:
search informatik.uni-frankfurt.de uni-frankfurt.de

# Finetuning:
options ndots:2 attempts:3 timeout:2 rotate no-check-names
sortlist 10.0.0.0 192.168.0.0/255.255.0.0
```

Listing 3.1: Beispiel für /etc/resolv.conf

Die Datei /etc/resolv.conf kann folgende Anweisungen enthalten:

Anweisung	Erklärung
# und ;	sind die Kommentarzeichen. Der Resolver ignoriert alle Zeilen, die mit einem dieser Zeichen beginnen.
domain *Domainname*	gibt eine Domain an, die der Resolver automatisch an alle relativen Hostnamen anhängen soll. Verwenden Sie statt der domain-Anweisung besser die neuere search-Anweisung, die *mehrere* Domains anhängen kann. **Achtung:** Die Anweisungen search und domain schließen sich gegenseitig aus. Wenn Sie beide Anweisungen in der /etc/resolv.conf angeben, gilt nur die zuletzt genannte.
lwserver	gibt die IP-Adresse und den Port des nächsten Lightweight Resolver Daemon (lwresd) an. Der lwresd wird in Abschnitt 3.2 beschrieben.
! nameserver *IP-Adresse*	gibt die IP-Adresse eines Nameservers an, den der Resolver ansprechen soll. Wenn Sie mehrere Nameserver nutzen möchten, können Sie bis zu drei nameserver-Anweisungen angeben. Der Resolver spricht dann zwar weiterhin den ersten Nameserver an. Wenn dieser nicht antwortet, sendet er die Anfrage jedoch an den zweiten und notfalls den dritten Nameserver. Wenn Sie die rotate-Option (s.u.) setzen, wird der Resolver bei folgenden Aufrufen die Nameserver in anderer Reihenfolge ansprechen.

Tabelle 3.1: Anweisungen der /etc/resolv.conf

Anweisung	Erklärung
	Geben Sie die IP-Adresse `127.0.0.1` an, wenn Sie auf den lokalen Nameserver zugreifen möchten. Dies entspricht dem Verhalten, wenn die Datei `/etc/resolv.conf` nicht existiert oder *keine* `nameserver`-Anweisung enthält.
`options attempts:` *`Versuche`*	gibt an, wie oft der Resolver jeden Nameserver maximal ansprechen darf, wenn keine Antwort kommt. **Standardwert:** `2`.
`options debug`	bewirkt die Ausgabe vieler Zwischenberichte auf die Konsole bei jeder Namensauflösung.
`options ndots:`*`n`*	gibt an, wie viele Punkte ein Hostname mindestens enthalten muss, damit der Resolver die Auflösung zuerst ohne angehängte Suchlistendomains (siehe `domain`- und `search`-Anweisungen) probiert. **Standardwert:** `1`
`options no-check-names`	verhindert, dass der Resolver die Auflösung für *ungültige* Namen verweigert. Gültige Namen setzen sich aus Buchstaben (A–Z, a–z), Ziffern (0–9) und Minuszeichen zusammen und beginnen oder enden nicht mit einem Minuszeichen.
`options rotate`	bewirkt, dass der Resolver die Nameserver bei jeder weiteren Auflösung eines Programms in einer anderen Reihenfolge anspricht und so die Last auf sie verteilt.
`options timeout:` *`Sekunden`*	gibt an, wie viele Sekunden der Resolver in der ersten Runde maximal auf eine Antwort von jedem Nameserver warten soll. Ab der zweiten Runde verdoppelt er diesen Wert und teilt ihn durch die Zahl der Nameserver. **Standardwert:** `5`
`search` *`Suchliste`*	sollte an Stelle der `domain`-Anweisung verwendet werden und gibt eine Liste der Domains an, die der Resolver auf der Suche nach einem DNS-Eintrag an alle relativen Hostnamen anhängen soll. Die Suche endet beim ersten Treffer.

Tabelle 3.1: Anweisungen der `/etc/resolv.conf` (Fortsetzung)

Anweisung	Erklärung
	Wenn Sie mehrere Domains angeben möchten, trennen Sie diese bitte mit Leerzeichen oder Tabulatorzeichen. Die gesamte Suchliste darf maximal sechs Domains enthalten und nicht länger als 255 Zeichen sein. Der Resolver hängt die Domains in der Reihenfolge an, die Sie in der Suchliste vorgeben. Entlasten Sie also Ihr Netz und fremde Nameserver, indem Sie lokalere Domains an den Anfang der Suchliste stellen! Wenn der gesuchte Name mehr Punkte enthält, als mit der `ndots`-Option angegeben ist, versucht der Resolver zuerst, den Namen *ohne* angehängte Suchlistendomains aufzulösen. Sonst probiert er dies zuletzt. Wenn die Datei `/etc/resolv.conf` weder eine `search`- noch eine `domain`-Anweisung enthält, verwendet der Resolver als Suchliste den lokalen Domainnamen, der sich von dem lokalen Hostnamen ableitet, falls Sie diesen als FQDN angegeben hatten. Der lokale Hostname wird beim Systemstart aus der Datei `/etc/hostname` (Debian), `/etc/nodename` (Solaris) oder `/etc/sysconfig/hostname` (Red Hat) gelesen und dem Kernel mit dem `hostname`-Befehl mitgeteilt.
`sortlist` *`Netzliste`*	hilft der Anwendung bei der Auswahl der IP-Adresse, wenn ein RRset mehrere A-Records enthält. Der geschickte Einsatz dieser Option kann Netzverkehr reduzieren, da viele Anwendungen nur die erste Adresse derartiger Antworten beachten. Der Resolver sortiert die IP-Adressen nach der Reihenfolge der angegebenen Netze und stellt alle nicht enthaltenen IP-Adressen an das Ende der Liste. Die Netze müssen mit Leerzeichen getrennt und im Format *`IP-Adresse`*`/`*`Netzmaske`* angegeben werden. Wenn Sie die Netzmaske weglassen, geht der Resolver von der natürlichen Netzmaske dieses Netzes (Klasse A bis E) aus. Die Netzliste darf maximal zehn Netze enthalten.

Tabelle 3.1: Anweisungen der `/etc/resolv.conf` (Fortsetzung)

3.1.2 Umgebungsvariablen

Wenn Sie die folgenden Umgebungsvariablen setzen, werden alle Programme, die gegen die glibc gelinkt sind, von den Vorgaben der `/etc/resolv.conf` abweichen:

Variable	Auswirkung
`LOCALDOMAIN`	überschreibt die Suchliste, die in der `/etc/resolv.conf` mit der `search`-Anweisung festgelegt wurde, z.B. `$ export LOCALDOMAIN="rbi rbi.informatik"`
`RES_OPTIONS`	überschreibt die Optionen, die in der `/etc/resolv.conf` angegeben wurden, z.B. `$ export RES_OPTIONS="timeout:1 attempts:10 debug"`
`HOSTALIASES`	legt den Dateinamen der `hosts.aliases`-Datei fest. Lesen Sie mehr dazu in Abschnitt 3.1.6.

Tabelle 3.2: Umgebungsvariablen der Resolver-Bibliothek

3.1.3 /etc/nsswitch.conf

Ursprünglich stand die gesamte Konfiguration eines Rechners in verschiedenen, lokalen Dateien, z.B. `/etc/hosts`, `/etc/passwd`, `/etc/group` und `/etc/services`. Betriebssystemfunktionen wie `gethostbyname()` schauten einfach in die entsprechenden Dateien. Zur Einführung der Verzeichnisdienste (DNS, NIS (yellow pages), NIS+ und Hesiod) wurden viele Betriebssystemfunktionen so umgeschrieben, dass sie z.B. erst in die Datei `/etc/hosts` schauten, anschließend das NIS und zuletzt das DNS befragten. Da einige Funktionen die Quellen in anderen Reihenfolgen durchsuchten, war es zeitweise schwierig abzuschätzen, aus welcher Quelle eine Funktion ihre Daten bezieht.

Die Name-Service-Switch-(NSS-)Bibliothek löst dieses Problem, indem sie den Betriebssystemfunktionen eine gemeinsame Schnittstelle zu allen Verzeichnisdiensten und anderen Quellen bietet, die zentral über die Datei `/etc/nsswitch.conf` konfigurierbar ist. Fast alle Betriebssystemfunktionen, die Informationen auflösen müssen, greifen auf diese Bibliothek zu.

```
# Hostnamen:
hosts:     files dns
```

```
networks:  files

# User:
passwd:    compat
group:     compat
shadow:    compat
netgroup:  nis

# TCP/IP Konfiguration:
protocols: db files
services:  db files
ethers:    db files
rpc:       db files
```

Listing 3.2: /etc/nsswitch.conf

Diese Konfiguration bewirkt z.B., dass die NSS-Bibliothek zur Auflösung von symbolischen Namen zu Netzadressen zuerst die Datei `/etc/hosts` nutzt und sich nur an das DNS wendet, wenn sie den gesuchten Eintrag dort nicht findet.

Mehr zum Thema: Details zu NSS finden Sie in der Manpage zu `nsswitch.conf` und mit `info libc "Name Service Switch"`.

3.1.4 /etc/host.conf

Die Datei `/etc/host.conf` konfiguriert die resolv+-Bibliothek. Insbesondere beschreibt sie die Reihenfolge, mit der der Resolver das DNS, die Datei `/etc/hosts` und das NIS bei jeder Namensauflösung konsultiert.

Beispiel:

```
order       hosts,bind,nis
multi       on
nospoof     on
```

```
spoofalert   off
reorder      off
```

Listing 3.3: /etc/host.conf

Die resolv+-Bibliothek ist immer noch in der glibc enthalten, obwohl sie funktional von den NSS-Modulen abgelöst wurde. Wenn auf Ihrem Rechner die Datei `/etc/nsswitch.conf` existiert, können Sie die Datei `/etc/host.conf` bedenkenlos löschen.

Mehr zum Thema: Details zur resolv+-Bibliothek finden Sie in der Manpage zu `host.conf`, `resolver` und `resolv+`.

3.1.5 /etc/hosts

Die Datei `/etc/hosts` stellt die einfachste Zuordnung zwischen IP-Adressen und Hostnamen dar und eignet sich zum lokalen Hinzufügen von Namenseinträgen, etwa im privaten Zwei-Rechner-Heimnetz, in dem kein Nameserver dauerhaft verfügbar ist.

```
# IP-Adr.    Hostname        Aliase
127.0.0.1    localhost       loopback localhost.localdomain
141.2.1.1    hera            hera.rbi
141.2.1.2    loghost

# Einträge für IPv6:
::1          ip6-localhost   ip6-loopback
fe00::0      ip6-localnet
ff00::0      ip6-mcastprefix
ff00::1      ip6-allnodes
ff00::2      ip6-allrouters
ff00::3      ip6-allhosts
```

Listing 3.4: /etc/hosts

Diese Datei wird ausgewertet, wenn die `hosts`-Sektion in `nsswitch.conf` einen Eintrag `files` enthält.

3.1.6 /etc/hosts.aliases

Jeder User kann eigene Aliase für Hostnamen in einer Textdatei notieren. Wenn er die Umgebungsvariable `HOSTALIASES` auf den Namen dieser Datei setzt, werden alle Programme, die zur Namensauflösung Funktionen der `libresolv` aufrufen (z.B. die Programme `ssh`, `host` und `telnet`), Hostnamen zuerst in dieser Datei umsetzen, bevor sie sich an den Nameserver wenden.

Achtung: Programme, die die Namensauflösung selbst erledigen (z.B. `ping` und `dig`) beachten die `HOSTALIASES`-Umgebungsvariable nicht.

Beispiel:

```
$ cat ~/hosts.aliases
dio         diokles.rbi
mathe       watson.rz.uni-frankfurt.de.
physik      achso.physik.uni-frankfurt.de.
$ export HOSTALIASES=~/hosts.aliases
$ host dio
diokles.rbi.informatik.uni-frankfurt.de A       141.2.20.7
diokles.rbi.informatik.uni-frankfurt.de A       141.2.18.1
$ telnet dio
Trying...
Connected to diokles.rbi.informatik.uni-frankfurt.de.
Escape character is '^]'.

HP-UX diokles B.11.11 U 9000/735 (ttyp1)

login:
```

Listing 3.5: Host-Aliases verwenden

Der Systemadministrator kann den Wert der `HOSTALIASES`-Variablen für alle User einstellen, z.B. in der Datei `/etc/profile` auf den Wert `/etc/hosts.aliases`.

3.2 Die Lightweight-Resolver-Bibliothek

Die Lightweight-Resolver-Bibliothek (`liblwres`) ist erst seit Version 9 in der BIND-Distribution und daher noch nicht in der aktuellen glibc enthalten (Stand: glibc 2.2.5, Juli 2004).
Im Gegensatz zum Standard-Resolver kann sie

- iterative Antworten (siehe Abschnitt 1.8) verarbeiten
- DNAME-Verweise (siehe Abschnitt 7.5.10) auflösen
- gleichzeitig IPv4- und IPv6-Adressen liefern
- die Ergebnisse anwendungsübergreifend zwischenspeichern

Dabei greift sie nicht direkt, sondern über den *Lightweight Resolver Daemon* (`lwresd`) auf die Nameserver zu.

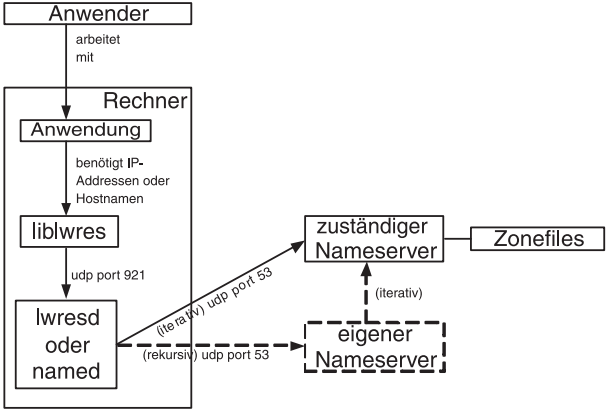

Abbildung 3.1: Kommunikation über die Lightweight-Resolver-Bibliothek

3.2.1 Konfiguration der liblwres

Alle Prozesse, die die `liblwres` verwenden, kommunizieren über UDP im »lightweight resolver protocol« mit dem `lwresd` oder einem lokalen BIND-`named`.

Unter Unix ist `lwresd` meist ein Symlink auf `named`, denn das Programm erkennt automatisch, über welchen Namen es aufgerufen wurde, und verhält sich entsprechend.

Die Adresse und den Port, auf dem die `liblwres` den `lwresd` anspricht, können Sie mit der `lwserver`-Anweisung in `/etc/resolv.conf` festlegen. Der Standardwert ist `127.0.0.1:921`.

3.2.2 Konfiguration des lwresd oder named

Der `lwresd` sucht seine Konfiguration in der Datei `/etc/bind/lwresd.conf`. Wenn diese Datei nicht existiert, verwendet er die in `/etc/resolv.conf` angegebenen Nameserver als Forwarder.

Alternativ zum `lwresd` können Sie den BIND-Nameserver (`named`) einsetzen. Er übernimmt die Aufgabe des `lwresd`, wenn er in seiner Konfigurationsdatei (`/etc/bind/named.conf`) eine `lwres`-Anweisung findet.

> **Mehr zum Thema:** Das Format der `lwresd.conf`-Datei entspricht dem der `lwres`-Anweisung, die in Abschnitt 6.3.6 beschrieben wird.

3.3 Windows

Dieser Abschnitt bezieht sich auf Microsoft Windows 2000, 2003 und XP.

3.3.1 Konfiguration

Windows setzt die DNS-Konfiguration aus schnittstellenübergreifenden und -spezifischen Einträgen der Registry zusammen:

```
[HKLM\SYSTEM\CurrentControlSet\Services\Tcpip\Parameters]
"NV Hostname"="terra"
"DataBasePath"="%SystemRoot%\System32\drivers\etc"
"NameServer"=""
"Domain"=""
"Hostname"="terra"
"SearchList"="rbi.informatik.uni-frankfurt.de,
    informatik.uni-frankfurt.de,uni-frankfurt.de"
"UseDomainNameDevolution"=dword:00000001
```

```
"AllowUnqualifiedQuery"=dword:00000000
"PrioritizeRecordData"=dword:00000001
"NV Domain"=""
...
```

Listing 3.6: Schnittstellenübergreifende DNS-Einstellungen in der Windows-Registry

```
[HKLM\SYSTEM\CurrentControlSet\Services\Tcpip\Parameters
    \Interfaces\{95851E06-2447-4474-A67B-F9BAABCCD83D}]
"EnableDHCP"=dword:00000001
"NameServer"="141.2.1.1,141.2.1.3"
"Domain"=""
...
```

Listing 3.7: DNS-Einstellungen einer Netzwerkschnittstelle in der Windows-Registry

Sie können diese Registry-Einträge mit dem in Windows enthaltenen Programm `regedit` betrachten und auch ändern. Die Änderungen werden sofort wirksam.

Sie können einen Teil dieser Eigenschaften auch mit einem grafischen Konfigurationstool ändern. Hierzu

1. rufen Sie die Netzwerkverbindungen auf (z.B. mit ⊞+R `control ncpa.cpl` ↵),
2. klicken doppelt auf eine Ihrer Netzwerkschnittstellen,
3. wählen das `Internetprotokoll` aus,
4. klicken auf die Buttons `Eigenschaften` und
5. `Erweitert`, und schließlich
6. auf die Lasche `DNS`.

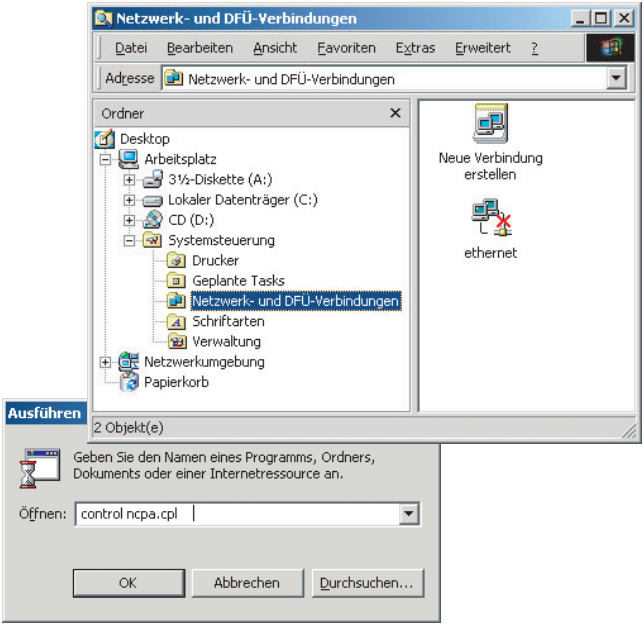

Abbildung 3.2: DNS-Konfiguration unter Windows (Schritte 1 und 2)

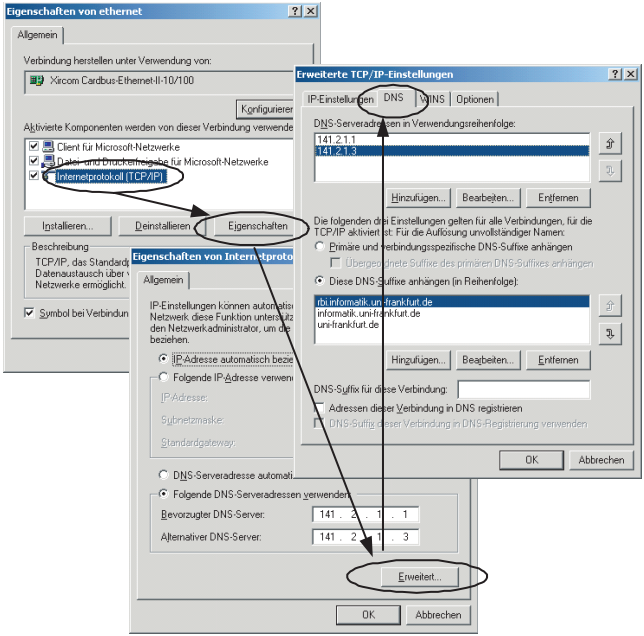

Abbildung 3.3: DNS-Konfiguration (Schritte 3 bis 6)

3.3.2 Admintools

Das Programm `ipconfig` ermöglicht folgende Funktionen:

Parameter	Auswirkung
`/registerdns`	sendet den lokalen Hostnamen und die IP-Adresse als Dynamic Update an den Nameserver. Dies ist höchstens bei DHCP-Konfiguration erforderlich, da Windows 2000 das Update bei statischer IP-Konfiguration automatisch bei jedem Booten und alle 24 Stunden sendet.
`/displaydns`	zeigt den Inhalt des Resolver-Cache. Der Resolver-Cache enthält alle DNS-Antworten, deren TTL noch nicht abgelaufen ist. Sie können den Resolver-Cache mit `net start dnscache` aktivieren und mit `net stop dnscache` deaktivieren.
`/flushdns`	löscht alle Records aus dem Resolver-Cache.

Tabelle 3.3: `ipconfig`-Funktionen

4 Abfragetools

Die BIND-Distribution enthält verschiedene Programme, mit denen Sie DNS-Abfragen durchführen können, z.B.

- auf der Shell (z.B. bei der Fehlersuche)
- in einem Shellscript
- aus einem anderen Programm heraus.

4.1 nslookup

`nslookup` ist das älteste und bekannteste Programm zur Abfrage des DNS. Es liegt der BIND-Distribution und fast allen Betriebssystemen (z.B. sämtlichen Windows- und Solaris-Versionen) bei.

Vorübergehend empfahl das ISC, `dig` oder `host` an Stelle von `nslookup` einzusetzen. Die `nslookup`-Versionen 9.0.0 bis 9.2.4rc7 und 9.3.0beta1 bis 9.3.0rc3 weisen darauf hin, wenn man sie ohne `-silent` aufruft:

```
$ nslookup
Note:  nslookup is deprecated and may be removed from
future releases. Consider using the `dig' or `host'
programs instead.  Run nslookup with the `-sil[ent]' option
to prevent this message from appearing.
> exit
$
```

Listing 4.1: `nslookup` verweist auf `host` und `dig`.

Spätere Versionen (z.B. 9.2.4 und 9.3.0) enthalten diesen Hinweis nicht mehr.

4.1.1 Aufruf

`nslookup` bietet zwei Betriebsmodi: Sie können `nslookup`

- *interaktiv* bedienen:
  `nslookup [-Optionen] [- Nameserver]`

`nslookup` gibt einen Prompt aus, an dem Sie Kommandos der folgenden Tabelle eingeben können. Beenden Sie die `nslookup`-Sitzung mit `exit`.

> **Achtung:** Wenn Sie für den interaktiven Modus einen Nameserver angeben möchten, beachten Sie bitte das Leerzeichen hinter dem Minuszeichen.

- *im Befehlszeilen-Modus* aufrufen:
  `nslookup [-Optionen] (Hostname|IP) [Nameserver]`
  `nslookup` ermittelt die Daten zu dem übergebenen Hostnamen oder der IP-Adresse und beendet sich automatisch.

4.1.2 Optionen

Sie können das Verhalten von `nslookup` mit verschiedenen Optionen steuern, die Sie

1. im interaktiven Modus als Kommando (`set Option=Wert`),
2. beim `nslookup`-Aufruf (`-Option=Wert`) oder
3. in der Datei `~/.nslookuprc`

angeben.

Wenn Optionen *mehrfach* gesetzt werden, haben Angaben im interaktiven Modus Vorrang vor Kommandozeilenoptionen, die wiederum Vorrang vor Angaben in `~/.nslookuprc` haben.

Einige Optionen erwarten, dass man ihnen eine Zeichenkette oder eine Zahl zuweist (z.B. `set q=MX`), andere Optionen erwarten keinen Wert (z.B. `set norecurse`).

`nslookup` kennt folgende Optionen:

Option	Beschreibung
`cl[ass]=(IN|CHAOS|HESIOD|ANY)`	gibt an, in welcher Klasse `nslookup` suchen soll. **Standardwert:** `IN`

Tabelle 4.1: `nslookup`-Optionen

Option	Beschreibung
[no]deb[ug]	schaltet zusätzliche Debug-Ausgaben an oder aus. **Standardwert:** nodebug
[no]d2	schaltet noch mehr Debug-Ausgaben an oder aus. **Standardwert:** nod2
domain=*Domainname*	entspricht der domain-Anweisung in /etc/resolv.conf (siehe Abschnitt 3.1.1). **Standardwert:** Der Wert der Umgebungsvariablen $LOCALDOMAIN, der domain-Wert aus /etc/resolv.conf oder die Domain des Kernel-Hostnamens. Wenn Sie die Domain *mehrfach* setzen (z.B. als Umgebungsvariable *und* im Kernel-Hostnamen), verwendet nslookup den nach dieser Aufzählung zuerst gefundenen Wert als Standardwert.
srchl[ist]=*Name1* [/*Name2...*]	entspricht der search-Anweisung in /etc/resolv.conf (siehe Abschnitt 3.1.1). **Standardwert:** Der Wert der Umgebungsvariablen $LOCALDOMAIN, der search-Anweisung in /etc/resolv.conf oder die Domain des Kernel-Hostnamens. Wenn Sie die Suchliste *mehrfach* setzen (z.B. als Umgebungsvariable *und* in /etc/resolv.conf), verwendet nslookup den nach dieser Aufzählung zuerst gefundenen Wert als Standardwert.
[no]defname	aktiviert oder deaktiviert das Anhängen des domain-Wertes an alle Namen, die keinen Punkt enthalten. **Standardwert:** defname
[no]sea[rch]	aktiviert oder deaktiviert das Anhängen der srchlist an alle Namen, die mindestens einen Punkt enthalten. **Achtung:** nslookup ignoriert den Wert der in Abschnitt 3.1.1 beschriebenen ndots-Option in /etc/resolv.conf. **Standardwert:** search
po[rt]=*Portnummer*	Der TCP/UDP-Port, auf dem der server erreichbar ist. **Standardwert:** 53

Tabelle 4.1: nslookup-Optionen (Fortsetzung)

Option	Beschreibung
(querytype|q[uery]|ty[pe])= (ANY|A|CNAME|MX|NS|PTR|SOA|...)	bestimmt den gesuchten Record-Typ. **Standardwert:** A oder (bei IP-Adressen): PTR
[no]rec[urse]	bestimmt, ob nslookup rekursive oder iterative Anfragen an den Nameserver sendet. Bei norecurse sendet der Nameserver nur Einträge aus seinem Cache (z.B. die Root-Nameserver) oder den eigenen Zonefiles. **Standardwert:** recurse
ret[ry]=Anzahl	entspricht der attempts-Option in /etc/resolv.conf (siehe Abschnitt 3.1.1). **Standardwert:** 4
ro[ot]= *Nameserver*	stellt ein, welchen Nameserver das root-Kommando auswählt. **Standardwert:** ns.internic.net Diese Option ist bei einigen Distributionen deaktiviert.
ti[meout]= *Sekunden*	entspricht der timeout-Option in /etc/resolv.conf (siehe Abschnitt 3.1.1). **Standardwert:** 5
[no]vc	verwendet TCP (vc) oder UDP (novc) zur Kommunikation mit dem Nameserver. **Standardwert:** TCP bei Zonetransfers, sonst UDP.
[no]ig[noretc]	legt fest, ob nslookup ungültige Antworten verwerfen soll. **Standardwert:** noignoretc
[no]sil[ent]	bestimmt, ob nslookup beim Programmstart den Hinweis auf die Programme host und dig ausgibt, die man an Stelle von nslookup verwenden soll. **Standardwert:** nosilent (BIND 9.0–9.2), silent (BIND 9.3)

Tabelle 4.1: nslookup-Optionen (Fortsetzung)

4.1.3 Interaktiver Modus

Kommandos

Am `nslookup`-Prompt können Sie folgende Kommandos eingeben:

Kommando	Auswirkung
(*Hostname*|*IP*) [*Nameserver*]	fragt den angegebenen Nameserver nach dem angegebenen Hostnamen oder der IP-Adresse. Wenn Sie keinen Nameserver angeben, spricht `nslookup` den Nameserver an, den Sie mit den Kommandos `server` oder `root` (s.u.) oder in der Datei `/etc/resolv.conf` eingestellt hatten.
(server|lserver) *Nameserver*	stellt den Nameserver für alle folgenden Abfragen ein. Im Gegensatz zu `lserver` fragt `server` die IP-Adresse des neuen Nameservers bei dem Nameserver ab, den Sie mit `server` eingestellt haben.
`root`	Stellt den Nameserver für alle folgenden Abfragen auf einen der Root-Nameserver (`ns.internic.net`) ein. Dieses Kommando ist bei einigen Distributionen deaktiviert.
`finger` [*Username*] [[>]> *Dateiname*]	zeigt Details zu den Anwendern an, die auf dem zuletzt abgefragten Host angemeldet sind. Das funktioniert natürlich nur, wenn der `fingerd` auf diesem Host läuft, was heute nur noch selten der Fall ist. Sie können die Ausgabe in eine lokale Datei umleiten, z.B. um sie später mit dem `view`-Kommando zu betrachten. Beispiel: `finger >out.txt` Wenn die Datei bereits existiert, können Sie durch Angabe von *zwei* Größerzeichen verhindern, dass `nslookup` die Datei überschreibt, und erreichen, dass es die Ausgabe an die bestehende Datei anhängt. Dieses Kommando ist bei einigen Distributionen deaktiviert.

Tabelle 4.2: `nslookup`-Kommandos

Kommando	Auswirkung
`ls [-(t *Typ* |d|a|h|s|)] *Domain* [[>]>*Dateiname*]`	zeigt alle Einträge der angegebenen Domain an. Das funktioniert natürlich nur, wenn der angesprochene Nameserver Ihrem Rechner Zonetransfers gestattet. Sie können angeben, welche Record-Typen Sie sehen möchten: `-t` *Typ* : nur Records dieses Typs (z.B. `NS`) `-d`: alle Records, entspricht *`-t ANY`* `-a`: entspricht *`-t CNAME`* `-h`: entspricht *`-t HINFO`* `-s`: entspricht *`-t WKS`* Wenn Sie nichts angeben, gilt `-t A` (nur A-Records). Sie können die Ausgabe in eine lokale Datei umleiten, z.B. um sie später mit dem `view`-Kommando zu betrachten. Beispiel: `ls -d uni-frankfurt.de. >out.txt` Wenn die Datei bereits existiert, können Sie durch Angabe von *zwei* Größerzeichen verhindern, dass `nslookup` die Datei überschreibt, und erreichen, dass es die Ausgabe an die bestehende Datei anhängt. Dieses Kommando ist bei einigen Distributionen deaktiviert.
`view` *Dateiname*	Sortiert den Inhalt der angegebenen Datei und zeigt ihn mit `more` oder `pg` an.
`(help|?)`	gibt eine kurze Übersicht aller `nslookup`-Kommandos aus.
`(exit|`STRG`+`D`)`	beendet die `nslookup`-Sitzung.
	Achtung: Versuchen Sie nicht, `nslookup` mit `quit` zu beenden. Dies würde den Hostnamen `quit` abfragen.
	(Die Hostmaster der Domain `ugcs.caltech.edu` haben inzwischen für den sehr häufig abgefragten Hostnamen `quit` einen CNAME eingerichtet, der auf `use.exit.to.leave.nslookup` verweist.)

Tabelle 4.2: `nslookup`-Kommandos (Fortsetzung)

Kommando	Auswirkung
`set all`	zeigt die Werte aller Optionen an. Die Beschreibung aller Optionen finden Sie in Abschnitt 4.1.2.
`set` *`Option`* `[=`*`Wert`*`]`	setzt den Wert einer Option. Die Beschreibung aller Optionen finden Sie in Abschnitt 4.1.2.

Tabelle 4.2: `nslookup`-Kommandos (Fortsetzung)

Beispiele

Im folgenden Beispiel ermittelt `nslookup` den Hostnamen und den Delegationspfad der IP-Adresse `141.2.15.24`.

Hierbei kommt die `norecursion`-Option zum Einsatz. Sie bewirkt, dass `nslookup` *iterative* Anfragen an die Nameserver sendet, die daraufhin nur mit Einträgen aus ihren Caches oder den eigenen Zonefiles antworten.

```
$ nslookup
> set norecursion
> 141.2.15.24
Server:         127.0.0.1
Address:        127.0.0.1#53

Non-authoritative answer:
*** Can't find 24.15.2.141.in-addr.arpa.: No answer

Authoritative answers can be found from:
.       nameserver = A.ROOT-SERVERS.NET.
.       nameserver = B.ROOT-SERVERS.NET.
.       nameserver = C.ROOT-SERVERS.NET.
[...]
>
```

Listing 4.2: Der eigene Nameserver verweist auf die Root-Nameserver.

```
(Fortsetzung:)
> server A.ROOT-SERVERS.NET.
Default server: A.ROOT-SERVERS.NET.
Address: 198.41.0.4#53
> 141.2.15.24
Server:         A.ROOT-SERVERS.NET.
Address:        198.41.0.4#53

Non-authoritative answer:
*** Can't find 24.15.2.141.in-addr.arpa.: No answer

Authoritative answers can be found from:
141.in-addr.arpa        nameserver = NS-PRI.RIPE.NET.
141.in-addr.arpa        nameserver = NS.LACNIC.NET.
141.in-addr.arpa        nameserver = SEC1.APNIC.NET.
141.in-addr.arpa        nameserver = TINNIE.ARIN.NET.
[...]
>
```

Listing 4.3: Der Root-Nameserver verweist auf die Regional-Internet-Registry-(RIR-)Nameserver.

```
(Fortsetzung:)
> server NS-PRI.RIPE.NET.
Default server: NS-PRI.RIPE.NET.
Address: 193.0.0.195#53
> 141.2.15.24
Server:         NS-PRI.RIPE.NET.
Address:        193.0.0.195#53

Non-authoritative answer:
*** Can't find 24.15.2.141.in-addr.arpa.: No answer

Authoritative answers can be found from:
2.141.in-addr.arpa
     nameserver = HERA.RBI.INFORMATIK.UNI-FRANKFURT.de.
2.141.in-addr.arpa
```

```
     nameserver = CHAPLIN.RZ.UNI-FRANKFURT.de.
[...]
>
```

Listing 4.4: Der RIR-Nameserver verweist auf die Local-Internet-Registry-(LIR-) Nameserver.

```
(Fortsetzung:)
> server HERA.RBI.INFORMATIK.UNI-FRANKFURT.de.
Default server: HERA.RBI.INFORMATIK.UNI-FRANKFURT.de.
Address: 141.2.1.1#53
> 141.2.15.24
Server:         HERA.RBI.INFORMATIK.UNI-FRANKFURT.de.
Address:        141.2.1.1#53

24.15.2.141.in-addr.arpa
     name = zeus.rbi.informatik.uni-frankfurt.de.
> exit
```

Listing 4.5: Der zuständige Nameserver beantwortet die Frage.

4.1.4 Befehlszeilen-Modus

Beispiele

Das folgende Beispiel zeigt die Auflösung eines Hostnamens:

```
$ nslookup www.informatik.uni-frankfurt.de
Server:         127.0.0.1
Address:        127.0.0.1#53

Non-authoritative answer:
www.informatik.uni-frankfurt.de
   canonical name = sphinx.rbi.informatik.uni-frankfurt.de.
Name:   sphinx.rbi.informatik.uni-frankfurt.de
Address: 141.2.15.17
$
```

Das Ergebnis lautet also 141.2.15.17.

Die Meldung Non-authoritative answer weist darauf hin, dass diese Antwort von einem Nameserver stammt, der nicht selbst für die Zone zuständig und daher vielleicht falsch ist. Sie können die zuständigen Nameserver der Zone selbst ermitteln:

```
$ nslookup -type=NS informatik.uni-frankfurt.de
Server:         127.0.0.1
Address:        127.0.0.1#53

Non-authoritative answer:
informatik.uni-frankfurt.de
     nameserver = hera.rbi.informatik.uni-frankfurt.de.
informatik.uni-frankfurt.de
     nameserver = hermes.rbi.informatik.uni-frankfurt.de.
[...]
```

Listing 4.6: Die zuständigen Nameserver ermitteln

... und direkt befragen (1. Option: Frage, 2. Option: Nameserver):

```
$ nslookup www.informatik.uni-frankfurt.de \
  hermes.rbi.informatik.uni-frankfurt.de.
Server:         hermes.rbi.informatik.uni-frankfurt.de.
Address:        141.2.1.3#53

www.informatik.uni-frankfurt.de
  canonical name = sphinx.rbi.informatik.uni-frankfurt.de.
Name:   sphinx.rbi.informatik.uni-frankfurt.de
Address: 141.2.15.17
$
```

Listing 4.7: Einen zuständigen Nameserver befragen

4.2 dig

Der »Domain Internet Groper« (`dig`) ist *das* Tool zur Abfrage von DNS-Servern an der Kommandozeile. Da es die Ergebnisse im Zonefileformat ausgibt, eignet es sich sehr gut zur Fehleranalyse.

`dig` wird vom ISC gepflegt und liegt der BIND-Distribution bei (z.B. im Debian-Package: `dnsutils`).

4.2.1 Aufruf

- Die vollständige Syntax zur Abfrage eines Namens lautet:

```
dig
  [@Nameserver]
  [-Systemoptionen]
  [Name|-x IP] [Typ] [Klasse]
  [+Abfrageoptionen]
```

Listing 4.8: `dig`-Aufruf (Übersicht)

- Die übliche Syntax beschränkt sich auf:
  `dig Name Typ`
  z.B. `dig www.uni-frankfurt.de A`
- Die Syntax zur Abfrage *mehrerer* Namen lautet:
  `dig [globale +Abfrageoptionen] Abfrage weitereAbfrage ...`
- `dig` (ohne Optionen) gibt eine Liste der Root-Nameserver aus.
- `dig -h` gibt eine kurze Liste möglicher Optionen aus.

`dig` 9.2.4 kennt die folgenden Optionen:

Option	Beschreibung
@Nameserver	gibt den Namen oder die IP-Adresse des Nameservers an, den `dig` fragen soll. Voreingestellt sind die Nameserver aus `/etc/resolv.conf`.

Tabelle 4.3: `dig`-Aufruf (Erklärung)

Option	Beschreibung
Name|`-x` *IP*	gibt den Hostnamen, Domainnamen oder die IP-Adresse an, die Sie abfragen möchten. Weitere Infos zur `-x`-Option finden Sie bei den Systemoptionen. **Standardwert:** . (Root)
Typ	gibt den Typ des Records an, den Sie abfragen möchten. Zusätzlich zu allen Record-Typen können Sie mit `ANY` den gesamten RRset, mit `AXFR` die gesamte Zone und mit `IXFR=`*Versionsnummer* die Änderungen der Zone seit der angegebenen Versionsnummer abfragen. **Standardwert:** `A` (falls Name gesetzt ist), `PTR` (falls `-x` gesetzt ist), sonst `NS`
Klasse	die Klasse, in der der Record gesucht wird (z.B. `CH`, `HS`, `IN`) **Standardwert:** `IN`
-Systemoptionen	siehe nächste Tabelle.
+Abfrageoptionen	siehe übernächste Tabelle.

Tabelle 4.3: `dig`-Aufruf (Erklärung) (Fortsetzung)

4.2.2 Systemoptionen

Option	Beschreibung
`-b` *SourceIP*	bestimmt die Source-IP-Adresse aller Netzwerk-Pakete, die `dig` erstellt. *SourceIP* sollte die Adresse einer lokalen Netzwerkschnittstelle sein. **Standardwert:** wird aus der Routingtabelle ermittelt.
`-c` *Klasse*	bestimmt, in welcher Klasse `dig` sucht (z.B. `CH`, `HS`, `IN`). **Standardwert:** `IN`
`-f` *Dateiname*	liest die Abfragen, die mit Zeilenumbrüchen getrennt sind, aus der angegebenen Datei.
`-i`	IPv6-Reverse-Adressen gemäß RFC2874 (siehe `-x`).

Tabelle 4.4: `dig`-Systemoptionen

Option	Beschreibung
`-k` *`Dateiname`*	bewirkt, dass `dig` die Abfragen mit dem TSIG-Schlüssel aus der angegebenen Datei signiert. Die Herstellung von TSIG-Schlüsseln mit `dnssec-keygen` wird in Abschnitt 12.4.1 beschrieben.
`-p` *`Portnummer`*	bestimmt den tcp/udp-Port, den `dig` auf dem Nameserver ansprechen soll. **Standardwert:** `53`
`-t` *`Typ`*	entspricht der Angabe des *`Typ`* hinter *`Name`* (siehe Tabelle 4.3).
`-x` *`IP-Adresse`* `[-n]`	sucht den Reverse-Eintrag (Hostnamen) zu der angegebenen IP-Adresse. `dig` dreht IPv4-Adressen automatisch um und hängt `in-addr.arpa.` an. Bei IPv6-Adressen verwendet `dig` (bis zur Version 9.2.3) entweder die Bitstring-Methode in `ip6.arpa` (RFC2874) oder (wenn Sie `-i` angeben) die Nibble-Methode in `ip6.int` (RFC1886). Da sich inzwischen jedoch die Nibble-Methode in `ip6.arpa` (RFC3596) durchgesetzt hat, ist die `-x`-Option bei IPv6-Adressen für Sie wahrscheinlich nutzlos. Ab Version 9.3 verwendet `dig` die korrekte Adressierung (Nibble in `ip6.arpa`, RFC3596).
`-y` *`Name:Schlüssel`*	bewirkt, dass `dig` die Abfragen mit dem als Argument angegebenen TSIG-Schlüssel signiert. Wenn Sie Bedenken haben, dass andere Anwender diesen Schlüssel in der Ausgabe von `ps` finden, sollten Sie den Schlüssel besser in einer Datei oder einer Named Pipe speichern und stattdessen `-k` verwenden. Die Herstellung von TSIG-Schlüsseln mit `dnssec-keygen` wird in Abschnitt 12.4.1 beschrieben.

Tabelle 4.4: `dig`-Systemoptionen (Fortsetzung)

4.2.3 Abfrageoptionen

Alle Abfrageoptionen sind mit einem Standardwert vorbelegt. Mit den folgenden Parametern können Sie die voreingestellten Werte ändern.

Soweit nicht anders angegeben, sind alle Optionen standardmäßig deaktiviert.

Abfrageoptionen zum Aufbau der Abfrage:

Option	Beschreibung
`+(search|defname)`	aktiviert das Anhängen der Suchliste (aus der `+domain`-Option oder `/etc/resolv.conf`). Dies ist standardmäßig deaktiviert.
`+adflag`	setzt das `Authentic-Data`- (AD-)Flag in der Abfrage, was jedoch keine Auswirkung hat.
`+cdflag`	setzt das `Checking-Disabled`- (CD-) Flag, was den Nameserver daran hindert, seine Antwort über DNSSEC zu signieren.
`+dnssec`	bewirkt, dass `dig` das `Dnssec-OK`- (DO-) Flag setzt, was den Nameserver vielleicht dazu bringt, seine Antwort zu signieren.
`+domain=`*`Domainname`*	legt die Domain fest, die `dig` automatisch an den gesuchten Namen anhängt. Aktiviert automatisch `+search`. **Standardwert:** Die Suchliste aus der `search`- oder `domain`-Anweisung in `/etc/resolv.conf`.
`+ndots=Anzahl`	gibt an, wie viele Punkte ein Name enthalten muss, damit `dig` ihn zuerst ohne angehängte Suchliste abfragt. **Standardwert:** der Wert der `ndots`-Option in `/etc/resolv.conf` oder `1`.
`+norecurse`	bewirkt, dass `dig` das `Recursion-Desired`- (RD-) Flag *nicht* setzt und damit eine iterative Anfrage sendet.

Tabelle 4.5: `dig`-Abfrageoptionen zum Aufbau der Abfrage

Option	Beschreibung
`+nssearch`	fragt die SOA-Records aller Nameserver ab, die für diese Zone autoritativ sind.
`+trace`	bewirkt, dass `dig` iterative Anfragen sendet (beginnend bei den Root-Nameservern) und den Antworten bis zum Ziel folgt. Dabei gibt es sämtliche Zwischenschritte aus.

Tabelle 4.5: `dig`-Abfrageoptionen zum Aufbau der Abfrage (Fortsetzung)

Abfrageoptionen zum Ausgabeformat:

Option	Beschreibung
`+[no]all`	schaltet *alle* Ausgabe-Optionen an oder aus.
`+multiline`	bewirkt, dass `dig` SOA-Records in »menschenlesbarem« Format (mit Zeilenumbrüchen und Kommentaren) ausgibt.
`+noadditional`	unterlässt die Ausgabe der »Additional-Section«.
`+noanswer`	unterlässt die Ausgabe der »Answer-Section«.
`+noauthority`	unterlässt die Ausgabe der »Authority-Section«.
`+nocmd`	unterlässt die Ausgabe der `dig`-Versionsnummer und der Abfrageoptionen beim Programmstart.
`+nocomments`	unterlässt die Ausgabe der Kommentarzeilen im Ergebnis.
`+noquestion`	unterlässt die Ausgabe der Abfrage, die mit dem Ergebnis zurückgesendet wird.
`+nostats`	unterlässt die Ausgabe der Performance-Werte.
`+qr`	zeigt die gesendete Abfrage an.
`+short [+identify]`	bewirkt, dass `dig` das Ergebnis in einem kürzeren Format ausgibt. Wenn `+identify` gesetzt ist, zeigt `dig` dabei trotzdem die IP-Adresse und den Port des zuständigen Nameservers an.

Tabelle 4.6: `dig`-Abfrageoptionen zum Ausgabeformat

Abfrageoptionen zu Netzwerk und Fehlerbehandlung:

Option	Beschreibung
+[no]tcp	bestimmt, ob dig den Nameserver per TCP oder UDP anspricht. **Standardwert:** TCP für Zonetransfers (Typen AXFR und IXFR), UDP für alle übrigen Abfragen.
+[no]vc	entspricht +[no]tcp. vc steht für »virtual circuit«, den Begriff für TCP aus den 1970ern.
+besteffort	bewirkt, dass dig versucht, defekte Antworten zu interpretieren.
+bufsize=Bytes	setzt die per EDNS0 mitgesendete UDP message buffer size auf einen Wert zwischen 0 und 65535 Bytes.
+ignore	bewirkt, dass dig bei unvollständigen Antworten keinen zweiten Versuch über TCP unternimmt.
+nofail	bewirkt, dass dig nach Ablauf des Timeouts zum nächsten Nameserver aus /etc/resolv.conf wechselt.
+time=Sekunden	gibt an, wie lange dig maximal auf die Antwort eines Nameservers wartet. **Standardwert:** 5
+tries=*Anzahl*	gibt an, wie viele Versuche dig maximal unternimmt, wenn es innerhalb des +timeouts keine Antwort auf eine UDP-Anfrage erhält. **Standardwert:** 3

Tabelle 4.7: dig-Abfrageoptionen zu Netzwerk und Fehlerbehandlung

4.2.4 Beispiele

Sie können mit dig z.B. die Mailserver einer Domain ermitteln:

```
$ dig MX vsb.informatik.uni-frankfurt.de
; <<>> DiG 9.2.1 <<>> MX vsb.informatik.uni-frankfurt.de
;; global options:  printcmd
;; Got answer:
```

```
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 31501
;; flags: qr rd ra;
;; QUERY: 1, ANSWER: 1, AUTHORITY: 0, ADDITIONAL: 0

;; QUESTION SECTION:
;vsb.informatik.uni-frankfurt.de. IN    MX

;; ANSWER SECTION:
vsb.informatik.uni-frankfurt.de.
       1659 IN MX 10 diamant-atm.vsb.cs.uni-frankfurt.de.

;; Query time: 656 msec
;; SERVER: 127.0.0.1#53(127.0.0.1)
;; WHEN: Mon Jul 19 13:58:06 2004
;; MSG SIZE  rcvd: 84 Listing
```

Listing 4.9: Die Mailserver einer Domain ermitteln

Mehr zum Thema: `MX` steht für »Mail Exchange«. MX-Records werden in Abschnitt 7.5.25 beschrieben.

Sie können die Ausgabe auf das Ergebnis beschränken:

```
$ dig +short MX vsb.informatik.uni-frankfurt.de
10 diamant-atm.vsb.cs.uni-frankfurt.de.
```

Listing 4.10: Die Mailserver einer Domain ermitteln (kürzere Ausgabe)

oder die Delegation einer IP-Adresse verfolgen:

```
$ dig -x 141.2.1.1 +trace +additional
; <<>> DiG 9.2.1 <<>> -x 141.2.1.1 +trace +tcp
;; global options:  printcmd
.   431049  IN      NS      A.ROOT-SERVERS.NET.
.   431049  IN      NS      B.ROOT-SERVERS.NET.
.   431049  IN      NS      C.ROOT-SERVERS.NET.
[...]
```

```
;; Received 436 bytes from 127.0.0.1#53(127.0.0.1) in 3 ms

141.in-addr.arpa.    86400   IN      NS      NS-PRI.RIPE.NET.
141.in-addr.arpa.    86400   IN      NS      NS.LACNIC.NET.
141.in-addr.arpa.    86400   IN      NS      SEC1.APNIC.NET.
141.in-addr.arpa.    86400   IN      NS      TINNIE.ARIN.NET.
[...]
;; Received 215 bytes from 192.5.5.241#53
;; (F.ROOT-SERVERS.NET) in 165 ms

2.141.in-addr.arpa. 172800  IN  NS
      HERA.RBI.INFORMATIK.UNI-FRANKFURT.de.
2.141.in-addr.arpa. 172800  IN  NS
      CHAPLIN.RZ.UNI-FRANKFURT.de.
[...]
;; Received 174 bytes from 192.134.0.49#53
;; (NS3.NIC.FR) in 18 ms

1.2.141.in-addr.arpa.   3600  IN  NS
      hermes.informatik.uni-frankfurt.de.
;; Received 88 bytes from 141.2.22.74#53
;; (CHAPLIN.RZ.UNI-FRANKFURT.de) in 6 ms

1.1.2.141.in-addr.arpa. 37722   IN  PTR
      hera.rbi.informatik.uni-frankfurt.de.
1.2.141.in-addr.arpa.   37722   IN  NS
     hera.rbi.informatik.uni-frankfurt.de.
1.2.141.in-addr.arpa.   37722   IN  NS
      hermes.rbi.informatik.uni-frankfurt.de.
[...]
;; Received 399 bytes from 217.160.131.135#53
;; (hermes.informatik.uni-frankfurt.de) in 12 ms
```

Listing 4.11: Hostnamen und Delegation einer IP-Adresse ermitteln

4.3 host

host ist ein DNS-Abfragetool, das zu Zeiten von BIND4 auf ftp.rutgers.edu erschien. Später teilte sich die Entwicklung, so dass heute drei verschiedene Implementierungen in Umlauf sind:

- Version 9.3.0 vom ISC, die der BIND9-Distribution beiliegt (Debian-Package: bind9-host).
- Version 991529, die Eric Wassenaar so erweitert hat, dass es fehlerhafte Einträge in Zonen sucht (Debian-Package: host).
- Version 20031203, mit der Greg A. Woods die Entwicklung seit dem Tod von Eric Wassenaar weiterführt (Download: ftp://ftp.weird.com/pub/local/host-*.tar.gz).

4.3.1 Aufruf

Aufruf: host [*Optionen*] *Name* [*Nameserver*], wobei

- *Name* für einen Hostnamen, einen Domainnamen oder eine IP-Adresse und
- *Nameserver* für den anzusprechenden Nameserver

steht. Wenn Sie keinen Nameserver angeben, verwendet host die Nameserver aus /etc/resolv.conf.

Wenn Sie host *ohne* Optionen aufrufen, gibt es eine kurze Liste einiger Optionen aus.

4.3.2 Optionen (ISC-Version)

Option	Beschreibung
-(v|d)	bewirkt, dass host zusätzliche Meldungen ausgibt.
-a	entspricht -t ANY und zeigt *alle* Einträge des angegebenen Namens an.
-C	gibt die SOA-Records aller Nameserver aus, die (laut der NS-Records) autoritativ für diese Zone sind. So können Sie testen, ob alle sekundären Nameserver auf dem aktuellen Stand sind.

Tabelle 4.8: host-Optionen (ISC-Version)

Option	Beschreibung
`-c (HS|CH)`	setzt die Klasse auf Hesiod oder Chaosnet.
`-l`	bewirkt, dass `host` einen Zonetransfer durchführt und den Inhalt der Zone ausgibt.
`-n`	bewirkt, dass `host` (bis zur Version 9.2.3) bei IPv6-Adressen die Nibble-Methode in `ip6.int` (RFC1886) anstelle der Bitstring-Methode in `ip6.arpa` (RFC2874) anwendet. Da sich inzwischen jedoch die Nibble-Methode in `ip6.arpa` (RFC3596) durchgesetzt hat, ist die `-n`-Option bei IPv6-Adressen für Sie wahrscheinlich nutzlos. Ab Version 9.3 verwendet `host` die korrekte Adressierung (Nibble in `ip6.arpa`, RFC3596), weshalb diese Option dort nicht mehr existiert.
`-N` *Anzahl*	gibt an, ab wie vielen Punkten im Hostnamen `host` *nicht* die Suchliste (aus `domain`- oder `search`-Anweisung in `/etc/resolv.conf`) anhängt. **Standardwert:** der Wert der `ndots`-Option in `/etc/resolv.conf` oder `1`.
`-r`	bewirkt, dass `host` das `Recursion-Desired`-(RD-)Flag *nicht* setzt und damit eine iterative Anfrage sendet.
`-R` *Anzahl*	gibt an, wie oft `host` die Anfrage wiederholen soll, wenn innerhalb des Timeouts keine Antwort eintrifft. **Standardwert:** `1`
`-T`	bewirkt, dass `host` den Nameserver über TCP anspricht. Ohne diese Option verwendet `host` TCP nur bei Zonetransfers und UDP bei allen anderen Record-Typen.
`-t` *Typ*	gibt den Record-Typ an, nach dem `host` fragen soll. **Standardwert:** `A`, `SOA` (falls `-C` angegeben) oder `PTR` (falls *Name* eine IP-Adresse ist).
`-w`	setzt das Timeout (`-W`) auf unendlich.
`-W` *Sekunden*	legt fest, wie lange `host` maximal auf Antwort des Nameservers wartet.

Tabelle 4.8: `host`-Optionen (ISC-Version) (Fortsetzung)

4.3.3 Optionen (Versionen 991529 und 20031203)

Option	Beschreibung
`-a` `--anything`	entspricht `-t ANY` und zeigt *alle* Einträge an, die der Nameserver zu dem angegebenen Namen kennt. **Achtung:** Wenn Sie einen Nameserver nach Einträgen einer Zone fragen, für die er *nicht* autoritativ ist, sehen Sie möglicherweise nicht alle Einträge, sondern nur die, die dieser zu diesem Zeitpunkt gecachet hat.
`-A` `--addrcheck`	Wenn Sie einen Hostnamen übergeben, prüft `host`, ob alle IP-Adressen dieses Hostnamens auf ihn zurückverweisen. Wenn Sie eine IP-Adresse übergeben, prüft `host`, ob sein Name auf diese IP-Adresse zurückverweist. Wenn Sie den Namen einer Zone und zusätzlich die Option `-l` angeben, prüft `host` für *alle* A- und PTR-Records dieser Zone, ob die zugehörigen PTR- und A-Records existieren.
`-B`	emuliert das Verhalten von BIND beim Anhängen der Suchliste. Im Gegensatz zur `-R`-Option stoppt `host` *nicht* bei RRsets, die nur andere Record-Typen enthalten.
`-C`	gibt die SOA-Records aller Nameserver aus, die (laut der NS-Records) autoritativ für diese Zone sind. So können Sie testen, ob alle sekundären Nameserver auf dem aktuellen Stand sind. `host` warnt Sie, wenn die Antworten voneinander abweichen oder nicht autoritativ sind.
`-c (HS|CH|CS)` `--class=...`	setzt die Klasse auf Hesiod, Chaosnet oder CSNET.
`--canoncheck`	aktiviert die Prüfung der CNAME-Verweisziele bei der Untersuchung ganzer Zonen, die Sie z.B. mit `-l` aufrufen können.

Tabelle 4.9: `host`-Optionen (Versionen 991529 und 20031203)

Option	Beschreibung
`--canonskip`	unterlässt die Prüfung der CNAME-Verweisziele bei der Untersuchung ganzer Zonen.
`--checkzone`	entspricht `-CAl` und bewirkt eine gründliche Prüfung aller Einträge in der angegebenen Zone.
`--cnamecheck`	prüft, ob das Verweisziel des angegebenen CNAME-Records existiert. Dabei folgt `host` bis zu zehn verketteten CNAMEs.
`--compare`	beantwortet die Abfrage aus dem Cache-Verzeichnis, wenn die Option `-dump` gesetzt ist und die SOA-Versionsnummer der Cachedatei mit der auf dem Nameserver übereinstimmt.
`-d` `--debug`	bewirkt, dass `host` jeden Nameserver-Zugriff ankündigt.
`-D`	entspricht `-H` und gibt zusätzlich die Namen aller »Doppel«-Hosts (Hosts, deren IP-Adresse mehr als einem Hostnamen zugeordnet sind) aus.
`-dd`	bewirkt, dass `host` noch mehr Meldungen als bei `-d` ausgibt.
`--dump=`*`Verzeichnis`*	schreibt alle übertragenen Zonen in einem Binärformat in das angegebene Cache-*`Verzeichnis`*. Auf die Inhalte dieses Cache können Sie mit der `--compare`-Option zugreifen.
`-E`	entspricht `-H` und gibt zusätzlich die Namen aller »Extrazone«-Hosts (Hosts in nichtdelegierten Subdomains) der Zone aus.
`-e` `--exclusive`	unterdrückt die Ausgabe aller Records, die nicht in der angegebenen Zone liegen (z.B. Glue-Records), sowie die Ausgabe der Additional- und Authority-Section.
`-f` *`Dateiname`* `--file=`*`Dateiname`*	schreibt zusätzlich zu den Ausgaben auf `STDOUT` alle ermittelten Resource-Records in eine Datei.

Tabelle 4.9: `host`-Optionen (Versionen 991529 und 20031203) (Fortsetzung)

Option	Beschreibung
`-F` *`Dateiname`*	schreibt alle Ausgaben, die sonst auf `STDOUT` erscheinen, in die Datei und gibt auf `STDOUT` nur Resource-Records aus.
`-G`	entspricht `-H` und gibt zusätzlich die Namen aller »Gateway«-Hosts (Hosts mit mehreren IP-Adressen) aus.
`-g` *`Länge`*	gibt nur Resource-Records aus, deren FQDN mindestens *`Länge`* Zeichen enthält.
`-H` `--hostcount`	kann an Stelle der `-l`-Option eingesetzt werden und gibt die Zahl der eindeutigen Hostnamen (A-Records) aus, die eine Zone enthält. »Technische« Einträge (`localhost`, der Zonenname und Glue-Records) werden nicht mitgezählt. Ohne weitere Optionen gibt `host` nur die Zahl, aber keinen Record selbst aus.
`--help`	gibt eine kurze Hilfe zu einigen Optionen aus.
`-i` `--inaddr`	bewirkt, dass `host` die angegebene IP-Adresse auch bei Zonelistings in einen Eintrag der Domain `in-addr.arpa` umwandelt.
`-I` *`Zeichenmenge`*	erklärt die angegebenen Zeichen zu gültigen Zeichen. Zum Beispiel verhindert `-I _`, dass `host` Warnungen bei Hostnamen ausgibt, die Unterstriche enthalten.
`-j` *`Minport`* `-J` *`Maxport`*	bestimmt den Bereich der Source-Portnummern aller Netzwerk-Pakete, die `host` erstellt. Wenn Sie nur eine der beiden Portnummern angeben, verwendet `host` diese ausschließlich. **Achtung:** Das Eingrenzen der Source-Portnummern kann bei TCP-Paketen (Zonetransfers und `-u`-Option) zu Problemen führen.
`-K`	gibt mehrere Timing-Informationen aus.

Tabelle 4.9: `host`-Optionen (Versionen 991529 und 20031203) (Fortsetzung)

Option	Beschreibung
`-l` `--list`	bewirkt, dass `host` einen Zonetransfer durchführt und den Inhalt der Zone ausgibt.
	Achtung: `host` unterschlägt einige Record-Typen, wenn Sie nicht zusätzlich `-a` setzen.
	Geben Sie die Optionen `-l -a -Z` an, wenn Sie eine Zone im Original-Zonefile-Format anzeigen möchten.
`-L` *Tiefe* `--depth=`*Tiefe*	bewirkt. dass die Zonenlistings (`-l`, `-H`, `-G`, `-C`, `-E`, `-D`, `-A`) rekursiv bis zu *Tiefe* Delegationen folgen. `-l` entspricht `-L 1`.
`--load=`*Verzeichnis*	bewirkt, dass `host` alle Antworten aus dem *Verzeichnis* liest, das Sie vorher mit `-dump` gefüllt hatten.
`-m`	entspricht `-t MAILB`, was die Ausgabe auf Records vom Typ MB, MR, MG und MINFO eingrenzt. Zusätzlich löst `host` MR- und MG-Records bis zu den entsprechenden MB-Records auf.
`-M`	gibt die MX-Records aller delegierten Zonen der angegebenen Zone aus.
`-n`	bewirkt, dass `host` aus der angegebenen NSAP-Adresse einen Reverse-Eintrag unter `nsap.int` bildet. Informationen zu NSAP-Records finden Sie in RFC1637 und RFC1706.
`-N Zone,Zone,...`	gibt an, für welche Zonen `host` *keinen* Zonetransfer durchführen soll.
`-N` *Zonen*	bewirkt, dass `host` die angegebenen Zonen von der Rekursion (`-L`) ausschließt. Wenn Sie mehrere Zonen angeben möchten, trennen Sie diese bitte mit Kommas.
`--nothing`	unterlässt die Ausgabe aller Resource-Records auf `STDOUT` und in die Datei.
`-o` `--suppress`	unterlässt die Ausgabe aller Resource-Records auf `STDOUT`.

Tabelle 4.9: `host`-Optionen (Versionen 991529 und 20031203) (Fortsetzung)

Option	Beschreibung
`-O` *`SourceIP`*	bestimmt die Source-IP-Adresse aller Netzwerk-Pakete, die `host` erstellt. *`SourceIP`* muss die Adresse einer lokalen Netzwerkschnittstelle sein.
! `-p` `--primary`	bewirkt, dass `host` sich ausschließlich an den primären Nameserver (aus dem SOA-Record) einer Zone wendet.
`-P`	bewirkt, dass `host` sich direkt an einen der autoritativen Nameserver der Zone wendet. In Verbindung mit einer der Zonelisting-Optionen bewirkt diese Option, dass `host` die NS-Records der Zone durch die delegierenden NS-Records der übergeordneten Zone ersetzt.
`-q` `--quiet`	unterlässt die Ausgabe aller Warnungen. `host` kennzeichnet Warnungen sonst mit `!!!`.
`-Q` `--quick`	beschleunigt die Untersuchung von Zonen, indem zeitaufwendige Tests ausgelassen werden.
! `-r` `--norecurs`	bewirkt, dass `host` das `Recursion-Desired`-(RD-)Flag *nicht* setzt und damit eine iterative Anfrage sendet.
`-R`	Normalerweise hängt `host` die Domain des lokalen Rechners an, wenn der gesuchte Name keinen Punkt enthält. Diese Option bewirkt, dass `host` sie auch bei Namen anhängt, die Punkte enthalten. Wenn dies nicht zum Erfolg führt, hängt `host` alle Parent-Domains (außer der TLD) an, bis es einen Eintrag findet.
`--recursive`	entspricht `-L "unendlich"` und bewirkt, dass `host` *allen* Delegationen folgt.
`--retry=count`	gibt die Anzahl der Versuche an, die `host` zu jedem Nameserver unternehmen soll, wenn das Timeout überschritten wird.

Tabelle 4.9: `host`-Optionen (Versionen 991529 und 20031203) (Fortsetzung)

Option	Beschreibung
`-S` `--statistics`	bewirkt, dass `host` eine Statistik der angegebenen Zone oder (bei Rekursion:) Zonen ausgibt, die u.a. die Zahl der Records pro Typ, Host-Arten und Delegationen nennt. Verwenden Sie diese Option zusätzlich zu einer Zonelisting-Option (`-l`, `-H`, `-G`, `-C`, `-E`, `-D`, `-A`).
`-s` *Sekunden* `--timeout=`*Sekunden*	gibt an, wie lange `host` auf Antwort vom Nameserver warten soll. Dieser Wert überschreibt die `timeout`-Option in `/etc/resolv.conf`, die in Abschnitt 3.1.1 beschrieben wird.
`--server=host`	gibt den Nameserver an, den `host` fragen soll.
`-T`	bewirkt, dass `host` die TTL auch anzeigt, wenn `-v` nicht gesetzt ist.
`-t` *Typ* `--type=`*Typ*	gibt den Record-Typ an, nach dem `host` fragen soll. **Standardwert:** `A`, `SOA` (falls `-C` angegeben) oder `PTR` (falls *Name* eine IP-Adresse ist).
`-u` `--tcp`	bewirkt, dass `host` Nameserver grundsätzlich über TCP anspricht. Normalerweise spricht `host` die Nameserver nur bei Zonetransfers über TCP an und verwendet ansonsten das schnellere, aber unzuverlässigere UDP.
`-_` `--undercheck`	gibt eine Warnung für jeden Host aus, dessen Name einen Unterstrich enthält.
`--usage`	gibt eine sehr kurze Auflistung verschiedener Optionen aus und entspricht dem Aufruf ohne Optionen.
`-V` `--version`	gibt die Versionsnummer von `host` aus.
`-v` `--verbose`	bewirkt, dass `host` die TTL und Klasse jedes Records sowie die Authority- und Additional-Section und verschiedene Zwischenmeldungen ausgibt.

Tabelle 4.9: `host`-Optionen (Versionen 991529 und 20031203) (Fortsetzung)

Option	Beschreibung
`-vv`	bewirkt, dass `host` noch mehr Meldungen als bei `-v` ausgibt.
`-w`	Setzt das Timeout (`-W`) auf unendlich.
`-W`	gibt alle Wildcard-(`*`-)Records einer Zone aus. Wildcard-Records werden in Abschnitt 7.3.5 beschrieben.
`-x (Mehrere Abfragen) [-X Nameserver]`	erlaubt die Abfrage *mehrerer* Namen. Wenn Sie dabei einen Nameserver vorgeben möchten, müssen Sie diesen mit `-X` ankündigen, damit `host` die Angabe nicht als weitere Abfrage interpretiert.
`-Y`	gibt alle ermittelten Resource-Records zusätzlich hexadezimal aus.
`-Z --full`	gibt alle Ergebnisse im Zonefile-Format aus.
`-z`	gibt eine Liste der delegierten Zonen der angegebenen Zone aus.

Tabelle 4.9: `host`-Optionen (Versionen 991529 und 20031203) (Fortsetzung)

Leider sind viele dieser Optionen weder in der Manpage noch in der Onlinehilfe dokumentiert.

4.3.4 Beispiele

Die folgenden Beispiele wurden mit `host`-Version 991529 ausgeführt:

```
$ host hera.rbi.informatik.uni-frankfurt.de
hera.rbi.informatik.uni-frankfurt.de      A      141.2.1.1
```

Listing 4.12: Einen Hostnamen auflösen

```
$ host -CAL 1 rbi.informatik.uni-frankfurt.de. >/dev/null
rbi.informatik.uni-frankfurt.de
    SOA record not found at ws-mue1.win-ip.dfn.de,
    server failure
```

```
rbi.informatik.uni-frankfurt.de
    has lame delegation to ws-mue1.win-ip.dfn.de
!!! hera-b1.rbi.informatik.uni-frankfurt.de
    address 141.2.1.1
    maps to hera.rbi.informatik.uni-frankfurt.de
[...]
```

Listing 4.13: Zonen-Statistik generieren

```
$ host -qz informatik.uni-frankfurt.de
rbi.informatik.uni-frankfurt.de
thi.informatik.uni-frankfurt.de
vsb.informatik.uni-frankfurt.de
[...]
```

Listing 4.14: Delegationen ermitteln

```
$ host -qM informatik.uni-frankfurt.de
rbi.informatik.uni-frankfurt.de
    MX record currently not present
thi.informatik.uni-frankfurt.de
    MX 20 thi.informatik.uni-frankfurt.de
vsb.informatik.uni-frankfurt.de
    MX 10 diamant-atm.vsb.cs.uni-frankfurt.de
[...]
```

Listing 4.15: Mailserver einer Domain ermitteln

4.4 aaaa, mx, ns, soa, txt, zone etc.

Mit einem kleinen Trick können Sie bei `host` (Versionen 991529 und 20031203) die Angabe des Record-Typs sparen: Wenn Sie das `host`-Binary über den Namen eines Record-Typs aufrufen, indem Sie es z.B. umbenennen, kopieren oder linken, wird es nach Records von diesem Typ suchen.

```
$ find /usr/bin/ -lname host -ls
lrwxrwxrwx  1 root root   4 Jul  7 2002 /usr/bin/aaaa -> host
lrwxrwxrwx  1 root root   4 Jul  7 2002 /usr/bin/mx -> host
lrwxrwxrwx  1 root root   4 Jul  7 2002 /usr/bin/ns -> host
lrwxrwxrwx  1 root root   4 Jul  7 2002 /usr/bin/soa -> host
```

```
lrwxrwxrwx  1 root root   4 Jul  7 2002 /usr/bin/txt -> host
lrwxrwxrwx  1 root root   4 Jul  7 2002 /usr/bin/zone -> host
```

Listing 4.16: Symlinks auf /usr/bin/host

Wenn Sie `host` als `zone` aufrufen, aktiviert es automatisch `-l`:

```
$ zone -aZ thi.informatik.uni-frankfurt.de
$ORIGIN thi.informatik.uni-frankfurt.de
@ 38400   IN   SOA   (
    hermes.rbi.informatik.uni-frankfurt.de.
    dns.cs.uni-frankfurt.de.
    2004052117 ; serial (version)
    10800      ; refresh period (3 hours)
    3600       ; retry interval (1 hour)
    3000000    ;expire time (4 weeks, 6 days, 17 hours, 20 mins)
    86400      ;default ttl (1 day)
    )
          38400 IN NS    hera.rbi.informatik.uni-frankfurt.de.
          38400 IN MX    20 @
          38400 IN A     141.2.6.106
lovelace  38400 IN A     141.2.6.106
kassandra 38400 IN A     141.2.6.105
www       38400 IN CNAME lovelace
[...]
```

Listing 4.17: Das gesamte Zonefile ausgeben

5 DNS in eigenen Programmen verwenden

5.1 In der Programmiersprache C

Jede Standard-C-Bibliothek enthält die Funktionen `gethostbyaddr()` und `gethostbyname()`:

- `gethostbyaddr()` ermittelt die Hostnamen und weiteren IP-Adressen zu einer gegebenen IP-Adresse.
- `gethostbyname()` ermittelt die IP-Adressen und weiteren Hostnamen zu einem gegebenen Hostnamen.

Die GNU-C-Bibliothek (glibc) bietet zusätzlich einige Thread- oder IPv6-fähige Varianten dieser Funktionen:

- `gethostbyname_r()`, `gethostbyaddr_r()`, `gethostbyname2()`, `gethostbyname2_r()` sowie `getipnodebyname()`, `getipnodebyaddr()`

Da die gesuchten Informationen aus verschiedenen Quellen bezogen werden können (z.B. aus DNS, NIS oder `/etc/hosts`), greift die glibc bei diesen Funktionen auf die Name-Service-Switch-Module (NSS) zurück, die eine flexible Konfiguration ermöglichen.

Die Resolver-Bibliothek (`libresolv.so`, `resolv.h`) enthält weitere Funktionen, die direkt auf das DNS zugreifen:

- `res_search()`, `res_query()`, `res_querydomain()`, `res_mkquery()`, `res_send()`, `dn_comp()`, `dn_expand()`

Sie stammt ursprünglich aus der BIND-Distribution, wurde inzwischen jedoch ebenfalls in die glibc integriert: Zum Beispiel enthält die glibc 2.2.5 die Resolver-Bibliothek aus BIND 8.2.3.

Mehr zum Thema: Details zur Konfiguration des NSS finden Sie im nächsten Abschnitt.

Weitere Informationen zur Programmierung finden Sie in den Manpages der genannten Funktionen, im *GNU C Library Reference Manual*, das Sie mit `info libc` aufrufen.

5.2 In anderen Programmiersprachen

Viele Programmiersprachen enthalten die Funktionen `gethostbyname()` und `gethostbyaddr()`, die auf die gleichnamigen Betriebssystemfunktionen zurückgreifen.

Darüber hinaus gibt es für viele Programmiersprachen Bibliotheken, die das DNS selbst implementieren, z.B.

- für Perl: `Net::DNS`
- für Python: `DNS`
- für Java: `java.net.InetAdress.getByName()` und `java.net.InetAdress.getByAddress()`

5.2.1 Perl-Beispiele

Alle IP-Adressen eines Hosts ausgeben

```
#!/usr/bin/perl -w
use Net::DNS;
my ($res, $query, $rr);
$res = new Net::DNS::Resolver; # liest /etc/resolv.conf
$query = $res->search("www.foo.com");
if ($query) {
   foreach $rr ($query->answer) {
     next unless $rr->type eq "A";
     print $rr->address, "\n";
   }
} else {
   print "query failed: ", $res->errorstring, "\n";
}
```

Listing 5.1: DNS-Abfrage mit Perl (aus der Net::DNS-Manpage)

Einen Zonetransfer durchführen

```
#!/usr/bin/perl -w
use Net::DNS;
my ($res, @zone, $rr);

# Abfrage:
$res = new Net::DNS::Resolver;
$res->nameservers("some-nameserver.foo.com");
@zone = $res->axfr("foo.com.");

# Ausgabe:
foreach $rr (@zone) {
   $rr->print;
}
```

Listing 5.2: Zonetransfer mit Perl

Dynamic Updates auf eine Zone senden

```
#!/usr/bin/perl -w
use Net::DNS;
my ($res, $update, $ans);

# Initialisieren:
$res = new Net::DNS::Resolver;
$res->nameservers("some-nameserver.foo.com");

# Nachricht vorbereiten:
$update = new Net::DNS::Update("foo.com");
$update->push("update", rr_del("old.foo.com"));
$update->push("update", rr_add("new.foo.com A 10.1.2.3"));
# Falls der Nameserver nur signierte Updates zulässt:
$update->sign_tsig("rndc-key," La/E5CjG9O+os1jq0a2jdA==";);
```

```
# Nachricht an den Nameserver senden:
$ans = $res->send($update);
print $ans ? $ans->header->rcode : $res->errorstring, "\n";
```

Listing 5.3: Dynamic Updates mit Perl (aus `comp.protocols.tcp-ip.domains`)

5.3 In Shellscripts

Wenn Sie Shellscripts schreiben oder keinen Wert auf Performance legen, können Sie zur Namensauflösung die Abfragetools aufrufen, die in Kapitel 4 beschrieben werden. Das Programm `host` eignet sich dafür wegen seiner vielen Optionen besonders gut.

Die aktuelle `host`-Version [`ftp://ftp.weird.com/pub/local/host.tar.gz`] enthält verschiedene Beispiel-Shellscripts:

- `nslookup.sh` emuliert das Programm `nslookup`.
- `rblookup.sh` testet eine IP-Adresse gegen verschiedene Realtime-Blacklists (RBLs) (Spam-Serverlisten).
- `nscheck.sh` vergleicht die NS-Records einer Zone mit ihrer Delegation.
- `mxlookup.sh` vergleicht die MX-Records einer Zone auf ihren verschiedenen Nameservern.

Teil II

Nameserver-Konfiguration

6 Die BIND-Konfigurationsdatei (named.conf)

Die zentrale Konfigurationsdatei für den BIND-Nameserver ist eine Textdatei, heißt `named.conf` und befindet sich im Verzeichnis `/etc/bind`, `/etc/named` oder direkt in `/etc` (abhängig von Betriebssystem und Distribution).

Sie bestimmt z.B., für welche Zonen der Nameserver zuständig ist, wo die Zonefiles liegen und welchen Rechnern der Nameserver rekursiv antworten darf.

Dieses Kapitel beschreibt

- die Konfiguration der BIND-Version 9.2.3, die vielen Linux-Distributionen beiliegt
- die neuen Möglichkeiten der Version 9.3.0, die am 23. September 2004 erschien und alle vorangegangenen BIND-Versionen ersetzen wird

6.1 Einlesen der Konfigurationsdatei

Änderungen, die Sie an der Konfigurationsdatei vornehmen, werden nicht sofort wirksam, sondern erst beim nächsten Neustart des `named`-Prozesses oder bei Aufruf von `rndc reload`.

Achtung: Prüfen Sie vor dem Einlesen der Konfigurationsdatei die Syntax mit dem Programm `named-checkconf`, das der BIND-Distribution beiliegt.

Mehr zum Thema: Der Neustart des Nameservers und der Aufruf von `rndc` wird in Abschnitt 11.2 beschrieben. `named-checkconf` wird in Abschnitt 13.5.1 beschrieben.

6.2 Tipps zur Notation

Die Konfigurationsdatei besteht aus

- *Anweisungen* (siehe Abschnitt 6.2.1)
- *Kommentaren* (siehe Abschnitt 6.2.2)
- Leerzeilen

Einige Anweisungen erfordern die Angabe einer *Adressliste* (siehe Abschnitt 6.2.3).

Achtung: Die Konfigurationsdatei muss mit einem Zeilenumbruch enden, sonst beendet sich BIND mit der Meldung `named.conf:XXX: unexpected end of input` (wobei `XXX` die Anzahl der Zeilen ist).

Die meisten Texteditoren (z.B. vim, emacs) fügen den Zeilenumbruch automatisch ein, bei anderen (z.B. notepad, wordpad) müssen Sie selbst eine Leerzeile am Ende der Datei anhängen.

6.2.1 Anweisungen

Anweisungen beginnen mit einem Schlüsselwort, gefolgt von Parametern, Unteranweisungen oder Adresslisten, und enden immer mit einem Semikolon.

Achtung: Vergessen Sie das Semikolon nicht!

Parameter müssen in *doppelten Anführungszeichen* (`""`) stehen, sofern es sich nicht um Schlüsselwörter (z.B. »`yes`«) handelt.

Unteranweisungen und Adresslisten müssen in *geschweiften Klammern* (`{}`) stehen.

Zur besseren Lesbarkeit können Sie zwischen den Bestandteilen einer Anweisung ein oder mehrere

- Leerzeichen,
- Tabulatorzeichen oder
- Zeilenumbrüche

einfügen. BIND unterscheidet *nicht* zwischen diesen Zeichen.

Beispiel:

```
options {
    directory "/var/named";
    allow-transfer {
       10/8;
       127/8;
    };
};

# ist identisch mit:

options{directory "/var/named";allow-transfer{10/8;127/8;};};
```

Listing 6.1: Formatierung von `named.conf`-Anweisungen

Listing 6.1 zeigt eine Anweisung (`options`), die zwei Unteranweisungen (`directory` und `allow-transfer`) in geschweiften Klammern enthält. Zur `directory`-Anweisung gehört ein Parameter (`"/var/named"`) in doppelten Anführungszeichen. Zur `allow-transfer`-Anweisung gehört eine Adressliste in geschweiften Klammern.

6.2.2 Kommentare

Sie können Ihre Konfigurationsdatei mit Kommentaren dokumentieren. BIND kennt drei verschiedene Kommentarstile:

- C-Stil: Der Kommentar steht zwischen `/*` und `*/` und kann Zeilenumbrüche enthalten.

  > **Achtung:** Versuchen Sie *nicht*, mehrere Kommentare dieses Stils ineinander zu schachteln, da der Kommentar für BIND beim ersten `*/` endet.

  Am besten verwenden Sie diesen Kommentarstil nur zum Auskommentieren größerer Blöcke.
- C++-Stil: Der Kommentar steht zwischen `//` und dem Zeilenende.
- Shell/Perl-Stil: Der Kommentar steht zwischen `#` und dem Zeilenende.

Beispiel:

In Listing 6.2 sind alle Zeilen, die *keinen* Kommentar darstellen, fett gesetzt:

```
# Forward-Mapping für das RBI-Netz (141.2.1/24):
zone rbi.informatik.uni-frankfurt.de IN {
    type master;
    file master/rbi;

    // Bei Änderungen die Secondaries anstupsen:
    notify yes; also-notify {141.2.1.3; 141.2.1.1;};

    /* auskommentiert, weil zurzeit jeder AXFR darf...
      allow-transfer {
          localhost;      // zum Debugging
          10/8;          //  das Testnetz im 3.OG
          141.2/16;     //   das Campusnetz
      };
    */
};
```

Listing 6.2: Beispiel einer Anweisung mit Kommentaren

6.2.3 Adresslisten

Einige Anweisungen erfordern die Angabe einer Adressliste, um beispielsweise den Zugriff auf eine Funktion des Nameservers einzuschränken.

Adresslisten setzen sich wie folgt zusammen:

```
Adressliste ::= { Adressangabe; [ Adressangabe; ...] }

Adressangabe ::=
[ ! ] (
   IPadresse [/Länge] |
   key Schlüsselname |
   acl_Name |
   { Adressliste }
)
```

Listing 6.3: Syntax von Adresslisten

Achtung: Vergessen Sie nicht das Semikolon, das nach jeder einzelnen Adressangabe und damit auch am Ende der Adressliste stehen muss!

Das Ausrufezeichen (`!`) invertiert die Aussage. BIND arbeitet die Adressangaben in der angegebenen Reihenfolge durch und stoppt beim ersten Treffer. Nur wenn vor diesem *kein* Ausrufezeichen steht, trifft die Adresse zu.

Die *Länge* gibt an, wie viele Bits der IP-Adresse verglichen werden sollen (beginnend beim Bit mit dem höchsten Wert). Wie in der CIDR-Notation üblich, können Sie die Bytes der Adresse, die außerhalb dieser Bits liegen, weglassen. Schreiben Sie z.B. `10/8` statt `10.0.0.0/8`.

Eine Adressangabe kann ACLs enthalten, die Sie zuvor mit der `acl{}`-Anweisung definiert hatten. Bitte beachten Sie, dass die ACLs `any`, `none`, `localhost` und `localnets` bereits vordefiniert sind.

Achtung: Die Anweisung `allow-query{ 10/8; !10.1/16;};` würde den Rechner `10.1.2.3` berechtigen, da die erste Adressangabe (`10/8`) zutrifft und kein Ausrufezeichen davor steht. Führen Sie deshalb zuerst speziellere Adressen vor allgemeineren Adressen auf und schreiben Sie besser `allow-query{ !10.1/16; 10/8;};`, wenn Sie das `10.1`er-Netz ausschließen möchten.

Achtung: `allow-update{ key geheim; 10/8;};` würde alle Rechner aus dem `10`er-Netz berechtigen, auch wenn diese den Schlüssel *nicht* besitzen. Wenn Sie nur die Rechner berechtigen möchten, die den Schlüssel besitzen UND aus dem `10`er-Netz stammen, müssen Sie die erste Adressangabe zweimal negieren, was zu `allow-update{ !{!key geheim;}; 10/8;};` führt.

Weitere Beispiele:

Anweisung	Erklärung
`allow-query {none;};`	Niemand darf zugreifen.
`allow-query {localnets;};`	Nur Rechner aus Netzen, in denen der Nameserver ein Interface hat, dürfen zugreifen.
`allow-query{ !10.1/16; 10/8; };`	Nur Rechner mit der IP-Adresse `10.*.*.*` (außer denen mit der IP-Adresse `10.1.*.*`) dürfen zugreifen.
`acl uninetz {141.2/16;}; allow-query{ !{ 10/8; 172.16/12; 192.168/24;}; uninetz; key "rbi-ns; };`	Nur Rechner mit öffentlichen Internet-Adressen (siehe RFC1918) dürfen zugreifen, wobei Rechner außerhalb des Uninetzes hierzu ein Passwort benötigen.

Tabelle 6.1: Beispiele zu Adresslisten

6.3 Anweisungen (BIND 9.2.3)

Eine Konfigurationsdatei für BIND 9.2.3 kann folgende Anweisungen enthalten:

Anweisung	Beschreibung
`acl`	gibt einer Adressliste einen Namen, den Sie in folgenden Anweisungen referenzieren können.
`controls`	konfiguriert die Zugriffsmöglichkeiten für das Admintool `rndc`.
`include`	liest Anweisungen aus einer weiteren Konfigurationsdatei ein.

Tabelle 6.2: Anweisungen in der Konfigurationsdatei

Anweisung	Beschreibung
`key`	beschreibt ein TSIG-Passwort, das der Nameserver zur Authentifizierung vorlegen oder erwarten kann.
`logging`	bestimmt, welche Nachrichten BIND in welche Logdateien schreibt.
`lwres`	konfiguriert die Zugriffsmöglichkeiten für die Lightweight Resolver Library.
`options`	konfiguriert die allgemeinen (d.h. `zone`-, `server`- und `view`-übergreifenden) Eigenschaften des Nameservers.
`server`	konfiguriert Eigenschaften eines anderen Nameservers, z.B. ob zur Kommunikation mit diesem ein Passwort erforderlich ist.
`trusted-keys`	beschreibt die öffentlichen DNSSEC-Schlüssel von Domains.
`view`	enthält Optionen und Anweisungen, die nur für ausgewählte Rechner/Netze gelten.
`zone`	beschreibt die Zuständigkeit des Nameservers für eine Zone, den Dateinamen des Zonefile und ggf. spezielle Eigenschaften dieser Zone.

Tabelle 6.2: Anweisungen in der Konfigurationsdatei (Fortsetzung)

Bitte beachten Sie:

- **Auswahl:** Alle Anweisungen sind optional. Die meisten Konfigurationsdateien enthalten nur die Anweisungen `zone` und `options`.
- **Mehrfachangaben:** Die Anweisungen `acl`, `include`, `key`, `lwres`, `server`, `view` und `zone` dürfen mehrmals in einer Konfigurationsdatei vorkommen, die Anweisungen `controls`, `logging`, `options` und `trusted-keys` nur einmal.
- **Reihenfolge:** Die `options`-Anweisung muss vor den `zone`- und `view`-Anweisungen und die `acl`-Anweisungen vor ihrer ersten Referenzierung stehen.
- **Schachtelung:** Viele Anweisungen können `include`-Anweisungen enthalten und die `view`-Anweisung kann `zone`-, `key`-, `server`- und `trusted-keys`-Anweisungen enthalten.

- **Ausschlüsse:** Wenn Sie mit Views arbeiten, darf keine `zone`-Anweisung außerhalb einer `view`-Anweisung stehen.

6.3.1 acl

Mit der `acl`-Anweisung geben Sie einer Adressliste einen Namen. Diesen Namen können Sie in *folgenden* Anweisungen überall dort einsetzen, wo eine Adressliste gefragt ist.

Achtung: Sie dürfen einen Namen nicht *vor* seiner Definition verwenden. Schreiben Sie die `acl`-Anweisungen daher am besten an den Anfang der Konfigurationsdatei.

Die `acl`-Anweisung hat folgende Syntax:

```
acl Name {
    Adressliste
};
```

Listing 6.4: Syntax der `acl`-Anweisung

Mehr zum Thema: Adresslisten werden in Abschnitt 6.2.3 beschrieben.

Achtung: Bitte beachten Sie, dass jede Adresse und die Adressliste selbst mit einem Semikolon endet.

Die folgenden ACLs sind bereits vordefiniert:

Name	Bedeutung
any	Alle Rechner
none	Kein Rechner

Tabelle 6.3: Vordefinierte ACLs

Name	Bedeutung
localhost	Der lokale Rechner
localnets	Die Rechner, die erreicht werden können, ohne den Dienst eines Routers in Anspruch zu nehmen.

Tabelle 6.3: Vordefinierte ACLs (Fortsetzung)

Beispiele:

```
///////////////// Definition einiger ACLs /////////////////
// Private IP-Adressen (laut RFC1918)
acl private1918 {
   10/8;
   172.16/12;
   192.168/24;
};
acl dmz {
   localnets;
};
// was übrig bleibt, ist dann wohl das Internet:
acl internet {
   !{private1918;dmz;localhost;};
};

////////////////////// ZUGRIFFSKONTROLLE ///////////////////
// Dieser interne Nameserver darf weder ins Internet
// zugreifen noch Anfragen aus dem Internet beantworten:
options {
  // ...
  blackhole { internet; };
};
```

Listing 6.5: Beispiel zur `acl`-Anweisung

Mehr zum Thema: Weitere Beispiele finden Sie in Abschnitt 12.5.1 bei den »Martian adresses«.

6.3.2 controls

Die `controls`-Anweisung bestimmt Zugriffsmöglichkeiten für das Admintool `rndc`.

Mit `rndc` teilen Sie BIND mit, dass er z.B. seine Zonefiles oder die Konfigurationsdatei neu einlesen oder sich beenden soll.

Wenn Sie es nicht anders angeben, greift `rndc` auf den TCP-Port 953 des Nameservers zu und authentifiziert sich mit dem Schlüssel aus `/etc/bind/rndc.key`.

Mehr zum Thema: Der Aufruf von `rndc` wird in Abschnitt 11.2 beschrieben.

Die `controls`-Anweisung hat folgende Syntax:

```
controls {
    inet
    (LocalIP|*|::)
        [ port Portnummer ]
        allow { Adressliste }
        keys { Schlüsselname; [ Schlüsselname; ...] }
    ;
    [ inet ... ;]
};
```

Listing 6.6: Aufruf der `controls`-Anweisung

Erklärung:

- Setzen Sie für *`LocalIP`* die IP-Adresse einer Netzwerkschnittstelle Ihres Nameservers ein, um den Zugriff über diese Netzwerkschnittstelle zu erlauben.
  `*` steht für alle IPv4- und `::` für alle IPv6-Netzwerkschnittstellen.
  Mit `127.0.0.1` (bei IPv6: `::1`) erlauben Sie nur lokalen Zugriff.

- Setzen Sie für *Portnummer* eine Zahl zwischen 0 und 65535 ein. Sie bestimmt die TCP-Portnummer, auf der BIND Anfragen von rndc entgegennimmt. Standardwert ist 953.
- Mit der allow-Anweisung bestimmen Sie, welche Rechner zum Zugriff berechtigt sind.
- In der keys-Anweisung müssen Sie mindestens einen Schlüssel angeben, damit rndc sich authentifizieren kann.

Mehr zum Thema: Adresslisten werden in Abschnitt 6.2.3 und Schlüssel in Abschnitt 6.3.4 beschrieben.

Beispiel:

```
key rndc-key {
    algorithm  hmac-md5;
    secret     "La/E5CjG9O+os1jq0a2jdA==";
};

controls {
    inet 127.0.0.1 port 953   // IPv4-loopback
         allow {localhost;}  keys {rndc-key;};
    inet ::1 port 953         // IPv6-loopback
         allow {localhost;}  keys {rndc-key;};
};
```

Listing 6.7: Beispiel zur controls-Anweisung

Achtung: Das Weglassen der controls-Anweisung reicht nicht aus, um den rndc-Zugriff vollständig zu sperren. Denn dann setzt BIND Standardwerte, die mit denen in Listing 6.7 übereinstimmen, und liest den Schlüssel aus /etc/bind/rndc.key.

Wenn Sie den rndc-Zugriff vollständig sperren möchten, schreiben Sie bitte controls{};.

6.3.3 include

Wenn BIND beim Einlesen der Konfigurationsdatei auf eine `include`-Anweisung trifft, fügt er die angegebene Datei an Ort und Stelle ein.

So können Sie z.B.

- die `key`-Anweisungen, die ja geheime Passwörter enthalten, in Dateien auslagern, die kein Anwender außer dem des Nameserver-Prozesses lesen kann.
- die Definition der Zonen einlesen, für die Sie einen sekundären Nameserver betreiben. Die Verteilung dieser Dateien könnten Sie per ftp/ssh/nfs usw. automatisieren.
- Ihre Konfigurationsdatei übersichtlicher gestalten.

Beispiele:

```
# Die ausgelagerten key-Anweisungen einlesen:
include "/etc/bind/named.keys";

# Die configs der fremden Zonen einlesen, für die
# wir secondary sind (kommen per ftp/scp):
include "/home/dns/zones.physik";
include "/home/dns/zones.chemie";
```

Listing 6.8: Beispiel zur `include`-Anweisung

Achtung: BIND unterstützt keine Wildcardzeichen (*, ?), um mehrere Dateien auf einmal einzubinden.

Sie können `include`-Anweisungen an beliebiger Stelle innerhalb anderer Anweisungen angeben.

6.3.4 key

Die `key`-Anweisung beschreibt einen TSIG-Schlüssel.

Mit TSIG-Schlüsseln kann BIND

- sich bei anderen Nameservern authentifizieren (siehe `server`-Anweisung in Abschnitt 6.3.8). Der Schlüssel wirkt dann wie ein Passwort.

- die Integrität einer Nachricht anhand ihrer Signatur sicherstellen.
- die Identität der Nameserver sicherstellen, die z.B. eine Abfrage oder ein Dynamic Update senden (siehe Adresslisten in Abschnitt 6.2.3).
- die Autorisierung der Programme
  - `rndc` (siehe `controls`-Anweisung in Abschnitt 6.3.2) und
  - `nsupdate` (siehe `allow-update`-Anweisung in den Abschnitten 6.3.7 und 6.3.11)

  sicherstellen.

Die `key`-Anweisung hat folgende Syntax:

```
key Schlüsselname {
    algorithm hmac-md5;
    secret "Passwort";
};
```

Listing 6.9: Syntax der key-Anweisung

Den somit definierten *`Schlüsselnamen`* können Sie in der `server`- oder `control`-Anweisung sowie in Adresslisten angeben.

Als *`Passwort`* können Sie jede Base64-kodierte Zeichenkette verwenden, deren Länge ein Vielfaches von 4 beträgt.

Mehr zum Thema: Base64 ist eine Abbildung, die jeweils drei 8-bittige Zeichen auf vier 6-bittige, druckbare ASCII-Zeichen [`A-Za-z0-9+/=`] abbildet.

Sie können gültige Passwörter mit den Programmen `mmencode`, `mimeencode`, `uuencode -m`, `dnssec-keygen` oder `recode l1..l1/b64` herstellen:

```
$ uuencode -m -
Achwie Gutdass Niemandweiss DassicH rUmPelStieLzCheN heiss
^D
begin-base64 644 -
QWNod2llIEd1dGRhc3MgTmllbWFuZHdlaXNzIERhc3NpY0ggclVtUGVsU3Rp
ZUx6Q2hlTiBoZWlzcw0K
====
$
```

Listing 6.10: Base64-Kodierung

(Dies führt zu `secret "QWNod2llIEd ... lzcw0K";`).

Achtung: Vermeiden Sie kurze Passwörter. Je länger ein Passwort ist, desto länger benötigt ein Hacker, um Ihr Passwort durch Ausprobieren herauszufinden (»brute-force-attack«).

Mehr zum Thema: Ein Beispiel zur `key`-Anweisung finden Sie in Listing 6.7 in Abschnitt 6.3.2 (`controls`-Anweisung).

Die Verwendung von TSIG-Schlüsseln und der Aufruf von `dnssec-keygen` werden in Abschnitt 12.4.1 beschrieben.

6.3.5 logging

Mit der `logging`-Anweisung bestimmen Sie, welche Nachrichten BIND in welche Logdateien schreibt.

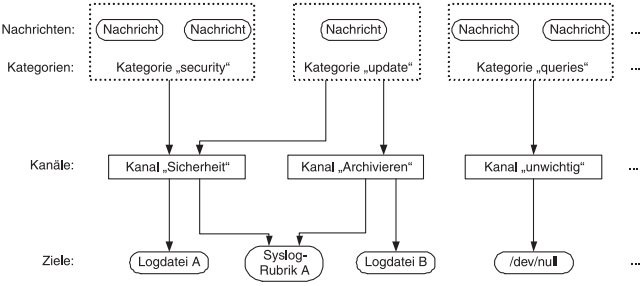

Abbildung 6.1: Beispiel zum Logging-Konzept

Konzept

- BIND generiert laufend verschiedene *Nachrichten*.
- Jede Nachricht gehört einer *Kategorie* (Themenbereich) an.
- Sie können für jede Kategorie festlegen, ob und in welche *Kanäle* BIND die Nachrichten weiterleitet.
- Sie können für jeden Kanal einen Eingangsfilter (Mindestpriorität), ein Ausgabeformat und verschiedene *Ziele* definieren.
- Jedes Ziel beschreibt den Namen einer Logdatei oder Syslog-Rubrik, in die BIND schreibt.

Syntax

Die `logging`-Anweisung hat folgende Syntax:

```
logging {
   [channel "Kanalname" { KanalEigenschaften; }; ]
   // ... weitere channel-Anweisungen;

   [category ("Kategoriename" | default) { KanalListe; }; ]
   // ... weitere category-Anweisungen;
};
```

Listing 6.11: Syntax der `logging`-Anweisung

Die `channel`- und `category`-Anweisungen werden in den folgenden Abschnitten beschrieben.

Standardverhalten

Wenn Sie *keine* `logging`-Anweisung angeben, geht BIND von folgenden Standardwerten aus:

```
logging {
     category "unmatched" { "null"; };
     category "default" {"default_syslog";"default_debug";};
};
```

Listing 6.12: Vorkonfiguration der `logging`-Anweisung

BIND sendet dann alle Nachrichten (außer den Warnungen zu Abfragen, die in keinen View passen oder deren Klasse unbekannt ist), die mindestens die Priorität `info` haben, in die Syslog-Rubrik `daemon`.

Der `syslogd` wird diese (je nach Konfiguration in `/etc/syslog.conf`) ignorieren oder sie in eine oder mehrere Logdateien (z.B. `/var/log/messages`, `/var/log/daemon.log` etc.) schreiben.

Wenn Sie BIND beim Start mit dem Parameter `-d` einen Debuglevel übergeben haben, schreibt BIND zusätzliche Debug-Nachrichten in die Datei `named.run`.

Kanäle

Die `channel`-Anweisung hat folgende Syntax:

```
channel Kanalname {
    (  file "Dateiname"
          [ versions ( Zahl | unlimited ) ]
          [ size Dateigröße [(K|M|G)] ]
       | syslog ( auth | authpriv | cron | daemon | ftp |
           kern | lpr | mail | news | syslog | user | uucp |
           local0 ... local7 )
       | stderr
       | null
    );
    [ severity ( critical | error | warning | notice |
             info | debug [ Debuglevel ] | dynamic ); ]
    [ print-category (yes|no); ]
    [ print-severity (yes|no); ]
    [ print-time (yes|no); ]
};
```

Listing 6.13: Syntax der `channel`-Anweisung

Erklärung:

Zuerst geben Sie das Ziel an, in das BIND alle Nachrichten schreiben soll, die auf diesem Kanal eingehen und hinreichende Priorität haben:

Ziel	Erklärung
`file "`*`Dateiname`*`" [ versions (`*`Zahl`*`|unlimited) ] [ size `*`Dateigröße`*` [(K|M|G)] ];`	schreibt die Nachrichten in die angegebene Logdatei. Mit der `versions`-Option können Sie die Zahl der Logdateien begrenzen. Wenn Sie nur die `versions`-Option setzen, archiviert BIND die bisherigen Logdateien bei jedem Programmstart: Aus *`Dateiname`* wird *`Dateiname`*`.0`, aus *`Dateiname`*`.0` wird *`Dateiname`*`.1` usw. bis zu *`Dateiname.Zahl`*, die gelöscht wird. Mit der `size`-Option können Sie die Größe der Logdatei begrenzen. Wenn Sie nur die `size`-Option setzen, hört BIND auf, in die Datei zu schreiben, sobald sie die angegebene *`Dateigröße`* erreicht. Wenn Sie die Dateigröße in Kilobyte (*1024), Megabyte (*1024^2) und Gigabyte (*1024^3) angeben möchten, setzen Sie bitte kein Leerzeichen zwischen Wert und Einheit, z.B. `20M`. Wenn Sie die `versions`- *und* die `size`-Option setzen, archiviert BIND die Logdateien bei jedem Erreichen der Dateigröße.
`syslog ( auth | authpriv | cron | daemon | ftp | kern | lpr | mail | news | syslog | user | uucp | local0..local7 );`	schreibt die Nachrichten in den Socket `/dev/log`, der mit dem lokalen `syslogd` verbunden ist. BIND kennt die Priorität bereits; die Rubrik müssen Sie (z.B. `daemon`) jedoch selbst angeben. In `/etc/syslog.conf` können Sie die weitere Verarbeitung für jede Rubrik und Priorität festlegen.
`stderr;`	schreibt die Nachrichten nach `STDERR`. Dies kann beim Debugging helfen, wenn Sie `named` mit der Option `-f` aufrufen.
`null;`	verwirft die Nachricht.

Tabelle 6.4: Ziele der `channel`-Anweisung

Mit der `severity`-Option bestimmen Sie die Mindestpriorität, die eine Nachricht haben muss, damit dieser Kanal sie beachtet:

Schlüsselwort	Erklärung
`severity ( critical | error | warning | notice | info | debug [ Level ] | dynamic );`	Zur Ordnung: Die wichtigste Priorität lautet `critical`, dann folgt `error` usw. bis zu `debug`, das die niedrigste Priorität hat. Bei `debug` können Sie eine natürliche Zahl als *Level* angeben (z.B. `debug 3`). Sie gibt an, bis zu welchem Debuglevel (`-d`-Option beim Aufruf von `named`) BIND Nachrichten in diesen Kanal schreibt. `dynamic` entspricht dem aktuellen Debuglevel (siehe `-d`-Option beim Aufruf von `named`). Wählen Sie diese Priorität, wenn Sie *alle* Nachrichten des aktuellen Debuglevels protokollieren möchten.

Tabelle 6.5: `severity`-Option der `channel`-Anweisung

Syslog protokolliert automatisch die Kategorie, Priorität und die Zeit. Bei Kanälen, die nicht über Syslog laufen, können Sie das Ausgabeformat mit den folgenden Optionen steuern:

Schlüsselwort	Erklärung
`print-category (yes|no);`	gibt die Kategorie mit aus.
`print-severity (yes|no);`	gibt die Priorität mit aus.
`print-time (yes|no);`	gibt Datum und Uhrzeit am Zeilenanfang aus. Dies ist bei Verwendung der `file`-Option sehr zu empfehlen.

Tabelle 6.6: Ausgabe-Optionen der `channel`-Anweisung

Die folgenden Channels sind vorkonfiguriert:

```
channel default_syslog {
    syslog daemon;      // Sendet an Syslog unter
                        // der Rubrik "daemon".
    severity info;      // Nur Nachrichten der Priorität
```

```
                        // info und wichtiger berücksichtigen.
};

channel default_debug {
    file "named.run";   // Schreibt in die Datei named.run
                        // im (beim Start von named)
                        // aktuellen Verzeichnis.
                        // Vorsicht: Wenn Sie named mit der
                        // Option -f aufgerufen haben, schreibt
                        // BIND stattdessen auf STDERR.
    severity dynamic;   // Nur etwas ausgeben, wenn BIND
                        // im Debugmodus (named -d ...) läuft.
 };

channel default_stderr {
    stderr;             // Schreibt nach STDERR.
    severity info;      // Nur Nachrichten der Priorität
                        // info und wichtiger berücksichtigen.
};

channel null {
   null;                // Nichts schreiben.
};
```

Listing 6.14: Vorkonfigurierte Kanäle

Achtung: Die vorkonfigurierten Kanäle können nicht geändert werden.

Kategorien

Mit der `category`-Anweisung weisen Sie den Nachrichten einer Kategorie einen oder mehrere Ausgabe-Kanäle zu.

Die `category`-Anweisung hat folgende Syntax:

```
category ( Kategoriename | default ){
    [ Kanalname; ...]
};
```

Listing 6.15: Syntax der `category`-Anweisung

BIND teilt die Nachrichten in folgende Kategorien ein:

Kategorie	Thema
`client`	Verarbeitung empfangener Abfragen
`config`	Feststellungen beim Einlesen der Konfigurationsdatei
`database`	Zugriffe auf die interne Datenbank, in der BIND Zonefiles und weitere Records cachet.
`default`	Alle Kategorien, zu denen keine `category`-Anweisung besteht. Diese Kategorie ist vorkonfiguriert, Details: siehe unten.
`dispatch`	Interne Verarbeitung der Abfragen im Nameserver
`dnssec`	Durchführung der TSIG- und DNSSEC-Protokolle
`general`	Alle sonstigen Nachrichten und die, die noch nicht eingeteilt wurden
`interfacemgr`	Netzschnittstellen, die aktiviert oder deaktiviert werden
`lame-servers`	Delegationen auf Nameserver, die sich für eine Zone nicht zuständig fühlen. Diese Situation entsteht durch Fehler bei der Delegation der Zone oder der Konfiguration des Nameservers.
`lwresd`	Kommunikation mit der Lightweight Resolver Library
`network`	Zugriffe auf das Netz
`notify`	Notify-Nachrichten (Bestandteil der Kommunikation zwischen primären und sekundären Nameservern)
`queries`	Die Details aller erhaltenen Abfragen. Diese Kategorie ist vorkonfiguriert, Details: siehe unten.
`resolver`	Rekursive Abfragen
`security`	Verweigerte und genehmigte Zugriffsversuche

Tabelle 6.7: Logging-Kategorien

Kategorie	Thema
`unmatched`	Erhaltene Abfragen zu Records, die in keinen der Views passen oder deren Klasse unbekannt ist. Diese Kategorie ist vorkonfiguriert, Details: siehe unten.
`update`	Dynamic Updates
`xfer-in`	Zonetransfer-Requests, die der Nameserver empfängt
`xfer-out`	Zonetransfer-Requests, die der Nameserver sendet

Tabelle 6.7: Logging-Kategorien (Fortsetzung)

Die `default`-Kategorie hat eine besondere Funktion: Sie erhält die Nachrichten aller Kategorien, zu denen es in der Konfigurationsdatei *keine* `category`-Anweisung gibt. Wenn Sie die `default`-Kategorie nicht selbst definieren, verwendet BIND die Einstellungen der voreingestellten `default`-Kategorie.

Achtung: Die Einstellungen der `default`-Kategorie greifen nur für die Kategorien, zu denen es *keine* `category`-Anweisung gibt. Wenn Sie die Nachrichten einer Kategorie, zu der es bisher noch keine `category`-Anweisung gab, *zusätzlich* in einen weiteren Kanal leiten möchten, sollten Sie die Kanäle der `default`-Kategorie in die selbst konfigurierte Kategorie übernehmen.

Die folgenden Kategorien sind vorkonfiguriert, können jedoch (im Gegensatz zu vorkonfigurierten Kanälen) ersetzt werden:

```
// Ausgabekanäle für die Requests, die in keinen
// der Views passen oder deren Klasse unbekannt ist:
category unmatched {
      null;  // verwerfen
};

// Ausgabekanäle für die Details der erhaltenen Abfragen:
category queries {
      null;  // verwerfen
```

```
};

// Ausgabekanäle für alle weiteren Nachrichten-Kategorien:
category default {
      default_syslog;  // siehe Listing 6.14
      default_debug;   //
};
```

Listing 6.16: Vorkonfigurierte Kategorien

Beispiele

Sie können die Nachrichten einer Kategorie *zusätzlich* in eine separate Logdatei schreiben:

```
logging {
  channel my_security_channel {
      file my_security_file;
      severity info;
  };
  category security {
      my_security_channel;
      default_syslog;
      default_debug;
  };
};
```

Listing 6.17: Wichtige Nachrichten zusätzlich in eine separate Logdatei schreiben

Sie können die Protokollierung aller Nachrichten einer Kategorie abschalten:

```
logging {
  category xfer-out { null; };
  category notify { null; };
};
```

Listing 6.18: Logging für einige Kategorien deaktivieren

Sie können die Protokollierung vollständig deaktivieren, was möglicherweise auf Nameservern sinnvoll ist, die keine lokale Festplatte besitzen:

```
logging {
  category default { null; };
};
```

Listing 6.19: Logging komplett deaktivieren

Sie sollten besser aber einen *Loghost* nutzen, um die Nachrichten vom `syslogd` des Nameservers zu empfangen und zu protokollieren.

Mehr zum Thema: Die Einrichtung eines Loghosts wird in der Manpage zu `syslogd.conf` beschrieben.

6.3.6 lwres

Wenn Sie die `lwres`-Anweisung in der Datei `named.conf` angeben, übernimmt der `named` die Aufgabe des `lwresd`: die Kommunikation mit den Programmen, die gegen die Lightweight Resolver Library gelinkt sind.

Mehr zum Thema: Die Lightweight Resolver Library wird in Abschnitt 3.2 beschrieben.

Die `lwres`-Anweisung hat folgende Syntax:

```
lwres {
    [ listen-on { IP-Adresse [port Portnummer]; [... ;] }; ]
    [ view "Viewname"; ]
    [ search { "Domainname" ; [... ;] }; ]
    [ ndots Zahl; ]
};
```

Listing 6.20: Syntax der `lwres`-Anweisung

Erklärung:

- Mit `listen-on` ermöglichen Sie den Zugriff über die Netzwerkschnittstelle mit der angegebenen *IP-Adresse* auf der angegebenen *Portnummer*. Sie können mehrere IP-Adressen und Ports angeben.

Wenn Sie die port-Option weglassen, lauscht BIND auf Port 921.
Wenn Sie die listen-on-Anweisung weglassen, lauscht BIND auf 127.0.0.1:921.

- Mit view erreichen Sie, dass BIND die eingehenden Anfragen mit der Konfiguration aus dem angegebenen Viewname bearbeitet.
  Wenn Ihre Konfigurationsdatei keine Views definiert, können Sie die view-Anweisung nicht verwenden.
- Mit search können Sie eine Suchliste angeben, deren Wirkung mit der in /etc/resolv.conf übereinstimmt. Bitte beachten Sie, dass die Domainnamen in Anführungszeichen stehen müssen.
- Mit ndots können Sie bestimmen, ab wie vielen Punkten BIND einen Domainnamen zuerst *ohne* angehängte Suchliste probieren soll. Dies entspricht der ndots-Option in /etc/resolv.conf.

Mehr zum Thema: Die Anweisungen und Optionen der Datei /etc/resolv.conf werden in Abschnitt 3.1.1 beschrieben.

Achtung: Wenn Ihre Konfigurationsdatei mehrere view- und lwres-Anweisungen enthält, achten Sie bitte darauf, dass jede Kombination aus *IP-Adresse* und *Portnummer* nur einmal vorkommt.

6.3.7 options

Sie können das Verhalten von BIND mit vielen Optionen steuern.

Einige Optionen wirken sich auf den gesamten Nameserver aus, andere können Sie für jede Zone individuell einstellen. Mit der view-Anweisung können Sie die Optionen sogar abhängig vom anfragenden Rechner oder dem verwendeten TSIG-Schlüssel setzen.

Mehr zum Thema: Die view-Anweisung wird in Abschnitt 6.3.10 beschrieben.

Die options-Anweisung umgibt alle globalen Optionen. Sie hat folgende Syntax:

```
options{
   Option Parameter;
   [ Option Parameter; ...]  // weitere Optionen
};
```

Listing 6.21: Syntax der options-Anweisung

Diese Optionen werden in den folgenden Abschnitten beschreiben:

Option	Beschreibung
additional-from-auth	siehe Tabelle 6.13: Optionen zum Antwortformat
additional-from-cache	siehe Tabelle 6.13: Optionen zum Antwortformat
allow-notify	siehe Tabelle 6.10: Zone-Optionen für Security
allow-query	siehe Tabelle 6.10: Zone-Optionen für Security
allow-recursion	siehe Tabelle 6.13: Optionen zum Antwortformat
allow-transfer	siehe Tabelle 6.10: Zone-Optionen für Security
allow-update-forwarding	siehe Tabelle 6.10: Zone-Optionen für Security
allow-v6-synthesis	siehe Tabelle 6.13: Optionen zum Antwortformat und Tabelle 6.24: Neue Optionen in BIND 9.3.0
also-notify	siehe Tabelle 6.11: Optionen zum Zonetransfer
alt-transfer-source[-v6]	siehe Tabelle 6.24: Neue Optionen in BIND 9.3.0
auth-nxdomain	siehe Tabelle 6.13: Optionen zum Antwortformat
avoid-(4|6)-udp-ports	siehe Tabelle 6.24: Neue Optionen in BIND 9.3.0
blackhole	siehe Tabelle 6.18: Sonstige Optionen
check-names	siehe Tabelle 6.24: Neue Optionen in BIND 9.3.0
cleaning-interval	siehe Tabelle 6.14: Optionen zu Performance- und Speichertuning

Tabelle 6.8: Übersicht aller Optionen

Option	Beschreibung
coresize	siehe Tabelle 6.14: Optionen zu Performance- und Speichertuning
datasize	siehe Tabelle 6.14: Optionen zu Performance- und Speichertuning
deallocate-on-exit	dient der Kompatibilität mit BIND8-Konfigurationsdateien und wird von BIND9 ignoriert.
dialup	siehe Tabelle 6.11: Optionen zum Zonetransfer
directory	siehe Tabelle 6.17: Optionen zu Pfad- und Dateinamen
disable-algorithms	siehe Tabelle 6.24: Neue Optionen in BIND 9.3.0
dnssec-enable	siehe Tabelle 6.24: Neue Optionen in BIND 9.3.0
dnssec-lookaside	siehe Tabelle 6.24: Neue Optionen in BIND 9.3.0
dnssec-must-be-secure	siehe Tabelle 6.24: Neue Optionen in BIND 9.3.0
dual-stack-servers	siehe Tabelle 6.24: Neue Optionen in BIND 9.3.0
dump-file	siehe Tabelle 6.17: Optionen zu Pfad- und Dateinamen
edns-udp-size	siehe Tabelle 6.24: Neue Optionen in BIND 9.3.0
fake-iquery	dient der Kompatibilität mit BIND8-Konfigurationsdateien und wird von BIND9 ignoriert.
fetch-glue	dient der Kompatibilität mit BIND8-Konfigurationsdateien und wird von BIND9 ignoriert.
files	siehe Tabelle 6.14: Optionen zu Performance- und Speichertuning
flush-zones-on-shutdown	siehe Tabelle 6.24: Neue Optionen in BIND 9.3.0
forward	siehe Tabelle 6.9: Allgemeine Zone-Optionen
forwarders	siehe Tabelle 6.9: Allgemeine Zone-Optionen

Tabelle 6.8: Übersicht aller Optionen (Fortsetzung)

Option	Beschreibung
has-old-clients	dient der Kompatibilität mit BIND8-Konfigurationsdateien und wird von BIND9 ignoriert.
heartbeat-interval	siehe Tabelle 6.15: Globale Optionen zu Zonetransfers
hostname	siehe Tabelle 6.24: Neue Optionen in BIND 9.3.0
host-statistics	dient der Kompatibilität mit BIND8-Konfigurationsdateien und wird von BIND9 ignoriert.
interface-interval	siehe Tabelle 6.16: Optionen zum Netzverhalten
ixfr-from-differences	siehe Tabelle 6.24: Neue Optionen in BIND 9.3.0
key-directory	siehe Tabelle 6.24: Neue Optionen in BIND 9.3.0
lame-ttl	siehe Tabelle 6.14: Optionen zu Performance- und Speichertuning
listen-on	siehe Tabelle 6.16: Optionen zum Netzverhalten
listen-on-v6	siehe Tabelle 6.16: Optionen zum Netzverhalten
maintain-ixfr-base	dient der Kompatibilität mit BIND8-Konfigurationsdateien und wird von BIND9 ignoriert.
match-mapped-addresses	siehe Tabelle 6.18: Sonstige Optionen
max-cache-size	siehe Tabelle 6.14: Optionen zu Performance- und Speichertuning
max-cache-ttl	siehe Tabelle 6.14: Optionen zu Performance- und Speichertuning
max-ixfr-log-size	dient der Kompatibilität mit BIND8-Konfigurationsdateien und wird von BIND9 ignoriert.
max-journal-size	siehe Tabelle 6.24: Neue Optionen in BIND 9.3.0
max-ncache-ttl	siehe Tabelle 6.14: Optionen zu Performance- und Speichertuning
max-refresh-time	siehe Tabelle 6.11: Optionen zum Zonetransfer

Tabelle 6.8: Übersicht aller Optionen (Fortsetzung)

Option	Beschreibung
max-retry-time	siehe Tabelle 6.11: Optionen zum Zonetransfer
max-transfer-idle-in	siehe Tabelle 6.11: Optionen zum Zonetransfer
max-transfer-idle-out	siehe Tabelle 6.11: Optionen zum Zonetransfer
max-transfer-time-in	siehe Tabelle 6.11: Optionen zum Zonetransfer
max-transfer-time-out	siehe Tabelle 6.11: Optionen zum Zonetransfer
memstatistics-file	siehe Tabelle 6.24: Neue Optionen in BIND 9.3.0
minimal-responses	siehe Tabelle 6.13: Optionen zum Antwortformat
min-refresh-time	siehe Tabelle 6.11: Optionen zum Zonetransfer
min-retry-time	siehe Tabelle 6.11: Optionen zum Zonetransfer
min-roots	ist bis einschließlich Version 9.3.0 noch nicht in BIND implementiert.
multi-master	siehe Tabelle 6.24: Neue Optionen in BIND 9.3.0
multiple-cnames	dient der Kompatibilität mit BIND8-Konfigurationsdateien und wird von BIND9 ignoriert.
named-xfer	dient der Kompatibilität mit BIND8-Konfigurationsdateien und wird von BIND9 ignoriert.
notify	siehe Tabelle 6.11: Optionen zum Zonetransfer
notify-source[-v6]	siehe Tabelle 6.11: Optionen zum Zonetransfer
pid-file	siehe Tabelle 6.17: Optionen zu Pfad- und Dateinamen und Tabelle 6.24: Neue Optionen in BIND 9.3.0
port	siehe Tabelle 6.16: Optionen zum Netzverhalten
preferred-glue	siehe Tabelle 6.24: Neue Optionen in BIND 9.3.0
provide-ixfr	siehe Tabelle 6.12: Optionen zum Serververhalten
querylog	siehe Tabelle 6.24: Neue Optionen in BIND 9.3.0
query-source[-v6]	siehe Tabelle 6.16: Optionen zum Netzverhalten

Tabelle 6.8: Übersicht aller Optionen (Fortsetzung)

Option	Beschreibung
`random-device`	siehe Tabelle 6.17: Optionen zu Pfad- und Dateinamen
`recursing-file`	siehe Tabelle 6.24: Neue Optionen in BIND 9.3.0
`recursion`	siehe Tabelle 6.13: Optionen zum Antwortformat
`recursive-clients`	siehe Tabelle 6.14: Optionen zu Performance- und Speichertuning
`request-ixfr`	siehe Tabelle 6.12: Optionen zum Serververhalten
`rfc2308-type1`	ist bis einschließlich Version 9.3.0 noch nicht in BIND implementiert.
`root-delegation-only`	siehe Tabelle 6.24: Neue Optionen in BIND 9.3.0
`rrset-order`	siehe Tabelle 6.24: Neue Optionen in BIND 9.3.0
`serial-queries`	dient der Kompatibilität mit BIND8-Konfigurationsdateien und wird von BIND9 ignoriert.
`serial-query-rate`	siehe Tabelle 6.14: Optionen zu Performance- und Speichertuning
`server-id`	siehe Tabelle 6.24: Neue Optionen in BIND 9.3.0
`sig-validity-interval`	siehe Tabelle 6.9: Allgemeine Zone-Optionen
`sortlist`	siehe Tabelle 6.16: Optionen zum Netzverhalten
`stacksize`	siehe Tabelle 6.14: Optionen zu Performance- und Speichertuning
`statistics-file`	siehe Tabelle 6.17: Optionen zu Pfad- und Dateinamen
`statistics-interval`	ist bis einschließlich Version 9.3.0 noch nicht in BIND implementiert.
`tcp-clients`	siehe Tabelle 6.14: Optionen zu Performance- und Speichertuning
`tcp-listen-queue`	siehe Tabelle 6.24: Neue Optionen in BIND 9.3.0

Tabelle 6.8: Übersicht aller Optionen (Fortsetzung)

Option	Beschreibung
tkey-dhkey	siehe Tabelle 6.17: Optionen zu Pfad- und Dateinamen
tkey-domain	siehe Tabelle 6.17: Optionen zu Pfad- und Dateinamen
topology	ist bis einschließlich Version 9.3.0 noch nicht in BIND implementiert.
transfer-format	siehe Tabelle 6.12: Optionen zum Serververhalten
transfers-in	siehe Tabelle 6.15:6 Globale Optionen zu Zonetransfers
transfer-source[-v6]	siehe Tabelle 6.11: Optionen zum Zonetransfer
transfers-out	siehe Tabelle 6.15: Globale Optionen zu Zonetransfers
transfers-per-ns	siehe Tabelle 6.15: Globale Optionen zu Zonetransfers
treat-cr-as-space	dient der Kompatibilität mit BIND8-Konfigurationsdateien und wird von BIND9 ignoriert.
use-alt-transfer-source	dient der Kompatibilität mit BIND8-Konfigurationsdateien und wird von BIND9 ignoriert.
use-id-pool	dient der Kompatibilität mit BIND8-Konfigurationsdateien und wird von BIND9 ignoriert.
use-ixfr	dient der Kompatibilität mit BIND8-Konfigurationsdateien und wird von BIND9 ignoriert.
version	siehe Tabelle 6.18: Sonstige Optionen und Tabelle 6.24: Neue Optionen in BIND 9.3.0
zone-statistics	siehe Tabelle 6.9: Allgemeine Zone-Optionen

Tabelle 6.8: Übersicht aller Optionen (Fortsetzung)

Zonen-spezifische Optionen

Dieser Abschnitt enthält Optionen, die Sie auch innerhalb einer zone-Anweisung angeben können. Wenn BIND eine Abfrage zu einer konfigurierten Zone

bearbeitet, bevorzugt er die Optionen aus der `zone`-Anweisung vor denen aus der `options`-Anweisung.

Mehr zum Thema: Die `zone`-Anweisung wird in Abschnitt 6.3.11 beschrieben.

Allgemein

Option	Beschreibung
! `forward (only|first);`	leitet Abfragen an die Nameserver weiter, die Sie mit der `forwarders`-Anweisung konfiguriert haben. Der als »Forwarder« bezeichnete Nameserver wird oft vom Internetprovider betrieben und kann viele Anfragen aus seinem Cache beantworten und so Ihre Zugriffe auf das DNS beschleunigen. Zonen, für die BIND autoritativ (d.h. primärer oder sekundärer Nameserver) ist, und Records, die sich bereits im Cache befinden, sind von der Weiterleitung ausgenommen. `only`: Wenn keiner der Forwarder antwortet, meldet BIND einen Fehler zurück. `first`: Wenn keiner der Forwarder antwortet, probiert BIND die Auflösung über die Root-Nameserver. **Standardwert:** `first`
! `forwarders { IP-Adresse [port Portnummer]; [ ...;] };`	konfiguriert die Nameserver für die Weiterleitung (siehe `forward`-Anweisung). **Standardwert:** Keine Forwarder
`sig-validity-interval Tage;`	gibt die Gültigkeit der DNSSEC-Signaturen an, die BIND automatisch für alle Records erstellt, die Sie mit Dynamic Updates in die Zone aufnehmen. Höchstwert: 10 Jahre (3660 Tage) **Standardwert:** 30 Tage.

Tabelle 6.9: Allgemeine Zone-Optionen

Option	Beschreibung
`zone-statistics (yes|no);`	bestimmt, ob BIND statistische Daten zu den eingehenden Abfragen erfassen soll. Sie können diese mit `rndc stats` auswerten. **Standardwert:** `yes`

Tabelle 6.9: Allgemeine Zone-Optionen (Fortsetzung)

Zugriffskontrolle

Option	Beschreibung
`allow-notify { Adressliste };`	bestimmt, auf wessen NOTIFY-Nachrichten BIND reagiert. **Standardwert:** nur auf NOTIFY-Nachrichten von primären Nameservern reagieren.
`allow-query { Adressliste };`	bestimmt, welche Rechner auf diesen Nameserver zugreifen dürfen. **Standardwert:** `any;` (das heißt, jeder darf)
`allow-transfer { Adressliste };`	bestimmt, welche Rechner Zonetransfers von diesem Nameserver anfordern dürfen. **Standardwert:** `any;` (das heißt, jeder darf)
`allow-update-forwarding { Adressliste };`	gibt an, von welchen Rechnern ein sekundärer Nameserver Zoneupdates entgegennimmt, um sie an den primären Nameserver weiterzuleiten. **Standardwert:** `none;` (das heißt, niemand darf)

Tabelle 6.10: Zone-Optionen für Security

Zonetransfers

Option	Beschreibung
`also-notify { IP-Adresse [port Portnummer]; [ ...;] };`	bestimmt, welchen Nameservern BIND bei Änderung lokaler Zonen NOTIFY-Nachrichten schicken soll (bei `notify yes` zusätzlich zu den »offiziellen« sekundären Nameservern laut NS-Records der Zone). **Standardwert:** keine

Tabelle 6.11: Optionen zum Zonetransfer

Option	Beschreibung
! `dialup ( yes | no | notify | notify-passive | refresh | passive);`	reduziert die Zugriffe auf das Netz, was bei Wählverbindungen Kosten sparen kann. `yes`: BIND sendet nur einmal pro `heartbeat-interval` für alle primären Zonen (auch wenn die Zone nicht geändert wurde) ein NOTIFY an die sekundären Nameserver und für alle sekundären Zonen (unabhängig vom Refresh-Timeout) einen SOA-Request an den primären Nameserver. `no`: sendet NOTIFY-Nachrichten bei jeder festgestellten Änderung des Zonefile und Refresh-Queries bei Ablauf des Refresh-Timeout. `notify`: sendet nur die NOTIFY-Nachrichten. `notify-passive`: sendet die NOTIFY-Nachrichten, aber *keine* Refresh-Queries. `refresh`: nur die Refresh-Queries. `passive`: unterlässt die automatischen Refresh-Queries. **Standardwert:** `no`
`max-refresh-time` *Sekunden*;	Die obere Grenze für das Refresh-Timeout in SOA-Records. **Standardwert:** unbegrenzt
`max-retry-time` *Sekunden*;	Die obere Grenze für das Retry-Timeout in SOA-Records. **Standardwert:** unbegrenzt
`max-transfer-idle-in` *Minuten*;	bestimmt, nach wie vielen Minuten ohne Fortschritt BIND den Download einer Zone vom primären Nameserver abbricht. Höchstwert: 28 Tage (40320 Minuten) **Standardwert:** 60 Minuten

Tabelle 6.11: Optionen zum Zonetransfer (Fortsetzung)

Option	Beschreibung
`max-transfer-idle-out` *Minuten*;	bestimmt, nach wie vielen Minuten ohne Fortschritt BIND das Senden einer Zone abbricht. Höchstwert: 28 Tage (40320 Minuten) **Standardwert:** 60 Minuten
`max-transfer-time-in` *Minuten*;	Die Zeit, nach der BIND den Download einer Zone von einem anderen Nameserver spätestens abbricht. Höchstwert: 28 Tage (40320 Minuten). **Standardwert:** 120 Minuten
`max-transfer-time-out` *Minuten*;	Die Zeit, nach der BIND eingehende Zonetransfer-Abfragen spätestens abbricht. Höchstwert: 28 Tage (40320 Minuten). **Standardwert:** 120 Minuten
`min-refresh-time` *Sekunden*;	Die untere Grenze für das Refresh-Timeout in SOA-Records. **Standardwert:** 0
`min-retry-time` *Sekunden*;	Die untere Grenze für das Retry-Timeout in SOA-Records. **Standardwert:** 0
`notify (yes|no|explicit);`	bestimmt, ob BIND für seine primären Zonen beim Starten und bei Änderung des Zonefiles NOTIFY-Nachrichten an die sekundären Nameserver sendet. `yes`: NOTIFY-Nachrichten an die Nameserver aus `also-notify` und den NS-Records der Zone senden. `no`: keine NOTIFY-Nachrichten senden. Die Slave-Server erfahren von der Änderung erst nach Ablauf des Refresh-Timeout. `explicit`: NOTIFY-Nachrichten nur an die Nameserver senden, die mit `also-notify` definiert sind. **Standardwert:** `yes`

Tabelle 6.11: Optionen zum Zonetransfer (Fortsetzung)

Option	Beschreibung
`notify-source[-v6]` (*IP-Adresse*|*) [port (*Portnummer*|*)] ;	bestimmt die Source-IP-Adresse und den Source-UDP-Port der NOTIFY-Nachrichten, von der BIND an sekundäre Nameserver sendet. Wenn Sie * anstelle der *IP-Adresse* angeben, verwendet BIND die IP-Adresse der Netzwerkschnittstelle, die den kürzesten Weg zum sekundären Nameserver hat. Wenn Sie keinen Port oder * anstelle der *Portnummer* angeben, verwendet BIND einen zufälligen UDP-Port größer 1024. **Achtung:** Diese IP-Adresse muss auf den sekundären Nameservern in der `masters`- oder einer `allow-notify`-Option freigeschaltet sein. **Standardwert:** `* port *`
`transfer-source[-v6]` (*IP-Adresse*|*) [port (*Portnummer*|*)] ;	bestimmt die Source-IP-Adresse der Zonetransfer-Requests sowie die Source-IP-Adresse und den Source-UDP-Port der Refresh-Queries, die BIND an die primären Nameserver sendet. Wenn Sie * anstelle der *IP-Adresse* angeben, verwendet BIND die IP-Adresse der Netzwerkschnittstelle, die den kürzesten Weg zum primären Nameserver hat. Wenn Sie keinen Port oder * anstelle der *Portnummer* angeben, verwendet BIND bei den Refresh-Queries einen zufälligen UDP-Port größer 1024. **Achtung:** Wenn Sie die Zonetransfer-Anfragen auf dem primären Nameserver mit der `allow-transfer`-Option auf eine Adressliste beschränkt haben, muss diese IP-Adresse dort enthalten sein. **Standardwert:** `* port *`

Tabelle 6.11: Optionen zum Zonetransfer (Fortsetzung)

Mehr zum Thema: Die NOTIFY-Nachricht wird in RFC1996 definiert.

Server-spezifische Optionen

Dieser Abschnitt enthält Optionen, die Sie auch innerhalb einer `server`-Anweisung angeben können. Wenn BIND mit einem Nameserver kommuniziert, den Sie in einer `server`-Anweisung beschreiben, bevorzugt er die Optionen aus der `server`-Anweisung vor denen aus der `options`-Anweisung.

Mehr zum Thema: Die `server`-Anweisung wird im nachfolgenden Abschnitt 6.3.8 beschrieben.

Option	Beschreibung
`provide-ixfr (yes|no);`	bestimmt, ob BIND als primärer Nameserver den Anfragen der sekundären Nameserver nach inkrementellen Zonetransfers nachkommt. `yes`: BIND sendet nur die geänderten Records (soweit möglich) `no`: BIND sendet die ganze Zone. **Standardwert:** `yes`
`request-ixfr (yes|no);`	bestimmt, ob BIND als sekundärer Nameserver versuchen soll, bei Zonetransfers anstelle der gesamten Zone nur die Änderungen vom primären Nameserver zu beziehen. **Standardwert:** `yes`
`transfer-format (one-answer | many-answers );`	bestimmt das Format der Zonetransfers. `one-answer`: ist RFC1035-kompatibel und erzeugt für jeden Record eine separate DNS-Nachricht. **Achtung:** Dies ist erforderlich, wenn z.B. ein sekundärer Nameserver Ihrer Zone unter Microsoft DNS-Server läuft, da dieser nur Nachrichten bis 16 Kilobyte versteht. `many-answers`: ist das schnellere Verfahren, das jedoch nur zwischen BIND-Servern funktioniert. BIND packt so viele Records wie möglich in jede DNS-Nachricht. **Standardwert:** `many-answers`

Tabelle 6.12: Optionen zum Serververhalten

Mehr zum Thema: Inkrementelle Zonetransfers werden in RFC1995 definiert.

Antwortformat

Die Optionen der folgenden Abschnitte können ausschließlich in der `options{}`-Anweisung stehen.

Option	Beschreibung
`additional-from-auth (yes|no);`	bestimmt, ob BIND die Zusatzinformationen (z.B. die IP-Adresse bei NS-, MX-, CNAME-, DNAME-Records) mitsendet, wenn er für die Zone selbst zuständig ist. Auf diese Weise können weitere Abfragen vorweggenommen werden. **Standardwert:** `yes`
`additional-from-cache (yes|no);`	bestimmt, ob BIND die Zusatzinformationen (z.B. die IP-Adresse bei NS-, MX-, CNAME-, DNAME-Records) mitsendet, wenn diese in seinem Cache enthalten sind. Auf diese Weise können weitere Abfragen vorweggenommen werden. **Standardwert:** `yes`
`allow-v6-synthesis { Adressliste };`	bestimmt, für welche Clients BIND bei der rekursiven Abfrage eines AAAA-Records zusätzlich heimlich nach A6-Records suchen darf. **Standardwert:** `none;`
`auth-nxdomain (yes|no);`	bewirkt, dass BIND alle negativen Antworten als autoritativ kennzeichnet, selbst wenn er für diesen Record gar nicht autoritativ ist. Eine negative Antwort sagt aus, dass ein Record nicht existiert. **Standardwert:** `no`

Tabelle 6.13: Optionen zum Antwortformat

Option	Beschreibung
`minimal-responses (yes|no);`	verhindert, dass BIND mit den Antworten optionale Zusatz- und Autoritätsinformationen sendet. **Standardwert:** `no`

`recursion (yes|no);`
 	
`allow-recursion { Adressliste };`
	legt fest, ob und welchen Rechnern BIND rekursiv antwortet, wenn diese es wünschen.
	`recursion yes;`: BIND löst alle rekursiven Queries rekursiv auf und sendet das Endergebnis an den anfragenden Resolver/Nameserver.
	`recursion no;`: BIND sendet nur Antworten, die in seinen eigenen Zonen oder bereits im Cache stehen. In allen anderen Fällen verweist BIND auf die zuständigen Nameserver (falls bekannt) oder die Root-Nameserver.
	
	Achtung: Unabhängig von diesen Optionen kann jeder Client auf die Daten zugreifen, die sich bereits im Cache des Nameservers befinden.
	
	Achtung: Rekursive Anfragen stellen die größte Belastung für einen Nameserver dar und können zu Denial-of-Service-(DoS-)Situationen führen. Daher sollten Sie Rekursion auf Ihren autoritativen Nameservern nach Möglichkeit verbieten.
	
	Standardwert: `recursion yes; allow-recursion{any;);`

Tabelle 6.13: Optionen zum Antwortformat (Fortsetzung)

Performance/Speicher-Tuning

Option	Beschreibung
`cleaning-interval Minuten;`	bestimmt, in welchem Abstand BIND abgelaufene Records aus seinem Cache wirft. Geben Sie 0 an, um die Bereinigung zu deaktivieren. Höchstwert: 28 Tage (40320 Minuten) **Standardwert:** 60 Minuten

Tabelle 6.14: Optionen zu Performance- und Speichertuning

Option	Beschreibung
`coresize (`*`Dateigröße`*`[(K|M|G)]| unlimited | default );`	ermöglicht Ihnen, die Größe von Coredumps zu begrenzen. `unlimited` entspricht dem Wert aus `ulimit -c`. `default` entspricht dem einkompilierten Wert. Diese Option steht unter Windows *nicht* zur Verfügung. **Standardwert**: `default`
`datasize (`*`Dateigröße`*`[(K|M|G)]| unlimited | default );`	ermöglicht Ihnen, die Größe des Datensegments zu begrenzen. **Achtung:** Dies kann zu Abstürzen führen und ist nicht zu empfehlen. Wenn Sie den Speicherverbrauch von BIND begrenzen möchten, setzen Sie besser die Optionen `max-cache-size` und `recursive-clients`. Diese Option steht unter Windows *nicht* zur Verfügung.
`files ( `*`Zahl`*` [(K|M|G)] | unlimited | default );`	ermöglicht Ihnen, die Zahl gleichzeitig geöffneter Dateien zu begrenzen. `unlimited` entspricht dem Wert aus `ulimit`. `default` entspricht `unlimited`. Diese Option steht unter Windows *nicht* zur Verfügung. **Standardwert**: `unlimited`
`lame-ttl `*`Sekunden`*`;`	bestimmt, wie lange BIND sich merken soll, dass ein Nameserver ein Lame-Server ist. Als Lame-Server bezeichnet man Nameserver, die nicht antworten oder nicht für die Zonen zuständig sind, die ihnen delegiert wurden. Mit dem Wert `0` verhindern Sie, dass BIND sich festgestellte Lame-Server merkt. Dies kann jedoch zu Performanceproblemen führen. Höchstwert: 1800 Sekunden **Standardwert:** 600 Sekunden

Tabelle 6.14: Optionen zu Performance- und Speichertuning (Fortsetzung)

Option	Beschreibung
`max-cache-size ( Größe [(K|M|G)] | unlimited );`	ermöglicht Ihnen, die Größe des Cache zu begrenzen. Wenn Sie mit Views arbeiten, können Sie jedem View eine eigene Cache-Größe zuweisen. Wenn der Cache die angegebene Größe erreicht, wirft BIND die Records mit der kleinsten TTL zuerst heraus. `unlimited`: BIND wirft die Records erst bei Ablauf der TTL aus dem Cache. **Standardwert**: `unlimited`
`max-cache-ttl Sekunden;`	ermöglicht Ihnen, die maximale Verweildauer eines Records im Cache zu begrenzen. Nach Ablauf der Verweildauer wirft BIND die Records aus dem Cache (ungeachtet ihrer TTL). **Standardwert:** 604800 Sekunden (eine Woche).
`max-ncache-ttl Sekunden;`	ermöglicht Ihnen, die maximale Verweildauer negativer Antworten im Cache zu begrenzen. Nach Ablauf der Verweildauer wirft BIND die negative Antwort aus dem Cache (ungeachtet ihrer TTL aus dem SOA-Record). Höchstwert: 604800 Sekunden (eine Woche) **Standardwert:** 10800 Sekunden (3 Stunden)
`recursive-clients Zahl;`	ermöglicht Ihnen, die Zahl gleichzeitig bearbeiteter, rekursiver Queries zu begrenzen. Dies kann den Speicherverbrauch von BIND reduzieren, da jede rekursive Query etwa 20 Kbyte beansprucht. **Standardwert:** `1000`
`serial-query-rate Zahl;`	ermöglicht Ihnen, die Zahl der gleichzeitigen Refresh-Queries zu begrenzen, die BIND für seine sekundären Zonen (nach Ablauf der Refresh-Timeouts) an die primären Nameserver sendet. **Standardwert:** `20`
`stacksize (Dateigröße [(K|M|G)]| unlimited | default );`	ermöglicht Ihnen, die Größe des Stacks zu begrenzen. `unlimited` entspricht dem Wert aus `ulimit -s`. `default` entspricht dem einkompilierten Wert. Diese Option steht unter Windows *nicht* zur Verfügung. **Standardwert**: `default`

Tabelle 6.14: Optionen zu Performance- und Speichertuning (Fortsetzung)

Option	Beschreibung
`tcp-clients` *`Zahl`*`;`	ermöglicht Ihnen, die Zahl gleichzeitiger Abfragen zu begrenzen, die TCP verwenden. **Standardwert**: `100`

Tabelle 6.14: Optionen zu Performance- und Speichertuning (Fortsetzung)

Zonetransfers

Option	Beschreibung
`heartbeat-interval` *`Minuten`*`;`	bestimmt, in welchem Abstand BIND Refresh-Queries und ggf. Zonetransfer-Anfragen sendet, wenn Sie die `dialup`-Option gesetzt haben. Mit dem Wert `0` deaktivieren Sie die Refresh-Queries vollständig. Höchstwert: 40320 Minuten (28 Tage) **Standardwert**: `60` Minuten
`transfers-in` *`Zahl`*`;`	ermöglicht Ihnen, die erlaubte Zahl von Zonetransfers zu begrenzen, die BIND bei anderen Nameservern anfordert. **Standardwert**: `10`
`transfers-out` *`Zahl`*`;`	ermöglicht Ihnen, die erlaubte Zahl gleichzeitiger Zonetransfers, die von anderen Nameservern angefordert werden, zu begrenzen. **Standardwert**: `10`
`transfers-per-ns` *`Zahl`*`;`	ermöglicht Ihnen, die Zahl der Zonetransfers zu begrenzen, die BIND maximal gleichzeitig an *einen* anderen Nameserver stellt, um von diesem mehrere Zonen zu erhalten. Sie können diesen Wert mit der `transfers`-Option der `servers`-Anweisung für jeden primären Nameserver separat einstellen. **Standardwert**: `2`

Tabelle 6.15: Globale Optionen zu Zonetransfers

Netz

Option	Beschreibung
`interface-interval` *Minuten*;	bestimmt, in welchem Abstand BIND nach neuen Netzschnittstellen sucht, um darauf Verbindungen entgegenzunehmen (wenn die `listen-on`-Option es gestattet). Höchstwert: 28 Tage (40320 Minuten) **Standardwert:** `60` Minuten
`listen-on` [`port` *Portnummer*] { *Adressliste* };	bestimmt die IPv4-Netzschnittstellen und UDP/TCP-Ports, auf denen BIND Anfragen entgegennimmt. Sie können mehrere Ports öffnen, indem Sie diese Option mehrmals angeben. Wenn Sie für die *Adressliste* `any` einsetzen, reagiert BIND auf allen IPv4-Schnittstellen. **Standardwert**: `listen-on port 53 { any; };`
`listen-on-v6` [`port` *Portnummer*] { (`any`|`none`); };	bestimmt die UDP/TCP-Ports, auf denen BIND Anfragen entgegennimmt, die auf IPv6-Schnittstellen eintreffen. Sie können mehrere Ports öffnen, indem Sie diese Option mehrmals angeben. **Achtung:** Es ist *nicht* möglich, einzelne IPv6-Schnittstellen auszuwählen: Sie können entweder alle oder keine IPv6-Schnittstelle aktivieren. Ohne Angabe dieser Option ist der Zugriff über IPv6 *nicht* möglich. **Standardwert**: `listen-on-v6 port 53 { none; };`
`port` *Portnummer*;	gibt den UDP/TCP-Port an, auf dem BIND andere Nameserver anspricht *und* selbst DNS-Anfragen entgegennimmt. **Achtung:** Wenn Sie den voreingestellten Wert ändern, erreicht BIND die Nameserver im Internet nicht mehr. Verwenden Sie besser die `listen-on`-Option, wenn Sie nur den Port ändern möchten, auf dem BIND Anfragen entgegennimmt. **Standardwert:** `53`

Tabelle 6.16: Optionen zum Netzverhalten

Option	Beschreibung

`query-source[-v6]`
`  [ address`
`  (IP-`*`Adresse`*`|*) ]`
`  [ port`
`  (`*`Portnummer`*`|*) ];`
	bestimmt die Source-IP-Adresse und (bei UDP-Paketen) den Source-Port der Abfragen, die BIND an andere Nameserver sendet.
	Wenn Sie * anstelle der *IP-Adresse* angeben, verwendet BIND die IP-Adresse der Netzwerkschnittstelle, die den kürzesten Weg zu dem Nameserver hat.
	Wenn Sie keinen Port oder * anstelle der *Portnummer* angeben, verwendet BIND auch bei UDP-Paketen einen zufälligen Source-Port größer 1024.
	Standardwert: `* port *`

`sortlist {`
`  {`*`SourceNetz`*
`  [;`*`Bevorzugte-Netze`*`];};`
`  [...]`
`};`
	bewirkt, dass BIND Antworten, die mehrere IP-Adressen enthalten, vorsortiert.
	Der geschickte Einsatz dieser Option kann Netzverkehr reduzieren, da viele Anwendungen nur die erste Adresse derartiger Antworten beachten.
	SourceNetz bezeichnet das Netz, aus dem die Anfrage stammen muss, damit diese Option wirksam wird. Wenn Sie hierfür eine Adressliste angeben möchten, setzen Sie diese bitte in geschweifte Klammern.
	BevorzugteNetze nennt das Netz, das BIND an den Anfang der Antwort stellen soll (»bevorzugt«). Wenn Sie hierfür eine Adressliste angeben möchten, setzen Sie diese bitte in geschweifte Klammern.
	Wenn Sie `BevorzugteNetze` *nicht* angeben, bevorzugt BIND das `SourceNetz`.
	Beispiele:
	Mit `sortlist{{10/8;};};` erreichen Sie, dass BIND bei Anfragen aus dem Netz 10/8 die Einträge dieses Netzes an den Anfang der Antwort stellt.
	Mit `sortlist{10/8; {10/8; 192.168/16};};` erreichen Sie, dass BIND bei Anfragen aus dem Netz 10/8 die Einträge der Netze 10/8 und 192.168/16 an den Anfang der Antwort stellt.

Tabelle 6.16: Optionen zum Netzverhalten (Fortsetzung)

Option	Beschreibung

	Mit
	`sortlist {`
	`  {localhost; localnets;};`
	`  {localnets;};`
	`};`
	erreichen Sie, dass BIND bei Anfragen von Rechnern der direkt angeschlossenen Netze und lokalen Programmen die Adressen der direkt angeschlossenen Netze bevorzugt.

Tabelle 6.16: Optionen zum Netzverhalten (Fortsetzung)

Pfad- und Dateinamen

Option	Beschreibung
`directory "`*`Verzeichnis`*`";`	gibt das Verzeichnis an, auf das sich alle relativen Pfadangaben in der Konfigurationsdatei und den Zonefiles beziehen. **Standardwert:** das aktuelle Verzeichnis beim Start von BIND.
`dump-file "`*`Dateiname`*`";`	gibt den Pfad- und Dateinamen der Datei an, in die BIND seine Datenbank dumpt, wenn man `rndc dumpdb` aufruft. **Standardwert:** `./named_dump.db`
`pid-file "`*`Dateiname`*`";`	gibt den Pfad- und Dateinamen der PID-Datei an, die BIND beim Start anlegt. Die PID-Datei ist eine kleine Textdatei, die die Prozessnummer des `named`-Prozesses enthält. **Standardwert:** `/var/run/named.pid`
`random-device "`*`Dateiname`*`";`	gibt den Pfad- und Dateinamen des Zufallsgenerator-Devices an. BIND verwendet dieses zum Erstellen von Signaturen unter DNSSEC. Diese Option steht unter Windows *nicht* zur Verfügung. **Achtung:** Setzen Sie diese Option auf `/dev/urandom`, wenn Ihr Zufallsgenerator gelegentlich leer läuft. **Standardwert:** `/dev/random`

Tabelle 6.17: Optionen zu Pfad- und Dateinamen

Option	Beschreibung
`statistics-file "`*`Dateiname`*`";`	gibt den Pfad- und Dateinamen der Datei an, an die BIND statistische Werte anhängt, wenn man `rndc stats` aufruft. **Standardwert:** `./named.stats`
`tkey-dhkey "`*`Dateiname`*`" "`*`Schlüsselname`*`";`	gibt den Dateinamen des Diffie-Hellmann-Schlüssels an (ohne den Pfad und die Endung `.key` und `.private`), den BIND zum verschlüsselten Übertragen automatisch erstellter TSIG-Schlüssel im TKEY-Verfahren benötigt. Das ISC empfiehlt, als *`Schlüsselname`* den Hostnamen des Nameservers einzusetzen.
`tkey-domain "`*`Domain`*`";`	gibt den Namen einer Domain an, den BIND an alle TSIG-Schlüsselnamen anhängt, die er aufgrund von TKEY-Requests selbst erstellt. Das ISC empfiehlt, die Domain des Nameservers einzusetzen.

Tabelle 6.17: Optionen zu Pfad- und Dateinamen (Fortsetzung)

Sonstige

Option	Beschreibung
`blackhole { `*`Adressliste`*` };`	ermöglicht Ihnen, eine Liste von Rechnern anzugeben, mit denen BIND niemals kommunizieren wird. Setzen Sie diese Option, wenn Sie BIND vor den Antworten eines falsch konfigurierten Nameservers, den Zugriffen eines aufdringlichen Clients oder Hackern mit Martian addresses schützen möchten. **Mehr zum Thema:** Ein Beispiel finden Sie in Abschnitt 12.5.1 bei den »Martian adresses«. **Standardwert:** `none`
`match-mapped-addresses (yes|no);`	aktiviert einen Workaround für einen Fehler im Linux-Kernel, bei dem IPv4-Clients TCP-Pakete in IPv6 verborgen an den IPv4-Zugriffskontrollen vorbeischleusen können. **Standardwert:** `no`

Tabelle 6.18: Sonstige Optionen

Option	Beschreibung
`version "Text";`	bestimmt den Text, den BIND auf die Frage nach `version.bind` in der Klasse `CH` sendet. Setzen Sie diese Option auf einen unsinnigen Text (z.B. `Microsoft DOS 3.12 DNS Service`), um die genaue Version von BIND zu verschleiern, in der Hoffnung, dass dadurch Hacker verwirrt werden und länger brauchen, um die Sicherheitslücken Ihres Nameservers zu finden. **Standardwert:** `BIND 9.2.3`

Tabelle 6.18: Sonstige Optionen (Fortsetzung)

6.3.8 server

Mit der `server`-Anweisung können Sie die Kommunikation zu einem anderen Nameserver konfigurieren.

Die `server`-Anweisung hat folgende Syntax:

```
server IP-Adresse {
    Option;
    [ weitere Optionen; ]
};
```

Listing 6.22: Syntax der `server`-Anweisung

Folgende Optionen sind in der `server`-Anweisung möglich:

Option	Beschreibung
`bogus (yes|no);`	verhindert die Kommunikation mit diesem Nameserver. **Standardwert:** `no`
`edns (yes|no);`	bestimmt, ob BIND versuchen soll, das EDNS-Protokoll einzusetzen. Das EDNS-Protokoll ermöglicht DNS-Nachrichten, die länger als 512 Byte sind. **Standardwert:** `yes`

Tabelle 6.19: Server-Optionen

Option	Beschreibung
`transfers` *Zahl*`;`	ermöglicht Ihnen, die Zahl der gleichzeitigen Zonetransfers zu begrenzen, die von oder zu diesem Server stattfinden. **Standardwert:** der Wert der globalen `transfers-per-ns`-Option
`keys` {*Schlüsselname*`;`}`;`	bestimmt einen TSIG-Schlüssel, mit dem BIND alle Pakete *an* diesen Nameserver signiert und dessen Signatur BIND bei allen Paketen *von* diesem Nameserver erwartet. Sie müssen diesen Schlüssel per `key`-Anweisung definieren.

`provide-ixfr;`
`request-ixfr;`
`transfer-format;`
	sind auch als globale Option möglich und werden daher bei der `options`-Anweisung in Abschnitt 6.3.7 unter »Server-spezifische Optionen« beschrieben.

Tabelle 6.19: Server-Optionen (Fortsetzung)

6.3.9 trusted-keys

Mit der `trusted-keys`-Anweisung können Sie den öffentlichen DNSSEC-Schlüssel einer Zone selbst angeben.

Wenn Sie DNSSEC einsetzen möchten, ist diese Anweisung für alle Zonen erforderlich, deren übergeordnete Zone nicht signiert ist (insbesondere, wenn Sie eine eigene Root-Zone betreiben).

Die `trusted-keys`-Anweisung hat folgende Syntax:

```
trusted-keys {
    Domainname Flags Protokoll Algorithmus Schlüssel;
    [ ...; ]
};
```

Listing 6.23: Syntax der `trusted-keys`-Anweisung

Mehr zum Thema: `trusted-keys` und DNSSEC werden in Abschnitt 12.3 beschrieben.

6.3.10 view

Mit der `view`-Anweisung können Sie das Verhalten von BIND abhängig von der Frage und dem Fragenden konfigurieren.

Die `view`-Anweisung enthält Bedingungen, Optionen und Anweisungen. Wenn alle Bedingungen auf eine Abfrage zutreffen, verwendet BIND die angegebenen Optionen und Anweisungen bei der Bearbeitung.

Die `view`-Anweisung hat folgende Syntax:

```
view "Bezeichnung"  [Klasse] {

      // Eine oder mehrere der folgenden Bedingungen:
      [ match-clients { Adressliste } ; ]
      [ match-destinations { Adressliste } ; ]
      [ match-recursive-only (yes|no) ; ]

      // aktiviert folgende...
      [ Optionen der options-Anweisung; ...]
      [ zone-Anweisungen; ...]
      [ key-Anweisungen; ...]
      [ server-Anweisungen; ...]
      [ trusted-keys-Anweisungen; ...]
};
```

Listing 6.24: Syntax der `view`-Anweisung

Beschreibung der Bedingungen:

Bedingung	Beschreibung
`match-clients { Adressliste };`	trifft zu, wenn der anfragende Rechner eine passende IP-Adresse hat. **Standardwert:** `any;`

Tabelle 6.20: `view`-Bedingungen

Bedingung	Beschreibung
`match-destinations { Adressliste } ;`	trifft zu, wenn der abgefragte Rechner eine passende IP-Adresse hat. **Standardwert:** `any`;
`match-recursive-only (yes|no) ;`	`yes`: trifft auf alle rekursiven Abfragen zu. `no`: trifft auf alle Abfragen zu. **Standardwert:** `no`

Tabelle 6.20: view-Bedingungen (Fortsetzung)

Beispiel

Die folgende (fiktive) Konfigurationsdatei definiert einen primären Nameserver für die Zone `informatik.uni-frankfurt.de`. Bei Abfragen dieser Zone aus dem internen Netz greift er auf ein anderes Zonefile zu als bei Abfragen aus anderen Netzen.

```
options { directory "/etc/named"; };

view "vondrinnen" {
      // Dieser View gilt für Zugriffe aus dem internen Netz
      match-clients { 141.2/16; };

      // Rekursive Anfragen von internen Clients sind OK
      recursion yes;

      // Interne Clients dürfen alle Einträge der Zone sehen
      zone "informatik.uni-frankfurt.de" {
            type master;
            file "informatik-vondrinnen.db";
      };
};

view "vondraussen" {
      // Alle Abfragen, bei denen der vorhergehende View
      // nicht gegriffen hat, kommen wohl von draußen.
      match-clients { any; };
```

```
        // Keine rekursiven Abfragen von draußen zulassen:
        recursion no;

        // Externe Clients dürfen nur die öffentlichen
        // Einträge sehen:
        zone "informatik.uni-frankfurt.de" {
             type master;
             file "informatik-vondraussen.db";
        };
};
```

Listing 6.25: Beispiel zur `view`-Anweisung

6.3.11 zone

Mit der `zone`-Anweisung können Sie das Verhalten von BIND für jede Zone individuell festlegen.

Die `zone`-Anweisung hat folgende Syntax:

```
zone ZonenName [Klasse] {
  type ( master | slave | stub | forward | hint);
  (
      file "Dateiname"; [database "rbt";]   |
      database "Modul [Optionen]";
  );

  // Zonentyp-spezifische Angaben (s.u.)
  // (allow-update, update-policy, masters, forwarders)

  // Ggf. zusätzliche Optionen, die auch in der
  // options-Anweisung möglich sind (siehe Abschnitt
  // 6.3.7 unter »Zone-spezifische-Optionen«)
};
```

Listing 6.26: Syntax der `zone`-Anweisung

Wenn Sie eine Klasse angeben, setzt BIND diese bei allen Records des Zonefiles ein, deren Klasse nicht angegeben ist und auch nicht von einem der vorangegangenen Records übernommen werden kann. Wenn Sie *keine* Klasse angeben, setzt BIND `IN` ein oder (falls Sie mit Views arbeiten) die Klasse der umgebenden `view`-Anweisung.

Mit der `file`-Option geben Sie den Dateinamen des Zonefiles an.

Alternativ zur `file`-Option können Sie mit der `database`-Option ein Modul angeben, das BIND beim Zugriff auf diese Zone aufruft. Verschiedene Datenbankmodule werden weiter unten beschrieben.

Zonen-Typen

Es gibt fünf verschiedene Zonen-Typen:

Master-Zonen

Zonen vom Typ `master` definieren einen primären Nameserver.

Wenn Sie Dynamic Updates Ihres Zonefiles zulassen möchten, müssen Sie eine der folgenden Optionen setzen:

Option	Beschreibung
! `allow-update { Adressliste };`	bestimmt, wer Dynamic Updates auf eine primäre Zone senden darf. Die Authentifizierung erfolgt über die IP-Adresse, einen TSIG-Schlüssel oder beides. **Standardwert:** `none`
`update-policy { (grant|deny) (Schlüsselname|*) ( name Hostname| subdomain Domain| wildcard Muster| self Schlüsselname) [RecordTypen|ANY]; [...;] };`	erlaubt oder verbietet einem durch TSIG-Schlüssel authentifizierten Rechner das Update einzelner, mehrerer oder aller Records einer Subdomain.

Tabelle 6.21: Optionen für primäre Zonen

Mit den Optionen `notify` und `also-notify`, die in Abschnitt 6.3.7 beschrieben werden, können Sie die sekundären Nameserver schon vor Ablauf ihrer Refresh-Timeouts über die Änderung informieren.

Beispiel:

```
acl uninetz {127/8; 10/8; 141.2/16;};

zone "rbi.informatik.uni-frankfurt.de" IN {
    type master;
    file "master/rbi";
    notify yes;
    also-notify {141.2.1.3; 141.2.1.1;};
    allow-transfer {uninetz;};
};
```

Listing 6.27: Beispiel einer Master-Zone

Mehr zum Thema: Die NOTIFY-Nachricht wird in RFC1996 definiert.

Slave- und Stub-Zonen

Zonen vom Typ `slave` und `stub` definieren einen sekundären Nameserver.

Ein sekundärer Nameserver bezieht seine Zonen über Zonetransfer von den primären Nameservern.

`stub`-Zonen benötigen etwas weniger Speicher als `slave`-Zonen, da BIND bei ihnen nur die NS-, SOA- und Glue-Records lädt. (Glue-Records sind die A-Records zu den NS-Records, die in einer delegierten Subdomain der Zone liegen, für die sie zuständig sind).

`stub`-Zonen waren ursprünglich dafür vorgesehen, Delegationen von den Zonefiles in die Konfigurationsdatei zu verlagern und dabei Glue-Records einzusparen. In der Praxis haben sie sich jedoch wegen des hohen Wartungsaufwands nie richtig durchgesetzt.

Sie müssen die primären Nameserver mit der `masters`-Anweisung angeben:

Option	Beschreibung

`masters`
`  [port` *Portnummer*`]`
`{`
` IP-Adresse`
`  [port` *Portnummer*`]`
`  [key` *Schlüsselname*`];`
` [...;]`
`};`
	gibt die primären Nameserver an, von denen BIND das Zonefile per Zonetransfer beziehen wird.

Tabelle 6.22: Optionen für sekundäre Zonen

Beispiel:

```
zone "uni-frankfurt.de" IN {
    type slave;
    masters {141.2.22.74;};
    file "slave/uni";
};
```

Listing 6.28: Beispiel einer Slave-Zone

Forward-Zonen

Zonen vom Typ `forward` leiten alle Abfragen, die sie nicht aus dem Cache beantworten können, rekursiv an die angegebenen Nameserver weiter.

Hierzu müssen Sie die Optionen `forwarders` und `forward` setzen, die bei der `options`-Anweisung in Abschnitt 6.3.7 beschrieben sind.

Beispiel:

```
// alle Reverse-Lookups an die zentrale Netzverwaltung
// weiterleiten:
zone "in-addr.arpa" IN {
    type forward;
    forward first;
    forwarders { 141.2.22.74; 141.2.149.10; };
};
```

Listing 6.29: Beispiel einer Forward-Zone

Hint-Zonen

Zonen vom Typ `hint` enthalten die Root-Nameserver einer Klasse.

BIND aktualisiert sie bei jedem Start.

Die Root-Nameserver der Klasse IN sind bereits in BIND einkompiliert, so dass die Angabe einer `hint`-Zone nur noch dann erforderlich ist, wenn Sie

- eine *eigene* Root-Zone aufsetzen möchten. Das ist in einigen Fällen geschickt, allerdings verlieren Sie dabei die Verbindung zum Internet.
- möchten, dass BIND Änderungen, die er beim routinemäßigen Zonetransfer der Root-Zone feststellt, in ein lokales Zonefile schreibt und somit beim nächsten Start darauf aufsetzen kann. Sie können das Zonefile der Root-Zone von `ftp://ftp.internic.net/domain/root.zone.gz` downloaden.

Beispiel:

```
zone "." IN {
    type hint;
    file "slave/root";
};
```

Listing 6.30: Beispiel einer `hint`-Zone

Datenbankmodule

Im Verzeichnis `contrib/sdb` der BIND9.3-Distribution befinden sich folgende Datenbankmodule:

- `bdb`: liest Zonefiles aus Berkeley4-Datenbanken.
- `dirdb`: übermittelt die Dateinamen eines Dateisystems als DNS-Records.
- `ldapdb`: fragt Records bei einem LDAP-Server ab.
- `pgsql`: liest die Records aus einer PostgreSQL-Datenbank.
- `tcl`: ruft ein tcl-Script auf, das ein Zonefile ausgibt.
- `time`: gibt die aktuelle Uhrzeit als DNS-Record aus.

Alle Module sind in der Programmiersprache C geschrieben. Wenn Sie eines davon nutzen möchten, müssen Sie dies im BIND-Sourcecode angeben, BIND neu kompilieren und eine entsprechende `zone`-Anweisung angeben:

```
zone "venaas.com" {
        type master;
        database "ldap ldap://129.241.20.67/dc=venaas,dc=com,\
o=DNS,dc=venaas,dc=no 172800";
};
```

Listing 6.31: Beispiel zur LDAP-Anbindung

6.4 Neuerungen von BIND 9.3.0

Am 23. September 2004 erschien die BIND-Version 9.3.0, die viele Neuerungen gegenüber der Version 9.2.3 enthält, die den meisten Linux-Distributionen beiliegt.

Dieser Abschnitt zeigt die Unterschiede.

6.4.1 logging

Die `logging`-Anweisung kennt zwei neue Kategorien:

Kategorie	Thema
`update-security`	Freigegebene und abgelehnte Update-Requests
`delegation-only`	Die Records, die aufgrund der gesetzten `delegation-only`-Option nicht gefunden wurden.

Tabelle 6.23: Neue `logging`-Kategorien in BIND 9.3.0

6.4.2 masters

Mit der neuen `masters`-Anweisung geben Sie einer Liste von primären Nameservern einen Namen. Diesen Namen können Sie bei der Definitionen sekundärer Zonen einsetzen.

Die `masters`-Anweisung hat folgende Syntax:

```
masters Name [port Portnummer] {
  ( MasterListe |
    ip_addr [port Portnummer] [key Schlüsselname]
  );
  [...]
};
```

Listing 6.32: Syntax der `masters`-Anweisung

6.4.3 options

Diese Optionen sind seit BIND 9.2.3 neu hinzugekommen, entfernt oder erweitert worden:

Option	Beschreibung
`allow-v6-synthesis`	Diese Option wurde entfernt, da Bitstring-Namen und A6-Records seit RFC3363 nicht mehr eingesetzt werden.

`alt-transfer-source[-v6]`
`  (`*`IP-Adresse`*`|*)`
`  [port` *`Portnummer`*`]`
`;`
`use-alt-transfer-source`
`  (yes|no);`
	gibt eine alternative Netzschnittstelle und Portnummer an. BIND verwendet sie, wenn die mit `transfer-source-[v6]` angegebene schon besetzt ist und die Option `use-alt-transfer-source` auf `yes` steht.
	Standardwert: `*`

`avoid-(4|6)-udp-ports{`
`  Portnummer;`
`  [...;]`
`};`
	bestimmt, welche UDP-Portnummern BIND *nicht* als zufälligen Source-Port verwenden darf.

`check-names`
`  ( master |`
`  slave |`
`  response)`
`  ( fail |`
`  warn |`
`  ignore);`
	bestimmt das Verhalten bei Records, die ungültige Zeichen enthalten.
	Standardwerte:
	`check-names master fail;`
	`check-names slave warn;`
	`check-names response ignore;`

`disable-algorithms`
`  `*`Domain`*` {`
`  `*`Algorithmus`*`;`
`  [`*`Algorithmus`*`;...]`
`};`
	deaktiviert die genannten DNSSEC-Algorithmen für alle Zonen, die in der genannten Domain liegen.

`dnssec-enable` `  (yes|no);`	bestimmt, ob BIND DNSSEC anwendet. **Standardwert:** `no`

Tabelle 6.24: Neue Optionen in BIND 9.3.0

Option	Beschreibung
`dnssec-lookaside` *Domain1* *Anker* *Domain2*;	weist DNSSEC an, die Authentifizierung an eine andere Domain weiterzuleiten.
`dnssec-must-be-secure` *Domain* (yes|no);	ignoriert alle Antworten der genannten Domain, die nicht mit DNSSEC bestätigt sind.
`dual-stack-servers` [port *Portnummer*] { (*Hostname*|*IP-Adresse*) [port *Portnummer*]; ... };	gibt eine Reihe von Nameservern an, die sowohl über IPv4 als auch über IPv6 an das Internet angeschlossen sind. Wenn Ihr Nameserver nur eines der beiden Protokolle beherrscht, ermöglicht dieser Parameter den Zugriff auf die Nameserver des anderen Protokolls.
`edns-udp-size` *Bytes*;	bestimmt die maximale Größe der von BIND erstellten EDNS-UDP-Pakete. Setzen Sie die Größe auf 512 Byte herab, wenn Ihre Firewall ein Problem mit größeren UDP-Paketen hat. Mindestwert: `512` Höchstwert: `4096` **Standardwert:** `4096`
`flush-zones-on-shutdown` (yes|no);	bewirkt, dass BIND die im Speicher geänderten Zonen beim Herunterfahren in die entsprechenden Zonefiles niederschreibt. **Standardwert:** `no`
`hostname` ("*Text*"|none);	bestimmt den Text, den BIND auf die Frage nach dem TXT-Record des Eintrags `hostname.bind` in der Klasse CH sendet. **Standardwert:** der lokale Hostname

Tabelle 6.24: Neue Optionen in BIND 9.3.0 (Fortsetzung)

Option	Beschreibung
! `ixfr-from-differences (yes|no);`	bewirkt, dass BIND die Änderungen, die er an Zonefiles feststellt, im Journal der Zone speichert. So können die sekundären Nameserver die Änderungen auch bei Zonen abfragen, die Sie nicht über Dynamic Updates pflegen. Diese Option können Sie auch in `zone`-Anweisungen für jede Zone individuell setzen. **Standardwert:** `no`
`key-directory "`*`Verzeichnis`*`";`	gibt das Verzeichnis an, in dem BIND bei Dynamic Updates die privaten und öffentlichen Schlüssel sucht. **Achtung:** Das Verzeichnis müssen Sie *absolut* angeben. Diese Option können Sie auch in `zone`-Anweisungen für jede Zone individuell setzen. **Standardwert:** der Wert der `directory`-Option
`max-journal-size (`*`Dateigröße`*` [(K|M|G)]| unlimited | default );`	ermöglicht Ihnen, die Größe der Journal-Dateien zu begrenzen. Eine Journal-Datei enthält die letzten Änderungen einer Zone und ist Voraussetzung für inkrementelle Zonetransfers. Diese Option können Sie auch in `zone`-Anweisungen für jede Zone individuell setzen. **Standardwert:** `unlimited`
`memstatistics-file "Dateiname";`	gibt die Datei an, in der BIND beim Herunterfahren seine Statistiken über die Speichernutzung hinterlässt. **Standardwert:** `named.memstats`
`multi-master (yes|no);`	gibt an, ob BIND Sie warnt, wenn die Versionsnummer einer Zone auf dem primären Nameserver geringer ist als die lokale. `no`: Warnung `yes`: keine Warnung

Tabelle 6.24: Neue Optionen in BIND 9.3.0 (Fortsetzung)

Option	Beschreibung
	Diese Option können Sie auch in `zone`-Anweisungen für jede Zone individuell setzen. **Standardwert:** `no`
`pid-file none;`	der neue Wert `none` verhindert, dass BIND ein PID-file anlegt. **Standardwert:** `/var/run/named.pid;`
`preferred-glue (A|AAAA|NONE);`	bestimmt die Reihenfolge, in der BIND Glue-Records sendet, wenn es zu einer Zone sowohl IPv6- als auch IPv4-Nameserver gibt. **Standardwert:** `NONE` (d.h. zufällige Reihenfolge)
`querylog (yes|no);`	bestimmt, ob BIND Abfragen protokollieren soll. Wenn Sie diese Option *nicht* angeben, protokolliert BIND Abfragen nur dann, wenn Ihre `logging`-Anweisung die `queries`-Kategorie konfiguriert.
`recursing-file "`*`Dateiname`*`";`	gibt den Pfad- und Dateinamen der Datei an, an die BIND die Liste aller aktuell bearbeiteten rekursiven Abfragen anhängt, wenn man `rndc recursing` aufruft. Die Liste gibt Ihnen Aufschluss über die aktuelle Belastung Ihres Nameservers. **Standardwert:** `./named.recursing`
`root-delegation-only [ exclude { "`*`Domain`*`"; ["`*`Domain`*`";...] ] };`	legt fest, dass die Root-Zone und die Top-Level-Domains nur Delegationen enthalten dürfen. Da dies z.B. für die `de`-Domain nicht zutrifft, können Sie die TLDs angeben, die weitere Records enthalten dürfen. Diese Option ist vermutlich nur für Betreiber von Root- oder TLD-Nameservern interessant, die ihre Zonen frei von falschen Glue-Records halten möchten.

Tabelle 6.24: Neue Optionen in BIND 9.3.0 (Fortsetzung)

Option	Beschreibung

`rrset-order {`
`  [class Klasse]`
`  [type Typ]`
`  [name Name]`
`  order`
`  (random|cyclic);`
`  ...;`
`};`
	ermöglicht eine noch umfangreichere Konfiguration der Record-Reihenfolge, als es mit der `sortlist`-Option möglich ist. Zum Beispiel ermöglicht der `cyclic`-Parameter eine Lastverteilung.
	Standardwert: `class=IN`, `type=ANY`, `name=*`, `order` unbestimmt.

`server-id`
`  ("`*`Text`*`"|`
`  none|`
`  hostname);`
	bestimmt den Text, den BIND auf die Frage nach dem TXT-Record des Eintrags `id.server` in der Klasse `CH` sendet.
	Standardwert: `none`

`tcp-listen-queue` `  `*`Zahl`*`;`	gibt die maximale Größe der Warteschlange an, in die der Kernel TCP-Pakete an BIND einreiht. Wenn die Warteschlange voll ist, weist der Kernel Pakete zurück. Mindestwert: `3` **Standardwert:** `3`
`version` `  ("`*`Text`*`"|none);`	bestimmt den Text, den BIND auf die Frage nach dem TXT-Record des Eintrags `version.bind` in der Klasse CH sendet. Diese Option gibt es schon seit BIND Version 8. Neu hinzugekommen ist das Schlüsselwort `none`, das die Frage unbeantwortet lässt.

Tabelle 6.24: Neue Optionen in BIND 9.3.0 (Fortsetzung)

Die `allow-v6-synthesis` entfällt, da BIND ab der Version 9.3 Bitstring-Namen und A6-Records nicht mehr beherrscht.

6.4.4 server

Sie können Ihre eigene Quelladresse mit BIND 9.3 für jeden remote Nameserver separat einstellen:

Option	Beschreibung

`transfer-source[-v6]`
`(`*`IP-Adresse`*`|*)`
`[port (`*`Portnummer`*`|*)] ;`
	siehe `options`-Anweisung in Abschnitt 6.3.7.

Tabelle 6.25: Neue Server-Optionen in BIND 9.3.0

6.4.5 view

BIND unterstützt ab Version 9.3 TSIG-Schlüssel bei den Bedingungen `match-clients` und `match-destinations`.

Beispiel:

```
key bekannt {
    algorithm  hmac-md5;
    secret     "La/E5CjG9O+os1jq0a2jdA==";
};

view "authentifiziert" {
    match-clients { key bekannt; };
    [ Optionen und Anweisungen... ]
}
```

Listing 6.33: TSIG-Schlüssel als `view`-Bedingung

6.4.6 zone

BIND 9.3 kennt einen neuen Zonen-Typ und fünf neue Zonen-Optionen:

Option	Beschreibung

`check-names`
`( fail |`
`warn |`
`ignore);`
	bestimmt das Verhalten bei Records, die ungültige Zeichen enthalten.
	Standardwert: `fail` (bei `type master`), `warn` (sonst).

`delegation-only` `(yes|no);`	`delegation-only yes;` entspricht `type delegation-only;` (s.u.). **Standardwert:** `no`

Tabelle 6.26: Neue `zone`-Optionen in BIND 9.3.0

Option	Beschreibung

`ixfr-from-differences;`
`max-journal-size;`
`multi-master;`
	Diese neuen Zone-Optionen können auch in der globalen `options`-Anweisung stehen und werden daher in Abschnitt 6.4.1 beschrieben.

`type delegation-only;`	bewirkt, dass BIND für diese Zone die Existenz aller Records abstreitet, die keine Delegation darstellen. Zur Delegation zählen die delegierenden NS-Records und (falls die A-Records der Nameserver selbst in der Domain liegen) die Glue-Records. Dieser Typ ist vermutlich nur für Betreiber von Root- oder TLD-Nameservern interessant, die ihre Zonen frei von falschen Glue-Records halten möchten. Glue-Records werden in Abschnitt 1.7 beschrieben.

Tabelle 6.26: Neue `zone`-Optionen in BIND 9.3.0 (Fortsetzung)

7 Zonefiles

Ein Zonefile ist eine Textdatei, aus der ein Nameserver die Records einer Zone liest.

Seit Version 9.1 kann BIND die Records alternativ aus anderen Quellen (z.B. relationalen Datenbanken und LDAP-Servern) lesen.

7.1 Einlesen der Zonefiles

Änderungen, die Sie an dem Zonefile vornehmen, werden nicht sofort wirksam, sondern erst beim nächsten Neustart des `named`-Prozesses oder bei Aufruf von `rndc reload`.

Achtung: Prüfen Sie vor dem Einlesen eines Zonefiles die Syntax mit dem Programm `named-checkzone`, das der BIND-Distribution beiliegt.

Mehr zum Thema: Der Neustart des Nameservers und der Aufruf von `rndc` werden in Abschnitt 11.2 beschrieben. `named-checkzone` wird in Abschnitt 13.5.2 beschrieben.

7.2 Syntax

Das Format von Zonefiles ist in RFC1035 beschrieben.

Ein Zonefile kann

- Records,
- Anweisungen (`$INCLUDE`, `$TTL`, `$ORIGIN`, `$GENERATE`),
- Kommentare (zwischen ; und dem Zeilenende) und
- Leerzeilen

enthalten.

Jeder Record besteht aus mehreren, teilweise optionalen Feldern:

- Host- oder Domainname
- TTL und Klasse (optional, in beliebiger Reihenfolge)
- Typ (siehe Abschnitt 7.5).
- Daten (typspezifisch)
- Kommentar (optional, zwischen ; und dem Zeilenende)

Schreiben Sie diese Felder (mit Leerzeichen oder Tabulatorzeichen getrennt) hintereinander in eine Zeile und schließen Sie mit einem Zeilenumbruch ab. Wenn Sie einen langen Record auf mehrere Zeilen verteilen möchten, müssen Sie in der Zeile mit dem ersten Zeilenumbruch eine öffnende und in der Zeile nach dem letzten Zeilenumbruch eine schließende, runde Klammer unterbringen.

Achtung: Die Klammern müssen außerhalb des Kommentars stehen.

Beispiel für ein Zonefile:

```
; Authority-Informationen:
ibm.com.     600 IN SOA   ( ; ist auf mehrere Zeilen verteilt
             ns.watson.ibm.com.   ; primärer NS
             nrt.watson.ibm.com.  ; email
             2004072700           ; Serial#
             3600 1800 604800     ; Refresh Retry Expire
             600)                 ; negative caching
;
ibm.com.     600 IN NS    ns.watson.ibm.com.
ibm.com.     600 IN NS    ns.austin.ibm.com.

; Mailserver:
ibm.com.     600 IN MX 10 e1.ny.us.ibm.com.
ibm.com.     600 IN MX 10 e2.ny.us.ibm.com.
ibm.com.     600 IN MX 10 e3.ny.us.ibm.com.

; Webserver:
```

```
ibm.com.     600 IN A  129.42.19.99  ; für http://ibm.com/
www.ibm.com. 600 IN A  129.42.16.99  ; für http://www.ibm.com/
www.ibm.com. 600 IN A  129.42.17.99
www.ibm.com. 600 IN A  129.42.18.99

; Die Delegationen der Subdomains watson, austin und us
; wurden zur besseren Übersichtlichkeit in
; diesem Beispiel ausgelassen.
```

Listing 7.1: Ausschnitt aus dem Zonefile von ibm.com.

7.3 Abkürzende Notation

In diesem Abschnitt erfahren Sie, wie Sie Ihre Zonefiles übersichtlicher gestalten können, indem Sie einzelne Felder oder ganze Records weglassen oder ersetzen.

Bis auf die `$GENERATE`-Anweisung und die Zeiteinheiten, die nur mit BIND möglich sind, werden alle Abkürzungen in RFC1035 beschrieben und sind damit fester Bestandteil des DNS.

7.3.1 Weglassen von Angaben

Weglassen des Namens

Bei allen Einträgen, die mit einem Leerzeichen oder Tabulatorzeichen beginnen, setzt BIND den Namen des vorangegangenen Records ein. Damit können Sie das Zonefile aus dem vorangegangenen Listing 7.1 kürzer schreiben als:

```
ibm.com.     600 IN SOA (
                        ns.watson.ibm.com.
                        nrt.watson.ibm.com.
                        2004072700
                        3600 1800 604800
                        600)
             600 IN NS  ns.watson.ibm.com.
             600 IN NS  ns.austin.ibm.com.
             600 IN MX  10 e1.ny.us.ibm.com.
             600 IN MX  10 e2.ny.us.ibm.com.
```

```
            600 IN MX    10 e3.ny.us.ibm.com.
            600 IN A     129.42.19.99
www.ibm.com. 600 IN A    129.42.16.99
            600 IN A     129.42.17.99
            600 IN A     129.42.18.99
```

Listing 7.2: Listing 7.1 nach Weglassen von Namen und Kommentaren

Weglassen der Domain

Wenn Sie Namen *relativ* (also ohne abschließenden Punkt) angeben, hängt BIND automatisch den Namen der Zone an. Der Name der Zone stammt aus der `zone{}`-Anweisung in der Konfigurationsdatei `named.conf`.

Beispiel:

Im Zonefile der Zone `ibm.com.` können Sie das vorangegangene Listing 7.2 kürzer schreiben als:

```
ibm.com.     600 IN SOA   ns.watson nrt.watson (
                          2004072700
                          3600 1800 604800
                          600)
             600 IN NS    ns.watson
             600 IN NS    ns.austin
             600 IN MX    10 e1.ny.us
             600 IN MX    10 e2.ny.us
             600 IN MX    10 e3.ny.us
             600 IN A     129.42.19.99
www          600 IN A     129.42.16.99
             600 IN A     129.42.17.99
             600 IN A     129.42.18.99
```

Listing 7.3: Listing 7.2 nach Weglassen der Domain

$ORIGIN-Anweisung

Mit der `$ORIGIN`-Anweisung können Sie den Domainnamen (relativ zur aktuellen Zone oder absolut) festlegen, den BIND an alle folgenden, relativen Namen anhängt.

In der Literatur verwendet man die `$ORIGIN`-Anweisung oft, um dem Leser zu zeigen, aus welcher Zone ein Beispiel stammt.

Beispiel:

```
$ORIGIN ibm.com.
; ...
            600 IN MX    10 e1.ny.us
; ...
$ORIGIN ny.us
e1          600 IN A     1.2.3.4
; ...
```

Listing 7.4: Beispiel zur `$ORIGIN`-Anweisung

Weglassen der Klasse

Wenn Sie keine Klasse angeben, verwendet BIND die Klasse

- des vorangegangenen Records,
- der `zone`-Anweisung in der Konfigurationsdatei,
- der umgebenden `view`-Anweisung in der Konfigurationsdatei (falls Sie mit Views arbeiten) oder
- `IN`.

Der erste Treffer zählt.

Achtung: Es ist eine gute Idee, die Klasse beim SOA-Record *nicht* wegzulassen. So können Sie Ihr Zonefile auch auf Nameservern einspielen, die keine oder eine andere Standardklasse haben.

Beispiel:

```
$ORIGIN ibm.com.
ibm.com.    600 IN  SOA   ns.watson nrt.watson (
                          2004072700
                          3600 1800 604800
                          600)
            600     NS    ns.watson
```

```
            600     NS    ns.austin
            600     MX    10 e1.ny.us
            600     MX    10 e2.ny.us
            600     MX    10 e3.ny.us
            600     A     129.42.19.99
www         600     A     129.42.16.99
            600     A     129.42.17.99
            600     A     129.42.18.99
```

Listing 7.5: Listing 7.3 nach Weglassen der Klasse

Mehr zum Thema: Abschnitt 7.4 beschreibt die möglichen Klassen.

$TTL-Anweisung und Weglassen der TTL

Wenn Sie keine TTL angeben, verwendet BIND

- die Angabe aus der letzten `$TTL`-Anweisung oder
- die TTL des letzten Records mit TTL-Angabe.

Der erste Treffer zählt. Wenn BIND die TTL auf diesem Weg nicht ermitteln kann, gibt er eine Warnung aus und verwendet

- das Negative-Caching-Timeout (das ist der letzte Wert) des SOA-Records.

Beispiel:

```
$ORIGIN ibm.com.
$TTL 600
ibm.com. IN SOA ns.watson nrt.watson (
                2004072700
                3600 1800 604800
                600)
           NS  ns.watson
           NS  ns.austin
           MX  10 e1.ny.us
           MX  10 e2.ny.us
           MX  10 e3.ny.us
           A   129.42.19.99
```

```
www          A   129.42.16.99
             A   129.42.17.99
             A   129.42.18.99
```

Listing 7.6: Listing 7.5 nach Weglassen der TTL

7.3.2 Zeiteinheiten

Zonefiles enthalten verschiedene Timeouts:

- das TTL-Feld jedes Records
- die `$TTL`-Anweisung
- die Refresh-, Retry-, Expire- und Negative-Caching-Felder des SOA-Records

RFC1035 legt fest, dass man diese Werte in Sekunden angibt. BIND versteht jedoch zusätzlich folgende Zeiteinheiten:

Dauer	Sekunden [RFC1035]	Mit Einheit [BIND]
Eine Sekunde	`1`	`1s`
Eine Minute	`60`	`1m`
Eine Stunde	`3600`	`1h`
Ein Tag	`86400`	`1d`
Eine Woche	`604800`	`1w`

Tabelle 7.1: Zeitangaben mit und ohne Einheiten

Wenn Sie verschiedene Einheiten addieren möchten, hängen Sie diese einfach zusammen: z.B. `1h30m` für »Anderthalb Stunden«.

Damit können Sie das vorangegangene Listing 7.6 übersichtlicher schreiben als:

```
$ORIGIN ibm.com.
$TTL 10m
ibm.com. IN SOA ns.watson nrt.watson (
                2004072700
                1h 30m 1w
```

```
                10m)
            NS  ns.watson
            NS  ns.austin
            MX  10 e1.ny.us
            MX  10 e2.ny.us
            MX  10 e3.ny.us
            A   129.42.19.99
www         A   129.42.16.99
            A   129.42.17.99
            A   129.42.18.99
```

Listing 7.7: Listing 7.6 mit Zeiteinheiten

7.3.3 @

BIND ersetzt das @-Zeichen mit

- dem Wert der letzten $ORIGIN-Anweisung oder
- dem Namen der Zone (aus der zone-Anweisung in der Konfigurationsdatei).

Der erste Treffer zählt.

Dies ermöglicht Ihnen, dasselbe Zonefile für mehrere Zonen einzusetzen (z.B. ibm.com und ibm.de).

Beispiel:

```
$ORIGIN ibm.com.
$TTL 10m
@ IN SOA ns.watson nrt.watson (
             2004072700
             1h 30m 1w
             10m)
     NS  ns.watson
     NS  ns.austin
     MX  10 e1.ny.us
     MX  10 e2.ny.us
```

```
     MX  10 e3.ny.us
     A   129.42.19.99
www  A   129.42.16.99
     A   129.42.17.99
     A   129.42.18.99
```

Listing 7.8: Listing 7.7 nach Einsatz von @

7.3.4 $GENERATE

Mit der `$GENERATE`-Anweisung können Sie mehrere Records erstellen, die sich nur in einem Zähler unterscheiden.
Der Record muss vom Typ NS, A, AAAA, CNAME, DNAME oder PTR sein.

> **Achtung:** Mit der `$GENERATE`-Anweisung können Sie *keine* MX-Records erstellen.

Die `$GENERATE`-Anweisung hat folgende Syntax:

```
$GENERATE Start-Ende[/Schrittweite] Record
```

Listing 7.9: Syntax der $GENERATE-Anweisung

Mit *Start*, *Ende* und *Schrittweite* geben Sie den Bereich an, den der Zähler durchläuft.

BIND setzt den Zähler anstelle aller Dollarzeichen im Namens- und Datenfeld ein.

> **Achtung:** Schreiben Sie `$$` oder `\$`, wenn Ihr Record ein *echtes* Dollarzeichen enthalten und die `$GENERATE`-Anweisung dieses nicht ersetzen soll.

Wenn Sie statt eines simplen Dollarzeichens den Ausdruck `${`*Offset*`,`*Stellen*`,`*Basis*`}` angeben, setzt BIND den Zähler um eine Zahl erhöht oder erniedrigt, mit führenden Nullen oder in einem anderen Zahlensystem ein:

Parameter	Beschreibung
Offset	Eine ganze Zahl, die BIND vor jeder Ausgabe zu dem Zähler addiert. Stellen Sie ihr ein Minuszeichen voran, um sie vom Zähler zu subtrahieren. **Standardwert:** `0`
Stellen	Die Mindestlänge des einzufügenden Wertes. BIND füllt die Darstellung des Zählers mit führenden Nullen auf, bis sie die Mindestlänge erreicht. **Standardwert:** `0`
Basis	Das Zahlensystem, in dem BIND den Zähler ausgibt. `d`: Dezimalsystem (Ziffern 0–9) `o`: Oktalsystem (Ziffern 0–7) `x`: Hexadezimalsystem (Ziffern 0–9, a–f) `X`: Hexadezimalsystem (Ziffern 0–9, A–F) **Standardwert:** `d`

Tabelle 7.2: Zähler-Formatierung mit ${...}

Beispiel:

```
$ORIGIN ibm.com.
$TTL 10m
@ IN SOA ns.watson nrt.watson (
             2004072700
             1h 30m 1w
             10m)
     NS  ns.watson
     NS  ns.austin
     MX  10 e1.ny.us
     MX  10 e2.ny.us
     MX  10 e3.ny.us
     A  129.42.19.99
$GENERATE 16-18   www   A     129.42.$.99
```

Listing 7.10: Listing 7.8 nach Einbau von $GENERATE

Die `$GENERATE`-Anweisung in diesem Beispiel erzeugt folgende Records:

```
www  A   129.42.16.99
     A   129.42.17.99
     A   129.42.18.99
```

Listing 7.11: Aus der $GENERATE-Anweisung entstehende Records

Mehr zum Thema: Die $GENERATE-Anweisung wird oft in Zusammenhang mit der Classless in-addr.arpa delegation eingesetzt, die in Abschnitt 9.1 beschrieben wird.

7.3.5 Wildcard-Records (*)

Wenn Sie an Stelle des Record-Namens einen * angeben, sendet BIND die Daten dieses Records auf alle Abfragen, die folgende Bedingungen erfüllen:

1. Die Zone enthält keinen Record (beliebigen Typs) mit dem gefragten Namen,
2. der gefragte Typ stimmt überein und
3. der gefragte Name liegt in der Zone, das heißt, er enthält keine Subdomainangabe, die in eine andere Zone delegiert ist.

Beispiel:

```
$ORIGIN ibm.com.
$TTL 10m
; [...]
@       MX    10 e1.ny.us
        MX    10 e2.ny.us
*       MX    10 e1.ny.us       ; WILDCARD-RECORD
        MX    10 e2.ny.us       ; WILDCARD-RECORD
www     A     129.42.16.99
        A     129.42.17.99
        MX    10 e1.ny.us       ; ist erforderlich...
        MX    10 e2.ny.us
us      NS    ns.watson         ; Delegation einer Subdomain
```

Listing 7.12: Beispiel zu Wildcard-Records

Die beiden Wildcard-Records in diesem Beispiel bewirken, dass Mails an beliebige Hostnamen (z.B. jfranken@das.gibts.doch.gar.nicht.ibm.com) ihren Weg zu den offiziellen Mailservern finden.

Die anderen MX-Records (für @ und www) sind erforderlich, weil die Bedingung 1 bei diesen Namen nicht erfüllt ist.

Achtung: Diese Wildcard-Records wirken *nicht* bei Mails an jfranken@barf.us.ibm.com, weil us in einer anderen Zone liegt und damit Bedingung 3 nicht erfüllt ist.

Mehr zum Thema: Siehe RFC1034 und die Datei draft-ietf-dnsext-wcard-clarify-02.txt, die dem BIND-9.3.0-Source beiliegt. Die Nachteile von Wildcard-MX-Records werden auf http://www.sendmail.org/faq/Section4.html#4.1 beschrieben.

7.3.6 $INCLUDE

Mit der $INCLUDE-Anweisung können Sie den Inhalt einer anderen Datei in das Zonefile einfügen.

Die $INCLUDE-Anweisung hat folgende Syntax:

```
$INCLUDE Dateiname [Origin]   [; Kommentar]
```

Listing 7.13: Syntax der $INCLUDE-Anweisung

Wenn Sie hinter dem Dateinamen einen Domainnamen angeben, wird BIND diesen an alle relativen Records der Datei anhängen.

Achtung: Wenn die eingefügte Datei $ORIGIN-Anweisungen enthält, setzt BIND den $ORIGIN nach dem Einfügen wieder auf den Wert zurück, der vor dem Einfügen bestand.

7.4 Record-Klassen

BIND kennt folgende Klassen:

Kürzel	Beschreibung
IN	Alle Netze, die das Internetprotokoll (IP) verwenden
CH oder CHAOS	Das CHAOSnet, ein am MIT eingesetztes LAN-Protokoll aus den 1970ern oder die Klasse, unter der BIND seine Versionsnummer, den Hostnamen und die System-ID anbietet
HS	Hesiod, ein Verzeichnisdienst (ähnlich NIS) aus dem Athena-Projekt des MIT

Tabelle 7.3: Record-Klassen

7.5 Record-Typen

7.5.1 Übersicht

Typ-Nummern und -Kürzel

Jeder Record-Typ hat eine eindeutige Nummer (zwischen `0` und `65535`), die im DNS-Protokoll zum Einsatz kommt. Zum Beispiel steht `12` für den »Pointer«-Record.

Im Zonefile können Sie die meisten Typen erfreulicherweise über ein Kürzel (z.B. `PTR`) angeben.

BIND erlaubt Ihnen seit der Version 9.1, den Typ *anonym*, also über seine Nummer, anzugeben (z.B. `TYPE12`). Diese Erweiterung des DNS ist in RFC3597 beschrieben und soll die Einführung neuer Record-Typen im Internet vereinfachen.

Wichtige Record-Typen

Die wichtigsten Record-Typen zur Beschreibung eines Hosts lauten:

Typ	anonym	Beschreibung	siehe Abschnitt
A	TYPE1	speichert eine IPv4-Adresse eines Hosts.	7.5.2
AAAA	TYPE28	speichert eine IPv6-Adresse eines Hosts.	7.5.4
CNAME	TYPE5	bezeichnet einen weiteren Namen (Alias), unter dem ein bereits bekannter Host *zusätzlich* erreichbar ist.	7.5.8
PTR	TYPE12	verweist ebenfalls auf einen A-Record, ist jedoch für Reverse queries (Fragen nach dem Host- oder Netznamen zu einer IP-Adresse) vorgesehen.	7.5.32

Tabelle 7.4: Wichtige Record-Typen für Hosts

Die wichtigsten Record-Typen zur Beschreibung einer Domain lauten:

Typ	anonym	Beschreibung	siehe Abschnitt
MX	TYPE15	nennt einen Mailserver der Domain.	7.5.25
NS	TYPE2	weist der angegebenen Zone einen Nameserver zu. Dabei handelt es sich entweder um einen Authority- oder einen Delegations-Eintrag.	7.5.27
SOA	TYPE6	nennt den primären Nameserver, die E-Mail-Adresse des Hostmasters und verschiedene Timeouts für die Kommunikation mit den sekundären Nameservern.	7.5.38
SRV	TYPE33	»Service«: gibt an, welche Server in dieser Domain für das angegebene Protokoll zuständig sind (für LDAP erforderlich, z.B. bei Microsoft ADS und GNU/Linux PAM).	7.5.39

Tabelle 7.5: Wichtige Record-Typen für Domains

Security-bezogene Record-Typen

Wenn Sie mit DNSSEC arbeiten, benötigen Sie folgende Typen:

Typ	anonym	Beschreibung	siehe Abschnitt
DNSKEY	TYPE48	speichert den öffentlichen Schlüssel eines Eintrags oder einer Zone.	7.5.11
DLV	TYPE65323	»DNSSEC Lookaside Validation«: speichert den Hashcode des DNSKEYs einer anderen Domain.	7.5.9
DS	TYPE43	»Delegation Signer«: speichert den Hashcode des DNSKEYs einer delegierten Subdomain.	7.5.12
NSEC	TYPE47	»Next Secure«: gibt den Namen des nächsten und die Typen des aktuellen RRset an. Daraus kann man ableiten, welche Records in einer Zone *nicht* existieren.	7.5.30
RRSIG	TYPE46	»Resource Record Signature«: enthält die Signatur für einen anderen Record.	7.5.35

Tabelle 7.6: Record-Typen für DNSSEC (ab BIND 9.3)

Wenn Sie SIG(0) oder TKEY einsetzen, sind folgende Typen für Sie wichtig:

Typ	anonym	Beschreibung	siehe Abschnitt
KEY	TYPE25	speichert den öffentlichen Schlüssel eines Eintrags.	7.5.16
SIG	TYPE24	enthält die Signaturen für andere Records.	7.5.37

Tabelle 7.7: Record-Typen für SIG(0) und TKEY

Mehr zum Thema: Beschreibungen zu TKEY und SIG(0) finden Sie in RFC2930, RFC2931 und Abschnitt 12.2.

Sonstige Record-Typen

Selten findet man folgende Record-Typen:

Typ	anonym	Beschreibung	siehe Abschnitt
CERT	TYPE37	»Certificate«: speichert verschiedene digitale Zertifikate (z.B. X509-Zertifikate oder signierte, öffentliche PGP-Schlüssel).	7.5.7
DNAME	TYPE39	weist den Resolver an, die Abfrage unter einem anderen Domainnamen zu wiederholen.	7.5.9
HINFO	TYPE13	»Host Information«: speichert technische Details (z.B. Prozessor und Betriebssystem) eines Hosts.	7.5.14
NAPTR	TYPE35	»Name authority Pointer«: speichert eine Regel zur Verarbeitung von URIs.	7.5.26
NSAP	TYPE22	»Network service access point«: speichert NSAP-Adressen.	7.5.28
PX	TYPE26	»Pointer to X400/RFC822-Mapping-Information«: definiert Gateways und Adressumsetzung für die Kommunikation mit X400-Mailsystemen.	7.5.33
TXT	TYPE16	speichert beliebige Texte und kann vielfältig eingesetzt werden.	7.5.41

Tabelle 7.8: Seltene Record-Typen

Und schließlich beherrscht BIND noch einige experimentelle, historische und obsolete Typen. Sie können diese Typen für Ihren eigenen Anwendungen in Ihrem Intranet einsetzen. Aber bitte verlassen Sie sich nicht darauf, dass diese auch im Internet funktionieren:

Typ	anonym	Beschreibung	siehe Abschnitt
A6	TYPE38	Teil einer IPv6-Adresse im Bitstring-Format	7.5.3

Tabelle 7.9: Record-Typen, die Sie nicht verwenden sollten

Typ	anonym	Beschreibung	siehe Abschnitt
AFSDB	TYPE18	»Andrew File System Data Base«: speichert den Namen eines AFS-Datenbankservers, der für diese Domain zuständig ist.	7.5.5
APL	TYPE42	»Address Prefix List«: speichert IP-Adressbereiche.	7.5.6
GPOS	TYPE27	»Global Position«: speichert geografische Positionsangaben. Wurde von LOC abgelöst.	7.5.13
ISDN	TYPE20	speichert ISDN-Adressen.	7.5.15
KX	TYPE36	»Key exchanger«: nennt die Key-Exchange-Systeme, die für den angegebenen Rechner/Domain zuständig sind (z.B. bei IPsec).	7.5.17
LOC	TYPE29	»Location«: speichert geografische Positionsangaben.	7.5.18
MB	TYPE7	»Mailbox«: nennt den Mailserver zu dem angegebenen Usernamen.	7.5.19
MD	TYPE3	»Mail destination«: nennt den primären Nameserver einer Domain.	7.5.20
MF	TYPE4	»Mail forwarder«: nennt einen weiteren Nameserver einer Domain.	7.5.21
MG	TYPE8	»Mail group«: nennt die Empfänger-E-Mail-Adressen einer Verteilerliste.	7.5.22
MINFO	TYPE14	»Mailbox or mail list information«: nennt die E-Mail-Adressen der Administratoren einer Mailingliste.	7.5.23
MR	TYPE9	»Mailbox Rename«: leitet Mails um.	7.5.24
NSAP-PTR	TYPE23	»NSAP-Pointer«: nennt den Hostnamen zu einer gegebenen NSAP-Adresse.	7.5.29

Tabelle 7.9: Record-Typen, die Sie nicht verwenden sollten (Fortsetzung)

Typ	anonym	Beschreibung	siehe Abschnitt
NXT	TYPE30	»Next«: gibt den Namen des nächsten und die Typen des aktuellen RRset an. Daraus kann man ableiten, welche Records in einer Zone *nicht* existieren. (Wurde durch NSEC ersetzt).	7.5.31
RP	TYPE17	»Responsible Person«: speichert die E-Mail-Adresse des zuständigen Administrators und den Namen eines TXT-Records, der beliebige Infos (z.B. die Raum- oder Rufnummer des Administrators) enthalten kann.	7.5.34
RT	TYPE21	»Route Through«: gibt die Namen von A-, ISDN- oder X25-Records an, unter deren Adressen der angegebene Host besser erreichbar ist.	7.5.36
SSHFP	TYPE44	»Secure shell fingerprint«: speichert SSH-Hostkeys.	7.5.40
UNSPEC	TYPE103	speichert beliebige, binäre Daten.	7.5.42
WKS	TYPE11	»Well Known Service«: gibt die Dienste an, die ein Host anbietet. Wurde durch SRV ersetzt.	7.5.43
X25	TYPE19	speichert X25-Adressen.	7.5.44

Tabelle 7.9: Record-Typen, die Sie nicht verwenden sollten (Fortsetzung)

In BIND nicht implementierte Record-Typen

Die folgenden Record-Typen sind mit BIND *nicht* möglich:

Typ	anonym	Beschreibung
ATMA	TYPE34	speichert eine ATM-Adresse. Dieser Record-Typ ist im Windows-2000-DNS-Server enthalten, wird jedoch in keiner RFC erwähnt.
EID	TYPE31	»Endpoint-Identifier« in der NIMROD-Routing-Architektur (siehe RFC1992 und RFC2103)

Tabelle 7.10: Record-Typen, die Sie mit BIND nicht verwenden können

Typ	anonym	Beschreibung
GID	TYPE102	Group identification
NIMLOC	TYPE32	»NIMROD locator« in der NIMROD-Routing-Architektur (siehe RFC1992)
SINK	TYPE40	»Kitchen Sink«: ermöglicht die Nutzung des DNS als relationale Datenbank. Dieser Record-Typ wurde auf `draft-ietf-dnsind-kitchen-sink-02.txt` vorgestellt, jedoch weder in die RFCs noch in den BIND-Sourcecode aufgenommen.
UID	TYPE101	User identification
UINFO	TYPE100	User information
WINS	?	speichert verschiedene Timeouts und die IP-Adressen eines WINS-Servers für eine Domain. Windows-2000- und Windows-XP-Clients fragen nach diesen Records, wenn sie einen A-Record nicht finden können. Beispiel: `@ IN WINS LOCAL L1 C10 172.16.72.3 172.16.72.4`
WINSR	?	speichert verschiedene Timeouts und die NetBIOS-Domäne, die an den zurückgegebenen NetBIOS-Namen anzuhängen ist. Windows-2000- und Windows-XP-Clients suchen in der `in-addr.arpa.`-Domain nach diesen Records, wenn sie einen PTR-Record nicht finden können. Beispiel: `@ IN WINSR LOCAL L1 C10 example.com.`

Tabelle 7.10: Record-Typen, die Sie mit BIND nicht verwenden können (Fortsetzung)

Achtung: Wenn Sie BIND als sekundären Nameserver für eine Zone einsetzen möchten, die auf einem Microsoft-DNS-Server läuft, werden Sie vielleicht feststellen, dass Zonetransfers wegen der WINS- und WINSR-Einträge fehlschlagen. Eine Lösung besteht darin, in der Microsoft-DNS-Konsole die Replikation der WINS-Einträge abzuschalten.

7.5.2 A

Typnummer: 1

Status: Standard

Beschreibung: Ein A-Record speichert eine IPv4-Adresse eines Hosts.

Wenn ein Host mehrere Netzwerkschnittstellen besitzt oder ein Dienst auf mehreren Servern redundant zur Verfügung steht, können Sie für jede seiner IPv4-Adressen einen eigenen A-Record erstellen und so die Netzlast verteilen sowie die Ausfallsicherheit erhöhen. Nameserver und Resolver können (bei entsprechender Konfiguration) die Reihenfolge der IP-Adressen so sortieren, dass die nächstgelegene Adresse am Anfang der Liste steht und daher von der Applikation zuerst angesprochen wird.

Beispiel: siehe Listing 7.1 in Abschnitt 7.2.

Mehr zum Thema: Siehe RFC1035.

7.5.3 A6

Typnummer: 38

Status: experimentell

Beschreibung: Ein A6-Record speichert einen Teil einer IPv6-Adresse im Bitstring-Format.

Der Reverse-Eintrag für die IPv6-Adresse `4321:0:1:2:3:4:567:89ab` lautet `b.a.9.8.7.6.5.0.4.0.0.0.3.0.0.0.2.0.0.0.1.0.0.0.0.0.0.0.1.2.3.4.ip6.arpa.` .

RFC2874 aus dem Jahr 2000 beschreibt die *Bitstring*-Methode, mit der man diese Adresse unter Verwendung von A6- und DNAME-Records flexibler als `\[x4321000000010002000300040056789ab/128].ip6.arpa.` oder `\[x000300040056789ab/64].\[b10/16].\[x432100000001/48].ip6.arpa.` delegieren kann.

Beispiel:

```
$ORIGIN EXAMPLE.
X               NS NS1.X
                NS NS2.X
```

```
NS1.X           A6 64 ::1:11:111:1111 SUBNET-1.IP6.X
NS2.X           A6 64 ::2:22:222:2222 SUBNET-2.IP6.X
SUBNET-1.IP6.X  A6 48 0:0:0:1::       IP6.X
SUBNET-2.IP6.X  A6 48 0:0:0:2::       IP6.X
IP6.X           A6 48 0::0            SUBSCRIBER-X.IP6.A.NET.
IP6.X           A6 48 0::0            SUBSCRIBER-X.IP6.B.NET.
```

Listing 7.14: A6-Records (aus RFC2874)

Diese Methode konnte sich jedoch nicht durchsetzen und wurde im Jahr 2002 per RFC3363 auf »experimentell« zurückgestuft. Die BIND-Versionen 9.0 bis 9.2.3 beherrschen Bitstring-Labels; zur Version 9.3 hat das ISC den Bitstring-Support aus dem BIND-Code entfernt.

Mehr zum Thema: Siehe RFC2874.

7.5.4 AAAA

Typnummer: 28

Status: Standard

Beschreibung: Ein AAAA-Record speichert eine IPv6-Adresse eines Hosts.

Eine IPv6-Adresse besteht aus acht Zahlen zwischen 0 und 65535 (inkl.), die man meist mit Doppelpunkten getrennt hexadezimal (0 bis FFFF) notiert. Einen Block zusammenhängender Nullen (am Anfang, in der Mitte oder am Ende der Adresse) können Sie »komprimieren«, d.h. durch »::« ersetzen.

Wenn ein Host mehrere Netzwerkschnittstellen besitzt oder ein Dienst auf mehreren Servern redundant zur Verfügung steht, können Sie für jede seiner IPv6-Adressen einen eigenen AAAA-Record erstellen und so die Netzlast verteilen sowie die Ausfallsicherheit erhöhen. Nameserver und Resolver können (bei entsprechender Konfiguration) die Reihenfolge der IP-Adressen so sortieren, dass die nächstgelegene Adresse am Anfang der Liste steht und daher von der Applikation zuerst angesprochen wird.

Beispiel:

```
$TTL 20m
mail  AAAA   FEDC:BA98:7654:3210:FEDC:BA98:7654:3210
www   AAAA   1080:0:0:0:8:800:200C:417A
      AAAA   1080::8:800:200C:417B ; Nullen komprimiert
      AAAA   1080::8:800:200C:417C
```

Listing 7.15: AAAA-Records

Mehr zum Thema: Siehe RFC1886 und zur Syntax von IPv6-Adressen RFC1884.

7.5.5 AFSDB

Typnummer: 18

Status: experimentell

Beschreibung: Ein »Andrew File System Data Base«-Record speichert den Namen eines Servers, der die Andrew-Filesystem-(AFS-)Datenbank dieser »AFS-Zelle« (Domain) führt.

Der AFSDB-Record hat folgende Syntax:

Untertyp Servername

wobei der Untertyp entweder 1 (AFS) oder 2 (Distributed Computing Environment (DCE)) ist.

Beispiel:

```
toaster.com.   AFSDB   1 bigbird.toaster.com.
toaster.com.   AFSDB   1 ernie.toaster.com.
toaster.com.   AFSDB   1 henson.toaster.com.
```

Listing 7.16: AFSDB-Records (aus RFC1183)

Achtung: Wenn der im Datenfeld angegebene Servername nicht mit einem Punkt endet, wird der Nameserver beim Einlesen des Zonefiles den Namen der Zone (`$ORIGIN`) anhängen.

Achtung: Sie dürfen hier keinen CNAME angeben.

Mehr zum Thema: Siehe RFC1183.

7.5.6 APL

Typnummer: 42

Status: experimentell

Beschreibung: Ein »Address Prefix List«: weist einem Netznamen einen oder mehrere IP-Adressbereiche zu.

Jeder Adressbereich hat folgende Syntax:

Protokoll:*Netz*/*Netzmaske*

wobei Sie für *Protokoll* entweder 1 (IPv4) oder 2 (IPv6) einsetzen.

Beispiele:

```
; RFC1101-like announcement of address ranges:
foo.example. IN APL 1:192.168.32.0/21 !1:192.168.38.0/28

; CIDR blocks covered by classless delegation
42.168.192.IN-ADDR.ARPA. IN APL (
      1:192.168.42.0/26
      1:192.168.42.64/26
      1:192.168.42.128/25 )

; Zone transfer restriction
_axfr.sbo.example. IN APL 1:127.0.0.1/32 1:172.16.64.0/22
; Anm.: Unterstrich ist legal, da es sich bei diesem Label
; nicht um einen Hostnamen handelt (siehe Abschnitt 1.4.2).

; List of address ranges for multicast
```

```
multicast.example. IN APL (
   1:224.0.0.0/4
   2:FF00:0:0:0:0:0:0:0/8)
```

Listing 7.17: APL-Records (aus RFC3123)

Mehr zum Thema: Siehe RFC3123.

7.5.7 CERT

Typnummer: 37

Status: Standard

Beschreibung: Ein »Certificate«-Record speichert verschiedene digitale Zertifikate (z.B. X509-Zertifikate oder signierte, öffentliche PGP-Schlüssel).

Syntax:

```
Username [IN] [TTL] CERT Typ  Algorithmus Schlüssel

Typ ::= (
  1   | PKIX   |
  2   | SPKI   |
  3   | PGP    |
  253 | URI    |
  254 | OID    )

Algorithmus ::= (
  1   | RSAMD5      |
  2   | DH          |
  3   | DSA         |
  4   | ECC         |
  252 | INDIRECT    |
  253 | PRIVATEDNS  |
  254 | PRIVATEOID  )
```

Listing 7.18: Syntax von CERT-Records

Geben Sie den Schlüssel im Base64-Format an. Zur besseren Übersichtlichkeit können Sie ihn mit Leerzeichen, Tabulatorzeichen oder Zeilenumbrüchen in beliebig viele Teile zerlegen.

Achtung: Wenn Sie den Schlüssel über mehrere Zeilen verteilen, müssen Sie die Zeilenumbrüche mit runden Klammern schützen.

Mehr zum Thema: Siehe RFC2538.

7.5.8 CNAME

Typnummer: 5

Status: Standard

Beschreibung: Ein canonical name (CNAME) referenziert einen bereits vorhandenen Host, so dass dieser *zusätzlich* auch unter dem Namen dieses Eintrags bekannt ist (Alias).

Beispiel:

```
$ORIGIN informatik.uni-frankfurt.de.
www  38400 IN CNAME sphinx.rbi
smtp 38400 IN CNAME hera.rbi
imap 38400 IN CNAME hera.rbi
time 38400 IN CNAME hera.rbi
ftp  38400 IN CNAME olymp.rbi.uni-frankfurt.de.
```

Listing 7.19: CNAME-Records

In diesem Beispiel erhält der Host `sphinx.rbi.informatik.uni-frankfurt.de.` den Namen `www.informatik.uni-frankfurt.de.` als Alias. Alle Clients (Nameserver und Resolver), die nach der IP-Adresse von `www.informatik.uni-frankfurt.de.` fragen, werden auf `sphinx.rbi.informatik.uni-frankfurt.de.` verwiesen. Die Clients fragen dann den Nameserver der Zone `rbi` nach der IP-Adresse von `sphinx`:

```
$ORIGIN rbi.informatik.uni-frankfurt.de.
sphinx 38400 IN A 141.2.15.17
```

Listing 7.20: Der A-Record, auf den der CNAME verweist

Damit hat der Resolver in Erfahrung gebracht, dass `www.informatik.uni-frankfurt.de.` die IP-Adresse `141.2.15.17` hat.

Achtung: Ein CNAME-Record sollte stets auf einen A- oder PTR-Record verweisen. Obwohl CNAMEs laut RFC1034 auf andere CNAMEs verweisen dürfen, sollte man sich nicht darauf verlassen, dass alle Resolver und Nameserver mit CNAME-Ketten und -Schleifen umgehen können.

Achtung: Versuchen Sie *nicht*, im Datenfeld der Record-Typen AFSDB, MB, MD, MF, MX, NS, NSAP-PTR, PTR, RT, SOA, SRV anstelle des Hostnamens (A- oder AAAA-Record) den Namen eines CNAME-Record anzugeben. Dies wird auf einigen Rechnern nicht zuverlässig funktionieren!

Achtung: Laut RFC1035 muss ein CNAME-Record der einzige Record seines Recordsets sein. Zum Beispiel wäre der Eintrag `@ CNAME www` ungültig, da das Recordset `@` bereits den SOA-Record enthält. Die Regelungen in `draft-ietf-dnsext-dnssec-protocol` erlauben jedoch, dass ein Recordset gleichzeitig NSEC-, RRSIG- und CNAME-Records enthält.

Achtung: Wenn das Verweisziel nicht mit einem Punkt endet, wird der Nameserver beim Einlesen des Zonefiles den Namen der Zone (`$ORIGIN`) anhängen.

Mehr zum Thema: Siehe RFC1035.

7.5.9 DLV

Typnummer: 65323

Status: Test, undokumentiert (DNSSEC)

Beschreibung: speichert den Hashcode des DNSKEYs einer anderen Domain.

Syntax: *Quersumme Algorithmus DigestType Digest*

Mehr zum Thema: siehe `draft-ietf-dnsext-delegation-signer-05.txt`

7.5.10 DNAME

Typnummer: 39

Status: Standard

Beschreibung: Ein DNAME-Record weist den Resolver an, die Abfrage unter einem anderen Domainnamen zu wiederholen.

Beispiel:

Wenn der Fachbereich Informatik den Namen seiner Subdomain in `cs` (»Computer Sciences«) ändern möchte, kann der Hostmaster mit einem einzigen DNAME-Record alle Abfragen, die noch auf den alten Domainnamen zugreifen, auf den neuen Domainnamen *umleiten*:

```
$ORIGIN informatik.uni-frankfurt.de.
@    DNAME   cs.uni-frankfurt.de.
     MX      10 mail.cs.uni-frankfurt.de.
```

Listing 7.21: Beispiel zu DNAME-Records

Achtung: Verlassen Sie sich *nicht* auf DNAME-Records. Obwohl die IETF sie im Jahr 1999 zum Internet-Standard erklärt hat, gibt es immer noch alte Resolver und Nameserver (z.B. BIND8), die diese DNS-Erweiterung nicht beherrschen.

Mehr zum Thema: Siehe RFC2672.

7.5.11 DNSKEY

Typnummer: 48

Status: Standard (DNSSEC)

Beschreibung: Ein DNSKEY-Record speichert einen öffentlichen Schlüssel, der bei DNSSEC zum Prüfen von Signaturen erforderlich ist.

Das Datenfeld eines DNSKEY-Records hat folgende Syntax:

Flags 3 *Algorithmus Schlüssel*

- Die möglichen Werte für das Flags-Feld lauten:

Wert	Beschreibung
0	Der Key gilt *nicht* für Zonen, sondern nur für einzelne Records (z.B. im TKEY- oder SIG(0)-Verfahren).
256	Der Schlüssel gilt für Zonen (für DNSSEC erforderlich).
257	Der Schlüssel gilt für Zonen und das Secure-Entry-Point-(SEP-)Flag ist gesetzt (siehe RFC3757).

Tabelle 7.11: Flags-Feld des DNSKEY-Records

- Den Algorithmus können Sie wahlweise numerisch oder über sein Kürzel angeben:

Wert	Kürzel	Beschreibung
1	RSAMD5	RSA/MD5 (siehe RFC2537), nicht bei Zonen einsetzbar
3	DSA	DSA/SHA-1(siehe RFC2536)
5	RSASHA1	RSA/SHA-1 (siehe RFC3110). Empfohlener Wert.
253	PRIVATEDNS	zur freien Verfügung (siehe `draft-ietf-dnsext-dnssec-records-*.txt`).

Tabelle 7.12: Algorithmus-Feld des DNSKEY-Records

Wert	Kürzel	Beschreibung
254	PRIVATEOID	zur freien Verfügung (siehe draft-ietf-dnsext-dnssec-records-*.txt).

Tabelle 7.12: Algorithmus-Feld des DNSKEY-Records (Fortsetzung)

- Geben Sie den Schlüssel im Base64-Format an. Zur besseren Übersichtlichkeit können Sie ihn mit Leerzeichen, Tabulatorzeichen oder Zeilenumbrüchen in beliebig viele Teile zerlegen.

Achtung: Wenn Sie den Schlüssel über mehrere Zeilen verteilen, müssen Sie die Zeilenumbrüche mit runden Klammern schützen.

Beispiel:

```
dskey.example.com. 86400 IN DNSKEY 256 3 5 (
              AQOeiiR0GOMYkDshWoSKz9Xz
              fwJr1AYtsmx3TGkJaNXVbfi/
              2pHm822aJ5iI9BMzNXxeYCmZ
              DRD99WYwYqUSdjMmmAphXdvx
              egXd/M5+X7OrzKBaMbCVdFLU
              Uh6DhweJBjEVv5f2wwjM9Xzc
              nOf+EPbtG9DMBmADjFDc2w/r
              ljwvFw==
            ) ;  key id = 60485
```

Listing 7.22: Beispiel eines DNSKEY-Records (aus draft-ietf-dnsext-dnssec-records-08.txt)

Mehr zum Thema: Siehe Abschnitt 12.3 und draft-ietf-dnsext-dnssec-records-*.txt auf http://www.ietf.org/internet-drafts/

7.5.12 DS

Typnummer: 43

Status: Standard (DNSSEC)

Beschreibung: Ein »Delegation Signer«-Record speichert den Hashcode des DNSKEYs einer delegierten Subdomain. Mit dem DS-Record bestätigt der Hostmaster, dass der angegebene Schlüssel dem Hostmaster der Subdomain gehört.

Achtung: Die Parent-Domain muss den DS-Record mit ihrem eigenen Schlüssel signieren und sowohl Parent- als auch Subdomain müssen ihre DNSKEY-Records selbst signieren. Wenn eine der Signaturen fehlt, ist die Zone ungesichert.

Das Programm `dnssec-signzone` kann die DS-Records für Sie erstellen.

Das Datenfeld eines DS-Records hat folgende Syntax:

Quersumme Algorithmus 1 *Hashcode*

- Die Quersumme des öffentlichen Schlüssels können Sie mit folgendem C-Code berechnen:

```
unsigned int
keytag (
   unsigned char key[], /* the RDATA part of the DNSKEY RR */
   unsigned int keysize /* the RDLENGTH */
   )
{
   unsigned long ac;    /* assumed to be 32 bits or larger */
   int i;               /* loop index */

   for ( ac = 0, i = 0; i < keysize; ++i )
      ac += (i & 1) ? key[i] : key[i] << 8;
   ac += (ac >> 16) & 0xFFFF;
```

```
    return ac & 0xFFFF;
}
```

Listing 7.23: Quersummenberechnung (aus `draft-ietf-dnsext-dnssec-records-08.txt`)

- Den *`Algorithmus`* entnehmen Sie bitte der Tabelle 7.12 in Abschnitt 7.5.11 (DNSKEY).
- Der *`Hashcode`* besteht aus 40 Hexadezimalziffern, die sich durch Anwenden der SHA1-Funktion auf das gesamte Datenfeld des DNSKEY-Records ergeben.

Beispiel:

```
dskey.example.com. 86400 IN DS 60485 5 1 (
                2BB183AF5F22588179A53B0A
                98631FAD1A292118 )
```

Listing 7.24: DS- zu dem DNSKEY-Record in Listing 7.22

Mehr zum Thema: Der DS-Record wird in RFC3658 und `draft-ietf-dnsext-dnssec-records-08.txt` (oder neuer) auf `http://www.ietf.org/internet-drafts/` beschrieben.

7.5.13 GPOS

Typnummer: 27

Status: experimentell

Beschreibung: Ein »Global Position«-Record beschreibt den Standort eines Hosts.

Geben Sie im Datenfeld folgende Werte an:

- der Breitengrad: Ein Dezimalbruch zwischen `-90` (Südpol) und `+90` (Nordpol). `0` steht für den Äquator.
- der Längengrad: Ein Dezimalbruch zwischen `-180` (westlich) und `+180` (östlich). `0` steht für den Längengrad von Greenwich.

- die Höhe: Ein Dezimalbruch in Metern zwischen `-6379000` (Erdmittelpunkt) und 10^{255}. `0` steht für die durchschnittliche Höhe des Meeresspiegels.

Achtung: Verwenden Sie einen Punkt als Dezimaltrennzeichen. Die Angaben dürfen nicht länger (genauer) als jeweils 256 Zeichen sein.

Beispiel:

```
$ORIGIN cs.curtin.edu.au.
; Die australische Stadt Perth liegt auf
;   32" 7' 19" südlicher Breite,
;   116" 2' 25" östlicher Länge
;   10 Meter über dem Meeresspiegel

; Die Rechner marsh und lillee stehen in Perth:
marsh       IN   A       134.7.1.1
            IN   GPOS    -32.6882 116.8652 10.0
lillee      IN   A       134.7.1.2
            IN   GPOS    -32.6882 116.8652 10.0

; Der Rechner hinault steht etwas außerhalb
; (in den Bergen, 10 Grad nördlich von Perth):
hinault     IN   A       134.7.1.23
            IN   GPOS    -22.6882 116.8652 250.0
```

Listing 7.25: Beispiel zum GPOS-Record (aus RFC1712)

Achtung: Das IETF empfiehlt, statt der GPOS-Records die präziseren LOC-Records zu verwenden.

Mehr zum Thema: Siehe RFC1712.

7.5.14 HINFO

Typnummer: 13

Status: Standard

Beschreibung: Ein »Host-Information«-Record speichert technische Details (z.B. Prozessor und Betriebssystem) eines Hosts.

Beispiel:

```
$ORIGIN ti.informatik.uni-frankfurt.de.
tiap1    A        141.2.3.150
         HINFO    "Apollo DN4500" "sr10.3"
tiap12   A        141.2.3.161
         HINFO    "Sun 4/65mono" "Solaris 1.0"
tiap5    A        141.2.3.154
         HINFO    "HP/Apollo 730" "HP/UX 8.7"
ticpr    A         141.2.3.171
         HINFO    "HP Laserjet4" "unavail"
tilv5    A        141.2.3.172
         HINFO    "Tektronix Chiptester" "unavail"
tipc1    A        141.2.3.180
         HINFO    "PC/AT SLIP" "MS DOS 3.1"
tipc11   A        141.2.3.190
         HINFO    "Atari ST SLIP" "TOS 1.4"
tipc15   A        141.2.3.194
         HINFO    "Apple MacIntosh" "Finder"
tisrv    A        141.2.3.173
         HINFO    "Emulex Terminalserver" "Performance 2.1"
```

Listing 7.26: Beispiel zu HINFO-Records

Diese Informationen wurden in den 1980ern von einigen Netzanwendungen verwendet, um z.B. den Zeichensatz schon beim Download von einem FTP-Server, der einen anderen Zeichensatz verwendet, anzupassen.

Achtung: Die Angabe des Betriebssystems ist heute nicht mehr zu empfehlen, da sie Hinweise für Hacker enthält und die Zeichensätze auf Applikationsebene umgewandelt werden.

Mehr zum Thema: Der HINFO-Record wird in RFC1035 definiert. Eine Liste historischer Prozessor- und Betriebssystemnamen finden Sie in RFC1010.

7.5.15 ISDN

Typnummer: 20

Status: experimentell

Beschreibung: Ein ISDN-Record gibt die ISDN-Rufnummer an, unter der ein Host erreichbar ist.

Ein Beispiel finden Sie in Listing 7.47 beim RT-Record (Abschnitt 7.5.36).

Mehr zum Thema: Siehe RFC1183.

7.5.16 KEY

Typnummer: 25

Status: Standard (DNSSEC)

Ein KEY-Record speichert einen öffentlichen Schlüssel zur Verwendung mit TKEY oder SIG(0).

Das Datenfeld des KEY-Records hat folgende Syntax:

`Flags Protokoll Algorithmus Schlüssel`

- RFC2535 beschreibt die möglichen Werte für das *`Flags`*- und *`Protokoll`*-Feld. In der Praxis werden Sie jedoch immer `0 3` angeben, was die *`Flags`* auf `USER` und das *`Protokoll`* auf `DNSSEC` setzt.

- Den *Algorithmus* entnehmen Sie bitte der Tabelle 7.12 in Abschnitt 7.5.11 (DNSKEY).
- Geben Sie den *Schlüssel* im Base64-Format an. Zur besseren Übersichtlichkeit können Sie ihn mit Leerzeichen, Tabulatorzeichen oder Zeilenumbrüchen in beliebig viele Teile zerlegen.

Beispiel:

```
dskey.example. KEY 0 3 1 (
     AQPwHb4UL1U9RHaU8qP+Ts5bVOU1s7fYbj2b3CCbzNdj
     4+/ECd18yKiyUQqKqQFWW5T3iVc8SJOKnueJHt/Jb/wt )
```

Listing 7.27: Beispiel zum KEY-Record (aus RFC3658)

Achtung: Wenn Sie den Schlüssel über mehrere Zeilen verteilen, müssen Sie die Zeilenumbrüche mit runden Klammern schützen.

Achtung: Verwenden Sie anstelle des KEY-Records den neueren DNSKEY-Record, wenn Sie einen Schlüssel für die DNSSEC-Signierung Ihrer Zone angeben möchten.

Mehr zum Thema: RFC2535 definiert den KEY-Record und RFC3445 begrenzt die einzusetzenden Flags und Protokolle.

7.5.17 KX

Typnummer: 36

Status: experimentell

Beschreibung: Ein »Key exchanger«-Record nennt die Key-Exchange-Systeme, die für den angegebenen Rechner oder die Domain zuständig sind (z.B. bei IPsec).

Das Datenfeld des KX-Records hat folgende Syntax:

Kosten Hostname

- Die *Kosten* verhält sich wie beim MX-Record (siehe Abschnitt 7.5.25).
- Geben Sie für *Hostname* den Namen eines CNAME-, A- oder AAAA-Records an.

Beispiel:

```
dialin.example.    KX  1 radius1.example.
                   KX  2 radius1.example.
                   KX  3 radius1.example.
```

Listing 7.28: Beispiel zum KX-Record

Achtung: Wenn das Key-Exchange-System nicht mit einem Punkt endet, wird der Nameserver beim Einlesen des Zonefiles den Namen der Zone (`$ORIGIN`) anhängen.

Mehr zum Thema: Siehe RFC2230.

7.5.18 LOC

Typnummer: 29

Status: experimentell

Beschreibung: Ein »Location«-Record beschreibt den Standort eines Hosts oder eines Netzes.

Das Datenfeld des LOC-Records hat folgende Syntax:

```
Breitengrad [Minuten [Sekunden]] (N|S)
Längengrad [Minuten [Sekunden]] (E|W)
Höhe[m]
[ Umkreis[m]
  [ HorizontaleAbweichung[m]
```

```
    [ VertikaleAbweichung[m] ]
  ]
]
```

Listing 7.29: Syntax des LOC-Datenfeldes

Beispiel:

```
$ORIGIN curtin.edu.au.
; Das Netz der Curtin-Uni liegt im australischen Perth:
;   32" 7' 19" südlicher Breite,
;   116" 2' 25" östlicher Länge
;   10 Meter über dem Meeresspiegel
@ LOC 32 7 19 S 116 2 25 E 10m
```

Listing 7.30: Beispiel zum LOC-Record

Mehr zum Thema: Siehe RFC1876.

7.5.19 MB

Typnummer: 7

Status: experimentell

Beschreibung: Ein »Mailbox«-Record nennt den Mailserver (A-Record!) zu dem angegebenen Usernamen.

Beispiel:

```
$ORIGIN ISI.EDU.
; "Normale" User lesen ihre Mail auf VENERA (notfalls VAXA)
@       MX      10 VENERA
        MX      20 VAXA
VENERA  A       10.1.0.52
        A       128.9.0.32
VAXA    A       10.2.0.27
        A       128.9.0.33
A       A       26.3.0.103
```

```
; Einige User (moe@isi.edu etc.) lesen ihre Mail jedoch auf A
MOE     MB      A.ISI.EDU.
LARRY   MB      A.ISI.EDU.
CURLEY  MB      A.ISI.EDU.

; Mitarbeiter im "Außendienst":
PAUL    MR      PAUL.LCS.MIT.EDU. ; Mails für paul@isi.edu
                                  ; bitte an
                                  ; paul@lcs.mit.edu senden

; Verteilerliste stooges@isi.edu :
STOOGES MG      MOE         ; moe@isi.edu ist eingeschrieben
        MG      LARRY
        MG      CURLEY
        MINFO   (
          STOOGES-REQUEST   ; stooges-request@isi.edu
                            ; zum Anmelden und bei Fragen
          ROOT.ISI.EDU.     ; root@isi.edu
        )                   ; erhält alle Zustellfehler
```

Listing 7.31: Beispiel zu MB-, MG- und MINFO-Records (aus RFC1035)

Achtung: Verlassen Sie sich nicht auf MB-Records, da die heutigen Mail Transfer Agents (MTA) nur noch nach A- und MX-Records suchen.

Achtung: Wenn der im Datenfeld angegebene Mailserver nicht mit einem Punkt endet, wird der Nameserver beim Einlesen des Zonefiles den Namen der Zone (`$ORIGIN`) anhängen.

Achtung: Sie dürfen hier keinen CNAME angeben.

Mehr zum Thema: Siehe RFC1035.

7.5.20 MD

Typnummer: 3

Status: experimentell

Beschreibung: Ein »Mail destination«(MD)-Record nennt den primären Mailserver (A-Record!) einer Domain, was dem Verhalten eines »MX 0«-Records entspricht.

Ein »Mail forwarder«(MF)-Record nennt einen weiteren Mailserver (A-Record!) einer Domain, was dem Verhalten eines »MX 10«-Records entspricht.

Beispiel:

```
$ORIGIN ISI.ARPA.
A  A  10.1.0.32
   MD A.ISI.ARPA. ; A ist primärer Mailserver für a.isi.arpa
   MF F.ISI.ARPA. ; F ist Failover-Mailserver für a.isi.arpa
```

Listing 7.32: Beispiel zu MD- und MF-Records (aus RFC833)

Achtung: Verwenden Sie besser MX-Records, da die heutigen MTAs MD- und MF-Records ignorieren. RFC1123 schlägt vor, dass Nameserver MD- und MF-Records beim Einlesen von Zonefiles überspringen.

Achtung: Wenn der im Datenfeld angegebene Mailserver nicht mit einem Punkt endet, wird der Nameserver beim Einlesen des Zonefiles den Namen der Zone (`$ORIGIN`) anhängen.

Achtung: Sie dürfen hier keinen CNAME angeben.

Mehr zum Thema: Siehe RFC833.

7.5.21 MF

Typnummer: 4

Status: experimentell

siehe MD-Record (Abschnitt 7.5.20).

7.5.22 MG

Typnummer: 8

Status: experimentell

Beschreibung: Ein »Mail group«(MG)-Record nennt die Empfängeradressen (MB-Records) einer Verteilerliste.

Ein »Mailbox or Mail list information«(MINFO)-Record nennt die beiden administrativen E-Mail-Adressen (MB-Records, zum Bestellen und bei Zustellfehlern) einer Verteilerliste.

Beispiel: siehe Listing 7.31 in Abschnitt 7.5.19 (MB-Record).

Achtung: Konfigurieren Sie Ihre Verteilerlisten besser in der `aliases`-Datei auf dem Mailserver oder setzen Sie eine Mailinglistensoftware (z.B. Majordomo) ein, da die heutigen MTAs MG- und MINFO-Records ignorieren.

Mehr zum Thema: Siehe RFC1035.

7.5.23 MINFO

Typnummer: 14

Status: experimentell

Beschreibung: siehe MG-Record (Abschnitt 7.5.22).

7.5.24 MR

Typnummer: 9

Status: experimentell

Beschreibung: Ein »Mailbox Rename«-Record soll bewirken, dass ein MTA die Mails für eine E-Mail-Adresse zu einer anderen E-Mail-Adresse umleitet, die in einer anderen Domain liegen kann.

Beispiel: siehe Listing 7.31 in Abschnitt 7.5.19 (MB-Record).

Achtung: Verlassen Sie sich nicht auf MB-Records, da die heutigen MTAs nur noch nach A- und MX-Records suchen.

Mehr zum Thema: Siehe RFC1035.

7.5.25 MX

Typnummer: 15

Status: Standard

Beschreibung: Mail-Exchanger-(MX-)Records nennen die Mailserver einer Domain. Das Datenfeld besteht aus zwei Teilen:

Feld	Beschreibung
Kosten	Eine ganze Zahl zwischen 0 und 65535. Wenn für eine Domain *mehrere MX-Records* bestehen, bestimmt dieser Wert die Reihenfolge, in der der sendende Mail Transfer Agent (MTA) die Mailserver der Domain anspricht: Er spricht den Mailserver mit den niedrigsten Kosten zuerst an. Wenn dieser nicht erreichbar ist, den mit den nächsthöheren usw. Wenn mehrere *Mailserver* mit denselben *Kosten* angegeben sind, soll der MTA (laut RFC1123) alle Mails gleichmäßig auf diese verteilen.

Tabelle 7.13: Datenfeld des MX-Records

Feld	Beschreibung
	Achtung: SPAM wird oft an den *Mailserver* mit den höchsten *Kosten* gesendet, da dieser in vielen Fällen beim Provider steht und keinen Spamschutz enthält.
Mailserver	Der Name eines A-Records.
	Achtung: Wenn der Mailserver nicht mit einem Punkt endet, wird der Nameserver beim Einlesen des Zonefiles den Namen der Zone (`$ORIGIN`) anhängen.
	Achtung: Sie dürfen hier keinen CNAME angeben.

Tabelle 7.13: Datenfeld des MX-Records (Fortsetzung)

Beispiel:

```
$ORIGIN informatik.uni-frankfurt.de.
@ 38400 IN MX 10 hera.rbi
@ 38400 IN MX 20 zeus.rbi

$ORIGIN rbi.informatik.uni-frankfurt.de.
hera 38400 IN A 141.2.1.1
zeus 38400 IN A 141.2.15.24
```

Listing 7.33: Beispiel zu MX-Records

Diese Konfiguration bewirkt, dass MTAs alle Mails für die Domain `informatik.uni-frankfurt.de` an die IP-Adresse `141.2.1.1` oder (falls diese nicht erreichbar ist) `141.2.15.24` senden.

Mehr zum Thema: Der MX-Record wird in RFC1035 definiert. Abschnitt 9.2.2 beschreibt die Verwendung mehrerer MX-Records für Failover. Eine genaue Beschreibung aller DNS-Abfragen für SMTP finden Sie in RFC2821.

7.5.26 NAPTR

Typnummer: 35

Status: Standard

Beschreibung: Ein »Name authority Pointer« speichert eine Regel zur Verarbeitung von URIs.

Uniform Resource Identifier (URI) beschreiben Dienste (z.B. den Geschäftsbericht einer Firma) und die verschiedenen Adressen im Netz (z.B. Webseiten und deren Mirrors, FTP-Server und E-Mail-Adressen), die diesen Dienst anbieten.

Man unterscheidet zwei verschiedene Arten von URIs:

- Uniform Resource Name (URN): benennen eine Ressource eindeutig.
  Beispiel: `urn:isbn:3-8266-1502-6`.
- Uniform Resource Locator (URL): beschreiben die Adresse einer Ressource im Netz.
  Beispiel: `http://www.amazon.de/search.asp?isin=3-8266-1502-6`

NAPTR-Records sind an der Auflösung von URNs zu URLs beteiligt.

Das Datenfeld eines NAPTR-Records hat folgende Syntax:

`Order Preference Flags Service Regexp Replacement`

Erklärung:

Feld	Beschreibung
`Order`	Eine natürliche Zahl zwischen 0 und 65535. Sie bestimmt die Reihenfolge, in der der URN-Resolver die NAPTR-Records prüft. Der URN-Resolver legt sich auf die *`Order`* des Records mit der kleinsten *`Order`* fest, dessen *`Regexp`* auf den ursprünglichen Namen zutrifft oder "" lautet.
`Preference`	Eine natürliche Zahl zwischen 0 und 65535. Der URN-Resolver kann bei der Auswahl aus den Records mit gleicher *`Order`* die Records mit kleineren Werten bevorzugen (muss aber nicht).

Tabelle 7.14: Felder eines NAPTR-Records

Feld	Beschreibung
Flags	Eine Folge von Buchstaben und Ziffern (Reihenfolge und Groß/Kleinschreibung unwesentlich), die die weitere Vorgehensweise beschreibt. Die Bedeutung dieser Flags ist von Anwendung zu Anwendung unterschiedlich. RFC2916 schlägt folgende Verwendung vor: "S": Das Ergebnis als SRV-Record auflösen. "A": Das Ergebnis als A- oder AAAA-Record auflösen. "U": Das Ergebnis als URI an die Anwendung zurückgeben. "P": Weitere NAPTR-Records zu dem Ergebnis abfragen (mit den in *Service* angegebenen Protokollen). "": Weitere NAPTR-Records zu dem Ergebnis abfragen.
Service	Gibt das Protokoll und einen oder mehrere Protokollauflösungsdienste an (mit Pluszeichen getrennt), die für diese NAPTR-Anwendung erforderlich sind. **Beispiel:** "thttp+I2L+I2C+I2R" Die möglichen Inhalte für dieses Feld sind von Anwendung zu Anwendung unterschiedlich. In einigen Fällen können Sie die Angabe des Protokolls oder der Auflösungsdienste weglassen und das Feld kann sogar leer sein ("").
Regexp *Replacement*	Ein regulärer Ausdruck, der den ursprünglichen Namen durch den Namen eines weiteren NAPTR-Records ersetzt. Den Text innerhalb der ersten, öffnenden, runden Klammer können Sie mit \1 referenzieren, den der zweiten mit \2 usw. **Beispiel:** "!^http://([^/:]+)!\1!i" ersetzt die URL einer Webseite durch den Hostnamen ihres Webservers. Wenn Sie alle Namen durch eine Konstante ersetzen möchten, geben Sie "" an und schreiben die Konstante in das *Replacement*-Feld. Die Felder *Regexp* und *Replacement* schließen sich gegenseitig aus.

Tabelle 7.14: Felder eines NAPTR-Records (Fortsetzung)

Beispiel:

```
;              order pref flags service regexp replacement
http.uri.arpa. NAPTR 100 90 "" "" "!^http://([^/:]+)!\1!i"
cid.uri.arpa.  NAPTR 100 10 "" ""                        (
                          "!^cid:.+@([^\.]+\.)(.*)$!\2!i")
foo.urn.arpa.  NAPTR 100 10 "s" "foolink+I2L+I2C" ""     (
                                 foolink.udp.example.com.)
               NAPTR 100 20 "s" "rcds+I2C" ""            (
                                   rcds.udp.example.com.)
               NAPTR 100 30 "s" "thttp+I2L+I2C+I2R"  ""  (
                                  thttp.tcp.example.com.)
```

Listing 7.34: NAPTR-Einträge

Mehr zum Thema: NAPTR RR (RFC3403), URN-Syntax (RFC2141), URI-Syntax (RFC2396), URI resolution services necessary for URN resolution (RFC2483)

7.5.27 NS

Typnummer: 2

Status: Standard

Beschreibung: Ein NS-Record weist der angegebenen Zone einen Nameserver (A- oder AAAA-Record!) zu.

Dabei handelt es sich entweder um einen Authority- oder einen Delegations-Eintrag:

- Wenn der Zonenname eines NS-Records @ lautet oder der aktuellen Zone entspricht, zählt der NS-Record zu den *Authority*-Informationen der Zone. Dieser Eintrag sollte in keinem Zonefile fehlen.
- Sonst handelt es sich um die *Delegation* einer Subdomain, die hiermit zu einer eigenen Zone wird.

Achtung: Die delegierenden NS-Records müssen mit den Authority-NS-Records (@) der delegierten Zone übereinstimmen.

Der Einsatz mehrerer Nameserver in einer Zone erhöht die Verfügbarkeit und ist sogar Aufnahmekriterium bei einigen TLD-Registries. Wenn eine Zone *mehrere NS-Records* enthält, werden Clients (andere Nameserver) alle Nameserver der Zone reihum ansprechen und sich dann auf den Nameserver konzentrieren, der am schnellsten geantwortet hatte. Dabei spielt es keine Rolle, ob diese Zone auf den Nameservern als Typ `master` oder `slave` konfiguriert ist.

Mehr zum Thema: Details zur Konfiguration von Zonen mit mehreren Nameservern finden Sie in Abschnitt 1.9 und Abschnitt 6.3.11.

Wenn der NS-Record auf einen A-Record verweist, der in einer Subdomain der aktuellen Zone liegt, und diese Subdomain auf separater Nameserverhardware läuft, müssen Sie eine Kopie des A-Records in der aktuellen Zone unterbringen. Dieses Verfahren wird in RFC1034 als »glueing« (kleben) und die Kopie als *»Glue-Record«* bezeichnet.

Beispiel zu Authority-NS-Records:

```
$ORIGIN informatik.uni-frankfurt.de.
@ 38400  IN   NS   hera.rbi
@ 38400  IN   NS   ws-muel.win-ip.dfn.de.
; ...
; Glue-Record (weil NS hera.rbi in Subdomain):
hera.rbi 38400 IN A 141.2.1.1
```

Listing 7.35: Authority-NS- und Glue-Records

Beispiele zu delegierenden NS-Records:

Die Zone `de.` entsteht durch Delegation der Domain `de` aus der Root(.)-Zone

Das Zonefile der Root-Zone finden Sie auf `ftp://ftp.internic.net/domain/root.zone.gz`.

```
$ORIGIN .
de.          NS      a.nic.de.
de.          NS      b.de.net.
de.          NS      c.de.net.
de.          NS      d.de.net.
de.          NS      e.nic.de.
de.          NS      f.nic.de.
de.          NS      g.de.net.
de.          NS      h.nic.de.
de.          NS      i.de.net.
de.          NS      j.nic.de.
de.          NS      k.nic.de.
; Glue-Records:
a.nic.de.    A       81.91.161.5
e.nic.de.    A       193.171.255.34
f.nic.de.    A       193.0.0.237
h.nic.de.    A       192.36.144.211
j.nic.de.    A       66.35.208.44
k.nic.de.    A       210.81.13.179
```

Listing 7.36: Delegation der `de.`-Zone

Die Zone `uni-frankfurt.de` entsteht durch Delegation der Domain `uni-frankfurt` aus der Zone `de.`:

```
$ORIGIN de.
uni-frankfurt     NS      chaplin.rz.uni-frankfurt.de.
uni-frankfurt     NS      ws-mue1.win-ip.dfn.de.
uni-frankfurt     NS      wattdenn.rz.uni-frankfurt.de.
; Glue Records:
chaplin.rz.uni-frankfurt.de.    A   141.2.22.74
wattdenn.rz.uni-frankfurt.de.   A   141.2.149.10
```

Listing 7.37: Delegation der Zone `uni-frankfurt.de.`

Die Zone `informatik.uni-frankfurt.de.` entsteht durch Delegation der Domain `informatik` aus der Zone `uni-frankfurt.de.`:

```
$ORIGIN uni-frankfurt.de.
informatik        NS      chaplin.rz.uni-frankfurt.de.
informatik        NS      ws-mue1.win-ip.dfn.de.
informatik        NS      wattdenn.rz.uni-frankfurt.de.
```

Listing 7.38: Delegation der Zone `informatik.uni-frankfurt.de.`

Achtung: Wenn der Name des Nameservers oder der Zone nicht mit einem Punkt endet, wird der Nameserver beim Einlesen des Zonefiles den Namen der Zone (`$ORIGIN`) anhängen.

Achtung: Sie dürfen keinen CNAME als Nameserver angeben.

Mehr zum Thema: Siehe RFC1035.

7.5.28 NSAP

Typnummer: 22

Status: Standard

Beschreibung: Ein »Network service access point«-Record speichert NSAP-Adressen.

NSAP-Adressen wurden hauptsächlich eingesetzt in Verbindung mit

- dem Connectionless Network Protocol (CLNP), das in RFC986 beschrieben ist, und
- dem Private Network-to-Network Interface (PNNI) Protocol in ATM, das u.a. in RFC1932 beschrieben ist.

Beispiel:

```
$ORIGIN nsap.nist.gov.
cisco1 NSAP (
         0x47.0005.80.005a00.0000.0001.e133.aaaaaa000151.00 )
       A 129.6.224.151
       A 129.6.225.151
       A 129.6.229.151
```

Listing 7.39: Beispiel eines NSAP-Records (aus RFC1706)

Mehr zum Thema: Siehe RFC1706.

7.5.29 NSAP-PTR

Typnummer: 23

Status: experimentell

Beschreibung: Ein »NSAP-Pointer«-Record nennt den Hostnamen zu einer gegebenen NSAP-Adresse.

NSAP-Adressen wurden hauptsächlich in Verbindung mit dem Connectionless Network Protocol (CLNP) eingesetzt, das in RFC986 beschrieben ist.

Beispiel:

```
$ORIGIN 11110031f67293.nsap-in-addr.arpa.
67894444333322220000 NSAP-PTR host.school.de.
```

Listing 7.40: Beispiel zum NSAP-PTR-Record (aus RFC1348)

Achtung: Das Verweisziel eines NSAP-PTR-Records wird grundsätzlich absolut angegeben und muss also auf einen Punkt enden.

Achtung: Sie dürfen keinen CNAME als Verweisziel angeben.

Mehr zum Thema: RFC1348 definiert diesen Typ und RFC1706 beschreibt, wie man ihn durch PTR-Records ersetzt.

7.5.30 NSEC

Typnummer: 47

Status: Standard (DNSSEC)

Beschreibung: Ein »Next Secure«-Record gibt den Namen des nächsten und die Typen des aktuellen RRset an. Daraus kann man ableiten, welche Records in einer Zone *nicht* existieren.

Wenn Sie den NSEC-Record signieren, kann der Anfragende auch die Echtheit der Antwort, dass ein Record *nicht* existiert, beglaubigen.

Achtung: Wer die NSEC-Records Ihrer Zone verfolgt, erhält ein vollständiges Abbild Ihres Zonefiles. Dagegen hilft auch das Verbieten von Zonetransfers nicht.

Achtung: Sie brauchen die NSEC-Records und deren Signaturen (RRSIG-Records) nicht selbst einzugeben; das Tool `dnssec-signzone` erledigt dies für Sie.

Beispiel:

```
alfa.example.com. 86400 IN NSEC (
        host.example.com.          ; nächster Host
        A MX RRSIG NSEC TYPE1234 ) ; Typen im RRset alfa
```

Listing 7.41: Beispiel zum NSEC-Record (aus `draft-ietf-dnsext-dnssec-records-08.txt`)

Der NSEC-Record des (alphabetisch) letzten RRset einer Zone verweist auf den Namen der Zone.

Mehr zum Thema: Der NSEC-Record wird in RFC3845 und der Aufruf von `dnssec-signzone` in Abschnitt 12.4.2 beschrieben.

7.5.31 NXT

Typnummer: 30

Status: experimentell

Beschreibung: Der NXT-Record hat dieselbe Aufgabe wie der NSEC-Record. Da viele NXT-Implementierungen jedoch nur Typnummern bis 127 erlauben, wurde der NXT-Record vom NSEC-Record abgelöst.

Mehr zum Thema: Die Definition des NXT-Records finden Sie in RFC2535 und Details zum NSEC-Record in Abschnitt 7.5.30.

7.5.32 PTR

Typnummer: 12

Status: Standard

Beschreibung: Ein »Pointer«-Record verweist auf einen A-Record. Er ist für Reverse queries (Fragen nach dem Host- oder Netznamen zu einer IP-Adresse) vorgesehen und befindet sich meist in der Domain `in-addr.arpa.`.

Beispiel:

```
$ORIGIN 1.2.141.in-addr.arpa.
0  IN PTR  rbi.informatik.uni-frankfurt.de. ; siehe Abs. 9.8
1  IN PTR  hera.rbi.informatik.uni-frankfurt.de.
2  IN PTR  diokles.rbi.informatik.uni-frankfurt.de.
3  IN PTR  hermes.rbi.informatik.uni-frankfurt.de.
```

Listing 7.42: PTR-Records

Ein Nameserver, der den Hostnamen zu der IP-Adresse `141.2.1.1` sucht, wird schließlich an den Nameserver der Zone `1.2.141.in-addr.arpa` gelangen, die-

sen nach dem PTR-Record von `1.1.2.141.in-addr.arpa.` fragen und die Antwort `hera.rbi.informatik.uni-frankfurt.de.` erhalten.

Achtung: Das Verweisziel eines PTR-Records wird grundsätzlich absolut angegeben und muss also auf einen Punkt enden.

Achtung: Sie dürfen keinen CNAME als Verweisziel angeben.

Mehr zum Thema: Siehe RFC1035.

7.5.33 PX

Typnummer: 26

Status: Standard

Beschreibung: Ein »Pointer to X400/RFC822-Mapping-Information«-Record definiert Gateways und Adressumsetzung für die Kommunikation mit X400-Mailsystemen.

Beispiel:

```
$ORIGIN x42d.de.
*.ADMD-pkz  IN  PX (
                    50              ; Preference
                    pkz.de.         ; MAP RFC822
                    ADMD-pkz.C-de.  ; MAP X400
                  )
```

Listing 7.43: Beispiel zum PX-Record (aus RFC2163)

Mehr zum Thema: Siehe RFC2163.

7.5.34 RP

Typnummer: 17

Status: experimentell

Beschreibung: Ein »Responsible Person«-Record speichert die E-Mail-Adresse des zuständigen Administrators und den Namen eines TXT-Records, der beliebige Infos (z.B. die Raum- oder Rufnummer des Administrators) enthalten kann.

Achtung: Stellen Sie allen Punkten, die im Usernamen der E-Mail-Adresse enthalten sind, einen Backslash (\) voran und ersetzen Sie den Klammeraffen (@) durch einen Punkt.

Achtung: Wenn die E-Mail-Adresse nicht mit einem Punkt endet, wird der Nameserver beim Einlesen des Zonefiles den Namen der Zone (`$ORIGIN`) anhängen.

Geben Sie einen Punkt anstelle des Namens des TXT-Records an, wenn Sie außer der E-Mail-Adresse keine weiteren Daten angeben möchten.

Beispiel:

```
$ORIGIN UMD.EDU.
TRANTOR  A   128.8.10.14
;            email                   txt
         RP  louie.trantor.          LAM1.people
         RP  petry.netwolf           petry.people
         RP  root.trantor            ops.CS
         RP  gregh.sunset.umd.edu.   .

LAM1.people  TXT  "Louis A. Mamakos (301) 454-2946"
petry.people TXT  "Michael G. Petry (301) 454-2946"
ops.CS       TXT  "CS Operations Staff (301) 454-2943"
```

Listing 7.44: Beispiel zu RP- und TXT-Records (aus RFC1183)

Mehr zum Thema: Siehe RFC1183.

7.5.35 RRSIG

Typnummer: 46

Status: Standard (DNSSEC)

Beschreibung: Ein »Resource Record Signature«: enthält die Signatur für einen anderen Record.

Achtung: Sie brauchen die RRSIG-Records nicht selbst einzugeben; das Tool `dnssec-signzone` erledigt dies für Sie.

Das Datenfeld des RRSIG-Records besteht aus folgenden Feldern:

```
Typ
Algorithmus
Tiefe
RecordTTL
GültigkeitsEnde GültigkeitsBeginn
Quersumme Signierer Signatur
```

Listing 7.45: Felder des RRSIG-Records

Erklärung:

Feld	Beschreibung
Typ	gibt den Typ des Records an, auf den sich diese Signatur bezieht. Sie können den Typ wahlweise über seinen Namen (z.B. `CNAME`) oder anonym (z.B. `TYPE5`) angeben.
Algorithmus	nennt das Format des Schlüssels und der Signatur. Sie können den Algorithmus numerisch oder als Kürzel angeben. Eine Liste möglicher Werte entnehmen Sie bitte der Tabelle 7.12.

Tabelle 7.15: Erklärung zu den Feldern des RRSIG-Records

Feld	Beschreibung
Tiefe	gibt die Zahl der Labels im FQDN dieses Records an, die nicht »*« oder »« lauten. Beispiele: 3 für `www.example.com.` 2 für `*.example.com.` 5 für `www.rbi.informatik.uni-frankfurt.de.`
RecordTTL	gibt die TTL des Records an, auf die sich diese Signatur bezieht.
GültigkeitsEnde	gibt Datum und Uhrzeit an, zu der diese Signatur verfällt. Geben Sie diesen Wert als Zahl der Sekunden seit dem 1. Januar 1970 oder *YYYYMMDDHHmmSS* (GMT) an.
GültigkeitsBeginn	gibt Datum und Uhrzeit an, ab der diese Signatur verwendet werden kann. Format wie beim *GültigkeitsEnde*.
Quersumme	gibt die Quersumme des Schlüssels an, der diese Signatur erstellt hat. Sie kann mit dem Algorithmus aus Listing 7.23 in Abschnitt 7.5.12 berechnet werden.
Signierer	Gibt die Domain an, deren Private-Key zum Erstellen dieser Signatur verwendet wurde. Bitte beachten Sie, dass die Domain den passenden Public-Key als DNSKEY-Record anbieten muss.
Signatur	Geben Sie den Signatur im Base64-Format an. Zur besseren Übersichtlichkeit können Sie ihn mit Leerzeichen, Tabulatorzeichen oder Zeilenumbrüchen in beliebig viele Teile zerlegen. **Achtung:** Wenn Sie den Schlüssel über mehrere Zeilen verteilen, müssen Sie die Zeilenumbrüche mit runden Klammern schützen.

Tabelle 7.15: Erklärung zu den Feldern des RRSIG-Records (Fortsetzung)

Beispiel:

```
host.example.com. 86400 IN RRSIG (
     A               ; Typ
     5               ; Algorithmus
     3               ; Tiefe
```

```
    86400           ; RecordTTL
    20030322173103  ; GültigkeitsEnde
    20030220173103  ; GültigkeitsBeginn
    2642            ; Quersumme
    example.com.    ; Signierer
    oJB1W6WNGv+ldvQ3WDG0MQkg5IEhjRip8WTr ; Signatur
    PYGv07h108dUKGMeDPKijVCHX3DDKdfb+v6o
    B9wfuh3DTJXUAfI/M0zmO/zz8bW0Rznl8O3t
    GNazPwQKkRN20XPXV6nwwfoXmJQbsLNrLfkG
    J5D6fwFm8nN+6pBzeDQfsS3Ap3o= )
```

Listing 7.46: Beispiel zum RRSIG-Record (aus `draft-ietf-dnsext-dnssec-records-08.txt`)

Mehr zum Thema: Der RRSIG-Record wird in Abschnitt 12.3 sowie in `draft-ietf-dnsext-dnssec-records-*.txt` auf `http://www.ietf.org/internet-drafts/` beschrieben.

7.5.36 RT

Typnummer: 21

Status: experimentell

Beschreibung: Ein »Route Through«-Record gibt die Namen von A-, ISDN- oder X25-Records an, unter deren Adressen der angegebene Host besser erreichbar ist.

Das Datenfeld eines RT-Records hat folgende Syntax:

`Preference Router`

- *`Preference`*: eine ganze Zahl zwischen 0 und 65535. Sie bestimmt die Reihenfolge, in der die Anwendung die Router anspricht.
- *`Router`*: der Name eines A-, ISDN- oder X25-Records.

Beispiel:

```
$ORIGIN prime.com.
sh     RT   2     relay.prime.com.
```

```
      RT   10   net.prime.com.
*     RT   90   relay.prime.com.
; Internet-Adressen:
net   A    1.2.3.4
      A    1.2.3.5
; ISDN-Rufnummer +1 (0)508 620 2800 ext 3217 (USA):
relay ISDN 150862028003217
; X25-Adresse mit Data Network Identification Code (DNIC)=3110
relay X25  311061700956
```

Listing 7.47: Beispiel zu RT-Records (teilw. aus RFC1183)

Achtung: Wenn der im Datenfeld angegebene Router nicht mit einem Punkt endet, wird der Nameserver beim Einlesen des Zonefiles den Namen der Zone (`$ORIGIN`) anhängen.

Achtung: Sie dürfen keinen CNAME als Router angeben.

Mehr zum Thema: Siehe RFC1183.

7.5.37 SIG

Typnummer: 24

Status: Standard (DNSSEC)

Beschreibung: Ein SIG-Record enthält die Signaturen für andere Records. Er stimmt syntaktisch mit dem RRSIG-Record überein.

Mehr zum Thema: Die Definition des SIG-Records finden Sie in RFC2535 und Details zum RRSIG-Record in Abschnitt 7.5.35.

7.5.38 SOA

Typnummer: 6

Status: Standard

Beschreibung: Jede Zone enthält genau einen Start-of-Authority-(SOA-)Record. Der SOA-Record nennt

- den primären Nameserver und die E-Mail-Adresse seines Hostmasters
- vier Angaben zum Abgleich mit sekundären Nameservern
- den Timeoutwert für negative Antworten

Das Datenfeld eines SOA-Records besteht also aus sieben Teilen:

Feld	Beschreibung
Primärer Nameserver	Der Name (A-Record!) eines Nameservers, der für diese Zone zuständig ist. Wenn mehrere Nameserver zuständig sind, ist der Master anzugeben. **Achtung:** Wenn der Name des Nameservers nicht mit einem Punkt endet, wird der Nameserver beim Einlesen des Zonefiles den Namen der Zone (`$ORIGIN`) anhängen. **Achtung:** Sie dürfen keinen CNAME als Nameserver angeben.
Hostmaster E-Mail	Die E-Mail-Adresse des Hostmasters. Das Klammeraffe-Zeichen (**@**), das in einer E-Mail-Adresse den Usernamen von der Domain trennt, müssen Sie durch einen Punkt ersetzen. Wenn Sie einen Hostmaster anschreiben möchten, müssen Sie den ersten Punkt, der nicht einem Backslash (\) folgt, durch einen Klammeraffen ersetzen. **Achtung:** Wenn der Username selbst Punkte enthält, müssen Sie diesen jeweils einen Backslash voranstellen. **Achtung:** Wenn die E-Mail-Adresse nicht mit einem Punkt endet, wird der Nameserver beim Einlesen des Zonefiles den Namen der Zone (`$ORIGIN`) anhängen.

Tabelle 7.16: Felder eines SOA-Records

Feld	Beschreibung
	Beispiel: In der Zone `informatik.uni-frankfurt.de.` würde `admin\.dns.rbi` die E-Mail-Adresse `admin.dns@rbi.informatik.uni-frankfurt.de` darstellen. **Empfehlung des RIPE:** `hostmaster` (entsprechend RFC2142)
Versionsnummer	Eine ganze Zahl zwischen -2147483646 und 2147483647. Sekundäre Nameserver beurteilen anhand der Versionsnummer im SOA-Record, ob die Zone seit dem letzten Download geändert wurde. **Empfehlung des RIPE:** Die Versionsnummer im Format `YYYYMMDDnn` notieren, z.B. `2004122401` für die erste Änderung am Heiligabend 2004.
Refresh Timeout	Der Zeitintervall (in Sekunden), in dem sekundäre Nameserver den SOA-Record der Zone vom primären Nameserver anfordern. Wenn ein sekundärer Nameserver dabei feststellt, dass die Versionsnummer größer als zuvor ist, wird er die gesamte Zone erneut vom primären Nameserver downloaden. **Empfehlung des RIPE:** 86400 Stunden (24 Stunden)
Retry Timeout	Der Zeitintervall (in Sekunden), in dem sekundäre Nameserver den SOA-Record der Zone vom primären Nameserver anfordern, wenn die »turnusmäßige« Abfrage (Refresh-Timeout) fehlschlägt. **Empfehlung des RIPE:** 7200 Sekunden (2 Stunden)
Expire Timeout	Der Zeitpunkt (in Sekunden seit dem letzten Kontakt zum primären Nameserver), ab dem ein sekundärer Nameserver eine Zone verwirft. **Empfehlung des RIPE:** 3600000 Sekunden (1000 Stunden)
Negative Caching Timeout	Die Zeit (in Sekunden), nach der ein Nameserver eine NXDOMAIN-Antwort vergessen soll. NXDOMAIN-Antworten teilen mit, dass die angefragte Domain *nicht* existiert. **Empfehlung**: 10800 Sekunden (3 Stunden)

Tabelle 7.16: Felder eines SOA-Records (Fortsetzung)

Mehr zum Thema: Die Empfehlungen des RIPE zu SOA-Records können Sie auf `http://www.ripe.net/ripe/docs/ripe-203.html` nachlesen.

Beispiel:

```
$ORIGIN informatik.uni-frankfurt.de.
@ 38400 IN SOA hermes.rbi hostmaster (
  2004052117  ; serial
  86400       ; refresh
  7200        ; retry
  3600000     ; expire
  10800       ; negcache
)
```

Listing 7.48: Beispiel zu einem SOA-Record

Viele Nameserver erlauben Zeilenumbrüche in Zonefiles, wenn die Bruchstellen mit runden Klammern umgeben sind.

Ein Semikolon bewirkt, dass der Nameserver den Rest der Zeile als *Kommentar* erkennt und ignoriert.

Damit kann man den SOA-Record aus dem letzten Beispiel wie folgt notieren:

```
$ORIGIN informatik.uni-frankfurt.de.
@ 38400 IN SOA (
  hermes.rbi.informatik.uni-frankfurt.de. ; Primärer NS
  dns.informatik.uni-frankfurt.de.        ; Hostmaster-E-Mail
  2004052117                              ; Versionsnummer
  10800                                   ; Refresh-Timeout
  6000                                    ; Retry-Timeout
  3000000                                 ; Expire-Timeout
  86400)                                  ; Negative-Timeout
```

Listing 7.49: Derselbe SOA-Record (übersichtlicher)

Mehr zum Thema: Siehe RFC1035.

7.5.39 SRV

Typnummer: 33

Status: Standard

Beschreibung: Ein »Service«-Record gibt an, welche Server in dieser Domain für das angegebene Protokoll zuständig sind.

Anwendungsbeispiele:

- Windows 2000 lokalisiert die Domaincontroller im Active Directory Service (ADS) über SRV-Records.
- Linux' PAM-LDAP-Modul lokalisiert LDAP-Server über SRV-Records.

Mehr zum Thema: Die Konfiguration eines Nameservers für ADS und LDAP wird in Abschnitt 8.6 beschrieben.

Das Datenfeld des SRV-Records hat folgende Syntax:

```
_Service._Protokoll[.Domain.] [IN] [TTL] SRV (
   Kosten
   Gewicht
   Portnummer
   Ziel)
```

Listing 7.50: Syntax des SRV-Records

Erklärung:

Feld	Beschreibung
_Service	Der Name eines Layer-5-Protokolls (aus der Datei `/etc/services`).
	Achtung: Stellen Sie dem Namen des Service einen Unterstrich (_) voran, um Kollisionen mit evtl. bestehenden Hostnamen zu vermeiden.
	Die Groß/Kleinschreibung dieses Feldes ist unwesentlich. **Beispiel:** `_ldap`

Tabelle 7.17: Felder des SRV-Records

Feld	Beschreibung
_Protokoll	Der Name eines Layer-4-Protokolls. **Achtung:** Stellen Sie dem Namen des Protokolls einen Unterstrich (_) voran, um Kollisionen mit evtl. bestehenden Hostnamen zu vermeiden. Die Groß/Kleinschreibung dieses Feldes ist unwesentlich. Beispiele: `_tcp` oder `_udp`
Domain *TTL*	siehe Abschnitt 7.2.
Kosten	Eine ganze Zahl zwischen 0 und 65535. Sie gibt die Reihenfolge an, in der die Anwendung die Server anspricht. Innerhalb einer *Kosten*-Stufe können Sie mit dem *Gewicht* eine feinere Abstufung treffen.
Gewicht	Eine ganze Zahl zwischen 0 und 65535. Sie erlaubt der Anwendung, die Last auf den Records derselben *Kosten* zu verteilen, wobei der Schwerpunkt bei den Records mit höherem *Gewicht* liegt.
Portnummer	Eine ganze Zahl zwischen 0 und 65535. Sie nennt die Portnummer (im Kontext des *Protokoll*), auf der das *Ziel* den *Service* anbietet.
Ziel	gibt den Namen (A- oder AAAA-Record!) eines Rechners an, der den *Service* auf der angegebenen *Portnummer* im angegebenen *Protokoll* anbietet. Geben Sie »`.`« an, um klarzustellen, dass Ihre Domain diesen Dienst *nicht* anbietet. **Achtung:** Wenn das Ziel nicht mit einem Punkt endet, wird der Nameserver beim Einlesen des Zonefiles den Namen der Zone (`$ORIGIN`) anhängen. **Achtung:** Sie dürfen hier keinen CNAME angeben.

Tabelle 7.17: Felder des SRV-Records (Fortsetzung)

Beispiel:

```
$ORIGIN example.com.
@  IN          SOA server.example.com. root.example.com. (
                   1995032001 3600 3600 604800 86400 )
               NS  server.example.com.
               NS  ns1.ip-provider.net.
               NS  ns2.ip-provider.net.
; foobar - use old-slow-box or new-fast-box if either is
; available, make three quarters of the logins go to
; new-fast-box.
_foobar._tcp    SRV 0 1 9 old-slow-box.example.com.
               SRV 0 3 9 new-fast-box.example.com.
; if neither old-slow-box or new-fast-box is up, switch to
; using the sysadmin's box and the server
               SRV 1 0 9 sysadmins-box.example.com.
               SRV 1 0 9 server.example.com.
server          A   172.30.79.10
old-slow-box    A   172.30.79.11
sysadmins-box   A   172.30.79.12
new-fast-box    A   172.30.79.13
; NO other services are supported
*._tcp          SRV  0 0 0 .
*._udp          SRV  0 0 0 .
```

Listing 7.51: Beispiel zum SRV-Record (aus RFC2782)

Mehr zum Thema: Der SRV-Record wird in RFC2782 definiert. Weitere Beispiele finden Sie in Abschnitt 8.6 bei den Zonefiles für den Active Directory Service (ADS).

7.5.40 SSHFP

Typnummer: 44

Status: experimentell

Beschreibung: Ein »Secure shell fingerprint«-Record speichert einen öffentlichen SSH-Hostkey.

Zukünftige SSH-Clients können mit dieser Information die Identität des SSH-Servers automatisch prüfen, was dem User die manuelle Bestätigung des Hostkey-Fingerprints bei der ersten Verbindung erspart.

Beispiel:

```
host.example.  SSHFP (
      2    ; Der Public-Key-Algorithmmus:     1=RSA, 2=DSS
      1    ; Der Hash-Alg. für Fingerprints:  1=SHA1
      123456789abcdef67890123456789a...   ; Der Public-Key)
```

Listing 7.52: Beispiel zum SSHFP-Record-Typ (aus `draft-ietf-secsh-dns-05.txt`)

BIND beherrscht diesen Record-Typ erst ab Version 9.3.

Mehr zum Thema: Der SSHFP-Record wird in `draft-ietf-secsh-dns-*.txt` auf `http://www.ietf.org/internet-drafts/` beschrieben.

7.5.41 TXT

Typnummer: 16

Status: Standard

Beschreibung: Ein TXT-Record speichert beliebige Texte.

Achtung: Wenn der Text Leerzeichen enthält, müssen Sie ihn in doppelte Anführungszeichen (") setzen.

Anwendungsbeispiele:

- Sie können die Rufnummern von Hostmastern eintragen (siehe Listing 7.44 beim RP-Record, Abschnitt 7.5.34).

- Sie können einen Gateway-Rechner angeben, was für Opportunistic IPSEC erforderlich ist.
  Gegeben sei folgender TXT-Record:

```
$ host -t txt 18.22.106.134.in-addr.arpa
18.22.106.134.in-addr.arpa
    text "X-IPsec-Server(10)=134.106.12.126 "
$
```

Listing 7.53: Beispiel zu Opportunistic IPSEC

  Wenn Sie *sicher* mit dem Rechner `134.106.22.18` kommunizieren möchten, der selbst kein IPSEC beherrscht, sich aber hinter dem Router `134.106.12.126` befindet, der IPSEC kann, können Sie mit den Informationen aus dem DNS einen IPSEC-Tunnel mit `134.106.12.126` aufbauen und darüber indirekt sicher mit `134.106.22.18` kommunizieren. Der Hostmaster versichert also, dass `134.106.12.126` als Stellvertreter für `134.106.22.18` auftreten darf und deren Kommunikation untereinander sicher ist.

Achtung: Da DNS-Abfragen oft über UDP übertragen werden, das keine Fragmentierung der Datagramme zulässt, sollten TXT-Records nicht länger als 512 Byte sein.

Mehr zum Thema: Der TXT-Record wird in RFC1035 definiert. Opportunistic IPSEC wird auf `http://wiki.openswan.org/index.php/OpportunisticEncryption` erklärt.

7.5.42 UNSPEC

Typnummer: 103

Status: experimentell

Beschreibung: Ein »Unspecified«-Record speichert beliebige, binäre Daten. Seine Syntax entspricht der des TXT-Records (siehe Abschnitt 7.5.41).

7.5.43 WKS

Typnummer: 11

Status: experimentell

Beschreibung: Ein »Well Known Service«-Record gibt die Dienste an, die ein Host anbietet.

Das Datenfeld eines WKS-Records hat folgende Syntax:

Adresse Protokoll Service [*Service ...*]

Erklärung:

Feld	Beschreibung
Adresse	nennt die IP-Adresse, unter der der Host die Dienste anbietet.
Protokoll	TCP oder UDP
Service	Eine Liste von Service-Namen (siehe /etc/services-Datei), z.B. smtp

Tabelle 7.18: Felder des WKS-Records

Beispiel:

```
$ORIGIN ARPA.
SRI-NIC.ARPA.  WKS  10.0.0.51  TCP  TELNET FTP SMTP
               WKS  10.0.0.51  UDP  TIME
               WKS  26.0.0.73  TCP  TELNET FTP SMTP
               WKS  26.0.0.73  UDP  TIME
```

Listing 7.54: Beispiel zum WKS-Record

Achtung: Die Portnummer des Service darf nicht größer als 255 sein. Verwenden Sie besser den SRV-Record (siehe Abschnitt 7.5.39), der die Dienste für die ganze Domain angibt und nebenbei eine gewichtete Lastverteilung ermöglicht.

Mehr zum Thema: Siehe RFC1033.

7.5.44 X25

Typnummer: 19

Status: experimentell

Beschreibung: Ein X25-Record gibt eine X25-Adresse an, unter der ein Host erreichbar ist.

Ein Beispiel finden Sie in Listing 7.47 beim RT-Record (Abschnitt 7.5.36).

Mehr zum Thema: Siehe RFC1183.

8 Beispiel-Konfigurationen

Die Beispiele in diesem Kapitel beziehen sich auf BIND 9.3.

8.1 Caching-Only-Nameserver

Ein Caching-Only-Nameserver ermöglicht Resolvern, die nur rekursive Abfragen senden können, den Zugriff auf das DNS, indem er die Abfragen iterativ an die zuständigen Nameserver weiterleitet und das Ergebnis zwischenspeichert (siehe Abschnitt 1.8 und Abbildung 8.1).

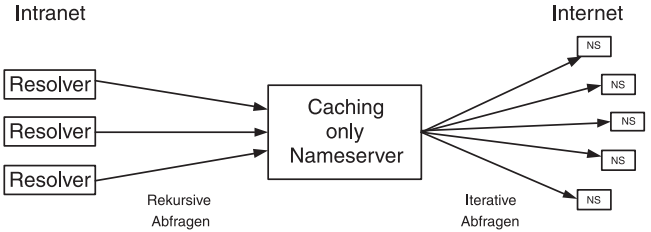

Abbildung 8.1: Caching-Only-Nameserver

Mit den nachstehenden Dateien konfigurieren Sie einen *Caching-Only-Nameserver* mit folgenden Eigenschaften:

- Er kann rekursive Anfragen von Stub-Resolvern beantworten.
- Der `localhost`-Eintrag (`127.0.0.1`) ist ihm bekannt.
- Bei Anfragen zu IP-Adressen in den typischen Broadcast-Netzen (`0.*` und `255.*`) versucht er gar nicht erst, im Internet einen Hostnamen zu finden.
- Er erlaubt die Fernsteuerung über das Admintool `rndc`.

8.1.1 Die Konfigurationsdatei

```
options {
        // Das Verzeichnis, in dem BIND dumps und
        // Statistiken anlegt:
        directory "/var/cache/bind";

        // Die regelmäßige Aktualisierung
        // der root-hints verhindern
        // (bei Dial-on-Demand sehr zu empfehlen):
        dialup yes;
        heartbeat-interval 0;

        // Zugriffskontrolle:
        allow-recursion{     // Wer darf rekursiv abfragen?
            localnets;       //   - alle bis zum Router
        };
};

zone "localhost" {
        type master;
        file "/etc/bind/db.local";
};
zone "127.in-addr.arpa" {
        type master;
        file "/etc/bind/db.127";
};
zone "0.in-addr.arpa" {
        type master;
        file "/etc/bind/db.empty";
};
zone "255.in-addr.arpa" {
        type master;
        file "/etc/bind/db.empty";
};
```

Listing 8.1: /etc/bind/named.conf

Achtung: Unter BIND4 und BIND8 mussten Sie zusätzlich die Root-Zone (.) definieren und Ihr Zonefile per ftp vom InterNIC downloaden. Da die Adressen der Root-Nameserver in BIND9 jedoch einkompiliert sind, können Sie diese Angabe ab BIND9 weglassen. BIND lädt die Root-Zone dann regelmäßig selbst von einem der Root-Nameserver.

8.1.2 Der TSIG-Schlüssel für rndc

Damit Sie den Nameserver mit dem Admintool `rndc` fernsteuern können, müssen Sie einen TSIG-Schlüssel an geeigneter Stelle ablegen:

```
key "rndc-key" {
        algorithm hmac-md5;
        secret "P/25ojDQpsJpMeK6nKvXmQ==";
};
```

Listing 8.2: /etc/bind/rndc.key

Mehr zum Thema: Der Aufruf von `rndc` wird in Abschnitt 11.2 beschrieben.

8.1.3 Die Zonefiles

Hier sind die in der Konfigurationsdatei referenzierten Zonefiles:

```
;
; BIND reverse data file for broadcast zone (0.*, 255.*)
;
$TTL    1w
@ IN SOA   localhost. root.localhost. (
                        1          ; Serial
                        1w         ; Refresh
                        1d         ; Retry
                        4w         ; Expire
                        1w )       ; Negative Cache TTL
   NS      localhost.
```

Listing 8.3: /etc/bind/db.empty

```
;
; BIND reverse data file for local loopback interface
;
$TTL    1w
@ IN   SOA   localhost. root.localhost. (
                        1          ; Serial
                        1w         ; Refresh
                        1d         ; Retry
                        4w         ; Expire
                        1w )       ; Negative Cache TTL
        NS      localhost.
1.0.0   PTR     localhost.
```

Listing 8.4: /etc/bind/db.127

```
;
; BIND data file for local loopback interface
;
$TTL    1w
@ IN   SOA     localhost. root.localhost. (
                        1          ; Serial
                        1w         ; Refresh
                        1d         ; Retry
                        4w         ; Expire
                        1w )       ; Negative Cache TTL
       NS      localhost.
       A       127.0.0.1
```

Listing 8.5: /etc/bind/db.local

8.2 Forwarding einrichten

Einige Internet-Provider betreiben öffentliche Caching-Only-Nameserver und stellen die Adressen dieser Server bei der Interneteinwahl automatisch auf den Rechnern ihrer Kunden ein.

Diese Nameserver sind mit schnellen Leitungen an das Internet angebunden und können viele Anfragen aus dem Cache beantworten. Daher können Sie Ihren Caching-Only-Nameserver beschleunigen, indem Sie ihn so einrichten,

dass er die Abfragen rekursiv an die Caching-Only-Nameserver des Providers weiterleitet.

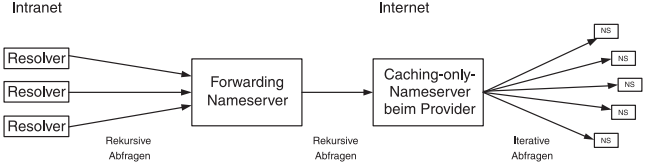

Abbildung 8.2: Forwarding Nameserver

Verwenden Sie hierzu die `forward`- und die `forwarders`-Option:

```
options {
   // ...
   forward first; // notfalls über die root-ns gehen
   forwarders {
       // T-Online Proxyserver (Privatkunden):
       217.237.151.225;   // Dortmund
       # 217.237.150.225; // Hamburg
       # 217.237.149.161; // Hannover
       # 217.237.150.97;  // Köln
       # 217.237.149.225; // Leipzig
       # 217.237.151.97;  // München
       # 217.237.151.161; // Ulm

       // Deutsche Telekom AG (Geschäftskunden):
       194.25.0.69;      // Frankfurt 1
       # 194.25.0.70;    // Frankfurt 2
       # 194.25.0.61;    // Hannover 1
       # 194.25.0.62;    // Hannover 2
       # 194.25.0.53;    // Leipzig 1
       # 194.25.0.54;    // Leipzig 2

       // Arcor:
       145.253.2.11;     // dns1.arcor-ip.de
```

```
        # 145.253.2.75;   // dns2.arcor-ip.de
        # 145.253.2.171;  // dns3.arcor-ip.de
        # 145.253.2.203;  // dns4.arcor-ip.de
        # 145.253.2.139;  // dns5.arcor-ip.de
        # 145.253.2.196;  // dns6.arcor-ip.de
        # 145.253.2.81;   // dns7.arcor-ip.de
        # 145.253.2.174;  // dns8.arcor-ip.de
        # 195.50.140.250; // dns10.arcor-ip.de
    };
};
```

Listing 8.6: Forwarder-Anweisung für /etc/bind/named.conf

Es reicht, wenn Sie zwei oder drei Forwarder auswählen, die sich in Ihrer Nähe befinden.

Wenn BIND keinen dieser Nameserver erreichen kann, bewirkt `forward first`, dass er die herkömmliche Auflösung (über die Root-Nameserver) durchführt.

Wenn Sie die `forward`-Option von `first` auf `only` ändern, spricht BIND in diesem Fall keine weiteren Nameserver an. Das macht Sinn, wenn Ihr BIND hinter einer Firewall steht und Sie den UDP-Zugriff auf wenige Zielrechner begrenzen möchten.

8.3 Primäre Zonen einrichten

Dieser Abschnitt beschreibt, wie Sie ihren Caching-Only-Nameserver zu einem autoritativen Nameservers aufrüsten.

8.3.1 Eine Domain hinzufügen

1. Hängen Sie folgende Zeilen an die Konfigurationsdatei (z.B. `/etc/bind/named.conf`) an:

```
zone "example.com" {
        type master;
        file "/etc/bind/db.example.com.";

        // Optionale Einstellungen:
        notify yes; // falls Sie einen sekundären NS betreiben
```

```
        // Zugriffskontrolle:
        allow-query{any;};   // Wer darf iterativ abfragen?
        allow-transfer{      // Wer darf Zonetransfers?
            192.168.42.1;    //   - primärer Nameserver
            192.168.42.2;    //   - sekundärer Nameserver
        };
};
```

Listing 8.7: Zone-Anweisung für eine Domain /etc/bind/named.conf

Die Zugriffskontrolle können Sie auch in die `options`-Anweisung verschieben. Dann gilt sie für *alle* Zonen.

Achtung: Wenn einer der sekundären Nameserver Ihrer Zone unter Microsoft DNS-Server läuft, sollten Sie diesem keine Nachrichten größer 16 Kbyte zumuten. Schreiben Sie in diesem Fall zusätzlich folgende Zeilen in die Konfigurationsdatei:

```
server IPdesMsDnsServers {
        transfer-format one-answer;
};
```

Listing 8.8: Zonetransfer für Microsoft DNS-Server ermöglichen

2. Dann erstellen Sie ein Zonefile:

```
$TTL 1d
@  IN SOA ns1.example.com. hostmaster (
                        2004081801 ; Serial
                        3h         ; Refresh
                        60m        ; Retry
                        2w         ; Expire
                        1d )       ; Negative Cache TTL
;
; Die zugehörigen IP-Netze (freiwillige Angabe, RFC1101)
```

```
@           PTR    192.168.42.0
            PTR    192.168.43.0
;
; Nameserver dieser Zone:
@           NS     ns1.example.com.
            NS     ns2.example.com.
ns1         A      192.168.42.1
ns2         A      192.168.42.2
;
; Mailserver:
@           MX      10 mailgate1.example.com. ; Primärer
            MX      20 mailgate2.example.com. ; Failover
*           MX      10 mailgate1  ; user@notexist.example.com
mailgate1   A       192.168.42.3
mailgate2   A       192.168.42.4

; Webserver (Round-Robin):
@           A       192.168.42.5  ; http://example.com/
            A       192.168.42.6  ;
www         A       192.168.42.5  ; http://www.example.com/
            A       192.168.42.6
_http._tcp  SRV     0 0 80 www    ; für moderne Browser
ftp         CNAME   www
; ... und so weiter
```

Listing 8.9: /etc/bind/db.example.com.

3. Aktivieren Sie die neue Konfiguration und beobachten Sie dabei das Logfile des Nameservers:

```
$ tail -n +0 -f /var/log/{syslog,messages} |grep named &
$ rndc reload
[...]named: loading configuration from '/etc/bind/named.conf'
[...]named: zone example.com/IN: loaded serial 2004081801
[...]named: zone example.com/IN: sending notifies
$
```

Listing 8.10: Die neue Zone aktivieren

Falls Sie sich in der Konfigurationsdatei oder im Zonefile vertippt hatten, wird BIND eine Warnung in das Logfile schreiben:

```
$ rndc reload
[...]named: loading configuration from '/etc/bind/named.conf'
[...]named: named.conf:72: unknown option 'allow-reKursion'
[...]named: reloading configuration failed: failure
$
```

Listing 8.11: Fehler in der Konfigurationsdatei festgestellt

```
$ rndc reload
[...]named: loading configuration from '/etc/bind/named.conf'
[...]named: dns_master_load: /etc/bind/db.example.com.:31:
            unknown RR type 'CMAME'
[...]named: zone example.com/IN: loading master file
            /etc/bind/db.example.com.: unknown class/type
$
```

Listing 8.12: Fehler im Zonefile festgestellt

8.3.2 Ein IP-Netz hinzufügen

Wenn Sie ein eigenes IP-Netz verwalten und auch IP-Adressen zu Namen auflösen möchten, sollten Sie hierfür eine so genannte *Reverse*-Zone anlegen.

Dazu gehen Sie wie beim Einfügen einer primären Domain vor:

1. Erweitern Sie die Konfigurationsdatei:

```
// Die IP-Adressen des Netzes 192.168.42.*
zone "42.168.192.in-addr.arpa" {
        type master;
        notify yes;
        file "/etc/bind/db.192.168.42";

        // Optionale Einstellungen:
        notify yes; // falls Sie einen sekundären NS betreiben
```

```
        // Zugriffskontrolle:
        allow-query{any;};   // Wer darf iterativ abfragen?
        allow-transfer{      // Wer darf Zonetransfers?
            192.168.42.1;    //   - primärer Nameserver
            192.168.42.2;    //   - sekundärer Nameserver
        };
};
```

Listing 8.13: Zone-Anweisung für ein IP-Netz in `/etc/bind/named.conf`

2. Erstellen Sie ein Zonefile:

```
$TTL 1d
@  IN SOA ns1.example.com. hostmaster.example.com. (
                      2004081801 ; Serial
                      3h         ; Refresh
                      60m        ; Retry
                      2w         ; Expire
                      1d )       ; Negative Cache TTL
;
; Nameserver dieser Zone:
@           NS     ns1.example.com.
            NS     ns2.example.com.
;
; Freiwillige Angaben gemäß RFC1101)
@           PTR    example.com.   ; Domain
            A      255.255.255.0  ; Netmask
;
; Host-Einträge:
; 0         PTR    FREI
1           PTR    ns1.example.com.
2           PTR    ns2.example.com.
3           PTR    mailgate1.example.com.
4           PTR    mailgate2.example.com.
5           PTR    www.example.com.
6           PTR    www.example.com.
```

```
; 7         PTR    FREI
; 8         PTR    FREI
; ...
; 255       PTR    FREI
```

Listing 8.14: /etc/bind/db.192.168.42

3. Aktivieren Sie die neue Konfiguration und beobachten dabei das Logfile des Nameservers, wie in Listing 8.10, Abschnitt 8.3.1 beschrieben.

8.4 Sekundäre Zonen einrichten

Es ist einfach, einen laufenden Nameserver zum sekundären Nameserver einer Zone zu »befördern«:

1. Kopieren Sie die Zone-Anweisung des primären Nameservers (siehe Listing 8.7, Abschnitt 8.3.1) in die Konfigurationsdatei des sekundären Nameservers und ersetzen dabei die Zeile
   `type master;`
   durch
   `type slave; masters{ IP Ihres primären NS;};`
2. Anschließend aktivieren Sie die neue Konfiguration und beobachten dabei das Logfile des Nameservers:

```
$ tail +0f /var/log/{syslog,messages} |grep named &
$ rndc reload
[...]named: loading configuration from '/etc/bind/named.conf'
[...]named: slave zone "example.com" (IN) loaded
            (serial 2004081801)
[...]named: Ready to answer queries.
$
```

   Und schon hat Ihr sekundärer Nameserver die neue Zone per Zonetransfer vom primären Nameserver geladen und ein Zonefile erstellt.

Achtung: Wenn auf dem primären Nameserver der Zonetransfer eingeschränkt ist (`allow-transfer{}`), muss der neue, sekundäre Nameserver dort aufgenommen werden.

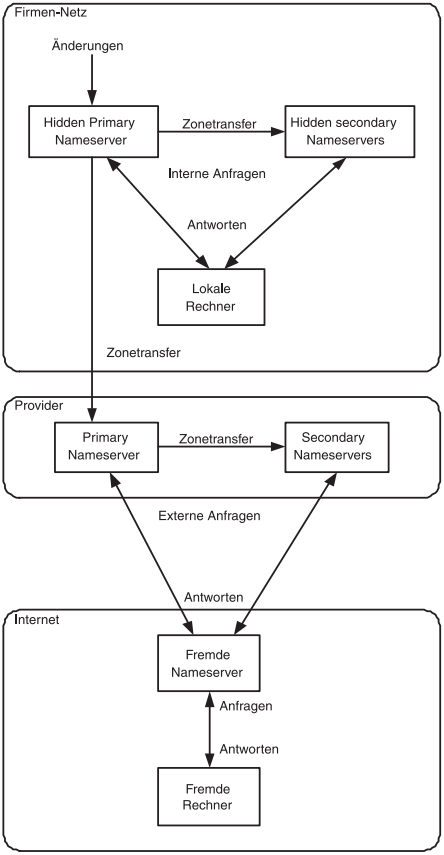

Abbildung 8.3: Hidden-Primary-Architektur

8.5 Die Hidden-Primary-Architektur

Viele Hostmaster möchten

- die Verfügbarkeit ihres Nameservers erhöhen
- die Netzlast reduzieren
- aus Sicherheitsgründen keine DNS-Zugriffe von außerhalb in ihr Netz zulassen

Mit Hilfe der *Hidden-Primary*-Architektur verlagern Sie diese Aufgaben an einen Dienstleister, der den Nameserverdienst (hoffentlich) professionell betreibt (siehe Abbildung 8.3).

Es gibt viele Anbieter auf dem Markt, die eine solche Hidden-Primary-Architektur auf Anfrage anbieten, allerdings werben die wenigsten für diesen Dienst.

Beispiele (Stand: Oktober 2004):

- Die Deutsche Telekom bietet den Nameservice auf `http://t-domain.de/dns` ab 0,80 €/Monat an.
- `http://www.dyndns.org` bietet den Nameservice ab 1,25 $/Monat an.
- `http://www.secondarydns.com` bietet den Nameservice mit 100% Verfügbarkeitsgarantie und Volumenbegrenzung ab 2,50 $/Monat an.

Wenn Sie *mehrere* Domains anmelden, erhalten Sie bis zu 80% Rabatt.

Die folgenden Schritte beschreiben die Konfiguration eines Hidden Primarys am Beispiel der Deutschen Telekom.

8.5.1 Konfiguration

Erlauben Sie dem Provider, die Zonen per Zonetransfer zu beziehen.

```
acl "telekom-diag" {  // Stand: 20.08.2004
   62.156.152.37;  // goofy.dns-oldenburg.de Eingangspruefung
   62.156.152.59;  // miraculix.dns-oldenburg.de
                   // (Pruefung manuell vom Bearbeiter)
   62.156.153.47;  // www.t-domain.de Eingangspruefung
   194.25.1.113;   // Diagnoseautomat fuer Secondaryeinrichtung
                   // (bei DTAG ZID Muenster)
   194.25.15.217;  // Diagnose bei Fehlerbearbeitung allgemein
```

```
   194.246.96/24;  // Diagnose Denic
};
acl "telekom-ns" { // siehe Delegation
   194.25.0/24;    // pns.dtag.de und secondary.dtag.DE
   195.244.245/24; // secondary*.dtag.NET
};
acl "internal" {  // Ihre internen Netze
       192.168.42/8;
       192.168.43/8;
};
acl "internal-ns" { // IP-Adressen Ihrer internen Nameserver
       192.168.42.1;
       192.168.42.2;
};

options {
   // ...
   allow-query{none;}; // muss explizit erlaubt werden
   allow-recursion{internal;};
};

// ...

zone "example.com" {
   type master;
   file "/etc/bind/db.example.com.";
   notify yes;
   also-notify    {192.168.42.2;}; // hidden sec. NS
   allow-query    {internal;    telekom-diag; telekom-ns;};
   allow-transfer {internal-ns; telekom-diag; telekom-ns;};
};
```

Listing 8.15: `/etc/bind/named.conf` (für Hidden Primary)

Achtung: Wenn Ihr Nameserver hinter einer Firewall steht, müssen Sie diese so einstellen, dass der Provider zugreifen kann.

8.5.2 Die Zonefiles

In den SOA- und NS-Records Ihrer Zonefiles dürfen Ihre eigenen Nameserver nicht auftauchen, denn schließlich möchten Sie ja den Zugriff auf diese verhindern. Schreiben Sie stattdessen die Namen der Nameserver des Providers (sofern bekannt) in die Zonefiles:

```
$TTL 1d
@  IN SOA pns.dtag.de. hostmaster (
                      2004081801 ; Serial
                      3h         ; Refresh
                      60m        ; Retry
                      2w         ; Expire
                      1d )       ; Negative Cache TTL
  NS   pns.dtag.de.
  NS   secondary004.dtag.net.
; ...
```

Listing 8.16: Offizielle Nameserver im Zonefile angeben

8.5.3 Anmeldung beim Provider

Sobald Ihre Konfigurationsdatei und die Zonefiles angepasst sind, können Sie Ihren Nameserver bei dem Provider anmelden. Wenn er die Zonefiles per Zonetransfer beziehen kann und sie keine Fehler enthalten, wird er Ihre Zonen auf seinen Nameservern aktivieren.

Falls die Domain schon vorher bestand, müssen Sie Ihren bisherigen Registrar bitten, die NS-Einträge in der Parentzone auf die Nameserver Ihres neuen Providers zu ändern. Dieser Vorgang bewirkt meist eine Vertragskündigung und wird bei `.de`-Domains »Konnektivitäts-Koordination« (KK-Antrag) genannt. Oft sendet der neue Provider Ihnen hierzu ein vorbereitetes Schreiben, das Sie nur noch unterschrieben an ihn zurückfaxen müssen.

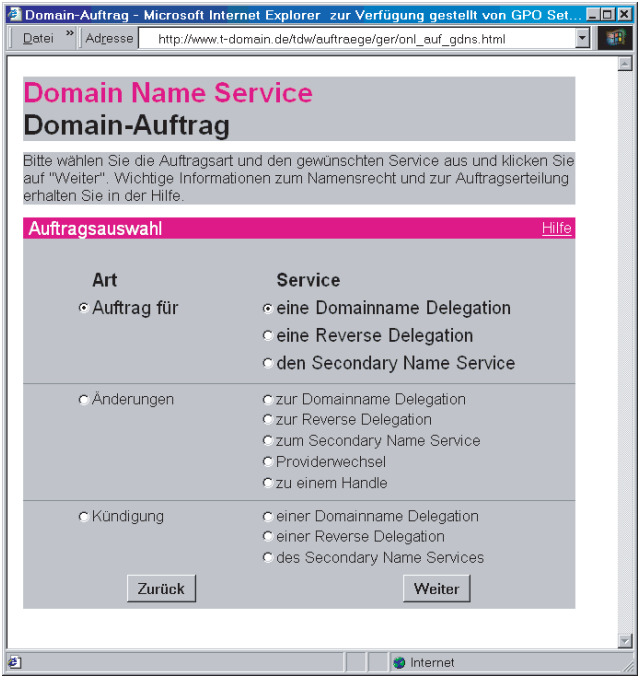

Abbildung 8.4: Hidden-Primary-Auftragsformular

8.6 Microsoft ADS

Windows-2000-Clients benötigen zu ihrer Orientierung im Netz Zugriff auf den Microsoft Active-Directory-Service (ADS).

Der ADS besteht aus folgenden Diensten:

- Domaincontroler (DC) erlauben die Abfrage von Objekten aus einem hierarchischen Namensraum über LDAP.

- Key-Distribution-Center-(KDC-)Server ermöglichen die Authentifizierung über Kerberos-Tickets.

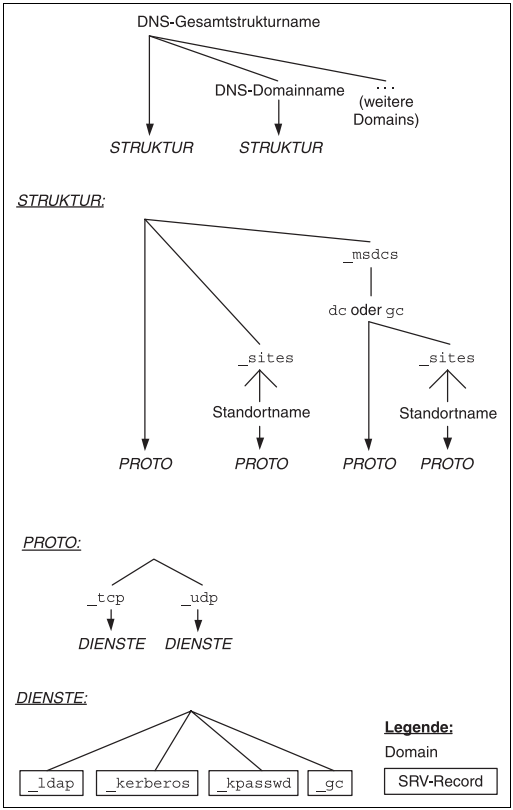

Abbildung 8.5: DNS-Records im ADS

- Global-Katalog-(GC-)Server haben den Überblick über alle Windows-Domänen Ihres Netzes.
- Kpasswd-Server erlauben den Usern, ihre Passwörter zu ändern.

Sie können die Verfügbarkeit dieser Dienste erhöhen, indem Sie sie auf mehrere Server aufteilen.

Die Clients ermitteln die Adressen der zuständigen Server über SRV-Records im DNS! Hierzu muss Ihr Nameserver eine Zone mit dem Namen Ihrer Windows-Gesamtstruktur (z.B. `local`) führen, die folgende Subdomains enthält:

- Eine Subdomain für jede Ihrer Windows-Domänen (z.B. `sales.local.`, `training.local.`)
- `_tcp`, `_udp`, `_sites` und `_msdcs` : Sie enthalten die Server des globalen Katalogs. Der Anmeldedienst auf den Domaincontrollern wird versuchen, diese Domains über Dynamic Updates zusätzlich unterhalb Ihrer eigenen Domänennamen anzulegen.

Wenn Sie den Nameserver für das ADS mit BIND betreiben möchten, führen Sie bitte folgende Schritte durch:

1. Erweitern Sie die BIND-Konfigurationsdatei um folgende Einträge:

```
acl "windowsdc" { IP-AdressenIhrerDomainController;};

zone "WindowsGesamtStruktur" {
   // kann Ihrer internen DNS-Domain entsprechen
   // (z.B. local.).
   // enthält Records der Subdomains _tcp und _udp
   type master;
   file "/etc/bind/db.WindowsGesamtStruktur";
};
zone "_sites.WindowsGesamtStruktur" {
   type master;
   file "/etc/bind/db._sites.WindowsGesamtStruktur ";
   // verwaltet sich über Dynamic Updates
   allow-update{windowsdc;};
};
zone "_msdcs. WindowsGesamtStruktur" {
```

```
   type master;
   file "/etc/bind/db._msdcs.WindowsGesamtStruktur ";
   // verwaltet sich über Dynamic Updates
   allow-update{windowsdc;};
};

// Die folgenden Zonen für jede WindowsDomäne
// anlegen (z.B. sales.local., training.local.)
zone "WindowsDomäne.WindowsGesamtStruktur" {
   type master;
   file "/etc/bind/db.WindowsDomäne.WindowsGesamtStruktur ";
   // verwaltet sich über Dynamic Updates
   allow-update{windowsdc;};
};
```

Listing 8.17: Konfigurationseinträge für ADS

2. Legen Sie ein Zonefile für Ihre Windows-Gesamtstruktur an:

```
$TTL 15m
@ IN             SOA             ...
                 NS              Nameserver.

; Subdomains:
_msdcs           NS              Nameserver.
_sites           NS              Nameserver.
WindowsDomäne1   NS              Nameserver. ; z.B. sales

; Domaincontroller der WindowsGesamtStruktur:
; _udp:
_kpasswd._udp    SRV 0 100  464  DomainController.
_kerberos._udp   SRV 0 100  88   DomainController.
_ldap._udp       SRV 0 100  389  DomainController.
_gc._udp         SRV 0 100  3268 DomainController.
; _tcp:
_kpasswd._tcp    SRV 0 100  464  DomainController.
```

```
_kerberos._tcp   SRV 0 100  88   DomainController.
_ldap._tcp       SRV 0 100  389  DomainController.
_gc._tcp         SRV 0 100  3268 DomainController.

; weitere Records
; (wenn Sie diese Zone auch außerhalb des ADS nutzen)
; [...]
```

Listing 8.18: /etc/bind/db.WindowsDomäne

Wenn Sie mehrere Nameserver oder DomainController haben, sollten Sie die gezeigten Records für jeden davon duplizieren.

Achtung: Stellen Sie sicher, dass alle Clients die A-Records der angegebenen *DomainController* und *Nameserver* abfragen können.

Die Installationsroutine von ADS schreibt alle erforderlichen Einträge in die Datei `%SystemRoot%\System32\Config\Netlogon.dns`.

3. Stellen Sie leere Zonefiles (z.B. aus Listing 8.3, Abschnitt 8.1.3) für die Zonen `_sites`, `_msdcs` und alle Windows-Domänen bereit. Die DomainController werden diese Zonefiles über Dynamic Updates füllen.
4. Aktivieren Sie die neue Konfiguration und beobachten Sie dabei das Syslog, wie in Listing 8.10, Abschnitt 8.3.1 beschrieben.

9 Tricks

In diesem Kapitel werden fortgeschrittene Konfigurationen anhand von Beispielen erläutert.

9.1 Classless in-addr.arpa delegation

Wenn Sie von einem Internetprovider einen Internetanschluss zur Verfügung gestellt bekommen, erhalten Sie neben der Hardware (Leitung, Modem, Router) einige IP-Adressen, die der Provider an Ihren Anschluss routet.

Früher hatten die Provider ihren Kunden oft großzügig ganze Klasse-C-Netze zugewiesen (z.B. `192.0.2.0/24`), die 256 IP-Adressen enthielten (`192.0.2.0` bis `192.0.2.255`). Allerdings waren davon nur 254 IP-Adressen für Hosts nutzbar, da zwei Adressen immer eine besondere Bedeutung haben: Die erste (alle Host-Bits 0) dient als »Network Address«, die letzte (alle Host-Bits 1) als »Broadcast Address« zum gleichzeitigen Adressieren aller Rechner des Teilnetzes.

Heute erhalten Sie u.a. aufgrund von Adressknappheit meist nur ein kleines Subnet mit z.B. 16 IP-Adressen.

> **Mehr zum Thema:** Gründe für die sparsame Verteilung von IP-Adressen finden Sie bei den *Aufgaben der ICANN* in Anhang B.

Die Größe eines Subnets ergibt sich aus der Netmask. Die Zahl hinter dem Schrägstrich gibt an, wie viele Bits der 32-bittigen IPv4-Adresse zur Identifikation des Netzes dienen; die restlichen 32-x Bits dienen dann zum Durchnummerieren der Rechner innerhalb des Subnets selbst. Zum Beispiel enthält das Netz `192.0.2.128/26` die 64 IP-Adressen von `192.0.2.128` bis `192.0.2.191`, wovon die Adresse `192.0.2.128` als Network Address und `192.0.2.191` als Broadcast Address eingesetzt wird.

Die nachstehende Tabelle 9.1 zeigt, wie viele Rechner in den verschiedenen Subnets enthalten sein können:

Netmask (CIDR-Notation)	Netmask (Dotted-Quad-Notation)	IP-Adressen	Rechner
/24	255.255.255.0	256	254
/25	255.255.255.128	128	126
/26	255.255.255.192	64	62
/27	255.255.255.224	32	30
/28	255.255.255.240	16	14
/29	255.255.255.248	8	6
/30	255.255.255.252	4	2
/31	255.255.255.254	2	0 (als Transfernetz bei P2P-Verbindungen)

Tabelle 9.1: Größen einiger Subnets

Die variable Netzgröße führt jedoch bei den PTR-Records für diese IP-Adressen im DNS zu einem Problem: Weil die Zone (z.B. `2.0.129.in-addr.arpa.`) die Einträge mehrerer Kunden enthält, wird der Provider Ihnen oder einem anderen Kunden nicht die gesamte Zone delegieren. Stattdessen könnte er die Einträge natürlich selbst pflegen:

```
$ORIGIN 2.0.192.in-addr.arpa.
$TTL ...
@ IN  SOA ...
      NS  ...
; ...

;
; subnet: 192.0.2.0/25 (192.0.2.0-192.0.2.127):
; Kunde : A
1               PTR     host1.example.com.
2               PTR     host2.example.com.
```

```
3               PTR     host3.example.com.
;
; subnet: 192.0.2.128/26 (192.0.2.128-192.0.2.191):
; Kunde : B
129             PTR     host1.B.domain.
130             PTR     host2.B.domain.
131             PTR     host3.B.domain.
;
; subnet: 192.0.2.192/26 (192.0.2.192-192.0.2.255):
; Kunde : C
193             PTR     host1.C.domain.
194             PTR     host2.C.domain.
195             PTR     host3.C.domain.
```

Listing 9.1: PTR-Records über Subnets

Wenn der Provider sich aber nicht mit Ihren PTR-Records selber befassen möchte, kann er die in RFC2317 beschriebene *Classless in-addr.arpa delegation* anwenden. Dabei erstellt er für jede IP-Adresse Ihres Subnets einen CNAME-Eintrag, der in eine Zone verweist, die Sie selbst bearbeiten können.

Diese Zone liegt normalerweise entweder

- unterhalb der Zone des übergeordneten Netzes, z.B. in der Subdomain `subnet128/26` und heißt dann `subnet128/26.2.0.192.in-addr.arpa.`. Dann muss der Provider sie mit NS-Records an Ihre Nameserver delegieren.

oder

- außerhalb der TLD `arpa.` (z.B. `reverse.example.com.`).

Beispiel:

```
$ORIGIN 2.0.192.in-addr.arpa.
$TTL ...
@ IN  SOA ...
      NS  ...
; ...

; subnet: 192.0.2.0/25 (192.0.2.0-192.0.2.127):
```

```
; Kunde:  A
$GENERATE 0-127       $ CNAME $.reverse.example.com.

; subnet: 192.0.2.128/26 (192.0.2.128-192.0.2.191):
; Kunde : B
subnet128/26            NS    pns.dtag.de.
$GENERATE 128-191     $ CNAME $.subnet128/26

; subnet: 192.0.2.192/26 (192.0.2.192-192.0.2.255):
; Kunde : C
subnet192/26            NS    pns.dtag.de.
$GENERATE 192-255     $ CNAME $.subnet192/26
```

Listing 9.2: Classless in-addr.arpa delegation

Mehr zum Thema: Die `$GENERATE`-Anweisung wird in Abschnitt 7.3.4 beschrieben.

In diesem Fall hat der Provider die Subdomains der beiden /26er-Netze `subnet128/26` und `subnet192/26` genannt. Er kann jedoch beliebige 8-Bit-Zeichenketten (z.B. `128-191`, `128:26`, `part2`, `kunde_B`, `[192.0.2.128;192.0.2.191]`) einsetzen, die bis zu 63 Zeichen lang sind.

Die Kunden tragen ihre PTR-Records in ihre Zonen ein:

```
$ORIGIN reverse.example.com.
$TTL ...
@ IN  SOA ...
      NS  ...
; ...
; 0   network
1     PTR     host1.example.com.
2     PTR     host2.example.com.
3     PTR     host3.example.com.
; 4   unused
```

```
; 5   unused
; ...
; 127 broadcast
```

Listing 9.3: Zonefile von `reverse.example.com`.

```
$ORIGIN subnet128/26.2.0.192.in-addr.arpa.
; ...
129   PTR     host1.B.domain.
130   PTR     host2.B.domain.
131   PTR     host3.B.domain.
```

Listing 9.4: Zonefile von `subnet128/26.2.0.192.in-addr.arpa`.

```
$ORIGIN subnet192/26.2.0.192.in-addr.arpa.
; ...
193   PTR     host1.C.domain.
194   PTR     host2.C.domain.
195   PTR     host3.C.domain.
```

Listing 9.5: Zonefile von `subnet192/26.2.0.192.in-addr.arpa`.

Mehr zum Thema: Siehe RFC2317.

9.2 Lastverteilung und Failover

9.2.1 Motivation

Wenn Sie einen Dienst auch bei einem Ausfall oder Überlastung des zuständigen Servers weiter anbieten möchten, müssen Sie

1. den Dienst auf weiteren Servern installieren (idealerweise auf anderer Hardware an anderen Orten) und
2. die Clients dazu bringen, die Verbindung
   - bei Ausfall eines Servers (*Failover*) oder
   - regelmäßig (*Lastverteilung*)

   zu einem der anderen Server aufzunehmen. Das DNS besitzt verschiedene Hilfsmittel, die dies für verschiedene Dienste ermöglichen, so dass Sie sich

in vielen Fällen die aufwendige Installation eines High-Availability-(HA-)-Clusters sparen können.

9.2.2 E-Mail

Wenn es mehrere Server gibt, die E-Mails für Ihre Domain entgegennehmen, geben Sie diese einfach in Form mehrerer MX-Records für den Namen Ihrer Domain an.

Ein Mail Transfer Agent (MTA) kann E-Mails entweder

- an einen MTA der gleichen oder einer anderen Domain *befördern* (»to relay«), dessen MX-Record geringere *Kosten* (die Zahl hinter MX) hat als er selbst,
- an einen fest eingestellten MTA (»Smarthost«) *weiterleiten* (»to forward«) oder
- dem Postfach des Empfängers *zustellen* (»to deliver«), so dass Mail-User-Agents (MUAs) wie z.B. Mutt, Notes, Outlook oder ein Webmailsystem darauf zugreifen können.

Sie können auch alle diese Möglichkeiten innerhalb einer Domain kombinieren.

Achtung: Geben Sie den MX-Records der befördernden MTAs höhere *Kosten*-Werte als allen anderen MX-Records, um Schleifen zu vermeiden.

Wenn Sie mehrere Mailserver mit den*selben* Kosten angeben, soll der MTA (laut RFC1123) alle Mails gleichmäßig auf diese verteilen.

Beispiel:

Die MX-Records für den Aufbau gemäß Abbildung 9.1 lauten:

```
$ORIGIN example.com.
; [...]
@           MX   10 primary
            MX   20 secondary
            MX   40 external.provider.com.
```

```
; [...]
primary     A    1.2.3.4
secondary   A    1.2.3.5
```

Listing 9.6: Failover mit MX-Records

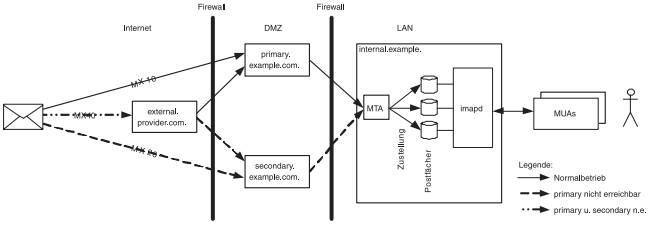

Abbildung 9.1: Typischer MX-Aufbau

Mehr zum Thema: Der MX-Record wird in Abschnitt 7.5.25 beschrieben. Eine genaue Beschreibung aller DNS-Abfragen für SMTP finden Sie in RFC2821.

9.2.3 DNS

Wenn es mehrere Server gibt, die DNS-Anfragen zu Ihrer Zone autoritativ beantworten können, geben Sie diese einfach in Form mehrerer NS-Records für den Namen Ihrer Zone an.

Dann werden Clients (andere Nameserver oder Resolver) alle Nameserver der Zone reihum ansprechen und sich anschließend auf den Nameserver konzentrieren, der am schnellsten geantwortet hatte. Dabei spielt es keine Rolle, ob diese Zone auf den Nameservern als Typ `master` oder `slave` konfiguriert ist.

Mehr zum Thema: Der NS-Record wird in Abschnitt 7.5.27 beschrieben.

Beispiel:

```
$ORIGIN example.com.
; [...]
@           NS   primary
            NS   secondary
            NS   external.provider.com.
; [...]
primary     A    1.2.3.4
secondary   A    1.2.3.5
; Glue-Records in der uebergeordneten Zone nicht vergessen
; (siehe Abschnitt 1.7)
```

Listing 9.7: Antwortzeit-Optimierung mit NS-Records

9.2.4 LDAP-Server

Windows-Clients und die PAM-LDAP-Library nutzen die SRV-Records ihrer DNS-Domain bei der Suche nach LDAP-Servern.

Zur Laststeuerung enthält der SRV-Record Felder für *Kosten* und *Gewichtung*: Server mit niedrigeren *Kosten* werden zuerst versucht zu kontaktieren, bevor Server mit höheren Kosten versucht werden. Die *Gewichtung* gestattet eine weitere Priorisierung bei Servern gleicher Kosten.

Mehr zum Thema: Der SRV-Record wird in Abschnitt 7.5.39 beschrieben.

Beispiel:

```
$ORIGIN example.com.
_ldap._tcp       SRV 0 66  389  primary
                 SRV 0 33  389  secondary
                 SRV 1 100 389  failover
```

Listing 9.8: Lastverteilung und Failover mit SRV-Records

Dieses Beispiel nennt drei LDAP-Server für die Domain `example.com`.. Die Last verteilt sich auf `primary` und `secondary`, wobei zwei Drittel aller LDAP-Abfra-

gen an `primary` gehen. Wenn ein Client weder `primary` noch `secondary` erreichen kann, wendet er sich an `failover`.

9.2.5 Alle weiteren Dienste

Wenn Sie zu einem Namen *mehrere* A- oder AAAA-Records angeben, wird BIND die Reihenfolge der Antworten bei jeder Abfrage rotieren (»Round Robin«).

Da viele Anwendungen nur den ersten Record der Antwortliste beachten und dieser durch die Rotation dauernd wechselt, bewirkt dies eine einfache Lastverteilung. Generell ist hier kein Failover möglich, auch wenn alle RRs durchgespielt werden, da die IP-Adressen ausgefallener Rechner weiterhin als Antwort geliefert werden.

Beispiel:

```
$ORIGIN ibm.com.
www         A   129.42.16.99
            A   129.42.17.99
            A   129.42.18.99
```

Listing 9.9: Mehrere A-Records für einen Namen

9.3 BIND-Version und Hostname eines Nameservers abfragen

Sie können *von außen* feststellen, mit welcher Softwareversion ein BIND-kompatibler Nameserver läuft, da BIND seine Versionsbezeichnung und einige andere Informationen als TXT-Record unter dem Namen `version.bind.` in der Klasse `CH` (CHAOSnet) ablegt:

```
$ dig +noall +answer @localhost ch txt \
    version.bind hostname.bind authors.bind
version.bind.    0    CH    TXT     "9.3.0"
hostname.bind.   0    CH    TXT     "terra"
authors.bind.    0    CH    TXT     "Bob Halley"
authors.bind.    0    CH    TXT     "David Lawrence"
authors.bind.    0    CH    TXT     "Danny Mayer"
```

```
authors.bind.    0    CH    TXT     "Damien Neil"
authors.bind.    0    CH    TXT     "Matt Nelson"
authors.bind.    0    CH    TXT     "Michael Sawyer"
authors.bind.    0    CH    TXT     "Brian Wellington"
authors.bind.    0    CH    TXT     "Mark Andrews"
authors.bind.    0    CH    TXT     "James Brister"
authors.bind.    0    CH    TXT     "Ben Cottrell"
authors.bind.    0    CH    TXT     "Michael Graff"
authors.bind.    0    CH    TXT     "Andreas Gustafsson"
$
$ dig @pns.dtag.de version.bind. txt CHAOS|grep -i "^V"
version.bind.           0       CH      TXT     "9.2.1"
$
```

Listing 9.10: Mit `dig` *die BIND-Version abfragen*

Das geht natürlich auch mit `bash` und `netcat`:

```
$ whatdns() {
echo 'begin-base64 644 -
p8IBAAABAAAAAAAAB3ZlcnNpb24EYmluZAAAAEAADCg==
====' | uudecode| nc -uw 1 $1 53 | strings | tail -n 1; }
$
$ whatdns pns.dtag.de
9.2.1
$ whatdns 141.2.1.1
4.9.7
$ whatdns ns.jfranken.de
Surely you must be joking
$
```

Listing 9.11: Bash-Funktion zur Abfrage der BIND-Version

Ab BIND Version 8 können Sie den Wert des `version.bind`-Records mit der `version`-Option einstellen:

```
options {
   // ...
```

```
    version "AmokDNS 47.11";
};
```

Ab BIND Version 9.3 können Sie das Verhalten zusätzlich mit den Konfigurationsoptionen `hostname` und `system-id` sowie dem folgenden View steuern:

```
view "_bind" chaos {
    recursion no;
    zone "version.bind" chaos {
        type master;
        database "_builtin version";  // mit "version" setzen
    };
    zone "hostname.bind" chaos {
        type master;
        database "_builtin hostname"; //mit "hostname" setzen
    };
    zone "authors.bind" chaos {
        type master;
        database "_builtin authors";    // fest einkompiliert
    };
    zone "id.server" chaos {
        type master;
        database "_builtin id";     // mit "server-id" setzen
    };
};
```

Listing 9.12: `_bind`-View von BIND 9.3.0

Sie können die Definitionen dieser Zonen um eigene Zonefiles, ACLs und Zonennamen erweitert in Ihre eigene Konfigurationsdatei übernehmen.

9.4 Identische Zonen einrichten

Sie können ein und dasselbe Zonefile für mehrere Zonen verwenden, die sich inhaltlich nicht unterscheiden.

Geben Sie hierzu in den `zone{}`-Anweisungen der BIND-Konfigurationsdatei jedes Mal denselben Dateinamen an:

```
zone "example.com" IN {
    type master;
    file "example.ALL";
    // ...
};
zone "example.info" IN {
    type master;
    file "example.ALL";
    // ...
};
zone "example.de" IN {
    type master;
    file "example.ALL";
    // ...
};
```

Listing 9.13: Ein Zonefile für mehrere Zonen verwenden

und verallgemeinern Sie das Zonefile, indem Sie die `$ORIGIN`-Anweisung entfernen, Domains weglassen und `@` einsetzen:

```
$TTL 10m
@ IN SOA ns hostmaster (
             2004072700
             1h 30m 1w
             10m)
     NS  ns
     NS  ns.provider.com.
     MX  10 mx
     MX  20 mx.provider.com.
     A   192.0.2.10
www  A   192.0.2.10
ftp  A   192.0.2.10
mx   A   192.0.2.11
ns   A   192.0.2.12
```

Listing 9.14: Generisches Zonefile

Achtung: Erlauben Sie für diese Zonen keine Dynamic Updates, da diese zu doppelten Einträgen in den Zonefiles und Journalfiles führen können.

9.5 Die SOA-Versionsnummer zurücksetzen

Wenn Sie die Versionsnummer eines SOA-Records versehentlich zu weit erhöht hatten und anschließend auf einen kleineren Wert zurücksetzen, werden Sie auf ein Problem stoßen: Die sekundären Nameserver, die die Zone in der Zwischenzeit bezogen hatten, werden keine Zonetransfers mehr anfordern, da sie der Meinung sind, eine aktuellere Version der Zone zu besitzen.

Es gibt jedoch eine Möglichkeit, die Versionsnummer *ordentlich* auf einen kleineren Wert zu setzen:

Vorgehensweise:

1. Ändern Sie die zu hohe Versionsnummer n auf $(n+2^{31}-1) \bmod 2^{32}$.
1. Warten Sie, bis alle sekundären Nameserver die Zone übertragen haben.
2. Ändern Sie die Versionsnummer auf den gewünschten Wert.

Achtung: Vergessen Sie nicht, nach jedem Speichern des Zonefiles `rndc reload` aufzurufen.

Erklärung:

SOA-Versionsnummern unterliegen der »Seriennummer-Arithmetik«, die in RFC1982 beschrieben ist:

- Eine Versionsnummer ist eine ganze Zahl zwischen 0 und 4.294.967.295 ($=2^{32}-1$).
  Beispiel: 2.005.122.401, was auf die erste Änderung am Heiligabend 2005 schließen lässt.
- Man kann eine Versionsnummer um 1 erhöhen. Wenn sie dabei 4.294.967.296 ($=2^{32}$) erreicht, wird sie auf 0 zurückgesetzt (Überlauf, modulo 2^{32}):
  $4.294.967.295+1 =_{SN} 0$
  $3.294.967.297+1.000.000.000 =_{SN} 1$

Das Verfahren ist dasselbe wie beim großen Zeiger einer Uhr, der nach der Minute 59 auf die Minute 0 zurückspringt ($59+1 =_{Uhr} 0$).

- Beim Vergleich zweier Versionsnummern ist zu beachten, dass eine Versionsnummer per Definition nur »größer« ist als alle 2.147.483.647 ($=2^{31}-1$) Versionsnummern, die ihr direkt vorangingen (Überlauf beachten!):
  $1 >_{SN} 0$
  $1 >_{SN} 3.294.967.297$
  $2.147.483.647 >_{SN} 1$
  $0 >_{SN} 2.147.483.648$
  Damit ist die Hälfte aller Versionsnummern größer und die andere Hälfte kleiner als jede gegebene Versionsnummer.

> **Achtung:** Unter der Seriennummer-Arithmetik ist 2^{31} »größer« als 1, aber *nicht* »größer« als 0.

Nachdem Sie die Versionsnummer um $2^{31}-1$ weitergezählt haben, ist sichergestellt, dass jede weitere Erhöhung um weniger als 2^{31} (und in diesem Bereich liegt dann der gewünschte Wert) die Zahl ebenfalls ver»größer«t.

> **Mehr zum Thema:** Die Seriennummer-Arithmetik wird in RFC1982 beschrieben.

9.6 Realtime-DNS

Es gibt Anwendungen des DNS, in denen sich Ihre Zonen so schnell ändern, dass der Cache von Resolvern und Nameservern zum Problem wird. Ein Beispiel ist das in Abschnitt 9.2.5 beschriebene Failover.

Folgende Möglichkeiten bieten sich an:

- Sie können das Zwischenspeichern eines Records verhindern, indem Sie seine TTL auf 0 setzen.

- Sie können das Zwischenspeichern der Nichtexistenz von Records einer Zone verhindern, indem Sie den »Negative Caching Timeout« der Zone (den letzten Wert des SOA-Records) auf 0 setzen.

Beachten Sie jedoch, dass dann vor jedem Verbindungsaufbau eine Namensauflösung erfolgt, was die Netze und Nameserver stark belasten kann.

9.7 Subdomains innerhalb einer Zone

Wie in Abschnitt 1.5 beschrieben, kann eine Zone mehrere Subdomains enthalten. Schreiben Sie die Records einfach mitsamt ihrer Subdomain in das Zonefile:

```
$ORIGIN informatik.uni-frankfurt.de.
; ...
www      CNAME  sphinx.rbi
www.cad  CNAME  bounty.em
cad      MX     10 hera.rbi
```

Listing 9.15: Subdomain cad

In diesem Beispiel werden die Subdomain `cad.informatik.uni-frankfurt.de` und der Rechner `www.cad.informatik.uni-frankfurt.de` angelegt. Mails an `user@cad.informatik.uni-frankfurt.de` landen auf dem Mailserver `hera.rbi.informatik.uni-frankfurt.de.`.

Wenn Sie wünschen, dass BIND auf die Abfrage fehlender Records und Subdomains einen Standardeintrag sendet, können Sie einen *Wildcard*-Record einsetzen.

Achtung: Wildcard-Records erzeugen mehr Probleme, als sie lösen, wenn man sie falsch einsetzt.

Mehr zum Thema: Wildcard-Records werden in Abschnitt 7.3.5 beschrieben.

9.8 Netze benennen

Sie können

- in Ihren Forward-Zonen die *Network address* und
- in Ihren Reverse-Zonen den *Netznamen* und die *Netmask*

angeben.

Diese Angaben werden derzeit noch nirgends benötigt, gelten jedoch unter Hostmastern als Qualitätsmerkmal.

Beispiel:

```
$ORIGIN example.com.
; ...
; Die zugehörigen IP-Netze (freiwillige Angabe, RFC1101)
@     PTR    192.168.42.0
      PTR    192.168.43.0
; ...
```

Listing 9.16: Netzadresse in der Forward-Zone

```
$ORIGIN 42.168.192.in-addr.arpa.
; ...
; Freiwillige Angaben gemaess RFC1101)
@     PTR    example.com.   ; Domain
      A      255.255.255.0  ; Netmask
```

Listing 9.17: Netzname und Netmask in der Reverse-Zone

Mehr zum Thema: Paul Mockapetris beschreibt diese Records in RFC1101.

Teil III

Nameserver-Administration

10 Installation und Update

10.1 Unterstützte Betriebssysteme

BIND9 läuft auf folgenden Betriebssystemen:

- GNU/Linux (Red Hat, SuSE, Debian, Slackware, Mandrake, ...)
- BSD (OpenBSD, NetBSD, FreeBSD, OS4)
- Solaris
- Mac OS X 10.1
- Weitere Unix-Derivate (OpenUnix, UnixWare, AIX, Tru64, HP-UX, Irix)
- Windows NT, 2000, XP, 2003

Mehr zum Thema: Eine aktuelle Übersicht der unterstützten Betriebssysteme finden Sie auf `http://www.isc.org/sw/bind/bind9.php`.

10.2 BIND9 installieren

10.2.1 Paketierte Binaries

Paketierte Binaries sind sehr leicht zu installieren, da alle erforderlichen Verzeichnisse, User und Konfigurationsdateien automatisch angelegt werden.

- **Für Linux und BSD**: Pakete sind im Lieferumfang Ihres Betriebssystems enthalten und können von den Mirrors des Betriebssystemherstellers heruntergeladen werden.
- **Für Solaris**: Solaris-Packages finden Sie auf `http://www.sunfreeware.com/`.
- **Für Windows**: Eine Binärdistribution inkl. Installationsprogramm finden Sie auf `ftp://ftp.isc.org/isc/bind/contrib/ntbind-9.3.0/BIND9.3.0.zip`.

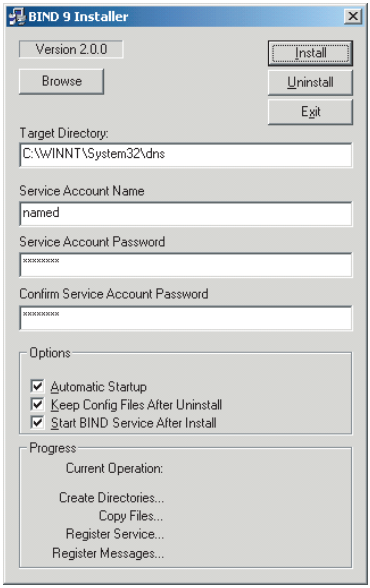

Abbildung 10.1: `BINDInstall.exe` unter Windows

10.2.2 Selbst kompilieren

Sie sollten BIND selbst kompilieren, wenn Sie

- BIND auf einem Betriebssystem einsetzen möchten, für das keine paketierten Binaries angeboten werden
- BIND in einer Version einsetzen möchten, die Ihr Betriebssystemhersteller (noch) nicht als paketiertes Binary anbietet
- der Meinung sind, optimierte Binaries erstellen zu können, die auf Ihrer Hardware performanter laufen als die paketierten Binaries
- den Source-Code an Ihre Anforderungen anpassen möchten

Zur Installation von BIND 9.3 auf einem Linux-System sind folgende Schritte erforderlich:

	Eingabe	Beschreibung
1	`apt-get install openssl`	Wenn Sie DNSSEC einsetzen möchten, installieren Sie die OpenSSL-Libraries.
2	`mkdir bind; cd bind`	Wechseln Sie in ein leeres Verzeichnis.
3	`wget ftp://ftp.isc.org/isc/bind9/9.3.0/bind-9.3.0.tar.gz`	Laden Sie den Source-Code vom FTP-Server des ISC herunter.
4	`tar xzvf bind-9.3.0.tar.gz`	Entpacken Sie den Source-Code.
5	`cd bind-9.3.0`	Wechseln Sie in das dabei entstandene Verzeichnis.
6	`./configure --help`	Entscheiden Sie, welche einstellbaren Parameter für Sie in Frage kommen.
7	`./configure --without-openssl --with-randomdev=/dev/urandom`	Erstellen Sie die Makefiles, indem Sie `configure` mit den Parametern aufrufen, für die Sie sich entschieden haben.
8	`make`	Erstellen Sie die Binaries.
9	`su -`	Werden Sie Root.
10	`cd ..../bind-9.3.0`	Wechseln Sie zurück in das Source-Verzeichnis.
11	`make install`	Kopieren Sie die Manpages und die erstellten Binaries nach `/usr/local/{bin,sbin,lib,man}`.
12	`mkdir /etc/bind`	Jetzt müssen Sie sich entscheiden, ob Sie BIND in einer Changeroot-Umgebung betreiben möchten (siehe Abschnitt 12.5.3). Legen Sie Verzeichnisse für Ihre Zonefiles, Konfigurationsdateien, PID- und Dump-Files an.

Tabelle 10.1: BIND selbst kompilieren

	Eingabe	Beschreibung
13	`vi /etc/bind/named.conf`	Erstellen Sie eine Konfigurationsdatei, die mindestens `options{directory "/etc/bind";};` enthält. Die Konfiguration von BIND wird in Kapitel 6 beschrieben. Beispiele finden Sie in Kapitel 8.
14	`rndc-confgen -a`	Erstellen Sie die Datei `rndc.key`, die den TSIG-Schlüssel zur Authentifizierung des Admintools `rndc` beim Nameserver enthält.
15	`echo nameserver 127.0.0.1 >> /etc/resolv.conf`	Konfigurieren Sie die lokale Resolver-Library.
16	`tail -n +0 -f /var/log/{syslog,messages} | grep named &`	Betrachten Sie das Syslog, während Sie den Nameserver starten.
17	`/usr/local/sbin/named`	Starten Sie den Nameserver.
18	`rndc status`	Prüfen Sie, ob der Nameserver läuft: `server is up and running` bedeutet, dass der Nameserver läuft. `rndc: connect failed: connection refused` bedeutet, dass der Nameserver nicht läuft.

19	`vi /etc/init.d/bind`
	`ln -s /etc/init.d/bind /etc/rc2.d/S60bind`
	`ln -s /etc/init.d/bind /etc/rc2.d/K20bind`
		Integrieren Sie den Nameserver in Ihre Startscripts.

Tabelle 10.1: BIND selbst kompilieren (Fortsetzung)

10.3 Anatomie eines BIND-Servers

10.3.1 Unter Unix

Eine typische Installation eines BIND-Servers auf einem Unix-System umfasst:

- **Programme:**
  - Server-Binaries: `/usr/sbin/{named,lwresd}` (`lwresd` kann ein symbolischer Link auf `named` sein.)
  - Server-Startscripts: `/etc/init.d/{bind9,lwresd}`
  - Abfragetools: `/usr/bin/{host,dig,nslookup,nsupdate}`
  - Admintools: `/usr/bin/nsupdate`, `/usr/sbin/{rndc, dnssec-keygen, dnssec-signkey, dnssec-signzone, named-checkconf, named-checkzone}`
- **Konfigurationsdateien:**
  - Konfigurationsdatei für BIND: `/etc/bind/named.conf`
  - Konfigurationsdatei für das Admintool `rndc`: `/etc/bind/rndc.conf`
  - Zonefiles: `/etc/bind/db.*` (siehe `file`-Option in `named.conf`)
  - Das Zonefile der Root-Zone: `/etc/bind/root.hints`
- **Logdateien:**
  - über Syslog: z.B. `/var/log/syslog` oder andere Orte, die Sie in `/etc/syslog.conf` konfigurieren können
  - ggf. andere, in `/etc/bind/named.conf` definierte Orte
- **Weitere, von BIND erzeugte Dateien:**
  - Dumps und Statistiken: `/var/cache/bind/*` (siehe `directory`-Option in `named.conf`)
  - Prozess-ID-(PID-)Files: `/var/run/named.pid` (siehe `pid-file`-Option in `named.conf`) und `/var/lock/subsys/named` (vom Startscript angelegt)
- **Bibliotheken:**
  - Includefiles: `/usr/include`
  - Libs: `/usr/lib/lib{dns,isc,lwres,isccc,isccfg,omapi}*`
- **Systemvoraussetzungen**
  (nicht im Lieferumfang von BIND enthalten):
  - Devices: `/dev/{null,random,urandom,zero}`
  - Sockets: `/dev/log`
  - Zeitzoneninfo: `/etc/localtime` oder `/usr/lib/zoneinfo/localtime`
- **Dokumentation:**
  - Manpages zu allen Programmen: `/usr/share/man/man*/*`
  - weitere Dateien: `/usr/share/doc/{bind9,bind9-doc,dnsutils}/*`

Achtung: Die Pfadangaben sind Debian GNU/Linux 3.1 entnommen und weichen bei anderen Betriebssystemen und Distributionen ggf. ab:

Zum Beispiel kann das Startscript `/etc/init.d/named` und das Syslog `/var/adm/messages` (Solaris) heißen und die Konfigurationsdateien unter `/etc/named` (Red Hat/FreeBSD) oder `/var/named` (Solaris/OpenBSD) liegen.

Wenn Sie BIND in einer Changeroot-Umgebung betreiben möchten, müssen Sie dafür sorgen, dass die

- Konfigurationsdateien,
- Systemvoraussetzungen und
- die Verzeichnisse der weiteren Dateien

innerhalb des Changeroot-Verzeichnisses (z.B. `/usr/lib/bind`) zur Verfügung stehen.

Mehr zum Thema: Die Konfiguration einer Changeroot-Umgebung wird in Abschnitt 12.5.3 beschrieben.

10.3.2 Unter Windows

Eine typische Installation eines BIND-Servers unter Windows umfasst:

- **Programme** im Verzeichnis `C:\WINNT\system32\dns\bin\`:
  - Das Installationsprogramm: `BINDInstall.exe`
  - Abfragetools: `dig.exe`, `host.exe`, `nslookup.exe`
  - Admintools: `dnssec-keygen.exe`, `dnssec-signzone.exe`, `named-checkzone.exe`, `named-checkconf.exe`, `nsupdate.exe`, `rndc.exe`, `rndc-confgen.exe`
- **Konfigurationsdateien** unter `C:\WINNT\system32`:
  - für BIND: `dns\etc\named.conf`
  - für das Admintool `rndc`: `dns\etc\rndc.conf`
  - für die übrigen Admintools: `drivers\etc\resolv.conf`
  - Zonefiles: `dns\etc\db.*` (siehe `file`-Option in `named.conf`)

- **Registry-Einträge** unter HKEY_LOCAL_MACHINE:
  - SYSTEM\CurrentControlSet\Enum\Root\LEGACY_NAMED\0000\
  - SYSTEM\CurrentControlSet\Services\named\
  - SOFTWARE\ISC\BIND\
  - SOFTWARE\Microsoft\Windows\CurrentVersion\Uninstall\ISC BIND\
- **Bibliotheken** unter C:\WINNT\system32\lib:
  - lib{bind9,lwres,isccc,isccfg,dns,isc}.dll
- **Dokumentation:**
  - bleibt in dem Verzeichnis liegen, in das Sie die Zip-Datei entpackt hatten

10.4 Auf BIND9 umsteigen

10.4.1 Von BIND4

Die Konfigurationsdatei konvertieren

Der Name und die Syntax der BIND-Konfigurationsdatei wurden für Version 8 neu definiert. Die folgende BIND4-Konfigurationsdatei (/etc/named.boot):

```
directory   /var/named
xfrnets     127.0.0.1&255.255.255.255 10.0.0.0 141.2.0.0
primary     rbi.uni-frankfurt.de                    master/rbi
secondary   uni-frankfurt.de 141.2.22.74 141.2.149.10 slave/uni
cache       .                                       slave/root
```

Listing 10.1: named.boot (BIND4)

würde ab BIND8 named.conf heißen und wie folgt aussehen:

```
acl uninetz {127/8; 10/8; 141.2/16;};
options {
    directory "/var/named";
    allow-transfer {none;};
    // Das Verhalten von BIND4 emulieren:
    query-source       address * port 53;
    transfer-source            * port 53;
    notify-source              * port 53;
```

```
};
zone "rbi.informatik.uni-frankfurt.de" IN {
    type master;
    notify yes; also-notify {141.2.1.3; 141.2.1.1;};
    file "master/rbi";
    allow-transfer {uninetz;};
};
zone "uni-frankfurt.de" IN {
    type slave;
    masters {141.2.22.74; 141.2.149.10;};
    file "slave/uni";
};
zone "." IN {
    type hint;
    file "slave/root";
};
```

Listing 10.2: `named.conf` (ab BIND8)

Beim Konvertieren hilft Ihnen ein Shellscript, das Ihrer BIND-Distribution beiliegt:

```
$ named-bootconf.sh </etc/named.boot >/etc/named.conf.neu
$
```

Listing 10.3: Konvertieren der Konfigurationsdatei

Wenn Ihr BIND4-Nameserver die Verbindung zu anderen Nameservern über eine Firewall aufbauen musste, werden Sie vielleicht feststellen, dass diese Kommunikation mit BIND9 nicht mehr funktioniert. Ursache ist der Source-Port, der bei BIND4 immer 53 war und nun zufällig ausgewählt wird. Listing 10.2 zeigt, wie Sie mit den Optionen `query-source`, `transfer-source` und `notify-source` das alte Verhalten wieder herstellen.

Zonefiles anpassen

Das Format der Zonefiles hat sich von BIND4 zu BIND9 nur geringfügig geändert:

- In Hinblick auf **SOA-Records, Default-TTL** und **Syntaxfehler in der Konfigurationsdatei** arbeitet BIND4 identisch zu BIND8, daher lesen Sie bitte weiter in Abschnitt 10.4.2 (Umstieg von BIND8 auf BIND9).
- **CNAME-Records:** Ein Recordset kann mehrere Records enthalten. Unter BIND4 durfte er neben anderen Record-Typen auch CNAMEs enthalten. Beispiele hierzu finden Sie im folgenden Listing 10.4. Da die CNAMEs jedoch zu Problemen in Form unterschiedlicher Implementierungen führten, dürfen Recordsets, die einen CNAME-Record enthalten, seit BIND8 keinen weiteren Record enthalten. (Ab BIND 9.3.0 dürfen sie zusätzlich RRSIG und NSEC-Records enthalten.) Verwenden Sie einfach A-Records anstelle der kombinierten CNAMEs.

Das folgende Zonefile funktioniert nur mit BIND4:

```
$ORIGIN informatik.uni-frankfurt.de.
@   IN SOA hermes.rbi dns (          ; Primärer NS und E-Mail
    2004052117                            ; Versionsnummer
    10800 3600 300000    ; Refresh-, Retry-, Expire-Timeouts
    3600)                                 ; Minimum-Timeout
;
    CNAME sphinx.rbi          ; ermöglicht http://informatik
    MX 10 hera.rbi                            ; Mailserver
;
www CNAME sphinx.rbi                           ; Webserver
    CNAME olymp.rbi                    ; Failover-Webserver
```

Listing 10.4: BIND4-Zonefile

und muss für BIND9 wie folgt angepasst werden:

```
$ORIGIN informatik.uni-frankfurt.de.
$TTL 3600
@   IN SOA hermes.rbi dns (          ; Primärer NS und E-Mail
    2004052117                            ; Versionsnummer
    10800 3600 300000    ; Refresh-, Retry-, Expire-Timeouts
    38400)                       ; Negative-Caching-Timeout
```

```
;
    A 141.2.15.17            ; ermöglicht http://informatik
    MX 10 hera.rbi                          ; Mailserver
;
www A 141.2.15.17                            ; Webserver
    A 141.2.16.66                    ; Failover-Webserver
```

Listing 10.5: BIND9-Zonefile

Die Unterschiede sind jeweils mit fetter Schrift markiert.

10.4.2 Von BIND8

Die Konfigurationsdatei anpassen

- **Optionen:** Einige BIND8-Optionen sind unter BIND9 nicht mehr verfügbar, heißen anders oder haben andere Standardwerte. Wenn Ihre Konfigurationsdatei davon betroffen ist, wird BIND9 dies im Syslog bzw. in Logdateien protokollieren.
- **Fehlertoleranz:** Wenn Ihre Konfigurationsdatei einen Syntaxfehler enthält, wird BIND9 sich sofort nach dem Start beenden! BIND8 hatte die fehlerhaften Anweisungen in einigen Fällen ignoriert. Lesen Sie das Syslog nach dem Start mit einer neuen Konfigurationsdatei.
- **Protokollierung:** BIND9 bietet andere Filterkriterien. Wenn Sie die Protokollierung für BIND8 konfiguriert hatten, müssen Sie diese für BIND9 noch einmal konfigurieren.
- **Source-Port**: BIND9 bietet eine feinere Konfiguration des Source-Ports. Wenn Sie unter BIND8 die Option `query-source` verwendet hatten, benötigen Sie vermutlich unter BIND9 für die Notify-Nachrichten und Refresh-Queries zusätzlich die Optionen `notify-source` und `transfer-source`.

Zonefiles anpassen

- **SOA-Records:** Die BIND-Versionen 4.* und 8.1 interpretieren das letzte Datenfeld des SOA-Records als »Minimum Timeout«, während Versionen ab 8.2 an dieser Stelle das »Negative Caching Timeout« erwarten.

Zum Unterschied:

- Das Minimum Timeout wird in RFC1035 beschrieben. Es gibt die TTL an, die der Nameserver bei allen Records einsetzt, für die entweder *keine* oder ein *kleinere* TTL angegeben ist.
- Das Negative Caching Timeout wird in RFC2308 beschrieben. Es gibt an, wie lange ein Nameserver die Auskunft, dass ein Record in dieser Zone *nicht* existiert, längstens zwischenspeichern darf.

- **Default-TTL:** Nameserver setzen gemäß RFC1035 und RFC2308 die TTL des vorhergehenden Records als Default-TTL ein, wenn ein Record keine TTL-Angabe hat. Wenn *keiner* der vorhergehenden Records eine TTL-Angabe hat, greifen die BIND-Versionen 4, 8, 9.2 und 9.3 auf den Wert aus dem letzten Datenfeld des SOA-Record zurück, während die Versionen 9.0 und 9.1 die gesamte Zone ignorieren! Um ganz sicherzugehen, verwenden Sie am besten grundsätzlich die `$TTL`-Anweisung (siehe Beispiel in Listing 10.5, Abschnitt *Zonefiles anpassen*).
- **Fehlertoleranz:** Wenn Ihr Zonefile Syntaxfehler enthält, werden BIND4 und BIND8 die *fehlerhaften Einträge* ignorieren, während BIND9 gleich die *gesamte Zone* ignoriert! Dieses Verhalten können Sie ab BIND 9.3.0 mit der `check-names`-Option steuern. Lesen Sie das Syslog nach jeder Aktualisierung Ihrer Zonefiles.
- **Glue-Records:** BIND4 und BIND8 versuchen, fehlende Glue-Records aus anderen lokalen Zonefiles zu ergänzen, während BIND9 dies nicht tut. Wenn Ihre Nameserver in einer Subdomain stehen, denken Sie bitte an die Glue-Records und achten Sie darauf, dass die delegierenden NS-Records Ihrer Domain mit den autoritativen der Subdomain übereinstimmen.

Admintools

Das von BIND8 bekannte Admintool `ndc` heißt ab BIND9 `rndc` und kann nun auch auf entfernte Nameserver zugreifen. Im Gegensatz zu `ndc` benötigt `rndc` einen TSIG-Schlüssel zur Authentifizierung am Nameserver.

Mehr zum Thema: Details zu `rndc` finden Sie in Abschnitt 11.2 (Aufruf und `rndc.conf`) und Abschnitt 6.3.2 (`controls`-Anweisung in `named.conf`).

11 Admintools

11.1 Aufruf von BIND

Dieser Abschnitt beschreibt den Aufruf von BIND unter verschiedenen GNU/Linux-Distributionen.

11.1.1 Starten

Um den Nameserver zu starten, rufen Sie das entsprechende Init-Script Ihres Betriebssystems auf, z.B.

- `/etc/init.d/bind9 start` (BIND9 unter Debian)
- `/etc/init.d/named start` (Red Hat)
- `net start "isc bind"` (Windows)

Mit den folgenden Befehlen stellen Sie ein, dass dies beim Booten automatisch geschieht:

- `update-rc.d bind9 defaults` (Debian)
- `chkconfig named on` (Red Hat)

11.1.2 Stoppen

Um den Nameserver zu beenden, rufen Sie entweder

- `rndc stop`,
- das oben genannte Init-Script oder den `net`-Befehl mit dem Parameter `stop`,
- `lsof -ti :53| xargs kill -TERM`,
- die in Listing 11.1, Abschnitt 11.1.5 vorgestellte Shellfunktion oder
- (nur notfalls:) `pkill -KILL named`

auf.

Achtung: Verlassen Sie sich *nicht* auf `killall` oder `pkill`, denn mit diesen Befehlen killen Sie leicht andere Prozesse, deren Name ebenfalls die Buchstabenfolge »named« enthält.

Achtung: Wenn Ihr Nameserver Dynamic Updates empfängt, sollten Sie ihn *nicht* mit `-KILL` beenden, weil sonst die letzten Änderungen an den Zonefiles und den Zone-Logs verloren gehen können.

11.1.3 Status

Mit den folgenden Befehlen können Sie feststellen, ob Ihr Nameserver läuft:

- `lsof -ti :53 || echo NOTFOUND`
  Wenn der Nameserver läuft, wird die PID des `named`-Prozesses ausgegeben. `lsof` ist zwar kein Standardbefehl, jedoch in allen Linux-Distributionen enthalten und für Solaris z.B. auf `http://www.sunfreeware.com/` erhältlich.
- `rndc status`
  Wenn der Nameserver läuft, gibt `rndc` als letzte Zeile `server is up and running` aus, ansonsten `rndc: connect failed: connection refused`.

Achtung: Verlassen Sie sich *nicht* auf `ps -ef| grep named` oder `pgrep`, denn mit diesen Befehlen sehen Sie auch andere Prozesse, deren Name ebenfalls die Buchstabenfolge »named« enthält.

11.1.4 Startparameter

Wenn Sie

- den `named`-Prozess ohne das Init-Script aufrufen oder
- das Init-Script ändern möchten,

können Sie `named` folgende Optionen übergeben:

Parameter	Beschreibung
`-4`	verhindert, dass BIND auf IPv6-Schnittstellen lauscht.
`-6`	verhindert, dass BIND auf IPv4-Schnittstellen lauscht.
! `-c` *`Dateiname`*	verwendet die angegebene Datei als Konfigurationsdatei (anstelle von `/etc/named.conf`).
`-d` *`DebugLevel`*	aktiviert Debug-Ausgaben. Setzen Sie den *`DebugLevel`* auf eine Zahl zwischen 0 und 9. Je größer die Zahl ist, desto mehr Details gibt BIND aus. Sie können die Ausgabe der Nachrichten mit der `logging{}`-Anweisung steuern, die in Abschnitt 6.3.5 beschrieben wird.
`-f`	bewirkt, dass der Prozess sich *nicht* in den Hintergrund legt. Sie können ihn dann mit Strg + C beenden. Wenn Sie zusätzlich einen `DebugLevel` festlegen, sehen Sie alle Nachrichten der `default`-Kategorie auf `STDERR`.
`-g`	wie `-f`, Sie sehen jedoch *sämtliche* Nachrichten auf `STDERR`.
`-n` *`CPUs`*	bestimmt die Zahl der Threads, die BIND erzeugt. Setzen Sie diese auf die Zahl der Prozessoren Ihres Nameserver-Rechners, wenn Sie feststellen, dass BIND nur einen Prozessor nutzt.
`-p` *`Portnummer`*	bestimmt die Portnummer, auf der BIND lauschen soll. **Standardwert:** `53`
`-s`	gibt verschiedene Statistiken aus, die nur für BIND-Entwickler interessant sind.
! `-t Verzeichnis`	verwendet das angegebene Verzeichnis als chroot-Umgebung. **Achtung:** Verwenden Sie diesen Parameter stets in Verbindung mit `-u`, weil root aus der chroot-Umgebung ausbrechen könnte.
! `-u` *`Username`*	bestimmt den Unix-Usernamen, zu dem BIND wechselt (setuid), sobald alle Sockets geöffnet sind.

Tabelle 11.1: `named`-Parameter

Parameter	Beschreibung
`-v`	gibt die Versionsbezeichnung aus.
`-X` *`Dateiname`*	liest den Cache aus der angegebenen Datei ein. Dies ermöglicht Ihnen, den Cache Ihres Nameservers, den Sie zuvor mit `rndc dumpdb` in eine Datei gesichert haben, über einen Neustart des Nameservers hinweg zu erhalten.

Tabelle 11.1: `named`-Parameter (Fortsetzung)

11.1.5 Signale

BIND reagiert auf folgende Signale:

#	Name	Reaktion
1	SIGHUP	liest die Konfigurationsdatei und alle Zonefiles neu ein und sendet NOTIFY-Nachrichten für alle Zonen an alle sekundären Nameserver.
2 15	SIGINT SIGTERM	fährt den Nameserver ordentlich herunter.
9	SIGKILL	beendet den Nameserver *sofort*. **Achtung:** Ein Verlust der letzten Änderungen an den Zonefiles und den Zone-Logs ist möglich.
19 18	SIGSTOP SIGCONT	hält den Nameserver an und setzt ihn fort.

Tabelle 11.2: Signale für BIND

Sie können die Signale mit folgender bash-Funktion an Ihren Nameserver senden (sofern er auf Port 53 lauscht):

```
$ function bindsignal {
>     lsof -ti :53 | xargs kill -${1:-SIGHUP};
> }
$ bindsignal            # Configs neu einlesen
$ bindsignal SIGTERM    # Herunterfahren
```

Listing 11.1: Shellfunktion zum Senden von Signalen

11.2 rndc

Mit dem Programm `rndc`, das der BIND-Distribution beiliegt, können Sie verschiedene Funktionen eines Nameservers »aus der Ferne« auslösen.

Im Gegensatz zu `ndc`, das den Nameserver über einen Unix-Domain-Socket ansprach, stellt `rndc` eine TCP-Verbindung zu dem Nameserver her und authentifiziert sich mit einem TSIG-Schlüssel. Daher können Sie mit `rndc` von *einem* Rechner aus *mehrere* lokale und entfernte Nameserver steuern.

11.2.1 Aufruf

Der Aufruf von `rndc` erfolgt nach folgendem Schema:

```
rndc [Optionen] Befehl
```

Die Optionen und Befehle werden in den folgenden Abschnitten beschrieben.

Optionen

Option	Beschreibung
`-c` *Dateiname*	verwendet die angegebene Datei als Konfigurationsdatei. **Standardwert:** `/etc/rndc.conf`
`-k` *Dateiname*	liest den TSIG-Schlüssel (zur Authentifizierung am Nameserver) aus der angegebenen Datei. Wenn Sie diese Option *nicht* angeben und die Konfigurationsdatei *nicht* existiert, verwendet BIND folgenden **Standardwert:** `/etc/rndc.key`
`-s` *Nameserver*	gibt die IP-Adresse oder den Namen des Nameservers an. **Standardwert:** Der Wert der `default-server`-Anweisung in der Konfigurationsdatei oder `localhost`.
`-p` *Portnummer*	bestimmt den TCP-Port, auf dem der Nameserver control-Verbindungen entgegennimmt. **Standardwert:** 953
`-V`	aktiviert zusätzliche Ausgaben.

Tabelle 11.3: `rndc`-Optionen

Option	Beschreibung
`-y Schlüsselname`	gibt den Namen des TSIG-Schlüssels an. Wenn Sie diese Option *nicht* angeben und die Konfigurationsdatei *nicht* existiert, verwendet BIND folgenden **Standardwert:** `rndc-key`

Tabelle 11.3: `rndc`-Optionen (Fortsetzung)

Befehle

Befehl	Reaktion des Nameservers
`dumpdb`	schreibt den Cache in eine Datei. Den Ort dieser Datei können Sie in der Konfigurationsdatei mit der `statistics-file`-Option festlegen (Standardwert: `/var/cache/bind/named_dump.db`).
`flush[view]`	löscht den Cache (des angegebenen Views). Führen Sie dies aus, wenn BIND zu viel Hauptspeicher verwendet.
`flushname Name [View]`	löscht den angegebenen Recordset aus dem Cache.
`freeze Zone [Klasse [View]]`	verhindert die Verarbeitung von Dynamic Updates auf die angegebene Zone. Dieser Befehl steht erst ab BIND 9.3 zur Verfügung.
`halt [-p]`	beendet den Nameserver sofort (ohne die letzten Änderungen der Zonefiles zu sichern). Wenn Sie `-p` angeben, gibt `rndc` dabei die PID des beendeten `named`-Prozesses aus.
`notrace`	setzt den Debuglevel auf 0.
`querylog`	schaltet das Protokollieren von Abfragen an oder aus.
`reconfig`	liest die Konfigurationsdatei neu ein und aktiviert nur die Zonen, die neu hinzugekommen sind.

Tabelle 11.4: `rndc`-Befehle

Befehl	Reaktion des Nameservers
`recursing`	schreibt die aktuell bearbeiteten, rekursiven Anfragen in die Recursing-Datei. Den Ort dieser Datei können Sie in der Konfigurationsdatei mit der `recursing-file`-Option festlegen (Standardwert: `/var/cache/bind/named.recursing`).

`refresh`
`Zone`
`[`*`Klasse`*` [`*`View`*`]]`
	veranlasst einen sofortigen Refresh der angegebenen sekundären Zone (d.h. SOA-Record anfordern und ggf. Zonetransfer).

`reload`	liest die Konfigurationsdatei neu ein. Für jede primäre Zone liest BIND das Zonefile ein und sendet NOTIFY-Nachrichten an alle sekundären Nameserver. Für jede sekundäre Zone fordert BIND einen Zonetransfer vom primären Nameserver an.

`reload`
`Zone`
`[`*`Klasse`*` [`*`View`*`]]`
	aktualisiert die angegebene Zone:
	Wenn es sich um eine primäre Zone handelt, liest BIND das Zonefile ein und sendet NOTIFY-Nachrichten an alle sekundären Nameserver.
	Wenn es sich um eine sekundäre Zone handelt, veranlasst BIND beim nächsten Heartbeat einen Refresh der angegebenen Zone (d.h. SOA-Record anfordern und ggf. Zonetransfer).
	Die Frequenz des Heartbeats können Sie mit der `heartbeat-interval`-Option festlegen, die in Abschnitt 6.3.7 beschrieben wird. Der Standardwert beträgt eine Stunde.

`retransfer`
`Zone`
`[`*`Klasse`*` [`*`View`*`]]`
	veranlasst einen sofortigen Zonetransfer der angegebenen sekundären Zone vom primären Nameserver.

Tabelle 11.4: `rndc`-Befehle (Fortsetzung)

Befehl	Reaktion des Nameservers
`stats`	gibt verschiedene statistische Werte in die Statistik-Datei aus. Den Ort dieser Datei können Sie in der Konfigurationsdatei mit der `statistics-file`-Option festlegen (Standardwert: `/var/cache/bind/named.stats`).
`status`	gibt den Status des Nameservers aus.
`stop [-p]`	schreibt die letzten Änderungen in die Zonefiles und beendet den Nameserver. Wenn Sie `-p` angeben, gibt `rndc` dabei die PID des beendeten `named`-Prozesses aus.
`trace[`*`DebugLevel`*`]`	setzt den Debuglevel auf den angegebenen Wert oder erhöht ihn um eine Stufe.

`thaw`
`Zone`
`[`*`Klasse`*` [`*`View`*`]]`
	aktualisiert die angegebene Zone (siehe `reload`) und setzt die Verarbeitung der Dynamic Updates für sie fort.
	Dieser Befehl steht erst ab BIND 9.3 zur Verfügung.

Tabelle 11.4: `rndc`-Befehle (Fortsetzung)

11.2.2 Konfiguration

rndc.conf

In der Datei `/etc/bind/rndc.conf` können Sie Standardwerte festlegen, die `rndc` verwendet, wenn beim Aufruf kein Nameserver, Port oder Schlüssel übergeben wird.

Wenn diese Datei nicht existiert, greift `rndc` standardmäßig auf `localhost:953` zu und liest den Schlüssel aus der Datei `/etc/bind/rndc.key`.

Mehr zum Thema: Ein Beispiel für die Datei `/etc/bind/rndc.key` finden Sie in Abschnitt 8.1.2.

Das Format der `rndc.conf`- ähnelt dem der `named.conf`-Datei, erlaubt jedoch nur drei Anweisungsblöcke:

Anweisung	Beschreibung

`options`	bestimmt, welchen Server, welche Portnummer und welchen Schlüssel `rndc` verwendet, wenn Sie diese *nicht* beim Aufruf oder in der `server`-Anweisung angeben.
	Beispiel:
	`options {`
	`  default-server localhost;`
	`  default-key rndc-key;`
	`  default-port 953;`
	`};`

`server`	bestimmt für einen Server, welchen Schlüssel und welche Portnummer `rndc` verwendet. Sie können diese Einstellungen aber beim Aufruf von `rndc` mit anderen Werten belegen.
	Die Konfigurationsdatei kann bis zu zehn `server`-Anweisungen enthalten.
	Beispiel:
	`server localhost {`
	`  key rndc-key;`
	`  port 953;`
	`};`

`key`	definiert einen TSIG-Schlüssel.
	Die Syntax entspricht der `key`-Anweisung in `named.conf` (siehe Abschnitt 6.3.4).
	Die Konfigurationsdatei kann mehrere `key`-Anweisungen enthalten.
	Beispiel:
	`key "rndc-key" {`
	`  algorithm hmac-md5;`
	`  secret "P/25ojDQpsJpMeK6nKvXmQ==";`
	`};`

Tabelle 11.5: Anweisungen in /etc/bind/rndc.conf

Sie (oder Ihr Distributor) können das Programm `rndc-confgen` zum Erstellen gültiger `rndc.conf`-Dateien aufrufen.

named.conf

In der Datei `/etc/bind/named.conf` können Sie mit der `controls`-Anweisung einstellen, auf welcher IP-Adresse und Portnummer BIND `rndc`-Verbindungen entgegennimmt, und eine Liste berechtigter Schlüssel angeben.

Wenn die Datei `/etc/bind/named.conf` keine `controls{}`-Anweisung enthält, liest BIND den Schlüssel aus `/etc/bind/rndc.key` und lauscht auf Port 953 der beiden Loopback-Interfaces (IPv4 und IPv6).

Mehr zum Thema: Die `controls`-Anweisung wird in Abschnitt 6.3.2 und die `key`-Anweisung in Abschnitt 6.3.4 beschrieben.

11.3 nsupdate

Das Programm `nsupdate` ist Bestandteil der BIND-Distribution und dient zum Senden von Dynamic Updates.

Aufruf:

`nsupdate -`*`Optionen`* [*`Dateiname`*]

`nsupdate` liest die Kommandos aus der angegebenen Textdatei oder (wenn Sie *keinen* *`Dateiname`* angeben) von STDIN ein und sendet sie an den Rechner, der im SOA-Record der betroffenen Zone als primärer Nameserver angegeben ist.

Zur Authentifizierung am Nameserver kann `nsupdate` Transaktionssignaturen (TSIG oder SIG(0)) einsetzen.

Mehr zum Thema:

Optionen und Kommandos: Abschnitte 11.3.1 und 11.3.2

`dnssec-keygen`: Abschnitt 12.4.1

Dynamic Updates: RFC2136

Absicherung der Updates mit TSIG oder SIG(0): RFC3007

Der HMAC-MD5-Algorithmus: RFC2104

Der TSIG-Algorithmus: RFC2845
DNSSEC: RFC2535
Der SIG(0)-Algorithmus: RFC2931 und RFC3535

11.3.1 Optionen

Sie können `nsupdate` mit folgenden Optionen aufrufen:

Option	Beschreibung
`-d`	aktiviert den Debug-Modus, in dem `nsupdate` jedes einzelne Update und die Antwort des Nameservers anzeigt.
`-k` *`Dateiname`*	gibt an, welcher Datei `nsupdate` den Schlüssel zur Authentifizierung entnehmen soll. Der *`Dateiname`* muss dem Schema `K`*`Schlüsselname`*`.+157+`*`beliebigertext`*`.private` entsprechen (z.B. `Kexample.com.+157+26160.private`). Bitte beachten Sie, dass eine Kopie dieser Datei mit der Endung `.key` im selben Verzeichnis liegen muss. Die Optionen `-k` und `-y` schließen sich gegenseitig aus.
`-r` *`Versuche`*	gibt an, wie oft `nsupdate` eine UDP-Nachricht wiederholen darf, wenn der Nameserver nicht innerhalb des mit `-u` angegebenen Timeouts antwortet. Geben Sie `0` an, um Wiederholungen zu vermeiden. Diese Option wirkt sich nur auf UDP-Datagramme aus, da die Wiederholungen bei TCP vom Betriebssystemkern ausgelöst werden. **Standardwert:** `3`
`-t` *`Sekunden`*	gibt an, wie lange ein Update maximal dauern darf, bevor `nsupdate` abbricht. Dies schließt alle UDP-Wiederholungen und TCP-Retransmits ein. **Standardwert:** `300` (fünf Minuten)

Tabelle 11.6: `nsupdate`-Optionen

Option	Beschreibung
`-u Sekunden`	gibt an, wie lange `nsupdate` bei UDP-Datagrammen auf die Antwort des Nameservers wartet. Wenn Sie `0` angeben, setzt `nsupdate` den Quotient aus dem Gesamttimeout (`-t`) und der Zahl der Wiederholungen (`-r`) ein. **Standardwert:** `3`
`-v`	bewirkt, dass `nsupdate` die Updates über TCP statt über UDP sendet. **Standardwert:** `nsupdate` sendet über UDP.
`-y` *`Schlüsselname`*`:` *`Schlüssel`*	gibt den Schlüssel zur Authentifizierung an der Kommandozeile an.
	Achtung: Bedenken Sie, dass andere User dieses Passwort in der Prozessliste oder ihrer Shell-History erspähen könnten.
	Daher sollten Sie den Schlüssel besser in eine Datei schreiben und `nsupdate` lediglich den Dateinamen mit `-k` übergeben.

Tabelle 11.6: `nsupdate`-Optionen (Fortsetzung)

11.3.2 Kommandos

`nsupdate` kennt drei verschiedene Arten von Kommandos:

- Änderungen
- Voraussetzungen, die erfüllt sein müssen, bevor die Änderung stattfindet
- Steuerungsanweisungen, die die Kommunikation mit dem Nameserver betreffen

Achtung: `nsupdate` sendet Ihre Eingaben nicht sofort an den Nameserver, sondern erst, wenn Sie eine Leerzeile oder das `send`-Kommando eingeben.

Beispiele finden Sie im nachfolgenden Abschnitt 11.3.3.

Änderungen

Kommando	Beschreibung
! `update add` *RecordName* *TTL* [*Klasse*] *Typ* *Daten*	weist den Nameserver an, der Zone einen neuen Record hinzuzufügen.
! `update delete` *RecordName* [*TTL*] [*Klasse*] [*Typ* [*Daten*]]	weist den Nameserver an, aus der Zone alle Records zu löschen, auf die der angegebene Name, die Klasse, der Typ *und* die Daten zutreffen.

Tabelle 11.7: nsupdate-Änderungen

Voraussetzungen

Kommando	Beschreibung
`prereq nxdomain` *RecordName*	bewirkt, dass der Nameserver die Änderungen nur dann durchführt, wenn es zu dem angegebenen Namen noch keinen Recordset gibt.
`prereq nxrrset` *RecordName* [*Klasse*] *Typ*	bewirkt, dass der Nameserver die Änderungen nur dann durchführt, wenn es zu dem angegebenen Namen noch keinen Record des angegebenen Typs gibt.
`prereq yxdomain` *RecordName*	bewirkt, dass der Nameserver die Änderungen nur dann durchführt, wenn es zu dem angegebenen Namen bereits einen Record (beliebigen Typs) gibt.
`prereq yxrrset` *RecordName* [*Klasse*] *Typ* [*Daten*]	bewirkt, dass der Nameserver die Änderungen nur dann durchführt, wenn es zu dem angegebenen Namen bereits einen Record mit den angegebenen Werten gibt.

Tabelle 11.8: nsupdate-Voraussetzungen

Steuerungsanweisungen

Kommando	Beschreibung
`answer`	zeigt die letzte Antwort des Nameservers an.
`class` *`Klasse`*	bestimmt, welche Klasse `nsupdate` bei allen Kommandos einsetzt, bei denen Sie die Klasse weglassen. **Standardwert:** `IN` (Internet)
`key` *`Schlüsselname`* *`Schlüssel`*	gibt den TSIG-Schlüssel zur Authentifizierung an. **Standardwert:** Die Angaben der Optionen `-y` und `-k`
`local` *`IP-Adresse`* [*`Portnummer`*]	gibt die lokale IP-Adresse und (bei UDP) die Portnummer an, die *`nsupdate`* als Absender verwendet. Bei TCP wird die Portnummer normalerweise zufällig gewählt, weshalb die Angabe einer Portnummer in Kombination mit der `-v`-Option nicht erforderlich ist und sogar zu Verbindungsproblemen führen kann. **Standardwert:** Ein zufälliger Port auf der Adresse, die (laut Routingtabelle) dem Nameserver am nächsten ist
`send` oder ↵	sendet alle zuvor eingestellten Kommandos (außer den Steuerungsanweisungen) an den Nameserver.
`server` *`Nameserver`* [*`Portnummer`*]	bestimmt den Nameserver und den UDP- bzw. TCP-Port, an den `nsupdate` die Updates sendet. **Standardwert:** Der Nameserver aus dem SOA-Record und Port 53.
`show`	zeigt alle Kommandos an, die beim nächsten `send`-Kommando an den Nameserver gesendet würden.
`zone` *`Domain`*	bestimmt die Zone, auf die sich die Kommandos beziehen. **Standardwert:** `nsupdate` versucht, die Zone von den übrigen Kommandos abzuleiten.

Tabelle 11.9: `nsupdate`-Steueranweisungen

11.3.3 Beispiele

So können Sie die IP-Adresse eines A-Records ändern:

```
$ nsupdate
> update delete oldhost.example.com A
> update add newhost.example.com 86400 A 172.16.1.1
> send
$
```

Listing 11.2: Beispiel zu `nsupdate` (aus der Manpage)

Wenn Sie einen CNAME-Record hinzufügen, sollten Sie vorher sicherstellen, dass in der Zone noch kein Record mit diesem Namen enthalten ist.

```
$ nsupdate
> prereq nxdomain nickname.example.com
> update add nickname.example.com 86400 CNAME \
somehost.example.com
> send
$
```

Listing 11.3: Beispiel zu `nsupdate-prereq` (aus der Manpage)

11.4 Grafische Konfigurationstools

Es gibt viele Programme zur grafischen Konfiguration von BIND, von denen einige im folgenden Abschnitt kurz vorgestellt werden.

11.4.1 webmin

Webmin ist ein OpenSource-Projekt. Es enthält CGI-Scripts, die HTML-Formulare zur Konfiguration verschiedener Linux-Programme ausgeben und verarbeiten.

Mit dem `BIND8`-Modul können fortgeschrittene Anwender Zonefiles und komplexe Konfigurationsdateien für BIND9 erstellen.

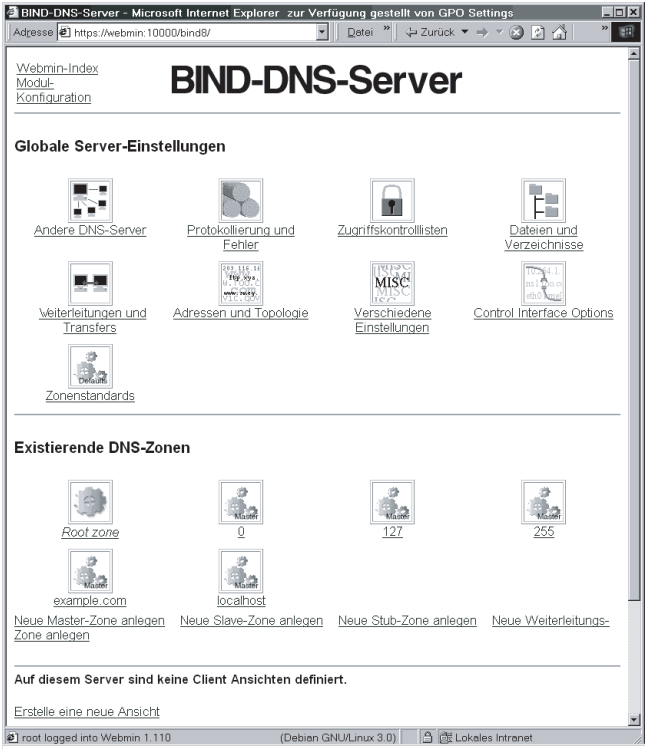

Abbildung 11.1: Webmins BIND-Modul: Hauptmenü

Webmin bietet zwar Formulare für die wichtigsten Record-Typen, alle anderen Records müssen Sie aber über den eingebauten Zonefile-Texteditor eingeben.

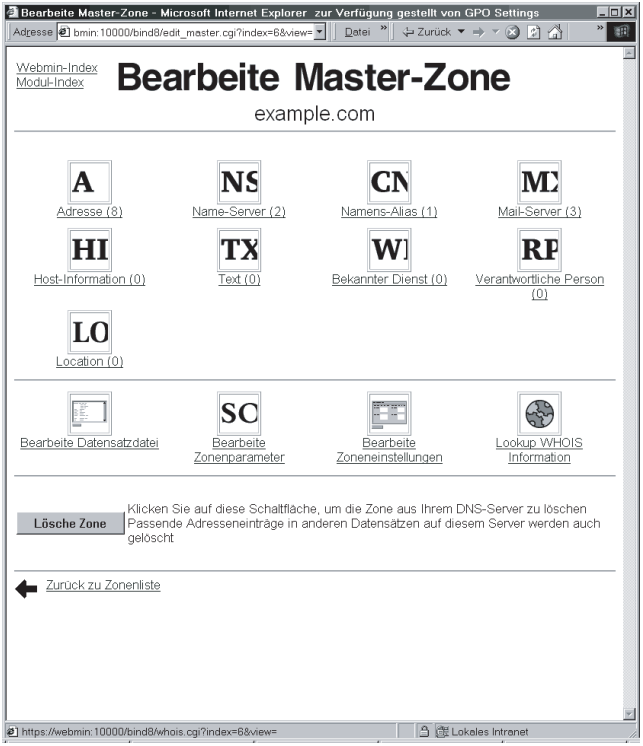

Abbildung 11.2: Webmins BIND-Modul: Zonefile-Editor

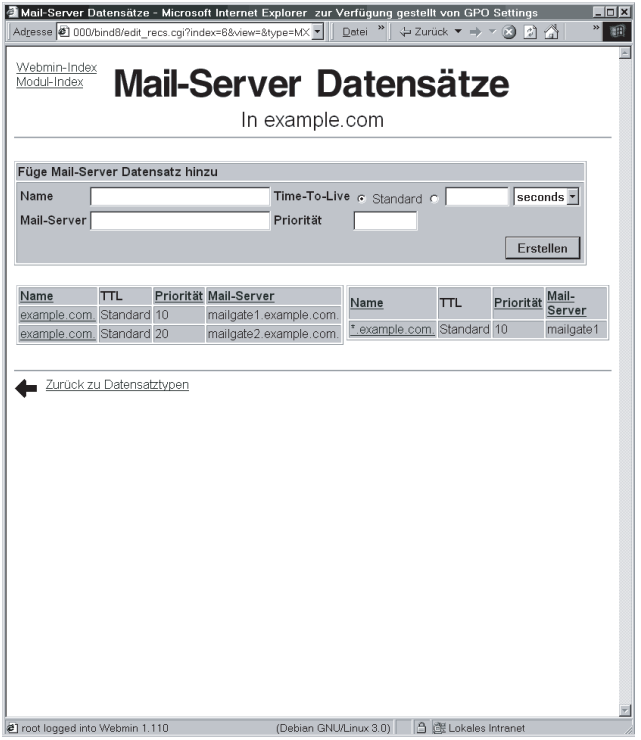

Abbildung 11.3: Webmins BIND-Modul: Record-Editor

Mehr zum Thema: Siehe `http://www.webmin.com/`

11.4.2 Lucent VitalQIP

VitalQIP ist ein professionelles IP-Managementsystem der Firma Lucent Technologies.

Es besteht aus verschiedenen *Admintools* (GUI, Web, CLI), mit denen Sie Ihre Zonen und Objekte in einer Oracle- oder Sybase-*Datenbank* bearbeiten, aus der ein *Enterprise-Server* DNS-Zonefiles und DHCP-Leases erstellt und damit Ihre *DHCP*- und *Nameserver* (BIND, Lucent oder Microsoft) aktualisiert.

Somit liegen alle Daten bzgl. DNS, DHCP und Netzen in einer zentralen Datenbank.

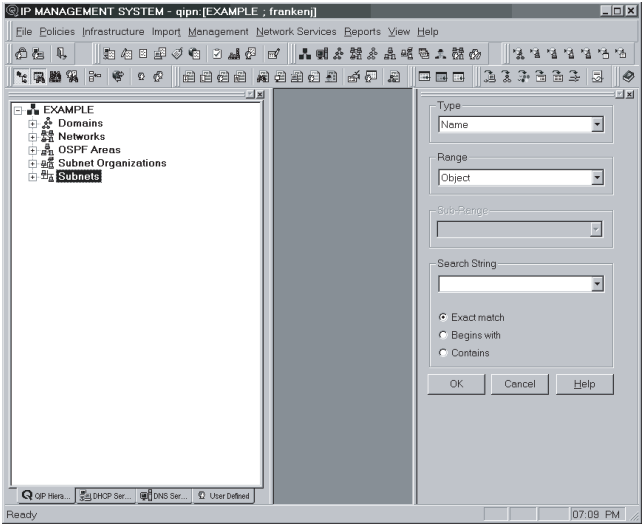

Abbildung 11.4: VitalQIP: Hauptfenster

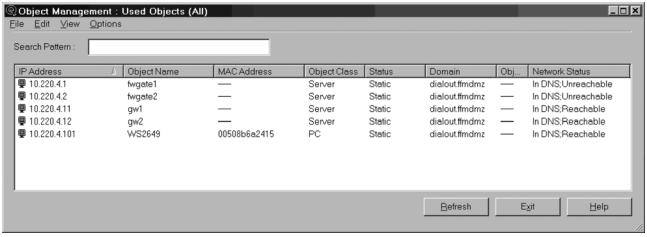

Abbildung 11.5: VitalQIP: Zonen-Editor

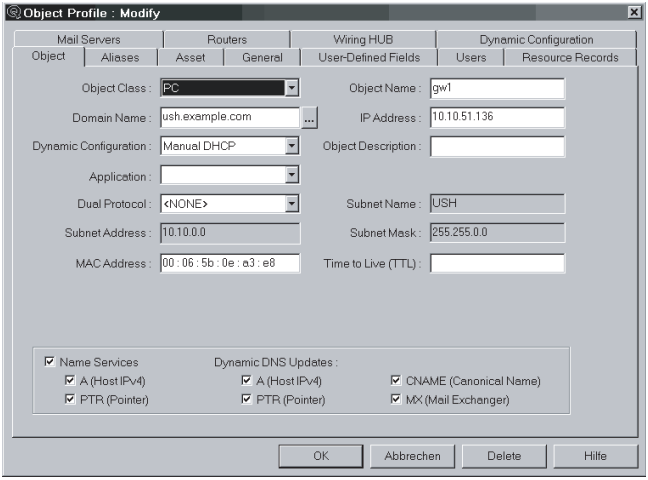

Abbildung 11.6: VitalQIP: Objekt-Editor

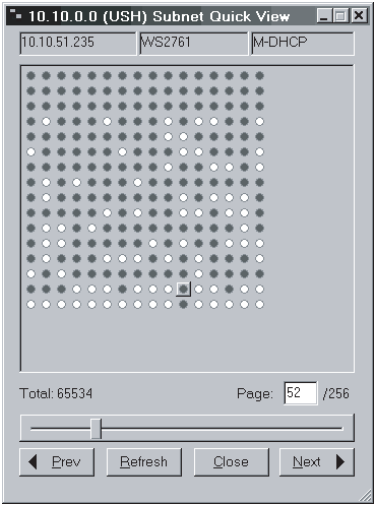

Abbildung 11.7: VitalQIP: Netz-Belegungsübersicht

Mehr zum Thema: Bitte wenden Sie sich an die n3k Informatik GmbH (`www.n3k.de`), die dieses Produkt in Deutschland vertreibt.

11.4.3 redhat-config-bind

`redhat-config-bind` ist eine X11-Anwendung, mit der Sie Zonefiles und `zone`-Anweisungen erstellen können. Sie liegt der Red-Hat-Linux-Distribution bei.

Allerdings kennt sie in der aktuellen Version (2.0.0) nur fünf verschiedene Record-Typen und kann einige Zonefiles, die mit anderen Tools erstellt wurden, *nicht* einlesen. Die Suche nach einem Record stellt sich bei großen Zonefiles schwierig dar.

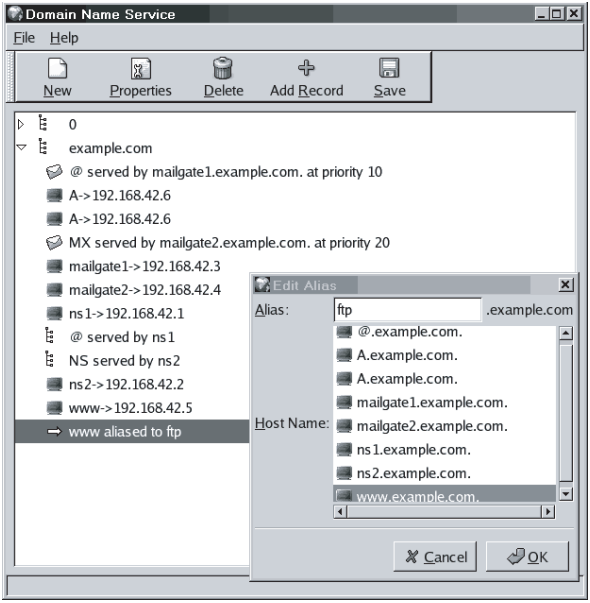

Abbildung 11.8: redhat-config-bind

Mehr zum Thema: `redhat-config-bind` wird in der Red-Hat-Dokumentation beschrieben.

12 Security

12.1 Motivation

Das DNS ist anfällig für *DNS-Spoofing*, das vom Bundesamt für Sicherheit in der Informationstechnik (BSI) wie folgt definiert wird:

> *Von »DNS-Spoofing« ist die Rede, wenn es einem Einbrecher gelingt, die Zuordnung zwischen einem Rechnernamen und der zugehörigen IP-Adresse zu fälschen, das heißt, dass ein Name in eine falsche IP-Adresse und umgekehrt umgewandelt wird.*

Durch Manipulation von Zonefiles oder mit IP-Spoofing können Hacker Zugriffe auf Onlinebanking-Seiten oder Onlineshops über ihre eigenen Server umleiten und dabei alle PINs, TANs und Kreditkartennummern aufzeichnen (»Man in the middle attack«).

Die Gefahr, die von diesen Angriffen ausgeht, ist riesig. Das BSI empfiehlt daher, in sensiblen Bereichen auf den Einsatz von DNS zu verzichten.

Die folgenden Abschnitte *Transaktionssignaturen* und *DNSSEC* beschreiben die offiziellen Verfahren zum Schutz vor DNS-Spoofing, die bis zum Oktober 2004 von den zuständigen Arbeitsgruppen der IETF erarbeitet wurden. Diese Verfahren eignen sich gut für private Netze, wären jedoch im Internet nur mit sehr hohem Aufwand umsetzbar.

Der Abschnitt *BIND härten* stellt verschiedene Maßnahmen vor, mit denen Sie einen Nameserver mit wenig Aufwand vor verschiedenen, direkten Angriffen (jedoch nicht vor DNS-Spoofing) schützen können.

12.2 Transaktionssignaturen

Mit Signaturen, die lediglich für eine Transaktion gültig sind, kann ein Nameserver sicherstellen, dass eine Nachricht

- vollständig und unverändert übertragen wurde (Integrität)

- von einem Rechner stammt, der ihm persönlich bekannt ist (Identität) und mit dem er ein gemeinsames Geheimnis (z.B. Passwort) kennt (Authentizität)

> **Achtung:** Transaktionssignaturen sind nur so sicher wie die geheimen Schlüssel, die ihnen zu Grunde liegen. Wenn ein Schlüssel von Dritten geknackt oder ausspioniert wird, ist die Aussage aller Transaktionssignaturen, die ab diesem Zeitpunkt damit erstellt werden, wertlos.

BIND kann Transaktionssignaturen zur Absicherung von Dynamic Updates, Zonetransfers, Fernsteuerungszugriffen (`control`/`rndc`) und normalen Abfragen einsetzen.

BIND kennt zwei unterschiedliche Transaktionssignatur-Verfahren:

- TSIG verwendet Shared Secrets.
- SIG(0) verwendet Public Keys.

Beide Verfahren erweitern die Nachricht um einen speziellen Record in der Additional-Section, der eine Signatur über die ursprüngliche Nachricht sowie einen Gültigkeitszeitraum für die Signatur enthält.

Die Signatur ist eine kryptografische Zusammenfassung der Nachricht, die nur der Inhaber des gemeinsamen Geheimnisses (TSIG) oder des privaten Schlüssels (SIG(0)) erstellen kann.

Der Gültigkeitszeitraum verhindert, dass der Nameserver die Nachricht noch einmal verarbeitet, wenn ein Dritter sie mitschneidet und zu einem späteren Zeitpunkt noch einmal abspielt (Replay-Attack).

> **Achtung:** Die Prüfung des Gültigkeitszeitraums schlägt fehl, wenn die Systemzeit der beiden Nameserver (in GMT transponiert) zu weit auseinander liegt. Die RFCs empfehlen Nameserverprogrammierern, Abweichungen bis zu fünf Minuten zu tolerieren.

Mehr zum Thema: Verschiedene Methoden zur Synchronisation von Systemzeiten finden Sie auf `http://www.jfranken.de/homepages/johannes/vortraege/ntp.de.html`.

12.2.1 TSIG und TKEY [shared secrets]

TSIG

Der TSIG-Algorithmus beschreibt die nachstehenden Verfahren als Erweiterung der RFC1035. Sie ermöglichen einem Nameserver, zu prüfen, ob der Absender einer DNS-Abfrage oder -Antwort ein bestimmtes Geheimnis (»Shared secret«) kennt.

Verfahren beim Sender:

Der Sender stellt einen TSIG-Record an das Ende der Additional-Section, der

- im Namensfeld den Namen eines Schlüssels (die Identität)
- im Datenfeld u.a. eine Signatur und ihren Gültigkeitszeitraum

enthält. Die Signatur (»Message Authentication Code« (MAC)) wird mit dem HMAC-MD5-Algorithmus aus der Nachricht und dem Shared secret berechnet.

Verfahren beim Empfänger:

Für die Auswertung der Signatur gelten zwei Regeln:

- Wer eine Signatur nicht bestätigen kann oder außerhalb des Gültigkeitszeitraums empfängt, muss die Nachricht verwerfen.
- Wenn es sich bei der empfangenen Nachricht um eine Abfrage handelt (also das QR-Bit nicht gesetzt ist), muss die Antwort mit demselben Schlüssel signiert werden.

Mehr zum Thema: TSIG wird in RFC2845, HMAC in RFC2104 und MD5 in RFC1321 beschrieben.

Das Erstellen von TSIG-Schlüsseln mit dem Programm `dnssec-keygen` wird in Abschnitt 12.4 und Abschnitt 6.3.4 (bei der `key`-Anweisung) beschrieben.

TKEY

TKEY ist ein Verfahren, bei dem TSIG-Schlüssel auf Anfrage generiert und verschlüsselt übertragen werden, so dass ein manueller Austausch der TSIG-Schlüssel nicht mehr erforderlich ist. Hierbei kommen der Diffie-Hellmann-(DH-)Key-Exchange-(KEX-)Algorithmus und ein TKEY-Record in der Additional-Section von DNS-Nachrichten zum Einsatz.

Wenn Sie TKEY verwenden möchten, müssen Sie

1. mit `dnssec-keygen` ein DH-Schlüsselpaar erstellen,
2. die Optionen `tkey-dhkey` und `tkey-domain` in der Konfigurationsdatei setzen.

Mehr zum Thema: TKEY wird in RFC2930 und der DH-KEX in RFC2539 beschrieben.

Das Erstellen von DH-Schlüsseln und KEY-Records mit dem Programm `dnssec-keygen` wird in Abschnitt 12.4 beschrieben.

Die Konfigurationsoptionen `tkey-dhkey` und `tkey-domain` werden in Abschnitt 6.3.7 beschrieben.

12.2.2 SIG(0) [Public-Key]

Genauso wie einen TSIG-Record kann BIND auch einen SIG-Record der Additional-Section hinzufügen. Ein SIG-Record enthält jedoch andere Informationen:

- eine Signatur, die mit einem Public-Key-Verfahren (z.B. RSA/MD5 oder DSA) aus der Nachricht und einem privaten Schlüssel erstellt wurde
- den DNS-Namen des KEY-Records, der den öffentlichen Schlüssel enthält
- den Gültigkeitszeitraum dieser Signatur

Da der öffentliche Schlüssel, der zum Prüfen der Signatur erforderlich ist, im DNS liegt, kann jeder Empfänger die Nachricht prüfen, ohne zuvor ein Shared secret mit dem Absender vereinbart zu haben.

Die Bezeichnung SIG(0) soll an die zwei unterschiedlichen Typen von SIG-Records erinnern:

- Typ 0 bezieht sich auf einen Host oder User.
- Typ 256 wurde in einer früheren Implementierung des DNSSEC zur Authentifizierung von Zonen eingesetzt, inzwischen jedoch durch den RRSIG-Record ersetzt.

Achtung: Das Erstellen und Vergleichen von SIG(0)-Records beansprucht die CPUs der beteiligten Nameserver wesentlich stärker, als dies beim TSIG-Algorithmus der Fall wäre.

Mehr zum Thema:

Das Erstellen von DH-Schlüsseln und KEY-Records mit dem Programm `dnssec-keygen` wird in Abschnitt 12.4 beschrieben.

SIG(0)-Algorithmus: RFC2931

SIG- und KEY-Record-Typen: RFC2535, Abschnitt 7.5.37 und 7.5.16

DSA-Algorithmus: RFC2536

RSA/MD5-Algorithmus: RFC2537

12.3 DNSSEC

Mit den »Domain Name System Security Extensions« (DNSSEC) können Resolver *echte* Antworten von *gefälschten* unterscheiden. Gefälschte Antworten können von Nameservern stammen, die Hackern zugänglich sind und aufgrund manipulierter Zonefiles (»DNS-Spoofing«) oder IP-Adressen (»IP-Spoofing«) angesprochen werden.

Vor dem Einsatz von DNSSEC sind verschiedene Vorbereitungen in allen (!) Zonen erforderlich:

- Die Hostmaster fügen in ihren Zonefiles zu jedem Recordset einen NSEC-Record ein, der die Typen des aktuellen und den Namen des alphabetisch nächsten Records angibt.

  Achtung: Dies erlaubt ein Ausspionieren der Zone ganz ohne Zonetransfer.

- Die Hostmaster erstellen für jede ihrer Zonen ein DSA- oder RSA-Schlüsselpaar:
  - Den öffentlichen Schlüssel veröffentlichen sie als DNSKEY-Record im Zonefile.
  - Mit dem privaten Schlüssel signieren sie alle Records der Zone (inkl. der NSEC- und DNSKEY-Records), indem sie jedem Recordset einen RRSIG-Record zur Seite stellen.
- Die übergeordnete Zone fügt bei der Delegation (zusätzlich zu den NS-Records) einen DS-Record ein, der die Prüfsumme des DNSKEY-Records der delegierten Zone enthält und ebenfalls mit einem RRSIG-Record signiert ist.

Mit dem DNSKEY-Record der Zone kann ein Resolver den RRSIG-Record und mit den DS-Records die gesamte Delegation ab der Root-Zone prüfen. Der RRSIG-Record des NSEC-Records signiert die Aussage, dass ein Record *nicht* existiert.

Achtung: Die Aussage einer Signatur verliert ihren Wert, wenn der private Schlüssel der Zone in die Hände Unbefugter fällt oder einer der Root-Nameserver eine gefälschte Ausgabe der Root-Zone verbreitet.

Mehr zum Thema: Die DNSKEY-, DS-, NSEC- und RRSIG-Records werden in Kapitel 7 beschrieben.

12.3.1 Status

DNSSEC wurde im März 1999 in RFC2535 definiert, doch es gab seitdem zahlreiche Nachbesserungen (z.B. RFC3008, RFC3090, RFC3755, RFC3757 und RFC3845). Eine zusammenfassende RFC lag zum Zeitpunkt der Erstellung dieses Buches (September 2004) leider noch nicht vor, aber es gab bei der IETF bereits verschiedene Entwürfe.

Zum aktuellen Zeitpunkt sind die Zonefiles der Rootzone und vieler TLDs noch unsigniert.

Mehr zum Thema: Siehe `http://www.dnssec.net/`

12.3.2 Konfiguration

Wenn Sie BIND für DNSSEC konfigurieren möchten:

1. Besorgen Sie sich die aktuellste Version von BIND. (Versionen vor 9.3.0 sind für DNSSEC unbrauchbar.)
2. Erstellen Sie die Schlüssel mit `dnssec-keygen` (siehe Abschnitt 12.4.1).
3. Bearbeiten Sie die Zonefiles:
   - Fügen Sie den DNSKEY-Record aus der `.key`-Datei in die Zonefiles ein.
   - Signieren Sie Ihre Zonen mit `dnssec-signzone` (siehe Abschnitt 12.4.2).
4. Bearbeiten Sie die Datei `named.conf` (siehe Kapitel 6):
   - Fügen Sie in der `zone{}`-Anweisung die Endung `.signed` an die Dateinamen der signierten Zonefiles an.

- Kopieren Sie den öffentlichen Schlüssel aus dem DNSKEY-Record der höchsten, signierten Zone in eine `trusted-keys`-Anweisung.
- Schreiben Sie `dnssec-enable yes;` und ggf. `dnssec-must-be-secure yes;` in die `options{}`-Anweisung.

5. Aktivieren Sie die neue Konfiguration (z.B. mit `rndc reload`, siehe Abschnitt 11.2).
6. Verwenden Sie `dnssec-makekeyset` (siehe Abschnitt 12.4.3) und (ggf. der Hostmaster Ihrer übergeordneten Zone) `dnssec-signkey` (siehe Abschnitt 12.4.4) zum Erzeugen der DS-Records.

> **Achtung:** Vergessen Sie nicht, in der Datei `named.conf` die Option `dnssec-enable` auf `yes` zu setzen.

12.4 Tools

12.4.1 dnssec-keygen

Mit `dnssec-keygen` können Sie Schlüssel für TSIG, TKEY, SIG(0) und DNSSEC erstellen.

`dnssec-keygen` erstellt zwei Dateien:

- K*SchlüsselName.+Algorithmusnummer+Quersumme*.private
- K*SchlüsselName.+Algorithmusnummer+Quersumme*.key

Die Verwendung dieser Dateien hängt vom gewählten Algorithmus ab:

Algorithmus	Algnr.	Schlüssel-Länge	Inhalt und Verwendung
RSAMD5 (RSA), (für DNSSEC oder SIG(0)):	001	512–2048	Die .private-Datei wird von Programmen wie dnssec-signkey oder dnssec-signzone gelesen. Die .key-Datei enthält einen DNSKEY-Record, den Sie in Ihr Zonefile kopieren oder $INCLUDEn können.

Tabelle 12.1: Inhalt der von dnssec-keygen erstellten Dateien

Algorithmus	Algnr.	Schlüssel-Länge	Inhalt und Verwendung
DH (für TKEY):	002	128–4096	Beide Dateien sind zum verschlüsselten Übertragen von TSIG-Schlüsseln im TKEY-Verfahren erforderlich. Geben Sie den Dateinamensstamm bei der tkey-dhkey-Option in Ihrer BIND-Konfigurationsdatei an.
DSA (für DNSSEC oder SIG(0)):	003	512–1024 und ein Vielfaches von 64	siehe RSAMD5
RSASHA1 (für DNSSEC oder SIG(0)):	005	512–2048	siehe RSAMD5
HMAC-MD5 (für TSIG):	157	1–512	Beide Dateien enthalten das Shared secret. Sie können dieses in eine key-Anweisung Ihrer BIND-Konfigurationsdatei kopieren und den Dateinamensstamm beim Aufruf von nsupdate oder rndc angeben.

Tabelle 12.1: Inhalt der von dnssec-keygen erstellten Dateien (Fortsetzung)

Aufruf

```
dnssec-keygen
   -a Algorithmus -b Schlüssellänge -n Typ
   [weitere Optionen]
   SchlüsselName
```

Listing 12.1: Aufruf von dnssec-keygen

Als *SchlüsselName* sollten Sie

- bei TSIG, TKEY und SIG(0): den Hostnamen Ihres Nameservers
- bei DNSSEC: den Namen Ihrer Zone

angeben.

Beschreibung der Optionen:

Option	Beschreibung
! `-a ( RSAMD5 | RSA | RSASHA1 | DSA | DH | HMAC-MD5 )`	bestimmt das Format des Schlüsselpaars. Eine Übersicht der Algorithmen finden Sie in Tabelle 12.1. Bei `HMAC-MD5` und `DH` aktiviert sich automatisch die `-k`-Option.
! `-b` *`Schlüssellänge`*	bestimmt die Länge des Schlüssels. Die Mindest- und Maximal-Länge hängen vom Algorithmus ab und werden in Tabelle 12.1 beschrieben.
`-e`	bewirkt, dass `dnssec-keygen` beim Erstellen von RSA-Schlüsseln einen größeren Exponenten verwendet.
`-g (2|5)`	verwendet 2 oder 5 als Primzahl bei der Erstellung von DH-Schlüsseln. **Standardwert:** Eine sehr lange Primzahl aus RFC2539.
`-h`	gibt eine kurze Übersicht aller Optionen von `dnssec-keygen` aus.
`-r ( keyboard | /dev/random | /dev/urandom )`	wählt einen Zufallszahlengenerator aus. Unter Linux blockiert `/dev/random` bei leerem Pool, während `/dev/urandom` dann theoretisch berechenbare Zahlenfolgen liefert. `keyboard` wertet Ihre Tastendrücke aus. **Standardwert:** `/dev/random`
`-s` *`Stärke`*	gibt die *Stärke* des Schlüssels an: ein Wert zwischen 0 und 15, der jedoch zurzeit noch keinerlei Auswirkung hat.
`-v` *`DebugLevel`*	aktiviert die Ausgabe zusätzlicher Meldungen.

Tabelle 12.2: `dnssec-keygen`: Allgemeine Optionen

Mit den folgenden Optionen bestimmen Sie die Felder des KEY- oder DNSKEY-Records in der `.key`-Datei (für SIG(0) und DNSSEC):

Option	Beschreibung
`-c Klasse`	gibt die Klasse des Records an. **Standardwert:** `IN`
`-f KSK`	erstellt einen Key-Signing-Key (KSK).
`-k`	bewirkt, dass `dnssec-keygen` KEY-Records statt DNSKEY-Records in die `key`-Datei schreibt. Erstellen Sie KEY-Records für Hosts (z.B. für SIG(0)) und DNSKEY-Records für Zonen (für DNSSEC).
`-n ( ZONE | HOST | ENTITY | USER | OTHER )`	gibt den Namenstyp an, der in das Flags-Feld des Records einfließt. Er bestimmt, ob sich dieser Schlüssel auf eine Zone, einen Host (= Entity), einen User oder etwas anderes bezieht. **Standardwert:** `256` (d.h. Zone)
`-p ProtokollNr`	gibt den Wert für das Protokoll-Feld an. Mögliche Werte: 1: TLS, 2: E-Mail, 3: DNSSEC, 4: IPSEC, 255: universell. **Standardwert:** 3 (d.h. DNSSEC)
`-t ( AUTHCONF | NOAUTHCONF | NOAUTH | NOCONF )`	bestimmt den Wert des Verwendungsfeldes. Dieses gibt an, ob der Schlüssel - zum Signieren (AUTHentication) oder - zur Verschlüsselung (CONFidentiality) eingesetzt werden darf. **Standardwert:** `0` (d.h. `AUTHCONF`)

Tabelle 12.3: `dnssec-keygen`-Optionen zur `.key`-Datei

Beispiel

Der folgende Aufruf erzeugt einen 768 Bit langen DSA-Schlüssel für die Zone `example.com`:

```
$ dnssec-keygen -a DSA -b 768 -n ZONE example.com
Kexample.com.+003+26160
$
```

Listing 12.2: Beispiel zu `dnssec-keygen` (aus der Manpage)

Die Datei `Kexample.com.+003+26160.private` enthält den privaten Schlüssel und die Datei `Kexample.com.+003+26160.key` den öffentlichen Schlüssel in Form eines DNSKEY-Records.

12.4.2 dnssec-signzone

`dnssec-signzone` signiert eine Zone mit Ihrem privaten Schlüssel, indem automatisch NSEC-, RRSIG- sowie ggf. DS- und DLV-Records hinzugefügt werden.

Aufruf

`dnssec-signzone [`*`Optionen`*`] `*`Zonefile Schlüssel...`*

Beschreibung der Optionen:

Option	Beschreibung
`-a`	prüft alle erstellten Signaturen.
`-c` *`Klasse`*	gibt explizit die Klasse der hinzugefügten Records an. **Standardwert:** `IN`
`-d` *`Verzeichnis`*	liest die Schlüssel-Dateien aus dem angegebenen *Verzeichnis*.
`-e (YYYYMMDDHHMMSS | +Sekunden )`	gibt das Verfallsdatum des RRSIG-Records an, entweder absolut oder relativ zum Beginn der Gültigkeit. **Standardwert:** `+2592000` (d.h. 30 Tage nach der Startzeit)
`-f` *`Dateiname`*	gibt den Namen der Ausgabedatei an, in die `dnssec-signzone` das signierte Zonefile schreibt. Standardwert: *`Zonefile`*`.signed`
`-g`	bewirkt, dass `dnssec-signzone` DS-Records für die untergeordneten Zonen erstellt. Dies setzt voraus, dass die entsprechenden Keyset-Dateien bereitstehen. Vorhandene DS-Records werden ersetzt.
`-h`	gibt eine kurze Übersicht aller Optionen von `dnssec-signzone` aus.

Tabelle 12.4: `dnssec-signzone`-Optionen

Option	Beschreibung
`-i` *Sekunden*	bestimmt die Frist, innerhalb der ein RRSIG-Record ablaufen muss, damit `dnssec-signzone` ihn erneuert (d.h. das Recordset neu signiert). **Standardwert:** Ein Viertel der Gültigkeitsdauer, die sich aus der Differenz zwischen Start- und Endezeit berechnet. **Beispiel:** Wenn eine Signatur 30 Tage gültig ist (der Standardwert, wenn man keine Start- und Endezeit angibt), erstellt `dnssec-signzone` sie neu, wenn sie innerhalb der nächsten 7,5 Tage abläuft.
`-k key`	behandelt den angegebenen Schlüssel als Key Signing Key (KSK), unabhängig von seinen Flags. Sie können diese Option mehrfach angeben, wenn Sie mehrere derartige Schlüssel haben.
`-l` *Domain*	bewirkt, dass `dnsssec-signzone` zusätzlich DLV-Records erstellt.
`-n` *Threads*	gibt an, wie viele Threads `dnssec-signzone` erstellen soll. Diese Zahl sollte mit der Zahl der CPUs in Ihrem Rechner übereinstimmen. **Standardwert:** Die Zahl der CPUs (falls erkennbar)
`-o Zone`	gibt den Namen der Zone an. **Standardwert:** Der Name des Zonefiles
`-p`	verwendet einen schnelleren, aber schlechteren Zufallszahlengenerator. Geben Sie entweder die `-p`- oder die `-r`-Option an, aber nicht beide.

`-r`
`( keyboard |`
`/dev/random |`
`/dev/urandom |`
`Datei )`
	wählt einen Zufallszahlengenerator aus.
	Unter Linux blockiert `/dev/random` bei leerem Pool, während `/dev/urandom` dann theoretisch berechenbare Zahlenfolgen liefert.
	Die *Datei* kann vorbereitete Zufallsfolgen enthalten.
	`keyboard` wertet Ihre Tastendrücke aus.
	Standardwert: `/dev/random`

Tabelle 12.4: `dnssec-signzone`-Optionen (Fortsetzung)

Option	Beschreibung
-s (YYYYMMDDHHMMSS | +Sekunden)	gibt den Gültigkeitsbeginn der RRSIG-Records an, entweder absolut oder relativ zum aktuellen Zeitpunkt. **Standardwert:** vor genau einer Stunde
-t	gibt eine statistische Zusammenfassung über die signierten Records aus.
-v DebugLevel	aktiviert die Ausgabe zusätzlicher Meldungen.
-z	ignoriert das Key-Signing-Key-(KSK-)Flag bei der Auswahl der zu signierenden Schlüssel.

Tabelle 12.4: dnssec-signzone-Optionen (Fortsetzung)

Beispiel

Der folgende Aufruf signiert das Zonefile db.example.com mit dem Schlüssel aus der Datei Kexample.com.+003+26160.private:

```
$ dnssec-signzone \
    -o example.com  \
    db.example.com  \
    Kexample.com.+003+26160
```

Listing 12.3: Beispiel zu dnssec-signzone (aus der Manpage)

Dabei entsteht ein signiertes Zonefile mit dem Namen db.example.com.signed, das Sie anstatt des ursprünglichen (unsignierten) Zonefiles in der zone-Anweisung der named.conf-Datei angeben können.

12.4.3 dnssec-makekeyset

dnssec-makekeyset signiert den DNSKEY-Record einer Zone und schreibt ihn mitsamt seiner Signatur in eine Keyset-Datei.

Senden Sie die Keyset-Datei an den Hostmaster Ihrer übergeordneten Zone (z.B. den Registrar Ihrer TLD), der daraus (z.B. mit dnssec-signzone oder dnssec-signkey) die DS- und RRSIG-(DS-)Records berechnet und seiner Zone hinzufügt.

Achtung: Gefälschte Keyset-Dateien sind eine Schwachstelle des DNSSEC. Wenn Sie Ihre Keyset-Datei per E-Mail verschicken, sollten Sie diese E-Mail mit PGP o.Ä. signieren. Vereinbaren Sie mit dem Empfänger, dass er nur signierte E-Mails von Ihnen bearbeitet.

Aufruf

```
dnssec-makekeyset
    [-a] [-h] [-t TTL] [-v DebugLevel]
    [-s Startzeit] [-e Endezeit]
    [-p] [-r Zufallszahlengenerator]
    SchlüsselDateinamensStamm ...
```

Listing 12.4: Aufruf von `dnssec-makekeyset`

Beschreibung der Optionen:

Option	Beschreibung
`-a`	prüft alle erstellten Signaturen.
! `-e` (*YYYYMMDDHHMMSS* | `+`*Sekunden* | `now+`*Sekunden*)	gibt das Verfallsdatum des RRSIG-(DNSKEY-)Records an, entweder absolut oder relativ zum Beginn der Gültigkeit oder dem aktuellen Zeitpunkt. **Standardwert:** `+2592000` (d.h. 30 Tage nach der Startzeit)
`-h`	gibt eine kurze Übersicht aller Optionen von `dnssec-makekeyset` aus.
`-p`	verwendet einen schnelleren, aber schlechteren Zufallszahlengenerator. Geben Sie entweder die `-p`- oder die `-r`-Option an, aber nicht beide.

Tabelle 12.5: `dnssec-makekeyset`-Optionen

Option	Beschreibung
-r (keyboard | /dev/random | /dev/urandom)	wählt einen Zufallszahlengenerator aus. Unter Linux blockiert /dev/random bei leerem Pool, während /dev/urandom dann theoretisch berechenbare Zahlenfolgen liefert. keyboard wertet Ihre Tastendrücke aus. **Standardwert:** /dev/random
-s (*YYYYMMDDHHMMSS* | *+Sekunden*)	gibt den Gültigkeitsbeginn des RRSIG-(DNSKEY-)Records an, entweder absolut oder relativ zum aktuellen Zeitpunkt. **Standardwert:** +0 (d.h. jetzt)
-t *TTL*	gibt die TTL des RRSIG-(DNSKEY-)Records an. **Standardwert:** 3600 (d.h. eine Stunde)
-v *DebugLevel*	aktiviert die Ausgabe zusätzlicher Meldungen.

Tabelle 12.5: dnssec-makekeyset-Optionen (Fortsetzung)

Beispiel

Der folgende Aufruf signiert den DSA-Schlüssel der Zone example.com (aus der Datei Kexample.com.+003+26160.private) mit sich selbst und erstellt die Datei keyset-example.com.:

```
$ dnssec-makekeyset -t 86400      \
   -s 20000701120000 -e +2592000  \
   Kexample.com.+003+26160
```

Listing 12.5: Beispiel zu dnssec-makekeyset (aus der Manpage)

12.4.4 dnssec-signkey

dnssec-signkey liest den signierten DNSKEY-Record aus der Keyset-Datei einer untergeordneten Domain, signiert ihn mit dem angegebenen Schlüssel und schreibt die entstehenden DS- und RRSIG-(DS-)Records in eine Signedkey-Datei.

Diese Signedkey-Datei kann der Hostmaster in sein Zonefile einfügen (z.B. mit der $INCLUDE-Anweisung).

Aufruf

```
dnssec-signkey
    [-a] [-h] [-t TTL] [-v DebugLevel]
    [-s Startzeit] [-e Endezeit]
    [-p] [-r Zufallszahlengenerator]
    KeysetDateiname
    SchlüsselDateinamensStamm ...
```

Listing 12.6: Aufruf von `dnssec-signkey`

Beschreibung der Optionen:

Option	Beschreibung
`-a`	prüft alle erstellten Signaturen.
`-e` (*YYYYMMDDHHMMSS* | +*Sekunden* | now+*Sekunden*)	gibt das Verfallsdatum des RRSIG-(DS-)Records an, entweder absolut oder relativ zum Beginn der Gültigkeit oder dem aktuellen Zeitpunkt. **Standardwert:** `+2592000` (d.h. 30 Tage nach der Startzeit)
`-h`	gibt eine kurze Übersicht aller Optionen von `dnssec-signkey` aus.
`-p`	verwendet einen schnelleren, aber schlechteren Zufallszahlengenerator. Geben Sie entweder die `-p`- oder die `-r`-Option an, aber nicht beide.
`-r` (keyboard | /dev/random | /dev/urandom)	wählt einen Zufallszahlengenerator aus. Unter Linux blockiert `/dev/random` bei leerem Pool, während `/dev/urandom` dann theoretisch berechenbare Zahlenfolgen liefert. `keyboard` wertet Ihre Tastendrücke aus. **Standardwert:** `/dev/random`
`-s` (*YYYYMMDDHHMMSS* | +*Sekunden*)	gibt den Gültigkeitsbeginn des RRSIG-(DS-)Records an, entweder absolut oder relativ zum aktuellen Zeitpunkt. **Standardwert:** `+0` (d.h. jetzt)

Tabelle 12.6: `dnssec-signkey`-Optionen

Option	Beschreibung
`-t TTL`	gibt die TTL der DS- und RRSIG-(DS-)Records an. **Standardwert:** `3600` (d.h. eine Stunde)
`-v DebugLevel`	aktiviert die Ausgabe zusätzlicher Meldungen.

Tabelle 12.6: dnssec-signkey-Optionen (Fortsetzung)

Beispiel

Mit folgendem Aufruf kann der Hostmaster der `.com.`-Zone einen DS-Record für die Zone `example.com.` erstellen.

`dnssec-signkey` schreibt einen DS-Record zu dem in der Datei `keyset-example.com.` enthaltenen, signierten DNSKEY-Record in die Datei `signedkey-example.com.`. Hierzu liest es den privaten DSA-Schlüssel der `com.`-Zone aus der Datei `Kcom.+003+51944.private`:

```
$ dnssec-signkey keyset-example.com. Kcom.+003+51944
```

Listing 12.7: Beispiel zu dnssec-signkey (aus der Manpage)

12.5 BIND härten

Theoretisch besteht die Möglichkeit, dass Hacker über einen Softwarefehler in BIND Anweisungen auf Ihrem Nameserver-Rechner ausführen und z.B. Dateien übertragen oder ändern.

Mit früheren BIND-Versionen bestand diese Möglichkeit auch *praktisch*. Da niemand Ihnen beweisen kann, dass die aktuelle BIND-Version auf Ihrem Betriebssystem und Ihrer Hardware fehlerfrei arbeitet, ist davon auszugehen, dass die aktuelle Version ebenfalls betroffen ist. Daher sollten Sie präventiv Vorsichtsmaßnahmen treffen, die die Auswirkung eines Fehlers begrenzen.

Dieser Abschnitt beschreibt einige Vorsichtsmaßnahmen für Unix-Systeme, die Sie leicht selbst umsetzen können.

12.5.1 Martian addresses ablehnen

Einige IP-Adressen sind im Internet nicht vergeben oder ausgeschlossen worden. Da leider noch viele Internet-Provider Pakete mit diesen Adressen im

Absender weiterleiten, versuchen Hacker, sich mit diesen Adressen an Firewalls vorbeizumogeln. Da man solch einem Paket seine Herkunft nicht ansehen kann, nennt man diese Adressen »Martian addresses« (Adressen vom Mars).

Nichts spricht dafür, dass BIND Anfragen von Martian addresses bearbeitet. Dabei würde vielleicht eine Sicherheitslücke ausgelöst und die Antwort würde ohnehin ihr Ziel nicht erreichen.

Hier ist ein Vorschlag für eine abgesicherte Konfiguration:

```
acl private {      // [RFC1918]:
  10/8;            // private Class A net  (/8)
  172.16/12;       // private Class B nets (/16)
  192.168/16;      // private Class C nets (/24)
};
acl iana-reserved {
  0/8;             // This Network [RFC1700]
  14/8;            // Public Data Networks [RFC1700]
  39/8;            // [RFC1797]
  128.0/16;        // Class A Boundary [RFC3330]
  169.254/16;      // Link Local (DHCP failure...)
  191.255/16;      // Class B Boundary [RFC3330]
  192.0.0/24;      // Class C Boundary [RFC3330]
  192.0.2/24;      // Test-Net
  192.42.172.0/24; // NextStep default
  192.88.99/24;    // 6to4 Relay Anycast [RFC3068]
  192.175.48/24;   // AS112
  198.18/15;       // BMW / Device Benchmark [RFC2544]
  223.255.255/24;  // Class C Boundary [RFC3330]
  !255.255.255.255; 240/4; // class E [RFC1700]
};
acl multicast {
  224/4; // class D [RFC3171]
};
acl broadcast {
  255/8;
```

```
  // fügen Sie hier die Net-directed-broadcast-addresses
  // Ihrer lokalen Segmente hinzu
};
acl loopback {
  127/8; // loopback [RFC1700]
};
acl tarpit { // Class-A networks not allocated by IANA
  // Stand: August 2004,
  // Quelle: http://www.cymru.com/Documents/bogon-list.html
   1/8;   2/8;   5/8;   7/8;  23/8;
  27/8;  31/8;  36/7;  41/8;  42/8;
  49/8;  50/8;  73/8;  84/7;  76/6;
  89/8;  90/7;  92/6;  96/3; 173/8;
 174/7; 176/5; 184/6; 190/8; 197/8;
};

acl martian {
  private;
  multicast; broadcast;
  iana-reserved; tarpit;
};

options {
  // ...
  blackhole {martian;};
};
```

Listing 12.8: Martian addresses ablehnen

12.5.2 Unprivilegierter User

Der `root`-User eines Unix-Systems hat vollen Zugriff auf alle lokalen Dateien und kann aus Changeroot-Umgebungen ausbrechen. Da diese Fähigkeiten für den Betrieb von BIND nicht erforderlich sind, sollten Sie BIND unter einer anderen User-ID betreiben.

Wenn Sie den `named`-Prozess unter einer anderen User-ID *aufrufen* (z.B. mit `sudo`), werden Sie jedoch feststellen, dass er sich sofort beendet, weil er ohne Root-Rechte nicht auf dem UDP-Port 53 lauschen darf.

Die Lösung besteht darin, `named` als `root` mit dem Parameter `-u` *`Username`* aufzurufen. Dann öffnet BIND *erst* alle erforderlichen Ports und Dateien und wechselt *anschließend* seine User-ID zu dem angegebenen User.

Achtung: Sie müssen allerdings sicherstellen, dass der unter diesem User laufende BIND-Prozess auf jeden Fall lesend auf die Konfigurationsdatei und die Zonefiles zugreifen kann. Im Fall von `slave`- und Dynamic-Update-Zonen muss der User sogar die Dateien überschreiben dürfen.

Das erreichen Sie beispielsweise, wenn die Dateien `root` gehören und für die persönliche Gruppe des Users lesbar bzw. schreibbar sind.

Achtung: Betreiben Sie BIND *nicht* unter `nobody`. Richten Sie einen separaten User ein (z.B. `named`), der das Homeverzeichnis `/dev/null`, die Loginshell `/bin/false` und eine eigene Gruppe hat.

Achtung: Wenn Sie mit der `listen-on{}`-Option in der Konfigurationsdatei die Netzschnittstellen explizit angegeben haben, kann BIND ohne Root-Rechte nicht auf Schnittstellen lauschen, die erst im laufenden Betrieb hinzukommen (z.B. ppp-Interfaces).

Mehr zum Thema: Der Aufruf von BIND wird in Abschnitt 11.1 beschrieben.

12.5.3 Changeroot-Umgebung

Das Dateisystem eines Unix-Rechners enthält viele Dateien, die für den Betrieb von BIND *nicht* erforderlich sind. Alle Dateien, die »world-readable« (`rw-r--r--`) sind (z.B. `/etc/passwd`) , könnte BIND sogar als unprivilegierter User lesen.

Mit einer Changeroot-Umgebung können Sie den Zugriff des `named`-Prozesses zumindest auf einen Teilbaum des Dateisystems begrenzen.

Zur Konfiguration der Changeroot-Umgebung sind folgende Schritte erforderlich:

1. Erstellen Sie ein Verzeichnis (z.B. `/var/lib/named`), das für den `named`-Prozess als Root-Verzeichnis (/) dienen wird. Erstellen Sie in dem Verzeichnis die Unterverzeichnisse `dev`, `etc/bind`, `var/run` und `usr/lib/zoneinfo`:

   ```
   $ mkdir /var/lib/named/
   $ cd /var/lib/named/
   $ mkdir -p dev etc var/run usr/lib/
   ```

   Listing 12.9: Verzeichnisse anlegen

2. Verschieben Sie die Konfigurationsdateien, Schlüssel und Zonefiles aus dem `/etc/bind/`-Verzeichnis (oder `/etc/named`) nach `/var/lib/named/etc/bind/`. Linken Sie das Konfigurationsverzeichnis anschließend nach `/etc/` zurück, damit die Hostmaster und Tools wie `rndc` ihre Dateien leichter wiederfinden:

   ```
   $ mv /etc/bind /var/lib/named/etc/
   $ ln -s /var/lib/named/etc/bind /etc/
   ```

   Listing 12.10: Konfigurationsdateien verschieben

3. Kopieren Sie die `localtime`-Datei (je nach Distribution: von `/etc/localtime`, `/etc/TIMEZONE`, `/etc/zoneinfo/localtime` oder `/usr/lib/zoneinfo/localtime`) in das entsprechende Verzeichnis der Changeroot-Umgebung. Wenn Sie diesen Schritt auslassen, gibt BIND in Logfiles die Uhrzeit in Greenwich Mean Time (GMT) an. Dies betrifft jedoch nur Log-Nachrichten, die Sie mit einer `channel{}`-Anweisung an Syslog vorbei schleusen.

4. Legen Sie die Device-Dateien `null`, `zero`, `random` und `urandom` im `dev`-Verzeichnis der Changeroot-Umgebung an. Die Major- und Minor-Nodenumbers müssen mit denen des *echten* `/dev/`-Verzeichnisses übereinstimmen. Unter Linux können Sie die Devices entweder kopieren (`cp -a` oder `rsync -a`) oder von Hand anlegen:

```
$ cd /var/lib/named/
$ mknod c 1 3 dev/null
$ mknod c 1 5 dev/zero
$ mknod c 1 8 dev/random
$ mknod c 1 9 dev/urandom
```

Listing 12.11: Devices anlegen unter Linux

5. Sorgen Sie dafür, dass der User, unter dem Sie den `named`-Prozess betreiben möchten, alle Dateien innerhalb der Changeroot-Umgebung lesen und die Zonefiles im Fall von `slave`- und Dynamic-Update-Zonen auch überschreiben darf, z.B. mit `cd /var/lib/named; chown -R root:named .; chmod -R 640 .; chmod g+w etc/bind/*`.
6. Fügen Sie dem Aufruf von `syslogd` in dem entsprechenden `/etc/init.d`- oder `/etc/rc.config`-Script den Parameter `-a /var/lib/named/dev/log` hinzu, damit der `syslogd` einen weiteren Socket im Zugriffsbereich des Nameservers erstellt.
   Unter Debian GNU/Linux schreiben Sie `SYSLOGD="-a /var/lib/named/dev/log"` an den Anfang der Datei `/etc/init.d/sysklogd`.
7. Stoppen und starten Sie den `syslogd`, damit er den Socket anlegt.
8. Fügen Sie dem Aufruf von `named` in dem entsprechenden `/etc/init.d`- oder `/etc/sysconfig/named`-Script den Parameter `-t /var/lib/named` hinzu.
9. Stoppen und starten Sie den `named`-Prozess und beobachten Sie dabei das Logfile:

```
$ tail -n +0 -f /var/log/{syslog,messages} |grep named &
$ /etc/init.d/bind9 stop      # oder /etc/init.d/named ...
$ /etc/init.d/bind9 start     # ... oder /etc/init.d/bind
```

Listing 12.12: BIND stoppen und starten

10. Testen Sie den Nameserver (z.B. mit `dig`, wie in Abschnitt 4.2 beschrieben).

> **Achtung:** Geben Sie in der Konfigurationsdatei weiterhin `/etc/bind/` an, wenn Sie das Verzeichnis `/var/lib/named/etc/bind/` meinen.

> **Achtung:** Weil `root` aus der Changeroot-Umgebung ausbrechen kann, sollten Sie einen in dieser Umgebung betriebenen BIND immer unter einer anderen User-ID laufen lassen (siehe Abschnitt 12.5.2).

> **Mehr zum Thema:** siehe chroot(2)-Manpage und das Chroot-BIND-Howto auf `http://www.losurs.org/docs/howto/Chroot-BIND.html`.

12.5.4 Tipps

Sie können noch mehr unternehmen, um Ihren Nameserver vor Ausfällen und Spionage zu schützen:

- Halten Sie Ihre BIND-Version aktuell. Die Vergangenheit hat gezeigt, dass neue Fähigkeiten einer Software oft auch neue Sicherheitslücken mit sich bringen, die sich einige Versionen lang halten.

> **Mehr zum Thema:** Die aktuelle BIND-Versionsnummer finden Sie auf `http://www.isc.org/`. Ein Diagramm zur Verweildauer von Sicherheitslücken im BIND-Code finden Sie in Abschnitt 2.1.2. Gehen Sie auf `http://www.isc.org/sw/bind/bind-lists.php`, um die Mailinglist `bind-announce@isc.org` zu abonnieren.

- Übergeben Sie den Nameservice für Ihre Internetzonen an Profis. Das kostet nur ein paar Cent im Monat und spart Ihnen unendliche Schmerzen.

  Mehr zum Thema: Die Hidden-Primary-Konfiguration wird in Abschnitt 8.5 beschrieben.

- Halten Sie Ihre BIND-Version aktuell.
- Betreiben Sie Ihren Nameserver unter einer unprivilegierten User-ID und in einer Changeroot-Umgebung.

  Mehr zum Thema: Siehe Abschnitte 12.5.2 und 12.5.3.

- Vermeiden Sie Single-Point-of-Failures (SPOF): Das DNS ermöglicht Ihnen den Betrieb *mehrerer* Nameserver, Mailserver und Webserver für dieselben Adressen. Kaufen Sie Server-Hardware mit Festplatten-RAIDs und redundanten Netzteilen.

  Mehr zum Thema: Die Konfiguration sekundärer Zonen wird in Abschnitt 6.3.11 und die Lastverteilung mit NS-, A- und MX-Records in Abschnitt 9.2 beschrieben.

- Setzen Sie die Option `interface-interval 0;`, wenn Ihr Nameserver keine dynamischen (z.B. ppp-) Interfaces hat.
- Halten Sie Ihre BIND-Version aktuell.
- Verhindern Sie die Abfrage Ihrer internen Zonen aus dem Internet (z.B. mit `allow-recursion{}`-, `allow-query{}`- und `view{}`-Anweisungen). Noch sicherer ist es, den autoritativen Nameserver Ihrer Internet-Zonen erst gar nicht auf derselben Hardware wie den rekursiven Nameserver für Ihr LAN zu betreiben. Dann können Sie auf den autoritativen Nameservern `recursion no;` setzen.

Mehr zum Thema: Kapitel 8 enthält Konfigurationen rekursiver und autoritativer Nameserver. Kapitel 6 erklärt die genannten Konfigurationsanweisungen.

- Erlauben Sie Zonetransfers und Dynamic Updates nur zwischen Ihren eigenen Nameservern und gegen TSIG-Schlüssel.

Mehr zum Thema: Abschnitt 6.2.3 enthält ein Beispiel für Adresslisten, die einen TSIG-Schlüssel *und* eine IP-Adresse erfordern.

- Verbieten Sie Dynamic Updates auf Ihre Haupt-Zone. Die Windows-2000-Rechner und DHCP-Server können Ihre Updates genauso gut an eine Subdomain senden.

Mehr zum Thema: Die `allow-update{}`- und `update-policy{}`-Anweisungen werden in Abschnitt 6.3.7 beschrieben.

- Vermeiden Sie kurze TSIG-Schlüssel, da diese schneller erraten werden können.
- Verhindern Sie die Abfrage der BIND-Version, der BIND-Autoren und des lokalen Hostnamens, damit Hacker länger brauchen, um die mit Sicherheit in BIND enthaltenen Sicherheitslücken zu finden.

Mehr zum Thema: Abschnitt 9.3 zeigt die in BIND voreingestellte `view{}`-Anweisung, die für diese Records verantwortlich ist. Kopieren Sie sie in Ihre Konfigurationsdatei und fügen Sie `allow-query{none;};` hinzu.

Und vor allem:

- Halten Sie Ihre BIND-Version aktuell.

13 Debugging

13.1 Fehlerkategorien

Verschiedene Fehler können die Verfügbarkeit Ihrer Zonen und der enthaltenen Records beeinträchtigen.

Die häufigsten Fehler sind:

- Syntaxfehler
  - in der Konfigurationsdatei (z.B. vergessene Semikolons)
  - in den Zonefiles (z.B. fehlende Punkte am Ende von FQDNs oder # statt ; als Kommentarzeichen)
- Semantikfehler
  - in der Konfigurationsdatei (z.B. fehlende Slave-Zonen)
  - in den Zonefiles (z.B. fehlende PTR-Records)
- Hardwaredefekte und Netzstörungen.

13.2 Empfehlungen

Zum Schutz vor Syntaxfehlern und Semantikfehlern können Sie

- grafische Admintools einsetzen, die die Konfiguration und Zonefiles für Sie erstellen (siehe Abschnitt 2.2 und 11.4)
- Programme einsetzen, die nach jeder Änderung die gesamte Konfiguration prüfen (siehe unten)

Zum Schutz vor Hardwaredefekten und Netzstörungen können Sie redundante Komponenten in unabhängigen Netzen einsetzen.

13.3 Prüfen der Konfiguration

Es gibt verschiedene Messpunkte, an denen Sie die Konfiguration eines Nameservers prüfen können (siehe Abbildung 13.1).

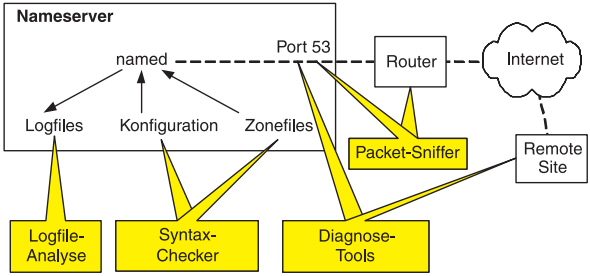

Abbildung 13.1: Ansatzpunkte zur Fehlersuche

Die folgenden Abschnitte dieses Kapitels beschreiben einige der Programme und Methoden, die Sie hierbei einsetzen können.

13.4 Logfile-Analyse

BIND schreibt alle Syntaxfehler, die er beim Einlesen der Konfigurations- und Zonefiles feststellt, in eine Logdatei. Die Logdatei finden Sie z.B. unter `/var/log/syslog`, `/var/log/messages` oder `/var/adm/messages` (abhängig vom Betriebssystem).

Achtung: Allerdings sollten Sie die Syntax prüfen, *bevor* Sie sie in Ihren Nameserver laden. Verwenden Sie hierzu die Syntax-Checker, die im folgenden Abschnitt vorgestellt werden.

13.5 Syntax-Checker

13.5.1 named-checkconf

`named-checkconf` ist Bestandteil der BIND-Distribution und dient zum Untersuchen Ihrer BIND-Konfigurationsdatei und der Zonefiles aller darin referenzierten, primären Zonen auf Syntaxfehler.

Aufruf:

`named-checkconf [-Optionen] [Dateiname]`

Wenn Sie keinen Dateinamen angeben, untersucht `named-checkconf` die Datei `/etc/named.conf`.

Sie können `named-checkconf` mit folgenden Optionen aufrufen:

Option	Auswirkung
`-t Verzeichnis`	führt die Untersuchung in der angegebenen Changeroot-Umgebung durch. Dies ist erforderlich, wenn Sie BIND in einer Changeroot-Umgebung ausführen und die Konfigurationsdatei absolute Pfadangaben enthält.
`-v`	`named-checkconf` gibt seine Versionsbezeichnung aus und beendet sich.
`-z [-j]`	untersucht auch die Zonefiles aller primären Zonen, die in der Konfigurationsdatei referenziert werden. Wenn Sie zusätzlich `-j` angeben, untersucht `named-checkconf` auch die Journal-Files dieser Zonen.

Tabelle 13.1: Optionen zu `named-checkconf`

Wenn `named-checkconf` Syntaxfehler entdeckt, gibt es diese aus und beendet sich mit Errorlevel 1.

13.5.2 named-checkzone

`named-checkzone` ist Bestandteil der BIND-Distribution und untersucht ein Zonefile auf Syntaxfehler.

Aufruf:

`named-checkzone [-Optionen] Zone Zonefile`

Übergeben Sie `named-checkzone` den Namen einer Zone und den vollen Dateinamen eines Zonefiles.

Zusätzlich können Sie folgende Optionen angeben:

Option	Auswirkung
`-c` *Klasse*	gibt die Klasse an, in der sich die angegebene Zone befindet. **Standardwert:** `IN`
`-d`	aktiviert die Ausgabe zusätzlicher Debugging-Meldungen
`-D`	gibt das Zonefile aus.
`-j`	untersucht zusätzlich das Journal-File der Zone (falls es existiert).
`-k (fail|warn|ignore)`	bestimmt das Verhalten bei Domain- oder Hostnamen, die ungültige Zeichen enthalten. **Standardwert:** `warn`
`-n (fail|warn|ignore)`	bewirkt, dass `named-checkzone` prüft, ob alle im Zonefile enthaltenen NS-Records auf gültige A-Records verweisen.
`-o` *Dateiname*	gibt den Dateinamen für das bereinigte Zonefile an.
`-q`	»Quiet-Mode«: `named-checkzone` unterlässt alle Ausgaben, signalisiert aber weiterhin das Untersuchungsergebnis im Errorlevel. Diese Option ist zur Verwendung in Shellscripts vorgesehen.
`-t` *Verzeichnis*	führt die Untersuchung in der angegebenen Changeroot-Umgebung durch. Dies ist erforderlich, wenn das Zonefile eine `$INCLUDE`-Anweisung auf einen absoluten Dateinamen enthält.
`-v`	`named-checkzone` gibt seine Versionsbezeichnung aus und beendet sich.
`-w` *Verzeichnis*	gibt das Verzeichnis an, das `named-checkzone` als Ausgangsverzeichnis für alle relativen `$INCLUDE`-Angaben verwendet. Dies sollte mit der `directory`-Option in `named.conf` übereinstimmen.

Tabelle 13.2: Optionen zu `named-checkconf`

Wenn named-checkzone Syntaxfehler entdeckt, gibt es diese aus und beendet sich mit Errorlevel 1. Wenn es keinen Fehler entdeckt, beendet es sich mit Errorlevel 0. Daher können Sie named-checkzone leicht in Shellscripts einbauen:

```
#!/bin/sh
for file in /home/ftp/incoming/db.*; do
    if named-checkzone $file >/tmp/$$; then
        mv $file /etc/bind/
        rndc reload
    else
        mailx -s "Error in $file" hostmaster </tmp/$$
    fi
    rm /tmp/$$
done
```

Listing 13.1: Zonefiles automatisch prüfen

13.5.3 Weitere Tools

nslint ist ein weit verbreitetes Tool zum Prüfen von Konfigurationsdateien und Zonefiles, das jedoch besser für BIND4 und BIND8 geeignet ist, da es die neuen Konfigurationsanweisungen von BIND9 nicht beherrscht.

13.6 Packet-Sniffer

Wenn Sie ein DNS-Problem feststellen, kann es bei der Suche nach der Ursache sehr hilfreich sein, einen Blick auf die DNS-Pakete zu werfen, die gerade eine Netzschnittstelle Ihres Nameservers oder einer weiteren Netzkomponente passieren.

Dieser Abschnitt stellt vier Tools vor, die die in IP-Paketen enthaltenen DNS-Nachrichten auswerten. Diese Tools können die IP-Pakete entweder

- live auf einer lokalen Netzschnittstelle mitschneiden und optional in eine pcap-Datei sichern oder
- aus einer pcap-Datei laden.

Oft ist es sinnvoll, die Pakete auf dem Nameserver mit tcpdump in eine pcap-Datei zu schreiben und die Datei später auf einer lokalen Workstation mit ethereal oder dnstop zu analysieren.

13.6.1 tcpdump, snoop

`tcpdump` (Linux) und `snoop` (Solaris) geben eine kurze Interpretation der im Mitschnitt enthaltenen Protokolle auf der Konsole aus.

Zum Beispiel würde die im nachstehenden Listing 13.2 gezeigte Nameserver-abfrage

```
$ dig www.ibm.com

; <<>> DiG 9.2.1 <<>> www.ibm.com
;; global options:  printcmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 6749
;; flags: qr rd ra;
;; QUERY: 1, ANSWER: 6, AUTHORITY: 4, ADDITIONAL: 4

;; QUESTION SECTION:
;www.ibm.com.        IN   A

;; ANSWER SECTION:
www.ibm.com.    1800 IN   A   129.42.16.99
www.ibm.com.    1800 IN   A   129.42.17.99
www.ibm.com.    1800 IN   A   129.42.18.99
www.ibm.com.    1800 IN   A   129.42.19.99
www.ibm.com.    1800 IN   A   129.42.20.99
www.ibm.com.    1800 IN   A   129.42.21.99

;; AUTHORITY SECTION:
ibm.com.        600  IN   NS  ns.watson.ibm.com.
ibm.com.        600  IN   NS  ns.almaden.ibm.com.
ibm.com.        600  IN   NS  internet-server.zurich.ibm.com.
ibm.com.        600  IN   NS  ns.austin.ibm.com.
```

```
;; ADDITIONAL SECTION:
ns.austin.ibm.com.              172534  IN  A  192.35.232.34
ns.watson.ibm.com.              600     IN  A  129.34.20.80
ns.almaden.ibm.com.             86400   IN  A  198.4.83.35
internet-server.zurich.ibm.com. 172530  IN  A  195.176.20.204

;; Query time: 223 msec
;; SERVER: 127.0.0.1#53(127.0.0.1)
;; WHEN: Sun Oct 17 14:12:14 2004
;; MSG SIZE  rcvd: 299
```

Listing 13.2: dig

folgende Ausgabe im `tcpdump` hervorrufen:

```
$ tcpdump port 53
tcpdump: listening on eth0
[...]
14:07:44.273879
     localhost.1069 > h.root-servers.net.53:
     28762 A? www.ibm.com. (29) (DF)
14:07:44.444020
     h.root-servers.net.53 > localhost.1069:
     28762- 0/13/13 (461) (DF)
14:07:44.452941
     localhost.53 > h.gtld-servers.net.53:
     53799 A? A.GTLD-SERVERS.NET. (36) (DF)
14:07:44.538477
     h.gtld-servers.net.53 > localhost.53:
     53799- 1/8/8 A a.gtld-servers.net (325) (DF)
14:07:44.541764
```

```
     localhost.1070 > a.gtld-servers.net.53:
     906 A? www.ibm.com. (29) (DF)
14:07:44.700894
     a.gtld-servers.net.53 > localhost.1070:
     906- 0/4/4 (203) (DF)
14:07:44.708920
     localhost.53 > h.gtld-servers.net.53:
     5762 A? internet-server.zurich.ibm.com. (48) (DF)
14:07:44.797631
     h.gtld-servers.net.53 > localhost.53:
     5762- 1/4/4 (215) (DF)
14:07:44.800569
     localhost.1070 > internet-server.zurich.ibm.com.53:
     13026 A? www.ibm.com. (29) (DF)
14:07:44.964708
     localhost.53 > internet-server.zurich.ibm.com.53:
     57103 [1au] PTR? 204.20.176.195.in-addr.arpa. (56) (DF)
14:07:45.714457
     localhost.1072 > ns.almaden.ibm.com.53:
     13026 A? www.ibm.com. (29) (DF)
14:07:45.928828
     ns.almaden.ibm.com.53 > localhost.1072:
     13026*- 6/4/2 A www.ibm.com. (267)
```

Listing 13.3: tcpdump

`tcpdump` oder `snoop` sind im Lieferumfang vieler Unix-Distributionen enthalten.

13.6.2 ethereal

`ethereal` ist ein grafischer Packet-Analyzer, der zwar nicht trivial zu bedienen ist, jedoch sehr weit reichende Filter- und Darstellungsmöglichkeiten bietet.

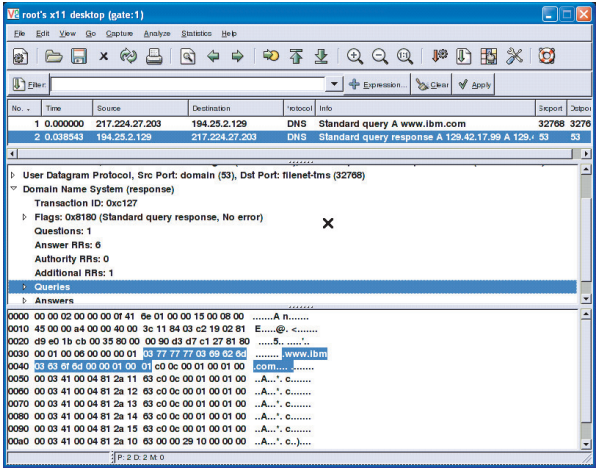

Abbildung 13.2: ethereal

`ethereal` ist Bestandteil vieler Linux-Distributionen und für Solaris auf `http://www.sunfreeware.com/` erhältlich. Es gibt auch eine Windows-Version.

13.6.3 dnstop

`dnstop` zeigt statistische Auswertungen zu den mitgeschnittenen DNS-Paketen an und aktualisiert diese live.

So können Sie z.B. die Anfragen pro

- anfragenden Client
- angefragten Nameserver
- Record-Typ
- Top-Level-Domain
- Second-Level-Domain

beobachten.

`dnstop` ist Bestandteil verschiedener Linux-Distributionen.

Mehr zum Thema: Siehe `http://dnstop.measurement-factory.com/`

13.7 Diagnose-Tools

Es gibt mehrere Programme, die Nameserver *von außen* testen, indem sie DNS-Abfragen an diesen senden. Diese Programme können Semantik- und Konfigurationsfehler feststellen (z.B. Lame-Delegations, fehlende PTR-Records oder fehlende, sekundäre Nameserver) oder die Antwortzeit messen.

13.7.1 dnsdoctor

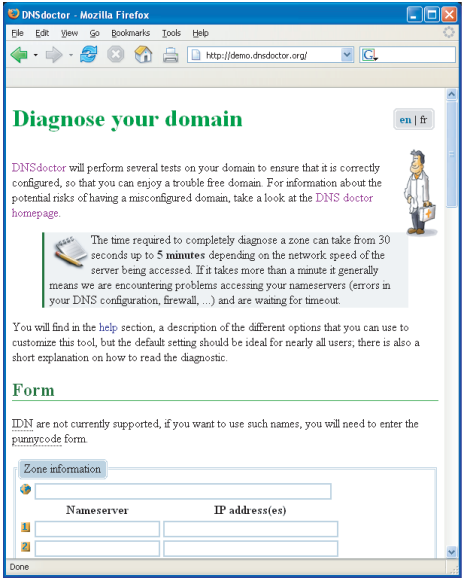

Abbildung 13.3: `dnsdoctor`

dnsdoctor ist ein umfangreiches Diagnose-Tool, das in seiner ersten Version im März 2004 erschien und regelmäßig weiterentwickelt wird.

Sie können dnsdoctor auf der Kommandozeile, als X11-Anwendung und als Webfrontend aufrufen.

Mit dem Webfrontend auf http://demo.dnsdoctor.org/ können Sie Ihre Nameserver über das Internet testen, ohne die Software vorher selbst installieren zu müssen.

dnsdoctor ist Bestandteil verschiedener Linux-Distributionen.

Mehr zum Thema: Siehe http://www.dnsdoctor.org

13.7.2 Weitere Tools

Es gibt viele weitere Tools, die einen Nameserver von außen testen.

Zum Beispiel:

- dlint stellt fehlende PTR- und A-Records fest.
- mrtg und smokeping können die Antwortzeit eines Nameservers regelmäßig messen und in vielen bunten Diagrammen (z.B. auf einer Webseite) grafisch darstellen. Diese winzigen Bilder, bei denen man sehr genau hinsehen muss, um etwas zu erkennen, werden oft als »Hostmaster-Porno« bezeichnet.
- dnstracer zeigt zu jeder DNS-Abfrage die Namen und Beziehungen aller beteiligten Nameserver an.

13.8 Support

Verschiedene Firmen (z.B. das ISC selbst) bieten kommerziellen Support für BIND an.

Darüber hinaus können Sie bei Problemen natürlich googlen oder versuchen, über die Mailinglist bind-users@isc.org kostenlos Hilfe von der Community zu erhalten.

Mehr zum Thema: Anmeldeformulare für die BIND-Mailinglists finden Sie auf `http://www.isc.org/sw/bind/bind-lists.php`. Ein Online-Archiv der Mailinglists finden Sie auf `http://www.isc.org/ops/lists/`.

Teil IV

Anhänge

A Geschichte und Verwaltung des Internets

Viele Organisationen sind »für das Internet zuständig«. Wer ihre Geschichte und Organisation kennt, kann sie leichter auseinander halten.

A.1 Die Geschichte des Internets

1958: Das US-Verteidigungsministerium gründet die Department of Defense Advanced Research Projects Agency (DARPA), die den technologischen und militärischen Vorsprung der USA vor dem Osten sichern soll. Die in der DARPA beschäftigten Wissenschaftler suchen dabei insbesondere nach Anwendungsgebieten für die damaligen Rechenanlagen.

1969: Die DARPA startet das ARPANET-Projekt, das die Möglichkeiten und Vorteile von Computernetzen analysieren soll. Zunächst werden vier Rechner an amerikanischen Forschungszentren über Telefonleitungen und das Network Control Protocol (NCP) miteinander verbunden.

1971: Peggy Karp schlägt in RFC226 vor, die Hostnummern durch globale Hostnamen zu ersetzen, die bis zu sechs Buchstaben lang sein können.

Die Erfindung der E-Mail und des Telnet-Protocol im Jahr **1972** sowie des File-Transfer-Protocol (FTP) im Jahr **1973** sorgen für eine rasche Ausbreitung des Netzes auf weitere amerikanische Universitäten:

Jahr	Hosts
1969	4
1971	15
1972	30
1974	60
1976	100

Tabelle A.1: Zahl der NCP-Hosts im ARPANET

1974: M. D. Kudlick definiert in RFC602 das Format der HOSTS.TXT-Datei.

1981: David Mills schlägt in RFC799 die Einführung von Domains vor, um den Namensraum für Hostnamen zu erweitern.

1982: Die DARPA erklärt den TCP/IP-Protokollstack, der **1974** erstmalig definiert wurde und aufgrund seiner Routingfähigkeit und austauschbaren Schichten dem NCP technologisch überlegen ist, zum offiziellen Standard. Alle Rechner im ARPANET, die noch NCP sprechen, müssen zum 1. Januar **1983** auf TCP/IP wechseln. Das ARPANET wird fortan Internet genannt und breitet sich nun noch schneller aus:

Jahr	Hosts
1982	250
1983	500
1984	1.000
1985	2.000
1989	100.000

Tabelle A.2: Zahl der TCP/IP-Hosts im ARPANET

1983: Paul Mockapetris erfindet das Domain Name System (DNS), wie in Kapitel 1 beschrieben. Das Network Information Center SRI-NIC am Stanford Research Institute verwaltet alle Top- und Second-Level-Domains.

1986: Die amerikanische National Science Foundation (NSF) gründet das NSFNET, das schnellere Leitungen als das ARPANET nutzt und den Aufbau regionaler Netze und den Anschluss weiterer Universitäten fördert. Mit der Hilfe von Merit Network Inc., MCI und IBM kann die NSF das Netz in den folgenden Jahren immer weiter beschleunigen und erweitern.

Viele Institutionen beschäftigen sich nun mit der Verwaltung von IP-Adressen, Portnummern, Domainnamen und der Dokumentation von Standards. Besonders hervorzuheben sind die Gründung der

- Internet Engineering Task Force (IETF) durch das Internet Architecture Board (IAB).

- Internet Assigned Numbers Authority (IANA) unter der Leitung von Jon Postel. Die IANA ist an der University of Southern California (USC) ansässig und für die Eindeutigkeit von Domainnamen, IP-Adressen und Portnummern zuständig.

1989: Die DARPA verkündet offiziell das Ende seines Projekts und löst sich auf.

1990: Der Schweizer Tim Berners-Lee erfindet das World Wide Web (WWW). Das Konzept (»Öffentliche Dateien haben Adressen im Netz und sind über Verweise miteinander verbunden«) vereinfacht die Nutzung und beschleunigt die Ausbreitung des Internets:

Jahr	Hosts
1991	500.000
1992	1.000.000
1993	2.000.000
1994	4.000.000

Tabelle A.3: Zahl der Hosts im Internet

1991: Die Defense Information Systems Agency (DISA) überträgt die Verwaltung der IP-Adressen und Domains vom SRI zunächst an SRI International und dann an die Firma Government Systems Inc. (GSI), die die Aufgaben an Network Solutions Inc. (NSI) outsourct [RFC1261].

1992: Die verschiedenen Verwaltungsinstitutionen schließen sich unter der Dachorganisation Internet Society (ISOC) zusammen, die sich über Spenden finanziert.

1993: Die NSF stimmt zu, dass NSI für fünf Jahre die Registry der `.com`/`.net`/`.org`-Domains übernimmt und gründet mit AT&T und General Atomics das Internet Network Information Center (InterNIC). Das InterNIC bietet über die Webseite `http://www.internic.net/` und per telnet/finger/whois verschiedene Informations- und Verzeichnisdienste an und ermöglicht die Registrierung von `.com`/`.net`/`.org`/`.gov`/`.edu`-Domains [RFC1400].

1994: Tim Berners-Lee gründet das World Wide Web Consortium (W3C), das sich um die Weiterentwicklung des WWW kümmert.

1995: NSI wird von der Science Applications International Corporation (SAIC) übernommen.

Das InterNIC führt ein Gebührensystem für den Unterhalt von .com/.net/.org-Domains ein, scheint aber dem Ansturm der weltweiten Kundschaft nicht gewachsen zu sein.

1996: Die IANA stellt fest, dass die Monopolstellung der NSI bei der Vergabe von IP-Adressen und .com/.net/.org-Domains schädlich sei. Die Beschränkungen für .com (Unternehmen), .net (Provider) und .org (Vereine) werden aufgehoben. In RFC2050 beschreiben NSI und IANA das System der Regional Internet Registries (RIR). Die NSI soll ab 1998 mehreren wetteifernden Registraren den Eintrag von .com/.net/.org-Domains ermöglichen.

1997: Der Hacker Eugene Kashpureff (AlterNIC) nutzt einen Fehler in BIND, um alle weltweiten DNS-Abfragen auf seine eigenen Rechner umzuleiten. Ein weiterer Fehler durch menschliches Versagen bei NSI setzt alle .com/.net/.org-Domains für einen Tag außer Gefecht. Nach diesen schweren Störfällen ordnet US-Präsident Bill Clinton im »Framework for Global Electronic Commerce« die Privatisierung der IP- und Domainvergabe an.

1998: Auf Vorschlag von Jon Postel legt die US-Regierung die Verantwortung für die Verwaltung des Netzes in die Hände der Internet Corporation for Assigned Names and Numbers (ICANN). Die ICANN ist eine nicht-kommerzielle Organisation, die von einem internationalen Gremium geführt wird und neben den Aufgaben der IANA auch neue gTLDs einführen darf.

1999: Die ICANN führt das Shared Registry System (SRS) ein, das den Betrieb mehrerer .com/.net/.org-Registrare ermöglicht. Bis zum Ende des Jahres melden sich etwa 100 Registrare an. Die ICANN vereinbart mit NSI, dass NSI die Registry für .com/.net bleibt.

2000: Die amerikanische Sicherheitsfirma VeriSign übernimmt NSI für 21 Milliarden US$.

2003: Die ICANN überträgt die .org-Registry von VeriSign an die Public Interest Registry (PIR). VeriSign bleibt Registry für die .com/.net-Domains.

7/2004: Das Internet umfasst mehr als 285 Millionen Rechner und mehr als 40 Millionen Domains in den gTLDs.

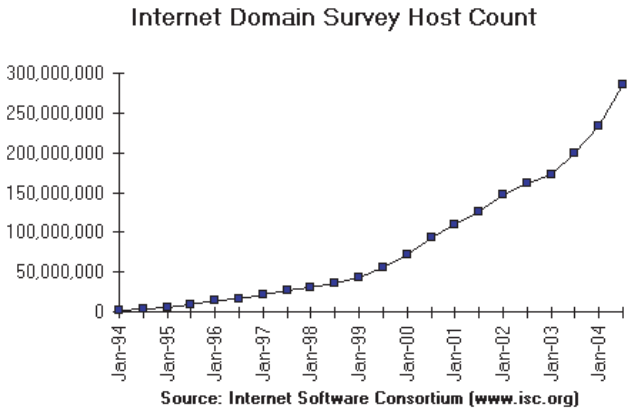

Abbildung A.1: Zahl der Hosts im Internet

Mehr zum Thema:

Die Geschichte des Internets ist auf http://www.zakon.org/robert/internet/timeline/ und http://www.isoc.org/internet/history/brief.shtml detailliert beschrieben.

Eine Übersicht aller RFCs, die das DNS betreffen, finden Sie in Anhang C.

Aktuelle Statistiken zur Verbreitung des Internets finden Sie auf http://www.zooknic.com/ und http://www.nw.com/.

A.2 Die ICANN

Die Internet Corporation for Assigned Names and Numbers (ICANN) steuert viele der Organisationen, die das Internet weiterentwickeln.

A.2.1 Aufbau

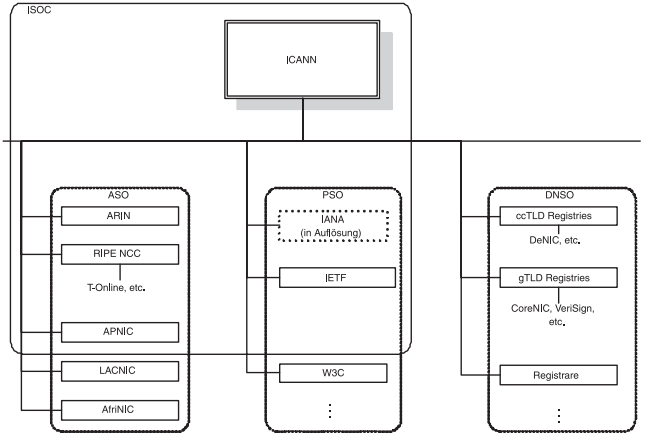

Abbildung A.2: Aufbau der ICANN (vereinfacht)

Die ICANN und einige ihrer Organe sind Mitglieder der **Internet Society** (ISOC), die sich und ihre Mitglieder über Spenden finanziert.

Die folgenden Organisationen sind der ICANN untergeordnet:

- Die **Address Supporting Organization** (ASO), die **Domain Name Supporting Organization** (DNSO), die **Protocol Supporting Organization** (PSO) sind Organisationseinheiten der ICANN.
- Die **Internet Assigned Numbers Authority** (IANA) ist die Vorgängerorganisation der ICANN und seit dem Tod von Jon Postel 1998 nicht mehr aktiv.

- Die **Internet Engineering Task Force** (IETF) betreibt öffentliche Mailinglisten zu technischen Problemen des Internets und ist Herausgeber der RFCs. Jeder kann sich beteiligen.
- Das **World Wide Web Consortium** (W3C) empfiehlt Standards für das Web, z.B. XHTML.

A.2.2 Aufgaben

Die ICANN hat die Hoheit über

- DNS-Top-Level-Domains, TCP-Portnummern und SNMP-MIBs – zurzeit delegiert an die Internet Asigned Numbers Authority (IANA)
- DNS-Domains – zurzeit delegiert an die verschiedenen Registries (siehe Anhang B) und deren Registrare
- die DNS-Root-Nameserver – zurzeit delegiert an verschiedene Rechenzentren
- IP-Adressen – zurzeit delegiert an die Regional Internet Registries (RIR), die im nachfolgenden Abschnitt beschrieben werden.

A.3 Die RIR

Die Regional Internet Registries (RIR) verteilen die IP-Adressen, die sie für die ICANN verwalten, an die Internetprovider (Local Internet Registries (LIR)). Einige RIRs bedienen sich dabei der Hilfe von National Internet Registries (NIR).

Mit der zunehmenden Verbreitung des Internets kam die Einsicht, dass die IP-Adressen eines Tages ausgehen werden. Nach einer großen Neuverteilung vieler IP-Adressen hat die IANA festgelegt, dass die RIRs zukünftig keinem LIR *mehr* IP-Adressen zuteilen dürfen, als er benötigt. Die LIRs erhalten eine gewisse Menge an IP-Adressen (»Allocation«) und ein »Assignment-Window« (z.B. /28), das angibt, bis zu welcher Größe sie Netze ohne Rücksprache selbst verteilen dürfen. Größere Netze erfordern die Genehmigung der RIR. Wenn alle Netze wie beantragt genutzt werden, kann die RIR das Assignment-Window vergrößern. Ansonsten wird sie es verkleinern.

Zurzeit sind die folgenden RIRs im Einsatz:

- Für Europa: Das Réseaux IP Européens Network Coordination Center (RIPE NCC)

- Für Nordamerika: Die American Registry for Internet Numbers (ARIN)
- Für Asien und Australien: Das Asia Pacific Network Information Centre (APNIC)
- Für Südamerika: Die Latin American and Caribbean IP Address Regional Registry (LACNIC)
- Für Afrika: Das African Network Information Center (AfriNIC)

A.4 Die Root-Nameserver

Die Root-Nameserver sind hauptverantwortlich für die Root-Zone (.), die die Delegationen (NS-Records) aller TLDs und einiger weiterer Domains enthält.

Die von NSI auf `ftp://ftp.internic.net/domain/named.root` veröffentlichte Liste aller Root-Nameserver wird von jedem Internet-Nameserver benötigt.

Der *primäre* Nameserver der Root-Zone heißt `a.root-servers.net` und steht bei der Firma VeriSign in Virginia (USA).

Die *sekundären* Nameserver sind unter `b.root-servers.net` bis `m.root-servers.net` erreichbar und werden von zwölf Organisationen betrieben, von denen

- neun in Nordamerika
- zwei in Europa
- eine in Asien

ihren Hauptsitz haben.

Die 13 Betreiber und die Standorte der Root-Nameserver sind auf `http://www.root-servers.org/` wie folgt angegeben (Stand: Oktober 2004):

Gruppe	Betreiber	Server-Standorte
A	VeriSign Global Registry Services	Dulles VA
B	Information Sciences Institute	Marina Del Rey CA
C	Cogent Communications	Herndon VA; Los Angeles; New York City; Chicago

Tabelle A.4: Betreiber und Standorte der Root-Nameserver

Gruppe	Betreiber	Server-Standorte
D	University of Maryland	College Park MD
E	NASA Ames Research Center	Mountain View CA
F	Internet Systems Consortium, Inc.	Ottawa; Palo Alto; San Jose CA; New York City; San Francisco; Madrid; Hong Kong; Los Angeles; Rome; Auckland; Sao Paulo; Beijing; Seoul; Moscow; Taipei; Dubai; Paris; Singapore; Brisbane; Toronto; Monterrey; Lisbon; Johannesburg; Tel Aviv; Jakarta; Munich
G	U.S. DOD Network Information Center	Vienna VA
H	U.S. Army Research Lab	Aberdeen MD
I	Autonomica/NORDUnet	Stockholm; Helsinki; Milan; London; Geneva; Amsterdam; Oslo; Bangkok; Hong Kong
J	VeriSign Global Registry Services	Dulles VA; Mountain View CA; Seattle WA; Amsterdam; Atlanta GA; Los Angeles CA; Miami; Stockholm; London; Tokyo; Seoul; Singapore; Sterling VA (2 locations, standby)
K	Reséaux IP Européens - Network Coordination Centre	London; Amsterdam; Frankfurt; Athens; Doha; Milan
L	Internet Corporation for Assigned Names and Numbers	Los Angeles
M	WIDE Project	Tokio; Seoul; Paris

Tabelle A.4: Betreiber und Standorte der Root-Nameserver (Fortsetzung)

Zu jedem dieser Betreiber gehört *eine* IP-Adresse und *eine* Autonomous System Number (ASN), anhand der die Router den Weg zum nächstgelegenen Server finden.

Mehr zum Thema: Eine Weltkarte, auf der die Niederlassungen der 13 Betreiber eingezeichnet sind, gibt es auf
`http://www.icann.org/correspondence/root-map.gif`.

B Top-Level-Domains

B.1 gTLDs

Die »generic Top-Level-Domains« (gTLDs) bestehen aus den Domains

- der USA (`mil`, `gov`, `edu`) und
- internationaler Gruppen.

Zurzeit (Juni 2004) sind die folgenden 15 gTLDs im Einsatz:

Domain	vorgesehen für	Registry
`.aero`	Luftfahrtunternehmen	Société Internationale de Télécommunications Aéronautiques (SITA)
`.arpa`	Address Routing Parameter Area	Internet Assigned Numbers Authority (IANA) und deren beauftragte RIRs, NIRs und LIRs. Details zur `arpa`- und insbesondere der `in-addr.arpa`-Domain finden Sie in Kapitel 1.
`.biz`	Unternehmen	Neulevel Inc.
`.com`	alles	VeriSign Global Registry Services
`.coop`	Vereine, Genossenschaften	Dot Cooperation LLC
`.edu`	Universitäten der USA	Educause
`.gov`	Politische Einrichtungen der USA	US General Services Administration
`.info`	alles	Afilias Limited
`.int`	Internationale Organisationen (z.B. das Rote Kreuz, die Europäische Union, Sita Equant)	IANA .int Domain Registry

Tabelle B.1: gTLDs (Stand: Mai 2004)

Domain	vorgesehen für	Registry
.mil	Militärische Einrichtungen der USA	US DoD Network Information Center
.museum	Museen	Museum Domain Management Association
.name	Natürliche Personen	Global Name Registry
.net	alles	VeriSign Global Registry Services
.org	alles	Public Interest Registry
.pro	Freiberufler (Ärzte, Anwälte etc.)	RegistryPro

Tabelle B.1: gTLDs (Stand: Mai 2004) (Fortsetzung)

Mehr zum Thema: Die Internet Assigned Numbers Authority (IANA) pflegt diese Liste und veröffentlicht sie auf http://www.iana.org/gtld/gtld.htm.

B.2 ccTLDs

Die Country-Code-Top-Level-Domains (ccTLDs) bezeichnen Länder. Die Country-Codes sind zwei Buchstaben lang und wurden 1981 in ISO3166 festgelegt.

Zurzeit (Juni 2004) sind die folgenden 246 ccTLDs im Einsatz:

Domain	Land
.ac	Ascension
.ad	Andorra
.ae	Vereinigte Arabische Emirate
.af	Afghanistan
.ag	Antigua und Barbuda

Tabelle B.2: ccTLDs (Stand: Mai 2004)

Domain	Land
.ai	Anguilla
.al	Albanien
.am	Armenien
.an	Niederländische Antillen (die Inseln Curaçao und Bonaire)
.ao	Angola
.aq	Antarktis
.ar	Argentinien
.as	Amerik. Samoa
.at	Österreich
.au	Australien
.aw	Aruba
.ax	Aland-Inseln (bei Finnland)
.az	Aserbaidschan
.ba	Bosnien-Herzegowina
.bb	Barbados
.bd	Bangladesch
.be	Belgien
.bf	Burkina Faso
.bg	Bulgarien
.bh	Bahrain
.bi	Burundi
.bj	Benin
.bm	Bermuda-Inseln
.bn	Brunei
.bo	Bolivien

Tabelle B.2: ccTLDs (Stand: Mai 2004) (Fortsetzung)

Domain	Land
.br	Brasilien
.bs	Bahamas
.bt	Bhutan
.bv	Bouvet-Insel (eine unbewohnte, zu Norwegen zählende Insel zwischen Südafrika und der Antarktis)
.bw	Botswana
.by	Weißrussland
.bz	Belize
.ca	Kanada
.cc	Cocos-Inseln
.cd	Kongo (Demokratische Republik, früher: Französisch-Kongo)
.cf	Zentralafrikanische Republik
.cg	Republik Kongo (früher: Zaire)
.ch	Schweiz
.ci	Elfenbeinküste
.ck	Cook-Inseln
.cl	Chile
.cm	Kamerun
.cn	China
.co	Kolumbien
.cr	Costa Rica
.cu	Kuba
.cv	Kap Verde
.cx	Die Weihnachtsinsel
.cy	Zypern
.cz	Tschechische Republik

Tabelle B.2: ccTLDs (Stand: Mai 2004) (Fortsetzung)

Domain	Land
.de	Deutschland
.dj	Dschibuti
.dk	Dänemark
.dm	Dominica
.do	Dominikanische Republik
.dz	Algerien
.ec	Ecuador
.ee	Estland
.eg	Ägypten
.eh	Westsahara
.er	Eritrea
.es	Spanien
.et	Äthiopien
.fi	Finnland
.fj	Fidschi-Inseln
.fk	Falklandinseln
.fm	Mikronesien
.fo	Färöer-Inseln
.fr	Frankreich
.ga	Gabun
.gb	England, Schottland und Wales (Großbritannien)
.gd	Grenada
.ge	Georgien
.gf	Französisch-Guayana
.gg	Guernsey

Tabelle B.2: ccTLDs (Stand: Mai 2004) (Fortsetzung)

Domain	Land
.gh	Ghana
.gi	Gibraltar
.gl	Grönland
.gm	Gambia
.gn	Guinea
.gp	Guadeloupe
.gq	Äquatorial-Guinea
.gr	Griechenland
.gs	Südgeorgien und die südlichen Sandwich-Inseln
.gt	Guatemala
.gu	Guam
.gw	Guinea-Bissau
.gy	Guyana
.hk	Hongkong
.hm	Heard- und McDonald-Inseln (bei der Antarktis)
.hn	Honduras
.hr	Kroatien
.ht	Haiti
.hu	Ungarn
.id	Indonesien
.ie	Irland
.il	Israel
.im	Isle of Man
.in	Indien
.io	Die britischen Inseln im Indischen Ozean

Tabelle B.2: ccTLDs (Stand: Mai 2004) (Fortsetzung)

Domain	Land
.iq	Irak
.ir	Iran
.is	Island
.it	Italien
.je	Die Insel Jersey
.jm	Jamaika
.jo	Jordanien
.jp	Japan
.ke	Kenia
.kg	Kirgisistan
.kh	Kambodscha
.ki	Kiribati
.km	Komoren
.kn	Saint Kitts und Nevis
.kp	Demokratische Volksrepublik Korea (Nordkorea)
.kr	Republik Korea (Südkorea)
.kw	Kuwait
.ky	Cayman-Inseln
.kz	Kasachstan
.la	Laos
.lb	Libanon
.lc	Santa Lucia
.li	Liechtenstein
.lk	Sri Lanka
.lr	Liberia

Tabelle B.2: ccTLDs (Stand: Mai 2004) (Fortsetzung)

Domain	Land
.ls	Lesotho
.lt	Litauen
.lu	Luxemburg
.lv	Lettland
.ly	Libyen
.ma	Marokko
.mc	Monaco
.md	Moldawien
.mg	Madagaskar
.mh	Marshall-Inseln
.mk	Mazedonien
.ml	Mali
.mm	Myanmar
.mn	Mongolei
.mo	Macao
.mp	Marianen
.mq	Martinique
.mr	Mauretanien
.ms	Montserrat
.mt	Malta
.mu	Mauritius
.mv	Malediven
.mw	Malawi
.mx	Mexiko
.my	Malaysia

Tabelle B.2: ccTLDs (Stand: Mai 2004) (Fortsetzung)

Domain	Land
.mz	Mosambik
.na	Namibia
.nc	Neu-Kaledonien
.ne	Niger
.nf	Norfolk-Insel
.ng	Nigeria
.ni	Nicaragua
.nl	Niederlande
.no	Norwegen
.np	Nepal
.nr	Nauru
.nu	Niue
.nz	Neuseeland
.om	Oman
.pa	Panama
.pe	Peru
.pf	Französisch-Polynesien (Tahiti)
.pg	Papua-Neuguinea
.ph	Philippinen
.pk	Pakistan
.pl	Polen
.pm	St. Pierre und Miquelon
.pn	Insel Pitcairn
.pr	Puerto Rico
.ps	Die palästinensischen Autonomiegebiete

Tabelle B.2: ccTLDs (Stand: Mai 2004) (Fortsetzung)

Domain	Land
.pt	Portugal
.pw	Palau
.py	Paraguay
.qa	Katar
.re	Réunion (zu Frankreich zählende Insel bei Südafrika)
.ro	Rumänien
.ru	Russland
.rw	Ruanda
.sa	Saudi-Arabien
.sb	Salomonen
.sc	Seychellen
.sd	Sudan
.se	Schweden
.sg	Singapur
.sh	St. Helena
.si	Slowenien
.sj	Die Inseln Svalbard und Jan Mayen
.sk	Slowakei
.sl	Sierra Leone
.sm	San Marino
.sn	Senegal
.so	Somalia
.sr	Surinam
.st	São Tomé und Príncipe
.sv	El Salvador

Tabelle B.2: ccTLDs (Stand: Mai 2004) (Fortsetzung)

Domain	Land
.sy	Syrien
.sz	Swasiland
.tc	Die Inseln Turks und Caicos
.td	Tschad
.tf	Die südlichen, französischen Territorien
.tg	Togo
.th	Thailand
.tj	Tadschikistan
.tk	Tokelau-Inseln
.tl	Timor
.tm	Turkmenistan
.tn	Tunesien
.to	Tonga
.tp	Ost-Timor
.tr	Türkei
.tt	Trinidad und Tobago
.tv	Tuvalu
.tw	Taiwan
.tz	Tansania
.ua	Ukraine
.ug	Uganda
.uk	Großbritannien
.um	Die kleineren Inseln vor den USA
.us	USA
.uy	Uruguay

Tabelle B.2: ccTLDs (Stand: Mai 2004) (Fortsetzung)

Domain	Land
.zu	Usbekistan
.va	Vatikan
.vc	St. Vincent und die Grenadinen
.ve	Venezuela
.vg	Jungfern-Inseln (zu England)
.vi	Jungfern-Inseln (zu USA)
.vn	Vietnam
.vu	Vanuatu
.wf	Die Inseln Wallis und Futuna
.ws	Samoa
.ye	Jemen
.yt	Mayotte
.yu	Kroatien, Slowenien, Mazedonien, Bosnien-Herzegowina, Serbien und Montenegro (früher: Jugoslawien)
.za	Republik Südafrika
.zm	Sambia
.zw	Simbabwe

Tabelle B.2: ccTLDs (Stand: Mai 2004) (Fortsetzung)

Mehr zum Thema: Die Internet Assigned Numbers Authority (IANA) pflegt diese Liste und veröffentlicht sie auf `http://www.iana.org/cctld/cctld-whois.htm`.

Eine Liste aller TLDs können Sie auch der DNS-Root-Zone entnehmen, deren Masterfile die Firma VeriSign unter `ftp://ftp.internic.net/domain/root.zone.gz` zur Verfügung stellt.

C RFCs zum Thema DNS

Dieses Kapitel beschreibt alle RFCs (Requests for Comments), die in diesem Buch erwähnt wurden oder im weitesten Sinne mit dem DNS zu tun haben. Die IETF ordnet jeder RFC einen Status zu.

- »UNKNOWN«, »EXPERIMENTAL«, »HISTORIC«
- »INFORMATIONAL«, »BEST CURRENT PRACTICE«
- »DRAFT STANDARD«, »PROPOSED STANDARD«, »STANDARD«

Wichtig sind nur die RFCs mit dem Status »BEST CURRENT PRACTICE«, »DRAFT STANDARD«, »PROPOSED STANDARD«, »STANDARD«, die noch nicht von einer anderen RFC abgelöst wurden. Ihre Beschreibungen sind grau hinterlegt.

Mehr zum Thema: Alle RFCs sind auf `http://www.ietf.org/rfc` erhältlich.

RFC226: Standardization of host mnemonics. P. M. Karp. Sep-20-1971. (Format: TXT=2012 bytes) (Obsoleted by RFC247) (Status: UNKNOWN)

RFC247: Proffered set of standard host names. P. M. Karp. Oct-12-1971. (Format: TXT=7122 bytes) (Obsoletes RFC226) (Status: UNKNOWN)

RFC606: Host names on-line. L. P. Deutsch. Dec-29-1973. (Format: TXT=6855 bytes) (Status: UNKNOWN)

RFC608: 0608 Host names on-line. M. D. Kudlick. Jan-10-1974. (Format: TXT=7010 bytes) (Obsoleted by RFC0810) (Status: UNKNOWN)

RFC799: Internet name domains. D. L. Mills. Sep-01-1981. (Format: TXT=13896 bytes) (Status: UNKNOWN)

RFC810: DoD Internet host table specification. E. J. Feinler, K. Harrenstien, Z. Su, V. White. Mar-01-1982. (Format: TXT=14196 bytes) (Obsoletes RFC0608) (Obsoleted by RFC0952) (Status: UNKNOWN)

RFC0811: Hostnames Server. K. Harrenstien, V. White, E. J. Feinler. Mar-01-1982. (Format: TXT=7771 bytes) (Obsoleted by RFC0953) (Status: UNKNOWN)

RFC819: Domain naming convention for Internet user applications. Z. Su, J. Postel. Aug-01-1982. (Format: TXT=35314 bytes) (Status: UNKNOWN)

RFC822: Standard for the format of ARPA Internet text messages. D. Crocker. Aug-13-1982. (Format: TXT=109200 bytes) (Obsoletes RFC0733) (Obsoleted by RFC2822) (Updated by RFC1123, RFC1138, RFC1148, RFC1327, RFC2156) (Also STD0011) (Status: STANDARD)

RFC833: Who talks TCP?. D. Smallberg. Dec-14-1982. (Format: TXT=42973 bytes) (Obsoletes RFC0832) (Obsoleted by RFC0834) (Status: UNKNOWN)

RFC881: Domain names plan and schedule. J. Postel. Nov-01-1983. (Format: TXT=23490 bytes) (Updated by RFC897) (Status: UNKNOWN)

RFC882: Domain names: Concepts and facilities. P. V. Mockapetris. Nov-01-1983. (Format: TXT=79776 bytes) (Obsoleted by RFC1034, RFC1035) (Updated by RFC973) (Status: UNKNOWN)

RFC883: Domain names: Implementation specification. P. V. Mockapetris. Nov-01-1983. (Format: TXT=175067 bytes) (Obsoleted by RFC1034, RFC1035) (Updated by RFC973) (Status: UNKNOWN)

RFC897: Domain name system implementation schedule. J. Postel. Feb-01-1984. (Format: TXT=15683 bytes) (Updates RFC881) (Updated by RFC921) (Status: UNKNOWN)

RFC920: Domain requirements. J. Postel, J. K. Reynolds. Oct-01-1984. (Format: TXT=27823 bytes) (Status: UNKNOWN)

RFC921: Domain name system implementation schedule – revised. J. Postel. Oct-01-1984. (Format: TXT=23318 bytes) (Updates RFC897) (Status: UNKNOWN)

RFC952: DoD Internet host table specification. K. Harrenstien, M. K. Stahl, E. J. Feinler. Oct-01-1985. (Format: TXT=12388 bytes) (Obsoletes RFC810) (Status: UNKNOWN)

RFC953: Hostname Server. K. Harrenstien, M. K. Stahl, E. J. Feinler. Oct-01-1985. (Format: TXT=8305 bytes) (Obsoletes RFC811) (Status: HISTORIC)

RFC972: Password Generator Protocol. F. J. Wancho. Jan-01-1986. (Format: TXT=3890 bytes) (Status: UNKNOWN)

RFC973: Domain system changes and observations. P. V. Mockapetris. Jan-01-1986. (Format: TXT=22364 bytes) (Obsoleted by RFC1034, RFC1035) (Updates RFC882, RFC883) (Status: UNKNOWN)

RFC974: Mail routing and the domain system. C. Partridge. Jan-01-1986. (Format: TXT=18581 bytes) (Obsoleted by RFC2821) (Also STD0010) (Status: HISTORIC)

RFC986: Guidelines for the use of Internet-IP addresses in the ISO Connectionless-Mode Network Protocol. R. W. Callon, H. W. Braun. Jun-01-1986. (Format: TXT=13950 bytes) (Obsoleted by RFC1069) (Status: UNKNOWN)

RFC1010: 1010 Assigned numbers. J. K. Reynolds, J. Postel. May-01-1987. (Format: TXT=78179 bytes) (Obsoletes RFC0990) (Obsoleted by RFC1060) (Status: HISTORIC)

RFC1032: Domain administrators guide. M. K. Stahl. Nov-01-1987. (Format: TXT=29454 bytes) (Status: UNKNOWN)

RFC1033: Domain administrators operations guide. M. Lottor. Nov-01-1987. (Format: TXT=37263 bytes) (Status: UNKNOWN)

RFC1034: Domain names – concepts and facilities. P. V. Mockapetris. Nov-01-1987. (Format: TXT=129180 bytes) (Obsoletes RFC973, RFC882, RFC883) (Updated by RFC1101, RFC1183, RFC1348, RFC1876, RFC1982, RFC2065, RFC2181, RFC2308, RFC2535) (Also STD0013) (Status: STANDARD)

RFC1035: Domain names – implementation and specification. P. V. Mockapetris. Nov-01-1987. (Format: TXT=125626 bytes) (Obsoletes RFC973, RFC882, RFC883) (Updated by RFC1101, RFC1183, RFC1348, RFC1876, RFC1982, RFC1995, RFC1996, RFC2065, RFC2136, RFC2181, RFC2137, RFC2308, RFC2535, RFC2845, RFC3425, RFC3658) (Also STD0013) (Status: STANDARD)

RFC1101: DNS encoding of network names and other types. P. V. Mockapetris. Apr-01-1989. (Format: TXT=28677 bytes) (Updates RFC1034, RFC1035) (Status: UNKNOWN)

RFC1122: Requirements for Internet Hosts – Communication Layers. R. Braden, Ed. October 1989. (Format: TXT=295992 bytes) (Updated by RFC1349) (Also STD0003) (Status: STANDARD)

RFC1123: Requirements for Internet Hosts – Application and Support. R. Braden, Ed. October 1989.
(Format: TXT=245503 bytes) (Updates RFC822) (Updated by RFC1349, RFC2181) (Also STD0003) (Status: STANDARD)

RFC1261: Transition of Nic Services. S. Williamson, L. Nobile. Sep-01-1991. (Format: TXT=4244 bytes) (Status: INFORMATIONAL)

RFC1178: Choosing a name for your computer. D. Libes. Aug-01-1990. (Format: TXT=18472 bytes) (Also FYI0005) (Status: INFORMATIONAL)

RFC1183: New DNS RR Definitions. C. F. Everhart, L. A. Mamakos, R. Ullmann, P. V. Mockapetris. Oct-01-1990. (Format: TXT=23788 bytes) (Updates RFC1034, RFC1035) (Status: EXPERIMENTAL)

RFC1261: 1261 Transition of Nic Services. S. Williamson, L. Nobile. Sep-01-1991. (Format: TXT=4244 bytes) (Status: INFORMATIONAL)

RFC1321: The MD5 Message-Digest Algorithm. R. Rivest. April 1992. (Format: TXT=35222 bytes) (Status: INFORMATIONAL)

RFC1348: DNS NSAP RRs. B. Manning. July 1992. (Format: TXT=6871 bytes) (Obsoleted by RFC1637) (Updates RFC1034, RFC1035) (Status: EXPERIMENTAL)

RFC1383: An Experiment in DNS Based IP Routing. C. Huitema. December 1992. (Format: TXT=32680 bytes) (Status: EXPERIMENTAL)

RFC1400: Transition and Modernization of the Internet Registration Service. S. Williamson. March 1993. (Format: TXT=13008 bytes) (Status: INFORMATIONAL)

RFC1464: Using the Domain Name System To Store Arbitrary String Attributes. R. Rosenbaum. May 1993. (Format: TXT=7953 bytes) (Status: EXPERIMENTAL)

RFC1480: The US Domain. A. Cooper, J. Postel. June 1993. (Format: TXT=100556 bytes) (Obsoletes RFC1386) (Status: INFORMATIONAL)

RFC1535: A Security Problem and Proposed Correction With Widely Deployed DNS Software. E. Gavron. October 1993. (Format: TXT=9722 bytes) (Status: INFORMATIONAL)

RFC1536: Common DNS Implementation Errors and Suggested Fixes. A. Kumar, J. Postel, C. Neuman, P. Danzig, S. Miller. October 1993. (Format: TXT=25476 bytes) (Status: INFORMATIONAL)

RFC1591: Domain Name System Structure and Delegation. J. Postel. March 1994. (Format: TXT=16481 bytes) (Status: INFORMATIONAL)

RFC1611: DNS Server MIB Extensions. R. Austein, J. Saperia. May 1994. (Format: TXT=58700 bytes) (Status: HISTORIC)

RFC1612: DNS Resolver MIB Extensions. R. Austein, J. Saperia. May 1994. (Format: TXT=61382 bytes) (Status: HISTORIC)

RFC1637: DNS NSAP Resource Records. B. Manning, R. Colella. June 1994. (Format: TXT=21768 bytes) (Obsoletes RFC1348) (Obsoleted by RFC1706) (Status: EXPERIMENTAL)

RFC1700: Assigned Numbers. J. Reynolds, J. Postel. October 1994. (Format: TXT=458860 bytes) (Obsoletes RFC1340) (Obsoleted by RFC3232) (Also STD0002) (Status: HISTORIC)

RFC1706: DNS NSAP Resource Records. B. Manning, R. Colella. October 1994. (Format: TXT=19721 bytes) (Obsoletes RFC1637) (Status: INFORMATIONAL)

RFC1712: DNS Encoding of Geographical Location. C. Farrell, M. Schulze, S. Pleitner, D. Baldoni. November 1994. (Format: TXT=13237 bytes) (Status: EXPERIMENTAL)

RFC1713: Tools for DNS debugging. A. Romao. November 1994. (Format: TXT=33500 bytes) (Also FYI0027) (Status: INFORMATIONAL)

RFC1750: Randomness Recommendations for Security. D. Eastlake 3rd, S. Crocker, J. Schiller. December 1994. (Format: TXT=73842 bytes) (Status: INFORMATIONAL)

RFC1794: DNS Support for Load Balancing. T. Brisco. April 1995. (Format: TXT=15494 bytes) (Status: INFORMATIONAL)

RFC1797: Class A Subnet Experiment. Internet Assigned Numbers Authority (IANA). April 1995. (Format: TXT=6779 bytes) (Status: EXPERIMENTAL)

RFC1876: A Means for Expressing Location Information in the Domain Name System. C. Davis, P. Vixie, T. Goodwin, I. Dickinson. January 1996. (Format: TXT=29631 bytes) (Updates RFC1034, RFC1035) (Status: EXPERIMENTAL)

RFC1876: A Means for Expressing Location Information in the Domain Name System. C. Davis, P. Vixie, T. Goodwin, I. Dickinson. January 1996. (Format: TXT=29631 bytes) (Updates RFC1034, RFC1035) (Status: EXPERIMENTAL)

RFC1884: IP Version 6 Addressing Architecture. R. Hinden, S. Deering, Eds. December 1995. (Format: TXT=37860 bytes) (Obsoleted by RFC2373) (Status: HISTORIC)

RFC1886: DNS Extensions to support IP version 6. S. Thomson, C. Huitema. December 1995. (Format: TXT=6424 bytes) (Obsoleted by RFC3596) (Updated by RFC2874, RFC3152) (Status: PROPOSED STANDARD)

RFC1912: Common DNS Operational and Configuration Errors. D. Barr. February 1996. (Format: TXT=38252 bytes) (Obsoletes RFC1537) (Status: INFORMATIONAL)

RFC1918: Address Allocation for Private Internets. Y. Rekhter, B. Moskowitz, D. Karrenberg, G. J. de Groot, E. Lear. February 1996. (Format: TXT=22270 bytes) (Obsoletes RFC1627, RFC1597) (Also BCP0005) (Status: BEST CURRENT PRACTICE)

RFC1932: IP over ATM: A Framework Document. R. Cole, D. Shur, C. Villamizar. April 1996. (Format: TXT=68031 bytes) (Status: INFORMATIONAL)

RFC1956: Registration in the MIL Domain. D. Engebretson, R. Plzak. June 1996. (Format: TXT=2923 bytes) (Status: INFORMATIONAL)

RFC1982: Serial Number Arithmetic. R. Elz, R. Bush. August 1996. (Format: TXT=14440 bytes) (Updates RFC1034, RFC1035) (Status: PROPOSED STANDARD)

RFC1992: The Nimrod Routing Architecture. I. Castineyra, N. Chiappa, M. Steenstrup. August 1996. (Format: TXT=59848 bytes) (Status: INFORMATIONAL)

RFC1995: Incremental Zone Transfer in DNS. M. Ohta. August 1996. (Format: TXT=16810 bytes) (Updates RFC1035) (Status: PROPOSED STANDARD)

RFC1996: A Mechanism for Prompt Notification of Zone Changes (DNS NOTIFY). P. Vixie. August 1996.
(Format: TXT=15247 bytes) (Updates RFC1035) (Status: PROPOSED STANDARD)

RFC2010: Operational Criteria for Root Name Servers. B. Manning, P. Vixie. October 1996. (Format: TXT=14870 bytes) (Obsoleted by RFC2870) (Status: INFORMATIONAL)

RFC2050: Internet Registry IP Allocation Guidelines. K. Hubbard, M. Kosters, D. Conrad, D. Karrenberg, J. Postel. November 1996.
(Format: TXT=28975 bytes) (Obsoletes RFC1466) (Also BCP0012) (Status: BEST CURRENT PRACTICE)

RFC2052: A DNS RR for specifying the location of services (DNS SRV). A. Gulbrandsen, P. Vixie. October 1996. (Format: TXT=19257 bytes) (Obsoleted by RFC2782) (Status: EXPERIMENTAL)

RFC2103: Mobility Support for Nimrod: Challenges and Solution Approaches. R. Ramanathan. February 1997. (Format: TXT=41352 bytes) (Status: INFORMATIONAL)

RFC2104: HMAC: Keyed-Hashing for Message Authentication. H. Krawczyk, M. Bellare, R. Canetti. February 1997. (Format: TXT=22297 bytes) (Status: INFORMATIONAL)

RFC2136: Dynamic Updates in the Domain Name System (DNS UPDATE). P. Vixie, Ed., S. Thomson, Y. Rekhter, J. Bound. April 1997.
(Format: TXT=56354 bytes) (Updates RFC1035) (Updated by RFC3007) (Status: PROPOSED STANDARD)

RFC2137: Secure Domain Name System Dynamic Update. D. Eastlake 3rd. April 1997. (Format: TXT=24824 bytes) (Obsoleted by RFC3007) (Updates RFC1035) (Status: PROPOSED STANDARD)

RFC2141: URN Syntax. R. Moats. May 1997.
(Format: TXT=14077 bytes) (Status: PROPOSED STANDARD)

RFC2142: Mailbox Names for Common Services, Roles and Functions. D. Crocker. May 1997. (Format: TXT=12195 bytes) (Status: PROPOSED STANDARD)

RFC2146: U.S. Government Internet Domain Names. Federal Networking Council. May 1997. (Format: TXT=26564 bytes) (Obsoletes RFC1816) (Status: INFORMATIONAL)

RFC2163: Using the Internet DNS to Distribute MIXER Conformant Global Address Mapping (MCGAM). C. Allocchio. January 1998. (Format: TXT=58789 bytes) (Obsoletes RFC1664) (Updated by RFC3597) (Status: PROPOSED STANDARD)

RFC2168: Resolution of Uniform Resource Identifiers using the Domain Name System. R. Daniel, M. Mealling. June 1997. (Format: TXT=46528 bytes) (Obsoleted by RFC3401, RFC3402, RFC3403, RFC3404) (Updated by RFC2915) (Status: EXPERIMENTAL)

RFC2181: Clarifications to the DNS Specification. R. Elz, R. Bush. July 1997. (Format: TXT=36989 bytes) (Updates RFC1034, RFC1035, RFC1123) (Updated by RFC2535) (Status: PROPOSED STANDARD)

RFC2182: Selection and Operation of Secondary DNS Servers. R. Elz, R. Bush, S. Bradner, M. Patton. July 1997. (Format: TXT=27456 bytes) (Also BCP0016) (Status: BEST CURRENT PRACTICE)

RFC2219: Use of DNS Aliases for Network Services. M. Hamilton, R. Wright. October 1997. (Format: TXT=17858 bytes) (Also BCP0017) (Status: BEST CURRENT PRACTICE)

RFC2230: Key Exchange Delegation Record for the DNS. R. Atkinson. November 1997. (Format: TXT=25563 bytes) (Status: INFORMATIONAL)

RFC2235: Hobbes' Internet Timeline. R. Zakon. November 1997. (Format: TXT=43060 bytes) (Also FYI0032) (Status: INFORMATIONAL)

RFC2247: Using Domains in LDAP/X.500 Distinguished Names. S. Kille, M. Wahl, A. Grimstad, R. Huber, S. Sataluri. January 1998. (Format: TXT=12411 bytes) (Status: PROPOSED STANDARD)

RFC2308: Negative Caching of DNS Queries (DNS NCACHE). M. Andrews. March 1998. (Format: TXT=41428 bytes) (Updates RFC1034, RFC1035) (Status: PROPOSED STANDARD)

RFC2315: PKCS #7: Cryptographic Message Syntax Version 1.5. B. Kaliski. March 1998. (Format: TXT=69679 bytes) (Status: INFORMATIONAL)

RFC2317: Classless IN-ADDR.ARPA delegation. H. Eidnes, G. de Groot, P. Vixie. March 1998. (Format: TXT=17744 bytes) (Also BCP0020) (Status: BEST CURRENT PRACTICE)

RFC2345: Domain Names and Company Name Retrieval. J. Klensin, T. Wolf, G. Oglesby. May 1998. (Format: TXT=29707 bytes) (Status: EXPERIMENTAL)

RFC2373: IP Version 6 Addressing Architecture. R. Hinden, S. Deering. July 1998. (Format: TXT=52526 bytes) (Obsoletes RFC1884) (Obsoleted by RFC3513) (Status: PROPOSED STANDARD)

RFC2374: An IPv6 Aggregatable Global Unicast Address Format. R. Hinden, M. O'Dell, S. Deering. July 1998. (Format: TXT=25068 bytes) (Obsoletes RFC2073) (Obsoleted by RFC3587) (Status: HISTORIC)

RFC2375: IPv6 Multicast Address Assignments. R. Hinden, S. Deering. July 1998. (Format: TXT=14356 bytes) (Status: INFORMATIONAL)

RFC2377: Naming Plan for Internet Directory-Enabled Applications. A. Grimstad, R. Huber, S. Sataluri, M. Wahl. September 1998. (Format: TXT=38274 bytes) (Status: INFORMATIONAL)

RFC2396: Uniform Resource Identifiers (URI): Generic Syntax. T. Berners-Lee, R. Fielding, L. Masinter. August 1998. (Format: TXT=83639 bytes) (Updates RFC1808, RFC1738) (Updated by RFC2732) (Status: DRAFT STANDARD)

RFC2483: URI Resolution Services Necessary for URN Resolution. M. Mealling, R. Daniel. January 1999. (Format: TXT=30518 bytes) (Status: EXPERIMENTAL)
RFC2517: Building Directories from DNS: Experiences from WWWSeeker. R. Moats, R. Huber. February 1999. (Format: TXT=14001 bytes) (Status: INFORMATIONAL)

RFC2535: Domain Name System Security Extensions. D. Eastlake 3rd. March 1999. (Format: TXT=110958 bytes) (Obsoletes RFC2065) (Updates RFC2181, RFC1035, RFC1034) (Updated by RFC2931, RFC3007, RFC3008, RFC3090, RFC3226, RFC3445, RFC3597, RFC3655, RFC3658) (Status: PROPOSED STANDARD)

RFC2536: DSA KEYs and SIGs in the Domain Name System (DNS). D. Eastlake 3rd. March 1999. (Format: TXT=11121 bytes) (Status: PROPOSED STANDARD)

RFC2537: RSA/MD5 KEYs and SIGs in the Domain Name System (DNS). D. Eastlake 3rd. March 1999. (Format: TXT=10810 bytes) (Obsoleted by RFC3110) (Status: PROPOSED STANDARD)

RFC2538: Storing Certificates in the Domain Name System (DNS). D. Eastlake 3rd, O. Gudmundsson. March 1999. (Format: TXT=19857 bytes) (Status: PROPOSED STANDARD)

RFC2539: Storage of Diffie-Hellman Keys in the Domain Name System (DNS). D. Eastlake 3rd. March 1999. (Format: TXT=21049 bytes) (Status: PROPOSED STANDARD)

RFC2540: Detached Domain Name System (DNS) Information. D. Eastlake 3rd. March 1999. (Format: TXT=12546 bytes) (Status: EXPERIMENTAL)

RFC2541: DNS Security Operational Considerations. D. Eastlake 3rd. March 1999. (Format: TXT=14498 bytes) (Status: INFORMATIONAL)

RFC2544: Benchmarking Methodology for Network Interconnect Devices. S. Bradner, J. McQuaid. March 1999. (Format: TXT=66688 bytes) (Obsoletes RFC1944) (Status: INFORMATIONAL)

RFC2553: Basic Socket Interface Extensions for IPv6. R. Gilligan, S. Thomson, J. Bound, W. Stevens. March 1999. (Format: TXT=89215 bytes) (Obsoletes RFC2133) (Obsoleted by RFC3493) (Updated by RFC3152) (Status: INFORMATIONAL)

RFC2606: Reserved Top Level DNS Names. D. Eastlake 3rd, A. Panitz. June 1999. (Format: TXT=8008 bytes) (Also BCP0032) (Status: BEST CURRENT PRACTICE)

RFC2611: URN Namespace Definition Mechanisms. L. Daigle, D. van Gulik, R. Iannella, P. Falstrom. June 1999. (Format: TXT=26916 bytes) (Obsoleted by RFC3406) (Also BCP0033) (Status: BEST CURRENT PRACTICE)

RFC2671: Extension Mechanisms for DNS (EDNS0). P. Vixie. August 1999. (Format: TXT=15257 bytes) (Status: PROPOSED STANDARD)

RFC2672: Non-Terminal DNS Name Redirection. M. Crawford. August 1999. (Format: TXT=18321 bytes) (Status: PROPOSED STANDARD)

RFC2673: Binary Labels in the Domain Name System. M. Crawford. August 1999. (Format: TXT=12379 bytes) (Updated by RFC3363, RFC3364) (Status: EXPERIMENTAL)

RFC2694: DNS extensions to Network Address Translators (DNS_ALG). P. Srisuresh, G. Tsirtsis, P. Akkiraju, A. Heffernan. September 1999. (Format: TXT=67720 bytes) (Status: INFORMATIONAL)

RFC2695: Authentication Mechanisms for ONC RPC. A. Chiu. September 1999. (Format: TXT=39286 bytes) (Status: INFORMATIONAL)

RFC2821: Simple Mail Transfer Protocol. J. Klensin, Ed. April 2001. (Format: TXT=192504 bytes) (Obsoletes RFC0821, RFC0974, RFC1869) (Status: PROPOSED STANDARD)

RFC2822: Internet Message Format. P. Resnick, Ed. April 2001. (Format: TXT=110695 bytes) (Obsoletes RFC0822) (Status: PROPOSED STANDARD)

RFC2825: A Tangled Web: Issues of I18N, Domain Names, and the Other Internet protocols. IAB, L. Daigle, Ed. May 2000. (Format: TXT=15612 bytes) (Status: INFORMATIONAL)

RFC2826: IAB Technical Comment on the Unique DNS Root. Internet Architecture Board. May 2000. (Format: TXT=13400 bytes) (Status: INFORMATIONAL)

RFC2845: Secret Key Transaction Authentication for DNS (TSIG). P. Vixie, O. Gudmundsson, D. Eastlake 3rd, B. Wellington. May 2000. (Format: TXT=32272 bytes) (Updates RFC1035) (Updated by RFC3645) (Status: PROPOSED STANDARD)

RFC2870: Root Name Server Operational Requirements. R. Bush, D. Karrenberg, M. Kosters, R. Plzak. June 2000. (Format: TXT=21133 bytes) (Obsoletes RFC2010) (Also BCP0040) (Status: BEST CURRENT PRACTICE)

RFC2874: DNS Extensions to Support IPv6 Address Aggregation and Renumbering. M. Crawford, C. Huitema. July 2000. (Format: TXT=44204 bytes) (Updates RFC1886) (Updated by RFC3152, RFC3226, RFC3363, RFC3364) (Status: EXPERIMENTAL)

RFC2915: The Naming Authority Pointer (NAPTR) DNS Resource Record. M. Mealling, R. Daniel. September 2000. (Format: TXT=41521 bytes) (Obsoleted by RFC3401, RFC3402, RFC3403, RFC3404) (Updates RFC2168) (Status: PROPOSED STANDARD)

RFC2916: E.164 number and DNS. P. Faltstrom. September 2000. (Format: TXT=18159 bytes) (Obsoleted by RFC3761) (Status: PROPOSED STANDARD)

RFC2929: Domain Name System (DNS) IANA Considerations. D. Eastlake 3rd, E. Brunner-Williams, B. Manning. September 2000. (Format: TXT=22454 bytes) (Also BCP0042) (Status: BEST CURRENT PRACTICE)

RFC2930: Secret Key Establishment for DNS (TKEY RR). D. Eastlake 3rd. September 2000. (Format: TXT=34894 bytes) (Status: PROPOSED STANDARD)

RFC2931: DNS Request and Transaction Signatures (SIG(0)s). D. Eastlake 3rd. September 2000. (Format: TXT=19073 bytes) (Updates RFC2535) (Status: PROPOSED STANDARD)

RFC3007: Secure Domain Name System (DNS) Dynamic Update. B. Wellington. November 2000. (Format: TXT=18056 bytes) (Obsoletes RFC2137) (Updates RFC2535, RFC2136) (Status: PROPOSED STANDARD)

RFC3008: Domain Name System Security (DNSSEC) Signing Authority. B. Wellington. November 2000. (Format: TXT=13484 bytes) (Updates RFC2535) (Updated by RFC3658) (Status: PROPOSED STANDARD)

RFC3026: Liaison to IETF/ISOC on ENUM. R. Blane. January 2001. (Format: TXT=11460 bytes) (Status: INFORMATIONAL)

RFC3068: An Anycast Prefix for 6to4 Relay Routers. C. Huitema. June 2001. (Format: TXT=20120 bytes) (Status: PROPOSED STANDARD)

RFC3071: Reflections on the DNS, RFC 1591, and Categories of Domains. J. Klensin. February 2001. (Format: TXT=24892 bytes) (Status: INFORMATIONAL)

RFC3090: DNS Security Extension Clarification on Zone Status. E. Lewis. March 2001. (Format: TXT=24166 bytes) (Updates RFC2535) (Updated by RFC3658) (Status: PROPOSED STANDARD)

RFC3110: RSA/SHA-1 SIGs and RSA KEYs in the Domain Name System (DNS). D. Eastlake 3rd. May 2001. (Format: TXT=14587 bytes) (Obsoletes RFC2537) (Status: PROPOSED STANDARD)

RFC3123: A DNS RR Type for Lists of Address Prefixes (APL RR). P. Koch. June 2001. (Format: TXT=14648 bytes) (Status: EXPERIMENTAL)

RFC3130: Notes from the State-Of-The-Technology: DNSSEC. E. Lewis. June 2001. (Format: TXT=22291 bytes) (Status: INFORMATIONAL)

RFC3171: IANA Guidelines for IPv4 Multicast Address Assignments. Z. Albanna, K. Almeroth, D. Meyer, M. Schipper. August 2001. (Format: TXT=15389 bytes) (Also BCP0051) (Status: BEST CURRENT PRACTICE)

RFC3197: Applicability Statement for DNS MIB Extensions. R. Austein. November 2001. (Format: TXT=8610 bytes) (Status: INFORMATIONAL)

RFC3225: Indicating Resolver Support of DNSSEC. D. Conrad. December 2001. (Format: TXT=11548 bytes) (Status: PROPOSED STANDARD)

RFC3226: DNSSEC and IPv6 A6 aware server/resolver message size requirements. O. Gudmundsson. December 2001. (Format: TXT=12078 bytes) (Updates RFC2535, RFC2874) (Status: PROPOSED STANDARD)

RFC3245: The History and Context of Telephone Number Mapping (ENUM) Operational Decisions: Informational Documents Contributed to ITU-T Study Group 2 (SG2). J. Klensin, Ed., IAB. March 2002. (Format: TXT=23758 bytes) (Status: INFORMATIONAL)

RFC3330: Special-Use IPv4 Addresses. IANA. September 2002. (Format: TXT=16200 bytes) (Status: INFORMATIONAL)

RFC3363: Representing Internet Protocol version 6 (IPv6) Addresses in the Domain Name System (DNS). R. Bush, A. Durand, B. Fink, O. Gudmundsson, T. Hain. August 2002. (Format: TXT=11055 bytes) (Updates RFC2673, RFC2874) (Status: INFORMATIONAL)

RFC3364: Tradeoffs in Domain Name System (DNS) Support for Internet Protocol version 6 (IPv6). R. Austein. August 2002. (Format: TXT=26544 bytes) (Updates RFC2673, RFC2874) (Status: INFORMATIONAL)

RFC3401: Dynamic Delegation Discovery System (DDDS) Part One: The Comprehensive DDDS. M. Mealling. October 2002. (Format: TXT=10172 bytes) (Obsoletes RFC2915, RFC2168) (Updates RFC2276) (Status: INFORMATIONAL)

RFC3402: Dynamic Delegation Discovery System (DDDS) Part Two: The Algorithm. M. Mealling. October 2002. (Format: TXT=38925 bytes) (Obsoletes RFC2915, RFC2168) (Status: PROPOSED STANDARD)

RFC3403: Dynamic Delegation Discovery System (DDDS) Part Three: The Domain Name System (DNS) Database. M. Mealling. October 2002. (Format: TXT=31058 bytes) (Obsoletes RFC2915, RFC2168) (Status: PROPOSED STANDARD)

RFC3404: Dynamic Delegation Discovery System (DDDS) Part Four: The Uniform Resource Identifiers (URI). M. Mealling. October 2002. (Format: TXT=40124 bytes) (Obsoletes RFC2915, RFC2168) (Status: PROPOSED STANDARD)

RFC3405: Dynamic Delegation Discovery System (DDDS) Part Five: URI.ARPA Assignment Procedures. M. Mealling. October 2002. (Format: TXT=19469 bytes) (Also BCP0065) (Status: BEST CURRENT PRACTICE)

RFC3425: Obsoleting IQUERY. D. Lawrence. November 2002. (Format: TXT=8615 bytes) (Updates RFC1035) (Status: PROPOSED STANDARD)

RFC3467: Role of the Domain Name System (DNS). J. Klensin. February 2003. (Format: TXT=84570 bytes) (Status: INFORMATIONAL)

RFC3445: Limiting the Scope of the KEY Resource Record (RR). D. Massey, S. Rose. December 2002. (Format: TXT=20947 bytes) (Updates RFC2535) (Status: PROPOSED STANDARD)

RFC3454: Preparation of Internationalized Strings ("stringprep"). P. Hoffman, M. Blanchet. December 2002. (Format: TXT=138684 bytes) (Status: PROPOSED STANDARD)

RFC3490: Internationalizing Domain Names in Applications (IDNA). P. Faltstrom, P. Hoffman, A. Costello. March 2003. (Format: TXT=51943 bytes) (Status: PROPOSED STANDARD)

RFC3491: Nameprep: A Stringprep Profile for Internationalized Domain Names (IDN). P. Hoffman, M. Blanchet. March 2003. (Format: TXT=10316 bytes) (Status: PROPOSED STANDARD)

RFC3492: Punycode: A Bootstring encoding of Unicode for Internationalized Domain Names in Applications (IDNA). A. Costello. March 2003. (Format: TXT=67439 bytes) (Status: PROPOSED STANDARD)

RFC3535: Overview of the 2002 IAB Network Management Workshop. J. Schoenwaelder. May 2003. (Format: TXT=42566 bytes) (Status: INFORMATIONAL)

RFC3596: DNS Extensions to Support IP Version 6. S. Thomson, C. Huitema, V. Ksinant, M. Souissi. October 2003. (Format: TXT=14093 bytes) (Obsoletes RFC3152, RFC1886) (Status: DRAFT STANDARD)

RFC3597: Handling of Unknown DNS Resource Record (RR) Types. A. Gustafsson. September 2003. (Format: TXT=17559 bytes) (Updates RFC2163, RFC2535) (Status: PROPOSED STANDARD)

RFC3645: Generic Security Service Algorithm for Secret Key Transaction Authentication for DNS (GSS-TSIG). S. Kwan, P. Garg, J. Gilroy, L. Esibov, J. Westhead, R. Hall. October 2003. (Format: TXT=56162 bytes) (Updates RFC2845) (Status: PROPOSED STANDARD)

RFC3646: DNS Configuration options for Dynamic Host Configuration Protocol for IPv6 (DHCPv6). R. Droms, Ed. December 2003. (Format: TXT=13312 bytes) (Status: PROPOSED STANDARD)

RFC3655: Redefinition of DNS Authenticated Data (AD) bit. B. Wellington, O. Gudmundsson. November 2003. (Format: TXT=15646 bytes) (Updates RFC2535) (Status: PROPOSED STANDARD)

RFC3658: Delegation Signer (DS) Resource Record (RR). O. Gudmundsson. December 2003. (Format: TXT=42120 bytes) (Updates RFC3090, RFC3008, RFC2535, RFC1035) (Updated by RFC3755) (Status: PROPOSED STANDARD)

RFC3731: Extensible Provisioning Protocol (EPP) Domain Name Mapping. S. Hollenbeck. March 2004. (Format: TXT=92527 bytes) (Status: PROPOSED STANDARD)

RFC3743: Joint Engineering Team (JET) Guidelines for Internationalized Domain Names (IDN) Registration and Administration for Chinese, Japanese, and Korean. K. Konishi, K. Huang, H. Qian, Y. Ko. April 2004. (Format: TXT=74963 bytes) (Status: INFORMATIONAL)

RFC3755: Legacy Resolver Compatibility for Delegation Signer (DS). S. Weiler. May 2004.
(Format: TXT=19812 bytes) (Updates RFC3658, RFC2535) (Updated by RFC3757, RFC3845) (Status: PROPOSED STANDARD)

RFC3757: Domain Name System KEY (DNSKEY) Resource Record (RR) Secure Entry Point (SEP) Flag. O. Kolkman, J. Schlyter, E. Lewis. May 2004.
(Format: TXT=16868 bytes) (Updates RFC3755, RFC2535) (Status: PROPOSED STANDARD)

RFC3761: The E.164 to Uniform Resource Identifiers (URI) Dynamic Delegation Discovery System (DDDS) Application (ENUM). P. Faltstrom, M. Mealling. April 2004.
(Format: TXT=41559 bytes) (Obsoletes RFC2916) (Status: PROPOSED STANDARD)

RFC3762: Telephone Number Mapping (ENUM) Service Registration for H.323. O. Levin. April 2004.
(Format: TXT=8450 bytes) (Status: PROPOSED STANDARD)

RFC3832: Remote Service Discovery in the Service Location Protocol (SLP) via DNS SRV. W. Zhao, H. Schulzrinne, E. Guttman, C. Bisdikian, W. Jerome. July 2004. (Format: TXT=15602 bytes) (Status: EXPERIMENTAL)

RFC3845: DNS Security (DNSSEC) NextSECure (NSEC) RDATA Format. J. Schlyter, Ed. August 2004.
(Format: TXT=14793 bytes) (Updates RFC3755, RFC2535) (Status: PROPOSED STANDARD)

Stand: August 2004

D Abkürzungsverzeichnis

In diesem Buch werden die folgenden Abkürzungen verwendet:

Abkürzung	Bedeutung
ADS	Active Directory Service
AFS	Andrew File System
ARPA	Address Routing Parameter Area
ARPANET	Advanced Research Projects Agency Network
ASN	Autonomous System Number
ASO	Address Supporting Organization
BIND	Berkeley Internet Name Domain
BSI	Bundesamt für Sicherheit in der Informationstechnik
ccTLD	country code Top Level Domain
CIDR	Classless Inter-Domain Routing
CLNP	Connectionless Network Protocol
CNAME	canonical name
DARPA	Department of Defense Advanced Research Projects Agency
DC	Domain Controler
DCA	Defense Communications Agency
DCE	Distributed Computing Environment
DDN	Defense Data Network
DDNS	Dynamic Delegation Discovery System
DH	Diffie-Hellmann
DIG	Domain Information Groper

Tabelle D.1: Verwendete Abkürzungen

Abkürzung	Bedeutung
DISA	Defense Information Systems Agency
DNS	Domain Name System
DNSO	Domain Name Supporting Organization
DNSSEC	Domain Name System Security Extensions
DoS	Denial of Service
DynDNS	Dynamic Domain Name System
EBNF	Erweiterte Backus-Naur-Form
FQDN	Fully Qualified Domain Name
FTP	File Transfer Protocol
GC	Global Catalog
GLIBC	GNU C-Library
GMT	Greenwich Mean Time
GNU	GNU is not Unix
GSI	Government Systems, Inc.
gTLD	generic Top Level Domain
HA	High Availability
IAB	Internet Architecture Board
IANA	Internet Assigned Number Authority
IBM	International Business Machines
ICANN	Internet Corporation for Assigned Names and Numbers
IDN	Internationalized Domain Names
IDNA	Internationalized Domain Names for Applications
IETF	Internet Engineering Task Force
IN-ADDR	Internet Address
INTERNIC	Internet Network Information Center
IP	Internet Protocol

Tabelle D.1: Verwendete Abkürzungen (Fortsetzung)

Abkürzung	Bedeutung
ISO	International Standards Organization
ISOC	Internet Society
JAG	Just Another Geek
KDC	Key Distribution Center
KEX	Key Exchange
KK	Konnektivitäts-Koordination
KSK	Key Signing Key
LIBLWRES	Lightweight Resolver Library
LIR	Local Internet Registry
LWRESD	Lightweight Resolver Daemon
MAC	Message Authentication Code
MIT	Massachusetts Institute of Technology
MTA	Mail Transfer Agent
NAPTR	Naming-Authority-Pointer
NCP	Network Control Protocol
NIC	Network Information Center
NIR	National Internet Registry
NS	Nameserver
NSAP	Network Service Access Point
NSF	National Science Foundation
NSI	Network Solutions, Inc.
NSS	Name Service Switch
PID	Process Identification
PIR	Public Internet Registry
PSO	Protocol Supporting Organization
RBI	Rechnerbetriebsgruppe Informatik

Tabelle D.1: Verwendete Abkürzungen (Fortsetzung)

Abkürzung	Bedeutung
RBL	Realtime Blacklist
RDN	Relative Distinguished Name
RFC	Request for comment
RIR	Regional Internet Registry
RR	Resource Record
SAIC	Science Applications International Corporation
SEP	Secure Entry Point
SOA	Start of Authority
SPOF	Single Point of Failure
SRI	Stanford Research Institute
SRS	Shared Registry System
TCP	Transmission Control Protocol
TLD	Top Level Domain
TSIG	Transaction Signature
TTL	Time to Live
UDN	Unqualified Domain Name
UDP	Unix Domain Protocol
URI	Uniform Resource Identifier
URL	Uniform Resource Locator
URN	Uniform Resource Name
US	United States
USA	United States of America
USC	University of Southern California
VC	Virtual Circuit
W3C	World Wide Web Consortium
WWW	World Wide Web

Tabelle D.1: Verwendete Abkürzungen (Fortsetzung)

S Stichwortverzeichnis

B

E

F

G

H

I

J

K

N

O

P

Q

R

S

T

U

V

W

X

Y

Z