RUS
Regionales
Rechenzentrum
der Universität
Stuttgart

BelWü
Landes-
forschungsnetz
Baden -
Württemberg

Lisa Golka (RUS)
Ulrike Dillmann, Jürgen Georgi (BelWü-Koordination)

Nameserver

Das Domain Name System und die Verwaltung
einer Internet-Domain in Deutschland

2. Auflage

2. Auflage

Vorwort

Das Ziel des vorliegenden Dokuments ist es, eine Anleitung für die Teilnahme im Domain Name System (DNS) und zum Betrieb eines Nameservers zu geben. Dieses Handbuch bietet keine Einführung in Verteilte Directory Systeme und verzichtet auf einer Bewertung dieser sowie auf einem Vergleich zwischen DNS und X.500.

Es wurde weitgehend versucht, keine Spezialkenntnisse des Domain Name Systems vorauszusetzen. Ein ergänzendes Studium der einschlägigen RFCs ist in jedem Fall vorteilhaft.

Das Handbuch ist in zwei Teile geteilt:

Der theoretische Teil (Kapitel 1–3) befaßt sich hauptsächlich mit der Methode, weltweit eindeutige und sinnvolle Namen für Netzknoten zu vergeben. Die Verfahrensweise, um die Zuordnung dieser Namen zu Internet Adressen zu erreichen, ist ebenfalls beschrieben.

Der praktische Teil (Kapitel 4–7) enthält Details und Hinweise, wie man einen BIND-Nameserver richtig konfiguriert und betreibt. Dies ist besonders wichtig, weil eine ordnungsgemäße Verwaltung einer Domain Voraussetzung für eine reibungslose Teilnahme am Internet ist.

Das Papier basiert auf den Erfahrungen mit der Einführung und dem Betrieb von Domain Nameservern am Rechenzentrum der Universität Stuttgart (RUS), sowie den Universitäten des Landesforschungsnetzes BelWü (Baden-Württembergs extended LAN).
Geschrieben wurde es im wesentlichen von Lisa Golka unter Mitwirkung von Ulrike Dillmann und Jürgen Georgi. Anregungen und Beiträge hat Michael Hebgen (Uni Heidelberg) geliefert. Weitere Anregungen sind von Ingo Kleinschroth (Uni Stuttgart), Matthias Melcher (Uni Heidelberg) und Jürgen Rauschenbach (DFN Verein) gekommen.
Wertvolle Bemerkungen haben Jochen Brüning (Uni Konstanz) und Detlef Schmidt (TU Braunschweig) beigetragen. Für Korrekturen sei R. Horn und K.-D. Mayer-Spohn gedankt; bei den LaTeX- Problemen halfen B. Burr und F. Keim.

Für weitere Anregungen und Verbesserungsvorschläge sind wir dankbar, da die Qualität einer solchen Anleitung von den eingeflossenen Erfahrungen vieler Personen abhängt. Verwenden Sie hierfür bitte eine E-Mail an `dns-handbuch@belwue.dbp.de`.

Das Handbuch kann per anonymous ftp vom Infoserver `rusmv1.rus.uni-stuttgart.de` (129.69.1.12) im Verzeichnis `info/netze/belwue/handbuecher` (`dns.ps` oder `dns.hp`) geholt werden; eine Hardcopy können Sie (notfalls) über obige Mailadresse erhalten und in beschränkter Auflage vom DFN-Verein (Herrn J. Rauschenbach, Pariserstr. 44, 1000 Berlin).

Stuttgart, den 12. August 1991

Peter Merdian
BelWü-Koordination

Inhaltsverzeichnis

Abbildungsverzeichnis

Kapitel 1

Das Domain Name System (DNS)

1.1 Einführung

Mit zunehmender Ausbreitung der Computer-Netzwerke hat auch die Anzahl der vernetzten Rechner und anderer Objekte der Datenkommunikation rapide zugenommen. Die Vergabe und Verwaltung der Namen dieser Objekte wurde dadurch nicht nur komplexer, sondern gewann immer mehr an Bedeutung.

Die ursprünglich verwendete Methode einer zentral gewarteten Host-Datei zur Verwaltung von Rechnernamen wurde schnell unausführbar. Man erkannte, daß nur ein dezentralisierter Naming-Service mit lokal verwalteten Namen mit dem rapiden Zuwachs der vernetzten Rechner Schritt halten kann. Schon im Jahre 1983 wurde der Grundstein dieses Dienstes durch die Definition des Domain Name Systems (DNS) gelegt.

Das DNS ist ein verteiltes Directory-System. Aufgabe eines Directory-Systems ist das Verwalten von Informationen über Objekte, in diesem Fall über Objekte der Datenkommunikation. Als zentrale Komponente dieses Systems wird die hierarchische Strukturierung des Domainnamensraums betrachtet, welche die dezentralisierte Verwaltung der Informationen ermöglicht. Mit Nameservice wird ein Dienst bezeichnet, der von mehreren miteinander kommunizierenden Systemen erbracht wird unter Verwendung von verteilten DNS-konformen Daten.

Obwohl das DNS unter dem Druck des schnell wachsenden Internet konzipiert wurde, ist es mächtig genug, um unterschiedliche Netzprotokolle zu unterstützen. Hier wird es jedoch als Teil des Internet betrachtet.

1.2 Die Grundidee des DNS

Ziel des DNS ist es, die dezentrale Verwaltung von Informationen über Objekte der Datenkommunikation zu ermöglichen. Diese dezentrale Verwaltung der Informationen, sowie die Eindeutigkeit der Objektnamen kann nur durch eine hierarchische Strukturierung des Namensraums erreicht werden; somit stellt diese die zentrale Komponente des DNS dar.

Eine Domain ist ein kompletter Ast dieser baumartigen Struktur. Es ist dann leicht, Äste des Baumes einzelnen Administratoren zuzuordnen. Der Administrator kann, nach Bedarf, die Kontrolle über Teile seines Astes an eine andere Person delegieren usw. Man bezeichnet eine Domain abzüglich aller delegierten Äste als Zone. Der Administrator einer Zone ist zur Verwaltung aller Namen in seinem Zonenbereich bevollmächtigt. Ein Beispiel: Der Administrator der Domain `ucla.edu` kann die Subdomain `ai.ucla.edu` an einen anderen vergeben, dadurch entstehen zwei Zonen für die Domain `ucla.edu`: die erste enthält `ucla.edu` mit allen Subdomains außer `ai.ucla.edu`, die zweite besteht nur aus der Domain `ai.ucla.edu`.

Durch die Baumstruktur des Namensraumes ist die Eindeutigkeit der Knotennamen weltweit gewährleistet. Der Datenadministrator einer Domain muß dafür sorgen, daß in seiner Domain die Namen eindeutig sind. Der Name `sun1.twg.com` und `sun1.ucla.edu` sind weltweit eindeutig.

```
                root (oder ".")
                 |
       +---------+---------+
       |                   |
      com                 edu              Top-level domains
       |                   |
       |                   |
  +----+----+         +----+----+
  |         |         |         |
  |         |         |         |
cisco      twg       mit      ucla         Second-level domains
            |                   |
            |                   |
         +--+--+             +--+--+----+-----+
         |     |             |     |    |     |
         |     |             |     |    |     |
        vax1  sun1          ai    cs   math  sun1
                                   |
                                   |
                                 vax1
```

Abbildung 1.1: Beispiel für die Baumstruktur des Namensraums

Die Verwaltung dieser Informationen wird von einem verteilten Directory-Dienst (Nameserver) unterstützt. Die häufigste Verwendung des Systems, das in der Funktionalität einem Telefonbuch (englisch Directory) entspricht, ist die Zuordnung von Name zu

Adresse. Die Daten können sowohl Benutzern wie auch Applikationen durch spezielle Software (Resolver) zur Verfügung gestellt werden. Inzwischen aber wird das DNS nicht nur zur Verwaltung von Hostinformationen verwendet, sondern auch für sogenannte Mail-Exchange Informationen. Letztere ermöglichen das Steuern von elektronischer Mail zu den für das Verteilen zuständigen Rechnern (siehe hierzu Kapitel 5 auf Seite 27).

Kapitel 2

Die Komponenten des DNS

Das DNS besteht aus drei Hauptkomponenten /RFC 1034/:

- **Dem Domainnamensraum mit den entsprechenden Resource Records (RR)**
  Die RR sind Datensätze, die die Objekte des Namensraums beschreiben.

- **Den Nameservern (NS)**
  Diese sind Programme, die über Informationen eines Teils des Namensraums verfügen. Wenn ein NS komplette Informationen über einen Teil des Namensraums besitzt, kann er als Authority für diesen Namensraum bezeichnet werden (im Gegensatz zum Caching-Only-Server, der über keine "authoritative" Informationen verfügt, siehe 2.3 auf Seite 7).

- **Den Resolvern**
  Sie stellen die aufrufbare Schnittstelle für die Kommunikation zwischen einem Benutzerprozeß und einem NS dar. Sie sind in der Lage, Informationen aus Nameservern zu extrahieren und dem aufrufenden Prozeß zur Verfügung zu stellen.

2.1 Der Domainnamensraum und die Resource Records

Das Domain Name System (DNS) ermöglicht die hierarchische Strukturierung eines Namensraums. Der Namensraum wird als Baum mit unterschiedlicher "Tiefe" dargestellt; die Blätter und Knoten des Baums werden als "Label" bezeichnet. Der volle Domainnamen eines Objekts des Raums besteht aus der Verkettung aller Label auf dem Pfad vom Objekt zu der Wurzel des Baums. Die einzelnen Label werden dabei durch Punkte getrennt. Z.B. lautet der volle Namen des Rechners rusvx1 rusvx1.rus.uni-stuttgart.de. (siehe Abbildung 3.1 auf Seite 14).

Label sind Zeichenketten mit einer Länge von 0–63 ASCII-Zeichen. Sie müssen mit einem alphabetischen Zeichen[1] anfangen. Der Label mit der Länge 0 ist für die Wurzel des Baums (root) reserviert, d.h. der komplette Name eines Objekts muß mit einem Punkt abgeschlossen werden. Solche Namen werden als absolute Domainnamen bezeichnet. Die gesamte Länge eines Namens sollte 255–n nicht überschreiten (n ist die Anzahl der Label des vollen Domainnamens).

Domains sind administrative Gebilde, die eine dezentralisierte Verwaltung von Namen erlauben. Die Struktur einer Domain sollte so gestaltet werden, daß sie möglichst die Struktur der kontrollierenden Organisation reflektiert: die Namen werden mit unterschiedlicher "Tiefe" gewählt, abhängig von der Größe und Komplexität einer Organisation. Beispielsweise kann eine Universität ihren Namensraum so unterteilen, daß dieser der Strukturierung der Universität in Fakultäten bzw. Instituten entspricht.

Eine administrative Entscheidung /RFC 920/ hat die Top-Level Domains festgelegt: Es sind ISO-Länder-Codes (z.B. DE für Deutschland) und die folgenden organisatorischen Kategorien:

MIL	US-Militär
GOV	US-Regierung
EDU	Bildungswesen
COM	Kommerzielle Einrichtungen
NET	Netzwerk- und Netzwerk-Management-Organisationen
ORG	Andere "non-profit" Organisationen
ARPA	Wird nicht mehr als Teil von Rechnernamen verwendet.
INT	Internationale Organisationen. Wurde später eingeführt; wird kaum verwendet.

Die Daten, die ein Objekt im DNS beschreiben, werden als ein Satz von Resource Records (RR) dargestellt. Es gibt mehrere RR-Typen, die unterschiedliche Informationen über ein Objekt enthalten. So wird z.B. im RR vom Typ A die Zuordnung Objektname - Adresse festgehalten. RR des Typs HINFO enthalten hostspezifische Informationen wie Hardware und Betriebssystem.

Hier nun die Datenfelder eines RR:[2]

<name> [<ttl>] [<class>] <type> <rdata>

<**name**>	Der Domainname des Objekts zu dem der RR gehört.
<**class**>	Protokollgruppe (IN = Internet, CH = Chaosnet, HS = Hesiod).

[1] Diese Vorschrift wurde bereits gelockert; Label dürfen mit numerischen Zeichen beginnen, aber nicht nur aus numerischen Zeichen bestehen, um mögliche Konflikte mit IP-Adressen zu vermeiden. /RFC1123/,/RFC1101/

[2] Eine ausführliche Beschreibung der RR-Formate findet man in /RFC 1033-35/ und den Domain-Administrators Guides verschiedener Software-Hersteller.

<**type**>	RR-Typ.
<**ttl**>	time to live (in Sekunden); Zeit wie lang dieser RR gültig ist und gecached werden darf.
<**rdata**>	Daten, die das Objekt beschreiben, zu dem dieses RR gehört. Die Datenformate sind abhängig vom RR-Typ (s.u.).

Hier einige RR-Typen, die am häufigsten[3] in der NS-Konfiguration verwendet werden:

RR	Funktionalität	RDATA-Feld
A	Die Adresse eines Hosts	32-bit IP-Adresse
CNAME	Definition eines Aliasnamen zu einem Canonical Name	Domainname (Canonical Name)
HINFO	Host-Info wie Typ und Betriebssystem	CPU und Betriebssystem
MX	Mail-Exchange	16-bit preference, Name des Mailhost
NS	Ein authoritativer Nameserver	Hostname
PTR	Zeiger (pointer) zu einem Domainnamen	Domainname
SOA	Definiert "Start Of Authority" für eine Zone	mehrere Felder (NS-Name, Fehler-Mailbox, Serial-Nr. der Zonendaten, mehrere Timer)

Beispiele:

```
rusvx2.rus.uni-stuttgart.de.  86400  IN  A      129.69.1.2
rusvx2.rus.uni-stuttgart.de.         IN  HINFO  VAX8810 VMS-5.1
2.1.69.129.in-addr.arpa.      86400  IN  PTR    rusvx2.rus.uni-stuttgart.de.
ibmvm.rus.uni-stuttgart.de.          IN  CNAME  rusvm1.rus.uni-stuttgart.de.
```

2.2 Die IN-ADDR.ARPA-Domains

Das Auffinden der Adresse eines Knotens im Internet-Namensraum geschieht durch eine gezielte Durchsuchung des hierarchischen Namensbaums und ist verhältnismäßig leicht. Umgekehrt ist das Auffinden des Namens eines Knotens, dessen Adresse bekannt ist, im Prinzip nicht möglich, ohne daß der ganze Baum durchsucht wird.

Um Letzteres zu erleichtern, wurde eine Domain eingeführt, welche Adressen als Teil eines Namens benutzt, der wiederum zu dem Domainnamen des gesuchten Rechner zeigt.

[3] Alle anderen RR-Typen, insbesondere WKS, MB, MG, MINFO und MR Resource Records werden in der Regel nicht verwendet.

Die Domain, mit der das "Adresse zum Namen"-Mapping verwirklicht wird, heißt IN-ADDR.ARPA. Namen in dieser Domain dienen als eine Art invertiertes Register, das die Zuordnung der Adresse zum Namen ermöglicht. Diese Namen werden so dargestellt:

```
<invertierte-IP-Adresse>.IN-ADDR.ARPA
```

wobei <invertierte-IP-Adresse> eine IP-Adresse repräsentiert, in der die Reihenfolge der Bytes umgedreht wurde. Solche Namen werden als Zeiger nach gültigen Namen des Internet-Namensraums verwendet. Letzteres wird mit Hilfe der sogenannten Pointer(PTR)-RRs realisiert. Z.B. im folgenden RR zeigt der IN-ADDR.ARPA-Name eines Rechners mit der IP-Adresse 129.69.1.31 zu seinem gültigen Internet-Namen (canonical name):

```
31.1.69.129.IN-ADDR.ARPA.  IN PTR  rusvm1.rus.uni-stuttgart.de.
```

Im allgemeinen sind die NS für die obersten IN-ADDR.ARPA-Domains den Root-Servern bekannt.

2.3 Nameserver

Ein Nameserver ist ein Programm, das über Informationen eines Teils des Namensraums verfügt und in der Lage ist, Fragen (Queries)[4] über diese Informationen zu beantworten.

Es gibt verschiedene Kategorien von NS:

1. **Root-Server:**
   Sie sind die obersten Authorities. Sie verfügen über komplette Informationen für alle Internet Top-Level Domains und sie sind in der Lage, den zuständigen NS für eine beliebige Subdomain zu ermitteln.[5]

2. **Top-level Domainserver:**
   Sie kennen alle NS für die Second-level Domains in ihrer Zone.[6]

[4] Das Format dieser Anfragen und der dazugehörenden Antworten sind in /RFC 1035/ beschrieben.

[5] Das Leben ist aber etwas komplizierter; mindestens bis August 1990 enthielten die Root- und Top-level Server nur Informationen über NS, die zuständig für "connected"-Netze waren, d.h. Informationen von Objekten in nicht "connected"-Netzen waren/sind entweder gar nicht oder über Umwege erreichbar, siehe auch Kapitel 3.2 "Die DE-Zone" auf Seite 10.

[6] Siehe Seite 2 für die Definition einer Zone. Technisch wird eine Zone als ein zusammenhängender Teil des Domainnamensraums betrachtet, welches durch eine Datei mit den RRs der Objekte der Zone realisiert wird. Diese Datei muß dann mit einem SOA-Record beginnen.

3. **Master-Server:**
   Sie sind immer "authoritative" für eine oder mehrere Zonen. Es gibt:

   - Primary Nameserver und
   - Secondary Nameserver

   Die Primary-NS laden ihre Zonendaten von einer Datei, während die Secondary-NS vom Primary-NS ihre Daten erhalten, welche (optional) in eigenen Backup-Dateien abgespeichert werden. Jedesmal, wenn ein Secondary-NS bootet, lädt er seine Daten aus der Backup-Datei (falls vorhanden) und überprüft durch den Vergleich der Serial-Nr. des SOA-RRs des Primary mit seiner eigenen die Aktualität der Daten. Um die Probleme zu vermeiden, die mit Ausfall oder Überlastung eines Nameservers verbunden sind, ist erforderlich, daß jede Zone des Domainnamensraums neben dem zuständigen Primary Nameserver einen oder besser mehrere Secondary Nameserver betreibt.

4. **Caching-Server:**
   Alle Nameserver sind Caching-Server, d.h. sie speichern die aus anderen NS erworbenen Kenntnisse über den Domainnamensraum. Diese Daten bleiben solange erhalten, wie das ttl-Feld (time to live) in dem entsprechenden RR angibt /RFC 1034/. Ein Caching-Only-Server ist ein Nameserver ohne Authority über eine Domain und somit ohne eigenen Daten. Er beantwortet Anfragen mit Informationen aus seinem Cache oder durch Fragen von "authoritative" Nameservern. Seine Antworten sind dann als "non-authoritative" bezeichnet. Caching-Server bieten somit einen Mechanismus zur Entlastung anderen zentralen NS an.

5. **Forwarder-Server:**
   Sie sind NS mit voller Internet-Konnektivität, d.h. sie können problemlos Informationen von den Root-Servern oder anderen Master-Servern holen. Sie werden von anderen Nameservern zur "Erledigung" von Abfragen benutzt, die diese nicht beantworten können oder "wollen". Dadurch kann der Forwarder in kurzer Zeit ein großes Cache bilden, was die Antwortzeit in vielen Fällen erheblich verkürzt. Ein Nameserver sollte einen Forwarder-Server benutzen, wenn die Verbindung zu den Root-Nameservern fehlt oder wenn eine überlastete USA-Leitung von zusätzlichem NS-Verkehr entlastet werden soll.

6. **Slave-Server:**
   Ein Slave-Server hat feste Vorgaben, welche anderen NS er für Queries, die er nicht selber beantworten kann, befragen darf. Diese sind in seiner Konfiguration als Forwarder eingetragen. Ein Slave wird niemals die Root-Server oder andere ihm übergeordnete NS direkt fragen, sondern nur die als Forwarder spezifizierten NS. Ein Slave-Server kann durchaus Master-Server für eine oder mehrere Zonen sein.

Das vorliegende Handbuch handelt von der Konfiguration der Master-Server; wenn im folgenden von Primary NS, Secondary NS oder NS die Rede ist, ist ein Master-Server gemeint.

2.4 Name-Resolver

Resolver sind Programme, die in der Lage sind, Informationen aus Nameservern abzufragen. Sie stellen eine aufrufbare Schnittstelle zum NS-Dienst dar. Der Resolver befindet sich auf dem gleichen Rechner wie das Programm, welches seine Dienste benötigt, muß aber meist Nameserver fragen, die sich auf entfernten Rechnern befinden. Resolver können eigenständige Programme oder sie können an die Applikation angebunden sein. Sie ersetzen den Applikationen daher den Zugriff auf die Hostdateien.

Es gibt prinzipiell zwei Abfragestrategien im DNS: Man kann die NS Top-down fragen bis die gesuchte Antwort gefunden ist, oder man kann einen Nameserver beauftragen, diese Aufgabe zu erledigen. Resolver verwenden eben diese zwei Methoden um Nameresolution zu erreichen. Dafür brauchen sie die IP-Adresse von mindestens einem NS, der in der Lage ist, auch nicht-lokale Fragen zu beantworten. Der Resolver kann nun seine Frage stellen mit gleichzeitigem Verlangen von "recursion" oder "no recursion". Im ersten Fall muß der gefragte NS andere NS kontaktieren, bis er die Frage des Resolvers vollständig beantwortet hat. Dann schickt er die Antwort an den Resolver zurück. Im zweiten Fall gibt der gefragte NS eine Liste von Nameservern zurück, die diese Fragen beantworten könnten. Der Resolver ist dann für die weitere Namensresolution verantwortlich.

Die üblichen Resolver Implementierungen reflektieren die oben geschilderten Abfragestrategien. Es gibt sogenannte "Stub"-Resolver, die nur die rekursive Methode verwenden. Die "Full"-Resolver können auf beide Methoden zurückgreifen. Die rekursive Methode wird nur dann verwendet, wenn sowohl der Resolver wie auch der NS sich auf ihre Verwendung geeinigt haben. Resolver, die Teil einer Applikation sind, sind in der Regel "Stub"-Resolver. Dies vereinfacht die Implementierung (alles erledigt der Nameserver).

In manchen Implementierungen ist noch eine wichtige Funktion des Resolvers vorhanden: Die erwobene Informationen werden durch Caching ttl-Sekunde lang aufgehoben, was die Netzwerk- und Nameserverbelastung erheblich reduzieren kann. Resolver, die nicht eigenständige Prozesse sind, sondern Teil einer Applikation, cachen die erworbene Kenntnisse nicht, sodaß die Informationen bei Bedarf neu geholt werden müssen.

Bekannte interaktiv aufrufbare Resolver ohne Caching-Funktion sind die Programme **nslookup** und **dig**. (siehe Kapitel 6 für Beispiele und Beschreibung).

Kapitel 3

Das DNS in Deutschland

3.1 Die Top-level Domain DE

Die Internet-Domain DE[1] wird vom Institut für Informatik der Universität Dortmund (siehe "Kontaktadressen" auf Seite 17) verwaltet. Die technischen Hauptaufgaben, die mit der Verwaltung dieser Domain verbunden sind, umfassen:

- Betreiben eines Primary Nameservers sowie Koordination des Secondary Nameserver-Dienst für DE.
- Das Registrieren der Subdomainnamen (2nd Level Domainnamen) unterhalb DE. Registrieren bedeutet hier, daß die neue Subdomain zusammen mit ihren Master-Nameservern in die offiziellen Nameserver für DE eingetragen wird. Für die Auswahl der 2nd Level Domainnamen werden bestimmte Richtlinien eingehalten (siehe Abschnitt 3.3).
- Technische Beratung für Domain Administratoren.

3.2 Die DE-Zone

Um die volle Funktionalität des DNS zu gewährleisten, müssen alle Second-level Domains bei dem Verwalter der entsprechende Top-level Domain registriert sein. Für die amerikanischen und IN-ADDR.ARPA Domains ist DDN-NIC zuständig. Für Deutschland der Verwalter der Domain DE. Diese sorgen dann dafür, daß entsprechende Einträge in die Top-level Domainserver gemacht werden.

[1] DE ist der ISO-Länder-Code für Deutschland.

Nun etwas Vorgeschichte: Bis August 1990 durften bei DDN-NIC nur Domains registriert werden, wenn die IP-Adressen der Knoten dieser Domain aus den "connected"-Netzen[2] entstammten. Dies hat zu der unglücklichen Situation geführt, daß für eine Top-Level Domain zwei verschiedene Datenmengen (Zonen) existieren können: Eine Zone enthält nur "connected" Second-level Domains, während die zweite alle enthält. Die erste Zone enthält außerdem keine Pointer zu den NSn der zweiten Zone, weil diese, obwohl sie durchaus "connected" sein könnten, Informationen über NS enthalten, die keinen Internet-Anschluß haben.

Wie aus dem /RFC1174/ zu entnehmen ist, hat sich inzwischen die politische Linie der IAB (Internet Activities Board) geändert. Es scheint, daß im DNS künftig keine "Diskriminierung" mehr anhand des Vorhandenseins des "connected status" gemacht werden wird. Hier ein Auszug aus diesem RFC:

The naming and routing should be completely decoupled. In particular, it should be possible to register both a name/domain, as well as address servers within the IN-ADDR Domain, independent of whether the client has "connected" status or not. This should be implemented immediately by the IR (Internet Registry) at the DDN-NIC. ...

Diese Situation hat bis Juni 1991 den Nameservice in Deutschland geprägt und dessen Beriebsfähigkeit beeinträchtigt. Erst Ende Juni 1991 hat die Verwaltung der Domain `DE` von der oben erwähnten politischen Änderung Gebrauch gemacht und die zwei `DE`-Zonen zusammengeführt. D.h. für einen NS in Deutschland mit Internet Zugang sind keine "dirty tricks" mehr notwendig, um die Adressen der Nameserver für `DE` zu erfahren:[3] Er erfährt sie (wie für jede andere Top-Level Domain) direkt von den Root-Nameservern. Somit hat die Zusammenführung der Daten die Betriebsfähigkeit des DNS in Deutschland stark unterstützt.

3.3 Die Second-level Domains unter `DE`

Die Namen für Second-level Domains werden in Zusammenarbeit mit dem Verwalter der Top-Level Domain bestimmt. Sie sind im allgemeinen derart gewählt, daß darunter die beantragende Organisation leicht zu identifizieren ist. Die Namen entsprechen weitgehend der Empfehlung des DFN-Vereins für dessen X.400 MHS-Namensgebung.

Beispielsweise gilt für die Universität Stuttgart:

Internet: `uni-stuttgart.de`
X.400(in RFC822 Notation): `uni-stuttgart.dbp.de`

---

[2] "Connected" sind die Netze, die mit dem globalen Internet verbunden sind. Was sich genauer unter dem Term "connected" verbirgt, ist im /RFC 1174/ ausführlich erklärt.

[3] z.B. statisches Eintragen der `DE`-Nameserver im eigenen Zonen-File (siehe S. 16).

Es gibt auch Ausnahmen: Die Universitäten Braunschweig, Kaiserslautern und Saarbrücken wurden wegen der Überlänge des Städtenamens als tu-bs, uni-kl und uni-sb registriert.[4]

In den meisten Fällen verwaltet das Rechenzentrum einer Organisation den Namensraum unterhalb der Second-level Domain und bestimmt, falls notwendig, dessen weitere Unterteilung (siehe unten). Das Betreiben von richtig konfigurierten und zuverlässigen Nameservern gehört weiter zu seinen Aufgaben und ist Voraussetzung für die Registrierung seiner Second-level Domain beim Domainadministrator für `DE`.

3.4 Die weitere Aufteilung einer Second-level Domain

Der Verwalter einer Second-level Domain sollte überlegen, ob deren weitere Aufteilung in Third-level Domains sinnvoll ist. Dabei ist zu bedenken, daß jede eingeführte Subdomain eine eigenständige Domain ist und z.B. in der Boot-Datei des Master-Nameservers explizit eingetragen werden muß.

Der Vorteil von Subdomains ist, daß die Rechnernamen nur innerhalb der Third-level Domain eindeutig sein müssen und daher nicht mehr zentral verwaltet werden müssen, was bei großen Organisationen eine Erleichterung des Verwaltungsaufwands sein könnte. Die Wahl der Namen der Third-level Domains wird in Abstimmung mit dem Verwalter der übergeordneten Domain getroffen.

Wie sollen die Namen ausgewählt werden, wenn man sich für die Einführung von Third-level Domains entscheidet?

Für Hochschuleinrichtungen existiert folgende, nicht unstrittiger, Vorschlag des DFN-Vereins:

```
f(oder i).xx-yyy.de
```

wobei xx den Typ der Hochschuleinrichtung (z.B. `uni`, `tu`, `fh` `usw.`), yyy den zugehörigen Städtenamen bezeichnet. f ist eine Fakultätsbezeichnung, die möglichst in nicht-abgekürzter Form eingesetzt werden soll. i ist ein Institutsname[5] der meist in abgekürzter Form erscheint. Nach dieser Konvention besteht ein Internet-Rechnername dann aus 4 Label:

```
r.f(oder i).xx-yyy.de
```

wobei r der Rechnername ist; Beispiel: theoch.chemie.uni-stuttgart.de oder dfuws1.rus.uni-stuttgart.de.

Hier die Gründe, die für obiges Schema der Namensgebung sprechen:

- X.400 und Internet Mail-Adressen können annähernd gleich gestaltet werden.

[4] Es gibt in U.S.A. durchaus bei einigen Mail-Forwardern die Beschränkung, daß "leafs" (Labels) nicht länger als 12 Character lang sein dürfen. /SCH91/

[5] Ein Institutsname wird dann verwendet, wenn das Institut keiner Fakultät zugehört.

- Die Domainnamen, obwohl ziemlich lang, sind verständlich und aussprechbar und mindestens für deutschsprechende leicht zu merken.
- Würde der Institutsnamen als 3. Label verwendet werden, könnten bei größeren Hochschuleinrichtungen leicht über 100 Subdomains entstehen. Außerdem sind die Institutsnamen häufig sehr lang, und die Abkürzungen oft nicht eindeutig und verwirrend.

Gegen das Schema spricht:

- Die Fakultätsnamen sind oft sehr lang und für nicht deustchsprechende schwer zu merken (Beispiel: Verfahrenstechnik)
- Es gibt Software, die mit Domainnamen von einer Gesamtlänge größer als 32 Zeichen Schwierigkeiten hat.[6]
- Es ist sicher nicht möglich, ein solches Schema für verschiedene unabhängige Organisationen konsequent einzusetzen. Ein Blick auf die schon verwendeten Namen beweist dies. Die Spezifikation des DNS selbst bietet keinerlei Vorschläge und Unterstützung bezüglich einer zentralisierten und somit einheitlichen Namensgebung. Im Gegensatz dazu steht das "Gebot" der dezentralen Domainverwaltung, welches nicht nur die Durchsetzung solcher Ideen erschwert, sondern die Legitimität der oben erwähnten Versuche selbst fraglich macht.

[6] DEC ULTRIX 4.X sendmail

```
+-------+ +-------+      +-------+     +-------+  +-------+  Top-level
| com   | | edu   |      |  de   |     |  fr   |  |  nl   |  domains
+-------+ +-------+      +-------+     +-------+  +-------+
                             |
   +----------+---------+----+----+--------+---------+
   |          |         |         |        |         |
+-------+ +-------+ +-------+ +-------+ +-------+ +-------+  Second-level
| gmd   | |uni-dor| |uni-stu| |tu-berl| |fht-ess| | mpg   |  domains
|       | |tmund  | |ttgart | |in     | |lingen | |       |
+-------+ +-------+ +-------+ +-------+ +-------+ +-------+
                        |
   +----------+---------+---------+--------+---------+
   |          |         |         |        |         |
+-------+ +-------+ +-------+ +-------+ +-------+ +-------+  Third-level
|chemie | |physik | |informa| |e-techn| | rus   | | ica   |  domains
|       | |       | |tik    | |ik     | |       | |       |
+-------+ +-------+ +-------+ +-------+ +-------+ +-------+
                                           |
                                  +--------+--------+
                                  |        |        |
                              +-------+ +-------+ +-------+
                              | cray2 | |rusvx1 | |rusvm1 |  Rechner
                              +-------+ +-------+ +-------+
```

Abbildung 3.1: Die Struktur des Namensraums DE (Ausschnitt)

3.5 Die Strukturierung der IN-ADDR.ARPA-Domains

Meist geht es um IN-ADDR.ARPA-Domains für Netze der Klassen B oder C, d.h. Domains der Form:

```
x.y.z.in-addr.arpa, oder
y.z.in-addr.arpa
```

Der Verwalter solcher Domains muß diese bei DDN-NIC in den USA registrieren lassen. Voraussetzung dafür ist, wie ja auch für die Second-level Domains, die Angabe von zwei zuverlässigen Nameservern für jede Domain sowie einer Kontaktperson (siehe "Domain Registration Template"). Das Vorhandensein des "connected" Status des Netzes ist nicht mehr erforderlich. Es ist prinzipiell möglich, diesen Namensraum analog zu einer Second-level Domain in Subdomains aufzuteilen; hier entsprechen die Subdomains Subnetzen, wie z.B. `1.69.129.in-addr.arpa`. Dies würde bedeuten, daß für Netze der Klasse B mit 8-Bit-Subnetzmaske bis zu 254 neue Subdomains entstünden, was den Verwaltungsaufwand erhöhen würde. Nur wenn organisatorische Einheiten innerhalb einer Organisation bereit sind, die eigenen Subnetze völlig unabhängig von den zentralen Verwaltern (Rechenzentrum) zu verwalten und in der Lage sind, zuverlässige Nameserver zu betreiben, werden sie ermutigt, den IN-ADDR.ARPA-Dienst für die eigenen Subnetze zu übernehmen und die entsprechenden IN-ADDR.ARPA-Subdomains zu kreieren.

3.6 Informationen über andere Namensräume

Da das DNS ein verteiltes Informationssystem ist, muß ein Nameserver in der Lage sein festzustellen, wo die Antwort für eine NS-Abfrage zu finden ist. Der NS benötigt deshalb Informationen über die Struktur des gesamten Namensraums, bzw. er muß wissen, welcher NS über die gesuchte Informationen verfügt. Die Root-Server sind in der Lage, anderen Nameservern die zuständigen Authorities für eine Internet Domain mitzuteilen und diese müssen dem lokalen NS bekannt sein. Durch den Anschluß ans Internet ist der Zugang zu den Root-NS für viele Organisationen unproblematisch. Für Nameserver ohne Internet-Zugang empfiehlt sich die Angabe von einem bis zwei "forwarder". Letztere müssen aber unbedingt Internet-Zugang haben.

Bis vor Kurzem (Juni 1991) gab es noch ein zusätzliches Problem bezüglich des Zugangs zur Informationen aus dem deutschen Namensraum. Dieses Problem wurde durch die Existenz von zwei Zonen für `DE` hervorgebracht (siehe "Die `DE`-Zone" auf Seite 10):

Von den Root-Servern konnte man nur Informationen über die amerikanische `DE`-Zone abfragen, weil sie keine Verweise zu den deutschen `DE`-Nameservern enthielten. Dadurch war das Auflösen von Domainnamen einer Organisation nicht immer erfolgreich, sondern davon abhängig, ob diese Organisation Zugang zum Internet hatte und somit "connected" war. Man hatte dann das Problem durch statisches Eintragen der deutschen DE-Nameserver in die NS-Dateien für die eigene Zone gelöst.

Hier nun die Adressen der offiziellen, von den Root-Nameservern verbreiteten Nameserver für DE, Stand Anfang Okt. 1991. Diese ändern sich leider oft und ohne Vorwarnung, deshalb ist diese Liste mit Vorsicht zu genießen.

Primary	
deins.Informatik.Uni-Dortmund.DE	192.35.64.34
Secondary	
iraun1.IRA.UKA.de	129.13.10.90
Sun01a.DESY.DE	131.169.200.2
sunic.sunet.SE	130.237.216.2
	192.36.125.2
ns.UU.net	137.39.1.3
ADM.BRL.mil	192.5.25.4
	192.5.21.30
mcsun.EU.net	192.16.202.1

Zur Überprüfung der aktuellen NS für DE bzw. der root-NS siehe Kapitel 6.1.3.

3.7 Kontaktadressen

Domain Registrierung:

Deutsche Second-level Domains:	Rüdiger Volk Universität Dortmund Informatik, IRB Postfach 500 500 4600 Dortmund 50 Tel.: 0231/755-4760 Fax: 0231/755-2386 `de-domain@unido.informatik.uni-dortmund.de` oder `de-domain@Germany.EU.net` oder `de-domain@unido.uucp` oder `de-domain@unipo.bitnet`
IN-ADDR.ARPA	Government Systems, Inc. Attn: Network Information Center 14200 Park Meadow Drive Suite 200 Chantilly, VA 22021 Tel.: +1-800-365-3642 u. +1-703-802-4535 Fax: +1-703-802-8376 `hostmaster@nic.ddn.mil` oder `registrar@nic.ddn.mil`

Anmeldungsformulare können auch per anonymous ftp bezogen werden: Für DE-Subdomains von `deins.Informatik.Uni-Dortmund.DE` (Datei `IP/formulare/domain`), für IN-ADDR.ARPA-Domains von `nic.ddn.mil` (Datei `netinfo/in-addr-template.txt`).

Beratung:

Rüdiger Volk	`rv@informatik.uni-dortmund.de` Tel. 0231/755-4760

Für das Landesforschungsnetz Baden-Württemberg, BelWü:

Jürgen Georgi	`georgi@belwue.dbp.de` Tel. 0711/685-5739
Lisa Golka	`golka@rus.uni-stuttgart.dbp.de` Tel. 0711/685-5983

Kapitel 4

Die Konfiguration eines BIND-Nameservers

Hier eine Zusammenfassung einiger Richtlinien, wie ein(e) Domainadministrator(in) einen BSD-BIND Nameserver als Primary und/oder Secondary für eine oder mehrere (Sub)-domains konfigurieren kann. Die Bedeutung der Daten der verschiedenen RR, die für die Konfiguration des Nameservers notwendig sind, wird als bekannt vorausgesetzt. Man kann Näheres in mehreren Handbüchern oder RFCs nachlesen. Die Beispiele beziehen sich auf die Konfiguration eines Nameservers als Primary für eine Second-Level Domain (`uni-stuttgart.de`) und die IN-ADDR.ARPA-Domain, die der Netz-Nummer für die Universität Stuttgart entspricht.

4.1 Die Konfigurationsdateien

Der Nameserver benutzt mehrere Dateien[1], um seine Daten zu laden. Hier ein Überblick:

```
              Dateien
NS-Typ        named.boot  named.hosts  named.rev  named.ca  named.local
primary          +            +            +          +          +
secondary        +            *            *          +          +
caching-only     +                                    +          +
slave-only       +                                    +          +

Legende:         +    Datei muss vorhanden sein;
                 *    kann auch weggelassen werden.
```

[1]Die Namen dieser Dateien sind frei wählbar; hier werden diejenigen Namen benutzt, die in entsprechenden Handbüchern als "default-Namen" verwendet werden.

Abbildung 4.1: Benötigte Konfigurationsdateien für einen BIND-Nameserver

4.1.1 named.boot

- Primary / Secondary Einträge.
  In der Boot-Datei muß für jede Domain, für die ein Nameserver Master ist, ein Primary oder Secondary Eintrag mit Angabe der Datei mit den Zonen-Daten vorhanden sein. [2] Für den Secondary Eintrag ist zusätzlich die IP-Adresse eines Master-NS für `<Domain>` notwendig. Die Syntax der Primary bzw. Secondary Einträge lautet:

  ```
  primary   <Domain> <Datei>
  secondary <Domain> <IP-Adresse des Primary> <Datei>
  ```

  Ähnliches gilt für die Inverse-Domains (IN-ADDR.ARPA-Domains):

  ```
  primary   <in-addr.arpa-Domain> <Datei>
  secondary <in-addr.arpa-Domain> <IP-Adresse des Primary> <Datei>
  ```

  Z.B.:
  `primary 69.129.in-addr.arpa /etc/bind/named.rev`

  Zusätzlich ist ein Eintrag der Form

  `primary 0.0.127.in-addr.arpa /etc/bind/named.local`

  notwendig, damit das lokale loopback-Interface für den localhost bekannt wird.

- Forwarders-Eintrag:
  Ein oder mehrere Forwarder sollen eingetragen werden, wenn der Nameserver keinen Zugang zu den root-Servern hat, bzw. sie nicht verwenden soll.

  ```
  forwarders <IP-Adr. des 1. Forwarder> <IP-Adr. des 2. Forwarder>
  ```

  Dadurch wird erreicht, daß Queries über Domains, für die der Nameserver weder Primary noch Secondary ist, an den Forwarder zwecks "Erledigung" geleitet werden. Es ist empfehlenswert, bevor man einen solchen Eintrag vornimmt, den betroffenen Systemverwalter darüber zu informieren. (Belastung des Netzes/Systems, entstehende Kosten usw.)

- Cache-Eintrag:
  Muß vorhanden sein.

- Slave-Eintrag:
  Ein Slave-Server benötigt immer einen Forwarders-Eintrag.

[2] Beim Secondary Betrieb werden die Zonen-Daten automatisch vom Primary Nameserver geholt.

4.1.2 named.ca

Diese Datei wird vom NS beim Booten/Neuladen gelesen und dient dazu, die Nameserver für die Root-Zone zu finden. Sie soll die Namen und Adressen der Root-Nameserver enthalten.[3] Eine Liste der aktuellen Root-Nameserver kann per anonymous ftp von `nic.ddn.mil` (Datei `netinfo/root-servers.txt`) bezogen werden. Die derzeitige Liste (Stand Mitte Sept. 1991) ist im Anhang A.2, S. 53 in den Beispielen für die NS-Konfiguration finden.

4.1.3 named.local

Die Datei dient der Zuordnung des Domainnamens für das loopback-Interface (`localhost`) zu seiner IP-Adressse. In der Regel bleibt diese Datei so, wie sie vom SW-Hersteller geliefert wurde; die einzige notwendige Änderung ist das Eintragen des eigenen Servers als NS für die Domain `0.0.127.in-addr.arpa` (siehe Anhang A "Beispiele für BIND-Konfigurationsdateien" auf Seite 52).

4.1.4 named.hosts

Diese Datei, auch Zonen-Datei genannt, enthält die Daten für alle Knoten in einer Zone. Die Daten sind in RR-Formaten aufgeführt. Es sind A,[4] HINFO und (selten) WKS Records. Hier sind auch die NS-Records für die Zone zu finden sowie, falls notwendig, Mailrouting Informationen (MX-Records).

An erster Stelle in der Datei soll ein SOA-Record für die entsprechende Domain mit einer Serialnummer > 0 stehen. Es ist sehr wichtig, nach jeder Änderung der Zonendaten (in named.hosts bzw. in eventuell vorhandenen include-Dateien) die Serialnummer des SOA-Records zu erhöhen, damit andere NS, die als Secondary für diese Domain fungieren, die Änderungen erfahren können. Die von Handbüchern empfohlen Punktnotation für die Serialnummer sollte besser nicht verwendet werden. BIND setzt solche Serialnummern in einigen Fällen falsch in Integerwerte um /BEER 91/; es kann sich dann eine Zahl ergeben, die kleiner ist als in der früheren Datei.[5] Man kann Zahlen bis ca. 4.3E +9 verwenden, da die Serialnummer durch einen 32-bit Integer ohne Vorzeichen dargestellt wird. Gute Erfahrungen haben wir mit dem Format jjjjmmddv (JahrMonatTagVersion, Beispiel 199107310) gemacht.

[3] Bei den meisten NS-Implementierungen werden beim Starten des NS die hier eingetragenen Root-NS auf Gültigkeit und Vollständigkeit überprüft.

[4] Neben den A-RRs für alle Rechner der Zone sollte auch ein A-RR für den qualifizierten Domainnamen des loopback-Interfaces (`localhost.<domain>`) eingetragen werden.

[5] BIND ersetzt den Punkt in der Serialnummer durch drei Nullen. Zum Beispiel wird aus 1.95 die Zahl 100095, während 2.1 in 20001 umgewandelt wird. Da 20001 jedoch kleiner ist als 100095, findet kein Zonen-Update statt. Man darf also die Anzahl der Stellen hinter dem Punkt nie verringern.

In den meisten Handbüchern und auch im /RFC 1033/ werden zu kleine Werte für die verschiedenen Timer empfohlen. Dies führt zu einer unnötigen Belastung des Netzes und insbesondere der (inter)nationalen Verbindungen. Hier ein Vorschlag:

Refresh:	21600	(6 Stunden)
Retry:	3600	(1 Stunde)
Expire:	604800	(7 Tage)
Default TTL (Minimum):	172800	(2 Tage)

Falls die Second-Level Domain in weitere "richtige"[6] Subdomains unterteilt wurde, muß eine "`named.hosts`" Datei analog zu dem obigen bestehen. Folgendes ist auf jeden Fall zu beachten:

- Um einen Zonen-Transfer für die Parent-Domain samt allen Subdomains zu erreichen, ist es notwendig, daß der Secondary-NS in seiner Boot-Datei Anweisungen folgender Form **pro** SOA-Record des Primary NS enthält:

  `secondary <subdomain> <ip-adresse> <Datei>`

  Regel: Zu jeder primary/secondary-Anweisung muß genau ein SOA-RR vorhanden sein und umgekehrt.

- Sehr wichtig ist auch das vollständige Auflisten aller Masterdomainserver (Primary und Secondary) für jede durch einen SOA-Record definierte Subdomain. Dies geschieht durch entsprechende NS-Records in den Zonendateien der Subdomains (siehe auch Abschnitt 4.1.6); z.B.:

```
rus.uni-stuttgart.de. IN SOA rusmv5.rus.uni-stuttgart.de. golka.rus.uni-stuttgart.de. (
                        199108190  ; serial
                        21600      ; refresh
                        3600       ; retry
                        604800     ; expire
                        172800 )   ; minimum
                      IN NS  rusmv8.rus.uni-stuttgart.de.
                      IN NS  rusmv5.rus.uni-stuttgart.de.
                      IN NS  noc.belwue.de.
```

Nur so kann die im /RFC 1034/ verlangte Nameserverredundanz gewährleistet werden.

4.1.5 named.rev

Die Datei enthält die Daten für eine IN-ADDR.ARPA-Subdomain, welche die Zuordnung IP-Adresse zum Domainnamen ermöglichen. Der Aufbau der Datei ist ähnlich wie bei

[6] "Richtige" Subdomains sind Zonen: sie beginnen mit einem SOA-Record und enthalten NS-Records für die entsprechenden Nameserver.

`named.hosts`. Ein SOA-Record für jede Netznummer (nicht Subnetznummer) ist absolut notwendig:

```
69.129.in-addr.arpa.  IN SOA rusmv8.rus.uni-stuttgart.de ...
```

Falls eine weitere Aufteilung dieser Domain notwendig sein sollte (in welchem Fall so etwas sinnvoll ist, wurde in Kapitel 3.5 auf Seite 15 diskutiert), müssen weitere Subdomains durch SOA-Records definiert werden und die entsprechende Zonen-Dateien vorhanden sein.

4.1.6 Das Delegieren von Subdomains und Glue Records

Wie kann man die Verwaltung einer Subdomain an eine andere Institution vergeben? Voraussetzung dafür ist, daß diese Institution mindestens über einen zuverlässigen Nameserver verfügt. Als zweiter Nameserver könnte der NS dienen, der für die übergeordnete Domain verantwortlich ist.

Man muß folgende Einträge in der named.hosts-Datei des NS, der die Parent-Domain verwaltet, vornehmen:

<abgetrennte Subdomain>	IN	NS	<Nameserver>
<Nameserver>	IN	A	<Nameserveradresse> ; ("glue" record)

z.B.

```
luftfahrt.uni-stuttgart.de.  IN NS majestix.luftfahrt.uni-stuttgart.de.
luftfahrt.uni-stuttgart.de.  IN NS noc.belwue.de.
majestix.luftfahrt.uni-stuttgart.de.  IN A 129.69.57.2 ; ("glue" record)
```

Nur in der Zonendatei des delegierenden Nameservers sind "Glue Records" notwendig, und zwar nur dann, wenn der Nameserver für die Subdomain zu der delegierten Zone gehört. Der Grund dafür ist, daß ohne "Glue Records" es nicht möglich ist, die Adresse des NS der delegierten Domain zu finden. Zum Beispiel würden wir mit einer Query nach der Adresse des Rechners `xx.luftfahrt.uni-stuttgart.de` nur den NS-Record für die Domain `luftfahrt.uni-stuttgart.de` (und keinen A-Record) von der `rusmv8` erfahren. Es wäre daher unmöglich, diesen NS zu erreichen, um den gesuchten RR für `xx.luftfahrt.uni-stuttgart.de` zu erhalten. Für Nameserver, die nicht in der gleichen Domain sind, dürfen keine A-Records vorhanden sein. Hierfür muß der zuständige NS befragt werden. Ein A-Record (Glue Record) für die `noc.belwue.de` ist daher nicht notwendig.

Ähnliches gilt für die Abtrennung von IN-ADDR.ARPA-Domains: z.B. braucht man für das Subnetz 129.69.57.0 :

```
57.69.129.in-addr.arpa.  IN NS majestix.luftfahrt.uni-stuttgart.de.
```

Es empfiehlt sich, den delegierenden Nameserver zusätzlich als Secondary NS für jede seiner delegierten Subdomains einzurichten, z.B. für die IN-ADDR.ARPA-Domains aller

(Sub)netze, die an andere NS abgegeben wurden. Dadurch ist der delegierende Nameserver für seine Subdomains weiterhin authoritativ, ohne jedoch dafür direkt verantwortlich zu sein. Zudem kann die Funktionsfähigkeit des Nameservices in den Subdomains vom Administrator des delegierenden Nameserver aus gut überwacht werden.[7]

4.1.7 Einige Tips und Warnungen

Hier einige Punkte, die man bei der Konfiguation und Betrieb eines BIND-Nameservers berücksichtigen sollte:

Zonen-Transfer

Unsere Erfahrungen mit dem Secondary-Mechanismus: Die ganze Angelegenheit scheint uns softwareabhängig zu sein. Es gab Fälle, allerdings mit BIND < 4.8, bei denen der Secondary NS bei einem mißglückten Zonen-Transfer Dateien mit der Länge Null angelegt hat. Nach dem Reboot hat er die leeren Dateien ohne zu protestieren gelesen, was zu folgendem Ergebnis führte: Er meinte, es gäbe keine Daten für die betroffene Subdomain und beantwortete jede Query mit der Meldung "no information". Also Vorsicht: Ab und zu sollte man die Dateien kontrollieren. Sind sie fehlerhaft, sollte man sie löschen und anschließend den Nameserverprozeß stoppen und wieder neu starten, um einen Zonen-Transfer zu erzwingen. Falls es wieder zu einem fehlerhaften Transfer kommt, sollte man sich mit dem Administrator der fremden Domain in Verbindung setzen, um abzuklären, ob nicht dort der Fehler zu finden sei.

Forwarder

Nicht bei allen Nameservern ist eine forwarder-Anweisung möglich; z.B. nicht mit IBM AIX-PS/2 /HEB90/.

secondary-Einträge

Die Angabe eine Backup-Datei ist manchmal nicht möglich; dies trifft bei IBM AIX-PS/2 zu /HEB91/.

named.ca

In diese Datei gehören nur Einträge über die aktuellen Root-Nameserver. Sie ist nicht zur Initialisierung des Cache mit zonenfremden Resource Records vorgesehen. Auf die Aktualität dieser Datei sollte geachtet werden.

Serialnummer

Falls die Serialnummer eines SOA-Records zu groß oder auch negativ wird: Geben Sie der Serialnummer wieder auf einen gültigen Wert und setzen Sie sich mit allen Administratoren der Secondary Nameserver für die zugehörige Domain in Verbindung. Diese müssen die

[7] z.B. durch Überprüfung des Refresh-Verhaltens und durch Inspektion der Zonendaten.

entsprechenden Backup-Dateien löschen, den Nameserverprozeß stoppen und wieder neu starten. Es sollte danach ein Zonen-Transfer stattfinden.

Absolute Domainnamen

Gewöhnen Sie sich daran, alle Domainnamen mit einem Punkt abzuschließen (absolute Domainnamen). Da weiß man, was man hat.

IN-ADDR.ARPA-Domains

Sie **müssen** bei DDN-NIC (Anschrift siehe Kontaktadressen auf S. 17) angemeldet werden. Die Anmeldung geschieht nicht automatisch mit dem Antrag auf eine Netznummer.

Groß-/Kleinschreibung der Domainnamen

Das DNS unterscheidet nicht zwischen Groß- und Kleinschreibung bei den Domainnamen. Das heißt, die Namen `TU-Berlin.de` und `tu-berlin.DE` sind gleich. Die Groß- bzw. Kleinschreibung wird trotzdem in der Regel beibehalten. Man kann daher durch eine mixed-case Schreibweise die Lesbarkeit langer Domainnamen erhöhen.

Include-Dateien

Es sei erwähnt, daß manche Nameserver Include-Anweisungen in den Konfigurationsdateien nicht erkennen. Andere wieder erwarten, daß diese Dateien Zonen-Dateien sind (d.h. mit einem SOA-Record beginnen), und bringen eine Fehlermeldung, wenn dies nicht der Fall ist. Aus diesem Grund wurde im vorliegenden Dokument von der Verwendung der Include-Anweisung in den NS-Konfigurations-Dateien Abstand genommen.

Origin-Anweisung

Diese Anweisung in einer Zonendatei dient dazu, die "origin", d.h. die angehängte Domain in einem nicht-absoluten Domainnamen zu ändern. Z.B. kann die Datei named.rev für die Domain 69.129.in-addr.arpa wie folgt strukturiert werden:

```
$origin 1.69.129.in-addr.arpa.      ; abschliessenden Punkt beachten
1       IN  PTR  rusvx1.rus.uni-stuttgart.de.
2       IN  PTR  rusvx2.rus.uni-stuttgart.de.
; ...
$origin 11.69.129.in-addr.arpa.     ; abschliessenden Punkt beachten
30      IN  PTR  rusc2.rus.uni-stuttgart.de.
; ...
$origin 45.69.129.in-addr.arpa.     ; abschliessenden Punkt beachten
1       IN  PTR  sirmel.physik.uni-stuttgart.de.
; ...
```

Wir raten allerdings vom Gebrauch dieser Anweisung ab, da hierdurch die Übersicht besonders bei großen Domains leicht verloren geht.

4.2 Resolver

Damit ein Rechner Zugang zum Nameserver-Dienst hat, muß er entweder lokal einen Nameserver betreiben, oder Anfragen an einen Nameserver auf einem entfernten Rechner stellen können. Letzteres bewirkt bei UNIX-Rechnern ein Eintrag in der Datei `/etc/resolv.conf`. Achtung: Die Zugriffsberechtigung für diese Datei muß das Lesen für jedermann (z.B. UNIX permission mode 644) erlauben. Beispiel:

```
nameserver 129.143.2.1
nameserver 129.69.1.9
nameserver 129.69.1.8
domain belwue.de
```

Ein `nameserver`-Eintrag spezifiziert die Adresse eines Rechners, der als Default-Nameserver von Resolvern in Anspruch genommen werden kann. Es empfiehlt sich, mehrere Nameserver-Einträge[8] zu machen (max. 3) für den Fall, daß ein Nameserver nicht erreichbar sein sollte. Der `domain`-Eintrag gibt eine Default-Domain an. Sie wird von Resolvern an Anfragen mit Namen angehängt, die nicht mit einem Punkt abgeschlossen sind.

Damit auf einem Rechner mit Nameserver-Zugang Applikations-Programme wie ping, ftp, telnet, sendmail etc. automatisch von den Nameserverdiensten Gebrauch machen können, müssen manchmal zusätzliche Einstellungen/Modifikationen des Betriebssystems oder der betreffenden Applikationen vorgenommen werden. Im folgenden werden einige Möglichkeiten skizziert, die jedoch stark von den Eigenheiten des verwendeten Betriebsystems abhängen.

1. Konfigurierbare Resolver-Schnittstellen. Die Abbildung Rechnername/IP-Adresse und umgekehrt wird von UNIX-Systemroutinen[9] ausgeführt, deren Funktionsweise auf manchen Rechnertypen einstellbar ist: Der Systemverwalter kann wählen, ob die gesuchte Information aus der Datei /etc/hosts, oder aus den NIS-Maps[10] oder über Nameserver-Abfrage ermittelt werden soll und in welcher Reihenfolge dies erfolgt.[11]

[8] z.B. ein Primary NS für die eigene Second-Level Domain und dessen Secondary Nameserver.

[9] die wichtigsten Routinen hierfür sind `gethostbyname(3)` und `gethostbyaddr(3N)`.

[10] NIS bedeutet "Network Information Service" (ehemals "Yellow Pages") und bezeichnet das Konzept einer verteilten Datenbank mit Host- und Password-Informationen für den Einsatz in lokalen Netzen.

[11] Bei DEC Ultrix und Silicon Graphics IRIX 3.3 ist die beschriebene freie Konfigurierbarkeit vorgesehen, SUN implementiert ein starreres Konzept: Zuerst werden die NIS-Maps konsultiert, anschließend /etc/hosts. Bei der NIS-Konfiguration kann jedoch spezifiziert werden, daß zusätzlich DNS-Abfragen gemacht werden, wenn im NIS die gesuchte Information nicht gefunden wird. Ein auf BIND 4.8.3. basierendes Patch der Shared Library Resolver-Routinen für SunOS 4.1 ermöglicht die Konfigurierung der Resolverschnittstelle auch ohne Rückgriff auf NIS. Quelle: Anonymous FTP Server `next.ucns.uga.edu` (File `pub/sunfixes/resolv+.tar`).

2. Austausch bzw. Neucompilierung von Netzwerk-Applikationsprogrammen. Hierbei werden lediglich die Applikationen durch solche ersetzt, welche zur Nameserver-Abfrage fähig sind. In manchen UNIX-Varianten gehören die DNS-fähigen Versionen der Programme schon zum Lieferumfang – SUNs SunOS z.B. beinhaltet `sendmail.mx`, eine `sendmail`-Version, welche MX-Records auswerten kann. Bei einer Neucompilierung muß neben den Quellen eine Bibliothek mit modifizierten Resolver-Routinen (z.B. `/usr/lib/libresolv.a`) zur Verfügung stehen.

3. Modifikation der relevanten System-Bibliotheken. Hierbei werden in der System-Bibliothek `/usr/lib/libc.a` die oben genannten Routinen zur Adress- und Namensauflösung durch die entsprechenden Resolver-Routinen ersetzt. Anschließend müssen die betroffenen Applikationen neu compiliert bzw. gebunden werden. Bei Betriebssystemen mit "shared libraries" führt die Modifikation der betroffenen Bibliothek (in der Regel `/usr/lib/libc.so.*`) auch ohne Neukompilierung von Programmen zu dem gewünschten Effekt, da die Resolver-Routinen erst zur Laufzeit an die Applikationen gebunden werden.

Kapitel 5

Mailrouting und das DNS

Das Internet Simple Mail Transfer Protocol (SMTP) stellt eine Ende-zu-Ende-Verbindung auf der Basis von TCP/IP zwischen sendendem umd empfangendem Mailsystem her. Eine solche Verbindung garantiert, daß die zu verschickende Nachricht solange auf dem Absender-Rechner verweilt, bis sie erfolgreich auf den Empfänger-Rechner übertragen werden kann.

Zur Adressierung der Empfänger-Rechner werden DNS-konforme Domainnamen verwendet. Eine Mailadresse besteht generell aus einem lokalen Teil (Mailbox, User-ID) und einem Domain-Teil (Domainname des Rechners):

<Local-part>@<Domain>

Zum Beispiel:

`postmaster@noc.belwue.de`

Die Aufgabe eines Mailsystems ist es, anhand der syntaktischen Analyse einer Zieladresse zu entscheiden, auf welche Weise die dazugehörige Nachricht zu verschicken ist. Hier kann man grob zwischen drei Fällen unterscheiden:

- <Domain> ist identisch mit dem Domainnamen des sendenden Rechners. Die Nachricht wird lokal ausgeliefert. Es wird das Netzwerk überhaupt nicht in Anspruch genommen.

- <Domain> ist ein Domainname eines Rechners, dessen IP-Adresse aus der eigenen Host-Tabelle entnommen werden kann. Danach wird eine Verbindung zum SMTP-Port des Zielrechners hergestellt.

- <Domain> ist dem sendenden Rechner zunächst unbekannt und das Mailsystem muß die IP-Adresse des Zielrechners über eine Nameserverabfrage auflösen. Danach wird eine Verbindung zum SMTP-Port des Zielrechners hergestellt.

Im letzten Fall wird deutlich, wie stark erfolgreiches Mail-Routing von einem zuverlässig betriebenen DNS abhängt. Ein Nameserver, der unvollständige bzw. falsche Informationen verbreitet oder erst gar nicht in Betrieb ist, kann die Ursache dafür sein, wenn E-Mail nicht an der Zieladresse ankommt.

5.1 MX Records

Die Spezifikation des DNS definiert für besondere Zwecke des Mail-Routings Mail Exchanger (MX) Resource Records.[1] Ein MX-Record spezifiziert einen Domainnamen und einen zugehörigen Rechner – den Mail Exchanger –, der in der Lage ist, Mail an eine E-Mailadresse mit diesem Domainnamen[2] auszuliefern. Es können mehrere Mail Exchanger für den gleichen Domainnamen angegeben werden. Mailsysteme können so konfiguriert werden, daß sie einen Nameserver nach MX Einträgen für den Domain-Teil einer gegebenden Zieladresse abfragen.[3] Die Syntax von MX-Records lautet:

<name> [<ttl>] [<class>] MX <preference value> <mail exchanger>

Beispiel:

```
belwue.de.     IN     MX      50   noc.belwue.de.
               IN     MX     100   ncc.belwue.de.
*.belwue.de.   IN     MX      50   noc.belwue.de.
               IN     MX     100   ncc.belwue.de.
```

Die ersten beiden MX-Records definieren `noc.belwue.de` und `ncc.belwue.de` als Mail Exchanger für den Domainnamen `belwue.de`. Eine an `postmaster@belwue.de` adressierte E-Mail wird an den Rechner `noc.belwue.de` geschickt, der dann für das weitere Ausliefern zuständig ist. Der Term <preference value> gibt die Reihenfolge an, die von einem Mailer befolgt werden muß, wenn mehrere Mail Exchanger für einen Domainnamen existieren. Mail Exchanger mit kleineren Präferenzwerten müssen bei der Verbindungsaufnahme bevorzugt werden. Falls z.B. der Rechner `noc.belwue.de` nicht erreichbar sein sollte, wird die E-Mail an `ncc.belwue.de` geschickt, da dieser Rechner den nächst höheren Präferenzwert aufweist. Sind mehrere Mail Exchanger mit gleichen Präferenzwerten angegeben, kann ein Mailer den Mail Exchanger, der die E-Mail vorrangig geliefert bekommt, zufallsgesteuert auswählen – der Auswahlalgorithmus ist hierbei nicht mehr festgelegt. Ist für <name> kein Mail Exchanger definiert, d.h. es gibt dazu keinen MX-Record, versuchen Mailsysteme, die Nachricht direkt an den mit <name> bezeichneten

[1] MB, MG, MINFO und MR Resource Records sind alternative Konzepte, die sich jedoch nicht durchgesetzt haben.

[2] Der Domain-Teil einer E-Mailadresse braucht kein Domainname eines existierenden Rechners zu sein. Siehe auch Abschnitt 5.2.1.

[3] WKS Records werden von allen bekannten SMTP-Systemen bei der MX-Verarbeitung nicht berücksichtigt. Hierzu /RFC-974/ und /RFC-1123/.

Rechner auszuliefern, falls für diesen ein A-Record definiert wurde. Diese Tatsache soll den/die Domainadministrator/in jedoch nicht veranlassen, MX-Records für Rechner zu unterschlagen, wenn A-Records definiert sind. Der geringere Arbeitsaufwand für die Pflege der DNS-Datenbasis wird durch eine höhere Belastung des Nameservers erkauft, da bei fehlendem MX-Record die Auflösung der Rechneradresse eine zusätzliche DNS-Abfrage erfordert.

Der zweite Satz vom MX-Records im obigen Beispiel verwendet Namen mit Wildcards (*). In diesem Fall kann für die Menge von Domainnamen, die in hinteren Teil die Zeichenkette `belwue.de` aufweisen (z.B. `nic.belwue.de`), ein gemeinsamer Mail Exchanger spezifiziert werden. Es ist darauf hinzuweisen, daß die Wildcard * auch für mehrere Labels Platzhalter sein kann (z.B. für den Rechner `bingo.foo.belwue.de` in der Subdomain `foo` von `belwue.de`). Desweiteren schließt `*.belwue.de` nicht (!) den Namen `belwue.de` ein. Wildcards verlieren jedoch ihre Wirkung,

- wenn die gefrage Zieldomain zu einer delegierten Zone gehört. Z.B. gilt das Muster `*.belwue.de` in einem MX RR nicht für den Domainnamen `info.stgt.belwue.de`, falls `stgt.belwue.de` eine delegierte Subdomain von `belwue.de` ist.
- wenn die gefragte Zieldomain oder ein Name zwischen der Wildcard Domain und der gefragten Zieldomain in einem Resource Record gleich welcher Art definiert wurde.

Der Algorithmus für die Erweiterung von Wildcards lautet somit:[4]

```
IF      there is ANY sort of RR for the machine
THEN    IF      there is an non-wildcard MX for it
        THEN    use those in order
        ELIF    there is a A record (or records)
        THEN    use it (/them)
        ELSE    fail
        FI
ELIF    there is a wildcard MX record
THEN    use it
ELSE    fail
FI
```

Hierzu ein Beispiel:

```
*.belwue.de.      IN    MX       50    noc.belwue.de.
noc.belwue.de.    IN    A              129.143.2.1
```

[4] Der Algorithmus wurde von `Piete.Brooks@cl.cam.ac.uk` auf der sun-nets Mailing Liste veröffentlicht. Weitere Einzelheiten zur Interpretation von Wildcard MX-Records sind in /RFC1034/ S. 25–26 zu finden.

```
                 IN    MX       50   noc.belwue.de.
foo.belwue.de.   IN    MX       50   bar.belwue.de.
bar.belwue.de.   IN    A             129.143.2.14
                 IN    MX       50   bar.belwue.de.
```

Der Mail Exchanger `noc.belwue.de` darf keine E-Mail für die Rechner `foo.belwue.de` oder `bingo.foo.belwue.de` abnehmen, auch nicht für den Fall, daß `bar.belwue.de` nicht betriebsbereit wäre. Er akzeptiert darüberhinaus auch nicht E-Mail an die Adressen `bar.belwue.de` oder `smurf.bar.belwue.de`, da für den Namen `bar.belwue.de` ein A-Record existiert. Wird jedoch gewünscht, daß alle E-Mail an Adressen innerhalb der Domain `noc.belwue.de` – mit Ausnahme von `foo.belwue.de` und `bar.belwue.de` – von dem MXer `noc.belwue.de` in Empfang genommen wird, sind weitere MX Records notwendig:

```
*.belwue.de.     IN    MX       50   noc.belwue.de.
noc.belwue.de.   IN    A             129.143.2.1
                 IN    MX       50   noc.belwue.de.
foo.belwue.de.   IN    MX       50   bar.belwue.de.
*.foo.belwue.de. IN    MX       50   noc.belwue.de.
bar.belwue.de.   IN    A             129.143.2.14
                 IN    MX       50   bar.belwue.de.
*.bar.belwue.de. IN    MX       50   noc.belwue.de.
```

Das obige Beispiel zeigt, daß Wildcard MX-Records oft nicht die Wirkung zeigen, die man erwartet. Sie sollten nur mit großer Vorsicht verwendet werden.

5.2 Anwendungsbeispiele

Wie oben schon erwähnt, kann trotz fehlendem MX-Record für eine gegebene Zieladresse eine SMTP-Verbindung zustande kommen. Voraussetzung dafür ist, daß für die Zieladresse im DNS ein A-Record definiert wurde, was bedeutet, daß sie einen IP-adressierbaren Rechner repräsentiert. Zusätzlich muß der Zielrechner eine spontane SMTP-Verbindung akzeptieren können, d.h. ein ständig "empfangsbereites" Mailsystem betreiben.[5] In allen anderen Fällen werden MX-Records notwendig. Einige typische Beispiele für den Einsatz von MX-Records werden im folgenden vorgestellt.

[5]Genauer gesagt muß das Mailsystem des Zielrechners einen "Server"-Prozeß betreiben, der ständig auf eintreffende SMTP-Verbindungen wartet und die zu übertragenden Daten auch unmittelbar abnehmen kann – selbst bei mehreren simultan aufgebauten Verbindungen. Hierzu sind insbesondere die Mailsysteme von Personal Computern meist nicht in der Lage.

5.2.1 "Offizielle" bzw. "bereinigte" Mailadressen

Häufig werden in Organisationen Mailadressen der Mitarbeiter so gewählt, daß keine Rechnerinformationen darin vorkommen. [6] Hierzu muß ein Mail Exchanger bereitgestellt werden, der alle Nachrichten an "bereinigte" Mailadressen innerhalb der Organisation weiterverteilen kann. Ebenso kann dafür gesorgt werden, daß alle aus der Organisation ausgehende Mail mit einer bereinigten Absenderadresse versehen wird.[7] Offizielle Mailadressen erlauben die Definition von Adressierungsregeln, mit denen aus Vor- und Nachnamen von Benutzern zusammen mit dem Domainnamen der Organisation Mailadressen (z.B. `donald.duck@disneyland.com`) generiert werden können, die den Vorteil aufweisen, nach außen hin auch dann gültig zu bleiben, wenn der Benutzer seinen Rechner wechselt.

MX-Records können auch zum Schutz von Rechnern eingesetzt werden, für die aufgrund von Sicherheitrisiken im Nameserver keine Einträge (A-Records, PTR-Records) gemacht werden, und deren IP-Adresse geheim gehalten werden muß. Ein solcher "verborgener" Rechner ist im Internet nur durch E-Mail erreichbar, nicht aber durch telnet oder ftp. Die Adressierung erfolgt wie oben über eine vom Rechnernamen bereinigte Mailadresse, die auf einen Mail Exchanger zeigt.

5.2.2 Mail-Gateway-Routing

Im weltweiten Netzwerkverbund ist E-Mail zur Zeit die einzige Kommunikationsform, die auch über die Grenzen der verschiedenen existierenden Protokollwelten hinweg funktioniert. So kann man z.B. von einem SMTP-Host aus Nachrichten an BITNET-, X.400- oder UUCP-Adressen verschicken. Die Vermittlung zwischen den unterschiedlichen Mailprotokollen erledigen Mail-Gateways. Die lange Zeit übliche Praxis, Angaben zum Gateway-Routing in Mailadressen unterzubringen[8] überfordert viele E-Mail-Benutzer und führt zudem zu Zieladressen, die vom jeweiligen Standpunkt des Absenders abhängig sind. Wesentlich eleganter ist es, die Routing-Information im DNS in Form von MX-Records abzulegen: Für eine Adresse in der fremden Protokollwelt existiert ein MX-Record, der auf ein Mail-Gateway zeigt, daß für die Konvertierung und Weiterleitung der Nachricht verantwortlich ist. Ein Benutzer muß nur noch wissen, in welcher Weise eine Mailadresse der fremden Protokollwelt in der RFC822-Notation notiert wird. Beispiel:

```
*.uni-stuttgart.dbp.de.    IN      MX       10      noc.belwue.de
```

[6] Im obigen Beispiel wird die Mailadresse `postmaster@belwue.de` verwendet, wobei der Domainname `belwue.de` kein Name eines Rechners ist.

[7] Dies kann entweder durch das Mailsystem an jedem lokalen Rechner oder an einem zentralen "Mailhost" durchgeführt werden. Im letzteren Fall bietet sich die Integration von Mail Exchanger/Mailhost an.

[8] Z.B. `user%final-host@gateway-host` oder `hop1!hop2!hop3!user@gateway-host`.

Hinter dem Namensraum `uni-stuttgart.dbp.de` stehen alle X.400-Adressen innerhalb der Universität Stuttgart (C=de, ADMD=dbp und PRMD=uni-stuttgart).[9] Der Mail Exchanger `noc.belwue.de` betreibt ein SMTP/X.400 Gateway, welches alle Nachrichten an diese Adressen zu den zugeständigen X.400 MTAs weiterleitet. Leider läßt sich die Routing-Information zu Mail-Gateways nicht immer so einfach im DNS unterbringen. Insbesondere bei den flachen Namensräumen von BITNET/EARN und UUCP sind Mailgateways nicht über weltweit gültige MX-Records ansprechbar.[10]

5.2.3 Spezifikation eines "Fallback"-Mailhosts

Wenn ein Rechner am Internet nicht in Betrieb ist, kann mit ihm keine SMPT-Verbindung aufgenommen werden. Eine Nachricht, die an solch einen Rechner adressiert ist, wird vom sendenden Rechner in der Regel ca. drei Tage in einer Warteschlage gehalten. Während dieser Zeit wird periodisch versucht, die Mail an den Zielrechner auszuliefern. Gelingt dies nicht, erhält der Absender seine Nachricht zusammen mit einer Fehlermeldung zurück. Mit einem MX-Record kann für einen nur sporadisch betriebenen Rechner ein Mailhost spezifiziert werden, der während der Stillstandzeiten die Nachrichten stellvertretend in Empfang nimmt. Beispiel:

```
small-pc.belwue.de.       IN      MX      50     small-pc.belwue.de.
                          IN      MX      100    mailmaster.belwue.de.
```

Ist der Rechner `small-pc.belwue.de` nicht erreichbar, springt für ihn der Rechner `mailmaster.belwue.de` ein, und bewahrt die Nachrichten für `small-pc` solange auf, bis dieser empfangsbereit ist. Auf welche Weise `small-pc` von `mailmaster` die aufbewahrten Nachrichten empfängt, kann von Fall zu Fall verschieden sein.[11] Einzelheiten darüber gehen über den Rahmen dieses Handbuchs hinaus.

Das Fallback-Konzept bietet sich auch zur Absicherung von Rechnern an, die in einer großer Organisation zentrale Kommunikationsdienste wie z.B. Mailhost, Mailgateway übernehmen:

```
noc.belwue.de.            IN      MX      50     noc.belwue.de.
                          IN      MX      100    ncc.belwue.de.
```

[9] Die Abbildung zwischen der O/R- und RFC822-Notation von X.400-Adressen spezifiziert /RFC987/.

[10] Es ist im Prinzip möglich, mit einem Wildcard MX-Record ein lokales SMTP/BITNET- oder SMTP/UUCP-Gateway zu spezifizieren. Dann aber muß gewährleistet sein, daß diese Information ausschließlich von solchen Rechnern abfragbar ist, die sich im "Einzugsbereich" des Gateways befinden. Wir raten jedoch davon ab.

[11] Z.B. könnte `mailmaster` eine besonders lange timeout-Zeit für die Weiterleitung der Mail verwenden, oder er könnte sie in einer lokalen Mailbox ablegen, auf `small-pc` bei gegebener Zeit über das "Post Office Protocol" /RFC 1081/ zugreifen kann.

Der Rechner noc.belwue.de fungiert als Mailhost für eine Großzahl von Workstations auf dem Campus der Universität Stuttgart. Eine Mail an eine nicht-lokale, d.h. nicht in der selben Domain des Absenders liegende Adresse wird zuerst an den Mailhost geschickt, der sie entsprechend seiner Routingstrategien an den Zielrechner oder an ein Mailgateway weiterleitet. Beim Ausfall des Mailhosts noc.belwue.de würde für eine große Zahl von Rechnern die Mail-Verbindung zur Außenwelt unterbrochen werden. Für diesen Fall steht der Fallback-Mailhost ncc.belwue.de bereit, der in diesem Fall die Dienste von noc.belwue.de übernehmen kann. Voraussetzung für den dynamischen Ersatz des Mailhosts ist die Fähigkeit des Mailsystems der lokalen Rechner, die IP-Adresse des Mailhosts über DNS-Abfragen aufzulösen.

5.3 Fehlerquellen

Im folgenden werden einige Fehlerquellen bei der Verwendung von MX-Records aufgezeigt, die für das Mail-Routing Überraschungen hervorrufen können.

Alias-Namen in MX-Records. Der Algorithmus zur Elimination von irrelevanten bzw. verbotenen Mail Exchangern /RFC974/ versagt, wenn in MX-Records Alias-Namen [12] verwendet werden. Dies kann zu Mail-Loops führen. Um derartige Probleme von vornherein auszuschließen, sollten Alias-Namen in MX-Records prinzipiell vermieden werden.

Verkettete MX Records. Eine Anordnung von MX-Records der Form:

```
*.foo.de.            IN    MX    0    host-a.foo.de.
host-a.foo.de.       IN    MX    0    host-b.bar.de.
```

hat die Wirkung, daß alle Nachrichten für Rechner in der Domain foo.de über den Mail Exchanger host-a.foo.de weitergeleitet werden. Der Rechner host-b.bar.de erhält nur Nachrichten, die an host-a.foo.de direkt adressiert sind, nicht jedoch Nachrichten, die an Rechner in der Domain foo.de adressiert sind.

Mail Exchanger mit Default-Mailhost. Es ist zulässig, daß ein Mail Exchanger mit einem Default-Mailhost [13]

kooperiert. In diesem Fall ist das Mailsystem des Mail Exchangers so konfiguriert, daß es ausgehende Nachrichten an nicht-lokale Adressen – d.h. Adressen außerhalb der eigenen Domain – sofort an den Default-Mailhost weiterleitet, der für das weitere Routing zuständig ist. Erhält jedoch ein Mail Exchanger eine Nachricht mit einer Zieladresse, die er laut MX-Record annehmen oder weiterleiten sollte, muß sein Mailsystem diese

[12] Alias-Namen dürfen prinzipiell nur in CNAME-Records auftreten.

[13] Ein Default-Mailhost oder auch Mail-Relay ist ein besonderer Rechner, an den andere Rechner solche E-Mail schicken können, die sie selbst nicht ausliefern wollen oder können – z.B. weil ihr Mailsystem keine MX Information auswerten kann. Häufig werden Rechner so konfiguriert, daß sie nur lokale Mail selbst ausliefern und nicht-lokale Mail dem Default-Mailhost überlassen.

Adresse auch “erkennen”. Ist dies nicht der Fall, wird die Zieladresse als nicht-lokal interpretiert und die Nachricht an den Default-Mailhost zurückgeschickt. Auf diese Weise können Mailschleifen zwischen Mail Exchanger und Mailhost hervorgerufen werden. Es muß deshalb vor dem Eintrag des MX-Records in das DNS sichergestellt werden, daß das Mailsystem des Mail Exchangers alle in den MX-Records spezifizierten Domainnamen akzeptiert.[14] Für den Zeitraum der Konfiguration und des Tests des Mailsystems eines Mail Exchangers sollte der statische Eintrag eines Default-Mailhosts außer Kraft gesetzt werden. Es ist anzuraten, Mail Exchanger so “intelligent” wie möglich zu machen, daß sie keinen Default-Mailhost benötigen.

[14] Für den SMTP-MTA `sendmail` kann dazu der Test-Modus (Aufruf `sendmail -bt`) herangezogen werden.

Kapitel 6

DNS-Information-Retrieval

Was tun bei "Unwissenheit"? Alle Informationen, welche die Nameserver besitzen, können über interaktiv aufrufbare Resolver abgefragt werden (siehe Kapitel 2.4 auf Seite 9). Die Anwendungen NSLOOKUP und DIG sind solche Resolver. Mit ihnen kann der Nameserver über Namen, Aliasnamen, Adressen, Maschinentypen und Mail-Routing-Information abgefragt werden.

Dieses Kapitel beschreibt nicht den vollständigen Befehlsumfang der Implementierungen, dafür gibt es jeweils eine interaktive Help-Funktion, bzw. zugehörige Manual Pages. Es soll anhand von Beispielen der Umgang mit den Applikationen gezeigt werden und auf einige Besonderheiten eingegangen werden.

6.1 NSLOOKUP

6.1.1 Beschreibung

Folgender Aufruf ist eine schnelle, nicht-interaktive Methode um die IP-Adresse eines Rechners zu erfragen. Ist keine Server-Adresse spezifiziert, dann wird der Default-Nameserver[1] gefragt.

nslookup hostname [server-address oder server-name]

Wird NSLOOKUP ohne Parameter aufgerufen, kommt man in den interaktiven Modus. Nun kann eine Serie von Anfragen gestellt werden.

Ein Nameserver gibt nur über Domains, für die er Primary oder Secondary ist, **"authoritative answers"**. "Non-authoritative answers" werden bei der Antwort entsprechend

[1]Siehe Kapitel 4.2 auf Seite 25 über /etc/resolv.conf.

gekennzeichnet. Möchte man eine ganz sichere Antwort haben, dann muß man den für diese Domain zuständigen Nameserver finden und diesen befragen.

```
> orga.chemie.uni-freiburg.de.
Server:  rusmv8.rus.uni-stuttgart.de
Address:  129.69.1.9

Non-authoritative answer:
Name:    orga.chemie.uni-freiburg.de
Address:  132.230.31.2

> set q=ns
> chemie.uni-freiburg.de.
Server:  rusmv8.rus.uni-stuttgart.de
Address:  129.69.1.9

Non-authoritative answer:
chemie.uni-freiburg.de  nameserver = sun1.ruf.uni-freiburg.de
Authoritative answers can be found from:
sun1.ruf.uni-freiburg.de        inet address = 132.230.1.1
> server sun1.ruf.uni-freiburg.de.
Default Server:  sun1.ruf.uni-freiburg.de
Address:  132.230.1.1

> set q=a
> orga.chemie.uni-freiburg.de.
Server:  sun1.ruf.uni-freiburg.de
Address:  132.230.1.1

Name:    orga.chemie.uni-freiburg.de
Address:  132.230.31.2
```

6.1.2 Kommandos

Mit dem Kommando Help bekommt man eine Liste aller möglichen Kommandos angezeigt. Hier folgen nun ausführlichere Beschreibungen zu einigen der Kommandos:

SET QUERY=X

Die Informationen einer Domain werden beim Nameserver mit den verschiedenen RR-Datensätzen gespeichert (siehe Kapitel 2.1 auf Seite 4). Bevor eine Anfrage an den Nameserver gestellt wird, setzt man den gewünschten Querytyp. Diese Typen entsprechen den Resource Records Bezeichnungen (u.a. A, CNAME, HINFO, NS, MB, MG, MINFO, MR, MX, PTR, SOA, WKS). Zusätzlich existiert ein Querytyp ANY, der sämtliche RR's abfragt.[2] Der Default ist ein Query nach A-Records.

Das Beispiel zeigt zuerst eine Abfrage mit Querytyp ANY, dann eine inverse Namensabfrage (IP-Adresse -> Name). Die IP-Adresse muß meist in dieser besonderen Form angegeben werden (siehe auch Kapitel 2.2 auf Seite 6); die zweite Variante ist nicht bei allen NSLOOKUP-Implementierungen möglich:

```
> set q=any
> rusvx2.rus.uni-stuttgart.de.
Server:  rusmv8.rus.uni-stuttgart.de
Address:  129.69.1.9

rusvx2.rus.uni-stuttgart.de     inet address = 129.69.1.2
rusvx2.rus.uni-stuttgart.de     CPU=VAX8810     OS=VMS
> set q=ptr
> 1.1.69.129.in-addr.arpa.
Server:  rusmv8.rus.uni-stuttgart.de
Address:  129.69.1.9

1.1.69.129.in-addr.arpa host name = rusvx1.rus.uni-stuttgart.de

> 129.69.1.1
Server:  rusmv8.rus.uni-stuttgart.de
Address:  129.69.1.9

1.1.69.129.in-addr.arpa host name = rusvx1.rus.uni-stuttgart.de
```

SET TIMEOUT=X
SET RETRY=X

Wird ein Query mit "*** Request to ... timed-out" beendet, dann sollte man die Anfrage wiederholen. Zusätzlich kann auch das Timeout-Intervall oder der Retrycount erhöht werden. Bringen auch wiederholte Versuche keinen Erfolg, dann funktioniert eventuell auch die IP-Verbindung zum zuständigen Nameserver nicht.

```
> server NS.NASA.GOV.
Default Server:  NS.NASA.GOV
Addresses:  128.102.16.10, 192.52.195.10

> rusvx2.rus.uni-stuttgart.de.
Server:  NS.NASA.GOV
```

[2] Mit query=any bekommt man sämtliche RRs nur dann, falls man einen "authoritavive" NS abfragt. In anderen Fällen wird man nur solche RRs bekommen, die zu dem Zeitpunkt der Abfrage im cache des NS vorhanden waren.

```
Addresses:  128.102.16.10, 192.52.195.10

*** Request to NS.NASA.GOV timed-out
> set timeout=10
> rusvx2.rus.uni-stuttgart.de.
Server:  NS.NASA.GOV
Addresses:  128.102.16.10, 192.52.195.10

Non-authoritative answer:
Name:    rusvx2.rus.uni-stuttgart.de
Address:  129.69.1.2
```

SET [NO]DEBUG

Mit dem Debugging-Mode sieht man ausführlich den Aufbau der Nameserveranfrage. Sie besteht aus 4 Teilen:

- die Frage und einen Returncode
- Resource Records als Antwort
- Authoritative Data
- Zugaben, als Ergänzung zu den RRs oben

Das Beispiel zeigt ein Hostquery mit "non-authoritative answer":

```
> set debug
> asterix.informatik.uni-stuttgart.de.
Server:  noc.belwue.de
Address:  129.143.2.1

res_mkquery(0, asterix.informatik.uni-stuttgart.de, 1, 1)
------------
Got answer:
  HEADER:
      opcode = QUERY, id = 2, rcode = NOERROR
      header flags:  response, want recursion, recursion avail.
      questions = 1,  answers = 1,  auth. records = 4,  additional = 4

  QUESTIONS:
      asterix.informatik.uni-stuttgart.de, type = A, class = IN
  ANSWERS:
  ->  asterix.informatik.uni-stuttgart.de     inet address = 129.69.219.2
  AUTHORITY RECORDS:
  ->  informatik.uni-stuttgart.de  nameserver = ifi.informatik.uni-stuttgart.de
  ->  informatik.uni-stuttgart.de  nameserver = dia.informatik.uni-stuttgart.de
  ->  informatik.uni-stuttgart.de  nameserver = rusmv8.rus.uni-stuttgart.de
  ->  informatik.uni-stuttgart.de  nameserver = unido.informatik.uni-dortmund.de
```

```
  ADDITIONAL RECORDS:
  ->  ifi.informatik.uni-stuttgart.de inet address = 129.69.211.1
  ->  dia.informatik.uni-stuttgart.de inet address = 129.69.211.12
  ->  rusmv8.rus.uni-stuttgart.de     inet address = 129.69.1.9
  ->  unido.informatik.uni-dortmund.de        inet address = 129.217.64.60

------------
Non-authoritative answer:
Name:    asterix.informatik.uni-stuttgart.de
Address:  129.69.219.2
```

SET [NO]DEFNAME

Per Default wird meist der eigene Domainname an den angefragten Namen angehängt. Führt diese Anfrage nicht zum Erfolg wird automatisch eine zweite Anfrage ohne den Domainnamen gestartet. Um diese Wiederholung zu vermeiden, sollten vollständige Domainadressen bei Anfragen mit einem '.' abgeschlossen werden. Ist der Parameter SET NODEFNAM eingestellt, dann wird keine Default-Domain angehängt.

```
> set debug
> rusvx2.rus.uni-stuttgart.de
Server:  rusmv8.rus.uni-stuttgart.de
Address:  129.69.1.9

res_mkquery(0, rusvx2.rus.uni-stuttgart.de.rus.uni-stuttgart.de, 1, 1)
------------
Got answer:
    HEADER:
        opcode = QUERY, id = 2, rcode = NXDOMAIN
        header flags:  response, auth. answer, want recursion, recursion avail.
        questions = 1,  answers = 0,  auth. records = 1,  additional = 0

    QUESTIONS:
        rusvx2.rus.uni-stuttgart.de.rus.uni-stuttgart.de, type = A, class = IN
    AUTHORITY RECORDS:
    ->  rus.uni-stuttgart.de    origin = rusmv8.rus.uni-stuttgart.de
        mail addr = golka.noc.belwue.de
        serial=910322, refresh=6 hours, retry=30 mins
        expire=41 days 16 hours, min=1 day

------------
Failed: NXDOMAIN, num. answers = 0, ns = 1, additional = 0
res_mkquery(0, rusvx2.rus.uni-stuttgart.de.uni-stuttgart.de, 1, 1)
------------
Got answer:
    HEADER:
        opcode = QUERY, id = 3, rcode = NOERROR
        header flags:  response, auth. answer, want recursion, recursion avail.
        questions = 1,  answers = 0,  auth. records = 0,  additional = 0
```

```
    QUESTIONS:
        rusvx2.rus.uni-stuttgart.de.uni-stuttgart.de, type = A, class = IN

------------
res_mkquery(0, rusvx2.rus.uni-stuttgart.de, 1, 1)
------------
Got answer:
    HEADER:
        opcode = QUERY, id = 4, rcode = NOERROR
        header flags:  response, auth. answer, want recursion, recursion avail.
        questions = 1,  answers = 1,  auth. records = 0,  additional = 0

    QUESTIONS:
        rusvx2.rus.uni-stuttgart.de, type = A, class = IN
    ANSWERS:
    ->  rusvx2.rus.uni-stuttgart.de     inet address = 129.69.1.2

------------
Name:    rusvx2.rus.uni-stuttgart.de
Address:  129.69.1.2

>rusvx1.rus.uni-stuttgart.de.
Server:  rusmv8.rus.uni-stuttgart.de
Address:  129.69.1.9

res_mkquery(0, rusvx1.rus.uni-stuttgart.de, 1, 1)
------------
Got answer:
    HEADER:
        opcode = QUERY, id = 5, rcode = NOERROR
        header flags:  response, auth. answer, want recursion, recursion avail.
        questions = 1,  answers = 1,  auth. records = 0,  additional = 0

    QUESTIONS:
        rusvx1.rus.uni-stuttgart.de, type = A, class = IN
    ANSWERS:
    ->  rusvx1.rus.uni-stuttgart.de     inet address = 129.69.1.1

------------
Name:    rusvx1.rus.uni-stuttgart.de
Address:  129.69.1.1
```

SET [NO]RECURSE

Mit diesem Parameter kann der Abfragemodus bestimmt werden. Rekursion bedeutet, daß der gefragte Nameserver solche Fragen, die er nicht beantworten kann, an andere in der Hierarchie höherstehende Nameserver weiterleitet und die Antwort dem fragenden Resolver als Ergebnis zurückgibt.

Bei Iteration (set norec) werden vom Nameserver nur solche Fragen aufgelöst, für die er Informationen als RR-Datensätze oder im Cache besitzt, ansonsten bekommt der Resolver als Antwort eine Liste von Nameservern, die weiter befragt werden können.

```
> set norec
> awesome.berkeley.edu.
Server:  rusmv8.rus.uni-stuttgart.de
Address:  129.69.1.9

Authoritative answers can be found from:
UCBARPA.Berkeley.EDU    inet address = 128.32.130.11
UCBVAX.Berkeley.EDU     inet address = 128.32.133.1
VIOLET.Berkeley.EDU     inet address = 128.32.136.22
> set rec
> awesome.berkeley.edu.
Server:  rusmv8.rus.uni-stuttgart.de
Address:  129.69.1.9

Name:    awesome.berkeley.edu
Address:  128.32.232.10
```

LS DOMAIN [> datei]

Eine Domain, wie z.B. `uni-stuttgart.de`, inklusive aller Subdomains, kann durch eine einzelne Zone repräsentiert werden oder, durch die Definition weiterer SOA-Records, in mehrere Zonen unterteilt sein. Das LS-Kommando liefert nur Informationen über Rechner, die in der abgefragten Zone definiert sind. Existieren weitere Zonen unterhalb dieser Domain, dann werden lediglich diese Zonennamen und die Rechneradressen der dafür zuständigen Nameserver mit ausgegeben, und nicht die Rechner dieser Subdomains.

Es muß immer ein für die Domain zuständiger Nameserver (Primary oder Secondary) abgefragt werden, anderenfalls bekommt man folgende Fehlermeldung:

```
> ls ruf.uni-freiburg.de.
[noc.belwue.de]
Host or domain name            Internet address
*** Error during listing of ruf.uni-freiburg.de.: No information
```

Die Listen sind oft sehr lang, daher gibt es die Möglichkeit den Output auf eine Datei zu schreiben. Per Default werden Rechnernamen und Internetadressen ausgegeben. Für Aliasnamen, HINFO und WKS gibt es die Optionen -a, -h und -s, die Option -d umfaßt alle Informationen.

Das Beispiel zeigt Ausschnitte der Domain `uni-stuttgart.de`. Man sucht Rechner in der Domain `uni-stuttgart.de` und findet zuerst nur weitere Subdomains, wie z.B.

architektur, chemie, rus, ... Um eine Liste aller Rechner der Uni-Stuttgart zu erstellen, muß jede Subdomain explizit abgefragt werden.

Folgendes ist zu beachten:
Der Output dieses Kommandos ist auf den einzelnen Systemen durchaus unterschiedlich. Nur wenige machen es richtig. Die Namen in der linken Spalte sollten den Rechnernamen so anzeigen, daß nur noch die abgefragte Domain oder gar nichts angehängt werden muß. Im Beispiel sollte bei der Liste von uni-stuttgart.de anstatt noc eigentlich noc.belwue.de und anstatt rusmv8 der Name rusmv8.rus angezeigt werden. Setzt man für das LS-Kommando das Debugging, dann bekommt man auf jeden Fall die vollständigen Namen (und als zusätzliche Information noch die "time-to-live"-Zeit).

```
> server noc.belwue.de.
Default Server:  noc.belwue.de
Address:  129.143.2.1

> ls uni-stuttgart.de.
[noc.belwue.de]
Host or domain name           Internet address
 uni-stuttgart                 server = rusmv8.RUS.Uni-Stuttgart.de
 rusmv8                        129.69.1.9
 uni-stuttgart                 server = noc.BelWue.de
 noc                           129.143.2.1
 ...
 chemie                        server = rusmv8.rus.uni-stuttgart.de
 rusmv8                        129.69.1.9
 chemie                        server = noc.belwue.de
 noc                           129.143.2.1
 informatik                    server = ifi.informatik.uni-stuttgart.de
 informatik                    server = dia.informatik.uni-stuttgart.de
 informatik                    server = unido.informatik.uni-dortmund.de
 unido                         129.217.64.60
 rus                           server = rusmv8.rus.uni-stuttgart.de
 rusmv8                        129.69.1.9
 rus                           server = noc.belwue.de
 noc                           129.143.2.1
 ...

> set debug
> ls rus.uni-stuttgart.de.
res_mkquery(0, rus.uni-stuttgart.de., 1, 252)
[rusmv8.rus.uni-stuttgart.de]
Host or domain name           Internet address
 rus.uni-stuttgart.de          server = rusmv8.rus.uni-stuttgart.de    86400
 rusmv8.rus.uni-stuttgart.de   129.69.1.9                      86400
```

```
rus.uni-stuttgart.de          server = rusmv5.rus.uni-stuttgart.de    86400
rusmv5.rus.uni-stuttgart.de    129.69.1.8                     86400
rus.uni-stuttgart.de          server = noc.belwue.de                  86400
noc.belwue.de                 129.143.2.1                    86400
rusx01.rus.uni-stuttgart.de    129.69.1.131                   86400
ruspc3101.rus.uni-stuttgart.de 129.69.31.2                    86400
ruspool01.rus.uni-stuttgart.de 129.69.21.21                   86400
ruspool04.rus.uni-stuttgart.de 129.69.21.24                   86400
nfhpc1.rus.uni-stuttgart.de    129.69.1.106                   86400
...
```

6.1.3 Weitere Beispiele

1. Welche Root-Nameserver kennt mein Nameserver?

   Je nachdem, ob man Rekursion oder nicht eingestellt hat, erhält man die Root-Nameserver, die der gefragte Nameserver in Erfahrung bringen kann, oder diejenigen, die ihm durch Caching oder feste Einträge in seinen Dateien bekannt sind.

```
> set norec
> set q=ns
> .
Server:  rusmv8.rus.uni-stuttgart.de
Address:  129.69.1.9

Non-authoritative answer:
(root)  nameserver = NS.NIC.DDN.MIL
(root)  nameserver = AOS.BRL.MIL
(root)  nameserver = KAVA.NISC.SRI.COM
(root)  nameserver = C.NYSER.NET
(root)  nameserver = TERP.UMD.EDU
(root)  nameserver = NS.NASA.GOV
(root)  nameserver = NIC.NORDU.NET
Authoritative answers can be found from:
NS.NIC.DDN.MIL  inet address = 192.112.36.4
AOS.BRL.MIL     inet address = 192.5.25.82
KAVA.NISC.SRI.COM       inet address = 192.33.33.24
C.NYSER.NET     inet address = 192.33.4.12
TERP.UMD.EDU    inet address = 128.8.10.90
NS.NASA.GOV     inet address = 128.102.16.10
NS.NASA.GOV     inet address = 192.52.195.10
NIC.NORDU.NET   inet address = 192.36.148.17
```

2. Welche sind die aktuellen Nameserver für DE?

```
> set q=ns
> de.
Server:  rusmv8.rus.uni-stuttgart.de
Address:  129.69.1.9

Non-authoritative answer:
de      nameserver = DEINS.INFORMATIK.UNI-DORTMUND.DE
de      nameserver = NS.UU.NET
de      nameserver = MCSUN.EU.NET
de      nameserver = SUN01A.DESY.DE
de      nameserver = ADM.BRL.MIL
de      nameserver = SUNIC.SUNET.SE
de      nameserver = IRAUN1.IRA.UKA.DE
de      nameserver = DNS-WEST.NERSC.GOV
de      nameserver = DFNNOC.GMD.DE
Authoritative answers can be found from:
DEINS.INFORMATIK.UNI-DORTMUND.DE        inet address = 192.35.64.34
NS.UU.NET       inet address = 137.39.1.3
MCSUN.EU.NET    inet address = 192.16.202.1
SUN01A.DESY.DE  inet address = 131.169.200.2
```

Nun kann die Liste nochmals überprüft werden, indem der Primary Nameserver für DE direkt gefragt wird:

```
> server DEINS.INFORMATIK.UNI-DORTMUND.DE
Default Server:  DEINS.INFORMATIK.UNI-DORTMUND.DE
Address:  192.35.64.34

> de.
Server:  DEINS.INFORMATIK.UNI-DORTMUND.DE
Address:  192.35.64.34

de      nameserver = deins.Informatik.Uni-Dortmund.DE
de      nameserver = Sun01a.DESY.DE
de      nameserver = ns.UU.net
de      nameserver = ADM.BRL.mil
de      nameserver = mcsun.EU.net
de      nameserver = sunic.sunet.SE
de      nameserver = iraun1.IRA.UKA.DE
de      nameserver = dfnnoc.GMD.DE
de      nameserver = DNS-WEST.NERSC.GOV
deins.Informatik.Uni-Dortmund.DE        inet address = 192.35.64.34
Sun01a.DESY.DE  inet address = 131.169.200.2
ns.UU.net       inet address = 137.39.1.3
ADM.BRL.mil     inet address = 192.5.25.4
```

```
ADM.BRL.mil     inet address = 192.5.21.30
mcsun.EU.net    inet address = 192.16.202.1
sunic.sunet.SE  inet address = 192.36.125.2
sunic.sunet.SE  inet address = 130.237.216.2
iraun1.IRA.UKA.DE       inet address = 129.13.10.90
dfnnoc.GMD.DE   inet address = 192.88.108.8
DNS-WEST.NERSC.GOV      inet address = 128.55.128.191
```

3. Kennt mein Nameserver einen Weg (MX-Record) für eine bestimmte Mailadresse?

   Man hat eine Mailadresse bekommen, die keinen Rechnernamen enthält, z.B. die Adresse dillmann@rus.uni-stuttgart.dbp.de. Jetzt möchte man wissen, ob der Nameserver eine Route kennt, um diese Mail auszuliefern. Das Beispiel zeigt, daß sämtliche Mail zu `rus.uni-stuttgart.dbp.de` als erstes zum Rechner `noc.belwue.de` transferiert wird.

```
> set q=mx
> rus.uni-stuttgart.dbp.de
Server:  sun1.ruf.uni-freiburg.de
Address:  132.230.1.1

rus.uni-stuttgart.dbp.de        preference = 50, mail exchanger = noc.belwue.de
noc.belwue.de   inet address = 129.143.2.1
```

6.2 DIG: Domain Information Groper

6.2.1 Beschreibung

Dieses Programm ist nicht standardmäßig auf den Rechnern verfügbar, sondern muß extra installiert werden. DIG ist als Public Domain Software auf folgenden Rechnern via anonymous ftp verfügbar:

Name	IP-Adresse	Verzeichnis
verena.isi.edu	128.9.0.32	PUB
rusmv1.rus.uni-stuttgart.de	129.69.1.12	kommunikation/nameserver

Ist DIG installiert, dann findet man eine ausführliche Beschreibung in den Manual-Pages. DIG wird meistens verwendet in der Form:

dig @<server> <domain> <query-type> <query-class>

<server>	optional: Nameserver, an den Query gestellt wird
<domain-name>	Domain, über welche Information gewünscht wird

<**query-type**>	Querytyp entsprechend den Resource-Records, (A (Standard), MX, NS, SOA, ANY, HINFO, AXFR)

Querytyp ANY fragt nach allen vorhandenen Resource-Records. Bei Querytyp AXFR entspricht der Output dem NSLOOKUP-Kommando LS -d, allerdings immer mit vollständigen Domainnamen (siehe Kapitel 6.1.2 auf Seite 41).

6.2.2 Beispiele

Beispiel 1: Mapping Domain-Adresse zu IP-Adresse.

```
dig kssun1.rus.uni-stuttgart.de

; <<>> DiG 2.0 <<>> kssun1.rus.uni-stuttgart.de
;; ->>HEADER<<- opcode: QUERY , status: NOERROR, id: 6
;; flags: qr aa rd ra ; Ques: 1, Ans: 1, Auth: 0, Addit: 0
;; QUESTIONS:
;;      kssun1.rus.uni-stuttgart.de, type = A, class = IN

;; ANSWERS:
kssun1.rus.uni-stuttgart.de.    86400   A       129.69.1.62

;; Sent 1 pkts, answer found in time: 0 msec
;; FROM: noc to SERVER: default -- 129.143.2.1
;; WHEN: Wed Apr  3 16:54:38 1991
;; MSG SIZE  sent: 45  rcvd: 61
```

Beispiel 2: Erstellen einer Informationsliste über die Domain `bates.edu`, wobei der Nameserver `mother.bates.edu` abgefragt wird.

```
dig bates.edu @mother.bates.edu axfr

; <<>> DiG 2.0 <<>> bates.edu @mother.bates.edu axfr
;; QUESTIONS:
;;      bates.edu, type = AXFR, class = IN

bates.edu.      3600    SOA     mother.bates.edu.  rob.mother.bates.edu. (
                        910322  ;serial
                        3600    ;refresh
                        300     ;retry
```

```
                        3600000 ;expire
                        3600 )  ;minim

bates.edu.      3600    NS      mother.bates.edu.
mother.bates.edu.       3600    A       134.181.1.3
bates.edu.      3600    NS      nic.near.net.
nic.near.net.   171487  A       192.52.71.4
bates.edu.      3600    MX      0 mother.bates.edu.
orlith.bates.edu.       3600    A       134.181.1.12
orlith.bates.edu.       3600    HINFO   Prime-EXL7330 UNIX
orlith.bates.edu.       3600    MX      10 mother.bates.edu.
mother.bates.edu.       3600    A       134.181.1.3
mother.bates.edu.       3600    HINFO   MIPS-120/5 UNIX
mother.bates.edu.       3600    MX      0 mother.bates.edu.
lab36.bates.edu.        3600    A       134.181.2.2
lab36.bates.edu.        3600    HINFO   AT&T-6286 DOS
exl7330.bates.edu.      3600    CNAME   orlith.bates.edu.
bigmac.bates.edu.       3600    A       134.181.1.13
bigmac.bates.edu.       3600    HINFO   MAC-IIfx A/UX
bat.bates.edu.  3600    A       134.181.1.1
bat.bates.edu.  3600    HINFO   PRIME-9750 PRIMOS
bat.bates.edu.  3600    MX      0 mother.bates.edu.
;  Matching SOA found
;; FROM: noc to SERVER: mother.bates.edu  134.181.1.3
;; WHEN: Wed Apr  3 17:05:47 1991
```

6.3 Weitere Resolver

Auf IBM-Rechnern unter dem Betriebssystem VM/CMS gibt es außer NSLOOKUP noch die Resolver QNS und NSQUERY. Alle Resolver können nur als Command-Line-Kommando mit Parametern und nicht im interaktiven Mode aufgerufen werden. Sie sind Public-Domain-Software und daher nicht immer auf den Rechnern vorhanden.

Kapitel 7

Domain Management

Im folgenden sind einige Punkte zusammengefaßt, die ein Domainadministrator bzw. Betreiber eines Nameservers beachten sollte.[1]

7.1 Zuverlässigkeit des Nameservers

Der Nameserver-Dienst muß zuverlässig und ununterbrochen vorhanden sein. Dies bedeutet in der Praxis:

1. Der Nameserver muß auf Rechnersystemen betrieben werden, die selten gebootet oder gar abgeschaltet werden müssen. Sind Stillstandzeiten unvermeidbar, sollten diese so kurz wie möglich gehalten werden und nur dann riskiert werden, wenn der Secondary Nameserver funktionsfähig ist.

2. Nameserver für dieselbe Domain sollten möglichst unabhängig voneinander betrieben werden. Zu vermeiden sind: Gemeinsame Stromversorgung, Aufstellung im gleichen Gebäude, gemeinsames Netz etc., also Situationen, in denen die Ursache für den Ausfall des einen Master-Nameservers (z.B. Stromausfall, Brand, Wasserschaden, Defekt eines Routers etc.) auch den anderen in Mitleidenschaft ziehen kann. Wünschenswert wäre, wenn Nameserver für dieselbe Domain an unterschiedliche, geographisch entfernte Netze angeschlossen sind.

7.2 Aktualität der Nameserverdaten

1. Die Nameserver-Daten müssen immer richtig und aktuell sein.
   Der Datenbestand eines Nameservers muß den aktuellen Stand der in der Domain

[1] /RFC 1032/ und /RFC 1033/ sind noch heute gute und aktuelle Ratgeber für Domain Administratoren.

betriebenen Objekte korrekt und vollständig wiederspiegeln. Es darf nicht vorkommen, daß der Rechner eines Netzteilnehmers erst Wochen nach der Zuteilung der IP-Adresse im Nameserver aufgenommen wird. Bei Änderung der Daten des Primary Nameservers muß der Secondary NS nach Ablauf des "Refresh"-Timers einen Transfer der Zonen-Daten durchführen. Dies sollte von den Administratoren des Primary NS **und** des Secondary NS gleichermaßen überprüft werden.

2. Wer große Domains verwaltet, soll an Möglichkeiten denken, das Erzeugen der Zonen-Dateien zu automatisieren.
   Das Verwalten großer Datenmengen in einfachen Textdateien, wie sie von Nameservern gelesen werden, ist sehr fehlerträchtig. Hier bietet sich die Datenhaltung mit Hilfe einer relationalen Datenbank an, die Werkzeuge zur Vermeidung von Redundanz und zur Gewährleistung der Konsistenz bereitstellt. Die Nameserver-Dateien selbst können dann aus den Daten der Datenbank generiert werden.

7.3 Konfiguration des Nameservers

Bei der Konfiguration des Nameservers muß folgendes beachtet werden (siehe dazu Kapitel 4):

1. Alle Nameserver für die Domain müssen in der Zonen-Datei angegeben werden.
   Bei der Delegierung von Subdomains müßen sowohl Primary NS als auch alle Secondary NS durch NS-Records spezifiziert werden. An dieser Stelle sind evt. auch A-Records als "glue records" nötig (siehe Kapitel 4.1.6 auf Seite 22). Alle existierenden Nameserver für eine Zone müssen auch in der Zonen-Datei des Primary NS mit NS-Records aufgeführt werden.

2. NS-Records für Nameserver außerhalb der eigenen oder einer delegierten Domain sollten vermieden werden.
   Obwohl häufig praktiziert, birgt der Eintrag externer Nameserver in der eigenen Zonen-Datei die Gefahr, daß eine solche Spezifikation ungültig werden kann, ohne daß dies rechtzeitig bemerkt wird. Da der Administrator für diese Domain von Ihrem Eintrag in der Regel nichts weiß, kann er Sie über eingetretene Änderungen (z.B. Wechsel des Rechnernamens) nicht unterrichten. Sollten NS-Records dieser Art unvermeidlich sein, empfiehlt sich eine regelmäßige Überprüfung dieser Einträge.

3. Die Timer im SOA-Record sollen angemessene Werte haben, insbesondere der "Expire"-Timer sollte einen Wert aufweisen, daß beim Ausfall des Primary NS der Secondary NS ausreichend lange für die Zone authoritativ bleibt.

4. Jede Änderung der Zonen-Daten muß durch Erhöhung der Serialnummer des SOA-Records abgeschlossen werden.

Nach dem Einlesen der aktualisierten Zonen-Daten sollte die Erhöhung der Serialnummer durch eine Nameserverabfrage überprüft werden. Dies gilt auch – nach Ablauf des "Refresh"-Timers – für alle Secondary NS der betroffenen Zone.

7.4 Betriebliche Aspekte

1. Der Zuständige für den Secondary Service muß informiert werden, falls neue Subdomains mit eigenem SOA-Record eingerichtet werden.
   Der Backup-Service eines Secondary NS gilt nur für die Zonen, die in dessen Konfigurationsdatei eingetragen sind. Wird dieser Service auch für die delegierte Subdomain gewünscht, muß dieser Eintrag explizit vorgenommen werden.

2. Falls der forwarder Mechanismus benutzt werden soll: Informieren Sie vorher den betroffenen Systemadministrator.

3. Der Name/IP-Adresse eines Nameservers soll nicht ohne triftigen Grund geändert werden.
   Bei unvermeidlichem Wechsel der Hardware sollte der neue Rechner möglichst den Namen und die IP-Adresse übernehmen. Falls man trotzdem die IP-Adresse bzw. den Namen ändern muß, sollte der Verwalter der Parent Domain unbedingt informiert werden, ebenso wie die Verwalter der Secondary Nameserver. Bitte beachten Sie auch, daß Sie auch den Verwalter der IN-ADDR.ARPA-Domain informieren müssen. Dringen Sie darauf, daß bei allen betroffenen Nameservern ein völliger Neustart des Nameserver-Prozesses durchgeführt wird, damit nicht im Cache die alten Daten "weiterleben". Das Verschwinden der alten Daten beim Namen-/Adressenwechsel kann beschleunigt werden, wenn vor der Umstellung die Timer-Zeiten im SOA-Record des Primary NS vorübergehend stark verkürzt werden.

7.5 Tools

Viele der – oft eintönigen – Aufgaben der Nameserver-Administration können mit Hilfe entsprechender Software-Tools automatisiert bzw. unterstützt werden. An dieser Stelle werden drei dieser Werkzeuge aufgeführt und kurz beschrieben. Wir sind überzeugt, daß viele Systemverwalter ähnliche eigenentwickelte Programme im Betrieb haben. Die hier getroffene Auswahl stellt keine Bewertung dar, sie soll nur exemplarische Programme für drei verschiedene Aufgaben der Nameserver-Administration vorstellen: die Überwachung des Nameserverprozesses, die Überprüfung der Konsistenz der Zonendaten von Nameservern untereinander und die Überprüfung des Zonen-Transfers zwischen Primary und Secondary Nameserver. Alle aufgeführten Tools sind nur für UNIX Rechner verwendbar.

7.5.1 ninit – named init program

ninit überwacht den lokalen Nameserverprozess in der Weise, daß es einen eventuellen Exodus oder ein "Aufhängen" sofort registriert und daraufhin den Nameserver neu startet. Darüberhinaus unternimmt ninit alle 60 Sekunden eine Namensauflösung über den lokalen Nameserver. Falls innerhalb einer kurzen Zeit keine Antwort erfolgt, wird der laufende Nameserverprozess durch ninit beendet und ein neuer Prozess initiiert. ninit setzt BIND 4.8.3 voraus. Quelle: Anonymous FTP Server `bitsy.mit.edu` (File `tytso/ninit.tar`).

7.5.2 doc – diagnose unhealthy DNS domains

doc ist ein sehr mächtiges Instrument zur Überprüfung der Konfiguration der relevanten Nameserver für eine bestimmte Domain. Bei der Prüfung führt doc im wesentlichen folgende Schritte durch:

1. Überprüfung aller authoritativen Nameserver der Parent Domain: Prüfung aller Parent Nameserver auf gleiche SOA Serien-Nummer sowie auf gleiche NS-Einträge fuer die zu prüfende Domain. Damit soll sicher gestellt werden, dass die Domain korrekt über die Parent Domain eingebunden ist.

2. Überprüfung aller Nameserver für die zu prüfende Domain: Prüfung aller Nameserver auf gleiche SOA Serien-Nummer und gleiche NS-Eintraege.

3. Vergleich der NS-Eintraege der Parent Domain und der zu prüfenden Domain.

4. Prüfung einer Adresse auf PTR Resource Records (meist der erste Nameserver).

Bei Fehlermeldungen sollten die erzeugten Log-Files durchgeschaut werden und die anderen Nameserver (weitere Secondary und alle Parent NS) kontaktiert werden, um die entsprechenden Einträge abzugleichen. doc benützt intern das interaktive Resolver-Frontend dig-2.0. Quelle: Anonymous FTP Server `venera.isi.edu` (File `pub/dig.2.0.tar.Z`).

7.5.3 zrmon – zone refresh monitor program

zrmon überwacht das Zusammenspiel zwischen Secondary und Primary Nameserver einer Domain aus der Sicht des Secondary Nameservers. Es überprüft, ob der Zonen-Refresh innerhalb der eingestellten Zeitspanne stattfindet und meldet bei fehlschlagendem Refresh die verbleibende Zeit bis zur Expiration der Zone. Erfolgreiche Zonen-Transfers werden gemeldet mit Angaben zur Änderung der Zonendaten (absolute und relative Differenz der Datenmenge). zrmon eignet sich vorwiegend für Rechner, die einen umfangreichen Secondary Service anbieten (mehr als 20 Zonen) und bei denen die manuelle Überprüfung der Zonendaten fast unmöglich ist. Quelle: Anonymous FTP Server `ftp.belwue.de` (File `pub/zrmon.tar`).

Anhang A

Beispiele für BIND-Konfigurationsdateien

A.1 named.boot

```
;  Sample primary boot file for bind server.
;
; type          domain                    source file or host      backup file
;
directory       twg$tcp:[netdist.etc]
; Root-Nameserver:
cache           .                         named.ca
; Secondary-Nameserver:
secondary       211.69.129.in-addr.arpa   129.69.211.1  named.rev_ifi211
secondary       212.69.129.in-addr.arpa   129.69.211.1  named.rev_ifi212
secondary       belwue.de                 129.143.2.1   named.hosts_bw
secondary       143.129.in-addr.arpa      129.143.2.1   named.rev_bw
; Primary-Nameserver:
primary         uni-stuttgart.de          named.hosts
primary         rus.uni-stuttgart.de      rus.hosts
primary         physik.uni-stuttgart.de   phys.hosts
primary         uni-stuttgart.dbp.de      dbp.mx
primary         69.129.in-addr.arpa       named.rev
primary         0.0.127.in-addr.arpa      named.local
primary         localhost                 named.loopback
```

A.2 named.ca

```
; Die neuesten Root-Nameserver erhaelt man per ftp von nic.ddn.mil.
; Initial cache data for root domain servers.
;
.                       999999  IN      NS      NS.NIC.DDN.MIL.
                        999999  IN      NS      KAVA.NISC.SRI.COM.
                        999999  IN      NS      AOS.BRL.MIL.
                        999999  IN      NS      C.NYSER.NET.
                        999999  IN      NS      TERP.UMD.EDU.
                        999999  IN      NS      NS.NASA.GOV.
                        999999  IN      NS      NIC.NORDU.NET.

;
;  Prep the cache (hotwire the addresses).  Order does not matter
;

NS.NIC.DDN.MIL.         999999  IN      A           192.112.36.4
KAVA.NISC.SRI.COM.      999999  IN      A           192.33.33.24
AOS.BRL.MIL.            999999  IN      A           192.5.25.82
C.NYSER.NET.            999999  IN      A           192.33.4.12
TERP.UMD.EDU.           999999  IN      A           128.8.10.90
NS.NASA.GOV.            999999  IN      A           128.102.16.10
NS.NASA.GOV.            999999  IN      A           192.52.195.10
NIC.NORDU.NET.          999999  IN      A           192.36.148.17
```

A.3 named.local

```
;       Address to hostname mappings for net 127.0.0
;
$ORIGIN 0.0.127.in-addr.arpa.
@     IN     SOA  rusmv8.rus.uni-stuttgart.de. golka.noc.belwue.de. (
                      19910101 ; Serial
                      3600     ; Refresh
                      300      ; Retry
                      604800   ; Expire  (7 Tage)
                      172800 ) ; Minimum (2 Tage)
      IN     NS   rusmv8.rus.uni-stuttgart.de.
1     IN     PTR  localhost.
```

A.4 named.loopback

```
;       Hostname to address mapping for loopback interface.
;
$ORIGIN localhost.
@     IN     SOA   rusmv8.rus.uni-stuttgart.de. golka.noc.belwue.de. (
                       199107010 ; Serial
                       21600     ; Refresh
                       1800      ; Retry
                       604800    ; Expire  (7 Tage)
                       172800 )  ; Minimum (2 Tage)
      IN     NS    rusmv8.rus.uni-stuttgart.de.
      IN     A     127.0.0.1
```

A.5 named.hosts

```
;       Hostname to address mappings.  Host information.
;
uni-stuttgart.de. IN SOA rusmv8.rus.uni-stuttgart.de. golka.noc.belwue.de. (
                       199103130 ; Serial
                       21600     ; Refresh
                       1800      ; Retry
                       604800    ; Expire  (7 Tage)
                       172800 )  ; Minimum (2 Tage)
; Nameserver fuer uni-stuttgart.de.:
            IN   NS  rusmv8.rus.uni-stuttgart.de.
            IN   NS  rusmv5.rus.uni-stuttgart.de.
            IN   NS  noc.belwue.de.
; Nameserver fuer delegierte Zonen:
informatik.uni-stuttgart.de.     IN  NS rusmv8.rus.uni-stuttgart.de.
                                 IN  NS ifi.informatik.uni-stuttgart.de.
physik.uni-stuttgart.de.         IN  NS rusmv8.rus.uni-stuttgart.de.
                                 IN  NS rusmv5.rus.uni-stuttgart.de.
                                 IN  NS noc.belwue.de.
rus.uni-stuttgart.de.            IN  NS rusmv8.rus.uni-stuttgart.de.
                                 IN  NS rusmv5.rus.uni-stuttgart.de.
                                 IN  NS noc.belwue.de.
;
;
; Glue-Records (fuer noc.belwue.de nicht notwendig, da in fremder Domain):
ifi.informatik.uni-stuttgart.de.       IN  A  129.69.211.1
rusmv8.rus.uni-stuttgart.de.           IN  A  129.69.1.9
rusmv5.rus.uni-stuttgart.de.           IN  A  129.69.1.8
```

A.6 phys.hosts

```
;       Hostname to address mappings.  Host information.
;
physik.uni-stuttgart.de. IN SOA rusmv8.rus.uni-stuttgart.de. golka.noc.belwue.de. (
                      910313    ; SERIAL
                      21600     ; REFRESH
                      1800      ; RETRY
                      604800    ; Expire  (7 Tage)
                      172800 )  ; Minimum (2 Tage)
; Nameserver fuer physik.uni-stuttgart.de.:
            IN   NS  rusmv5.rus.uni-stuttgart.de.
            IN   NS  rusmv8.rus.uni-stuttgart.de.
            IN   NS  noc.belwue.de.
; keine Glue-Records notwendig, da alle Nameserver in fremder Domain.
;
physik.uni-stuttgart.de.           IN  MX 10  ph3hp840.physik.uni-stuttgart.de.
physik.uni-stuttgart.de.           IN  MX 50  noc.belwue.de.
;
localhost.physik.uni-stuttgart.de. IN  A      127.0.0.1
;
sirmel.physik.uni-stuttgart.de.    IN  A      129.69.45.1
                                   IN  HINFO  MVAX-II/GPX      VMS
iftvx1.physik.uni-stuttgart.de.    IN  A      129.69.45.2
                                   IN  HINFO  VAX8300          VMS
ph3hp840.physik.uni-stuttgart.de.  IN  A      129.69.46.2
                                   IN  HINFO  HP840            UNIX
a900.physik.uni-stuttgart.de.      IN  A      129.69.46.3
                                   IN  HINFO  HP1000           RTE-A
oscar.physik.uni-stuttgart.de.     IN  A      129.69.74.11
                                   IN  HINFO  VAX3500          VMS
romeo.physik.uni-stuttgart.de.     IN  A      129.69.74.12
                                   IN  HINFO  MVAX-2000        VMS
malefiz.physik.uni-stuttgart.de.   IN  A      129.69.221.139
                                   IN  HINFO  PC-AT/386        MS-DOS
; ...
```

A.7 named.rev

```
;       Address to hostname mappings.
;
69.129.in-addr.arpa. IN  SOA rusmv8.rus.uni-stuttgart.de. golka.noc.belwue.de. (
                                910313   ; Serial
                                21600    ; Refresh
                                3600     ; Retry
                                604800   ; Expire  (7 Tage)
                                172800 ) ; Minimum (2 Tage)
; Nameserver fuer 69.129.in-addr.arpa.:
                            IN      NS    rusmv8.rus.uni-stuttgart.de.
                            IN      NS    rusmv5.rus.uni-stuttgart.de.
                            IN      NS    noc.belwue.de.
; Delegierte Zonen und deren Nameserver:
133.69.129.in-addr.arpa.    IN      NS    apatix.iao.fhg.de.
50.69.129.in-addr.arpa.     IN      NS    sun-1.intes-stuttgart.de.
                            IN      NS    rusmv8.rus.uni-stuttgart.de.
113.69.129.in-addr.arpa.    IN      NS    sun-1.intes-stuttgart.de.
                            IN      NS    rusmv8.rus.uni-stuttgart.de.
211.69.129.in-addr.arpa.    IN      NS    ifi.informatik.uni-stuttgart.de.
                            IN      NS    dia.informatik.uni-stuttgart.de.
                            IN      NS    rusmv8.rus.uni-stuttgart.de.
212.69.129.in-addr.arpa.    IN      NS    ifi.informatik.uni-stuttgart.de.
                            IN      NS    dia.informatik.uni-stuttgart.de.
                            IN      NS    rusmv8.rus.uni-stuttgart.de.
;
;
1.45.69.129.in-addr.arpa.      IN   PTR  sirmel.physik.uni-stuttgart.de.
2.45.69.129.in-addr.arpa.      IN   PTR  iftvx1.physik.uni-stuttgart.de.
2.46.69.129.in-addr.arpa.      IN   PTR  ph3hp840.physik.uni-stuttgart.de.
3.46.69.129.in-addr.arpa.      IN   PTR  a900.physik.uni-stuttgart.de.
11.74.69.129.in-addr.arpa.     IN   PTR  oscar.physik.uni-stuttgart.de.
12.74.69.129.in-addr.arpa.     IN   PTR  romeo.physik.uni-stuttgart.de.
139.221.69.129.in-addr.arpa.   IN   PTR  malefiz.physik.uni-stuttgart.de.
1.1.69.129.in-addr.arpa.       IN   PTR  rusvx1.rus.uni-stuttgart.de.
2.1.69.129.in-addr.arpa.       IN   PTR  rusvx2.rus.uni-stuttgart.de.
; ...
```

Anhang B

Literaturliste

/BEER91/ : P. Beertema, EUnet, Mail an ripe-org Mailliste, März 1991.

/HEB90/ : M. Hebgen, Univ. Heidelberg, Mail an Belwü-Nameserver Mailliste, Feb. 1990.

/HEB91/ : M. Hebgen, Univ. Heidelberg, Private Mitteilung, Jun. 1991.

/RFC 920/ : J. Postel, J. Reynolds, “Domain Requirements”, USC/Information Sciences Institute, Oct. 1984.

/RFC 974/ : C. Partridge, “Mail Routing and the Domain System”, CSNET CIC BBN Laboratories Inc., Jan. 1986.

/RFC 987/ : S. E. Kille, “Mapping between X.400 and RFC 822”, University College London, Jun. 1986.

/RFC 1032/ : M. Stahl, “Domain Administrator's Guide”, SRI International, Nov. 1987.

/RFC 1033/ : M. Lottor, “Domain Administrator's Operations Guide”, SRI International, Nov. 1987.

/RFC 1034/ : P. Mockapetris, “Domain names – Concepts and Facilities”, USC/Information Sciences Institute, Nov. 1987.

/RFC 1035/ : P. Mockapetris, “Domain names – Implementation and Specification”, USC/Information Sciences Institute, Nov. 1987.

/RFC 1081/ : M. Rose, “Post Office Protocol – Version 3”, Network Working Group, The Wollongong Group, Nov. 1988.

/RFC 1101/ : P. Mockapetris, “DNS Encoding of Network Names and Other Types”, USC/Information Sciences Institute, Apr. 1989.

/RFC 1123/ : R. Branden (Editor), "Requirements for Internet Hosts – Application and Support", Internet Engineering Task Force, Oct. 1989.

/RFC 1174/ : Network Working Group, "IAB Recommended Policy on Distibuting Internet Identifier Assignment and IAB Recommended Policy Change to Internet 'Connected' Status", USC/Information Sciences Institute, Aug. 1990.

/SCH91/ : D. J. Schmidt, Rechenzentum, Univ. Braunschweig, Private Mitteilung, Mai 1991.

/GI-Köln90/ : C. Kalle, A. Clauberg, S. Faßbender: Teilnehmerunterlagen zum Tutorium "Praxis von Directory Services", GI, Deutsche Informatik Akademie, Juni 1990

Index