Technische Universität München

Institut für Informatik

Diplomarbeit

Erstellung einer Managementschnittstelle für DHCP-Server

Aufgabensteller: Professor H.- G. Hegering
Betreuer: Stephen Heilbronner
Bearbeiter: Gernot Riegert

19. August 1996

Ich versichere, daß ich diese Diplomarbeit selbständig verfaßt und nur die angegebenen Quellen und Hilfsmittel verwendet habe.

München, den 15.August 1996

Inhaltsverzeichnis

Kapitel 1

Einleitung

In den letzten Jahren wurden rapide technische Fortschritte in der Entwicklung tragbarer Computersysteme, bei Funk– und Satellitennetzen und bei der weltweiten Vernetzung von Rechnern erzielt. Damit wurde die Voraussetzung für das sogenannte *Mobile Computing* geschaffen. Heute ist es fast schon üblich, daß Anwender ihre persönlichen Computersysteme wie Notebooks stets bei sich tragen. Vielerorts besteht aber noch die Schwierigkeit, diese Systeme auch problemlos in Netze zu integrieren (sprich zu konfigurieren), die nicht zum heimatlichen Netz gehören, wenn man die Ressourcen des fremden Netzes nutzen will. Dies wirft viele neue Fragen für das Netz- und Systemmanagement auf, für die größtenteils noch keine angemessenen Lösungen existieren.
Die gängigen Konfigurationsprotokolle für TCP/IP-Systeme (RARP, BOOTP) haben den Nachteil, daß sie sich nur bedingt bzw. überhaupt nicht für den Einsatz mit Mobilsystemen eignen. Als Abhilfe für diese Problemsituation wurde eine Erweiterung des BOOTP-Protokolls entwickelt, welches diese Probleme löst.
Das *Dynamic Host Configuration Protocol* (DHCP) wird heute schon in vielen Bereichen zur Konfiguration von Computersystemen eingesetzt, die an ein Rechnernetz angeschlossen werden, zum Beispiel auch unter Windows NT. Dabei ist noch kein überzeugendes Konzept entwickelt worden, eine mit DHCP konfigurierte Umgebung einem integrierten Netz- und Systemmanagement zugänglich zu machen. Daten über die Konfiguration von IP–Systemen werden in allen zur Zeit vorhandenen DHCP-Implementierungen lokal von den sogenannten *DHCP-Servern* gehalten; es gibt zur Zeit keine Möglichkeit,

zentral die Konfiguration von verschiedenen DHCP-Umgebungen zu steuern bzw. zu überwachen.

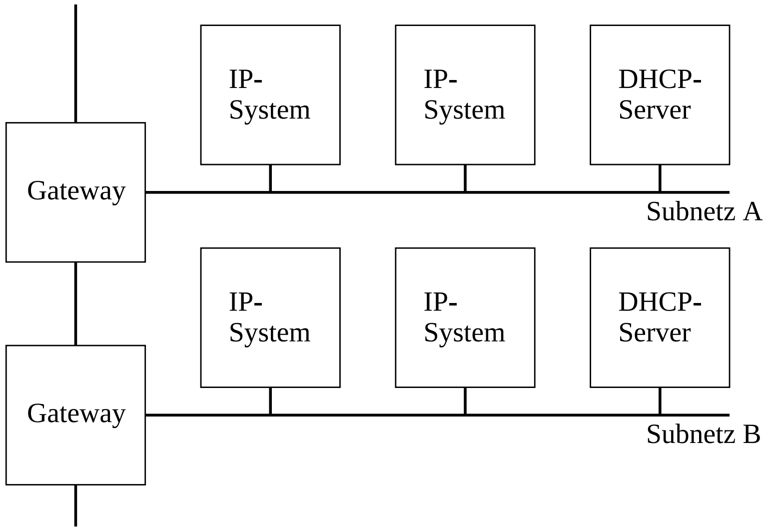

Abbildung 1.1: Ausschnit eines mit DHCP konfigurierten Netzes

Hier setzt diese Diplomarbeit an: Durch die Integration einer SNMP-Schnittstelle wird angestrebt, DHCP-Umgebungen für ein Netz- und Systemmanagement zugänglich zu machen. Um zu dieser Integration zu kommen, wird das DHCP-Protokoll zunächst aufgrund seiner Definition vorgestellt (Kapitel 2), um eine Grundlage für das Verständnis seines Vorgehens zu erhalten. Weiter werden Szenarien untersucht (Kapitel 3), unter denen DHCP eingesetzt werden kann. Aus den Szenarien werden dann die Anforderungen gefolgert, die an ein DHCP-Management gestellt werden (Kapitel 5). Als Konsequenz aus den Anforderungen wird darauf ein Modell für das DHCP-Management entwickelt, welches sich dann in einer *Management Information Base* (MIB) niederschlägt.

Kapitel 2

Das DHCP-Protokoll

2.1 Einleitung

Das Internet spielt bereits heute eine wichtige Rolle in der Industrie und in der Wissenschaft. Firmen setzen Computer für ihre Zwecke ein und vernetzen sie. Grundlage für diese Vernetzung ist eine Protokollvereinbarung namens TCP/IP (*Transmission Control Protocol / Internet Protocol*), welche bei der Mehrzahl der heutigen Datenkommunikationsnetze als Grundlage für die Datenübertragung dient.
Um eine möglichst effiziente Nutzung solcher Rechner zu ermöglichen, streben die Netzbetreiber es häufig an, daß Ressourcen wie Drucker oder Datenspeichergeräte über das Netz aus genutzt werden können. Ebenso sollen Computer, die von Benutzern bedient werden sollen, bestmöglich ausgelastet werden. Für einen Rechner in einem Netz ergibt sich unter anderem die Problematik des Neustarts oder Bootens. Um die verschiedenen Parameter zu steuern, die ein Rechner für den Bootvorgang benötigt, können verschiedene vordefinierte Bootmechanismen verwendet werden.

2.2 Motivation für Bootmechanismen

Bei Computersystemen, die ohne Bootdiskette oder ähnliches booten, existiert für gewöhnlich im ROM oder vergleichbaren Speichermedien ein sogenanntes Startup-Programm. Zur Kostenminimierung und aus Kompatibilitätsgründen versieht jeder Computerhersteller seine Produkte mit jeweils dem gleichen Startup-Programm. Da Rechner mit unterschiedlichen IP-Adressen den selben Bootvorgang durchlaufen, kann der Code des Startups nicht die IP-Adresse des Systems enthalten. Das bedeutet, daß diese Systeme von einem anderen Rechner ihre IP-Adresse erhalten müssen. Außerdem benötigen diese Systeme zusätzliche Informationen wie einen File Server, bei dem sie Daten holen und speichern können, oder den nächstgelegenen IP-Router [Com95].

2.2.1 RARP

Die erste und einfachste Alternative zur Lösung dieses Problems, das *Reverse Adress Resolution-* oder *RARP*-Protokoll, bietet Computersystemen die Möglichkeit, sich eine IP-Adresse zuteilen zu lassen, wenn ein Computer über keine Möglichkeiten verfügt, seine IP-Adresse lokal zu speichern, z.B. auf einer Festplatte. Im einzelnen wendet sich dabei ein Rechner mittels eines Broadcasts auf sein Subnetz an einen sogenannten *RARP-Server* und teilt ihm seine Hardwareadresse mit, für die der Server die zugeordnete IP-Adresse kennt. Der Server teilt daraufhin dem Rechner seine IP-Adresse mit. Die Übertragung der Informationen erfolgt über die physikalische Netzverbindung. Bei der Verwendung von Ethernet beispielweise hat eine RARP-Anfrage den Standard-Vorspann, Ethernet-Quell- und Ziel-Adressen und Pakettyp-Felder am Beginn des Ethernet-Pakets. Der Datenteil enthält dann die 28-Oktett-RARP-Nachricht [Com95].

RARP hat drei schwerwiegende Nachteile: Als erstes setzt es den direkten Zugriff auf die Netz-Hardware voraus. Diese Tatsache macht es einem Anwendungsprogrammierer schwer bzw. unmöglich, ein Serverprogramm zu erstellen. Zweitens benötigt RARP den direkten Austausch von Paketen zwischen einer Clientmaschine und dem Rechner, der die Rolle des Servers übernimmt. Der Server kann aber dem Client als einzige Information die 4-Oktett-IP-Adresse mitteilen. Und drittens verhindert die Tatsache, daß RARP die Hardware-Adresse eines Rechners zu dessen Identifizierung benutzt, den Einsatz von RARP in Netzen, die Hardwareadressen dynamisch vergeben [FMMT84],[Com95].

2.2.2 BOOTP

Als Ausweg aus dieser Problemlage unter RARP wurde das *Bootstrap-* oder *BOOTP-Protokoll* entwickelt. Wie RARP setzt BOOTP ein Client-Server-artiges Konzept voraus und benötigt nur ein einzelnes Paket, um die Konfigurationsinformation zu übermitteln. Dabei erfährt ein konfigurationswilliger Host von einem BOOTP-Server seine IP-Adresse, die Adresse eines Routers und eines File Servers sowie firmenspezifische Parameter ([Wim93]), die der Host für das Booten benötigt [CG85], [Com95].

Dies funktioniert folgendermaßen: Ein konfigurationswilliges Computersystem sendet einen sogenannten *BOOTREQUEST* als Broadcast an einen oder mehrere *BOOTP-Server*. Als Transportprotokoll wird das *User Datagram Protocol* verwendet. Der BOOTP-Server antwortet mit einem *BOOTREPLY*, in dem er dem Client eine IP-Adresse und andere Parameter für die Netzkonfiguration vermittelt, die es aus einer Konfigurationsdatei ausliest. Diese Datei wird von den Netzbetreibern erstellt. Falls BOOTP-Server und BOOTP-Client sich nicht auf dem selben Subnetz befinden, leitet ein sogenannter *Relay Agent* den Broadcast auf die gewünschten Subnetze weiter.

2.2.3 DHCP

Mit der Entwicklung von mobilen Systemen wie Systemen mit drahtloser Übertragung oder Laptops mit Netzanschluß ergibt sich das Problem, daß BOOTP zu statisch für die Konfiguration dieser Systeme ist. Die Ursache liegt darin, daß BOOTP nur gestattet, Konfigurationsinformationen einem Hostsystem statisch zuzuordnen. Außerdem muß bei BOOTP der Netzadministrator für jeden Host die Konfigurationsinformationen einzeln in einem Konfigurations-File des BOOTP-Servers niederlegen. Dadurch ist BOOTP nicht in der Lage, einzelne Rechner dynamisch mit einer Konfiguration zu versorgen. Große Probleme bekommt man mit BOOTP vor allem dann, wenn Rechner sehr flexibel in unterschiedlichen Netzen eingesetzt werden oder wenn weniger freie IP-Adressen vorhanden sind als Interessenten für eine IP-Adresse [Com95].
Einen Ausweg aus dieser Problemlage bietet das sogenannte *Dynamic Host Configuration Protocol*, kurz *DHCP* genannt. DHCP wird heute schon in vielen Bereichen zur Konfiguration von IP-Systemen eingesetzt und eignet sich

vor allem auch gut zur Konfiguration mobiler Systeme [Com95].
DHCP erweitert das BOOTP-Protokoll, indem es einen Rechner oder IP-System die Möglichkeit bietet, seine IP-Adresse schnell und dynamisch zu erhalten. Um einen sogenannten *DHCP-Server* in die Lage zu versetzen, IP-Systeme dynamisch mit IP-Adressen zu versorgen, muß ein Netzadministrator ihm einen Satz IP-Adressen zur Verfügung stellen. Über diese IP-Adressen muß der DHCP-Server frei verfügen können, d.h., sie dürfen nicht von anderen Systemen im Netz belegt sein.
Wenn nun ein Rechner, der sogenannte *DHCP-Client*, in das Netz eintritt, nimmt er mit dem DHCP-Server Kontakt auf [Dro93], [Dro96b]. Der Server wählt eine seiner freien Adressen aus und vermittelt sie dem neuen Rechner. Dabei unterstützt DHCP auch BOOTP-Anfragen. Auf die Architektur von DHCP wird unter 2.3 näher eingegangen.
Bei DHCP gibt es dabei drei Arten der Adreßvergabe [Dro93], [Dro96b]: Bei der *manuellen* Vergabe von IP-Adressen werden diese wie beim *BOOTP-Protokoll* von einem Administrator vergeben. Beim *automatischen* Vergabeverfahren bekommt ein neu hinzugekommener Rechner vom Server automatisch eine permanente IP-Adresse zugeteilt. Beim *dynamischen* Vergabeverfahren bekommt der neu Hinzugekommene wie beim automatischen Verfahren eine IP-Adresse, jedoch nur für eine beschränkte Zeit, die sogenannte *Leasezeit*. Dabei wird der Begriff *Lease* auch als Bezeichnung für die Erlaubnis verwendet, die Netzressourcen zu nutzen, welche der DHCP-Server zugänglich macht. Bei der Vergabe von Adressen ist zu beachten, daß ein System, welches mehrere IP-Adressen benötigt, für jede Adresse eine eigene Adressenanfrage unter DHCP starten muß.
Das Verhalten des DHCP-Clients läßt sich aus dem in Abbildung 2.1 dargestellten Ablaufdiagramm ersehen.

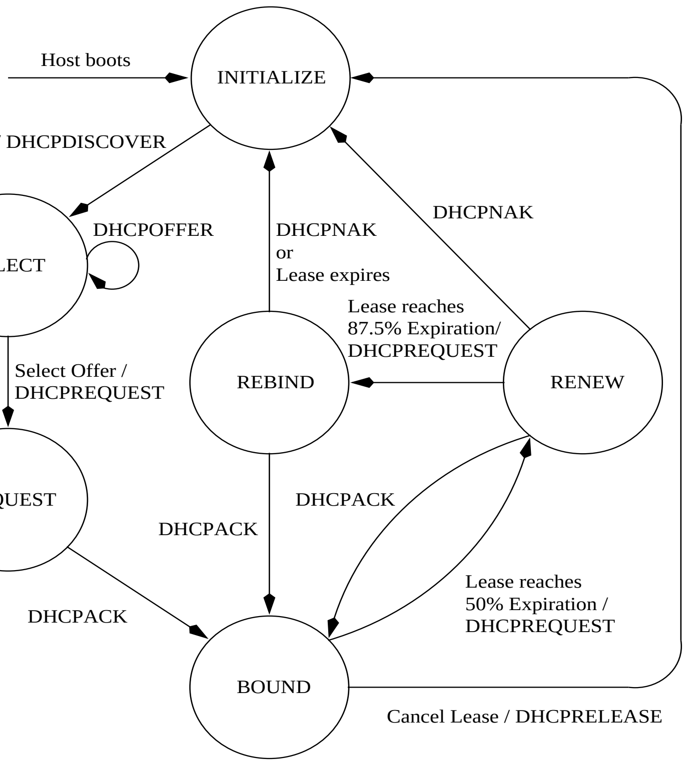

Abbildung 2.1: Zustandsübergangsdiagramm eines DHCP-Clients

(Nach [Come95])

2.3 Beschreibung von DHCP

Angelehnt an das Abstraktionsmodell für Managementanwendungen, die man dabei in Modelle der Organisation, Information, Funktion und der zugrundeliegenden Kommunikation unterteilt, wird hier versucht, eine anschauliche Beschreibung des DHCP-Protokolls zu geben. Als Grundlage wird die aktuelle Definition des DHCP-Protokolls für IP Version 4 verwendet [Dro93], [Dro96b].

2.3.1 Organisationsmodell

DHCP ist Client-Server-artig organisiert. Auf der einen Seite steht der *DHCP-Server*, der die Konfigurationsinformationen für ein oder mehrere Subnetze bereitstellt. Auf der anderen Seite steht der *DHCP-Client*. Dieser ist ein Hostsystem in einem Subnetz, das sich vom DHCP-Server die Informationen für seine Konfiguration sowie eine IP-Adresse geben läßt. Sollten DHCP-Server und DHCP-Client sich auf unterschiedlichen Subnetzen befinden, sorgt ein sogenannter *Relay Agent* für die Weiterleitung der Broadcast-Meldungen zwischen Server und Client [Dro93],[Dro96b],[Com95].

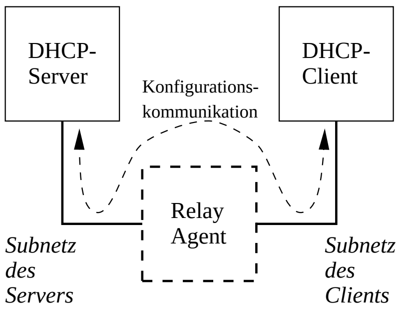

Abbildung 2.2: Kommunikation DHCP-Server - DHCP-Client

2.3.2 Informationsmodell

Die Informationen für die Konfiguration sowie die IP-Adresse werden dem *DHCP-Client* mithilfe der sogenannten *DHCP Options* mitgeteilt. Diese werden an Protokolldateneinheiten/PDUs angehängt, die zwischen DHCP-Server und DHCP-Client ausgetauscht werden. Die Länge des Option-Teils einer DHCP-PDU ist dabei variabel mit einer Mindestlänge von 312 Oktetts.
Welche Konfigurationsinformationen bzw. IP-Adressen zur Verfügung stehen, entnimmt ein DHCP-Server einer Datei mit Konfigurationsinformationen, auf die er lesend zugreift. Es wird weiterhin im Rahmen des Protokolls davon ausgegangen, daß sowohl beim Server als auch beim Client eine konsistente Datenhaltung existiert, d. h. , sowohl der Server als auch der Client merken sich die aktuelle Konfiguration des Clients. Dies geschieht beim Server sowohl innerhalb einer Datenstruktur zur Laufzeit wie auch innerhalb

einer Datenbasis, die als Schutz vor Ausfällen des Serverprogramms in einem File niedergelegt wird. Das genaue Format dieser Datenbasis und der Datenstruktur wird aber vom Protokoll nicht vorgeschrieben; das Format hängt von der Implementierung innerhalb der DHCP-Server-/Clientsoftware ab.

2.3.3 Funktionsmodell

Die Funktionen von DHCP werden durch PDUs erbracht, die unter DHCP Namen tragen, welche ihre Funktion kennzeichen. Bei DHCP gibt es zwei Arten, welche sich durch ihre Verwendung entweder durch den DHCP-Server oder durch den DHCP-Client unterscheiden [Dro93],[Dro96b].

PDUs des DHCP-Clients

Der Client benutzt folgende PDUs:

DHCP-PDU	Bedeutung
DHCPDISCOVER	Anfrage an beliebigen DHCP-Server wg. Konfiguration
DHCPREQUEST	Anfrage an spezifischen Server wg. Konfiguration
DHCPDECLINE	Ablehnungsmeldung, weil IP-Adresse schon benutzt wird
DHCPINFORM	Anfrage nach Konfiguration; externe IP-Adresse vorhanden
DHCPRELEASE	Freigabe von Ressourcen

PDUs des DHCP-Servers

Vom Server werden folgende PDUs benutzt:

DHCP-PDU	Bedeutung
DHCPOFFER	Angebot IP-Adresse und Konfiguration
DHCPACK	Bestätigung zugeteilter IP-Adresse und Konfiguration
DHCPNAK	Nichtbestätigung von IP-Adresse und Konfiguration

Der Aufbau einer PDU ist dabei folgender [Dro93]:

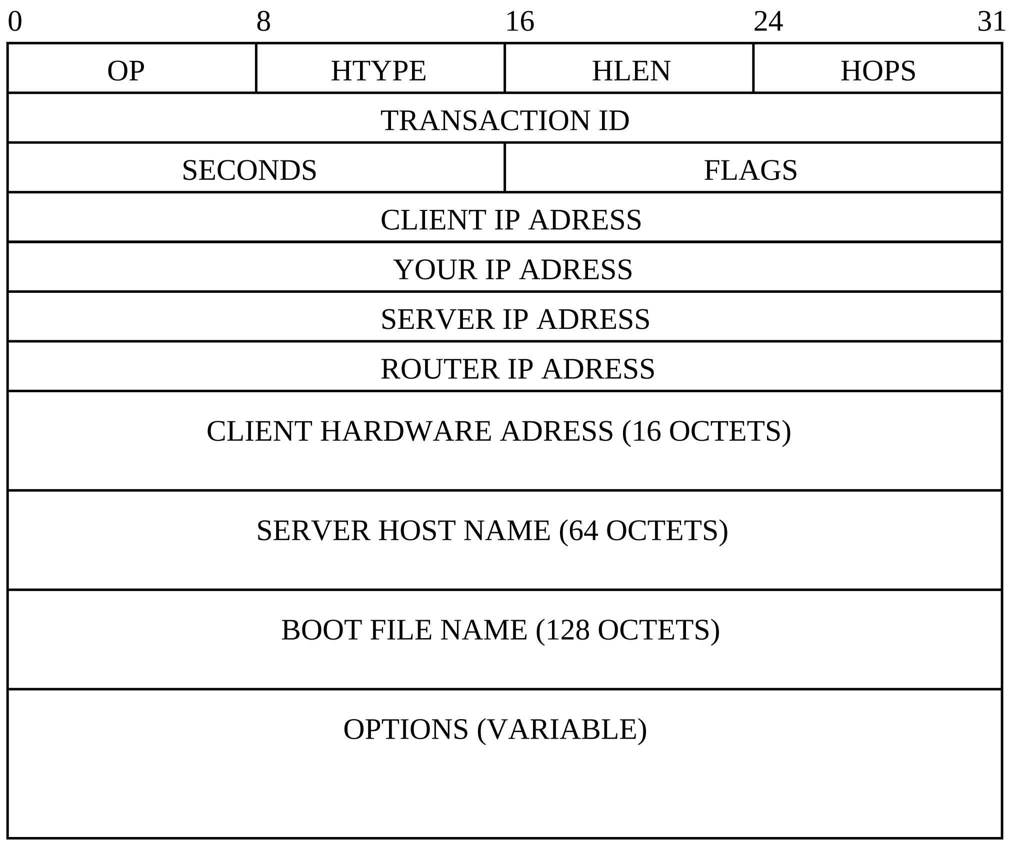

Abbildung 2.3: Aufbau einer DHCP-Nachricht

Die einzelnen Felder bedeuten dabei [Dro93]:

Felder einer DHCP-PDU

OP	OP-Code der Nachricht: 1 = BOOTREQUEST 2 = BOOTREPLY
HTYPE	Art der Hardwareadresse, z. B. 1 = 10mb Ethernet
HLEN	Länge der Hardwareadresse
HOPS	Vom Client auf 0 gesetzt, wird vom Relay Agent benutzt, falls über Relay Agents gebootet wird
TRANSACTION ID	Identifikator der Verbindung zwischen Client und Server
SECONDS	Zeitspanne seit Beginn der Konfiguration
FLAGS	Bit 0 dient als Broadcast-Flag; Bit 1-15 sind Null
CLIENT IP ADRESS	IP-Adresse des Clients; nichtleer falls Client im BOUND, RENEW oder REBINDING Zustand ist
YOUR IP ADRESS	angebotene IP-Adresse
SERVER IP ADRESS	Adresse des Servers, der beim Bootstrap benutzt werden soll; nur bei DHCPOFFER, DHCPACK
ROUTER IP ADRESS	Adresse des Relay Agents; nur beim Booten mit Relay Agent benutzt
CLIENT HARDWARE ADRESS	Hardware-Adresse des Clients
SERVER HOST NAME	Optionaler Server Host Name
BOOT FILE NAME	Name des Bootfiles; in DHCPDISCOVER generischer Name oder Null, in DHCPOFFER mit genauer Directoryangabe
OPTIONS	Konfigurationsparameter

2.3.4 Kommunikationsmodell

Die Kommunikation des DHCP-Protokolls benutzt das *User Datagram Protocol/UDP* als Transportprotokoll [Dro93], [Dro96b], [Com95]. Das bringt mit sich, daß durch die Nichtüberwachung des PDU-Transports eventuell Pakete verlorengehen können, die zwischen DHCP-Server und DHCP-Client ausgetauscht werden. Durch Wiederholung einer Übertragung wird aber von Seiten von DHCP sichergestellt, daß keine Lücken in der Informationsübertragung stattfinden. Diese Wiederholung geschieht mit Hilfe von fest definierten Algorithmen, welche nach dem sogenannten *Radomized Exponential Backoff*-Verfahren arbeiten [Dro93], [Dro96b]. Die Kommunikation zwischen Server und Client geschieht unter Benutzung der Ports 67 und 68. Der Server benutzt dabei Port 67, der Client Port 68, um PDUs wie DHCPDISCOVER, DHCPREQUEST etc. zu verschicken.

2.4 Sicherheitsmechanismen unter DHCP

Beim Einsatz von Computern stellen sich oft auch Probleme wie die eines nichtauthorisierten Zugriffs auf die bereitgestellten Ressourcen durch Benutzer. Aus dieser Problemlage heraus hat man sich schon seit längerem Verfahren überlegt, wie Rechner vor ungewolltem Zugriff zu schützen sind. Jeder Benutzer eines UNIX-Systems beispielsweise kennt wohl die Login-Prozedur, bei der man ein nur dem Benutzer bekanntes Paßwort eingeben muß, bevor man die bereitgestellten Dienste des Systems nutzen kann. Gerade beim Einsatz von mobilen Systemen wie Laptops, die über eine Infrarot- oder eine andere drahtlose Verbindung an ein Netz angschlossen sind, kann die Notwendigkeit einer Authentifizierung von zugangsberechtigten Benutzern entstehen.

Mittlerweile existieren schon zahlreiche Verfahren, um die Sicherheit im Umgang mit Computern zu gewährleisten [KPS96]. Bei vielen Verfahren werden übertragene Daten mit kryptologischen Verfahren verschlüsselt, wobei Codes als Schlüssel verwendet werden, die nur dem Benutzer bzw. dem Netzbetreiber bekannt sind. Bei den Verschlüsselungsverfahren kennt man *Sender* und *Empfänger* von kodierten Nachrichten. Bemühungen um die Sicherheit in IP-Netzen werden von Workgroups wie der IP Security Workgroup vorgenommen [Wor].
Welches oder welche Verfahren zur Sicherung eines Systems oder Netzes verwendet werden, hängt zum größten Teil von der sogenannten *Policy* der Betreiber ab. Die Policy wiederum, also die Strategie, mit der die Sicherheit erreicht werden soll, hängt teils von den lokalen Gegebenheiten, teils von den individuellen Ansprüchen der Netzbetreiber ab.
Für DHCP gibt es zur Zeit die für die Lösung von Sicherheitsfragen nur einen theoretischen Vorschlag [Dro96a]. Im Rahmen dieser Arbeit interessiert vor allem, wie sich ein solcher Sicherheitsansatz in einer Managementumgebung für DHCP verwirklichen läßt. Dabei soll aber nicht auf alle Verfahren eingegangen werden, die zur Zeit theoretisch umsetzbar wären, vielmehr soll anhand eines Beispiels erklärt werden, wie man grundsätzlich an dieses Problem herangehen kann.
Abhängig von der Sicherheits-Policy der Netzbetreiber kann es gewünscht werden, DHCP-Meldungen durch eine Authentifizierung überprüfbar auf ihre Herkunft und Berechtigung zu machen. Es ist vorstellbar, daß falsche DHCP-

Server (s.2.3) versuchen, DHCP-Clients zu konfigurieren oder daß falsch konfigurierte DHCP-Server versuchen, DHCP-Clients ebenso falsch zu konfigurieren.
Zum Ermöglichen des Authentifizierens von DHCP-Meldungen wird deshalb ein *Message Authentification Code* (MAC) eingeführt, der in jeder PDU enthalten sein soll.
Bei der Authentifizierung von DHCP-Nachrichten ergibt sich allerdings folgendes Problem: DHCP-Clients benutzen den sogenannten *limited broadcast* für die Kommunikation mit den DHCP-Servern. Deshalb setzen die Relay Agents für die Kommunikation mit DHCP-Servern außerhalb des lokalen Netzes in die weitervermittelten DHCP-Nachrichten ihre Adresse in das giadress-Feld (Relay Agentenadresse) ein, damit der DHCP-Server später seine Antwort an den Client zurückschicken kann. Außerdem machen die Relay Agents einen Eintrag in das hops-Feld der Nachricht. Das bedeutet, wenn der Client eine DHCP-Meldung mit einer sogenannten *Message Digest*-Funktion verschlüsselt, können die Informationen giadress und hops nicht verschlüsselt werden. Diese Tatsache wird bei den folgenden Verschlüsselungsmechanismen berücksichtigt.

2.4.1 Authentifizierung für DHCP-Meldungen

Die Authentifizierung von DHCP-Nachrichten funktioniert folgenderweise [Dro96a]:
Es werden ein Sender S und ein Empfänger R definiert, die einen geheimen Schlüssel K vereinbart haben, der exklusiv nur für den Nachrichtenaustausch zwischen S und R dient. S und R sind dabei ein DHCP-Server und ein DHCP-Client. S bildet einen MAC (Message Authentification Code), indem es einen Zähler mit K verschlüsselt. Dieser Zähler sollte monoton wachsen und groß genug sein, um einen *Network Time Protocol*-Zeitwert nach RFC 1305 [Mil92] enthalten zu können. Der MAC sieht also folgendermaßen aus:

$$\text{Zähler},\ f(K, MD(\text{Nachricht} + \text{Zähler}))$$

MD ist eine Message Digest-Funktion, die noch festgelegt werden muß.

Der MAC ist Bestandteil der DHCP Nachricht, die von S generiert wird. Die Kodierungsfunktion f muß derart gestaltet sein, daß der Schlüssel K sich nicht aus dem MAC ableiten lässt und daß $f(K, MD(\text{Nachricht} + \text{Zähler}))$

sich nicht erraten lässt.
Wenn R die Nachricht empfängt, kann er den Schlüssel K aus $f(K, MD(\text{Nachricht+Zähler}))$ ableiten um zu verifizieren, daß der Sender der Nachricht K kennt. Wenn als Zähler die aktuelle Tageszeit verwendet wird, kann man die Gefahr von Nachrichtenwiederholungsangriffen vermeiden.

2.4.2 Schlüsselmanagement

[Dro96a]: Es wird vorausgesetzt, daß der Schlüssel K dem Client über einen abhörsicheren Mechanismus mitgeteilt wird. Der Client speichert K lokal; der Server verwaltet die K's für alle authorisierten Clients.
Damit man nicht zentral eine Liste von per Zufall erzeugten Schlüsseln verwalten muß, wird der Schlüssel für jeden Client unter Zuhilfenahme eines clientspezifischen Wertes erzeugt. Die Erzeugungsfunktion für K hat folgendes Aussehen:

$$K = f(MK, \text{Client-spezifischer Wert})$$

Dabei ist MK ein geheimer Master Key/Schlüssel und f eine Kodierungsfunktion, wie sie schon im vorhergehenden Abschnitt beschrieben wurde.
Jeder DHCP-Client muß eine exklusive Client ID besitzen, mit der seine Interaktion mit dem DHCP-Server identifiziert werden kann. Die Client ID kann eine MAC-Adresse sein oder eine Herstellernummer; ihre Implementierung hängig von den Ansprüchen des Netzadministrators ab. Damit erfüllt die Client ID die Forderung des clientspezifischen Wertes, der in der Schlüsselgenerierungsformel benötigt wird.

Aus unseren Vorgaben folgt, daß ein nicht authorisierter Client keinen K-Schlüssel erzeugen kann, weil er den Master Key MK nicht kennt. Weiterhin kann der DHCP-Server eine neue Nachricht auf ihre Berechtigung überprüfen, indem er K aus der Client ID generiert. Für ihm bekannte Clients kann der DHCP-Server entweder die K's immer dynamisch aus der Client ID in den DHCP-Nachrichten ziehen oder alle K's vorverarbeiten und in einer Art Cache speichern.

2.4.3 Kodierungsfunktion

Als Kodierungsfunktion wird TBD (s.[KPS96]) verwendet. Eine Möglichkeit ist, eine Anleihe bei DNSSEC (s.[KPS96]) zu machen, bei dem die Kodierungsfunktion in der Nachricht durch einen Identifikator ersehbar wird. Dabei kann dieser Identifikator entweder für keine Kodierung, eine Default Kodierung nach dem Public Key Cryptographic Standard von DNSSEC oder andere Kodierungen stehen [Dro96a].

2.4.4 Verifikation des Inhalts einer Nachricht

[Dro96a] Als weitverbreitetete Mechanismus für die Generierung von Digests, der die Unversehrtheit von Nachrichten bestätigen kann, wird MD5 (s.[KPS96]) verwendet. Die einzige Modifikation für DHCP besteht darin, daß man aus den am Anfang beschriebenen Gründen giaddr und hops als Nullen in die MD5-Formel einsetzt. Wenn man all dies in Betracht zieht, erstellt der Sender S folgendes:

- die DHCP Nachricht
- den Zähler
- $f(K, MD5(\text{Nachricht} - (\text{'giaddr' und 'hops'}) + \text{Zähler}))$

wobei K der Schlüssel des Clients ist. Der Empfänger R kann dann den Inhalt der Nachricht anhand des MD5-Wertes überprüfen und nachvollziehen, ob S den Schlüssel K kannte, indem er diesen aus dem Wert von $f(K, MD5(\text{Nachricht} - (\text{'giaddr' und 'hops'}) + \text{Zähler}))$ ableitet.

2.4.5 Zusammenfassung

Das hier dargestellte Schema ist sowohl auf DHCPv4 als auch auf DHCPv6 anwendbar. Sollte es in die DHCPv4- oder v6-Spezifikation (s.a. 4) mitaufgenommen werden, so wird die Information für die Authentifikation von DHCP-Nachrichten als DHCP Option festgelegt werden [Dro96a].

2.5 Zusammenfassung

In diesem Kapitel wurde das DHCP-Protokoll vorgestellt. Dabei wurde zunächst auf die Struktur von DHCP-Nachrichten eingegangen, bevor DHCP in Anlehnung an das Abstraktionsmodell aus dem Netz- und Systemmanagementbereich genauer beschrieben wurde. Ferner wurde ein Ansatz aufgezeigt, wie sich die Authentifizierung von DHCP-Nachrichten gestalten könnte. Diese Informationen sollen als Basis für die folgenden Kapitel dienen.

Kapitel 3

Szenarien für den Einsatz von DHCP

3.1 Einleitung

Nachdem das DHCP-Protokoll im zweiten Kapitel vorgestellt worden ist, kann man nun genauer die Szenarien untersuchen, unter denen DHCP eingesetzt wird bzw. eingesetzt werden kann. Aus diesen Szenarien soll dann im folgenden Kapitel die Anforderungen für ein DHCP-Managment entwickelt werden.
Auf der Hand liegt, wie schon unter 2.2.3 angedeutet, der Einsatz von DHCP mit mobilen Systemen mit oder ohne Mobile IP. Ebenso wichtig ist der Einsatz mit stationären Systemen, der heute in der Praxis schon eine große Rolle spielt. Weiter wird auf die Verwendung von DNS unter DHCP eingegangen. Um später die Managementanforderungen, die aus dem Protokollverhalten von DHCP für IPv4 und IPv6 entstehen, entwickeln zu können, werden beide Protokollverhalten dargestellt, wobei der Einsatz von DHCPv6 wegen der Änderungen, die sich momentan immer noch ergeben, von den anderen getrennt in einem eigenen Kapitel untersucht wird. Ein interessanter Aspekt der Anwendung von DHCP ist schließlich das Renumbering von Subnetzen.

3.2 Konfiguration mobiler Systeme auf dem TCP/IP-Netz

Tragbare Computersysteme liefern bereits heute oft die effizienteste Möglichkeit, eine Arbeitsumgebung an einen anderen Ort zu transferieren. Typische Anwendungsfälle sind z. B. die folgenden:

- Mitarbeiter eines Unternehmens wechseln bei der Bearbeitung bestimmter Projekte übergangsweise in andere Abteilungen bzw. an andere Standorte

- Besucher und Gäste eines Unternehmens möchten z. B. während Arbeitstreffen oder Konferenzen Teile der lokalen Infrastruktur (Drucker, E-Mail, Internet-Zugang etc.) mitbenutzen

- Mitarbeiter von Partnerunternehmen benötigen z. B. für Vorführungen, SW-Installation oder SW-Wartung Zugang zum lokalen Netz für ihr mitgebrachtes System.

Das Dynamic Host Configuration Protocol ist in besondererer Weise für die Konfiguration mobiler Systeme geeignet. Unter mobilen Systemen versteht man zum Beispiel Laptops mit Netzzugang, Computer, die drahtlos mit dem Internet verbunden sind, etc. .

Um ein mobiles System in einem Subnetz erfolgreich benutzen zu können, müssen dem mobilen System einige Parameter zugeteilt werden. Dies sind zum Beispiel die IP-Adresse, die die Identifikation des Systems unter TCP/IP ermöglicht, Domain Name Server, welche es mit DNS-Namen versorgen, etc. . Voraussetzung für den Einsatz von mobilen Systemen unter DHCP ist die Installation eines DHCP-Client-Programms. Dieses nimmt dann Kontakt mit einem DHCP-Server auf (falls vorhanden) und läßt sich die Daten für die Netzkonfiguration geben. Zu diesen Daten gehört die IP-Adresse des Host-Systems, der nächste Router etc. . Es wird vorausgesetzt, daß das sogenannte *dynamische* Adreßvergabeverfahren (s. Kapitel 2) eingesetzt wird, da die anderen Vergabeverfahren permanente IP-Adressen vergeben bzw. den Einsatz eines Netzadministrators voraussetzen. Sollte ein Host sich dem Ende der Leasezeit nähern und seinen Lease nicht verlängern können, muß er sich um einen neuen Lease bemühen. Wenn ein Host vor dem Ende der Leasezeit sich

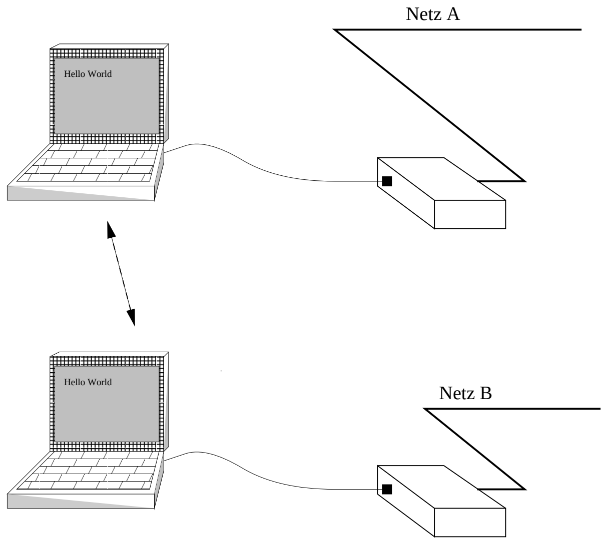

Abbildung 3.1: Der Laptop als mobiles System

wieder aus dem Netz verabschieden will, sollte die DHCP-Client-Software ein DHCPRELEASE (s. 2.3.3) absetzen, um die IP-Adresse für einen neuen Benutzer freizugeben. Im Falle eines Absturzes des Host-Systems bleibt die zugeteilte IP-Adresse auf jeden Fall die gesamte Leasedauer dem Host erhalten. Um dieser Situation zu begegnen, sollte der Host sich die Konfiguration "merken". Wie dies passiert, hängt von der Implementierung der Client-Software ab.
Die hier geschilderten Vorgänge sind unter dem Abschnitt 3.4 bzw. beim Einsatz von DHCPv6 in Kapitel 4 näher beschrieben.

3.2.1 DHCP und Mobile-IP

Im Rahmen des Einsatzes mobiler Systeme mit DHCP muß man auch den Einsatz von *Mobile-IP* berücksichtigen. Dabei wird jedem mobilen Knoten in einem Netz (z. B. ein Laptop mit Netzanschluß) eine sogenannte *Home Adress* zugeordnet, unter der der Knoten im Netz erreichbar ist. Wenn sich der mobile Knoten aber in einem anderen Netz als seinem Heimatnetz befinden sollte, müssen die für ihn bestimmten Datagramme so geroutet werden, daß sie auch beim Host (unter seiner Home Adress) ankommen. Zu die-

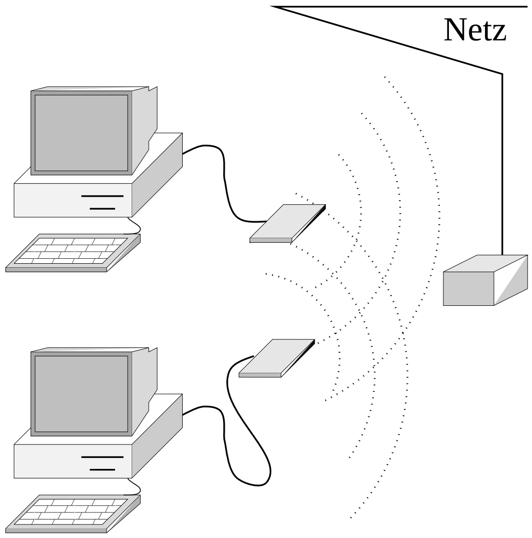

Abbildung 3.2: Drahtlose Verbindung

sem Zweck gibt es unter Mobile-IP die sogenannte *Care-of-Adresse*, an die während der Abwesenheit des Knotens die Datagramme geschickt werden, die eigentlich für die Home Adress gedacht waren. Der sogenannte Home Agent kümmert sich um diese Versendung von Datagrammen [Per96b]. Mehr Informationen zu Mobile-IP erhält man auch aus aktuellen Arbeiten zu diesem Thema [MS96],[Lan96],[Unt].
Unter DHCP wird Mobile-IP durch die DHCP Option Mobile IP Home Agent (Nr.68) unterstützt, unter der der DHCP-Server dem mobilen DHCP-Client mehrere Adressen von Home Agents mitteilen kann [Ale93], [AD96].

3.3 Konfiguration auf dem TCP/IP-Netz fest installierter Systeme

Auch bei fest installierten Systemen wie Unix Workstations oder Windows NT-Rechnern wird DHCP zunehmend eingesetzt, um IP-Systeme über das Netz zu konfigurieren. Bei Windows 3.11/95/NT ist DHCP sogar im Lieferumfang enthalten. In der Praxis wird DHCP vorwiegend in diesem Bereich eingesetzt. Die meisten Serversoftwarevarianten unterstützen standardmäßig nur die *manuelle* oder *automatische* Vergabe von IP-Adressen (s. Kapitel 2), bei denen Adressen permanent vergeben werden. Das heißt, der Einsatz von DHCP zur Konfiguration mobiler Systeme ist noch nicht so verbreitet wie der Einsatz bei fest installierten Systemen.

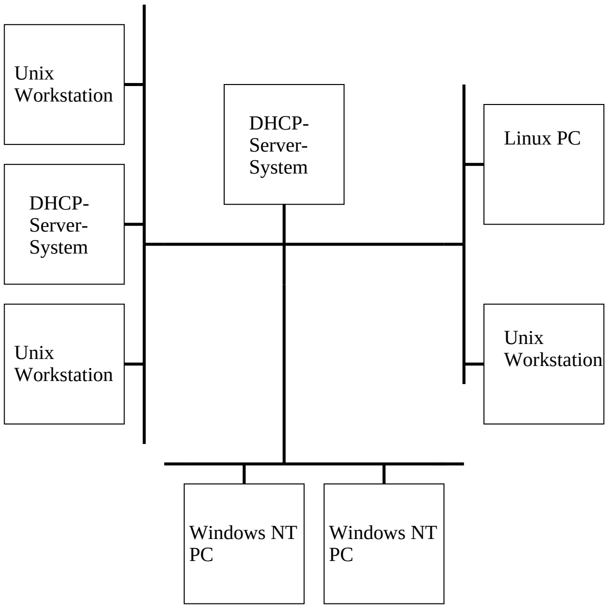

Abbildung 3.3: DHCP im Einsatz mit fest installierten Systemen

Damit sind die fest installierten Systeme für einen DHCP-"Provider" leichter zu überwachen bzw. zu managen, da DHCP-Server und Clients stationär sind und keine dynamische Adressen- bzw. Konfigurationsverwaltung nötig ist. Da die Verwendung von DHCP mit stationären Systemen typisch ist, werden an dieser Stelle die Situationen untersucht, die sich aus seinem Einsatz ergeben. Diese basieren auf der Verwendung der aktuellen Version von DHCP, welche im Fachjargon der DHCP-Newsgroups auch *DHCPv4* genannt wird, wegen des zugrundeliegenden IPv4-Protokolls. Ebenso gibt es *DHCPv6* für das neue IPv6-Protokoll; auf dieses wird in einem eigenen Kapitel eingegangen. Die Szenarien, die sich hier ergeben, sind mit den Szenarien identisch, welche bei mobilen Systemen entstehen. Wegen des hauptsächlichen Einsatzes von DHCP mit stationären Systemen soll aber nur an dieser Stelle auf sie eingegangen werden.

3.4 Szenarien unter DHCPv4

Es wird im folgenden vom einem (Sub-)Netz mit mindestens einem *DHCP-Server* und einem *DHCP-Client* ausgegangen. Zum besseren Verständnis der vorgestellten Szenarien wird der reguläre Ablauf einer Konfiguration unter DHCP in Abbildung 3.4 dargestellt.

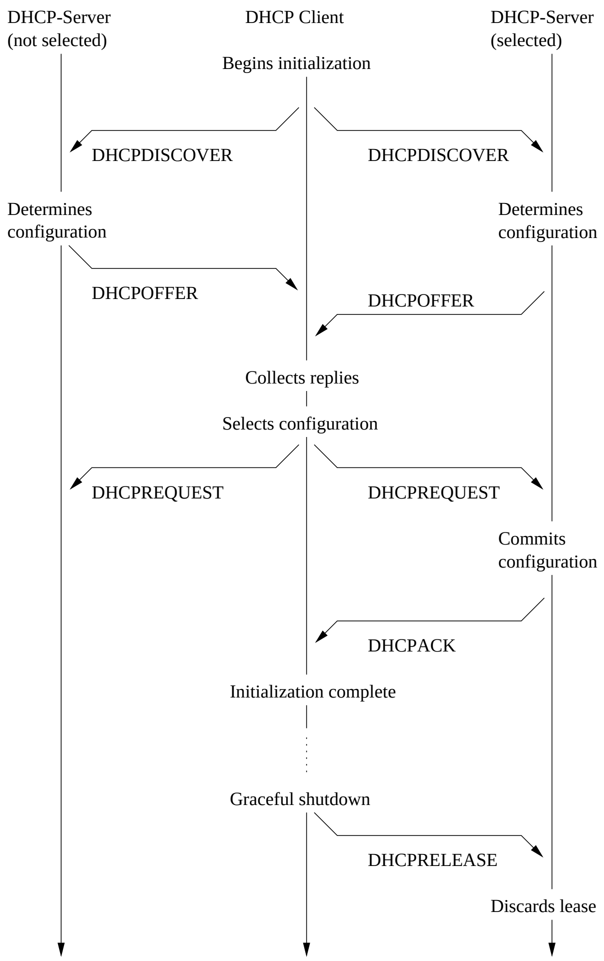

Abbildung 3.4: Zeitliniendiagramm eines typischen Client/Server-Dialogs unter DHCP

Im folgenden sollen Situationen untersucht werden, die sich im Einsatz von DHCP zur Konfiguration von Computersystemen ergeben.

Folgende Szenarien können unterschieden werden:

3.4.1 Client meldet sich neu an

Wenn ein Hostsystem in eine Netzumgebung integriert werden will, sei es ein mobiles System oder ein stationäres, welches aufgrund einer Policy-Entscheidung der Netzbetreiner nur zeitweise die Netzressourcen benutzen können soll, spielen sich folgende Abläufe ab:
Der Client benötigt eine IP-Adresse und Konfigurationsdaten.

Da er keine Serveradresse kennt, sendet er ein DHCPDISCOVER als Broadcast ([Dro93]) auf das Subnetz.
Broadcast bedeutet, daß die Meldung an die Adresse 255.255.255.255 gesendet wird, was bewirkt, daß sie an alle Systeme, die in diesem Subnetz sind, weitergeleitet wird. Den Relay Agents kommt dabei die Aufgabe zu, als eine Art Gateway das DHCPDISCOVER auch auf andere Subnetze weiterzuleiten (s.a. 2.3) [Dro93], [Dro96b].

Dieses DHCPDISCOVER kann bereits Wunschparameter des DHCP-Client enthalten, z. B. die gewünschte IP-Adresse oder die gewünschte Lease-Zeit.

Die DHCP-Server, die das DHCPDISCOVER mitbekommen, überpüfen, ob sie freie IP-Adressen in ihrem Adreßpool haben und stellen eine DHCPOFFER-PDU zusammen, in der sie dem Client auch schon bestimmte Konfigurationsparameter anbieten können.
Falls kein DHCP-Server antwortet, weil z. B. keiner aktiv ist oder gerade kein DHCP-Server freie Adressen hat, wiederholt der Client nach einem definierten Wartealgorithmus (s.2.3) sein DHCPDISCOVER [Dro93].

Empfängt der DHCP-Client innerhalb einer definierten Zeit mehrere DHCPOFFERs, so sammelt er die Angebote und wählt eines davon aus. Mit diesen Daten stellt er eine DHCPREQUEST-PDU zusammen und schickt sie direkt an den Server, von dem dieses Angebot kam.

Nun gibt es mehrere Möglichkeiten des weiteren Geschehens:

- Server empfängt kein DHCPREQUEST,
  z. B. wegen Netzproblemen (Transport mit UDP).
  Wenn der Server kein DHCPREQUEST empfängt, startet er auch keine Aktion.

- Server empfängt DHCPREQUEST und entnimmt der Meldung, daß sie nicht für ihn bestimmt ist.
  In diesem Fall verwirft der Server das DHCPREQUEST.

- Server empfängt DHCPREQUEST und entnimmt der Meldung, daß sie für ihn bestimmt ist.
  Jetzt gibt es wieder mehrere Möglichkeiten:

  - Server hat freie IP-Adresse:
    Er erstellt ein DHCPACK, merkt sich die Konfiguration in einer internen Tabelle und sendet das DHCPACK an den Client.
    Falls dieser feststellt, daß die IP-Adresse schon benutzt wird, beendet er die Konfiguration mit einem DHCPDECLINE und beginnt von neuem mit einem DHCPDISCOVER.
  - Server hat keine freie IP-Adresse:
    Er schickt eine DHCPNAK-PDU an den Client.
    Diese bedeutet, daß der Client sich momentan nicht von diesem Server konfigurieren lassen kann.

3.4.2 Client will Lease verlängern

Ein Client ist von einem DHCP-Server konfiguriert worden und versucht nun, seine Leasezeit verlängern zu lassen.

Der Client sendet als erstes ein DHCPREQUEST an den Server, von dem er konfiguriert wurde.

Dieses enthält in der Requested IP Adress Option die gewünschte Adresse.

Die Server, die die Konfiguration des Clients kennen, entscheiden nun:

- Der Request/die Nachfrage kann erfüllt werden:
  Der Server sendet ein DHCPACK an den Client

- Dem Request kann nicht entsprochen werden, weil z. B. der Client sich in einem neuen Subnetz befindet:
  Der Server sendet ein DHCPNAK an den Client.

Mögliche Aktionen des Clients:

- Der Client wartet vergeblich auf ein DHCPACK:
  Das führt nach einer definierten Zeit zu einem Timeout;
  Der Client versucht daraufhin, nach dem definierten Wiederholalgorithmus, welcher nach der sogenannten Randomized Exponential Backoff-Strategie (s. 2.3.)verfährt, sein DHCPREQUEST zu wiederholen [Dro93]. Er kann aber seine aktuelle Adresse bis zum Ende der Leasezeit weiterbenutzen.

- Der Client hat DHCPNAK empfangen:
  Der Client muß nun den Konfigurationsprozeß von neuem starten *(DHCPDISCOVER etc.)*

- Der Client hat DHCPACK empfangen:
  Er testet die zugeteilte IP-Adresse, was folgende Möglichkeiten eröffnet:

  - Die Adresse wird nicht von einem anderen System benutzt:
    Der Client ist fertig konfiguriert.

  - Die Adresse wird von einem anderen System benutzt:
    Der Client muß ein DHCPDECLINE senden und die Konfiguration von neuem starten.

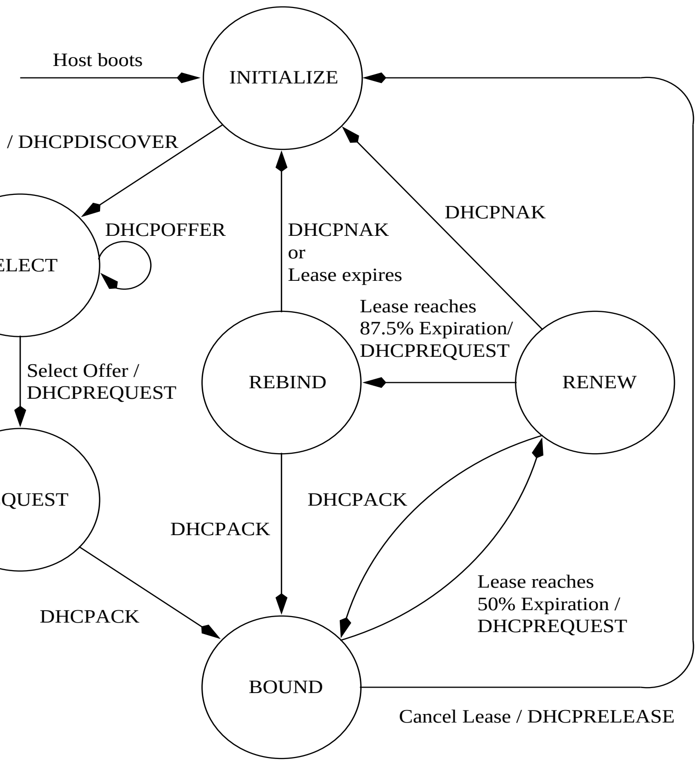

Abbildung 3.5: Zustandsübergangsdiagramm eines DHCP-Clients

(Nach [Come95])

3.4.3 Client will Lease beenden

Falls der Client aus irgendwelchen Gründen seine IP-Adresse nicht mehr benötigt, weil z. B. ein Laptop nach einem harten Arbeitstag nicht mehr gebraucht wird, so kann er seine Adresse freigeben. Dies geschieht, indem der Client ein DHCPRELEASE an den/ die Server sendet, von dem er konfiguriert wurde. Der oder die Server, die seine Konfiguration kennen, löschen daraufhin seine Konfigurationsinformation und kennzeichnen intern seine IP-Adresse als verfügbar.

3.4.4 Client hat externe IP-Adresse

Für diesen Zweck ist in DHCP die DHCPINFORM-PDU vorgesehen. Der betreffende Client sendet DHCPINFORM auf das Netz. Die erreichten Server stellen daraufhin die Client-lokalen Konfigurationsdaten zusammen und senden sie als Unicast in einem DHCPACK an den Client.

3.4.5 kritische Situationen

Beim Einsatz von DHCP kann man sich verschiedene Krisensituationen vorstellen, die evtl. behandelt werden müssen:

- Mehrere Server auf dem Netz vergeben die gleiche IP-Adresse an unterschiedliche Clients:
  Diese Situation ist in der Realität nicht denkbar, da es unter dem DHCP-Protokoll nicht möglich ist, mehrere DHCP-Server mit sich überschneidenden Adreßbereichen auszustatten, ohne von vornherein Konsistenzprobleme in Kauf zu nehmen. Diese Möglichkeit wird von den Netzbetreibern also auf jeden Fall beim Gebrauch der derzeitigen Implmentierungen von DHCP-Servern vermieden.

- DHCP-Server vergibt fehlerhafte Konfigurationsinformationen:
  Dieser Fall könnte z.B. dann auftreten, wenn sich die Netztopologie verändert (Netzkomponten wie Drucker etc. bekommen eine andere IP-Adresse oder werden ganz aus dem Netz entfernt), aber die Netzbetreiber es versäumen, die den DHCP-Servern zur Verfügung stehenden Konfigurationsdaten wie Routeradressen etc. zu aktualisieren.
  Sollte diese Situation eintreten, so liegt es in der Verantwortung der Betreiber der DHCP-Serversoftware, die Konsistenz der Daten zu gewährleisten.

- DHCP-Client sendet DHCPREQUEST ohne vorheriges DHCPDISCOVER:
  Diese Situation ist nicht realisierbar, da bei einem DHCPREQUEST die Serverhardwareadresse und eine Transaction ID (s. 2.3) benötigt wird, um es zu verschicken. Diese kann der Client aber nur kennen, wenn er mindestens einmal ein DHCPDISCOVER gesendet hat.

- DHCP-Client lehnt Lease mit DHCPDECLINE/DHCPRELEASE ab, nutzt ihn aber trotzdem:
  Diese Situation könnte z.B. dann eintreten, wenn ein Benutzer sich einen unendlichen Lease erschleichen möchte, weil er weiß, daß der DHCP-Server keine benutzten IP-Adressen vergibt (dies könnte in Abhängigkeit von der Implementierung der Serversoftware vorkommen).
  Sollte ein derartiges Verhalten von Netzbenutzern auftreten, so liegt es in der Verantwortung des Netzbetreibers, durch geeignete Maßnahmen wie das Kontrollieren von Logdateien der Server dieses Verhalten unmöglich zu machen.

- DHCP-Client erhält ein DHCPNAK, verhält sich aber als fertig konfiguriert:
  Diese Situation ist nicht real, da DHCP-Server DHCPNAKs nur versenden, falls eine IP-Adresse, die sie vergeben wollen, schon benutzt wird.

- Client besitzt eine IP-Adresse aus einer DHCP-Konfiguration und will sich über ein DHCPINFORM Daten über die Netzkonfiguration holen:
  Für diesen Fall muß in der Server-Software eine Fehlerbehandlungsroutine vorgesehen sein, wie z.B. ein Eintrag in die Logdatei des Serverprogramms.
  Diese Situation dürfte ziemlich selten vorkommen, da ein DHCP-Client für eine Neukonfiguration nur ein erneutes DHCPREQUEST senden muß, in welchem er die geforderten Parameter verlangt.

3.5 DHCP und DNS

3.5.1 Einleitung

Das *Domain Name System (DNS)*([Com95],[Moc87a], [Moc87b]), mit dem die Benutzung von domainspezifischen Bezeichnern für Netzsystemen ermöglicht wird, wird auch von DHCP unterstützt. Die *Domain* entspricht einem festgelegten Bereich, in dem für bestimmte IP-Adressen neue Bezeichner bestimmt werden. Die Unterstützung von DNS geschieht mithilfe von DHCP-Options wie z. B. der Domain Name Option und der Domain Name Server Option [Ale93],[AD96].

3.5.2 DHCP und FQDNs

DHCP unterstützt auch die Verwendung von sogenannten *Fully Qualified Domain Names (FQDNs)* statt numerischer IP-Adressen. Das bedeutet, daß für Parameter/Options wie zum Beispiel LPR-Server keine 4-Oktett-Adressen mehr angegeben werden müssen, sondern daß stattdessen FQDNs angegeben werden können. Damit werden Änderungen in der Netztopologie möglich ohne die Notwendigkeit, numerische Internetadressen in Konfigurationsfiles umzuschreiben. Die notwendigen Adreßänderungen finden über das Domain Name System statt.
Die Unterstützung von FQDNs erfolgt mittels einer speziellen DHCP FQDN Option [RD96].

3.6 DHCP und Dynamic DNS

[Rek96]: Bei der Benutzung von DHCP und DNS ergibt sich allerdings folgendes Problem: Bei der Konfiguration mittels DHCP wird die DNS-Datenbasis nicht automatisch aktualisiert, spezifisch die Name-zu-IP-Adresse und IP-Adresse-zu-Name-Bindungen, die von DNS unterstützt werden. Als Abhilfe kann man mit DHCP das sogenannte *Dynamic DNS*- V 27 erfahren benutzen, um diesem Problem abzuhelfen und den Namensraum konsistent zu halten.

Unter DNS gibt es zwei Arten von sogenannten *Resource Records (RRs)*,die die Beziehungen zwischen IP-Adressen und Fully Qualified Domain Names darstellen: Auf der einen Seite wird die Beziehung / das *Mapping* eines FQDNs zu einer IP-Adresse im sogenannten *A Resource Record* festgehalten, auf der anderen Seite die Beziehung IP-Adresse zu FQDN im *PTR Resource Record*.

Wie schon erwähnt, wird im DHCP-Protokoll zwar ermöglicht, einem DHCP-Client eine IP-Adresse zukommen zu lassen, der Update des DNS ist aber normalerweise nicht vorgesehen. Dieses Problem läßt sich mit Einsatz des Dynamic DNS lösen und zwar gibt es da zwei Möglichkeiten:

- Der DHCP-Client erneuert den A Resource Record, der DHCP-Server erneuert den PTR Resource Record

- Der DHCP-Server erneuert sowohl den A Resource Record als auch den PTR Resource Record

Um eine allgemeingültige Regelung innerhalb eines Systems zu treffen und weil der Client die Konfiguration unter DHCP anstößt, muß der Client festlegen, welche der Möglichkeiten verwirklicht wird.
Diese Entscheidung ist aber von der Policy der Netzbetreiber abhängig, welche gegebenenfalls auf die Implementierung der Clients dahingehend Einfluß nehmen müssen.
Der einzige Unterschied zwischen den beiden Möglichkeiten ist, daß in einem Fall der DHCP-Client den FQDN-zu-IP-Adresse-Mapping Eintrag macht; bei beiden Möglichkeiten erneuert der Server das IP-Adresse-zu-FQDN-Mapping.

Um den IP-Adresse-zu-FQDN-Eintrag zu erneuern, muß der DHCP-Server den Fully Qualified Domain Name des Clients kennen. Da er diesen nicht von Anfang an kennt, muß der Client ihm diesen mitteilen. Dies geschieht mithilfe der Client FQDN Option [Rek96]. Mit dieser kann der Client dem Server außerdem mitteilen, ob er selbst das FQDN-zu-IP-Adresse-Mapping durchführt oder ob dies der Server für ihn erledigen soll.

Falls nun der Client dieses Mapping selbst durchführen will, so fügt er die Client FQDN Option in die DHCPREQUEST-Meldung an den DHCP-Server ein, von dem er sich konfigurieren lassen will und setzt ein bestimmtes Flag in der Option. An diesem erkennt der Server die Absicht des Clients.

Sobald der Client das DHCPACK vom Server erhält, erneuert er den A Resource Record nach den Regeln für dynamische Updates des DNS [VTRB96].

Wenn der Server das Update des A Resource Records durchgeführt hat, kann der Client den Erfolg des Updates nur aus einem DNS Query ableiten.

Zum Server: Wenn der Server auf ein DHCPREQUEST mit Client FQDN Option mit einem DHCPACK reagiert, *(d. h. wenn er dem Client eine IP-Adresse zuteilen und ihn konfigurieren will (s. 2.3))*, dann initiiert er das Update des PTR Resource Records [VTRB96]. Dies geschieht aber erst, nachdem er das DHCPACK an den Client abgeschickt hat.

3.7 Renumbering mit DHCP

DHCP ist aufgrund seiner Definition in der Lage, IP-Systeme dynamisch zu konfigurieren. Ein DHCP-Client kann jederzeit einen veränderten Parameter der Netzkonfiguration erneuern, indem er eine DHCPRQUEST-Meldung an den DHCP-Server schickt. Ein besonders interessantes Szenario für diese dynamische Rekonfigurierung stellt das sogenannte *Renumbering* in IP-Netzen dar.
Unter Renumbering versteht man eine Neuvergabe von IP-Adressen, die in der Realität zum Beispiel dann nötig wird, wenn ein Netz von einem neuen Netzprovider bedient wird. Teilweise ist DHCP von sich aus in der Lage, Renumbering durchzuführen. Dies ist der Fall, wenn ein DHCP-Client versucht, einen bestehenden Lease zu verlängern. Der DHCP Server kann dann über ein DHCPNAK dem Client bedeuten, sich eine neue Adresse zuteilen zu lassen.
Da Renumbering nicht standardmäßig Bestandteil des DHCPv4-Protokolls ist, muß man DHCP-Server dazu bringen, es zu unterstützen. Weiterhin ist es in DHCPv4 nicht vorgesehen, daß der DHCP-Server seine Clients zu einer Rekonfiguration auffordern kann. Diese Option wird erst in DHCPv6 zur Verfügung stehen (s. Kapitel 4). Der Server muß deshalb für ein geplantes Renumbering auf die Anfrage der Clients für die Leaseverlängerung zurückgreifen.
Das hier vorgestellte Verfahren bringt mit sich, daß der Client zeitweilig zwei Adressen benutzen kann, seine alte und die neu zugeteilte. Dies wird ermöglicht durch eine zusätzliche DHCP-Option ([Gil96]), durch die der Server das Renumbering beim Client anstoßen kann, die sogenannte *Renumbering Option*. Diese wird bei einem DHCPACK eingefügt und bedeutet dem Client, daß er sich um eine neue IP-Adresse bemühen soll. Die neue Adresse wird im folgenden als *Post-Renumbering-Adresse* bezeichnet. Der Client benutzt dann die neue Option in einem DHCPDISCOVER, um das Renumbering anzustoßen. Der Server kann die einzelnen Renumbering-Dialoge mit den Clients mittels eines *Magic Cookie-Codes* identifizieren, der von den Clients nicht verändert werden darf [Gil96].

Des weiteren wird davon ausgegangen, daß ein DHCP-Server mit den neuen Post-Renumbering-Adressen konfiguriert wird, bevor die Relay Agents, die auf ihn verweisen, rekonfiguriert werden. Diese Strategie vermeidet Situationen, wo ein DHCP-Client ein DHCPACK empfängt, das ihn zum Renumbering auffordert, aber der Relay Agent selbst noch keine Post-Renumbering-Adress besitzt.In diesem Falle könnte der Relay Agent das DHCPDISCOVER für die neue Adresse nicht weiterleiten. Nachdem der Client das Renumbering erfolgreich abgeschlossen hat, wird davon ausgegangen, daß er seine alte "Pre-Renumbering"-Adresse verwirft.

3.7.1 Ablauf des Renumberings beim DHCP-Client

Der Ablauf der Bemühung des Clients um eine neue IP-Adresse und Konfiguration wurde schon als Zustandsdiagramm in Kapitel 2 vorgestellt. Im folgenden wird auf diesen Ablauf Bezug genommen, indem die Namen der einzelnen Zustände des Ablaufs (INITIALIZE, REQUEST, etc.) verwendet werden.
Wenn nun ein Client ein DHCPACK mit Renumbering-Option empfängt, das ihn vom REQUEST- in den BOUND-Zustand übergehen ließe, überprüft er, ob er schon wegen einer Post-Renumbering-Adresse mit einem DHCP-Server kommuniziert:

- Falls nicht, startet er einen zweiten DHCP-Dialog entsprechend dem Diagramm 2.1 aus Kapitel 2 (s.a. 3.4) mit dem Server, um eine neue (Post- Renumbering-) Adresse zu erhalten.
  Das bedeutet, daß ab diesem Moment vom Client zwei DHCP-Konfigurationsprozesse in Gang gehalten werden, einer für die alte Adresse, einer für die neue.

- Falls schon ein Renumber-Dialog existiert, sendet der Client ein DHCP-RELEASE an den Renumber-DHCP-Server und startet einen neuen Renumber-Dialog.

Wenn der Client das DHCPACK schon im BOUND, RENEW oder REBIND-Zustand empfängt, markiert er den laufenden Adreßvergabeablauf als "entwertet" und initiiert einen neuen. Der neue Ablauf startet im INITIALIZE-Zustand. Die mit DHCPACK empfangene DHCP-Option muß nun in das DHCPDISCOVER kopiert werden, mit dem der neue Ablauf beginnt. Der Client muß bei Abschluß des Renumberings entscheiden, wann er den alten Lease durch den neuen ersetzt.

Wenn der Client das DHCPACK ohne Renumbering-Option empfängt, überprüft er, ob er einen als "entwertet" bezeichneten Konfigurationsprozeß besitzt, *(d. h. , ob er sich um ein Renumbering bemüht hat)* :

- Falls ja, soll er die neuen Konfigurationsparameter übernehmen und für seine Netzkommunikation benutzen. Das Renumbering ist damit für ihn abgeschlossen. Der alte, "entwertete" Lease läuft dann aus, ohne erneuert zu werden.
- Falls nicht, hat er sich nicht um ein Renumbering bemüht und ist damit *(mit der Pre-Renumbering-Adresse)* konfiguriert.

3.7.2 Ablauf des Renumberings beim DHCP-Server

Der Server muß in seiner Client-Adressen-Datenbasis zusätzlich die Information speichern, ob eine IP-Adresse zu den Pre-Renumbering- oder zu den Post-Renumbering-Adressen gehört oder nicht. Wenn nun ein DHCP-REQUEST beim Server ankommt, muß der überprüfen, ob die Adresse des Clients zu der einen oder anderen Gruppe gehört. Wenn der Client seine Pre-Renumber -Adresse verlängern will, fügt der Server die Renumbering-Option in das DHCPACK ein, wobei er als Leasezeit nur den verbleibenden Rest des aktuellen Leases verordnet. Sollte der Server über keine freie Post-Renumbering-Adresse verfügen, dann läßt er den Konfigurationsdialog mit dem Client ablaufen unter Verwendung der Pre-Renumbering-Adresse, ohne den Client zum Renumbering aufzufordern.

3.7.3 Renumbering und der Relay Agent

Die einzige Anforderung an den Relay Agent besteht darin, daß er bei der Weitergabe einer DHCP-Nachricht in das Feld ROUTER IP ADRESS (s.a. Abbildung 2.3) seine (Post-Renumbering)-Adresse einträgt.

3.8 Zusammenfassung

In diesem Kapitel wurden unterschiedlichste Szenarien für den Einsatz von DHCP vorgestellt, angefangen von dem Einsatz bei mobilen Systemen, Mobile-IP, dem Einsatz bei stationären Systemen, der heute am weitesten verbreitet ist, bis zu Szenarien, die sich aus dem Protokollablauf ergeben. Als mögliche Einsatzszenarien wurde die Benutzung von DNS und Dynamic DNS vorgestellt, sowie Renumbering unter DHCP. Dieses Kapitel dient damit als Ausgangsbasis für die Entwicklung von Anforderungen an ein DHCP-Management.

Kapitel 4

Änderungen bei DHCPv6

4.1 Einleitung

Eine große Problematik des Internet stellt u. a. der für zukünftige Zwecke zu geringe Adreßraum in IPv4, der zur Zeit noch hauptsächlich verwendeten Version des *Internet Protocols*, dar. Schon für das Jahr 2005 wird mit dem Erreichen der Grenzen des derzeitigen Adreßraumes gerechnet. Als Abhilfe für das Adreßraumproblem und Nachfolger von IPv4 wurde *IPv6* entwickelt [SH95], dessen Entstehungsprozeß zur Zeit aber noch nicht abgeschlossen ist. Laut seiner Spezifikation verfügt der Internetbenutzer neben anderen Neuerungen künftig über 128-Bit- statt 32-Bit Adressen. Informationen über IPv6 sind auch aus aktuellen Arbeiten beziehbar [MS96].

Im Zusammenhang mit der Entwicklung von IPv6 wurden auch Überlegungen angestellt, wie das DHCP-Protokoll an die neue IP-Version angepaßt werden kann. Diese Entwicklung ist zum Zeitpunkt dieser Diplomarbeit noch voll im Gange. Das entstehende *DHCPv6* wird zwar schon von einigen DHCP-Serversoftwarevarianten unterstützt, kann aber erst dann sinnvoll eingesetzt werden, wenn sich das DHCPv6-Protokoll gefestigt hat. Das hier vorgestellte Protokollverhalten von DHCPv6 ist ebenfalls noch in Entwicklung und gibt nur einen temporären Stand wieder. Ebenso wie DHCP für IPv4 ist es im Rahmen dieser Diplomarbeit auf seine Einsetzbarkeit untersucht worden. Dabei hat es sich herausgestellt, daß es zur Zeit noch keinen Sinn macht, Folgerungen aus den Protokollfestlegungen zu ziehen, da diese sich zur Zeit immer noch sehr stark verändern bzw. einige Details noch gar nicht berücksichtigt sind [BP96],[Per95],[Per96a]. DHCPv6 wird hier vor allem im Hinblick auf künftige Tätigkeiten in dieser Richtung vorgestellt.

4.2 Änderungen zu DHCPv4

Bei DHCPv6 ändern sich einige Dinge, die man von DHCPv4 gewohnt ist. Die von DHCPv4 gewohnten Relay Agents heißen jetzt nur noch *Relays*. Auch ihr Verhalten ändert sich: Während bei DHCPv4 die Relay Agents eine relativ passive Rolle spielen und vor allem bei Beginn der Kommunikation zwischen DHCP-Server und DHCP-Client wichtig sind, übernehmen sie bei DHCPv6 einige Aufgaben der DHCP-Server, die auch in Hinsicht auf das Netzmanagement interessant sind. Weiterhin gibt es im neuen Protokoll den Begriff der *DHCP Agents*. Diese spielen eine Vermittler- und kontrollierende Rolle bei der Kommunikation zwischen DHCPv6-Clients und DHCPv6-Servern. Die Rolle von DHCP Agents kann sowohl von DHCPv6 Servern als auch Relays ausgefüllt werden.

Außerdem wird bei DHCPv6 die Größe der zwischen Server und Client hin- und herlaufenden PDUs nicht mehr festgeschrieben; sie schwankt je nach Anforderung und ermöglicht damit auch eine Verringerung der Netzlast.

Zusammen mit den Aufgaben ändern sich auch die Bezeichnungen für die PDUs: Sie heissen jetzt nicht mehr DHCPDISCOVER, DHCPOFFER, etc. sondern *DHCP Solicit, DHCP Request, DHCP Advertise* etc. . Ein weiterer neuer Terminus ist die sogenannte *link-local-Adresse*: Eine link-local-Adresse dient dazu, benachbarte IPv6-Knoten im Netz zu erreichen, die am gleichen Link angeschlossen sind. Zum Beispiel alle Interfaces haben eine solche link-local-Adresse.

Zum besseren Verständnis des Protokollablaufs wird hier eine Übersicht der von Client, Relay und Server benutzten Meldungen gegeben:

4.3 PDUs des DHCPv6-Clients

Vom DHCPv6-Client werden folgende PDUs benutzt:

DHCPv6-PDU	Bedeutung
DHCP Solicit	Anfrage wg. IP-Adresse und Konfiguration
DHCP Request	Anforderung von IP-Adresse und Konfiguration oder geänderten parametern nach DHCP Reconfigure
DHCP Release	(an Server) Freigabe von Ressourcen

4.4 PDUs des DHCPv6-Relays

Vom DHCPv6-Relay werden folgende PDUs benutzt:

DHCPv6-PDU	Bedeutung
DHCP Advertise	Angebot von Serveradressen
DHCP Request	an Server weitergeleitete Anforderung des Clients
DHCP Reply	an Client weitergeleitete IPv6-Adresse und Konfiguration

4.5 PDUs des DHCPv6-Servers

Vom DHCPv6-Server werden folgende PDUs benutzt:

DHCPv6-PDU	Bedeutung
DHCP Advertise	Angebot IP-Adresse und Konfiguration (Server-Bit gesetzt)
DHCP Reply	Zuteilung von IPv6-Adresse und Konfiguration oder Mitteilung, daß DHCP Request an falschen Server geschickt wurde
DHCP Reconfigure	(an Client) Aufforderung zur Neukonfiguration/Renumbering

4.6 Szenarien unter DHCPv6

Ähnlich wie bei DHCPv4 ergeben sich auch unter DHCPv6 Situationen, die für das Management von Interesse sein könnten. Wie schon im vorhergehenden Kapi Folgende Szenarien ergeben sich unter DHCPv6:

4.6.1 Client meldet sich neu an

Ein Client taucht neu in einem Netz auf. Um von einem Server konfiguriert zu werden und eine IP-Adresse zu bekommen, muß er zunächst einen DHCP Agenten lokalisieren.
Zu diesem Zwecke sendet er als Multicast ein DHCP Solicit an die Adresse FF02:0:0:0:0:0:1:0 mit TTL=1.

Ein DHCP Agent (Server oder Relay) empfängt das DHCP Solicit und sendet ein *DHCP Advertise* mit allen Serveradressen, für die er werben soll über dieselbe Verbindung zurück.

Es ist möglich, den DHCP Agent periodisch DHCP Advertise-Meldungen an die All-DHCPv6-Clients Multicast-Adresse schicken zu lassen (mit TTL=1), um die Clients so oft wie möglich über den Zugang zu DHCPv6-Servern zu informieren.

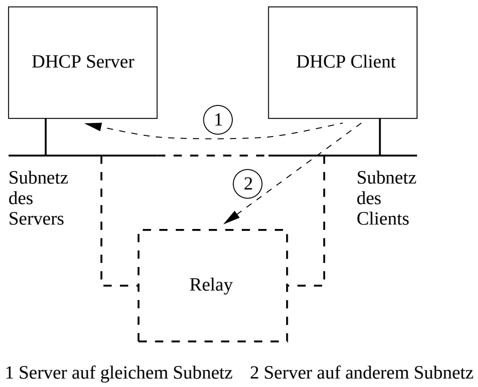

Abbildung 4.1: DHCP Solicit : Mögliche Wege

Falls ein DHCP-Server das DHCP Solicit empfängt, stellt er ein DHCP Advertise zusammen, in dem er das sogenannte S-Bit (für Server) setzt. Das Advertise schickt er an die IPv6-Adresse des Interfaces, von dem er das Solicit empfangen hat.

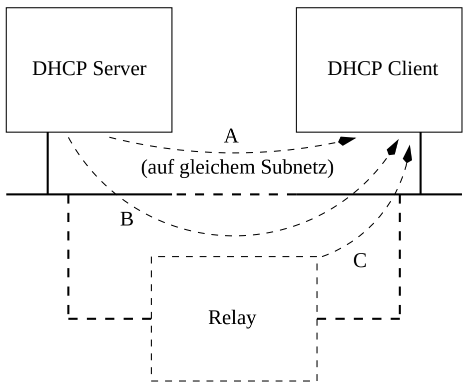

Abbildung 4.2: DHCP Advertise : Mögliche Wege

Der Client empfängt das DHCP Advertise.

Jetzt bieten sich mehrere Möglichkeiten:

- Client hat Advertise ohne Serveradressen und ohne gesetztes Server-Bit empfangen:
  Dann verwirft er das Advertise.

- Client hat Advertise mit gesetztem ServerBit empfangen:
  Das Advertise kam von einem Server und der Client kann sich eventuell angebotene Parameter für die Netzkonfiguration aussuchen.
  Jetzt kann der Client ein DHCP Request zusammenstellen, also eine Bitte um eine IPv6-Adresse plus Konfigurationsparametern und schickt diese an die Agentenadresse.

- Der Client hat schon einmal ein DHCP Request gesendet, aber keine Antwort erhalten:
  Dann muß er erneut einen identischen DHCP Request senden.
  Für die Retransmission von DHCPv6-PDUs werden anders als bei DHCPv4 (s. 2.3) Grenzzeitparameter festgelegt, welche vom Netzbetreiber bestimmt werden können [BP96])

Ein Relay empfängt das DHCP Request und überprüft folgende Eigenschaften der Meldung:

1. kommt der Request von einer link-local-Adresse (s. o.)?

2. stimmt die link-local-Adresse mit dem link-local-Adress-Feld im Header des DHCP Request überein?

3. stimmt das agent-adress-Feld im Request mit der Adresse des Interfaces überein, von dem der Request empfangen wurde?

4. Ist das S-Bit gesetzt?

Falls alle Bedingungen erfüllt sind, schickt das Relay den Request an den Server weiter, dessen Adresse es dem DHCP Request entnimmt.
Andernfalls verwirft es den Request.

Der Server empfängt den Request vom Relay und startet folgende Aktionen: Er überpruft anhand der Transaction ID, ob der Request wiederholt wurde: *(Diese ID kennzeichnet jeden einzelnen Client-Server-Dialog.)*

- Falls der Request wiederholt wurde:
  Server schickt denselben DHCP Reply an den Client wie nach dem ersten Request.

- Falls der Request nicht wiederholt wurde:
  Der Server überprüft, ob die Adresse im link-local-Feld gültig ist: Ist sie

  - nicht gültig,
    verwirft er den Request
  - gültig,
    prüft er, ob das S-Bit gesetzt ist:

  - Falls das S-Bit nicht gesetzt ist:
    Dann verwirft er den Request
    Vermutung, ist im DHCPv6-Draft nicht beschrieben
  - Falls das S-Bit gesetzt ist:
    Dann überprüft der Server, ob seine eigene Adresse mit der IPv6-Destination-Adresse im Request übereinstimmt:

    * Falls sie nicht übereinstimmt, verwirft er den Reqest und sendet ein dementsprechendes DHCP Reply an den Agenten.
    * Falls sie übereinstimmt, entnimmt er dem Request die link-local-Adresse und die Agentenadresse und kreiert einen Eintrag für den Client.
      Dieser enthält unter dem Titel (link-local-Adresse, Agentenadresse) die Konfigurationsdaten für den Client.
      Der Server sendet ein DHCP Reply an den Client mit den Konfigurationsparametern.
      Sollte die empfangene link-local-Adresse und die empfangene Agentenadresse nicht zu irgendeinem anderen Binding (link-local-Adresse,Agentenadresse) passen, kann der Server ein neues Binding kreieren oder ein DHCP Reply an den Agenten mit Error Code 5 zurücksenden.

Empfängt der Relay ein DHCP Reply, dann überprüft er:

1. Ist das sogenannte L-Bit gesetzt, d. h. will ein lokaler Server an einen bestimmten Client auf dem lokalen Subnetz senden?
2. Hat das link-local-Feld der Meldung den Präfix FE80::00?

Falls beide Bedingungen erfüllt sind, sendet er den Reply an die link-local-Adresse, die er dem DHCP Reply entnimmt.
Die Alternativreaktion, wenn mindestens eine oder beide Bedingungen nicht erfüllt sind, wird vom Protokoll noch nicht festgelegt.

Der Client empfängt den DHCP Reply vom Relay und überprüft als erstes dessen Transaction ID, die seinen spezifischen Kontaktversuch zum DHCP-Server kennzeichnet. Ist diese

- nicht identisch mit der eines seiner DHCP Requests:
  Dann verwirft der Client den Reply und trägt eine Errormeldung in sein Log ein.
- identisch mit der ID eines seiner DHCP Requests:
  Dann überprüft er, ob das L-Bit in der Meldung gesetzt ist *(Server auf gleichem Subnetz)*:
  - Falls das L-Bit gesetzt ist:
    Der Client überprüft, ob die Source Adress im IPv6-Header identisch ist mit der Agentenadresse unter dieser Transaction ID:
    * Falls die Adressen nicht identisch sind,
      Dann verwirft der Client den Reply und trägt einen Fehler in die Log-Datei ein
    * Falls die Adressen identisch sind,
      holt er sich die Konfigurationsparameter aus den Extensions des Reply und ist damit konfiguriert.
  - L-Bit nicht gesetzt: Der Client überprüft, ob die Source Adresse im IPv6-Header identisch ist mit der Serveradresse unter der Transaction ID:
    * Falls nicht, verwirft er den Reply und trägt den Fehler ins Logbuch ein

∗ Falls sie identisch sind, holt er sich die Konfigurationsparameter aus den Extensions des Reply und ist damit konfiguriert.

4.6.2 Client will Ressourcen freigeben

Mit Ressourcen ist z. B. die IPv6-Adresse des Clients gemeint. Falls der Client Ressourcen freigeben will, sendet er ein DHCP Release an den Server, dessen Adresse er aus der Server-Ressourcen-Zuordnungsliste entnimmt. Diese Liste ist im Protokoll (noch) nicht näher beschrieben. Man kann sie sich wohl als eine Art Hash-Tabelle vorstellen.

4.6.3 Server will Client(s) neu konfigurieren

(Dieses Verfahren wird vom Internet-Draft als Work in Progress bezeichnet.) Nachdem es unter DHCPv4 dem Server nicht möglich ist, Clients zur Neukonfigurierung/Renumbering aufzufordern (s.2.4), haben sich die Entwickler der neuen Version daran gemacht, diesen Mißstand zu beheben. Sollte sich an der Situation in einer bestimmten unter DHCPv6 konfigurierten IPv6-Umgebung etwas ändern (z. B. ein Line Printer ist von einem Gebäude in ein anderes umgezogen und hat eine andere IP-Adresse), so kann man mit DHCPv6 ganz elegant die notwendigen Änderungen für die DHCP-Clients durchführen.

Dies geschieht mit Hilfe der DHCP Reconfigure-PDU. Wenn eine Änderung in der Netzkonfiguration der Clients notwendig wird, sieht der Server in seinem Datenbestand seiner betreuten Clients nach, welche von der Änderung betroffen sind und schickt an diese ein DHCP Reconfigure. Als Extensions hängt er die zu verändernden Parameter an.

Die Clients empfangen das Reconfigure vom Server und senden einen neuen DHCP Request an diesen mit den gewünschten Parametern als Extensions.

Der weitere Ablauf entspricht dem schon oben geschilderten Verhalten des Clients, der ein DHCP Request sendet bis zur endgültigen Konfiguration.

4.7 Zusammenfassung

In diesem Kapitel wurde eine Art vorläufiger Überblick über das DHCPv6-Protokoll gegeben. Da dieses Protokoll noch relativ neu ist und starken

Veränderungen unterworfen ist, wird es nicht in die Erstellung eines DHCP-Managements mit einbezogen.

Kapitel 5

Managementanforderungen für DHCP

5.1 Einleitung

Um die Voraussetzungen für ein integriertes Netz- und Systemmanagement für DHCP-Umgebungen zu schaffen, muß man als erstes die Szenarien für den Einsatz von DHCP kennen. Nachdem diese im dritten Kapitel vorgestellt wurden, können jetzt die Anforderungen untersucht werden, die ein Netzadministrator an das Management eines mit DHCP konfigurierten Subnetzes hat.

Um das Netz- und Systemmmanagement einer mithilfe von DHCPv4 konfigurierten Netzumgebung zu bewerkstelligen, benötigt man auf der einen Seite eine Zentrale, welche es ermöglicht, das Netz zu überwachen und auf es einzuwirken. Diese Zentralfunktion soll eine *Management-Anwendung* für DHCP erbringen. Auf der anderen Seite muß man sich überlegen, welche Anforderungen an die DHCP-Server oder DHCP-Clients gestellt werden. Abbildung 5.1 gibt einen Einblick in ein Einsatzszenario unter diesem Gesichtspunkt.
Wenn man davon ausgeht, daß DHCP-Server bzw. DHCP-Clients im Sinne des Managements in die Lage versetzt werden müssen, mit der Managementzentrale zu kommunizieren, dann stellt sich zunächst die Frage, ob es Sinn macht, daß sowohl Server als auch Clients in die Lage ver-

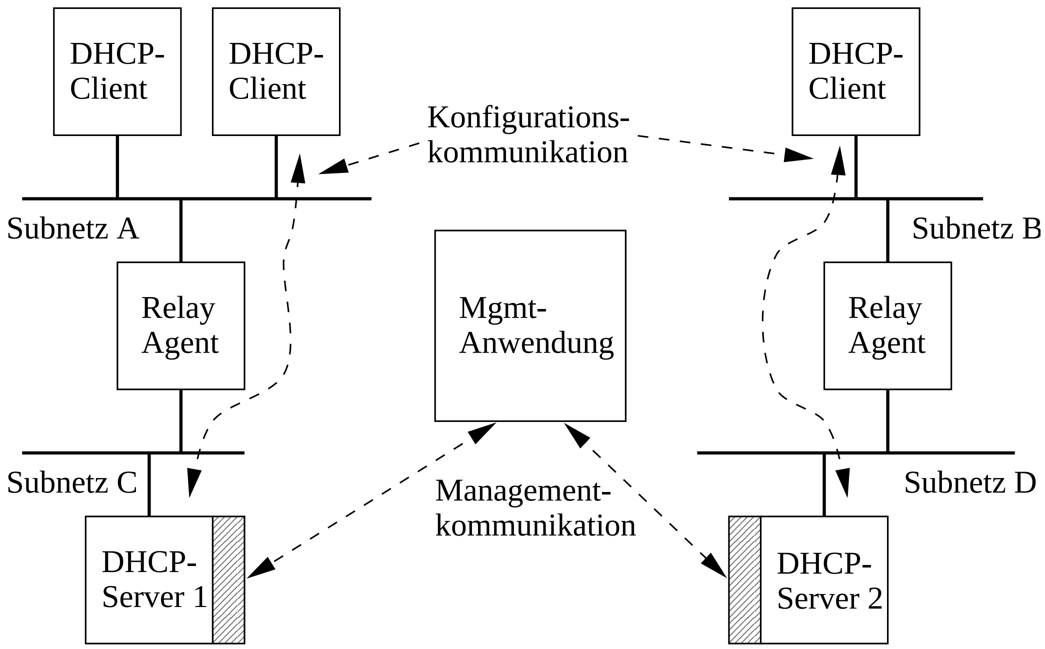

Abbildung 5.1: Einsatz von DHCP im Managementszenario

setzt werden sollen, dies zu tun. Wenn man Server und Client mit der Managementzentrale kommunizieren läßt, muß man sowohl in der Server- als auch in der Clientsoftware Schnittstellen für diese Kommunikation implementieren. Wenn man nun von Szenarien ausgeht, wo ein mobiles System in eine fremde Netzumgebung integriert werden soll, kann es zu der Situation kommen, daß ein Client aus einer nicht gemanagten Umgebung in eine gemanagte überwechselt. Daraus könnte das Problem entstehen, daß für unterschiedliche Netzumgebungen unterschiedliche DHCP-Softwarevarianten erforderlich werden. Außerdem kommt es durch die Implementierung einer Management-Schnittstelle für DHCP-Clients bei Netzen mit einer Vielzahl von IP-Systemen zu einer Erhöhung der Netzlast, da jeder DHCP-Client mit der Zentrale kommunizieren muß. Wenn man nur die DHCP-Server mit einer Management-Schnittstelle ausstattet, kann man sowohl die Server als auch über diese die konfigurierten Clients überwachen, was als sinnvolle Lösung für das Management erscheint. Allerdings soll damit nicht gesagt werden, daß das Management von DHCP-Client-Computerystemen keinen Sinn macht. Diese

Entscheidung hängt vielmehr von der Policy der Netzbetreiber ab.

Zusammengefaßt sind also Anforderungen an die Management-Anwendung und an die DHCP-Server von Interesse. Im folgenden soll zunächst auf die Anforderungen an die Management-Anwendung eingegangen werden.

5.2 Anforderungen an die Management-Anwendung

Wie schon weiter oben angeführt, ist die Management-Anwendung der zentrale Punkt für das Management einer mit DHCPv4 konfigurierten Umgebung. Nachdem es noch keine Vorgabe für eine DHCP-Management-Anwendung gibt, muß an dieser Stelle überlegt werden, welche Anforderungen an sie gestellt werden.
Auf der einen Seite soll sie einem Netzüberwacher die Möglichkeit bieten, die Vorgänge im Netz zu überwachen und zu steuern (Benutzerschnittstelle). Wie diese Benutzerschnittstelle aussieht, ist dem Implementierer der Management-Anwendung vorbehalten und soll hier nicht weiter diskutiert werden, da die Anforderungen an die Benutzerschnittstelle stark von der Erfordernissen und Wünschen der Netzbetreiber abhängen.

Falls es durch die Implementierung der DHCP-Server-Software zu Krisensituationen wie unter 3.4.5 kommen kann, muß die Management-Anwendung dazu beitragen können, diese Situationen aufzulösen. Dies kann durch Meldungen an die Netzadministration über die Benutzerschnittstelle geschehen. Eine genaue Vorschrift für die Behandlung dieser Krisen hängt aber weitgehend von der Implementierung der Server-Software ab und soll hier nicht weiter als nötig behandelt werden (s.a. 5.2.2). Ebenso ergeben sich aus diesen Situationen spezielle Anforderungen an DHCP-Server, welche unter 5.3 behandelt werden.

Es bleibt nun zu untersuchen, welche Anforderungen an die Datenhaltung der Management-Anwendung gestellt werden. Auf der anderen Seite muß sie die Daten über die von ihr betreuten Subnetze überwachen und pflegen. Das heißt nicht, daß sie nur Daten über DHCP-Server und Clients halten muß, vielmehr kann sie auch Daten über Systeme halten, die nicht mit DHCP konfiguriert wurden. Wie die Datenhaltung im Endeffekt aussieht, hängt mit von der Management-Policy der Netzbetreiber ab. Im Brennpunkt dieser Diplomarbeit steht aber das DHCP-Management, so daß nur die Daten untersucht werden, die mit DHCP konfigurierte Systeme betreffen.

Diese Daten lassen sich in zwei Gruppen unterteilen. Als erstes sind die Daten zu nennen, welche die Konfigurationsvorgaben für das gesamte von der DHCP-Management-Anwendung betreute Netz wiedergeben, die sogenannten *Grundkonfigurationsdaten*. Die zweite Gruppe besteht aus den Daten über die *aktuelle Situation* im Einsatzbereich der DHCP-Server und enthält die vergebenen IP-Adressen mit den Parametern, welche die konfigurierten Hosts kennzeichnen.

Aus dem Einsatz von DHCP unter verschiedenen Szenarien, die in Kapitel 3 vorgestellt wurden, ergeben sich ebenfalls Parameter, welche entweder zur Datenhaltung der Grundkonfiguration gehören oder zur Datenhaltung über die sich im Netz befindenden DHCP-Clients. Im folgenden sollen zunächst untersucht werden, welche Daten zur Grundkonfiguration gehören, bevor auf die Daten über die aktuelle Situation eingegangen wird.

5.2.1 Grundkonfigurationsdaten

Bei den Daten der Grundkonfiguration kann man von ihrer Verwendung her wiederum zwei Gruppen betrachten.

Die erste Gruppe besteht aus den Daten, die nur die Management-Anwendung halten muß, weil sie den DHCP-Servern nicht übermittelt werden müssen, wie z.B. die IP-Adresse eines DHCP-Client-Systems in einem anderen Subnetz. Welche Daten bestimmten DHCP-Servern nicht übermittelt werden müssen, hängt hauptsächlich von der Netztopologie ab und muß von den Netzbetreibern entschieden werden.

Die zweite Gruppe besteht aus den Daten, die sowohl die Management-Anwendung als auch die DHCP-Server halten müssen, um eine erfolgreiche Konfiguration der DHCP-Clients zu gewährleisten wie die verfügbaren IP-

Adressen etc. . Zu dieser Gruppe gehören Informationen über die Grundkonfiguration wie:

- die verfügbaren IP-Adressen (IPv4 oder v6).
  Diese ermöglichen dem oder den DHCP-Servern, neu ankommende Client-Anfragen zu bedienen.

- die durchschnittliche Lease-Zeit (s. 3.4.) für bestimmte Subnetze.
  Von diesen Werten benötigt der DHCP-Server aber nur die Lease-Zeiten der Netze, die er konfiguriert.

- die maximale Leasezeit für bestimmte Subnetze.
  Von diesen Werten benötigt der DHCP-Server ebenfalls nur die Lease-Zeiten der Netze, die er konfiguriert.

- die Domain-Namen der vom DHCP-Server betreuten Subnetze.
  Diese benötigt der Server, um sie den Clients mitzuteilen.

- Konfigurationen für bestimmte durch die Hardwareadresse erkennbare Hosts.
  Es kann vorkommen, daß einige Systeme z.B. durch ihre herstellerspezifischen Eigenschaften spezielle Parameter benötigen. Beispielsweise könnte im Zusammenhang mit exotischen Betriebssystemen Zugriffe auf spezielle Server notwendig sein.

- aus dem Einsatz von Renumbering (s. 3.7.) :
  Post-Renumbering-Adressen von Netzkomponenten wie Routern etc.
  Diese dienen dazu, daß der DHCP-Server nach dem Anstoßen eines Renumberings DHCP-Clients konfigurieren kann.

- aus dem Einsatz von *Mobile IP* (s.3.2):
  Die Adressen der Home Agents aus der Mobile IP Home Agent-Option (Nr. 68)

- aus der Authentifizierung von DHCP-PDUs (s. 2.4.):
  für jeden Client die Schlüssel, mit dem die DHCP-Nachrichten verschlüsselt werden bzw. andere Authentifikationsparameter

Die bis jetzt genannten Daten müssen ebenfalls von der Management-Anwendung gehalten werden, damit sie bei einer Änderung der Grundkonfiguration durch die Netzmanager die DHCP-Server über die neue Lage informieren kann. Eine solche Änderung könnte zum Beispiel nötig werden, wenn für ein Subnetz der Andrang auf die IP-Adressen so groß ist, daß die Netzbetreiber beschließen, die maximalen Leasezeiten zu verkürzen, um mehr Hostsystemen den Zugang zu ermöglichen.

Außer den bisher genannten Parametern gibt es noch etliche subnetzspezifische Standardparameter für die Netzgrundkonfiguration, die ebenfalls von Server und Management-Anwendung gehalten werden müssen. Einige dieser Parameter sind:

- Subnet Mask
- Router-Adressen
- Timer Server-Adressen
- Name Server-Adressen
- Domain Name Server-Adressen
- Log Server-Adressen
- LPR Server-Adressen (Drucker-Server)

Weitere Parameter dieser Art lassen sich dem RFC 1533 und seinen Nachfolge-Internet-Drafts entnehmen [Ale93], [AD96].

5.2.2 Daten über die aktuelle Netzkonfiguration

Außer den Grundkonfigurationsinformationen sollen Informationen über die *aktuelle Netz-Situation* bei Server und Management-Anwendung gespeichert werden. Zu diesen Informationen gehören unter anderem:

- die zugeteilten IP-Adressen.
  Diese benötigt der Server, um Mehrfachzuteilungen vermeiden zu können.

- die Hardwareadressen der Clients.
  Diese dienen den Servern zur eindeutigen Identifikation der Client-Systeme.

- bei mehrfachen Servern:
  die IP-Adresse des Servers, der den Client konfiguriert hat.
  Damit soll u.a. ermöglicht werden, die korrekte Funktion der Server-Software zu überwachen.

- aus der Verwendung von Relay Agents:
  die Relay Agent-Adresse, über die die Konfigurationskommunikation gelaufen ist.
  Diese benötigt man evtl. zum Nachvollziehen der Wege, über die die Kommunikation zwischen Server und Client gelaufen ist

- Beginn und Dauer der Lease

- aus dem Einsatz von BOOTP:
  Vermerke, daß bestimmte Clients über BOOTP konfiguriert wurden.

- aus dem Einsatz von Renumbering (s. 3.7.):
  Post-Renumbering-Adressen der Clients

- aus dem Einsatz von Renumbering:
  für die betroffenen Clients ein Vermerk, daß der Client zum Renumbering aufgefordert wurde.
  Dies ist am zweckmäßigsten durch die Angabe des MAGIC-COOKIE-Codes zu realisieren, mit dem die Renumber-Kommunikation zwischen Server und Client eindeutig gekennzeichnet wird.

- Aus dem Einsatz von FQDN-Namen aus Dynamic DNS (s.3.6.) statt IP-Adressen:
  der FQDN-Name des Clients

- aus der Verwendung von DynDNS (s.3.6.):
  Meldungen bezüglich der Resource Records

- bei Systemen, die sich per DHCPINFORM konfigurieren lassen:
  deren Kennzeichnung
  (IP-Adresse, Hardwareadresse, etc.)

- Falls Hardwareadressen nicht erkannt werden konnten: die diese Systeme spezifizierenden Informationen

- bei Anzeichen von Ressourcenmißbrauch:
  die Identität des betreffenden Clients
  (IP-Adresse, Hardwareadresse, Subnetzadresse)

Abhängig von der Management-Policy der Netzbetreiber kann man sich vorstellen, daß auch Parameter über das Verhalten der DHCP-Clients im Netz kontrolliert werden sollen.

Verhalten der Clients im Netz

Um das Verhalten der DHCP-Clients im Netz nachzuvollziehen, muß man sich fragen, welche Aktionen für das Management von Interesse sind. Dies sind zum Beispiel der Eintritt des Clients in das Netz, das Verlassen des Netzes oder der Versuch der Verlängerung eines Leases. Unter dem Gesichtspunkt des Einsatzes mehrerer DHCP-Server in einem Netzbereich ist es eventuell auch von Interesse, die einzelnen Server zu identifizieren, welche Konfigurationen vorgenommen haben. Die Entscheidung, diese Identifikation zu verwirklichen, hängt aber vom Aufbau des Netzes ab.

Systematisch aufgelistet ergeben sich folgende interessante Parameter für jeden Client:

- Für den Eintritt eines Hostsystems in das Netz:
  Einträge über die DHCPDISCOVER-PDUs, die den Server schon erreicht haben.

- kennzeichnend für die erfolgreiche Konfiguration eines Clients:
  Einträge über die DHCPACKs, die an ihn abgesandt wurden mit dem Zeitpunkt der Versendung.
  Die erfolgreiche Konfiguration läßt sich nomalerweise nicht nachweisen, da der Server dem Client das DHCPACK schickt, ohne eine Rückantwort zu erhalten. D.h., wenn z.B. die Netzverbindung zum Client den Transport des DHCPACKS zum Client verhindert hat, erfährt der Server nichts davon. Das heißt, wenn keine Kontrolle des Erfolgs

der Konfiguration des Clients eingeführt wird, dann ist für den Server eine IP-Adresse evtl. als vergeben gekennzeichnet, obwohl sie nicht durch den Client verwendet wird.

- für das Verlassen des Netzes:
  Einträge über die DHCPRELEASEs des Clients mit der Zeit ihrer Versendung.

- Für Systeme, die eine externe Adresse haben und sich nur Konfigurationsdaten holen:
  Einträge über deren DHCPINFORM-PDUs plus der Zeit der Versendung.

- Für Systeme, die sich unberechtigt Lease erschleichen wollen:
  Einträge über die PDUs, anhand derer der Server den Mißstand festgestellt hat (DHCPDECLINEs, DHCPRELEASEs, etc.)

Falls auch DHCPv6 (s. Kapitel 4.) eingesetzt wird, interessieren auch:

- Für den Eintritt eines Hostsystems:
  Einträge über die DHCP Solicits, die der Server erhalten hat.

- für die erfolgreiche Konfiguration eines Clients:
  Einträge über die DHCP Replys.

- für das Verlassen des Netzes:
  Einträge über die DHCP Releases der Clients.

- falls der Server Clients zur Neukonfiguration / zum Renumbering aufgefordert hat:
  Einträge über die DHCP Reconfigure-PDUs, die an die Clients geschickt wurden.

Basierend auf diesen Parametern kann man sich vorstellen, daß der oder die Netzbetreiber den Verkehrsfluß auf dem Netz oder die Verweildauer bestimmter Clients im Netz überwachen wollen, um die Qualität der angebotenen Netzdienste oder die Nutzung des Konfigurationsangebots zu überprüfen. Das heißt, die Netzbetreiber benötigen evtl. eine Art Netzstatistik. Im folgenden soll untersucht werden, wie sich diese zusammensetzen kann.

5.2.3 Netzstatistik

Um einen Überblick über die Situation im DHCP-Subnetz bezüglich des Netzverkehrs zu erhalten, muß man überlegen, welche Daten vom DHCP-Server gesammelt werden können. Das wird hauptsächlich durch die Frage festgelegt, welche Daten dem DHCP-Server zugänglich sind. Die dem DHCP-Server zugänglichen Daten ergeben sich hauptsächlich aus seiner Konfigurationskommunikation mit den DHCP-Clients, d. h. , der Server kann diese Daten nur den PDUs entnehmen, die zwischen ihm und dem Client hin- und herlaufen. Dies wiederum bedeutet, daß diese in Bezug auf ihre Zusammensetzung, auf die Zeit ihrer Versendung etc. analysierbar zu machen sind. Für diese Analyse genügt es aber, die Daten zu halten, um im Fehlerfalle dem Management die Fehlersuche zu erleichtern; eine Analyse durch den Server erscheint nicht sinnvoll, da auf diese Weise ein großer Teil der Rechenkapazität, die dem DHCP-Server zur Verfügung steht, für eventuell unnötige Operationen verschwendet würde.

- Anzahl der empfangenen DHCPDISCOVER-, DHCPREQUEST-, DHCPDECLINE-, DHCPINFORM- DHCPRELEASE-PDUs sowie BOOTREQUESTs, die der Server empfangen hat
- Anzahl der gesendeten DHCPOFFERs, DHCPACKS und DHCPNAKs und BOOTREPLYs
- Anzahl der fehlerhaft empfangenen PDUs
- eventuell zu den korrekt empfangenen bzw. gesendeten PDUs den Zeitpunkt der Versendung

5.3 Anforderungen an die DHCP-Server

Nachdem unter 5.2 schon auf die gewünschte Datenhaltung durch die DHCP-Server eingegangen wurde und es für ein sinnvolles Management keine Daten gibt, die einzig und allein beim Server gehalten werden, muß nun geklärt werden, welche weiteren Anforderungen an die DHCP-Server gestellt werden.
Für die Kommunikation mit der Management-Anwendung muß beim DHCP-Server eine Schnittstelle implementiert werden, über die der Server mit der Management-Anwendung kommunizieren kann. Wenn ein DHCP-Server für

das Management wichtige Aktionen durchführt wie das Versenden eines DHCPACKs, um einen DHCP Client zu konfigurieren, muß er eine Meldung an die Managementanwendung absetzen, damit dieser die Möglichkeit gegeben wird, die Datenhaltung in der virtuellen Datenbasis zu modifizieren und gegebenenfalls Aktionen anzustoßen.
Ausgangssituationen für für das Management wichtige Aktionen des DHCP-Servers sind zum Beispiel:

- ein Client sendet ein DHCPDECLINE an den Server
  Das heißt, die dem zugeteilte Adresse wurde schon benutzt (s.2.3).

- ein Client ist länger als die vorgeschriebene Leasezeit im Netz
  Dies kann der Server anhand seiner Daten über die von ihm betreuten Clients herausfinden.

- ein nicht zugangsberechtigtes Client-System versucht, sich vom DHCP-Server konfigurieren zu lassen
  Dies kann der Server z.B. an einem falschen Authentifikationscode der DHCP-PDU (vgl. 2.4) erkennen.

- der Server erkennt, daß ein Client illegal (d.h., indem er gegen die Protokollvorschriften von DHCP verstößt) eine IP-Adresse benützt (s.a. 3.4.5). *Dies kann der Server erkennen, indem er von Clients freigegebene bzw. abgelehnte Adressen überprüft, ob sie wirklich freigegeben wurden bzw. ob sie wirklich schon gegen sein Wissen benutzt wurden, was bei einer konsistenten Datenhaltung durch Management-Anwendung und Server unmöglich sein sollte.*

Weitere Situationen, auf die der DHCP-Server reagieren muß, lassen sich aus dem Auftreten von Fehlern ableiten. Deshalb wird jetzt die Behandlung von Fehlern durch den DHCP-Servern untersucht.

5.3.1 Fehlerbehandlung durch den DHCP-Server

Im Rahmen der Arbeit eines DHCP-Servers kann es zu verschiedenen Fehlersituationen kommen. Sofern diese Fehler von Problemen der Netzhardware wie z.B. nicht funktionierenden Routern herrühren, sind sie jedoch nicht für das DHCP-Management relevant, sondern müssen im Rahmen bereits bestehender Managementkonzepte für diese Zwecke gehandhabt werden.

Fehler in der Netzkommunikation

Sollte es bei der Übertragung von DHCP-PDUs zu Fehlern kommen, sodaß PDUs nicht übertragen werden, so wird das entstehende Problem schon durch das DHCP-Protokoll selbst behoben. Diese aus dem Protokoll resultierende Fähigkeit der DHCP-Server rührt von der Benutzung von UDP als Übertragungsprotokoll her (s. 2.3),([Dro93], [Dro96b]), was eventuell zur Nichtübertragung von PDUs führen kann. Sollten DHCP-PDUs fehlerhaft übertragen werden, können sie nicht vom DHCP-Server verarbeitet werden; evtl. kann der Server sie für die Netzstatistik speichern.
In diesem Falle könnte der Server dem Management eine Mitteilung zukommen lassen, daß ein Fehler aufgetreten ist. Ob diese Option ausgeführt wird oder nicht, ist allerdings Management-Policy-abhängig.

Fehler im Programmablauf des Servers

Sollte es beim DHCP-Server zu Fehlern im Programmablauf kommen, muß die Serversoftware neu gestartet werden. Wenn zu diesem Zeitpunkt DHCP-Clients in einem Kommunikationsdialog mit dem DHCP-Server stehen, so wird ebenfalls durch die Befolgung der DHCP-Vorschriften garantiert, daß die DHCP-Clients bedient werden, falls mehrere DHCP-Server für die Clients erreichbar sind. Denn egal in welchem Zustand (s. Abbildung 2.1) die betreffenden Clients sich befinden, sorgt DHCP durch seine Timeout- bzw. Wiederholalgorithmen (s.2.3) dafür, daß die Clients ggf. nach einer definierten Zeit ihre Versuche, sich von dem inaktiven Server konfigurieren zu lassen, aufgeben und in den INITIALIZE-Zustand (s. Abbildung 2.1) zurückkehren, von wo sie andere DHCP-Server erreichen können. Wenn der abgestürzte DHCP-Server der einzig verfügbare war oder alle anderen auch nicht erreichbar sind, so müssen die DHCP-Clients so lange warten, bis die Server-Software wieder läuft. Um Situationen vorzubeugen, daß z.B. DHCP-Server längere Zeit nicht für DHCP-Clients erreichbar sind, sollte die Management-Anwendung die Server periodisch ansprechen, um ihre korrekte Funktion zu überprüfen.

Ein Problem für die Datenhaltung tritt auf, falls ein Client kurz vor dem Absturz der Serversoftware konfiguriert worden ist und schon seine neue IP-Adresse benutzt, aber der Server nicht mehr dazu gekommen ist, die erfolgreiche Konfiguration an die Management-Anwendung weiterzumelden. In diesem Falle befindet sich ein konfigurierter Client im Netz, von dem das Management nichts weiß, so daß andere Server versuchen können, die schon benutzte IP-Adresse anderen Clients zu vermitteln. Dieses Problem läßt sich aber vermeiden, indem der DHCP-Server oder der Client eine IP-Adresse vor dem Eintritt in die SELECT- bzw. BOUND-Phase auf etwaige Benutzung überprüfen. Diese Lösung ist auch als Soll-/(SHOULD)-Bestimmung im DHCP-Protokoll vorgesehen [Dro93], [Dro96b]. Um die Meldung des Clients zu gewährleisten, sollten die DHCP-Server jederzeit eine lokale Datenbasis über konfigurierte Clients halten, damit sie aus der geschilderten Problemsituation heraus nach einem Neustart diese Meldung jederzeit nachholen können.

5.4 Zusammenfassung

Bei den Untersuchungen der verschiedenen Managementanforderungen hat sich herausgestellt:
Für ein Management von DHCP-Netzumgebungen benötigt man eine Management-Anwendung, welche das Management zentral steuert. Ebenso muß man die DHCP-Serversoftware über Managementschnittstellen in die Lage versetzen, mit dieser Anwendung zu kommunizieren. Beide gemeinsam müssen Daten über die Netztopologie halten; diese kann man in Grundkonfigurations- und Daten über die aktuelle Netzsituation einteilen, evtl. kann als dritte Gruppe auch eine Netzstatistik erstellt werden. Mit wenigen Ausnahmen müssen diese Daten von Anwendung und DHCP-Servern gehalten werden; die Management-Anwendung soll dabei als Informationsquelle für die DHCP-Server in punkto Netzkonfiguration dienen. Die DHCP-Server wiederum sollen die Management-Anwendung über Änderungen in der Netzumgebung wie neu erschienene DHCP-Clients informieren. Sollten Fehler bei der Konfiguration der Clients auftreten, werden diese entweder durch strikte Befolgung des DHCP-Protokolls behoben, welche von den Netzbetreibern durchgesetzt werden muß, oder man kann sie durch geeignete Maßnahmen wie das Absetzen einer Meldung an die Management-Anwendung oder das Speichern von Informationen über Fehler dem Management zugänglich machen.

Kapitel 6

Internet-Management von DHCP-Architekturen

6.1 Einleitung

In den bisherigen Kapiteln wurde das DHCP-Protokoll vorgestellt, unter welchen Szenarien es eingesetzt werden kann und welche Anforderungen an ein DHCP-Management gestellt werden. Nun soll näher darauf eingegangen werden, wie sich das Management unter DHCP konfigurierter Umgebungen gestalten soll.

6.2 SNMP als Managementgrundlage

6.2.1 Motivation

Zunächst stellt sich dabei die Frage, welche Management-Architektur verwendet werden soll. Dabei kommen zwei der heute im Datenkommunikationsbereich gegebenen Alternativen in Frage: OSI- und SNMP-Management. Die erste Alternative, das OSI-Management, wird bis heute nur von wenigen Netzbetreibern verwendet, da es zwar theoretisch sehr elegant zu handhaben ist, aber in der Praxis in der Implementierung einen hohen Aufwand erfordert.

Die zweite Alternative, das *Simple Network Management Protocol*

(SNMP) [CFSD90],[RM90], wird in den meisten Netzumgebungen eingesetzt, so daß es als das Standardprotokoll für Datenkommunikationsmanagement angesehen werden kann. Wegen seiner weiten Verbreitung und seiner guten Handhabbarkeit wurde es für das Management von DHCP-Umgebungen ausgewählt.

6.2.2 Beschreibung des SNMP-Managements

Dabei muß man etwas näher auf das Management unter SNMP eingehen: Eine sogenannte *Managementplattform* ist der zentrale Kern einer SNMP-Management-Umgebung. Hier soll der Netzadministrator mit Hilfe der Management-Anwendung das Netz überwachen. Über eine Benutzerschnittstelle wird dem Netzadministrator die Möglichkeit gegeben, die aktuelle Situation im Netz zu überprüfen bzw. Änderungen anzustoßen.

Die *Managed Objects* generell sind dabei eine Art Repräsentation der wirklichen, physikalischen Netzkomponenten. Ein Managed Object charakterisiert dabei eine Netzkomponente durch zur Komponente gehörende Parameter, die *Objektattribute* [Mar91],[Fei95],[JS93]. Auch die DHCP-Server und -Clients in obigem Diagramm sind als MOs (Abkürzung für Managed Objects) zu sehen.

Der *SNMP-Hauptagent* ist einer von vielen SNMP-Agenten, die als Verwalter für diverse Unternetze, die aber nicht unbedingt organisatorisch oder physikalisch getrennt sein müssen, den Kontakt zur Management-Plattform aufrechterhalten. Die Ursache für den Einsatz dieser Agenten liegt in der Tatsache, daß unter Unix auf einem Rechner nur ein einziger Prozeß über eine Schnittstelle, den sogenannten *Port*, verfügen darf. Da nun die vielen MOs für den Austausch von Managementinformationen auf den Kontakt mit der Management-Plattform angewiesen sind, übernimmt ein Prozeß, der Hauptagent, diese Aufgabe. Der Hauptagent seinerseits stellt Schnittstellen zur Verfügung, über die er mit den *Subagenten* kommuniziert. Die Subagenten sind die anderen MOs in seinem Zuständigkeitsbereich. Als Protokoll für die Kommunikation zwischen Haupt- und Subagenten wird *SNMP DPI* [WCC+94] verwendet. DPI wird vor allem deshalb verwendet, weil es eine Standardschnittstelle zwischen Hauptagent und MO darstellt.

6.2.3 DHCP-Server als Subagenten

Der DHCP-Server (bzw. die DHCP-Server) übernimmt beim DHCP-Managment die Rolle des Subagenten und kommuniziert über eine noch zu erstellende DPI-Schnittstelle mit dem Hauptagenten. Für diese Kommunikation empfängt er PDUs, die Informationen und Befehle weiterleiten und setzt PDUs ab, die Informationen über seinen Zustand weitergeben. Das heißt, der DHCP-Server muß über die von ihm konfigurierten Hosts Rechenschaft ablegen können bzw. Befehle der Management-Plattform erfüllen, indem er bestimmte Aktionen anstößt.

6.2.4 Kommunikation unter SNMP

Interessant ist auch die Frage, wie sich der Datenverkehr für das SNMP-Management einer mit DHCP konfigurierten Umgebung gestaltet. Bei SNMP werden sämtliche Operationen auf MOs auf das Auslesen oder Verändern von Variablen der MOs zurückgeführt. Das bedeutet, daß die Management-Anwendung mit dem Agenten im wesentlichen Operationen zum Lesen von Variablenwerten (*GET*) und zum Verändern dieser Werte (*SET*) austauscht. Eine Ausnahme bilden die sogenannten *Traps*. Traps sind in SNMP spontane Ereignismeldungen (*Event Notifications*), die von einem Agenten (nicht von einem Objekt!) an eine Managementplattform geschickt werden. Dieses Verfahren hat zwei Vorteile ([JS93]) :

- für einen Agenten müssen nur zwei Grundfunktionen implementiert werden: GET und SET
- die aufwendige Implementierung eines imperativen Managementkommandos (wie M-ACTION im OSI-Management) wird vermieden

Die Operationen bestehen in der Regel aus zwei Phasen, der Anforderung (request) und der Rückantwort (response). In der Anforderungsphase wird der Kommunikationspartner (z.B. der Hauptagent) aufgefordert, eine Operation auszuführen. Dieser reagiert mit einer Rückantwort, in der er die Operation bestätigt oder zurückweist. In einigen Fällen (z. B. bei GET-Operationen) enthält die Rückantwort auch die Ergebnisse der Operation.

Grundsätzlich existieren zwei verschiedene Verfahren, um den Zustand eines MOs kontinuierlich zu überwachen: Polling und spontane Meldungen.

Beim Polling werden die Werte von Variablen periodisch abgefragt, wobei die Abfragezyklen entsprechend der Wichtigkeit der Variablen unterschiedlich lang gehalten werden. Im anderen Fall sendet der Agent bei Auftreten definierter Ereignisse eine spontane Meldung an die Managementplattform, in welcher die Art des Ereignisses mitgeteilt wird. (In unserem Falle z.B. das Auftauchen eines DHCP-Clients)

Beide Verfahren haben ihre Nachteile: Polling belastet das Netz mit Management-Verkehr, die dem eigentlichen Datenverkehr nicht mehr zur Verfügung steht. Es ist leicht möglich, mit falsch gesetzten Polling-Intervallen ein ganzes Netz in die Knie zu zwingen [JS93].

Das Versenden von Traps dagegen erfordert im Agenten eine gewisse Grundintelligenz, die verhindert, daß bei nichtigen Anlässen ganze Meldungsstürme durch das Netz rauschen, ebenfalls Übertragungskapazität beanspruchen und die Management-Anwendung stark beanspruchen [JS93].

Im praktischen Betrieb hat sich herausgestellt, daß das Optimum in einer sinnvollen Kombination von Polling und spontanen Meldungen besteht, wobei das Mischungsverhältnis sowie die Intelligenz der Agenten auch von den Betriebserfordernissen, also der Policy des Betreibers abhängt [JS93]. Welche Strategie im Endeffekt eingesetzt wird, hängt wieder stark von der Netzumgebung und der Management-Policy des Netzbetreibers ab, so daß hier keine Festlegung erfolgen soll. In einem Netz mit viel Übertragungskapazität und wenig Netzlast kann man das Gewicht eher zu Gunsten des Pollings verlagern als in einem Netz mit vielen Teilnehmern, wenig Übertragungskapazität der Netzverbindungen und viel Netzlast.

6.2.5 Die Management Information Base (MIB) bei SNMP

Ebenso wie das OSI-Modell ([JS93]) definiert auch SNMP eine virtuelle Datenbasis, in der die Objektklassen und Instanzen aller Managed Objects vorgegeben werden können. Diese Datenbasis wird wie bei OSI MIB genannt und wird im RFC 1213 definiert [RM91],[Mar91],[Fei95],[JS93].
Die bisherige Arbeit zielt unter anderem darauf ab, eine solche MIB zu kon-

struieren, die auch die DHCP-Gegebenheiten berücksichtigt.
Die Frage ist nun, welche Anforderungen an die Datenhaltung bzw. das Verhalten des DHCP-Servers gestellt werden. Es wird zunächst zur Einführung davon ausgegangen, daß die Management-Anwendung als Zentrum für das Management sämtliche Daten für die virtuelle Datenbasis hält und Event Notifications von den DHCP-Servern verarbeitet wie z.B. die erfolgreiche Konfiguration eines neuen Clients. Die Server wiederum halten lokal die Daten über die aktuelle Netzkonfiguration, welche sie für die Verteilung von IP-Adressen und die Konfiguration der von ihnen betreuten Subnetze benötigen. Diese Daten erhalten die DHCP-Server über SET-Operationen, die die Management-Anwendung absetzt.

6.3 Realisierung des DHCP-Managements

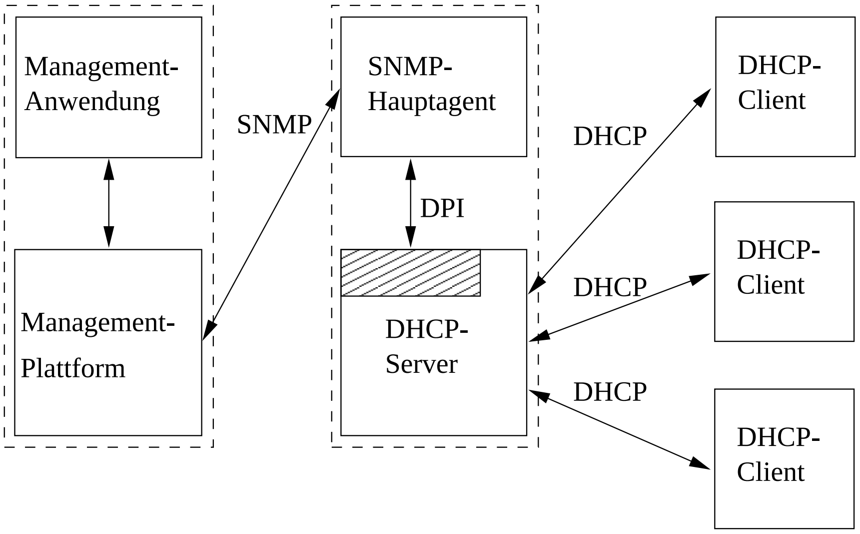

Abbildung 6.1: Managementumgebung für DHCP-konfigurierte Systeme

Wie im vorhergehenden Kapitel angesprochen, gibt es einige Anforderungen an das Management von DHCP-Umgebungen. Eine Managementanwendung soll zentral die Situation im Netz überwachen und auf der einen Seite Daten halten über die Netzkonfiguration, auf der anderen Seite auf Veränderungen im Netz reagieren wie die Konfiguration eines neuen DHCP-Clients durch einen DHCP-Server. Es wurde überlegt, daß man bei der Datenhaltung zwischen Daten der Grundkonfiguration, welche als Basis für die Konfiguration von Clients dienen, Daten über die aktuelle Netzsituation sowie Daten für die Netzstatistik unterscheidet.

Jetzt soll ein Konzept zur Verwirklichung dieser Vorstellungen und Überlegungen vorgestellt werden. Dabei wird zunächst auf die Management-Anwendung, dann auf die DHCP-Server eingegangen.

6.3.1 Management-Anwendung

Die Management-Anwendung wird als Zentrale für das Management der DHCP-Umgebung gesehen. Wie schon in den vorhergehenden Kapiteln erläutert,soll sie auf der einen Seite die Daten über die Grundkonfiguration, die aktuelle Netzsituation sowie der Netzstatistik halten, auf der anderen Seite soll sie Trap-Meldungen über neu konfigurierte DHCP-Clients verarbeiten und Meldungen absetzen, welche es den DHCP-Servern ermöglichen, ohne Konsistenzprobleme bei der Adreßvergabe zu arbeiten. Über eine Benutzerschnittstelle wird dem Netzadministrator die Möglichkeit gegeben, die aktuelle Situation im Netz zu überprüfen bzw. Änderungen anzustoßen.

Im folgenden wird zuerst eine Beschreibung der Komponenten der Anwendung abgegeben werden, bevor ihr Verhalten im Betrieb des Netzes vorgestellt wird, wenn mit DHCP konfiguriert wird.

Management-Datenbasis

Die Datenbasis (s.a. Abb. 6.2) enthält die Konfigurationsinformationen über die mit DHCP konfigurierten Systeme (Grundkonfigurationsdaten, Daten über die aktuelle Netzsituation, Netzstatistik). Nachdem im vorhergehenden Kapitel schon ausführlich auf diese Daten eingegangen wurde, sollen sie hier nicht weiter behandelt werden.

SNMP-Schnittstelle

Um die Kommunikation mit den SNMP-Subagenten zu ermöglichen, muß die Management-Anwendung über eine SNMP-Schnittstelle verfügen. Über

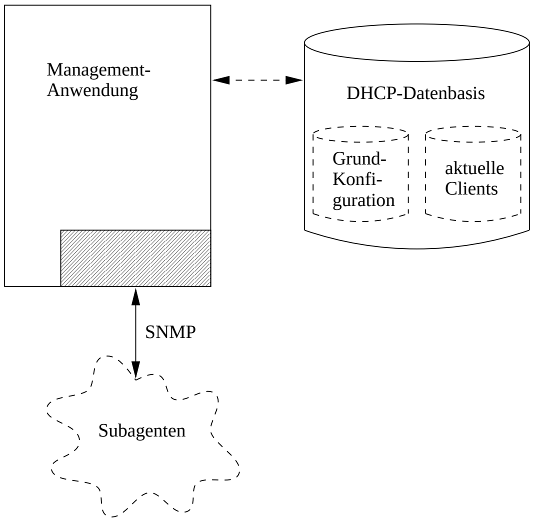

Abbildung 6.2: Management-Anwendung mit Komponenten

diese sollen dann die Meldungen (Traps) von den DHCP-Clients empfangen werden und weiter die SET- und GET-Operationen zur Beeinflussung des Netzes abgesetzt werden.

6.3.2 Verhalten von Management-Anwendung und DHCP-Servern in verschiedenen Szenarien

Neben der Datenhaltung in der Datenbasis wird für ein sinnvolles Management an die Managementanwendung auch die Anforderung gestellt, auf verschiedene Szenarien flexibel reagieren können. Mögliche Szenarien sind zum Beispiel, daß ein DHCP-Client erfolgreich konfiguriert wurde oder daß ein Client eine angebotene IP-Adresse ablehnt, weil sie schon benutzt wird (s. 3.4). Um das Verhalten der Management-Anwendung so genau wie möglich zu beschreiben, wird hier ein Vorschlag abgegeben, wie sie in bestimmten Situationen zu reagieren hat. Dabei wird gleichzeitig das erforderliche Verhalten der DHCP-Serversoftware erläutert, da es aufgrund der Verzahnung der Abläufe in den Szenarien keinen Sinn machen würde, dieses getrennt zu betrachten.

DHCP-Client wurde erfolgreich konfiguriert

Wenn ein DHCP-Server einem DHCP-Client eine DHCPACK-PDU geschickt hat und ihn damit konfiguriert hat, schickt er einen Trap an die Management-Anwendung, in dem die zum Update der Aktuelle-Situation-Daten in der Datenbasis erforderlichen Parameter wie die zugeteilte IP-Adresse, die Hardwareadresse des Clients etc. (s.5.2.2) enthalten sind. Die Management-Anwendung sollte nun eine (vom Admisinstrator festzulegende) gewisse Zeitspanne warten, bevor sie die neuen Parameter in die Datenbasis überträgt. Sobald diese Zeitspanne verstrichen ist, aktualisiert sie die Datenbasis und übermittelt den DHCP-Servern auf dem Weg über die Hauptagenten mit einer SET-Meldung die vergebene IP-Adresse, um Mehrfachzuteilungen zu vermeiden.

IP-Adresse schon vergeben

Im Falle, daß ein DHCP-Client von einem DHCP-Server ein DHCPACK erhalten hat (s. 3.4) und bei der Überprüfung der IP-Adresse feststellt, daß diese schon benutzt wird, sendet er ein DHCPDECLINE an den DHCP-Server und unternimmt einen neuen Konfigurationsversuch (s. 3.4). Mit dem DHCP-DECLINE informiert er den Server über die schon vergebene IP-Adresse. Der Server setzt hierauf eine Trap-Meldung an die Management-Anwendung ab, welche die Parameter enthält, welche den Client charakterisieren (Hardwareadresse etc.) plus die betroffene IP-Adresse.

Die Management-Anwendung empfängt die Trap-PDU und entnimmt ihr, welche IP-Adresse betroffen ist. Wenn diese auch nach den aktuellen Informationen in der Datenbasis als vergeben bekannt ist, kann sie die Meldung entweder verwerfen oder die Daten z.B. für die Netzstatistik speichern. Sollte die Adresse aber als nicht vergeben gekennzeichnet sein, sollte sie über die Benutzerschnittstelle dem Administrator eine Meldung über die Situation zukommen lassen, da evtl. ein Mißbrauch der IP-Adresse vorliegt (vgl. 3.4.5). Wie diese Meldung aussieht, hängt von der Implementierung der Benutzerschnittstelle ab und soll hier nicht weiter erläutert werden.

Lease wurde verlängert

Wenn ein DHCP-Client seinen Lease verlängert, d.h., die Erlaubnis, eine gewisse IP-Adresse weiter zu benutzen, dann hat er auf ein DHCPREQUEST (s. 3.4) ein DHCPACK vom Server erhalten. Bei dieser Gelegenheit sendet der DHCP-Server, nachdem er das DHCPACK abeschickt hat, einen Trap an die Managementanwendung. Dieser Trap muß nun im Prinzip nur noch die Daten enthalten, die die Management-Anwendung die Informationen ableiten lassen, daß der Lease des Clients verlängert wurde, also die IP-Adresse, Hardwareadresse und die Daten über die Leasezeit.
Die Managementanwendung muß nun ähnlich wie bei einer Neukonfiguration die Daten in der Datenbasis auf einen aktuellen Stand bringen. Sie muß hingegen nicht die in Frage kommenden DHCP-Server über die Änderungen informieren, da diese die betreffende IP-Adresse ja schon seit der Erstkonfiguration kennen.

Lease ist abgelaufen

Falls die Leasezeit eines DHCP-Clients abläuft, ohne daß der Client sich um eine Verlängerung bemüht oder der DHCP-Server wegen Netzproblemen von diesen Bemühungen nichts mitbekommt, kann man verschiedene Strategien zur Behandlung dieser Situation in Betracht ziehen: Entweder der DHCP-Server, der den Client konfiguriert hat, sendet einen Trap an die Management-Anwendung, daß der betreffende Client nicht mehr über die IP-Adresse verfügt, oder die Management-Anwendung überprüft selbst die Leasezeiten und trägt gegebenenfalls abgelaufene Leases aus der Datenbasis aus. Bei beiden Alternativen wird davon ausgegangen, daß DHCP-Clients nicht versuchen, länger als ihre maximale Leasezeit ihre IP-Adresse zu benutzen. Dies ist einerseits vom DHCP-Protokoll so vorgeschrieben, andererseits liegt eine Durchsetzung dieser Forderung im Interesse jedes Netzbetreibers, so daß man im Falle einer Nichteinhaltung einem Benutzer z.B. den weiteren Zugang zum Netz verwehren könnte.

Die erste Alternative des Trapversendens bringt erhöhte Netzlast mit sich; da die Daten über die Leasezeit der Clients ja in der Datenbasis vorhanden sind, kann die Managementanwendung gegebenenfalls die IP-Adresse noch einmal auf ihre Verwendung überprüfen und anschließend die Datenbasis aktualisieren. Sollte der Fall eintreten, daß eine IP-Adresse weiter in Gebrauch ist, obwohl die Leasezeit abgelaufen ist, so sollte die Management-Anwendung eine dementsprechende Meldung an die Netzadministration über die Benutzerschnittstelle abgeben, damit das Problem weiter behandelt wird.

Client hat Ressourcen freigegeben

Falls ein DHCP-Client seine IP-Adresse nicht mehr benötigt, kann er die von ihm genutzten Netzressourcen durch Versenden eines DHCPRELEASE an den DHCP-Server, von dem er konfiguriert wurde, freigeben (s. 3.4). Wenn nun ein DHCP-Server ein DHCPRELEASE empfängt, überprüft er, ob die Adresse wirklich freigegeben wurde (evtl. über ein Ping auf die Adresse). Ob die Adresse frei ist oder nicht, muß er der Management-Anwendung über einen Trap weitermelden. Die Management-Anwendung muß nun die spezifischen Dateneinträge über den freigegebenen Lease in der Datenbasis löschen bzw. den Netzadministrator über den Mißbrauch der Adresse informieren.

Client hat externe IP-Adresse

Wenn ein DHCP-Client eine externe IP-Adresse hat und nur Konfigurationsdaten für ein bestimmtes Subnetz benötigt, weil er die dort angebotenen Netzdienste nutzen will, so kann er mittels einer DHCPINFORM-PDU DHCP-Server die nötigen Parameter anfordern (s. 3.4).

Wie der DHCP-Server reagiert bzw. wie diese Anfrage behandelt wird, hängt von der Vergabe-Politik der Netzbetreiber ab: Wenn nicht gewünscht wird, daß bestimmte Netzdienste von bestimmten Benutzern genutzt werden (z.B. Angehörigen einer fremden Firma), so kann der Zugang beschränkt werden, indem man Parameter nur den DHCP-Clients zukommen läßt, die sich auf "genehmigten" Subnetzen befinden. Wenn der Zugriff aber erlaubt ist, so hängt es von der Management-Policy der Betreiber ab, ob die reine Vergabe von Konfigurationsparametern an externe Systeme ebenfalls überwacht werden soll oder nicht. Falls die Vergabe nicht überwacht wird, so genügt es, wenn der angesprochene DHCP-Server die Parameter in einem DHCPACK übermittelt. Falls eine Überwachung gewünscht wird, um z.B. die Nutzung von Netzressourcen kontrollieren zu können, so muß der DHCP-Server zuerst überprüfen, ob die IP-Adresse des Clients auch wirklich extern ist (vgl. 3.4.5), bevor er ein DHCPACK absendet und die Management-Anwendung informiert, an welchen Client die Parameter vergeben wurden bzw. werden sollten (IP-Adresse, Hardwareadresse, etc.). Wenn die Management-Anwendung den Trap mit der Information erhält, so muß sie die Datenbasis um den Eintrag des externen Systems erweitern bzw. im Falle des Mißbrauchs des DHCPINFORM den Netzadminstrator benachrichtigen.

Verhalten bei Sicherheitsproblemen

Im Rahmen des Einsatzes von sicherheitsfördernden Diensten wie der schon vorgestellten Authentifikation von DHCP-PDUs (s. 2.4) können sich Situationen ergeben, in denen der DHCP-Server feststellt, daß ein Mißbrauch der Netzressourcen wie IP-Adressen etc. vorliegt. In diesem Fall muß der DHCP-Server die Information über die Identität des mißbrauchenden Systems (IP-Adresse, Harwareadresse, etc.) der Management-Anwendung über eine Trap-Meldung zukommen lassen. Wie die Management-Anwendung auf diese Meldung reagiert, hängt stark von der eingesetzten Sicherheitsstrategie bzw. dem spezifischen Sicherheitsproblem ab und kann hier nicht erschöpfend behandelt werden.

6.3.3 DHCP-Server

Nachdem unter 6.3.2 schon auf das erforderliche Verhalten des DHCP-Servers in verschiedenen Szenarien eingegegangen wurde, muß nun noch ein Einblick in die erforderlichen Komponenten des DHCP-Servers gegeben werden. Außerdem muß für den Fall eines Neustarts bzw. Absturzes des DHCP-Servers eine Strategie angegeben werden, um Inkonsistenzen in der Datenhaltung zu vermeiden.

DPI-Schnittstelle

Diese Schnittstelle benötigt der DHCP-Server, um mit dem SNMP-Hauptagenten zu kommunizieren. Wie schon mehrfach angesprochen, empfängt der DHCP-Server über diese Schnitstelle GET- und SET-Meldungen, welche die Management-Anwendung angestoßen hat und versendet Traps über außergewöhnliche Ereignisse, wie sie schon im letzten Abschnitt unter 6.3.2 erwähnt wurden. Da diese Schnittstelle nicht Be-

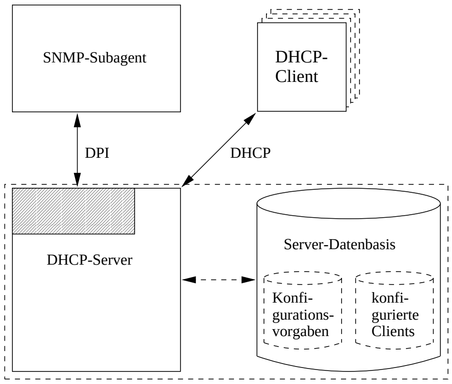

Abbildung 6.3: DHCP-Server im Managementszenario

standteil der aktuellen Implementierungen von DHCP-Software ist, muß sie für eine Verwirklichung des DHCP-Managements in die ausgewählte Server-Software integriert werden.

DHCP-Schnittstelle

Über diese Schnittstelle wird die Konfigurationskommunikation mit den DHCP-Clients abgewickelt. Diese Schnittstelle ist bei den DHCP-Server-Softwarevarianten standardmäßig enthalten.

Server-Datenbasis

In der Datenbasis des DHCP-Servers werden die Daten gehalten, welche der Server zur Konfiguration von IP-Systemen benötigt (freie IP-Adressen, Subnet Masks, Router, etc.), sowie die Daten über bestehende Leases, die vom Server berücksichtigt werden müssen wie bereits vergebene IP-Adressen.

Verhalten bei Neustart

Falls ein DHCP-Server-Programm neu gestartet werden muß (z.B. nach einem Absturz) , so muß es zunächst überprüfen, ob es vor Abbruch des Programms einen DHCP-Client konfiguriert hat und diesen nicht an die Management-Anwendung weitergemeldet hat. Man könnte dies z.B. dadurch realisieren, daß der Server nach jedem DHCPACK eine Datei lokal anlegt, welche die relevanten Daten des Client-Systems enthält und diese nach erfolgreicher Weitermeldung wieder löscht. Außerdem muß der Server dafür sorgen, daß er die Parameter über die aktuellen Konfigurationsvorgaben von der Management-Anwendung erhält (Grundkonfigurations- und Aktuelle-Situations-Daten, evtl. nur Aktuelle-Situations-Daten, falls sich die Grundkonfiguration des Netzes nicht ändert). Dies erreicht es durch das Versenden eines Traps, in dem es die erforderlichen Paramater erbittet. Aus diesem Trap muß hervorgehen, wo der DHCP-Server sitzt, damit die Management-Anwendung die für ihn bestimmten Parameter zusammenstellen kann, d.h., der Trap muß z.B. die IP-Adresse des Rechners, auf dem die Serversoftware läuft, enthalten.

Wenn die Managementanwendung diesen Trap empfängt, so stellt sie die Grundkonfigurations- sowie die Aktuelle-Situation-Daten zusammen, welche der DHCP-Server zur Betreuung "seiner" Subnetze benötigt und übermittelt sie dem DHCP-Server über ein SET. Um die Identifikation des DHCP-Servers und damit die Festlegung der von ihm zu betreuenden Subnetze oder auszuwählenden Konfigurationsparameter in der Datenbasis der Management-Anwendung zu ermöglichen, muß ein eindeutiger Identifikator für den spezifischen DHCP-Server gefunden werden. Da auf einem IP-System wegen der Belegung der Systemports maximal nur ein DHCP-Serverprogramm laufen kann, könnte man als ID für den Server die IP-Adresse des IP-Systems verwenden.

6.4 Zusammenfassung

In diesem Kapitel wurde aufgrund der im vorhergehenden Kapitel untersuchten Anforderungen an das DHCP-Management ein Vorschlag abgegeben, wie dieses Management realisiert werden kann. Dabei wurde entschieden, SNMP als Grundlage für die Managementarchitektur zu wählen. Weiterhin wurde aufgezeigt, wie die geforderte Management-Anwendung und die DHCP-Server in einer SNMP-Architektur zusammenarbeiten müssen, um ein sinnvolles Management von DHCP zu ermöglichen, sowie die Voraussetzungen, die von Servern und Management-Anwendung erfüllt werden müssen. Im folgenden Kapitel kann nun auf die Entwicklung einer SNMP-MIB für DHCP-Umgebungen eingegangen werden.

Kapitel 7

Entwicklung einer DHCP-MIB

7.1 Einleitung

Nachdem im vorhergehenden Kapitel ein Vorschlag gemacht wurde, wie das Management einer mit DHCP konfigurierten Netzumgebung aussehen kann, muß nun noch eine Festlegung getroffen werden, wie die relevanten Daten in einer Management Information Base (s. 6.2.5) vorzugeben sind.

7.2 Motivation für die DHCP-MIB

Wie schon unter 6.2.5 erwähnt, wird in einer MIB ein Modell angegeben, wie die Informationen über verwaltete Netzressourcen zu halten sind. Zu diesen Informationen gehören auch die Nachrichten, welche der Management-Anwendung die Managed Objects betreffend zugestellt werden. Da SNMP-Management auf allen möglichen Systemen und Plattformen betrieben wird, muß es einen Weg geben, die MIB-Information so zu gestalten, daß sie auch auf allen Plattformen ausgewertet werden kann. Dies wird unter SNMP durch Zuhilfenahme von *SMI*, einem Abkömmling von *ASN.1* erreicht. SMI steht dabei für Structure and Identification of Management Information, ASN.1 für Abstract Syntax Notation One.

7.2.1 Motivation für den Einsatz von ASN.1 / SMI

ASN.1 ist ein wichtiges Hilfsmittel der SNMP-Dokumente zur Bechreibung von Managed Objects und Protokollnachrichten [JS93]. Der Einsatz von Datenbeschreibungssprachen wie ASN.1 hat primär nichts mit Netzmanagement zu tun, sondern greift bereits viel früher, nämlich bei der allgemeinen Datenkommunikation.

Zwei Anwendungen, die über das Netz Daten austauschen, stellen deren Strukturen lokal oft sehr unterschiedlich dar, was sowohl an den verwendeten Programmiersprachen, als auch an der internen Datendarstellung der beteiligten Rechner liegen kann.

Ein allgemein anerkannter Ansatz zur Lösung dieses Problems liegt darin, eine formale Beschreibungssprache für Daten einzuführen, mit der die Struktur der auszutauschenden Daten auf beiden Seiten in der gleichen Form spezifiziert wird. Damit ist sichergestellt, daß beide Seiten eindeutig wissen, "was gemeint ist" .

Da über die "Leitung" zwischen den beteiligten Rechnern aber nur Byte- oder Bitfolgen gehen können, benötigt man Regeln, nach denen die formal beschriebenen Datenstrukturen "serialisiert" werden können. Und umgekehrt muß aus dem seriellen Datenstrom wieder die ursprüngliche Datenstruktur rekonstruiert werden. Diese Regeln nennt man *Kodierregeln (Encoding Rules)*.

Schreibt man nun auf jedem Rechner, der mit anderen Rechnern Daten austauschen soll, ein Kodiermodul, der einen von einer Anwendung übergebenen Datenblock unter Berücksichtigung der zugehörigen formalen Beschreibung in den "genormten" Datenstrom konvertiert und ebenso ein Dekodiermodul, das einen "genormten" Datenstrom erkennt und unter Berücksichtigung der zugehörigen formalen Beschreibung den richtigen Datenblock im Anwendungsformat erzeugt, dann kann - zumindest auf dieser Ebene - jeder Rechner mit jedem anderen Rechner Daten austauschen.

Diese Problematik wird im OSI-Schichtenmodell auf Ebene 6 (Presentation) behandelt. Die formale Beschreibung der Datenstrukturen wird *ab-*

stract syntax genannt, die Struktur des serialisierten Datenstroms *transfer syntax* und die spezifische Darstellung auf einem Rechner *local syntax*. Zur Beschreibung der abstrakten Syntax wird ASN.1 (Abstract Syntax Notation One) verwendet, die in der ISO-Norm 8824 beschrieben ist. Die Transfersyntax entspricht den sogenannten *Basic Encoding Rules* von ASN.1, die in der ISO-Norm 8825 enthalten sind. Bild 7.1 zeigt diese Zusammenhänge an einem einfachen Beispiel und für eine Übertragungsrichtung.

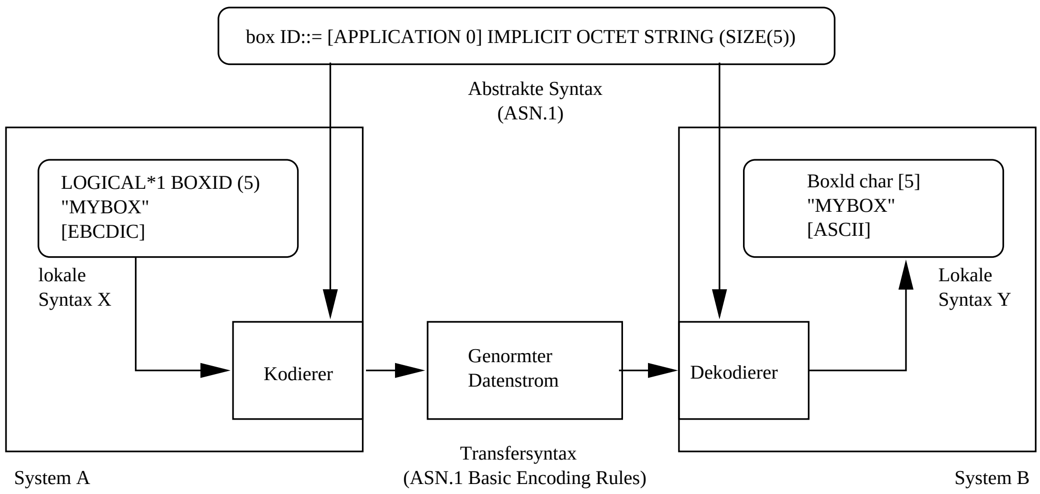

Abbildung 7.1: Beispiel für eine Kodierung mit ASN.1

(Nach [JS93])

In SNMP wird ein von ASN.1 abstammendes Protokoll namens SMI zur Beschreibung der Managed Objects und der Protokollnachrichten verwendet. Die Kodierung erfolgt nach den ASN.1 Basic Encoding Rules. Für ein tieferes und weitergehendes Verständnis von ASN.1 wird auf die Literatur zu diesem Thema verwiesen [ISOa], [ISOb], [Gor90]. SMI wird im RFC1155 definiert [RM90].

7.3 Ausführung der MIB

Nachdem nun die Grundlage für das Verständnis von ASN.1 als Mittel der Datenbeschreibung gelegt ist, soll nun auf die Ausführung der MIB für DHCP eingegangen werden. Diese soll als Modell für die Datenhaltung sowohl für die Management-Anwendung als auch für die DHCP-Server dienen. Das bedeutet, daß sie als Vorgabe für Daten der Grundkonfiguration, der aktuellen Situation im Netz und für die Nachrichten dienen soll, die zwischen Management-Anwendung und DHCP-Servern ausgetauscht werden.

Die genaue Ausführung dieser MIB entnehme der interessierte Leser dem Anhang. An dieser Stelle soll nun vor allem auf die Philosophie der Datenmodelle eingegangen werden.

Wie schon im vorhergehenden Kapitel erörtert, kann man die Daten in einer mit DHCP konfigurierten Netzumgebung in Grundkonfigurationsdaten und Daten über die aktuelle Situation im Netz einteilen. Beide Datenarten hängen stark mit dem Aufbau des Netzes ab, in dem sie eingesetzt werden. Dies bedeutet zum Beispiel, daß für ein "kleines" Netz mit einem DHCP-Server und einigen DHCP-Clients die Datenhaltung sich deutlich unterscheidet von einem "großen" Netz mit vielen Subnetzen, auf denen jeweils mindestens ein DHCP-Server installiert ist, welcher unterschiedlich viele DHCP-Clients bedient.

Um eine sinnvolle, d.h. effiziente Datenhaltung ausgehend von der MIB zu ermöglichen, muß man sich also überlegen, wie die Informationen in der MIB strukturiert sein müssen, um möglichst vielen Anforderungen in unterschiedlichsten Einsatzszenarien gerecht zu werden. Diese Frage soll nun näher untersucht werden.

7.3.1 Datenmodellierung

Bei der Modellierung der Datenvorgaben der MIB muß man zwischen beliebig vielen Alternativen auswählen, wie diese Modellierung bei der Implementierung umgesetzt werden kann, um diese Umsetzung bei der Erstellung der MIB zu berücksichtigen. Als naheliegendes Beispiel für die Art und Weise, wie z.B. die Ausgangsdaten für die Konfiguration von Subnetze modelliert werden können, könnte man die Daten für bestimmte Subnetze in Form einer Tabelle abspeichern, bei der unter dem Namen / der Adresse des Subnetzes alle Parameter für dieses Subnetz wie die Adressen von Routern, Relay Agents, Subnet Mask, etc. aufgeführt werden. Ebenso naheliegend könnte man einen objektorientierten Ansatz wählen, bei dem alle Konfigurationsparameter einmal für alle Subnetze festgelegt werden und für jedes Subnetz nur die Parameter neu definiert werden, welche in diesem Subnetz andere Werte besitzen, z.B. eine andere Line Printer Adresse etc. . Gerade bei Netzen mit einer hohen Anzahl von unterschiedlich zu konfigurierenden Subnetzen ergibt sich durch den objektorientierten Ansatz eine große Platzersparnis im Vergleich zu der tabellenartigen Alternative. Andererseits ist es bei der Tabelle einfacher, Parameter auszulesen und weiterzuverarbeiten, da sich die Parameter in einem Stück behandeln lassen. Wegen der hohen Platzersparnis in "großen" Netzen und der starken Vereinfachung von Änderungen beim objektorientierten Ansatz (z.B. beim Renumbering müssen nur wenige Parameter geändert werden) sollte der objektorientierte Ansatz weiter verfolgt werden. Dieser Ansatz wird also bei der Erstellung der MIB verstärkt berücksichtigt.

7.3.2 Struktur der MIB

Angelehnt an die Unterteilung der Daten aus Kapitel 5 wird nun die DHCP-MIB modelliert. Nachdem SMI nur eingeschränkte Möglichkeiten der Datenmodellierung liefert, kann man eine Gruppe von Parametern entweder als Liste oder als Tabelle zusammenfassen. Damit ergibt sich die folgende Struktur der MIB-Informationen:

- Grundkonfigurationseinträge
- Einträge für die aktuelle Situation
- Netzstatistik

Diese sollen kurz weiter ausgeführt werden:

Grundkonfigurationseinträge

Aus der angestrebten objektorientierten Datenhaltung ergibt sich, daß einerseits die Angabe einer Standardkonfiguration des Netzes ermöglicht werden muß, andererseits die Angabe der Konfiguration von spezifischen Subnetzen, bei der nur die sich unterscheidenden Parameter angegeben werden sollen. Als Vorgabe wird deshalb eine Liste mit Konfigurationsparametern definiert, an deren Beginn eine sogenannte *Konfigurationsregion* angegeben wird, in

welcher diese Parameter gelten sollen. Diese Region könnte beispielsweise ein Stockwerk eines Gebäudes sein, in welchem ein bestimmter Drucker steht etc. . Mit am einfachsten läßt sich diese Region durch einen Integerwert kennzeichnen, der die Nummer der Region bezeichnet.
Außer den DHCP Options nach RFC1533 werden hier auch Parameter berücksichtigt, die für Renumbering oder die Authentifizierung von DHCP-Nachrichten benötigt werden sowie die in dem Subnetz zur Vergabe vorgesehenen IP-Adressen. Es wird davon ausgegangen, daß diese Vorgaben auf der einen Seite der Managment-Anwendung zur Mitteilung der Konfigurationsdaten dienen und auf der anderen Seite den DHCP-Servern, um die Konfigurationsvorgaben für die von ihnen betreuten Subnetze lokal zu halten bzw. um die DHCP-Clients zu konfigurieren.

Einträge für die aktuelle Situation

Für eine Übersicht über die aktuelle Situation im Netz sollen folgende Daten gehalten werden können:

- die aktiven und inaktiven DHCP-Server
- eine Liste der aktuell vergebenen Leases mit den die Leases charakterisierenden Informationen, u.a. auch der FQDN des Clients; aus diesem Eintrag soll man auch die Informationen über vergebene IP-Adressen, "mißbrauchte" IP-Adressen etc. ziehen können
- eine Liste aller Subnetze mit den Informationen, welche für den Zugriff auf die Netze erforderlich sind
- eine Liste aller Hardwareadressen/Clients, welche nicht erkannt werden konnten mit den zugehörigen Parametern
- eine Liste der externen Clients, die über DHCPINFORM konfiguriert wurden
- Informationen für Clients, welche ein Renumbering durchlaufen

Alle Daten werden wieder unter der Nummer einer Region gehalten. Sinnvollerweise soll die Management-Anwendung alle angeführten Daten halten und die DHCP-Server nur die Daten, welche für ihre Arbeit notwendig sind.

Netzstatistik

Als Ausgangsbasis für eine Netzstatistik sollen folgende Daten gehalten werden:

- die Zeit, welche die Serversoftware läuft
- die Version der Serversoftware
- der letzte Zugriff auf die Server-Datenbasis
- die Zeit, die für die Freigabe von IP-Adressen aus ausgelaufenen Leases veranschlagt wird
- die Anzahlen der von den Servern empfangenen bzw. abgesendeten DHCP-PDUs
- die empfangenen/gesendeten BootRequests/ Replys
- die Zahl der aktiven und abgelaufenen Leases
- die Anzahl der in einem Subnetz aktiven Server
- die Anzahl der fehlerhaft empfangenen PDUs
- das Ausmaš des festgesetllten Mišbrauchs von IP-Adressen

7.4 Zusammenfassung

In diesem Kapitel wurde erläutert, wie die Daten in einer DHCP-MIB modelliert werden sollen. Dazu wurde zunächst der Einsatz von SMI zur Beschreibung der MIB-Objekte motiviert und anschließend aufbauend auf den erarbeiteten Managementanforderungen die Objekte festgelegt. Die genaue Ausführung der MIB ist dem Anhang A zu entnehmen.

Kapitel 8

Resümee

Im Rahmen dieser Diplomarbeit wurde das DHCPv4- und DHCPv6-Protokoll untersucht. Als Basis für ein DHCP-Management wurden zunächst Szenarien betrachtet, welche sich unter DHCPv4 ergeben. Es stellte sich im Laufe der Diplomarbeit heraus, daß DHCPv6 für eine Berücksichtigung noch nicht gefestigt genug ist. Grund dafür ist die relativ kurz zurückliegende Entstehungszeit - die ersten betreffenden Internet-Drafts datieren auf Ende 1995.

Aus den Szenarien ergaben sich Anforderungen an das Management, für welches ein Lösungsansatz vorgestellt wurde. Aus den Managementanforderungen und dem vorgestellten Ansatz für das DHCP-Management wurde schließlich eine DHCP-MIB entwickelt, welche die Anforderungen von Management-Anwendungs- wie DHCP-Server-Datenhaltung erfüllt.

Ein wichtiger Bestandteil, die Implementierung der SNMP-Schnittstetlle in einem DHCP-Server-Programm, konnte nicht verwirklicht werden. Die Gründe dafür liegen u.a. in der Auswahl der Server-Software der Internet Software Corporation, welche sich quasi parallel zum Zeitablauf dieser Diplomarbeit unvorhergesehen dynamisch weiterentwickelt hat, so daß Analysen dieser Software sinnlos wurden, da die zugrundeliegende Serversoftware im Wochenabstand eine neue Version erfuhr (mit Änderungen der Datenstrukturen). Diese Implementierung wird jedoch dennoch verwirklicht, im Rahmen eines Fortgeschrittenenpraktikums an diesem Lehrstuhl.

Ich möchte mich bei meinem Betreuer, Stephen Heilbronner, für die vielfältigen Hilfestellungen bedanken, die er mir im Laufe dieser Diplomarbeit hat

angedeihen lassen. Ich habe durch diese Diplomarbeit viel Wissen und viele Erfahrungen gesammelt, die mir in meinem künftigen Berufsleben sicher von Nutzen sein werden.

München, den 15.August 1996

Gernot Riegert

Kapitel 9

DHCP-MIB

```
-- This is a MIB for DHCP in a scenario as mentioned in the
-- preceding chapters of this diploma thesis
DHCP-MIB DEFINITIONS ::=
BEGIN

IMPORTS
        Counter, Gauge, NetworkAdress, IpAdress, TimeTicks FROM RFC1155-SMI
        DisplayString FROM RFC1158-MIB
        OBJECT-TYPE FROM RFC-1212;

internet                OBJECT IDENTIFIER ::= { iso org(3) dod(6) 1 }
experimental            OBJECT IDENTIFIER ::= { internet 3 }
lrz                     OBJECT IDENTIFIER ::= { experimental 100 }
lrz-unix                OBJECT IDENTIFIER ::= { lrz 2 }
dhcpConfiguration       OBJECT IDENTIFIER ::= { lrz-unix 11 }
dhcpcurrentSituation    OBJECT IDENTIFIER ::= { lrz-unix 12 }
dhcpNetworkStatistics   OBJECT IDENTIFIER ::= { lrz-unix 13 }
dhcpOptionType          OBJECT IDENTIFIER ::= { dhcpConfiguration 1 }
currentSituationType    OBJECT IDENTIFIER ::= { dhcpcurrentSituation 1 }
networkStatisticsType   OBJECT IDENTIFIER ::= { dhcpNetworkStatistics 1 }

-- DHCP Options and region specific parameters for Configuration
-- IP adress parameters:
dhcpOptionSubnetAdress          OBJECT IDENTIFIER ::= { dhcpOptionType 1 }
dhcpOptionSubnetMask            OBJECT IDENTIFIER ::= { dhcpOptionType 2 }
dhcpOptionRouter                OBJECT IDENTIFIER ::= { dhcpOptionType 3 }
dhcpOptionTimeServer            OBJECT IDENTIFIER ::= { dhcpOptionType 4 }
dhcpOptionNameServer            OBJECT IDENTIFIER ::= { dhcpOptionType 5 }
```

```
dhcpOptionDomainNameServer      OBJECT IDENTIFIER ::= { dhcpOptionType 6 }
dhcpOptionLogServer             OBJECT IDENTIFIER ::= { dhcpOptionType 7 }
dhcpOptionCookieServer          OBJECT IDENTIFIER ::= { dhcpOptionType 8 }
dhcpOptionLprServer             OBJECT IDENTIFIER ::= { dhcpOptionType 9 }
dhcpOptionImpressServer         OBJECT IDENTIFIER ::= { dhcpOptionType 10 }
dhcpOptionResLocServer          OBJECT IDENTIFIER ::= { dhcpOptionType 11 }
dhcpOptionSwapServer            OBJECT IDENTIFIER ::= { dhcpOptionType 12 }
dhcpOptionPolicyFilterAdress    OBJECT IDENTIFIER ::= { dhcpOptionType 13 }
dhcpOptionPolicyFilterMask      OBJECT IDENTIFIER ::= { dhcpOptionType 14 }
dhcpOptionBroadcastAddress      OBJECT IDENTIFIER ::= { dhcpOptionType 15 }
dhcpOptionRouterSolAddress      OBJECT IDENTIFIER ::= { dhcpOptionType 16 }
dhcpOptionStatRouteDestAdress   OBJECT IDENTIFIER ::= { dhcpOptionType 17 }
dhcpOptionStaticRouteRouter     OBJECT IDENTIFIER ::= { dhcpOptionType 18 }
dhcpOptionNISServer             OBJECT IDENTIFIER ::= { dhcpOptionType 19 }
dhcpOptionNTPServer             OBJECT IDENTIFIER ::= { dhcpOptionType 20 }
dhcpOptionNetBiosOverTcpIpNameServer OBJECT IDENTIFIER ::= {dhcpOptionType 21}
dhcpOptionNetBiosOverTcpIpDatagramDistributionServer OBJECT IDENTIFIER ::=
{ dhcpOptionType 22 }
dhcpOptionXWindowSystemFontServer  OBJECT IDENTIFIER ::= { dhcpOptionType 23 }
dhcpOptionXWindowSystemDisplayManager OBJECT IDENTIFIER ::=
{ dhcpOptionType 24 }
dhcpOptionNisPlusDomainServer   OBJECT IDENTIFIER ::= { dhcpOptionType 25 }
dhcpOptionMobileIpHomeAgent     OBJECT IDENTIFIER ::= { dhcpOptionType 26 }
dhcpOptionSmtpServer            OBJECT IDENTIFIER ::= { dhcpOptionType 27 }
dhcpOptionPostOfficeProtocolServer OBJECT IDENTIFIER ::= { dhcpOptionType 28 }
dhcpOptionNntpServer            OBJECT IDENTIFIER ::= { dhcpOptionType 29 }
dhcpOptionDefaultWWWServer      OBJECT IDENTIFIER ::= { dhcpOptionType 30 }
dhcpOptionDefaultFingerServer   OBJECT IDENTIFIER ::= { dhcpOptionType 31 }
dhcpOptionDefaultIRCServer      OBJECT IDENTIFIER ::= { dhcpOptionType 32 }
dhcpOptionStreetTalkServer      OBJECT IDENTIFIER ::= { dhcpOptionType 33 }
dhcpOptionSTDAServer            OBJECT IDENTIFIER ::= { dhcpOptionType 34 }
specialAdress                   OBJECT IDENTIFIER ::= { dhcpOptionType 35 }
-- name parameters:
dhcpOptionDomainName            OBJECT IDENTIFIER ::= { dhcpOptionType 36 }
dhcpOptionSubnetName            OBJECT IDENTIFIER ::= { dhcpOptionType 37 }
dhcpOptionHostName              OBJECT IDENTIFIER ::= { dhcpOptionType 38 }
dhcpOptionNISDomain             OBJECT IDENTIFIER ::= { dhcpOptionType 39 }
dhcpOptionRequIpAddress         OBJECT IDENTIFIER ::= { dhcpOptionType 40 }
dhcpOptionTftpServerName        OBJECT IDENTIFIER ::= { dhcpOptionType 41 }
dhcpOptionClientId              OBJECT IDENTIFIER ::= { dhcpOptionType 42 }
dhcpOptionUserClassInfo         OBJECT IDENTIFIER ::= { dhcpOptionType 43 }
-- authentification of dhcp messages
encodingFunction                OBJECT IDENTIFIER ::= { dhcpOptionType 44 }
-- octet parameters:
```

```
dhcpOptionMeritDumpFile         OBJECT IDENTIFIER ::= { dhcpOptionType 45 }
dhcpOptionRootPath              OBJECT IDENTIFIER ::= { dhcpOptionType 46 }
dhcpOptionExtensionsPath        OBJECT IDENTIFIER ::= { dhcpOptionType 47 }
dhcpOptionIpForwardingEnableDisable OBJECT IDENTIFIER ::= { dhcpOptionType 48 }
dhcpOptionNonLocalSrcRtgEnDisable  OBJECT IDENTIFIER ::= { dhcpOptionType 49 }
dhcpOptionDefaultIpTtl          OBJECT IDENTIFIER ::= { dhcpOptionType 50 }
dhcpOptionAllSubnetsLocal       OBJECT IDENTIFIER ::= { dhcpOptionType 51 }
dhcpOptionPerformMaskDiscovery  OBJECT IDENTIFIER ::= { dhcpOptionType 52 }
dhcpOptionMaskSupplier          OBJECT IDENTIFIER ::= { dhcpOptionType 53 }
dhcpOptionPerformRouterDiscovery OBJECT IDENTIFIER ::={ dhcpOptionType 54 }
dhcpOptionTrailerEncapsulation  OBJECT IDENTIFIER ::= { dhcpOptionType 55 }
dhcpOptionEthernetEncapsulation OBJECT IDENTIFIER ::= { dhcpOptionType 56 }
dhcpOptionTcpKeepaliveGarbage   OBJECT IDENTIFIER ::= { dhcpOptionType 57 }
dhcpOptionVendorSpecInfo        OBJECT IDENTIFIER ::= { dhcpOptionType 58 }
dhcpOptionNetBiosOverTcpIpNodeType OBJECT IDENTIFIER ::= { dhcpOptionType 59 }
dhcpOptionNetBiosOverTcpIpScope OBJECT IDENTIFIER ::= { dhcpOptionType 60 }
dhcpOptionOptionOverload        OBJECT IDENTIFIER ::= { dhcpOptionType 61 }
dhcpOptionBootfileName          OBJECT IDENTIFIER ::= { dhcpOptionType 62 }
dhcpOptionVendorClassIdentifier OBJECT IDENTIFIER ::= { dhcpOptionType 63 }
-- integer parameters:
dhcpOptionTimeOffset    OBJECT IDENTIFIER ::= { dhcpOptionType 64 }
dhcpOptionBootfileSize  OBJECT IDENTIFIER ::= { dhcpOptionType 65 }
dhcpOptionMaxDatagramReassemblySize OBJECT IDENTIFIER ::=
{ dhcpOptionType 66 }
dhcpOptionPathMtuAgingTimeout OBJECT IDENTIFIER ::= { dhcpOptionType 67 }
dhcpOptionInterfaceMTU  OBJECT IDENTIFIER ::= { dhcpOptionType 68 }
dhcpOptionArpCacheTimeout OBJECT IDENTIFIER ::= { dhcpOptionType 69 }
dhcpOptionTcpDefaultTTL  OBJECT IDENTIFIER ::= { dhcpOptionType 70 }
dhcpOptionTcpKeepaliveInterval OBJECT IDENTIFIER ::= { dhcpOptionType 71 }
dhcpOptionIpAddressLeaseTime    OBJECT IDENTIFIER ::= { dhcpOptionType 72 }
dhcpOptionRenewalTimeValueT1    OBJECT IDENTIFIER ::= { dhcpOptionType 73 }
dhcpOptionRebindingTimeValueT2  OBJECT IDENTIFIER ::= { dhcpOptionType 74 }
-- the entries for Path Mtu Plateau table (one for each entry; if more entries
-- are needed, OIDs must be added accordingly
-- see dhcp option path mtu table in rfc 1533
dhcpOptionPathMtuPlateauEntryOne OBJECT IDENTIFIER ::= { dhcpOptionType 75 }
dhcpOptionPathMtuPlateauEntryTwo OBJECT IDENTIFIER ::= { dhcpOptionType 76 }

-- current situation parameters:
-- IP adresses:
allocatedIpAdress       OBJECT IDENTIFIER ::= { currentSituationType 1 }
usedRelayAgent          OBJECT IDENTIFIER ::= { currentSituationType 2 }
configuringServer       OBJECT IDENTIFIER ::= { currentSituationType 3 }
postRenumberAdress      OBJECT IDENTIFIER ::= { currentSituationType 4 }
```

```
-- Names:
clientId               OBJECT IDENTIFIER ::= { currentSituationType 5 }
clientName             OBJECT IDENTIFIER ::= { currentSituationType 6 }
class                  OBJECT IDENTIFIER ::= { currentSituationType 7 }
annotation             OBJECT IDENTIFIER ::= { currentSituationType 8 }
renumberMagicCookie    OBJECT IDENTIFIER ::= { currentSituationType 9 }
-- integers
bindexpires            OBJECT IDENTIFIER ::= { currentSituationType 10 }
-- octets:
hwAdress               OBJECT IDENTIFIER ::= { currentSituationType 11 }
vendorId               OBJECT IDENTIFIER ::= { currentSituationType 12 }
bootpclient            OBJECT IDENTIFIER ::= { currentSituationType 13 }
illegalUseDetected     OBJECT IDENTIFIER ::= { currentSituationType 14 }
doesRenumbering        OBJECT IDENTIFIER ::= { currentSituationType 15 }


-- This table describes the inheritance relationships between
-- regions of configuration

regionInheritanceTable OBJECT-TYPE
SYNTAX SEQUENCE OF RegionInheritanceEntry
ACCESS not-accessible
STATUS mandatory
DESCRIPTION
"This table marks the inheritance relationships between regions."
::= { dhcpConfiguration 2 }

regionInheritanceEntry OBJECT-TYPE
SYNTAX RegionInheritanceEntry
ACCESS not-accessible
STATUS mandatory
INDEX {name}
::= { regionInheritanceTable 1 }

RegionInheritanceEntry ::= SEQUENCE {
name                   DisplayString,
parent                 DisplayString
}
name OBJECT-TYPE
SYNTAX DisplayString
ACCESS read-write
STATUS mandatory
DESCRIPTION
"Region Identifier"
```

```
::= { dhcpRegionConfigInheritanceEntry 1 }

parent OBJECT-TYPE
SYNTAX DisplayString
ACCESS read-write
STATUS mandatory
DESCRIPTION
"Region Identifier of the region's parent region"
::= { dhcpRegionConfigInheritanceEntry 2 }


-- The following table is the fundament for region-specific configuring
-- with DHCP

dhcpRegionConfigTable OBJECT-TYPE
SYNTAX SEQUENCE OF DHCPRegionConfigEntry
ACCESS not-accessible
STATUS mandatory
::= { dhcpConfiguration 3 }

dhcpRegionConfigEntry OBJECT-TYPE
SYNTAX DHCPRegionConfigEntry
ACCESS not-accessible
STATUS mandatory
INDEX {regionName,configIdent}
::= {dhcpRegionConfigTable 1}

DHCPRegionConfigEntry ::= SEQUENCE{
regionName                DisplayString,
configIdent               INTEGER,
dhcpOptionType            OBJECT IDENTIFIER,
dhcpOptionValue           OCTET STRING
}

-- For certain subnets that differ in some parameters, there has to be
-- a means to identify the specific region of configuration according to
-- our object-oriented philosophy;
-- the configuration region is to be a administrator-defined area for
-- which certain configuration values are valid; this is according to
-- the object-oriented approach of data storage
regionName OBJECT-TYPE
SYNTAX DisplayString
ACCESS read-write
STATUS mandatory
```

```
DESCRIPTION
"Region Identifier. the configuration region is to be a
administrator-defined area for which certain configuration values are valid
 according to the object-oriented approach of data storage"
::= { dhcpRegionConfigEntry 1 }

configIdent OBJECT-TYPE
SYNTAX INTEGER
ACCESS read-write
STATUS mandatory
DESCRIPTION
"Index of the configuration of the region. Via this index multiple different
configuration entries for the same DHCP options are enabled."
::= { dhcpRegionConfigEntry 2 }

dhcpOptionType OBJECT-TYPE
SYNTAX OBJECT IDENTIFIER
ACCESS read-write
STATUS mandatory
::= { dhcpRegionConfigEntry 3 }

dhcpOptionValue OBJECT-TYPE
SYNTAX OCTET STRING
ACCESS read-write
STATUS mandatory
::= { dhcpRegionConfigEntry 4}


-- For Renumbering: A table for pre- and post-renumbering IP adresses
-- These adresses are unspecific and may identify servers, relay agents and
-- clients

dhcpRenumberTable OBJECT-TYPE
SYNTAX SEQUENCE OF RenumberEntry
ACCESS not-accessible
STATUS mandatory
::= { dhcpConfiguration 4 }

renumberEntry OBJECT-TYPE
SYNTAX RenumberEntry
ACCESS not-accessible
STATUS mandatory
INDEX {regionID,preRenumberAdress}
::= { dhcpRenumberTable 1 }
```

```
RenumberEntry ::= SEQUENCE {
regionID                        DisplayString,
indexNo                         INTEGER,
preRenumberAdress               IpAdress,
postRenumberAdress              IpAdress
}

regionID OBJECT-TYPE
SYNTAX DisplayString
ACCESS read-write
STATUS mandatory
DESCRIPTION
"Region ID"
::= { renumberEntry 1 }

indexNo OBJECT-TYPE
SYNTAX INTEGER
ACCESS read-write
STATUS mandatory
DESCRIPTION
"Enables several entries under one region"
::= { renumberEntry 2 }

preRenumberAdress OBJECT-TYPE
SYNTAX IpAdress
ACCESS read-write
STATUS mandatory
::= { renumberEntry 3 }

postRenumberAdress OBJECT-TYPE
SYNTAX IpAdress
ACCESS read-write
STATUS mandatory
::= { renumberEntry 4 }


-- now the available IP adresses for each region

availableIpAdressTable  OBJECT-TYPE
SYNTAX SEQUENCE OF AvailableIpAdressEntry
ACCESS not-accessible
STATUS mandatory
::= { dhcpConfiguration 5 }
```

```
availableIpAdressEntry OBJECT-TYPE
SYNTAX AvailableIpAdressEntry
ACCESS not-accessible
STATUS mandatory
INDEX {region,indexNo}
::= { availableIpAdressTable 1 }

AvailableIpAdressEntry ::= SEQUENCE {
region                  DisplayString,
indexNo                 INTEGER,
startAdress             IpAdress,
endAdress               IpAdress
}

region OBJECT-TYPE
SYNTAX DisplayString
ACCESS read-write
STATUS mandatory
DESCRIPTION
"Region Id."
::= { availableIpAdressEntry 1 }

indexNo  OBJECT-TYPE
SYNTAX INTEGER
ACCESS read-write
STATUS mandatory
DESCRIPTION
"Indicates different adress ranges"
::= { availableIpAdressEntry 2 }

startAdress OBJECT-TYPE
SYNTAX IpAdress
ACCESS read-write
STATUS mandatory
DESCRIPTION
"Start adress for an adress range."
::= { availableIpAdressEntry 3 }

endAdress OBJECT-TYPE
SYNTAX IpAdress
ACCESS read-write
STATUS mandatory
DESCRIPTION
```

```
"End adress of an adress range."
::= { availableIpAdressEntry 4 }


-- Current Situation parameters: Lease Information

currentLeaseTable OBJECT-TYPE
SYNTAX SEQUENCE OF CurrentLeaseEntry
ACCESS not-accessible
STATUS mandatory
::= { dhcpcurrentSituation 2 }

currentLeaseEntry OBJECT-TYPE
SYNTAX CurrentLeaseEntry
ACCESS not-accessible
STATUS mandatory
INDEX {regionId,indexNo}
::= { currentLeaseTable 1 }

CurrentLeaseEntry::= SEQUENCE {
regionId                        DisplayString,
indexNo                         INTEGER,
currentSituationType            OBJECT IDENTIFIER,
currentSituationValue           OCTET STRING
}

regionId OBJECT-TYPE
SYNTAX DisplayString
ACCESS read-write
STATUS mandatory
DESCRIPTION
"Identifying number for specific regions of configuration."
::= { currentLeaseEntry 1 }

indexNo OBJECT-TYPE
SYNTAX INTEGER
ACCESS read-write
STATUS mandatory
DESCRIPTION
"Identifying number for specific leases."
::= { currentLeaseEntry 2 }


currentSituationType OBJECT-TYPE
```

```
SYNTAX OBJECT IDENTIFIER
ACCESS read-write
STATUS mandatory
DESCRIPTION
"Current Situation column."
::= { currentLeaseEntry 3 }

currentSituationValue OBJECT-TYPE
SYNTAX IpAdress
ACCESS read-write
STATUS mandatory
DESCRIPTION
"Current Situation parameters."
::= { currentLeaseEntry 4 }


-- now information about active servers in a region

activeServerTable OBJECT-TYPE
SYNTAX SEQUENCE OF ActiveServerEntry
ACCESS not-accessible
STATUS mandatory
::= { dhcpcurrentSituation 3 }

activeServerEntry OBJECT-TYPE
SYNTAX ActiveServerEntry
ACCESS not-accessible
STATUS mandatory
INDEX {region,serverIndex}
::= { activeServerTable 1}

ActiveServerEntry ::= SEQUENCE {
region                          DisplayString,
serverIndex                     INTEGER,
activeServerIpAdress            IpAdress,
-- serserverStatus denotes the status of the server
-- e.g. running, down
activeServerStatus              DisplayString
}

region OBJECT-TYPE
SYNTAX DisplayString
ACCESS read-write
STATUS mandatory
```

```
DESCRIPTION
"Region Identifier"
::= { activeServerEntry 1 }

serverIndex OBJECT-TYPE
SYNTAX INTEGER
ACCESS read-write
STATUS mandatory
DESCRIPTION
"Number of Server"
::= { activeServerEntry 2 }

activeServerIpAdress OBJECT-TYPE
SYNTAX IpAdress
ACCESS read-write
STATUS mandatory
::= { activeServerEntry 3 }

activeServerStatus OBJECT-TYPE
SYNTAX DisplayString
ACCESS read-write
STATUS mandatory
::= { activeServerEntry 4 }


-- Current information about active Relay Agents in the region
activeRelayAgentTable OBJECT-TYPE
SYNTAX SEQUENCE OF ActiveRelayAgentEntry
ACCESS not-accessible
STATUS mandatory
DESCRIPTION
"Table of active (or inactive) Relay Agents"
::= { dhcpcurrentSituation 4 }

activeRelayAgentEntry OBJECT-TYPE
SYNTAX ActiveRelayAgentEntry
ACCESS not-accessible
STATUS mandatory
DESCRIPTION
"activeRelayAgentEntry"
INDEX {region, indexNo}
::= { dhcpCurrentLeaseTable 1 }

ActiveRelayAgentEntry ::= SEQUENCE {
```

```
region                          DisplayString,
indexNo                         INTEGER,
relayAgentAdress                IpAdress,
relayAgentStatus                DisplayString
}

region OBJECT-TYPE
SYNTAX DisplayString
ACCESS read-write
STATUS mandatory
DESCRIPTION
"Region Id"
::= { activeRelayAgentEntry 1 }

indexNo OBJECT-TYPE
SYNTAX INTEGER
ACCESS read-write
STATUS mandatory
DESCRIPTION
"Number of Relay Agent"
::= { activeRelayAgentEntry 2 }

relayAgentAdress OBJECT-TYPE
SYNTAX IpAdress
ACCESS read-write
STATUS mandatory
DESCRIPTION
"relay agent adress value"
::= { activeRelayAgentEntry 3 }

relayAgentStatus OBJECT-TYPE
SYNTAX DisplayString
ACCESS read-write
STATUS mandatory
DESCRIPTION
"Relay agent status value"
::= { activeRelayAgentEntry 4 }


-- Current information about unknown clients in the region
unknownClientTable OBJECT-TYPE
SYNTAX SEQUENCE OF UnknownClientEntry
ACCESS not-accessible
STATUS mandatory
```

```
DESCRIPTION
""
::= { dhcpcurrentSituation 5 }

unknownClientEntry OBJECT-TYPE
SYNTAX UnknownClientEntry
ACCESS not-accessible
STATUS mandatory
INDEX {regionNumber,unknownHwAdress}
::= { unknownClientTable 1 }

UnknownClientEntry ::= SEQUENCE {
regionNumber                  DisplayString,
indexNo                       INTEGER,
unknownHwAdress               OCTET STRING,
htype                         INTEGER,
hlen                          INTEGER,
lastseen                      INTEGER,
viaRelayAgent                 IpAdress
}

regionNumber OBJECT-TYPE
SYNTAX DisplayString
ACCESS read-write
STATUS mandatory
DESCRIPTION
"Region ID"
::= { unknownClientEntry 1 }

indexNo OBJECT-TYPE
SYNTAX INTEGER
ACCESS read-write
STATUS mandatory
DESCRIPTION
"Enumeration of unknown client"
::= { unknownClientEntry 2 }

unknownHwAdress OBJECT-TYPE
SYNTAX  OCTET STRING
ACCESS  read-write
STATUS  mandatory
DESCRIPTION
"HW adress of unknown system"
::= { unknownClientEntry 3 }
```

```
htype OBJECT-TYPE
SYNTAX INTEGER
ACCESS  read-write
STATUS  mandatory
DESCRIPTION
"hwType of system"
::= { unknownClientEntry 4 }

hlen OBJECT-TYPE
SYNTAX  INTEGER
ACCESS  read-write
STATUS  mandatory
DESCRIPTION
"HwLength of system"
::= { unknownClientEntry 5 }

lastseen OBJECT-TYPE
SYNTAX  INTEGER
ACCESS  read-write
STATUS  mandatory
DESCRIPTION
"Last time the system tried to be configured"
::= { unknownClientEntry 6 }

-- the relay agents that served the unknown client
viaRelayAgent OBJECT-TYPE
SYNTAX  IpAdress
ACCESS  read-write
STATUS  mandatory
DESCRIPTION
"Relay Agent that was used by the unknown system"
::= { unknownClientEntry 7 }

-- Current information about external clients in the region
-- external clients use DHCPINFORM to be configured

externalClientTable OBJECT-TYPE
SYNTAX SEQUENCE OF ExternalClientEntry
ACCESS not-accessible
STATUS mandatory
DESCRIPTION
"Table of external clients. External clients are systems that use
DHCPINFORM to be configured"
```

```
::= { dhcpcurrentSituation 6 }

externalClientEntry OBJECT-TYPE
SYNTAX ExternalClientEntry
ACCESS not-accessible
STATUS mandatory
INDEX {regionNumber,exHwAddress}
::= { dhcpCurrentLeaseTable 1 }

ExternalClientEntry ::= SEQUENCE {
regionNumber                    DisplayString,
indexNo                         INTEGER,
exClientId                      DisplayString,
exClientAdress                  IpAdress,
exHwAddress                     OCTET STRING,
exHwtype                        INTEGER,
exHlen                          INTEGER,
exLastseen                      INTEGER,
exRelayagent                    IpAdress
}
regionNumber OBJECT-TYPE
SYNTAX DisplayString
ACCESS  read-write
STATUS  mandatory
DESCRIPTION
"Region ID"
::= { externalClientEntry 1 }

indexNo OBJECT-TYPE
SYNTAX INTEGER
ACCESS  read-write
STATUS  mandatory
DESCRIPTION
"Enumeration index"
::= { externalClientEntry 2 }

exClientId OBJECT-TYPE
SYNTAX DisplayString
ACCESS  read-write
STATUS  mandatory
DESCRIPTION
"Client Identifier for external system."
::= { externalClientEntry 3 }
```

```
exClientAdress OBJECT-TYPE
SYNTAX IpAdress
ACCESS  read-write
STATUS  mandatory
DESCRIPTION
"IP Adress of the external system"
::= { externalClientEntry 3 }

exHwAddress OBJECT-TYPE
SYNTAX OCTET STRING
ACCESS  read-write
STATUS  mandatory
DESCRIPTION
"Hardware adress of external system"
::= { externalClientEntry 4 }

exHwtype OBJECT-TYPE
SYNTAX  INTEGER
ACCESS  read-write
STATUS  mandatory
DESCRIPTION
"Hardware type of external system"
::= { externalClientEntry 5 }

exHlen OBJECT-TYPE
SYNTAX  INTEGER
ACCESS  read-write
STATUS  mandatory
DESCRIPTION
"Hardware length"
::= { externalClientEntry 6 }

exLastseen OBJECT-TYPE
SYNTAX INTEGER
ACCESS  read-write
STATUS  mandatory
DESCRIPTION
"Last time the external client entered the region"
::= { externalClientEntry 7 }

-- the relay agents that were used by external clients
exRelayagent OBJECT-TYPE
SYNTAX  IpAdress
ACCESS  read-write
```

```
STATUS  mandatory
DESCRIPTION
"Relay Agent that was used by external system"
::= { externalClientEntry 8 }


-- Network statistics:
statisticsTable OBJECT-TYPE
SYNTAX SEQUENCE OF StatEntry
ACCESS not-accessible
STATUS mandatory
DESCRIPTION
"Data structure for description of server-specific statistics in a region"
::= { dhcpNetworkStatistics 2 }

statEntry OBJECT-TYPE
SYNTAX StatEntry
ACCESS not-accessible
STATUS mandatory
INDEX{regionId,indexNo}
::= { statisticsTable 1 }

StatEntry ::= SEQUENCE {
regionId                DisplayString,
indexNo                 INTEGER,
serverAdress            IpAdress,
serveruptime            INTEGER,
lastdbaccess            INTEGER,
discovers               INTEGER,
offers                  INTEGER,
requests                INTEGER,
declines                INTEGER,
acks                    INTEGER,
nacks                   INTEGER,
releases                INTEGER,
informs                 INTEGER,
bootprequests           INTEGER,
bootpreplys             INTEGER,
noOfLeases              INTEGER,
badPdus                 INTEGER
}

regionId OBJECT-TYPE
SYNTAX DisplayString
```

```
ACCESS  read-write
STATUS  mandatory
DESCRIPTION
"Identifier of the region"
::= { statEntry 1 }

indexNo OBJECT-TYPE
SYNTAX INTEGER
ACCESS  read-write
STATUS  mandatory
DESCRIPTION
"Enumeration Index"
::= { statEntry 2 }

serverAdress OBJECT-TYPE
SYNTAX IpAdress
ACCESS  read-write
STATUS  mandatory
DESCRIPTION
"Identifier of special server"
::= { statEntry 3 }

serveruptime OBJECT-TYPE
SYNTAX  INTEGER
ACCESS  read-write
STATUS  mandatory
DESCRIPTION
"The Uptime of the server since the last server restart"
::= { statEntry 4 }

-- Last time the server databasis was accessed
lastdbaccess OBJECT-TYPE
SYNTAX  INTEGER
ACCESS  read-write
STATUS  mandatory
DESCRIPTION
"Last access to server database"
::= { statEntry 5 }

-- The following are messages sent and received by the DHCP server
discovers OBJECT-TYPE
SYNTAX  INTEGER
ACCESS  read-write
STATUS  mandatory
```

```
DESCRIPTION
"number of dhcpdiscovers that reached the server"
::= { statEntry 6 }

offers OBJECT-TYPE
SYNTAX  INTEGER
ACCESS  read-write
STATUS  mandatory
DESCRIPTION
"number of dhcpoffers that were sent by the server"
::= { statEntry 7 }

requests OBJECT-TYPE
SYNTAX  INTEGER
ACCESS  read-write
STATUS  mandatory
DESCRIPTION
"number of dhcprequests that reached the server"
::= { statEntry 8 }

declines OBJECT-TYPE
SYNTAX  INTEGER
ACCESS  read-write
STATUS  mandatory
DESCRIPTION
"number of dhcpdeclines that reached the server"
::= { statEntry 9 }

acks OBJECT-TYPE
SYNTAX  INTEGER
ACCESS  read-write
STATUS  mandatory
DESCRIPTION
"number of dhcpacks that were sent by the server"
::= { statEntry 10 }

nacks OBJECT-TYPE
SYNTAX  INTEGER
ACCESS  read-write
STATUS  mandatory
DESCRIPTION
"number of dhcpnaks that were sent by the server"
::= { statEntry 11 }
```

```
releases OBJECT-TYPE
SYNTAX  INTEGER
ACCESS  read-write
STATUS  mandatory
DESCRIPTION
"number of dhcpdeclines that reached the server"
::= { statEntry 12 }

informs OBJECT-TYPE
SYNTAX  INTEGER
ACCESS  read-write
STATUS  mandatory
DESCRIPTION
"number of dhcpinforms that reached the server"
::= { statEntry 13 }

bootprequests OBJECT-TYPE
SYNTAX  INTEGER
ACCESS  read-write
STATUS  mandatory
DESCRIPTION
"number of bootprequests that reached the server"
::= { statEntry 14 }

bootpreplys OBJECT-TYPE
SYNTAX  INTEGER
ACCESS  read-write
STATUS  mandatory
DESCRIPTION
"number of bootpreplys that were sent by the server"
::= { statEntry 15 }

noOfLeases OBJECT-TYPE
SYNTAX  INTEGER
ACCESS  read-write
STATUS  mandatory
DESCRIPTION
"number of leases (active and expired) in the region"
::= { statEntry 16 }

-- Error statistics
badPdus OBJECT-TYPE
SYNTAX INTEGER
ACCESS  read-write
```

```
STATUS  mandatory
DESCRIPTION
"The number of incorrectly received pdus"
::= { statEntry 17 }

END
```

Abbildungsverzeichnis

Literaturverzeichnis

[AD96] S. Alexander and Ralph Droms. DHCP Options and BOOTP Vendor Extensions, draft-ietf-dhc-options-1533update-04.txt, May 1996.

[Ale93] S. Alexander. DHCP Options and BOOTP Vendor Extensions RFC 1533 Cathegory: Standards Track , October 1993.

[BP96] J. Bound and C. Perkins. Dynamic Host Configuration Protocol for IPv6 (DHCPv6), draft-ietf-dhc-dhcpv6-05.txt, June 1996.

[CFSD90] Jeffrey D. Case, Mark S. Fedorm, Martin S. Schoffstall, and James R. Davin. RFC 1157: A Simple Network Management Protocol, May 1990.

[CG85] Bill Croft and John Gilmore. RFC 951 : BOOTSTRAP PROTOCOL (BOOTP), September 1985.

[Com95] Douglas E. Comer. *Internetworking with TCP/IP*, volume 1 : Principles, Protocols and Architecture. Prentice Hall, third edition, 1995.

[Dro93] Ralph Droms. Dynamic Host Configuration Protocol, RFC 1541 Cathegory: Standards Track, October 1993.

[Dro96a] Ralph Droms. Authentication for DHCP Messages draft-ietf-dhc-authentication-02.txt, February 1996.

[Dro96b] Ralph Droms. Dynamic Host Configuration Protocol, draft-ietf-dhc-dhcp-07.txt, May 1996.

[Fei95] Sidnie Feit. *SNMP – A Guide to Network Management*. McGraw-Hill, 1995.

[FMMT84] Ross Finlayson, Timothy Mann, Jeffrey Mogul, and Marvin Thimer. RFC 903 : A Reverse Adress Resolution Protocol, June 1984.

[Gil96] Lowell Gilbert. Graceful renumbering of networks with DHCP, draft-ietf-dhc-renumbering-00.txt, April 1996.

[Gor90] Gora, W. and Speyerer, R. *ASN.1 - Abstract Syntax Notation One*. Datacom, 1990.

[HNag] S. Heilbronner and B. Neumair. Management mobiler Endsysteme: Anforderungen bei der Integration in bestehende Managementsysteme. In Dieter Wall, editor, *Organisation und Betrieb von DV-Versorgungsstrukturen*, November Deutscher Universitätsverlag.

[ISOa] ISO. ISO8824 : Open Systems Interconnection - Specification of Abstract Syntax, Notation One (ASN.1).

[ISOb] ISO. ISO8825 : Open Systems Interconnection - Specification of Basic Encoding, Rules for Abstract Syntax Notation One (ASN.1).

[JS93] Rainer Janssen and Wolfgang Schott. *SNMP : Konzepte – Verfahren – Plattformen*. DATACOM, 1993.

[KPS96] Charlie Kaufman, Radia Perlman, and Mike Speciner. *Network Security*. Prentice Hall, 1996.

[Lan96] Thomas Lankes. Fortgeschrittenenpraktikum Mobile-IP : Einrichtung einer Testumgebung unter Linux. Technical report, Ludwigs-Maximilians-Universität München, Germany, Institut für Informatik, 1996.

[Mar91] Marshall T. Rose. *The Simple Book: An Introduction to Management of TCP/IP-based Internets*. Prentice Hall, 1991.

[Mil92] David L. Mills. RFC 1305: Network Time Protocol (Version 3): Specification, Implementation and Analysis, March 1992.

[Moc87a] P. Mockapetris. RFC 1034: Domain Names - Concepts and Facilities, November 1987.

[Moc87b] P. Mockapetris. RFC 1035: Domain Names - Implementation and Specification, November 1987.

[MS96] Martin Maurer and Oliver Schabenberger. Hauptseminar Moderne Kommunikationskonzepte, Protokolle und Dienste : Mobile Endsysteme in TCP/IP-Netzen - Prinzipien und Infrastrukturanforderungen am Beispiel Mobile-IP. Technical report, Ludwigs-Maximilians-Universität München, Germany, Institut für Informatik, June 1996.

[Mü96] Gerhard Müller. Hauptseminar Moderne Kommunikationskonzepte, Protokolle und Dienste: IPv6 - Adressierung, Kompatibilität zu IPv4, Migrationsstrategien. Technical report, Ludwigs-Maximilians-Universität München, Germany, Institut für Informatik, June 1996.

[Per95] C. Perkins. Options for DHCPv6, draft-ietf-dhc-v6opts-00.txt, November 1995.

[Per96a] C. Perkins. Extensions for DHCPv6, draft-ietf-dhc-v6exts-01.txt, June 1996.

[Per96b] C. Perkins. IP Mobility Support: draft-ietf-mobileip-protocol-17.txt, May 1996.

[RD96] Yakov Rekhter and Ralph Droms. FQDN DHCP Option, draft-ietf-dhc-fqdn-opt-01.txt, April 1996.

[Rek96] Yakov Rekhter. Interaction between DHCP and DNS: draft-ietf-dhc-dhcp-dns-00.txt, February 1996.

[RM90] M. Rose and K. McCloghrie. RFC 1155: Structure and Identification of Management Information for TCP/IP-based Internets, May 1990.

[RM91] M. Rose and K. McCloghrie. RFC 1213: Management Information Base for Network Management of TCP/IP-based internets: MIB-II, March 1991.

[SH95] S.Deering and R. Hinden. RFC1883, Internet Protocol, Version 6, December 1995.

[Unt] Gertraud Unterreitmeier. Diplomarbeit Management von Sicherheitsdiensten für mobile Systeme; until february 1997.

[VTRB96] Paul Vixie, Susan Thomson, Yakov Rekhter, and Jim Bound. Dynamic Updates in the Domain Name System (DNS UPDATE), draft-ietf-dnsind-dynDNS-09.txt, March 1996.

[WCC+94] B. Wijnen, G. Carpenter, K. Curran, A. Sehgal, and G. Waters. Simple Network Management Protocol Distributed Protocol Interface Version 2.0, March 1994.

[Wim93] W. Wimer. RFC 1542 (Standards Track): Clarifications and Extensions for the Bootstrap Protocol, October 1993.

[Wor] IP Security Workgroup. http://www.ietf.cnri.reston.va.us/html.charters/wg-dir.html.