

ITM-Praktikum
Versuch 2: Niedrige Protokolle /
Broadcastprotokolle

Andreas Klingler, Hannes Stahl, Simon Lüke

13. Januar 2010

Inhaltsverzeichnis

1	vorbereitende Fragen	2
1.1	<i>DNS</i> -Hierarchie	2
1.2	Auflösung eines Hostnamens	2
1.3	Hierarchie bei IP-Adressen?	2
1.4	Auflösung einer IP-Adresse in einen Hostnamen	2
1.5	Die wichtigsten DNS Datentypen	2
1.6	Authorative Antworten	3
1.7	Zonen	3
1.8	Dynamisches DNS	3
1.9	Sicherheit bei <i>Dynamischem DNS</i>	3
1.10	Dynamische Adresszuweisung	3
1.11	Leasing	3
1.12	<i>BOOTP</i>	4
1.13	Probleme bei <i>BOOTP</i>	4
1.14	<i>DHCP</i>	4
1.15	Bedeutung verschiedener <i>DHCP</i> -Pakete	5
1.16	<i>DHCP</i> -Relay-Agent	5
1.17	TFTP	5
1.18	PXE	5
1.19	Bootvorgang via PXE	5
2	Versuchsdurchführung	6
2.1	Domain Name Service	6
2.1.1	Namensauflösung	6
2.1.2	<i>Root</i> -Server und Hierarchie im <i>DNS</i>	7
2.1.3	Andere <i>Resource Records</i>	7
2.1.4	Reverse Lookup	7
2.2	Konfiguration eines <i>Domain Name Servers</i>	8
2.3	Dynamisches <i>DNS</i>	9
2.4	<i>Dynamic Host Configuration Protocol (DHCP)</i>	9
2.5	<i>Diskless Client</i> per <i>DHCP/TFTP</i> booten	10

1 vorbereitende Fragen

1.1 DNS-Hierarchie

Das *Domain Name System (DNS)* ermöglicht es einem Domainnamen (mindestens) eine IP Adresse zuzuordnen, ähnlich einem Telefonbuch. Dabei weisen die Domainnamen eine **hierarchische Baumstruktur** auf. Die einzelnen Ebenen werden durch Punkte getrennt, die höchste steht dabei am weitesten rechts:

Beispiel einer Domain: e-technik.uni-ulm.de.

1. Die höchste Ebene ist der ., welcher die Wurzel aller Domains darstellt. Dieser Punkt wird üblicherweise oft weggelassen.
2. .de ist die (*Country-* oder) *Topleveldomain (CCTLD)*.
3. uni-ulm ist eine *Secondleveldomain*.
4. e-technik ist eine *Third-Level Domain*, umgangssprachlich als *Subdomain* bezeichnet.

1.2 Auflösung eines Hostnamens

1. Der angeforderte Hostname wird dem **Resolver des Betriebssystems** übergeben. Dieser startet eine rekursive Anfrage an seinen konfigurierten **Nameserver**.
2. Der **Nameserver** sucht iterativ den für die Zone der angeforderten **Topleveldomain** zuständigen Nameserver.
3. Anhand dieses Servers, kann die **Secondleveldomain** aufgelöst werden. Hier kann u. U. ein weiterer Auflöseschritt nötig sein, sind **Subdomains** vorhanden.
4. Wenn die Domain vollständig aufgelöst ist, wird die IP Adresse an den ersten **Nameserver** zurückgegeben, welcher das Ergebnis an unseren **Resolver** weiterreicht.

1.3 Hierarchie bei IP-Adressen?

Prinzipiell kann man IP-Adressen durch die Vergabe von Netzmasken hierarchisch untergliedern; an der Adresse an sich (4 Oktette) kann man jedoch **keine Hierarchie** ablesen.

1.4 Auflösung einer IP-Adresse in einen Hostnamen

Prinzipiell ist die Auflösung in diese Richtung nicht vorgesehen. Über das *DNS* ist es nicht einfach möglich, den zu einer IP gehörigen Hostnamen zu finden, denn im schlechtesten denkbaren Fall müssten alle weltweit verteilt vorgehaltenen *DNS*-Einträge durchsucht werden. Deshalb gibt es für die umgekehrte Auflösung extra Server, die solche Listen verwalten (*rDNS*). Dabei bleibt es dem IP-Adressen Nutzer freigestellt, seine IP Adresse auf den Servern in-addr.arpa zu listen, weshalb viele Adressen dort nicht zu finden sind. Außerdem ist es möglich, dass eine IP Adresse über mehrere Domainnamen angesprochen werden kann.

1.5 Die wichtigsten DNS Datentypen

- **A-Record:** *IPv4* Adresseintrag
- **AAAA-Record:** *IPv6* Adresseintrag
- **MX Record:** *Mail Exchange Server*
- **NS Record:** *Name Server*
- **CNAME Record:** *Canonical Name*

- **PTR Resource Record:** *Reverse lookup*

1.6 Authorative Antworten

Bei **Authorativen Antworten** auf DNS Anfragen sind Richtigkeit und Aktualität der Datensätze gesichert, wohingegen es sich bei **Non-Authorativen Antworten** lediglich um „gecachte“, also vielleicht veraltete, Einträge eines nicht direkt für die Zone verantwortlichen *Nameservers* handelt.

1.7 Zonen

Eine Zone gibt den Zuständigkeitsbereich eines *Nameservers* an. Mittels *NS Records* wird auf darunter liegende Zonen verwiesen.

1.8 Dynamisches DNS

Beim dynamischen DNS verändert sich die IP-Adresse hinter einem Namen von *Zeit zu Zeit*. Beispielsweise bei Adressvergabe durch *DHCP* kann ein Rechner (z.B. bei jeder Einwahl beim Provider) eine neue IP bekommen, der Domainname soll aber immer auf den gleichen Rechner zeigen. Dies wird mit dynamischen *DNS*-Einträgen realisiert, indem der Eintrag immer nach Vergabe einer neuen IP auf dem zuständigen Nameserver geändert wird. Dies birgt diverse Gefahren:

- Datensätze könnten böswillig verändert werden, werden keine entsprechenden Sicherheitsvorkehrungen getroffen.
- Datensätze könnten veraltet sein, so dass der Eintrag auf einen fremden Rechner verweist.

1.9 Sicherheit bei Dynamischem DNS

Eine Möglichkeit wäre eine Authentifizierung zu benutzen die vor unautorisierten Eintragsänderungen schützt. Der zweite o.g. Punkt lässt sich durch kurze *TTL*-Werte einschränken. Die Zeit bis ein Eintrag verfällt verbleibt als Restrisiko. Um dieses weiter zu reduzieren, könnte man vor Abschalten des Servers die Adresse auf eine vertrauenswürdige Seite umleiten lassen.

1.10 Dynamische Adresszuweisung

Für die dynamische Adresszuweisung wird ein Server benötigt, der die Adresslisten überwacht und freie Adressen auf Nachfrage vergeben kann. Dabei muss das Netz und der Server eine Möglichkeit bereitstellen, ohne (IP-)Adresse eine Verbindung zwischen Server und Client aufzubauen, damit eine freie Adresse übermittelt werden kann.

Das ist auch der Grund, warum eine erste Anfrage per Broadcast geschehen muss: Der Client hat ja selbst keine Adresse, geschweige denn eine Zieladresse eines *DHCP/BOOTP*-Servers! So „fragt“ er einfach alle Teilnehmer nach einer Adresse.

1.11 Leasing

Leasing bedeutet ins Deutsche übersetzt „anmieten“. In Bezug auf die Adressvergabe soll dies bedeuten, dass eine Adresse von einem Client nur „gemietet“ ist und nicht dauerhaft für ihn bestimmt ist. Nach einer bestimmten Zeit muss die Adresse entweder erneuert oder zurückgegeben werden.

1.12 BOOTP

BOOTP steht für Bootstrap Protokoll und dient der Zuweisung einer IP-Adresse und weiterer Parameter wie Subnetzmaske, Gateway etc. an ein Gerät, wie z.B. ein Terminal oder ein Netzwerkdrucker. Für PCs kommt meist die Erweiterung *DHCP* zum Einsatz.

Zur Anforderung (*boot request*) sendet der Client ein UDP-Paket mit seiner MAC-Adresse und eine 4 Byte lange Zufallszahl an die Broadcastadresse. Für die Zuweisung eines Adresssatzes müssen diese im *boot reply* wieder enthalten sein. Da auch die Antwort an die Broadcastadresse gesendet wird, wurden die Ports 68 (Client) und 67 (Server) dafür reserviert um Missverständnisse zu vermeiden. Darüber hinaus wird dem Client nur eine IP-Adresse zugewiesen, wenn seine MAC-Adresse in einer Tabelle steht. Optional kann der Server noch weitere Angaben versenden:

- IP & Hostname eines Bootservers um per *TFTP* zu starten
- Name & Pfad der Bootdatei
- Verzeichnisname der Bootpartition

1.13 Probleme bei BOOTP

- Adressen sind im Gegensatz zu *DHCP* relativ statisch in einer Datenbank vermerkt. Bei vielen wechselnden Clients sind die Adressen schnell aufgebraucht.
- *BOOTP* konnte nicht alle nötigen Parameter zur Netzwerkkonfiguration der Clients aufnehmen. Als Behelfslösung wurden manche Daten in das „herstellerspezifische Feld“ eingetragen. *DHCP* unterstützt die wichtigsten Konfigurationsparameter.
- Die ursprüngliche Implementierung von *BOOTP* war es auch nicht möglich, die Konfiguration im laufenden Betrieb zu erneuern. Stattdessen musste das System neu gebootet werden. *DHCP* dagegen verlängert seine IP-Adresse selbstständig im laufenden Betrieb.

1.14 DHCP

DHCP wird für die dynamische Adressvergabe in Netzwerken verwendet. Es kann dabei eine komplette Netzwerk Konfiguration an einen neuen Client übergeben werden, so dass das System sofort arbeiten kann ohne dass der Nutzer etwas über das Netz wissen muss.

1. Ein Client sendet unter der Absender-IP 0.0.0.0 einen *UDP Broadcast* an Port 67 mit einem *DHCP DISCOVER*. In dieser Nachricht befindet sich auch seine MAC-Adresse und eine zufällig generierte 4 Byte lange Zahl.
2. Verschiedene *DHCP*-Server senden ebenfalls per *UDP Broadcast* *DHCP OFFER* auf den Port 68. Der Client erkennt die für ihn bestimmten Antworten anhand seiner MAC-Adresse und der Zufallszahl.
3. Der Client wählt das „beste“ Angebot aus (z. B. anhand der Leasingzeit) und sendet einen *DHCP REQUEST*. Jetzt weiß der Server, dass der Client diese Adresse haben will, gleichzeitig gilt diese Nachricht als Absage für alle weiteren *DHCP OFFER*.
4. Im nächsten Schritt bestätigt der Server die IP mit einem *DHCP ACK* oder lehnt sie mit einem *DHCP NACK* ab.
5. Kurz bevor der Client die IP verwendet, prüft er selbst nochmals die Verfügbarkeit der IP. Sollte sich herausstellen, dass die IP in Verwendung ist, schickt der Client ein *DHCP DECLINE* auf das Netzwerk.
6. Wenn ein Rechner den *Lease* nicht mehr verlängern will, kann er die IP offiziell freigeben mit einem *DHCP RELEASE*.

1.15 Bedeutung verschiedener DHCP-Pakete

1. *DHCP-DISCOVER*: *UDP-Broadcastnachricht* von einem Client, der eine IP-Adresse beziehen will.
2. *DHCP-OFFER*: Angebot einer IP-Adresse von einem *DHCP-Server*.
3. *DHCP-REQUEST*: Der Client hat ein Angebot ausgewählt und will es mit dieser Nachricht annehmen/verlängern.
4. *DHCP-ACK*: Der Server bestätigt den *REQUEST*, wenn die Adresse noch frei ist und der Client sie benutzen darf.
5. *DHCP-NACK*: Der Server lehnt den *REQUEST* ab, weil die Adresse evtl. anderweitig vergeben wurde.
6. *DHCP-DECLINE*: Sollte die IP doch schon vergeben sein, sendet der Client eine *DECLINE* Nachricht.
7. *DHCP-RELEASE*: Freigabe einer „geleaste“ IP durch den Client.

1.16 DHCP-Relay-Agent

Ein *DHCP-Relay-Agent* kann *DHCP/BOOTP* Pakete erkennen und über Subnetze hinweg zu einem *DHCP-Server* routen. Der Agent muss also auf einem Router laufen! Ohne ihn bräuchte man für jedes Subnetz einen eigenen *DHCP-Server*.

1.17 TFTP

Das *Trivial File Transfer Protocol (TFTP)* ist ein bewusst einfach gehaltenes Protokoll, um Daten von einem Server lesen und schreiben zu können. *TFTP* nutzt *UDP* als Transportschicht. Es beinhaltet keine Authentifizierung, Dateirechteverwaltung oder ein einfaches Auflisten der vorhandenen Dateien. Es wird beispielsweise für festplattenlose Clients verwendet um das Betriebssystem von einem Server zu laden.

Nach einer Anfrage des Clients am Server Port 69, wechselt der Server für seine Antwort (angefragte Datei wird gesendet oder Empfangsbereitschaft signalisiert) auf einen Port außerhalb des von der IANA reservierten Bereichs, also Port > 1024.

1.18 PXE

Das *Pre Execution Environment (PXE)* ist eine Software bzw. Firmware, um einen Rechner von einem Netzwerkservers booten zu können. Mit Hilfe von *PXE* können Daten über das Netzwerk geladen und dann ausgeführt werden.

1.19 Bootvorgang via PXE

- Das *PXE* des Clients sendet einen *extended DHCP-Broadcast* über Port 67, um Informationen zum weiteren Booten anzufordern.
- Der Server sendet daraufhin eine entsprechende *extended DHCP OFFER* ebenfalls per Broadcast über Port 68 zurück. Darin ist z.B. zur Identifizierung die bereits vom *PXE-Client* mitgeteilte *UUID* bzw. *GUID* enthalten.
- Außerdem versucht die *PXE-Firmware* per *DHCP* eine IP-Adresse zugewiesen zu bekommen.

- Versuchen mit einer gültigen IP-Adresse sendet der *PXE*-Client ein *DHCPINFORM*-Paket über Port 4011 an den Boot-Server seiner Wahl. Alternativ kann das Paket auch wieder als Broadcast per Port 67 gesendet werden.
- Daraufhin antwortet der angesprochene und nach den in *DHCPINFORM* gewünschten Informationen zuständige Server mit einem *extended DHCPACK* an den Port des Clients von dem das *DHCPINFORM*-Paket kam.

Alternativ kann *BOOTP* genutzt werden, um eine IP-Konfiguration und eine Netzwerkressource, die das Image eines Betriebssystems zur Verfügung stellt, zu erhalten, soll z.B. ein clientenspezifisches Image verteilt werden.

Das *PXE* wird üblicherweise von einem kleinen *BIOS* auf der Netzwerkkarte initiiert.

2 Versuchsdurchführung

2.1 Domain Name Service

2.1.1 Namensauflösung

In Listing 1 findet sich die Zusammenfassung des *Wireshark*-Mittschnitts einer Namensauflösung: von 10.2.1.1 aus soll www.uni-ulm.de „angepingt“ werden.

Listing 1: *Wireshark*-Mittschnitt Namensauflösung, ausgelöst durch *ping*-Kommando.

```

1 0.000000000 10.2.1.1 10.2.1.2 DNS Standard query A www.uni-ulm.de
2 0.039019000 10.2.1.2 10.0.0.99 DNS Standard query A www.uni-ulm.de
3 0.040150000 10.2.1.2 10.0.0.99 DNS Standard query NS <Root>
4 0.041681000 10.0.0.99 10.2.1.2 DNS Standard query response NS
  H.ROOT-SERVERS.NET NS I.ROOT-SERVERS.NET NS J.ROOT-SERVERS.NET NS
  K.ROOT-SERVERS.NET NS L.ROOT-SERVERS.NET NS M.ROOT-SERVERS.NET NS
  A.ROOT-SERVERS.NET NS B.ROOT-SERVERS.NET NS C.ROOT-SERVERS.NET NS
  D.ROOT-SERVERS.NET NS E.ROOT-SERVERS.NET NS F.ROOT-SERVERS.NET NS
  G.ROOT-SERVERS.NET
5 0.042635000 10.0.0.99 10.2.1.2 DNS Standard query response A 134.60.1.25
[... ]
8 0.048774000 10.2.1.2 10.2.1.1 DNS Standard query response A 134.60.1.25
9 0.049529000 10.2.1.1 134.60.1.25 ICMP Echo (ping) request
10 0.050007000 134.60.1.25 10.2.1.1 ICMP Echo (ping) reply
11 0.050308000 10.2.1.1 10.2.1.2 DNS Standard query PTR
  25.1.60.134.in-addr.arpa
12 0.052850000 10.2.1.2 10.0.0.99 DNS Standard query PTR
  25.1.60.134.in-addr.arpa
13 0.056567000 10.0.0.99 10.2.1.2 DNS Standard query response PTR
  www.rz.uni-ulm.de
14 0.058789000 10.2.1.2 10.2.1.1 DNS Standard query response PTR
  www.rz.uni-ulm.de
15 1.052448000 10.2.1.1 134.60.1.25 ICMP Echo (ping) request
16 1.052925000 134.60.1.25 10.2.1.1 ICMP Echo (ping) reply

```

- **Paket 1:** Anfrage (*DNS Standard query*) des Clients an seinen *Nameserver* (10.2.1.2).
- **Paket 2,3:** 10.2.1.2 fragt wiederum den *Nameserver* des Subnetzes (10.0.0.99) nach www.uni-ulm.de; Außerdem wird nach den *Root*-Servern gefragt.
- **Paket 4:** Antwort von 10.0.0.99, die die *Root*-Servernamen enthält.

- **Paket 5:** Antwort von 10.0.0.99, die die angefragte IP 134.60.1.25 von www.uni-ulm.de enthält.
- Die Pakete 6+7 enthielten eine ARP Abfrage, die im Listing ausgespart wurde.
- **Paket 8:** 10.2.1.2 beantwortet wiederum dem ursprünglichen Client die Anfrage.
- **Paket 9+10:** Das *ping*-Kommando kann Pakete an die gefundene IP-Adresse versenden.
- **Paket 11-14:** Ein *reverse lookup* des *ping*-Kommandos, wahrscheinlich für die Kommandozeilenausgabe.
- **Paket 15 ff:** Nächster Ping.

2.1.2 Root-Server und Hierarchie im DNS

Auf die Abfrage der Wurzel des *DNS* antwortet der Nameserver *a.root-server.net* mit den 14 zur Zone, deren *SOA* er ist, gehörigen *Root*-Servern. Diese oberste Ebene ist nicht „authoritativ“. Im zusätzlichen Abschnitt übermittelt er die *IPv4*- und teilweise auch schon vorhandenen *IPv6*-Adressen von 8 der Server.

Die *Root*-Server-Abfrage für die *.de*-Domäne liefert 6 *Nameserver*-Einträge und für uni-ulm.de erhalten wir dieselben 6, da die *Root*-Server die Informationen für die *Second-Level-Domain* nicht selber vorhalten und so auf die zuständigen, in Listing 2 aufgeführten Server verweisen. Die zurückgegebenen Server sind natürlich jeweils „authoritativ“.

Listing 2: „authoritativ“ Namensserver der de-Domäne

de.	172800	IN	NS	F.NIC.de.
de.	172800	IN	NS	Z.NIC.de.
de.	172800	IN	NS	C.DE.NET.
de.	172800	IN	NS	A.NIC.de.
de.	172800	IN	NS	S.DE.NET.
de.	172800	IN	NS	L.DE.NET.

Mit dem Befehl `dig @dns1.uni-ulm.de rz.uni-ulm.de` erhält man den für *rz.uni-ulm.de* zuständigen „authoritativ“ Server *dns1.uni-ulm.de*. Fragt man diesen nun mit `dig @dns1.uni-ulm.de login.rz.uni-ulm.de` nach *login.rz.uni-ulm.de* gibt er neben den anderen „authoritativ“ Servern der Zone auch die IP 134.60.1.41 des Loginservers zurück.

2.1.3 Andere Resource Records

Für das Versenden von Emails wird ein sogenannter *MX record* benötigt. `dig @dns2.uni-ulm.de uni-ulm.de any` liefert hierfür den Eintrag

uni-ulm.de.	172800	IN	MX	50 smtp.uni-ulm.de.
-------------	--------	----	----	---------------------

zurück, aus dem ein *Mailtransferagent* erfahren kann, dass eine Email mit dem der *Domain Part* *uni-ulm.de* an *smtp.uni-ulm.de* weitergeleitet werden soll.

2.1.4 Reverse Lookup

Mit dem Befehl `dig -x 134.60.237.254` ist es möglich die gewünschte IP-Adresse rückwärts aufzulösen. Dazu werden, wie man in der *ANSWER SECTION* sieht die *in-addr.arpa*. Server abgefragt, welche gewillten Nutzern den Dienst des *reverse lookup* zur Verfügung stellen.

Listing 3: Ausschnitt *reverse lookup* für 134.60.237.254

```
;; QUESTION SECTION:
;254.237.60.134.in-addr.arpa.    IN      PTR

;; ANSWER SECTION:
254.237.60.134.in-addr.arpa. 172671 IN  PTR      pa-gw.rz.uni-ulm.de.
```

2.2 Konfiguration eines Domain Name Servers

„caching-only“ Namensserver Die Abfrage (z.B. `apfelmus:~# dig @localhost web.de`) beim zum *Nameserver* gemachten Rechner *apfelmus* liefert die gewünschte IP zurück, die Antwort ist nicht „authoritativ“. Außerdem wird mitgeliefert welche Server „authoritativ“ sind; im Bsp. sind das *nsx1.web.de.* und *nsx2.web.de.*

Master Namensserver Wie in Listing 4 zu sehen ist, hat die Konfiguration geklappt, allerdings war in der Datenbank eine falsche IP-Adresse eingetragen. Die Antwort, wenn die Abfrage von *kaiserschmarrn* aus erfolgt, ist „authoritativ“.

Listing 4: Abfrage des Master Nameservers einer eigenen Domäne

```
apfelmus:~# dig @localhost
      kaiserschmarrn.dnsversuch.itm.e-technik.uni-ulm.de

; <<>> DiG 9.3.4 <<>> @localhost
      kaiserschmarrn.dnsversuch.itm.e-technik.uni-ulm.de
; (1 server found)
;; global options: printcmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 52376
;; flags: qr aa rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 1, ADDITIONAL: 1

;; QUESTION SECTION:
;kaiserschmarrn.dnsversuch.itm.e-technik.uni-ulm.de. IN A

;; ANSWER SECTION:
kaiserschmarrn.dnsversuch.itm.e-technik.uni-ulm.de. 2345 IN A 10.2.1.5

;; AUTHORITY SECTION:
dnsversuch.itm.e-technik.uni-ulm.de. 604800 IN NS
      ns.dnsversuch.itm.e-technik.uni-ulm.de.

;; ADDITIONAL SECTION:
ns.dnsversuch.itm.e-technik.uni-ulm.de. 604800 IN A 10.2.1.2

;; Query time: 4 msec
;; SERVER: 127.0.0.1#53(127.0.0.1)
;; WHEN: Wed May 20 15:42:04 2009
;; MSG SIZE rcvd: 117

apfelmus:~#
```

2.3 Dynamisches DNS

Nach der Konfiguration war es wie in Listing 5 zu sehen ist, möglich über eine sichere Verbindung (mittels *private-public-key*-Verfahren) die Informationen auf dem Nameserver zu aktualisieren.

Listing 5: Update des *Nameservers*

```
kaiserschmarrn:~# nsupdate -k Ktest.+157+10674.private
> update delete kaiserschmarrn.dnsversuch.itm.e-technik.uni-ulm.de. A
> send
> update add kaiserschmarrn.dnsversuch.itm.e-technik.uni-ulm.de 86400 A
    193.99.144.80
> send
kaiserschmarrn:~#
```

Die danach folgende Abfrage mit *dig* war erfolgreich.

2.4 Dynamic Host Configuration Protocol (DHCP)

Bei der Nutzung von *DHCP* konnten die Pakete, die auszugsweise in Listing 6 zu finden sind mitgeschnitten werden:

Listing 6: *Wireshark*-Mitschnitt Adresszuweisung per *DHCP*.

```
[...]
94  55.177929  0.0.0.0  255.255.255.255  DHCP  DHCP Discover  - Transaction ID
    0x23b49e7d
95  55.178832  10.0.0.99  10.2.1.2  DHCP  DHCP Offer    - Transaction ID
    0x23b49e7d
96  55.257661  0.0.0.0  255.255.255.255  DHCP  DHCP Request  - Transaction ID
    0x23b49e7d
97  55.258301  10.0.0.99  10.2.1.2  DHCP  DHCP ACK      - Transaction ID
    0x23b49e7d
[...]
121 85.320572  10.2.1.2  10.0.0.99  DHCP  DHCP Request  - Transaction ID
    0x23b49e7d
122 85.321499  10.2.1.100  10.2.1.2  DHCP  DHCP ACK      - Transaction ID
    0x23b49e7d
[...]
```

- **Paket 94:** *DHCP Discover*, *apfelmus* schickt eine Broadcastnachricht als „Bitte“ eine IP-Adresse zu erhalten; mit dabei ist die *Transaction ID* 0x23b49e7d zur Identifikation der Antwort, die in Paket 95 folgt.
- **Paket 95:** *DHCP Offer*, geht an die schon vorher von *apfelmus* genutzte IP, die dem DHCP-Server schon bekannt war, wieder wird die *Transaction ID* mitgesendet.
- **Paket 96:** *DHCP Request* Der Client *apfelmus* erbittet die Benutzung der zugeteilten IP, wiederum als *Broadcast* und mit *Transaction ID*.
- **Paket 97:** *DHCP ACK* Der Server bestätigt die Benutzung der IP-Adresse.
- **Paket 121,122:** 30 Sekunden später erfolgt eine weiterer kurzer „*handshake*“, um eine fortdauernd gewünschte Benutzung der IP-Adresse zu signalisieren; dies wiederholt sich alle 30 Sekunden.

2.5 Diskless Client per DHCP/TFTP booten

- Im Gegensatz zum vorigen Abschnitt, bei dem der *DHCP*-Client schon eine IP-Adresse genutzt hatte und diese dem *DHCP*-Server schon bekannte Adresse dem Client wiederum zugewiesen wurde, erfolgt beim Bootvorgang die *DHCP*-Kommunikation ausschließlich über die globale Broadcastadresse 255.255.255.255.
- Schon im *DHCP Discover*-Paket fragt der Client per *Bootstrap Protocol* nach einem ladbaren Bootimage. Das zu ladende Image und der entsprechende Server wird auch schon in der *DHCP Offer* mitgeteilt. Dies war im vorigen Abschnitt – bei reinem DHCP – nicht der Fall.
- Daraufhin beginnt der Client per *TFTP* das Image herunterzuladen. Da die Übertragung per *UDP* erfolgt und damit eine Absicherung des Datenstroms fehlt, muss jeder vom Server übertragene Daten-*Block* vom Client bestätigt werden.
- Genau genommen, wird zuerst *PreExecutionEnvironment*, dann eine Konfigurationsdatei dazu und dann der eigentlich zu bootende Kernel übertragen.
- Beim starten des Kernels wird dann nach *DHCP* und *TFTP*, welche zum laden desselben benötigt wurden, NFS verwendet, um das Dateisystem des startenden Systems einzubinden.