Hochschule
Bonn-Rhein-Sieg
Fachbereich Informatik

Servicemigration von IPv4- zu IPv4-/IPv6-Dual-Stack-Umgebungen

Abschlussarbeit

zur Erlangung des Grades Bachelor of Science
im Studiengang Computer Science

vorgelegt von
Andreas Brück

Erstbetreuer: Prof. Dr. Martin Leischner
Hochschule Bonn-Rhein-Sieg

Zweitbetreuer: Prof. Dr. Kerstin Uhde
Hochschule Bonn-Rhein-Sieg

Eingereicht am: 8. März 2013

Eidesstattliche Erklärung

Ich versichere an Eides statt, die von mir vorgelegte Arbeit selbstständig verfasst zu haben. Alle Stellen, die wörtlich oder sinngemäß aus veröffentlichten oder nicht veröffentlichten Arbeiten anderer entnommen sind, habe ich als entnommen kenntlich gemacht. Sämtliche Quellen und Hilfsmittel, die ich für die Arbeit benutzt habe, sind angegeben. Die Arbeit hat mit gleichem Inhalt bzw. in wesentlichen Teilen noch keiner anderen Prüfungsbehörde vorgelegen.

.....

(Ort, Datum) (Unterschrift)

Danksagung

Ich möchte mich an dieser Stelle bei all denen bedanken, die mich bei der Anfertigung meiner Bachelorarbeit so tatkräftig unterstützt haben. Besonderer Dank gilt:

Prof. Dr. Martin Leischner und Prof. Dr. Kerstin Uhde die mich während meiner Abschlussarbeit betreut und organisatorisch enorm unterstützt haben.

Dirk Gebhardt für das umfangreiche Korrekturlesen und die Bekämpfung des Kommata Chaos.

Martina Kannnen für ihre konstruktive Kritik, ihre unglaubliche Motivation und die unzähligen Stunden die Sie für mich geopfert hat. Ausserdem verdanke ich ihr meine große Leidenschaft zur Telekommunikation. Vielen Dank dafür Martina!

Meiner Mutter Gabriele Brück, die mir jeden Tag gezeigt hat warum der Weg wichtiger ist als das Ziel.

Inhaltsverzeichnis

Abbildungsverzeichnis

Tabellenverzeichnis

Abkürzungsverzeichnis

API	Application Programming Interface
ASCII	American Standard Code for Information Interchange
BIND	Berkeley Internet Name Domain Server
DHCP	Dynamic Host Configuration Protocol
DNS	Domain Name System
FTP	File Transfer Protocol
HTTP	Hypertext Transfer Protocol
HTTPS	Hypertext Transfer Protocol Secure
IE	Internet Explorer
IETF	Internet Engineering Task Force
IPv4/v6	Internet Protocol Version 4/6
IIS	Internet Information Services
IMAP	Internet Message Access Protocol
ISC	Internet Systems Consortium
NTP	Network Time Protocol
POP3	Post Office Protocol 3
ProFTPD	Pro FTP Daemon
Radvd	Router Advertisement Daemon
RDNSS	Recursive DNS Server
RFC	Request for Comments
SCTP	Stream Control Transmission Protocol
SLAAC	Stateless Address Autoconfiguration
SMTP	Simple Mail Transfer Protocol
SNMP	Simple Network Management Protocol
TCP	Transmission Control Protocol
UDP	User Datagram Protocol

1 Einleitung

Aufgrund des stetig steigenden Bedarfs an IP-Adressen und der gegebenen Restriktion des Adressraums von IPv4, wird eine Migration nach IPv6 in naher Zukunft unumgänglich. Da eine Vielzahl an Diensten und Applikationen weiterhin ausschließlich über IPv4 kommunizieren, werden Mechanismen zur Kompatibilität beider Protokolle benötigt. Mit Hilfe von Translation von Adressen, Tunneling und Dual-Stack wird dieses Problem behoben. Dual-Stack ist dabei der am häufigsten empfohlene Mechanismus und unterstützt IPv4 sowie IPv6 parallel auf einem Host [Hag11, S. 45f.]. Bis IPv4 aus allen Netzen verschwindet, werden laut Experten mehrere Jahre vergehen [EK09, S. 121]. In der Übergangsphase werden die beiden IP-Protokolle im Parallelbetrieb laufen. Für Services, wie beispielsweise HTTP, bedeutet dies eine notwendige Servicemigration für den IPv4-/IPv6-Dual-Stack-Betrieb (s. Abbildung 1.1).

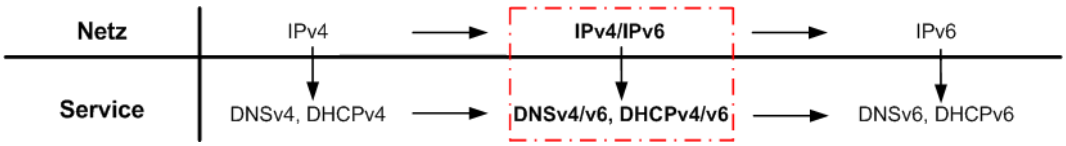


Abbildung 1.1: Dual-Stack-Betrieb als Übergangsphase

1.1 Ziel der Arbeit

Ziel dieser Arbeit ist es, einen Leitfaden für die Servicemigration von einer reinen IPv4- zu einer IPv4-/IPv6-Dual-Stack-Umgebung zu entwickeln. Dieser soll als systematische Hilfestellung für Administratoren dienen, die in ihrem Unternehmen eine Dual-Stack-Umgebung einsetzen wollen. Dabei werden folgende Fragestellungen beantwortet:

- Wie spezifiziere ich die erforderlichen Dual-Stack-Erweiterungen eines Services?
- Wie wähle ich eine geeignete server-/ clientseitige Dual-Stack-Implementierung aus?
- Was ist bei der Installation und Konfiguration einer Dual-Stack-Implementierung zu beachten?

- Wie teste ich eine Implementierung auf ihre Dual-Stack-Fähigkeit?
- Welche Reihenfolge ist bei der Migration der Dual-Stack-Services sinnvoll?
- Welche weiteren Aspekte sind zu beachten (z.B. Organisatorische Maßnahmen, Sicherheit, Netzmanagement etc.)?

Die im Rahmen dieser Arbeit migrierten Services DHCP, DNS, HTTP und FTP, dienen dabei als Hilfestellung in Form von konkreten Beispielen. Zur Eingrenzung dieses Themas ist es erforderlich zu erwähnen, dass ausschließlich die gleichen Funktionalitäten, die für IPv4 bereits gegeben sind, für IPv6 bereitgestellt werden. In dieser Arbeit wird nicht auf Funktionalitäten eingegangen, die ausschließlich für IPv6 spezifiziert sind, wie bespielsweise Mobile IPv6 oder Quality of Service.

1.2 Methodik

Ausgehend von den oben erwähnten Fragestellungen wurden vier exemplarische Services ausgewählt, um notwendige Erfahrungen zu sammeln. Für jeden Service wurden die entsprechenden *Request for Comments (RFC)* recherchiert, welche die Erweiterungen für IPv6 spezifizieren. Auf Grundlage der daraus gewonnen Informationen, wurden geeignete server- und clientseitige Implementierungen ausgewählt und auf ihre Dual-Stack-Fähigkeit überprüft. Dabei wurde jede Implementierung auf ihre Serviceverfügbarkeit und -funktionalität getestet. Aus den daraus gewonnen Erkenntnissen wird ein Leitfaden zur Servicemigration entwickelt.

1.3 Aufbau der Arbeit

In Kapitel 2 wird zunächst der Begriff Service zur weiteren Verwendung in dieser Arbeit definiert. Zudem werden die Aufgaben der Services, die im darauf folgenden Kapitel migriert werden, dargelegt. Näher erläutert werden zudem die Socketkommunikation, sowie wichtige Aspekete im Bezug auf IPv6-Adressen.

In Kapitel 3 erfolgt die Auswahl vier exemplarischer Services, die im Rahmen dieses Kapitels in einer Testumgebung migriert und getestet werden. Die Testergebnisse werden anschließend zusammengefasst und analysiert.

In Kapitel 4 erfolgt, auf Grundlage der Testergebnisse aus Kapitel 3, der Leitfaden zur Servicemigration.

Abschließend werden die Ergebnisse dieser Arbeit in Kapitel 5 zusammengefasst dargestellt, gefolgt von einem Ausblick auf weitere Aspekte der Servicemigration in einer Dual-Stack-Umgebung.

2 Grundlagen

Im folgenden Kapitel werden die Grundlagen erläutert, die für das bessere Verständnis dieser Arbeit notwendig sind. Zunächst wird der Begriff Service definiert und die in Kapitel 3 migrierten Services vorgestellt. Darauf folgend wird die Socketkommunikation in Bezug auf beide IP-Protokolle beschrieben. Abschließend werden notwendige Grundlagen, bezüglich wichtiger Aspekte der IPv6-Adressierung, erläutert. Grundsätzliche Informationen zu IPv6 und dessen Funktionsweise sind im RFC 2460 - *"Internet Protocol, Version 6 (IPv6) Specification"* [DH98] spezifiziert. Für weitere Informationen bezüglich Dual-Stack wird [Hag06] und [Hag09] empfohlen.

2.1 Services

Im weiteren Verlauf werden die Grundlagen, der im Rahmen dieser Arbeit migrierten Services, erläutert. Zunächst jedoch wird der Begriff *Service* zur weiteren Verwendung definiert. Prof. Dr. Paul Levi und Prof. Dr. Ulrich Rembold definieren den Begriff Service folgendermaßen:

"Ein Dienst (Service) ist eine Gruppe von zusammengehörenden, standardisierten Funktionen (Dienstprimitiven), die in sehr ähnlicher Form immer wieder von Dienstnehmern (Clients) angefordert und durch spezialisierte Dienstgeber (Server) erbracht wird. Die Qualität (Güte) eines Dienstes wird gemessen an seinen Kosten, seiner Antwortzeit, seiner Zuverlässigkeit und seiner Sicherheit (Security)" [LR03, S. 463].

Prof. Dr. Andrew S. Tanenbaum und Prof. Dr. David J. Wetherall, gliedern einen Service zudem in verbindungsorientiert und verbindungslos:

"Connection-oriented service is modeled after the telephone system. To talk to someone, you pick up the phone, dial the number, talk, and then hang up. Similiary, to use a connection-oriented network service, the service user first establishes a connection, uses the connection, and then releaes a connection" [TW11, S. 57].

"In contrast to connection-oriented service, connectionless service is modeled after the postal system. Each message (letter) carries the full destination address, and each one is routed through the intermediate nodes inside the system independent of

all the subsequent messages“ [TW11, S. 57f.].

Im weiteren Verlauf der Arbeit wird eine Kombination der oben verwendeten Definitionen für den Begriff Service verwendet. Ein Service ist ein Dienst der im Netzwerk eine für ihn spezifizierte Aufgabe erfüllt und verbindungsorientiert (über TCP) oder verbindungslos (über UDP) arbeitet.

Dynamic Host Configuration Protocol (DHCP)

DHCP ist ein Protokoll zur Autokonfiguration von Clients und ist im RFC 2131 - *“Dynamic Host Configuration Protocol“* spezifiziert [Dro97]. Über DHCP werden IP-Adressen und weitere Netzparameter wie DNS-Serveradressen, Default-Route etc. verteilt. Abbildung 2.1 stellt den Kommunikationsablauf von DHCP grafisch dar.

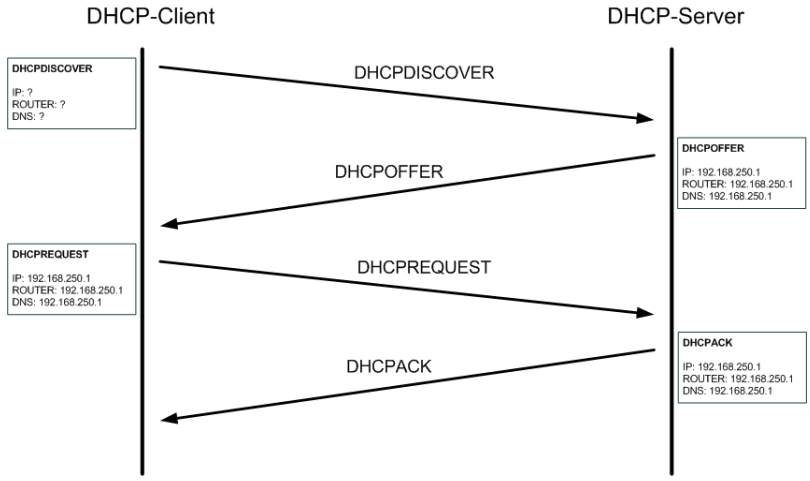


Abbildung 2.1: DHCP-Kommunikationsablauf (vgl. [Ago07, S. 20])

Über einen Broadcast sendet der Client ein DHCPDISCOVER an alle DHCP-Server im selben Netz. Antwortet ein DHCP-Server mit einer Netzkonfiguration (DHCPOFFER), kann der Client die Konfiguration durch einen DHCPREQUEST annehmen. Abschließend quittiert der DHCP-Server die Adresszuweisung durch ein DHCPACK an den Client [Ago07, S. 17ff.].

Domain Name System

Bei DNS handelt es sich um ein Protokoll, welches einen Service zur Namensauflösung bereitstellt. Dabei werden Domainnamen, wie *google.de*, in IP-Adressen aufgelöst und umgekehrt. Die IP-Adressen werden auf einem DNS-Server in sogenannten Resource Records gespeichert. Die Anfrage erfolgt über einen Resolver, welcher von der Anwendung den aufzulösenden Domainamen erhält. Ist die IP-Adresse nicht bereits im

Cache vorhanden, erfolgt eine Anfrage an den DNS-Server [ML05, S. 153]. Abbildung 2.2 zeigt die DNS-Architektur. DNS wird in den RFC's 1034 - *"DOMAIN NAMES - CONCEPTS AND FACILITIES"* [Moc87a] und 1035 - *"DOMAIN NAMES - IMPLEMENTATION AND SPECIFICATION"* [Moc87b] spezifiziert.

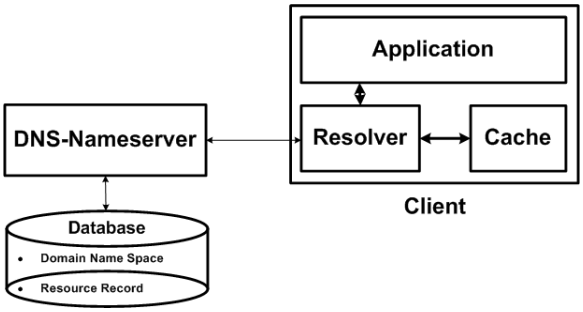


Abbildung 2.2: DNS-Architektur (vgl. [ML05, S. 153])

Hypertext Transfer Protocol (HTTP)

Bei HTTP handelt es sich um ein Anfrage-Antwort-Protokoll, mit dessen Hilfe Daten zwischen Webservern und -clients übertragen werden (s. Abbildung 2.3). Die Header des Protokolls werden dabei im *American Standard Code for Information Interchange*, kurz ASCII, Format angegeben. HTTP ist im RFC 2616 - *"Hypertext Transfer Protocol – HTTP/1.1"* spezifiziert [FGM+99].

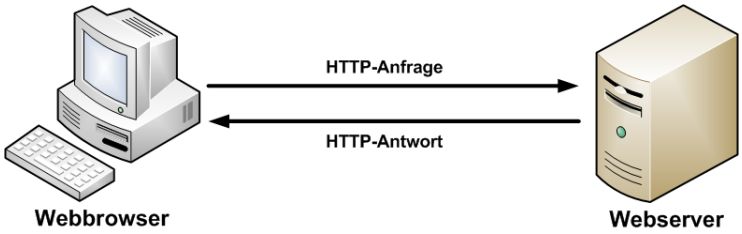


Abbildung 2.3: Austausch von Webinhalten (vgl. [TW12, S. 737])

File Transfer Protocol (FTP)

Das File Transfer Protocol ist ein häufig genutztes Netzwerkprotokoll, welches zum Austausch von Programmen und Dateien zwischen Rechnersystemen, über IP-Netzwerke, verwendet wird. FTP wird unter anderem zur Übertragung von Dateien auf einen Webserver genutzt. Spezifiziert wird FTP im RFC 959 - *"FILE TRANSFER PROTOCOL (FTP)"* [PR85]. FTP nutzt zur Übertragung der Kommandos einen separaten Kanal. Somit sind Steuerungsdaten und Übertragungsdaten voneinander getrennt. Zur Übertragung von Daten wird zwischen einem aktiven und passiven Modus un-

terschieden. Bei einer aktiven FTP-Verbindung wird der Verbindungsaufbau für den Datenkanal serverseitig initiiert (s. Abbildung 2.4). Die aktive Verbindung wird durch den FTP-Befehl *PORT* eingeleitet. Dem Port-Befehl werden eine IPv4-Adresse und eine Portnummer als Argument übergeben [PR85].

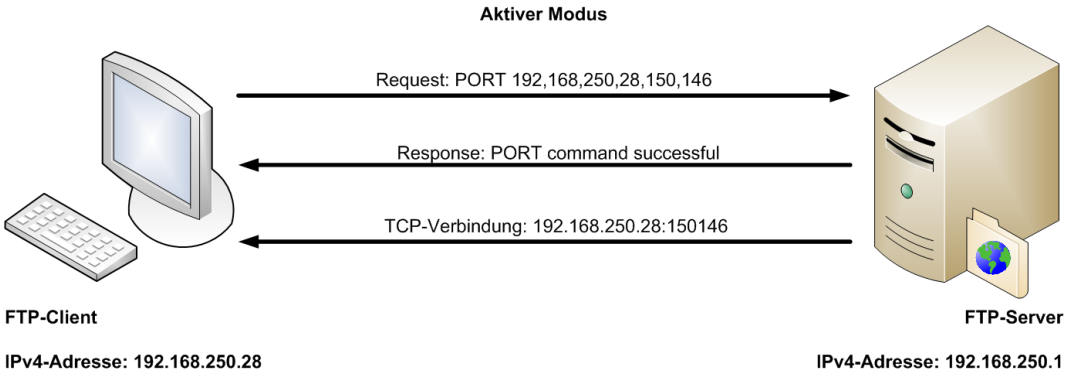


Abbildung 2.4: Aktive Verbindung durch Nutzung des PORT-Kommandos

Hingegen wird bei einer passiven Verbindung der Client lediglich informiert, über welchen Zielport er den Datenkanal zum Server aufbauen soll (s. Abbildung 2.5). Die passive Verbindung wird durch den Befehl *PASV* eingeleitet. Der Server antwortet mit einer IPv4-Adresse und dem entsprechenden Zielport [PR85].

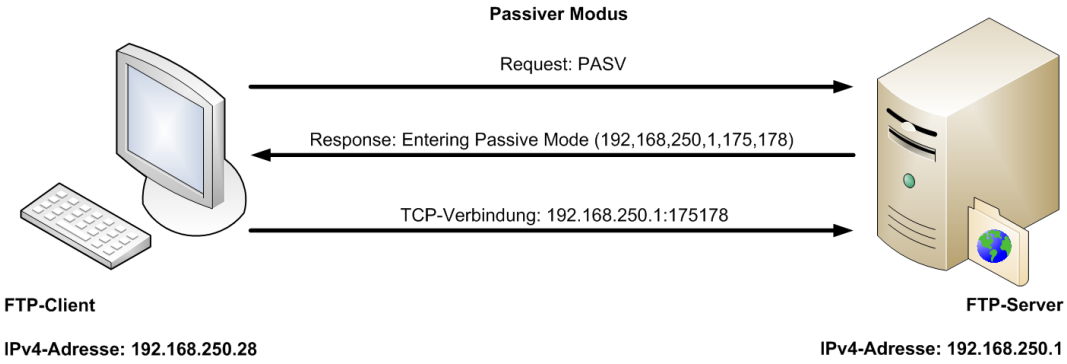


Abbildung 2.5: Passive Verbindung durch Nutzung des PASV-Kommandos

2.2 Sockets

Für die Kommunikation zwischen Anwendungsprozessen, welche sich nicht auf dem selben Host befinden, werden client- und serverseitig Endpunkte eingerichtet. Diese Endpunkte werden Sockets genannt und bestehen aus einer IP-Adresse des Hosts und einem Port [TW12, S. 629]. Über den Socket identifiziert der Server den Service und stellt fest, an welchen Anwendungsprozess er die Daten weiterreichen muss [TW12, S. 581]. Die Nutzung von Sockets wird über ein *Application Programming Interface (API)* realisiert, welches dem Programmierer vom Betriebssystem bereitgestellt wird

[PD03, S. 31]. Abbildung 2.6 stellt den internen Kommunikatonsablauf zweier Anwendungsprozesse über Sockets dar.

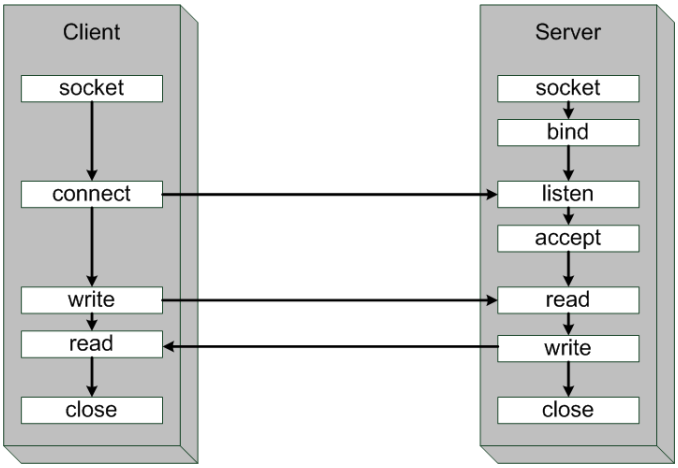


Abbildung 2.6: Socketkommunikation (vgl. [Olb03, S. 9])

Tabbelle 2.1 stellt dabei die Funktionen der einzelnen Systemaufrufe über die API auf dem Server und Client dar.

Systemaufrufe	Client	Server
socket	Erzeugen eines freien Sockets	Erzeugen eines freien Sockets
bind	-	Zuordnen einer IP-Adresse
connect	Verbindungsanfrage an Server	-
listen	-	Warten auf Verbindungsanfragen
accept	-	Annahme und Bestätigung der Verbindung
write	Daten an Server senden	Daten an Client senden
read	Daten vom Server empfangen	Daten vom Client empfangen
close	Schließen des Sockets	Schließen des Sockets

Tabelle 2.1: Socket-Systemaufrufe (vgl. [Olb03, S. 10])

Beim Systemaufruf *socket* wird die Protokollfamilie, des zu erstellenden Sockets

angegeben. Zur Verwendung von IPv4 wird dafür *AF_INET* angegeben, bei IPv6 wird dafür eine neue Protokollfamilie, namens *AF_INET6*, spezifiziert. IPv4 und IPv6 verfügen somit über einen jeweils eigenen Portbereich. Bei beiden IP-Protkollen wird bei den Portbereichen zwischen UDP- und TCP-Verbindungen unterschieden, somit sind im Dual-Stack-Betrieb 4 Portbereiche vorhanden (s. Tabelle 2.2).

IP-Version	Protokollfamilie	Transportprotokoll
IPv4	AF_INET	UDP
IPv4	AF_INET	TCP
IPv6	AF_INET6	UDP
IPv6	AF_INET6	TCP

Tabelle 2.2: Portbereiche

Zur Bereitstellung von IPv6-Sockets sind im RFC 2553 *"Basic Socket Interface Extensions for IPv6"*, Erweiterungen der API spezifiziert. Diese API-Erweiterungen stellen die Grundfunktionalitäten für IPv6, welche für TCP- und UDP-Anwendungen notwendig sind, bereit [GTBS99]. Erweiterte Funktionen der API für IPv6-Sockets, wie beispielsweise der Zugriff auf ICMPv6-Funktionen sind im RFC 2292 *"Advanced Sockets API for IPv6"* definiert [ST98]. RFC 2553 stellt unter anderem die Funktion zur Nutzung von IPv4-mapped IPv6-Adressen bereit, welche in Kapitel 2.3 vorgestellt werden.

2.3 IPv6-Adressen

Im folgenden werden für diese Arbeit zwei Aspekte im Bezug auf IPv6-Adressen erläutert. Dabei wird nicht auf die Funktionsweise, sowie der grundlegende Aufbau von IPv6, eingegangen. Für Informationen dieser Art wird [Hag09] empfohlen.

Syntax Problem

In der Regel wird die Kontaktaufnahme einer Anwendung zu einem Server, welcher einen Service bereitstellt, über Domainnamen realisiert. So wird beipielsweise in einem Browser der Domainname *netlab.inf.h-brs.de* angegeben, um den Webserver der Netzlabore der Hochschule Bonn-Rhein-Sieg zu kontaktieren. Wird jedoch, anstelle eines Domainnamens, eine explizite IPv6-Adresse angegeben, so muss sichergestellt sein dass die Anwendung in der Lage ist diese als IPv6-Adresse zu interpretieren. In vielen Anwendungen wird nur ein Doppelpunkt als Separator zwischen IP-Adresse und Portnummer verwendet, um daraufhin den Service adressieren zu können. Dies kann bei IPv6-Adressen jedoch zu Fehlern führen. Tabelle 2.3 zeigt die korrekte Adressierung eines IPv4-Services, jedoch die fehlerhafte Adressierung eines

IPv6-Services.

IP-Version	IP-Adresse	Port	Socket
IPv4	192.168.250.1	21	192.168.250.1:21
IPv6	2001:638:408:250::1	21	2001:638:408:250::1:21

Tabelle 2.3: Fehlerhafte Adressierung eines IPv6-Services

Aufgrund der Notation der IPv6-Adresse, bei der die Blöcke durch Doppelpunkte getrennt sind, ist die Anwendung nicht in der Lage zu interpretieren, welcher Teil nun zur IPv6-Adresse und Portnummer gehört. Ein erster Ansatz könnte sein, dass der letzte Block einfach als Portnummer interpretiert wird. Aber gerade durch die bei IPv6 übliche, abkürzende Schreibweise bei denen aufeinanderfolgende Null-Blöcke durch zwei Doppelpunkte ersetzt werden, ist die Anwendung so nicht in der Lage zu interpretieren, wann der letzte Block und somit die Portnummer beginnt. Im RFC 2732 - *"Format for Literal IPv6 Addresses in URL's"* wird spezifiziert, wie eine IPv6-Adresse in einem *Uniform Resource Locator (URL)* korrekt angegeben wird [HCM99]. Abhilfe schafft die Nutzung von eckigen Klammern, in denen die IPv6-Adresse eingeschlossen ist. Tabelle 2.4 zeigt beispielhaft den zuvor beschriebenen Sachverhalt. Somit ist die Anwendung in der Lage die IPv6-Adresse zu interpretieren und den Service zu adressieren. Eine weitere Fehlerquelle ist, das manche Anwendungen die IPv6-Adresse als Hostnamen interpretieren. Das Resultat ist eine Weiterleitung des für die Anwendung vermeintlichen Hostnamen an einen DNS-Server. Dieser antwortet mit einer Fehlermeldung und die Anwendung bricht den Verbindungsversuch ab.

IP-Adresse	Port	Socket
[2001:638:408:250::1]	21	[2001:638:408:250::1]:21

Tabelle 2.4: Korrekte Adressierung eines IPv6-Services

IPv4-mapped IPv6-Adressen

Bei IPv4-mapped IPv6-Adressen handelt es sich um einen IPv6-Adresstyp, welcher im RFC 4291 - *"IP Version 6 Addressing Architecture"* [DH06] spezifiziert ist. Mit Hilfe dieses Adresstyps können IPv4-Adressen als IPv6-Adressen repräsentiert werden. Dadurch können eingehende IPv4-Verbindungen durch IPv6-Sockets bearbeitet werden, da durch das Betriebssystem die IPv4-Adressen zuvor intern gemapped werden. Abbildung 2.7 zeigt den Aufbau einer solchen IPv6-Adresse. Die ersten 80 Bits werden durch Nullen repräsentiert, gefolgt von *FFFF* (16 Bit) und der IPv4-

Adresse (32-Bit) [Hag06, S. 45].

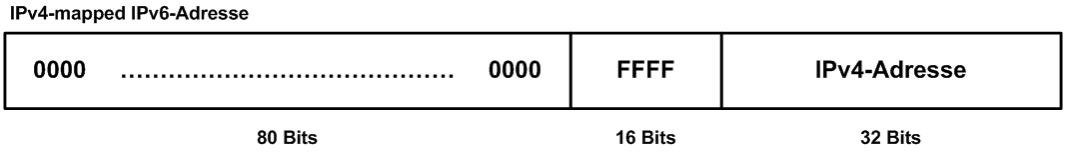


Abbildung 2.7: Aufbau einer IPv4-mapped IPv6-Adresse

3 Services im Dual-Stack-Betrieb

In diesem Kapitel werden zunächst vier Services ausgewählt, die im weiteren Verlauf des Kapitels auf ihre Dual-Stack-Fähigkeit überprüft werden. Dazu erfolgt zunächst eine Beschreibung der Vorgehensweise zur Analyse der Services. Anschließend wird die Testumgebung vorgestellt, in der die Services migriert werden. Darauf folgend werden für jeden Service server- und clientseitige Implementierungen auf den zuvor vorgestellten Betriebssystemen installiert, konfiguriert und getestet. Dabei wird geprüft ob ein Service die gleiche Funktionalität, die für IPv4 bereitgestellt ist, auch für IPv6 gewährleistet. Aufgrund des hohen Umfangs der Arbeit, wird ausschließlich auf die Konfiguration für den Dual-Stack-Betrieb eingegangen. Für die grundlegende Konfiguration des jeweiligen Services, wird auf weitere Literatur verwiesen. Im Anschluss daran erfolgt für jeden Service eine Zusammenfassung, in der alle getesteten Implementierungen und deren Ergebnisse aufgelistet sind.

3.1 Auswahl der Services

Um eine aussagekräftige Auswahl zu treffen, wurden Services ausgewählt, die in Bezug auf ihre Dual-Stack-Erweiterungen/Änderungen unterschiedlich sind. Dies bedeutet, dass beispielsweise der Service DHCP für IPv6 anders angepasst wurde, als FTP. Dadurch ergeben sich womöglich andere Maßnahmen für den Leitfaden. Folgende vier Services wurden im Rahmen dieser Arbeit migriert und getestet:

- DHCP - Dynamic Host Configuration Protocol
- DNS - Domain Name System
- HTTP - Hypertext Transfer Protocol
- FTP - File Transfer Protocol

Bei den ausgewählten Services handelt es sich um jeweils zwei UDP- und zwei TCP-Services. Welche Auswirkungen diese beiden Transportprotokolle haben, wird im späteren Verlauf dieser Arbeit diskutiert. Mit DHCP und DNS wurden zudem zwei Services ausgewählt, welche die Grundfunktionalität eines Netzes bereitstellen. HTTP ist aus unternehmerischer Sicht ein bedeutender Service, da über einen Webserver die Leistungen eines Unternehmens weltweit publiziert werden. Um potentielle Kunden aus unterschiedlichen Ländern zu erreichen, deren Infrastruktur jeweils auf

IPv6 basiert, ist die Migration von HTTP unumgänglich. Mit FTP wurde ein Service gewählt, der aufgrund seiner Erweiterungen für IPv6 interessant ist.

3.2 Vorgehensweise zur Serviceanalyse

Um einen Service auf seine Dual-Stack-Tauglichkeit überprüfen zu können, wird der Service schrittweise analysiert.

Abbildung 3.1: Vorgehensweise zur Serviceanalyse

1. Analyse der Dual-Stack-Erweiterungen

Im ersten Schritt werden die Anpassungen des Services für den Dual-Stack-Betrieb beschrieben. Für jeden Service wird recherchiert, welche Änderungen am Protokoll erforderlich sind: Neukonzeption, Befehlserweiterung oder Anpassungen zur Adressierung des Services. Diese Änderungen sind in den jeweiligen RFC's spezifiziert. Um eine Aussage über die Dual-Stack-Fähgkeit eines Services oder einer Implementierung dieses Services zu treffen, ist es erforderlich die RFC's zu analysieren.

2. Auswahl der Server-/Clientimplementierungen

Im darauf folgenden Schritt, wird für die Betriebssysteme Ubuntu Server 12.04 und Windows Server 2012 jeweils eine Serverimplementierung eines Services ausgewählt. Es ist dabei durchaus möglich, dass eine serverseitige Implementierung eines Services Dual-Stack-fähig ist, eine Clientimplementierung jedoch nicht in der Lage ist, den Service über beide Protokolle (IPv4/IPv6) zu nutzen. Um dies zu berücksichtigen,

werden für die im Kapitel 3.3 aufgelisteteten Betriebssysteme, jeweils die bekanntesten clientseitigen Implementierungen getestet. Die Auswahl der Implementierungen, erfolgt durch Empfehlungen aus Literatur, Verbreitung der Implementierung im produktiven Einsatz und/oder Zuspruch durch die verschiedenen Betriebssystem-Communities.

3. Installation und Konfiguration der Implementierungen

Daraufhin werden die server- und clientseitigen Implementierungen zunächst installiert und konfiguriert. Die Konfiguration, beschränkt sich dabei ausschließlich auf die Parameter, die für den Dual-Stack-Betrieb erforderlich sind. Aus administrativer und sicherheitstechnischer Sicht sollte ein Service nicht über alle auf dem Server verfügbaren Schnittstellen erreichbar sein. Deshalb wird die gezielte Addressbindung eines Services bei der Konfiguration beschrieben. Für die grundlegende Konfiguration eines Services, wird auf weitere Literatur verwiesen.

4. Testen der Implementierungen

Nach Abschluss der Konfiguration werden die Serviceimplementierungen für den Dual-Stack-Betrieb getestet. Dabei werden die Serviceverfügbarkeit und die Servicefunktionalität getestet. Bei der Serviceverfügbarkeit wird geprüft, ob der Service im Netzwerk erreichbar ist. Dazu werden auf dem Server mit Hilfe von Tools die notwendigen Sockets geprüft, auf denen der Service lauscht. Um diesen Zustand zu diagnostizieren, wird unter Ubuntu Server 12.04, sowie Windows Server 2012, das Tool *netstat* verwendet. Bei netstat handelt es sich um ein Diagnose-Kommandozeilenprogramm, welches Informationen über die im System vorhandenen Netzwerkschnittstellen bereitstellt. Netstat ist standardmäßig unter Ubuntu Server 12.04 und Windows Server 2012 installiert. Bei der Servicefunktionalität wird überprüft ob die gewünschten Funktionalitäten des Services für beide IP-Protokolle erbracht werden. Beispielsweise muss ein DNS-Server im Dual-Stack-Betrieb in der Lage sein auf DNS-Anfragen mit A und AAAA Records, in denen die IP-Adressen gespeichert sind, zu antworten. Dabei ist die Servicefunktionalität abhängig vom jeweiligen Service. Wichtig ist, dass in dieser Arbeit lediglich die Grundfunktionaliät eines Services getestet wird. Es wird also getestet, ob die selben Funktionalitäten die für IPv4 gegeben sind, auch für IPv6 erbracht werden.

5. Bewertung der Implementierungen

Zum Schluss erfolgt, auf Grundlage der Parameter Servceverfügbarkeit und -funktionalität, eine Bewertung der Implementierungen und eine abschließende Zusammenfassung der Ergebnisse. Abhängig vom Service, werden Besonderheiten im Hinblick

auf die Administration des Services beschrieben.

3.3 Testumgebung

In diesem Kapitel wird die Testumgebung vorgestellt und grafisch dargestellt. Damit die Services für den Dual-Stack-Betrieb getestet werden können, wird eine Vielzahl unterschiedlicher Betriebssysteme und Anwendungen benötigt. Serverseitig werden diese Betriebssysteme eingesetzt:

- Ubuntu Server 12.04
- Windows Server 2012

Ubuntu ist eine beliebte Linux-Distribution, welche auf das umfangreiche Debian-Repository zurückgreift und in beiden Netzlaboren zum Einsatz kommt. Bei Windows Server 2012 handelt es sich um die zum jetzigen Stand aktuellste Windows Server Version.

Clientseitig werden die folgenden Betriebssysteme in ihrer aktuellen Version verwendet:

- Android 2.3.6
- iOS 6.1
- Mac OS X 10.8
- Ubuntu Desktop 12.04
- Windows XP
- Windows 7
- Windows 8

Android ist ein von der Open Handset Alliance entwickeltes Betriebssystem für mobile Geräte (Smartphones, Tablets, etc.). Android 2.3.6 ist mit 45,6% die am häufigsten verbreiteste Android Version [Sta13]. Bei iOS handelt es sich um ein von Apple entwickeltes Betriebssystem, für deren eigene Produkte wie iPAD oder iPhone. Apples Mac OS X hingegen ist ein Betriebssystem für Mac-Computer wie dem iMac oder der MacBook-Reihe. Daneben handelt es sich bei Ubuntu Desktop 12.04 um eine weit verbreitete Linux-Distribution mit grafischer Benutzeroberfläche für Clients. Zum Schluss werden Microsofts Desktopbetriebssysteme Windows XP, Windows 7 und Windows 8 untersucht. Da die Anwendungen je nach Service und Betriebssystem variieren, werden diese in den Kapiteln der jeweiligen Services vorgestellt. Zur Durchführung der Tests wird ein Testrechner verwendet, welcher die in den jeweiligen Kapiteln notwendigen Tools bereithält. Die beiden Server verfügen über eine

statische IPv4-/IPv6-Adressierung. Die restlichen Hosts, erhalten ihre Parameter für die Netzwekkonfiguration über DHCPv4/v6. Abbildung 3.2 stellt die Testumgebung grafisch dar.

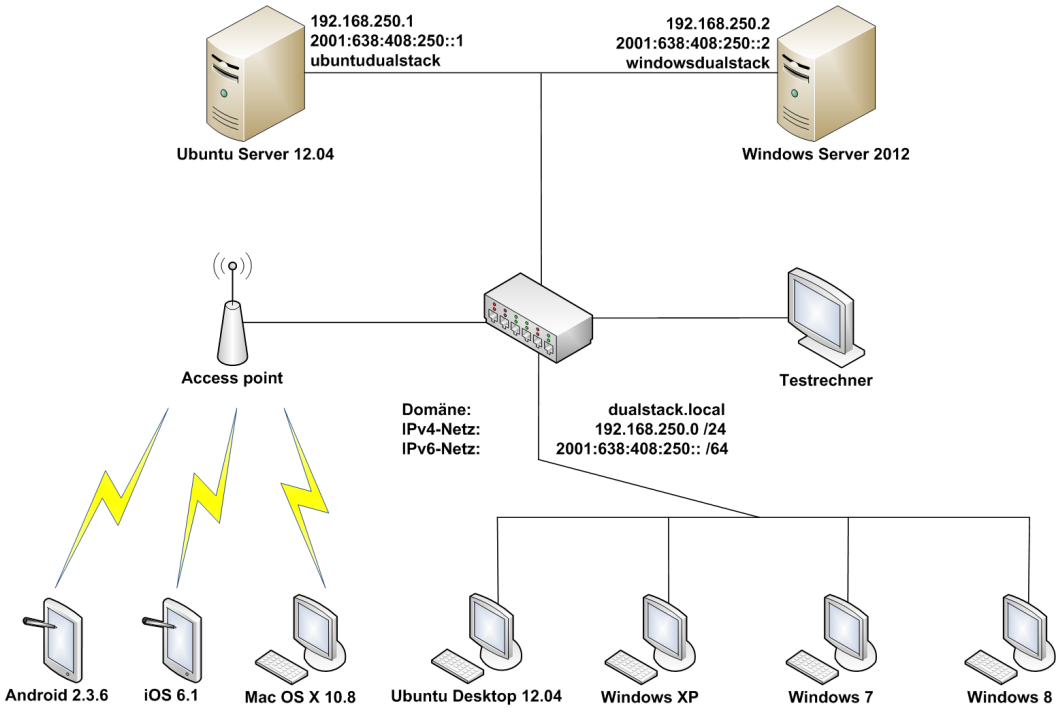


Abbildung 3.2: Testumgebung

3.4 Dynamic Host Configuration Protocol (DHCP)

Zur Autokonfiguration von IPv6-Clients wurde das Protokoll DHCP neu spezifiziert und wird zur Abgrenzung, als DHCPv6 bezeichnet. DHCP verwendet im Header 32-Bit große Felder für IPv4-Adressen, wodurch IPv6-Adressen nicht genutzt werden können. Abbildung 3.3 zeigt den Aufbau des Headers von DHCP. Die Felder *ciaddr, yiaddr, siaddr* und *giaddr* enthalten die IPv4-Adressen [Dro97].

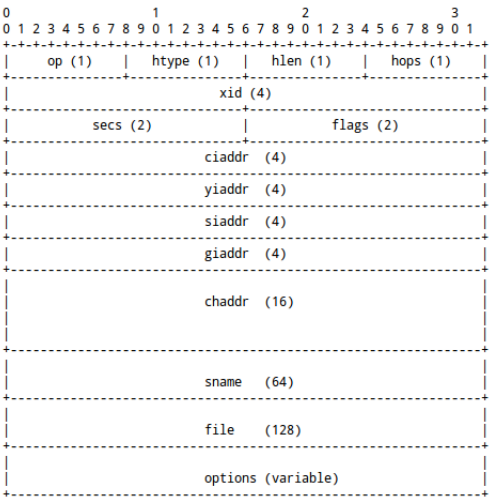


Abbildung 3.3: Aufbau des DHCP-Headers [Dro97]

3.4.1 Dual-Stack-Erweiterungen

Im RFC 3315 - *"Dynamic Host Configuration Protocol for IPv6 (DHCPv6)"* wird DHCP für IPv6 neu spezifiziert [RDBV+03]. Im Gegensatz zu seinem Pendant DHCP, kann DHCPv6 nicht alleine für die Autokonfiguration von IPv6-Clients verwendet werden. Der Grund dafür ist, dass im derzeitigen RFC 3315 nicht die Verteilung der Default-Route spezifiziert ist. Die Default-Route wird über sogenannte Router-Advertisements im Netzwerk verteilt. Router-Advertisements werden von einem Router periodisch, oder auf Anfrage eines Clients, über die Multicast-Adresse *FF02::1 (All-Nodes)* an alle Hosts, die sich am selben Link befinden, verteilt. Router-Advertisements regeln zudem, über welche Art der Autokonfiguration ein Host seine Netzwerkkonfiguration beziehen soll. Dabei wird zwischen folgenden beiden Methoden unterschieden [Hag09, S. 324]

- Stateless Address Autoconfiguration (SLAAC)
- Stateful Address Autoconfiguration (DHCPv6)

SLAAC wurde im RFC 4862 - *"IPv6 Stateless Address Autoconfiguration"* von der Internet Engineering Task Force (IETF) spezifiziert [TNJ07]. Bei SLAAC wird für die Autokonfiguration von Hosts kein DHCP benötigt. Über Router-Advertisements werden IPv6-Präfixe an alle Hosts am selben Link versendet. Mit Hilfe des Präfixes und des vom Host selbst generierten Hostteils wird die IPv6-Adresse erstellt. Der Hostanteil der IPv6-Adresse wird entweder zufällig (Privacy Extension) oder mit Hilfe der MAC-Adresse nach dem EUI-64-Verfahren, vom Host generiert. Durch die im RFC 6106 - *"IPv6 Router Advertisement Options for DNS Configuration"* spezifizierte Option *Recursive DNS Server (RDNSS)*, kann zudem die IPv6-Adresse eines DNS-Servers über Router-Advertisements verteilt werden [JPBM10]. Die Stateful Address Autoconfiguration entspricht dem DHCPv6. Analog zu DHCP werden über DHCPv6 alle Parameter, bis auf die Defaul-Route, verteilt. Des Weiteren ist eine hybride Kombination der beiden Autokonfigurationsmethoden möglich. Über SLAAC werden dazu beipielsweise die IPv6-Präfixe und der DNS-Server verteilt. Weitere Parameter, wie ein NTP-Server, werden über DHCPv6 bezogen [Hag09, S. 324]. Damit ein Host eine der Autokonfigurationsmethoden einleitet, werden in den Router-Advertisements folgende Flags gesetzt:

- Das M-Flag (*Managed Address Configuration*) sorgt dafür, das ein Host seinen DHCPv6-Client startet und seine Netzwerkkonfiguration über DHCPv6 bezieht
- Das O-Flag (*Other Stateful Configuration*) sorgt dafür, das alle Parameter bis auf die Adresszuweisung über DHCPv6 bezogen werden (hybride Methode)

Ist das M-Flag nicht gesetzt, erfolgt die Autokonfiguration über SLAAC [Hag09, S. 104]. Abbildung 3.4 veranschaulicht den Kommunikationsablauf zur IPv6-Autokonfiguration.

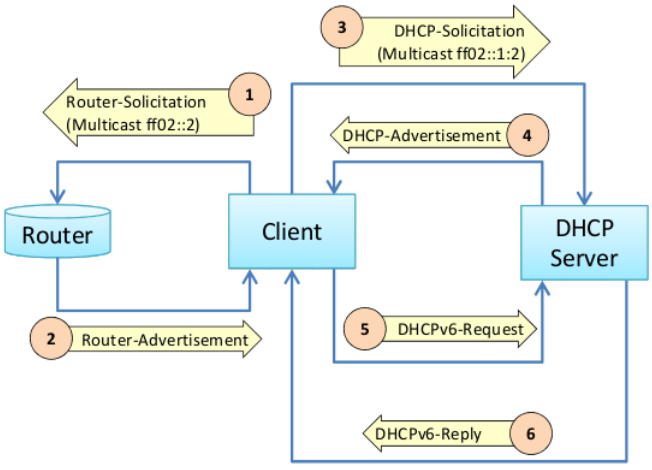


Abbildung 3.4: Kommunikationsablauf der IPv6-Autokonfiguration (vgl. [SKL12, S. 73])

1. Zunächst sendet der Client eine Router-Solicitation Nachricht an die Multicast-Adresse *FF02::2*.

2. Als Reaktion darauf, antwortet der Router mit einem Router-Adverstisement.

3. Abhängig von den gesetzten Flags im Router-Advertisement, sendet der Client einen DHCP-Solicitation Nachricht an die Multicast-Adresse *FF02:1:2 (DHCP-v6-Server)*.

4. Alle am Link vorhandenen DHCPv6-Server antworten mit einem DHCP-Advertisement. Diese Nachricht enthält die Parameter zur Netzkonfiguration.

5. Daraufhin wählt der Client zwischen den angebotenen Netzwerkkonfigurationen eine aus und antwortet dem jeweiligen DHCPv6-Server.

6. Zuletzt speichert der DHCPv6-Server seine Zuweisung unter einer Client-ID ab und schickt einen DHCPv6-Reply [SKL12, S. 72].

3.4.2 Serverseitige Implementierungen

Im Folgenden werden die beiden Serverbetriebssysteme Ubuntu Server 12.04 und Windows Server 2012 für die IPv6-Autokonfiguration vorbereitet und anschließend getestet. Für beide Betriebssysteme wird ein Router-Advertisement-Dienst, sowie eine DHCPv6-Implementierung benötigt. Unter Ubuntu erledigt die Aufgabe des Router-Advertisement-Dienstes der *Router-Advertisement Daemon (radvd)* und die Referenzimplementierung *ISC-DHCP-Server* den DHCPv6-Dienst. Auf dem Windows Server 2012, werden die bereits integrierten Dienste im Server-Manager von Windows verwendet.

Radvd

Der Router-Advertisement Daemon befindet sich standardmäßig im Ubuntu-Repository und wird in der Version 1.8.3 installiert, konfiguriert und getestet.

Installation und Konfiguration

Die Installation des Router-Advertisement-Dienstes wird über folgenden Befehl realisiert:

```
root@ubuntudualstack:~# apt-get install radvd
```

Abbildung 3.5: Radvd - Installation

In der Datei */etc/radvd.conf* werden alle notwendigen Konfigurationen vorgenommen. Soll die Autokonfiguration über SLAAC stattfinden, ohne weitere Parameter über DHCPv6 zu beziehen, so wird der radvd wie in Abbildung 3.6 konfiguriert. Mit Hilfe des Parameters *prefix* wird das zu verteilende IPv6-Präfix festgelegt. Über den Parameter *RDNSS* wird die IPv6-Adresse des DNS-Servers bekannt gegeben.

```
interface eth1 {
        AdvSendAdvert on;
        prefix 2001:638:408:250::/64 {};
        RDNSS 2001:638:408:250::1 {};
};
```

Abbildung 3.6: Radvd - Konfiguration über SLAAC ohne weitere Parameter

Sollen weitere Parameter (z.B. NTP-Server) über DHCPv6 bezogen werden, so wird die Konfiguration um den Parameter *AdvOtherConfigFlag on* ergänzt (s. Abbildung 3.7). Dadurch wird das O-Flag im Router-Advertisement gesetzt.

```
interface eth1 {
        AdvSendAdvert on;
        AdvOtherConfigFlag on;
        prefix 2001:0638:0408:0250::/64 {};
        RDNSS 2001:638:408:250::1 {};
};
```

Abbildung 3.7: Radvd - Konfiguration über SLAAC mit weiteren Parametern über DHCPv6

Zur Konfiguration über DHCPv6 (Stateful), werden die Parameter *prefix* und *AdvOtherConfigFlag on* entfernt. An dessen Stelle wird der Parameter *AdvManagedFlag on* plaziert (s. Abbildung 3.8. Hiermit wird das M-Flag im Router-Advertisement gesetzt. Nach einem Neustart des radvd mit Hilfe des Befehls */etc/init.d/radvd restart*, wird die Konfiguration wirksam.

```
interface eth1 {
        AdvSendAdvert on;
        AdvManagedFlag on;
};
```

Abbildung 3.8: Radvd - Konfiguration über DHCPv6 einleiten

Testphase

Zur Überprüfung des Router-Advertisement-Dienstes wird das Tool *rdisc6* verwendet. Rdisc6 ist in der Lage Router-Solicitation-Nachrichten zu erzeugen, die über die

Multicast-Adresse *FF02::2* an alle Router gesendet werden. Antwortet ein Router mit einem Router-Advertisement, interpretiert rdisc6 die Nachricht und filtert die wichtigsten Informationen für den Benutzer. Abbildung 3.9 zeigt die Nutzung, sowie das Ergebnis von rdisc6, für den zuvor konfigurierten Router-Advertisement-Dienstes. Anhand des Ergebnisses ist zu erkennen, dass das Präfix *2001:638:408:250::/64* und die IPv6-Adresse des DNS-Servers verteilt wird. Zudem ist das O-Flag gesetzt. Der Radvd unter Ubuntu Server 12.04 ist somit für die IPv6-Autokonfiguration geeignet.

```
andreas@andreas-laptop:~$ rdisc6 eth0
ff02::2 wird angeboten (ff02::2) auf eth0 ...

Sprungbegrenzung          :          64 (      0x40)
Zustandsorientierte Adresskonf. :    Nein
Andere zustandsorientierte Konf. :    Ja
Router-Einstellung        :       mittel
Router-Lebensdauer        :         1800 (0x00000708) Sekunden
Erreichbarkeitszeit       :  nicht spezifiziert (0x00000000)
Rückübertragungszeit      :  nicht spezifiziert (0x00000000)
 Präfix                   : 2001:638:408:250::/64
  Gültigkeitszeit         :        86400 (0x00015180) Sekunden
  Wunschzeit              :        14400 (0x00003840) Sekunden
 Rekursiver DNS-Server    : 2001:638:408:250::1
  DNS-Server-Lebensdauer  :          600 (0x00000258) Sekunden
 Quell-Verknüpfungsebenen-Adresse: 00:1B:21:2B:13:A8
 von  fe80::21b:21ff:fe2b:13a8
```

Abbildung 3.9: Rdisc6 - Überprüfen des Router-Advertisement-Dienstes

ISC-DHCP-Server

Der ISC-DHCP-Server befindet sich standardmäßig im Ubuntu-Repository und wird in der Version 4.1.ESV-R4-0 installiert, konfiguriert und getestet.

Installation und Konfiguration

Der DHCP-Server wird über folgenden Befehl installiert:

```
root@ubuntudualstack:~# apt-get install isc-dhcp-server
```

Abbildung 3.10: ISC-DHCP-Server - Installation

In der Datei */etc/dhcp/dhcpd6.conf* werden alle notwendigen Konfigurationen vorgenommen (s. Abbildung 3.11). Durch den Parameter *range6* wird der IPv6-Adressbereich festgelegt, der über DHCPv6 verteilt werden soll. Die Parameter *option dhcp6.-name-servers* und *option dhcp6.sntp-servers* definieren die Adressen für DNS- und NTP-Server. Nach einem Neustart des Services über */etc/init.d/isc-dhcp-server6 restart*, sind die Konfigurationen wirksam. Für die Konfiguration von DHCPv4 wird

[Ubu13] empfohlen.

```
subnet6 2001:638:408:250::/64{
        range6 2001:638:408:250::100 2001:638:408:250:ffff:ffff:ffff:ffff;
        option dhcp6.name-servers 2001:638:408:250::1;
        option dhcp6.sntp-servers 2001:638:408:250::1;
        }
```

Abbildung 3.11: ISC-DHCP-Server - Konfiguration

Testphase

Im folgenden wird die DHCP-Serverimplementierung auf ihre Dual-Stack-Tauglichkeit getestet. Zunächst wird die Serviceverfügbarkeit diagnostiziert. Dabei wird sicher gestellt, dass der Service seinen Dienst über IPv4 und IPv6 im Netzwerk zur Verfügung stellt. Ein Test mit Hilfe von netstat zeigt, dass der DHCP-Service für beide IP-Protokolle verfügbar ist (s. Abbildung 3.12).

```
root@ubuntudualstack:~# netstat -lnp|egrep ':67|:547'
udp        0      0 0.0.0.0:67              0.0.0.0:*                           1103/dhcpd
udp6       0      0 :::547                  :::*                                24211/dhcpd
```

Abbildung 3.12: ISC-DHCP-Server - Serviceverfügbarkeit

Abschließend wird mit Hilfe des Tools *dhclient* die Servicefunktionalität des ISC-DHCP-Servers getestet. Abbildung 3.13 zeigt die erfolgreiche Netzwerkkonfiguration. Die Serverimplementierung ISC-DHCP-Server ist Dual-Stack-fähig.

```
root@andreas-laptop:/# dhclient -6 -s 2001:638:408:250::1
root@andreas-laptop:/# ifconfig
eth0      Link encap:Ethernet  Hardware Adresse c8:0a:a9:18:d9:7c
          inet Adresse:192.168.250.28  Bcast:192.168.250.255  Maske:255.255.255.0
          inet6-Adresse: 2001:638:408:250:f10c:a855:f892:bccd/64 Gültigkeitsbereich:Global
          inet6-Adresse: fe80::ca0a:a9ff:fe18:d97c/64 Gültigkeitsbereich:Verbindung
```

Abbildung 3.13: ISC-DHCP-Server - Servicefunktionalität

Microsoft Router-Advertiment-Dienst

Der Router-Advertisment-Dienst ist fester Bestandteil von Windows Server 2012. Eine Installation des Dienstes ist nicht erforderlich.

Konfiguration

Die Konfiguration erfolgt über die Powershell (s. Abbildung 3.14). Der Parameter *"advertise=enabled"* aktiviert den Dienst, wobei *"advertisedefaultroute=enabled"* die

Default-Route bekannt gibt. Mit Hilfe von *forwarding=enabled* wird dafür gesorgt, dass die Router Lifetime auf ungleich 0 gesetzt wird. Soll das M-Flag im Router-Advertisement gesetzt werden, so wird der Befehl um *"managed=enabled"* ergänzt. Analog dazu wird das O-Flag über den Parameter *"otherstateful=enabled"* gesetzt.

```
PS C:\> netsh interface ipv6 set interface "Ethernet 2" forwarding=enabled advertise=enabled advertisedefaultroute=enabled
```

Abbildung 3.14: Microsoft Radv-Dienst - Konfiguration des Dienstes

Zur Verteilung eines IPv6-Präfixes über Router-Advertisements, ist folgender Befehl notwendig:

```
PS C:\> netsh interface ipv6 set route 2001:638:408:250::/64 interface="Ethernet 2" publish=yes
```

Abbildung 3.15: Microsoft Radv-Dienst - Verteilen eines IPv6-Präfixes

Testphase

Ein Test zeigt, das der Router-Advertisment-Dienst seine Arbeit korrekt verrichtet (S. Abbildung 3.16).

```
andreas@andreas-laptop:~$ rdisc6 eth0
ff02::2 wird angeboten (ff02::2) auf eth0 ...

Sprungbegrenzung          :    undefiniert (      0x00)
Zustandsorientierte Adresskonf. :     Ja
Andere zustandsorientierte Konf. :    Ja
Router-Einstellung        :       mittel
Router-Lebensdauer        :         1800 (0x00000708) Sekunden
Erreichbarkeitszeit       :  nicht spezifiziert (0x00000000)
Rückübertragungszeit      :  nicht spezifiziert (0x00000000)
 Quell-Verknüpfungsebenen-Adresse: 00:1B:21:2B:13:A8
 MTU                      :         1500 Byte (gültig)
 Präfix                   : 2001:638:408:250::/64
  Gültigkeitszeit         :      2592000 (0x00278d00) Sekunden
  Wunschzeit              :       604800 (0x00093a80) Sekunden
 Route                    : 2001:638:408:250::/64
  Routen-Einstellung      :       mittel
  Routen-Lebensdauer      :     endlos (0xffffffff)
 von  fe80::4472:ca9e:635:4b2c
```

Abbildung 3.16: Rdisc6 - Überprüfen des Router-Advertisement-Dienstes

Microsoft DHCP Service

Nach erfolgreicher Inbetriebnahme des Router-Advertisement-Dienstes, erfolgt die Installtion des DHCP-Servers.

Installation und Konfiguration

Installieren lässt sich der DHCP-Server über den integrierten Server Manager. Dazu sind folgende Schritte erforderlich:

- Start –> Server-Manager –> Verwalten –> Rollen und Features hinzufügen
- Installationstyp: *Rollenbasierte oder featurebasierte Installation* auswählen
- Serverauswahl: Auswählen des Servers auf dem der Service verfügbar sein soll
- Serverrollen: DHCP-Server auswählen
- Features: keine Auswahl

Nach der Installation ist der DHCP-Server ohne Neustart aktiv. Die Konfiguration erfolgt über *Start –> Server-Manager –> Tools –> DHCP*. Über *«Server (Rechtsklick)» –> Bindungen hinzufügen/entfernen*, werden die IP-Adressen ausgewählt über die der Service verfügbar sein soll (s. Abbildung 3.17). Im Anschluss daran werden die Adressbereiche konfiguriert. Für die Konfiguration von DHCPv4 wird [Dav12, S. 491f.] empfohlen.

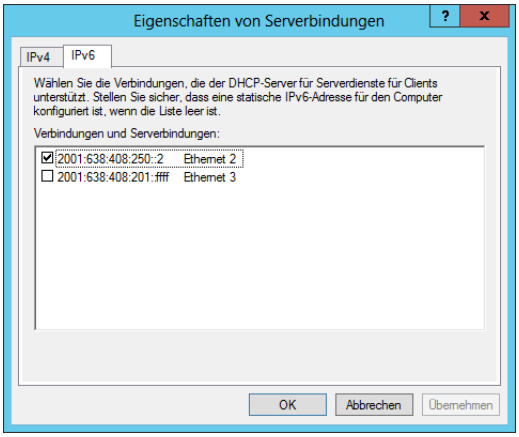


Abbildung 3.17: Microsoft DHCP Service - Binden der IP-Adressen

Über *IPv6 –> Neuer Bereich* wird die Konfiguration des IPv6-Adressbereiches eingeleitet. Nach Festlegung eines Namens (z.B. Privates Netz) für den IPv6-Adressbereich, erfolgt die die Eingabe des IPv6-Präfixes (s. Abbildung 3.18). Im nächsten Schritt können IPv6-Adressen angegeben werden, die nicht verteilt werden dürfen. Abschließend wird die Leasedauer festgelegt, woraufhin die Konfiguration aktiv wird.

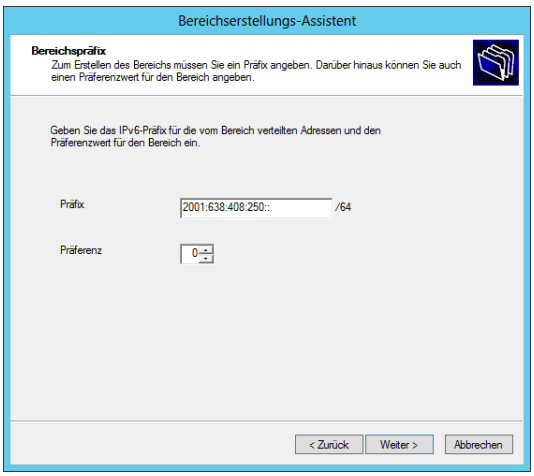


Abbildung 3.18: Microsoft DHCP Service - Festlegen des IPv6-Adressbereiches

Testphase

Ein Test mit Hilfe von netstat zeigt, dass der Service über beide IP-Protokolle verfügbar ist (s. Abbildung 3.19).

```
PS C:\Users\Administrator> netstat -an|findstr ":67 :547"
Aktive Verbindungen
  Proto  Lokale Adresse          Remoteadresse          Status
  UDP    192.168.250.2:67        *:*
  UDP    [2001:638:408:250::2]:547  *:*
```

Abbildung 3.19: Microsoft DHCP Service - Serviceverfügbarkeit

Ein weiterer Test zur Überprüfung der Funktionalität zeigt, dass der Microsoft DHCP-Service Dual-Stack-fähig ist (s. Abbildung 3.20).

```
root@andreas-laptop:/home/andreas# dhclient -6 -s 2001:638:408:250::2 && ifconfig
eth0      Link encap:Ethernet  Hardware Adresse c8:0a:a9:18:d9:7c
          inet Adresse:192.168.250.11  Bcast:192.168.250.255  Maske:255.255.255.0
          inet6-Adresse: 2001:638:408:250:f10c:a855:f892:bccd/64 Gültigkeitsbereich:Global
          inet6-Adresse: fe80::ca0a:a9ff:fe18:d97c/64 Gültigkeitsbereich:Verbindung
```

Abbildung 3.20: Microsoft DHCP Service - Servicefunktionalität

3.4.3 Clientseitige Implementierungen

Im Folgenden wird auf die Ergebnisse des Artikels *"IPv6-Autokonfiguration für Clients"* referenziert [SKL12]. Im Rahmen dieser Arbeit wurden, unter anderem, die gleichen Betriebssysteme getestet. Tabelle 3.1 zeigt die Ergebnisse.

	Windows XP	Windows 7	Windows 8	Ubuntu 12.04	Mac OS X 1.8	iOS 6.1	Android 2.3.6
IPv6-Support	Installation erforderlich	Ja	Ja	Ja	Ja	Ja	Ja
IPv6-Adresse über Router Advertise-ment	Ja	Ja	Ja	Ja	Ja	Ja	Geräte-abhängig
Default-Route über Router Advertise-ment	Ja	Ja	Ja	Ja	Ja	Ja	Ja
RDNSS-Option	Nein	externer Code erforderlich	externer Code erforderlich	Ja	Ja	Ja	Nein
Privacy Extensions	per Default	per Default	per Default	per Default	per Default	per Default	manuell
DHCPv6-Client-Support	Installation erforderlich	Ja	Ja	Ja	Ja	Ja	Nein
DHCPv6-Client über O-/M-Flag steuerbar	nein	Ja	Ja	Ja	Ja	Ja	Nein

Tabelle 3.1: Testergebnisse der IPv6-Clientautokonfiguration (vgl. [SKL12, S. 74])

Bis auf Windows XP, unterstützen alle getesteten Betriebssysteme IPv6 per Default. Eine Nachrüstung für Windows XP ist mit Hilfe des Befehls *netsh interface ipv6 install* möglich. Die IPv6-Adresse konnte über die IPv6-Präfixverteilung über Router-Advertisements von allen Betriebbsystemen generiert werden. Nach Angaben von [SKL12] ist dies für Android jedoch geräteabhängig. Auch die Verteilung der Default-Route über Router-Advertisments war bei allen Clients erfolgreich. Im Gegensatz zu dem im Artikel getesteten Android 4.03 wurde unter Android 2.3.6 die Default-Route eingetragen. Die Verteilung der DNS-Serveradresse über Router-Advertisments war lediglich unter Ubuntu 12.04, Mac OS X 1.8 und iOS 6.1 möglich. Windows 7/8 kann durch das externe Open-Source-Werkzeug *rdnssd-win32* nachgerüstet werden. Die Privacy Extensions werden mit Ausnahme von Android, bei allen Clients per Default generiert. Bis auf Windows XP und Android, ist bei allen Betriebssystemen ein DHCPv6-Client vorhanden, welcher sich über die Router-Advertisement-Flags steuern lässt. Unter Windows XP ist es möglich einen externen DHCPv6-Client, wie beispielsweise *dibbler*, zu installieren. Android verzichtet vollständig auf einen DHCPv6-Client [SKL12, S. 74ff.].

3.4.4 Zusammenfassung

In Tabelle 3.2 werden die Ergebnisse der DHCP-Serverimplementierungen dargestellt. Durch Kombination der Router-Advertisements und DHCPv6, wird Stateless und Stateful Address Autoconfiguration für IPv6 im vollen Umfang realisiert. Beide Implementierungen sind in der Lage die Autokonfiguration für den Dual-Stack-Betrieb zu gewährleisten.

DHCP-Serverimplementierung	Dual-Stack-Betrieb
ISC-DHCP-Server	Ja
Microsoft DHCP Service	Ja

Tabelle 3.2: Testergebnisse der DHCP-Serverimplementierungen

Das ältere Betriebssystem Windowsx XP kann durch weiteren administrativen Aufwand im Dual-Stack-Betrieb problemlos eingesetzt werden. Android hingegen ist nur eingeschränkt einsetzbar. Zum einen kann Android keine IPv6-Adresse über DHCPv6 beziehen, zum anderen ist eine Konfiguration eines IPv6-DNS-Servers nicht möglich. Zwar kann Android die Namensauflösung über IPv4 realisieren, würde jedoch die IPv4-Konnektivität ausfallen, so wäre auch eine Kommunikation über IPv6 nicht mehr möglich. Alle weiteren Betriebssysteme können ohne Probleme in einer Dual-Stack-Umgebung genutzt werden (s. Kapitel 3.4.3).

3.5 Domain Name System (DNS)

Zur Namensauflösung von Hostnamen zu IPv6-Adressen ist keine Neuspezifikation des DNS-Protokolls erforderlich. Da DNS keine IPv4-Adressen im Protokollheader verwendet werden, sind lediglich einige Erweiterungen des DNS-Protokolls notwendig.

3.5.1 Dual-Stack-Erweiterungen

Um einen Host oder Service über IPv4 und IPv6 zu erreichen, sind in der Regel mindestens zwei DNS-Einträge notwendig. Im RFC 3596 - *"DNS Extensions to Support IP Version 6"* [THKS03] wird dazu für IPv6-Adressen der Record AAAA spezifiziert, welcher auch umgangsprachlich Quad-A genannt wird [Hag09, S. 353f.]. Zudem muss der Befehlssatz der DNS-Resolver so erweitert werden, das es möglich ist AAAA Records abzufragen.

AAAA Records und IP6.ARPA Domain

Quad-A Records enthalten eine einzelne gespeicherte IPv6-Adressen (128-Bit) und sind somit vier mal so groß wie die in den A Records gespeicherten IPv4-Adressen (32-Bit) [THKS03]. Besitzt ein Host mehrere unterschiedliche IPv6-Adressen über diese er erreichbar sein soll, so kann pro Adresse ein AAAA Record hinzugefügt werden [Hag09, S. 354]. Tabelle 3.3 stellt einen DNS-Eintrag für den Dual-Stack Host *ubuntudualstackserver* mit der IPv4-Adresse *192.168.250.1* und der IPv6-Adresse *2001:638:408:250::1* beispielhaft dar. Mit Hilfe des in Tabelle 3.3 dargestellten DNS-Eintrags ist ein DNS-Server in der Lage, einen anfragenden Client, sowohl die IPv4- als auch die IPv6-Adresse des Zielhosts mitzuteilen.

IPv4/IPv6 DNS-Eintrag			
ubuntudualstackserver	IN	A	192.168.250.1
	IN	AAAA	2001:638:408:250::1

Tabelle 3.3: DNS-Eintrag für IPv4/IPv6

Für die IP-zu-Hostnamen Auflösung sind für IPv6 wie bei IPv4, Einträge in eine Reverse Lookup Domain erforderlich. Dazu wurde für IPv6 im RFC 3596 die Domain *IP6.ARPA* spezifiziert. Jede Subdomain unter IP6.ARPA repräsentiert vier Bits einer IPv6-Adresse, welche als Hexadezimalzahl kodiert ist. Der Subdomainname für die IPv6-Adresse *2001:638:408:250::1* beispielsweise lautet *1.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.5.2.8.0.4.8.3.6.1.0.0.2.ip6.arpa.*. Dabei dürfen keine führenden Nullen fehlen. Um auf einen Host mit dieser IPv6-Adresse zu verweisen, sind den Domainnamen PTR Records zugewiesen[THKS03]. Tabelle 3.4 veranschaulicht den IPv6 Eintrag für die

Adresse *2001:638:408:250::1*. Somit ist es möglich eine IPv6-Adresse in einen Domain-/Hostnamen aufzulösen.

IPv6-Eintrag		
1.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.5.2.8.0.4.8.3.6.1.0.0.2.ip6.arpa.		
IN	PTR	ubuntudualstackserver.local.

Tabelle 3.4: IPv6 Eintrag in die Reverse Lookup Domain

DNS-Resolver

Ein DNS-Resolver ist der Client-Teil eines DNS-Services. Über den DNS-Resolver werden Anfragen für A und AAAA Records getstellt. Der DNS-Resolver ist üblicherweise standardmäßig im Bertiebssystem integriert oder Bestandteil einer Applikation und ist zwingend für die DNS-Kommunikation erforderlich. Deshalb besitzt zudem jeder DNS-Namensserver einen Resolver, um DNS-Anfragen an andere DNS-Namensserver zu stellen. Der Resolver sollte in der Lage sein, DNS-Anfragen für A und AAAA Records sowohl über IPv4, als auch IPv6 zu stellen. So wird sichergestellt das die Kommunikation zwischen Nameserver und Host, die auschließlich entweder IPv4 oder IPv6 nutzen, sichergestellt ist [Hag06, S. 243f.]. In Kapitel 3.5.3 werden die DNS-Resolver der verschiedenen Betriebssysteme getestet.

3.5.2 Serverseitige Implementierungen

In diesem Kapitel werden die bekanntesten DNS-Serverimplementierungen unter Ubuntu 12.04 und Windows Server 2012 installiert, konfiguriert und für den Dual-Stack-Betrieb getestet. Dabei handelt es sich unter Ubuntu Server 12.04, um den von der Universität Berkeley entwickelten klassischen DNS-Server *Berkeley Internet Name Daemon (BIND)* mit der Version 9.8.1-P1. BIND hat nach [DKK+12, S. 727] die mit Abstand größte Präsenz im Internet. Unter Windows Server 2012 wird der bereits integrierte *Microsoft DNS Service* getestet. Die Implementierungen werden so konfiguriert, dass die in Kapitel 3.3 dargestellten Serversysteme, nach der Namensauflösung, über beide IP-Protokolle erreichbar sind. Auch ein Reverse-Mapping (IP-Adresse zu Hostname), soll möglich sein. Die Testdomain ist dabei *dualstack.local*. Die Installation, Konfiguration und Tests erfolgen unter Ubuntu Server 12.04 ausschließlich über die Konsole. Unter Windowsserver 2012 wird zudem die grafische Benutzeroberfläche verwendet.

BIND

Im Folgenden wird die DNS-Serverimplementierung BIND v9.8.1-P1 auf dem Testsystem installiert, konfiguriert und getestet. Nach Angaben von [Hag06, S. 242] unterstützt BIND IPv6 ab Version 8.2.3.

Installation und Konfiguration

BIND befindet sich standardmäßig im Ubuntu-Repository und lässt sich somit leicht über den Befehl in Abbildung 3.21 installieren.

```
root@ubuntudualstack:~# apt-get install bind9
```

Abbildung 3.21: BIND - Installation

Nach Abschluss der Installation, erfolgt nun die Konfiguration von BIND. Für grundlegende Konfigurationen, welche unabhängig von der Dual-Stack-Funktionalität sind empfiehlt sich das Buch [ALL06]. Zunächst wird der BIND-Namensserver so konfiguriert, dass dieser auf IPv6-Anfragen reagiert. Dazu wird in der Konfigurationsdatei */etc/bind/named.conf.options* im Block *options*, der in Abbildung 3.22 dargestellte Parameter hinzugefügt.

```
listen-on-v6 { any; };
```

Abbildung 3.22: BIND - Binden aller IPv6-Adressen

Indem BIND automatisch seinen Dienst über alle IPv4-Adressen bereitstellt, ist für IPv4 keine weitere Konfiguration erforderlich. Ist es notwendig den Service ausschließlich über bestimmte IPv4- und IPv6-Adressen zu binden, sind andere Konfigurationsschritte notwendig. Soll der DNS-Server explizit nur auf Anfragen, die an die IPv6-Adresse *2001:638:408:250::1* gerichtet sind, antworten, so wird der in Abbildung 3.22 dargestellte Parameter modifiziert (s. 3.23).

```
listen-on-v6 { 2001:638:408:250::1; };
```

Abbildung 3.23: BIND - Binden einer bestimmten IPv6-Adresse

Für die Bindung an bestimmte IPv4-Adressen, wie bespielsweise an die IPv4-Adresse *192.168.250.1* ist in der selben Datei folgender Parameter notwendig (s. Abbildung 3.24). Soll der Service über mehrere explizite IPv4-/IPv-Adressen erreichbar sein, so

werden lediglich die neuen IP-Adressen, wie in Abbildung 3.25 zu sehen, durch ein Semikolon voneinander getrennt.

```
listen-on { 192.168.250.1; };
```

Abbildung 3.24: BIND - Binden einer bestimmten IPv4-Adresse

```
listen-on { 192.168.250.1;10.20.116.10;10.20.117.1};
listen-on-v6 { 2001:638:408:250::1;2001:638:408:201::2;};
```

Abbildung 3.25: BIND - Binden mehrerer IPv4-/IPv6-Adressen

Nach einem Neustart des Services mit Hilfe des Konsolenbefehls *sudo /etc/init.d/bind9 restart*, ist der DNS-Server Dual-Stack-fähig. Des Weiteren bietet BIND weitere Einstellungsmöglichkeiten zur Optimierung des DNS-Services im Dual-Stack-Betrieb. Sollen Anfragen an andere DNS-Server nur über ein bestimmtes Interface erfolgen, so lässt sich dies mit dem in Abbildung 3.26 dargestellten Einträgen, im Block *options* in der Konfigurationsdatei */etc/bind/named.conf.options*, realisieren.

```
listen-on { 192.168.250.1;};
listen-on-v6 { 2001:638:408:250::1;};
```

Abbildung 3.26: BIND - Festlegen der Abfrageinterfaces

Eine weitere interessante Option unter BIND ist der Umgang mit DNS-Servern die nicht Dual-Stack-fähig sind, also entweder nur über IPv4 oder IPv6 kommunizieren. Ein mögliches Szenario wäre, dass ein IPv6-only DNS-Server einen Domainnamen aufzulösen versucht, welcher nur von einem IPv4-only DNS-Server bedient wird [Liu11, S. 28f]. BIND bietet dazu den in Abbildung 3.27 dargestellten Befehl, welcher es ermöglicht DNS-Anfragen an anderssprachige DNS-Server über einen Mittelsmann zu stellen. Dieser DNS-Server ist Dual-Stack-fähig und nutzt beide IP-Protokolle. Die in Abbildung 3.27 dargestellte Konfiguration, wäre für einen IPv6-only DNS-Server notwendig. Bei einem IPv4-only DNS-Server wird die IPv6-Adresse einfach durch eine IPv4-Adresse ersetzt.

```
dual-stack-servers {2001:638:408:250::2;};
```

Abbildung 3.27: BIND - Weiterleitung der Abfrage an einen DNS Dual-Stack-Server

Im nächsten Schritt werden die nötigen A und AAAA Record Einträge für die Hosts

ubuntudualstack und *windowsdualstack* in der Domain *dualstack.local* vorgenommen. Die Domaindateien befinden sich standardmäßig im Verzeichnis */etc/bind/*. Dazu werden in der Zonendatei der Domain *dualstack.local*, die in Abbildung 3.28 dargestellten Einträge hinzugefügt.

```
ubuntudualstack         IN      A       192.168.250.1
                        IN      AAAA    2001:638:408:250::1
windowsdualstack        IN      A       192.168.250.2
                        IN      AAAA    2001:638:408:250::2
```

Abbildung 3.28: BIND - A und AAAA Recordeinträge in der Domain dualstack.local.

Für die Rückwärtsauflösung von IP-Adressen in Hostnamen, sind weitere Einträge in den jeweiligen Subdomains notwendig. Bei IPv4 werden dazu in der Zonendatei der Subdomain *250.168.192.in-addr.arpa.* folgende Einträge vorgenommen (s. Abbildung 3.29).

```
1       IN      PTR     ubuntudualstack.dualstack.local.
2       IN      PTR     windowsdualstack.dualstack.local.
```

Abbildung 3.29: BIND - PTR Einträge für IPv4

Bei IPv6 ist ein Eintrag in die Zonendatei der Subdomain *0.5.2.0.8.0.4.0.8.3.6.0.1 .0.0.2.ip6.arpa.* notwendig (s. Abbildung 3.30).

```
1.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0         IN      PTR     ubuntudualstack.dualstack.local.
2.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0         IN      PTR     windowsdualstack.dualstack.local.
```

Abbildung 3.30: BIND - PTR Einträge für IPv6

Damit die Aktualisierungen der Zonendateien wirksam werden, ist es erforderlich in jeder Zonendatei die Seriennummer des SOA Records zu erhöhen. Abbildung 3.31 veranschaulicht den Eintrag der Seriennummer. Üblicherweise wird dabei oft das Datum der Änderung und ein Zähler verwendet.

```
2013012500      ;Serial
```

Abbildung 3.31: BIND - Aktualisierung der Zone durch erhöhen der Seriennummer

Testphase

Ein Test mit Hilfe von netstat zeigt das der DNS-Service über beide IP-Protokolle verfügbar ist (s. Abbildung 3.32).

```
root@ubuntudualstack:~# netstat -lnp|grep :53
tcp        0      0 192.168.250.1:53        0.0.0.0:*               LISTEN      2726/named
tcp6       0      0 2001:638:408:250::1:53  :::*                    LISTEN      2726/named
udp        0      0 192.168.250.1:53        0.0.0.0:*                           2726/named
udp6       0      0 2001:638:408:250::1:53  :::*                                2726/named
```

Abbildung 3.32: BIND - Serviceverfügbarkeit

Im nächsten Schritt wir die Servicefunktionalität getestet. Dazu wird das Tool *dig (domain information groper)* verwendet. Dig ist ein Werkzeug um DNS-Informationen von Nameservern abzufragen und wird für die Analyse und Fehlersuche bei DNS verwendet. Das Tool ist in der Lage gezielt DNS-Server über beide IP-Protokolle nach A, AAAA und PTR Records abzufragen. Bei DNS im Dual-Stack-Betrieb sind vier Abfrage-Konstellationen möglich, welche in Tabelle 3.5 dargestellt werden. Im ersten Fall, erfolgt eine DNS-Anfrage über IPv4 für einen A Record. Im zweiten Fall, wird ein AAAA Record über IPv4 angefragt. Im dritten Fall, wird über IPv6 der A Record angefragt. Zuletzt erfolgt eine DNS-Anfrage über IPv6, für einen AAAA Record.

IP-Version	Record-Typ
IPv4	A
IPv4	AAAA
IPv6	A
IPv6	AAAA

Tabelle 3.5: DNS - Abfrage-Konstellationen

Die ersten beiden Testfälle sind notwendig um eine Aussage über die Servicefunktionalität via IPv4 zu ermöglichen. Die letzten beiden Testfälle hingegen sind für die Servicefunktionalität über IPv6 erforderlich. Mit Hilfe des Befehls *dig @192.168.250.1 -t any ubuntudualstack.dualstack.local* wird zunächst die IPv4- und IPv6-Adresse des Hosts *ubuntudualstack* über IPv4 angefragt. Abbildung 3.33 stellt das positive Ergebnis dar.

```
;; ANSWER SECTION:
ubuntudualstack.dualstack.local. 43200 IN A     192.168.250.1
ubuntudualstack.dualstack.local. 43200 IN AAAA  2001:638:408:250::1
```

Abbildung 3.33: BIND - A/AAAA Recordantwort über IPv4

In einem weiteren Test, mit Hilfe des Befehls *dig @2001:638:408:250::1 -t any ubuntudualstack.dualstack.local*, wird die IPv4- und IPv6-Adresse des Hosts *ubuntudualstack* diesmal über IPv6 angefragt. Wie in Abbildung 3.34 zu sehen, ist auch dieser Test erfolgreich.

```
;; ANSWER SECTION:
ubuntudualstack.dualstack.local. 43200 IN A     192.168.250.1
ubuntudualstack.dualstack.local. 43200 IN AAAA  2001:638:408:250::1
```

Abbildung 3.34: BIND - A/AAAA Recordantwort über IPv6

Zuletzt wird die Rückwärtsauflösung getestet. Durch verwenden der Befehle *dig @192.168.250.1 -x 192.168.250.1* und *dig @2001:638:408:250::1 -x 2001:638:408:250::1* wird getestet ob die IP-Adressen zu Hostnamen aufgelöst werden. Die beiden Abbildungen 3.35 und 3.36 belegen die erfolgreiche Auflösung der beiden IP-Adressen. Die DNS-Serviceimplementierung BIND ist Dual-Stack-fähig.

```
;; ANSWER SECTION:
1.250.168.192.in-addr.arpa. 43200 IN    PTR     ubuntudualstack.dualstack.local.
```

Abbildung 3.35: BIND - Rückwärtsauflösung einer IPv4-Adressse

```
;; ANSWER SECTION:
1.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.5.2.0.8.0.4.0.8.3.6.0.1.0.0.2.ip6.arpa. 43200 IN PTR ubuntudualstack.dualstack.local.
```

Abbildung 3.36: BIND - Rückwärtsauflösung einer IPv6-Adressse

Microsoft DNS Service

Im folgenden wird die DNS-Serverimplementierung Microsoft DNS Service auf dem Testsystem installiert, konfiguriert und getestet.

Installation und Konfiguration

Installieren lässt sich der Service über den integrierten Server Manager. Dazu sind folgende Schritte erforderlich:

- Start –> Server-Manager –> Verwalten –> Rollen und Features hinzufügen
- Installationstyp: *Rollenbasierte oder featurebasierte Installation* auswählen
- Serverauswahl: Auswählen des Servers auf dem der Service verfügbar sein soll
- Serverrollen: DNS-Server auswählen
- Features: keine Auswahl

Im nächsten Schritt erfolgt die Konfiguration des DNS-Services. Über *Start –> Server-Manager –> Tools –> DNS* lässt sich der Service im DNS-Manager konfigurieren. Damit der DNS-Server gezielt auf bestimmten Schnittstellen lauscht wird

über einen Rechtsklick auf den *DNS-Server –> Eigenschaften* das entsprechende Konfigurationsmenü aufgerufen. Wie in Abbildung 3.37 zu sehen, lassen sich im Reiter *Schnittstellen* die einzelnen IP-Adressen auswählen. Die Einstellungen sind ohne Neustart des Servers aktiv.

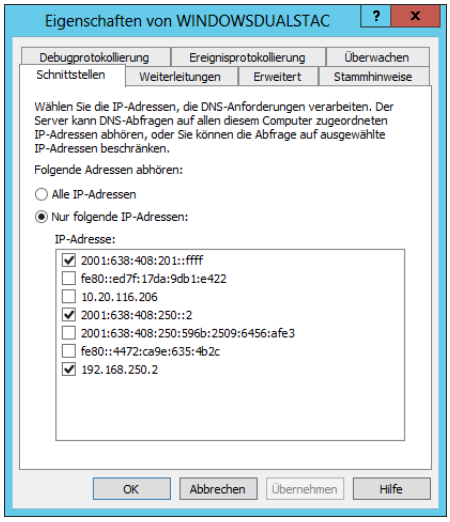


Abbildung 3.37: Microsoft DNS Service - Schnittstellen binden

Im nächsten Schritt werden die A und AAAA Records angelegt. Zunächst wird eine Forward-Lookupzone benötigt, welche die A und AAAA Records speichert. Dazu sind folgende Schritte im DNS-Manager notwendig:

- Auswahl des Servers auf dem die Zone erstellt werden soll
- Forward-Lookupzonen (Rechtsklick) –> Neue Zone
- Zonentyp: Primäre Zone
- Zonenname: dualstack.local
- Zonendatei: dualstack.local.dns

Nach Abschluss der Konfiguration wird die neue Zone unter *Forward-Lookupzonen* hinzugefügt. Anschließend werden die A und AAAA Records durch einen Rechtsklick auf *dualstack.local –> Neuer Host (A oder AAAA)* hinzugefügt. Abbildung 3.38 zeigt beispeilhaft das Anlegen eines AAAA Records für den Host *windowsdualstack*. Das anlegen eines A Records funktioniert identisch. Jedoch wird im Feld *IP-Adresse*, die IPv4-Adresse angegeben.

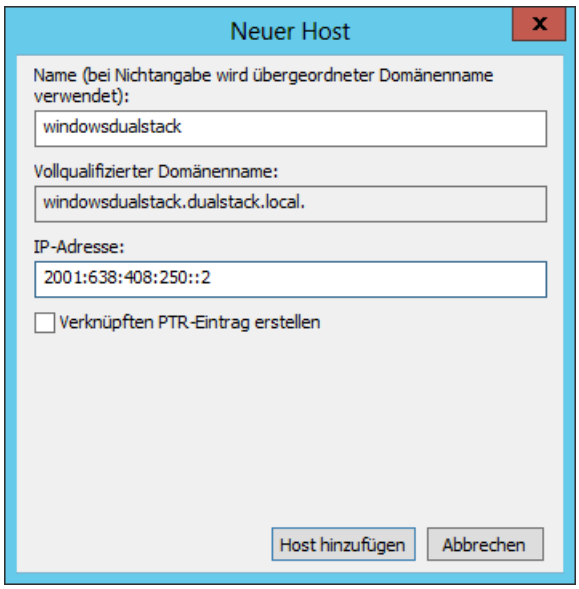


Abbildung 3.38: Microsoft DNS Service - AAAA Record anlegen

Für die Rückwärtsauflösung von IP-Adressen zu Hostnamen, werden zwei Reverse-Lookupzonen benötigt. Dies wird für die IPv6-Reverse-Lookupzone folgendermaßen realisiert:

- Auswahl des Servers auf dem die Zone erstellt werden soll
- Reverse-Lookupzonen (Rechtsklick) –> Neue Zone
- Zonentyp: Primäre Zone
- IP-Version: IPv6-Reverse-Lookupzone
- Präfix der IPv6-Adresse: 2001:638:408:250::/64

Für das anlegen einer IPv4-Reverse-Lookupzone muss im Konfigurationsfenster *IPv4-Reverse-Lookupzone* ausgewählt und die Netztwerk-ID des IPv4-Netzes angegeben werden. In den Reverse-Lookupzonen können daraufhin die PTR Records, für die IP-Adress-zu-Hostnamen-Auflösung, angelegt werden. Über einen Rechtsklick auf die entsprechende Reverse-Lookupzone, können über die Option *Neuer Zeiger (PTR)* die PTR Records angelegt werden. Abbildung 3.39 zeigt dies beispielhaft für die IPv6-Adresse des Windows-Servers. Nach Abschluss der Konfiguration ist der DNS-Server im Dual-Stack-Betrieb einsetzbar.

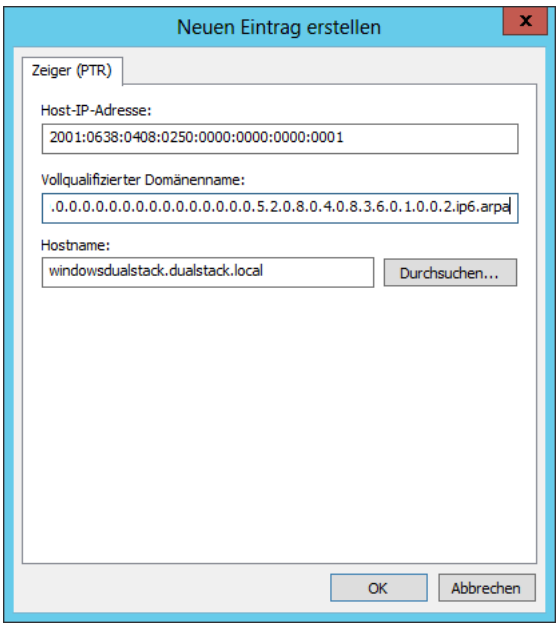


Abbildung 3.39: Microsoft DNS Service - PTR Record anlegen

Testphase

Abbildung 3.40 zeigt das der DNS-Service auf dem Server verfügbar ist.

```
Aktive Verbindungen
  Proto  Lokale Adresse          Remoteadresse          Status
  TCP    127.0.0.1:53            0.0.0.0:0              ABH?REN
  TCP    192.168.250.2:53        0.0.0.0:0              ABH?REN
  TCP    [::1]:53                [::]:0                 ABH?REN
  TCP    [2001:638:408:250::2]:53  [::]:0                  ABH?REN

  UDP    127.0.0.1:53            *:*
  UDP    192.168.250.2:53        *:*
  UDP    [::]:5355               *:*
  UDP    [::1]:53                *:*
  UDP    [2001:638:408:250::2]:53  *:*
```

Abbildung 3.40: Microsoft DNS Service - Serviceverfügbarkeit

Wie bereits bei der DNS-Serverimplementierung BIND unter Ubuntu Server 12.04, werden mit Hilfe des Tools dig zuerst die Records via IPv4 abgefragt. Abbildung 3.41 zeigt den Erfolg der Abfrage.

```
;; ANSWER SECTION:
windowsdualstack.dualstack.local. 3600 IN A     192.168.250.2
windowsdualstack.dualstack.local. 3600 IN AAAA  2001:638:408:250::2
```

Abbildung 3.41: Microsoft DNS Service - A/AAAA Recordantwort über IPv4

Ein weiterer Test zur Abfrage der Records über IPv6 fällt ebenfalls positiv aus (s. Abbildung 3.42).

```
;; ANSWER SECTION:
windowsdualstack.dualstack.local. 3600 IN A     192.168.250.2
windowsdualstack.dualstack.local. 3600 IN AAAA  2001:638:408:250::2
```

Abbildung 3.42: Microsoft DNS Service - A/AAAA Recordantwort über IPv6

Zuletzt wird die Rückwärtsauflösung von IPv4- und IPv6-Adresssen zu Hostnamen getestet. Dies erfolgt mit Hilfe der Befehle *dig @192.168.250.2 -x 192.168.250.2* und *dig @2001:638:408:250::2 -x 2001:638:408:250::2*. Die Abbildungen 3.43 und 3.44 zeigen, das beide Tests für IPv4 und IPv6 erfolgreich sind. Die DNS-Serverimplementierung Microsoft DNS Service ist Dual-Stack-fähig.

```
;; ANSWER SECTION:
2.250.168.192.in-addr.arpa. 3600 IN     PTR     windowsdualstack.dualstack.local.
```

Abbildung 3.43: Microsoft DNS Service - Rückwärtsauflösung einer IPv4-Adressse

```
;; ANSWER SECTION:
2.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.5.2.0.8.0.4.0.8.3.6.0.1.0.0.2.ip6.arpa. 3600 IN PTR windowsdualstack.dualstack.local.
```

Abbildung 3.44: Microsoft DNS Service - Rückwärtsauflösung einer IPv6-Adressse

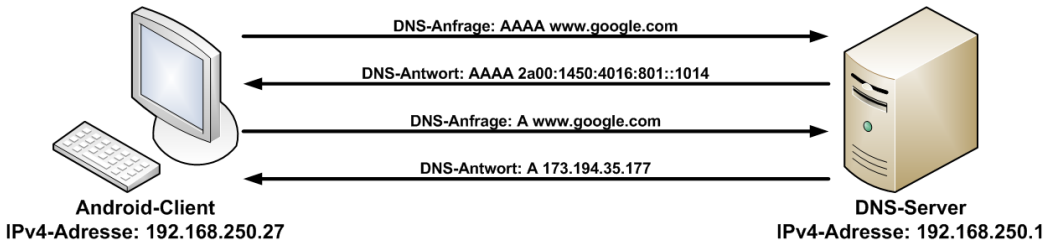
3.5.3 Clientseitige Implementierungen

In diesem Abschnitt werden die DNS-Resolver der in Kapitel 3.3 vorgestellten Betriebssysteme unter die Lupe genommen. Dabei wird getestet ob der Client in der Lage ist DNS-Anfragen für A und AAAA Records zu stellen. Des Weiteren wird überprüft über welches IP-Protokoll die DNS-Anfragen vorrangig transportiert werden oder ob eine parallele Anfrage über beide Protokollversionen stattfindet. In Kapitel 3.4 wurde bereits die Autokonfiguration erläutert, welche es unter anderem ermöglicht die DNS-Serveradresse automatisch zu verteilen. Bei Betriebssystemen, die Probleme bei der Nutzung der Autokonfiguration haben, wird versucht die DNS-Serveradresse statisch zu konfigurieren.

Android

Wie bereits in Kapitel 3.4.3 beschrieben, ist eine automatische Konfiguration der DNS-Serveradressse für IPv6 unter Android nicht möglich. Eine statische Konfiguration für IPv6 ist ebenfalls nicht möglich. Somit kann keine DNS-Anfrage über IPv6 gestellt werden. Eine DNS-Anfrage zur Auflösung von *google.com* über IPv4 zeigt, dass der DNS-Resolver von Android in der Lage ist A und AAAA Records anzufragen. Abbildung 3.45 stellt den Kommunkationsverlauf anhand einer Grafik und eines

Wireshark-Mitschnittes dar.

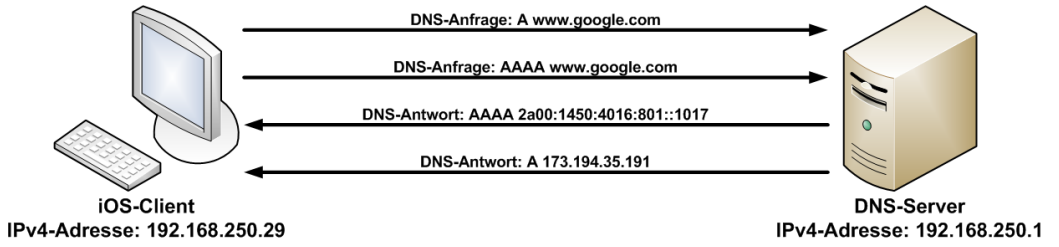


No.	Time	Source	Destination	Protocol	Length	Info
69	46.414889	192.168.250.27	192.168.250.1	DNS	74	Standard query AAAA www.google.com
70	46.419230	192.168.250.1	192.168.250.27	DNS	174	Standard query response AAAA 2a00:1450:4016:801::1010
71	46.421377	192.168.250.27	192.168.250.1	DNS	74	Standard query A www.google.com
72	46.425661	192.168.250.1	192.168.250.27	DNS	226	Standard query response A 173.194.35.179 A 173.194.35.180

Abbildung 3.45: DNS-Kommunikation unter Android

iOS

Die Autokonfiguration der DNS-Server-Adresse über DHCPv6 und Routeradvertisements funktioniert unter iOS einwandfrei. Die Namensauflösung von *google.com* erfolgt im Dual-Stack-Betrieb über IPv4. Beide Recordtypen, A und AAAA, werden angefragt (s. Abbildung 3.46).

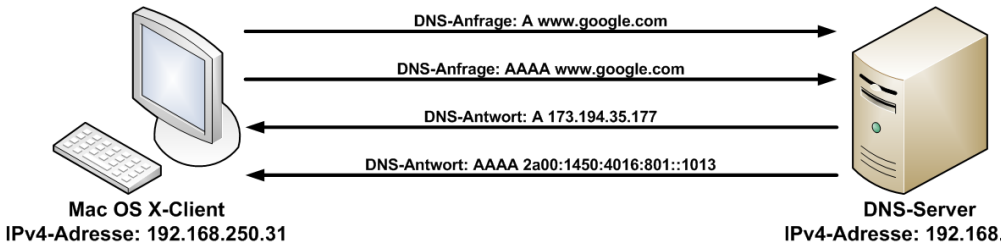


No.	Time	Source	Destination	Protocol	Length	Info
13	0.660330	192.168.250.29	192.168.250.1	DNS	73	Standard query A www.google.de
14	0.660690	192.168.250.29	192.168.250.1	DNS	73	Standard query AAAA www.google.de
15	0.686062	192.168.250.1	192.168.250.29	DNS	183	Standard query response AAAA 2a00:1450:4016:801::1017
16	0.686087	192.168.250.1	192.168.250.29	DNS	203	Standard query response A 173.194.35.191 A 173.194.35.183

Abbildung 3.46: DNS-Kommunikation unter iOS

Mac OS X

Wie bei iOS funktioniert die Autokonfiguration der DNS-Serveradresse ohne Probleme. Aber auch die DNS-Kommunikation ist identisch. Die A und AAAA Records werden im Dual-Stack-Betrieb ausschließlich über IPv4 angefragt (s. Abbildung 3.47).

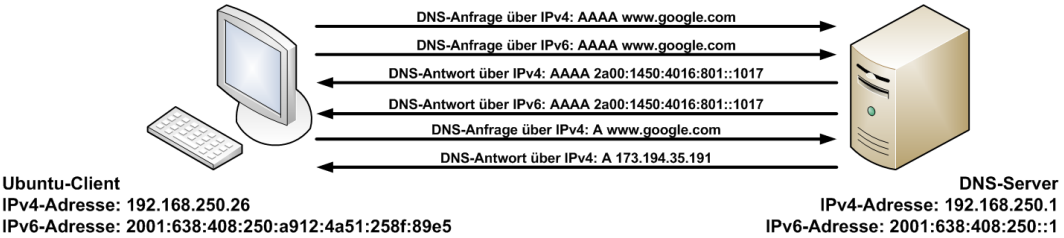


No.	Time	Source	Destination	Protocol	Length	Info
15	0.757042	192.168.250.31	192.168.250.1	DNS	74	Standard query A www.google.com
16	0.757236	192.168.250.31	192.168.250.1	DNS	74	Standard query AAAA www.google.com
17	0.761447	192.168.250.1	192.168.250.31	DNS	226	Standard query response A 173.194.35.177 A 173.194.35.178
18	0.761470	192.168.250.1	192.168.250.31	DNS	174	Standard query response AAAA 2a00:1450:4016:801::1013

Abbildung 3.47: DNS-Kommunikation unter Mac OS X

Ubuntu Desktop 12.04

Unter Ubuntu erfolgt die DNS-Kommunikation parallel über beide IP-Protokolle. Über IPv6 wird nur der AAAA Record angefragt. Über IPv4 werden beide Records angefragt. Abbildung 3.48 veranschaulicht die DNS-Kommunikation unter Ubuntu.

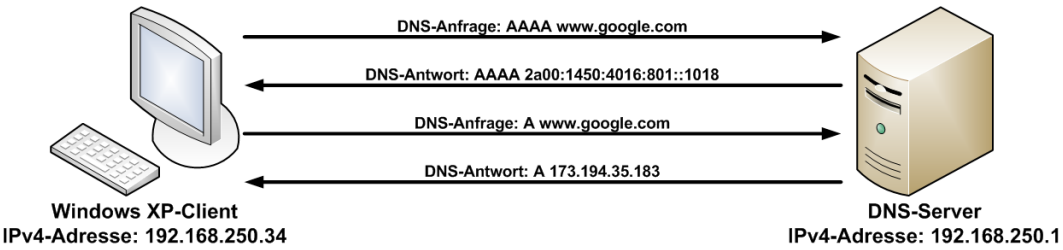


No.	Time	Source	Destination	Protocol	Length	Info
32	6.053203	192.168.250.26	192.168.250.1	DNS	73	Standard query AAAA www.google.de
33	6.053246	2001:638:408:250:a912:4a51:258f:89e5	2001:638:408:250::1	DNS	93	Standard query AAAA www.google.de
34	6.053572	192.168.250.1	192.168.250.26	DNS	183	Standard query response AAAA 2a00:1450:4016:801::1018
35	6.053593	2001:638:408:250::1	2001:638:408:250:a912:4a51:258f:89e5	DNS	203	Standard query response AAAA 2a00:1450:4016:801::1018
36	6.053814	192.168.250.26	192.168.250.1	DNS	73	Standard query A www.google.de
37	6.054222	192.168.250.1	192.168.250.26	DNS	203	Standard query response A 173.194.35.183 A 173.194.35.

Abbildung 3.48: DNS-Kommunikation unter Ubuntu

Windows XP

Mit Hilfe des Befehls *netsh interface ipv6 add dns "«Interfacename»" «IPv6-Adresse» index=1* in der Konsole, ist es möglich die IPv6-Adresse des DNS-Servers unter Windows XP statisch einzutragen. Bei einer DNS-Anfrage wird jedoch das IPv4-Protokoll bevorzugt genutzt. Abbildung 3.49 stellt die DNS-Kommunikation unter Windows XP grafisch dar.

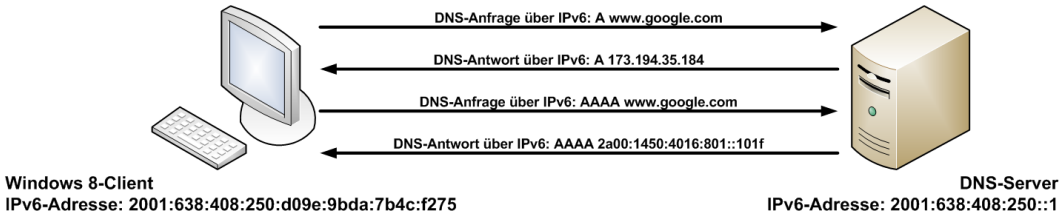


No.	Time	Source	Destination	Protocol	Length	Info
115	52.2680	192.168.250.34	192.168.250.1	DNS	69	Standard query AAAA google.de
116	52.3002	192.168.250.1	192.168.250.34	DNS	179	Standard query response AAAA 2a00:1450:4016:801::1018
117	52.3046	192.168.250.34	192.168.250.1	DNS	69	Standard query A google.de
118	52.3190	192.168.250.1	192.168.250.34	DNS	199	Standard query response A 173.194.35.183 A 173.194.35.184

Abbildung 3.49: DNS-Kommunikation unter Windows XP

Windows 7/8

Beide Versionen von Windows verhalten sich im Ablauf ihrer DNS-Kommunikation identisch. Sind DNS-Serveradressen für beide IP-Protokolle durch die Autokonfiguration eingetragen, so erfolgt die DNS-Kommunikation im Dual-Stack-Betrieb ausschließlich über IPv6. Über IPv6 werden der A und der AAAA Record angefragt. Abbildung 3.50 zeigt den Kommunikationsablauf unter Windows 7 und 8.



No.	Time	Source	Destination	Protocol	Length	Info
27	17.4930	2001:638:408:250:d09e:9bda:7b4c:f275	2001:638:408:250::1	DNS	93	Standard query A www.google.de
28	17.4932	2001:638:408:250::1	2001:638:408:250:d09e:9bda:7b4c:f275	DNS	223	Standard query response A 173.194.35.184 A 173.194.35.191
29	17.4936	2001:638:408:250:d09e:9bda:7b4c:f275	2001:638:408:250::1	DNS	93	Standard query AAAA www.google.de
30	17.4939	2001:638:408:250::1	2001:638:408:250:d09e:9bda:7b4c:f275	DNS	203	Standard query response AAAA 2a00:1450:4016:801::101f

Abbildung 3.50: DNS-Kommunikation unter Windows 7/8

3.5.4 Zusammenfassung

In Tabelle 3.6 werden die Ergebnisse der DNS-Serverimplementierungen dargestellt. Beide Serverimplementierungen unterstützen das Forward-Mapping (Domainname zu IP-Adresse) mit Hilfe von A und AAAA Records. Auch das Reverse-Mapping funktioniert bei beiden Implementierungen für beide IP-Protokolle. Zudem unterstützt BIND und der Microsoft DNS Service die DNS-Kommunikation über IPv4 und IPv6. Beide Implemetierungen sind im Dual-Stack-Betrieb einsetzbar.

DNS-Serverimplementierung	Dual-Stack-Betrieb
BIND	Ja
Microsoft DNS Service	Ja

Tabelle 3.6: Testergebnisse der DNS-Serverimplementierungen

Die DNS-Clients hingegen verhalten sich sehr unterschiedlich. Indem Android nicht fähig ist die IPv6-Adresse des DNS-Servers per Autokonfiguration zu speichern, ist eine DNS-Anfrage über IPv6 nicht möglich. Android ist jedoch in der Lage über IPv4, A und AAAA Records von einem Nameserver anzufragen. Somit ist eine anschließende Kommunikation, nach der Namensauflösung, über beide IP-Protokolle möglich. Fällt die IPv4-Konnektivität aus, so kann Android den DNS-Service nicht mehr in Anspruch nehmen. Auch iOS und Mac OS X verhalten sich identisch. Beide Betriebssysteme sind zwar in der Lage die IPv6-Adresse des DNS-Servers per Autokonfiguration zu speichern, verwenden jedoch bevorzugt IPv4 für die DNS-Kommunikation. Fällt IPv4 aus, so sind allerdings beide Betiebssysteme in der Lage die DNS-Kommunikation über IPv6 auszuführen. Bei Ubuntu erfolgt die DNS-Kommunikation über beide IP-Protokolle parallel. Über IPv6 wird jedoch lediglich der AAAA Record angefragt. Über IPv4 hingegen werden beide Recordtypen angefordert. Ganz anders Verhalten sich Windows 7 und 8. Beide Betriebssysteme verwenden in einer Dual-Stack-Umgebung vorrangig IPv6 als Übertragungsprotokoll für die DNS-Kommunikation. Im Gegensatz zu Ubuntu, werden bei Windows 7 und 8 A und AAAA Records über IPv6 angefragt. Unter Windows XP ist eine statische Eintragung der IPv6-Adressse des DNS-Servers möglich. Im Gegensatz zu seinen Nachfolgern erfolgt die DNS-Kommunikation im Dual-Stack-Betrieb über IPv4. Fällt die IPv4-Konnektivität aus, so ist Windows XP in der Lage DNS-Anfragen über IPv6 zu versenden. Tabelle 3.7 stellt das Ergebnis der DNS-Clients zusammenfassend dar.

Betriebssystem	Record-Typen	IP-Support	Priorität
Android 2.3.6	A/AAAA	IPv4	-
iOS 6.1	A/AAAA	IPv4/IPv6	IPv4
Mac OS X 10.8.2	A/AAAA	IPv4/IPv6	IPv4
Ubuntu Desktop 12.04	A/AAAA	IPv4/IPv6	IPv4/IPv6
Windows XP	A/AAAA	IPv4/IPv6	IPv4
Windows 7	A/AAAA	IPv4/IPv6	IPv6
Windows 8	A/AAAA	IPv4/IPv6	IPv6

Tabelle 3.7: Testergebnisse der DNS-Clientimplementierungen

Android lässt sich in einer Dual-Stack-Umgebung nur eingeschränkt nutzen, da eine Namensauflösung über IPv6 nicht möglich ist. Für Windows XP muss zuvor manuell administriert werden. Alle anderen Betriebssysteme können ohne weiteren Arbeitsaufwand in einer Dual-Stack-Umgebung eingesetzt werden.

3.6 Hypertext Transfer Protocol (HTTP)

HTTP ist unabhängig vom IPv4-Protokoll aufgebaut. Eine Neuspezifikation oder Anpassung des Protokolls für IPv6 ist somit nicht notwendig.

3.6.1 Dual-Stack-Erweiterungen

Bei Anwendungen, die HTTP nutzen sollte darauf geachtet werden, dass die IPv6-Adresse in eckigen Klammern steht, so wie es im RFC 2732 spezifiziert ist [HCM99]. Die Nutzung von HTTP über IPv4 und IPv6, birgt jedoch ein weiteres Problem, welches im folgenden Abschnitt behandelt wird.

Happy Eyeball-Mechanismus

Der Happy Eyeball-Mechanismus wird im RFC 6555 - *"Happy Eyeballs: Success with Dual-Stack Hosts"* spezifiziert [WY12]. Ziel dieses Mechanismuses ist es, Verzögerungen zu vermeiden, die bei Dual-Stack-Hosts auftreten können. Versucht ein Client eine Verbindung zu einem Service, der ausschließlich über IPv4 erreichbar ist, zunächst über IPv6 aufzubauen und wechselt erst nach wenigen Sekunden zu IPv4, entsteht eine Verzögerung, die für den Anwender, der beispielsweise eine Webseite aufruft, nicht hinnehmbar ist. Mit Hilfe des Happy Eyeball-Mechanismus wird versucht diesem Problem entgegenzuwirken. Die Performance einer Anwendung sollte nicht durch das IP-Protokoll beeinflusst werden. Abbildung 3.51 veranschaulicht das Problem.

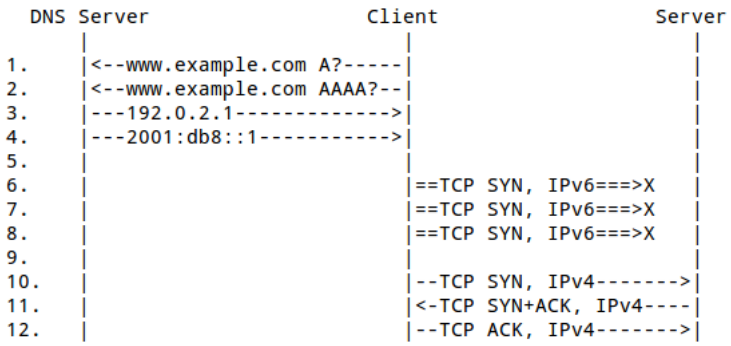


Abbildung 3.51: Happy Eyeball - Fehlverhalten einer Anwendung [WY12]

Der Webbrowser sendet zunächst eine DNS-Anfrage an einen Nameserver. Dieser antwortet mit einer IPv4- und einer IPv6-Adresse. Angenommen wird, dass der Webbrowser so implemetiert ist, dass dieser IPv6-Adressen bevorzugt. Ist der Service jedoch nicht über IPv6-erreichbar, entsteht eine Verzögerung, bevor der Webbrowser anschließend die Verbindung über IPv4 aufbaut [WY12].

Wird der Happy Eyeball-Mechanismus implementiert so wird das Problem, wie in Abbildung 3.52 zu sehen, gelöst. Nachdem der Webbrowser den Domainnamen aufgelöst hat, sendet dieser parallel zwei TCP SYNs über IPv4 und IPv6. Erfolgt keine Bestätigung für den IPv6-Pfad, so wird der bereits initialisierte IPv4-Pfad verwendet. Somit werden Verzögerungen auf ein Minimum reduziert. Damit der Netzwerktraffic nicht unnötig ansteigt, wird das Ergebnis, über welchen Pfad der Service erreichbar ist, für maximal 10 Minuten im Cache gespeichert. Sollte der IPv6-Pfad nutzbar sein, so wird dieser vor IPv4 bevorzugt. Der Happy Eyeball-Mechanismus ist nicht ausschließlich für Webbrowser geeignet, sondern für eine Vielzahl von Anwendungen, wie beispielsweise FTP-, Email- und Instant-Messaging-Clients [WY12].

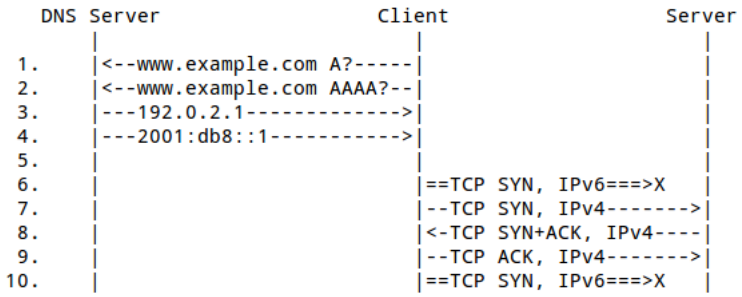


Abbildung 3.52: Happy Eyeball - Empfohlenes Verhalten einer Anwendung [WY12]

3.6.2 Serverseitige Implementierungen

Im Folgenden werden jeweils eine Serverimplementierung unter Ubuntu Server 12.04 und Windows Server 2012 getestet. Beide Implementierungen werden installiert, konfiguriert und in einer Dual-Stack-Umgebung getestet. Dabei ist es notwendig, dass alle Webinhalte über IPv4 und IPv6 erreichbar sind. Zudem soll der verschlüsselte Zugang über *Hypertext Transfer Protocol Secure (HTTPS)* mit beiden IP-Protokollen möglich sein, da viele Webseiten Login-Funktionen anbieten. Unter Ubuntu Server 12.04 wird der Webserver *Apache v2.2.22* auf seine Dual-Stack-Fähigkeit getestet. Nach Angaben von [DKK+12, S. 219], werden 65% aller Webangebote über Apache-Webserver angeboten. Unter Microsoft Windows Server 2012 wird der bereits integrierte *Microsoft Internet Information Service v8.0* im Dual-Stack-Betrieb geprüft.

Apache

Der Webserver Apache unterstützt nach [Sto07, S. 94], IPv6 seit Version 2.0 stabil.

Installation und Konfiguration

Apache befindet sich standardmäßig im Ubuntu-Repository und wird somit über fol-

genden Befehl installiert:

```
root@ubuntudualstack:~# apt-get install apache2
```

Abbildung 3.53: Apache - Installation

Nach Abschluss der Installation erfolgt die Konfiguration des Webservers für den Dual-Stack-Betrieb. Der Webserver wird so konfiguriert das dieser auf HTTP- und HTTPS-Anfragen antwortet, welche über die IP-Adressen *192.168.250.1* und *2001:638:408:250::1* eingehen. Für die Grundlegende Konfiguration des Apache-Webserver empfiehlt sich [DKK+12, S. 219ff]. Damit der Server auf allen IPv4- und IPv6-Adressen lauscht, genügen die in Abbildung 3.54 dargestellten Einträge in der Datei */etc/apache2/ports.conf*.

```
Listen 80

<IfModule mod_ssl.c>
    Listen 443
</IfModule>
```

Abbildung 3.54: Apache - Binden an alle IP-Adressen

Für die explizite Bindung von IP-Adressen müssen die Listen-Direktiven in der selben Konfigurationsdatei angepasst werden. Abbildung 3.55 zeigt die erforderlichen Modifizierungen. Zu beachten ist, dass die IPv6-Adressen in eckigen Klammern plaziert werden. Sind weitere Adressbindungen erforderlich, so werden diese einfach über weitere Listen-Direktiven hinzugefügt. Nach einem anschließenden Neustart mit Hilfe des Befehls *sudo /etc/init.d/apache2 restart* werden die neuen Einstellungen übernommen.

```
Listen 192.168.250.1:80
Listen [2001:638:408:250::1]:80

<IfModule mod_ssl.c>
    Listen 192.168.250.1:443
    Listen [2001:638:408:250::1]:443
</IfModule>
```

Abbildung 3.55: Apache - Binden expliziter IP-Adressen

Testphase

Nach erfolgreicher Installation und Konfiguration wird die Serviceverfügbarkeit des Apache-Webservers getestet. Mit Hilfe von *netstat* wird überprüft, ob der Service auf den zuvor konfigurierten Sockets lauscht. Abbildung 3.56 zeigt, dass der Apache

über die konfigurierten Sockets adressiert werden kann.

```
root@ubuntudualstack:~# netstat -ln|egrep '80|443'
tcp        0      0 192.168.250.1:80        0.0.0.0:*               LISTEN
tcp        0      0 192.168.250.1:443       0.0.0.0:*               LISTEN
tcp6       0      0 2001:638:408:250::1:80  :::*                    LISTEN
tcp6       0      0 2001:638:408:250::1:443 :::*                    LISTEN
```

Abbildung 3.56: Apache - Serviceverfügbarkeit

Des Weiteren wird mit Hilfe des zeichenorientierten Browsers *lynx* der Verbindungsaufbau zum Service über IPv4 und IPv6 getestet. Über lynx wird eine Verbindung zum Service über die IP-Adressen *192.168.250.1* und *2001:638:408:250::1* auf den Ports 80 und 443 hergestellt. Als Ergbnis wird die auf dem Webserver abgelegte Webseite im Textmodus dargestellt. So wird überprüft ob der Service über die konfigurierten Sockets erreicht wird. Abbildung 3.57 stellt den erfolgreichen Verbindungsaufbau, mit Hilfe des Konsolenbfehls *lynx https://[2001:638:408:250::1]* dar. Alle weiteren Tests über IPv4/IPv6 und Port 80/443 ergaben das selbe Resultat. Die Serverimplementierung Apache v2.2.22 ist Dual-Stack-fähig.

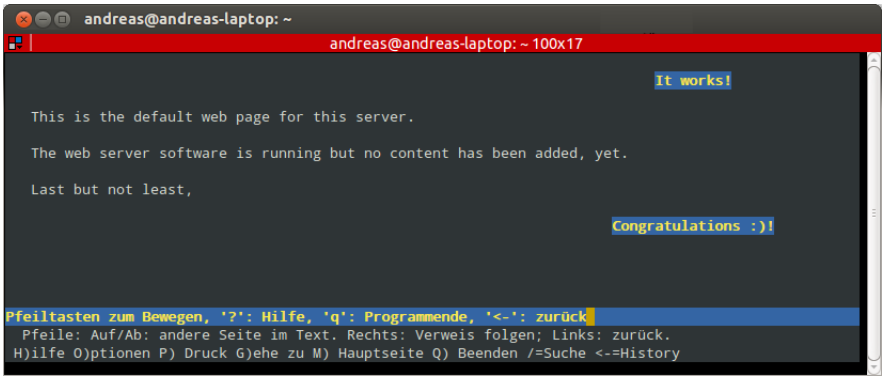


Abbildung 3.57: Apache - Servicefunktionalität

Microsoft Internet Information Service

Im Folgenden wird der Microsoft Internet Information Service, kurz IIS, auf dem Windows Server 2012 Testsystem getestet.

Installation und Konfiguration

Der Webserver wird über den in Windows Server 2012 integrierten Server-Manager installiert. Dazu sind folgende Schritte notwendig:

- Server-Manager –> Verwalten –> Rollen und Features hinzufügen

- Installationstyp: *Rollenbasierte oder featurebasierte Installation* auswählen
- Serverauswahl: Auswählen des Servers, auf dem der Service installiert werden soll
- Serverrollen: Webserver (IIS)
- Features: keine Auswahl

Anschließend ist ein Neustart des Servers erforderlich. Nach der Initialisierung des Webservers, wird die Konfiguration für den Dual-Stack-Betrieb vorgenommen. Der Webserver lauscht standardmäßig auf allen vefügbaren IP-Adressen auf Port 80. Somit ist der Internet Information Service bereits standardmäßig für den Dual-Stack-Betrieb konfiguriert. Für die allgemeine Konfiguration des Webservers empfiehlt sich die Microsoft TechNet-Bibliothek [Mic13]. Damit der Webservice ausschließlich über explizite Adressen erreichbar ist und zudem auch https verfügbar ist, sind unter

- Start –> Informationsdienste (IIS)-Manager ausführen
- Server auswählen –> Sites
- (Rechtsklick) auf die gewünschte Webseiten-Freigabe –> *Bindungen bearbeiten...* auswählen

die in Abbildung 3.58 dargestellten Konfigurationen notwendig. Über die Schaltfläche *Hinzufügen...*, lassen sich weitere Sitebindungen für http oder https hinzufügen. Die Änderungen sind ohne Neustart des Services aktiv. Der Server bietet daraufhin seinen Service über die konfigurierten Sockets an.

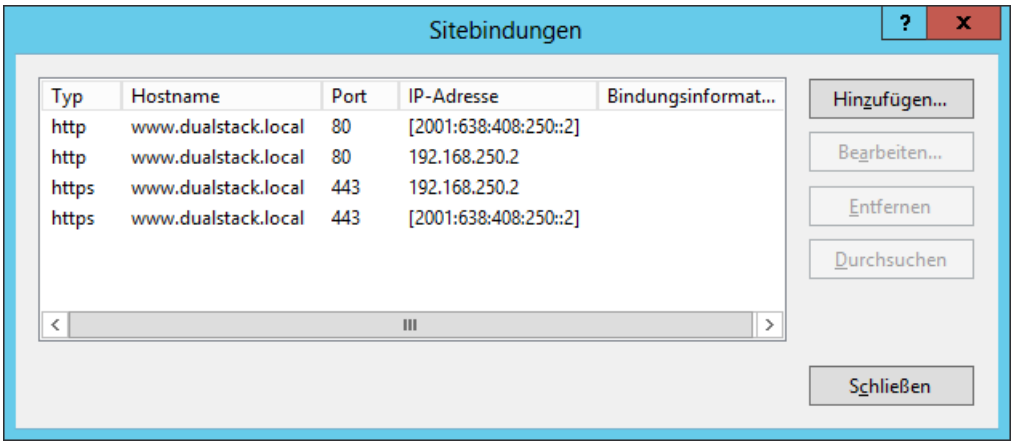


Abbildung 3.58: IIS - Konfiguration

Testphase

Zunächst wird die Serviceverfügbarkeit mit Hilfe des Befehls *netstat -an|findstr ":443 :80"* auf dem Server geprüft. Abbildung 3.59 zeigt, dass der Windowsserver auf Port 80 und 443 über alle IP-Adressen lauscht.

Abbildung 3.59: IIS - Serviceverfügbarkeit

Wird der Service, wie zuvor beschrieben, ausschließlich an bestimmte IP-Adressen gebunden, so liefert netstat trotzdem dasselbe Ergebnis. Ein Versuch den Service über eine andere nicht explizit gebunde IP-Adresse zu erreichen, schlägt jedoch fehl. Der IIS verwirft die Anfrage auf diesem Socket. Im folgenden wird versucht den Service mit Hilfe des Tools *lynx*, über IPv4 und IPv6 zu erreichen. Dabei werden die Anfragen über http und https gesendet. Abbildung 3.60 zeigt den erfolgreichen Verbindungsaufbau bei allen vier Testfällen. Microsofts Internet Information Service v8.0 ist Dual-Stack-fähig.

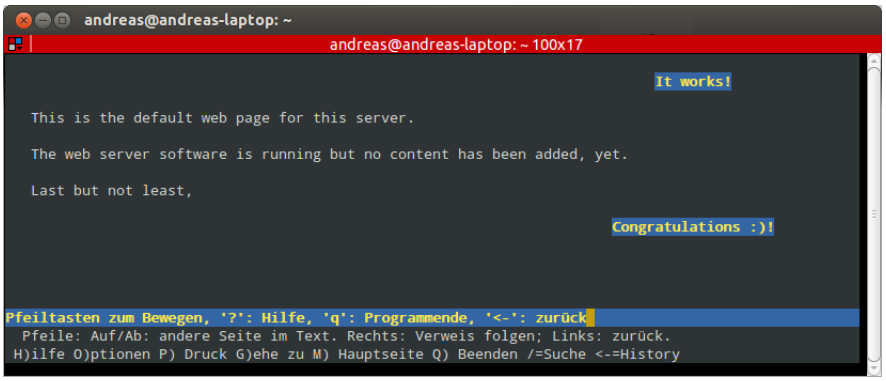


Abbildung 3.60: IIS - Servicefunktionalität

3.6.3 Clientseitige Implementierungen

Im Folgenden werden für die verschiedenen Betriebssysteme die bereits auf dem System vorinstallierten Standardbrowser getestet. Tabelle 3.8 zeigt eine Auflistung aller getesteten Webbrowser. Es wird geprüft, ob die einzelnen Webbrowser IPv6 unterstützen, welches IP-Protokoll bevorzugt genutzt wird und ob der der Happy Eyeball-Mechanismus implementiert wurde. Als Ziel wurde die Homepage der Netzlabore der Hochschule Bonn-Rhein-Sieg auf der Domain *www.netlab.inf.h-brs.de* verwendet, welche im Dual-Stack-Betrieb läuft. Für den Test des Happy Eyeball-Mechanismuses

wird der IPv6-Pfad zum Webbrowser unterbrochen.

Webbrowser	Betriebssystem
Browser v2.3.6	Android 2.3.6
Safari v6.1	iOS 6.1
Safari v6.0.2	Mac OS X 1.8
Mozilla Firefox v18.0.2	Ubuntu Desktop 12.04
Internet Explorer 8	Windows XP
Internet Explorer 9	Windows 7
Internet Explorer 10	Windows 8

Tabelle 3.8: HTTP - Liste der getesteten Webbrowser

Browser unter Android 2.3.6

Browser ist standardmäßig unter Android verfügbar. Ein Test zeigte das eine Verbindung über beide IP-Protokolle möglich ist und IPv6 bevorzugt verwendet wird. Zur Nutzung einer expliziten IPv6-Adresse, anstelle des Domainnamens, wird diese in eckigen Klammern geschrieben. Abbildung 3.61 zeigt den Kommunikationsablauf des Webbrowsers und den implementierten Happy Eyeball-Mechanismus. Nachdem der Client den Hostnamen aufgelöst hat, initiiert dieser parallel eine TCP-Verbindung über IPv4 und IPv6. Da die IPv6-Verbindung defekt ist, schlägt der Verbindungsaufbau über IPv6 fehl. Daraufhin wird die Kommunikation sofort über IPv4 fortgeführt, wodurch keine Verzögerung entsteht.

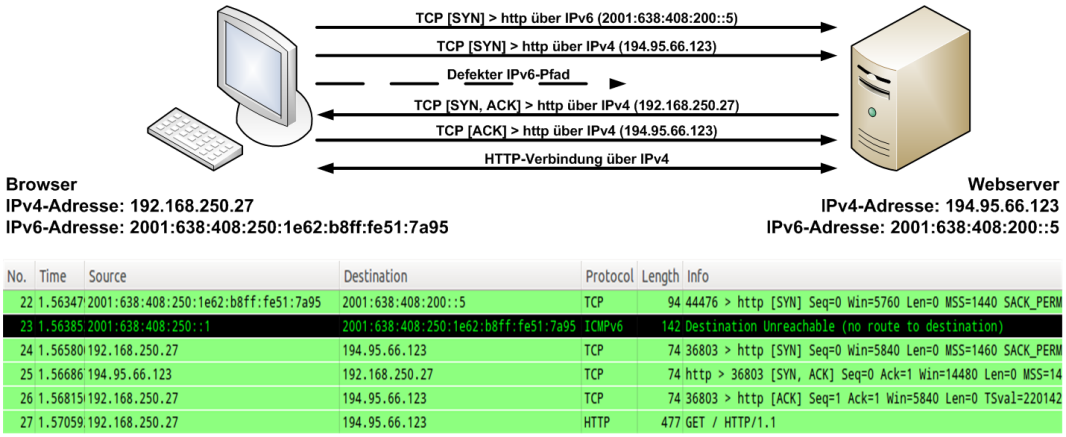


Abbildung 3.61: Browser - Verhalten bei einem defekten IPv6-Pfad

Safari unter iOS 6.1 und Mac OS X 1.8

Bei Safari handelt es sich um den Standardbrowser der Betriebssysteme iOS und Mac OS X. Ein Verbindungsaufbau zum Webserver war über beide Protokolle möglich. Wie bereits beim Android-Browser wird die IPv6-Adresse in eckigen Klammern geschrieben, sofern kein Domainname zum Aufruf der Webseite verwendet wird. Bei Erhalt eines A und AAAA Records, wird IPv6 bevorzugt verwendet. Der Happy Eyeball-Mechanismus scheint implementiert zu sein, jedoch in einer anderen Form (s. Abbildung 3.62). Es fällt auf, dass der Webbrowser zuerst über IPv4 kommuniziert und erst im späteren Verlauf der Kommunikation mehrmals versucht eine IPv6-Verbindung aufzubauen. Ist der IPv6-Pfad jedoch nicht unterbrochen, so wird vorrangig IPv6 verwendet. Anhand des Wireshark-Mitschnittes ist zu sehen, dass der Safari-Browser, nicht wie der Android-Browser, ausschließlich versucht zwei Verbindungen parallel über IPv4 und IPv6 aufzubauen, sondern von vornherein andere Daten (z.B. Routingstatistiken, RTT-Time etc.) auswertet, um das bessere IP-Protokoll für die Verbindung zu spezifizieren.

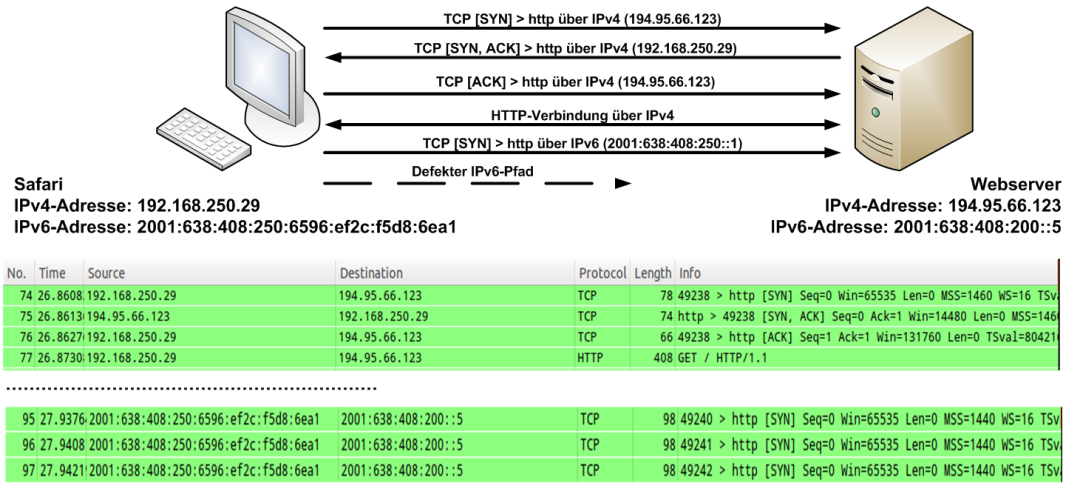
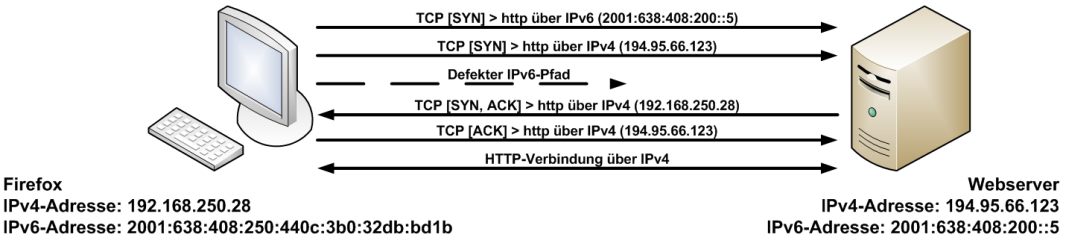


Abbildung 3.62: Safari - Verhalten bei einem defekten IPv6-Pfad

Mozilla Firefox unter Ubuntu Desktop 12.04

Unter Ubuntu Desktop 12.04 ist der Firefox Browser standardmäßig vorinstalliert und unterstützt IPv6 seit der Version 1.5. Werden keine Domainnamen verwendet, sondern die explizite IPv6-Adresse, so wird diese in eckigen Klammern geschrieben. Eine Verbindung zum Webserver ist über beide IP-Protokolle möglich. Sofern der DNS-Server eine IPv6-Adresse liefert, wird die Verbindung über IPv6 bevorzugt. Des Weiteren ist der Happy Eyeball-Mechanismus im Firefox Webbrowser implementiert. Abbildung 3.63 zeigt den Kommunikationsablauf bei einem defekten IPv6-Pfad. Dieser ist identisch zum Kommunikationsablauf des Android Browsers.

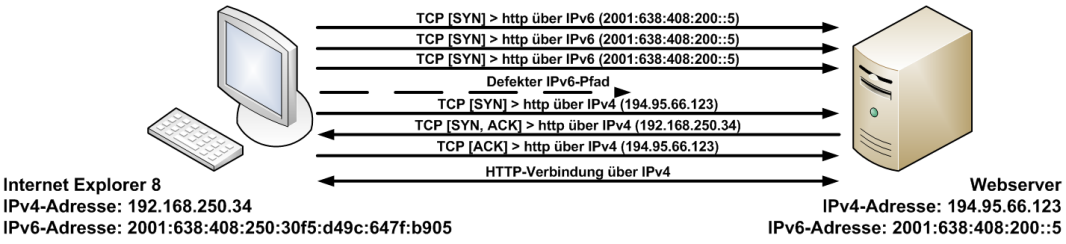


No.	Time	Source	Destination	Protocol	Length	Info
14	5.76729	2001:638:408:250:440c:3b0:32db:bd1b	2001:638:408:200::5	TCP	94	50746 > http [SYN] Seq=0 Win=14400 Len=0 MSS=1440 SACK_PERM
15	5.76768	2001:638:408:250::1	2001:638:408:250:440c:3b0:32db:bd1b	ICMPv6	142	Destination Unreachable (no route to destination)
16	5.76780	192.168.250.28	194.95.66.123	TCP	74	33907 > http [SYN] Seq=0 Win=14600 Len=0 MSS=1460 SACK_PERM
17	5.76882	194.95.66.123	192.168.250.28	TCP	74	http > 33907 [SYN, ACK] Seq=0 Ack=1 Win=14480 Len=0 MSS=146
18	5.76886	192.168.250.28	194.95.66.123	TCP	66	33907 > http [ACK] Seq=1 Ack=1 Win=14656 Len=0 TSval=160463
19	5.77963	192.168.250.28	194.95.66.123	HTTP	392	GET / HTTP/1.1

Abbildung 3.63: Firefox - Verhalten bei einem defekten IPv6-Pfad

Internet Explorer unter Windows XP

Üblicherweise ist der Standardbrowser unter Windows XP, der Internet Explorer, der in der getesteten Version 8 beide IP-Protokolle unterstützt. Die IPv6-Adresse wird in der Adresszeile in eckigen Klammern platziert. IPv6 wird bei der HTTP-Kommunikation standardmäßig bevorzugt. Eine Analyse der HTTP-Kommunikation zeigt, dass der Internet Explorer unter Windows XP über keine Happy Eyeball-Implementierung verfügt. Abbildung 3.64 zeigt, dass die Kommunikation über IPv4 erst nach 21 Sekunden erfolgt, sofern ein Verbindungsaufbau über IPv6 nicht möglich ist.

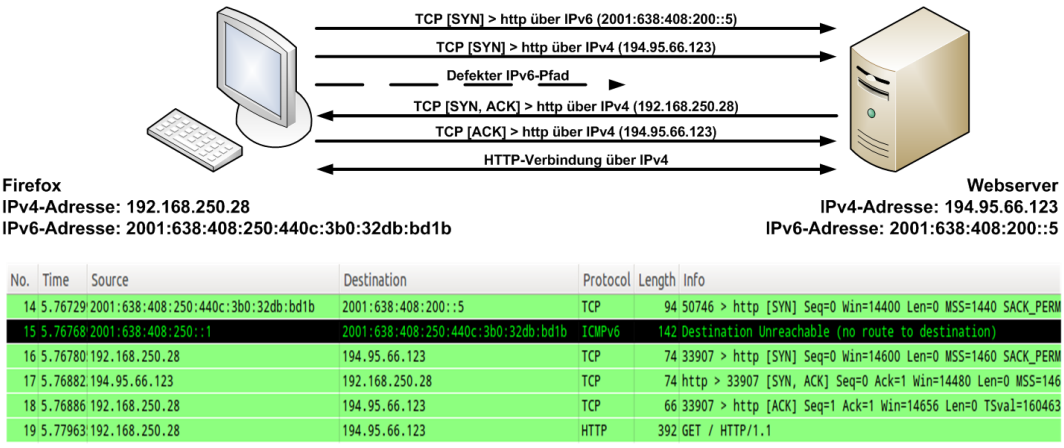


No.	Time	Source	Destination	Protocol	Length	Info
164	16.8695	2001:638:408:250:30f5:d49c:64f7:b905	2001:638:408:200::5	TCP	78	dcutility > http [SYN] Seq=0 Win=16384 Len=0 MSS=1440
167	16.8701	2001:638:408:250::1	2001:638:408:250:30f5:d49c:64f7:b905	ICMPv6	126	Destination Unreachable (no route to destination)
177	19.8145	2001:638:408:250:30f5:d49c:64f7:b905	2001:638:408:200::5	TCP	78	dcutility > http [SYN] Seq=0 Win=16384 Len=0 MSS=1440
178	19.8147	2001:638:408:250::1	2001:638:408:250:30f5:d49c:64f7:b905	ICMPv6	126	Destination Unreachable (no route to destination)
182	25.8495	2001:638:408:250:30f5:d49c:64f7:b905	2001:638:408:200::5	TCP	78	dcutility > http [SYN] Seq=0 Win=16384 Len=0 MSS=1440
183	25.8497	2001:638:408:250::1	2001:638:408:250:30f5:d49c:64f7:b905	ICMPv6	126	Destination Unreachable (no route to destination)
188	37.9210	192.168.250.34	194.95.66.123	TCP	78	ssslog-mgr > http [SYN] Seq=0 Win=65535 Len=0 MSS=1460 WS=8
189	37.9219	194.95.66.123	192.168.250.34	TCP	74	http > ssslog-mgr [SYN, ACK] Seq=0 Ack=1 Win=14480 Len=0 MS
190	37.9220	192.168.250.34	194.95.66.123	TCP	66	ssslog-mgr > http [ACK] Seq=1 Ack=1 Win=262144 Len=0 TSval=
191	37.9224	192.168.250.34	194.95.66.123	HTTP	681	GET / HTTP/1.1

Abbildung 3.64: Internet Explorer 8- Verhalten bei einem defekten IPv6-Pfad

Internet Explorer unter Windows 7/8

Der Windows Standardbrowser Internet Explorer ist, wie bei seinen Vorgängern auch, unter Windows 7 und 8 vorhanden. Beide Internet Explorer Versionen verhalten sich identisch. Wie alle anderen getesteten Browser unterstützt der Internet Explorer IPv4 und IPv6. Wird kein Domainname verwendet, so wird die IPv6-Adresse in eckigen Klammern geschrieben. Wird der Domainname verwendet und der Browser erhält als Antwort seiner DNS-Anfrage einen A und AAAA Record, wird IPv6 bevorzugt verwendet. Diese aktuelleren Internet Explorer besitzen im Gegensatz zu ihren Vorgängern ebenfalls eine Implementierung des Happy Eyeball-Mechanismus. Entgegen der Implementierung im Firefox, werden keine TCP SYN-Verbindungen parallel über IPv4 und IPv6 versendet. Auch ein erneuter Verbindungsversuch über IPv6 während einer bereits bestehenden HTTP-Kommunikation über IPv4, wie beim Safari, entfällt. Scheinbar wird beim Internet Explorer auf Grundlage bestimmter Parameter wie RTT-Time etc. bereits vorher entschieden, welches IP-Protokoll verwendet wird. Abbildung 3.65 zeigt den Kommunikationsablauf, bei einem defekten IPv6-Pfad.



No.	Time	Source	Destination	Protocol	Length	Info
14	5.76729	2001:638:408:250:440c:3b0:32db:bd1b	2001:638:408:200::5	TCP	94	50746 > http [SYN] Seq=0 Win=14400 Len=0 MSS=1440 SACK_PERM
15	5.76768	2001:638:408:250::1	2001:638:408:250:440c:3b0:32db:bd1b	ICMPv6	142	Destination Unreachable (no route to destination)
16	5.76780	192.168.250.28	194.95.66.123	TCP	74	33907 > http [SYN] Seq=0 Win=14600 Len=0 MSS=1460 SACK_PERM
17	5.76882	194.95.66.123	192.168.250.28	TCP	74	http > 33907 [SYN, ACK] Seq=0 Ack=1 Win=14480 Len=0 MSS=146
18	5.76886	192.168.250.28	194.95.66.123	TCP	66	33907 > http [ACK] Seq=1 Ack=1 Win=14656 Len=0 TSval=160463
19	5.77963	192.168.250.28	194.95.66.123	HTTP	392	GET / HTTP/1.1

Abbildung 3.65: Internet Explorer 8/9- Verhalten bei einem defekten IPv6-Pfad

3.6.4 Zusammenfassung

Beide Serverimplementierungen Apache und Microsofts Internet Information Service ermöglichen es Webinhalte über bei IP-Protokolle anzubieten. Zudem ist es ebenfalls möglich, die verschlüsselte Übertragung mittels HTTPS über IPv4 und IPv6 zu nutzen. Beide Webserver sind für den Dual-Stack-Einsatz geeignet (s. Tabelle 3.9).

Webserverimplementierung	Dual-Stack-Betrieb
Apache	Ja
Microsofts IIS	Ja

Tabelle 3.9: Testergebnisse der Webserverimplementierungen

Des Weiteren unterstützen alle getesteten Webbrowser IPv6 standardmäßig und bevorzugen dieses bei einem intakten IPv6-Pfad. Wird eine explizite IPv6-Adresse im Adressfeld des Webbrowsers verwendet, wie beispielsweise *2001:638:408:200::5*, so muss diese in eckigen Klammern stehen. Zudem unterstützt, bis auf den Internet Explorer 8, jede Implementierung den Happy Eyeball-Mechanismus, um Latenzzeiten beim Laden von Webinhalten aufgrund von defekten IPv6-Pfaden zu minimieren. Von der Nutzung des Internet Explorers 8 unter Windows XP im Dual-Stack-Betrieb ist auf Grund der hohen Latenzzeit abzuraten. Firefox, Safari und der Internet Explorer ab Vesion 9 verwenden voneinander unterschiedliche Implementierungen. Ohne ausgiebige Tests lässt sich keine Aussage darüber treffen, welche Implementierung am effizientesten ist. Fakt ist, dass alle Implementierungen ihren Zweck erfüllen und das Frustpotenzial des Anwenders deutlich reduzieren. Tabelle 3.10 stellt das Ergebnis übersichtlich dar.

Webbrowser	IP-Support	Priorität	Happy Eyeball
Browser	IPv4/IPv6	IPv6	Ja
Firefox	IPv4/IPv6	IPv6	Ja
Safari	IPv4/IPv6	IPv6	Ja
Internet Explorer 8	IPv4/IPv6	IPv6	Nein
Internet Explorer 9/10	IPv4/IPv6	IPv6	Ja

Tabelle 3.10: Testergebnisse der Webbrowser

Alle getesteten Webbrowser sind Dual-Stack-fähig und können in einer Dual-stack-Umgebung problemlos eingesetzt werden. Da der Internet Explorer zwar Dual-Stack-fähig ist, aber der Happy Eyeball-Mechanismus in der Version 8 nicht implementiert ist und IE 9 und 10 unter Windows XP nicht nutzbar sind, ist zu empfehlen hier einen alternativen Browser, wie Firefox zu verwenden.

3.7 File Transfer Protocol (FTP)

FTP verwendet im Protokollheader keine IPv4-Adressen, eine Neuspezifikation des Protokolls ist somit nicht notwendig. Jedoch verwenden die in RFC 959 spezifizierten FTP-Befehle *PASV* und *PORT* als Argumente IPv4-Adressen zur Initialisierung von aktiven oder passiven Datenübertragungen [PR85]. Eine Erweiterung oder Anpassung der Befehle für die Verwendung von IPv6 ist deshalb erforderlich.

3.7.1 Dual-Stack-Erweiterungen

Die Erweiterungen zur Unterstützung von IPv6 sind im RFC 2428 - *"FTP Extensions for IPv6 and NATs"* enthalten [AO98]. Die dort spezifizierten Erweiterungen des Befehlssatzes von FTP ermöglichen den Dual-Stack-Betrieb. Zur Nutzung beider IP-Protokolle sind die FTP-Kommandos PORT und PASV durch die Kommandos EPRT und EPSV jeweils ausgetauscht worden. Im Gegensatz zum PORT-Kommando, ist es mit Hilfe des EPRT-Kommandos möglich eine aktive Verbindung zum FTP-Server über IPv4 oder IPv6 herzustellen. Die aktive Verbindung über IPv6 wird durch das EPRT-Kommando vom Client initiiert (s. Abbildung 3.66).

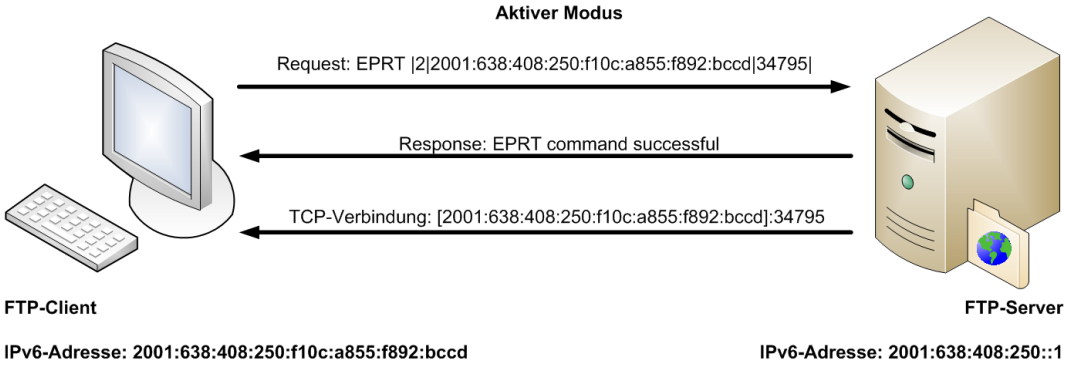


Abbildung 3.66: EPRT-Kommando für IPv6

Die passive Datenübertragung über IPv6 wird durch das Kommando EPSV vom Client eingeleitet (s. Abbildung 3.67). Wird anstelle einer IPv6-Adresse eine IPv4-Adresse verwendet, so ist das EPRT-Kommando *EPRT |1|192.168.250.1|6275|*. Das EPSV-Kommando bleibt identisch [AO98]. Aus Gründen der Abwärtskompatibilität können für IPv4 weiterhin die Kommandos PORT und PASV verwendet werden. Durch die neuen Befehle ist FTP protokollunabhängig. In Zukunft können somit durch geringe Anpassungen, weitere Protokolle der Netzwerkschicht genutzt werden.

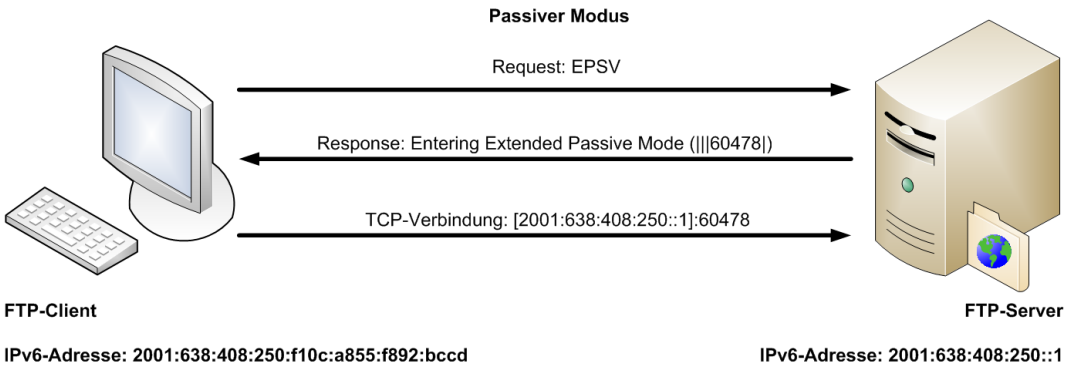


Abbildung 3.67: EPSV-Kommando für IPv6

Wie bei HTTP und vielen weiteren Protokollen die TCP als Transportprotokoll nutzen, spielt auch bei FTP der Happy Eyeball-Mechanismus eine wichtige Rolle (s. Kapitel 3.6).

3.7.2 Serverseitige Implementierungen

Im Folgenden werden jeweils eine Serverimplementierungen unter Ubuntu Server 12.04 und Windows Server 2012 getestet. Unter Ubuntu Server 12.04 wird eine der unter Linux meist verbreitesten FTP-Implementierungen *ProFTPD* in der Version 1.3.4a untersucht. Unter Windows Server 2012 wird der von Microsoft standardmäßig integrierte *Microsoft FTP Service* überprüft.

ProFTPD

Bei ProFTPd handelt es sich um eine FTP-Implementierung für Linux-Systeme. ProFTPD unterstützt ab Version 1.3.x IPv6 und implementiert nach Angaben von [Pro13b] den RFC 2428. ProFTPD wird zunächst auf dem Ubuntu Server 12.04 Testsystem installiert.

Installation und Konfiguration

Da ProFTPD im Ubuntu-Repository enthalten ist, muss lediglich das Paket *proftpd-basic* installiert werden. Dies erfolgt über den Befehl:

```
root@ubuntudualstack:~# apt-get install proftpd-basic
```

Abbildung 3.68: ProFTPD - Installation

Im nächsten Schritt erfolgt die Konfiguration des Services. Für die grundlegende Konfiguration des Services wird [Pro13a] empfohlen. Für den Dual-Stack-Betrieb muss

folgender Eintrag in der Konfigurationsdatei */etc/proftpd/proftpd.conf* vorgenommen werden:

```
UseIPv6                         on
```

Abbildung 3.69: ProFTPD - Konfiguration

Nach einem Neustart des Daemons durch den Befehl */etc/init.d/proftpd restart* wird die neue Konfiguration eingelesen. Ist es notwendig den Service an bestimmte IPv4- /IPv6-Adressen zu binden, sind die in Abbildung 3.70 dargestellten Parameter der Konfigurationsdatei hinzuzufügen.

```
DefaultAddress          192.168.250.1 2001:638:408:250::1
SocketBindTight         on
```

Abbildung 3.70: ProFTPD - IP-Bindung

Testphase

Im ersten Schritt wird die Serviceverfügbarkeit getestet. Mit Hilfe von *netstat -lnp|grep :21* wird die Ausgabe nach Port 21 gefiltert, auf dem der FTP-Dienst standardmäßig lauscht. Im Falle der Nutzung von ProFTPD fällt auf, wie in Abbildung 3.71 zu sehen, dass der Dienst lediglich auf allen IPv6-Schnittstellen horcht. Dies erweckt zunächst den Anschein, dass der Service nur über IPv6 erreichbar ist.

```
root@ubuntudualstack:~# netstat -lnp|grep :21
tcp6       0      0 :::21                   :::*                    LISTEN      14364/proftpd
```

Abbildung 3.71: ProFTPD - Serviceverfügbarkeit

Der Grund dafür ist, dass ProFTPD intern mit IPv4-mapped IPv6 Adressen arbeitet, welche in Kapitel 2 genauer erläutert wurden. Somit ist der Dienst in der Lage jede Anfrage, unabhängig von der IP-Version, über einen tcp6-Port abzuwickeln. Dieser Umstand wird in Abbildung 3.72 dargestellt, wo ein Verbindungsaufbau via IPv4 über einen tcp6-Port stattgefunden hat.

```
root@ubuntudualstack:~# netstat -lntap|grep :21
tcp6       0      0 :::21                   :::*                    LISTEN      14364/proftpd
tcp6       0      0 192.168.250.1:21        192.168.250.20:39961    VERBUNDEN   14645/proftpd
```

Abbildung 3.72: ProFTPD - IPv4-Konnektivität über tcp6-Port

Zur Prüfung der Servicefunktionalität wird das Kommandozeilenprogramm ftp verwendet. Mit Hilfe von ftp wird zuerst versucht, eine Verbindung zum FTP-Server

über IPv4 aufzubauen. Abbildung 3.73 zeigt den erfolgreichen Verbindungsaufbau zum Server.

```
andreas@andreas-laptop:~$ ftp 192.168.250.1
Connected to 192.168.250.1.
220 ProFTPD 1.3.4a Server (FTP-Dual-Stack-Server) [::ffff:192.168.250.1]
```

Abbildung 3.73: ProFTPD - Servicefunktionalität über IPv4

Ein weiterer Test mit Hilfe von ftp, zeigt den erfolgreichen Verbindungsaufbau über das IPv6-Protokoll (s. Abbildung 3.74).

```
andreas@andreas-laptop:~$ ftp 2001:638:408:250::1
Connected to 2001:638:408:250::1.
220 ProFTPD 1.3.4a Server (FTP-Dual-Stack-Server) [2001:638:408:250::1]
```

Abbildung 3.74: ProFTPD - Servicefunktionalität über IPv6

ProFTPD stellt den Service für beide IP-Protokolle zur Verfügung. Grund dafür ist, dass beim Server die Befehle EPRT und EPSV implementiert sind. Abbildung 3.75 zeigt, wie der Client mit Hilfe des Befehls *FEAT* den Server nach seinem Befehlssatz anfragt. Die FTP-Implementierung ProFTPD ist Dual-Stack-fähig.

No.	Time	Source	Destination	Protocol	Length	Info
68	72.0145	2001:638:408:250:f10c:a855:f892:bccd	2001:638:408:250::1	FTP	92	Request: FEAT
69	72.0148	2001:638:408:250::1	2001:638:408:250:f10c:a855:f892:bccd	FTP	101	Response: 211-Features:
70	72.0148	2001:638:408:250::1	2001:638:408:250:f10c:a855:f892:bccd	FTP	93	Response: EPRT
71	72.0148	2001:638:408:250::1	2001:638:408:250:f10c:a855:f892:bccd	FTP	93	Response: EPSV

Abbildung 3.75: ProFTPD - Anfrage des Serverbefehlssatzes

Microsoft FTP Service

Unter Windows Server 2012 wird die FTP-Serverimplementierung *Microsoft FTP Service* getestet.

Installation und Konfiguration

Der Microsoft FTP Service lässt sich über den integrierten Server-Manager installieren. Voraussetzung ist, dass der Dienst *Webserver (IIS)* bereits auf dem System installiert ist. Zur Installation des FTP-Services sind folgende Schritte notwendig:

- Server-Manager –> Verwalten –> Rollen und Features hinzufügen
- Installationstyp: *Rollenbasierte oder featurebasierte Installation* auswählen
- Serverauswahl: Auswählen des Servers, auf dem der Service installiert werden soll

- Serverrollen: FTP-Server auswählen (unter Webserver (IIS))
- Features: keine Auswahl

Nachdem Installationsprozess, ist ein Neustart des Servers notwendig. Für die grundlegende Konfiguration des Services wird [Fra13] empfohlen. Damit der Microsoft FTP Service Dual-Stack-tauglich wird, ist es ausreichend bei der Erstkonfiguration einer FTP-Freigabe unter Bindungs- und SSL-Einstellungen den in Abbildung 3.76 dargestellten Parameter zu übernehmen. Daraufhin wird der Service an alle Netzwerkschnittstellen gebunden und ist somit über alle verfügbaren IPv4-/IPv6-Adressen des Servers erreichbar.

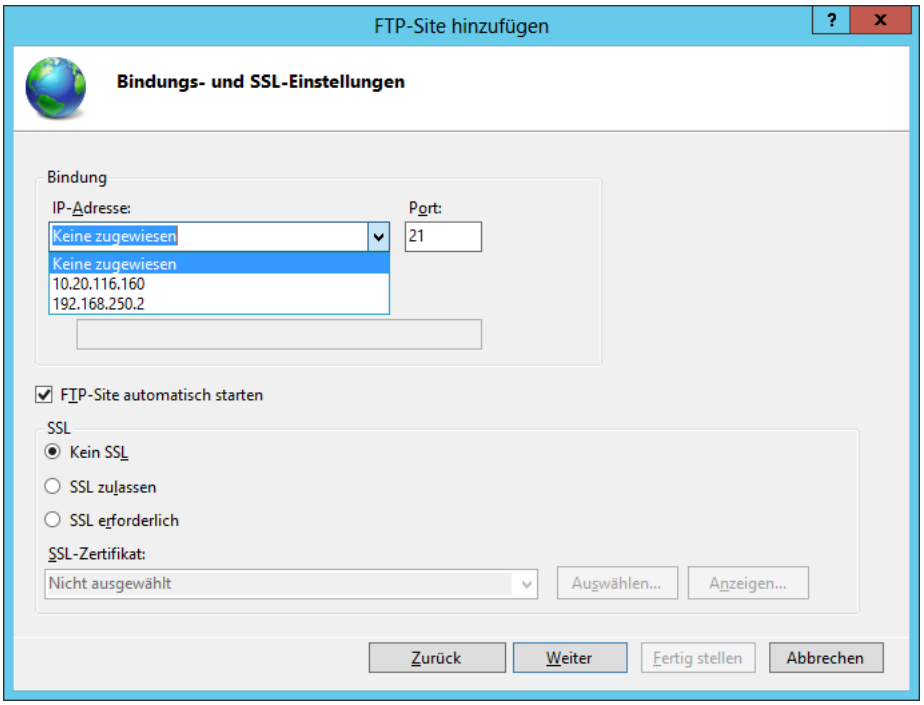


Abbildung 3.76: Microsoft FTP Service - Konfiguration

Ist es erforderlich den Service nur über bestimmte IP-adressen zur Verfügung zu stellen, geschieht dies folgendermaßen:

- Start –> Informationsdienste (IIS)-Manager ausführen
- Server auswählen –> Sites
- (Rechtsklick) auf die gewünschte FTP-Freigabe –> *Bindungen bearbeiten...* auswählen

Abbildung 3.77 stellt das Interface für Sitebindungen dar. Dort lassen sich gezielt IPv4- und IPv6-Adressen binden.

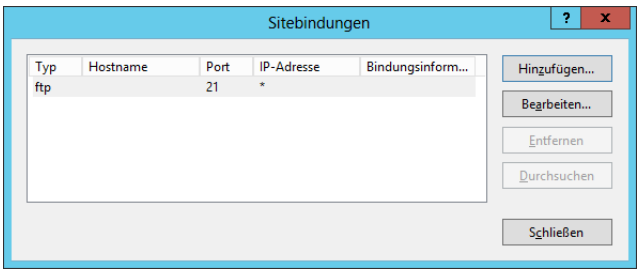


Abbildung 3.77: Microsoft FTP Service - IP-Bindung

In Abbildung 3.78 wird der FTP-Service beispielhaft an die IPv6-Adresse *2001:638:408:250::2* gebunden. Die Änderungen werden ohne Neustart des Services wirksam.

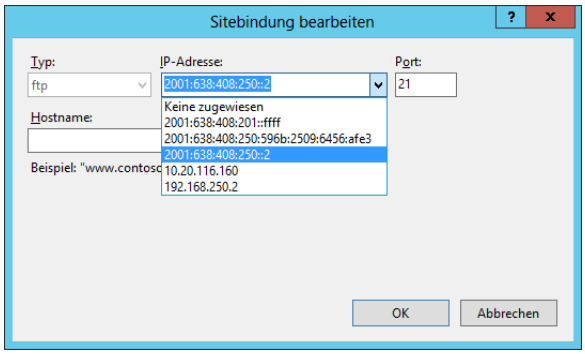


Abbildung 3.78: Microsoft FTP Service - IP-Bindung hinzufügen

Testphase

In der ersten Phase wird die Serviceverfügbarkeit diagnostiziert. Abbildung 3.79 zeigt, dass der Service über alle IPv4-/IPv6-Adressen verfügbar ist.

```
Aktive Verbindungen
  Proto  Lokale Adresse          Remoteadresse          Status
  TCP    0.0.0.0:21             0.0.0.0:0              ABH?REN
  TCP    [::]:21                [::]:0                 ABH?REN
```

Abbildung 3.79: Microsoft FTP Service - Serviceverfügbarkeit

In Phase zwei erfolgt nun die Überprüfung der Servicefunktionalität. Abbildung 3.80 stellt einen erfolgreichen Verbindungsaufbau über IPv4 dar.

```
andreas@andreas-laptop:~$ ftp 192.168.250.2
Connected to 192.168.250.2.
220 Microsoft FTP Service
```

Abbildung 3.80: Microsoft FTP Service - Servicefunktionalität über IPv4

Ein weiterer Test zeigt den erfolgreichen Verbindungsaufbau über IPv6 (s. Abbildung 3.81). Die FTP-Implementierung Microsoft FTP Service stellt den Dienst über beide IP-Protokolle zur Verfügung. Beide Befehle EPRT und EPSV, werden vom Server unterstützt. Die FTP-Serverimplementierung Microsoft FTP Service ist Dual-Stack-fähig.

```
andreas@andreas-laptop:~$ ftp 2001:638:408:250::2
Connected to 2001:638:408:250::2.
220 Microsoft FTP Service
```

Abbildung 3.81: Microsoft FTP Service - Servicefunktionalität über IPv6

3.7.3 Clientseitige Implementierungen

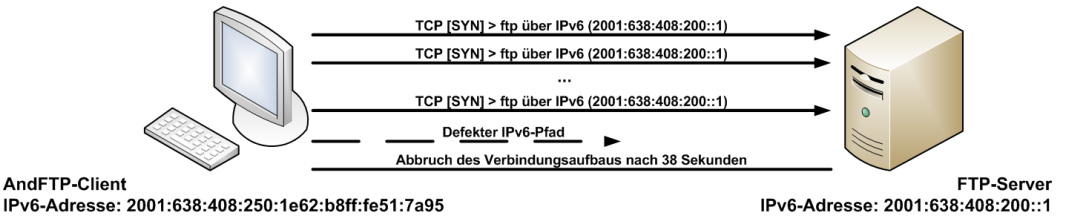
In diesem Kapitel wird für jedes Betriebssystem die weitverbreitetsten FTP-Clients auf ihre Dual-Stack-Fähigkeit getestet. Es wird geprüft, ob beide IP-Protokolle unterstützt werden und welches bevorzugt verwendet wird. Des Weiteren wird getstet ob die Clients über eine Implementierung des Happy Eyeball-Mechanismuses verfügen. Als Gegenstelle dient der FTP-Server *ftp.netlab.inf.h-brs.de*. Tabelle 3.11 zeigt eine Auflistung der getesteten FTP-Clients.

FTP-Client	Betriebssystem
AndFTP v2.3	Android
FTP Sprite Free v1.1.3	iOS
FileZilla v3.6	Mac OS X, Ubuntu, Windows 7/8/XP

Tabelle 3.11: FTP - Liste der getesteten Clients

AndFTP unter Android

Die im Google Play Store beliebteste Clientimplementierung ist AndFTP (Stand: Januar 2013). Getestet wurde die Version 3.2. AndFTP unterstützt nach Angaben des Entwicklers Lysesoft IPv6 seit Version 2.3 und wurde explizit um den Befehl EPSV erweitert [And13]. Für den Dual-Stack-Betrieb sind keine speziellen Konfigurationen notwendig. Beide IP-Protokollen werden unterstützt. IPv6-Adressen werden nach der Namensauflösung bevorzugt verwendet. Eine explizite Angabe einer IPv6-Adresse geschieht in eckigen Klammern. Abbildung 3.82 zeigt, dass der FTP-Client über keine Happy Eyeball-Implementierung verfügt. Der Verbindungsversuch wird nach 38 Sekunden abgebrochen, ohne das ein Versuch über IPv4 erfolgt. AndFTP ist aufgrund einer fehlenden Happy Eyeball-Implementierung nicht für den Dual-Stack-Betrieb geeignet.



No.	Time	Source	Destination	Protocol	Length	Info
33	12.5266	2001:638:408:250:1e62:b8ff:fe51:7a95	2001:638:408:200::1	TCP	94	35598 > ftp [SYN] Seq=0 Win=5760 Len=0 MSS=1440 SACK_PERM=1
39	15.5253	2001:638:408:250:1e62:b8ff:fe51:7a95	2001:638:408:200::1	TCP	94	35598 > ftp [SYN] Seq=0 Win=5760 Len=0 MSS=1440 SACK_PERM=1
46	20.1425	2001:638:408:250:1e62:b8ff:fe51:7a95	2001:638:408:200::1	TCP	94	51238 > ftp [SYN] Seq=0 Win=5760 Len=0 MSS=1440 SACK_PERM=1
51	21.5409	2001:638:408:250:1e62:b8ff:fe51:7a95	2001:638:408:200::1	TCP	94	35598 > ftp [SYN] Seq=0 Win=5760 Len=0 MSS=1440 SACK_PERM=1
54	23.1500	2001:638:408:250:1e62:b8ff:fe51:7a95	2001:638:408:200::1	TCP	94	51238 > ftp [SYN] Seq=0 Win=5760 Len=0 MSS=1440 SACK_PERM=1
72	29.1659	2001:638:408:250:1e62:b8ff:fe51:7a95	2001:638:408:200::1	TCP	94	51238 > ftp [SYN] Seq=0 Win=5760 Len=0 MSS=1440 SACK_PERM=1
84	41.1993	2001:638:408:250:1e62:b8ff:fe51:7a95	2001:638:408:200::1	TCP	94	53069 > ftp [SYN] Seq=0 Win=5760 Len=0 MSS=1440 SACK_PERM=1
101	44.2047	2001:638:408:250:1e62:b8ff:fe51:7a95	2001:638:408:200::1	TCP	94	53069 > ftp [SYN] Seq=0 Win=5760 Len=0 MSS=1440 SACK_PERM=1
117	50.2127	2001:638:408:250:1e62:b8ff:fe51:7a95	2001:638:408:200::1	TCP	94	53069 > ftp [SYN] Seq=0 Win=5760 Len=0 MSS=1440 SACK_PERM=1

Abbildung 3.82: AndFTP - Verhalten bei einem defekten IPv6-Pfad

FTP Sprite Free unter iOS

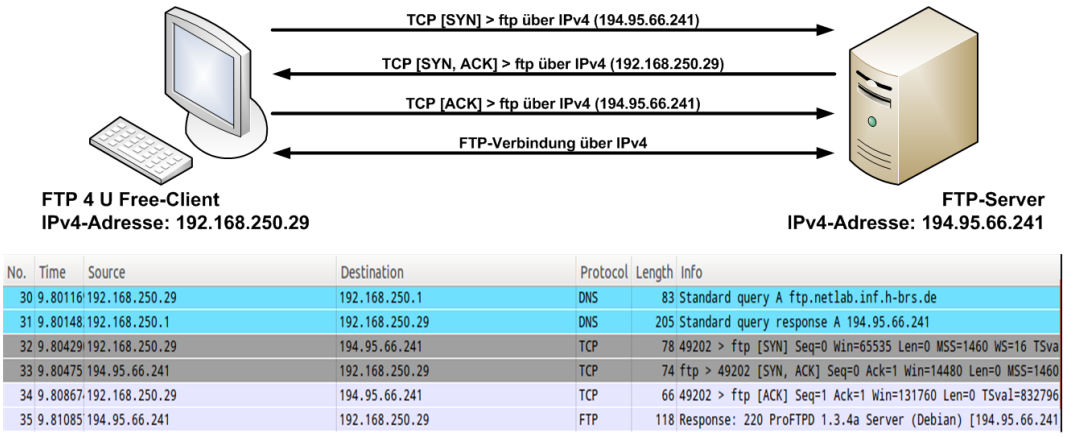
Unter iOS 6.1 wurde der im App Store beliebteste FTP-Client FTP Sprite Free v1.1.3 getestet (Stand: Januar 2013). Der FTP-Client wird über den App Store heruntergeladen und installiert. Eine Konfiguration für den Dual-Stack-Betrieb ist nicht vorhanden und wie das Testergebnis zeigt auch nicht notwendig. FTP Sprite Free ermöglicht nur den Verbindungsaufbau über IPv4. Die Servicefunktionalität über IPv6 ist nicht gegeben. Ein Test in dem die IPv6-Adressen in eckigen Klammern eingegeben wurde, schlug fehl. Grund dafür ist die fehlerhafte Interpretation der IPv6-Adresse, welche die Anwendung als Hostname interpretiert und somit den Service nicht adressieren kann (s. Abbildung 3.83).

Abbildung 3.83: FTP Sprite Free - Fehlinterpretation der Anwendung

Ftp 4U-Free unter iOS

Als Alternative wurde der FTP-Client Ftp 4U-Free in der Version 1.3 getestet. FTP 4U-Free unterstützt beide IP-Protokolle. Wird jedoch ein Domainname anstelle der expliziten IPv6-Adresse verwendet, so wird ausschließlich IPv4 verwendet. Der Grund dafür ist, dass der DNS-Resolver der Anwendung, keinen AAAA Record anfragt

(s. Abbildung 3.84). Ein Test zur Überprüfung des Happy Eyeball-Mechanismus ist somit nicht relevant.

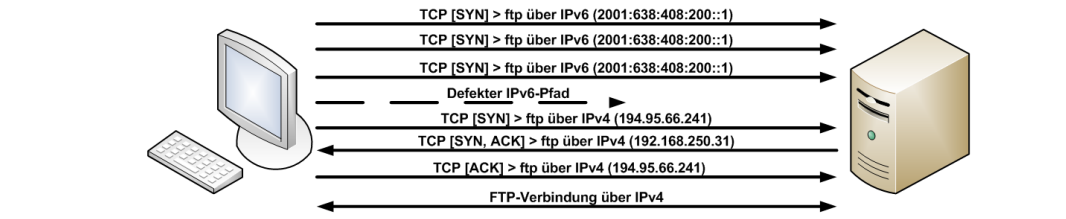


No.	Time	Source	Destination	Protocol	Length	Info
30	9.80116	192.168.250.29	192.168.250.1	DNS	83	Standard query A ftp.netlab.inf.h-brs.de
31	9.80148	192.168.250.1	192.168.250.29	DNS	205	Standard query response A 194.95.66.241
32	9.80429	192.168.250.29	194.95.66.241	TCP	78	49202 > ftp [SYN] Seq=0 Win=65535 Len=0 MSS=1460 WS=16 TSva
33	9.80475	194.95.66.241	192.168.250.29	TCP	74	ftp > 49202 [SYN, ACK] Seq=0 Ack=1 Win=14480 Len=0 MSS=1460
34	9.80867	192.168.250.29	194.95.66.241	TCP	66	49202 > ftp [ACK] Seq=1 Ack=1 Win=131760 Len=0 TSval=832796
35	9.81085	194.95.66.241	192.168.250.29	FTP	118	Response: 220 ProFTPD 1.3.4a Server (Debian) [194.95.66.241

Abbildung 3.84: Ftp 4U-Free - Fehlverhalten des DNS-Resolvers

FileZilla unter Mac OS X, Ubuntu, Windows 7/8

FileZilla ist nach [Hei13] der beliebteste FTP-Client für die oben genannten Betriebssysteme. Somit ist es nicht erforderlich für jedes Betriebssystem einen anderen FTP-Client zu nutzen, sondern diese Synergie zu nutzen. Getestet wurde FileZilla in der Version 3.6. Filezilla lässt sich über [Hei13] herunterladen und anschließend installieren. Nach Angaben des Entwicklers unterstützt der FTP-Client bereits seit Version 3.1.0-beta1 IPv6 [Fil13b]. Beide IP-Protokolle werden unterstützt und IPv6 wird priorisiert. Für die Nutzung einer explixiten IPv6-Adresse ist es erforderlich, diese in eckigen Klammern zu schreiben. Abbildung 3.85 zeigt, das Filezilla über eine Happy Eyeball-Implementierung verfügt.

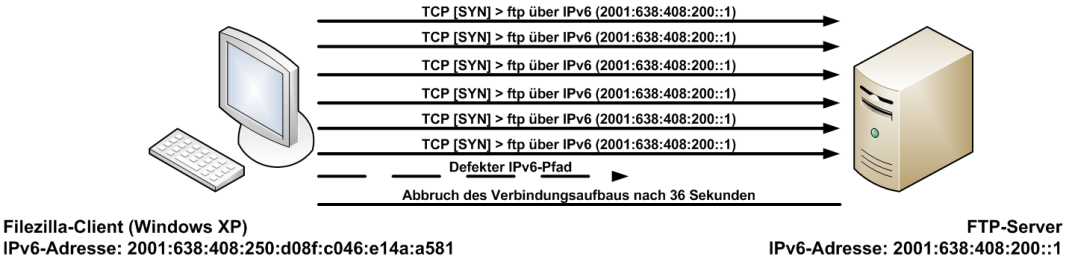


No.	Time	Source	Destination	Protocol	Length	Info
16	2.35345	2001:638:408:250:65e2:c4a9:d728:fce8	2001:638:408:200::1	TCP	98	49184 > ftp [SYN] Seq=0 Win=65535 Len=0 MSS=1440 WS=16 TSva
17	2.35348	2001:638:408:250::1	2001:638:408:250:65e2:c4a9:d728:fce8	ICMPv6	146	Destination Unreachable (no route to destination)
19	3.35566	2001:638:408:250:65e2:c4a9:d728:fce8	2001:638:408:200::1	TCP	98	49184 > ftp [SYN] Seq=0 Win=65535 Len=0 MSS=1440 WS=16 TSva
20	3.35599	2001:638:408:250::1	2001:638:408:250:65e2:c4a9:d728:fce8	ICMPv6	146	Destination Unreachable (no route to destination)
21	4.46379	2001:638:408:250:65e2:c4a9:d728:fce8	2001:638:408:200::1	TCP	98	49184 > ftp [SYN] Seq=0 Win=65535 Len=0 MSS=1440 WS=16 TSva
22	4.46412	2001:638:408:250::1	2001:638:408:250:65e2:c4a9:d728:fce8	ICMPv6	146	Destination Unreachable (no route to destination)
23	4.46537	192.168.250.31	194.95.66.241	TCP	78	49185 > ftp [SYN] Seq=0 Win=65535 Len=0 MSS=1460 WS=16 TSva
24	4.46573	194.95.66.241	192.168.250.31	TCP	74	ftp > 49185 [SYN, ACK] Seq=0 Ack=1 Win=14480 Len=0 MSS=1460
25	4.46647	192.168.250.31	194.95.66.241	TCP	66	49185 > ftp [ACK] Seq=1 Ack=1 Win=131760 Len=0 TSval=846717
26	4.46863	194.95.66.241	192.168.250.31	FTP	118	Response: 220 ProFTPD 1.3.4a Server (Debian) [194.95.66.241

Abbildung 3.85: FileZilla - Verhalten bei einem defekten IPv6-Pfad

FileZilla unter Windows XP

Anders als unter Mac OS X, Ubuntu und Windows 7/8, verfügt die Version für Windows XP über keine Implementierung des Happy Eyeball-Mechanismuses (s. Abbildung 3.86). Der Verbindungsversuch wurde nach 36 Sekunden abgebrochen, ohne dass ein Versuch über IPv4 initialisiert wurde. Alle anderen Eigenschaften stellten sich als identisch heraus.



No.	Time	Source	Destination	Protocol	Length	Info
20	3.93354	2001:638:408:250:d08f:c046:e14a:a581	2001:638:408:200::1	TCP	78	cap > ftp [SYN] Seq=0 Win=16384 Len=0 MSS=1440
24	4.03848	2001:638:408:250::1	2001:638:408:250:d08f:c046:e14a:a581	ICMPv6	126	Destination Unreachable (no route to destination)
27	7.15162	2001:638:408:250:d08f:c046:e14a:a581	2001:638:408:200::1	TCP	78	cap > ftp [SYN] Seq=0 Win=16384 Len=0 MSS=1440
28	7.15195	2001:638:408:250::1	2001:638:408:250:d08f:c046:e14a:a581	ICMPv6	126	Destination Unreachable (no route to destination)
36	13.7138	2001:638:408:250:d08f:c046:e14a:a581	2001:638:408:200::1	TCP	78	cap > ftp [SYN] Seq=0 Win=16384 Len=0 MSS=1440
37	13.7141	2001:638:408:250::1	2001:638:408:250:d08f:c046:e14a:a581	ICMPv6	126	Destination Unreachable (no route to destination)
61	29.8721	2001:638:408:250:d08f:c046:e14a:a581	2001:638:408:200::1	TCP	78	1027 > ftp [SYN] Seq=0 Win=16384 Len=0 MSS=1440
62	29.8723	2001:638:408:250::1	2001:638:408:250:d08f:c046:e14a:a581	ICMPv6	126	Destination Unreachable (no route to destination)
66	33.0727	2001:638:408:250:d08f:c046:e14a:a581	2001:638:408:200::1	TCP	78	1027 > ftp [SYN] Seq=0 Win=16384 Len=0 MSS=1440
67	33.0730	2001:638:408:250::1	2001:638:408:250:d08f:c046:e14a:a581	ICMPv6	126	Destination Unreachable (no route to destination)
74	39.6351	2001:638:408:250:d08f:c046:e14a:a581	2001:638:408:200::1	TCP	78	1027 > ftp [SYN] Seq=0 Win=16384 Len=0 MSS=1440
75	39.6354	2001:638:408:250::1	2001:638:408:250:d08f:c046:e14a:a581	ICMPv6	126	Destination Unreachable (no route to destination)

Abbildung 3.86: FileZilla unter Windows XP- Verhalten bei einem defekten IPv6-Pfad

3.7.4 Zusammenfassung

Tabelle 3.12 zeigt das beide getesteten FTP-Serverimplementierungen Dual-Stack-fähig sind. Dies bedeutet das sowohl ProFTPd unter Ubuntu Server 12.04, als auch der Microsoft FTP Service unter Windows Server 2012 für eine Dual-Stack-Umgebung einsetzbar sind. Beide Implementierungen unterstützen die Befehle EPRT und EPSV.

FTP-Serverimplementierung	Dual-Stack-Betrieb
ProFTPD	Ja
Microsoft FTP Service	Ja

Tabelle 3.12: Testergebnisse der FTP-Serverimplementierungen

Tabelle 3.13 hingegen zeigt, dass nicht alle FTP-Clients Dual-Stack-fähig sind. Der FTP-Client FTP Sprite Free ist aufgrund des fehlenden IPv6-Supports nicht im Dual-Stack-Betrieb einsetzbar. Des Weiteren sollten die Clients AndFTP, FTP 4U-Free und FileZilla (unter Windows XP), aufgrund einer fehlenden Happy Eyeball-Implementierung nicht eingesetzt werden. FileZilla unter Mac OS X 1.8, Ubuntu

Desktop 12.04 und Windows 7/8 lässt sich hingegen bedenkenlos in einer Dual-Stack-Umgebung einsetzen.

FTP-Client	IP-Support	Priorität	Happy Eyeball
AndFTP	IPv4/IPv6	IPv6	Nein
FTP Sprite Free	IPv4	-	-
FTP 4U Free	IPv4/IPv6	IPv4	Nein
FileZilla (WinXP)	IPv4/IPv6	IPv6	Nein
FileZilla	IPv4/IPv6	IPv6	Ja

Tabelle 3.13: Testergebnisse der FTP-Clients

3.8 Analyse der Testergebnisse

Auf Grundlage der Testergebnisse können Aussagen darüber getroffen werden, welche Änderungen und optionalen Erweiterungen für einen Service im Dual-Stack-Betrieb erforderlich sind. Wie aufwändig diese Änderungen sind, ist dadurch bedingt wie stark der Service vom IPv4-Protokoll abhängig ist. DHCP beispielsweise weist eine sehr starke IPv4-Abhängikeit auf. FTP und DNS besitzen eine vergleichsweise geringe Abhängigkeit und HTTP ist IPv4 unabhängig. Aus den gewonnen Erfahrungen konnten die folgenden Anpassungen für einen Service ermittelt werden, welche die Entwicklung des Leitfadens im nächsten Kapitel ermöglichen.

- Neuspezifikation des Services

Ist ein Service stark IPv4-abhängig, so muss der Service neu spezifiziert werden. Dies ist beipielsweise bei DHCP der Fall, welches im Header 32-Bit große Felder für IPv4-Adressen verwendet, wodurch IPv6-Adressen nicht genutzt werden können. Das Resultat ist, dass es bei der Autokonfiguration von Hosts nicht möglich ist, das bereits für IPv4 im RFC 2132 spezifizierte Protokoll DHCP ebenfalls für IPv6 zu verwenden. Deshalb wurde das Protokoll für IPv6 neu spezifiziert. Dies hat den Vorteil, dass eine vollkommen neue Architektur eines Services spezifiziert werden kann. Dadurch können leichter neue Funktionalitäten integriert und eventuell vorhandene Designschwächen der alten Protokollarchitektur beseitigt werden. Bei einer Neuspezifikation eines Services sollte geprüft werden, ob jeweils eine unterschiedliche Serviceimplementierung für IPv4 und IPv6 erforderlich ist.

- Erweiterung des Protokollbefehlssatzes

Für einige Services, wie beipsielsweise FTP war es nicht notwendig, das Protokoll neu zu spezifizieren. Eine Erweiterung des Befehlssatzes des Protokolls war für FTP ausreichend. Die Erweiterung des Befehlssatzes ist erforderlich, wenn die Protokolle untereinander im Nutzdatenteil (nicht im Header) IP-Adressen nutzen, um bestimmte IP-Versionsabhängige Optionen abzufragen oder auch neue Funktionen bereitzustellen. Neue Befehle und Funktionen werden in den jeweiligen RFC's spezifiziert. FTP beispielsweise, benötigt die neuen Befehle *EPRT* und *EPSV*, um eine aktive oder passive Datenübertragung über IPv6 zu initialisieren. Zuvor wurden bei FTP die Befehle *PORT* und *PSV* genutzt, welche IPv4-Adressen verwenden. Verfügen die Server- und Clientimplementierungen jedoch nicht über den selben Befehlssatz, so sind diese nicht in der Lage, untereinander im Dual-Stack-Betrieb zu kommunizieren. Des Weiteren ist es erforderlich, den Service abwärtskompatibel zu halten, indem dieser auch alte IPv4-only Befehle akzeptiert. Dadurch ist sichergestellt, dass IPv4-only Clients einen Service im Dual-Stack-Betrieb auch weiterhin ausschließlich über IPv4 nutzen können.

- Syntax Problem

Obwohl die korrekte Nutzung von IPv6-Adressen, zur Vermeidung des Syntaxproblems, im RFC 2732 spezifiziert ist, zeigten die Testergebnisse, dass einige Anwendungen die IPv6-Adressen nicht korrekt interpretieren. Im Falle des FTP-Clients *FTP Sprite Free* unter iOS, wurde die IPv6-Adresse als DNS-Name interpretiert. Eine solche Fehlinterpretation der IPv6-Adresse verhindert einen erfolgreichen Kommunikationsaufbau.

- Dual-Stack-Service über IPv6-Sockets

Wie bei den beiden FTP-Serverimplementierungen zu sehen war, nutzen einige Serverimplementierungen IPv4-mapped IPv6-Adressen. Mit Hilfe dieser Adressen sind Services in der Lage, eingehende IPv4-Verbindungen über IPv6-Sockets zu verarbeiten. Dies hat unter anderem bei Linux-Umgebungen den Vorteil, das nur ein Prozess bzw. Daemon gestartet wird und somit in der Regel auch nur eine Konfigurationsdatei für den Dual-Stack-Betrieb eines Services erforderlich ist [MM05, S. 198f.]. Des Weiteren müssen weniger Sockets erstellt werden, womit Systemressourcen gespart werden. Ein Service kann dadurch auschließlich mit Hilfe von IPv6-Sockets programmiert werden, ist jedoch in der Lage seine Funktionalität für beide IP-Protokolle zur Verfügung zu stellen. Das Betriebssystem bildet die einegehenden IPv4-Adressen auf IPv6-Adressen ab. Die Netzwerkkommunkation bleibt davon unberrührt, somit erfolgt die Kommunikation zwischen den Hosts weiter über IPv4 [MM05, S. 241f.]. IPv4-only Host sind somit in der Lage Services zu nutzen, die ausschließlich über IPv6-Sockets realisiert sind. Dieser Ansatz fördert zudem die Verringerung der Nutzung von IPv4-Sockets, bei der Programmierung von Dual-Stack-Services.

- Erweiterungen für TCP-Services

Auf Grund der Eigenschaft, dass ein Dual-Stack-Service über beide IP-Protokolle genutzt werden kann, ist es erforderlich sinnvolle Erweiterungen zu spezifizieren. Vor allem bei TCP-Services, welche eine verbindungsorientierte Kommunikation aufbauen, ist es erforderlich, dass Anwendungen über eine Implementierung des Happy Eyeball-Mechanismus verfügen, um Antwortzeiten zu verringern. Die Testergebnisse zeigen, dass eine Vielzahl von Clientimplmentierungen über keinen Happy Eyeball-Mechanismus verfügen. Dies hat zur Folge, dass der Kommunikationsaufbau teilweise um über 30 Sekunden verzögert wird oder der Verbindungsaufbau vollständig scheitert. Ein so schwerwiegendes Fehlverhalten ist in der heutigen Zeit nicht vertretbar, deshalb sollten TCP-Services ohne Happy Eyeball-Implementierung nicht verwendet werden.

- Erweiterungen für UDP-Services

Wie bei einer verbindungsorientierten Kommunikation über TCP, kann es auch zu Verzögerungen kommen bei verbindunglosen Kommunikationen über UDP. Der RFC 6555 spezifiziert den Happy Eyeball-Meachnismus ausschließlich für verbindungsorientierte Transportprotokolle wie TCP und SCTP. Erwähnt wird jedoch das ein ähnlicher Mechanismus für verbindungslose Transportprotokolle wie UDP ebenfalls sinnvoll ist, sofern die Anwendung eine Anfrage-Antwort-Semantik nutzt [WY12]. So ist es möglich, dass beispielsweise der UDP-Service DNS von einer hohen Antwortzeit betroffen ist (s. Abbildung 3.87). Dieses Fehlverhalten würde sich bei der darüber liegenden Anwendung, welche DNS zur Namensauflösung nutzt, ebenfalls bemerkbar machen. Eine Fragestellung die daraus resultiert ist, ob es sinnvoll ist eine geeignete Happy Eyeball-Implementierung für TCP- und UDP-Services auf Betriebssystemebene zu realisieren, die ihren Dienst allen Anwendungen zur Verfügung stellt.

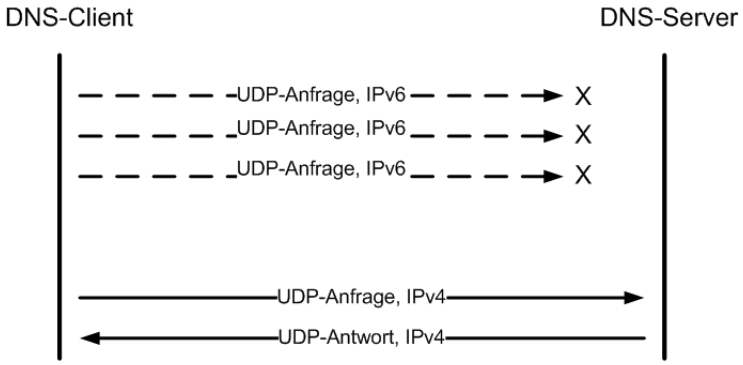


Abbildung 3.87: Fehlverhalten von UDP-Services

4 Leitfaden zur Servicemigration

Durch die Migration der in Kapitel 3 vorgestellten Services wird in diesem Kapitel auf Grundlage der daraus gewonnenen Erkenntnisse, ein Leitfaden zur Servicemigration in den Dual-Stack-Betrieb vorgestellt. Dieser Leitfaden enthält eine Abfolge von Aufgaben, die für die Servicemigration notwendig sind. Sinnvollerweise sollte die Servicemigration nicht überhastet und in einem zu engen Zeitfenster erfolgen. Dies könnte zu Ausfällen von bereits vorhandenen Services, oder zu Beeinträchtigungen des gesamten Netzes führen. Eine sorgfältige Planung ist deshalb unabdingbar [Hag09, S. 439]. Des Weiteren sollte ein Service nach seiner Migration ausgiebig getestet werden, bevor der nächste Service migriert wird. Dies hat den Vorteil, dass die Fehlersuche somit besser eingegrenzt wird und die verantwortlichen Administratoren Zeit haben, sich mit dem neuen IPv6-Protokoll auseinanderzusetzen. Im Folgenden werden die notwendigen Schritte zur Servicemigration erläutert.

1. Aufbau einer Testumgebung

Vor der Servicemigration in einer Liveumgebung, sollte eine Testumgebung aufgebaut werden, anhand derer die weiteren Schritte getestet werden können. Diese Vorgehensweise löst auftretende Probleme bereits von vornherein in der Testumgebung, sodass die Liveumgebung des Unternehmens nicht betroffen ist.

2. Auswahl des Services

Im ersten Schritt muss der Service festgelegt werden, welcher in eine Dual-Stack-Umgebung migriert werden soll. Sind noch keine Services migriert worden, so sollte zunächst der Kernservice DHCP migriert werden, da ohne eine gültige IPv6-Netzkonfiguration, ein Host keinen Service über IPv6 nutzen kann. Als zweiter Service sollte der Kernservice DNS migriert werden. Alle weiteren Services können in beliebiger Reihenfolge migriert werden.

3. Spezifikation der Dual-Stack-Erweiterungen

Wurde zuvor ein Service ausgewählt, sollte anaylsiert werden, welche Erweiterungen für den Dual-Stack-Betrieb spezifiziert sind. Der RFC 3790 - *"Survey of IPv4 Addresses in Currently Deployed IETF Internet Area Standards Track and Experimental Documents"* [MN04] und RFC 3795 - *"Survey of IPv4 Addresses in Current-*

ly Deployed IETF Application Area Standards Track and Experimental Documents" [SN04] enthalten eine Liste von Netzwerkprotokollen, die auf ihre IPv4-Abhängigkeit analysiert wurden. Dort wird auf alle weiteren RFC's referenziert, die für den Dual-Stack-Betrieb des Services erforderlich sind. Abhängig davon kann eine geeignet Implementierung ausgewählt werden.

4. Auswahl der Serverimplementierung

Im nächsten Schritt wird eine geeignete Serverimplementierung für den Dual-Stack-Betrieb ausgewählt. Da die meisten Unternehmen bereits eine Implementierung des Services für IPv4 im Betrieb haben, sollte als erstes spezifiziert werden um welche Implementierung es sich handelt. Viele Implementierungen verfügen über ein sogenanntes *Changelog* auf ihrer Homepage. Dort lässt sich verifizieren, welche Funktionalitäten oder RFC's in den verschiedenen Versionen implementiert wurden und ab welcher Version IPv6 unterstützt wird. Sind keine Angaben vorhanden, so muss die Dual-Stack-Fähigkeit der Implementierung durch einen Test bestätigt werden (Schritt 6). Wurde der Service für IPv6 neu konzipiert, muss eventuell eine neue Implementierung ausgewählt werden. Die Implementierung, die den Service bereits zuvor für IPv4 bereitstellt, kann dann weiter verwendet werden. Ist der Service nicht neu konzipiert wurden und die vorhandene Implementierung unterstützt die notwendigen Funktionalitäten für den Dual-Stack-Betrieb nicht, so muss eine neue Implementierung eingesetzt werden.

5. Installation und Konfiguration der Serverimplementierung

Ist eine geeignete Implementierung ausgewählt worden oder bereits vorhanden, so muss diese für den Dual-Stack-Betrieb konfiguriert werden. Eine neue Implementierung ist zunächst zu installieren. In den meisten Fällen sind bei der Installation keine speziellen Angaben für IPv6 erforderlich, da die Entwickler die IPv6-Funktionalitäten heutzutage standardmäßig kompilieren. Für die Konfiguration der Implementierung im Dual-Stack-Betrieb ist es sinnvoll, sich die Dokumentation des Entwicklers oder bei Linux-Implementierungen zudem die man-page anzuschauen. Alternativ sind Fachzeitschriften wie beispielsweise die iX zu empfehlen, welche regelmäßig Fachartikel zu dieser Thematik veröffentlichen. Eine Vielzahl von Büchern mit Konfigurationsbeschreibungen zu den Services, stellte sich beim derzeitigen Stand als ungeeignet dar, da die Konfiguration für den Dual-Stack-Betrieb nicht thematisiert wurde. Bei Linux-Implementierungen sind für die Konfiguration Änderungen in einer entsprechenden Konfigurationsdatei vorzunehmen. Diese Änderungen beschränken sich in der Regel auf bestimmte Parameterzuweisungen, Angabe der zu bindenen IP-Adressen und Ports. Unter Windows Serversystemen erfolgt die Konfiguration über den integrierten Server-Manager. Als Hilfestellung dienen dazu die in

dieser Arbeit migrierten Services in Kapitel 3.

6. Testen der Serverimplementierung

Unabhängig von den Angaben des Entwicklers, sollte die Serverimplementierung auf ihre Serviceverfügbarkeit und -funktionalität getestet werden. Ob ein Service über die zuvor konfigurierten Socket-Adressen auf Anfragen lauscht, kann mit Hilfe des Tools *netstat* geprüft werden. Netstat zeigt alle lauschenden und hergestellten Verbindungen für UDP und TCP über beide IP-Protokolle an. Dadurch kann festgestellt werden, ob der Service im Netzwerk zur Vefügung steht. Für die Überprüfung der Funktionalität eines Services, sind abhängig vom Service, verschiedene Tools erforderlich. Nachstehend in Tabelle 4.1, eine Übersicht einiger Tools.

Service	Tools
DHCP	dhclient, rdisc6
DNS	dig, nslookup
HTTP	lynx
FTP	ftp
IMAP	telnet, openSSL
SNMP	snmpwalk, snmpget
SMTP, POP3	telnet

Tabelle 4.1: Tools zum testen von Services

7. Auswahl der Clientimplementierungen

Nach erfolgreicher Inbetriebnahme der Serverimplementierung wird festgestellt, welche Betriebssysteme im Unternehmen verwendet werden. Empfehlenswert ist es, Implementierungen zu verwenden, die im Unternehmen bereits für IPv4 eingesetzt werden und für eine große Anzahl von Betriebssystem zur Verfügung stehen. Wie bei den Serverimplementierungen verfügen die Clientimplementierungen in der Regel auch über ein *Changelog* auf der Homepage des Entwicklers. Dort lässt sich verifizieren, ob IPv6 unterstützt wird und ab welcher Version. Des Weiteren ist es sinnvoll darauf zu achten, ob die Implementierungen sinnvolle Erweiterungen enthalten, wie den Happy Eyeball-Mechanismus bei TCP-Services. Sind darüber keine Informationen verfügbar, so kann das anhand eines Tests ermittelt werden (Schritt 9).

8. Installation und Konfiguration der Clientimplementierungen

Abhängig vom Betriebssystem erfolgt die Installation der Clientimplementierung

über ein Software-Center (Android, iOS, Ubuntu) oder über eine ausführbare Datei (Mac OS X, Windows). In der Regel sind während der Installation keine Optionen bezüglich des Dual-Stack-Betriebes einzustellen. Auch eine Konfiguration sollte nicht erforderlich sein. Die Nutzung von IPv4 oder IPv6 sollte durch sinvolle Implementierungen, wie der Default Address Selection und des Happy Eyeball-Mechanismuses, automatisch in der Anwendung erfolgen.

9. Testen der Clientimplementierung

Zur Überprüfung der Funktionalität einer Clientimplementierung, wird die Anwendung an sich verwendet. Dabei wird versucht, über diese, den Service über IPv4 und IPv6 zu nutzen. Ein konkretes Beispiel dafür wäre der Zugriff vom Firefox-Webbrowser auf einen Webserver. Wird der Zugriff über IPv6 getestet, so wird die IPv6-Adresse in der Regel nach [HCM99] in eckigen Klammern geschrieben. Dies verhindert die Fehlinterpretation einer IPv6-Adresse, aufgrund des Syntax Problems (s. Kapitel 2.3). Zur Überprüfung, welche IP-Version bevorzugt genutzt wird oder zur Überprüfung von Erweiterungen, wie dem Happy Eyeball-Mechanismus, ist es notwendig den Netzwerkverkehr zu analysieren (s. Kapitel 3.6). Zum Mitschneiden des Netzwerkverkehrs eignen sich die Tools *tcpdump* und *wireshark*. Im RFC 6556 - *"Testing Eyeball Happiness"* werden dafür Methoden spezifiziert, um eine Anwendung im Black-Box-Verfahren auf eine Happy Eyeball-Implementierung zu testen [Bak12]. Verfügt die Anwendung über keine Implementierung des Happy Eyeball-Mechanismus, so sollte diese Anwendung nicht verwendet werden. Fällt das Testergebnis negativ aus, so wird eine den Anforderungen besser entsprechende Clientimplementierung ausgewählt (Schritt 7).

10. Sicherheitsspezifische Aspekte

Sind IPv4-Services in eine Dual-Stack-Umgebung migiriert worden, sollte darauf geachtet werden, dass die Einstellungen der Firewall aktualisiert werden. Dabei ist zu beachten, das aufgrund der neuen Protokollfamilie *AF_INET6* neue Portbereiche verwendet werden. Neu konzipierte Services, wie beispielsweise DHCPv6 verwenden zudem neue Portnummern. Im Gegensatz zu DHCPv4, welches die Ports 67/68 verwendet, nutzt DHCPv6 die Ports 546/547. Des Weiteren empfiehlt sich die Evaluierung eines geeigneten Intrusion Detection Systems (IDS). Durch die Migration von Services in eine IPv4-/IPv6-Dual-Stack-Umgebung sind neue Angriffe auf das Netzwerk möglich, gegen die eine Absicherung stattfinden sollte. Zurzeit erkennen eine Vielzahl von Intrusion Detection Systemen IPv6-spezifische Angriffsmuster [Mar13, S. 48]. Ein guter Artikel, der sich mit der Sicherheit für IPv6 in Dual-Stack-Netzen beschäftigt ist *"Umgestellt - IPv6 sicher ausrollen in kleinen und großen Organisationen"* von Stefan Marksteiner [Mar13].

11. Netzmanagementspezifische Aspekte

Durch Migration von einer reinen IPv4-Umgebung zu einer IPv4-/IPv6-Dual-Stack-Umgebung und die damit verbundenen Neuerungen, erhöht sich die Komplexität des gesamten Netzes. Dies hat starke Auswirkungen auf das Netzmanagement. Eine Vielzahl von Netzmanagementtools unterstützt IPv6 nur eingeschränkt. Somit kann beispielsweise pro Host nur eine IPv6-Adresse eingetragen werden, obwohl gerade bei IPv6 mehrere Adressen pro Interface üblich sind. Deshalb werden Hosts doppelt angelegt und führen zu einer steigenden Unübersichtlichkeit. Zudem müssen für IPv6 Alarme spezifiziert werden, die auch für IPv4 notwendig sind. Dadurch wird nicht mehr lediglich unterschieden, ob ein Service verfügbar ist, sondern es wird festgestellt ob der Service über IPv4, IPv6 oder beide IP-Protokolle erreichbar ist. Im Rahmen einer Masterarbeit mit dem Titel *"Netzmanagement von IPv4/v6-Dual-Stack-Umgebungen - Konzeption, Evaluierung und Implementierung"* von Markus Becker [Bec12], wird das Netzmanagement einer Dual-Stack-Umgebung grundlegend behandelt. Wird in einem Unternehmen zudem eine Configuration Managment Database (CMDB) gepflegt, ist es notwendig alle IPv6-Services, -Adressen, -Netze etc. der CMDB hinzuzufügen.

12. Betriebssystemspezifische Aspekte

Aufgrund der funktionellen Einschränkungen der Betriebssysteme Android und Windows XP im Dual-Stack-Betrieb, ist es ratsam diese nicht in einer Dual-Stack-Umgebung einzusetzen. Indem Android stetig weiter entwickelt wird, ist anzunehmen dass in naher Zukunft Verbesserungen im Bezug auf IPv6 zu erwarten sind. Dies ist zudem jedoch abhängig von den Endgeräteherstellern, welche das Android-Betriebssystem in der Regel ihren Wünschen anpassen. Windows XP hingegen wird nicht weiter entwickelt [Zei13] und verursacht einen enormen administrativen Aufwand bei der Inbetriebnahme in einer Dual-Stack-Umgebung. Zudem ist zu erwarten, dass zukünftige Anwendungen, die weitere IPv6-Funktionalitäten bereitsstellen, nicht unter Windows XP verfügbar sein werden. Es ist empfehlenswert bei einer Migration von einer reinen IPv4-Umgebung zu einer IPv4-/IPv6-Dual-Stack-Umgebung, Windows XP durch neuere Betriebssysteme wie Windows 7 oder 8 zu ersetzen.

5 Zusammenfassung und Ausblick

Ziel der Arbeit war es einen Leitfaden zur Servicemigration von IPv4 zu IPv4-/IPv6-Dual-Stack-Umgebungen zu entwerfen. Dieser soll Administratoren, die in ihrem Unternehmen eine Dual-Stack-Umgebung einführen wollen, als Hilfestellung dienen. Dazu wurden zunächst vier exemplarische Services ausgewählt und die jeweiligen Erweiterungen für IPv6 recherchiert. Auf Grundlage der recherchierten RFC's wurden geeignete server- und clientseitige Implementierungen ausgewählt. Diese Implementierungen wurden in eine Testumgebung migiriert und auf ihre Serviceverfügbarkeit und -funktionalität im Dual-Stack-Betrieb getestet.
Die migrierten Services haben gezeigt, dass serverseitig alle getesteten Implementierungen ihre Funktionalität im Dual-Stack-Betrieb erfüllen. Clientseitig zeigte sich jedoch, dass besonders bei FTP Verbesserungen notwendig sind. Nur einer der FTP-Clients, verfügte über eine Implementierung des Happy Eyeball-Mechanismus. Alle weiteren FTP-Clients zeigten enorme Verzögerungen bei der Antwortzeit oder brachen den Verbindungsversuch ab. Bei HTTP zeigte sich lediglich einer der Webbrowser, aufgrund einer fehlenden Happy Eyeball-Implementierung, als ungeeignet. DNS-seitig waren alle Betriebssysteme in der Lage DNS-Anfragen für IPv6-Adressen zu senden. Abhängig vom Betiebssystem wurden die Anfragen vorrangig über IPv4, IPv6 oder parallel über beide IP-Protokolle gestellt. Android hingegen unterstützt ausschließlich IPv4 für die Anfrage von AAAA-Records. Im Hinblick auf die Autokonfiguration konnten alle Betriebssysteme bis auf Android, über Router-Advertisements und DHCPv6 konfiguriert werden. Eine Autokonfiguration für Android ist aufgrund des fehlenden DHCPv6-Clients zur Zeit nur über Router-Advertisements möglich. Da unter Windows XP statische Eintragungen und Installationen erforderlich sind, ist ein vergleichsweise hoher administrativer Aufwand entsteht.
Die Migration zeigte, dass die Services in unterschiedlicher Weise abhängig von IPv4 sind. DHCP wurde aufgrund der starken Abhängigkeit neu spezifiziert. Für DNS und FTP waren Erweiterungen für IPv6 erforderlich. So wurde für FTP der Befehlssatz erweitert und für DNS ein neuer Ressource Record, sowie eine neue Reverse-Lookupdomain eingeführt. HTTP hingegen ist IPv4-unabhängig und benötigt keine Anpassungen für IPv6. Unabhängig vom Service zeigte sich, dass weitere optionale Erweiterungen sinnvoll sind. So spielt der für TCP-Services spezifizierte Happy Eyeball-Mechanismus bei Hosts im Dual-Stack-Betrieb, eine wichtige Rolle. Ohne eine Implementierung des Mechanismus kommt es zu hohen Antwortzeiten, beim Ausfall der IPv6-Konnektivität. Ein ähnlicher Mechanismus wäre auch für UDP-Services mit einer Anfrage-Antwort-Semantik sinnvoll. Zur Zeit liegt jedoch keine Spezifika-

tion vor. Empfehlenswert, wäre es einen entsprechenden Mechanismus für TCP- und UDP-Services auf Betriebssystemebene bereitzustellen. Des Weiteren wurde beobachtet, dass die meisten Implementierungen Probleme mit der Interpretation von IPv6-Adressen zeigten. Deshalb ist es erforderlich explizite IPv6-Adressangaben in eckigen Klammern einzugeben. Eine weitere interessante Beobachtung zeigte, das einige serverseitige Implementierungen ihren Service ausschließlich über IPv6-Sockets zur Verfügung stellen. Dies ist durch die Verwendung von IPv4-mapped IPv6-Adressen möglich.
Aus den daraus gewonnen Erfahrungen entstand der Leitfaden zur Servicemigration. Abschließend hat die Arbeit gezeigt, dass weitere Fragestellungen behandelt werden sollten. Zur Optimierung des Leitfadens, könnten weitere Services wie beispielsweise NTP, SMTP, SNMP etc. migriert werden. Aus den daraus gewonnen Erkenntnissen könnten möglicherweise neue Aspekte diskutiert werden. Auch die Evaluierung der unterschiedlichen Happy Eyeball-Implementierungen, im Hinblick auf die Performance und weitere Parameter, könnten eine Referenzimplementierung für den Happy Eyeball-Mechanismus definieren. Zudem ist es sinvoll zu analysieren, welche Art von Happy Eyeball-Implementierung am besten für UDP-Services geeignet ist. Abschließend könnten die verschiedenen Services im Hinblick auf ihre Funktionalitäten untersucht werden, die auschließlich für IPv6 spezifiziert wurden.

Literaturverzeichnis

[Ago07] Agouros, Konstantinos: *DNS/DHCP - Grundlagen und Praxis*. 1. Aufl. München : Open Source Press, 2007. – ISBN 978–3–937514–35–2

[ALL06] Albitz, Paul ; Liu, Cricket ; Loukides, Mike: *DNS and BIND*. 5. rev. ed. Sebastopol, CA : O'Reilly Media, Inc., 2006. – ISBN 978–0–596–10057–5

[And13] *FTP, FTPS, SCP and SFTP Android client - AndFTP*. `http://www.lysesoft.com/products/andftp/index.html`. Version: Januar 2013

[AO98] Allman, M. ; Ostermann, S.: *FTP Extensions for IPv6 and NATs*. RFC 2428 (Proposed Standard). `http://tools.ietf.org/html/rfc2428`. Version: September 1998 (Request for Comments)

[Bak12] Baker, F.: *Testing Eyeball Happiness*. RFC 6556 (Informational). `http://tools.ietf.org/html/rfc6556`. Version: April 2012 (Request for Comments)

[Bec12] Becker, Markus: Netzmanagement von IPv4/v6-Dual-Stack-Umgebungen - Konzeption, Evaluierung und Implementierung. (2012)

[Dav12] Davies, Joseph: *Understanding IPv6*. Sebastopol, CA : O'Reilly Media, Inc., 2012. – ISBN 978–0–7356–5914–8

[DH98] Deering, S. ; Hinden, R.: *Internet Protocol, Version 6 (IPv6) Specification*. RFC 2460 (Draft Standard). `http://tools.ietf.org/html/rfc2460`. Version: Dezember 1998 (Request for Comments)

[DH06] Deering, S. ; Hinden, R.: *IP Version 6 Addressing Architecture*. RFC 4291 (Draft Standard). `http://tools.ietf.org/html/rfc4291`. Version: Februar 2006 (Request for Comments)

[DKK+12] Deimeke, Dirk ; Kania, Stefan ; Kühnast, Charly ; Semmelroggen, Stefan ; Soest, Daniel v.: *Linux-Server - Das Administrationshandbuch*. 2. Aufl. Bonn : Galileo Press GmbH, 2012. – ISBN 978–3–836–21879–5

[Dro97] Droms, R.: *Dynamic Host Configuration Protocol.* RFC 2131 (Draft Standard). http://tools.ietf.org/html/rfc2131. Version: März 1997 (Request for Comments)

[EK09] Endres, Johannes ; Kaps, Reiko: Ende der Enge - Das Internet-Protokoll Version 6 löst Probleme und schürt Ängste. In: *c't - Magazin für Computertechnik Nr. 14* (2009)

[FGM+99] Fielding, R. ; Gettys, J. ; Mogul, J. ; Frystyk, H. ; Masinter, L. ; Leach, P. ; Berners-Lee, T.: *Hypertext Transfer Protocol – HTTP/1.1.* RFC 2616 (Draft Standard). http://tools.ietf.org/html/rfc2616. Version: Juni 1999 (Request for Comments)

[Fil13a] *Documentation - Filezilla Wiki.* http://wiki.filezilla-project.org/Documentation. Version: Januar 2013

[Fil13b] *Filezilla - The free FTP solution.* http://filezilla-project.org/. Version: Januar 2013

[Fra13] Frank, Bernhard: *IIS für Einsteiger Teil 5: FTP.* http://blogs.technet.com/b/bernhard_frank/archive/2011/05/27/iis-f-252-r-einsteiger-teil-5-ftp-oder-wie-lade_ich-dateien-auf-meinen-webserver.aspx. Version: Januar 2013

[GTBS99] Gilligan, R. ; Thomson, S. ; Bound, J. ; Stevens, W.: *Basic Socket Interface Extensions for IPv6.* RFC 2553 (Informational). http://tools.ietf.org/html/rfc2553. Version: März 1999 (Request for Comments)

[Hag06] Hagen, Silvia: *IPv6 Essentials.* 2nd ed. Sebastopol, CA : O'Reilly Media, Inc., 2006. – ISBN 978–0–596–10058–2

[Hag09] Hagen, Silvia: *IPv6 - Grundlagen - Funktionalität - Integration.* 2. Auflage. Norderstedt : BoD – Books on Demand, 2009. – ISBN 978–3–952–29422–2

[Hag11] Hagen, Silvia: *Planning for IPv6.* 1. Aufl. Sebastopol, CA : O'Reilly Media, Inc., 2011. – ISBN 978–1–449–30539–0

[HCM99] Hinden, R. ; Carpenter, B. ; Masinter, L.: *Format for Literal IPv6 Addresses in URL's.* RFC 2732 (Proposed Standard). http://tools.ietf.org/html/rfc2732. Version: Dezember 1999 (Request for Comments)

[Hei13] *heise Software-Verzeichnis*. `http://www.heise.de/download/`. Version: Januar 2013

[JPBM10] Jeong, J. ; Park, S. ; Beloeil, L. ; Madanapalli, S.: *IPv6 Router Advertisement Options for DNS Configuration*. RFC 6106 (Proposed Standard). `http://tools.ietf.org/html/rfc6106`. Version: November 2010 (Request for Comments)

[Liu11] Liu, Cricket: *DNS und BIND im IPv6*. Sebastopol, CA : O'Reilly Media, Inc., 2011. – ISBN 978–3–86899–180–2

[LR03] Levi, Paul ; Rembold, Ulrich: *Einführung in die Informatik für Naturwissenschaftler und Ingenieure*. 4. aktualisierte Auflage. München, Wien : Hanser Verlag, 2003. – ISBN 3–446–21932–2

[Mar13] Marksteiner, Stefan: Umgestellt - IPv6 sicher ausrollen in kleinen und großen Organisatonen. In: *iX - Magazin für Professionelle Informationstechnik Nr. 3* (2013)

[Mic13] *Hostingfreundliche Webserverplattform (IIS): Szenarioübersicht*. `http://technet.microsoft.com/de-de/library/hh831818.aspx`. Version: 2013

[ML05] Mun, Youngsong ; Lee, Hyewon K.: *Understanding IPv6*. 1. Aufl. Berlin, Heidelberg : Springer, 2005. – ISBN 978–0387–25429–6

[MM05] Murphy, Niall R. ; Malone, David: *IPv6 Network Administration*. 1. Aufl. Sebastopol, CA : O'Reilly Media, Inc., 2005. – ISBN 978–0–596–00934–2

[MN04] Mickles, C. ; Nesser, P.: *Survey of IPv4 Addresses in Currently Deployed IETF Internet Area Standards Track and Experimental Documents*. RFC 3790 (Informational). `http://tools.ietf.org/html/rfc3790`. Version: Juni 2004 (Request for Comments)

[Moc87a] Mockapetris, P.: *DOMAIN NAMES - CONCEPTS AND FACILITIES*. RFC 1034 (Internet Standard). `http://tools.ietf.org/html/rfc1034`. Version: November 1987 (Request for Comments)

[Moc87b] Mockapetris, P.: *DOMAIN NAMES - IMPLEMENTATION AND SPECIFICATION*. RFC 1035 (Internet Standard). `http://tools.ietf.org/html/rfc1035`. Version: November 1987 (Request for Comments)

[Olb03] Olbrich, Alfred: *Netze - Protokolle - Spezifikationen - Die Grundlagen Für Die Erfolgreiche Praxis*. Aschaffenburg : Vieweg, 2003. – ISBN 3–528–08546–3

[PD03] Peterson, Larry L. ; Davie, Bruce S.: *Computernetze*. 04003. Aufl. Heidelberg : Dpunkt-Verlag, 2003. – ISBN 3–89864–242–9

[PR85] Postel, J. ; Reynolds, J.: *FILE TRANSFER PROTOCOL (FTP)*. RFC 959 (Standard). http://tools.ietf.org/html/rfc959. Version: Oktober 1985 (Request for Comments)

[Pro13a] *The ProFTPD-Project: Example Configurations*. http://www.proftpd.org/docs/example-conf.html. Version: Januar 2013

[Pro13b] *The ProFTPD-Project: FTP Related RFCs (Request For Comments)*. http://www.proftpd.org/docs/rfc.html. Version: Januar 2013

[RDBV+03] R. Droms, Ed. ; Bound, J. ; Volz, B. ; Lemon, T. ; Perkins, C. ; Carney, M.: *Dynamic Host Configuration Protocol for IPv6 (DHCPv6)*. RFC 3315 (Proposed Standard). http://tools.ietf.org/html/rfc3315. Version: Juli 2003 (Request for Comments)

[SKL12] Schneider, Christian ; Kannen, Martina ; Leischner, Martin: IPv6-Autokonfiguration von Clients. In: *iX - Magazin für Professionelle Informationstechnik Nr. 11* (2012)

[SN04] Sofia, R. ; Nesser, P.: *Survey of IPv4 Addresses in Currently Deployed IETF Application Area Standards Track and Experimental Documents*. RFC 3795 (Informational). http://tools.ietf.org/html/rfc3795. Version: Juni 2004 (Request for Comments)

[ST98] Stevens, W. ; Thomas, M.: *Advanced Sockets API for IPv6*. RFC 2292 (Informational). http://tools.ietf.org/html/rfc2292. Version: Februar 1998 (Request for Comments)

[Sta13] Statista: *Android - Anteil der Versionen Ende Januar 2013*. http://de.statista.com/statistik/daten/studie/180113/umfrage/anteil-der-verschiedenen_android-versionen-auf-geraeten-mit_android-os/. Version: Januar 2013

[Sto07] Stockebrand, Benedikt: *IPv6 in Practice - A Unixer's Guide to the Next Generation Internet*. Berlin, Heidelberg : Springer, 2007. – ISBN 978–3–540–24524–7

[THKS03] Thomson, S. ; Huitema, C. ; Ksinant, V. ; Souissi, M.: *DNS Extensions to Support IP Version 6*. RFC 3596 (Draft Standard). http://tools.ietf.org/html/rfc1035. Version: Oktober 2003 (Request for Comments)

[TNJ07] Thomson, S. ; Narten, T. ; Jinmei, T.: *IPv6 Stateless Address Autoconfiguration*. RFC 4862 (Draft Standard). `http://tools.ietf.org/html/rfc4862`. Version: September 2007 (Request for Comments)

[TW11] Tanenbaum, Andrew S. ; Wetherall, David J.: *Computer Networks*. 5. aktualisierte Auflage. Amsterdam (Niederlande), Seattle (USA) : Pearson Studium, 2011. – ISBN 978–0–13–255317–9

[TW12] Tanenbaum, Andrew S. ; Wetherall, David J.: *Computernetzwerke*. 5. aktualisierte Auflage. München : Pearson Studium, 2012. – ISBN 978–3–868–94137–1

[Ubu13] *ISC-DHCPD - Wiki*. `http://wiki.ubuntuusers.de/ISC-DHCPD`. Version: Februar 2013

[WY12] Wing, D. ; Yourtchenko, A.: *Happy Eyeballs: Success with Dual-Stack Hosts*. RFC 6555 (Proposed Standard). `http://tools.ietf.org/html/rfc6555`. Version: April 2012 (Request for Comments)

[Zei13] *Microsoft: Zeitbombe Windows XP*. `http://www.zeit.de/digital/internet/2013-02/windows-xp-sicherheitsrisiko`. Version: Februar 2013