Technische Universität München
Lehrstuhl für Kommunikationsnetze

Eine Internet-Architektur mit Trennung von Locator und Identifier

Christoph Ludwig Spleiß

Vollständiger Abdruck der von der Fakultät Elektrotechnik und Informationstechnik der Technischen Universität München zur Erlangung des akademischen Grades eines

Doktor-Ingenieurs (Dr.-Ing.)

genehmigten Dissertation.

Vorsitzender:		Univ.-Prof. Dr.-Ing. Hans-Georg Herzog
Prüfer der Dissertation:	1.	Univ.-Prof. Dr.-Ing. Jörg Eberspächer
	2.	Univ.-Prof. Dr. rer. nat. Paul Müller, Technische Universität Kaiserslautern

Die Dissertation wurde am 19.01.2012 bei der Technischen Universität München eingereicht und durch die Fakultät für Elektrotechnik und Informationstechnik am 17.05.2012 angenommen.

Eine Internet-Architektur mit Trennung von Locator und Identifier

Christoph Ludwig Spleiß

26. Juli 2012

Kurzfassung

In dieser Arbeit wird eine revolutionäre Architektur für das Internet der Zukunft vorgestellt. Sie basiert auf der Trennung von Locator und Identifier und ist damit in der Lage, auftretende Probleme wie die Größe der Routingtabellen sowie die mangelnde Unterstützung für mobile Endgeräte zu lösen. Neu gegenüber existierenden Konzepten sind ein zweiteiliger Locator, der Unterstützung für heterogene Netze und Endgeräte bietet sowie ein lebenslang gültiger Identifier für alle Geräte. Durch eine Erweiterung zur Objektunterstützung ist es möglich, neben Geräten auch Inhaltsobjekte mit eindeutigem Identifier anzusprechen. Zusätzlich wird ein Mechanismus entworfen, der Inhaltsobjekte abhängig von Popularität und Aufrufverteilung ressourceneffizient im Netz verteilt und Kosten für die Bereitstellung minimiert. Varianten zur Migration vom heutigen Internet zum Internet der Zukunft werden aufgezeigt. Die neuen Konzepte wurden teilweise im Experimentalnetz „German Lab“ implementiert.

Abstract

This thesis presents a new “clean slate” architecture for the future Internet, which is based on the paradigm of locator/identifier separation. This paradigm can cope with arising problems like routing table growth or lack of mobility support for devices. Novelties compared to existing approaches are a two-tier locator, which enables support for heterogeneous networks and devices as well as a unique identifier for devices with lifetime validity. This architecture is further extended to support content, which allows for addressing content objects with distinct identifiers. Additionally, an algorithm is developed that distributes content objects in the Internet according to popularity and access distribution. This is done in a resource efficient way, thus minimizing provisioning costs. Furthermore, migration strategies from the current Internet to the future Internet are developed and discussed. Parts of this concept have been implemented in the “German Lab” experimental facility.

There is always a way to do it better. Find it!

— Thomas A. Edison

Vorwort

Diese Arbeit ist das Ergebnis meiner Forschungstätigkeiten als wissenschaftlicher Mitarbeiter am Lehrstuhl für Kommunikationsnetze der Technischen Universität München.

Mein besonderer Dank gilt meinem Betreuer und Doktorvater, Herrn Prof. Dr.-Ing. Jörg Eberspächer. Er gab mir in vielen wertvollen Diskussionen und Gesprächen die richtigen Anregungen zur rechten Zeit und hat mit seinem Engagement diese Arbeit ermöglicht. Durch vielseitige Tätigkeiten, sowohl in Industrieprojekten als auch im Bereich der Lehre, konnte ich an seinem Lehrstuhl viele wertvolle Erfahrungen zu sammeln.

Die Grundlagen für meine Forschungen basieren zu einem großen Teil auf den gewonnenen Ergebnissen des vom Bundesministerium für Bildung und Forschung geförderten Projekts German-Lab (G-Lab). Mein Dank gilt dabei besonders meinem Kollegen und Projektpartner Oliver Hanka, durch dessen konstruktive Zusammenarbeit mit mir über die Jahre hinweg viele Grundsteine dieser Arbeit gelegt wurden. Des Weiteren gilt mein Dank den Kollegen der Projektpartner der Universität Würzburg, des Karlsruher Institute of Technology sowie der Technischen Universität Kaiserslautern für die gute Kooperation, den vielen Telefonkonferenzen und Treffen, bei denen auch die soziale Komponente definitiv nicht vernachlässigt wurde. In diesem Zusammenhang danke ich besonders Herrn Prof. Dr. rer. nat. Paul Müller aus Kaiserslautern für die Bereitschaft, das Zweitgutachten dieser Arbeit zu übernehmen.

Des Weiteren bedanke ich mich bei meinen ehemaligen Arbeitskollegen Dr. Michael Eichhorn, Dr. Moritz Kiese, Dr. Ulrike Korger, Dr. Gerald Kunzmann, Bernhard Lichtinger, Silke Meister, Christian Merkle, Dr. Robert Nagel, Martin Pfannenstein, Dr. Robert Prinz und Dr. Stefan Zöls für die schöne Zeit in einer angenehmen Arbeitsatmosphäre, in der auch viele Freundschaften entstanden sind. Ohne deren Unterstützung wäre diese Arbeit nicht möglich gewesen.

Dr. Martin Maier danke ich für die jederzeit schnelle und absolut unkomplizierte Lösung jedes erdenklichen administrativen Problems, auch wenn dieses erst im weiteren Sinne etwas mit Forschung oder Lehre zu tun hatte. Die vielen Stunden in seinem Büro beim Lösen unzähliger technischer Probleme, sein Geschick mit hunderten geöffneten Fenstern zu hantieren, sowie seine jederzeit lockeren Sprüche werde ich nicht vergessen.

Von meinen Studenten möchte ich mich besonders bei Clemens Schuwerk und Quirin Scheitle bedanken, die einen großen Beitrag zum Gelingen dieser Arbeit beigetragen haben.

Nicht zuletzt danke ich ganz besonders meinen Eltern Monika und Ludwig Spleiß, die mir das Studium an der TU München ermöglicht und mich auf meinem Weg jederzeit unterstützt haben. Ebenso danke ich all meinen Freunden für ihre vielseitige Unterstützung, Verständnis und Geduld.

München, Juli 2012 Christoph Spleiß

Inhaltsverzeichnis

1. Einführung

Das Internet ist aus der heutigen Gesellschaft nicht mehr wegzudenken. Es hat die Art und Weise verändert, wie Personen miteinander kommunizieren, Informationen beschaffen und diese austauschen. Das Internet ist das Rückgrat der modernen, globalisierten Wirtschaft, die ihre Geschäftsprozesse darüber abwickelt und esbildet die Existenzgrundlage für viele Geschäftszweige, die Hardware- und Softwareprodukte herstellen.

Die heutige Internet-Architektur entstand vor mehr als 40 Jahren im Zuge eines Forschungsprojekts. Dessen Ziel war die Verbindung einiger weniger Forschungsstandorte, um damals teure Rechenkapazitäten miteinander zu teilen. Den damals festgelegten Planungszielen, die eine einfache und leicht erweiterbare Architektur vorsahen, verdankt das Internet seinen heutigen Erfolg. Aus ursprünglich einfachen Diensten zur Dateiübertragung mittels FTP oder der persönlichen Kommunikation über E-Mail haben sich komplexe Dienste wie Peer-to-Peer (P2P), soziale Netze oder Content Distribution Network (CDN) entwickelt [MHTGK09]. Ebenso hat die Anzahl der an das Internet angeschlossenen Geräte dramatisch zugenommen, ebenso wie der dadurch erzeugte Verkehr.

Diesem Wachstum an Komplexität, Verkehr und Teilnehmeranzahl ist das Internet in seiner jetzigen Form auf Dauer nicht mehr gewachsen. Die Anzahl der IP-Adressen, die jedes einzelne Gerät zur Kommunikation benötigt, werden in den nächsten Jahren vollständig verbraucht sein. Als weitere Folge des rapiden Wachstums nimmt auch die Größe der Routingtabellen im Kernnetz so schnell zu, dass die Verarbeitung dieser Tabellen für die Router zunehmend problematisch wird. Darüber hinaus war das Internet ursprünglich nur für einen kleinen geschlossenen Benutzerkreis gedacht, weshalb keinerlei Sicherheitsmechanismen zur Überprüfung von Integrität und Authentizität vorgesehen sind. Überdies wird das Internet zunehmend zur Informationsbeschaffung verwendet. Dieser neuen Anforderung trägt das aktuelle Internet nur ungenügend Rechnung, das ausschließlich auf die Kommunikation zwischen Geräten ausgelegt ist.

Vielen zusätzlichen Erweiterungen, die gezielt einzelne Probleme entschärfen, ist es zu verdanken, dass das Internet nach wie vor funktionstüchtig ist. Diese Erweiterungen können jedoch oft nicht in Kombination angewandt werden, da sich viele davon gegenseitig behindern. Aus diesem Grund wird sowohl von Regierungen, Industrie und Forschungseinrichtungen der Ruf laut, eine neuartige Internet-Architektur zu entwerfen, die den zukünftigen Anforderungen gewachsen ist. In den letzten Jahren sind dafür viele verschiedene Ansätze und Konzepte entwickelt worden. Diese Arbeiten bieten zwar eine Lösung für eine Vielzahl der Schwächen der heutigen Internet-Architektur, bieten jedoch kein Konzept das die aktuellen Probleme berücksichtigt und zugleich den kommenden Anforderungen, besonders im Bezug auf die zunehmende Inhaltsfokussierung des Internets, gewachsen ist. Diese Arbeit leistet mit einem Konzept für eine Internet-Architektur basierend auf der Trennung von Locator und Identifier ihren Beitrag, das Internet zukunftsfähig zu machen.

1.1. Einordnung der Arbeit

Generell wird bei Konzepten für das Internet der Zukunft zwischen so genannten evolutionären und revolutionären Ansätzen unterschieden. Erstere zeichnen sich dadurch aus, dass sie durch inkrementelle Änderungen zur aktuellen Architektur auftretende Probleme gezielt lösen. Die heutige Internet-Architektur wurde bis jetzt ausschließlich mit evolutionären Ansätzen erweitert. Im Gegensatz dazu zielen revolutionäre Ansätze auf eine komplette Neugestaltung der Internet-Architektur ab. Diese Ansätze beschäftigen sich mit der Änderung bestehender Routing- und Adressierungskonzepte, um das Internet unter Berücksichtigung der neuen Anforderungen zukunftsfähig zu machen. Dabei wird die bestehende physikalische Topologie und Infrastruktur nicht angetastet, um eine mögliche neue Architektur ohne große Änderung der Hardwarekomponenten einführen zu können.

Die vorliegende Arbeit ist den revolutionären Ansätzen zuzuordnen. Es wird eine neue Internet-Architektur vorgestellt, die auf der Trennung von Locator und Identifier basiert. Dieses Konzept greift die Problematik durch die semantische Überladung der IP-Adresse auf, die aktuell sowohl zum Identifizieren eines Gerätes verwendet wird (Identifier), als auch um die Daten dorthin zu routen (Locator). Im Gegensatz zu anderen Konzepten bleibt auch die administrative Topologie des Internets, das aus einzelnen Teilnetzen, den Autonomen Systemen (AS) besteht, unangetastet. Damit wird auch wirtschaftlichen und politischen Forderungen nach Souveränität nachgekommen.

Die Problematik der mangelnden Unterstützung für Inhalte der aktuellen Internet-Architektur kann durch das Konzept der Trennung von Locator und Identifier ebenfalls gelöst werden. Durch die Möglichkeit, auch Inhaltsobjekten einen Identifier zuzuteilen, können diese direkt angesprochen werden.

Nach der Grundidee der Trennung von Locator und Identifier erfolgt jegliche Kommunikation nur über den Identifier. Die Daten werden jedoch anhand des Locators zum Ziel geroutet. Es wird daher ein System benötigt, das den zu einem Identifier gehörenden Locator zurückliefert. Ein solches System wird als Mapping-System bezeichnet und ist integraler Bestandteil einer Architektur basierend auf der Trennung von Locator und Identifier.

1.2. Beitrag der Arbeit

Das Ziel dieser Arbeit ist die Bereitstellung eines durchgängigen Konzepts für eine zukünftige Internet-Architektur. Dieses Konzept setzt auf der bestehenden physikalischen sowie administrativen Topologie des Internets auf und stellt ein neues Adressierungsverfahren für Geräte sowie Inhalte und Personen vor. Damit wird der zukünftigen Entwicklung des Internets Rechnung getragen, das zunehmend zur Informationsbeschaffung und Kommunikation verwendet wird. Das Adressierungsverfahren basiert auf der Trennung von Locator und Identifier und ermöglicht so eine inhärente Unterstützung für Mobilität. Da die Kommunikation ausschließlich über den Identifier erfolgt, kann sich der Locator jederzeit ändern, ohne dass eine aktive Kommunikation unterbrochen wird. Ebenso hilft diese Trennung bei der Reduzierung der Routingtabellengröße, da der Locator völlig unabhängig vom Identifier vergeben werden kann.

Eine wichtige Voraussetzung für eine zukünftige Internet-Architektur ist die Akzeptanz bei den Betreibern des Internets. Diese ist durch die Einführung eines zweigeteilten Locators sowie der unangetasteten Topologie gegeben. Der Locator trägt zum einen dazu bei, dass die Größe der Routingtabellen weiter reduziert wird. Zum anderen erlaubt er den Betreibern für das lokale Routing innerhalb ihres Netzes eigene Adressierungsschemata zu verwenden. Für die Kommunikation zwischen Betreibern wird ein global einheitliches Schema verwendet. Dieses erlaubt das Routing anhand vertraglicher Richtlinien.

Für den Identifier, der für die logische Kommunikation verwendet wird, werden Schemata zur Benennung, Zuteilung und Registrierung vorgestellt. Ein Identifier im vorgestellten Konzept ist permanent gültig und ändert sich über die Lebensdauer des ihm zugewiesenen Objekts nicht. Ein Objekt kann dabei ein Gerät, ein Inhaltsobjekt oder eine Person darstellen.

Für das notwendige Mapping-System werden zwei Konzepte auf der Basis von verteilten Hash-Tabellen (DHT) vorgestellt. Während das erste Konzept besonders auf Leistungsfähigkeit abzielt, so berücksichtigt das zweite Konzept besonders Vertrauensbeziehungen zwischen den Betreibern.

Für die zunehmende Fokussierung des Internets auf die Informationsbeschaffung wird die vorgestellte Internet-Architektur um das neuartige Konzept der Inhaltsmobilität erweitert. Dabei können sich Inhaltsobjekte frei im Netz bewegen und werden immer in der Nähe der Nutzer abgespeichert. Die Inhaltsobjekte werden von einem Algorithmus abhängig von Popularität und Aufrufverteilung ressourceneffizient im Netz auf dedizierte Server verteilt. Die Unterstützung für Inhalte wird durch einen hybriden Speicherdienst abgerundet, der zusätzlich zu den dedizierten Servern auf ein P2P-System zurückgreift, um die Kosten aus der Sicht des Netzes weiter zu senken.

Zusätzlich zur vorgestellten Architektur werden Varianten zur Migration vom aktuellen Internet hin zur vorgestellten Architektur beschrieben. Mittels Tunneling- und Translationsverfahren wird eine Koexistenz der aktuellen mit der neuen Architektur ermöglicht. Aus dieser Koexistenz heraus kann letztendlich schrittweise die vollständige Migration zum Internet der Zukunft erfolgen. Zur Verifikation der vorgestellten Architektur wurden essentielle Komponenten daraus auf der Exerimental-Plattform „German-Lab“ implementiert.

1.3. Struktur der Arbeit

Die weiteren Ausführungen der Arbeit untergliedern sich in sechs Kapitel. In Kapitel 2 wird eine Übersicht über die heutige Internet-Architektur gegeben. Dabei werden die ursprünglichen Planungsziele des Internets vorgestellt sowie die zunehmenden Probleme und Schwächen, die sich aus dem Wandel der Anforderungen ergeben, aufgezeigt. Darüber hinaus werden Konzepte zu evolutionären und revolutionären Ansätzen vorgestellt.

Kapitel 3 beschreibt eine neue Internet-Architektur basierend auf der Trennung von Locator und Identifier. Es wird der zweiteilige Locator eingeführt sowie das Namensschema für Geräte-Identifier.

In Kapitel 4 wird die vorgestellte Architektur um die Unterstützung für Inhalte erweitert. Dazu wird das bereits bestehende Namensschema ergänzt, so dass auch Inhaltsob-

jekten und Personen ein eindeutiger Identifier zugewiesen werden kann. Darüber hinaus wird das Konzept der Inhaltsmobilität mit dem dazugehörigen Algorithmus sowie ein hybrider Speicherdienst eingeführt.

Kapitel 5 stellt verschiedene Migrationsstrategien vor, die einen fließenden Übergang von der heutigen zur der in dieser Arbeit vorgestellten Architektur umgesetzt werden kann. Die Lösung liegt dabei in einer Migrationsphase, in der beide Architekturen koexistieren. Aus dieser Koexistenz erfolgt anschließend die komplette Umstellung.

In Kapitel 6 wird eine Implementierung der vorgestellten Architektur beschrieben. Dabei wurde ein Kernel-Modul entwickelt, das die Kommunikation zwischen Geräten unterstützt und eine einfache Programmierschnittstelle zu den Anwendungen zur Verfügung stellt.

Die Ergebnisse der Arbeit werden in Kapitel 7 zusammengefasst. Mit Ausnahme der Zusammenfassung veranschaulicht Abbildung 1.1 den grundlegenden Aufbau dieser Arbeit.

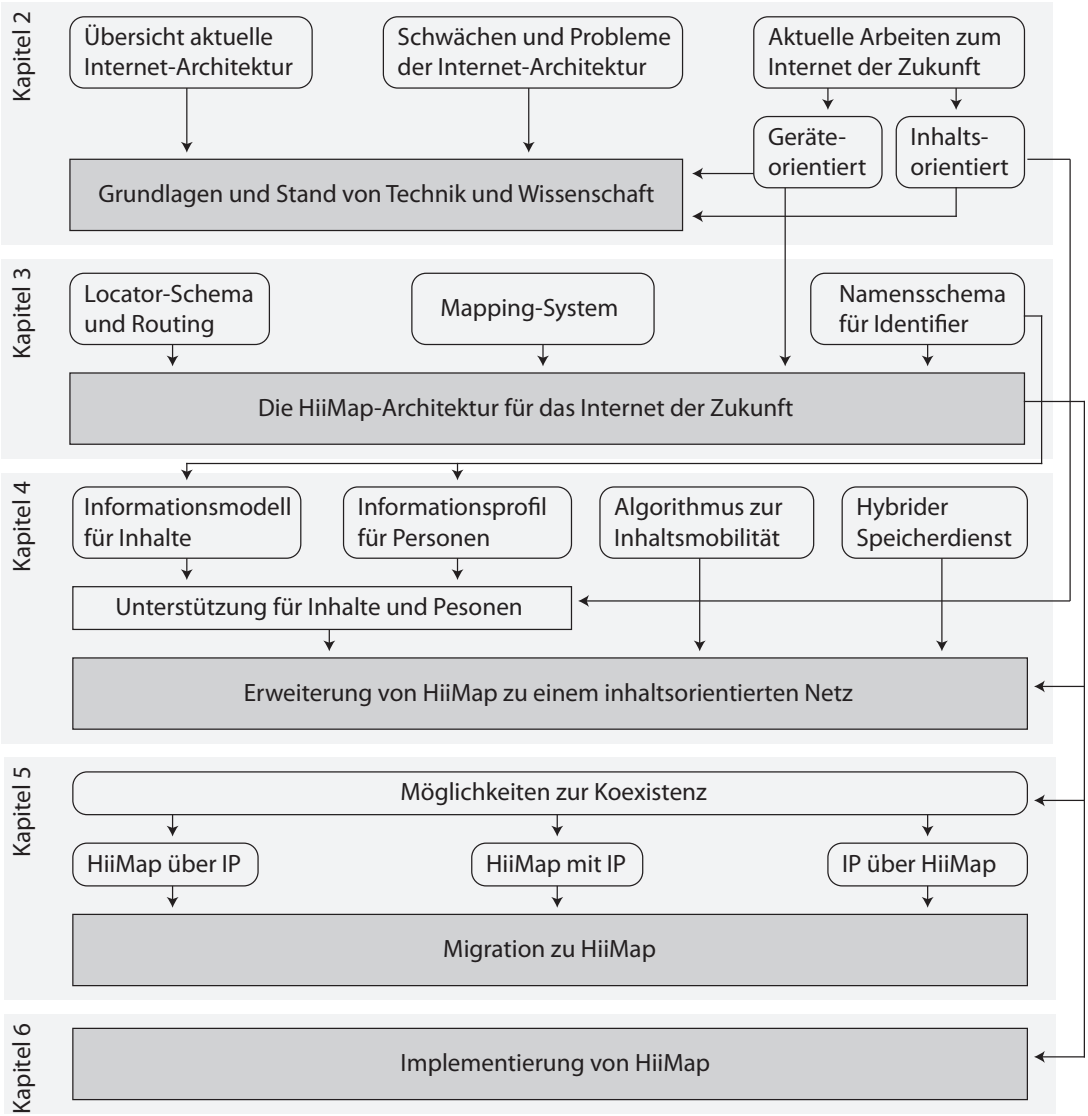


Abbildung 1.1.: Übersicht über den Aufbau der Dissertation

2. Grundlagen und Stand von Technik und Wissenschaft

Dieses Kapitel beschreibt die Grundlagen für die Problemstellungen, die in den nachfolgenden Kapiteln untersucht werden. Zunächst wird im ersten Abschnitt eine Übersicht über die aktuelle Internet-Architektur gegeben. Neben den ursprünglichen Planungszielen des Internets werden auch die Methoden zur Umsetzung dieser Ziele diskutiert.

Der zweite Abschnitt behandelt die größten Schwächen und Lücken der aktuellen Internet-Architektur. Diese sind im Besonderen die zunehmende Adressknappheit, die immer stärker wachsenden Routingtabellen im Kernnetz, sowie die mangelnde Unterstützung für Mobilität, Inhalte und Sicherheitsfunktionen.

Im dritten Abschnitt werden verschiedene grundlegende Planungsansätze für das Internet der Zukunft vorgestellt. Neben den evolutionären Ansätzen, die dem Internet auf den heutigen Stand der Technik verholfen haben, werden auch revolutionären Ansätze vorgestellt. Der anschließende Abschnitt stellt einige aktuelle Forschungsarbeiten aus dem Bereich der revolutionären Ansätze vor, dabei wird das Konzept zur Trennung von Locator und Identifier besonders hervor gehoben. Auch das neue Paradigma des inhaltsbezogenen Netzes (engl. Content Centric Networking) wird in diesem Abschnitt vorgestellt.

2.1. Übersicht über die aktuelle Internet-Architektur

Bereits 1962 stellten Licklider und Clark in [LC62] ihr Konzept des „Galactic Network“ vor. Es handelt sich dabei um eine Vision eines weltweiten Netzes aus Maschinen, die Daten und Programme speichern, auf die jedermann von überall aus zugreifen kann. Aus dieser Idee und vielen anderen Konzepten, unter anderem Arbeiten zur Theorie der Paketvermittlung von Kleinrock in [Kle61], entstand 1969 an der Advanced Research Projects Agency (ARPA) der Vorläufer des heutigen Internets, das ARPANET [LCC+97].

2.1.1. Planungsziele des Internets

Das grundlegende Ziel des ARPANETs bestand zunächst darin, einzelne Forschungsstandorte effizient zu vernetzen, um Rechenkapazität zu sparen und Ressourcen zu teilen. Ein wichtiger Schritt war dabei auch die Anbindung des ARPA Packet Radio Networks an das ARPANET, damit Benutzer des Funknetzes auf Dienste der Maschinen des ARPANET zugreifen konnten. Ausgehend von dieser Problemstellung wurden für das Internet anschließend folgende Planungsziele festgelegt, die nach Priorität sortiert sind [Cla88]:

1. Bestehende Netze müssen miteinander verbunden werden können.

2. Das Internet muss trotz Ausfall von Komponenten funktionstüchtig bleiben.

3. Das Internet muss eine Vielzahl an Kommunikationsdiensten unterstützen.

4. Das Internet muss unterschiedliche physikalische Netzinfrastrukturen unterstützen.

5. Das Internet muss eine verteile Verwaltung der verfügbaren Ressourcen ermöglichen.

6. Die Internet-Architektur muss kosteneffizient sein.

7. Neue Geräte und Maschinen müssen mit geringem Aufwand an das Internet angeschlossen werden können.

8. Die Benutzung von Ressourcen muss zuordenbar sein.

Es gilt zu beachten, dass die Reihenfolge dieser Ziele essentiellen Einfluss auf die Architektur des aktuellen Internet hatte. Da die ARPA (heute DARPA) dem US Verteidigungsministerium unterstand, sollte das Internet auch in militärischem Umfeld einsetzbar sein. Aus diesem Grund stehen Ziele wie das Zusammenschalten von Netzen und die Robustheit bei einem Ausfall von Komponenten an oberster Stelle der Prioritätenliste. Das heutige Internet stellt dabei lediglich eine stetige evolutionäre Weiterentwicklung des ARPANETs dar und basiert nach wie vor auf den Planungszielen seines Vorgängers. Clark postuliert in [Cla88], dass eine vollständig andere Internet-Architektur entstanden wäre, wenn die Planungsziele in einer anderen Reihenfolgen angeordnet worden wären.

2.1.2. Methoden zur Umsetzung der Ziele

Um die wichtigsten Planungsziele des Internets in der oben genannten Reihenfolge umzusetzen, wurden verschiedene Konstruktionsgrundregeln verwendet. Diese werden im Folgenden diskutiert und sind in [Fel07] zusammengefasst. Dabei werden auch die Planungsziele aufgezeigt, die durch die entsprechende Konstruktionsgrundregel erfüllt werden.

Schichtenmodell Die Verwendung eines Netzwerk-Stacks auf der Basis einzelner Schichten, wie in [Zim80] vorgeschlagen, führt zu einer Reduktion der Komplexität des Kommunikationssystems und zu einer Kapselung der Funktionalität. Dabei stellt jede einzelne Schicht einen bestimmten Dienst für die nächsthöhere Schicht zur Verfügung, unter Verwendung der bereitgestellten Dienste der darunter liegenden Schichten. Auf der Seite des Senders durchlaufen die Daten die Schichten von oben nach unten, auf der Seite des Empfängers von unten nach oben. Das OSI-Referenzmodell besteht aus sieben Schichten (vgl. Abbildung 2.1), die von unten nach oben folgende Aufgaben besitzen:

Übertragung: Die unterste, erste Schicht, ist verantwortlich für die Kodierung und Übertragung der Daten auf dem jeweilig verwendeten physikalischen Medium. Sie stellt der nächsthöheren Schicht eine einheitliche Schnittstelle zum bit-weisen Übertragen der Daten zur Verfügung.

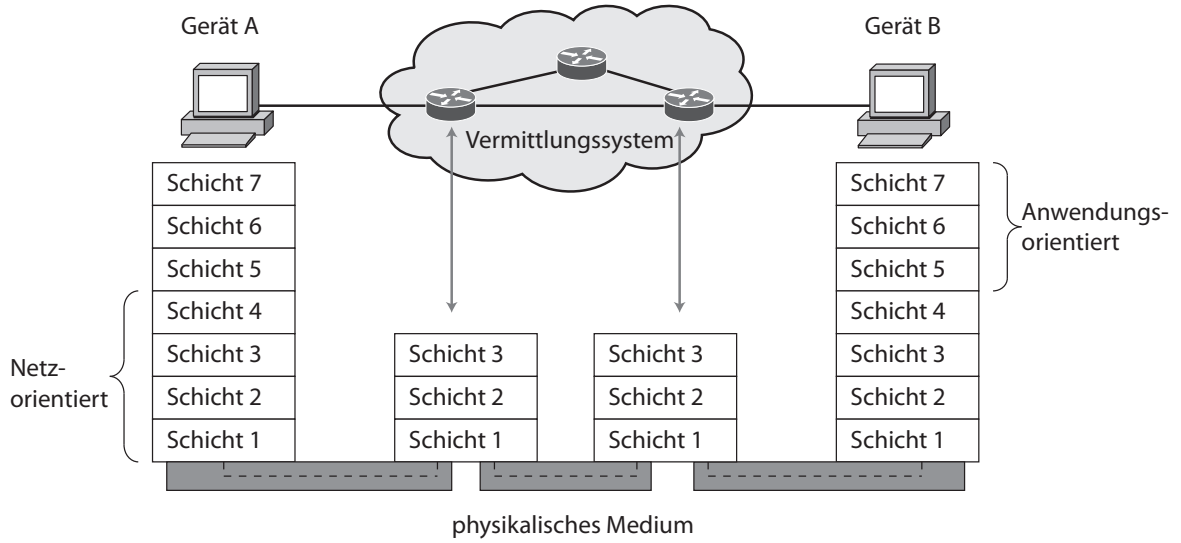


Abbildung 2.1.: OSI-Schichtenmodell

Sicherung: Die zweite Schicht ist für die Kommunikation direkt benachbarter Stationen verantwortlich. Dies wird auch als Forwarding bezeichnet. Jede Netzschnittstelle besitzt dazu eine eigene eindeutige Adresse, die MAC-Adresse. Die zu übertragenden Bits werden dabei in einzelne Rahmen zusammengefasst und mit einer Prüfsumme zur Fehlererkennung versehen.

Vermittlung: Die dritte Schicht, oftmals auch als IP-Schicht bezeichnet, stellt als Dienst die Kommunikation zwischen Endgeräten zur Verfügung. Dazu werden die zu übermittelnden Daten in Pakete gefasst, weshalb diese Schicht auch als Paketvermittlungsschicht bezeichnet wird. Sie beinhaltet ein Adressierungsschema für Geräte, wie beispielsweise die IP-Adresse, und ist für die Wegefindung der Pakete (Routing) verantwortlich. Das IP-Protokoll [RFC81a] in seiner vierten Version (IPv4) hat sich seit 1981 als Quasi-Standard für das Paketvermittlungsprotokoll im heutigen Internet etabliert.

Transport: Die vierte Schicht bietet den anwendungsorientierten Schichten fünf bis sieben eine einheitliche Schnittstelle, um den eigentlichen Kommunikationsprozess zu abstrahieren. Zu den Aufgaben der Transportschicht zählt unter anderem das Bereitstellen einer gesicherten Ende-zu-Ende Verbindung zwischen zwei Endgeräten. Dies beinhaltet Methoden zur Flusskontrolle, Stauvermeidung und Fehlererkennung, wie sie beispielsweise im Transmission Control Protocol (TCP) [RFC81b] realisiert sind. Das User Datagram Protocol (UDP) dagegen erlaubt es, Daten ohne gesicherte Verbindung zu einem anderen Endgerät zu senden. Damit handelt es sich um zwei Transportdienste für verschiedene Anforderungen, die durch das Internet angeboten werden und dem 3. Planungsziel entsprechen.

Sitzung: Die fünfte Schicht ist die unterste anwendungsorientierte Schicht. Sie stellt Dienste für den geregelten Auf- und Abbau von Kommunikationsbeziehungen bereit und ist für deren Wiederaufbau nach Störungen im Transportsystem verantwortlich.

Darstellung: Die Darstellungsschicht als sechste Schicht ist für die systemunabhängige

Darstellung der Daten verantwortlich. Sie ermöglicht den syntaktisch korrekten Austausch zwischen verschiedenen Systemen. Funktionen wie Datenkompression oder Verschlüsselung werden ebenfalls von dieser Schicht übernommen. Beschreibungssprachen wie ASN.1 oder XML sind hier anzusiedeln.

Anwendung: Die Anwendungsschicht stellt die oberste der sieben Schichten des OSI-Modells dar. Zu ihr gehören Protokolle der eigentlichen kommunizierenden Anwendung wie SSH und Telnet (Fernwartung), FTP und NFS (Dateiübertragung), sowie viele andere.

Die Verwendung von einzelnen Kommunikationsschichten erlaubt damit eine einfache Zusammenschaltung bestehender Netze, da die jeweiligen Unterschiede durch die Verwendung von Schichten nach oben hin abstrahiert werden. Die Planungsziele 1, 3 und 4 werden damit unterstützt. Darüber hinaus können auf diese Weise verschiedene Kommunikationsdienste angeboten werden. Abbildung 2.1 zeigt schematisch eine Datenübertragung von zwei Geräten A und B über ein Vermittlungssystem im OSI-Schichtenmodell. Das Vermittlungssystem ist nur bis einschließlich der Vermittlungsschicht in der Datenübertragung involviert.

Paketvermittlung Im Gegensatz zur Leitungsvermittlung, bei der eine Nachrichtenverbindung über einen temporär durchgeschalteten, exklusiven Übertragungskanal mit fester Datenrate erfolgt, werden die Daten bei einem paketvermittelten System in einzelne Pakete aufgeteilt. Diese Pakete besitzen Adressinformationen über Absender und Empfänger und können unabhängig voneinander das Netz durchlaufen. Jedes Paket kann dabei die volle Leitungskapazität verwenden, muss sich aber unter Umständen an Vermittlungsknoten in eine Warteschlange einreihen, während andere Pakete zuerst bedient werden. Darüber hinaus besteht die Möglichkeit, Pakete zu verwerfen, um eine Überlast des Vermittlungsknotens zu vermeiden. Dabei wird jedes Paket grundsätzlich gleich behandelt, was zu der so genannten „best effort“ Qualitätszusicherung führt. Dadurch, dass an weiterleitenden Knoten kein Zustand der aktuellen Verbindungen gespeichert werden muss, kann ein zustandsloses Routingsystem auf der Vermittlungsschicht verwendet werden. Damit wird das 6. Planungsziel erfüllt, das auf eine kosteneffiziente Internet-Architektur abzielt.

Das Konzept der Paketvermittlung wurde von Kleinrock in [Kle61] vorgestellt und 1969 zum ersten Mal praktisch angewandt. Dabei wurden zwei Computer an der UCLA (University of California, Los Angeles) und am SRI (Stanford Research Institute) erfolgreich mit so genannten IMPs (Interface Message Processor) zusammengeschaltet, die als Vermittlungsknoten dienten.

Zusammengeschaltete Netze Das aktuelle Internet besteht aus einer Vielzahl einzelner Netze, die miteinander zusammengeschaltet sind. Daraus leitet sich auch der Name des Internets ab, eine Zusammensetzung der Begriffe *interconnected (zusammengeschaltet)* und *Network (Netz)*. Die Zusammenschaltung wurde besonders durch die Einführung des Schichtenmodells ermöglicht, das mit der Vermittlungsschicht eine einheitliche Schnittstelle für verschiedene darunterliegende Architekturen bietet. Ein einzelnes Teilnetz des Internets wird dabei als Autonomes System (AS) oder Domäne bezeichnet und

von einem Betreiber verwaltet. Die Entscheidung, wie Pakete im Internet geroutet werden, erfolgt anhand der IP-Adresse des Ziels und Routingtabellen, die auf jedem Router existieren und die mit Hilfe eines Routingprotokolls erstellt und aktualisiert werden. Dabei kommen innerhalb eines AS Protokolle wie Open Shortest Path First (OSPF) oder das Intermediate System to Intermediate System Protocol (IS-IS) zum Einsatz, welche die Verkehrslenkung aus der Sicht des Betreibers optimieren. An den Grenzen eines AS bilden Border Gateway Router (BGR) den Übergang in den so genannten Inter-Domain-Bereich, in dem das Border Gateway Protocol (BGP) zum Einsatz kommt. BGP ist ein Pfadvektorprotokoll, das es erlaubt, Routen abhängig von einem zwischen zwei Betreibern geschlossenen Vertragsregelwerk, zu steuern. Dabei können für bestimmte IP-Adressen mehrere Pfade existieren, wobei immer der aus Betreibersicht kostengünstigste gewählt wird. Das 2. Planungsziel wird dadurch erreicht, dass es das Routingsystem erlaubt, Pakete im Bedarfsfall auf anderem Weg durch das Netz zu leiten. Durch Kooperationsverträge zwischen Betreibern wird darüber hinaus auch das 5. Planungsziel gewährleistet, das eine verteilte Verwaltung bestehender Ressourcen ermöglicht.

Ende-zu-Ende Prinzip Die Grundfunktion des Internets beschränkt sich ausschließlich auf die Weiterleitung von Daten von einer Quelle zu einer Senke. Um Überlast zu vermeiden ist es möglich, dass Pakete verworfen werden. Da das Routingsystem zustandslos ist, erfolgt von Seiten des Internets keine Fehlermeldung. Es ist daher die Aufgabe des Endsystems, die Verbindung zu steuern und zu überwachen, und im Falle von verlorenen Paketen diese erneut anzufordern. Man spricht daher von der Platzierung der Intelligenz im Endsystem. Das Internet selbst stellt lediglich ein unzuverlässiges Übertragungssystem dar. Wird eine zuverlässige Übertragung benötigt, so muss das Endgerät den entsprechenden Dienst, wie beispielsweise TCP, bereitstellen [SRC84]. Die grundsätzliche Beschränkung des Internets auf eine konkrete Funktion, nämlich die Zustellung von Datenpaketen, tragen zum Erreichen der Planungsziele 1, 2 und 6 bei.

Dennoch gibt es zunehmend Zweifel an der strikten Einhaltung des Ende-zu-Ende Prinzips sowie an ausschließlich intelligenten Endgeräten [Moo02]. So kann es durchaus sinnvoll sein, zusätzliche Funktionen und Intelligenz im Netz, anstatt bei den Endgeräten zu platzieren. Dies ist der Fall, wenn entweder alle Anwendungen, oder zumindest die Mehrheit davon profitieren. In diesem Fall erspart die Platzierung einer Funktionalität im Netz die Mehrfachimplementierung auf Anwendungsebene.

Durch diese vier Konstruktionsgrundregeln werden die Planungsziele des Internets mit der höchsten Priorität (1 mit 6) erreicht. Mit zusätzlichen Erweiterungen und Protokollen wie BOOTP [CG85], DHCP [Dro97] oder SNMP [CFSD90], werden schlussendlich auch die letzten beiden Planungsziele 7 und 8 erreicht.

2.2. Schwächen der aktuellen Internet-Architektur

Während sich die grundlegende Internet-Architektur in den letzten 30 Jahren kaum verändert hat, haben sich die Anforderungen und besonders das Kommunikationsumfeld stark gewandelt. Das ursprünglich nur für eine geschlossene Benutzergruppe und eine

überschaubare Anzahl von Geräten entworfene ARPANET hat sich zum globalen Netz entwickelt, an das mehrere Milliarden Geräte sowohl von kommerziellen, als auch von privaten Nutzern angeschlossen sind. Besonders seit der Einführung des World Wide Web (WWW) im Jahre 1990, das zum ersten Mal die Informationsbeschaffung vor der ausschließlichen Kommunikation zwischen Geräten in den Vordergrund rückte, sind die Teilnehmerzahlen rasant gewachsen.

Diesen neuen Anforderungen aufgrund des geänderten Nutzerverhaltens, sowie der Vielzahl angeschlossener Geräte, kann das Internet auf absehbare Zeit nicht mehr standhalten. Zwar wurden durch zusätzliche Erweiterungen und neue Protokolle auftretende Probleme immer wieder gelöst, teilweise entstanden dadurch aber auch neue Probleme. Auf einige dieser Erweiterungen wird im Kapitel 2.3.1 genauer eingegangen. Die größten Schwachstellen der aktuellen Architektur werden im Folgenden aufgezeigt und zusammengefasst.

2.2.1. Adressknappheit

Das aktuell auf der Vermittlungsschicht verwendete IP Internet Protokoll in seiner vierten Version (IPv4) verwendet 32 Bit breite Adressen. Prinzipiell können damit etwa 4,3 Milliarden Geräte angesprochen werden. Bereits heute sind jedoch mehr als 6 Milliarden Geräte mit dem Internet verbunden, was nur durch Mechanismen wie das Einführen von privaten Netzbereichen mit anschließender Adressumsetzung möglich ist. Auf dieses Verfahren wird in Kapitel 2.3.1.1 genauer eingegangen. Für das Jahr 2015 werden laut Studien mehr als 12 Milliarden Geräte erwartet [CIS11], womit auch Lösungen mit privaten Adressbereichen zunehmend an ihre Grenzen stoßen werden. Zu dem grundlegend zu knappen Adressbereich kommen jedoch noch weitere Faktoren hinzu, die die Adressproblematik weiter verschärfen.

Ineffiziente Nutzung des Adressraums Zusammen mit dem IPv4-Protokoll [RFC81a] wurden so genannte Netzklassen eingeführt, welche die IPv4-Adresse in zwei Bereiche aufteilt. Einen Adressbereich für das Netz und einen für die im Netz angeschlossenen Geräte. Dadurch wird das Routing vereinfacht, da sich leicht feststellen lässt, ob sich eine Ziel-IP-Adresse im gleichen oder in einem fremden Netz aufhält. Von den so genannten „Klasse A“-Netzen existieren nur 127 Stück, in jedem davon können jedoch theoretisch über 16 Millionen Geräte angeschlossen werden. Von den „Klasse B“-Netzen existieren etwa 65000 mit ebenso vielen Geräten pro Netz. Von „Klasse C“ existieren 16 Millionen Netze, mit maximal 254 Geräten pro Netz. Die Unterscheidung der jeweiligen Klassen erfolgt mit den drei höchstwertigen Bits der IP-Adresse. Da der Adressraum der Klassen B und C für einige große Unternehmen und Universitäten zu gering war, wurde ein Klasse A-Netz zugewiesen. Dadurch wurden jedoch viele theoretisch mögliche freie IPv4-Adressen unbrauchbar gemacht, da diese nicht mehr für andere zur Verfügung stehen. Dieses Problem wurde durch die Einführung von Subnetzmasken [MP85] und Classless Inter-Domain Routing (CIDR) [RFC93] 1993 weitgehend beseitigt. Durch die zusätzliche bitweise Angabe der Subnetzmaske können die noch nicht vergebenen IPv4-Adressen deutlich feiner und effizienter in Netz- und Gerätebereich aufgeteilt werden.

Internet-Demographie und permanente Verbindung In Entwicklungs- und Schwellenländern wie China und Indien war vor 10 bis 15 Jahren nur eine verschwindend kleine Anzahl von Geräten mit dem Internet verbunden. Jetzt besitzen in diesen aufstrebenden, bevölkerungsreichen Nationen eine Vielzahl von privaten Haushalten einen Internet-Anschluss, so dass die Adressknappheit verschärft wird. Hinzu kommt die Tatsache, dass heutige Breitbandverbindungen, im Gegensatz zu den früheren Einwahlverbindungen per Modem, eine quasi permanente Konnektivität besitzen. Da eine Einwahlverbindung üblicherweise nur eine begrenzte Zeit aufrecht erhalten wird, konnte eine geringe Anzahl an verfügbaren IPv4-Adressen unter den einzelnen Teilnehmern im Zeitmultiplex aufgeteilt werden. Bei einer permanenten Breitbandverbindung ist jedoch eine IPv4-Adresse nicht mehr für andere verfügbar.

Mobile Geräte Die Zunahme von internetfähigen Mobilgeräten wie Smartphones, Tablets oder PDAs führt dazu, dass auch diese Geräte eine IP-Adresse benötigen. Dabei gilt auch für diese Geräte, solange Netzabdeckung besteht, grundsätzlich eine permanente Konnektivität mit quasi fest zugewiesener IP-Adresse. Zusätzlich besitzen diese Geräte üblicherweise mehrere Netzschnittstellen, denen je eine eigene IP-Adresse zugewiesen ist.

Die Internet Assigned Numbers Authority (IANA), welche für die Vergabe der IP-Adressen zuständig ist, hat am 3. Februar 2011 den letzten freien, nicht reservierten /8-IPv4-Adressblock (entspricht Klasse A) an die für den Atlantik- und Pazifikraum zuständige Registrierungsstelle APNIC vergeben [Asi11]. Damit wurde die sogenannte „exhaustion Phase“ [Org11] eingeläutet, welche vorsieht, die letzten verbliebenen fünf /8-Adressblöcke an die jeweiligen zuständigen regionalen Registrierungsstellen zu verteilen. Die APNIC rechnet damit, dass der letzte Block für voraussichtlich weitere fünf Jahre ausreichen wird. Danach ist der nutzbare Adressraum definitiv vollständig ausgeschöpft. Die Skalierbarkeit des Internets ist mit dem IPv4-Adressraum damit an ihre Grenzen gestoßen.

2.2.2. Wachsende Routingtabellengrößen

Eine weitere Problematik, die ebenfalls die Skalierbarkeit des Internets betrifft, ist die immer schneller wachsende Anzahl von Einträgen in den BGP-Routingtabellen im Kernnetz[1]. IP-Adressblöcke, die über eine gemeinsame Verbindung erreicht werden können, lassen sich zu einem so genannten Routingpräfix mit Angabe der Subnetzmaske zusammenfassen. Jedes Routingpräfix stellt dabei einen Eintrag in der BGP-Routingtabelle dar. Grundsätzlich sind nur maximal so viele Routingpräfixe nötig, wie IP-Adressblöcke vergeben sind. Dennoch wächst die Anzahl der Einträge in den Routingtabellen schneller als die eigentlich zu adressierenden IPv4-Adressblöcke [MXZ+05]. Diese Entwicklung ist in Abbildung 2.2 dargestellt. Die Datensätze stammen von [Hus11a] und [Hus11b]. Im Dezember 2011 existierten etwa viermal mehr Einträge in den BGP-Routingtabellen als

[1] Unter dem Begriff Kernnetz, auch Default Free Zone (DFZ) bezeichnet, versteht man diejenigen AS, die keine Standard-Routen mehr besitzen. Stattdessen besitzen diese eine so genannte vollständige BGP-Routingtabelle, in der alle aktuell erreichbaren Ziele aufgelistet sind.

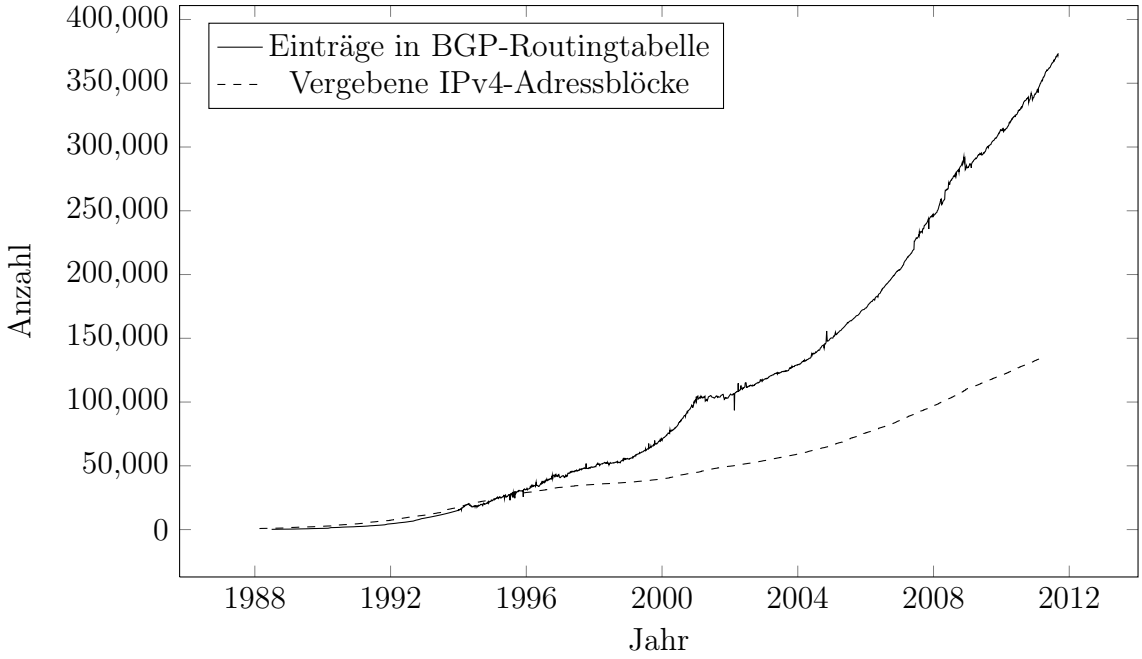


Abbildung 2.2.: Entwicklung von vergebenen IPv4-Adressblöcken und BGP-Einträgen

vergebene IPv4-Adressblöcke. Dies führt dazu, dass die Last auf die Router im Kernnetz stark zunimmt, da für jedes einzelne weiterzuleitende Paket die komplette Routingtabelle abgearbeitet werden muss. Darüber hinaus ist auch die Wartung und Aktualisierung derart großer Tabellen sehr rechenaufwändig. Ursache für diese Entwicklung und die Deaggregation des Adressraumes, bei der auch IPv4-Adressbereiche einen eigenen Eintrag besitzen, die eigentlich durch ein allgemeineres Präfix abgedeckt sind, werden in [CMU+10, MXZ+05] zusammengefasst und lassen sich auf folgende Punkte zurückführen:

Multihoming Unter Multihoming versteht man eine Technik, mit der das IP-Netz eines Kunden oder eines Unternehmens über mehrere verschiedene Betreiber angebunden ist. Dieses Verfahren wird angewandt, um die Ausfallsicherheit zu erhöhen, da im Fehlerfall der aktiven Verbindung auf eine Ersatzverbindung umgeschaltet werden kann. Dies führt jedoch zu einem Zuwachs der BGP-Routingtabellen, da der entsprechende IP-Adressbereich dadurch nicht mehr in einem allgemeinen Routingpräfix zusammengefasst werden kann, das einen größeren Netzbereich abdeckt. Stattdessen ist für den multihomed Netzbereich explizit ein eigenes Präfix in der BGP-Routingtabelle erforderlich, das die verschiedenen möglichen Routen zu diesem Adressbereich enthält. Der Adressraum wird dadurch deaggregiert und die BGP-Tabelle vergrößert.

Betreiberunabhängige Adressen Diese Art IP-Adressbereich wird einem Endnutzer direkt von der zuständigen Registrierungsstelle zugeteilt und ist unabhängig von einem bestimmten Betreiber. Derartige Adressen sind besonders für größere Unternehmen attraktiv, die eine Neunummerierung aller Geräte in ihrem Netz im Falle eines Betrei-

berwechsels vermeiden wollen. Der aktuelle Netzbetreiber des Endnutzers muss jedoch für diesen Netzbereich ein eigenes BGP-Routingpräfix ankündigen, da sich dieser IP-Adressbereich nicht in das Präfix aggregieren lässt, welches die sonstigen IP-Adressen des Betreibers abdeckt.

Verkehrsplanung Im Inter-Domain-Bereich existieren so gut wie keine Möglichkeiten zur Verkehrsplanung, abgesehen von der Modifikation der BGP-Routingtabelle durch das künstliche Einfügen zusätzlicher Routingpräfixe [FBR03]. Dies betrifft AS, die aufgrund von Multihoming mit mehr als nur einem weiteren AS verbunden sind. Diese AS können ihren zugeteilten IP-Adressbereich bewusst in kleinere Bereiche aufteilen und für jeden dieser Bereiche ein eigenes Präfix via BGP ankündigen. Dies ermöglicht zwar eine gezielte Steuerung des Datenverkehrs, deaggregiert jedoch den Adressbereich, was ebenfalls zu einer Vergrößerung der BGP-Routingtabelle führt.

2.2.3. Fehlende Sicherheit

Die aktuelle Internetarchitektur war für eine geschlossene Benutzergruppe ausgelegt, in der keine Mechanismen zur Überprüfung von Authentizität und Integrität sowie zur Verschlüsselung nötig waren. Sicherheit wurde daher auch nicht in die Liste der Planungsziele für das Internet aufgenommen. Mit der Öffnung des Internets für die Allgemeinheit stellt das Fehlen von Sicherheitsmechanismen aber zunehmend ein grundsätzliches Problem dar. Zwar sieht das Ende-zu-Ende Prinzip vor, dass jede Anwendung, die einen bestimmten Sicherheitsmechanismus benötigt, diesen selbst bereitstellen muss, dennoch gilt der Grundsatz, dass ein System immer nur so sicher ist, wie seine unsicherste Komponente.

Deshalb wird heute von fast allen Benutzern für Transaktionen, die über das Internet abgewickelt werden, Sicherheit gefordert. Sei es beim Einkaufen über das Internet, beim Online-Banking oder bei der Telepräsenz. Diese Beispiele erfordern sowohl Authentizität, Integrität als auch Verschlüsselung für die übertragenen Daten. Für die häufigsten Anwendungsfälle ist letztere zwar nicht nötig, es ist jedoch im Interesse jeder Anwendung, die Daten über das Internet austauscht, Datenmanipulation zu erkennen und die Echtheit des Absenders sicherzustellen zu können.

2.2.4. Fehlende Unterstützung von Mobilität

Eine weitere Schwäche der aktuellen Internet-Architektur, die sich erst in den letzten zehn Jahren mit dem zunehmenden Aufkommen von mobilen Endgeräten gezeigt hat, ist die fehlende Unterstützung für Mobilität. Darunter versteht man die Möglichkeit, den Netzzugangspunkt eines Endgerätes zu wechseln, ohne dass bestehende Verbindungen getrennt werden. Ursache für diese Problematik ist die IP-Adresse, die semantisch mit zwei grundsätzlich verschiedenen Aufgaben überladen ist. Zum einen wird die IP-Adresse zum Identifizieren des Endgerätes verwendet, zum anderen aber auch, um dessen Aufenthaltsort im Internet zu bestimmen und die Daten dorthin zu routen. Datenverbindungen, die von einer Anwendung auf der Basis einer bestimmten IP-Adresse aufgebaut wurden, werden unweigerlich getrennt, falls das mobile Endgerät seinen Netzzugangspunkt wechselt und eine neue IP-Adresse erhält. Abbildung 2.3 zeigt diese Problematik.

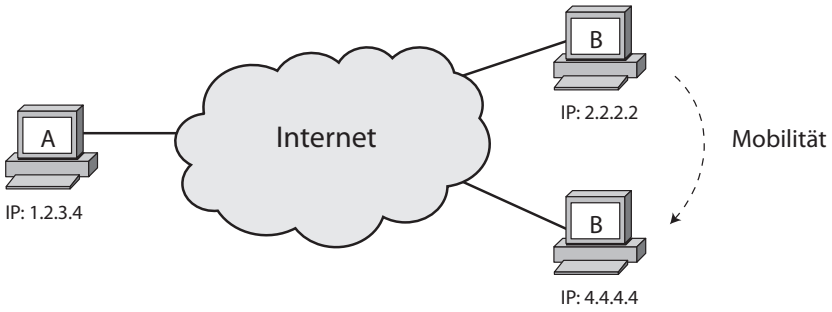


Abbildung 2.3.: Wechsel der IP-Adresse aufgrund von Mobilität

Gerät B wechselt seinen Netzzugangspunkt und erhält eine neue IP-Adresse. Eine aktive Kommunikation mit Gerät A wird in diesem Fall getrennt.

Damit die Skalierbarkeit des Routings sichergestellt werden kann, sind IP-Adressen hierarchisch aufgebaut. Um Mobilität unter der weiteren Verwendung der IP-Adresse zu ermöglichen, muss entweder die hierarchische Adressstruktur aufgegeben, oder eine zusätzliche Adresse eingeführt werden. Aufgrund der bereits problematischen Skalierungseigenschaften des Internets scheidet der erste Vorschlag aus, für den zweiten existieren jedoch bereits Lösungen. Mit dem Mobile IP-Konzept (Kapitel 2.3.1.4) kommen Adressenumsetzer, so genannte Agenten, zum Einsatz. Andere Konzepte sehen eine komplette Trennung der IP-Adresse in zwei unabhängige Adressen für die Identifizierung und für das Routing vor. Diese Ansätze werden in Kapitel 2.4.1 vorgestellt.

2.2.5. Mangelnde Unterstützung für Inhalte

Mit der Einführung des WWW durch Tim Berners-Lee in [BL89] kam 1989 eine neue Erweiterung zum Internet hinzu, die es zum ersten Mal erlaubte, Informationen über so genannte Web-Seiten bereitzustellen und diese mit einer geeigneten Bescheibungssprache darzustellen und miteinander zu verknüpfen (HTML). Zur Übertragung wurde das anwendungsorientierte Protokoll HTTP eingeführt, das TCP als Transportprotokoll benutzt. Heute wird das WWW von praktisch allen Unternehmen verwendet, um ihre Produkte und sich selbst zu präsentieren. Zeitungen und Verlage veröffentlichen darüber Nachrichten und Artikel, und Portale wie Youtube ermöglichen den Konsum von Medien. Das WWW erzeugt heute den mit Abstand meisten Datenverkehr im Internet im Vergleich zu anderen Diensten wie Email, FTP oder Telnet [BDF+09, BMM+08].

Besonders beliebte Web-Seiten stellen das WWW jedoch vor schwierige Herausforderungen, da Geräte, welche populäre Inhalte speichern, oft durch die Menge der Anfragen überlastet werden. Zudem existiert im WWW kein Schema, Inhalte unabhängig von dem Gerät, auf dem sie abgespeichert sind, zu adressieren. Da in jedem Uniform Ressource Identifier (URI), der einen Inhalt im WWW global eindeutig adressiert, immer der Gerätename bzw. die IP-Adresse angegeben wird, ist die entsprechende Information an diesen Speicherort gebunden. Ändert sich der Speicherort, muss auch der URI geändert werden.

2.3. Ansätze zur Weiterentwicklung des Internets

Aufgrund der ständig wachsenden Anforderungen an das Internet bezüglich Skalierbarkeit und Benutzerverhalten muss dessen Architektur ständig erweitert werden. Dabei existieren zwei grundlegend verschiedene Ansätze, wie das Internet für zukünftige Anforderungen gerüstet werden kann.

2.3.1. Evolutionäre Ansätze

Unter evolutionären Ansätzen versteht man die Weiterentwicklung des Internets mit inkrementellen Veränderungen, wie dem Hinzufügen neuer Protokolle und Erweiterungen, ohne jedoch die Grundarchitektur anzutasten. In den letzten 30 Jahren wurde das Internet erfolgreich mit evolutionären Ansätzen weiterentwickelt. Immer neue Erweiterungen, die zur bereits bestehenden Architektur hinzugefügt wurden, konnten die jeweils neu auftretenden Probleme weitgehend lösen und werden im Folgenden vorgestellt.

2.3.1.1. Network Address Translation (NAT)

Bereits Anfang der 1990er Jahre hat man erkannt, dass IP-Adressen eine begrenzte Ressource darstellen und mit Network Address Translation (NAT) einen Mechanismus eingeführt, der den Bedarf an IP-Adressen drastisch reduziert [EF94]. Vorraussetzung dafür waren die in [RMKdG94] eingeführten privaten IP-Adressen, die zunächst nur von Unternehmen verwendet wurden, die keinen Anschluss an das Internet besaßen. Im Gegensatz zu den öffentlichen IP-Adressen, die im Internet geroutet werden und nur jeweils einmal existieren, können private IP-Adressen in physikalisch getrennten Netzen mehrfach verwendet werden. Mit NAT wird Geräten mit privater IP-Adresse die Kommunikation im Internet ermöglicht, wobei für ein komplettes privates Netzsegment nur eine öffentliche IP-Adresse benötigt wird.

Die Umsetzung vom privaten in den öffentlichen IP-Adressbereich wird von einem NAT-Translator übernommen, der als Standard-Gateway für alle internen Geräte fungiert. Der Translator besitzt dazu zwei Netzschnittstellen, von denen eine mit dem internen Netz und die andere mit dem Internet verbunden ist und entsprechende IP-Adressen besitzen. In den IP-Paketen, die von internen Geräten zu Zielen im Internet versendet werden, wird die Quell-IP-Adresse durch den Translator in dessen öffentliche IP-Adresse getauscht. Zusätzlich werden auch die Portnummern der Transportprotokolle wie TCP oder UDP getauscht, um Zweideutigkeit bei der Rückübersetzung der Antwortpakete zu vermeiden. Der NAT-Translator muss dazu zusätzlich die Informationen aus der Transportschicht auswerten und verwaltet dazu eine NAT-Tabelle. In dieser sind alle aktuellen Verbindungen mit den entsprechenden IP- und Transportprotokollinformationen aufgelistet, um die Antwortpakete wieder an das korrekte interne Gerät nebst richtiger TCP- oder UDP-Portnummer zustellen zu können.

Durch das Umschreiben der Packet-Header verstößt NAT jedoch gegen das Ende-zu-Ende Prinzip. Geräte in einem privaten Netzsegment können nicht ohne weiteres von Geräten außerhalb erreicht werden, da sie für außenstehende Geräte nicht sichtbar sind. Dies wird erst möglich, wenn ein internes Gerät initial Daten sendet, beziehungsweise wenn

feste Einträge in der NAT-Tabelle gesetzt werden. Im Falle des verbindungsorientierten TCP bleibt der Eintrag dabei so lange gültig, bis die Verbindung beendet wird. Deutlich komplizierter ist der Sachverhalt bei UDP, da für Datagrammpakete keine eigene Verbindung aufgebaut wird. Zwar werden auch für UDP-Pakete Einträge in der NAT-Tabelle erstellt, diese werden jedoch wieder entfernt, wenn innerhalb weniger Sekunden kein neues Paket gesendet wird. Abhilfe schaffen nur Mechanismen wie Keep-Alive-Pakete ohne Inhalt, die nur dazu dienen den NAT-Tabelleneintrag zu erhalten.

Mit Techniken wie UDP bzw. TCP Hole Punching ist es möglich, initial Daten an Geräte hinter einem NAT-Translator zu senden. Besonders Anwendungen, die auf Peer-to-Peer Protokollen basieren, machen sich diese Technik zu Nutze [FS09]. Dieser Aufwand ist jedoch sehr hoch, da immer ein Hilfs-Gerät mit öffentlicher IP-Adresse benötigt wird, zu dem eine permanente Verbindung gehalten wird.

2.3.1.2. Internet Protokoll Version 6 (IPv6)

Trotz der Einführung von NAT zeichnete sich bereits früh der Trend ab, dass der IPv4-Adressbereich in naher Zukunft erschöpft sein wird. Um auch das Ende-zu-Ende Prinzip wieder wahren zu können, wurde 1998 das Internet Protokoll mit der Versionsnummer sechs (IPv6) eingeführt, das den Adressraum von 32 Bit auf 128 Bit erweitert [DH98]. IPv6 ist ein eigenes Vermittlungsprotokoll und stellt die gleichen Dienste für die Transportschicht zur Verfügung wie sein Vorgänger IPv4. Die Protokolle der Transportschicht müssen daher nicht geändert werden. Der erweiterte Adressraum ist so dimensioniert, dass auch langfristig genügend öffentliche IPv6-Adressen zur Verfügung stehen[2]. Die Trennung zwischen Netz- und Gerätebereich der IPv6-Adresse erfolgt mit Subnetzmasken, wodurch von Beginn an ein unnötiges Verschwenden von IPv6-Adressen vermieden wird.

IPv6 stellt eine langfristige Lösung zur Behebung der Adressknappheit der alten IPv4-Adressen dar. Da mit IPv6 jedem am Internet angeschlossenen Gerät eine öffentliche und routingfähige Adresse zugewiesen werden kann, wird auch NAT grundsätzlich überflüssig. Da IPv6 ein eigenes Vermittlungsprotokoll darstellt, ist der Parallelbetrieb mit IPv4 möglich und gestattet eine einfache Migration, ohne an einem Stichtag komplett auf das neue Vermittlungsprotokoll umstellen zu müssen. Das Problem der wachsenden BGP-Tabellengrößen wird durch IPv6 jedoch verschärft, da durch den deutlich vergrößerten Adressraum ebenfalls deutlich mehr Präfixe benötigt werden.

2.3.1.3. Sicherheit mit IPsec und TLS

Aus der Forderung nach einer Möglichkeit zur verschlüsselten Datenübertragung mit der Möglichkeit zur Überprüfung von Authentizität und Integrität, wurden die Protokolle Internet Protocol Security (IPsec) [KA98] und Transport Layer Security (TLS) [DA98] eingeführt. Beide können die geforderten Schutzziele wie Authentizität, Integrität und

[2]Mit einem Adressraum von 128 Bit entfallen auf jeden Quadratmeter Erde einschließlich Ozeane $7 \cdot 10^{23}$ IPv6-Adressen. Selbst bei ineffizienter Nutzung bleiben laut [Hui94] noch weit über 1000 IPv6-Adressen pro Quadratmeter Erdoberfläche.

Vertraulichkeit für die Datenübertragung garantieren, setzen aber jeweils an verschiedenen Stellen im Schichtenmodell an.

IPsec Dieses Sicherheitsprotokoll ist im Schichtenmodell zwischen der Vermittlungsschicht und der Transportschicht einzuordnen und stellt als Dienst mehrere Möglichkeiten zur Datensicherheit zur Verfügung. Mittels eines zusätzlich eingefügten Authentication Header (AH) wird ein Hashwert aus einer Prüfsumme über das gesamte Datenpaket und einem Schlüssel übertragen. Der Empfänger kann diese Prüfsumme zusammen mit dem gleichen Schlüssel ebenfalls berechnen. Stimmen die Werte nicht überein, wurde das Paket verändert und wird vom Empfänger verworfen.

Mit dem AH kann Integrität und Authentizität der Übertragung gewährleistet werden, jedoch noch keine Verschlüsselung. Um dieses Schutzziel ebenfalls zu erreichen, verwendet IPsec die Encapsulating Security Payload (ESP). Mit ESP wird die gesamte Information ab der Transportschicht mittels eines kryptographischen Verfahrens und dem dazugehörigen Schlüssel verschlüsselt. Authentifizierungsinformationen in Form einer Prüfsumme werden an das Ende des Datenpaketes gestellt und sind optional.

Während mit IPsec alle ursprünglich geforderten Schutzziele erreicht werden können, ist dieses Protokoll jedoch nicht zusammen mit NAT verwendbar. Da NAT die Quell- und Ziel-IP-Adresse verändert, stimmen bei der Verwendung des AH die Prüfsummen auf der Empfängerseite nicht mehr überein. Im Fall von ESP sind die Informationen aus der Transportschicht für den NAT-Translator nicht mehr lesbar, wodurch keine Zuordnungen in der NAT-Tabelle eingetragen werden können. IPsec wird vor allem von Unternehmen und Firmen verwendet, um Standorte verschlüsselt zu vernetzen.

TLS Im Gegensatz zu IPsec ist TLS in der anwendungsorientierten Schicht angesiedelt und lässt die Informationen der netzorientierten Schichten unangetastet beziehungsweise verwendet diese nicht für die Berechnung von Prüfsummen. TLS kann daher auch zusammen mit NAT eingesetzt werden und erreicht alle geforderten Schutzziele. TLS basiert auf symmetrischer Verschlüsselung mit gleichen Schlüsseln. Zum Austausch dieser Schlüssel wird eine Public-Key Infrastruktur (PKI) eingesetzt, die auf einem asymmetrischen Verschlüsselungsverfahren basiert. Dabei werden mit mathematischen Verfahren, z.B. RSA [RSA78], jeweils ein privater und öffentlicher Schlüssel generiert, wobei der öffentliche Schlüssel zum Verschlüsseln von Nachrichten sowie zum Überprüfen von Signaturen verwendet wird. Der private Schlüssel wird zum Entschlüsseln von Nachrichten verwendet und zum Erstellen von Signaturen. Mittels Zertifikaten wird der öffentliche Schlüssel für die Allgemeinheit verfügbar gemacht.

Da TLS ein Verschlüsselungsprotokoll auf Anwendungseben ist, müssen die kommunizierenden Anwendungen dieses beherrschen und können nicht, wie bei IPsec, auf einen Dienst aus der netzorientierten Schicht zugreifen. Zudem entstehen potentielle Sicherheitslücken durch verschiedene Implementierungsformen in den entsprechenden Anwendungen. TLS wird heute hauptsächlich für verschlüsselte Webseiten über HTTPS verwendet.

2.3.1.4. Mobile IP

Mit Mobile IP existiert ein Protokoll, das trotz der semantischen Überbelegung der IP-Adresse zu Zwecken der Endgeräteidentifikation und des Routings, Geräten den Wechsel der IP-Adresse ermöglicht, und diese trotzdem für andere Geräte unter einer bestimmten IP-Adresse erreichbar bleiben. Je nach verwendetem Vermittlungsprotokoll, IPv4 oder IPv6, kommen dabei verschiedene Verfahren zum Einsatz [BH07].

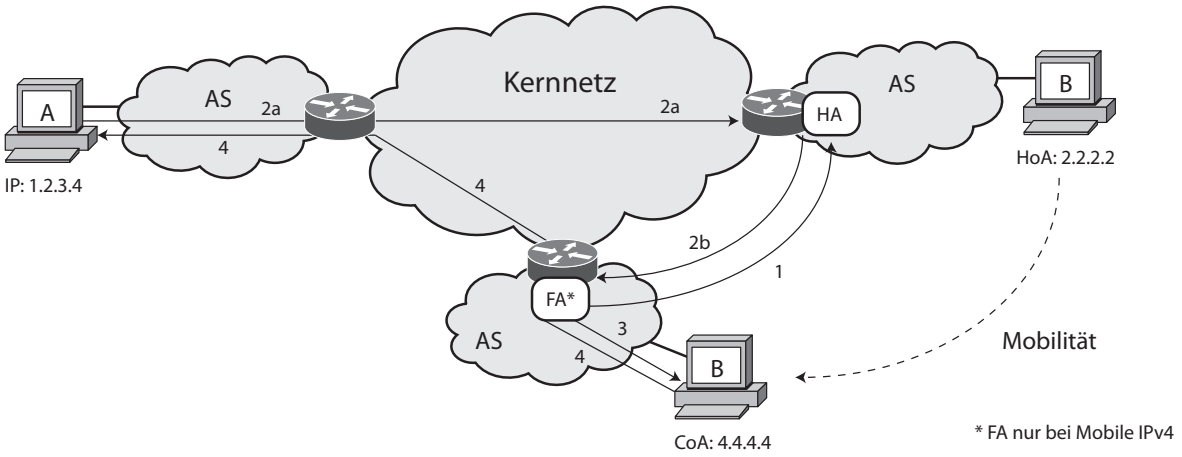


Abbildung 2.4.: Mobilitätsunterstützung mit Mobile IP

Mobile IPv4 Die IP-Adresse eines mobilen Geräts (B) in seinem Heimatnetz wird als Home Address (HoA) bezeichnet (vgl. Abbildung 2.4). Wechselt dieses Gerät seinen Netzzugangspunkt temporär zu einem anderen Netzbetreiber, müssen die Datenpakete in das Fremdnetz nachgesendet werden. Dazu wird dem mobilen Gerät im besuchten Fremdnetz eine temporäre Care-of-Address (CoA) zugewiesen. Das mobile Gerät wird dabei im Heimatnetz und im besuchten Netz von so genannten Mobility Agents betreut. Der Home Agent (HA) ist im Heimatnetz angesiedelt, der Foreign Agent (FA) im besuchten Fremdnetz. Nach der Zuteilung einer CoA durch den FA an das mobile Gerät wird der HA vom FA darüber informiert (1). Dieser leitet jetzt ankommende Datenpakete, welche an die HoA adressiert sind (2a) und daher im Heimatnetz ankommen, mittels einer IP-in-IP-Kapselung an den FA im Fremdnetz weiter (2b). Dieser sendet das Paket anschließend an die CoA des mobilen Geräts (3). Antwortpakete an Geräte (4), die mit dem mobilen Gerät kommunizieren, müssen über den FA gesendet werden, können dann aber direkt, ohne den Umweg über den HA, geroutet werden [Per10].

Während mit diesem Verfahren prinzipiell Mobilität voll unterstützt wird, bringt es einige Nachteile mit sich. So sind die Routen für Pakete, die immer zunächst an den HA und anschließend an den FA gesendet werden müssen, um das mobile Gerät zu erreichen, sehr ineffizient. Eine eventuell kürzere direkte Route zwischen beiden Geräte kann nicht verwendet werden. Durch die IP-in-IP-Kapselung zwischen HA und mobilem Gerät wird der zu übertragende Overhead deutlich erhöht. Darüber hinaus ist dieses Verfahren nur mit den Helferinstanzen HA und FA einsetzbar.

Mobile IPv6 Das Mobilitätsprotokoll für IPv6 bietet im Vergleich zur IPv4-Version zusätzliche Möglichkeiten, um Mobilität effizienter zu gestalten. Ein wichtiger Unterschied ist der Wegfall des FA bei Mobile IPv6. Dem mobilen Gerät wird vom Router des besuchten Fremdnetzes direkt die CoA zugewiesen, die das Gerät direkt an den HA im Heimatnetz übermittelt. Die initiale Kommunikation mit dem mobilen Gerät erfolgt zunächst über den HA, der das Paket ebenfalls über IP-in-IP-Kapselung, diesmal aber direkt an das mobile Gerät weiterleitet. Dieses kann anschließend direkt und ohne Umweg über einen FA antworten. Unterstützen beide kommunizierenden Geräte Mobile IPv6, so ist es möglich, für die folgenden Datenpakete ein sogenanntes Binding Update zu senden, damit der Kommunikationspartner des mobilen Geräts direkt die CoA verwenden kann. Pakete können damit deutlich effizienter geroutet werden, da der Umweg über den HA entfällt. Unterstützt ein Kommunikationspartner kein Mobile IPv6, müssen dessen Pakete zum mobilen Gerät jedoch weiterhin über den HA gesendet werden, ebenso wie die initiale Kommunikation immer über den HA erfolgen muss [JPA04].

Auch Mobile IPv6 kommt damit nicht ohne Helferinstanz aus, benötigt jedoch nur noch den HA. Nachteile durch ineffiziente Routen können nur vermieden werden, wenn beide Geräte das Verfahren unterstützen.

2.3.1.5. Caching

Geräte, die im WWW besonders beliebte Inhalte bereit stellen, können sehr schnell überlastet werden. Um diese Problematik in den Griff zu bekommen, wurde mit Caching eine Lösung für dieses Problem eingeführt. Das Prinzip von Caching wird auch in vielen anderen Bereichen der EDV verwendet und ist immer ähnlich: Einmal verwendete Daten und Informationen werden näher beim Nutzer zwischengespeichert, um sie bei einem erneuten Zugriff schneller zur Verfügung stellen zu können.

Im WWW versucht man damit, die Latenzzeiten beim Aufruf von Informationen zu verringern sowie die Verfügbarkeit des Web-Servers zu erhöhen, indem der Datentransfer reduziert wird und Spitzen abgefedert werden. Erreicht wird dies durch Zwischenspeichern der entsprechenden Daten und Informationen. Dies kann sowohl direkt auf dem anfragenden Gerät geschehen, die häufigste Form sind aber von den Betreibern bereit gestellte Proxy-Caches und transparente Caches [Hus99]. Im Falle eines Proxy-Caches schickt das Gerät des Konsumenten seine Anfrage direkt an den Proxy-Cache, der die Anfrage verarbeitet, die Daten zustellt und gleichzeitig zwischenspeichert. Anfragen für die gleichen Daten kann er lokal aus seinem Speicher bedienen. Bei einem transparenten Cache dagegen wird die Anfrage aus Sicht des Konsumenden direkt an den gewünschten Web-Server gesendet. Dazwischenliegende Router des Betreibers können den Paketinhalt analysieren und die Daten wie ein Proxy nach Eintreffen zwischenspeichern, um sie bei weiteren Anfragen lokal bedienen zu können. Die transparente Analyse aller Datenpakete ist jedoch besonders rechenintensiv und verletzt das Ende-zu-Ende Prinzip, ohne dass dem Nutzer dies bewusst ist (im Gegensatz zur expliziten Verwendung eines Proxy-Caches).

Für jeden Cache sind darüber hinaus geeignete Verdrängungs- und Ersetzungsstrategien nötig, da die verfügbare Speicherkapazität endlich ist und die Daten unter Umständen veraltete Informationen enthalten. Gebräuchlich sind dabei die Strategien Least

Frequently Used (LFU), bei der Objekte aus dem Cache entfernt werden, die im Vergleich zu anderen am wenigsten oft aufgerufen werden, und Least Recently Used (LRU), bei der die ältesten nicht verwendeten Objekte entfernt werden. Die so genannte Cache-Kohärenz stellt auch eines der größten Probleme dieser Technik dar, da derjenige, der Daten und Informationen bereitstellt, nicht weiß ob, wo und in welcher Version seine Daten zwischengespeichert sind. Zwar kann den Inhalten im WWW explizit mitgeteilt werden, wie lange Daten zwischengespeichert werden dürfen, es ist jedoch nicht garantiert dass ein Nutzer dadurch immer die aktuellste Version enthält. Caching ist darüber hinaus nicht anwendbar für Daten, die mit TLS oder IPsec verschlüsselt wurden, da diese für zwischengeschaltete Systeme nicht lesbar sind.

2.3.1.6. Probleme evolutionärer Ansätze

Das Internet ist bis jetzt erfolgreich mit evolutionären Ansätzen so weiterentwickelt worden, dass es den aktuellen Anforderungen noch standhalten kann. Dabei wurden jedoch viele Erweiterungen eingebaut, die jeweils eine Lösung für ein spezielles Problem erbrachten. Leider behindern sich viele der hinzugefügten Erweiterungen gegenseitig, schränken die neuen Funktionen ein oder schließen sich sogar vollkommen aus. In Tabelle 2.1 werden die vorgestellten evolutionären Lösungen bezüglich ihrer Kompatibilität untereinander verglichen. Dabei wird sichtbar, welche Erweiterungen miteinander verwendet werden können (✓), bei welchen eine Kombination nicht sinnvoll ist, da sie ähnliche Probleme lösen (–), und welche nicht miteinander kombiniert werden können (×). Besonders NAT und Caching, die eine großen Teil zur Skalierungsfähigkeit der aktuellen Internet-Architektur beitragen, behindern andere Erweiterungen.

Tabelle 2.1.: Kompatibilität evolutionärer Lösungen zueinander

	NAT	IPsec	TLS	IPv6	Mobile IP	Caching
NAT		×	✓	–	×	✓
IPsec	×		–	✓	✓	×
TLS	✓	–		✓	✓	×
IPv6	–	✓	✓		✓	✓
Mobile IP	×	✓	✓	✓		✓
Caching	✓	×	×	✓	✓	

2.3.2. Revolutionäre Ansätze

Aus den bisher dargestellten Schwachstellen lässt sich die Forderung ableiten, die Internet-Architektur ausschließlich unter Berücksichtigung der neuen Bedürfnisse von Grund auf neu zu entwerfen. Man spricht daher von einem revolutionären Ansatz. Dieser gestattet es bewusst, eine neue Architektur zu entwerfen, die nicht auf der alten basiert und auch nur eingeschränkt damit kompatibel ist.

2.3.2.1. Planungsziele für ein Internet der Zukunft

Die Planungsziele für eine neue Internet-Architektur lassen sich dabei nicht mehr in eine nach Priorität sortierte Liste bringen, da sich die Prioritäten je nach Anwendungsfall verschieben können. Zu den bisherigen Planungszielen, die nach wie vor ihre Berechtigung besitzen, kommen folgende neue Planungsziele für eine zukünftige Internet-Architektur hinzu [BCSW00]:

- **Skalierbarkeit**: Das Internet muss in der Lage sein, Billionen von Endgeräten zu adressieren, wobei die Gerätelandschaft höchst heterogen sein wird. Das Ende-zu-Ende-Prinzip soll dabei gewahrt werden.
- **Mobilität**: Die Mobilität von Endgeräten ohne Trennung der aktuell laufenden Kommunikation muss sichergestellt sein.
- **Sicherheit**: Das Internet muss integrierte Dienste für Informationssicherheit anbieten, die bei Bedarf ohne Einschränkung und für jede Kommunikation verwendet werden können.
- **Unterstützung für Inhalte**: Das Internet muss nicht nur die Möglichkeit bieten, Geräte zu adressieren, sondern auch in der Lage sein, Informationen und Inhalte, unabhängig von deren Speicherort, zu adressieren.

Eine neue Architektur, die auf diesen zusätzlichen Zielen basiert, muss so flexibel gestaltet sein, dass sich diese Ziele gegenseitig in keinster Weise behindern, sondern gegenseitig ergänzen und unterstützen.

2.3.2.2. Umsetzung von revolutionären Ansätzen

Die Herangehensweise für revolutionäre Internet-Architekturen unterscheidet sich maßgeblich von der von evolutionären Erweiterungen. So ist das Grundgerüst, auf dem die neue Architektur aufbaut, noch nicht bekannt. Daher müssen zunächst rein theoretische Überlegungen angestellt werden, wie alle Planungsziele in einer neuen Architektur vereinigt werden können. Diese Überlegungen müssen jedoch auch zwingend als Prototyp in einer Testumgebung überprüft werden, um die Machbarkeit zu beweisen sowie um zeigen zu können, dass die neue Architektur tatsächlich notwendige Verbesserungen mit sich bringt.

Der Einsatz einer Testplattform, mit der eine neue Architektur sowohl unter idealen, als auch unter realen Bedingungen getestet werden kann, ist daher unabdingbar. Aus den gewonnenen Ergebnissen der Prototypen kann dann das Konzept der Architektur erneut verfeinert werden, und so letztendlich zu einem soliden Vorschlag für eine zukünftige Internet-Architektur heranreifen.

Als Ergebnis eines revolutionären Ansatzes muss jedoch nicht zwingend eine vollständig neue Internet-Architektur stehen. Es ist durchaus auch möglich, Teile der bestehenden Architektur zu beizubehalten, Teile zu streichen und neue hinzuzufügen. Als ein Beispiel für einen revolutionären Ansatz, der letztendlich doch evolutionär umgesetzt wurde, ist IPv6. Obwohl IPv6 nicht alle aktuellen Probleme lösen kann und nicht alle Planungsziele

erreicht, werden dadurch einige Probleme deutlich entschärft und die Kompatibilität mit der aktuellen Architektur ist gegeben.

2.4. Aktuelle Arbeiten zum Internet der Zukunft

Die Internet-Architektur wird kontinuierlich weiter entwickelt. Neben den bereits vorgestellten und existierenden Erweiterungen werden in diesem Abschnitt einige Konzepte vorgestellt, die aktuell in der Entwicklungsphase sind und sowohl den evolutionären als auch den revolutionären Ansätzen zugeordnet werden können. Zunächst wird das Konzept der Trennung von Locator und Identifier behandelt, auf dem mehrere Architekturvorschläge aufbauen. Anschließend werden Konzepte für ein damit verbundenes Mapping-System vorgestellt, bevor zum Schluss Konzepte eines inhaltsorientierten Internets vorgestellt werden.

2.4.1. Konzepte zur Trennung von Locator und Identifier

Als eine der Hauptursachen für die Probleme der aktuellen Internet-Architektur bezüglich Skalierbarkeit des Routingsystems und der mangelnden Unterstützung für Mobilität wird in [MZF07] die überladene Semantik der IP-Adresse verantwortlich gemacht. Sie verbindet eine global eindeutige Kennziffer eines Gerätes (Identifier) mit dessen Ort (Locator). Router verwenden darüber hinaus den Netzteil der IP-Adresse und dessen hierarchischen Aufbau, um Datenpakete durch das Internet zu leiten. Dazu stellte Yakov Rechter folgende These auf: *„Addressing can follow topology or topology can follow addressing. Choose one.“* Für ein skalierbares Routing muss daher entweder das Adressierungsschema der Topologie angepasst werden, oder die Topologie der Adressierung. Da IP-Adressen aber Organisationen, Betreiben und Firmen zugeordnet werden, unabhängig von der darunter liegenden hierarchischen Topologie, ist es nach [MZF07] schwierig bis unmöglich, Stabilität und Skalierbarkeit zu erreichen.

Als Folgerung aus den Problemen, die durch die semantische Überladung der IP-Adresse entstanden sind, fordern Standardisierungsgremien wie die IETF sowie das IAB eine Trennung des Adressraums in zwei voneinander völlig unabhängige Adressen. Dieses Konzept wird *Locator/Identifier Separation* genannt. So wird der Identifier ausschließlich als eindeutige Kennziffer für Geräte verwendet, der Locator dagegen bestimmt den Aufenthaltsort des Gerätes und kann sich vollständig an die Topologie des Netzes anpassen, was zu deutlich kleineren Routingtabellen führt. Die Autoren von [QIdLB07] konnten in Simulationen nachweisen, dass sich die BGP-Routingtabellen in der DFZ, unter der Anwendung der Trennung von Locator und Identifier, deutlich reduzieren lassen. Die Anzahl von Einträgen in den BGP-Routingtabellen kann damit, im Vergleich zu den vergeben Adressblöcken, um Größenordnungen verringert werden. Aktuell ist das Gegenteil der Fall, wie Abbildung 2.2 zeigt.

Anforderungen wie Multihoming tragen folglich nicht mehr zum Wachstum der Routingtabellen bei, da es möglich ist jedem Identifier mehrere Locatorwerte zuzuordnen. Durch zusätzliches Einführen von Metriken für die Locatorwerte, die einem Identifier zugeordnet sind, kann zudem der Verkehr gezielt gesteuert werden. Der Routingtabel-

lenzuwachs durch Traffic Engineering kann damit ebenfalls reduziert werden. Darüber hinaus wird Mobilität durch die Trennung von Locator und Identifier inhärent unterstützt, da der einem Gerät zugewiesene Identifier gleich bleibt, wenn das Gerät seinen Netzzugangspunkt ändert. Lediglich der Locator ändert sich, was bei der Kommunikation zwar berücksichtigt werden muss, für die Anwendungen aber transparent geschieht.

Die Umsetzung einer Trennung von Locator und Identifier kann auf zwei verschiedene Arten erfolgen, die sich vor allem in ihrer Kompatibilität mit der aktuellen Internet-Architektur unterscheiden. Diese werden im Folgenden vorgestellt, zusammen mit jeweils einem konkreten Konzept als Beispiel.

2.4.1.1. Transparente Ansätze

Bei einem transparenten Ansatz zur Trennung von Locator und Identifier sind sich die Endknoten dieser semantischen Trennung nicht bewusst. Aus Endgerätesicht existiert nach wie vor nur eine Adresse, üblicherweise die IP-Adresse, welche die Aufgaben von Locator und Identifier übernimmt. Lediglich die BGR an den Grenzen der AS setzen dieses Konzept um. Vorteile ergeben sich daraus bezüglich der Kompatibilität, da aus der Sicht des Endgeräts keinerlei Änderung erfolgen muss. Ebenso muss das Routing innerhalb der AS nicht verändert werden. Der Problematik der Routingtabellengröße in der DFZ kann damit bis zu einem gewissen Punkt erfolgreich entgegengewirkt werden. Vorteile bezüglich Mobilität ergeben sich durch einen transparenten Ansatz jedoch nicht, da die Trennung von Locator und Identifier nur im Kernnetz vorgenommen wird, aber nicht an den eigentlichen Kommunikationsendpunkten.

LISP Ein konkreter Vorschlag eines transparenten Ansatzes zur Trennung von Locator und Identifier ist das Locator/Identifier Separation Protocol (LISP) [Mey08, FFML09b] der Firma Cisco. Es handelt sich dabei um ein einfaches IP-in-IP Tunneling Protokoll mit dem Ziel, als inkrementelle Erweiterung zur aktuellen Internet-Architektur eingeführt werden zu können und dennoch die Vorteile bezüglich Reduktion der Routingtabellengrößen zu erreichen. Dadurch, dass ausschließlich Modifikationen an den BGR notwendig sind, ist LISP noch den evolutionären Ansätzen zuzuordnen. Für die Adressierung verwendet LISP so genannte Endpoint Identifier (EID) und Routing Locator (RLOC), die jeweils gewöhnlichen IP-Adressen entsprechen. EID werden dabei allen Endgeräten zugeteilt, der RLOC ist ausschließlich für Tunnel-Router vorgesehen, deren Aufgabe die BGR übernehmen. Innerhalb eines LISP-AS erfolgt das Routing direkt mit den EID. Soll ein Paket an einen EID gesendet werden, die sich nicht im gleichen LISP-AS befindet, wird das Paket an einen BGR weitergeleitet. Dieser fungiert als so genannter Ingress Tunnel Router (ITR) und leitet das Paket eingekapselt in einem anderen IP-Paket an den Egress Tunnel Router (ETR) im Ziel-AS. Zwischen den BGR erfolgt das Routing mithilfe des RLOC. Den zu einem EID gehörigen RLOC erfragen die BGR aus einem Datenbanksystem. Im Falle von Multihoming können dabei für einen EID auch mehrere RLOC existieren. Abbildung 2.5 zeigt das LISP Adressierungsschema.

Durch die ausschließliche Verwendung von RLOC im Inter-Domain-Bereich wird der zu routende Adressraum deutlich reduziert, was die Tabellengröße der BGR verkleinert. Für LISP existieren verschiedene Versionsstufen, die sich vor allem in dem verwende-

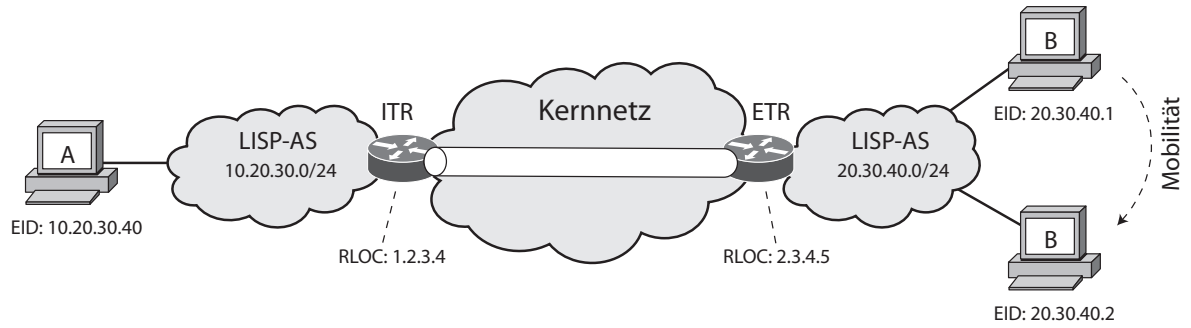


Abbildung 2.5.: Transparente Trennung von Locator und Identifier mit LISP

ten Datenbanksystem, auch Mapping-System genannt, unterscheiden. Diese werden im Folgenden Abschnitt vorgestellt. Für die LISP-Version 1 ist zudem der EID auch im Inter-Domain-Bereich noch routingfähig. Erst in LISP 2 und LISP 3 wird dafür ausschließlich der RLOC verwendet. Der EID ist ab dann nur noch lokal routingfähig. Die einzelnen Versionen unterstützen dabei die inkrementelle Einführung von LISP, da immer nur kleinere Änderungen umgesetzt werden müssen.

Bezüglich Mobilität bietet LISP jedoch keine Vorteile, da der EID nicht vollständig von Routingaufgaben losgelöst ist. Im Fall von Mobilität wird dem Gerät ein neuer EID zugeordnet (vgl. Abbildung 2.5) und eine bestehende Datenübertragung unterbrochen. Darüber hinaus ist die zu erwartende Last auf die Tunnel-Router sehr hoch, da jedes zu kapselnde Paket initial die dafür nötige Information aus dem Mapping-System beschaffen muss.

2.4.1.2. Endknotenbewusster Ansatz

Eine weitere Umsetzungsmethode zur Trennung von Locator und Identifier setzt darauf auf, dass sich die Endgeräte dieser Trennung bewusst sind. Jedes Gerät besitzt einen eindeutigen Identifier und einen Locator pro verbundener Netzschnittstelle. Die logische Kommunikation erfolgt ausschließlich auf der Basis des Identifiers. Im gesamten Internet sind jedoch lediglich die Locatoren routingfähig und jedes Gerät muss vor der Kommunikation mit einem anderen Gerät selbstständig den aktuell gültigen Locator zu einem bestimmten Identifier mithilfe des Mapping-Systems auflösen. Derartige Ansätze können das Problem der wachsenden BGP-Routingtabellen erfolgreich lösen und bieten durch die konsequente Trennung von Locator und Identifier perfekte Voraussetzungen für Mobilität. Nachteil dieser Ansätze sind notwendige Modifikationen im Netzwerkstack sowie auf Anwendungsebene bei allen Endgeräten. Endknotenbewusste Ansätze zählen daher immer zu der Gruppe der revolutionären Ansätze. Abbildung 2.6 zeigt die Vorteile dieses Ansatzes bezüglich Mobilität. Die Kommunikation erfolgt ausschließlich über Identifier, ein Wechsel des Locators wird vom Netzwerk-Stack verarbeitet und erfolgt für die Anwendungen transparent.

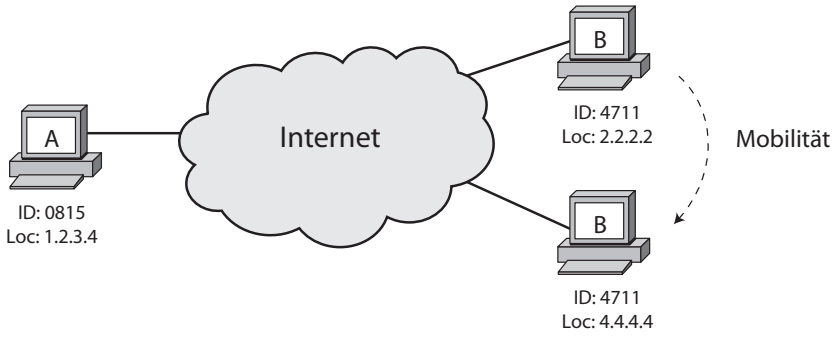


Abbildung 2.6.: Endknotenbewusste Trennung von Locator und Identifier

HIP Ein Vorschlag aus diesem Bereich ist das so genannte Host Identity Protocol (HIP) [RFC06]. HIP verwendet für den Locator weiterhin IP-Adressen, für Applikationen wird die IP-Adresse jedoch durch den Host Identity Tag (HIT) ersetzt, das als Identifier dient. Da das Routing weiterhin auf IP-Adressen basiert, sind für HIP keine speziellen Gateways oder Modifikationen in der Vermittlungsschicht erforderlich.

Besonderen Fokus legt HIP auf die Bereitstellung von Sicherheitsmechanismen, die inhärent in der Architektur verankert werden. Der Host Identifier (HI) ist der öffentliche Schlüssel eines asymmetrischen Schlüsselpaares, wodurch Integrität, Authentizität und Verschlüsselung mittels PKI möglich sind. Der HI als eigentlicher Identifier wird jedoch nicht in den Packet-Headern verwendet, da seine Länge je nach verwendetem Algorithmus variieren kann. Aus diesem Grund wird das HIT für die Identifikation des Gerätes verwendet, das mittels Hashfunktion aus dem HI berechnet wird und eine feste Länge von 128 Bit besitzt.

Mit HIP kann eine deutliche Reduktion der BGP-Routingtabellen erreicht werden, wenn sämtliche IP-Adressen entsprechend der Topologie hierarchisch neu vergeben werden, was jedoch mit einem deutlichen Aufwand verbunden ist. Neben den nötigen Modifikationen des Netzwerkstacks und den Schnittstellen zu den Anwendungen besitzt HIP den Nachteil, dass der HIT fest an den HI gebunden ist. Sollte das Schlüsselpaar aufgrund einer Kompromittierung getauscht werden müssen, ändert sich auch der als Identifier verwendete HIT. Bezüglich Migrationsfähigkeit ist HIP zu vergleichen mit IPv6, das ebenfalls Modifikationen des Netzwerk-Stacks und der Anwendungsebene erfordert.

2.4.1.3. Weitere Konzepte

Zu den jeweiligen Ansätzen existieren noch mehr konkrete Vorschläge. Das Core Router Integrated Overlay (CRIO) verwendet ebenfalls einen transparenten Ansatz und erreicht durch die Ankündigung von Super-Präfixen in der DFZ sehr kleine Routingtabellen. Nachteil sind jedoch die längeren Pfade durch das Netz aufgrund der starken Aggregierung.

Weitere Vertreter aus der Gruppe der endknotenbewussten Ansätze sind Himalis [KI10], Hair [FCM+09], GLI-Split [MHK10] und MILSA [PPJB08]. Diese Ansätze verwenden den Identifier ausschließlich zur Geräteidentifikation und den Locator nur für das Routing.

Die Ansätze unterscheiden sich besonders im Aufbau des Identifiers bzw. Locators und verwenden zum Teil deutlich verschiedene Konzepte für das Mapping-System.

2.4.2. Mapping-Konzepte für Systeme mit Locator/Identifier-Trennung

Ein unverzichtbarer Bestandteil jeder Architektur, die auf der Trennung von Locator und Identifier basiert, ist das so genannte Mapping-System. Es ist dafür zuständig, den zu einem bestimmten Identifier aktuell gültigen Locator in einem Mappingeintrag zu speichern und anfragenden Geräten zur Verfügung zu stellen. Pro Mappingeintrag können dabei auch mehrere Locator-Werte gespeichert werden, um Multihoming zu realisieren.

Abhängig vom gewählten Ansatz zur Realisierung der Trennung von Locator und Identifier sind die Anforderungen an das Mapping-System sehr unterschiedlich. Für den endknotenbewussten Ansatz sind die Anforderungen bezüglich Geschwindigkeit zweitrangig, da bereits das Endgerät den korrekten Locator auflösen muss und damit das Paket geroutet werden kann. Der transparente Ansatz jedoch verlangt nach einem extrem schnellen System, da die Mappinganfragen von Routern auf dem Übertragungsweg aufgelöst werden müssen. Diese müssen Pakete so lange zwischenspeichern, bis die Antwort aus dem Mapping-System eingetroffen ist, ansonsten werden Pakete verworfen.

Zur Umsetzung eines Mapping-Systems existieren je nach Anforderung verschiedene Möglichkeiten, die sich in vier Kategorien klassifizieren lassen [MHH10]. Die Lösung mit einer einzelnen zentralen Datenbank, auch Direct Map Database (DMB) genannt, ist aus Skalierungsgründen für ein Mapping-System, das alle globalen Anraten bearbeiten muss, nicht praktikabel. Sie wird an dieser Stelle jedoch trotzdem erwähnt, da einige Lösungen bei ihren Konzepten auf eine DMB als Teilsystem zurückgreifen. Alle vier Kategorien sind in Abbildung 2.7 dargestellt.

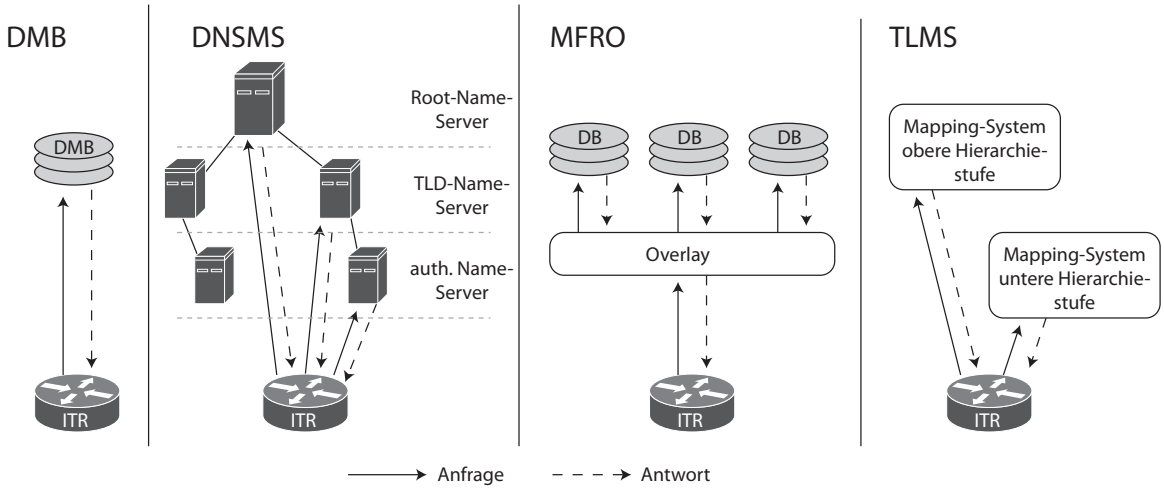


Abbildung 2.7.: Klassifikation von Mapping-Systemen

2.4.2.1. Domain Name System based Mapping System (DNSMS)

Das Domain Name System (DNS) existiert bereits seit 1983 und wurde von Mockapetris in [Moc87a, Moc87b] eingeführt. Es stellt einen verteilten Verzeichnisdienst dar, um Gerätenamen, die deutlich einprägsamer sind als IP-Adressen, in diese aufzulösen. Der Gerätename ist dabei, genau wie die Topologie des DNS, hierarchisch baumförmig aufgebaut. Ausgehend von der Wurzel verzweigt sich die Struktur zunächst zu den Top Level Domains (TLD), dann zu den Second Level Domains. Anschließend folgen beliebig viele Subdomains. Ein vollständiger Gerätename setzt sich dabei aus seinem eigenen Namen inklusive aller Domänennamen zusammen und wird als Full Qualified Domain Name (FQDN) bezeichnet.

Jede Domäne besitzt einen so genannten autoritativen DNS-Server, der für die Namensauflösung der in dieser Domäne registrierten Geräte zuständig ist. Anfragen von Geräten werden zunächst an einen DNS-Resolver geschickt, der die Anfragen dann direkt an den autoritativen DNS-Server sendet, falls ihm dieser bekannt ist. Falls nicht, werden die Anfragen in der Hierarchie ganz nach oben zur Wurzel gesendet (DNS-Root-Server). Der DNS-Root-Server hat Kenntnis über alle Top Level Domain-Server und diese wiederum über alle autoritativen Server. Die Anfrage wird damit entsprechend der Hierarchie von oben nach unten bearbeitet. Da der DNS-Root-Server einen einzelnen Ausfallpunkt darstellt und darüber hinaus eine hohe Anfragelast verarbeiten muss, ist dieser auf insgesamt 13 Server (Root-Server „A“ bis „M“) weltweit gespiegelt. Auf den einzelnen Resolvern wird Caching verwendet, um die Last auf die Root-Server abzufedern. Dem Caching verdankt DNS seine aktuelle Leistungsfähigkeit, da die Root-Server ansonsten hoffnungslos überlastet wären.

Das DNS kann jedoch auch dazu verwendet werden, Identifier in die entsprechenden Locator-Wertes aufzulösen. Die Aussage, die aktuelle Internet-Architektur stellt mit DNS bereits eine Locator/Identifier-getrennte Lösung dar, ist jedoch so nicht zulässig, da der Gerätename ausschließlich in den anwendungsorientierten Schichten verwendet wird. Auf den netzorientierten Schichten kommt nur noch die IP-Adresse zum Einsatz. Eine Zuordnung zwischen Gerätenamen und IP-Adresse damit nicht mehr möglich ist.

Um ein DNSMS als Mapping-System zu verwenden, sind jedoch Identifier nötig, die der gleichen hierarchischen Struktur folgen wie die Topologie des DNS. Kommen flache Identifier ohne Hierarchie zum Einsatz, ist ein DNSMS nur mit sehr hohem administrativen Aufwand zu realisieren.

Verschiedene Konzepte setzen auf einem DNSMS auf, um die Einführung eines komplett neuen Datenbanksystems zu vermeiden. LISP verwendet in Version 2, in der EID bereits nicht mehr im Kernnetz routingfähig sind, ein DNSMS. Ebenso verwendet HIP in seiner Standardversion ein DNSMS trotz flacher Identifier. HIP sieht diesbezüglich vor, in den DNS-Root-Servern für jedes existierende HIT einen eigenen Eintrag vorzuhalten, der auf die entsprechende Second-Level-Domain verweist, die letztendlich das HIT in den Locator auflösen kann. Da in diesem Fall jede Mappinganfrage zunächst an einen DNS-Root-Server geleitet wird, ist die dadurch erzeugte Last auf diese extrem hoch, obwohl diese die Anfrage nur delegieren müssen. Dieser Ansatz wird daher als nur schwer umsetzbar eingestuft.

2.4.2.2. Map-Request Forwarding Overlay (MRFO)

Ein auf einem MRFO basierenden Mapping-System besteht aus einer verteilten Datenbank, bei der jedes teilnehmende Gerät nur für einen bestimmten Teil der zu speichernden Mappingeinträge verantwortlich ist. Ein MRFO sind üblicherweise als Distributed Hash Table (DHT) realisiert, die den Identifier als Indexschlüssel für die Mappingeinträge verwenden und die ein eigenes Indexschema nutzen, um die Datenbank auf die einzelnen Geräte optimal zu verteilen. Strukturierte Peer-to-Peer Protokolle wie Chord [SMK+01] können zum Aufbau einer DHT verwendet werden. Ein MRFO hat den Vorteil, dass es keinen zentralen Ausfallpunkt besitzt. Fällt ein Teil des Overlays aus, sind nur die darauf gespeicherten Datensätze betroffen. Durch Mechanismen wie Replikation kann auch diese Ausfallwahrscheinlichkeit minimiert werden. DHTs sind darüber hinaus in hohem Maße skalierungsfähig, da das System einfach durch das Hinzufügen neuer Geräte erweitert werden kann und die Konfiguration automatisch vom verwendeten Protokoll übernommen wird.

Abhängig vom verwendeten Protokoll zum Aufbau und Nachschlagen in der DHT kann ein MRFO auch für transparente Ansätze verwendet werden, bei denen Router auf dem Übertragungsweg das Mapping durchführen und daher besonders kurze Antwortzeiten benötigen. LISP Version 3 setzt auf ein eigenständiges Mappingsystem unabhängig von DNS. Als mögliche Variante eines MRFO kommen dabei die Konzepte LISP+ALT (Alternative Topology) [FFML09a] und LISP-DHT [MI08] zum Einsatz. LISP+ALT setzt auf ein eigenes Mapping-Overlay, das noch nicht als DHT realisiert ist sondern eine hierarchische Struktur besitzt. Da die EID ebenfalls hierarchisch aufgebaut ist, wird LISP+ALT ähnlich wie DNS aufgebaut. Dabei ist LISP+ALT aber deutlich leistungsfähiger, da ausschließlich Router und keine Endgeräte das Mapping-System verwenden. In der Version LISP-DHT wird das Peer-to-Peer Protokoll Chord zum Aufbau einer DHT verwendet, in der die Mappingeinträge gespeichert werden. Dabei profitiert LISP-DHT von den Skalierungseigenschaften einer DHT. Auch das Mapping-System HIP-DHT [Ahr07] für HIP verwendet eine DHT basierend auf dem OpenDHT-Projekt und soll DNS als primäres Mapping-System ablösen.

2.4.2.3. Two-Level Mapping System (TLMS)

Ein TLMS besitzt zwei getrennte Datenbankstufen zum Auflösen von Mappinganfragen, die hierarchisch angeordnet sind. Jede Stufe kann dabei als DMB oder als MRFO umgesetzt werden. Von den oberen Hierarchiestufen existiert üblicherweise nur eine Instanz, von der unteren Stufe beliebig viele. Vorraussetzung ist dabei jedoch ein Identifier mit mindestens zwei Hierarchiestufen, beziehungsweise einem flachen Identifier und einem Präfix. Die Anfrage an ein TLMS ist in zwei Schritte unterteilt. Zunächst wird die Anfrage an die obere Hierarchiestufe gestellt. Anhand des Identifier-Präfixes oder des hierarchischen Aufbaus wird zurückgeliefert, welche untere Hierarchiestufe letztendlich den Mappingeintrag des Identifiers speichert. Die zweite Anfrage geht direkt an diese Instanz der unteren Hierarchie. Durch Caching der Präfixe mit zugehörigem Mapping-System der unteren Stufe kann die obere Stufe weitgehend entlastet und ein schnelles Auflösen von Mappinganfragen garantiert werden.

In DHT-MAP [LQZ09] wird ein TLMS für LISP Version 3 vorgestellt. Jedes AS betreibt in diesem Fall eine DMB und stellt jeweils eine Instanz der unteren Hierarchiestufe dar. Dabei ist jedes AS für die Mappingeinträge seiner zugeteilten EID zuständig. Als obere Hierarchiestufe kommt ein auf dem strukturierten Peer-to-Peer Protokoll CAN [RFH+01] basiertes MRFO zum Einsatz.

Die Leistungsfähigkeit und Skalierbarkeit einer zukünftigen Internet-Architektur, die auf der Trennung von Locator und Identifier aufbaut, wird maßgeblich von dem verwendeten Mapping-System bestimmt. Dessen Leistung bestimmt letztendlich, wie schnell Pakete zum Ziel weitergeleitet werden. Die Verwendung von verteilten Datenbanken mit DHT stellt besonders aufgrund ihrer Skalierbarkeit und Erweiterbarkeit eine vielversprechende Lösung für ein Mapping-System des zukünftigen Internets dar.

2.4.3. Inhaltsorientierte Netze

Jacobson nennt in [JST+09] drei Generationen in der Entwicklung von Kommunikationsnetzen. Die erste Generation war das leitungsvermittelte Telefonnetz, das einzelne Leitungen mit einer Telefonnummer adressiert, um einen Teilnehmer zu erreichen. Die zweite Generation ist das heutige paketvermittelte Internet, das Endgeräte adressieren kann, was aktuell durch die IP-Adresse geschieht. Die nächste Generation ist laut Jacobson ein Netz, das Informationen und Inhalte sowie deren Zustellung und Verteilung in den Vordergrund stellt.

Bereits mit der Einführung des WWW wurde ein Schritt in diese Richtung unternommen. Mit einem URI ist es möglich, einen bestimmten Inhalt im Netz zu adressieren und diesen abzurufen. Dennoch basiert das WWW auf einer Internet-Architektur, die ausschließlich das Adressieren von Endgeräten unterstützt. Ein bestimmter Inhalt kann deshalb immer nur über die Adresse des Gerätes erreicht werden, das den Inhalt speichert. Für den Benutzer ist der Speicherort eines Inhalts jedoch nicht von Belang, lediglich die Aktualität des Inhalts und der verzögerungsfreie Abruf stehen im Vordergrund. Nicht mehr das „Wo“ steht damit im Vordergrund, sondern vielmehr das „Was“. Dabei muss sichergestellt werden, dass einem bestimmten Inhalt eine Kennung bzw. ein Identifier zugewiesen wird, der sich über die Lebensdauer des Inhalts nicht ändert und dem Benutzer hochverfügbar und mit kurzen Verzögerungszeiten angeboten werden kann. Zudem muss es dem Benutzer möglich sein, die Authentizität der Inhalte zu überprüfen, unabhängig vom Speicherort.

2.4.3.1. Digital Object Identifier (DOI)

Die Einführung eines digitalen Identifier für Objekte (DOI) jeglicher Art stellt zwar noch kein neues Konzept eines inhaltsorientierten Netzes dar, bildet aber eine entscheidende Grundlage dafür und wird deshalb an dieser Stelle vorgestellt. Das Konzept von der International DOI Foundation (IDF) in [Pas08] basiert auf dem informatischen Konzept so genannter Handles[3]. Ein DOI weist Objekten jeglicher Art eine eindeutige Kennung

[3]Ein Handle stellt in der Informatik einen eindeutigen Schlüssel für digitale Objekte dar. Diese sind aber, im Gegensatz zu Zeigern, vom Speicherort der Objekte entkoppelt.

zu, die eine unbegrenzte Gültigkeit besitzt und unabhängig vom Speicherort der Objekte ist. Dies bezieht sich auch ausdrücklich auf nichtdigitale Objekte wie Druckmedien, denen ebenfalls ein DOI zugewiesen werden kann.

Ein DOI ist hierarchisch aufgebaut und besteht aus einem Präfix und einem Suffix, die durch einen Schrägstrich voneinander getrennt sind. Ein Beispiel für einen DOI lautet: *10.1000/12345*. Dabei ist 10.1000 das Präfix und 12345 das Suffix.

DOI-Präfix Per Konvention beginnt jeder DOI mit der Nummer 10, gefolgt von einem Punkt. Anschließend folgt die einer Organisation zugeteilte Nummer. Eine Organisation, die DOI für Objekte erstellen möchte, kann unbegrenzt viele Präfixe erhalten, womit es beispielsweise für einen Verlag möglich ist, jeder Zeitschriftenserie ein eigenes Präfix zuzuweisen. Um die Einzigartigkeit von DOI sicherzustellen, werden die Präfixe von so genannten Registration Agencies (RA) vergeben. Die Länge des Präfix ist eine vierstellige Nummer. Es ist darüber hinaus möglich, Sub-Präfixe zu verwenden, die ebenfalls mittels Punkt getrennt werden.

DOI-Suffix Der zweite Teil eines DOI dient dazu, ein Objekt aus dem für das Präfix definierten Bereich eindeutig zu identifizieren. Das Suffix hat keine global eindeutig definierte Struktur, kann aber bereits existierende Strukturen, wie beispielsweise die ISBN-Nummer für Bücher, verwenden. Es darf aus einer beliebigen Anzahl alphanumerischen Zeichen bestehen.

Der DOI dient lediglich als Referenz auf den Speicher- oder Lagerort eines gewissen Objekts. Die Zuordnung zwischen DOI und Objekt erfolgt mittels einer Datenbank, die weitere Informationen über die Erreichbarkeit des Objekts enthält. Für digitale Objekte kann dies eine URL sein, für Bücher und Druckmedien Informationen über den herausgebenden Verlag oder eine Bücherei. Mittels der Webseite `http://dx.doi.org` können diese Informationen abgerufen werden.

Ändert sich der Speicherort eines bestimmten Objekts, so muss lediglich der Eintrag in der DOI-Datenbank aktualisiert werden. Der DOI für das Objekt selbst behält weiterhin seine Gültigkeit und erfüllt damit eine wichtige Forderung eines inhaltsorientierten Netzes: Jedem Objekt wird eine Kennung unabhängig seines Speicherortes zugewiesen.

2.4.3.2. Content Distribution Network (CDN)

Ein Konzept, die Adressierung von Inhalten zumindest auf den anwendungsorientierten Schichten unabhängig vom Speicherort zu gestalten und dabei die Zustellung der Inhalte hin zum Benutzer auf Geschwindigkeit zu optimieren, ist das so genannte CDN. Ein CDN wird von einem so genannten Content-Provider betrieben, dessen Kunden die Auslieferung von Informationen an die Nutzer möglichst schnell und verzögerungsfrei gestalten möchten. Dies soll darüber hinaus für die Konsumenten völlig transparent geschehen, so dass auf deren Seite keinerlei Modifikation nötig ist. Der Content Provider baut dazu ein eigenes Netz aus Replika-Servern, das die zur Verfügung gestellten Inhalte von eine Ursprungsserver des Betreibers abholt. Angefragte Inhalte werden dann anhand bestimmter Metriken von dem jeweils optimalen Server an den Konsumenten ausgeliefert.

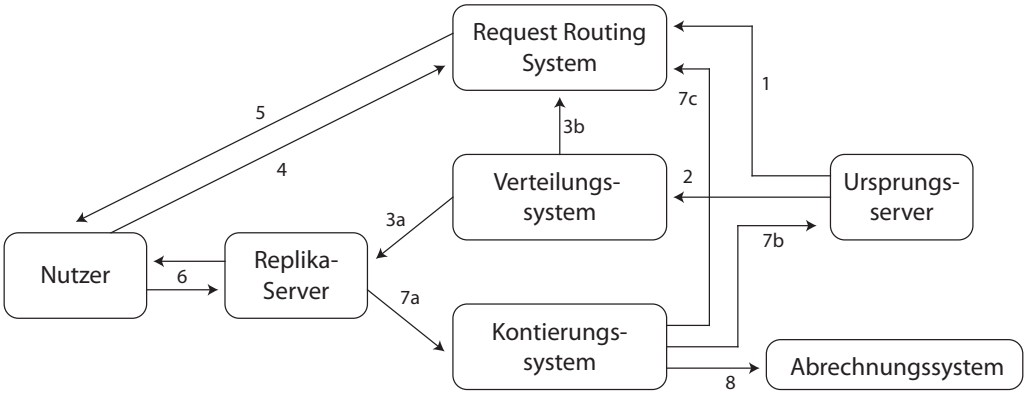


Abbildung 2.8.: Systemarchitektur und Komponenten eines CDN

Dabei spielen beispielsweise die örtliche Lage, die Distanz in AS Hops, niedrige Latenz- und Zugriffszeiten sowie eine ausreichende Datenrate eine Rolle. Zum Marktführer in dieser Branche gehört das Unternehmen Akamai [Aka].

Um dies zu bewerkstelligen muss das CDN in der Lage sein, auf eventuelle auftretende Störungen reagieren zu können um die Metriken neu zu berechnen. Darüber hinaus muss auch sichergestellt werden, dass der Kunde, der seine Inhalte über ein CDN verteilen möchte, diese jederzeit ändern kann und die neuen Inhalte anschließend auf die Replika-Server kopiert werden. Dabei ist, wie im Falle von Caching, besonders auf die Kohärenz der Information zu achten.

Komponenten eines CDN Die Systemarchitektur und die einzelnen Komponenten eines CDN nach [Pen08] sind in Abbildung 2.8 dargestellt und werden im Folgenden näher erläutert.

- **Replika-Server**: Dieser Server speichert Kopien der Inhalte, die zur Verfügung gestellt werden sollen. Ein Replika-Server wird am Rand des Netzes, nahe des Nutzer platziert.

- **Request Routing-System**: Dieses System wählt zu jeder Anfrage den passenden Replika-Server aus und leitet die Anfrage an diesen weiter.

- **Verteilungssystem**: Dieses System ist für die Verteilung der Inhalte auf die einzelnen Replika-Server zuständig. Dies geschieht entweder direkt über das Internet oder, wenn der Inhalt echtzeitkritisch ist, auch über Satellitenverbindungen.

- **Ursprungsserver**: Dieser Server stellt die Inhalte, die über das CDN verteilt werden sollen, initial zur Verfügung. Der Ursprungs-Server wird dabei nicht vom CDN-Betreiber betrieben, sondern vom Kunden selbst administriert.

- **Kontierungssystem**: Dieses System fasst Statistiken über die abgerufenen Inhalte zusammen. Diese werden an das Distribution System zur Optimierung der

Inhaltsverteilung sowie an das Abrechnungssystem und an den Ursprungs-Server weitergegeben.

- **Abrechnungssystem**: Dieses System berechnet die entstandenen Kosten für die jeweils zur Verfügung gestellten Inhalte und stellt sie den Kunden in Rechnung.

Verteilung und Zustellung von Inhalten Damit die Inhalte eines Kunden über ein CDN zur Verfügung gestellt werden, muss der URI des Inhalts an das CDN übergeben werden (1). Dabei gilt zu beachten, dass die DNS-Einträge durch das CDN so modifiziert werden, dass der URI des Inhalts, der auch einen Gerätenamen enthält, jetzt nicht mehr auf den Ursprungsserver zeigt, sondern auf ein Gerät aus dem Request Routing System des CDN. Der Inhalt wird anschließend vom Ursprungsserver an das Verteilungssystem übergeben (2), das den Inhalt an die Replika-Server verteilt (3a) und das Request Routing System über die einzelnen Kopien informiert (3b).

Stellt ein Benutzer eine Anfrage mittels URI, wird diese direkt an das Request Routing System geleitet (4), das die Anfrage zu einem geeigneten Replika-Server weiterleitet (5). Der Benutzer verbindet sich dann direkt zu dem zurückgelieferten Replika-Server und ruft die Daten ab (6). Der Replika-Server informiert anschließend das Kontierungssystem (7a), das wiederum den Ursprungsserver (7b) sowie das Request Routing System (7c) benachrichtigt. Aus den gesammelten Informationen errechnet das Abrechnungssystem die Kosten für den Kunden (8).

Request Routing System Nach einer erhaltenen Anfrage muss das Request Routing System zunächst den passenden Replika-Server auswählen. Als Metrik können Laufzeiten und Hops zwischen anfragendem Gerät und Request Routing System verwendet werden, um die Position des Benutzers abzuschätzen. Da diese Methode meist kein zufriedenstellendes Ergebnis liefert und sehr viele andere Einflüsse die Schätzung beeinflussen können, wird zudem noch die Last der Replika-Server berücksichtigt.

Nachdem der passende Replika-Server bestimmt ist, muss dieser dem Gerät des anfragenden Benutzers mitgeteilt werden. Dazu existieren verschiedene Möglichkeiten. Beim *Client Multiplexing* wird an den Benutzer mit der DNS-Antwort eine Liste an verfügbaren Replika-Servern ausgeliefert. Das Gerät des Benutzers wählt dann selbstständig einen Replika-Server aus der Liste aus. Da der Benutzer keine weitere Information über die zurückgelieferten Replika-Server besitzt, ist nicht ausgeschlossen, dass er einen Server mit hoher Last auswählt, was zu einer verzögerten Zustellung führt. Wird *DNS Indirection* verwendet, wählt bereits das Request Routing System einen passenden Replika-Server aus und liefert diesen mit der DNS-Antwort zurück. Nachteil dabei ist, dass DNS für statische Zuordnungen konzipiert wurde und Caching in dieser Variante nicht eingesetzt werden kann. Die einfachste, aber auch ineffizienteste Möglichkeit einen Replika-Server zuzuteilen, ist *HTTP redirection*. Dabei werden alle Anfragen an den Ursprungsserver gesendet und dieser leitet auf Anwendungsebene mittels der Weiterleitungsfunktion des HTTP-Protokolls die Anfrage an einen Replika-Server weiter.

Ein CDN bietet den Vorteil, dass es auch auf der aktuellen Internet-Architektur eingesetzt werden kann und eine schnelle und zuverlässige Bereitstellung von Inhalten ermöglicht. Es fällt damit in die Gruppe der evolutionären Ansätze. Dennoch erfordert ein

CDN eine aufwendige Planung und einen sehr hohen administrativen Aufwand, wodurch nur zahlende Kunden ihre Inhalte auf diese Weise hochverfügbar im Internet bereitstellen können.

2.4.3.3. Content Centric Network (CCN)

Die aktuelle Internet-Architektur basiert auf dem IP-Protokoll als einheitliche Schnittstelle zwischen allen Kommunikationspartnern. Jacobson et al. stellen in [JST+09] das CCN vor und führen dort eine inhaltsbasierte Schicht ein, die als neue einheitliche Schnittstelle für alle Kommunikationspartner dienen soll. Diese setzt auf IP als reinem Vermittlungsprotokoll auf, wobei auch mehrere Verbindungen über verschiedene Netzschnittstellen gleichzeitig unterstützt werden. Die einzelnen Verbindungsschnittstellen, sowie die Schnittstelle zu den Applikationen, werden als *faces* bezeichnet.

Die Kommunikation innerhalb des CCN erfolgt mittels zwei Nachrichtentypen:

- `Interest`: Diese Nachricht wird von einem Benutzer über alle Netzschnittstellen versendet und bekundet seinen Wunsch, einen bestimmten Inhalt zu erhalten, der mit einer eindeutigen Kennung identifiziert wird.
- `Data`: Ein Gerät, das eine Interest-Nachricht erhält und den gewünschten Inhalt speichert, sendet diese Nachricht und informiert somit den Benutzer, wie er den Inhalt beziehen kann.

Die Funktionsweise der neuen CCN-Schicht basiert dabei auf drei Datenbanken, die miteinander die gesamte Funktionalität des CCN bereitstellen. In der Forwarding Information Base (FIB) werden die faces gespeichert, über die bestimmte Inhalte erreicht werden können. Die CCN-FIB ist dabei vergleichbar mit einer IP-FIB, nur dass Identifier für Inhalte anstatt IP-Adressen mit den zugehörigen faces eingetragen sind. Die zweite Datenbank ist der Content Store. Data-Nachrichten für einen bestimmten Inhalt werden dort in einem Pufferspeicher zwischengespeichert, damit Inhalte, die potentiell für mehrere Benutzer von Interesse sind, schneller verfügbar sind und direkt von dem Gerät bedient werden können, das die Interest-Nachricht empfangen hat. Mit Verdrängungsstrategien wie LFU oder LRU wird die Größe des Pufferspeichers begrenzt. Die dritte Datenbank ist die Pending Interest Table (PIT), die zur Nachverfolgung von Interest-Nachrichten dient, die in Richtung Inhaltsquelle weitergeleitet werden. Sobald die zu einem PIT-Eintrag passende Data-Nachricht empfangen wird, wird der PIT-Eintrag gelöscht.

Für neu eingehende Interest-Nachrichten wird zunächst der Content Store durchsucht. Ist eine Data-Nachricht passend zur Anfrage im Puffer, wird diese direkt ausgeliefert. Ansonsten wird überprüft, ob für die empfangene Interest-Nachricht bereits ein Eintrag in der PIT existiert. Ist dies der Fall, wird das anfragende face zu dem bestehenden Eintrag hinzugefügt. Die Data-Nachricht wird dann auch an dieses face weitergeleitet. Sollte auch kein PIT-Eintrag existieren, wird die Interest-Nachricht anhand der FIB weitergeleitet und ein PIT-Eintrag erzeugt. Andernfalls wird die Interest-Nachricht verworfen.

Als Identifier, die von CCN auf der inhaltsbasierten Schicht zum Routen der Anfragen verwendet werden, kommen URIs mit dem gleichen Aufbau wie im WWW zum Einsatz. Diese sind hierarchisch und können daher für Routingaufgaben verwendet werden.

Ein CCN basiert auf der Einführung einer neuen einheitlichen Kommunikationsschicht auf allen Geräten. Es wird daher den revolutionären Ansätzen zugeordnet. Da das Routingsystem durch die PIT nicht zustandslos ist, kann die Wartung der PIT bei sehr vielen Anfragen besonders rechenaufwändig werden. Für den Fall, dass neue PIT-Einträge verworfen werden müssen, verschärft dies die Problematik sogar, da dann die Interest-Nachricht weitergeleitet wird, was zusätzlich die Netzlast erhöht.

2.4.3.4. Network of Information (NetInf)

Mit der in [ADM+08, Dan09] vorgestellten Architektur NetInf wird ebenfalls eine Architektur zur Adressierung von Inhalten vorgestellt. NetInf basiert jedoch nicht auf dem Konzept des Identifier-basierten Routings, sondern macht vom Konzept zur Trennung von Locator und Identifier Gebrauch. Dabei wird, genau wie bei der Adressierung von Geräten, ein Mapping-System benötigt, welches zunächst den Identifier eines Inhalts in die Locator-Werte der Geräte umsetzt, die den entsprechenden Inhalt speichern.

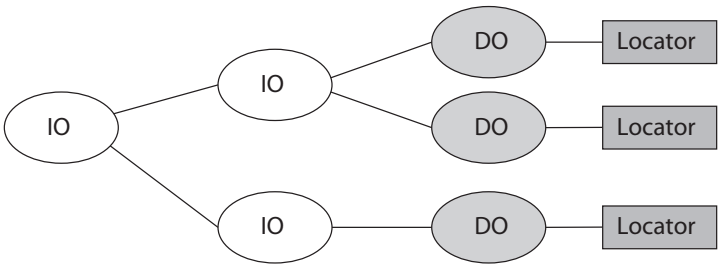


Abbildung 2.9.: NetInf Informationsmodell

Ein besonderes Augenmerk legt NetInf auf die Repräsentation der Information. Dazu wird ein Informationsmodell für Inhalte vorgestellt (vgl. Abbildung 2.9), das den Inhalt mit Hilfe von Informationsobjekten (IO) beschreibt. Jedes IO enthält Informationen über mögliche verschiedene Repräsentationsformate und verweist entweder auf ein anderes IO, oder aber auf ein Datenobjekt (DO), das letztendlich die möglichen Locator-Werte enthält, unter denen der Inhalt erreicht werden kann. Nach der Idee von NetInf kann ein Informationsobjekt (IO) nicht nur digitalen Inhalten zugewiesen werden, sondern auch realen Objekten wie Gebäuden oder anderen Ressourcen, deren Eigenschaften dadurch beschrieben werden.

Das Mapping-System von NetInf wählt dabei einen anderen Ansatz als Mapping-Systeme für rein gerätebasierte Architekturen. In NetInf erhält jedes IO einen eigenen Mappingeintrag, in dem die Beschreibung dieses IO abgespeichert ist. Sind mehrere IO miteinander verknüpft, müssen so viele Mappingabfragen erfolgen, bis ein Eintrag das gewünschte DO enthält. Anschließend kann anhand des Locators eine Verbindung mit dem entsprechenden Gerät aufgebaut werden. Die IO selbst besitzen flache Identifier einer festen Länge, die über Suchmaschinen gefunden werden können.

Der Zugriff auf die NetInf-Architektur erfolgt über eine Applikationsschnittstelle und setzt auf der aktuellen Internet-Architektur auf, weshalb NetInf als evolutionärer Ansatz bezeichnet werden kann. Inhalte können dabei sowohl auf dedizierten Geräten, als auch in

CDN oder Peer-to-Peer-Netzen gespeichert werden, die die NetInf-Applikationsschnittstelle unterstützen.

Als Mapping-System schlägt NetInf aufgrund des flachen Namensraums eine DHT vor. Obwohl diese zwar mit der Anzahl der Teilnehmer skalieren, kann das Auflösen eines IO bis zum eigentlichen DO relativ lange dauern, da abhängig von der verwendeten DHT pro verknüpftem DO eine Größenordnung von $\mathcal{O}(\log N)$ Hops innerhalb der DHT benötigt werden, um den entsprechenden Eintrag zu finden.

2.5. Zusammenfassung

In diesem Kapitel wurden die Grundlagen und Probleme der aktuellen Architektur des Internets beschrieben, die für den nachfolgenden Entwurf einer neuen Internet-Architektur relevant sind.

Der erste Teil enthielt einen Überblick über die ursprünglichen Planungsziele des Internets und der daraus resultierenden aktuellen Internetarchitektur. Im zweiten Teil wurden die Schwächen und Probleme dieser Architektur aufgezeigt, die besonders durch das rapide Wachstum der Teilnehmerzahlen entstanden sind. Im dritten Teil werden dafür bereits bestehende, evolutionäre Lösungsansätze aufgezeigt, die jedoch zueinander oft inkompatibel sind, woraus die Forderung nach dem Entwurf einer neuen, revolutionären Internet-Architektur laut wird. In diesem Zusammenhang konnte besonders die semantisch überladene IP-Adresse als Ursache für eine Vielzahl der Probleme bestimmt werden, die aktuell sowohl zur Identifikation von Geräten, als auch zum Routing verwendet wird. Neue Architekturentwürfe basieren daher häufig auf der Trennung dieser beiden Funktionen, wie auch die in Kapitel 3 vorgestellte Architektur.

Während die meisten der angesprochenen Probleme besonders die Skalierbarkeit des Internets für den reinen Datentransfer betreffen, konnte des weiteren auch die mangelnde Unterstützung von Inhalten als Problem identifiziert werden. Da das Internet zunehmend zur Informationsbeschaffung verwendet wird, rückt die reine Adressierung von Geräten in den Hintergrund. Dadurch, dass von Inhaltsobjekten aus Lastverteilungsgründen oft mehrere Objekte existieren, ist ein Identifikationsschema für Inhalte nötig, das unabhängig vom speichernden Gerät ist. Es wurden einige Konzepte diesbezüglich vorgestellt, die als Grundlage für die im Kapitel 4 vorgestellte Erweiterung der neuartigen Architektur aus Kapitel 3 dienen.

3. Die HiiMap Architektur für das Internet der Zukunft

Der Entwurf von Konzepten für das Internet der Zukunft ist von einer Reihe von Faktoren abhängig. Zum einen sollen möglichst viele Schwachstellen der aktuellen Internet-Architektur beseitigt und dadurch die Zukunftsfähigkeit sichergestellt werden, zum anderen soll die neue Architektur möglichst wenig in die aktuelle Netz- und Anwendungs-Infrastruktur eingreifen. Eine neuartige Internet-Architektur muss sowohl von den Netzbetreibern, als auch von den Software- und Anwendungsentwicklern akzeptiert werden. In diesem Kapitel wird die *Hierarchical Internet Mapping Architecture (HiiMap)* vorgestellt, ein Konzept für eine neuartige Internet-Architektur, die auf der Trennung von Locator und Identifier basiert. Im Gegensatz zu anderen Konzepten vereint HiiMap die Vorteile einer Locator/Identifier-Architektur in Bezug auf Mobilität, Heterogenität der Endgeräte und Skalierbarkeit des Routingsystems sowie die Möglichkeit, auch abstrakte Endpunkte wie Informationen und Personen zu adressieren. Dies wird durch eine flexible Gestaltung von Identifier und Locator ermöglicht. HiiMap bietet darüber hinaus ein sicheres Kommunikationsumfeld durch die Verwendung einer PKI, mit der die Schutzziele der Informationstechnik wie Integrität, Authentizität und Verschlüsselung erreicht werden. Einzuordnen ist HiiMap in die Gruppe der revolutionären Architekturen. Das bedeutet, dass eine Kompatibilität zur aktuellen Internet-Architektur nur unter bestimmten Voraussetzungen gegeben ist. Für HiiMap sind Modifikationen des Netzwerk-Sstacks in jedem Gerät notwendig, dennoch bleibt der aktuelle Aufbau des Internets mit seinen einzelnen AS unangetastet.

Dieses Kapitel befasst sich mit der Beschreibung der HiiMap-Architektur für Endgeräte. Die Unterstützung von Inhalten und Informationen sowie Personen mit den nötigen Erweiterungen wird in Kapitel 4 behandelt. Im ersten Abschnitt wird ein Überblick über die Unterschiede zu bestehenden Architekturkonzepten gegeben, die ebenfalls auf der Trennung von Locator und Identifier basieren. Dabei wird besonders auf die Struktur des Locators eingegangen und auf die Vorteile, die sich durch dessen Zweiteilung in Bezug auf eine heterogene Gerätelandschaft ergeben. Im zweiten und dritten Abschnitt werden zwei Realisierungsmöglichkeiten für das Mapping-System mittels DHT vorgestellt, das ein unverzichtbarer Bestandteil jedes Locator/Identifier-getrennten Systems ist. Nach einem Vergleich und einer Bewertung der beiden Systeme wird im vierten Abschnitt ein heterogenes Namenschema für Identifier zusammen mit Strategien zur Vergabe und Registrierung vorgestellt. Der fünfte Abschnitt befasst sich mit den verschiedenen Formen von Endgerätemobilität. Dabei werden Mechanismen aufgezeigt, wie diese vollständig in HiiMap unterstützt werden. Im letzten Abschnitt werden für HiiMap nötige Modifikationen im Netzwerk-Stack aufgezeigt. Dabei wird ein modulares System beschrieben, das auf Funktionsblöcken basiert und welches das inflexible Schichtenmodell ablösen kann.

3.1. Abgrenzung und Unterschiede zu bestehenden Konzepten

Bereits in Kapitel 2 wurden Konzepte zur Weiterentwicklung des Internets vorgestellt. Dabei können Konzepte, die auf der Trennung von Identifier und Locator basieren, in einem rein geräteorientierten Internet erfolgreich Probleme bezüglich Skalierbarkeit und Mobilität lösen. Auch HiiMap verwendet als revolutionärer Ansatz das Prinzip der Trennung von Locator und Identifier, unterscheidet sich jedoch von existierenden Konzepten in einigen maßgeblichen Punkten. Abbildung 3.1 zeigt die Einordnung von HiiMap in bereits bestehende Konzepte und entsprechende Unterschiede. Diese werden Folgenden erläutert.

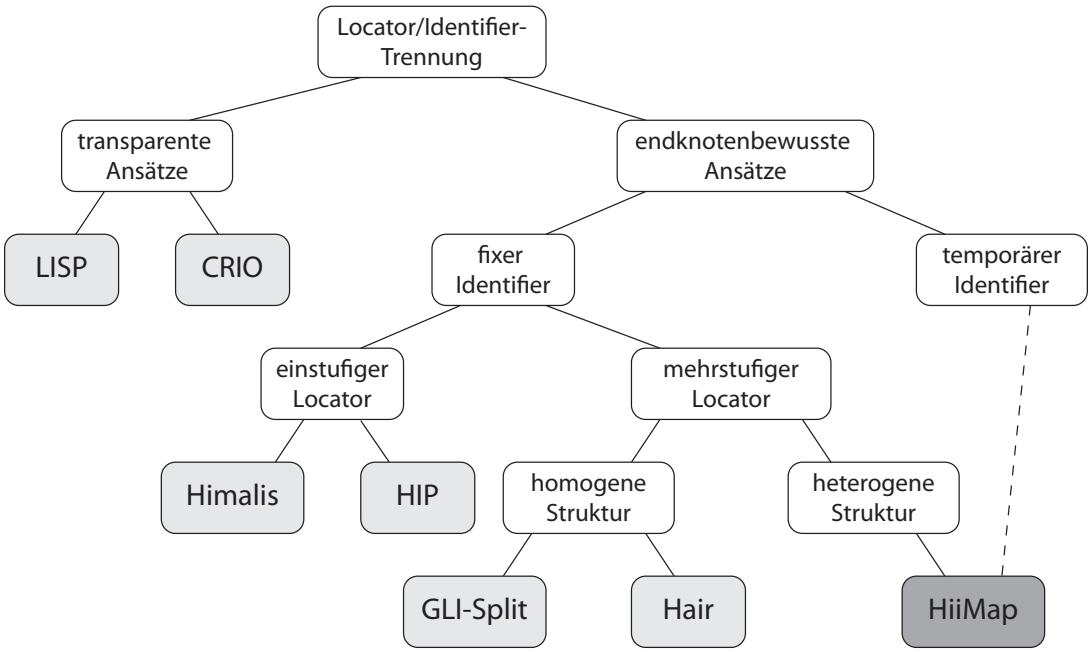


Abbildung 3.1.: Einordnung von HiiMap in bereits bestehende Konzepte

HiiMap ist den endknotenbewussten Ansätzen zuzuordnen. Jedes Gerät ist sich der Trennung von Locator und Identifier bewusst und Kommunikation erfolgt aus Sicht der Anwendung ausschließlich mit dem Identifier. HiiMap verwendet dazu einen so genannten Unique Identifier (UID), der über die gesamte Lebensdauer des ihm zugewiesenen Objekts seine Gültigkeit behält und diesen fixen Wert nicht ändert, auch bei Mobilität des Objekts. Als Alleinstellungsmerkmal bietet HiiMap im Gegensatz zu allen anderen Konzepten zusätzlich die Möglichkeit einem Gerät einen temporären UID zuzuweisen.

Um maximale Kompatibilität mit der aktuellen Topologie des Internets zu gewährleisten, sowie politische, infrastrukturelle, als auch wirtschaftliche Aspekte zu berücksichtigen, basiert HiiMap auf einzelnen AS. Für das Routing verwendet HiiMap einen mehrstufigen Locator, der aus zwei Teilen besteht. Ein Teil wird für das Inter-Domain-Routing verwendet (Global Gateway Unique Identifier (GGUID)), der andere ausschließlich für

Intra-Domain-Routing (Local Temporary Address (LTA)). Darüber hinaus erlaubt der Locator in der zweiten Stufe eine heterogene Struktur. Damit ist es möglich, in dieser Stufe ein beliebiges Adressierungsschema zu verwenden, das an die Anforderungen des Netzes und der Endgeräte angepasst ist.

Der Übergang vom Intra-Domain-Bereich eines AS in den Inter-Domain-Bereich wird jeweils von so genannten Core Network Routern (CNR) übernommen. Diese ersetzen die bekannten BGR und erweitern sie um einige Funktionalitäten, die zusammen mit dem Locatorschema im Folgenden Abschnitt erläutert werden. Im Intra-Domain-Bereich kommen so genannte Local Network Router (LNR) zum Einsatz, die an das Schema der verwendeten LTA angepasst sind.

Abbildung 3.2 zeigt eine Übersicht über die HiiMap-Architektur. Im dunkelgrau dargestellten Inter-Domain-Bereich wird nur der GGUID zum Routing verwendet, im hellgrau dargestellten Intra-Domain-Bereich nur die LTA. Darüber hinaus besitzt jedes Gerät einen UID.

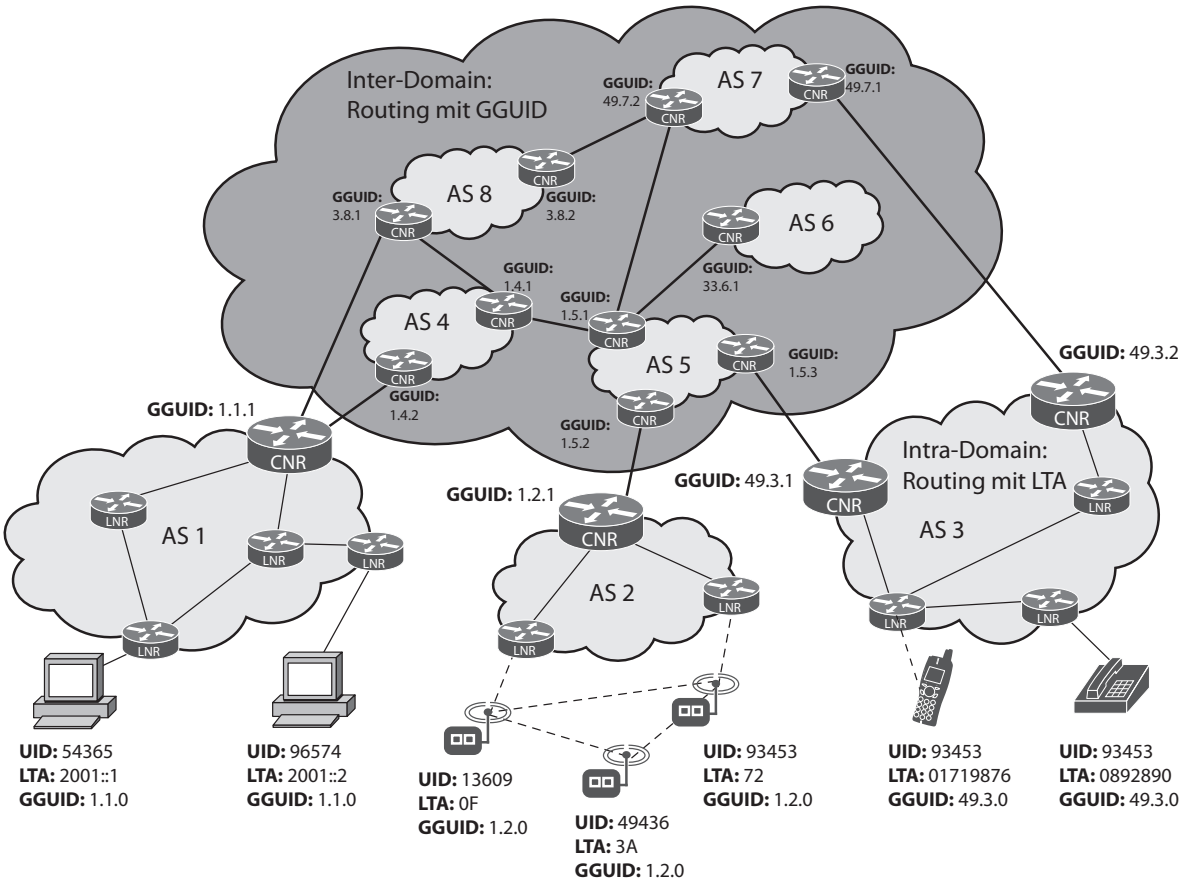


Abbildung 3.2.: Übersicht der HiiMap-Architektur

3.2. Locator und Routing

Der vollständige Locator, der einem Gerät zugewiesen wird und für das Routing notwendig ist, besteht aus zwei Teilen. Diese werden im Folgenden detailliert vorgestellt.

3.2.1. Local Temporary Address (LTA)

Der lokale Teil des Locators, LTA genannt, ist eine lokale routingfähige Adresse eines beliebigen Vermittlungsprotokolls, die jeder Netzschnittstelle eines Gerätes vom jeweils angeschlossenen Betreiber temporär zugewiesen wird und nur in jeweils diesem AS gültig ist. Ändert oder wechselt das Gerät seinen Netzzugangspunkt an einer seiner Netzschnittstellen, wird ihm an dieser Schnittstelle eine neue LTA zugeordnet. Ein großer Vorteil, der sich aus der Trennung des Locators in zwei Stufen ergibt, ist die Tatsache, dass jeder Betreiber in seinem AS ein beliebiges Vermittlungsprotokoll und Adressierungsschema verwenden kann. So kann ein Betreiber beispielsweise intern ein hierarchisches Adressierungsschema ähnlich IPv4 mit 32 Bit verwenden, ein anderer setzt auf ein Schema ähnlich IPv6, IPX oder ein beliebiges anderes. In Abbildung 3.2 wird dies verdeutlicht. Der Telefonnetzbetreiber mit der AS-Kennung 3 verwendet dabei Rufnummern als LTA. Der Sensornetz-Betreiber mit der AS-Kennung 2 verwendet ein Adressierungsschema, das auf flachen, nicht hierarchischen Adressen basiert. Der Betreiber von AS 1 verwendet ein hierarchisches Adressierungsschema mit 128 Bit ähnlich IPv6. Innerhalb eines AS wird der Verkehr von den LNR weitergeleitet. Diese müssen nur das lokal verwendete Vermittlungsprotokoll verarbeiten können.

3.2.2. Global Gateway Unique Identifier (GGUID)

Der globale Teil des Locators, GGUID genannt, gibt an, über welchen Core Network Router (CNR) das AS, in dem sich das entsprechende Gerät gerade befindet, erreicht werden kann. Dabei gelten besondere Regeln für den Aufbau der GGUID, der als Besonderheit im Inter-Domain-Bereich sowohl die Aufgaben des Identifiers als auch des Locators beinhaltet und für das Routing zuständig ist. Er wird ausschließlich einem CNR zugewiesen, der damit nur *eine* Adresse besitzt, die für das Routing verwendet

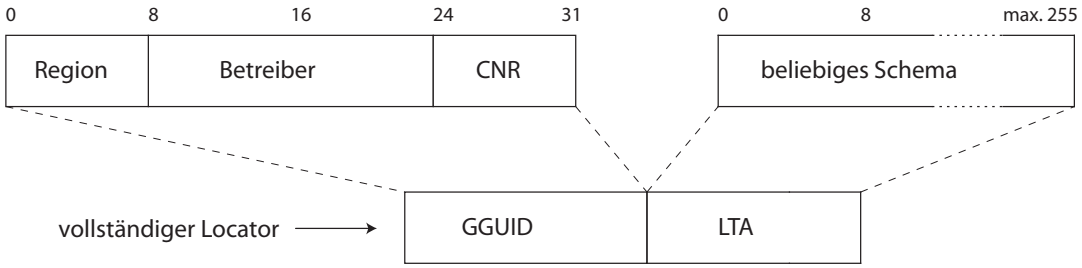


Abbildung 3.3.: Zweistufiger Locator in HiiMap mit GGUID und LTA

wird und als Kennung dient. Der GGUID ist hierarchisch aufgebaut und besteht aus 32 Bit, die in drei Blöcke aufgeteilt werden (vgl. Abbildung 3.3). Der erste Block mit 8 Bit identifiziert die Region bzw. das Land, der zweite Block mit 16 Bit den Betreiber, wobei dieser Wert global eindeutig ist und nicht pro Region neu vergeben werden kann. Der dritte Block mit 8 Bit identifiziert die einzelnen CNR des Betreibers. Das AS eines Netzbetreibers kann dabei auch über mehrere Regionen verteilt sein, wobei sich der GGUID der CNR in diesem AS dann nicht nur im letzten Block, sondern auch im ersten Block unterscheiden. Der mittlere Block mit der global eindeutigen Betreiber-Kennung ändert sich dabei nicht. Es wird davon ausgegangen, dass mit 32 Bit ein ausreichend großer Adressraum für die Adressierung aller CNR auch im Hinblick auf die Zukunft des Internets zur Verfügung steht. Voraussetzung für den Betrieb ist, dass alle CNR auf ihren externen Netzschnittstellen das gleiche Vermittlungsprotokoll sprechen. Die CNR fungieren dabei als Umsetzer zwischen dem gemeinsamen globalen Vermittlungsprotokoll und dem jeweils frei zu wählenden internen Vermittlungsprotokoll. Sie müssen daher beide Protokolle beherrschen.

LTA und GGUID bilden zusammen den gesamten global gültigen Locator eines Geräts, der in Abbildung 3.3 dargestellt ist. Durch die Trennung in diese zwei voneinander unabhängigen Teile ist HiiMap auch für eine heterogene Netzinfrastruktur geeignet. Dabei können auch Geräte einfach kommunizieren, die jeweils ein anderes LTA-Schema verwenden, da die Ziel-LTA zwar im Packet-Header mitgeführt wird, von dem sendenden Gerät jedoch nicht ausgewertet werden muss. Ein Gerät kann dabei mehrere LTA- und GGUID-Paare besitzen, falls es über mehrere Netzschnittstellen verbunden ist. Der exakte CNR wird für Routingzwecke nicht spezifiziert, um es den Betreibern zu ermöglichen, ihre Wegeführung für Pakete dynamisch anzupassen. Dazu wird der letzte Block des GGUID bei der Vergabe an Geräte zu Null gesetzt. Durch die global eindeutige Betreiberkennung ist der gesamte Locator dennoch gültig. Der Betreiber ersetzt diesen Wert dann an geeigneter Stelle mit der Kennung des zuständigen CNR.

3.2.3. Routing und Forwarding

HiiMap unterscheidet beim Routing und Forwarding zwischen dem Inter-Domain-Bereich und dem Intra-Domain-Bereich. Während im Inter-Domain-Bereich ausschließlich der GGUID routingfähig ist, so ist es im Intra-Domain-Bereich die LTA. Für das Forwarding verwendet HiiMap im Inter-Domain-Bereich ebenfalls den GGUID, im Intra-Domain-Bereich wird dafür der UID verwendet. Stellvertretend für beide möglichen Adressen wird im Folgenden der Begriff Forwarding-UID (FW-UID) dafür verwendet. Die genaue Funktionsweise, sowie die damit verbundene Reduktion der Routingtabellengröße, wird im Folgenden erklärt.

3.2.3.1. Inter-Domain

Durch die Verwendung eines hierarchischen Adressierungsschemas für den GGUID, ähnlich des E.164 Nummernplan der ITU-T für internationale Telefonnummern, lassen sich ganze Regionen zu einem Routingpräfix zusammen fassen. Auf diese Weise wird die Größe der Routingtabellen in den CNRs drastisch reduziert, da im Idealfall für alle Betreiber

einer bestimmten Region nur ein Routingpräfix, nämlich das der Region, nötig ist. Aufgrund von Peering-Abkommen und Traffic-Engineering können dennoch zusätzliche Routingtabelleneinträge erforderlich sein. Die Anzahl dafür fällt aber im Vergleich zum Zuwachs, der durch Deaggregation aufgrund von Multihoming oder betreiberunabhängigen Adressen in der aktuellen Internet-Architektur erzeugt wird, sehr gering aus [CMU+10]. Dadurch, dass in HiiMap Geräte ausschließlich über ihre UID angesprochen werden, besteht kein Bedarf mehr für betreiberunabhängige Adressen. Auch Multihoming, das aus Gründen der Ausfallsicherheit in Zukunft weiterhin zur gängigen Praxis gehören wird, hat keinen Einfluss mehr auf die Routingtabellengröße, da einem UID mehrere Locatorwerte zugeteilt werden können.

Zum Erstellen der Routingeinträge und Austauschen der Routinginformationen zwischen benachbarten AS verwendet HiiMap ein Pfadvektorprotokoll ähnlich BGP, da dieses die Möglichkeit bietet, Routen abhängig von Kooperationsverträgen zwischen den Betreibern zu konfigurieren.

Ausgehend von den Einträgen der Routingtabellen in den CNR werden die Daten schrittweise durch das Netz hin zum Ziel weitergeleitet. Der erste CNR, der ein Paket innerhalb seines AS empfängt, das für ein anderes AS bestimmt ist, wird als Ausgangs-CNR bezeichnet. Dieser leitet das Paket anhand des Ziel-GGUID und den Einträgen seiner Routingtabelle an einen benachbarten CNR weiter, der wiederum als Eingangs-CNR fungiert. Für das Weiterleiten wird der GGUID als FW-UID verwendet. Ein Transit-AS, das auf dem Weg zum Ziel-AS durchlaufen wird, setzt die Pakete nicht in das lokal verwendete Adressierungsschema basierend auf der LTA um. Stattdessen werden die Pakete anhand der Routingtabelle mit einer GMPLS (Generalized Multi Protocol Label Switching) Verbindung direkt zum zuständigen Ausgangs-CNR geleitet. Ist der Eingangs-CNR des Ziel-AS erreicht, leitet dieser das Paket an seiner internen Netzschnittstelle zum nächsten LNR weiter.

Während die Kennung des CNR bei der Vergabe des GGUID an die Endgeräte nicht spezifiziert wird, so ist dies beim Austausch der Routinginformation und beim Forwarding zwingend nötig, da in diesem Fall die einzelnen CNR explizit angesprochen werden.

3.2.3.2. Intra-Domain

Innerhalb eines AS ist die Wahl des Routingprotokolls von dem verwendeten Vermittlungsprotokoll abhängig, wobei letzteres vom Betreiber frei gewählt werden kann. Möglich sind Link-State-Routingprotokolle ähnlich OSPF oder IS-IS, oder Distanzvektorprotokolle wie das Routing Information Protocol (RIP) oder das Interior Gateway Routing Protocol (IGRP). Zusätzlich unterstützen die LNR das Generalized Multi Protocol Label Switching (GMPLS), um Transitverkehr der CNR weiterzuleiten.

Datenpakete eine Geräts werden zunächst an den benachbarten LNR gesendet. Zum Weiterleiten von Paketen an benachbarte Geräte innerhalb eines AS wird direkt die UID als FW-UID verwendet. Entspricht die Betreiberkennung des Ziel-GGUID dem des Betreibers, wird das Paket mit dem lokal verwendeten Adressierungsschema direkt von den LNR zum Ziel gesendet. Handelt es sich um eine andere Betreiberkennung, wird das Paket an einen CNR geleitet.

3.2.4. Packet Header

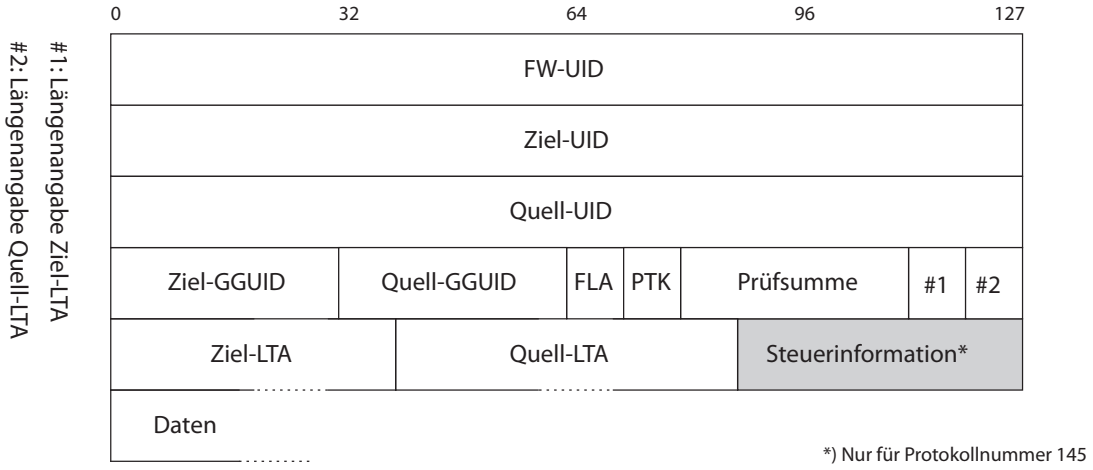


Abbildung 3.4.: HiiMap-Header

Abbildung 3.4 zeigt den HiiMap Packet Header mit allen benötigten Feldern. Der FW-UID bildet den ersten Eintrag, gefolgt von den UID des Ziels und der Quelle. Im Inter-Domain-Bereich sind die ersten 96 Bit der FW-UID zu Null gesetzt, da der in diesem Fall verwendete GGUID lediglich 32 Bit breit ist. Anschließend folgen die GGUID von Ziel und Quelle. Auf die GGUID-Felder folgen ein Flag-Feld (FLA) sowie ein Protokoll-Feld (PTK), die beide 8 Bit breit sind. Das Flag-Feld ist in Abbildung 3.5 links dargestellt, wovon aktuell zwei Bits verwendet werden:

- `LUP` (Location Update): Dieses Bit signalisiert die Aktualisierung der Locatorwerte. Es kann direkt in einer laufenden Datenübertragung gesetzt sein, oder als Paket mit leeren Nutzdaten versendet werden.
- `ENC` (Encryption): Dieses Bit signalisiert eine verschlüsselte Datenübertragung. Ver- und Entschlüsselungsmodule werden für diese Übertragung im Netzwerkstack aktiviert.

Die Grau hinterlegten Felder in Abbildung 3.4 werden aktuell noch nicht verwendet und können für zukünftige Erweiterungen verwendet werden.

Das Protokoll-Feld PTK enthält Informationen über ein mögliches verwendetes Transportprotokoll. Dabei kommen vorgegebene Werte aus [AB08, IAN11] für bereits existierende Transportprotokolle zum Einsatz, die um weitere Werte, wie in Tabelle 3.1 dargestellt, ergänzt werden. Für die Protokoll-Nummern 0 bis 142 folgt im Anschluss an das Quell-LTA-Feld direkt das erste Bit des Headers des verwendeten Transportprotokolls. Der Wert 143 wird für Tunneling-Verfahren reserviert, die für die Migration zu Hiimap notwendig sind und im Kapitel 5 diskutiert werden. Die Werte 144 und 145 werden von HiiMap selbst zum Signalisieren eigener Übertragungsmodi verwendet. Die Werte ab 146 sind für zukünftige Erweiterungen frei. Wird kein zusätzliches Transportprotokoll verwendet, übernimmt HiiMap vollständig die Aufgaben eines Transportprotokolls.

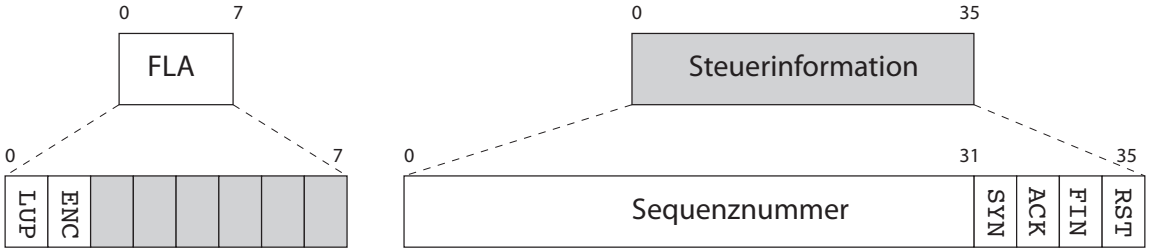


Abbildung 3.5.: Flag-Feld und Steuerinformationen im Header

Für eine gesicherte HiiMap-Verbindung mit der Protokoll-Nummer 144 wird dabei am Ende des Quell-LTA-Felds ein weiteres Datenfeld mit Steuerinformationen eingefügt. Das Feld Steuerinformationen ist in Abbildung 3.5 rechts dargestellt und enthält neben einer 32 Bit breiten Sequenznummer noch vier weitere Flags zur Verbindungssteuerung. Die Sequenznummer wird benötigt, um Datenpakete sortieren und erneut anfordern zu können, falls diese im Netz verworfen oder beschädigt werden. Zur Steuerung der gesicherten Verbindung werden folgende vier Flags verwendet:

- `SYN, ACK`: Diese Bits werden für den Drei-Wege-Handschlag beim Aufbau einer gesicherten Verbindung verwendet. `ACK` wird darüber hinaus verwendet, um Sequenznummern bzw. ganze Nummernblöcke zu bestätigen, ähnlich TCP.

- `FIN`: Dieses Bit wird verwendet um eine bestehende Verbindung zu beenden und diese frei zu geben. Es zeigt an, dass keine weiteren Daten mehr gesendet werden.

- `RST`: Im Gegensatz zu `FIN` wird dieses Bit gesetzt, um eine bestehende Verbindung bzw. den Versuch eines Verbindungsaufbaus bewusst abzubrechen.

Die Nutzdaten einer gesicherten HiiMap-Verbindung werden direkt im Anschluss an das Feld Steuerinformation gesendet.

Tabelle 3.1.: Werte für das Protokoll-Feld

Protokoll-Nummer	Funktion
0–142	Vorgegebene Werte aus [AB08, IAN11]
143	Reserviert für HiiMap-über-IP Tunneling
144	Gesicherte HiiMap-Verbindung
145	Verbindungsloses HiiMap-Datagramm
146–255	frei für zukünftige Erweiterungen

Ein HiiMap-Datagramm mit der Protokoll-Nummer 145 stellt eine ungesicherte Übertragung dar und verwendet daher keine Steuerinformationen. Die Nutzdaten beginnen daher direkt nach dem Quell-LTA-Feld.

Nach dem Protokoll-Feld folgt im HiiMap-Header eine 32 Bit breite Prüfsumme. Diese wird aus dem gesamten HiiMap-Paket mit Ausnahme der FW-UID berechnet. Für die Berechnung wird die Prüfsumme selbst auf Null gesetzt.

Auf die Prüfsumme folgen zwei Felder zur Längenangabe der Ziel- und Quell-LTA. Diese Angabe ist nötig, da die LTA je nach verwendetem Adressierungsschema in den einzelnen AS eine unterschiedliche Länge besitzen kann.

Die Datenfelder im Header sind so angeordnet, dass eine zügige Verarbeitung der Pakete auch auf Hardwarebasis erfolgen kann, da alle Datenfelder mit fester Länge am Anfang des Headers platziert sind. CNRs im Inter-Domain Bereich können direkt auf die benötigten Felder des Ziel-GGUID und des FW-UID zugreifen, ohne vorher Längenfelder auswerten zu müssen. Selbst LNR in den AS der einzelnen Betreiber können das Ziel-LTA-Feld direkt einlesen ohne vorher das Längenfeld auszuwerten, da die Länge der LTA im eigenen Netz bekannt ist. Lediglich die beiden kommunizierenden Geräte, die die Endpunkte der Kommunikation darstellen, müssen die Längenfelder und die zugehörige LTA auswerten, da die Daten für die Antwortpakete an das jeweilige andere Gerät benötigt werden.

3.2.5. Sicherheit durch PKI

HiiMap verwendet das Prinzip einer PKI [DH76], um Schutzziele der Informationssicherheit wie Authentizität, Integrität und Datenverschlüsselung zu erreichen. Jedes Gerät generiert dabei initial einen öffentlichen und einen privaten Schlüssel. Der private Schlüssel wird von jedem Gerät geheim gehalten und dient zum Entschlüsseln und Signieren von Nachrichten. Mit dem öffentlichen Schlüssel, der zusammen mit den Locator-Informationen eines jeden Gerätss im Mapping-System gespeichert wird, können andere Geräte Nachrichten für dieses bestimmte Gerät verschlüsseln. Derartig verschlüsselte Nachrichten können nur mit dem dazugehörigen privaten Schlüssel wieder lesbar gemacht werden. Darüber hinaus dient der öffentliche Schlüssel dazu, Signaturen zu verifizieren.

Im Gegensatz zum HIP [MJNH08], das den Identifier aus dem Hash-Wert des öffentlichen Schlüssels eines Knotens generiert, besteht in HiiMap keine feste Verbindung zwischen UID und öffentlichem Schlüssel. Ein jederzeitiger Austausch des Schlüsselpaares ist dadurch möglich, ohne dass sich der UID eines Knotens ändert. Auf diese Weise wird der Eigenschaft eines UID Rechnung getragen, für die Lebensdauer eines Gerätes unverändert seine Gültigkeit zu behalten.

Durch die Integration einer PKI in die HiiMap-Architektur, die es erlaubt, den öffentlichen Schlüssel eines jeden Gerätes bereits automatisch zusammen mit den Locator-Informationen zu erhalten, können die geforderten Sicherheitsaspekte inhärent erreicht werden. Dennoch sollen Funktionalitäten wie Verschlüsselung nur als Option zur Verfügung gestellt werden, die Benutzung sollte aber nicht zwingend erfolgen und die Entscheidung darüber der Anwendung überlassen werden. Geräte wie Sensoren, die Daten über das Internet austauschen, besitzen unter Umständen nicht genügend Rechenkapazität um komplexe kryptographische Berechnungen durchzuführen. Das ausführliche Konzept

zur Integration der PKI in HiiMap wird in [HF11] und [HEP+11] vorgestellt.

3.3. Mapping System

Wie bereits in Abschnitt 2.4.2 erläutert, ist ein Mapping-System eine grundlegende Voraussetzung jeder Internet-Architektur, die auf der Trennung von Locator und Identifier basiert. Da HiiMap ebenso auf diesem Konzept basiert, wird ein Mapping-System für die Umsetzung des UID in den Locator benötigt. In Anbetracht der Prognosen über die Teilnehmerzahl im Internet der Zukunft muss dieses System in hohem Maße belastbar, zuverlässig und ausfallsicher, sowie skalierbar sein. Besonders mobile Endgeräte, die ihre Mapping-Einträge bei jedem Locatorwechsel aktualisieren müssen, erzeugen eine hohe Last auf das Mapping System. HiiMap bietet zwei Konzepte für ein Mapping System basierend auf der Grundlage von DHT, die sich jedoch grundlegend voneinander unterscheiden. In der ersten Lösung kommt ein zweistufiges DHT-System zum Einsatz [HSKE09]. Die zweite Lösung dagegen setzt auf einzelne DHT-Regionen zusammen mit einer zentralen Instanz und bietet darüber hinaus erweiterte Sicherheitsfunktionen [HSK+09]. Die beiden Ansätze werden im Folgenden beschrieben. Die jeweils zu erwartende Last wird abgeschätzt und am Ende des Abschnitts bewertet und verglichen.

3.3.1. HiiMap Lösung 1: Zweistufige DHT

Der erste Lösungsansatz, auch HiiMap v1 genannt, setzt auf einem zweistufigen System aus DHTs auf, um die zu einem bestimmten UID gehörenden Mapping-Einträge zu finden. Er gehört damit zur Gruppe der TLMS, wobei jede einzelne Stufe wiederum der Gruppe der MRFO zuzuschreiben ist. Dabei werden Informationen zur LTA und zum GGUID in den jeweils verschiedenen Stufen gespeichert, um das System leistungsfähig zu gestalten. Die erste Stufe – die lokale DHT – speichert dabei die Zuordnung von UID zu LTA. Die zweite Stufe – die globale DHT – speichert die zu einem UID gehörigen GGUID.

3.3.1.1. Globale DHT

Die globale DHT besitzt Mapping-Einträg für jeden registrierten UID und speichert den im aktuellen Moment für das Gerät gültigen GGUID. Der entsprechende Eintrag im Mapping-System ist in Abbildung 3.6 dargestellt. Dabei kann, wie bereits erwähnt, der letzte Block des GGUID zu Null gesetzt werden, so dass die Daten ausschließlich anhand der Betreiberkennung geroutet werden können. Die Anzahl der GGUID, die das Gerät aktuell besitzt, werden als eigenes Datenfeld im globalen Mapping-Eintrag abgespeichert. Zusätzlich besitzt jeder Mapping-Eintrag einen Time-To-Live (TTL) Wert, um dem Eintrag für eventuelles Caching eine begrenzte Lebensdauer zuzuteilen.

An der globalen DHT nehmen alle CNR aller Betreiber Teil. Als Indexschlüssel für den Aufbau der DHT und die Zuständigkeit des Wertebereichs wird dabei der GGUID der einzelnen CNRs verwendet. Der UID dient als Indexschlüssel zur Verteilung der Mapping-Einträge auf die jeweils zuständigen CNR. Die Mapping-Einträge und damit auch die durch Anfragen erzeugte Last werden somit gleichmäßig auf alle Betreiber weltweit aufgeteilt. Betreiber von größeren AS erzeugen mehr Anfragen, da mehr Teilnehmer

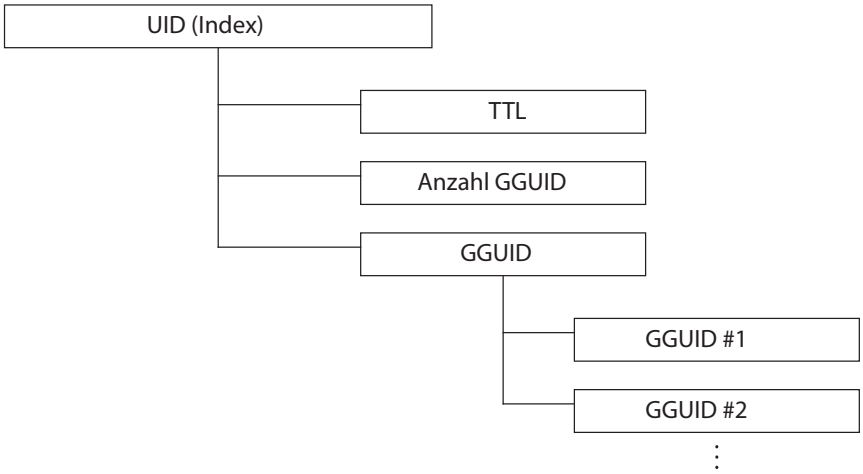


Abbildung 3.6.: Mapping-Eintrag in der globalen DHT

angeschlossen sind. Da diese Betreiber aber auch mehrere CNR besitzen, übernehmen diese damit auch einen größeren Teil der Last, die durch den Anfrageverkehr erzeugt wird. Der Ansatz mit DHT hat zwei entscheidende Vorteile. Zum einen ist eine Baumstruktur mit zentralen Instanzen – wie im heutigen DNS – sehr fehleranfällig entlang des Weges zur Wurzel. Auch der Ausfall der Wurzel selbst führt dazu, dass das System nur noch sehr eingeschränkt funktioniert. Zum anderen führt die Dezentralisierung dazu, dass keine Organisation die komplette Kontrolle über das gesamte Mapping-System besitzt. Jeder Teilnehmer an der DHT ist nur für einen kleinen Teil der insgesamt gespeicherten Einträge verantwortlich. Durch Mechanismen wie Replikation, die es erlauben gleiche Einträge redundant auf mehrere Knoten zu verteilen, wird die Zuverlässigkeit noch weiter erhöht. Um die DHT auch vor sich bösartig verhaltenden oder falsch konfigurierten CNR zu schützen, können zusätzlich so genannte Blacklists geführt werden, die bestimmte CNR oder ganze Betreiber aus der DHT ausschließen können. Dafür sind jedoch Mechanismen nötig, die das kooperative Hinzufügen und Entfernen von Einträgen unterstützen, um die Liste nicht selbst zu missbrauchen.

Um die Nachschlagezeit in der globalen DHT möglichst gering zu halten, verwendet HiiMap v1 ein One-Hop Protokoll, wie D1HT [MA06]. Verglichen mit DHT-Protokollen wie Chord, das Pfadlängen in der Ordnung von $\mathcal{O}(logN)$ besitzt, hat D1HT eine Pfadlänge von $\mathcal{O}(1)$ und kann den gewünschten Knoten damit ohne Zwischenschritt erreichen. Dies resultiert in einem Geschwindigkeitsvorteil gegenüber Lösungen mit längeren Lookup-Pfaden, jedoch auch in großen Lookup-Tabellen, da für jeden Knoten der DHT in der Lookup-Tabelle ein eigener Eintrag existiert. Des weiteren erzeugt eine One-Hop DHT in dynamischen Netzen sehr hohen Signalisierungsverkehr, da die Lookup-Tabellen auf allen Knoten aktualisiert werden müssen, falls ein Knoten neu zur DHT hinzukommt oder diese verlässt. Da die Fluktuationsrate für CNR jedoch als sehr gering angesehen werden kann, im Bereich von wenigen Ereignissen pro Monat, ist kein erhöhtes Verkehrsaufkommen durch die One-Hop DHT zu erwarten, verglichen mit Protokollen der Größenordnung von $\mathcal{O}(logN)$ bezüglich der Pfadlängen.

Die Anzahl an CNR, die an der globalen DHT teilnehmen, wird auf einen Bereich

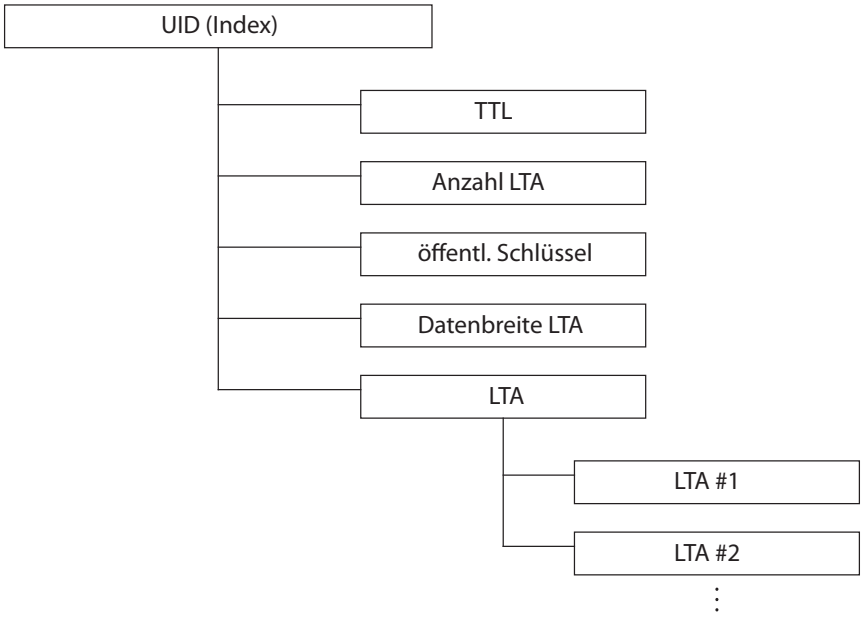


Abbildung 3.7.: Mapping-Eintrag in der lokalen DHT

von 100.000 bis maximal 200.000 geschätzt, ausgehend von 50.000 bis 100.000 AS mit im Durchschnitt jeweils zwei CNR. Monnerat konnte in seinen Studien [MA06] belegen, dass D1HT auch für über eine Million Teilnehmer bei einer mittleren Fluktuationsrate von 2,9 Ereignissen pro Stunde skaliert.

3.3.1.2. Lokale DHT

In jedem AS existiert eine eigene lokale DHT, welche die Mapping-Einträge (vgl. Abbildung 3.7) für die in diesem AS angeschlossenen Geräte verwaltet. Die lokale DHT ist zuständig für das Umsetzen eines UID in die aktuell gültige LTA. Der lokale Mapping-Eintrag ist dabei ähnlich aufgebaut wie der globale; anstatt des GGUID werden aber die aktuell gültigen LTA abgespeichert. Für die LTA existiert zusätzlich ein Feld, das die Datenbreite der LTA angibt, da diese vom Betreiber abhängig ist. Darüber hinaus wird der öffentliche Schlüssel des Geräts für die PKI mit abgespeichert.

An einer lokalen DHT partizipieren alle LNR, die sich in diesem AS befinden. Als Indexschlüssel kann dabei entweder der UID der einzelnen LNR dienen, oder der Betreiber verwendet ein eigenes Schema, um die DHT nach seinen Bedürfnissen optimal aufzubauen. Obwohl von den LNR eine höhere Fluktuationsrate erwartet wird als von den CNR, verwendet HiiMap v1 lokal ebenfalls eine One-Hop DHT. Da die Anzahl an LNR innerhalb eines AS deutlich geringer ist als die Anzahl an CNR, kann angenommen werden, dass die lokale DHT ebenfalls skaliert. Die Verwendung der lokalen DHT erlaubt es zudem gezielt Geräte als Datenquelle zu verwenden, die sich im gleichen AS befinden. Damit kann teurer Inter-Domain-Verkehr umgangen werden. Kommen mehrere UID als

Datenquelle in Frage, kann durch Anfrage an die lokale DHT herausgefunden werden, ob sich entsprechende Geräte im gleichen AS befinden. Derartige Geräte werden bevorzugt verwendet. Ein Gerät kann mit seinem UID auch in mehreren verschiedenen lokalen DHT gespeichert sein, falls es aufgrund von Multihoming Verbindungen zu mehreren AS besitzt. Ebenso sind mehrere LTA im gleichen AS möglich.

3.3.1.3. Mapping Lookup

HiiMap v1 verwendet einen rekursiven Ansatz, um Mapping-Anfragen zu beantworten. Die Mapping-Anfrage eines Gerätes A für die Mapping-Information eines Gerätes B lässt sich in drei Phasen einteilen, wobei das Gerät A selbst aktiv nur an der ersten beteiligt ist. Diese drei Phasen sind in Abbildung 3.8 dargestellt und nachfolgend beschrieben.

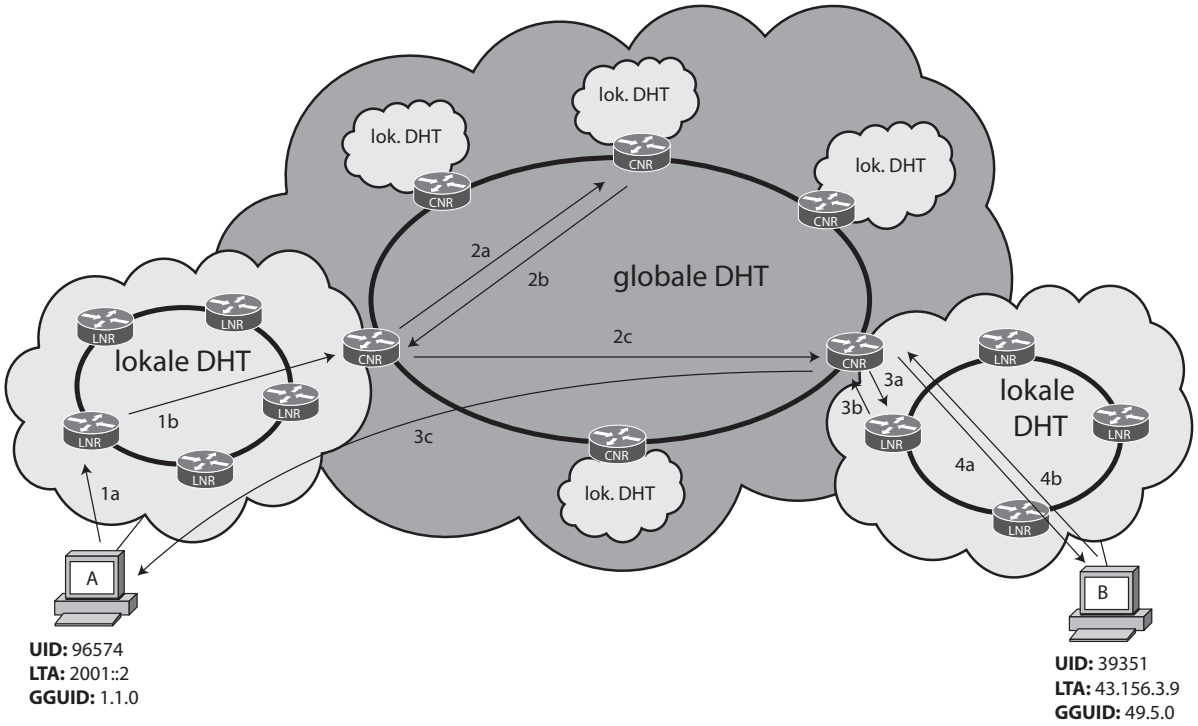


Abbildung 3.8.: Ablauf eines Mapping Lookups in HiiMap v1

1. Die Mapping-Anfrage wird von Gerät A in der ersten Phase zunächst an den nächsten LNR geschickt (1a). Die Anfrage enthält dabei die eigene UID und LTA von Gerät A. Der LNR überprüft, ob die angefragt UID von Gerät B mit Hilfe der lokalen DHT beantwortet werden kann. Ist dies der Fall, befindet sich Gerät B im gleichen AS. Die aktuell gültige LTA wird an Gerät A zurückgeliefert, das mit der LTA die Daten direkt lokal versenden kann. Ist die Anfrage nicht erfolgreich, leitet der LNR die Anfrage an einen CNR des AS weiter (1b).

2. In der zweiten Phase wird mit Hilfe der globalen DHT der GGUID und damit das AS bzw. der Betreiber identifiziert, in dem sich Gerät B aktuell befindet. Die-

se Anfrage, die vom Ausgangs-CNR initiiert wird (2a), liefert in jedem Fall ein Ergebnis, da alle gültigen UID in der globalen DHT registriert sind. Das Ergebnis wird an den Ausgangs-CNR zurückgeliefert (2b), der zusammen mit dem zurückgelieferten GGUID und seiner eigenen Routingtabelle anschließend einen geeigneten Eingangs-CNR im Ziel-AS auswählt, an den die Anfrage abermals weitergeleitet wird (2c).

3. In der dritten Phase initiiert der Eingangs-CNR des Ziel-AS eine Anfrage in dessen lokaler DHT (3a), die dort ebenfalls eindeutig beantwortet werden kann (3b). Mit Hilfe der LTA überprüft der Eingangs-CNR anschließend durch das Versenden einer Ping-Nachricht (4a), ob Gerät B tatsächlich in diesem AS erreichbar ist. Verläuft dieser Test positiv (4b), wird direkt eine Antwort an Gerät A zurückgesandt (3c). Diese Antwort enthält den aktuell gültige GGUID und die LTA von Gerät B, so dass Gerät A direkt Daten dorthin versenden kann. Die direkte Antwort der Mapping-Anfrage an Gerät A ist möglich, da dessen LTA und GGUID in allen Anfragen mitgeführt werden.

Die Mapping-Anfrage ist ein eigenständiger Prozess, der vor der eigentlichen Kommunikation durchgeführt werden muss. Es werden keine Nutzdaten ohne gültige Locator-Informationen ausschließlich mit UID versendet, wie es bei verschiedenen anderen Mappingverfahren, beispielsweise LISP, der Fall ist. Dies hat den Vorteil, dass dazwischenliegende Knoten keine Nutzdaten zwischenspeichern müssen, bis die dafür entsprechenden Locator-Informationen aus dem Mapping System eintreffen.

Ein Nachteil des zweistufigen Lookup-Prozesses ist die Verzögerung bis die Nutzdaten gesendet werden können. Im ungünstigsten Fall, wenn Geräte in verschiedenen AS miteinander kommunizieren möchten, sind maximal drei Lookups in je einer One-Hop DHT notwendig. Nachdem der Lookup-Prozess abgeschlossen ist, sind aber keine weiteren Anfragen mehr nötig und die Pakete können dank der relativ kleinen Routingtabellen der CNR sehr schnell zum Ziel weitergeleitet werden. Echtzeitanwendungen wie Telefonie- oder Videokonferenz-Anwendungen, bei denen es auf geringe Latenzzeiten während der Übertragung ankommt, profitieren dabei von diesem Designkonzept. Ändert sich der Locator eines Gerätes, so wird der neue Locator durch eine Update-Meldung bandintern allen aktuellen Kommunikationspartnern mittels gesetztem LUP-Flag mitgeteilt. Zeitgleich erfolgt auch eine Aktualisierung des Mapping-Eintrags im Mapping-System.

Um die Anzahl der Lookups und damit die Latenzzeit bis zur Sendung der Nutzdaten zu verringern, können Mechanismen wie Caching verwendet werden, die jedoch einen Kompromiss zwischen Aktualität der Mapping-Information und Lookup-Zeit bedeuten. Dabei speichert ein CNR eine hohe Zahl an Mapping-Einträgen, aber nur für kurze Zeit. Ein LNR speichert weniger Mapping-Einträge, diese jedoch für längere Zeit. In den Mapping-Einträgen können TTL-Werte für das Caching in CNR und LNR vorgeben werden. Auf einem LNR werden aber nur Mapping-Einträge von Geräten aus fremden AS gespeichert, damit die Mapping-Information der lokalen Geräte immer aus der DHT erfragt werden muss, und daher aktuell ist. Im Gegensatz zur Variante ohne Caching, bei der maximal drei Lookups für einen Mapping-Eintrag nötig sind, sind mit Caching im ungünstigsten Fall fünf Lookups nötig. Dies ist der Fall, wenn ein CNR in der globalen DHT in Phase zwei eine veraltete Information aus seinem Cache ausliefert. Die

vermeintliche lokale DHT beantwortet die Suchanfrage schließlich mit einem Fehler. Bei der anschließenden zweiten Anfrage wird durch ein Flag signalisiert, dass keine Cache-Informationen verwendet werden dürfen und in jedem Fall ein DHT-Lookup erfolgen muss.

3.3.1.4. Lastabschätzung

Ein kritischer Punkt eines jeden Mapping-Systems ist seine Skalierbarkeit und Belastbarkeit. In HiiMap v1 betrifft dies im Besonderen die Last, welche die globale DHT verarbeiten muss, da diese bei allen Mapping-Anfragen involviert ist, deren Ziel-AS ungleich dem Quell-AS ist. Relevant für die Last, die jeden einzelnen CNR erwartet, sind dabei die Anfragerate der Mapping-Anfragen λ_{Q}, die Datenrate R und der benötigte Speicherplatz S für die Mapping-Einträge, für die er zuständig ist. Um diese Werte abzuschätzen, werden folgende Annahmen getroffen, die auf Verkehrsstatistiken des Leibnitz Rechenzentrums (LRZ) und Prognosen zur zukünftigen Teilnehmerzahl im Internet basieren:

- Anfragerate pro Gerät[1]: $\lambda_{\mathrm{G}} = 0,025\frac{1}{\mathrm{s}}$
- globale DHT mit $N_{\mathrm{CNR}} = 100.000$ CNR
- Anzahl Geräte: $N_{\mathrm{G}} = 20 \cdot 10^9$ registrierte Geräte
- CNR Cache-Hit-Rate[2]: $\alpha = 0,5$
- Paketgröße: $L_{\mathrm{P}} = 200$ Byte pro übertragenem Datenpaket
- $n_{\mathrm{Q}} = 2$ Datenpakete für eine Anfrage an die DHT
- $n_{\mathrm{L}} = 2$ Datenpakete für einen DHT-Lookup

Aus den getroffenen Annahmen kann die Ankunftsrate der Mapping-Anfragen λ_{Q} berechnet werden, die jeder einzelne CNR im Schnitt bedienen muss. Für die Berechnung wird der ungünstige Fall betrachtet, bei dem die Mapping-Anfragen nicht bereits durch den lokalen Cache der LNR innerhalb eines AS beantwortet werden können. Jede Mapping-Anfrage wird daher an einen CNR geleitet, unabhängig von der Cache-Hit-Rate α. Ausgehend von der Anfragerate pro Gerät, der Anzahl der Geräte N_{G} und Anzahl an CNR N_{CNR} ergibt sich für die Anfragerate λ_{Q} eines einzelnen CNR:

$$\lambda_{\mathrm{Q}} = \frac{N_{\mathrm{G}} \cdot \lambda_{\mathrm{G}}}{N_{\mathrm{CNR}}} \tag{3.1}$$

Relevant für die Last der CNR ist neben der Anfragerate λ_{Q} auch die Paketrate λ_{P} und die damit verbundene Datenrate R der empfangenen und gesendeten Pakete. Diese sind

[1] Das LRZ betreibt ein AS mit der Nummer 12816 und geschätzten 40.000 angeschlossenen Geräten. Der DNS-Resolver des LRZ verarbeitet zu Spitzenzeiten etwa 1.000 Anfragen pro Sekunde. Dies entspricht 0,025 Anfragen pro Sekunde pro Gerät.

[2] Messungen der DNS Cache-Hit-Rate in [JSBM01] ergeben $\alpha \in \{0,8..0,86\}$. Aufgrund der kleineren TTL-Werte von HiiMap v1 im Vergleich zu DNS wird ein geschätzter Wert von $\alpha = 0,5$ verwendet.

jedoch von der Cache-Hit-Rate abhängig. Im Falle eines Cache-Hit beim Ausgangs-CNR muss dieser nur $n_{\mathrm{Q}} = 2$ Pakete verarbeiten (Anfrage- und Antwortpaket). Ein DHT-Lookup ist damit nicht nötig. Bei einem Cache-Miss muss der Ausgangs-CNR $n_{\mathrm{L}}+n_{\mathrm{Q}} = 4$ Pakete verarbeiten. Für die Paketrate λ_{P} ergibt sich damit:

$$\lambda_{\mathrm{P}} = \left[\alpha \cdot n_{\mathrm{Q}} + (1-\alpha) \cdot (n_{\mathrm{L}} + n_{\mathrm{Q}})\right] \cdot \lambda_{\mathrm{Q}} \tag{3.2}$$

Die Paketgröße L_{P} mit 200 Byte ist eine Abschätzung, die sich aus den notwendigen Feldern des Paketinhaltes zusammensetzt. Neben dem Paketheader mit Quell- und Ziel-UID, den jeweiligen Locator-Informationen und Informationen zum Forwarding müssen auch alle aktuell gültigen Locatorwerte übertragen werden. Ausgehend von im Schnitt fünf Locator-Einträgen pro Gerät ergibt sich eine Paketgröße von etwa 200 Byte. Zusammen mit der Paketrate λ_{P} lässt sich die Datenrate R berechnen, die für das Beantworten von Mapping-Anfragen pro CNR benötigt wird:

$$R = \lambda_{\mathrm{P}} \cdot L_{\mathrm{P}} = \left[\alpha \cdot n_{\mathrm{Q}} + (1-\alpha) \cdot (n_{\mathrm{L}} + n_{\mathrm{Q}})\right] \cdot \frac{N_{\mathrm{G}} \cdot \lambda_{\mathrm{G}}}{N_{\mathrm{CNR}}} \cdot L_{\mathrm{P}} \tag{3.3}$$

Ausgehend von den obigen Annahmen kann die Anfragerate λ_{Q} und die Datenrate R für jeden CNR quantitativ bestimmt werden. Für λ_{Q} ergeben sich 5.000 Anfragen pro Sekunde, die Datenrate R beträgt 24 Mbit/s. Die Datenrate R ist dabei sehr pessimistisch abgeschätzt, da die Paketgröße der Anfragepakete an die DHT kleiner sind als die Antwortpakete. Der für die Mapping-Einträge benötigte Speicherplatz S beträgt dabei pro CNR 40 MByte, ausgehend von einer angenommenen Größe der Mapping-Einträge von ebenfalls 200 Byte. Bereits bei aktuellen Hardwarearchitekturen ist mit diesen Werten ein reibungsloser Betrieb sichergestellt. Zukünftige Hardware wird noch um ein vielfaches leistungsfähiger sein, weshalb HiiMap v1 in Bezug auf die Last der globalen DHT als praktikabel angesehen werden kann.

In dieser Abschätzung wird besonders die Eigenschaft der One-Hop DHT sichtbar, Suchanfragen in nur einem Schritt auflösen zu können. Würde ein anderes DHT-System zum Einsatz kommen, beispielsweise Chord mit einer Suchpfadlänge von $1/2 \cdot (\log_2 N_{\mathrm{CNR}})$ und einer daraus resultierenden Paketzahl $n_{\mathrm{L,Chord}} = 1/2 \cdot (\log_2 N_{\mathrm{CNR}})+1$, entspräche dies einer Datenrate $R_{\mathrm{Chord}} = 82,43$ Mbit/s, die eine deutlich höhere Grundlast darstellt.

3.3.2. HiiMap Lösung 2: Mapping mit Regionen

Mit HiiMap v1 wurde ein Konzept eines Mapping-Systems auf der Grundlage von zwei DHT-Stufen vorgestellt. Im Falle einer globalen DHT, an der alle Betreiber beteiligt sind, setzt dieses Konzept eine Vertrauensbasis zwischen den Betreibern voraus. In einem globalen Zusammenschluss vieler Teilnetze kann davon jedoch nicht ausgegangen werden. Mit dem zweiten Lösungsansatz für ein Mapping-System, HiiMap v2 genannt, wird eine Weiterentwicklung von HiiMap v1 vorgestellt. Diese basiert ebenfalls auf einer DHT, ist jedoch nicht auf globale Vertrauensbeziehungen zwischen den Betreibern angewiesen [HSK+09].

3.3.2.1. Problematik einer globalen DHT

Da die Mapping-Einträge in einer DHT mit Hilfe einer Hash-Funktion pseudozufällig auf alle Teilnehmer verteilt werden, ist zwangsläufig jeder Betreiber für das Abspeichern von Mapping-Einträgen der anderen Betreiber verantwortlich. Daraus erwächst eine Vertrauensproblematik, da Betreiber die Macht besitzen, Mapping-Anfragen für Geräte bestimmter anderer Betreiber zu blockieren oder sogar umzuleiten, indem falsche Informationen zurückgeliefert werden. Es gibt keine Möglichkeit ein derartig bösartiges Verhalten zu erkennen, wenn keine gemeinsame Vertrauensbasis vorhanden ist.

Eine Vertrauensbasis zwischen Betreibern existiert auf Basis untereinander abgeschlossener Verträge. Derartige Verträge existieren jedoch nur zwischen Betreibern, die ihre AS miteinander verschaltet haben und darüber Daten austauschen und weiterleiten. In der globalen DHT in HiiMap v1 sind hingegen alle Betreiber weltweit vertreten. Dabei ist es praktisch unmöglich, gegenseitige Verträge zwischen allen globalen Betreibern zu schließen, um eine gemeinsame Vertrauensbasis zu schaffen.

Es gibt mehrere Möglichkeiten, das Vertrauensproblem trotz der Verwendung von DHT zu umgehen. LISP-DHT beispielsweise schlägt vor, die Identifier betreiberabhängig an die angeschlossenen Geräte zu verteilen. Damit kann sichergestellt werden, dass ein Betreiber nur solche Identifier verteilt, für dessen Bereich er laut Zuordnungsvorschrift der DHT auch zuständig ist. Da sich bei diesem Vorschlag aber der Identifier ändert, sobald der Betreiber gewechselt ist, steht dies im Gegensatz zum Konzept des sich nicht ändernden UID in HiiMap. Die Vorteile eines Locator/Identifier-getrennten Systems bezüglich Mobilität sind darüber hinaus nicht mehr gegeben.

Es wäre denkbar, dass ein Betreiber *immer* für den Bereich seiner vergebenen Identifier zuständig ist, auch wenn das dazugehörige Gerät in den Verantwortungsbereich eines anderen Betreibers gewechselt hat. Diese Lösung birgt jedoch zwei Risiken: Zum einen hat der Betreiber die alleinige „Macht“ über den Mapping-Eintrag zu einem von ihm erstellten Identifier und kann potentiell hohe Gebühren für den Mapping-Dienst verlangen, sollte das dazugehörige Gerät den Betreiber wechseln. Bleibt zum anderen der Mapping-Dienst gebührenfrei, könnte dies unter Umständen Folgen für den Betreiber haben. Wenn das gewechselte Gerät sehr populär ist, entstehen sehr viele Mapping-Anfragen für dieses Gerät. Der Betreiber würde für diese zusätzliche erhöhte Last keine Entschädigung bekommen. Diese Problematik existiert auch in HiiMap v1, da die Aufrufrate von Geräten im Internet nicht gleich verteilt ist. Sie ist abhängig von den Inhalten und Informationen die diese Geräte zur Verfügung stellen. Sind Inhalte besonders populär, werden diese Geräte häufiger angefragt.

Eine andere Möglichkeit, um die Gültigkeit eines Identifiers für die Lebensdauer des Gerätes sicherzustellen, ist die Aufteilung des Werte-Bereichs der DHT. Dabei ist immer der Betreiber, in dem sich das Gerät aktuell befindet, für dessen Mapping-Eintrag zuständig. Dies ist jedoch nur möglich, wenn der neue Betreiber einen CNR in die DHT einfügt, der anhand seines GGUID den entsprechenden Wertebereich verwaltet, in dem sich der Identifier befindet. Da der Wertebereich vor und nach dem entsprechenden Identifier nach wie vor zum alten Betreiber gehört, würde dies zu einem extrem hohen administrativen Aufwand führen.

Eine weitere Möglichkeit zur Vermeidung des Vertrauensproblems wäre ein vollstän-

dig zentrales Mapping-System. Da das Mapping-System jedoch immer angefragt wird, sobald zwei Geräte miteinander kommunizieren, wäre die Last auf ein zentrales System extrem hoch und würde darüber hinaus einen einzelnen Ausfallpunkt schaffen. Da der Namensraum des UID keiner Hierarchie folgt, ist auch ein dezentrales System mit einer Baumstruktur ähnlich DNS, das die Last auf die einzelnen Äste verteilt, nicht praktikabel.

3.3.2.2. Mapping-Regionen

Die Eigenschaft einer DHT, Inhalte nach einem festen Abbildungsschema auf die Teilnehmer der DHT zu verteilen, stellt eine besondere Herausforderung bezüglich Vertrauenswürdigkeit dar. Dennoch ist eine DHT aufgrund ihrer Skalierbarkeit, Robustheit und Erweiterungsfähigkeit eine vielversprechende Option für ein leistungsfähiges Mapping-System. HiiMap v2 setzt daher auf eine hybride Lösung bestehend aus mehreren regionalen DHTs und einer zentralen globalen Verwaltungsstelle, um die Vorteile beider Systeme zu vereinen. Wie auch in HiiMap v1 gibt es in HiiMap v2 zwei Hierarchiestufen, wobei die Regionen die untere darstellen und die zentrale Instanz die obere darstellt. HiiMap v2 gehört damit ebenfalls zur Gruppe der TLMS, die obere Hierarchiestufe ist jedoch als DMB realisiert.

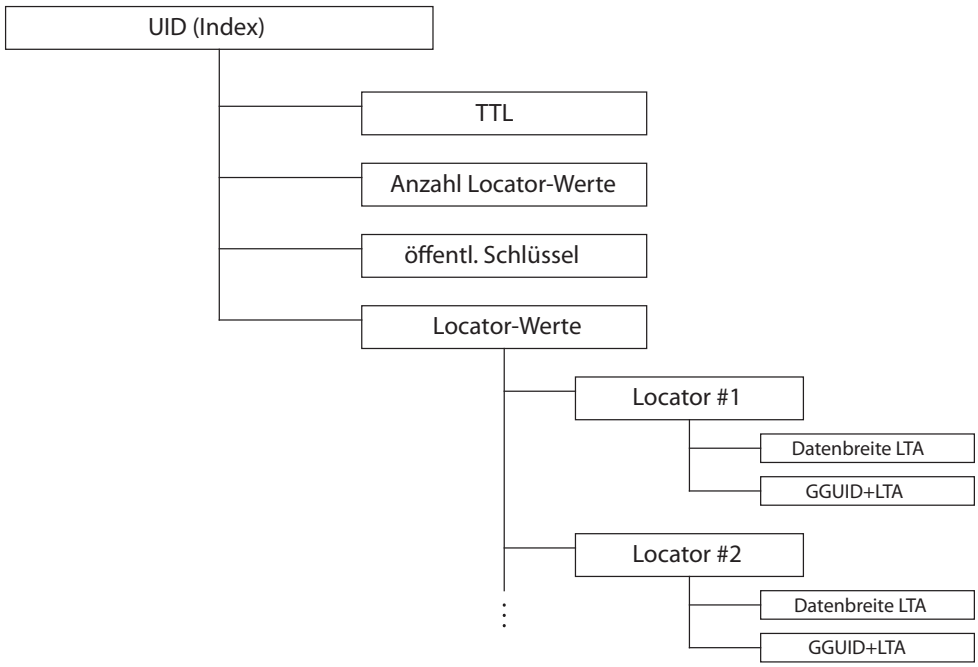


Abbildung 3.9.: Mapping-Eintrag in einer Region

Im Gegensatz zu HiiMap v1 werden in HiiMap v2 die Mapping-Anfragen vollständig von einer bestimmten Region beantwortet. Das bedeutet, dass zwischen dem lokalen und dem globalen Mapping-Eintrag nicht mehr differenziert wird und sowohl LTA als auch GGUID mit einer Anfrage übermittelt werden. Der entsprechende regionale Mapping-

Eintrag ist in Abbildung 3.9 dargestellt. Um die Region zu identifizieren, die für das Mapping einer bestimmten UID zuständig ist, wird ein so genanntes Regionales Präfix (RP) eingeführt. Das RP ist 8 Bit breit und wird vor die UID gesetzt (vgl. Abbildung 3.10). Es besteht aber keine feste Kopplung einer UID an ein bestimmtes RP. Ein Gerät wird nach wie vor ausschließlich mit seiner UID identifiziert, das RP ist nur eine zusätzliche Information, um die für das Mapping zuständige Region aufzufinden.

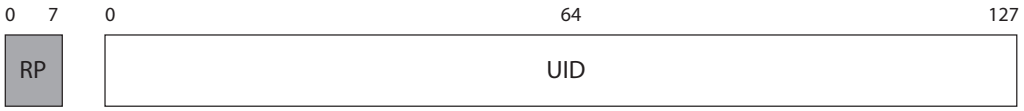


Abbildung 3.10.: UID mit Regionalem Präfix (RP)

Eine durch das RP identifizierte Region ist ein Zusammenschluss mehrerer Betreiber, welche die Mapping-Einträge aller Geräte verwalten, die in ihren jeweiligen ASen angeschlossen sind. Mit HiiMap v2 wird vorgeschlagen, den Zusammenschluss von Betreibern zu Regionen auf Länderbasis durchzuführen, da dies zwei wichtige Vorteile bringt. Zum einen zeigen Länder einen relativ stabilen Zustand, das heißt es ist verhältnismäßig selten, dass ein neues Land entsteht oder ein bestehendes sich auflöst. Aus diesem Grund sind nur selten Modifikationen des RP nötig. Ein weiterer Vorteil ist die einheitliche Rechtslage, die in einem Land herrscht. Auf diese Weise ist es für die Betreiber deutlich einfacher, gemeinsame Verträge zu schließen und eine Vertrauensbasis aufzubauen, da für jeden gleiches Recht gilt. Kleinere Länder mit ähnlicher Rechtslage können sich dabei auch zu einer Region zusammenschließen um den administrativen Aufwand zu reduzieren. Da jede Region aus gleichberechtigten Betreibern besteht, wird eine übergeordnete Verwaltungsinstanz pro Region benötigt. Lokale Registrierungsstellen wie die DeNIC, die aktuell für die DNS-TLD einzelner Länder zuständig sind, können derartige Aufgaben übernehmen und sind auch für die Vergabe des UID zuständig. Die Registrierungsstelle wird dabei von den Betreibern finanziert und untersteht einer staatlichen oder gemeinnützigen Aufsichtsbehörde.

Die Aufteilung auf länderbezogene Regionen beinhaltet jedoch auch Nachteile für Betreiber, die AS über mehrere Regionen verteilt betreiben. Da sie aus Sicht des Mapping-Systems in mehreren Regionen operieren und die jeweiligen Vorschriften einhalten müssen, führt dies zu einem administrativen Mehraufwand für diese Betreiber. Dennoch wird sich keine Granularität für die Zusammensetzung von Regionen finden lassen, die auf der einen Seite eine stabile Rechtsgrundlage bietet und auf der anderen Seite die spezifische Verbreitung einzelner Betreiber berücksichtigt.

Eine Region ist immer für das Auflösen von Mapping-Anfragen für Geräte verantwortlich, die in einem der zur Region gehörenden Betreiber initial registriert sind. Dies ist vergleichbar mit dem Home Location Register (HLR) im GSM Mobilfunk. Selbst wenn sich das Gerät temporär in einem AS aufhält, das zu einer anderen Region gehört, ändert sich das RP nicht und das Mapping wird von der Ursprungsregion übernommen. Nur wenn ein Gerät permanent zu einem Betreiber einer anderen Region wechselt, ändert sich die Zuständigkeit für den Mapping-Eintrag und damit auch das RP für dieses Gerät.

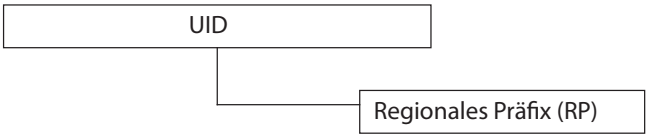


Abbildung 3.11.: Mapping-Eintrag in der GR

In HiiMap v2 wird das zu verwendende Speichersystem für Mapping-Einträge grundsätzlich nicht fest vorgeschrieben. Um die zu erwartende Last auf das Mapping-System zu bewältigen, schlägt HiMap v2 ebenfalls eine One-Hop DHT innerhalb einer Region vor. Dabei sind nicht, wie in HiiMap v1, die einzelnen LNR Teilnehmer der DHT, sondern jeder Betreiber stellt dafür dedizierte Knoten zur Verfügung. Da jedoch für externe Geräte, die Anfragen an die Region senden, die Topologie der DHT nicht bekannt ist, existiert ein so genannter Load Balancer (LB), der als Anlaufstelle für Mapping-Anfragen fungiert. Der LB kennt die aktuelle Topologie der DHT sowie die Auslastung der einzelnen Knoten und steht unter der Verwaltung der lokalen Registrierungsstelle. Er verteilt die eintreffenden Mapping-Anfragen an die einzelnen Knoten in Abhängigkeit von deren Auslastung. Damit der LB mit einem RP eindeutig identifiziert werden kann, besitzt er einen GGUID, auch wenn er nicht die Aufgaben eines CNR übernimmt. Für den GGUID werden die Felder der Betreiberkennung und der CNR-Kennung auf Null gesetzt. Der GGUID beinhaltet daher ausschließlich den Wert des RP. Kleinere Regionen mit einer geringen Anzahl an angeschlossenen Geräten können auch eine zentrale Datenbank als Mapping-System verwenden.

3.3.2.3. Globale Registrierungs-/Verwaltungsstelle (GR)

Zusätzlich zu den Regionen und deren lokalen Registrierungsstellen wird eine Globale Registrierungsstelle (GR) benötigt. Diese bietet drei Dienste an:

- Die GR speichert zu jeder UID das entsprechende RP. Ist das RP zu einer bestimmten UID nicht bekannt, wird die Anfrage zuerst an die GR gesendet, die den globalen Mapping-Eintrag (vgl. Abbildung 3.11) mit dem entsprechenden RP zurück liefert. Anschließend wird die entsprechende Region angefragt.
- Die GR ist für die Vergabe des UID und das Sicherstellen seiner Einzigartigkeit verantwortlich. Das Abwickeln der Registrierung eines neuen UID kann jedoch an die lokalen Registrierungsstellen delegiert werden.
- Die GR dient als Ankerpunkt für eine Zertifikats-Infrastruktur zusammen mit der in HiiMap integrierten PKI.

Um den reibungslosen Betrieb der ersten beiden Dienste sicherzustellen, muss die GR in hohem Maße ausfallsicher sein. Dies ist ohne großen Aufwand realisierbar, wenn die zu erwartende Last auf die GR überschaubar bleibt. Die heutigen DNS-Root-Server belegen dies. Dabei kann die GR, genau wie die 13 DNS-Root-Server (A-M), auf mehreren Knoten weltweit gespiegelt sein.

Die Realisierung eines weltweiten Ankerpunktes für eine Zertifikats-Infrastruktur stellt jedoch eine weitaus kompliziertere Aufgabe dar, da die GR unabhängig und neutral gegenüber allen teilnehmenden Geräten und Betreibern agieren muss. HiiMap v2 schlägt daher vor, die GR den Vereinten Nationen (UN) und nicht den Gesetzen eines bestimmten Landes unterzuordnen. Da bis auf wenige Ausnahmen alle Länder Teil der UN sind, können sie diese als gemeinsame Grundlage für eine Vertrauensbasis anerkennen. Mit der Arbeitsgruppe „Global Alliance for ICT and Development“ (GUID) besitzt die UN bereits eine Arbeitsgruppe, die sich um globale Fragen der Informations- und Kommunikationstechnik kümmert und der die GR unterstellt werden könnte. Darüber hinaus müssen nur Angelegenheiten von globalem Interesse von der GR behandelt werden, da regionale Sachverhalte in den einzelnen Regionen geklärt werden können.

Die GR beziehungsweise jede Instanz der GR besitzt, ebenfalls wie die LB der einzelnen Regionen, eine Kennung in der Form eines GGUID. Betreiberkennung und CNR-Kennung sind auf Null gesetzt und der Wert für die regionale Kennung wird auf den Wert 1 festgelegt. Mit der Anycast-Methode ist es möglich, dass jede GR-Instanz die gleiche GGUID besitzt. Durch Modifikation der Routingtabellen der CNR kann dabei immer die nächste Instanz der GR angefragt werden. Sollte eine Instanz ausfallen, werden die Anfragen anhand der Routingtabellen an eine weiter entfernte Instanz geleitet.

Abbildung 3.12 zeigt die Topologie des HiiMap v2 Mapping-Systems mit drei Mapping-Regionen und der GR. Jede Region besitzt einen LB als Anlaufstelle für Mapping-Anfragen.

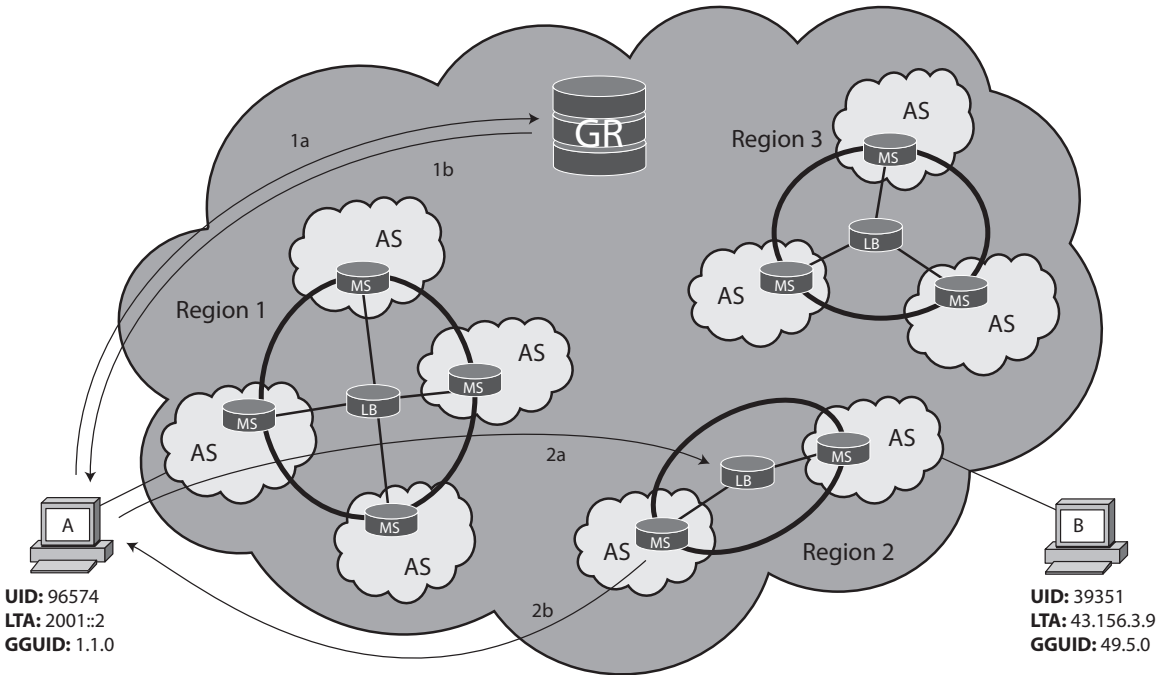


Abbildung 3.12.: Topologie und Ablauf einer Mapping-Anfrage in HiiMap v2

3.3.2.4. Mapping Lookup

HiiMap v2 verwendet im Gegensatz zu HiiMap v1 bei Anfragen an das Mapping-System einen iterativen Ansatz. Die Mapping-Anfrage eines Gerätes A für die Mappinginformation eines Gerätes B lässt sich in zwei Phasen einteilen, die in Abbildung 3.12 dargestellt sind. Da es sich um eine iterative Anfrage handelt, ist Gerät A in jeder Phase beteiligt:

1. Hat Gerät A für die gewünschte UID noch kein RP, wird zunächst eine Anfrage an die GR gestellt (1a). Diese teilt Gerät A das aktuell gültige RP zur angefragten UID mit (1b). Da sich das RP zu einer UID nur sehr selten ändert, speichert Gerät A das zu dieser UID gehörige RP lokal in einem Cache. Die TTL-Werte sind dabei in einer Größenordnung von Tagen und Monaten angesetzt.

2. Hat Gerät A das RP von der GR erhalten, beziehungsweise lokal in seinem Cache gespeichert, kontaktiert Gerät A direkt den Load Balancer der entsprechenden Region (2a). Dessen GGUID ist durch das RP implizit vorgegeben. Der Load Balancer initiiert eine Anfrage in der DHT der entsprechenden Region. Der zuständige Knoten liefert den Mapping-Eintrag von Gerät B direkt an Gerät A zurück (2b). Dieser enthält sowohl die GGUID des entsprechenden Betreibers als auch die LTA.

Auch in HiiMap v2 ist die Mapping-Anfrage ein eigenständiger Prozess, der vor der eigentlichen Kommunikation abgeschlossen sein muss. Durch die Verringerung der Anzahl der Lookup-Vorgänge im Vergleich von HiiMap v1 von im Normalfall maximal drei auf maximal zwei, wird die Verzögerung bis zum Senden der Nutzdaten reduziert. Einzig im Falle einer veralteten Cache-Information für das RP ereignen sich drei Lookup-Vorgänge. Dies geschieht jedoch nur in dem extrem seltenen Fall, dass ein Gerät die Region wechselt, während andere Geräte das alte RP noch in ihrem Cache halten. Im Fall von veralteten Cache-Informationen werden in HiiMap v1 jedoch fünf Lookup-Vorgänge benötigt.

3.3.2.5. Mapping-Proxy

Zusätzlich zu den Mapping-Regionen und der GR kann in HiiMap v2 ein Mapping-Proxy verwendet werden, der als Anlaufstelle für Mapping-Anfragen vor dem Lookup in der GR bzw. der Mapping-Region dient. Mit einem Mapping-Proxy ist es möglich, Geräte innerhalb eines privaten Netzes von außen zu verstecken, in dem sie nicht in dem öffentlich zugänglichen regionalen Mapping-System oder der GR abgespeichert werden. Innerhalb dieses privaten Netzes können die Locator-Werte aber über den Mapping-Proxy aufgelöst werden. Der Mapping-Proxy kann dabei selbst als kleines Mapping-System angesehen werden, der mit einer zentralen Datenbank sowohl das RP als auch den Locator zu einer bestimmten UID auflösen kann. Für Anfragen an einen UID, die der Proxy nicht aus seiner lokalen Datenbank beantworten kann, kommt der reguläre Lookup mit GR und Mapping-Region zum Einsatz, der dann vom Mapping-Proxy initiiert wird. Geräte, deren Locator nur über einen lokalen Mapping-Proxy gefunden werden kann, können dennoch mit Geräten, die öffentlich zugängliche Mapping-Einträge besitzen, Daten austauschen. Da bei einem Datenaustausch in den Paketen auch die UID sowie der Locator des Absenders übertragen werden, können auch Antwortpakete zugestellt werden.

3.3.2.6. Lastabschätzung

Die in Bezug auf die Verarbeitungslast kritische Komponente in HiiMap v2 ist die zentral realisierte GR. Für die Bestimmung der Last sind dabei wiederum die Anfragerate der Mapping-Anfragen λ_{Q}, die Datenrate R und der Speicherplatz S relevant. Zur Ermittlung der auf die GR wirkende Last werden bezüglich Teilnehmerzahl und Anfragerate die gleichen Werte wie für die Lastabschätzung von HiiMap v1 verwendet. Diese sind zusätzlich zu den neu getroffenen Annahmen aus Vollständigkeitsgründen nochmals aufgelistet:

- Anfragerate pro Gerät: $\lambda_{\mathrm{G}} = 0,025\frac{1}{\mathrm{s}}$
- Anzahl Geräte: $N_{\mathrm{G}} = 20 \cdot 10^9$ registrierte Geräte
- Instanzen der GR: $N_{\mathrm{GR}} = 100$ Instanzen
- RP Cache-Hit-Rate der Geräte: $\alpha = 0,9$
- Paketgröße: $L_{\mathrm{P}} = 50$ Byte pro übertragenem Paket
- $n_{\mathrm{Q\_GR}} = 2$ Pakete für Anfrage an die GR mit Antwort

Es wird davon ausgegangen, dass die Geräte einen Großteil der RP-Informationen zu den angefragten UIDs aus ihrem eigenen Cache bedienen können. Aus diesem Grund wird $\alpha = 0,9$ gewählt. Die noch verbleibende Anzahl an Anfragen wird auf alle Instanzen der GR paritätisch verteilt. Damit ergibt sich die Anfragerate λ_{Q} pro GR-Instanz von:

$$\lambda_{\mathrm{Q}} = \frac{N_{\mathrm{G}} \cdot \lambda_{\mathrm{G}}}{N_{\mathrm{GR}}} \cdot (1 - \alpha) \tag{3.4}$$

Zusammen mit der Paketgröße L_{P} und der Anzahl der Pakete pro Anfrage $n_{\mathrm{Q\_GR}}$ kann die zu erwartende Datenrate R pro GR-Instanz angegeben werden mit:

$$R = \lambda_{\mathrm{Q}} = \frac{N_{\mathrm{G}} \cdot \lambda_{\mathrm{G}}}{N_{\mathrm{GR}}} \cdot (1 - \alpha) \cdot L_{\mathrm{P}} \cdot n_{\mathrm{Q\_GR}} \tag{3.5}$$

Die Paketgröße ist dabei deutlich kleiner als bei der Lastabschätzung für HiiMap v1, da die GR nur das zur UID gehörige RP und nicht die gesamte Locator-Information zurückliefert.

Quantitativ ergibt sich pro GR-Instanz aus diesen Annahmen eine Anfragerate λ_{Q} von $500.000\frac{1}{\mathrm{s}}$ und einer Datenrate R von 400 MBit/s. Der Speicherplatz S, der pro Instanz benötigt wird, beläuft sich dabei ausgehend von einer Mapping-Eintragsgröße von ebenfalls 50 Byte auf 1 TByte.

Die zu erwartende Datenrate sowie der Speicherbedarf bewegen sich dabei in einer Größenordnung, die auch von heutigen Hardwarearchitekturen bereits ohne Probleme bewältigt werden kann. Lediglich die Anfragerate stellt eine große Herausforderung für die einzelnen GR-Instanzen dar. Die Autoren in [VC09] zeigen, dass ein DNS-Server mit aktueller Hardwareausstattung Anfrageraten von bis zu $15.000\ \frac{1}{\mathrm{s}}$ bearbeiten kann. Ausgehend davon müsste jede Instanz der GR aus 33 Servern der aktuellen Leistungsklasse bestehen, um die Anfragen verarbeiten zu können. Die Datenrate pro Server würde sich damit auf 12 MBit/s reduzieren.

3.3.3. Zusammenfassung der Mappingkonzepte

Mit HiiMap v1 und HiiMap v2 wurden zwei Konzepte für Mapping-Systeme vorgestellt, die beide auf DHT basieren und daher hohe Skalierbarkeit aufweisen. Der Ansatz einer globalen DHT birgt jedoch ernstzunehmende Probleme bezüglich den Manipulationsmöglichkeiten von Daten aufgrund von fehlenden Vertrauensbeziehungen der einzelnen Teilnehmer der DHT.

Mit der Einführung von Mapping-Regionen, dem dazugehörigen RP vor jeder UID und der GR als zentrale, der UN unterstellten Instanz kann dieses Problem behoben werden. Da die GR als Ankerpunkt für eine Zertifikats-Infrastruktur dient, die so viele Teilnehmer wie möglich anerkennen, kann eine gemeinsame Vertrauensbasis geschaffen werden. Regionen mit gemeinsamer Rechts- und Gesetzeslage unterstützen dabei den Zusammenschluss der Betreiber innerhalb dieser Region zu einem regionalen Mapping-System. Die Zertifizierung der öffentlichen Schlüssel der Geräte durch die GR unterstützt darüber hinaus die neu geschaffene Vertrauensbasis in die Kommunikation der Geräte untereinander.

Auch die Handhabung von Mapping-Einträgen hat sich zwischen beiden Versionen geändert. Während in HiiMap v1 eine lokale DHT nur die LTA und die globale DHT nur den GGUID für einen bestimmte UID speichern, werden in HiiMap v2 die kompletten Locator-Informationen in den Regionen gespeichert. Updates des GGUID durch Mobilität beeinflussen in HiiMap v2 die GR als globale Instanz damit nicht.

Tabelle 3.2 vergleicht die Eigenschaften beider Konzepte. Die Werte für die Anfragerate, die Datenrate und den Speicherplatz beziehen sich dabei jeweils auf eine Instanz der jeweiligen globalen Komponenten. In HiiMap v1 ist dies demnach ein CNR der globalen DHT und in HiiMap v2 eine Instanz der GR. Obwohl die zu erwartende Last in HiiMap v2 deutlich höher ist als in HiiMap v1, so ist dieses Konzept aufgrund seines Vorteils bei der Manipulationssicherheit der Mapping-Einträge und der geringeren Anzahl benötigter Lookup-Schritte HiiMap v1 vorzuziehen. Die weiteren Ausführungen der vorliegenden Arbeit beziehen sich ausschließlich auf das HiiMap v2-Konzept .

Tabelle 3.2.: Vergleich von HiiMap v1 und HiiMap v2

	HiiMap v1	HiiMap v2
Manipulationssicherheit	– –	++
Anfragerate	5.000 $\frac{1}{s}$	500.000 $\frac{1}{s}$
Datenrate	24 MBit/s	400 MBit/s
Speicherplatz	40 MByte	1 TByte
Lookup-Schritte (Cache-Miss)	3	2
Lookup-Schritte (Cache veraltet)	5	3

3.4. Namensschema für UID

Eine wichtige Aufgabe beim Entwurf einer zukünftigen Internet-Architektur, die auf einer Trennung von Locator und Identifier aufbaut, ist die Entwicklung von Methoden und Schemata zum Erzeugen, Benennen und Registrieren von Identifiern. Da die Kommunikation immer an den Identifier gebunden ist, wird dieser als Steuer- und Kontrollinformation in den Kommunikationsprotokollen und Packet-Headern verwendet. Aus diesem Grund bestehen Identifier üblicherweise aus einer binären Zeichenfolge fester Länge, die für Menschen nur schwer einprägsam sind. Um auf ein zweites Datenbanksystem ähnlich DNS zu verzichten, das menschenfreundliche Kennungen wie Geräte- und Domänennamen zunächst in Identifier umsetzt die anschließend vom Mapping-System in Locator-Werte aufgelöst werden, wird im Folgenden ein Namensschema für UIDs vorgestellt [SK11a, SK11b], das es erlaubt, die UID aus einem einprägsamen Klartext selbst zu generieren. Das Schema ist dabei so flexibel gestaltet, dass es im Sinne eines heterogenen Kommunikationsumfeldes nicht nur für Geräte geeignet ist, sondern auf beliebige Instanzen wie z.B. Inhalte, Informationen oder Personen ausgeweitet werden kann.

3.4.1. Anforderungen an die UID

Die ITU hat Grundanforderungen für Identifier in Locator/Identifier-getrennten Systemen in [ITU09] zusammengefasst. Die wichtigsten Punkte, die auch auf UIDs in HiiMap zutreffen, sind folgende:

- Ein UID muss vollständig von allen Routingaufgaben entbunden sein.
- Ein UID identifiziert eindeutig den logischen Endpunkt einer Kommunikation.
- Jedem UID kann mehr als ein Locator zugewiesen sein. Ein UID ändert sich nicht beim Wechsel eines Locators.
- Eine aufgebaute Verbindung ist an den UID gebunden und darf durch einen Locatorwechsel nicht getrennt werden.

Zusätzlich zu den Anforderungen der ITU werden für HiiMap weitere Eigenschaften benötigt, die wie folgt zusammengefasst werden können:

- Ein UID muss in der Lage sein, nicht nur Geräte adressieren zu können, sondern auch Informationen, Inhalte und Personen. Das Schema muss so gestaltet sein, dass weitere Erweiterungen möglich sind.
- Jeder Kommunikationsteilnehmer muss den UID seines Kommunikationspartners aus einem Klartext-Namen erzeugen können.
- Ein UID ist global eindeutig, es müssen aber temporäre UID ausgestellt werden können.
- Ein UID muss in einer Form vorliegen, so dass er in einer DHT gespeichert werden kann.

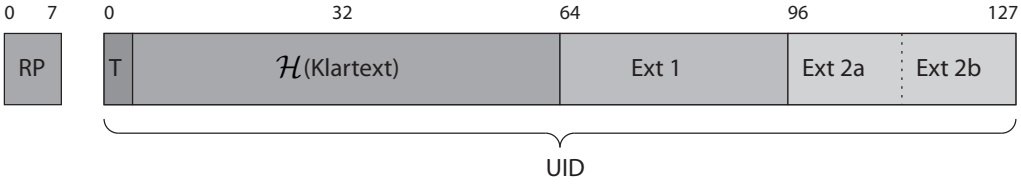


Abbildung 3.13.: Allgemeines UID-Schema mit vorangestelltem RP

Aus den definierten Anforderungen wird im Folgenden ein allgemeingültiges und umfassendes Namensschema für einen UID vorgestellt.

3.4.2. Allgemeines UID-Schema

Die Forderung der netzorientierten Schichten nach Binärwerten mit fester Länge wird unterstützt von den DHTs. Diese verwenden den UID als Indexwert für die Speicherung der Mapping-Einträge und benötigen dafür ebenfalls Binärwerte mit fester Länge. Die Verwendung einer Hash-Funktion ist daher naheliegend. HiiMap teilt dabei den UID in feste, vorher definierte Felder auf, die allen Teilnehmern bekannt sind. Bild 3.13 zeigt den allgemeinen Aufbau eines UID in HiiMap mit dem vorangestellten RP.

Das *Typ*-Feld T kennzeichnet, welches Objekt mit der UID identifiziert wird. In der vorliegenden Arbeit werden dabei konkret fünf Typen behandelt, die in Tabelle 3.3 aufgelistet sind. Dieser Abschnitt behandelt zunächst die Typen $T = 1$ (statische Geräte) und $T = 2$ (nichtstatische Geräte). Da das Typ-Feld die Bits mit dem höchsten Stellenwert repräsentiert, ist es möglich, im Mapping-System verschiedene interne Datenbanken für die einzelnen Objekte zu verwenden. Dies ist sinnvoll, da Mapping-Einträge für Inhalte, Informationen und Personen komplexer aufgebaut sind als für Geräte. Ausgehend von einem 128 Bit breiten UID entfallen 8 Bit auf das Typ-Feld, 56 Bit auf den Hash-Wert des Klartextnamens $\mathcal{H}$(Klartext) sowie je 32 Bit auf die Erweiterungsfelder Ext 1 und Ext 2 (a+b). Mit welchen Werten die einzelnen Felder beschrieben werden wird im Laufe dieser Arbeit konkret für drei Instanzen vorgestellt: Geräte, Informationen und Inhalte, sowie Personen. Im Rahmen dieses Abschnitts werden aber zunächst nur UID für Geräte behandelt.

3.4.3. UID für Geräte

In der aktuellen Internet-Architektur können ausschließlich Geräte direkt adressiert werden. DNS wird dazu verwendet, den Gerätenamen beziehungsweise den FQDN, in die IP-Adresse umzusetzen. Ähnlich der hierarchischen Struktur der Gerätenamen im DNS besteht auch das UID-Schema für Geräte aus einer Hierarchie, die jedoch auf zwei Ebenen beschränkt ist: eine globale und eine lokale Ebene. Während der globale Teil eine komplette Domäne, beispielsweise ein Unternehmen oder eine Universität identifiziert, wird der lokale Teil dazu verwendet, einzelne Geräte innerhalb dieser Domäne zu identifizieren. Der Begriff Domäne beschreibt in diesem Zusammenhang den Verwaltungsbereich, dem ein oder mehrere Geräte unterstehen und ist nicht mit einem AS, das ebenfalls als

Tabelle 3.3.: Übersicht der verschiedenen UID-Typen

T	Typ	$\mathcal{H}$(Klartext)	Ext 1	Ext 2a	Ext 2b
1	statische Geräte-UID	Domänenname, z.B. *meinefirma*	$\mathcal{H}$(lokaler Gerätename), z.B. *pc-arbeit-003*	0	für die Mapping-Anfrage
				$1..n-1$	standardisierte Werte für bekannte Dienste
				$n..2^{16}$	freie Werte für neue Dienste
2	nichtstatische Geräte-UID	Betreiberkennung, z.B. *betreiber123*	siehe Typ 1	siehe Typ 1	
3	nicht änderbare Informationen oder Inhalte	Name oder Bezeichnung des (Haupt-) Inhaltsobjekts	• $\mathcal{H}$(Name des Unterobjekts) • Nummer/ID: z.B. DOI • Wert 0: oberste Ebene der Objektbeschreibung	Repräsentationsformat	Version des Objekts
4	frei änderbare Information oder Inhalte	Name oder Bezeichnung des (Haupt-)Inhaltsobjekts	siehe Typ 3	siehe Typ 3	siehe Typ 3
5	Personen	Vor- und Nachname	Nummer vergeben durch Meldebehörde	0	Kontaktinformation der Person
				1	Locator des Geräts
				$2..n-1$	standardisierte Werte für bekannte Dienste
				$n..2^{16}$	freie Werte für neue Dienste
			Wert 0: für Verzeichnis	0	

Domäne bezeichnet wird, zu verwechseln. UID für Geräte werden im Folgenden noch weiter in zwei verschiedene Typen unterteilt, damit den Bedürfnissen sowohl von großen Unternehmen als auch von Privatpersonen, die mit Geräten ans Internet angeschlossen sind, Rechnung getragen wird.

3.4.3.1. Statische Geräte-UID

Der UID-Typ mit dem Wert $T = 1$ beschreibt einen sich niemals ändernden UID, der durch Anwenden einer Hash-Funktion auf den Klartext-Gerätenamen generiert werden kann. Da ein derartiger UID einem Gerät fest zugewiesen wird, ist ein Registrierungsprozess bei einer regionalen Registrierungsstelle erforderlich. Den statischen Geräte-UID verwenden hauptsächlich Unternehmen oder privaten Personen, die einen registrierten UID möchten, der ihren Geräten dauerhaft zugewiesen und vollkommen unabhängig vom Netzbetreiber ist.

$\boldsymbol{\mathcal{H}}$**(Klartext)**: Der Hash-Wert des Domänennamens des Klartext-Gerätenamens wird in dieses Feld eingesetzt. Ein Beispiel für einen derartigen Domänennamen eines Gerätes kann sein: *meinefirma* oder *vorname-nachname*. Dieser Wert wird auf dem Gerät über Systemeinstellungen oder Systemdateien konfiguriert und ist vergleichbar mit der Datei */etc/domain* unter Linux.

Ext 1: Dieses Feld wird mit dem Hash-Wert des lokalen Klartext-Gerätenamens aufgefüllt. Der lokale Gerätename ist dabei innerhalb einer bestimmten Domäne eindeutig. Ein Beispiel dafür lautet *pc-arbeit-003* und wird ebenfalls über Systemdateien wie */etc/hostname* unter Linux konfiguriert. Zusammen mit dem Domänennamen ist ein Gerät im Internet eindeutig identifizierbar.

Ext 2(a+b): Dieses Feld wird dazu verwendet, einen bestimmten Dienst auf dem Gerät zu identifizieren. Das Feld kann dabei mit heutigen TCP- oder UDP-Ports verglichen werden. Anwendungen oder Dienste, die auf eingehende Verbindungen oder Daten warten, müssen ihren gewünschten Ext 2-Wert beim Netzwerkstack registrieren, um Daten zu empfangen. Dabei sind, wie im heutigen Internet, einige Werte für bestimmte Dienste fest vorgegeben. Andere dagegen können frei gewählt werden. Wird ein Mapping-Eintrag für einen bestimmten Geräte-UID angefragt, so wird der Ext 2-Wert auf Null gesetzt, da das Gerät bereits mit den Hash-Werten aus globalen und lokalen Gerätenamen eindeutig identifiziert ist. Ext 2 wird erst bei der Kommunikation mit dem entsprechenden Gerät verwendet, um dort den richtigen Dienst zu erreichen.

Zum Schutz der Privatsphäre ist es möglich, für bestimmte lokale Gerätenamen keine globalen und regionalen Mapping-Einträge zu generieren. Dies ist sinnvoll für Unternehmen, die ihre interne Gerätestruktur nicht durch eine öffentlich zugängliche Datenbank sichtbar machen wollen. In diesem Fall ist ein Mapping-Proxy nötig, der interne Kommunikation zwischen den Geräten ermöglicht und als Schnittstelle zum öffentlichen Mapping-System dient.

Wird Ext 1 auf Null gesetzt, ist es möglich, einzelne zentrale Kontaktpunkte zu erzeugen. Ein Unternehmen kann damit z.B. einen Load Balancer verwenden, der als UID nur den Hash-Wert des Klartext-Domänennamens besitzt, um Anfragen an durch Ext 2 spezifizierte Dienste an entsprechende lokale Geräte weiterzuleiten.

3.4.3.2. Nichtstatische Geräte-UID

Im Gegensatz zum statischen Geräte-UID und der grundsätzlichen Idee vom sich nie verändernden Identifier, wird jedoch immer der Bedarf an einem nichtstatischen UID vorhanden sein. Dieser zeichnet sich dadurch aus, dass er nicht explizit registriert werden muss, einem Gerät für eine bestimmt Zeit zugewiesen werden und anschließend wieder an den Aussteller zurück geht, wenn er nicht mehr benötigt wird. Diesem Typ von UID wird der Wert $T = 2$ zugewiesen. Die relevante Benutzergruppe für nichtstatische UID sind beispielsweise private Haushalte mit einer DSL-, Kabel- oder Einwählverbindung und Geräten, die über einen Heimrouter[3] angeschlossen sind. Um im Internet Daten austauschen zu können ist eine eigene UID pro Gerät obligatorisch. Diese muss jedoch nicht zwingend einem Gerät fest durch die Registrierungsstelle zugewiesen sein, sondern kann auch nur temporär vergeben werden.

$\mathcal{H}$(**Klartext**): Bei einem nichtstatischen Geräte-UID wird der globale Teil des Gerätenamens vom Betreiber zugewiesen. Die Zuweisung erfolgt dabei an das Gerät, das aus Sicht des Betreibers den Netzabschluss darstellt und ist im Normalfall ein Heimrouter. Diese Zuweisung kann mit der Zuteilung eines IPv6-Präfixes im heutigen Internet verglichen werden. Für einen nichtstatischen UID kann dieser globale Teil entweder ebenfalls aus einem Klartextnamen mit Hash-Funktion bestehen, z.B. dem Hash-Wert von *betreiber123*, es kann jedoch auch eine binäre Zeichenfolge verwendet werden, die nicht aus einem Klartextnamen mit Hash-Funktion erzeugt wird und dem Betreiber von der Registrierungsstelle zugeteilt wurde. Dieser globale Teil ist so lange gültig wie ein gültiger Vertrag mit dem entsprechenden Betreiber besteht, auch im Falle von Mobilität und wechselnden Netzzugangspunkten. Wird der Betreiber gewechselt, wird dem entsprechenden Gerät vom neuen Betreiber ebenfalls ein neuer globaler Teil zugewiesen.

Ext 1: Genau wie beim statischen Geräte-UID besteht das Ext 1-Feld aus dem Hash-Wert des lokalen Gerätenamens. Ein lokaler Gerätename ist daher auch für den nichtstatischen Geräte-UID obligatorisch.

Ext 2 (a+b): Die Belegung des Ext 2-Feldes ist identisch mit der Belegung beim statischen Geräte-UID.

Es gilt zu beachten, dass der globale Teil des UID beim nichtstatischen Geräte-UID nicht zwingend aus dem Hash-Wert eines Klartext-Domänennamens bestehen muss, da für diesen UID-Typ die maschinelle Erreichbarkeit und Kommunikationsfähigkeit des Gerätes im Vordergrund steht. Soll ein bestimmtes Gerät durch einen einprägsamen Klartextnamen erreichbar sein, ist der statische UID vorzuziehen.

3.4.4. Registrierung und Zuteilung des Geräte-UID

Da jeder UID, unabhängig von seinem aktuellen RP per Definition weltweit eindeutig ist, muss sichergestellt werden, dass zu einem beliebigen Zeitpunkt jeweils nur ein Gerät eine entsprechenden Geräte-UID besitzt. Des weiteren muss sichergestellt sein, dass UIDs nicht zu bösartigen Zwecken gekapert und missbraucht werden können. Die in

[3] Als Heimrouter werden üblicherweise Geräte bezeichnet, welche die Internetverbindung auf Kundenseite terminieren und weiteren Geräten im Haushalt zur Verfügung stellen [Bun10]. Heimrouter können dabei auch Aufgaben eines lokalen Mapping-Systems, bzw. die eines Mappingproxy übernehmen.

HiiMap integrierte PKI hilft dabei mit geeigneten Methoden, derartige Probleme zu vermeiden. Anhand von Zustandsdiagrammen wird der Registrierungsvorgang im Folgenden für statische und für nichtstatische Geräte-UIDs beschrieben. Der Registrierungsvorgang beinhaltet auch die Zuteilung eines für den Datenaustausch im Internet gültigen Locators.

3.4.4.1. Statischer Geräte-UID

Der Registrierungsprozess für einen statischen Geräte-UID kann mit der Registrierung eines DNS-Domänennamens verglichen werden. Für das Zustandsdiagramm in Abbildung 3.14 werden dabei bestimmte Nachrichten benötigt, auf die Aktionen folgen. Werte in runden Klammern geben Parameter für Nachrichten an, die zwingend übermittelt werden müssen. Werte in eckigen Klammern sind optionale Parameter. SIG gibt an, ob eine Nachricht oder ein Parameter signiert ist.

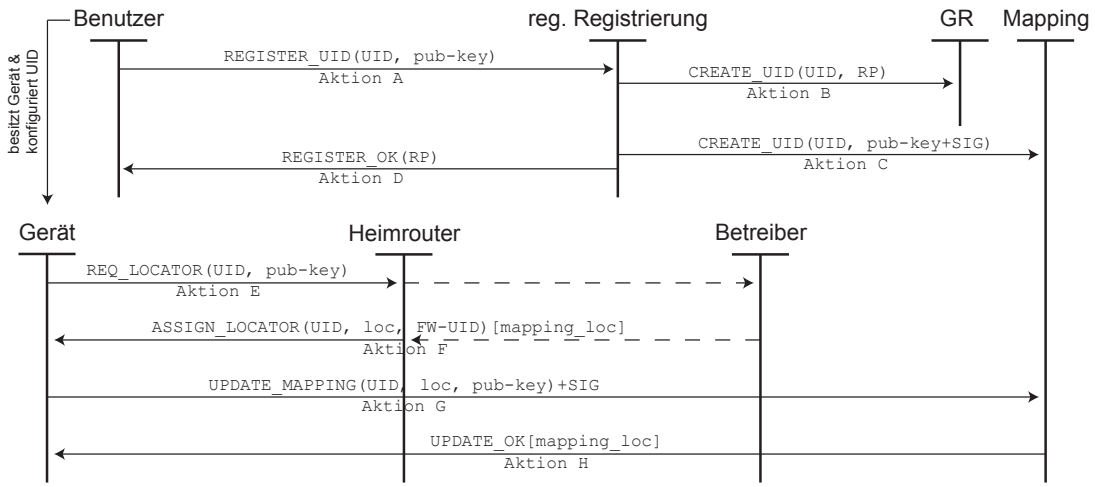


Abbildung 3.14.: Zustandsdiagramm Registrierung eines statischen Geräte-UID

Nachrichtentypen:

- `REGISTER_UID (pub-key)`: Der Besitzer eines Gerätes möchte für dieses eine statische Geräte-UID beantragen und sendet diese Nachricht zusammen mit dem öffentlichen Schlüssel des Gerätes (pub-key) an die regionale Registrierungsstelle. An dieser wird Aktion A ausgeführt.
- `CREATE_UID (UID, RP)`: Mit dieser Nachricht fordert die Registrierungsstelle das Erstellen eines Mapping-Eintrags in der GR an. Dazu wird das RP der aktuellen Region zusammen mit dem UID übermittelt und führt in der GR zu Aktion B.
- `CREATE_UID (UID, pub-key+SIG)`: Die gleiche Nachricht, jedoch mit anderen Parametern, wird an das regionale Mapping-System gesendet mit der Aufforderung,

den initialen Mapping-Eintrag anzulegen. Neben der UID wird der öffentliche Schlüssel des Geräts, der von der Registrierungsstelle signiert wurde, übermittelt. Im Mapping-System wird darauf Aktion C ausgeführt.

- `REGISTER_OK (RP)`: Diese Nachricht wird von der Registrierungsstelle zurück an den Benutzer gesendet, sobald `CREATE_UID` an das lokale Mapping-System gesendet wurde. Diese Nachricht bestätigt die korrekte Registrierung und löst beim Benutzer Aktion D aus.

Nachdem dem Gerät ein gültiger UID zugewiesen und der initiale Mapping-Eintrag erzeugt wurde, wird im nächsten Schritt aufgezeigt, wie das Gerät aktuelle Locator-Informationen bekommt. Der folgende Prozess ist dabei jedes Mal nötig, wenn das Gerät seinen Netzzugangspunkt wechselt.

- `REQ_LOCATOR (UID, pub-key)`: Diese Nachricht wird von einem Gerät an jeder aktiven Netzschnittstelle versendet, der noch kein Locator zugewiesen wurde. Das Gerät fordert mit dieser Nachricht einen gültigen Locator an und übermittelt seinen UID, sowie seinen privaten Schlüssel. Diese Nachricht wird dabei nicht gezielt als Unicast-Nachricht an ein bestimmtes Gerät (LNR des Betreibers oder Heimrouter) geschickt, sondern als Broadcast-Nachricht auf der Sicherungsschicht, da das zuständige Gerät noch nicht direkt per UID angesprochen werden kann. Diese Nachricht löst am zuständigen Gerät Aktion E aus.

- `ASSIGN_LOCATOR (UID, loc, FW-UID) [mapping_loc]`: Mit dieser Nachricht wird einen Locator zugeteilt. Als Parameter wird zusätzlich zum Locator `loc` (GGUID + LTA) der UID des anfragenden Gerätes übermittelt, damit die Nachricht das korrekte Gerät erreicht. Darüber hinaus wird mit dem Locator der FW-UID des nächsten LNR bzw. Heimrouters, der als Standard-Gateway dient, übertragen. Als optionaler Wert kann direkt der Locator des Mapping-Systems übermittelt werden, der jedoch nur notwendig ist, falls ein Mapping-Proxy zum Einsatz kommt. Ansonsten ist das regionale Mapping-System, bzw. dessen Load Balancer, durch das RP implizit bekannt. Am empfangenden Gerät folgt darauf Aktion F.

- `UPDATE_MAPPING (UID, loc, pub-key)+SIG`: Diese Nachricht wird an das regionale Mapping-System bzw. den explizit angegebenen Mapping-Proxy gesendet, um den Mapping-Eintrag für UID mit `loc` zu aktualisieren. Als Parameter wird der öffentliche Schlüssel des Geräts benötigt. Zusätzlich wird die gesamte Nachricht mit dem privaten Schlüssel signiert. Am Mapping-System wird Aktion G ausgeführt.

- `UPDATE_OK [mapping_loc]`: Diese Nachricht bestätigt die erfolgreiche Aktualisierung des Mapping-Eintrags. Als optionalen Wert kann wiederum der Locator des Mapping-Systems übermittelt werden, beispielsweise um die nächste Aktualisierung nicht über den Load Balancer, sondern direkt über einen Knoten der regionalen DHT durchzuführen. Am Gerät folgt darauf Aktion H.

Aktionen:

- `Aktion A`: Die Registrierungsstelle überprüft, ob der gewünschte UID noch verfügbar ist und erzeugt im positiven Fall die Nachrichten `CREATE_UID` für die GR und das Mapping-System. Darüber hinaus wird der öffentliche Schlüssel des Geräts von der Registrierungsstelle signiert.
- `Aktion B`: Die GR erstellt im globalen Mapping-System einen Eintrag, der zum UID das übermittelte RP speichert. Diese Aktion hat keine weitere Nachricht zur Folge.
- `Aktion B`: Das Mapping-System erstellt den initialen, leeren Mapping-Eintrag für den UID. Änderungen an diesem Eintrag sind nur erlaubt, wenn die entsprechende Nachricht mit dem zum öffentlichen Schlüssel passenden privaten Schlüssel signiert wurde. Diese Aktion hat ebenfalls keine weitere Nachricht zur Folge.
- `Aktion D`: Nach Empfang von `REGISTER_OK` kann der Benutzer sein Gerät mit dem zugewiesenen UID konfigurieren.
- `Aktion E`: Das für die Vergabe zuständige Gerät (Heimrouter oder LNR) überprüft optional den privaten Schlüssel und reserviert einen Locator-Wertebereich für dieses Gerät. Dabei wird nicht ein einzelner Locator-Wert reserviert, sondern ein ganzer Bereich, so dass das entsprechende Gerät selbst neue Locator-Werte verteilen kann. Anschließend wird die Nachricht `ASSIGN_LOCATOR` versendet.
- `Aktion F`: Das Gerät übernimmt den zugeteilten Locator-Wert und aktualisiert anschließend mit der Nachricht `UPDATE_MAPPING` seinen Mapping-Eintrag.
- `Aktion G`: Das Mapping-System überprüft die Signatur der Nachricht sowie den öffentlichen Schlüssel. Verläuft die Überprüfung erfolgreich, wird als Antwort die Nachricht `UPDATE_OK` gesendet.
- `Aktion H`: Der Aktualisierungsvorgang ist erfolgreich beendet. Das Gerät aktiviert die Netzschnittstelle für die Anwendungsebene und ist bereit zur Kommunikation.

Dieser Ablauf beschreibt die Registrierung als rein technischen Prozess. Eventuelle Abrechnungsschritte für kostenpflichtige Dienste, wie die Bereitstellung des UID durch die Registrierungsstelle, sind hier nicht berücksichtigt. Für derartige Schritte fällt unter Umständen auch noch die Erhebung personenbezogener Daten an, die im vorgestellten Zustandsdiagramm nicht integriert sind.

3.4.4.2. Nichtstatischer Geräte-UID

Dieser Geräte-UID-Typ wird benötigt, um Geräten temporär einen UID zuweisen zu können, ohne sich diese explizit von der Registrierungsstelle ausstellen lassen zu müssen. Dabei soll auch der nichtstatische Geräte-UID die gleiche Flexibilität bezüglich wechselnden Locator-Werten besitzen wie der statische Geräte-UID, jedoch mit der Ausnahme, dass er nicht permanent an ein bestimmtes Gerät gebunden ist. Der Zuteilungsprozess

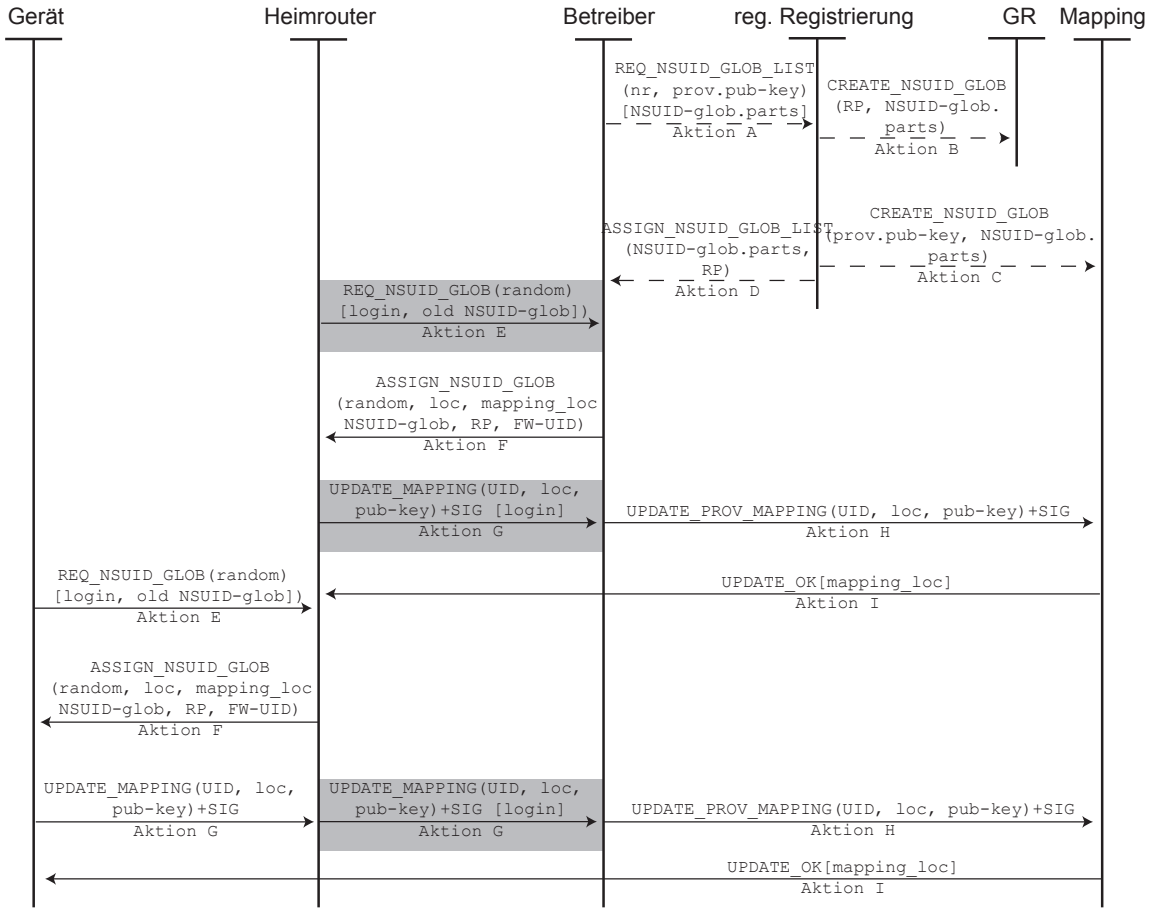


Abbildung 3.15.: Zustandsdiagramm Registrierung eines nichtstatischen Geräte-UID

ist dabei deutlich komplexer als beim statischen Geräte-UID, da die Registrierungsstelle nicht mit eingebunden ist und ein Missbrauch dennoch ausgeschlossen werden muss. Zu den bereits behandelten Nachrichten aus dem Zustandsdiagramm für den statischen Geräte-UID kommen für den nichtstatischen Geräte-UID noch die im Folgenden beschriebenen nachrichten hinzu (vgl. Abbildung 3.15). In der Abbildung grau hinterlegte Felder bedeuten, dass für diese Aktionen der Betreiber die Echtheit der Nachrichten sicherstellen muss, um das missbräuchliche Erschleichen von nichtstatischen Geräte-UIDs zu verhindern:

Nachrichtentypen:

- `REQ_NSUID_GLOB_LIST (nr, prov.pub-key) [NSUID-glob-parts]`: Bevor ein Betreiber nichtstatische UID verteilen kann, muss er bei der entsprechenden regionalen Registrierungsstelle eine Liste von globalen Teilen für nichtstatische UID beantragen. Der Betreiber kann dabei optional eine Liste mit gewünschten globalen Teilen angeben. Mit `nr` wird angegeben, wie viele Teile beantragt werden. Zusätzlich wird der öffentliche Schlüssel des Betreibers benötigt. An der Registrierungsstelle wird Aktion A ausgeführt.

- `CREATE_NSUID_GLOB (RP, NSUID-glob.parts)`: Mit dieser Nachricht fordert die Registrierungsstelle das Erstellen der Mapping-Einträge in der GR für die zugewiesenen nichtstatischen globalen Teile mit dem RP der entsprechenden Region an. Dies führt an der GR zu Aktion B.

- `CREATE_NSUID_GLOB (prov.pub-key, NSUID-glob.parts)`: Die Registrierungsstelle fordert das Erstellen der initialen Mapping-Einträge für die zugewiesenen globalen Teile der nichtstatischen Geräte-UIDs am regionalen Mapping-System an. Der lokale Teil (Ext 1) wird dabei auf Null gesetzt. Der öffentliche Schlüssel des Betreibers wird dabei ebenfalls übermittelt. Dies führt zu Aktion C am Mapping-System.

- `ASSIGN_NSUID_GLOB_LIST (NSUID-glob-parts, RP)`: Nach dem Erzeugen der Einträge im Mapping-System übermittelt die Registrierungsstelle dem Betreiber mit dieser Nachricht die zugeteilten globalen Teile für nichtstatische UID, sowie das gültige RP. Dies löst beim Betreiber Aktion D aus.

- `REQ_NSUID_GLOB (random) [login, old NSUID-glob]`: Mit diese Nachricht wird ein globaler Teil für einen nichtstatischen UID angefordert. Die Nachricht wird von jedem Gerät versandt, das keine fest zugewiesene statische UID besitzt. Da diese Nachricht auch missbraucht werden kann, um eine Vielzahl an UID für bösartige Zwecke zu erlangen, kann der Betreiber Login-Daten für dieses Kommando verlangen, um die Echtheit des Benutzers zu überprüfen. Im Gegensatz dazu kann der Benutzer einen vormals gültigen globalen Teil eines UID mitliefern, so dass der Betreiber ihm, falls möglich, den gleichen erneut zuweist. Um die Anfragen mehrerer Geräte voneinander zu unterscheiden, muss jedes Gerät einen Zufallswert mitliefern, damit die Antworten eindeutig zugestellt werden können. Die Nachricht wird, wie `REQ_LOCATOR` für statische UID, als Broadcast versandt und löst am zuständigen Gerät Aktion E aus.

- `ASSIGN_NSUID_GLOB (random, NSUID-glob, RP, loc, FW-UID, mapping_loc)`: Diese Nachricht weist dem anfragenden Gerät den globalen Teil der nichtstatischen UID, den Locator, das RP sowie die Forwarding-UID des nächsten Routers zu. Darüber hinaus wird der Locator des Mapping-Systems mitgeliefert. Der Locator ist für einen nichtstatischen Geräte-UID notwendig, da Update-Nachrichten zunächst an den Mapping-Proxy des Betreibers geleitet werden müssen und initial nur der Betreiber Mapping-Einträge im Mapping-System modifizieren kann. Diese Nachricht wird ebenfalls als Broadcast verschickt. Die Zufallsvariable hat dabei den gleichen Wert wie bei der Anfrage. Am empfangenden Gerät wird Aktion F ausgelöst.

- `UPDATE_MAPPING (UID, loc, pubkey)+SIG [login]`: Mit dieser Nachricht wird der Mapping-Eintrag aktualisiert. Die Nachricht wird entweder an den Mapping-Proxy des Betreibers oder direkt an das Mapping-System geschickt, initial jedoch immer zunächst an den Betreiber. Dieser kann Login-Daten verlangen, um einen Missbrauch durch gefälschte Aktualisierungsnachrichten Dritter zu verhindern. Am empfangenden Gerät wird Aktion G ausgeführt.

- `UPDATE_PROV_MAPPING (UID, loc, pub-key)+SIG`: Mit dieser Nachricht aktualisiert der Betreiber im Auftrag eines Geräts dessen Mapping-Eintrag. Der Betreiber authentifiziert sich durch Signatur der Nachricht. Am Mapping-System wird daraufhin Aktion H ausgeführt.

- `UPDATE_OK [mapping_loc]`: Diese Nachricht bestätigt die erfolgreiche Aktualisierung des Mapping-Eintrags. Als optionaler Wert kann wiederum der Locator des Mapping-Systems übermittelt werden, um die nächste Aktualisierung nicht über den Mapping-Proxy, sondern direkt durchführen zu können. Am Gerät folgt darauf Aktion I.

Aktionen:

- `Aktion A`: Die Registrierungsstelle weist dem Betreiber eine Liste mit der angeforderten Anzahl von globalen Teilen für nichtstatische Geräte-UID zu. Es wird versucht, dem Wunsch des Betreibers bezüglich der gewünschten Teile nachzukommen. Anschließend werden die Nachrichten `CREATE_NSUID_GLOB` an die GR und das regionale Mapping-System gesendet.

- `Aktion B`: Die GR erstellt im globalen Mapping-System die Einträge für die übermittelte Liste und speichert dazu das RP. Diese Aktion hat keine weitere Nachricht zur Folge.

- `Aktion C`: Das Mapping-System erstellt für jeden globalen Teil des zugewiesenen UID jeweils einen initialen Mapping-Eintrag mit Ext 1=0. Änderungen sind nur durch den Betreiber erlaubt, der die entsprechende Aktualisierungsnachricht mit dem korrekten privaten Schlüssel signieren muss. Diese Aktion hat ebenfalls keine weitere Nachricht zur Folge.

- `Aktion D`: Der Betreiber speichert die ihm zugewiesenen globalen Teile für nichtstatische Geräte-UID und kann diese jetzt Geräten zuweisen.

- `Aktion E`: Empfängt ein LNR diese Nachricht, überprüft er anhand der mitgelieferten Login-Daten, ob das Gerät berechtigt ist, den globalen Teil eines nichtstatischen Geräte-UID anzufragen. Im positiven Fall erfolgt die Zuteilung durch die Nachricht `ASSIGN_NSUID_GLOB`. Empfängt ein Heimrouter diese Nachricht, teilt dieser wiederum dem anfragenden Gerät seinen ihm zugewiesenen globalen Teil zu.

- `Aktion F`: Das Gerät besitzt jetzt, zusammen mit dem lokalen Gerätenamen, einen gültigen nichtstatischen Geräte-UID sowie einen Locator-Wert. Im Anschluss wird der Mapping-Eintrag aktualisiert.

- `Aktion G`: Der Mapping-Proxy des Betreibers überprüft die Login-Daten und aktualisiert den Mapping-Eintrag mit der Nachricht `UPDATE_PROV_MAPPING`. Ein Heimrouter leitet diese Nachricht an den Mapping-Proxy des Betreibers weiter und ergänzt die Login-Daten.

- `Aktion H`: Das Mapping-System überprüft die Signatur des Betreibers und aktualisiert den Mapping-Eintrag. Zusätzlich wird der öffentliche Schlüssel des Gerätes für zukünftige Änderungen freigeschaltet, so dass diese ohne den Mapping-Proxy des Betreibers erfolgen können. Als Antwort wird `UPDATE_OK` direkt zum betroffenen Gerät gesendet.

- `Aktion I`: Der Aktualisierungsvorgang ist erfolgreich beendet. Das Gerät aktiviert die Netzschnittstelle für die Anwendungsebene und ist bereit zur Kommunikation.

In dem klassischen Szenario mit Heimrouter fordert dieser zunächst vom Betreiber den benötigten nichtstatischen globalen Teil der UID an. Allen weiteren angeschlossenen Geräten, die ihre `REQ_NSUID_GLOB`-Nachrichten per Broadcast versenden, teilt der Heimrouter den ihm zugewiesenen globalen Teil der UID zu. Auch das initiale Aktualisieren der Mapping-Einträge für die angeschlossenen Geräte muss zunächst über den Heimrouter als Mapping-Proxy abgewickelt werden, da sich nur dieser direkt beim Betreiber authentifizieren kann und zunächst nur der Betreiber die Berechtigung zum Ändern der Mapping-Einträge besitzt.

Es ist möglich, Geräte mit statischem UID an einem Heimrouter mit nichtstatischem UID zu betreiben. Nach der Zuteilung eines Locators senden diese Geräte die Nachricht `UPDATE_MAPPING` entweder direkt an das Mapping-System, oder an den Heimrouter der ihnen als Mapping-Proxy zugeteilt wurde. Dieser leitet die Anfrage an den Betreiber weiter, der aber unter Umständen erkennt, dass es sich nicht um eine nichtstatische Geräte-UID aus seinem Bereich handelt. Die Anfrage wird dann ohne eigene Signatur durch den Betreiber direkt an das Mapping-System übergeben.

3.5. Endgeräte-Mobilität in HiiMap

Aufgrund der wachsenden Anzahl von mobilen und drahtlosen Geräten im heutigen Kommunikationsumfeld ist die Unterstützung von Mobilität bezüglich der Datenverbindung eines der wichtigsten Ziele beim Entwurf einer zukünftigen Internet-Architektur. Eine besondere Herausforderung besteht darin, Verbindungen über mehrere physikalische Netzzugangspunkte hinweg unterbrechungsfrei aufrecht zu erhalten. Darunter versteht man den Wechsel der aktiven Netzschnittstelle und eventuell auch den Wechsel der Zugangstechnologie, die mit der Vergabe einer neuen Adresse für die Paketvermittlung einhergehen. Als Beispiel dafür kann der Wechsel von WLAN zu Ethernet oder UMTS genannt werden. Mobilität an der gleichen Netzschnittstelle, wie im Mobilfunk oder innerhalb eines WLAN-Netzes, wird hier nicht betrachtet. Diese Art von Mobilität stellt für die Vermittlungsadresse kein Problem dar, da der Wechsel von Basisstationen oder Access Points von den unteren Schichten (Schicht 1 und 2) übernommen wird und für die Vermittlungsschicht transparent abläuft.

Im Gegensatz zum Mobile-IP-Konzept im heutigen Internet unterstützt HiiMap den Wechsel von Netzzugangspunkten, ohne dass eine zusätzliche Helferinstanz benötigt wird. Eine bestehende Verbindung wird dabei nicht unterbrochen. Der Wechsel des Netzzugangspunkts kann dabei beliebig oft erfolgen, im Moment des Wechsels müssen jedoch

kurzfristig sowohl die alte als auch die neue Netzverbindung existieren, um eine ununterbrochene Verbindung gewährleisten zu können. HiiMap profitiert dabei in hohem Maße vom Prinzip der Trennung von Locator und Identifier. Da jede Verbindung und Datenübertragung ausschließlich den UID als Endpunkt besitzt und an diesen gebunden ist, kann sich der Locator jederzeit ändern. Die netzorientierten Schichten verwalten dabei dynamisch den zu jeder aktiven Datenübertragung gehörenden UID und die entsprechenden Locator-Werte.

Jedes Gerät in HiiMap, das einen Netzzugangspunkt verlässt und einen neuen betritt, ist selbst dafür verantwortlich, dass alle benötigten Instanzen über die neuen Locator-Werte informiert werden. Dies geschieht in zwei Schritten:

1. Alle Kommunikationspartner über die neuen Locator-Werte benachrichtigen, mit denen aktive Verbindungen bestehen oder Daten ausgetauscht werden. Dies geschieht bandintern innerhalb der laufenden Kommunikation.

2. Aktualisieren des Mapping-Eintrags mit den neuen Locator-Werten.

Empfängt ein Kommunikationspartner während einer laufenden Datenübertragung eine Benachrichtigung über einen Locator-Wechsel eines bestimmten UID, so werden sofort alle nachfolgenden Daten für diesen UID an den neuen Locator gesendet.

In den folgenden beiden Abschnitten werden für die verschiedenen möglichen Formen von Mobilität, Roaming und Umzug, die notwendigen Schritte und Prozeduren aufgezeigt, damit eine reibungslose Kommunikation gewährleistet werden kann. Dabei wird an gegebener Stelle jeweils zwischen Geräten mit statischem und nichtstatischem UID differenziert.

3.5.1. Temporäre Mobilität (Roaming)

Temporäre Mobilität, auch *Roaming* genannt, ist die Form von Mobilität, die bei mobilen Endgeräten am Häufigsten auftritt. Grundsätzlich versteht man dabei jede Art von Wechsel des Netzzugangspunktes. Dieser kann durch einen tatsächlichen Standortwechsel ausgelöst werden, wenn beispielsweise ein Smartphone den Bereich eines WLAN verlässt und auf UMTS umschaltet. Roaming kann jedoch auch bei stationären Geräten auftreten, falls beispielsweise ein Arbeitsplatzrechner an eine andere Netzwerkdose angeschlossen wird. Ausschlaggebend für die temporäre Mobilität ist, dass sich das RP eines Geräts nicht ändert. Dabei ist immer die Heimatregion, d.h. die Region, von der das entsprechende Gerät das RP erhalten hat, für den Mapping-Eintrag verantwortlich, auch wenn das Gerät temporär den Netzzugangspunkt eines Betreibers einer anderen Region verwendet. Dies gilt sowohl für statische als auch für nichtstatische Geräte-UID.

In Abbildung 3.16 ist das Zustandsdiagramm im Falle von Roaming dargestellt. Es wird davon ausgegangen, dass Gerät A und Gerät B aktuell Daten austauschen und Gerät A seinen Netzzugangspunkt zu einem neuen Heimrouter ändert. Während die aktuelle Netzschnittstelle noch aktiv ist, sendet Gerät A auf der zweiten Netzschnittstelle die Anfrage zur Zuteilung eines gültigen Locators. Die Nachrichten und Aktionen zum Bezug eines neuen Locators sind identisch mit den bereits im Abschnitt 3.4.4.1 vorgestellten. Wurde der Locator zugeteilt, sendet Gerät A umgehend eine `UPDATE_LOCATOR`-Nachricht

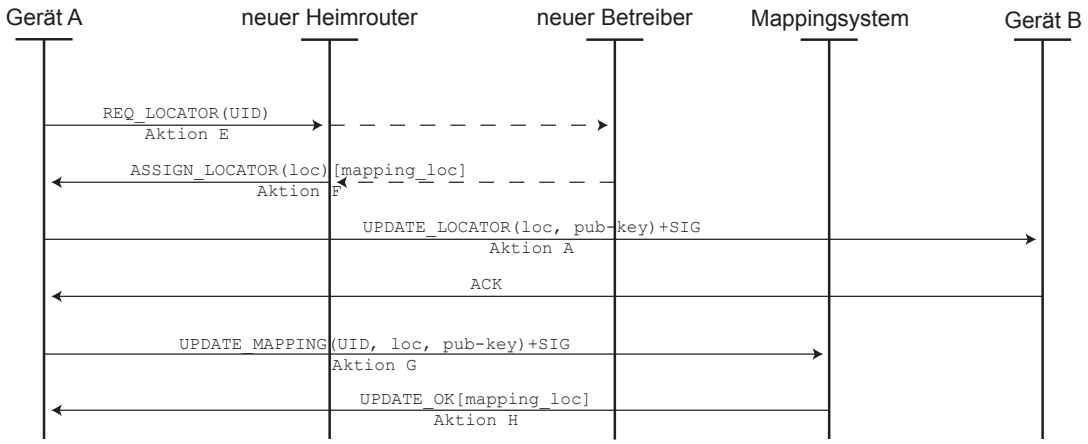


Abbildung 3.16.: Zustandsdiagramm Roaming für Geräte

an Gerät B, zusammen mit dem neuen Locator und dem öffentlichen Schlüssel des Geräts. Eine `UPDATE_LOCATOR`-Nachricht zeichnet sich durch das gesetzte LUP-Flag aus und kann auch Nutzdaten enthalten. Die gesamte Nachricht wird zusätzlich signiert, um zu verhindern, dass ein Angreifer den Datenverkehr von Gerät B zu Gerät A umleitet. An Gerät B wird daraufhin Aktion A ausgeführt. Dies umfasst die Übernahme des neuen Locator-Werts für Gerät A und das Bestätigen mittels `ACK` an diesen Locator. Bis die Bestätigung von Gerät B eintrifft werden die Daten von Gerät B an Gerät A noch über die alte Netzschnittstelle gesendet, die bis zum Eintreffen der Bestätigung auch aktiv bleiben muss, um eine Unterbrechung der Verbindung zu verhindern. Anschließend deaktiviert Gerät A die alte Netzschnittstelle und aktualisiert seinen Mapping-Eintrag. Die Nachrichten und Aktionen sind wiederum identisch mit denen in Abschnitt 3.4.4.1.

Soll Gerät A aufgrund von Multihoming mehrere aktive Netzschnittstellen besitzen, so wird die bestehende Schnittstelle nicht deaktiviert und stattdessen nur der Mapping-Eintrag mit den zusätzlichen neuen Locator-Werten aktualisiert. Eine Aktualisierung an die Gegenstelle ist in diesem Fall nicht nötig, da die bestehende Netzschnittstelle weiterhin aktiv bleibt.

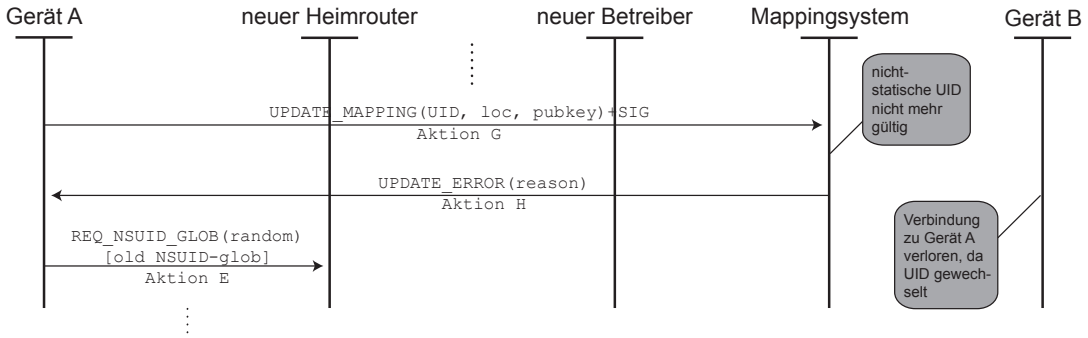


Abbildung 3.17.: Zustandsdiagramm Roaming im Fehlerfall

Bei der Aktualisierung von Mapping-Einträgen können auch Fehler auftreten. Da Mapping-Einträge, abgesehen von ihrer initialen Aktualisierungsphase, immer nur im Fall von Roaming aktualisiert werden müssen, werden die Fehlerfälle hier an dieser Stelle behandelt. Das Mapping-System kann beim Ausführen von Aktion G folgende Fehlerfälle feststellen (vgl. Abbildung 3.17), die dem Gerät mit `UPDATE_ERROR (reason)` mitgeteilt werden und an diesem Aktion H ausführen.

- `SYSTEM_N/A`: Das Mapping-System ist vorübergehend nicht erreichbar, beispielsweise wegen Überlastung oder aufgrund von Wartungsarbeiten. Aktion H am Gerät sieht in diesem Fall das periodische Wiederholen der Aktualisierung vor.
- `AUTHENTICATION_ERROR`: Die Signatur der `UPDATE_MAPPING`-Nachricht stimmt nicht mit dem im Mapping-System hinterlegten öffentlichen Schlüssel überein. In diesem Fall handelt es sich sehr wahrscheinlich um einen Manipulationsversuch. Das Mapping-System verbietet für eine gewisse Zeit weitere Aktualisierungsversuche. Sollte sich das Schlüsselpaar bewusst geändert haben, muss der neue öffentliche Schlüssel über die Registrierungsstelle im Mapping-System aktualisiert werden. Das Gerät weist beim Ausführen von Aktion H den Benutzer darauf hin, seinen Schlüssel im Mapping-System zu aktualisieren.
- `NSUID_GLOB_REVOKE`: Bei einem nichtstatischen Geräte-UID kann der Sonderfall eintreten, dass der Betreiber den globalen Teil des nichtstatischen UID für ungültig erklärt. Dies kann beispielsweise geschehen, falls der Vertrag mit einem Kunden ausläuft, dessen Geräten der entsprechende globale Teil eines nichtstatischen UID zugewiesen wurde. In diesem Fall schlägt die Aktualisierung beim Mapping-System fehl, da der Betreiber Aktualisierungsvorgänge für die entsprechenden globalen Teile sperren lies. Dies ist möglich, da der Betreiber bei nichtstatischen Geräte-UIDs immer über Administratorrechte der entsprechenden Mapping-Einträge verfügt. Aktion H am Gerät fordert anschließend einen komplett neuen globalen Teil für einen nichtstatischen UID an. Die Nachricht und entsprechende Aktion ist im Abschnitt 3.4.4.2 bereits beschrieben. Die Verbindung mit der Gegenstelle wird getrennt, da sich der UID ändert.

3.5.2. Dauerhafter Umzug

Während sich im Fall von Roaming das RP eines Gerätes nicht ändert, da es nur temporär seinen Netzzugangspunkt wechselt, so geht die zweite Form von Mobilität, der dauerhafte Umzug, mit einem Wechsel des RP einher. Dieser Fall tritt ein, wenn ein Unternehmen oder eine Privatperson mit den zugeordneten Geräten dauerhaft in eine andere Region umzieht. Dabei behalten Geräte ihren statischen UID unverändert bei, nur das RP für diesen UID ändert sich. Für nichtstatische Geräte-UID ist das nicht möglich, da die globalen Teile des entsprechenden UID an den Betreiber gebunden sind.

Da bei einem dauerhaften Umzug neben den Registrierungsstellen und den Mapping-Systemen der alten und neuen Region auch die GR involviert sind, erfolgen die nötigen Schritte aus technischer Sicht nicht automatisch, sondern müssen auf Anwenderebene eingeleitet werden.

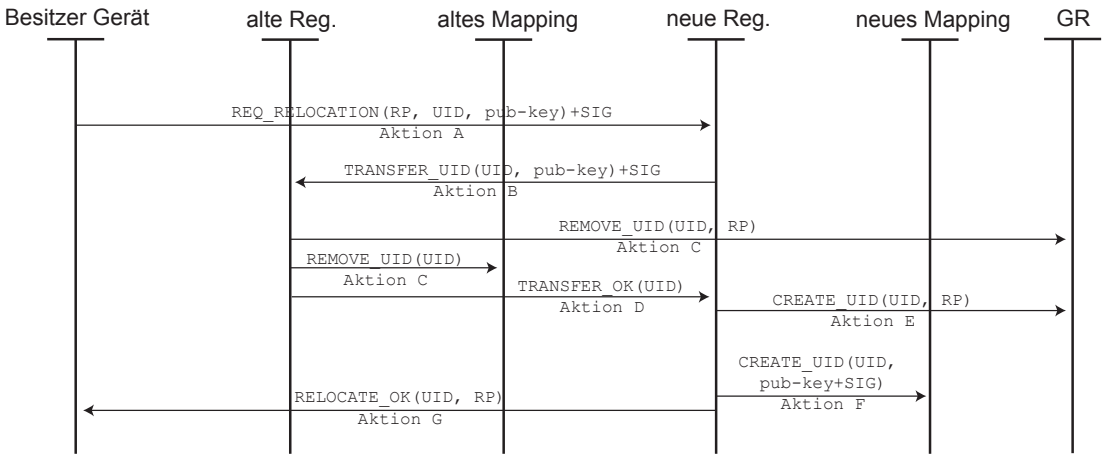


Abbildung 3.18.: Zustandsdiagramm für den dauerhaften Umzug eines UID

In Abbildung 3.18 ist das Zustandsdiagramm mit allen nötigen Nachrichten, den darauf folgenden Aktionen und beteiligten Instanzen für den Umzug eines statischen Geräte-UID dargestellt.

Nachrichten:

- `REQEST_RELOCATION (RP, UID, pub-key)+SIG`: Der Besitzer des Geräts beantragt mit dieser Nachricht bei der Registrierungsstelle der neuen Region den Umzug der statischen Geräte-UID. Die Nachricht enthält neben dem alten RP die umzuziehende UID sowie den öffentlichen Schlüssel des Geräts und wird mit dem privaten Schlüssel signiert. Es folgt Aktion A an der neuen Registrierungsstelle.

- `TRANSFER_UID (UID, pub-key)+SIG`: Nach dem Einleiten des Transfers wird diese Nachricht zur alten Registrierungsstelle geschickt, an der Aktion B ausgeführt wird. Die Nachricht wird mit dem öffentlichen Schlüssel der neuen Registrierungsstelle signiert.

- `REMOVE_UID (UID, RP)`: Das Mapping-System der alten Region sowie die GR werden angewiesen, die Mapping-Einträge für diesen UID zu entfernen. An den beiden Systeme wird Aktion C ausgeführt. Der Parameter RP wird nur für die Nachricht an die GR benötigt.

- `TRANSFER_OK (UID)`: Nach den Löschbefehlen teilt die alte Registrierungsstelle der neuen mit, dass der Umzug aus ihrer Sicht abgeschlossen ist. Darauf folgt Aktion D an der neuen Registrierungsstelle.

- `CREATE_UID (UID, RP)`: Die neue Registrierungsstelle fordert mit dieser Nachricht das Erstellen des Mapping-Eintrags in der GR an. Dazu wird das RP der neuen Region zusammen mit dem UID übermittelt. Dies führt zur Aktion E an der GR.

- `CREATE_UID (UID, pub_key+SIG)`: Die gleiche Nachricht, jedoch mit anderen Parametern, wird an das Mapping-System der neuen Region gesendet mit der Aufforderung, den initialen Mapping-Eintrag anzulegen. Neben der UID wird der öffentliche Schlüssel des Geräts, der von der Registrierungsstelle signiert wurde, übermittelt. Dies führt zu Aktion F am neuen Mapping-System.

- `RELOCATION_OK (UID, RP)`: Im letzten Schritt wird dem Benutzer der erfolgreiche Umzug, inklusive des neuen RP, mitgeteilt. Er führt Aktion G aus.

Aktionen:

- Aktion A: Die Registrierungsstelle der neuen Region leitet den Transfer der UID ein.

- Aktion B: Die alte Registrierungsstelle überprüft die Signatur der Nachricht und veranlasst das Löschen der Mapping-Einträge in der alten Region, sowie in der GR.

- Aktion C: Der Mapping-Eintrag für den mitgelieferten UID wird gelöscht. Diese Aktion hat keine weitere Nachricht zur Folge.

- Aktion D: Der entsprechende UID untersteht jetzt der neuen Registrierungsstelle.

- Aktion E: Die GR erstellt im globalen Mapping-System einen Eintrag, der zum UID das übermittelte RP speichert. Diese Aktion hat keine weitere Nachricht zur Folge.

- Aktion F: Das Mapping-System erstellt den initialen, leeren Mapping-Eintrag für den UID. Änderungen sind nur erlaubt, wenn die entsprechende Nachricht mit dem zum öffentlichen Schlüssel passenden privaten Schlüssel signiert wurde. Diese Aktion hat keine weitere Nachricht zur Folge.

- Aktion G: Der Benutzer konfiguriert sein Gerät mit dem neuen RP und kann anschließend damit erneut kommunizieren.

Geräte mit nichtstatischem UID versuchen im Falle eines dauerhaften Umzugs ihre Mapping-Einträge in der alten Region zu aktualisieren. Wenn der Vertrag mit dem alten Betreiber ordnungsgemäß gekündigt wurde, antwortet das Mapping-System mit dem Fehler `NSUID_GLOB_REVOKE`, woraufhin das Gerät den globalen Teil einer neuen nichtstatischen UID beantragt.

3.6. Neugestaltung des Protokollstacks

Aufgrund der Neugestaltung der Routing- und Forwarding-Architektur und der Einführung eines Mapping-Systems, ist für HiiMap eine Neugestaltung des Protokollstacks nötig.

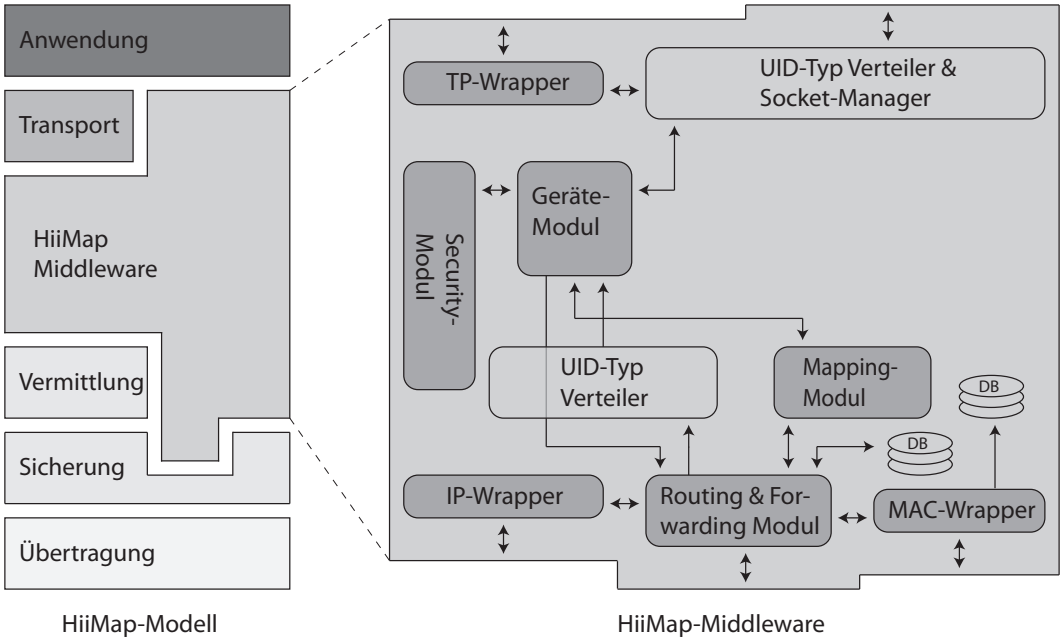


Abbildung 3.19.: HiiMap-Stack-Modell mit der HiiMap-Middleware

3.6.1. Middleware-Konzept

Für die HiiMap-Architektur wird der gesamte Netzwerk- und Protokoll-Stack auf Basis einer so genannten *Middleware* oder *Diensteschicht* neu gestaltet, um nicht ausschließlich Komponenten aus der Vermittlungsschicht für das geänderte Routing- und Adressierungsschema anzupassen. Eine Middleware steht dabei im Gegensatz zum OSI-Schichtenmodell, das eine strikte hierarchische Trennung der Aufgaben einzelnen Schichten vorsieht. Bereits heute wird das Schichtenprinzip nicht mehr streng eingehalten, indem mit so genannten Cross-Layer-Ansätzen auf Informationen nicht ausschließlich benachbarter Schichten zugegriffen wird. Dieser Zugriff gestaltet sich mitunter jedoch sehr kompliziert, da die Schnittstellen nur für die jeweils benachbarte Schicht vorgesehen sind. Darüber hinaus muss die Cross-Layer-Funktionalität meist auf Anwendungsebene realisiert werden. Sie steht daher anderen Anwendungen mit dem gleichen Bedürfnis nicht zur Verfügung.

Um die neue Architektur gleich von Beginn an flexibel gestalten zu können, ohne auf derartige Notlösungen zurückgreifen zu müssen, wird HiiMap im Protokoll-Stack als Middleware umgesetzt, die auf einzelnen funktionalen Modulen basiert (vgl. Abbildung 3.19). Jedes Modul stellt dabei Schnittstellen zur Verfügung, auf die andere Module sowie Applikationen direkt zugreifen können [RWH+09]. Funktionen, die von bestimmten Anwendungen besonders häufig verwendet werden, können hierbei als eigenes Modul in der Middleware realisiert werden [VMEK+08, VMW+09], was Mehrfachimplementierung der gleichen Funktion auf Anwendungsebene vermeidet.

3.6.2. Module der HiiMap-Middleware

Die Basisversion der HiiMap-Middleware für ausschließliche Kommunikation zwischen Geräten besteht aus mehreren funktionalen Modulen, deren Aufgaben im Folgenden erklärt werden. Zusätzlich sind auch einige Wrapper-Module nötig, welche die Kompatibilität mit bestehenden Anwendungen und der Sicherungsschicht gewährleisten.

Routing-& Forwarding-Modul Dieses Modul ist für die Verarbeitung der zu sendenden und empfangenden Daten zuständig und muss das Adressierungsschema des Netzes unterstützen, an welches das Gerät angeschlossen ist. Es erstellt den HiiMap-Header und übernimmt die Aufgaben der Vermittlungs- und Transportschicht. Dabei unterstützt es den gesicherten verbindungsorientierten Datenversand sowie den verbindungslosen Datagrammversand. Mithilfe des Protokoll-Felds im Packet-Header wird der entsprechende Datenversandmodus ausgewählt und mit den Flags gesteuert. Eingehende Datenpakete werden an den UID-Typ Verteiler übergeben, der sie an das entsprechende Modul weiterleitet. In einer eigenen Datenbank (DB) speichert dieses Modul die UID/Locator-Tupel aller offenen Verbindungen und Datenübertragungen. Im Falle eines Locator-Wechsels werden umgehend Aktualisierungsnachrichten versendet. Das Modul greift auf eine vorhandene Sicherungsschicht zur Datenübertragung zu und verwendet den HiiMap FW-UID als Adressierungsschema.

MAC-Wrapper Unterstützt die Sicherungsschicht ausschließlich ihr eigenes Adressierungsschema, erfolgt die Kommunikation des Routing- & Forwarding-Moduls mit der Sicherungsschicht über dieses Wrapper-Modul. Der MAC-Wrapper übernimmt dabei die Zuordnung von MAC-Adresse und UID und speichert diese in einer eigenen Datenbank, die mit dem ARP-Cache verglichen werden kann. Der FW-UID wird in diesem Fall nicht verwendet.

IP-Wrapper Ist das an ein Gerät angeschlossene Netz noch nicht HiiMap-fähig, so erfolgt jegliche Kommunikation des Routing- & Forwarding-Moduls über den IP-Wrapper. Dieser kapselt und versendet HiiMap-Datenpakete als IP-Pakete mit der Protokollnummer 143. Ebenso werden eingehende Pakete mit dieser Protokollnummer an das Routing- & Forwarding-Modul weitergereicht. Die einzelnen Möglichkeiten zur Kommunikation über IP werden in Kapitel 5 erläutert.

Security-Modul Dieses Modul ist für alle Funktionen zuständig, welche die Informationssicherheit der Daten betreffen. Es übernimmt die Ver- und Entschlüsselung von Datenpaketen sowie die Überprüfung und Erstellung von Signaturen. Über einheitliche Schnittstellen stellt das Security-Modul diese Funktionen allen anderen Modulen zur Verfügung.

Mapping-Modul Dieses Modul ist für sämtliche Funktionen zuständig die das Mapping-System und die logische Verbindung zwischen dem UID und Locator betreffen. Das Modul

bearbeitet Anfragen anderer Module an das Mapping-System und aktualisiert die Mappingeinträge des eigenen Geräts im Falle eines Locator-Wechsels. Antworten aus dem Mapping-System übergibt der Block an den entsprechenden anfragenden Funktionsblock zur Weiterverarbeitung. Der Mapping-Block speichert darüber hinaus in einem eigenen Cache die RP aller angefragten UID, um dadurch die GR zu entlasten.

Geräte-Modul Dieses Modul ist für die Kommunikation mit anderen Geräten sowie für die Bereitstellung der Schnittstelle für Dienste zuständig. Es verwaltet den UID des Geräts und erhält alle Datenpakete, die an diesen UID gesendet werden. Anwendungen, die Dienste anbieten und Daten empfangen möchten, müssen ihre Dienste-Kennung (Ext 2) an diesem Modul registrieren. Eingehende Anfragen werden an die zuständige Anwendung weitergeleitet bzw. abgewiesen, falls keine entsprechende Anwendung registriert ist.

UID-Typ-Verteiler (mit Socket-Manager) Der Typ-Verteiler existiert zweimal in der Middleware. Zum einen ist er dafür zuständig, eingehende Datenpakete anhand des UID-Typs an das entsprechenden Modul weiterzuleiten. Zum anderen stellt der Verteiler auch die Schnittstelle zu den Applikationen zur Verfügung, deren Verbindungsanfragen je nach UID-Typ ebenfalls an das entsprechende Modul geleitet werden. Letzterer übernimmt zusätzlich die Aufgabe des Socket-Managers. Dabei werden kommunizierenden Anwendungen eindeutige nummerierte Sockets zur Kommunikation zugewiesen. Eingehende Daten werden von den Modulen an diese Sockets zugestellt und von der Anwendung übernommen.

Für die Kommunikation zwischen Geräten entspricht die Socket-Nummer dem Ext 2-Feld des UID. Für ausgehende Daten wird vom Socket-Manager eine beliebige freie Nummer gewählt.

TP-Wrapper HiiMap erlaubt es mit dem TP-Wrapper zwischen Middleware und Anwendung noch bestehende Transportprotokolle und Schichten zu verwenden. HiiMap arbeitet in diesem Fall als reines Vermittlungsprotokoll. Dies erleichtert aus Anwendungssicht den Übergang von der aktuellen Architektur hin zu HiiMap, da vorhandene Schnittstellen weiter verwendet werden können. Der TP-Wrapper nimmt Datenanfragen von anderen Transportprotokollen an, korrigiert eventuelle Prüfsummen und übergibt die Daten an den Socket Manager.

3.7. Zusammenfassung

In diesem Kapitel wurde HiiMap vorgestellt, ein Konzept für eine zukünftige Internet-Architektur, die auf der Trennung von Locator und Identifier aufbaut. HiiMap besitzt als Besonderheit einen zweigeteilten Locator und ist daher auch für ein höchst heterogenes Kommunikationsumfeld geeignet. Der Identifier, in HiiMap mit UID bezeichnet, hat die vorteilhafte Eigenschaft, sich über die gesamte Lebensdauer des ihm zugewiesenen Geräts nicht zu ändern.

Es wurden zwei mögliche Realisierungen für Mapping-Systeme aufgezeigt, die einen unverzichtbaren Bestandteil einer auf der Trennung von Locator und Identifier basierten Internet-Architektur darstellen. Aufgrund der besseren Vertrauens- und Sicherheitseigenschaften wurde für weitere Arbeiten die Lösung HiiMap v2 ausgewählt, die auf Mapping-Regionen zusammen mit einer zentralen globalen Registrierungsstelle basiert.

Es wurde ein Schema zur Erzeugung und Benennung für den UID vorgestellt, das es erlaubt, den UID aus einem einprägsamen Klartext zu erzeugen und dabei alle technischen Anforderungen erfüllt um in den netzorientierten Schichten eingesetzt werden zu können. Dabei wurden auch die nötigen Schritte zur Registrierung und Zuteilung des UID behandelt.

Da eine Trennung von Locator und Identifier beste Voraussetzungen für mobile Endgeräte liefert, wurden Mechanismen aufgezeigt, wie Verbindungen und Datenübertragungen unterbrechungsfrei fortbestehen können, auch wenn ein Gerät in der Zwischenzeit seinen Netzzugangspunkt ändert.

Abschließend wurden notwendige Modifikationen des Netzwerk-Stacks aufgezeigt, die mit einem revolutionären Ansatz einher gehen. Dabei setzt Hiimap auf eine flexible Middleware mit einzelnen Funktionsmodulen, die miteinander interagieren können und jederzeit erweiterbar sind.

4. Erweiterung von HiiMap zu einem inhaltsorientierten Netz

In Kapitel 3 wurde HiiMap als Vorschlag für eine neue Internet-Architektur der Zukunft vorgestellt. Dabei wurde ausschließlich auf die Möglichkeit der Geräteadressierung eingegangen. Es zeichnet sich jedoch der Trend ab, dass die Adressierung von einzelnen Geräten im Internet zunehmend an Bedeutung verliert und stattdessen die Informationsbeschaffung und Kommunikation über das Internet immer mehr in den Vordergrund rückt [JST+09, Dan09].

In diesem Kapitel wird HiiMap daher zu einem inhaltsorientierten Netz erweitert, wobei die Möglichkeiten zur Geräteadressierung vollständig erhalten bleiben. Mit den konzipierten Erweiterungen erlaubt es HiiMap, Inhaltsobjekten jeder Art ebenso wie Personen einen eigenen UID zuzuordnen und mit Hilfe eines Informationsmodells zu beschreiben, so dass man darauf zugreifen kann. Um die Verteilung dieser Inhalte im Internet möglichst ressourceneffizient zu gestalten, wird die neuartige Idee der Inhaltsmobilität vorgestellt. Inhalten wird es damit ermöglicht, sich frei im Netz zu bewegen und immer dort vorgehalten zu werden, wo die Nachfrage nach diesen Inhalten am größten ist. Dazu ist zunächst eine Analyse der aktuellen Inhaltsverteilung sowie der damit verbundenen Kosten im aktuellen Internet notwendig, um ein Modell des Algorithmus zur Inhaltsmobilität zu entwerfen. Mittels einer hybriden Speicherlösung auf Basis eines Peer-to-Peer-Systems und dedizierten Speicher-Servern ist HiiMap zudem für den immer mehr aufkommenden *user generated content*, auch Web 2.0 genannt, gerüstet. Darunter versteht man Inhalte, die nicht von einer bestimmten Institution erzeugt werden, sondern direkt von den Endnutzern. Beispiele dafür sind Videoportale wie Youtube oder soziale Netzwerke wie Facebook. HiiMap bietet eine Lösung, um den aktuellen Problemen einer mangelnden Inhaltsunterstützung entgegenzuwirken. Dies erfolgt durch eine direkte Adressierung von Inhalten und die Bereitstellung eines Mechanismus, die Inhalte automatisch und effizient im Netz zu verteilen.

Die folgenden Abschnitte befassen sich zunächst mit der Definition von Inhalten und Informationen. Anschließend wird zusammen mit einem Informationsmodell für Inhalte ein Namensschema für den UID vorgestellt, dem Inhaltsobjekte zugewiesen werden können. Dieses Namensschema ist mit dem bereits eingeführten vollständig kompatibel. Ebenso werden UID für Personen nach dem gleichen Schema eingeführt, zusammen mit einem Informationsprofil, das es erlaubt, mögliche Kommunikationskanäle zu Personen zu definieren. Beide neuen Namensschemata werden begleitet von Verfahren zur Registrierung und Zuteilung des jeweiligen UID. Der vierte Abschnitt beschäftigt sich mit einem Nachschlagemechanismus, der in der Lage ist, kleine Fehler in den Klartexten, die zur Beschreibung von Inhalten und Personen verwendet werden, zu korrigieren. Im fünften Abschnitt werden Messungen zur Inhaltsverteilung im aktuellen Internet vorgestellt und ausgewertet, die für die Modellierung des Algorithmus zur Inhaltsmobilität benö-

tigt werden, der im sechsten Abschnitt vorgestellt und diskutiert wird. Dabei werden die Kosten für Übertragung und Speicherung von Inhalten aus der Sicht des Netzes optimiert. Der siebte Abschnitt stellt das Konzept eines hybriden Speichersystems vor, mit dem Inhalte abhängig von ihrer Popularität in verschiedenen Systemen kosteneffizient abgespeichert werden können. Im letzten Abschnitt werden die nötigen Erweiterungen der HiiMap-Middleware für die vollständige Unterstützung von Inhalten und Personen vorgestellt.

4.1. Definition des Inhaltsbegriffs

Unter dem Begriff Inhalt (englisch content) im Sinne eines inhaltsorientierten Netzes versteht man jegliche Art von informationstragenden Medien, die für den Benutzer einen Mehrwert bieten. Dazu gehören Musikdateien, Videos, Dokumente in verschiedenen Formaten, aber auch Webseiten, die wiederum aus einzelnen Seiten, Dateien und Informationsobjekten zusammengesetzt sind. Jede einzelne informationstragende Einheit wird als Inhaltsobjekt oder Informationsobjekt (IO) bezeichnet und aktuell im WWW mittels eines URI eindeutig identifiziert. Inhalte lassen sich dabei grob in die zwei verschiedenen Kategorien dynamisch und statisch einteilen. Zu den statischen Inhalten gehören Bilder, Grafiken und Videos, die sich nicht oder nur selten ändern. Diese Kategorie von Inhalten kann ohne größere Probleme gecached und über ein CDN verbreitet werden. Dynamische Inhalte dagegen, wie sie besonders in sozialen Netzwerken und Online-Shops auftreten, werden zur Laufzeit aus Datenbanken erzeugt und sind meist für den Benutzer personalisiert. Da diese Inhalte eine Gültigkeit von oft nur wenigen Sekunden haben, können Mechanismen wie Caching nicht angewandt werden.

Ein IO muss jedoch nicht ausschließlich ein digitales Objekt repräsentieren. Auch ein Gebäude oder eine Sehenswürdigkeit kann ein IO darstellen. Mittels eines geeigneten Informationsmodells können entsprechende Informationen über die Geschichte des Bauwerks, Öffnungszeiten und andere Eigenschaften als digital erreichbares IO repräsentiert werden. Ebenso sind auch Ressourcen eines Unternehmens wie z.B. Räume oder Inventar eigene IO, deren Eigenschaften als digital abrufbare Information verfügbar gemacht werden können.

Auch Dienste, die einem Benutzer einen Mehrwert bieten und aus verschiedenen Komponenten wie Datenbanken und Web-Server zusammengesetzt sind, können ein IO darstellen. Das Vorgehen der Dienstekomposition wird dabei als Service Oriented Architecture (SOA) bezeichnet. Ein Dienst zeichnet sich dadurch aus, dass nicht nur Daten und Informationen bereitgestellt werden, sondern diese nach Anforderungen des Benutzers zunächst in gewisser Weise verarbeitet werden. Ein Dienst kann beispielsweise eine Telefonkonferenz oder eine Suchmaschine sein, die je nach Benutzervorgaben einen virtuellen Gesprächsraum instanziiert oder Suchergebnisse anhand eines Suchbegriffs zurück liefert. Die zur Verfügung gestellten Funktionen und Leistungen eines Dienstes sowie die dafür nötigen Eingaben und die dafür verwendeten Schnittstellen können durch ein IO repräsentiert und der Dienst adressiert werden. Das in dieser Arbeit vorgestellt Informationsmodell beschränkt sich nur auf Medieninhalte, eine Erweiterung auf Dienste ist aber möglich.

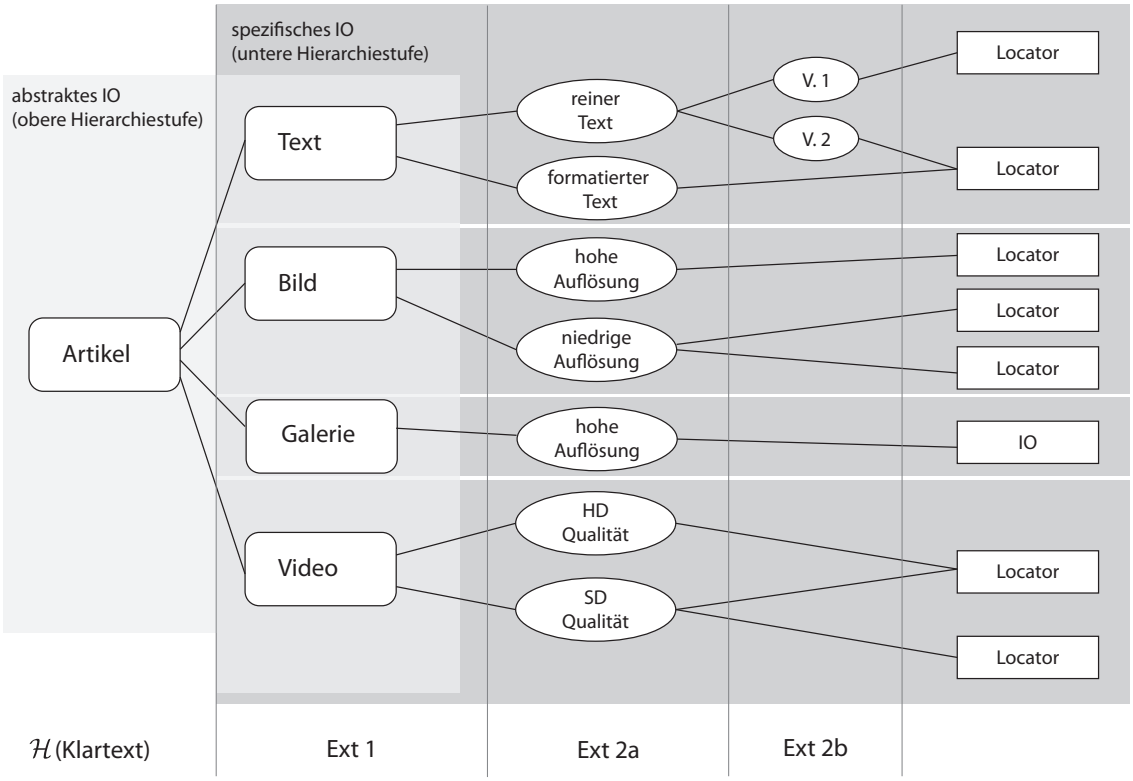


Abbildung 4.1.: Informationsmodell für Inhalte mit zugehörigen Teilen der UID

4.2. Adressierung von Inhalten

Im folgenden Abschnitt wird das Schema vorgestellt, mit dem auch Inhalten nach dem Paradigma des inhaltsorientierten Netzes jeweils ein eigener UID zugewiesen werden kann. Dabei wird das bereits in Kapitel 3 vorgestellte Schema für den UID um zusätzliche Typen erweitert. Zunächst wird jedoch das Informationsmodell vorgestellt, mit dem Inhalte in HiiMap repräsentiert werden und das die Erstellung eines eigenen UID für das entsprechende Inhaltsobjekt vereinfacht.

4.2.1. Informationsmodell für Inhalte

Das HiiMap Informationsmodell ist angelehnt an das in [Dan09] vorgestellte Informationsmodell des NetInf-Konzepts, das mehrere IOe miteinander verknüpft, die letztendlich auf ein Datenobjekt (DO) zeigen. Ein DO stellt dabei den Verweis auf die eigentliche Bit-Repräsentation des Inhalts dar. Im Gegensatz zu NetInf führt HiiMap dagegen in seinem Informationsmodell zwei Hierarchiestufen ein, die es erlauben, inhaltlich ähnliche IO in einem Modell zusammenzufassen.

Sowohl jedes IO als auch jedes DO in HiiMap besitzt einen eigenen UID. IO sind aufgrund der Hierarchie aufgeteilt in abstrakte IO und spezifische IO. Ein abstraktes IO in

der oberen Hierarchiestufe besteht aus einer allgemeinen Inhaltsbeschreibung, an die in der unteren Hierarchiestufen direkt beliebig viele spezifische IOe angefügt werden können. Diesen spezifischen Objekten können anschließend verschiedene Repräsentationen zugeteilt werden und diesen wiederum verschiedene Versionen, wobei jede Repräsentation und Version ebenfalls einen eigenen UID besitzt. Anschließend folgen die Locator-Werte, unter denen das entsprechende DO abgerufen werden kann, oder alternativ ein Verweis auf ein neues IO oder Informationsmodell. Für IO, die Medieninhalte darstellen, kann die Repräsentation beispielsweise Informationen über die Bitrate, Auflösung, verwendete Codecs oder die Dateigröße enthalten.

Da für Referenzen auf andere IO ausschließlich der UID zum Einsatz kommt und dieser vom Speicherort unabhängig ist, müssen IO rückwirkend nicht geändert werden, sollte sich der Speicherort der Inhalte eines referenzierten IO ändern. Darüber hinaus können auch mehrere IOe auf den gleichen Locator zeigen, wenn das entsprechende Gerät mehrere DO speichert. Durch die eindeutige Identifikation des gewünschten DO mittels des UID kann das speichernde Gerät das korrekte DO ausliefern. Ein beispielhaftes Infomationsmodell eines Zeitungsartikels ist in Abbildung 4.1 dargestellt. Als Beschreibungssprache für ein Informationsmodell kommt XML zum Einsatz. Anhang A.1 enthält die zur Abbildung 4.1 gehörige Beschreibung des Informationsmodells in XML.

IO werden in HiiMap ebenfalls im Mapping-System abgespeichert, jedoch enthält ein Mapping-Eintrag jeweils ein komplettes Informationsmodell, das mehrere IO mit den dazugehörigen UID beinhaltet. Im Vergleich zu NetInf kann die Anfragelast auf das Mapping-System dadurch reduziert werden, da zwei verknüpfte IO-Ebenen mit einer Anfrage bedient werden können.

4.2.2. UIDs für Inhalte

Jede inhaltsorientierte Netzarchitektur zeichnet sich dadurch aus, dass einem Inhaltsobjekt eine eigene und eindeutige Kennung zugewiesen wird, die sie im Idealfall vom physikalischen Speicherort vollständig entkoppelt. Das Konzept der Trennung von Locator und Identifier liefert dafür perfekte Voraussetzungen. HiiMap ermöglicht es, jedem beliebigen Inhaltsobjekt einen eigenen UID zuzuweisen, die auf das Informationsmodell des entsprechenden Objekts zeigt, dieses beschreibt und Informationen enthält, wie es beschafft werden kann. HiiMap verwendet für den Inhalts-UID das bereits in Kapitel 3.4 vorgestellte Namensschema [SK11a, SK11b], wobei die einzelnen Felder des UID neue Aufgaben bezüglich der Identifikation übernehmen. Diese korrelieren mit dem Aufbau des Informationsmodells, dessen Anzahl möglicher Stufen exakt mit der Anzahl der Felder einer UID übereinstimmt.

Ebenso wie für Geräte soll es auch für Inhaltsobjekte möglich sein, den UID aus einem einprägsamen Klartextnamen zu generieren, um möglichst ohne Suchmaschine direkt den gewünschten Inhalt zu finden. Aufgrund der Tatsache, dass durch die Verwendung von Hash-Funktionen der Klartextname exakt bekannt sein muss, um den korrekte UID zu erzeugen, wird das Mapping-System im Folgenden Abschnitt um eine auf N-Grammen basierende Funktionalität erweitert, die es erlaubt, kleinere Fehler im Klartextnamen zu korrigieren.

Ein UID für Inhalte übernimmt grundsätzlich die gleichen Aufgaben wie ein DOI. Un-

terschiede bestehen jedoch im Aufbau, sowie im verwendeten Datenbanksystem. Während bei einem DOI das Suffix beliebige Formen annehmen kann, existiert für den UID eine vorgegebene Struktur. Diese ist nötig, um auf ein zusätzliches Datenbanksystem zu verzichten und die zum UID gehörige Inhaltsbeschreibung ebenfalls im Mapping-System abzuspeichern.

Im Folgenden werden UID für Inhalte noch in weitere zwei Typen unterteilt, um auch den Bedürfnissen von Inhaltsobjekten, die von einer breiten Gemeinschaft bearbeitet werden dürfen, Rechnung zu tragen. Als RP eines UID für Inhalte wird die Region gewählt, in welcher der Ersteller des Inhalts beheimatet ist.

4.2.2.1. Nicht öffentlich änderbare Inhalte

Der UID-Typ mit dem Wert $T = 3$ wird einem IO zugewiesen, das zwar öffentlich abgerufen werden kann, jedoch ausschließlich vom Ersteller des Objekts modifiziert werden darf. Der Aufbau des UID spiegelt dabei ebenfalls die Hierarchie des im IO enthaltenen Informationsmodells wieder.

$\boldsymbol{\mathcal{H}}$(**Klartext**): Dieses Feld wird verwendet, um das abstrakte IO in einem Informationsmodell zu benennen. Aus einem sinnvollen und ausdrucksstarken Klartextnamen, der das gesamte Informationsmodell hinreichend beschreibt, wird der Hash-Wert gebildet, der diesem Feld zugewiesen wird. Obgleich die Findung eines Klartextnamens eine herausfordernde Aufgabe darstellt, kann angelehnt an das Informationsmodell aus Abbildung 4.1 beispielsweise *sueddeutsche-sport* als Klartextnamen verwendet werden, um einen dargestellten Zeitungsartikel zu benennen. Ebenso kann *sueddeutsche* als neues abstraktes IO in einem anderen Informationsmodell auf diesen Artikel referenzieren. Auf ähnliche Weise kann z.B. auch der Name eines Künstlers für dieses Feld verwendet werden. Das damit benannte Informationsmodell enthält dann als spezifische IO einzelne Musikstücke, Alben oder Filme.

Ext 1: Dieses Feld dient dazu, spezifische IO im Informationsmodell zu beschreiben, die semantisch dem abstrakten IO untergeordnet sind. Dabei können sowohl der Hash-Wert eines Klartextnamens als Wert für Ext 1 verwendet werden als auch andere Nummerierungsverfahren zum Einsatz kommen, wie beispielsweise der DOI. Aufgrund dessen variabler Länge muss dafür jedoch eine Hash-Funktion angewandt werden. Ebenso ist der Wert 0 für Ext 1 möglich. Damit wird explizit nur das abstrakte IO angesprochen, das jedoch seinerseits die möglichen Werte für Ext 1 auflistet, um auf das damit verbundene spezifische IO direkt zugreifen zu können. Auf diese Weise kann ein Benutzer IO abfragen, dem die spezifischen IO noch nicht bekannt sind.

Ext 2a: Mit diesem Feld wird zwischen den einzelnen Repräsentationsformaten eines spezifischen IOs unterschieden. Ext 2a ist der letzte obligatorische Wert eines UID, bevor im Informationsmodell ein Verweis auf ein DO oder ein weiteres IO erfolgen kann. Das Schema zur Benennung von Ext 2a kann dabei beliebig gewählt werden. Es bietet sich jedoch eine fortlaufende Nummerierung beginnend bei 1 an, um auf diese Weise auch im Falle von unbekannten möglichen Repräsentationsformen immer eine gültige Repräsentation zu bekommen. Anhang A.1 zeigt beispielhaft mögliche Repräsentationsformate, die in XML beschrieben sind. Besitzt bereits Ext 1 den Wert 0, so haben die Felder Ext 2a und Ext 2b keine Bedeutung mehr.

Ext 2b: Dieser Wert dient dazu, verschiedene Versionen einer bestimmten Inhaltsrepräsentation eines Objekts zu unterscheiden und einzeln zu identifizieren. Dennoch ist Ext 2b ein optionales Feld, das nicht zwingend verwendet werden muss, falls nur eine Version eines Objekts existiert. In diesem Fall wird Ext 2b auf Null gesetzt. Per Konvention zeigt der Wert 0 im Falle von mehreren vorhandenen Versionen immer auf die aktuelle Version. Im Informationsmodell enthält jede Versionsangabe immer einen Verwies auf entweder ein DO oder eine UID. Letzterer zeigt auf ein neues Informationsmodell.

Je nachdem, welche Teile des UID bekannt sind, liefert das Mapping-System die notwendigen Informationen zurück. Unbekannte Teile werden bei der Abfrage dabei immer auf Null gesetzt. Das Mapping-System liefert dann ein Informationsmodell zurück, das alle möglichen Werte für den auf Null gesetzten Wert enthält. Jede einzelne Version bzw. Repräsentation im Informationsmodell enthält darüber hinaus eine Signatur durch den Ersteller, wodurch Missbrauch vermieden werden kann.

4.2.2.2. Öffentlich änderbare Inhalte

Der UID-Typ mit dem Wert $T = 4$ wird einem IO zugewiesen, das im Gegensatz zu UID vom Typ $T = 3$ öffentlich geändert werden können. Dies bezieht sich jedoch nur auf das Einfügen neuer spezifischer IO oder Repräsentationen bzw. auf das Ändern derselben. Eine Änderung eines IO ist mit dem Erstellen einer neuen Version gleichzusetzen. Ein eigener UID-Typ ist dabei nötig, da zunehmend Informationen gemeinschaftsbasiert generiert und modifiziert werden. Dieses Phänomen wird auch als Crowdsourcing [LST07] bezeichnet. Dabei werden Aufgaben, in diesem Beispiel das Bereitstellen und Pflegen von Informationen, an die Masse der Benutzer ausgelagert. Ein Beispiel hierfür bietet die freie Wissensplattform Wikipedia, bei der jeder neue Artikel erstellen und vorhandene bearbeiten kann.

Die Aufgaben der einzelnen Felder sind identisch mit der Belegung für nichtöffentlich änderbare UID. Ein Unterschied besteht jedoch für die Handhabung der Mapping-Einträge. Das Mapping-System für einen UID des Typs $T = 3$ erlaubt ausschließlich Modifikationen durch den Ersteller, der diese Berechtigung durch eine Signatur mit seinem privaten Schlüssel nachweisen muss. Für den Typ $T = 4$ erlaubt das Mapping-System Änderungen von jedermann. Diese Änderung beziehen sich jedoch nur auf das Hinzufügen von neuen Versionen oder Repräsentationen. Der neue Eintrag im Informationsmodell enthält dazu eine Signatur mit dem privaten Schlüssel desjenigen, der die Änderung durchgeführt hat. Entfernt werden kann eine bestimmte Version eines IO jedoch nicht von jedermann, sondern ausschließlich von demjenigen der diese erstellt hat. Eine Ausnahme bildet der initiale Ersteller des Objekts, der auch Löschrechte auf fremde Einträge besitzt und das abstrakte IO im Informationsmodell signiert hat.

4.2.3. Mapping-Einträge für Inhalte

Mapping-Einträge für Inhalte unterscheiden sich grundlegend von Einträgen für Geräte. Während sich Einträge für Geräte untereinander ausschließlich in der Anzahl der hinterlegten Locatoreinträge unterscheiden, so ist die Speicherung eines gesamten Informationsmodells mit allen möglichen Repräsentationen und Versionen deutlich komplexer.

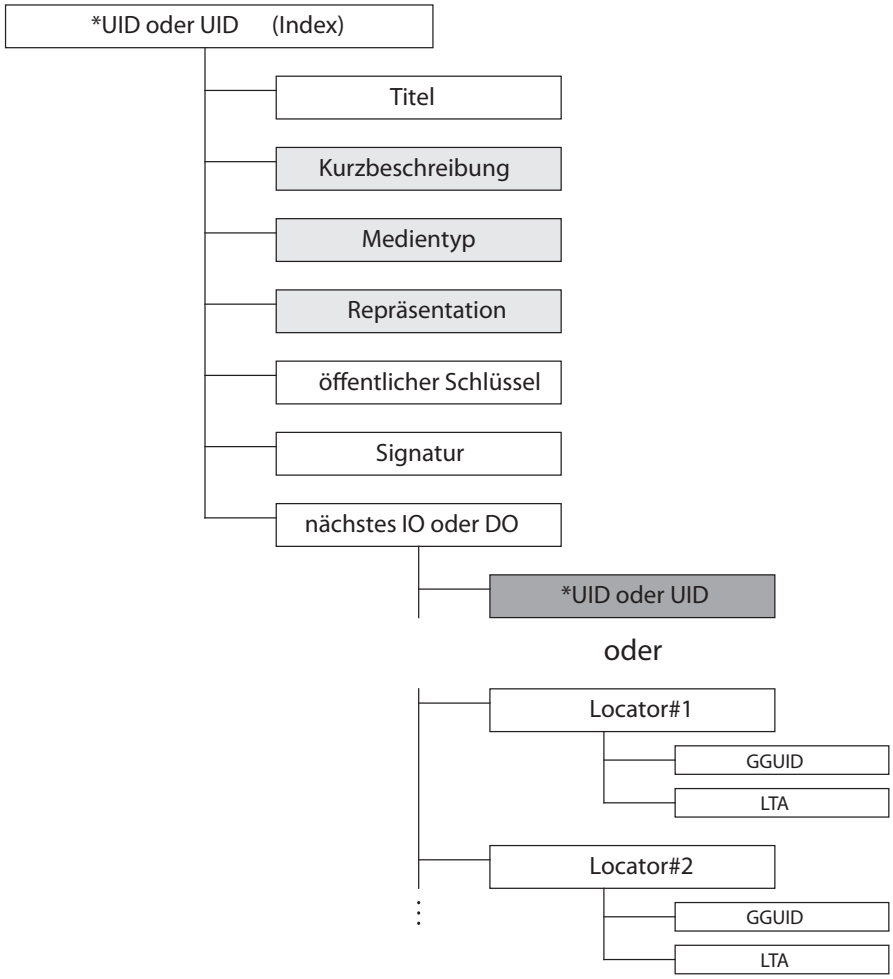


Abbildung 4.2.: Mapping-Eintrag des Informationsmodells für Inhalte mit *UID/UID

Da die dem Mapping-System zugrunde liegende Datenbank auf den jeweiligen Bedarfsfall optimiert wird, wird für Inhalte eine eigene Datenbank im Mapping-System benötigt. Da das Typ-Feld des UID die höchstwertigen Bits enthält und der Wertebereich klar definiert und abgetrennt ist, ist es ohne Aufwand möglich, den Verwaltungsbereich der DHT für verschiede Typen auf verschiedene Datenbanken aufzuteilen.

Um die Möglichkeit zu bewahren, abhängig von den angegebenen Werten des UID die jeweils relevanten Informationen mit geringem Rechenaufwand für die Datenbank zurückzuliefern, existiert pro IO ein eigener Mapping-Eintrag, der in Abbildung 4.2 dargestellt ist. Als Index-Wert für die DHT dient dabei der *UID bzw. der UID, je nachdem wie viele Werte angegeben werden. Ein *UID bezeichnet dabei die Sonderform eines UID, bei dem noch nicht alle Werte bekannt sind. Die Aufteilung der Mapping-Einträge in der DHT erfolgt dabei nur anhand des Typ-Feldes und des Hash-Werts des Klartextes. Damit wird sichergestellt, dass spezifische IO in der gleichen Datenbank gespeichert sind wie das dazugehörige abstrakte IO.

Abhängig davon, ob es sich um ein abstraktes oder spezifisches IO handelt, sind einige

Felder des Mapping-Eintrags optional. Diese sind in der Abbildung hellgrau dargestellt. So ist die Angabe des Medientyps ebenso wie die Repräsentation nur im Falle eines spezifischen IOs obligatorisch. Eine Kurzbeschreibung kann für beide IO-Arten angegeben werden, ist jedoch nicht zwingend erforderlich. Verpflichtende Werte jedoch sind die Angabe des Titels, des öffentlichen Schlüssels des Erstellers sowie dessen Signatur, und ein Verweis auf das nächste IO bzw. das letztendliche DO. Ein Verweis auf ein weiteres IO erfolgt durch die Angabe eines *UID oder eines vollständigen UID. Handelt es sich im Index um einen *UID, so werden dort alle möglichen Werte für das nächste nicht spezifizierte *UID-Feld gelistet. Ist der Index ein vollständig spezifizierter UID, kann auch ein *UID oder ein UID eines anderen IO folgen. Im Falle eines DO erfolgt die Angabe eines oder mehrerer Locator-Werte.

Wird auf ein weiteres IO referenziert (in Abbildung 4.2 dunkelgrau dargestellt), muss die Datenbank eine Sonderfunktion übernehmen. In diesem Fall wird der Datenbankeintrag des referenzierten IO zusätzlich an den Anfragenden ausgeliefert, jedoch ohne die darin enthaltene Information über das nächste IO oder DO. Dies ist nötig, um Informationen wie Titel oder mögliche Kurzbeschreibung zu den verknüpften IO bereitzustellen. Die andere Möglichkeit, diese Informationen als zusätzliche Einträge im *UID- oder UID-Feld bereitzustellen, wird nicht gewählt, da bei einer Änderung eines IO auch die darauf verweisenden IO geändert werden müssten. Deren Anzahl ist aber nicht bekannt.

Obig genannte Sonderfunktion wird jedoch nicht angewandt, falls ein *UID oder UID referenziert wird, dessen Index-*UID sich vom $\mathcal{H}$(Klartext)-Feld des referenzierenden IO unterscheidet. In diesem Fall kann nicht garantiert werden, dass sich der Mapping-Eintrag des referenzierten *UID oder UID auf dem gleichen Knoten der DHT befindet. Das Abrufen der referenzierten IO von einem anderen Knoten der DHT würde ansonsten eine zusätzliche Last auf das Mapping-System bedeuten.

4.2.4. Registrierung und Zuteilung von Inhalts-UIDs

Ebenso wie für den Geräte-UID ist für den Inhalts-UID ein Registrierungsprozess notwendig, um dessen Einzigartigkeit sicherzustellen und um Missbrauch vorzubeugen. Er ist vergleichbar mit dem für den Geräte-UID, unterscheidet sich jedoch in den Anforderungen an das Mapping-System beim Hinzufügen von spezifischen IO. So muss es für einen Inhaltsersteller möglich sein, nur das abstrakte IO mit dem *UID zu registrieren, der ausschließlich den Hash-Wert des Klartextes besitzt, um anschließend ohne weiteren Registrierungsprozess beliebig viele spezifische IO im Mapping-System abzuspeichern.

In Abbildung 4.3 ist das Zustandsdiagramm für die Registrierung eines Inhalts-UID dargestellt. Im Folgenden werden die einzelnen Nachrichtentypen zusammen mit der daraus resultierenden Aktion erläutert. Das Zustandsdiagramm ist sowohl für öffentlich änderbare als auch für nichtöffentlich änderbare Inhalte gültig, lediglich einige Aktionen sind unterschiedlich. Werte in Klammern müssen zwingend mit einer Nachricht übermittelt werden, Werte in eckigen Klammern sind optional.

Nachrichtentypen:

- `REGISTER_UID(*UID, pub-key)`: Der Ersteller eines Inhalts möchte einen Inhalts-

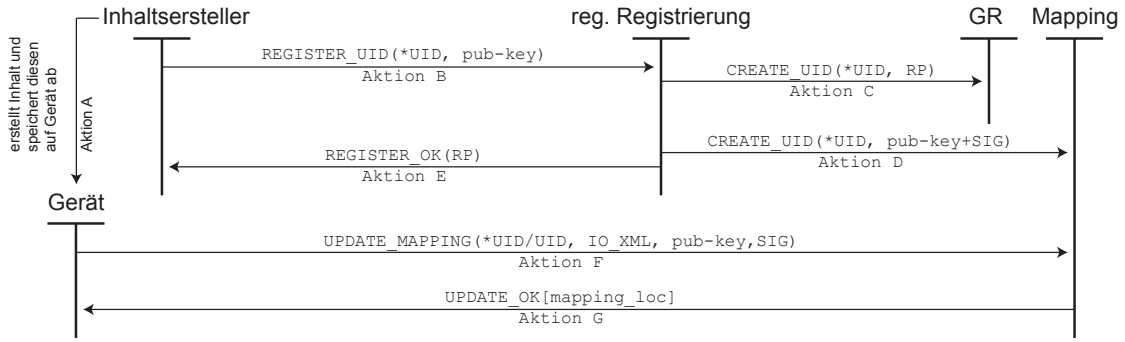


Abbildung 4.3.: Zustandsdiagramm Registrierung Inhalts-UIDs

UID beantragen. Dazu übermittelt er $\mathcal{H}$(Klartext) als *UID, zusammen mit seinem öffentlichen Schlüssel (pub-key) an die regionale Registrierungsstelle. Diese Nachricht führt an der Registrierungsstelle die Aktion B aus.

- `CREATE_UID(*UID, RP)`: Mit dieser Nachricht fordert die Registrierungsstelle das Erstellen eines Mapping-Eintrags in der GR an. Dazu wird das RP der Region, in welcher der Inhalt verwaltet wird, zusammen mit dem *UID übermittelt. Diese Nachricht führt an der GR die Aktion C aus.

- `CREATE_UID(*UID, pub-key+SIG)`: Die gleiche Nachricht, jedoch mit anderen Parametern, wird an das Mapping-System mit der Aufforderung gesendet, den initialen Mapping-Eintrag anzulegen. Übermittelt werden neben dem *UID der öffentliche Schlüssel des Erstellers, der von der Registrierungsstelle signiert wurde. Am Mapping-System wird anschließend Aktion D ausgeführt.

- `REGISTER_OK(RP)`: Diese Nachricht wird von der Registrierungsstelle zurück an den Inhaltsersteller gesendet, sobald `CREATE_UID` an das Mapping-System übermittelt wurde. Die Nachricht bestätigt die korrekte Registrierung und löst Aktion E beim Inhaltsersteller aus.

- `UPDATE_MAPPING(*UID/UID, IO_XML, pub-key, SIG)`: Mit dieser Nachricht werden neue IO erzeugt, die zum initial registrierten *UID gehören. Dazu wird der *UID oder UID des IO übermittelt, sowie die XML-Beschreibung des IO: `IO_XML`. Die gesamte Nachricht wird vom privaten Schlüssel des Erstellers signiert. Im Mapping-System wird darauf Aktion F ausgelöst.

- `UPDATE_OK[mapping_loc]`: Das Mapping-System bestätigt mit dieser Nachricht eine Aktualisierung oder Erstellung. Als optionalen Wert kann der Locator des Mapping-Systems übermittelt werden. Damit ist es möglich, explizit auf einen Knoten der Mapping-DHT zuzugreifen, ohne über den Load-Balancer zu gehen. Diese Nachricht löst Aktion F beim Sender der `UPDATE_MAPPING`-Nachricht aus.

Aktionen:

- `Aktion A`: Nachdem der Inhaltsersteller den Inhalt auf dem Gerät abgespeichert hat, wird auf die `REGISTER_OK`-Nachricht gewartet.

- `Aktion B`: Die Registrierungsstelle überprüft, ob der gewünschte UID noch verfügbar ist und erzeugt im positiven Fall die Nachricht `CREATE_UID` für die GR und das Mapping-System. Darüber hinaus wird der öffentliche Schlüssel des Erstellers von der Registrierungsstelle signiert.

- `Aktion C`: Die GR erstellt im globalen Mapping-System einen Eintrag, der zum *UID das übermittelte RP speichert. Diese Aktion hat keine weitere Nachricht zur Folge.

- `Aktion D`: Das Mapping-System erstellt den initialen, leeren Mapping-Eintrag für *UID. Handelt es sich bei der *UID um den Typ $T = 3$ (nichtöffentlich änderbarer Inhalt), so gestattet das Mapping-System für diesen *UID Aktualisierungen, Änderungen oder das Hinzufügen weiterer *UID oder UID mit gleichem $\mathcal{H}$(Klartext) nur, wenn die entsprechende Nachricht mit dem zum öffentlichen Schlüssel passenden privaten Schlüssel signiert wurde. Handelt es sich dagegen um einen *UID vom Typ $T = 4$, so ist das Hinzufügen weiterer *UID oder UID für jedermann möglich.

- `Aktion E`: Nach dem Empfang von `REGISTER_OK` weist der Inhaltsersteller das Gerät, das den Inhalt speichert, an, das Informationsmodell des Inhalts als IO zu publizieren.

- `Aktion F`: Das Mapping-System ändert oder erstellt, abhängig vom Typ der UID und der Signatur, bestehende Einträge im Mapping-System. Die Änderung eines bestehenden Eintrags ist sowohl für Typ $T = 3$ als auch $T = 4$ nur vom Ersteller dieses Eintrags möglich. Für öffentlich änderbare Inhalte muss eine neue Version angelegt werden.

- `Aktion G`: Nach dem Empfang der `UPDATE_OK`-Nachricht wird das IO als veröffentlicht markiert. Zusätzlich wird, falls vorhanden, die Information des Mapping-Locators ausgewertet.

4.3. Adressierung von Personen

Nicht nur ausschließlich Inhaltsobjekte stehen im Fokus des zukünftigen Internets, auch der Benutzer als Person rückt mehr und mehr in den Vordergrund. Bereits mit der Einführung des Email-Dienstes [Cro82] stand zum ersten Mal die Person im Mittelpunkt der Kommunikation über das Internet. Mit dem Aufkommen von sozialen Netzen, Voice over IP (VoIP)-Telefonie und Instant Messaging existieren immer mehr Kommunikationsformen, welche auf die Person als eigentlichen Endpunkt der Kommunikation abzielen. Dabei ist das Gerät, das eine Person zur Kommunikation benutzt, nicht weiter von Interesse, lediglich dass diese Person erreicht werden kann. Einzig der Kommunikationskanal (Email, VoIP, Instant-Messenger, usw.) muss gewählt werden können.

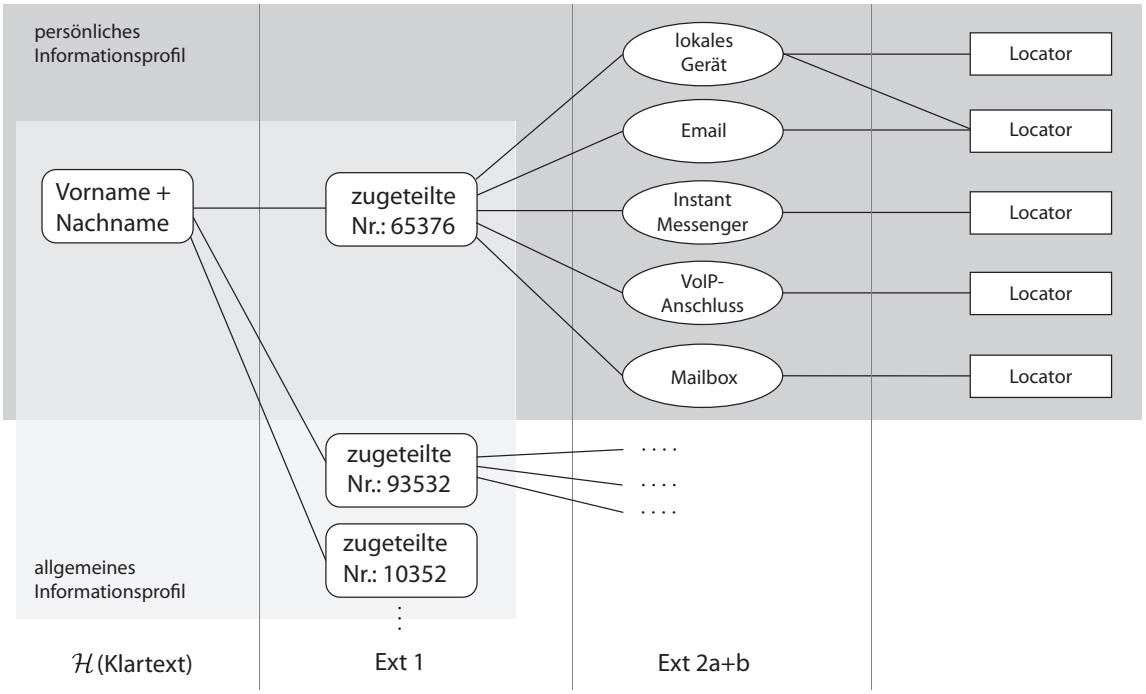


Abbildung 4.4.: Informationsprofil für Personen mit zugehörigen Teilen der UID

In diesem Abschnitt wird daher ein UID für Personen eingeführt, womit Personen adressierter werden. Dieser UID wird einer Person zugewiesen und vereinfacht die Kommunikation zwischen Personen deutlich. Der persönliche UID kann darüber hinaus zur Authentifizierung bei jeglichen Online-Transaktionen verwendet werden.

4.3.1. Informationsprofil für Personen

Im Gegensatz zum Informationsmodell bei Inhalten verweist der UID für Personen auf ein Informationsprofil für Personen, wie in [SK07] vorgestellt. Es wird ebenfalls im Mapping-System abgespeichert und enthält alle möglichen Kommunikationskanäle, über welche die entsprechende Person erreicht werden kann. Ein Beispiel für ein persönliches Informationsprofil wird in Abbildung 4.4 gezeigt und ist dunkelgrau dargestellt. Die dazugehörige Beschreibung, die in XML erfolgt, wird im Anhang A.2 aufgezeigt. Jede Person kann dabei, abhängig von den verwendeten Kommunikationskanälen, eigene Einträge im persönlichen Informationsprofil erzeugen. Jedem dieser Einträge können anschließend ein oder mehrere Locator-Werte zugeteilt werden, die auf das Gerät verweisen, das den entsprechenden Kommunikationsdienst bereitstellt. Ein persönliches Informationsprofil, das exakt eine bestimmte Person identifiziert, definiert sich dabei durch den Vor- und Nachnamen der Person sowie genau einem Eintrag einer zugeteilten Nummer. Im Gegensatz dazu kann das allgemeine Informationsprofil mehrere zugeteilte Nummern zu einem Namen enthalten (hellgrau dargestellt). Damit ist es möglich, einen Indexdienst zu realisieren, der zwischen Personen mit gleichem Vor- und Nachnamen unterscheidet. Welches Informationsprofil angesprochen wird, bestimmt die Belegung der einzelnen Felder des UID, die im Folgenden vorgestellt werden.

4.3.2. UIDs für Personen

Wie für Geräte und Inhalte folgt der UID für Personen dem allgemeinen Schema für UID in HiiMap, dem der UID-Typ mit dem Wert $T = 5$ zugeordnet wird. Als Grundlage für das RP für diese UID-Typen wird die Region gewählt, in der die betreffende Person eine Staatsbürgerschaft besitzt. Bei doppelten Staatbürgerschaften wird die Region des Hauptwohnsitzes gewählt. Die einzelnen Felder des UID spiegeln dabei den Aufbau des Informationsprofils wieder und setzen sich folgendermaßen zusammen:

$\mathcal{H}$(**Klartext**): Dieses Feld wird dazu verwendet, um die Gruppe aller persönlichen Informationsprofile und das allgemeine Informationsprofil zu einem bestimmten Namen zu benennen. Es wird gebildet aus dem Hash-Wert von Vor- und Nachnamen einer Person. Aufgrund der Tatsache, dass mehrere Personen den gleichen Namen besitzen können, wird ein weiteres Feld benötigt, um eine Person eindeutig zu identifizieren.

Ext 1: Dieses Feld wird dazu verwendet, um eine Person eindeutig von anderen Personen gleichen Namens zu unterscheiden. Um Zweideutigkeit und Missbrauch zu vermeiden, wird dieser Wert von einer staatlichen Behörde zugewiesen. Der Abschnitt 4.3.4 über die Registrierung von UID für Personen geht detailliert darauf ein. Der Wert kann grundsätzlich mit einer Personalausweis- oder Sozialversicherungsnummer verglichen werden. Mit dem Wert 0 kann das allgemeine Informationsprofil abgerufen werden, das als Personenverzeichnis ähnlich einem Telefonbuch dient. Aus Gründen der Privatsphäre steht es dabei jeder Person frei, in einem derartigen Verzeichnis gelistet zu sein oder nicht.

Ext 2(a+b): Dieses Feld dient dazu, den entsprechenden Kommunikationskanal aus dem Informationsprofil auszuwählen. Einige Werte dafür sind fest vorgegeben, andere können von neuen Kommunikationsdiensten frei verwendet werden. Der Wert 0 wird dazu verwendet, um das komplette, alle möglichen Kontaktinformationen enthaltende persönliche Informationsprofil der Person zu adressieren und aus dem Mapping-System abzurufen. Der Wert 1 erlaubt es, in diesem Eintrag den Locator des Gerätes abzuspeichern, an dem sich die entsprechende Person aktuell aufhält. Dies ist für die eigentliche Gerätekommunikation sinnvoll, falls z.B Dateien ausgetauscht werden sollen. Dieser Eintrag ist jedoch optional und muss nicht verwendet werden, da auf diese Weise Bewegungsprofile der Person erstellt werden können. In [Han10] wird diese Thematik ausführlich behandelt und Lösungen vorgestellt. Weitere Werte sind für Dienste wie Email, VoIP und gängigen Instant-Messaging Diensten fest vorgegeben. Die verbliebenen, nicht fest vergebenen Werte stehen neuen Diensten, die Kommunikationskanäle anbieten, zur freien Verfügung.

4.3.3. Mapping-Einträge für Personen

Für Personen-UID des Typs $T = 5$ muss es wie für den Inhalts-UID möglich sein, abhängig von den spezifizierten Werten des UID die jeweils relevanten Informationen aus dem Mapping-System zu beziehen. Ein Informationsprofil besteht daher aus mehreren Mapping-Einträgen, die jeweils miteinander verknüpft sind und verschiedene Informationen enthalten. Die allgemeine Darstellung eines derartigen Eintrags ist in Abbildung 4.5 dargestellt. Die hellgrau hinterlegten Felder sind dabei nicht in jedem Mapping-Eintrag vorhanden, sondern abhängig davon, welche Felder des UID spezifiziert sind. Eine Über-

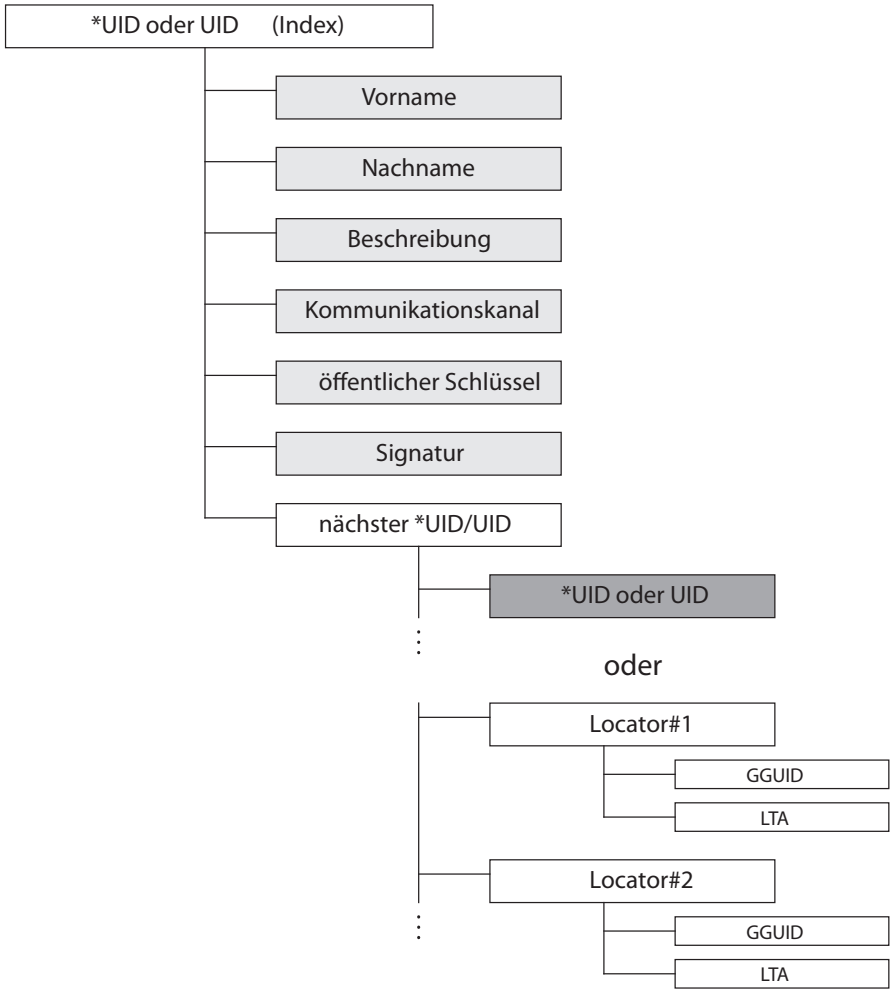


Abbildung 4.5.: Mapping-Eintrag des Informationsprofils für Personen mit *UID/ UID

sicht ist in Tabelle 4.1 zu sehen.

Eine Sonderstellung besitzt das Feld „nächster *UID/UID“. Dort werden alle möglichen Werte für das nächste nicht spezifizierte *UID-Feld aufgelistet. Je nachdem, ob es sich im Index um einen *UID oder UID handelt, sind dort Verweise auf weitere *UID bzw. UID, oder aber Locator-Werte zu finden. Letztere sind nur vorhanden, wenn es sich um den Mapping-Eintrag eines vollständig spezifizierten UID handelt. Wird der Mapping-Eintrag für einen *UID abgefragt, so fügt das Mapping-System beim Zurückliefen des Ergebnisses, genau wie bei Inhalts-UID, zusätzlich den Mapping-Eintrag des referenzierten *UID und UID ein (im Bild dunkelgrau dargestellt), jedoch ohne deren „nächste *UID/UID“-Feld.

Diese Verfahrensweise ermöglicht es, trotz der Flexibilität, verschiedene Informationen abhängig von den spezifizierten UID-Werten abzufragen, nur eine Struktur für Datenbankeinträge zu verwenden. Darüber hinaus muss bei Änderungen des Informationsprofils nur der jeweils betroffene Mapping-Eintrag geändert werden. Nur beim Hinzufügen oder Entfernen von Informationen muss auch der auf einen Eintrag referenzierende Eintrag

geändert werden.

4.3.4. Registrierung und Zuteilung von Personen-UIDs

Für einen Personen-UID ist wie für alle anderen UID-Typen ein eigener Registrierungs- und Zuteilungsprozess nötig. Um die Eindeutigkeit eines Personen-UID zu gewährleisten, kommt jedoch anstatt der Registrierungsstelle eine staatliche Behörde wie das Einwohnermeldeamt zum Einsatz. Dessen Aufgabe besteht darin, den Wert Ext 1 für den UID jeder Person eindeutig zu vergeben, um auch Personen mit gleichem Vor- und Nachnamen voneinander unterscheiden zu können.

Das Zustandsdiagramm für die Registrierung und Zuteilung der Personen-UID ist in Abbildung 4.6 dargestellt und wird im Folgenden beschrieben. Es gilt die gleiche Syntax für Nachrichten wie für Inhalts- und Geräte-UID. Bei den Nachrichten zwischen Person und Behörde handelt es sich nicht notwendigerweise um digitale Nachrichten.

Nachrichtentypen:

- `REGISTER_UID(*UID, pub-key)`: Mit dieser Nachricht stellt eine Person den Antrag auf die Ausstellung eines persönlichen UID durch eine Behörde. Dieser Antrag enthält den Hash-Wert aus Vor- und Nachnamen der Person sowie deren öffentlichen Schlüssel. Dieser Antrag führt bei der Behörde zur Aktion A.
- `CREATE_UID(*UID, RP)`: Die Behörde fordert mit dieser Nachricht das Erstellen eines Mapping-Eintrags in der GR an. Dazu wird das RP der Region, in welcher die Person gemeldet ist, zusammen mit dem *UID übermittelt. Diese Nachricht führt an der GR Aktion B aus.
- `CREATE_UID(*UID, pub-key+SIG)`: Die gleiche Nachricht wird mit anderen Parametern an das Mapping-System gesendet. Dieses legt den initialen, leeren Mapping-Eintrag für den *UID an. Neben dem *UID wird auch der signierte öffentliche Schlüssel der Person übertragen. Am Mapping-System wird anschließend Aktion C ausgeführt.

Tabelle 4.1.: Angaben im Mapping-Eintrag abhängig von belegten UID-Feldern

	Vor- und Nachname	Beschreibung	Kommunikationskanal	öff. Schlüssel + Signatur
*UID aus $\mathcal{H}$(Klartext)	×			×
*UID aus $\mathcal{H}$(Klartext) + Ext 1	×	×		×
UID aus $\mathcal{H}$(Klartext) + Ext 1 + Ext 2a+b			×	

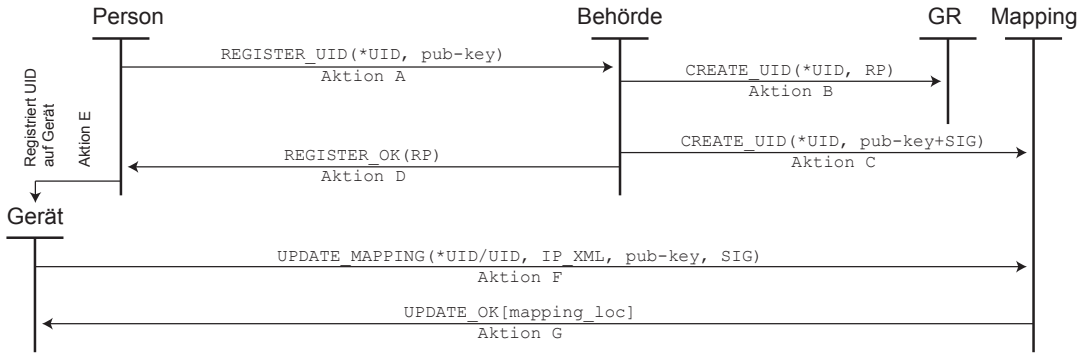


Abbildung 4.6.: Zustandsdiagramm Registrierung UID für Personen

- `REGISTER_OK(RP)`: Die Behörde sendet diese Nachricht an die Person zurück, sobald die `CREATE_UID`-Nachricht an das Mapping-System gesendet wurde. Die Nachricht bestätigt die korrekte Registrierung und löst bei der Person Aktion D aus.
- `UPDATE_MAPPING(*UID/UID, IP_XML, pub-key, SIG)`: Mit dieser Nachricht erstellt das Gerät der Person ein neues, von den verwendeten Kommunikationskanälen abhängiges Informationsprofil. Mit der Nachricht werden der *UID oder UID, sowie das Informationsprofil mit den benötigten Informationen als XML übermittelt. Um die Echtheit der Nachricht zu verifizieren wird außerdem der öffentliche Schlüssel sowie eine Signatur übertragen. Diese Nachricht löst im Mapping-System Aktion F aus.
- `UPDATE_OK[mapping_loc]`: Die Aktualisierung oder Erstellung eines Personen-UID wird vom Mapping-System mit dieser Nachricht bestätigt. Als optionalen Wert kann ein alternativer Locator des Mapping-Systems übermittelt werden. Diese Nachricht löst am Gerät Aktion G aus.

Aktionen:

- `Aktion A`: Die Behörde teilt der Person eine Nummer zu, die sie von anderen Personen mit gleichem Namen unterscheidet. Diese Nummer kann grundsätzlich öfter vergeben werden, jedoch nie an Personen mit gleichem Vor- und Nachnamen. Diese Nummer wird im Ext 1-Feld des persönlichen UID verwendet. Die Behörde signiert darüber hinaus den öffentlichen Schlüssel der Person und erstellt die Nachrichten `CREATE_UID` für die GR und das Mapping-System.
- `Aktion B`: Die GR erstellt im globalen Mapping-System einen Eintrag, der das übermittelte RP zum *UID speichert. Auf diese Aktion folgt keine weitere Nachricht.
- `Aktion C`: Das Mapping-System erstellt den initialen, leeren Mapping-Eintrag für den *UID. Zum Aktualisieren, Ändern oder Hinzufügen weiterer *UID oder UID mit gleichem Hash- und Ext 1-Wert muss die entsprechende Nachricht mit dem zum öffentlichen Schlüssel passenden privaten Schlüssel signiert sein.

- `Aktion D`: Der Benutzer kann den persönliche UID verwenden. Dazu muss er ihn an einem Gerät konfigurieren. Dies löst am Gerät die Aktion E aus.

- `Aktion E`: Das Gerät kann jetzt die persönlichen Kommunikationsdienste der Person mit einem Informationsprofil im Mapping-System veröffentlichen und damit für die Allgemeinheit verfügbar machen. Dies geschieht mit der `UPDATE_MAPPING`-Nachricht.

- `Aktion F`: Das Mapping-System überprüft die Signatur und ändert bzw. erstellt den zur *UID oder UID gehörigen Mapping-Eintrag anhand der XML-Beschreibung.

- `Aktion G`: Nach dem Empfang de `UPDATE_OK`-Nachricht werden die Kommunikationsdienste als veröffentlicht markiert. Zusätzlich wird, falls vorhanden, die Information des Mapping-Locators ausgewertet und künftig verwendet.

4.4. N-Gramm-basierter Nachschlagemechanismus

Die Idee des Namensschemas für UID basiert auf der Generierung aus einem gut einprägsamen Klartext mittels Hash-Funktion, um ein zusätzliches Datenbanksystem zu vermieden. Eine Eigenschaft von Hash-Funktionen ist, dass eine minimale Änderung des Eingangswerts eine sehr große Veränderung des Ergebniswerts bedeutet. Ein bestimmte UID kann daher nur gefunden werden, wenn der Klartext exakt bekannt ist und keine Schreibfehler aufgetreten sind. Das gleiche Problem tritt bei phonetisch gleich klingenden Klartexten auf. Wünschenswert ist daher ein Nachschlagemechanismus, der als eigener Dienst in der Lage ist, trotz Tippfehler oder falscher Schreibweise den korrekten Klartext zu finden. Um auf eine zusätzliche Datenbank zu verzichten, soll dieser Dienst vom Mapping-System übernommen werden. Aufgrund dessen dezentraler Struktur erfordert dies jedoch eine besondere Herangehensweise. Die Autoren in [HHH+02] schlagen für eine DHT einen N-Gramm-basierten Nachschlagemechanismus vor, der in der Lage ist, Platzhaltersuchen durchzuführen. Dieses Verfahren wird auch von den Autoren in [HCW97] und [PBLC10] eingesetzt, um Schreibfehler in Worten zu korrigieren. Eigene, im Folgenden vorgestellte Untersuchungen zeigen, dass sich die Qualität der Suchergebnisse dadurch drastisch verbessern lässt.

4.4.1. Generierung von N-Grammen

Das Prinzip des Nachschlagemechanismus basiert auf N-Grammen, die aus dem gleichen Klartext erzeugt werden wie auch der UID. Dazu wird zunächst der Klartext in Teilwörter der Länge N zerlegt. Anschließend wird der Hash-Wert jedes dieser Teilwörter zusammen mit dem kompletten Klartext ebenfalls in einer Datenbank im Mapping-System abgespeichert [HHH+02].

Typische Werte zur Generierung von N-Grammen sind $N = 2; 3$. Für $N = 3$ wird beispielsweise der Klartext P mit `sueddeutsche` in $I = 10$ so genannte Trigramme h_i mit $i = 1, ..., I$ zerlegt: `sue, ued, edd, dde, deu, eut, uts, tsc, sch, che`. Für jedes dieser Trigramme h_i wird der Hash-Wert $\mathcal{H}(h_i)$ berechnet und zusammen mit dem

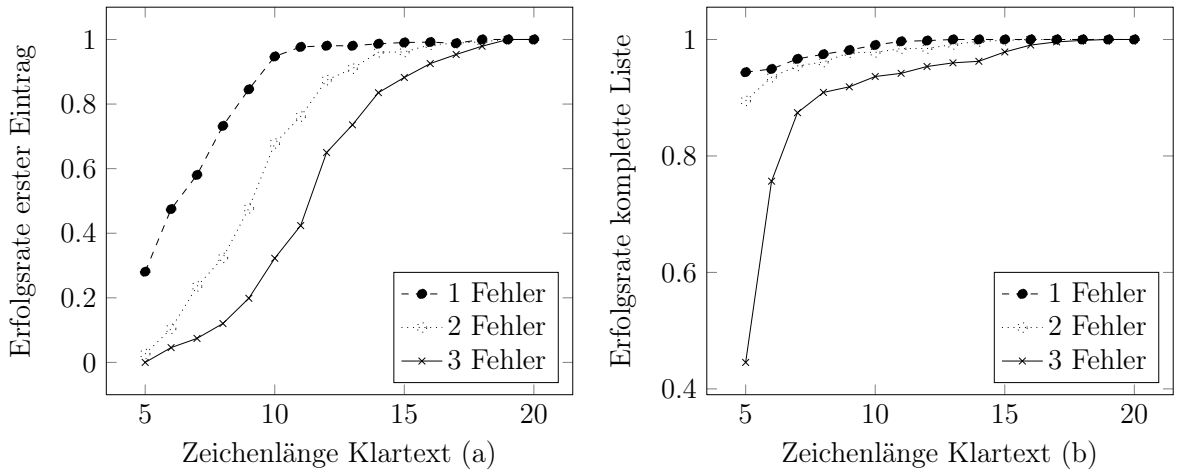


Abbildung 4.7.: Trefferrate für korrekten Klartext bei N-Gramm-unterstützter Suche mit N=3

Klartext P als Tupel $\langle \mathcal{H}(h_i); P \rangle$ im Mapping-System abgespeichert. Dazu kann ebenfalls eine eigene Datenbank verwendet werden. Sollte sich der Mapping-Eintrag des eigentlichen UID ändern, müssen die dazugehörigen N-Gramme nicht geändert werden, da diese ausschließlich den Klartext zum entsprechenden UID enthalten.

4.4.2. Abfrage von UID mit N-Grammen

Jedesmal wenn ein UID im Mapping-System angefragt wird, kommt zunächst der aus dem Klartext berechnete UID zum Einsatz. Nur wenn dabei nicht das gewünschte Ergebnis zurückgeliefert wird, da beispielsweise ein Schreibfehler aufgetreten ist, kommt die N-Gramm-unterstützte Suche zum Einsatz. Dabei wird es dem Benutzer überlassen, ob er diese erneute Suche durchführen möchte.

Zunächst müssen die notwendigen N-Gramme aus dem bekannten, eventuell fehlerbehafteten Klartext berechnet und an das Mapping-System übergeben werden. Dieses sucht anschließend alle gleichen N-Gramme und sortiert diese nach der Häufigkeit des jeweils enthaltenen Klartextes. Je nach Fehlerzahl ist der korrekte Klartext in der zurückgelieferten Liste sehr weit oben vertreten oder sogar an erster Stelle.

Die Möglichkeit, Schreibfehler im Klartext zu korrigieren, wurde durch eigene Simulationen verifiziert. Dazu wurden aus einem Wörterbuch der englischen Sprache mit etwa 100.000 Wörter zunächst die Trigramme aller Wörter erzeugt. Anschließend wurden zufällig 5.000 Wörter ausgewählt und diesen jeweils ein bis drei Fehler hinzugefügt. Mit den Trigrammen ($N = 3$) wurden schließlich Suchen durchgeführt und überprüft, ob sich das korrekte Wort in der Ergebnisliste befindet. Die Ergebnisliste wurde auf 10 Einträge begrenzt. Zur Auswertung kamen zwei verschiede Verfahren zum Einsatz, deren Ergebnis in Abbildung 4.7 dargestellt ist.

Im ersten Verfahren wurde nur der erste Eintrag in der Tabelle berücksichtigt und überprüft, ob es sich dabei um das gesuchte Wort handelt. Das Ergebnis der Erfolgsrate ist in Abbildung 4.7a dargestellt und über die Anzahl der Buchstaben aufgetragen. Dabei

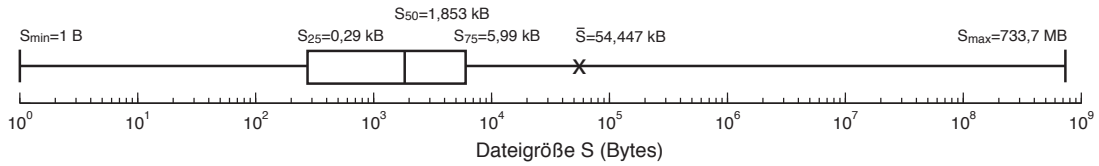


Abbildung 4.8.: Verteilung der Dateigrößen mit Mittelwert

ist zu erkennen, dass die Erfolgsrate stark abhängig von der Länge des Wortes ist. Längere Wörter resultieren in einer höheren Anzahl an Trigrammen, womit die Wahrscheinlichkeit für das richtige Wort an erster Stelle steigt. Selbst bei drei Fehlern und einer Wortlänge von 20 Buchstaben befand sich immer das korrekte Wort an erster Stelle.

Im zweiten Verfahren wurde die komplette Liste ausgewertet, ob das korrekte Wort darin enthalten ist. Das Ergebnis der Erfolgsrate ist in Abbildung 4.7b dargestellt. Wie zu erwarten, liegt die Erfolgsrate dabei deutlich höher. Nachteil dieses Verfahrens ist die Tatsache, dass die gesamte Liste benutzerseitig ausgewertet werden muss, während im ersten Verfahren automatisiert der erste Eintrag der Liste verwendet werden kann. Die Auswertung dieser Liste wird dabei vom Inhalts-Modul in der HiiMap-Middleware übernommen, auf die später eigegangen wird. Sollte das zweite Verfahren zum Einsatz kommen, ist dieses Modul auch für die Repräsentation der Ergebnisliste gegenüber dem Benutzer zuständig.

4.5. Eigenschaften von Inhalten im heutigen Internet

Bevor im Abschnitt 4.6 die Idee der Inhaltsmobilität und der dazugehörige Algorithmus vorgestellt wird, müssen zunächst einige Eigenschaften von Inhalten im heutigen Internet bestimmt werden. Dies umfasst Angaben zur Verteilung der Dateigrößen, der Beliebtheit von Inhalten und Pfadlängen im Sinne von AS-Hops. Einige dieser Daten sind durch Messungen mit einem Web-Proxy über einen Zeitraum von zwei Monaten im Jahr 2009 erhoben worden, wobei die Ergebnisse der Messungen für die spätere Modellierung des Algorithmus sowie für die Simulation benötigt werden. Jeder einzelne Bestandteil einer Webseite wurde als eigenes Inhaltsobjekt gewertet. Für weitere benötigte Daten wird an gegebener Stelle auf andere Messungen verwiesen. Da nach [BDF+09] seit dem Jahre 2007 der durch das WWW verursachte Verkehrsanteil den größten Teil des im Internet übertragenen Verkehrs darstellt, können die durch den Web-Proxy erhobenen Daten als repräsentativ angesehen werden.

4.5.1. Dateigrößen

Die Auswertungen des Versuchs in [MWE00] konnten für das Jahr 2000 eine durchschnittliche Dateigröße von 11,6 kB und einen Median von 3,03 kB angeben. Aufgrund der Tatsache, dass Inhaltsobjekte, besonders Medien wie Bilder, Musik und Videos, in den letzten Jahren deutlich an Popularität zugenommen haben, ist davon auszugehen, dass die Durchschnittsdateigröße jetzt höher liegt. Aus den eigenen Messungen im Jahr 2009 mittels Web-Proxy kann ein Durchschnittswert der Dateigröße $\overline{S}$ von 54,447 kB

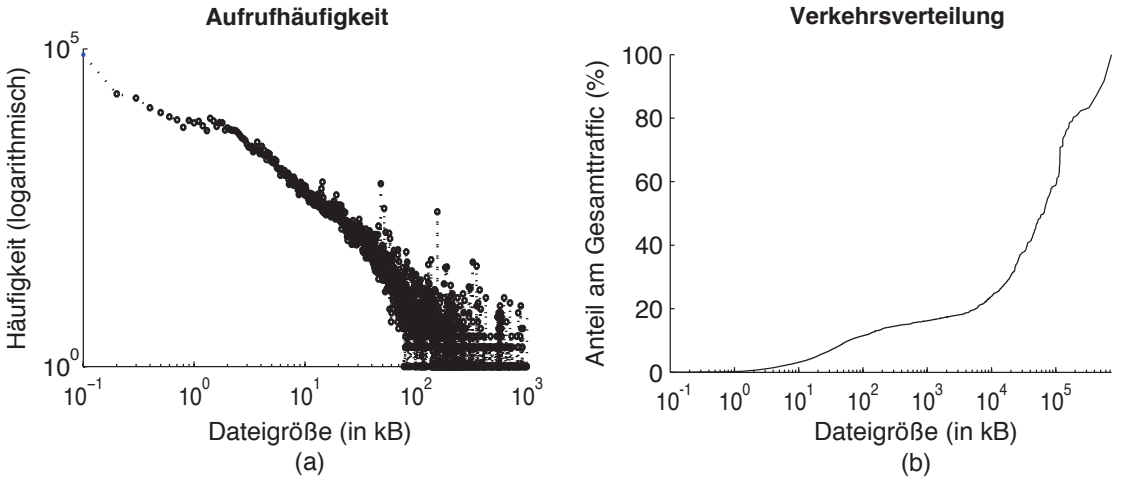


Abbildung 4.9.: Aufrufhäufigkeit und Verkehr über Dateigröße

angegeben werden, der damit deutlich höher liegt als im Jahr 2000. Der Median dagegen liegt mit 1,853 kB unter den Werten von [MWE00] (siehe Abbildung 4.8). Daraus lässt sich schließen, dass es zwar deutlich mehr sehr kleine Dateien als große gibt, die großen aber dafür so groß sind, dass sie den Durchschnittswert deutlich nach oben ziehen. Eine derartige Verteilung der Dateigröße wird endlastig oder heavy tailed genannt.

Zu den besonders kleinen Dateien, die bei fast jedem Zugriff auf eine Web-Seite übertragen werden, zählen unter anderem das so genannte CSS-Stylesheet, das die Layoutinformationen der Seite enthält sowie Favicons, die als Webseitenlogo in der Adresszeile des Browsers eingeblendet werden. Besonders große Dateien werden durch längere Videos sowie CD- und DVD-Abbilder erzeugt. Abbildung 4.9a zeigt einen Ausschnitt der Aufrufhäufigkeit über die Dateigröße. Dabei ist die Endlastigkeit der Verteilung gut sichtbar. Nach [Dow05] folgt die Verteilung dabei in guter Näherung einer logarithmischen Normalverteilung. Abbildung 4.9b zeigt die Verkehrsverteilung und damit den Anteil am Gesamtübertragungsvolumen, aufgetragen über die Dateigröße. Dabei wird sichtbar, dass die Dateien, die sehr häufig aufgerufen werden, nur einen sehr kleinen Anteil zum Gesamtübertragungsvolumen beitragen. Erst die relativ selten aufgerufenen Dateien über 10 MB tragen maßgeblich dazu bei.

4.5.2. Beliebtheit von Inhalten

Wie die Dateigrößen ist auch die Beliebtheit von Inhaltsobjekten im Internet nicht gleich verteilt. Zum einen gibt wenige Inhalte, die sehr beliebt sind und damit häufig aufgerufen werden, zum anderen gibt es sehr viele Inhalte die nur sehr selten angefragt werden. Die Autoren in [MWE00] zeigen, dass die Beliebtheit aller Inhaltsobjekte mit dem Zipfschen Gesetz sehr gut angegeben werden kann. Dabei kann die Beliebtheit eines beliebigen Inhaltsobjektes, dargestellt durch die Aufrufhäufigkeit P, auf folgende Weise angegeben werden:

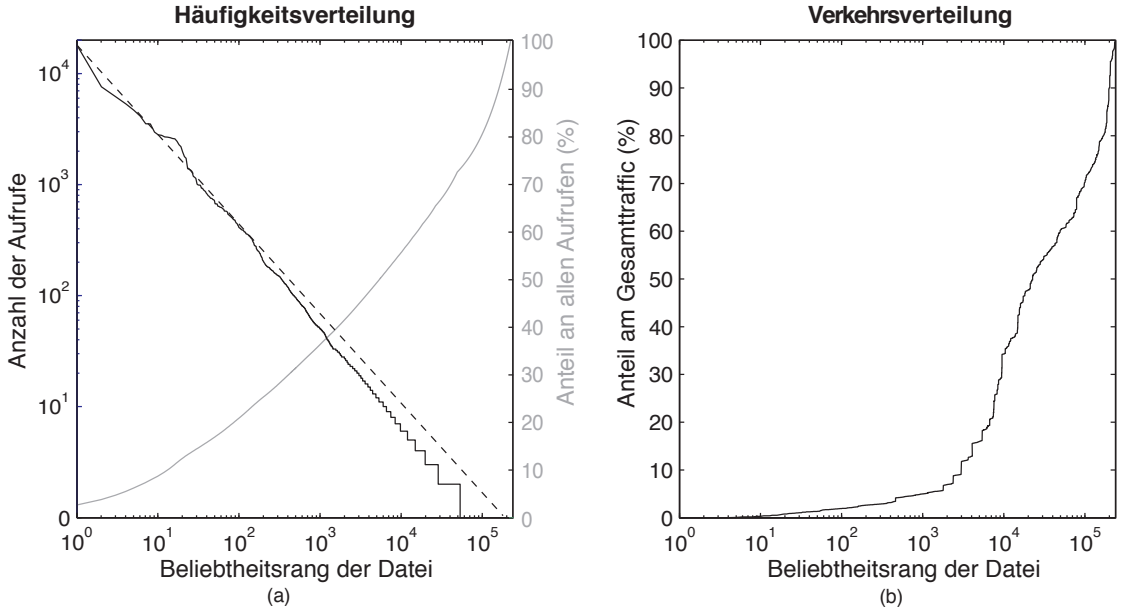


Abbildung 4.10.: Häufigkeitsverteilung und Verkehrsverteilung von Inhaltsobjekten

$$P \sim \left(\frac{1}{r}\right)^{\beta} \tag{4.1}$$

Dabei entspricht r dem Rang des Dokuments, wenn alle Inhaltsobjekte nach Beliebtheit sortiert werden. Der Parameter β charakterisiert die Verteilung eindeutig und wurde in den Versuchen in [MWE00] mit Werten zwischen 0,75 und 0,85 bestimmt. Die eigene Auswertung ergab einen Wert von 0,83. Die Beliebtheit ist damit direkt proportional zur Exponentialfunktion des reziproken Werts des Rangs. Charakteristisch für die Zipf-Verteilung ist die Linearität in der doppelt logarithmischen Darstellung. Der Exponent β äußert sich dabei durch die Steigung der Geraden in dieser Darstellung. In Abbildung 4.10a ist die Häufigkeitsverteilung für die eigenen Messungen mittels Web-Proxy angegeben. Die gestrichelte Linie entspricht dabei der Zipf-Verteilung mit dem Exponenten $\beta = 0,83$. Für die Abbildung wurden 237.616 Objekte betrachtet die insgesamt 675.804 mal aufgerufen wurden. Die Stufenform der Grafik für unbeliebte Ränge lässt sich dadurch erklären, dass in diesem Bereich auf mehrere Objekte gleich oft zugegriffen wurde. Diese besitzen daher den gleichen Rang. Darüber hinaus ist zu erkennen, dass die beliebtesten 5.000 Objekte für insgesamt 50% der Aufrufe verantwortlich sind, jedoch für weniger als 20% des übertragenen Datenvolumens, das kumulativ über den Beliebtheitsrang in Abbildung 4.10b dargestellt ist.

4.5.3. Pfadlängen

Unter einem Pfad versteht man den durch das Routing-System bestimmten Weg eines Datenpakets durch das Internet, dessen Ursprung und Ziel üblicherweise in einem AS am Rand des Netzes liegen. Die Anzahl an Hops zwischen Quelle und Ziel definiert

die Pfadlänge. Die in diesem Abschnitt besprochenen Pfadlängen gelten dabei für alle Datenpakete und beschränken sich nicht auf Daten von Inhaltsobjekten. Die Autoren in [LLFS05] zeigen, dass sich die Pfadlängen im Internet mit einer Gammaverteilung approximieren lassen. Die mittlere Pfadlänge wurde gemessen zu 15,57 Hops, der mittlere kürzeste Pfad zu 11,4 Hops. Dies entspricht einer Standardabweichung von $\sigma = 3,99$.

Relevant für die nachfolgend vorgestellte Inhaltsmobilität ist jedoch die Pfadlänge bezogen auf die Anzahl an AS-Hops. Dabei zählt jedes durchquerte AS auf dem Weg von der Quelle zum Ziel als ein Hop. Das Archipelago Projekt in [CAI09] sowie RIPE NCC in [Rip11] untersuchen die Anzahl der durchquerten AS eines Pfades. Der Mittelwert ist dabei relativ starken Schwankungen ausgesetzt und bewegte sich im August 2011 zwischen 4,6 und 4,9 AS-Hops je Pfad. Beide Untersuchungen bestätigen darüber eine Varianz der AS-Pfadlängen zwischen 2,7 und 4 AS-Hops.

4.6. Ressourceneffiziente Inhaltsmobilität

Ausgehend von den erhobenen Daten über Zugriffsstatistik, Dateigrößenverteilung und AS-Pfadlängen wird im Folgenden Abschnitt die Idee der Inhaltsmobilität in der HiiMap-Architektur eingeführt. Während mit Mobilität üblicherweise die Bewegung von Personen und Geräten in Verbindung gebracht wird, so erlaubt es dieses neuartige Konzept, dass sich Inhaltsobjekte ebenso frei im Netz bewegen können und nicht an einen festen Speicherort gebunden sind. Mittels des in [SS11b] vorgestellten Algorithmus kann dabei jeweils der kosteneffizienteste Speicherort für ein Inhaltsobjekt aus der Sicht des gesamten Netzes gewählt werden.

4.6.1. Idee und Motivation

Die Ausweitung des Mobilitätsbegriffs auf Inhalte betrifft in HiiMap grundsätzlich jedes Inhaltsobjekt und bedeutet letztendlich eine dynamische Zuteilung des Speicherortes. Dies beinhaltet sowohl die Replikation, als auch das Verschieben eines Objekts. Die Entscheidung, ab wann für ein Inhaltsobjekt ein neuer Speicherplatz effizienter ist, wird von einem Algorithmus übernommen, der die anfallenden Kosten für verschiedene Szenarien berechnet. Die Inhaltsmobilität steht dabei ausnahmslos für alle digital repräsentierbaren Inhaltsobjekte zur Verfügung, unabhängig ob es sich dabei um kommerzielle oder private Inhalte handelt.

Eine effiziente Verteilung lässt sich dadurch erreichen, dass einzelne Objekte, die von vielen Benutzern angefragt werden, in geeigneter Weise in die Nähe dieser Benutzer „wandern“. Einen ähnlichen Ansatz verfolgt auch das Konzept von CDN. Diese kopieren jedoch alle Objekte ihrer Kunden unabhängig von der Anfragelast auf Replika-Server. Darüber hinaus können sie nicht auf ein Adressierungsschema zurückgreifen, dass unabhängig vom Speicherort ist. Mittels Caching können Inhalte ebenfalls in der Nähe des Benutzers gespeichert werden, jedoch ist in diesem Fall die Anzahl der Replika nicht bekannt und Cache-Kohärenz nicht gewährleistet. Caches werden darüber hinaus nicht effizient ausgenutzt, wie die Autoren in [Zhe04] nachweisen. Da jedes Objekt zunächst im Cache gespeichert wird, werden im Durchschnitt 60%–70% der Objekte im Cache

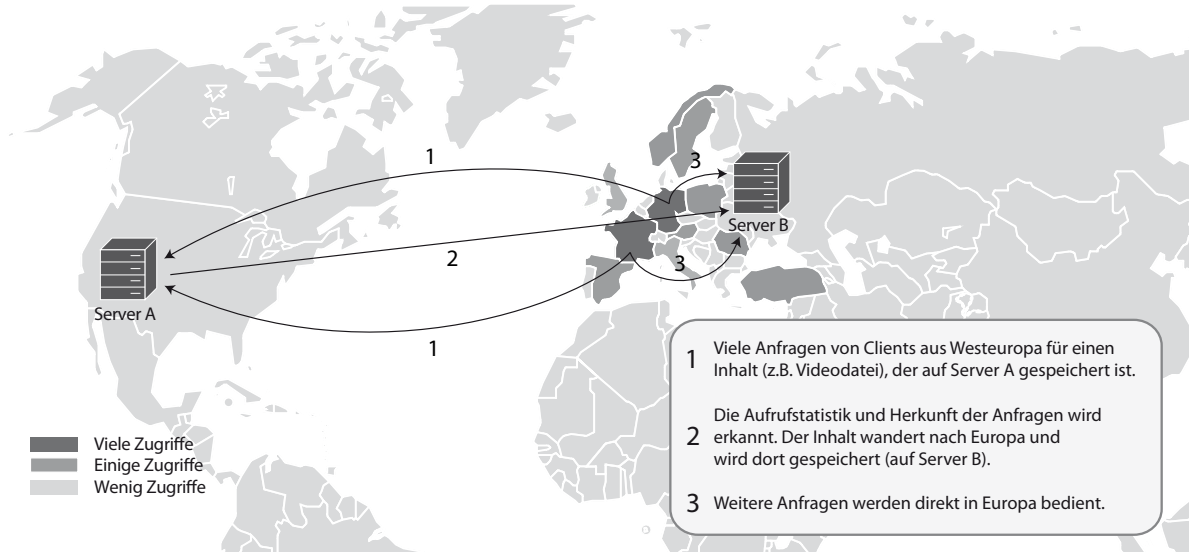


Abbildung 4.11.: Prinzip der Inhaltsmobilität

nur einmal aufgerufen und vergeuden damit Platz für potentiell beliebtere Inhalte. Aufgrund der endlastigen Verteilung der Dateigrößen werden darüber hinaus große Dateien vorgehalten, die ebenfalls nur sehr selten erneut aufgerufen werden und Platz belegen.

Mit dem Prinzip der Inhaltsmobilität wird das Internet ausgehend von einem reinen Übertragungssystem zu einem intelligenten Netz erweitert. Dies bedeutet, dass das Netz im Sinne eines inhaltsorientierten Netzes selbstständig Inhaltsobjekte replizieren und verschieben kann und den dafür nötigen Speicherplatz zur Verfügung stellt. Die Kosten für die Bereitstellung von Inhalten, die besonders durch das Bereitstellen von Hardware für Speicherplatz und Inter-Domain-Verkehr anfallen, werden dabei aus der Sicht des Netzes minimiert.

Abbildung 4.11 zeigt ein Anwendungsszenario der Inhaltsmobilität anhand eines Videos von Youtube. Unter der Annahme, dass Videos von Youtube immer auf einem Server in den USA gespeichert werden, müssen alle Anfragen von dort bedient werden. Dies führt zu hohen Latenzzeiten und Inter-Domain-Kosten, falls das Video bevorzugt außerhalb der USA aufgerufen wird. Für ein bestimmtes Video ist in der Weltkarte hervorgehoben, von welchen Ländern aus die meisten Aufrufe erfolgen. Dabei zeigt sich, dass dieses Video hauptsächlich in Westeuropa aufgerufen wird. Sprachen führen dazu, dass Inhalte wie Videos, Musik und Texte nur regional beliebt sind. Wird vom Netz mittels Algorithmus zur Inhaltsmobilität erkannt, dass ein Großteil der Anfragen aus Europa kommt, kann das Video zwischenzeitlich oder auch permanent auf einen Speicherserver in Europa verschoben werden. Das Netz übernimmt dann auch die korrekte Anpassung der IO im Mapping-System.

4.6.2. Voraussetzungen

Zwingende Voraussetzung für die Einführung des Prinzips der Inhaltsmobilität ist die Möglichkeit, Inhalte direkt mittels eigenem UID anzusprechen und die damit verbundene

Trennung von Speicherort (Locator) und Identifier. Für die Ausführung des Mobilitätsalgorithmus und die damit verbundene Kostenberechnung ist eine intelligente Instanz im Netz zuständig, welche die Mobilität von Inhalten koordiniert und die Berechtigung besitzt, die im Mapping-System gespeicherten IO zu modifizieren. Das Wandern von Inhaltsobjekten ist prinzipiell mit dem Roamingvorgang für Geräte vergleichbar. Dabei können Inhaltsobjekte kurz- oder langfristig auf anderen Geräten zwischengespeichert werden, es bleibt jedoch immer die Ursprungsregion für den Mapping-Eintrag verantwortlich. Inhaltsmobilität ist nur auf statische Inhaltsobjekte anwendbar, da diese im Vergleich zu dynamischen nicht erst zur Laufzeit erzeugt werden .

Zusätzlich muss das Mapping-System in der Lage sein, anhand der Position des jeweils anfragenden Benutzers nur die jeweils passenden Locator-Werte bzw. eine sortierte Liste mit Locatoren zurückzuliefern, so dass die anfragenden Geräte auch die kosteneffizienteste Kopie anfragen. Aufgrund der Ergebnisse der Autoren in [KRR02], die in ihren Versuchen zu Replikationsheuristiken den Schluss ziehen, dass nur mit einer koordinierten Heuristik Inhalte sinnvoll verteilt werden können, wird der Mobilitätsalgorithmus von einer zentralen Instanz pro Region berechnet. Darüber hinaus müssen Server mit Speicherplatz zur Verfügung stehen, auf denen je nach Bedarf Inhaltsobjekte abgelegt werden können.

4.6.3. Ablauf und Anwendungsszenarien

Um auf das Zugriffsverhalten auf Inhaltsobjekte dynamisch reagieren zu können, muss eine Statistik über deren Aufrufhäufigkeit geführt werden. Dies beinhaltet nicht nur die Anzahl, sondern auch die Herkunft der Anfragen. Daten über den Ursprung der Anfrage können anhand des GGUID erhoben werden, der aufgrund seines hierarchischen Aufbaus eine Einordnung der Region und des Betreibers erlaubt. Ausgehend von dieser Statistik kann abgeschätzt werden, ob ein Verschieben oder Kopieren von Inhalten auf andere Speicherorte sinnvoll ist. Falls ein Verschieben oder Kopieren nicht sinnvoll ist bleibt der Inhalt auf dem ursprünglichen Speicherort weiterhin verfügbar. Die Abschätzung erfolgt durch eine Prognose der Statistikwerte in naher Zukunft sowie durch eine Kostenfunktion, die tatsächliche monetäre Kosten, administrativen Aufwand und Auslastung zueinander in Relation setzt. Diese Kosten werden für verschiedene Szenarien bestimmt und durch Auswahl des Szenarios mit den geringsten Kosten minimiert.

Die möglichen Ergebnisse dieser Abschätzung lassen sich in drei Kategorien zusammenfassen:

1. Das Inhaltsobjekt bleibt auf dem Ursprungsserver und wird nicht verteilt.
2. Das Inhaltsobjekt wird vom Ursprungsserver auf einen anderen Server verschoben, der anhand der Gesamtkosten besser für die Bereitstellung des Inhaltsobjekts geeignet ist. Es bleibt keine Kopie auf dem Ursprungsserver.
3. Das Inhaltsobjekt wird anhand bestimmter Optimierungskriterien auf einen oder mehrere Server repliziert.

Für den ersten Fall sind keine weiteren Aktionen notwendig, da das aktuelle Verteilungsszenario bereits die geringsten Kosten bietet. Im zweiten Fall muss ein Server gefunden

werden, der besser geeignet ist als der Ursprungsserver. Ein wichtiges Kriterium ist dabei die Nähe zum Großteil der Konsumenten, wobei Nähe in diesem Zusammenhang nur bedingt mit der räumlichen Distanz gleichzusetzen ist, sondern im Sinne der Kommunikationstechnik vielmehr mit kurzen Pfadlängen und kurzen Verzögerungen. Physikalisch bedingt sind diese jedoch auch von der räumlichen Distanz abhängig. Im dritten Fall muss der Speicherort der einzelnen Replikate gefunden und die optimale Anzahl an Servern bestimmt werden, so dass die Gesamtkosten minimiert werden können. Beide Optimierungsaufgaben sind nicht trivial und von verschiedenen Faktoren abhängig, die später genauer erläutert werden.

Nach der Festlegung der benötigten Server und deren Ort müssen die Inhaltsobjekte dorthin verteilt werden, was letztendlich einem Aufruf des entsprechenden Inhaltsobjekts von einem der ausgewählten Speicherserver bedeutet. Anschließend erfolgt die Anpassung des Mapping-Eintrags des entsprechenden IO. Da immer die Ursprungsregion für den Mapping-Eintrag eines Inhaltsobjekts zuständig ist und damit keine Aktualisierungen am RP erfolgen müssen, ist die GR von der Inhaltsmobilität nicht betroffen. Zwar kann der Fall eintreten, dass Mapping-Anfragen von einer Region beantwortet werden müssen, die eine große Distanz zum Konsumenten aufweist, jedoch ist es das Ziel der Inhaltsmobilität, die konkrete Zustellung des Inhaltsobjekts effizienter zu gestalten.

Abbildung 4.12 zeigt schematisch die einzelnen Schritte, die im Falle von Inhaltsmobilität der Reihe nach durchlaufen werden. Dabei wollen die Benutzer 1 bis 4 auf ein Inhaltsobjekt zugreifen, das auf Server A in der Region 1 abgespeichert ist. Die Benutzer sind dabei stellvertretend für eine große Benutzergruppe. Zunächst wird die GR angefragt, um das zur Inhalts-UID gehörige RP abzurufen (1). Es wird dabei vereinfacht davon ausgegangen, dass das gewünschte DO durch den Inhalts-UID bereits vollständig spezifiziert ist und diesbezüglich keine weitere Auswertung erfolgen muss. Im nächsten Schritt (2) wird das Mapping-System der entsprechenden Region angefragt, um die Locator-Werte des Speicherorts des DO zu erhalten. Der gewünschte Inhalt kann dann direkt vom Urspungsserver geladen werden (3). Erkennt dieser eine hohe Anfragelast, kann er das Mapping-System der Region 1 als zuständige zentrale Instanz für Inhaltsmobilität anweisen, ein alternatives Verteilungsszenario aufzustellen. Dieses Vorgehen ist in Schritt (4) zusammengefasst. Weitere Anfragen für das Inhaltsobjekt (5) werden vom Mapping-System, abhängig vom Ort der Konsumenten, an den entsprechend nächsten Speicherserver geleitet (6).

4.6.4. Modellierung und Annahmen

In diesem Abschnitt werden die Modelle für die Speicherserver und das Netz vorgestellt, die für eine ressourceneffiziente Zustellung von Inhaltsobjekten und den dazugehörigen Algorithmus erforderlich sind. Darüber hinaus wird für den weiteren Entwurf des Algorithmus das Netz selbst als Verwaltungsinstanz angesehen, so dass die Aufgaben der Verteilung und Optimierung von Inhaltsobjekten direkt von diesem und nicht von Dritten, wie einem CDN, übernommen werden. Damit wird der Forderung in [KCC+07, JST+09] zum Übergang zu einem intelligenten Netz, das eine einfache Bereitstellung von Inhalten ermöglicht, Rechnung getragen. Folgende Annahmen werden dabei zugrunde gelegt:

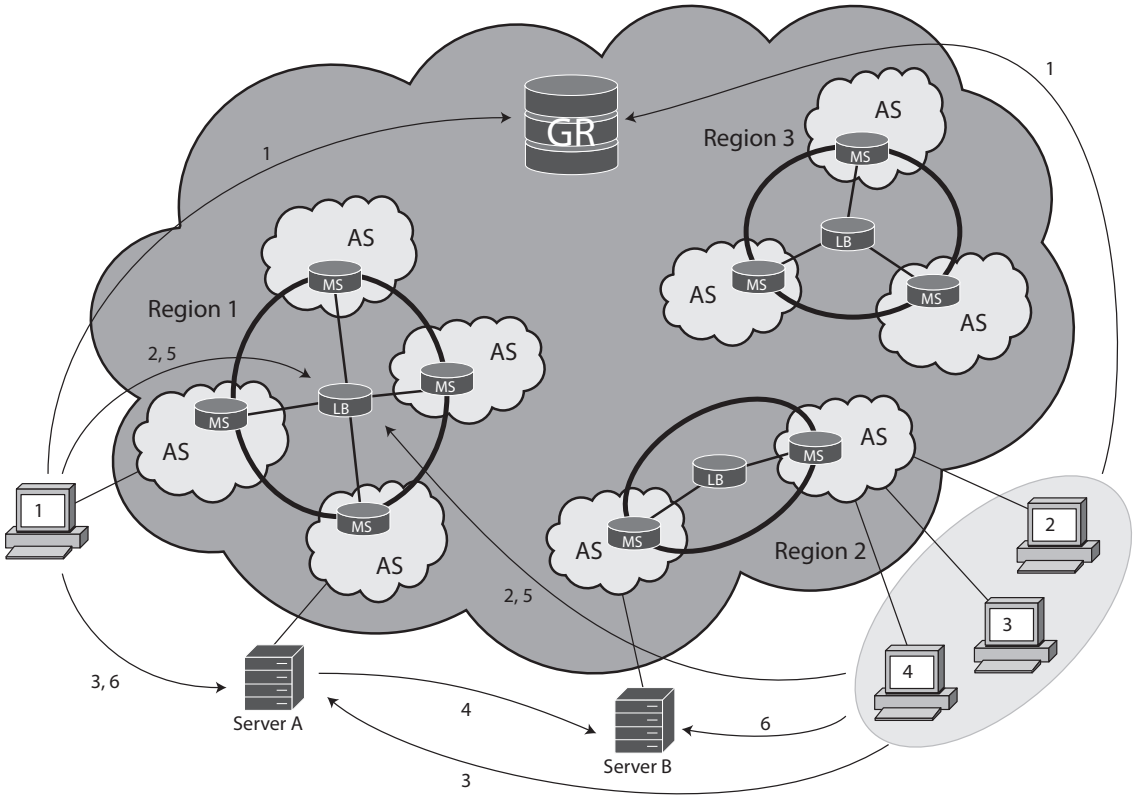


Abbildung 4.12.: Ablaufschema der Inhaltsmobilität in HiiMap

- In jeder Region stehen dedizierte Server zur Speicherung von Inhaltsobjekten bereit. Diese werden von den Netzbetreibern bereitgestellt. Auslastungsdaten über die einzelnen Server, sowie die AS der Betreiber stehen dem Algorithmus zur Verfügung.
- Ein Verwaltungsserver pro Region ist für die Ausführung des Algorithmus zur Inhaltsmobilität verantwortlich. Die Verwaltung betrifft dabei alle in dieser Region initial erstellen Inhaltsobjekte.
- Eine Aufrufstatistik inklusive der Herkunft der Anfragen wird dem Algorithmus entweder vom Mapping-System oder von den Speicherservern zur Verfügung gestellt.

4.6.4.1. Servermodell

Zur Berechnung der Kosten für die Bereitstellung und Verteilung eines Inhaltsobjektes muss die Auslastung der Speicherserver ermittelt werden können. Dazu wird das M/M/1/K- Wartesystem mit einer Bedieneinheit (BE), einer Warteschlange der Länge K und unendlich vielen Quellen modelliert [CANK03, HKSC04]. Mit dem Parameter K wird festgelegt, wie viele Anfragen maximal auf eine Bearbeitung warten können, bevor

Anfragen abgewiesen werden. Die Modellierung mit nur einer Bedieneinheit stellt zwar eine deutliche Vereinfachung dar, da heutige Systeme mehrere Prozessoren zur Verfügung haben und mit Kind-Prozessen quasi mehrere Anfragen gleichzeitig bearbeiten können. Diese Modellierung kann nach [HKSC04] jedoch trotzdem angewandt werden.

Die Ankunfts- und Bedienprozesse eines M/M/1/K-Wartesystems (Abbildung 4.13) werden in der Verkehrstheorie als Markoff-Prozess behandelt, die sich als gedächtnislos auszeichnen. Der mittlere Rufabstand ist dabei negativ exponentiell um den Mittelwert $1/\lambda$ verteilt. Ebenso folgt die Bediendauer einer negativ exponentiellen Verteilung mit dem Mittelwert $1/\epsilon$. Dabei ist λ die mittlere Ankunftsrate und ϵ die mittlere Bedienrate.

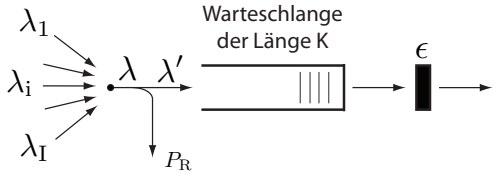


Abbildung 4.13.: Warteschlangenmodell eines M/M/1/K-Servers

Jeder Server speichert insgesamt I verschiedene Inhaltsobjekte. Jedes Inhaltsobjekt besitzt eine Aufrufrate λ_i mit $\mathrm{i} = 1, ..., I$. Die Gesamtaufrufrate aller Inhaltsobjekte auf einem Server ergibt sich aus der Summe der einzelnen Aufrufraten zu.

$$\lambda = \sum_{\mathrm{i}=1}^{I} \lambda_\mathrm{i} \tag{4.2}$$

Die Anzahl der Aufrufe eines Inhaltsobjekts i innerhalb einer Zeitperiode T kann bestimmt werden mit:

$$x_{\mathrm{i,t}} = \lambda_\mathrm{i} \cdot T \tag{4.3}$$

Die Auslastung des Systems ρ mit einer Bedieneinheit lässt sich anhand der Ankunftsrate λ und der Bedienrate ϵ berechnen zu:

$$\rho = \frac{\lambda}{\epsilon} \tag{4.4}$$

Aufgrund der endlichen Warteschlange mit K Plätzen können bei hoher Auslastung nicht alle Anfragen bedient werden. Dies führt dazu, dass einige Anfragen abgelehnt werden müssen. Mit Hilfe der Ablehnungswahrscheinlichkeit P_R kann die effektive Ankunftsrate λ' berechnet werden:

$$\lambda' = (1 - P_\mathrm{R}) \tag{4.5}$$

$$P_\mathrm{R} = \begin{cases} \frac{(1-\rho)\rho^K}{(1-\rho^{K+1})} & \text{für } \rho \neq 1 \\ \frac{1}{K+1} & \text{für } \rho = 1 \end{cases} \tag{4.6}$$

Entsprechend [Pen08] wird in diesem Modell $K = 8$ gewählt. Die Bedienrate ϵ eines Servers ist abhängig von mehreren Faktoren, wie der verwendeten Hardware, der Netzanbindung und der Größe des Inhaltsobjekts. Der limitierende Faktor für einen Speicherserver, der ausschließlich statische Inhaltsobjekte bearbeitet, ist dabei fast ausschließlich die Netzanbindung. Demgegenüber spielt die Leistungsfähigkeit des Prozessors eine untergeordnete Rolle [SHB06, MBS+09]. Im Folgenden wird für einen Speicherserver eine Brutto-Übertragungsrate von 1 Gbit/s angenommen. Die Netto-Übertragungsrate wird aufgrund des Überhangs durch Packet-Header für die Übertragung auf 70% der Brutto-Übertragungsrate gesetzt. Ausgehend von der mittleren Dateigröße $\overline{S} = 54,447\text{kB}$ kann die mittlere Bediendauer h berechnet werden mit $h = \frac{54,447\text{kB} \cdot 8}{0,7 \cdot 1\ \text{Gbit/s}}$, wobei gilt: $\epsilon = 1/h$. Damit ergibt sich eine Bedienrate von $\epsilon = 1607\frac{1}{\text{s}}$

Desweiteren wird die Annahme getroffen, dass die Hardware-Komponenten des Servers, wie Festplatte und CPU, in der Lage sind, die errechnete Rate zu verarbeiten. Die Auslastung eines Servers ρ wird damit vollständig über die Auslastung der Netzanbindung modelliert. Die Autoren in [MBS+09] konnten in ihrem Versuch mit einem Apache-Webserver eine tatsächliche Bedienrate von knapp $\epsilon = 10.770\frac{1}{\text{s}}$ Anfragen pro Sekunde für eine 10 kB große Testdatei messen. Umgerechnet auf die hier verwendete Dateigröße ergibt sich eine Bedienrate von knapp unter 2.000 Anfragen pro Sekunde, was die Größenordnung der hier getroffenen Annahme bestätigt.

4.6.4.2. Netzmodell

Der Modellentwurf für das Internet stellt aufgrund dessen Dynamik eine schwierige Aufgabe dar. Der zu entwickelnde Algorithmus setzt auf der Hiimap-Architektur auf, die das Netz in verschiedene Regionen einteilt. Jede Region besitzt eine Verwaltungsinstanz und besteht grundsätzlich aus mehreren AS. Im hier verwendeten Netzmodell wird jedoch vereinfacht angenommen, dass jede Region aus genau einem AS besteht. Mittels einer Distanzmatrix M werden die benötigten Hops zwischen den Regionen definiert. Desweiteren wird davon ausgegangen, dass die Last, die durch die Bereitstellung von Inhalten mittels Speicherserver erfolgt, im Vergleich zur restlichen Last in jeder Region vernachlässigt werden kann. Stellvertretend für den Zusammenschluss mehrerer AS zu einer Region wird jeder Region eine Auslastung ρ_{R} zugeteilt, die sich im Bereich von 30%-70% bewegt. Diesen Annahmen liegen Messungen von TeleGeography Research [Tel11a] zugrunde, die Messungen zur Auslastung der Inter-Domain-Verbindungen eines jeweiligen Kontinents durchführen. Da die Aufteilung der Regionen in diesem Modell weitestgehend Kontinenten entspricht, ist diese Annahme vertretbar.

Die anfallenden Kosten, die durch eine Netzanbindung entstehen, lassen sich in zwei Bereiche einteilen. Zum einen entstehen für einen Provider Kosten durch den Anschluss an einem Austauschknoten, an dem mehrere Betreiber angeschlossen werden können. Der Austauschknoten DeCIX in Frankfurt berechnet beispielsweise etwa 1.000 USD für einen Anschluss mit der Datenrate 1 Gbit/s. Die Preise für Anschlüsse mit höherer Datenrate steigen linear. Zusätzlich wird noch ein Vertrag mit einem Betreiber benötigt, der die Daten weiterleitet. Die Kosten dafür bewegen sich im Bereich von 10 USD pro Mbit/s und Monat [Tel11a]. Da das Volumen nicht berechnet wird, fallen keine weiteren Kosten für den Betreiber mehr an. Um Inhaltsobjekte aber aus der Sicht des Netzes

Algorithmus 4.1 Auswahlalgorithmus zum Aufruf des Hauptalgorithmus

Eingabe: $\lambda_{\mathrm{i}}^{(max)}, \lambda_{\mathrm{i}}^{(min)}, \overline{D}_{\mathrm{n}}^{(max)}, \rho_{\mathrm{n}}^{(max)}$

```
for all Objekte i auf Server n do
  if λ_i^(e) ≥ λ_i^(max) then
    Starte Hauptalgorithmus für Objekt i
  else if λ_i^(e) ≤ λ_i^(min) then
    Starte Hauptalgorithmus für Objekt i
  else if D̄_n ≥ D̄_n^(max) then
    Starte Hauptalgorithmus für Objekt i
  else if ρ_n ≥ ρ_n^(max) then
    Starte Hauptalgorithmus für Objekt i
  end if
end for
```

ressourceneffizient verteilen zu können, sind in der Kostenberechnung noch Kosten C_{D} für die Distanz in AS-Hops D zu addieren. Diese orientieren sich an den Vertragskosten der Betreiber.

4.6.5. Algorithmus zur Inhaltsmobilität

Der folgende Abschnitt stellt einen Algorithmus vor, mit dessen Hilfe das Netz entscheiden kann, ob Inhaltsobjekte kopiert oder repliziert werden sollen, um deren Zustellung ressourceneffizient zu gestalten. Der Algorithmus kann dabei durch Gewichtung einzelner Parameter grundsätzlich auf jedes beliebige Anwendungsszenario angepasst werden.

Aufgrund der enormen Anzahl an Inhaltsobjekten im Internet ist es sinnvoll, zunächst eine Vorauswahl zu treffen, welche Objekte mobil sein dürfen. So können dynamisch generierte Inhaltsobjekte von vornherein ausgeschlossen werden, da diese erst zur Laufzeit generiert werden, oft personalisiert sind und nur eine sehr kurze Gültigkeit besitzen. Ebenso kann für manche statische Inhaltsobjekte Mobilität unerwünscht sein, da diese aufgrund von Urheberrechten nicht in bestimmten Ländern oder Regionen abgelegt werden dürfen.

Inhaltsobjekte können außerdem in verschiedene Klassen eingeteilt werden, wenn sie unterschiedliche Anforderungen bezüglich Verzögerung und Übertragungsrate benötigen. Mögliche Klassen sind hierbei „Webseite“, „Download“ und „Streaming“. Für das Streaming ist besonders eine konstante Datenrate, sowie eine geringe Verzögerung wichtig. Für den Download dagegen spielt die Verzögerung eher eine untergeordnete Rolle, dafür ist die Übertragungsrate das entscheidende Kriterium. Für eine Webseite wird eine geringe Verzögerung gewünscht, damit der Benutzer diese möglichst zügig betrachten kann. Jede Klasse bestimmt dabei individuelle Parameter des Algorithmus und damit die Kosten der Bereitstellung von Inhaltsobjekten.

Da ein periodischer Aufruf des Algorithmus für jedes Inhaltsobjekt keine praktikable Lösung darstellt, können die Speicherserver, die ein bestimmtes Inhaltsobjekt bereitstellen, bei einer zentralen Instanz pro Region eine Untersuchung und damit den Start des Algorithmus anstoßen. Die Aufgabe der zentralen Instanz kann dabei vom Mapping-

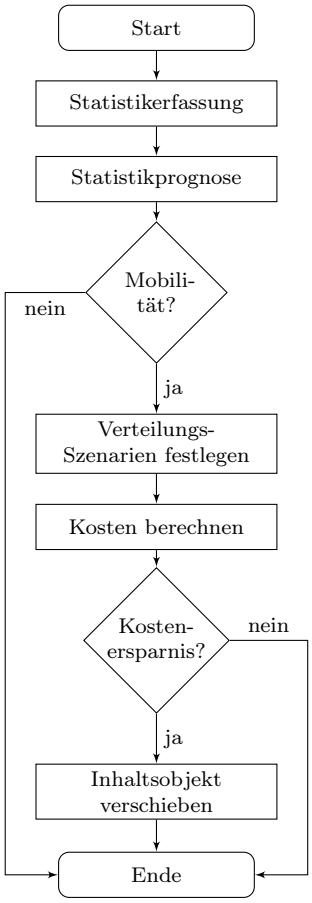


Abbildung 4.14.: Ablaufdiagramm des Hauptalgorithmus zur Inhaltsmobilität

System übernommen werden. Da dieses üblicherweise als DHT realisiert ist, kann jeder Algorithmus auf jedem Knoten der DHT für die Inhaltsobjekte ausgeführt werden, deren UID er speichert. Auf diese Weise ist auch Ausfallsicherheit gewährleistet. Der Speicherserver führt eine Statistik über den Ursprung und die Häufigkeit der Anfragen. Sollte die Last des Servers zu groß werden oder die mittlere Distanz $\overline{D}$ der Anfragen zunehmen und eine definierte maximale mittlere Distanz $\overline{D}^{(\max)}$ überschreiten, kann der Server diese Statistikwerte an die Verwaltungsinstanz übergeben, die anschließend den Auswahlalgorithmus ausführt. Dieser ist als Pseudocode in Algorithmus 4.1 dargestellt. Ebenso kann einem Inhaltsobjekt eine erwartete Aufrufrate $\lambda_{\mathrm{i}}^{(\mathrm{e})}$ zusammen mit einer Obergrenze $\lambda_{\mathrm{i}}^{(\max)}$ und Untergrenze $\lambda_{\mathrm{i}}^{(\min)}$ zugeteilt werden, damit der Algorithmus gestartet wird. Ist mindestens eine Bedingung des Auswahlalgorithmus erfüllt, startet der Auswahlalgorithmus den Hauptalgorithmus, der ein neues Verteilungsszenario für das betreffende Inhaltsobjekt erstellt.

Der Ablauf des Hauptalgorithmus ist in Abbildung 4.14 dargestellt. Die Arbeitsschritte des Algorithmus werden in den folgenden Abschnitten erläutert.

4.6.5.1. Statistikerfassung und Prognose

Nach dem Aufruf des Hauptalgorithmus durch den Auswahlalgorithmus werden zunächst Informationen über die Aufrufrate $\lambda_{i,n}$ für den Inhalt i auf jedem Speicherserver n sowie die Herkunft der Anfragen und die Lastinformationen der Speicherserver n gesammelt. Die Summe der einzelnen Aufrufraten ergibt die Gesamtaufrufrate λ_i des Inhaltsobjekts i. Zusammen mit den Ortsinformationen der Anfragen wird ein Profil erstellt, dass die Verteilung der Aufrufe pro Region und Zeitintervall T angibt.

Die Aufrufraten für die betroffenen Inhalte werden von der zentralen Instanz direkt von den Speicherservern abgefragt und in eine Statistik über Aufrufhäufigkeit und Herkunft der Anfragen aufgenommen. Die Statistikdaten könnten auch direkt vom Mapping-System bezogen werden, aufgrund von Caching von Mapping-Einträgen sind diese Daten jedoch unter Umständen ungenau. Im Gegenzug sind dem Mapping-System anhand des Mapping-Eintrags dafür alle Replika bekannt, so dass die entsprechenden Speicherserver auf Anfrage direkt aktuelle Informationen, auch über ihre eigene Auslastung, liefern können.

Um passende Speicherserver für die Verteilung von Inhalten zu finden, sind jedoch auch Auslastungsdaten von solchen Servern erforderlich, die aktuell andere Inhalte speichern und nicht zusammen mit den Daten dieser Inhaltserhebung geliefert werden. Diese Daten können dem Algorithmus aber durch Monitoringinheiten im Netz zur Verfügung gestellt werden. Derartige Ansätze werden bereits in heutigen CDN verwendet. Daher wird an dieser Stelle auf [Pen08] verwiesen. Auslastungsdaten werden nur von denjenigen Regionen erhoben, die aufgrund des erstellten Aufrufprofils für die Platzierung besonders in Frage kommen.

Aus den ermittelten Statistikwerten können Zugriffswerte prognostiziert werden. Diese Prognose ist wichtig, um auf eine sich ändernde Beliebtheit schnell reagieren zu können. Für die Prognose wird auf eine bestimmte Anzahl bereits vergangener Messwerte zurückgegriffen, aus denen der Prognosewert berechnet wird. Da im Folgenden davon ausgegangen wird, dass genügend Ressourcen für die Berechnung der Neuverteilung vorhanden sind, werden nur wenige Prognosewerte (in der Größenordnung eins bis zwei) berechnet, um den Schätzfehler, der mit der Anzahl der prognostizierten Werte deutlich anwächst, gering zu halten.

Ein prognostizierter Schätzwert lässt sich mit folgender Formel berechnen. Dabei bezeichnet a_t den Grundwert und b_t den Trendwert. m_t bezeichnet die Änderung der Werte pro Zeitintervall.

$$\hat{x}_{\mathrm{t}+1} = a_\mathrm{t} + b_\mathrm{t} \tag{4.7}$$

$$m_\mathrm{t} = x_\mathrm{t} - x_{\mathrm{t}-1} \tag{4.8}$$

Verschiedene Prognoseverfahren unterscheiden sich darin, wie der jeweilige Grund- und Trendwert berechnet wird. Im Zuge der Arbeit wurden drei Verfahren betrachtet: die Methode mit festen Koeffizienten, die exponentielle Glättung 2. Ordnung sowie das Winters-Verfahren. Ausgewählt wurde letztendlich die Methode mit festen Koeffizienten, da diese das beste Ergebnis bezogen auf schnelle Reaktion auf Änderungen und Schätzfehler bietet.

Bei der Methode mit festen Koeffizienten wird als Grundwert der gleitende Mittelwert der letzten p Werte verwendet, um statistische Ausreißer besser kompensieren zu können. Für den Grundwert a_t gilt damit:

$$a_\mathrm{t} = \frac{1}{p} \sum_{n=t-p+1}^{t} x_\mathrm{n} \tag{4.9}$$

Der Trendwert b_t wird aus den mit μ gewichteten letzten Änderungen m berechnet. Für den Algorithmus werden die letzten drei Änderungen berücksichtigt, die mit den Werten $\mu_1 = 0,5$, $\mu_2 = 0,3$ und $\mu_3 = 0,2$ gewichtet werden. b_t wird damit berechnet mit:

$$b_\mathrm{t} = \mu_1 m_\mathrm{t} + \mu_2 m_{\mathrm{t}-1} + \mu_3 m_{\mathrm{t}-2} \tag{4.10}$$

Zusammen mit der Formel 4.7 kann die Prognose der Zugriffswerte zum Zeitpunkt $t + 1$ berechnet werden. Dieser Prognosewert wird im Folgenden für das Aufstellen der Verteilungsszenarien verwendet.

4.6.5.2. Verteilungsszenarien

Nach erfolgter Statistikprognose stellt der Algorithmus für ein bestimmtes Inhaltsobjekt verschiedene Verteilungsszenarien auf. Unter einem Verteilungsszenario versteht man dabei die Auswahl einer oder mehrerer Regionen sowie die Anzahl der Speicherserver innerhalb einer Region. Dieses so genannte „Replica Placement“ ist auch ein aktuelles Forschungsthema im Bereich von CDN und wird den entsprechenden Betreibern oft geheim gehalten, da es eine komplexe Aufgabe darstellt.

Die Autoren in [QPV01, Pen08] stellen in diesem Zusammenhang verschiedene graphentheoretische und heuristische Lösungsalgorithmen vor, die aus R möglichen Orten R^* auswählen und eine bestimmte Kostenfunktion minimieren. Da die graphentheoretischen Ansätze sehr rechenaufwändig und für den Einsatzzweck des Replica Placement in der Realität nicht anwendbar sind, kommt ein heuristischer Algorithmus zum Einsatz. Dieser liefert zwar nicht die optimale Lösung, ist dafür sehr schnell berechenbar. Zur Auswahl stehen dabei der Greedy- sowie der Hot-Spot-Algorithmus. Eine Evaluierung dieser Heuristiken in [QPV01] hat ergeben, dass der Greedy-Algorithmus die beste Performance für das Problem des Replica Placement liefert. Aus diesem Grund greift auch der Algorithmus zur Inhaltsmobilität auf den Greedy-Algorithmus zurück.

Beim Greedy-Algorithmus wird in jeder Iteration ein neuer Ort zur Auswahl der möglichen Orte hinzugefügt. Bereits zuvor werden ungeeignete Orte ausgeschlossen, falls ein Inhaltsobjekt beispielsweise aus rechtlichen Gründen dort nicht gespeichert werden darf. Im ersten Durchlauf werden zunächst die Kosten für alle Orte einzeln berechnet und der günstigste ausgewählt. In jedem weiteren Schritt wird ein neuer Ort hinzugefügt, der zusammen mit dem bereit ausgewählten zu den niedrigsten Kosten führt, bis die Zahl R^* erreicht ist und keine günstigere Verteilung mehr gefunden werden kann. Dieser Ablauf ist als Pseudocode in Algorithmus 4.2 dargestellt.

Algorithmus 4.2 Greedy-Algorithmus zur Auswahl geeigneter Regionen

Eingabe: R^*, Regionsliste, Aufrufstatistik

```
1: Sortiere ungeeignete Regionen aus Liste
2: Berechne Kosten für alle Regionen in Regionsliste
3: Füge Region mit geringsten Kosten zur Auswahlliste hinzu
4: for Auswahllistenlänge ≤ R* do
5:    for Alle Regionen nicht auf der Auswahlliste do
6:       Berechne Neuverteilung der Anfragen
7:       Berechne Kosten für diese Verteilung
8:    end for
9:    Füge Region mit geringsten Kosten zur Liste hinzu
10:   Entferne Region mit geringsten Kosten von Regionsliste
11: end for
```

Aufgrund der Annahme, dass in jeder Region dedizierte Speicherserver vorhanden sind, muss die Frage, wo Server im Netz platziert werden, nicht geklärt werden. Es muss vielmehr bestimmt werden, auf welche Server innerhalb der Regionen ein Inhaltsobjekt platziert werden soll. Der Berechnungsaufwand des Greedy-Algorithmus liegt in der Größenordnung von $\mathcal{O}(R^* \cdot R^2)$. Da in Hiimap maximal 254 Regionen möglich sind, kann eine obere Schranke für den Rechenaufwand angegeben werden.

Verteilung auf Regionen Zunächst werden mit dem Greedy-Algorithmus R^* Regionen bestimmt, in denen ein bestimmtes Inhaltsobjekt zur Verfügung gestellt werden kann. Als Kostenmetrik dient dabei zunächst nur die Aufrufrate sowie die Distanz zu den Aufrufenden. Netz- und Serverauslastung sind in diesem Schritt noch nicht relevant. Darüber hinaus wird angenommen, dass eine Anfrage immer von der nächsten vorhandenen Kopie des Inhaltsobjekts bedient wird. Durch die Ortsinformationen in der GGUID ist dies möglich.

Es wird ferner angenommen, dass die Anfragerate, die eine Region bedienen kann, theoretisch unbegrenzt ist. Dies ist in der Realität nicht der Fall, es ist jedoch so gut wie ausgeschlossen, dass ein Inhaltsobjekt ausschließlich in einer Region bedient wird und dort sämtliche zur Verfügung stehenden Ressourcen verbraucht.

Aufgrund von verschiedenen Einflüssen zeigen Regionen unterschiedliche Präferenzen für Inhaltsobjekte. Besonders Sprache und Kultur sorgen dafür, dass Inhaltsobjekte üblicherweise nur regional beliebt sind. Daher ist eine Verteilung auf verschiedene Regionen sinnvoll, um Kosten durch unnötig lange Übertragungswege zu sparen.

Verteilung auf Speicherserver in einer Region Sind die Regionen festgelegt, muss das Inhaltsobjekt innerhalb einer Region auf einen oder mehrere Server verteilt werden. Abhängig von der Anzahl R^* der ausgewählten Regionen ist die Last pro Region bestimmend, die letztendlich zur Anzahl der benötigten Server pro Region führt. Da vorausgesetzt wird, dass innerhalb einer Region dedizierte Speicherserver zur Verfügung stehen, muss deren Platzierung nicht berücksichtigt werden.

Die Speicherserver werden anhand ihrer Auslastung ρ_n sowie der jeweiligen Auslastung

Algorithmus 4.3 Greedy-Algorithmus zur Auswahl der Speicherserver in einer Region

Eingabe: Region r, Aufrufrate $\lambda_{\mathrm{i,r}}$, Serverliste

1: Berechne Anzahl benötigter Server S_{r}
2: Sortiere Serverliste aufsteigend nach Auslastung
3: Wähle die ersten S_{r} Server der Liste
4: **if** zu wenig Ressourcen vorhanden **then**
5: Wähle weiteren Server aus der Liste
6: **end if**

des Netzes ρ_{AS}, die der Auslastung der Region entspricht, ausgewählt. Dabei werden zuerst die Server mit der geringsten Summe $\rho_n + \rho_{\mathrm{AS}}$ ausgewählt. Anhand des Servermodells besitzt jeder Speicherserver eine maximale Bedienrate von ϵ_n bei einer mittleren Dateigröße $\overline{S}$. Um die verfügbare Kapazität, die auf einem bestimmten Server gespeichert werden kann, auf viele Inhaltsobjekte aufzuteilen, kann einem bestimmten Objekt nur eine maximale Bedienrate von $\epsilon_{\max} = q \cdot \frac{\epsilon_n \cdot \overline{S}}{S_i}$ zugewiesen werden. Diese maximale Bedienrate ist abhängig von der Größe des Objekts S_i sowie dem Prozentwert q der Maximalauslastung, die dieses Objekt maximal verursachen darf. Die Anzahl benötigter Server S_r in der Region r für ein Inhaltsobjekt ergibt sich dadurch mit:

$$S_r = \left\lceil \frac{\lambda_{i,r}}{\epsilon_{\max}} \right\rceil \tag{4.11}$$

Die Festlegung, dass ein Inhaltsobjekt einen Speicherserver nur zu einem bestimmten Teil auslasten darf, ist aus folgenden Gründen sinnvoll:

- Eine Verteilung der Inhaltsobjekte wird damit erzwungen. Dies fördert zum einen die Ausfallsicherheit, zum anderen wandern die Objekte schneller zum Benutzer. Auftretende Spitzen in der Aufrufstatistik können darüber hinaus von mehreren Servern besser ausgeglichen werden.
- Würde ein Objekt einen Speicherserver sehr stark auslasten, würden andere Objekte auf dem gleichen Server benachteiligt, da auch die Ausfallwahrscheinlichkeit steigt.
- Für eine optimale Verteilung müsste die Verteilung aller existierenden Objekte für jedes einzelne Objekt neu berechnet werden. Diese Aufgabe ist nicht in sinnvoller Zeit berechenbar. Daher wird zugunsten der Berechenbarkeit die Maximalauslastung pro Server begrenzt, so dass Kapazitäten für weitere Objekte vorhanden sind.

Der Algorithmus zur Auswahl der Speicherserver innerhalb einer Region ist im Pseudocode 4.3 dargestellt.

4.6.5.3. Kostenfunktion und Metriken

Für jedes aufgestellte Verteilungsszenario bezüglich Regionen und Speicherserver innerhalb einer Region müssen die Kosten berechnet werden, um das günstigste Szenario auswählen zu können. Zusammen mit den Informationen aus den vorhergegangenen Schritten

kann jetzt die Kostenfunktion aufgestellt werden. Die Kosten werden allgemein repräsentiert durch die Variable C mit der abstrakten Einheit KE (Kosteneinheit), die auf US-Dollar normiert ist.

Kostenfunktion für die Bereitstellung eines Inhaltsobjekts mit einem Server Im folgenden werden die mittleren monatlichen Kosten $C_{i,n}$ für die Bereitstellung des Inhaltsobjekts i mit dem Speicherserver n und der mittleren Aufrufrate $\lambda_{i,n}$ berechnet.

$$C_{i,n} = \Big[S_i \cdot \overline{D}_n \cdot C_{\mathrm{D}} \cdot \psi_{\mathrm{S}}(\rho_n, B_i) \cdot \psi_{\mathrm{AS}}(\rho_{\mathrm{AS}}, B_i) + \phi_R(\rho_n)\Big] \cdot \lambda_{i,n} + \phi_{\mathrm{IS}}(\rho_n, \rho_{i,n}) \tag{4.12}$$

Diese Kosten setzen sich aus den Übertragungskosten je Anfrage, den Kosten ϕ_{R} für eine abgelehnte Anfrage, sowie den Infrastrukturkosten ϕ_{I} zusammen. Die Übertragungskosten sind abhängig von der Dateigröße S_{i}, der mittleren Distanz $\overline{D}_{\mathrm{n}}$ jeder Anfrage zum Server n sowie den Kosten C_{D} pro kB und Hop. Zusätzlich werden die Übertragungskosten mit den Gewichtsfunktionen ψ_{S} und ψ_{AS} gewichtet, die es erlauben, die Auslastung des Speicherservers bzw. des AS (in diesem Fall der Region) zusätzlich zu gewichten. Diese Gewichtsfunktionen erlauben es, je nach Inhaltsklasse B_{i}, eine Anfrage abhängig von der jeweiligen Last zusätzlich zu verteuern. Es sind drei Klassen definiert: Download mit $B = 1$, Web mit $B = 2$ und Streaming mit $B = 3$. Die Last kann daher für Klassen mit $B > 1$ zusätzlich verteuert werden, so dass der Algorithmus früher entscheidet, dieses Objekt auf einen weiteren Server zu replizieren. Die Gewichtungsfunktionen orientieren sich dabei an den Wartezeiten eines M/M/1 Wartesystems:

$$\psi_{\mathrm{S/AS}} = \frac{\rho_{n/\mathrm{AS}}}{1 - \rho_{n/\mathrm{AS}}} \cdot B_i + 1 \tag{4.13}$$

Die entstehenden Kosten für das Bereitstellen der Infrastruktur werden mit der Funktion ϕ_{IS} berechnet. Hier wird nicht unterschieden ob es sich um den Ursprungsserver des Inhaltsobjekts, oder um einen dedizierten Speicherserver des inhaltsorientierten Netzes handelt. Diese Kosten setzen sich aus Fixkosten für die Server, sowie lastabhängigen Kosten zusammen. Die gesamten Fixkosten werden unter allen Inhaltsobjekten I, die auf einem Server gespeichert werden, gleich verteilt. Dies entspricht dem Faktor p mit $p = \frac{1}{I}$. Kosten für den Verkehr sind bereits in den Kosten pro Aufruf enthalten.

$$\phi_{\mathrm{IS}} = p \cdot C_{\mathrm{Server,fix}} + C_{\mathrm{Server,Anbindung}} \cdot \frac{\rho_{i,n}}{\rho_n} \tag{4.14}$$

Kosten für eine abgelehnte Anfrage ϕ_R werden mit der Ablehnungswahrscheinlichkeit P_{R} aus Formel 4.6 berechnet. Diese setzen sich aus den fünffachen Kosten für eine erfolgreiche Übertragung zusammen. Die Kosten für eine abgelehnte Anfrage werden damit künstlich verteuert, da abgelehnte Anfragen vermieden werden sollen, um die Zuverlässigkeit der Bereitstellung zu gewährleisten.

Algorithmus 4.4 Entscheidungsfindung

Eingabe: Szenarien, Statistik, Schwellwert

```
1: for Alle Szenarien do
2:     Berechne Kosten für aktuelle und prognostizierte Werte
3:     Bilde Mittelwert
4: end for
5: Wähle Szenario mit kleinsten mittleren Kosten
6: Berechne Gewinn im Vergleich zur jetzigen Verteilung
7: if Gewinn ≥ Schwellwert then
8:     Entscheide für dieses Szenario
9: end if
```

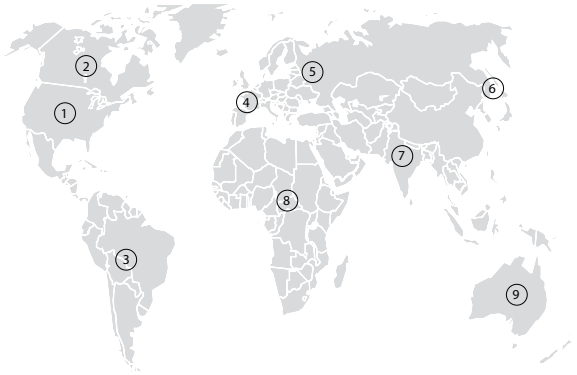
Gesamtkosten eines Verteilungsszenarios Nachdem die Kosten für jeden einzelnen Server berechnet sind, können die Gesamtkosten eines Verteilungsszenarios für ein Inhaltsobjekt C_i mit Formel 4.15 angegeben werden. Diese setzen sich zusammen aus der Summe der Kosten pro Server sowie den Kosten für die Signalisierung V und den Kosten der Verteilung auf die ausgewählten Server. Die Kosten für Signalisierung und Verteilung können jedoch im Folgenden vernachlässigt werden, da sie im Vergleich zu den Kosten für die Zustellung des Inhaltsobjekts zum Benutzer um mehrere Größenordnungen geringer sind. Die Verteilungskosten entsprechen dabei genau einem Aufruf des Inhaltsobjekts.

$$C_i = \sum_{\text{Alle Server}} C_{i,n} + V + \sum_{\text{Alle Server}} (S_i \cdot D_{i,n} \cdot C_{\text{D}}) \approx \sum_{\text{Alle Server}} C_{i,n} \tag{4.15}$$

4.6.5.4. Entscheidung und Durchführung

In der Entscheidungsfindung, ob ein neues Verteilungsszenario gewählt werden soll, werden die berechneten Kosten der Szenarien miteinander verglichen. Mit den Werten der Statistikprognose werden außerdem die aktuellen Werte für ein Szenario mit den zu erwartenden prognostizierten Werten verglichen. Um einen ständigen Wechsel zwischen zwei Szenarien zu vermeiden, wird ein neues Szenario erst ausgewählt, wenn der dadurch erreichte Gewinn an KE einen bestimmten Schwellwert überschreitet. Wird der Schwellwert nicht überschritten, bleibt das aktuelle Szenario bestehen. Der Algorithmus zur Entscheidungsfindung ist in Algorithmus 4.4 als Pseudocode dargestellt.

Ist der durch das neue Szenario erreichte Gewinn größer als der Schwellwert, wird dieses Szenario gewählt. Die zentrale Instanz reserviert anschließend auf den gewählten Speicherservern die Ressourcen für das Inhaltsobjekt und weist den am geringsten ausgelasteten Speicherserver des aktuellen Szenarios an, das Inhaltsobjekt auf die neuen Speicherserver zu kopieren. Ein eventuelles Löschen des Inhaltsobjekts auf einigen Servern wird ebenfalls von der Instanz veranlasst. Anschließend werden die neuen Locator-Werte in den Mapping-Einträgen der entsprechenden IO aktualisiert. Diese beschreiben das Verteilungsszenario für ein bestimmtes Objekt eindeutig und werden als Ausgangswert für eine erneute Berechnung verwendet.



$$\boldsymbol{D} = \begin{bmatrix} 1 & 2 & 4 & 4 & 6 & 6 & 8 & 9 & 4 \\ 2 & 1 & 4 & 5 & 7 & 6 & 9 & 11 & 5 \\ 4 & 4 & 1 & 6 & 8 & 9 & 11 & 14 & 7 \\ 4 & 5 & 6 & 1 & 2 & 5 & 6 & 7 & 5 \\ 6 & 7 & 8 & 2 & 1 & 3 & 5 & 8 & 6 \\ 6 & 6 & 9 & 5 & 3 & 1 & 2 & 9 & 5 \\ 8 & 9 & 11 & 6 & 5 & 2 & 1 & 8 & 3 \\ 9 & 11 & 14 & 7 & 8 & 9 & 8 & 1 & 7 \\ 4 & 5 & 7 & 5 & 6 & 5 & 3 & 7 & 1 \end{bmatrix}$$

Abbildung 4.15.: Verteilung der Regionen mit zugehöriger Distanzmatrix $\boldsymbol{D}$

4.6.6. Simulation und Auswertung

Der im vorherigen Abschnitt vorgestellte Algorithmus wurde vollständig in Matlab implementiert und mit realistischen Werten auf seine Tauglichkeit überprüft. In diesem Abschnitt werden zunächst die verwendeten Parameter vorgestellt und anschließend die Ergebnisse der Simulation diskutiert und bewertet.

4.6.6.1. Parameter

Für die Simulation sind verschiedene Eingangsparameter nötig, die dem Algorithmus zur Verfügung gestellt werden müssen. Dies ist zum einen eine Distanzmatrix, welche die nötigen mittleren Hops zwischen zwei Regionen beinhaltet, sowie die Kosten pro Hop. Des weiteren werden Fixkosten für die Server sowie von der Anschlussdatenrate abhängige Kosten benötigt. Darüber hinaus werden Aufrufstatistiken für ein Inhaltsobjekt benötigt, um die Performance des Algorithmus zu testen.

Regionen und Distanz Zur Vereinfachung der Simulation wurden neun verschiedene Regionen ausgewählt, die sich in etwa an den Kontinenten orientieren und in Abbildung 4.15 dargestellt sind. Zur Berechnung der mittleren Distanz einer Anfrage wurde eine Distanzmatrix $\boldsymbol{D}$ festgelegt, in der die jeweils mittlere Anzahl benötigter Hops zwischen zwei Regionen eingetragen ist. Die Distanzen orientieren sich an den Messungen des Archipelago-Projekts [CAI09] sowie an der geographischen Entfernung zu großen Internet-Austauschknoten. Regionen mit schwacher Infrastruktur wie Afrika haben dabei deutlich höhere Pfadlängen wie die Regionen USA oder Europa.

Es gilt zu beachten, dass die Beschränkung auf neun Regionen und die damit verbundene Distanzmatrix eine starke Vereinfachung darstellt. Der Algorithmus ist jedoch in der Lage, auch mit einer deutlich höheren Anzahl zu arbeiten. Der Rechenaufwand ist aufgrund der Grenze von maximal 254 Regionen nach oben beschränkt.

Für die Abschätzung der Kosten für einen AS-Hop wird, ausgehend von den Messungen zu den mittleren Pfadlängen, angenommen, dass ein Pfad im Mittel vier AS-Hops lang ist und die Transitkosten den Übertragungskosten mit 10 USD pro 1 Mbit/s pro Monat

entsprechen. Die Kosten C_{D} pro Hop und kB/s lassen sich damit abschätzen mit:

$$C_{\mathrm{D}} = \frac{10 \text{ USD} \cdot 8}{1000 \frac{\mathrm{kB}}{\mathrm{s}} \cdot 4 \text{ Hop}} = 0,02 \left[\frac{\text{USD}}{\frac{\mathrm{kB}}{\mathrm{s}} \cdot \text{Hop}} \right] \tag{4.16}$$

Speicherserver Es wurde angenommen, dass in jeder Region beliebig viel Kapazität zur Bedienung von Anfragen vorhanden ist. Im Simulationsmodell stehen jeder Region daher 1.000 Speicherserver mit einer mittleren Bedienrate von $\epsilon = 1500 \frac{1}{\mathrm{s}}$ zur Verfügung. Jeder Server wird mit einer zufälligen Auslastung zwischen 50% und 100% initialisiert. Weltweit existieren in diesem Modell daher 9.000 Server. Der Anteil der Fixkosten für Dateien mit einer Größe von bis zu 300kB wird mit $p = 0,001$ festgelegt, für größere Dateien mit $p = 0,01$.

Als Fixkosten pro Server werden monatliche Kosten für Strom, Wartung, usw. von 2.000 USD angenommen, die mittels Faktor p auf die einzelnen Inhaltsobjekte aufgeteilt werden. Die Kosten für den Anschluss an einen Austauschknoten werden mit 500 USD für einen 1 Gbit/s-Port modelliert. Dieser Wert beruht auf den aktuellen Kosten im Jahre 2011 von knapp 1.000 USD, geht aber von der Annahme aus, dass die Tendenz sinkender Preise weiter anhält.

Zugriffsfunktionen und Aufrufverteilung Zur Simulation des Algorithmus wurden verschiedene Zugriffsfunktionen festgelegt, die sich an tatsächlichen Aufrufstatistiken von Videos aus dem Onlineportal Youtube orientieren. Diese sind in Abbildung 4.16 dargestellt. Im Fall (a) steigt die tägliche Zugriffszahl gleichmäßig an, verharrt auf diesem Niveau, und fällt langsam wieder ab. Im Fall (b) handelt es sich um einen sprunghaften Anstieg der täglichen Zugriffszahl. Diese Phänomen tritt auf, wenn beispielsweise ein Inhaltsobjekt durch ein anderes, sehr beliebtes Inhaltsobjekt referenziert wird. Im letzten Fall (c) wird ein kurzer sprunghafter Anstieg betrachtet, der aber sogleich wieder auf das Normalniveau abfällt.

Zusätzlich zur Aufrufstatistik ist eine Verteilung der Aufrufe auf die verschiedenen Regionen notwendig. Die Simulation erfolgte mit drei verschiedenen Verteilungen der Aufrufrate. In Tabelle 4.2 ist dargestellt, wie sich die angenommene Häufigkeit der Anfragen auf die entsprechenden Regionen aufteilt. Die erste Verteilung entspricht einem global beliebten Inhaltsobjekt. Regionen mit starker Internetnutzung sind dabei auch für

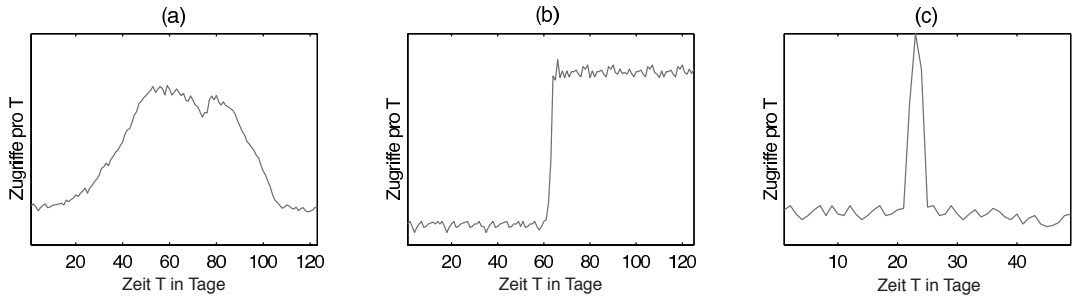


Abbildung 4.16.: Zugriffsfunktionen

Tabelle 4.2.: Aufrufverteilung auf die Regionen

	R1	R2	R3	R4	R5	R6	R7	R8	R9
Verteilung 1	20%	5%	5%	20%	15%	20%	5%	5%	5%
Verteilung 2	80%	10%	0%	10%	0%	0%	0%	0%	0%
Verteilung 3	40%	7%	0%	40%	8%	0%	0%	0%	5%

den Großteil der Anfragen verantwortlich. Verteilung zwei repräsentiert ein ausschließlich in einer Region beliebtes Inhaltsobjekt. In der dritten Verteilung ist ein Inhaltsobjekt ebenfalls nur regional beliebt, dafür aber in zwei Regionen.

4.6.6.2. Auswertung und Diskussion

In diesem Abschnitt werden die Ergebnisse der Simulation auf Basis des entwickelten Algorithmus zur Inhaltsmobilität vorgestellt und ausgewertet. Im ersten Schritt kommt dabei eine lineare Zugriffsfunktion zum Einsatz, um die Arbeitsweise des Algorithmus zu erläutern. Die Ergebnisse des Greedy-Algorithmus, der die einzelnen Verteilungszenarien aufstellt, werden zunächst als Zwischenschritt extra betrachtet. Anschließend wird auch der Entscheidungsprozess in einem weiteren Schritt mit einbezogen, so dass das jeweils günstigste Szenario ausgewählt wird. Zum Schluss werden die entsprechenden Ergebnisse für die jeweiligen Zugriffsfunktionen vorgestellt.

Kostenverlauf der einzelnen Szenarien In den Tabellen 4.3, 4.4 und 4.5 sind die verschiedenen Verteilungsszenarien, abhängig von der Aufrufverteilung und der Anzahl der gewählten Regionen R^* dokumentiert. Die Anzahl der ausgewählten Regionen für die Bereitstellung eines Inhaltsobjekts R^* entspricht dabei der Nummer des jeweiligen Szenarios. Die Werte in der Tabelle geben an, wieviel Prozent der Anfragen von der entsprechenden Region beantwortet werden müssen. Für die Verteilung wird angenommen, dass eine Anfrage immer an die nächste verfügbare Kopie geroutet wird. Entspricht R^* gleich der Anzahl an Regionen, in denen der Inhalt aufgerufen wird, so entspricht dieses Szenario exakt der entsprechenden Verteilung. Die Ergebnisse des Greedy-Algorithmus für die einzelnen Verteilungen ist dabei unabhängig von der Anzahl der Aufrufe und für alle Zugriffsfunktionen gleich. Erst mit dem Algorithmus-Schritt der Entscheidung wird das von der Zugriffsfunktion günstigste Szenario ausgewählt.

Die Kosten der jeweiligen Szenarien, aufgetragen über eine linear steigende Zugriffszahl, sind in Abbildung 4.17 für drei verschiedene Inhaltsklassen mit unterschiedlicher Dateigröße für alle drei Verteilungen aufgetragen. Der Zeitbezug entspricht $T = 24$h. Die Dateigröße der Inhaltsklasse Web mit 300 kB entspricht dabei einer Webseite mit allen eingebundenen Elementen, Die Inhaltsklasse Download mit 5 MB entspricht einer Musikdatei oder einem Bild, und die Klasse Streaming mit 20 MB entspricht einem Video von etwa drei Minuten Länge.

Unabhängig von der Dateigröße ähneln sich alle Verläufe einer bestimmten Verteilung. Der Verlauf der Kosten ist jedoch stark von der Aufrufverteilung abhängig. Je nach Distanz zu einer bestimmten Region lassen sich dadurch teils erhebliche Kosten einspa-

Tabelle 4.3.: Ergebnisse Greedy-Algorithmus für Verteilung 1

	R1	R2	R3	R4	R5	R6	R7	R8	R9
Szenario 1 ———				100%					
Szenario 2 ———				75%		25%			
Szenario 3 ———	35%			40%		25%			
Szenario 4 - - - - -	35%			35%		25%		5%	
Szenario 5 - - - - -	30%		5%	35%		25%		5%	
Szenario 6 - - - - -	30%		5%	20%	15%	25%		5%	
Szenario 7 -·-·-·-	25%		5%	20%	15%	25%		5%	5%
Szenario 8 -·-·-·-	20%	5%	5%	20%	15%	25%		5%	5%
Szenario 9 -·-·-·-	20%	5%	5%	20%	15%	20%	5%	5%	5%

Tabelle 4.4.: Ergebnisse Greedy-Algorithmus für Verteilung 2

	R1	R2	R3	R4	R5	R6	R7	R8	R9
Szenario 1 ———	100%								
Szenario 2 ———	90%			10%					
Szenario 3 ———	80%	10%		10%					

Tabelle 4.5.: Ergebnisse Greedy-Algorithmus für Verteilung 3

	R1	R2	R3	R4	R5	R6	R7	R8	R9
Szenario 1 ———				100%					
Szenario 2 ———	52%			48%					
Szenario 3 ———	47%			48%					5%
Szenario 4 - - - - -	47%			40%	8%				5%
Szenario 5 - - - - -	40%	7%		40%	8%				5%

ren. Dies ist besonders bei Verteilung 3 sichtbar, in der es zwei Hauptaufrufgebiete gibt. Sobald eine zweite Region hinzugefügt wird, sinken die Kosten deutlich für höhere Zugriffszahlen. Die Hinzunahme weiterer Regionen führt jedoch zu keiner nennenswerten Verbesserung. Für niedrige Zugriffszahlen sind die Szenarien mit mehreren Regionen sogar deutlich teurer. Dennoch spielt die Dateigröße eine Rolle bei der Kostenbetrachtung. So ist es für größere Dateien deutlich früher rentabel, das Inhaltsobjekt auf mehrere Regionen zu replizieren.

Der stufenförmige Charakter des Verlaufs, der in den Diagrammen zu erkennen ist, stammt von der Anzahl der verwendeten Speicherserver, die für die Verteilung verwendet werden. Sobald bei der Berechnung des Szenarios ein neuer Speicherserver hinzugenommen wird, springt die Kostenfunktion aufgrund der Fixkosten des Servers. Die Fixkosten erklären auch, warum keine Kurve im Nullpunkt startet, da je nach Verteilungsszenario

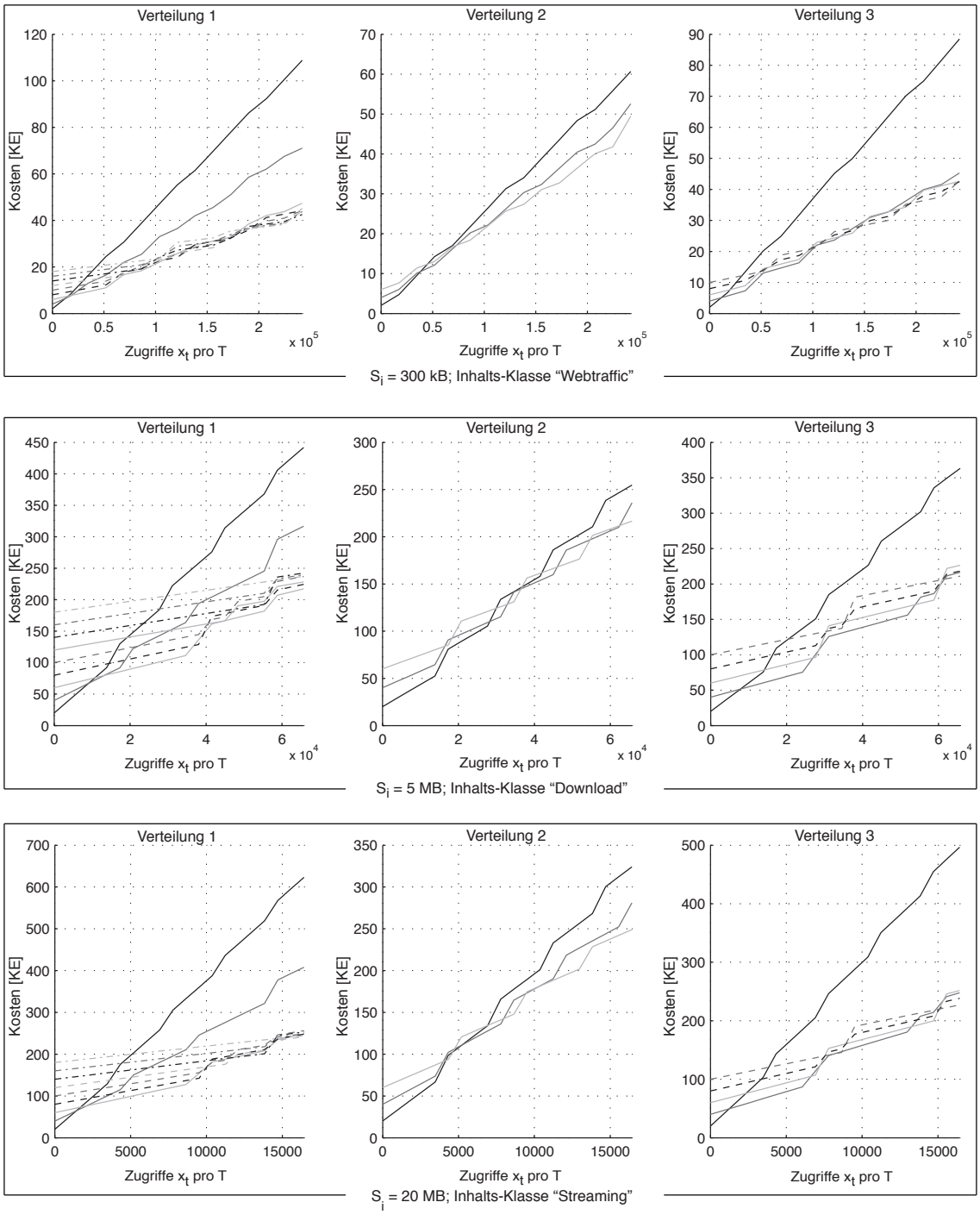


Abbildung 4.17.: Verlauf der Kosten der einzelnen Szenarien über linear steigende Zugriffszahlen

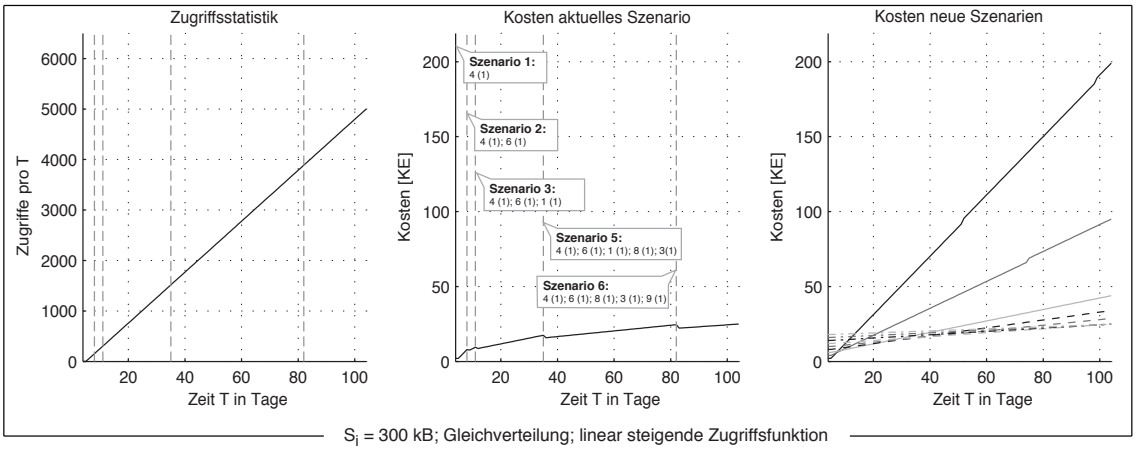


Abbildung 4.18.: Entscheidung des Algorithmus bei linear steigender Zugriffsfunktion

in jeder Region mindestens ein Server benötigt wird.

Entscheidung Nach der Aufstellung der einzelnen Verteilungsszenarien folgt die Entscheidung für das jeweils günstigste. In diesem Abschnitt werden daher die Ergebnisse des kompletten Algorithmus zur Inhaltsmobilität vorgestellt. Im Gegensatz zur Ablaufbeschreibung wird der Algorithmus jedoch periodisch mit dem Intervall T aufgerufen. Abbildung 4.18 zeigt die Entscheidungspunkte des Algorithmus und die daraus resultierenden Kosten für die linear steigende Zugriffsfunktion sowie bei einer Gleichverteilung der Aufrufe über alle Regionen. In diesem konkreten Fall kommt keines der aufgestellten Verteilungsszenarien zum Einsatz, stattdessen wird davon ausgegangen, dass die Aufrufe auf alle Regionen gleich verteilt sind. Jede Region trägt damit etwa 11% zur Gesamtaufrufrate bei. Die Ergebnisse können damit besser erläutert werden. Dargestellt sind jeweils die Aufrufe über die Zeit (Zugriffsstatistik) , die entstandenen Kosten unter Zuhilfenahme des Algorithmus (Kosten aktuelles Szenario) , sowie die Kosten aller möglichen Szenarien (Kosten neue Szenarien).

Vertikal gestrichelte graue Linien markieren Zeitpunkte, zu denen ein Wechsel des Verteilungsszenarios stattfindet. Die Beschriftung dazu gibt die ausgewählten Regionen sowie in Klammern die Anzahl der verwendeten Server an. Im aktuell dargestellten Fall wird in jeder Region jeweils nur ein Server verwendet. Für höhere Zugriffszahlen ändert sich deren Anzahl. Die Entscheidung, wann das Inhaltsobjekt verschoben wird, erfolgt anhand des Mittelwerts der aktuellen und prognostizierten Kosten. Im aktuellen Fall werden alle Anfragen zunächst aus der Region 4 bedient. Mit wachsender Zugriffszahl werden nacheinander zwei, drei, fünf und sechs Regionen hinzugenommen. Eine Verteilung auf vier Regionen wird nicht gewählt, da aufgrund der Hysterese der Entscheidung der erzielte Gewinn nicht ausreicht.

Auffällig hierbei ist die geringe Anzahl benötigter Aufrufe, um die Verteilung eines Inhaltsobjekts kostengünstiger zu gestalten. Dies ist besonders auf die Gleichverteilung der Aufrufherkunft zurückzuführen, da die mittlere Distanz $\overline{D}$ zum nächsten Server durch

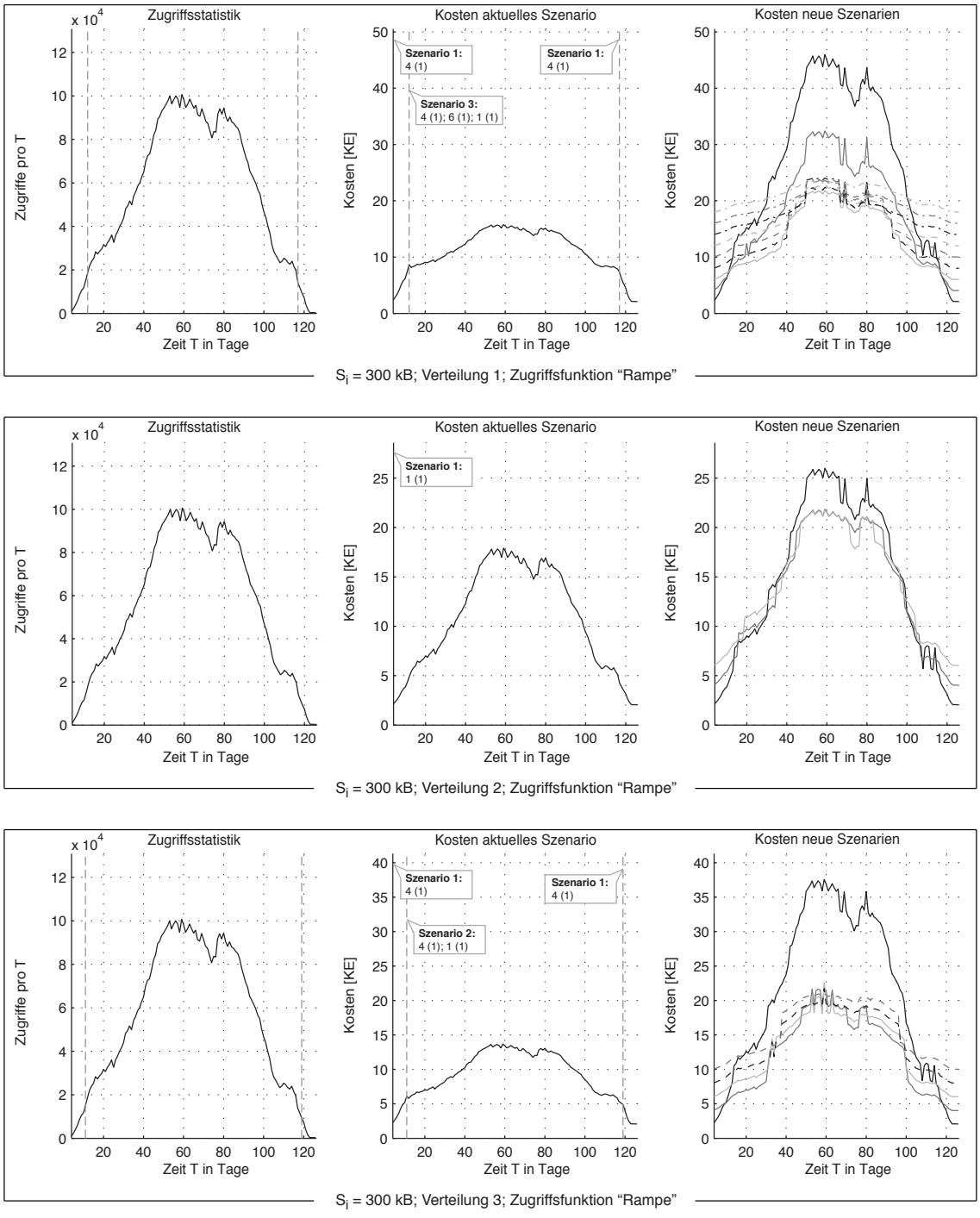


Abbildung 4.19.: Verlauf der Kosten der einzelnen Szenarien für Zugriffsfunktion (a)

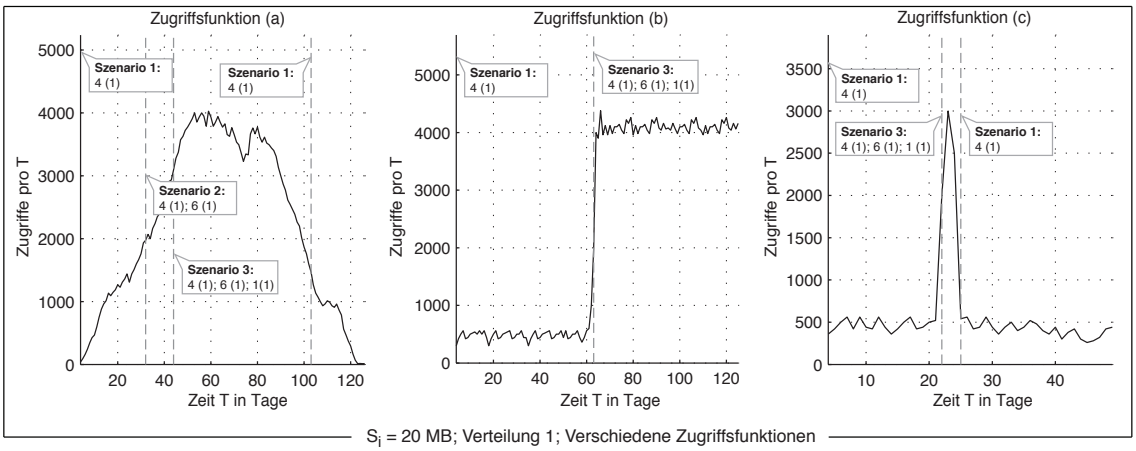


Abbildung 4.20.: Ergebnisse des Algorithmus für alle Zugriffsfunktionen

die Hinzunahme einer weiteren Region stark verringert wird. Bereits bei sechs Regionen ist diese sehr nahe am Optimum $\overline{D} = 1$, falls in jeder Region mindestens ein Server verwendet wird.

Abbildung 4.19 zeigt die Entscheidungen des Algorithmus für die Zugriffsfunktion (a) aus Abbildung 4.16 für alle drei Aufrufverteilungen und einem Inhaltsobjekt mit der Dateigröße von 300 kB. Für die Zugriffsverteilungen 1 und 3 ist zu erkennen, dass ab einer Aufrufrate von etwa 20.000 Aufrufen täglich ein Wechsel der Szenarien statt findet. Aufgrund der globalen Zugriffsverteilung 1 werden dabei nach der Region 4 mit einem Server zusätzlich die Regionen 6 und 1 mit je einem Server hinzugenommen. Für die Zugriffsverteilung 3 wird nur Region 1 mit einem Server hinzugefügt. Nachdem die Aufrufrate gegen Ende wieder abnimmt, entscheidet der Algorithmus, die Replika und die zusätzlichen Regionen zu entfernen und wieder zum Ursprungsszenario zurückzukehren. Im Vergleich dazu findet bei der regionalen Zugriffsverteilung 2, trotz einer maximalen Aufrufrate von fast 100.000 pro Tag, kein Wechsel des Szenarios statt. Auch die Hinzunahme eines weiteren Servers ist noch nicht erforderlich.

Abbildung 4.20 zeigt den Vergleich der drei Zugriffsfunktionen für die Verteilungsfunktion 1 und einer Dateigröße von 20 MB der teuersten Inhaltsklasse Streaming. Dabei ist zu erkennen, dass der Algorithmus bei einem sprunghaften Anstieg sehr schnell auf neue, kostengünstigere Szenarien, umstellt. Die Verzögerung von einer Periode ist jedoch nicht zu vermeiden, da der Algorithmus periodisch aufgerufen wird und der Anstieg erst beim nächsten Durchlauf berücksichtigt werden kann. Ebenso ist die Hysterese für die Zugriffsfunktion (c) gut sichtbar, da der Wechsel zurück auf das ursprüngliche Verteilungsszenario an einer anderen Schwelle der Zugriffszahlen erfolgt.

Bewertung und Fazit Um einen abschließenden Vergleich zwischen traditionellem Hosting mit einem Server in einer Region, Caching, und dem Algorithmus zur Inhaltsmobilität herstellen zu können, werden in Abbildung 4.21 die Kosten dieser drei Verfahren verglichen. Für den Vergleich wurde eine 20 MB Datei der Klasse Streaming sowie die globale Aufrufverteilung 1 verwendet. Die Kosten für das traditionelle Hosting werden dabei von einem Server in Region 4 übernommen, der für das Szenario mit nur einer Re-

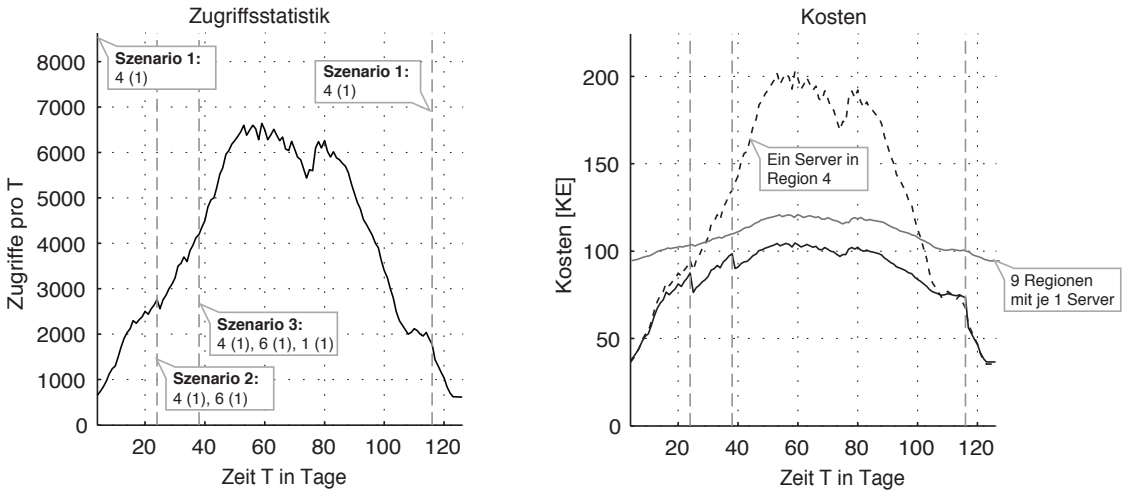


Abbildung 4.21.: Vergleich der Kosten des Algorithmus mit traditionellem Hosting und Caching

gion den kostengünstigsten Wert darstellt (gestrichelten Linie). Die Kosten für Caching (graue Linie) wurden so modelliert, dass in jeder Region ein Server das Inhaltsobjekt bereitstellt (neun Regionen mit je einem Server). Diese Annahme ist zulässig, da im Falle von Caching jedes Inhaltsobjekt, unabhängig von seiner Aufrufrate, bereits beim ersten Aufruf im Cache vorgehalten wird und daher in jeder Region gespeichert wird. Die Kosten unter Verwendung der Inhaltsmobilität werden durch die durchgezogene Linie dargestellt.

Der Verlauf der Kosten ist für geringe Zugriffszahlen zwischen traditionellem Hosting und Inhaltsmobilität zunächst gleich, da ein Server in Region 4 bereits die kostengünstigste Lösung darstellt. Die Kosten für Caching sind bereits drastisch erhöht, da die Fixkosten der Server für diese Aufrufrate deutlich höher sind als die Kosten verursacht durch die höhere mittlere Distanz. Ab etwa 2.500 Zugriffen täglich entscheidet der Algorithmus, dass eine Aufteilung auf die Regionen 4 und 6 mit je einem Server günstiger ist, da sich die mittlere Distanz $\overline{D}$ deutlich verringern lässt und die Gesamtkosten trotz drei verwendeter Server geringer sind. Für weiter steigende Zugriffe wird schließlich noch Region 1 mit einem Server verwendet. Im Vergleich dazu steigt die Last des Servers bei traditionellem Hosting mit nur einem Server zusätzlich zur mittleren Distanz an, was die Kosten weiter erhöht, da auch die Ausfallwahrscheinlichkeit zunimmt. Die Kosten für Caching steigen dagegen nur moderat an, da für höhere Aufrufzahlen die Verteilung auf alle Regionen zunehmend effizienter wird. Dennoch sind die Kosten für Caching immer noch höher im Vergleich zur Inhaltsmobilität. Für dieses konkrete Szenario lassen sich die Kosten in Phasen mit hohen Zugriffszahlen durch Verwendung des Algorithmus zur Inhaltsmobilität im Vergleich zu traditionellem Hosting auf fast die Hälfte reduzieren. Verglichen mit Caching kann Inhaltsmobilität besonders in Phasen mit geringen Zugriffszahlen helfen, die Kosten gering zu halten.

Grundsätzlich kann aus den Simulationsergebnissen festgestellt werden, dass die erreichbare Kostenersparnis durch Inhaltsmobilität stark von der Größe der Datei, sowie der Zugriffszahl abhängig ist. Je größer die Datei, desto höher die Kosten pro Aufruf und

desto höher der Nutzen durch eine Verringerung der mittleren Distanz. Trotz der stark vereinfachten Modellierung des Netzes mit nur neun Regionen werden die Möglichkeiten zur Kostenreduktion durch Inhaltsmobilität deutlich aufgezeigt. Da der Algorithmus modular und flexibel gestaltet ist, können jederzeit neue Berechnungen mit aktuellen Werten bezüglich Kosten für Server und Netz sowie einer realistischeren Anzahl an Regionen erneut durchgeführt werden.

4.7. Konzept eines hybriden Speicherdienstes in HiiMap

In diesem Abschnitt wird, als Ergänzung zu dem Algorithmus zur Inhaltsmobilität und bereits existierender Peer-to-Peer-Technik (P2P) das Konzept eines hybriden Speicherdienstes vorgestellt. Dieser Speicherdienst verfolgt dabei das Konzept einer so genannten Speicher-Cloud, die für den Benutzer absolut transparent eine nahezu unbegrenzte Speicherkapazität zur Verfügung stellt und Inhaltsobjekte kosteneffizient auf das jeweils günstigere Speichersystem verteilt.

Das Konzept basiert darauf, dass jeder Benutzer einen Teil seiner eigenen Speicherkapazität den anderen Benutzern zur Verfügung stellt. Der Speicherdienst ist dabei besonders auf die Bedürfnisse des immer mehr aufkommenden user-generated content zugeschnitten, so dass Benutzer ihre selbst erstellten Inhalte in der Cloud abspeichern und publizieren können. Diese verteilt die Inhalte zunächst mit Hilfe des P2P-Systems auf andere Benutzer. Wird ein Inhaltsobjekt besonders beliebt und damit die Last auf das Gerät im P2P-Netz, das diesen Inhalt speichert sehr groß, kommt der Algorithmus zur Inhaltsmobilität zum Einsatz. Dieser verteilt das Inhaltsobjekt dann abhängig von der Anfagelast kosteneffizient auf dedizierte Server. Der Speicherdienst selbst stellt lediglich zwei Funktionen zur Verfügung: Speichern und Löschen. Funktionen wie Suchen und Abrufen werden nicht explizit vom Speicherdienst bereitgestellt, sondern sind grundlegende Funktionen des inhaltsorientierten Netzes.

Die Verwaltung des Speicherdienstes wird für jede Region zentral vom Mapping-System übernommen. Die Aufgaben für diese zentrale Instanz bestehen im wesentlichen aus der Verwaltung der Benutzerkontingente sowie der Modifikation der Mapping-Einträge für Inhalte, die von einem Speichersystem in das andere migrieren.

Da Inhalte öffentlich zugänglich sind, empfiehlt sich dieser Dienst nur für Daten, die auch für die Öffentlichkeit bestimmt sind. Sollen private Daten aus Redundanzgründen mit dem Speicherdienst abgespeichert werden, müssen diese zuvor verschlüsselt werden.

4.7.1. Verwendete Speicherkonzepte

Aufgrund der Tatsache, dass die Beliebtheit von Inhalten im Internet einer Zipf-Verteilung folgt, stellt der hybride Speicherdienst eine extrem kosteneffiziente Lösung dar. Da es sehr viele Inhalte mit sehr geringen Aufrufzahlen gibt, werden diese im P2P-Netz abgespeichert. Da es sich hierbei um ein kooperatives dezentrales System handelt, ist kein Wartungsaufwand erforderlich. Ebenso ist das System skalierungsfähig, da mit jedem neuen Teilnehmer auch zusätzlicher Speicherplatz bereitgestellt wird. Die Kosten dafür

werden zu gleichen Teilen von den Teilnehmern getragen. Für extrem beliebte Inhalte stellt schließlich ein dediziertes, vom Netz verwaltetes Speichersystem mit ressourceneffizienter Inhaltsverteilung die beste Lösung dar. Ebenso wird damit für beliebte Inhalte die Verfügbarkeit erhöht.

4.7.1.1. Peer-to-Peer (P2P) Netz

Ein P2P-Netz zeichnet sich grundsätzlich dadurch aus, dass alle Teilnehmer gleichberechtigt sind und sowohl als Datenspeicher als auch als Datenkonsument auftreten. Bezüglich der Organisation der Teilnehmer und einer etwaigen Topologie existieren verschiedene Protokolle, die sich in zentralisierte, strukturierte und unstrukturierte P2P-Netze einteilen lassen [Zöl08, Kun09].

Für den Speicherdienst kommt eine Kombination aus einem zentralen und unstrukturierten P2P-Netze zum Einsatz. Da jedem Inhalt eine eindeutige Kennung zugewiesen ist, die über das Mapping-System abgerufen werden kann und die letztendlich den Locator zum entsprechenden DO beinhaltet, stellt das Mapping-System eine zentrale Instanz dar, die Kenntnis über alle Objekte und deren Speicherort besitzt. Die Inhaltsobjekte selbst werden durch die zentrale Instanz unstrukturiert und zufällig auf alle Teilnehmer des P2P-Netzes verteilt. Dies bietet die Möglichkeit, dass Geräte das Speichern von Inhaltsobjekten aufgrund von Überlast ablehnen können. Im Falle eines strukturierten P2P-Netzes wären Geräte nach der jeweiligen Abbildungsvorschrift zur Speicherung verpflichtet. Um die Verfügbarkeit durch Replikation zu erhöhen, wird das DO auf G Geräten abgespeichert. Verlässt ein Gerät das P2P-Netz, wird dies an die zentrale Instanz signalisiert, die ein neues Gerät auswählt. Die entsprechenden DO werden von den verbleibenden Replika neu geladen.

Im Gegensatz zum klassischen P2P-Konzept werden im vorgestellten Speicherkonzept von anderen Geräten geladene Dateien selbst nicht wieder als zusätzliche Quelle zur Verfügung gestellt. Obwohl dies aufgrund der häufigen Replikation zwar zu einer hohen Verfügbarkeit von beliebten Inhalten führt, ist diese Eigenschaft hier unerwünscht, da im Gegenzug weniger Platz für unbeliebtere Objekte zur Verfügung steht. Beliebte Dateien sollen stattdessen von dedizierten Servern bereitgestellt werden.

Der Zusammenschluss von Benutzern zu einem P2P-Netz erfolgt regional, um hohe Verzögerungen durch große räumliche Distanzen zu vermeiden. Leistungsstarke Geräte von Benutzern innerhalb einer Region schließen sich damit zu einem P2P-Netz zusammen. Auf Inhalte, die in P2P-Netzen anderer Regionen gespeichert werden, kann aber jederzeit zugegriffen werden.

Um zu vermeiden, dass bei jedem Locator-Wechsel eines Geräts die IO der Inhalte, die das Gerät bereitstellt, geändert werden müssen, kann im IO anstatt einem Locator-Wert auch der UID eines diesen Inhalt speichernden Gerätes eingetragen werden. Während diese Option grundsätzlich dafür vorgesehen ist, auf weitere IO zu referenzieren, führt das anfragende Gerät im Fall eines referenzierten UID des Typs $T = 1$ oder $T = 2$ dafür eine Mapping-Anfrage durch und verwendet die zurückgelieferten Locator-Werte zum Abruf des Inhaltsobjekts.

4.7.1.2. Dedizierte Speicherserver des inhaltsorientierten Netzes

Sehr beliebte Inhalte werden auf dedizierten Speicherservern bereitgestellt, die von jeder Region zur Verfügung gestellt werden. Diese Server werden, wie auch das Mapping-System, kooperativ von allen Betreibern einer Region verwaltet. Somit ist die Verteilung und Speicherung beliebter Inhalte nicht mehr die Aufgabe eines einzelnen kommerziellen Betreibers, sondern des Netzes selbst. Die dedizierten Speicherserver verwenden den Algorithmus zur Inhaltsmobilität, so dass beliebte Inhalte kosteneffizient verteilt werden können. Die Entscheidung, wann ein Inhaltsobjekt vom P2P-Netz auf einen dedizierten Speicherserver kopiert wird, entscheidet das dafür zuständige Gerät im P2P-Netz abhängig von seiner Auslastung.

4.7.2. Funktionsablauf des hybriden Speicherdienstes

Der hybride Speicherdienst erlaubt das transparente Abspeichern und anschließende effiziente Verteilen von Inhalten für grundsätzlich jeden Benutzer. Inhalte werden zunächst immer im P2P-Netz abgespeichert. Erst bei Erreichen der Lastgrenze eines Gerätes im P2P-Netz erfolgt das Verschieben in das andere System.

Die durch das Bereitstellen eines Inhaltsobjekts i erzeugte Datenrate r_i auf einem Gerät n errechnet sich aus der Anfragerate λ_i dieses Objekts und dessen Dateigröße S_i mit:

$$r_{n,\mathrm{i}} = S_i \cdot \lambda_{n,i} \tag{4.17}$$

Die Summe der einzelnen Datenraten r_i aller I Objekte, die das Gerät n bereitstellt, wird als $R_{n,\mathrm{ges}}$ bezeichnet mit:

$$R_{n,\mathrm{ges}} = \sum_{i=1}^{I} r_{n,i} \tag{4.18}$$

Die Auslastung eines Geräts n im P2P-System wird durch den Parameter Λ_n modelliert. Dieser berechnet sich aus der gesamten verfügbaren Datenrate R des Geräts, sowie der Summe der durch alle Inhaltsobjekte erzeugten Datenrate $R_{n,\mathrm{ges}}$. Für Λ_n gilt:

$$\Lambda_n = \frac{R_{n,\mathrm{ges}}}{R} \tag{4.19}$$

Der Maximalwert für Λ_n beträgt 1. In diesem Fall wird die gesamte verfügbare Datenrate für das Bereitstellen von Inhaltsobjekten verwendet.

4.7.2.1. Vorraussetzungen

Für den Betrieb des hybriden Speicherdienstes, besonders unter Verwendung des P2P-Netzes, sind jedoch einige Voraussetzungen nötig. Diese sind im Folgenden zusammengefasst.

- Für den Dienst ist eine Registrierung nötig, um die Daten für das P2P-Netz zu erhalten und um den Speicherplatz zu kontingentieren. Der Speicherplatz ist personenbezogen. Jeder Benutzer kann mit mehreren Geräten am Speicherdienst teilnehmen.

- Zur Inhaltsspeicherung im P2P-Netz werden nur leistungsfähige Geräte verwendet. Jedes Gerät, das am P2P-Netz teil nimmt, muss eine bestimmte gesamte Mindestdatenrate $R_{\text{ges,min}}$ sowie eine bestimmte Mindestdatenrate pro Objekt r_{min} bereit stellen können. Es kommen daher hauptsächlich stationäre Geräte mit fester Internetanbindung in Frage. Die Lastgrenze $\Lambda_{n,\max}$, ab der ein Gerät Inhaltsobjekte an die regionalen Speicherserver übergibt, kann vom Benutzer des Geräts festgelegt werden, muss jedoch mindestens doppelt so hoch sein wie die durch die Mindestdatenrate verursachte Grundlast.

- Benutzer leistungsstarker Geräte, die am P2P-Netz teilnehmen, sollen einen bestimmten Teil der Speicherkapazität ihrer Geräten für den Speicherdienst zur Verfügung stellen.

- Jeder Benutzer darf Inhalte abspeichern. Benutzer, die durch leistungsstarke Geräte Speicherplatz zur Verfügung stellen, dürfen mehr Speicherplatz durch eigene Inhalte belegen. Für Benutzer, die keinen eigenen Speicherplatz einbringen, ist jedoch eine reduzierte Speichermenge frei. Dies schafft den Anreiz, dass Benutzer selbst Speicher zur Verfügung stellen. Der Speicherplatz, den ein Benutzer belegen darf, ist dabei um ein Vielfaches größer als die Kapazität, die er selbst einbringt.

4.7.2.2. Speichern von Inhalten und Auswahl des Systems

Inhalte können von registrierten Benutzern mittels der Funktion „Speichern“ im Speicherdienst abgelegt werden. Dabei wird der Inhalt zunächst immer im P2P-Netz abgespeichert. Nach der Registrierung des Inhalts-UID werden durch die zentrale Instanz die Geräte bestimmt, die den Inhalt abspeichern. Anschließend wird der Inhalt auf diese Geräte transferiert und der Mapping-Eintrag aktualisiert. Das Gerät des Benutzers, der das Inhaltsobjekt publiziert, steht dabei nicht als Quelle zur Verfügung, außer es befindet sich unter den G zufällig ausgewählten Geräten. Das Publizieren von fremden Inhalten, die bereits eine UID besitzen, ist nicht möglich, da nur der Inhaltsersteller die Speicherfunktion ausführen und Mapping-Einträge modifizieren darf.

Verschieben auf dedizierte Speicherserver Steigt an einem der Geräte im P2P-Netz die durch ein Inhaltsobjekt i verursachte Datenrate r_{i} so stark an, dass diese für mehr als 50% der Maximalauslastung $\Lambda_{n,\max}$ dieses Geräts verantwortlich ist, wird dieses Objekt an die dedizierten regionalen Speicherserver übergeben. Wird die Maximalauslastung durch mehrere beliebte Objekte erreicht, ohne dass ein bestimmtes Objekt für mehr als 50% der Maximalauslastung verantwortlich ist, werden die aufrufstärksten Objekte, verschoben, bis die aktuelle Auslastung wieder auf unter 50% der Maximalauslastung fällt. Damit wird eine Hysterese erzeugt, die abwechselnde kopieren zwischen beiden Systemen verhindern soll.

Das Gerät, an dem die Überlastsituation auftritt, initiiert bei der zentralen Instanz den Transfer auf die Speicherserver der Region. Diese ruft das entsprechende Inhaltsobjekt einmalig auf und verteilt es anschließend nach dem Algorithmus der Inhaltsmobilität kosteneffizient im Netz, auch regionsübergreifend. Anschließend wird der Mapping-Eintrag des Objekts entsprechend angepasst und die Löschung aller Replika im P2P-Netz veranlasst.

Verschieben zurück ins P2P-Netz Wenn die Beliebtheit eines Objekts auf den Speicherservern einer Region unterhalb die Mindestdatenrate $r_{\min}$ sinkt, wird dieses Objekt zurück ins P2P-Netz kopiert. Die zentrale Instanz bestimmt dazu zufällig G Geräte, welche die weitere Speicherung des Objekts übernehmen. Kann ein Gerät Aufgrund seiner Auslastung Λ_n das Objekt nicht mehr annehmen, wird nach dem Zufallsprinzip ein anderes Gerät gewählt. Obwohl jedes Gerät als Grundanforderung die Rate $r_{\min}$ verarbeiten können muss, so kann das Gerät durch das Bereitstellen mehrerer Objekte dennoch überlasten, falls die Summe der Datenraten $R_{n,\text{ges}}$ zu hoch wird.

Entfernen von Inhalten Eine weitere durch den hybriden Speicherdienst bereitgestellte Funktion ist „Löschen“. Da der Speicherdienst für Inhalte bestimmt ist, die von Benutzern erzeugt sind, besitzt jeder Ersteller eines Inhaltsobjekts auch die Löschrechte. Die Löschfunktion wird immer an die zentrale Instanz gemeldet, die zunächst die Berechtigung überprüft. Anschließend werden alle im Mapping-Eintrag aufgelisteten Replika entfernt, unabhängig davon, in welchem System der Inhalt aktuell gespeichert wird. Geräte im P2P-Netz erhalten dazu den Löschbefehl, dem sie nachkommen müssen. Ebenso entfernen alle dedizierten Speicherserver das Objekt. Im letzten Schritt wird auch der Mapping-Eintrag entfernt und der UID freigegeben.

4.7.3. Kostenbetrachtung

Wie im Abschnitt 4.6.5 über Inhaltsmobilität gezeigt, sind bei Inhalten, die nur selten aufgerufen wurden, die Betriebskosten des Speicherservers dominant gegenüber den Kosten, die durch Datentransfer anfallen. Durch das Verwenden des hybriden Speicherdienstes, der selten angefragte Inhalte in ein P2P-Netz auslagert, lassen sich die Kosten weiter reduzieren, da für den Betrieb des P2P-Netzes aus der Sicht des Netzes keine Kosten anfallen. Das P2P-Netz stellt einen kooperativen Speicher dar, zu dem jeder Benutzer einen Teil beiträgt und die Kosten von allen Benutzern geteilt werden. Das System ist darüber hinaus skalierbar, da neue Benutzer automatisch neuen Speicherplatz einbringen.

Beliebte Inhalte, die im P2P-Netz Geräte überlasten und zu erhöhten Datenübertragungskosten führen würden, werden von dedizierten Speicherservern mittels Inhaltsmobilität ressourcen- und kosteneffizient verteilt. Die Kosten für den Betrieb der dedizierten Speicherserver werden von den Netzbetreibern übernommen, die diese Kosten auf die Benutzer anteilig umlegen können.

4.8. Inhalte- und Personen-Modul der HiiMap-Middleware

Die Unterstützung zur Adressierung von Inhalten und Personen muss in jedem Gerät, das auf diese Funktionalität zurückgreifen will, vorhanden sein. Durch die modulare Gestaltung der HiiMap-Middleware auf der Basis einzelner funktionaler Module, kann diese Fähigkeit durch das Hinzufügen eines weiteren Moduls erreicht werden. In Abbildung 4.22 ist die vollständige HiiMap-Middleware mit Inhalte- und Personen-Modul dargestellt. Die Aufgaben der neuen Module, sowie die Interaktion mit bereits vorhandenen, werden im Folgenden vorgestellt.

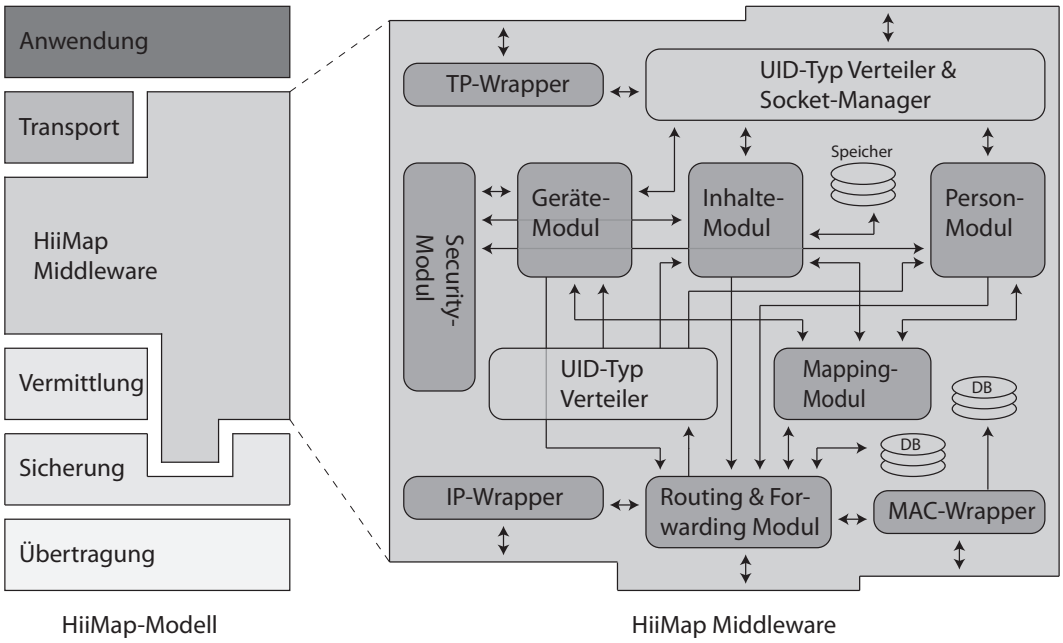


Abbildung 4.22.: HiiMap-Middleware mit Inhalts- und Personen-Modul

Inhalte-Modul Dieses Modul ist für den Empfang von Inhalten, die Auswertung der auf Anfragen zurückgelieferten IOe, sowie die Bereitstellung eigener Inhalte verantwortlich.

Die Anfragen von Anwendungen für einen Inhalts-UID werden vom UID-Typ Verteiler an dieses Modul weitergereicht. Es übergibt diese Anfrage an das Mapping-System und wertet das erhaltene IO aus. In diesem Zusammenhang kann seitens des Moduls bereits eine Auswahl der möglichen Repräsentationen erfolgen, beispielsweise minimale oder maximale Bitrate für Audio- oder Videodateien. Ist lediglich eine mögliche Option vorhanden, wird sofort das entsprechende DO geladen. Andernfalls werden die möglichen Repräsentationen und Versionen an die Anwendung übergeben, die diese geeignet Darstellt.

Das Modul ist des Weiteren für die Bereitstellung von Inhalten durch das Gerät verantwortlich. Dies schließt auch Inhalte mit ein, die über den hybriden Speicherdienst

transparent auf diesem Gerät gespeichert werden. Inhalte, die durch das Gerät selbst bereitgestellt werden, müssen mit ihrem UID am Modul registriert werden und das Modul muss Zugriff auf den Speicherort besitzen. In Kooperation mit dem Mapping-Modul werden außerdem die Locator-Werte im IO aktualisiert.

Für Teilnehmer am P2P-Netz des hybriden Speicherdienstes nimmt das Modul die DO von Inhalten entgegen und speichert diese in dem vom Benutzer ausgewiesenen Speicher. Das Modul ist für die Einhaltung der Lastgrenzen verantwortlich und benachrichtigt im Überlastfall die zuständige Instanz.

Anfragen für Inhalte werden vom UID-Typ Verteiler an dieses Modul geleitet, welches das gewünschte DO an den Anfragenden übermittelt.

Personen-Modul Das Personen-Modul ist für jegliche Art von Kommunikation zwischen Benutzern verantwortlich. Jede Anwendung, die einen Kommunikationsdienst wie Chat, Email, Instant Messaging oder VoIP bereit stellt, registriert diesen am Personen-Modul. Ausgehend von den registrierten Diensten und Einstellungen des Benutzers modifiziert dieses Modul in Kooperation mit dem Mapping-Modul das unter dem UID des Benutzers gespeicherte Informationsprofil. Das Setzen des persönlichen UID erfolgt durch den Benutzer, beispielsweise mit Chipkarte, die auch die notwendigen Schlüssel speichert. Alternativ ist dies auch über eine manuelle Systemkonfiguration mit auf einem Speichermedium hinterlegten Schlüsseln möglich.

4.9. Zusammenfassung

In diesem Kapitel wurde HiiMap zu einem inhaltsorientierten Netz erweitert und dem Trend Rechnung getragen, dass die Informationsbeschaffung im Internet vor der eigentlichen Kommunikation zwischen Geräten in den Vordergrund rückt. Zu diesem Zweck wurde das Namensschema für UID aus Kapitel 3 erweitert, um auch Inhalten jeder Art einen eigenen UID zuordnen zu können. Diese Inhaltsobjekte werden mit einem Informationsmodell beschrieben und im Mapping-System abgespeichert. Das Informationsmodell erlaubt die Unterscheidung verschiedener Repräsentationen und Versionen eines bestimmten Inhaltsobjekts. Die nötigen Schritte zur Registrierung und Zuteilung von einer UID für Inhalte sind ausführlich behandelt.

In diesem Zuge wurde auch die Möglichkeit berücksichtigt, Personen einen eindeutigen UID zuzuordnen. Das bereits vorhandene Namensschema wurde dazu erneut erweitert. Jede Person wird durch ein Informationsprofil beschrieben, das verschiedene Möglichkeiten zur Kontaktaufnahme mit dieser Person enthält. Durch den einer Person zugewiesenen UID kann das Informationsprofil abgerufen werden, das ebenfalls im Mapping-System gespeichert ist.

Da alle Namensschemata auf der Generierung aus einem Klartextnamen zusammen mit einer Hash-Funktion basieren, wurde ein Nachschlagemechanismus auf der Basis von N-Grammen vorgestellt, der in der Lage ist, kleinere Fehler im Klartextnamen automatisch zu korrigieren.

Nach der Diskussion von Beliebtheit und Dateigrößenverteilung der Inhalte im heutigen Internet wird ein Algorithmus zur ressourceneffizienten Inhaltsmobilität vorgestellt.

Unter Inhaltsmobilität versteht man die Eigenschaft, dass sich Inhalte frei im Netz bewegen können und immer nahe beim Benutzer gespeichert sind. Je nach Popularität werden Inhalte dabei auf mehreren Servern an verschiedenen Orten gespeichert. Der vorgestellte Algorithmus zur Inhaltsmobilität basiert auf einem Greedy-Algorithmus und stellt ressourceneffiziente Verteilungsszenarien auf, die im Vergleich zu einem traditionellen Hosting die Kosten aus der Sicht des Netzes einsparen. Der Algorithmus greift dazu auf Statistikwerte über Aufrufhäufigkeit und Aufrufort zurück. Mittels Simulation wurde die Funktionsweise des Algorithmus überprüft und mit traditionellem Hosting sowie Caching verglichen. Der Algorithmus stellt dabei zu jeder Aufrufverteilung stets die kostengünstigste Lösung dar.

Die Vorstellung eines hybriden Speicherdienstes, basierend auf einem P2P-Netz sowie dem Algorithmus zur Inhaltsmobilität, vervollständigen das Konzept des inhaltsorientierten Netzes. Durch das P2P-Netz steht dem Dienst ein praktisch unbegrenzter Speicherplatz zur Verfügung, der unter allen Benutzern aufgeteilt wird. Unpopuläre Objekte werden vom Speicherdienst im P2P-Netz gespeichert und verursachen damit keine Kosten für die Wartung und Instandhaltung von Speicherservern. Beliebte Inhalte werden auf dedizierten Speicherservern kopiert, welche Inhaltsobjekte unter Verwendung des Algorithmus zur Inhaltsmobilität ressorceneffizient verteilen und bereit stellen.

Im letzten Abschnitt werden die zusätzlichen Module der HiiMap-Middleware vorgestellt, welche die notwendigen Funktionen zum Umsetzen des Konzepts des inhaltsorientierten Netzes nötig sind.

5. Migration zu HiiMap

HiiMap ist ein revolutionäres Konzept für eine zukünftige Internetarchitektur. Die Einführung des Konzepts erfordert daher einen höheren Aufwand auf Seiten der Betreiber und Nutzer, da es sich nicht ausschließlich um eine evolutionäre Weiterentwicklung einzelner Bereiche der aktuellen Internet-Architektur handelt. HiiMap geht mit Veränderungen an der Vermittlungsschicht einher, die aktuell fast ausschließlich auf dem IP-Protokoll basiert. Eine Kompatibilität mit der bestehenden Architektur ist nur in begrenztem Rahmen gegeben, so dass eigene Verfahren zur Migration der aktuellen Internet-Architektur zu HiiMap erforderlich sind. Aufgrund der enorm hohen Anzahl an Geräten im Internet kann nicht an einem bestimmten Stichtag komplett auf HiiMap umgestellt werden. Eine solche Umstellung wurde 1982 angewandt, als das DARPANET auf das damals neue TCP/IP-Protokoll umgestellt wurde. Dennoch musste dieser Schritt langfristig geplant werden, obwohl damals lediglich einige hundert Rechner an diesem Netz angeschlossen waren. Laut [Han06] war dies jedoch das erste und das letze Mal, dass eine derartige Umstellung an einem Tag durchgeführt werden konnte. Für die Einführung von HiiMap liegt die Lösung in einer Migrationsphase, in der sowohl die aktuelle Internetarchitektur als auch HiiMap koexistieren und parallel betrieben werden. Die Problematik kann ansatzweise mit der Einführung von IPv6 verglichen werden, das im Vergleich zum weltweit etablierten IPv4 ebenfalls eine neues Vermittlungsprotokoll darstellt.

Der erste Abschnitt im Folgenden Kapitel beschreibt zunächst die Gemeinsamkeiten von HiiMap mit der aktuellen Internetarchitektur und stellt die wesentlichen Unterschiede heraus, die für eine Migration berücksichtigt werden müssen. Anschließend wird die Integration des HiiMap-Protokolls auf Endgeräten sowie der Aufbau des Mapping-Systems besprochen, das als integraler Bestandteil einer Internet-Architektur, basierend auf der Trennung von Locator und Identifier notwendig ist. Im dritten Abschnitt werden verschiedene Arten zur Koexistenz von IP und HiiMap dargestellt. Dabei werden alle möglichen Fälle betrachtet, wie aus technischer Sicht eine Interoperabilität zwischen beiden Architekturen erreicht werden kann. Diese ist nötig, damit die neue Architektur sowohl bei den Betreibern als auch bei den Benutzern Akzeptanz finden kann. Neue Komponenten, die für die Umsetzung der Protokolle notwendig sind, werden an gegebener Stelle erläutert.

5.1. Gemeinsamkeiten und Unterschiede

Um verschiedene Migrationskonzepte hin zur HiiMap-Architektur entwickeln zu können ist es zunächst notwendig, die Gemeinsamkeiten und Unterschiede mit der aktuellen Internet-Architektur zu identifizieren.

Die HiiMap-Architektur basiert auf der Topologie des aktuellen Internets. Das be-

deutet, dass HiiMap ebenfalls auf AS aufbaut, die von einzelnen Betreibern verwaltet werden und untereinander verbunden sind. Es sind daher keine Änderungen der Topologie sowie der verwendeten physikalischen Infrastruktur nötig, die der ersten Schicht im OSI-Schichtenmodell entspricht.

Aufgrund der Tatsache, dass HiiMap jedoch ein neuartiges Adressierungskonzept basierend auf der Trennung von Locator und Identifier verfolgt, ist eine Änderung des global verwendeten Vermittlungsprotokolls nötig. Im Sinne einer vollständigen Migration hin zu HiiMap betrifft dies jedes an das Internet angeschlossene Gerät. Das mit HiiMap verbundene, zwingend notwendige Mapping-System stellt außerdem eine Instanz dar, die in vergleichbarer Form bis dato noch nicht existiert. DNS kann dabei nicht mit dem Mapping-System verglichen werden, da DNS nur einen Zusatzdienst darstellt, um Gerätenamen in die IP-Adresse aufzulösen, die eigentliche Kommunikation auf IP-Basis davon jedoch unberührt ist. HiiMap benötigt dagegen für die Umsetzung des UID in die Locator-Werte der Geräte zwingend das Mapping-System, das demzufolge als integraler, vollkommen neuer Bestandteil angesehen werden muss.

Innerhalb eines AS kann HiiMap zwar ein beliebiges Adressierungsschema verwenden, also auch IPv4 oder IPv6. Dies betrifft jedoch ausschließlich das Schema und nicht das damit verbundene gesamte Protokoll. So kann HiiMap als LTA zwar IPv6-Adressen mit dem damit verbundenen Routingschema verwenden, jedoch wird nicht der IPv6-Header für die Übertragung verwendet, sondern ausschließlich der HiiMap-Header mit der eingebetteten IPv6-Quell- und Zieladresse. Gleiches gilt für den GGUID, der aufgrund seiner Breite von 32 Bit prinzipiell das Adressierungsschema IPv4 verwenden könnte, jedoch ohne den IPv4-Header.

Darüber hinaus sind aufgrund der Gestaltung von HiiMap als Middleware, Modifikationen an den Applikationen notwendig, damit das volle Potential von HiiMap, besonders im Bezug auf die Möglichkeit des inhaltsorientierten Netzes, genutzt werden kann. Aber auch ohne die Unterstützung für Inhalte kann die Middleware alle bereits existierenden Schnittstellen zu den Applikationen bereitstellen oder als reines Vermittlungsprotokoll für bereits bestehende Transportprotokolle wie TCP oder UDP dienen. Die Migration kann somit als abgeschlossen angesehen werden, sobald HiiMap flächendeckend verbreitet ist. Anschließend können die neuen Funktionen, die über die ausschließliche Kommunikation zwischen Geräten hinausgehen, umgesetzt werden. Ebenso kann HiiMap auf bestehenden Sicherungsschichten aufsetzen und bereits bestehende Adressen, wie z.B. die MAC-Adresse, für die Nachbar-zu-Nachbar-Kommunikation verwenden, bevor HiiMap sein eigenes Adressierungsschema basierend auf dem UID in dieser Schicht verwendet. Auf diese Weise wird eine zusätzliche Datentabelle, welche den UID auf die Adresse der Vermittlungsschicht abbildet, überflüssig. In aktuellen IP-Netzen übernimmt die ARP-Tabelle diese Aufgabe.

5.2. Integration von HiiMap in das heutige Internet

Die Integration der HiiMap-Middleware in die aktuelle Internet-Architektur, von der aus letztendlich die vollständige Migration erfolgen kann, lässt sich in zwei Schwerpunkte einteilen. Zum einen muss die Middleware als Softwareprodukt für alle existierenden Be-

triebssysteme entwickelt werden. Da die Middleware modular aufgebaut ist und aus verschiedenen Funktionsblöcken besteht, können die verschiedenen Module auch schrittweise eingeführt werden. Als obligatorisch ist die Unterstützung für Geräte in der Middleware anzusehen. Dieser Funktionsblock ist für die Interoperabilität mit der aktuellen Internet-Architektur zwingend notwendig. Die Unterstützung für Inhalte durch das Inhalte-Modul kann dagegen nachträglich als Aktualisierung hinzugefügt werden.

Zum anderen muss das Mapping-System aufgebaut werden, das eine Voraussetzung für die Kommunikation über HiiMap ist. Auch dabei gilt, dass zunächst die Unterstützung für Geräte gegeben sein muss. Mit schrittweisen Aktualisierungen kann die Datenbank auch für Inhalte und Personen erweitert werden.

5.2.1. HiiMap und IP im Schichtenmodell

Eine schrittweise Umstellung auf HiiMap hat zur Folge, dass einige Geräte in der Lage sein müssen sowohl die alten Protokolle, als auch das neue HiiMap-Protokoll zu unterstützen. Da es sich hierbei um zwei verschiedene Protokolle auf der Vermittlungsschicht handelt, ist es möglich, diese parallel und unabhängig voneinander zu betreiben. Dieses Verfahren wird als *Dual-Stack* bezeichnet, da ein entsprechendes Gerät zwei unabhängige Protokoll-Stacks besitzt. Ein Beispiel für erfolgreiche Interoperabilität und Migration mittels Dual-Stack ist die Koexistenz von IPv4 und IPv6. Ein Gerät, das bereits IPv4 und IPv6 beherrscht und zusätzlich noch mit HiiMap erweitert wird, entspricht prinzipiell einem Gerät mit drei Protokoll-Stacks. Stellvertretend für beide Varianten von IP ist im Folgenden nur von einem Stack die Rede. Die Entscheidung, welcher Protokoll-Stack für die Kommunikation verwendet wird, trifft die kommunizierende Anwendung. Diese muss die Schnittstelle zum entsprechenden Protokoll unterstützen.

5.2.2. Neues Adressierungsschema für die Sicherungsschicht

HiiMap lässt die Aufgaben der Sicherungsschicht unangetastet, bringt jedoch ein eigenes Adressierungsschema für die Kommunikation mit benachbarten Geräten mit, das auf dem UID basiert. Ein Vorteil ergibt sich durch die Einsparung einer weiteren Zuordnungstabelle zwischen Adressen der Vermittlungs- und Sicherungsschicht. Um Dual-Stack mit IP und HiiMap zu realisieren ist es notwendig, dass beide Protokoll-Stacks die gleiche Sicherungsschicht verwenden. Im Parallelbetrieb mit IP verwendet HiiMap daher zunächst noch nicht das eigene Adressierungsschema basierend auf dem UID, sondern MAC-Adressen, falls beispielsweise Ethernet zum Einsatz kommt. Dieses Verfahren ist aus einem weiteren Grund nötig, da aktuelle Ethernet-Switching-Komponenten die Informationen aus der Schicht 2 üblicherweise auf Hardwarebasis auswerten, die fest auf die 48 Bit breite MAC-Adressen eingestellt ist und oft nicht mittels Softwareupdate auf das neue HiiMap-Vermittlungsschema umgestellt werden kann. HiiMap kann daher die Aufgaben der Sicherungsschicht erst dann vollständig übernehmen, wenn die dafür benötigte Hardware ebenfalls darauf umgestellt wurde. Die Umstellung zur vollen Integration der Sicherungsschicht läuft dabei in drei Schritten ab, die im Folgenden erläutert werden.

1. Setzt HiiMap aufgrund von Dual-Stack mit IP vollständig auf Ethernet und MAC-

Adressen auf, kommt auf der Sicherungsschicht eine Modifikation des NDP (Neighbor Discovery Protocol) von IPv6 zum Einsatz, das die Zuordnung von UID zur MAC-Adresse ermöglicht. Dieses Protokoll wird HUD (HiiMap UID Discovery)-Protokoll genannt. Die Zuordnung wird einer der HUD-Tabelle gespeichert.

2. Wird in einem AS oder einer Domäne ausschließlich HiiMap ohne Dual-Stack mit IP verwendet, kann auf der Sicherungsschicht der FW-UID verwendet werden. Falls Hardware auf der Sicherungsschicht nur 48 Bit breite Adressen verarbeiten kann, verwendet HiiMap einen Kompatibilitätsmodus. Dabei wird der FW-UID auf 48 Bit begrenzt und setzt sich vollständig aus dem Ext 1-Wert des Geräte-UID sowie aus den niederwertigsten 16 Bits des Hashwerts des Klartextes zusammen. Da es sich um die Nachbarkommunikation handelt, sind gleiche Werte für Ext 1 dabei ohnehin ausgeschlossen.

3. Wenn im letzten Schritt die Hersteller von Switching-Komponenten ihre Geräte auf die Verarbeitung von 128 Bit breiten Schicht 2-Adressen erweitert haben, kann HiiMap auf dieser Schicht vollständig sein eigenes Adressierungsschema basierend auf 128 Bit breiten UIDs übernehmen.

5.2.3. Aufbau des Mapping-Systems

Das Mapping-System als integraler Bestandteil der HiiMap-Architektur ist die erste Komponente, die bei der Migration hin zu HiiMap bereit gestellt sein muss. Dies ist in der Tatsache begründet, dass dieses System bei jeglicher Kommunikation konsultiert wird, um den jeweils aktuell gültigen Locator des Ziels zurück zu liefern. HiiMap ist als hierarchisches Mapping-System mit einer zentralen GR und mehreren dezentralen Regionen geplant. Der Aufbau des Mapping-Systems folgt dabei dieser regionalen Hierarchie.

5.2.3.1. Globale Registrierungsstelle (GR)

Da die GR die Wurzel der Mapping-Hierarchie und den Ankerpunkt für eine Zertifikatsinfrastruktur darstellt, muss diese als erstes angelegt werden. Die GR untersteht einer gemeinnützigen Organisation ähnlich der GUID-Arbeitsgruppe der vereinten Nationen und kann zunächst aus nur einer Instanz weltweit bestehen, bevor diese aus Gründen der Lastenverteilung und Ausfallsicherheit mehrfach global repliziert wird. Die Konnektivität der GR erfolgt dabei anfangs ausschließlich über IP, da noch kein AS existiert, das auf HiiMap basiert.

Neben der technischen Aufgabe der GR, den UID in das dazugehörige RP aufzulösen, fallen noch Verwaltungsaufgaben an. Diese umfassen die Einteilung des Internets in Regionen und der damit verbundenen Vergabe von RP, die Zuteilung von Betreiberkennungen für den GGUID, die Signatur von UIDs sowie das Sicherstellen der Einzigartigkeit von diesen. Die letzten beiden Aufgaben können dabei an regionale Registrierungsstellen ausgelagert werden, die der GR unterstehen und ebenfalls auf die neuen Prozessabläufe umgestellt werden müssen.

5.2.3.2. Regionen

Nachdem die GR technisch und administrativ funktionsfähig ist, können die ersten regionalen Mapping-Systeme gebildet werden. Der erste Betreiber in einer Region, der in seinem AS HiiMap zusätzlich einführt bzw. komplett auf HiiMap umstellt, muss die Infrastruktur für das regionale Mapping-System anbieten. Diese Instanz des Mapping-Systems stellt damit zunächst die gesamte Region dar gemäß der Idee, dass jeder Betreiber einen Teil des regionalen Mapping-Systems übernimmt. Da eine Region üblicherweise als DHT realisiert wird, entspricht dieser Fall einer DHT mit nur einem Teilnehmer. Stellen weitere Betreiber im Bereich der durch die GR definierten Region auf HiiMap um, so müssen diese ebenfalls einen infrastrukturellen Beitrag zum Mapping-System dieser Region leisten. Dies geschieht durch das Hinzufügen weiterer Knoten in der bereits vorhandenen DHT. Es besteht darüber hinaus die Möglichkeit, dass sich Betreiber zu einer Region zusammenschließen, die von der GR ursprünglich nicht vorhergesehen ist. Dies ist grundsätzlich möglich, jedoch muss dafür ein eigenes RP bei der GR beantragt werden.

Aufgrund der Tatsache, dass eine HiiMap-Kommunikation zwischen zwei Geräten auch über IP mittels Tunneling erfolgen kann, ohne dass ein Betreiber auf HiiMap umgestellt hat, der die initiale Mapping-Region bereit stellt, kann auch die regionale Registrierungsstelle den ersten Knoten der Mapping-Region bereitstellen.

Die technische Umsetzung der Mapping-Region erfolgt durch Geräte, auf denen in einer Datenbank die Mappingeinträge zu den in dieser Region registrierten UID gespeichert sind. Ab dem ersten Schritt obligatorisch ist die Fähigkeit des Mapping-Systems, Geräte-UID zu speichern und aufzulösen. Die Erweiterung der Datenbank für die UID-Typen von Inhalten und Personen kann erst in einem späteren Schritt erfolgen und ist abhängig von der Migrationsbereitschaft der teilnehmenden Betreiber. Wie bei der GR muss die Erreichbarkeit über IP sichergestellt sein, damit frühzeitige Anwender von HiiMap auch weiterhin von anderen Anwendern erreicht werden können, die noch nicht auf HiiMap umgestellt haben. Diese können zwar das Mapping-System nicht direkt kontaktieren, mit Mechanismen wie Tunneling oder Protokoll-Translation, die im nächsten Abschnitt vorgestellt werden, kann dies jedoch indirekt erreicht werden. Dabei müssen bestimmte Helferinstanzen, wie Tunnelrouter, das Mapping-System auch außerhalb einer HiiMap-Domäne anfragen können.

5.3. Möglichkeiten der Koexistenz

Im folgenden Abschnitt werden verschiedene Möglichkeiten zur Koexistenz von IP und HiiMap vorgestellt. Während diese beiden Architekturen grundsätzlich völlig unabhängig von einander existieren können, muss jedoch in der Migrationsphase die Interoperabilität zwischen beiden sichergestellt sein. Diesbezüglich kommen mehrere Arten der Koexistenz infrage (vgl. Abbildung 5.1), die im Folgenden aufgelistet sind und anschließend näher betrachtet werden.

- **HiiMap-Kommunikation über IP-Netze**: In diesem Fall sind drei Möglichkeiten der Vernetzung zu unterschieden. Die Gemeinsamkeit dabei besteht darin, dass alle Verfahren einen Tunneling-Mechanismus verwenden, um HiiMap-Daten

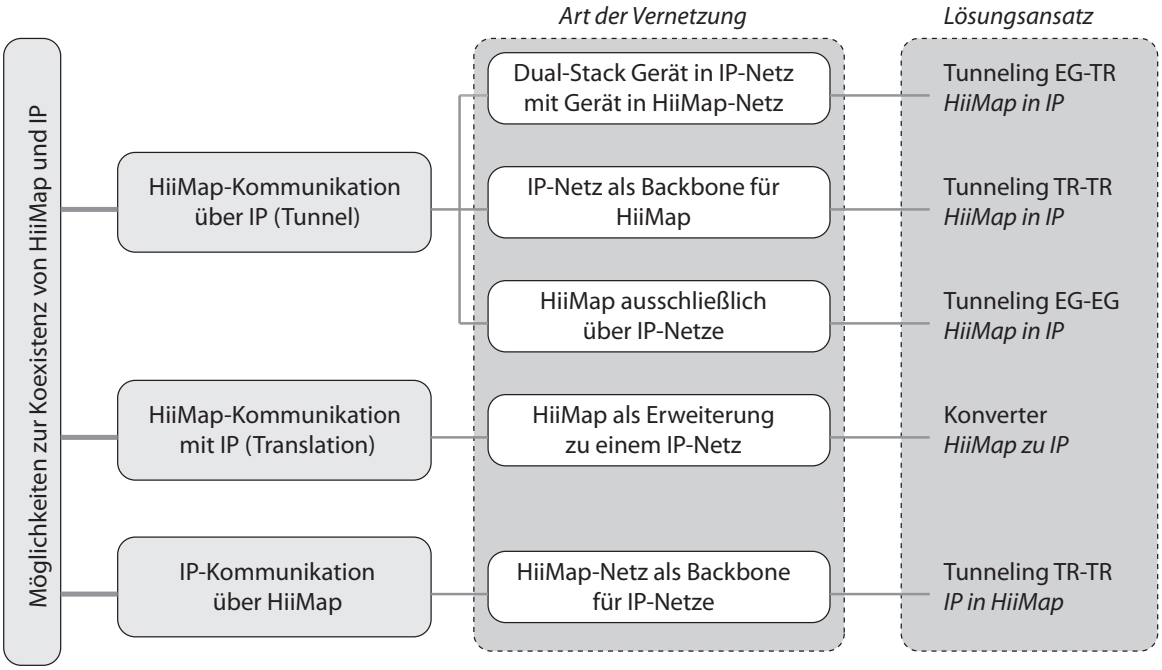


Abbildung 5.1.: Möglichkeiten zur Koexistenz von IP und HiiMap

über IP-Netze zu tunneln. Im ersten Fall kommuniziert ein Dual-Stack Gerät in einem IP-Netz über einen Tunnelrouter mit einem HiiMap-Gerät in einem reinen HiiMap-Netz. Der zweite Fall betrachtet ein IP-Netz als Transportnetz zwischen zwei HiiMap-Netzen. Die Geräte unterstützen dabei ausschließlich HiiMap. Der dritte Fall beschreibt die Vernetzung zweier Dual-Stack Geräte über ausschließlich IP.

- **HiiMap-Kommunikation mit IP-Netzen**: Dieser Fall betrachtet die Vernetzung eine Gerätes, das ausschließlich IP unterstützt, mit einem Gerät, das ausschließlich HiiMap unterstützt. Dabei kommt ein Protokoll-Konverter zum Einsatz.

- **IP-Kommunikation über HiiMap-Netze**: In diesem Fall wird ein HiiMap-Netz dazu verwendet, zwei IP-Netze zu verbinden. Dieser Fall tritt auf, wenn ein Transit-Betreiber auf HiiMap umgestellt hat, seine Kunden jedoch bis jetzt nur IP unterstützen.

In den folgenden Unterpunkten werden diese Fälle detailliert vorgestellt und die einzelnen Schritte der Kommunikation erläutert. Die Existenz eines Mapping-Systems mit IP- und HiiMap-Konnektivität wird vorausgesetzt. Für Zwecke der Interoperabilität beider Systeme wird auch auf das DNS zurückgegriffen. Dessen Notwendigkeit wird an geeigneter Stelle aufgezeigt.

Für Migrationszwecke werden außerdem zwei neue UID-Typen eingeführt, die für bestimmte Vernetzungsarten benötigt werden, bei denen eine IP-Adresse als UID verwendet

wird. Ausgehend von der entsprechen IP-Version werden folgende Typen zusätzlich definiert:

- Typ $T = 32$: Dieser Typ entspricht einer global routingfähigen IPv6-Adresse. Nach der Spezifikation der Adressstruktur von IPv6 in [DH98] besitzen derartige Adressen im höchstwertigen 8 Bit Adressblock hexadezimal immer den Wert 0x20 (dezimal Wert 32).
- Typ $T = 0$: Dieser Typ entspricht einer in IPv6 eingebetteten IPv4-Adresse. Dabei werden die höchstwertigen 96 Bits zu 0 gesetzt, anschließend folgen die 32 Bit der IPv4 Adresse. Da dadurch die höchstwertigen 8 Bit ebenfalls 0 sind, entspricht dies dem UID-Typ 0. Dieses Verfahren mit eingebetteten IPv4-Adressen ist notwendig, um 32 Bit breite IPv4 Adressen auf 128 Bit abzubilden, die der Breite eines UID entsprechen.

5.3.1. HiiMap-Kommunikation über IP mittels Tunnel (HiiMap in IP)

Im folgenden werden Möglichkeiten vorgestellt, wie ein Tunneling-Mechanismus dazu verwendet werden kann, HiiMap-Pakete über ein IP-Netz zu transportieren. Dabei kann das Tunneling entweder direkt zwischen Dual-Stack Endgeräten erfolgen, oder aber, falls sich ein Gerät bereits in einem Netz befindet das vollkommen auf HiiMap umgestellt hat, zwischen Endgerät und einem so genannten Tunnelrouter. Genauso ist auch ein Tunnel zwischen zwei Tunnelroutern möglich, wenn zwei HiiMap-Netze über ein IP-Netz miteinander verbunden sind.

Obwohl Tunneling eine unkomplizierte und schnelle Methode ist neue Vermittlungsprotokolle einzuführen, so ist die dadurch erzeugte Blindlast durch doppelte Packet-Header relativ hoch. Tunneling sollte daher nur als Übergangslösung hin zu einem neuen Verfahren gesehen werden.

5.3.1.1. Tunnel zwischen Endgerät und Tunnelrouter

Bei dieser Art der Vernetzung kommuniziert ein Gerät A mit Dual-Stack HiiMap/IP in einem IP-Netz mit einem Gerät B, das bereits ausschließlich HiiMap unterstützt, wobei zwischen beiden Geräten eine reine HiiMap-Kommunikation stattfindet. Dabei besitzt das Dual-Stack Gerät IP-Konnektivität, das HiiMap-Gerät befindet sich in einem bereits vollständig auf HiiMap umgestellten Netz. Um die Konnektivität zwischen diesen beiden Netzen zu ermöglichen, muss das bereits auf HiiMap umgestellte Netz einen Tunnelrouter (TR) bereit stellen, der auch Konnektivität mit einem IP-Netz besitzt. Diese Aufgabe kann von einem CNR übernommen werden, der am Netzübergang ebenfalls IP beherrschen muss. Über einen Tunnel zwischen dem Dual-Stack Gerät, das ein HiiMap-Paket in ein IP-Paket einkapselt und dieses zum TR sendet, kann die Konnektivität zwischen diesen beiden Netzen hergestellt werden.

Der Zugang zu HiiMap mittels Dual-Stack von einem IP-Netz über einen TR kann dabei vom Betreiber des auf HiiMap umgestellten Netzes als Dienst angeboten werden, für den sich Benutzer registrieren müssen. Dies ist vergleichbar mit dem Dienst

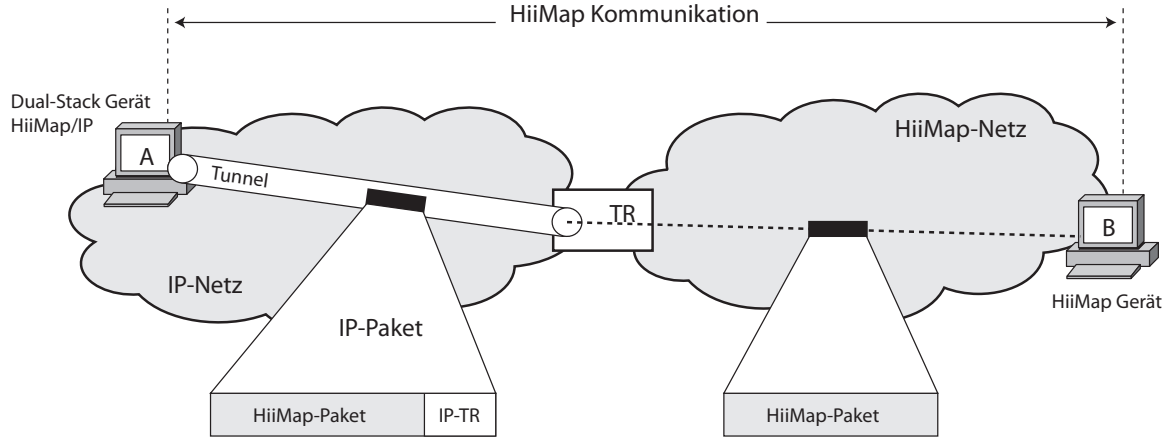


Abbildung 5.2.: Vernetzung von Dual-Stack-Gerät und HiiMap-Gerät mittels Tunnel

SixXS [PM11], der Benutzern mit ausschließlicher IPv4-Konnektivität ebenfalls mittels TR und Dual-Stack auch IPv6 anbieten kann. Durch die Registrierung bei diesem Dienst wird auch die IP-Adresse des TR sowie des Mapping-Systems dem Dual-Stack Gerät mitgeteilt. Folgende Voraussetzungen müssen aber für eine erfolgreiche Kommunikation gegeben sein:

- Sowohl Gerät A als auch Gerät B besitzen einen gültigen UID vom Typ 1 oder 2. Die Registrierung des UID für Gerät A kann zusammen mit der Registrierung für den Tunnel-Dienst über die regionale Registrierungsstelle abgewickelt werden.

- Gerät A registriert beim Mapping-System folgende Locator-Werte für seinen UID: LTA=aktuelle IP-Adresse; GGUID=0.

- Gerät B ist mit LTA und GGUID des HiiMap-Netzes beim Mapping-System registriert.

Der Ablauf der Kommunikation zur in Abbildung 5.2 dargestellten Vernetzung zwischen Gerät A und Gerät B wird im Folgenden erläutert. Zunächst stellt Gerät A eine Mapping-Anfrage für den UID von Gerät B. Es wird davon ausgegangen, dass dieser UID dem Gerät A bekannt ist. Nach dem Empfang der Locator-Information erzeugt Gerät A ein HiiMap-Paket und verwendet als Quell-LTA die eigene aktuelle IP-Adresse und als Quell-GGUID den Wert 0. Dieses HiiMap-Paket wird von Gerät A mit seiner eigenen IP als Absenderadresse in ein IP-Paket gekapselt und an die IP-Adresse des TR gesendet. Damit wird ein IP-Tunnel zwischen Gerät A und dem TR gebildet, in dem das HiiMap-Paket transportiert wird.

Das IP-Paket wird vom TR entkapselt und das darin enthaltene HiiMap-Paket unter Verwendung der Ziel-Locator-Werte und des von diesem HiiMap-Netz verwendeten Vermittlungsprotokolls zu Gerät B geroutet. Dieses empfängt das Paket und wertet die Daten aus. Das HiiMap-Antwortpaket wird an den UID von Gerät A adressiert, mit

dessen IP-Adresse als Ziel-LTA und der Ziel-GGUID 0. Diese Werte werden vom empfangenen Paket übernommen. Nach der Routingvorschrift in HiiMap wird jeder GGUID, dessen Betreiberkennung ungleich der eigenen GGUID ist, an einen CNR geleitet, der auch die Aufgabe des TR übernimmt. Dieser erkennt am Wert 0 der Ziel-GGUID, dass es sich um ein Dual-Stack Gerät handelt, das nur über IP erreichbar ist und kapselt das HiiMap-Paket in ein IP-Paket. Die IP-Adresse des Ziels übernimmt er aus dem Ziel-LTA-Feld des HiiMap-Pakets.

Da bei dieser Art der Vernetzung zwischen beiden Geräten eine reine HiiMap Kommunikation zustande kommt, unterstützt diese Variante auch die Adressierung von Inhalten und Personen. Voraussetzung ist lediglich eine mit dem Inhalts-Modul ausgestattete HiiMap-Middleware auf den kommunizierenden Geräten.

5.3.1.2. Tunnel zwischen zwei Tunnelroutern

Eine weitere Möglichkeit zur Vernetzung ist in Abbildung 5.3 dargestellt. Dabei kommt ein auf IP basierendes Transit-Netz zwischen bereits vollständig auf HiiMap umgestellten Netzen zum Einsatz. Jedes HiiMap-Netz besitzt wie im vorherigen Fall einen TR, an dem der Übergang ins IP-Netz erfolgt.

Um HiiMap-Pakete von einem HiiMap-fähigen Netz über IP in ein anderes zu senden, müssen die TR in der Lage sein, die zu einem GGUID zugehörige IP-Adresse des zuständigen TR herausfinden zu können. Dafür existieren zwei Möglichkeiten, die jeweils eine eigene Routingtabelle mit dieser Zuordnung benötigen. Zum einen kann ein eigenes Routingprotokoll verwendet werden, das diese Tabelle verwaltet und aktualisiert. Zum anderen kann diese Zuordnung mit Hilfe von DNS und einer eigenen neuen Top-Level-Domain wie *.gguid* realisiert werden. Dabei kann der Ziel-GGUID vom TR wie ein Gerätename im DNS abgefragt werden. Als Antwort wird die IP-Adresse des entsprechenden TR zurück geliefert. Im folgenden wird die Variante unter Verwendung des DNS verwendet, weil dadurch die Routingtabelle ohne die Einführung eines zusätzlichen Routingprotokolls genutzt werden kann. Die Ergebnisse der DNS-Abfrage werden mit Zeitstempel in diese Tabelle eingetragen und besitzen eine lange Gültigkeit, da nur selten Änderungen zu erwarten sind. Die Last auf DNS wird dadurch deutlich reduziert.

Für eine erfolgreiche Kommunikation werden folgende Punkte vorausgesetzt:

- Beide HiiMap-Geräte besitzen einen gültigen UID vom Typ 1 oder 2 und sind mit ihren zugewiesenen Locatorwerten im Mapping-System registriert.
- Die TR der HiiMap-Netze besitzen für ihre GGUID gültige DNS-Einträge unter der .gguid-Top-Level-Domäne.

Sind diese Voraussetzungen gegeben, ist eine Kommunikation zwischen zwei Geräten mit ausschließlicher HiiMap-Konnektivität über ein IP-Transitnetz möglich. Die Kommunikation zwischen den Geräten A und B aus Abbildung 5.3 läuft wie folgt ab. Gerät A erfragt zunächst mit dem bekannten UID von Gerät B dessen Locator-Informationen aus dem Mapping-System. Anschließend erzeugt es mit diesen und seinen eigenen Locator-Werten ein HiiMap-Paket. Da der Ziel-GGUID unterschiedlich ist, wird das Paket gemäß dem in diesem HiiMap-Netz verwendeten Routingverfahren an einen CNR geroutet, der auch als TR fungiert.

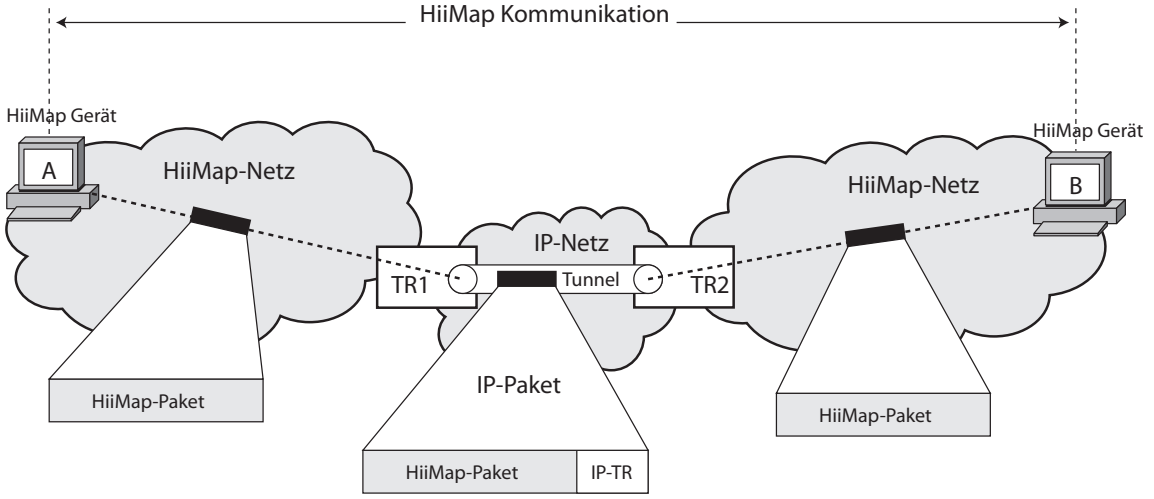


Abbildung 5.3.: Vernetzung zweier HiiMap-Geräte über ein IP-Netz mittels Tunnel

Der CNR versucht das Paket anhand seiner globalen Routingtabelle direkt über HiiMap in das Ziel-Netz zu routen. Existiert dafür jedoch kein Eintrag, übernimmt der CNR direkt die Funktion des TR und versucht, das HiiMap-Paket mittels IP-Tunnel in das Ziel-Netz zu senden. Dazu ermittelt TR1 über das DNS die IP des TR, der für die gesuchte Ziel-GGUID zuständig ist. Anschließend erzeugt der TR ein IP-Paket mit der Zieladresse von TR2, in dem das HiiMap-Paket eingekapselt wird.

TR2 empfängt das IP-Paket, entkapselt das darin enthaltene HiiMap-Paket und sendet dieses direkt zu Gerät B. Die Prozedur für Antwortpakete verläuft gleich, lediglich die Mappinganfrage zu Beginn ist nicht mehr nötig, da die Locator-Informationen von Gerät A bereits bekannt sind. Da mit dieser Art der Vernetzung zwischen beiden kommunizierenden Geräten eine reine HiiMap-Kommunikation stattfindet, wird auch die Adressierung von Personen und Inhalten unterstützt, eine entsprechend ausgestattete Middleware vorausgesetzt.

5.3.1.3. Tunnel zwischen zwei Endgeräten

Neben den beiden bisher beschriebenen Formen der Vernetzung kann eine HiiMap-Kommunikation zwischen zwei Dual-Stack Geräten auch ausschließlich über bereits bestehende IP-Netze geführt werden. Dabei müssen beide Geräte die gleiche IP-Version beherrschen, da die Tunnelendpunkte direkt bei den Geräten liegen.

Dieses Verfahren stellt prinzipiell die einfachste Variante dar, wie eine Kommunikation über HiiMap erfolgen kann, ohne dass ein Betreiber sein Netz auf HiiMap umstellt. Die Vernetzung ist in Abbildung 5.4 dargestellt. Es gelten dabei folgende Voraussetzungen:

- Gerät A und B besitzen einen gültigen UID vom Typ 1 oder 2. Beide registrieren folgende Locator-Werte beim Mapping-System: LTA=jeweils aktuelle IP; GGUID=0.
- Beide Geräte müssen die gleiche IP-Version verwenden. Ein weiterer Tunnel der Art

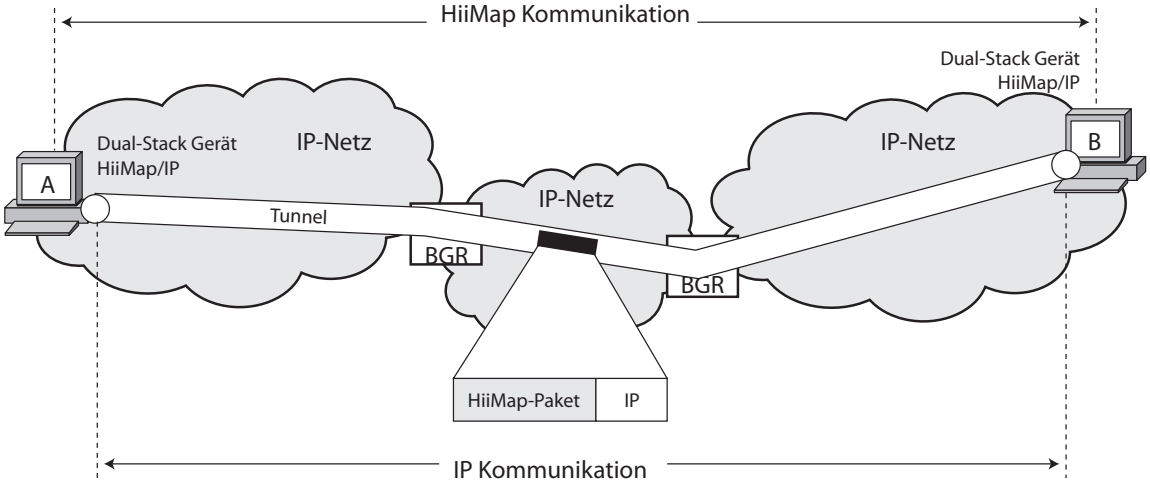


Abbildung 5.4.: Vernetzung zweier Dual-Stack-Geräte über IP-Netze mittels Tunnel

IPv6-in-IPv4 ist zwar grundsätzlich möglich, sollte aber wegen der damit erzeugten Blindlast vermieden werden.

Für eine HiiMap-Kommunikation zwischen den beiden Geräten A und B erfragt Gerät A zunächst die Locator-Informationen von Gerät B mit dessen bekanntem UID. Gerät A erkennt anhand des GGUID mit dem Wert 0, dass es sich um ein Dual-Stack Gerät handelt, das über IP angebunden ist. Da Gerät A selbst ein Dual-Stack Gerät mit IP-Anbindung ist, erkennt es, dass ein IP-Tunnel zur HiiMap-Kommunikation nötig ist.

Gerät A erstellt ein HiiMap-Paket mit den bekannten Locator-Werten und kapselt dieses in ein IP-Paket, wobei die LTA von Gerät B als Ziel-IP verwendet wird. Das IP-Paket wird zu Gerät B gesendet. Dieses entfernt den IP-Header und wertet die Informationen aus dem HiiMap-Paket aus. Die Antwort erfolgt nach dem gleichen Schema.

5.3.2. Kommunikation zwischen HiiMap und IP mittels Translation

Mit der Vernetzung mittels Translation wird die Kommunikation eines HiiMap-Gerätes mit einem IPv6-Gerät ermöglicht, wobei keines der beiden Geräte einen Dual-Stack besitzt. Ein Tunnel zwischen beiden Geräten ist daher ausgeschlossen, da sie kein gemeinsames Vermittlungsprotokoll beherrschen. Über einen Konverters (KV), der eine Protokolltranslation von HiiMap zu IPv6 und umgekehrt durchführt, ist dennoch eine Kommunikation zwischen derartigen Geräten möglich. Dieser Fall ist in Abbildung 5.5 dargestellt und kommt zur Anwendung, wenn ein bestehendes IPv6-Netz mit einem reinen HiiMap-Netz erweitert werden soll. Gerät A besitzt dabei ausschließlich IPv6-Konnektivität und Gerät B ausschließlich HiiMap-Konnektivität. Der KV befindet sich an der Schnittstelle beider Netze.

Da in diesem Fall IP-Pakete zu HiiMap-Paketen und umgekehrt übersetzt und nicht gekapselt werden, gelten folgende Vorraussetzungen und Einschränkungen:

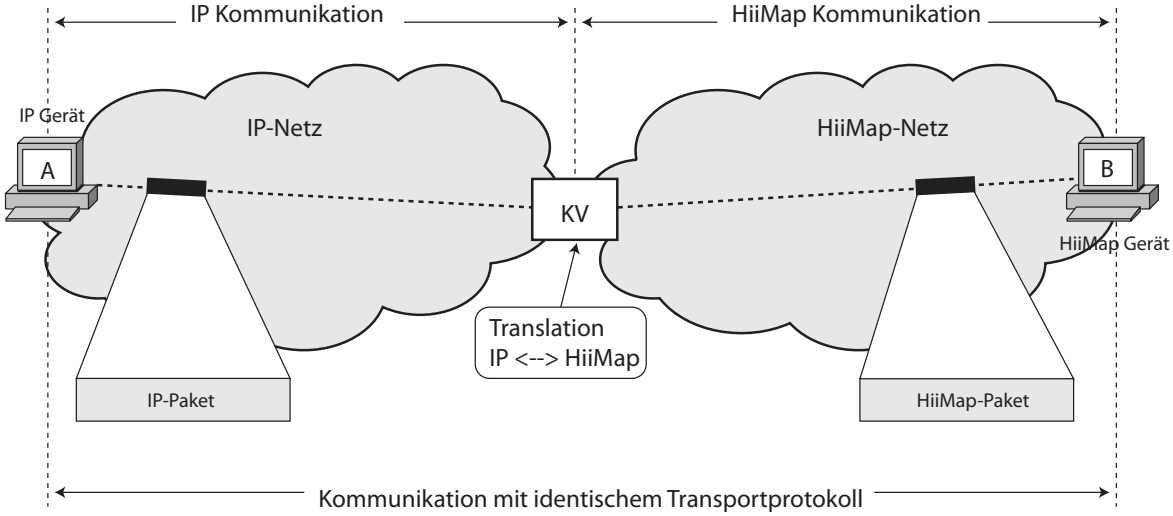


Abbildung 5.5.: Vernetzung von HiiMap-Gerät und IP-Gerät über Konverter

- Gerät A muss IPv6 unterstützen.
- Gerät B besitzt einen gültige UID vom Typ 1 oder 2 und ist mit seiner zugewiesenen LTA und seinem GGUID beim Mapping-System registriert.
- Beide Geräte müssen das gleiche Transportprotokoll verwenden. HiiMap dient dabei ausschließlich als Vermittlungsprotokoll.
- Der UID von Gerät B muss Gerät A bekannt sein. Dies kann über eine Suchmaschine oder über eigene DNS-Einträge für Geräte im HiiMap-Netz erfolgen. Dabei können Gerätenamen im DNS-Format in den entsprechenden UID umgesetzt werden. Da diese, wie IPv6-Adressen, ebenfalls 128 Bit breit sind, ist dies ohne weiteres möglich.
- Gerät A verwendet den UID von Gerät B als Ziel-IP-Adresse. Daher muss im IPv6-Netz eine Route existieren, die IPv6-Adressen, deren höchstwertiger 8 Bit Adressblock die Werte 1 oder 2 enthält, zum KV routet.
- Gerät B verwendet für die Kommunikation mit Gerät A dessen IPv6-Adresse als UID. Dazu wird der neu eingeführte UID-Typ 32 verwendet.

Die Kommunikation der Geräte A und B mittels Translation läuft folgendermaßen ab: Zunächst erfragt Gerät A mit Hilfe einer Suchmaschine oder über DNS den UID von Gerät B, der aus der Sicht von Gerät A eine normale IPv6-Adresse ist. Gerät A erzeugt ein IPv6-Paket mit dem UID von Gerät B als Ziel-IP-Adresse. Nach der Routingvorschrift in diesem IP-Netz erreicht dieses Paket den KV.

Der KV erkennt anhand der Ziel-IP-Adresse, dass es sich nicht um eine normale globale IPv6-Adresse, sondern um einen HiiMap-UID handelt und sucht aus dem Mapping-System die aktuell dafür gültigen Locator-Werte. Danach entfernt der KV den IP-Header

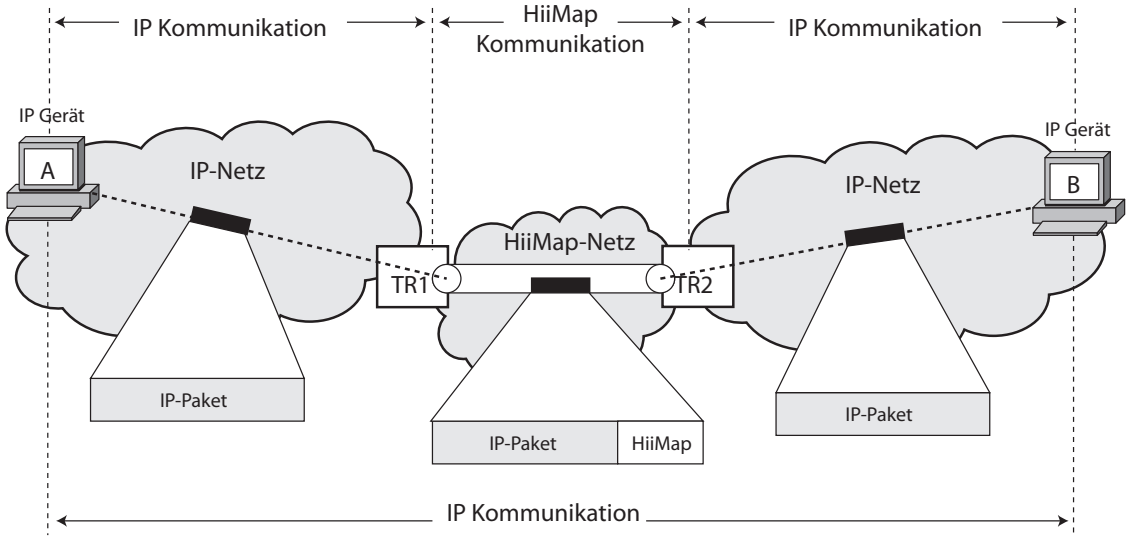


Abbildung 5.6.: Vernetzung von IP-Endgeräten über ein HiiMap-Netz

und erzeugt einen neuen HiiMap-Header mit der Ziel-LTA und -GGUID von Gerät B. Als Quell-UID und -LTA wird die IPv6-Adresse von Gerät A verwendet. Der Quell-GGUID wird zu 0 gesetzt. HiiMap übernimmt in diesem Fall ausschließlich die Aufgaben der Vermittlungsschicht. Die Informationen aus der Transportschicht werden unverändert übernommen.

Gerät B empfängt das HiiMap-Paket und erkennt anhand des Quell-UID-Typs, dass es sich hierbei um eine IPv6-Adresse handelt. Für die Antwort wird ein HiiMap-Paket mit der IPv6-Adresse von Gerät A als Ziel-UID und -LTA erzeugt. Die Ziel-GGUID wird zu 0 gesetzt. Pakete mit einem von der eigenen Betreiberkennung abweichenden Ziel-GGUID werden zum KV geroutet.

Der KV erkennt anhand der Ziel-UID, dass es sich um ein IP-Gerät handelt und erstellt ein IP-Paket mit dem Ziel-UID als Ziel-IP-Adresse. Die Transportschicht wird ebenfalls direkt aus dem HiiMap-Paket übernommen.

Der KV übernimmt für das HiiMap-Netz die Aufgabe des CNR als Standard-Gateway, sofern dieses Netz nur über ein IP-Netz erreichbar ist. Besitzt das Netz mehrere CNR, die Konnektivität zu weiteren HiiMap-Netzen ermöglichen, muss eine eigene Route für den GGUID-Wert 0 zum KV eingerichtet werden.

Aufgrund der Tatsache, dass in diesem Fall der Vernetzung eine Protokoll-Translation zwischen zwei verschiedenen Vermittlungsprotokollen stattfindet und auf beiden Seiten das gleiche Transportprotokoll verwendet werden muss, wird ausschließlich die Adressierung von Geräten unterstützt.

5.3.3. IP-Kommunikation über HiiMap

Die Vernetzung von IP-Netzen kann auch über ein HiiMap-Netz als Transitnetz erfolgen, wie in Abbildung 5.6 dargestellt ist. Diese Lösung ist sinnvoll, falls der Betreiber eines Transit-Netzes auf HiiMap umstellt, die an ihn angeschlossenen Netze aber wei-

terhin IP verwenden. Um die IP-Pakete durch das HiiMap-Netz zu senden, kommt ein Tunneling-Verfahren mit zwei TR zum Einsatz. Im Unterschied zu den bereits vorgestellten Verfahren wird hierbei das IP-Paket in ein HiiMap-Paket gekapselt. Für den ersten TR muss daher die Zuordnung von Ziel-IP-Adresse zum GGUID des dafür zuständigen zweiten TR möglich sein. Ähnlich dem umgekehrten Tunneling von HiiMap in IP ist daher eine Routingtabelle nötig, die mit einem eigenen Routingprotokoll verwaltet wird, oder es wird auf das Mapping-System zurückgegriffen. Um auf ein zusätzliches Routingprotokoll zu verzichten wird die Variante mit dem Mapping-System gewählt. Eine rückwärtige DNS-Anfrage mit den bereits existierenden .gguid-Einträgen kann nicht verwendet werden, da in dem aktuellen Fall das IP-Präfix gesucht wird, das der zweite TR erreichen kann, dessen IP-Adresse jedoch nicht explizit bekannt ist. Im Mapping-System wird daher das IP-Präfix des IP-Netzes als UID des Typs $T = 0$ oder $T = 32$ angelegt, je nachdem ob es sich um ein IPv4- oder ein IPv6-Netz handelt. Der Mapping-Eintrag enthält die aktuell gültige GGUID des zuständigen TR. Für einen UID des Typs $T = 0$ und $T = 32$ existieren keine Einträge für die LTA.

Der Ablauf der Kommunikation der IP-Geräte A und B wird im Folgenden beschrieben, wobei dafür folgende Annahmen getroffen werden:

- Gerät A und B besitzen ausschließlich IP-Konnektivität. Es kann sowohl IPv4 als auch IPv6 verwendet werden, jedoch müssen beide Endgeräte das gleiche Protokoll anwenden. Die IP-Adressen sind gegenseitig bekannt, beispielsweise über DNS.
- Die Zuordnung zwischen Ziel-IP-Adresse und dafür zuständigem GGUID muss möglich sein. Dies erfolgt mit Mapping-Einträgen, die IP-Präfixe auf den GGUID der TR abbilden, über welche die TR erreicht werden können.
- Für einen UID des Typs $T = 0$ und $T = 32$ ist das Mapping-System in der Lage, „unscharfe“ Suchen mit der größten Übereinstimmung durchzuführen. Dies bedeutet, dass der Mapping-Eintrag zurückgeliefert wird, bei dem die meisten Stellen der höchstwertigen Bits des UID übereinstimmen. Dieses Verfahren ist nötig, da aus der Ziel-IP-Adresse das Netz-Präfix nicht direkt abgeleitet werden kann.

Gerät A erzeugt ein IP-Paket mit der Ziel-IP-Adresse von Gerät B. Der Tunnelrouter TR1, der als Standard-Gateway konfiguriert ist, empfängt das Paket und stellt eine Anfrage an das Mapping-System mit der Ziel-IP-Adresse als UID. Das Mapping-System erkennt anhand des UID-Typs, dass es sich um eine IP-Adresse handelt und liefert den GGUID des zuständigen zweiten Tunnelrouters TR2 zurück. Dieses Ergebnis wird von TR1 zwischengespeichert, um weitere Anfragen für das zurückgelieferte IP-Präfix direkt beantworten zu können.

Mit dem erhaltenen Ziel-GGUID und seinem eigenen GGUID erzeugt TR1 anschließend ein HiiMap-Paket. Als Ziel-UID wird die Ziel-IP-Adresse verwendet, als Quell-UID die Quell-IP-Adresse. Die Werte für die Ziel-LTA bleiben in diesem Fall leer, da das Paket ausschließlich anhand des GGUID im HiiMap-Netz geroutet wird. Das IP-Paket wird vollständig in das HiiMap-Paket gekapselt.

TR2 empfängt das HiiMap-Paket, erkennt am Typ des UID, dass es sich hierbei um ein gekapseltes IP-Paket handelt und entpackt dieses. Das IP-Paket wird anschließend zu Gerät B gesendet. Das gleiche Verfahren wird für die Antwort zurück angewandt.

Diese Art der Vernetzung ermöglicht es, eine IP-Kommunikation zwischen Endgeräten vollkommen transparent über ein HiiMap-Netz zu leiten. Die benötigten Tunnelrouter übernehmen aus der Sicht des HiiMap-Netzes die Aufgabe eines CNR und aus der Sicht des IP-Netzes die Aufgabe eines BGR.

5.3.4. Funktionsbeschreibung der Helferkomponenten

Für die Koexistenz zwischen HiiMap und IP kommen zwei Arten von Helferkomponenten zum Einsatz: Tunnelrouter TR und Konverter KV. Jede dieser Komponenten übernimmt verschiedene Aufgaben in Bezug auf die Weiterleitung der Daten. Dennoch muss nicht jede Helferkomponente als jeweils einzelnes Gerät realisiert sein. Es ist vielmehr möglich, dass ein Gerät gleichzeitig die Aufgaben eines TR, KV sowie eines BGR und CNR übernehmen kann. Ein derartiges multifunktionales Gerät wird im Folgenden als Super-Gateway (SGW) bezeichnet. Die Auswahl des jeweiligen Betriebsmodus, der die Aufgaben einer Helferkomponente übernimmt, wird dabei über bestimmte Wertekombinationen in den IP- und HiiMap-Headern gesteuert. Damit kann an den Netzgrenzen eine einzige Router-Implementierung für jede erdenkliche Art der Vernetzung eingesetzt werden.

Es wird davon ausgegangen, dass jedes SGW mindestens zwei Netzschnittstellen besitzt, die je nach Art der Vernetzung mit einem IP-Netz oder einem HiiMap-Netz verbunden sind.

5.3.4.1. Empfang eines IP-Pakets

Abbildung 5.7 zeigt das Ablaufdiagramm des SGW beim Empfang eines IP-Pakets. Zunächst wird überprüft, ob die Ziel-IP-Adresse einem UID entspricht. Dies ist der Fall, wenn die höchstwertigen 8 Bit der IPv6 Adresse die Werte 1 oder 2 besitzen, entsprechend dem Typ $T = 1$ oder $T = 2$ für Geräte-UID. Das SGW übernimmt anschließend den Betriebsmodus Konverter. Andernfalls sind weitere Abfragen nötig. Im folgenden Weden die nötigen Betriebsmodi und ihre Funktion beschrieben.

Konverter Im Betriebsmodus Konverter findet beim Empfang eines IP-Pakets eine Protokoll-Translation von IP zu HiiMap statt. Dazu erfragt das SGW aus dem Mapping-System zunächst die Locator-Informationen für den Ziel-UID. Anschließend wird der IP-Header vom Paket entfernt und ein neuer HiiMap-Header erzeugt. Dieser enthält als Quell-UID und -LTA die IP-Adresse des Absenders. Der Quell-GGUID wird zu 0 gesetzt.

Wird als Transportprotokoll TCP verwendet, ergibt sich ein Sonderfall bezüglich der Prüfsumme im TCP-Header, da diese auch aus Informationen des IP-Header berechnet wird (TCP Pseudo-Header) [RFC81b]. Per Definition werden für die Übertragung von TCP mittels HiiMap (Protokollnummer 6) die korrespondierenden Felder des HiiMap-Header als Pseudo-Header verwendet, um die Prüfsumme zu berechnen. Damit kann die Prüfsumme direkt übernommen werden.[1] Anschließend wird das HiiMap-Paket über die

[1] Verwenden zwei HiiMap-Geräte TCP als Transportprotokoll, muss der TP-Wrapper der HiiMap-Middleware die TCP-Prüfsumme erneut berechnen. Ansonsten muss die TCP-Implementierung speziell für HiiMap modifiziert werden, um die richtigen Werte für den Pseudo-Header zu erhalten.

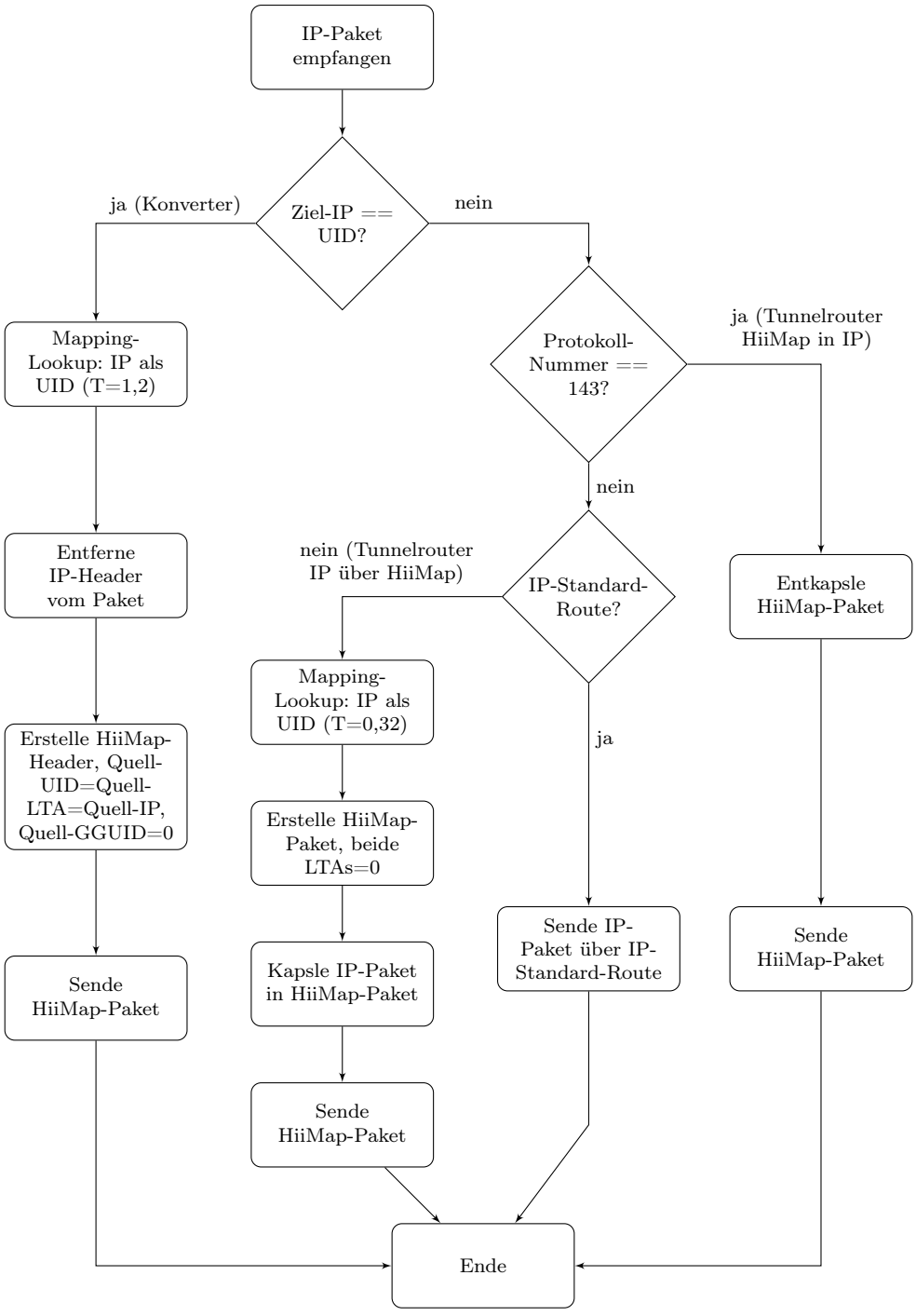


Abbildung 5.7.: Ablaufdiagramm beim Empfang eines IP-Pakets

HiiMap-Schnittstelle zum Ziel-Gerät gesendet.

Entspricht die Ziel-IP-Adresse nicht einem UID, wird im nächsten Schritt die Protokollnummer des IP-Pakets überprüft. Ist diese 143, handelt es sich um einen Tunnel der Art „HiiMap in IP“. Das SGW übernimmt dann die Aufgabe eines Tunnelrouters. Entspricht die Protokollnummer nicht 143, ist eine weitere Abfrage nötig.

Tunnelrouter (HiiMap in IP) Dieser Betriebsmodus kommt zum Einsatz, wenn ein Dual-Stack Gerät mit IP-Konnektivität mit einem HiiMap-Gerät kommuniziert. In diesem Fall ist das HiiMap-Paket im IP-Paket eingekapselt. Das SGW entfernt den IP-Header und entkapselt auf diese Weise das HiiMap-Paket. Dieses wird anschließend über die HiiMap-Schnittstelle zum Ziel-Gerät gesendet.

Entspricht die Protokollnummer nicht 143, wurde ein normales IP-Paket empfangen. Das SGW überprüft in diesem Fall, ob eine Standard-Route für IP-Pakete existiert. Ist dies der Fall, übernimmt er den Betriebsmodus eines normalen BGR. Andernfalls muss das IP-Paket über HiiMap durch einen Tunnel der Art „IP über HiiMap“ gesendet werden. In diesem Fall kommt erneut der Betriebsmodus Tunnelrouter zum Einsatz.

BGR Existiert eine Standard-Route für IP, so kann das empfangene Paket direkt über IP geroutet werden, ohne dass ein Tunnel oder eine Translation nötig ist. Das Paket wird über diese Route gesendet. Das SGW fungiert daher als normaler BGR.

Tunnelrouter (IP über HiiMap) Existiert keine Standard-Route für IP, so handelt es sich um ein Transit-Netz, das bereits auf HiiMap umgestellt hat. Das SGW muss daher das IP-Paket über einen HiiMap-Tunnel zum Ziel senden. Dazu erfolgt zunächst eine Mapping-Anfrage, welches SGW für die entsprechende Ziel-IP-Adresse zuständig ist. Die Anfrage erfolgt mit der Ziel-IP als UID. Aufgrund des UID-Typs (0 für IPv6 und 32 für IPv6) liefert das Mapping-System den Eintrag mit der besten Übereinstimmung zurück, der den GGUID des Ziel-SGW enthält.

Anschließend erzeugt das SGW ein HiiMap-Paket mit den Daten des Mapping-Systems, wobei sowohl die Quell- als auch die Ziel-LTA zu 0 gesetzt werden. Das IP-Paket wird vollständig in das HiiMap-Paket gekapselt und die Protokollnummer im HiiMap-Paket auf 143 gesetzt. Im letzten Schritt wird das HiiMap-Paket über die HiiMap-Schnittstelle zum Ziel-SGW versendet.

5.3.4.2. Empfang eines HiiMap-Pakets

Abbildung 5.8 zeigt das Ablaufdiagramm des SGW beim Empfang eines HiiMap-Pakets. Nach dem Empfang wird zunächst der Typ des Ziel-UID überprüft. Handelt es sich dabei um einen UID, der nicht vom Typ $T = 0$ oder $T = 32$ ist, erfolgt eine weitere Überprüfung der Ziel-GGUID. Ist dieser 0, handelt es sich um einen Tunnel der Art „HiiMap in IP“. Das SGW schaltet damit in den Betriebsmodus Tunnelrouter. Im folgenden werden die benötigten Betriebsmodi und ihre Funktion beschrieben.

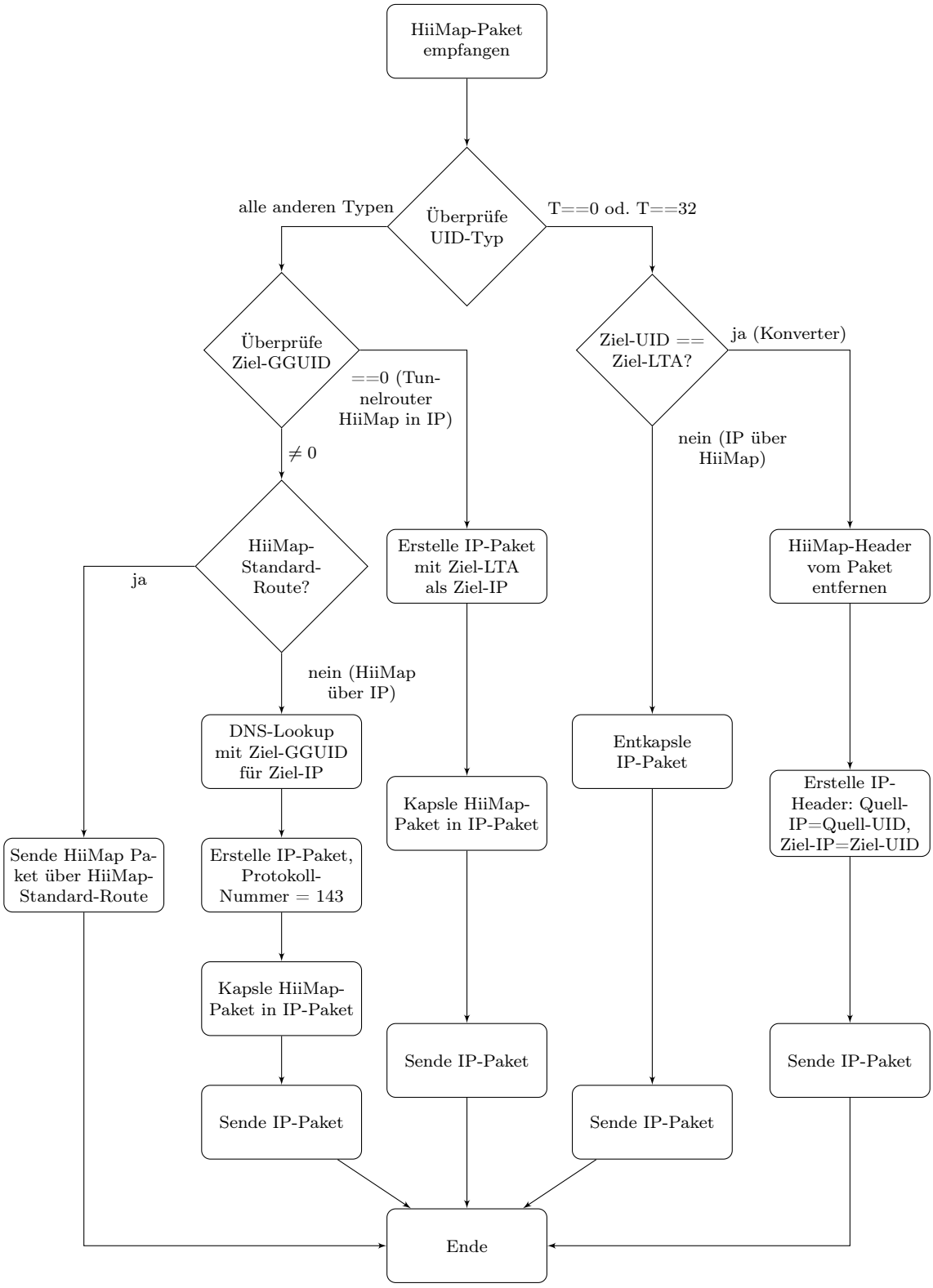


Abbildung 5.8.: Ablaufdiagramm beim Empfang eines HiiMap-Pakets

Tunnelrouter (HiiMap in IP) Dieser Betriebsmodus kommt zum Einsatz, wenn ein HiiMap-Gerät Daten an ein Dual-Stack Gerät sendet, das nur IP-Konnektivität besitzt. Das SGW erstellt in diesem Fall ein IP-Paket mit der Ziel-LTA als Ziel-IP und seiner eigenen IP-Adresse als Absender-IP. Das HiiMap-Paket wird in dieses IP-Paket gekapselt und die Protokollnummer im IP-Header auf 143 gesetzt. Anschließend wird das Paket auf der IP-Schnittstelle versendet.

Besitzt der Ziel-GGUID einen Wert ungleich 0, so handelt es sich grundsätzlich um ein Standard-HiiMap-Paket. In diesem Fall erfolgt eine Überprüfung, ob eine Standard-Route für HiiMap-Pakete existiert, sodass dieser GGUID direkt mit HiiMap erreicht werden kann. In diesem Fall übernimmt das SGW die Aufgabe eines CNR. Anderenfalls handelt es sich um einen Tunnel der Art „HiiMap über IP“, wofür erneut der Betriebsmodus des Tunnelrouters zum Einsatz kommt.

CNR Existiert eine HiiMap-Route für den entsprechenden Ziel-GGUID, so kann das empfangene Paket direkt ohne Tunnel oder Translation zum Ziel-CNR gesendet werden. Das SGW fungiert als CNR.

Tunnelrouter (HiiMap über IP) Falls keine Route für HiiMap-Pakete existiert, so handelt es sich um ein reines IP-Transit-Netz. Die HiiMap-Pakete müssen daher über einen IP-Tunnel zum Ziel gesendet werden. Zunächst erfolgt eine DNS-Anfrage des SGW, um die IP-Adresse des Ziel-SGW anhand des Ziel-GGUID zu erhalten.

Mit dieser IP-Adresse erzeugt der SGW ein IP-Paket und verwendet seine eigene IP-Adresse als Absender-Adresse. Die Protokollnummer im IP-Header wird auf 143 gesetzt und anschließend das HiiMap-Paket in das IP-Paket gekapselt. Im letzten Schritt wird das IP-Paket über die IP-Schnittstelle an das Ziel-SGW gesendet.

Entspricht der Ziel-UID-Typ eines empfangenen Pakets den Werten $T = 0$ oder $T = 32$, so handelt es sich um eine als UID dargestellte IP-Adresse. In einem weiteren Schritt werden Ziel-UID und Ziel-LTA miteinander verglichen. Stimmen beide überein, wechselt das SGW in den Betriebsmodus Konverter. Anderenfalls handelt es sich um einen Tunnel der Art „IP über HiiMap“, der mit dem Betriebsmodus Tunnelrouter bearbeitet wird.

Konverter Im Betriebsmodus Konverter findet beim Empfang eines HiiMap-Pakets eine Protokoll-Translation von HiiMap zu IP statt. Da im HiiMap-Header alle benötigten Informationen zur Erstellung des IP-Pakets bereits enthalten sind, ist keine weitere DNS- oder Mapping-Anfrage nötig. Das SGW entfernt den HiiMap-Header vom Paket und erstellt einen neuen IP-Header mit dem Quell-UID als Quell-IP-Adresse und dem Ziel-UID als Ziel-IP-Adresse. Im Falle von TCP als Transportprotokoll (Protokollnummer 6) muss die Prüfsumme nicht erneut berechnet werden, da diese nach der Berechnungsvorschrift für HiiMap unverändert bleibt. Anschließend wird das IP-Paket auf der IP-Schnittstelle an das Zielgerät gesendet.

Tunnelrouter (HiiMap über IP) Entspricht der Ziel-UID nicht der Ziel-LTA, handelt es sich um ein IP-Paket, das über ein HiiMap-Transit-Netz getunnelt wird. Das so SGW entkapselt das IP-Paket, indem der HiiMap-Header entfernt wird. Anschließend wird das IP-Paket auf der IP-Schnittstelle zum Zielgerät gesendet.

5.4. Zusammenfassung

In diesem Kapitel wurden Konzepte zur Koexistenz von HiiMap und IP vorgestellt. Da eine Migration von IP zu HiiMap an einem Stichtag aufgrund der hohen Anzahl an Geräten nicht umsetzbar ist, stellen Konzepte zur Koexistenz beider Systeme die einzige Möglichkeit für eine vollständige Migration zu HiiMap dar. Alle potentiellen Arten der Vernetzung von HiiMap mit IP wurden mit entsprechenden Lösungsansätzen vorgestellt, wobei jeweils auf die Mechanismen Tunneln oder Translation zum Übertragen der Daten zurückgegriffen wurde. In diesem Zusammenhang wurde auch das UID-Schema für Geräte um zwei weitere Typen erweitert, die es erlauben IP-Adressen als UID zu kodieren.

Darüber hinaus wurde das Konzept eines Super-Gateway (SGW) vorgestellt. Dabei handelt es sich um eine Funktionsbeschreibung einer Implementierung, die je nach Anforderung automatisch die Aufgabe eines Konverters, Tunnelrouters, BGR oder CNR übernehmen kann. Die Bedingungen, anhand deren der jeweilige Betriebsmodus ausgewählt wird, werden dabei ausschließlich über die Belegung der Datenfelder in den Packet-Headern definiert. Somit ist keine feste Konfiguration eines bestimmten Betriebsmodus notwendig.

6. Implementierung von HiiMap

Um die Machbarkeit der HiiMap-Architektur auch in einem realen Umfeld unter Beweis zu stellen, wurde ei Prototyp implementiert. Der Fokus lag dabei auf dem einfachen Versenden und Empfangen von Datenpaketen, so dass im Folgenden ausschließlich das Konzept der Geräteadressierung umgesetzt wurde. Im Zuge der Arbeit wurde auch ein Kernel-Modul für Linux entworfen, das es erlaubt, auf einfache Art und Weise mit der bereits existierenden Socket-API unter Auswahl einer neuen Adressfamilie, eine HiiMap-Kommunikation ausschließlich auf der Basis eines UID durchzuführen. Das Kernel-Modul übernimmt sämtliche notwendigen Schritten zur Umsetzung der Trennung von Locator und Identifier. Locatorwechsel und Aktualisierungen der Mapping-Einträge erfolgen für die Anwendung dabei völlig transparent. Für Test und Evaluierung des Prototyps wurde die G-Lab Testplattform verwendet [SGH+10, G-L].

Im folgenden Kapitel wird zunächst die verwendete Experimental-Plattform erläutert. Anschließend erfolgt eine Beschreibung der technischen Umsetzung des HiiMap-Protokolls. Diese umfasst die Festlegung des Packet-Headers sowie der verwendeten Topologie und des benötigten Discovery-Mechanismus. Anschließend wird die Implementierung des Kernel-Moduls sowie der neu eingeführten Adressfamilie und ihre Verwendung in der Socket-API beschrieben. Im letzten Abschnitt wird eine einfache Demo-Anwendung vorgestellt, die zur Erfassung von Messdaten sowie zur Verifizierung der inhärenten Mobilitätsunterstützung verwendet wird. Die Test- und Messergebnisse werden im Anschluss vorgestellt.

6.1. Die G-Lab Experimental-Plattform

Eine Experimental-Plattform stellt ein unverzichtbares Werkzeug zum Testen und Evaluieren neuer Internet-Architekturen und Protokolle dar. Besonders für revolutionäre Architekturen, die aufgrund geänderter Adressierungsschemata nicht ohne weiteres im aktuellen Internet getestet werden können, sind Experimentalplattformen unabdingbar [WHSM10]. Diese erlauben im Idealfall die Generierung und Weiterleitung von Paketen mit eigenem Header und bieten eine Testumgebung, die möglichst frei von unkontrollierbaren Effekten wie langen Paketlaufzeiten und -verlusten ist. Dennoch sollte es die Testumgebung auch ermöglichen sein, negative Effekte zu erzeugen, um Prototypen nicht nur unter idealisierten, sondern auch unter realistischen Bedingungen testen zu können.

Das vom Deutschen Bundesministerium für Bildung und Forschung (BMBF) geförderte Projekt German-Lab (G-Lab) umfasst neben theoretischen Forschungsvorhaben, in deren Rahmen auch die HiiMap-Architektur entwickelt wurde, eine Experimentalplattform bestehend aus etwa 180 Geräten. Diese sind verteilt auf sechs Universitäten in ganz Deutschland, um die theoretisch ausgearbeiteten Konzepte zu evaluieren. Jedes Gerät ist mit mehreren 1 GBit/s-Netzschnittstellen ausgestattet, die Standorte selbst sind eben-

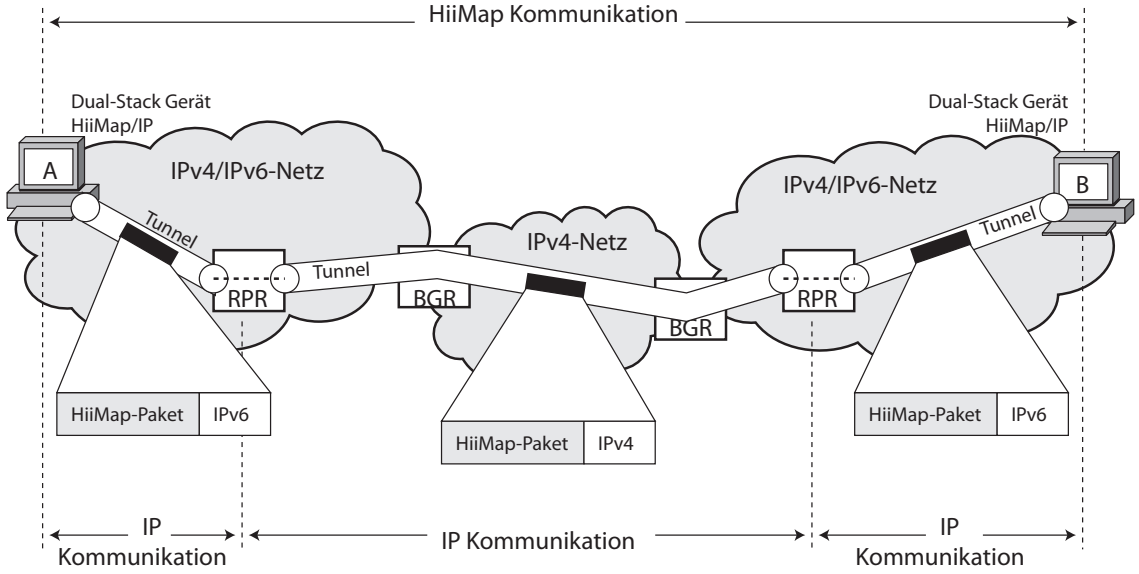


Abbildung 6.1.: HiiMap-Kommunikation zwischen zwei Dual-Stack-Geräten über IP-Netze mittels Tunnel zwischen RPR

falls jeweils mit 1 GBit/s über das Deutsche Forschungsnetz untereinander verbunden. Dank Virtualisierung kommen als Testumgebung sowohl Planet-Lab [Pla], als auch das eigens für G-Lab entwickelte ToMaTo [SHG+11] zum Einsatz. Mit ToMaTo kann eine beliebige Test-Topologie erstellt werden, die anschließend unter Zuhilfenahme virtueller Maschinen in der Experimentalplattform instantiiert wird. Dabei kann gewählt werden, auf welchem physikalischen Gerät der Experimentalplattform welche virtuelle Maschine instantiiert werden soll. Zusätzlich können Effekte wie Paketlaufzeiten, Bitfehler- und Paketverlustraten auf jeder einzelnen Verbindung emuliert werden. Jedes einzelne virtuelle Gerät innerhalb der ToMaTo-Topologie erhält eine öffentliche IPv4-Adresse und ist damit weltweit erreichbar. Die Konfiguration erfolgt per SSH und VNC. Für die Implementierung von HiiMap wird die ToMaTo-Umgebung gewählt, da diese es darüber hinaus erlaubt, eigene Betriebssystem-Images zu verwenden.

6.2. Umsetzung des HiiMap-Protokolls

In diesem Abschnitt wird der Packet-Header für die Implementierung von HiiMap definiert, die verwendete Topologie und das Tunnelig-Verfahren über IP vorgestellt, sowie das Discovery-Verfahren erläutert, mit dem notwendige Parameter konfigurationslos an die Endgeräte übermittelt werden können.

6.2.1. Topologie und Tunneling-Verfahren

Aufgrund der Tatsache, dass die Kommunikation zwischen den Geräte der G-Lab Experimentalplattform standortübergreifend ausschließlich mit IP möglich ist, wird HiiMap

im Tunnelmodus implementiert. Die für die Implementierung verwendete Topologie entspricht dabei im wesentlichen der Vernetzung von zwei Dual-Stack Geräten mittels direktem Tunnel zwischen den Endgeräten (vgl. Abschnitt 5.3.1.3). Um jedoch die Umsetzung des Adressierungsschemas zwischen LTA und GGUID zu verifizieren, wird der CNR als so genannter Re-Pack Router (RPR) realisiert (vgl. Abbildung 6.1). Dabei erfolgt die Kommunikation innerhalb eines Standortes der Experimentalplattform mit IPv6, zwischen den Standorten, quasi im Inter-Domain-Bereich, erfolgt die Kommunikation mit IPv4. Der RPR übernimmt die Umsetzung zwischen den beiden IP-Protokollen. Aus der Sicht der Endgeräte verhält sich der RPR wie ein TR.

Als Mapping-System kommt eine eigene Java-Implementierung einer D1HT-DHT zum Einsatz. Diese orientiert sich vollständig an der Spezifikation von HiiMap in der zweiten Version.

6.2.2. Kernel-Modul für Linux 2.6

Um eine einfach zu verwendende Schnittstelle für bestehende und zukünftige Anwendungen für das HiiMap-Protokoll bereitzustellen, wird ein eigenes Kernel-Modul mit neuer Adressfamilie entwickelt, das auf Geräten mit Linux-Betriebssystem und Kernelversion 2.6 eingesetzt werden kann [SS11a]. Die Implementierung umfasst sowohl den Stack der Endgeräte, sowie den der RPR. Es wird ausschließlich der Geräte-Block umgesetzt, da besonders die Kommunikation zwischen Geräten unter der Verwendung der Trennung von Locator und Identifier, sowie verschiedener Vermittlungsprotokolle untersucht werden soll. Inhalts- sowie Personenunterstützung sind damit nicht implementiert.

6.2.2.1. HiiMap Packet-Header

Der Packet-Header entspricht der Spezifikation in Abschnitt 3.2.4. Es erfolgt jedoch noch keine Berechnung der Prüfsumme, ebenso besitzen die Datenfelder für beide LTA eine feste Breite von 128 Bit, da hierfür ausschließlich IPv6-Adressen zum Einsatz kommen. Es erfolgt darüber hinaus ausschließlich eine ungesicherte Datagramm-übertragung, weshalb das Feld Steuerinformationen ebenfalls noch nicht verwendet wird. Die Größe des Paket-Headers beträgt mit den aktuell definierten Feldern 80 Byte.

6.2.2.2. Discovery Verfahren

Eine Zuteilung der LTA an das Endgerät ist für Tunneling-Verfahren nicht notwendig, da die IP-Adresse dafür verwendet wird. Es ist jedoch notwendig, dass die LTA sowie der GGUID des RPR bzw. CNR ebenso wie die IP-Adresse des Mapping-Systems dem Endgerät mitgeteilt werden. Zu diesem Zweck wird ein konfigurationsloses Discovery-Verfahren implementiert, das die erforderlichen Daten verteilt.

Jedes Gerät sendet dazu auf den Netzschnittstellen, denen noch kein RPR zugeteilt ist, eine `DISCOVERY_REQUEST`-Nachricht als Broadcast. Ein im Netzsegment vorhandener RPR antwortet darauf mit einer `DISCOVERY_REPLY`-Nachricht. Diese wird direkt an das anfragende Gerät gesendet und enthält die Locator-Informationen des RPR sowie die IP-Adresse des Mapping-Systems.

Diese Werte werden im /proc/net-Verzeichnis gespeichert und können auch manuell editiert werden. Die Konfiguration des UID erfolgt manuell beim Starten des Kernel-Moduls. Per Startskript kann der UID gemäß dem allgemeinen UID-Schema aus dem Gerätenamen berechnet und dem Modul übergeben werden. Nach der Zuteilung dieser Werte aktualisiert das Gerät seinen Mapping-Eintrag.

Diese Discovery-Nachrichten werden nur eingesetzt, wenn als Vermittlungsprotokoll IP verwendet wird. In einem reinen HiiMap-Netz werden die Nachrichten `REQ_LOCATOR` und `ASSIGN_LOCATOR` bzw. `REQ_NSUID_GLOB` und `ASSIGN_NSUID_GLOB` verwendet.

6.2.2.3. Socket-API und neue Adressfamilie

Im Gegensatz zu einer HiiMap-Implementierung auf Anwendungsebene, bei der sämtliche Aufgaben der Trennung von Locator und Identifier von der Anwendung selbst übernommen werden müssen, erlaubt es ein Kernel-Modul, für alle Anwendungen eine einheitliche Schnittstelle zur Verfügung zu stellen. Dabei werden alle notwendigen Aufgaben vom Kernel-Modul selbst und für die Anwendungen transparent abgewickelt. Es wird daher, zusätzlich zu den bereits existierenden Adressfamilien AF_INET für IPv4 und AF_INET6 für IPv6, noch die Adressfamilie AF_HIIMAP eingeführt. Beim Öffnen eines Sockets mit dieser Adressfamilie, wie im nachfolgenden Beispiel gezeigt, wird dem Kernel signalisiert, dass HiiMap für die Datenübertragung verwendet wird.

```
1 socket(AF_HIIMAP, SOCK_DGRAM);
2 sendto(UID);
```

Das Öffnen eines HiiMap-Sockets unterscheidet sich im Vergleich zu IPv4 oder IPv6 damit nur in der verwendeten Adressfamilie. Im Gegensatz zu IP werden Daten jedoch nicht an eine IP-Adresse gesendet, sondern an einen UID. Dem Kommando *sendto* wird damit ein UID übergeben. Anschließen erfolgt von Seiten des Kernel-Moduls die Abfrage der Locator-Werte vom Mapping-System mittels `LOCATOR_REQUEST`. Nach dem Empfang der Locator-Werte erzeugt das Kernel-Modul ein IP-Paket mit der Ziel-IP des RPR. Im Nutzdatenbereich dieses IP-Pakets folgt der HiiMap-Header und im Anschluss die Nutzdaten. Das IP-Paket enthält die Protokollnummer 143.

Das Kernel-Modul unterstützt aktuell ausschließlich die Datagrammübertragung. Ein gesicherter Datentransport ist noch nicht möglich. Aufgrund der engen Verbindung von TCP mit IP müsste der TCP-Header vom HiiMap-Modul selbst erzeugt werden, daher kann TCP in dieser Version als Transportprotokoll noch nicht verwendet werden.

6.2.2.4. Unterstützung für Mobilität

Die inhärente Unterstützung von Mobilität ist einer der Hauptvorteile bei der Trennung von Locator und Identifier. Um diese bereit zu stellen sind für das Kernel-Modul folgende Aufgaben zu erfüllen:

UID-Cache Zu jedem UID, zu dem Daten gesendet werden, wird ein Eintrag im UID-Cache erzeugt. Zum einen wird damit die Last auf das Mapping-System reduziert, da

Anfragen auch lokal bedient werden können. Zum anderen ist dieser Cache nötig, um die Geräte, mit denen aktuell Daten ausgetauscht werden, über Änderungen der eigenen Locator-Werte informieren zu können. Jedes Cache-Element besitzt einen Zeitstempel, der zurückgesetzt wird, falls der UID verwendet wird.

Überwachung der Netzschnittstellen Mobilität eines Gerätes ist dadurch gekennzeichnet, dass es seine aktive Netzschnittstelle wechselt oder neue Locator-Werte auf einer bestehenden Schnittstelle zugewiesen bekommt. Das Kernel-Modul überwacht daher alle verfügbaren Netzschnittstellen und führt das Discovery-Verfahren aus, sobald eine Änderung an der Netzschnittstelle stattfindet. In diesem Zug werden auch die Mapping-Einträge aktualisiert.

Benachrichtigung der Kommunikationspartner Im Falle eines Locator-Wechsels werden nach der Aktualisierung der Mapping-Einträge alle Kommunikationspartner, die im UID-Cache gespeichert sind, darüber informiert. Dies geschieht durch Setzen des `LOC_UPD`-Bits im HiiMap-Header. Die Kommunikationspartner übernehmen daraufhin die neuen Locator-Werte in ihren UID-Cache.

6.2.2.5. Konfiguration des Moduls

Das Kernel-Modul kann sowohl die Aufgaben eines Endgerätes als auch die eines RPR übernehmen. Der jeweilige Betriebsmodus wird beim Start mittels verschiedener Parameter festgelegt. Diese Parameter werden nach dem Start im /proc/net-Verzeichnis des Dateisystems aufgelistet und können geändert werden, indem neue Werte an die entsprechende Position geschrieben werden. Folgende Parameter existieren:

- *global_uid* setzt den UID des Geräts. Dieser Wert kann ausgehend vom Gerätenamen und dem UID-Schema durch ein Start-Skript generiert werden.
- *global_router*. Ist dieser Wert gesetzt, leitet das Modul Pakete, die nicht für dieses Gerät bestimmt sind, weiter. Dazu wird die Information aus dem HiiMap-Header ausgewertet und ein neues IP-Paket mit gekapseltem HiiMap-Paket erzeugt, das entweder an den nächsten RPR oder an das Ziel-Gerät gesendet wird. Wird der Betriebsmodus von Endgerät zu RPR gewechselt, muss das Modul neu gestartet werden.
- *buffer_mappinglta* speichert die IP-Adresse des Mapping-Systems. Im Betriebsmodus RPR muss dieser Wert manuell gesetzt werden, damit er über das Discovery-Verfahren verteilt werden kann. Im Betriebsmodus Endgerät wird dieser Wert nicht angegeben, sondern automatisch über das Discovery-Verfahren ermittelt.
- *global_uid_presentation* speichert GGUID bzw. IPv4-Adresse des RPR und muss in diesem Betriebsmodus manuell gesetzt werden. Für Endgeräte wird dieser Wert ebenfalls über das Discovery-Verfahren ermittelt.
- *ignore_interface* erlaubt es, spezielle Netzschnittstellen von HiiMap auszuschließen. Diese Schnittstellen werden damit auch nicht auf Änderungen hin überwacht.

6.3. Demo-Anwendung und Messungen

Der Test des Kernel-Moduls erfolgt mit der in Abbildung 6.2 dargestellten Topologie. Diese wurde mit dem Topologie-Generator ToMaTo auf der G-Lab Experimentalplattform erzeugt.

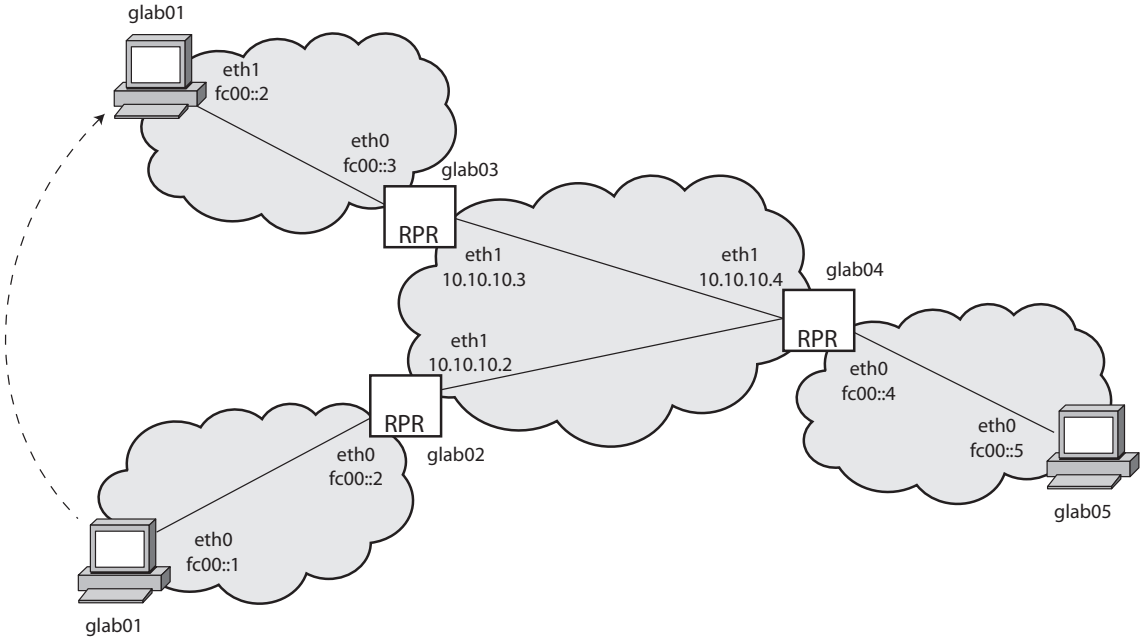


Abbildung 6.2.: Aufbau der Test-Topologie für das Kernel-Modul

Die Geräte glab02, glab03 und glab04 sind als RPR konfiguriert, wobei die Netzschnittstellen eth0 jeweils an ein lokales Netz angebunden ist. Diese lokalen Netze stellen im Experiment jeweils ein AS dar. Als Adressierungsschema für die LTA wird in den lokalen Netzen IPv6 verwendet. Über die Netzschnittstellen eth1 sind die RPRs jeweils miteinander verbunden. Dies stellt den Inter-Domain-Bereich dar. Als Adressierungsschema für die GGUID wird IPv4 verwendet. Die jeweils für die RPR vergebenen Adressen sind in Tabelle 6.1 aufgelistet.

Tabelle 6.1.: Zuteilung LTA und GGUID der RPRs

RPR	LTA (eth0 - IPv6)	GGUID (eth1 - IPv4)
glab02	fc00::2	10.10.10.2
glab03	fc00::3	10.10.10.3
glab04	fc00::4	10.10.10.4

Die Geräte glab01 und glab05 sind als normale Endgeräte konfiguriert, die ihre Konfiguration bezüglich GGUID und Mapping-LTA per Discovery-Mechanismus vom jeweiligen RPR beziehen. Die IPv6-Adresse, die als LTA dient, wird per DHCPv6 ebenfalls

vom jeweiligen RPR zugewiesen, jedoch unabhängig vom HiiMap Kernel-Modul. Die Ausgangskonfiguration der Endgeräte ist in Tabelle 6.2 aufgelistet.

Tabelle 6.2.: Ausgangskonfiguration der Endgeräte

Gerätename	UID	LTA	GGUID
glab01	014a43813c1c4ea1584fe60900000000	fc00::1	10.10.10.2
glab05	019b08b41c9199f3f916ab3500000000	fc00::5	10.10.10.4

Im folgenden kommunizieren die Geräte glab01 und glab05 miteinander. Dazu wurde eine Konsolenanwendung implementiert, mit der über einen HiiMap-Socket gegenseitig Textnachrichten ausgetauscht und Dateien versendet werden können. Zum Öffnen des Sockets wird lediglich der UID der Gegenstelle benötigt. Die Dienstkennung im Ext 2-Feld des UID für Textnachrichten ist 1, für den Dateiaustausch 2.

6.3.1. Mobilitätsunterstützung

Mit diesem Experiment soll die Mobilitätsunterstützung für den Fall Roaming von HiiMap überprüft werden. Dazu werden in zwei Durchläufen einmal Textnachrichten und einmal eine Datei zwischen beiden Endgeräten übertragen. Während der Übertragung wird Mobilität des Gerätes glab01 emuliert, indem dessen Netzschnittstelle eth0 deaktiviert wird und gleichzeitig die Netzschnittstelle eth1 aktiviert wird, die jedoch an einem anderen AS angeschlossen ist. Das Gerät „bewegt“ sich damit vom Zuständigkeitsbereich von RPR glab02 in den Bereich von RPR glab03.

Zur jeweiligen Übertragung wurde ein rudimentäres Stop-and-Go-Verfahren verwendet, welches jedes übertragene Datenpaket einzeln bestätigt. Ein Übertragungsfenster wie in TCP wird aufgrund der Komplexität nicht verwendet. Wird ein Paket innerhalb einer Sekunde nicht bestätigt, wird dieses bis zu 10 mal erneut versendet.

Textnachrichten Bei diesem Experiment tauschen zwei Personen an den jeweiligen Geräten Textnachrichten in Form eines Chats miteinander aus. Zu einem bestimmten Zeitpunkt wird Mobilität emuliert, die Textnachrichten vom Gerät glab05 erreichen jedoch ohne weiteres zutun das Gerät glab01, das den Locatorwechsel direkt an seinen Kommunikationsparter gemeldet hat. Aus der Sicht der Anwendung bleibt die Verbindung bestehen. Lediglich die Nachrichten, die im Moment des Umschaltens versendet wurden, müssen erneut an die neuen Locator-Werte übertragen werden. Für die Benutzer verläuft der Roamingvorgang völlig transparent ohne merkliche Verzögerung.

Dateitransfer Im zweiten Durchlauf wird eine 100 kByte große Datei von glab05 an glab01 übertragen und der Netzzungangspunkt von glab01 während der Übertragung wie im vorherigen Experiment einmal gewechselt. Es werden die Übertragungszeiten mit und ohne Roamingvorgang miteinander verglichen. Zusätzlich wurde im weiteren Verlauf des Experiments eine Paketverlustrate auf der Verbindung zwischen glab05 und dessen RPR glab04, emuliert. Damit wurde gezeigt, dass die Datei trotz Paketverlust vollständig

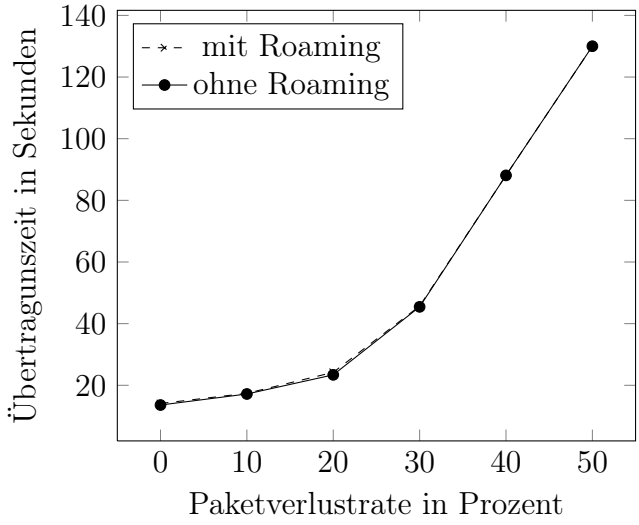


Abbildung 6.3.: Übertragungszeiten mit Paketverlusten

empfangen werden kann. Für die verschiedene Paketverlustraten wurden ebenfalls die Übertragungszeiten gemessen. Dabei erfolgte bei jeder Messung ein Roamingvorgang. Die Ergebnisse sind in Abbildung 6.3 dargestellt.

Es ist zu Erkennen, dass die Übertragungszeiten ausschließlich durch die Paketverlustrate erhöht werden. Die zusätzliche Übertragungszeit durch den Roamingvorgang ist dabei vernachlässigbar. Dies zeigt, dass das Kernel-Modul sehr schnell zwischen den aktiven Netzschnittstellen umschalten kann und die neuen Locatorwerte sofort übernommen werden. Die grundsätzlich relativ lange Übertragungszeit für die 100 kByte große Testdatei resultiert aus dem Stop-and-Go-Verfahren und einer durchschnittlichen gemessenen Paketumlaufzeit von 132 ms zwischen glab05 und glab01. Es kann damit nur alle 132 ms ein Paket gesendet werden, dessen Nutzdaten abzüglich der Packet-Header noch 1300 Byte beträgt. Dies führt im fehlerlosen Fall zu einer Übertragungsrate von etwa 80 kBit/s.

In diesem Experiment konnte gezeigt werden, dass die Implementierung des HiiMap Kernel-Moduls in der Lage ist, während einer Datenübertragung die aktive Netzschnittstelle fast verzögerungsfrei und für die Anwendung völlig transparent zu wechseln. Sowohl die Textnachrichten, als auch die Testdatei, wurden an der Gegenstelle jeweils vollständig und fehlerfrei empfangen.

6.3.2. Datendurchsatz

In einem weiteren Experiment wurde der maximal mögliche Datendurchsatz des HiiMap Kernel-Moduls im Datagrammbetrieb mit der Kombination UDP und IPv6 verglichen. Dazu wurden 1 GByte Daten aus dem Arbeitsspeicher über eine fehlerfreie Verbindung ohne Roamingvorgang von glab05 an glab01 übermittelt. Sowohl HiiMap als auch UDP über IPv6 sind in der Lage, die gesamte verfügbare Datenrate von 1 GBit/s fast vollständig auszulasten. Tabelle 6.3 zeigt die jeweils gemessene Netto- und Bruttodatenrate.

Während die Bruttodatenrate bei beiden Protokollen fast gleichauf liegt, so ist die

Tabelle 6.3.: Vergleich Nettodatenrate HiiMap mit UDP

	Nettodatenrate	Bruttodatenrate	Overhead
HiiMap Datagramm	855,3 MBit/s	925,1 MBit/s	8%
UDP mit IPv6	896,4 MBit/s	924,6 MBit/s	3,2%

Nettodatenrate bei HiiMap etwas geringer. Dies ist auf den größeren Packet-Header von HiiMap zurückzuführen. Im Vergleich zu UDP mit IPv6, die zusammen eine Header-Größe von 48 Byte ausmachen, benötigt HiiMap mit IPv6 120 Byte. HiiMap mit IPv6 besitzt damit einen Overhead von 8% verglichen zu 3,2% bei UDP mit IPv6 und einer gesamten Paketgröße von 1500 Byte.

Da die aktuelle Implementierung des HiiMap Kernel-Moduls in der Lage ist, eine 1 GBit/s Anbindung vollständig auszulasten, wurde darüber hinaus versucht, den maximal möglichen Datendurchsatz mit diesem Kernel-Modul zu bestimmen. Dieser ist jedoch stark von der verwendeten Hardware abhängig. Um einen Vergleich ziehen zu können, wurden die Leistungsdaten ebenfalls mit UDP verglichen.

Der Versuchsaufbau besteht in diesem Experiment aus lediglich einem Gerät, das Daten an sein eigenes Null-Device[1] schreibt und daher unabhängig von der Datenrate der Netzanbindung ist. Es ist auf diese Weise möglich, den Maximaldurchsatz zu bestimmen. Auf dem entsprechenden Gerät wurde eine Route erzeugt, die auf das Null-Device zeigt. Zusätzlich wurde im Mapping-System ein UID angelegt, die den gleichen GGUID wie das Testgerät besitzt mit einer zur Null-Device-Route passenden LTA. Gemäß Definition werden Daten an ein Gerät im gleichen Subnetz direkt per LTA adressiert, auf diese Weise können die Daten lokal an das Null-Device gesendet werden.

Auf der verwendeten virtuellen Hardware in ToMaTo stand ein Kern eines Intel Xeon Quad Core L5420 mit 2,5 GHz Taktfrequenz zur Verfügung. Es wurde eine 10 GByte Testdatei aus dem Arbeitsspeicher für das Experiment verwendet. Der jeweils für UDP mit IPv6 und HiiMap mit IPv6 erreichte Bruttodurchsatz ist in Tabelle 6.4 aufgezeigt.

Tabelle 6.4.: Vergleich Maximaldurchsatz HiiMap mit UDP

	Bruttodurchsatz
HiiMap Datagramm	3,604 GBit/s
UDP mit IPv6	10,850 GBit/s

Die Messung zeigt, dass HiiMap auf der verwendeten Hardware in der Lage ist, einen maximalen Datendurchsatz von 3,6 GBit/s zu verarbeiten. Ausgehend davon, dass UDP aufgrund der ausgereiften Implementierung das optimale Ergebnis liefert, kann das HiiMap Kernel-Modul in der aktuellen Version etwa ein Drittel des maximal möglichen Durchsatzes erreichen, der jedoch hardwareabhängig ist. Durch weitere Verbesserungen in der Implementierung kann dieser weiter verbessert werden.

[1] Das Null-Device bezeichnet eine virtuelle Gerätedatei, die alle Daten, die dorthin geschrieben werden, ohne weitere Bearbeitung verwirft.

6.4. Zusammenfassung

In diesem Kapitel wurde eine Implementierung von HiiMap vorgestellt, welche die Trennung von Locator und Identifier umsetzt und die Kommunikation zwischen Geräten unterstützt. Die Implementierung wurde als Kernel-Modul für Linux 2.6 umgesetzt, womit eine einfache und gegenüber IP nahezu unveränderte Schnittstelle zu den Applikationen angeboten wird. Im Gegensatz zu IP wird eine Datenübertragung aber an einen UID und nicht an eine IP gebunden.

HiiMap verwendet für den Prototyp eine Tunneling-Lösung, um die Datenpakete über das Internet zu übertragen. Dazu kommt im Inter-Domain-Bereich IPv4 und im Intra-Domain-Bereich IPv6 zum Einsatz. RPR an den Netzübergängen setzen die Pakete auf das jeweils andere Vermittlungsprotokoll um.

Das implementierte Kernel-Modul wurde in der G-Lab Experimentalplattform unter Zuhilfenahme des ToMaTo-Topologiewerkzeugs getestet. Dabei erfolgten Datenübertragungen rein anhand des UID, sowie eines einfachen Stop-and-Go-Verfahrens zur Erkennung von Paketverlusten. Die Mobilitätsunterstützung wurde durch das Wechseln der aktiven Netzschnittstelle überprüft. Ebenso erfolgten Übertragungsmessungen mit Paketverlusten. Die Daten konnten jeweils fehlerfrei empfangen werden. Die aktuelle Version der Implementierung ist in der Lage, eine physikalische Verbindung von 1 Gbit/s voll auszulasten. Zusätzlich wurde der maximal mögliche Datendurchsatz auf der verwendeten Hardware ermittelt, den die vorliegende Implementierung des Kernel-Moduls verarbeiten kann. Dieser entspricht 33% des Maximaldurchsatzes von UDP.

7. Zusammenfassung

Hervorgerufen durch das zunehmende Wachstum des Internets ist die Skalierbarkeit der aktuellen Internet-Architektur nicht mehr gegeben. Ursprünglich entworfen für wenige Supercomputer, sind heute Milliarden verschiedenster Geräte am Internet angeschlossen. Viele davon sind mobile Endgeräte wie Smartphones, Notebooks oder Tablets. Mobilität in Bezug auf die unterbrechungsfreie Kommunikation trotz Wechsel des Netzzugangspunkts wird aktuell nur mangelhaft unterstützt. Ebenso reichen die aktuell verfügbaren IP-Adressen ohne zusätzliche Hilfsmittel wie NAT bereits nicht mehr aus, um alle am Internet angeschlossenen Geräte anzusprechen. Zusätzlich hat sich durch den riesigen Benutzerkreis auch die Art und Weise verändert, wie das Internet verwendet wird. So steht heute die Beschaffung und der Austausch von Information sowie die persönliche Kommunikation im Vordergrund gegenüber der klassischen Geräte-zu-Geräte-Kommunikation.

In der vorliegenden Arbeit werden zunächst die Schwächen und Probleme der aktuellen Architektur ausführlich analysiert und bereits existierende Lösungsansätze aufgezeigt. Aus der Forderung, die Internet-Architektur unter der Berücksichtigung der geänderten Anforderungen von Grund auf neu zu entwerfen, wird eine neue Internet-Architektur mit dem Namen HiiMap vorgestellt. HiiMap basiert auf dem Konzept der Trennung von Locator und Identifier und verwendet für das Routing einen zweigeteilten Locator. Damit ist es für ein höchst heterogenes Kommunikationsumfeld geeignet. Zusätzlich verwendet HiiMap einen sich niemals ändernden Identifier, der als UID bezeichnet wird. Dank dieser beiden Eigenschaften bietet HiiMap eine Lösung für die größten Probleme und Schwächen der aktuellen Internet-Architektur, wie die Adressknappheit, die großen Routingtabellen sowie die mangelnde Mobilitätsunterstützung. Für das Mapping-System, das als integraler Bestandteil einer jeden Architektur benötigt wird, werden zwei Ansätze vorgestellt, die auf der Trennung von Locator und Identifier basieren. Während das eine System besonders auf Leistungsfähigkeit abzielt, berücksichtigt das andere Vertrauensbeziehungen zwischen den Betreibern. Für den UID wird ein Namensschema ausgearbeitet, das es ermöglicht, den UID aus einem Klartextnamen selbst zu erzeugen.

In Kapitel 4 wird HiiMap zu einem vollständigen inhaltsorientierten Netz erweitert. Dies beinhaltet ein Schema zum Beschreiben und Benennen von Inhaltsobjekten und Personen. Das bereits vorhandene Namensschema wird dazu erweitert. Zusätzlich wird der Mobilitätsbegriff auf Inhalte ausgeweitet. Zusammen mit dem neu vorgestellten Algorithmus zur Inhaltsmobilität können sich Inhaltsobjekte frei im Netz bewegen und werden immer in der Nähe der Benutzer abgespeichert, welche an diesen Objekten interessiert sind. Diese Verteilung von Inhalten durch den Algorithmus erfolgt dabei ressourceneffizient, so dass die Kosten für die Inhaltsbereitstellung aus der Sicht des Netzes minimiert werden. Mithilfe von Simulationen werden die erwarteten Ergebnisse bestätigt. Die Gesamtkosten sind dabei immer geringer oder gleich gegenüber der Bereitstellung von Inhalten mit nur einem Server mit oder ohne Caching. Die Erweiterung zum inhaltsorien-

tierten Netz wird durch den vorgestellten hybriden Speicherdienst vervollständigt. Dieser greift je nach Popularität eines Inhaltsobjekts auf ein P2P-System, oder auf dedizierte Speicherserver, die den Algorithmus zur Inhaltsmobilität verwenden, zurück.

Ein wichtiger Punkt bei jedem Konzept, das sich mit einer neuen Internet-Architektur beschäftigt, sind Methoden zur Migration. Die Erweiterung zum inhaltsorientierten Netz wird dabei zunächst außer Acht gelassen, da zuerst die Kommunikation zwischen den Geräten möglich sein muss. Dazu werden Koexistenzmethoden von HiiMap und IP vorgestellt, die auf den Mechanismen Tunneling und Translation aufbauen.

Zusätzlich sind essentielle Komponenten von HiiMap als Kernel-Modul für Linux 2.6-Systeme auf der Experimentalplattform G-Lab implementiert. Die Machbarkeit und Vorteile dieser Architektur sind durch Versuche und Messungen belegt.

Zusammenfassend wird mit HiiMap ein vielversprechender Ansatz für eine Internet-Architektur der Zukunft vorgestellt, der als Grundlage für eine zukünftige Internet-Architektur dienen kann. Durch das Konzept der Trennung von Locator und Identifier können viele akute Probleme der heutigen Architektur erfolgreich gelöst werden. Weitere Forschungsarbeit ist jedoch beim Erstellen von Sicherheitskonzepten nötig. Zwar bietet HiiMap eine integrierte PKI, gezielte Anwendungsfälle sind aber noch zu untersuchen. Ebenso bezieht sich das Konzept der Inhaltsmobilität ausschließlich auf statische Inhalte, die nicht zur Laufzeit erzeugt werden. Die Erweiterung dieses Konzepts auf dynamisch generierte Inhalte bildet zusätzlich eine Basis für zukünftige Arbeiten.

Es existieren viele Arbeiten, die sich mit einer zukünftigen Internet-Architektur beschäftigen. Die vorliegende Arbeit betrachtet aber nicht gezielte Probleme, sondern bietet einen Ansatz für eine ganzheitliche Lösung, damit das Internet auch weiterhin eine tragende Rolle in unserer Gesellschaft spielen kann und den zukünftigen Anforderungen gewachsen ist.

A. XML-Beschreibungen

A.1. Informationsobjekt für Inhalte

```
<abstract IO>
       <*UID>37394583145301348500000000000</*UID>
       <title>Zeitungsartikel</title>
       <summary>Hier steht eine Zusammenfassung des Artikels</summary>
       <specific IO>
              <*UID>37394583145301348532490000000</*UID>
              <mediatype>text</mediatype>
              <representation>
                     <*UID>37394583145301348532496549000</*UID>
                     <format>text/plain-UTF8</format>
                     <version>
                            <UID>37394583145301348532496549001</UID>
                            <number>1</number>
                            <link>
                                   <locator>5436.85325</locator>
                            </link>
                     </version>
                     <version>
                            <UID>37394583145301348532496549002</UID>
                            <number>2</number>
                            <link>
                                   <locator>5436.92342</locator>
                            </link>
                     </version>
              </representation>
              <representation>
                     <UID>37394583145301348532490486000</UID>
                     <format>RichText</format>
                     <link>
                            <locator>5436.92342</locator>
                     </link>
              </representation>
       </specific IO>
       <specific IO>
              <*UID>37394583145301348554390000000</*UID>
              <mediatype>picture</mediatype>
              <representation>
                     <UID>37394583145301348554394832000</UID>
                     <format>jpg</format>
                     <resolution>4742x3161</resolution>
                     <filesize>5432 kbyte</filesize>
                     <link>
                            <locator>5436.54536</locator>
                     </link>
              </representation>
              <representation>
                     <UID>37394583145301348554399125000</UID>
                     <format>gif</format>
                     <resolution>640x480</resolution>
                     <filesize>76 kbyte</filesize>
                     <link>
                            <locator>5436.54356</locator>
                            <locator>5436.97636</locator>
                     </link>
              </representation>
       </specific IO>
```

```
        <specific IO>
                <*UID>37394583145301348554390000000</*UID>
                <mediatype>picture</mediatype>
                <representation>
                        <UID>373945831453013485543948320000</UID>
                        <format>jpg</format>
                        <resolution>4742x3161</resolution>
                        <filesize>5432 kbyte</filesize>
                        <link>
                                <locator>5436.54536</locator>
                        </link>
                </representation>
                <representation>
                        <UID>373945831453013485543991250000</UID>
                        <format>gif</format>
                        <resolution>640x480</resolution>
                        <filesize>76 kbyte</filesize>
                        <link>
                                <locator>5436.54356</locator>
                                <locator>5436.97636</locator>
                        </link>
                </representation>
        </specific IO>
        <specific IO>
                <*UID>37394583145301348595340000000</*UID>
                <mediatype>picture/collection</mediatype>
                <representation>
                        <description>high resolution picture gallery</description>
                        <link>
                                <IO>"UID of next IO"</IO>
                        </link>
                </representation>
        </specific IO>
        <specific IO>
                <*UID>37394583145301348545790000000</*UID>
                <mediatype>video</mediatype>
                <representation>
                        <UID>739458314530134854579432600 0</UID>
                        <format>MPEG4</format>
                        <resolution>1920x1080</resolution>
                        <vcodec>h264</vcodec>
                        <acodec>AC3</acodec>
                        <bitrate>8 MBit/s</bitrate>
                        <link>
                                <locator>5436.12667</locator>
                        </link>
                </representation>
                <representation>
                        <UID>373945831453013485457956330 00</UID>
                        <format>AVI</format>
                        <resolution>320x240</resolution>
                        <vcodec>mpg1</vcodec>
                        <acodec>mp3</acodec>
                        <bitrate>300 kBit/s</bitrate>
                        <link>
                                <locator>5436.12667</locator>
                                <locator>5436.34233</locator>
                        </link>
                </representation>
        </specific IO>
</abstract IO>
```

A.2. Informationsprofil für Personen

```
<personalIP>
       <*UID>59245024754969281653760000000</*UID>
       <firstname>Hans</firstname>
       <lastname>Maier</lastname>
       <description>Arcisstraße 21, 80333 München</description>
       <signature>4452059698334</signature>
       <pubkey>al74ulqlg8a84uglia8fh</pubkey>
       <channel>
              <UID>59245024754969281653760000001</UID>
              <type>local device</type>
                     <link>
                            <locator>6357.3945</locator>
                            <locator>5532.1094</locator>
                     <link>
       </channel>
       <channel>
              <UID>59245024754969281653760000002</UID>
              <type>Email</type>
                     <link>
                            <locator>5532.1094</locator>
                     </link>
       </channel>
       <channel>
              <UID>59245024754969281653760000003</UID>
              <type>Instant Messenger</type>
                     <link>
                            <locator>2462.5453</locator>
                     </link>
       </channel>
       <channel>
              <UID>59245024754969281653760000004</UID>
              <type>VoIP</type>
                     <link>
                            <locator>3642.9982</locator>
                     </link>
       </channel>
       <channel>
              <UID>59245024754969281653760000005</UID>
              <type>Email</type>
                     <link>
                            <locator>3642.1695</locator>
                     </link>
       </channel>
</personalIP>
```

Abbildungsverzeichnis

Tabellenverzeichnis

Abkürzungsverzeichnis

AS	Autonomes System
AH	Authentication Header
BGP	Border Gateway Protocol
BGR	Border Gateway Router
CCN	Content Centric Network
CDN	Content Distribution Network
CIDR	Classless Inter-Domain Routing
CoA	Care-of-Address
CNR	Core Network Router
CRIO	Core Router Integrated Overlay
DHT	Distributed Hash Table
DFZ	Default Free Zone
DMB	Direct Map Database
DNS	Domain Name System
DNSMS	Domain Name System based Mapping System
DO	Datenobjekt
DOI	Digital Object Identifier
EID	Endpoint Identifier
ESP	Encapsulating Security Payload
ETR	Egress Tunnel Router
FA	Foreign Agent
FIB	Forwarding Information Base
FQDN	Full Qualified Domain Name

GR	Globale Registrierungsstelle
GGUID	Global Gateway Unique Identifier
HA	Home Agent
HI	Host Identifier
HiiMap	Hierarchical Internet Mapping Architecture
HIP	Host Identity Protocol
HIT	Host Identity Tag
HoA	Home Address
IANA	Internet Assigned Numbers Authority
IGRP	Interior Gateway Routing Protocol
IO	Informationsobjekt
IPsec	Internet Protocol Security
IS-IS	Intermediate System to Intermediate System Protocol
ITR	Ingress Tunnel Router
LB	Load Balancer
LFU	Least Frequently Used
LISP	Locator/Identifier Separation Protocol
LNR	Local Network Router
LRU	Least Recently Used
LTA	Local Temporary Address
GMPLS	Generalized Multi Protocol Label Switching
MRFO	Map-Request Forwarding Overlay
NAT	Network Address Translation
NetInf	Network of Information
OSPF	Open Shortest Path First
P2P	Peer-to-Peer
PIT	Pending Interest Table

PKI	Public-Key Infrastruktur
RIP	Routing Information Protocol
RLOC	Routing Locator
RP	Regionales Präfix
RPR	Re-Pack Router
SGW	Super-Gateway
SOA	Service Oriented Architecture
TCP	Transmission Control Protocol
TLD	Top Level Domain
TLMS	Two-Level Mapping System
TLS	Transport Layer Security
TR	Tunnelrouter
TTL	Time-To-Live
UDP	User Datagram Protocol
UID	Unique Identifier
URI	Uniform Ressource Identifier
VoIP	Voice over IP
WWW	World Wide Web

Literaturverzeichnis

Publikationen des Autors

[FS09] W. Fritz und C. Spleiß, *Distributed Lookup-Service to Establish VPN-Connections in Next Generation Internet*, In Proceedings of the 1st IEEE Student Conference Germany (Hamburg), Mai 2009, Seiten 37–40.

[HSK+09] O. Hanka, C. Spleiß, G. Kunzmann, J. Eberspächer und A. Bauer, *HiiMap: Hierarchical Internet Mapping Architecture*, In Proceedings of the 1st International Conference on Future Information Networks, ICFIN (Peking, China), IEEE, Oktober 2009, Seiten 17–24.

[HSKE09] O. Hanka, C. Spleiß, G. Kunzmann und J. Eberspächer, *A novel DHT-based Network Architecture for the Next Generation Internet*, In Proceedings of the 8th International Conference on Networks (Cancun, Mexiko), März 2009, Seiten 332–341.

[RWH+09] M. Röhricht, H. Wippel, O. Hanka, C. Spleiß, B. Reuther, A. Siddiqui und D. Schwerdel, *Towards a Service-Oriented Future Internet Architecture*, Whitepaper, G-Lab Phase 1 ASP1 & ASP6, 2009.

[SK07] C. Spleiß und G. Kunzmann, *Decentralized Supplementary Services for Voice-over-IP Telephony*, Dependable and Adaptable Networks and Services, Lecture Notes in Computer Science, Band 4606, Springer Berlin / Heidelberg, 2007, Seiten 62–69.

[SK11a] C. Spleiß und G. Kunzmann, *A Naming Scheme for Identifiers in a Locator/Identifier-Split Internet Architecture*, In Proceedings of the 10th International Conference on Networks (Sint Maarten, Niederlande), Januar 2011, Seiten 57–62.

[SK11b] C. Spleiß und G. Kunzmann, *Naming, Assigning and Registering Identifiers in a Locator/Identifier-Split Internet Architecture*, International Journal On Advances in Internet Technology Band 4 (2011), Nr. 3 und 4.

[SS11a] C. Spleiß und Q. Scheitle, *Pushing HiiMap into the Kernel: An easy-to-use Implementation of a real Locator/Identifier-Split*, Forschungsbericht, Technische Universität München, Institute of Communication Networks, LKN-TR-6, Januar 2011.

[SS11b] C. Spleiß und C. Schuwerk, *An Algorithm for Content Mobility in a Future Internet Architecture*, In Proceedings of the International Conference on Communications, ICC, Workshop (Kyoto, Japan), IEEE, Juni 2011.

[WHSM10] H. Wippel, O. Hanka, C. Spleiß und D. Martin, *Evaluation of Future Network Architectures and Services in the G-Lab Testbed*, In Proceedings of the 6th International ICST Conference on Testbeds and Research Infrastructures for the Development of Network & Communities, LNICST 46 (Berlin), Mai 2010, Seiten 587–589.

Allgemeine Publikationen

[AB08] J. Arkko und S. Bradner, *IANA Allocation Guidelines for the Protocol Field*, IETF Request for Comments 5237, 2008.

[ADM+08] B. Ahlgren, M. D'Ambrosio, M. Marchisio, I. Marsh, C. Dannewitz, B. Ohlman, K. Pentikousis, O. Strandberg, R. Rembarz und V. Vercellone, *Design Considerations for a Network of Information*, In Proceedings of the 2008 ACM CoNEXT Conference (Madrid), ACM, Dezember 2008, Seite 66 ff.

[BCSW00] R. Braden, D. Clark, S. Shenker und J. Wroclawski, *Developing a Next-Generation Internet Architecture*, Whitepaper, DARPA, Juli 2000.

[BDF+09] P. Borgnat, G. Dewaele, K. Fukuda, P. Abry und K. Cho, *Seven years and one day: Sketching the Evolution of Internet Traffic*, In Proceedings of INFOCOM 2009, IEEE, 2009, Seiten 711–719.

[BH07] A. Badach und E. Hoffmann, *Technik der IP-Netze: Funktionsweise, Protokolle und Dienste*, 2. Auflage, Carl Hanser Verlag, München, 2007.

[BMM+08] N. Basher, A. Mahanti, A. Mahanti, C. Williamson und M. Arlitt, *A Comparative Analysis of Web and Peer-to-Peer Traffic*, In Proceeding of the 17th International Conference on World Wide Web, ACM, 2008, Seiten 287–296.

[CANK03] J. Cao, M. Andersson, C. Nyberg und M. Kihl, *Web Server Performance Modeling Using an M/G/1/K* ps Queue*, In Proceedings of the 10th International Conference on Telecommunications, ICT, Band 2, IEEE, 2003, Seiten 1501–1506.

[CFSD90] J. Case, M. Fedor, M. Schoffstall und J. Darvin, *A Simple Network Management Protocol (SNMP)*, IETF Request for Comments 1157, Mai 1990.

[CG85] B. Croft und J. Gilmore, *Bootstrap Protocol (BOOTP)*, IETF Request for Comments 951, September 1985.

[CIS11] *Cisco Visual Networking Index, Forecast and Methodology, 2010–2015*, Whitepaper, Cisco Systems, Inc., 2011.

[Cla88] D. Clark, *The Design Philosophy of the DARPA Internet Protocols*, ACM SIGCOMM Computer Communication Review Band 18 (1988), Nr. 4, Seiten 106–114, ACM.

[CMU+10] L. Cittadini, W. Muhlbauer, S. Uhlig, R. Bush, P. François und O. Maennel, *Evolution of Internet Address Space Deaggregation: Myths and Reality*, IEEE Journal on Selected Areas in Communications Band 28 (2010), Nr. 8, Seiten 1238–1249, IEEE.

[Cro82] D. Crooker, *Standard for the Format of ARPA Internet Text Messages*, IETF Request for Comments 822, August 1982.

[DA98] T. Dierks und C. Allen, *The TLS Protocol Version 1.0*, IETF Request for Comments 2246, Januar 1999 1998.

[Dan09] C. Dannewitz, *NetInf: An Information-Centric Design for the Future Internet*, In Proceedings of 3rd GI/ITG KuVS Workshop on The Future Internet (München), Mai 2009.

[DH76] W. Diffie und M. Hellman, *New Directions in Cryptography*, IEEE Transactions on Information Theory Band 22 (1976), Nr. 6, Seiten 644–654, IEEE.

[DH98] S. Deering und R. Hinden, *Internet Protocoll Version 6 (IPv6) Specification*, IETF Request for Comments 2460, Dezember 1998.

[Dow05] A.B. Downey, *Lognormal and Pareto Distributions in the Internet*, Computer Communications Band 28 (2005), Nr. 7, Seiten 790–801, Elsevier.

[Dro97] R. Droms, *Dynamic Host Configuration Protocol*, IETF Request for Comments 2131, März 1997.

[EF94] K. Egevang und P. Francis, *The IP Network Address Translator (NAT)*, IETF Request for Comments 1631, Mai 1994.

[FBR03] N. Feamster, J. Borkenhagen und J. Rexford, *Guidelines for Interdomain Traffic Engineering*, ACM SIGCOMM Computer Communication Review Band 33 (2003), Nr. 5, Seiten 19–30, ACM.

[FCM+09] A. Feldmann, L. Cittadini, W. Mühlbauer, R. Bush und O. Maennel, *HAIR: Hierarchical Architecture for Internet Routing*, In Proceedings of the 2009 Workshop on Re-architecting the Internet, ACM, 2009, Seiten 43–48.

[Fel07] A. Feldmann, *Internet clean-slate design: What and why?*, ACM SIGCOMM Computer Communication Review Band 37 (2007), Nr. 3, Seiten 59–64, ACM.

[Han06] M. Handley, *Why the Internet only just works*, BT Technology Journal Band 24 (2006), Nr. 3, Seiten 119–129, Springer.

[Han10] O. Hanka, *A Privacy Service for Locator/Identifier-Split Architectures Based on Mobile IP Mechanisms*, Proceedings of the 2nd International Conference on Advances in Future Internet (AFIN) (Venedig), IEEE, 2010, Seiten 6–10.

[HCW97] S. Harding, W. Croft und C. Weir, *Probabilistic Retrieval of OCR Degraded Text Using N-grams*, Research and Advanced Technology for Digital Libraries (1997), Seiten 345–359, Springer.

[HEP+11] O. Hanka, M. Eichhorn, M. Pfannenstein, J. Eberspächer und E. Steinbach, *A Distributed Public Key Infrastructure based on Threshold Cryptography for the HiiMap Next Generation Internet Architecture*, Future Internet Band 3 (2011), Nr. 1, Seiten 14–30.

[HF11] O. Hanka und W. Fritz, *An Holistic Approach to Public/Private-Key Based Security in Locator/Identifier-split Architectures*, International Journal On Advances in Security Band 3 (2011), Nr. 3, Seiten 135–145.

[HHH+02] M. Harren, J. Hellerstein, R. Huebsch, B. Loo, S. Shenker und I. Stoica, *Complex Queries in DHT-based Peer-to-Peer Networks*, Peer-to-Peer Systems (2002), Seiten 242–250, Springer.

[HKSC04] K. Hosanagar, R. Krishnan, M. Smith und J. Chuang, *Optimal Pricing of Content Delivery Network (CDN) Services*, In Proceedings of the 37th Annual Hawaii International Conference on System Sciences, IEEE Computer Society, Januar 2004.

[Hui94] C. Huitema, *The H Ratio for Address Assignment Efficiency*, IETF Request for Comments 1715, November 1994.

[Hus99] G. Huston, *Web Caching*, The Internet Protocol Journal Band 2 (1999), Nr. 3, Seiten 2–20, Cisco Systems, Inc.

[ITU09] ITU, *Draft Recommendation ITU-T Y.2015: General Requirements for ID/Locator Separation in NGN*, International Telecommunication Union, 2009.

[JPA04] D. Johnson, C. Perkins und J. Arkko, *Mobility Support in IPv6*, IETF Request for Comments 3775, 2004.

[JSBM01] J. Jung, E. Sit, H. Balakrishnan und R. Morris, *DNS Performance and the Effectiveness of Caching*, In Proceedings of the 1st ACM SIGCOMM Workshop on Internet Measurement, ACM, 2001, Seiten 153–167.

[JST+09] V. Jacobson, D.K. Smetters, J.D. Thornton, M.F. Plass, N.H. Briggs und R.L. Braynard, *Networking Named Content*, In Proceedings of the 5th International Conference on Emerging Networking Experiments and Technologies (Rom), CoNEXT '09, ACM, Dezember 2009, Seiten 1–12.

[KA98] S. Kent und R. Atkinson, *Security Architecture for the Internet Protocol*, IETF Request for Comments 2401, November 1998.

[KCC+07] T. Koponen, M. Chawla, B.G. Chun, A. Ermolinskiy, K.H. Kim, S. Shenker und I. Stoica, *A Data-Oriented (and Beyond) Network Architecture*, ACM SIGCOMM Computer Communication Review Band 37 (2007), Nr. 4, Seiten 181–192, ACM.

[KI10] V.P. Kafle und M. Inoue, *HIMALIS: Heterogeneity Inclusion and Mobility Adaptation through Locator ID Separation in New Generation Network*, IEICE TRANSACTIONS on Communications Band 93 (2010), Nr. 3, Seiten 478–489.

[Kle61] L. Kleinrock, *Information Flow in Large Communication Nets*, RLE Quarterly Progress Report Band 1 (1961).

[KRR02] J. Kangasharju, J. Roberts und K.W. Ross, *Object Replication Strategies in Content Distribution Networks*, Computer Communications Band 25 (2002), Nr. 4, Seiten 376–383, Elsevier.

[Kun09] G. Kunzmann, *Performance Analysis and Optimized Operation of Structured Overlay Networks*, Dissertation, Technische Universität München, München, 2009.

[LC62] J.C.R. Licklider und W.E. Clark, *On-Line Man-Computer Communication*, Proceedings of the Spring Joint Computer Conference, ACM, Mai 1962, Seiten 113–128.

[LCC+97] B.M. Leiner, V.G. Cerf, D.D. Clark, R.E. Kahn, L. Kleinrock, D.C. Lynch, J. Postel, L.G. Roberts und S. Wolff, *A Brief History of the Internet*, Communications of the ACM Band 40 (1997), Nr. 2, Seiten 102–108.

[LLFS05] J. Leguay, M. Latapy, T. Friedman und K. Salamatian, *Describing and Simulating Internet Routes*, NETWORKING 2005. Networking Technologies, Services, and Protocols; Performance of Computer and Communication Networks; Mobile and Wireless Communications Systems (2005), Seiten 659–670, Springer.

[LQZ09] H. Luo, Y. Qin und H. Zhang, *A DHT-based Identifier-to-Locator Mapping Approach for a Scalable Internet*, IEEE Transactions on Parallel and Distributed Systems Band 20 (2009), Nr. 10, Seiten 1790–1802, IEEE Computer Society.

[LST07] B. Libert, J. Spector und D. Tapscott, *We are Smarter Than Me: How to Unleash the Power of Crowds in Your Business*, 1. Auflage, Prentice Hall, September 2007.

[MA06] L.R. Monnerat und C.L. Amorim, *D1HT: A Distributed One Hop Hash Table*, Parallel and Distributed Processing Symposium, 2006. IPDPS 2006. 20th International, IEEE, 2006, Seite 10 ff.

[MBS+09] T. Moyer, K. Butler, J. Schiffman, P. McDaniel und T. Jaeger, *Scalable Web Content Attestation*, Annual Computer Security Applications Conference, ACSAC (Honolulu), Dezember 2009, Seiten 95–104.

[Mey08] D. Meyer, *The Locator Identifier Separation Protocol (LISP)*, The Internet Protocol Journal Band 11 (2008), Nr. 1, Seiten 23–36, Cisco Systems Inc.

[MHH10] M. Menth, M. Hartmann und M. Höfling, *Mapping Systems for Loc/ID Split Internet Routing*, Forschungsbericht, Universität Würzburg, Lehrstuhl für Informatik III, TR-472, 2010.

[MHK10] M. Menth, M. Hartmann und D. Klein, *Global Locator, Local Locator, and Identifier Split (GLI-Split)*, Forschungsbericht, University of Würzburg, Institute of Computer Science, 2010.

[MHTGK09] M. Menth, M. Hartmann, P. Tran-Gia und D. Klein, *Future Internet Routing: Motivation and Design Issues*, it-Information Technology Band 50 (2009), Nr. 6, Seiten 358–366, Oldenbourg Wissenschaftsverlag GmbH München, Germany.

[MI08] L. Mathy und L. Iannone, *LISP-DHT: Towards a DHT to map Identifiers onto Locators*, In Proceedings of the 2008 ACM CoNEXT Conference (Madrid), ACM, Dezember 2008, Seite 61.

[MJNH08] R. Moskowitz, P. Jokela, P. Nikander und T. Henderson, *Host Identity Protocol*, IETF Request for Comments 5021, 2008.

[Moc87a] P. Mockapetris, *Domain Names – Concepts and Facilities*, IETF Request for Comments 1034, November 1987.

[Moc87b] P. Mockapetris, *Domain Names – Implementation and Specification*, IETF Request for Comments 1035, November 1987.

[Moo02] T. Moors, *A critical review of "End-to-end Arguments in System Design"*, In Proceedings of the International Conference on Communications, ICC (New York), IEEE, 2002, Seiten 1214–1219.

[MP85] J. Mogul und J. Postel, *Internet Standard Subnetting Procedure*, IETF Request for Comments 950, August 1985.

[MWE00] A. Mahanti, C. Williamson und D. Eager, *Traffic Analysis of a Web Proxy Caching Hierarchy*, IEEE Network Band 14 (2000), Nr. 3, Seiten 16–23, IEEE.

[MXZ+05] X. Meng, Z. Xu, B. Zhang, G. Huston, S. Lu und L. Zhang, *IPv4 Address Allocation and the BGP Routing Table Evolution*, ACM SIGCOMM Computer Communication Review Band 35 (2005), Nr. 1, Seiten 71–80, ACM.

[MZF07] D. Meyer, L. Zhang und K. Fall, *Report from the IAB Workshop on Routing and Addressing*, IETF Request for Comments 4984, September 2007.

[Pas08] N. Paskin, *Digital Object Identifier (DOI) System*, Encyclopedia of Library and Information Sciences (2008), Taylor and Francis New York.

[PBLC10] E. Pitler, S. Bergsma, D. Lin und K. Church, *Using web-scale N-grams to Improve base NP Parsing Performance*, In Proceedings of the 23rd International Conference on Computational Linguistics, Association for Computational Linguistics, 2010, Seiten 886–894.

[Pen08] G. Peng, *CDN: Content Distribution Network*, Forschungsbericht, State University of New York at Stony Brook, Department of Computer Science, Februar 2008.

[Per10] C. Perkins, *IP Mobility Support for IPv4, Revised*, IETF Request for Comments 5944, 2010.

[PPJB08] J. Pan, S. Paul, R. Jain und M. Bowman, *MILSA: A Mobility and Multihoming Supporting Identifier Locator Split Architecture for Naming in the Next Generation Internet*, Global Telecommunications Conference, GLOBECOM, IEEE, Dezember 2008, Seiten 1–6.

[QIdLB07] B. Quoitin, L. Iannone, C. de Launois und O. Bonaventure, *Evaluating the Benefits of the Locator/Identifier Separation*, In Proceedings of 2nd ACM/IEEE International Workshop on Mobility in the Evolving Internet Architecture (Kyoto), MobiArch '07, ACM, 2007, Seiten 1–6.

[QPV01] L. Qiu, V.N. Padmanabhan und G.M. Voelker, *On the Placement of Web Server Replicas*, In the Proceedings of the 20th Annual Joint Conference of the IEEE Computer and Communications Societies, INFOCOM 2001, Band 3, IEEE, 2001, Seiten 1587–1596.

[RFC81a] *Internet Protocol*, IETF Request for Comments 791, September 1981.

[RFC81b] *Transmission Control Protocol*, IETF Request for Comments 793, September 1981.

[RFC93] *Classless Inter-Domain Routing (CIDR): an Address Assignment and Aggregation Strategy*, IETF Request for Comments 1519, September 1993.

[RFC06] *Host Identity Protocol (HIP) Architecture*, IETF Request for Comments 4423, Mai 2006.

[RFH+01] S. Ratnasamy, P. Francis, M. Handley, R. Karp und S. Shenker, *A Scalable Content-Addressable Network*, ACM SIGCOMM Computer Communication Review Band 31 (2001), Nr. 4, Seiten 161–172.

[RMKdG94] Y. Rekhter, B. Moskowitz, D. Karrenberg und G. de Groot, *Address Allocation for Private Internets*, IETF Request for Comments 1597, März 1994.

[RSA78] R.L. Rivest, A. Shamir und L. Adleman, *A Method for Obtaining Digital Signatures and Public-Key Cryptosystems*, Communications of the ACM Band 21 (1978), Nr. 2, Seiten 120–126, ACM.

[SGH+10] D. Schwerdel, D. Günther, R. Henjes, B. Reuther und P. Müller, *German-Lab Experimental Facility*, Future Internet-FIS 2010 (2010), Seiten 1–10, Springer.

[SHB06] B. Schroeder und M. Harchol-Balter, *Web Servers Under Overload: How Scheduling Can Help*, ACM Transactions on Internet Technology (TOIT) Band 6 (2006), Nr. 1, Seiten 20–52, ACM.

[SHG+11] D. Schwerdel, D. Hock, D. Günther, B. Reuther, P. Tran-Gia und P. Müller, *ToMaTo-A Network Experimentation Tool*, 7th International Conference on Testbeds and Research Infrastructures for the Development of Networks and Communities (TridentCom 2011), 2011.

[SMK+01] I. Stoica, R. Morris, D. Karger, M.F. Kaashoek und H. Balakrishnan, *Chord: A Scalable Peer-to-Peer Lookup Service for Internet Applications*, ACM SIGCOMM Computer Communication Review Band 31 (2001), Nr. 4, Seiten 149–160, ACM.

[SRC84] J. Saltzer, D. Reed und D. Clark, *End-To-End Arguments in System Design*, ACM Transactions on Computer Systems Band 2 (1984), Nr. 4, Seiten 277–288.

[VC09] M. L. Van Name und B. Catchings, *BIND DNS Performance on Dell and Sun Servers Running Solaris*, Forschungsbericht, Principled Technologies, Inc., Januar 2009.

[VMEK+08] L. Völker, D. Martin, I. El Khayat, C. Werle und M. Zitterbart, *An Architecture for Concurrent Future Networks*, In Proceedings of the 2nd GI/ITG KuVS Workshop on the Future Internet (Karlsruhe), Citeseer, 2008.

[VMW+09] L. Völker, D. Martin, C. Werle, M. Zitterbart und I. El Khayat, *Selecting Concurrent Network Architectures at Runtime*, In Proceedings of the International Conference on Communications, ICC (Dresden), IEEE, 2009.

[Zhe04] D. Zheng, *Differentiated Web Caching – A Differentiated Memory Allocation Model on Proxies*, PhD-Thesis, Queens University, Ontario, Kanada, 2004.

[Zim80] H. Zimmermann, *OSI Reference Model—The ISO Model of Architecture for Open Systems Interconnection*, IEEE Transactions on Communications Band 28 (1980), Nr. 4, Seiten 425–432, IEEE.

[Zöl08] S. Zöls, *Peer-to-Peer Networking in Heterogeneous Communication Environments*, Dissertation, Technische Universität München, München, 2008.

Zitierte Webseiten

[Ahr07] J. Ahrenholz, *HIP DHT Interface*, `http://tools.ietf.org/html/draft-ahrenholz-hiprg-dht-01`, IETF Internet Draft, Februar 2007.

[Aka] *Akamai Technologies, Inc.*, `http://www.akamai.com`.

[Asi11] Asian Pacific Network Information Center (APNIC), *Two /8s allocated to APNIC from IANA*, `http://www.apnic.net/publications/news/2011/delegation`, Februar 2011.

[BL89] T. Berners-Lee, *Information Management: A Proposal*, `http://www.w3.org/History/1989/proposal.html`, März 1989.

[Bun10] Bundesamt für Sicherheit in der Informationstechnik, *BSI-Studie: DNSSEC-Unterstützung durch Heimrouter*, `https://www.bsi.bund.de/cae/servlet/contentblob/995592/publicationFile/63661/DNSSEC_pdf.pdf`, Referat Internetsicherheit, 2010.

[CAI09] CAIDA: The Cooperative Association for Internet Data Analysis, *Archipelago Monitor Statistics*, `http://www.caida.org/projects/ark/statistics`, 2009.

[FFML09a] D. Farinacci, V. Fuller, D. Meyer und D. Lewis, *LISP Alternative Topology (LISP+ALT)*, `http://tools.ietf.org/html/draft-fuller-lisp-alt-05`, IETF Internet Draft, Februar 2009.

[FFML09b] D. Farinacci, V. Fuller, D. Meyer und D. Lewis, *Locator/ID Separation Protocol (LISP)*, `http://tools.ietf.org/html/draft-farinacci-lisp-12`, IETF Internet Draft, März 2009.

[G-L] *G-Lab*, `http://www.german-lab.de`.

[Hus11a] G. Huston, *AS65000 BGP Routing Table Analysis Report*, `http://bgp.potaroo.net/as2.0/bgp-active.html`, 2011.

[Hus11b] G. Huston, *IPv4 Address Pools—Advertised*, `http://bgp.potaroo.net/ipv4-stats/prefixes_adv_pool.txt`, 2011.

[IAN11] IANA Internet Assigned Numbers Authority, *Protocol Numbers*, `http://www.iana.org/assignments/protocol-numbers/protocol-numbers.xml`, 2011.

[Org11] The Number Resource Organisation, *Free Pool of IPv4 Address Space Depleted*, `http://www.nro.net/news/ipv4-free-pool-depleted`, Februar 2011.

[Pla] *Planet-Lab*, `http://www.planet-lab.org`.

[PM11] P. Pelt und J. Massar, *SixXS - IPv6 Deployment & Tunnel Broker*, `http://www.sixxs.net/main`, 2011.

[Rip11] Ripe NCC: Réseaux IP Européens Network Coordination Centre, *Average AS Path Length*, `http://www.ris.ripe.net/risreport/all/total/asr.shtml`, 2011.

[Tel11a] TeleGeography Research, *Global Internet Geography*, `http://www.telegeography.com/research-services/global-internet-geography/`, 2011.