Entwicklung eines sicheren Protokolls für ein Peer-to-Peer-Hotspot–Accounting-System

Diplomarbeit am Institut für Telematik
Prof. Dr. M. Zitterbart
Fakultät für Informatik
Universität Karlsruhe (TH)

Von
cand. inform.
Manuel Thiele

Betreuer:
Prof. Dr. habil. M. Zitterbart
Dr.rer.nat.Thomas Fuhrmann
Dipl. Ing. Kendy Kutzner

Tag der Anmeldung: 15.08.2004
Tag der Abgabe: 15.02.2005

Ich erkläre hiermit, dass ich die vorliegende Arbeit selbständig verfasst und keine anderen als die angegebenen Quellen und Hilfsmittel verwendet habe.
Karlsruhe, den 15.Februar 2005.

Danksagung:

An dieser Stelle möchte ich mich bei Frau Prof. Dr. Zitterbart für die Möglichkeit bedanken, diese Diplomarbeit am Institut für Telematik erarbeiten zu können.

Herrn Dr. Fuhrmann – von dem auch diese interessante Themenstellung stammt - und besonders Herrn Dipl. Ing. Kendy Kutzner danke ich für fruchtbare Hinweise und anregende Diskussionen, die für den Fortschritt meiner Arbeit wichtig waren.

Inhaltsverzeichnis

Abbildungsverzeichnis

Verzeichnis der Tabellen

Kapitel I

Einleitung

Heutzutage ist das Internet eine der zentralen Telekommunikations-Infrastrukturen. Es wird von einer Vielzahl von Medien und Diensten genutzt, welche für viele Menschen sehr wichtig geworden sind. Beispiele hierfür sind das World Wide Web, E-Mails, IRC oder Filesharing.

Des Weiteren wird von den Menschen seit dem letzten Jahrzehnt immer mehr Mobilität gewünscht. Man möchte jederzeit an jedem Ort all das nutzen können, was einem zu Hause zur Verfügung steht. Ein gutes Beispiel hierfür ist die inzwischen weite Verbreitung von Mobiltelefonen. Diese Wünsche nach Mobilität sind besonders stark in jungen Bevölkerungsschichten vertreten und bei Berufstätigen, die häufig unterwegs sind, (siehe [10], [09], [08]).

Im Bereich des Internetzugangs ist diese „Allgegenwärtigkeit“ zur Zeit nur bedingt möglich. So ist der Zugang zum Internet über die Mobiltelekommunikation (GSM, GPRS, UMTS) relativ teuer. Bei der Nutzung von (Internet-) Providern, die über W-LAN-Access-Points verfügen, ist trotz eines inzwischen einheitlichen Kommunikationsstandards (IEEE 802.11[bag]) der Zugang räumlich begrenzt und kompliziert. Hierbei deckt jeder der Anbieter mit seinen Access-Points jeweils nur einen kleinen Raum ab, weswegen sich der Benutzer je nach Standort über verschiedene Provider ins Internet „einwählen“ muss. Bevor ein Benutzer sich aber einwählen darf, muss der Benutzer beim Provider registriert sein, wofür eine teilweise umständliche Anmeldung notwendig ist. Roaming, wie es bei Mobiltelefonanbietern praktiziert wird, wird praktisch nicht angeboten.

Auf Grund der Bequemlichkeit und Erschwinglichkeit von W-LAN-Geräten sind diese in Privathaushalten zunehmend weit verbreitet. Häufig besitzen solche Haushalte auch einen Breitbandanschluß (DSL, Kabel, …) gekoppelt mit einer sowohl zeitlich als auch vom Volumen her unbegrenzten Flatrate.

Diese weite Verbreitung eröffnet die Möglichkeit einer Kooperation vieler solcher Privatleute durch einen Zusammenschluss zu einer Gemeinde. In dieser könnte jedes Mitglied Zugang zum Internet über den Anschluss eines anderen Mitgliedes erhalten. Im Gegenzug müsste dieses aber jedem anderen Mitglied der Gemeinde Zugang über seinen eigenen Anschluss gewähren. Im Idealfall wäre hierdurch ein mindestens in Ballungsgebieten flächendeckender und kostengünstiger Internetzugang über W-LAN möglich.

I.1. Gliederung der Arbeit

Die Arbeit ist in sechs Teile untergliedert. Im ersten befindet sich die Einleitung. Der zweite enthält Beschreibungen von Gegebenheiten, wie beispielsweise von Verhalten von Menschen. Der darauf folgende Teil enthält den Hauptteil der Arbeit. In ihm werden Lösungsmöglichkeiten für das Problem zuerst diskutiert und dann jeweils eine Möglichkeit ausgewählt. Eine Wahl von Parametern und eine Überprüfung der Angemessenheit der Protokolle finden sich im vierten Teil der Arbeit. Offene Punkte und Anregungen, was man noch weiter verändern und verbessern könnte, finden sich im vorletzten Kapitel. Die Arbeit wird von einer Zusammenstellung der Referenzen vervollständigt.

Kapitel II

Das Szenario

II.1. Das Ziel

Ziel dieser Arbeit ist es, eine einfache, kostengünstige und weit verbreitete Möglichkeit zu schaffen, um Zugriff auf das Internet zu erhalten, ohne an einen bestimmten Ort gebunden zu sein. Hierzu soll die Gruppe der Privatleute mit pauschalem Breitband-Internetzugang und W-LAN-Equipment zur Bildung und anschließenden Vergrößerung einer Gemeinde angeregt werden, in der jedes Mitglied über andere Zugriff auf das Internet erhalten kann. Diese Aufgabe soll ein vollkommen dezentrales Softwaresystem übernehmen.

Der Zugriff von „außen“ auf das Internet soll über W-LAN vonstatten gehen, was natürlich voraussetzt, dass jedes Gemeindemitglied entsprechende Hardware besitzt. W-LAN ermöglicht ohne großen Aufwand und Kosten bei der Hardwareinstallation einen flächigen Zugriff auf das entsprechende Netzwerk und somit das Internet. Des Weiteren ist es verfügbar, standardisiert (nach IEEE 802.11 [bag]) und es entstehen kaum laufende Kosten während des Betriebes.

Eine weitere Voraussetzung für die Mitgliedschaft muss eine hinreichend schnelle Verbindung zum Internet sein. Diese muss auch mehreren Benutzern zugleich genügend Kapazität zur Verfügung stellen, ohne dass dem Besitzer hierdurch starke Beeinträchtigungen entstehen. Deswegen wird ein DSL- oder ein anderer Breitband-Anschluss vorausgesetzt. Dieser sollte zeitlich und im Volumen unbegrenzt sein. Des Weiteren sollte es mit dem entsprechenden Vertrag auch legal sein, dass mehrere Benutzer ihn zur gleichen Zeit verwenden. Es fällt dabei in die Verantwortung jedes einzelnen Mitgliedes, dass es diese Voraussetzungen schafft. Wenn es dieses nicht einhält, muss es selbst das finanzielle Risiko bzw. die möglichen rechtlichen Konsequenzen tragen.

Zur Zugangskontrolle zum Internet und für weitere nötige Dienste muss jedes Mitglied zu guter Letzt einen Linux-PC zur Verfügung stellen. Dieser darf auch älteren Baudatums oder mit dem ohnehin nötigen Labtop identisch sein.

II.2. Der Status Quo des W-LAN-Zugriffes

Nach Elias C. Efstathiou and George C. Polyzos ist der Status Quo der W-LAN Verfügbarkeit, wie sie in ihren Arbeiten zur Verbesserung der W-LAN-Verfügbarkeit beschreiben (siehe [14], [13]), wie folgt: In Amerika wird W-LAN hauptsächlich von einer Vielzahl kleiner und kleinster Unternehmen angeboten. Deswegen ist dort der Markt stark fragmentiert. In Europa ist der Internetzugang über W-LAN noch nicht so weit entwickelt. Dieses wird hauptsächlich damit begründet, dass die bekannten und etablierten Mobiltelefonanbieter den Markt unter sich aufteilen werden.

Außer den eigentlichen W-LAN-Service-Providern gibt es so genannte Aggregatoren. Diese besitzen selber keine Access-Points, sondern arbeiten mit einer Vielzahl von kleinen Unternehmen zusammen. Dabei übernimmt der Aggregator die Führung der Benutzerdaten inklusive der Abrechnung für die einzelnen Provider. Hierdurch kann dieser eine relativ große Abdeckung erhalten, ohne viel investieren zu müssen.

Des Weiteren gibt es Roaming-Verträge zwischen einigen wenigen Providern, beispielsweise zwischen iPass und T-Mobile. Hierdurch können Benutzer des einen über das Netzwerk des anderen surfen, ohne sich erneut registrieren zu müssen.

Zusätzlich gibt es noch die öffentliche Hand oder private Vereine von beispielsweise W-LAN begeisterten Personen, welche Internetzugänge bereitstellen. Diese sind häufig im Gegensatz zu denen der kommerziellen Anbieter kostenlos benutzbar.

Trotzdem decken selbst die größten dieser Netzzusammenschlüsse nur einen relativ kleinen Bereich ab. Des Weiteren behindert noch eine Inkompatibilität der AAA-Systeme (Authentication, Authorization and Accounting) ein problemloses Roaming. Hier gibt es aber auch Ansätze die Systeme zu standardisieren (siehe [12]).

Zu guter Letzt wird von der „Athens University of Economics and Business“ ein Peer-to-Peer-Netzwerk entwickelt, welches versucht, die Brücke zwischen kommerziellen und nichtkommerziellen Anbietern zu schlagen, um eine hohe Abdeckung zu erreichen. Hierbei wird versucht, auch die bis jetzt ungenutzten Ressourcen der Privatnutzer zu mobilisieren. Um aber auch für kommerzielle Anbieter interessant zu sein, wird eine Art Währung eingeführt (siehe [14], [13]). Nach der Meinung des Autors ist diese Art der Abrechnung nicht geeignet, Privatnutzern einen genügenden Anreiz zu bieten, um sich stark in dem System zu engagieren (Details unter III.1.1).

Normalerweise wird von den Anbietern übertragenes Datenvolumen oder Onlinezeit abgerechnet. Dabei werden von Europäischen Mobilfunkunternehmen teilweise sehr große Taktschritte wie beispielsweise eine Stunde gewählt (siehe [31],[32]). Dieses wird von den Medien damit begründet, dass W-LAN und UMTS miteinander konkurrieren. Einerseits wollen die Mobilfunkbetreiber Geld mit W-LAN erwirtschaften, aber andererseits darf W-LAN nicht so attraktiv werden, dass die teuren UMTS-Lizenzen wertlos werden (siehe [31]).

Ein weiterer Punkt ist die sehr geringe Nutzung der kommerziell angebotenen Hotspots. So wurde von „Austrian Hotspot“ eine durchschnittliche Surfdauer der registrierten Benutzer von nur 1,7 min / Tag gemeldet (siehe[32]). Ein anderer Betreiber („BT Openzone“) meldete es als Erfolg, dass bei ihm durchschnittlich eine Stunde akkumulierter Onlinezeit pro Hotspot und Tag erreicht werden würde. Hierzu ist noch zu sagen, dass dieser Provider viele Hotspots in sehr guter Lage besitzt wie Gatwick Airport, mehrere Hilton Hotels oder das „Earl's Court exhibition center“ in London (siehe [33]). Allerdings kann dieses Phänomen auch mit den Kosten des Zuganges zusammenhängen. Wenn der Zugang umsonst ist, so wird dieser erheblich besser angenommen (siehe [32]).

II.3. Zugriffs-Garantien in der Gemeinde

Es kann in der geplanten Gemeinde keinerlei Garantien für die Verfügbarkeit des Internetzugriffes an bestimmten Orten oder zu bestimmten Zeiten geben. Dieses ist primär durch die Hoheitsrechte der Mitglieder an ihren eigenen Geräten begründet. Beispielsweise hat jedermann das Recht, seinen PC an- oder abzuschalten, wann er will. Dieses Recht wird nur sehr ungern durch den Beitritt zu einer Gemeinde aufgegeben. Aus diesem Grund würde eine solche Vorschrift das Wachstum der Gemeinde wahrscheinlich behindern, wobei die Wirksamkeit solch einer Vorschrift zusätzlich fraglich wäre.

Bestenfalls kann die Entscheidung durch daraus resultierende Vor- und Nachteile (siehe II.4.5) beeinflusst werden. Die örtliche Unsicherheit entsteht aber auch durch das „Wachsen“ der Gemeinde. Die Orte, an denen ein Zugang zum Internet möglich ist, sind an Haushalte gebunden und werden nicht systematisch ausgebaut. Deswegen sind die Standorte der Hotspots nicht gleichmäßig verteilt (vergleicheII.4.2), sondern werden durch das teilweise zufällige Hinzukommen von neuen Mitgliedern bestimmt.

II.4. Handlungsmuster der Benutzer

II.4.1. Das Prinzip der Nutzenmaximierung

Aus der Sicht der Gemeinde ist es wünschenswert, dass sich jedes Mitglied möglichst stark einbringt. Denn je häufiger die einzelnen Mitglieder durchschnittlich bereit sind, als Provider zu fungieren, desto größer ist die Anzahl der Leute, die zu einem bestimmten Zeitpunkt dieses tun. Je größer diese Zahl wiederum ist, desto größer ist auch die Abdeckung, die die Gemeinde erreicht, aus der einfachen Rechnung heraus, dass jedes Mitglied, welches gerade als Provider fungiert, ein bestimmtes Gebiet abdeckt. Durch die höhere Abdeckung wird der Nutzen der Gemeinde als ganzes für jedes einzelne Mitglied wiederum erhöht.

Ein hoher Nutzen für die Mitglieder bedeutet sowohl eine geringe Austrittsrate als auch eine hohe Eintrittsrate. Dieses entsteht dadurch, dass ein hoher Nutzen einer hohen Attraktivität sowohl für Mitglieder als auch für Nichtmitglieder entspricht.

Aus diesen Gründen sollte es ein Ziel für die Entwicklung der Gemeinde sein, dass, wie oben schon gesagt, jedes Mitglied sich maximal einbringt. Idealerweise wären also alle Mitglieder 24 Stunden am Tag, 7 Tage die Woche und 365 Tage im Jahr online und bereit, anderen als Provider zu fungieren.

Aus der Sicht des Einzelnen sollte es aber durchaus möglich sein, den Rechner auszuschalten, wenn man beispielsweise nicht zuhause ist, und es sollte trotzdem noch möglich sein, die Gemeinde in Anspruch zu nehmen. Dieses sollte auch über einen längeren Zeitraum wie z.B. während eines Urlaubes möglich sein. Auch dieses würde dadurch, dass es den Nutzen des einzelnen erhöht, der Gemeinde als Ganzes helfen.

II.4.2. Die Benutzerstrukturen

Der Nutzen der Gemeinde für den Einzelnen bestimmt sich aber auch aus der Größe und der Struktur derselbigen. Wenn es zu viele Orte gibt, an denen ein potentielles Mitglied gerne online wäre, aber niemand als Provider fungiert, dann ist für diese Person die Teilnahme an der Gemeinschaft nutzlos. Deswegen wird er entweder gar nicht Mitglied sein oder sich nicht engagieren. Idealerweise sollte es deshalb zu jeder Zeit an jedem Ort möglich sein, ins Internet zu gelangen.

Dieses ist jedoch unrealistisch. Da über die Gemeinde kein Profit im eigentlichen Sinne (wohl aber ein Vorteil) zu erzielen ist, wird es auch keinen systematischen Aufbau der Infrastruktur geben. Die Infrastruktur wird an Haushalte und deren Verteilung gebunden sein. Dementsprechend wird die Dichte in Ballungsgebieten sehr viel höher sein als in dünner besiedelten Bereichen. Des Weiteren ist es wahrscheinlich, dass in Wohngegenden das Netz dichter ist als in Geschäftsbezirken. Begründen lässt sich dies dadurch, dass der Nutzen für Privatpersonen entsteht und Geschäfte nur bedingt Interesse an einer mobilen Internetlösung ohne jegliche Garantien haben.

Zusätzlich gibt es in Ballungsgebieten aber auch viel mehr Menschen als in ländlichen Gegenden und damit auch einen größeren Bedarf an Internetzugängen. Zu der an sich schon höheren Personendichte kommen noch Gäste aus dem Umland dieser Stadt oder aus anderen Gebieten, welche den Bedarf weiter erhöhen. Hierdurch werden die Access-Points trotz ihrer relativ hohen Dichte relativ stark frequentiert.

Innerhalb der Ballungsgebiete ist allerdings der Bedarf auch nicht gleich verteilt. Wenn man zu Hause ist, wird man über den eigenen Anschluss surfen, nur wenn man dort nicht ist, braucht man die Gemeinde. Normalerweise verlässt man sein Zuhause, um Besorgungen zu machen, in der Freizeit oder aus beruflichen Gründen. Die Gemeinde wird also in Einkaufs-, Erholungs- und Geschäftsgebieten stärker in Anspruch genommen werden als in solchen Gebieten, die nicht für diese Art der Aktivitäten geeignet sind. Wohngebiete, also die Bereiche, die eine besonders hohe Abdeckung haben, sind allerdings nicht unbedingt in diese Kategorien einzuordnen. Deswegen kann es zu einer ungleichen Belastung der Hotspots kommen.

II.4.3. Von Leistungsminimierung und Missbrauch

In II.4.1 wurde geschrieben, dass sich idealerweise alle Mitglieder 24 Stunden am Tag, 7 Tage die Woche und 365 Tage im Jahr engagieren sollten. Allerdings ist davon auszugehen, dass der größte Teil der Gemeinde egoistisch ist und sein Engagement für die gewünschte Leistung minimiert. Dieses entspricht der Standardtheorie der Ökonomen, die von dem so genannten „Homo economicus“ ausgehen (siehe [02]).

Nach dieser Theorie wird sich das Engagement der Mitglieder nach einer Kosten-Nutzen-Rechnung richten. Diese werden sich überlegen, was sie von der Gemeinschaft erwarten und wie viel sie dafür „zu bezahlen“ haben. Des Weiteren werden sie sich überlegen, wie viel sie bereit sind, für jede Leistung zu zahlen. Nach dieser Abwägung wird jeder sein Verhalten richten und dabei sein Engagement für die gewählte Leistung minimieren (siehe [14]). Die Abwägung wird dabei sowohl individuell als auch gruppenspezifisch unterschiedlich ausfallen. Man nehme beispielsweise jemanden an, der nur an einem Ort Internetzugriff braucht. Dieser wird sehr viel eher einen lokalen Anbieter benutzen als jemand, der an vielen verschiedenen Orten in das Internet will. Der Letztere müsste sich nämlich an jedem Ort einzeln anmelden, was einen großen Aufwand erfordert.

Wenn ein starkes Engagement der Mitglieder erreicht werden soll (siehe II.4.1), darf also nicht jedes Mitglied soviel die Gemeinde in Anspruch nehmen können, wie es will, sondern es muss Regeln geben, die das Engagement der Mitglieder mit den Leistungen koppeln, die diese erwarten dürfen.

Allerdings wird das Minimieren des Engagements sogar noch weiter gehen, als man vermuten sollte. So werden Regeln, die nicht überwacht werden, von vielen gar nicht beachtet und damit gebrochen. Solch eine Regel ist bei Gnutella, dass jeder der Benutzer so viele Dateien freigibt, wie es ihm möglich ist (siehe [35]). Allerdings werden mehr als die Hälfte der Dateien von nur sieben Prozent der Benutzer bereitgestellt. 26% der Benutzer bieten sogar gar keine Dateien (siehe [02]) an.

Aus diesem Grund sollte jede Regel, die aufgestellt wird, auch von der Gemeinschaft überwacht und ihre Missachtung mit Sanktionen belegt werden. Dieses wird normalerweise die meisten Benutzer dazu veranlassen, sich regelkonform zu verhalten.

Es werden jedoch immer wenige Prozente der Mitglieder bleiben, die weiterhin gewillt sind, ihren Vorteil auf Kosten der anderen Benutzer zu erschleichen oder die versuchen, der Gemeinde aus anderen Gründen mutwillig Schaden zuzufügen. Dieses wird teilweise mit erheblicher krimineller Energie einhergehen.

Wenn man solches Verhalten nicht dulden will, muss man dieses Verhalten schon während der Entwicklung des Systems und der benutzten Protokolle bedenken. Man muss jedem Mitglied der Gemeinde misstrauen (siehe [37]).

Um bei der Entwicklung der konkreten Algorithmen einfache Angriffspunkte identifizieren zu können, ist eine genauere Kategorisierung der Angreifer von Vorteil. So gibt es folgende drei Bedrohungsmodelle (siehe [01]):

1. Einzeltäter (siehe II.4.3.1),
2. normale Gruppen (siehe II.4.3.2) und
3. gezielt zusammengestellte Gruppen (siehe II.4.3.3).

II.4.3.1. Missbrauch durch Einzeltäter

Einzeltäter sind einzelne Menschen, die ohne „fremde“ Hilfe versuchen, ihren Vorteil auf Kosten von anderen zu ergattern. Diese können bereit sein, viel Zeit und eventuell Geld zu investieren, um einen Vorteil zu erlangen. Hierbei werden sie sich nur insoweit an vorgegebene Regeln halten, wie sie dieses unbedingt müssen, um nicht ihren Vorteil wieder zu „verlieren“.

Dieses Modell wird das wohl am häufigsten zu findende sein. Es birgt aber auch das geringste Bedrohungspotential in sich, da diese Täter relativ wenige Möglichkeiten haben zu betrügen und die Betrugsfälle relativ einfach aufgedeckt oder verhindert werden können. Dieses Modell sollte aber trotzdem ernst genommen werden.

In diese Kategorie fallen auch diejenigen, die keine nennenswerte kriminelle Energie besitzen, aber die Regeln missachten, wenn sie nicht überprüft werden und dieses Verhalten mit Sanktionen belegt ist.

Wenn ein Einzeltäter über genügend kriminelle Energie verfügt, kann er auch als Gruppe (siehe II.4.3.2) in Erscheinung treten. Dieses passiert, wenn er sich mit mehreren Konten und eventuell mehreren Rechnern gleichzeitig im System anmeldet (Sybil Attack, siehe [36]).

II.4.3.2. Missbrauch durch normale Gruppen

Gruppen bestehen aus einer Reihe von Mitgliedern, die sich kennen und gegenseitig absprechen, um der Gruppe einen Vorteil zu verschaffen. Hierzu verhalten sich deren Mitglieder ähnlich wie Einzeltäter (siehe II.4.3.1). Solch eine Gruppe hat aber mehr Möglichkeiten als ein Einzeltäter. Sie können sich gegenseitig decken oder koordiniert die Gemeinschaft an mehreren Stellen gleichzeitig angreifen.

Besonders vorteilhaft für die Gruppe ist es, wenn sie möglichst viele ihrer Mitglieder in verantwortliche Positionen der Gemeinschaft einschleusen kann. Hierdurch kann sie eventuell eine vollständige Kontrolle über die eigenen Kontodaten erlangen und diese beliebig manipulieren.

Das Bedrohungspotential wächst hierbei mit der Größe der Gruppe. Beides kann außerdem noch weiter erhöht werden, indem sich jedes Mitglied mehrere Identitäten zulegt (Sybil Attack, siehe [36]).

Es wird angenommen, dass solche Gruppen im Vergleich zu der Größe der Gemeinde relativ klein sind. Wenn dem nicht so ist, so ist eine Mitgliedschaft in der Gemeinde nicht erstrebenswert, da diese nur zur Ausbeutung der „unbedarften“ Mitglieder besteht.

Dieses Modell wird seltener sein als das der Einzeltäter, birgt aber ein erheblich höheres Bedrohungspotential in sich.

II.4.3.3. Missbrauch durch gezielt zusammengestellte Gruppen

Durch gezielte Sicherheitsmaßnahmen (z.B. zufällige Auswahl von den Verantwortungsträgern) kann man verhindern oder wenigstens stark erschweren, dass eine Gruppe ihre Mitglieder in die verantwortlichen Positionen der Gemeinschaft bringt. Diese Mechanismen erschweren es aber nur, gezielt Personen in eine Position zu bringen. Um aber eine Person in verantwortlicher Position an Betrügereien zu hindern, braucht man andere Mechanismen, die noch aufwändiger sind als jene, um Gruppen daran zu hindern, ihre Mitglieder in bestimmte Positionen zu bringen. Dieses können sich kriminelle Elemente zu Nutze machen und gezielt die Verantwortungsträger einer Gemeinschaft ansprechen und anwerben, mit ihnen gemeinsame Sache zu machen. Das Anwerben kann dabei auf verschiedene Art und Weise stattfinden. Die Palette reicht von Bestechung bis hin zur Bedrohung. Im Falle dieser Gemeinschaft könnte ein Mitglied beispielsweise mit DOS-Attacken bedroht werden.

Dieses Modell ist das seltenste in dieser Zusammenstellung, ist aber auch das am schwersten zu kontrollierende und damit das gefährlichste Modell.

II.4.3.4. Missbrauch aus emotionalen Gründen

Zusätzlich gibt es noch Leute, die aus anderen Gründen Angriffe auf diverse Ziele im Internet durchführen. Die Einstufung dieser ist aber mit jenen identisch, die aus Gewinngründen angreifen. Die „anderen“ Gründe sind im weitesten Sinne emotionale Gründe. In „Hacking Attacks - How and Why„ ([07])wird folgende Aufzählung vorgenommen:

1. um Prestige zu gewinnen,
2. um den Nervenkitzel zu erhalten,
3. um Aufmerksamkeit zu bekommen,
4. weil es eine geistige Herausforderung ist,
5. einfach aus Neugierde,
6. um hilfreich zu sein, indem man Sicherheitslücken aufdeckt oder
7. um Rache zu nehmen

Dabei muss nicht ein einzelner dieser Gründe zum entsprechenden Verhalten führen, sondern es können auch mehrere beteiligt sein. So gibt es etliche Gruppen im Internet, die sich gegenseitig und innerhalb der Gruppe versuchen, mit ihrem letzen „Hack“ zu übertreffen und dabei möglichst viel Aufmerksamkeit zu erlangen.

In einer Darstellung der Gefahren durch Hacker eines Providers ([49]) und in Selbsterklärungen von Hackern ([50], [51]) wird angegeben, dass die meisten „Hacker“ weder aus monetären Gründen noch aus Rache ihrer Beschäftigung nachgehen. Allerdings muss man sich dabei auch im Klaren sein, dass auch eine Minderheit großen Schaden anrichten kann.

II.4.4. Allgemeine Prinzipien der Gerechtigkeit

Eine weitere Kraft im Verhalten des Menschen ist sein Gerechtigkeitssinn, der das Verhalten mitbestimmt und der bei den Abrechnungsmodalitäten beachtet werden sollte.

Das Standardmodell der Ökonomen sagt voraus, dass der „Homo Economicus“ alles akzeptiert und begrüßt, bei dem er nur den geringsten Gewinn erzielen kann (siehe [02]). Experimentaldaten zeigen aber, dass dem nicht so ist. Wird beispielsweise eine Verteilung als stark ungerecht empfunden, wird sie nicht akzeptiert und abgelehnt. Dieses geht so weit, dass diejenigen, die so empfinden, auch eigene Nachteile in Kauf nehmen, um diese Ablehnung zu zeigen. Auf der anderen Seite sind manche aber auch bereit mehr zu geben, als sie eigentlich

müssten, um Gerechtigkeit zu erzielen oder um eine gerechte Sache zu unterstützen. Hierbei nehmen sie auch eigene Nachteile in Kauf. Solch ein Verhalten nennt man Altruismus. Dieser ist aber zu unterscheiden von eigennützigem Verhalten, bei dem ein nicht sofort erkennbarer Nutzen, wie beispielsweise Ansehen oder Freundschaft, entsteht.

Dieser Sinn für Gerechtigkeit ist aber individuell unterschiedlich stark ausgeprägt. Der Altruismus von Menschen ist das, was viele unreglementierte Peer-to-Peer-Netze trägt (siehe [14] ‚2.1 „Free-riding, altruism and rules"). Um aber den Nutzen der Gemeinde zu maximieren (siehe II.4.1), sollten die Regeln in der Gemeinschaft (siehe II.4.3) als gerecht empfunden werden.

Dass Menschen zu Gunsten von Gerechtigkeit auf einen eigenen Nutzen verzichten, sieht man beispielsweise an einem hierfür entwickelten bekanntem Experiment, dem „Ultimatum-Spiel" (beispielsweise siehe [16]):

> *Das Spiel wird von zwei Teilnehmern gespielt, die sich nicht kennen. Der erste erhält provisorisch eine bestimmte Geldsumme. Diese kann er beliebig zwischen dem zweiten Spieler und sich selbst aufteilen. Hierauf kann der zweite das Angebot annehmen oder ablehnen. Nimmt er es an, so erhält jeder der beiden die entsprechende Summe. Lehnt er es aber ab, so darf keiner der beiden Spieler Geld mit nach Hause nehmen.*
>
> *Bei dem Spiel werden die Regeln beiden Spielern im Voraus erklärt, aber es besteht keine Möglichkeit miteinander Absprachen zu tätigen.*

Das Standardmodell sagt vorher, dass der zweite Spieler jedes Angebot annehmen wird, bei dem er überhaupt etwas bekommt. Denn wenn er so ein Angebot ablehnen würde, erhielte er ja weniger, als wenn er es annehmen würde.

Experimente zeigen jedoch, dass dem nicht so ist. Im westlichen Kulturkreis werden durchschnittlich zwischen 30% und 40% der Gesamtsumme angeboten. Normalerweise werden dabei Angebote, die unter 20% liegen, ausgeschlagen. In anderen Kulturen ist dieses jedoch anders, wobei der Zusammenhang zwischen der Kultur und der Höhe der gemachten Angebote sowie die Mindesthöhe der angenommenen Angebote noch erforscht wird.

Aus diesen Daten sieht man, dass der Mensch bereit ist, einen eigenen Verlust hinzunehmen, um eine ungerechte Verteilung zu „rügen".

Ein entsprechendes Verhalten wurde in neueren Versuchen auch Affen zugeschrieben, weswegen eine genetische Komponente für den „Unwillen" bei Ungerechtigkeiten postuliert wird (siehe[17]).

Wie es zu solchem Verhalten kommt, bei dem Ungerechtigkeiten auch auf eigene Kosten „gerügt" wird, versucht die Theorie der „Ungerechtigkeits-Aversion" (siehe [18]) zu erklären. Diese besagt, dass der Mensch nicht nur an seinem eigenen Gewinn interessiert ist, sondern auch an der Verteilung der Gewinne zwischen den „Handelspartnern". Ist die Verteilung ungerecht, so entsteht dadurch ein „Unnutzen". Anders ausgedrückt sinkt durch die Ungerechtigkeit der Nutzen beziehungsweise der Gewinn, den ein Individuum erzielt hat. Ist nun der „Unnutzen" größer als der Nutzen, so wird das „Geschäft" nicht mehr akzeptiert und deswegen ausgeschlagen.

Ein anderes Spiel zeigt, dass Menschen auch altruistisch sein können:

> *Dieses Spiel wird wieder von zwei einander nicht bekannten Personen gespielt. Diesmal erhält der erste Spieler allerdings eine bestimmte Geldsumme, die ihm nicht mehr genommen werden kann. Dies könnte beispielsweise 600€ sein. In diesem Beispiel muss dann der erste dem zweiten Spieler einen Betrag zusprechen, der zwischen 600€ und 1200€ liegt, welchen dieser dann vom Spielleiter ausgezahlt bekommt. Spricht der erste dem zweiten nicht die volle Summe zu, erhält er, zusätzlich zu dem Geld, das er schon erhalten hat, 1/24 des Differenz zwischen dem zugesprochenen Betrag und 1200€.*

Spricht er zum Beispiel dem Mitspieler 1000€ zu, so ist die Differenz zwischen dem Maximalbetrag und dem zugesprochenen 200€. Deswegen erhält der erste Spieler seine 600€ plus (200/24)€, also 608,33€.

Bei diesem Spiel wählten aber 2/3 der Spieler die 1200€ für den zweiten, wodurch sie selbst (nur) 600€ anstatt die maximal möglichen 625€ erhalten.

Diese starke Tendenz zu einer ungleichen Verteilung zu Ungunsten des Verteilers kann mit der Theorie der „Ungerechtigkeits-Aversion“ nicht erklärt werden.

Eine andere Theorie (siehe [20]) aber kann beides erklären. Diese besagt, dass dem Menschen, abgesehen vom eigenen Gewinn, noch die soziale Effizienz, die Gleichheit und das Wohl desjenigen wichtig ist, dem es am schlechtesten in der entsprechenden Gruppe geht. Hierzu gibt es soziale Normen, die das Verhalten und die Verteilungen festlegen. Diese sind allgemein bekannt. Wenn einer die Norm missachtet, während der andere die Norm beachtet, so bekommt der erstere Schuldgefühle. Des Weiteren empfindet derjenige Zorn, der die Norm beachtet hat, aber nicht nach ihr behandelt wurde.

Leute, die Schuldgefühle haben, können hierbei ihren Gewinn nicht so genießen wie solche ohne Schuldgefühle. Deswegen ist deren Nutzen geringer als der entsprechende Gewinn ohne Schuldgefühle.

Des Weiteren geht mit dem Gefühl des Zornes auch das der Rache einher. Dieses lässt entsprechende Menschen bereit sein, etwas zu „bezahlen“, um den zu bestrafen, der sie nicht nach den Normen behandelt hat.

Weitere Experimentaldaten zeigen, dass Systeme, in denen der Gewinn auf Faktoren beruht, die nicht von dem entsprechenden Individuum beeinflussbar sind, als ungerecht empfunden werden.

Zu guter Letzt gibt es noch ein Prinzip, dass man bei der Analyse von gerechten Verteilungssystemen entdeckt hat (siehe[21]). Dieses Prinzip wird Kohärenz genannt und besagt: „Gerechtigkeit für alle Beteiligten muss auf das selbe hinauslaufen wie Gerechtigkeit für eine beliebige Teilmenge von ihnen“(aus [21]).

Eine Verteilungsvorschrift, die diesem Grundsatz genügt, ist die proportionale Verteilung nach Aristoteles. Diese ist allgemein bekannt und wird in der westlichen Kultur häufig angewendet. Sie besagt, dass jemand den Prozentsatz von etwas zu Verteilendem, bekommt, der dem Prozentsatz seiner Ansprüche an der Summe aller Ansprüche entspricht.

Ein nicht so bekanntes Verteilungsprinzip stammt aus der Grundlage der jüdischen Zivil-, Straf- und Religionsgesetzgebung, dem Talmud. In diesem Werk wird ein Gut, auf das zwei Personen Ansprüche erheben, so verteilt, dass jeder zuerst den Teil des Gutes erhält, auf den der andere keinen Anspruch erhebt. In einem zweiten Schritt wird dann der Rest gleich auf die beiden Parteien verteilt. Dabei wird davon ausgegangen, dass es keine Ansprüche gibt, die größer sind als das zu verteilende Gut. Deswegen wird ein Anspruch der größer ist als das ganze auf den Maximalbetrag gekürzt. Dieses Verteilungsprinzip sorgt dafür, dass der absolute Verlust aller Beteiligten gleich groß ist (abgesehen von der Kürzung).

In der folgenden Tabelle werden ein paar Beispiele für die beiden Verteilungen gegeben. Es wird dabei zwischen 2 Personen ein Vermögen von 120€ und 240€ verteilt. Die erste Person hat Ansprüche auf 240€ und die zweite auf 120€.

Bei der proportionalen Verteilung erhält die erste Person immer 200/300-stel (300=200+100) =66%, die zweite 100/300=1/3 des zu verteilenden Vermögens.

Beim Talmud ist dieses anders. Im ersten Beispiel werden zunächst alle Ansprüche auf 120€ gekürzt, da es maximal 120€ zu verteilen gibt. Hiernach haben alle dieselben Ansprüche und erhalten deswegen auch dasselbe. Im zweiten Beispiel müssen keine Ansprüche gekürzt werden. Die erste Person erhält also zuerst die 120€ auf die die zweite keinen Anspruch erhebt. Die restlichen 120€ werden danach gleich verteilt. Die erste Person erhält also 120€+120€/2=180€ während die zweite nur 120€/2 erhält.

Vermögen	Ansprüche			
	240€		120€	
	Talmud	Proportional	Talmud	Proportional
120€	60,00 €	80,00 €	60,00 €	40,00 €
240€	180,00 €	160,00 €	60,00 €	80,00 €

Tabelle 1: Vergleich der zugesprochenen Geldmenge bei einer proportionalen Verteilung und bei der nach dem Talmud

II.4.5. Allgemeine Prinzipien bei der Verhaltenssteuerung durch Anreize

Wie unter II.4.3 angesprochen wurde, sollen die Regeln als Anreiz dienen, damit sich die Mitglieder möglichst stark engagieren (siehe II.4.1). Durch die Regeln wird einem Mitglied, welches ein bestimmtes Verhalten gezeigt oder eine bestimmte Leistung vollbracht hat, eine Belohnung zugesprochen.

An dieser Stelle der Arbeit soll noch offen bleiben, was genau als gewünschtes Verhalten (in der restlichen Arbeit als Leistung bezeichnet) und was genau als Belohnung angesehen wird, da unter III.1 genauer diskutiert wird, was hierbei am besten geeignet ist.

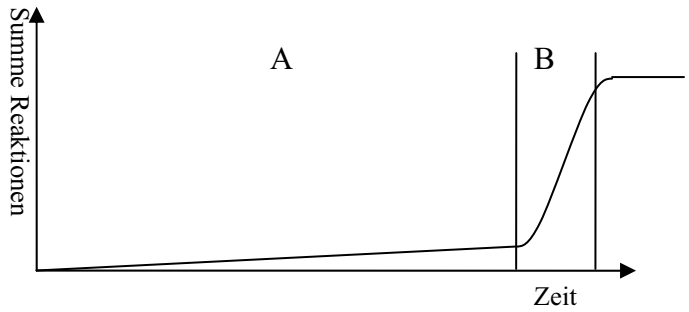

Abbildung 1: Häufigkeit des Hebelbetätigens einer Ratte in der Skinnerbox

Im so genannten Behaviorismus (siehe [22], [23], [24], [25], [26], [27], [28], [29]) wird versucht, Verhaltensweisen durch Belohnungen und Strafen zu verändern. Der Vorgang wird dort „Operante Konditionierung“ genannt. Dabei kann man mit Belohnungen dafür sorgen, dass Verhalten häufiger gezeigt wird und mit Strafen sorgt man für eine Verminderung der Häufigkeit von bestimmtem Verhalten. In dieser Arbeit interessieren deswegen hauptsächlich die Belohungen.

Ursprünglich wurde von den Behavioristen das Lernverhalten von Tieren untersucht.

In einem Versuch wird beispielsweise eine hungrige Ratte in einen Käfig gesetzt. Die einzige Nahrungsquelle ist ein Futternapf. Nachdem die Ratte sich daran gewöhnt hat aus dem

Napf zu fressen, wird ein Hebel neben dem Napf installiert. Dieser ist über eine Elektronik mit einem automatischen Futterspender verbunden, so dass immer dann eine Portion Futter in den Napf gegeben wird, wenn die Ratte auf den Hebel tritt.

Nachdem die Ratte beispielsweise die ersten dreimal zufällig auf den Hebel getreten ist (Abbildung 1:A), fängt sie an, immer häufiger und schneller den Hebel zu bedienen (Abbildung 1:B). Sie hat gelernt, dass man Futter erhält, wenn man auf den Hebel tritt. Nach einiger Zeit nimmt in diesem Experiment die Häufigkeit, mit der der Hebel bedient wird, jedoch relativ schnell ab (Abbildung 1:C). Dieses passiert, sobald eine Ratte gesättigt ist. Nimmt man die Ratte vorübergehend aus dem Käfig, wartet bis sie wieder hungrig ist und setzt sie dann wieder in den Käfig, fängt sie sogleich wieder an, sehr häufig den Hebel zu bedienen.

In der Sprache der Behavioristen wurde das Verhalten, das Treten des Hebels, durch eine Belohnung, also die Gabe von Futter, verstärkt. Dabei heißt Verstärkung, dass die Wahrscheinlichkeit steigt, dass ein bestimmtes Verhalten gezeigt wird.

Hierbei gibt es zwei Arten von Belohnungen: positive und negative. Positive Belohnung heißt, dass ein positiver Reiz (z.B. Futter) hinzugefügt wird, während eine negative Belohnung das Entfernen eines negativen Reizes (z.B. das Entfernen von Elektroschocks) ist. Wenn man die Wahrscheinlichkeit, dass ein Verhalten gezeigt wird, verringern will, muss man positiv oder negativ bestrafen. Positiv bestraft wird, wenn ein unangenehmer Reiz (z.B. ein Elektroschock) bei einem bestimmten Verhalten hinzugefügt wird. Negativ bestrafen heißt es, wenn ein angenehmer Reiz (z.B. Futter) entfernt wird.

Prof. Skinner, einer der führenden Behavioristen, fand heraus, dass sowohl ein kausaler als auch ein sehr enger zeitlicher Zusammenhang zwischen Verhalten und Belohnung wichtig ist.

Die Belohnung muss dabei sehr schnell auf das Verhalten folgen. Je länger der zeitliche Abstand ist, desto weniger wird das Verhalten verstärkt.

Der zweite wichtige Punkt ist der kausale Zusammenhang. Wenn die Belohnung teilweise auch dann erfolgt, wenn das gewünschte Verhalten nicht gezeigt wird, so wird nur sehr schwer oder gar nicht erkannt, dass die Belohnung und das Verhalten zusammengehören.

Des Weiteren muss in der ersten Phase, in der die Verbindung zwischen Verhalten und Belohnung gelernt werden muss, jedes Mal, wenn das Verhalten gezeigt wird, auch belohnt werden. Dieses wird „kontinuierlicher Verstärkungsplan“ genannt (siehe [24]). Wenn keine kontinuierliche Verstärkung erfolgt, wird die Verbindung geschwächt und bei mehrmaligem Ausbleiben der Belohnung sogar ganz „gelöscht“.

Erst wenn die Verbindung fest gelernt worden ist, kann man die Anzahl der Belohnungen verringern. Dieses muss allerdings schleichend geschehen, um die Verbindung nicht zu löschen. Wenn man die Anzahl der Belohnungen verringert, kann man nach verschiedenen Planarten verstärken. Hierbei gibt es Quotenpläne und Intervallpläne. Bei Quotenplänen wird belohnt, nach dem das gewünschte Verhalten eine bestimmte Anzahl Male gezeigt wurde. Intervallpläne sind solche, bei denen nach einer Belohnung eine bestimmte Zeit abgewartet wird und dann beim nächsten Zeigen des Verhaltens die Belohnung gegeben wird.

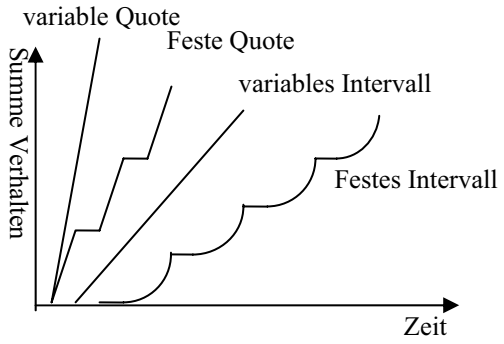

Abbildung 2: Kumulierte Häufigkeit von erwünschtem Verhalten bei verschiedenen Planungstypen bzw. Belohnungsmethoden

Des Weiteren kann man die Pläne fest oder variabel gestalten. Bei den festen Plänen erfolgt die Belohnung immer nach einer festen Anzahl bzw. Zeitspanne. Bei den variablen Plänen wird die Anzahl bzw. die Zeit, nach der das nächste Mal belohnt wird, jedes Mal zufällig (aber mit einem bestimmten Erwartungswert) festgelegt.

Hierbei wird das Verhalten bei Quotenplänen allgemein häufiger gezeigt als bei Intervallplänen. Des Weiteren sind variable Pläne besser als feste. Zusätzlich gibt es bei festen Plänen eine „Leistungseinbuße“ direkt nach einer Belohnung (siehe Abbildung 2). Allgemein erhält man noch durch stärkere Belohnungen auch ein häufigeres Zeigen des gewünschten Verhaltens.

Bestrafungen wirken im Allgemeinen schwächer als Belohnungen. Um überhaupt Resultate zu erzielen, muss bei jedem Auftreten des Verhaltens bestraft werden. Des Weiteren haben Bestrafungen keinen Einfluss auf die Verhaltensweisen, welche mittels Verstärkung schon gut gelernt wurde. Um also dafür zu sorgen, dass Verhalten nicht gezeigt wird, muss jedes Mal sehr hart bestraft werden und zwar bevor sich das entsprechende Verhalten gefestigt hat. Hierbei sei erwähnt, dass meistens Fehlverhalten positive Belohnungen nach sich zieht, wenn keine Bestrafung erfolgt. Beispielsweise erhält man beim Diebstahl das Diebesgut, ohne dafür zu bezahlen. Deswegen wird solch ein Verhalten relativ schnell gelernt.

Wenn man eine Ratte, die den Zusammenhang zwischen Hebel und Futter gelernt hat, in einen entsprechenden Käfig setzt, in dem das Bedienen des Hebels keine Futtergabe auslöst, so wird diese Ratte am Anfang das bekannte Verhalten zeigen. Dieses wird aber mit der Zeit abnehmen. Der Hebel wird also immer seltener bedient. Der Vorgang wird bei den Behavioristen „Extinktion“ genannt. Hierbei vergisst das Tier langsam den Zusammenhang von Futter und der Betätigung des Hebels. Nach einer Extinktionsphase muss dementsprechend die Verbindung zwischen Belohnung und Verhalten neu gelernt werden.

Dies fanden die Behavioristen zuerst bei Tieren heraus, übertrugen es aber dann auch auf Menschen. Die Prinzipien funktionieren dabei gleich. Zusätzlich kann man Menschen aber auch darauf hinweisen, was sie zu tun oder lassen haben, um belohnt zu werden.

Allerdings können durch Operante Konditionierung nur einfache Handlungsweisen gelernt werden, die nicht auf andere Aufgabengebiete übertragen und auch nicht verallgemeinert werden können. Da solche Lerninhalte in der heutigen sehr flexiblen Welt selten sind, ist die Anwendung der Operanten Konditionierung auf den Menschen umstritten. Sie ist trotzdem weit verbreitet, beispielsweise wird sie in der Schule zur Motivation von Schülern angewendet (siehe [26]).

Die Anforderungen dieser Arbeit an das Motivationssystem entsprechen den Vorraussetzungen, die für die Operante Konditionierung notwendig sind. Es werden nur einfache Handlungsweisen gefordert und die Lerninhalte müssen nicht verallgemeinert werden. Deswegen ist es in diesem Fall sinnvoll, das Wissen über den Ablauf der Operante Konditionierung anzuwenden.

II.5. Allgemeine Probleme von Peer-to-Peer-Gemeinden

Da die Gemeinde über keine zentralen Strukturen verfügt, muss das System vollständig dezentral aufgebaut sein. Solche Systeme, die nur aus verteilten und gleichberechtigten Teilen, so genannten Peers, bestehen, nennt man Peer-to-Peer-Netze. Deren allgemeine Hauptproblematik sind die hohe Fluktuation der Knoten durch den Ausfall und das Hinzukommen von Computern (siehe II.5.1), die Notwendigkeit sehr skalierbar zu sein (siehe II.5.2) und eine starke Heterogenität des Netzes (siehe II.5.3). Probleme, die durch das spezielle Design entstehen, werden unter III.2 diskutiert (siehe [15]).

II.5.1. Problem der Rechnerausfälle im Netz

In solch einem Peer-to-Peer–Netzwerk ist mit einer sehr hohen Dynamik zu rechnen, die eventuell durch die Regeln (siehe III.1) vermindert wird. Hierbei wird ständig ein relativ hoher Prozentsatz an Computern aus dem Netz genommen. Gleichzeitig verbindet sich eine große Anzahl Rechner ständig neu mit dem Netz. Dieses Verlassen und Eintreten in das Netz ist dabei pseudozufällig. Es ist für das Netz nicht erkennbar, welcher Computer als nächstes ausfällt oder zu welchem Zeitpunkt ein bestimmter dieses tut. Es lässt sich bestenfalls die Ausfallswahrscheinlichkeit eines Computers über einen bestimmten Zeitraum berechnen, wenn man zugrunde legt, dass der dahinter stehende Mensch sich so verhält wie ein durchschnittliches Mitglied.

Des Weiteren passieren diese Ausfälle ohne oder nur mit kurzer Vorwarnzeit, so dass man sich auf den Ausfall der entsprechenden Komponente nicht oder nur sehr schlecht vorbereiten kann. Zusätzlich ist die Zeit, die eine Komponente am Stück online ist und am Netzwerk teilnimmt, relativ kurz. Dieses Verhalten kann aber geändert werden (siehe II.4.5). Bei Gnutella beispielsweise liegt der Median dieser bei nur 60 Minuten (siehe [02]). Hierdurch wird auch die maximale Dauer einer Verbindung festgelegt. Je nach Design des Netzes kann die Verbindungsdauer aber auch erheblich niedriger liegen.

Diese Fluktuation bringt viele Probleme mit sich und muss bei fast allen Aspekten des Systems mit bedacht werden. So darf beispielsweise die Suchfunktion solch eines Netzes auf keinem Index basieren bzw. dieser muss ständig aktualisiert werden. Ein weiteres Beispiel ist das Problem, überhaupt Mitglieder des Netzes zu finden, bevor man sich mit dem Netz verbunden hat. Hierbei kann von keinem Rechner wirklich angenommen werden, dass dieser online ist.

II.5.2. Problem der Skalierbarkeit des Netzwerkes

Bei der Entwicklung eines Peer-to-Peer-Netzes und in der Anlaufphase wird dieses so gut wie keine Teilnehmer haben. Wenn allerdings die Umstände stimmen, kann es sein, dass innerhalb von kurzer Zeit die Teilnehmeranzahl drastisch steigt. So hat beispielsweise das Filesharing-Netzwerk Gnutella mehr als eine Million aktiver Teilnehmer, von denen zu jedem bestimmten Zeitpunkt ca. 10.000 online sind (siehe [02]). Andere Netze haben sogar noch mehr Teilnehmer, so hat Overnet 85.000 Teilnehmer, die online sind (siehe [34]). Dieses stellt starke Ansprüche an die Skalierbarkeit des Netzes, da dieses seine Funktionalität und seinen Ressourcenverbrauch nicht stark (maximal O(n)) ändern darf. Wenn dem nicht so wäre, würde das Netz ab einer bestimmten Größe nicht mehr praktikabel sein. Beispielsweise darf sich nicht die benötigte Rechenkapazität um Größenordnungen steigern, sonst würde der Größe des Netzes allein hierdurch eine Grenze gesetzt werden. Bei einem Wachstum der benötigten Ressourcen von O(n) hält sich das Wachstum der benötigten mit dem der im Netz zur Verfügung stehenden gerade die Wage.

Damit die Umstände günstig für solch eine Entwicklung sind, muss natürlich erst einmal die Idee stimmen und gut ausgearbeitet sein. Des Weiteren sollte die Umsetzung einfach, ansprechend und robust sein. Auch sollten keine Probleme bei einem rasanten Wachstum entstehen. Zusätzlich sollte der Nutzen leicht ersichtlich sein.

Zu guter Letzt sollte die Zeit für diese Idee auch gekommen sein. Wenn dem nicht so ist, also das entsprechende Thema noch nicht für die Gesellschaft von Interesse ist, so nützt häufig auch keine gute Idee. Deswegen ist auch eine gute Portion Glück und eine gute PR für einen Erfolg wichtig.

II.5.3. Technische Heterogenität des Netzes

Ein Peer-to-Peer-Netz muss außerdem noch mit einer hohen Heterogenität in allen Bereichen zurechtkommen. Dabei gibt es erst einmal solche, die durch die Benutzer ausgelöst werden. Womit man dabei rechnen muss, wurde in den vorherigen Abschnitten (II.4) diskutiert. Auf der technischen Seite können die bereitstehenden Hardwareressourcen um mehrere Größenordnungen differieren. Hierbei sind sowohl der bereitstehende Speicher als auch die Rechenkapazität zu nennen.

Ähnlich, wenn auch auf Grund der Vorraussetzungen (DSL) nicht so stark, können die Eigenschaften des Netzwerkes unterschiedlich sein. Sowohl die Übertragungszeit als auch die Übertragungsgeschwindigkeit sind hier die uneinheitlichen Größen.

Zuletzt sind auch das Betriebssystem und die dazugehörenden Bibliotheken mindestens in verschiedenen Versionen zu finden. Um eine möglichst große Verbreitung zu erreichen, können außerdem noch verschiedene Betriebssysteme unterstützt werden.

Kapitel III

Diskussion von Lösungsmöglichkeiten

III.1. Spielregeln des Abrechnungssystems

Eine Möglichkeit ist es, die Gemeinde vollkommen liberal (also ohne feste Regeln) aufzubauen und darauf zu vertrauen, dass sich jeder so stark einbringt, wie er kann. In existierenden Netzwerken sieht man allerdings, dass diese weit unterhalb ihres optimalen „Nutzens“ (siehe II.4.1) agieren (siehe Freerider bei Gnutella [02]). Aus diesem Grunde sind Regeln und Anreize zur Steuerung des Verhaltens der Mitglieder (siehe II.4.5) wichtig. Dementsprechend muss das Engagement der Mitglieder und deren Belohnungen abgerechnet werden. Dieses sollte am besten von einem Softwaresystem übernommen werden, damit keine unnötige Arbeit für die Mitglieder entsteht. Das Abrechnungssystem muss auch sicher sein, um einen Missbrauch (siehe II.4.3) zu verhindern.

III.1.1. Diskussion der Lösungsansätze von existierenden Netzwerken

Der Ansatz eines Peer-to-Peer-Netzwerkes, in dem der Internetzugang geteilt wird und welches Anreize für das Engagement der Mitglieder schafft, wurde bereits von Elias C. Efstathiou und George C. Polyzos durchdacht (siehe [14],[13]). Hierbei wurde aber als Zielgruppe, im Gegensatz zu dieser Arbeit, auch explizit kommerzielle Anbieter gesehen. Daraus resultieren Restriktionen bei dem Design des Netzes.

Wireless-Internet-Service-Provider (WISP) erhalten normalerweise Geld, wenn jemand über sie ins Internet geht. Der entsprechende Betrag errechnet sich unter anderem aus ihren eigenen (festen und laufenden) Kosten und einem Gewinn. Benutzt aber jemand ihren Zugang ohne Geld zu bezahlen, so entspricht dieses einem „Verlust“, der durch einen anderen entsprechend großen Nutzen ausgeglichen werden muss. Jemanden ohne Bezahlung ins Internet zu lassen, muss also belohnt werden.

Um ihren eigenen Kunden jederzeit eine Verbindung ins Internet zu ermöglichen, müssen WISPs das entsprechende Equipment während der Geschäftszeiten laufen lassen. Deswegen muss die Gemeinschaft dieses nicht noch zusätzlich belohnen.

Dementsprechend wurde in „A Peer-to-Peer Approach to Wireless LAN Roaming“ ([14]) auch ein System aus Anreizen gewählt, in dem Onlinezeit gegen Onlinezeit abgerechnet und getauscht wird. Der eigentliche Gewinn dabei ist eine erhöhte Abdeckung für jedes Mitglied.

Für Privatpersonen entstehen im Szenario dieser Diplomarbeit (Flatrate, keine Nachteile beim Surfen) keine oder nur ein minimaler Verlust (Stromkosten) durch das Weiterleiten der Daten von einem Fremden. Allerdings sind sie im Normalfall nicht immer „online“ und bereit als Provider zu fungieren. Deswegen benötigen Privatpersonen im

Gegensatz zu den kommerziellen Anbietern einen Anreiz, um „bereit zu sein“, aber keinen, um die Daten von einem Fremden zu übermitteln. Des Weiteren wird von den Privatpersonen auch nur ein kleiner Teil der ihnen zur Verfügung stehenden Ressourcen (beispielsweise Bandbreite) genutzt (siehe [14]).

Der Anreiz bei direktem Tausch lässt sich für Privatpersonen am ehesten als variabler Intervallplan modellieren. Zu jeder Zeiteinheit kommt mit einer bestimmten Wahrscheinlichkeit ein Mitglied der Gemeinde vorbei, um zu surfen. Die Anzahl der Belohnungen hängt also nicht direkt davon ab, wie stark sich jemand engagiert, sondern von der Häufigkeit, mit der andere Mitglieder „vorbeikommen“, um zu surfen. Dieses entspricht einer bestimmten Zeitspanne. Zusätzlich ist der Abstand zwischen zwei „Belohnungen“ variabel. Diese Verstärkung entspricht also in der Sprache der Behavioristen (siehe II.4.5) einem variablen Intervallplan.

Solch ein Plan ist für die Anlernphase, also wenn jemand der Gemeinde beitritt, schlecht. In dieser Phase ist kontinuierliches Verstärken sehr wichtig. Des Weiteren sind Intervallpläne auch im späteren Verlauf nicht optimal zur Verstärkung des Verhaltens geeignet (siehe II.4.5).

Ein weiterer Nachteil der direkten Abrechnung und des direkten Tausches bzw. der Integration der kommerziellen Anbieter ist es, dass die Belohnung relativ gering ausfallen muss. Nimmt man Privatpersonen als Hauptzielgruppe, so kann man auf die geringen Kosten und die überschüssigen Kapazitäten zurückgreifen, um mehr zu belohnen. Dieses steigert auch das Engagement (siehe II.4.5).

Zusätzlich sind kommerzielle Anbieter meistens an stark frequentierten Plätzen angesiedelt (siehe [13]). Privatpersonen sind dieses im Allgemeinen nicht. Deswegen werden kommerzielle Anbieter stärker belohnt als Privatpersonen. Die Belohnungen für Letztere können dabei auf extrem geringe Werte fallen. Ein Beispiel hierfür wären Privatpersonen, die auf dem Land leben. Dass Privatpersonen im Allgemeinen weniger belohnt werden, verringert deren durchschnittliches Engagement. Zusätzlich entspricht die Abhängigkeit der Stärke der Belohnung vom Wohnort auch einer Abhängigkeit von Faktoren, die nichts mit dem Verhalten zu tun haben. Dieses wird als ungerecht empfunden und verringert deswegen den Nutzen für Privatpersonen (siehe II.4.4) weiter, während es für kommerzielle Anbieter, die rein wirtschaftlich handeln, unbedeutend ist. Wenn der Nutzen zu gering ist, nehmen die entsprechenden Personen nicht mehr am Netz teil (siehe II.4.3). Die Abhängigkeit der Belohnungshöhe von dem Ort führt also dazu, dass der Internetzugang nur auf die Ballungszentren konzentriert sein wird. Mit der Entfernung zu den Ballungszentren nimmt ja auch die Anzahl derjenigen ab, die surfen wollen. Dementsprechend nimmt auch die Anzahl an Belohnungen ab. Ab einem bestimmten Schwellenwert wird zu wenig belohnt, um das gewollte Verhalten zu erzielen.

Aus allen diesen Gründen ist der Ansatz von Elias C. Efstathiou und George C. Polyzos ([13],[14]) nicht ideal für Privatpersonen geeignet.

III.1.2. Sicherheit vor Betrug im Belohnungssystem

Unter dem Grad der Sicherheit des Abrechnungssystems ist hier die prozentuale Anzahl an Betrügereien (siehe II.4.3) zu verstehen, die begangen werden. Diese soll gering gehalten werden. Es gibt zwei grundsätzliche Arten diese gering zu halten:

1. Überprüfungen,
2. Nutzlosigkeit von Betrügereien.

Die Werte, welche abgerechnet werden (die Abrechnungsgrößen), lassen sich unterschiedlich gut überprüfen. Beispielsweise lässt sich sehr gut kontrollieren, ob jemand online ist oder nicht. Hierzu muss man einfach eine Reihe von Anfragen über das Internet an ihn senden. Wenn er antwortet, so ist er online, wenn er auf keine der Anfragen antwortet, so

ist dieses bestenfalls fraglich. Eine entsprechende Überprüfung ist auch sehr leicht zu automatisieren.

Sehr schwer ist hingegen zu überprüfen, ob jemand einen W-LAN-Anschluss hat und über diesen Internetzugang gewährt. Hierbei kann man sich entweder auf die Aussage des zu Überprüfenden verlassen oder eine vertrauenswürdige Person muss dieses kontrollieren. Im letzteren Fall muss sich diese Person in die Region begeben, in der das Internet angeboten wird. Dieses kann teilweise erhebliche Kosten verursachen. Des Weiteren existiert keine per se vertrauenswürdige Person in dieser Gemeinde, weswegen man die Prüfer zufällig auswählen sollte, was wenigstens ein gewisses Maß an Vertrauenswürdigkeit herstellt.

Bei der „Nutzlosigkeit von Betrügereien“ sieht die Situation anders aus. Hierbei wird davon ausgegangen, dass Betrug nur dann begangen wird, wenn für den Betrüger ein Gewinn erzielt werden kann. Dieses entspricht dem Standardmodell des Menschen (siehe [02]).

Wenn das Abrechnungssystem aber so gestaltet ist, dass derjenige, der versucht zu betrügen, keinen Gewinn oder sogar einen Verlust hierdurch erwirtschaftet, wird es so gut wie keine Betrugsfälle geben. Wenn zusätzlich weder der Gemeinschaft noch einem ehrlichen Mitglied ein Schaden durch einen Betrug entsteht, ist es außerdem noch unbedeutsam, ob betrogen wird oder nicht, da sich dieses nur auf den Betrüger selbst und dort negativ auswirkt.

Dieses erreicht man zum Beispiel durch eine besondere Abrechnung der übertragenen Bytes. Hierbei erhält man nur das Recht, z.B. halb so viele Bytes über Gemeindemitglieder zu übertragen, wie man selber für andere übertragen hat.

In diesem Fall ist es ziemlich nutzlos, zwei Konten anzumelden und diese abwechselnd übereinander „surfen“ zu lassen, ohne dabei eine echte Leistung zu erbringen. Hierbei würde sich die Anzahl der Bytes, die übertragen werden dürfen, ja verringern. Wenn man für das Übertragen von Bytes aber mehr gutgeschrieben bekommen würde, als man tatsächlich überträgt, wäre dieses ein erfolgreiches Betrugsszenario.

III.1.3. Diskussion der Methoden der Abrechnung

Es gibt zwei Modelle, wie die Abrechnung erfolgen kann. Diese lassen sich allerdings in ihren Extremfällen ineinander überführen. Man kann entweder „Punkte“ als Währung betrachten, so dass für Leistungen bezahlt werden muss, oder als Zeichen des Ranges, nach dem der Service bemessen wird.

III.1.3.1. Währungsmodell

Werden die Punkte wie Geld gehandelt, so erhält man Punkte für bestimmte Leistungen, die man im Auftrage von jemand anderem erbringt. Wenn man selber Leistungen von anderen in Anspruch nimmt, so gibt man Punkte an diesen weiter, man bezahlt mit den Punkten. Dabei kann man selber natürlich nur so lange Leistungen in Anspruch nehmen, wie man genügend Punkte besitzt, um die Leistung zu bezahlen.

Hierbei ist es möglich, einen Faktor zwischen Bezahltem und Erhaltenem einzubauen. Bezahlt jemand eine bestimmte Anzahl an Punkten, so erhält der, der bezahlt wird, diese Anzahl an Punkten um den Faktor verändert. Dabei kann der Faktor sowohl größer als eins als auch kleiner als eins sein.

Dieses Modell bietet dabei einen relativ starken Anreiz, sich nicht betrügen zu lassen, also jede Transaktion zu kontrollieren. Wenn man betrogen wird, verliert man ja an potentieller Leistung. Die Stärke dieses Anreizes schwankt dabei mit der Größe des Faktors. Dieses ist aber nicht nur Anreiz, sondern fordert auch ein sehr sicheres Abrechnungssystem.

Außerdem gibt es hierbei das Problem, dass die Summe aller zur Verfügung stehenden Punkte sinken kann. Dadurch würde auch die durchschnittliche Punktzahl sinken, die jeder besitzt. Hieraus würde wiederum folgern, dass die Summe der Leistungen sinkt, die in

Anspruch genommen werden können. Dieses entspräche wiederum einer sinkenden Nützlichkeit der Gemeinde für jedes einzelne Mitglied. Entsprechende Probleme gibt es auch bei realen Währungen, wenn die Summe des im Umlauf befindlichen Geldes sinkt. Hier wird das Phänomen „Deflation“ genannt.

Der „Verlust“ an Punkten kann an mehreren Stellen entstehen. Eine Möglichkeit ist, dass der Faktor zu gering ist, beispielsweise kleiner als eins. Hierdurch nimmt bei jeder Transaktion die Anzahl der Punkte ab. Aber auch wenn der Faktor größer als eins ist, kann die Anzahl der Punkte abnehmen. Ein Beispiel hierfür ist das „Stilllegen“ eines Kontos. Wenn ein Mitglied mit einer bestimmten Anzahl an Punkten zeitweise oder dauerhaft nicht mehr am Austausch teilnimmt, so entzieht er der Gemeinde diese Punkte für den entsprechenden Zeitraum.

Das gleiche Problem entsteht, wenn jemand (um den Faktor bereinigt) mehr Punkte einnimmt, als er ausgibt. Dieser Fall ist eigentlich in solch einer Gemeinde erwünscht, da dadurch der Gemeinde mehr Nutzen entsteht, als Leistungen von der Gemeinde gefordert werden. In diesem Fall vermindert es jedoch die im Umlauf befindlichen Punkte mit den oben beschriebenen Folgen.

Andererseits kann aber bei einem zu großen Faktor die Anzahl der Punkte beständig wachsen. Hierbei droht die Gefahr, dass die Gemeinde sich „überschuldet“. Jeder Punkt ist ja das Versprechen, eine bestimmte Leistung in Anspruch nehmen zu dürfen. Diese muss aber auch von der Gemeinde erbracht werden können. Da man nicht davon ausgehen kann, dass die Mitglieder altruistisch sind (siehe II.4.3), werden diese, wenn sie genügend Anspruch auf Leistungen haben, sich nicht mehr weiter einbringen. Durch dieses Verhalten können aber weniger Leistungen von der Gemeinde erbracht werden. Die Versprechungen können also nicht mehr „eingelöst werden“. Dabei sinkt aber auch der Nutzen, den die Gemeinde bietet, genau so, als wenn die Punkteanzahl sinken würde. Das Entsprechende heißt bei realen Währungen „Inflation“.

Um die Konsequenzen von Inflation und Deflation zu vermeiden, kontrolliert die entsprechende Zentralbank einer Währung die Geldmenge, die sich im Umlauf befindet, sehr genau. Entsprechendes müsste auch im Peer-to-Peer-Netzwerk stattfinden. Dafür müssten Punkte zu bestimmten Zeitpunkten neu erstellt und dann verteilt werden, so dass die Menge der im Umlauf befindlichen Punkte konstant bleibt. Hierbei gibt es aber erst einmal das Problem, dass diejenigen, die die Punkte erstellen, dieses nicht eigennützig tun. Da man in einem Peer-to-Peer-Netzwerk nicht davon ausgehen kann, dass man einer Person trauen kann, muss eine möglichst große und möglichst zufällig zusammengestellte Gruppe dieses übernehmen. Idealerweise könnten Punkte nur von allen Mitgliedern eines Netzwerkes gemeinsam erstellt werden.

Das zweite Problem ist die Verteilung der Punkte, da dieses gerecht ablaufen sollte. Wenn sie gleichmäßig auf die Konten verteilt werden, so lässt sich dieses von einer Person ausnutzen, um mehr zu erhalten, als ihr normalerweise zustehen würde. Hierzu muss diese Person einfach mehrere Konten kreieren oder unter ihre Kontrolle bringen. Dies könnte sie dadurch erreichen, dass sie beispielsweise neue Konten von Familienmitgliedern oder Freunden erstellen lässt, die eigentlich kein Interesse an der Gemeinde haben.

Eine weitere Möglichkeit ist es, die Punkte nach der Punktanzahl zu verteilen, die jemand besitzt. Hierdurch wird ein steigender Teil der Punkte an „tote“ Konten vergeben. Es ist davon auszugehen, dass auf solchen Konten, die nicht mehr benutzt werden, noch Restpunkte zu finden sind. Deswegen würde auch ein Teil der neuen Punkte auf solche Konten verteilt werden. Dadurch erhöht sich die Anzahl der Punkte stetig, die sich auf toten Konten befinden. Es wird aber gleichzeitig die Anzahl der sich im Umlauf befindlichen Punkte gleichgehalten (siehe oben), um Prozesse zu vermeiden, die einer Inflation oder Deflation ähneln. Deswegen erhöht sich der Anteil der Punkte aus „toten“ Konten im Vergleich zur Gesamtpunktzahl, was wiederum zu einem stärkeren Wachstum dieses Anteils

sorgt. Dieses hat zwei Nachteile. Erstens erhöht sich die Anzahl der zu generierenden Punkte und damit auch der dafür zu betreibende Aufwand ohne direkte Notwendigkeit. Der zweite und gewichtigere Nachteil ist, dass Personen, die sich eine Zeit lang nicht engagieren, trotzdem belohnt werden. Dieses verringert den Anreiz sich zu engagieren.

Dagegen könnte man die Punktevergabe auf jenem Engagement (eventuell um die Nutzung der Gemeinde bereinigt) basieren lassen, welches seit dem letzten Punkte Generieren gezeigt wurde. Dieses ist aber nur eine indirekte Erhöhung des Faktors.

III.1.3.2. Rangmodell

In dem anderen Modell werden die Punkte nur als eine Art „Rangzeichen" betrachtet. Hierbei erhält man zwar für eine erbrachte Leistung Punkte, muss aber für das in Anspruchnehmen von einer Leistung keine Punkte bezahlen. Die Punkte regeln hierbei nur, ob, in welchem Umfang oder in welcher Qualität eine Leistung bezogen werden darf. Die genaue Bemessung des Leistungsumfanges wird dabei gesondert abgerechnet.

Dadurch, dass das Erbringen von Leistungen und deren Inanspruchnahme nur noch indirekt gekoppelt sind, gibt es viel mehr Freiheiten in deren Zusammenspiel. Denkbar sind beispielsweise nicht lineare Zusammenhänge zwischen diesen Größen. Man kann aber auch als Belohnung zeitbasierte Leistungsmengen (5h/Tag) oder die Qualität der Leistungen festlegen, ohne deren Quantität zu ändern. Diese Freiräume kann man zur Verwirklichung von Zielen nutzen.

Wenn man ein nicht absolutes Modell der Nutzenabrechnung benutzt, so ist es ratsam, auch das Engagement nicht absolut zu rechnen. Hierbei ist mit „absolut" gemeint, dass sich die entsprechende Größe endlos, ohne verringert zu werden, aufaddiert. Ein Beispiel für eine absolute Größe wäre die gesamte akkumulierte Onlinezeit eines Benutzers. Bei nicht absoluter Rechnung werden die Größen für die bezogene oder die erbrachte Leistung in Relation zu einer anderen Größe, z.B. der Zeit, gesetzt. Beispielsweise könnte man die Anzahl der erhalten Punkte pro Monat wählen.

Misst man das Engagement absolut und den Nutzen nicht absolut, so bietet dieses keinen nachhaltigen Anreiz zum Engagieren. Wenn eine bestimmte „Rangstufe" durch Engagement erreicht wurde, so ist das Ziel des Benutzers erreicht. Hiernach benötigt er keine weiteren Belohnungen und engagiert sich deswegen auch nicht weiter, da er keinen Nutzen mehr aus den Belohnungen und damit auch nicht aus seinem Arbeitseinsatz ziehen kann.

Ist es aber andersherum, wird also das Engagement nicht absolut gemessen, aber die Nutzung absolut, so muss der Benutzer sich immer mehr engagieren, um sein Ziel zu erreichen. Dieses entsteht, da normalerweise Ziele in Relation zu der Zeit stehen. Ein Benutzer will beispielsweise jeden Tag seine Emails abrufen. Dieses verbraucht jeden Tag geringe Mengen an Nutzenspunkten. Die gebrauchte Nutzensmöglichkeit akkumuliert sich jedoch über die Zeit zu einer Nutzmenge, die größer ist als die, die einem zusteht, wenn man maximal engagiert ist.

Bei einer absoluten Rechnung beider Größen ist außerdem die Gefahr einer Inflation relativ groß. Letztere entsteht, wenn dauerhaft mehr Nutzen erwirtschaftet als in Anspruch genommen wird. Dadurch erhöht sich der zugesprochene Nutzen ständig. Bei nicht absoluten Systemen ist diese Gefahr nicht sehr hoch, da nur das Engagement über einen bestimmten Zeitraum gerechnet wird und so eine Obergrenze besitzt.

Dabei ist zu beachten, dass Gerechtigkeit und Stärke des Anreizes sich hierbei nur im Einzelfall prüfen lassen.

Auf der anderen Seite entsteht der Nachteil, dass ein nicht so großes Interesse besteht, Betrügereien zu verhindern. Man erleidet ja keinen direkten Verlust, wenn man in einen Betrugsfall verwickelt wird.

Dieses Modell zeichnet sich also durch viele Freiheitsgrade aus und sollte deswegen gewählt werden, wenn man diese zur Verwirklichung von Zielen nutzen kann (genauere Begründung unter III.1.5).

III.1.4. Diskussion von Abrechnungsgrößen

Es gibt zwei grundlegende Modelle, welche Größen abgerechnet werden können. Man kann entweder den eigentlichen Nutzen oder aber sekundäre Leistungen als Basis heranziehen.

III.1.4.1. Abrechnung des eigentlichen Nutzens

In diesem Fall ist der eigentliche Nutzen, dass jemand über eine andere Person Zugang zum Internet erhält. Hierbei kann man verschiedenste Größen benutzen. Man könnte die Anzahl an Menschen, die über einen Zugang surfen, die akkumulierte Zeit, die diese Leute im Internet verweilen, oder die Anzahl der Bytes, die hierbei ausgetauscht werden, verwenden.

Diese Leistungen können praktisch nicht von der Gemeinde überprüft werden. Theoretisch gibt es aber zwei Möglichkeiten, solche Leistungen zu überprüfen. Die erste Möglichkeit zur Überprüfung wäre die genaue Analyse des Systems aus Hardware und Software von dem Paar aus Leistungsnehmer und Leistungsgeber. Hierbei müsste die korrekte Funktion des Systems zeitnah zu der Leistung verifiziert werden, da sonst die Möglichkeit besteht, das System wieder in den Sollzustand zurückzuversetzen. Diese Analyse kann aber nur vor Ort geschehen, da sonst die Möglichkeit besteht, dass das System korrektes Verhalten sowohl der Hardware als auch der Software für die entsprechenden Zugriffskanäle simuliert.

Die zweite Möglichkeit wäre es, zusätzliche Hardware zu installieren, die nur von vertrauenswürdigen Personen verändert und installiert werden kann. Der Leistungsnehmer und der Leistungsgeber dürfen dabei keine Möglichkeit haben, die Funktionsweise der Geräte zu verändern. Diese müssten die entsprechenden Daten dann bestätigen. Aber auch hier ist es wichtig, dass eine vertrauenswürdige Person die entsprechenden Geräte kontrolliert, damit das Verhalten der Geräte beispielsweise nicht von einer Software simuliert oder deren Einbau geändert wird. Dass es nur wenig praktikabel ist, vertrauenswürdige Personen vor Ort haben zu müssen, wurde schon in III.1.2 besprochen.

Ohne solche Sicherheitsmaßnahmen ist man auf die Aussagen der Beteiligten angewiesen, dass die Daten korrekt sind.

Ein sehr einfacher „Angriff“ auf solch ein ungeschütztes Abrechnungssystem wäre es, wenn sich jemand mehrfach anmelden würde. Dann wäre es möglich, die Konten „übereinander surfen“ zu lassen, ohne dass überhaupt eine Leistung erbracht wird. So ist der Angriff von einem Einzeltäter ausführbar (siehe II.4.3.1). Selbst wenn die Gemeinde sicherstellt, dass jede Person nur ein Konto haben kann, ist der gleiche „Angriff“ auch mit Hilfe von Freunden möglich, was dem einer Gruppe (siehe II.4.3.2) entsprechen würde. Dementsprechend ist die erste Möglichkeit (Kontrolle) Betrug auszuschließen nicht praktikabel. Man muss also auf die zweite Möglichkeit (Nutzlosigkeit von Betrügereien) zurückgreifen (siehe III.1.2). Dieses heißt, dass ein dauerhafter Nachteil dem Leistungsnehmer durch das Surfen entstehen muss, der mindestens so groß ist wie der Vorteil, der dem Nutzen des Leistungsgebers entspricht. Ist dem nicht so, hat das Paar aus Surfer und Provider einen Nettogewinn erwirtschaftet, wenn sie vortäuschen, „übereinander zu surfen“. Ein System, das Betrügereien auf so einfache Art und Weise ermöglicht, kann nicht mehr die Aufgabe erfüllen, durch Anreize (siehe II.4.5) den Nutzen zu maximieren (siehe II.4.1) und kann sehr leicht als ungerecht (II.4.4) angesehen werden. Aus diesem Grund können nur das „Währungsmodell“ oder „Rangmodelle“, die eng verwandt mit dem Währungsmodell sind, für die Abrechnung gewählt werden. Bei dem Währungsmodell muss der Faktor (vgl.

III.1.3.1, III.1.3) außerdem noch kleiner oder gleich eins sein. Dabei muss die Generierung von Punkten, so sie nach der Höhe des Engagements vergeben werden, in die Berechnung des Faktors einbezogen werden. Die Konsequenzen eines solchen Systems wurden unter III.1.3.1 beschrieben.

Aber auch wenn das Währungsmodell gewählt wird, gibt es Probleme. Diese wurden unter „Diskussion der Lösungsansätze von existierenden Netzwerken“ (III.1.1) erörtert.

III.1.4.2. Abrechnung sekundärer Leistungen

Sekundäre Leistungen sind Leistungen, die zwar erbracht werden müssen, aber nicht das Surfen selbst betreffen. Möglichkeiten hierfür sind, ob jemand Benutzerkonten verwaltet oder wie lange jemand online ist. Diese Leistungen lassen sich sehr gut überprüfen, da sie automatisch kontrolliert werden können. Des Weiteren entsprechen sie dem, was bei Privatpersonen verstärkt werden muss. Außerdem messen sie das Verhalten selbst, weswegen mit ihnen eine gerechte Verteilung der Belohnungen möglich ist. Hierbei kann jedoch nur in Einzelfällen das Währungssystem eingesetzt werden, da häufig keine benennbare Person existiert, die diese Leistungen in Auftrag gibt. Die Leistungen sind für die Gemeinschaft zwar nötig, aber keinem spezifischen Benutzer zuzuordnen. Ein Beispiel für solch eine Leistung wäre die Zeit, die jemand am Peer-to-Peer-Netz partizipiert. Dieses hat keinen eigentlichen Nutznießer und kann maximal auf den nächsten Fremdbenutzter umgelegt werden. Das Umlegen entspricht aber wieder der Abrechnung des eigentlichen Nutzens. Ein Beispiel, wo das Währungsmodell sinnvoll wäre, ist die Abrechnung, ob und wie viele Benutzerkonten jemand verwaltet. Hierfür könnte man eine Kontoführungsgebühr einführen.

Ein schwerwiegendes Problem ist allerdings, dass diese Leistungen nicht unbedingt mit dem eigentlichen Nutzen korreliert sein müssen. Wenn jemand online ist, muss er nicht gleichzeitig bereit sein, als Provider zu fungieren. Selbst wenn alle nötigen Programme laufen, kann der entsprechende W-LAN-Access-Point offline sein oder, was noch schwerer zu überprüfen ist, manipuliert sein, so dass niemand über diesen surfen kann. Allerdings entsteht auch kein großer Nutzen dadurch, dass man nicht als Provider fungiert. W-LAN Equipment muss man auf jeden Fall besitzen, um Nutzen aus der Gemeinde ziehen zu können. Dieses kann man auch dazu verwenden, den Zugang zum Internet über sich zu ermöglichen. Bei einer Flatrate oder zeitbasierten Internettarifen entstehen die gleichen Kosten, ob man nun als Provider dient oder nicht. Wenn man betrügt, fällt allerdings ein Übertragungsvolumen weg. Dieses kann auf zwei Arten ein Vorteil für einen sein. Einerseits kann man immer den gesamten Anschluss für sich nutzen und hat so nie Performanceeinbußen, dadurch dass jemand anderes den Anschluss mitbenutzt. Andererseits kann man eventuell auch die Kosten einer Flatrate umgehen und einen Übertragungsvolumen basierten Tarif benutzen. Diese beiden Vorteile kann man zunichte machen, indem ständig größere Mengen an Bytes zwischen unbeschäftigten Teilnehmern hin und her geschickt werden. Diese Down- bzw. Uploadlast sollte im Idealfall genauso groß sein wie die, die durch eine Fremdbenutzung entsteht. Um den zweiten Vorteil zunichte zu machen, genügt allerdings schon eine erheblich geringere Last, da sich über die Zeit schnell ein großes Byte-Volumen akkumuliert, welches teurer wäre als eine Flatrate. Bei dieser Maßnahme bleibt ein letzter Nutzen von regelwidrigem Verhalten, nämlich eine Vereinfachung der Einrichtung der entsprechenden Dienste. Diesem kann man durch eine größtmögliche Automatisierung der Installation entgegenwirken. Da allerdings die Anbindung des W-LAN-Equipments an den Computer, auf dem die Dienste laufen, beliebig ist und mehrere Computer dazwischen geschaltet sein können, ist dieses nicht immer vollständig machbar.

Das Abrechnen von sowohl der eigentlichen Leistung als auch der sekundären Leistungen ist nur sehr bedingt sachdienlich. Die Nachteile der einzelnen Systeme lassen sich

nicht durch Einführung des anderen Systems kompensieren. Stattdessen bekommt man ein System, welches sowohl die Nachteile des einen als auch die des anderen aufweist.

Um den primären Nutzen vernünftig abrechnen zu können, muss ein währungsähnliches System mit weiteren Restriktionen benutzt werden. Dementsprechend muss dieses mit all seinen Nachteilen auch im kombinierten System gewählt werden. Deswegen sind die Möglichkeiten, die bei der Belohnung von sekundären Leistungen benutzt werden können, stark begrenzt. Andererseits verringert sich aber auch nicht der Nachteil, der durch Inflation und Deflation entsteht. Es können zwar Punkte durch sekundäre Leistungen entstehen, aber die Menge solcher Punkte ist nicht oder nur schwer vorhersehbar. Deswegen müssen weitere Mechanismen benutzt werden, die die Menge der im Umlauf befindlichen Punkte steuern.

Eine erhöhte Korrelation zwischen eigentlichem Nutzen und der Punktanzahl ist auch nur dann gegeben, wenn die Menge an Punkten, die durch primäre Leistungen erwirtschaftet werden, die Anzahl der Punkte, die man für sekundäre Leistungen erhält, deutlich übersteigt. Dieses entspricht aber prinzipiell einer reinen Abrechnung des eigentlichen Nutzens.

III.1.5. Entscheidung

Die Hauptzielgruppe sind hier Privatpersonen. Die Zeit, in der jemand bereit ist, anderen über seinen Anschluss Zugang zum Internet zu gewähren, entspricht am ehesten dem Aufwand, den eine Privatperson zeigen muss (siehe III.1.1). Deswegen wird diese als Abrechnungsgröße des Engagements gewählt (siehe III.1.4.2).

Um den aus der Sicht des Netzes nicht gewünschten Vorteil einer Person zu minimieren, den diese hat, wenn sie keine Flatrate stellt, wird das System außerdem eine geringe Grundlast (down- und upload) von wenigen Kilobytes erzeugen (siehe III.1.4).

Das Rangmodell hat im Gegensatz zum Währungsmodell viel mehr Freiheitsgrade im Zusammenspiel zwischen Engagement und Belohnung. Um die Belohnungen möglichst optimal (siehe II.4.5) zur Nutzenmaximierung (siehe II.4.1) einsetzen zu können, werden diese Freiheitsgrade gebraucht und deswegen wird das Rangmodell eingesetzt.

Hierbei werden beide Größen (Engagement und Nutzen) nicht absolut gerechnet, sondern in Relation zu der Zeit, der Einfachheit halber jeweils über ein Zeitintervall. Diese müssen dabei nicht gleich groß sein.

III.1.5.1. Abrechnung des Engagements

Durch die nicht absolute Rechnung des Engagements wird das aktuelle Verhalten bewertet. Deswegen kann man sich nicht beliebig lange auf einem alten hohen Engagement „ausruhen“. Hierbei sollte aber das Intervall, über welches das Engagement abgerechnet wird, groß genug sein, um kurzfristige Fluktuationen auszugleichen. Des Weiteren sollte auch während mittelfristiger Inaktivitäten, wie Störungen bei den benötigten technischen Geräten oder einem Urlaub, die Gemeinde in Anspruch genommen werden können. Um auch während eines Urlaubes über die Gemeinde ins Internet gelangen zu können, sollte der Abrechnungszeitraum erheblich größer sein, als dieser dauern könnte. Dieses wird dadurch begründet, dass die Gemeinde solche Vorkommnisse nicht vom vorsätzlichen Nicht-Engagement unterscheiden kann und damit auch beide Fälle gleich behandeln muss. Deswegen verringert sich der Zahlenwert stark, der das Engagement repräsentiert, wenn die Inaktivitätsperiode lang im Verhältnis zum Abrechnungszeitraum ist. Ist beispielsweise das Abrechnungsintervall 12 Tage und fährt jemand neun Tage in den Urlaub, so beträgt der Zahlenwert seines Engagements nur noch ¼ des Wertes, den er vor dem Urlaub hatte. Bei einer nichtlinearen Umrechnung des Engagements in den Nutzen kann sich der zugesprochene Nutzen über den Urlaub entweder mehr oder weniger stark verringern. Dieses passiert je

nachdem wie die Zuordnung von Engagement zu Belohnung aussieht. Wenn das Wachstum der Belohnungs- /Engagementfunktion stärker als linear ist, so sinkt der Nutzen schneller als hier beschrieben. Ist das Wachstum jedoch schwächer, so sinkt der Nutzen langsamer.

Hier wäre ein möglicher Wert für die Abrechnung ein Zeitraum von drei Monaten. Hierbei wird maximal von einem vier bis sechs Wochen dauernden Urlaub ausgegangen. Dieses entspricht dadurch 1/3 bis maximal ½ des Abrechnungszeitraumes. Deswegen sinkt der Zahlenwert des Engagements höchstens auf die Hälfte, weswegen noch ein genügender Service angeboten werden dürfte.

III.1.5.2. Abrechnung des Nutzens

Die zu der Zeit relative Rechnung des Nutzens hat andere Vorteile als die relative Abrechnung des Engagements. Einer davon ist, dass eine Sättigung der Bedürfnisse erst später als bei einer absoluten Nutzenrechnung eintritt. Dieses entsteht, da Belohnungen innerhalb des entsprechenden Intervalls ausgegeben werden müssen und sonst verfallen. Deswegen muss ein Mitglied soviel erwirtschaften, wie es maximal und nicht durchschnittlich in einem Zeitintervall benötigt. Der Unterschied zwischen dem Durchschnitt und dem Maximalwert verfällt dabei im Normalfall. Je länger hierbei der Zeitraum ist, über den abgerechnet wird, desto ähnlicher ist das System einer absoluten Abrechnung. Dementsprechend nähern sich der Wert des Maximums und der des Durchschnittes an. Andererseits könnte es Akzeptanzprobleme geben, wenn das Intervall zu kurz wird. Dieses ist dadurch zu begründen, dass der Durchschnitt und das Maximum zu weit differieren. Dadurch verfällt sehr viel Nutzen und die nominale Belohnung hat mit der echten Belohnung, also der Zeit, die man im Internet ist, nur noch wenig zu tun.

Um diesen Zusammenhang zu verdeutlichen, sei hier ein Extrembeispiel genannt. Wenn das Intervall für die Belohnungen 10min ist und man beispielsweise 2min (pro 10min) zugesprochen bekommt, so kann man theoretisch 4,8 Stunden pro Tag im Internet surfen. Wenn man aber einen Zugang zum Internet haben will, muss man nach nur 2min Aktivität erst einmal 8min warten, bevor man die nächste Aktivität beginnen kann. Für ein normales „Internetvergnügen“ braucht man hier die maximale Belohnung. Solche Effekte treten bei Intervallen auf, die höchstens im Stundenbereich liegen.

Aber auch wenn das Intervall im Tagesbereich liegt, gibt es noch Einschränkungen, auch wenn diese schon erheblich weniger dramatisch sind als die im Minuten- oder Stundenbereich. Diesem liegt die Annahme zugrunde, dass der Tagesablauf über die Woche differiert und damit auch höchst wahrscheinlich das Konsumverhalten unterschiedlich ausfällt. Eine solche Differenz ist beispielsweise der Unterschied zwischen einem Werktag und dem Wochenende. Man nehme also an, jemand surft nur an Wochenenden. Solch ein Mensch würde bei einer Abrechnung in Tagesschritten 5/7 seiner Belohnung verfallen lassen müssen. Aus diesen Gründen muss ein Kompromiss über die Länge des Intervalls gefunden werden. Hierfür wird vom Autor der Wert von einer Woche favorisiert. Durch die gesetzliche Festlegung des Sonntags als normalerweise freien Tag ergibt sich für die meisten Menschen ein einwöchiger Rhythmus. Für Schichtarbeiter, die jede Woche in einer anderen Schicht arbeiten, gilt dieses natürlich nur bedingt.

III.1.5.3. Ausformung der Anreize

Für die Vergabe der Belohnungen gibt das Kapitel II.4.5 viele Hinweise und Anregungen. Der Behaviorismus schreibt eine kontinuierliche und starke Belohnung vor, wenn die Bindung zwischen Verhalten und Belohnung noch nicht stabil besteht. Des Weiteren wird vorgeschlagen, mit steigender Stabilität der Bindung langsam den kontinuierlichen in einen variablen Quotenplan zu überführen. Hierbei kann auch die Stärke der Belohnung

reduziert werden. Nach dem Sättigungspunkt sollte die Stärke der Belohnung aber wieder steigen.

Eine schwache Bindung ist natürlich beim Eintritt in die Gemeinde gegeben. Sie kann aber auch nach Abwesenheit oder sonstigen Störungen entstehen. Eine schwache Bindung sollte laut der Behavioristen auch ein seltenes Zeigen des entsprechenden Verhaltens nach sich ziehen. Deswegen kann das Engagement (also die Zeit, die jemand online war) als Indikator für die Stärke der Bindung genutzt werden.

In der Realisierung wird die Berechnung der Belohnung in zwei Schritte unterteilt. Der erste ist die Berechnung des Engagements und der zweite ist die Berechnung des Nutzens. Im ersten Schritt wird ein variabler Quotenplan realisiert, während im zweiten die eigentliche Belohnung aus dem Engagement berechnet wird. Bei der Berechnung der Belohnung wird das Engagement zugrunde gelegt und die Stärke der eigentlichen Belohnungen so angepasst, wie der Behaviorismus es vorschlägt.

III.1.5.3.1. Berechnung des Engagements

Die Berechnung des Engagements erfolgt schrittweise, so dass man zwischen dem eigentlichen und dem gutgeschriebenen Engagement unterscheiden muss. Dabei wird in bestimmten Abständen, welche hier Belohnungsintervalle genannt werden, das gutgeschriebene Engagement erhöht. Um solch eine Erhöhung zu erhalten muss derjenige über das ganze Belohnungsintervall hinweg online sein.

Am Anfang, also bei kleinem gutgeschriebenen Engagement, wird dieses jede Minute erhöht. Das entspricht einer kontinuierlichen Belohnung, weil der eigentliche Nutzen aus dem gutgeschriebenen Engagement berechnet wird. In diesem Zeitraum entspricht das eigentliche dem gutgeschriebenen Engagement relativ genau.

Bei steigendem gutgeschriebenen Engagement, also bei einer festeren Bindung, soll die Belohnung aber in einen variablen Quotenplan übergehen. Hierzu wird die Länge des Belohnungsintervalls erhöht und gleichzeitig variabel, also zufällig, gestaltet. Dementsprechend wird die Länge eines Belohnungsintervalls zufällig gewürfelt. Das Minimum der Länge ist dabei eine Minute, das Maximum wird an das gutgeschriebene Engagement angepasst (siehe Abbildung 3). Geht die Person innerhalb dieser Zeit offline, so verfällt das Engagement seit dem Ende des letzten Belohnungsintervalls. Bleibt sie aber die ganze Zeit online, wird ihr eine bestimmte Menge von Engagement gutgeschrieben. Hierbei wird ihr soviel Zeit gutgeschrieben wie durchschnittlich das Belohnungsintervall bei dem entsprechenden Engagement lang ist, also die Hälfte des Maximums der Länge des Belohnungsintervalls.

Angenommen bei einem bestimmten Engagement ist das Belohnungsintervall maximal 30min lang. Dementsprechend wird eine gleichverteilte Zufallslänge zwischen einer Minute und 30min gewürfelt. Diese könnte beispielsweise drei Minuten sein, aber genauso gut 28min. Hier wird die Gleichverteilung gewählt, damit es eine relativ hohe Chance von extremen Werten gibt. Dementsprechend ist auch die Chance relativ hoch überproportional belohnt zu werden. Dass es eine genauso hohe Chance gibt, unterproportional belohnt zu werden, spielt eine geringere Rolle.

Nach Ablauf dieser Zeit wird das gutgeschriebene Engagement um 16 min erhöht. Die 16 Minuten entsprechen dem gerundeten Erwartungswert, also dem Minimum (eine Minute) plus dem Mittel der gewürfelten Zufallszahl (diese liegt hier zwischen 0 und 29). Dieses passiert, egal ob das Intervall nur drei Minuten lang war oder 28 min. Hierdurch kann es sein, dass man überproportional (Zufallslänge kleiner 15min) oder unterproportional (Zufallslänge über 15min) belohnt wird. Im Durchschnitt wird es sich aber mitteln und damit dem eigentlichen Engagement entsprechen.

Die Berechnung des maximalen Länge eines Belohnungsintervalls erfolgt nach der Formel:

1+Engagement in Minuten/2232

Die 2232 entsprechen der Anzahl an Minuten in drei Monaten, wobei hier der Einfachheit halber jeder Monat mit 31 Tagen angenommen wird. Die Zahl von drei Monaten stammen aus dem Absatz über die Abrechnung des Engagements (siehe III.1.5.1).

Durch diese Formel wird bei vollem Engagement einem Maximum des Belohnungsintervalls von 60min berechnet, bei minimalem Engagement einem Maximum von einer Minute. Dazwischen wird linear interpoliert. Das Ganze ist in Abbildung 3 graphisch aufgetragen. Sowohl das Maximum als auch die Linearität der Interpolation müssten durch Studien bestätigt werden. Dieses würde aber den Rahmen dieser Arbeit sprengen.

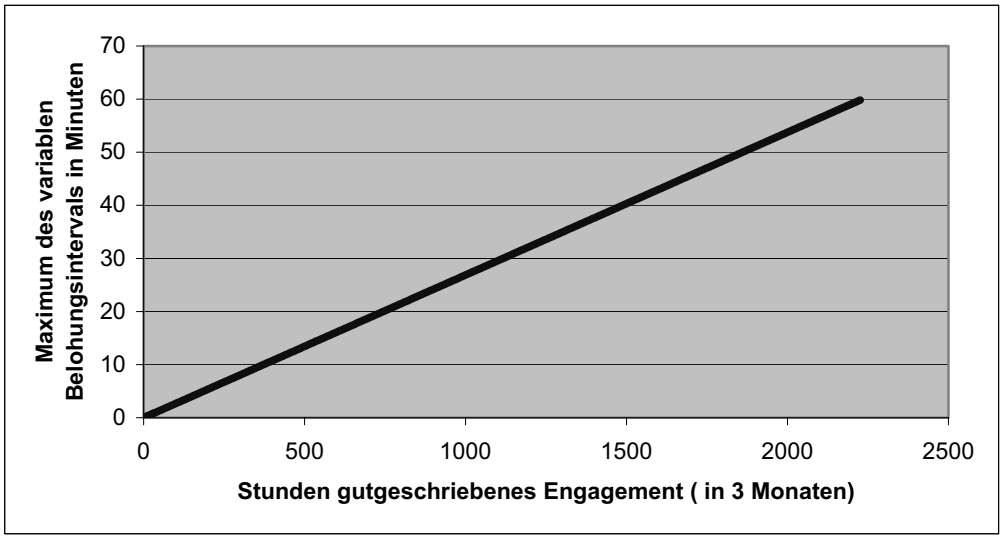

Abbildung 3: Abhängigkeit zwischen dem maximalen Belohungsintervall und dem schon geleisteten gutgeschriebenen Engagement.

III.1.5.3.2. Berechnung der Belohnung aus dem gut geschriebenen Engagement

Wie unter III.1.5.2 beschrieben wurde, darf man eine bestimmte Anzahl an Stunden pro Woche bei einem bestimmten Engagement den Internetzugang von anderen Mitgliedern benutzen (hier Nutzen genannt). Um dieses zu realisieren, wird in fixen kurzen Intervallen (beispielsweise fünf Minuten) aus dem gutgeschriebenen Engagement die Nutzensmenge für dieses Intervall berechnet. Die Resultate dieser Berechnungen der letzten Woche werden aufaddiert und bilden den zugesprochenen Nutzen (in Minuten Surfzeit).

Um die Forderungen des Behaviorismus erfüllen zu können, muss bei einem geringen Engagement relativ stark belohnt werden. Die Belohnungsstärke muss dann stark abnehmen, damit der Sättigungspunkt, der dem zu erwartenden durchschnittlichen Verbrauch entspricht, möglichst spät erreicht wird. Ein Beispiel für eine entsprechende Funktion ist die Logarithmusfunktion. Ist der Sättigungspunkt erreicht, muss die Belohnungsstärke wieder zunehmend ansteigen. Ein Beispiel für solch eine Funktion ist die Exponentialfunktion. Mit diesen beiden Funktionen und einem Sättigungspunkt von einer Stunde pro Tag (sieben

Stunden pro Woche) bei einem Engagement von ca. 16h/Tag (110h/Woche) wurde die Abbildung 4 dargestellt. Hierbei wird vom Nullpunkt bis zum Sättigungspunkt die Funktion

$$\log_{10}(\text{Stunden pro Woche}+1)*3{,}5$$

benutzt, während von dort an die Funktion

$$\text{EXP}((\text{Stunden pro Woche}-96)/14)+4{,}4403486$$

verwendet wird. Diese Funktionen sind prinzipiell willkürlich gewählt, mussten aber die oben genannten Eigenschaften erfüllen, streng monoton ansteigend sein und wertmäßig ineinander übergehen.

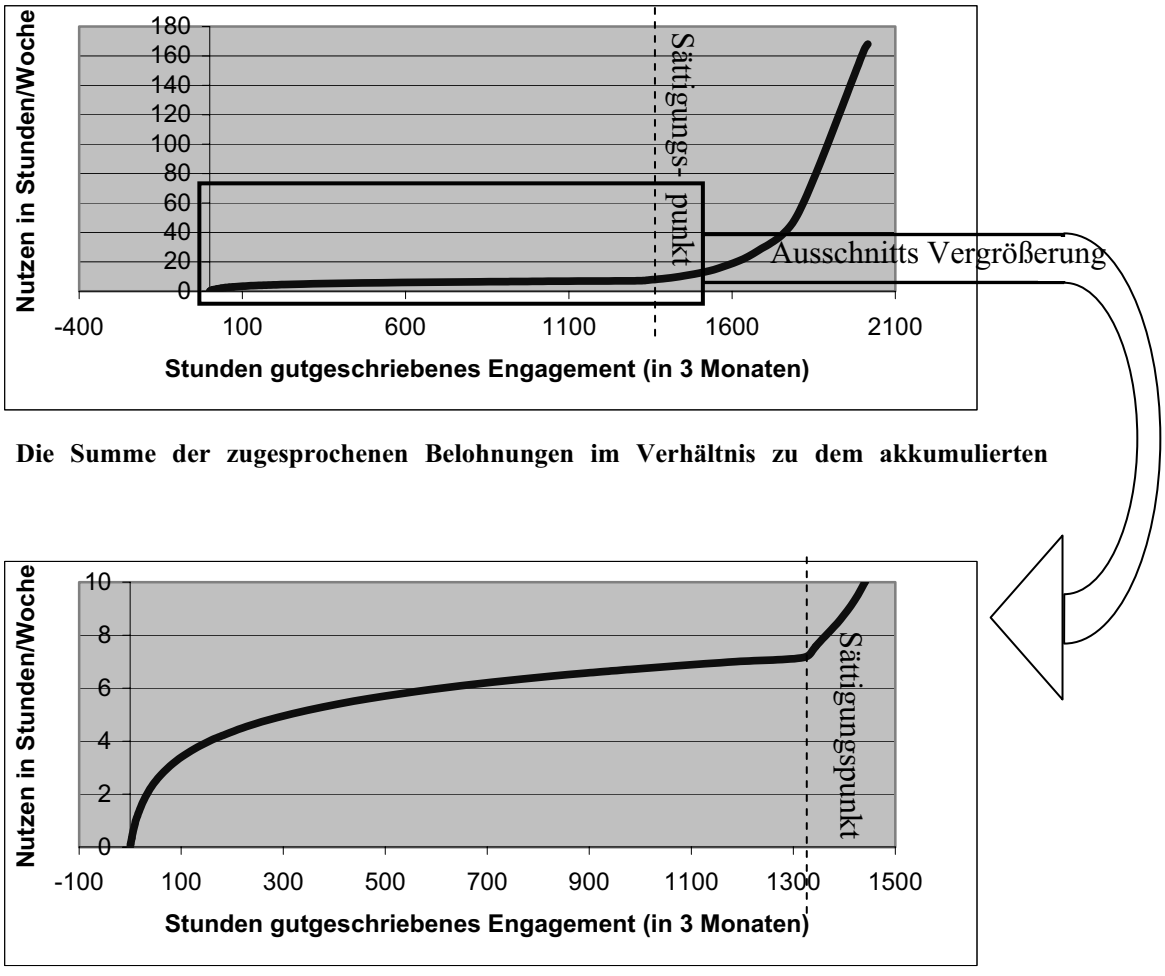

Abbildung 4: Die Summe der zugesprochenen Belohnungen im Verhältnis zu dem akkumulierten Engagement.

Abbildung 5: Ausschnittsvergrößerung der Abbildung 4

Es wird angenommen, dass die Variabilität und die Länge des Belohnungsintervalls dazu führt, dass die Benutzer möglichst selten ihren Onlinestatus verändern. Zusammen mit der Annahme des Sättigungswertes von 16 Stunden ergibt sich daraus, dass im Folgenden davon ausgegangen werden kann, dass die Benutzer mehrere Stunden pro Tag an einem Stück online sind. Als Richtgröße könnte man hier die 16 Stunden des Sättigungswertes nehmen. Dieses hat einen relativ großen technischen Nutzen (siehe III.2.3).

III.1.5.4. Gerechtigkeit im gewählten System

Dieses System der Belohnung ist unabhängig von äußeren Faktoren und nur abhängig von dem Verhalten des Einzelnen, wodurch auch die Forderung nach Gleichheit der Benutzer erfüllt ist. Damit ist es im Großen und Ganzen gerecht (siehe II.4.4).

Das Prinzip der Kohärenz (siehe II.4.4) wird allerdings nicht erfüllt. Würde man auch dieses beachten, so würde der Sättigungspunkt bei einem viel geringeren Engagement erreicht werden. Unter der Annahme, dass mehrere Benutzer genau gleich lang online sind, müssen diese in einem gerechten System auch die gleiche Belohnung erhalten. Sollten sie sich zusammentun, müssen sie dementsprechend auch den Gesamtgewinn gleich zwischen sich teilen. Dementsprechend muss nach dem Prinzip der Kohärenz jemand, der sich Y-fach so stark engagiert, auch die Y-fache Belohnung erhalten. Dieses ist damit zu begründen, dass eine Gruppe aus Y Personen, die sich zusammentun, gleich stark engagieren und untereinander teilen, genauso viel erhalten müssen, als wenn jeder einzelne seine Ansprüche gegenüber der Gemeinschaft einzeln angeben würde:

$$Y*F(X)=F(Y*X)$$

X= Engagement F(X)=Belohnung Y= Anzahl an Personen

Diese Forderung zwingt zu einem linearen Verlauf der Belohnung im Hinblick auf das Engagement. Zusammengenommen mit der Forderung, dass bei maximalem Engagement auch maximal belohnt wird, wird der Sättigungspunkt sehr viel früher erreicht als bei einem nicht linearen Verlauf.

Für die Nutzenmaximierung muss der Zugewinn an Gerechtigkeit und die damit verbundene höhere Akzeptanz gegen die Verschiebung des Sättigungspunktes abgewogen werden. Der Autor ist der Meinung, dass sich Personen in einem nicht kohärenten System mit einem späten Sättigungspunkt mehr engagieren als in einem kohärenten mit relativ frühem Sättigungspunkt. Dieses wird damit begründet, dass Kohärenz nur einen geringen Zuwachs an Gerechtigkeit bietet, während das hier entworfene Belohnungssystem nach dem Behaviorismus relativ starke Effekte erzielen sollte.

III.2. Abrechnungssystem-Design

III.2.1. Grundlegende Ziele beim Design

Bei dem Design des Abrechnungssystems und des dazugehörenden Netzes wird versucht, mehrere Ziele zu verwirklichen. Das Wichtigste ist dabei, ein gewisses Maß an Sicherheit zu gewähren. Wenn das System einfach und beliebig manipulierbar ist, so hat es wenig Sinn. Das Beste wäre es, wenn man Garantien für die Sicherheit oder andere wichtige Eigenschaften des Systems geben könnte. Dieses ist aber häufig entweder überhaupt nicht möglich oder aber mit einem so großen Aufwand verbunden, dass es nicht praktikabel ist.

Wenn man keine Garantien geben kann, so sollte man anstreben, Aussagen über die Wahrscheinlichkeit vom Nichtzutreffen der entsprechenden Eigenschaft treffen zu können. Zum Beispiel wäre es gut, wenn man sagen können, dass zu 99% jemand wirklich der ist, der er behauptet zu sein. Hierbei wäre es sehr vorteilhaft, wenn solche Aussagen beweisbar oder wenigstens qualitativ und / oder quantitativ einsichtig wären.

Ein weiteres jedoch untergeordnetes Ziel ist es, die Annahmen zu minimieren, die vorab getroffen werden müssen, um bestimmte Eigenschaften zu erzielen (siehe [44]). Dieses

Ziel ist jedoch nicht so wichtig, da die Funktionsfähigkeit des Netzes nur in wenigen Fällen von solchen Entscheidungen abhängt. Bei einem endgültigem Entwurf, bei dem ein voll funktionsfähiges Netz optimiert wird, sind dieses jedoch wichtige Punkte. Diese Vorab-Annahmen sind entweder Worst-Case-Annahmen, Durchschnittswerte oder Best-Case-Annahmen. Wenn man den Worst-Case benutzt, so wird man im Durchschnitt zu viel Aufwand betreiben, während bei der Benutzung eines Mittelwertes hin und wieder die gewünschte Eigenschaft nicht gewährleistet ist, weil man zu wenig Aufwand betreibt. Bei der Best-Case-Annahme wird noch häufiger zu wenig Aufwand betrieben.

Um dieses Ziel zu erfüllen, kann man entweder versuchen Algorithmen zu finden, die ohne die entsprechenden Werte auskommen, oder man versucht, diese während des Betriebes des Netzes zu messen.

Beim Messen von Werten sollte man aber aufpassen, dass man so selten wie möglich auf Wissen über den globalen Status des Netzes zugreift (siehe [46]), da dieses im Konflikt mit dem Ziel der Skalierbarkeit (siehe II.5.2) steht. Das Ermitteln eines Wertes über den globalen Status des Netzes ist nämlich häufig mit großem Aufwand verbunden. Letzterer entsteht dadurch, dass wenigstens potentiell jedes Mitglied des Netzes befragt werden muss. Dieses entspricht einem Aufwand von O(n) und skaliert damit nicht besonders gut. Wenn aber jedes Mitglied diesen Aufwand betreiben muss, hat man insgesamt einen Aufwand von O (n^2), was bei der geforderten Skalierungsfähigkeit so gut wie nicht mehr möglich ist.

III.2.1.1. Sicherheit auf der Designebene

III.2.1.1.1. Mehrheitsentscheid und Redundanz

Eine Maßnahme, um auf der Designebene Sicherheit zu gewährleisten, ist es, dass wichtige Fragen oder Aufgaben nicht von einer, sondern von mehreren Identitäten (je nach Beispiel echte Personen oder technische Instanzen) beantwortet bzw. übernommen werden. Dabei sollte die Auswahl der Identitäten eine mehr oder weniger zufällige sein. Wenn dem so ist, sinkt die Wahrscheinlichkeit stark, dass man falsch informiert ist, ohne dieses zu merken.

Man nehme beispielsweise an, dass X Prozent einer Gruppe falsche Antworten gibt. Wird nur eine Identität gefragt, so kann die Antwort richtig oder falsch sein. Man kann aber nicht unterscheiden, ob das eine oder das andere zutrifft. Dementsprechend ist die Wahrscheinlichkeit, dass man eine falsche für eine richtige Antwort hält, genauso groß wie die, an jemanden zu geraten, der falsche Antworten gibt. Dementsprechend ist die Wahrscheinlichkeit ebenfalls X. Fragt man jedoch zwei Identitäten, so kann es sein, dass man gleiche oder unterschiedliche Antworten erhält. Wenn die Antworten die gleichen sind, so ist die Wahrscheinlichkeit, dass die Antwort falsch ist, nur noch X^2. Dieses entspricht der Wahrscheinlichkeit an zwei „unlautere“ Identitäten zu geraten. Wenn die Antworten aber unterschiedlich sind, so weiß man wenigstens, dass jemand einen, eventuell absichtlich, falsch informiert. Man kann dabei sogar die Wahrscheinlichkeit ausrechnen, mit der man die richtige Antwort auswählt. Nach diesem Wert kann man dann eine Antwort annehmen, wegen zu großer Unsicherheit verwerfen oder versuchen, die Wahrscheinlichkeit der richtigen Auswahl durch zusätzliche Anfragen zu erhöhen.

Bei einer weiteren Verallgemeinerung erhält man mehrere verschiedene Antworten. Diese Antworten können aber nur richtig oder falsch sein.

Normalerweise weiß man dabei, wie viele verschiedene Antworten gleichzeitig richtig sein können. Wenn man beispielsweise nach Wegen zu einem Ziel fragt, so kann man auch verschiedene Antworten erhalten, die alle richtig sein können. Wenn man aber danach fragt, wie ein Kontostand gerade ist, so gibt es nur eine richtige Antwort.

Dementsprechend kann man verschiedene Erklärungsmodelle für die Antwort erstellen, die man erhalten hat. Für jedes dieser Erklärungsmodelle kann man nun eine Wahrscheinlichkeit berechnen, die angibt, wie wahrscheinlich dieses Modell ist, wenn X% der Leute falsche Antworten geben.

Die Wahrscheinlichkeit, dass das entsprechende Erklärungsmodell das richtige ist, kann man dadurch berechnen, dass man die Wahrscheinlichkeit des Auftauchens dieses Modells durch die Summe der Wahrscheinlichkeiten des Auftauchens aller Modelle teilt.

Die Wahrscheinlichkeit des Auftauchens eines bestimmten Erklärungsmodells kann man mit dem Wissen über die Anzahl der richtigen (r) und falschen (f) Antworten errechnen. Dabei ist die Wahrscheinlichkeit, r richtige und f falsche Antworten zu erhalten, gleich

$$(1\text{-}X)^r(X)^f.$$

Diese Formel ist aber nur für parallele Anfragen richtig. Beispielsweise beim Routing hängt eine Antwort von der vorhergehenden ab. Wenn eine einzige Antwort in einer solchen Frage-Antwort-Kette falsch ist, so ist die Antwort der ganzen Kette als falsch anzusehen. Dementsprechend steigt die Wahrscheinlichkeit, eine falsche Antwort zu erhalten, exponentiell mit der Länge der Kette. Die Wahrscheinlichkeit, dass solch eine Frage-Antwort-Kette der Länge k mit einer richtigen Antwort beendet wird, ist

$$(1\text{-}X)^k.$$

Diese Formel kann man in die obige Formel einsetzten, um die Wahrscheinlichkeit zu berechnen, mit der man eine richtige Antwort erhält, wenn man mehreren Frage-Antwort-Ketten benutzt.

Damit diese Maßnahme aber etwas nutzt, ist die Zufälligkeit der gefragten Identitäten wichtig. Hierbei geht es darum, dass die Wahrscheinlichkeitsrechnung überhaupt anwendbar ist. Wenn vorhersehbar ist, wer gefragt wird, so können Identitäten, die betrügen wollen, versuchen, die entsprechenden Stellen zu besetzen. Hierdurch würden beispielsweise nur Betrüger gefragt werden, obwohl eine Mehrzahl an redlichen Identitäten zur Verfügung stünde. Dementsprechend sollte die Auswahl derjenigen, die gefragt werden, zufällig sein, wodurch es Betrügern sehr erschwert wird entsprechende Posten einzunehmen.

III.2.1.1.2. Wissensmangel als Schutz vor Angriffen

Im Allgemeinen braucht man für effiziente Angriffe Wissen. Je weniger ein Angreifer weiß und je schwieriger es ist, solch ein Wissen zu erlangen, desto aufwändiger wird es, ein unlauteres Ziel zu erreichen. An dieser Stelle werden zwei Beispiele für fehlendes Wissen gegeben, wodurch Angreifer behindert werden können.

III.2.1.1.2.1. Anonymität

Das erste Beispiel ist die Korrelation von Identitäten. Um beispielsweise ein Ziel durch Bestechung oder Erpressung zu realisieren, muss man wissen, wer einem dabei helfen kann, wenn man diesen erpresst, und man muss einen Weg finden, den Menschen hinter der Identität zu kontaktieren. Wenn es innerhalb des Netzes keine Möglichkeit hierfür gibt, muss man dieses über andere Protokolle tun. Hierzu muss man die Identität im Netz zu anderen Identitäten wie beispielsweise einer IP-Adresse oder einer Postanschrift in Beziehung setzen können. Wenn dieses nicht möglich ist (Anonymität), so sind solche Angriffe auch nicht bzw. nur sehr schwer möglich. Je weniger solcher Beziehungen von verschiedenen Adressräumen bekannt sind, desto sicherer ist das System, bzw. desto aufwändiger sind Angriffe (siehe [40]).

III.2.1.1.2.2. Datenlokalisation

Eine weitere Information, die man für Angriffe nutzen kann, ist die Aufgabenverteilung. Um Daten manipulieren zu können, muss man Zugriff auf die entsprechenden „Originale“ erlangen. Dieses lässt sich aber nur effizient bewerkstelligen, wenn man sich in eine bestimmte „Richtung bewegen“ kann und auch weiß, welche „Richtung“ die richtige (also zu den Daten hin) ist.

Die Möglichkeit sich zu „bewegen“ ist meistens gegeben, da die Stabilisierung eines Netzes ein gewisses Maß an Aufgabenumverteilung erfordert. Dieses kann ausgenutzt werden, um neue Aufgaben zu übernehmen. Eine andere Möglichkeit ist gegeben, wenn neue Mitglieder oder jene, welche nach (kurzer) Abwesenheit wieder online kommen, eine dynamische Zuweisung von Aufgaben erhalten.

Ist außerdem eine Metrik über der Aufgabenverteilung bekannt, so ist es möglich, sich mindestens langsam und gezielt durch das Netz zu „bewegen“. Solch eine Metrik wird aber auch gebraucht, wenn es eine effiziente Möglichkeit zum Abfragen der Daten geben soll.

III.2.1.1.3. Diskussion des Entscheidungsfindungsprozesses

Beim Design von Peer-to-Peer-Netzen überlässt man bei einer 1-zu–n-Beziehung die Entscheidungen gerne dem Antragsteller. Dieses ist unter der Annahme von Protokoll konformen Verhalten durchaus gerechtfertigt, da es häufig einfacher zu realisieren ist und / oder weniger Nachrichtenaufwand nötig ist als das Treffen von Entscheidungen durch diejenigen, an die der Antrag geht.

Die Art, wie man das Peer-to-Peer-Netz findet und sich mit diesem verbindet, ist ein Beispiel für so ein Design. Hierbei möchte der „Neue“ ein Teil des Netzes werden, ist also der Antragsteller. Im Normalfall entscheidet dabei dieser, an welche Stelle des Netzes er tritt. Je nach Methode bleibt er mit dem ersten, den er kontaktiert hat, verbunden oder beginnt eine Suche nach einer besser geeigneten Stelle, an der er dann integriert wird.

Wenn es aber um sicherheitsrelevante Entscheidungen geht (wie beispielsweise die Einwahl), so bietet man dadurch Angreifern eine einfache Möglichkeit dieses auszunutzen, wenn sie sich nicht protokollkonform verhalten. Der Angreifer erhält damit die Möglichkeit die Entscheidung so zu treffen, wie es für ihn und nicht wie es für das Netz am nützlichsten ist.

Aus diesem Grund ist es wichtig, dass man solche Entscheidungen trotz des Mehraufwandes und der erhöhten Komplexität beim Design vom Netz treffen lässt. Dabei ist zu beachten, dass die Initiative immer vom Antragsteller ausgeht, der sich damit denjenigen aussuchen kann, den er „anspricht“. Dementsprechend kann er sich auch jemanden auswählen, mit dem er eine Gruppe bildet (siehe II.4.3.2, II.4.3.3) und der von der „anderen Seite aus“ eine Entscheidung in seinem Sinne fällt. Dementsprechend sollten im Idealfall auch hierbei Sicherheitsmaßnahmen getroffen werden, so dass Entscheidungen nicht von einer einzelnen Identität getroffen werden können (siehe III.2.1.1.1).

III.2.1.1.4. Audits

Bei hohen Sicherheitsanforderungen kann man so genannte Audits einführen. Hierbei wird die Handlungsweise von Identitäten durch Fremde begutachtet und überprüft. Dieses sollte am Besten so geschehen, dass die entsprechenden Identitäten nicht merken, dass sie überprüft werden. Wenn bei der Überprüfung falsche Handlungsweisen oder Betrügereien entdeckt werden, so sollte dieses negative Konsequenzen für die entsprechende Identität haben.

Hierdurch sind Betrügereien viel schlechter durchzuführen, da der Betrüger nicht weiß, ob er gerade überprüft wird oder ein potentielles Betrugsopfer Anfragen an ihn stellt.

Wenn die Audits häufig und die Konsequenzen hart genug sind, so lohnt sich der Betrug nicht mehr, da die Gewinnerwartung für den Betrug geringer oder gleich dem eines regelkonformen Verhaltens ist (siehe [01]).

Um solche Audits durchführen zu können, müssen aber erst einmal alle relevanten Daten und Entscheidungen protokolliert und offen gelegt werden. Zusätzlich müssen auch die Sachen, die überprüft werden sollen, deterministischer Natur sein, so dass dieses auch getan werden kann. Und zu guter Letzt müssen Nachrichten und Aktionen eindeutig Identitäten zuzuordnen sein. Dieses widerspricht aber der Forderung, dass man möglichst viel Wissen verbergen sollte, da dieses für Angreifer nützlich sein kann. Dementsprechend ist solch eine Maßnahme im Einzelfall abzuwägen, ob es mehr positive als negative Folgen hat.

III.2.2. Verwaltungsstruktur

Die Daten über ein Konto müssen bei jeder Inanspruchnahme der Gemeinde vorhanden sein. Zur Bereitstellung der Daten gibt es generell zwei Möglichkeiten. Entweder der Benutzer verwaltet seine Daten selbst, oder sie werden innerhalb des Netzes von anderen Gemeindemitgliedern gespeichert(siehe [01]).

III.2.2.1. Selbstverwaltung

Wenn ein Benutzer seine Daten selbst verwaltet, so ergibt sich das Problem, dass er ein Interesse daran hat, seine Daten falsch zu veröffentlichen. Deswegen müssen die Daten regelmäßig überprüft werden. Hierzu müssen die entsprechenden Größen zusätzlich auch im Netz gespeichert werden, da diese sonst nicht überprüft werden können. Hierbei müssen die Kontrollen sehr häufig durchgeführt werden, so dass kaum ein Betrug unbemerkt bleibt. Dieses ist aus dem Behaviorismus zu begründen, da sich sonst das entsprechende Verhalten verfestigt und dann dauerhaft gezeigt wird. Des Weiteren muss auch eine Liste derjenigen online gehalten werden, die Betrogen haben, damit jenen kein Service auf Grund der falschen Daten geboten wird.

Zusammengefasst muss ein zweites System aufgebaut werden, welches der Fremdverwaltung (siehe III.2.2.2) sehr ähnlich ist. Es entsteht also kein Vorteil gegenüber der Fremdverwaltung. Es treten aber Probleme bei der Buchführung auf. Die positiven Größen fallen nicht unbedingt an demselben Computer an wie die negativen. Erstere fallen normalerweise an einem Desktopcomputer an, der fest mit dem Internet verbunden ist. Dieser wird im Folgenden Router genannt. Die negativen Größen fallen jedoch normalerweise an einem Laptop an. Deswegen müssen die beiden Computer synchronisiert werden. Hierzu muss der Router entweder immer eingeschaltet sein, wenn die Gemeinde in Anspruch genommen wird, oder die entsprechenden Computer müssen synchronisiert werden, bevor ein Mitglied die Gemeinde in Anspruch nimmt. Wenn der Router einen Hardwaredefekt hat, kann gar nicht auf diese Daten zugegriffen werden, weswegen bei solch einem Defekt die Gemeinde nicht benutzt werden kann, obwohl gerade hier eine erhöhte Notwendigkeit entstehen würde.

III.2.2.2. Fremdverwaltung

Wenn die Daten von jemand anderem verwaltet werden, hat dieser kein direktes Interesse die Daten zu fälschen. Deswegen muss auch nicht so stark kontrolliert werden wie bei der Selbstverwaltung (siehe III.2.2.1). Es sollte aber versucht werden sicherzustellen, dass nicht doch jemand durch Zufall seine eigenen Daten verwaltet. Hierbei muss man davon ausgehen, dass aktiv und durch Betrug versucht wird, die Kontrolle über Daten zu erlangen, um diese verändern zu können. Ganz ohne Kontrollen geht es aber auch nicht, wenn dieses

sichergestellt ist. In bestimmten Situationen kann es doch ein Interesse geben, Daten falsch zu speichern. Das korrekte Speichern von Daten ist mit einem Arbeitsaufwand verbunden, deswegen wäre es sinnvoll dieses nicht zu leisten. Allerdings müsste dafür der Besitzer des Computers Arbeit investieren (d.h. das entsprechende Programm umprogrammieren), damit der Computer ein klein wenig entlastet würde. Dieses ist unwahrscheinlich, da der Arbeitsaufwand in Relation zum Nutzen sehr viel größer ist. Freunde oder Gruppenmitglieder decken zu wollen (siehe II.4.3.2) oder Bestechungen zu erhalten (siehe II.4.3.3), sind andere Motive für eine Fälschung der Daten. Dieses Szenario ist wahrscheinlicher, da sich hier ein höherer Gewinn erwirtschaften lässt, der den Arbeitsaufwand rechtfertigen kann.

Zusätzlich zu dem geringeren Kontrollaufwand gibt es noch den Vorteil, dass im Normalfall bei Betrugsversuchen diese Information einfach zu den anderen Daten eines Kontos hinzugefügt werden kann. Man braucht keine extra Negativliste wie bei der Selbstverwaltung (siehe III.2.2.1). Des Weiteren kann man Mehrfachspeicherung (siehe III.2.1.1.1) benutzen, um die Sicherheit der Informationen weiter zu erhöhen. Auch gibt es hier keine Probleme die Daten online zu halten. Aus diesen Gründen wird hier die Fremdverwaltung benutzt.

III.2.3. Das Netz

Um eine Fremdverwaltung realisieren zu können, benötigt man eine Suchmöglichkeit. Diese ist dafür da, dass man den oder die aktuellen Verwalter des entsprechenden Kontos herausfinden kann, um Anfragen zu stellen oder um ihnen Ereignisse melden zu können. Dabei gibt es zwei Grundtypen von Peer-to-Peer-Netzen, die beide eine Suche erlauben. Dieses sind die strukturierten und die unstrukturierten Netze.

III.2.3.1. Struktur des Netzes

III.2.3.1.1. Eigenschaften unstrukturierter Netze

In unstrukturierten Netzen wird die Anordnung der Netzkomponenten, welche Computer also miteinander verbunden sind oder voneinander wissen, nicht aktiv gesteuert, um bestimmte Eigenschaften des Netzes zu erreichen. Es wird auch nicht gewusst, welche Inhalte Nachbarknoten speichern. Deswegen kann man nur suchen, indem man entweder die Anfrage flutet oder über Zufallspfaden durch das Netz schickt (siehe [03]). Beide Methoden sind nicht gezielt und deswegen relativ ineffizient (siehe [39]). Zusätzlich skalieren sie recht schlecht (siehe [38]). Durch diese Art der Suche lassen sich auch Duplikate von Daten schlecht finden und dadurch bei Updates schlecht synchron halten (siehe [38]). Auf Grund der nicht kontrollierten Struktur lassen sich, erst recht im Hinblick auf böswillige Benutzer, nur sehr bedingt Aussagen treffen (siehe [39]). Der Vorteil solcher Netze ist, dass sie keine Probleme haben, wenn Computer ausfallen oder sich neu verbinden. Beim Ausfall sind nämlich keine oder kaum Reparaturen der Struktur nötig. Zusätzlich bieten sie keine Metrik zum Finden von Inhalten bzw. von Aufgaben, was positiv im Sinne der Sicherheit ist (siehe III.2.1.1.2.2).

III.2.3.1.2. Eigenschaften strukturier Netze

Strukturierte Netze verhalten sich ganz anders. Sie kontrollieren die Netzstruktur, wodurch eine schnelle und effiziente Suche möglich wird. Auch skalieren sie besser und man kann aus der Kenntnis der Struktur heraus beweisbare Aussagen über das Verhalten treffen. Außerdem ist es einfacher, alle Duplikate eines Datensatzes zu finden und zu verändern. Hierdurch kann man mehrere Duplikate eines Datensatzes anlegen, was Betrügereien

erschwert (siehe III.2.1.1.1). Zusätzlich verringert die effiziente Suche auch die Wahrscheinlichkeit eines Betruges, da sich die Frage-Antwort-Ketten stark verkürzen (siehe III.2.1.1.1).

Um diese Vorteile erzielen zu können, muss man aber die Struktur ständig kontrollieren und, wenn nötig, reparieren. Deswegen muss ein gewisser Aufwand beim Ausfall einzelner Computer betrieben werden. Hier kann man ein Netz beispielsweise in Form eines Ringes annehmen. In diesem verbindet sich jeder Computer mit seinem Vorgänger und seinem Nachfolger, welche durch ein Ordnungskriterium definiert werden. Wenn jetzt ein Computer aus dem Netz ausscheidet, so ändert sich für den Vorgänger und den Nachfolger dieses Computers die Vorgänger/Nachfolgerrelation und es müssen so neue Verbindungen aufgebaut werden.

Bei sehr schnellem Wechsel der Computer, die am Netz teilnehmen, kann dieses zu einem erheblichen Ressourcenverbrauch führen. Ein weiterer Nachteil ist, dass die Metrik, welche eine effiziente Suche erlaubt, es für Kriminelle einfacher macht, gezielt die Verwaltung von bestimmten Datensätzen zu übernehmen (siehe III.2.1.1.2.2).

Der Autor der Arbeit geht davon aus, dass die Anreize (siehe III.1.5) genügen, um ein relativ stabiles Netz zu erhalten, so dass die Vorteile der strukturierten Netze deren Nachteile überwiegen.

III.2.3.2. Arten von strukturierten Netzen

Auch bei strukturierten Netzen gibt es verschiedene Grade an Ordnung. Die eine Möglichkeit ist es, dass man eine deterministische, strikte Ordnung für das Netz entwirft, in der die Verbindungen zwischen den Knoten im Voraus festgelegt sind (z.B. De-Bruijn-Graph [43], siehe III.2.3.2.2). Die andere Möglichkeit ist es, nur allgemeine Regeln für die Verbindungen festzulegen (z.B. P-Grid [44], siehe III.2.3.2.1). Hierdurch ist die genaue Struktur des Netzes im Voraus nicht bekannt, man kann nur Aussagen über bestimmte Eigenschaften machen, welche durch Regeln festgelegt sind.

III.2.3.2.1. Eigenschaften wenig strukturierte Netze

Wenn nur die grobe Struktur des Netzes im Voraus festgelegt wird, so wird nicht so viel Wissen über den aktuellen Zustand beim Aufbau und Erhalt der Struktur benötigt. Es genügt, wenn nur wenig Aufwand für diese Ziele betrieben wird. Deswegen sind solche Netze robuster gegen eine hohe Fluktuation der teilnehmenden Computer.

Ein sehr wichtiger Nachteil entsteht aber, wenn bei der Suche auf Redundanz geachtet werden soll (siehe III.2.1.1.1). Hierbei wird die Wahrscheinlichkeit vermindert eine falsche Antwort zu erhalten, indem eine Suche mehrmals durchgeführt wird. Ein Beispiel, welches durch die hier genannten Maßnahmen umgangen werden kann, ist folgendes: Um zu betrügen, beantwortet jemand, obwohl er gar nicht für die entsprechenden Daten zuständig ist, eine Anfrage falsch. Dadurch, dass aber verschiedene Wege benutzt werden, kann dieser nicht alle Anfragen abfangen und falsch beantworten, sondern nur jene, die auch über den entsprechenden Computer gerouted werden. Die anderen Anfragen werden mit relativ hoher Wahrscheinlichkeit richtig beantwortet.

Dementsprechend ist, wie schon unter III.2.3.1.2 beschrieben wurde, bei der Suche das schwächste Glied wichtig. Wenn also mehrere Suchaktionen einen einzigen Knoten gemeinsam haben, so steigt die Wahrscheinlichkeit stark, dass beide Routinginformationen falsch sind, gegenüber einem Szenario mit nicht überlappenden Pfaden. Um also den vollen Nutzen von III.2.1.1.1 erreichen zu können, müssen die Pfade der verschiedenen Suchaktionen vollkommen disjunkt sein. Wegen der geringen Strukturierung des Netzes kann dieses aber nicht im vorab geprüft oder garantiert werden. Deswegen muss es also während

der Suchanfragen geprüft werden. Damit diese nicht manipuliert werden können, muss außerdem iterativ gesucht werden, da bei einer rekursiven Suche wieder das Phänomen des schwächsten Gliedes einer Kette auftauchen würde. Eine iterative Suche steht nicht im Einklang mit der Forderung nach Anonymität (siehe III.2.1.1.2).

III.2.3.2.2. Eigenschaften deterministischer, stark strukturierter Netze

In deterministischen, stark strukturierten Netzen lässt sich eine Überlappung von Suchpfaden im Voraus prüfen, da die Struktur des Netzes im Voraus bekannt ist. Deswegen kann man eine rekursive Suche benutzen, wodurch eine teilweise Anonymität möglich wird.

Allerdings muss ein relativ hoher Aufwand betrieben werden, um die Netz-Struktur aufrecht zu erhalten, da auch bei relativ kleine Abweichungen die gegebenen Garantien nicht mehr gehalten werden können. Deswegen sind solche Netze wenig robust gegen eine hohe Fluktuation.

Die Anreize (siehe III.1.5) sollten dafür sorgen, dass im Netz wenige Ausfälle und Neuintegrationen geschehen (siehe III.1.5.3.2). Hier wird eine Onlinezeit von mehreren Stunden an einem Stück angenommen, was man im Verhältnis zu Onlinezeiten von nur einer Stunde oder einigen Minuten (siehe II.5.1) in anderen Netzen sehen muss. Deswegen sind die Vorteile der deterministischen, stark strukturierten Netze schwerwiegender als die Nachteile. Hieraus leitet der Autor seine Entscheidung ab, ein solches Netz zu Grunde zu legen.

III.2.3.3. Andere Kriterien der zugrunde liegenden Struktur

Zusätzlich zu den in den anderen Abschnitten von III.2.3.1 besprochenen Kriterien sollen in diesem Abschnitt weitere dargelegt werden und eine abschließende Entscheidung über die zu Grunde liegende Struktur gefällt werden (Daten aus [43]).

Ein wichtiges Kriterium für ein Peer-to-Peer-Netzwerk ist sein Durchmesser, da dieser die maximale Distanz angibt, in der zwei Knoten zueinander liegen. Je kleiner der Durchmesser ist, desto schneller und effizienter kann ein beliebiger Knoten gefunden werden (siehe Tabelle 2 und Tabelle 3). Der theoretisch minimale und damit optimale Durchmesser solch eines Netzes liegt bei

$$\left\lceil \log_k(N(k-1)+1) \right\rceil - 1$$

N=Anzahl der Knoten K=Eingangs-/Ausgangs-Grad eines Knotens (siehe [43]).

Des Weiteren ist auch sehr wichtig, mit wie vielen Verbindungen der entsprechende Durchmesser erzielt wird (Grad /siehe Tabelle 2). Durch eine Erhöhung des Grades lässt sich der Durchmesser verkleinern. Allerdings ist der Grad auch mit einem relativ hohen Aufwand verbunden. Erstens ist die Ressource „Verbindung“ relativ stark begrenzt (32k pro Computer) und zweitens müssen diese Verbindungen regelmäßig überprüft werden, um den Durchmesser zu erhalten. Dazu ist ein relativ hoher Aufwand an Nachrichten nötig.

Graph	Eingangs-/Ausgangs-Grad	Durchmesser
De-Bruijn	k	$\log_k N$
Trie	k+1	$2\log_k N$
Chord	$\log_2 N$	$\log_2 N$
CAN	2d	$1/2dN^{1/d}$
Pastry	$(b-1)\log_b N$	$\log_b N$
Classic Butterfly	k	$2\log_k N(1-o(1))$

Tabelle 2: Asymptotische Grad- / Durchmesser-Eigenschaften der verschiedenen Graphen (aus [43], N=Anzahl der Knoten, K=Verbindungs-Grad eines Knotens, d=Dimension des Graphen)

Ein noch wichtigeres Kriterium als der Durchmesser ist die durchschnittliche Distanz zwischen zwei beliebigen Knoten, da letztere direkt den Suchaufwand widerspiegelt (siehe Tabelle 3).

k	Moore		De-Bruijn		Chord		CAN		Classic Butterfly		Trie	
	D	Avg dist	D	Avg dist	D	Avg dist	D	Avg dist	D	Avg dist	D	Avg dist
2	-	17.9	20	18.3	-	-	Huge	Huge	31	22.4	-	-
3	-	11.7	13	11.9	-	-	-	-	20	14.7	40	-
4	-	9.4	10	9.5	-	-	1000	500	16	11.8	26	-
10	-	5.8	6	5.9	-	-	40	19.8	10	7.3	13	-
20	-	4.5	5	4.6	20	10	20	10.0	8	5.7	10	-
50	-	3.5	4	3.5	-	-	-	-	7	4.3	8	-
100	-	2.98	3	2.98	-	-	-	-	5	3.65	6	-

Tabelle 3: Durchmesser (D) und durchschnittliche Distanz(Avg. dist) von verschiedenen Netzen mit Grad K und N=10^6 Knoten, bei `–` liegen keine Daten vor (Tabelle aus [43])

Aus diesen Daten sieht man, dass Peer-to-Peer-Netze, die auf den De-Bruijn-Graphen oder auf dem Butterfly basieren, einen kleineren Durchmesser, eine kleinere durchschnittliche Distanz und einen geringeren Grad haben als die anderen Strukturen. Damit skalieren diese Netze besser (siehe II.5.2). Netze mit der Struktur des De-Bruijn-Graphen sind einfacher als solche, die auf einem Butterfly basieren, was vorteilhaft ist (siehe [41]).

Ein weiterer Punkt ist es, wie viele nicht überlappende Pfade es zwischen zwei beliebigen Knoten in einem Netz gibt. Dabei sind besonders diejenigen wichtig, welche durch einen Greedy-Routing-Algorithmus gefunden werden. Hierbei sind zwei Pfade nicht überlappend, wenn sie außer dem Start und dem Endknoten keine gemeinsamen Knoten haben. Solche unabhängigen Pfade stabilisieren einerseits das Netz bei Ausfällen beispielsweise gegen eine Teilung in zwei Netze, andererseits ermöglichen sie aber auch eine Erhöhung der Betrugssicherheit durch Redundanz (siehe III.2.1.1.1).

Ersteres kommt zustande, da die Kommunikation zwischen zwei Knoten oder zwei Netzteilen immer noch gewährleistet ist, wenn ein Knoten ausfällt. Würden alle Pfade über nur einen zwischen den Anfangs- und dem Endknoten liegenden Knoten gehen, so würde der Ausfall dieses, die Kommunikation zwischen diesen unmöglich machen. Hierdurch würde sich dieses in zwei getrennte Netze teilen.

Solche Pfade wurden in einem empirischen Vergleich zwischen Chord und einem De-Bruijn-Graphen(aus [43]) analysiert. Hierbei wurden vergleichbare Parameter für die Netze gewählt (Größe und Ein/Ausgangsgrad der Knoten).

Es hat sich dabei gezeigt, dass es in Chord wenige solche Pfade gibt (ca. 50% der Knoten hatten nur zwei solche Routen), während im De-Bruijn-Graphen relativ viele solcher Routen zur Verfügung stehen (über 90% der Knoten hatten neun verschiedene Pfade, während die restlichen sogar durch 10 Pfade miteinander verbunden waren, bei einem Ein-/Ausgangsgrad von 10).

Des Weiteren schwankt die Länge der alternativen Pfade unter Chord stark. Hierbei liegt die durchschnittliche Länge solcher Pfade weit über der durchschnittlichen Länge der Pfade, über die normalerweise geroutet wird. Bei De-Bruijn-Graphen ist dieses nicht so. Die durchschnittliche Länge der alternativen Pfade ist so gut wie konstant und liegt sehr nahe an der der Verbindung mit der kürzesten Entfernung (ca. um einen Hop länger). Diese durchschnittliche Länge ist sehr wichtig, wenn man die Pfade zur Absicherung gegen Betrug nutzen will, da man sie bei jeder Anfrage gebraucht.

Die hauptsächliche Schwachstelle von Netzen, die auf dem De-Bruijn-Graphen beruhen, ist die fehlende Robustheit gegen Unbalanciertheit. Wenn solch ein Netz aus der Balance gerät, so entartet das Netz schnell und erhöht dabei die durchschnittliche Distanz und den Durchmesser stark (siehe [42], [45]).

Der Unbalanciertheit des Netzes, wenn Knoten das Netz verlassen oder hinzukommen, kann man mit einem erhöhtem Aufwand entgegenwirken. Je höher diese Fluktuation jedoch ist, desto höher wird der Aufwand. Da aber eine geringe Fluktuation in dem hier vorgeschlagenem System zu erwarten ist, ist der Aufwand vertretbar und die Vorteile der De-Bruijn-Graphen überwiegen.

III.2.4. Zusammenfassung der Designentscheidungen

In dieser Arbeit soll ein Peer-to-Peer-Netz erstellt werden, in dem die Kontodaten von unabhängigen, (pseudo-)zufällig ausgewählten Identitäten gespeichert werden, damit der Kontoinhaber möglichst schwer die entsprechenden Daten fälschen kann (siehe III.2.2). Zusätzlich soll ein Konto immer von mehreren Identitäten gespeichert werden, um sowohl eine erhöhte Sicherheit gegen Fälschung als auch gegen den Ausfall(siehe II.5.1) zu bieten. Ähnliches gilt auch für das Finden derjenigen, die ein Konto speichern (siehe III.2.3.2). Als weitere Sicherheitsmaßname gegen Fälschung sollen sicherheitskritische Entscheidungen möglichst nicht vom Antragsteller selbst entschieden werden, sondern von einer neutralen Instanz (siehe III.2.1.1.3). Hinzu kommt ein durch Rekursion teilweise anonymes Routing (siehe III.2.1.1.2). In einer nicht implementierten hochsicheren Version werden noch Audits einbezogen. Um dieses alles zu ermöglichen wird ein De-Bruijn-Graph als Grundstruktur des Netzes gewählt.

III.3. Detailliertes Design

III.3.1. De-Bruijn-Graphen mit beliebiger Anzahl an Knoten

Der De-Bruijn ist eigentlich nur mit einer Knotenanzahl von K^n definiert, wobei K die Basis des De-Bruijn-Graphen ist und n eine beliebige natürliche Zahl. Im Folgenden wird n auch als der Level des Graphen bezeichnet. Ein Peer-to-Peer-Netz hat aber eine beliebige Anzahl an Computern. Deswegen muss man den eigentlichen De-Bruijn-Graphen so erweitern, dass er seine Eigenschaften nicht verliert, er aber auf beliebig vielen Elemente definiert ist.

Eine Möglichkeit wäre es, „überzählige“ Computer auf irgendeine Art und Weise zu sammeln. Dieses könnte man unbegrenzt tun, also ohne den Graphen in die nächste Größe zu überführen, auch wenn genügend Computer zur Verfügung stünden. Solch ein Verfahren skaliert jedoch nur so gut wie die Struktur, in der man sammelt. Dieses entspricht dem ursprünglichen Problem, ein skalierbares Peer-to-Peer-Netz zu konzipieren. Dieser Ansatz ist also nicht sinnvoll.

Man könnte aber den Graphen in den Folgegraphen überführen, wenn genügend Computer bereitstehen. Dieses würde aber das Wissen über die Gesamtzahl an Computern voraussetzen (globales Wissen), welches zu erlangen im Allgemeinen nicht skaliert.

Eine weitere Methode ist es, einen dynamischen Übergang zwischen den Graphen verschiedener Größe zu schaffen. Man lässt dabei zu, dass Knotenpunkte des Netzes sich auf verschiedenen Leveln befinden. Hierbei ist es jedoch schwer, die Eigenschaften des Graphen zu erhalten

III.3.1.1. Durchmesser bei Levelunterschieden

Der Durchmesser solch eines Netzes wird dabei von dem Knoten mit dem höchsten Level im Netz bestimmt. Daraus folgt, dass ein möglichst geringer Unterschied zwischen dem Knoten mit dem geringsten und dem mit höchstem Level bestehen sollte. Notwendig ist hierbei nur ein Unterschied von eins.

Eine erste Maßnahme, um die Unterschiede möglichst gering zu halten, ist folgende: Man fügt Computer, die sich mit dem Netz verbinden, an den Stellen ein, die den geringsten Level haben. Dieses kann man näherungsweise aber trotzdem mit guten Resultaten durch Mehrpunktproben oder durch deterministische Minimumsuche erreichen (siehe [42]).

Diese Methode funktioniert aber nur so lange, wie genügend Computer sich mit dem Netz neu verbinden, also solange das Netz wächst oder eine stabile Größe hat. Sobald die Anzahl der Teilnehmer im Netz sich verringert, wird ein Teil auf dem „alten“ Level bleiben, während ein anderer sein Level verringert. Hierdurch erhöht sich der Levelunterschied. Je schneller das Netz schrumpft, desto wahrscheinlicher ist dieses. Sich neu verbindende Computer immer an die Minimalstellen zu setzen, wird dieses Problem zwar mildern, aber nicht beseitigen, da es davon zu wenige gibt.

Um diesem entgegenzuwirken, kann man Knoten, die einen (erheblich) höheren Level als ihre Umgebung haben, zu einem erneuten Ausführen der Bootprozeduren veranlassen. Durch solch ein Verhalten kann man sicherstellen, dass zwischen zwei benachbarten Knoten maximal ein Levelunterschied von eins existiert. Dieses bedeutet für den ganzen Graphen einen maximalen Levelunterschied, der seinem Durchmesser entspricht. Der praktische Wert wird besser sein, jedoch nicht optimal (gleich eins). Allerdings tritt dabei nach der Theorie eine neue Sicherheitslücke auf. Einem Computer, dessen Besitzer bösen Absichten hat, wird dadurch ermöglicht durch das Netz zu wandern. Hierzu tritt dieser dem Netz bei und sammelt Informationen aus seiner Umgebung im Netz. Als nächsten Schritt wechselt er den Standort, eventuell gezielt mittels der gesammelten Informationen. Dieses tut er so lange, bis er eine Stelle im Netz erreicht hat, die seinen Ansprüchen genügt.

Dieses ist aber nur theoretisch eine neue Sicherheitslücke, da dieses praktisch auch ohne die im Protokoll vorgesehene Möglichkeit dazu realisierbar war. Dieses entsteht, da jeder Computer jederzeit das Netz verlassen kann. Des Weiteren kann sich so ein Computer auch jederzeit wieder mit dem Netz verbinden. Dieses kann man nicht verhindern, wenn man nicht verbietet, dass sich überhaupt jemand ins Netz integriert. Begründen lässt sich das dadurch, dass man die Identität eines Computers nicht gegen seinen Willen feststellen kann. Man braucht dessen Kooperation.

In einem weiteren Schritt kann man einen Knoten aus mehreren Computern bestehen lassen. Wenn ein Knoten genügend Computer enthält, so wird dieser in K (Basis des De-Bruijn-Graphen) Knoten geteilt, deren Level um eins höher ist als das des Ursprungsknotens. Wenn die Anzahl an Computern in den Knoten wieder sinkt, werden sie wieder zu einem Knoten niedrigeren Levels zusammengeführt.

Hierdurch können geringere Unterschiede, bis zu einer Differenz von einem einzigen Computer, erkannt werden, weswegen man die Obergrenze des Unterschiedes zwischen zwei benachbarten Knoten auf einen Computer festlegen kann. Dieses führt wiederum dazu, dass es im Graphen einen maximalen Unterschied von Computern pro Knoten gibt, der seinem Durchmesser entspricht. Dabei muss man aber zusammengehörige Knoten rechnerisch wieder auf dem Level zusammenführen, auf dem sich der Knoten mit dem niedrigsten Level befindet. Dementsprechend gibt es einen garantieren, maximalen Levelunterschied, der der aufgerundeten Division aus Durchmesser und maximaler Knotengröße entspricht. Bei einer maximalen Größe der Knoten, die leicht größer als der Durchmesser ist, entspricht dieses dem optimalen Wert von eins.

Dass ein Knoten aus mehreren Computern besteht, hat noch andere gewichtige Vorteile, die unter III.3.2 besprochen werden.

Mit den hier besprochenen Mitteln kann man einen sehr geringen bis optimalen Levelunterschied erreichen und somit den geringen Durchmesser der De-Bruijn-Graphen erhalten.

III.3.1.2. Verlinkung bei Levelunterschieden

Die eigentliche Regel bei der Verlinkung von Knoten im De-Bruijn-Graphen ist, dass man das höchstwertige Zeichen der ID eines Knotens löscht und ein Zeichen auf der anderen Seite hinzufügt.

In einem Netz, welches sowohl „räumlich“ als auch zeitlich verschiedene Level annehmen kann, müssen diese Regeln erweitert werden. Es muss entschieden werden, wie die Verlinkung aussehen soll, wenn benachbarte Knoten verschiedene Level haben. Dieses hat nämlich weit reichende Konsequenzen für die Anzahl an überlappungsfreien Wegen zwischen zwei Knoten.

Hierbei gibt es zwei konsistente Modelle, wenn man alte Nachbarschaften bei einem Levelwechsel von einem oder von einer Gruppe von Knoten ausnutzen will. Bei beiden geht man von einem beliebig, aber endlich großen De-Bruijn-Graphen aus. Bei dem einen Modell werden alle Knoten des übergeordneten Graphen zu einem Knoten zusammengefasst, deren niederwertigste Zeichen übereinstimmen. Dabei bestimmt der Level eines Knotens, wie viele Zeichen übereinstimmen müssen und dementsprechend die ID des Knotens bilden. Daraus folgt wiederum, dass die ID eines Knotens an den höherwertigen Zeichen erweitert oder gekürzt wird.

Das zweite Modell ist nach demselben Prinzip aufgebaut, bloß dass die ID eines Knotens die höchstwertigen Zeichen des übergeordneten Graphen darstellt. Deswegen muss auch die ID in Richtung der niederwertigen Zeichen erweitert werden.

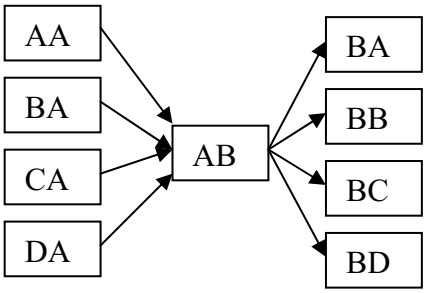

Abbildung 6: Ausschnitt eines Level-2-De-Bruijn-Graphen der Basis 4

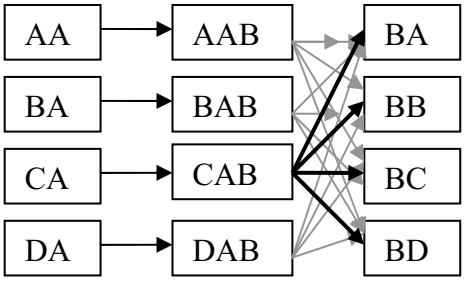

Abbildung 7: Kanten im De-Bruijn-Graphen bei einem Knoten mit höherem Level, bei Erweiterung der ID bei den höherwertigen Zeichen

Man nehme jetzt das Beispiel eines Graphen mit der Basis vier an, dessen Zeichen A,B,C und D sind. Hierbei nehme man weiterhin an, dass sich erst einmal alle Knoten auf dem Level zwei befinden. Jetzt untersuche man den Knoten AB in den beiden Modellen, wenn er sein Level verändert. Normalerweise sind seine Vorgänger die Knoten AA, BA, CA und DA. Seine Nachfolger sind BA, BB, BC und BD (siehe Abbildung 6). Bei den Abbildungen sollte man aber nicht vergessen, dass die anderen Knoten (nicht AB) sowohl weitere Ausgangskanten, als auch weitere Eingangskanten haben.

Bei der ersten Möglichkeit, bei der die ID nach links erweitert wird, spaltet sich der Knoten AB zu den Knoten AAB, BAB, CAB und DAB auf (siehe Abbildung 7). Die Verlinkungsregel ist dabei, dass bei Verbindungen zu Knoten eines höheren Levels ein Zeichen hinzugefügt wird, aber kein Zeichen gelöscht wird.

Die Verlinkungsregel zu Knoten eines niedrigeren Levels ist, dass man zwei Zeichen löscht und ein neues Zeichen hinzufügt.

Durch diese Regeln ist der Ausgangsgrad der Knoten konstant, trotz Levelunterschiede, allerdings schwankt der Eingangsgrad eines Knotens zwischen eins und K^2. Das Maximum wird erreicht, wenn alle Vorgänger eines Knotens einen Level höher sind als der Knoten selber. Das Minimum wird dann erreicht, wenn der Knoten ein Level höher ist als seine Vorgänger.

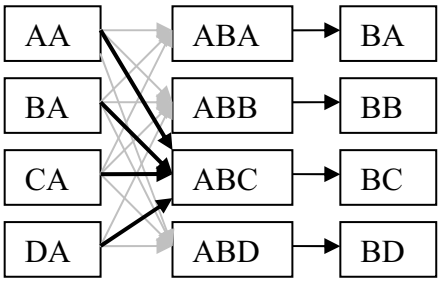

Abbildung 8: Kanten im De-Bruijn-Graphen bei einem Knoten mit höherem Level, bei einer Erweiterung der ID bei den niederwertigen Zeichen

Bei der zweiten Möglichkeit, bei der die ID nach rechts erweitert wird, spaltet sich der Knoten AB in die Knoten ABA, ABB, ABC und ABD auf. Hierbei ist die Verlinkungsregel für Knoten eines höheren Levels, dass man ein Zeichen löscht und zwei Zeichen hinzufügt.

Nach der Regel, wie man Knoten niedrigeren Levels verlinkt, löscht man ein Zeichen, ohne ein neues Zeichen einzuschieben.

Bei diesen Regeln bleibt der Eingangsgrad konstant, während der Ausgangsgrad zwischen eins und K^2 schwankt.

Die starke Schwankung der Ein- bzw. Ausgangsgrade in den beiden Modellen wird noch unter III.3.3 auf eine lineare Schwankung reduziert.

Das Routing erfolgt in beiden Varianten wie folgt: Erst wird die Zieladresse maskiert, also bei der ersten Möglichkeit die Level niederwertigsten Zeichen stehen gelassen, während es bei der zweiten Möglichkeit die entsprechend höchstwertigen Zeichen sind. Hierauf wird das längste Präfix der maskierten Zieladresse herausgesucht, welches ein Postfix von der Knoten-ID ist. Das darauf folgende Zeichen gibt normalerweise den Weg an, auf dem weiter gerouted werden soll. Wenn ein zweites Zeichen benötigt wird, so wird das zweite darauf folgende Zeichen benutzt.

Wenn es überhaupt Levelunterschiede gibt, gilt die Aussage bei beiden Möglichkeiten nicht mehr, dass es von jedem Knoten des De-Bruijn-Graphen K Wege zu jedem anderen Knoten gibt, welche sich mit großer Wahrscheinlichkeit nicht überlappen. Im ersten Fall gibt es Knoten, die nur auf einem Weg erreicht werden können, während es im zweiten Fall Knoten gibt, die nur auf einem Weg andere Knoten erreichen können.

Im ersten Fall ist es relativ schwierig und von zweifelhaftem Sinn, etwas dagegen zu tun. In diesem Fall müssen nämlich die Vorgänger eines Knotens die zusätzlichen Kanten generieren. Diesen Knoten ist aber nicht unbedingt zu trauen. Zusätzlich muss dieses während des gesamten Routingvorgangs beachtet werden, da dieser Sonderfall erst im letzten „Hop“ auftritt.

Im zweiten Fall muss der Knoten eines höheren Levels selber für den zusätzlichen Ausgangsgrad sorgen. Dieses ist nur wichtig, wenn der Knoten selber ein neues Paket redundant senden will. Das heißt, dass die Routingregeln immer gelten, sobald eine Nachricht den Ursprungsknoten verlassen hat. Dieser könnte aber noch „alte“ Verbindungen aufrechterhalten, zu Knoten, die jetzt keine Nachfolger mehr sind, es aber auf einem niedrigeren Level waren. In diesem Beispiel würden sowohl ABA, ABB, ABC als auch ABD die Verbindungen zu BA, BB, BC und BD behalten. Beim normalen Routing werden diese aber nicht beachtet, nur wenn der Knoten selber ein Paket redundant losschicken will, benutzt er diese Verbindungen. Aus diesem Grund wird die zweite Möglichkeit in dieser Arbeit benutzt.

III.3.2. Knoten aus mehreren Computern

Wenn, wie unter III.3.1.1 besprochen, ein Knoten aus mehreren Computern besteht, so hat das nicht nur Einfluss darauf, wie groß die Levelunterschiede im Netz sind. Ein genauso wichtiger Einfluss wird auf die Ausfallssicherheit der Knoten ausgeübt. Die Wahrscheinlichkeit, dass ein Knoten nicht komplett ausfällt, steigt exponentiell mit der Anzahl an Computern in diesem. Dabei ist vom Normalfall auszugehen. Dieses bedeutet, dass kein gezielter Angriff gegen einen Knoten durchgeführt wird oder dass eine Abhängigkeit der Computer untereinander existiert. Deswegen sind die Ausfälle auch unabhängig.

Dieses hat wiederum großen Einfluss auf die Verbindungen zwischen den Knoten, da diese auch im gleichen Maße sicherer werden. Im Allgemeinen wird das Netz dadurch stabiler und einfacher zu reparieren sein, wenn nicht der unwahrscheinliche Fall eintritt, dass ein ganzer Knoten ausfällt. Um dieses aber verwirklichen zu können, müssen die Mitglieder eines Knotens entweder alle Mitglieder aller direkten Nachfolge- oder aber die aller direkten Vorgängerknoten kennen.

Ein weiterer Punkt ist durch die nötige Redundanz der Speicherung gegeben. Redundanz im Sinne von Betrugssicherheit benötigt Anonymität, während dieses bei der Redundanz, welche gegen den Ausfall von Computern benutzt wird, nicht benötigt wird. Hierbei ist diese sogar kontraproduktiv, da die Computer miteinander kommunizieren müssen, um die Daten konsistent zu halten. Wenn die Computer gegenseitig ihre IPs kennen, so können sie direkt miteinander kommunizieren. Wenn die Computer aber anonym sind, so müssen sie über das Netz kommunizieren, was ein Vielfaches an Nachrichten und damit Bandbreite, als auch ein Vielfaches an Zeit beansprucht.

Zusätzlich würde man ein zweites anonymes, sicheres Netz entwickeln müssen, wenn man versuchen würde, die Identität (IP) der Computer innerhalb eines Knotens zu verbergen. Dieses entspricht dem ursprünglichen Problem der Entwicklung eines Peer-to-Peer-Netzes. Dementsprechend wäre es sinnvoller, Knoten nur aus einem Computer bestehen zu lassen und die dadurch entstehenden Probleme anderweitig zu lösen.

Außerdem ist es wenig sinnvoll, Betrugssicherheit auf Komponenten aufzubauen, deren Existenz nicht gesichert ist. Vielmehr gehört zur Betrugssicherheit auch eine relativ hohe Ausfallssicherheit. Es muss ja davon ausgegangen werden, dass die entsprechenden Dienste, welche einen Betrug verhindern, ständig verfügbar sind.

Dementsprechend kann man ein anonymes Netz bauen, in dem ein Inhalt auf mehreren Knoten gespeichert wird. Die Computer dieser Knoten kennen sich gegenseitig nicht. Hierdurch wird die Redundanz gegenüber Betrugsversuchen erreicht.

Für die Ausfallssicherheit ist aber zusätzliche Redundanz erforderlich. Diese wird dadurch realisiert, dass ein Knoten aus mehreren Computern besteht. Diese können effizient miteinander kommunizieren, da sie sich gegenseitig kennen.

III.3.3. Entkoppelung des Verlinkungsgrades von der Zeichenanzahl

Ein De-Bruijn-Graph wird durch seine Basis gekennzeichnet. Die Basis (hier mit K bezeichnet) legt sowohl den Verbindungsgrad (hier G) fest als auch die Anzahl an Zeichen (hier Z), die an einer Stelle einer Knoten-ID möglich sind.

$$Z=K=G$$

Wenn so ein Graph wächst, müssen Knoten zerteilt werden. Bei einem Knoten, der geteilt wird, wird im einfachsten Fall ein Zeichen zu der Ursprungs-ID hinzugefügt. Damit auch nach der Teilung ein komplettes und konsistentes Netz vorhanden ist, muss ein Knoten

in Z neue Knoten geteilt werden. Wenn ein Knoten minimal aus einer bestimmten Anzahl an Computern (MIN) bestehen soll, kann ein Knoten frühestens geteilt werden, wenn MIN mal Z Computer im Knoten vorhanden sind. Es entstehen ja Z neue Knoten und jeder muss mindestens MIN Computer enthalten.

Hieraus kann man sehen, dass Z einen starken Einfluss auf die Skalierbarkeit hat, beziehungsweise G durch Z begrenzt wird. Dieses würde beispielsweise aber wiederum den Durchmesser vergrößern. Diese Koppelung von Z und G ist aber nicht notwendig. Prinzipiell kann man die Anzahl an Zeichen (Z) und damit den Grad der Teilung eines Knotens sowie die Anzahl an Kanten (G) frei bestimmen. Dabei muss nur G eine Potenz von Z sein und Z größer als 1 sein.

$$G=Z^M$$

(sei M eine beliebige natürliche Zahl (>0))

Dadurch ändern sich die Linkregeln nur unbedeutend. Statt eines Zeichens werden M Zeichen im Normalfalle gelöscht und genauso viele andere hinzugefügt. Bei einem Levelunterschied besteht aber jetzt kein Zwang mehr, M Zeichen auf diese Art zu verarbeiten, sondern es reicht ein Zeichen aus. Deswegen schwankt der Ein-/Ausgangsgrad bei dem hier gewählten Verlinkungsmodell auch nicht mehr zwischen eins und G^2, sondern nur noch zwischen

$$Z^{M-1}=G/Z \quad \text{und} \quad Z^{M+1}=G*Z.$$

Auch die Knotengröße schwankt nur noch zwischen MIN und MIN*Z. Zusätzlich sind auch G und Z nicht mehr direkt gekoppelt. Hierdurch hat man also Designspielraum und/oder Skalierbarkeit gewonnen.

Um ein optimales Ergebnis dieser Maßnahme zu erzielen, sollte Z gleich zwei sein, da dieses die kleinste mögliche Zahl ist und somit eine minimale Schwankung der Knotengröße und des Verlinkungsgrades erreicht wird.

Ab hier bezieht sich das Level nicht mehr auf die Basis des De-Bruijn-Graphen, sondern auf Z.

Ursprünglich (nur Ersetzung der Zeichen)

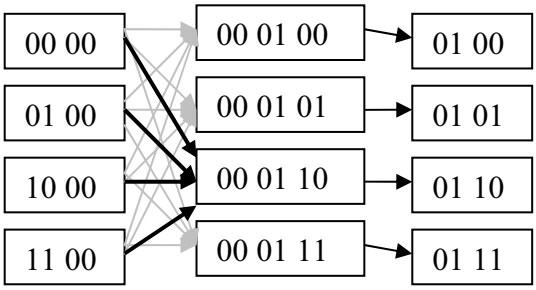

Abbildung 9: Kanten im De-Bruijn-Graphen bei einem Knoten mit höherem Level und einer Ersetzung der einzelnen Zeichen durch eine Zeichenfolge

Entkoppelt

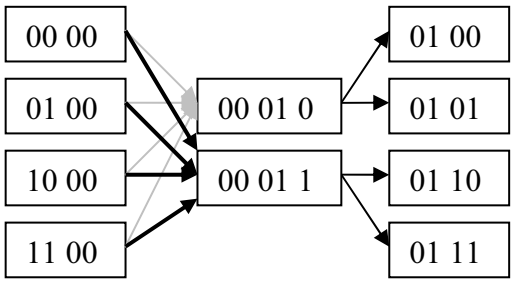

Abbildung 10: Kanten im De-Bruijn-Graphen bei einem Knoten mit höherem Level und der Entkoppelung des Verlinkungsgrades von der Zeichenanzahl

(siehe Abbildung 8 bloß A=00 B=01 C=10 D=11)

III.4. Entwurf der Protokolle

III.4.1. Das Peer-to-Peer-Netz

Der Autor dieser Arbeit hat das Peer-to-Peer-Netz in mehrere Unterprotokolle geteilt, um eine erhöhte Übersichtlichkeit zu erreichen:

1. Bootprotokoll (siehe III.4.1.1),
2. Knotenmanagementprotokoll (siehe III.4.1.2),
3. Verbindungsmanagement-Protokoll (siehe III.4.1.3) und
4. Routingprotokoll (siehe III.4.1.4)

Dabei werden nicht alle Datenstrukturen von nur einem der Protokolle benutzt. Ein Teil wird direkt oder indirekt auch von mehreren Protokollen benutzt. Man kann sie aber nach dem hauptsächlichen Gebrauch zuordnen. In diesem Kapitel wird das Protokoll selbst erklärt, detailliertere Informationen findet man in dem kommentierten Sourcecode.

Das Bootprotokoll sorgt dafür, dass sich neue Computer an einer passenden Stelle im Netzwerk eingliedern, wobei das Hauptaugenmerk auf der Balancierung des Netzwerkes liegt. Dementsprechend ist ein Knoten passend, der weniger Computer als seine Umgebung beherbergt und/oder ein niedrigeres Level als diese hat. Hierbei benutzt das Protokoll sowohl das Routing- als auch das Verbindungsmanagement- sowie das Knotenmanagement-Protokoll (das Bootprotokoll siehe III.4.1.1).

Das Knotenmanagementprotokoll ist für die Verwaltungsaufgaben innerhalb eines Knotens zuständig. Dementsprechend verwaltet es die lokal auf jedem Computer vorgehaltene Liste der Mitglieder eines Knotens. Hier ist neben dem Hinzufügen und Entfernen von Elementen auch das Abgleichen der Listen zwischen verschiedenen Mitgliedern eines Knotens zu nennen. Zusätzlich muss erkannt werden, wenn Mitglieder den Knoten verlassen, ohne sich abzumelden. Zu guter Letzt kommt noch das Teilen (Split) und Zusammenführen (Join) von Knoten auf dieser Ebene hinzu, wenn die Grenzen der Mitgliederanzahl eines solchen verletzt werden (das Protokoll siehe III.4.1.2).

Das Verbindungsmanagement-Protokoll ist für die Verbindungen zwischen zwei benachbarten Knoten zuständig. Folglich verwaltet es Listen der Mitglieder von benachbarten Knoten. Diese werden mit den Listen der eigentlichen Knoten konsistent gehalten. Deswegen müssen die Listen aktualisiert werden, wenn Nachbarn Ereignisse wie neue Mitglieder, Splits oder ähnliches bekannt geben. Folglich benachrichtigt das Protokoll die Nachbarn, wenn Änderungen im eigenen Knoten stattfinden. Eine weitere Aufgabe ist es, Verbindungen bei Splits und Joins umzuordnen, damit die Verbindungen denen im De-Bruijn-Graphen entsprechen. Zu guter Letzt hat es noch die Aufgabe (ähnlich wie das Bootprotokoll), die Levelunterschiede im Netz möglich gering zu halten, indem es Unterschiede in der Computeranzahl zwischen verschiedenen Knoten erkennt und diese ausgleicht (siehe III.3.1.1 & III.3.2) (das Protokoll siehe III.4.1.3).

Das Routingprotokoll sorgt für die Beförderung von Paketen zu einem Zielknoten oder Computer. Hierbei werden die Verbindungen des Knoten- und Verbindungsmanagement-Protokolls benutzt. Hinzu kommt noch die Umleitung von Paketen, um Anonymität zu gewährleisten (das Protokoll siehe III.4.1.4).

Im Folgenden wird aus praktischen Gründen nicht zwischen dem Computer, den Programmen auf dem Computer und dem Besitzer des Computers unterschieden.

III.4.1.1. Das Bootprotokoll

Die Hauptaufgabe des Bootprotokolls ist es, Levelunterschiede im Netz möglichst gering zu halten, aber zusätzlich ist es sicherheitsrelevant. Es soll möglichst schwer gemacht werden, Informationen über das Netz zu sammeln, da diese helfen können, Angriffe zu planen oder durchzuführen (siehe III.2.1.1.2). Des Weiteren soll auch verhindert werden, dass derjenige, der ins Netz eingegliedert werden will (künftig der Anfrager genannt), entscheiden kann, wo er eingegliedert wird.

Für das Minimieren der Levelunterschiede ist eine Suche im Netz nach „Minimumknoten" notwendig. Unter dem „Minimumknoten" wird hier der Knoten verstanden, der aus den wenigsten Computern besteht, nachdem alle Knoten auf einem Level zusammengeführt wurden. Um den Aufwand zu verringern wird allerdings nicht global nach diesem gesucht, sondern es wird nach einem lokalen Minimumknoten gesucht.

Diese Suche kann entweder von demjenigen, der die Anfrage stellt, selber durchgeführt werden oder dezentral vom Netz. Im Anfrager kann so eine Suche zwar leichter implementiert werden, allerdings ist dafür eine Offenlegung einer Reihe von Knotendaten inklusive der IPs notwendig. Dieses widerspricht aber der Anonymitätsforderung aus III.2.1.1.2. Zusätzlich kann der Anfrager dann selber entscheiden, wo er im Netz eingegliedert wird, was für Angriffe genutzt werden kann (siehe III.2.1.1.3).

Deswegen wurde eine Suche durch das Netz selber gewählt. Hierbei kann der Anfrager bei korrektem Ablauf des Protokolls nicht entscheiden, wo er eingegliedert wird und erhält auch nur Informationen über die Umgebung, in der er eingegliedert werden soll. Dieses erhöht den Aufwand der Informationsbeschaffung erheblich gegenüber einer direkten Anfrage nach entsprechenden Daten an einen Knoten. Dieses ist auf jeden Fall ein guter Schutz gegenüber Einzeltätern (siehe II.4.3.1).

Dementsprechend verläuft das Protokoll so, dass ein Anfrager ein vorher bekanntes Mitglied des Netzes kontaktiert und diesem den Verbindungswunsch mitteilt („WTC"= „**W**ant **T**o **C**onnect"). Dieser bestätigt daraufhin, dass er sich um die Anfrage kümmert (OK). Die Bestätigung ist hierbei notwendig, da es nicht geklärt ist, ob der Kontaktierte wirklich ein Mitglied des Netzes ist. Bei dieser Anfrage wird auch die nach außen sichtbare IP des Anfragers festgestellt. Wenn entweder die Verbindung nicht zustande kommt oder die Antwort (OK) nach einer bestimmten Zeit noch nicht angekommen ist, werden weitere Computer aus einer Liste ausprobiert (siehe Abbildung 11: 1).

Als nächstes wird die Anfrage (WTC) durch das Routingprotokoll (siehe III.4.1.4) an einen zufälligen Knoten im Netz weitergeleitet (siehe Abbildung 11: 2). Dieses wird vorgenommen, da davon auszugehen ist, dass die meisten dieser Anfragen an einer relativ kleinen Anzahl an Computern ankommen. Wenn immer von diesen Punkten aus gesucht werden würde, würde sich dieses negativ auf die Levelunterschiede im Netz auswirken. Dieser Teil des Protokolls ist nach der Meinung des Autors kaum sicherheitsrelevant.

Nachdem jetzt ein zufälliger Startpunkt gewählt ist, fängt die eigentliche Minimumsuche an. Hierfür prüft der Ringmaster (siehe III.4.1.2) des Knotens, der aktuell die Anfrage bearbeitet, welcher der Knoten in seiner Umgebung levelbereinigt die geringste Anzahl an Computern beinhaltet. Diese Information wird vom Verbindungsmanagement-Protokoll (siehe III.4.1.3) bereitgestellt. Hierbei ist es vorteilhaft, wenn nicht nur die Nachfolgeknoten, sondern auch die Vorgängerknoten in die Suche einbezogen werden. Ansonsten kann hierdurch ein ungleichmäßiges Auffüllen der Knoten entstehen, welches wieder ausgeglichen werden muss (siehe III.4.1.3.3). Die Anfrage wird an den Ringmaster des entsprechenden Knotens weitergeleitet. Hat der aktuelle Knoten aber selber weniger Computer als seine Umgebung und kann noch Mitglieder aufnehmen, wird zum nächsten Schritt übergegangen (siehe Abbildung 11: 3). Die Anzahl der Computer im eigenen Knoten wird vom Knotenmanagementprotokoll (siehe III.4.1.2) erfragt. Wenn durch den Suchalgorithmus der eigene Knoten ausgewählt wurde, um den Anfrager hinzuzufügen,

dieses aber daran scheiterte, dass zu viele Computer bereits im Knoten vorhanden waren, wird die Anfrage erneut an einen zufälligen Ort im Netz geroutet.

Der nächste Schritt besteht aus der Zusammenstellung der Daten, die den Knoten charakterisieren (durch die anderen Protokolle), welche daraufhin an den Anfrager gesendet werden, mit der Aufforderung sich in dem entsprechenden Knoten einzugliedern (CH= „**C**onnect **H**ere“) (siehe Abbildung 11: 4).

Durch diese Vorgehensweise werden auch die Möglichkeiten von normalen kriminellen Gruppen (siehe II.4.3.2) eingeschränkt, allerdings können sie das Protokoll in bestimmten Situationen aushebeln. Sie können nämlich Mitglieder gezielt an alle jene Stellen schicken, an denen schon Mitglieder sitzen. In dem Fall muss das Mitglied, welches sich an der entsprechenden Stelle befindet, einfach simulieren, dass es eine Anfrage bekommen und entschieden hat den Computer hinzuzufügen.

Ein wichtiger Punkt in diesem Protokoll ist, dass so gut wie keine Statusinformationen weitergegeben werden müssen, die dafür benutzt werden könnten, an eine gewünschte Stelle zu gelangen. Die einzige Statusinformation dieses Protokolls ist, ob das Verwürfeln des Startpunktes (siehe Abbildung 11 : 2) schon stattgefunden hat oder nicht. Dieser Status ist aber so gut wie nicht sicherheitsrelevant.

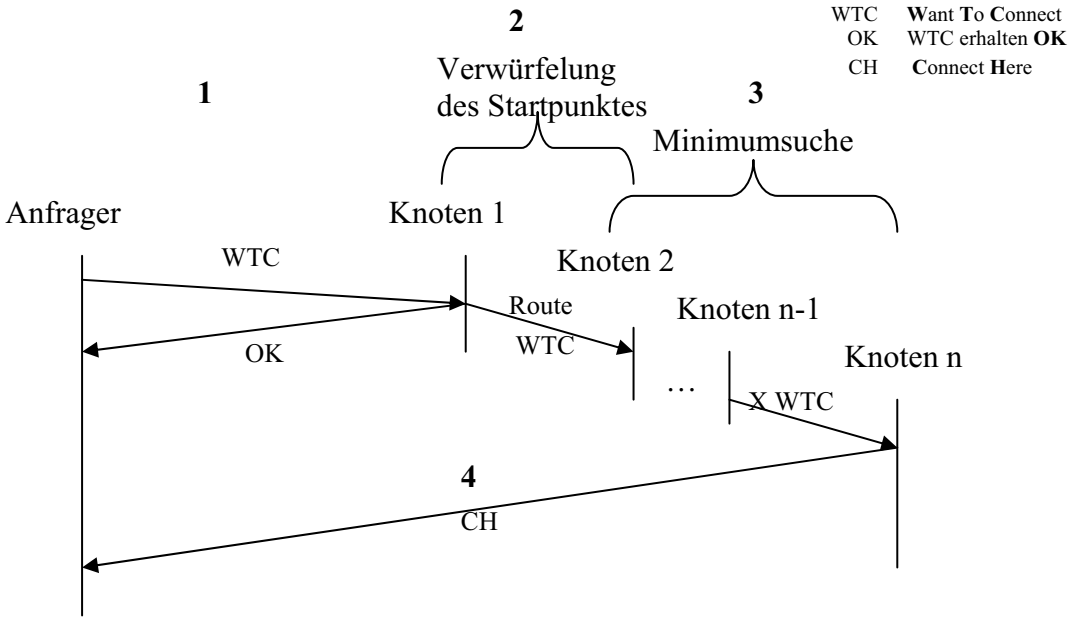

Abbildung 11: Schematischer Verlauf des Bootprotokolls über die Zeit

III.4.1.2. Das Knotenmanagementprotokoll

Ein Knoten besteht aus einer Anzahl Computer, welche im Folgenden auch Mitglieder genannt werden. Die Anzahl hat dabei eine Obergrenze. Die Computer sind einander bekannt (siehe III.3.2) und sind voll vermascht. Dieses wurde gewählt, da es sich um eine relativ kleine und begrenzte Anzahl an Elementen handelt. Der dafür nötige Aufwand hält sich in solch einer Gruppe in Grenzen. Allerdings bietet diese Methode den Vorteil einer sehr direkten und effizienten Kommunikation (siehe [48]). Deswegen ist der Autor der Meinung, dass die Vorteile die Nachteile (den Aufwand) überwiegen.

Bei den Texten, die die Abkürzungen und Paketnamen erklären, wird eine abweichende Groß- und Kleinschreibung benutzt, mit der die Zusammensetzung der Abkürzungen verdeutlicht werden soll. Hierbei werden nur jene Buchstaben groß geschrieben, die die Abkürzungen bilden.

Um die Parallelität und damit die Komplexität bei der Bearbeitung von Ereignissen zu verringern, werden die Mitglieder anhand einer Zufallszahl (Ring-Position genannt (RP)) geordnet. Der Computer mit der kleinsten Zufallszahl wird hier als „Ringmaster" bezeichnet. Die Vergabe und der weitere Gebrauch der Zufallszahl werden in den folgenden Abschnitten genauer beschrieben.

Um ein Mindestmaß an Sicherheit zu gewährleisten, werden nur Pakete und Verbindungen bearbeitet, die von einem bekannten Mitglied des Knotens kommen. Ausnahmen von dieser Regel sind nur gewisse Pakete bei der Zusammenführung zweier Knoten, welche gerouted werden müssen („**K**notenmanagement-**W**ant-to-**J**oin" = KWJ & „**K**notenmanagement-**J**oin-**O**k" = KJO, siehe III.4.1.2.4). Wenn ein Paket von einem unbekannten Computer kommt, wird dieses ignoriert und gelöscht. Verbindungen von unbekannten Computern werden eine Zeit lang aufbewahrt und dann geschlossen, wenn sie nicht in dem entsprechenden Zeitraum einem neuen Mitglied zugeordnet werden können.

Der Autor dieser Arbeit geht bei dem Protokoll von Sekunden genau synchron gehenden Uhren aus. Dieses lässt sich beispielsweise mittels ntpdate erreichen und stellt deswegen nur eine sehr geringe Einschränkung dar.

III.4.1.2.1. Verwaltung von Mitgliederlisten

Das Bootprotokoll teilt dem Knotenmanagement-Protokoll mit, wenn ein Computer dem Knoten hinzugefügt werden soll. Hierauf würfelt der Ringmaster die Ring-Position des neuen Mitgliedes und fügt die Daten in die lokale Liste ein. Zusätzlich teilt er allen anderen Mitgliedern des Knotens alle relevanten Daten (inkl. RP) des Neuen in einer „Neues-Mitglied"-Nachricht (KNM=„**K**notenmanagement-**N**eues-**M**itglied"-Nachricht) mit. Außerdem fügt er die Mitgliederliste des Knotens zu dem „Connect-Here"-Paket (CH) des Bootprotokolls hinzu. Zu guter Letzt wird das Verbindungsmanagement-Protokoll über den Neuzugang informiert.

Wenn ein Mitglied den Knoten verlassen will, weil es beispielsweise offline geht, so teilt es dieses allen anderen Mitgliedern mit, die die entsprechenden Daten daraufhin löschen. Hierbei benutzt er eine KML-Nachricht (KML= „**K**notenmanagement-**M**itglied-**L**eaving"-Nachricht)

III.4.1.2.2. Überprüfung und Konsistenthaltung der Mitgliederlisten

Zum Konsistenthalten der Mitgliederlisten und um Mitglieder entfernen zu können, die den Knoten verlassen haben, ohne dieses den anderen Mitgliedern mitzuteilen, sendet jeder Computer eines Knotens regelmäßig eine Keep-Alive-Nachricht (KKA= „**K**notenmanagement-**K**eep-**A**live"-Nachricht) an das Mitglied, welches die nächst kleinere Ringposition hat. Ausnahme ist nur der Ringmaster, welcher an den Computer mit der höchsten Ringposition seine Keep-Alive-Pakete schickt. Dieses System hat eine gewisse Ähnlichkeit mit dem, welches in „One Hop lookups for peer-to-peer overlays" ([48]) beschrieben wird. In solch einer KKA-Nachricht wird eine Prüfsumme über die vorhandenen Mitglieder und der Level des Knotens mitgeschickt.

III.4.1.2.2.1. Entfernen von nicht reagierenden Mitgliedern

Wenn auf eine Keep-Alive-Nachricht nach einer bestimmten Zeit keine Antwort zurückgekommen ist, werden weitere Keep-Alive-Nachrichten an das entsprechende Mitglied geschickt, auch wenn dieser nicht mehr derjenige ist, an welchen dem Standart gemäß die KKA-Nachrichten gesendet werden. Wenn weiterhin keine Antwort auf irgendeins der Pakete kommt, so wird in regelmäßigen Abständen versucht, die Verbindung neu aufzubauen und über diese dann wiederum eine KKA-Nachricht zu verschicken. Wenn nach einer gewissen weiteren Zeit niemand geantwortet hat, so wird der entsprechende Computer für „tot“ erklärt. Es wird eine KML-Nachricht an alle Mitglieder geschickt, dass der entsprechende Rechner den Knoten verlassen hat und dementsprechend wird der zu diesem gehörende Eintrag aus allen Mitgliederlisten des Knotens gelöscht.

Wenn aber doch, zu irgendeinem Zeitpunkt vor dem Löschen, eine Antwort auf eine der ausgesandten Keep-Alive-Nachrichten zurückkommt, so wird der ganze Vorgang zurückgesetzt.

III.4.1.2.2.2. Konsistenthaltung der Mitgliederlisten

Ein Mitglied, welches solch ein Keep-Alive-Paket erhält, prüft, ob die Daten mit den eigenen übereinstimmen oder nicht. Wenn sie einander entsprechen, so wird eine Nachricht zurückgesandt, dass die Daten übereinstimmen (KOK = „**K**notenmanagement-Keep-Alive-**OK**“). Wenn die Level nicht übereinstimmen, so werden entweder die Daten des letzten Teilens (KSP= „**K**notenmanagement-**SP**lit“-Nachricht) oder die der letzten Zusammenführung (KDJ= „**K**notenmanagement-**D**o-**J**oin“-Paket) erneut gesendet. Zusätzlich wird ein KOK-Paket verschickt, um den Erhalt der KKA-Nachricht zu bestätigen.

Wenn die Prüfsumme zwei Mal hintereinander nicht identisch ist, so wird die lokale Mitgliederliste an das entsprechende Mitglied gesandt („**K**notenmanagement-**WR**ong-checksum“-Nachricht =KWR). Dieses vergleicht daraufhin seine eigene mit der fremden Mitgliedsliste.

An jene Mitglieder, welche in der eigenen, aber nicht in der fremden Liste auftauchen, wird eine KKA-Nachricht versandt. Zusätzlich wird eine KNM-Nachricht (siehe III.4.1.2.1) an den geschickt, von dem das KWR-Paket stammte.

Zu Mitgliedern, welche in der fremden, aber nicht in der eigenen Liste auftauchen, wird versucht, eine Verbindung aufzubauen. Wenn das geschehen ist, werden sie als neues Mitglied akzeptiert. Zusätzlich wird eine KKA-Nachricht über diese Verbindung gesandt. Die weitere Bearbeitung der KKA-Nachrichten wurde unter III.4.1.2.2.1 beschrieben. Wenn der Verbindungsaufbau fehlschlägt, so wird eine KML-Nachricht (siehe III.4.1.2.1) an denjenigen geschickt, von dem die KWR-Nachricht stammte.

III.4.1.2.3. Teilung eines Knotens (Split)

Das Teilen eines Knotens ist sehr einfach. Der Ringmaster prüft regelmäßig die Größe des Knotens. Wenn diese einen bestimmten Wert überschreitet, ein Mindestabstand zum letzten Split oder Join eingehalten wurde und auch das Verbindungsmanagement-Protokoll zustimmt, werden an alle Mitglieder zwei Listen geschickt, welche die zwei neuen Knoten repräsentieren (KSP-Paket=“**K**notenmanagement-**SP**lit“-Paket).

Der Wert lässt sich durch das Minimum errechnen. Hierzu muss man dieses verdoppeln, damit es auch nach dem Split nicht unterschritten wird. Damit nicht nach einem Split gleich wieder ein Join folgt, weil nur ein Mitglied einen der beiden Knoten verlassen hat,

wird außerdem ein zusätzlicher Faktor eingebaut. Dementsprechend ist das Maximum beispielsweise gleich:

Minimum*2 (Knoten) *1,5 (Schwankungssicherheit)

Hierbei muss der Mindestabstand eingehalten werden und ist die Zustimmung des Verbindungsmanagement-Protokolls notwendig, um nicht die Vorraussetzung zu verletzen, dass zwischen zwei benachbarten Knoten maximal ein Levelunterschied von eins bestehen darf. Durch den Abstand hat das VMP Zeit, alle Informationen bei den Nachbarn sicher zu korrigieren und, wenn nötig, eine Ebnung zu erreichen.

Je nachdem, welchem der neuen Knoten man zugeordnet ist, wird die Knoten-ID verändert und alle Mitglieder entfernt, welche zum anderen Knoten gehören. Zusätzlich wird noch das Verbindungsmanagement-Protokoll informiert, dass der Knoten sich geteilt hat und in welchem neuen Knoten sich der eigene Computer befindet.

Die IPs der Mitglieder des anderen Knotens werden aber noch eine bestimmte Zeit aufbewahrt und falls einer von diesen den Split noch nicht mitbekommen hat, wird diesem die Splitnachricht erneut gesendet. Wenn die KSP-Nachricht zu einem Mitglied verloren geht, welches im gleichen Knoten verbleibt, so übernimmt der Keep-Alive-Teil des Protokolls die Berichtigung. Um doppelt gesendete Nachrichten zu erkennen, enthalten alle Splitnachrichten eine Timestamp ihres ursprünglichen Versandes. Dementsprechend kann eine nicht aktuelle Nachricht dadurch erkannt werden, dass diese einen kleineren oder gleich großen Zeitstempel besitzt wie die letzte gültige Nachricht. Hierbei ist nicht wirklich eine sekundengenaue Synchronisation nötig, allerdings müssen auch nach einem Wechsel des Ringmasters künftige Ereignisse eine größeren Zeitstempel haben als das letzte Event des ausgeschiedenen Ringmasters.

III.4.1.2.4. Zusammenlegung zweier Knoten (Join)

In einem Knoten (hier Anfrageknoten genannt) prüft dessen Ringmaster in regelmäßigen Abständen die Größe des Knotens. Wenn diese zu klein ist und sowohl ein Mindestabstand zum letzten Split oder Join eingehalten ist als auch das Verbindungsmanagement-Protokoll zustimmt, routet er eine „Want to Join“ Nachricht (KWJ= „**K**notenmanagement-**W**ant-**T**o-**J**oin“-Nachricht) an den „Zwilling“. Wenn innerhalb eines bestimmten Zeitraumes keine Antwort auf diese Anfrage kommt und immer noch zu wenige Computer im Knoten vorhanden sind, wird die Anfrage wiederholt.

Der Zwilling ist durch seine ID gekennzeichnet. Um die entsprechende ID zu erhalten, wird einfach das letzte signifikante Bit der eigenen Knoten-ID invertiert.

In dem KWJ-Paket sind die IPs und die anderen Daten über den lokalen Knoten enthalten. Dieses ist vorteilhaft, da nicht bekannt ist, ob der Ringmaster wechselt, bevor eine Antwort des Zwillings zurückkommt. Wegen dieser sicherheitsrelevanten Daten werden diese Pakete auch nicht normal geroutet. Sonst wäre es ein Bruch der Anonymitätsforderung (siehe III.2.1.1.2), da alle intermediären Knoten die entsprechenden Daten lesen könnten. Stattdessen werden die Pakete über den Vorgänger des Knotens geroutet. Ab einer Alphabetgröße des De-Bruijn-Graphen von vier hält dieser garantiert Verbindungen zu sowohl dem Anfrage-Knoten als auch zu dessen Zwilling. Hierdurch wird eine sehr kurze Verbindung zwischen den beiden Knoten gewählt und außerdem entsteht kein Sicherheitsproblem, da die entsprechenden IPs so oder so in diesem Knoten bekannt sind.

Der Zwilling kann entweder viele Mitglieder haben oder wenige. Wenn die Mitgliederzahl eines Zwillings eine bestimmte Grenze überschreitet, so wird deren Anzahl entsprechend verringert. Hierbei führen, nachdem sie eine entsprechende Nachricht vom Ringmaster erhalten haben und sich ordnungsgemäß aus dem Knoten abgemeldet haben, die

„überzähligen“ Mitglieder das Bootprotokoll erneut aus, indem sie eine „WTC“-Nachricht zu dem Anfrage-Knoten routen. Dabei wird die Verwürfelung der Startposition umgangen. Hierdurch wird aller Wahrscheinlichkeit nach das Problem des Anfrageknotens von zu wenigen Mitgliedern beseitigt. Durch die Benutzung des Bootprotokolls steht es allerdings nicht fest, dass genau der Anfrageknoten derjenige ist, in dem die „gesendeten“ Computer hinzugefügt werden. Hierdurch ist das System sicherer.

Wenn die Anzahl der Mitglieder des Zwillings relativ gering ist, also unterhalb der Grenze liegt, so gibt es wiederum zwei Fälle. Entweder der Zwilling hat einen höheren oder den gleichen Level wie der des Anfrage-Knotens.

Wenn der Level höher ist, wird die KWJ-Nachricht verworfen und ein Join auf dem entsprechend höheren Level eingeleitet. Hierdurch erhöht sich die Anzahl der Mitglieder im Knoten, wodurch bei der nächsten KWJ-Anfrage auf diese geantwortet werden kann. Zusätzlich nähert sich der Level des Zwillings auch dem des Anfrage-Knotens an. Sollte also der Mitgliedermangel im Anfrage-Knoten andauern, kann nach genügend oft wiederholten Joins des Zwillings ein Join zwischen dem Anfrage-Knoten und dem Zwilling durchgeführt werden.

Wenn der Level des Zwillings aber gleich dem des Anfrage-Knotens ist, wird eine Zusammenführung der Knoten eingeleitet. Hierzu merkt sich der Ringmaster des Zwillings die Daten des Anfrage-Knotens. Des Weiteren routet er zu diesem seine eigene IP und seinen Port, so dass eine direkte Verbindung zwischen den beiden Knoten aufgebaut werden kann. Dabei wird die Routingadresse neu berechnet. Die dazugehörende Nachricht wird mit KJO abgekürzt (KJO=„**K**notenmanagement-**J**oin-**O**k“-Paket). Wenn diese Nachricht verloren geht, muss sie durch ein KWJ erneut angeregt werden. Dass hierbei die Zieladresse neu berechnet wird und ein Routing stattfindet, sorgt dafür, dass nicht beliebige Computer ein Join mit einem bestimmten Knoten ausführen können. Wenn solch ein Computer eine KWJ Anfrage stellt, so erhält er nie die Antwort darauf, kann weder einen Join durchführen, noch hat er Informationen über den „Zwilling“ gewonnen.

Die direkte Verbindung ist notwendig, da bei einer Bekanntgabe des Joins die Nachbarn ihre Routingeinträge so verändern, dass es keine Routingverbindung zwischen den beiden Knoten mehr gibt. Deswegen muss eine direkte Verbindung zwischen den beiden Knoten bestehen, bevor der Join bekannt gegeben wird.

Wenn der Ringmaster des Anfrage-Knotens die KJO-Nachricht erhält, so baut er die Verbindung zu dem anderen Ringmaster auf. Schlägt dieses fehl, so wird das Protokoll zurückgesetzt.

Kommt eine Verbindung vom Anfrage-Knoten beim Ringmaster des Zwillings an, so ist dieses ein Signal für diesen den Join durchzuführen. Dementsprechend legt er hier den Zeitpunkt des Ereignisses fest und verteilt die Daten des Anfrage-Knotens unter allen Mitgliedern des Zwillingsknotens mit der Aufforderung einen Join „durchzuführen“. Die entsprechende Nachricht ist das „**K**notenmanagement-**D**o-**J**oin“-Paket (=KDJ). Gleichzeitig schickt er ein Paket mit den Daten des Zwillings über die neue Verbindung („**K**notenamanagement-**V**erteile-**J**oin“=KVJ).

Erhält der Ringmaster des Anfrage-Knotens das KVJ-Paket, errechnet er daraus eine KDJ-Nachricht und verteilt diese im Anfrage-Knoten. Wenn er aber kein solches Paket nach einer bestimmten Zeit erhalten hat, so schickt er eine „**K**notenmanagement-**R**equest v**J**“-Nachricht (=KRJ), die den Ringmaster des Zwillings dazu veranlasst das KVJ-Paket noch einmal zu schicken. Wenn trotzdem keine KVJ-Nachricht eintrifft, wird das Protokoll komplett zurückgesetzt, da anscheinend der Ringmaster des Zwillings ausgefallen ist. Dabei wird gehofft, dass der eigentliche Join noch nicht bekannt gegeben wurde.

Erhält ein Mitglied eines Knotens ein KDJ-Paket, so berichtigt es die Datenstrukturen entsprechend der im Paket enthaltenen Daten und benachrichtigt das Verbindungsmanagement-Protokoll über den Join. Geht das DJ-Paket aber verloren, erhält

das Mitglied dieses noch einmal zugesandt, was durch die Keep-Alive-Nachrichten ausgelöst wird (siehe III.4.1.2.2). Auch hier werden doppelte oder veraltete Nachrichten mittels des Zeitpunkts des Ereignisses erkannt.

Das Protokoll hat an einer Stelle einen Schwachpunkt. Wenn beispielsweise der Ringmaster des Anfrageknotens ausfällt, nachdem er eine Verbindung zum Ringmaster des Zwillings aufgebaut hat, ist ein nicht korrigierbarer Fehlerzustand erreicht. Eine Kontaktaufnahme zwischen den Knoten mittels Routing ist der Wahrscheinlichkeit nach nicht mehr möglich, da die Nachbarn den Join schon vollzogen haben. Es existiert aber auch im Anfrageknoten kein Wissen mehr über den Zwillingsknoten, weswegen es keine direkte Kontaktaufnahme geben kann. Allgemein entsteht dieser Zustand immer dann, wenn im Zwillingsknoten das DJ-Paket schon verschickt wurde (also die Zusammenführung bekannt gegeben wurde), aber dieses im Anfrageknoten nicht erfolgt. Das Erreichen solch eines Zustandes ist jedoch sehr unwahrscheinlich. Zusätzlich müsste die Komplexität des Protokolls stark erhöht werden, um solch einen Fehlerzustand unmöglich zu machen. Deswegen wurde diese Schwachstelle in dieser Arbeit nicht beseitigt.

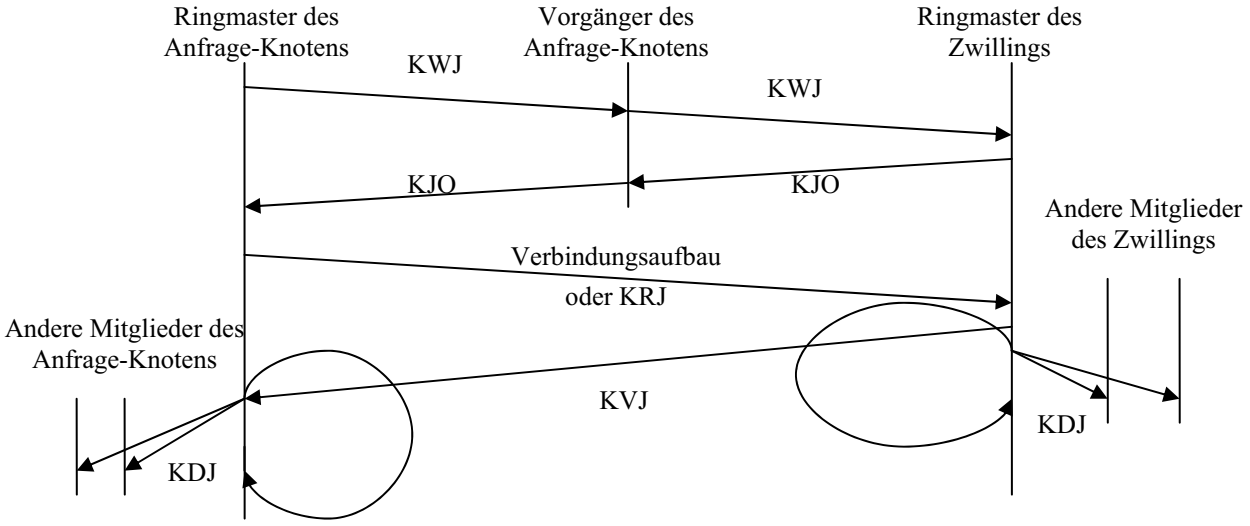

Abbildung 12: Schematischer Verlauf des Join-Protokolls, bei einer Knotenzusammenlegung.

III.4.1.3. Verbindungsmanagement-Protokoll

Das Verbindungsmanagement-Protokoll verwaltet die Verbindungen zwischen den Knoten. Hierbei gibt es K/2 bis 2*K ausgehende und K eingehende Kanten (siehe III.3.3). Beim Verbindungsaufbau werden genügend Informationen übermittelt, um eine Verbindung auf einer Verbindung im De-Bruijn-Graphen abbilden zu können.

Um diese Verbindungen erhalten zu können, muss mindestens einer der Partner die IPs des anderen Partners kennen. Hierbei sollte der entsprechende Partner aus Redundanzgründen nicht nur eine IP kennen, sondern die IPs aller Mitglieder des Partnerknotens besitzen. Dementsprechend müssen alle Änderungen in der Mitgliederliste eines lokalen Knotens an alle solche Partner mitgeteilt werden. Damit auch die Verbindungen erhalten werden, die denen im De-Bruijn-Graphen entsprechen, müssen auch Änderungen der ID mit zugehörendem Level mitverteilt werden. Bei jeder dieser Nachrichten wird die Repräsentation des entsprechenden Nachbarknotens geändert.

III.4.1.3.1. Verbindungsordnung

Bei dem unter III.3.1 beschriebenen Netz sind ausgehende Verbindungen maximal durch die letzten $\log_2(K)+1$ gültigen Bits der Knoten-ID des verlinkten Knotens gekennzeichnet. Dieses gilt immer dann, wenn der Level des verlinkten Knotens um eins höher ist als der des lokale. Wenn die Level gleich sind, so sind nur $\log_2(K)$ Bits notwendig, welche mit einem Null-Bit am Ende ergänzt werden. In diesem Fall wird dann der nächste Eintrag in der Liste der Verbindungen leer gelassen. Bei einem tieferen Level des verlinkten Knotens sind sogar nur $\log_2(K)-1$ Bits notwendig, weswegen zwei zusätzliche Nullbits hinten angefügt werden. Außerdem werden die nächsten vier Einträge übersprungen.

Bei einkommenden Verbindungen ist dieses etwas anders. Hier sind nicht die unteren gültigen Bits des Knotens, von dem die Verbindung kommt, das Ordnungskriterium, sondern die obersten Bits der ID des entsprechenden Knotens. Außerdem sind es hier immer die gleiche Anzahl an Bits, nämlich $\log_2(K)$ Bits.

Eine weitere Komplikation tritt ein, wenn in den IDs noch nicht genügend Bits für eine vollständige Ordnung bereit stehen. Dieses ist in den frühen Aufbaustadien des Netzes gegeben oder allgemein, wenn die Knoten des Netzes einen Level haben, der geringer ist als M (siehe III.3.3; $\log_2$(De-Bruijn-Basis)). In diesem Fall werden zusätzliche Bits gebraucht. Damit die Umordnungen bei Split und Join immer die gleichen sind, müssen diese Bits bei ankommenden Verbindungen unten an die ID angefügt werden. Hierbei hängt die Anzahl der nötigen zusätzlichen Bits von dem Level des Knotens ab, von dem die Verbindungen stammen. Bei ausgehenden Verbindungen müssen diese Bits aber oben zu der ID hinzugefügt werden. Da bei Levelunterschieden zwischen den benachbarten und dem lokalen Knoten Nullbits zu der ID hinzugefügt werden, hängt die Anzahl an zusätzlichen Bits von dem lokalen Level ab und nicht von dem des Knotens, zu dem die Verbindung geht.

III.4.1.3.2. Konsistenthaltung der Mitgliederlisten der Nachbarn

Zur Konsistenthaltung wird ähnlich wie beim Knotenmanagement-Protokoll Keep-Alive-Nachrichten (VKA=„**V**erbindungsmanagement-**K**eep-**A**live-**N**achrichten“) an alle Nachbarn gesandt. In solchem Paket ist sowohl der Level (inklusive des Synchronisationszeitpunktes siehe KMP (III.4.1.2)) als auch die aus dem Knotenmanagement bekannte Prüfsumme über die Mitglieder desjenigen Knotens, an den die Nachricht gesandt wird, enthalten.

Wenn beim Erhalten eines solchen Paketes der Level nicht übereinstimmt, so wird je nach Differenz entweder die letzte Split-Nachricht (VNS=„**V**erbindungsmanagement-**N**eigbour-**S**plitting“) oder die letzte Join-Nachricht (VNJ=„**V**erbindungsmanagement-) versendet. Dieses wird aber nur gemacht, wenn gleichzeitig der Synchronisationszeitpunkt des Levels nicht größer als der Zeitpunkt des letzten lokalen Ereignisses ist. Trifft dieses nicht zu, so hat der Nachbar ein Ereignis schon mitbekommen, bevor der Computer im dem lokalen Knoten eine entsprechende Nachricht direkt erhalten hat.

Wenn der Level, aber nicht die Prüfsumme, mit den lokalen Daten übereinstimmt, wird die gesamte Liste der Mitglieder in einer VWS-Nachricht („**V**erbindungsmanagement-**W**rong-check**S**um“=VWS) zurückgesandt, worauf die Repräsentation entsprechend korrigiert wird. Stimmen die Prüfsummen aber überein, so wird der Partner benachrichtigt, dass alles in Ordnung ist (VOK-Nachricht=„**V**erbindungsmanagement-alles-**OK**“-Nachricht).

Wenn die Antwort auf eine Keep-Alive-Nachricht eine bestimmte Zeitlang nicht erfolgt, so wird die entsprechende Verbindung geschlossen. Wenn die Forderung verletzt ist, dass mindestens eine Verbindung zu jedem Knoten besteht, wird eine neue Verbindung zu einem zufälligen Mitglied des Nachbarknotens aufgebaut.

III.4.1.3.3. Ebenhaltung des Netzes

Unter „Ebenhalten“ wird hier das Minimieren von Levelunterschieden verstanden. Um dieses zu erreichen, wird die levelbereinigte Anzahl der Computer der Nachbarknoten überprüft und nur Unterschiede von maximal einem Computer zugelassen. Wenn in einem Knoten festgestellt wird, dass ein höherer Unterschied besteht, so meldet sich der Ringmaster des entsprechenden Knotens mit einer gewissen Wahrscheinlichkeit ab und führt das Bootprotokoll unter Umgehung der Verwürfelung aus. Hierdurch werden solche Unterschiede verringert, ohne dass es einen Determinismus geben würde, der ausgenutzt werden kann, um in bösartiger Absicht durch das Netz zu wandern. Das Wandern ist zwar noch möglich, allerdings kann es passieren, dass man an einer ganz anderen Stelle landet, als man wollte.

Hier wird nur mit einer Wahrscheinlichkeit gearbeitet, um die Anzahl an Bewegungen von Computern zwischen Knoten zu verringern, da dieses jedes Mal einen relativ großen Aufwand bedeutet. Zur Berechnung der Wahrscheinlichkeit wird davon ausgegangen, dass jeder Nachbar des Knotens, der zu wenige Mitglieder hat, die gleiche Anzahl an Computern besitzt. Unter diesem Szenario soll sich in der ersten „Runde“ gleich die optimale Anzahl an Computern bewegen. Dementsprechend ist die Wahrscheinlichkeit einer Bewegung gleich der Größe des Unterschiedes durch die Anzahl der Nachbarknoten.

Um diese Vergleiche überhaupt sinnvoll durchführen zu können, muss, wie oben angesprochen wurde, aber vorher eine Levelbereinigung durchgeführt werden. Letztere ist bedeutsam zur „Ebenhaltung“ des Netzes.

In dieser Implementierung werden alle Knoten rechnerisch auf dem um eins verringerten Level des lokalen Knotens zusammengeführt. Hierbei wird der Aufwand umgangen, den Knoten mit dem minimalen Level der Nachbarschaft festzustellen. Dieses Minimum könnte nicht geringer als der angenommene Wert sein, da nur Levelunterschiede von einem Level in der Nachbarschaft zugelassen sind. Wenn das Minimum höher gewesen sein sollte, so wird die Levelkorrektur bei allen Nachbarn gleichmäßig angewandt, es entsteht also nach Überlegungen des Autors kein Schaden.

Um eine exakte Levelbereinigung durchzuführen, müsste dafür die Anzahl der Mitglieder von nicht benachbarten Knoten erfragt werden, was mit einem relativ hohen Aufwand verbunden ist. Dieses kann aber umgangen werden, indem man annimmt, solch ein Knoten habe die gleiche Anzahl Mitglieder wie der Knoten, den man kennt. Wäre der andere Knoten größer, sollte dieser ein Mitglied abgeben und nicht der jetzige. Deswegen ist die Verringerung der eigentlichen Differenz durch diese Annahme nicht schädlich. Der entsprechend größere Knoten führt die umgekehrte Berechnung durch und erhält deswegen einen zu großen Unterschied, der dafür sorgt, dass er mit einer theoretisch zu hohen Wahrscheinlichkeit ein Mitglied verliert. Hierdurch gleicht sich das System aus. Wenn der entsprechende Knoten nicht zur Nachbarschaft gehört, so ist die berechnete Wahrscheinlichkeit im schlimmsten Fall etwas zu gering. Hier steigt die Wahrscheinlichkeit, dass nicht im ersten Schritt die ideale Anzahl an Computern ihre Knotenzugehörigkeit ändern. Dieses ist kein großer Nachteil, da erwartet wird, dass die Zeit zur Sammlung der entsprechenden Größen nicht entscheidend kleiner ist als die bis zur nächsten Runde.

Ein sehr wichtiger Punkt ist, dass die Unterschiede in der Mitgliederanzahl sowohl zu den Nachfolgern als auch zu den Vorgängern gering gehalten werden. Zusätzlich soll aus Sicherheitsgründen die Entscheidung über die Mitglieder nur im lokalen Knoten getroffen werden. Dabei ist es auch in dieser Version noch möglich, Mitglieder aus einem bestimmten Knoten zu entfernen (durch falsche Größenangaben). Es ist aber nicht möglich, selber diese Mitglieder auszuwählen.

Das Mitglied, welches die Entscheidung trifft, dass aus dem lokalen Knoten ein Mitglied das Bootprotokoll erneut durchführen soll, (in diesem Fall der Ringmaster) muss dementsprechend mindestens über die aktuelle Anzahl der Mitglieder von allen Nachbarknoten informiert sein. Dementsprechend muss er zu allen Nachbarn Verbindungen

offen halten. Um dieses gewährleisten zu können, muss er entweder selber in der Lage sein, Verbindungen zu allen diesen Nachbarn zu öffnen, oder es muss eine Garantie geben, dass jeder Nachbarknoten mindestens eine Verbindung zum entsprechenden Ringmaster hält. Die letzte Variante ist relativ instabil und problematisch zu erhalten, wenn beispielsweise ein Ringmaster ausfällt. Aus diesem Grund wurde die erste Variante gewählt. Es gibt zwar den Nachteil, dass die Anonymität (siehe III.2.1.1.2) sich leicht verringert, aber dieses ist nur geringfügig. Ein zusätzlicher Vorteil ist es, dass dadurch die Anzahl der Mitglieder der Nachbarknoten nicht erfragt werden müssen, sondern schon lokal vorliegt.

III.4.1.4. Das Routingpotokoll

Alle hier verwendeten Routing-Systeme sind weder reihenfolgetreu, noch sind sie gegen den Verlust oder das mitschneiden von Paketen gesichert.

Es gibt in diesem Netz verschiedene Anforderungen für das Routing:

1. Zuerst gibt es Pakete, die beispielsweise als Anfragen zu einem bestimmten Knoten geroutet werden müssen. Hierbei ist es egal, welches der Mitglieder dieses Knotens die Anfrage bearbeitet (siehe III.4.1.4.1).
2. Bei manchen Paketen ist es nötig, dass auf sie geantwortet werden kann. Diese Antworten müssen zu einem ganz bestimmten Computer (dem ursprünglichen Versender des Paketes) (zurück-)geroutet werden (siehe III.4.1.4.1).
3. Durch die dritte Art des Routings wird eine bestimmte Art von Audits möglich. Hierbei muss man dauerhaft einen Computer in einem Knoten ansprechen können (siehe III.4.1.4.2).

III.4.1.4.1. Routing von Anfrage und Antwort

Die erste Aufgabe wird durch den De-Bruijn-Graphen festgelegt und realisiert (siehe III.3.1.2).

Ein wichtiger Punkt für das Netzwerk ist die Anonymität seiner Mitglieder. Deswegen wird im Allgemeinen keine Absenderadresse in den Routingpaketen enthalten sein. Die Anonymität des Absenders sollte aber auch gewahrt bleiben, wenn es eine Antwortmöglichkeit geben muss.

Eine Möglichkeit dieses zu realisieren wäre, dass bei jedem Routingvorgang einer Anfrage gespeichert wird, von wem diese kam. Eine Antwort würde dann genau die gleichen Computer durchlaufen wie die entsprechende Anfrage, bloß in umgekehrter Reihenfolge. Dieses wäre ein anonymes System, allerdings muss eine relativ große Menge an Statusinformationen für jeden Routingvorgang gespeichert werden. Des Weiteren wäre es sehr problematisch die Antwortmöglichkeit sicherzustellen, wenn einer oder mehrere der Computer ausfallen würden, die zwischen dem Sender und dem Empfänger liegen.

Eine zweite Möglichkeit ist es, eine eindeutige Absenderadresse anzugeben. Die Forderung nach Anonymität kann man in diesem Fall durch zwei Schritte erreichen. In dem ersten wird nur anonymisiert, welcher Computer innerhalb eines Knotens die Antwort erhalten soll. Hierzu wird für jede Anfrage mit Antwortmöglichkeit eine genügend lange (hier wurden 64Bit gewählt) Zufallszahl (ab jetzt Verbindung genannt) gewürfelt und sich für eine bestimmte Zeit gemerkt. Die Verbindung ist genügend lang, wenn es praktisch nicht vorkommt, dass zwei Mitglieder eines Knotens zu einem Zeitpunkt die gleiche Verbindung haben. Eine Antwort wird nun an den entsprechenden Knoten und diese Verbindung adressiert. Wenn die Antwort durch normales Routing diesen Knoten erreicht, so wird sie an alle Mitglieder des Knotens gesendet. Jedes Mitglied überprüft nun, ob die Verbindung in seiner Liste enthalten ist oder nicht. Wenn sie enthalten ist, so bearbeitet es die Nachricht, wenn sie nicht enthalten ist, so verwirft es diese.

Im zweiten Schritt wird die Anfrage nicht direkt an den Empfänger verschickt, sondern über einen oder mehrere andere zufällige Knoten. Diese legen ihrerseits eine Verbindung an und tragen diese im Paket ein, bevor sie es weiterleiten. Dabei wird zu dieser Verbindung auch gespeichert, für wen sie angelegt wurde, so dass Pakete, die an die Umleitung adressiert wurden, an den ursprünglichen Versender der Anfrage weitergeleitet werden können. Dabei ist für den Anfrager nicht bekannt, über welchen Computer des umleitenden Knoten die Anfrage läuft. Aus diesem Grund kann man keine zweite Anfrage über eine umgeleitete Verbindung schicken. Es können jedoch mehrere Antwort-Pakete über eine Umleitung geschickt werden. Dieses ist jedoch kein Nachteil, sondern eher ein Vorteil, da es hierdurch unwahrscheinlich ist, dass ein Computer einen kompletten Kommunikationsverlauf mitbekommt. In solch einem Fall wäre denkbar, dass der entsprechende Computer Informationen gewinnt, die eventuell einen Angriff ermöglichen würden. Dieses im Allgemeinen zu verhindern, ist Aufgabe der Anwendungsschicht.

Bei diesem System ist sowohl die Anonymität des Absenders als auch des Empfängers relativ hoch, obwohl der Absender eine Antwort erhalten kann. Um ein höheres Maß an (beweisbarer) Anonymität zu erreichen, könnte man das System noch durch Verschlüsselung der Pakete erweitern (siehe [52],[60]). Hierdurch würde jeder der Computer, welche die Anfrage umleiten, nur seinen Vorgänger und seinen Nachfolger kennen.

Ein weiterer Vorteil ist, dass relativ wenige Statusinformationen (je nach Anzahl der Umleitungen) benötigt werden. Auch treten kaum Probleme durch das Offline-Gehen von Computern auf, da die einzigen nötigen die umleitenden Computer sind (abgesehen von Empfänger und Versender).

Allgemein wird jedes Paket, welches gerouted werden soll, im ersten Hop nicht gezielt gerouted, sondern an einen zufälligen Nachbarn verschickt. Hierdurch wird die Vielzahl der möglichen Pfade ausgenutzt, ohne viele Pakete versenden zu müssen.

In den folgenden Texten wird ein Routing über einen zufälligen Knoten, der die Pakete weiterleitet, durch „umgeleitet“ „mit Umleitung“ und Ähnlichem gekennzeichnet. Bei einem Routing ohne Umleitung wird dieses mit Formen des Wortes „direkt“ benannt.

III.4.1.4.2. Computerpräzises Routing für Audits

Dieses System ist für eine bestimmte Art von Audits nicht geeignet. Bei dieser Art fragt man einen Computer mehrmals in Abständen nach einem bestimmten Datensatz und dessen Verlauf. Dabei müssen die Datensätze nachvollziehbar auseinander hervorgehen. Ist dieses nicht der Fall, so wurde Betrug begangen. Mit dem oben beschriebenen Modell des computerpräzisen Routings könnte man über die Verbindungen unterscheiden, von wem solch ein Audit gesandt wurde. Hierbei kann man zwar nicht die Absenderadresse benennen, aber man könnte nachverfolgen, wie die Antwort bei der letzen Anfrage gelautet hat, und dementsprechend antworten. Deswegen muss eine Adresse, welche für diese Art der Audits geeignet ist, für alle Anfrager gleich sein. Dieses kann aber nur gewährleistet sein, wenn mindestens die Computer in den Nachbarknoten die Adresse einem Computer zuordnen können und diese Zuordnung überprüfen können. Hierfür könnte man beim Einfügen eines Computers in einen Knoten eine statische Verbindung würfeln und diese den Nachbarknoten bekannt geben. Die entsprechenden Verbindungen nur im eigenen Knoten bekannt zu geben, nützt nur wenig, da die Knotenzusammensetzung relativ einfach durch ein Mitglied eines Knotens verändert werden kann (siehe III.4.1.1, IV.1.2.2). Solche Verbindungen sind jedoch erheblich weniger anonym als jene im vorherigen Abschnitt (III.4.1.4.1).

Diese Art des Routings wurde nicht implementiert, da die Sicherheitsanforderungen der Anwendung nicht hoch genug waren, um den Aufwand von Audits zu rechtfertigen (siehe III.4.2).

III.4.2. Das Abrechnungssystem

Bei dem hier verwirklichten Abrechnungssystem handelt es sich um eines mit relativ geringen Sicherheitsanforderungen, da so gut wie keine Kosten durch und während der Mitgliedschaft entstehen. Aus diesen Gründen wird hier auch ein einfaches und damit nicht hochsicheres System ohne Audits implementiert. Es wird an den entsprechenden Stellen jedoch eine zweite Lösung angeboten, welche ein erhöhtes Maß an Sicherheit beispielsweise durch die Möglichkeit von Audits bietet. Hierdurch entsteht aber ein erheblicher Mehraufwand, beispielsweise bei der dann benötigten Komplexität der Protokolle.

Das eigentliche System ist zweigeteilt. Auf der einen Seite agiert jedes Mitglied, welches online ist, im Auftrag des Netzes, indem es Inhalte wie beispielsweise Kontostände speichert. Auf der anderen Seite vertritt jedes dieser Mitglieder seine eigenen Interessen gegenüber dem Netz.

Ein sehr wichtiger Punkt bei der Speicherung der Kontostände ist, dass diese auf mehreren voneinander unabhängigen Knoten aufbewahrt werden. Je höher die Anzahl an Knoten, desto sicherer ist das System in diesem Aspekt. Im Folgenden werden die entsprechenden Knoten für ein bestimmtes Konto AH-X genannt, was für Account Holding steht. Das X entspricht dabei einer Zahl und gibt an, der wievielte Knoten bei der Berechnung der AHs dies ist. Die ID des AH-0 ist gleich der Kontonummer. Die ID des AH-X wird aus der ID des vorherigen AHs berechnet, also desjenigen mit der Ordnungsnummer X-1 (AH(X)= f(AH(X-1))). Details über die Berechnung werden an entsprechenden Stellen der weiteren Beschreibung dargelegt.

Im diesem Kapitel werden folgende Punkte genauer diskutiert:

1. Netznahe Aufgaben,
2. Abbuchung des Nutzens,
3. Beantragung eines neuen Kontos,
4. Online-Anmeldung eines Kontos,
5. Überprüfung des Online-Status und
6. Abgleich der Kontostände zwischen AHs .

Netznahe Aufgaben sind solche, welche durch das zugrunde liegende Netz hervorgerufen werden. Im Fall des hiesigen Netzes ergeben sich zwei Aufgaben. Erstens muss die Datenbank bei Splits und Joins (siehe III.4.1) entsprechend der Veränderung der Knoten-ID angepasst werden und zweitens muss die Übertragung der Datenbank an neue Mitglieder geleistet werden.

In dem Abschnitt „Abbuchung des Nutzens“ wird das Protokoll vorgestellt, mit dem „Nutzen“ von einem Konto abgebucht werden kann, wenn der Kontobesitzer über ein anderes Mitglied der Gemeinde surfen will.

Der Abschnitt „Beantragung eines neuen Kontos“ erläutert das Protokoll zum Anlegen eines neuen Kontos. Dieses wird immer gebraucht, wenn ein neues Mitglied in die Gemeinde kommt. Was passiert, wenn ein Mitglied dem Netzwerk beitritt, welches schon ein Konto besitzt, findet man unter „Online-Anmeldung eines Kontos“.

Damit man sich nicht online melden kann, ohne es wirklich zu sein, wird immer geprüft, ob diejenigen, die online gemeldet sind, dieses auch sind. Wie dieses abläuft, wird unter „Überprüfung des Online-Status“ beschrieben.

Damit auch alle AHs einen wenigstens annähernd gleichen Kontostand führen, wird unter „Abgleich der Kontostände zwischen AHs “ beschrieben, wie die Kontostände auf den verschiedenen AHs abgeglichen werden.

In Klammern wird in den folgenden Abschnitten der Name einer Nachricht angegeben, der bei der Implementierung benutzt wurde.

III.4.2.1. Sicherheitsprinzipien für Audits

Um Audits durchführen zu können, ist es auf der Protokollebene erst einmal wichtig, Identitäten eindeutig identifizieren zu können („authentication“). Hinzu kommt, dass auch jede Nachricht eindeutig einer Identität zugeordnet werden kann. Die Nachrichten dürfen dabei nicht (mit „zumutbarem“ Aufwand) fälschbar sein. Des Weiteren muss das Verhalten jeder Identität nachvollziehbar gemacht werden. Auf Nachfrage muss also jeder eine Liste seiner letzten Operationen überreichen, in der auch die entsprechenden Partner enthalten sind.

Des Weiteren sollte die Aussage einer Nachricht nur durch den Inhalt dieser Nachricht bestimmt werden. Es sollte außerdem explizit dargelegt werden, unter welchen Bedingungen auf eine Nachricht wie reagiert wird, so dass bei einer Überprüfung des Protokolls entschieden werden kann, ob diese Bedingungen akzeptabel sind oder nicht. Zusätzlich sollte die Identität eines Auftraggebers explizit in einer Nachricht genannt werden, wenn es bei der Bedeutung der Nachricht auf diese Identität ankommt (frei aus [47]).

III.4.2.2. Netznahe Aufgaben

Netznahe Aufgaben sind solche, welche durch das zugrunde liegende Netz hervorgerufen werden. Im Fall des hiesigen Netzes ergeben sich zwei Aufgaben: Erstens muss die Datenbank bei Splits und Joins (siehe III.4.1) entsprechend der Veränderung der Knoten-ID angepasst und zweitens muss die Übertragung der Datenbank an neue Mitglieder durchgeführt werden.

Bei einem Split müssen Konten gelöscht werden. Dabei werden jeweils jene gelöscht, bei denen das Präfix der Kontonummern nicht mehr der Knoten-ID entspricht. Hierdurch werden in den neuen Knoten jeweils unterschiedliche Konten gelöscht und global gesehen gehen keine Informationen verloren.

Bei Joins muss dieser Vorgang wieder rückgängig gemacht werden. Es müssen jeweils die Konten, welche im jeweils anderen Knoten gespeichert wurden, auf die eigenen Mitglieder kopiert werden.

Eine sehr ähnliche Aufgabe entsteht auch bei der Eingliederung von neuen Mitgliedern in einen Knoten. Hierbei muss die gesamte Datenbank eines Knotens auf das neue Mitglied übertragen werden.

Hierbei ist zu beachten, dass die Datenbank eine relativ große Menge an Daten enthält. Deswegen ist es wenig sinnvoll, die gesamte Datenbank als ein Paket zu übertragen, was einen relativ langen Zeitraum beanspruchen würde, in der das darunter liegende Netz nicht agieren könnte. Die natürliche Teilung der Datenbank ist diejenige, welche sie in ihre einzelnen Konten teilt.

Hierfür werden die Kontonummern in den entsprechenden Netznachrichten verpackt und in einem zweiten Schritt erst die Kontodaten einzeln angefordert und geschickt.

Bei dem hinausgezögerten Kopieren der Datenbank entsteht die Gefahr, dass diejenigen Mitglieder, die die entsprechenden Kontodaten gespeichert haben, den Knoten verlassen haben, bevor die Daten kopiert sind. Hierdurch würden die entsprechenden Daten verloren gehen. Dieses ist jedoch auf Grund der Anzahl der Mitglieder sowie einer relativ geringen Fluktuationsrate, welche durch das Abrechnungssystem erzeugt wird (siehe III.1.5), relativ unwahrscheinlich und wird deswegen ohne Gegenmaßnahmen in Kauf genommen.

Bei dem Eingliedern neuer Mitglieder kann man durch die Einzelübertragung der Kontodaten außerdem die Asymmetrie der Übertragungsraten beispielsweise von DSL ausnutzen. Hierzu werden die Daten von verschiedenen Mitgliedern angefragt. Dadurch wird die häufig geringere Upstream-Kapazität mehrerer Mitglieder genutzt, um die höhere Downstream-Kapazität des entsprechenden Mitgliedes nutzen zu können.

Damit das gleiche Protokoll auch bei Join-Operationen genutzt werden kann, werden die Mitglieder, von denen die Daten erfragt werden, zufällig ausgesucht. Kann ein Mitglied

erfragte Daten nicht liefern, ignoriert es die Anfrage einfach. Hat ein Computer die Daten eines Kontos nach einer bestimmten Zeit noch nicht erhalten, so fragt er ein weiteres zufälliges Mitglied nach diesen.

III.4.2.3. Abbuchung des Nutzens

Wenn ein Mitglied der Gemeinde (hier Surfer genannt) über ein anderes (hier Anfrager genannt) surfen will, so gibt er diesen Wunsch, zusammen mit seiner Kontonummer, diesem bekannt (DSU-Nachricht= „**D**atenbank-want-to-**SU**rf“-Nachricht). Der Anfrager fragt mit einer Umleitung nach dem Kontostand bei jedem der AHs an und schickt eine für jeden AH individuelle Zufallszahl mit (DGK-Nachricht=„**D**atenbank-**G**et-**K**onten“-Nachricht). Diese AHs antworten (direkt) mit dem Kontostand und dem öffentlichen Schlüssel des Surfers (DSK-Nachricht=„**D**atenbank-**S**end-**K**onto“-Nachricht). Wenn das entsprechende Konto auf einem AH-X nicht existiert, so sendet dieser einen Kontostand von Null zurück.

Der Anfrager sammelt und sortiert diese Antworten. Zusätzlich überprüft er anhand der Zufallszahlen, ob auch jeder der AHs selber geantwortet hat oder ob einer versucht hat, für einen anderen Knoten zu antworten. Um die Wahrscheinlichkeit weiter zu steigern, dass eine Nachricht auch wirklich vom entsprechenden AH stammt, kann man auf die Antwort wieder (mit Umleitung) antworten. Hier erhält man durch das zufällige Routen nur eine weitere direkte Antwort, wenn die Nachricht auch wirklich aus dem Zielknoten stammt. Damit aber solche direkten von umgeleiteten unterschieden werden können, müssen die umgeleiteten gekennzeichnet werden. Sonst könnte jeder, der eine Anfrage abfängt, die Antwort über den eigentlichen Knoten umgeleitet zurücksenden. Somit wäre keine Überprüfung möglich.

Bei der Sortierung ist das erste Kriterium das Engagement und das zweite Kriterium die verbleibende Surfzeit. Als nächstes wird ein bestimmter Prozentsatz der Antworten (beispielsweise 33%) entfernt, welche am weitesten unten einsortiert wurden. Danach werden die Werte des zu unterst einsortieren Eintrages als reale Werte genommen.

Der Prozentsatz sollte hierbei (weit) unter 50% liegen. Hierbei wird ausgenutzt, dass die Wahrscheinlichkeit, dass ein Angriff das Ziel einer Kontostandserniedrigung hatte, sehr viel kleiner ist und mit weniger Aufwand betrieben wird, als das Ziel einer Kontostandserhöhung (siehe II.4.3.4). Deswegen wird hier ein höherer Aufwand zur Erhöhung des Kontostandes gegen eine vereinfachte Kontostandserniedrigung eingetauscht. Dieses ist im Gegensatz zur normalen Mehrheitsbildung (der Wert mit den meisten Zusendungen wird genommen) oder der Wahrscheinlichkeitsmaximierung (siehe III.2.1.1.1) zu sehen.

Wenn der Kontostand ausreichend hoch ist, wird der Surfer durch eine DGC-Nachricht (=„**D**atenbank-**G**et-**C**heck“) angewiesen, eine „Abbuchungsanweisung“ (in einer DSC-Nachricht=„**D**atenbank-**S**end-**C**heck“ verpackt) an den Anfrager zu schicken. In dieser sind die Uhrzeit und der abzubuchende Betrag enthalten. Er ist außerdem mit dem geheimen Schlüssel des Surfers verschlüsselt. Mit dem öffentlichen Schlüssel, den der Anfrager von den AHs erhalten hat, überprüft dieser den Inhalt der Abbuchungsanweisung. Wenn hier alles in Ordnung ist, wird die Anweisung über eine Umleitung an jeden der AHs weitergeleitet (DEC-Nachricht=„**D**atenbank-**E**inreichen-**C**heck“).

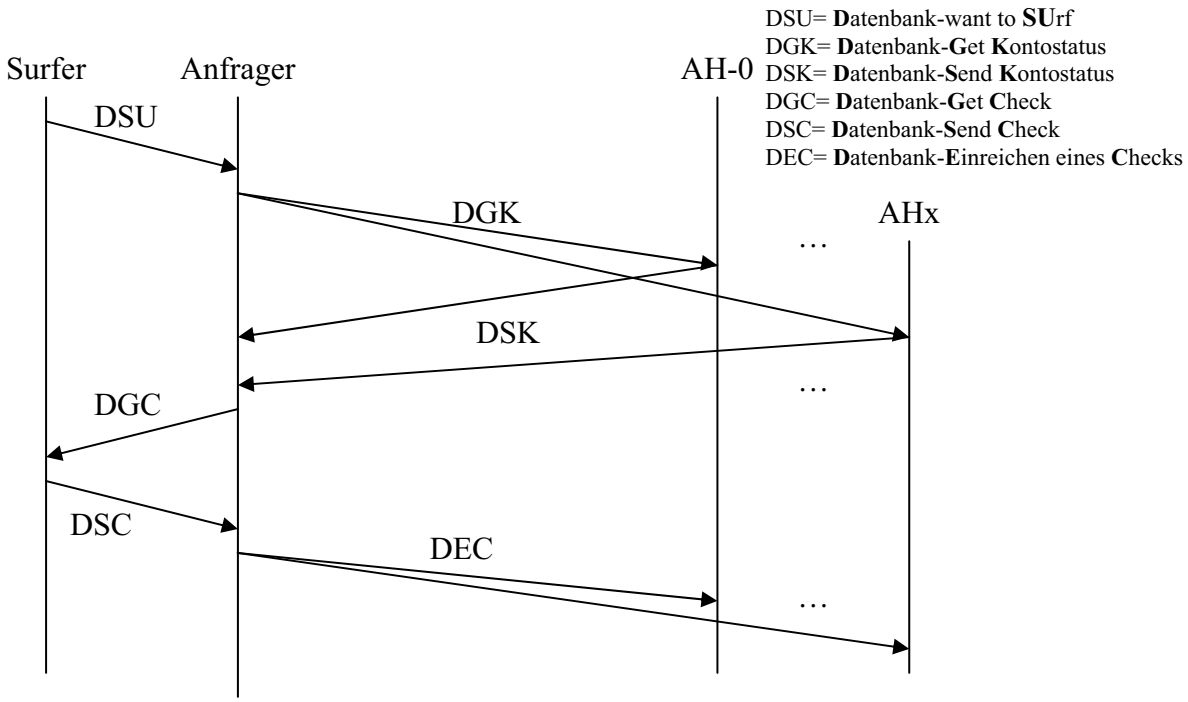

Abbildung 13: Schematischer Verlauf des einfachen Protokolls zur Abbuchung von Onlinezeit

In einer einfachen Version empfangen die AHs diese Abbuchung, buchen den Betrag ab und löschen die Anweisung. In einer aufwändigeren und sichereren Version werden Audits eingeführt. Deswegen werden die Checks eine bestimmte Zeit (die Aufbewahrungszeit) gespeichert. Dabei werden sie zuerst noch nicht verrechnet. Dieses passiert erst nach Ablauf der Aufbewahrungszeit. (Verlauf der Audits unter III.4.2.7.1).

III.4.2.4. Beantragung eines neuen Kontos

Das Neuanlegen eines Kontos hat eine relativ hohe Sicherheitsrelevanz, da bei diesem Prozess diejenigen Knoten bestimmt werden, auf dem die Kontodaten gespeichert werden (AHs).

Dementsprechend kann man versuchen, an diesem Punkt die Wahl der AHs so zu beeinflussen, dass sie zu den eigenen Absichten am Besten passen und man mit wenig Aufwand die Kontrolle über die Daten erlangen kann.

Auf Grund der hohen Sicherheitsrelevanz werden zwei Versionen beschrieben. Eine, die auch sehr hohen Sicherheitsanforderungen genügt (siehe III.4.2.4.2), und eine zweite, die den benötigten Aufwand erheblich reduziert, aber immer noch ein gutes Maß an Sicherheit bietet (siehe III.4.2.4.1).

Allgemein wird ein Konto entweder aufgrund von Beweisen gelöscht oder, weil mehr als 3 Monate kein Engagement mehr verzeichnet wurde.

III.4.2.4.1. Einfache Version der Neuanlage eines Kontos

Hier erfolgt die Kontovergabe in fünf einfachen Schritten, welche nicht die Sicherheit der Vergabe bedingen. Diese wird durch einen anderen Mechanismus, der auch beschrieben wird, hergestellt.

In einem ersten Schritt generiert der Anfrager einen geheimen und einen öffentlichen Schlüssel (beispielsweise RSA-Schlüssel). Den öffentlichen Schlüssel verschickt er mit einer Anfrage nach einem neuen Konto über eine Umleitung an einen zufälligen Knoten (DRA-

Nachricht=„**D**atenbank-**R**equest-**A**ccount“). Dieser nimmt seine Knoten-ID als Anfang der Kontonummer und würfelt zufällig den Rest der Kontonummer. Daraufhin gibt er direkt auch den entsprechenden AHs Bescheid, dass sie das Konto anlegen (DAN-Nachricht=„**D**atenbank-new-**A**ccount-**N**otification“). Wenn diese das getan und das mittels einer DNO-Nachricht (=„**D**atenbank-**N**otification-**O**k“) mitgeteilt haben, legt er selbst das Konto an und schickt dem Anfrager die Kontonummer (DAN-Nachricht=„**D**atenbank-**A**ccount-**N**umber“). Hierbei muss erwähnt werden, dass es hier bei der Nennung von AH jeweils nur um einen Repräsentanten des AHs handelt. Nur wenn eine Aktion des ganzen Knotens nötig ist, verteilt dieser die entsprechende Aufforderung an alle anderen Mitglieder des Knotens.

Dieses begründet aber auch eine Schwachstelle. Wenn der Computer ausfällt, der die Anfrage im AH-0 entgegengenommen hat, und nachdem er dieses an die anderen AHs weitergegeben hat, aber bevor er das Protokoll beenden kann, so entsteht ein „verwaistes“ Konto auf den anderen AHs. Zu diesem gibt es keinen AH-0, es wird aber auch von niemandem benutzt. Dem Anfrager entsteht kein Schaden, da er einfach (nach einem Timeout) ein neues Konto beantragen kann. Dieser Fall ist relativ selten. Außerdem hat er kaum und nur leichte Konsequenzen. Solche Konten werden nach 3 Monaten von selbst durch das allgemeine Löschverfahren gelöscht. Innerhalb dieser Zeit wird eine kleine Menge an Bandbreite für das Synchronisieren der AHs gebraucht, was jedoch fehlschlägt. Aus diesen Gründen wurde dieser Sonderfall nicht weiter bearbeitet.

Solange die Berechnung der AHs nicht speziellen Anforderungen genügt, ist dieses Protokoll, wie oben erwähnt, unsicher. Diese Unsicherheit entsteht dadurch, dass eine zufällige Wahl von Zahlen der beliebigen entspricht. In diesem Fall kann also der Anfrager sich gezielt jemanden aussuchen, der die Kontonummer vergibt. Mögliche Wahlen für Betrüger wären hier sie selbst oder ein Mitbetrüger. Der Rest der Kontonummer wird dann auch entsprechend der Wünsche gewählt. Deswegen ist die gesamte Kontonummer frei wählbar.

Durch eine bekannte Berechnung der AHs ist es im Allgemeinen also möglich auch die AHs frei zu wählen. Die Freiheit der Wahl hängt dabei von den Eigenschaften der Berechnungsfunktion ab. Je freier diese Wahl ist, desto unsicherer ist dadurch das System.

Um dem System trotzdem Sicherheit zu verleihen, werden gewisse Forderungen an die Berechnungsfunktion gestellt. Die Wichtigste ist, dass jedem Knoten seine Partner AHs fest zugewiesen werden. Hierdurch ist nur der AH-0 frei wählbar. Ein Angreifer muss also in eine bestimmte Anzahl an Knoten gezielt gelangen. Dieses ist im hier benutzten Netz relativ schwierig, weswegen das System wiederum relativ sicher ist.

Um dieses zu gewährleisten, muss die Berechnung der AH-X aus der Kontonummer und damit aus der Knoten-ID unabhängig vom Level (siehe III.3.2) des Knotens sein. Dieses bedeutet wiederum, dass bei der entsprechenden Berechnung ein Bit nur niederwertigere Bits beeinflussen darf.

Eine weitere daraus resultierende Forderung an die Berechnung ist, dass die Entfernung zwischen den verschiedenen AHs auf dem De-Bruijn-Graphen möglichst groß sein sollte. Wie unter III.2.1.1.2 erstmals beschrieben wurde, gibt es die Möglichkeit, durch das Netz gezielt zu wandern. Hierbei hängt der Aufwand hierfür stark von der De-Bruijn-Entfernung der beiden Knoten zueinander ab. Da man einen bestimmten Prozentsatz der AHs kontrollieren muss (siehe III.4.2.3), um ein Konto manipulieren zu können, steigt der Aufwand hierzu sowohl mit der Entfernung der AHs zueinander als auch mit deren Anzahl.

Der Autor dieser Arbeit hat keine Funktion gefunden, die die zweite Eigenschaft nachweislich hat. Allerdings erfüllen Funktionen wie sie bei Pseudozufallsgeneratoren oder Hashes, die in der Kryptographie verwendet werden, diese Eigenschaft wenigstens statistisch. Deswegen wird hier solch eine Funktion als Basis verwendet.

Um die erste Eigenschaft auch noch zu erreichen, wird eine mehrstufige Berechnung gewählt. Hierzu werden die Y höchstwertigen Bits benutzt, um das Y-ste Bit der Nachfolge-ID festzulegen. Da diese Funktionen nicht Bit-, sondern Byteweise arbeiten, werden die benötigten, aber nicht definierten Bits durch eine feste Bitfolge (beispielsweise mit Nullen) aufgefüllt. Dementsprechend wird das erste Bit der ID des AH-1 aus dem ersten Bit der ID des AH-0 berechnet. Das zweite Bit wird aus den ersten zwei Bits berechnet und so weiter. Hierdurch ist die Levelunabhängigkeit des Verfahrens gewährleistet (siehe oben).

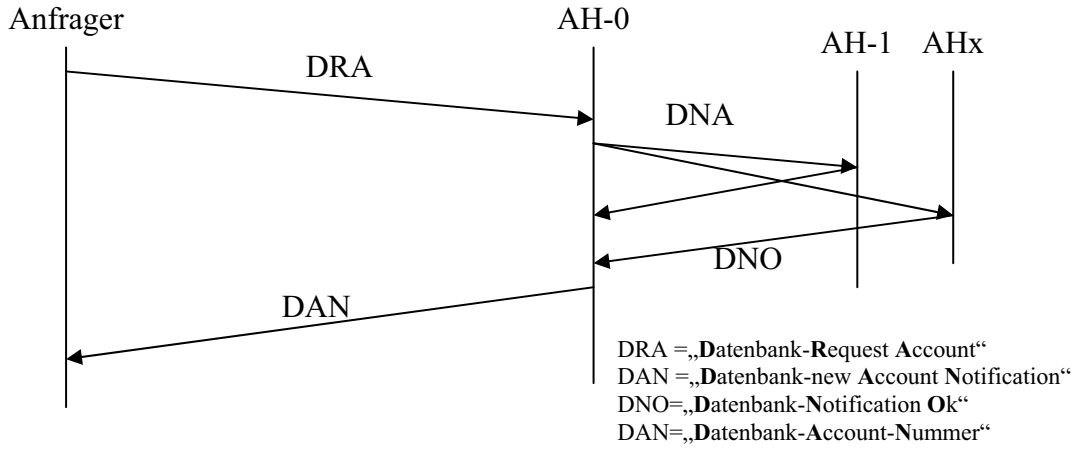

Abbildung 14: Schematischer Verlauf des einfachen Protokolls zum Neuanlegen eines Kontos

III.4.2.4.2. Hochsicherheits-Version der Neuanlage eines Kontos

In dieser Version geht es darum, die Sicherheit der Kontovergabe noch weiter zu erhöhen. Hierzu wird die Kontonummer nicht von einem, sondern von mehreren Teilnehmern generiert. Die Teilnehmer sind diejenigen, welche durch die Funktion berechnet werden, die unter III.4.2.4.1 beschrieben wurde. Ergänzt werden sie von dem Anfrager selber. Mit Hilfe von Coin-Flipping-Algorithmen (siehe [53]) kann man die Vergabe der Kontonummer so gestalten, dass sie immer noch zufällig ist, selbst wenn ein sehr hoher Prozentsatz der Teilnehmer einer kriminellen Gruppe (siehe II.4.3.2) angehört. Ein möglicher Coin-Flipping-Algorithmus wäre, dass jeder der Beteiligten eine Zufallszahl generiert. Hierauf senden sie den kryptographischen Einweghash dieser Zahl an die anderen Beteiligten. Damit kennt keiner die Zahl eines anderen, aber jeder kann sicherstellen, dass die Zahl zu diesem Zeitpunkt feststeht und nicht geändert werden kann. In einem zweiten Schritt sendet jeder seine Zahl an die anderen und überprüft, ob die Zahlen, die er erhält, auch wirklich den entsprechenden Hashwert haben. Stimmen diese Werte überein, ist die endgültige Zufallszahl das XOR aller dieser Zahlen.

Liegt die Kontonummer fest, werden auf direktem Wege die entsprechenden AHs kontaktiert. Diese überprüfen jeder für sich, ob alle Beteiligten an der Zufallszahlgenerierung dieser zustimmen, also ob alles korrekt abgelaufen ist. Hierbei wird aber eine Umleitung benutzt. Ist dem so, wird das Konto angelegt. Geht irgendwann etwas schief, wird kein Konto angelegt. Zusätzlich wird kein Knoten zugelassen, der sowohl mit bei der Generierung der Zufallszahl mitgeholfen hat als auch ein AH für das Konto ist. So etwas führt zu einer Neugenerierung der Kontonummer.

Die Funktion, welche die AHs berechnet, sollte in diesem Fall nicht levelunabhängig (siehe III.4.2.4.1) sein. Jene zur Berechnung der Partner bei der Zufallszahlengenerierung sollte aber levelabhängig sein. Durch die Levelunabhängigkeit der Funktion zur Berechnung der AHs gibt es nur geringe Überschneidungen bei den AHs verschiedener Konten in einem Knoten, wodurch es wiederum sehr viel schwerer wird, mehr als ein Konto zu manipulieren.

Um Audits zu ermöglichen, muss in jedem der Knoten, die an der Generierung der Zufallszahl beteiligt waren, gespeichert werden, dass sie es tatsächlich waren. Außerdem muss in den Kontodaten auch enthalten sein, welche Knoten beim Anlegen des Kontos „geholfen" haben. Die Audits sind nötig, da sonst eine kriminelle Gruppe eine bestimmte Anzahl an willkürlichen Knoten „übernehmen" könnte und dann eine Kontonummer suchen könnte, deren AHs genau diese Knoten wären. Wenn die Zahl gefunden wäre, könnten sie dieses einfach anlegen. Durch die Audits müssen aber zusätzlich die festen Partner, die beim Generieren der Kontonummer beteiligt waren, mit übernommen werden, was relativ schwer ist (siehe III.4.2.4.1).

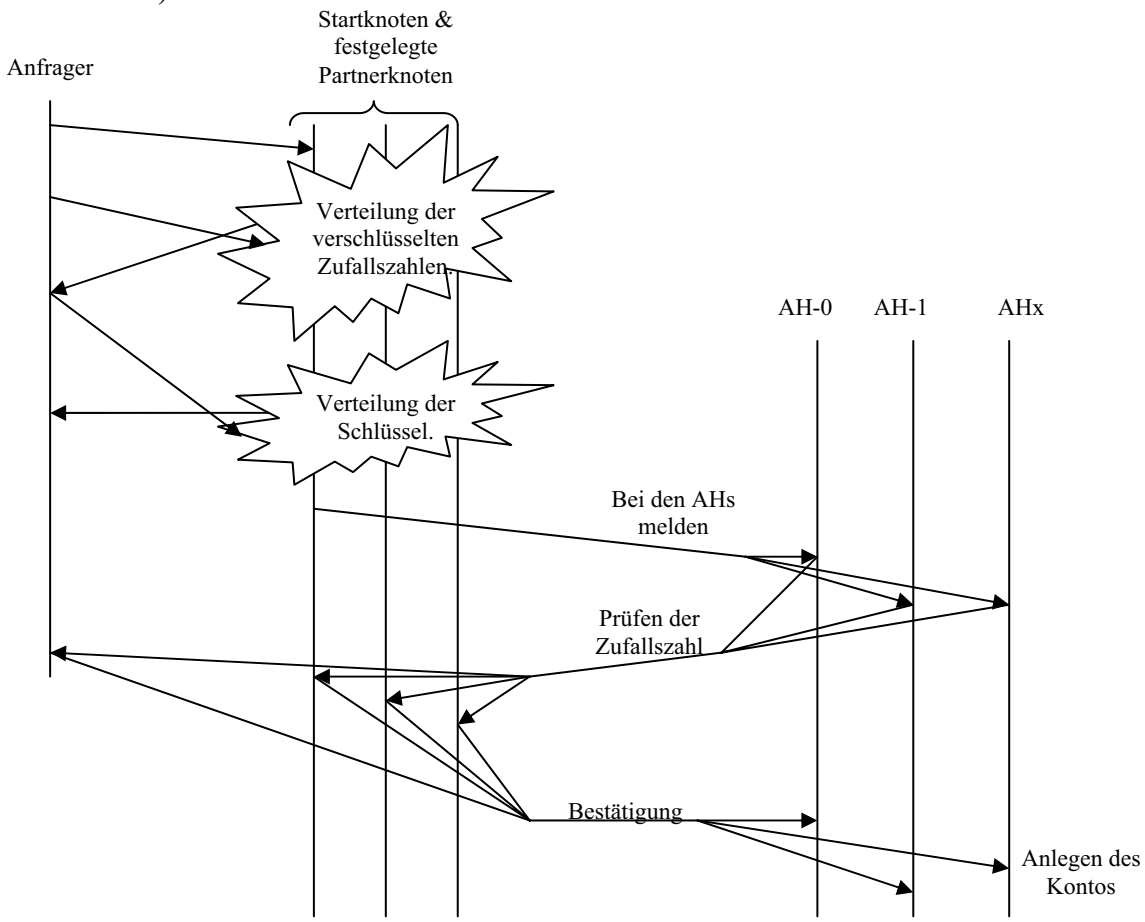

Abbildung 15: Schema des Verlaufes einer Implementierung des hochsicheren Protokolls zum Anlegen neuer Konten

III.4.2.5. Online-Anmeldung eines Kontos

Ein Mitglied der Gemeinde hat ein Interesse daran, dass es auch wirklich online gemeldet ist, solange es online ist. Um dieses sicherzustellen, schickt deswegen jedes Mitglied mittels Umleitungen in regelmäßigen Abständen Anfragen an jeden seiner AHs (DCE-Nachricht=„**D**atenbank-**C**heck-ob-**E**ingetragen"). Wenn diese Nachricht ankommt, überprüft das Mitglied des AHs, welches die Anfrage erhält (Empfänger genannt), ob das

entsprechende Konto online gemeldet ist. Wenn dem so ist, wird dieses auf direktem Wege dem Anfrager mitgeteilt (DON-Nachricht=„**D**atenbank-bist-**ON**line-gemeldet“). Wenn das nicht zutrifft, wird auch dieses dem Anfrager auf direktem Wege mitgeteilt, aber es wird der öffentliche Schlüssel des Empfängers mitgeschickt (DEB-Nachricht=„**D**atenbank-nicht-online-**E**xecute-**B**in-online“). Dadurch kann der Anfrager seine IP so verschlüsseln, dass nur der Empfänger dieses entschlüsseln kann. Dieses tut er dann auch und schickt seine IP über eine Umleitung an den Empfänger (DBO-Nachricht=„**D**atenbank-**B**in-**O**nline“). Dieser entschlüsselt die IP und meldet allen Mitgliedern des Knotens, dass der Anfrager online ist. Die Angabe der IP an dieser Stelle erklärt sich aus ihrer Benutzung beim Prüfen, ob jemand online ist (siehe III.4.2.6).

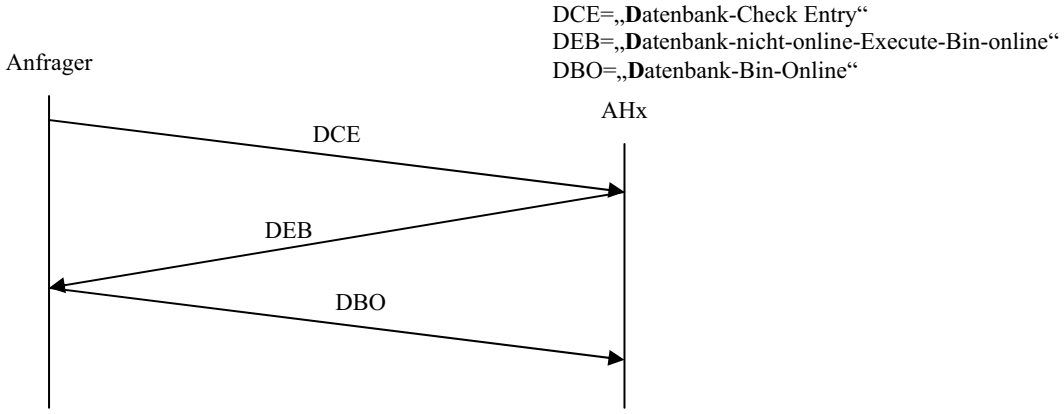

Abbildung 16: Schematischer Verlauf des einfachen Protokolls zum Onlinemelden

III.4.2.5.1. Sichere Version der Anmeldung

In einer zweiten sichereren Version wird noch überprüft, dass eine IP nicht unter zwei Kontonummern im Peer-to-Peer-Netz angemeldet ist, bevor der Anfrager als online gilt. Hierdurch kann es noch weiter erschwert werden, dass ein Computer mehrere Konten auflädt.

Hierzu muss eine zweite Datenbank vorhanden sein, in der gespeichert ist, welche IPs von Mitgliedern benutzt werden. Dabei wird aber nur gespeichert, ob eine IP benutzt wird, nicht aber von wem.

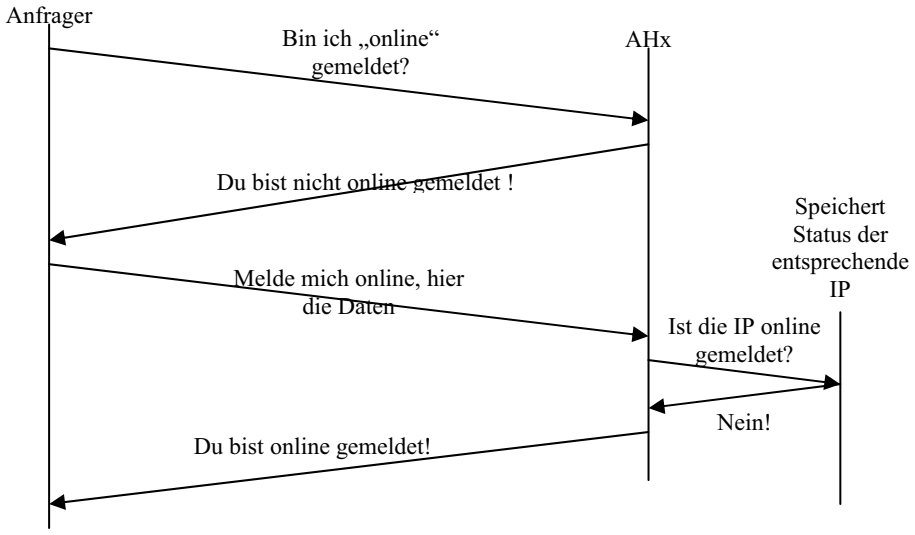

Abbildung 17: Schematischer Verlauf des aufwändigeren, aber sichereren Version des Online-Anmeldens

Will sich ein Mitglied als online melden, so wird überprüft, ob die IP schon benutzt wird. Wenn das nicht zutrifft, wird die IP als benutzt eingetragen. Trifft dieses aber zu, wird die Onlinemeldung nicht akzeptiert. Solch ein Eintrag sollte über einen Time-out-Mechanismus verfügen, durch den eine IP ausgetragen wird, falls eine gewisse Zeit keine Nachricht eintrifft, dass sie weiter benutzt wird. Entsprechende Nachrichten werden von den AHs versandt, wenn sie überprüft haben, dass jemand noch online ist(siehe III.4.2.6.1).

III.4.2.6. Überprüfung des Online-Status

Damit auch wirklich nur die echte Onlinezeit von jemand (im Folgenden Kunde genannt) abgerechnet wird, müssen die AHs überprüfen, ob dieser auch wirklich online ist. Hierzu muss sich der Kunde regelmäßig bei jedem AH melden. Um aber einigermaßen sicher zu sein, dass der Kunde sich persönlich meldet, sollte seine Nachricht mit dem geheimen Schlüssel signiert sein. Um sicherzustellen, dass diese Nachrichten nicht vorbereitet werden können, sollte der AH dem Kunden eine Zufallszahl senden, die dieser verschlüsselt zurückschicken muss. Dabei sollte keine Information enthalten sein, mit deren Hilfe man auf die ID des Kunden schließen kann. Dieses gilt vor allem für den Fall, dass jemand mehrere Konten gleichzeitig aufladen will. Erhält er eine Aufforderung sich zurückzumelden, soll er bestenfalls raten können, welches seiner Konten gemeint ist. Deswegen wird auch die Anweisung per TCP/IP an den Kunden geschickt. Bei den zur Verfügung stehenden anderen Adressierungsmethoden könnte sonst über die Adresse eine Unterscheidbarkeit erreicht werden (siehe III.4.1.4). Bei der Verschickung des Aufrufes über TCP/IP kann man jedoch diese Unterscheidbarkeit vermeiden. Hierzu schickt nicht ein Computer des entsprechenden AHs die Nachricht per TCP/IP, sondern lässt dieses von einem zufälligen Stellvertreter tun. Da keine Informationen über die Kontonummer oder ähnliches im Paket enthalten sind, werden so gut wie keine für Kriminelle nützliche Informationen bekannt gegeben. Aus solch einer Nachricht kann man ausschließlich ersehen, dass es ein Mitglied der Gemeinde mit der entsprechenden IP gibt. Dieses lässt sich nach der Meinung des Autors aber so gut wie nicht ausnützen. Ein zweiter wichtiger Vorteil des Sendens durch einen Stellvertreter ist, dass die Anonymität der Mitglieder des AHs gewahrt bleibt.

Dementsprechend läuft das Protokoll wie folgt ab: In regelmäßigen Abständen sendet ein Mitglied jedes der verschiedenen AHs eine Nachricht an einen zufälligen Knoten (DFC-Nachricht=„**D**atenbank-**F**orward-**C**heck-if-online“), in der eine Antwortmöglichkeit, eine Zufallszahl und die IP des Kunden enthalten ist. Zur Wahrung der Ununterscheidbarkeit wird als Antwortmöglichkeit nur die Verbindungsnummer (siehe III.4.1.4) und die Ordnungsnummer des AHs angegeben, nicht aber dessen Knoten-ID.

Erreicht das Paket den zufälligen Knoten, baut der entsprechende Computer eine TCP/IP-Verbindung zur angegebenen IP auf und verschickt darüber das Paket (DCO-Paket=„**D**atenbank-**C**heck-if-**O**nline“). Der Kunde entnimmt dem Paket die Daten, verschlüsselt sie mit seinem geheimen Schlüssel, vervollständigt die Zieladresse und routed die Informationen über eine Umleitung an den AH zurück (DAO-Nachricht=„**D**atenbank-**A**nswer-**O**nline“). Kommt dieses Paket an, gilt der Kunde als zurückgemeldet.

Wenn aber keine Rückmeldung nach einem bestimmten Zeitpunkt erfolgt ist, wird erneut eine Aufforderung zur Rückmeldung verschickt. Dieses wiederholt sich einige Male in relativ kurzen Zeitabständen. Wenn am Ende trotzdem noch keine Rückmeldung erfolgte, so wird angenommen, dass der Kunde offline gegangen ist.

Die Bestimmung, welches Mitglied für die Prüfung eines Kontos zuständig ist, ob der Besitzer noch online ist, findet in mehreren Schritten statt. Wenn ein Konto neu erstellt wird, nehmen alle Mitglieder eines Knotens an, sie wären für dieses zuständig und würfeln jeder eine zufällige Priorität für das entsprechende Konto. Desgleichen nimmt ein Mitglied an, welches neu zu einem Knoten kommt, es sei für alle Konten zuständig und würfelt gleichfalls

Prioritäten. Wenn jemand der Meinung ist, er wäre für ein Konto zuständig, wartet er eine zufällige Zeit. Ist diese abgelaufen, führt er die Prüfung des Onlinestatus durch und verteilt das Ergebnis, die Kontodaten und die eigene Priorität im Knoten. Hierbei übernehmen die anderen Mitglieder des Knotens einfach den Kontostand desjenigen, der zuständig ist.

Wenn ein Mitglied solch eine Nachricht für ein Konto erhält, für das er denkt, dass er zuständig ist, vergleicht er seine eigene Priorität mit der aus der Nachricht. Ist seine Priorität kleiner, so ist er von da ab nicht mehr für das Konto zuständig, sind die Prioritäten gleich, würfelt er eine neue Priorität.

Wenn eine bestimmte Zeit keine solche Nachricht gekommen ist, beispielsweise weil das zuständige Mitglied den Knoten verlassen hat, fühlen sich die anderen Mitglieder wieder zuständig.

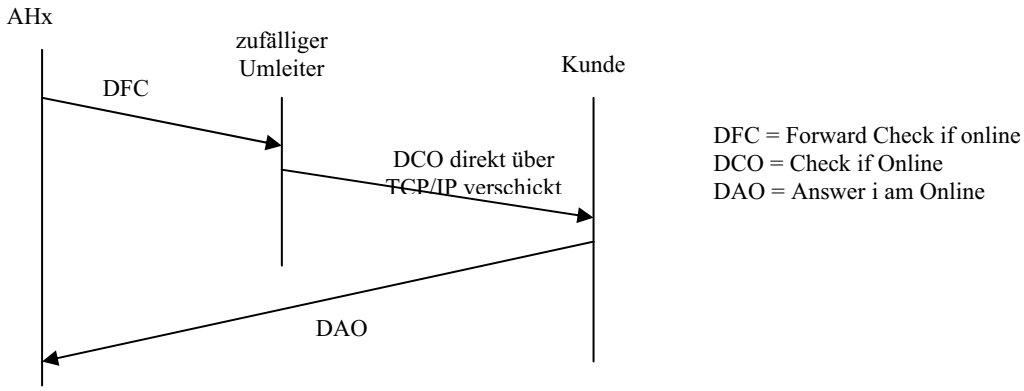

Abbildung 18: Schematischer Verlauf des einfachen Protokolls zur Überprüfung des Online-Status

III.4.2.6.1. Hochsicherheitsversion mit Audits

Um den Vorgang der Überprüfung durch Audits nachvollziehbar zu machen, ist es nötig, dass es mindestens einen unabhängigen Zeugen gibt. Dieser Zeuge wird nicht zufällig, sondern pseudozufällig mittels eines Pseudozufallsgenerators gewählt. Dabei wird die Ausgangszahl des Generators für Audits zugänglich gemacht, genauso wie die Folge der Zufallszahlen und damit die ID der Zeugen. Zusätzlich wird noch der Zeitpunkt des Verschickens des Aufrufes gespeichert. Ein entsprechender Eintrag wird eine bestimmte Zeit aufbewahrt (siehe III.4.2.3). Jedes Konto hat dabei seine eigene Ausgangszahl für den Generator.

Die Aufforderung sich zurückzumelden wird jetzt (mit Umleitung) an den Zeugen geschickt. Dieser wählt zufällig einen anderen Knoten (den Vermittler) aus, der eine TCP/IP-Verbindung zum Kunden erstellt und für eine bestimmte Zeit Pakete zwischen dem Zeugen und dem Kunden vermittelt. Dieser weitere Beteiligte ist notwendig, damit die IP des Zeugen dem Kunden nicht bekannt ist, wodurch der Zeuge angreifbar werden würde.

In dem Aufruf muss in dieser Version des Protokolls der öffentliche Schlüssel des Kunden enthalten sein. Nachdem die Verbindung zwischen Zeuge und Kunde besteht, schickt ersterer eine Zufallszahl. Der Kunde muss die Zufallszahl verschlüsselt zurücksenden. Tut er dieses nicht oder schickt er eine falsche Zahl zurück, wird der Vorgang vom Zeugen abgebrochen. Geschieht dieses jedoch korrekt, wird die eigentliche Aufforderung zur Rückmeldung an den Kunden geschickt. Hierdurch kann der Zeuge bestätigen, dass jemand mit der entsprechenden IP (die er nicht mitspeichert) zu dem entsprechenden Zeitpunkt online war und den entsprechenden geheimen Schlüssel besaß. Dementsprechend werden im Knoten der öffentliche Schlüssel und der Zeitpunkt der Zeugenschaft gespeichert. Die IP wird nicht

mitgespeichert bzw. bekannt gegeben, damit kein Zusammenhang zwischen IP und Kontonummer hergestellt werden kann. Derjenige, der ein Kontoeintrag überprüft kennt ja die Kontonummer des Kontos. Der Zeuge selber kann diesen Zusammenhang auch nicht herstellen, da er zwar die IP kennt, aber er keinen Hinweis auf die Kontonummer hat. Er könnte höchstens wahllos öffentliche Schlüssel anfragen und mit dem vergleichen, der in der Nachricht enthalten war.

Hierdurch kann jeder, der es möchte und solange die entsprechenden Daten gespeichert werden, überprüfen, ob wirklich eine Anfrage abgesendet wurde und ob diese auch an jemanden ging, der im Besitz des geheimen Schlüssels war. Wird bei solch einem Vorgang eine Inkonsistenz entdeckt, so wird das Konto von dem, der den falschen Kontostand herausgegeben hat, gelöscht.

Aus diesem Grund kann auch nicht einfach der Kontostand desjenigen übernommen werden, der zuständig ist, sondern es werden nur Änderungen übernommen, wenn diese nachgeprüft wurden.

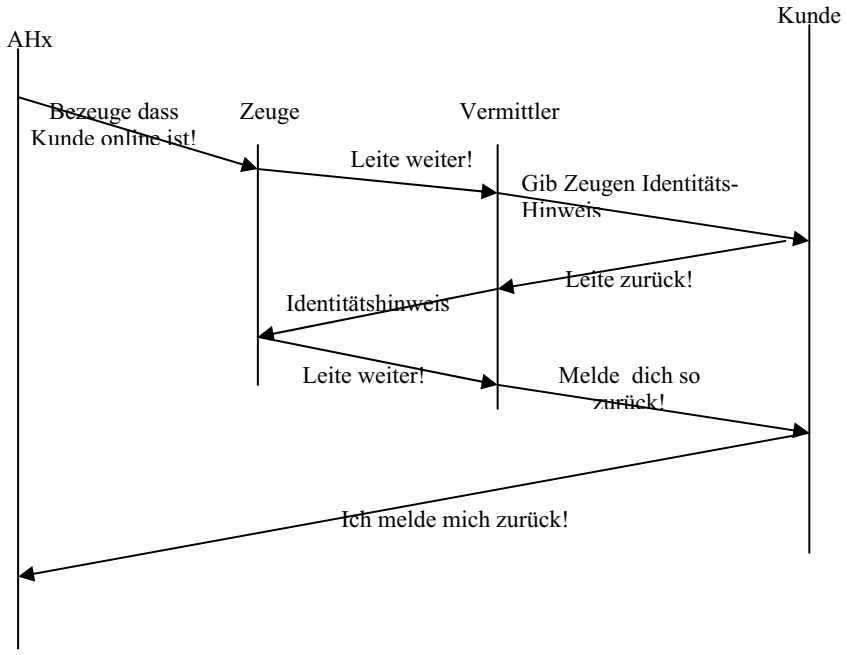

Abbildung 19: Schematischer Verlauf des Protokolls zur Überprüfung des Online-Status mit der Möglichkeit für Audits

III.4.2.7. Abgleich der Kontostände zwischen AHs

Eine weitere Aufgabe ist es, die Kontostände zwischen den AHs abzugleichen. Hierzu verschickt das Mitglied eines AHs (der Anfrager), welches für ein Konto zuständig ist (siehe III.4.2.6), eine Anfrage auf direktem Wege an die anderen AHs. In dieser Anfrage ist auch der öffentliche Schlüssel dieses Mitgliedes enthalten. Wird so eine Nachricht erhalten, werden die entsprechenden Kontodaten mit dem mitgelieferten Schlüssel verschlüsselt und an den Anfrager zurückgeschickt, wenn dieser in einem AH sitzt.

Der Anfrager sammelt die Daten der anderen AHs. Wenn er alle Datensätze hat, sortiert er diese, löscht die untersten und aktualisiert seinen eigenen Datensatz nach dem noch verbleibenden untersten (gleiche Prinzip wie bei III.4.2.3). Dieses wird auch an die anderen Mitglieder des Knotens weitergegeben.

III.4.2.7.1. Hochsicherheits-Version mit Audits

In dieser Version werden die Daten genauso gesammelt wie in der einfachen Version. Die einzige Änderung ist, dass der Herausgeber ausdrücklich genannt wird und dieser mit seinem privaten Schlüssel unterschreibt. Eine zweite Änderung ist, dass die Daten verglichen und bei Unterschieden die entsprechenden „Beweise“ geprüft werden.

Hierzu müssen alle Abbuchungen in dem Kontostand vorhanden sein, welche älter als eine gewisse Zeit sind. Diese Zeit ist eine Karenzzeit, welche Paketverluste, Versandzeiten u.Ä. berücksichtigen soll.

Wenn dem nicht so ist, wird (mit Hilfe von Umleitungen) die entsprechende Abbuchung mit dem Kontostand an die AHs desjenigen geschickt, der den falschen Kontostand herausgegeben hat. Hierdurch ist der Betrug „bewiesen“ (sehr wahrscheinlich) und deswegen wird dessen Konto gelöscht. Wenn eine Inkonsistenz innerhalb der Karenzzeit passiert, wird dem entsprechenden Knoten nur die Abbuchungsaufforderung geschickt, so dass er sie von da ab berücksichtigen kann.

Um die Daten des Online-Status zu kontrollieren, wird zuerst die Sequenz der Zeugen überprüft. Dann werden jene Einträge der Kontodaten durch eine Befragung der Zeugen überprüft, bei denen man sich uneinig ist, ob der Kontoinhaber online oder offline war. Gibt ein Zeuge auf Anfrage (über eine Umleitung) die Antwort (direkt geroutet), dass er den entsprechenden Vorgang nicht bezeugen kann, so wird dieses wieder (mit Umleitung) eingereicht und das Konto dessen, der einen nicht bezeugten Eintrag hatte, gelöscht. Dieses ermöglicht allerdings auf einfache Art und Weise, jemandem das Konto löschen zu lassen, wenn man als Zeuge fungiert. Allerdings ist es relativ schwer, gezielt Zeuge bei einer Onlineprüfung von jemanden bestimmten zu werden.

Wenn es bezeugte Onlinezeiten gibt, die im eigenen Knoten nicht vorhanden sind, so werden diese übernommen. Entsprechende Einträge zu fälschen ist genauso schwer, wie das Löschen, welches im vorherigen Paragraphen beschrieben wurde.

Des Weiteren wird noch geprüft, ob die Daten richtig konsolidiert wurden. Hierzu werden die entsprechenden Kontostände aufbewahrt und mit den aktuellen verglichen. Hierbei müssen sich die Summen (vom Engagement oder dem verbrauchten Nutzen) genau soweit geändert haben, wie die vorher vorliegenden Beweise dieses veranlasst haben. Ist dem nicht so, so werden der alte und der neue Kontostand wiederum eingereicht und das Konto gelöscht.

Kapitel IV

Parameteranpassung und Evaluation

IV.1. Parameteranpassung

IV.1.1. Berechnungen der Ausfallssicherheit

Ein Knoten besteht mindestens aus einer bestimmten Anzahl Computer, die alle die gleichen Informationen tragen. Solange ein Computer noch online ist, ist noch keine Information verloren gegangen und ein Knoten ist noch voll funktionstüchtig. Aus demselben Grund sind auch die Verbindungen zu Nachbarknoten stabil. Eine Reihe von Protokollen (siehe III.4.1) sorgen dafür, dass diese minimale Anzahl Computer nicht unterschritten wird. Dafür brauchen sie aber eine gewisse Zeit (kurz die Reparaturzeit). Dementsprechend ist die Wahrscheinlichkeit, dass ein Knoten ausfällt und die Informationen verloren gehen, durch die Wahrscheinlichkeit bestimmt, dass nach Eintritt eines solchen Falles alle verbleibenden Computer in einem Knoten innerhalb der Reparaturzeit ausfallen. Die Ausfallwahrscheinlichkeit eines Knoten ist also kleiner als

$$p(\text{Ausfall})^{\text{minimale Anzahl der Computer} - 1}$$

p(Ausfall) = Ausfallswahrscheinlichkeit eines Computers innerhalb der Reparaturzeit.

Wenn die Maßnahmen des Abrechnungssystems (siehe III.1.5) wirken, sollte ein Computer durchschnittlich 16h pro Tag an einem Stück online sein. Eine weitere Annahme sei, dass die Protokolle innerhalb von fünf Minuten (abgekürzt fmin) einen stabilen Zustand wieder herstellen können. Diese Zahl ist innerhalb der Computerwelt eine relativ lange Zeit, aber sie muss auch Timeouts, den Aufbau von Verbindungen und eventuell recht komplexe Nachrichtenvorgänge beinhalten. Zusätzlich soll dieses ja auch eine obere Grenze sein, die eingehalten wird. Das würde heißen, dass p(Ausfall)=1(Ausfall)/192fmin ist. Wenn also mehr als fünf Computer in einem Knoten sind, ist die Wahrscheinlichkeit (pro fmin), dass ein Knoten ausfällt, ungefähr gleich $3.832*10^{-12}$. Dementsprechend liegt der Erwartungswert, dass ein Knoten ausfällt, bei ca. 2482 Jahre, wenn es ständig 1000 Knoten gibt. Bei nur mehr als vier Computern würde unter den gleichen Umständen dieser Wert bei 12,9 Jahre liegen.

Hierbei ist nur normales Verhalten berücksichtigt. Bei Angriffen im Allgemeinen kann man auf diese Art und Weise keine Aussagen treffen, da es sehr stark auf die Art des Angriffes und andere nur im Einzelfall bestimmbare Werte ankommt.

IV.1.2. Rechnung zur der Betrugssicherheit

Bei der Berechnung der Sicherheit gibt es zwei Aspekte, die beachtet werden müssen. Dieses ist einmal die Sicherheit des Routings und zweitens die Sicherheit der abgefragten Daten. Die Berechnung kann immer nur modellhaft erfolgen, da es jeweils auf eine Vielzahl an Unbekannten ankommt, die nur im Einzelfall zuverlässig zu ermitteln wären. Bei den folgenden Betrachtungen wird jeweils davon ausgegangen, dass es nicht oder nur mit relativ hohem Aufwand möglich ist, Mitglied in einem bestimmten Knoten zu werden, in dem noch kein Mitglied der eigenen Gruppe (II.4.3.2) sich befindet.

IV.1.2.1. Wahrscheinlichkeit eines erfolgreichen Routingangriffes

Bei Angriffen auf das Routing gibt es zwei verschiedene Szenarien, die verschiedene Aussichten auf Erfolg haben. Ein allgemeiner Vorteil eines Angriffes auf das Routing ist, dass relativ schwer nachzuverfolgen ist, wer den Angriff durchgeführt hat, da jeder in der Kette potentiell verantwortlich sein könnte.

Im ersteren Fall kennt man denjenigen, von dem aus die Anfrage gestartet wird. Dieses ist das am meisten Erfolg versprechende Szenario. Hier kennt man jeden Knoten, der auf dem Weg vom Anfrager zu dem Angefragten passiert wird. Man muss zwar auf jeden der möglichen Wege präsent sein, aber es ist unwichtig, an welcher Stelle der Kette von Routingvorgängen man eingreift. Die Wahrscheinlichkeit, dass man in einen der entsprechenden Pfade eingreifen kann, ist proportional zu seiner Länge. Dass dieser Angriff auf das Routing erfolgreich ist, ist gleich der Wahrscheinlichkeit, mit der jemand gezielt in alle Verbindungen zwischen einem Anfrager und dem Angefragten gelangt. Die Wahrscheinlichkeit, dass jemand gezielt auf einen Pfad gelangt, entspricht dem Produkt aus derjenigen, in einen bestimmten Knoten zu gelangen, und der Länge des entsprechenden Pfades. Diese ist wiederum durch die Gesamtzahl der Knoten gegeben.

$$P(\text{Angriff erfolgreich})=(R*1/(\#K))^{\#P*\#A}$$

R=Routinglänge
#K=Gesamtzahl der Knoten im Netz
#P=Anzahl an parallelen Anfragen pro Angefragtem
#A= Anzahl der Angefragten

Im zweiten Szenario kennt man den Anfrager nicht, sondern nur den Angefragten. Ziel ist es dabei, die Meisten der an diesen gerichteten Nachrichten abzufangen. Dazu muss man auch die Mehrheit (wenn nicht alle) der Pfade besetzen, die zu einem bestimmten Knoten führen. Hierbei hat man aber nur sehr bedingt eine Wahlmöglichkeit, wo man das Routing angreift, da die Anzahl an Knoten, die besetzt werden müssen, exponentiell mit der Entfernung vom Zielknoten steigt. Wenn man die direkten Vorgängerknoten eines Knotens besetzt, so genügt dieses. Wenn man aber einen dieser Knoten nicht erreicht, sondern den Vorgänger des Vorgängers, so muss man auch die meisten der Vorgänger des Vorgängers besetzten. Man muss also nicht mehr einen, sondern viele Knoten übernehmen.

Deswegen bedeutet es einen viel größeren Aufwand, das Routing in solch einem Fall anzugreifen, als direkt die Kontodaten zu verändern. Der Autor dieser Arbeit ist deswegen der Meinung, dass dieses Szenario vernachlässigt werden kann, solange das erste Szenario beachtet wird.

IV.1.2.2. Wahrscheinlichkeit einer erfolgreichen Kontoverfälschung

Da ein bösartiges Mitglied eines Knotens relativ einfach die Anzahl von Elementen mit entsprechenden Absichten in dem Knoten erhöhen kann, wird im Folgenden davon ausgegangen, dass ein Knoten bösartig ist, sobald ein einziges Mitglied diese Eigenschaft hat.

Dieser Null-Eins-Effekt kommt durch zwei Punkte zustande: Der erste ist die fehlende Anonymität, der zweite ist, dass die Redundanz innerhalb eines Knotens zur Sicherung gegen Ausfälle gebraucht wird. Hierdurch muss ein Knoten neue Mitglieder aufnehmen, sobald er zu wenige hat. Diesen Zustand kann man beispielsweise mittels DoS-Angriffen von außen erreichen. Der Angreifer könnte aber auch einfach so lange warten, bis auf „natürliche“ Art und Weise ein Mangel an Mitgliedern zu verzeichnen ist und so die Chance deutlich erhöhen, dass ein Computer, der das Bootprotokoll ausführt, in den entsprechenden Knoten gelangt. Deswegen ist es auch nicht notwendig ein Mehrheitsentscheid beim Hinzufügen neuer Mitglieder durchzuführen.

Das Null-Eins-Verhalten kann aber toleriert werden, da die Betrugssicherheit so oder so durch verschiedene einander nicht bekannte Knoten hergestellt wird.

Aus diesen Überlegungen heraus ist es sinnvoll, nur die Anzahl der voneinander getrennten Knoten zur Berechnung der Betrugssicherheit heranzuziehen. Dabei ist zu beachten, dass eine Mindestgröße des Netzes von Nöten ist, so dass diese Knoten nicht die IPs voneinander kennen.

Unter diesen Vorraussetzungen ist die Wahrscheinlichkeit, dass ein Angriff erfolgreich ist, gleich der, dass man in eine Mehrheit der Knoten, die ein Konto speichern, ein Mitglied einschleusen kann. Als grobe Vereinfachung kann man annehmen, dass der Knoten, in den man durch das Bootprotokoll gelangt, ein zufälliger ist. Dementsprechend ist die Wahrscheinlichkeit, in einen bestimmten zu gelangen, gleich eins durch die Gesamtzahl der Knoten.

$$P(\text{Angriff erfolgreich})=(1/(\#K))^{\#A}$$

#K=Gesamtzahl der Knoten im Netz #A= Anzahl der Angefragten

IV.1.2.3. Wahrscheinlichkeit korrekter Daten.

Zusätzlich zur Wahrscheinlichkeit, dass ein Angriff gelingt, ist aber auch wichtig, wie wahrscheinlich es ist, dass man von korrekten Daten ausgeht. Dieses wird wiederum davon bestimmt, wie wahrscheinlich ein Angriff auf die gegebene Verbindung ist und von der Wahrscheinlichkeit, dass dieser gelingt. Hierbei wird davon ausgegangen, dass der am meisten Erfolg versprechende Angriff durchgeführt wird.

$$((1-p(\text{Angriff}))+P(\text{Angriff})*(1-\text{MAX}(p(\text{Angriff erfolgreich})))$$

Im Folgenden wird davon ausgegangen, dass 95% der Gemeindemitglieder rechtschaffend sind und dementsprechend 5% der Menschen betrügen wollen. Die Betrüger versuchen jeweils ihre eigenen Kontodaten zu manipulieren, so dass auch die Wahrscheinlichkeit eines Angriffes auf ein Konto bei 5% liegt. Des Weiteren wird davon ausgegangen, dass es 1000 Knoten gibt und immer zwei verschiedene Knoten auf zwei verschiedenen Wegen angefragt werden können. Dabei würde die Wahrscheinlichkeit, dass man von den korrekten Daten ausgeht, schon bei 99.999995% liegen. Hierbei wurde mit einem Angriff auf die Konten gerechnet. Bei einem Angriff auf das Routing liegt die Wahrscheinlichkeit sogar bei 99,9999999%. Aus diesen Berechnungen schließt der Autor, dass die beschriebenen Sicherheitsmaßnahmen genügend groß sind.

IV.2. Evaluation

Zur Evaluation der Protokolle wurden diese in zwei lauffähigen Programmen implementiert. Eines läuft als „Demon“-Prozess im Hintergrund und stellt die Funktionalität bereit. Zur Benutzerinteraktion gibt es ein zweites Programm, welches Parameter in Pakete umwandelt und diese per TCP/IP an den „Dämon“ schickt. Hierbei sind neben Debugging-Funktionalität nur das Beenden des Dämons und das Abbuchen von Nutzen vorgesehen.

Der Dämon benutzt nur einen Thread und beinhaltet hauptsächlich Elemente, die auf einkommende Pakete reagieren. Zusätzlich enthält er zeitgesteuerte Elemente, welche Nachrichtenketten / Protokollabläufe starten.

IV.2.1. Zeigen der Angemessenheit

Um die Angemessenheit der Protokolle zu zeigen, wurden mehrere Versuche durchgeführt, in denen unterschiedliche Aspekte des Systems beleuchtet werden sollten. Hierzu wurde eine bestimmte Anzahl an Dämonprozessen in zwei unterschiedlichen Umgebungen gestartet. Die eine besteht aus einem Server mit einem Dual-Opteron, bei dem das Loopback-Interface als Netzwerk benutzt wurde, und die andere aus zwei PCs, wobei sowohl das Loopback-Interface dieser als auch ein 10-Mbit-Ethernet-Interface gebraucht wurden, welches die beiden PCs verband.

Bei beiden Versuchsgruppen wurden die Parameter so gewählt, dass die Netze deutlich höher als im erwarteten Realfall belastet waren. Nur so kann ausgetestet werden, dass unter realen Bedingungen mit hoher Wahrscheinlichkeit keine Probleme zu erwarten sind.

Es wurde eine De-Bruijn-Basis von vier und eine Knotengröße von sieben bis 18 Mitgliedern benutzt. Die einzelnen Instanzen beendeten sich mit einer Wahrscheinlichkeit von $3*10^{-4}$ pro Sekunde ohne Vorwarnung. Dieser Wert ist sehr hoch gewählt und stellt damit auch eine sehr große Belastung für das Netz dar. Auf Grund der Anreize wird in einem realen Netz nur eine Ausfallsrate von ca. $1{,}7*10^{-5}$ erwartet (siehe IV.2.2.2 oder anders hergeleitet: ein Ausfall in 16 Stunden). Im Versuch wird also eine ca. 17-fach erhöhte Ausfallsrate benutzt. Wenn dabei keine größeren Probleme auftauchen, so sind auch unter realen Bedingungen keine zu erwarten.

In den Tabellen ist der Level (siehe III.3.1 mit der Änderung aus III.3.3) durch „Lv“ abgekürzt.

Kenndaten der Versuche, welche keine Fehler darstellen, werden in dem Anhang C zusammen mit Fehlern dargelegt. Dabei ist mit „Bewegung“ gemeint, dass eine Instanz erneut das Bootprotokoll ausführt, weil ein Nachbarknoten weniger Mitglieder hat als der lokale (siehe III.4.1.3.3). Wenn durch solch eine „Bewegung“ die Minimalgröße des lokalen Knotens verletzt werden würde, wird stattdessen ein Join ausgelöst. Dieses wird in der Tabelle unter „Unebenheit ➔WTJ“ geführt.

IV.2.1.1. Versuche auf dem Opteron

Versuch	Ausfalls-verzögerung	#Instanzen: Maximum	#Instanzen: Minimum	Dauer in h:min	fehlerhafte Einwahl	Mobbing	keine Synchro-nisation in %	verwaiste Konten	verlorene Routing Pakete in %	Sonstiges
Op1	1600	2173 Lv7	103 Lv3	5:49	3	1	0,029	1	0,014	
Op2	1600	2203 Lv7	129 Lv3&4	6:24	0	1	0,023	3	0,064	
Op3	1600	2182 Lv7	110 Lv3	5:05	0	3	0,025	3	0,062	
Op4	0	1624 Lv7	99 Lv3	5:49	6	9	0,053	2	0,053	*[1]
Op5*[2]	0	879 Lv6	118 Lv3&4	5:48	88*[3]	0	0,030	4	0,026	

Tabelle 4: Kennzahlen bei Belastungstestläufen des Netzwerkes auf einem Opteron

Erläuterungen und Wertungen werden im folgenden Text gegeben, eine ausführlichere Tabelle ist im Anhang C vorhanden.

*[1] ein großer Levelunterschied

*[2] Ausfallrate $=12*10^{-3}$

*[3] Ausgelöst durch eine Bewegungen einer Eintrittspforte während des Aufbaus

Zuerst wurden Experimente auf dem Opteron durchgeführt, wobei ein einzelner Dämon veranlasst wurde, ein neues Netz zu erstellen, in welchem nur dieser selbst enthalten war. Daraufhin wurden 100 Instanzen im Drei-Sekunden-Takt zum Netz hinzugefügt, wobei die erste Instanz der Eintrittspunkt für das Netz war. Bei diesen wurde der Ausfall verhindert, um sowohl einen stabilen Eintrittspunkt zum Netzwerk zu haben als auch, um ein gewissen „Grundstock“ an Elementen für das Netz zu haben. Daraufhin wurden weitere 2150 Elemente über drei verschiedene Eintrittspunkte im Ein-Sekunden-Takt dem Netz zugeführt.

Ziel der Versuche war es, das Netz mit einem möglichst hohen Level aufzubauen und dann auf ca. fünf Prozent der gesamten Instanzanzahl schrumpfen zu lassen. Deswegen wurde in den ersten drei Versuchen (OP1-OP3) der Ausfall von Instanzen um 1600 Sekunden (nach dem Start einer Instanz) aufgeschoben (Ausfallsverzögerung). Hierdurch ergibt sich ein reibungsloserer Aufbau und die Maximalanzahl an Elementen des Netzes ist erheblich höher als ohne die Verzögerung.

Die Reduktion ist dabei deutlich stärker, als sie in realen Netzen zu erwarten ist. Nach Messungen (siehe [02]) schwankt die Nutzeranzahl, die online ist, in einem unreglementierten Netz wie Gnutelle mittelfristig um ca. 33% (Minimum bis Maximum über eine Woche). Des Weiteren muss man noch einrechnen, dass nicht immer alle Mitglieder innerhalb einer Woche online sind. Aufgrund der hier dargebrachten Anreize sollten allerdings unter der Annahme einer gleichmäßigen Verteilung der Onlinezeiten über den Tag jederzeit 66% der Mitglieder online sein. Wenn man zusätzlich die 33% Schwankung einrechnet, kommt man auf eine Reduktion auf 44% der gesamten Mitgliederzahl. Dementsprechend enthält eine Reduktion auf fünf Prozent eine große Sicherheitsmarge, so dass in der Realität weniger Fehler auftauchen sollten als in den hier gezeigten Versuchen.

In zwei weiteren Experimenten (Op4, Op5) sollte demonstriert werden, dass durch die Ausfallsverzögerung keine Einschränkung der Aussagekraft der Versuche entsteht. Dafür wurde ein Versuch (Op4) wie die anderen (Op1-Op3) durchgeführt, bloß ohne die

Ausfallsverzögerung. Im zweiten wurden die Bedingungen durch eine Erhöhung der Ausfallsrate auf $1{,}2*10^{-3}$ weiter verschärft.

Im Folgenden werden einige Fehler beschrieben und gedeutet, die bei den Versuchen entstanden sind. Hier sei aber darauf hingewiesen, dass es sich um relativ seltene und leichte Fehler handelt, welche nur lokale Konsequenzen haben. Spätestens eine Neueingliederung der entsprechenden Instanz löst das Problem vollständig. Dabei ist die Anzahl an Neueingliederungen, welche durch Fehler hervorgerufen werden, erheblich geringer als solche, welche von den Protokollen veranlasst werden. Diese zusätzlichen Neustarts haben keine negativen Konsequenzen und führen höchstens zu einer weiteren Ebnung des Netzes.

Während der Versuche entstanden bis zu vier „verwaiste“ Konten (siehe Tabelle 4), bei denen bei der Erstellung die Instanz im AH-0 ausfiel, bevor das Konto dort vollständig angelegt werden konnte, aber nachdem es schon auf den anderen Ahs angelegt wurde. In solch einem Fall wird das Protokoll nicht beendet und der Anfrager erhält keine Kontonummer. Der Anfrager beantragt nach einem Timeout erneut ein Konto. Auf dem AH-1 bleibt aber ein verwaistes Konto zurück, welches keinen Besitzer hat und zu dem es kein AH-0 gibt. Dieses Konto wird aber nach 3 Monaten automatisch gelöscht. Solche Konten verursachen während ihrer Existenz eine geringe Netzlast (durch Synchronisationsversuche), stellen aber sonst keinerlei Problem dar (siehe III.4.2.4).

Während eines Versuches entstand bei einer Instanz ein Levelunterschied zwischen den Nachbarn und dem lokalen Knoten von mehr als eins. Da dieses ein Einzelphänomen ist, ist es durch ungünstige Umstände begründet gewesen, in denen diese Instanz ein Split oder Join zu spät erhalten hat. Dieses ist wahrscheinlich durch mehrere verlorene Pakete entstanden. Die Integrität des Netzes ist dadurch aber nicht gefährdet gewesen. Die einzelne Instanz muss hier neu gestartet werden. Dieses ist wie die Daten zeigen ein seltener Fehler, aber auch dieser hat praktisch keine Konsequenzen, da ein automatischer Neustart ihn beheben kann.

Um die Zuverlässigkeit des Routings zu testen, wurden während des Laufens des Netzes Routingpakete von einem festen Eintrittspunkt an zufällige Punkte des Netzes von einer externen Quelle geschickt. Wenn ein solches sein Ziel erreichte, so schickte die entsprechende Instanz direkt ein TCP/IP-Paket an die Quelle zurück. Dabei wurden ca. 7500 Pakete pro Versuch verschickt, von denen nicht einmal ein Promille (siehe Tabelle 4) verloren ging. Solcher Paketverlust kann entstehen, wenn ein Paket verschickt wird, während die Verbindung schon von der anderen Seite aus geschlossen wird. Eine andere Möglichkeit ist es, dass die Instanz ausfällt, die gerade ein Paket bearbeitet. Da das Routingsystem nicht zuverlässig ist, sind solche Ereignisse eingeplant und werden normalerweise durch wiederholtes Senden der Nachricht oder spätere Wiederholung der entsprechenden Aktion ausgeglichen. Die hier gemessene Rate, mit der Pakete verloren gehen, ist dabei als gut zu bezeichnen.

Zusätzlich wurden nicht vollzogene Synchronisationen der AHs (siehe III.4.2.7) gemeldet. Hierbei lag die Gesamtanzahl an Synchronisationen bei ca. 300.000 pro Sitzung, von denen wiederum nicht einmal ein Promille (siehe Tabelle 4) nicht durchgeführt werden konnte (hierbei sind die fehlerhaften Synchronisationen von „verwaisten“ Konten herausgerechnet). Diese Fehler werden durch fehlerhaftes Routing ausgelöst und werden kurze Zeit später nachgeholt. Hierbei entstehen keine Probleme.

Während des Laufes eines Netzes tritt manchmal das Phänomen auf, dass das Knoten-Management-Protokoll (KMP) Instanzen für nicht mehr vorhanden hält, die aber noch voll funktionsfähig sind. Hierbei wird die entsprechende Instanz aus dem Knoten entfernt. Dieser Vorgang, das Entfernen von voll funktionstüchtigen Mitgliedern, wird hier „Mobbing“ genannt. Es tritt immer dann auf, wenn aus unterschiedlichsten Gründen eine Instanz nicht schnell genug auf KKA-Nachrichten reagiert. Dieses kann eine Überlastung des Prozessors sein, aber auch andere Ursachen haben, wie beispielsweise verloren gegangene Nachrichten.

Je größer dabei der entsprechende Timeout ist und je häufiger Versendungen wiederholt werden, desto seltener tritt das Phänomen auf. Der Nachteil eines langen Timeouts ist, dass das Protokoll länger braucht, um ausgefallene Elemente aus dem Knoten zu entfernen. Bei den Versuchen auf dem Opteron wurde ein Timeout von 22 Sekunden gewählt. Dabei traten solche Ereignisse selten auf (siehe Tabelle 4). Eine automatische Neueingliederung behebt den Fehler. Dementsprechend ist auch dieser Punkt ohne Konsequenzen.

Zu guter Letzt wurde noch die Anzahl der Instanzen erhoben, die aufgrund von zufälligen Gegebenheiten nicht in das Netz integriert wurden („fehlerhafte Einwahl“). Solche Zufälle können beispielsweise sein, wenn eine „Eintrittspforte“ gerade das Bootprotokoll wiederholt (siehe III.4.1.3.3) oder eine entsprechende WTC-Nachricht verloren geht. Hier bietet sich an, dass der Benutzer das Programm einfach neu startet, was aber auch automatisch passieren könnte. Bei der Mehrzahl der Versuche war dieses ein sehr seltenes Ereignis, auch weil die Eintrittspforten des Netzes kein zufälliges Ausfallen durchführten. Im Versuch Op5 führte jedoch eine der Eintrittspforten das Bootprotokoll noch während des Aufbaus erneut durch. Dieses verursachte eine relativ große Anzahl an solchen Fehlern. In der Realität kommen aber mehrere Eintrittspforten für jede Instanz in Frage, so dass der Fehler noch viel seltener auftreten würde.

IV.2.1.2. Versuche auf den PCs

Versuch	Ausfalls-verzögerung	#Instanzen: Maximum	#Instanzen: Minimum	Dauer in h:min	fehlerhafte Einwahl	Mobbing	keine Synchro-nisation in %	verwaiste Konten	verlorene Routing Pakete in %
PC1*[1]	0	321 Lv5	13 Lv0	3:20	12	21	0,166	2	0,021
PC2	0	347 Lv5	12 Lv0	3:15	2	1	0,086	0	0,048
PC3	0	372 Lv5	11 Lv0	3:03	2	9	0,003*[2]	1	0

Tabelle 5: Kennzahlen bei Belastungstestläufen des Netzwerkes auf zwei PCs.

Erläuterungen und Wertungen werden im folgenden Text gegeben, eine ausführlichere Tabelle ist im Anhang C vorhanden.

*[1] Knotenmanagement-Protokoll-Timeout =20 Sekunden

*[2] zusätzlich traten 140 Synchronisationsfehler bei einer Instanz auf, wobei diese Instanz darauf die Verbindung zum Netz (vollständig) verloren hatte. (Knoten verschwunden).

Ziel der Versuche auf den PCs war es, die Funktionsweise der Protokolle auch auf einem realen Netzwerk zu überprüfen. Auf Grund der geringen Leistungsfähigkeit der PCs wurden auf jedem nur 200 Instanzen gestartet, so dass es insgesamt 400 Elemente im Netz gab.

Beim ersten Versuch (PC1) wurden die gleichen Parameter übernommen, die sich für den Server als günstig herausgestellt hatten. Wohl durch die Verzögerungen des Netzwerkes stellte sich hierbei aber eine höhere Mobbingrate ein (siehe Tabelle 5), welche in weiteren Versuchen (PC2,PC3) durch eine Verlängerung des Timeouts des KMPs ausgeglichen wurde.

Insgesamt verhielt sich das Netz hier ähnlich wie auf dem Server (siehe Anhang C). Es war nur eine leichte Erhöhung der Anzahl der Instanzen festzustellen, bei denen die Einwahl in das Netz fehlerhaft war.

In einem Fall - im Versuch PC3 - trat ein Fehler auf, bei dem auf einer einzelnen Instanz erst einmal alle Synchronisationsversuche scheiterten (180 an der Zahl) und einige Zeit später gemeldet wurde, dass der Kontakt zu mindestens einem Nachbarknoten vollständig verloren gegangen wäre (Knotenverlust siehe IV.2.2.4). Der Autor deutet dieses als eine plötzlich korrumpierte Nachbarschaftstabelle. Dabei werden Knoten als Nachbarn gesehen, die keine Nachbarn mehr sind. Wenn die fehlerhafte Instanz jetzt eine Verbindung zum Nachbarknoten aufbaut, so wird diese zwar vom Kernel akzeptiert, aber die Anwendung schließt sie sofort wieder. Die fehlerhafte Instanz erkennt dabei auf Grund von Implementierungsdetails nicht, dass der Nachbar gar keiner ist. Dadurch ist keine Verbindung zum Netz mehr gegeben und somit können auch keine Konten synchronisiert werden. Da keine Veränderungen mehr von den scheinbaren Nachbarn bekannt gegeben werden, veralten die Repräsentationen so lange, bis keine Instanz von einem der scheinbaren Nachbarknoten mehr online ist und deswegen gemeldet wird, dass ein Knotenverlust auftrat. Bei dieser Erklärung ist die Ursache der fehlerhaften Nachbarschaftstabelle unbekannt. Die Erklärung bedarf deswegen der Überprüfung, was allerdings problematisch ist, da solch ein Fall bis jetzt nur einmal in allen Versuchen (nicht nur den hier dokumentierten) auftrat. Aufgrund des seltenen Auftretens und der lokalen Begrenzung der Auswirkungen ist er aber nicht von großer Bedeutung.

Bei den Versuchen war außerdem auffallend, dass der Hauptteil, der durch das Netz gebrauchten Rechenzeit vom Kernel (also Systemzeit) war. Dieses könnte man dadurch erklären, dass die Prozeduren zur Bearbeitung von Paketen sehr einfach sind und kaum Rechenzeit beanspruchen und so der Hauptteil der Zeit für die Nachrichtenversendung gebraucht wird. Eine andere Möglichkeit dieses zu erklären wäre, dass der hohe Anteil an Kernelzeit durch die Vielzahl an Taskwechseln und das damit verbundene Scheduling zustande kommt.

Des Weitern war interessant, dass es hauptsächlich während der Aufbauphasen der Netze zu scheinbar willkürlichen Zeitpunkten zu einem sprunghaften Anstieg des Rechenzeitbedarfs kam, der dann nach einer oder mehreren Sekunden genauso sprunghaft wieder zurückging. Dabei änderte sich das Verhältnis zwischen Kernel- und Userspace-Rechenzeit scheinbar nicht. Außerdem war manchmal der Rechenzeitanstieg auf beiden Computern parallel zu beobachten. Deswegen ist davon auszugehen, dass es sich hierbei um eine Eigenschaft des Netzes handelt. Der Rückgang der Rechenzeit spricht dabei dafür, dass es sich um vorübergehende Zustände im Netz handelt. Dieses könnten beispielsweise behebbare Fehlerzustände oder Ereignisketten sein, bei denen plötzlich die Nachrichtenanzahl drastisch ansteigt (weitere Beschreibungen unter V.1.2). Diese Beobachtung könnte Anlass bieten, um die entsprechenden Zustände genauer zu untersuchen. Da aber keine Fehlermeldungen mit diesem Phänomen einhergehen, ist es als unkritisch einzustufen.

IV.2.1.3. Zusammenfassung

Mit diesen Versuchen konnte gezeigt werden, dass die Programme und damit die Protokolle auch der hier gewählten starken Belastung gewachsen sind. Dabei wurden erheblich härtere Bedingungen benutzt, als sie in der Realität zu erwarten sind.

Dabei traten nur Fehler auf, welche leichte und lokale Konsequenzen haben. Man kann davon ausgehen, dass unter realen Bedingungen noch weniger Fehler entstehen, welche auch nur lokal sind und leichte Konsequenzen haben. Solche Fehler sind aber ohne Bedeutung und könnten automatisch behoben werden.

Daraus folgert der Autor dieser Arbeit, dass mit diesen Testläufen gezeigt wurde, dass die Programme und damit auch die Protokolle, die in diesen verwirklicht sind, für die gestellte Aufgabe mindestens prinzipiell geeignet sind.

IV.2.2. Genauere Betrachtungen

In weiteren Experimenten wurde versucht, ein genaueres Verständnis für das Netz und seine Grenzen zu erhalten. Dabei wurden teilweise noch extremere Bedingungen erstellt als bei dem Zeigen der Angemessenheit. Hierbei wurde bewusst das Netz an seine Grenzen geführt, welche jedoch weit außerhalb der Werte liegen, die in der Realität erwartet werden.

IV.2.2.1. Bandbreitenverbrauch

Als ein weiterer Schritt wurde in einem Netz der Bandbreitenverbrauch gemessen. Hierfür wurde ein Netz mit 160 „Benutzern“ auf einem PC aufgebaut und die Belastung zu zwei verschiedenen Zeitpunkten gemessen. Dabei kamen im Bereich der maximalen Anzahl an Mitgliedern (ca. 140 Mitglieder), die online waren, ca. 55kb/sek. für das gesamte Netz zustande. Am Ende des Versuches (ca. 30 Mitglieder) wurde eine ähnliche Messung noch einmal durchgeführt, wobei 42kb/sek. für das gesamte Netz gemessen wurden. Dabei fällt auf, dass dieser Wert relativ wenig gesunken war, obwohl sich die Anzahl an Mitgliedern, die online waren, auf ein Viertel reduziert hatte. Hieraus könnte man folgern, dass die Hauptlast durch die Anzahl der Konten entsteht und nicht über die Anzahl an Mitgliedern skaliert, die online sind. Diese Erklärung müsste aber noch genauer überprüft werden.

IV.2.2.2. Grenzen des Netzaufbaues

Weiter wurde versucht, die Leistungsgrenzen des Netzes beim Hinzufügen und Entfernen von Mitgliedern auszutesten. Hierzu wurde nur das Netz benutzt, ohne Belastung durch die Anwendung und ohne dass sich Instanzen zufällig selber beendeten.

Hierbei wurde zweimal versucht, 200 Instanzen in kurzer Folge in das Netz einzugliedern. Im ersten Versuch wurde ein Abstand von einer Sekunde zwischen dem Start der einzelnen Instanzen gewählt. Dabei wurde ein Maximum von 192 laufenden Instanzen auf dem Level vier erreicht. Dementsprechend hatten nur acht Instanzen es nicht geschafft, sich in das Netz zu integrieren. Hierbei könnte die Ursache sein, dass der Kernel keine neuen Verbindungen mehr akzeptierte (wegen einer zu großen Anzahl an unbestätigten Verbindungen). Es könnte aber auch ein Verlust von Routingpaketen oder einfach eine zu lange Zeitspanne bis zur Beantwortung der entsprechenden Anfrage schuld sein.

Im zweiten Versuch wurde der Startabstand auf 0,5 Sekunden halbiert. Hierbei fanden 36 Instanzen das Netz nicht und es wurde von zwei Instanzen gemeldet, dass es einen Levelunterschied zwischen ihnen und den Nachbarn von mehr als einem Level gäbe. Diese Elemente beendeten sich daraufhin selbst. Da nur zwei Instanzen diesen „großen Levelunterschied“ gemeldet haben, ist davon auszugehen, dass es keine Inkonsistenz im Netz gab, sondern nur eine „verzögerte“ Nachrichtenübermittlung hieran schuld war. Dementsprechend könnte entweder ein Split im eigenen Knoten nicht schnell genug mitgemacht worden sein und/oder es wurde der Split eines Nachbarn zu spät gemeldet.

Hier scheint die Grenze des Netzes, neue Mitglieder aufzunehmen, erreicht zu sein, ohne inkonsistent zu werden. Die Anzahl an Instanzen, die nicht mehr das Netz finden, steigt rapide an und die Grenze, an der die Nachrichten nicht mehr schnell genug verarbeitet werden können, ist fast erreicht. Um die Geschwindigkeit, mit der sich Nachrichten ausbreiten, zu erhöhen, könnte man nicht für jedes neue Mitglied eine einzelne Nachricht schicken, sondern eine bestimmte Zeit warten und dann die Informationen über alle neuen Mitglieder aus diesem Zeitraum zusammen verschicken.

In einem realen Netz ist die Ereignisrate pro Knoten wichtig. In diesem Fall wurde eine „Hinzufügerate“ von maximal 2 Ereignissen pro Sekunde erreicht (0,5 Sekunden Abstand in einem Knoten (Level null)). Um die Rate für ein reales Netz ausrechnen zu

können, müssen mehrere Annahmen getroffen werden. Zuerst gehe ich von einem statischen Netz aus, also davon, dass die Netzgröße über den ganzen Tag gleich bleibt. Dieses ist zwar nicht besonders realistisch, aber vereinfacht die Rechnung. Zusätzlich wird angenommen, dass ein Knoten 18 Mitglieder (Minimum* 2 Knoten * Schwankungssicherheit =6*2*1,5; siehe III.4.1.2.3 und IV.1.1) hat, welche jeweils 16 Stunden (siehe III.1.5) pro Tag online bleiben. Dementsprechend werden 27 Personen gebraucht, damit (durchschnittlich) am ganzen Tag die 18 Mitglieder online sind. Folglich gibt es in 24 Stunden 27 Eintritte und 27 Austritte, was einer Ein- bzw. Austrittsrate von 1,125 Personen pro Stunde und Knoten entspricht. Durch die hohe Differenz zwischen den hier gezeigten Grenzen des Netzes und den erwarteten Anforderungen ist ein genügend großer Sicherheitspuffer vorhanden.

IV.2.2.3. Grenzen des Netzabbaus

In weiteren Versuchen wurde ein „rohes“ Netz von 200 Instanzen benutzt. Diesmal wurde versucht, die maximale Geschwindigkeit des Abbaus eines Netzes zu messen. Hierfür wurden die Elemente in der gleichen Reihenfolge beendet, in der sie dem Netz hinzugefügt wurden. Durch den zufälligen Wert der Ringposition des Knotenmanagementprotokolls (siehe III.4.1.2) ist die Position der Instanzen im Netz zufällig. Um nicht die „Länge“ des Timeouts des KMP-Protokolls zu messen, meldete sich hier jedes Element ordnungsgemäß ab.

In einem ersten Versuch wurden die Elemente mit einem Abstand von einer Sekunde (plus einer unbekannten Versendungszeit der entsprechenden Nachricht) entfernt. Der Abbau lief ohne Probleme. Dabei führten 51 Instanzen erneut das Bootprotokoll aus, um Unterschiede im Level/Mitgliederzahl von Nachbarn auszugleichen. Sechsmal waren bei solch einem Unterschied nicht genügend Mitglieder vorhanden, weswegen ein Join beantragt wurde.

In einem zweiten Versuch wurde der Abstand auf 0,5 Sekunden halbiert (plus Versandzeit) und auch dabei lief alles problemlos. Bei einer weiteren Halbierung des Abstandes auf 0,25 Sekunden (plus Versandzeit) kam es zu einem Absturz des Netzes, als es noch 20 Mitglieder im Netz gab, welche sich auf dem Level drei befanden. Hierbei ist unter Absturz zu verstehen, dass ein Knoten alle seine Mitglieder verloren hatte, bevor dieses von den Protokollen ausgeglichen werden konnte. Ein Abstand von 0,1 Sekunden führte zu einem fast sofortigen Absturz des Netzes.

Damit man eine realistische Rate ableiten kann, muss man mindestens zwei Knoten haben, die Mitglieder austauschen können. Ein Abstand von 0,5 Sekunden entspricht folglich einer maximalen Rate (auf dem Level eins gleich zwei Knoten) von 1 Ereignis pro Sekunde pro Knoten. Dieser Wert ist wiederum erheblich höher als die 1,125 Personen pro Stunde und Knoten, welche unter „Grenzen des Netzaufbaues“ (IV.2.2.2) als Ausfallsrate für einen Knoten errechnet wurden.

IV.2.2.4. Tests zur Knotengröße und Ausfallwahrscheinlichkeit

In dem theoretischen Teil der Arbeit wurde gefordert, dass es praktisch nicht vorkommt, dass alle Mitglieder eines Knotens diesen verlassen, bevor dieses vom Netz ausgeglichen werden kann. Dieses wird als Knotenverlust im Folgenden bezeichnet. Ein solcher Verlust führt in jedem Fall zu einem Datenverlust und durch die nicht vorgesehen Gegenmaßnahmen zu einem Verlust der Integrität des Netzes.

Dementsprechend wurde der Zusammenhang zwischen der Ausfallswahrscheinlichkeit und der Knotengröße untersucht, um die theoretisch begründete Forderung praktisch zu verifizieren.

In diesen Versuchen wurde versucht, 200 Elemente im Dreisekundentakt dem Netz hinzuzufügen. Im ersten Teil war dabei die Ausfallwahrscheinlichkeit mit $1{,}2*10^{-3}$ sehr hoch angesetzt und wurde nicht verändert. Die Knotengrößen wurden jedoch variiert. Der Timeout des KMP's war dabei auf 30 Sekunden gesetzt und nach 15 Sekunden fanden die Reconnect-Versuche statt (siehe III.4.1.2). Dabei wurde die Zeit gemessen, bis es zu einem Knotenverlust kam bzw. bis das Netz auf den Level Null geschrumpft war. Es wurde eine so große Ausfallwahrscheinlichkeit gewählt, damit der Übergang zwischen einem instabilem Netz, in welchem es häufig Knotenverluste gibt, zu einem stabilem, in dem diese sehr selten vorkommen, gezeigt werden kann.

Mit jeder Knotengröße wurden jeweils vier Versuche durchgeführt, um die Zeiten ein wenig statistisch zu sichern. Der Verlust passierte in den meisten Fällen noch während des Aufbaus. Die dabei entstanden Daten sind in der folgenden Tabelle zusammengestellt, wobei die Zeiten von Versuchen, in denen es zu keinen Absturz kam, eingeklammert sind:

Knotengröße (MIN.-MAX.)	**2-4**	**3-6**	**4-10**	**10-25**
Dauer Versuch 1 (in sec)	101	772	973	(2100)
Dauer Versuch 2 (in sec)	205	478	1060	(1500)
Dauer Versuch 3 (in sec)	238	209	(2345)	(1380)
Dauer Versuch 4 (in sec)	272	660	902	(1700)
Mittelwert	204	529,8	1320	
Standardabweichung	64,0	212,8	594,4	

Tabelle 6: Zeit bis zum Absturz des Netzes bei fester Ausfallswahrscheinlichkeit und variabler Knotengröße

Die Zeiten von Versuchen, in denen es zu keinen Absturz kam, sind eingeklammert.

Man erkennt also, wie erwartet, einen starken Zuwachs der Stabilität des Netzes mit steigender Größe der Knoten.

Im zweiten Teil wurde die Ausfallswahrscheinlichkeit variiert, während die Knotengröße mit 3-6 „Computern" konstant gehalten wurde.

Ausfallwahrscheinlichkeit	**$2{,}4*10^{-3}$**	**$1{,}2*10^{-3}$**	**$0{,}6*10^{-3}$**
Dauer Versuch 1 (in sec)	393	772	(>1500)
Dauer Versuch 2 (in sec)	390	478	(>1500)
Dauer Versuch 3 (in sec)	442	209	(>1500)
Dauer Versuch 4 (in sec)	380	660	(>1500)
Mittelwert	401	529,8	-
Standardabweichung	27,7	212,8	-

Tabelle 7: Zeit bis zum Absturz des Netzes bei variabler Ausfallswahrscheinlichkeit und fester Knotengröße

Die Zeiten von Versuchen, in denen es zu keinen Absturz kam, sind eingeklammert.

Wie erwartet zeigen die Daten einen deutlichen Einfluss der Ausfallswahrscheinlichkeit bzw. der Knotengröße auf die Stabilität des Netzes. Damit ist gezeigt, dass man ein stabiles Netz erzielt, wenn man die Knotengröße richtig an die Ausfallswahrscheinlichkeit anpasst.

Kapitel V

Ausblick und offene Punkte

Hauptsächlich kann man die Arbeit in drei Bereichen weiterführen. Erstens kann man die Tests und Analysen erweitern. Hierbei bieten sich vor allem Praxistests an. Zweitens kann man unter anderem die aus solchen Tests gewonnenen Erkenntnisse zur Verbesserung der Protokolle nützen. Und drittens kann bzw. muss man noch Zusatzfunktionalitäten erarbeiten und implementieren.

V.1. Tests und Analysen

Bei den Tests gibt es zwei besonders wichtige Punkte. Zum Ersten ist eine Analyse des Benutzerverhaltens in Hinblick auf das Abrechnungsverfahren (siehe III.1.5) interessant. Der zweite Punkt von gesteigertem Interesse ist eine genaue Analyse des Verhaltens des Peer-to-Peer-Netzes unter realen Bedingungen.

V.1.1. Untersuchung des Benutzerverhaltens

Unter Kapitel II wurde aus der Literatur eine Analyse des zu erwartenden Benutzerverhaltens erstellt und unter III.1 zu einem Abrechnungsverfahren weiterentwickelt. Dieses stützt sich zwar auf viele Versuche und Studien, allerdings sind diese in anderen Kontexten gemacht worden. Zusätzlich spielen hier sehr viele äußere Faktoren mit, die nicht unbedingt vollständig erfasst wurden. Da dem Autor dieser Arbeit auch kein vergleichbares Abrechnungssystem bekannt ist, handelt es sich bei dem hier vorgestellten um ein hauptsächlich theoretisch begründetes System.

Dementsprechend ist es sinnvoll, die theoretische Analyse durch praktische Untersuchungen zu ergänzen und zu bestätigen. Dazu sind mehrere verschiedene implementierte Abrechnungssysteme nötig, so dass man die dadurch gewonnenen Ergebnisse miteinander vergleichen kann. Es ist nämlich nicht damit zu rechnen, dass die benutzten Mechanismen nur bewusst wirken. Daher kann man nicht alleine mit Fragebögen oder ähnlichem die Reaktionen der Benutzer erfassen.

Außerdem braucht man auch eine genügende Teilnehmerzahl und relativ viel Zeit, um die ermittelten Daten statistisch zu sichern. Um dieses zusätzlich zur Entwicklung des Systems zu realisieren, ist ein erheblich höher Zeitaufwand nötig, als er hier zur Verfügung stand.

Interessante Punkte sind hierbei, abgesehen von dem Belohnungsverhalten, auch die Auswirkungen von Wertänderungen verschiedener Parameter, wie beispielsweise die des Sättigungspunktes (siehe III.1.5).

V.1.2. Analyse des Protokoll-Verhaltens

Ähnlich wie beim Benutzerverhalten wurde auch bei dem Verhalten der Protokolle auf eine theoretische Überlegung zurückgegriffen. Das Zusammenspiel von verschiedenen Protokollen kann aber recht komplex und sehr schwer vorhersehbar sein. Es wurde zwar praktisch getestet und damit gezeigt, dass das System funktioniert, allerdings bleiben dabei viele Unbekannte, wie es im Detail arbeitet bzw. ob es nicht in bestimmten Situationen Optimierungsmöglichkeiten oder Fehlerzustände gibt. Zusätzlich wurden alle Versuche mit einer sehr kleinen Anzahl an realen Computern durchgeführt, auf denen jeweils eine größere Menge an Instanzen lief. Für realitätsnähere Versuche sollten jedoch wenige Instanzen auf einer Vielzahl an Computern über eine sehr lange Zeit laufen. Dabei sollten es sowohl kurzfristige als auch langfristige Schwankungen der Benutzeranzahl vorhanden sein. Kurz gesagt, es sollte, wenn nicht in der Realität, so doch in einem Szenario getestet werden, welches so realitätsnah wie nur möglich ist.

Ein weiterer Punkt, der genauer untersucht werden sollte, ist die Geschwindigkeit, mit der sich Veränderungen im Netz ausbreiten. Dieser Parameter bestimmt die maximale Veränderungsrate, die das Netz verkraften kann, ohne inkonsistent zu werden. Hierbei gibt es mit großer Wahrscheinlichkeit Zustände, in denen schneller reagiert wird, und solche, in denen es länger dauert. Die Identifizierung solcher „schlechten“ Konstellationen wäre wünschenswert, so dass man diese umgehen kann oder das Protokoll so verändern kann, dass es auch mit diesen Zuständen gut zurechtkommt.

Des Weiteren sollte nach „Flimmer-Zuständen“ gesucht werden. Hiermit sind Situationen gemeint, in der eine begrenzte Menge an Zuständen innerhalb einer bestimmten Zeit zyklisch ineinander übergehen. Diese können durch nicht beachtete Sonderfälle innerhalb eines Protokolls oder durch das nicht optimale Zusammenspiel verschiedener Protokolle entstehen.

Zusätzlich sollte auf die Synchronisation von Ereignissen und Nachrichten geachtet werden. Hierdurch kann es beispielsweise zu „Nachrichten-Explosionen“ kommen, welche ein Netz schwer beeinträchtigen können. Hiermit ist gemeint, dass sehr viele Nachrichten in kurzer Zeit entstehen. Dieses kann durch instabile Zustände hervorgerufen werden, bei denen ein einziges Ereignis viele weitere hervorruft. Ein Beispiel hierfür wäre, dass alle Knoten ihren Level in sehr kurzer Zeit wechseln, weil ein einziger Computer das Netz verlassen hat.

Ferner ist noch die Unbestimmtheit des Zieles bei der Durchführung des Bootprotokolls von Interesse. Hiermit ist gemeint, wie vorbestimmbar der Endknoten ist, wenn mit lokal vorhandenem Wissen versucht wird, den Knoten vorauszusagen, in den man landet, wenn man den zweiten Teil (die Suche) des Bootprotokolls durchführen lässt. Diese Ungewissheit ist ein wichtiger Teil der Sicherheitsmaßnahmen des Netzes. Lässt sich der Endknoten mit hoher Wahrscheinlichkeit vorhersagen, ist das Wandern durch das Netz relativ einfach.

V.2. Verbesserung der Protokolle und des Abrechnungsverfahrens

Aus dem in den Tests und Analysen gesammelten Wissen kann man dann Parameter weiter anpassen oder, wo nötig, Protokolle ändern oder gar neu entwerfen.

Denkbar ist beispielsweise beim Abrechnungsverfahren, dass man den Sättigungspunkt, nachdem man ihn in Tests ermittelt hat, anpasst und damit die gesamte Berechnung des Nutzens aus dem Engagement verändert.

Eine andere Änderung, diesmal im Bootprotokoll, ist, dass man bei einer hohen Vorhersagbarkeit des Endknotens (siehe V.1.2) das Protokoll von einer einfachen deterministischen Suche zu einer mittels „simulierten Abkühlung“ (simulated annealing) oder

einem anderen indeterministischen Verfahren hin ändert. Das Problem dabei ist, dass diese Verfahren zwar eher das globale Minimum finden und indeterministisch sind, aber dafür brauchen sie auch mehr Zeit. Ob dieses akzeptabel ist, müssen Messungen der Reaktionsgeschwindigkeit zeigen.

Eine weitere mögliche Anpassung ist die Entwicklung eines vollparallelen Designs, welches ohne die Synchronisation der Uhren auskommt. Dieses bedeutet aber einen erheblichen Mehraufwand an Rechenkapazität und / oder Nachrichten. Deswegen und wegen der erheblich höheren Komplexität bei der Entwicklung solch eines Protokolls (ohne dass dadurch notwendigerweise bessere Resultate erzielt werden könnten) wurde dieses hier nicht entwickelt.

V.3. Zusatzfunktionalitäten

V.3.1. Routing, bei Abbuchung des Nutzens („underlay routing“)

Eine Zusatzfunktionalität, die für die Gemeinde zwar nötig ist, aber nicht direkt zu den Protokollen gehört, ist das „Freischalten“ eines Benutzers zum Surfen, wenn dieser eine Abbuchung eingereicht hat. Weil es nicht direkt zu den Protokollen gehört und einiges an Zeit kostet, wurde dieses nicht implementiert. Theoretisch könnte man zwar mittels einer Firewall und einem „NAT“ (Network Address Translation) dieses implementieren, aber hierdurch würde die IP des „Providers“ (dem Mitglied, über das gesurft wird) im Internet sichtbar. Dementsprechend wäre dessen Anonymität nicht mehr gegeben.

Deswegen sollten Alternativen wenigstens in Betracht gezogen werden. Eine Möglichkeit wäre das Routen von den entsprechenden Paketen durch das Netz, welche erst an einer zufälligen Stelle wieder zu normalen IP-Paketen werden. Eine andere Möglichkeit wäre die Benutzung öffentlicher Anonymiser bzw. des Onion-Routing-Networks (siehe[60], [52]). Aber wenigstens die erste Möglichkeit erfordert noch eine Menge an Analysen (beispielsweise welche Routinglasten verträgt das Netz) und anderen Überlegungen.

V.3.2. Hotspot-Finder

Eine Erweiterungsmöglichkeit ist das Implementieren eines „Hotspot-Finders“. Hiermit ist ein System gemeint, mit welchem man Hotspots finden und lokalisieren kann, welche in der Nähe eines bestimmten Ortes sind. Dieses hätte den Vorteil, dass man herausfinden kann, ob in der Nähe eines zukünftigen Aufenthaltsortes die Gemeinde in Anspruch genommen werden kann oder nicht. Außerdem kann man den Nutzen, den man wirklich bekommt, abschätzen, bevor man der Gemeinde beitritt.

Allerdings hat diese Erweiterung auch mehrere Nachteile. Die Nützlichkeit dieses Systems ist nur begrenzt, da nicht bekannt ist, ob zu dem Zeitpunkt, an dem man surfen will, solch ein Hotspot auch wirklich aktiv ist. Zusätzlich ist solch ein Service auch nur verfügbar, wenn man schon im Internet ist. Dementsprechend kann man nicht darauf zugreifen, wenn man ihn am dringendsten braucht, nämlich wenn man unvorhergesehen ins Internet will oder wenn sich herausstellt, dass all diejenigen Mitglieder gerade offline sind, die man sich vorher herausgesucht hatte, um über sie zu surfen.

Hinzu kommt noch das Problem, dass durch den Hotspot-Finder die Anonymitätsforderung (siehe III.2.1.1.2) gebrochen wird. Durch solch ein System ist es möglich, die Anschrift und damit den Namen von vielen Mitgliedern herauszufinden.

Zu guter Letzt soll noch auf Implementierungsprobleme hingewiesen werden. Hier ist zu sagen, dass sich ein Knoten seiner realen Position bewusst sein muss. Dieses können entweder die Postadresse oder aber die GPS-Koordinaten sein. Ersteres hat den Vorteil, dass es am ehesten der normalen menschlichen Denkweise entspricht, allerdings muss dann auch eine Abstandsmessung über diese Metrik entwickelt werden. Dieses ist sehr aufwändig und erfordert vollständige Karten von dem Gebiet, in dem die Gemeinde verteilt ist.

Die Benutzung des GPS-Systems hat den Nachteil, dass man dazu weitere Hardware zur Feststellung der Koordinaten benötigt.

Kapitel VI

Zusammenfassung

Ziel der Arbeit war die Entwicklung von sicheren Protokollen für ein Peer-to-Peer-Hotspot-Accounting-System. Dabei waren Privatpersonen die Zielgruppe. Um dieses zu erreichen, wurde eine Analyse der Vorraussetzungen durchgeführt und ein darauf basierendes Abrechnungsverfahren entwickelt, welches die Privatpersonen dazu bewegen sollte, sich möglichst stark zu engagieren. Ein Peer-to-Peer-Netzwerk wurde entwickelt und implementiert, welches Grundvorrausetzungen für ein sicheres Abrechnungssystem schafft. Zusätzlich wurde ein sicheres Hotspot-Accounting-System entworfen und verwirklicht, welches das Netzwerk als Basis benutzt. Das Gesamtsystem wurde ausgetestet und es wurde dabei gezeigt, dass es den angenommenen Anforderungen gewachsen ist.

Kapitel VII

Referenzen

Viele der hier genannten Dokumente finden sie in elektronischer Form auf der beiliegenden CD-ROM (siehe Anhang D).

[01] Paper, Tsuen-Wan Ngan, Dan S. Wallach, and Peter Druschel,“Enforcing Fair Sharing of Peer-to-Peer Resources”, In Proc. IPTPS, Berkeley, CA, Feb. 2003, Second International Workshop on Peer-to-Peer Systems

[02] Paper, Stefan Saroiu, P. Krishna Gummadi, Steven D. Gribble, “A Measurement Study of Peer-to-Peer File Sharing Systems”,Seattle,2002

[03] Paper, Yatin Chawathe, et al., „Making Gnutella-like P2P Systems Scalable“, Sigcomm 03, Karlsruhe, 25-29 August 2003

[04] Paper, Rieck K., “Epidemische Algorithmen, Seminar Informationsmanagement im Web“, Berlin, Sommer 2003

[05] Paper, Kamvar,S.D., Schlosser M.T., Garcia-Molina H.,“The EigenTrust Algorithm for Reputation Management in P2p Networks“, Budapest, May 2003

[06] Paper, Tamilmani K., Mohr A.E., “Trading with Credits: An Incentive Framework for P2P File Sharing Networks”,New York, 2004

[07] Webpage, “Crucial Paradigm Business Solutions”,“Hacking Attacks - How and Why”, Available from: http://www.crucialparadigm.com/resources/tutorials/website-web-page-site-optimization/hacking-attacks-how-and-why.php Accessed on: 1.12.2004

[08] Webpage, unbekannte Autoren, ÖAMTC AKADEMIE, 2000, Delphi - Studie “Zukunft der Mobilität“ , Available from: http://www.oeamtc.at/netautor/pages/resshp/anwendg/1007720.html Accessed on: 28.5.2004

[09] Webpage, unbekannte Autoren, “Mobilität”, Available from: http://www.stmwivt.bayern.de/energie/energiespartipps/frames/06mobil.html

[10] Webpage, DPA,10.7.2002, “Statistik Familien haben Handy und PC”, published at TecChannel, Available from: http://www.tecchannel.de/news/20020710/thema20020710-8072.html Accessed on: 28.5.2004

[11] Paper, Raùl Lòpez Pèrez, “Anger and Guilt: An Alternative to InequityAversion Models”, Barcelona, April 2004

[12] Recommendation, Anton, B., Bullock, B., and Short, J. Best Current Practices for Wireless Internet Service Provider Roaming, Wi-FiAlliance Public Document, Feb. 2003. http://www.wi-fi.org.

[13] Paper, Elias C. Efstathiou and George C. Polyzos, “The Peer-to-Peer Wireless Network Confederation: Enabling Global Wi-Fi Roaming”, Athen,2004

[14] Paper, Elias C. Efstathiou and George C. Polyzos, “A Peer-to-Peer Approach to Wireless LAN Roaming”,Athen, September 2003

[15] Paper, David Barkai, „Technologies for Sharing and Collaboratin on the Net“, IEE 2002

[16] Webpage , Bruce Bower, „A Fair Share of the Pie”,Science News, Feb. 2002 Vol 161, published at: http://www.starbuilders.org/fft/articles/fairshare.html Accessed on: 29.5.2004

[17] Webpage, Sean Markey,”Monkeys Show Sense of Fairness, Study Says”, National Geographic News, 17 September 2003, published at: http://news.nationalgeographic.com/news/2003/9/0917_030917_monkeyfairness.html , Accessed on: 30.5.2004

[18] Paper, Florian Englmaier and Achim Wambach, “Contracts and Inequity Aversion”, München&Nürnberg, September 2002

[19] Paper, William R. Nelson Jr.,“A Fair Consensus: Experimenting with the Mercy Game”,Buffalo ,2002

[20] Paper, James Konow, “A positive theory of economic fairness”,Journal of Economic Behavior & Organization, Vol. 31 (1996) p13-35

[21] Artikel, Michel Balinski, “Die Mathematik der Gerechtigkeit”, Spektrum der Wissenschaft, März 2004, p 90-97

[22] Buch, Philip G. Zimbardo und Richard J. Gerring, „Psychologie“, Auflage 7., 1996, Springer, ISBN 3-540-64633-7, p206-226

[23] Webpage, J. David Perry,” Unit 2: Behaviorism ”, Indiana University, Bloomington, Published at: http://education.indiana.edu/~p540/webcourse/behave.html Accessed on: 17.6.04

[24] Webpage, M.H. Dembo,“Learning Theories --- A Primer Exercise“, University of Southern California,1998, Published at: http://www.duq.edu/~tomei/ed711psy/b_skin.htm, Accessed on 17.06.2004

[25] Webpage, W. Hitt, “Principles for Using Behavior Modification”, 1998, Published at: http://chiron.valdosta.edu/whuitt/col/behsys/ Accessed on 17.06.2004

[26] Webpage, Unbekannter Autor, “A Primer on Education”, 2002, Published at: http://www.learnativity.com/edpsych.html, Accessed on 17.06.2004

[27] Webpage, B.F. Skinner,”A Brief Survey of Operant Behavior”, published at: http://www.bfskinner.org/Operant.asp, Accessed on 17.06.2004

[28] Script, J.E.R. Staddon, „Adaptive Behavior and Learning“,”Chapter 5 Reward and Punishment“, Duke University, 2003, published at http://psychweb.psych.duke.edu/%20department/jers/abl/Chapter05.pdf, Accessed on 9.06.2004

[29] Buch, Werner Correl, “Einführung in die Pädagogische Psychologie”, Auflag 3,1969, Verlag Ludwig Auer, p 69-89

[30] Webpage, T-Mobile, “T-Mobile, Pressemitteilung am 18.März 2004 über die Datenoption 18/3, Bonn/Hannover”, Published at: http://www.t-mobile.de/presse/cebit2004/daten/1,5667,8235-_,00.html ,Accessed on: 30.6.2004

[31] Webpage, Anonymer Autor, “UMTS-Anbieter präsentieren erste WLAN-Tarife, Abrechnung nach dem Parkuhrprinzip”, Published at: http://www.zdnet.de/mobile/tkomm/0,39023192,39121564-2,00.htm , Accessed on:30.6.2004

[32] Webpage, Erich Brenner und Wilfried Eckl-Dorna , „WLAN - Millionengeschäft oder Flop? Neben UMTS werden immer mehr öffentlich zugängliche Funknetze für das mobile Surfen im Internet aufgebaut“, Published at: http://www.ispa.at/www/getFile.php?id=346, Accessed on 5.7.2004

[33] Webpage, Carlo Longino, “Horses for Courses” Juli 2003, Published at: http://www.thefeature.com/article?articleid=44626&ref=1703119, Accessed on: 5.7.2004

[34] Paper, R. Bhagwan, S. Savage and G. M. Voelker “Understanding Availability”, San Diego, 2003

[35] Webpage: anonymer Autor, Gnutella Network Portugal, “Some Basic Rules”, published at http://faqgnetpt.no.sapo.pt/Main.htm, Accessed on: 7.7.04

[36] Paper, J.R. Douceur, “The Sybil Attack”,Microsoft Research, First Int. Workshop on Peer-to-Peer Systems (IPTPS 2002), Cambridge, Massachusetts, März 2002

[37] Paper, E.Sit, R. Morris,“Security Considerations for Peer-to-Peer Distributed Hash Tables”,Laboratory for Computer Science, MIT, Cambridge, Massachusetts, 2002

[38] Paper, A. Datta, “MobiGrid:Peer to Peer Overlay and Mobile Ad-Hoc Network Rendezvous – a Data Management Perspective”, CAISE 2003, Doctoral Symposium, Österreich, 2003

[39] Paper, A.Datta, M. Hauswirth, K. Aberer, “Beyond ‘web of trust’: Enabling P2P E-commerce”, Proceedings of the IEEE International Conference on E-Commerce, 2003

[40] Paper, A. Singh, L. Liu, “Trust Me: Anonymous Management of Trust Relationships in Decentralized P2P Systems”, Proceedings of the IEEE International Conference on Peer-to-Peer-Computing, 2003

[41] Script, C.Schindelhauer, “Algorithmen für Peer-to-Peer-Netzwerke”, ”9. Vorlesung”, Paderborn, 2004

[42] Paper, X. Wang, Y. Zhang, X. Li, D. Loguinov, “On ZoneBalancing of Peer-to-Peer Networks: Analysis of Random Node Join”, Sigmetrics/Performance, New York, Juni 2004

[43] Paper, D. Loguinov, A. Kumar, V. Rai, S. Ganesh, “Graph-Theoretic Analysis of Structured Peer to Peer Systems: Routing Distances and Fault Resilience”, Sigcomm, Karlsruhe, August 2003

[44] Paper, K. Aberer, P. Cudré-Mauroux, A. Datta, et al., “P-Grid: A Self-organizing Structured P2P System”, Lausanne, 2003

[45] Technical Report, A. Datta, S. Girdzijauskas, K. Aberer,” On De-Bruijn routing in distributed hash tables: There and back again”, EPFL Technical Report, IC/2004/41, Lausanne, 2004

[46] Paper, K. Aberer,”P-Grid: A Self-organizing Access Structure for P2P Information Systems”,6th Int. Conference on Coorperative Information Systems, Trenta, Italien, Sept 2001

[47] Paper, Stefan Savage, Neal Cardwell, David Wetherall, Tom Anderson, “TCP Congestion Control with a Misbehaving Receiver”, ACM SIGCOMM Computer Communication Review”, Volumen 29 , Ausgabe 5, Oktober 1999

[48] Paper, A. Gupta, B. Liskov, and R. Rodrigues. “One hop lookups for peer-to-peer overlays” In Proc. of the Ninth Workshop on Hot Topics in Operating Systems (HotOS-IX), Lihue, HI, Mai 2003.

[49] Webpage, Rowland Baker, ”Internet Security for Microsoft Windows Using Broadband Connections”, 2000, Webpage Available from: http://www.broadband-help.com/cm_security.asp?pg=4, Accessed on: 31.11.2004

[50] Webpage, Anonymer Autor der Hackergruppe “Port 7 Alliance”, “Why do hackers hack”, Available from : http://www.port7alliance.com/whyhack.html Accessed on: 30.11.2004

[51] Webpage, Ninja Squirrel, “Computer Hackers are Good People Too!”, Webpage Available from: http://www.textfiles.com/hacking/goodkids.hac Accessed on 30.11.2004

[52] Paper, D. Chaum, “Untraceable electronic mail, return addresses, and digital pseudonyms” Communications of the ACM, 24(2): Seiten84 bis 88, 1981

[53] Buch, Bruce Schneier, ”Applied Cryptography, Second Edition: Protocols, Algorithms, and Source Code in C“, John Wiley & Sons, Inc., ISBN: 0471128457,Publication Date: 01/01/96

[54] Linux man pages: The Linux Dokumentation Project, ”man Pages”, Available from http://www.tldp.org/docs.html#man, Version 1.56-6, 2003

[55] Webpage, Ray Dillinger, “Fair Coin Tosses and Die Rolls”,Cyclopedia Cryptologia, 2001, Available from http://www.disappearing-inc.com/F/faircointoss.html, Accessed on 30.12.2004

[56] Email: Geoff Thorpe, “Re: purify errors in openssl crypto”, 18.Juli 2003, Openssl-Mailinglist “openssl-users@openssl.org”, Available from http://www.mail-archive.com/openssl-users@openssl.org/msg31853.html Accessed on 17.12.2004

[57] Email: Geoff Thorpe, “Re: [openssl.org #521] [PATCH] Avoid uninitialized data in randombuffer”, 13 März 2003, openssl-users@openssl.org”, Available from http://www.mail-archive.com/openssl-dev@openssl.org/ msg15497.html, Accessed on 17.12.2004

[58] Buch, Peter Altken und Bradley L. Jones, Übersetzer Frank Langenau, “C in 21 Tagen”,2000, Markt + Technik Verlag, ISBN 3-8272-5727-1

[59] Webpages, anonyme Autoren, “The GNU C Library Reference Manual”, Edition 0.10, Juli 2001, Free Software Foundation, Inc. Available from: http://www.gnu.org/software/libc/manual/index.html

[60] Webpages, anonyme Autoren, “The Onion-Routing-Homepage”, US Naval Research Lab; Available from : http://www.onion-router.net ; Accessed on 22.01.2005

Anhang A. Abkürzungen

Die Buchstaben, die die Abkürzung bilden, sind in Großbuchstaben geschrieben und werden fett gedruckt. Die Funktion von Paketen u.Ä. wird im Anhang B beschrieben.

AH-X	**A**ccount-**H**olding-knoten-**X**
CH	**C**onnect-**H**ere
DAN	**D**atenbank-**A**ccount-**N**umber
DAO	**D**atenbank-**A**nswer-**O**nline
DAT	**D**atenbank-**A**bbruch-**T**ransaktion
DBO	**D**atenbank-**B**in-**O**nline
DCE	**D**atenbank-**C**heck-**E**ntry
DCO	**D**atenbank-**C**heck-if-**O**nline
DEB	**D**atenbank-**E**xecute-**B**in-online
DEC	**D**atenbank-**E**inreichen-des-**C**hecks
DER	**D**atenbank-**ER**ror
DFC	**D**atenbank-**F**orward-**C**heck-if-online
DGC	**D**atenbank-**G**et-**C**heck
DGI	**D**atenbank-**G**et-**I**nformation
DGK	**D**atenbank-**G**et-**K**onto
DLA	**D**atenbank-**L**okal-**A**nswer-online
DLC	**D**atenbank-**L**okal-verteilen-des-**C**hecks
DLF	**D**atenbank-**L**okal-o**F**fline
DLN	**D**atenbank-**L**okal-**N**eues-konto
DLO	**D**atenbank-**L**okal-verteilen-bin-**O**nline
DLS	**D**atenbank-**L**okal-keep-**S**ynchronised
DNA	**D**atenbank-**N**otification-new-**A**ccount
DNO	**D**atenbank-**N**otification-**O**k
DON	**D**atenbank-**ON**line
DRA	**D**atenbank-**R**equest-**A**ccount
DSA	**D**atenbank-**S**ynchronisation-**A**ccount
DSC	**D**atenbank-**S**end-**C**heck
DSI	**D**atenbank-**S**end-**I**nformation
DSK	**D**atenbank-**S**end-**K**onto
DSR	**D**atenbank-**S**ynchronisation-**R**equest-konto-information
DSU	**D**atenbank-want-to-**SU**rf
KBD	**K**notenmanagement-**B**eweg-**D**ich
KDJ	**K**notenmanagement-**D**o-**J**oin
KJO	**K**notenmanagement-**J**oin-**O**k
KKA	**K**notenmanagement-**Keep-Alive**
KML	**K**notenmanagement-**M**itglied-**L**eaving
KNC	**K**notenmanagement-**N**ew-**C**onnection
KNM	**K**notenmanagement-**N**eues-**M**itglied
KOK	**K**notenmanagement-Keep-Alive-**OK**
KRJ	**K**notenmanagement-**R**equest-**J**oin

KSP	**K**notenmanagement-**SP**lit
KVJ	**K**notenmanagement-**V**erteile-**J**oin
KWJ	**K**notenmanagement-**W**ant-to-**J**oin
KWR	**K**notenmanagement-**WR**ong-checksum
Lv	**L**e**V**el
OK	want-to-connect-empfangen-**OK**
VKA	**V**erbindungsmanagement-**K**eep-**A**live
VNC	**V**erbindungsmanagement-**N**ew-**C**onnection
VNJ	**V**erbindungsmanagement-**N**eighbor-**J**oin
VNL	**V**erbindungsmanagement-**N**eighbor-**L**eaving
VNN	**V**erbindungsmanagement-**N**euer-**N**achbar
VNS	**V**erbindungsmanagement-**N**eighbor-**S**plit
VOK	**V**erbindungsmanagement-Keep-Alive-**OK**
VWS	**V**erbindungsmanagement-**W**rong-check**S**um
WTC	**W**ant-**T**o-**C**onnect

Anhang B. Paket Nachrichten

Nachricht	Wird in dieser Datei bearbeitet (s=senden ,e=empfangen)	Trigger (➔ bedeutet siehe Beschreibung)	Nachfolgenachricht	Gültigkeit (G=Global / L=Lokaler- knoten)	Referenz	Beschreibung
DER	S=* E=anwendung/db.c	D*	-	G	-	Ein unbekannte Kontonummer wurde in einer Anfrage genannt.
DGI	S=E=anwendung/net/db_dgi.inc	➔	DSI	L	III.4.2.2	Fordert eine Kopie eines bestimmten Kontos an, um die Datenbank zu vervollständigen. Dementsprechend wird die Nachricht immer versandt, wenn zu einem Konto die Daten fehlen.
DSI	S=E=anwendung/net/db_dgi.inc	DGI	DSI	L	III.4.2.2	Schickt eine Kopie eines Kontos, um die Datenbank zu vervollständigen.
DLS	S=E=anwendung/net/db_lokal_ka.inc	Zeit	-	L	III.4.2.2	Hält synchronisiert die Konten innerhalb eines Knotens.
DCE	S= anwendung/db_startup.inc E=anwendung/db_binonline.inc	Zeit	DON/ DEB	G	III.4.2.5	Lässt überprüfen ob das genannte Konto online ist oder nicht.
DON	S=E=anwendung/db_binonline.inc	DCE	-	G	III.4.2.5	Enthält die Nachricht, dass ein best. Konto als online gemeldet ist.

DEB	S=E=anwendung/db_binonline.inc	DCE	DBO	G	III.4.2.5	Enthält die Nachricht, dass ein bestimmtes Konto nicht als online gemeldet ist.
DBO	S=E=anwendung/db_binonline.inc	DEB	DLO	G	III.4.2.5	Meldet das Konto online an und ordnet diesem eine IP zu.
DLO	S=E=anwendung/db_binonline.inc	DBO	-	L	III.4.2.5	Verteilt die Nachricht in einem Knoten, dass ein Konto online gemeldet wurde.
DRA	S= anwendung/db_startup.inc E=anwendung/db_neuaccount.inc	➔	DANN	G	III.4.2.4	Wenn jemand kein Konto besitzt, so beantragt er hiermit ein Konto
DNA	S=E=anwendung/db_neuaccount.inc	DRA	DNO	G	III.4.2.4	Benachrichtigt die AHx (x>0) über das Anlegen eines Kontos
DNO	S=E=anwendung/db_neuaccount.inc	DNA	DAN+DLN	G	III.4.2.4	Das Konto wurde im entsprechenden AHx (x>0) angelegt
DAN	S=E=anwendung/db_neuaccount.inc	DNO	-	G	III.4.2.4	Gibt dem Anfrager die Kontonummer zurück
DLN	S=E=anwendung/db_neuaccount.inc	DNO	-	L	III.4.2.4	Verteilt innerhalb eines Knotens die Nachricht ein Konto anzulegen
DFC	S=E=anwendung/db_onlinecheck.inc	Zeit	DCO	G	III.4.2.6	Fordert irgendein Mitglied eines zufälligen Knotens auf eine Aufforderung per TCP/IP zu verschicken sich zurückzumelden
DCO	S=E=anwendung/db_onlinecheck.inc	DFC	DAO	G	III.4.2.6	Eine Aufforderung sich bei einem best. AH zurückzumelden
DAO	S=E=anwendung/db_onlinecheck.inc	DCO	DLF	G	III.4.2.6	Rückmeldung eines Kontoinhabers bei einem AH
DLF	S=E=anwendung/db_onlinecheck.inc	➔	-	L	III.4.2.6	Knoteninterne Verteilung der Information, dass jemand nicht auf eine Aufforderung zum Zurückmelden reagiert hat der zukünftig als offline geführt wird
DLA	S=E=anwendung/db_onlinecheck.inc	DAO	-	L	III.4.2.6	Jemand hat sich zurückgemeldet und ist weiterhin online
DSR	S=E=anwendung/db_ah_check.inc	Zeit	DSA	G	III.4.2.7	Aufforderung die Daten für eine Synchronisation der AHs für ein bestimmtes Konto zu schicken
DSA	S=E=anwendung/db_ah_check.inc	DSR	-	G	III.4.2.7	Die Daten eines Kontos zur Synchronisation der AHs
DSU	S=client.c E=anwendung/db_abbuchen.inc	➔	DGK	G	III.4.2.3	Ein Kontoinhaber möchte über jemanden anderes surfen und signalisiert dieses diesem

DGK	anwendung/db_abbuchen.inc	DSU	DSK	G	III.4.2.3	Anfrage nach dem Kontostand von jemandem der surfen will
DSK	anwendung/db_abbuchen.inc	DGK	DAT/DGC	G	III.4.2.3	Der Kontostand von jemandem der surfen will
DAT	S=anwendung/db_abbuchen.inc E=client.c	DSK	-	G	III.4.2.3	Es wird aufgrund zu geringem Kontostand jemanden verweigert zu surfen.
DGC	S=anwendung/db_abbuchen.inc E=client.c	DSK	DSC	G	III.4.2.3	Der Kontostand reicht zum surfen aus, Anforderung der Abbuchung
DSC	S=client.c E=anwendung/db_abbuchen.inc	DGC	DEC	G	III.4.2.3	Schicken einer Abbuchungsanweisung an jemanden, über den man surfen will.
DEC	S=E=anwendung/db_abbuchen.inc	DSC	DLC	G	III.4.2.3	Einreichen einer Abbuchungsanweisung bei den AHs
DLC	S=E=anwendung/db_abbuchen.inc	DEC	-	L		Verteilen einer Abbuchungsanweisung im Knoten
WTC	S=E=bp/bp.c	➔	OK+CH	G	III.4.1.1	Jemand will sich neu mit dem Peer-to-Peer-Netz verbinden
OK	S=E=bp/bp.c	WTC	-		III.4.1.1	Gibt dem Anfrager Bescheid, dass eine WTC-Anfrage weitergeleitet wird.
CH	S=E=bp/bp.c	WTC	-	G	III.4.1.1	Enthält die Daten und eine Aufforderung sich in einen Knoten zu integrieren.
KNC	S=kmp/kmp_p.c E=kmp/kmp_neu.inc	➔	-	L	-	Identifiziert eine neue TCP/IP Verbindung als zu KMP gehörig
KNM	S=E=kmp/kmp_neu.inc	➔	VNN	L	III.4.1.2.1	Bekanntmachung der IP eines neuen Mitgliedes
KML	S=E=kmp/kmp_leave.inc	➔	VNL	L	III.4.1.2.1	Das Mitglied mit einer best. IP verlässt den Knoten
KKA	S=E=kmp/kmp_Keep-Alive.inc	Zeit	KOK/KWR	L	III.4.1.2.2	Nachricht zur Synchronisation der Mitgliederlisten innerhalb eines Knotens und Prüfung der Verbindungen
KOK	S=E=kmp/kmp_Keep-Alive.inc	KKA	-	L	III.4.1.2.2	Die Mitgliederlisten sind synchron
KWR	S=E=kmp/kmp_Keep-Alive.inc	KKA	-	L	III.4.1.2.2	Es wurde eine Inkonsistenz der Mitgliederlisten festgestellt, die Daten um diese zu lösen.
KSP	S=E=kmp/kmp_split.inc	➔	VNS	L	III.4.1.2.3	Aufforderung zur Ausführung einer Knoten-Teilung wegen zu vieler Mitglieder im Knoten
KWJ	S=E=kmp/kmp_join.inc	➔	KJO/KBD	G	III.4.1.2.4	Anfrage nach Unterstützung an den Partnerknoten wegen einer zu geringen Mitgliederzahl

KJO	S=E=kmp/kmp_join.inc	KWJ	KRJ	G	III.4.1.2.4	Eine Zusammenlegung der Partnerknoten wird vorgeschlagen als Antwort auf KWJ
KRJ	S=E=kmp/kmp_join.inc	KJO	KVJ	L	III.4.1.2.4	Fordert die Mitgliederliste des Partnerknotens an
KVJ	S=E=kmp/kmp_join.inc	KRJ	KDJ	L	III.4.1.2.4	Schickt die Mitgliederliste des Partnerknotens
KDJ	S=E=kmp/kmp_join.inc	KVJ KRJ	VNJ	L	III.4.1.2.4	Fordert zu einer Zusammenlegung entsprechend dem Inhaltes des Paketes auf
KBD	S=E=kmp/kmp_join.inc	KWJ	-	L	III.4.1.2.4	Fordert jemanden auf die Bootprocedure erneut zu starten um dem Partnerknoten zu helfen.
VNC	S=vmp/vmp_p.c E=vmp/vmp.c	➔	-	G		Identifizierung einer neuen TCP/IP Verbindung als zu VMP gehörig
VNN	S=E=vmp/vmp_neu.inc	KNM	-	G	III.4.1.3.2	Ein neues Mitglied ist im Nachbarknoten hinzugekommen
VNL	S=E=vmp/vmp_leave.inc	KML	-	G	III.4.1.3.2	Ein Mitglied hat den Nachbarknoten verlassen
VKA	S=E=vmp/vmp_Keep-Alive.inc	Zeit	VOK/VWS	G	III.4.1.3.2	Nachricht zur Konsistenthaltung der Repräsentationen von Knoten auf Nachbarknoten und Prüfung der Verbindungen
VOK	S=E=vmp/vmp_Keep-Alive.inc	VKA	-	G	III.4.1.3.2	Die Repräsentation eines Nachbarn ist korrekt und die Verbindung funktioniert.
VWS	S=E=vmp/vmp_Keep-Alive.inc	VKA	-	G	III.4.1.3.2	Die Repräsentation eines Nachbarn ist inkorrekt (inkl. Daten zur Korrektur) aber die Verbindung funktioniert
VNS	S=E=vmp/vmp_split.inc	KSP	-	G	III.4.1.3.2	Der Nachbarknoten hat sich geteilt
VNJ	S=E=vmp/vmp_join.inc	KDJ	-	G	III.4.1.3.2	Der Nachbarknoten hat sich mit seinem Partner zusammengelegt.

Routing:

Hier werden die Paket-Optionen beschrieben, nicht die Arten wie man die Routingfunktion aufruft, hierzu bitte die Kommentare bei rp/rp.c#rp_route lesen!

Wenn die Ziel-Verbindung im Paketkopf auf Null gesetzt ist, so handelt es sich nur um Knoten präzises Routing, wenn sie ungleich Null ist, so handelt es sich um Computer präzises Routing.

Befehlsnummer		Beschreibung
1	RP_LOKAL	Dieses Packet war Computer präzise und wurde schon an alle Mitglieder im Knoten verteilt
2	RP_ANTWORT	Auf dieses Packet kann geantwortet werden
3	RP_ANTWORT | RP_LOKAL	Sowohl 1 als auch 2
4	RP_MIRROR	Anlegen einer Umleitung für Antwortpakete und Weiterleiten des Paketes

Anhang C. Kenndaten bei Belastungstests des Netzwerkes

Eine Diskussion der Zahlen und ihrer Aussagen findet man unter IV.2.1.

Versuch	Ausfalls-verzögerung	#Instanzen: Maximum	#Instanzen: Minimum	Dauer in h:min	fehlerhafte Einwahl	Mobbing	keine Synchro-nisation in %	verwaiste Konten	verlorene Routing Pakete in %	Bewegungen	Unebenheiten ➔WTJ	Joins	Splits
Op1	1600	2173 Lv7	103 Lv3	5:49	3	1	0,029	1	0,014	1165	54	121	127
Op2	1600	2203 Lv7	129 Lv3&4	6:24	0	1	0,023	3	0,064	1097	25	116	127
Op3	1600	2182 Lv7	110 Lv3	5:05	0	3	0,025	3	0,062	1114	33	120	127
Op4	0	1624 Lv7	99 Lv3	5:49	6	9	0,053	2	0,053	1025	121	121	128
Op5*[1]	0	879 Lv6	118 Lv3&4	5:48	88*[2]	0	0,030	4	0,026	898	16	53	63
PC1	0	321 Lv5	13 Lv0	3:20	12	21	0,166	2	0,021	177	17	31	31
PC2*[3]	0	347 Lv5	12 Lv0	3:15	2	1	0,086	0	0,048	184	10	32	31
PC3*[3]	0	372 Lv5	11 Lv0	3:03	2	9	0,003 *[4]	1	0	169	16	32	31

Tabelle 8: Vollständige Übersicht der Kennzahlen bei Belastungstestläufen des Netzwerkes.

*[1] Ausfallrate $=12*10^{-3}$

*[2] Ausgelöst durch eine Bewegungen einer Eintrittspforte während des Aufbaus

*[3] Knoten-Management-Protokoll-Timeout =30 Sekunden

*[4] zusätzlich traten 140 Synchronisationsfehler bei einer Instanz auf, wobei diese Instanz darauf die Verbindung zum Netz (vollständig) verloren hatte. (Knoten verschwunden).

Anhang D. CD mit Sourcecode

Auf der beiliegenden CD ist eine elektronische Version dieser Arbeit enthalten. Zusätzlich sind noch die digital vorliegenden Referenzen beigelegt. Der kommentierte Sourcecode vervollständigt die Daten. Die Readme-Datei ist als Startpunkt vorgesehen.